Synchronization and Diversity of Solutions*

Emmanuel Arrighi[1,2,4], Henning Fernau[2],
Mateus de Oliveira Oliveira[1,3], Petra Wolf[1,5]

[1]University of Bergen
[2]University of Trier
[3]University of Stockholm
[4]University of Lyon
[5]University of Bordeaux

* originally presented at AAAI 2023

Kaiserslautern, Theorietag, October 2023

Table of Contents

Synchronizing Words

- A word w is said to be synchronizing for a DFA A if there is some state q of A such that any state q' is sent to q by w.

Synchronizing Words

- A word w is said to be synchronizing for a DFA A if there is some state q of A such that any state q' is sent to q by w.
- The most elementary problem is to determine whether a given DFA has a synchronizing word. This can be decided in poly-time.

Synchronizing Words

- A word w is said to be synchronizing for a DFA A if there is some state q of A such that any state q' is sent to q by w.
- The most elementary problem is to determine whether a given DFA has a synchronizing word. This can be decided in poly-time.
- Nevertheless, in several applications, one is interested in finding a synchronizing word satisfying certain additional constraints.
- Here, the complexity landscape changes drastically:
  even determining the existence of a synchronizing word satisfying additional regularity constraints is NP-hard.

Diversity of Solutions: Motivation

- *Diversity of solutions:* find a *small* set of solutions that are sufficiently diverse from one another.

Diversity of Solutions: Motivation

- *Diversity of solutions:* find a *small* set of solutions that are sufficiently diverse from one another.
- Motivation:
  - Enumeration of all (minimal) solutions often too costly.
  - Even if analysts might want to wait long to see all solutions, they might not want to wait too long between seeing any two subsequent solutions (polynomial delay).
  - Possibly better approach: Show only few ‘typical’ solutions that show the whole solution space exemplarily.
  - Seminal paper: Baste *et al.*: Diversity of solutions: An exploration through the lens of fixed-parameter tractability theory. *Artificial Intelligence* 303, 2022.

Diversity of Solutions: Motivation

- *Diversity of solutions:* find a *small* set of solutions that are sufficiently diverse from one another.
- Motivation:
  - Enumeration of all (minimal) solutions often too costly.
  - Even if analysts might want to wait long to see all solutions, they might not want to wait too long between seeing any two subsequent solutions (polynomial delay).
  - Possibly better approach: Show only few ‘typical’ solutions that show the whole solution space exemplarily.
  - Seminal paper: Baste *et al.*: Diversity of solutions: An exploration through the lens of fixed-parameter tractability theory. *Artificial Intelligence* 303, 2022.
- **How to adapt the framework of solution diversity to the context of synchronization?**

Diversity of Solutions: Synchronizing Words

- How to adapt the framework of solution diversity to the context of synchronization?
- Problem: usual notions of diversity of solutions based on Hamming distance are not appropriate to measure diversity between strings.

Diversity of Solutions: Synchronizing Words

- How to adapt the framework of solution diversity to the context of synchronization?
- Problem: usual notions of diversity of solutions based on Hamming distance are not appropriate to measure diversity between strings.
  - distinct solutions may have distinct length. Even strings of the same length that are very similar to each other may have very large Hamming distance: $w = abab...ab$ and $w' = baba...ba$.

Diversity of Solutions: Synchronizing Words

- How to adapt the framework of solution diversity to the context of synchronization?
- Problem: usual notions of diversity of solutions based on Hamming distance are not appropriate to measure diversity between strings.
  - distinct solutions may have distinct length. Even strings of the same length that are very similar to each other may have very large Hamming distance: $w = abab...ab$ and $w' = baba...ba$.
- Solution: base our diversity measure in the notion of *edit distance*.

Diversity of Solutions: Synchronizing Words

- How to adapt the framework of solution diversity to the context of synchronization?
- Problem: usual notions of diversity of solutions based on Hamming distance are not appropriate to measure diversity between strings.
  - distinct solutions may have distinct length. Even strings of the same length that are very similar to each other may have very large Hamming distance: $w = abab...ab$ and $w' = baba...ba$.
- Solution: base our diversity measure in the notion of *edit distance*.
- Problem: Set of solutions S in which any two of them are far apart from each other may still not capture solution diversity in our context.
  - If w is a synchronizing word, then any xwy is synchronizing.

Diversity of Solutions: Synchronizing Words

- How to adapt the framework of solution diversity to the context of synchronization?
- Problem: usual notions of diversity of solutions based on Hamming distance are not appropriate to measure diversity between strings.
  - distinct solutions may have distinct length. Even strings of the same length that are very similar to each other may have very large Hamming distance: $w = abab...ab$ and $w' = baba...ba$.
- Solution: base our diversity measure in the notion of *edit distance*.
- Problem: Set of solutions S in which any two of them are far apart from each other may still not capture solution diversity in our context.
  - If w is a synchronizing word, then any xwy is synchronizing.
- Solution: Let each word in S be subsequence-minimal synchronizing.

Diversity of Solutions: Motivation of Our Work

- Motivation: Why all this?
- Applications to conformant planning, multi-agent systems, robotics, secret sharing, etc.

Diversity of Solutions: Motivation of Our Work

- Motivation: Why all this?
- Applications to conformant planning, multi-agent systems, robotics, secret sharing, etc.
- Synchronizing words may be viewed as a way of resetting agents whose behavior is modeled or governed by a DFA.
  Diversity is a way of achieving resilience.

Diversity of Solutions: Motivation of Our Work

- Motivation: Why all this?
- Applications to conformant planning, multi-agent systems, robotics, secret sharing, etc.
- Synchronizing words may be viewed as a way of resetting agents whose behavior is modeled or governed by a DFA.
  Diversity is a way of achieving resilience.
- Enumeration is infeasible (poly-delay unlikely):
- The extension problem of synchronizing words wrt. subsequence ordering is NP-hard. (F. & Hoffmann, JALC 2019).
- Interesting technicality: subword (infix) vs. subsequence

Edit Distance

- Let $u = u[1]u[2]\ldots u[n] \in \Sigma^n$.
  We use the following elementary edit operations on u (steps):

Edit Distance

- Let $u = u[1]u[2]\dots u[n] \in \Sigma^n$.
  We use the following elementary edit operations on u (steps):
  1. Insertion: for $i \in \{0, \dots, n\}$, insert letter $a \in \Sigma$ after position i, yielding $u[1..i]\, a\, u[i+1..n]$.
  2. Deletion: for $i \in \{1, \dots, n\}$, delete the entry $u[i]$ from u, resulting in $u[1..i-1]\, u[i+1..n]$.
  3. Replacement: for $i \in \{1, \dots, n\}$, replace $u[i]$ with some $a \in \Sigma \setminus \{u[i]\}$, getting $u[1..i-1]\, a\, u[i+1..n]$.

Edit Distance

- Let $u = u[1]u[2]\ldots u[n] \in \Sigma^n$.
  We use the following elementary edit operations on u (steps):
  1. Insertion: for $i \in \{0, \ldots, n\}$, insert letter $a \in \Sigma$ after position i, yielding $u[1..i]\, a\, u[i+1..n]$.
  2. Deletion: for $i \in \{1, \ldots, n\}$, delete the entry $u[i]$ from u, resulting in $u[1..i-1]\, u[i+1..n]$.
  3. Replacement: for $i \in \{1, \ldots, n\}$, replace $u[i]$ with some $a \in \Sigma \setminus \{u[i]\}$, getting $u[1..i-1]\, a\, u[i+1..n]$.
- The edit distance between u and w, denoted by $\Delta(u, w)$, is defined as the minimum s such that u can be edited to w in s steps.

Subsequence Minimal Synchronizing Words

- A word w is a subsequence-minimal synchronizing word for a DFA A if
  - w is synchronizing for A and
  - no proper subsequence w' of w is synchronizing for A.

Subsequence Minimal Synchronizing Words

- A word w is a subsequence-minimal synchronizing word for a DFA A if
  - w is synchronizing for A and
  - no proper subsequence w' of w is synchronizing for A.
- The set of subsequence-minimal synchronizing words is finite.

Subsequence Minimal Synchronizing Words

- A word w is a subsequence-minimal synchronizing word for a DFA A if
  - w is synchronizing for A and
  - no proper subsequence w' of w is synchronizing for A.
- The set of subsequence-minimal synchronizing words is finite.
- Subsequence minimality imposes a **qualitative level of dissimilarity** between two words in a prospective subset W of solutions.
  $\rightsquigarrow$ increased representativeness of W.

Synchronization Diversity

Definition 1 (Synchronization Diversity)

Let $A = (Q, \Sigma, \delta)$ be a DFA. We say that a set of words $\{w_1, \ldots, w_r\}$ has synchronization diversity k if the following holds.

Synchronization Diversity

Definition 1 (Synchronization Diversity)

Let $A = (Q, \Sigma, \delta)$ be a DFA. We say that a set of words $\{w_1, \ldots, w_r\}$ has synchronization diversity k if the following holds.

1. For each $i, j \in \{1, ..., r\}$ with $i \neq j$, $\Delta(w_i, w_j) \geq k$.
2. For each $i \in \{1, ..., r\}$, w_i is a subsequence-minimal synchronizing word for A.

Synchronization Diversity

Definition 1 (Synchronization Diversity)

Let $A = (Q, \Sigma, \delta)$ be a DFA. We say that a set of words $\{w_1, \ldots, w_r\}$ has synchronization diversity k if the following holds.

1. For each $i, j \in \{1, ..., r\}$ with $i \neq j$, $\Delta(w_i, w_j) \geq k$.
2. For each $i \in \{1, ..., r\}$, w_i is a subsequence-minimal synchronizing word for A.

Item 2 adds a qualitative component to the pure quantitative first item.

Table of Contents

Useful Facts

Lemma 2 (Higman's Lemma)

Let Σ be an alphabet and $L \subseteq \Sigma^*$.
There is a unique finite set $S \subseteq L$ satisfying the following properties.

1. *For each word $w \in L$, some word $u \in S$ is a subsequence of w.*
2. *Each word $u \in S$ is subsequence-minimal for L.*

Useful Facts

Lemma 2 (Higman's Lemma)

Let Σ be an alphabet and $L \subseteq \Sigma^*$.
There is a unique finite set $S \subseteq L$ satisfying the following properties.

1. *For each word $w \in L$, some word $u \in S$ is a subsequence of w.*
2. *Each word $u \in S$ is subsequence-minimal for L.*

Lemma 3 (Subsequence Automaton)

Let $A = (Q, \Sigma, \delta)$ be an NFA. One can build in time $O(|A|)$ a $|Q|$-state NFA $\mathrm{Subseq}(A)$ with $L(\mathrm{Subseq}(A)) = \{u \ : \ \exists w \in L(A), \ u \leq w\}$.

Useful Facts

Lemma 2 (Higman's Lemma)

Let Σ be an alphabet and $L \subseteq \Sigma^*$.
There is a unique finite set $S \subseteq L$ satisfying the following properties.

1. *For each word $w \in L$, some word $u \in S$ is a subsequence of w.*
2. *Each word $u \in S$ is subsequence-minimal for L.*

Lemma 3 (Subsequence Automaton)

Let $A = (Q, \Sigma, \delta)$ be an NFA. One can build in time $O(|A|)$ a $|Q|$-state NFA $\mathrm{Subseq}(A)$ with $L(\mathrm{Subseq}(A)) = \{u \ : \ \exists w \in L(A), \ u \leq w\}$.

Lemma 4 (Subsequence-Minimal Words)

Let $A = (Q, \Sigma, \delta)$ be a DFA. One can construct in time $O(2^{|Q|} \cdot |A|)$ a DFA $\mathrm{MinSubseq}(A)$ with $|Q| \cdot 2^{|Q|}$ states s.t.
$L(\mathrm{MinSubseq}(A)) = \{u \ : \ u \in L(A) \wedge \forall w \in L(A), w \not< u\}$.

Synchronizing Words

Lemma 5 (Synchronizing Words)

Let $A = (Q, \Sigma, \delta)$ *be a DFA. One can construct in time* $O(2^{|Q|} \cdot |A|)$ *a DFA* $\mathrm{Sync}(A)$ *with at most* $2^{|Q|}$ *states such that*

$$L(\mathrm{Sync}(A)) = \{u \ : \ u \text{ is synchronizing for } A\}.$$

Corollary 6

Let $A = (Q, \Sigma, \delta)$ *be a DFA. One can construct in time* $O(2^{2^{|Q|}} \cdot 2^{|A|})$ *a DFA* $\mathrm{SyncMinSubseq}(A)$ *with at most* $2^{2^{|Q|}+|Q|}$ *many states accepting*

$$L(\mathrm{SyncMinSubseq}(A)) = \{u \in \Sigma^* : u \text{ is a subsequence-minimal synchronizing word for } A\}.$$

Open: Optimality of this construction?

Convolution: Pairwise Concatenation with Blanks

Let $u, v \in \Sigma^*$, $\square \notin \Sigma$.

$$u \otimes v = \begin{cases} \varepsilon, & \text{if } u = v = \varepsilon \\ (u[1], \square) \cdot (\mathrm{tail}(u) \otimes \varepsilon), & \text{if } u \neq \varepsilon, v = \varepsilon \\ (\square, v[1]) \cdot (\varepsilon \otimes \mathrm{tail}(v)), & \text{if } u = \varepsilon, v \neq \varepsilon \\ (u[1], v[1]) \cdot (\mathrm{tail}(u) \otimes \mathrm{tail}(v)), & \text{if } u \neq \varepsilon, v \neq \varepsilon \end{cases}$$

For instance, the convolution of word $u = ababa$ with $v = abb$ is the word

$$u \otimes v = (a, a)(b, b)(a, b)(b, \square)(a, \square).$$

Convolution: Pairwise Concatenation with Blanks

Let $u, v \in \Sigma^*$, $\square \notin \Sigma$.

$$u \otimes v = \begin{cases} \varepsilon, & \text{if } u = v = \varepsilon \\ (u[1], \square) \cdot (\mathrm{tail}(u) \otimes \varepsilon), & \text{if } u \neq \varepsilon, v = \varepsilon \\ (\square, v[1]) \cdot (\varepsilon \otimes \mathrm{tail}(v)), & \text{if } u = \varepsilon, v \neq \varepsilon \\ (u[1], v[1]) \cdot (\mathrm{tail}(u) \otimes \mathrm{tail}(v)), & \text{if } u \neq \varepsilon, v \neq \varepsilon \end{cases}$$

For instance, the convolution of word $u = ababa$ with $v = abb$ is the word

$$u \otimes v = (a, a)(b, b)(a, b)(b, \square)(a, \square).$$

Lemma 7

The language $L_k^{\square}$ of all words over the alphabet $(\Sigma \cup \{\square\})^{\times k}$ that can be obtained as the convolution of k words over Σ can be accepted by a DFA with 2^k many states.
There is no DFA with less than 2^k many states that accepts $L_k^{\square}$.

Edit Distance vs Diversity of Solutions

Lemma 8

Let Σ *be an alphabet with* $|\Sigma| \geq 2$ *and* $k \in \mathbb{N}_{>0}$. *There is an NFA* $\mathrm{Edit}^{<}(\Sigma, k)$ *with* $2^{O(k \log |\Sigma|)}$ *many states that accepts*

$$L(\mathrm{Edit}^{<}(\Sigma, k)) = \{u \otimes w \ : \ \Delta(u, w) < k\}. \quad (1)$$

This result is also asymptotically optimal.

Edit Distance vs Diversity of Solutions

Lemma 8

Let Σ be an alphabet with $|\Sigma| \geq 2$ and $k \in \mathbb{N}_{>0}$. There is an NFA $\mathrm{Edit}^{<}(\Sigma, k)$ with $2^{O(k \log |\Sigma|)}$ many states that accepts

$$L(\mathrm{Edit}^{<}(\Sigma, k)) = \{u \otimes w \ : \ \Delta(u, w) < k\}. \tag{1}$$

This result is also asymptotically optimal.

Lemma 9

Let Σ be an alphabet with $|\Sigma| \geq 2$ and $k \in \mathbb{N}_{>0}$. There is a DFA $\mathrm{Edit}^{\geq}(\Sigma, k)$ with $2^{2^{O(k \log |\Sigma|)}}$ states accepting the following language.

$$L(\mathrm{Edit}^{\geq}(\Sigma, k)) = \{u \otimes w \ : \ \Delta(u, w) \geq k\}. \tag{2}$$

Min Diversity

Let $W \subseteq \Sigma^*$ be a finite set of strings. The *min-diversity* of W, denoted by $\mathrm{MinDiv}(W)$ is defined as the minimum edit distance among any pair of strings in W.

$$\mathrm{MinDiv}(W) = \min_{u,w \in W} \Delta(u, w) \tag{3}$$

Lemma 10

Let $A = (Q, \Sigma, \delta, q_0, F)$ be DFA over Σ, $|\Sigma| \geq 2$. For each $r, k \in \mathbb{N}^+$, one can construct a DFA $\mathrm{MinDiv}(A, r, k)$ with $2^{r^2 \cdot 2^{O(k \cdot \log |\Sigma|)}} \cdot |Q|^r$ many states accepting the language of all compound words $u_1 \otimes \cdots \otimes u_r \in (\Sigma \cup \{\Box\})^{\times r}$ that satisfy

$$\forall i \in [r](u_i \in L(A)) \wedge \mathrm{MinDiv}(\{u_1, \ldots, u_r\}) \geq k.$$

Its construction takes $O\left(2^{r^2 \cdot 2^{O(k \log |\Sigma|)}} \cdot |A|^r\right)$ time.

Main Technical Result

Theorem 11

Let $A = (Q, \Sigma, \delta)$ be a DFA and $B = (Q', \Sigma, \delta', Q'_0, F')$ be an NFA. One can determine in time $O(f_A(r,k) \cdot |Q'|^r \log(|Q'|))$ if there is a set $W \subseteq L(B)$ with r strings such that each word in W is subsequence-minimal synchronizing for A and $\mathrm{MinDiv}(W) \geq k$.

Proof of the Main Technical Result

Proof.

By combining Corollary 6 with Lemma 10, we can construct a DFA A' accepting the language of all compound words $u_1 \otimes \cdots \otimes u_r \in (\Sigma \cup \{\square\})^{\times r}$ such that for each $i \in [r]$, u_i is a subsequence-minimal synchronizing word for A, and $\mathrm{MinDiv}(\{u_1, \ldots, u_r\}) \geq k$.

Proof of the Main Technical Result

Proof.

By combining Corollary 6 with Lemma 10, we can construct a DFA A' accepting the language of all compound words $u_1 \otimes \cdots \otimes u_r \in (\Sigma \cup \{\square\})^{\times r}$ such that for each $i \in [r]$, u_i is a subsequence-minimal synchronizing word for A, and $\mathrm{MinDiv}(\{u_1, \ldots, u_r\}) \geq k$. The DFA A' has $2^{r^2 \cdot 2^{O(k \cdot \log |\Sigma|)}} \cdot (2^{|Q|})^r$ many states and can be constructed in similar time.

Proof of the Main Technical Result

Proof.

By combining Corollary 6 with Lemma 10, we can construct a DFA A' accepting the language of all compound words $u_1 \otimes \cdots \otimes u_r \in (\Sigma \cup \{\square\})^{\times r}$ such that for each $i \in [r]$, u_i is a subsequence-minimal synchronizing word for A, and $\mathrm{MinDiv}(\{u_1, \ldots, u_r\}) \geq k$. The DFA A' has $2^{r^2 \cdot 2^{O(k \cdot \log |\Sigma|)}} \cdot (2^{|Q|})^r$ many states and can be constructed in similar time. Conversely, using an $(r+1)$-fold product automaton construction, an NFA B' with $(2|Q'|)^r + 1$ many states can be constructed that accepts

$$\{u_1 \otimes u_2 \cdots \otimes u_r : \forall i \in [r](u_i \in L(B))\}\,.$$

Checking if the product automaton C of A' and B' accepts any compound words solves the proposed problem. This final check takes time linear in the size of C, which is $O\left(2^{r^2 \cdot \left(|Q| + 2^{O(k \cdot \log |\Sigma|)}\right)} \cdot |Q'|^r \log(|Q'|)\right)$. □

Table of Contents

The Curse of Subsequence Minimality 1

Problem Name: $\textsc{Min-SubSequence-SW}$
Given: DFA $A = (Q, \Sigma, \delta)$ and a word $w \in \Sigma^*$ synchronizing A.
Question: Is w a minimal synchronizing word with respect to the subsequence order?

The Curse of Subsequence Minimality 1

Problem Name: Min-SubSequence-SW
Given: DFA $A = (Q, \Sigma, \delta)$ and a word $w \in \Sigma^*$ synchronizing A.
Question: Is w a minimal synchronizing word with respect to the subsequence order?

Theorem 12

Min-SubSequence-SW *is coNP-complete, even for DFAs over a binary input alphabet.*

The Curse of Subsequence Minimality 1

Problem Name: $\textsc{Min-SubSequence-SW}$
Given: DFA $A = (Q, \Sigma, \delta)$ and a word $w \in \Sigma^*$ synchronizing A.
Question: Is w a minimal synchronizing word with respect to the subsequence order?

Theorem 12

$\textsc{Min-SubSequence-SW}$ *is coNP-complete, even for DFAs over a binary input alphabet.*

Proof Idea: Reduce from $\textsc{Hitting Set}$ to $\textsc{Co-Min-SubSequence-SW}$.

This is a complexity-theoretic justification of Corollary 6.

The Curse of Subsequence Minimality 2

Recall that any synchonizable DFA has a synchronizing word of at most cubic length.
(Quadratic length? Černý's Conjecture!)

The Curse of Subsequence Minimality 2

Recall that any synchonizable DFA has a synchronizing word of at most cubic length.
(Quadratic length? Černý's Conjecture!)

Proposition 13

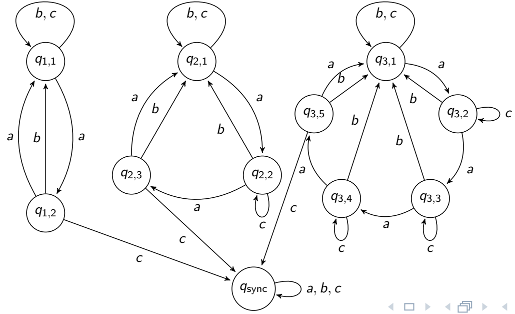
Some subsequence-minimal (or also infix-minimal) synchronizing words can be of exponential length, even for DFAs with a ternary input alphabet.

The Curse of Subsequence Minimality 2

Recall that any synchonizable DFA has a synchronizing word of at most cubic length.
(Quadratic length? Černý's Conjecture!)

Proposition 13

Some subsequence-minimal (or also infix-minimal) synchronizing words can be of exponential length, even for DFAs with a ternary input alphabet.



The Curse of Subsequence Minimality 3

Theorem 14

Let A be a DFA. Then, on input A, computing the diversity of the set of all subsequence-minimal synchronizing words of A is #P-hard, even on ternary input alphabets.

The Curse of Subsequence Minimality 3

Theorem 14

Let A be a DFA. Then, on input A, computing the diversity of the set of all subsequence-minimal synchronizing words of A is #P-hard, even on ternary input alphabets.

Proof Idea: Reduce from $\#$ $3\text{-}\mathrm{SAT}$.

Table of Contents

Conformant Planning 1

A *planning domain* can be abstracted as a 4-tuple $D = (Q, \Sigma, \delta, P)$, where Q is a set of *states*, Σ is a set of *actions*, $\delta \subseteq Q \times \Sigma \times Q$ is a transition relation, and P is a function that assigns a set $P(q)$ of *propositions* (or *beliefs*) to each state $q \in Q$.

Conformant Planning 1

A *planning domain* can be abstracted as a 4-tuple $D = (Q, \Sigma, \delta, P)$, where Q is a set of *states*, Σ is a set of *actions*, $\delta \subseteq Q \times \Sigma \times Q$ is a transition relation, and P is a function that assigns a set $P(q)$ of *propositions* (or *beliefs*) to each state $q \in Q$.

Have you seen this concept before?

Conformant Planning 1

A *planning domain* can be abstracted as a 4-tuple $D = (Q, \Sigma, \delta, P)$, where Q is a set of *states*, Σ is a set of *actions*, $\delta \subseteq Q \times \Sigma \times Q$ is a transition relation, and P is a function that assigns a set $P(q)$ of *propositions* (or *beliefs*) to each state $q \in Q$.
Have you seen this concept before?
The first three components clearly correspond to the first three components of an NFA.

Conformant Planning 1

A *planning domain* can be abstracted as a 4-tuple $D = (Q, \Sigma, \delta, P)$, where Q is a set of *states*, Σ is a set of *actions*, $\delta \subseteq Q \times \Sigma \times Q$ is a transition relation, and P is a function that assigns a set $P(q)$ of *propositions* (or *beliefs*) to each state $q \in Q$.
Have you seen this concept before?
The first three components clearly correspond to the first three components of an NFA.
$P(q)$ is the set of beliefs that are known to hold at state q, and a transition $(q, a, q') \in \delta$ indicates that the set of beliefs $P(q)$ should be updated to $P(q')$ if action a is taken from state q.

Conformant Planning 2

A *planning problem* is a triple (D, I, G) where $D = (Q, \Sigma, \delta, P)$ is a planning domain, $I \subseteq Q$ is a set of *initial states*, and $G \subseteq Q$ is a set of *goal states*.

Conformant Planning 2

A *planning problem* is a triple (D, I, G) where $D = (Q, \Sigma, \delta, P)$ is a planning domain, $I \subseteq Q$ is a set of *initial states*, and $G \subseteq Q$ is a set of *goal states*. Now, apart from P, we really have an NFA!

Conformant Planning 2

A *planning problem* is a triple (D, I, G) where $D = (Q, \Sigma, \delta, P)$ is a planning domain, $I \subseteq Q$ is a set of *initial states*, and $G \subseteq Q$ is a set of *goal states*. Now, apart from P, we really have an NFA!
A word $u \in \Sigma^*$ is called a *plan*. An action a is said to be *applicable* in a state q if there is some state q' such that $(q, a, q') \in \delta$. Such an action a is applicable in a set of states S if it is applicable in *every* state of S.
A plan u is said to be *applicable* in a set of states S if either $u = \varepsilon$ and S is non-empty, or $u = au'$ for some action a and some plan $u' \in \Sigma^*$, a is applicable on S and u' is applicable in $\delta(S, a)$.

Conformant Planning 2

A *planning problem* is a triple (D, I, G) where $D = (Q, \Sigma, \delta, P)$ is a planning domain, $I \subseteq Q$ is a set of *initial states*, and $G \subseteq Q$ is a set of *goal states*. Now, apart from P, we really have an NFA!

A word $u \in \Sigma^*$ is called a *plan*. An action a is said to be *applicable* in a state q if there is some state q' such that $(q, a, q') \in \delta$. Such an action a is applicable in a set of states S if it is applicable in *every* state of S.

A plan u is said to be *applicable* in a set of states S if either $u = \varepsilon$ and S is non-empty, or $u = au'$ for some action a and some plan $u' \in \Sigma^*$, a is applicable on S and u' is applicable in $\delta(S, a)$.

Definition 15 (Conformant Plan)

Let (D, I, G) be a planning problem with planning domain $D = (Q, \Sigma, \delta, P)$. A plan $u \in \Sigma^*$ is conformant for (D, I, G) if the following conditions are satisfied:

1. u is applicable in I, and
2. $\delta(I, u) \subseteq G$.

Conformant Planning 3

Lemma 16 (Conformant Words)

Let $D = (Q, \Sigma, \delta, P)$ *be a planning domain, and* (D, I, G) *be a planning problem. One can construct in time* $|\Sigma| \cdot 2^{O(|Q|)}$ *a DFA* $\mathrm{Conf}(D, I, G)$ *with* $2^{|Q|}$ *states such that*

$$L(\mathrm{Conf}(D, I, G)) = \{u \in \Sigma^* \ : \ u \textit{ is conformant for } (D, I, G)\}.$$

Conformant Planning 3

Lemma 16 (Conformant Words)

Let $D = (Q, \Sigma, \delta, P)$ be a planning domain, and (D, I, G) be a planning problem. One can construct in time $|\Sigma| \cdot 2^{O(|Q|)}$ a DFA $\mathrm{Conf}(D, I, G)$ with $2^{|Q|}$ states such that

$$L(\mathrm{Conf}(D, I, G)) = \{u \in \Sigma^* \ : \ u \text{ is conformant for } (D, I, G)\}.$$

The proof of this lemma is similar to the power-automaton construction,

Conformant Planning 3

Lemma 16 (Conformant Words)

Let $D = (Q, \Sigma, \delta, P)$ be a planning domain, and (D, I, G) be a planning problem. One can construct in time $|\Sigma| \cdot 2^{O(|Q|)}$ a DFA $\mathrm{Conf}(D, I, G)$ with $2^{|Q|}$ states such that

$$L(\mathrm{Conf}(D, I, G)) = \{u \in \Sigma^* \ : \ u \textit{ is conformant for } (D, I, G)\}.$$

The proof of this lemma is similar to the power-automaton construction, BUT:
Let $\mathrm{Conf}(D, I, G)$ be the DFA $\mathcal{A}$ whose set of states $\mathcal{Q}$ is the set of all subsets of Q, ...
The transition function $\Delta : \mathcal{Q} \times \Sigma \to \mathcal{Q}$ sends each pair $(S, a) \in \mathcal{Q} \times \Sigma$ to the state $\{q' \ : \ \exists q \in S, (q, a, q') \in \delta\}$ if a is applicable in every state of S, and to the state $\emptyset$ if a is not applicable from some state of S.

Conformant Planning 4

A conformant plan u is *subsequence-minimal* if no proper subsequence of u is a conformant plan for (D, I, G).

Corollary 17

One can construct in time $O(|\Sigma| \cdot 2^{2^{|Q|}+|Q|})$ *a DFA* $\mathrm{ConfMinSubseq}(A)$ *with at most* $2^{2^{|Q|}+|Q|}$ *many states accepting the language*

$$L(\mathrm{ConfMinSubseq}(A)) = \{u \in \Sigma^* : u \textit{ is subsequence-minimal conformant for } (D, I, G)\}.$$

Conformant Planning 4

A conformant plan u is *subsequence-minimal* if no proper subsequence of u is a conformant plan for (D, I, G).

Corollary 17

One can construct in time $O(|\Sigma| \cdot 2^{2^{|Q|}+|Q|})$ *a DFA* $\mathrm{ConfMinSubseq}(A)$ *with at most* $2^{2^{|Q|}+|Q|}$ *many states accepting the language*

$$L(\mathrm{ConfMinSubseq}(A)) = \{u \in \Sigma^* : u \text{ is subsequence-minimal conformant for } (D, I, G)\}.$$

Theorem 18

Given an NFA $B = (Q', \Sigma, \delta', Q'_0, F')$*, one can determine in time* $O(f_D(r, k) \cdot |Q'|^r \log(|Q'|))$ *whether there is a set* $W \subseteq L(B)$ *with* r *plans s.t. (1) each plan in* W *is subsequence-minimal conformant for* (D, I, G) *and (2)* $\mathrm{MinDiv}(W) \geq k$*.*

Conclusions

- Be brave!
  Sell automata-theoretic constructions / results outside our circles.
- Don't get frustrated if you fail the first time.

Conclusions

- Be brave!
  Sell automata-theoretic constructions / results outside our circles.
- Don't get frustrated if you fail the first time.
- Try to read through papers of "other modern areas".
  You will often find automata theory "in disguise".
  This is a good starting point for your research!

Conclusions

- Be brave!
  Sell automata-theoretic constructions / results outside our circles.
- Don't get frustrated if you fail the first time.
- Try to read through papers of "other modern areas".
  You will often find automata theory "in disguise".
  This is a good starting point for your research!
- New technical challenges will easily show up.
  For instance: What about subword-minimality in our project?
  Or: What about representative sets of synchronizing words?