On two Algorithmic Problems about Synchronizing Automata

Mikhail V. Berlinkov,
Institute of Mathematics and Computer Science,
Ural Federal University (Ekaterinburg, Russia),
berlm@mail.ru

DLT 2014, Ekaterinburg

Synchronizing Automata

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA – model of a discrete reactive system.

- A word v is reset for $\mathscr{A}$ if $|\delta(Q, v)| = 1$ – allows to reestablish the control under the system.
- $\mathscr{A}$ is called synchronizing if it possesses some reset word.
- Denote by $rt(\mathscr{A})$ the **minimum length** of reset words for $\mathscr{A}$ – reset threshold of $\mathscr{A}$.

Synchronizing Automata

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA – model of a discrete reactive system.

- A word v is reset for $\mathscr{A}$ if $|\delta(Q, v)| = 1$ – allows to reestablish the control under the system.
- $\mathscr{A}$ is called synchronizing if it possesses some reset word.
- Denote by $rt(\mathscr{A})$ the **minimum length** of reset words for $\mathscr{A}$ – reset threshold of $\mathscr{A}$.

Synchronizing Automata

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA – model of a discrete reactive system.

- A word v is reset for $\mathscr{A}$ if $|\delta(Q, v)| = 1$ – allows to reestablish the control under the system.
- $\mathscr{A}$ is called synchronizing if it possesses some reset word.
- Denote by $rt(\mathscr{A})$ the **minimum length** of reset words for $\mathscr{A}$ – reset threshold of $\mathscr{A}$.

Synchronizing Automata

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA – model of a discrete reactive system.

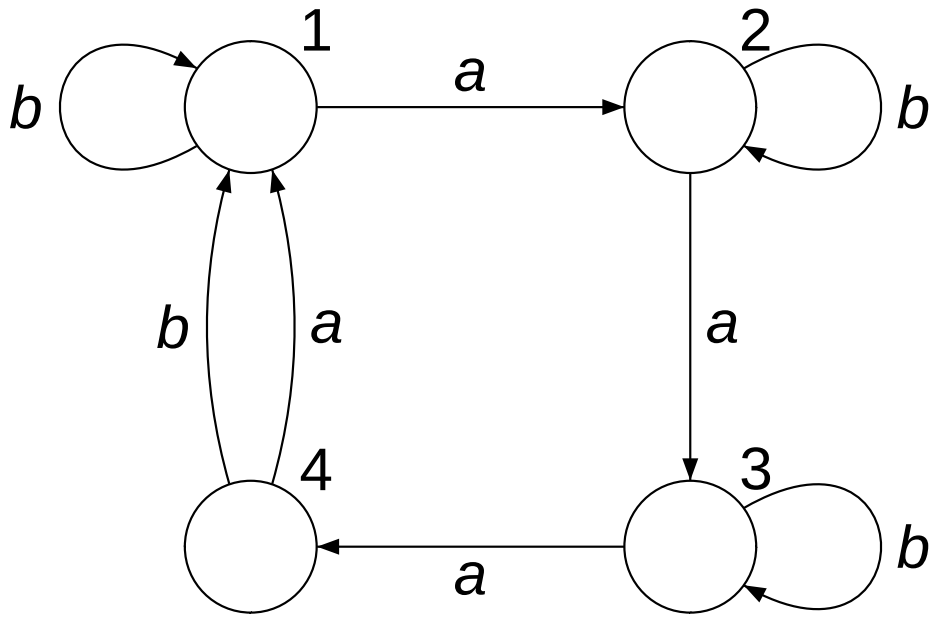
- A word v is reset for $\mathscr{A}$ if $|\delta(Q, v)| = 1$ – allows to reestablish the control under the system.
- $\mathscr{A}$ is called synchronizing if it possesses some reset word.
- Denote by $rt(\mathscr{A})$ the **minimum length** of reset words for $\mathscr{A}$ – reset threshold of $\mathscr{A}$.

Synchronizing Automata

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA – model of a discrete reactive system.

- A word v is reset for $\mathscr{A}$ if $|\delta(Q, v)| = 1$ – allows to reestablish the control under the system.
- $\mathscr{A}$ is called synchronizing if it possesses some reset word.
- Denote by $rt(\mathscr{A})$ the **minimum length** of reset words for $\mathscr{A}$ – reset threshold of $\mathscr{A}$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



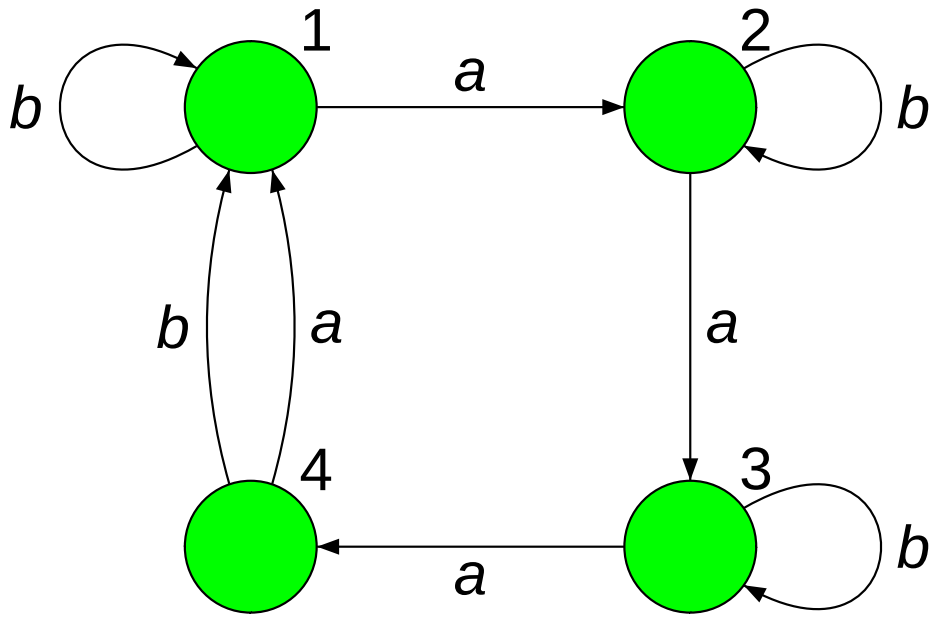
A reset word is $v = baababaaab$.

$\delta(Q, v) =$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



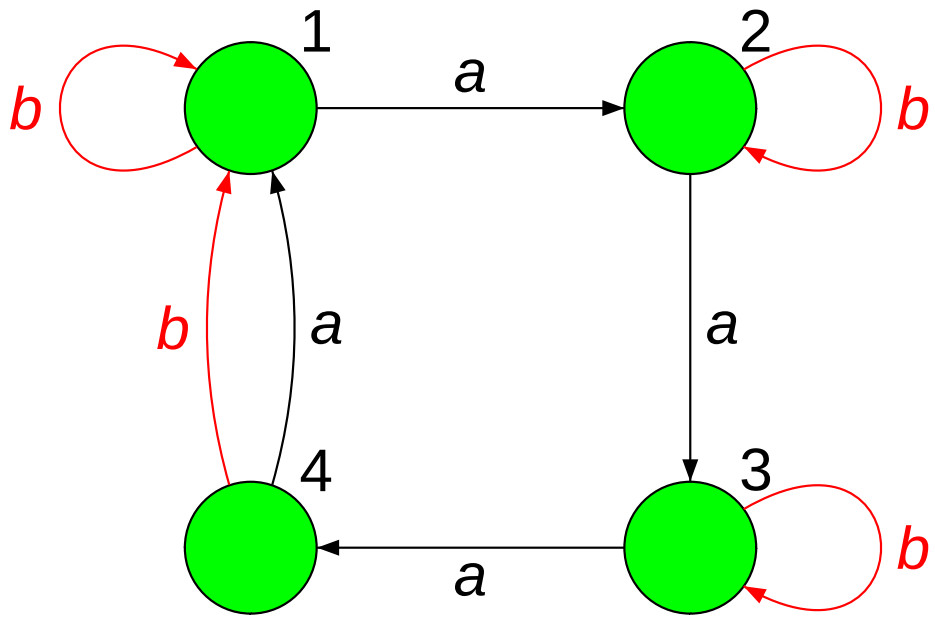
A reset word is $v = baababaaab$.

$\delta(Q, v) =$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

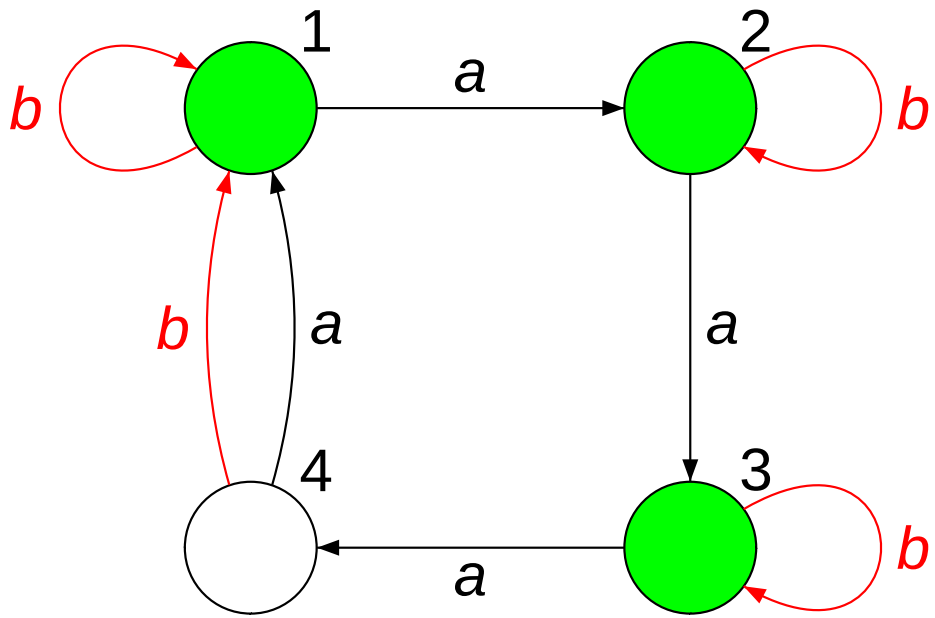
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{1, 2, 3, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



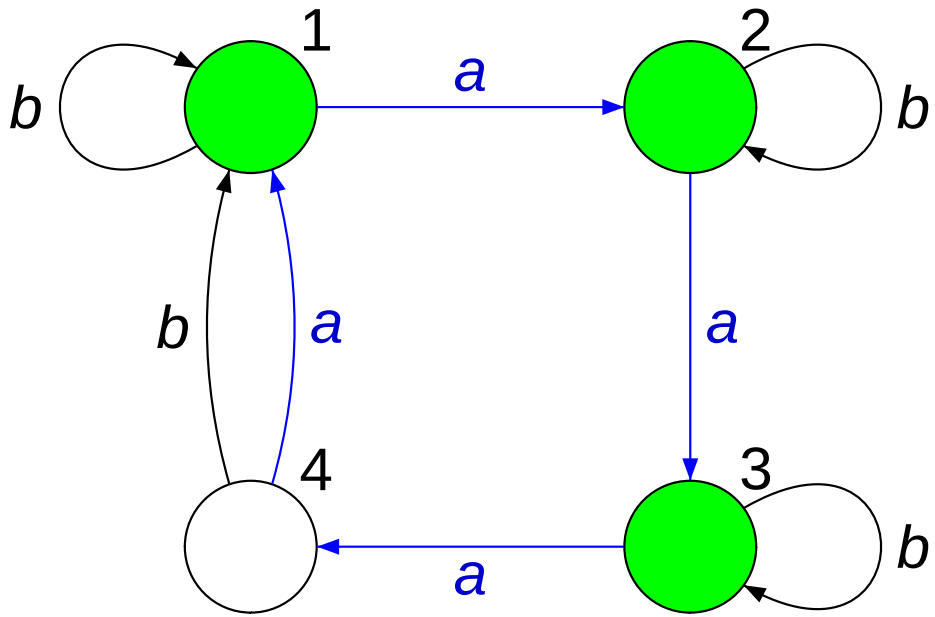
A reset word is $v = baababaaab$.

$\delta(Q, v) = \{1, 2, 3\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



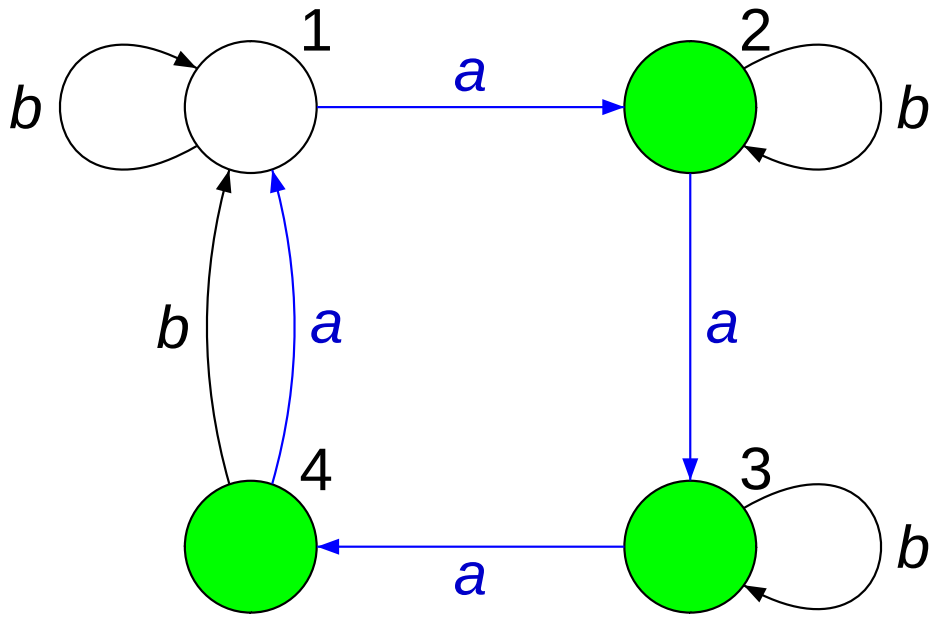
A reset word is $v = baababaaab$.

$\delta(Q, v) = \{1, 2, 3\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

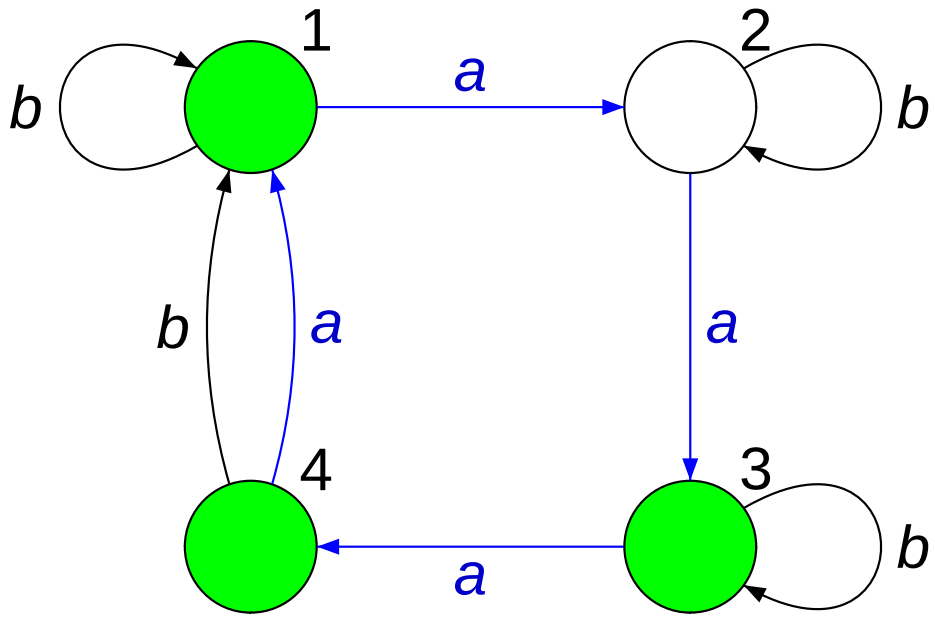
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{2, 3, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm

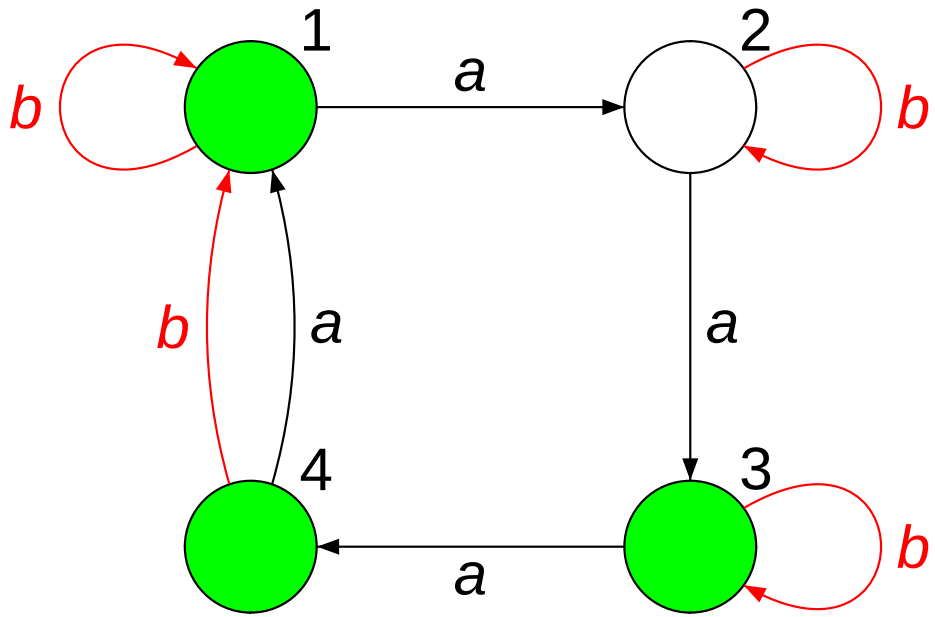


A reset word is $v = baababaaab$.
$\delta(Q, v) = \{1, 3, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

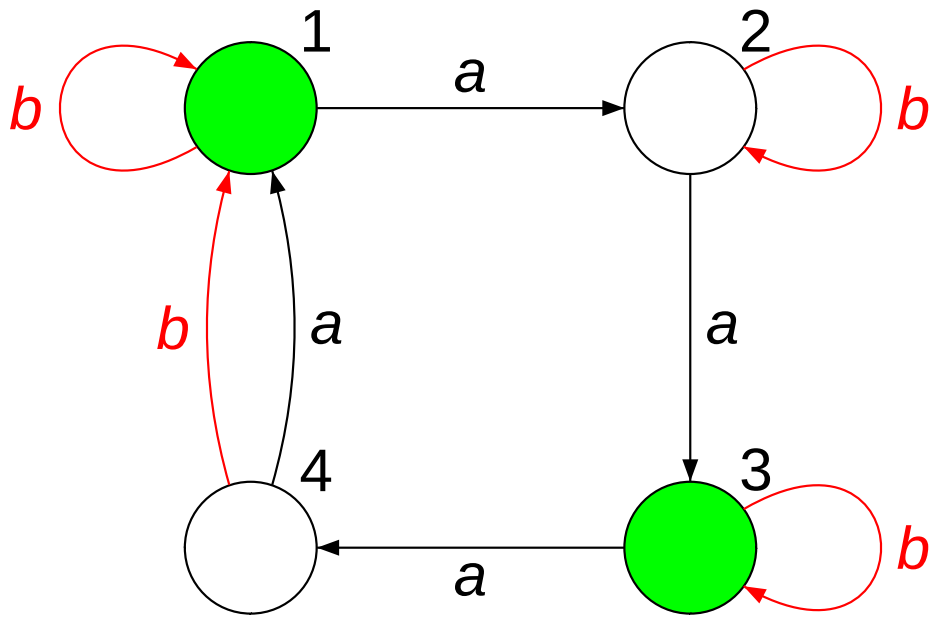
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baabbabaaab$.
$\delta(Q, v) = \{1, 3, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

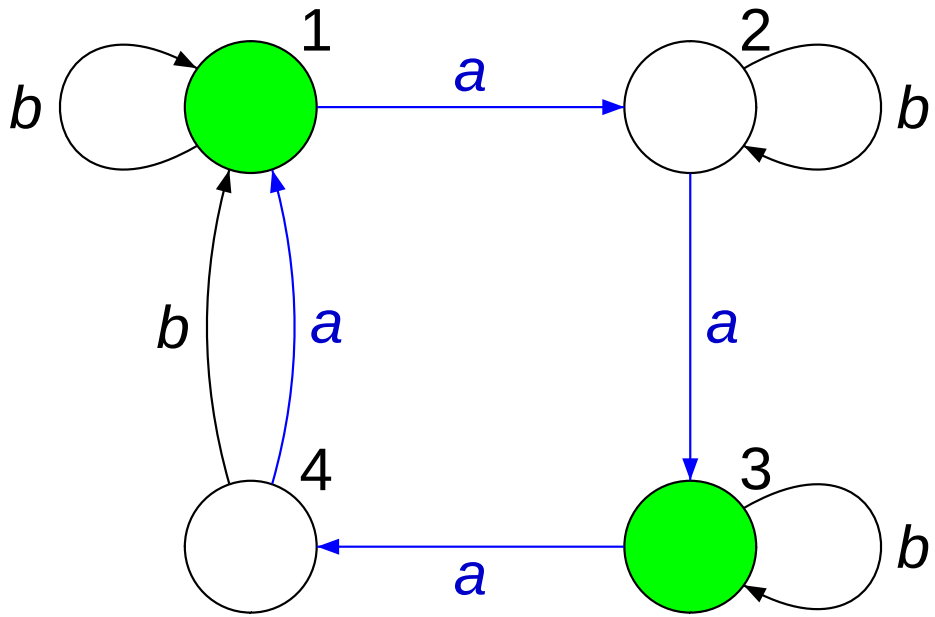
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{1, 3\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

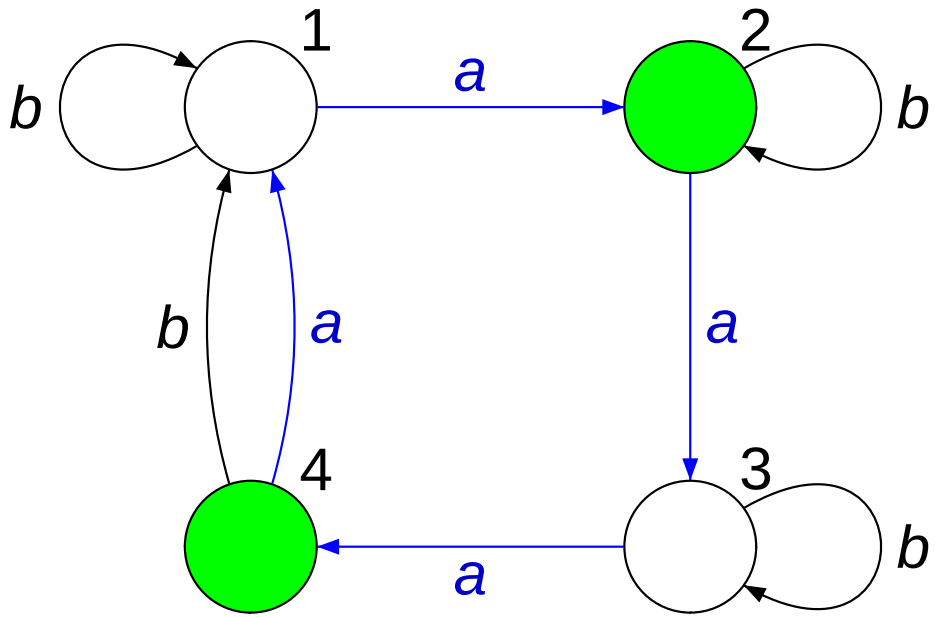
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{1, 3\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

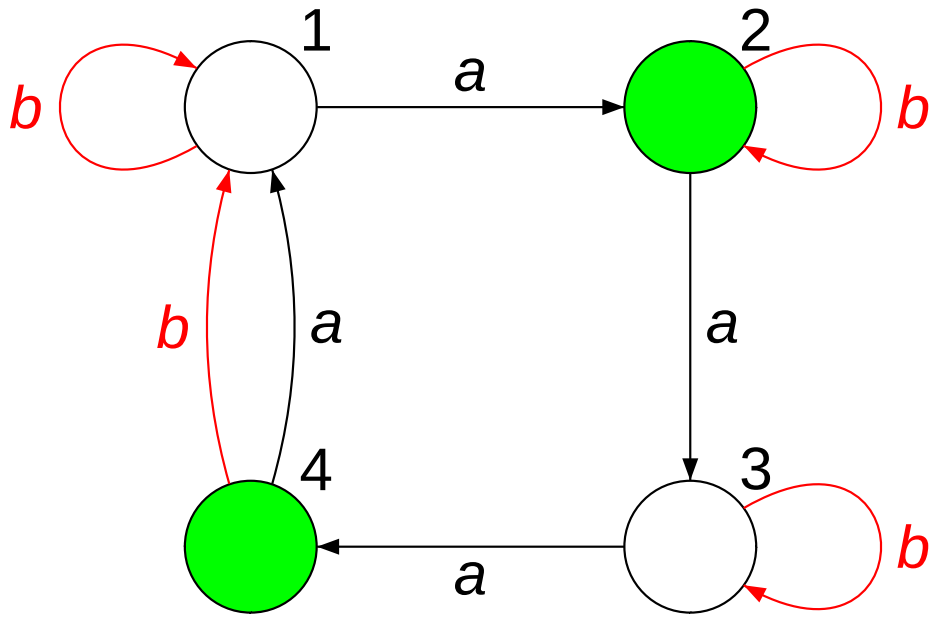
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{2, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

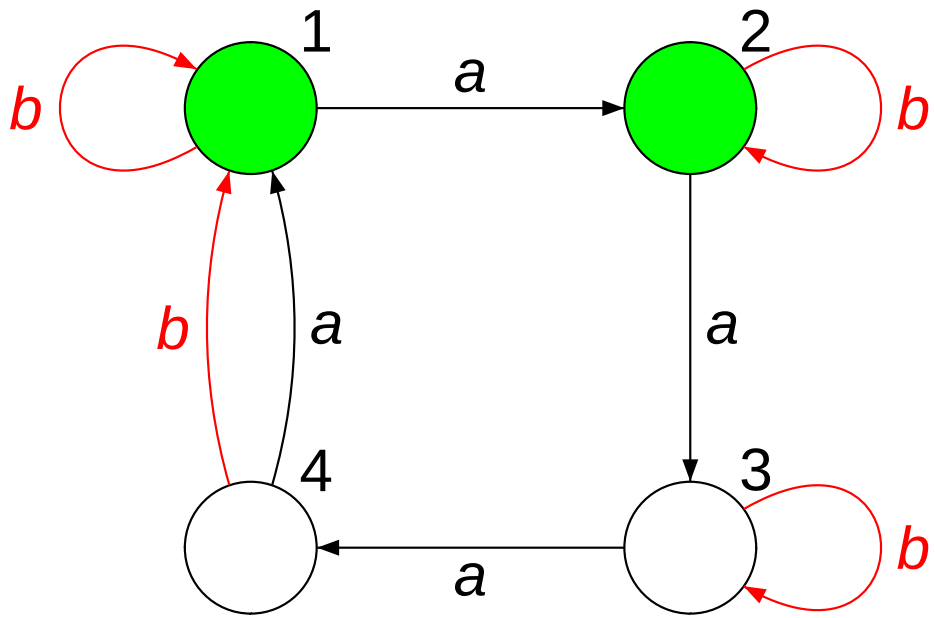
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{2, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



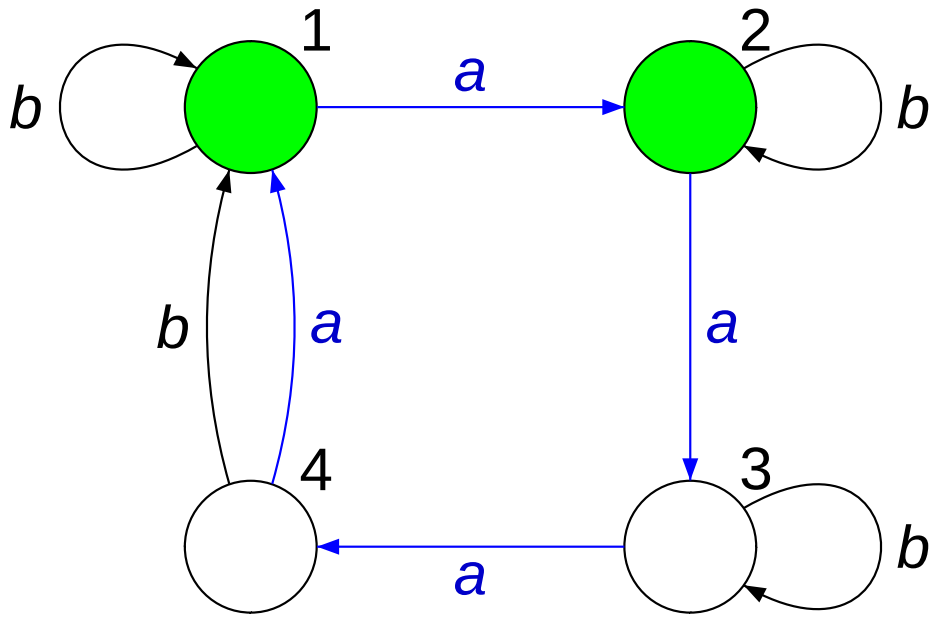
A reset word is $v = baababaaab$.

$\delta(Q, v) = \{1, 2\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

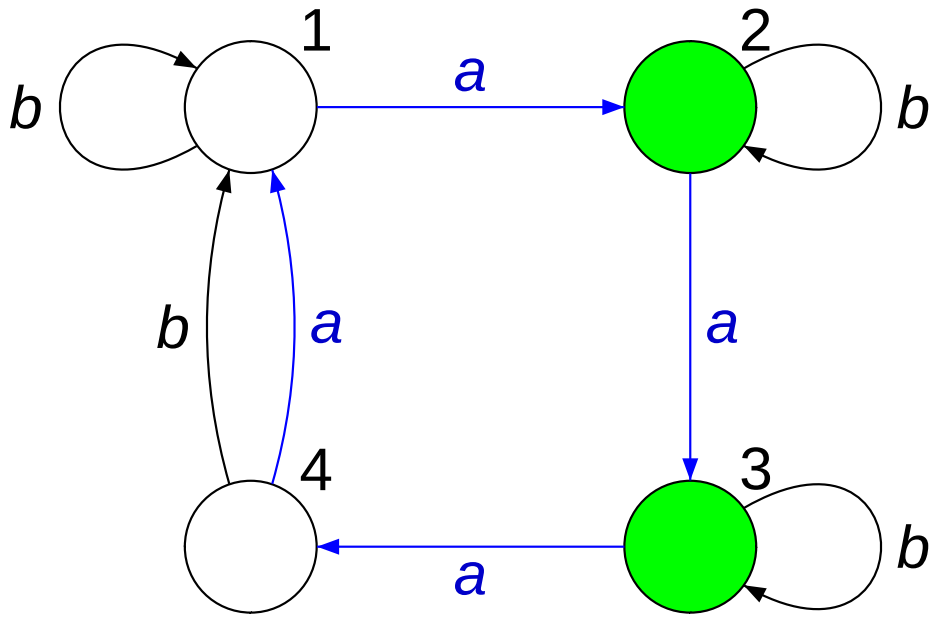
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{1, 2\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



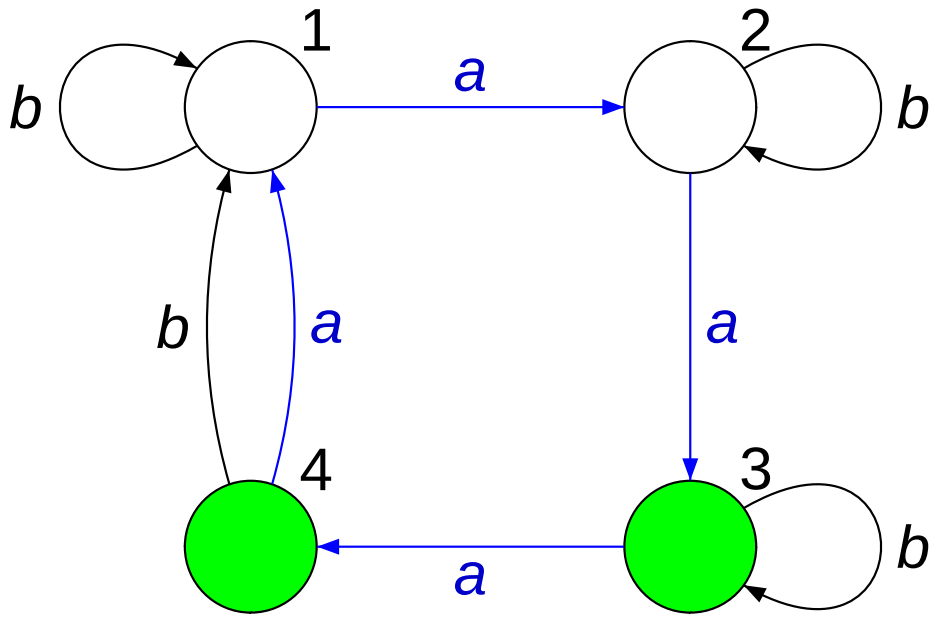
A reset word is $v = baababaaab$.

$\delta(Q, v) = \{2, 3\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

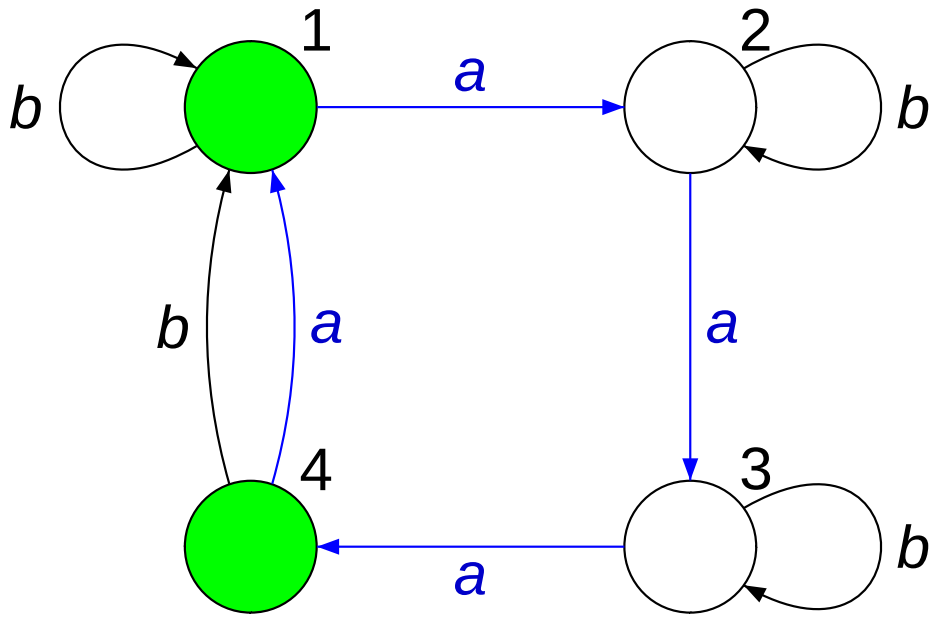
Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{3, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



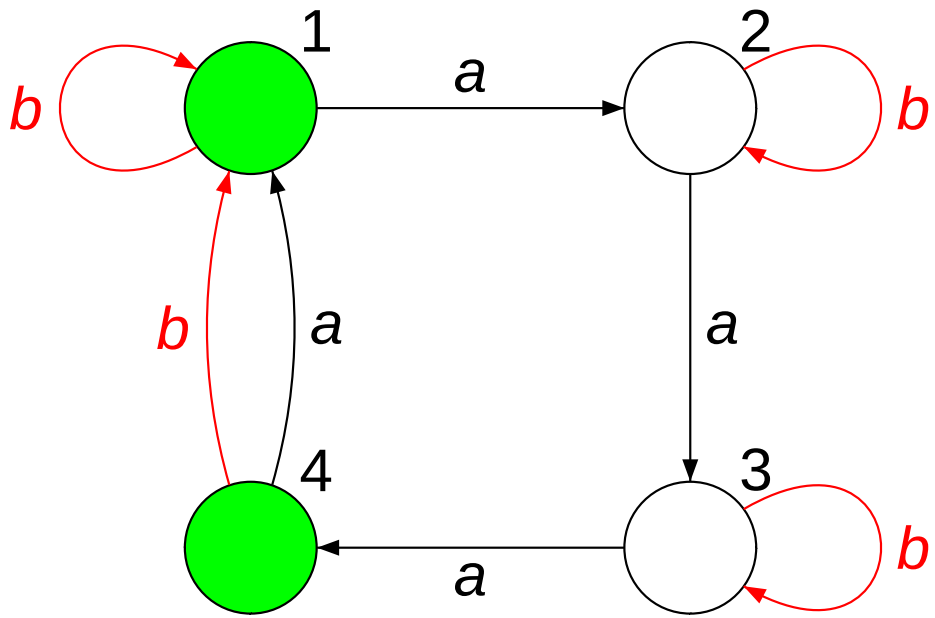
A reset word is $v = baababaaab$.

$\delta(Q, v) = \{1, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



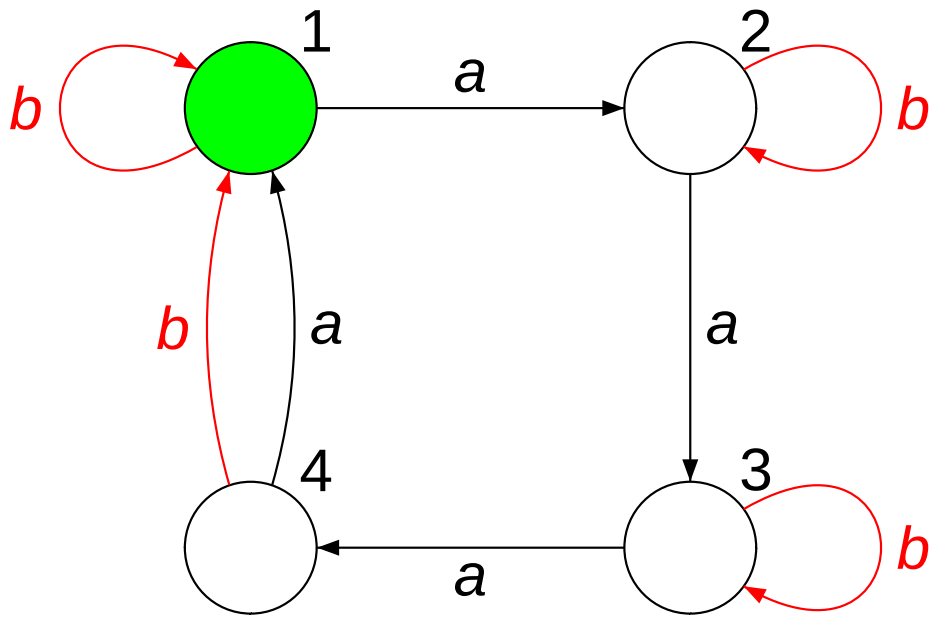
A reset word is $v = baababaaab$.

$\delta(Q, v) = \{1, 4\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm

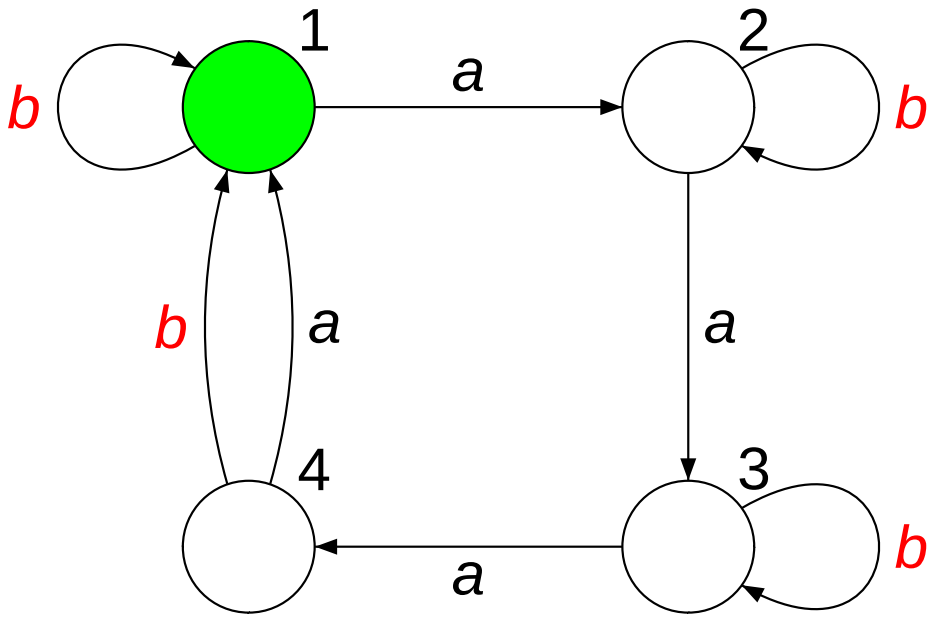


A reset word is $v = baababaaab$.
$\delta(Q, v) = \{1\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.

In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Synchronize Automaton $\mathscr{A}$ by *Greedy* algorithm



A reset word is $v = baababaaab$.
$\delta(Q, v) = \{1\}$

Since $|\delta(Q, v)| = 1$, the word v is a reset word for $\mathscr{A}$ whence $rt(\mathscr{A}) \leq |v| = 10$.
In fact the shortest reset word for $\mathscr{A}$ is ba^3ba^3b of length 9 whence $rt(\mathscr{A}) = 9 < |v|$.

Testing for Synchronization and Result 1

Result 1

Testing a given reachable partial binary automaton for synchronization is *PSPACE*-complete. There is a series of reachable partial binary automata with reset thresholds of order $2^{\Omega(n)}$.

Vojtech,2014:

- Given a strongly connected binary automaton $\mathscr{A} = \langle Q, \{a, b\}, \delta \rangle$ and a subset of states $S \subset Q$, it is *PSPACE*-complete to decide whether or not S can be synchronized in $\mathscr{A}$.
- There is a series of strongly connected binary automata $\mathscr{A}_n = \langle Q_n, \{a, b\}, \delta_n \rangle$ and corresponding subsets $S_n \subset Q_n$ such that the minimum length of synchronizing words for S_n in $\mathscr{A}_n$ has order $2^{\Omega(n)}$.

Testing for Synchronization and Result 1

Result 1

Testing a given reachable partial binary automaton for synchronization is *PSPACE*-complete. There is a series of reachable partial binary automata with reset thresholds of order $2^{\Omega(n)}$.

Vojtech,2014:

- Given a strongly connected binary automaton $\mathscr{A} = \langle Q, \{a, b\}, \delta \rangle$ and a subset of states $S \subset Q$, it is *PSPACE*-complete to decide whether or not S can be synchronized in $\mathscr{A}$.
- There is a series of strongly connected binary automata $\mathscr{A}_n = \langle Q_n, \{a, b\}, \delta_n \rangle$ and corresponding subsets $S_n \subset Q_n$ such that the minimum length of synchronizing words for S_n in $\mathscr{A}_n$ has order $2^{\Omega(n)}$.

Testing for Synchronization and Result 1

Result 1

Testing a given reachable partial binary automaton for synchronization is *PSPACE*-complete. There is a series of reachable partial binary automata with reset thresholds of order $2^{\Omega(n)}$.

Vojtech,2014:

- Given a strongly connected binary automaton $\mathscr{A} = \langle Q, \{a, b\}, \delta \rangle$ and a subset of states $S \subset Q$, it is *PSPACE*-complete to decide whether or not S can be synchronized in $\mathscr{A}$.
- There is a series of strongly connected binary automata $\mathscr{A}_n = \langle Q_n, \{a, b\}, \delta_n \rangle$ and corresponding subsets $S_n \subset Q_n$ such that the minimum length of synchronizing words for S_n in $\mathscr{A}_n$ has order $2^{\Omega(n)}$.

Complete and Strongly-Connected Cases

Suppose $\mathscr{A}$ is complete (1) or partial strongly connected (2); Then

- $\mathscr{A}$ is synchronizing if and only if each pair $\{p, q\}$ can be synchronized, i.e. $|\delta(\{p, q\}, u)| = 1$ for some word u [(1) – Černý, 1964], [(2) – Travers et al. 2012].
- There is a quadratic (in n) time algorithm which decides whether $\mathscr{A}$ is synchronizing or not and a cubic time algorithm which returns a reset word of length $O(n^3)$.

B. in ArXiv, 2013

- The probability of being synchronizable for binary random complete automata with n states equals $1 - \Theta(\frac{1}{n})$.
- Testing for synchronization can be done in linear *expected* time for (1) and $O(n^{1.5})$ expected time for (2).

Complete and Strongly-Connected Cases

Suppose $\mathscr{A}$ is complete (1) or partial strongly connected (2); Then

- $\mathscr{A}$ is synchronizing if and only if each pair $\{p, q\}$ can be synchronized, i.e. $|\delta(\{p, q\}, u)| = 1$ for some word u [(1) – Černý, 1964], [(2) – Travers et al. 2012].
- There is a quadratic (in n) time algorithm which decides whether $\mathscr{A}$ is synchronizing or not and a cubic time algorithm which returns a reset word of length $O(n^3)$.

B. in ArXiv, 2013

- The probability of being synchronizable for binary random complete automata with n states equals $1 - \Theta(\frac{1}{n})$.
- Testing for synchronization can be done in linear *expected* time for (1) and $O(n^{1.5})$ expected time for (2).

Complete and Strongly-Connected Cases

Suppose $\mathscr{A}$ is complete (1) or partial strongly connected (2); Then

- $\mathscr{A}$ is synchronizing if and only if each pair $\{p, q\}$ can be synchronized, i.e. $|\delta(\{p, q\}, u)| = 1$ for some word u [(1) – Černý, 1964], [(2) – Travers et al. 2012].
- There is a quadratic (in n) time algorithm which decides whether $\mathscr{A}$ is synchronizing or not and a cubic time algorithm which returns a reset word of length $O(n^3)$.

B. in ArXiv, 2013

- The probability of being synchronizable for binary random complete automata with n states equals $1 - \Theta(\frac{1}{n})$.
- Testing for synchronization can be done in linear *expected* time for (1) and $O(n^{1.5})$ expected time for (2).

Complete and Strongly-Connected Cases

Suppose $\mathscr{A}$ is complete (1) or partial strongly connected (2); Then

- $\mathscr{A}$ is synchronizing if and only if each pair $\{p, q\}$ can be synchronized, i.e. $|\delta(\{p, q\}, u)| = 1$ for some word u [(1) – Černý, 1964], [(2) – Travers et al. 2012].
- There is a quadratic (in n) time algorithm which decides whether $\mathscr{A}$ is synchronizing or not and a cubic time algorithm which returns a reset word of length $O(n^3)$.

B. in ArXiv, 2013

- The probability of being synchronizable for binary random complete automata with n states equals $1 - \Theta(\frac{1}{n})$.
- Testing for synchronization can be done in linear *expected* time for (1) and $O(n^{1.5})$ expected time for (2).

Complete and Strongly-Connected Cases

Suppose $\mathscr{A}$ is complete (1) or partial strongly connected (2); Then

- $\mathscr{A}$ is synchronizing if and only if each pair $\{p, q\}$ can be synchronized, i.e. $|\delta(\{p, q\}, u)| = 1$ for some word u [(1) – Černý, 1964], [(2) – Travers et al. 2012].
- There is a quadratic (in n) time algorithm which decides whether $\mathscr{A}$ is synchronizing or not and a cubic time algorithm which returns a reset word of length $O(n^3)$.

B. in ArXiv, 2013

- The probability of being synchronizable for binary random complete automata with n states equals $1 - \Theta(\frac{1}{n})$.
- Testing for synchronization can be done in linear *expected* time for (1) and $O(n^{1.5})$ expected time for (2).

Computing the Reset Threshold and Result 2

Given a k-letter n-state synchronizing automaton $\mathscr{A}$, to compute its reset threshold.

Unless $P = NP$, there are no polynomial-time algorithm for the following variations.

- exact [Rystsov, 1980; Eppstein, 1990],
- approximation within any constant factor for $k = 2$ [B, CSR 2010],
- approximation within $c \log n$ for $k \uparrow$ [Gerbush, Heeringa, 2011].

Result 2

No polynomial-time algorithm can approximate the reset threshold within $0.5c \log n$ factor for binary automata.

A reduction from Set-Cover is used for which there is $\ln(n) - \ln(\ln(n)) + \Theta(1)$ polynomial approximation [Lovász, 1975].
Is there also polynomial logarithmic approximation of reset threshold?

Computing the Reset Threshold and Result 2

Given a k-letter n-state synchronizing automaton $\mathscr{A}$, to compute its reset threshold.

Unless $P = NP$, there are no polynomial-time algorithm for the following variations.

- exact [Rystsov, 1980; Eppstein, 1990],
- approximation within any constant factor for $k = 2$ [B. CSR 2010],
- approximation within $c \log n$ for $k \uparrow$ [Gerbush, Heeringa, 2011],

Result 2

No polynomial-time algorithm can approximate the reset threshold within $0.5c \log n$ factor for binary automata.

A reduction from Set-Cover is used for which there is $\ln(n) - \ln(\ln(n)) + \Theta(1)$ polynomial approximation [Lovász, 1975].

Is there also polynomial logarithmic approximation of reset threshold?

Computing the Reset Threshold and Result 2

Given a k-letter n-state synchronizing automaton $\mathscr{A}$, to compute its reset threshold.

Unless $P = NP$, there are no polynomial-time algorithm for the following variations.

- exact [Rystsov, 1980; Eppstein, 1990],
- approximation within any constant factor for $k = 2$ [B. CSR 2010],
- approximation within $c \log n$ for $k \uparrow$ [Gerbush, Heeringa, 2011],

Result 2

No polynomial-time algorithm can approximate the reset threshold within $0.5c \log n$ factor for binary automata.

A reduction from Set-Cover is used for which there is $\ln(n) - \ln(\ln(n)) + \Theta(1)$ polynomial approximation [Lovász, 1975].
Is there also polynomial logarithmic approximation of reset threshold?

Computing the Reset Threshold and Result 2

Given a k-letter n-state synchronizing automaton $\mathscr{A}$, to compute its reset threshold.

Unless $P = NP$, there are no polynomial-time algorithm for the following variations.

- exact [Rystsov, 1980; Eppstein, 1990],
- approximation within any constant factor for $k = 2$ [B. CSR 2010],
- approximation within $c \log n$ for $k \uparrow$ [Gerbush, Heeringa, 2011],

Result 2

No polynomial-time algorithm can approximate the reset threshold within $0.5c \log n$ factor for binary automata.

A reduction from Set-Cover is used for which there is $\ln(n) - \ln(\ln(n)) + \Theta(1)$ polynomial approximation [Lovász, 1975].

Is there also polynomial logarithmic approximation of reset threshold?

Computing the Reset Threshold and Result 2

Given a k-letter n-state synchronizing automaton $\mathscr{A}$, to compute its reset threshold.

Unless $P = NP$, there are no polynomial-time algorithm for the following variations.

- exact [Rystsov, 1980; Eppstein, 1990],
- approximation within any constant factor for $k = 2$ [B. CSR 2010],
- approximation within $c \log n$ for $k \uparrow$ [Gerbush, Heeringa, 2011],

Result 2

No polynomial-time algorithm can approximate the reset threshold within $0.5c \log n$ factor for binary automata.

A reduction from Set-Cover is used for which there is $\ln(n) - \ln(\ln(n)) + \Theta(1)$ polynomial approximation [Lovász, 1975].
Is there also polynomial logarithmic approximation of reset threshold?

Computing the Reset Threshold and Result 2

Given a k-letter n-state synchronizing automaton $\mathscr{A}$, to compute its reset threshold.

Unless $P = NP$, there are no polynomial-time algorithm for the following variations.

- exact [Rystsov, 1980; Eppstein, 1990],
- approximation within any constant factor for $k = 2$ [B. CSR 2010],
- approximation within $c \log n$ for $k \uparrow$ [Gerbush, Heeringa, 2011],

Result 2

No polynomial-time algorithm can approximate the reset threshold within $0.5c \log n$ factor for binary automata.

A reduction from Set-Cover is used for which there is $\ln(n) - \ln(\ln(n)) + \Theta(1)$ polynomial approximation [Lovász, 1975].
Is there also polynomial logarithmic approximation of reset threshold?

Computing the Reset Threshold and Result 2

Given a k-letter n-state synchronizing automaton $\mathscr{A}$, to compute its reset threshold.

Unless $P = NP$, there are no polynomial-time algorithm for the following variations.

- exact [Rystsov, 1980; Eppstein, 1990],
- approximation within any constant factor for $k = 2$ [B. CSR 2010],
- approximation within $c \log n$ for $k \uparrow$ [Gerbush, Heeringa, 2011],

Result 2

No polynomial-time algorithm can approximate the reset threshold within $0.5c \log n$ factor for binary automata.

A reduction from Set-Cover is used for which there is $\ln(n) - \ln(\ln(n)) + \Theta(1)$ polynomial approximation [Lovász, 1975].

Is there also polynomial logarithmic approximation of reset threshold?

Computing the Reset Threshold and Result 2

Given a k-letter n-state synchronizing automaton $\mathscr{A}$, to compute its reset threshold.

Unless $P = NP$, there are no polynomial-time algorithm for the following variations.

- exact [Rystsov, 1980; Eppstein, 1990],
- approximation within any constant factor for $k = 2$ [B. CSR 2010],
- approximation within $c \log n$ for $k \uparrow$ [Gerbush, Heeringa, 2011],

Result 2

No polynomial-time algorithm can approximate the reset threshold within $0.5c \log n$ factor for binary automata.

A reduction from Set-Cover is used for which there is $\ln(n) - \ln(\ln(n)) + \Theta(1)$ polynomial approximation [Lovász, 1975].
Is there also polynomial logarithmic approximation of reset threshold?

Thank you! Any questions?