Synchronizing automata with LTL constraints

Nicolas Markey
IRISA, CNRS & Inria & Univ. Rennes, France

joint work with Nathalie Bertrand and Hugo Francon.
Coffee machine
Coffee machine

sugar? yes/no

always ends in the sugar state
Coffee machine

- sugar?
- more sugar?

[Green checkmark] Always ends in the sugar state.
Coffee machine

sugar?

more sugar?

milk?
Coffee machine

- sugar?
  - yes
  - no

- more sugar?
  - yes
  - no

- milk?
  - yes
  - no

"yes" always ends in the sugar state.
Coffee machine

Insert coin

“yes” always ends in the sugar state
Coffee machine

insert coin → sugar
  yes → more sugar
  no → milk
        yes/no → sugar
  yes/no → milk

always ends in the sugar state
Coffee machine

insert coin

sugar

yes

more sugar

no

yes/no

milk

yes/no

“no · yes · no · yes” always ends in the sugar state
Synchronizing words

Definition

A finite-state automaton $\mathcal{A} = (S, \Sigma, \delta)$ is synchronizable if there exists $w \in \Sigma^*$ such that $\delta(s, w) = \delta(s', w)$ for all $s, s' \in S$.

In (most of) the sequel, automata are deterministic and complete.
Synchronizing words

**Definition**
A finite-state automaton \( \mathcal{A} = (S, \Sigma, \delta) \) is **synchronizable** if there exists \( w \in \Sigma^* \) such that \( \delta(s, w) = \delta(s', w) \) for all \( s, s' \in S \).

In (most of) the sequel, automata are **deterministic** and **complete**.

**Proposition**

*Synchronizability can be decided in polynomial space.*
Synchronizing words

**Definition**
A finite-state automaton \( \mathcal{A} = (S, \Sigma, \delta) \) is **synchronizable** if there exists \( w \in \Sigma^* \) such that \( \delta(s, w) = \delta(s', w) \) for all \( s, s' \in S \).

In (most of) the sequel, automata are deterministic and complete.

**Proposition**
Synchronizability can be decided in **polynomial space**.

**Proof**
Consider the **power automaton** \( \mathcal{P}(\mathcal{A}) = (S', \Sigma, \delta') \):
- \( S' = 2^S \);
- \( \delta'(P, \sigma) = \{ \delta(s, \sigma) \mid s \in P \} \).

Non-deterministically look for a path from \( S \) to a singleton state.
Deciding the existence of a synchronizing word

**Theorem (Čer64)**

Synchronizability can be decided in non-deterministic log. space.

Deciding the existence of a synchronizing word

**Theorem (Čer64)**

Synchronizability can be decided in non-deterministic log. space.

**Proof**

**Lemma**

$A$ is synchronizable iff

for all $s_i, s_j$, there exists $w_{ij}$ such that $\delta(s_i, w_{ij}) = \delta(s_j, w_{ij})$.

Deciding the existence of a synchronizing word

**Theorem (Čer64)**

Synchronizability can be decided in *non-deterministic log. space*.

**Proof**

**Lemma**

A is synchronizable iff

\[
\text{for all } s_i, s_j, \text{ there exists } w_{ij} \text{ such that } \delta(s_i, w_{ij}) = \delta(s_j, w_{ij}).
\]

Deciding the existence of a synchronizing word

Theorem (Čer64)

Synchronizability can be decided in non-deterministic log. space.

Proof

Lemma

A is synchronizable iff

for all \( s_i, s_j \), there exists \( w_{ij} \) such that \( \delta(s_i, w_{ij}) = \delta(s_j, w_{ij}) \).

\[ \begin{array}{c}
S_0 \\
S_1 \\
S_2 \\
S_3 \\
\end{array} \rightarrow \begin{array}{c}
S_1 \\
S_2 \\
S_3 \\
\end{array} \]

\[ w_{01} \]

Deciding the existence of a synchronizing word

**Theorem (Čer64)**

Synchronizability can be decided in non-deterministic log. space.

**Proof**

**Lemma**

\( A \) is synchronizable iff

for all \( s_i, s_j \), there exists \( w_{ij} \) such that \( \delta(s_i, w_{ij}) = \delta(s_j, w_{ij}) \).

Deciding the existence of a synchronizing word

Theorem (Čer64)

Synchronizability can be decided in non-deterministic log. space.

Proof

Lemma

A is synchronizable iff

for all \( s_i, s_j \), there exists \( w_{ij} \) such that \( \delta(s_i, w_{ij}) = \delta(s_j, w_{ij}) \).

Deciding the existence of a synchronizing word

**Theorem (Čer64)**

Synchronizability can be decided in non-deterministic log. space.

**Proof**

**Lemma**

A is synchronizable iff

for all \( s_i, s_j \), there exists \( w_{ij} \) such that \( \delta(s_i, w_{ij}) = \delta(s_j, w_{ij}) \).

This gives a synchronizing word (if any) of size \( O(n^3) \).

**Černý's Conjecture**

The shortest synchronizing word (if any) has size at most \((n - 1)^2\).

Synchronizing words in different contexts

- Subset-synchronization;

**Theorem (Rys83, Mar10)**

Subset-synchronizability in complete DFAs is \textit{PSPACE-complete}.
Synchronizability in NFAs is \textit{PSPACE-complete}.
Careful-synchronizability in incomplete DFAs is \textit{PSPACE-complete}.

Synchronizing words in different contexts

- Subset-synchronization;
- Synchronization in MDPs and probabilistic automata;

Theorem (Shi14)

Almost-sure synchronizability in MDPs is PSPACE-complete. Almost-sure synchronizability in PAs is undecidable.

Synchronizing words in different contexts

- Subset-synchronization;
- Synchronization in MDPs and probabilistic automata;
- Synchronization in timed automata;

**Theorem (Shi14)**

*Synchronizability in timed automata is **PSPACE-complete**.*

---


Synchronizing words in different contexts

- Subset-synchronization;
- Synchronization in MDPs and probabilistic automata;
- Synchronization in timed automata;
- Synchronization with energy constraints;

**Theorem (Shi14)**

*Synchronizability under energy constraints is PSPACE-complete.*

---


Synchronizing words in different contexts

- Subset-synchronization;
- Synchronization in MDPs and probabilistic automata;
- Synchronization in timed automata;
- Synchronization with energy constraints;
- Synchronization with dynamic constraints;
- Synchronization with conditions on the order of states (e.g. last occurrence of $s$ must precede first occurrence of $s'$).

**Theorem (Wol20)**

Synchronizability with constraint \( \text{last}(s) < \text{last}(s') \) is in NP.
Synchronizability with constraint \( \text{last}(s) < \text{first}(s') \) is PSPACE-complete.

Linear-time Temporal Logic

- atomic propositions: ●, ●, ...
- Boolean combinators: ¬φ, φ ∨ ψ, φ ∧ ψ, ...
- temporal modalities:
  - Xφ
  - φ U ψ
  - true U φ ≡ F φ
  - ¬F ¬φ ≡ G φ

“next φ”
“φ until ψ”
“eventually φ”
“always φ”

Linear-time Temporal Logic

- atomic propositions: ○, ○, ...
- Boolean combinators: \( \neg \varphi, \varphi \lor \psi, \varphi \land \psi \), ...
- temporal modalities:
  - \( X \varphi \) → next \( \varphi \)
  - \( \varphi U \psi \) → \( \varphi \) until \( \psi \)
  - true \( U \varphi \equiv F \varphi \) → eventually \( \varphi \)
  - \( \neg F \neg \varphi \equiv G \varphi \) → always \( \varphi \)

Example

\text{last}(s) < \text{first}(s') \) can be expressed (roughly) as

\[
(\neg s') U (s \land X G \neg s)
\]

Synchronizing words with LTL constraints

**Theorem (VW86,DV13)**

For any LTL formula $\varphi$, there exists an exponential-size NFA $F_\varphi$ (with a single accepting state $q_f$) accepting exactly the finite traces satisfying $\varphi$.

--


Synchronizing words with LTL constraints

**Theorem (VW86,DV13)**

For any LTL formula $\varphi$, there exists an exponential-size NFA $F_\varphi$ (with a single accepting state $q_f$) accepting exactly the finite traces satisfying $\varphi$.

**Theorem**

Synchronizability with LTL constraints is $\text{PSPACE}$-complete.

---


Synchronizing words with LTL constraints

**Theorem (VW86,DV13)**

For any LTL formula \( \varphi \), there exists an exponential-size NFA \( F_\varphi \) (with a single accepting state \( q_f \)) accepting exactly the finite traces satisfying \( \varphi \).

**Theorem**

Synchronizability with LTL constraints is \textit{PSPACE-complete}.

**Proof**

- (Non-deterministically) look for a path in the powerset automaton \( \mathcal{P}(\mathcal{A} \otimes F_\varphi) \) to a singleton-state \( \{s, q_f\} \).
- Hardness follows from [Wol20].

---


Conclusion and future work

Conclusion

- synchronizability with LTL constraints
- **PSPACE-complete** in the general case;
- remains **PSPACE-complete** for
  - subset-synchronizability;
  - fixed LTL formulas;
  - small fragments of LTL (such as \( L^+(G) \)).

Possible directions for future work

- suitable restrictions of LTL with lower complexity;
  (little hope...)
- synchronization with branching-time logics \([CD16,Wol20]\)


Conclusion and future work

**Conclusion**
- synchronizability with LTL constraints
- **PSPACE-complete** in the general case;
- remains **PSPACE-complete** for
  - subset-synchronizability;
  - fixed LTL formulas;
  - small fragments of LTL (such as $L^+(G)$).

**Possible directions for future work**
- suitable restrictions of LTL with lower complexity;
  - (little hope...)
- synchronization with branching-time logics [CD16,Wol20]