Constrained Synchronization and Subset Synchronization Problems for Weakly Acyclic Automata

Stefan Hoffmann[1]

[1]University of Trier

International Conference on Developments in Language Theory (DLT) 2021,
August 16 – August 20, 2021
Porto, Portugal

Notation

Let Σ be some finite alphabet.

- Languages: subsets of Σ^* (the free monoid)
- Automata $A = (\Sigma, Q, \delta, s_0, F)$ with input alphabet Σ, state set Q, transition function $\delta : Q \times \Sigma \to Q$, start state s_0 and final states F.
- Semi-Automata: Like automata, but without a start state or final states.
- Regular language: Languages described by finite automata as labels of paths from the start to some final state.

Synchronizing Automata

- $A = (\Sigma, Q, \delta)$, $\delta : Q \times \Sigma \to Q$: DCSA, i.e., deterministic complete semi-automaton.
- $w \in \Sigma^*$ synchronizing if $|\delta(Q, w)| = 1$.
- A synchronizing if A admits a synchronizing word.

Synchronizing Automata

- $A = (\Sigma, Q, \delta)$, $\delta : Q \times \Sigma \to Q$: DCSA, i.e., deterministic complete semi-automaton.
- $w \in \Sigma^*$ synchronizing if $|\delta(Q, w)| = 1$.
- A synchronizing if A admits a synchronizing word.
  For DCSA, this property can be checked in P, BUT:
- SYNC: Given DCSA A and $k \geq 0$, check if A has a synchronizing word of length at most k is NP-complete. (Rystsov 1980, Eppstein 1990)

Synchronizing Automata

- $A = (\Sigma, Q, \delta)$, $\delta : Q \times \Sigma \to Q$: DCSA, i.e., deterministic complete semi-automaton.
- $w \in \Sigma^*$ synchronizing if $|\delta(Q, w)| = 1$.
- A synchronizing if A admits a synchronizing word. For DCSA, this property can be checked in P, BUT:
- Sync: Given DCSA A and $k \geq 0$, check if A has a synchronizing word of length at most k is NP-complete. (Rystsov 1980, Eppstein 1990)
- Famous combinatorial **conjecture** attributed to Černý: Each synchronizing DCSA has a synchronizing word of length at most $(|Q| - 1)^2$. **Open question** for > 50 years.
- Recent improvements on the leading coefficients of a cubic upper bound: Szykuła STACS 2018, Shitov JALC 2019.

Constrained Synchronization

- Motivation for synchronization problems: Bring system into **defined state without** a priori **knowledge about the current state**.

Constrained Synchronization

- Motivation for synchronization problems: Bring system into **defined state without** a priori **knowledge about the current state**.
- In practical: Arbitrary (uncontrolled) reset sequence undesirable.
- Bring system in reset-mode first, reset it, and bring it back to operation-mode.

Constrained Synchronization

- Motivation for synchronization problems: Bring system into **defined state without** a priori **knowledge about the current state**.
- In practical: Arbitrary (uncontrolled) reset sequence undesirable.
- Bring system in reset-mode first, reset it, and bring it back to operation-mode.
- Command has (simplified) structure ab^*a.
- Find synchronizing word which is contained in ab^*a (NP-complete).

Constrained Synchronization

- Motivation for synchronization problems: Bring system into **defined state without** a priori **knowledge about the current state**.
- In practical: Arbitrary (uncontrolled) reset sequence undesirable.
- Bring system in reset-mode first, reset it, and bring it back to operation-mode.
- Command has (simplified) structure ab^*a.
- Find synchronizing word which is contained in ab^*a (NP-complete).

Fix a partial deterministic finite automaton (PDFA) $\mathcal{B} = (\Sigma, P, \mu, p_0, F)$.

Definition

$L(\mathcal{B})$-Constr-Sync
Input: DCSA $A = (\Sigma, Q, \delta)$.
Question: Is there a synchronizing word w for A with $w \in L(\mathcal{B})$?

History of L-Constr-Sync

- $L(B)$-Constr-Sync introduced in (Fernau et al., MFCS 2019)
- There, a completey classification for small constraint automata with $|Q| \leq 2$ and $|\Sigma| \leq 3$ was given. The problem is, depending on the constraint language, either PSPACE-complete or in P.
- For polycyclic constraint automata, the problem is always in NP (Hoffmann, ICTCS 2020).
- For commutative regular constraint languages, a trichotomy result was achieved (Hoffmann, COCOON 2020); showing that only NP-complete, PSPACE-complete or problems in P arise.
- Complete classifications for $|Q| \leq 3$ obtained (COCOON, 2021). Only problems which are NP-complete, PSPACE-complete, or in P occur.
- For letter-bounded constraint languages, a dichotomy between P and NP-completeness was shown (FCT, 2021).

Weakly Acyclic Automata

Weakly Acyclic Automata (Ryzhikov 2019)

DCSA $A = (\Sigma, Q, \delta)$ is called weakly acyclic, if there exists an ordering $q_1, q_2, \ldots, q_n$ of its states such that if $\delta(q_i, x) = q_j$ for $x \in \Sigma$, then $i \leq j$.

- A DCSA A is weakly acyclic if and only if the only loops are self-loops.
- These automata are also known as acyclic (Jiŕasková & Masopust, 2012) or partially ordered (Brzozowski & Fich, 1980).

This work

Here, we look at $L(\mathcal{B})$-CONSTR-SYNC for weakly acyclic input automata. Formally:

$L(\mathcal{B})$-WAA-CONSTR-SYNC
Input: *Weakly Acyclic Semi-Automaton* $\mathcal{A} = (\Sigma, Q, \delta)$.
Question: *Is there a synchronizing word* w *for* $\mathcal{A}$ *with* $w \in L(\mathcal{B})$?

Overview of Complexity for Different Types of Input Automata

Input Aut. Type	Complexity Class	Hardness	Reference
General Automata	PSPACE	PSPACE-hard	Fernau et al. (2019)
With Sink State	PSPACE	PSPACE-hard	Fernau et al. (2019)
Weakly Acyclic	NP	NP-hard	present work
TTSPL	NP	NP-hard	present work
Simple Idempotents[1]	P for $\|\Sigma\| = 2$ and Constr. Aut ≤ 3 states	-	unpublished
Commutative[2]	P	-	unpublished

The stated hardness results are obtained with the constraint $a(b+c)^*$ in the first four cases.

[1] $\mathcal{A} = (\Sigma, Q, \delta)$ has the property that for every $a \in \Sigma$ either $\delta(Q, a) = Q$ or $|\delta(Q, a)| = |Q| - 1$ and $\delta(q, aa) = \delta(q, a)$ for all $q \in Q$.

[2] $\mathcal{A} = (\Sigma, Q, \delta)$ has the property that for every $a, b \in \Sigma$ and $q \in Q$ we have $\delta(q, ab) = \delta(q, ba)$.

Constrained Synchronization for Weakly Acyclic Automata

Proposition

Let A be a weakly acyclic automaton with n states and $\mathcal{B} = (\Sigma, P, \mu, p_0, F)$ be a fixed PDFA. Then, a shortest synchronizing word $w \in L(\mathcal{B})$ for A has length at most $|P|\binom{n}{2}$.

Theorem

For any PDFA $\mathcal{B}$, we have $L(\mathcal{B})$-WAA-CONSTR-SYNC $\in NP$.

Questions

What precise complexities inside of NP are realizable? Are there constraint automata giving NP-complete problems?

Classification for Small Constraint Automata

Proposition

For the following constraint languages, the constrained synchronization problem for weakly acyclic automata is NP-hard:

$$
\begin{array}{lll}
a(b+c)^* & (a+b+c)(a+b)^* & (a+b)(a+c)^* \\
(a+b)^*c(a+b)^* & a^*b(a+c)^* & a^*(b+c)(a+b)^* \\
a^*b(b+c)^* & (a+b)^*c(b+c)^* & a^*(b+c)(b+c)^*.
\end{array}
$$

Classification for Small Constraint Automata

Proposition

For the following constraint languages, the constrained synchronization problem for weakly acyclic automata is NP-hard:

$$\begin{array}{lll} a(b+c)^* & (a+b+c)(a+b)^* & (a+b)(a+c)^* \\ (a+b)^*c(a+b)^* & a^*b(a+c)^* & a^*(b+c)(a+b)^* \\ a^*b(b+c)^* & (a+b)^*c(b+c)^* & a^*(b+c)(b+c)^*. \end{array}$$

The general problem $L(\mathcal{B})$-Constr-Sync, for the above constraint languages, is PSPACE-complete. However, for the constraint languages

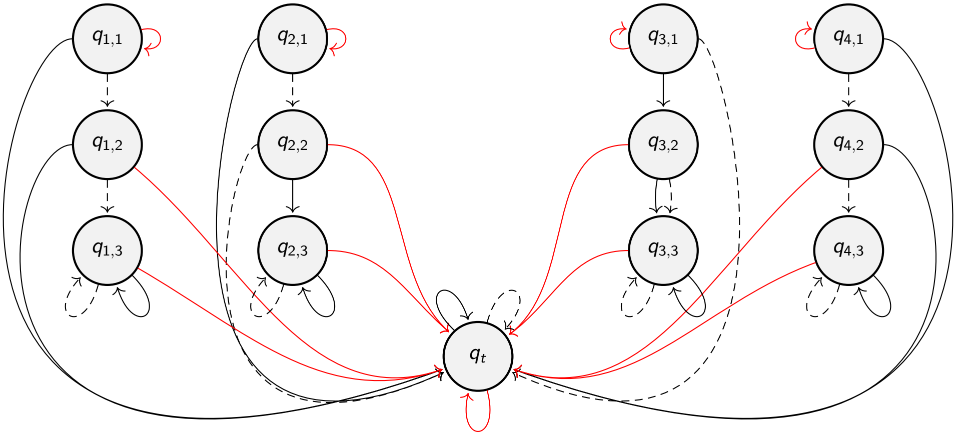
$$((a+b)^*c) \qquad ((a+b)^*ca^*)$$

the general problem is PSPACE-complete, but $L(\mathcal{B})$-WAA-Constr-Sync $\in$ P.

The Reduction (A Modification of the Rystsov-Eppstein Construction)

Example for $c(a+b)^*$ from a SAT instance with two variables, letter c red.

$(x_1 \vee x_2) \quad \wedge \quad (x_1 \vee \neg x_2) \quad \wedge \quad (\neg x_1) \quad \wedge \quad (x_1 \vee x_2)$



Classification for Small Constraint Automata

Classification

For constraint PDFAs $\mathcal{B}$ with at most two states over an at most ternary alphabet, $L(B)$-WAA-Constr-Sync is NP-complete precisely in the cases:

$$\begin{array}{lll} a(b+c)^* & (a+b+c)(a+b)^* & (a+b)(a+c)^* \\ (a+b)^*c(a+b)^* & a^*b(a+c)^* & a^*(b+c)(a+b)^* \\ a^*b(b+c)^* & (a+b)^*c(b+c)^* & a^*(b+c)(b+c)^* \end{array}$$

and polynomial time solvable otherwise.

Subset Synchronization Problems

Definition

SYNC-FROM-SUBSET
Input: $A = (\Sigma, Q, \delta)$ and $S \subseteq Q$.
Question: Is there a word w with $|\delta(S, w)| = 1$?

Definition

SYNC-INTO-SUBSET
Input: $A = (\Sigma, Q, \delta)$ and $S \subseteq Q$.
Question: Is there a word w with $\delta(Q, w) \subseteq S$?

Definition

SETTRANSPORTER
Input: $\mathcal{A} = (\Sigma, Q, \delta)$ and two subsets $S, T \subseteq Q$.
Question: Is there a word $w \in \Sigma^*$ such that $\delta(S, w) \subseteq T$?

Subset Synchronization Problems

- Sync-From-Subset is NP-complete for at least binary fixed Σ, and in P for unary alphabets (Ryzhikov 2019).
- Sync-Into-Subset is in P for any fixed alphabet Σ.
- SetTransporter is NP-complete for at least binary fixed Σ, and in P for unary alphabets.

Two-Terminal Series-Parallel Automata Graphs with Loops

Definition (Fernau & Bruchertseifer, 2019)

A directed (multi-)graph G is two-terminal series-parallel with loops, or TTSPL for short, with terminals s (the source) and t (the sink), if it can be produced by a sequence of the following operations:

Two-Terminal Series-Parallel Automata Graphs with Loops

Definition (Fernau & Bruchertseifer, 2019)

A directed (multi-)graph G is two-terminal series-parallel with loops, or TTSPL for short, with terminals s (the source) and t (the sink), if it can be produced by a sequence of the following operations:

1. Create a new graph, with two vertices s and t and a single arc directed from the source s to the sink t.

Two-Terminal Series-Parallel Automata Graphs with Loops

Definition (Fernau & Bruchertseifer, 2019)

A directed (multi-)graph G is two-terminal series-parallel with loops, or TTSPL for short, with terminals s (the source) and t (the sink), if it can be produced by a sequence of the following operations:

1. Create a new graph, with two vertices s and t and a single arc directed from the source s to the sink t.
2. Given two TTSPL X and Y, with sources s_X and s_Y, respectively, and sinks t_X and t_Y, respectively, form a new graph $G = P(X, Y)$ by identifying $s = s_X = s_Y$ and $t = t_X = t_Y$. This is known as the parallel composition of X and Y.

Two-Terminal Series-Parallel Automata Graphs with Loops

Definition (Fernau & Bruchertseifer, 2019)

A directed (multi-)graph G is two-terminal series-parallel with loops, or TTSPL for short, with terminals s (the source) and t (the sink), if it can be produced by a sequence of the following operations:

1. Create a new graph, with two vertices s and t and a single arc directed from the source s to the sink t.
2. Given two TTSPL X and Y, with sources s_X and s_Y, respectively, and sinks t_X and t_Y, respectively, form a new graph $G = P(X, Y)$ by identifying $s = s_X = s_Y$ and $t = t_X = t_Y$. This is known as the parallel composition of X and Y.
3. Given two TTSPL X and Y, with sources s_X and s_Y, respectively, and sinks t_X and t_Y, respectively, form a new graph $G = S(X, Y)$ by identifying $s = s_X$, $t_X = s_Y$ and $t = t_Y$. This is known as the series composition of X and Y.

Two-Terminal Series-Parallel Automata Graphs with Loops

Definition (Fernau & Bruchertseifer, 2019)

A directed (multi-)graph G is two-terminal series-parallel with loops, or TTSPL for short, with terminals s (the source) and t (the sink), if it can be produced by a sequence of the following operations:

1. Create a new graph, with two vertices s and t and a single arc directed from the source s to the sink t.
2. Given two TTSPL X and Y, with sources s_X and s_Y, respectively, and sinks t_X and t_Y, respectively, form a new graph $G = P(X, Y)$ by identifying $s = s_X = s_Y$ and $t = t_X = t_Y$. This is known as the parallel composition of X and Y.
3. Given two TTSPL X and Y, with sources s_X and s_Y, respectively, and sinks t_X and t_Y, respectively, form a new graph $G = S(X, Y)$ by identifying $s = s_X$, $t_X = s_Y$ and $t = t_Y$. This is known as the series composition of X and Y.
4. Add a loop to a terminal node of a given TTSPL graph.

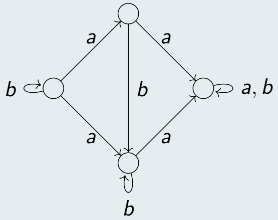
TTSPL Automata

TTSPL Automata

Automata whose automaton graph is a TTSPL graph, i.e., a series-parallel graph with self-loops and two terminals.

Observation

TTSPL automata are a proper subclass of the weakly acyclic automata. For example, the following weakly acyclic automaton is not a TTSPL automaton.

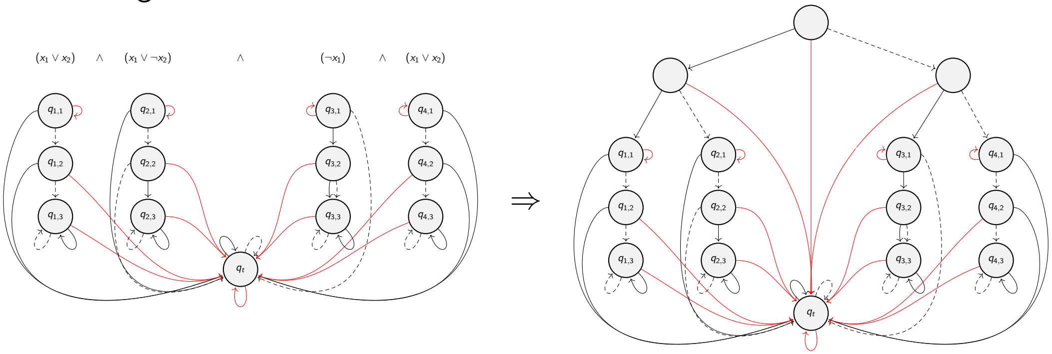


Constrained Synchronization for TTSPL input automata

Natural Question

How do our results relate to this subclass?

Transforming WAAs to TTSPLs:



Constrained Synchronization for TTSPL input automata

Natural Question

How do our results relate to this subclass?

TTSPL input automata

So, we find the same complexity classification as before when the problem is restricted to TTSPL automata.