Careful Synchronization of Partial Automata

Mikhail Volkov

Ural State University, Ekaterinburg, Russia

SATA, Sept 1st, 2008

Complete automata

First consider a complete DFA: $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

- Q is the state set
- Σ is the input alphabet
- $\delta : Q \times \Sigma \to Q$ is the transition function

Here δ is supposed to be everywhere defined.

Complete automata

First consider a complete DFA: $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

- Q is the state set
- Σ is the input alphabet
- $\delta : Q \times \Sigma \to Q$ is the transition function

Here δ is supposed to be everywhere defined.

SATA, Sept 1st, 2008

Complete automata

First consider a complete DFA: $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

- Q is the state set
- Σ is the input alphabet
- $\delta : Q \times \Sigma \to Q$ is the transition function

Here δ is supposed to be everywhere defined.

Complete automata

First consider a complete DFA: $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

- Q is the state set
- Σ is the input alphabet
- $\delta : Q \times \Sigma \to Q$ is the transition function

Here δ is supposed to be everywhere defined.

Complete automata

First consider a complete DFA: $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

- Q is the state set
- Σ is the input alphabet
- $\delta : Q \times \Sigma \to Q$ is the transition function

Here δ is supposed to be everywhere defined.

Synchronization of complete automata

Definition

A DFA $\mathscr{A}$ is called synchronizing if there exists a word $w \in \Sigma^*$ whose action resets $\mathscr{A}$, that is, leaves the automaton in one particular state no matter which state in Q it started at: $\delta(q, w) = \delta(q', w)$ for all $q, q' \in Q$.

In symbols, $|\delta(Q, w)| = 1$. Here $|\delta(Q, w)| = \{\delta(q, w) \mid q \in Q\}$.
Any w with this property is a reset word for $\mathscr{A}$.

Synchronization of complete automata

Definition

A DFA $\mathscr{A}$ is called synchronizing if there exists a word $w \in \Sigma^*$ whose action resets $\mathscr{A}$, that is, leaves the automaton in one particular state no matter which state in Q it started at: $\delta(q, w) = \delta(q', w)$ for all $q, q' \in Q$.

In symbols, $|\delta(Q, w)| = 1$. Here $|\delta(Q, w)| = \{\delta(q, w) \mid q \in Q\}$.
Any w with this property is a reset word for $\mathscr{A}$.

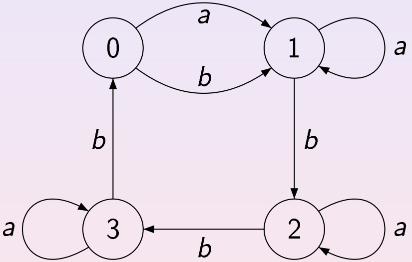
Synchronization of complete automata

Definition

A DFA $\mathscr{A}$ is called synchronizing if there exists a word $w \in \Sigma^*$ whose action resets $\mathscr{A}$, that is, leaves the automaton in one particular state no matter which state in Q it started at: $\delta(q, w) = \delta(q', w)$ for all $q, q' \in Q$.

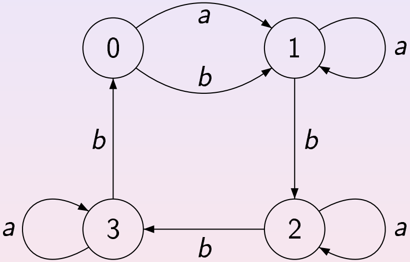
In symbols, $|\delta(Q, w)| = 1$. Here $|\delta(Q, w)| = \{\delta(q, w) \mid q \in Q\}$. Any w with this property is a reset word for $\mathscr{A}$.

An example



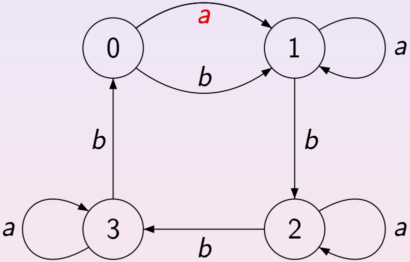
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



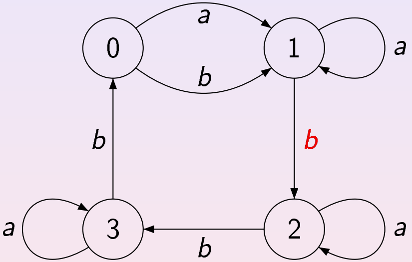
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



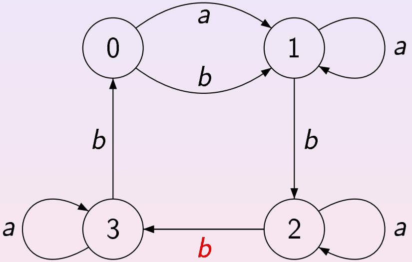
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



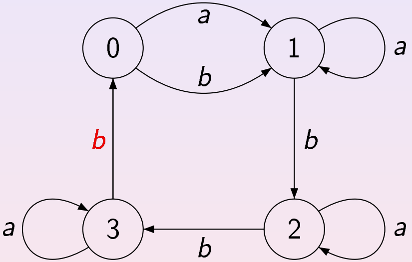
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



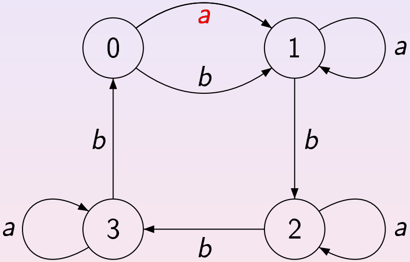
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



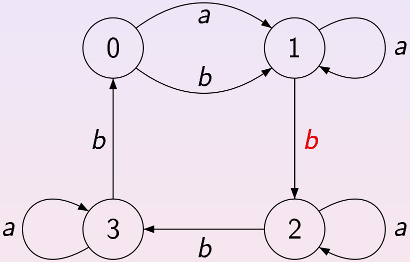
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



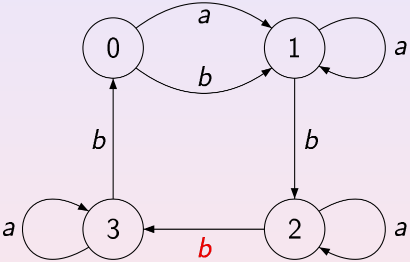
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



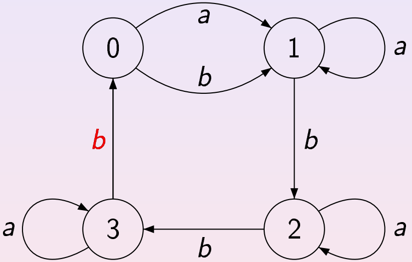
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



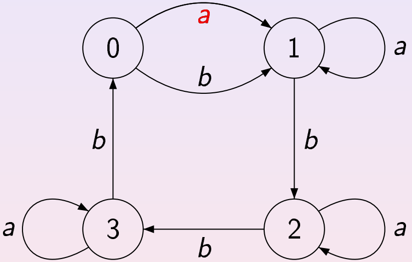
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



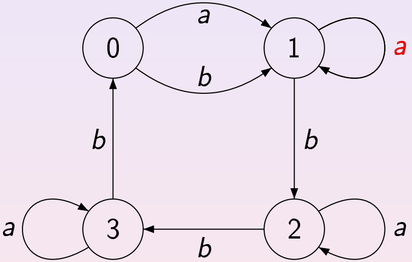
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



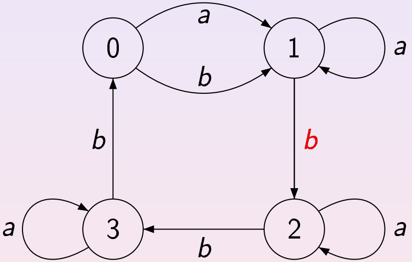
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



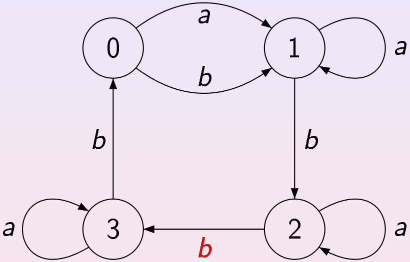
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



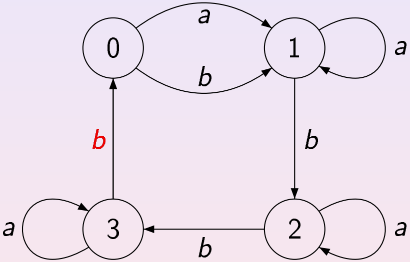
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



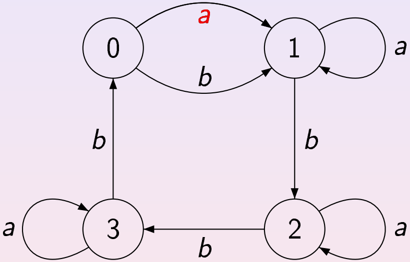
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



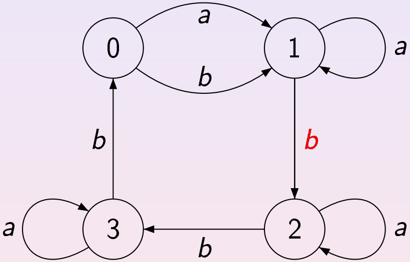
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



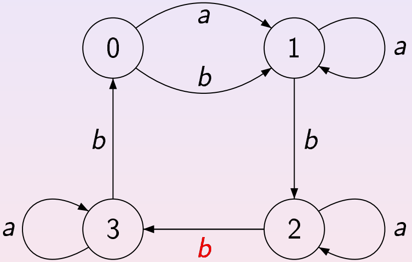
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



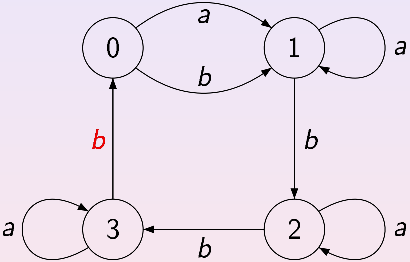
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



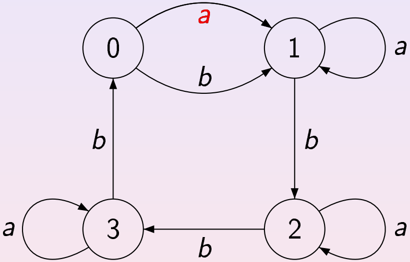
A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

An example



A reset word is *abbbabbba*: applying it at any state brings the automaton to the state 1

History and motivation

The notion was formalized in 1964 in a paper by J. Černý (Poznámka k homogénnym eksperimentom s konečnými automatami, Matematicko-fyzikalny Časopis Slovensk. Akad. Vied, 14, no.3, 208–216 [in Slovak]) though implicitly it had been around since at least 1956.

The idea of synchronization is pretty natural and of obvious importance: we aim to restore control over a device whose current state is not known.

Think of a satellite which loops around the Moon and cannot be controlled from the Earth while "behind" the Moon (Černý's original motivation).

History and motivation

The notion was formalized in 1964 in a paper by J. Černý (Poznámka k homogénnym eksperimentom s konečnými automatami, Matematicko-fyzikalny Časopis Slovensk. Akad. Vied, 14, no.3, 208–216 [in Slovak]) though implicitly it had been around since at least 1956.

The idea of synchronization is pretty natural and of obvious importance: we aim to restore control over a device whose current state is not known.

Think of a satellite which loops around the Moon and cannot be controlled from the Earth while "behind" the Moon (Černý's original motivation).

History and motivation

The notion was formalized in 1964 in a paper by J. Černý (Poznámka k homogénnym eksperimentom s konečnými automatami, Matematicko-fyzikalny Časopis Slovensk. Akad. Vied, 14, no.3, 208–216 [in Slovak]) though implicitly it had been around since at least 1956.

The idea of synchronization is pretty natural and of obvious importance: we aim to restore control over a device whose current state is not known.

Think of a satellite which loops around the Moon and cannot be controlled from the Earth while “behind” the Moon (Černý’s original motivation).

Partial automata

Now consider a partial DFA: $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

The only (but essential!) difference is that now the transition function δ is allowed to be partial, that is $\delta(q, a)$ can be undefined for some $q \in Q$ and $a \in \Sigma$.

Challenge

What is an appropriate extension of the synchronizability concept to partial automata?

Partial automata

Now consider a partial DFA: $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.
The only (but essential!) difference is that now the transition function δ is allowed to be partial, that is $\delta(q, a)$ can be undefined for some $q \in Q$ and $a \in \Sigma$.

Challenge

What is an appropriate extension of the synchronizability concept to partial automata?

Partial automata

Now consider a partial DFA: $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.
The only (but essential!) difference is that now the transition function δ is allowed to be partial, that is $\delta(q, a)$ can be undefined for some $q \in Q$ and $a \in \Sigma$.

Challenge

What is an appropriate extension of the synchronizability concept to partial automata?

Careful synchronization

Say that $w = a_1 \cdots a_\ell$ with $a_1, \ldots, a_\ell \in \Sigma$ is a careful reset word for a partial DFA $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ if:

1. $\delta(q, a_1)$ is defined for all $q \in Q$,
2. $\delta(q, a_i)$ with $1 < i \leq \ell$ is defined for all $q \in \delta(Q, a_1 \cdots a_{i-1})$,
3. $|\delta(Q, w)| = 1$.

Thus, w is an instruction that the automaton is always able to follow and that brings it to a state which is independent of the initial state.

Careful synchronization

Say that $w = a_1 \cdots a_\ell$ with $a_1, \ldots, a_\ell \in \Sigma$ is a careful reset word for a partial DFA $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ if:

1. $\delta(q, a_1)$ is defined for all $q \in Q$,
2. $\delta(q, a_i)$ with $1 < i \leq \ell$ is defined for all $q \in \delta(Q, a_1 \cdots a_{i-1})$,
3. $|\delta(Q, w)| = 1$.

Thus, w is an instruction that the automaton is always able to follow and that brings it to a state which is independent of the initial state.

Careful synchronization

Say that $w = a_1 \cdots a_\ell$ with $a_1, \ldots, a_\ell \in \Sigma$ is a careful reset word for a partial DFA $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ if:

1. $\delta(q, a_1)$ is defined for all $q \in Q$,
2. $\delta(q, a_i)$ with $1 < i \leq \ell$ is defined for all $q \in \delta(Q, a_1 \cdots a_{i-1})$,
3. $|\delta(Q, w)| = 1$.

Thus, w is an instruction that the automaton is always able to follow and that brings it to a state which is independent of the initial state.

Careful synchronization

Say that $w = a_1 \cdots a_\ell$ with $a_1, \ldots, a_\ell \in \Sigma$ is a careful reset word for a partial DFA $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ if:

1. $\delta(q, a_1)$ is defined for all $q \in Q$,
2. $\delta(q, a_i)$ with $1 < i \leq \ell$ is defined for all $q \in \delta(Q, a_1 \cdots a_{i-1})$,
3. $|\delta(Q, w)| = 1$.

Thus, w is an instruction that the automaton is always able to follow and that brings it to a state which is independent of the initial state.

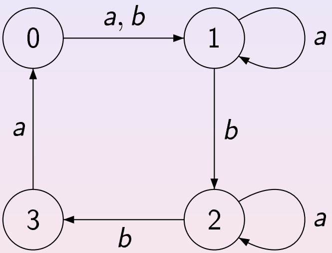
Careful synchronization

Say that $w = a_1 \cdots a_\ell$ with $a_1, \ldots, a_\ell \in \Sigma$ is a careful reset word for a partial DFA $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ if:

1. $\delta(q, a_1)$ is defined for all $q \in Q$,
2. $\delta(q, a_i)$ with $1 < i \leq \ell$ is defined for all $q \in \delta(Q, a_1 \cdots a_{i-1})$,
3. $|\delta(Q, w)| = 1$.

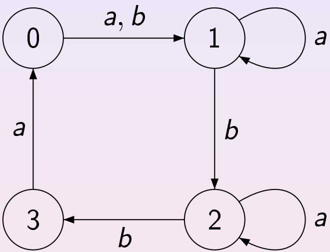
Thus, w is an instruction that the automaton is always able to follow and that brings it to a state which is independent of the initial state.

An example



A careful reset word is *aababaa*: applying it at any state is possible and brings the automaton to the state 1

An example



A careful reset word is *aababaa*: applying it at any state is possible and brings the automaton to the state 1

"History" and motivation

The notion seems to be quite natural but it has appeared only recently in a paper by M. Ito and M. Ito and K. Shikishima-Tsuji (Some results on directable automata, in J. Karhumäki, H. Mauer, Gh. Păun, and G. Rozenberg (eds.), Theory Is Forever. Essays Dedicated to Arto Salomaa on the Occasion of His 70th Birthday. [Lect. Notes Comp. Sci. **3113**] Springer, 2004, 125–133).

Most results about careful synchronization are due to P. Martjugin, a PhD student in our group.

Motivation: all applications from the complete case carry over + "control under lack of information".

"History" and motivation

The notion seems to be quite natural but it has appeared only recently in a paper by M. Ito and M. Ito and K. Shikishima-Tsuji (Some results on directable automata, in J. Karhumäki, H. Mauer, Gh. Păun, and G. Rozenberg (eds.), Theory Is Forever. Essays Dedicated to Arto Salomaa on the Occasion of His 70th Birthday. [Lect. Notes Comp. Sci. **3113**] Springer, 2004, 125–133).

Most results about careful synchronization are due to P. Martjugin, a PhD student in our group.

Motivation: all applications from the complete case carry over + "control under lack of information".

"History" and motivation

The notion seems to be quite natural but it has appeared only recently in a paper by M. Ito and M. Ito and K. Shikishima-Tsuji (Some results on directable automata, in J. Karhumäki, H. Mauer, Gh. Păun, and G. Rozenberg (eds.), Theory Is Forever. Essays Dedicated to Arto Salomaa on the Occasion of His 70th Birthday. [Lect. Notes Comp. Sci. **3113**] Springer, 2004, 125–133).

Most results about careful synchronization are due to P. Martjugin, a PhD student in our group.

Motivation: all applications from the complete case carry over + "control under lack of information".

Power automata

Question

Given an automaton, how to determine whether or not it is synchronizing?

For the complete case, a straightforward solution comes from the classic power automaton construction.

The power automaton $\mathcal{P}(\mathscr{A})$ of a complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$.

$w \in \Sigma^*$ is a reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}(\mathscr{A})$ starting at Q and ending at a singleton.

Power automata

Question

Given an automaton, how to determine whether or not it is synchronizing?

For the complete case, a straightforward solution comes from the classic power automaton construction.

The power automaton $\mathcal{P}(\mathscr{A})$ of a complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$.

$w \in \Sigma^*$ is a reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}(\mathscr{A})$ starting at Q and ending at a singleton.

Power automata

Question

Given an automaton, how to determine whether or not it is synchronizing?

For the complete case, a straightforward solution comes from the classic power automaton construction.

The power automaton $\mathcal{P}(\mathscr{A})$ of a complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$.

$w \in \Sigma^*$ is a reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}(\mathscr{A})$ starting at Q and ending at a singleton.

Power automata

Question

Given an automaton, how to determine whether or not it is synchronizing?

For the complete case, a straightforward solution comes from the classic power automaton construction.

The power automaton $\mathcal{P}(\mathscr{A})$ of a complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$.

$w \in \Sigma^*$ is a reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}(\mathscr{A})$ starting at Q and ending at a singleton.

Power automata

Question

Given an automaton, how to determine whether or not it is synchronizing?

For the complete case, a straightforward solution comes from the classic power automaton construction.

The power automaton $\mathcal{P}(\mathscr{A})$ of a complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$.

$w \in \Sigma^*$ is a reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}(\mathscr{A})$ starting at Q and ending at a singleton.

Power automata

Question

Given an automaton, how to determine whether or not it is synchronizing?

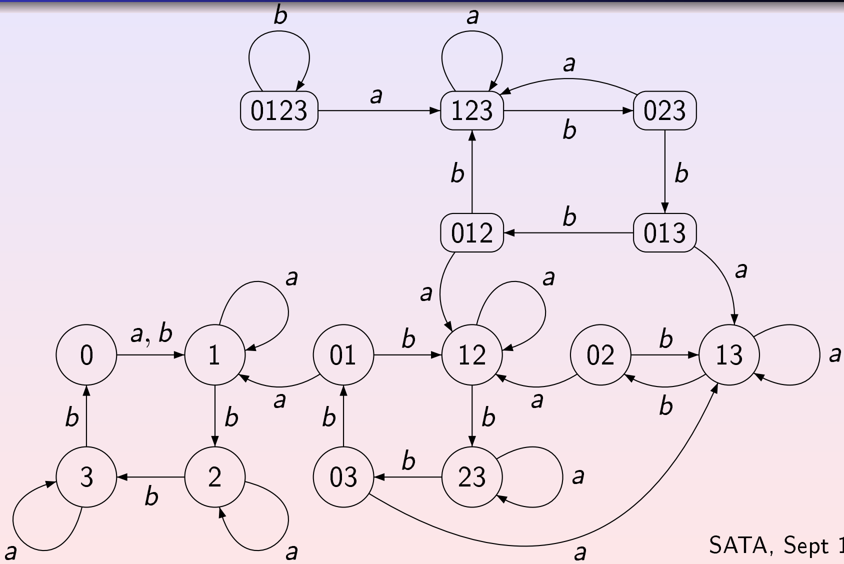
For the complete case, a straightforward solution comes from the classic power automaton construction.

The power automaton $\mathcal{P}(\mathscr{A})$ of a complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

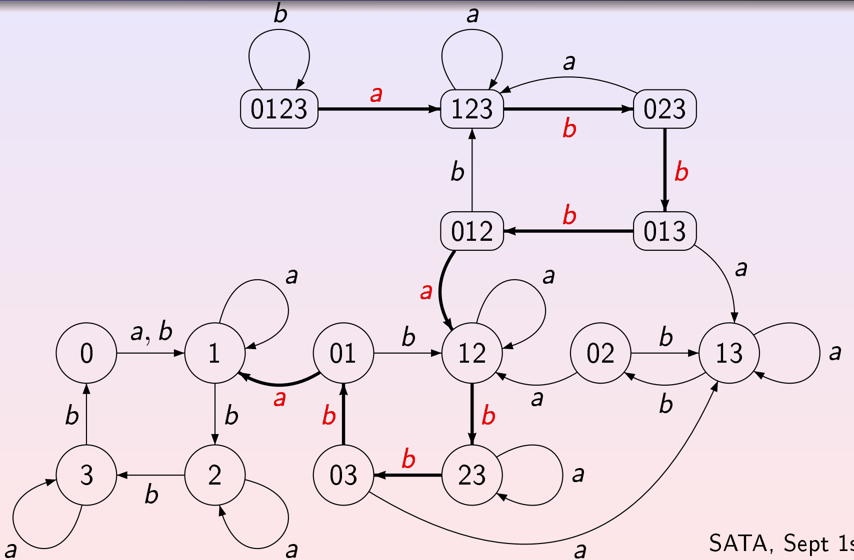
- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$.

$w \in \Sigma^*$ is a reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}(\mathscr{A})$ starting at Q and ending at a singleton.

An example


b
a
a
0123
a
123
023
b
b
b
b
012
013
a
a
a
a
a
a, b
b
0
1
01
12
02
b
13
a
a
a
b
b
b
b
b
3
2
03
b
23
a
b
a
a
a

An example



SATA, Sept 1st, 2008

Partial power automata

A similar approach works for the partial case.

Here we define the partial power automaton $\mathcal{P}'(\mathscr{A})$ of a partial DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$ provided $\delta(p, a)$ is defined for each $p \in P$; otherwise $\delta(P, a)$ is undefined.

$w \in \Sigma^*$ is a careful reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton.

Partial power automata

A similar approach works for the partial case.
Here we define the partial power automaton $\mathcal{P}'(\mathscr{A})$ of a partial DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$ provided $\delta(p, a)$ is defined for each $p \in P$; otherwise $\delta(P, a)$ is undefined.

$w \in \Sigma^*$ is a careful reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton.

Partial power automata

A similar approach works for the partial case.

Here we define the partial power automaton $\mathcal{P}'(\mathscr{A})$ of a partial DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$ provided $\delta(p, a)$ is defined for each $p \in P$; otherwise $\delta(P, a)$ is undefined.

$w \in \Sigma^*$ is a careful reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton.

Partial power automata

A similar approach works for the partial case.

Here we define the partial power automaton $\mathcal{P}'(\mathscr{A})$ of a partial DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$ provided $\delta(p, a)$ is defined for each $p \in P$; otherwise $\delta(P, a)$ is undefined.

$w \in \Sigma^*$ is a careful reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton.

Partial power automata

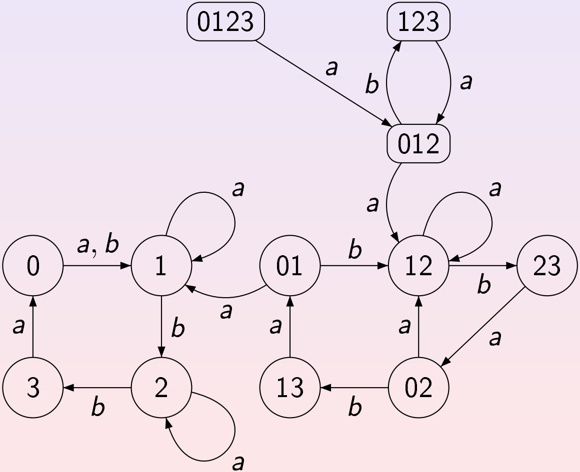
A similar approach works for the partial case.

Here we define the partial power automaton $\mathcal{P}'(\mathscr{A})$ of a partial DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$:

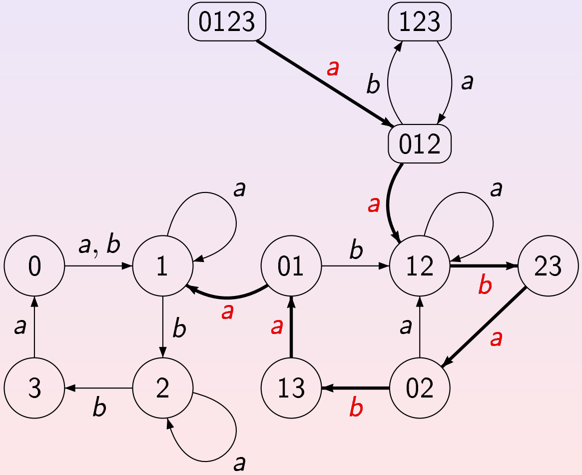
- states are the non-empty subsets of Q,
- $\delta(P, a) = \{\delta(p, a) \mid p \in P\}$ provided $\delta(p, a)$ is defined for each $p \in P$; otherwise $\delta(P, a)$ is undefined.

$w \in \Sigma^*$ is a careful reset word for $\mathscr{A}$ iff w labels a path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton.

An example



An example



SATA, Sept 1st, 2008

Efficiency: complete case

Algorithms arising from the power set construction are exponential w.r.t. the size of $\mathscr{A}$.

For the complete case, the following result by Černý gives a polynomial algorithm:

Proposition (Černý, 1964)

A complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta\rangle$ is synchronizing iff for every $q, q' \in Q$ there exists a word $w \in \Sigma^*$ such that $\delta(q, w) = \delta(q', w)$.

Efficiency: complete case

Algorithms arising from the power set construction are exponential w.r.t. the size of $\mathscr{A}$.

For the complete case, the following result by Černý gives a polynomial algorithm:

Proposition (Černý, 1964)

A complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is synchronizing iff for every $q, q' \in Q$ there exists a word $w \in \Sigma^*$ such that $\delta(q, w) = \delta(q', w)$.

Efficiency: complete case

Algorithms arising from the power set construction are exponential w.r.t. the size of $\mathscr{A}$.

For the complete case, the following result by Černý gives a polynomial algorithm:

Proposition (Černý, 1964)

A complete DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is synchronizing iff for every $q, q' \in Q$ there exists a word $w \in \Sigma^*$ such that $\delta(q, w) = \delta(q', w)$.

Efficiency: partial case

In the partial case, we have the following result by Martjugin:

Theorem (Martjugin, in print)

Checking if a given partial DFA is carefully synchronizing is PSPACE-complete.

Efficiency: partial case

In the partial case, we have the following result by Martjugin:

Theorem (Martjugin, in print)

Checking if a given partial DFA is carefully synchronizing is PSPACE-complete.

Upper bounds

The power automaton construction gives an upper bound for the length of the shortest reset word of a given synchronizing automaton $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ (complete or partial): if the automaton has n states, it has a reset word of length at most $2^n - n - 1$. (The shortest path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton does not visit any state twice and ends at the first singleton it reaches.)

For the complete case, we know a much better bound due to Pin:

Theorem (Pin, 1983)

A complete synchronizing automaton with n states has a reset word of length at most $\dfrac{n^3 - n}{6}$.

Upper bounds

The power automaton construction gives an upper bound for the length of the shortest reset word of a given synchronizing automaton $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ (complete or partial): if the automaton has n states, it has a reset word of length at most $2^n - n - 1$. (The shortest path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton does not visit any state twice and ends at the first singleton it reaches.)

For the complete case, we know a much better bound due to Pin:

Theorem (Pin, 1983)

A complete synchronizing automaton with n states has a reset word of length at most $\dfrac{n^3 - n}{6}$.

Upper bounds

The power automaton construction gives an upper bound for the length of the shortest reset word of a given synchronizing automaton $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ (complete or partial): if the automaton has n states, it has a reset word of length at most $2^n - n - 1$. (The shortest path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton does not visit any state twice and ends at the first singleton it reaches.)

For the complete case, we know a much better bound due to Pin:

Theorem (Pin, 1983)

A complete synchronizing automaton with n states has a reset word of length at most $\frac{n^3 - n}{6}$.

Upper bounds

The power automaton construction gives an upper bound for the length of the shortest reset word of a given synchronizing automaton $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ (complete or partial): if the automaton has n states, it has a reset word of length at most $2^n - n - 1$. (The shortest path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton does not visit any state twice and ends at the first singleton it reaches.)

For the complete case, we know a much better bound due to Pin:

Theorem (Pin, 1983)

A complete synchronizing automaton with n states has a reset word of length at most $\dfrac{n^3 - n}{6}$.

Upper bounds

The power automaton construction gives an upper bound for the length of the shortest reset word of a given synchronizing automaton $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ (complete or partial): if the automaton has n states, it has a reset word of length at most $2^n - n - 1$. (The shortest path in $\mathcal{P}'(\mathscr{A})$ starting at Q and ending at a singleton does not visit any state twice and ends at the first singleton it reaches.)

For the complete case, we know a much better bound due to Pin:

Theorem (Pin, 1983)

A complete synchronizing automaton with n states has a reset word of length at most $\dfrac{n^3 - n}{6}$.

Upper vs lower bounds

We do not know any reasonable upper bound for the partial case. It definitely cannot be polynomial because there exist non-polynomial lower bounds.

The first such bound has been found by Ito and Shikishima-Tsuji:

Theorem (Ito and Shikishima-Tsuji, 2004)

There exists a series of partial carefully synchronizing automata with n states and shortest careful reset words of length $O(2^{\frac{n}{2}})$.

This lower bound has been slightly strengthened by Martjugin who constructed a series of partial carefully synchronizing automata with shortest careful reset words of length $O(3^{\frac{n}{3}})$.

Upper vs lower bounds

We do not know any reasonable upper bound for the partial case. It definitely cannot be polynomial because there exist non-polynomial lower bounds.

The first such bound has been found by Ito and Shikishima-Tsuji:

Theorem (Ito and Shikishima-Tsuji, 2004)

There exists a series of partial carefully synchronizing automata with n states and shortest careful reset words of length $O(2^{\frac{n}{2}})$.

This lower bound has been slightly strengthened by Martjugin who constructed a series of partial carefully synchronizing automata with shortest careful reset words of length $O(3^{\frac{n}{3}})$.

Upper vs lower bounds

We do not know any reasonable upper bound for the partial case. It definitely cannot be polynomial because there exist non-polynomial lower bounds.

The first such bound has been found by Ito and Shikishima-Tsuji:

Theorem (Ito and Shikishima-Tsuji, 2004)

There exists a series of partial carefully synchronizing automata with n states and shortest careful reset words of length $O(2^{\frac{n}{2}})$.

This lower bound has been slightly strengthened by Martjugin who constructed a series of partial carefully synchronizing automata with shortest careful reset words of length $O(3^{\frac{n}{3}})$.

Upper vs lower bounds

We do not know any reasonable upper bound for the partial case. It definitely cannot be polynomial because there exist non-polynomial lower bounds.

The first such bound has been found by Ito and Shikishima-Tsuji:

Theorem (Ito and Shikishima-Tsuji, 2004)

There exists a series of partial carefully synchronizing automata with n states and shortest careful reset words of length $O(2^{\frac{n}{2}})$.

This lower bound has been slightly strengthened by Martjugin who constructed a series of partial carefully synchronizing automata with shortest careful reset words of length $O(3^{\frac{n}{3}})$.

Upper vs lower bounds

We do not know any reasonable upper bound for the partial case. It definitely cannot be polynomial because there exist non-polynomial lower bounds.

The first such bound has been found by Ito and Shikishima-Tsuji:

Theorem (Ito and Shikishima-Tsuji, 2004)

There exists a series of partial carefully synchronizing automata with n states and shortest careful reset words of length $O(2^{\frac{n}{2}})$.

This lower bound has been slightly strengthened by Martjugin who constructed a series of partial carefully synchronizing automata with shortest careful reset words of length $O(3^{\frac{n}{3}})$.

Open problems

Thus, we have the following open problems:

Upper Bound Problem

Does there exists a subexponential (in # of states) upper bound for the length of careful reset words of partial carefully synchronizing automata?

Lower Bound Problem

Does there exists an exponential (in # of states) lower bound for the length of careful reset words of partial carefully synchronizing automata?

SATA, Sept 1st, 2008

Open problems

Thus, we have the following open problems:

Upper Bound Problem

Does there exists a subexponential (in # of states) upper bound for the length of careful reset words of partial carefully synchronizing automata?

Lower Bound Problem

Does there exists an exponential (in # of states) lower bound for the length of careful reset words of partial carefully synchronizing automata?

SATA, Sept 1st, 2008

Open problems

Thus, we have the following open problems:

Upper Bound Problem

Does there exists a subexponential (in # of states) upper bound for the length of careful reset words of partial carefully synchronizing automata?

Lower Bound Problem

Does there exists an exponential (in # of states) lower bound for the length of careful reset words of partial carefully synchronizing automata?

SATA, Sept 1st, 2008

The Černý automata

Recall that the situation is not easy even for the complete case.

We know an upper bound: there always exists a reset word of length $\dfrac{n^3 - n}{6}$. What about a lower bound?

In his 1964 paper Černý constructed a series $\mathscr{C}_n$, $n = 2, 3, \dots,$ of synchronizing automata over 2 letters. The states of $\mathscr{C}_n$ are the residues modulo n, and the input letters a and b act as follows:

$$\delta(0, a) = 1,\ \delta(m, a) = m \text{ for } 0 < m < n,\ \delta(m, b) = m+1 \pmod{n}.$$

The automaton in our first example is $\mathscr{C}_4$.

SATA, Sept 1st, 2008

The Černý automata

Recall that the situation is not easy even for the complete case. We know an upper bound: there always exists a reset word of length $\frac{n^3-n}{6}$. What about a lower bound?

In his 1964 paper Černý constructed a series $\mathscr{C}_n$, $n = 2, 3, \ldots,$ of synchronizing automata over 2 letters. The states of $\mathscr{C}_n$ are the residues modulo n, and the input letters a and b act as follows:

$$\delta(0, a) = 1,\ \delta(m, a) = m \text{ for } 0 < m < n,\ \delta(m, b) = m+1 \pmod{n}.$$

The automaton in our first example is $\mathscr{C}_4$.

The Černý automata

Recall that the situation is not easy even for the complete case.

We know an upper bound: there always exists a reset word of length $\dfrac{n^3-n}{6}$. What about a lower bound?

In his 1964 paper Černý constructed a series $\mathscr{C}_n$, $n = 2, 3, \ldots,$ of synchronizing automata over 2 letters. The states of $\mathscr{C}_n$ are the residues modulo n, and the input letters a and b act as follows:

$$\delta(0,a) = 1,\ \delta(m,a) = m \text{ for } 0 < m < n,\ \delta(m,b) = m+1 \pmod{n}.$$

The automaton in our first example is $\mathscr{C}_4$.

The Černý automata

Recall that the situation is not easy even for the complete case. We know an upper bound: there always exists a reset word of length $\dfrac{n^3-n}{6}$. What about a lower bound?

In his 1964 paper Černý constructed a series $\mathscr{C}_n$, $n = 2, 3, \dots,$ of synchronizing automata over 2 letters. The states of $\mathscr{C}_n$ are the residues modulo n, and the input letters a and b act as follows:

$$\delta(0,a) = 1,\ \delta(m,a) = m \text{ for } 0 < m < n, \delta(m,b) = m+1 \pmod{n}.$$

The automaton in our first example is $\mathscr{C}_4$.

The Černý automata

Recall that the situation is not easy even for the complete case. We know an upper bound: there always exists a reset word of length $\dfrac{n^3 - n}{6}$. What about a lower bound?

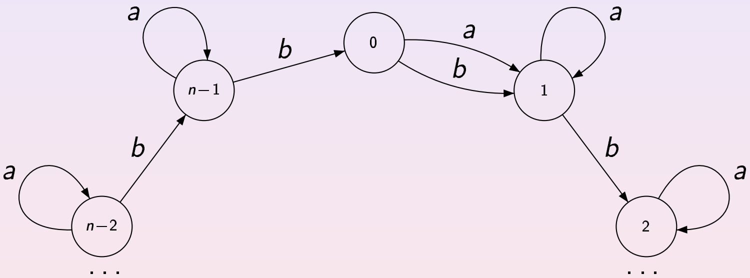
In his 1964 paper Černý constructed a series $\mathscr{C}_n$, $n = 2, 3, \ldots,$ of synchronizing automata over 2 letters. The states of $\mathscr{C}_n$ are the residues modulo n, and the input letters a and b act as follows:

$$\delta(0, a) = 1,\ \delta(m, a) = m \text{ for } 0 < m < n,\ \delta(m, b) = m+1 \pmod{n}.$$

The automaton in our first example is $\mathscr{C}_4$.

The Černý automata

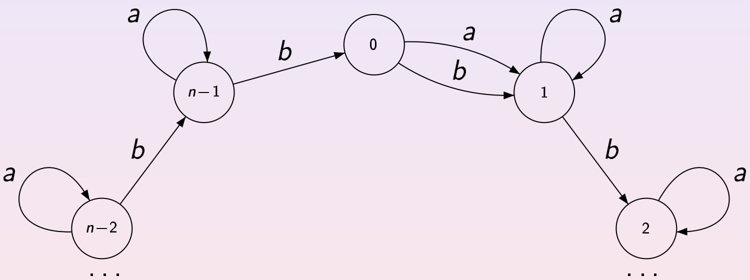
Here is a generic automaton from the Černý series:



Černý has proved that the shortest reset word for $\mathscr{C}_n$ is $(ab^{n-1})^{n-2}a$ of length $(n-1)^2$.

The Černý automata

Here is a generic automaton from the Černý series:



Černý has proved that the shortest reset word for $\mathscr{C}_n$ is $(ab^{n-1})^{n-2}a$ of length $(n-1)^2$.

The Černý conjecture

Černý conjectured that the automata $\mathscr{C}_n$ represent the worst possible case, that is, every complete synchronizing automata with n states admits a reset word of length at most $(n-1)^2$.

This simply looking conjecture is arguably the most longstanding open problem in the combinatorial theory of finite automata.

The Černý conjecture

Černý conjectured that the automata $\mathscr{C}_n$ represent the worst possible case, that is, every complete synchronizing automata with n states admits a reset word of length at most $(n-1)^2$.

This simply looking conjecture is arguably the most longstanding open problem in the combinatorial theory of finite automata.

Classic Road Coloring Problem

Strongly connected synchronizing automata

Assume that a complete synchronizing automaton is strongly connected as a digraph. Such an automaton can be reset to any state. That is, to every state q of the automaton one can assign an instruction (a reset word) w_q such that following w_q one will surely arrive at q from any initial state.

Classic Road Coloring Problem

Strongly connected synchronizing automata

Assume that a complete synchronizing automaton is strongly connected as a digraph. Such an automaton can be reset to any state. That is, to every state q of the automaton one can assign an instruction (a reset word) w_q such that following w_q one will surely arrive at q from any initial state.

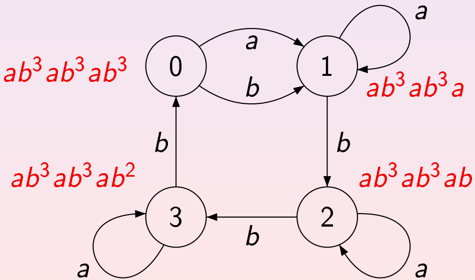
Strongly connected synchronizing automata

Assume that a complete synchronizing automaton is strongly connected as a digraph. Such an automaton can be reset to any state. That is, to every state q of the automaton one can assign an instruction (a reset word) w_q such that following w_q one will surely arrive at q from any initial state.

Classic Road Coloring Problem

Strongly connected synchronizing automata

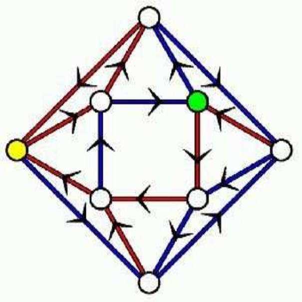
Assume that a complete synchronizing automaton is strongly connected as a digraph. Such an automaton can be reset to any state. That is, to every state q of the automaton one can assign an instruction (a reset word) w_q such that following w_q one will surely arrive at q from any initial state.



Classic Road Coloring Problem

Strongly connected synchronizing automata

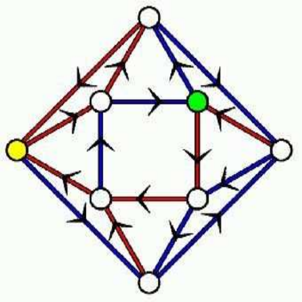
We think of such an automaton as of a transport network scheme where arrows correspond to roads and labels are treated as colors of the roads.



Then for each node N there is a sequence of colors that brings one to N from anywhere.

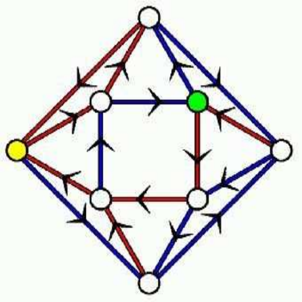
Strongly connected synchronizing automata

We think of such an automaton as of a transport network scheme where arrows correspond to roads and labels are treated as colors of the roads.



Then for each node N there is a sequence of colors that brings one to N from anywhere.

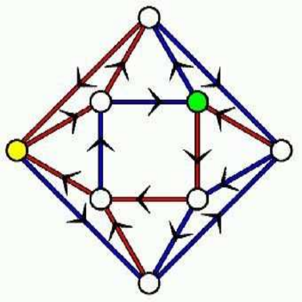
Strongly connected synchronizing automata



For the green node: blue-blue-red-blue-blue-red-blue-blue-red.

For the yellow node: blue-red-red-blue-red-red-blue-red-red.

Strongly connected synchronizing automata

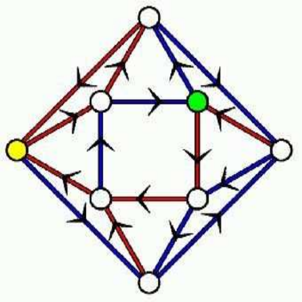


For the green node: blue-blue-red-blue-blue-red-blue-blue-red.

For the yellow node: blue-red-red-blue-red-red-blue-red-red.

SATA, Sept 1st, 2008

Strongly connected synchronizing automata



For the green node: blue-blue-red-blue-blue-red-blue-blue-red.

For the yellow node: blue-red-red-blue-red-red-blue-red-red.

SATA, Sept 1st, 2008

Classic Road Coloring Problem

Road coloring

Now suppose that we have a transport network, that is, a strongly connected digraph.

We aim to help people to orientate in it, and as we have seen, a neat solution may consist in coloring the roads such that our digraph becomes a synchronizing automaton. When is such a coloring possible?

In other words: which strongly connected digraphs may appear as underlying digraphs of complete synchronizing automata?

SATA, Sept 1st, 2008

Road coloring

Now suppose that we have a transport network, that is, a strongly connected digraph.

We aim to help people to orientate in it, and as we have seen, a neat solution may consist in coloring the roads such that our digraph becomes a synchronizing automaton. When is such a coloring possible?

In other words: which strongly connected digraphs may appear as underlying digraphs of complete synchronizing automata?

Classic Road Coloring Problem

Road coloring

Now suppose that we have a transport network, that is, a strongly connected digraph.

We aim to help people to orientate in it, and as we have seen, a neat solution may consist in coloring the roads such that our digraph becomes a synchronizing automaton. When is such a coloring possible?

In other words: which strongly connected digraphs may appear as underlying digraphs of complete synchronizing automata?

Classic Road Coloring Problem

Road coloring

Now suppose that we have a transport network, that is, a strongly connected digraph.

We aim to help people to orientate in it, and as we have seen, a neat solution may consist in coloring the roads such that our digraph becomes a synchronizing automaton. When is such a coloring possible?

In other words: which strongly connected digraphs may appear as underlying digraphs of complete synchronizing automata?

Classic Road Coloring Problem

Road coloring

An obvious necessary condition: all vertices should have the same out-degree.

A less obvious necessary condition is called aperiodicity or primitivity: the g.c.d. of lengths of all cycles should be equal to 1.

Road coloring

An obvious necessary condition: all vertices should have the same out-degree.

A less obvious necessary condition is called aperiodicity or primitivity: the g.c.d. of lengths of all cycles should be equal to 1.

Classic Road Coloring Problem

Road coloring

An obvious necessary condition: all vertices should have the same out-degree.
A less obvious necessary condition is called aperiodicity or primitivity: the g.c.d. of lengths of all cycles should be equal to 1.

SATA, Sept 1st, 2008

Road coloring

The name *Road Coloring Conjecture* was used for the claim that the two necessary conditions (constant out-degree and primitivity) are in fact sufficient.

In other words: every strongly connected primitive digraph with constant out-degree admits a synchronizing coloring.

The Road Coloring Conjecture was explicitly stated by Adler, Goodwyn and Weiss in 1977 (Equivalence of topological Markov shifts, Israel J. Math., 27, 49–63). In an implicit form it was present already in an earlier memoir by Adler and Weiss (Similarity of automorphisms of the torus, Memoirs Amer. Math. Soc., 98 (1970)) almost 40 years ago. Finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman.

Road coloring

The name *Road Coloring Conjecture* was used for the claim that the two necessary conditions (constant out-degree and primitivity) are in fact sufficient.

In other words: every strongly connected primitive digraph with constant out-degree admits a synchronizing coloring.

The Road Coloring Conjecture was explicitly stated by Adler, Goodwyn and Weiss in 1977 (Equivalence of topological Markov shifts, Israel J. Math., 27, 49–63). In an implicit form it was present already in an earlier memoir by Adler and Weiss (Similarity of automorphisms of the torus, Memoirs Amer. Math. Soc., 98 (1970)) almost 40 years ago. Finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman.

Classic Road Coloring Problem

Road coloring

The name *Road Coloring Conjecture* was used for the claim that the two necessary conditions (constant out-degree and primitivity) are in fact sufficient.

In other words: every strongly connected primitive digraph with constant out-degree admits a synchronizing coloring.

The Road Coloring Conjecture was explicitly stated by Adler, Goodwyn and Weiss in 1977 (Equivalence of topological Markov shifts, Israel J. Math., 27, 49–63). In an implicit form it was present already in an earlier memoir by Adler and Weiss (Similarity of automorphisms of the torus, Memoirs Amer. Math. Soc., 98 (1970)) almost 40 years ago. Finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman.

Classic Road Coloring Problem

Road coloring

The name *Road Coloring Conjecture* was used for the claim that the two necessary conditions (constant out-degree and primitivity) are in fact sufficient.

In other words: every strongly connected primitive digraph with constant out-degree admits a synchronizing coloring.

The Road Coloring Conjecture was explicitly stated by Adler, Goodwyn and Weiss in 1977 (Equivalence of topological Markov shifts, Israel J. Math., 27, 49–63). In an implicit form it was present already in an earlier memoir by Adler and Weiss (Similarity of automorphisms of the torus, Memoirs Amer. Math. Soc., 98 (1970)) almost 40 years ago. Finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman.

Road coloring

The name *Road Coloring Conjecture* was used for the claim that the two necessary conditions (constant out-degree and primitivity) are in fact sufficient.

In other words: every strongly connected primitive digraph with constant out-degree admits a synchronizing coloring.

The Road Coloring Conjecture was explicitly stated by Adler, Goodwyn and Weiss in 1977 (Equivalence of topological Markov shifts, Israel J. Math., 27, 49–63). In an implicit form it was present already in an earlier memoir by Adler and Weiss (Similarity of automorphisms of the torus, Memoirs Amer. Math. Soc., 98 (1970)) almost 40 years ago. Finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman.

Careful road coloring

From the viewpoint of transportation networks the constant out-degree condition does not seem to be natural. We rather want to find a synchronizing coloring for an arbitrary strongly connected digraph Γ, the number of colors being the maximal out-degree of Γ. This naturally leads to

Careful Road Coloring Problem

Under which conditions does a strongly connected digraph Γ admit a carefully synchronizing coloring?

Careful road coloring

From the viewpoint of transportation networks the constant out-degree condition does not seem to be natural. We rather want to find a synchronizing coloring for an arbitrary strongly connected digraph Γ, the number of colors being the maximal out-degree of Γ.

This naturally leads to

Careful Road Coloring Problem

Under which conditions does a strongly connected digraph Γ admit a carefully synchronizing coloring?

Careful road coloring

From the viewpoint of transportation networks the constant out-degree condition does not seem to be natural. We rather want to find a synchronizing coloring for an arbitrary strongly connected digraph Γ, the number of colors being the maximal out-degree of Γ. This naturally leads to

Careful Road Coloring Problem

Under which conditions does a strongly connected digraph Γ admit a carefully synchronizing coloring?

Careful road coloring

Primitivity is still necessary but it is not sufficient anymore. We conclude with an example (due to Ananichev) of a strongly connected primitive digraph that admits no carefully synchronizing coloring.

SATA, Sept 1st, 2008

Careful road coloring

Primitivity is still necessary but it is not sufficient anymore. We conclude with an example (due to Ananichev) of a strongly connected primitive digraph that admits no carefully synchronizing coloring.

SATA, Sept 1st, 2008

Classic Road Coloring Problem

Careful road coloring

Primitivity is still necessary but it is not sufficient anymore. We conclude with an example (due to Ananichev) of a strongly connected primitive digraph that admits no carefully synchronizing coloring.