

## Multi-modal Spatial Crowdsourcing for Enriching Spatial Datasets

Gummidi, Srinivasa Raghavendra Bhuvan

DOI (link to publication from Publisher):  
[10.54337/aau429768599](https://doi.org/10.54337/aau429768599)

Publication date:  
2021

Document Version  
Publisher's PDF, also known as Version of record

[Link to publication from Aalborg University](#)

Citation for published version (APA):  
Gummidi, S. R. B. (2021). *Multi-modal Spatial Crowdsourcing for Enriching Spatial Datasets*. Aalborg Universitetsforlag. <https://doi.org/10.54337/aau429768599>

### General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal -

### Take down policy

If you believe that this document breaches copyright please contact us at [vbn@aub.aau.dk](mailto:vbn@aub.aau.dk) providing details, and we will remove access to the work immediately and investigate your claim.



# **MULTI-MODAL SPATIAL CROWDSOURCING FOR ENRICHING SPATIAL DATASETS**

**BY  
SRINIVASA RAGHAVENDRA BHUVAN GUMMIDI**

DISSERTATION SUBMITTED 2021



**AALBORG UNIVERSITY**  
DENMARK



---

# Multi-modal Spatial Crowdsourcing for Enriching Spatial Datasets

---

Ph.D. Dissertation  
Srinivasa Raghavendra Bhuvan Gummidi

Dissertation submitted February, 2021

A thesis submitted to the Technical Faculty of IT and Design at Aalborg University (AAU) and the Department of Computer and Decision Engineering at Université Libre de Bruxelles (ULB), in partial fulfillment of the requirements within the scope of the IT4BI-DC programme for the joint Ph.D. degree in Computer Science. The thesis is not submitted to any other organization at the same time.

Dissertation submitted: February 24, 2021

PhD supervisor: Prof. Torben Bach Pedersen  
Aalborg University, Denmark

AAU PhD Co-Supervisor: Prof. Xike Xie  
University of Science and Technology of China, China

ULB PhD Supervisor: Prof. Esteban Zimányi  
Universite Libre de Bruxelles, Belgium

AAU PhD Committee: Associate Professor Chenjuan Guo (chair)  
Aalborg University  
Associate Professor Michela Bertolotto  
University College Dublin  
Professor Xiaofeng Gao  
Shanghai Jiao Tong University

ULB PhD Committee: Prof. Dimitris Sacharidis,  
Universite Libre de Bruxelles  
Assoc. Prof. Michela Bertolotto,  
University College Dublin  
Prof. Xiaofeng Gao  
Shanghai Jiao Tong University, China

PhD Series: Technical Faculty of IT and Design, Aalborg University

Department: Department of Computer Science

ISSN (online): 2446-1628  
ISBN (online): 978-87-7210-901-5

Published by:  
Aalborg University Press  
Kroghstræde 3  
DK – 9220 Aalborg Ø  
Phone: +45 99407140  
aauf@forlag.aau.dk  
forlag.aau.dk

© Copyright: Copyright by Srinivasa Raghavendra Bhuvan Gummid. Author has obtained the right to include the published and accepted articles in the thesis, with a condition that they are cited, DOI pointers and/or copyright/credits are placed prominently in the references.

Printed in Denmark by Rosendahls, 2021

# Abstract

Despite the advances in technology, many tasks are not effectively resolved by computer-based automated algorithms. These tasks range from image recognition to entity resolution and require the human cognitive ability to improve the algorithms' performance. Crowdsourcing helps resolve such tasks by providing a platform to engage ordinary workers (crowd) to harness their capabilities. Generally, conventional crowdsourcing applications focuses on tasks that workers can resolve via the internet. However, tasks related to real-world scenarios with spatial aspects like traffic information and environmental data collection cannot be resolved virtually and require physical on-location operations. Spatial crowdsourcing (SC), a particular class of crowdsourcing, provides a platform to solve such spatial nature tasks by harnessing the crowd's potential. The widespread availability of location empowered smartphones has boosted SC's application potential ranging from smartphone sensor-based data collection to personal service-based delivery applications. A typical SC application operates on different modes depending on the requirements of the spatial tasks, workers, and task requesters. The different modes can be related to, but not limited to, task complexity, the required number of responses, type of assignment/scheduling problem addressed, type of constraints considered, and task publishing mode. When an SC application operates on multiple modes, this thesis defines such SC application as a Multi-modal SC application.

Given the significance of spatial datasets like OpenStreetMap (OSM) in research, this thesis explores multi-modal SC's potential to address the problems faced by spatial datasets regarding the coverage/ sparsity and quality. The objective is to enrich the spatial datasets like OSM and improve their spatial entities attribute/ tag coverage and quality by utilizing multi-modal SC. This thesis also enhances spatial entities attributes by linking spatial entities across multiple spatial datasets by employing a machine learning-based multi-modal SC application. Furthermore, this thesis encourages worker participation in multi-modal SC by customizing task assignment and scheduling strategies for a potential group of new SC workers, public transportation users, to enrich spatial datasets. The following paragraphs highlight the con-

tributions of this thesis.

Firstly, this thesis provides a comprehensive literature survey of multi-modal SC. The survey describes the SC usage workflow and identifies the fundamental difference between spatial and conventional crowdsourcing. The survey introduces a taxonomy to classify the existing SC literature based on the commonalities and differences. Furthermore, the survey elucidates the current literature's issues/challenges and presents some potential research directions for the future. This thesis addresses three challenges of SC highlighted by the survey concerning enrichment of spatial datasets, improvement of worker participation in multi-modal SC campaigns, and improvement of task assignment and scheduling strategies. The survey benefits the researchers in gaining a comprehensive view of the existing research in SC and facilitating the comparison of different task assignment and scheduling algorithms by highlighting their advantages and drawbacks.

Secondly, this thesis addresses the challenge of enriching semantic tag information of spatial entities in the OSM dataset and improving worker participation in OSM. This thesis aims to improve the quantity and quality of the tags associated with the OSM's spatial entities through the multi-modal SC approach in push-based/server-assignment mode. The thesis defines different task assignment problems for maximizing 1) the total number of task assignments, 2) the entities' coverage, 3) the number of verifiable task assignments. Focusing on the use case of road networks in OSM, this thesis proposes an integrated framework to extract the tasks and assign road segments/ junctions to workers through algorithms based on different constraints in offline and batch-based worker input model scenarios. An experimental evaluation reveals that the proposed junctions-based algorithms result in five times as many unique assignments and seven times as many verifiable assignments as the baseline max-flow based algorithm and around half the average distance travelled per task than the baseline algorithm.

Thirdly, this thesis addresses the challenge of enriching spatial datasets by linking spatial entities across multiple spatial datasets like Google Places, Flickr, and Krak. Linking spatial entities has a vast potential to offer rich attribute information regarding the spatial entities. However, linking the spatial entities from different sources involves finding out whether they represent the same physical entity or not, leading to a spatial entity linkage (SEL) problem. This thesis aims to resolve the SEL problem by exploiting the wisdom of SC workers. Given the reward budget limitations, this thesis proposes a hybrid Skycrowd solution incorporating machine learning-based SC and automatic labeling (AL) techniques. The proposed Skycrowd solution performs 30% better than the automatic labeling of the entire SEL tasks set with respect to F-measure. It achieves an F-measure value of 0.91 by spending a fraction (7%) of the reward budget required for crowdsourcing the entire set of SEL tasks.



Finally, this thesis addresses the challenge of improving worker participation and improving the task assignment and scheduling algorithms by considering the worker movement information. This thesis considers a new group of workers travelling via public transportation to improve online task assignment and task scheduling strategies by incorporating transit route information. This thesis assumes that the new group of workers can perform assigned tasks at transit stops on their route while waiting to change for the next bus/ train. This thesis defines the Transit-based Task Assignment (TTA) problem to maximize the average worker rewards considering different constraints like worker transit route (WTR) adherence and task deadlines. Furthermore, this thesis defines a credibility-based TTA problem to ensure quality crowdsourced responses. Moreover, this thesis also defines a flexible-TTA problem assuming a threshold reward to convince a worker to stay longer at a transit stop for a high reward task, thereby relaxing the WTR model's rigid nature. This thesis proposes algorithms to solve the three different problems and performed an extensive evaluation of the implemented algorithms. The FlexibleTTA problem algorithm outperforms other algorithms by 55% regarding the number of assigned tasks and at least 35% regarding the worker's average reward.

In conclusion, this thesis performs a comprehensive survey of the existing SC literature and proposes a framework comprising novel multi-modal SC applications for enriching spatial datasets. In addition to addressing the spatial dataset enrichment issue, the proposed applications address other SC issues by suggesting strategies to improve worker participation and improving task assignment/ scheduling algorithms. As part of future work, this thesis suggests implementing a comprehensive multi-modal SC solution to enrich the OSM dataset, integrating the proposed individual multi-modal SC applications regarding OSM tag enrichment, spatial entity linkage, and transit-based task assignment.



# Resumé

På trods af teknologiske fremskridt løses mange opgaver ikke effektivt af computerbaserede automatiserede algoritmer. Disse opgaver spænder fra billedgenkendelse til enhedssafklaring og kræver den menneskelige kognitive evne til at forbedre algoritmernes præstation. Crowdsourcing hjælper med at løse sådanne opgaver ved at tilbyde en platform til at engagere almindelige brugere (crowd) til at udnytte deres evner. Generelt fokuserer konventionelle crowdsourcing-applikationer på opgaver, som brugere kan løse via internettet. Opgaver relateret til virkelige scenarier med geografiske aspekter som trafikinformation og indsamling af miljødata kan dog ikke løses virtuelt og kræver fysiske handlinger på lokationen. Spatial Crowdsourcing (SC), en særlig klasse af Crowdsourcing, leverer en platform til at løse sådanne geografiske opgaver ved at udnytte brugernes potentiale. Den udbredte tilgængelighed af lokationsbestemte smartphones har øget SCs applikationspotentiale lige fra smartphone-sensorbaseret dataindsamling til personlige servicebaserede leveringsapplikationer. En typisk SC-applikation fungerer i forskellige tilstande afhængigt af kravene til de geografiske opgaver, brugere og opgavestillere. De forskellige tilstande kan være relateret til, men ikke begrænset til, opgavekompleksitet, det ønskede antal svar, type af tildelings- og planlægningsproblemer der behandles, type af begrænsninger, der overvejes, og opgavepublikationstilstand. Når en SC-applikation fungerer i flere tilstande, definerer denne afhandling sådanne SC-applikationer som en multimodal SC-applikation.

I betragtning af betydningen af geografiske datasæt som OpenStreetMap (OSM) i forskning, undersøger denne afhandling multimodal SC's potentiale til at løse de problemer, som geografiske datasæt står over for med hensyn til dækning/ sparsomhed og kvalitet. Målet er at forbedre de geografiske datasæt som f.eks. OSM og forbedre deres geografiske enhedsattributter og kvalitet ved at bruge multimodal SC. Denne afhandling forbedrer også attributter for geografiske enheder ved at forbinde geografiske enheder på tværs af flere geografiske datasæt ved at anvende en maskinlæringsassisteret multimodal SC-applikation. Desuden lægger denne afhandling op til brugerdeltagelse i multimodal SC ved at tilpasse opgavetildeling og plan-

lægningsstrategier for en potentiel gruppe af nye SC-brugere, brugere af offentlig transport, til at forbedre de geografiske datasæt. De følgende afsnit fremhæver bidragene fra denne afhandling.

For det første giver denne afhandling en omfattende litteraturundersøgelse af multimodal SC. Undersøgelsen beskriver SC-brugerworkflow og identificerer den grundlæggende forskel mellem spatial og konventionel crowdsourcing. Undersøgelsen introducerer en taksonomi for at klassificere den eksisterende SC-litteratur baseret på ligheder og forskelle. Endvidere belyser undersøgelsen den aktuelle litteraturs problemer/ udfordringer og præsenterer nogle potentielle retninger for fremtidig forskning. Denne afhandling behandler tre udfordringer fra SC fremhævet af undersøgelsen vedrørende forbedring af geografiske datasæt, forbedring af brugerdeltagelse i multimodale SC-kampagner og forbedring af opgavetildeling og planlægningsstrategier. Undersøgelsen gavner forskerne ved at få et samlet overblik over den eksisterende forskning i SC og lette sammenligningen af forskellige opgavetildelings og planlægningsalgoritmer ved at fremhæve deres fordele og ulemper.

For det andet behandler denne afhandling udfordringen med at forbedre semantiske annoteringsoplysninger om geografiske enheder i OSM-datasættet og forbedre brugerdeltagelse i OSM. Denne afhandling har til formål at forbedre kvantiteten og kvaliteten af de tags, der er knyttet til OSMs geografiske enheder gennem den multimodale SC-tilgang der kaldes push-baseret/ server-tildelingstilstand. Afhandlingen definerer forskellige opgavetildelingsproblemer til maksimering af 1) det samlede antal opgavetildelinger, 2) enhedernes dækning, 3) antallet af verificerbare opgavetildelinger. Denne afhandling fokuserer på brugen af vejnetværk i OSM og foreslår en integreret ramme til at udtrække opgaverne og tildele vejsegmenter/ vejkrøds til brugere gennem algoritmer baseret på forskellige begrænsninger i offline og batchbaserede brugerinput-scenarier. En eksperimentel evaluering afslører, at de foreslåede krydsbaserede algoritmer resulterer i fem gange så mange unikke opgaver og syv gange så mange verificerbare opgaver som baseline max-flow-baserede algoritmer og omkring halvdelen af den gennemsnitlige tilbagelagte afstand pr. opgave end baseline-algoritmen.

For det tredje adresserer denne afhandling udfordringen med at forbedre geografiske datasæt ved at forbinde geografiske enheder på tværs af flere geografiske datasæt som Google Places, Flickr og Krak. Sammenkædning af geografiske enheder har et stort potentiale til at tilbyde righoldige attributoplysninger om de geografiske enheder. At kæde de geografiske enheder fra forskellige kilder indebærer imidlertid at finde ud af, om de repræsenterer den samme fysiske enhed eller ej, hvilket fører til et problem med spatial enhedslinkning (SEL). Denne afhandling har til formål at løse SEL-problemet ved at udnytte den viden, som SC-brugere har. I betragtning af begrænsningerne i belønningsbudgettet foreslår denne afhandling en hybrid

Skycrowd-løsning, der indeholder maskinlæringsassisteret SC og automatiske mærkning (AL). Den foreslåede Skycrowd-løsning fungerer 30% bedre end den automatisk mærkning af hele SEL-opgaver, der er indstillet med hensyn til F-measure. Den opnår en F-measure-værdi på 0,91 ved at bruge en brøkdel (7%) af det belønningsbudget, der kræves til crowdsourcing af hele SEL-opgaverne.

Endelig behandler denne afhandling udfordringen med at forbedre brugerdeltagelse og forbedre opgavetildeling og planlægningsalgoritmer ved at tage højde for oplysninger om brugerbevægelse. Denne afhandling overvejer en ny gruppe brugere, der rejser via offentlig transport for at forbedre online opgavetildeling og opgaveplanlægningsstrategier ved at inkorporere oplysninger om ruteinformationer. Denne afhandling forudsætter, at den nye gruppe af brugere kan udføre tildelte opgaver ved ophold på deres rute, mens de venter på at skifte til næste bus/ tog. Denne afhandling definerer det rejsebaserede opgavetildelings(TTA)-problem for at maksimere de gennemsnitlige brugerbelønninger i betragtning af forskellige begrænsninger som overholdelse af transportrute (WTR) og deadlines for opgaver. Derudover definerer denne afhandling et troværdighedsbaseret TTA-problem for at sikre kvalitet i crowdsource-svarene. Desuden definerer denne afhandling et fleksibelt TTA-problem, under forudsætning af en tærskelbelønning, for at overbevise en bruger om, at blive længere ved et ophold, for en opgave med høj belønning, og derved lempelse af WTR-modellens stive natur. Denne afhandling foreslår algoritmer til løsning af de tre forskellige problemer og beskriver udførelsen af en omfattende evaluering af de implementerede algoritmer. Fleksibel TTA-problemalgoritmen overgår andre algoritmer med 55% med hensyn til antallet af tildelte opgaver og mindst 35% med hensyn til brugernes gennemsnitlige belønning.

Sammenfattende giver denne afhandling en omfattende oversigt over den eksisterende SC-litteratur og foreslår en konstruktion, der omfatter nye multimodale SC-applikationer til forbedring af geografiske datasæt. Ud over at løse problemet med forbedring af geografiske datasæt, adresserer de foreslåede applikationer andre SC-problemer ved at foreslå strategier til forbedring af brugerdeltagelse og forbedring af opgavetildeling og planlægningsalgoritmer. Som en del af det fremtidige arbejde foreslår denne afhandling implementering af en omfattende multimodal SC-løsning til forbedring af OSM-datasættet, integrering af de foreslåede individuelle multimodale SC-applikationer vedrørende OSM-tagforbedring, geografisk enhedstilknytning og rejsebaseret opgavetildeling.



# Résumé

Malgré les progrès technologiques, beaucoup de problèmes ne sont toujours pas solvables algorithmiquement. Ces problèmes vont de la reconnaissance d'images à la résolution d'entités, et nécessitent les capacités de raisonnement de l'humain pour améliorer les performances des algorithmes. Le Crowdsourcing aide à résoudre ces problèmes en fournissant une plateforme où il est possible de mettre à contribution des utilisateurs ordinaires (crowd) et utiliser leurs compétences pour résoudre des problèmes. En général, le Crowdsourcing se concentre sur des tâches solvables via internet, ce qui n'est pas adapté à certaines tâches localisées comme l'analyse du trafic automobile ou l'environnement, qui nécessitent une présence physique. Le Crowdsourcing spatial (SC), est une forme particulière de Crowdsourcing qui fournit les outils nécessaires à la résolution de ce type de problèmes. La grande disponibilité de téléphones avec géolocalisation a permis une plus grande utilisation de systèmes de SC. En général, les applications de SC opèrent avec différents modes selon les tâches à accomplir. Ces modes peuvent être liés à la complexité de la tâche, les entrées attendues de l'utilisateur, l'allocation des tâches, diverses autres contraintes, et la publication de ces tâches elles-mêmes. Quand une application de SC utilise plusieurs modes, il s'agit d'une application multimodale.

Etant donné l'importance de jeux de données comme OpenStreetMap (OSM) dans la recherche académique, cette thèse explore les problèmes liés à l'utilisation de jeux de données spatiaux, en particulier les problèmes de qualité et de couverture des données. L'objectif étant d'enrichir ces données et d'améliorer leurs attributs spatiaux en les utilisant avec du SC multimodal. Cette thèse améliore aussi ces entités spatiales en les combinant à travers plusieurs jeux de données spatiaux grâce à un modèle d'apprentissage automatique multimodal. De plus, cette thèse propose d'améliorer l'implication des utilisateurs dans les systèmes SC multimodaux grâce à une optimisation intelligente de l'assignement des tâches pour les nouveaux utilisateurs. Les paragraphes suivants résumeront les contributions scientifiques de cette thèse.

Tout d'abord, cette thèse propose une revue de l'état de l'art exhaustive

des systèmes de SC multimodaux. Cette revue s'attache à décrire les utilisations et différences entre les systèmes de SC traditionnels et multimodaux. Cette revue de l'état de l'art inclue une taxonomie originale pour classifier les systèmes existant selon leurs caractéristiques. De plus, les problèmes et limitations des systèmes existant, ainsi que les pistes de recherche future, sont couverts par cette revue de l'état de l'art. Cette thèse résout trois problèmes identifiés dans l'état de l'art concernant l'enrichissement des données spatiales, la participation des utilisateurs, ainsi que les stratégies d'allocation et d'ordonnancement de ces tâches. Cette revue de l'état de l'art permet aux chercheurs du domaine d'avoir une vue d'ensemble de l'état actuel de la recherche dans le domaine du SC et facilite la comparaison entre différentes approches d'allocation et d'ordonnancement des tâches de SC.

En second lieu, cette thèse résout les défis liés à l'enrichissement des tags sémantiques des entités spatiales dans le jeu de données OSM, ainsi que la participation des utilisateurs à OSM. L'objectif de cette thèse est d'améliorer à la fois la quantité et la qualité des tags associés aux entités spatiales de OSM à travers l'approche multimodale du SC via un mode "push-based/server-assignment". Cette thèse définit plusieurs problèmes d'allocation des tâches visant à maximiser 1) le nombre total de tâches allouées 2) la couverture des entités 3) le nombre de tâches vérifiables allouées. En se concentrant sur le réseau routier d'OSM, cette thèse propose un Framework pour extraire les tâches d'attribution de segments routier/jonctions aux utilisateurs. Ce Framework fonctionne grâce à des algorithmes contraints par des modèles de travailleurs offline et des entrées par batch. L'évaluation expérimentale montre que les algorithmes basés sur les jonctions produisent cinq fois plus d'allocation de tâches uniques, et sept fois plus d'allocation de tâches vérifiables que l'algorithme de référence utilisant une approche "max-flow", tout en réduisant de moitié la distance moyenne parcourue par tâche.

Troisièmement, cette thèse résout les défis liés à l'enrichissement de jeu de données spatiaux en liant les entités spatiales entre plusieurs jeux de données spatiaux comme Google Places, Flickr, et Krak. Lier des entités spatiales à le potentiel de fournir des informations riches sur celles-ci. Cependant, lier les entités depuis plusieurs sources implique de trouver si elles représentent réellement la même entité physique, ce qui est un problème de "spatial entity linkage" (SEL). Cette thèse cherche à résoudre le problème SEL par l'utilisation des connaissances des utilisateurs de systèmes de crowdsourcing spatiaux. Étant donné les limitations de budgets de récompense, cette thèse propose une solution hybride intitulée Skycrowd employant des techniques de machine-learning pour le crowdsourcing spatial, ainsi que des techniques d'étiquetage automatique. Skycrowd surpasse de 30% les stratégies d'étiquetage automatique de l'entièreté des tâches SEL vis à vis de la F-mesure. L'algorithme obtient une F-mesure de 0.91 tout en ne dépensant qu'une fraction (7%) du budget de récompense nécessaire à la complétion de



l'entièreté des tâches SEL.

Finalement, cette thèse s'occupe du problème de l'amélioration de la participation des utilisateurs, l'amélioration de l'allocation des tâches, ainsi que l'ordonnancement de ces tâches, en prenant en compte les données de déplacement des utilisateurs. Cette thèse s'intéresse à un nouveau groupe d'utilisateurs voyageant dans les transports publics, dans le but d'améliorer l'allocation en ligne des tâches, ainsi que les stratégies d'ordonnancement incorporant les itinéraires. Cette thèse part du principe que les nouveaux groupes d'utilisateurs peuvent accomplir les tâches qui leur sont assignées aux arrêts pendant qu'ils attendent leur prochain bus/train. Cette thèse définit le problème de "Transit-based Task Assignment" (TTA) pour maximiser la récompense utilisateur moyenne tout en prenant en compte diverses contraintes liées à l'itinéraire des utilisateurs (worker transit route, WTR), ainsi que les deadlines des tâches. De plus, cette thèse définit un problème TTA basé sur la crédibilité garantissant la qualité des réponses crowdsourcés. Cette thèse définit aussi un problème TTA flexible encourageant les utilisateurs à rester plus longtemps à leurs arrêts grâce à des tâches mieux récompensées, réduisant de fait la rigidité naturelle des modèles WRT. Cette thèse propose des algorithmes pour résoudre ces trois différents problèmes couplés à une évaluation exhaustive de leur implémentation. L'algorithme solvant le problème TTA Flexible surpasse les autres algorithmes de 55% pour les tâches assignées et au moins de 35% pour les récompenses utilisateur moyenne.

En conclusion, cette thèse fournit une revue exhaustive de l'état de l'art du SC et propose un framework implémentant des applications multimodales de crowdsourcing pour enrichir les jeux de données spatiaux. En plus de résoudre le problème d'enrichissement des jeux de données spatiaux, les applications proposées permettent de résoudre d'autres problèmes de SC par la suggestion de stratégies améliorant la participation des utilisateurs ainsi que des algorithmes d'ordonnancement et d'allocation de tâches. Concernant la recherche future, cette thèse suggère l'implémentation d'une solution SC multimodale exhaustive pour enrichir le jeu de données OSM, l'intégration des applications SC multimodale proposées pour enrichir les tags de OSM, lier les entités spatiales, et une allocation des tâches basée sur le transit.



# Acknowledgements

I want to thank the following people for their support and encouragement throughout my PhD journey.

I want to express my sincerest gratitude to my PhD supervisor, Professor Torben Bach Pedersen, a constant support source throughout my PhD journey. He encouraged my efforts, was patient with my shortcomings and delays, guided my research with excellent insights and prompt feedback. He motivated me to regain balance on many occasions and got me on track towards completion. Likewise, I want to express my thanks to my PhD Co-supervisors, Professor Xike Xie and Prof. Esteban Zimányi, for providing helpful feedback and supporting my PhD research.

Further, I want to thank all my colleagues at Aalborg University and Université Libre de Bruxelles. Especially, Nguyen Ho, Suela Isaj, Rudra Pratap Deb Nath, Ilkcan Keles, Søren Jensen, Aamir Saleem, Idris Mim, Bijay Neupane, Rohit Kumar, Olivier Pelgrin, Olga Rybnytska, Simon Pedersen, Lawan Subba, Tobias Jepsen, Robert Waury, Faisal Orakzai, Felipe Costa, Kim Mathiasen, Nurefsan Gur, Mahmoud Sakr, and Long Van Ho for their encouragement and support. My special thanks to Maria Lyngby Karlsen and Kent Høegh Jensen for translating the thesis abstract to danish. Also, thanks to Olivier Pelgrin for translating the thesis abstract to french. I express my appreciation and gratitude to the administrative staff at Aalborg University and Université Libre de Bruxelles. I want to thank Professor Gang Liu, from the University of South Denmark, for his support and words of encouragement during the last phase of my PhD.

I also want to take this opportunity to acknowledge my family's support. My mother, Syamala Devi Gummidi is my biggest source of strength, and she enabled me with her love and affection. Next, my uncle, Sri Hari Avala, deserves thanks for his help in taking care of things back home. Further, I want to thank my family members, Suryakantham Avala, Simhachalam Kongarapu, Padmavathi Kongarapu, Satyakala Eagala, Vijaya Lakshmi Avala, Srinivas Rao Kongarapu Sravani Gorla, Sai Deepak Gorla, Parimala, Ganesh Avala, Divya Avala, Bhagya Lakshmi Wiyyapu, and A Manikanta Swamy.

There were a lot of ups and downs throughout this strenuous PhD journey. Therefore, I have to mention two things that always brought a smile to my face as a token of appreciation. The flawless perfection of Indian Cricket's Superstar Sachin Tendulkar's straight drives and the effortless grace of Indian Movie's Megastar Chiranjeevi Konidela's dance raised my spirits every time.

*Finally, I would like to dedicate this PhD thesis to the loving memory of my father, Madhava Rao Gummidi, and my sister, Sri Devi.*

Srinivasa Raghavendra Bhuvan Gummidi  
Odense, Denmark, February 24, 2021

*This research has been funded by the European Commission through the Erasmus Mundus Joint Doctorate "Information Technologies for Business Intelligence" - Doctoral College (IT4BI-DC).*

# Contents

Abstract	iii
Dansk Résumé	vii
Résumé (Abstract in French)	xi
Acknowledgements	xv
Thesis Details	xxiii

<b>I Thesis Summary</b>	<b>1</b>
-------------------------	----------

Thesis Summary	3
1 Introduction . . . . .	3
1.1 Motivation and Background . . . . .	3
1.2 Thesis Structure . . . . .	7
2 Spatial Crowdsourcing Literature Review . . . . .	9
2.1 Motivation and Problem Statement . . . . .	9
2.2 Spatial Crowdsourcing Workflow . . . . .	9
2.3 Modes in SC . . . . .	12
2.4 Challenges in SC . . . . .	19
2.5 Challenges Addressed in this Thesis . . . . .	20
3 Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap . . . . .	20
3.1 Motivation and Problem Statement . . . . .	20
3.2 Proposed Framework Architecture . . . . .	22
3.3 Task Generator Module . . . . .	23
3.4 Task Assignment Module . . . . .	24
3.5 Quality Control . . . . .	34
3.6 Experimental Evaluation of Assignment Algorithms . .	35
3.7 Discussion . . . . .	36

4	Spatial Entity Linkage with the Aid of Spatial Crowdsourcing .	37
4.1	Motivation and Problem Statement . . . . .	37
4.2	Preliminaries and Problem Definition . . . . .	38
4.3	Skycrowd Solution . . . . .	40
4.4	Experimental Evaluation . . . . .	47
4.5	Discussion . . . . .	50
5	Transit-based Task Assignment in Spatial Crowdsourcing . . .	50
5.1	Motivation and Problem Statement . . . . .	50
5.2	Preliminaries . . . . .	51
5.3	Types of Assignment Problems . . . . .	54
5.4	Algorithms . . . . .	57
5.5	Experimental Evaluation . . . . .	63
5.6	Discussion . . . . .	65
6	Summary of Contributions . . . . .	66
7	Future Work . . . . .	68
	References . . . . .	70

## II Papers 75

<b>A</b>	<b>A Survey of Spatial Crowdsourcing</b>	<b>77</b>
1	Introduction . . . . .	79
2	Spatial Crowdsourcing Infrastructure . . . . .	82
2.1	Requester . . . . .	82
2.2	Worker . . . . .	83
2.3	Spatial task . . . . .	84
2.4	Spatial Crowdsourcing Server . . . . .	85
3	Classification . . . . .	87
3.1	Spatial Tasks . . . . .	87
3.2	Modes of Publishing Spatial Tasks . . . . .	88
3.3	Types of Problems . . . . .	89
3.4	Constraints . . . . .	90
3.5	Privacy Protection . . . . .	91
3.6	Crowdsourced Data Aggregation . . . . .	91
3.7	SC Applications . . . . .	92
4	Task Matching Problem . . . . .	92
4.1	Introduction . . . . .	92
4.2	Offline Scenario: Known Arrival order of Tasks and Workers . . . . .	93
4.3	Online Scenario: Dynamic Arrival of Tasks and workers	96
5	Task Scheduling Problem . . . . .	100
5.1	Introduction . . . . .	100

## Contents

5.2	Offline Input Model: One worker- Multiple Tasks Scenario . . . . .	101
5.3	Online Input Model: One Worker- Multiple Tasks Scenario . . . . .	104
5.4	Combination with Task Matching Problems: Multiple Workers - Multiple Tasks Scenario . . . . .	106
6	Spatial Constraints . . . . .	110
6.1	Overview . . . . .	110
6.2	Spatial Region . . . . .	110
6.3	Maximum travel distance . . . . .	112
6.4	Direction of worker's commute . . . . .	112
7	Quality Constraints . . . . .	113
7.1	Overview . . . . .	113
7.2	Worker's Expertise . . . . .	113
7.3	Worker's Reputation/Reliability . . . . .	115
8	Budget Constraints . . . . .	116
8.1	Introduction . . . . .	116
8.2	Reward Models and Incentive Mechanisms . . . . .	116
9	Privacy Protection . . . . .	118
9.1	Introduction . . . . .	118
9.2	Cloaking Techniques . . . . .	119
9.3	Differential Privacy-based Techniques . . . . .	120
9.4	Encryption Techniques . . . . .	121
9.5	Impact on different constraints . . . . .	122
9.6	Protecting Task and Requesters' Location Privacy . . . . .	123
10	Truth Discovery and Crowdsourced Data Aggregation . . . . .	124
10.1	Introduction . . . . .	124
10.2	Non-Iterative Aggregation . . . . .	124
10.3	Iterative Aggregation . . . . .	125
10.4	Truth Discovery in Spatial Crowdsourcing . . . . .	126
11	Applications . . . . .	127
11.1	Data Collection . . . . .	127
11.2	Query Answering . . . . .	128
11.3	Personal Service . . . . .	128
11.4	Example SC applications . . . . .	129
12	Discussion . . . . .	130
12.1	Task Matching and Scheduling issues . . . . .	130
12.2	Privacy Issues . . . . .	132
12.3	Truth Inference Models . . . . .	132
12.4	Lack of real world datasets . . . . .	133
12.5	Lacking User Participation . . . . .	133
13	Future Research Directions . . . . .	134
14	Summary . . . . .	136

References . . . . .	137
<b>B Push-based Spatial Crowdsourcing for Enriching Semantic Tags in OpenStreetMap</b>	<b>147</b>
1 Introduction . . . . .	149
2 Preliminaries and Problem Definition . . . . .	151
2.1 Preliminaries . . . . .	151
2.2 Problem Definition . . . . .	153
3 Push-based SC for OSM Framework . . . . .	157
3.1 Task Generator Module: . . . . .	157
3.2 Task Assignment Module: . . . . .	158
3.3 Quality Control Module: . . . . .	159
3.4 Updating OSM Database Module: . . . . .	160
3.5 Use case: Road Networks . . . . .	160
4 Optimization Objective: Maximizing Total Valid Task Assignments . . . . .	160
4.1 Max Flow-based Task Grouping (TG) . . . . .	160
4.2 Direct Assignment with Road Segments (DA-RS) . . . . .	161
4.3 Direct Assignment with Junctions (DA-J) . . . . .	162
5 Optimization Objective: Maximizing Unique Task Assignments	162
5.1 Unique Assignments with Road Segments (U-RS) . . . . .	163
5.2 Unique Assignments with Junctions (U-J) . . . . .	163
6 Optimization Objective: Maximizing Verifiable Task Assignments . . . . .	164
6.1 Verifiable Assignments with Road Segments (V-RS) . . . . .	165
6.2 Verifiable Assignments with Junctions (V-J) . . . . .	166
7 Experimental Evaluation . . . . .	167
7.1 Experiment Setup . . . . .	167
7.2 Scalability . . . . .	168
7.3 Effect of varying maxT . . . . .	173
7.4 Effect of varying maxT on Check-ins dataset . . . . .	173
8 Related Work . . . . .	174
9 Conclusions and Future Work . . . . .	175
References . . . . .	177
<b>C Spatial Entity Linkage with the Aid of Spatial Crowdsourcing</b>	<b>179</b>
1 Introduction . . . . .	181
2 Related Work . . . . .	182
3 Problem Definition . . . . .	183
4 SkyCrowd Solution . . . . .	185
5 QuadSky Approach . . . . .	187
5.1 Extracting SEL tasks . . . . .	187
5.2 Automatic Labelling of SEL tasks . . . . .	188



## Contents

6	Identifying the grey area . . . . .	189
7	Crowdsourcing the Grey Area . . . . .	190
7.1	SC-based Active Learning Approach . . . . .	191
7.2	Transitive Closure of Unresolved Grey Area Tasks . . . . .	194
7.3	Clustering of Unresolved Grey Area Tasks . . . . .	194
7.4	Crowdsourcing the Cluster Samples . . . . .	195
7.5	Inferring the Cluster Labels based on Crowdsourced Samples . . . . .	196
8	Experimental Analysis . . . . .	196
8.1	Performance of Skycrowd solution . . . . .	197
9	Conclusions and Future Work . . . . .	200
	References . . . . .	201
<b>D</b>	<b>Transit-based Task Assignment in Spatial Crowdsourcing</b>	<b>205</b>
1	Introduction . . . . .	207
2	Preliminaries . . . . .	209
3	Transit-based Task Assignment . . . . .	211
3.1	Problem Definition . . . . .	211
3.2	Maximum Weighted Bipartite Matching (MWBM) Algorithm . . . . .	214
3.3	Minimum Distance based Direct Assignment (DA) Algorithm . . . . .	216
3.4	Credible Transit-based Task Assignment Algorithm (CTA) . . . . .	217
4	Flexible Transit-based Task Assignment . . . . .	219
4.1	Problem Definition . . . . .	219
4.2	Flexible Transit Route-based Direct Assignment Algorithm . . . . .	220
5	Experimental Evaluation . . . . .	222
5.1	Experiment Setup . . . . .	222
5.2	Scalability with the size of workers data set . . . . .	226
5.3	Scalability with the size of Tasks data set . . . . .	228
5.4	Effect of varying the number of workers and the number of tasks on CTA . . . . .	229
5.5	Effect of varying maxTravelTime, threshold reward on Flexible-DA . . . . .	230
5.6	Effect of varying minimum worker credibility threshold value on CTA . . . . .	230
5.7	Summary . . . . .	230
6	Related Work . . . . .	231
7	Conclusions and Future Work . . . . .	232
	References . . . . .	233

<b>E</b>	<b>Push-based Spatial Crowdsourcing for Enriching Semantic Tags in OpenStreetMap</b>	<b>235</b>
1	Introduction . . . . .	237
2	Preliminaries . . . . .	238
3	Algorithms . . . . .	240
3.1	Max Flow-based Task Grouping . . . . .	240
3.2	Direct Assignment with Road Segments . . . . .	241
3.3	Direct Assignment with Junctions . . . . .	241
4	Experimental Evaluation . . . . .	242
5	Related Work . . . . .	244
6	Conclusions . . . . .	244
	References . . . . .	245

# Thesis Details

**Thesis Title:** Multi-modal Spatial Crowdsourcing for Enriching Spatial Datasets  
**Ph.D. Student:** Srinivasa Raghavendra Bhuvan Gummidi  
**Supervisors:** Prof. Torben Bach Pedersen, Aalborg University  
Prof. Xike Xie, University of Science and Technology of China  
Prof. Esteban Zimányi, Université Libre de Bruxelles

The main body of the thesis consists of the following papers.

- [A] Srinivasa Raghavendra Bhuvan Gummidi, Xike Xie, and Torben Bach Pedersen. “A Survey of Spatial Crowdsourcing”. In: *ACM Transactions on Database Systems*, Article No.:8, ACM, March 2019.
- [B] Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach Pedersen, Xike Xie, and Esteban Zimányi. “Push-based Spatial Crowdsourcing for Enriching Semantic Tags in OpenStreetMap”. In: *preparation for submission to Geoinformatica Journal*.
- [C] Suela Isaj, Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach Pedersen, and Xike Xie. “Spatial Entity Linkage with the Aid of Spatial Crowdsourcing”. In: *Unpublished Manuscript*.
- [D] Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach Pedersen, and Xike Xie. “Transit-based Task Assignment in Spatial Crowdsourcing”. In: *Proceedings of 32nd International Conference on Scientific and Statistical Database Management (SSDBM2020)*, Vienna, Austria, Article 13, 1–12, ACM, 2020.

In addition to the main papers, Paper E has also been included. Paper B is the extended version of Paper E, thus Paper E can be considered as a subset of Paper B.

- [E] Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach Pedersen, Xike Xie, and Esteban Zimányi. "Push-based Spatial Crowdsourcing for Enriching Semantic Tags in OpenStreetMap". In: *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Chicago, USA*, pp. 532-535, ACM, 2019.

This thesis has been submitted for assessment in partial fulfilment of the PhD degree. The thesis is based on the scientific papers, which are listed above. Parts of the papers are used directly or indirectly in the extended summary of the thesis. As a part of the assessment, co-author statements have been made available to the assessment committee and are also available at the Technical Faculty of IT and Design at Aalborg University and the Faculty of Engineering at Université Libre de Bruxelles. The permission for using the published articles in the thesis has been obtained from the corresponding publishers with the conditions that they are cited and DOI pointers and/or copyrights/credits are placed prominently in the references.

Srinivasa Raghavendra Bhuvan Gummidi  
Aalborg University, February 24, 2021

**Part I**

**Thesis Summary**



# Thesis Summary

## 1 Introduction

### 1.1 Motivation and Background

The widespread availability of ubiquitous smartphones has facilitated advanced research on the topic of spatial crowdsourcing (SC) [24]. SC harnesses the potential of the crowd of workers to perform real-world tasks with a strong spatial nature, and with requirements that cannot be fulfilled remotely. Typically, a requester requests a spatial task to the SC-server, which in-turn assigns the requested spatial task to be performed by the registered workers or publishes the requested spatial task to be selected by the registered workers. For example (See Fig. 1), a requester has requested a delivery task which involves the worker picking up the package at location, POI A, and deliver it to the requester in the Building B. SC offers avenues to utilize both the smartphones' sensor information as well as the workers efforts. A worker's skillset can be considered during recruitment for performing a given task. Furthermore, in SC, the worker/ participant effort is often associated with constraints such as spatiotemporal availabilities, capabilities, quality, and budget. Such limitations necessitate optimized assignment and diligent planning for effective utilization of the workers.

A typical SC application has the potential to operate in different modes based on the application requirements (See Fig. 2). The different modes are classified based on the task complexity, the number of responses required for solving the task, spatial nature of the task, task publishing type, type of problem addressed, type of constraints considered, type of privacy protection measures, data aggregation technique employed and nature of the application. For example, the tasks that involve deliveries of items should not be assigned to multiple workers (Single-response mode). Similarly, tasks that involve data collection require multiple responses from different workers to ascertain the truth through data aggregation and truth inference techniques, for instance, tasks to collect traffic data (Multiple-responses mode). The different modes of SC are discussed in detail in Section 2. When an SC ap-

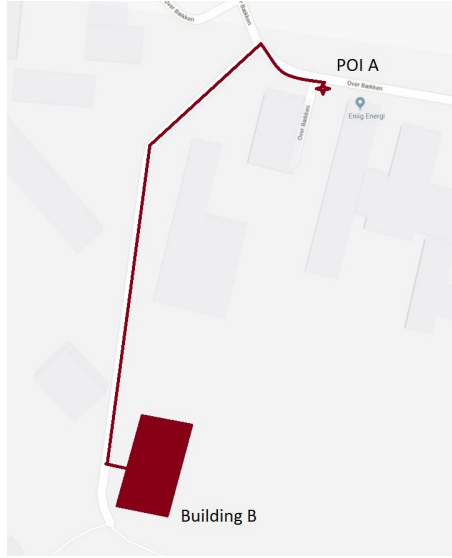


Fig. 1: Example spatial task (reproduced from Paper A [24])

plication operates in more than one mode, we define it as a *Multi-modal SC* application. Multi-modal SC applications further the potential of information collection by adjusting to realistic application requirements. In addition to the data collection and query answering tasks, Multi-modal SC can be extended to service complex spatial tasks like employing a set of workers with diverse skills to renovate the house, hiring workers to perform household chores, etc. Some of the multi-modal SC applications are environmental data collection application (NoiseTube platform [57]), transportation application (Uber<sup>1</sup>, and freelance marketplace application (Gigwalk<sup>2</sup> and TaskRabbit<sup>3</sup>).

Spatial datasets like OpenStreetMap<sup>4</sup>, Google Maps<sup>5</sup>, Flickr<sup>6</sup> provide the basis for research and applications in many domains. However, these spatial datasets suffer from problems related to coverage /sparsity and quality. Given its vast potential, the multi-modal SC can be employed to address this important problem of inadequacy of spatial datasets like OpenStreetMap. The objective is to enrich the spatial datasets and improve their coverage and quality. For instance, a multi-modal SC approach can enable active worker participation by assigning workers to collect semantic tag information of nearby OpenStreetMap spatial entities to improve the quality

<sup>1</sup><https://www.uber.com>

<sup>2</sup><http://www.gigwalk.com>

<sup>3</sup><https://www.taskrabb.it.com>

<sup>4</sup><https://openstreetmap.org/>

<sup>5</sup><https://www.maps.google.com>

<sup>6</sup><https://www.flickr.com>



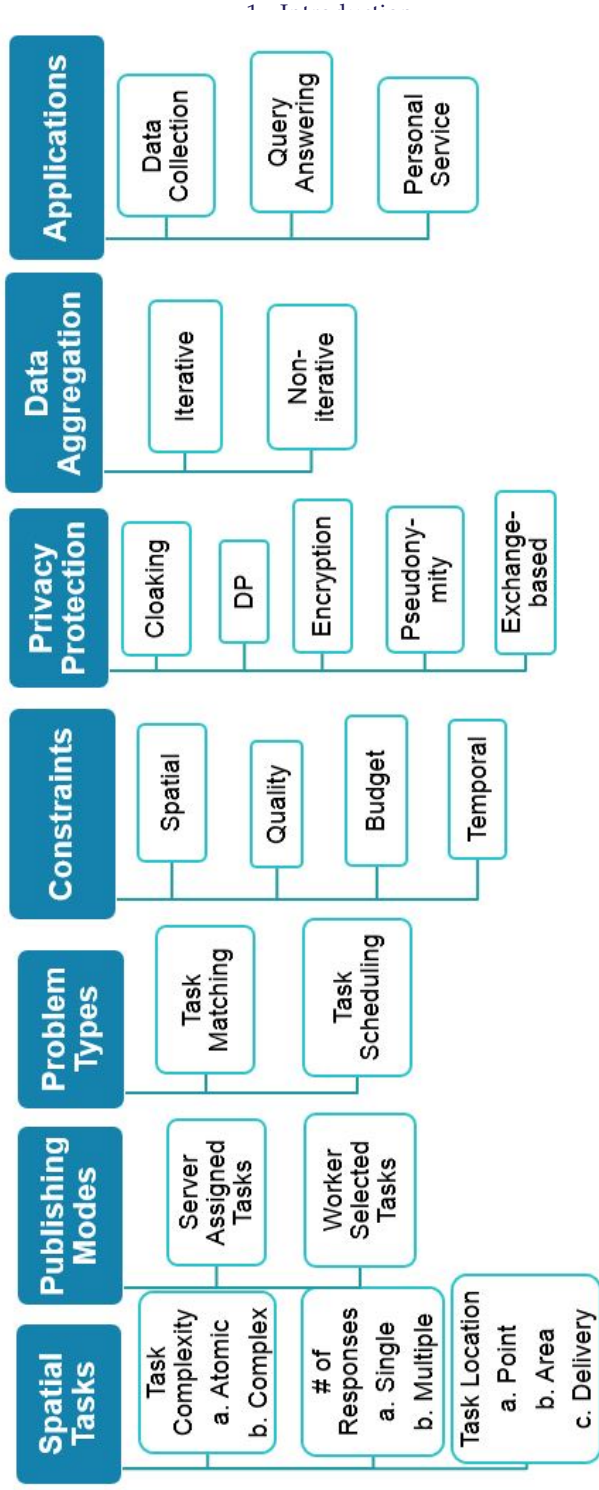


Fig. 2: Multi-modal SC: Enumeration of different modes (reproduced from Paper A [24])

and coverage. The multi-modal SC approach can be further extended to enrich spatial datasets by employing machine learning techniques to resolve the spatial entity linkage problem for integrating data related to spatial entities across multiple sources. The spatial entity linkage problem [31] involves finding out whether the spatial entities from various sources present at the approximately same geographical location proximity refer to the same real-world entity or two different entities. To further understand the problem, consider the following example. Two spatial entities A and B with locations  $\langle lat, long \rangle$  and  $\langle lat + 0.0001, long + 0.0001 \rangle$ , and associated tags 'Ram' and 'Rama' respectively, might refer to the same physical entity. New groups of moving workers like public transportation users should be attracted to improve the worker participation of the multi-modal SC approach for enriching spatial datasets. By targeting workers who transit by public transportation, the transit-based multi-modal SC approach enables a new way of performing tasks and thus, facilitates workers to earn rewards during their daily commute. This new group of transit workers, that travels by public transportation services brings forth challenges related to real-time task assignment and task scheduling. This thesis explores the idea of employing multi-modal SC in combination with machine learning techniques for enriching spatial datasets and encourages the worker participation of multi-modal SC approach by attracting a new group of moving workers travelling via public transit.

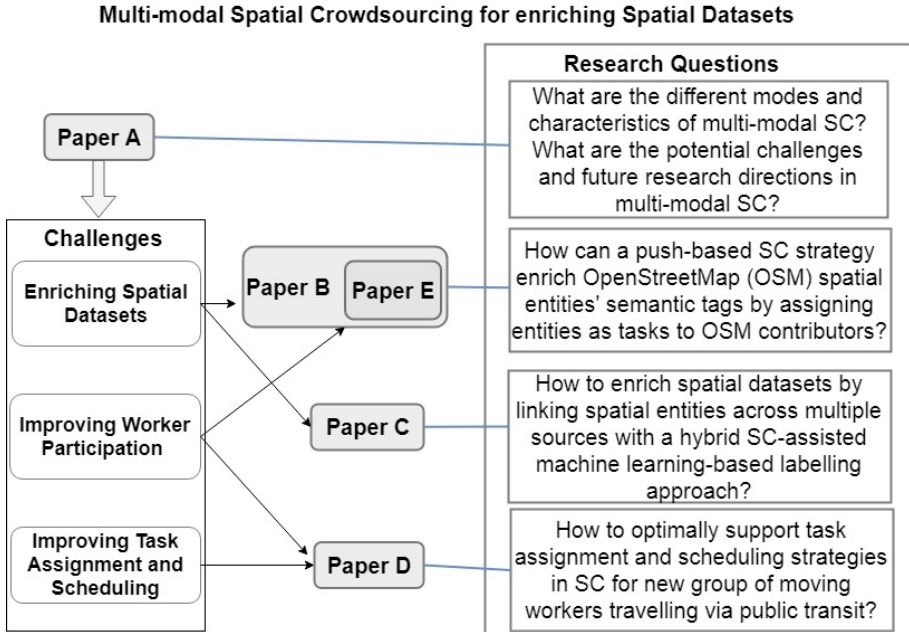


Fig. 3: Thesis Structure

### 1.2 Thesis Structure

The thesis is organized as follows (See Fig. 3):

Part I provides the motivation for the thesis and the summary of the five papers included in the thesis. Part II reproduces the five papers in full with minor layout changes. Paper A [24] provides a comprehensive overview of the existing SC literature, describes the SC usage workflow, introduces taxonomy, discusses the different issues and challenges faced by current SC approaches, and lists some future research directions. Through Papers B, C, D, and E, we address the following challenges of SC: enrichment of spatial datasets, improvement of worker participation, and enhancement of task assignment and scheduling strategies (See Fig. 3).

Papers B, E, and C [22, 23, 30] focus on enriching spatial datasets by exploiting the potential of multi-modal SC. To elaborate, Papers B & E addresses the issue of missing tag details of OSM spatial entities to improve the OSM data quality and coverage. Paper E introduces the concept of employing multi-modal SC for enriching the semantic tag information of OSM by actively assigning workers to nearby spatial entities. Furthermore, Paper B extends Paper E and proposes a comprehensive multi-modal SC application framework which includes task generator, task assignment, quality control and OSM update modules.

The spatial datasets can also be enriched through the data integration of spatial entities across multiple data sources like Google<sup>7</sup>, Flickr, Krak<sup>8</sup>, and OpenStreetMap. However, it is a huge challenge to infer whether the spatial entities across sources within close geographical proximity refer to the same real-world entity or two different entities. Paper C addresses this spatial entity linkage problem and proposes a hybrid method to enrich spatial datasets by linking spatial entities across multiple sources with an SC-assisted machine learning-aided labelling approach.

The proposed multi-modal SC applications proposed in Papers B, E, and C are adequate for enriching spatial datasets. However, the success of the proposed multi-modal SC applications is dependent on worker participation. Thus, to improve worker participation Paper D [21] focuses on a new group of moving workers who commutes by public transport and can perform tasks during their daily commute. Paper D focuses on addressing the challenges faced by the new group of public transit workers by designing customized task matching and task scheduling strategies.

The example in Fig. 4 showcases how the different parts of the thesis can work together on an overall problem of enriching datasets using machine learning and transit-based multi-modal SC approach. The semantic tags of OSM spatial entities can be enriched by identifying the corresponding spatial

---

<sup>7</sup>[www.google.com](http://www.google.com)

<sup>8</sup>[www.krak.dk](http://www.krak.dk)

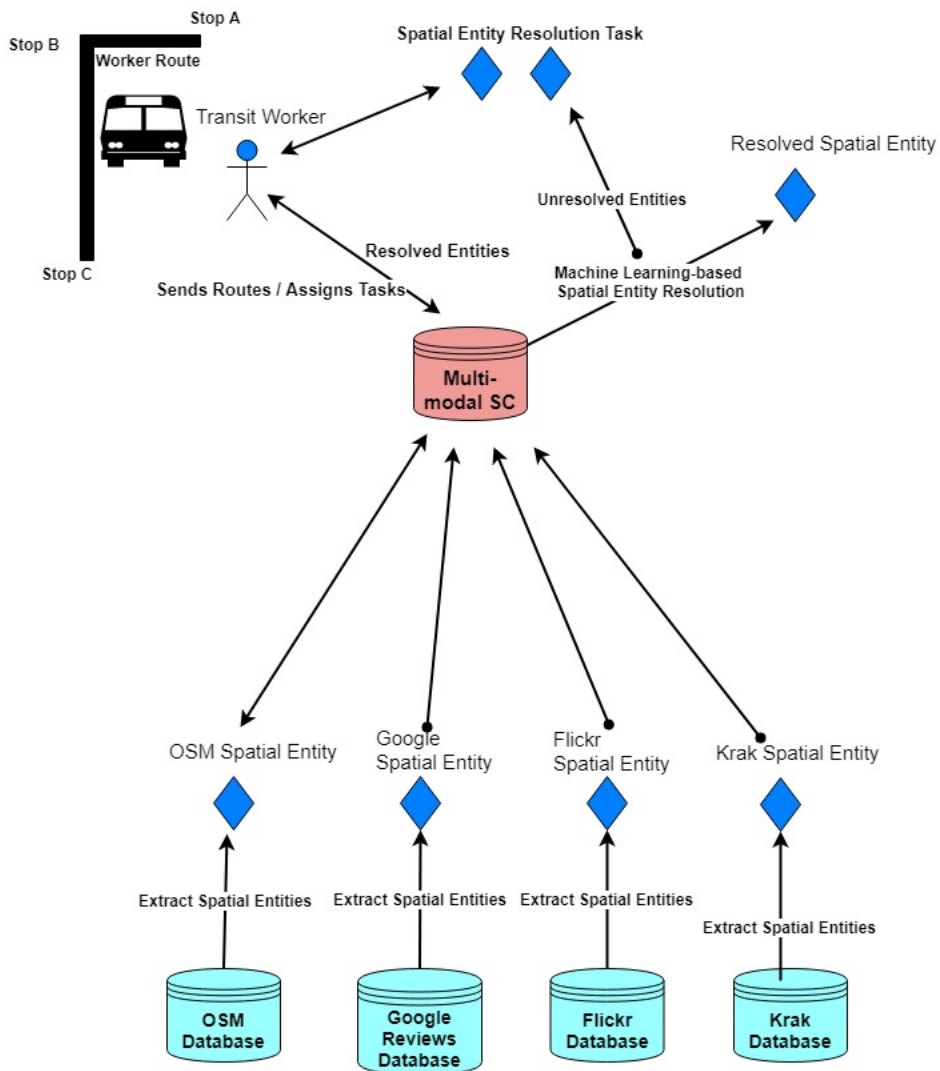


Fig. 4: An example showcasing the potential synergy between different parts of the thesis

entities in other sources like Google, Krak, and Flickr. To link the corresponding spatial entities to OSM spatial entities, we employ a hybrid iterative SC-aided labelling approach by combining multi-modal SC with machine learning [30]. The hybrid iterative SC-aided labelling approach resolves a portion of the spatial entities by employing a machine learning-aided automatic labelling method. The proposed machine learning-aided automatic labelling method is an extension of the method proposed in [31]. The remaining unresolved portion of spatial entities are designed as spatial entity resolution tasks [30]. The spatial entity resolution tasks are assigned to workers traveling via public transportation considering their routes and availability. The workers will visit the location of spatial entities during the waiting periods of their transit route, to verify whether the entities across sources refer to same physical entity.

Fig. 3 shows how the papers are related and their corresponding research questions. The recommended order of reading the thesis is : Paper A ==> Paper B ==> Paper C ==> Paper D. Reading Paper E is optional.

## 2 Spatial Crowdsourcing Literature Review

### 2.1 Motivation and Problem Statement

As mentioned in Section 1, a recent trend of increased interest in SC research necessitated a comprehensive compilation/ survey of existing SC techniques and its application scenarios. Moreover, we performed the survey (Paper A [24]) to identify the challenges impacting the progress of SC and the prospective directions for future research. It should be noted that there are existing surveys/ tutorials/ short articles [18, 52, 64, 78] on SC, however, they do not offer a comprehensive view with additional technical details for a better understanding of SC. For instance, our survey offers a technical perspective of SC, and a comparison between different SC strategies highlighting the relationship between different types of constraints and their impact on different optimization objectives. Additionally, we discuss the data aggregation methods, associated truth inference models, and privacy preservation techniques employed in SC, and classifies the different applications of SC based on their functionality.

### 2.2 Spatial Crowdsourcing Workflow

*Spatial Task, Requester, Worker, and SC server* are the major components in SC. To understand the interactions and relationships between the different components of SC, we describe the spatial task lifecycle in SC (see Fig. 5). The requester initiates the process by posting spatial task/s to the SC-server (Step 1 of Fig. 5). Based on the type of task publishing mode (*Server-Assigned/*

*Worker-Selected*), the SC-server assigns the spatial tasks to the selected workers or lists the spatial tasks on a platform where the workers can choose according to their interests (Steps 2 & 3 of Fig. 5). Subsequently, the worker performs the task by visiting the task location and sends the collected information/ answer for the task query to the SC-server (Steps 5 & 6 of Fig. 5). The workers' responses are aggregated/ processed by the SC-server and forwarded to the requester (Steps 7 & 8 of Fig. 5). Finally, the requesters verify the quality of the task responses and provide feedback to the SC-server (Step 9 of Fig. 5). In the remainder of this subsection, we will elaborate on the four components of SC infrastructure (reproduced from Paper A [24]).

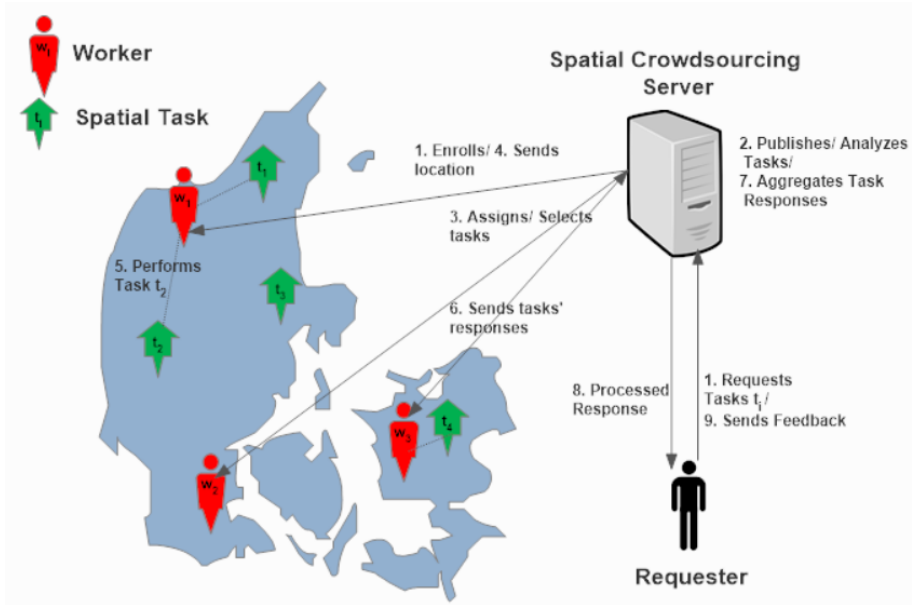


Fig. 5: Spatial Crowdsourcing Workflow Scenario (reproduced from Paper A [24])

### Requester (reproduced from Paper A [24])

A requester is a real-world entity like a person or an organization that requests for a particular spatial task to be completed by the crowd at a specific location (see Fig. 5). A requester designs the task and sets the conditions that need to be satisfied for performing the task. The requester has certain responsibilities such as accepting the answers provided or data collected by the crowd and giving them feedback [9].

### Worker (reproduced from Paper A [24])

A worker is a person that participates in the process of SC, with an objective to perform the assigned/ selected spatial task.

**Worker Definition:** Different definitions mentioned in the literature for a worker  $W$ , like:

$$W = \langle l, R, maxT, Re, E, \theta \rangle \quad (1)$$

Most of the definitions essentially had the following basic and optional attributes:

Basic attributes [36]

- Current physical location of the worker  $l$
- Region of interest  $R$
- Maximum number of tasks  $maxT$

Optional attributes:

- Expected reward  $Re$  [14]
- Skillset  $E$  [14]
- Worker's reputation score calculated by SC-server  $\theta$  [37]

### Spatial task (reproduced from Paper A [24])

A spatial task is requested by the requester and is fulfilled by the worker. It is a location-specific activity like answering a question about a local restaurant, taking pictures of a local tourist spot, or collecting noise pollution data. Moreover, a SC task will be defined by the requester with a set of requirements for assigning or allowing a person from the crowd to work on the task. The requirements might include the minimum level of skills required to fulfil a task, the number of answers needed, the expertise of the user, the task deadline, and their experience in solving similar tasks.

**Spatial Task Definition:** Different definitions mentioned in the literature for a spatial task  $t$ , like:

$$t = \langle l, q, ti_i, ti_e, r, \alpha, E, Cat \rangle \quad (2)$$

Most of the definitions essentially had the following basic and optional attributes:

Basic attributes [36]

- Physical location of the task  $l$
- Query description  $q$

- Issuing time  $ti_i$
- Expiration time  $ti_e$

Optional attributes:

- Associated reward  $r$  [14]
- Minimum threshold for worker's reputation  $\alpha$  [37]
- Expected worker's skillset  $E$  [10]
- Type/category of task  $Cat$  [25]
- Maximum number of workers  $maxW$  [75]

### Spatial Crowdsourcing Server (reproduced from Paper A [24])

The SC-server facilitates the requester to request a task and the workers to be assigned or select the tasks. It delegates the communication between the requester and the worker of the spatial task, facilitates the process to satisfy the task requirements, assigns tasks to workers based on location, helps improving the quality of the outcome by executing different strategies, identifies the anomalies and detects fraudulent responses, and protects the privacy of the involved stakeholders.

## 2.3 Modes in SC

Based on the differences identified in the surveyed SC literature, we determined the modes (see Fig. 2) in SC based on the following classification criteria: the type of problem addressed, the modes of publishing spatial tasks, the different constraints considered, the type of application, the type of spatial tasks, and the different privacy-preserving and data aggregation techniques employed. In this subsection, we will detail the different modes in SC.

### Spatial Tasks

Spatial task modes are dependent on their complexity, location type, and the required number of worker responses.

#### Spatial task complexity:

- **Atomic tasks:** The task cannot be divided down into sub-tasks, and can be performed by one worker [36].

---

<sup>9</sup>[www.mturk.com](http://www.mturk.com)



## 2. Spatial Crowdsourcing Literature Review

		Spatial Task Publishing Modes			
		Server Assigned Tasks		Worker Selected Tasks	
Constraints	Spatial constraints	a. Spatial Region [13, 36, 37] b. Direction of worker's Commute [8, 11]		Spatial Region [15, 44]	
	Quality Constraints	a. Worker Reputation: [11, 37, 50] b. Worker Expertise: [10, 14, 63] c. Truth Inference [20, 27, 48]		Qualification tests [45, 53]	
	Temporal Constraints	a. Deadline of Task [36] b. Delivery Task PickUp time and Deadline [58] c. Deadline for Destination [44]		Deadline for task [15]	
	Budget Constraints	Reward models and Incentive Mechanisms [14, 19, 26, 40, 72]		Fixed Rewards for Tasks <sup>9</sup> [2]	
Scenarios		Online		Offline	
Types of Problems	Task Matching	MAB [25], GOMA [65], f-MTC, d-MTC [59, 67], FTOA [66], TOM [56]		MTA [36], MSA [63], MTMCA [14], RDB-SC [11], MRA [76], PAPA [35], DPTA [61], MS-SC [10]	-NA- -NA-
	Task Scheduling	GALS [16], TRACCS [8]		MTS [15]	OnlineRR [44], Auction-SC [3] MTS [15]
Privacy Protection		a. Differential Privacy-based [60, 61, 74] b. Spatial Cloaking [35, 49] c. Encryption-based [54, 55]		a. Pseudonymity [12] b. Exchange-based [73]	

**Table 1:** Multi-modal Spatial Crowdsourcing based on different constraints (reproduced from Paper A [24])

- **Complex tasks:** A complex task can be divided into sub-tasks and require workers to complete all the sub-tasks to complete the complex task [5, 10, 36, 63].

**Number of responses:** A spatial task can be completed by a single worker response or multiple worker responses.

**Task's Physical location:** A spatial task location can be defined as a single geographical point, or a line segment or a region (See Fig. 1).

### Modes of Publishing Spatial Tasks

SC-server facilitates two spatial task publishing modes [36]:

- **Server Assigned Tasks:** Given a spatial task, the SC-server selects

workers according to the task constraints and worker requirements.

- **Worker Selected Tasks:** The SC-server publishes the given spatial task in a list to enable workers to select tasks based on their interest.

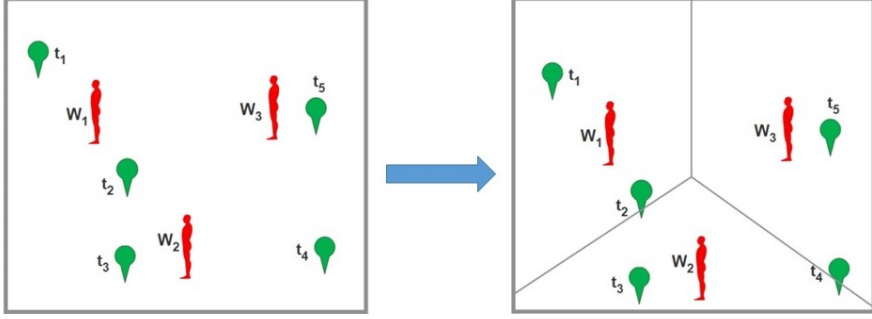


Fig. 6: Task Matching Problem Example (reproduced from Paper A [24])

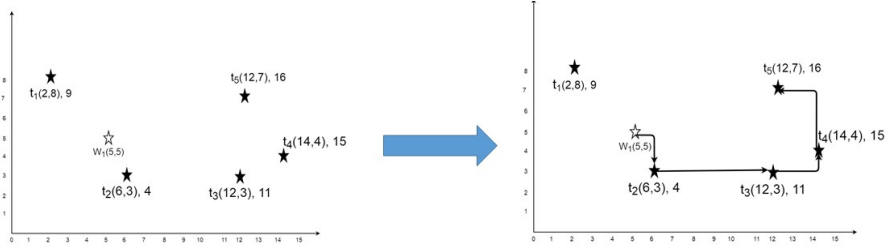


Fig. 7: Task Scheduling Problem Example (reproduced from Paper A [24])

## Types of Problems

Based on the function of the surveyed optimization problems in SC literature, we broadly categorize them into two different categories:

- **Task Matching:** involves assignment of a given set of spatial tasks to the available set of workers in *Server Assigned Tasks* mode [36] (See Fig. 6).
- **Task Scheduling:** involves scheduling of the server-assigned/ worker-selected set of tasks for the workers to achieve different optimization objectives like maximizing the tasks completed by the worker, maximizing the incentives for the workers (See Fig. 7).

Furthermore, the optimization problems can be classified based on the input model (online/ offline) and the optimization goals (workers' perspective/

SC-server's perspective). Table 1 provides an elaborate overview of the different optimization problems under the two task publishing modes.

### Constraints

As discussed in Section 2.2, there exists different constraints for tasks and workers, that can be categorized into four categories. Table 2 illustrate some of the different task matching and task scheduling problems along with their constraint details.

- **Spatial Constraints:** Workers and requesters can register their spatial preferences like preferred regions to work [59], the distance she is willing to travel [8], to the SC-server. SC-server would consider these preferences as spatial constraints during the task matching or scheduling process.
- **Temporal Constraints:** Similar to spatial constraints, workers and requesters can register their temporal preferences like the task deadline [36], or the worker's availability [44].
- **Quality Constraints:** Requesters can register their expected level of quality from worker responses to the SC-server. The SC-server fulfils the requester's quality expectations by performing pre-task qualification tests [53] or assignment based on previous histories [37]/abilities [14] of the worker.
- **Budget Constraints:** SC-server receives the budget limitations from requesters and reward expectations from the workers. SC-server considers these constraints while optimizing the incentive scenarios for workers and requesters.

### Privacy Protection

Privacy protection is a critical aspect in SC, due to the potential privacy risks regarding storing workers/ requesters information in a third-party server like SC-server. Given the centrality of workers' and tasks' locations to SC, different privacy-preserving techniques were employed to address the privacy concerns of the different stakeholders like differential privacy [61]. We can broadly categorize these privacy-preserving techniques into the following five types [62]:

- **Pseudonymity Techniques:** uncouple the person's identity and the submitted data [12](reproduced from Paper A [24]).
- **Cloaking Techniques:** hide the exact locations of the workers in a cloaked region [35, 49](reproduced from Paper A [24]).

**Table 2:** Summary of the Surveyed Task Assignment and Scheduling Problems: The acronyms used according to category are: **Constraints:** SPC - Spatial, TC - Temporal, BC - Budget, QC - Quality, **Dynamic nature of Inputs:** TA, WA - Task and Worker Arrival, **Heuristics:** Gr-Greedy, TrC-Travel Cost, LE-Location Entropy, NN-Nearest Neighbor. **Privacy:** PP- Privacy Protection (reproduced from Paper A [24])

Problem Type	Problem Name	Support for Constraints				Dynamic Arrival		Reduced Problem	Algorithm Heuristic	Limitations
		SPC	TC	QC	BC	PP	TA	WA		
	MTA [36]	X	X					Max-Flow	Gr	a. Does not work well in a dynamic setting, esp. greedy strategy. b. Workers are assumed to accept tasks that are assigned to them. c. Trusts the worker to perform the task correctly. d. Does not consider the worker movement patterns
								MinCost-MaxFlow	LE	
	MSA [63]	X	X	X				MWBM	Gr	
								MinCost-MWBM	LE	
	MTMCA [14]	X	X	X	X			MinCost-MaxFlow	TC	a. Worker is assumed to be constantly moving in a specific direction with a certain velocity. b. No Option to limit the distance a W can travel, resulting in assignment of far away tasks.
	RDB-SC [11]	X	X	X				Number Partition	Gr	
	MIRA [76]	X	X					SCP [4]	Gr	
	PAPA [35]	X			X			Set Cover	Gr	
	DPTA [61]	X			X				TC	a. Does not take into account the movement of workers and the dynamic arrival of tasks like it is prone to homogeneity attack, or the privacy relies on the value of $k$ . b. Does not take into account the movement of workers c. Produces sub-optimal assignment due to fake workers in the geocast region.
	MS-SC [10]	X	X	X				Set Cover	Gr	
Task Matching	OSTA [25]					X		MAB	g-D&C	a. Assumes the cost of performing a task proportional to travel cost, and Euclidean distance function is used to calculate travel cost. b. Only one worker can be assigned to the task
									Adaptive	
									Spatial Context	
									GrOffline	
	Fixed Budget MTC [59, 67]	X	X		X		X	MCP	Gr	a. Limited to hyper local tasks that do not require the workers to travel. b. Worker cannot try to maximize his reward by performing sequence of tasks. c. The performance is hindered due to the dynamic arrival of workers, as the budget is allocated per time periods d. Offline algorithms, though not practical, helps in tackling the randomness of the real-world online problem.
									LE	
									GrOffline	
									Temporal	
	Dynamic Budget MTC [59, 67]	X	X		X		X	MAB	Gr	a. Dependent on the accuracy of the initial prediction model b. Not beneficial when the workers are more in number than the tasks.
									LE	
	FTOA [66]	X	X				X	Offline Guide: Max-Flow	TrC	a. Focuses on micro-tasks that are simple and trivial. b. Focuses on maximizing the benefit of SC-server and not the worker.
	GOMA [65]	X	X	X	X		X	OMWBM	Gr	

## 2. Spatial Crowdsourcing Literature Review

Table 2: (Continued) (reproduced from Paper A [24])

Problem Type	Problem Name	Support for Constraints				pp	Dynamic Arrival		Reduced Problem	Algorithm Heuristic	Limitations
		SpC	TC	QC	BC		TA	WA			
Task Scheduling	MTS [15]	X	X		X				TSP	Temporal NN MPH	a. Not applicable in the case of dynamic arrival of tasks b. The set of tasks is preselected for the worker.
	OnlineRR [44]	X	X			X			OPTW	Temporal NN MCS	a. Set of tasks arrive dynamically, however schedules plan for only one worker. b. Does not take into account some constraints like quality
	MTSMW [16]	X	X						Matching: MTA Scheduling: Routing	Same as MTA TC	a. Assignment module has similar limitations as the MTA problem b. Assumes new task insertion does not effect task order.
	TRACCS [8]	X	X						Orienteering	Gr ILS	a. Partial order is maintained among the nodes visited. b. Preference to individual benefits than globally optimal solution.
	OnlineTASC [3]	X	X		X		X	X	Min-Len-Ham	Random Ranking NN Most Free Time Best Insertion Best Dist.	a. Does not consider the required time for completion of task b. Does not support complex tasks
Task Matching & Scheduling	DSTA [77]	X	X							Gr	a. Dynamic arrival of tasks and workers is not considered b. Does not consider the required time for completion of task

- **Exchange-based Techniques:** exchange the crowdsourced information among the workers before disclosing it to the untrusted SC-server, to obscure the individual workers [73](reproduced from Paper A [24]).
- **Encryption-based Techniques:** hide the identity and the workers' location from the SC-server [54, 55](reproduced from Paper A [24]).
- **Differential Privacy-based Techniques:** distort the workers location information by adding artificial noise [60, 61, 74](reproduced from Paper A [24]).

### Crowdsourced Data Aggregation

As discussed earlier, some tasks need multiple workers responses, for example, determining the traffic at a junction. Unfortunately, due to workers' incompetence or malice, the responses could be conflicting with each other. In such scenarios, the SC-server needs to ascertain the truth to fulfil the task from the crowdsourced data using different aggregation techniques in combination with various spatial attributes. We categorize these aggregation techniques into two categories [28].:

- **Non-iterative Aggregation:** aggregates the responses for each question to a single value. Examples are Majority Voting [41], Honeypot [42], and Expert Label Injected Crowd Estimation(ELICE) [38](reproduced from Paper A [24]).
- **Iterative Aggregation:** calculates the aggregated value iteratively for each question based on the expertise of the workers who answered and updates the worker expertise iteratively based on the answer given by the worker. Examples are Expectation Maximization (EM) [29], Generative Model of Labels, Abilities, and Difficulties (GLAD) [68], Supervised Learning from Multiple Experts(SLME) [51], and Iterative Learning(ITER) [34](reproduced from Paper A [24]).

### SC Applications

SC offers an extensive repertoire of applications servicing a wide range of domain, from ride-sharing services to environmental monitoring, from data collection during disasters to food delivery to homes. Based on the surveyed SC applications, we identified three types of SC applications into three broad categories based on sensor utilization, human knowledge, and human efforts.

- **Data Collection:** involves collecting data from the sensors on the worker's phone instead of utilizing the human intellect. For example, mapping the wifi strength in different parts of the building.

- **Query Answering:** involves exploiting the worker's knowledge to answer the task query. For example, we require the worker's knowledge to determine the crowd at a restaurant.
- **Personal Service:** involves physical effort from the workers, like pick-up and delivery of a package/ food/ groceries/ etc.

### 2.4 Challenges in SC

Based on the surveyed SC literature, we have identified some challenges faced by the current SC approaches.

- **Enriching Spatial Datasets:** Popular spatial crowdsourcing-based datasets like OSM find limited utility due to the lack of adequate semantic tag information associated with their spatial entities. The lack of adequate tag information severely hinders the OSM's data quality. Given the importance of spatial datasets like OSM, it is essential to improve their data quality. Enrichment of the spatial datasets like OSM with additional tag information for existing spatial entities enhances the quality. Furthermore, it was observed that there is a dearth of real-world spatial datasets for supporting and evaluating the SC algorithms. Most of the SC works evaluate their proposed algorithms using synthetic datasets and adapted location-based social network datasets. Through enrichment of spatial datasets with multi-modal SC, a real-world dataset could be curated for evaluating different SC works as well. Moreover, server-based task assignment methods need to be improved for better utilization of the OSM contributor/ worker's time and efforts. Additionally, task assignment methods have to ensure more significant coverage of crowdsourced entities and facilitate the verifiability of crowdsourced responses by assigning each spatial entity to at least two workers.
- **Improving worker participation:** The majority of the current SC approaches attract limited user participation. Although some use incentive mechanisms to increase user participation, it is still limited to users with knowledge about SC.
- **Improving task assignment and scheduling:** The majority of the current task matching and scheduling approaches ignore the worker's movement information. Workers' movement information is vital for improving the task assignment and scheduling strategies. Additionally, they optimize for the benefit of either workers or tasks instead of finding a middle-ground that is beneficial to both workers and tasks.

- **Privacy Issues:** There are many privacy issues yet to be addressed in the current SC approaches. For example, different worker constraints like spatial, budget, quality constraints are not considered for privacy protection. Additionally, the task location's privacy is not protected, raising concerns for potential adversary attacks at the task location. Furthermore, the current SC data aggregation/ truth inference mechanisms do not address individual's privacy concerns.

## 2.5 Challenges Addressed in this Thesis

Through this thesis, we have addressed three of the four challenges mentioned in Section 2.4.

- **Enriching spatial datasets:** Papers B, E, and C [22, 23, 30] focus on the challenge to enrich spatial datasets. Specifically, papers B and E focus on enriching semantic tags of spatial entities in OpenStreetMap dataset. Additionally, paper C focuses on extracting additional tags for the spatial entities by integrating different spatial datasets like Google, Krak, and Flickr by resolving the spatial entity linkage problem.
- **Improving worker participation:** Papers B, E, and D [21–23] focus on the challenge of improving the worker participation. Specifically papers B and E proposes a push-based SC approach to improve the OSM worker/ contributor participation by actively pushing nearby spatial entities as tasks. Furthermore, paper D focuses on targeting a new group of workers, that commute daily using the public transportation services. Paper D harnesses the waiting period at transit stops in a worker public transportation route to perform tasks and earn rewards during daily commute.
- **Improving task assignment and scheduling:** Paper D focuses on improving the task assignment and scheduling strategies by considering the worker movement information. Paper D proposes custom task assignment and scheduling strategies to assign tasks during waiting periods at transit stops along the worker transit route.

## 3 Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap

### 3.1 Motivation and Problem Statement

OpenStreetMap is a spatial crowdsourcing platform which facilitates amateur cartographers/ geoinformatics professionals/ volunteers to digitally map the



### 3. Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap

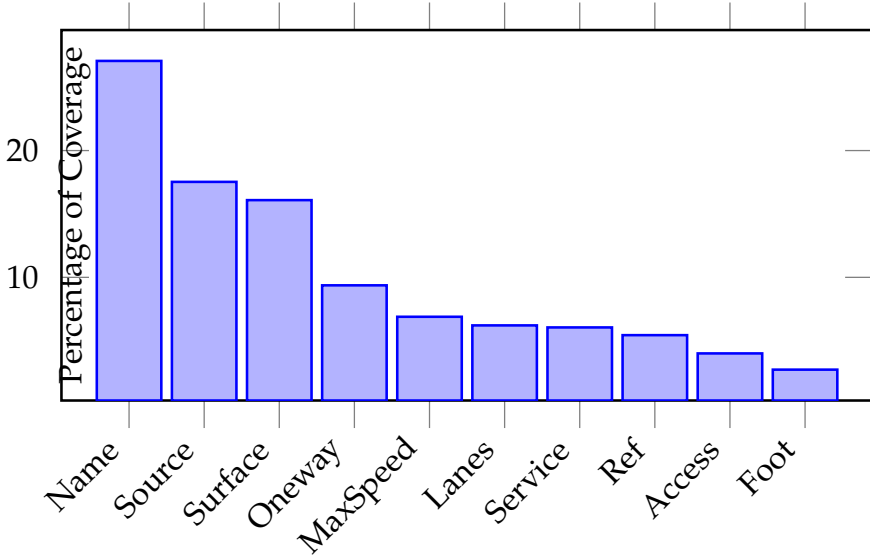


Fig. 8: OSM Tag Coverage for Road Segments (Reproduced from Paper B [22])

spatial features in their geographical areas of interest. The digitization process is aided by satellite images/ orthographic photos and dependent on the local knowledge of the contributor/worker to verify the digitally mapped spatial features.

OpenStreetMap (OSM) data quality is hindered by the missing tag details of spatial entities. For example, let us take the case of the road network in the OSM database. According to the OSM standards, each road segment can be tagged with 27 standard semantic tags (<https://taginfo.openstreetmap.org/>) like the type of the road, the name of the road, etc. Additionally, the contributor/ worker can define some new tags as well. However, the current road network's tag coverage in the OSM database is not adequate to adhere to the high-quality standards of other mapping and routing services like Google Maps ([www.maps.google.com](http://www.maps.google.com)). As seen in Fig. 8, we have observed that the coverage of the top ten tags for highways or road segments in the OSM database is inadequate, with even the *Name* tag being associated with only 27.07% of the total road segments in the entire OSM database. Only the top three tags, like *Name*, *Source*, and *Surface*, are associated with at least 10% of the total road segments. Given the importance of OSM, we need to develop mechanisms to enrich the semantic tags associated with OSM spatial entities to improve OSM data quality.

Furthermore, the conventional OSM contributions are voluntary in nature

and are usually made by a contributor without the local knowledge of the mapped area. This could result in inefficient mapping and an insufficient number of tags for the mapped spatial feature. And in some extreme cases, there is a potential for intentional/ unintentional mislabeling of tags as well. Consequently, it is imperative for the SC-server to directly assign the spatial entities to the contributors/ workers to avoid such pitfalls considering the different spatiotemporal constraints and to ensure that the mapped spatial entities contain a relevant and an adequate number of tags.

To investigate the potential for employing a server-assignment based multi-modal SC approach to enrich OSM dataset, we propose a comprehensive framework to crowdsource additional semantic tags for the existing spatial entities. The details regarding the framework’s architecture are discussed in the next subsection.

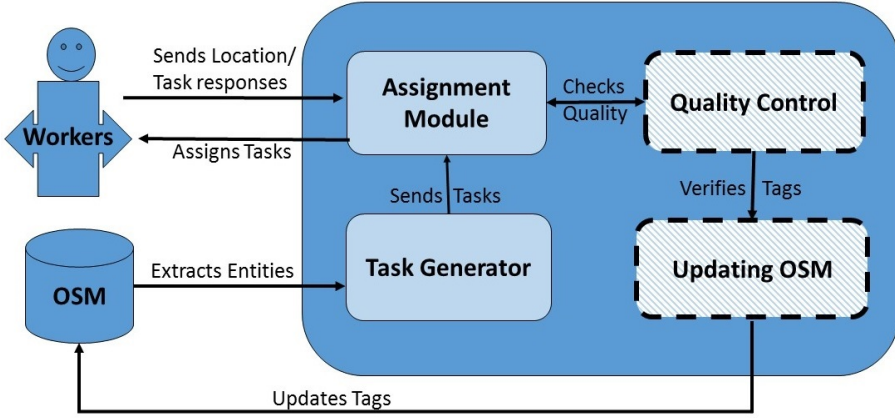


Fig. 9: Push-based multi-modal SC Framework for OSM (Reproduced from Paper B [22])

### 3.2 Proposed Framework Architecture

The architecture for the proposed server-assignment (push-based) multi-modal SC approach is illustrated in Fig. 9. The proposed framework constitutes four modules: the Task Generator module, the Task Assignment module, the Quality Control module, and the Updating OSM Database module. Additionally, communication with two external stakeholders is facilitated by the framework; the OSM database and the SC workers. For instance, the framework communicates with the OSM database to extract the spatial entities for assignment and to update the collected semantic tags. Owing to the unavailability of real crowdsourced responses to evaluate the functionality of the *Updating OSM Database* and *Quality Control* modules, we have limited the focus of the paper to *Task Generator* and *Task Assignment* modules. However,

### 3. Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap

we facilitate the verification aspect of the *Quality Control* module by ensuring multiple crowdsourcing responses for the same spatial entity for inferring the verified truth by aggregation. We focus on the use-case of the OSM road network to better understand the effectiveness of the proposed framework.

### 3.3 Task Generator Module

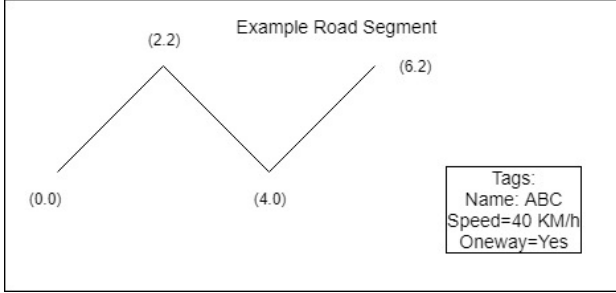


Fig. 10: Example Road Segment (Reproduced from Paper B [22] & Paper E [23])

The Task Generator module extracts spatial entities from the OSM database and generates tasks for each spatial entity. For example, building entities can be extracted as building tasks whose information can be collected at the building location. The task generator module sends the extracted tasks to the task assignment module. The task generator module is designed to re-query the spatial entities from the OSM database in a periodic manner; for example, the tasks will be extracted once every year for updating the tags.

The Task Generator module creates a set of road segment tasks  $RST$  based on the extracted road segment from the previous extraction step. A created road segment task  $rst \in RST$  is defined as:

**Definition 1.** Road Segment task:(Reproduced from Paper B [22] & Paper E [23])

A road segment task  $rst$  contains a line segment with  $m$  nodes. The start node and end node of the line segment are located at  $l_1$  and  $l_m$ , respectively. The road segment task  $rst$  has information regarding the  $n$  tags ( $tag_n$ ) along with their values ( $val_n$ ). The total number of tags  $n$  will be at least 27, as we would like to retrieve the information about the 27 standard tags, along with the existing custom tags. The road segment task  $rst$  has no temporal constraints associated with it, owing to the dynamic nature of the road network. However, the assignment of the task is tracked by the attribute *assigned*.

$$rst = \langle \langle tag_1, val_1 \rangle, \langle tag_2, val_2 \rangle, \dots, \langle tag_n, val_n \rangle, \langle l_1, l_2, \dots, l_m \rangle, assigned \rangle$$

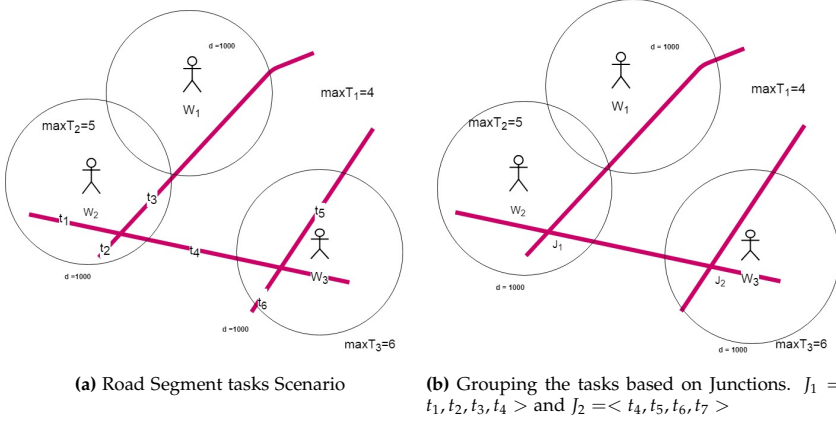


Fig. 11: Grouping the tasks based on Junctions (Reproduced from Paper B [22] & Paper E [23])

Consider the example in Fig. 10, the location set of the road segment  $\langle (0,0), (2,2), (4,0), (6,2) \rangle$  is mapped to node locations  $\langle l_1, l_2, l_3, l_4 \rangle$ . The tag-set of the road segment will be mapped to the respective tags of the road segment task and the tags that are not present in the road segment will be marked as empty, i.e.,  $\langle \langle \text{Speed} = "40\text{Km/h}" \rangle, \langle \text{Name} = "ABC" \rangle, \langle \text{Oneway} = "Yes" \rangle, \langle \text{Lanes} = "" \rangle, \langle \text{Surface} = "" \rangle, \dots \rangle$ . (Reproduced from Paper B [22] & Paper E [23])

We also group road segments at junctions and create a set of junction tasks with the roads segments intersecting at the junction. We formally define Junctions as:

**Definition 2.** Junction task J: (Reproduced from Paper B [22] & Paper E [23])  
A Junction  $j$  is an intersecting point of two or more road segment tasks.

$$j = \langle jid, l_j, \langle rst_1, rst_2, \dots \rangle \rangle \quad (3)$$

, where  $jid$  is the junction id,  $\langle rst_1, rst_2, \dots \rangle$  are the list of road segment tasks meeting at location  $l_j$  of the junction  $j$ .

### 3.4 Task Assignment Module

The Task Assignment module objective is to effectively assign workers to the tasks received from the task generator module. The task assignment module collects the current location information of the workers regularly to assign tasks that are in the vicinity of the workers. Furthermore, the tasks are static in nature, contrasting the dynamic nature of workers, i.e., in a given time interval, the tasks stay the same, whereas the number of available workers and their geographical position varies. The task assignment module follows the batch-based assignment process that conducts the assignment of work-

### 3. Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap

Types of Task Assignment	Constraints			
	Maximum Tasks	Distance	Unassigned	Multiple Assignments
Road Segment Task Assignment				
Unique Road Segment Task Assignment				
Verifiable Road Segment Task Assignment				
Junction Task Assignment				
Unique Junction Task Assignment				
Verifiable Junction Task Assignment				

Table 3: Types of Task Assignment

ers in batches, instead of performing an assignment as soon as a worker is available. The tasks can be assigned to multiple workers, and each worker can be assigned a maximum threshold of tasks to perform. Once the tasks are assigned to the workers, the workers perform the tasks and send the tags for the tasks back to the task assignment module. The task assignment collects the responses sent by the workers and forwards it to the quality control module.

To discuss the different types of assignment types and optimization problems, we will first define a worker in our framework:

**Definition 3.** Worker:(Reproduced from Paper B [22] & Paper E [23])

A worker, denoted by  $w$ , is an amateur cartographer willing to perform an assigned task by travelling to the task’s road segment. Worker  $w$  has an identification number  $wid$  along with a location  $l$  that she reports to the assignment module. Additionally, Worker  $w$  specifies her preferences for accepting a task like the maximum number of tasks she’s willing to perform,  $maxT$  and the maximum distance  $d$  she is willing to travel for performing a task. A worker is defined as:  $w = \langle wid, l, maxT, d \rangle$ .

## Types of Task Assignments

In the context of improving OSM data quality, we have identified three ways of effectively enhancing the semantic tag information: increase the volume of tags, reduce the number of spatial entities with zero or minimal tags, and to ensure verifiability of collected tags. To support the identified semantic tag enhancement methods, we propose six types of assignment (see Table 3), namely:

- Valid Road Segment Task Assignment: involves assignment of a road segment task to the worker. It is a tuple of the form  $\langle w, rst \rangle$ , where  $w$  is assigned to  $rst$ , given the following constraints are satisfied. (Reproduced from Paper B [22] & Paper E [23])
  - Maximum Tasks constraint: The worker should not be assigned more road segment tasks than the  $w.maxT$  value.
  - Distance Constraint: The worker  $w$  should not be assigned tasks that are farther than  $d$ , i.e.,  $dist(w, rst) \leq w.d$
- Unique Road Segment Task Assignment: ensures that only unassigned road segment tasks are considered for the task assignment. It is a set of unique assignments of the form  $\langle w, rst \rangle$ , in which a road segment task  $rst$  is assigned to a worker  $w$ , while satisfying the previously described maximum tasks and distance constraints. Additionally, an unique road segment task assignment has to satisfy the following unassigned constraint: (Reproduced from Paper B [22])
  - Unassigned constraint: The task  $rst$  should be unassigned, i.e.,  $rst.assigned = 0$
- Verifiable Road Segment Task Assignment: ensures that the road segment task has at least two crowdsourced responses to infer verifiable tags. It is a set of assignments of the form  $\langle w, rst \rangle$ , in which a road segment task  $rst$  is assigned to a worker  $w$ , while satisfying the previously described maximum tasks and distance constraints. Additionally, a verifiable road segment task assignment has to satisfy the following multiple assignment constraint: (Reproduced from Paper B [22])
  - Multiple Assignment constraint: The task  $rst$  should be assigned to atleast one worker before, i.e.,  $rst.assigned \geq 1$
- Junction Task Assignment: involves assignment of a junction task to the worker. It is a tuple of the form  $\langle w, J \rangle$ , where worker  $w$  is assigned to a Junction  $J$ , given the maximum junction tasks and distance constraints are satisfied.

### 3. Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap

- Maximum Junction Tasks constraint: The worker should not be assigned more junction tasks than the  $w.maxT$  value.
- Distance Constraint: The worker  $w$  should not be assigned junction tasks that are farther than  $d$ , i.e.,  $dist(w, j) \leq w.d$
- Unique Junction Task Assignment: ensures that only unassigned junction tasks are considered for the task assignment. It is a tuple of the form  $\langle w, J \rangle$ , where worker  $w$  is assigned to a Junction  $J$ , given the unassigned, maximum junction tasks, and distance constraints are satisfied.
  - Unassigned constraint: The junction task  $J$  should be unassigned, i.e.,  $J.assigned = 0$
- Verifiable Junction Task Assignment: ensures that the road segment tasks grouped with the junction task has at least two crowdsourced responses to infer verifiable tags. It is a tuple of the form  $\langle w, J \rangle$ , where worker  $w$  is assigned to a Junction  $J$ , given the multiple assignments, maximum junction tasks, and distance constraints are satisfied.

#### Task Assignment optimization problems

The Task Assignment Module supports the following optimization problems corresponding to the different task assignment types:

- **Maximum Road Segment Task Assignment Problem (MRSTA):** an optimization problem to maximize the total number of valid road segment task assignments. For a better understanding of the MRSTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). The definition of the offline-MRSTA problem is given below: (The problem definitions below are reproduced from Paper B [22] & Paper E [23])  
**Offline Maximum Road Segment Task Assignment:** [23] Assume an input set of road segment tasks  $RST$ , a set of available workers  $W$ , and a set of valid task assignments  $VA$ . The offline Maximum Road Segment Task Assignment (Offline-MRSTA) problem is an optimization problem with the goal of maximizing the total number of road segment task assignments.

$$OffMR(RST, W) = \arg \max_{VA \in 2^{RST \times W} \text{ s.t. } VA \text{ is valid}} (|VA|)$$

, where  $OffMR(RST, W)$  is the maximal number of assignments.

**Batch-based Maximum Road Segment Task Assignment:** The Batch-

based Maximum Road Segment Task Assignment (Batch-MRSTA) problem is an optimization problem with a goal of maximizing the total number of valid road segment task assignments for continuous batches of workers  $W$  at each time instance  $i$ . At each time instance, the assignment problem for the batch of workers, along with the road segment tasks can be solved by reducing it to the Offline-MRSTA problem. Thus, the Batch-MRSTA can be solved by a series of Offline-MRSTA problems at each time instance.

$$OnMR(RST, W) = \arg \max_{i \in T} |OffMR(RST, W_i)|$$

, where  $W_i$  is the set of workers at time instance  $i$ , and  $OnMR(RST, W)$  is the optimal number of assignments. We assume, that the workers' arrive in batches at each time instance.

- **Maximum Unique Road Segment Task Assignment Problem (MURSTA):** is an optimization problem to maximize the total number of unique road segment task assignments. For a better understanding of the MURSTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). The definition of the offline-MURSTA problem is given below:(The problem definitions below are reproduced from Paper B [22])

**Offline Maximum Unique Road Segment Task Assignment:** Assume an input set of road segment tasks  $RST$ , a set of available workers  $W$ , and a set of unique task assignments  $UA$ . The offline Maximum Unique Road Segment Task Assignment (Offline-MURSTA) problem is an optimization problem with the goal of maximizing the total number of unique road segment task assignments.

$$OffMR(RST, W) = \arg \max_{UA \in 2^{RST \times W} \text{ and } UA \text{ is unique}} (UA)$$

, where  $OffMR(RST, W)$  is the set of maximal unique assignments.

**Batch-based Maximum Unique Road Segment Task Assignment:** The Batch-based Maximum Unique Road Segment Task Assignment problem (Batch-MURSTA) is an optimization problem with a goal of maximizing the total number of unique road segment task assignments for continuous batches of workers  $W_i$  at each time instance  $i$ . At each time instance, the assignment problem for the batch of workers, along with the updated road segment tasks  $RST_i$  can be solved by reducing it to



### 3. Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap

the Offline-MURSTA problem. Thus, the Batch-MURSTA can be solved by a series of Offline-MURSTA problems at each time instance.

$$OnMR(RST, W) = \sum_{i \in T} OffMR(RST_i, W_i)$$

, where  $OnMR(RST, W)$  is the set of maximal unique assignments,  $W_i$  is the set of workers, and  $RST_i$  is the set of updated road segment tasks at time instance  $i$ . We assume, that the workers' arrive in batches at each time instance. After every batch, the road segment task set will be updated to reflect the assignments.

- **Maximum Verifiable Road Segment Task Assignment Problem (MVRSTA):** is an optimization problem to maximize the total number of verifiable road segment task assignments. For a better understanding of the MVRSTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). The definition of the offline-MVRSTA problem is given below:(The problem definitions below are reproduced from Paper B [22])

**Offline Maximum Verifiable Road Segment Task Assignment:** Assume an input set of road segment tasks  $RST$ , a set of available workers  $W$ , and a set of verifiable task assignments  $VA$ . The offline Maximum Verifiable Road Segment Task Assignment (Offline-MVRSTA) problem is an optimization problem with the goal of maximizing the total number of verifiable road segment task assignments.

$$OffMRV(RST, W) = \arg \max_{VA \in 2^{RST \times W} \text{ and } VA \text{ is verifiable}} (VA)$$

, where  $OffMRV(RST, W)$  is the set of maximal verifiable assignments.

**Batch-based Maximum Verifiable Road Segment Task Assignment:** The Batch-based Maximum Verifiable Road Segment Task Assignment problem (Batch-MVRSTA) is an optimization problem with a goal of maximizing the total number of verifiable road segment task assignments for continuous batches of workers  $W_i$  at each time instance  $i$ . At each time instance, the assignment problem for the batch of workers, along with the updated road segment tasks  $RST_i$  can be solved by reducing it to the Offline-MVRSTA problem. Thus, the Batch-MVRSTA can be solved by a series of Offline-MVRSTA problems at each time instance.

$$OnMRV(RST, W) = \sum_{i \in T} OffMRV(RST_i, W_i)$$

, where  $OnMRV(RST, W)$  is the set of maximal verifiable assignments,  $W_i$  is the set of workers, and  $RST_i$  is the set of updated road segment tasks at time instance  $i$ . We assume, that the workers' arrive in batches at each time instance. After every batch, the road segment task set will be updated to reflect the assignments.

- **Maximum Junction Task Assignment Problem:** is an optimization problem to maximize the total number of junction task assignments and subsequently, their corresponding road segment task assignments.
- **Maximum Unique Junction Task Assignment Problem:** is an optimization problem to maximize the total number of unique junction tasks and the corresponding total number of unique road segment task assignments.
- **Maximum Verifiable Junction Task Assignment Problem:** is an optimization problem to maximize the total number of verifiable junction task assignment and subsequent verifiable road segment task assignments.

Example: Given a set of 7 road segment tasks and 2 batches of incoming workers with  $maxT=2$

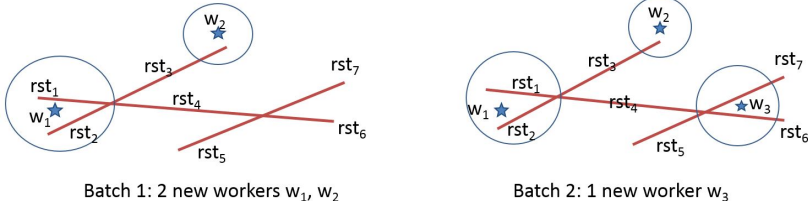


Fig. 12: Running Example

## Algorithms

We propose seven algorithms to solve the proposed task assignment optimization problems, two for the maximum junction problems and one each for the remaining optimization problems. For better understanding of the algorithms, we use a running example described in Fig. 12.

- **Baseline Greedy Algorithm (Gr):** solves the maximum road segment task assignment problem by reducing it to max flow problem for each batch, and solving it by using the Ford-Fulkerson algorithm [39].
- **Direct Assignment with Road Segments (DA-RS):** solves the maximum road segment task assignment problem, by assigning each available worker in the batch, a  $maxT$  road segment tasks that are nearest

### 3. Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap

---

#### ALGORITHM 1: DIRECT ASSIGNMENT WITH ROAD SEGMENTS (REPRODUCED FROM PAPER B [22] & PAPER E [23])

---

**Input:** A non-emptyset of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of road segment tasks  $RST_i$  at time instance  $i$ .  $OnMR(RST, W)$  is the optimal set of assignments before the time instance  $i$

**Output:** The Optimal set of assignments  $OnMR(RST, W + W_i)$  at time instance  $i$

```

1  $OnMR(RST, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3   foreach  $rst \in RST$  do
4     if  $dist(rst, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
5        $add\ assignment < w, rst > to OnMR(RST, W_i)$ 
6        $MaxT(w) \leftarrow MaxT(w) - 1$ 
7      $OnMR(RST, W + W_i) \leftarrow OnMR(RST, W) \cup OnMR(RST, W_i)$ 
8 return  $OnMR(RST, W + W_i)$ 

```

---



---

#### ALGORITHM 2: UNIQUE ASSIGNMENTS WITH ROAD SEGMENTS (U-RS) (REPRODUCED FROM PAPER B [22])

---

**Input:** A non-emptyset of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of road segment tasks  $RST$  at time instance  $i$ .  $OnMR(RST, W)$  is the optimal set of unique assignments before the time instance  $i$

**Output:** The Optimal set of unique assignments  $OnMR(RST_i, W + W_i)$  at time instance  $i$

```

1  $OnMR(RST, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3    $RST_w \leftarrow \emptyset$ 
4   foreach  $rst \in RST$  do
5     if  $dist(rst, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
6        $RST_w \leftarrow RST_w \cup rst$ 
7    $sortByAssignedAscending(RSt_w)$ 
8   foreach  $rst1 \in RST_w$  do
9      $rst1.assigned++$ 
10     $RST_i \leftarrow UpdateAssignment(RST)$ 
11     $MaxT(w) \leftarrow MaxT(w) - 1$ 
12     $OnMR(RST_i, W + W_i) \leftarrow OnMR(RST, W) \cup OnMR(RST_i, W_i)$ 
13 return  $OnMR(RST_i, W + W_i)$ 

```

---

to their reported location (See Fig. 13). The algorithm pseudocode is described in Algorithm 1.

- **Unique Assignments with Road Segments (U-RS):** solves the maximum unique road segment assignment problem, by ensuring the max-

---

**ALGORITHM 3: VERIFIABLE ASSIGNMENTS WITH ROAD SEGMENTS (V-RS) (REPRODUCED FROM PAPER B [22])**


---

**Input:** A non-emptyset of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of road segment tasks  $RST$  at time instance  $i$ .  $OnMRV(RST, W)$ , the optimal set of verifiable assignments before the time instance  $i$

**Output:** The Optimal set of verifiable assignments  $OnMRV(RST_i, W + W_i)$  at time instance  $i$

```

1  $OnMRV(RST, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3    $RST_w \leftarrow \emptyset$ 
4   foreach  $rst \in RST$  do
5     if  $dist(rst, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
6        $RST_w \leftarrow RST_w \cup rst$ 
7   foreach  $rst1 \in RST_w$  do
8     if  $rst1.assigned == 1$  then
9        $rst1.assigned ++$ 
10       $RST_i \leftarrow UpdateAssignment(RST)$ 
11       $MaxT(w) \leftarrow MaxT(w) - 1$ 
12       $OnMRV(RST_i, W + W_i) \leftarrow OnMRV(RST, W) \cup OnMRV(RST_i, W_i)$ 
13 if  $MaxT(w) > 0$  then
14    $sortByAssignedAscending(RST_w)$ 
15   foreach  $rst1 \in RST_w$  do
16      $rst1.assigned ++$ 
17      $RST_i \leftarrow UpdateAssignment(RST)$ 
18      $MaxT(w) \leftarrow MaxT(w) - 1$ 
19      $OnMRV(RST_i, W + W_i) \leftarrow OnMRV(RST, W) \cup OnMRV(RST_i, W_i)$ 
20 return  $OnMRV(RST_i, W + W_i)$ 

```

---

Batch 1: Assignment			Batch 2: Assignment		
Worker ID	Updated MaxT	Assigned Tasks	Worker ID	Updated MaxT	Assigned Tasks
$w_1$	0	$rst_1, rst_2$	$w_1$	0	
$w_2$	1	$rst_3$	$w_2$	1	
			$w_3$	0	$rst_6, rst_7$

Fig. 13: Running Example: solved by DA-RS

imization of unique road segments assigned to workers. The road segment tasks are tracked based on their previous assignment history, and the unassigned road segment tasks are prioritized for assignment to workers. The algorithm pseudocode is described in Algorithm 2.

- **Verifiable Assignments with Road Segments (V-RS):** solves the maximum verifiable road segment assignment problem, by maximizing the number of road segment tasks that are assigned to at least two workers

### 3. Push-based Spatial Crowdsourcing for enriching semantic tags in OpenStreetMap

while satisfying the maximum tasks and the distance constraints. The algorithm pseudocode is described in Algorithm 3.

- **Max-flow based Task Grouping (TG):** is a maximum flow-based approach for solving the maximum junction task assignment problem, by creating a flow network graph of junctions and workers and employing the Ford-Fulkerson algorithm [39] (See Fig. 14). The algorithm pseudocode is described in Algorithm 4.

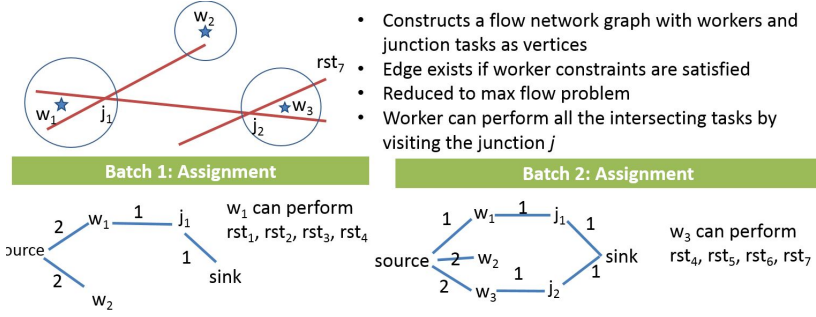


Fig. 14: Running Example: solved by TG

- **Direct Assignment with Junctions (DA-J):** solves the maximum junction task assignment problem, by assigning each available worker in the batch, a  $maxT$  junction tasks that are nearest to their reported location (See Fig. 15).

Worker ID	Updated MaxT	Assigned Junction Tasks	Tasks	Batch
$w_1$	1	$j_1$	$rst_1, rst_2, rst_3, rst_4$	1
$w_2$	2			1
$w_3$	1	$j_2$	$rst_4, rst_5, rst_6, rst_7$	2

Fig. 15: Running Example: solved by DA-J

- **Unique Assignments with Junctions (U-J):** solves the maximum unique junction assignment problem, by ensuring the maximization of unique junctions assigned to workers. The junction tasks are tracked based on their previous assignment history, and the unassigned junction tasks are prioritized for assignment to workers.
- **Verifiable Assignments with Junctions (V-J):** solves the maximum verifiable junction assignment problem, by maximizing the number of junc-

tion tasks that are assigned to at least two workers while satisfying the maximum tasks and the distance constraints.

---

**ALGORITHM 4:** MAX FLOW-BASED TASK GROUPING (TG) (REPRODUCED FROM PAPER B [22]) & PAPER E [23]

---

**Input:** A set of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of junction tasks  $J$  at time instance  $i$ .  $OnMR(J, W)$  is the optimal set of assignments before the time instance  $i$ .

**Output:** The Optimal set of assignments  $OnMR(J, W + W_i)$  at time instance  $i$

```

1  $OnMR(J, W + W_i) \leftarrow NULL$ 
2  $capacity \leftarrow 1$ 
3 if  $W_i \neq \emptyset$  then
4    $V \leftarrow W_i \cup J$ 
5   foreach  $w \in W_i$  do
6      $E.addEdge(V_0, w, maxT, 0)$ 
7     foreach  $j \in J$  do
8        $E.addEdge(j, V_{|V|+1}, capacity, 0)$ 
9       if  $dist(j, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
10         $E.addEdge(w, j, capacity, dist(j, w))$ 
11   construct flow network graph  $G(V, E)$ 
12   calculate the min travel cost maximum flow
13   find the assignment  $OnMR(J, W_i)$ 
14   Update maxT values of  $W_i$ 
15    $OnMR(J, W + W_i) \leftarrow OnMR(J, W) \cup OnMR(J, W_i)$ 
16 return  $OnMR(J, W + W_i)$ 

```

---

### 3.5 Quality Control

In this section, we briefly describe the out of focus Quality Control module. The Quality Control module aggregates the collected crowdsourced responses (tag values) from the task assignment module to infer the true tag values. Verifiability is very important in OSM, and it can be achieved “if and only if independent users observing the same feature would make the same observation every time” [69]. The quality control module utilizes the wisdom of the crowd through the noniterative aggregation method, Majority Voting [41], to infer the verifiable truth from the collected crowdsourced responses. As mentioned previously, the task assignment module assigns the same task to multiple workers. It considers the values of the tags for the task that the majority agrees as the verifiable truth. Other noniterative aggregation methods can be used instead of Majority Voting [41], like Honeypot [42], and Expert Label Injected Crowd Estimation(ELICE) [38]. Furthermore, we

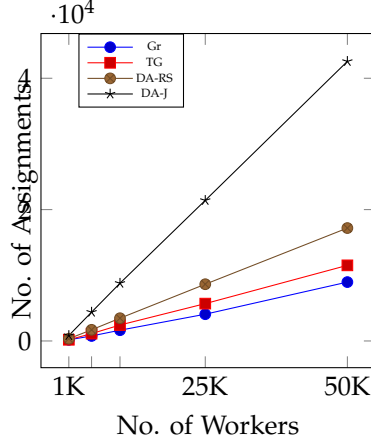
can conduct qualification tests for the workers to ascertain their expertise in providing tag values to OSM spatial entities.

## 3.6 Experimental Evaluation of Assignment Algorithms

To evaluate the proposed assignment algorithms, we have considered the study area of Aalborg, Denmark. First, the Aalborg road network is extracted from the OSM and converted as road segment tasks and junction tasks. Accordingly, we generated 17503 road segment tasks and 7203 junction tasks. Subsequently, we have customized the check-ins dataset collected by [31] in the region of Aalborg, Denmark, for the workers' dataset. Furthermore, synthetic workers are generated to evaluate the scalability of the algorithms. The proposed algorithms are compared against the greedy max-flow based baseline task assignment approach.

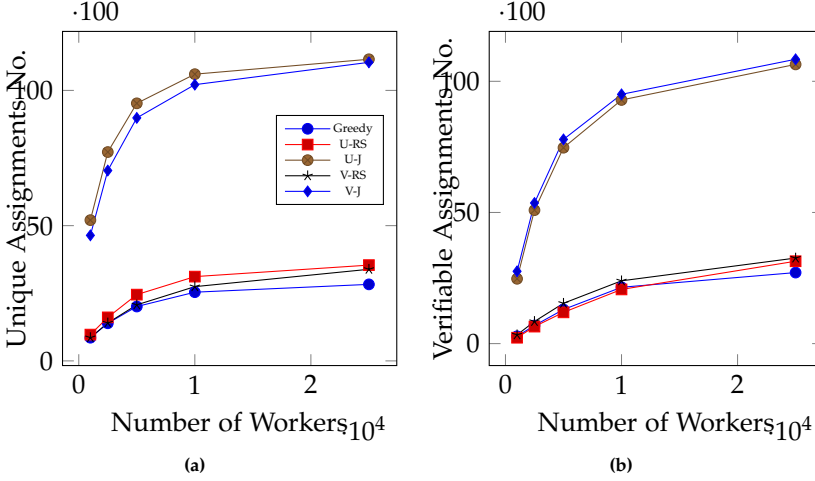
Fig. 16a illustrates the effect of varying the number of workers on the number of valid road segment assignment tasks. Algorithms, *Direct Assignment for Road Segments (DA-RS)* and *Direct Assignment for Junctions (DA-J)* outperforms the baseline *Greedy (Gr)* algorithm by two and five times, respectively. Mainly, *DA-J* performs the best with five times more valid road segment assignments than the baseline approach. Similarly, *DA-RS* results in two times more valid road segment assignments than the baseline approach. The reason, junctions-based algorithms dominate is due to the access to more number of road segments at the junctions.

Fig. 17a and 17b showcases the effect of varying worker size on the number of unique and verifiable road segment task assignments, respectively. Junctions-based algorithms, *Unique Assignments with Junctions (U-J)* and *Verifiable assignments with Junctions (V-J)*, results in the highest number of assigned unique road segments and the highest number of verifiable task assignments. Especially, *U-J* and *V-J* performs the best with six times as many and five times as many unique road segment assignments than the greedy baseline algorithm. Similarly, *U-J* and *V-J* results in at least seven times as many more verifiable road segment assignments than the baseline. *U-J* performs better than *V-J* in case of unique assignments, and vice versa in case of verifiable assignments. It can also be observed that the junctions-based algorithms, *U-J* and *V-J* yields converge as the workers count increases. The reason could be pointed to the limited number of tasks that can be performed by the workers. Since the tasks are limited and the workers are increasing, the number of unique assignments and the number of verifiable assignments also converges. Moreover, the values of *U-J* and *V-J* has a steep jump during the initial increase in workers. However, as the number of workers crosses 10K, there is just a marginal increase in the number of unique assignments. This phenomenon can again be explained by the limited nature of road segment tasks. ((Reproduced from Paper B [22]))



(a)

**Fig. 16:** Effect of varying worker size on number of valid road segment task assignments (Reproduced from Paper B [22] & Paper E [23]).



(a)

(b)

**Fig. 17:** Effect of varying worker size on :a. Number of unique road segment task assignments. b. Number of verifiable road segment task assignments (Reproduced from Paper B [22]).

### 3.7 Discussion

The push-based SC for OSM framework works on the principle of pushing the OSM spatial entities to the amateur cartographers/ workers, who are willing to add the tags voluntarily. We have conformed to the current OSM contributions model, which is of altruistic in nature. Instead of waiting for the workers to map the OSM areas remotely, we actively push the spatial entities to the nearby worker to collect more semantic tags. By encouraging active server-based task assignment, the framework encourages more worker participation. However, to further improve workers participation, the push-



#### 4. Spatial Entity Linkage with the Aid of Spatial Crowdsourcing

based SC for OSM framework can also offer monetary rewards as incentives for performing the tasks. It would be challenging to optimize the budgets spent on incentives, considering the worker's reputation and expertise. Especially, in a privacy-enabled environment, where the worker profiles will be hidden to the push-based SC for OSM framework, making it more difficult to calculate the incentives for the workers.

[24] observed that the existing SC frameworks with incentive mechanisms are still limited to the workers with previous knowledge about SC concepts. In our case as well, the advanced SC framework is limited to workers who had previously contributed to OSM as amateur cartographers. However, there is a limited number of amateur cartographers with OSM mapping experience. Consequently, we have to attract the general public to improve the number of OSM spatial entities' tags. However, it is challenging to attract the general public to participate in the push-based SC for OSM framework. To address this challenge, we target a new group of workers using public transport service for assigning the SC tasks (discussed in detail in Section 5).

Furthermore, to enrich a spatial crowdsourcing-based dataset like OSM, we can augment the resource-intensive spatial crowdsourcing tag collection approach by identifying the corresponding entities in other data sources like *Google*, *Flickr*, *Krak*, etc. Solving the spatial entity linkage problem can resolve the OSM affiliated spatial entities across data sources. The identified corresponding entities could provide contextual tag information without the need for spatial crowdsourcing. We employ a hybrid machine learning-aided multi-modal SC approach for solving the spatial entity linkage problem (discussed in detail in Section 4).

## 4 Spatial Entity Linkage with the Aid of Spatial Crowdsourcing

### 4.1 Motivation and Problem Statement

With the advent of location-based sources, it is imperative to infer whether the spatial entities across sources within close geographical proximity refer to the same real-world entity or two different entities. For example, a spatial entity with the name "Naesbyhoved skov" is located at (55.4119,10.3775) with associated keywords "A-la-carte", "Dine-in" in source A, and a spatial entity with name "Restaurant Næsbyhoved skov" is located at (55.412,10.376) with associated keywords "Take-away", "restaurant" in source B refers to the same physical entity. However, it is not straightforward to determine that these two spatial entities in different sources refer to the same physical entity as the attribute information is different. We need to develop a solution to automatically link the spatial entities that refer to the same physical entities across

different sources to provide richer semantic spatial entities. This problem is defined as *Spatial Entity Linkage Problem* [32].

Currently, the spatial entity linkage problem is solved by different automatic labeling (AL) methods [6, 32, 33, 47]. However, they are dependent on the quality of the attributes of the spatial entities and potentially mislabels many spatial entity pairs. For instance, the automatic labeling method proposed in [32] could only deliver a *F-measure* value of 0.71, *precision* of 0.86 and *recall* of 0.61, leaving a lot of scope for improvement. Ideally, the Spatial Entity Linkage problem can be resolved with multi-modal SC by exploiting workers' wisdom. However, it would be expensive to crowdsource all the spatial entity pairs given a limited workers resource. Therefore, there is a need to develop a hybrid solution that employs automatic labeling for the entire spatial entity pairs set and multi-modal SC techniques for the entity pair cases where the attributes are inadequate for enabling automatic labeling. Furthermore, to improve the effectiveness and reduce SC's cost, a machine learning-aided multi-modal SC approach needs to be developed.

## 4.2 Preliminaries and Problem Definition

**Definition 4.** (Spatial Entity): A *Spatial Entity*, denoted by  $s$ , is a real world place like a shop, or a public utility center, etc. It has a geographical location point  $s.p$ , and a set of different attributes  $\{s.a_1, s.a_2, \dots, s.a_n\}$  like address of the entity  $s.a_i$  (Reproduced from Paper C [30]).

We define the *Spatial Entity Linkage task* in line with the *Spatial Entity Linkage problem* [32].

**Definition 5.** (Spatial Entity Linkage task): A *Spatial Entity Linkage (SEL) task*, denoted by  $t$ , represents a classification task with the objective to determine whether a pair of spatial entities  $s_1$  and  $s_2$ , represent the same physical entity.  $t = \langle s_1, s_2, l \rangle$ , where the label  $l$  represents whether the entities are a match (Reproduced from Paper C [30]).

$$l = \begin{cases} l^+ & \text{if } s_1 \text{ and } s_2 \text{ belong to the same physical spatial entity} \\ l^- & \text{otherwise} \end{cases}$$

The pairs labeled as  $l^+$  are referred as the *positive class* and the pairs labeled as  $l^-$  are referred as the *negative class*. In our context, we define *True Positives* (TP) as actual positive pairs labeled as positives, *True Negatives* (TN) as actual negative pairs labeled as negatives, *False Positives* (FP) as actual negative pairs labeled as positives, and *False Negatives* (FN) as actual positive pairs labeled as negatives. The *precision* of a classifier is  $precision = \frac{TP}{TP+FP}$  and the *recall* is  $recall = \frac{TP}{TP+FN}$ . In order to measure the effectiveness of the

#### 4. Spatial Entity Linkage with the Aid of Spatial Crowdsourcing

classifier, we use the *F-measure* ( $F1$ ) =  $2 \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$  (Reproduced from Paper C [30]).

When an SEL task is solved by an AL method, it will not cost anything. However, the *F-measure* of the AL results is dependent on the quality of the AL. Whereas in the case of SC, when an SEL task is resolved, we assume that the label provided by the crowd is always correct, i.e., the ground truth. Consequently, the F-measure of the SC results is a perfect 1.0. However, a reward needs to be paid to the crowd, resulting in extra cost (Reproduced from Paper C [30]).

To determine whether to use only AL or only SC or a combination of both is dependent on the given reward budget. Let us denote by  $R$  the given *reward budget* for solving a set of SEL tasks and by  $R^*$  the required *reward budget* to solve all the SEL tasks (Reproduced from Paper C [30]).

$$\begin{cases} R \geq R^* & \text{Only SC} \\ R = 0 & \text{Only AL} \\ 0 < R < R^* & \text{Combination of AL and SC} \end{cases}$$

If the given reward budget is more than the budget required to solve all the SEL tasks ( $R \geq R^*$ ), then we can solve the *Spatial Entity Linkage* problem by just using SC. If there no given budget ( $R = 0$ ), the *SEL* problem has rely just on AL. In the case of limited given budget ( $0 < R < R^*$ ), we have to use a combination of AL and SC. Finding the best trade-off between AL and SC leads to an optimization problem, the *Spatial Entity Linkage with the aid of Spatial Crowdsourcing* problem (Reproduced from Paper C [30]). The problem is defined as:

**Definition 6.** (Spatial Entity Linkage with the aid of Spatial Crowdsourcing): Given a reward budget  $R$  and a set of SEL tasks  $SEL_{ALL}$ , Spatial Entity Linkage with the aid of Spatial Crowdsourcing (SELSC) problem is an optimization problem to distribute SEL tasks between AL and SC, with an objective of maximizing the F-measure of the resultant labeled SEL tasks (Reproduced from Paper C [30]).

$$\arg \max_{SEL_{AL} \sqcup SEL_{SC} = SEL_{ALL}} F1(SEL_{AL}, SEL_{SC}) \quad (4)$$

, where  $SEL_{AL}$ ,  $SEL_{SC}$  are the sets of SEL tasks assigned to AL and SC, respectively.

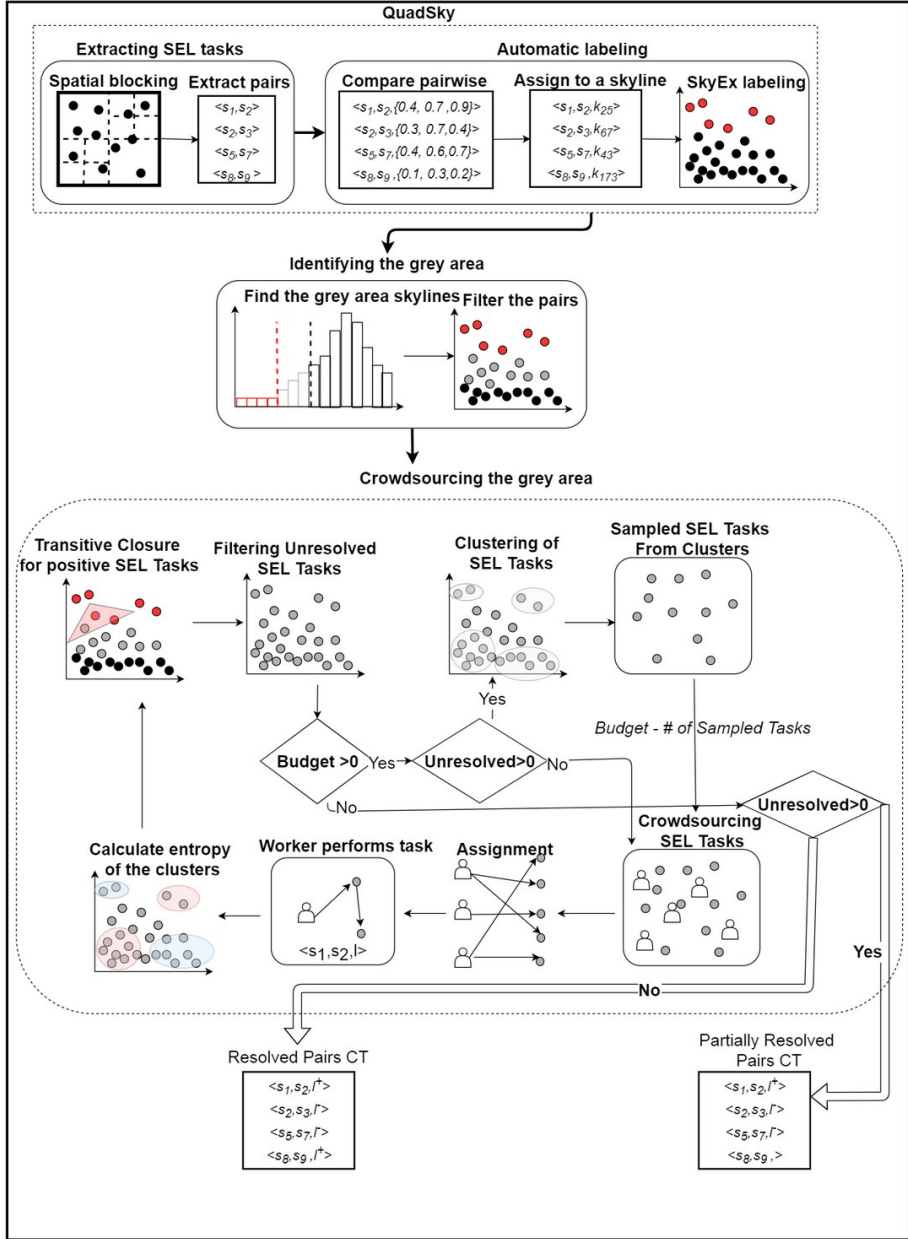


Fig. 18: SkyCrowd Solution (Reproduced from Paper C [30])

### 4.3 Skycrowd Solution

This section describes our proposed solution to the SELSC problem, *SkyCrowd*. *SkyCrowd* resolves the SELSC problem by employing *Skyline-based*

#### 4. Spatial Entity Linkage with the Aid of Spatial Crowdsourcing

*Automatic Labeling (AL)* [32] and *Machine learning-aided Spatial Crowdsourcing (SC)* techniques. The *SkyCrowd* workflow (See Fig. 18) starts by taking a set of spatial entities  $S$  as input and extracting a set of SEL tasks with spatial entity pairs. The extracted SEL tasks are automatically labeled utilizing the *Skyline-based Automatic Labeling (AL)* technique and provide labels for all the SEL tasks. However, the skyline-based AL technique mislabels some SEL due to misleading attribute information. Skycrowd identifies the potentially mislabeled tasks and labels them as tasks under the grey area. The SEL tasks that fall under a grey area are resolved using the SC-based active learning approach. Finally, *Skycrowd* outputs resolved/ partially resolved SEL tasks set  $CT$  depending on the reward budget. The different phases in the *SkyCrowd* workflow are elaborated in detail below(See Fig. 18):

##### Extracting SEL Tasks

Given a set of entities  $S$ , the *Skycrowd* solution extracts the SEL tasks containing entity pairs that require resolution. To reduce the number of spatial entity comparison pairs, a spatial blocking technique, *QuadFlex* [32], is employed to extract the SEL tasks. The technique involves inserting the spatial entities into a quadtree-like structure. The *QuadFlex* algorithm is detailed in [32]. These leaves of the inserted tree are the spatial blocks of spatial entities respecting the spatial proximity and area's density. Skycrowd extracts the pairs within each spatial block and transforms them into SEL tasks of pairs of spatial entities  $\langle s_i, s_j \rangle$  with an empty label.

##### Automatic Labeling of SEL Tasks

*Skycrowd* employs a skyline-based automatic labeling technique, [32] to resolve the label of the extracted SEL tasks. The SEL tasks' pairs are compared on their attribute values to estimate their utility. We use Levenshtein distance [43] to compare names, and Wu&Palmer Similarity measure (wup) [71] to compare the addresses. The similarities between attributes are assessed based on the Pareto Optimal law [7] and the *SkyEx* algorithm used in [32]. Based on the utility, tasks are assigned to the skyline or the Pareto optimal frontier. Lower utility values result in assignment to higher skylines (less chance for a match) and vice versa. Finally, the cut-off skyline level  $k$  that signifies clear separation between positive and negative classes is fixed by the *SkyEx* algorithm [32], based on the best trade-off between precision and recall values of the experiments. Furthermore, the *SkyEx* algorithm labels the SEL tasks based on the assigned skylines and the fixed cut-off skyline  $k$ . The tasks belonging to a skyline of  $k$  level or less will be labeled positive  $l^+$  and the remaining will be labeled negative  $l^-$ .

## Identifying the Grey Area

Since labeling the pairs is based on the skyline, we assume that the skylines can help detect clear separation between the positive and negative classes. However, the  $k$  skyline fixed by the SkyEx algorithm could result in mislabeling of the tasks fall near the cut-off  $k^{th}$  skyline. Skycrowd proposes a technique for identifying the SEL tasks that are more likely to be mislabeled based on their proximity to  $k^{th}$  skyline. The technique identifies a skyline  $k^*$ , such that all the tasks with at most skyline  $k^*$  are labeled as positive. Furthermore, the tasks assigned to a skyline between  $(k^*, k]$  are labeled as negative and are referred to as the grey area.

For example, Fig. 19 shows the procedure for fixing  $k$  in the left figure. We select the value of  $k$  that yields the highest *F-measure*. After that (green line), we start loosing in precision and *F-measure* falls. From the green line to the first blue line (right figure), is identified as the grey area. Algorithm 5 describes the procedure for identifying the grey area tasks. We find the tasks that lie in the skylines between  $k^*$  (skyline that yields the highest *F-measure*) and  $k$  of SkyEx algorithm of the previous phase. For the grey area, we can decide to crowdsource all, or use the SC-based active learning approach (see the next section). Thus, we overwrite the labels for the grey area and finally, return the labeled pairs (Reproduced from Paper C [30]).

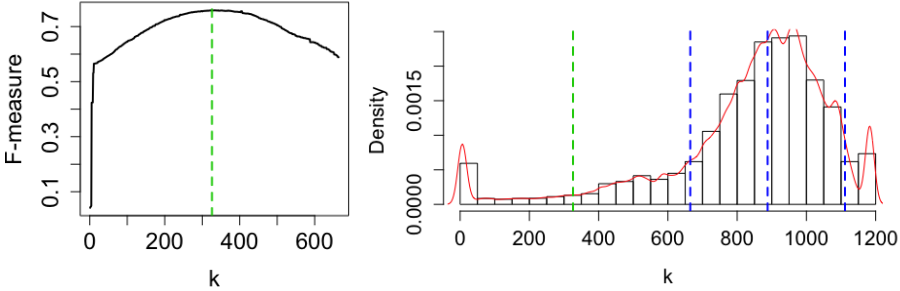


Fig. 19: Identifying the Grey Area (Reproduced from Paper C [30])

---

**ALGORITHM 5:** Identifying SEL tasks in Grey Area (Reproduced from Paper C [30])

---

**Input:** A set of SEL tasks and their skylines  $\{\langle s_i, s_j, ks \rangle\}$ , SkyEx Algorithm fixed skyline  $k$  and a reward budget  $R$

**Output:** A set of grey area SEL tasks  $\{\langle s_i, s_j, l \rangle\}$

- 1 Find  $k^*$  that yields the highest *F-measure*
  - 2 Label  $\{\langle s_i, s_j, ks \rangle | ks > k^* \text{ and } ks \leq k\}$  as negative and added to grey area SEL task set
  - 3 **return**  $\{\langle s_i, s_j, l \rangle\}$
-

#### 4. Spatial Entity Linkage with the Aid of Spatial Crowdsourcing

---

##### ALGORITHM 6: CROWDSOURCING THE GREY AREA (REPRODUCED FROM PAPER C [30])

---

**Input:** A set of grey area tasks  $S$  and their skylines  $\{\langle s_i, s_j, k \rangle\}$ , a set of total SEL tasks  $TS$ , a set of workers  $W$ , entropy Threshold  $e_T$ , reward Budget  $R$ , Cochran's formula constants  $Co1 = 384.16$  and  $Co2 = 383.16$

**Output:** A set of resolved tasks  $CT$  with their label  $\{\langle s_i, s_j, l \rangle\}$

```

1 Inferred Tasks  $IS \leftarrow \emptyset$ 
2 Unresolved Tasks  $US \leftarrow \emptyset$ 
3 Crowdsourced Tasks  $crowdTasks \leftarrow \emptyset$ 
4 while  $R > 0$  and  $US \neq \emptyset$  do
5    $S_t \leftarrow transitiveClosureSearch(TS)$ 
6    $S \leftarrow S \setminus S_t$ 
7    $US \leftarrow S$ 
8    $Clusters\ C \leftarrow DBSCANClustering(US)$ 
9   foreach  $c \in C$  do
10     $sampleSize \leftarrow (Co1 + c.size) / (Co2 + c.size)$ 
11     $crowdTasks \leftarrow performCrowdsourcing(sampleSize, c, W, R)$ 
12     $US \leftarrow US \setminus crowdTasks$ 
13     $R = R - \sum_{i=1}^{crowdTasks.size} crowdTasks[i].r$ 
14     $pos \leftarrow (no\ of\ positives\ in\ crowdTasks) / SampleSize$ 
15     $e \leftarrow -(pos * \log[2](pos)) - (1 - pos)(\log[2](1 - pos))$ 
16    if  $e < e_T$  then
17       $US \leftarrow US \setminus c$ 
18       $IS \leftarrow IS \cup c$ 
19      if  $pos > 0.5$  then
20        Label all remaining tasks in cluster  $c$  as  $l^+$ 
21      else
22        Label all remaining tasks in cluster  $c$  as  $l^-$ 
23  Resolved Tasks  $CT \leftarrow CT \cup crowdTasks \cup IS$ 
24  Update Total Tasks Set  $TS \leftarrow TS \cup crowdTasks \cup IS$ 
25   $S \leftarrow S \setminus crowdTasks \setminus IS$ 
26 if  $US = \emptyset$  and  $T > 0$  then
27   Label the inferred SEL tasks with crowdsourcing
28    $crowdTasks \leftarrow performCrowdsourcing(T, IS, W)$ 
29    $CT \leftarrow CT \cup crowdTasks$ 
29 return  $CT$ 

```

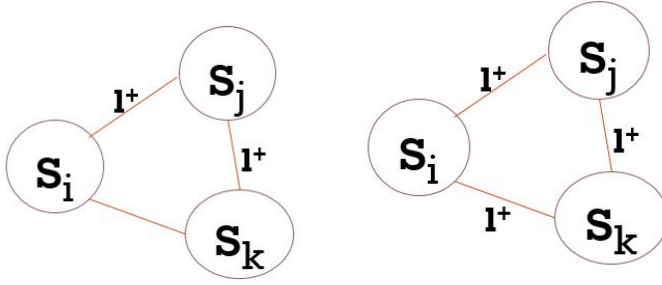
---

#### Crowdsourcing the Grey Area

SC will resolve the SEL tasks identified as part of the grey area. Workers are assigned SEL tasks to maximize the F1 gain, given a reward budget. Given the cost of SC, it might not be feasible to crowdsource all the SEL tasks in the grey area, i.e.,  $R < R_{grey}$ . Therefore, we propose optimization strategies facilitating active learning in SC to reduce costs. As illustrated in Fig. 18, the

proposed SC-based active learning approach consists of four iterative steps, checking for the transitive closure of positive SEL tasks, clustering of the unresolved grey area tasks, crowdsourcing samples of the clusters, and inferring the label of the cluster based on the entropy of crowdsourced sample. Furthermore, we propose proximity-based and past visit-based assignment algorithm to further reduce the SC cost.

The four iterative steps of the approach terminate when the budget exhausts ( $R = 0$ ), or when all the tasks are resolved. The intuition regarding the inferred labels of clusters based on entropy is to resolve tasks without spending any budget. If there is any budget left after the auto-inference of labels, the unused budget will be used to crowdsource the inferred tasks to gain more accuracy (Lines 26-28 of Algorithm 6). Based on the reward budget  $R$ , the SC-based active learning approach has two outcomes. First, if there is insufficient budget, then the outcome would consist of partially resolved tasks. Second, if there is sufficient budget, then the outcome consists of fully resolved tasks (Reproduced from Paper C [30]).



(a) Triangle of SEL tasks (b) Deduced label  
Fig. 20: Transitive Closure Example (Reproduced from Paper C [30])

**Transitive Closure of Unresolved Grey Area Tasks** First, we check the unresolved grey area tasks with regard to the transitive closure property (Line 5 of Algorithm 6). Particularly, the triangles that contain the positive label tasks from the left tail (or from previous iterations) and the unresolved grey tasks are compared for the transitive closure. For instance, given a triangle of three SEL tasks with their respective labels,  $\langle \langle s_i, s_j, l^+ \rangle, \langle s_j, s_k, l^+ \rangle, \langle s_k, s_i, - \rangle \rangle$ , the label of the task  $\langle s_k, s_i, - \rangle$  can be deduced to the positive label  $l^+$ , by the transitive closure property (See Fig. 20). The tasks labeled after checking the transitive closure property are removed from the unresolved task set (Line 6 and 7 of Algorithm 6), and the remaining unresolved tasks are clustered in the next step. After inferring the clusters' labels based on the crowdsourced samples in the fourth step, we again check for the transitive closure property, including the resolved tasks in the previous iteration (Reproduced from



#### 4. Spatial Entity Linkage with the Aid of Spatial Crowdsourcing

---

**ALGORITHM 7:** Task Assignment Algorithm based on past visits (Reproduced from Paper C [30])

---

**Input:** A set of sample tasks  $SEL$ , workers  $W$  with base reward  $C$ , and reward budget  $R$

**Output:** A set of assignments  $VA$  with reward  $\langle w, sel, r \rangle$

```

1 Probability  $p \leftarrow \text{random}(0,1)$ 
2 foreach  $sel \in SEL$  do
3   if  $R > 0$  then
4     Find workers who visited task location  $l$  in the past
4      $ExperiencedWorkers \leftarrow \text{findPastVisitWorkers}(l)$ 
5     Find workers who visited close to the task location  $l$  in the past
5      $CloseWorkers \leftarrow \text{findPastVicinityWorkers}(l)$ 
6     if  $ExperiencedWorkers.size > 0$  then
7       if  $R > C$  then
8          $R \leftarrow R - C$ 
9          $VA \leftarrow VA \cup \langle ExperiencedWorkers[0], sel, C \rangle$ 
10    else if  $CloseWorkers.size > 0$  then
11      if  $p > 0.5$  then
12        if  $R > C$  then
13           $R \leftarrow R - C$ 
14           $VA \leftarrow VA \cup \langle CloseWorkers[0], sel, C \rangle$ 
15      else
16        Find closest worker to the task location  $w \leftarrow \text{closestWorker}(W, sel)$ 
17        reward  $r \leftarrow \text{serviceRate} * ((\text{dist}(w, sel.s_{\{1/2\}}) +$ 
17           $\text{dist}(sel.s_1, sel.s_2)) / \text{workerSpeed}) + C$ 
18        if  $R > r$  then
19           $R \leftarrow R - r$ 
20           $VA \leftarrow VA \cup \langle w, sel, r \rangle$ 
21 return  $VA$ 

```

---

Paper C [30]).

**Clustering of Unresolved Grey Area Tasks** After filtering the tasks resolved by transitive closure property, we employ a density-based clustering algorithm, DBSCAN [17], for finding clusters consisting of similar pairs among the remaining unresolved tasks (Line 8 of Algorithm 6). We chose DBSCAN as it is robust to outliers and noise. The tasks that do not get assigned to a cluster (the noise) are labeled through crowdsourcing (See Crowdsourcing the grey area phase of Fig. 18). Then, we pick a sample from each cluster for crowdsourcing, and the sample size is calculated by following the *Cochran formula* [70]. We consider 95 % confidence and 0.5 as the estimated proportion of the cluster tasks with positive labels. Based on the considerations, we calculate the sample size for a given cluster  $c$  as follows:

$$sampleSize = (384.16 + c.size) / (383.16 + c.size)$$

From the given cluster  $c$ , we select the  $sampleSize$  tasks based on the skyline. The smaller the skyline of the task  $k$ , the better the task is to be a positive label. We are less confident about their label for the tasks that fall into deeper skylines in the grey area. Consequently, we need to select tasks within deeper skyline levels for the crowdsourcing samples, thus leading to the next step of crowdsourcing the samples from the clusters (Reproduced from Paper C [30]).

**Crowdsourcing the Cluster Samples** The selected samples from each cluster are assigned to workers, and the worker’s responses regarding the task’s label are collected. Given the limited number of available workers, we have to maximize the number of tasks that can be solved. To proceed further, let us define a worker in SC:

**Definition 7.** (Worker): A worker, denoted by  $w$ , is a person willing to perform an assigned task by travelling to both the entities’ locations. Worker  $w$  has the id of worker  $wID$ , a set of visited locations and the time of visit at the location  $\{[visitedLocation, visitTime], \dots\}$  (Reproduced from Paper C [30]).

$$w = \langle wID, \{[visitedLocation, visitTime], \dots\} \rangle$$

We believe that a worker with experience of visiting the task location or being in the vicinity of the task location is better suited for completing the task than a worker who has no experience with the task location. Furthermore, a worker with a past visit experience regarding the task location is not required to travel to the task location again to complete the task, thereby saving the travel cost aspect of the reward budget. To exploit the workers’ past visit information, we propose a past-visits based task assignment algorithm (See

#### 4. Spatial Entity Linkage with the Aid of Spatial Crowdsourcing

Algorithm 7). We define the valid past visit-based task assignment set to enable assignments based on the workers' past visits.

**Definition 8.** (Valid Past Visit-based Task Assignment Set): Given a set of workers  $W = \{w_1, w_2, \dots\}$  and a set of spatial entity matching tasks  $T = \{t_1, t_2, \dots\}$ , a Valid Past Visit-based Task Assignment (VPVTA), is a 3-tuple of form  $\langle w, t, r \rangle$ , where  $w$  is assigned to  $t$  with an associated reward  $r$ , and the worker should have previously visited either of the locations of the task's spatial entities.

$$t.l_1/t.l_2 \in \{w.visitedLocation_1, w.visitedLocation_2, \dots\}$$

For a given task  $t$ , if there is no possibility of a past visit-based task assignment, the algorithm checks whether any worker was in the vicinity of the task (around 50 meters) during their past visits and assigns the task with a 50% chance (Lines 5 and 10-14 of Algorithm 7). If there are no past worker visits in the vicinity of the task, then the task will be assigned to the closest worker to the task location. Post crowdsourcing of the SEL tasks in the cluster samples, the crowdsourced labels are analyzed for inferring the cluster labels.

**Inferring the Cluster Labels based on Crowdsourced Samples** The collected crowdsourced labels are aggregated based on the cluster, and the ratio of positive labels in the crowdsourced tasks and the corresponding entropy are calculated (Steps 14 and 15 of Algorithm 6). The cluster's label is estimated based on the relation between a given entropy threshold and the crowdsourced sample's entropy. We assume that the cluster's majority label would be same as the crowdsourced sample's majority label. If the entropy is less than the entropy threshold, then the cluster's label will be the cluster sample's popular label, and all the non-crowdsourced tasks in the cluster will be labeled as the cluster's label (Lines 16 - 22 of Algorithm 6). However, if the entropy is greater than the entropy threshold, the cluster's label is not estimated.

#### 4.4 Experimental Evaluation

To evaluate the proposed *Skycrowd* solution, we perform experiments on a real dataset, collected as in [31]. The dataset had 75,541 spatial entities, which originate from four sources, 51.50% from Google Places, 46.22% from Krak ([www.krak.dk](http://www.krak.dk)), 0.03% from Foursquare, and 2.23% from Yelp. The pairs of spatial entities which are at most 50 m apart are compared pairwise and labeled with the *SkyEx* algorithm [32]. Thus, we obtain 293,833 labeled pairs of spatial entities. We utilize the check-ins dataset from [31] for simulating

crowdsourcing workers. The workers’ dataset has 19980 workers available for assignment. For simplicity purposes, we assume the reward for each task as one \$, irrespective of the distance travelled by workers for completing the task. Therefore, a reward budget of 33155 \$ would resolve 33155 SEL tasks (100% of the grey area) through crowdsourcing (Reproduced from Paper C [30]).

We evaluate our proposed skycrowd solution in terms of *precision*, *recall* and *F-measure*, where  $F\text{-measure} (F1) = 2 \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$ . The crowdsourcing replies are simulated as if consulting an oracle and revealing the label. The skycrowd solution is compared against the results of crowdsourcing all the 293,833 tasks (Full), automatic labeling of all the tasks (Automatic), and crowdsourcing all the 33,155 tasks in the grey area (Hybrid-Full). Furthermore, we evaluate the effect of varying the budget and varying the entropy threshold on the *F-measure*, the average distance travelled by worker, and the number of inferred tasks (Reproduced from Paper C [30]).

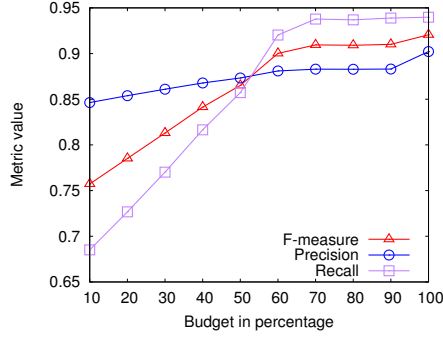
	Automatic	Skycrowd	Hybrid Full	Full
<b>Precision</b>	0.86	0.88	0.90	1
<b>Recall</b>	0.61	0.94	0.94	1
<b>F-measure</b>	0.71	0.91	0.92	1
<b>Budget</b>	0	22100	33,155	293,833

**Table 4:** Comparing Automatic vs Skycrowd vs Hybrid-Full vs Full Crowdsourcing in terms of budget and different metrics (Reproduced from Paper C [30])

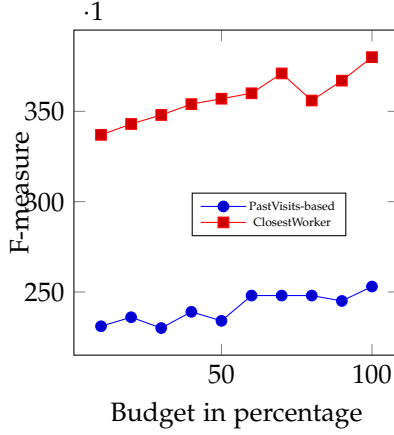
**Skycrowd vs Automatic vs Hybrid Full vs Full Crowdsourcing** The *Skycrowd* solution significantly improves the results of the automatic labeling, from an *F-measure* of 0.71 to 0.91 and the *Hybrid-full* improves it to 0.92 (see Table 4). However, the trade-off regarding budget and *F-measure* achieved by *Skycrowd* is more profitable than for crowdsourcing all the grey area because it uses 64% of its budget to achieve similar *F-measure*. The *Skycrowd* solution achieves a *F-measure* of 0.91 for just 7% of the full crowdsourcing budget. The active learning-aided inference methods used in the *Skycrowd* solution are proven effective since there is a negligible increase in *F-measure* values with reward budgets greater than 64%.

**Effect of varying reward budget on F-measure and Average Distance** The initial increase in the reward budget results in a sharp increase in the recall and F-measure values (see Fig. 21). However, after the 64% budget mark, both the F-measure and recall values flatten. This behavior can be attributed to the fact that *Skycrowd* solution resolves all the grey area SEL task with 64% of the grey area reward budget by inferring the labels through the SC-based

#### 4. Spatial Entity Linkage with the Aid of Spatial Crowdsourcing



**Fig. 21:** Effect of varying reward budget for Skycrowd on the F-measure, precision, and recall (Reproduced from Paper C [30])



(a)

**Fig. 22:** Effect of varying reward budget on the average distance travelled by worker per assignment (Reproduced from Paper C [30])

active learning approach. The little benefit in the F-measure, precision and recall values after the 64% budget mark can be attributed to the effectiveness of the inference methods utilized by the *Skycrowd* solution.

On a similar note, we compared the effect of varying reward budget on the average distance travelled by workers per assignment. The proposed past visits-based task assignment algorithm of the *Skycrowd* is compared against a baseline, *ClosestWorker* task assignment algorithm that assigns the task to the closest worker (See Fig. 22a). The past visits-based task assignment algorithm outperforms the *ClosestWorker* task assignment algorithm by at least 30 %, as the past visits-based task assignment algorithm facilitates completion of task without visiting the task location if the worker has experience with the task location.

## 4.5 Discussion

To resolve the spatial entity linkage problem utilizing the concept of machine learning-aided spatial crowdsourcing, we proposed the hybrid *Skycrowd* solution that finds a synergy to employ both automatic labeling and active learning-aided spatial crowdsourcing techniques. By resolving the spatial entity linkage problem, the *Skycrowd* solution enriches the spatial datasets. As suggested in Section 3, the proposed solution could help identify OSM spatial entities' tag information by extracting attribute information from the corresponding spatial entities in other spatial datasets. However, before employing *Skycrowd* solution to enrich the spatial crowdsourcing dataset OSM, the real-time implementation issues regarding the crowdsourcing of tasks needs to be analyzed. The current solution does not consider workers' dynamic arrival and online assignment of tasks for crowdsourcing. Furthermore, the current task assignment algorithm does not consider workers' attributes like reputation and expertise. Moreover, in cases where multiple tasks are assigned to the same worker, the tasks are not assigned based on the worker's availability. To address these challenges, we propose new transit-based task assignment algorithms considering the worker's availability, movement and credibility/ reputation (discussed in detail in Section 5).

# 5 Transit-based Task Assignment in Spatial Crowdsourcing

## 5.1 Motivation and Problem Statement

The success of an SC application is dependent on the task assignment and scheduling strategies. Majority of the current literature designs the task assignment and scheduling strategies based on the assumption that the worker continues to stay at the reported location. However, as the workers move around in the geographical space, this assumption might not reflect the real-world scenario. Furthermore, workers' task acceptance rate would be improved by considering the workers' movement information as it facilitates the identification of the right time to assign tasks to a worker. Consider the example of worker transit movement information. Intuitively, a worker can perform tasks at the transit stops in her daily public transport route, namely the origin stop, the intermediate stops (if any), and the destination stop.

**Example 1:** Consider the example in Fig.23. There are two workers  $W_1$  and  $W_2$  and three noise data collection tasks  $T_1$ ,  $T_2$ , and  $T_3$  (See Fig. 23a). The transit routes of  $W_1$  and  $W_2$  before and after assignment can be seen in Fig. 23b & 23c, respectively. The worker travels to the assigned task from the transit stop and returns to the transit stop after recording the noise levels at

## 5. Transit-based Task Assignment in Spatial Crowdsourcing

the task location with her smartphone. For instance, worker  $W_1$  travels from  $W_{1,b}$  to  $T_1$  and returns back to  $W_{1,b}$  after performing the task to continue with the transit route. (Reproduced from Paper D [21])

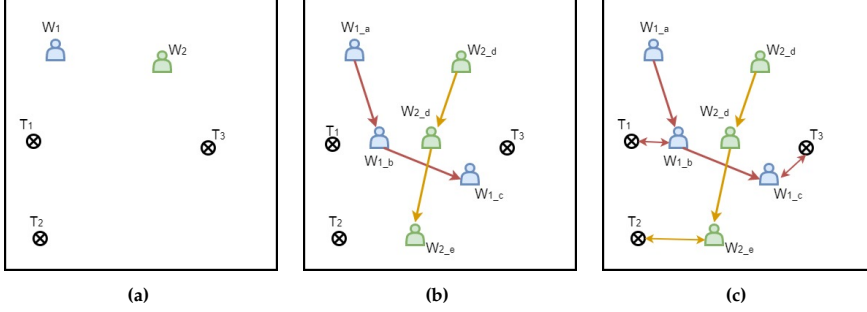


Fig. 23: Transit-based Task Assignment example (Reproduced from Paper D [21])

The assignment and scheduling algorithms should represent a more accurate real-world scenario by considering worker movement information [24]. By considering the worker transit information, we can target a new group of SC workers, public transport users. The aim is to harness the waiting periods in a worker transit route at different transit stops for assigning tasks to workers. However, worker's priority to adhere to the transit route to reach the destination on time and inflexible public transport schedule complicates the task assignment process. Furthermore, the worker seeks to maximize her earnings by choosing tasks that provide maximum rewards, leading to a **Transit-based Task Assignment (TTA) problem**. Consequently, there is a need for a new task assignment algorithm to solve the TTA problem. Additionally, worker credibility scores and worker-defined threshold reward are considered to extend the TTA problem to improve worker response quality and facilitate worker routes' flexibility, respectively.

### 5.2 Preliminaries

The preliminaries are reproduced from Paper D [21].

**Definition 9. (Worker):** A worker, denoted by  $w$ , is a person willing to perform an assigned task by travelling to the task's location. Worker  $w$  has a transit stop set  $WTS$  that contains worker transit stops  $wts$ , belonging to the transit route she follows every day, threshold reward  $thresRew$  represents the expected compensation for not following the fixed transit route  $wr$ ,  $strtTime$  represents the start of the transit trip,  $maxTrvlTime$  represents the total time the worker is willing to spend on her daily route  $wr$ ,  $servRate$  represents the service price charged by the worker  $w$  per hour, and  $credibility$  represents the worker credibility score. A worker is defined as:

$$w = \langle WTS, strtTime, thresRew, maxTrvlTime, servRate, credibility \rangle$$

$$WTS = \{wts_o, \dots, wts_i, \dots, wts_d\}$$

, where  $wts_o$  represents the origin transit stop,  $wts_d$  represents the destination transit stop, and  $wts_i$  represents an intermediate transit stop of the worker transit route. (Reproduced from Paper D [21])

We define the credibility of the worker as the probability of worker performing an assigned task correctly. The credibility of the worker is defined similarly as the reputation score in [37]. The credibility of a worker can be determined based on the historical information of workers' answers stored in the SC-server. We assume that the worker credibility scores are stored and maintained at the SC-server (Reproduced from Paper D [21]).

Furthermore, we defined the worker route as a series of sequential worker transit stops. As mentioned earlier, our intuition is that the worker can perform tasks during their waiting period at the worker transit stops. Worker transit stops are associated with the real-world public transportation stop at a certain geographical location. Accordingly, we define worker transit stop as (Reproduced from Paper D [21]):

**Definition 10.** (Worker Transit Stop): A worker transit stop, denoted by  $wts$ , is at location  $l$ , and has *arrivalTime* and *departureTime*, that represent the arrival and departure timings at the transit stop of the worker  $w$ . *assignedTask* represents the task, if assigned to the transit stop. The worker transit stop is defined as:

$$wts = \langle l, arrivalTime, departureTime, w, assignedTask \rangle$$

We modeled the arrival time at the origin transit stop as the *strtTime* of the worker  $w$ , and the departure time at the destination transit stop as the *strtTime* + *maxTrvlTime* of the worker  $w$  (Reproduced from Paper D [21]).

We define a spatial task as a task associated with a geographical location. The spatial task definition is inspired by [36].

**Definition 11.** (Spatial task): A *spatial task*, denoted by  $t$ , contains a query  $q$  to be answered at location  $l$ . The query is asked at time *issueTime* and will expire at time *expiryTime*. The task takes *taskDuration* time to complete. The task will be guaranteed to result in a correct response, if a worker with at least *minWorkerCred* credibility is assigned to the task (Reproduced from Paper D [21]).

$$t = \langle q, l, issueTime, expiryTime, taskDuration, minWorkerCred \rangle$$

The worker has to visit location  $l$  to perform the task during the time interval between issue time and expiry time. Note that the worker has to visit the task location at least *taskDuration* minutes before the *expiryTime*. For



## 5. Transit-based Task Assignment in Spatial Crowdsourcing

simplicity, we assume that the worker can complete the task with a single response. The task can still be assigned to the worker with less credibility than  $\text{minWorkerCred}$ . However, then the quality of the response is not guaranteed (Reproduced from Paper D [21]).

In [1], it is mentioned that tasks with “extrinsic incentives” like monetary reward would attract more workers, and affects the speed of accomplishing the task. We offer monetary rewards to workers for accomplishing the task. However, instead of a fixed reward per task, we define the reward associated with spatial tasks in proportion to the time spent by the worker to perform the task. We assume that there will be a base reward for performing the task. In addition, we assume that the workers expect a fixed hourly payment rate as compensation for the time spent on performing the task. The time spent is calculated based on the time taken to reach the task location from the worker transit stop, the time taken to do the task, and the time taken to return to the transit stop (Reproduced from Paper D [21]).

**Definition 12.** (Reward): The worker  $w$  will receive reward  $r(w, t)$  after the completion of task  $t$  at transit stop  $w.wts$ . We assume that the reward  $r$  is affinely dependent on the distance between the transit stop’s location  $w.wts.l$  and the task’s location  $t.l$ , and the task duration  $t.taskDuration$ .

$$r(w, t) = w.servRate * (2 * \text{dist}(w.wts.l, t.l) / \text{walkingSpeed} + t.taskDuration) + c$$

, where  $c$  is the fixed reward for all the tasks, for example 10 Kroner,  $\text{serviceRate}$  is the fixed hourly compensation rate charged by the worker  $w$ , for example 60 Kroner per hour, and  $\text{walkingSpeed}$  is the average walking speed of workers ( $m/s$ ). For simplicity, we have assumed all the workers to have the same service rate (Reproduced from Paper D [21]).

**Definition 13.** (Distance from transit stop to task):

$\text{dist}(w.wts.l, t.l)$  denotes the distance that a worker  $w$  at a transit stop  $w.wts$  needs to travel to reach  $t$ . Generally speaking, the distance from a transit stop to task denotes the walking distance from the worker transit stop to the task location (Reproduced from Paper D [21]).

For simplicity, we assume that a worker would perform at most a single task at every transit stop. Intuitively, a worker  $w$  cannot accept all the tasks without considering the additional travel time. Therefore,  $\text{maxTrvlTime}$  is used to limit the amount of time a worker will spend on completing the transit route. The travel time is calculated based on the transit network. The transit info can be reliably extracted from the public transport web services and *Google Maps*. After the worker makes her task inquiry, the SC-server would then try to assign the tasks according to the worker and update her route (Reproduced from Paper D [21]).

Constraints	Types of Assignment Problems		
	Rigid Route-based		Flexible Route-based
	Transit-based	Credible Transit-based	Flexible Transit-based
Task Deadline			
Transit Stop Time			
Minimum Credibility			
Threshold Reward			
Travel Time			

**Table 5:** Types of Task Assignment Problems

### 5.3 Types of Assignment Problems

To maximize the rewards received by transit workers performing tasks along their routes and to ensure a certain level of quality in the task responses, we have identified three task assignment optimization problems (see Table 5):

#### **Transit-based Task Assignment (TTA) problem:**

Transit-based Task Assignment (TTA) problem involves assigning tasks to transit stops along the workers' routes. Owing to the task deadline constraint and rigid public transport schedules, the SC-server has to optimize the task assignment and scheduling strategy to achieve the objective to maximize the net reward for the individual workers performing the assigned tasks. TTA problem is an optimization problem to maximize the workers' net reward performing tasks on their routes. To better understand the TTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). For every incoming batch of workers and tasks, we can solve the batch-based TTA problem by treating the individual batch as an offline TTA problem. Subsequently, we can solve the offline-TTA problem by reducing it to the maximum weighted bipartite matching (MWBM) problem. The definition of the offline and batch-based TTA problem is given below (The problem definitions are reproduced from Paper D [21]):

**Definition 14.** (Offline Transit-based Task Assignment Problem): Assume an input set of tasks  $T$  and a set of workers  $W$  along with their set of worker transit stops  $WTS$ . The offline Transit-based Task Assignment (Offline-TTA) problem is an optimization problem with the objective to maximize the sum

## 5. Transit-based Task Assignment in Spatial Crowdsourcing

of the new rewards received by the individual workers. The Offline-TTA problem finds an optimal Transit Stop Task Assignment set, denoted by  $TA(WST, T)$ , with average net reward of  $r_{TA}$ , is a set of 3-tuples of the form  $\langle wts, t, r \rangle$ , where  $wts$  is assigned to  $t$  with an associated reward  $r$ , given the following constraints are satisfied (Reproduced from Paper D [21]):

- **Transit Stop Time Constraint:** The time required to complete the task, i.e., travel time from the transit stop to the task location and returning to the transit stop and the  $taskDuration$  is less than the time spent at the transit stop, i.e.,

$$(2 * distance(wts.l, t.l) / walkingSpeed) + t.taskDuration < (wts.departureTime - wts.arrivalTime)$$

- **Task Deadline Constraint:** The worker's arrival time at the transit stop should be at least  $t.taskDuration$  before the task deadline, i.e.,

$$t.expiryTime \geq wts.arrivalTime + t.taskDuration$$

**Definition 15.** (Online Batch-based Transit-based Task Assignment problem): Assume a set of batches  $B$ , with each incoming batch  $b \in B$  comprising a set of unassigned tasks  $T_b$  and available workers  $W_b$  along with their transit routes  $WTS_b$ . The Online Batch-based Transit-based Task Assignment (Batch-TTA) problem is an optimization problem with the goal of finding an offline Transit Stop Task Assignment set  $TA(WTS_b, T_b)$  that maximizes the net rewards( $r_{TA}$ ) received by the individual workers  $w \in W_b$  at their worker transit stops  $wts \in WTS_b$  by performing assigned tasks  $t \in T_b$  for each incoming batch  $b$  (Reproduced from Paper D [21]).

### Credible Transit-based Task Assignment(CTA) Problem

To ensure the desired level of quality for the worker responses, we propose the *Credible Transit-based Task Assignment (CTA)* problem. The CTA problem extends the TTA problem by including the minimum worker credibility threshold constraint for the spatial task as the probability of the task being performed correctly. In other words, a spatial task can be assigned to the worker if and only if the worker's credibility is greater than or equal to the minimum worker credibility threshold of the spatial task(The problem definitions are reproduced from Paper D [21]).

**Definition 16.** (Minimum Worker Credibility Threshold Constraint): The worker's credibility score should be atleast the task's MWCT.

$$w.credibility \geq t.minWorkerCred \quad (5)$$

### Flexible Transit-based Task Assignment (Flexible-TTA) problem

The aforementioned TTA and CTA problems strictly follow the worker routes without any deviations. However, the fixed worker route constraint can be relaxed if a lucrative incentive is offered. We assume that the worker will stay longer at a transit stop if a neighbouring task has a higher reward than a specific threshold value, despite the travel stop time constraint. Considering this, we propose the Flexible TTA problem, where **the worker transit route is no longer considered fixed and can be changed**. For a better understanding of the Flexible-TTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). Considering these constraints, we define the offline and batch-based Flexible-TTA problems as (The problem definitions are reproduced from Paper D [21]):

**Definition 17.** (Offline Flexible Transit-based Task Assignment problem): Assume an input set of spatial tasks  $T$  and a set of workers  $W$  along with their set of worker transit stops  $WTS$ . The offline Transit-based Task Assignment (Offline Flexible-TTA) problem is an optimization problem with an objective to maximize the net rewards of individual workers. Offline Flexible-TTA finds an optimal Flexible Transit Stop Task Assignment set, denoted by  $FTA(WTS, T)$  with net reward  $r_{FTA}$ , is a set of 3-tuples of the form  $\langle wts, t, r \rangle$ , where  $wts$  is assigned to  $t$  with an associated reward  $r$ , given the following constraints are satisfied (Reproduced from Paper D [21]):

- Threshold Reward Constraint : The reward  $r$  should be greater than the worker  $w$ 's  $thresRew$ , i.e.,  $r > w.thresRew$
- Task Deadline Constraint: The task deadline should be later than the worker's arrival time at the transit stop, i.e.,  $t.expiryTime > wts.arrivalTime$
- Travel Time Constraint: The new reconstructed transit route with the updated  $wts.departureTime$  at transit stop  $wts$  should allow the worker  $w$  to reach the destination before  $w.startTime + w.maxTrolTime = destThresholdTime$ , i.e.,

$$destThresholdTime > wts.arrivalTime + t.taskDuration + (2 * dist(wts.l, t.l) / walkingSpeed)$$

**Definition 18.** (Batch-based Flexible Transit-based Task Assignment Problem): Assume a set of batches  $B$ , with each incoming batch  $b \in B$  comprising a set of unassigned tasks  $T_b$  and available workers  $W_b$  along with their transit routes  $WTS_b$ . The Batch-based Flexible Transit-based Task Assignment (Batch-based Flexible-TTA) problem is an optimization problem with the goal

## 5. Transit-based Task Assignment in Spatial Crowdsourcing

of finding an optimal Flexible Transit Stop Task Assignment  $FTA(WTS_b, T_b)$  that maximizes the net rewards ( $r_{FTA}$ ) received by the individual workers  $w \in W_b$  at their worker transit stops  $wt_s \in WTS_b$  by performing assigned task  $t \in T_b$  for every batch  $b$  (Reproduced from Paper D [21]).

Task	Issue Time	Expiry Time	Transit Stop	Reward	Worker Credibility Threshold
$t_1$	8:01 AM	5:00 PM	$wt_{s1}$	20	0.7
$t_2$	8:15 AM	5:00 PM	$wt_{s1}$	25	0.7
$t_3$	8:30 AM	5:00 PM	-	10	0.6
$t_4$	9:00 AM	5:00 PM	$wt_{s3}$	20	0.6

**Table 6:** Example 2: Tasks and associated transit stops (Reproduced from Paper D [21])

Stop	Arrival	Departure	Credibility	Threshold Reward
$wt_{s1}$	8:00 AM	8:20 AM	0.6	30
$wt_{s2}$	8:40 AM	9:00 AM	0.6	30
$wt_{s3}$	9:20 AM	9:40 AM	0.6	30

**Table 7:** Example 2: Transit Stops (Reproduced from Paper D [21])

### 5.4 Algorithms

We propose four algorithms to solve the proposed task assignment optimization problems, two for the TTA problem and one each for the remaining optimization problems. For a better understanding of the algorithms, we use a running example described in Tables 6 and 7. Consider the scenario in Fig. 24a, where the worker  $w$  follows a transit route with three transit stops ( $wt_{s1}, wt_{s2}, wt_{s3}$ ) (The schedule, credibility scores and threshold reward values are mentioned in Table 7). There are four tasks sent to the SC-server (details are in Table 6). It can be noticed that at stop  $wt_{s1}$ , two tasks ( $t_1, t_2$ ) satisfy the travel stop time constraint. However, task  $t_1$  is issued before task  $t_2$ . Similarly, it can be noticed that none of the tasks satisfies the travel stop time constraint at the transit stop  $wt_{s2}$  (Reproduced from Paper D [21]).

#### Online Baseline Algorithm

As a baseline, we consider the online input model, where the worker/ task arrives dynamically to the system. The SC-server will not have any prior information about the WTRs in the online input model. We assume that the worker would notify the SC-server whenever she is available to perform a task. In our transit-based context, the worker will notify the SC-server

---

**ALGORITHM 8: ONLINE TRANSIT-BASED TASK ASSIGNMENT (OLA) (REPRODUCED FROM PAPER D [21])**

---

**Input:** An incoming task  $t$  or a recently available worker  $w$  at transit stop  $wts$  along with available workers  $W$  at worker transit stops  $WTS$ , and a set of previously unassigned tasks  $T$ .  $c$  is the fixed reward per task.

**Output:** Task Assignment  $\langle w, wts, t \rangle$  with reward  $maxReward$  and minimum travel distance  $minDist$

```

1   $maxReward \leftarrow 0$ ;
2   $minDist \leftarrow \infty$ ;
3   $removeExpiredTasks(T)$ ;
4   $removeUnavailbleWorkers(W)$ ;
5  if New Task Arrival  $t$  then
6       $T \leftarrow T \cup \{t\}$ ;
7      foreach  $wts \in WTS$  do
8           $maxDistAtStop \leftarrow walkingSpeed * (wts.departureTime -$ 
9               $wts.ArrivalTime - t.taskDuration) / 2$ ;
10         if  $dist(wts.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$  then
11              $minDist \leftarrow dist(wts.l, t.l)$ ;
12              $maxReward \leftarrow r(w, t)$ ;
13              $Assignment \leftarrow \langle w, wts, t \rangle$ ;
14              $T \leftarrow T \setminus \{t\}$ ;
15              $W \leftarrow W \setminus \{w\}$ ;
16              $WTS \leftarrow WTS \setminus \{wts\}$ ;
17 else if New Worker Arrival  $w$  at  $wts$  then
18      $W \leftarrow W \cup \{w\}$ ;
19      $WTS \leftarrow WTS \cup \{wts\}$ ;
20      $maxDistAtStop \leftarrow$ 
21          $walkingSpeed * (wts.departureTime - wts.ArrivalTime - t.taskDuration) / 2$ ;
22     foreach  $t \in T$  do
23         if  $dist(wts.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$  then
24              $minDist \leftarrow dist(wts.l, t.l)$ ;
25              $maxReward \leftarrow r(w, t)$ ;
26              $Assignment \leftarrow \langle w, wts, t \rangle$ ;
27              $T \leftarrow T \setminus \{t\}$ ;
28              $W \leftarrow W \setminus \{w\}$ ;
29              $WTS \leftarrow WTS \setminus \{wts\}$ ;
30 return  $Assignment \langle w, wts, t \rangle$ 

```

---

## 5. Transit-based Task Assignment in Spatial Crowdsourcing

whenever she reaches a transit stop. The SC-server tries to assign a task once a worker becomes available immediately. The baseline online transit-based task algorithm is denoted by *OLA* (See Algorithm 8) (Reproduced from Paper D [21]).

---

**ALGORITHM 9:** MAX. WEIGHTED BIPARTITE MATCHING (REPRODUCED FROM PAPER D [21])

---

**Input:** An incoming batch  $b$  consisting of a non-empty set of workers  $W_b$  with associated set of worker transit stops  $WTS_b$  with  $thresRew$  as the minimum Threshold reward for flexible transits and  $maxTrvlTime$  as maximum travel time, a set of tasks  $T_b$ .  $TA(WTS, T)$  is the optimal set of assignments before the batch  $b$ .  $c$  is the fixed reward per task.

**Output:** The Optimal set of assignments  $TA(WTS \cup WTS_b, T \cup T_b)$  with the average Reward  $r_{TA}$  and minimum travel distance  $minDist$  for batch  $b$

```

1  $TA(WTS \cup WTS_b, T \cup T_b) \leftarrow NULL;$ 
2  $TA(WTS_b, T_b) \leftarrow NULL;$ 
3  $WeightedGraph\ G \leftarrow NULL;$ 
4 foreach  $t \in T_b$  do
5    $G.addVertex(t);$ 
6 foreach  $wts \in WTS_b$  do
7    $G.addVertex(wts);$ 
8   foreach  $t \in T_b$  do
9      $maxDistAtStop \leftarrow walkingSpeed * (wts.departureTime -$ 
10       $wts.ArrivalTime - t.taskDuration) / 2;$ 
11     if  $dist(wst.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$  then
12        $edgeWeight \leftarrow r(w, t);$ 
13        $G.addEdge(wts, t, edgeWeight);$ 
14  $TA(WTS_b, T_b) \leftarrow MWBM.getMatching(G, WTS_b, T_b);$ 
15  $TA(WTS \cup WTS_b, T \cup T_b) \leftarrow TA(WTS, T) \cup TA(WTS_b, T_b);$ 
16  $removeAssignedAndExpiredTasks(T \cup T_b);$ 
17  $removeAssignedWorkerStps(WTS \cup WTS_b);$ 
18 return  $TA(WTS \cup WTS_b, T \cup T_b)$ 

```

---

Stop	Task Assigned	Reward
$wts_1$	$t_1$	20
$wts_2$	None	0
$wts_3$	$t_4$	20

**Table 8:** Eg. 2 DA Solution (Reproduced from Paper D [21])

### Maximum Weighted Bipartite Matching (MWBM) Algorithm

MWBM algorithm tries to solve the *Batch-based TTA* problem by dividing it into individual *offline-TTA* problems for each incoming batch (See Algo-

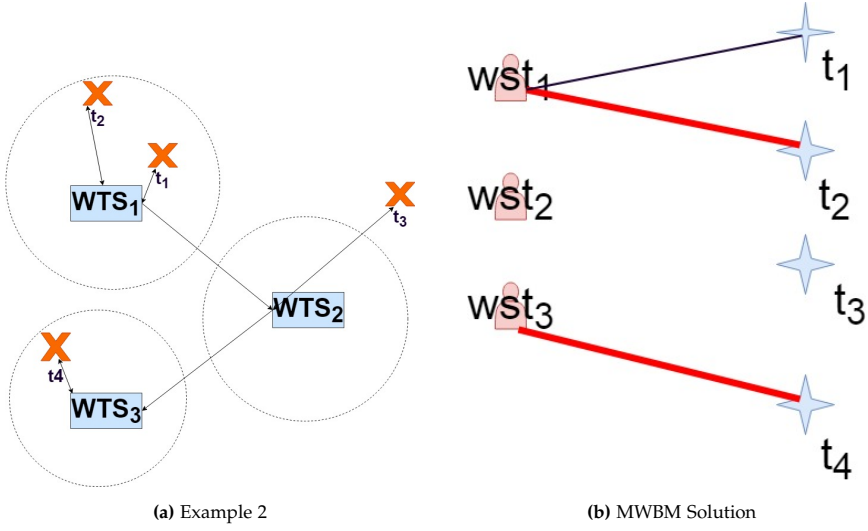


Fig. 24: (Reproduced from Paper D [21])

Stop	Task Assigned	Reward
$wts_1$	None	0
$wts_2$	None	0
$wts_3$	$t_4$	20

Table 9: Eg. 2 CTA Solution (Reproduced from Paper D [21])

Stop	Task Assigned	Reward	New Arrival
$wts_1$	$t_1$	20	NA
$wts_2$	$t_3$	35	NA
$wts_3$	$t_4$	20	9:45 AM

Table 10: Eg. 2 Flexible-DA Solution (Reproduced from Paper D [21])



rithm 9). Each individual *offline-TTA* problem can be reduced to a maximum weighted bipartite matching problem. The *offline-TTA* problem is solved by employing the maximum weight bipartite matching algorithm [46]. The running example is solved by MWBM algorithm in Fig. 24.

---

**ALGORITHM 10:** MIN. DISTANCE-BASED ASSIGNMENT (REPRODUCED FROM PAPER D [21])

---

**Input:** Same as Algorithm 9

**Output:** Same as Algorithm 9

```

1   $TA(WTS \cup WTS_b, T \cup T_b) \leftarrow NULL;$ 
2   $TA(WTS_b, T_b) \leftarrow NULL;$ 
3   $maxReward \leftarrow 0;$ 
4   $minDist \leftarrow \infty;$ 
5  foreach  $wts \in WTS_b$  do
6       $maxDistAtStop \leftarrow$ 
         $walkingSpeed * (wts.departureTime - wts.ArrivalTime - t.taskDuration) / 2;$ 
7       $feasibleTask \leftarrow NULL;$ 
8      foreach  $t \in T_b$  do
9          if  $dist(wts.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$  then
10             if  $minDist > dist(wts.l, t.l) \wedge maxReward < r$  then
11                  $minDist \leftarrow dist(wts.l, t.l);$ 
12                  $maxReward \leftarrow r(w, t);$ 
13                  $feasibleTask \leftarrow t;$ 
14   $TA(WTS_b, T_b) \leftarrow TA(WTS_b, T_b) \cup TA(wts, feasibleTask);$ 
15 Lines 14 – 16 from MWBM Algorithm 9 return  $TA(WTS \cup WTS_b, T \cup T_b)$ 

```

---

### Minimum Distance based Direct Assignment (DA) Algorithm

We propose a direct assignment-based algorithm (DA) to solve the TTA problem by prioritizing tasks closer to the transit stop along the worker transit route. DA algorithm solves the batch-based TTA by transforming it into individual offline-TTA problems for each incoming batch. (See Algorithm 10). DA algorithm tries to maximize workers' rewards while minimizing the travel distance for the workers to the assigned tasks. The running example is solved by DA algorithm in Table 8.

### Credible Transit-based Task Assignment (CTA) Algorithm

The CTA algorithm tries to solve the CTA problem by finding the best feasible task at each worker transit stop that satisfies all the constraints, including the MWCT constraint. CTA algorithm tries to maximize the workers' rewards and minimize their travel distances to the assigned tasks (See Algorithm 11). The running example is solved by CTA algorithm in Table 9

---

**ALGORITHM 11: CREDIBLE TRANSIT-BASED TASK ASSIGNMENT (CTA) (REPRODUCED FROM PAPER D [21])**

---

**Input:** Same as Algorithm 9

**Output:** Same as Algorithm 9

```
8 Algorithm Same As DA except Line 9 Replaced By Below Psuedocode
9 if  $\text{dist}(wts.l, t.l) \leq \text{maxDistAtStop} \wedge t.\text{expiryTime} >$   
    $wts.\text{arrivalTime} \wedge t.\text{minWorkerCred} \leq w.\text{credibility}$  then
10 return  $TA(WTS \cup WTS_b, T \cup T_b)$ 
```

---

---

**ALGORITHM 12: FLEXIBLE DIRECT ASSIGNMENT (REPRODUCED FROM PAPER D [21])**

---

**Input:** Same as Algorithm 9, except  $FTA(WTS, T)$  is the optimal set of assignments instead of  $TA(WST, T)$

**Output:** Same as Algorithm 9, except with  $FTA(WTS \cup WTS_b, T \cup T_b)$

```
8 Same as DA Algorithm 10, except lines 9 – 13 replaced by below psuedocode
9 if  $\text{dist}(wts, t) \leq \text{maxDistAtStop} \wedge t.\text{expiryTime} > wts.\text{arrivalTime}$  then
10   | Lines 10 – 13 from DA Algorithm 10
11 else if  $w.\text{threshold} \geq r$  then
12   |  $wts.\text{departureTime} \leftarrow wts.\text{arrivalTime} + (2 * \text{dist}(wts, t) / \text{WalkingSpeed});$ 
13   |  $\text{ReconstructRoutes}(wts, w.\text{destination},$   
   |    $wts.\text{departureTime});$ 
14   | if  $\text{ReconstructedRouteDestinationTime} < \text{startTime} + w.\text{maxTravelTime}$  then
15   |   | Lines 10 – 13 from DA Algorithm 10
16 return  $FTA(WTS \cup WTS_b, T \cup T_b)$ 
```

---

### Flexible Transit Route-based Direct Assignment (Flexible-DA) Algorithm

Flexible-DA algorithm tries to solve the Batch-based Flexible-TTA problem by breaking into individual *Offline Flexible-TTA* problems for every incoming batch (See Algorithm 12). This algorithm tries to maximize the rewards received by workers and simultaneously seeks to reduce the travel distance to the tasks. Additionally, it facilitates search space expansion for the worker in an event where there is no task adhering to the worker’s transit route. The search space expands by looking for tasks beyond the transit stop time constraint that offer reward higher than the threshold reward value. The running example is solved by Flexible-DA algorithm in Table 10.

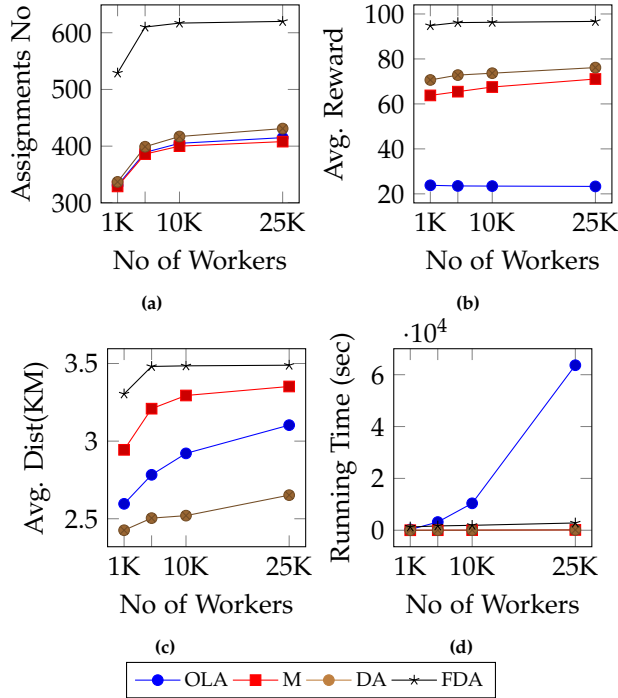


Fig. 25: Effect of varying the number of workers on :a. Number of Assignments.b Average Reward received by a worker c. Average Distance travelled by a worker (KM) d. Average Running Time (seconds) (Reproduced from Paper D [21]).

## 5.5 Experimental Evaluation

To evaluate the proposed assignment algorithms, we have considered the study area of Aalborg, Denmark. First, we generated synthetic workers and their routes between the residential and commercial zones in Aalborg, Denmark, using the Rejseplanen Public Transport Service, Denmark. Second, we

generated synthetic tasks around the city of Aalborg, Denmark. The workers are generated with uniform values of *thresRew* and *maxTrolTime* parameters. The workers' credibility parameters are generated with random values between 0.2 to 0.9 due to workers' historical information's unavailability. The tasks are generated with varying values of *issueTime* between "7:00 to 21:00". The default *expiryTime* is set as twenty four hours from the *issueTime*. The tasks are generated with equal values of minimum worker credibility threshold parameters (Reproduced from Paper D [21]).

Figure 25a demonstrates the impact of varying the number of workers on the number of assignments. *Flexible-DA* assigns the highest number of tasks among all the evaluated algorithms. *Flexible-DA* results in 55% more assignments than the *OLA*, *MWBM*, and *DA* algorithms. *Flexible-DA* bypasses the transit stop time constraint in cases where there are no feasible tasks at the transit stops, which facilitates more assignments. *DA* has performed marginally better than the *OLA* and *MWBM* algorithms.

With regard to the average reward received by the worker, *Flexible-DA* performs better than the other algorithms (See Fig. 25b). The average reward received by the worker in the *Flexible-DA* algorithm has increased by 35% when compared to *DA*, and around 50% when compared to *MWBM*. *Flexible-DA* results in three times higher reward than the baseline algorithm *OLA*. The reason is that in the case of *Flexible-DA*, the SC-server has the knowledge about the worker's future spatiotemporal movement that facilitates the worker to extend the stay at different transit stops of her route. Furthermore, *MWBM* and *DA* result in nearly three times higher reward than the baseline algorithm *OLA*. *Flexible-DA* performs better due to the availability of tasks with rewards higher than the threshold, that satisfies the maximum travel time constraint. *DA* delivers 15% better average reward than *MWBM* (Reproduced from Paper D [21]).

It can be observed from Fig. 26a and Fig. 26b, that as the *MWCT* value increases, the number of assignments, the average travel distance, and the average reward showcases a downward trend. Given that the number of worker and the number of tasks are fixed at 5K, the number of potential assignments with the satisfied credibility constraint will be reduced as the *MWCT* value increases. For example, when the *MWCT* value is increased to 0.6 from 0.5, the workers with credibility values ranging between [0.5,0.6) becomes ineligible to perform tasks owing to the credibility constraint. The running time is not impacted by the increase of *MWCT* value (Reproduced from Paper D [21]).

Fig. 27a and Fig. 27b showcase the positive impact of increasing *maxTravelTime* value on the number of assignments, average travel distance, average reward and the running time. An increase in the *maxTravelTime* value would make more tasks eligible for the assignment in the *Flexible-DA* algorithm.

Similarly, Fig. 27c and Fig. 27d showcases the negative impact of the in-

## 5. Transit-based Task Assignment in Spatial Crowdsourcing

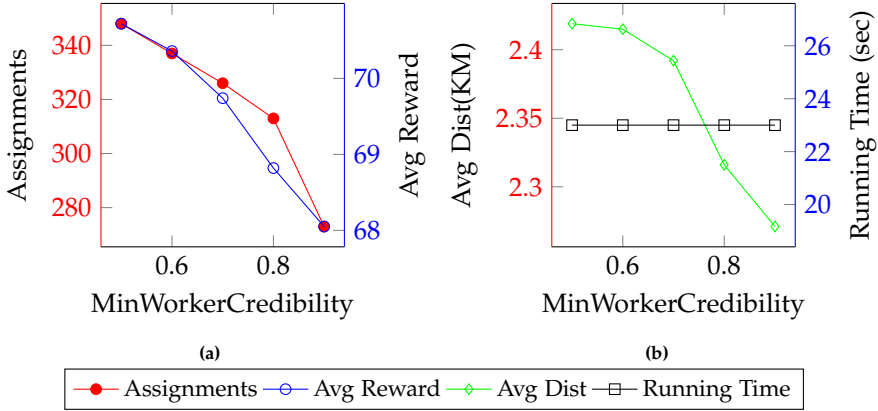


Fig. 26: Effect of varying minimum worker credibility threshold (a,b) on *Credible Transit-based Task Assignment Algorithm* (Reproduced from Paper D [21]).

creasing threshold reward value of the worker on the number of assignments, the average travel distance and the average running speed. Such behaviour is expected since a higher threshold reward reduces the eligibility of many tasks for the flexible transit route-based task assignment. However, the average reward value increases as the threshold reward increases as the priority is given to tasks with higher rewards.

### 5.6 Discussion

We proposed a task assignment model that exploits the workers' transit information to offer an alternative strategy for the online spatial task assignment in SC. This paper modeled the potential task assignment opportunities in a fixed worker transit route followed strictly. Additionally, we modeled a flexible transit route scenario assuming that the worker would be willing to delay her trip if a task offers more than a threshold reward. We defined the three variants of the TTA problem, *offline-TTA*, *Batch-TTA*, and *Flexible-TTA*, to maximize worker rewards considering the fixed and the flexible worker transit models, respectively (Reproduced from Paper D [21]).

The proposed transit-based task assignment algorithms targeting the new group of SC workers (public transport users) can contribute towards the enrichment of spatial datasets. The algorithms' effectiveness has to be evaluated and finetuned by considering more real-world cases in the context of spatial dataset enrichment.

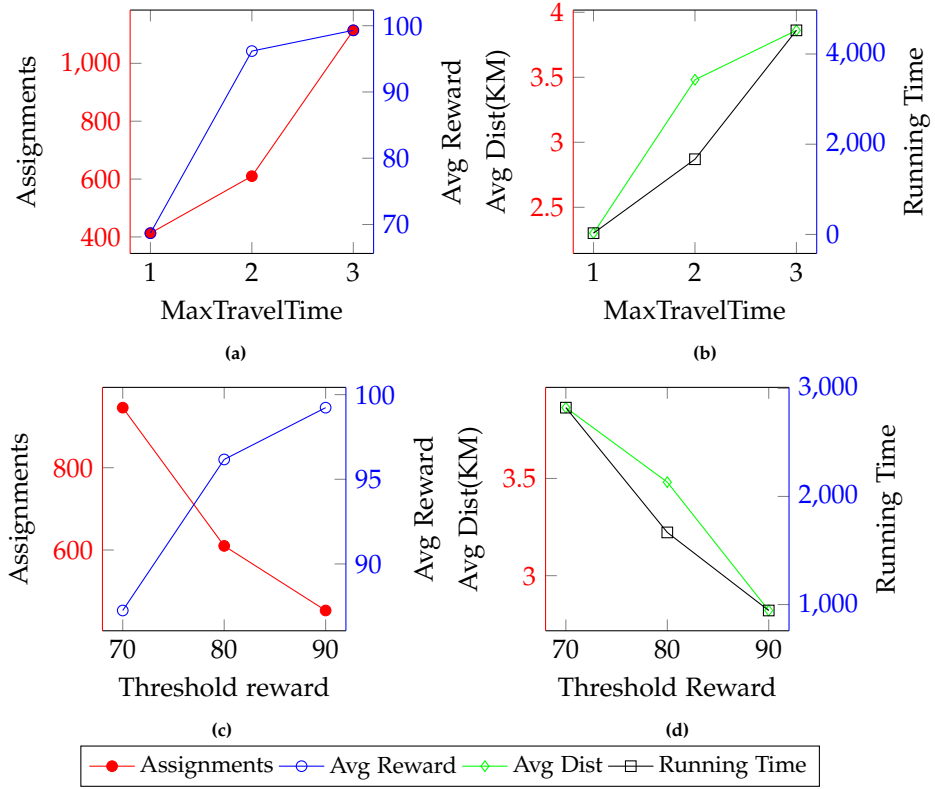


Fig. 27: Effect of varying maxTravelTime (a,b) and threshold reward (c, d) on *Flexible-DA* (Reproduced from Paper D [21]).

## 6 Summary of Contributions

In conclusion, this thesis performs a comprehensive survey of the existing SC literature and proposes novel multi-modal crowdsourcing applications for enriching spatial datasets. In addition to addressing the spatial dataset enrichment issue, the proposed applications address other issues highlighted by the survey (Paper A [24]) by suggesting strategies for improving worker participation and improving task assignment/ scheduling algorithms. The thesis comprises five papers, and their contributions are mentioned below. Paper E [23] was later extended by Paper B [22]; therefore, Paper E's contributions are discussed before Paper B's.

Paper A [24] provides a comprehensive overview of the current SC literature and enumerates the fundamental differences between conventional crowdsourcing and spatial crowdsourcing. The survey introduces a taxonomy to classify the existing literature based on the type of the task serviced, publishing mode, addressed problem types, constraints considered,

## 6. Summary of Contributions

employed privacy protection measures, utilized data aggregation measures, and the application’s nature. The survey provides a detailed comparison of existing task assignment and scheduling problems and highlights the drawbacks. Furthermore, Paper A discusses the challenges faced by the current SC literature regarding the enrichment of spatial datasets, improvement of worker participation, improvement of task assignment and scheduling strategies, and privacy issues. In addition, the survey suggests some potential future research directions.

Paper E [23] explores the potential of a push-based multi-modal SC application to assist the enrichment of OpenStreetMap (OSM) spatial entities’ semantic tag information. Focusing on the use case of OSM road networks, Paper E defines the maximum road segment task assignment optimization problem to maximize the number of road segment tasks to the workers. The maximum road segment task assignment optimization problem is analyzed using both the offline and batch worker input models. Two algorithms are proposed to solve the batch-based maximum road segment task assignment problem. Furthermore, based on the concept of acquiring road segment information at the junctions, Paper E proposes an additional junction-based algorithm to maximize the number of assigned road segment tasks. The experimental evaluation showcases the effectiveness of the proposed algorithms in terms of the number of assignments compared to the baseline.

Paper B [22] extends Paper E [23] to provide an end-to-end integrated framework for enabling a push-based multi-modal SC application for OSM semantic tags enrichment. Furthermore, Paper B defines two additional optimization problems to improve the coverage of entities and ensure the crowdsourced OSM tags’ verifiability. The integrated framework facilitates the extraction of OSM spatial entities from the OSM database, assigning spatial entities as tasks to workers, aggregates and validates the crowdsourced tag information, and updates the OSM database with the validated tag information. Paper B provides task assignment algorithms to solve additional optimization problems to maximize the number of unique task assignments and maximize the number of verifiable task assignments. The experimental evaluation shows that the proposed algorithms outperform the baseline. Paper B addresses SC’s spatial dataset enrichment challenge by improving the quality and quantity of tags in the OSM dataset. Furthermore, Paper B tries to improve worker participation by actively pushing tasks to OSM contributors.

Paper C [30] explores the possibility of supplementing automatic labeling (AL) methods with multi-modal SC applications to improve their labeling efficiency while solving the spatial entity linkage problem. Paper C defines the *spatial entity linkage with the aid of spatial crowdsourcing* problem to find the optimal balance between AL and SC usage for labeling, given a certain reward budget. Paper C provides a hybrid *Skycrowd* solution to solve the spatial entity linkage with the aid of the spatial crowdsourcing problem. The *Skycrowd*

solution employs skyline-based automatic labeling and active learning-aided SC techniques to maximize the *F-measure* of the resultant labeled spatial entity pairs. The experimental evaluation highlights the success of the proposed *Skycrowd* solution in delivering the best trade-off for the *F-measure* and the reward budget. Paper C specifically addresses the challenge of spatial dataset enrichment in SC.

Paper D [21] explores the potential of incorporating public transportation users as SC workers for performing tasks during their daily commute. Paper D analyzes the different worker transit route models and defines the transit task assignment problem to assign tasks to workers during the waiting periods at different transit stops. Moreover, two variations of the transit task assignment problem are defined to accommodate the credibility constraints and flexible worker transit route models. Paper D provides three custom task assignment algorithms for solving the three variant of the transit task assignment problems aligning with the schedule of public transportation. Experimental evaluation demonstrates the proposed algorithms' effectiveness compared to the baseline online assignment algorithm. Paper D focuses on improving the real-time task assignment and scheduling challenges faced by SC. Additionally, Paper D addresses the challenge of improving SC worker participation by designing strategies to recruit a new group of workers using public transportation.

## 7 Future Work

This thesis proposes an integrated framework comprising different multi-modal SC applications regarding OSM tag enrichment, spatial entity linkage, and transit-based task assignment. However, this thesis does not explore the potential implementation issues or complications when combining all these individual multi-modal SC applications into one. As part of future work, this thesis suggests implementing a comprehensive multi-modal SC solution to enrich the OSM dataset, integrating all the proposed individual multi-modal SC applications.

Similarly, the workers' privacy concerns could be a significant implementation issue for integrating the proposed individual multi-modal SC applications since this thesis does not address any privacy concerns of the workers. Therefore, the proposed integrated framework should be evaluated in a privacy-enabled setting and should incorporate privacy protection measures for dynamic task assignment to ensure workers' privacy. Additionally, new truth inference models and quality assurance models need to be developed for the privacy-protected multi-modal SC applications.

Furthermore, this thesis assumes that workers accept the assigned tasks and performs the tasks. However, workers' task acceptance behavior and the



## 7. Future Work

practical issues faced by workers while performing a task need to be studied based on crowdsourced responses from real SC workers. Similarly, while assigning a task to multiple workers, this thesis assumes that the workers perform a task independent of each other without any collusion. However, in practice, workers might copy off each other, resulting in a reduction in the quality of post-crowdsourcing truth inference models. Consequently, copy-detection methods should be employed while processing worker responses.

Worker participation determines the success of a proposed multi-modal SC application. This thesis proposes strategies to improve OSM contributor participation by actively pushing tasks and customizing task assignment and scheduling to encourage the participation of a new group of SC workers (public transportation users). However, the proposed strategies need to be evaluated with real crowdsourcing workers to verify their effectiveness. Similar to public transportation users, other groups of workers need to be targeted who might not be familiar with multi-modal SC.

In addition to the above mentioned future research directions, we present future work specific to each of the individual multi-modal SC applications below:

The push-based multi-modal SC application for OSM semantic tag enrichment (Paper B [22]) requires new strategies to maximize the number of tags captured by the worker during their visit to the task location. Similarly, as multiple workers can be assigned the same task, there is a need to develop methods that diversify the tags collected by workers during their visits. Additionally, there is a need to develop strategies to update the OSM database with the crowdsourced tag details following the local OSM community's rules and OSM's bulk update policy.

The task assignment algorithm presented in *Skycrowd* solution (Paper C [30]) needs to be improved to consider the dynamic arrival of workers and their constraints related to availability, the preferred region of work, maximum tasks per day, minimum expected reward. Similarly, there is a need to develop task scheduling algorithms to avoid scheduling conflicts while assigning multiple tasks to a single worker. Additionally, the *Skycrowd* solution should be evaluated with other clustering techniques like k-means to compare their impact on the F-measure.

Finally, there are several promising future research directions regarding the transit-based task assignment (Paper D [21]). First, there is a need to evaluate the proposed assignment algorithms' time-effectiveness against the auction model, where the workers can bid on the task. Second, there is a need to improve the algorithm to support sequential dependency among tasks, i.e., the transit-based task assignment should ensure that the tasks are performed in a specific order.

## References

- [1] M. Allahbakhsh, B. Benatallah, A. Ignjatovic, H. R. Motahari-Nezhad, E. Bertino, and S. Dustdar, "Quality control in crowdsourcing systems: Issues and directions," *IEEE Internet Computing*, vol. 17, pp. 76–81, 2013.
- [2] F. Alt, A. S. Shirazi, A. Schmidt, U. Kramer, and Z. Nawaz, "Location-based crowdsourcing: extending crowdsourcing to the real world," in *NordiCHI*, 2010, pp. 13–22.
- [3] M. Asghari and C. Shahabi, "On on-line task assignment in spatial crowdsourcing," in *Big Data*, 2017, pp. 395–404.
- [4] J. Bar-Ilan, G. Kortsarz, and D. Peleg, "Generalized submodular cover problems and applications," *Theoretical Computer Science*, vol. 250, no. 1, pp. 179–200, 2001.
- [5] K. Benouaret, R. Valliyur-Ramalingam, and F. Charoy, "Answering complex location-based queries with crowdsourcing," in *Collaboratecom*, 2013, pp. 438–447.
- [6] B. Berjawi, E. Chesneau, F. Duchateau, F. Favetta, C. Cuntty, M. Miquel, and R. Laurini, "Representing uncertainty in visual integration." in *DMS*, 2014.
- [7] Y. Censor, "Pareto optimality in multiobjective problems," *Applied Mathematics and Optimization*, 1977.
- [8] C. Chen, S.-F. Cheng, A. Gunawan, A. Misra, K. Dasgupta, and D. Chander, "Traccs: A framework for trajectory-aware coordinated urban crowd-sourcing," in *HCOMP*, 2014, pp. 30–40.
- [9] Z. Chen, C. C. Cao, L. Chen, and C. J. Zhang, "gMission : A General Spatial Crowdsourcing Platform," in *VLDB Endowment*, 2014, pp. 1629–1632.
- [10] P. Cheng, X. Lian, L. Chen, J. Han, and J. Zhao, "Task assignment on multi-skill oriented spatial crowdsourcing," *TKDE*, vol. 28, no. 8, pp. 2201–2215, 2016.
- [11] P. Cheng, X. Lian, Z. Chen, R. Fu, L. Chen, J. Han, and J. Zhao, "Reliable diversity-based spatial crowdsourcing by moving workers," *VLDB Endowment*, vol. 8, no. 10, pp. 1022–1033, 2015.
- [12] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos, "Anonymsense: privacy-aware people-centric sensing," in *MobiSys*, 2008, pp. 211–224.
- [13] H. Dang, T. Nguyen, and H. To, "Maximum Complex Task Assignment: Towards Tasks Correlation in Spatial Crowdsourcing," in *IIWAS*, 2013, p. 77.
- [14] K.-H. Dang and K.-T. Cao, "Towards reward-based spatial crowdsourcing," in *ICCAIS*, 2013, pp. 363–368.
- [15] D. Deng, C. Shahabi, and U. Demiryurek, "Maximizing the number of worker's self-selected tasks in spatial crowdsourcing," in *GIS*, 2013, pp. 324–333.
- [16] D. Deng, C. Shahabi, and L. Zhu, "Task matching and scheduling for multiple workers in spatial crowdsourcing," in *GIS*, 2015, p. 21.
- [17] M. Ester, H. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *KDD*, 1996.

## References

- [18] W. Feng, Z. Yan, H. Zhang, K. Zeng, Y. Xiao, and Y. T. Hou, "A survey on security, privacy and trust in mobile crowdsourcing," *IEEE Internet of Things Journal*, pp. 2971–2992, 2017.
- [19] Z. Feng, Y. Zhu, Q. Zhang, L. M. Ni, and A. V. Vasilakos, "Trac: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing," in *INFOCOM*, 2014, pp. 1231–1239.
- [20] D. A. Garcia-Ulloa, L. Xiong, and V. Sunderam, "Truth discovery for spatio-temporal events from crowdsourced data," *VLDB Endowment*, vol. 10, no. 11, pp. 1562–1573, 2017.
- [21] S. R. B. Gummidi, T. B. Pedersen, and X. Xie, "Transit-based task assignment in spatial crowdsourcing," in *SSDBM2020*. ACM, 2020, pp. 1–12.
- [22] S. R. B. Gummidi, T. B. Pedersen, X. Xie, and E. Zimányi, "Push-based spatial crowdsourcing for enriching semantic tags in openstreetmap," *Unpublished Manuscript*.
- [23] —, "Push-based spatial crowdsourcing for enriching semantic tags in openstreetmap," in *SIGSPATIAL '19*. ACM, 2019, pp. 532–535.
- [24] S. R. B. Gummidi, X. Xie, and T. B. Pedersen, "A survey of spatial crowdsourcing," *ACM Transactions on Database Systems (TODS)*, vol. 44, no. 2, p. 8, 2019.
- [25] U. U. Hassan and E. Curry, "A Multi-armed Bandit Approach to Online Spatial Task Assignment," in *UIC*, 2014, pp. 212–219.
- [26] S. He, D.-H. Shin, J. Zhang, and J. Chen, "Toward optimal allocation of location dependent tasks in crowdsensing," in *INFOCOM*, 2014, pp. 745–753.
- [27] H. Hu, Y. Zheng, Z. Bao, G. Li, J. Feng, and R. Cheng, "Crowdsourced poi labelling: Location-aware result inference and task assignment," in *ICDE 2016*, 2016, pp. 61–72.
- [28] N. Q. V. Hung, N. T. Tam, L. N. Tran, and K. Aberer, "An evaluation of aggregation techniques in crowdsourcing," in *WISE*, 2013, pp. 1–15.
- [29] P. G. Ipeirotis, F. Provost, and J. Wang, "Quality management on amazon mechanical turk," in *ACM SIGKDD workshop on human computation*, 2010, pp. 64–67.
- [30] S. Isaj, S. R. B. Gummidi, T. B. Pedersen, and X. Xie, "Spatial entity linkage with the aid of spatial crowdsourcing," *Unpublished Manuscript*.
- [31] S. Isaj and T. B. Pedersen, "Seed-driven geo-social data extraction," in *SSTD '19*. ACM, 2019, pp. 11–20.
- [32] S. Isaj, E. Zimányi, and T. B. Pedersen, "Multi-source spatial entity linkage," in *Proceedings of the 16th International Symposium on Spatial and Temporal Databases*, 2019, pp. 1–10.
- [33] R. Karam, F. Favetta, R. Kilany, and R. Laurini, "Integration of similar location based services proposed by several providers," in *NDT*, 2010.
- [34] D. R. Karger, S. Oh, and D. Shah, "Iterative learning for reliable crowdsourcing systems," in *Advances in neural information processing systems*, 2011, pp. 1953–1961.
- [35] L. Kazemi and C. Shahabi, "A privacy-aware framework for participatory sensing," *ACM SIGKDD Explorations Newsletter*, vol. 13, no. 1, pp. 43–51, 2011.

## References

- [36] —, “Geocrowd: enabling query answering with spatial crowdsourcing,” in *GIS*, 2012, pp. 189–198.
- [37] L. Kazemi, C. Shahabi, and L. Chen, “Geotrucrowd: trustworthy query answering with spatial crowdsourcing,” in *GIS*, 2013, pp. 314–323.
- [38] F. K. Khattak and A. Salleb-Aouissi, “Quality control of crowd labeling through expert evaluation,” in *NIPS 2nd Workshop on Computational Social Science and the Wisdom of Crowds*, vol. 2, 2011, p. 5.
- [39] J. Kleinberg and E. Tardos, *Algorithm design*. Pearson Education India, 2006.
- [40] I. Koutsopoulos, “Optimal incentive-driven design of participatory sensing systems,” in *INFOCOM*, 2013, pp. 1402–1410.
- [41] L. I. Kuncheva, C. J. Whitaker, C. A. Shipp, and R. P. Duin, “Limits on the majority vote accuracy in classifier fusion,” *Pattern Analysis & Applications*, vol. 6, no. 1, pp. 22–31, 2003.
- [42] K. Lee, J. Caverlee, and S. Webb, “The social honeypot project: protecting online communities from spammers,” in *WWW*, 2010, pp. 1139–1140.
- [43] V. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” in *Soviet physics doklady*, 1966.
- [44] Y. Li, M. L. Yiu, and W. Xu, “Oriented online route recommendation for spatial crowdsourcing task workers,” in *Advances in Spatial and Temporal Databases*. Springer, 2015, pp. 137–156.
- [45] X. Liu, M. Lu, B. C. Ooi, Y. Shen, S. Wu, and M. Zhang, “Cdas: a crowdsourcing data analytics system,” *VLDB Endowment*, vol. 5, no. 10, pp. 1040–1051, 2012.
- [46] K. Mehlhorn and S. Näher, *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, 1999. [Online]. Available: <http://www.mpi-sb.mpg.de/%7Emehlhorn/LEDAbook.html>
- [47] A. Morana, T. Morel, B. Berjawi, and F. Duchateau, “Geobench: a geospatial integration tool for building a spatial entity matching benchmark,” in *ACM SIGSPATIAL*, 2014, p. 533–536.
- [48] R. W. Ouyang, M. Srivastava, A. Toniolo, and T. J. Norman, “Truth discovery in crowdsourced detection of spatial events,” *TKDE*, vol. 28, no. 4, pp. 1047–1060, 2016.
- [49] L. Pournajaf, L. Xiong, V. Sunderam, and S. Goryczka, “Spatial task assignment for crowd sensing with cloaked locations,” in *IEEE MDM*, 2014, pp. 73–82.
- [50] L. Pu, X. Chen, J. Xu, and X. Fu, “Crowdlet: Optimal worker recruitment for self-organized mobile crowdsourcing,” in *INFOCOM*, 2016, p. 5.
- [51] V. C. Raykar, S. Yu, L. H. Zhao, A. Jerebko, C. Florin, G. H. Valadez, L. Bogoni, and L. Moy, “Supervised learning from multiple experts: whom to trust when everyone lies a bit,” in *ICML*, 2009, pp. 889–896.
- [52] F. Restuccia, N. Ghosh, S. Bhattacharjee, S. K. Das, and T. Melodia, “Quality of information in mobile crowdsensing: Survey and research challenges,” *TOSN*, vol. 13, no. 4, p. 34, 2017.

## References

- [53] J. Rogstadius, V. Kostakos, A. Kittur, B. Smus, J. Laredo, and M. Vukovic, "An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets." in *ICWSM*, 2011, pp. 17–21.
- [54] Y. Shen, L. Huang, L. Li, X. Lu, S. Wang, and W. Yang, "Towards preserving worker location privacy in spatial crowdsourcing," in *GLOBECOM*, 2015, pp. 1–6.
- [55] M. Shin, C. Cornelius, D. Peebles, A. Kapadia, D. Kotz, and N. Triandopoulos, "Anonymsense: A system for anonymous opportunistic sensing," *Pervasive and Mobile Computing*, vol. 7, no. 1, pp. 16–30, 2011.
- [56] T. Song, Y. Tong, L. Wang, J. She, B. Yao, L. Chen, and K. Xu, "Trichromatic online matching in real-time spatial crowdsourcing," in *ICDE*, 2017, pp. 1009–1020.
- [57] M. Stevens and E. D'Hondt, "Crowdsourcing of pollution data using smartphones," in *Workshop on Ubiquitous Crowdsourcing*, 2010.
- [58] D. Sun, K. Xu, H. Cheng, Y. Zhang, T. Song, R. Liu, and Y. Xu, "Online delivery route recommendation in spatial crowdsourcing," *World Wide Web*, pp. 1–22, 2018.
- [59] H. To, L. Fan, L. Tran, and C. Shahabi, "Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints," in *PerCom*, 2016, pp. 1–8.
- [60] H. To, G. Ghinita, L. Fan, and C. Shahabi, "Differentially private location protection for worker datasets in spatial crowdsourcing," *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 934–949, 2017.
- [61] H. To, G. Ghinita, and C. Shahabi, "A framework for protecting worker location privacy in spatial crowdsourcing," *VLDB Endowment*, vol. 7, no. 10, pp. 919–930, 2014.
- [62] H. To and C. Shahabi, *Location Privacy in Spatial Crowdsourcing*. Springer International Publishing: In Handbook of Mobile Data Privacy, 2018, pp. 167–194.
- [63] H. To, C. Shahabi, and L. Kazemi, "A server-assigned spatial crowdsourcing framework," *ACM Transactions on Spatial Algorithms and Systems*, vol. 1, no. 1, p. 2, 2015.
- [64] Y. Tong, L. Chen, and C. Shahabi, "Spatial crowdsourcing: Challenges, techniques, and applications," *VLDB Endowment [Tutorial]*, vol. 10, no. 12, pp. 1988–1991, 2017.
- [65] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen, "Online mobile micro-task allocation in spatial crowdsourcing," in *ICDE*, 2016, pp. 49–60.
- [66] Y. Tong, L. Wang, Z. Zhou, B. Ding, L. Chen, J. Ye, and K. Xu, "Flexible online task assignment in real-time spatial data," *VLDB Endowment*, vol. 10, no. 11, pp. 1334–1345, 2017.
- [67] L. Tran, H. To, L. Fan, and C. Shahabi, "A real-time framework for task assignment in hyperlocal spatial crowdsourcing," *ACM Transactions on Intelligent Systems and Technology*, p. 37, 2017.
- [68] J. Whitehill, T.-f. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo, "Whose vote should count more: Optimal integration of labels from labelers of unknown expertise," in *Advances in neural information processing systems*, 2009, pp. 2035–2043.

## References

- [69] O. Wiki, "Verifiability in openstreetmap," <https://wiki.openstreetmap.org/wiki/Verifiability>, accessed November 28, 2019.
- [70] R. F. Woolson, J. A. Bean, and P. B. Rojas, "Sample size for case-control studies using cochrane's statistic," *Biometrics*, pp. 927–932, 1986.
- [71] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *ACL*, 1994.
- [72] P. Yang, N. Zhang, S. Zhang, K. Yang, L. Yu, and X. Shen, "Identifying the most valuable workers in fog-assisted spatial crowdsourcing," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1193–1203, 2017.
- [73] B. Zhang, C. H. Liu, J. Lu, Z. Song, Z. Ren, J. Ma, and W. Wang, "Privacy-preserving qoi-aware participant coordination for mobile crowdsourcing," *Computer Networks*, pp. 29–41, 2016.
- [74] L. Zhang, X. Lu, P. Xiong, and T. Zhu, "A differentially private method for reward-based spatial crowdsourcing," in *ATIS*, 2015, pp. 153–164.
- [75] X. Zhang, Z. Yang, Y.-J. Gong, Y. Liu, and S. Tang, "Spatialrecruiter: maximizing sensing coverage in selecting workers for spatial crowdsourcing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 5229–5240, 2017.
- [76] X. Zhang, Z. Yang, Y. Liu, and S. Tang, "On reliable task assignment for spatial crowdsourcing," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2016.
- [77] Y. Zhao, Y. Li, Y. Wang, H. Su, and K. Zheng, "Destination-aware task assignment in spatial crowdsourcing," in *CIKM*, 2017, pp. 297–306.
- [78] Y. Zhao and Q. Han, "Spatial crowdsourcing: current state and future directions," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 102–107, 2016.

# **Part II**

# **Papers**





# Paper A

## A Survey of Spatial Crowdsourcing

Srinivasa Raghavendra Bhuvan Gummidi, Xike Xie, and Torben  
Bach Pedersen

The paper has been published in the  
*ACM Transactions on Database Systems*  
vol. 44, no. 2, p. 8, 2019. DOI: <https://doi.org/10.1145/3291933>

## Abstract

*Widespread usage of advanced mobile devices has led to the emergence of a new class of crowdsourcing called spatial crowdsourcing. Spatial crowdsourcing advances the potential of a crowd to perform tasks related to real-world scenarios involving physical locations, which were not feasible with conventional crowdsourcing methods. The main feature of spatial crowdsourcing is the presence of spatial tasks that require workers to be physically present at a particular location for the task fulfillment. Research related to this new paradigm has gained momentum in recent years, thus necessitating a comprehensive survey to offer a bird's eye view of the current state of spatial crowdsourcing literature. In this paper, we discuss the spatial crowdsourcing infrastructure and identify the fundamental differences between spatial and conventional crowdsourcing. Furthermore, we provide a comprehensive view of the existing literature by introducing a taxonomy, elucidate the issues/challenges faced by different components of spatial crowdsourcing, and suggest potential research directions for the future.*

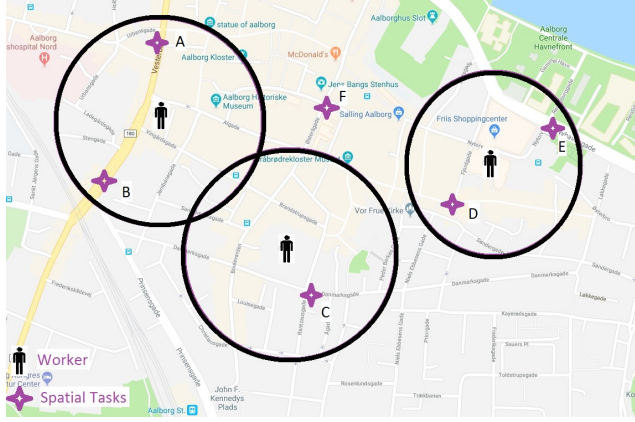
© 2019 ACM. Reprinted, with permission from Srinivasa Raghavendra Bhuvan Gummididi, Xike Xie, and Torben Bach Pedersen. A Survey of Spatial Crowdsourcing. In: *ACM Transactions on Database Systems*, vol. 44, no. 2, p. 8, 2019. <https://doi.org/10.1145/3291933>  
*The layout has been revised.*

# 1 Introduction

The proliferation of advanced mobile devices opened up a new crowdsourcing avenue (spatial crowdsourcing), to harness the potential of the crowd to perform real-world tasks with a strong spatial nature, which are not supported by conventional crowdsourcing (CC) techniques. CC techniques [9] focus on transactions that are carried out entirely through the medium of the Internet. However, in the real-world crowdsourcing scenarios such as crowdsourcing disaster response [140] and news reporting [118], tasks are likely to have spatial requirements that cannot be fulfilled virtually and require physical on-location operations. Spatial crowdsourcing (SC) is a particular class of crowdsourcing; that deals with such spatial tasks.

The phrase “Spatial Crowdsourcing” was introduced in 2012 by Kazemi et al [61], though the technique was already in use for some years. The eBird (<https://ebird.org>) project [100] is one of the earliest SC-based applications for collecting information about bird sightings from the bird-watching enthusiasts. SC is also referred in the literature by the following phrases: “Place-Centric Crowdsourcing” [21, 59, 116], “Location-based Crowdsourcing” [2, 7, 117], “Participatory Sensing” [10, 32, 40], “Mobile Crowdsourcing” [11, 127], and “Mobile Crowdsensing” [44, 124]. To elaborate, “Participatory Sensing” and “Mobile Crowdsensing” belong to a subset of SC, that involves utilization of smartphones of users to collect the different sensor data of the smartphones at different locations. In general, a typical participatory sensing or mobile crowdsensing application does not harness the wisdom or knowledge of the crowd. However, SC offers avenues to utilize both the sensor information as well as the knowledge possessed by the humans. A participant’s skillset can be considered during recruitment for performing a given job or task. Furthermore, in SC, the human/participant effort is often associated with constraints such as spatiotemporal availabilities, capabilities, and rewards. Such limitations make human knowledge and common sense a valuable resource that should be carefully planned than in the case of mobile crowdsensing. Therefore, techniques for effective utilization of human knowledge are desired and developed, like task matching and subsequent scheduling of tasks agreed by the participant. For example, consider the SC application for collecting traffic information at different spots in the city, as described below.

**Spatial Crowdsourcing Example:** Fig. A.1 depicts an SC example of collecting traffic information at different points in the city, with the help of the crowd. The figure shows three participants with neighbourhood boundaries depicted by the black circles around the participant location, along with the different neighbouring tasks. The participants can only collect the traffic information at the task locations that lie in their neighbourhoods. The tasks



**Fig. A.1:** Spatial Crowdsourcing: Performing different traffic data collection tasks in the city

involve harnessing the human knowledge to retrieve qualitative details, utilization of the sensors in the participant’s smartphone, or both. For instance, tasks A & E involve capturing pictures of the road at a particular time. Tasks B & C consist of estimating the volume of traffic that passes through that point in a 5-minute interval during a predefined period of the day. Tasks D & F requires inputs from participants to verify whether a traffic jam is typical during a particular time of the day at those task locations. The SC application attempts to match the participants with the tasks based on their availabilities and constraints. Moreover, the SC application offers to schedule the accepted tasks as well.

Typically, the tasks involved in SC require movement of the contributor or worker to the tasks’ locations, to perform the task. The task’s requester would specify the task’s location along with the necessary details like task completion deadlines, task descriptions, associated rewards, skills, and reputation expected from the worker. SC has the potential for collecting information for a broad range of applications in domains like environmental data collection (NoiseTube platform [99]), transportation (Uber, <https://www.uber.com>), journalism [69, 70], and business intelligence (Gigwalk, <http://www.gigwalk.com> and TaskRabbit, <https://www.taskrabbit.com>). In addition to the data collection and query answering tasks, SC can be extended to service complex spatial tasks like employing a set of workers with diverse skills to renovate the house, hiring workers to perform household chores, etc.

Recently, research on SC has gained momentum; consequently, many techniques are proposed for various application scenarios. Therefore, compiling and organizing the existing research regarding SC will be beneficial for researchers to gain a comprehensive view of current research and potential directions for future studies. Although there exist surveys of conventional

## 1. Introduction

crowdsourcing like [20, 71, 80, 130], none of them delves into the topic of SC in detail. Recently, a short article [137] briefly discussed the current state and existing challenges. However, [137] does not offer a technical perspective and a comparison between the existing strategies in SC. Similarly, a tutorial reviewing the current state of SC research was published in August 2017 [111]. In comparison, our survey additionally offers a comprehensive analysis of the relationships between different types of constraints and compares different optimization problems in detail and highlight the limitations. Furthermore, our survey classifies the spatial tasks, and discusses the truth inference models and data aggregation methods in SC. Similarly, the survey [41] presents an overview of existing applications on crowdsensing until mid-2011. It does not address technical details such as task matching, assignment, and scheduling which are crucial for SC data management. Recently, the survey [90] has summarized the main aspects of Quality of Information (QoI) in mobile crowdsensing. However, the survey only deals with the estimation methods of QoI present in the current mobile crowdsensing literature. The present paper provides all these contributions and aims to improve the reviews above by performing a comprehensive review of SC by surveying the research done in the area until May 2018. This paper mainly focuses on the techniques and their comparisons along with the applications pertaining specifically to SC. We provided an overview of the different problems in SC as well as how different constraints of SC impact each other. This paper summarizes all the major aspects of SC, however it does not delve into the security and privacy aspects in detail like the survey [37] which focuses only on the security, privacy and trust aspects of mobile crowdsourcing.

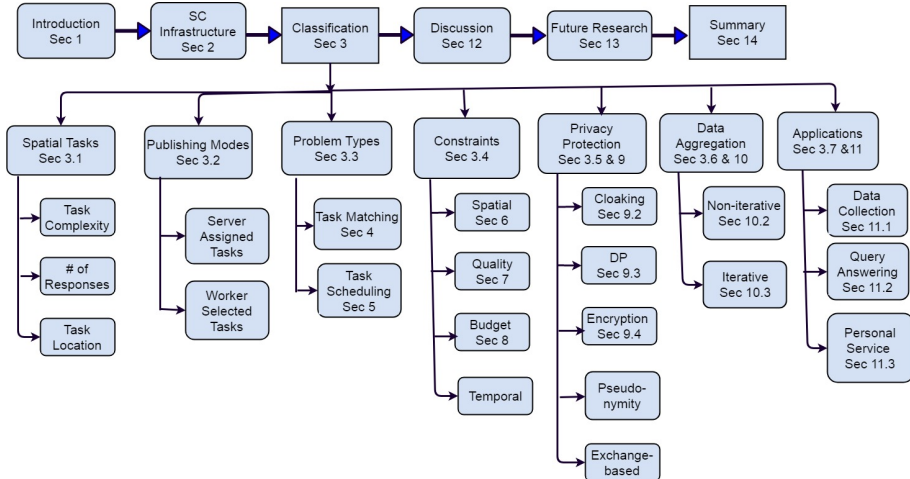


Fig. A.2: Structure of the Paper

This paper is organized into different sections as shown in Fig. A.2.

*Section 2: Spatial Crowdsourcing Infrastructure* describes in detail the unique features of SC relative to CC concerning the building blocks of crowdsourcing. *Section 3: Classification* introduces our taxonomy for organizing the existing literature into different broad categories based on various criteria like types of spatial tasks, task publishing modes, type of optimization problems, addressed constraints, privacy protection techniques, data aggregation techniques, and type of applications. As can be seen in Fig. A.2, the part on classification contains the majority of the material in the paper and would have led to deep nesting in its hierarchy. To improve the readability, we have instead decided to flatten the hierarchy by discussing the prominent sub-categories in separate sections.

Accordingly, the two types of problems in SC are discussed in *Section 4: Task Matching Problem* and *Section 5: Task Scheduling Problem*, respectively. Constraints that affect the setting of such problems are discussed in *Section 6: Spatial Constraints*, *Section 7: Quality Constraints*, and *Section 8: Budget Constraints*, respectively<sup>1</sup>. Additionally, a comparison is performed between different techniques of SC and the limitations are identified, based on the association among various constraints. *Section 9* details the different privacy protection techniques and the impact of privacy protection on task matching problems. *Section 10* describes the data aggregation techniques that are commonly employed for truth detection in SC. *Section 11: Applications* details three common types of SC applications. The challenges/issues posed in the reviewed SC literature are discussed in *Section 12: Discussion*. Finally, in *Section 13: Future Research Directions* and *Section 14: Summary*, we provide potential research directions and the summary.

## 2 Spatial Crowdsourcing Infrastructure

Spatial crowdsourcing infrastructure comprises of four major components: requester, spatial task, worker, and SC-server (see Fig. A.3). This section elaborates each SC infrastructure component and outlines the differences concerning the core components in both CC [49] and SC [61] systems.

### 2.1 Requester

A requester is a real-world entity like a person or an organization that requests for a particular spatial task to be completed by the crowd at a specific location (see Fig. A.3). Similar to the CC, the requester is the initiating point in the SC process. A requester designs the task and sets the conditions that need to be satisfied for performing the task. The requester has certain responsibilities such as accepting the answers provided or data collected by the

---

<sup>1</sup>Temporal constraints are minor issues in the context of SC, and are not discussed in detail.

## 2. Spatial Crowdsourcing Infrastructure

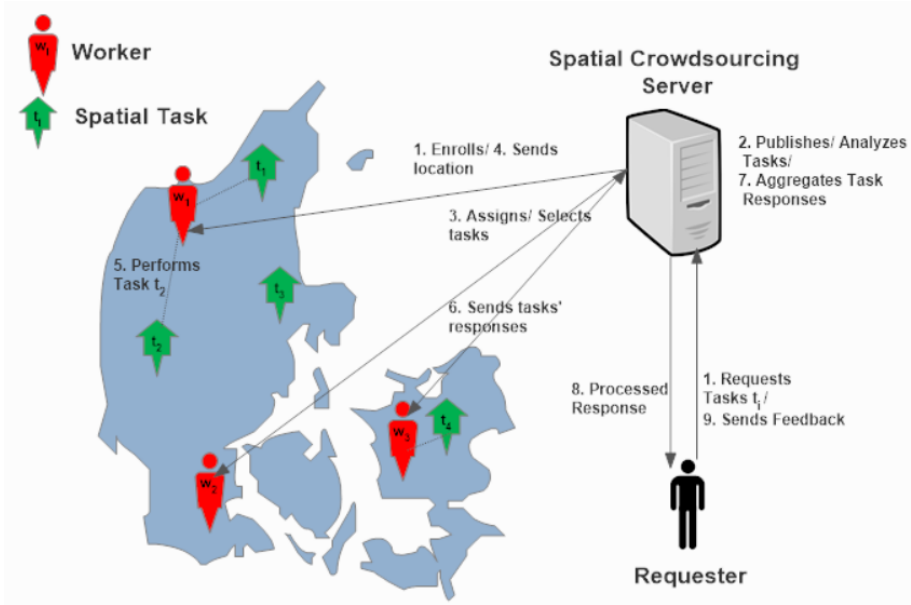


Fig. A.3: Spatial Crowdsourcing Workflow Scenario

crowd and giving them feedback [14]. The role of the requester is the same in SC and CC, except for the geospatial knowledge required while specifying the spatial tasks and the responsibility to respect the privacy of the worker by not exploiting the worker's task location visit information.

### 2.2 Worker

A worker is a person that participates in the process of SC, with an objective to perform the assigned/selected spatial task. The main difference between the features of the worker mentioned in the CC system and the SC is the motivation to **physically move** to a particular geographical location to perform a spatial task.

**Worker Definition:** Different definitions mentioned in the literature for a worker  $W$ , like:

$$W = \langle l, R, \max T, Re, E, \theta \rangle \quad (\text{A.1})$$

Most of the definitions essentially had the following basic and optional attributes:

Basic attributes [61]

- Current physical location of the worker  $l$
- Region of interest  $R$

- Maximum number of tasks  $maxT$

Optional attributes:

- Expected reward  $Re$  [27]
- Skillset  $E$  [27]
- Worker's reputation score calculated by SC-server  $\theta$  [62]

## 2.3 Spatial task

As discussed earlier, a spatial task is fundamentally different from the CC task due to the condition that the worker should be physically present at the task's location. A spatial task is requested by the requester and is fulfilled by the worker. It is a location-specific activity like answering a question about a local restaurant, taking pictures of a local tourist spot, or collecting noise pollution data. Moreover, a SC task will be defined by the requester with a set of requirements for assigning or allowing a person from the crowd to work on the task. The requirements might include the minimum level of skills required to fulfil a task, the number of answers needed, the expertise of the user, the task deadline, and their experience in solving similar tasks.

**Spatial Task Definition:** Different definitions mentioned in the literature for a spatial task  $t$ , like:

$$t = \langle l, q, ti_i, ti_e, r, \alpha, E, Cat \rangle \quad (A.2)$$

Most of the definitions essentially had the following basic and optional attributes:

Basic attributes [61]

- Physical location of the task  $l$
- Query description  $q$
- Issuing time  $ti_i$
- Expiration time  $ti_e$

Optional attributes:

- Associated reward  $r$  [27]
- Minimum threshold for worker's reputation  $\alpha$  [62]
- Expected worker's skillset  $E$  [15]
- Type/category of task  $Cat$  [47]



- Maximum number of workers  $maxW$  [134]

### 2.4 Spatial Crowdsourcing Server

Through the SC-server, the requester requests a task to be performed by the crowd, the workers register to be assigned or select the tasks. It delegates the communication between the requester and the worker of the spatial task, facilitates the process to satisfy the task requirements, assigns tasks to workers based on location, helps improving the quality of the outcome by executing different strategies, identifies the anomalies and detects fraudulent responses, and protects the privacy of the involved stakeholders. Furthermore, there are additional functionalities for SC related to communication between different stakeholders.

**Communication between different stakeholders:** SC-server enables the broadcast of information between the requesters who request tasks and the crowd [14, 49]. According to [49], there are four major features for a CC platform: “Crowd-related interactions”, “Requester-related interactions”, “Task-related interactions”, and “Platform-related facilities”. This categorization is relevant for the SC paradigm as well. In addition to the features mentioned before in [49], some additional features need to be adapted in the case of SC. The adaptations are:

#### a. For the Crowd

- The power to disclose or hide their locations to the SC-server.
- Ability to select tasks that need to be performed near their location.
- Ability to specify a spatial region in which they wish to work, i.e., the geographical extent they can travel for performing a spatial task.
- Ability to provide location privacy-protection of workers.
- Ability to reject assigned tasks.

#### b. For Requested Tasks

- Determining the maximum acceptable distance of the workers in the vicinity of the task.
- Defining the geographical location where the requested task needs to be performed.
- Ability to hide the location where the task needs to be performed or only to be shown to workers who accepted or selected the task.

		Spatial Task Publishing Modes			
		Server Assigned Tasks		Worker Selected Tasks	
Constraints	Spatial con- straints	a. Spatial Region [26, 61, 62] b. Direction of worker's Commute [12, 17]		Spatial Region [29, 73]	
	Quality Con- straints	a. Worker Reputation: [17, 62, 88] b. Worker Expertise: [15, 27, 109] c. Truth Inference [45, 50, 85]		Qualification tests [76, 92]	
	Temporal Con- straints	a. Deadline of Task [61] b. Delivery Task PickUp time and Deadline [101] c. Deadline for Destination [73]		Deadline for task [29]	
	Budget Con- straints	Reward models and Incentive Mechanisms [27, 38, 48, 64, 129]		Fixed Rewards for Tasks <sup>2</sup> [2]	
Scenarios		Online	Offline	Online	Offline
Types of Problems	Task Matching	MAB [47], GOMA [113], f-MTC, d-MTC [104, 115], FTOA [114], TOM [98]	MTA [61], MSA [109], MTMCA [27], RDB-SC [17],MRA [135], PAPA [60], DPTA [106], MS-SC [15]	-NA-	- NA-
	Task Scheduling	GALS [30], TRACCS [12]	MTS [29]	OnlineRR [73], Auction-SC [5]	MTS [29]
Privacy Protection		a. Differential Privacy-based [105, 106, 133] b. Spatial Cloaking [60, 87] c. Encryption-based [96, 97]		a. Pseudonymity [22] b. Exchange-based [132]	

**Table A.1:** Classification of Spatial Crowdsourcing based on different constraints

**Spatial Task Lifecycle:** Fig. A.3 illustrates the lifecycle of a typical SC task. The lifecycle of a spatial task begins with the requester requesting the SC-server to facilitate the fulfilment of specified spatial tasks (Step 1 of Fig. A.3). The SC-server publishes the requested tasks actively by pushing them to selected workers or passively by publishing them as a list from which the workers can choose based on their interests (Steps 2 & 3 of Fig. A.3). The worker performs the assigned/chosen task and responds to the SC-server with the collected information/answer for the task query (Steps 5 & 6 of Fig. A.3). The SC-server processes/aggregates the responses from workers and delegates them to the requesters (Steps 7 & 8 of Fig. A.3). The requesters validate the task responses and provide feedback regarding the quality of the task response (Step 9 of Fig. A.3). The workflow is similar to the lifecycle of a CC task, except for the consideration of spatial characteristics like travel time/distance between potential workers and spatial tasks, and the spatiotemporal behaviour of workers.

### 3 Classification

The current literature in SC can be classified into distinct categories based on the following criteria: the type of problem addressed, the modes of publishing spatial tasks, the different constraints considered, the type of application, the type of spatial tasks, and the different privacy preserving and data aggregation techniques employed. This section provides a brief overview of the different categories and elaborates them in the following sections (see Table A.1).

#### 3.1 Spatial Tasks

Similar to CC tasks, we can classify spatial tasks based on their complexity and the required number of workers. Additionally, we can categorize them by the spatial extent of the task's location.

**Spatial task complexity:** Tasks are classified into two types based on their complexity.

- **Atomic tasks:** An atomic task, as the name implies, cannot be divided down into sub-tasks. Atomic tasks are simple and can be performed by one worker [61]. An example of an atomic task is to answer yes or no to the query "Is it raining now?".
- **Complex tasks:** A complex task consists of two or more related activities that need to be performed collaboratively for accomplishing the spatial task. These complex tasks will be divided into atomic sub-tasks that require specific skills [7, 15, 61, 109]. For instance, a complex task like renovating a shop at a particular location involves multiple activities like procuring materials, performing repairs, and painting.

**Number of responses:** A spatial task can be assigned to either one or more workers. By assigning to multiple workers, the quality of the outcome can be improved by being able to assess the majority answers. However, it is not certain to receive responses regarding the assigned tasks from the workers; they can either reject or ignore the tasks. In case the worker does not reply; then the server will wait till the expiration time of the spatial task [47]. A spatial task can be classified into two types based on the number of responses, single and multiple responses.

**Task's Physical location:** The task location could be a particular point, a line segment or a region in geographical space.(See Fig. A.4)

---

<sup>2</sup>[www.mturk.com](http://www.mturk.com)

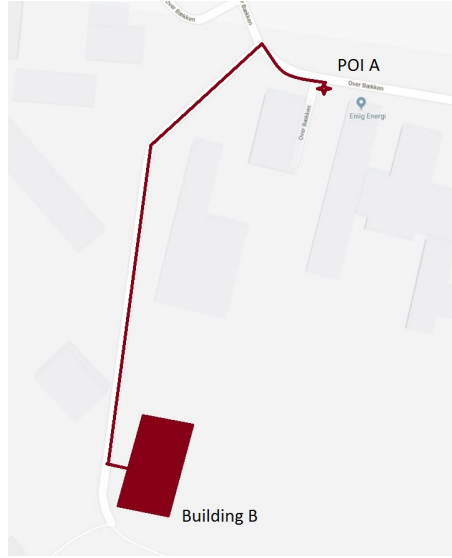


Fig. A.4: Spatial task type based on task's location condition

- **Point task:** The spatial task is required to be performed at that particular point location [51, 61, 62]. For example, a worker has to visit the point of interest *A* (As seen in Fig. A.4) for performing a task like enquiring the day's menu of a university cafeteria, where the worker has to take a picture of the menu details of the cafeteria to complete the task.
- **Area/Region task:** The spatial task is required to be performed by the worker in a region of geographical space, instead of a particular location [104]. For example, a worker has to visit the building *B* in the university campus for collecting noise pollution data of a building. The task can be carried out by collecting information at any point in the building.
- **Delivery Task:** The spatial task has two parts: pick up the package from the point of interest *A* and deliver the package to the building *B* [101]. For example, collecting a food package from the restaurant at location *A* and delivering it to the customer located at *B*.

### 3.2 Modes of Publishing Spatial Tasks

There are two ways of publishing spatial tasks [61] on the SC-server:

- **Server Assigned Tasks:** The SC-server chooses available suitable workers for a given spatial task based on different parameters like their proximity to the spatial task, availability to perform the task, abilities to

### 3. Classification

match the requirements of the task, reliability of the worker to assure some degree of quality in task outcomes.

- **Worker Selected Tasks:** Workers select spatial tasks of their interest from a given list published by the SC-server.

The two task publishing modes, Server Assigned Tasks (SAT) and Worker Selected Tasks (WST) are discussed implicitly along with the other classification categories, as every work in SC follows either one of them. Furthermore, most of the existing work is dominated by the SAT task publishing mode as the SC-server can exert control in this mode by pushing tasks to workers. On the contrary, in the WST mode workers select the tasks based on their interest with little or no influence from the SC-server. Therefore, the SAT mode is discussed more in detail when compared to WST mode.

### 3.3 Types of Problems

We can classify the numerous optimization problems that are addressed in the SC literature into two different categories: Task Matching and Task Scheduling.

- **Task Matching:** Given a set of spatial tasks  $T$  and a set of workers  $W$ , the SC-server tries to match the workers to the spatial tasks. This problem exists only when the SC-server publishes the task through *Server Assigned Tasks* mode [61] (discussed in Section 4).
- **Task Scheduling:** Given a set of tasks that are selected by the worker [29] or assigned to a particular worker [30], the task scheduling problem tries to schedule the order of tasks to achieve different optimization goals like maximizing the number of tasks completed by a worker, maximizing the rewards received by the workers, etc. The problem of task scheduling is unique to SC (discussed in Section 5).

Besides, the algorithms proposed for solving the problems of task matching and task scheduling require knowledge of the inputs like worker's location at different times of the day and tasks information. Based on the knowledge of the inputs, we can categorize them into two scenarios (discussed in Section 4 and 5):

- **Offline Scenario:** In this scenario, the information about the inputs is available to the SC-server from the beginning, i.e., the algorithm has the complete knowledge about the inputs that arrive at a future time. In this scenario, the algorithms can maximize the global objective functions like maximizing the number of tasks assigned [61]. This scenario, though not effective in real-world case scenarios of SC, would help understand the difficulties in the online scenario, due to the uncertainty

of inputs arrival to the system. This scenario is effective in certain special cases where the schedule of the worker and the arrival of tasks are known beforehand to the system.

- **Online Scenario:** In this scenario, the algorithm receives inputs in a streaming fashion; and thus, the algorithm has to process each input immediately [58]. This scenario represents the majority real-world case scenarios of SC, where the workers and tasks are dynamic [5, 98, 112–114], i.e., their arrival orders are not known beforehand.

**Optimization goals:** Typically, task matching and task scheduling problems described in the SC literature optimize certain aspects like maximizing the number of tasks assigned. The optimization goals can be categorized from either the perspective of workers or the SC-server [137] (discussed in Section 4 and Section 5):

- **Worker’s Perspective:** From a worker’s perspective, the goal would be to maximize the net reward that she receives from the SC-server for performing the assigned or selected tasks [113]. Generally, the net reward is the difference between the reward associated with the task and the travel cost incurred to travel to the location of the task. In the case of voluntary SC, the worker’s goal would be to minimize the total travelling distance.
- **SC-server’s Perspective:** From the SC-server’s perspective, the goal would be to maximize the number of assigned tasks [61], maximize the number of accepted tasks [106], maximize the quality score [17, 109], minimize the incentives paid to the worker [27], minimize the total travel costs of all workers [30], maximize the perfect matching [35] with no unstable pairs [123] and maximize the number of tasks completed by worker before deadline [29].

### 3.4 Constraints

In an ideal situation, the spatial tasks can be assigned to all the workers to retrieve better results. However, this is not the case in reality, due to the different constraints of the workers like preferred region of work, preferred task types, and the travelling cost taken for performing the task. We can classify the various constraints broadly into the following four categories:

- **Spatial Constraints:** relate to the spatial preferences of a worker or a task, like the preferred locations where she likes to work [61], the maximum travel distance she is willing to travel to perform tasks [12], and the preferred region within which the task should be performed [104] (discussed in detail in Section 6).

### 3. Classification

- **Temporal Constraints:** relate to the deadline of the task [61], or the time duration during which the worker is available for work [73]. Temporal constraints are common CC constraints; hence they are not further detailed.
- **Quality Constraints:** refer to the expected quality of the responses from the workers after performing the task. They usually involve pre-task qualification tests [92] or assignment based on previous histories [62]/abilities [27] of the worker (discussed in detail in Section 7).
- **Budget Constraints:** refer to the restrictions on the incentive mechanism in rewards-based SC, where reward will be offered to workers for performing the task. Some of the constraints are worker's expected reward or total threshold reward for the requester's requested tasks (discussed in detail in Section 8).

### 3.5 Privacy Protection

Due to the sensitive nature of the worker's location, preserving location privacy is the responsibility of the SC-server. The worker might not trust the SC-server to share her physical location. Therefore, workarounds are proposed to address the privacy concerns like using the concept of differential privacy [106]. The privacy-preserving techniques can be categorized into the following five types [108] (discussed in detail in Section 9):

- **Pseudonymity Techniques:** uncouple the person's identity and the submitted data [22].
- **Cloaking Techniques:** hide the exact locations of the workers in a cloaked region [60, 87].
- **Exchange-based Techniques:** exchange the crowdsourced information among the workers before disclosing it to the untrusted SC-server, to obscure the individual workers. [132]
- **Encryption-based Techniques:** hide the identity and the workers' location from the SC-server [96, 97]
- **Differential Privacy-based Techniques:** distort the workers location information by adding artificial noise [105, 106, 133]

### 3.6 Crowdsourced Data Aggregation

In many cases, tasks in SC require multiple responses from the workers, for example for ascertaining the traffic situation of an area. However, the multiple responses from workers may be both agreeing and conflicting. To

establish the truth, the crowdsourced data needs to be aggregated and relayed back to the task requester as a single value. There are numerous methods in CC for crowdsourced data aggregation, categorized based on their computing model [53]:

- **Non-iterative Aggregation:** aggregates the responses for each question to a single value. Examples are Majority Voting [65], HoneyPot [68], and Expert Label Injected Crowd Estimation(ELICE) [63].
- **Iterative Aggregation:** calculates the aggregated value iteratively for each question based on the expertise of the workers who answered and updates the worker expertise iteratively based on the answer given by the worker. Examples are Expectation Maximization (EM) [54], Generative Model of Labels, Abilities, and Difficulties (GLAD) [122], Supervised Learning from Multiple Experts(SLME) [89], and Iterative Learning(ITER) [57].

For truth inference, SC extends the above-mentioned data aggregation methods by considering different spatial attributes like distance to the task from the worker’s location [50], task location popularity/influence, location visit tendencies [85]. These methods are discussed in Section 10.

### 3.7 SC Applications

Many applications are developed implementing the methods proposed in SC-literature, for serving different purposes like collecting data during disasters, environmental data collection, delivering packages, and ride sharing services. We can broadly classify the existing applications into three categories based on the nature of human involvement, data collection, query answering, and personal service. The three common types of applications are discussed in detail in Section 11.

The remaining sections are organized based on the *type of problems* and the different *constraints* considered. We have chosen this organisation instead of organizing the sections according to the classification schema, as the latter would have led to massive redundancy among sections. For example, the majority of the SC literature fall within the two types of task publishing modes. Therefore, they are discussed implicitly as part of each section to avoid redundancy.

## 4 Task Matching Problem

### 4.1 Introduction

As discussed earlier, the task matching problem involves assigning a given set of tasks to suitable workers based on various conditions. The problem



## 4. Task Matching Problem

exists exclusively for the *Server Assigned Tasks* publishing mode as the *Worker Selected Tasks* publishing mode does not require the SC-server to choose the workers [61]. Typically, a task matching problem aims to achieve optimization goals benefitting the SC-server. For instance, maximizing the number of tasks assigned [61, 109], minimizing the cost incurred by the server [26, 115], improving the quality of task responses [62] or goals benefitting the workers like maximizing the reward received by the worker [12].

To define the task matching/assignment problem, let us say that at a given time instance, there are a set of tasks  $(t_1, t_2, t_3, \dots)$  requested to the SC-server by the requesters [109]. These tasks consist of the task/query  $(q)$  that needs to be performed at location  $(l)$  before the deadline. These tasks will be assigned to the workers available at the particular time instance. The available workers are determined through the “Task Inquiries” [109] made by the workers. Through these *Task Inquiries*, workers  $(W_1, W_2, W_3, \dots)$  share their locations  $l_i$  to the SC-server along with their constraints like spatial region and the maximum number of tasks that can be performed per time instance. The task assignment set contains the pairs of worker-spatial task matches  $(\langle W, t \rangle)$  at time instance  $s$ .

### 4.2 Offline Scenario: Known Arrival order of Tasks and Workers

As mentioned in Section 3.3, in the offline scenario the SC-server has complete knowledge about the inputs that arrive at a future time, in this case, the tasks’ and workers’ arrival order. Given a set of workers and tasks, the ideal way of solving the matching problem is to assign all the workers to all the tasks. However, in reality, it is not possible owing to the different constraints of the spatial task and the workers. For instance, one would end up with a scenario where workers are required to travel long distances just to solve the task, which is not likely to be accepted by the worker to perform the spatial task. However, a higher number of assignments results in solving a greater number of spatial tasks requested by the requesters. This leads to an optimization problem of **maximizing the number of task assignments** by the SC-server (“Maximum Task Assignment Problem”) [61]. Similarly, some of the other optimization problems according to different optimization objectives can be defined as:

- **Maximum Score Assignment Problem** [109]: The objective is to maximize the overall worker expertise score that results in higher quality results, where  $score(w, t)$  is the value indicating the compatibility between worker  $w$  and task  $t$  based on worker expertise.
- **Maximum Task Minimum Cost Assignment** [27]: The objective is to maximize the number of expert matches while minimizing the total

cost paid to workers.

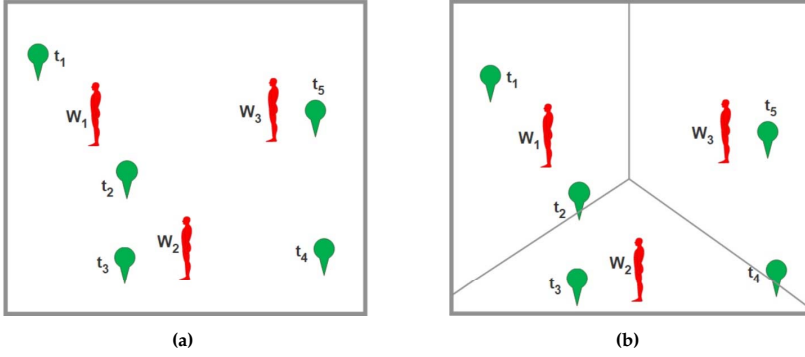
- **Maximum Task Scheduling with Multiple Workers [30]:** The objective is to maximize the number of completed tasks and minimize the sum of the average travel cost per task over all the workers
- **Utility-aware Social Event-participant Planning [95]:** The objective is to maximize the total utility score while satisfying the travel budget constraints.
- **Offline Latency-oriented Task Completion problem [131]:** The objective is to minimize the latency or the number of workers required for performing a set of tasks with high quality.
- **Weighted Spatial Matching [123]:** The objective is to find a fair assignment where there are no unstable pairs.
- **Minimum-cost Maximum Task Assignment Problem [61, 109]:** The objective is to maximize the number of assigned tasks while minimizing the total travel cost.
- **Maximum Quality Task Assignment Problem [16]:** The objective is to maximize the overall quality score of assignments under travelling budget constraints across multiple time instances.

The ways to address this maximum task assignment optimization problem, by considering the constraints, will be discussed in the next sections. To understand the offline variant of the task matching problem, let us consider the following example.

**Assignment Problem Example:** Fig. A.5a shows three workers with five spatial tasks in a 2D region. The workers  $\{W_1, W_2, W_3\}$  are required to collect pictures from the locations of the spatial tasks  $\{t_1, t_2, t_3, t_4, t_5\}$ . The spatial tasks were provided to the SC-server by the requester. Table A.2 provides the information of worker task preferences and rewards associated with tasks. Our goal is to assign workers to these tasks based on the conditions mentioned by the requester. Kindly note that the example problem will be used throughout this survey to understand the different algorithms mentioned in the SC literature.

Devoid of any constraints, the above-mentioned task matching problem can be solved by assigning the tasks near the workers [60]. To solve this problem, the SC-server computes a *Voronoi diagram* based on the locations sent by the workers at a particular time instance. Subsequently, the SC-server assigns the spatial tasks close to the workers based on their *Voronoi cells*. Applying this solution to our SC assignment problem, a *Voronoi diagram* with the three workers in the 2D region is generated(see Fig. A.5b) and the assignment is mentioned in Table A.3.

#### 4. Task Matching Problem



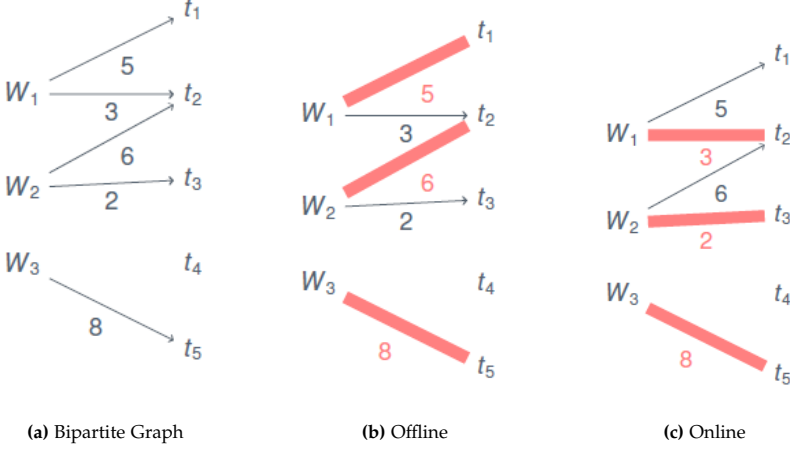
**Fig. A.5:** a. Base Problem:  $\{W_i\}$  represents workers and  $\{t_i\}$  represents spatial tasks. b. Task Matching: Voronoi cells of the workers' locations.

Worker ID	Preferred Spatial Task	Reward
$W_1$	$t_1$	5
	$t_2$	3
$W_2$	$t_2$	6
	$t_3$	2
$W_3$	$t_5$	8

**Table A.2:** Workers task preferences and rewards

Worker ID	Assigned Spatial Tasks
$W_1$	$t_1$
$W_2$	$t_2, t_3, t_4$
$W_3$	$t_5$

**Table A.3:** Assignment with Voronoi Diagrams



**Fig. A.6:** Offline and Online Scenario: Maximum Weighted Bipartite Matching with total reward of 19 and 13 respectively

However with the constraints mentioned in Table A.2, the task matching problem with the goal of maximizing the rewards received by workers, is solved by reducing it to a *Maximum Weighted Bipartite Matching* (MWBM) problem [94]. To reduce the SC task matching problem to the MWBM problem, a bipartite graph  $(G = (V, E))$  is created with vertices  $V = \mathcal{W} \cup \mathcal{T}$ , where each worker  $W_j$  maps to a vertex in set  $\mathcal{W}$  and each task  $t_k$  maps to a vertex in set  $\mathcal{T}$  [109]. A vertex  $W_j$  in  $\mathcal{W}$  is connected to a vertex  $t_k$  in  $\mathcal{T}$  with an edge  $e_{j,k} \in E$ , if the task  $t_k$  is a preferred task of the worker  $W_j$  (see Fig. A.6a). The weight of the edge  $e_{j,k}$  is the reward associated with the preferred task. By solving the MWBM problem with the assumption that one worker can only perform one task, the workers are assigned to the tasks that give them the highest reward [109]. The worker  $W_1$  has edges with both  $t_1$  and  $t_2$ , however  $t_1$  is assigned as it offers better reward than  $t_2$ . Similarly,  $W_2$  has been assigned to  $t_2$  and  $W_3$  has been assigned to  $t_5$  (see Fig. A.6b). As none of the workers preferred  $t_4$ , it is not assigned to anyone. A total reward of 19 is achieved in the offline scenario.

### 4.3 Online Scenario: Dynamic Arrival of Tasks and workers

In the above example, all the workers and tasks are known to the SC-server, however, in a real-world scenario, the solicitation of tasks and the availability of workers are dynamic in nature. The number of tasks cannot be constant as the requesters can provide new tasks to the SC-server at every instance of time and the existing tasks can expire once their deadline is reached. Similarly, regarding the number of available workers, they can increase or decrease at every instance of time. Therefore, the SC-server does not have the

#### 4. Task Matching Problem

*global knowledge* required for assigning the spatial tasks to the workers in a globally optimal way. This input model scenario is referred as the **online scenario** [58]. In other words, the SC-server has no information regarding the arrival orders of the workers and tasks to the system. Owing to this constraint, a *locally optimal* solution can be found for the spatial task assignment at every time instance. The locally optimal solution at every time instance is then combined to provide the solution of the online task matching problem.

A locally optimal solution is akin to the offline scenario, as the available workers and tasks are known to the system at the current time instance. Therefore, the above-mentioned offline methods can be employed to solve the optimization problem at every time instance. However, in the case of online scenario, only a partial bipartite graph can be constructed as the arrival order of tasks and workers is unknown. Therefore, the locally optimal assignment will be performed on the partial bipartite graph at each time instance, resulting in sub-optimal results. Considering the previous example in Section 4.2, let us consider the order of arrival of tasks and workers as  $\langle W_1, t_2, W_2, t_4, t_1, W_3, t_3, t_5 \rangle$ . When  $t_2$  arrives, the available worker  $W_1$  will be assigned to it, even though it offers a lesser reward when compared to  $t_1$ , that arrives at a later time instance. Similarly  $W_2$  is assigned to  $t_3$  and  $W_3$  to  $t_5$  (see Fig. A.6c). The total reward is reduced to 13, whereas offline scenario achieved 19. Therefore, the effectiveness of the online task assignment is dependent on the arrival order of the tasks and workers.

To address the online task assignment problem efficiently, many solutions are proposed like Least Location Entropy Priority (LLEP) [61, 109], Individualized Models for Intelligent Routing of Tasks (IMIRT) [47], Two-phase based Global Online Allocation (TGOA) algorithm, Global Online Micro-task Allocation (GOMA) [113], Online Fixed/Dynamic-based Maximum Task Coverage Algorithm [115], Hierarchically Separated Tree based randomized online algorithms (HST-Greedy) [112], and Prediction-oriented Online Task Assignment in Real-time Spatial Data (POLAR) algorithm [114].

To tackle the randomness of the online task assignment problem, the historical information regarding the workers' movement behaviour, tasks' location information, and workers' task acceptance behaviour is utilized. The Least Location Entropy Priority strategy [61, 109, 115] improves on the greedy strategy of finding locally optimal solutions at each time instance, by exploiting the workers' historical information shared to the SC-server. Location Entropy [23] estimates the diversity of unique visitors to a location, in this case, location is assumed to be a grid cell. Higher location entropy is synonymous with more workers visiting the location with an even distribution of visits among the workers. Therefore, tasks lying in areas with lower location entropy have lesser chances to be completed as fewer workers visit those locations. Higher priority is given to the tasks present in smaller location entropy areas to maximize the assigned tasks, by reducing it to a *minimum-*

*cost MWBM* problem, i.e., finding the MWBM of minimal total cost/entropy. The minimum-cost MWBM problem is solved in two steps: finding the maximum matching using the Hungarian algorithm and minimizing the cost of the matching by applying integer programming technique like the branch and bound. Consider the example in Fig. A.6c, it represents the output of the maximal matching of the MWBM problem. In the minimum-cost MWBM problem, the cost of the edges is set as the entropy of the workers at the locations of the tasks. According to [109], there will be a 35 % improvement in the performance of LLEP when compared to the greedy approach. Therefore, the LLEP approach would result in a total reward of 17.

Furthermore, the tasks assigned by the SC-server could be rejected or accepted by the worker, adding to the complexity of online task assignment problem. Therefore, a task assignment will be considered as successful only if the worker accepts to perform it. In [47], the authors propose an IMIRT (Individualized Models for Intelligent Routing of Tasks) framework for online task assignment, which focuses on modelling the task acceptance behaviour of the workers from the past data. Subsequently, the task acceptance behavior models are utilized for optimizing the number of successful assignments. This framework assumes that the workers are known to the SC-server, and only the tasks appear dynamically.

The framework profiles the task acceptance behaviour of the workers and utilizes that information to improve/maximize the assignment success rate, i.e., the ratio of the number of tasks workers accept to perform against the total number of assigned tasks by the SC-server. The SC-server checks for the workers and would assign the task to the ones with a higher probability of accepting the task. The online assignment problem is formalized as a multi-armed bandit [91] model, where each worker  $W_j$  is considered as an arm, assignment  $a_{i,j}$  of the task  $t_i$  is equivalent to playing an arm and the resulting reward  $y_{i,j}$  is the probability of success. For a newly arriving task, selecting a worker with the highest probability of success  $y_{i,j}$  would result in maximization of the assignment success rate. The type of task acceptance behavior model has a huge impact on the online spatial task assignment. For instance, if each worker has a fixed behavior of task acceptance, i.e.,  $p_{i,j} = p_{i',j}$  for any tasks  $i$  and  $i'$ , it could be modeled as a Binomial process with parameter  $p_j$ . To improve the efficiency of task assignment, a new strategy *SpatialUCB* is proposed by integrating task acceptance behavior model with spatial contextual information like travel distance from task  $t_i$  to worker  $W_j$  location  $d_{i,j}$  and type of task  $e_i$ .

The IMIRT framework [47] only takes into account the dynamic nature of tasks, and assumes the workers to be static. In [113], the authors proposed algorithms based on the online model where both tasks and workers can appear dynamically. A Two-phase-based Global Online Allocation (TGOA) algorithm is proposed that divides the set of workers and tasks into two equal

#### 4. Task Matching Problem

groups based on their arrival orders. The input set of workers and tasks is estimated through the historical records for a given time interval, thereby eliminating the uncertainty of arrival order. The greedy strategy would be applied to the first half of the input set, where an arriving task or worker would be paired with its complementary that has highest utility value. For the second half of the data set, an optimal strategy is adopted to find the optimal global match to the arriving worker/task. This approach provides competitive ratio guarantee of  $1/4$  under the online random order model. Considering the previous example in Section 4.2, let us consider the order of arrival of tasks and workers as  $\langle W_1, t_1, W_2, t_4, t_2, W_3, t_3, t_5 \rangle$ . THE TGOA algorithm splits the input set into 2 sets of 4 according to their arrival orders, i.e.  $\langle W_1, t_1, W_2, t_4 \rangle$  and  $\langle t_2, W_3, t_3, t_5 \rangle$ . On the first half of the vertices, TGOA performs greedy assignment, therefore  $W_1$  is assigned to  $t_1$  and  $W_2$  is assigned to  $t_4$ . For the second half of the vertices, TGOA performs a global optimal match, therefore,  $W_3$  is assigned to  $t_5$ . The total reward achieved through this method is 15.

The above approaches assume that the worker would either be assigned to a task immediately after being available to the SC-server or waits for a task at the same reported location until the specified deadline. This assumption is impractical as the worker would, most likely, not stay idle at the same location waiting for a task to be assigned. The worker tends to roam around in the hope of finding a task to perform. Moreover, if the worker waits for the task at the same location, then she might miss out on many potentially matching tasks that appear in a different neighborhood.

Instead of making the worker wait, it would be beneficial to guide the worker to a neighborhood where new potential matching tasks might appear. [114] proposes a two-step framework, the *Flexible Two-sided Online Task Assignment* (FTOA) model. The framework allows workers to either stay at the same location waiting for a task or to be guided to a different location if the worker is not assigned a task on being available to SC-server. This approach utilizes the historical data for predicting the number of tasks and workers in a specific spatiotemporal range. The spatial and temporal dimensions are partitioned into *grids* and *time slots*, respectively. Subsequently, an offline guide is created by performing offline matching of the predicted tasks and workers according to the spatiotemporal divisions (grids and time slots) and deadline constraints.

The offline guide contains the potential task-worker assigned pairs with individual nodes representing tasks/workers in a particular grid at a specific slot of time (the grid-time slot pair is referred to as object type). This offline guide would be used in *Prediction-based Online task Assignment in Real-time Spatial Data* (POLAR) algorithm. According to the algorithm, a newly arrived task or worker would be matched with the existing nodes of the same type (same time slot and grid) in the offline guide. The matched node would

be occupied by the newly arrived task/worker (object). Each node of the offline guide can only be occupied by one task/worker of the same type. If there exists a real worker/task corresponding to the occupied node in the offline guide, the nodes are assigned to each other in the online model. If no actual tasks are corresponding to the occupied nodes in the offline guide, then the algorithm would suggest the worker to move to a grid where the potential matching task would appear at a future time. However, there might be some unpredicted tasks/workers appearing to the SC-server. The POLAR algorithm ignores such unpredicted tasks/workers as they cannot be matched with the nodes of the offline guide. An optimized POLAR algorithm (POLAR-OP) is proposed to handle the unpredicted tasks/workers. The POLAR-OP allows the nodes of the offline guide to be associated with more than one task/worker, i.e., each node of the offline guide can be reused by multiple objects.

The competitive ratios of POLAR and POLAR-OP are 0.4 and 0.47, respectively, under the independently and identically distribution (i.i.d) model [36]. Both POLAR and POLAR-OP, processes each newly arrived task/worker by looking up the offline guide in  $\mathcal{O}(1)$  time. However, the success of this approach is dependent on the accuracy of the prediction model used in the initial stage. Furthermore, this approach of moving the workers in advance is not beneficial in the cases where there are more workers than the tasks. In such cases, simple greedy approaches outperform the POLAR algorithm.

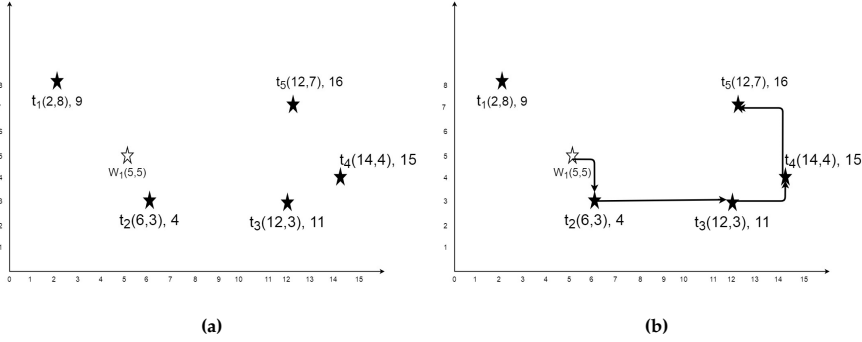
## 5 Task Scheduling Problem

### 5.1 Introduction

In the previous Section 4, task matching problems focus on assigning tasks to workers. However, to complete a task, the worker has to travel to the physical location of the assigned task. Given a set of assigned/selected tasks, travelling to every task before their respective deadlines might not be possible as there might exist tasks with overlapping deadlines, long travel times between task locations, etc. Therefore, a plan is needed to complete as many assigned/selected tasks as possible, to maximize the reward collected, leading to the optimization problem of creating a schedule/task sequence that maximizes the number of tasks performed by the worker. The spatial tasks sequence is generated taking into consideration both the travel cost and expiration time of the tasks. This optimization problem is referred as *Maximum Task Scheduling* (MTS) problem [29].



## 5. Task Scheduling Problem



**Fig. A.7:** Maximum Task Scheduling Problem in WST scenario: a. Worker  $W_1$  with the tasks  $\{t_1, t_2, t_3, t_4, t_5\}$  along with their locations and expiration time, and b. Maximum Valid Task Sequence of the MTS problem

### 5.2 Offline Input Model: One worker- Multiple Tasks Scenario

The maximum task scheduling problem [29] addresses the offline variant of task scheduling problems, where all the input parameters are known to the SC-server beforehand. The SC-server knows the worker and the set of tasks assigned/self-selected by her. To elaborate, let us consider the SC task scheduling scenario with a set of four tasks  $\{t_1, t_2, t_3, t_4\}$  selected by the worker  $W_1$  (see Fig. A.7a). Each task has an expiration time  $d_i$  and the worker needs to travel to that task location before the expiration time. In this example, let us assume that the cost of travelling between two adjacent grid cells is one time unit. The worker initial location is (5,5) at time 0. The task  $t_2$  is located at (6,3) that is scheduled to expire after 4 time units. The travel cost between the two tasks' locations is 9.

The worker has to choose the relevant subsets or the sequence of tasks to maximize the number of accomplished tasks considering the travel costs as well as deadlines of the task sequence. A sequence of tasks where all of its tasks can be completed is termed as *valid task sequence* and the one corresponding to the maximum number of tasks is termed as *Maximum Valid Task Sequence*. *Maximum Task Scheduling* (MTS) problem is to find this *Maximum Valid Task Sequence*. MTS problem varies from the general job scheduling problems [82] with respect to the time required for performing the task. In job scheduling problems, the time required for each job is known in advance, whereas in the MTS problem the time required is dependent upon the travel time between the tasks, which in turn depends on the orders in which tasks are performed.

For example in Fig. A.7b, if the worker starts with the task  $t_1$ , then the travel cost from the worker's current location to the task  $t_1$  is 6 time units and the deadline to perform  $t_1$  is 9 time units. As the worker reaches  $t_1$  at

time 6 i.e., before the deadline of  $t_1$ , therefore the task  $t_1$  can be performed by the worker  $W_1$ . At time 6, the worker cannot perform the task  $t_2$  since the deadline for it (4 time units) has passed. Similarly, the tasks  $t_3$ ,  $t_4$  and  $t_5$  cannot be performed, as the travel cost exceeds the deadline of those tasks, i.e., travel cost + current time for all these tasks exceed the maximum deadline period of the tasks, i.e., deadline of  $t_5$  (16). Similarly, if the worker starts from the task  $t_2$ , then she could be able to complete the tasks  $t_3$ ,  $t_4$  and  $t_5$  as well (see Fig. A.7b). Therefore, choosing the sequence of performing the tasks plays a major role in determining the number of tasks performed by the worker. A longer sequence of the tasks subset would result in higher number of tasks performed.

*Maximum Task Scheduling* (MTS) problem is proved as NP-hard in [29] by reducing it to a specialized version of Traveling Salesman Problem. For a small set of tasks, the MTS problem can be solved by finding the *Maximum Valid Task Sequence* using the brute-force approach. In this method, they check for all possible combinations of the valid task sequences and check for the sequence with the maximum number of completed tasks. However, this method is computationally expensive as computing all task sequences for  $n$  tasks will be  $O(n!)$ . Two strategies (exact algorithms) were proposed based on dynamic programming and branch-and-bound strategy, to address this issue.

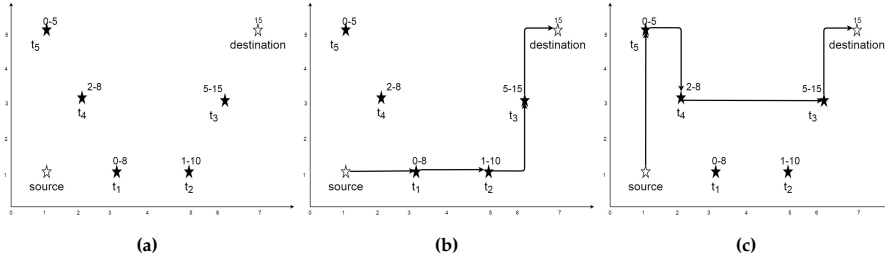
**Exact Algorithms:** The dynamic programming approach investigates the sets of tasks ignoring the order of task sequence. The search space is generated by expanding the sets of tasks iteratively in ascending order of set size from 2 to  $n$ . Therefore,  $2^n - 1$  subsets of tasks need to be investigated to find the *Maximum Valid Task Sequence*. This approach is optimized further by removing the invalid subsets of tasks from examination. An invalid set is defined as a task set without any valid task sequence in its combinations. In the branch-and-bound approach, the search space is represented as a tree, where depth-first search along with pruning unpromising branches is conducted recursively, till a feasible solution with *Maximum Valid Task Sequence* is found. The branch-and-bound approach is more efficient than the dynamic programming approach regarding space requirements.

In the dynamic programming approach, there are at most  $O(n \cdot 2^n)$  sub-problems, and each one can be solve in linear time. Therefore, the time complexity of dynamic programming is  $O(n^2 \cdot 2^n)$ , which is faster than the brute-force approach's  $O(n!)$  running time. On the other hand, the time complexity of the branch-and-bound approach is proportional to the search tree size. If the branches are pruned, and all the unnecessary nodes are removed then the time complexity of the branch-and-bound would be better than dynamic programming. It should be noted that if all the branches of the tree need to be searched without pruning branches, then the worst case time complexity of branch-and-bound would be  $O(n!)$ . However, in practice

## 5. Task Scheduling Problem

it is a rarity for a tree to be searched without pruning the branches, therefore branch-and-bound approach would be better than the brute-force approach.

**Approximation Algorithms:** Both the branch-and-bound and dynamic programming approaches result in an exponential time complexity, which is not suitable for real-world applications. Therefore, approximation algorithms are proposed in [29] based on different heuristics like *Least Expiration Time Heuristic*, *Nearest Neighbor Heuristic* and *Most Promising Heuristic*. With the Least Expiration Time Heuristic, task sequences are formed greedily by selecting tasks with least expiration time. In the case of Nearest Neighbor Heuristic, the nearest available task to the last task in the sequence is added. Finally, the Most Promising Heuristic helps in choosing the most promising branch in each iteration instead of searching the entire tree. The approximation algorithms output with less accuracy compared with exact algorithms but quicker response. It was noticed that in real-world applications, it is sufficient to report a small number of tasks assigned to a worker, while the remaining solution is computed offline by the server. This leads to the proposal of progressive algorithms [29], where approximation algorithms are used to find out a small number of tasks for a given worker. The optimal task sequence for the remaining tasks is found out by the exact algorithms like branch-and-bound. Progressive algorithms improve the response time when compared to exact algorithms and accuracy when compared to approximation algorithms. However, there are chances that worker's potential tasks may be arrogated by other workers, when the worker is on the way to perform the initial tasks. Moreover, workers may prefer to see the entire task sequence before starting work.



**Fig. A.8:** Online Route Recommendation Problem example with source, destination and arrival time at destination=15 in WST scenario: a. Worker  $W_1$  with the tasks  $\{t_1, t_2, t_3, t_4, t_5\}$  along with their locations, release time and expiration time. b. Result Route of the OnlineRR problem with Nearest Neighbor Heuristic. c. Result Route of the OnlineRR problem with Earliest Deadline Heuristic

### 5.3 Online Input Model: One Worker- Multiple Tasks Scenario

One of the limitations of the MTS problem is that it calculates the maximum valid task sequence for a particular snapshot of time. The outcome task sequence does not update according to the arrival of new tasks that lie in the spatial region of the worker, which are requested to the SC system by the requesters. Furthermore, according to [2] workers often take tasks that are present in their route, for example, daily commute from home to work, resulting in extra constraints like destination and arrival time at the destination. Therefore, considering both these requirements would further improve the outcome of the task scheduling problem. Taking both these requirements into consideration, [73] introduced an *Online Route Recommendation Problem* (OnlineRR) for workers who choose their tasks.

For example, consider Fig. A.8a which shows a set of tasks  $\{t_1, t_2, t_3, t_4, t_5\}$  along with their locations, release times, and deadlines. The figure also shows the source and destination points of the worker  $W_1$  and the time taken to arrive at the destination (15). The OnlineRR finds the longest sequence of tasks which can be performed considering that the worker should be present at the destination at time 15. To solve this problem, [73] presents two approaches: greedy approach and re-routing through a complete search at a time snapshot.

The greedy approach was chosen with the aim of reducing response times since the global view of the problem is not possible due to the continuous task and workers' location updates. In the greedy approach, the next task is chosen according to heuristics, such as the nearest task to the worker's current location, tasks with earliest deadlines and, maximizing the search space of feasible tasks. Furthermore, for choosing the next task it should satisfy the condition of reaching the task's location before the task's deadline.

For instance, consider Fig. A.8b, it depicts the result route outcome of the greedy approach with the nearest neighbor heuristic. The worker at time 0 is at the source location, and she searches for the closest task in her vicinity.  $t_1$  is selected as it is the nearest with 2 units distance and takes 2 time units to travel to task's location. Now the current location of the worker (at time 2) is updated to the  $t_1$  location. Worker again begins her search for the nearest neighbor whose travel distance is less than the deadline minus the current time, in this case, it's  $t_2$  that is 2 units away from  $t_1$ . The current location will be updated to  $t_2$  at time 4. Similarly,  $t_3$  will be selected and the current location will be updated to it at time 7 and the route to (source,  $t_1$ ,  $t_2$ ,  $t_3$ ). Now at time 7, there are no feasible tasks available to perform as the worker cannot reach the tasks' location the deadlines. Therefore, the worker heads to the destination which is 3 time units away. Finally at time 10, the destination of the worker will be reached and the result route will be (source,  $t_1$ ,  $t_2$ ,  $t_3$ ,

## 5. Task Scheduling Problem

destination).

Similarly, in Fig. A.8c, the result route of the greedy approach with earliest deadline heuristic is depicted. The difference from the nearest neighbor is that the preference or high score will be given to tasks that have earlier deadlines than the remaining available deadlines. For example, at time 0, the earliest deadline for a task is  $t_5$  with the deadline at 5 time units, consequently  $t_5$  is selected. In similar fashion, maximum candidate space heuristic is used to calculate the score of tasks based on the distance.

The second approach to solve the OnlineRR problem is the re-routing method, where the optimal solution is calculated at the current snapshots recursively as long as there exists a new feasible task  $p$ . For instance, consider the example in Fig. A.8a, at time 0, worker has two feasible tasks ( $t_1, t_5$ ) and re-route algorithm computes the task sequence  $R_0$  (source,  $t_1, t_5$ , destination). According to the task sequence, the worker moves to  $t_1$  at time 2 and new feasible tasks are found  $\{t_2, t_4\}$ . Subsequently, the task sequence is re-computed to  $R_1$  ( $t_1, t_4$ , destination). Following the re-computed task sequence, the worker moves to  $t_4$  at time 5 and found one new feasible task  $t_3$  and the new route  $R_2$  ( $t_4, t_3$ , destination). Following the new route, the worker moves to  $t_3$  at time 9, where there are no more feasible tasks. Therefore, the worker has travelled the following route (source,  $t_1, t_5, t_4, t_3$ , destination). It covers more tasks than the other heuristics mentioned earlier.

The previous scheduling problem deals with a single worker along with multiple point tasks. The problem gets complicated while scheduling for delivery tasks that have both the source and the destination. The problem of scheduling delivery tasks for a single worker, The *Online Delivery Route Recommendation Problem* is discussed in [101]. The optimization problem objective is to maximize the worker's income. Following a prediction-based approach, an algorithm is proposed considering three factors, namely; the distance between worker's location and origin of the feasible delivery task, the distance between origin and destination of the feasible delivery task, and the future demand originating from the destination of each delivery task. This algorithm has a shortcoming if the worker has a personal destination that she should reach before a deadline. With the prediction-based approach, it is possible to be far away from the destination and having to spend much time to return to the destination, consequently losing on the potential tasks and rewards. To address this shortcoming, the prediction-based algorithm has been extended [101], to consider the impact of distance between delivery task's destination and worker's destination. The current approach can plan for a group of delivery tasks for a single worker. New methods needs to be proposed for tackling the scenario of multiple workers with multiple delivery tasks.

## 5.4 Combination with Task Matching Problems: Multiple Workers - Multiple Tasks Scenario

Table A.4 summarizes the different task matching and task scheduling problems in the SC literature. Task scheduling is necessary to assist the worker to complete as many assigned tasks as possible. However, there are some drawbacks of the above-discussed task scheduling problems. They do not address the issue of re-matching the tasks that cannot be performed by the worker with other available workers and subsequent re-scheduling for the re-matched tasks. Consequently, inefficient scheduling results are produced in multiple workers scenario. One solution to address these shortcomings is to combine task matching and scheduling problems and carry them out iteratively. This combination approach would improve the task acceptance and completion rates due to the enhanced awareness of SC-server. The SC-server can determine whether a matched task can be completed by the worker through task scheduling algorithms.

As observed in Table A.4, there are examples of SC optimization problems that combine both task matching and task scheduling. Furthermore, the previously discussed task scheduling problems focus on a single worker and multiple tasks scenario. In [30], a combination of task matching and scheduling problem is proposed to minimize the travel cost of multiple workers and maximize the number of completed tasks. The authors propose two solutions to address this problem: *Global Assignment and Local Scheduling* (GALS) and *Local Assignment and Local Scheduling* (LALS). GALS approach performs task matching and scheduling sequentially and utilizes the output of task scheduling to refine the task assignments. To elaborate, GALS performs an initial task assignment resulting in a set of tasks assigned to each worker, subsequently assigned tasks are scheduled for each worker and finally, the task assignment is refined with the unscheduled tasks from the previous step and available workers. GALS approach performs the scheduling and updating task assignment steps iteratively until the termination condition is reached. The iteration condition is satisfied when there are no potential worker-task pairs available in the input set for updating task assignments. The initial global task assignment and assignment update phase are more computationally expensive than the local scheduling phase, due to the size of search space for performing the assignments.

Bisection-based LALS approach is proposed to improve the scalability of GALS approach. Initially, this approach recursively divides the input dataset of workers and tasks in a top-down approach, until the size of the partition is less than a predefined threshold. Then, the resulting partitions are merged in a bottom-up approach according to a predefined threshold. If the combined workload of the sibling partitions is less than the threshold, then the sibling partitions are merged to create a new partition. However, if the combined

## 5. Task Scheduling Problem

workload of the two sibling partitions is more than the threshold, then a local assignment and scheduling is performed on the individual sibling partitions. The remaining workers and tasks from the individual partitions are then merged to create a new partition. The merging is performed iteratively until the root node is reached in the partition tree. The intuition behind merging the remaining workers and tasks of the sibling partitions is to utilize the spatial properties of the binary tree for improving the quality of assignments and scheduling.

The aforementioned paper [30] focuses on maximizing the number of completed tasks while minimizing the travel cost for the workers benefiting both the SC-server and workers. In [95], a Utility-aware Social Event-participant Planning (USEP) problem was proposed to focus on benefiting the workers by maximizing the total utility score for the event attendance schedules while satisfying the worker's travel budget constraints. To tackle this problem, [95] proposes a greedy-based heuristic algorithm and dynamic programming based approximation algorithms. Similarly, in [12], a coordinated task assignment approach, Trajectory-Aware Coordinated Urban CrowdSourcing (TRACCS), is proposed to assign a sequence of tasks to worker considering their movement patterns.

In the above approaches, task assignment and subsequent scheduling is performed in batches. However, in real-time online assignments, it would be necessary to assign and schedule tasks as soon as they enter the system. Consequently, the SC-server is performing schedule operations for multiple workers for every new task, which is difficult to support and thus results in scalability issues [5]. To tackle this problem, [5] proposes a novel auction-based framework Auction-SC, where the workload is split between the workers and the SC-server. Workers bid on the newly arriving tasks, and the highest bidder would get the task assigned to them based on the auction framework proposed in [4]. The workers would submit their schedules to the SC-server to assess on the feasibility of the new task to their valid schedule and for computing their optimal bids. Hence, this approach tries to benefit both the worker by maximizing her profits and the SC-server by maximizing the number of completed tasks. This approach does not consider the time required for performing a task, and does not support complex tasks. Moreover, the case where the worker fails to adhere to the task sequence is not considered. Similarly, [136] proposes a solution to the offline destination aware task assignment problem, that utilizes task dependencies among workers, for dividing them into independent clusters and deadline constraints of workers' destination and tasks' expiration time. This approach would result in an optimal assignment without the need of re-assignment and re-scheduling. However, this approach does not consider the time required for performing a task which would have a more significant impact during scheduling. Furthermore, it does not consider the dynamic arrival of tasks and workers.

**Table A.4:** Summary of the Surveyed Task Assignment and Scheduling Problems: The acronyms used according to category are: **Constraints:** SpC -Spatial, TC -Temporal, BC -Budget, QC -Quality, **Dynamic nature of Inputs:** TA, WA -Task and Worker Arrival. **Heuristics:** Gr-Greedy, TC-Travel Cost, LE-Location Entropy, NN-Nearest Neighbor. **Privacy:** PP- Privacy Protection

Problem Type	Problem Name	Support for Constraints				Dynamic Arrival		Reduced Problem	Algorithm Heuristic	Limitations
		SpC	TC	QC	BC	PP	TA	WA		
Task Matching	MTA [61]	X	X					Max-Flow	Gr	<p>a. Does not work well in a dynamic setting, esp. greedy strategy.</p> <p>b. Workers are assumed to accept tasks that are assigned to them.</p> <p>c. Trusts the worker to perform the task correctly.</p> <p>d. Does not consider the worker movement patterns</p> <p>a. Worker is assumed to be constantly moving in a specific direction with a certain velocity.</p> <p>b. No Option to limit the distance a W can travel, resulting in assignment of far away tasks.</p> <p>a. Does not take into account the movement of workers and the dynamic arrival of tasks</p> <p>a. K-anonymity based approach has limitations like it is prone to homogeneity attack, or the privacy relies on the value of k.</p> <p>a. Does not take into account the movement of workers</p> <p>b. Produces sub-optimal assignment due to fake workers in the geocast region.</p> <p>a. Assumes the cost of performing a task proportional to travel cost, and Euclidean distance function is used to calculate travel cost.</p> <p>a. Does not account for the dynamic nature of workers</p> <p>b. Only one worker can be assigned to the task</p> <p>a. Limited to hyper local tasks that do not require the workers to travel.</p> <p>b. Worker cannot try to maximize his reward by performing sequence of tasks.</p> <p>c. The performance is hindered due to the dynamic arrival of workers, as the budget is allocated per time periods</p> <p>d. Offline algorithms, though not practical, helps in tackling the randomness of the real-world online problem.</p> <p>a. Dependent on the accuracy of the initial prediction model</p> <p>b. Not beneficial when the workers are more in number than the tasks.</p> <p>a. Focuses on micro-tasks that are simple and trivial.</p> <p>b. Focuses on maximizing the benefit of SC-server and not the worker.</p>
								MinCost-MaxFlow	LE	
								MWBM	Gr	
								MinCost-MWBM	LE	
	MTMCA [27]	X	X	X				MinCost-MaxFlow	TC	
	RDB-SC [17]	X	X	X				Number Partition		
	MIRA [135]		X	X				SCP [6]	Gr	
	PAPA [60]	X				X		Set Cover	Gr	
	DPTA [106]	X				X			TC	
	M5-SC [15]	X	X	X				Set Cover	Gr	
Task Matching	OSTA [47]					X		MAB	Spatial Context	<p>a. Focuses on maximizing the benefit of SC-server and not the worker.</p>
								MCG	Gr-Offline	
									Gr	
									Temporal	
	Fixed Budget MTC [104, 115]	X	X		X		X	MCP	LE	
	Dynamic Budget MTC [104, 115]	X	X		X		X	MCP	Gr-Offline	
									Gr	
								MAB	Temporal	
									LE	
	FTOA [14]	X	X				X	Offline Guide: Max-Flow	TC	
	GOMA [113]	X	X	X	X		X	OMWBM	Gr	



## 5. Task Scheduling Problem

Table A.4: (Continued)

Problem Type	Problem Name	Support for Constraints				pp	Dynamic Arrival		Reduced Problem	Algorithm Heuristic	Limitations
		SpC	TC	QC	BC		TA	WA			
Task Scheduling	MTS [29]	X	X		X				TSP	Temporal NN MPH	a. Not applicable in the case of dynamic arrival of tasks b. The set of tasks is preselected for the worker.
	OnlineRR [73]	X	X				X		OPTW	Temporal NN MCS	a. Set of tasks arrive dynamically, however schedules plan for only one worker. b. Does not take into account some constraints like quality
	MTSMW [30]	X	X						Matching: MTA Scheduling: Routing	Same as MTA TC	a. Assignment module has similar limitations as the MTA problem b. Assumes new task insertion does not effect task order.
	TRACCS [12]	X	X						Orienteering	Gr ILS	a. Partial order is maintained among the nodes visited. b. Preference to individual benefits than globally optimal solution.
	OnlineTASC [5]	X	X		X		X	X	Min-Len-Ham	Random Ranking NN Most Free Time Best Insertion Best Dist.	a. Does not consider the required time for completion of task b. Does not support complex tasks
Task Matching & Scheduling	DSTA [136]	X	X							Gr	a. Dynamic arrival of tasks and workers is not considered b. Does not consider the required time for completion of task

## 6 Spatial Constraints

### 6.1 Overview

Spatial Constraints refer to the different spatial preferences of the worker or the task. The SC-server assigns or publishes tasks to the workers, while satisfying specified constraints. In the case of Worker Selected Tasks (WST) publishing mode, it is hard to enforce the spatial constraints without the knowledge of worker's location. Some of the applications like Gigwalk (<http://www.gigwalk.com/>), TaskRabbit (<https://www.taskrabbit.com/>) enquire the workers location or pin code to serve the tasks in their proximity. After the selection of tasks in the WST mode, spatial constraints can be enforced while scheduling a plan for completing the tasks. For example, in [73] there are spatial constraints regarding route planning starting from the source and reaching the destination at a particular time. The SC-server can include as many tasks as possible in the plan, provided these conditions were met.

On the contrary, in the case of Server Assigned Tasks (SAT) publishing mode, spatial constraints can be strictly enforced. Some of the types of spatial constraints are the spatial region, maximum travel distance, and direction of worker's commute. The spatial constraints are usually dependent on the current location of the worker, for example, preferring tasks that are within 500 metres from the current location of the worker. In the case of the offline scenario, the worker's location remains static or known to the SC-server, thus the spatial constraints are not expensive to construct. However, in the case of online scenario, the worker's location is received by the SC-server in a streaming fashion, therefore a change in worker's location triggers a re-construction of spatial constraints of the worker.

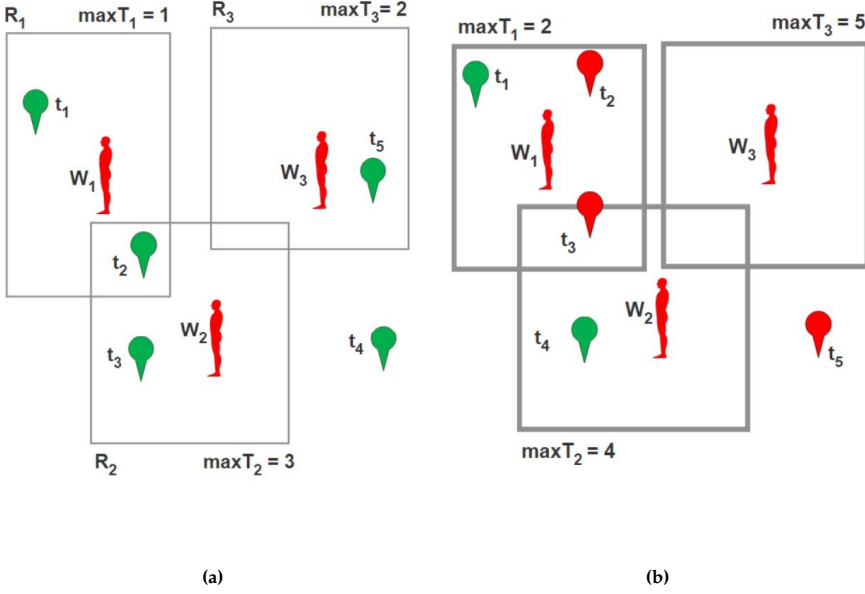
### 6.2 Spatial Region

A worker can define his preferred spatial region as a minimum bounding rectangle [61, 62, 109] or as a bounding circle with her location [113]. It is assumed that the worker is willing to travel to any of the tasks that lie within the defined spatial ranges. Similarly, requester can specify spatial constraints on the task, like in the case of region tasks [104, 115] where the worker could be located anywhere in a given circular range.

**Example.** To understand the effect of spatial constraints on a task assignment, let us consider the basic assignment problem of Section 4.2 and enforce the spatial region constraint of the workers. Fig. A.9a showcases an example scenario representing the spatial tasks and the workers along with their preferred spatial regions and the maximum number of tasks that the worker can perform. As discussed in previous sections regarding the task assignments, there exists a many-to-many relationship between tasks and workers, for in-

## 6. Spatial Constraints

stance, a task can be assigned to multiple workers and vice versa. However, the relationship is subject to satisfying the conditions: the task should lie in the preferred spatial region of the worker, and the threshold of the maximum number of tasks performed by worker is not violated.



**Fig. A.9:** a.SC in SAT mode with workers' constraints: Spatial Regions  $\{R_i\}$  and the maximum number of tasks  $\{\max T_i\}$  b. An expertise matching example of workers set  $\{W_i\}$  and spatial tasks  $\{t_i\}$ : Same colour(red) between workers and spatial tasks represent an expertise match.

One possible assignment would be between the spatial task  $t_5$  and worker  $W_3$  as the task lies in the region of  $R_3$  and the  $\max T_3$  value is 2. Similarly, spatial tasks  $t_1$  and  $t_2$  lies in the region  $R_1$  of the worker  $W_1$ . However, both the tasks cannot be assigned to  $W_1$ , due to the maximum number of tasks performed constraint ( $\max T_1$  value is only 1). Therefore, either  $t_1$  or  $t_2$  will be assigned to  $W_1$ . Since, the spatial task  $t_2$  lies also in the region of  $W_2$ , who has a higher value for  $\max T$  ( $=3$ ), the task  $t_2$  is assigned to  $W_2$  to maximize the number of assignments. On the other hand, as  $t_4$  does not lie under any spatial region of the available workers, it remains unassigned. This assignment problem could be solved by the strategies proposed in [61] by finding out the locally optimal solution.

Constraints	Spatial	Quality	Budget
Spatial	MTA [61]	MSA [109], RDB-SC [17], MCTA [62], GOMA [113], MTMCA [27]	f-MTC/d-MTC [104, 115], MTMCA [27], GOMA [113]
Quality	MSA [109], RDB-SC [17], MCTA [62], GOMA [113], MTMCA [27]	MRA [135]	MS-SC [15], MTMCA [27], GOMA [113], Crowdlet [88], MQA [16]
Budget	f-MTC/d-MTC [104, 115], MTMCA [27], GOMA [113]	MS-SC [15], MTMCA [27], GOMA [113], Crowdlet [88], MQA [16]	-NA-

**Table A.5:** Comparison of task assignment protocols with multiple constraints

### 6.3 Maximum travel distance

A worker can set a maximum travelling distance she is willing to travel for performing the assigned tasks beyond her commute or movement trajectory [12]. The server should intuitively assign tasks to maximize the number of tasks performed and the overall rewards received without violating the maximum travelling distance threshold of individual workers.

### 6.4 Direction of worker's commute

The spatial constraint about the direction of worker's movement would help in assigning tasks to a worker without a significant deviation from her path [17]. For example, if the assigned task lies in the opposite direction of worker's movement in a directional road network, then the travel distance to the task would be longer due to the necessary direction change in worker's commute. Given the longer travel distance, the likelihood for a task to be rejected by the worker is high. Furthermore, assigning the tasks based on the direction of worker's commute might improve the diversity of spatial angles regarding the information queried by the task. For instance, if the task involves a taking a picture of a public monument, assigning the task to different workers who travel through the task location in different directions would result in a wide range of images taken from different angles, thus improving the diversity of spatial angles.

## 7 Quality Constraints

### 7.1 Overview

In this section, we focus on the different quality constraints that assure a certain level of quality in the task responses from the workers. In the WST publishing mode, the server can conduct some qualification tests related to the skills required for the task before allowing the worker to view the task like in the case of [92]. However, apart from this, the server cannot exert any control to assure quality. On the contrary, in the case of SAT mode, the server can exploit the workers' profiles and find reliable workers with necessary skills to deliver a better quality task responses [1]. Two aspects of a worker's profile could affect the quality of task's outcome, namely a worker's expertise and reputation/reliability. These aspects of a worker's profile would be utilized during assignment of tasks in the form of constraints. The constraints could be; a required expertise level in certain skills to perform a task, and a minimum reputation/reliability measure for a task to be assigned.

### 7.2 Worker's Expertise

Expertise indicates the capabilities/skills of the worker in performing a specific type of tasks. For example expertise in, taking a photograph, tutoring a high school subject, painting a house, etc. Expertise on a particular skill is usually measured as a degree, like how skilled a worker is in tutoring a subject. Worker can specify the skills or expertise she possesses to the SC-server through submitting documents proving their credentials or experience. Alternatively, the server has the capability to assess the skills of the worker based on the historical information of tasks performed and their respective rating from the requester. Various studies have proposed methods [15, 27, 109] in the SAT publishing mode that utilizes the expertise aspect to assign tasks to capable workers. In the works mentioned in earlier sections, the spatial task assignment has the assumption of uniform expertise among all the workers.

However, workers are more likely to have varying expertise, for example, some may be good in taking expert pictures, while others excel in describing things at a location. Similarly, the spatial tasks types may vary, for instance, taking a high-quality picture of a monument at a physical location. Therefore, if a worker has the expertise that matches the spatial task type, then the resulting quality would be better than if the skills do not match the task [34]. In [27], an expertise constraint is introduced to the task assignment problem where the expertise of the worker has to match the expertise required of the task. The worker is defined to have a particular expertise  $E_w$  with a degree  $D_w$  and the task with required degree  $D_t$  of expertise  $E_t$ . The assignment constraint for assigning a spatial expert task is defined as:  $E_w = E_t \wedge D_w \geq D_t$

The expertise  $E_w$  or  $E_t$  refers to a specific skill, therefore the major limitation of this definition is that the worker's expertise is limited to only one skill. Similarly, the spatial task cannot demand expertise from the worker in multiple skills. The optimization problem has the goal of maximizing the number of assigned spatial expert tasks and minimizing the total cost reward to workers.

However, in a real-world scenario, a worker could possess expertise in multiple skills and a task would also require multiple types of skills. [109] addresses one of the limitations of [27] regarding the issue of multiple skills of the worker. The expertise of the worker  $W$  is defined as a set of skills  $E$  and assigned required skill as a task type  $e$  attribute to the task  $t$  definition. A score is assigned to all potential worker-task pairs  $\langle W, t \rangle$ , indicating the performance of the worker  $W$  based on her expertise. An expertise match  $\langle W, t \rangle$  is made when the worker  $W$  possesses the expected skill of the task  $t$  in the preferred spatial region  $R$ , i.e.,  $e \in E$ , and a higher score is assigned to the expertise match. On the contrary, if there are only workers available without the necessary skills required for the task, then a *nonexpertise* match, provided the task satisfies the spatial constraints. A lower score is assigned to these *nonexpertise* matches. By maximizing the overall score of the assignments, the total number of expertise matches would be maximized.

To understand this scenario, let us extend the basic assignment problem by adding the spatial and expertise constraints (see Fig. A.9b). Tasks and workers with the same expertise are represented in the same color. It can be noticed that the spatial task  $t_5$  will not be assigned to any of the three workers. Furthermore, worker  $W_3$  has no spatial tasks in its spatial region. Therefore none of the tasks will be assigned to her. It can be observed that there is an expertise match between  $W_1$  with  $t_1$ ,  $t_2$  and  $t_3$ . Out of these,  $\langle W_1, t_1 \rangle$  is a non-expertise match and the remaining two matches  $\langle W_1, t_2 \rangle$  and  $\langle W_1, t_3 \rangle$  are expertise match. As observed, there are chances for an expertise match or non-expertise match. Each match will be given a score, and the aggregate score will be counted for each time instance. The problem is to maximize the aggregate score of the assignments made by the SC-server. This problem is called *Maximum Score Assignment Problem* [109], that can be reduced to the *Maximum Weighted Bipartite Matching Problem*. The method proposed in [109] does not address the case of the task requiring multiple skills for being assigned to a worker. [15] addresses this limitation by defining the skill set required for the task to be assigned.

The discussed works assume that the worker does not move dynamically in the geographic space. However, workers are more likely to move around. [16] addresses the maximum quality task assignment (MQA) problem that assigns moving workers to spatial tasks while satisfying the budget constraints of travel cost. The optimization target is to maximize the overall quality score, considering the current and future workers/tasks. An accurate

prediction approach is proposed to predict the quality location distributions of workers and tasks.

### 7.3 Worker's Reputation/Reliability

A worker's reputation/reliability refers to the probability that a task is completed correctly by the worker. It reflects the quality of the task response that can be expected from the worker. All the previously mentioned works assume that the worker can be trusted and tasks can be performed correctly. Therefore, assigning a task to a single worker would suffice in getting the expected outcome. However, in often cases, the assumption does not hold true as the workers might not be skilled enough to perform the task correctly. Moreover, some of the workers may have the malicious intent to exploit the system. Consequently, the confidence probability of the tasks and the workers should be considered, resulting in more than one worker being assigned a task with an aggregate reputation score satisfying the required confidence probability [62].

In [62], a worker  $W$  is defined along with the reputation score  $r$ , i.e., probability to perform the task correctly. The task  $t$  is defined along with the confidence threshold  $\alpha$ , which is the minimum reputation score required for an assignment. A correct match is defined when a task  $t$  is assigned to set of workers with an aggregate reputation score  $\sum r_i$  greater than or equal to the task's confidence threshold  $\alpha$  ( $\sum r_i \geq \alpha$ ). However, this approach does not focus on the dynamic nature of arrival of tasks and workers. Furthermore, the workers' movement is not considered during task assignment.

[17] tries to address the shortcomings of [62], by integrating the movement of workers with the help of the direction of movement and velocity at which the worker is travelling. The knowledge of workers' movement is used to improve the diversity of task responses that could be received from different spatial viewpoints. For example, one worker can take a picture of a monument from the west-side and another from the east-side. They address the challenge of the online scenario by designing a grid index to enable efficient updates of workers/tasks. [88] considers the online scenario of workers recruitment process. This approach takes into account the worker ability model and proposes methods to improve the overall quality of the workers recruited.

The worker's reputation/reliability is estimated during the process of truth discovery through aggregation of crowdsourced data [45, 50, 85, 90]. In the subsequent sub-section we will discuss the truth discovery process and estimation of worker's reputation/reliability.

## 8 Budget Constraints

### 8.1 Introduction

Spatial tasks involve an incentive for the worker who performs the task. The incentives could be intrinsic or extrinsic in nature [1]. The intrinsic incentives could be based on personal interest, altruism of the workers, and the extrinsic incentives involving monetary rewards [27, 48, 67], virtual credits [14, 66]. Associating financial rewards to tasks would encourage more workers to take up the task, and accelerates the completion of task [81]. With the increased worker participation, the overall quality of the outcome might improve, although not with guarantees. With monetary rewards involved, spatial tasks attract fraud workers trying to maximize their benefits by providing false information. In contrast, when a spatial task needs to be performed without any extrinsic rewards, then the chance of the task being carried out by a committed worker could be higher.

In often cases in the SC literature, the spatial tasks does not involve any monetary incentives. Such voluntary tasks may have less chance of attracting workers [2], compared with tasks with incentives. Especially for spatial tasks requiring workers with a particular skill set to perform the task [27, 109]. Usually, workers with the required expertise expect an associated reward for performing the task. These kinds of tasks are termed as *Spatial Expert Tasks*. Merely associating extrinsic incentives to the task would not be sufficient as there is a need to control the incentives for workers with respect to the quality of information provided and their associated costs, such as travel cost, smartphone battery consumption cost, data transmission cost, and manual efforts involved [64]. Moreover, there could be additional constraints involved such as budget constraints from the requester end and reliability requirements [35, 38, 39]. Therefore, appropriate incentive mechanisms are required for SC systems. The relationships between the spatial, quality and budget constraints are illustrated in Table A.5.

### 8.2 Reward Models and Incentive Mechanisms

The reward models are defined based on the two types of incentive mechanisms: *platform-centric incentive mechanism* and *user-centric incentive mechanism* [128]. In the *platform-centric incentive mechanism*, the SC-server exercises control over the allocation of rewards to the workers. In the *user-centric incentive mechanism*, the workers exercise control over the payment, by denoting the price for which they are willing to perform the tasks. Based on the incentive mechanisms, the reward models are classified as :

- *Platform-centric:*



## 8. Budget Constraints

- **Fixed reward for the task:** The rewards for the spatial tasks are fixed, and the worker, who performs the task, receives the same amount irrespective of her costs involved, and she does not have any say in it [25].
  - **Dynamic reward for the task:** The rewards for the spatial tasks are dynamic, and the worker who performs the task receives the amount depending on the quality of service provided and the costs involved [78].
  - **Fixed Budget for Time Periods:** The SC-server assigns a fixed budget for each time period, where the maximum number of workers should be selected in that time period with the budget [104, 115].
  - **Dynamic Budget for Time Periods:** The SC-server assigns a fixed budget for the entire campaign, where the workers are selected wisely to allocate the total budget to the different time periods of the campaign [104, 115].
- **User-centric:**
    - **Reward expected by the worker** The worker has the option to set the amount she would like to receive for performing tasks. This value could be dynamic, and that could be updated with the task enquiries sent by the worker to the SC-server [27, 126].

However, the support for the reward models mentioned above is dependent on the type of task publishing mode chosen. For instance, with the WST task publishing mode, *user-centric* reward models are not viable, as the SC-server lacks knowledge regarding the worker requirements. Although, workers can apply a filter to select tasks with a minimum reward from the set of available tasks. Moreover, the WST mode is not efficient for *platform-centric* reward models, as there is little control on screening the workers, apart from conducting qualification tests and post-processing the outcome delivered by the workers and assessing the quality to reward appropriately [2, 14, 92].

On the other hand, the SAT task publishing mode supports both the *platform-centric* and *user-centric* reward models. As the server chooses the worker that can work on the spatial tasks, the incentives could be managed according to the different constraints of the task and the worker. The truthfulness of the worker's costs could be analyzed, and the payments could be calculated [35], especially in the cases of complex tasks where a budget is provided for the entire complex task, that should be divided among the sub-tasks. The general assumption is that the worker would try to maximize her benefit by competing with her peers. Therefore in order to be truthful, the workers should specify their cost independent of the other workers costs [128].

The reward models in the SC literature are primarily of *platform-centric* nature, corresponding to the SAT task publishing mode. Consequently, the SC applications tend to focus on benefitting the SC-server. However, the optimization goals could be altered to benefit the SC-server or the workers. For example, the optimization problem proposed in [27] minimizes the rewards expended by the SC-server, thereby benefitting the SC-server. On the contrary, the optimization problem proposed in [113], focuses on benefitting the worker by maximizing the rewards received by the worker. Although, it assumes that the SC will benefit from the fees it earns on successfully assigned tasks. [48] proposes a pricing mechanism based on bargaining theory, to help both the SC-server and the workers to determine the rewards associated with the tasks. The reward of the tasks are dependent on the cost incurred by the workers to perform the task and the number of available workers. A higher number of available workers would result in lower rewards for tasks. Similarly, [19] utilizes the geo-social relationships to develop a diversity-driven and socially aware reward mechanism to improve the value of information by improving diversity among the recruited workers. [129] utilizes the fog platform [77] to identify the most valuable workers by harnessing their historical records. A budget constrained worker selection model is proposed that focus on learning about the workers skill and effort. Table A.5 showcases the relationship between the different constraints.

## 9 Privacy Protection

### 9.1 Introduction

In most cases, the worker is required to disclose her location information to the third-party SC-server for task assignment or response verification regarding the visit to the task location. However, this shared location information is highly sensitive and susceptible to privacy attacks from adversaries like a potentially untrustworthy SC-server, leading to privacy concerns from workers. Similarly, a task location published online or shared by many workers would result in a privacy breach. With the knowledge of workers' location information, the adversaries can infer the sensitive data of workers like their health information based on the visits to specific hospitals, religious preferences based on the visits to temple/mosque, lifestyle preferences based on their visits to different leisure places. Even in some cases, where the worker uses a fake identity, the location information could still be used to identify the movement patterns of the worker revealing details like the worker's home and workplaces [52], which in turn can be used to reveal her identity.

The privacy concerns of the workers should be addressed by ensuring the location privacy is not violated by the SC system. Otherwise, the workers

may refuse to participate in the SC process resulting in failure of SC. Therefore, workers' and tasks' location information should be protected while publishing tasks in different modes.

In the case of WST task publishing mode, the worker does not share her identity and location. However, she is still vulnerable to privacy risk during the task completion and reporting phases. For instance, the worker needs to travel to the task's location during a particular time period to perform the task. With this knowledge, a potential adversary can stalk the workers by creating fake tasks. For example, in [120], the adversaries generated fake accidents to stalk users. Pseudonymity techniques [97] or exchange-based techniques [132] are used as countermeasures to disassociate one's identity with the uploaded data or to increase uncertainty by merging location information of different workers.

Most of the works related to privacy preservation focus on the SAT publishing mode, as all the workers share their location information with the SC-server for task assignment. Cloaking [60, 87], differential privacy-based [105, 106, 133], and encryption-based [96, 97, 139] techniques are used to protect location privacy of the workers during task assignment. As the name suggests, encryption-based approaches hide the identity and location of workers through encryption. For instance, [96] proposed a secure task assignment strategy to protect privacy in the SAT task publishing mode.

### 9.2 Cloaking Techniques

With the use of cloaking techniques, the workers' locations are hidden in cloaked regions. The most used form of cloaking in the SC literature is spatial  $k$ -anonymity that generates a cloaked region for each worker containing  $k - 1$  other workers. In [60], a privacy framework named PiRi (Partial-inclusivity and Range independence) is proposed that preserves privacy during task assignment. The framework assumes that the workers trust each other and do not reveal sensitive information about each other. It considers the SC-server as an adversary and workers cannot share their locations. The worker cloaks her location with  $k - 1$  nearest peers and sends the range queries that are again cloaked among  $k - 1$  peers to avoid range dependency leaks. The range queries are formed utilizing the maximum radius among all the  $k$  peers inside the cloaked region. The framework tries to minimize the number of queries submitted to the SC-server to avoid all-inclusivity leaks. No constraints are considered in this framework.

G-STAC (Global optimization - Spatial Task Assignment with Cloaked locations) and L-STAC (Local optimization - Spatial Task Assignment with Cloaked locations) methods proposed in [87] consider the maximum travel distance of the worker (spatial constraint) while preserving privacy using spatial  $k$ -anonymity techniques. Privacy guarantee of  $k$ -anonymity based

techniques depends upon the specified  $k$  value. Choosing an appropriate  $k$  value could be difficult as the frequency of workers' visits is not considered. Consequently, the likelihood of an attack on the worker with the most number of visits to a location would be higher. Moreover,  $k$ -anonymity based techniques are prone to homogeneity attacks [79] when all the  $k$  workers are present at the same location.

### 9.3 Differential Privacy-based Techniques

Differential Privacy-based techniques refer to distortion of the original location information of the workers by addition of artificial noise. Differential Privacy (DP) [33] addresses participant concerns regarding the leakage of personal information. DP ensures that the released results would not be affected even if the participant removes her data from the input data set. Therefore, an adversary cannot guess whether a participant has participated in the database or not. It would be difficult for an adversary, even with prior knowledge, to infer the data about an individual from the DP's published sanitized data. DP is the most used strategy to protect workers' locations during task assignment [105–107, 133], that addresses the above-mentioned issues of cloaking techniques like prone to homogeneity attacks and privacy guarantee dependency on  $k$  value.

[106] proposes a DP-based privacy framework for SC that performs privacy-aware spatial task assignment. The workers do not trust the SC-server to share their location and identity information. However, they will share their locations to the trusted Cellular Service Provider (CSP). CSP collects the locations and releases a private spatial decomposition (PSD) according to the privacy budget  $\epsilon$ , that was agreed upon by the workers. The CSP can disclose the location information according to the terms agreed with the subscribed workers. The SC-server receives tasks and accesses the PSD to construct a geocast region (GR) that contains sufficient workers such that the queried task is accepted with high probability. The SC-server initiates a geocast communication [84] process to disseminate task  $t$  to the workers in the GR. In this case, the SC-server is not allowed to directly establish a contact with the workers inside the GR as it would result in identifying whether a contacted worker is real or fake. Therefore, the communication could be either done through CSP for all the workers or the CSP contacts one of the workers present in the GR and the message would be conveyed on a hop-by-hop basis to the remaining workers in the GR.

Considering privacy concerns complicate the task assignment scenario and reduces the effectiveness and efficiency of the assignment strategies. As the PSD contains false data, there might be scenarios where the geocast region does not contain any workers, and the task has been again queried against the PSD resulting in significant overhead.

The techniques proposed in [106, 133] are based on the scenario where the worker's location does not change, however, in often cases the worker location changes based on her dynamic movement. To enquire the latest location of the workers, SC-server has to request a new PSD release to perform task assignment. However, disclosing multiple versions of the sanitised PSD would be vulnerable to attacks. Conversely, if the PSD increases the noise on every subsequent release, then the data could become potentially useless as SC-server would continuously request for the latest PSD release. [105] tackled this problem by investigating privacy budget allocation techniques across consecutive PSD releases.

The previous studies utilizing the DP-based privacy-preserving task assignment frameworks assume a trusted entity to sanitize the location data [106, 133]. As there is no explicit trust relationship between any two parties in SC, a broader privacy setting where all SC parties are curious, i.e. they learn as much as possible about the inputs, but not malicious would be more appropriate. [110] proposes a three-stage privacy-aware framework, SCGuard, that protects workers and tasks locations without assuming any trusted entity. In the first stage, the SC-server selects a set of potential candidate workers for a given task based on the proximity (calculated based on the perturbed locations), and forwards them to the requester. In the second stage, the requester identifies the most reachable worker from the given set of workers sent by the SC-server and sends the task location to the identified worker. In the third and final stage, the selected worker would accept or reject the received task based on the task's location, i.e., whether the task lies inside the preferred region of the worker. The SCGuard framework assumes a semi-honest adversary model. However, in real-world scenarios this model might not hold as the requesters intent can be malicious, for example, requesters can fake tasks to estimate the workers' locations. Similarly, this approach focuses on the task assignment at a single time instance instead of considering the dynamic nature of workers and tasks. Furthermore, this work assumes that the spatial task does not require multiple workers for completing the task. However, there are many types of tasks that require responses from multiple workers to ensure quality, for instance, reporting the traffic at a road junction.

### 9.4 Encryption Techniques

Encryption techniques refer to encoding the location information of the workers and tasks so that it can be decoded only by authorized entities with a decryption key. [97] proposes an encryption-based approach to protect workers, identity and location. The *Task service* utilizes onion encryption to hide the worker's identity and location through the *anonymizing network*, Tor (<http://www.torproject.org/>). Though the approach is designed to ensure

workers' anonymity, there is a possibility for attacks like end-to-end timing correlation attack, as Tor does not try to protect against an attacker with access to the incoming and outgoing traffic of the Tor network. The *Task Service* can perform a timing attack on the worker's location by linking multiple task requests. To prevent such attacks, the workers connect to *Task Service* at random intervals.

In [97], the authors have only focused on the WST mode, where the server has very little control over the task assignment. [96] proposes an encryption-based task assignment approach to protect workers' location privacy in the SAT mode. The proposed privacy framework utilizes a semi-honest third party - *Privacy Service Provider(PSP)* and collects encrypted data from workers along with the encrypted location information. The SC-server performs the worker-task matching by communicating with the PSP in the encrypted domain. The worker close to the task and with a high degree of interest to the task would be chosen. The framework is not relying on a trusted third party as in the case of [106] and is robust to semi-honest adversaries. However, the semi-honest adversary model may not always hold in the real-world scenarios due to its limited privacy protection, as the SC-server and PSP might not follow the specific protocol or the requesters can be malicious.

Similarly, in [75], the locations of workers and tasks are protected by homomorphic encryption. This encryption-based task assignment approach utilizes the worker-task distances computed from the encrypted data. After assignment, the workers receive the encrypted task locations and decrypt them to obtain the exact locations. However, as with all the encryption-based approaches, the overhead computation cost is high when compared to cloaking and DP-based techniques. To improve the overhead computation cost, [74] proposed a strategy to eliminate some complex operations to achieve privacy protection with acceptable overheads. [74] combines partially homomorphic encryption schemes to efficiently perform complex assignment operations with encrypted data.

## 9.5 Impact on different constraints

While protecting the worker and task location privacy, the SC-server loses the ability to gain knowledge about individual workers for making better task assignments through a learning and optimization process. However, some of the constraints like travel budgets [87] and rewards [132, 133] could be incorporated while assigning tasks to workers (see Table A.6). It can be observed that there is very little literature available in SC that combines quality constraints and privacy protection approaches. This is because it is very difficult to assure a certain level of quality from the task outcomes, due to the lack of information about the workers. The approaches to privacy protection and task's quality constraints are negatively impacted by each other as

## 9. Privacy Protection

	<b>Spatial</b>	<b>Quality</b>	<b>Budget</b>
Privacy	STAC [87], Hu et al [51]	SUR Proto- col [125]	Zhang et al [133], QOI [132]

**Table A.6:** Comparison of privacy protected task assignment protocols with multiple constraints

privacy-protected SC might hinder the quality assessment of the individual responses and vice versa. For example, if a task requires a worker with a minimum reputation score and certain skills to be assigned, then the SC-server should be able to know the lower bounds of individual workers' reputation scores and their skillset. However, in a privacy-enabled setting, the SC-server does not possess the knowledge of individual workers needed to assure the requester that the task is assigned satisfying the quality constraints. For example, in [139] the SC-server in the cloud broadcasts the requested task to all the workers, as the workers' information is encrypted, thus reducing the control exerted by the SC-server to assure a certain level of quality.

However, [125] proposes a Secure User Recruitment (SUR) protocol to assure a certain level of quality while protecting the workers' privacy. SUR does not depend on encryption/decryption operations or any other trusted third-party servers. SUR is based on a secret sharing scheme between the different workers. The recruitment decision lies with the workers as the computations are performed at the worker's end. This method incurs a huge computation and communication cost between the workers during the computation process. Furthermore, the SC-server has to broadcast the tasks for all the workers, who would then jointly perform the recruitment.

### 9.6 Protecting Task and Requesters' Location Privacy

The majority of the current SC literature focuses on protecting the workers' location privacy. However, it ignores the need to protect the task and the requester's location privacy. With task locations being public, there is a privacy risk as the task locations can indirectly reveal the requesters' location. For example, if a requester is posting tasks in the same area, then chances are that the requester's home or workplace would be located in the same area. In [74, 75] the tasks' and requesters' locations are protected through an encryption-based privacy preserving framework. Similarly, [110] protects tasks' and requesters' location privacy through perturbation-based privacy preserving framework.



## 10 Truth Discovery and Crowdsourced Data Aggregation

### 10.1 Introduction

Truth discovery [43, 72] in SC involves identifying trustworthy information from the responses received from the workers. Truth discovery is relevant when a task is performed by multiple workers, wherein every worker provides uniform answers or conflicting answers. For example, if the task involves identifying the crowd situation in a restaurant at a particular time, there is a high likelihood of receiving differing answers from the workers. The worker might intentionally provide a wrong answer to complete the tasks for earning rewards, or to accumulate points. Sometimes, the differing answers could be unintentional, like the worker has answered about the crowd situation at a different time of the day, which is not relevant to a task. Similarly, some of the spatial tasks are of qualitative nature and require subjective responses. As subjective responses are dependent on the perception of the workers, all of the workers might be answering correctly according to their background. For instance, the task of labelling the crowd at a supermarket might evoke different responses which are all “correct” in the minds of the workers. The workers from a village background might find the supermarket to be crowded, whereas the workers from a city background might classify it as normal. In such cases, identifying the truthful outcome is of utmost importance. All the responses from the workers are aggregated into a single value before relaying the information to the task requester. There are numerous methods in CC to model the worker’s reliability and the quality of the aggregated value of the collected task responses [53, 90]. Some of the methods are non-iterative in nature like Majority Voting [65], Honeypot [68], and Expert Label Injected Crowd Estimation (ELICE) [63]. Other methods are of iterative nature like Expectation Maximization [54], Generative Model of Labels, Abilities, and Difficulties (GLAD) [122], Supervised Learning from Multiple Experts (SLME) [89], and Iterative Learning (ITER) [57].

### 10.2 Non-Iterative Aggregation

The non-iterative aggregation techniques compute a single aggregated value for each question utilizing heuristics like the most common answer. In this section, we will discuss three common non-iterative aggregation techniques, namely Majority Voting, Honeypot and ELICE.

**Majority Voting:** Majority Voting aggregates to the most recurring specific value among the crowdsourced responses [65]. This aggregation technique does not have any preprocessing involved and does not consider the workers’



expertise. Therefore, in the case of potential spammers, the technique might produce an incorrect outcome.

**Honeypot:** Honeypot aggregation technique is similar to *Majority Voting*, with an additional preprocessing step for detecting fraudulent workers [68]. A set of questions with known answers is randomly merged into the original question set to set up a trap for cheaters. The workers who fail to answer a predetermined number of the added questions (with known answers) will be considered as frauds and removed from the worker set. The probability of each possible value will then be computed with the remaining workers based on the *Majority Voting* technique. However, the success of this technique depends on the set of questions with known answers. In cases, where such questions are not available or if the answers to questions are of a subjective nature, then there is a chance for incorrectly identifying the frauds.

**Expert Label Injected Crowd Estimation (ELICE):** ELICE is similar to *Honeypot*, except that the question set with known answers would also be used to estimate the expertise level of each worker and the difficulty level of each question [63]. As each answer is weighted by worker expertise and question difficulty, it performs better than the *Majority Voting* and *Honeypot*. However, it has the similar disadvantages as *Honeypot*, with success based on the question set with known answers.

### 10.3 Iterative Aggregation

The iterative aggregation techniques consist of a sequence of iterations, that computes probabilities of answers for each question in each iteration until convergence [54]. In this technique, the set of questions with known answers are not required. In this section, we will consider the four iterative aggregation techniques: EM, GLAD, SLME, and ITER.

**Expectation Maximization (EM):** The EM technique simultaneously estimates all the probabilities of answers for each question iteratively in two steps: expectation and maximization [54]. In the expectation step, the probabilities of answers for each question are estimated according to the current estimates of their expertise. In the maximization step, this technique re-estimates the expertise of workers based on the current probability of the answer for each question. The process will stop after the probabilities of answers for each question are unchanged in subsequent iterations. Due to the possibility of numerous iterations for reaching convergence, cost of execution might be an issue. Furthermore, EM provides a locally optimal solution rather than a globally optimal one. [28] improves the EM method using pruning and search-based approach to provide a globally optimal solution.

**Generative Model of Labels, Abilities, and Difficulties (GLAD):** This technique is similar to EM, except that it estimates toughness of the question along with workers' expertise [122] like the non-iterative aggregation tech-

nique ELICE [63]. The technique tries to address two special cases: a worker with higher expertise is more likely to answer a question correctly, and a question with higher difficulty has a lower probability of being answered correctly. The initial values of the worker expertise and question toughness impacts the performance of this technique.

**Supervised Learning from Multiple Experts (SLME):** This technique is similar to EM, however, instead of a confusion matrix, it characterizes the worker expertise by *sensitivity*, which is the ratio of positive answers that are correctly answered and *specificity*, which is the ratio of negative answers that are correctly answered. Due to the limitation of binary labeling for sensitivity and specificity, the SLME technique is incompatible with multiple labels.

**Iterative Learning (ITER):** This technique [57] is based on standard belief propagation. Similar to GLAD [122] and ELICE [63], ITER also estimates the toughness of the question and the worker expertise. However, unlike the other aggregation methods which assume the reliability of all answers provided by the worker as a single value, ITER computes the reliability for each answer separately. Similarly, for each worker, the toughness of a question is computed. The worker expertise is estimated as the sum of the reliability of her answers weighted by the toughness of answered questions. Unlike the other aggregation techniques, the initial values of answer reliability and question toughness do not impact the performance of ITER.

## 10.4 Truth Discovery in Spatial Crowdsourcing

However, the current literature in CC does not consider the spatial attributes of the workers and the tasks. They are not effective when both the worker's reliability and her mobility is uncertain. Moreover, there are other spatial attributes that should be considered for truth discovery, like distance to the task from the worker's location [50], task location popularity/influence, location visit tendencies [85]. [50] proposed a probabilistic graphical model for truth inference in crowdsourced POI labelling. The inference model utilizes the location data of the workers and the POIs, along with the POI location influence, and the worker's reliability. The estimated worker reliability and the POI influence would be utilized by the task assignment module to recommend relevant POIs to workers while ensuring quality. Based on the responses collected from the assigned tasks, the inference model will update the worker's reliability information. However, this model does not require the workers to be physically present at the POIs in order to label them. Furthermore, it does not consider the worker's individual visit information to the POI locations to ascertain the trustworthiness of the worker.

The model proposed in [50] requires continuous tracking of the location of the worker, which could lead to a privacy issue. To address this, [85] proposed a probabilistic graphical model, Truth Finder for Spatial Events (TSE),

that does not require continuous location tracking of the workers, it instead considers location popularity and the individual worker historic visit indicators to the locations of tasks, in addition to the worker's reliability. However, estimating location popularity does not resolve the issue whether an individual worker would visit that location or not. Hence, an improved model, Personalised Truth Finder for Spatial Events (PTSE), based on personal location visit tendencies of workers is proposed. Similar to [85], [45] has proposed a truth inference algorithm based on the Bayesian estimation model. This model considers the state of the event or task at different time instants based on the received worker responses.

## 11 Applications

As discussed earlier, SC paradigm has the potential of solving many real-world problems like collecting motion traces from building inhabitants to construct floor-plans [3]. According to a survey conducted on crowdsourcing systems in 2011 by Yuen et al, [130], we can classify the crowdsourcing applications into the following four groups: Voting System, Information Sharing Systems, Game and Creative System. Inspired from this classification categories, we have grouped the SC applications into three broad categories based on the utilization of sensors, human knowledge, and human efforts: data collection, query answering, and personal service. Generally, the tasks involving quantifiable information fall under the data collection category and the ones involving qualitative information fall under the query answering category.

### 11.1 Data Collection

Data Collection refers to the process of gathering information from the different sensors of workers' smartphones at particular locations, without utilizing the workers' knowledge capabilities. Most of the applications belonging to this category are labelled with "mobile crowdsensing" instead of SC. To describe the data collection process, let us take an example of building indoor floor plans from the traces of movement by the workers with the help of their smartphones [3]. These motion traces based on the inertial sensors in the workers' smartphones, which are collected and processed later to build accurate indoor floor plans. Similarly, there are applications to collect data about the public transport drivers' routing behaviour to detect traffic anomalies [86]. A different version of the public transportation crowd-sourced routing algorithm is proposed in [24]. Furthermore, there are applications to collect the accelerometer data to detect the movement of vehicles in a parking lot to determine the parking availability [83].

Many SC applications concentrate on post-processing of the collected

data, to improve the quality of the collected data or to generate meaningful data from the raw data. For example, in [3], the collected raw sensor data is processed to generate accurate motion traces of the users. Similarly, post-processing is employed to improve the geo-spatial linked open data ([56], [55]). Post-processing could be performed in real-time [8, 86] or after the completion of data collection phase [3]. The decision to choose a real-time post-processing option would be based on the objective of the SC application.

## 11.2 Query Answering

Query Answering is another class of SC application that involves harnessing the worker's knowledge to answer a group of questions, related to a specific location or region. Unlike the case of data collection, the information gathering process is not just limited to collecting sensor data of workers' smartphones. Query Answering also utilizes the worker's skill and the ability for answering the queries/tasks. For example, CrowdHelp [8] utilizes query answering concept to improve emergency response for patients during a disaster. The system helps workers assess a patient's physical condition and symptoms, through a series of queries. Similarly, high-quality maps are generated based on the inputs from the crowd [13].

The query answering applications can be further divided based on the type of queries answered. Some of the different types of queries are simple binary (Yes/No) queries, multiple options queries, tagging the images with relevant tags, categorizing different images, describing images, and classifying the type of a spatial feature [119].

Applications consider spatial constraints while collecting the answers from the workers. For example, the option of answering a query is only visible to the workers in the vicinity of the question's geographical location [46]. Similarly, the answers collected are prioritized according to the proximity of the worker to the queries' location. For instance, in [18], the priority was given to the answers provided by the identified local experts among the Twitter users.

## 11.3 Personal Service

Personal Service is another class of SC application that involves an additional human effort to perform the task, like pick-up and delivery of a package/-groceries/food order<sup>3</sup>, taxi calling and ride sharing<sup>4</sup>, performing a task like painting/cleaning/lawn mowing<sup>5</sup>. The personal service applications have to consider the different spatio-temporal, budget and quality constraints for

---

<sup>3</sup>[www.ubereats.com](http://www.ubereats.com)

<sup>4</sup>[www.uber.com](http://www.uber.com)

<sup>5</sup>[www.taskrabbit.com](http://www.taskrabbit.com)

solving the task matching and task scheduling problems. For example, the Ubereats<sup>3</sup> application involves collecting the food packages from the restaurant and delivery to the customers. The Ubereats SC application matches the food delivery tasks with the worker and plans the route for delivering them.

### 11.4 Example SC applications

In this section, we choose a representative set of applications, including Uber<sup>4</sup>, TaskRabbit<sup>5</sup>, and gMission [14], and discuss their features and relevance to SC.

**Uber:** is an (unlicensed) taxi calling and a peer-to-peer ride-sharing SC application, wherein passengers attempt to hire a taxi or share a ride. It belongs to the personal service class of SC applications. During the operation, Uber's back-end server matches the service requesters and service providers, i.e., drivers, in accordance with their spatiotemporal proximity. For taxi-calling service, the server matches the passenger to the nearest available drivers, according to their selected taxi type. For carpooling services like *UberPool* ride-sharing service, the server matches the potential passengers, who are willing to share the ride to the nearest available driver according to the respective passengers' and driver's source and destination locations. The general optimization target is to improve the service delivery rate and the earnings of the service providers. The server considers constraints like the maximum number of trip requests per day for the driver, pick up time and vehicle choice of the passenger. In particular, constraints such as driver's destinations are further added to ensure that the pickup locations of trip requests will not be far from the destination of the driver.

**TaskRabbit<sup>5</sup>:** is an online and mobile marketplace, wherein requesters can post tasks that can be performed by verified workers. It belongs to both the query answering and personal service class of SC applications. TaskRabbit services tasks like cleaning the apartment, picking up and delivering the groceries from the supermarket, researching for a party, handyman work. Based on the description of the task provided by the requester, TaskRabbit matches the tasks' required skills with the workers' specified skills within a neighborhood and provides a list of qualified workers for the requester to choose according to their hourly rates. For suggesting workers to the requester, TaskRabbit utilizes both the worker skills, that are verified through a vetting process and worker reputation, based on the feedback from the past tasks.

**gMission [14]:** is an open-sourced, general purpose SC application that supports a variety of spatial tasks. The application belongs to both the data collection and query answering classes of SC applications. gMission offers credits as an incentive for performing a task. For a newly registered user to request for a task, one has to spend the credits earned through performing tasks. gMission uses *K-nearest neighbors* algorithm for task assignment, uti-

lizes *Majority Voting* for response aggregation, and improves task workload distribution among workers through dynamic weights ascertained by previous assignments. However, gMission do not consider worker expertise/skill for assigning tasks to workers.

Recently, there is a massive surge in SC applications servicing different purposes. However, the majority of the current SC applications do not conform to the latest developments in SC literature for the deployment of more advanced applications. For instance, except for a few applications like Uber, the majority of existing SC applications do not recommend workers to move to a different place for more tasks, based on the historical information [131]. Also, general SC applications like gMission can improve the user participation by including an incentive mechanism such as monetary rewards for tasks with respect to the levels of workers' expertise or reputation [27, 109]. Moreover, privacy concerns can be addressed for existing SC applications [37], e.g., gMission, so that the sensitive identity or location information is not leaked during the running of applications. Furthermore, scheduling mechanisms [73] can be incorporated in SC applications, such as gMission and TaskRabbit, for enhancing the task completion ratio, if the user participation ratio is high.

## 12 Discussion

Although research is gaining momentum in SC, it is still in the nascent stage with a lot of open research issues. In this section, we will discuss the different challenges and limitations of existing research work. These challenges are related to task matching and scheduling problems, truth inference models, privacy issues, and the lack of real-world datasets.

### 12.1 Task Matching and Scheduling issues

The task matching and scheduling issues are related to the dynamic arrival of tasks and workers, the optimization goals, the immutability constraint, and the workers' movement patterns.

The majority of the work done in SC does not account for the dynamic arrival of workers and tasks to the SC-server during the task assignment phase [15, 26, 61] or task scheduling phase [29]. They work on the assumption that the SC-server possesses the complete knowledge of the inputs sets of tasks and workers. This renders their solutions inefficient in dynamic real-time environments. Recently, some solutions were proposed to address the task matching problems in online scenarios like [47, 104, 115], however, only tasks are considered to be dynamic. In [113], the authors proposed a solution to factor the dynamic nature of both workers and tasks arrival. However,

[113] does not provide support for constraints like quality and privacy.

The current task matching approaches do not consider the “workplaces” [98] aspect of SC. For instance, InterestingSport (<http://www.quyundong.com/>) is a SC application that helps users to find suitable trainers and book sport facilities in real-time. Therefore, in addition to checking the availabilities of the users and trainers, the SC-server has to take into account the availability of the sport facilities. [98] formulates a Trichromatic online matching in real time SC (TOM) problem that considers workplaces along with the tasks and the workers during the assignment. The “workplaces” extension could be easily accommodated in the existing worker-task matching framework by utilizing the “workplaces” information as additional constraints to the worker-task matching problem. However, further analysis should be done to ascertain the impact this additional constraint has on existing constraints like privacy, budget and quality.

Similarly, [73] proposes solutions to schedule dynamically arriving tasks to a single static worker. There is a lack of research for scheduling dynamically arriving tasks for multiple workers. [30] tries to address this issue by considering a predicted set of tasks and worker. An iterative strategy is proposed for sequentially assigning a set of tasks to workers and scheduling the assigned tasks to each worker. However, this strategy does not consider the compatibility of a task to the worker’s schedule before assignment, thereby resulting in an overhead of re-assigning the unscheduled task to a different worker.

The optimization goals for the different task matching problems in the SC-literature are focused on benefitting either the worker or the task. If the SC-server assigns tasks to workers based on the preferences of tasks, then it would benefit the tasks. For example, assigning tasks to workers that have a minimum reputation [62] or a given set of skills [109]. Similarly, if the SC-server assigns tasks to workers based on their preferences, it would benefit the workers. For example, if the SC-server tries to maximize the reward received by workers [27]. Benefitting either the tasks or the workers would result in inefficient assignment for the other. For example, the workers may fail to find tasks that they would like to perform and the SC-server may fail to find workers that provide better quality responses. Therefore, to address this dilemma, it would be beneficial if both the workers and tasks preferences are taken into account in the task assignment scheme.

Almost all the solutions proposed to solve the task matching problems follows the rule of immutability for task assignments, i.e., when a task is assigned to a worker, it cannot be revoked. The immutability constraint is introduced to prune the search space by removing the assigned tasks. Although this constraint reduces the search space, there are cases where this rule might not be beneficial. Some of these cases are the online scenarios with the objectives of improving task acceptance rate and quality of task responses. For



example, a new task with better utility/reward or a new worker with better reputation/skill might be available after assignment. Therefore, revoking the immutability constraint on task assignments would help in re-assignment of tasks to improve the reward/utility received, leading to a higher task acceptance. Furthermore, the overall quality of task responses by the workers would be improved.

Furthermore, workers tend to accept tasks close to their commute. Therefore, considering the workers' movement patterns while assigning tasks [12] would improve the task acceptance rate. However, there are a lot of open issues regarding the utilization of the workers' movement patterns for task assignment. For instance, the worker route is not dynamically updated, even if there is a better task available with higher utility, i.e., less travel cost and higher pay. Similarly, with the existing strategies, there is a possibility that some workers monopolize the task assignments with their willingness to travel long distances, resulting in the reduction of worker diversity.

## 12.2 Privacy Issues

Privacy concerns are one of the most fundamental problems of the worker. Although some literature in SC addressed this issue (as discussed in Section 9), there are a lot of open issues that need to be addressed. Privacy concerns are fuelled by the workers' lack of trust on the third party SC-server. To address this, some solutions utilize the concept of differential privacy to get the workers aggregated data from worker-trusted cellular service providers [106], to anonymize workers from the SC-server. However, the SC-server would still have the knowledge of the task locations and the time interval during which the assigned worker would visit the task location, resulting in a serious privacy breach. Furthermore, by anonymizing the worker to the SC-server, the support for individual spatial constraints and quality constraints is hindered with the existing strategies. Although [125] deals with the privacy-enabled quality assurance problem, it does not consider the differing requirements of the tasks and the differing spatial constraints of the workers. Furthermore, the worker's reliability scores cannot be updated based on the success/failure of the assigned tasks in [125].

## 12.3 Truth Inference Models

In the existing truth inference models of SC, it is difficult to infer truth while considering worker's location privacy. For example, this is the case in location obfuscation, wherein the worker's location is generalized to protect the exact worker's location. In such cases, the existing truth inference models [85] simply ignore the workers with obfuscated locations. However, considering the individual workers' location privacy concerns, new models are needed to



tackle these issues.

Similarly, the current truth inference models of SC assumes that the workers respond independently of each other, i.e., there would not be any copying or sharing of answers among the workers. However, in reality, there could be a case of copying among the workers, that could result in improper estimation of worker reliability, thereby affecting the quality of the task responses. Therefore, copy-detection methods are needed while inferring truth from the worker responses [31]. To prevent collusion between workers, it is important to quantify the probability that the workers collude with each other based on the quality of their responses to better assign the tasks. For example, in CC literature, [102] has proposed a three-step based  *$\theta$ -secure task assignment approach* for task assignment avoiding collusion between workers. Similar approaches are needed in SC considering the spatial characteristics of workers and tasks. Furthermore, new approaches should be proposed similar to [28] for providing a global optimal solution in estimating the task ground truth and worker expertise in SC.

#### 12.4 Lack of real world datasets

The primary challenge faced by all the solutions proposed in the SC literature is to evaluate the strategies based on real-world datasets. Due to the lack of publicly available real-world datasets, SC algorithms are evaluated utilizing synthetic datasets, that are generated based on different distribution functions. Some SC works utilize few real-world datasets related to location-based social networks (LBSN) like Gowalla (<http://snap.stanford.edu/data/loc-gowalla.html>) and Bright-kite (<http://snap.stanford.edu/data/loc-brightkite.html>), where users can check-in to different points of interest in their vicinity. These LBSN datasets are adapted to the SC scenario by assuming the check-in spots as task locations, users as workers and a user checking into a spot is considered as accepting the task [61, 137]. To et al. [103], advanced this strategy to generate synthetic SC datasets with realistic spatiotemporal properties and constraints adapted from geosocial datasets like Gowalla and Yelp ([http://yelp.com/dataset\\_challenge](http://yelp.com/dataset_challenge)). The advantage with these datasets is that they exhibit the workers nature of preferring to perform nearby tasks. However, there might be different geosocial phenomena in SC that are not observed with either synthetic or adapted datasets. There is a need to design a SC platform to collect real-world data for researchers to advance the research in SC.

#### 12.5 Lacking User Participation

For an SC application to be successful, it should be able to attract many task requesters and workers. Most of the existing applications are based on volun-

tary participation, and as performing tasks involve spending time, effort and resources such as smartphone’s memory and battery, it is difficult to attract workers without offering rewards. Moreover, with the privacy concerns, the users might not be willing to participate in the SC application. To motivate users to participate in SC, incentive mechanisms are developed that involve monetary rewards [27, 48, 67] and virtual credits [14, 66]. Though incentive mechanisms have a positive impact on improving the user participation, it is still limited to the users who are aware of the SC paradigm. To harness the true potential of SC, new methods should be developed to enable the general public to become aware of the SC application and to convince them to participate in SC applications.

### 13 Future Research Directions

In this section, we provide some of the potential research directions in SC, based on the discussion in Section 12.

#### **Improving Task Assignment Protocols in Online Scenarios:**

As discussed in Section 12, task assignment protocols should be improved to tackle the uncertainty of the dynamic real-time SC environment. Although, different solutions like [113] are proposed to address this, further research needs be performed for improving the efficiency and for ascertaining the impact on different constraints like quality and privacy. In particular, strategies should be proposed to perform task assignment in an online scenario with privacy-enabled SC.

Furthermore, as discussed in Section 12, the existing task assignment strategies are restricted by the immutability constraint on task assignments. However, in uncertain online scenarios, it would be beneficial to revoke this constraint and check for all the suitable tasks and workers. There would be drawbacks for revoking the immutability constrain, like the revoked task might not find a suitable worker afterwards and could remain unassigned or expire before being assigned, the worker might have started the travel to the current task location, wherein changing the assignment would cause confusion and dissatisfaction. However, if the workers and SC-server have a prior agreement regarding the potential change in assignments and a set of pre-requisites are agreed upon, like a threshold time for updating tasks, then the immutability constraint could be beneficial and practical. Correspondingly, strategies should be proposed allowing the update of assigned tasks based on the available tasks and workers respecting the different constraints like deadlines and required expertise.

#### **Assignment Protocols that Benefit Both Workers and Tasks:**

As discussed in Section 12, the existing task matching protocols either attempt to assign tasks to workers based on workers’ preferences or tasks pref-

erences. Optimizing the benefits for both the tasks and the workers would improve the success of the assignment protocol. A new assignment strategy could be proposed by adapting the solution proposed in the CC literature [138] with spatiotemporal context. Zheng et al, [138] has proposed a task assignment framework in CC, called Task Assignment with Mutual Benefit Awareness (TAMBA), to offer mutual benefit to workers and tasks, based on their preferences extracted from the historical data. Similarly, [5] have proposed an auction-based framework, Auction-SC, that would benefit both workers and the SC-server by allowing workers to bid on the arriving tasks.

#### **Integration of task publishing modes:**

There is a need for an effective framework that combines both task publishing modes (WST and SAT), to provide effective solutions for improving the efficiency of SC and the task acceptance rate. For instance, the workers can select their preferred tasks via WST mode. The case where multiple workers are opting for the same task, the SC-server can employ the SAT mode to resolve the conflict.

#### **Privacy-enabled Truth Inference Models:**

The current truth inference models in SC, can avoid real-time location tracking of the workers and exploit the historical information for inferring truth and to worker reliability models. However, in a privacy-protected SC, the individual worker locations would be unknown to the SC-server, which the current inference models cannot support. Therefore, new truth inference models should be proposed for the different location privacy models to ensure quality to the task requester. Furthermore, new techniques are needed to process complex textual or multimedia information to assess the trustworthiness of the responses [90].

#### **Quality Assurance-Privacy Preservation Trade off:**

As discussed in Section 9.1, privacy preservation and quality assurance negatively affect each other as privacy-protected SC might hinder the quality of the responses and vice versa, especially in the case of single response tasks with differing requirements. Therefore, a trade-off mechanism is needed to balance the privacy requirements of workers and quality constraints of tasks set by requesters. For instance, with some cloaking techniques like k-anonymity and quality constraint information like reliability and expertise skill set can be aggregated on the client side of the workers. An aggregate query of k workers can be sent to SC-server along with the cloaked region and quality constraints information for task assignment. Furthermore, the location information of tasks should also be protected from the SC-server along with the worker information to avoid privacy breach.

#### **Harnessing Worker Movement Patterns:**

Mining workers' movement patterns would help us in predicting workers' movement behavior and her availability for performing tasks. The predicted information about workers' movement would allow the SC-server to know

about the workers' arrival order, thereby reducing the uncertainty in the on-line task assignment scenario. The outcome of such assignments would be the trajectory of the worker containing the assigned tasks. Furthermore, an additional constraint like the order of the tasks, should be considered during task assignment, if there exists a sequential dependency between the tasks, i.e., task  $t_A$  should be performed after task  $t_B$ . In [121], a worker mobility prediction model was proposed to align the workers mobility with spatial task requirements for task assignment.

#### **Improving User Participation in SC:**

As discussed in Section 12, there is a need to devise better strategies to attract users to improve the participation in SC applications. The existing strategies like incentive mechanisms are useful to an extent. However, they are still limited to the userbase familiar with SC. The new strategies should expand the reach of the SC applications to users who are not so familiar with SC as well.

#### **Harnessing Geo-Social Network Information:**

Current SC literature contains little work considering the geosocial relationships between the workers that could be helpful in enriching the worker's profile. Location influence [93] of workers can be utilized to provide a partial ranking to workers in team-oriented task planning [42]. Resulting partial ranking helps SC-server to select the leaders for the teams. Furthermore, location influence concept can be used in allocating rewards to the workers in a dynamic budget reward model. workers with the highest location influence ranking would be attracted to perform the task by offering higher reward. By attracting the highly ranked location influencers, their followers are attracted to the task location. Thus, increasing the worker diversity of the location and improving the chances for a task to be assigned in the neighborhood.

## **14 Summary**

In this survey, we reviewed the existing literature related to SC from a technical perspective. We distinguished different topics in the research and proposed our taxonomy to organize them. We noticed that the architecture of SC adapts the structure of CC to serve spatiotemporal interests. Furthermore, we observed that the majority of the existing work focuses on task matching along with varying constraints since the SC-server exerts more control on the task matching proceedings. Similarly, we observed a significant density of research focusing on offline scenarios. Our comparison study revealed the shortcomings of the different strategies and identified relationships between the various constraints of SC. The quality constraints are found to be negatively impacted by the privacy protection approaches and positively correlated with the budget constraints. The identified shortcomings and chal-

allenges are related to the task assignment in online scenarios, the dynamic movement of workers, the privacy-quality trade-off, and the geosocial relationships. We suggest future work to address these challenges and advance the application spectrum of SC.

## Acknowledgements

*We are grateful to anonymous reviewers for their constructive comments on this work. This research has been funded in part by the European Commission through the Erasmus Mundus Joint Doctorate "Information Technologies for Business Intelligence - Doctoral College" (IT4BI-DC), Fundamental Research Funds for the Central Universities, National Natural Science Foundation of China (No. 61772492, No. 61672487), and Natural Science Foundation of Jiangsu Province (No. BK20171240). Xike Xie is supported by the CAS Pioneer Hundred Talents Program.*

## References

- [1] M. Allahbakhsh, B. Benatallah, A. Ignjatovic, H. R. Motahari-Nezhad, E. Bertino, and S. Dustdar, "Quality control in crowdsourcing systems: Issues and directions," *IEEE Internet Computing*, vol. 17, pp. 76–81, 2013.
- [2] F. Alt, A. S. Shirazi, A. Schmidt, U. Kramer, and Z. Nawaz, "Location-based crowdsourcing: extending crowdsourcing to the real world," in *NordiCHI*, 2010, pp. 13–22.
- [3] M. Alzantot and M. Youssef, "Crowdinside : Automatic construction of indoor floorplans," in *GIS*, 2012, pp. 99–108.
- [4] M. Asghari and C. Shahabi, "An on-line truthful and individually rational pricing mechanism for ride-sharing," in *GIS*, 2017, p. 7.
- [5] —, "On on-line task assignment in spatial crowdsourcing," in *Big Data*, 2017, pp. 395–404.
- [6] J. Bar-Ilan, G. Kortsarz, and D. Peleg, "Generalized submodular cover problems and applications," *Theoretical Computer Science*, vol. 250, no. 1, pp. 179–200, 2001.
- [7] K. Benouaret, R. Valliyur-Ramalingam, and F. Charoy, "Answering complex location-based queries with crowdsourcing," in *Collaboratecom*, 2013, pp. 438–447.
- [8] L. I. Besaleva and A. C. Weaver, "Crowdhelp: Application for improved emergency response through crowdsourced information," in *UbiComp*, 2013, pp. 1437–1446.
- [9] D. C. Brabham, "Crowdsourcing as a model for problem solving: An introduction and cases," *Convergence*, vol. 14, no. 1, pp. 75–90, 2008.
- [10] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, "Participatory sensing," in *ACM SenSys Workshop, World-Sensor-Web*, 2006, pp. 1–5.

## References

- [11] G. Chatzimilioudis, A. Konstantinidis, C. Laoudias, and D. Zeinalipour-Yazti, "Crowdsourcing with smartphones," *IEEE Internet Computing*, vol. 16, no. 5, pp. 36–44, 2012.
- [12] C. Chen, S.-F. Cheng, A. Gunawan, A. Misra, K. Dasgupta, and D. Chander, "Traccs: A framework for trajectory-aware coordinated urban crowd-sourcing," in *HCOMP*, 2014, pp. 30–40.
- [13] X. Chen, X. Wu, X.-Y. Li, X. Ji, Y. He, and Y. Liu, "Privacy-aware high-quality map generation with participatory sensing," *IEEE Transactions on Mobile Computing*, vol. 15, no. 3, pp. 719–732, 2016.
- [14] Z. Chen, C. C. Cao, L. Chen, and C. J. Zhang, "gMission : A General Spatial Crowdsourcing Platform," in *PVLDB*, 2014, pp. 1629–1632.
- [15] P. Cheng, X. Lian, L. Chen, J. Han, and J. Zhao, "Task assignment on multi-skill oriented spatial crowdsourcing," *TKDE*, vol. 28, no. 8, pp. 2201–2215, 2016.
- [16] P. Cheng, X. Lian, L. Chen, and C. Shahabi, "Prediction-based task assignment in spatial crowdsourcing," in *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE, 2017, pp. 997–1008.
- [17] P. Cheng, X. Lian, Z. Chen, R. Fu, L. Chen, J. Han, and J. Zhao, "Reliable diversity-based spatial crowdsourcing by moving workers," *PVLDB*, vol. 8, no. 10, pp. 1022–1033, 2015.
- [18] Z. Cheng, J. Caverlee, H. Barthwal, and V. Bachani, "Who is the barbecue king of texas?: a geo-spatial approach to finding local experts on twitter," in *SIGIR*, 2014, pp. 335–344.
- [19] M. H. Cheung, F. Hou, and J. Huang, "Make a difference: Diversity-driven social mobile crowdsensing," in *INFOCOM*, 2017, pp. 1–9.
- [20] A. I. Chittilappilly, L. Chen, and S. Amer-Yahia, "A survey of general-purpose crowdsourcing techniques," *TKDE*, vol. 28, no. 9, pp. 2246–2266, 2016.
- [21] Y. Chon, N. D. Lane, Y. Kim, F. Zhao, and H. Cha, "Understanding the coverage and scalability of place-centric crowdsensing," in *UbiComp*, 2013, pp. 3–12.
- [22] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos, "Anonymsense: privacy-aware people-centric sensing," in *MobiSys*, 2008, pp. 211–224.
- [23] J. Cranshaw, E. Toch, J. Hong, A. Kittur, and N. Sadeh, "Bridging the gap between physical location and online social networks," in *UbiComp*, 2010, pp. 119–128.
- [24] T. T. Cuong, "Crowdroute: A crowd-sourced routing algorithm in public transit networks," in *GIS*, 2013, pp. 9–14.
- [25] G. Danezis, S. Lewis, and R. J. Anderson, "How much is location privacy worth?" in *WEIS*, vol. 5, 2005.
- [26] H. Dang, T. Nguyen, and H. To, "Maximum Complex Task Assignment: Towards Tasks Correlation in Spatial Crowdsourcing," in *IIWAS*, 2013, p. 77.
- [27] K.-H. Dang and K.-T. Cao, "Towards reward-based spatial crowdsourcing," in *ICCAIS*, 2013, pp. 363–368.

## References

- [28] A. Das Sarma, A. Parameswaran, and J. Widom, "Towards globally optimal crowdsourcing quality management: The uniform worker setting," in *SIGMOD*. ACM, 2016, pp. 47–62.
- [29] D. Deng, C. Shahabi, and U. Demiryurek, "Maximizing the number of worker's self-selected tasks in spatial crowdsourcing," in *GIS*, 2013, pp. 324–333.
- [30] D. Deng, C. Shahabi, and L. Zhu, "Task matching and scheduling for multiple workers in spatial crowdsourcing," in *GIS*, 2015, p. 21.
- [31] X. L. Dong, L. Berti-Equille, and D. Srivastava, "Truth discovery and copying detection in a dynamic world," *PVLDB*, vol. 2, no. 1, pp. 562–573, 2009.
- [32] P. Dutta, P. M. Aoki, N. Kumar, A. Mainwaring, C. Myers, W. Willett, and A. Woodruff, "Common sense: participatory urban sensing using a network of handheld air quality monitors," in *SenSys*, 2009, pp. 349–350.
- [33] C. Dwork, "Differential privacy," in *Automata, languages and programming*. Springer, 2006, pp. 1–12.
- [34] J. Fan, G. Li, B. C. Ooi, K.-I. Tan, and J. Feng, "icrowd: An adaptive crowdsourcing framework," in *SIGMOD*. ACM, 2015, pp. 1015–1030.
- [35] Y. Fan, H. Sun, Y. Zhu, X. Liu, and J. Yuan, "A Truthful Online Auction for Tempo-spatial Crowdsourcing Tasks," in *SOSE*, 2015, pp. 332–338.
- [36] J. Feldman, A. Mehta, V. Mirrokni, and S. Muthukrishnan, "Online stochastic matching: Beating  $1-1/e$ ," in *FOCS*, 2009, pp. 117–126.
- [37] W. Feng, Z. Yan, H. Zhang, K. Zeng, Y. Xiao, and Y. T. Hou, "A survey on security, privacy and trust in mobile crowdsourcing," *IEEE Internet of Things Journal*, pp. 2971–2992, 2017.
- [38] Z. Feng, Y. Zhu, Q. Zhang, L. M. Ni, and A. V. Vasilakos, "Trac: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing," in *IN-FOCOM*, 2014, pp. 1231–1239.
- [39] Z. Feng, Y. Zhu, Q. Zhang, H. Zhu, J. Yu, J. Cao, and L. M. Ni, "Towards truthful mechanisms for mobile crowdsourcing with dynamic smartphones," in *ICDCS*, 2014, pp. 11–20.
- [40] R. K. Ganti, N. Pham, H. Ahmadi, S. Nangia, and T. F. Abdelzaher, "Greengps: a participatory sensing fuel-efficient maps application," in *MobiSys*, 2010, pp. 151–164.
- [41] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39.
- [42] D. Gao, Y. Tong, Y. Ji, and K. Xu, "Team-oriented task planning in spatial crowdsourcing," in *APWeb-WAIM*, 2017, pp. 41–56.
- [43] J. Gao, Q. Li, B. Zhao, W. Fan, and J. Han, "Truth discovery and crowdsourcing aggregation: A unified perspective," *PVLDB*, vol. 8, no. 12, pp. 2048–2049, 2015.
- [44] R. Gao, M. Zhao, T. Ye, F. Ye, Y. Wang, K. Bian, T. Wang, and X. Li, "Jigsaw: Indoor floor plan reconstruction via mobile crowdsensing," in *MobiCom*, 2014, pp. 249–260.



## References

- [45] D. A. Garcia-Ulloa, L. Xiong, and V. Sunderam, "Truth discovery for spatio-temporal events from crowdsourced data," *PVLDB*, vol. 10, no. 11, pp. 1562–1573, 2017.
- [46] J. Gonçalves, D. Ferreira, S. Hosio, Y. Liu, J. Rogstadius, H. Kukka, and V. Kostakos, "Crowdsourcing on the spot: altruistic use of public displays, feasibility, performance, and behaviours," in *UbiComp*, 2013, pp. 753–762.
- [47] U. U. Hassan and E. Curry, "A Multi-armed Bandit Approach to Online Spatial Task Assignment," in *UIC*, 2014, pp. 212–219.
- [48] S. He, D.-H. Shin, J. Zhang, and J. Chen, "Toward optimal allocation of location dependent tasks in crowdsensing," in *INFOCOM*, 2014, pp. 745–753.
- [49] M. Hosseini, A. Shahri, K. Phalp, J. Taylor, and R. Ali, "Crowdsourcing: A taxonomy and systematic mapping study," *Computer Science Review*, vol. 17, pp. 43–69, 2015.
- [50] H. Hu, Y. Zheng, Z. Bao, G. Li, J. Feng, and R. Cheng, "Crowdsourced poi labelling: Location-aware result inference and task assignment," in *ICDE 2016*, 2016, pp. 61–72.
- [51] J. Hu, L. Huang, L. Li, M. Qi, and W. Yang, "Protecting location privacy in spatial crowdsourcing," in *APWeb*, 2015, pp. 113–124.
- [52] C. Huang, D. Wang, and S. Zhu, "Where are you from: Home location profiling of crowd sensors from noisy and sparse crowdsourcing data," in *INFOCOM*, 2017, pp. 1–9.
- [53] N. Q. V. Hung, N. T. Tam, L. N. Tran, and K. Aberer, "An evaluation of aggregation techniques in crowdsourcing," in *WISE*, 2013, pp. 1–15.
- [54] P. G. Ipeirotis, F. Provost, and J. Wang, "Quality management on amazon mechanical turk," in *ACM SIGKDD workshop on human computation*, 2010, pp. 64–67.
- [55] R. Karam and M. Melchiori, "A crowdsourcing-based framework for improving geo-spatial open data," in *SMC*, 2013, pp. 468–473.
- [56] —, "Improving geo-spatial linked data with the wisdom of the crowds," in *EDBT Workshop*, 2013, pp. 68–74.
- [57] D. R. Karger, S. Oh, and D. Shah, "Iterative learning for reliable crowdsourcing systems," in *Advances in neural information processing systems*, 2011, pp. 1953–1961.
- [58] R. M. Karp, "On-line algorithms versus off-line algorithms: How much is it worth to know the future?" in *Proc. of the Information Processing IFIP Congress (1)*, vol. 12, 1992, pp. 416–429.
- [59] R. Kawajiri, M. Shimosaka, and H. Kahima, "Steered crowdsensing: Incentive Design towards Quality-Oriented Place-Centric Crowdsensing," in *UbiComp*, 2014, pp. 691–701.
- [60] L. Kazemi and C. Shahabi, "A privacy-aware framework for participatory sensing," *ACM SIGKDD Explorations Newsletter*, vol. 13, no. 1, pp. 43–51, 2011.
- [61] —, "Geocrowd: enabling query answering with spatial crowdsourcing," in *GIS*, 2012, pp. 189–198.



## References

- [62] L. Kazemi, C. Shahabi, and L. Chen, "Geotrucrowd: trustworthy query answering with spatial crowdsourcing," in *GIS*, 2013, pp. 314–323.
- [63] F. K. Khattak and A. Salleb-Aouissi, "Quality control of crowd labeling through expert evaluation," in *NIPS 2nd Workshop on Computational Social Science and the Wisdom of Crowds*, vol. 2, 2011, p. 5.
- [64] I. Koutsopoulos, "Optimal incentive-driven design of participatory sensing systems," in *INFOCOM*, 2013, pp. 1402–1410.
- [65] L. I. Kuncheva, C. J. Whitaker, C. A. Shipp, and R. P. Duin, "Limits on the majority vote accuracy in classifier fusion," *Pattern Analysis & Applications*, vol. 6, no. 1, pp. 22–31, 2003.
- [66] K.-c. Lan, C.-M. Chou, and H.-Y. Wang, "An incentive-based framework for vehicle-based mobile sensing," *Procedia Computer Science*, vol. 10, pp. 1152–1157, 2012.
- [67] J.-S. Lee and B. Hoh, "Dynamic pricing incentive for participatory sensing," *Pervasive and Mobile Computing*, vol. 6, no. 6, pp. 693–708, 2010.
- [68] K. Lee, J. Caverlee, and S. Webb, "The social honeypot project: protecting online communities from spammers," in *WWW*, 2010, pp. 1139–1140.
- [69] J. Lehmann, C. Castillo, M. Lalmas, and E. Zuckerman, "Finding news curators in twitter," in *WWW(TheWebConf)*, 2013, pp. 863–870.
- [70] —, "Transient news crowds in social media." in *ICWSM*, 2013.
- [71] G. Li, J. Wang, Y. Zheng, and M. J. Franklin, "Crowdsourced data management: A survey," *TKDE*, vol. 28, no. 9, pp. 2296–2319, 2016.
- [72] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, and J. Han, "A survey on truth discovery," *ACM SIGKDD Explorations Newsletter*, vol. 17, no. 2, pp. 1–16, 2016.
- [73] Y. Li, M. L. Yiu, and W. Xu, "Oriented online route recommendation for spatial crowdsourcing task workers," in *Advances in Spatial and Temporal Databases*. Springer, 2015, pp. 137–156.
- [74] A. Liu, W. Wang, S. Shang, Q. Li, and X. Zhang, "Efficient task assignment in spatial crowdsourcing with worker and task privacy protection," *GeoInformatica*, pp. 1–28, 2017.
- [75] B. Liu, L. Chen, X. Zhu, Y. Zhang, and C. Zhang, "Protecting location privacy in spatial crowdsourcing using encrypted data," in *EDBT*, 2017, pp. 478–481.
- [76] X. Liu, M. Lu, B. C. Ooi, Y. Shen, S. Wu, and M. Zhang, "Cdas: a crowdsourcing data analytics system," *PVLDB*, vol. 5, no. 10, pp. 1040–1051, 2012.
- [77] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun, "Fog computing: Focusing on mobile users at the edge," *arXiv preprint arXiv:1502.01815*, 2015.
- [78] T. Luo, H.-P. Tan, and L. Xia, "Profit-maximizing incentive for participatory sensing," in *INFOCOM*, 2014.
- [79] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," in *ICDE*, 2006, p. 24.

## References

- [80] A. Marcus, A. Parameswaran *et al.*, “Crowdsourced data management: Industry and academic perspectives,” *Foundations and Trends® in Databases*, vol. 6, no. 1-2, pp. 1–161, 2015.
- [81] W. Mason and D. J. Watts, “Financial incentives and the performance of crowds,” *ACM SigKDD Explorations Newsletter*, vol. 11, no. 2, pp. 77–85, 2010.
- [82] J. M. Moore, “An  $n$  job, one machine sequencing algorithm for minimizing the number of late jobs,” *Management science*, vol. 15, no. 1, pp. 102–109, 1968.
- [83] A. Nandugudi, T. Ki, C. Nuessle, and G. Challen, “Pocketparker: pocketsourcing parking lot availability,” in *UbiComp*, 2014, pp. 963–973.
- [84] J. C. Navas and T. Imielinski, “Geocast—geographic addressing and routing,” in *MobiCom*, 1997, pp. 66–76.
- [85] R. W. Ouyang, M. Srivastava, A. Toniolo, and T. J. Norman, “Truth discovery in crowdsourced detection of spatial events,” *TKDE*, vol. 28, no. 4, pp. 1047–1060, 2016.
- [86] B. Pan, Y. Zheng, D. Wilkie, and C. Shahabi, “Crowd sensing of traffic anomalies based on human mobility and social media,” in *GIS*, 2013, pp. 344–353.
- [87] L. Pournajaf, L. Xiong, V. Sunderam, and S. Goryczka, “Spatial task assignment for crowd sensing with cloaked locations,” in *MDM*, 2014, pp. 73–82.
- [88] L. Pu, X. Chen, J. Xu, and X. Fu, “Crowdlet: Optimal worker recruitment for self-organized mobile crowdsourcing,” in *INFOCOM*, 2016, p. 5.
- [89] V. C. Raykar, S. Yu, L. H. Zhao, A. Jerebko, C. Florin, G. H. Valadez, L. Bogoni, and L. Moy, “Supervised learning from multiple experts: whom to trust when everyone lies a bit,” in *ICML*, 2009, pp. 889–896.
- [90] F. Restuccia, N. Ghosh, S. Bhattacharjee, S. K. Das, and T. Melodia, “Quality of information in mobile crowdsensing: Survey and research challenges,” *TOSN*, vol. 13, no. 4, p. 34, 2017.
- [91] H. Robbins, “Some aspects of the sequential design of experiments,” in *Herbert Robbins Selected Papers*. Springer, 1985, pp. 169–177.
- [92] J. Rogstadius, V. Kostakos, A. Kittur, B. Smus, J. Laredo, and M. Vukovic, “An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets,” in *ICWSM*, 2011, pp. 17–21.
- [93] M. A. Saleem, R. Kumar, T. Calders, X. Xie, and T. B. Pedersen, “Location influence in location-based social networks,” in *WSDM*, 2017, pp. 621–630.
- [94] A. Schrijver, *Combinatorial optimization: polyhedra and efficiency*. Springer Science & Business Media, 2002, vol. 24.
- [95] J. She, Y. Tong, and L. Chen, “Utility-aware social event-participant planning,” in *SIGMOD*. ACM, 2015, pp. 1629–1643.
- [96] Y. Shen, L. Huang, L. Li, X. Lu, S. Wang, and W. Yang, “Towards preserving worker location privacy in spatial crowdsourcing,” in *GLOBECOM*, 2015, pp. 1–6.

## References

- [97] M. Shin, C. Cornelius, D. Peebles, A. Kapadia, D. Kotz, and N. Triandopoulos, "Anonymsense: A system for anonymous opportunistic sensing," *Pervasive and Mobile Computing*, vol. 7, no. 1, pp. 16–30, 2011.
- [98] T. Song, Y. Tong, L. Wang, J. She, B. Yao, L. Chen, and K. Xu, "Trichromatic online matching in real-time spatial crowdsourcing," in *ICDE*, 2017, pp. 1009–1020.
- [99] M. Stevens and E. D'Hondt, "Crowdsourcing of pollution data using smart-phones," in *Workshop on Ubiquitous Crowdsourcing*, 2010.
- [100] B. L. Sullivan, C. L. Wood, M. J. Iliff, R. E. Bonney, D. Fink, and S. Kelling, "ebird: A citizen-based bird observation network in the biological sciences," *Biological Conservation*, vol. 142, no. 10, pp. 2282–2292, 2009.
- [101] D. Sun, K. Xu, H. Cheng, Y. Zhang, T. Song, R. Liu, and Y. Xu, "Online delivery route recommendation in spatial crowdsourcing," *WWW*, pp. 1–22, 2018.
- [102] H. Sun, B. Dong, B. Zhang, W. H. Wang, and M. Kantarcioglu, "Sensitive task assignments in crowdsourcing markets with colluding workers," in *ICDE*, 04 2018.
- [103] H. To, M. Asghari, D. Deng, and C. Shahabi, "Scawg: A toolbox for generating synthetic workload for spatial crowdsourcing," in *PerCom Workshop*, 2016, pp. 1–6.
- [104] H. To, L. Fan, L. Tran, and C. Shahabi, "Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints," in *PerCom*, 2016, pp. 1–8.
- [105] H. To, G. Ghinita, L. Fan, and C. Shahabi, "Differentially private location protection for worker datasets in spatial crowdsourcing," *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 934–949, 2017.
- [106] H. To, G. Ghinita, and C. Shahabi, "A framework for protecting worker location privacy in spatial crowdsourcing," *PVLDB*, vol. 7, no. 10, pp. 919–930, 2014.
- [107] —, "Privgeocrowd: A toolbox for studying private spatial crowdsourcing," in *ICDE*, 2015, pp. 1404–1407.
- [108] H. To and C. Shahabi, *Location Privacy in Spatial Crowdsourcing*. Springer International Publishing: In Handbook of Mobile Data Privacy, 2018, pp. 167–194. [Online]. Available: [https://doi.org/10.1007/978-3-319-98161-1\\_7](https://doi.org/10.1007/978-3-319-98161-1_7)
- [109] H. To, C. Shahabi, and L. Kazemi, "A server-assigned spatial crowdsourcing framework," *ACM Transactions on Spatial Algorithms and Systems*, vol. 1, no. 1, p. 2, 2015.
- [110] H. To, C. Shahabi, and L. Xiong, "Privacy-preserving online task assignment in spatial crowdsourcing with untrusted server," in *ICDE*, 2018.
- [111] Y. Tong, L. Chen, and C. Shahabi, "Spatial crowdsourcing: Challenges, techniques, and applications," *PVLDB[Tutorial]*, vol. 10, no. 12, pp. 1988–1991, 2017.
- [112] Y. Tong, J. She, B. Ding, L. Chen, T. Wo, and K. Xu, "Online minimum matching in real-time spatial data: experiments and analysis," *PVLDB*, vol. 9, no. 12, pp. 1053–1064, 2016.

## References

- [113] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen, "Online mobile micro-task allocation in spatial crowdsourcing," in *ICDE*, 2016, pp. 49–60.
- [114] Y. Tong, L. Wang, Z. Zhou, B. Ding, L. Chen, J. Ye, and K. Xu, "Flexible online task assignment in real-time spatial data," *PVLDB*, vol. 10, no. 11, pp. 1334–1345, 2017.
- [115] L. Tran, H. To, L. Fan, and C. Shahabi, "A real-time framework for task assignment in hyperlocal spatial crowdsourcing," *ACM Transactions on Intelligent Systems and Technology*, p. 37, 2017.
- [116] U. ul Hassan and E. Curry, "Flag-verify-fix: adaptive spatial crowdsourcing leveraging location-based social networks," in *GIS*.
- [117] H. Väättäjä, T. Vainio, and E. Sirkkunen, "Location-based crowdsourcing of hyperlocal news: dimensions of participation preferences," in *GROUP*, 2012, pp. 85–94.
- [118] B. Van der Haak, M. Parks, and M. Castells, "The future of journalism: Networked journalism," *Int. Journal of Communication*, vol. 6, p. 16, 2012.
- [119] M. Vasardani, S. Winter, K.-F. Richter, L. Stirling, and D. Richter, "Spatial interpretations of preposition "at"," in *GIS GEOCROWD*, 2012, pp. 46–53.
- [120] G. Wang, B. Wang, T. Wang, A. Nika, H. Zheng, and B. Y. Zhao, "Defending against sybil devices in crowdsourced mapping services," in *MobiSys*, 2016, pp. 179–191.
- [121] L. Wang, Z. Yu, Q. Han, B. Guo, and H. Xiong, "Multi-objective optimization based allocation of heterogeneous spatial crowdsourcing tasks," *IEEE Transactions on Mobile Computing*, pp. 1637–1650, 2018.
- [122] J. Whitehill, T.-f. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo, "Whose vote should count more: Optimal integration of labels from labelers of unknown expertise," in *Advances in neural information processing systems*, 2009, pp. 2035–2043.
- [123] R. C.-W. Wong, Y. Tao, A. W.-C. Fu, and X. Xiao, "On efficient spatial matching," in *PVLDB*. PVLDB, 2007, pp. 579–590.
- [124] M. Xiao, J. Wu, L. Huang, Y. Wang, and C. Liu, "Multi-task assignment for crowdsensing in mobile social networks," in *INFOCOM*, 2015, pp. 2227–2235.
- [125] M. Xiao, J. Wu, S. Zhang, and J. Yu, "Secret-sharing-based secure user recruitment protocol for mobile crowdsensing," in *INFOCOM*, 2017.
- [126] X. Xie, H. Chen, and H. Wu, "Bargain-based stimulation mechanism for selfish mobile nodes in participatory sensing network," in *SECON*, 2009, pp. 1–9.
- [127] T. Yan, M. Marzilli, R. Holmes, D. Ganesan, and M. Corner, "mcrowd: a platform for mobile crowdsourcing," in *SenSys*, 2009, pp. 347–348.
- [128] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing," in *MobiCom*, 2012, pp. 173–184.
- [129] P. Yang, N. Zhang, S. Zhang, K. Yang, L. Yu, and X. Shen, "Identifying the most valuable workers in fog-assisted spatial crowdsourcing," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1193–1203, 2017.

## References

- [130] M.-C. Yuen, I. King, and K.-S. Leung, "A survey of crowdsourcing systems," in *SocialCom*, 2011, pp. 766–773.
- [131] Y. Zeng, Y. Tong, L. Chen, and Z. Zhou, "Latency-oriented task completion via spatial crowdsourcing," in *ICDE*, 2018.
- [132] B. Zhang, C. H. Liu, J. Lu, Z. Song, Z. Ren, J. Ma, and W. Wang, "Privacy-preserving qoi-aware participant coordination for mobile crowdsourcing," *Computer Networks*, pp. 29–41, 2016.
- [133] L. Zhang, X. Lu, P. Xiong, and T. Zhu, "A differentially private method for reward-based spatial crowdsourcing," in *ATIS*, 2015, pp. 153–164.
- [134] X. Zhang, Z. Yang, Y.-J. Gong, Y. Liu, and S. Tang, "Spatialrecruiter: maximizing sensing coverage in selecting workers for spatial crowdsourcing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 5229–5240, 2017.
- [135] X. Zhang, Z. Yang, Y. Liu, and S. Tang, "On reliable task assignment for spatial crowdsourcing," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2016.
- [136] Y. Zhao, Y. Li, Y. Wang, H. Su, and K. Zheng, "Destination-aware task assignment in spatial crowdsourcing," in *CIKM*, 2017, pp. 297–306.
- [137] Y. Zhao and Q. Han, "Spatial crowdsourcing: current state and future directions," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 102–107, 2016.
- [138] L. Zheng and L. Chen, "Mutual benefit aware task assignment in a bipartite labor market," in *ICDE*, 2016, pp. 73–84.
- [139] G. Zhuo, Q. Jia, L. Guo, M. Li, and P. Li, "Privacy-preserving verifiable set operation in big data for cloud-assisted mobile crowdsourcing," *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 572–582, 2017.
- [140] M. Zook, M. Graham, T. Shelton, and S. Gorman, "Volunteered geographic information and crowdsourcing disaster relief: a case study of the haitian earthquake," *World Medical & Health Policy*, vol. 2, no. 2, pp. 7–33, 2010.

## References

# Paper B

## Push-based Spatial Crowdsourcing for Enriching Semantic Tags in OpenStreetMap

Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach  
Pedersen, Xike Xie, and Esteban Zimányi

The paper is in: *preparation for submission to Geoinformatica.*

## Abstract

*Our paper aims to improve the quantity and quality of the tags associated with the OpenStreetMap (OSM) through the push-based spatial crowdsourcing (SC) approach. To understand the applicability of push-based SC approach for OSM contributors/ workers, we study different task assignment problems based on three optimization objectives: 1) to maximize the total number of task assignments, 2) to maximize the coverage of the entities, i.e., more unique entities assigned to workers, 3) to maximize the number of verifiable task assignments (multiple workers assigned to the task). To solve the different task assignment problems, we have proposed an integrated push-based SC framework for OSM, that regularly extracts the OSM spatial entities, assigns the entities to workers and updates the verified tags collected from the workers to OSM. We validate the proposed push-based SC framework for OSM by focusing on the use case of road networks in OSM. The integrated framework comprises of four modules, namely, Task Generator, Task Assignment, Quality Control, and Updating OSM Database. The Task Generator module extracts the OSM spatial entities at regular intervals and forwards them as tasks to the Task Assignment module. The Task Assignment module assigns the tasks to workers and forwards the collected crowdsourced responses to the Quality Control module. The crowdsourced responses will be aggregated using noniterative aggregation techniques like Majority Voting in the Quality Control module to identify the verifiable truth, i.e., two or more workers responding with similar tags for the OSM entity. The verified tags and their spatial entities will be updated to the OSM database at regular intervals by the Updating OSM Database module. The proposed junctions-based algorithms result in at least five times as many total and unique assignments, seven times as many verifiable assignments as the baseline. Similarly, the junctions-based algorithms result in half the average distance travelled per task than the baseline, max-flow based algorithm.*

*The layout has been revised.*



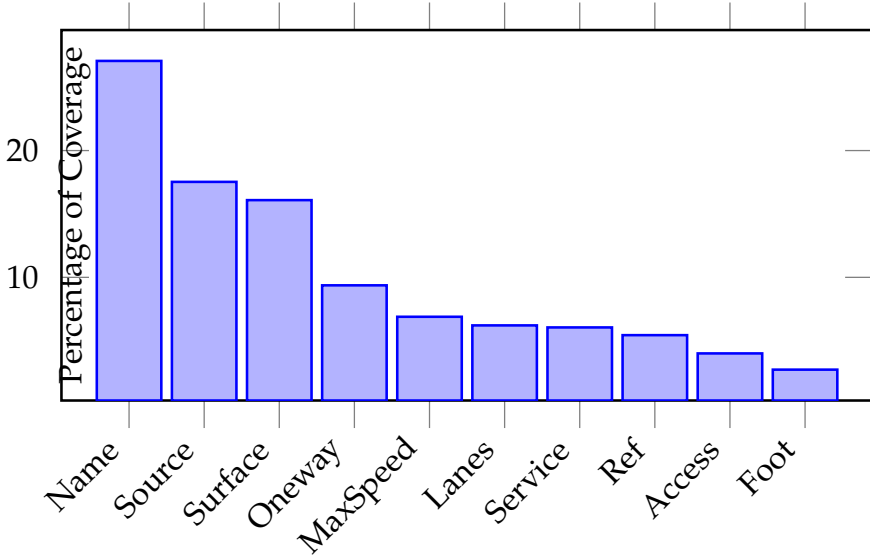


Fig. B.1: OSM Tag Coverage for Road Segments

## 1 Introduction

OpenStreetMap’s (OSM) data quality is hindered by the lack of adequate tags associated with the spatial entities; for example, the top three tags related to the road network are available for just around 10% of the road segments in OSM (See Fig. B.1). Furthermore, conventional OSM voluntary contributions are user-initiated, which might result in vandalism or inefficient mapping. Furthermore, a user without local knowledge might be mapping the area, resulting in a small number of tags associated with the OSM spatial entity. Consequently, there is a need to actively push the contributors to enrich the OSM database with relevant tags.

A previous short paper [7] was the first to investigate the issue and proposed employing spatial crowdsourcing (SC) to collect relevant tags but only provided simplistic problem definition and basic assignment strategies. In comparison, this paper facilitates workers to maximize their contribution by maximizing their total road segment assignments, unique road segment task assignments, and verifiable road segment task assignments. Furthermore, this paper proposes a comprehensive framework that encompasses the end-to-end process of enriching OSM semantic tags. Additionally, this paper studies the problem of expanding assignment coverage for the OSM entities, by focusing on assigning as many unique entities to workers as

possible. Consequently, the possibility of duplicate or redundant assignments gets reduced, resulting in efficient utilization of a limited resource, i.e., workers. Furthermore, this paper facilitates the OSM concept of verifiability [32] by studying an additional problem of assigning at least two workers to the OSM entity, as ensuring verifiability requires independent workers making the same observation regarding an assigned OSM entity. The proposed framework supports the standard spatial crowdsourcing tasks like point tasks [9, 13], area tasks [25], and delivery tasks [24]. In addition, the proposed framework also supports the case of OSM road segments, where the worker is allowed to perform the task at any place on the road segment.

We study the applicability of push-based SC approach by focusing on three important optimization objectives of the OSM task assignment problems, like maximizing the total number of road segment assignments, maximizing the coverage of assignments (maximizing number of unique road segment task assignments), and maximizing the verifiable task assignments (a task assignment is considered verifiable if it is assigned to two or more workers). With more number of unique task assignments more number of OSM tags belonging to different OSM entities can be collected and with more number of verifiable task assignments more number of correct tags can be collected, therefore the considered optimizations would improve the quantity and quality of the collected OSM tags.

To solve the three task assignment problems studied in the paper, we propose an advanced push-based SC framework for OSM and validate it by focusing on the use case of road networks in OSM. We propose a framework with four modules, namely, *Task Generator*, *Task Assignment*, *Quality Control*, and *Updating OSM Database*. The *Task Generator* module extracts the OSM spatial entities at regular intervals and forwards them as tasks to the *Task Assignment* module. The *Task Assignment* module assigns the tasks to workers and forwards the collected crowdsourced responses to the *Quality Control* module. The crowdsourced responses will be aggregated using noniterative aggregation techniques like *Majority Voting* in the *Quality Control* module to identify the verifiable truth. The verified tags, along with their spatial entities, will be updated to the OSM database at regular intervals by the *Updating OSM Database* module.

**In this paper, we limit our focus on the task generator and task assignment modules of the framework.** Especially, we study the algorithmic techniques needed by the task assignment module for scalable and effective task assignment. For the quality control module, we are going to reuse the existing SC data aggregation techniques like *majority voting* [17].

The main contributions we offer in this paper are:

- We study and define the optimization problems of maximizing the total number of assignments, maximizing the number of unique assign-

## 2. Preliminaries and Problem Definition

ments, and maximizing the number of verifiable assignments for improving the quality and quantity of OSM tags through push-based SC approach.

- We address the studied task assignment problems by proposing a framework for push-based SC for OSM with four modules, namely, *Task Generator*, *Task Assignment*, *Quality Control*, and *Updating OSM Database*.
- Focusing on the use case of road networks in OSM, we propose different task assignment algorithms to assign road segments/ junctions to workers based on different constraints in both offline and batch-based worker input model scenarios.
- We test the applicability of our proposed methods through an extensive experimental evaluation based on extracted OSM road segments from Aalborg, Denmark.

The remainder of the paper is organized as follows. In Section 2, we define the preliminaries and the three task assignment problems based on different optimization objectives. In Section 3, we present the solution, push-based SC for OSM framework, to the defined problems and discuss the different modules of the framework briefly. Sections 4, 5, 6, and 7 are discussed with a focus on the use case of OSM road networks. In Section 4, we present the algorithms proposed to solve the optimization problem for maximizing the total number of road segment task assignments. Similarly, in Sections 5 and 6 we present the algorithms proposed to solve the optimization problems for maximizing the unique and verifiable road segment task assignments, respectively. Section 7 presents the experimental evaluation of the proposed task assignment methods in Sections 4, 5, and 6. Section 8 discusses the related work in the SC literature. Finally, Section 9 summarizes the proposed push-based SC for OSM framework and the findings regarding our proposed approaches and discusses future work.

## 2 Preliminaries and Problem Definition

### 2.1 Preliminaries

This section introduces some basic concepts that will be used throughout this paper.

**Road Segment task:** [7] A road segment task  $rst$  contains a line segment with  $m$  nodes. The start node and end node of the line segment are located at  $l_1$  and  $l_m$ , respectively. The road segment task  $rst$  has information regarding the  $n$  tags ( $tag_n$ ) along with their values ( $val_n$ ). The total number of tags  $n$  will be at least 27, as we would like to retrieve the information about the 27

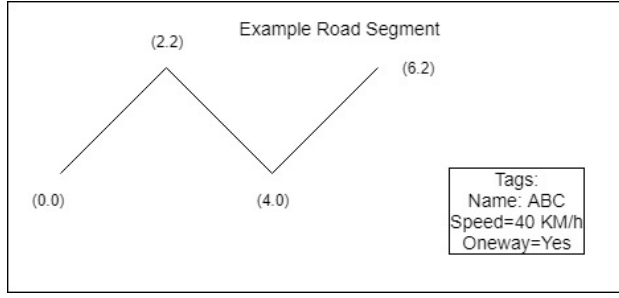


Fig. B.2: Example Road Segment [7]

standard tags, along with the existing custom tags. The road segment task  $rst$  has no temporal constraints associated with it, owing to the dynamic nature of the road network. However, the assignment of the task is tracked by the attribute *assigned*.

$$rst = \langle \langle tag_1, val_1 \rangle, \langle tag_2, val_2 \rangle, \dots, \langle tag_n, val_n \rangle, \langle l_1, l_2, \dots, l_m \rangle, assigned \rangle$$

Consider the example in Fig. B.2, the location set of the road segment  $\langle (0,0), (2,2), (4,0), (6,2) \rangle$  is mapped to node locations  $\langle l_1, l_2, l_3, l_4 \rangle$ . The tag-set of the road segment will be mapped to the respective tags of the road segment task and the tags that are not present in the road segment will be marked as empty, i.e.,  $\langle \langle Speed = "40Km/h" \rangle, \langle Name = "ABC" \rangle, \langle Oneway = "Yes" \rangle, \langle Lanes = "" \rangle, \langle Surface = "" \rangle, \dots \rangle$ .

**Worker:** [7] A worker, denoted by  $w$ , is an amateur cartographer willing to perform an assigned task by travelling to the task's road segment. Worker  $w$  has an identification number  $wid$  along with a location  $l$  that she reports to the assignment module. Additionally, Worker  $w$  specifies her preferences for accepting a task like the maximum number of tasks she's willing to perform,  $maxT$  and the maximum distance  $d$  she is willing to travel for performing a task. A worker is defined as:  $w = \langle wid, l, maxT, d \rangle$

Typically, a road segment on a road network could be visited at a junction for retrieving the tag information. Therefore, we create junction tasks to facilitate performing tasks related to road segments intersecting at the junction (See Fig. B.3b). We formally define Junction tasks and their cost function as follows.

**Junction task  $j$ :** [7] A junction task  $j$  is an intersecting point of two or more road segment tasks.  $j = \langle jid, l_j, \langle rst_1, rst_2, \dots \rangle \rangle$ , where  $jid$  is the junction id and  $\langle rst_1, rst_2, \dots \rangle$  are the list of road segment tasks meeting at location  $l_j$  of the junction  $j$ .

**Cost Function of a Junction task:** [7] The cost function of a junction task  $c(j, w)$  is:  $c(j, w) = dist(j, w)$ , where  $dist(j, w)$  is the distance between the junction task's location and the worker's location.

## 2. Preliminaries and Problem Definition

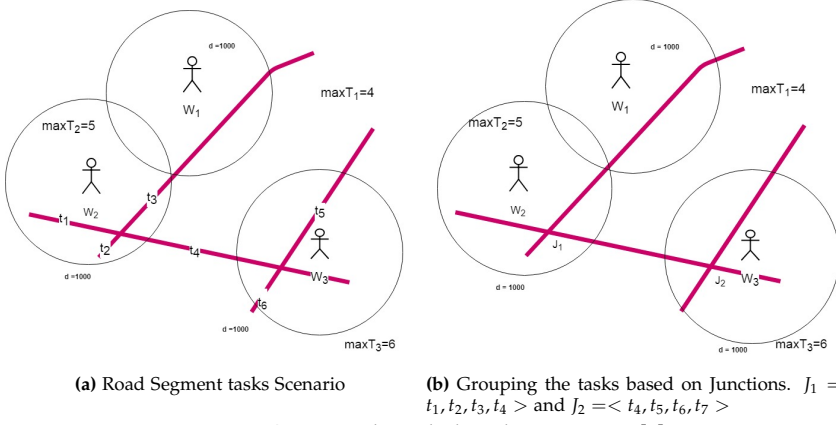


Fig. B.3: Grouping the tasks based on Junctions [7].

## 2.2 Problem Definition

Given the constraints and different objectives, we define the following types of task assignment problems according to the optimization objectives: maximizing total number of task assignments, maximizing the number of unique task assignments, and maximizing the number of verifiable task assignments.

### Maximizing Total Number of Assignments

In line with the objective of enriching the OSM database, we have to define our problem to add as many tag values for the different road segment tasks  $rst \in RST$  as possible. Therefore, the total number of road segment tasks assigned to workers should be maximized to increase the possibility of obtaining more tags for the road segments, considering workers' constraints. A task assignment to a worker is considered valid if and only if all the constraints are satisfied. In accordance, we define the valid task assignment set as follows:

**Valid Task Assignment Set:** [7] Given a set of workers  $W = \{w_1, w_2, \dots\}$  and a set of road segment tasks  $RST = \{rst_1, rst_2, \dots\}$ , a *Valid Task Assignment Set*, denoted by  $VA$ , is a set of assignments of the form  $\langle w, rst \rangle$ , in which a road segment task  $rst$  is assigned to a worker  $w$ , while satisfying the following constraints:

- **Maximum Tasks constraint:** The worker  $w$  should not be assigned more road segment tasks than the  $maxT$  value.
- **Distance Constraint:** The worker  $w$  should not be assigned tasks that are farther than  $d$ , i.e.,  $dist(w, rst) \leq d$

The valid task assignment set constraints give rise to our optimization problem, *Maximum Road Segment Task Assignment problem (MRSTA)* [7]. The inputs for the MRSTA problem are the set of workers  $W$  and the set of road segment tasks  $RST$ . The set of road segment tasks stay constant, as the road segment tag information requires frequent assignments to update the associated tag details. However, the set of workers would vary as the workers might not be online/ available all the time. For a better understanding of the MRSTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). The definition of the offline-MRSTA problem is given below:

**Offline Maximum Road Segment Task Assignment:** [7] Assume an input set of road segment tasks  $RST$ , a set of available workers  $W$ , and a set of valid task assignments  $VA$ . The offline Maximum Road Segment Task Assignment (Offline-MRSTA) problem is an optimization problem with the goal of maximizing the total number of valid road segment task assignments.

$$OffMR(RST, W) = \arg \max_{VA \in 2^{RST \times W} \text{ s.t. } VA \text{ is valid}} (|VA|)$$

, where  $OffMR(RST, W)$  is the maximal number of assignments.

**Batch-based Maximum Road Segment Task Assignment:** [7] The Batch-based Maximum Road Segment Task Assignment problem (Batch-MRSTA) is an optimization problem with a goal of maximizing the total number of valid road segment task assignments for continuous batches of workers  $W_i$  at each time instance  $i$ . At each time instance, the assignment problem for the batch of workers, along with the road segment tasks can be solved by reducing it to the Offline-MRSTA problem. Thus, the Batch-MRSTA can be solved by a series of Offline-MRSTA problems at each time instance.

$$OnMR(RST, W) = \sum_{i \in T} OffMR(RST, W_i)$$

, where and  $OnMR(RST, W)$  is the optimal number of assignments and  $W_i$  is the set of workers at time instance  $i$ . We assume, that the workers' arrive in batches at each time instance.

### Maximizing Unique Assignments

In line with the objective of enriching the OSM database, we have to define our problem to add tag values for as many different road segment tasks  $rst \in RST$  as possible. Therefore, the total number of unique road segment tasks assigned to workers should be maximized to increase the possibility of obtaining more tags for the road segments, considering workers' constraints. We define the unique task assignment set as follows:

## 2. Preliminaries and Problem Definition

**Unique Task Assignment Set:** Given a set of workers  $W = \{w_1, w_2, \dots\}$  and a set of road segment tasks  $RST = \{rst_1, rst_2, \dots\}$ , a *Unique Task Assignment Set*, denoted by  $UA$ , is a set of unique assignments of the form  $\langle w, rst \rangle$ , in which a road segment task  $rst$  is assigned to a worker  $w$ , while satisfying the following constraints:

- Unassigned constraint: The task  $rst$  should be unassigned, i.e.,  $rst.assigned = 0$
- Maximum Tasks constraint: The worker  $w$  should be assigned to atmost  $maxT$  road segment tasks.
- Distance Constraint: The worker  $w$  should be assigned tasks that are within the  $d$ , i.e.,  $dist(w, rst) \leq d$

The unique task assignment constraints give rise to our optimization problem, *Maximum Unique Road Segment Task Assignment problem (MURSTA)*. The inputs for the MURSTA problem are the set of workers  $W$  and the set of road segment tasks  $RST$ . However, the set of workers would vary as the workers might not be online/ available all the time. For a better understanding of the MURSTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). The definition of the offline-MURSTA problem is given below:

**Offline Maximum Unique Road Segment Task Assignment:** Assume an input set of road segment tasks  $RST$ , a set of available workers  $W$ , and a set of unique task assignments  $UA$ . The offline Maximum Unique Road Segment Task Assignment (Offline-MURSTA) problem is an optimization problem with the goal of maximizing the total number of unique road segment task assignments.

$$OffMR(RST, W) = \arg \max_{UA \in 2^{RST \times W} \text{ and } UA \text{ is unique}} (UA)$$

, where  $OffMR(RST, W)$  is the set of maximal unique assignments.

**Batch-based Maximum Unique Road Segment Task Assignment:** The Batch-based Maximum Unique Road Segment Task Assignment problem (Batch-MURSTA) is an optimization problem with a goal of maximizing the total number of unique road segment task assignments for continuous batches of workers  $W_i$  at each time instance  $i$ . At each time instance, the assignment problem for the batch of workers, along with the updated road segment tasks  $RST_i$  can be solved by reducing it to the Offline-MURSTA problem. Thus, the Batch-MURSTA can be solved by a series of Offline-MURSTA problems at each time instance.

$$OnMR(RST, W) = \sum_{i \in T} OffMR(RST_i, W_i)$$

, where  $OnMR(RST, W)$  is the set of maximal unique assignments,  $W_i$  is the set of workers, and  $RST_i$  is the set of updated road segment tasks at time instance  $i$ . We assume, that the workers' arrive in batches at each time instance. After every batch, the road segment task set will be updated to reflect the assignments.

### Maximizing Verifiable Assignments

As mentioned earlier, verifiability [32] is a critical concept in OSM. To facilitate verifiability in the *Quality Control* module of our framework, we try to assign at least two workers for a single road segment task, such assignments are defined as *Verifiable Task Assignments*.

However, there might be cases where a road segment is not accessible by multiple workers owing to the distance and  $maxT$  constraints. Therefore, to ensure the facilitation of verifiability, we try to maximize the number of verifiable road segment tasks. We define the verifiable task assignment set as follows:

**Verifiable Task Assignment Set:** Given a set of workers  $W = \{w_1, w_2, \dots\}$  and a set of road segment tasks  $RST = \{rst_1, rst_2, \dots\}$ , a *Verifiable Task Assignment Set*, denoted by  $VA$ , is a set of unique assignments of the form  $\langle w, rst \rangle$ , in which a road segment task  $rst$  is assigned to a worker  $w$ , while satisfying the following constraints:

- Multiple Assignment constraint: The task  $rst$  should be assigned to atleast one worker before, i.e.,  $rst.assigned \geq 1$
- Maximum Tasks constraint: The worker  $w$  should be assigned to atmost  $maxT$  road segment tasks.
- Distance Constraint: The worker  $w$  should be assigned tasks that are within the  $d$ , i.e.,  $dist(w, rst) \leq d$

To ensure the maximization of the verifiable road segment task assignments, we track the assignments of road segments and prioritize the tasks that were only assigned, one worker. These issues give rise to our optimization problem, *Maximum Verifiable Road Segment Task Assignment problem (MVRSTA)*. The inputs for the MVRSTA problem are the set of workers  $W$  and the set of road segment tasks  $RST$ . However, the set of workers would vary as the workers might not be online/ available all the time. For a better understanding of the MVRSTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). The definition of the offline-MVRSTA problem is given below:

**Offline Maximum Verifiable Road Segment Task Assignment:** Assume an input set of road segment tasks  $RST$ , a set of available workers  $W$ , and a set



### 3. Push-based SC for OSM Framework

of verifiable task assignments  $VA$ . The offline Maximum Verifiable Road Segment Task Assignment (Offline-MVRSTA) problem is an optimization problem with the goal of maximizing the total number of verifiable road segment task assignments.

$$OffMRV(RST, W) = \arg \max_{VA \in 2^{RST \times W} \text{ and } VA \text{ is verifiable}} (VA)$$

, where  $OffMRV(RST, W)$  is the set of maximal verifiable assignments.

**Batch-based Maximum Verifiable Road Segment Task Assignment:** The Batch-based Maximum Verifiable Road Segment Task Assignment problem (Batch-MVRSTA) is an optimization problem with a goal of maximizing the total number of verifiable road segment task assignments for continuous batches of workers  $W_i$  at each time instance  $i$ . At each time instance, the assignment problem for the batch of workers, along with the updated road segment tasks  $RST_i$  can be solved by reducing it to the Offline-MVRSTA problem. Thus, the Batch-MVRSTA can be solved by a series of Offline-MVRSTA problems at each time instance.

$$OnMRV(RST, W) = \sum_{i \in T} OffMRV(RST_i, W_i)$$

, where and  $OnMRV(RST, W)$  is the set of maximal verifiable assignments,  $W_i$  is the set of workers, and  $RST_i$  is the set of updated road segment tasks at time instance  $i$ . We assume, that the workers' arrive in batches at each time instance. After every batch, the road segment task set will be updated to reflect the assignments.

## 3 Push-based SC for OSM Framework

To address the defined task assignment problems in Section 2, we propose a push-based SC for OSM framework. The architecture of the proposed push-based SC for OSM framework is illustrated in Fig. B.4. The proposed framework contains four modules: the Task Generator module, the Task Assignment module, the Quality Control module, and the Updating OSM Database module. The framework enables communication with two external stakeholders: OSM database for extracting the entities and updating the tags; and the spatial crowdsourcing workers.

### 3.1 Task Generator Module:

The Task Generator module extracts spatial entities from the OSM database and generates tasks for each spatial entity. For example, building entities

can be extracted as building tasks whose information can be collected at the building location. The task generator module sends the extracted tasks to the task assignment module. The task generator module is designed to re-query the spatial entities from the OSM database in a periodic manner; for example, the tasks will be extracted once every year for updating the tags.

### Extraction of Road Segments

OSM offers support to extract the spatial entities based on certain tag types. We extract the road segments from the OSM database, which has the tag “highways”. For example, we have extracted the road segments with the tag “highways” associated with the geographical area around the city of Aalborg, Denmark (as seen in Fig. B.5).

The extracted OSM dataset of road segments of Aalborg, Denmark has 17601 road segments with 80906 nodes (See Fig. B.5). The task generator module creates a set of road segment tasks *RST* based on the extracted road segments.

## 3.2 Task Assignment Module:

The Task Assignment module objective is to effectively assign workers to the tasks received from the task generator module. The task assignment module collects the current location information of the workers regularly to assign tasks that are in the vicinity of the workers. Furthermore, the tasks are static in nature, contrasting the dynamic nature of workers, i.e., in a given time interval, the tasks stay the same, whereas the number of available workers varies. The task assignment module follows the batch-based assignment pro-

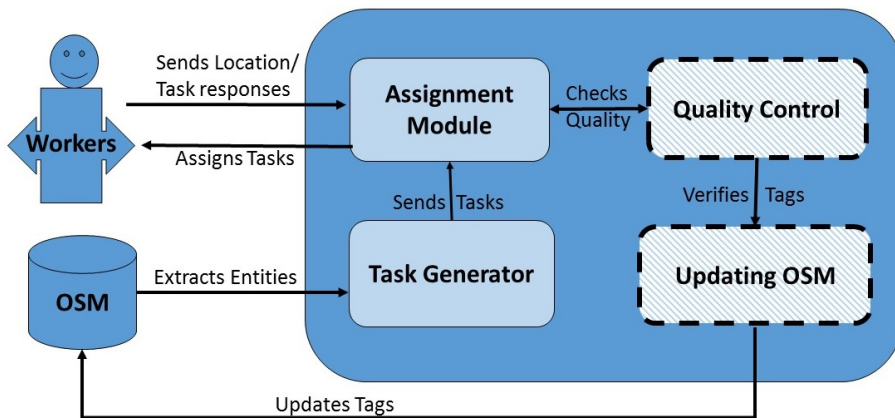


Fig. B.4: Push-based SC Framework for OSM

### 3. Push-based SC for OSM Framework

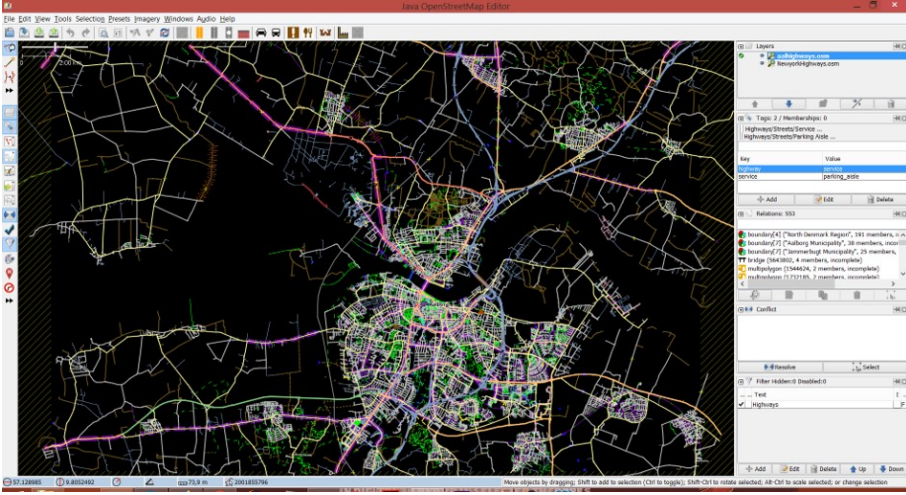


Fig. B.5: OSM Tag Coverage in Aalborg, Denmark

cess that conducts the assignment of workers in batches, instead of performing an assignment as soon as a worker is available. The tasks can be assigned to multiple workers, and each worker can be assigned a maximum threshold of tasks to perform. Once the tasks are assigned to the workers, the workers perform the tasks and send the tags for the tasks back to the task assignment module. The task assignment collects the responses sent by the workers and forwards it to the quality control module. The task assignment module will be discussed in detail in the following sections by focusing on the OSM road networks use case scenario.

### 3.3 Quality Control Module:

The Quality Control module aggregates the collected crowdsourced responses (tag values) from the task assignment module to infer the true tag values. Verifiability [32] is very important in OSM, and it can be achieved “if and only if independent users observing the same feature would make the same observation every time”. The quality control module utilizes the wisdom of the crowd through the noniterative aggregation method, Majority Voting [17], to infer the verifiable truth from the collected crowdsourced responses. As mentioned previously, the task assignment module assigns the same task to multiple workers, and the values of the tags for the task that the majority agrees with would be considered as the verifiable truth. Other noniterative aggregation methods can be used instead of Majority Voting [17], like Honeypot [18], and Expert Label Injected Crowd Estimation(ELICE) [15]. Furthermore, we can conduct qualification tests for the workers to ascertain their expertise in providing tag values to OSM spatial entities.

### 3.4 Updating OSM Database Module:

Once the tags are verified from the quality control module, the Updating OSM Database module aggregates the verified tags and initiates the bulk import process for OSM [31]. The module needs to first discuss the import details with the OSM import community and the local community that gets their information updated. After considering the feedback from the communities, the module performs the import to the OSM database considering the server resources limitations. The updates will be processed in small batches to avoid overload on the server and quicker updates.

### 3.5 Use case: Road Networks

To demonstrate the applicability of the push-based SC for OSM framework for enriching tags in OSM, we have focused on the use case of road networks. As discussed in Section 1, the associated tag information for OSM road networks is not adequate. We will apply our approach to enrich the tags associated with road segments in OSM. We present two types of tasks to transform road network entities to spatial crowdsourcing tasks: *Road Segment Tasks* and *Junction Tasks*. *Road Segment Tasks* are associated with individual road segments in the road network. Whereas, *Junction tasks* are associated with the junctions where the road segments intersect.

## 4 Optimization Objective: Maximizing Total Valid Task Assignments

This section describes in detail the proposed assignment algorithms for the MRSTA problem. The offline-MRSTA problem can be solved by utilizing the existing solutions [13] for reducing it to a max-flow problem. We build our solutions to the Batch-MRSTA problem by employing some of the existing solutions, proposed in [13] as subroutines. In order to solve the Batch-MRSTA problem, we propose three methods, namely: *Max Flow-based Task Grouping*, *Direct Assignment with Road Segments*, and *Direct Assignment with Junctions* [7]. The proposed methods follow a locally optimal assignment strategy as the SC-server has no knowledge about the arrival times of the new workers [13]. Therefore, at every instance of time, the SC-server tries to assign the available tasks to the available workers.

### 4.1 Max Flow-based Task Grouping (TG)

The offline-MRSTA problem can be reduced to a maximum flow problem [7]. Therefore, we can use any algorithm that can compute the maximum flow in

#### 4. Optimization Objective: Maximizing Total Valid Task Assignments

the network to solve the offline-MRSTA problem, like the Ford-Fulkerson algorithm [16]. We can solve the batch-MRSTA problem by solving the offline-MRSTA problem at each time instance. However, this method does not consider the inherent nature of road networks for assigning the tasks. Consider the example in Fig. B.3a, there are three workers and seven road segment tasks in the example. With the max flow-based task grouping (TG) method (See Algorithm 13), the worker  $w_2$  can go to the junction where the four road segments  $(t_1, t_2, t_3, t_4)$  meet ( $J_1$ ), to perform the four tasks, instead of visiting individual road segment tasks. Thus, the task grouping method can achieve the objective of assigning seven road segment tasks by assigning workers to only two junction tasks, ( $J_1$  and  $J_2$ ). As only worker  $w_2$  can reach  $J_1$ , it will be assigned to her, similarly  $J_2$  for  $w_3$  (See Fig. B.3b). Once the tasks are grouped at different junctions, the assignment problem can be solved by following the solution for the minimum-cost maximum task assignment problem [13]. The total number of assignments of road segment tasks can be extrapolated from the number of junction tasks assigned.

---

##### ALGORITHM 13: MAX FLOW-BASED TASK GROUPING (TG) [7]

---

**Input:** A set of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of junction tasks  $J$  at time instance  $i$ .  
 $OnMR(J, W)$  is the optimal set of assignments before the time instance  $i$ .

**Output:** The Optimal set of assignments  $OnMR(J, W + W_i)$  at time instance  $i$

```

1  $OnMR(J, W + W_i) \leftarrow NULL$ 
2  $capacity \leftarrow 1$ 
3 if  $W_i \neq \emptyset$  then
4    $V \leftarrow W_i \cup J$ 
5   foreach  $w \in W_i$  do
6      $E.addEdge(V_0, w, maxT, 0)$ 
7     foreach  $j \in J$  do
8        $E.addEdge(j, V_{|V|+1}, capacity, 0)$ 
9       if  $dist(j, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
10         $E.addEdge(w, j, capacity, dist(j, w))$ 
11   construct flow network graph  $G(V, E)$ 
12   calculate the min travel cost maximum flow
13   find the assignment  $OnMR(J, W_i)$ 
14   Update  $maxT$  values of  $W_i$ 
15    $OnMR(J, W + W_i) \leftarrow OnMR(J, W) \cup OnMR(J, W_i)$ 
16 return  $OnMR(J, W + W_i)$ 

```

---

## 4.2 Direct Assignment with Road Segments (DA-RS)

Apart from the max flow-based task grouping method mentioned above, we have considered the direct method of assigning the  $maxT$  nearest road seg-

ment tasks to every worker, that is closer than the threshold *distance* value. Following this heuristic method, we propose *Direct Assignment with Road Segments (DA-RS)* [7] (See Algorithm 14), which will be searching for the nearest *maxT* road segment tasks for each worker in the current batch at time instance *i* and assign them to the workers if they satisfy the distance constraint.

---

**ALGORITHM 14: DIRECT ASSIGNMENT WITH ROAD SEGMENTS [7]**


---

**Input:** A non-empty set of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of road segment tasks  $RST_i$  at time instance *i*.  $OnMR(RST, W)$  is the optimal set of assignments before the time instance *i*

**Output:** The Optimal set of assignments  $OnMR(RST, W + W_i)$  at time instance *i*

```

1  $OnMR(RST, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3   foreach  $rst \in RST$  do
4     if  $dist(rst, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
5        $add\ assignment\ < w, rst > to OnMR(RST, W_i)$ 
6        $MaxT(w) \leftarrow MaxT(w) - 1$ 
7    $OnMR(RST, W + W_i) \leftarrow OnMR(RST, W) \cup OnMR(RST, W_i)$ 
8 return  $OnMR(RST, W + W_i)$ 

```

---

### 4.3 Direct Assignment with Junctions (DA-J)

Similar to the DA-RS method, we have considered the direct method of assigning the *maxT* nearest junction tasks to every worker, that is closer than the threshold *distance* value. Following this heuristic-based method, we propose *Direct Assignment with Junctions (DA-J)* [7]. This heuristic-based method will be searching for nearest *maxT* junction tasks for each worker in the current batch and assign them to the workers if they satisfy the distance constraint.

## 5 Optimization Objective: Maximizing Unique Task Assignments

This section describes in detail the proposed assignment algorithms for the MURSTA problem. In order to solve the Batch-MURSTA problem, we propose two methods, namely: *Unique Assignments with Road Segments (U-RS)*, and *Unique Assignments with Junctions (U-J)*.

## 5. Optimization Objective: Maximizing Unique Task Assignments

---

### ALGORITHM 15: UNIQUE ASSIGNMENTS WITH ROAD SEGMENTS (U-RS)

---

**Input:** A non-empty set of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of road segment tasks  $RST$  at time instance  $i$ .  $OnMR(RST, W)$  is the optimal set of unique assignments before the time instance  $i$

**Output:** The Optimal set of unique assignments  $OnMR(RST_i, W + W_i)$  at time instance  $i$

```

1  $OnMR(RST, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3    $RST_w \leftarrow \emptyset$ 
4   foreach  $rst \in RST$  do
5     if  $dist(rst, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
6        $RST_w \leftarrow RST_w \cup rst$ 
7    $sortByAssignedAscending(RST_w)$ 
8   foreach  $rst1 \in RST_w$  do
9      $rst1.assigned++$ 
10     $RST_i \leftarrow UpdateAssignment(RST)$ 
11     $MaxT(w) \leftarrow MaxT(w) - 1$ 
12     $OnMR(RST_i, W + W_i) \leftarrow OnMR(RST, W) \cup OnMR(RST_i, W_i)$ 
13 return  $OnMR(RST_i, W + W_i)$ 

```

---

### 5.1 Unique Assignments with Road Segments (U-RS)

Maximizing the number of unique assignments of road segment tasks would result in more number of tags collected from the workers. To ensure the maximization of the unique road segment task assignments, we track the assignments of road segments and prioritize the unassigned tasks. Following this approach, we propose *Unique Assignments with Road Segments (U-RS)* Algorithm (See Algorithm 15). This algorithm will be searching for nearest  $maxT$  unassigned road segment tasks for each worker in the current batch at time instance  $i$  and assign them to the workers if they satisfy the distance constraint. Though, the algorithm is designed to solve the Batch-MURSTA problem, it also facilitates assignment of previously assigned tasks if there are no unassigned tasks that satisfy the distance constraint of the workers. In such cases, we sort the tasks that satisfy the distance constraint based on their number of previously assigned workers to prioritize least assigned tasks for the assignment.

### 5.2 Unique Assignments with Junctions (U-J)

The previous algorithm did not consider the inherent nature of road networks for assigning the tasks. For example, consider the example in Fig. B.3a. There



**ALGORITHM 16: UNIQUE ASSIGNMENTS WITH JUNCTIONS (U-J)**

**Input:** A non-empty set of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of junction tasks  $J$  at time instance  $i$ .  $OnMR(J, W)$  is the optimal set of unique assignments before the time instance  $i$

**Output:** The Optimal set of unique assignments  $OnMR(J_i, W + W_i)$  at time instance  $i$

```

1  $OnMR(J, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3    $J_w \leftarrow \emptyset$ 
4   foreach  $j \in J$  do
5     if  $dist(j, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
6        $J_w \leftarrow J_w \cup j$ 
7    $sortByAssignedAscending(J_w)$ 
8   foreach  $j_1 \in J_w$  do
9      $j_1.assigned++$ 
10     $J_i \leftarrow UpdateAssignment(J)$ 
11     $MaxT(w) \leftarrow MaxT(w) - 1$ 
12     $OnMR(J_i, W + W_i) \leftarrow OnMR(J, W) \cup OnMR(J_i, W_i)$ 
13 return  $OnMR(J_i, W + W_i)$ 

```

are three workers and seven road segment tasks in the example. In the earlier approaches, the worker  $w_2$  has to go to individual road segment tasks like  $(t_1, t_2, t_3, t_4)$  for performing the tasks. However, in a real-world scenario, she can go to the junction where these four road segments meet ( $J_1$ ), to perform the four tasks, thus saving on the travel distance. As only worker  $w_2$  can reach  $J_1$ , it will be assigned to her, similarly  $J_2$  for  $w_3$  (See Fig. B.3b).

Similar to the U-RS algorithm, we propose *Unique Assignments with Junctions (U-J)* (See Algorithm 16). This algorithm will be searching for nearest  $maxT$  unassigned junction tasks for each worker in the current batch at time instance  $i$  and assign them to the workers if they satisfy the distance constraint. If there are no unassigned junction tasks that satisfy the distance constraint, then we sort the tasks that satisfy the distance constraint based on their number of previously assigned workers to prioritize least assigned tasks for the assignment.

## 6 Optimization Objective: Maximizing Verifiable Task Assignments

This section describes in detail the proposed assignment algorithms for the MVRSTA problem. In order to solve the Batch-MVRSTA problem, we pro-



## 6. Optimization Objective: Maximizing Verifiable Task Assignments

---

### ALGORITHM 17: VERIFIABLE ASSIGNMENTS WITH ROAD SEGMENTS (V-RS)

---

**Input:** A non-empty set of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of road segment tasks  $RST$  at time instance  $i$ .  $OnMRV(RST, W)$ , the optimal set of verifiable assignments before the time instance  $i$

**Output:** The Optimal set of verifiable assignments  $OnMRV(RST_i, W + W_i)$  at time instance  $i$

```

1  $OnMRV(RST, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3    $RST_w \leftarrow \emptyset$ 
4   foreach  $rst \in RST$  do
5     if  $dist(rst, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
6        $RST_w \leftarrow RST_w \cup rst$ 
7   foreach  $rst1 \in RST_w$  do
8     if  $rst1.assigned == 1$  then
9        $rst1.assigned++$ 
10       $RST_i \leftarrow UpdateAssignment(RST)$ 
11       $MaxT(w) \leftarrow MaxT(w) - 1$ 
12       $OnMRV(RST_i, W + W_i) \leftarrow OnMRV(RST, W) \cup OnMRV(RST_i, W_i)$ 
13 if  $MaxT(w) > 0$  then
14    $sortByAssignedAscending(RST_w)$ 
15   foreach  $rst1 \in RST_w$  do
16      $rst1.assigned++$ 
17      $RST_i \leftarrow UpdateAssignment(RST)$ 
18      $MaxT(w) \leftarrow MaxT(w) - 1$ 
19      $OnMRV(RST_i, W + W_i) \leftarrow OnMRV(RST, W) \cup OnMRV(RST_i, W_i)$ 
20 return  $OnMRV(RST_i, W + W_i)$ 

```

---

pose two methods, namely: *Verifiable Assignments with Road Segments (V-RS)*, and *Verifiable Assignments with Junctions (V-J)*.

### 6.1 Verifiable Assignments with Road Segments (V-RS)

To solve the Batch-MVRSTA problem, we propose *Unique Assignments with Road Segments (U-RS)* Algorithm (See Algorithm 17). This algorithm will search for the nearest  $maxT$  road segment tasks for each worker, that are previously assigned one worker, in the current batch at time instance  $i$  and assign them to the workers if they satisfy the distance constraint. Though the algorithm is designed to solve Batch-MVRSTA problem, it also facilitates assignment of unassigned tasks if there are no previously single worker assigned tasks that satisfy the distance constraint. In such cases, we sort the tasks that satisfy the distance constraint based on their number of previously assigned workers to prioritize least assigned tasks for the assignment, in simple words,

we prioritize the unassigned tasks.

---

**ALGORITHM 18: VERIFIABLE ASSIGNMENTS WITH JUNCTIONS (V-J)**


---

**Input:** A non-empty set of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of junction tasks  $J$  at time instance  $i$ .  $OnMR(J, W)$  is the optimal set of assignments before the time instance  $i$

**Output:** The Optimal set of assignments  $OnMR(J_i, W + W_i)$  at time instance  $t$

```

1  $OnMR(J, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3    $J_w \leftarrow \emptyset$ 
4   foreach  $j \in J$  do
5     if  $dist(j, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
6        $J_w \leftarrow J_w \cup rst$ 
7   foreach  $j1 \in J_w$  do
8     if  $j1.assigned == 1$  then
9        $j1.assigned++$ 
10       $J_i \leftarrow UpdateAssignment(J)$ 
11       $MaxT(w) \leftarrow MaxT(w) - 1$ 
12       $OnMR(J_i, W + W_i) \leftarrow OnMR(J, W) \cup OnMR(J_i, W_i)$ 
13   if  $MaxT(w) > 0$  then
14      $sortByAssignedAscending(J_w)$ 
15     foreach  $j1 \in J_w$  do
16        $j1.assigned++$ 
17        $J_i \leftarrow UpdateAssignment(J)$ 
18        $MaxT(w) \leftarrow MaxT(w) - 1$ 
19        $OnMR(J_i, W + W_i) \leftarrow OnMR(J, W) \cup OnMR(J_i, W_i)$ 
20 return  $OnMR(J_i, W + W_i)$ 

```

---

## 6.2 Verifiable Assignments with Junctions (V-J)

Similar to the V-RS and U-J algorithms, we propose *Verifiable Assignments with Junctions (V-J)*. The intuition is that a worker visiting a junction would be able to retrieve tags of all the road segments intersecting at that junction. This algorithm will be searching for nearest  $maxT$  single worker assigned junction tasks for each worker in the current batch at time instance  $i$  and assign them to the workers if they satisfy the distance constraint. If there is no single worker assigned junction tasks that satisfy the distance constraint, then we sort the tasks that satisfy the distance constraint based on their number of previously assigned workers to prioritize least assigned tasks for the assignment, i.e., we prioritize unassigned junction tasks (See Algorithm 18).

## 7 Experimental Evaluation

Parameters	Value Range
Synthetic workers	1K,2.5K,5K,10K,25K
Synthetic Workers based on Check-ins [10]	6770
maxT	3,6,12
distance threshold	1000

**Table B.1:** Experiment Parameters

### 7.1 Experiment Setup

**DataSet:** It is hard to find real datasets that reflect the workers in the real world. However, we have customized the check-ins dataset collected by [10] in the region of Aalborg, Denmark, for the experiments. However, the check-in information collected is mostly of tourists in the region and are limited (6770 unique users). Therefore, we have also used additional synthetic datasets for workers ranging from 1000 to 50,000 in number. As mentioned before, in Section 3, we have extracted the OSM road segments in the Aalborg region of Denmark. Furthermore, we have derived the junctions in the extracted OSM road network, that connects two or more road segments. Accordingly, we generated 17503 road segment tasks and 7203 junction tasks.

**Synthetic Dataset for Workers:** For the synthetic dataset, we have generated a varied number of workers from 1K to 25K. The workers are distributed according to the different residential and commercial zones in the geographical extent of the extracted road network. The workers are generated with uniform values of *maxT* and *distance* parameters.

**Synthetic Dataset for Workers based on real check-ins:** We have customized the check-ins dataset of [10] for the experiments by synthetically generating uniform values of *maxT* and *distance* parameters. Only the check-ins that fall under the geographical extent of the extracted road network are considered for assignment. There are 6770 workers in the real dataset.

**Algorithms:** We have conducted our evaluation based on the assignment approaches presented in this paper. First, we have evaluated the seven proposed algorithms against the baseline, the max-flow assignment algorithm (*Greedy*) from [13] for the extracted road segment tasks. We found out that the number of road segments assigned to workers increases significantly when the assignment is made at the junctions, without a significant increase in the average travel cost.

**Configuration and Measures:** We compared the different assignment al-

gorithms based on the following measures: number of unique road segments assigned to workers, number of verifiable road segments assigned to workers, and the average travel cost for the assigned road segment or junction task. Furthermore, we vary the number of simulated workers from 1K to 25K to evaluate the scalability of our approaches. To simulate a real scenario, we simulated a batch of workers (for, e.g. 50) for every time instance, which will be assigned to different road segments or junction tasks. We have set the size of the batch as 1000 and the distance threshold as 1000 meters for all our experiments. The distance between the workers and the road segments/junctions are calculated as the road network distance. Similarly, we varied the  $maxT$  value (3 to 12) of the workers to evaluate its effect on the above-mentioned different measures. Additionally, we have evaluated the effect of varying  $maxT$  values for a real dataset with 6770 workers. The default values for the scalability experiments are depicted in bold in the *Experiment Parameters* Table B.1. All algorithms were implemented in Java utilizing Postgresql<sup>1</sup> with PostGIS and pgrouting extensions. All experiments were conducted on the Windows 8.1 OS with Intel Core i7-5600 CPU@ 2.60G HZ and 12 GB memory.

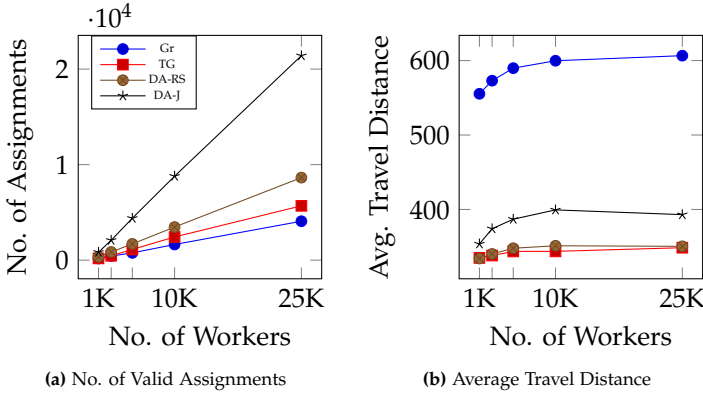


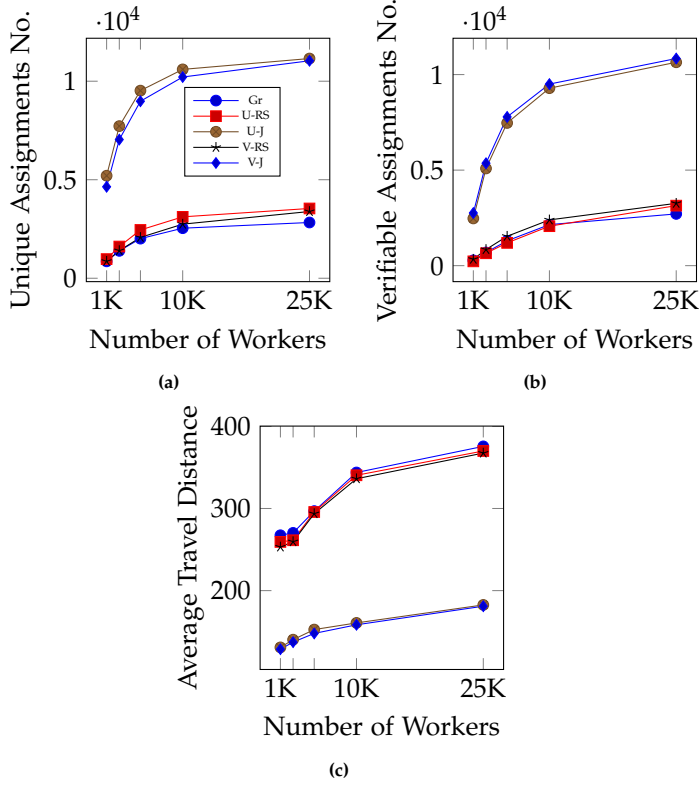
Fig. B.6: Effect of varying number of workers [7]

## 7.2 Scalability

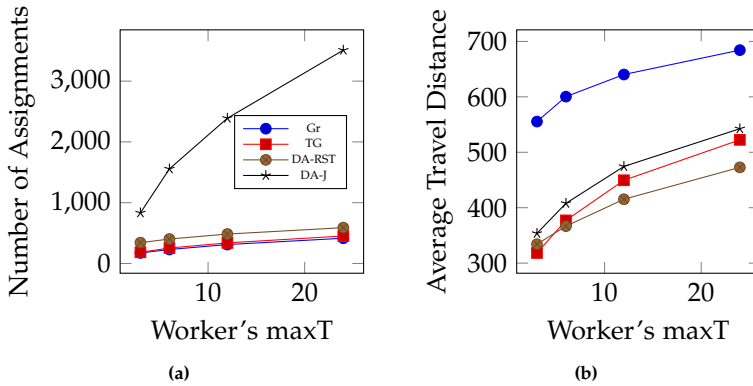
In this set of experiments, we evaluated the scalability of our proposed road task assignment algorithms by varying the number of workers from 1K to 25K. Figure B.6a illustrates the effect of the varying amount of workers on the number of assigned valid road tasks directly or indirectly through junction tasks. When compared to *Gr*, the number of road segment assignments has increased by five times, while the average travel distance for a task reduced

<sup>1</sup><https://www.postgresql.org/>

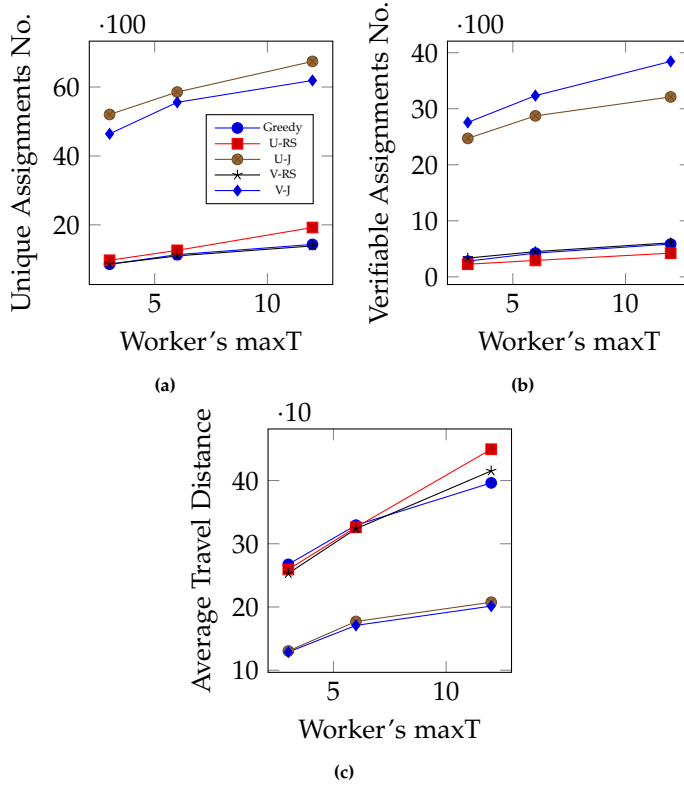
## 7. Experimental Evaluation



**Fig. B.7:** Effect of varying number of workers on :a. Number of unique road segment task assignments. b. Number of verifiable road segment task assignments. c. Average Travel Distance per assigned task.

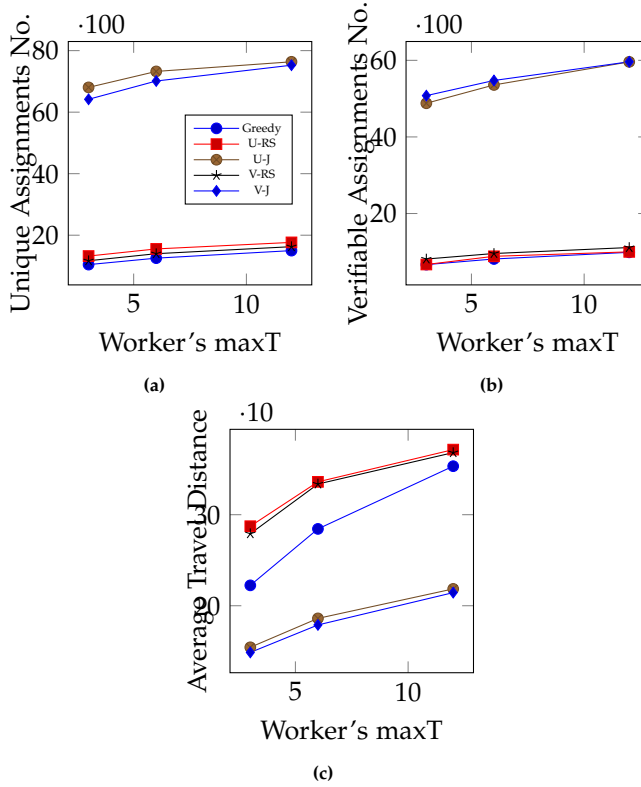


**Fig. B.8:** Effect of varying worker's maxT parameter on :a. Number of valid Assignments. b. Average Travel Distance.



**Fig. B.9:** Effect of varying worker's maxT parameter on :a. Number of unique road segment assignments. b. Number of verifiable road segment assignments. c. Average Travel Distance.

## 7. Experimental Evaluation



**Fig. B.10:** Effect of varying worker's maxT parameter on a synthetic dataset generated based on check-ins dataset [10] :a. Number of unique road segment assignments. b. Number of verifiable road segment assignments. c. Average Travel Distance.

by 35 % (See Fig. B.6b). It can also be noticed that *TG* outperforms *DA-J* with respect to average travel distance (See Fig. B.7c). The reason is due to the additional road segment tasks covered by *DA-J* when compared to *TG*. With respect to the number of assigned unique road segments, *junctions-based algorithms*: *U-J* and *V-J* have outperformed the other algorithms. Especially, *Unique assignments with Junctions (U-J)* yielded the best result when compared to the other approaches. When compared to baseline algorithm *Greedy approach (Greedy)* [13], *U-J* yielded six times as many unique road segment assignments. Similarly, *V-J* yielded a 5.4 times as many unique assignments when compared to the baseline. The reason, *junctions-based algorithms* dominate is due to the access to more number of road segments at the junctions. It can also be noticed that *U-RS* outperforms *V-RS* and the baseline *Greedy* by 10% with respect to number of unique road segment assignments (See Fig. B.7a). It can also be observed that the *junctions-based algorithms*, *U-J* and *V-J* yields converge as the workers count increases. The reason could be pointed to the limited number of tasks that can be performed by the workers. Since the tasks are limited and the workers are increasing, the number of unique assignments and the number of verifiable assignments also converges. Moreover, the values of *U-J* and *V-J* has a steep jump during the initial increase in workers. However, as the number of workers crosses 10K, there is just a marginal increase in the number of unique assignments. This phenomenon can again be explained by the limited nature of road segment tasks.

Furthermore, with respect to the verifiable task assignments, i.e., a task has been assigned to two or more workers, the *junctions-based algorithms* *V-J* and *U-J* yields seven times as many verifiable assignment as the other algorithms (See Fig. B.7b). Especially, *V-J* yielded the best result with at least eight times as many verifiable tasks compared to *Greedy*, *U-RS*, and *V-RS*. Similar to the phenomenon observed between *U-J* and *V-J* during the unique road segment assignments, *V-J* and *U-J* converge on the verifiable road segment task assignments as the number of workers increase. It can also be observed that the number of verifiable road segment tasks has a steep jump as the workers increase; however, it marginally increases once it crosses 10K workers.

With respect to the average travel distance per assigned road segment or junction task, the *junctions-based algorithms* *U-J* and *V-J* yield half as much travel distance as the other algorithms (See Fig. B.7c). It can be noticed that *V-J* performs marginally better than *U-J* and *V-RS* performs marginally better than *Greedy* and *U-RS* respectively. The reason is that the addition of a new unique road segment might result in greater travel distance than performing an already assigned closer road segment task.



### 7.3 Effect of varying $\max T$

In this set of experiments, we evaluated our algorithms by varying the  $\max T$  values of the workers. Figure B.8a illustrates the effect of varying the  $\max T$  value of workers on the number of assigned road segments tasks directly or indirectly through junction tasks. It can be observed that the *DA-J* approach outperforms all the other algorithms proposed to solve the MRSTA problem. Among the proposed algorithms to solve MRSTA problem, *DA-RS* perform the best with respect to the average travel distance (See Fig. B.8b). Fig. B.9a illustrates the effect of varying the  $\max T$  value of workers on the number of assigned unique road segments tasks directly or indirectly through junction tasks. As the  $\max T$  value increases, a general trend of increase in the number of unique tasks, verifiable tasks and an increase in average travel distance per task can be observed. It can be observed that the junctions-based algorithms, *U-J* and *V-J* outperform all the other algorithms with respect to the different measures. It can be observed that *U-J* performs better than *V-J* in case of unique assignments, and the vice versa in case of verifiable assignments (See Fig. B.9b). Similarly, *V-J* result in 10% lesser average travel distance compared to *U-J* (See Fig. B.9c).

### 7.4 Effect of varying $\max T$ on Check-ins dataset

In this set of experiments, we evaluated our algorithms by varying the  $\max T$  values of the workers on the synthetic dataset generated using check-ins [10]. Fig. B.10a illustrates the effect of varying the  $\max T$  value of real dataset workers on the number of assigned road segments tasks directly or indirectly through junction tasks. It can be observed that the junctions-based algorithms, *U-J* and *V-J* outperform all the other algorithms in terms of all the measures like the number of unique assignments ( See Fig. B.10a), the number of verifiable assignments ( See Fig. B.10b), and the average travel distance per task ( See Fig. B.10c). *U-J* and *V-J* outperform the baseline *Greedy* by at least 4.7 times as many unique assignments and 4.2 times as many verifiable assignments, respectively. Similar to the synthetic dataset, *V-J* perform the best with respect to the average travel distance per junction task (See Fig. B.10c). *U-J* and *V-J* yield half the average travel distance when compared to the baseline. We have evaluated the impact of varying the batch size on our approaches. We have observed marginal effects on the performance measures like the number of unique road segment assignments, the number of verifiable task assignments, and the average travel distance per task, due to the varying batch size; hence we decided not to discuss them more in detail.

## 8 Related Work

Spatial Crowdsourcing (SC) [8] harnesses the potential of a crowd to perform real-world spatial tasks that are not supported by conventional crowdsourcing techniques. Typically, the workers in SC move to the tasks' locations to perform tasks. The SC-server supports two types of task publishing modes [13]: Server-Assigned Task (SAT) or push-based publishing mode and Worker Selected Task (WST) or pull-based publishing mode. Due to the level of control exerted by the SC-server in the SAT publishing mode, research gained momentum in SC literature related to SAT publishing mode [26]. The SAT publishing mode, in the context of assignment of workers to tasks, involves the problem of SC-server choosing the workers for the tasks. Typically, these assignment problems aim to achieve optimization goals like maximizing the number of tasks assigned [13], minimizing the cost incurred by the server [30], improving the quality of task responses [4] or goals benefitting the workers like maximizing the reward received by the worker [3]. When the worker(s) performs the accepted task(s), then the information collected will be sent back to the SC-server. The server would, in turn, process the collected information [1].

Typically, the workers and tasks arrive dynamically to the SC-server, and SC-server has to assess the task requirements and worker preferences to assign each incoming worker to available tasks immediately [12]. This scenario is termed as online task assignment problem [6, 13, 26, 29, 30]. This scenario represents the majority real-world case scenarios of SC, where the workers and tasks are dynamic [2, 6, 23, 27–29], i.e., their arrival orders are not known beforehand. The lack of prior knowledge about the workers' or tasks' arrival leads to an additional constraint for the task assignment optimization problem. Furthermore, the task requirements, worker preferences, and the SC-server objectives leads to different type of constraints like spatial constraints [3, 25], temporal constraints [13, 19], quality constraints [5, 14, 22], and budget constraints [6, 33]. Majority of the current work does not consider all the constraints during the task assignment process.

After the task assigned to the worker, the worker visits the task location and performs the task. Upon completing the task, the worker sends the response to the SC-server. In many cases, a single task might require multiple responses to identify the truth through different data aggregation methods like, Majority Voting [17], Honeypot [18], and Expert Label Injected Crowd Estimation(ELICE) [15]. In the case of OSM, to ensure the verifiability [32], the road segment task has to be assigned to at least two workers, and the truth should be inferred by employing the data aggregation methods.

We believe SAT publishing mode of SC opens up a new avenue for OpenStreetMap (OSM), with regard to enriching the existing OSM entities with

additional tags, verifying the existing tags of OSM entities, and creating new OSM entities with appropriate tags. Generally, OSM entities are mapped remotely during mapathons [21], and the tag information is limited to the features visible on the earth observation imagery. The main advantage of using SAT publishing mode of SC is to accrue accurate ground truth information, as the worker assigned to the OSM entity is required to visit the OSM entity physically. Though there are some examples in the literature of utilizing the tags of OSM entities, for example, OSM crowdsourced data is used for proposing better routes that offer better touristic value [11], *Humanitarian Openstreetmap Team (HOT)* [20]. Our previous short paper [7] has studied the possibility of using SAT publishing mode in SC for enriching the OSM routing network. However, the short paper only provided a simplistic problem definition and basic assignment strategies. This paper proposes an integrated framework for facilitating push-based SC considering additional optimization objectives like improving the coverage of entities and ensuring verifiability of collected tags. Similarly, HOT is limited to conducting context specific surveys during disasters to map and collect information about a specific area to assist the humanitarian efforts. In contrast, this paper proposes a general end-to-end framework for enriching OSM entities' semantic tags by encouraging OSM contributors to collect tags of their nearby spatial entities.

## 9 Conclusions and Future Work

In this paper, we studied the applicability of push-based SC for enriching semantic tag information of OSM spatial entities. To achieve the goal of improving the quantity and quality of OSM entities' tags, we defined three task assignment problem with optimization objectives like; maximizing the total number of road segment task assignments, maximizing the number of unique road segment task assignments, and maximizing the number of verifiable road segment task assignments. To solve the defined problems, we proposed a push-based SC for OSM framework for enriching the semantic tags in OSM utilizing SC's SAT task publishing mode. The framework significantly extends the idea of our previous paper [7] to use spatial crowdsourcing for enriching semantic tags in OSM. The proposed framework consists of four modules namely: *Task Generator*, *Task Assignment*, *Quality Control*, and *Updating OSM Database*. The *Task Generator* module extracts the OSM spatial entities at regular intervals and forwards them as tasks to the *Task Assignment* module. The *Task Assignment* module assigns the tasks to workers and forwards the collected crowdsourced responses to the *Quality Control* module. The responses will be aggregated using non-iterative aggregation techniques like *Majority Voting* in the *Quality Control* module to identify the verifiable truth. The verified tags, along with their spatial entities, will be updated to

the OSM database at regular intervals by the *Updating OSM Database* module.

We have focused only on the task generator and task assignment modules and discussed the framework through the use case of OSM road networks. For the *Task Assignment* module, we have proposed seven feasible task assignment algorithms with three different optimization objectives like *TG*, *DA-RS*, *DA-J* for maximizing the total number of assignments, *U-RS*, *U-J* for maximizing the number of unique task assignments, and *V-RS*, *V-J* for maximizing the number of verifiable road segment task assignments. The algorithms (*DA-J*, *U-J*, and *V-J*) are built on the notion that the workers can provide information to all the road segments that meet at the visited junctions. We observed that the junctions-based algorithms *DA-J*, *U-J*, and *V-J* outperforms the other algorithms with respect to different measures, like the number of assigned road segments, the number of assigned unique road segments, the number of verifiable task assignments and the average travel distance per task. Especially, *DA-J* performs the best with five times more road segment assignments than the baseline. Similarly, *U-J* and *V-J* performs the best with six times as many and 5.4 times as many unique road segment assignments than the baseline algorithm. Moreover, *U-J* and *V-J* results in at least seven times as many more verifiable road segment assignments than the baseline. Furthermore, *U-J* and *V-J* result in around half the travel distance than the other algorithms. *U-J* performs better than *V-J* in case of unique assignments, and vice versa in case of verifiable assignments.

There are several promising directions for future work. First, there is a need to propose new quality-privacy trade-off methods to protect workers' and tasks' locations that can facilitate the budget-constrained SC task assignment and enable truth inference for push-based SC for OSM framework. Second, to exploit the full potential of workers visiting the spatial entity's location, new strategies need to be devised to maximize the number of tags captured by workers in a single visit. Additionally, new methods need to be developed to diversify the tags added by workers during their visits.

## Acknowledgements

*This research has been funded in part by the European Commission through the Erasmus Mundus Joint Doctorate Information Technologies for Business Intelligence - Doctoral College (IT4BI-DC). Xike Xie is supported by the CAS Pioneer Hundred Talents Program, Fundamental Research Funds for the Central Universities, National Natural Science Foundation of China (No. 61772492), and Natural Science Foundation of Jiangsu Province (No. BK20171240).*

## References

- [1] M. Alzantot and M. Youssef, "Crowdinside : Automatic construction of indoor floorplans," in *GIS*, 2012, pp. 99–108.
- [2] M. Asghari and C. Shahabi, "On on-line task assignment in spatial crowdsourcing," in *Big Data*, 2017, pp. 395–404.
- [3] C. Chen, S.-F. Cheng, A. Gunawan, A. Misra, K. Dasgupta, and D. Chander, "Traccs: A framework for trajectory-aware coordinated urban crowd-sourcing," in *HCOMP*, 2014, pp. 30–40.
- [4] P. Cheng, X. Lian, L. Chen, and C. Shahabi, "Prediction-based task assignment in spatial crowdsourcing," in *ICDE. IEEE*, 2017, pp. 997–1008.
- [5] K.-H. Dang and K.-T. Cao, "Towards reward-based spatial crowdsourcing," in *ICCAIS*, 2013, pp. 363–368.
- [6] S. R. B. Gummidi, T. B. Pedersen, and X. Xie, "Transit-based task assignment in spatial crowdsourcing," in *SSDBM2020. ACM*, 2020, pp. 1–12.
- [7] S. R. B. Gummidi, T. B. Pedersen, X. Xie, and E. Zimányi, "Push-based spatial crowdsourcing for enriching semantic tags in openstreetmap," in *SIGSPATIAL '19. ACM*, 2019, pp. 532–535.
- [8] S. R. B. Gummidi, X. Xie, and T. B. Pedersen, "A survey of spatial crowdsourcing," *ACM Transactions on Database Systems (TODS)*, vol. 44, no. 2, p. 8, 2019.
- [9] J. Hu, L. Huang, L. Li, M. Qi, and W. Yang, "Protecting location privacy in spatial crowdsourcing," in *APWeb*, 2015, pp. 113–124.
- [10] S. Isaj and T. B. Pedersen, "Seed-driven geo-social data extraction," in *SSTD '19. ACM*, 2019, pp. 11–20.
- [11] G. Jossé, K. A. Schmid, A. Züfle, G. Skoumas, M. Schubert, M. Renz, D. Pfoser, and M. A. Nascimento, "Knowledge extraction from crowdsourced data for the enrichment of road networks," *Geoinformatica*, vol. 21, no. 4, pp. 763–795, 2017.
- [12] R. M. Karp, "On-line algorithms versus off-line algorithms: How much is it worth to know the future?" in *Proc. of the Information Processing IFIP Congress (1)*, vol. 12, 1992, pp. 416–429.
- [13] L. Kazemi and C. Shahabi, "Geocrowd: enabling query answering with spatial crowdsourcing," in *GIS*, 2012, pp. 189–198.
- [14] L. Kazemi, C. Shahabi, and L. Chen, "Geotrucrowd: trustworthy query answering with spatial crowdsourcing," in *GIS*, 2013, pp. 314–323.
- [15] F. K. Khattak and A. Salieb-Aouissi, "Quality control of crowd labeling through expert evaluation," in *NIPS 2nd Workshop on Computational Social Science and the Wisdom of Crowds*, vol. 2, 2011, p. 5.
- [16] J. Kleinberg and E. Tardos, *Algorithm design*. Pearson Education India, 2006.
- [17] L. I. Kuncheva, C. J. Whitaker, C. A. Shipp, and R. P. Duin, "Limits on the majority vote accuracy in classifier fusion," *Pattern Analysis & Applications*, vol. 6, no. 1, pp. 22–31, 2003.

## References

- [18] K. Lee, J. Caverlee, and S. Webb, "The social honeypot project: protecting online communities from spammers," in *WWW*, 2010, pp. 1139–1140.
- [19] Y. Li, M. L. Yiu, and W. Xu, "Oriented online route recommendation for spatial crowdsourcing task workers," in *Advances in Spatial and Temporal Databases*. Springer, 2015, pp. 137–156.
- [20] OpenStreetMap, "Humanitarian OpenStreetMap Team," <https://www.hotosm.org/>, 2021, [Online; accessed 28-Jan-2021].
- [21] —, "Mapathon," <https://wiki.openstreetmap.org/wiki/Mapathon>, 2021, [Online; accessed 28-Jan-2021].
- [22] J. Rogstadius, V. Kostakos, A. Kittur, B. Smus, J. Laredo, and M. Vukovic, "An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets." in *ICWSM*, 2011, pp. 17–21.
- [23] T. Song, Y. Tong, L. Wang, J. She, B. Yao, L. Chen, and K. Xu, "Trichromatic online matching in real-time spatial crowdsourcing," in *ICDE*, 2017, pp. 1009–1020.
- [24] D. Sun, K. Xu, H. Cheng, Y. Zhang, T. Song, R. Liu, and Y. Xu, "Online delivery route recommendation in spatial crowdsourcing," *World Wide Web*, pp. 1–22, 2018.
- [25] H. To, L. Fan, L. Tran, and C. Shahabi, "Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints," in *PerCom*, 2016, pp. 1–8.
- [26] H. To, C. Shahabi, and L. Kazemi, "A server-assigned spatial crowdsourcing framework," *ACM Transactions on Spatial Algorithms and Systems*, vol. 1, no. 1, p. 2, 2015.
- [27] Y. Tong, J. She, B. Ding, L. Chen, T. Wo, and K. Xu, "Online minimum matching in real-time spatial data: experiments and analysis," *VLDB*, vol. 9, no. 12, pp. 1053–1064, 2016.
- [28] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen, "Online mobile micro-task allocation in spatial crowdsourcing," in *ICDE*, 2016, pp. 49–60.
- [29] Y. Tong, L. Wang, Z. Zhou, B. Ding, L. Chen, J. Ye, and K. Xu, "Flexible online task assignment in real-time spatial data," *VLDB Endowment*, vol. 10, no. 11, pp. 1334–1345, 2017.
- [30] L. Tran, H. To, L. Fan, and C. Shahabi, "A real-time framework for task assignment in hyperlocal spatial crowdsourcing," *ACM Transactions on Intelligent Systems and Technology*, p. 37, 2017.
- [31] O. Wiki, "Import guidelines for openstreetmap," <https://wiki.openstreetmap.org/wiki/Import/Guidelines>, accessed November 28, 2019.
- [32] —, "Verifiability in openstreetmap," <https://wiki.openstreetmap.org/wiki/Verifiability>, accessed November 28, 2019.
- [33] L. Zhang, X. Lu, P. Xiong, and T. Zhu, "A differentially private method for reward-based spatial crowdsourcing," in *ATIS*, 2015, pp. 153–164.

# Paper C

## Spatial Entity Linkage with the Aid of Spatial Crowdsourcing

Suela Isaj, Srinivasa Raghavendra Bhuvan Gummidi, Torben  
Bach Pedersen, and Xike Xie

Unpublished Manuscript

## Abstract

*Linking spatial entities across multiple location-based sources can offer rich semantic information about the spatial entities. However, to link the spatial entities from different sources, it must be established whether they represent the same physical entity or not, leading to a spatial entity linkage problem. In this paper, we aim to resolve the spatial entity linkage problem by exploiting the spatial crowdsourcing workers' wisdom. However, given the reward budget limitations, we propose a hybrid Skycrowd solution incorporating machine learning-aided spatial crowdsourcing (SC) and automatic labeling (AL) techniques. The Skycrowd solution comprises of four phases: 1) extracting the pairs of entities from a set of entities across multiple location-based sources and creating spatial entity linkage (SEL) tasks, 2) employing skyline-based AL technique to label the SEL tasks, 3) identifying the grey area of skylines wherein the tasks labeled by AL technique has high chances of mislabeling, and 4) crowdsourcing the tasks lying in the grey area utilizing an SC-based active learning approach. The proposed Skycrowd solution performs 30% better than the automatic labeling of the entire set of SEL tasks with respect to F-measure and achieves a F-measure value of 0.91 by spending a fraction (7%) of the reward budget required for crowdsourcing the entire set of SEL tasks. The SC-based active learning approach achieves similar results to crowdsourcing all the SEL tasks in the grey area by spending just 64% of the reward budget required for crowdsourcing the entire grey area.*

*The layout has been revised.*



# 1 Introduction

The growing trend of location-based services (LBS) [18] has unearthed a huge database of spatial entities [13] with spatial attributes like location, address and aspatial attributes like name, phone number, etc. These spatial entities can serve rich semantic information to spatial data-based systems like context-based route planners, geo-recommender systems, etc., better than simple spatial objects with only the location attribute. Since spatial entities can share location, like in the case of shops in a shopping mall, their identities depend on the attribute information. However, the attribute information of the spatial entities could be inconsistent across multiple sources like Google Places <sup>1</sup>, OpenStreetMap <sup>2</sup>, etc. For example, a spatial entity with name "Naesbyhoved skov" is located at (55.4119,10.3775) with associated keywords "A-la-carte", "Dine-in" in source A, and a spatial entity with name "Restaurant Næsbyhoved skov" is located at (55.412,10.376) with associated keywords "Take-away", "restaurant" in source B refers to the same physical entity. However, it is not straightforward to determine that these two spatial entities in different sources refer to the same physical entity as the attribute information is different. We need to develop a solution to automatically link the spatial entities that refer to the same physical entities across different sources to provide richer semantic spatial entities. This problem is defined as *Spatial Entity Linkage Problem* in [13].

The spatial entity linkage problem across different location-based sources was solved by [13] by applying a skyline-based automatic labeling technique. However, the method could only deliver a *F-measure* value of 0.71, *precision* of 0.86 and *recall* of 0.61. In this paper, we aim to improve the *F-measure* by incorporating machine learning-aided spatial crowdsourcing (SC) techniques for the tasks that are potentially mislabeled by the automatic labeling method. The intuition is that the SC workers can visit the spatial entities' locations and verify whether they belong to the same physical entity. To resolve the spatial entity linkage problem using spatial crowdsourcing, we define the *spatial entity linkage with the aid of spatial crowdsourcing problem* to maximize the *F-measure* for a given reward budget. We propose a four-phase *Skycrowd* solution to address the defined problem, which includes extraction of spatial entity linkage tasks, automatic labeling of spatial entity linkage tasks using the SkyEx algorithm proposed in [13], identification of the grey area containing the tasks that are potentially mislabeled by the automatic labeling technique, and finally crowdsourcing the tasks in the grey area utilizing a SC-based active learning approach.

The main contributions we offer in this paper are:

---

<sup>1</sup><https://cloud.google.com/maps-platform/places>

<sup>2</sup>[www.openstreetmap.org](http://www.openstreetmap.org)

- We study and define the optimization problem for maximizing the *F-measure* of the resultant labeled spatial entity linkage tasks solved by a combination of automatic labeling and spatial crowdsourcing techniques for a given reward budget.
- We propose a hybrid solution, *Skycrowd* for addressing the defined *spatial entity linkage with the aid of spatial crowdsourcing problem* to minimize the reward budget utilization and to maximize the *F-measure*.
- We evaluate the effectiveness of the proposed *Skycrowd* solution by comparing against the performance of solutions employing crowdsourcing to resolve all tasks, all the grey area tasks, and only employing automatic labeling.

The remainder of the section is organized as follows: In Section 2, we describe the related work regarding spatial entity linkage problem and spatial crowdsourcing. In Section 3, we define the preliminaries and the spatial entity linkage with the aid of spatial crowdsourcing problem. In Sections 4, 5, 6, 7, we detail the proposed *Skycrowd* solution and its different parts. Section 8 presents the experimental evaluation of the proposed *Skycrowd* solution. Finally, we conclude in Section 9 and propose some potential future research directions.

## 2 Related Work

The problem of spatial entity linkage is similar to the general entity resolution problem but differs in the treatment of the spatial attributes. For example, the entity resolution problem usually uses blocks to organize potential candidates together [17] instead of performing all the comparisons; in the case of spatial entities, the blocks are based on the spatial coordinates of the entities [3, 14, 16, 19]. Moreover, the shapes of the spatial data require special treatment in the comparison between each other, which go beyond Euclidean distance metrics. The spatial entity resolution problem differs from the spatial data integration problem [1, 2, 20, 23] since the spatial entities are identified not only by the coordinates but also by other attributes such as the name, the semantics, etc. Having the same coordinates does not necessarily mean that the spatial entities are the same. The approaches used for spatial data integration include supervised learning [19] or threshold-based algorithms [3, 14, 16]. The supervised learning approaches use the labels of the data to learn a model and test it on the rest. The threshold-based approaches compare the attributes and base the decision whether to match two spatial entities on a similarity score, which is discovered through experiments. In addition to these methods, the work in [13] performs the pairwise comparisons

### 3. Problem Definition

of the spatial entities and uses a skyline based technique to make the decision, which does not need labeled data and avoids the extensive experiments for defining a threshold that could be dataset-dependent.

However, the results achieved by these automatic techniques could be improved by human reasoning. It is difficult for an algorithm to decide precisely on a real-world entity. Thus, spatial crowdsourcing techniques [11] could be of interest. The workers in spatial crowdsourcing can be assigned to their neighboring spatial entities to verify whether they belong to the same physical entity. There are several works that use crowdsourcing for entity resolution problem (not spatial). They can be categorized as full crowdsourcing or hybrid methods. The former [full crowdsourcing] use only crowdsourcing to integrate the entities. Their contribution is on using the crowdsourcing budget wisely by choosing the next most beneficial question [6, 9, 10, 21] and proposing metrics to measure the benefit of the queries [9]. The work in [10] goes further by crowdsourcing every step of the problem, from the blocking until the accuracy estimation. The latter [hybrid] have an algorithm in place to solve a part of the problem and then use the budget on the crowdsourcing [22, 24]. For example, the Jaccard similarity of the entities is used in [24] to detect potential candidates, and then the budget is organized to create crowdsourcing tasks. In [22], a machine learning model learns first the behavior and classifies the pairs. Crowdsourcing is used for those pairs with low confidence from the classifier. Finally, some of the works contribute to accommodating errors from the crowdsourced tasks [6, 21], while others consider the crowdsourcing as an oracle that is always right [9, 22].

In contrast to the previous works, we solve the problem of spatial entity linkage with a combination of a skyline-based approach and spatial crowdsourcing. We first organize the spatial entities in blocks, compare them pairwise, and label them. We further improve the results of the algorithm by detecting a grey area in the skyline procedure, and we resolve these pairs through a novel machine learning-aided spatial crowdsourcing technique. The proposed technique incorporates active learning approach through cluster-based sampling and transitive closure checks in order to minimize the reward budget. Moreover, our spatial crowdsourcing solution uses the activity of the users in social networks to assign tasks in accordance with the worker’s expertise and recent activity.

## 3 Problem Definition

In this section, we define the preliminaries that will be used throughout the paper. Furthermore, we will define the *Spatial Entity Linkage with the aid of Spatial Crowdsourcing* problem.

**Definition 19.** (Spatial Entity): A *Spatial Entity*, denoted by  $s$ , is a real world place like a shop, or a public utility center, etc. It has a geographical location point  $s.p$ , and a set of different attributes  $\{s.a_1, s.a_2, \dots, s.a_n\}$  like address of the entity  $s.a_i$ .

As mentioned in the Section 1, some of the spatial entities might be redundant and refer to the same physical entity. To resolve whether a given pair of spatial entities are the same, we define the *Spatial Entity Linkage task* in line with the *Spatial Entity Linkage problem* [13].

**Definition 20.** (Spatial Entity Linkage task): A *Spatial Entity Linkage (SEL) task*, denoted by  $t$ , represents a classification task with the objective to determine whether a pair of spatial entities  $s_1$  and  $s_2$ , represent the same physical entity.  $t = \langle s_1, s_2, l \rangle$ , where the label  $l$  represents whether the entities are a match.

$$l = \begin{cases} l^+ & \text{if } s_1 \text{ and } s_2 \text{ belong to the same physical spatial entity} \\ l^- & \text{otherwise} \end{cases}$$

We refer to the pairs labeled as  $l^+$  as the *positive class* and to those labeled as  $l^-$  as the *negative class*. In our context, we define *True Positives* (TP) as actual positive pairs labeled as positives, *True Negatives* (TN) as actual negative pairs labeled as negatives, *False Positives* (FP) as actual negative pairs labeled as positives, and *False Negatives* (FN) as actual positive pairs labeled as negatives. The *precision* of a classifier is  $precision = \frac{TP}{TP+FP}$  and the *recall* is  $recall = \frac{TP}{TP+FN}$ . In order to measure the effectiveness of the classifier, we use the *F-measure* ( $F1$ ) =  $2 \frac{precision * recall}{precision + recall}$ .

The spatial entity linkage problem has been solved with different automatic labeling methods (AL) in [3, 13, 14, 16]. The decision to label a pair is usually based on the attributes of the spatial entities. Sometimes, it is difficult for automatic labeling methods to decide on the label of a pair. For these cases, a SEL task can be resolved by the wisdom of the crowd. *Therefore, a combination of automatic labeling (AL) and spatial crowdsourcing (SC) [11] can together solve the spatial entity linkage problem better.*

When an SEL task is solved by an AL method, it will not cost anything. However, the *F-measure* of the AL results is dependent on the quality of the AL. Whereas in the case of SC, when an SEL task is resolved, we assume that the label provided by the crowd is always correct, i.e., the ground truth. Consequently, the *F-measure* of the SC results is a perfect 1.0. However, a reward needs to be paid to the crowd, resulting in extra cost.

To determine whether to use only AL or only SC or a combination of both is dependent on the given reward budget. Let us denote by  $R$  the given *reward budget* for solving a set of SEL tasks and by  $R^*$  the required *reward*

#### 4. SkyCrowd Solution

*budget* to solve all the SEL tasks.

$$\begin{cases} R \geq R^* & \text{Only SC} \\ R = 0 & \text{Only AL} \\ 0 < R < R^* & \text{Combination of AL and SC} \end{cases}$$

If the given reward budget is more than the budget required to solve all the SEL tasks ( $R \geq R^*$ ), then we can solve the *Spatial Entity Linkage* problem by just using SC. If there no given budget ( $R = 0$ ), the *SEL* problem has rely just on AL. In the case of limited given budget ( $0 < R < R^*$ ), we have to use a combination of AL and SC. Finding the best trade-off between AL and SC leads to an optimization problem, the *Spatial Entity Linkage with the aid of Spatial Crowdsourcing* problem. The problem is defined as:

**Definition 21.** (Spatial Entity Linkage with the aid of Spatial Crowdsourcing): Given a reward budget  $R$  and a set of SEL tasks  $SEL_{ALL}$ , Spatial Entity Linkage with the aid of Spatial Crowdsourcing (SELSC) problem is an optimization problem to distribute SEL tasks between AL and SC, with an objective of maximizing the F-measure of the resultant labeled SEL tasks.

$$\arg \max_{SEL_{AL} \sqcup SEL_{SC} = SEL_{ALL}} F1(SEL_{AL}, SEL_{SC}) \quad (C.1)$$

, where  $SEL_{AL}$ ,  $SEL_{SC}$  are the sets of SEL tasks assigned to AL and SC, respectively.

## 4 SkyCrowd Solution

In this section, we introduce our proposed solution to the SELSC problem, *SkyCrowd*. *SkyCrowd* employs *Skyline-based Automatic Labeling (AL)* [13] and *Machine learning-aided Spatial Crowdsourcing (SC)* techniques for resolving the SELSC problem. *SkyCrowd* takes a set of spatial entities  $S$  as input and outputs a resolved/ partially resolved SEL tasks set  $CT$ . *SkyCrowd* consists of the following phases (See Fig. C.1):

- **Extracting SEL Tasks:** Given a set of entities  $S$ , we extract the SEL tasks containing entity pairs that require resolution. We utilize the spatial blocking technique, *QuadFlex* [13], to extract the SEL tasks.
- **Automatic Labeling of SEL Tasks:** We employ a skyline-based automatic labeling technique, [13] to resolve the extracted SEL tasks. The pairs of entities of SEL tasks are compared on their attribute values to estimate their utility. The tasks with same utility are assigned to the

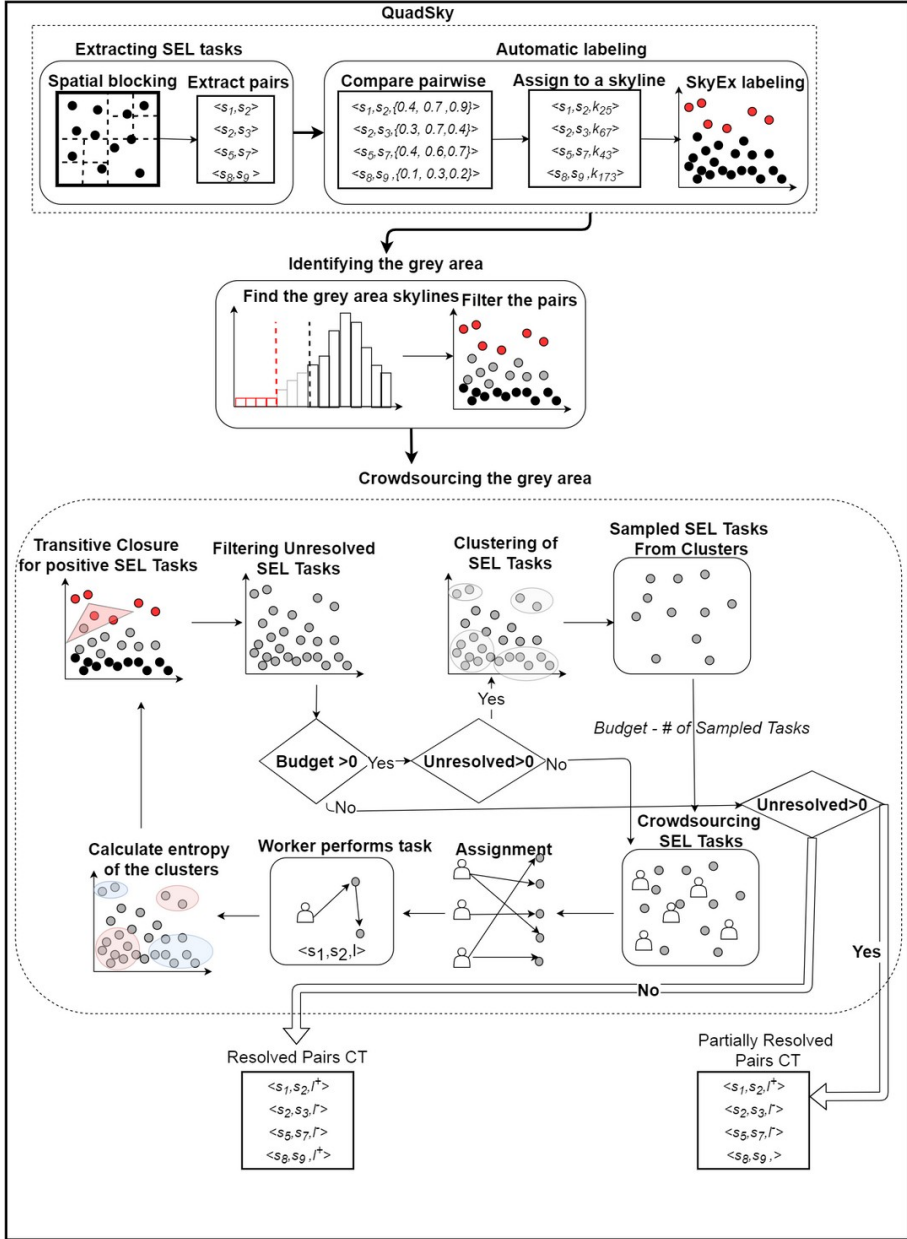


Fig. C.1: SkyCrowd Solution

same skyline or the same pareto optimal frontier. Lower similarity values of the attributes result in assignment to higher skylines and vice versa. Finally, the *SkyEx* algorithm [13] fixes the cut-off skyline level

## 5. QuadSky Approach

$k$  that best separates the positive and negative classes, and labels the SEL tasks based on the assigned skylines. The tasks that are assigned to a skyline of at most the  $k^{th}$  level, will be labeled positive  $l^+$  and the remaining will be labeled negative  $l^-$ .

- **Identifying the Grey Area:** Since the decision for labeling the pairs is based on the skyline, the pairs that fall near the cut-off  $k^{th}$  skyline might be mislabeled. We propose a technique for identifying the SEL tasks that are more likely to be mislabeled based on their proximity to  $k^{th}$  skyline. The technique identifies a skyline  $k^*$ , such that all the tasks with at most the skyline  $k^*$  level are labeled as positive. The tasks that are assigned to the skyline in the interval  $(k^*, k]$  are labeled negative and referred to as the *grey area*.
- **Crowdsourcing the Grey Area:** The SEL tasks identified as part of the grey area will be resolved by SC. Workers are assigned SEL tasks to maximize the F1 gain, given a reward budget. Given the cost of SC, it might not be feasible to crowdsource all the SEL tasks in the grey area, i.e.,  $R < R_{grey}$ . Therefore, we propose optimization strategies facilitating active learning in SC to reduce the cost. As illustrated in Fig. C.1, the proposed SC-based active learning approach consists of four iterative steps, checking for transitive closure of positive SEL tasks, clustering of unresolved grey area tasks, crowdsourcing samples of the clusters, and inferring the label of the cluster based on the entropy of crowdsourced sample. Furthermore, we propose proximity-based and past visit-based assignment algorithms to further reduce the SC cost.

In the following sections, we will elaborate the above-mentioned phases in detail.

## 5 QuadSky Approach

In this section, we describe the first two phases of the *Skycrowd* solution (See Fig. C.1). We employ the *QuadSky* approach [13] for extracting and automatic labeling the SEL tasks. *QuadSky* consists of a spatial blocking technique, which aids the extraction of SEL tasks, comparing pairwise the spatial entities in an SEL task, assigning them a skyline, and finally, labeling the pairs.

### 5.1 Extracting SEL tasks

The spatial entity linkage problem involves comparing the spatial entities and deciding which pairs belong to the same physical entity. A blocking technique that organizes potential matches into the same blocks reduces the comparisons significantly. We use the blocking technique in [13]. The spatial



entities are inserted into a structure similar to a quadtree named *QuadFlex*. *QuadFlex* respects the distance between entities and the density of a node, and also allows the assignment of a spatial entity to more than one node.

*QuadFlex* is dependent on two parameters: the diagonal of the node  $m$  and the density  $d$ . The diagonal of the node represents the maximal distance between two spatial entities in a block. For example, if  $m = 50$ , then, all the spatial entities in this block are at most to meters far from each other. If for a block,  $m > 50$ , then the *QuadFlex* tree will split further into four children. Similarly, the blocks are limited by a density value. We prefer to split further the blocks with a high density because they usually correspond to city centers, crowded areas etc, where it is common to have spatial entities close to each other. If the density is higher than the predefined  $d$ , the *QuadFlex* tree will split again into four children. Another characteristic of *QuadFlex* is that it allows neighboring spatial entities to belong to more than one block. Similarly to a quadtree, *QuadFlex* will physically split into four non-overlapping children, but the spatial entity assignment will allow a logical overlap of the neighbors. The algorithm of *QuadFlex* is detailed in [13]. After all the spatial entities are inserted into the *QuadFlex* tree, we retrieve its leaves. These leaves are the spatial blocks of spatial entities that respect the spatial proximity and density of the area. We extract pairs within each block and transform them into SEL tasks of pairs of spatial entities  $\langle s_i, s_j, p_i, p_j \rangle$ .

## 5.2 Automatic Labelling of SEL tasks

The SEL tasks produced by the spatial blocking could be fully solved by spatial crowdsourcing if  $R \geq R^*$  (See Section 7). In that case, SELSC would consist only of SC. However, if  $R < R^*$ , we can automatically label the pairs and resolve a part of the SEL tasks with an algorithmic solution and use the reward budget wisely on problematic areas. After having the blocks, we compare the spatial entities pairwise regarding their name, address, and categories. The names are compared using Levenshtein distance [15], the addresses for an exact match of street name and building number after transformations (different formats, detection of postal codes, etc.) and categories using Wu&Palmer similarity measure (wup) [26] from relationships discovered from Wordnet [8].

The result of the pairwise comparison is a set of pairs of spatial entities  $\langle s_i, s_j \rangle$  and their similarity scores of  $n$  attributes  $\delta_1, \delta_2, \dots, \delta_n$ . To decide whether these similarities are significant for linking the entities, we rely on the Pareto Optimal law [5] and the *SkyEx* algorithm used in [13]. *SkyEx* abstracts the similarity function of all the attributes by the concept of *utility*. A pair  $\langle a, b \rangle$  has a higher utility than  $\langle c, d \rangle$  as long as one of the attribute similarities of  $\langle a, b \rangle$  is higher compared to  $\langle c, d \rangle$  while the others remain constant. The pairs that have the same utility form a *skyline*. *SkyEx* assigns a skyline of level  $k$  to the compared



## 6. Identifying the grey area

pairs. For example, the similarities of the name, address, and categories of the pair  $\langle s_1, s_2 \rangle$  are 0.4, 0.7, 0.9, respectively. *SkyEx* assembles all these similarities to a skyline, so  $\langle s_1, s_2 \rangle$  is assigned to a skyline of level  $k$ . The first levels of skylines are the best candidates. Hence, the bigger the  $k$ , the less likely two pairs are a match. Through experiments, we fix the skyline level  $k$  that can best separate the classes. The pairs that belong to a skyline at most  $k$  are labeled as  $l^+$ , and the rest, as  $l^-$  (See Automatic Labeling phase of Fig. C.1).

## 6 Identifying the grey area

The automatic labeling performed in the previous phase is on the assumption that the skylines can detect a good separation of the classes. *SkyEx* algorithm [13] fixes a  $k$  value such that offers the best trade-off between precision and recall. However, *SkyEx* mislabels the pairs when approaching  $k$ . Furthermore, even though the skylines show robust results for separating the classes when going deeper into the skyline levels, the precision of the labeling starts to fall. Thus, we need a technique to identify the area where AL tend to mislabel the SEL tasks.

First of all, we need to separate the potential candidates from the rest. *A pair of spatial entities from  $S$  is more likely to be a non-match than to be a match.* This means that we expect a dataset with a high class imbalance. A pair that is not likely to match is usually associated with low similarity values of the attributes and, consequently, a higher skyline level (big  $k$ ). Moreover, the best pairs (part of the first skylines) are a minority compared to the rest of the dataset. They are expected to form a *long tail* that can be distinguished from the rest of the population. Thus, we can easily detect these pairs by a histogram of the density of points in the skylines.

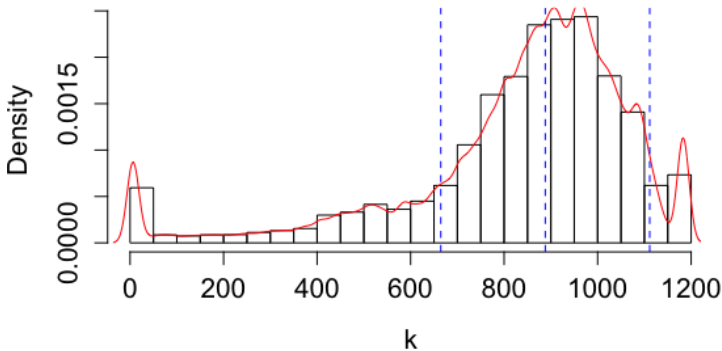


Fig. C.2: Histogram of point density in skylines

Fig. C.2 shows the distribution of the pairs of points in our dataset (see

Section 5). The x-axis shows the skyline level  $k$  while the y-axis represents the density of the points in that skyline. As expected, it is more common for two points to fall into deeper skylines (high  $k$ ). A small number of pairs gather around the first skylines, forming a left tail. It is highly likely that these pairs belong to the positive class.

**Definition 22.** (Left Tail) The skyline population that lies on the left of the distribution of the majority of the pairs is denoted as the *left tail*.

The *left tail* lies approximately between  $[0, m - z_\alpha * \sigma]$  where  $m$  is the median of the distribution,  $\sigma$  is the variance and  $\alpha$  is the tolerance interval. The median is a better solution than the mean for skewed distributions such as ours. An interesting observation is that *the potential candidates in the left tail, forming 17% of the population, obey the Pareto principle [4], a principle applied in several fields, which states that 20% of the population owns 80% of the wealth.* In our context, it can be interpreted as *examining 20% of the selected pairs (initial skylines) will detect 80% of the positive pairs.* From our experiments, we found out that examining 17% of the pairs in *left tail* discovers 85% of the positive class.

However, examining all 17% of the population might not always be a feasible solution if the reward budget of the crowdsourcing is limited. In that case, the pairs in the first skylines are already very good candidates, so it might not be necessary to use the budget on those. Thus, we introduce the concept of the *grey area*, the skyline levels with potential candidates that are neglected by the labeling algorithm.

**Definition 23.** (Grey Area) The grey area consists of pairs of spatial entities that are positioned after the  $k^{th}$  skyline to the right border of *left tail*.

For example, Fig. C.3 shows the procedure for fixing  $k$  in the left figure. We select the value of  $k$  that yields the highest *F-measure*. After that (green line), we start loosing in precision and *F-measure* falls. From the green line to the first blue line (right figure), is identified as the grey area. Algorithm 19 describes the procedure for identifying the grey area tasks. We find the tasks that lie in the skylines between  $k^*$  (skyline that yields the highest F-measure) and  $k$  of SkyEx algorithm of the previous phase (See Identifying the grey area phase of Fig. C.1). For the grey area, we can decide to crowdsource all, or use the SC-based active learning approach (see the next section). Thus, we overwrite the labels for the grey area and finally, return the labeled pairs.

## 7 Crowdsourcing the Grey Area

In this section, we describe the process of crowdsourcing the SEL tasks in the grey area. Ideally, the reward budget  $R$  should facilitate crowdsourcing of

## 7. Crowdsourcing the Grey Area

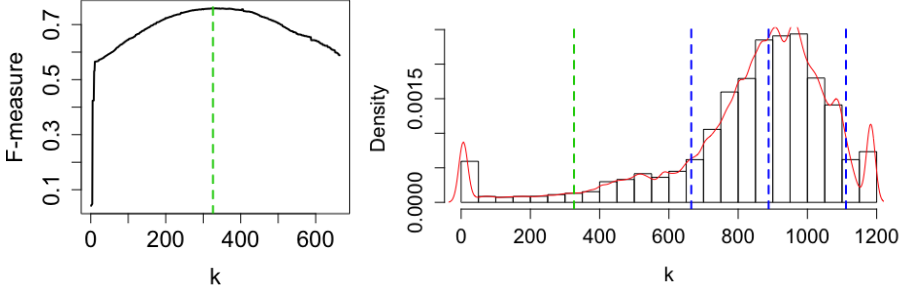


Fig. C.3: Identifying SEL tasks in Grey Area

---

### ALGORITHM 19: Identifying SEL tasks in Grey Area

---

**Input:** A set of SEL tasks and their skylines  $\{\langle s_i, s_j, ks \rangle\}$ , SkyEx Algorithm fixed skyline  $k$  and a reward budget  $R$

**Output:** A set of grey area SEL tasks  $\{\langle s_i, s_j, l \rangle\}$

- 1 Find  $k^*$  that yields the highest *F-measure*
  - 2 Label  $\{\langle s_i, s_j, ks \rangle | ks > k^* \text{ and } ks \leq k\}$  as negative and added to grey area SEL task set
  - 3 **return**  $\{\langle s_i, s_j, l \rangle\}$
- 

all the grey areas' tasks. However, in most cases the budget is not enough to crowdsource all the grey area tasks, and we need to optimize the limited workers resource to make the best use of the given budget, thus leading to an optimization problem of minimizing the number of SEL tasks in the grey area that require worker assignments for resolution.

### 7.1 SC-based Active Learning Approach

To resolve the optimization problem, we propose an SC-based active learning approach (See Algorithm 20), that tries to resolve as many tasks as possible given a reward budget  $R$ . The intuition behind the SC-based active learning approach is that the approach learns from the crowdsourced sample tasks during each iteration. There are four iterative steps (See Crowdsourcing the grey area section in Fig. C.1) in the approach; 1) Transitive closure of the unresolved grey area tasks, 2) Clustering of the unresolved grey area tasks, 3) Crowdsourcing samples from the clusters, and 4) Inferring the label for the remaining tasks in the cluster based on the entropy threshold. The four iterative steps of the approach terminate when the budget exhausts ( $R = 0$ ), or when all the tasks are resolved. The intuition regarding the inferred labels of clusters based on entropy is to resolve tasks without spending any budget. If there is any budget left after the auto-inference of labels, the unused budget will be used to crowdsource the inferred tasks to gain more

**ALGORITHM 20: CROWDSOURCING THE GREY AREA**

**Input:** A set of grey area tasks  $S$  and their skylines  $\{\langle s_i, s_j, k \rangle\}$ , a set of total SEL tasks  $TS$ , a set of workers  $W$ , entropy Threshold  $e_T$ , reward Budget  $R$ , Cochran's formula constants  $Co1 = 384.16$  and  $Co2 = 383.16$

**Output:** A set of resolved tasks  $CT$  with their label  $\{\langle s_i, s_j, l \rangle\}$

```

1 Inferred Tasks  $IS \leftarrow \emptyset$ 
2 Unresolved Tasks  $US \leftarrow \emptyset$ 
3 Crowdsourced Tasks  $crowdTasks \leftarrow \emptyset$ 
4 while  $R > 0$  and  $US \neq \emptyset$  do
5    $S_t \leftarrow transitiveClosureSearch(TS)$ 
6    $S \leftarrow S \setminus S_t$ 
7    $US \leftarrow S$ 
8    $Clusters\ C \leftarrow DBSCANClustering(US)$ 
9   foreach  $c \in C$  do
10     $sampleSize \leftarrow (Co1 + c.size) / (Co2 + c.size)$ 
11     $crowdTasks \leftarrow performCrowdsourcing(sampleSize, c, W, R)$ 
12     $US \leftarrow US \setminus crowdTasks$ 
13     $R = R - \sum_{i=1}^{crowdTasks.size} crowdTasks[i].r$ 
14     $pos \leftarrow (no\ of\ positives\ in\ crowdTasks) / SampleSize$ 
15     $e \leftarrow -(pos * \log[2](pos)) - (1 - pos)(\log[2](1 - pos))$ 
16    if  $e < e_T$  then
17       $US \leftarrow US \setminus c$ 
18       $IS \leftarrow IS \cup c$ 
19      if  $pos > 0.5$  then
20        Label all remaining tasks in cluster  $c$  as  $l^+$ 
21      else
22        Label all remaining tasks in cluster  $c$  as  $l^-$ 
23    Resolved Tasks  $CT \leftarrow CT \cup crowdTasks \cup IS$ 
24    Update Total Tasks Set  $TS \leftarrow TS \cup crowdTasks \cup IS$ 
25     $S \leftarrow S \setminus crowdTasks \setminus IS$ 
26 if  $US = \emptyset$  and  $T > 0$  then
27   Label the inferred SEL tasks with crowdsourcing
28    $crowdTasks \leftarrow performCrowdsourcing(T, IS, W)$ 
29    $CT \leftarrow CT \cup crowdTasks$ 
29 return  $CT$ 

```

accuracy (Lines 26-28 of Algorithm 20).

Based on the reward budget  $R$ , the SC-based active learning approach has two outcomes. First, if there is insufficient budget, then the outcome would consist of partially resolved tasks. Second, if there is sufficient budget, then the outcome consists of fully resolved tasks. In the following sections, we detail the remaining three critical steps in detail.

## 7. Crowdsourcing the Grey Area

---

### ALGORITHM 21: Task Assignment Algorithm based on past visits

---

**Input:** A set of sample tasks  $SEL$ , workers  $W$  with base reward  $C$ , and reward budget  $R$

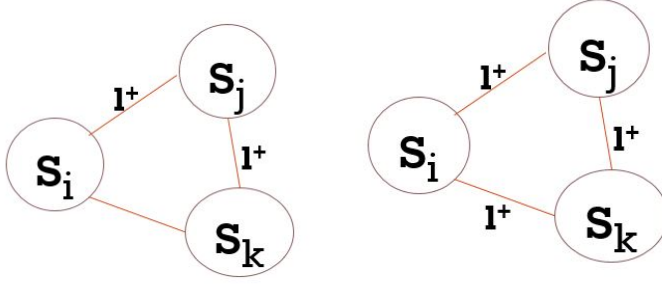
**Output:** A set of assignments  $VA$  with reward  $\langle w, sel, r \rangle$

```

1 Probability  $p \leftarrow \text{random}(0,1)$ 
2 foreach  $sel \in SEL$  do
3   if  $R > 0$  then
4     Find workers who visited task location  $l$  in the past
4      $ExperiencedWorkers \leftarrow \text{findPastVisitWorkers}(l)$ 
5     Find workers who visited close to the task location  $l$  in the past
5      $CloseWorkers \leftarrow \text{findPastVicinityWorkers}(l)$ 
6     if  $ExperiencedWorkers.size > 0$  then
7       if  $R > C$  then
8          $R \leftarrow R - C$ 
9          $VA \leftarrow VA \cup \langle ExperiencedWorkers[0], sel, C \rangle$ 
10    else if  $CloseWorkers.size > 0$  then
11      if  $p > 0.5$  then
12        if  $R > C$  then
13           $R \leftarrow R - C$ 
14           $VA \leftarrow VA \cup \langle CloseWorkers[0], sel, C \rangle$ 
15    else
16      Find closest worker to the task location  $w \leftarrow \text{closestWorker}(W, sel)$ 
17      reward  $r \leftarrow \text{serviceRate} * ((\text{dist}(w, sel.s_{1/2}) +$ 
17       $\text{dist}(sel.s_1, sel.s_2)) / \text{workerSpeed}) + C$ 
18      if  $R > r$  then
19         $R \leftarrow R - r$ 
20         $VA \leftarrow VA \cup \langle w, sel, r \rangle$ 
21 return  $VA$ 

```

---



(a) Triangle of SEL tasks

(b) Deduced label

Fig. C.4: Transitive Closure Example

## 7.2 Transitive Closure of Unresolved Grey Area Tasks

First, we check the unresolved grey area tasks with regard to the transitive closure property (Line 5 of Algorithm 20). Particularly, the triangles that contain the positive label tasks from the left tail (or from previous iterations) and the unresolved grey tasks are compared for the transitive closure. For instance, given a triangle of three SEL tasks with their respective labels,  $\langle \langle s_i, s_j, l^+ \rangle, \langle s_j, s_k, l^+ \rangle, \langle s_k, s_i, - \rangle \rangle$ , the label of the task  $\langle s_k, s_i, - \rangle$  can be deduced to the positive label  $l^+$ , by the transitive closure property (See Fig. C.4). The tasks labeled after checking the transitive closure property are removed from the unresolved task set (Lines 5 and 6 of Algorithm 20), and the remaining unresolved tasks are clustered in the next step. After inferring the clusters' labels based on the crowdsourced samples in the fourth step, we again check for the transitive closure property, including the resolved tasks in the previous iteration.

## 7.3 Clustering of Unresolved Grey Area Tasks

After filtering the tasks resolved by transitive closure property, we employ a density-based clustering algorithm, DBSCAN [7], for finding clusters consisting of similar pairs among the remaining unresolved tasks (Line 8 of Algorithm 20). We chose DBSCAN as it is robust to outliers and noise. The tasks that do not get assigned to a cluster (the noise) are labeled through crowdsourcing (See Crowdsourcing the grey area phase of Fig. C.1). Then, we pick a sample from each cluster for crowdsourcing, and the sample size is calculated by following the *Cochran formula* [25]. We consider 95 % confidence and 0.5 as the estimated proportion of the cluster tasks with positive labels. Based on the considerations, we calculate the sample size for a given cluster  $c$  as follows:

$$sampleSize = (384.16 + c.size) / (383.16 + c.size)$$

From the given cluster  $c$ , we select the *sampleSize* tasks based on the skyline. The smaller the skyline of the task  $k$ , the better the task is to be a positive label. We are less confident about their label for the tasks that fall into deeper skylines in the grey area. Consequently, we need to select tasks within deeper skyline levels for the crowdsourcing samples, thus leading to the next step of crowdsourcing the samples from the clusters.

## 7.4 Crowdsourcing the Cluster Samples

To crowdsource the selected samples from each cluster, we have to assign the sample tasks to workers and collect the worker's responses regarding the task's label (Line 11 of Algorithm 20). We limit the current discussion to workers' assignment of tasks and assume that the workers resolve the task and provide us with information about the task's label. However, given the limitation of workers' availability in SC [11]. Therefore, we have to maximize the number of tasks that can be solved with limited workers. Furthermore, given the workers' different constraints, the SC-server has to maximize the number of assignments. To proceed further, let us define a worker in SC:

**Definition 24.** (Worker): A worker, denoted by  $w$ , is a person willing to perform an assigned task by travelling to both the entities' locations. Worker  $w$  has the id of worker  $wID$ , a set of visited locations and the time of visit at the location  $\{[visitedLocation, visitTime], \dots\}$ .

$$w = \langle wID, \{[visitedLocation, visitTime], \dots\} \rangle$$

We believe that a worker who visited or was in the vicinity (50 metres) of the task's locations before will be better in completing the task than a worker who never visited them. We propose a task assignment algorithm to exploit the workers' past visit information and improve the quality of responses from the workers. Incidentally, we can also minimize the travel costs by assigning tasks to workers based on past visits, as the workers do not need to visit the task location to complete the task. To enable assignments based on the workers' past visits, we define the valid past visit-based task assignment set.

**Definition 25.** (Valid Past Visit-based Task Assignment Set): Given a set of workers  $W = \{w_1, w_2, \dots\}$  and a set of spatial entity matching tasks  $T = \{t_1, t_2, \dots\}$ , a Valid Past Visit-based Task Assignment (VPVTA), is a 3-tuple of form  $\langle w, t, r \rangle$ , where  $w$  is assigned to  $t$  with an associated reward  $r$ , and the worker should have previously visited either of the locations of the task's spatial entities.

$$t.l_1/t.l_2 \in \{w.visitedLocation_1, w.visitedLocation_2, \dots\}$$

We propose a task assignment algorithm (See Algorithm 21) to maximize the number of past visit-based task assignments to get more accurate answers. However, for a given task *sel* if there is no possibility of a past visit-based task assignment, we will try to find if any worker was in the vicinity of the task (around 50 meters) during their past visits. If yes, then the task will be assigned to the worker as a close worker visit task assignment with a 50% chance (Lines 5 and 10-14 of Algorithm 21). Otherwise, the task will be assigned to the closest worker based on the current location. After the assignment of the sample tasks from the clusters, the crowdsourced labels of the sample tasks are collected and analyzed for inferring the cluster labels.

## 7.5 Inferring the Cluster Labels based on Crowdsourced Samples

After collecting crowdsourced labels of the samples from each cluster, we calculate the ratio of positive labels in the sample tasks and the corresponding entropy (Lines 14 and 15 of Algorithm 20). Given an entropy threshold, we decide whether the cluster can be labeled based on the entropy value and the ratio of the crowdsourced sample's positive labels. We assume that the cluster's majority label would be same as the crowdsourced sample's majority label. If the entropy is less than the given entropy threshold, all the cluster's remaining tasks will be labeled based on the majority label (Lines 16 - 22 of Algorithm 20) of the sampled tasks, i.e., if the majority are positive, then the remaining tasks in the cluster will be labelled positive. Similarly, if most of the sampled tasks are negative, then the remaining tasks in the cluster will be labeled negative.

However, if the entropy is greater than the entropy threshold, then the cluster's remaining tasks will not be labelled and will be checked for transitive closures based on the crowdsourced samples. If the clusters' remaining tasks are not labeled by the transitive closure either, then all the unresolved tasks of all the clusters will be reclustered after removing the crowdsourced samples and inferred tasks. Consequently, the reclustered tasks will be sampled, crowdsourced, inferred, and reclustered until the termination condition of budget exhaustion or zero unresolved tasks is reached (See Crowdsourcing the grey area phase of Fig. C.1).

## 8 Experimental Analysis

In this section, we perform experiments on a real dataset, collected as in [12]. The dataset had 75,541 spatial entities, which originate from four sources, 51.50% from Google Places, 46.22% from Krak ([www.krak.dk](http://www.krak.dk)), 0.03% from Foursquare, and 2.23% from Yelp. The pairs of spatial entities which are at



## 8. Experimental Analysis

most 50 m apart are compared pairwise and labeled with the *SkyEx* algorithm [13]. Thus, we obtain 293,833 labeled pairs of spatial entities. In order to improve the quality of the labeling, we identify the grey area, which consists of 33,155 pairs. In the experimental evaluation, we will try to improve the grey area’s labelling by using the SC-based active learning approach. We utilize the check-ins dataset from [12] for simulating the workers dataset. The workers dataset has 19980 workers available for assignment. For simplicity purposes, we assume the reward for each task as 1, irrespective of the distance travelled by workers for completing the task. Therefore, a reward budget of 33155 would resolve 33155 SEL tasks through crowdsourcing.

### 8.1 Performance of Skycrowd solution

We evaluate our proposed skycrowd solution in terms of *precision*, *recall* and *F-measure*, where  $F\text{-measure (F1)} = 2 \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$ . The replies from crowdsourcing are simulated as if consulting an oracle and revealing the label. The skycrowd solution is compared against the results of crowdsourcing all the 293,833 tasks (Full), automatic labeling of all the tasks (Automatic), and crowdsourcing all the 33,155 tasks in the grey area (Hybrid-Full). Furthermore, we evaluate the effect of varying the budget and varying the entropy threshold on the *F-measure*, the average distance travelled by worker, and the number of inferred tasks. The experiment parameters are listed in Table C.1. The default values are highlighted in bold.

Parameters	Value Range
entropy Threshold	0.05, 0.10, <b>0.15</b> , 0.20, 0.25
DBSCAN: parameters	Minimum Cluster:10 Minimum Distance: 0.02
Number of workers	19980
Reward Budget	3315(10%), 6630 (20%), 9945(30%), 13260(40%), <b>16575 (50%)</b> , 19890(60%), 23205 (70%), 26520 (80%), 29835 (90%), 33155 (100%)

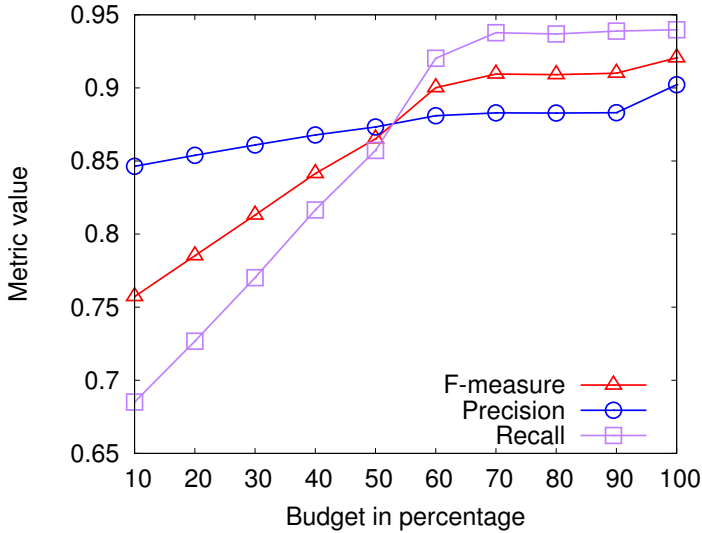
Table C.1: Experiment Parameters

**Skycrowd vs Automatic vs Hybrid Full vs Full Crowdsourcing** In this experiment, we compare the budget consumption and the results achieved by the different solutions, *Automatic*, *Skycrowd*, *Hybrid-full*, and *Full crowdsourcing*. The results are presented in Table C.2. The automatic labeling method

	Automatic	Skycrowd	Hybrid Full	Full
<b>Precision</b>	0.86	0.88	0.90	1
<b>Recall</b>	0.61	0.94	0.94	1
<b>F-measure</b>	0.71	0.91	0.92	1
<b>Budget</b>	0	22100	33,155	293,833

**Table C.2:** Comparing Automatic vs Skycrowd vs Hybrid-Full vs Full Crowdsourcing in terms of budget and different metrics

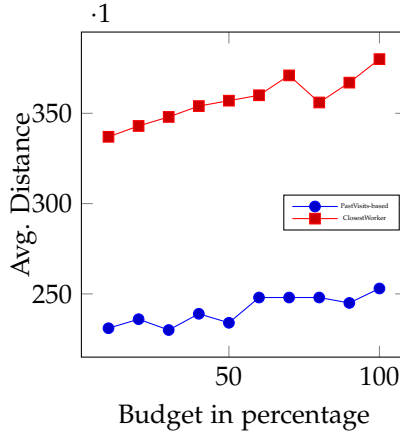
(no crowdsourcing) achieves satisfactory values of precision but only 0.61 of recall, and thus *F-measure* of 0.72. If we spend a budget of 33,155 (11% of the full crowdsourcing) for crowdsourcing all the tasks in the grey area (Hybrid-full), *F-measure* reaches 0.92, *precision* 0.9 and *recall* 0.94. However, *Skycrowd* achieves similar results (*F-measure* of 0.91, *precision* 0.88 and *recall* 0.94.) as *Hybrid-full* with 64% of the *Hybrid-full* budget and 7% of the *Full* crowdsourcing budget. It is worth remarking here that, when compared with *Skycrowd* the extra budget for *Hybrid-full* does not improve the *recall*. In fact, the extra budget can improve the *precision* to 0.9, which is not essential when we have already 0.88 with *Skycrowd*.



**Fig. C.5:** Effect of varying reward budget for Skycrowd on the F-measure, precision, and recall

**Effect of varying reward budget on F-measure and Average Distance** In this experiment, we evaluate the effect of varying reward budget on the results like *F-measure*, *precision*, and *recall* (see Fig. C.5). It can be noted that the recall and F-measure values increased sharply at the beginning as the reward

## 8. Experimental Analysis



(a)

**Fig. C.6:** Effect of varying reward budget on the average distance travelled by worker per assignment

budget increases and stabilized around the 64% budget mark. Similarly, precision increased from 0.84 at 10% budget to 0.882 at 64 % budget. At the 64% reward budget, the Skycrowd solution resolves all the SEL tasks in the grey area by inferring the labels through the SC-based active learning approach. The negligible increase in the F-measure, precision and recall values after the stabilization point highlights the effectiveness of the inference methods utilized by the *Skycrowd* solution.

Similarly, we evaluate the effect of varying reward budget on the average distance travelled by a worker per assignment (See Fig. C.6a). We compared the past visits-based task assignment algorithm of the Skycrowd with a baseline, ClosestWorker task assignment algorithm that assigns the task to the closest worker. The past visits-based task assignment algorithm outperforms ClosestWorker task assignment algorithm by at least 30 %. The reason is that in the past visits-based task assignment algorithm, some of the assignments based on past visits do not need to visit the spatial entity task location. Furthermore, a general trend can be observed that as the reward budget increases the average distance travelled by the worker increases. The trend can be attributed to the selection of tasks in the clusters for crowdsourcing. As more reward budget is available, farther tasks to the workers will be crowdsourced to improve the cluster sample’s entropy for labeling.

**Effect of varying entropy threshold on F-measure and inferred tasks** In this experiment, we evaluate the effect of varying entropy threshold with 50% reward budget on F-measure and the number of tasks resolved by inference. We can notice a sharp jump in the F-measure value (see Fig. C.7a)

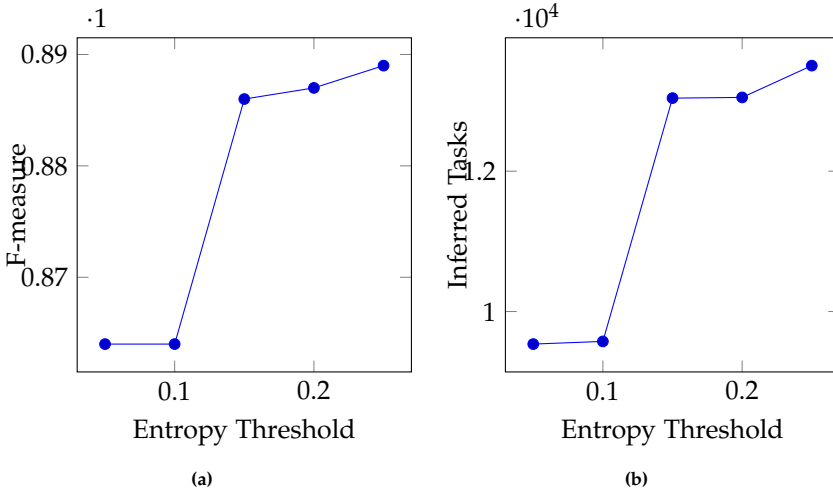


Fig. C.7: Effect of varying entropy threshold value on :a. F-measure. b. Tasks resolved by Inference.

when the entropy threshold value increases from 0.10 to 0.15. Similarly, there is a sharp increase in the number of tasks resolved by inference when the entropy threshold value increases from 0.10 to 0.15. Consequently, the entropy threshold value of 0.15 is considered for the experiments based on this dataset.

**Summary** The *Skycrowd* solution significantly improves the results of the automatic labeling, from an *F-measure* of 0.71 to 0.91 and the *Hybrid-full* improves it to 0.92. However, the trade-off regarding budget and *F-measure* achieved by *Skycrowd* is more profitable than for crowdsourcing all the grey area because it uses 64% of its budget to achieve similar *F-measure*. The *Skycrowd* solution achieves a *F-measure* of 0.91 for just 7% of the full crowdsourcing budget. The active learning-aided inference methods used in the *Skycrowd* solution are proven effective since there is a negligible increase in *F-measure* values with reward budgets greater than 64%.

## 9 Conclusions and Future Work

In this paper, we aim to resolve the spatial entity linkage problem utilizing the concept of machine learning-aided spatial crowdsourcing. To achieve this objective, we defined the *Spatial Entity Linkage* tasks and the *Spatial Entity Linkage with the aid of Spatial Crowdsourcing* optimization problem to maximize the *F-measure* value of the resultant labeled tasks given a reward budget. We proposed a hybrid *Skycrowd* solution that employs skyline-based automatic

labeling and machine learning-aided spatial crowdsourcing techniques to resolve this problem. The *Skycrowd* solution consists of four steps: a) extracting SEL tasks from a given set of entities, b) automatic labeling of SEL tasks using SkyEx algorithm [13], c) identifying the set of tasks that are potentially mislabeled by the automatic labeling method, and d) crowdsourcing the tasks that are potentially mislabeled using an SC-based active learning approach.

The proposed *Skycrowd* solution provides an *F-measure* of 0.91 for a fraction (7%) of the full crowdsourcing budget (if all SEL tasks are crowdsourced). The *Skycrowd* solution achieves 30% higher *F-measure* value than the automatic labeling solution. Furthermore, for 64% of the grey area budget, the SC-based active learning approach achieves an *F-measure* value of 0.91 against the 0.92 achieved by crowdsourcing the entire grey area.

There are several promising directions for future work. First, there is a need to compare the clustering technique's efficacy (DBSCAN) used in the SC-based active learning approach against other clustering techniques like Kmeans. Second, the current task assignment algorithm assumes that all the workers are available for assignment, i.e., the offline task assignment strategy. The task assignment algorithm needs to be improved to accommodate workers' dynamic arrival and availability constraints. Furthermore, task scheduling needs to be considered while assigning multiple tasks to a single worker. Third, the *Skycrowd* solution needs to be tested with crowdsourced responses from real workers to fine-tune the solution for practical implementation issues.

## Acknowledgements

*This research has been funded in part by the European Commission through the Erasmus Mundus Joint Doctorate Information Technologies for Business Intelligence - Doctoral College (IT4BI-DC). Xike Xie is supported by the CAS Pioneer Hundred Talents Program, Fundamental Research Funds for the Central Universities, National Natural Science Foundation of China (No. 61772492), and Natural Science Foundation of Jiangsu Province (No. BK20171240).*

## References

- [1] R. Abdalla, "Geospatial data integration," in *GeoICT*, 2016.
- [2] S. Balley, C. Parent, and S. Spaccapietra, "Modelling geographic data with multiple representations," *IJGIS*, 2004.
- [3] B. Berjawi, E. Chesneau, F. Duchateau, F. Favetta, C. Cunty, M. Miquel, and R. Laurini, "Representing uncertainty in visual integration." in *DMS*, 2014.
- [4] G. E. Box and R. D. Meyer, "An analysis for unreplicated fractional factorials," *Technometrics*, vol. 28, no. 1, pp. 11–18, 1986.

## References

- [5] Y. Censor, "Pareto optimality in multiobjective problems," *Applied Mathematics and Optimization*, 1977.
- [6] C. Chai, G. Li, J. Li, D. Deng, and J. Feng, "Cost-effective crowdsourced entity resolution: A partial-order approach," in *Proceedings of the 2016 International Conference on Management of Data*. ACM, 2016, pp. 969–984.
- [7] M. Ester, H. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *KDD*, 1996.
- [8] C. Fellbaum, "Wordnet," in *Theory and applications of ontology: computer applications*, 2010.
- [9] D. Firmani, B. Saha, and D. Srivastava, "Online entity resolution using an oracle," *Proceedings of the VLDB Endowment*, vol. 9, no. 5, pp. 384–395, 2016.
- [10] C. Gokhale, S. Das, A. Doan, J. F. Naughton, N. Rampalli, J. Shavlik, and X. Zhu, "Corleone: hands-off crowdsourcing for entity matching," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 2014, pp. 601–612.
- [11] S. R. B. Gummidi, X. Xie, and T. B. Pedersen, "A survey of spatial crowdsourcing," *TODS*, vol. 44, no. 2, p. 8, 2019.
- [12] S. Isaj and T. B. Pedersen, "Seed-driven geo-social data extraction," in *SSTD '19*. ACM, 2019, pp. 11–20.
- [13] S. Isaj, E. Zimányi, and T. B. Pedersen, "Multi-source spatial entity linkage," in *Proceedings of the 16th International Symposium on Spatial and Temporal Databases*, 2019, pp. 1–10.
- [14] R. Karam, F. Favetta, R. Kilany, and R. Laurini, "Integration of similar location based services proposed by several providers," in *NDT*, 2010.
- [15] V. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, 1966.
- [16] A. Morana, T. Morel, B. Berjawi, and F. Duchateau, "Geobench: a geospatial integration tool for building a spatial entity matching benchmark," in *ACM SIGSPATIAL*, 2014.
- [17] G. Papadakis, J. Svirsky, A. Gal, and T. Palpanas, "Comparative analysis of approximate blocking techniques for entity resolution," *PVLDB*, 2016.
- [18] J. Schiller and A. Voisard, *Location-based services*. Elsevier, 2004.
- [19] V. Sehgal, L. Getoor, and P. D. Viechnicki, "Entity resolution in geospatial data integration," in *GIS*, 2006.
- [20] P. Tabarro, J. Pouliot, R. Fortier, and L. Losier, "A webgis to support gpr 3d data acquisition: A first step for the integration of underground utility networks in 3d city models," *ISPRS*, 2017.
- [21] V. Verroios and H. Garcia-Molina, "Entity resolution with crowd errors," in *2015 IEEE 31st International Conference on Data Engineering*. IEEE, 2015, pp. 219–230.
- [22] N. Vesdapunt, K. Bellare, and N. Dalvi, "Crowdsourcing algorithms for entity resolution," *Proceedings of the VLDB Endowment*, vol. 7, no. 12, pp. 1071–1082, 2014.

## References

- [23] V. Walter and D. Fritsch, "Matching spatial data sets: a statistical approach," *IJGIS*, 1999.
- [24] J. Wang, T. Kraska, M. J. Franklin, and J. Feng, "Crowder: Crowdsourcing entity resolution," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1483–1494, 2012.
- [25] R. F. Woolson, J. A. Bean, and P. B. Rojas, "Sample size for case-control studies using cochrane's statistic," *Biometrics*, pp. 927–932, 1986.
- [26] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *ACL*, 1994.

## References



# Paper D

## Transit-based Task Assignment in Spatial Crowdsourcing

Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach  
Pedersen, and Xike Xie

The paper has been published in the  
*SSDBM 2020: 32nd International Conference on Scientific and Statistical Database  
Management*  
Article 13, 1–12, 2020. DOI:<https://doi.org/10.1145/3400903.3400929>

## Abstract

*Worker movement information can help the spatial crowdsourcing platform to identify the right time to assign a task to a worker for successful completion of the task. However, the majority of the current assignment strategies do not consider worker movement information. This paper aims to utilize the worker movement information via transits in an online task assignment setting. The idea is to harness the waiting periods at different transit stops in a worker transit route (WTR) for performing the tasks. Given the limited availability of workers' waiting periods at transit stops, task deadlines and workers' preference of performing tasks with higher rewards, we define the Transit-based Task Assignment (TTA) problem. The objective of the TTA problem is to maximize the average worker rewards for motivating workers, considering the fixed worker transit models. We solve the TTA problem by considering three variants, step-by-step, from offline to batch-based online versions. The first variant is the offline version of the TTA, which can be reduced to a maximum bipartite matching problem, and be leveraged for the second variant. The second variant is the batch-based online version of the TTA, for which, we propose dividing each batch into an offline version of the TTA problem, along with additional credibility constraints to ensure a certain level of worker response quality. The third variant is the extension of the batch-based online version of the TTA (Flexible-TTA) that relaxes the strict nature of the WTR model and assumes that a task with higher reward than a worker-defined threshold value will convince the worker to stay longer at the transit stop. Through our extensive evaluation, we observe that the algorithm solving the Flexible-TTA problem outperforms the algorithms proposed to solve other variants of the TTA problems, by 55% in terms of the number of assigned tasks, and by at least 35% in terms of average reward for the worker. With respect to the baseline (online task assignment) algorithm, the algorithm solving the Flexible-TTA problem results in three times higher reward and at least three times faster runtime.*

© 2019 ACM. Reprinted, with permission from Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach Pedersen, and Xike Xie. Transit-based Task Assignment in Spatial Crowdsourcing. In: *SSDBM 2020: 32nd International Conference on Scientific and Statistical Database Management*, Article 13, 1–12, 2020. <https://doi.org/10.1145/3400903.3400929>

*The layout has been revised.*

# 1 Introduction

Most of the existing spatial crowdsourcing (SC) assignment strategies do not consider the movement of workers in the spatio-temporal dimensions and assign tasks based on workers' current/ reported locations [8]. Assigning tasks to a worker at the right time is critical to the success of an SC application. Workers' movement information help us identify the right time for assigning a task to the worker. For example, consider the transit movement information of a worker. Intuitively, a worker following her daily transit route can perform tasks at the transit stops in the route, namely the origin stop, the intermediate stops (if any), and the destination.

This paper aims to target a new group of workers for spatial crowdsourcing, namely passengers in public transport, by harnessing their waiting periods at different transit stops in a regular worker transit route to offer an alternative strategy for the online task assignment in SC. Given the constraints of transport, the worker will try to strictly adhere to the transit route without any deviation or delay. Consequently, a task can only be assigned at a transit stop if the travel for performing the task does not affect the transit route of the worker, i.e., the worker will not miss the bus at the transit stop or be late to the destination. Among all the reachable tasks near a transit stop, the worker would like to choose the task with maximum reward to maximize her earnings. We assume that the SC-server allows the individual workers to register their transit routes or upload their daily travel data in exchange for better maximization of their reward calculation from the SC-server. **Example**

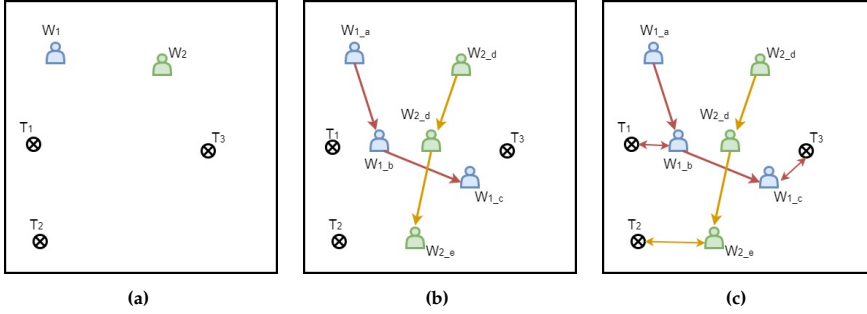


Fig. D.1: Transit-based Task Assignment example.

1: Consider the example in Fig.D.1. There are two workers  $W_1$  and  $W_2$  and three noise data collection tasks  $T_1$ ,  $T_2$ , and  $T_3$  (See Fig. D.1a). The transit routes of  $W_1$  and  $W_2$  before and after assignment can be seen in Fig. D.1b & D.1c, respectively. The worker travels to the assigned task from the transit stop and returns to the transit stop after recording the noise levels at the task location with her smartphone. For instance, worker  $W_1$  travels from  $W_{1,b}$  to  $T_1$  and returns back to  $W_{1,b}$  after performing the task to continue with the

transit route.

To summarize, there is a need to develop new algorithms to solve the Transit-based task assignment (TTA) problem for harnessing the waiting periods at different transit stops in a worker transit route. Moreover, workers' preferences should be considered for improving the number of successful assignments. To improve the quality of the task responses, we consider the worker credibility scores and employ a minimum worker credibility threshold constraint on tasks. Additionally, to maximize the workers' reward, we also model the case of flexible transit route which relaxes the strict nature of the worker transit model and assumes that a task with higher reward than a worker-defined threshold value will convince the worker to stay longer at the transit stop.. In this case, we assume that the routes are flexible, and she is willing to spend more time at a transit stop if the reward is high enough.

The TTA problem is resolved by considering different input models, offline and batch-based online versions. The offline version of the TTA can be reduced to a maximum bipartite matching problem, and be leveraged for the batch-based online input version. The batch-based online version of the TTA is solved by dividing each batch into an offline version of the TTA problem, along with additional credibility constraints to ensure a certain level of worker response quality. Furthermore, the batch-based online version of the TTA is extended (Flexible-TTA) to facilitate relaxation of the strict nature of the WTR model with an assumption that a task with a higher reward than a worker-defined threshold value will convince the worker to stay longer at the transit stop. We study the three versions of the TTA problem and propose algorithms to solve them.

The main contributions offered in this paper are:

- We present algorithms based on the Server-Assigned Task mode [9], to improve the task assignments by exploiting the workers' transit route information.
- We formulate the Transit-based Task Assignment (TTA) problem, that aims to maximize the average net reward received by a set of workers, considering transit stop time and deadline constraints.
- We prove that the offline version of TTA problem is reducible to the Maximum-weighted Bipartite Matching problem.
- We further study the online batch-based versions of the TTA using the offline version and propose algorithms to solve the problem.
- Additionally, we propose the *Credible Transit-based Task Assignment* algorithm to harness the worker credibility information to ensure a desired

level of quality in the responses.

- We formulate the Flexible-TTA problem, that extends the TTA problem by allowing changes to the worker routes based on their threshold reward and maximum travel time constraints, and propose an algorithm to solve it.
- We test the applicability of the proposed algorithms through an extensive experimental evaluation based on the simulated worker transit routes and tasks in Aalborg, Denmark.

The remainder of the paper is organized as follows. In Section 2, we discuss a set of preliminaries in the context of transit-based SC. In Section 3, we formally define the offline and online batch-based versions of the TTA problem and explain our assignment algorithms for solving the batch-based TTA problem. Additionally, in the same section, we describe the *Credible Transit-based Task Assignment* algorithm in detail. Thereafter, in Section 4 we formally define the offline and online batch-based versions of the *Flexible-TTA* problem and explain our assignment algorithm for solving the batch-based Flexible-TTA problem. Section 5 presents the experimental results. In Section 6, we review the related work. Finally, in Section 7 we conclude and discuss the future directions of this study.

## 2 Preliminaries

This section introduces some basic concepts that will be used throughout this paper. First, we define the worker with their worker transit routes.

**Definition 26.** (Worker): A worker, denoted by  $w$ , is a person willing to perform an assigned task by travelling to the task's location. Worker  $w$  has a transit stop set  $WTS$  that contains worker transit stops  $wts$ , belonging to the transit route she follows every day, threshold reward  $thresRew$  representing the expected compensation for not following the fixed transit route  $wr$ ,  $strtTime$  represents the start of the transit trip,  $maxTrvlTime$  represents the total time the worker is willing to spend on her daily route  $wr$ ,  $servRate$  represents the service price charged by the worker  $w$  per hour, and  $credibility$  represents the worker credibility score. A worker is defined as:

$$w = \langle WTS, strtTime, thresRew, maxTrvlTime, servRate, credibility \rangle$$

$$WTS = \{wts_0, \dots, wts_i, \dots, wts_d\}$$

, where  $wts_0$  represents the origin transit stop,  $wts_d$  represents the destination transit stop, and  $wts_i$  represents an intermediate transit stop of the worker transit route.

We define the credibility of the worker as the probability of worker performing an assigned task correctly. The credibility of the worker is defined similarly as the reputation score in [10]. The credibility of a worker can be determined based on the historical information of workers' answers stored in the SC-server. We assume that the worker credibility scores are stored and maintained at the SC-server.

Furthermore, we defined the worker route as a series of sequential worker transit stops. As mentioned earlier, our intuition is that the worker can perform tasks during their waiting period at the worker transit stops. Worker transit stops are associated with the real-world public transportation stop at a certain geographical location. Accordingly, we define worker transit stop as:

**Definition 27.** (Worker Transit Stop): A worker transit stop, denoted by  $wts$ , is at location  $l$ , and has  $arrivalTime$  and  $departure - Time$ , that represent the arrival and departure timings at the transit stop of the worker  $w$ .  $assignedTask$  represents the task, if assigned to the transit stop. The worker transit stop is defined as:

$$wts = \langle l, arrivalTime, departureTime, w, assignedTask \rangle$$

We modeled the arrival time at the origin transit stop as the  $strtTime$  of the worker  $w$ , and the departure time at the destination transit stop as the  $strtTime + maxTrolTime$  of the worker  $w$ .

We define a spatial task as a task associated with a geographical location. The spatial task definition is inspired by [9].

**Definition 28.** (Spatial task): A *spatial task*, denoted by  $t$ , contains a query  $q$  to be answered at location  $l$ . The query is asked at time  $issueTime$  and will expire at time  $expiryTime$ . The task takes  $taskDuration$  time to complete. The task will be guaranteed to result in a correct response, if a worker with at least  $minWorkerCred$  credibility is assigned to the task.

$$t = \langle q, l, issueTime, expiryTime, taskDuration, minWorkerCred \rangle$$

The worker has to visit location  $l$  to perform the task during the time interval between issue time and expiry time. Note that the worker has to visit the task location at least  $taskDuration$  minutes before the  $expiryTime$ . For simplicity, we assume that the worker can complete the task with a single response. The task can still be assigned to the worker with less credibility than  $minWorkerCred$ . However, then the quality of the response is not guaranteed.

In [1], it is mentioned that tasks with "extrinsic incentives" like monetary reward would attract more workers, and affects the speed of accomplishing the task. We offer monetary rewards to workers for accomplishing the task. However, instead of a fixed reward per task, we define the reward associated

### 3. Transit-based Task Assignment

with spatial tasks in proportion to the time spent by the worker to perform the task. We assume that there will be a base reward for performing the task. In addition, we assume that the workers expect a fixed hourly payment rate as compensation for the time spent on performing the task. The time spent is calculated based on the time taken to reach the task location from the worker transit stop, the time taken to do the task, and the time taken to return to the transit stop.

**Definition 29.** (Reward): The worker  $w$  will receive reward  $r(w, t)$  after the completion of task  $t$  at transit stop  $w.wts$ . We assume that the reward  $r$  is affinely dependent on the distance between the transit stop's location  $w.wts.l$  and the task's location  $t.l$ , and the task duration  $t.taskDuration$ .

$$r(w, t) = w.servRate * (2 * dist(w.wst.l, t.l) / walkingSpeed + t.taskDuration) + c$$

, where  $c$  is the fixed reward for all the tasks, for example 10 Kroners,  $serviceRate$  is the fixed hourly compensation rate charged by the worker  $w$ , for example 60 Kroners per hour, and  $walkingSpeed$  is the average walking speed of workers ( $m/s$ ). For simplicity, we have assumed all the workers to have the same service rate.

**Definition 30.** (Distance from transit stop to task):

$dist(w.wst.l, t.l)$  denotes the distance that a worker  $w$  at a transit stop  $w.wts$  needs to travel to reach  $t$ . Generally speaking, the distance from a transit stop to task denotes the walking distance from the worker transit stop to the task location.

For simplicity, we assume that a worker would perform at most a single task at every transit stop. Intuitively, a worker  $w$  cannot accept all the tasks without considering the additional travel time. Therefore,  $maxTrolTime$  is used to limit the amount of time a worker will spend on completing the transit route. The travel time is calculated based on the transit network. The transit info can be reliably extracted from the public transport web services and *Google Maps*. After the worker makes her task inquiry, the SC-server would then try to assign the tasks according to the worker and update her route.

## 3 Transit-based Task Assignment

### 3.1 Problem Definition

Given the different constraints, the objective of the SC-server is to maximize the net reward for the individual workers, received through performing assigned tasks. We consider two different input models for the TTA problem:

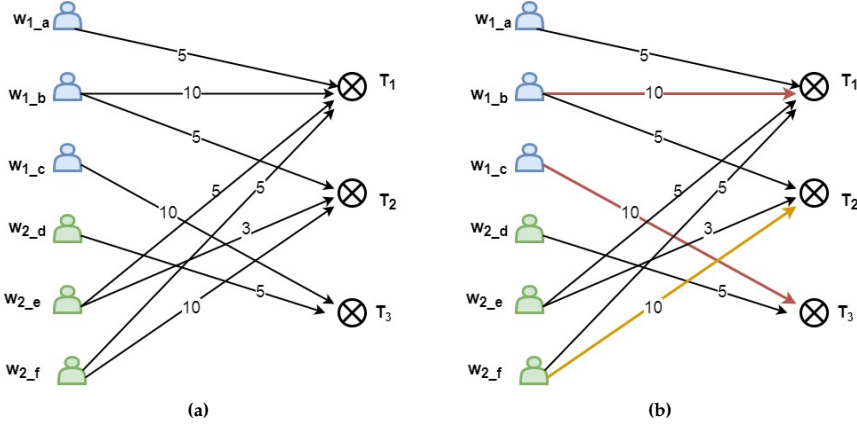


Fig. D.2: Reduction of Example 1 to MWBM problem

offline and batch-based. In the offline model, all the tasks and the workers along with their transit stops will be known beforehand to the SC-server. However, in the batch-based input model, the SC-server performs the task assignment for every incoming batch at regular time intervals. In the batch-based input model, the unassigned tasks along with the worker transit stops will be added to the next batch. For every new batch of workers and tasks, the SC-server tries to assign the newly available transit stops and tasks along with the unassigned and available, transit stops and tasks from the previous batch. Furthermore, in the TTA problem, **the worker transit route is considered fixed and will not be changed**. Considering these objectives & the input models, we define the offline TTA problem as:

**Definition 31.** (Offline Transit-based Task Assignment Problem): Assume an input set of tasks  $T$  and a set of workers  $W$  along with their set of worker transit stops  $WTS$ . The offline Transit-based Task Assignment (Offline-TTA) problem is an optimization problem with the objective to maximize the sum of the new rewards received by the individual workers. The Offline-TTA problem finds an optimal Transit Stop Task Assignment set, denoted by  $TA(WST, T)$ , with average net reward of  $r_{TA}$ , is a set of 3-tuples of the form  $\langle wts, t, r \rangle$ , where  $wts$  is assigned to  $t$  with an associated reward  $r$ , given the following constraints are satisfied:

- **Transit Stop Time Constraint:** The time required to complete the task, i.e., travel time from the transit stop to the task location and returning to the transit stop and the *taskDuration* is less than the time spent at the transit stop, i.e.,

$$(2 * \text{distance}(wts.l, t.l) / \text{walkingSpeed}) + t.\text{taskDuration} < (wts.\text{departureTime} - wts.\text{arrivalTime})$$



### 3. Transit-based Task Assignment

- **Task Deadline Constraint:** The worker's arrival time at the transit stop should be at least  $t.taskDuration$  before the task deadline, i.e.,

$$t.expiryTime \geq wts.arrivalTime + t.taskDuration$$

With the following theorem, we can solve the offline-TTA problem by reducing it to the maximum weighted bipartite matching problem.

**Theorem 1.** The offline-TTA problem is reducible to the maximum weighted bipartite matching(MWBM) problem.

*Proof.* We prove the theorem for a set of workers  $W$  with an associated set of worker transit stops  $WTS = \{wts_1, wts_2, \dots\}$  and a set of tasks  $T = \{t_1, t_2, \dots\}$ . Let  $G = (V, E)$  be an undirected graph whose vertices can be partitioned as  $V = WTS \cup T$ , where each transit stop  $wts_i$  maps to a vertex in  $WTS$  and each task  $t_j$  maps to a vertex in  $T$ . If the *task deadline* and *transit stop time constraints* are satisfied between  $wts_i$  and  $t_i$ , then there is an edge  $e_{i,j} \in E$  connecting the vertex  $wts_i$  in  $WTS$  and vertex  $t_i$  in  $T$ . As every edge  $e_{i,j} \in E$  has one end in  $W$  and another end in  $T$ , the graph  $G$  is a bipartite graph. We set the edge weight for every edge  $e_{i,j} \in E$  to the reward  $r$  associated with task  $t$  and the worker transit stop  $wts$ . Since, a worker can perform only one task at each transit stop,  $\langle wts_i, t_j \rangle$  is a valid match only if both  $wts_i$  and  $t_j$  appear in at most one edge in  $E$ . Finally, the offline-TTA problem is solved by finding the maximum matching in weighted bipartite graph  $G$ .  $\square$

Fig. D.2a depicts the bipartite graph  $G$  with weights for the example mentioned in Section 1. The left side set consists of the workers' transit points as nodes and the right side set contains the tasks as nodes. Fig D.2b depicts the maximum weighted bipartite graph with maximum weighted edges highlighted.

**Definition 32.** (Online Batch-based Transit-based Task Assignment problem): Assume a set of batches  $B$ , with each incoming batch  $b \in B$  comprising a set of unassigned tasks  $T_b$  and available workers  $W_b$  along with their transit routes  $WTS_b$ . The Online Batch-based Transit-based Task Assignment (Batch-TTA) problem is an optimization problem with the goal of finding an offline Transit Stop Task Assignment set  $TA(WTS_b, T_b)$  that maximizes the net rewards( $r_{TA}$ ) received by the individual workers  $w \in W_b$  at their worker transit stops  $wts \in WTS_b$  by performing assigned tasks  $t \in T_b$  for each incoming batch  $b$ .

We propose two algorithms to solve the Batch-TTA problem, namely, the Maximum Weighted Bipartite Matching-based(MWBM) and the Minimum Distance based Direct Assignment (DA) algorithms. The proposed algorithms follow a locally optimal assignment strategy as the SC-server has

Task	Issue Time	Expiry Time	Transit Stop	Reward	Worker Credibility	Threshold
$t_1$	8:01 AM	5:00 PM	$wts_1$	20	0.7	
$t_2$	8:15 AM	5:00 PM	$wts_1$	25	0.7	
$t_3$	8:30 AM	5:00 PM	-	10	0.6	
$t_4$	9:00 AM	5:00 PM	$wts_3$	20	0.6	

Table D.1: Example 2: Tasks and associated transit stops

Stop	Arrival	Departure	Credibility	Threshold Reward
$wts_1$	8:00 AM	8:20 AM	0.6	30
$wts_2$	8:40 AM	9:00 AM	0.6	30
$wts_3$	9:20 AM	9:40 AM	0.6	30

Table D.2: Example 2: Transit Stops

minimum knowledge of the availability of new workers and tasks in the subsequent batches [9]. The algorithms will be explained through the help of the following running example:

**Example 2:** Consider the scenario in Fig. D.3a, where the worker  $w$  follows a transit route with three transit stops ( $wts_1, wts_2, wts_3$ ) (The schedule, credibility scores and threshold reward values are mentioned in Table D.2). There are four tasks sent to the SC-server (details are in Table D.1). It can be noticed that at stop  $wts_1$ , two tasks ( $t_1, t_2$ ) satisfy the travel stop time constraint. However, task  $t_1$  is issued before task  $t_2$ . Similarly, it can be noticed that none of the tasks satisfy the travel stop time constraint at transit stop  $wts_2$ . In the following subsections, we will observe how different task assignment algorithms would result in different assignments.

### 3.2 Maximum Weighted Bipartite Matching (MWBM) Algorithm

In MWBM algorithm, we solve the *Batch-based TTA* problem by dividing it into individual *offline-TTA* problems for each incoming batch of available workers and unassigned tasks. Thus, we can solve the batch-based TTA by solving the individual *offline-TTA* problems for each incoming batch. (See Algorithm 22). According to Theorem 1, the *offline-TTA* problem can be reduced to a maximum weighted bipartite matching problem. Therefore, we can employ the maximum weight bipartite matching algorithm [11] to solve

### 3. Transit-based Task Assignment

---

**ALGORITHM 22: MAX. WEIGHTED BIPARTITE MATCHING**


---

**Input:** An incoming batch  $b$  consisting of a non-empty set of workers  $W_b$  with associated set of worker transit stops  $WTS_b$  with  $thresRew$  as the minimum Threshold reward for flexible transits and  $maxTrvlTime$  as maximum travel time, a set of tasks  $T_b$ .  $TA(WTS, T)$  is the optimal set of assignments before the batch  $b$ .  $c$  is the fixed reward per task.

**Output:** The Optimal set of assignments  $TA(WTS \cup WTS_b, T \cup T_b)$  with the average Reward  $r_{TA}$  and minimum travel distance  $minDist$  for batch  $b$

```

1   $TA(WTS \cup WTS_b, T \cup T_b) \leftarrow NULL;$ 
2   $TA(WTS_b, T_b) \leftarrow NULL;$ 
3   $WeightedGraph\ G \leftarrow NULL;$ 
4  foreach  $t \in T_b$  do
5     $G.addVertex(t);$ 
6  foreach  $wts \in WTS_b$  do
7     $G.addVertex(wts);$ 
8    foreach  $t \in T_b$  do
9       $maxDistAtStop \leftarrow walkingSpeed * (wts.departureTime -$ 
10          $wts.ArrivalTime - t.taskDuration) / 2;$ 
11      if  $dist(wst.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$  then
12         $edgeWeight \leftarrow r(w, t);$ 
13         $G.addEdge(wts, t, edgeWeight);$ 
14   $TA(WTS_b, T_b) \leftarrow MWBM.getMatching(G, WTS_b, T_b);$ 
15   $TA(WTS \cup WTS_b, T \cup T_b) \leftarrow TA(WTS, T) \cup TA(WTS_b, T_b);$ 
16   $removeAssignedAndExpiredTasks(T \cup T_b);$ 
17   $removeAssignedWorkerStps(WTS \cup WTS_b);$ 
18  return  $TA(WTS \cup WTS_b, T \cup T_b)$ 

```

---

the *offline-TTA* problem. *MWBM* tries to solve the *batch-based TTA* by setting the task reward as the edge weight.

*MWBM* constructs the bipartite graph  $G$  and establishes edges between worker transit stop vertexes and tasks vertexes if they satisfy the task deadline and transit stop time constraints. Therefore for each worker transit stop, it has to search for all the tasks that satisfy the transit stop time and task deadline constraints to create all the potential edges. Consequently, the time complexity of this algorithm is directly dependent on the number of vertices  $n (|WTS_b| + |T_b|)$ , the number of edges  $m$  and the batches  $p$  along with the time complexity of the maximum weight bipartite matching algorithm, i.e.,  $O(n(m + n \log n) * p)$  [11].

Consider *Example 2*, *MWBM* constructs the bipartite graph with all the potential edges between the worker transit stops and the tasks. *MWBM* tries to find the edge with maximum weight for assigning tasks to transit stops (See Fig. 24b, where red edge represents assignment). For worker transit stop  $wts_1$ ,  $t_2$  is preferred over  $t_1$  as  $t_2$  offers higher reward.

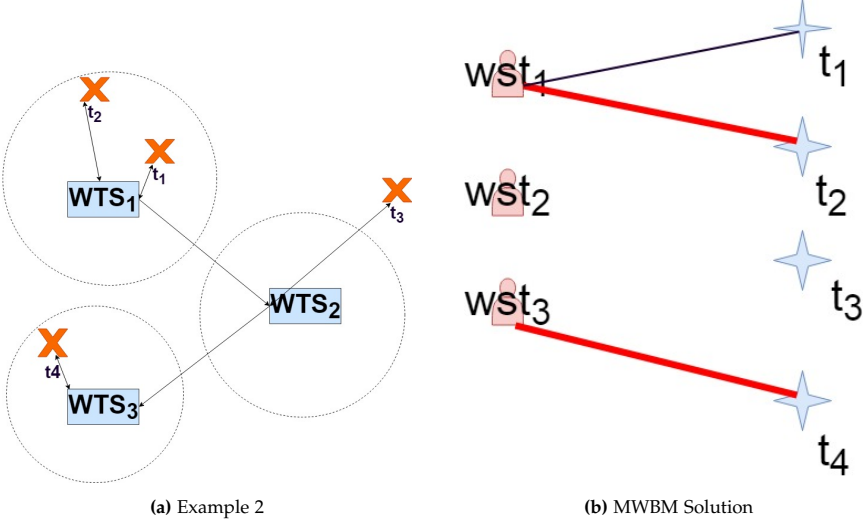


Fig. D.3

### 3.3 Minimum Distance based Direct Assignment (DA) Algorithm

Generally, workers are more likely to accept tasks that are closer to them. However, the MWBM algorithm above does not try to prioritize tasks that are closer to the worker transit stop of the worker. Furthermore, the bipartite graph has to be constructed for every incoming batch of workers and tasks. The construction of the bipartite graph has the time complexity  $O(t * (1 + w))$ , where  $t$  and  $w$  represent the number of tasks and workers, respectively. The constructed bipartite graph size increases over time, with the addition of new incoming tasks and workers to the previously unassigned tasks and workers. Consequently, the construction of the bipartite graph and the subsequent matching for every incoming batch progressively becomes more time-consuming. Since the transit-based online task assignments need to be performed in real-time and there is a need to prioritize tasks that are closer, we propose a direct assignment-based algorithm (DA). Similar to the MWBM algorithm, DA algorithm also involves solving the batch-based TTA by breaking it into individual offline-TTA problems for each incoming batch. (See Algorithm 23). DA algorithm tries to achieve the objective of batch-TTA by maximizing the rewards received by workers and simultaneously tries to minimize the travel distance for the workers, with respect to tasks assigned.

The direct assignment algorithm tries to find the best feasible task at each worker transit stop that provides the maximum reward and involves minimum travel distance to the task. Therefore, it has to search for all the tasks that satisfy the distance and task deadline constraints before assigning the

### 3. Transit-based Task Assignment

---

**ALGORITHM 23: MIN. DISTANCE-BASED ASSIGNMENT**


---

**Input:** Same as Algorithm 22

**Output:** Same as Algorithm 22

```

1  $TA(WTS \cup WTS_b, T \cup T_b) \leftarrow NULL;$ 
2  $TA(WTS_b, T_b) \leftarrow NULL;$ 
3  $maxReward \leftarrow 0;$ 
4  $minDist \leftarrow \infty;$ 
5 foreach  $wts \in WTS_b$  do
6    $maxDistAtStop \leftarrow$ 
7      $walkingSpeed * (wts.departureTime - wts.ArrivalTime - t.taskDuration) / 2;$ 
8    $feasibleTask \leftarrow NULL;$ 
9   foreach  $t \in T_b$  do
10    if  $dist(wts.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$  then
11      if  $minDist > dist(wts.l, t.l) \wedge maxReward < r$  then
12         $minDist \leftarrow dist(wts.l, t.l);$ 
13         $maxReward \leftarrow r(w, t);$ 
14         $feasibleTask \leftarrow t;$ 
15  $TA(WTS_b, T_b) \leftarrow TA(WTS_b, T_b) \cup TA(wts, feasibleTask);$ 
16 Lines 14 – 16 from MWBM Algorithm 22 return  $TA(WTS \cup WTS_b, T \cup T_b)$ 

```

---

task to the transit stop. Consequently, the time complexity of DA algorithm is directly dependent on the number of worker transit stops ( $n$ ), tasks ( $m$ ) and the batches ( $p$ ), i.e.,  $O(n * m * p)$ .

Consider *Example 2*, the DA algorithm assigns the nearest task with greater reward to the transit stop (See Table. D.3). For stop  $wts_1$ ,  $t_1$  is preferred over  $t_2$  as  $t_1$  offers better reward-to-distance ratio.

### 3.4 Credible Transit-based Task Assignment Algorithm (CTA)

In the previous algorithms, we have studied transit-based task assignment without ensuring a desired level of quality in the task responses from the workers. It was observed that workers might knowingly or unknowingly provide wrong answers to the queries associated with the spatial tasks [10]. To ensure a desired level of quality for the worker responses, we propose the *Credible Transit-based Task Assignment* algorithm. The algorithm solves the extended *Batch-TTA* problem (Definition 32) that includes the minimum worker credibility threshold (MWCT) constraint for the spatial task as the probability of the task being performed correctly. In other words, a spatial task can be assigned to the worker if and only if the worker's credibility is greater than or equal to the minimum worker credibility threshold of the spatial task.

The CTA algorithm is an extended DA algorithm for solving the updated Batch-based TTA problem with the minimum worker credibility threshold

**ALGORITHM 24: CREDIBLE TRANSIT-BASED TASK ASSIGNMENT (CTA)****Input:** Same as Algorithm 22**Output:** Same as Algorithm 22

---

```

8 Algorithm Same As DA except Line 9 Replaced By Below Psuedocode
9 if  $\text{dist}(wts.l, t.l) \leq \text{maxDistAtStop} \wedge t.\text{expiryTime} >$ 
    $wts.\text{arrivalTime} \wedge t.\text{minWorkerCred} \leq w.\text{credibility}$  then
10 return  $TA(WTS \cup WTS_b, T \cup T_b)$ 

```

---

**ALGORITHM 25: FLEXIBLE DIRECT ASSIGNMENT****Input:** Same as Algorithm 22, except  $FTA(WTS, T)$  is the optimal set of assignments instead of  $TA(WST, T)$ **Output:** Same as Algorithm 22, except with  $FTA(WTS \cup WTS_b, T \cup T_b)$ 


---

```

8 Same as DA Algorithm 23, except lines 9 – 13 replaced by below psuedocode
9 if  $\text{dist}(wts, t) \leq \text{maxDistAtStop} \wedge t.\text{expiryTime} > wts.\text{arrivalTime}$  then
10    $\lfloor$  Lines 10 – 13 from DA Algorithm 23
11 else if  $w.\text{threshold} \geq r$  then
12    $wts.\text{departureTime} \leftarrow wts.\text{arrivalTime} + (2 * \text{dist}(wts, t) / \text{WalkingSpeed});$ 
13    $\text{ReconstructRoutes}(wts, w.\text{destination},$ 
    $wts.\text{departureTime});$ 
14   if  $\text{ReconstructedRouteDestinationTime} < \text{startTime} + w.\text{maxTravelTime}$  then
15      $\lfloor$  Lines 10 – 13 from DA Algorithm 23
16 return  $FTA(WTS \cup WTS_b, T \cup T_b)$ 

```

---

constraint (as defined below).

**Definition 33.** (Minimum Worker Credibility Threshold Constraint): The worker's credibility score should be atleast the task's MWCT.

$$w.\text{credibility} \geq t.\text{minWorkerCred} \quad (\text{D.1})$$

The CTA algorithm tries to find the best feasible task at each worker transit stop that satisfies all the constraints, including the MWCT constraint and involves maximum reward and minimum travel distance to the task. Therefore, it has to search for all the tasks that satisfy the credibility, distance, and task deadline constraints before a task is assigned to a worker. Consequently, the time complexity of the CTA algorithm is directly dependent on the number of worker transit stops ( $n$ ), tasks ( $m$ ) and the batches ( $p$ ), i.e.,  $O(n * m * p)$ .

Consider *Example 2*, the CTA algorithm assigns the nearest task that satisfies the constraints with greater reward to the transit stop (See Table. D.4). Only one assignment  $wts_3, t_4$  satisfies the credibility constraint, i.e., worker credibility is at least equal to the MWCT.

## 4. Flexible Transit-based Task Assignment

Stop	Task As- signed	Reward
$wts_1$	$t_1$	20
$wts_2$	None	0
$wts_3$	$t_4$	20

Table D.3: Eg. 2 DA Solution

Stop	Task As- signed	Reward
$wts_1$	None	0
$wts_2$	None	0
$wts_3$	$t_4$	20

Table D.4: Eg. 2 CTA Solution

Stop	Task As- signed	Reward	New Arrival
$wts_1$	$t_1$	20	NA
$wts_2$	$t_3$	35	NA
$wts_3$	$t_4$	20	9:45 AM

Table D.5: Eg. 2 Flexible-DA Solution

## 4 Flexible Transit-based Task Assignment

### 4.1 Problem Definition

In the previous section, we have studied task assignment for fixed transit routes. However, it was observed that workers would modify their route for performing tasks, if a lucrative incentive is offered. In our context, we assume that if a task with higher reward than a certain threshold value will convince the worker to stay longer at the transit stop, and perform the task, despite the travel stop time constraint. Considering this, we propose the Flexible TTA problem, where **the worker transit route is no longer considered fixed and can be changed**. A worker  $w$  can search for a task  $t$  with a reward greater than the threshold, that will result in prolonging the stay at the worker transit stop  $wts$ , i.e., the *departureTime* at  $wts$  will be changed. Consequently, the arrival and departure times of next transit stops in the route will be changed. We assume that a flexible transit stop task assignment can happen at a transit stop  $wts$  if and only if the worker cannot find a task satisfying the travel stop time constraint at the worker transit stop. Considering these constraints, we

define the offline Flexible-TTA problem as:

**Definition 34.** (Offline Flexible Transit-based Task Assignment problem): Assume an input set of spatial tasks  $T$  and a set of workers  $W$  along with their set of worker transit stops  $WTS$ . The offline Transit-based Task Assignment (Offline Flexible-TTA) problem is an optimization problem with an objective to maximize the net rewards of individual workers. Offline Flexible-TTA finds an optimal Flexible Transit Stop Task Assignment set, denoted by  $FTA(WTS, T)$  with net reward  $r_{FTA}$ , is a set of 3-tuples of the form  $\langle wts, t, r \rangle$ , where  $wts$  is assigned to  $t$  with an associated reward  $r$ , given the following constraints are satisfied:

- **Threshold Reward Constraint :** The reward  $r$  should be greater than the worker  $w$ 's  $thresRew$ , i.e.,  $r > w.thresRew$
- **Task Deadline Constraint:** The task deadline should be later than the worker's arrival time at the transit stop, i.e.,  $t.expiryTime > wts.arrivalTime$
- **Travel Time Constraint:** The new reconstructed transit route with the updated  $wts.departureTime$  at transit stop  $wts$  should allow the worker  $w$  to reach the destination before  $w.startTime + w.maxTrolTime = destThresholdTime$ , i.e.,

$$destThresholdTime > wts.arrivalTime + t.taskDuration + (2 * dist(wts.l, t.l) / walkingSpeed)$$

**Definition 35.** (Batch-based Flexible Transit-based Task Assignment Problem): Assume a set of batches  $B$ , with each incoming batch  $b \in B$  comprising a set of unassigned tasks  $T_b$  and available workers  $W_b$  along with their transit routes  $WTS_b$ . The Batch-based Flexible Transit-based Task Assignment (Batch-based Flexible-TTA) problem is an optimization problem with the goal of finding an optimal Flexible Transit Stop Task Assignment  $FTA(WTS_b, T_b)$  that maximizes the net rewards ( $r_{FTA}$ ) received by the individual workers  $w \in W_b$  at their worker transit stops  $wts \in WTS_b$  by performing assigned task  $t \in T_b$  for every batch  $b$ .

## 4.2 Flexible Transit Route-based Direct Assignment Algorithm

We propose an extended DA algorithm to solve the Batch-based Flexible-TTA problem. In this *Flexible Transit Route-based Direct Assignment* algorithm (Flexible-DA), we consider the flexible transit route scenario, where the worker transit routes can be changed; for example, the worker might arrive late to the final destination. We assume that the worker will change her transit route if she fails to find a task that satisfies the travel stop time constraint at the



#### 4. Flexible Transit-based Task Assignment

transit stop and if a task with a threshold reward  $thresRew$  is present in the vicinity of the transit stop, resulting in a delay in the departure time at the transit stop. Flexible Transit Route-based assignments involve validation of the new departure timings at worker transit stops and to check whether the worker can reach the final destination before exhausting the maximum travel time  $maxTrvlTime$  after performing a potential task. Therefore, it utilizes a transit routing service (for example, Rejseplanen ([rejseplanen.dk](http://rejseplanen.dk))) to perform these checks, involving a REST-based API call to the routing service. Given, the high cost of the REST-based API (each request costs 0.7 seconds) calls, we do not extend the MWBM algorithm as it involves the construction of the complete bipartite graph resulting in a massive amount of requests to the routing service. Instead, we extend the DA algorithm to include the FTA assignments.

In Flexible-DA algorithm, we solve the Batch-based Flexible-TTA problem, similar to the Batch-TTA problem by breaking into individual *Offline Flexible-TTA* problems for every incoming batch (See Algorithm 25). This algorithm tries to achieve the batch-based Flexible-TTA to maximize the rewards received by workers and simultaneously seeks to reduce the travel distance to the tasks by the workers. Additionally, whenever a worker transit stop cannot find a task adhering to the fixed route, the worker expands the search space by looking for tasks beyond the transit stop time constraint which satisfies her threshold reward value. If a task with higher reward than the threshold is found, then Flexible-DA reconstructs the route with the new delayed departure time from the worker transit stop. If the reconstructed route reaches the final destination before exhausting the  $maxTrvlTime$ , then the matching will be valid, and the task  $t$  is assigned to the stop  $wt_s$ .

Consider *Example 2*, the Flexible-DA algorithm assigns the nearest task with greater reward to the transit stop (See Table. D.5). Furthermore, if a transit stop has no tasks satisfying its transit stop time constraint, the Flexible-DA algorithm searches further than the transit stop time constraint for tasks offering reward above the threshold. In this case, at transit stop  $wt_{s_2}$ , no tasks are within the vicinity. Consequently, the worker transit stop will be assigned to task  $t_3$ , that offers more than the threshold reward expected by the worker, i.e., 35. However, due to travelling longer than expected, the worker will miss the transit that departs at 9:00 AM, and will have to take the next bus to reach the next transit stop  $wt_{s_3}$ , reflected by the new arrival time 9:40 AM in Table D.5.

**ALGORITHM 26: ONLINE TRANSIT-BASED TASK ASSIGNMENT (OLA)**

**Input:** An incoming task  $t$  or a recently available worker  $w$  at transit stop  $wts$  along with available workers  $W$  at worker transit stops  $WTS$ , and a set of previously unassigned tasks  $T$ .  $c$  is the fixed reward per task.

**Output:** Task Assignment  $\langle w, wts, t \rangle$  with reward  $maxReward$  and minimum travel distance  $minDist$

```

1   $maxReward \leftarrow 0$ ;
2   $minDist \leftarrow \infty$ ;
3   $removeExpiredTasks(T)$ ;
4   $removeUnavailbleWorkers(W)$ ;
5  if New Task Arrival  $t$  then
6       $T \leftarrow T \cup \{t\}$ ;
7      foreach  $wts \in WTS$  do
8           $maxDistAtStop \leftarrow walkingSpeed * (wts.departureTime -$ 
9               $wts.ArrivalTime - t.taskDuration) / 2$ ;
10         if  $dist(wts.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$  then
11              $minDist \leftarrow dist(wts.l, t.l)$ ;
12              $maxReward \leftarrow r(w, t)$ ;
13              $Assignment \leftarrow \langle w, wst, t \rangle$ ;
14              $T \leftarrow T \setminus \{t\}$ ;
15              $W \leftarrow W \setminus \{w\}$ ;
16              $WTS \leftarrow WTS \setminus \{wts\}$ ;
17 else if New Worker Arrival  $w$  at  $wts$  then
18      $W \leftarrow W \cup \{w\}$ ;
19      $WTS \leftarrow WTS \cup \{wts\}$ ;
20      $maxDistAtStop \leftarrow$ 
21          $walkingSpeed * (wts.departureTime - wts.ArrivalTime - t.taskDuration) / 2$ ;
22     foreach  $t \in T$  do
23         if  $dist(wts.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$  then
24              $minDist \leftarrow dist(wts.l, t.l)$ ;
25              $maxReward \leftarrow r(w, t)$ ;
26              $Assignment \leftarrow \langle w, wst, t \rangle$ ;
27              $T \leftarrow T \setminus \{t\}$ ;
28              $W \leftarrow W \setminus \{w\}$ ;
29              $WTS \leftarrow WTS \setminus \{wts\}$ ;
30 return  $Assignment \langle w, wst, t \rangle$ 

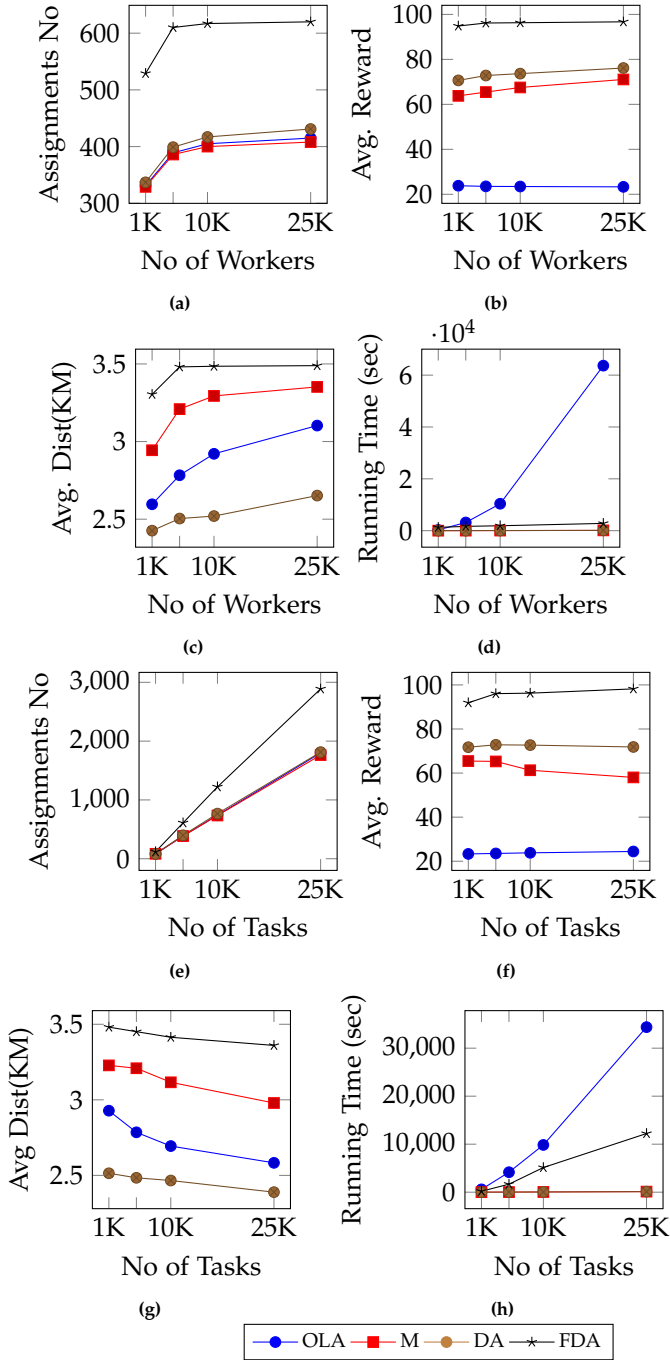
```

## 5 Experimental Evaluation

### 5.1 Experiment Setup

**Workers Transit Routes DataSet:** It is hard to find real datasets that reflect the workers and their transit movement information in the real world. Consequently, we have used realistic datasets for generating workers ranging from

## 5. Experimental Evaluation



**Fig. D.4:** Effect of varying the number of workers on :a. Number of Assignments.b Average Reward received by a worker c. Average Distance travelled by a worker (KM) d. Average Running Time (seconds). Effect of varying the number of tasks on :e. Number of Assignments.f Average Reward received by a worker g. Average Distance travelled by a worker (KM) h. Average Running Time (seconds).

1K to 25K in number. The worker transit routes are generated by identifying the residential and commercial zones in the city of Aalborg, Denmark. Two periods were set for workers going from homes in residential zones to workplaces in commercial zones in the morning “7:00 to 11:00” and workers returning from work to home in the evening “16:00 to 21:00”. The resulting set contained the origin, destination and start time information for each worker. We constructed the worker transit routes by using that information to create a REST API call to the Rejseplanen Public Transport Routing Service, Denmark. The routing service returns the transit details from the given origin and destination transit stops, that includes the arrival and departure at each transit stop, excluding the destination. At the destination, only the arrival is returned. The workers are generated with uniform values of *thresRew* and *maxTrolTime* parameters. The workers credibility parameters are generated with random values between 0.2 to 0.9, due to the unavailability of workers’ historical information. As the credibility values are dependedent on the correctness of the worker responses and independent of other factors like geographical area, etc., we believe that an uniform distribution would reflect the real-world scenario.

**Synthetic Dataset for Tasks:** For the synthetic dataset, we have generated a varied number of tasks from 1K to 25K. The tasks are distributed randomly in the geographical extent of the Aalborg City, Denmark. The tasks are generated with varying values of *issueTime* between “7:00 to 21:00”. The default *expiryTime* is set as twenty four hours from the *issueTime*. The tasks are generated with equal values of minimum worker credibility threshold parameters.

**Algorithms:** We have conducted our evaluation based on the assignment algorithms presented in this paper. First, we have evaluated the fixed transit route algorithms like *MWBM*, *DA*, *CTA* and the flexible transit route algorithm *Flexible-DA*. Our algorithms are based on the batch-based input model.

**Baseline Algorithm:** As a baseline, we consider the online input model, where the worker/ task arrives dynamically to the system. The SC-server will not have any prior information about the WTRs in the online input model. We assume that the worker would notify the SC-server whenever she is available to perform a task. In our transit-based context, the worker will notify the SC-server whenever she reaches a transit stop. The SC-server tries to assign a task once a worker becomes available immediately. The baseline online transit-based task algorithm is denoted by *OLA* (See Algorithm 26).

With the arrival of each new task or worker, the search space of the *OLA* algorithm grows and simultaneously shrinks with every assignment. Consequently, the time complexity of the *OLA* algorithm is directly dependent on the number of worker transit stops ( $n$ ), and tasks ( $m$ ), i.e.,  $O(n * (m^2) + (n^2) * m)$ .

**Configuration and Measures:** We compare the different assignment algorithms based on the following measures: Number of Assigned tasks to the

## 5. Experimental Evaluation

Parameters	Value Range
Number of workers (Transit stops)	1K (2419), <b>5K (12303)</b> , 10K (24407),25K (60941)
Number of Tasks	1K, <b>5K</b> ,10K,25K
Maximum Travel Time of worker	1, <b>2</b> ,3
Worker credibility	Randomly generated between 0.2 - 0.9
Minimum worker credibility threshold of task	0.5,0.6, <b>0.7</b> ,0.8,0.9
Worker's Threshold Reward	70, <b>80</b> ,90
Average Walking Speed	1.46 m/s [6]

**Table D.6:** Experiment Parameters

workers, Average travel distance for the assigned task, Average reward per worker, and the running time. We vary the number of workers from 1K to 25K and the number of tasks from 1K to 25K to evaluate the scalability of our algorithms. To simulate a real application scenario, we simulate a batch of workers and tasks every hour. The batches contain varying sizes of workers and tasks as they are randomly generated during different periods. For example, 25K worker dataset has a maximum batch size of 3184, a minimum of 2437, a mean of 2777 and a median size of 2600. Similarly, 25K tasks dataset has a maximum batch size of 381, a minimum of 328, a mean of 357, and a median size of 358. The CTA algorithm is evaluated by varying the number of workers (1K to 25K), the number of tasks (1K to 25K), and the minimum worker credibility threshold constraint (0.5 to 0.9). Similarly, we evaluate the effect of varying the *maxTrolTime* parameter of the workers (from 1 to 3 hours) to evaluate its effect on the above-mentioned different measures. Furthermore, we evaluate the effect of varying the threshold reward parameter of workers to determine its effect on the above-mentioned measures. The default values for the scalability experiments are depicted in bold in the *Experiment Parameters* Table D.6. All algorithms were implemented in Java utilizing Postgresql with PostGIS and pgrouting extensions and Jgraph

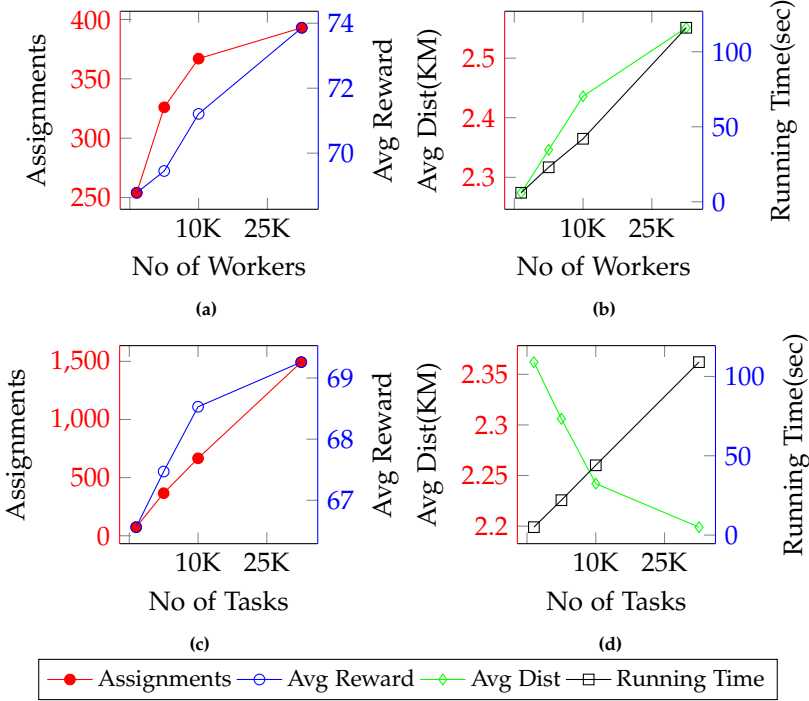


Fig. D.5: Effect of varying workers (a,b) and tasks (c, d) on Credible Transit-based Task Assignment algorithm (CTA).

library [12]. All experiments were conducted on the Windows 8.1 OS with Intel Core i7-5600 CPU@ 2.60G HZ and 12 GB memory.

## 5.2 Scalability with the size of workers data set

In this set of experiments, we evaluated the scalability of our proposed transit-based assignment algorithms by varying the number of workers from 1K to 25K. Figure D.4a illustrates the effect of the varying the number of workers on the number of assigned tasks directly. With respect to the number of assigned tasks, the *Flexible-DA* has outperformed the other algorithms. When compared to the *OLA*, *MWBM*, and *DA* algorithms, the number of worker transit stop assignments has increased more than 55%. The reason is that *Flexible-DA* supports flexible transit routes, which can bypass the transit stop time constraint in cases where are no feasible tasks at the transit stops. *DA* has performed marginally better than the *OLA* and *MWBM* algorithms.

With regard to the average reward received by the worker, *Flexible-DA* performs better than the other algorithms (See Fig. D.4b). The average reward received by the worker in the *Flexible-DA* algorithm has increased by 35%

## 5. Experimental Evaluation

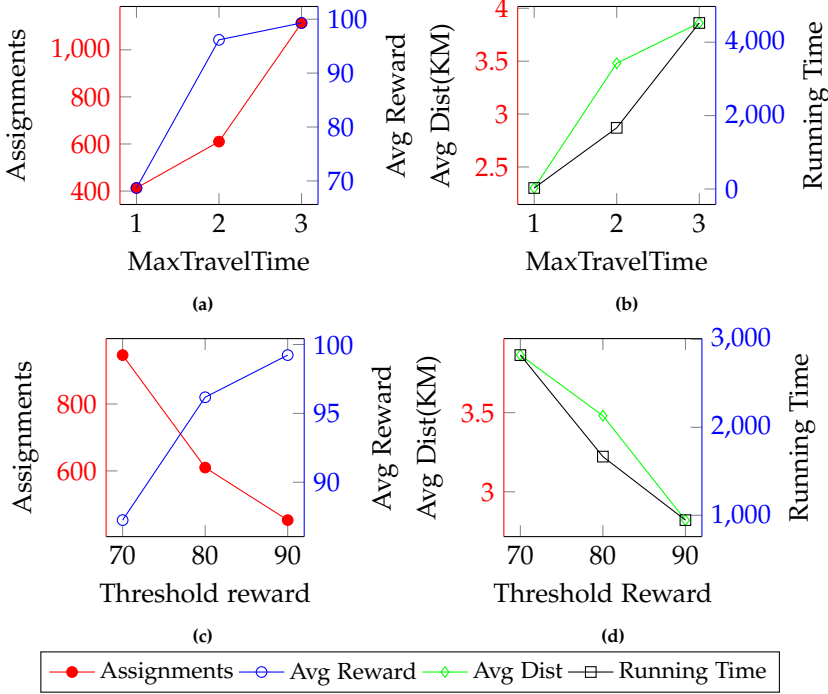
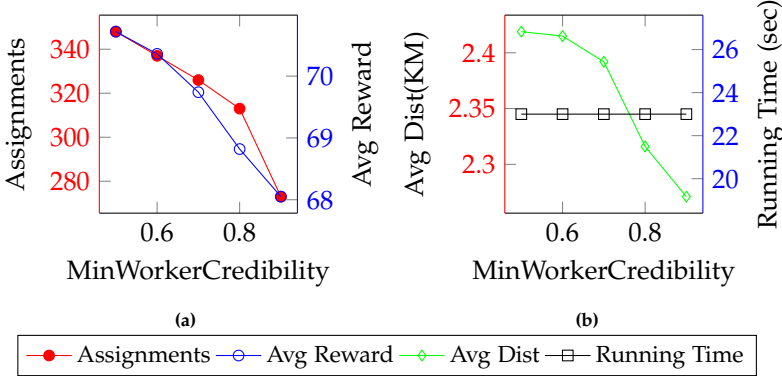


Fig. D.6: Effect of varying maxTravelTime (a,b) and threshold reward (c, d) on Flexible-DA.

when compared to *DA*, and around 50% when compared to *MWBM*. *Flexible-DA* results in three times higher reward than the baseline algorithm *OLA*. The reason is that in the case of *Flexible-DA*, the SC-server has the knowledge about the worker's future spatiotemporal movement that facilitates the worker to extend the stay at different transit stops of her route. Furthermore, *MWBM* and *DA* result in nearly three times higher reward than the baseline algorithm *OLA*. *Flexible-DA* performs better due to the availability of tasks with rewards higher than the threshold, that satisfies the maximum travel time constraint. *DA* delivers 15% better average reward than *MWBM*. With respect to the average distance travelled by the worker, *DA* outperforms *MWBM* by 20%, *OLA* by 30% and *Flexible-DA* by 50% (See Fig. D.4c). The reason is that *DA* gives more priority to tasks that are closer to the transit stop. Furthermore, with regard to the running time, the baseline algorithm *OLA* has the worst performance among the other algorithms. Due to the increase of search space with every new arrival of worker/ task, a substantial increase of running time is observed for the *OLA* algorithm (See Fig. D.4d). *Flexible-DA* has the worst performance among the proposed algorithms, due to the REST API calls for validation and route reconstruction activities. The fixed-route based algorithms (*MWBM*, and *DA*) is at least 40-times faster than



**Fig. D.7:** Effect of varying minimum worker credibility threshold (a,b) on *Credible Transit-based Task Assignment Algorithm*.

the baseline *OLA*. The proposed algorithms (*MWBM*, and *DA*) have almost similar performance as the input size increases.

### 5.3 Scalability with the size of Tasks data set

In this set of experiments, we evaluated the scalability of our proposed algorithms by varying the tasks from 1K to 25K. Figure D.4e illustrates the effect of varying tasks on the number of assigned tasks directly. Regarding the number of assigned tasks, the *Flexible-DA* has outperformed the other algorithms, and the increase is more evident as the number of tasks increases. When compared to *OLA*, *MWBM*, and *DA* algorithms, the number of worker transit stop assignments has increased more than 60%. As *Flexible-DA* supports flexible transit routes, the workers can travel further by delaying their departure time to gain more tasks.

Regarding the average reward received by the worker, *Flexible-DA* has resulted in four times higher reward than the baseline algorithm *OLA*, 35% better than *MWBM* algorithm, and around 20% better than *DA* algorithm (See Fig. D.4f). *MWBM*, and *DA* result in around three times higher reward than the baseline algorithm *OLA*. With respect to the average distance travelled by the worker, *DA* outperforms *Flexible-DA* by 50%, *OLA* by 20%, and *MWBM* by 30% (See Fig. D.4g). The reason is that *DA* gives more priority to tasks that are closer to the transit stop. Furthermore, with regard to the running time, the baseline algorithm *OLA* has the worst performance among the other algorithms. Due to the increase of search space with every new arrival of worker/ task, a substantial increase of running time is observed for the *OLA* algorithm (See Fig. D.4h). *Flexible-DA* has the worst performance among the proposed algorithms, due to the REST API calls for validation and route reconstruction activities with each call costing 0.7 seconds. The other



proposed algorithms (*MWBM*, and *DA*) have almost similar performance as the input size increases.

### 5.4 Effect of varying the number of workers and the number of tasks on CTA

In this set of experiments, we evaluated the *Credible Transit-based Task Assignment Algorithm (CTA)* by varying the number of workers and the number of tasks. Fig. D.5a shows the effect of varying the number of workers on the number of assignments and the average reward received by the worker. As the evaluation of CTA is based on additional parameters like worker credibility and minimum worker credibility threshold of the task, it cannot be directly compared with the other methods proposed in this paper. Instead, this sub-section focuses exclusively on the special aspects of the CTA algorithm. An upward trend can be observed regarding the number of assignments, and the average reward as the number of workers increases from 1K to 25K. The upward trend can be attributed to the increased availability of workers with higher credibility for task assignment. However, it can be noticed that the jump in the number of assignments when the workers increase from 5k to 10K is sharper than when the workers increase from 10k to 25K. The reason is that the majority of the new workers with credibility higher than the minimum worker credibility threshold (0.7) are not close to the tasks, thereby failing the distance and deadline constraints.

Similarly, Fig. D.5b shows the effect of varying the number of workers on the average distance travelled by the worker for performing the assigned tasks and the running time. It can be observed that the average travel distance for a worker and the running time increases gradually as the number of workers increases. The average travel distance for a worker increases due to different reasons like the increase in the number of tasks assigned to a single worker thus, adding extra travel distance, and the increase in the assignment of tasks that are located relatively far from the transit stop.

Fig. D.5c shows the effect of varying the number of tasks on the number of assignments and the average reward received by the worker. It can be observed that as the number of tasks increases, the number of assignments, and the average reward increases gradually. The reason is that more tasks within the close proximity of transit stops satisfying deadline constraint are available for assignment. The gradual increase of the number of assignments can be attributed to the uniform distribution for tasks generation.

Similarly, Fig. D.5d shows the effect of varying the number of tasks on the average distance travelled by the worker for performing the assigned tasks and the running time. It can be observed that the average travel distance per worker decreases as the number of tasks increases; in contrast, the running time increases. The average travel distance for a worker decreases due to the

availability of more tasks in the near vicinity of the workers' transit stops satisfying the different constraints.

## 5.5 Effect of varying maxTravelTime, threshold reward on Flexible-DA

In this set of experiments, we evaluated the *Flexible-DA* by varying the max-TravelTime and the threshold reward values of the worker. We considered only the *Flexible-DA* as the other algorithms are not impacted by the max-TravelTime and threshold reward values. It can be observed from Fig. D.6a and Fig. D.6b, that as the maxTravelTime value increases, the number of assignments, average travel distance, average reward and the running time increases. The reason is that as the maxTravelTime increases as more tasks become eligible for the workers to perform in the *Flexible-DA* algorithm.

Similarly, in Fig. D.6c and Fig. D.6d, it can be observed that the increase in the threshold reward value of the worker, results in a decrease in the number of assignments, the average travel distance and the average running speed. The reason is that as the threshold reward increases, fewer tasks will be eligible for the flexible transit route-based task assignments. However, the average reward value increases as the threshold reward increases as the priority will be given to tasks with higher rewards.

## 5.6 Effect of varying minimum worker credibility threshold value on CTA

In this set of experiments, we evaluated the *CTA* by varying the minimum worker credibility threshold (MWCT) values of the task. We considered only the *CTA* as the *MWCT* values do not impact the other algorithms. It can be observed from Fig. D.7a and Fig. D.7b, that as the *MWCT* value increases, the number of assignments, the average travel distance, and the average reward showcases a downward trend. Given that the number of worker and the number of tasks are fixed at 5K, the number of potential assignments with the satisfied credibility constraint will be reduced as the *MWCT* value increases. For example, when the *MWCT* value is increased to 0.6 from 0.5, the workers with credibility values ranging between  $[0.5, 0.6)$  becomes ineligible to perform tasks owing to the credibility constraint. The running time is not impacted by the increase of *MWCT* value.

## 5.7 Summary

We found out that the *Flexible-DA* outclasses the fixed transit-based algorithms in the measure of the number of assigned tasks and the average reward. However, the *Flexible-DA* is highly time-consuming than the fixed

transit-based algorithms. With respect to the baseline online task assignment algorithm (*OLA*), *Flexible-DA* results in three times higher reward and at least three times faster runtime. Similarly, *DA* outclasses the other fixed transit-based algorithm (*MWBM*) in the measure of the average travel distance. With regard to the credibility-based assignment algorithm, the measures of assigned tasks, the average reward for an assigned task, and the running time increases gradually as the number of workers and the number of tasks increase. With regard to the average travel distance, *CTA* results in a gradual increase as the number of workers increases, and a gradual decrease as the number of tasks increases. Furthermore, we noticed a downward trend for all the measures for *CTA* as the minimum worker credibility threshold values increase.

## 6 Related Work

Spatial Crowdsourcing (SC) [8] harnesses the potential of a crowd to perform real-world spatial tasks that are not supported by conventional crowdsourcing techniques. Typically, the workers in SC move to the tasks' locations to perform tasks. The SC-server supports two types of task publishing modes [9]: Server-Assigned Task (SAT) publishing mode and Worker Selected Task (WST) publishing mode. Due to the level of control exerted by the SC-server in the SAT publishing mode, research gained momentum in SC literature related to SAT publishing mode [5, 7, 14]. The SAT publishing mode, in the context of assignment of workers to tasks, involves the problem of SC-server choosing the workers for the tasks. Typically, the workers and tasks arrive dynamically to the SC-server, thereby leading to uncertainty in the task assignment process. This scenario is termed as online task assignment problem [9, 14–16]. Typically, these assignment problems aim to achieve optimization goals like maximizing the number of tasks assigned [9, 14], minimizing the cost incurred by the server [16], improving the quality of task responses [3] or goals benefitting the workers like maximizing the reward received by the worker [2].

To reduce the uncertainty in the online task assignment scenario, SC-server can exploit the workers' movement information to identify the workers' arrival order [8]. There are some existing works [2, 4, 13], that try to harness the workers' movement information. For instance, [4] tries to solve a single-worker-multiple-tasks type of task scheduling problem. They assume that there are a set of available tasks that have to be incorporated into the worker's route based on her budget on a detour. They combine the two objectives of minimize detour and maximize reward by using the skyline queries (finding the set of non-dominated paths). They prove the problem to be NP-hard by reducing it to the Travelling Salesman Problem and proposes exact

and heuristics-based approximate solutions for solving it. However, they do not consider the dynamic nature of workers and tasks arrival. Furthermore, they do not consider some temporal aspects like time required to perform tasks and maximum travel time that a worker can afford. We try to address the dynamic nature by proposing a task assignment problem with a batch-based input model of workers' transit routes and tasks' arrival. Additionally, we also consider the time required to perform tasks and the maximum travel time of the worker.

## 7 Conclusions and Future Work

In this paper, we proposed a task assignment model that exploits the workers' transit information to offer an alternative strategy for the online spatial task assignment in SC. This paper modeled the potential task assignment opportunities in a fixed worker transit route that is followed strictly. Additionally, we modeled a flexible transit route scenario with an assumption that the worker would be willing to delay her trip if a task offers more than a threshold reward. We defined the three variants of the transit-based task assignment (TTA) problem, *offline-TTA*, *Batch-TTA*, and *Flexible-TTA*, with a goal to maximize worker rewards considering the fixed and the flexible worker transit models, respectively.

We prove that the offline version of the TTA problem can be reduced to a maximum weighted bipartite matching problem. We utilize the offline version of the defined problem to solve the online batch-based versions of them. Two algorithms are proposed for solving the Batch-TTA problem; namely, *MWBM* and *DA*. Additionally, to ensure a certain level of worker response quality, we proposed a *CTA* algorithm considering the worker credibility information for task assignment. *CTA* assigns tasks to workers that satisfy the minimum worker credibility threshold constraint. Furthermore, for the Batch-based Flexible-TTA problem, we proposed an extended version of the *DA*, *Flexible-DA* considering the flexible worker transit model. We compared our proposed algorithms to a baseline algorithm, *OLA* that models the online assignment without considering the routing information. Through our extensive evaluation, we observed that the *Flexible-DA* outperforms the other proposed algorithms by 55% in terms of the number of assigned tasks, and at least 35% in terms of average reward for the worker. With respect to the baseline algorithm *OLA*, *Flexible-DA* algorithm results in four times the higher reward and *DA*, and *MWBM* algorithms result in nearly three times the higher reward. *DA* outperforms the other algorithms by at least 20% in terms of average travel distance. With respect to the running time, the fixed-route based algorithms (*MWBM*, and *DA*) is at least 40-times faster than the baseline *OLA*. The *Flexible-DA* algorithm is slower than the fixed-route

based algorithms due to the REST API calls and route reconstruction activities. However, *Flexible-DA* is at least three times faster than the baseline *OLA* with respect to runtime.

There are several promising directions for future work. First, we need to consider promising worker movement models to identify more real-world cases to improve the chances for the tasks to be assigned. Second, we need to consider new worker movement models that can relax the immutability concept of task assignments, i.e., workers could have the choice to exchange their assigned task for a newly available task with a higher reward. Furthermore, we need to compare our assignment model against a model where the workers can bid for the tasks to validate the time-effectiveness of bidding versus server-assignment in a moving worker scenario.

## Acknowledgements

*This research has been funded in part by the European Commission through the Erasmus Mundus Joint Doctorate Information Technologies for Business Intelligence - Doctoral College (IT4BI-DC). Xike Xie is supported by the CAS Pioneer Hundred Talents Program, Fundamental Research Funds for the Central Universities, National Natural Science Foundation of China (No. 61772492), and Natural Science Foundation of Jiangsu Province (No. BK20171240).*

## References

- [1] M. Allahbakhsh, B. Benatallah, A. Ignjatovic, H. R. Motahari-Nezhad, E. Bertino, and S. Dustdar, "Quality control in crowdsourcing systems: Issues and directions," *IEEE Internet Computing*, vol. 17, pp. 76–81, 2013.
- [2] C. Chen, S.-F. Cheng, A. Gunawan, A. Misra, K. Dasgupta, and D. Chander, "Traccs: A framework for trajectory-aware coordinated urban crowd-sourcing," in *HCOMP*, 2014, pp. 30–40.
- [3] P. Cheng, X. Lian, L. Chen, and C. Shahabi, "Prediction-based task assignment in spatial crowdsourcing," in *ICDE*. IEEE, 2017, pp. 997–1008.
- [4] C. F. Costa and M. A. Nascimento, "In-route task selection in crowdsourcing," in *GIS*, 2018, pp. 524–527.
- [5] D. Deng, C. Shahabi, and L. Zhu, "Task matching and scheduling for multiple workers in spatial crowdsourcing," in *GIS*, 2015, p. 21.
- [6] K. Fitzpatrick, M. A. Brewer, and S. Turner, "Another look at pedestrian walking speed," *TRR*, vol. 1982, no. 1, pp. 21–29, 2006.
- [7] S. R. B. Gummidi, T. B. Pedersen, X. Xie, and E. Zimányi, "Push-based spatial crowdsourcing for enriching semantic tags in openstreetmap," in *GIS*, 2019, p. 532–535.

## References

- [8] S. R. B. Gummidi, X. Xie, and T. B. Pedersen, "A survey of spatial crowdsourcing," *ACM Transactions on Database Systems (TODS)*, vol. 44, no. 2, p. 8, 2019.
- [9] L. Kazemi and C. Shahabi, "Geocrowd: enabling query answering with spatial crowdsourcing," in *GIS*, 2012, pp. 189–198.
- [10] L. Kazemi, C. Shahabi, and L. Chen, "Geotrucrowd: trustworthy query answering with spatial crowdsourcing," in *GIS*, 2013, pp. 314–323.
- [11] K. Mehlhorn and S. Näher, *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, 1999. [Online]. Available: <http://www.mpi-sb.mpg.de/%7Emehlhorn/LEDAbook.html>
- [12] D. Michail, J. Kinable, B. Naveh, and J. V. Sichi, "Jgrapht—a java library for graph data structures and algorithms," *arXiv preprint arXiv:1904.08355*, 2019.
- [13] D. Sun, K. Xu, H. Cheng, Y. Zhang, T. Song, R. Liu, and Y. Xu, "Online delivery route recommendation in spatial crowdsourcing," *WWW*, pp. 1–22, 2018.
- [14] H. To, C. Shahabi, and L. Kazemi, "A server-assigned spatial crowdsourcing framework," *ACM TSAS*, vol. 1, no. 1, p. 2, 2015.
- [15] Y. Tong, L. Wang, Z. Zhou, B. Ding, L. Chen, J. Ye, and K. Xu, "Flexible online task assignment in real-time spatial data," *PVLDB*, vol. 10, no. 11, pp. 1334–1345, 2017.
- [16] L. Tran, H. To, L. Fan, and C. Shahabi, "A real-time framework for task assignment in hyperlocal spatial crowdsourcing," *TOIST*, p. 37, 2017.

# Paper E

## Push-based Spatial Crowdsourcing for Enriching Semantic Tags in OpenStreetMap

Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach  
Pedersen, Xike Xie, and Esteban Zimányi

The paper has been published in the  
*SIGSPATIAL '19: Proceedings of the 27th ACM SIGSPATIAL International  
Conference on Advances in Geographic Information Systems*  
pp. 532–535, 2019. DOI: <https://doi.org/10.1145/3347146.3359365>

## Abstract

*OpenStreetMap (OSM) is a popular community-driven mapping platform with voluntary contributions from (amateur) cartographers. However, it is a difficult process for the cartographer to identify the areas where she can best contribute to OSM. Furthermore, the current OSM spatial entities are missing many tags; for example, top three road network tags, Name, Source, and Surface, are available only for the 10% of the total road segments. Our paper aims to improve the quantity and quality of the road network tags by actively pushing the nearest road segments for the cartographer to be mapped. We propose a push-based spatial crowdsourcing method to achieve this objective, and validate it by focusing on road segments in OSM. Specifically, we formally define the batch-based maximum road segment task assignment problem and suggest methods based on heuristics like travel distance and road segment task grouping. Finally, our experimental evaluation verify the applicability of our assignment solutions by comparing the resulting number of assigned tasks. With regard to the number of assigned road segments, our junctions-based and road segment-based heuristic methods, outperform the baseline methods by five and two times, respectively.*

© 2019 ACM. Reprinted, with permission from Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach Pedersen, Xike Xie, and Esteban Zimányi. Push-based Spatial Crowdsourcing for Enriching Semantic Tags in OpenStreetMap. In: *SIGSPATIAL '19: Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 532–535, 2019. <https://doi.org/10.1145/3347146.3359365>

*The layout has been revised.*



# 1 Introduction

The quality of the services offered by the OpenStreetMap (OSM) <sup>1</sup> is highly dependent on the annotations associated with the cartographic elements. For example, to calculate the route between two points in the geographical space, the routing service should know different details like whether the roads connecting the points are accessible by cars or not, whether the road is one way or not. Unfortunately, current OSM data's quality is hindered by the lack of tags associated with the spatial entities.

For example, let us take the case of the road network in the OSM database. According to the OSM standards, each road segment can be tagged with 27 standard semantic tags like the type of the road, the name of the road ([taginfo.openstreetmap.org](http://taginfo.openstreetmap.org)). Additionally, the contributor can define some new tags as well. However, the current road network's tag coverage is not adequate to adhere to the high-quality standards of other mapping and routing services like Google Maps ([www.maps.google.com](http://www.maps.google.com)). We have observed that the coverage of the top ten tags for highways or road segments in the OSM database is inadequate, with even the *Name* tag being associated with only 27.07% of the total road segments in the entire OSM database. The top three tags, *Name*, *Source*, and *Surface*, are available only for the 10% of the total road segments.

To summarize, there is a need to actively push the contributors to enrich the current OSM database with as many tags as possible. The Server Assigned Tasks (SAT) publishing mode of Spatial Crowdsourcing (SC) facilitates pushing cartographic entities to contributors for enriching tags. However, given the limited availability of contributors, efficient task assignment strategies are needed to increase the contributions from the (amateur) cartographers, while ensuring that the contributor is not overburdened with travel costs. Within this context, the main contributions of this paper are:

- We present a solution based on the SAT (push-based) publishing mode of SC for the process of enriching semantic tags of the OSM road network.
- We propose methods to assign road segments/ junctions to workers based on different constraints in both offline and batch-based worker input model scenarios.
- We facilitate the workers to maximize their contribution by maximizing their road segment/ junction task assignments by proposing effective heuristic methods.

---

<sup>1</sup>[www.openstreetmap.org](http://www.openstreetmap.org)

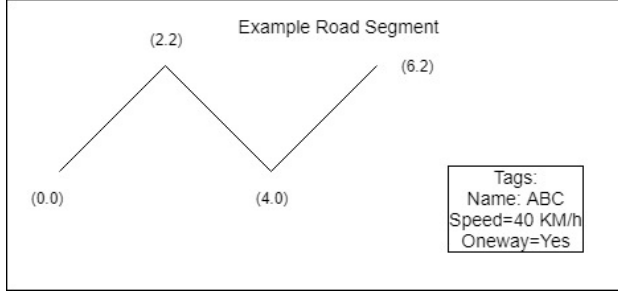


Fig. E.1: Example Road Segment

- We test the applicability of our proposed methods through an extensive experimental evaluation based on extracted OSM road segments from the region of Aalborg, Denmark.

## 2 Preliminaries

In this section, we define a set of preliminaries in the context of push-based spatial crowdsourcing. First, we formally define a road segment task.

**Road Segment task:** A road segment task  $rst$  contains a line segment with  $m$  nodes. The start node and end node of the line segment are located at  $l_1$  and  $l_m$ , respectively. The road segment task  $rst$  has information regarding the  $n$  tags ( $tag_n$ ) along with their values ( $val_n$ ). The total number of tags  $n$  will be at least 27, as we would like to retrieve the information about the 27 standard tags, along with the existing custom tags. The road segment task  $rst$  has no temporal constraints associated with it, owing to the dynamic nature of the road network.

$$rst = \langle \langle tag_1, val_1 \rangle, \langle tag_2, val_2 \rangle, \dots, \langle tag_n, val_n \rangle, \langle l_1, l_2, \dots, l_m \rangle \rangle$$

Consider the example in Fig. E.1, the location set of the road segment  $\langle (0,0), (2.2), (4.0), (6.2) \rangle$  is mapped to node locations  $\langle l_1, l_2, l_3, l_4 \rangle$ . The tag-set of the road segment will be mapped to the respective tags of the road segment task and the tags that are not present in the road segment will be marked as empty, i.e.,  $\langle \langle Speed = "40Km/h" \rangle, \langle Name = "ABC" \rangle, \langle Oneway = "Yes" \rangle, \langle Lanes = "" \rangle, \langle Surface = "" \rangle, \dots \rangle$ .

**Worker:** A worker, denoted by  $w$ , is an amateur cartographer willing to perform an assigned task by travelling to the task's road segment. Worker  $w$  has an identification number  $wid$  along with a location  $l$  that she reports to the assignment module. Additionally, Worker  $w$  specifies her preferences for accepting a task like the maximum number of tasks she's willing to perform,  $maxT$  and the maximum distance  $d$  she is willing to travel for performing a task. A worker is defined as:  $w = \langle wid, l, maxT, d \rangle$

## 2. Preliminaries

**Valid Task Assignment Set:** Given a set of workers  $W = \{w_1, w_2, \dots\}$  and a set of road segment tasks  $RST = \{rst_1, rst_2, \dots\}$ , a *Valid Task Assignment Set*, denoted by  $VA$ , is a set of assignments of the form  $\langle w, rst \rangle$ , in which a road segment task  $rst$  is assigned to a worker  $w$ , while satisfying the following constraints:

- **Maximum Tasks constraint:** The worker  $w$  should not be assigned more road segment tasks than the  $maxT$  value.
- **Distance Constraint:** The worker  $w$  should not be assigned tasks that are farther than  $d$ , i.e.,  $dist(w, rst) \leq d$

In line with the objective of enriching the OSM database, we have to define our problem to add as many tag values for the different road segment tasks  $rst \in RST$  as possible. Therefore, the total number of road segment tasks assigned to workers should be maximized to increase the possibility of obtaining more tags for the road segments, considering workers' constraints. These issues give rise to our optimization problem, *Maximum Road Segment Task Assignment problem (MRSTA)*. The inputs for the MRSTA problem are the set of workers  $W$  and the set of road segment tasks  $RST$ . The set of road segment tasks stay constant, as the road segment tag information requires frequent assignments to update the associated tag details. However, the set of workers would vary as the workers might not be online/ available all the time. For a better understanding of the MRSTA problem, we consider both the input models where the workers are available all the time (offline model) and are intermittently available (batch model). The definition of the offline-MRSTA problem is given below:

**Offline Maximum Road Segment Task Assignment:** Assume an input set of road segment tasks  $RST$ , a set of available workers  $W$ , and a set of valid task assignments  $VA$ . The offline Maximum Road Segment Task Assignment (Offline-MRSTA) problem is an optimization problem with the goal of maximizing the total number of road segment task assignments.

$$OffMR(RST, W) = \arg \max_{VA \in 2^{RST \times W} \text{ s.t. } VA \text{ is valid}} (|VA|)$$

, where  $OffMR(RST, W)$  is the maximal number of assignments.

**Batch-based Maximum Road Segment Task Assignment:** The Batch-based Maximum Road Segment Task Assignment problem (Batch-MRSTA) is an optimization problem with a goal of maximizing the total number of road segment task assignments for continuous batches of workers  $W_i$  at each time instance  $i$ . At each time instance, the assignment problem for the batch of workers, along with the road segment tasks can be solved by reducing it to the Offline-MRSTA problem. Thus, the Batch-MRSTA can be solved by a series of Offline-MRSTA problems at each time instance.

$$OnMR(RST, W) = \sum_{i \in T} OffMR(RST, W_i)$$

, where  $OnMR(RST, W)$  is the optimal number of assignments and  $W_i$  is the set of workers at time instance  $i$ . We assume, that the workers' arrive in batches at each time instance.

### 3 Algorithms

This section describes in detail the proposed assignment algorithms. The offline-MRSTA problem can be solved by utilizing the existing solutions [4] for reducing it to a max-flow problem. We build our solutions to the Batch-MRSTA problem by employing some of the existing solutions, proposed in [4] as subroutines. In order to solve the Batch-MRSTA problem, we propose three methods, namely: *Max Flow-based Task Grouping*, *Direct Assignment with Road Segments*, and *Direct Assignment with Junctions*. The proposed methods follow a locally optimal assignment strategy as the SC-server has no knowledge about the arrival times of the new workers [4]. Therefore, at every instance of time, the SC-server tries to assign the available tasks to the available workers.

#### 3.1 Max Flow-based Task Grouping

The offline-MRSTA problem can be reduced to a maximum flow problem. Therefore, we can use any algorithm that can compute the maximum flow in the network to solve the offline-MRSTA problem, like the Ford-Fulkerson algorithm [5]. We can solve the batch-MRSTA problem by solving the offline-MRSTA problem at each time instance. However, this method does not consider the inherent nature of road networks for assigning the tasks. Consider the example in Fig. E.2a, there are three workers and seven road segment tasks in the example. With the max flow-based task grouping (TG) method (See Algorithm 27), the worker  $w_2$  can go to the junction where the four road segments  $(t_1, t_2, t_3, t_4)$  meet ( $J_1$ ), to perform the four tasks, instead of visiting individual road segment tasks. Thus, the task grouping method can achieve the objective of assigning seven road segment tasks by assigning workers to only two junction tasks, ( $J_1$  and  $J_2$ ). As only worker  $w_2$  can reach  $J_1$ , it will be assigned to her, similarly  $J_2$  for  $w_3$  (See Fig. E.2b).

To proceed further, let us formally define junction tasks and the cost function of a junction task.

**Junction task  $j$**  : A junction task  $j$  is an intersecting point of two or more road segment tasks.  $j = \langle jid, l_j, \langle rst_1, rst_2, \dots \rangle \rangle$ , where  $jid$  is the junction id and  $\langle rst_1, rst_2, \dots \rangle$  are the list of road segment tasks meeting at location

### 3. Algorithms

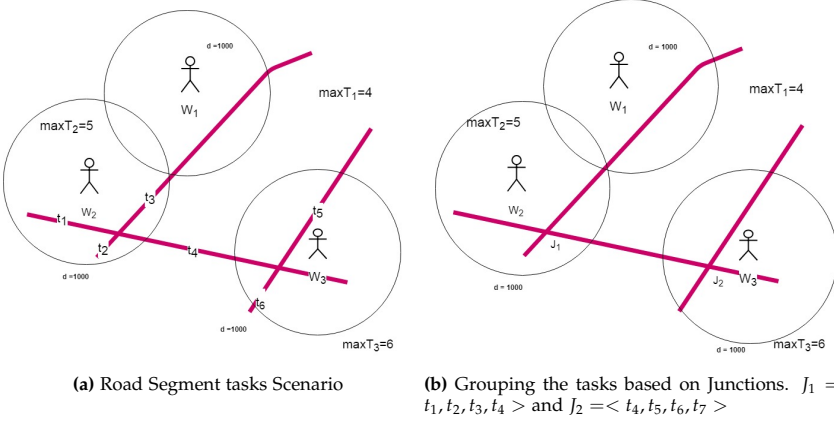


Fig. E.2: Grouping the tasks based on Junctions.

$l_j$  of the junction  $j$ .

**Cost Function of a Junction task:** The cost function of a junction task  $c(j, w)$  is :  $c(j, w) = dist(j, w)$ , where  $dist(j, w)$  is the distance between the junction task's location and the worker's location.

Once the tasks are grouped at different junctions, the assignment problem can be solved by following the solution for the minimum-cost maximum task assignment problem [4]. The total number of assignments of road segment tasks can be extrapolated from the number of junction tasks assigned.

### 3.2 Direct Assignment with Road Segments

Apart from the max flow-based task grouping method mentioned above, we have considered the direct method of assigning the  $maxT$  nearest road segment tasks to every worker, that is closer than the threshold *distance* value. Following this heuristic method, we propose *Direct Assignment with Road Segments (DA-RS)* (See Algorithm 28), which will be searching for the nearest  $maxT$  road segment tasks for each worker in the current batch at time instance  $i$  and assign them to the workers if they satisfy the distance constraint.

### 3.3 Direct Assignment with Junctions

Similar to the DA-RS method, we have considered the direct method of assigning the  $maxT$  nearest junction tasks to every worker, that is closer than the threshold *distance* value. Following this heuristic-based method, we propose *Direct Assignment with Junctions (DA-J)*. This heuristic-based method will be searching for nearest  $maxT$  junction tasks for each worker in the current batch and assign them to the workers if they satisfy the distance constraint.

**ALGORITHM 27: MAX FLOW-BASED TASK GROUPING (TG)**

**Input:** A set of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of junction tasks  $J$  at time instance  $i$ .  
 $OnMR(J, W)$  is the optimal set of assignments before the time instance  $i$ .

**Output:** The Optimal set of assignments  $OnMR(J, W + W_i)$  at time instance  $i$

```

1  $OnMR(J, W + W_i) \leftarrow NULL$ 
2  $capacity \leftarrow 1$ 
3 if  $W_i \neq \emptyset$  then
4    $V \leftarrow W_i \cup J$ 
5   foreach  $w \in W_i$  do
6      $E.addEdge(V_0, w, maxT, 0)$ 
7     foreach  $j \in J$  do
8        $E.addEdge(j, V_{|V|+1}, capacity, 0)$ 
9       if  $dist(j, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
10         $E.addEdge(w, j, capacity, dist(j, w))$ 
11   construct flow network graph  $G(V, E)$ 
12   calculate the min travel cost maximum flow
13   find the assignment  $OnMR(J, W_i)$ 
14   Update maxT values of  $W_i$ 
15    $OnMR(J, W + W_i) \leftarrow OnMR(J, W) \cup OnMR(J, W_i)$ 
16 return  $OnMR(J, W + W_i)$ 

```

**ALGORITHM 28: DIRECT ASSIGNMENT WITH ROAD SEGMENTS**

**Input:** A non-emptyset of workers  $W_i$  with  $MaxT(W_i)$  as maximum tasks accepted and  $Dist(W_i)$  as maximum distance, a set of road segment tasks  $RST_i$  at time instance  $i$ .  $OnMR(RST, W)$  is the optimal set of assignments before the time instance  $i$

**Output:** The Optimal set of assignments  $OnMR(RST, W + W_i)$  at time instance  $i$

```

1  $OnMR(RST, W + W_i) \leftarrow NULL$ 
2 foreach  $w \in W_i$  do
3   foreach  $rst \in RST$  do
4     if  $dist(rst, w) \leq Dist(w) \wedge MaxT(w) > 0$  then
5       add assignment  $\langle w, rst \rangle$  to  $OnMR(RST, W_i)$ 
6        $MaxT(w) \leftarrow MaxT(w) - 1$ 
7      $OnMR(RST, W + W_i) \leftarrow OnMR(RST, W) \cup OnMR(RST, W_i)$ 
8 return  $OnMR(RST, W + W_i)$ 

```

## 4 Experimental Evaluation

**DataSet:** It is hard to find real datasets that reflect the workers in the real world. Consequently, we have used synthetic datasets for the number of workers ranging from 1K to 50K. The workers are distributed randomly in the

#### 4. Experimental Evaluation

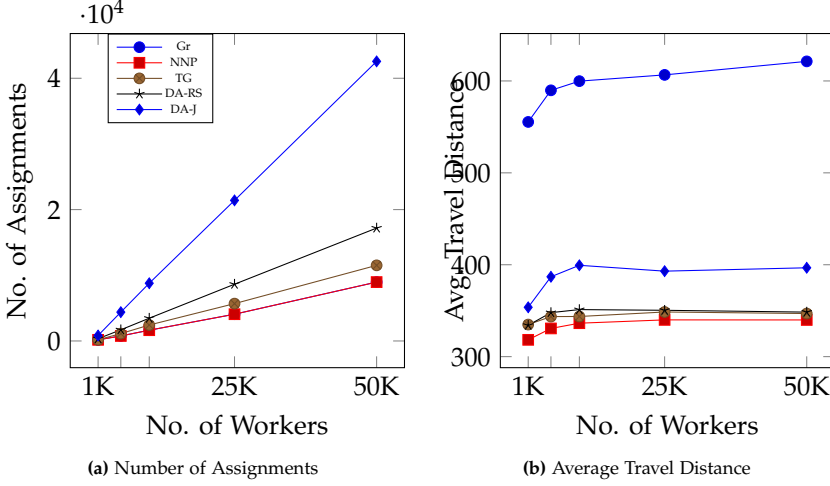


Fig. E.3: Effect of varying worker size

geographical extent of the extracted road network. The workers are generated with uniform values of *maxT* and *distance* parameters. We have extracted the OSM road segments in the Aalborg region of Denmark for the road segment tasks. Furthermore, we have derived the junctions in the extracted OSM road network, that connect two or more road segments. Aalborg Road network generated 17,503 road segment and 7,203 junction tasks.

**Algorithms, Configuration and Measures:** We have conducted our evaluation based on the assignment methods presented in this paper along with two baseline max flow-based methods mentioned in [4]: Greedy (Gr) and Nearest Neighbor Priority (NNP). We compare the different assignment methods based on the following measures: the number of road segments assigned to the workers and the average travel cost for the assigned road segment or junction task. To simulate a real application scenario, we simulate a batch of workers (for, e.g. 50) for every time instance, which will be assigned to different road segments and junction tasks. All algorithms were implemented in Java utilizing Postgresql (<https://www.postgresql.org/>) with the PostGIS and pgRouting extensions. All experiments were conducted on the Windows 8.1 OS with Intel Core i7-5600 CPU@ 2.60G HZ and 12 GB memory.

**Scalability with the size of workers data set:** Figure E.3a illustrates the effect of the varying amount of workers on the number of assigned road tasks directly or indirectly through junction tasks. With respect to the number of assigned road segments, DA-RS and DA-J have outperformed the other methods. Especially, DA-J yielded the best result when compared to the other methods. When compared to Gr, the number of road segment assignments has increased by five times, while the average travel distance for a task reduced by 35 % (See Fig. E.3b). It can also be noticed that TG outperforms DA-J with respect to average travel distance (See Fig. E.3b). The reason is

due to the additional road segment tasks covered by *DA-J* when compared to *TG*.

Similarly, the *DA-J* method also outperforms *NNP* by five times for the number of assigned road segments. However, we notice that the *NNP* has less average travel distance for a task than the *DA-J*. The reason is that *DA-J* requires workers to visit the junctions instead of the nearest road segments for performing the task.

Furthermore, with regard to the number of road segment task assignments, *DA-RS* outperforms *Gr*, *NNP*, *TG* by more than two times. In addition, *TG* outperforms the two baselines in terms of the number of assigned road segments by nearly 50 %. The average travel distance per task in the *DA-RS* increases marginally when compared with the *NNP*.

## 5 Related Work

Push-based SC, in the context of the assignment of workers to tasks, has the SC-server choosing the workers for the tasks [3]. Typically, these assignment problems aim to achieve optimization goals like maximizing the number of tasks assigned [4, 6], minimizing the cost incurred by the server [7], improving the quality of task responses [2] or goals benefitting the workers like maximizing the reward received by the worker [1].

We believe the SAT publishing mode of SC opens up a new avenue for OpenStreetMap (OSM), with regard to enriching the existing OSM entities with additional tags, verifying the existing tags of OSM entities, and creating new OSM entities with appropriate tags. The main advantage of using SAT is to accrue accurate ground truth information, as the worker assigned to the OSM entity is required to visit the OSM entity physically. We believe that we are the first to study the potential of SAT for enriching the OSM routing network. Also, our proposed method can potentially be applied for emerging crowdsourcing-based data-driven applications, such as traffic flow analysis for vehicular communications [8], and urban voluntary services [9].

## 6 Conclusions

In this paper, we proposed heuristic-based methods for assigning road segment tasks to workers for enriching the OSM road networks. We proposed three methods (*TG*, *DA-RS*, and *DA-J*) for enabling the OSM road segments assignments to the workers. The methods (*TG* and *DA-J*) are built on the notion that the workers can provide information to all the road segments that meet at the visited junctions. With respect to the number of assigned road segments, we observed that *DA-RS* and *DA-J* outperforms the baselines by two and five times, respectively .



## Acknowledgements

*This research has been funded in part by the European Commission through the Erasmus Mundus Joint Doctorate Programme IT4BI-DC. Xike Xie is supported by the CAS Pioneer Hundred Talents Program, Fundamental Research Funds for the Central Universities, National Natural Science Foundation of China (No. 61772492), and Natural Science Foundation of Jiangsu Province (No. BK20171240).*

## References

- [1] C. Chen, S.-F. Cheng, A. Gunawan, A. Misra, K. Dasgupta, and D. Chander, "Traccs: A framework for trajectory-aware coordinated urban crowd-sourcing," in *HCOMP*, 2014, pp. 30–40.
- [2] P. Cheng, X. Lian, L. Chen, and C. Shahabi, "Prediction-based task assignment in spatial crowdsourcing," in *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE, 2017, pp. 997–1008.
- [3] S. R. B. Gummidi, X. Xie, and T. B. Pedersen, "A survey of spatial crowdsourcing," *ACM Transactions on Database Systems (TODS)*, vol. 44, no. 2, p. 8, 2019.
- [4] L. Kazemi and C. Shahabi, "Geocrowd: enabling query answering with spatial crowdsourcing," in *GIS*, 2012, pp. 189–198.
- [5] J. Kleinberg and E. Tardos, *Algorithm design*. Pearson Education India, 2006.
- [6] H. To, C. Shahabi, and L. Kazemi, "A server-assigned spatial crowdsourcing framework," *ACM Transactions on Spatial Algorithms and Systems*, vol. 1, no. 1, p. 2, 2015.
- [7] L. Tran, H. To, L. Fan, and C. Shahabi, "A real-time framework for task assignment in hyperlocal spatial crowdsourcing," *ACM Transactions on Intelligent Systems and Technology*, p. 37, 2017.
- [8] Y. Wang, L. Huang, T. Gu, H. Wei, K. Xing, and J. Zhang, "Data-driven traffic flow analysis for vehicular communications," in *IEEE INFOCOM 2014*, 2014, pp. 1977–1985.
- [9] X. Xie, P. Jin, M. L. Yiu, J. Du, M. Yuan, and C. S. Jensen, "Enabling scalable geographic service sharing with weighted imprecise voronoi cells," *TKDE*, vol. 28, no. 2, pp. 439–453, 2016.

ISSN (online): 2446-1628  
ISBN (online): 978-87-7210-901-5

AALBORG UNIVERSITY PRESS