Synchronizing Strategies under Partial Observability

Larsen, Kim Guldstrand; Laursen, Simon; Srba, Jiri

Published in:
CONCUR 2014 – Concurrency Theory

DOI (link to publication from Publisher):
10.1007/978-3-662-44584-6_14

Publication date:
2014

Document Version
Accepted author manuscript, peer reviewed version

Citation for published version (APA):
Larsen, K. G., Laursen, S., & Srba, J. (2014). Synchronizing Strategies under Partial Observability. In P. Baldan, & D. Gorla (Eds.), CONCUR 2014 – Concurrency Theory (Vol. 8704, pp. 188-202). Springer Publishing Company. https://doi.org/10.1007/978-3-662-44584-6_14

Synchronizing Strategies under Partial Observability

Kim G. Larsen, Simon Laursen, and Jiří Srba

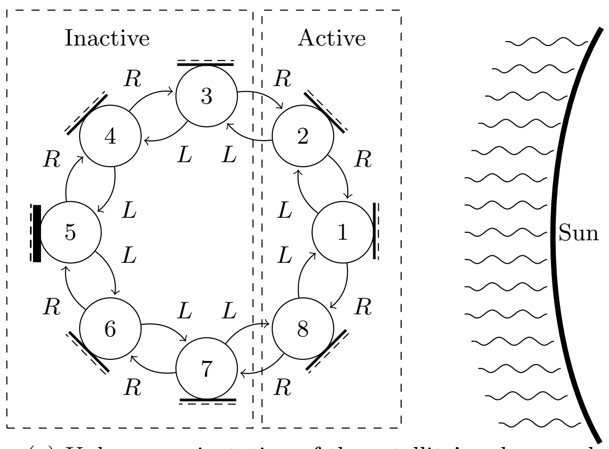
Aalborg University, Department of Computer Science
Selma Lagerlöfs Vej 300, 9220 Aalborg East, Denmark
{kgl,simlau,srba}@cs.aau.dk

Abstract. Embedded devices usually share only partial information about their current configurations as the communication bandwidth can be restricted. Despite this, we may wish to bring a failed device into a given predetermined configuration. This problem, also known as resetting or synchronizing words, has been intensively studied for systems that do not provide any information about their configurations. In order to capture more general scenarios, we extend the existing theory of synchronizing words to synchronizing strategies, and study the synchronization, short-synchronization and subset-to-subset synchronization problems under partial observability. We provide a comprehensive complexity analysis of these problems, concluding that for deterministic systems the complexity of the problems under partial observability remains the same as for the classical synchronization problems, whereas for nondeterministic systems the complexity increases already for systems with just two observations, as we can now encode alternation.

1 Introduction

In February last year (2013), Aalborg University launched an experimental satellite [3] designed by students. There was a failure during the initialization phase executed by the satellite at the orbit, resulting in unknown orientation of the solar panel. This caused significant problems with energy supply and very limited communication capabilities of the satellite, especially when transmitting information that is energetically more expensive than receiving it. The task was to command the satellite from the Earth so that it returned to some predefined well-known position.

A simplified model of the problem is depicted in Figure 1a. In the example, we assume for illustration purposes that there are only eight possible rotation positions of a single solar panel, numbered by 1 to 8 in the figure. The thin lines with a dashed surface indicate the direction the panel is facing in a given position. This determines whether the panel is active and produces energy (facing towards light) or inactive and does not produce any energy. The thick line at position 5 indicates the current (unknown) position of the solar panel. The satellite cannot communicate the exact position of the solar panel, instead it is only capable of transmitting information as to whether the current position produces energy



(a) Unknown orientation of the satellite's solar panel

```
*Synchronizing strategy that brings the system to state 1:*

**if** Inactive **then**
  **repeat** rotate $L$
  **until** Active;
**else repeat** rotate $R$
   **until** Inactive;
   rotate $L$;
**endif**
rotate $L$;
```

(b) Synch. strategy

Fig. 1: Satellite with partial observability and its synchronizing strategy

(observation Active) or not (observation Inactive). The panel can be commanded to rotate one position to the left (action L) or to the right (action R) and our task is to bring it from any possible (unknown) position into the position 1 where it produces most energy. As we cannot observe the actual position of the panel, we need to find a strategy that relies only on the fact whether the panel is Active or Inactive. Such a strategy indeed exists as shown in Figure 1b.

The classical concept of synchronizing words [6] for deterministic finite automata dates back more than 50 years and it concerns the existence of a word that brings a given automaton from any of its states to a single state (see [22,24] for recent survey papers). However, for our example in Figure 1a it is clear that there is no single synchronizing word—in this classical setting—over $\{L, R\}$ that can bring the panel into the same position. Instead, we need to design a strategy that relies on a partial information about the system, in our case on whether the panel is Active or Inactive.

1.1 Our Contribution

We introduce a general synthesis problem for synchronizing *strategies* of systems with *partial observability.* We deal with this problem in the setting of finite-state automata where each state has a single observation from a finite set of observations; we call the model *labelled transition system with partial observability* (LTSP). The task is to suggest a strategy for adaptive generation of actions based on the so-far seen observations. Such a strategy should bring the system from any possible initial state into a single synchronizing state. We also consider two other variants of the synchronization synthesis problem (i) with a bound on the maximal length of (runs of) the strategy (short-synchronization) and (ii) synchronization from a given set of states to a given set of states (subset-to-subset synchronization). We provide a comprehensive complexity study of these problems in the setting of total deterministic (DFA), partial deterministic (PFA) and

		Classical synchronization	**Partial Observability**
		No information, $\lvert\mathcal{O}\rvert = 1$	No restriction on $\mathcal{O}$
Synchronization	DFA	NL-complete [6, 24]	**NL-complete** (Thm. 5)
	PFA	PSPACE-complete [15]	**PSPACE-complete** (Thm. 4)
	NFA	PSPACE-complete [15, 21]	**EXPTIME-complete** (Thm. 2, 7)
Short-synch.	DFA	NP-complete [11]	**NP-complete** (Thm. 5)
	PFA	PSPACE-complete [15]	**PSPACE-complete** (Thm. 4)
	NFA	PSPACE-complete [15]	**EXPTIME-complete** (Thm. 2, 7)
Subset-to-subset	DFA	PSPACE-complete [20]	**PSPACE-complete** (Thm. 4)
	PFA	PSPACE-complete ([20], on-the-fly)	**PSPACE-complete** (Thm. 4)
	NFA	PSPACE-complete ([20], on-the-fly)	**EXPTIME-complete** (Thm. 2, 6)

Table 1: Summary of complexity results (new results are in bold)

nondeterministic (NFA) finite automata. Our results, compared to the classical synchronization problems, are summarized in the right column of Figure 1.

Our first technical contribution is a translation from the synthesis of history-dependent synchronizing strategies on LTSP to the synthesis of memoryless winning reachability strategies for a larger two-player knowledge game. This allows us to argue for the EXPTIME containment of the synchronization problem on NFA. However, for DFA and PFA the knowledge game is insufficient to obtain tight complexity upper-bounds. For this reason, and as our second contribution, we define a notion of aggregated knowledge graph allowing us to derive a PSPACE containment for PFA and NL containment for DFA, despite the double-exponential size of the aggregated knowledge graph in the general nondeterministic case.

In order to complement the complexity upper-bounds with matching lower-bounds, we provide as our third contribution a novel polynomial-time reduction from alternating linear bounded automata into the synchronization problems for NFA with partial observability. This is achieved by showing that a combination of the partial observability and nondeterminism can capture alternation. This technique provides matching lower-bounds for all our three synchronization problems on NFA. The lower-bounds for DFA and PFA are derived from the classical problem setting.

In addition, we describe a polynomial-time reduction from a setting with an arbitrary number of observations to an equivalent setting with just two observations, causing only a logarithmic overhead as a factor in the size of the system. Thus all the lower-bound results mentioned earlier remain valid even when restricting the synchronizing strategy synthesis problem to only two observations.

1.2 Related work

The study of synchronizing words initiated by Černý [6] is a variant of our more general strategy synthesis problem where all states return the same observation, and the existence of synchronizing words, short synchronizing words and subset-to-subset synchronizing words have been in different contexts studied up to now; see [22, 24] for recent surveys. The computational complexities of word synchronization problems for DFA, PFA and NFA are summarized in left column of Table 1. Note that the NL-completeness for the classical synchronization problem on DFA (not explicitly mentioned in the literature) follows directly from the fact that the problem of synchronizing all states is equivalent to checking that any given pair of states can be synchronized [6, 24]. The PSPACE containment of subset-to-subset word synchronization for NFA and PFA follows from [20] by running the algorithm for DFA in an on-the-fly manner, while guessing step-by-step the synchronizing path.

Through the last years there has been an increasing interest in novel settings of the synchronization problem. Volkov et al. [12] study the problem for deterministic automata with positive cost on transitions, and constrain the cost of the synchronizing word. They also study a synchronization game, where the player who wants to synchronize the system proposes only every second character in the synchronizing word. Doyen et al. [9, 10] study the existence of infinite synchronizing words in a probabilistic setting. The theory of synchronizing words have also seen several practical applications, for instance in biocomputing [2], model-based testing [4], and robotics [1].

The notion of homing sequences [13, 16] is related to the study of synchronizing words and to our study of synchronizing strategies. A homing sequence is a sequence of input actions that makes it possible to determine the current state of the system by looking at the outputs from the system. Homing sequences are studied on the model of Mealy machine, essentially a DFA where each transition produces an output from a given finite alphabet (see [22] for a recent survey). Homing sequences have been, among others, studied in an adaptive variant where the next input symbol is determined by the knowledge of the previous outputs. This is related to our synchronizing strategies that depend on the history of observations, however, there are no complexity results for adaptive homing sequence on nondeterministic systems.

Pomeranz and Reddy [17] suggest to combine synchronizing words and adaptive homing sequences. They first apply a homing sequence and then find a word that brings the machine to one particular state. The theory is applied to sequential circuit testing for deterministic systems and their adaptive synchronization problem can be seen as a subclass of our systems with partial observability (the output actions of a Mealy machine can be encoded as observations).

The idea of gathering the knowledge of possible states where the system can be after a sequence of observable actions, formalized in the notion of knowledge game, is inspired by a similar technique from [5, 7]. Our aggregated knowledge graph technique is related to the super graph construction used in [14]. The complexity of the conditional planning problem from artificial intelligence have also recently been studied under different observability assumptions [19].

Finally, regarding our EXPTIME lower bound, similar complexity results are available for reachability on finite games with partial observability. In [18] the authors study reachability games where both players have only a partial information about the current configuration of the game and show 2-EXPTIME-hardness of deciding whether the first player has a winning strategy. Our synchronization problem for NFA can be also seen as a game, however, here the second player (nondeterminism) has a full information. This variant, called semiperfect-information game, was studied in [8] for a parity objective (including reachability) and the authors show that the problem is both in NP and coNP. Our synchronization problem for NFA is similar to the semiperfect-information game, however, with a very different objective of synchronizing from any given state. This is documented by the fact that the synchronization problem under partial observability for NFA becomes EXPTIME-complete.

2 Definitions

We shall now formally rephrase our problem. We define labelled transition systems with partial observability, introduce synchronizing strategies and formulate the three decision problems we are interested in.

Definition 1. *A* labelled transition system with partial observability *(LTSP) is a quintuple $T = (S, Act, \to, \mathcal{O}, \gamma)$ where S is a set of states, Act is an action alphabet, $\to \subseteq S \times Act \times S$ is the transition relation, written $s \xrightarrow{a} s'$ whenever $(s, a, s') \in \to$, $\mathcal{O}$ is a nonempty set of observations, and $\gamma : S \to \mathcal{O}$ is a function mapping each state to an observation.*

We shall study the synchronization problems for three natural subclasses of LTSP, namely DFA (deterministic finite automata), PFA (partial finite automata) and NFA (nondeterministic finite automata). An LTSP is called NFA if S, Act and $\mathcal{O}$ are all finite sets. If the transition relation is also deterministic, i.e. for every $s \in S$ and $a \in Act$ there is at most one $s' \in S$ such that $s \xrightarrow{a} s'$, then we call it PFA. If the transition relation is moreover complete, i.e. for all $s \in S$ and $a \in Act$ there is exactly one $s' \in S$ such that $s \xrightarrow{a} s'$, then we have a DFA. In the rest of the paper we focus on the NFA class and its PFA and DFA subclasses (implicitly assuming partial observability).

For the rest of this section, let $T = (S, Act, \to, \mathcal{O}, \gamma)$ be a fixed LTSP. A *path* in T is a finite sequence $\pi = s_1 a_1 s_2 a_2 \dots a_{n-1} s_n$ where $s_i \xrightarrow{a_i} s_{i+1}$ for all i, $1 \le i < n$. The length of π is the number of transitions, denoted as $|\pi| = n - 1$. The last state s_n in such a path π is referred to as $last(\pi)$. The set of all finite paths in T is denoted by $paths(T)$. The observation sequence of π is the unique sequence of state observations $\gamma(\pi) = \gamma(s_1)\gamma(s_2)\ldots\gamma(s_n)$.

A *strategy* on T is a function from finite sequences of observations to next actions to be taken, formally

$$\delta : \mathcal{O}^+ \to Act \cup \{done\}$$

where $done \notin Act$ is a special symbol signalling that the achieved path is maximal. In the rest of the paper we consider only strategies that are feasible and terminating. A strategy δ is *feasible* if the action proposed by the strategy is executable from the last state of the path; formally we require that for every $\pi = s_1a_1s_2a_2\ldots a_{n-1}s_n \in paths(T)$ that *follows* the strategy, meaning that $\delta(\gamma(s_1a_1s_2a_2\ldots s_i)) = a_i$ for all i, $1 \leq i < n$, either $\delta(\gamma(\pi)) = done$ or there is at least one $s' \in S$ such that $last(\pi) \xrightarrow{\delta(\gamma(\pi))} s'$. A strategy δ is *terminating* if it does not generate any infinite path, in other words there is no infinite sequence $\pi = s_1a_1s_2a_2\ldots$ such that $s_i \xrightarrow{a_i} s_{i+1}$ and $\delta(\gamma(s_1a_1s_2a_2\ldots s_i)) = a_i$ for all $i \geq 1$.

Given a subset of states $X \subseteq S$ and a feasible, terminating strategy δ, the set of all maximal paths that follow the strategy δ in T and start from some state in X, denoted by $\delta[X]$, is defined as follows:

$$\begin{aligned}\delta[X] = \{\pi = s_1a_1s_2a_2\ldots a_{n-1}s_n \in paths(T) \mid s_1 \in X \text{ and } \delta(\gamma(\pi)) = done \\ \text{and } \delta(\gamma(s_1a_1s_2a_2\ldots s_i)) = a_i \text{ for all } i,\ 1 \leq i < n\ \}\ .\end{aligned}$$

The set of final states reached when following δ starting from X is defined as $last(\delta[X]) = \{last(\pi) \mid \pi \in \delta[X]\}$ and the length of δ from X is defined as $length(\delta[X]) = \max\{|\pi| \mid \pi \in \delta[X]\}$. By $length(\delta)$ we understand $length(\delta[S])$.

We now define a synchronizing strategy that guarantees to bring the system from any of its states into a single state.

Definition 2 (Synchronizing strategy). *A strategy δ for an LTSP $T = (S, Act, \to, \mathcal{O}, \gamma)$ is* synchronizing *if δ is feasible, terminating and $last(\delta[S])$ is a singleton set.*

Note that synchronizing strategy for NFA means that any execution of the system (for all possible nondeterministic choices) will synchronize into the same singleton set. It is clear that a synchronizing strategy can be arbitrarily long as it relies on the full history of observable actions. We will now show that this is in fact not needed as we can find strategies that do not perform unnecessary steps.

Let $T = (S, Act, \to, \mathcal{O}, \gamma)$ be an LTSP and let $\omega \in \mathcal{O}^+$ be a sequence of observations. We define the set of possible states (called belief) where the system can be after observing the sequence ω by

$$belief(\omega) = \{last(\pi) \mid \pi \in paths(T),\ \gamma(\pi) = \omega\}\ .$$

A strategy δ for T is *belief-compatible* if for all $\omega_1, \omega_2 \in \mathcal{O}^+$ with $belief(\omega_1) = belief(\omega_2)$ we have $\delta(\omega_1) = \delta(\omega_2)$.

Lemma 1. *If there is a synchronizing strategy δ for a finite LTSP $T = (S, Act, \rightarrow, \mathcal{O}, \gamma)$ then T has also a belief-compatible synchronizing strategy δ' such that $length(\delta') \leq 2^{|S|}$ and $length(\delta') \leq length(\delta)$.*

We shall now define three versions of the synchronization problem studied in this paper. The first problem simply asks about the existence of a synchronizing strategy.

Problem 1 (Synchronization). Given an LTSP T, is there a synchronizing strategy for T?

The second problem of short-synchronization moreover asks about the existence of a strategy shorter than a given length bound. This can be, for instance, used for finding the shortest synchronizing strategy via the bisection method.

Problem 2 (Short-Synchronization). Given an LTSP T and a bound $k \in \mathbb{N}$, is there a synchronizing strategy δ for T such that $length(\delta) \leq k$?

Finally, the general subset-to-subset synchronization problem asks to synchronize only a subset of states, reaching not necessarily a single synchronizing state but any state from a given set of final states.

Problem 3 (Subset-to-Subset Synchronization). Given an LTSP T and subsets $S_{from}, S_{to} \subseteq S$, is there a feasible and terminating strategy δ for T such that $last(\delta[S_{from}]) \subseteq S_{to}$?

If we restrict the set of observations to a singleton set (hence the γ function does not provide any useful information about the current state apart from the length of the sequence), we recover the well-known decision problems studied in the body of literature related to the classical word synchronization (see e.g. [22, 24]). Note that in this classical case the strategy is now simply a fixed finite sequence of actions.

3 Complexity Upper-Bounds

In this section we shall introduce the concept of knowledge game and aggregated knowledge graph so that we can conclude with the complexity upper-bounds for the various synchronization problems with partial observability.

3.1 Knowledge game

Let $T = (S, Act, \rightarrow, \mathcal{O}, \gamma)$ be a fixed LTSP. We define the set of successors from a given state $s \in S$ under the action $a \in Act$ as $succ(s, a) = \{s' \mid s \xrightarrow{a} s'\}$. For $X \subseteq S$ we define

$$succ(X, a) = \begin{cases} \{s' \in succ(s, a) \mid s \in X\} & \text{if } succ(s, a) \neq \emptyset \text{ for all } s \in X \\ \emptyset & \text{otherwise} \end{cases}$$

such that $succ(X, a)$ is nonempty iff every state from X enables the action a. We also define a function $split : 2^S \to 2^{(2^S)}$

$$split(X) = \{\{s \in X \mid \gamma(s) = o\} \mid o \in \mathcal{O}\} \setminus \emptyset$$

that partitions a given set of states X into the equivalence classes according to the observations that can be made.

We can now define the knowledge game, a two-player game played on a graph where each node represents a belief (a set of states where the players can end up by following a sequence of transitions). Given a current belief, *Player 1* plays by proposing a possible action that all states in the belief can perform. *Player 2* then determines which of the possible next beliefs (partitionings) the play continues from. *Player 1* wins the knowledge game if there is a strategy so that any play from the given initial belief reaches the same singleton belief $\{s\}$. Formally, we define the knowledge game as follows.

Definition 3. *Given an LTSP $T = (S, Act, \to, \mathcal{O}, \gamma)$, the corresponding* knowledge game *is a quadruple $G(T) = (\mathcal{V}, \mathcal{I}, Act, \Rightarrow)$ where*

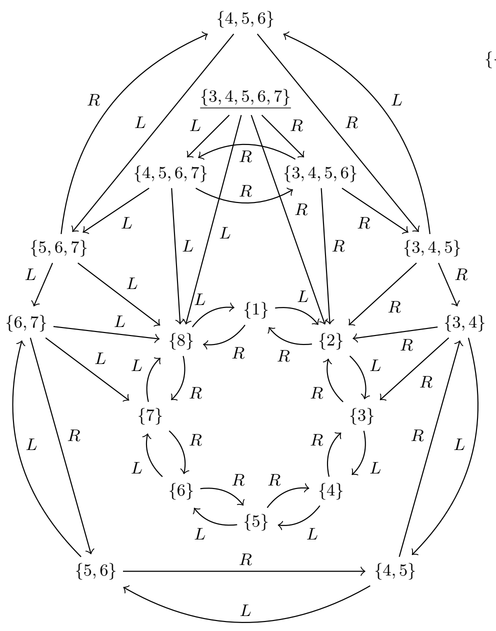
– $\mathcal{V} = \{V \in 2^S \setminus \emptyset \mid \{V\} = split(V)\}$ *is the set of all unsplittable beliefs,*
– $\mathcal{I} = split(S)$ *is the set of initial beliefs, and*
– $\Rightarrow \subseteq \mathcal{V} \times Act \times \mathcal{V}$ *is the transition relation, written $V_1 \overset{a}{\Rightarrow} V_2$ for $(V_1, a, V_2) \in \Rightarrow$, such that $V_1 \overset{a}{\Rightarrow} V_2$ iff $V_2 \in split(succ(V_1, a))$.*

Example 1. In Figure 2a we show the knowledge game graph for our running example from Figure 1a. We only display the part of the graph reachable from the initial belief consisting of states $\{3, 4, 5, 6, 7\}$ where the solar panel is inactive. Assume that we want to synchronize from any of these states into the state 8. This can be understood as a two-player game where from the current belief *Player 1* proposes an action and *Player 2* picks a new belief reachable in one step under the selected action. The question is whether *Player 1* can guarantee that any play of the game reaches the belief $\{8\}$. This is indeed the case and the strategy of *Player 1* is, for example, to repeatedly propose the action L until the belief $\{8\}$ is eventually reached.

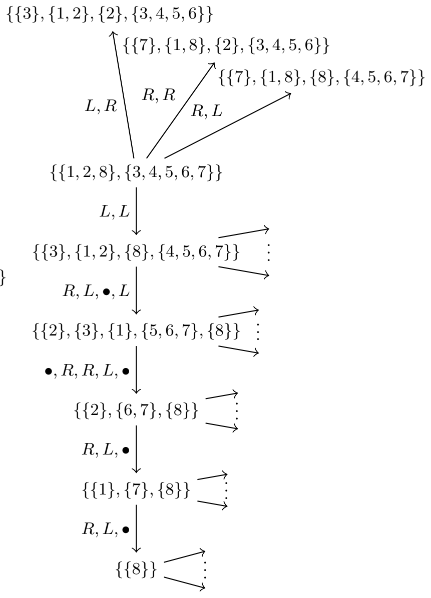
We shall now formalize the rules of the knowledge game. A *play* in a knowledge game $G(T) = (\mathcal{V}, \mathcal{I}, Act, \Rightarrow)$ is a sequence of beliefs $\mu = V_1 V_2 V_3 \ldots$ where $V_1 \in \mathcal{I}$ and for all $i \geq 1$ there is $a_i \in Act$ such that $V_i \overset{a_i}{\Rightarrow} V_{i+1}$. The set of all plays in $G(T)$ is denoted $plays(G(T))$.

A *strategy* (for *Player 1*) is a function $\rho : \mathcal{V} \to Act$. A play $\mu = V_1 V_2 V_3 \ldots$ *follows* the strategy ρ if $V_i \xRightarrow{\rho(V_i)} V_{i+1}$ for all $i \geq 1$. Note that the strategy is memoryless as it depends only on the current belief.

Player 1 wins the game $G(T)$ if there is $s \in S$ and a strategy ρ such that for every play $\mu = V_1 V_2 V_3 \ldots \in plays(G(T))$ that follows ρ there exists an $i \geq 1$ such that $V_i = \{s\}$.



(a) A knowledge game example rooted at $\{3,4,5,6,7\}$



(b) A fragment of the aggregated knowledge graph

Fig. 2: Examples of a knowledge game and an aggregated knowledge graph.

The length of a play $\mu = V_1V_2V_3\ldots$ for reaching a singleton belief $\{s\}$ is $\mathit{length}(\mu, s) = i - 1$ where i is the smallest i such that $V_i = \{s\}$. The length of a winning strategy ρ in the game $G(T)$ that reaches the singleton belief $\{s\}$ is

$$\mathit{length}(\rho) = \max_{\mu \in \mathit{plays}(G(T)),\ \mu \text{ follows } \rho} \mathit{length}(\mu, s)\ .$$

Theorem 1. *Let $T = (S, Act, \rightarrow, \mathcal{O}, \gamma)$ and let $G(T) = (\mathcal{V}, \mathcal{I}, Act, \Rightarrow)$ be the corresponding knowledge game where $\mathcal{I} = split(S)$. Then* Player 1 *wins the knowledge game $G(T)$ iff there is a synchronizing strategy for T. Moreover for any winning strategy ρ in the game $G(T)$ there is a synchronizing strategy δ for T such that $length(\rho) = length(\delta)$, and for any synchronizing strategy δ for T there is a winning strategy ρ in the game $G(T)$ such that $length(\rho) \leq length(\delta)$.*

Proof. Assume that *Player 1* wins the knowledge game $G(T)$ with the strategy ρ so that all plays reach the belief $\{s\}$. We want to find a synchronizing strategy δ for T. Let the initial observation be $o_1 \in \mathcal{O}$; this gives the initial belief $V_1 = \{t \in S \mid \gamma(t) = o_1\}$. We can now use the winning strategy ρ to determine the first action of our synchronizing strategy $\delta(o_1) = \rho(V_1)$. By executing the action $\rho(V_1)$, we get the next observation o_2. Now assume that we have a sequence of observations $o_1o_2\ldots o_{i-1}o_i$. We can inductively determine the current belief V_i

as

$$V_i = \{t \in \mathit{succ}(V_{i-1}, \rho(V_{i-1})) \mid \gamma(t) = o_i\}$$

for all $i > 1$. This gives us the synchronizing strategy

$$\delta(o_1 o_2 \dots o_{i-1} o_i) = \begin{cases} \mathit{done} & \text{if} \quad V_i = \{s\} \\ \rho(V_i) & \text{otherwise} \end{cases}$$

that guarantees that all plays follow the winning strategy ρ. Hence in any play there exists an $i \geq 1$ such that $V_i = \{s\}$. By this construction it is clear that $\mathit{length}(\rho) = \mathit{length}(\delta)$.

For the other direction, assume that there is a synchronizing strategy δ for T. Then we know from Lemma 1 that there exists also a belief-compatible synchronizing strategy δ' where $\mathit{length}(\delta') \leq \mathit{length}(\delta)$. We want to find a winning strategy ρ for *Player 1* in $G(T)$. As we know by construction that all states in a belief V have the same observation, we use the notation $\gamma(V) = o$ if $\gamma(t) = o$ for all $t \in V$. Let the initial belief be $V_1 \in \mathcal{I}$. We use the synchronizing strategy δ' to determine the first action that *Player 1* winning strategy should propose by $\rho(V_1) = \delta'(\gamma(V_1))$. Now *Player 2* determines the next belief V_2 such that $V_1 \xRightarrow{\delta'(\gamma(V_1))} V_2$. In general, assume inductively that we reached a belief V_i along the play $\mu = V_1 V_2 \dots V_i$. The winning strategy from V_i is given by

$$\rho(V_i) = \delta'(\gamma(V_1)\gamma(V_2)\dots\gamma(V_i)) \ .$$

Note that this definition makes sense because δ' is belief-compatible (and hence different plays in the knowledge game that lead to the same belief will propose the same action). From the construction of the strategy and by Lemma 1 it is also clear that $\mathit{length}(\rho) = \mathit{length}(\delta') \leq \mathit{length}(\delta)$. □

We conclude with a theorem proving EXPTIME-containment of the three synchronization problems for NFA (and hence clearly also for PFA and DFA).

Theorem 2. *The synchronization, short-synchronization and subset-to-subset synchronization problems for NFA are in EXPTIME.*

The proof is done by exploring in polynomial time the underlying, exponentially large, graph of the knowledge game.

3.2 Aggregated knowledge graph

Knowledge games allowed us to prove EXPTIME upper-bounds for the three synchronization problems on NFA, however, it is in general not possible to guess winning strategies for *Player 1* in polynomial space. Hence we introduce the so-called aggregated knowledge graph where we ask a simple reachability question (one player game). This will provide better complexity upper-bounds for deterministic systems, despite the fact that the aggregated knowledge graph can be exponentially larger than the knowledge game graph.

Definition 4. *Let $G(T) = (\mathcal{V}, \mathcal{I}, Act, \Rightarrow)$ be a knowledge game. The aggregated knowledge graph is a tuple $AG(G(T)) = (\mathscr{C}, \mathcal{C}_0, \rightrightarrows)$ where*

- *$\mathscr{C} = 2^{\mathcal{V}} \setminus \emptyset$ is the set of configurations (set of aggregated beliefs),*
- *$\mathcal{C}_0 = \mathcal{I}$ is the initial configuration (set of all initial beliefs), and*
- *$\rightrightarrows \subseteq \mathscr{C} \times \mathscr{C}$ is the transition relation such that $\mathcal{C}_1 \rightrightarrows \mathcal{C}_2$, standing for $(\mathcal{C}_1, \mathcal{C}_2) \in \rightrightarrows$, is possible if for every $V \in \mathcal{C}_1$ there is an action $a_V \in Act \cup \{\bullet\}$ such that $V \overset{a_V}{\Longrightarrow} V'$ for at least one V' (by definition $V \overset{\bullet}{\Rightarrow} V$ if and only if $|V| = 1$), ending in $\mathcal{C}_2 = \{V' \mid V \in \mathcal{C}_1 \text{ and } V \overset{a_V}{\Longrightarrow} V'\}$.*

Example 2. Figure 2b shows a fragment of the aggregated knowledge graph for our running example from Figure 1a. The initial configuration is the aggregation of the initial beliefs and each transition is labelled with a sequence of actions for each belief in the aggregated configuration. The suggested path shows how to synchronize into the state 8. Note that the action $\bullet$, allowed only on singleton beliefs, stands for the situation where the belief is not participating in the given step.

Theorem 3. *Let $G(T) = (\mathcal{V}, \mathcal{I}, Act, \Rightarrow)$ be a knowledge game and let $AG(G(T)) = (\mathscr{C}, \mathcal{C}_0, \rightrightarrows)$ be the corresponding aggregated knowledge graph. Then $\mathcal{C}_0 \rightrightarrows^* \{\{s\}\}$ for some state s if and only if* Player 1 *wins the knowledge game $G(T)$. Moreover, for any winning strategy ρ in $G(T)$ that reaches the singleton belief $\{s\}$ we have $\mathcal{C}_0 \rightrightarrows^{length(\rho)} \{\{s\}\}$, and whenever $\mathcal{C}_0 \rightrightarrows^n \{\{s\}\}$ then there is a winning strategy ρ in $G(T)$ such that $length(\rho) \leq n$.*

The proof is done by translating the path in the aggregated knowledge graph into a winning strategy for *Player 1* in the knowledge game, and vice versa.

The aggregated knowledge graph can in general be exponentially larger than its corresponding knowledge game as the nodes are now subsets of beliefs (that are subsets of states). Nevertheless, we can observe that for DFA and PFA, the size of configurations in $AG(G(T))$ cannot grow.

Lemma 2. *Let T be an LTSP generated by DFA or PFA. Let $AG(G(T)) = (\mathscr{C}, \mathcal{C}_0, \rightrightarrows)$ be the corresponding aggregated knowledge graph. Whenever $C \rightrightarrows C'$ then $\sum_{V \in C} |V| \geq \sum_{V' \in C'} |V'|$.*

Theorem 4. *The synchronization, short-synchronization and subset-to-subset synchronization problems for DFA and PFA are decidable in PSPACE.*

Proof. By Theorem 3 and Theorem 1 we get that we can reach the configuration $\{\{s\}\}$ for some $s \in S$ in the aggregated graph $AG(G(T))$ if and only if there is a synchronizing strategy for the given LTSP T. From Lemma 2 we know that for DFA and PFA the size of each aggregated configuration reachable during any computation is bounded by the size of the set S and therefore can be stored in polynomial space. As PSPACE is closed under nondeterminism, the path to the configuration $\{\{s\}\}$ for some $s \in S$ can be guessed, resulting in a polynomial-space algorithm for the synchronizing problem. Theorem 3 also implies that the length of the shortest synchronizing strategy in T is the same as the length of the shortest path to the configuration $\{\{s\}\}$ for some s, giving us that the short-synchronization problems for DFA and PFA are also in PSPACE. Regarding the subset-to-subset synchronization problem from the set S_{from} to the set S_{to}, we can in a straightforward manner modify the aggregated knowledge graph so that the initial configuration is produced by splitting S_{from} according to the observations and we end in any configuration consisting solely of beliefs V that satisfy $V \subseteq S_{to}$ (while allowing the action $\bullet$ from any such belief to itself). $\square$

Finally, for the synchronization and short-synchronization problems on DFA, we can derive even better complexity upper-bounds by using the aggregated knowledge graph.

Theorem 5. *The synchronization problem on DFA is in NL and the short-synchronization problem on DFA is in NP.*

The first claim is proved using our aggregated knowledge graph together with a generalization of the result from [6, 24] saying that all pairs of states in the system can synchronize iff all states can synchronize simultaneously. For the second claim we show that the shortest synchronizing strategy in DFA has length at most $(n-1)n^2$ where n is the number of states. The strategy can be guessed (in the aggregated knowledge graph) and verified in nondeterministic polynomial time.

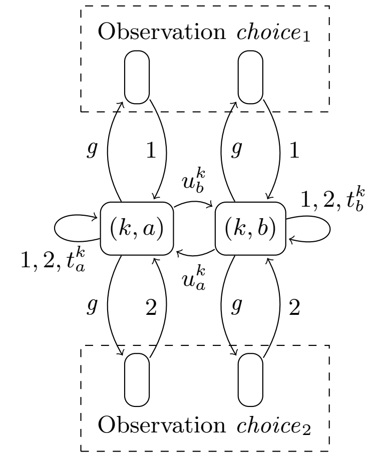
4 Complexity Lower-Bounds

We shall now describe a technique that will allow us to argue about EXPTIME-hardness of the synchronization problems for NFA.

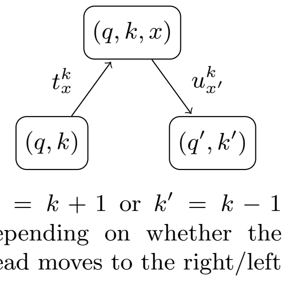
Theorem 6. *The subset-to-subset synchronization problem is EXPTIME-hard for NFA.*

Proof (Sketch). By a polynomial time reduction from the EXPTIME-complete [23] acceptance problem for alternating linear bounded automaton over the binary alphabet $\{a, b\}$. W.l.o.g. we assume that the existential and universal choices do not change the current head position and the tape content and we have special deterministic states for tape manipulation. We shall construct an LTSP over three observations $\{default, choice_1, choice_2\}$.

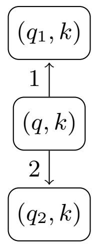
Each tape cell at position k is encoded as in Figure 3a. The actions t_a^k and t_b^k can reveal the current content of the cell, while the actions u_a^k and u_b^k are used to update the stored letter. The current control state q and the head position k are remembered via newly added states of the form (q, k). If (q, k) corresponds to a deterministic state, it will (by a sequence of two actions t_x^k and $u_{x'}^k$ as depicted in Figure 4a) test whether the k'th cell stores the required letter x and then it will update it to x'. For the pair (q, k) where q is an existential state, we add the transitions as in Figure 3c. Clearly, the strategy can select the action 1 or 2 in order to commit to one of the choices and all tape cells just mimic the selected



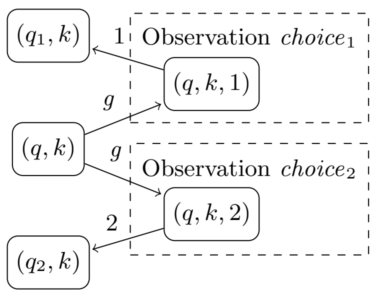
(a) Tape cell


(q, k, x)
t_x^k
$u_{x'}^k$
(q, k)
(q', k')
$k' = k + 1$ or $k' = k - 1$ depending on whether the head moves to the right/left

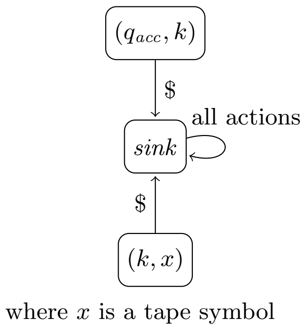
(b) Tape update



(c) Existential choice



(d) Universal choice



(e) Sink state

Fig. 3: Encoding idea

action via self-loops. So far we did not need any observations as the introduced states all belong to *default*.

The tricky part is regarding the transitions from the pair (q, k) where q is a universal state. The situation is depicted in Figure 3d. Here the strategy can propose only the action g while the nondeterminism is in control of whether we end up in $(q, k, 1)$ or $(q, k, 2)$. However, this choice is revealed by the observation *choice*$_1$ or *choice*$_2$, respectively. Notice that the nondeterminism in the cell encoding does not have to follow the same observation as in the control part. Nevertheless, if this happens, the strategy is allowed to "split" into two separate continuations.

Finally, if the accepting control state q_{acc} is reached, we allow to enter a new state *sink* under a new action \$, not only from (q_{acc}, k) but also from any cell state (k, a) and (k, b) as depiced in Figure 4d. This is the only way in which the

LTSP can synchronize, assuming that we only start from the states $(q_0, 1)$ where q_0 is the initial state and the cell positions that correspond to the initial content of the tape. This assumption is valid as we only consider the subset-to-subset synchronization problem in this theorem. □

Theorem 7. *The synchronization and short-synchronization problems are EXPTIME-hard for NFA.*

Proof (Sketch). Given the construction for the subset-to-subset synchronization problem, we need to guarantee that the execution starts from the predefined states also in the general synchronization problem. Hence we introduce additional transitions together with a new state *init* having the observation *default*. These transitions add a new action # that bring us from any state into one of the initial states of the subset-to-subset problem. There is also a new #-labelled transition from *init* into the initial control state and *init* has no other transitions. This implies that any synchronizing strategy must start by performing the action #. Note that for the short-synchronization case, we use Lemma 1 giving us an exponential upper-bound on the length of the shortest synchronizing strategy. □

The reader may wonder whether three different observations are necessary for proving EXPTIME-hardness of the synchronizing problems or whether one can show the hardness only with two. By analysis of the construction, we can observe that two observations are in fact sufficient. Moreover, there is a general polynomial-time reduction from a given synchronization problem with an arbitrary number of observations to just two observations, while increasing the size of the system by only a logarithmic factor.

Theorem 8. *The synchronization, short-synchronization and subset-to-subset synchronization problems on DFA, PFA and NFA are polynomial-time reducible to the equivalent problems with only two observations.*

Proof (Sketch). Let $T = (S, Act, \rightarrow, \mathcal{O}, \gamma)$ be a given finite LTSP and let $\ell = \lceil \log |\mathcal{O}| \rceil$. The idea is to encode every observation in binary, so that we need only ℓ bits for each observation. Now instead of entering a state s in the original system, we enter instead a chain of newly added states of length $\ell - 1$ that reveal via the binary observations 0/1 the actual observation of the state s (where s reveals the last bit). □

References

[1] D.S. Ananichev and M.V. Volkov. Synchronizing monotonic automata. *Theoretical Computer Science*, 327(3):225 – 239, 2004.

[2] Yaakov Benenson, Rivka Adar, Tamar Paz-Elizur, Zvi Livneh, and Ehud Shapiro. Dna molecule provides a computing machine with both data and fuel. *Proceedings of the National Academy of Sciences of the USA*, 100(5):2191–2196, March 2003.

[3] Thomas Boel. Årsag til fejl på aalborg-satellit: Solcellerne vendte væk fra solen. Ingeniøren (Weekly national news magazine about engineering), 8. March 2013. http://http://ing.dk/artikel/aarsag-til-fejl-paa-aalborg-satellit-solcellerne-vendte-vaek-fra-solen-156828.

[4] Manfred Broy, editor. *Model-based testing of reactive systems: advanced lectures*, volume 3472 of *LNCS*. Springer, 2005.

[5] Franck Cassez, Alexandre David, Kim G. Larsen, Didier Lime, and Jean-Franois Raskin. Timed control with observation based and stuttering invariant strategies. In *ATVA*, volume 4762 of *LNCS*, pages 192–206. Springer, 2007.

[6] Ján Černý. Poznámka k. homogénnym experimentom s konecnými automatmi. *Mat. fyz. čas SAV*, 14:208–215, 1964.

[7] Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, and Jean franois Raskin. Algorithms for omega-regular games with imperfect information. In *CSL'06*, volume 4207 of *LNCS*, pages 287–302. Springer, 2006.

[8] Krishnendu Chatterjee and Thomas A Henzinger. Semiperfect-information games. In *FSTTCS'05*, pages 1–18. Springer, 2005.

[9] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Infinite synchronizing words for probabilistic automata. In *MFCS'11*, volume 6907 of *LNCS*, pages 278–289. Springer, 2011.

[10] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Synchronizing objectives for markov decision processes. In *iWIGP*, EPTCS, pages 61–75, 2011.

[11] D. Eppstein. Reset sequences for monotonic automata. *SIAM Journal on Computing*, 19(3):500–510, 1990.

[12] Fedor Fominykh and Mikhail Volkov. P(l)aying for synchronization. In *CIAA*, volume 7381 of *LNCS*, pages 159–170. Springer, 2012.

[13] Arthur Gill. State-identification experiments in finite automata. *Information and Control*, 4(23):132 – 154, 1961.

[14] Moez Krichen. State identification. In Broy [4], pages 35–67.

[15] Pavel Martyugin. Computational complexity of certain problems related to carefully synchronizing words for partial automata and directing words for nondeterministic automata. *Theory of Com. Systems*, pages 1–12, 2013.

[16] Edward F. Moore. Gedanken Experiments on Sequential Machines. In *Automata Studies*, pages 129–153. Princeton U., 1956.

[17] Irith Pomeranz and Sudhakar M. Reddy. Application of homing sequences to synchronous sequential circuit testing. *IEEE Trans. Computers*, 43(5), 1994.

[18] John H Reif. The complexity of two-player games of incomplete information. *Journal of computer and system sciences*, 29(2):274–301, 1984.

[19] Jussi Rintanen. Complexity of conditional planning under partial observability and infinite executions. In *ECAI*, pages 678–683, 2012.

[20] I.K. Rystsov. Polynomial complete problems in automata theory. *Information Processing Letters*, 16(3):147 – 151, 1983.

[21] I.K. Rystsov. Rank of a finite automaton. *Cybernetics and Systems Analysis*, 28(3):323–328, 1992.

[22] Sven Sandberg. Homing and synchronizing sequences. In Broy [4], pages 5–33.

[23] M. Sipser. *Introduction to the Theory of Computation.* Course Technology, 2006.

[24] Mikhail V. Volkov. Synchronizing automata and the Černý conjecture. In *Language and automata theory and applications*, pages 11–27. Springer, 2008.

A Proofs from Section 2

Lemma 1. *If there is a synchronizing strategy δ for a finite LTSP $T = (S, Act, \rightarrow, \mathcal{O}, \gamma)$ then T has also a belief-compatible synchronizing strategy δ' such that $length(\delta') \leq 2^{|S|}$ and $length(\delta') \leq length(\delta)$.*

Proof. Let δ be a synchronizing strategy for T. Let $\omega_1, \omega_2 \in \mathcal{O}^+$ be such that $belief(\omega_1) = belief(\omega_2)$. Then the strategy $\delta_{\omega_1 \rightarrow \omega_2}$ defined as

$$\delta_{\omega_1 \rightarrow \omega_2}(\omega) = \begin{cases} \delta(\omega_2\omega') & \text{if } \omega = \omega_1\omega' \\ \delta(\omega) & \text{otherwise} \end{cases}$$

is also a synchronizing strategy for T; a fact that follows from the definition of $\delta[X]$ and Definition 2.

We shall now argue that if there is a synchronizing strategy for T then there is also one of length at most $2^{|S|}$. By contradiction assume that the shortest synchronizing strategy δ for T has length $length(\delta) > 2^{|S|}$. Among such shortest synchronizing strategies, we pick one that has the smallest number of paths from $paths(T)$ of length $length(\delta)$. For this strategy δ, there is now a path $\pi = s_1 a_1 s_2 a_2 \dots a_{n-1} s_n \in \delta(S)$ where $|\pi| = length(\delta) > 2^{|S|}$ such that $s_1 \in S$, $\delta(\gamma(\pi)) = done$, and $\delta(\gamma(s_1 a_1 s_2 a_2 \dots s_i)) = a_i$ for all i, $1 \leq i < n$. Let $\omega = \gamma(\pi)$ be the sequence of observations on this path. As $|\omega| > 2^{|S|}$ and we have only $2^{|S|}$ possible beliefs, necessarily $\omega = \omega_1\omega_1'$ and $\omega_1 = \omega_2\omega_2'$ such that ω_2' is nonempty and $belief(\omega_1) = belief(\omega_2)$. Then the strategy $\delta_{\omega_1 \rightarrow \omega_2}$ is also a synchronizing strategy for T but has a smaller number of the longest paths (of length $length(\delta)$) in $\delta_{\omega_1 \rightarrow \omega_2}$ than δ. This contradicts our assumption and we can conclude that the shortest synchronization strategy for T has length at most $2^{|S|}$. Clearly, $length(\delta_{\omega_1 \rightarrow \omega_2}) \leq length(\delta)$.

Assume now a synchronizing strategy δ such that $length(\delta) \leq 2^{|S|}$. Then there are only finitely many $\omega_1, \omega_2 \in \mathcal{O}^+$ that can be observed on some path that follows δ such that $belief(\omega_1) = belief(\omega_2)$ and $\delta(\omega_1) \neq \delta(\omega_2)$. We can now repeatedly use the strategy substitution $\delta_{\omega_1 \rightarrow \omega_2}$ if $|\omega_2| \leq |\omega_1|$, or $\delta_{\omega_2 \rightarrow \omega_1}$ if $|\omega_1| < |\omega_2|$, in order to construct a belief-compatible strategy for T of length at most $2^{|S|}$ a not longer than $length(\delta)$.

B Proofs from Section 3

Theorem 2. *The synchronization, short-synchronization and subset-to-subset synchronization problems for NFA are in EXPTIME.*

Proof. We shall first discuss the existence of synchronizing strategy. Let $T = (S, Act, \rightarrow, \mathcal{O}, \gamma)$ and let $G(T) = (\mathcal{V}, \mathcal{I}, Act, \Rightarrow)$ be the corresponding knowledge game where $\mathcal{I} = split(S)$. By Theorem 1 there is a synchronizing strategy for T if and only if *Player 1* has a winning strategy in the knowledge game. The game $G(T)$ is of exponential size w.r.t. the system T but we can decide whether *Player 1* has a winning strategy in $G(T)$ in polynomial time (in the size of

$G(T)$) by adapting a standard backward reachability algorithm for alternating automata. We mark some singleton belief $\{s\} \in \mathcal{V}$ (we try all possible singleton beliefs if we do not succeed for $\{s\}$). Then we iteratively mark every belief $V \in \mathcal{V}$ such that there is an action $a \in Act$ enabled in V and every V', where $V \overset{a}{\Rightarrow} V'$, is already marked. Once no more beliefs can be marked, we check whether all initial beliefs are marked. If this is the case then there is a winning strategy for *Player 1*. If on the other hand this is not the case for any singleton belief $\{s\}$ then *Player 1* does not have a winning strategy. This gives us an exponential algorithm for the synchronization problem.

Regarding the short-synchronization problem, Theorem 1 together with Lemma 1 imply that there is a synchronization strategy in T of length at most k iff there is a winning strategy for *Player 1* in $G(T)$ of length at most k. By modifying the marking algorithm so that it for each marked belief remembers also the length of the shortest path to the singleton belief, we can check whether all initial beliefs are marked such that their length is at most k.

Finally, for the subset-to-subset synchronization problem from the set S_{from} to the set S_{to}, we can modify the game $G(T)$ such that the initial beliefs are $\mathcal{I} = split(S_{from})$ and a play is winning if it reaches a belief V such that $V \subseteq S_{to}$. The corresponding modification in the marking algorithm is straightforward.

We can so conclude that all three problems are solvable (via the knowledge game graphs) in exponential time w.r.t. to the size of the finite LTSP T. $\square$

Theorem 3. *Let $G(T) = (\mathcal{V}, \mathcal{I}, Act, \Rightarrow)$ be a knowledge game and let $AG(G(T)) = (\mathscr{C}, \mathcal{C}_0, \rightrightarrows)$ be the corresponding aggregated knowledge graph. Then $\mathcal{C}_0 \rightrightarrows^* \{\{s\}\}$ for some state s if and only if* Player 1 *wins the knowledge game $G(T)$. Moreover, for any winning strategy ρ in $G(T)$ that reaches the singleton belief $\{s\}$ we have $\mathcal{C}_0 \rightrightarrows^{length(\rho)} \{\{s\}\}$, and whenever $\mathcal{C}_0 \rightrightarrows^n \{\{s\}\}$ then there is a winning strategy ρ in $G(T)$ such that $length(\rho) \leq n$.*

Proof. Assume that for some s the path $\eta = \mathcal{C}_0 \rightrightarrows^n \{\{s\}\}$ exists in the aggregated knowledge graph $AG(G(T))$. Then there is also a path $\eta' = \mathcal{C}_0 \rightrightarrows^{n'} \{\{s\}\}$ such that for each belief we always propose the same action, no matter what configuration it is in. This follows by the argument that we can use the same strategy for each belief that appears in the aggregated knowledge graph by following the one that makes the path shortest (reaches the belief $\{s\}$ faster). Clearly $|\eta'| \leq |\eta|$. Now we construct a winning strategy ρ for *Player 1* in the knowledge game $G(T)$ such that $\rho(V) = a_V$ where $V \in C \in \eta'$ and a_V is the proposed action for the belief V in the configuration C. This is a winning strategy for *Player 1* as it reaches the belief $\{s\}$. By construction of the strategy it is also clear that $length(\rho) \leq n$.

Now in the other direction, assume that *Player 1* wins the knowledge game $G(T)$ with the strategy ρ by bringing all plays to the belief $\{s\}$ where $s \in S$. We want to show that there is a path $\mathcal{C}_0 \rightrightarrows^* \{\{s\}\}$ in aggregated knowledge graph $AG(G(T))$. We start with the initial configuration $\mathcal{C}_0 = \{V_0 \mid V_0 \in \mathcal{I}\}$. Assume that we already have a prefix of the path up to the configuration $\mathcal{C}$. The next configuration is then

$$\mathcal{C}' = \{V' \mid V \in \mathcal{C} \text{ and } V \xRightarrow{a_V} V' \text{ where } a_V = \rho(V)$$
$$\text{if } V \neq \{s\} \text{ and } a_V = \bullet \text{ otherwise } \} \ .$$

It is clear that this will bring us to the configuration $\{\{s\}\}$ as all beliefs now follow the winning strategy ρ. By counting the number of steps in the path, we get that $\mathcal{C}_0 \rightrightarrows^{length(\rho)} \{\{s\}\}$. □

Lemma 2. *Let T be an LTSP generated by DFA or PFA. Let $AG(G(T)) = (\mathscr{C}, \mathcal{C}_0, \rightrightarrows)$ be the corresponding aggregated knowledge graph. Whenever $C \rightrightarrows C'$ then $\sum_{V \in C} |V| \geq \sum_{V' \in C'} |V'|$.*

Proof. Let $V \in C$ and let $X = \{V' \mid V \xRightarrow{a_V} V'\}$. It is now enough to argue that $|V| \geq \sum_{V' \in X} |V'|$. Recall that $V' \in X$ iff $V' \in split(succ(V, a_V))$. Clearly $|succ(V, a_V)| \leq |V|$ for DFA and PFA. The rest follows from the fact that the function *split* only partitions $succ(V, a_V)$ and hence does not increase its size. □

Theorem 5. *The synchronization problem on DFA is in NL and the short-synchronization problem on DFA is in NP.*

Proof. Using the same arguments as in Theorem 4, we can derive that the question of synchronizing two given states s_1 and s_2 can be done in nondeterministic logarithmic space using the aggregated knowledge graph (by Lemma 2 we know that on any path in the aggregated graph we need to remember at most two states). Observe now that we can synchronize all pairs of states independently if and only if all states can be synchronized at once (a straightforward generalization of the result from the classical setting without partial observability [6,24]).

For the containment of the short-synchronization problem in NP, we first notice that if n states can be synchronized then the shortest synchronizing strategy has length at most $(n-1)n^2$. This follows from the fact that each pair-wise synchronization takes at most n^2 steps (in the aggregated knowledge graph with configurations containing just two states we get a loop if the length of the path exceeds n^2). Clearly synchronizing one state takes zero steps. Now by induction, assume that synchronizing the first i states into a state t requires at most $(i-1)n^2$. Let s be $(i+1)$'th state that we should also synchronize with. From the initial state s, we follow the strategy for synchronizing the first i states, arriving into a state s'. Now we extend the strategy so that we pairwise synchronize t and s', taking at most n^2 steps as argued above. Hence to synchronize $i+1$ states we need at most $(i-1)n^2 + n^2 = in^2$ steps, providing us with the conclusion that synchronizing n states requires a strategy of length at most $(n-1)n^2$. Such a strategy can be guessed and verified in nondeterministic polynomial time. □

C Proofs from Section 4

First, we introduce the acceptance problem of alternating linear bounded automata that is known to be EXPTIME-complete (see e.g. [23]). For technical convenience, we present a less standard variant of the problem where only internal control states can read/write on the tape while the existential and universal states that enable branching do not modify or even know the tape content. The presented variant has the same expressive power as the standard model.

An alternating linear bounded automaton (ALBA) is a tuple $\mathcal{M} = (Q_i, Q_\forall, Q_\exists, Q, \Sigma, q_0, q_{acc}, \vdash, \dashv, \delta_1, \delta_2)$ where

- Q_i is a set of internal control states,
- $Q_\forall$ is a set of universal control states,
- $Q_\exists$ is a set of existential control states,
- $Q = Q_i \uplus Q_\exists \uplus Q_\forall$ is the set of all control states,
- $\Sigma = \{a, b\}$ is the input alphabet,
- $q_0 \in Q$ is the initial state,
- $q_{acc} \in Q$ is the accepting state,
- $\delta_1 : Q_i \times (\Sigma \cup \{\vdash, \dashv\}) \to Q \times (\Sigma \cup \{\vdash, \dashv\}) \times \{L, R\}$ is a transition relation for internal states such that
  - $\delta_1(q, \vdash) = (q', \vdash, R)$,
  - $\delta_1(q, \dashv) = (q', \dashv, L)$,
  - $\delta_1(q, x) = (q', x', D)$,

  where $q \in Q_i$, $q' \in Q$, $x, x' \notin \{\vdash, \dashv\}$ and $D \in \{L, R\}$,
- $\delta_2 : (Q_\exists \cup Q_\forall) \to Q \times Q$ is a transition relation for existential and universal states.

For each $q \in Q_\forall \cup Q_\exists$ the transition relation $\delta_2(q)$ returns a pair of two elements (q_1, q_2), referred to as the *first* and *second* successor, respectively.

A *configuration* of an ALBA $\mathcal{M}$ is the current state, the position of the head on the tape and the tape content (starting and ending with the end-markers). We denote configurations as $w_1 q w_2$ where $q \in Q$ is the current state, and $w_1 = \vdash w_1'$ and $w_2 = w_2' \dashv$ where $w_1', w_2' \in \Sigma^*$ represent the tape content such that the head points to the first letter of w_2. The *initial configuration* for the input word w is $c_0 = \vdash q_0 w \dashv$. Depending on the control state a configuration is called *internal* if $q \in Q_i$, *existential* if $q \in Q_\exists$, *universal* if $q \in Q_\forall$ and *accepting* if $q = q_{acc}$.

A *step of computation* is a relation $\to$ between configurations defined as follows (where $x, x', y \in \Sigma \cup \{\vdash, \dashv\}$, $w_1, w_2 \in (\Sigma \cup \{\vdash, \dashv\})^*$ such that all configurations start with $\vdash$ and end with $\dashv$):

- $w_1 q x w_2 \;\to\; w_1 x' q' w_2$ whenever $q \in Q_i$ and $\delta_1(q, x) = (q', x', R)$,
- $w_1 y q x w_2 \;\to\; w_1 q' y x' w_2$ whenever $q \in Q_i$ and $\delta_1(q, x) = (q', x', L)$, and
- $w_1 q w_2 \to w_1 q' w_2$ whenever $q \in Q_\exists \cup Q_\forall$, $\delta_2(q, x) = (q_1, q_2)$ and $q' = q_1$ or $q' = q_2$.

A *computation tree* for $\mathcal{M}$ on an input $w \in \Sigma^*$ is a possibly infinite configuration-labelled tree rooted with the initial configuration $c_0 = \vdash q_0 w \dashv$ such that every node labelled with a configuration c satisfies:

- if c is accepting then it is a leaf,
- if c is internal or existential then it has one child c' such that $c \to c'$, and

– if c is universal then it has two children labelled with the first and the second successor of c.

An ALBA $\mathcal{M}$ *accepts* a string $w \in \{a, b\}^*$ iff there is a finite computation tree for $\mathcal{M}$ on w such that all leaves are accepting configurations. It is well known that the problem whether an ALBA accepts a string w is EXPTIME-complete.

Theorem 6. *The subset-to-subset synchronization problem is EXPTIME-hard for NFA.*

Proof. We shall now provide a polynomial-time reduction from the acceptance problem for ALBA to synchronization problems on NFA with partial observability. Assume a given ALBA $\mathcal{M} = (Q_i, Q_\forall, Q_\exists, Q, \Sigma, q_0, q_{acc}, \vdash, \dashv, \delta_1, \delta_2)$ and a string w. We construct a finite LTSP (NFA) $T = (S, Act, \rightarrow, \mathcal{O}, \gamma)$ where

– $S = \{sink\} \cup \{(q, k) \mid q \in Q,\ 0 \leq k \leq |w| + 1\} \cup$
$\{(q, k, a), (q, k, b) \mid q \in Q_i,\ 1 \leq k \leq |w|\} \cup$
$\{(q, 0, \vdash), (q, |w| + 1, \dashv)\} \cup$
$\{(q, k, 1), (q, k, 2) \mid q \in Q_\forall,\ 0 \leq k \leq |w| + 1\} \cup$
$\{(k, x), (k, x, 1), (k, x, 2) \mid 1 \leq k \leq |w|,\ x \in \{a, b\}\ \} \cup$
$\{(0, \vdash), (0, \vdash, 1), (0, \vdash, 2)\} \cup$
$\{(|w| + 1, \dashv), (|w| + 1, \dashv, 1), (|w| + 1, \dashv, 2)\}$,
– $Act = \{t^k_x, u^k_x \mid 1 \leq k \leq |w|,\ x \in \{a, b\}\} \cup$
$\{t^0_\vdash, u^0_\vdash, t^{|w|+1}_\dashv, u^{|w|+1}_\dashv\} \cup \{g, 1, 2, \$, \#\}$,
– $\mathcal{O} = \{default, choice_1, choice_2\}$, and
– the transition relation $\rightarrow$ together with γ is given in Figure 4. Note that all states that are not marked with the observations $choice_1$ or $choice_2$, are assigned by default the observation *default*.

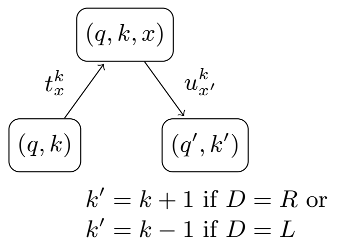
The construction above provides a reduction to the subset-to-subset synchronization problem where we assume that we synchronize from the initial states

– $(q_0, 1)$ and
– (k, x_k) for all k, $0 \leq k \leq |w| + 1$, such that the input word w is of the form $x_1 x_2 \ldots x_n$ and $x_0 = \vdash$ and $x_{n+1} = \dashv$

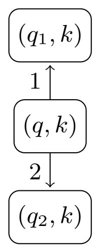
into the set $\{sink\}$. During the simulation of the given ALBA, we shall preserve the invariant that there is exactly one active state of the form (q, k) representing that we are at the control state q and the head is at position k. Also for every tape cell at position k, we remember the current symbol stored in each cell by being either in the state (k, a) or (k, b) (with the exception of the end-markers that can store only one symbol).

Consider now that the active control state is (q, k). There are four cases according to whether q is internal, existential, universal or accepting control state.

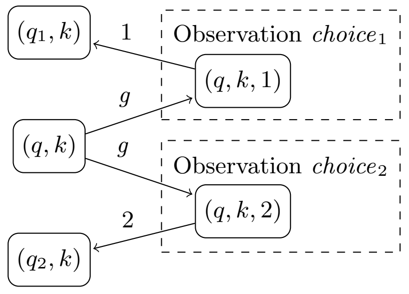
– Let $q \in Q_i$. The corresponding transitions are depicted in Figure 4a. The state (q, k) can perform the test action t^k_a or t^k_b depending on whether the k'th



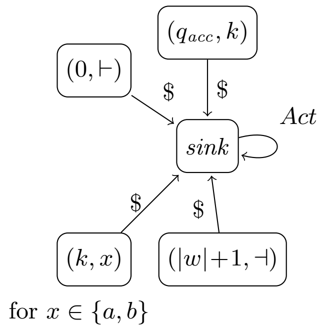
(a) $\delta_1(q,x) = (q', x', D)$, $D \in \{L, R\}$



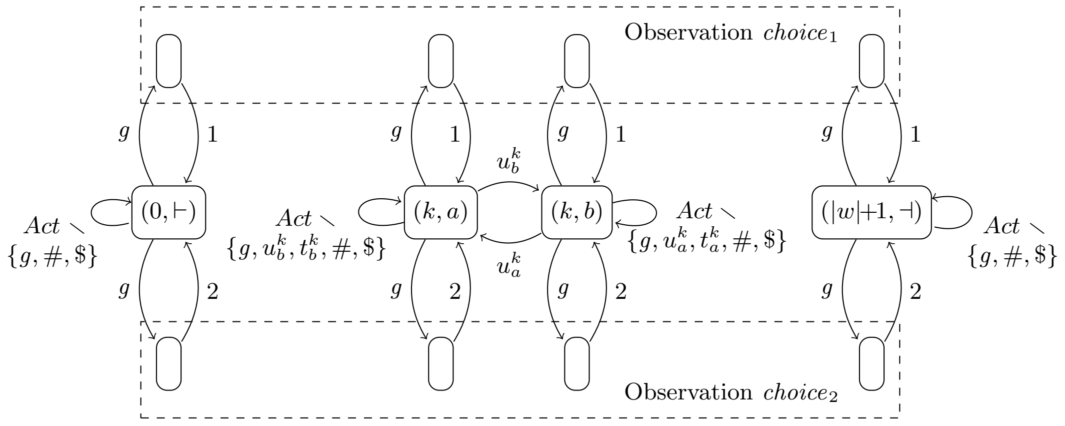
(b) $\delta_2(q) = (q_1, q_2)$, $q \in Q_\exists$



(c) $\delta_2(q) = (q_1, q_2)$, $q \in Q_\forall$



(d) Sink state



(e) Tape cell

Fig. 4: Encoding of the δ-function (k ranges over all its possible values)

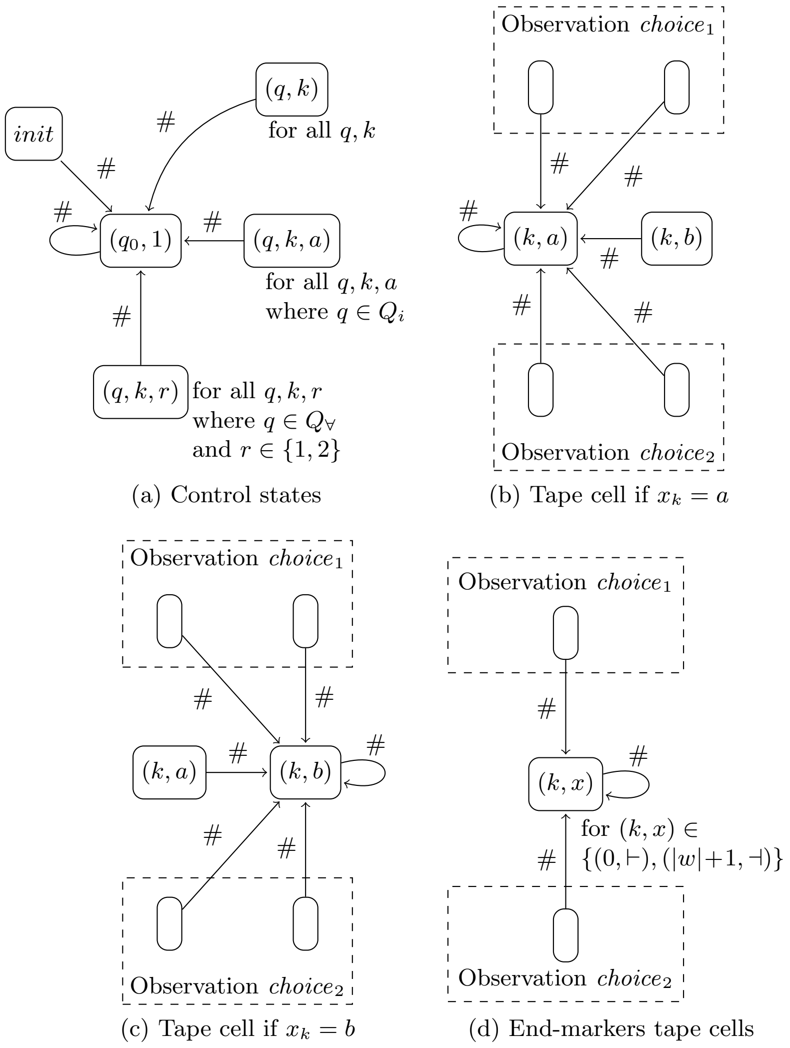
tape cell (Figure 4e) is in the state (k, a) or (k, b), respectively. Choosing a wrong test action means that the tape cell cannot perform the chosen action, implying that this may not be a synchronizing strategy. After the appropriate test action was chosen, we have to necessarily perform the update action $u^k_{x'}$ bringing us to the state (q', k') where the head is moved accordingly and this action has to synchronize with the k'th tape cell so that the new tape symbol x' is updated accordingly. Notice that any other tape cell, save the one on k'th position, simply mimics these two actions via self-loops in their current states.

- Let $q \in Q_\exists$. The corresponding transitions are depicted in Figure 4b. Here we can freely choose the successor control state by pick the action 1 or 2 according to the first and second successor given by the δ_2 function. Clearly, all tape cells will mimic the chosen action using a self-loop.
- Let $q \in Q_\forall$. The corresponding transitions are depicted in Figure 4c. After performing the guessing action g, the system nondeterministically decides whether to enter $(q, k, 1)$ or $(q, k, 2)$ and we have to investigate the possible continuation from both situations. However, as they are in different observation classes, we can split our strategy and design different continuations for these two possibilities. The main point is now about the tape cells in Figure 4e. They have also a nondeterministic choice about going to state with observation choice_1 or choice_2 but they do not have to follow the control state choice. However, if they do not follow it, we can observe such a behavior and design an alternative strategy for the tape cell, continuing the simulation like if the opposite choice was taken in the control states.
- Let $q = q_{acc}$. Then the transitions in Figure 4d apply and we can move using the action \$ into a global state called *sink* that is the only state that allows to synchronize both control states and tape cells. Clearly, any tape cell is able to perform \$ and enter the synchronizing state *sink* at any time, but only the accepting control state is able to enter the sink state.

This completes the proof and we have show that the constructed subset-to-subset synchronization problem has a synchronizing strategy if and only if the given ALBA accepts the input string, giving us the following hardness result. □

Theorem 7. *The synchronization and short-synchronization problems are EXPTIME-hard for NFA.*

Proof. In order to prove EXPTIME-hardness for the existence of synchronization strategy from any given initial state, we need to introduce additional transitions together with a new state *init* as depicted in Figure 5. These transitions add a new action # in such a way that any synchronizing strategy has to start by performing the action #. If any other action should be chosen instead of # then it is impossible to synchronize the state *init*. It is now clear that performing this initialization brings the system to the set of initial states in the subset-to-subset problem discussed above, deriving the following theorem. Note that for the short-synchronization case, we use Lemma 1 giving us an exponential upper-bound on the length of the shortest synchronizing strategy. □


$init$
(q,k)
for all q,k
$(q_0,1)$
(q,k,a)
for all q,k,a where $q \in Q_i$
(q,k,r)
for all q,k,r where $q \in Q_\forall$ and $r \in \{1,2\}$
#
Observation $choice_1$
Observation $choice_2$
(k,a)
(k,b)
(k,x)
for $(k,x) \in \{(0,\vdash),(|w|+1,\dashv)\}$

(a) Control states

(b) Tape cell if $x_k = a$

(c) Tape cell if $x_k = b$

(d) End-markers tape cells

Fig. 5: Initialization for the input $w = x_1x_2\dots x_n$ by adding a new state *init*

Theorem 8. *The synchronization, short-synchronization and subset-to-subset synchronization problems on DFA, PFA and NFA are polynomial-time reducible to the equivalent problems with only two observations.*

Proof. Let $T = (S, Act, \to, \mathcal{O}, \gamma)$ be a given finite LTSP and let $\ell = \lceil \log |\mathcal{O}| \rceil$. We can assume that all observations from $\mathcal{O}$ are written in binary and contain exactly ℓ bits (including the leading zeros). Then the i'th bit of an observation $o \in \mathcal{O}$ is denoted as o^i. We construct an LTSP $T' = (S', Act', \to', \mathcal{O}', \gamma')$ such that

- $S' = S \cup \{s_0, s_1, s_2, \ldots, s_{\ell-1} \mid s \in S\}$,
- $Act' = Act \cup \{\bullet, \#\}$,
- $\to' = \{(s, a, s_1) \mid s' \xrightarrow{a} s\} \cup \{(s_1, \bullet, s_2), (s_2, \bullet, s_3), \ldots, (s_{\ell-2}, \bullet, s_{\ell-1}), (s_{\ell-1}, \bullet, s) \mid s \in S\} \cup \{(s_1, \#, s_1), (s_2, \#, s_1), (s_3, \#, s_1), \ldots, (s_{\ell-1}, \#, s_1), (s, \#, s_1) \mid s \in S\} \cup \{(s_i, a, s_i) \mid s \in S,\ 1 \le i < \ell,\ a \in Act \smallsetminus \{\bullet, \#\}\} \cup \{(s, \bullet, s) \mid s \in S\} \cup \{(s_0, a, s_1) \mid s \in S,\ a \in Act\}$,
- $\mathcal{O}' = \{0, 1\}$, and
- $\gamma'(s) = \gamma(s)^\ell$ and $\gamma'(s_i) = \gamma(s)^i$ for all $s \in S$ and all i, $1 \le i < \ell$, and $\gamma'(s_0) = 0$ for all $s \in S$.

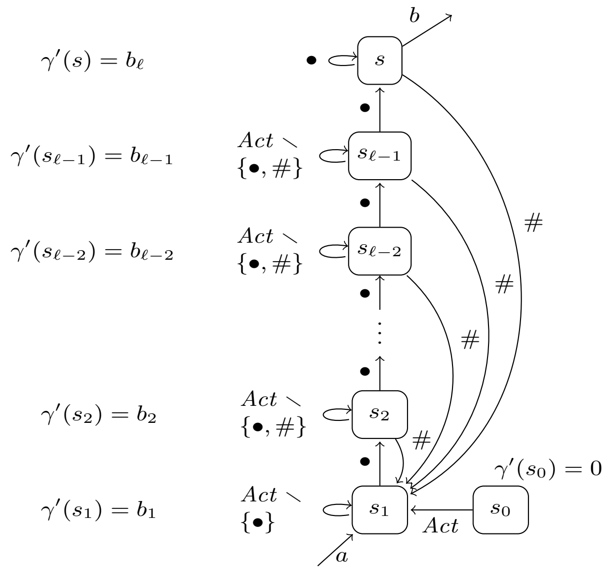


Fig. 6: New states for every $s \in S$ where $\gamma(s) = b_1 b_2 \ldots b_\ell$; the arrow labelled with a represents incoming transitions to s and the one labelled with b outgoing transitions from s.

The construction is depicted in Figure 6. The main idea is now that instead of entering the state s in the original system, we enter the newly added state

s_1 associated with s. Now by performing the sequence of actions $\bullet$, we obtain step-by-step the knowledge about the observation in the state s. The added self-loops are necessary only for the case of DFA in order to have a complete transition relation and the states s_0 (not reachable from any other states) are important only for arguing about the length of the synchronizing strategy. Now we can show that the synchronization, short-synchronization and subset-to-subset synchronization problems have a solution in T iff they have a solution in T'.

Assume a synchronizing strategy in the original LTSP T. The equivalent strategy in T' will simply initially perform the action # so that is gets the possibility to read the whole information about the observation in the given state. After this the actions in the strategy T are separated by ℓ-1 actions $\bullet$ in order to obtain a strategy for T'. Such a sequence provides a full information about the original observation in the state s, allowing us to follow faithfully the given synchronizing strategy from T also in T'.

On the other hand, the system T' does not provide any additional information by performing # compared with T; note that after # the system T' must perform the actions $\bullet$ until reaching the original state s as it is impossible to synchronize the system in the newly added states $s_0, \ldots, s_{\ell-1}$ and exercising the self-loops in the newly added states does not help either. Hence any synchronizing strategy in T' can be transformed into a synchronizing strategy in T by leaving out the intermediate actions $\bullet$ and #.

For the short-synchronization problem for T, checking if the length of the strategy is at most k, we will instead ask in T' whether there is a strategy of length at most $k \cdot (\ell - 1) + 1$. □