Synchronizing Finite Automata

II-III. The Road Coloring Problem

Mikhail Volkov

Ural Federal University, Ekaterinburg, Russia

Recap

Deterministic finite automata (DFA): $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

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

$\mathscr{A}$ 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 $q, q' \in Q$.

$|Q\,.\,w| = 1$. Here $Q\,.\,v = \{\delta(q, v) \mid q \in Q\}$.

Any w with this property is a reset word for $\mathscr{A}$.

Recap

Deterministic finite automata (DFA): $\mathscr{A} = \langle Q, \Sigma, \delta\rangle$.

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

$\mathscr{A}$ 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 $q, q' \in Q$.

$|Q\,.\,w| = 1$. Here $Q\,.\,v = \{\delta(q, v) \mid q \in Q\}$.

Any w with this property is a reset word for $\mathscr{A}$.

Recap

Deterministic finite automata (DFA): $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

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

$\mathscr{A}$ 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 $q, q' \in Q$.

$|Q\,.\,w| = 1$. Here $Q\,.\,v = \{\delta(q, v) \mid q \in Q\}$.

Any w with this property is a reset word for $\mathscr{A}$.

Recap

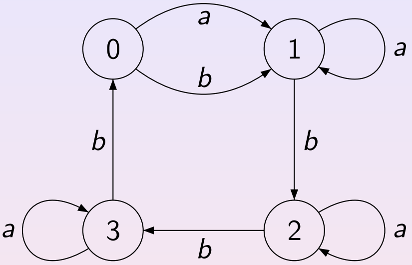
Deterministic finite automata (DFA): $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$.

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

$\mathscr{A}$ 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 $q, q' \in Q$.
$|Q\,.\,w| = 1$. Here $Q\,.\,v = \{\delta(q, v) \mid q \in Q\}$.

Any w with this property is a reset word for $\mathscr{A}$.

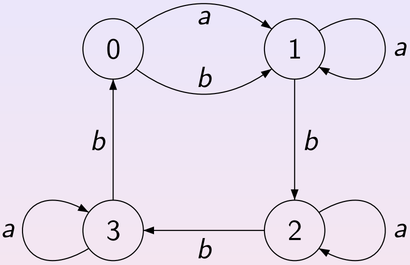
Example



A reset word is *abbbabbba*. In fact, it is the shortest reset word for this automaton.

The Černý Conjecture: each synchronizing automaton with n states has a reset word of length $(n-1)^2$.

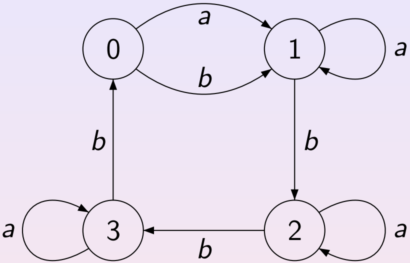
Example



A reset word is *abbbabbba*. In fact, it is the shortest reset word for this automaton.

The Černý Conjecture: each synchronizing automaton with n states has a reset word of length $(n-1)^2$.

Example



A reset word is *abbbabbba*. In fact, it is the shortest reset word for this automaton.

The Černý Conjecture: each synchronizing automaton with n states has a reset word of length $(n-1)^2$.

Studying synchronizing automata, it is natural to restrict to the strongly connected case. For instance, it suffices to prove the Černý conjecture for this case, the general case would be an easy consequence.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a synchronizing automaton with n states. Consider the set S of all states to which $\mathscr{A}$ can be synchronized and let $m = |S|$. If $q \in S$, then there exists a reset word $w \in \Sigma^*$ such that $Q.w = \{q\}$. For each $a \in \Sigma$, we have $Q.wa = \{q \,.\, a\}$ whence wa also is a reset word and $\delta(q, a) \in S$. Thus, restricting the function δ to $S \times \Sigma$, we get a subautomaton $\mathscr{S}$ with the state set S. Obviously, $\mathscr{S}$ is synchronizing and strongly connected.

Studying synchronizing automata, it is natural to restrict to the strongly connected case. For instance, it suffices to prove the Černý conjecture for this case, the general case would be an easy consequence.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a synchronizing automaton with n states. Consider the set S of all states to which $\mathscr{A}$ can be synchronized and let $m = |S|$. If $q \in S$, then there exists a reset word $w \in \Sigma^*$ such that $Q.w = \{q\}$. For each $a \in \Sigma$, we have $Q.wa = \{q \,.\, a\}$ whence wa also is a reset word and $\delta(q, a) \in S$. Thus, restricting the function δ to $S \times \Sigma$, we get a subautomaton $\mathscr{S}$ with the state set S. Obviously, $\mathscr{S}$ is synchronizing and strongly connected.

Studying synchronizing automata, it is natural to restrict to the strongly connected case. For instance, it suffices to prove the Černý conjecture for this case, the general case would be an easy consequence.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a synchronizing automaton with n states. Consider the set S of all states to which $\mathscr{A}$ can be synchronized and let $m = |S|$. If $q \in S$, then there exists a reset word $w \in \Sigma^*$ such that $Q.w = \{q\}$. For each $a \in \Sigma$, we have $Q.wa = \{q\,.\,a\}$ whence wa also is a reset word and $\delta(q, a) \in S$. Thus, restricting the function δ to $S \times \Sigma$, we get a subautomaton $\mathscr{S}$ with the state set S. Obviously, $\mathscr{S}$ is synchronizing and strongly connected.

Studying synchronizing automata, it is natural to restrict to the strongly connected case. For instance, it suffices to prove the Černý conjecture for this case, the general case would be an easy consequence.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a synchronizing automaton with n states. Consider the set S of all states to which $\mathscr{A}$ can be synchronized and let $m = |S|$. If $q \in S$, then there exists a reset word $w \in \Sigma^*$ such that $Q.w = \{q\}$. For each $a \in \Sigma$, we have $Q.wa = \{q \,.\, a\}$ whence wa also is a reset word and $\delta(q, a) \in S$. Thus, restricting the function δ to $S \times \Sigma$, we get a subautomaton $\mathscr{S}$ with the state set S. Obviously, $\mathscr{S}$ is synchronizing and strongly connected.

Studying synchronizing automata, it is natural to restrict to the strongly connected case. For instance, it suffices to prove the Černý conjecture for this case, the general case would be an easy consequence.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a synchronizing automaton with n states. Consider the set S of all states to which $\mathscr{A}$ can be synchronized and let $m = |S|$. If $q \in S$, then there exists a reset word $w \in \Sigma^*$ such that $Q.w = \{q\}$. For each $a \in \Sigma$, we have $Q.wa = \{q\,.\,a\}$ whence wa also is a reset word and $\delta(q, a) \in S$. Thus, restricting the function δ to $S \times \Sigma$, we get a subautomaton $\mathscr{S}$ with the state set S. Obviously, $\mathscr{S}$ is synchronizing and strongly connected.

If the Černý conjecture holds true for strongly connected synchronizing automata, $\mathscr{S}$ has a reset word v of length $(m-1)^2$.

Now consider the partition π of Q into $n-m+1$ classes one of which is S and all others are singletons. Then π is a congruence of the automaton $\mathscr{A}$.

We recall the notion of a congruence and the related notion of the quotient automaton w.r.t. a congruence in the next slide. They will be essentially used in this lecture!

If the Černý conjecture holds true for strongly connected synchronizing automata, $\mathscr{S}$ has a reset word v of length $(m-1)^2$.

Now consider the partition π of Q into $n-m+1$ classes one of which is S and all others are singletons. Then π is a congruence of the automaton $\mathscr{A}$.

We recall the notion of a congruence and the related notion of the quotient automaton w.r.t. a congruence in the next slide. They will be essentially used in this lecture!

If the Černý conjecture holds true for strongly connected synchronizing automata, $\mathscr{S}$ has a reset word v of length $(m-1)^2$.

Now consider the partition π of Q into $n-m+1$ classes one of which is S and all others are singletons. Then π is a congruence of the automaton $\mathscr{A}$.

We recall the notion of a congruence and the related notion of the quotient automaton w.r.t. a congruence in the next slide. They will be essentially used in this lecture!

Congruences and Quotient Automata

An equivalence π on the state set Q of a DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is called a congruence if $(p, q) \in \pi$ implies $\big(\delta(p, a), \delta(q, a)\big) \in \pi$ for all $p, q \in Q$ and all $a \in \Sigma$. For π being a congruence, $[q]_\pi$ is the π-class containing the state q.

The *quotient* $\mathscr{A}/\pi$ is the DFA $\langle Q/\pi, \Sigma, \delta_\pi \rangle$ where $Q/\pi = \{[q]_\pi \mid q \in Q\}$ and the function δ_π is defined by the rule $\delta_\pi([q]_\pi, a) = [\delta(q, a)]_\pi$.

Congruences and Quotient Automata

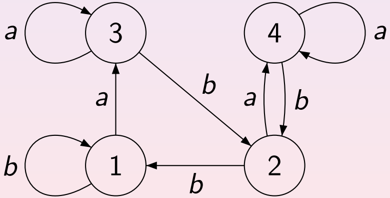
An equivalence π on the state set Q of a DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is called a congruence if $(p, q) \in \pi$ implies $\big(\delta(p, a), \delta(q, a)\big) \in \pi$ for all $p, q \in Q$ and all $a \in \Sigma$. For π being a congruence, $[q]_\pi$ is the π-class containing the state q.

The *quotient* $\mathscr{A}/\pi$ is the DFA $\langle Q/\pi, \Sigma, \delta_\pi \rangle$ where $Q/\pi = \{[q]_\pi \mid q \in Q\}$ and the function δ_π is defined by the rule $\delta_\pi([q]_\pi, a) = [\delta(q, a)]_\pi$.

Congruences and Quotient Automata

An equivalence π on the state set Q of a DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is called a congruence if $(p, q) \in \pi$ implies $\big(\delta(p, a), \delta(q, a)\big) \in \pi$ for all $p, q \in Q$ and all $a \in \Sigma$. For π being a congruence, $[q]_\pi$ is the π-class containing the state q.

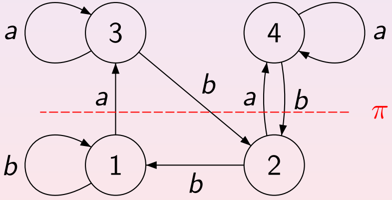
The *quotient* $\mathscr{A}/\pi$ is the DFA $\langle Q/\pi, \Sigma, \delta_\pi \rangle$ where $Q/\pi = \{[q]_\pi \mid q \in Q\}$ and the function δ_π is defined by the rule $\delta_\pi([q]_\pi, a) = [\delta(q, a)]_\pi$.



Congruences and Quotient Automata

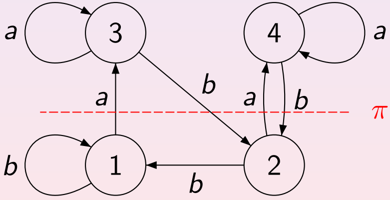
An equivalence π on the state set Q of a DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is called a congruence if $(p, q) \in \pi$ implies $\big(\delta(p, a), \delta(q, a)\big) \in \pi$ for all $p, q \in Q$ and all $a \in \Sigma$. For π being a congruence, $[q]_\pi$ is the π-class containing the state q.

The *quotient* $\mathscr{A}/\pi$ is the DFA $\langle Q/\pi, \Sigma, \delta_\pi \rangle$ where $Q/\pi = \{[q]_\pi \mid q \in Q\}$ and the function δ_π is defined by the rule $\delta_\pi([q]_\pi, a) = [\delta(q, a)]_\pi$.



Congruences and Quotient Automata

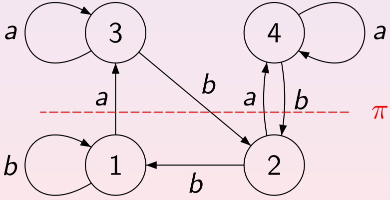
An equivalence π on the state set Q of a DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is called a congruence if $(p, q) \in \pi$ implies $\big(\delta(p, a), \delta(q, a)\big) \in \pi$ for all $p, q \in Q$ and all $a \in \Sigma$. For π being a congruence, $[q]_\pi$ is the π-class containing the state q.
The *quotient* $\mathscr{A}/\pi$ is the DFA $\langle Q/\pi, \Sigma, \delta_\pi \rangle$ where $Q/\pi = \{[q]_\pi \mid q \in Q\}$ and the function δ_π is defined by the rule $\delta_\pi([q]_\pi, a) = [\delta(q, a)]_\pi$.

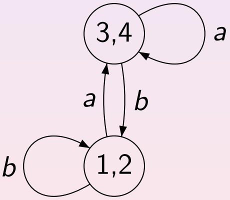


Congruences and Quotient Automata

An equivalence π on the state set Q of a DFA $\mathscr{A} = \langle Q, \Sigma, \delta\rangle$ is called a congruence if $(p, q) \in \pi$ implies $\big(\delta(p, a), \delta(q, a)\big) \in \pi$ for all $p, q \in Q$ and all $a \in \Sigma$. For π being a congruence, $[q]_\pi$ is the π-class containing the state q.

The *quotient* $\mathscr{A}/\pi$ is the DFA $\langle Q/\pi, \Sigma, \delta_\pi\rangle$ where $Q/\pi = \{[q]_\pi \mid q \in Q\}$ and the function δ_π is defined by the rule $\delta_\pi([q]_\pi, a) = [\delta(q, a)]_\pi$.





Return to our reasoning: let π be the partition of Q into $n-m+1$ classes one of which is S and all others are singletons. Then π is a congruence of $\mathscr{A}$.

Clearly, the quotient $\mathscr{A}/\pi$ is synchronizing and has S as a unique sink.

Strongly Connected Digraphs

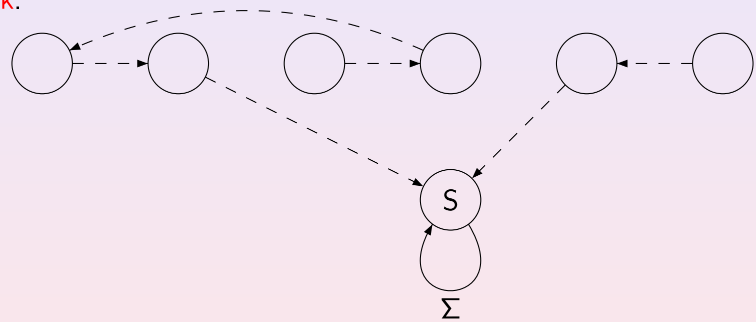
Return to our reasoning: let π be the partition of Q into $n-m+1$ classes one of which is S and all others are singletons. Then π is a congruence of $\mathscr{A}$.

Clearly, the quotient $\mathscr{A}/\pi$ is synchronizing and has S as a unique sink.

Strongly Connected Digraphs

Return to our reasoning: let π be the partition of Q into $n - m + 1$ classes one of which is S and all others are singletons. Then π is a congruence of $\mathscr{A}$.

Clearly, the quotient $\mathscr{A}/\pi$ is synchronizing and has S as a unique sink.



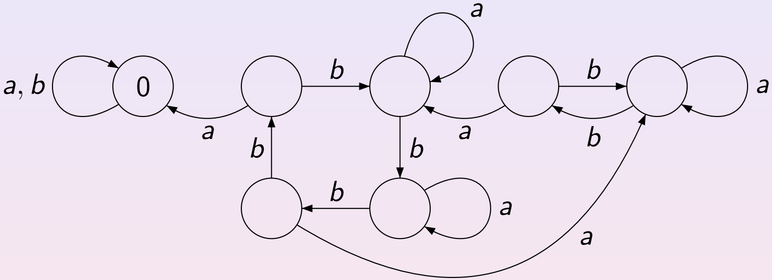
Automata with a Unique Sink

If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.

The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins $(k-1 \geq t \geq 1)$ is at most $k-t$. The total length is $\leq 1+2+\cdots+(k-1) = \frac{k(k-1)}{2}$.

Automata with a Unique Sink

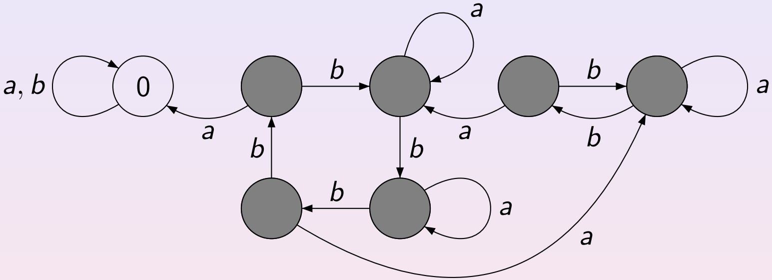
If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.



The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins $(k-1 \geq t \geq 1)$ is at most $k-t$. The total length is $\leq 1+2+\cdots+(k-1) = \frac{k(k-1)}{2}$.

Automata with a Unique Sink

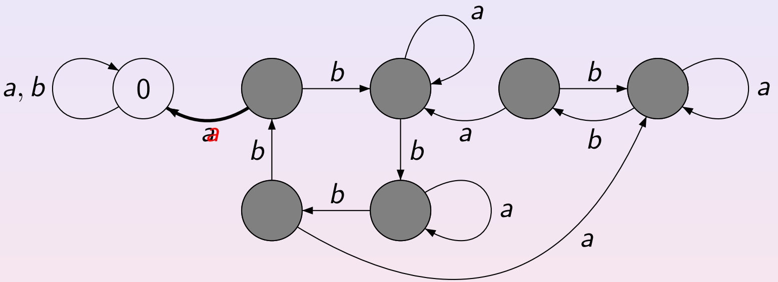
If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.



The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins ($k-1 \geq t \geq 1$) is at most $k-t$. The total length is $\leq 1+2+\cdots+(k-1) = \frac{k(k-1)}{2}$.

Automata with a Unique Sink

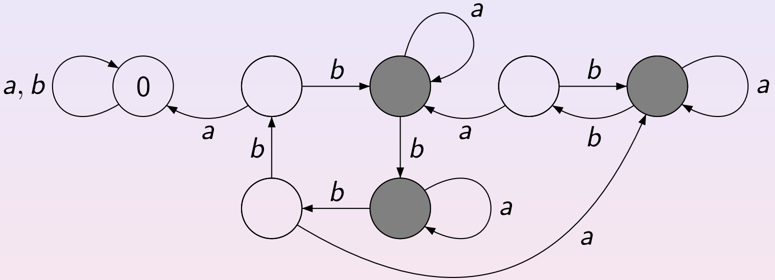
If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.



The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins ($k-1 \geq t \geq 1$) is at most $k-t$. The total length is $\leq 1+2+\cdots+(k-1)=\frac{k(k-1)}{2}$.

Automata with a Unique Sink

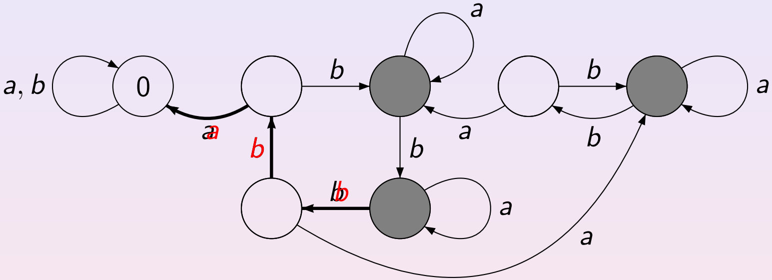
If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.



The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins $(k-1 \geq t \geq 1)$ is at most $k-t$. The total length is $\leq 1+2+\cdots+(k-1) = \frac{k(k-1)}{2}$.

Automata with a Unique Sink

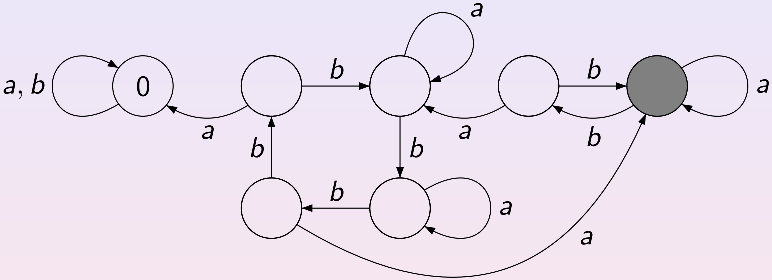
If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.



The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins ($k-1 \geq t \geq 1$) is at most $k-t$. The total length is $\leq 1 + 2 + \cdots + (k-1) = \frac{k(k-1)}{2}$.

Automata with a Unique Sink

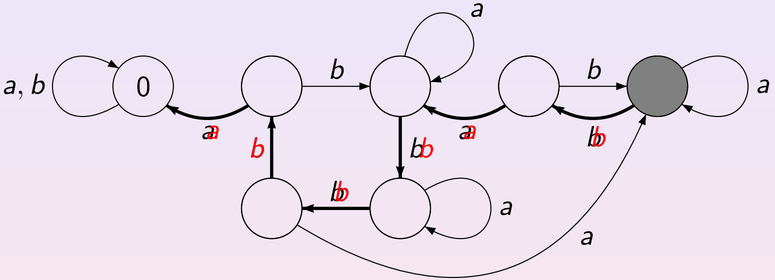
If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.



The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins $(k-1 \geq t \geq 1)$ is at most $k-t$. The total length is $\leq 1+2+\cdots+(k-1) = \frac{k(k-1)}{2}$.

Automata with a Unique Sink

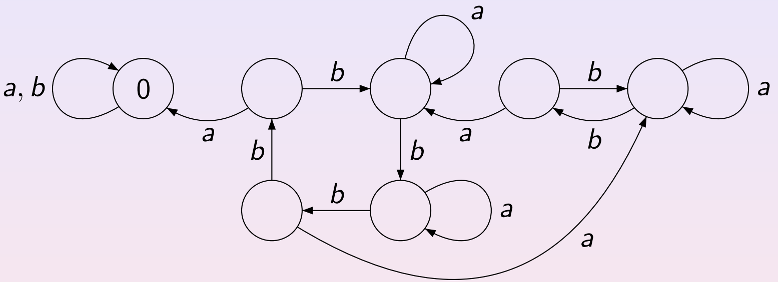
If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.



The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins $(k-1 \geq t \geq 1)$ is at most $k-t$. The total length is $\leq 1+2+\cdots+(k-1) = \frac{k(k-1)}{2}$.

Automata with a Unique Sink

If a synchronizing automaton with k states has a unique sink, then it has a reset word of length $\leq \frac{k(k-1)}{2}$.



The algorithm makes at most $k-1$ steps and the length of the segment added in the step when t states still hold coins $(k-1 \geq t \geq 1)$ is at most $k-t$. The total length is $\leq 1+2+\cdots+(k-1) = \frac{k(k-1)}{2}$.

Strongly Connected Digraphs

Return to our reasoning: the quotient $\mathscr{A}/\pi$ is synchronizing with a unique sink and has $n-m+1$ states. Hence, $\mathscr{A}/\pi$ has a reset word u of length $\frac{(n-m+1)(n-m)}{2}$. Then $Q\,.\,u \subseteq S$.
Recall that we have assumed that the automaton $\mathscr{S}$ has a reset word v of length $(m-1)^2$. Then $S.v$ is a singleton, whence also $Q.uv \subseteq S.v$ is a singleton. Thus, uv is a reset word for $\mathscr{A}$, and the length of this word does not exceed

$$\frac{(n-m+1)(n-m)}{2} + (m-1)^2 \leq (n-1)^2.$$

Strongly Connected Digraphs

Return to our reasoning: the quotient $\mathscr{A}/\pi$ is synchronizing with a unique sink and has $n-m+1$ states. Hence, $\mathscr{A}/\pi$ has a reset word u of length $\frac{(n-m+1)(n-m)}{2}$. Then $Q\,.\,u \subseteq S$.

Recall that we have assumed that the automaton $\mathscr{S}$ has a reset word v of length $(m-1)^2$. Then $S.v$ is a singleton, whence also $Q.uv \subseteq S.v$ is a singleton. Thus, uv is a reset word for $\mathscr{A}$, and the length of this word does not exceed

$$\frac{(n-m+1)(n-m)}{2} + (m-1)^2 \leq (n-1)^2.$$

Return to our reasoning: the quotient $\mathscr{A}/\pi$ is synchronizing with a unique sink and has $n-m+1$ states. Hence, $\mathscr{A}/\pi$ has a reset word u of length $\frac{(n-m+1)(n-m)}{2}$. Then $Q\,.\,u \subseteq S$.

Recall that we have assumed that the automaton $\mathscr{S}$ has a reset word v of length $(m-1)^2$. Then $S.v$ is a singleton, whence also $Q.uv \subseteq S.v$ is a singleton. Thus, uv is a reset word for $\mathscr{A}$, and the length of this word does not exceed

$$\frac{(n-m+1)(n-m)}{2} + (m-1)^2 \leq (n-1)^2.$$

Return to our reasoning: the quotient $\mathscr{A}/\pi$ is synchronizing with a unique sink and has $n-m+1$ states. Hence, $\mathscr{A}/\pi$ has a reset word u of length $\frac{(n-m+1)(n-m)}{2}$. Then $Q\,.\,u \subseteq S$.
Recall that we have assumed that the automaton $\mathscr{S}$ has a reset word v of length $(m-1)^2$. Then $S.v$ is a singleton, whence also $Q.uv \subseteq S.v$ is a singleton. Thus, uv is a reset word for $\mathscr{A}$, and the length of this word does not exceed

$$\frac{(n-m+1)(n-m)}{2} + (m-1)^2 \leq (n-1)^2.$$

Return to our reasoning: the quotient $\mathscr{A}/\pi$ is synchronizing with a unique sink and has $n-m+1$ states. Hence, $\mathscr{A}/\pi$ has a reset word u of length $\frac{(n-m+1)(n-m)}{2}$. Then $Q\,.\,u \subseteq S$.
Recall that we have assumed that the automaton $\mathscr{S}$ has a reset word v of length $(m-1)^2$. Then $S.v$ is a singleton, whence also $Q.uv \subseteq S.v$ is a singleton. Thus, uv is a reset word for $\mathscr{A}$, and the length of this word does not exceed

$$\frac{(n-m+1)(n-m)}{2} + (m-1)^2 \leq (n-1)^2.$$

Return to our reasoning: the quotient $\mathscr{A}/\pi$ is synchronizing with a unique sink and has $n-m+1$ states. Hence, $\mathscr{A}/\pi$ has a reset word u of length $\frac{(n-m+1)(n-m)}{2}$. Then $Q\,.\,u \subseteq S$.
Recall that we have assumed that the automaton $\mathscr{S}$ has a reset word v of length $(m-1)^2$. Then $S.v$ is a singleton, whence also $Q.uv \subseteq S.v$ is a singleton. Thus, uv is a reset word for $\mathscr{A}$, and the length of this word does not exceed

$$\frac{(n-m+1)(n-m)}{2} + (m-1)^2 \leq (n-1)^2.$$

Strongly Connected Digraphs

Thus, we assume that our synchronizing automata are strongly connected as digraphs.

Observe that 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.

$ab^3ab^3ab^3$ ab^3ab^3a

$ab^3ab^3ab^2$ ab^3ab^3ab

Thus, we assume that our synchronizing automata are strongly connected as digraphs.

Observe that 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.

$ab^3ab^3ab^3$ ab^3ab^3a

$ab^3ab^3ab^2$ ab^3ab^3ab

Strongly Connected Digraphs

Thus, we assume that our synchronizing automata are strongly connected as digraphs.

Observe that 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.

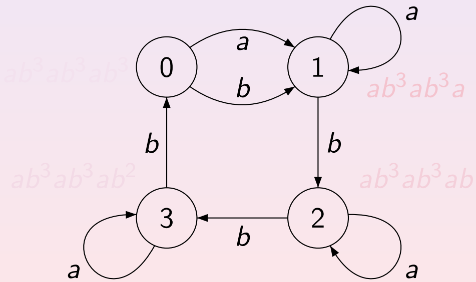
$ab^3ab^3ab^3$ ab^3ab^3a

$ab^3ab^3ab^2$ ab^3ab^3ab

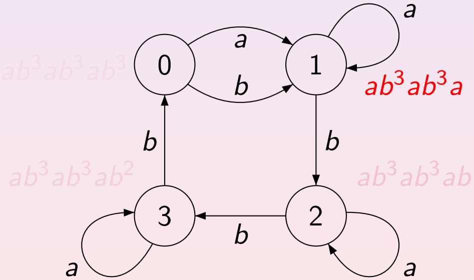
Strongly Connected Digraphs

Thus, we assume that our synchronizing automata are strongly connected as digraphs.

Observe that 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.

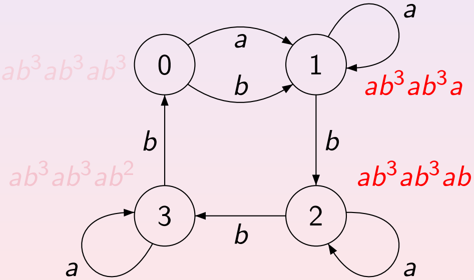


Thus, we assume that our synchronizing automata are strongly connected as digraphs.
Observe that such an automaton can be reset to any state. That is, to every state q of the automaton one can assign an instruction (a reset word) w_q such that following w_q one will surely arrive at q from any initial state.



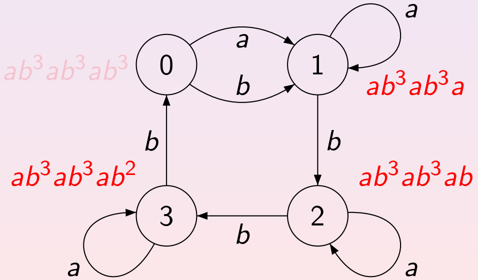
Thus, we assume that our synchronizing automata are strongly connected as digraphs.

Observe that 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.

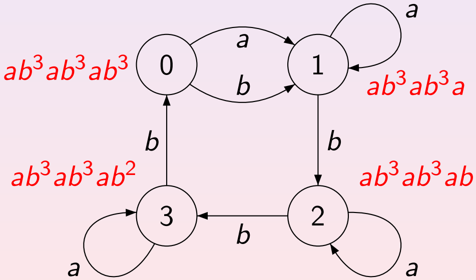


Thus, we assume that our synchronizing automata are strongly connected as digraphs.

Observe that 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.

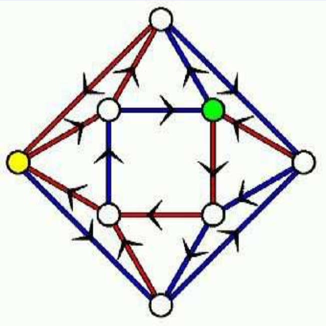


Thus, we assume that our synchronizing automata are strongly connected as digraphs.
Observe that such an automaton can be reset to any state. That is, to every state q of the automaton one can assign an instruction (a reset word) w_q such that following w_q one will surely arrive at q from any initial state.



Example

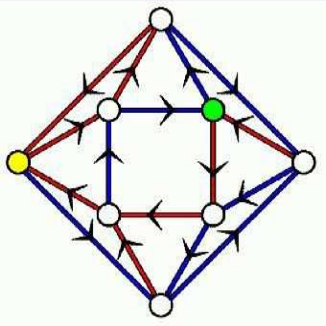
Now think of the automaton as of a scheme of a transport network in which arrows correspond to roads and labels are treated as colors of the roads.



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

Example

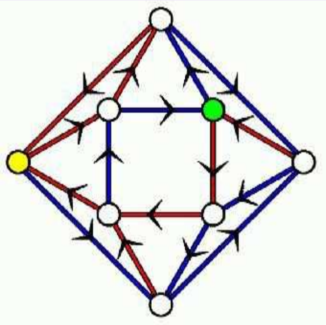
Now think of the automaton as of a scheme of a transport network in which arrows correspond to roads and labels are treated as colors of the roads.



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

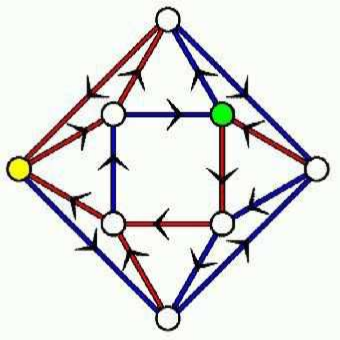
Example

Now think of the automaton as of a scheme of a transport network in which arrows correspond to roads and labels are treated as colors of the roads.



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

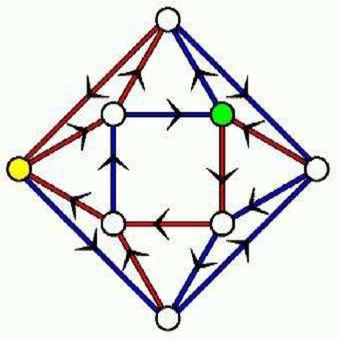
Solution to the Example



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

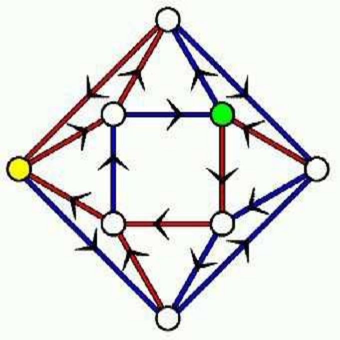
For the yellow node: blue-red-red-blue-red-red-blue-red-red.

Solution to the Example



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.



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.

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 synchronizing automata?

An obvious necessary condition:

all vertices should have the same out-degree.

In what follows we refer to this as to the constant out-degree condition.

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 synchronizing automata?

An obvious necessary condition:

all vertices should have the same out-degree.

In what follows we refer to this as to the constant out-degree condition.

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 synchronizing automata?

An obvious necessary condition:
all vertices should have the same out-degree.
In what follows we refer to this as to the constant out-degree condition.

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 synchronizing automata?

An obvious necessary condition:
all vertices should have the same out-degree.
In what follows we refer to this as to the constant out-degree condition.

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 synchronizing automata?

An obvious necessary condition:
all vertices should have the same out-degree.
In what follows we refer to this as to the constant out-degree condition.

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

To see why primitivity is necessary, suppose that $\Gamma = (V, E)$ is a strongly connected digraph and $k > 1$ is a common divisor of lengths of its cycles. Take a vertex $v_0 \in V$ and, for $i = 0, 1, \ldots, k-1$, let

$$V_i = \{v \in V \mid \exists \text{ path from } v_0 \text{ to } v \text{ of length } i \pmod{k}\}.$$

Clearly, $V = \bigcup_{i=0}^{k-1} V_i$. We claim that $V_i \cap V_j = \varnothing$ if $i \neq j$.

Necessity of Primitivity

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

To see why primitivity is necessary, suppose that $\Gamma = (V, E)$ is a strongly connected digraph and $k > 1$ is a common divisor of lengths of its cycles. Take a vertex $v_0 \in V$ and, for $i = 0, 1, \ldots, k-1$, let

$$V_i = \{v \in V \mid \exists \text{ path from } v_0 \text{ to } v \text{ of length } i \pmod{k}\}.$$

Clearly, $V = \bigcup_{i=0}^{k-1} V_i$. We claim that $V_i \cap V_j = \varnothing$ if $i \neq j$.

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

To see why primitivity is necessary, suppose that $\Gamma = (V, E)$ is a strongly connected digraph and $k > 1$ is a common divisor of lengths of its cycles. Take a vertex $v_0 \in V$ and, for $i = 0, 1, \ldots, k-1$, let

$$V_i = \{v \in V \mid \exists \text{ path from } v_0 \text{ to } v \text{ of length } i \pmod{k}\}.$$

Clearly, $V = \bigcup_{i=0}^{k-1} V_i$. We claim that $V_i \cap V_j = \varnothing$ if $i \neq j$.

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

To see why primitivity is necessary, suppose that $\Gamma = (V, E)$ is a strongly connected digraph and $k > 1$ is a common divisor of lengths of its cycles. Take a vertex $v_0 \in V$ and, for $i = 0, 1, \dots, k-1$, let

$$V_i = \{v \in V \mid \exists \text{ path from } v_0 \text{ to } v \text{ of length } i \pmod{k}\}.$$

Clearly, $V = \bigcup_{i=0}^{k-1} V_i$. We claim that $V_i \cap V_j = \varnothing$ if $i \neq j$.

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

To see why primitivity is necessary, suppose that $\Gamma = (V, E)$ is a strongly connected digraph and $k > 1$ is a common divisor of lengths of its cycles. Take a vertex $v_0 \in V$ and, for $i = 0, 1, \dots, k-1$, let

$$V_i = \{v \in V \mid \exists \text{ path from } v_0 \text{ to } v \text{ of length } i \pmod{k}\}.$$

Clearly, $V = \bigcup_{i=0}^{k-1} V_i$. We claim that $V_i \cap V_j = \varnothing$ if $i \neq j$.

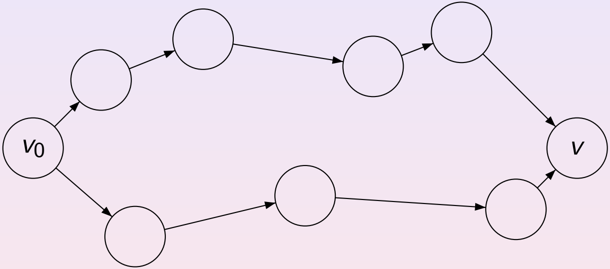
Necessity of Primitivity

Let $v \in V_i \cap V_j$ where $i \neq j$. This means that in Γ there are two paths from v_0 to v: of length $\ell \equiv i \pmod{k}$ and of length $m \equiv j \pmod{k}$.

There is also a path from v to v_0 of length, say, n. Combining it with the two paths above we get a cycle of length $\ell + n$ and a cycle of length $m + n$.

Necessity of Primitivity

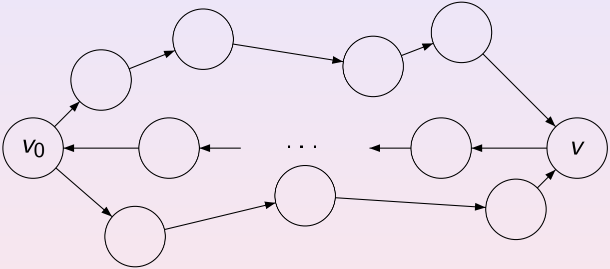
Let $v \in V_i \cap V_j$ where $i \neq j$. This means that in Γ there are two paths from v_0 to v: of length $\ell \equiv i \pmod{k}$ and of length $m \equiv j \pmod{k}$.



There is also a path from v to v_0 of length, say, n. Combining it with the two paths above we get a cycle of length $\ell + n$ and a cycle of length $m + n$.

Necessity of Primitivity

Let $v \in V_i \cap V_j$ where $i \neq j$. This means that in Γ there are two paths from v_0 to v: of length $\ell \equiv i \pmod{k}$ and of length $m \equiv j \pmod{k}$.



There is also a path from v to v_0 of length, say, n. Combining it with the two paths above we get a cycle of length $\ell + n$ and a cycle of length $m + n$.

Necessity of Primitivity

Since k divides the length of any cycle in Γ, we have $\ell + n \equiv i + n \equiv 0 \pmod{k}$ and $m + n \equiv j + n \equiv 0 \pmod{k}$, whence $i \equiv j \pmod{k}$, a contradiction.

Thus, V is a disjoint union of $V_0, V_1, \ldots, V_{k-1}$, and by the definition each arrow in Γ leads from V_i to $V_{i+1 \pmod{k}}$.

Then Γ definitely cannot be converted into a synchronizing automaton by any labelling of its arrows: for instance, no paths of the same length ℓ originated in V_0 and V_1 can terminate in the same vertex because they end in $V_{\ell \pmod{k}}$ and in $V_{\ell+1 \pmod{k}}$ respectively.

Since k divides the length of any cycle in Γ, we have $\ell + n \equiv i + n \equiv 0 \pmod{k}$ and $m + n \equiv j + n \equiv 0 \pmod{k}$, whence $i \equiv j \pmod{k}$, a contradiction.

Thus, V is a disjoint union of $V_0, V_1, \ldots, V_{k-1}$, and by the definition each arrow in Γ leads from V_i to $V_{i+1 \pmod{k}}$.

Then Γ definitely cannot be converted into a synchronizing automaton by any labelling of its arrows: for instance, no paths of the same length ℓ originated in V_0 and V_1 can terminate in the same vertex because they end in $V_{\ell \pmod{k}}$ and in $V_{\ell+1 \pmod{k}}$ respectively.

Since k divides the length of any cycle in Γ, we have $\ell + n \equiv i + n \equiv 0 \pmod{k}$ and $m + n \equiv j + n \equiv 0 \pmod{k}$, whence $i \equiv j \pmod{k}$, a contradiction.

Thus, V is a disjoint union of $V_0, V_1, \ldots, V_{k-1}$, and by the definition each arrow in Γ leads from V_i to $V_{i+1 \pmod{k}}$.

Then Γ definitely cannot be converted into a synchronizing automaton by any labelling of its arrows: for instance, no paths of the same length ℓ originated in V_0 and V_1 can terminate in the same vertex because they end in $V_{\ell \pmod{k}}$ and in $V_{\ell+1 \pmod{k}}$ respectively.

The Road Coloring Conjecture claims 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)).

The Road Coloring Conjecture claims 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)).

Road Coloring Conjecture

The Road Coloring Conjecture claims 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)).

Road Coloring Conjecture

The Road Coloring Conjecture claims 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*.

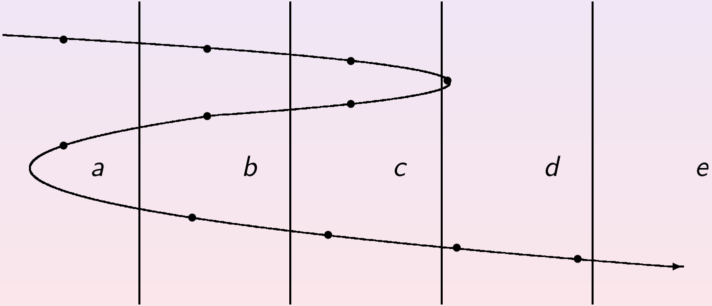
The Road Coloring Conjecture was explicitly stated by Adler, Goodwyn and Weiss in 1977 (Equivalence of topological Markov shifts, Israel J. Math., 27, 49–63). In an implicit form it was present already in an earlier memoir by Adler and Weiss (Similarity of automorphisms of the torus, Memoirs Amer. Math.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

Road Coloring Conjecture

The original motivation for the Road Coloring Conjecture comes from symbolic dynamics, see Marie-Pierre Béal and Dominique Perrin's chapter "Symbolic Dynamics and Finite Automata" in Handbook of Formal Languages, Vol.I. Springer, 1997.

Road Coloring Conjecture

The original motivation for the Road Coloring Conjecture comes from symbolic dynamics, see Marie-Pierre Béal and Dominique Perrin's chapter "Symbolic Dynamics and Finite Automata" in Handbook of Formal Languages, Vol.I. Springer, 1997.



A symbolic trajectory $\ldots abcdcbabcdd\ldots$

An archetypical object of symbolic dynamics is the collection of all labels of the bi-infinite walks on a finite automaton (a sofic subshift).

However the conjecture is natural also from the viewpoint of the "reverse engineering" of synchronizing automata as presented here.

The Road Coloring Conjecture has attracted much attention. There were several interesting partial results, and finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman. The solution is published in: The Road Coloring Problem, Israel J. Math. 172 (2009) 51–60. Trahtman's solution got much publicity.

Road Coloring Conjecture

An archetypical object of symbolic dynamics is the collection of all labels of the bi-infinite walks on a finite automaton (a sofic subshift).

However the conjecture is natural also from the viewpoint of the “reverse engineering” of synchronizing automata as presented here.

The Road Coloring Conjecture has attracted much attention. There were several interesting partial results, and finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman. The solution is published in: The Road Coloring Problem, Israel J. Math. 172 (2009) 51–60. Trahtman's solution got much publicity.

Road Coloring Conjecture

An archetypical object of symbolic dynamics is the collection of all labels of the bi-infinite walks on a finite automaton (a sofic subshift).

However the conjecture is natural also from the viewpoint of the "reverse engineering" of synchronizing automata as presented here.

The Road Coloring Conjecture has attracted much attention. There were several interesting partial results, and finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman. The solution is published in: The Road Coloring Problem, Israel J. Math. 172 (2009) 51–60. Trahtman's solution got much publicity.

Road Coloring Conjecture

An archetypical object of symbolic dynamics is the collection of all labels of the bi-infinite walks on a finite automaton (a sofic subshift).

However the conjecture is natural also from the viewpoint of the "reverse engineering" of synchronizing automata as presented here.

The Road Coloring Conjecture has attracted much attention. There were several interesting partial results, and finally the problem was solved (in the affirmative) in August 2007 by Avraham Trahtman. The solution is published in: The Road Coloring Problem, Israel J. Math. 172 (2009) 51–60. Trahtman's solution got much publicity.

Stability

Trahtman's proof heavily depends on a neat idea of stability which is due to Karel Culik II, Juhani Karhumäki and Jarkko Kari (A note on synchronized automata and Road Coloring Problem, Int. J. Found. Comput. Sci., 13 (2002) 459–471). Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. We define the relation $\sim$ on Q as follows:

$$q \sim q' \Longleftrightarrow \forall u \in \Sigma^* \; \exists v \in \Sigma^* \; q\,.\,uv = q'.uv\,.$$

$\sim$ is called the *stability relation* and any pair (q, q') such that $q \sim q'$ is called *stable*. It is immediate that $\sim$ is a congruence of the automaton $\mathscr{A}$. Also observe that $\mathscr{A}$ is synchronizing iff all pairs are stable.

Stability

Trahtman's proof heavily depends on a neat idea of stability which is due to Karel Culik II, Juhani Karhumäki and Jarkko Kari (A note on synchronized automata and Road Coloring Problem, Int. J. Found. Comput. Sci., 13 (2002) 459–471). Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. We define the relation $\sim$ on Q as follows:

$$q \sim q' \iff \forall u \in \Sigma^* \ \exists v \in \Sigma^* \ q \,.\, uv = q'.uv.$$

$\sim$ is called the *stability relation* and any pair (q, q') such that $q \sim q'$ is called *stable*. It is immediate that $\sim$ is a congruence of the automaton $\mathscr{A}$. Also observe that $\mathscr{A}$ is synchronizing iff all pairs are stable.

Trahtman's proof heavily depends on a neat idea of stability which is due to Karel Culik II, Juhani Karhumäki and Jarkko Kari (A note on synchronized automata and Road Coloring Problem, Int. J. Found. Comput. Sci., 13 (2002) 459–471). Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. We define the relation $\sim$ on Q as follows:

$$q \sim q' \iff \forall u \in \Sigma^* \; \exists v \in \Sigma^* \; q \,.\, uv = q'.uv.$$

$\sim$ is called the *stability relation* and any pair (q, q') such that $q \sim q'$ is called *stable*. It is immediate that $\sim$ is a congruence of the automaton $\mathscr{A}$. Also observe that $\mathscr{A}$ is synchronizing iff all pairs are stable.

Trahtman's proof heavily depends on a neat idea of stability which is due to Karel Culik II, Juhani Karhumäki and Jarkko Kari (A note on synchronized automata and Road Coloring Problem, Int. J. Found. Comput. Sci., 13 (2002) 459–471). Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. We define the relation $\sim$ on Q as follows:

$$q \sim q' \iff \forall u \in \Sigma^* \ \exists v \in \Sigma^* \ q \,.\, uv = q'.uv.$$

$\sim$ is called the *stability relation* and any pair (q, q') such that $q \sim q'$ is called *stable*. It is immediate that $\sim$ is a congruence of the automaton $\mathscr{A}$. Also observe that $\mathscr{A}$ is synchronizing iff all pairs are stable.

Trahtman's proof heavily depends on a neat idea of stability which is due to Karel Culik II, Juhani Karhumäki and Jarkko Kari (A note on synchronized automata and Road Coloring Problem, Int. J. Found. Comput. Sci., 13 (2002) 459–471). Let $\mathscr{A} = \langle Q, \Sigma, \delta\rangle$ be a DFA. We define the relation $\sim$ on Q as follows:

$$q \sim q' \iff \forall u \in \Sigma^* \ \exists v \in \Sigma^* \ q\,.\,uv = q'.uv.$$

$\sim$ is called the *stability relation* and any pair (q, q') such that $q \sim q'$ is called *stable*. It is immediate that $\sim$ is a congruence of the automaton $\mathscr{A}$. Also observe that $\mathscr{A}$ is synchronizing iff all pairs are stable.

We say that a coloring of a digraph with constant out-degree is *stable* if the resulting automaton contains at least one stable pair (q, q') with $q \neq q'$. The crucial observation by Culik, Karhumäki and Kari is

Proposition CKK. *Suppose every strongly connected primitive digraph with constant out-degree and more than 1 vertex has a stable coloring. Then the Road Coloring Conjecture holds true.*

The proof is rather straightforward: one inducts on the number of vertices in the digraph. If Γ admits a stable coloring and $\mathscr{A}$ is the resulting automaton, then the quotient automaton $\mathscr{A}/{\sim}$ admits a synchronizing recoloring by the induction assumption.
Then it remains to lift the correct coloring of $\mathscr{A}/\sim$ to a synchronizing coloring of Γ.

Stability

We say that a coloring of a digraph with constant out-degree is *stable* if the resulting automaton contains at least one stable pair (q, q') with $q \neq q'$. The crucial observation by Culik, Karhumäki and Kari is

Proposition CKK. *Suppose every strongly connected primitive digraph with constant out-degree and more than 1 vertex has a stable coloring. Then the Road Coloring Conjecture holds true.*

The proof is rather straightforward: one inducts on the number of vertices in the digraph. If Γ admits a stable coloring and $\mathscr{A}$ is the resulting automaton, then the quotient automaton $\mathscr{A}/{\sim}$ admits a synchronizing recoloring by the induction assumption.
Then it remains to lift the correct coloring of $\mathscr{A}/\sim$ to a synchronizing coloring of Γ.

Stability

We say that a coloring of a digraph with constant out-degree is *stable* if the resulting automaton contains at least one stable pair (q, q') with $q \neq q'$. The crucial observation by Culik, Karhumäki and Kari is

Proposition CKK. *Suppose every strongly connected primitive digraph with constant out-degree and more than 1 vertex has a stable coloring. Then the Road Coloring Conjecture holds true.*

The proof is rather straightforward: one inducts on the number of vertices in the digraph. If Γ admits a stable coloring and $\mathscr{A}$ is the resulting automaton, then the quotient automaton $\mathscr{A}/{\sim}$ admits a synchronizing recoloring by the induction assumption.
Then it remains to lift the correct coloring of $\mathscr{A}/\sim$ to a synchronizing coloring of Γ.

Stability

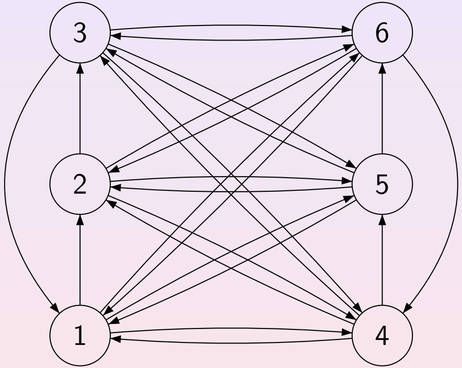
We say that a coloring of a digraph with constant out-degree is *stable* if the resulting automaton contains at least one stable pair (q, q') with $q \neq q'$. The crucial observation by Culik, Karhumäki and Kari is

Proposition CKK. *Suppose every strongly connected primitive digraph with constant out-degree and more than 1 vertex has a stable coloring. Then the Road Coloring Conjecture holds true.*

The proof is rather straightforward: one inducts on the number of vertices in the digraph. If Γ admits a stable coloring and $\mathscr{A}$ is the resulting automaton, then the quotient automaton $\mathscr{A}/{\sim}$ admits a synchronizing recoloring by the induction assumption.
Then it remains to lift the correct coloring of $\mathscr{A}/\sim$ to a synchronizing coloring of Γ.

Example

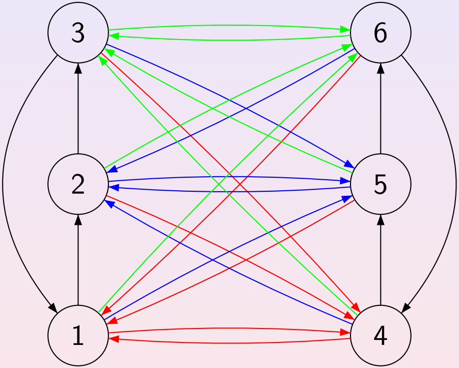
Look at the following digraph Γ and one of its colorings. It is not synchronizing (the states 1 and 4 cannot be synchronized).



One can see that the stability relation is the partition 123 | 456.

Example

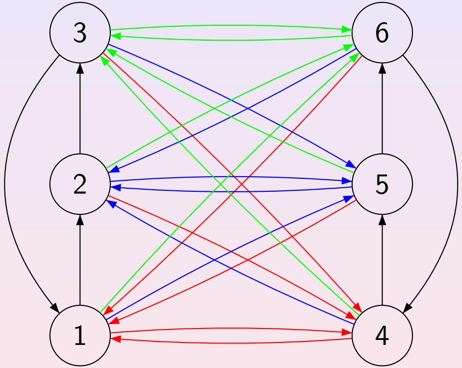
Look at the following digraph Γ and one of its colorings. It is not synchronizing (the states 1 and 4 cannot be synchronized).



One can see that the stability relation is the partition 123 | 456.

Example

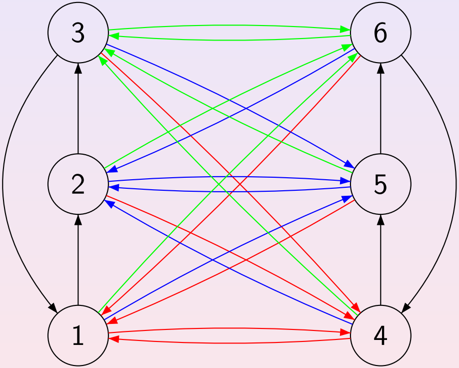
Look at the following digraph Γ and one of its colorings. It is not synchronizing (the states 1 and 4 cannot be synchronized).



One can see that the stability relation is the partition 123 | 456.

Example

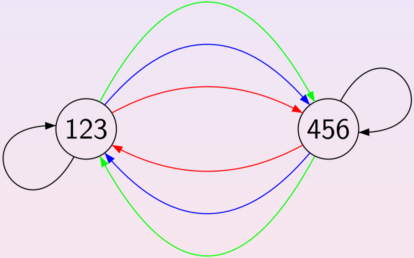
Look at the following digraph Γ and one of its colorings. It is not synchronizing (the states 1 and 4 cannot be synchronized).



One can see that the stability relation is the partition $123 \mid 456$.

Example

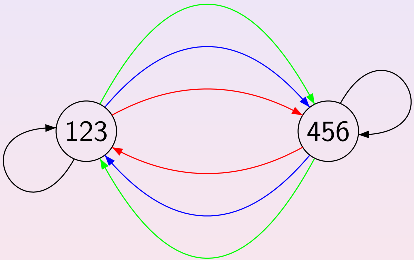
This is the quotient automaton of the above coloring. It is easy to recolor this quotient to get a synchronizing automaton.

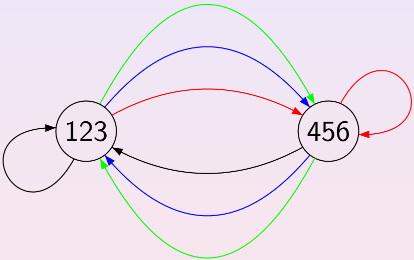


Red is a reset word for the new coloring.

Example

This is the quotient automaton of the above coloring. It is easy to recolor this quotient to get a synchronizing automaton.

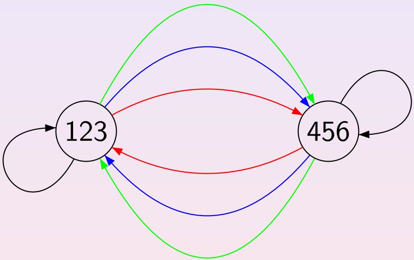




Red is a reset word for the new coloring.

Example

This is the quotient automaton of the above coloring. It is easy to recolor this quotient to get a synchronizing automaton.



Red is a reset word for the new coloring.

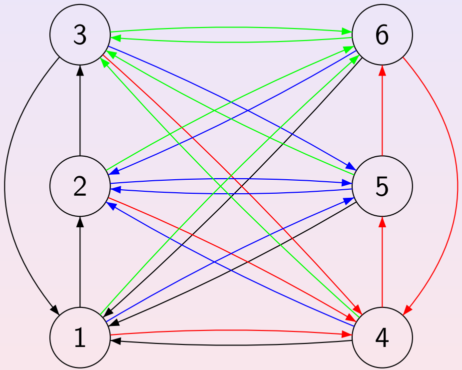
Example

Now it easy to lift the synchronizing coloring of the quotient to a synchronizing coloring of the initial digraph.

Red-Blue a reset word for the new coloring.

Example

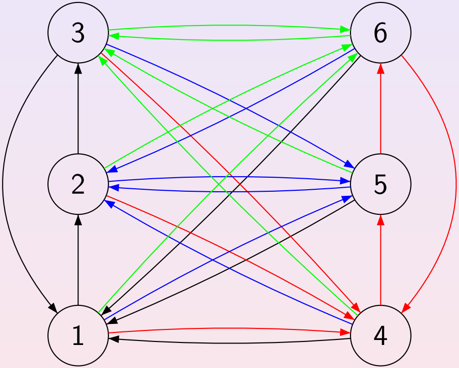
Now it easy to lift the synchronizing coloring of the quotient to a synchronizing coloring of the initial digraph.



Red-Blue a reset word for the new coloring.

Example

Now it easy to lift the synchronizing coloring of the quotient to a synchronizing coloring of the initial digraph.



Red-Blue a reset word for the new coloring.

Trahtman's Proof

Trahtman has managed to prove exactly what was needed to use Proposition CKK: every strongly connected primitive digraph with constant out-degree and more than 1 vertex has a stable coloring. Thus, Road Coloring Conjecture holds true.

The proof is clever but not too difficult.
For brevity, we call strongly connected primitive digraphs with constant out-degree and more than 1 vertex admissible.

Trahtman's Proof

Trahtman has managed to prove exactly what was needed to use Proposition CKK: every strongly connected primitive digraph with constant out-degree and more than 1 vertex has a stable coloring. Thus, Road Coloring Conjecture holds true.

The proof is clever but not too difficult.
For brevity, we call strongly connected primitive digraphs with constant out-degree and more than 1 vertex admissible.

Trahtman's Proof

Trahtman has managed to prove exactly what was needed to use Proposition CKK: every strongly connected primitive digraph with constant out-degree and more than 1 vertex has a stable coloring. Thus, Road Coloring Conjecture holds true.

The proof is clever but not too difficult.
For brevity, we call strongly connected primitive digraphs with constant out-degree and more than 1 vertex admissible.

Deadlocks and Cliques

First, we need a couple of notions.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. A pair (p, q) of distinct states is a deadlock if $\forall w \in \Sigma^* \; p \,.\, w \neq q \,.\, w$. If an automaton is not synchronizing, it must have deadlocks!

Moreover, if a pair (p, q) is not stable, then for some word $u \in \Sigma^*$ the pair $(p \,.\, u, q \,.\, u)$ is a deadlock.

A clique F is any subset of Q of maximum cardinality such that every pair of states in F is a deadlock.

Clearly, if F is a clique, so is $F \,.\, u$ for every $u \in \Sigma^*$.

First, we need a couple of notions.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. A pair (p, q) of distinct states is a deadlock if $\forall w \in \Sigma^* \ p \, . \, w \neq q \, . \, w$. If an automaton is not synchronizing, it must have deadlocks!

Moreover, if a pair (p, q) is not stable, then for some word $u \in \Sigma^*$ the pair $(p \, . \, u, q \, . \, u)$ is a deadlock.

A clique F is any subset of Q of maximum cardinality such that every pair of states in F is a deadlock.

Clearly, if F is a clique, so is $F \, . \, u$ for every $u \in \Sigma^*$.

First, we need a couple of notions.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. A pair (p, q) of distinct states is a deadlock if $\forall w \in \Sigma^* \; p \,.\, w \neq q \,.\, w$. If an automaton is not synchronizing, it must have deadlocks!

Moreover, if a pair (p, q) is not stable, then for some word $u \in \Sigma^*$ the pair $(p \,.\, u, q \,.\, u)$ is a deadlock.

A clique F is any subset of Q of maximum cardinality such that every pair of states in F is a deadlock.

Clearly, if F is a clique, so is $F \,.\, u$ for every $u \in \Sigma^*$.

First, we need a couple of notions.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. A pair (p, q) of distinct states is a deadlock if $\forall w \in \Sigma^*\ p \,.\, w \neq q \,.\, w$. If an automaton is not synchronizing, it must have deadlocks!
Moreover, if a pair (p, q) is not stable, then for some word $u \in \Sigma^*$ the pair $(p \,.\, u, q \,.\, u)$ is a deadlock.

A clique F is any subset of Q of maximum cardinality such that every pair of states in F is a deadlock.
Clearly, if F is a clique, so is $F \,.\, u$ for every $u \in \Sigma^*$.

Deadlocks and Cliques

First, we need a couple of notions.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. A pair (p, q) of distinct states is a **deadlock** if $\forall w \in \Sigma^* \ p \,.\, w \neq q \,.\, w$. If an automaton is not synchronizing, it must have deadlocks!
Moreover, if a pair (p, q) is not stable, then for some word $u \in \Sigma^*$ the pair $(p \,.\, u, q \,.\, u)$ is a deadlock.

A **clique** F is any subset of Q of maximum cardinality such that every pair of states in F is a deadlock.
Clearly, if F is a clique, so is $F \,.\, u$ for every $u \in \Sigma^*$.

Deadlocks and Cliques

First, we need a couple of notions.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA. A pair (p, q) of distinct states is a deadlock if $\forall w \in \Sigma^* \ p \,.\, w \neq q \,.\, w$. If an automaton is not synchronizing, it must have deadlocks!
Moreover, if a pair (p, q) is not stable, then for some word $u \in \Sigma^*$ the pair $(p \,.\, u, q \,.\, u)$ is a deadlock.

A clique F is any subset of Q of maximum cardinality such that every pair of states in F is a deadlock.
Clearly, if F is a clique, so is $F \,.\, u$ for every $u \in \Sigma^*$.

Lemma 1. *Let* $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ *be an automaton. If* $F, G \subseteq Q$ *are two cliques in* $\mathscr{A}$ *such that*

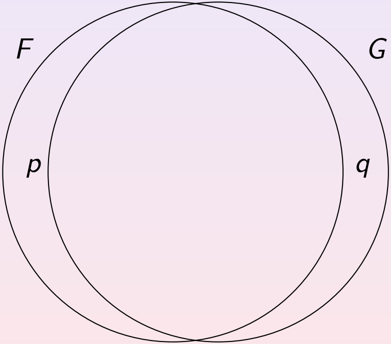
$$|F| - |F \cap G| = |G| - |F \cap G| = 1,$$

then $\mathscr{A}$ *has a stable pair.*

Lemma 1. *Let* $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ *be an automaton. If* $F, G \subseteq Q$ *are two cliques in* $\mathscr{A}$ *such that*

$$|F| - |F \cap G| = |G| - |F \cap G| = 1,$$

then $\mathscr{A}$ *has a stable pair.*



Proof. Suppose that $|F| - |F \cap G| = |G| - |F \cap G| = 1$ and let p be the only element in $F \setminus G$ and q the only element in $G \setminus F$. If the pair (p, q) is not stable, then for some word $u \in \Sigma^*$, the pair $(p \,.\, u, q \,.\, u)$ is a deadlock. Then all pairs in $(F \cup G) \,.\, u$ are deadlocks and $|(F \cup G) \,.\, u| = |F| + 1$, a contradiction.

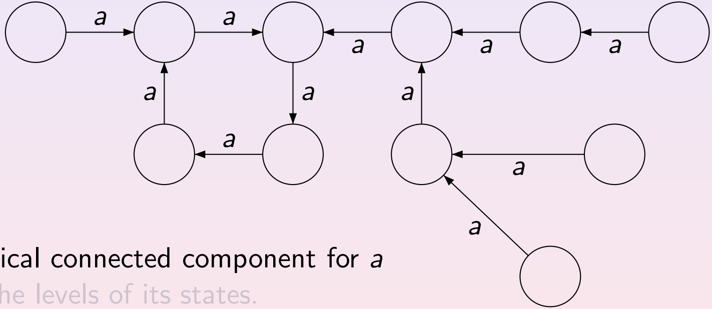
Proof. Suppose that $|F| - |F \cap G| = |G| - |F \cap G| = 1$ and let p be the only element in $F \setminus G$ and q the only element in $G \setminus F$. If the pair (p, q) is not stable, then for some word $u \in \Sigma^*$, the pair $(p \,.\, u, q \,.\, u)$ is a deadlock. Then all pairs in $(F \cup G) \,.\, u$ are deadlocks and $|(F \cup G) \,.\, u| = |F| + 1$, a contradiction.

Proof. Suppose that $|F| - |F \cap G| = |G| - |F \cap G| = 1$ and let p be the only element in $F \setminus G$ and q the only element in $G \setminus F$. If the pair (p, q) is not stable, then for some word $u \in \Sigma^*$, the pair $(p \,.\, u, q \,.\, u)$ is a deadlock. Then all pairs in $(F \cup G) \,.\, u$ are deadlocks and $|(F \cup G) \,.\, u| = |F| + 1$, a contradiction.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA, $a \in \Sigma$. We want to assign to its states a parameter called the level w.r.t. a.

A typical connected component for a and the levels of its states.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA, $a \in \Sigma$. We want to assign to its states a parameter called the level w.r.t. a.

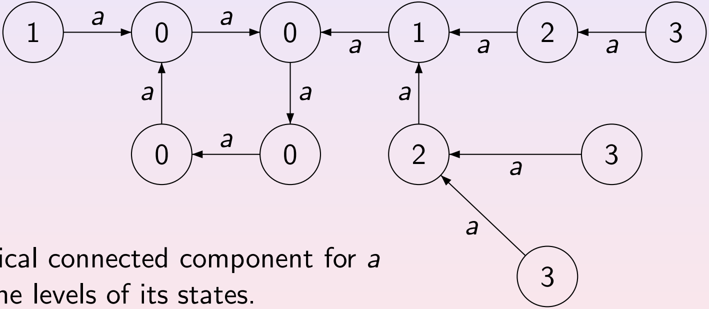


A typical connected component for a

and the levels of its states.

Levels w.r.t. a Letter

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA, $a \in \Sigma$. We want to assign to its states a parameter called the level w.r.t. a.



A typical connected component for a and the levels of its states.

Lemma on Level

Lemma 2. *Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a strongly connected automaton such that all states of maximal level $L > 0$ w.r.t. $a \in \Sigma$ belong to the same tree. Then $\mathscr{A}$ has a stable pair.*

Proof. Let M be the set of all states of level L w.r.t a. Then $p\,.\,a^L = q\,.\,a^L$ for all $p, q \in M$ whence no pair of states from M forms a deadlock. Thus, if $C \subseteq Q$ is a clique then $|C \cap M| \leq 1$. Take a clique C such that $|C \cap M| = 1$ (it exists since $\mathscr{A}$ is strongly connected). Then $F = C.a^{L-1}$ is a clique that has all its states except one in the a-cycles. If m is the l.c.m. of the lengths of all a-cycles, $r\,.\,a^m = r$ for any r in any a-cycle. Hence $G = F\,.\,a^m$ is a clique such that

$$|F| - |F \cap G| = |G| - |F \cap G| = 1.$$

By Lemma 1 $\mathscr{A}$ has a stable pair.

Lemma on Level

Lemma 2. *Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a strongly connected automaton such that all states of maximal level $L > 0$ w.r.t. $a \in \Sigma$ belong to the same tree. Then $\mathscr{A}$ has a stable pair.*

Proof. Let M be the set of all states of level L w.r.t a. Then $p \,.\, a^L = q \,.\, a^L$ for all $p, q \in M$ whence no pair of states from M forms a deadlock. Thus, if $C \subseteq Q$ is a clique then $|C \cap M| \leq 1$. Take a clique C such that $|C \cap M| = 1$ (it exists since $\mathscr{A}$ is strongly connected). Then $F = C . a^{L-1}$ is a clique that has all its states except one in the a-cycles. If m is the l.c.m. of the lengths of all a-cycles, $r \,.\, a^m = r$ for any r in any a-cycle. Hence $G = F \,.\, a^m$ is a clique such that

$$|F| - |F \cap G| = |G| - |F \cap G| = 1.$$

By Lemma 1 $\mathscr{A}$ has a stable pair.

Lemma on Level

Lemma 2. *Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a strongly connected automaton such that all states of maximal level $L > 0$ w.r.t. $a \in \Sigma$ belong to the same tree. Then $\mathscr{A}$ has a stable pair.*

Proof. Let M be the set of all states of level L w.r.t a. Then $p\,.\,a^L = q\,.\,a^L$ for all $p, q \in M$ whence no pair of states from M forms a deadlock. Thus, if $C \subseteq Q$ is a clique then $|C \cap M| \leq 1$. Take a clique C such that $|C \cap M| = 1$ (it exists since $\mathscr{A}$ is strongly connected). Then $F = C.a^{L-1}$ is a clique that has all its states except one in the a-cycles. If m is the l.c.m. of the lengths of all a-cycles, $r\,.\,a^m = r$ for any r in any a-cycle. Hence $G = F\,.\,a^m$ is a clique such that

$$|F| - |F \cap G| = |G| - |F \cap G| = 1.$$

By Lemma 1 $\mathscr{A}$ has a stable pair.

Lemma 2. *Let* $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ *be a strongly connected automaton such that all states of maximal level* $L > 0$ *w.r.t.* $a \in \Sigma$ *belong to the same tree. Then* $\mathscr{A}$ *has a stable pair.*

Proof. Let M be the set of all states of level L w.r.t a. Then $p \,.\, a^L = q \,.\, a^L$ for all $p, q \in M$ whence no pair of states from M forms a deadlock. Thus, if $C \subseteq Q$ is a clique then $|C \cap M| \leq 1$. Take a clique C such that $|C \cap M| = 1$ (it exists since $\mathscr{A}$ is strongly connected). Then $F = C.a^{L-1}$ is a clique that has all its states except one in the a-cycles. If m is the l.c.m. of the lengths of all a-cycles, $r \,.\, a^m = r$ for any r in any a-cycle. Hence $G = F \,.\, a^m$ is a clique such that

$$|F| - |F \cap G| = |G| - |F \cap G| = 1.$$

By Lemma 1 $\mathscr{A}$ has a stable pair.

Lemma on Level

Lemma 2. *Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a strongly connected automaton such that all states of maximal level $L > 0$ w.r.t. $a \in \Sigma$ belong to the same tree. Then $\mathscr{A}$ has a stable pair.*

Proof. Let M be the set of all states of level L w.r.t a. Then $p \,.\, a^L = q \,.\, a^L$ for all $p, q \in M$ whence no pair of states from M forms a deadlock. Thus, if $C \subseteq Q$ is a clique then $|C \cap M| \leq 1$. Take a clique C such that $|C \cap M| = 1$ (it exists since $\mathscr{A}$ is strongly connected). Then $F = C.a^{L-1}$ is a clique that has all its states except one in the a-cycles. If m is the l.c.m. of the lengths of all a-cycles, $r \,.\, a^m = r$ for any r in any a-cycle. Hence $G = F \,.\, a^m$ is a clique such that

$$|F| - |F \cap G| = |G| - |F \cap G| = 1.$$

By Lemma 1 $\mathscr{A}$ has a stable pair.

Lemma 2. *Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a strongly connected automaton such that all states of maximal level $L > 0$ w.r.t. $a \in \Sigma$ belong to the same tree. Then $\mathscr{A}$ has a stable pair.*

Proof. Let M be the set of all states of level L w.r.t a. Then $p \,.\, a^L = q \,.\, a^L$ for all $p, q \in M$ whence no pair of states from M forms a deadlock. Thus, if $C \subseteq Q$ is a clique then $|C \cap M| \leq 1$. Take a clique C such that $|C \cap M| = 1$ (it exists since $\mathscr{A}$ is strongly connected). Then $F = C.a^{L-1}$ is a clique that has all its states except one in the a-cycles. If m is the l.c.m. of the lengths of all a-cycles, $r \,.\, a^m = r$ for any r in any a-cycle. Hence $G = F \,.\, a^m$ is a clique such that

$$|F| - |F \cap G| = |G| - |F \cap G| = 1.$$

By Lemma 1 $\mathscr{A}$ has a stable pair.

Lemma 2. *Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a strongly connected automaton such that all states of maximal level $L > 0$ w.r.t. $a \in \Sigma$ belong to the same tree. Then $\mathscr{A}$ has a stable pair.*

Proof. Let M be the set of all states of level L w.r.t a. Then $p \,.\, a^L = q \,.\, a^L$ for all $p, q \in M$ whence no pair of states from M forms a deadlock. Thus, if $C \subseteq Q$ is a clique then $|C \cap M| \leq 1$. Take a clique C such that $|C \cap M| = 1$ (it exists since $\mathscr{A}$ is strongly connected). Then $F = C.a^{L-1}$ is a clique that has all its states except one in the a-cycles. If m is the l.c.m. of the lengths of all a-cycles, $r \,.\, a^m = r$ for any r in any a-cycle. Hence $G = F \,.\, a^m$ is a clique such that

$$|F| - |F \cap G| = |G| - |F \cap G| = 1.$$

By Lemma 1 $\mathscr{A}$ has a stable pair.

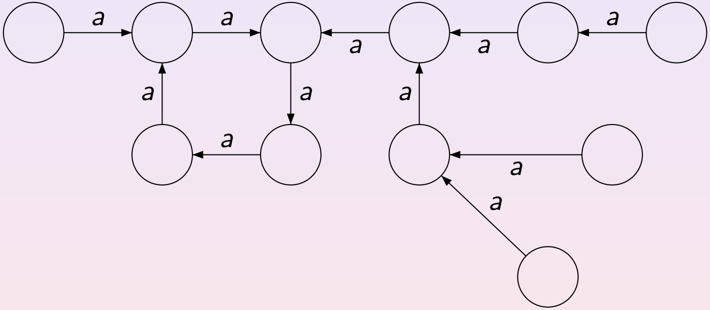
Lemma 2. *Let $\mathscr{A} = \langle Q, \Sigma, \delta\rangle$ be a strongly connected automaton such that all states of maximal level $L > 0$ w.r.t. $a \in \Sigma$ belong to the same tree. Then $\mathscr{A}$ has a stable pair.*

Proof. Let M be the set of all states of level L w.r.t a. Then $p\,.\,a^L = q\,.\,a^L$ for all $p, q \in M$ whence no pair of states from M forms a deadlock. Thus, if $C \subseteq Q$ is a clique then $|C \cap M| \leq 1$. Take a clique C such that $|C \cap M| = 1$ (it exists since $\mathscr{A}$ is strongly connected). Then $F = C.a^{L-1}$ is a clique that has all its states except one in the a-cycles. If m is the l.c.m. of the lengths of all a-cycles, $r\,.\,a^m = r$ for any r in any a-cycle. Hence $G = F\,.\,a^m$ is a clique such that

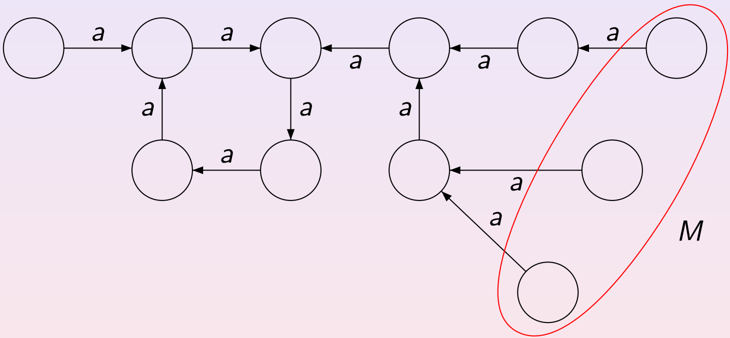
$$|F| - |F \cap G| = |G| - |F \cap G| = 1.$$

By Lemma 1 $\mathscr{A}$ has a stable pair.

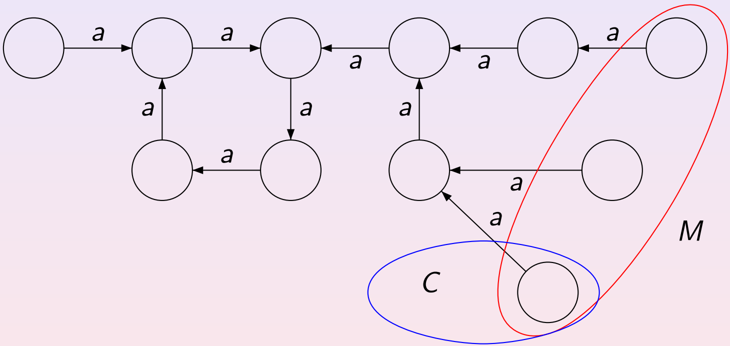
Lemma on Level



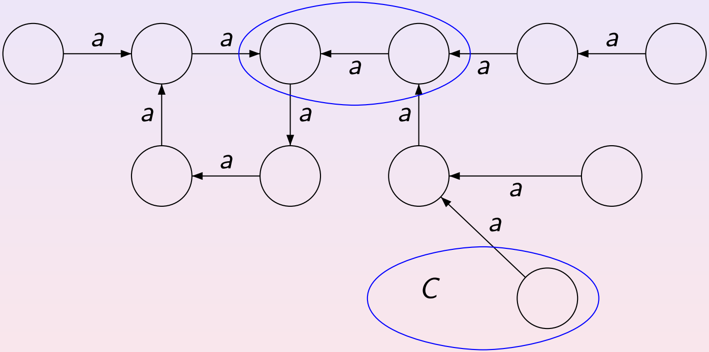
Lemma on Level



Lemma on Level

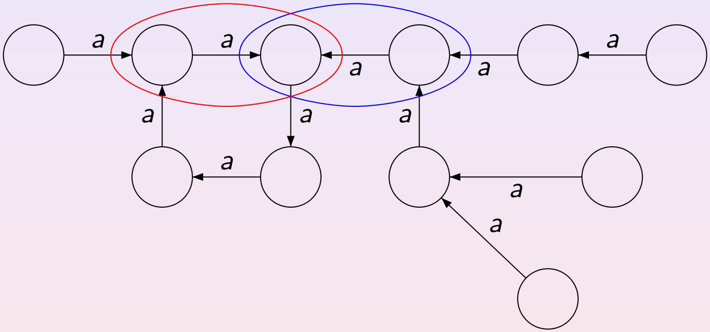


$$F = C \,.\, a^{L-1}$$



Lemma on Level

$$G = F\,.\,a^m \quad F = C\,.\,a^{L-1}$$



Reduction

Recall, that we try to prove that every admissible digraph Γ has a stable coloring. By Lemma 2 for this it suffices to show that every such Γ may be colored into an automaton satisfying the premise of the lemma. This is, of course, much easier task because basically we only need to deal with one color, that is, with the action of one letter.

Take an arbitrary admissible digraph Γ. We start with an arbitrary coloring of Γ, take an arbitrary color (=letter) a, and induct on the number N of states that do not lie on any a-cycle in the initial coloring.

Reduction

Recall, that we try to prove that every admissible digraph Γ has a stable coloring. By Lemma 2 for this it suffices to show that every such Γ may be colored into an automaton satisfying the premise of the lemma. This is, of course, much easier task because basically we only need to deal with one color, that is, with the action of one letter.

Take an arbitrary admissible digraph Γ. We start with an arbitrary coloring of Γ, take an arbitrary color (=letter) a, and induct on the number N of states that do not lie on any a-cycle in the initial coloring.

Reduction

Recall, that we try to prove that every admissible digraph Γ has a stable coloring. By Lemma 2 for this it suffices to show that every such Γ may be colored into an automaton satisfying the premise of the lemma. This is, of course, much easier task because basically we only need to deal with one color, that is, with the action of one letter.

Take an arbitrary admissible digraph Γ. We start with an arbitrary coloring of Γ, take an arbitrary color (=letter) a, and induct on the number N of states that do not lie on any a-cycle in the initial coloring.

Reduction

Recall, that we try to prove that every admissible digraph Γ has a stable coloring. By Lemma 2 for this it suffices to show that every such Γ may be colored into an automaton satisfying the premise of the lemma. This is, of course, much easier task because basically we only need to deal with one color, that is, with the action of one letter.

Take an arbitrary admissible digraph Γ. We start with an arbitrary coloring of Γ, take an arbitrary color (=letter) a, and induct on the number N of states that do not lie on any a-cycle in the initial coloring.

Induction Basis

Suppose that $N = 0$. This means that all states lie on a-cycles.

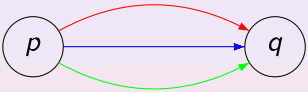
We say that a vertex p of Γ is a bunch if all edges that begin at p lead to the same vertex q.

If all vertices in Γ are bunches, then there is just one a-cycle (since Γ is strongly connected) and all cycles in Γ have the same length. This contradicts the assumption that Γ is primitive.

It is quite interesting that this is the only place in the whole proof where the primitivity condition is invoked.

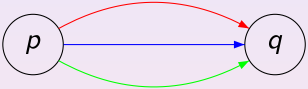
Induction Basis

Suppose that $N = 0$. This means that all states lie on a-cycles. We say that a vertex p of Γ is a bunch if all edges that begin at p lead to the same vertex q.



If all vertices in Γ are bunches, then there is just one a-cycle (since Γ is strongly connected) and all cycles in Γ have the same length. This contradicts the assumption that Γ is primitive.
It is quite interesting that this is the only place in the whole proof where the primitivity condition is invoked.

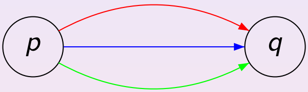
Suppose that $N = 0$. This means that all states lie on a-cycles. We say that a vertex p of Γ is a bunch if all edges that begin at p lead to the same vertex q.



If all vertices in Γ are bunches, then there is just one a-cycle (since Γ is strongly connected) and all cycles in Γ have the same length. This contradicts the assumption that Γ is primitive.

It is quite interesting that this is the only place in the whole proof where the primitivity condition is invoked.

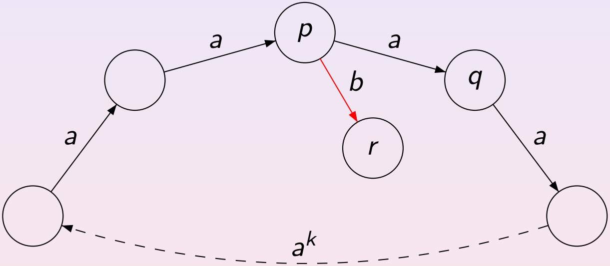
Suppose that $N = 0$. This means that all states lie on a-cycles.
We say that a vertex p of Γ is a bunch if all edges that begin at p lead to the same vertex q.



If all vertices in Γ are bunches, then there is just one a-cycle (since Γ is strongly connected) and all cycles in Γ have the same length.
This contradicts the assumption that Γ is primitive.
It is quite interesting that this is the only place in the whole proof where the primitivity condition is invoked.

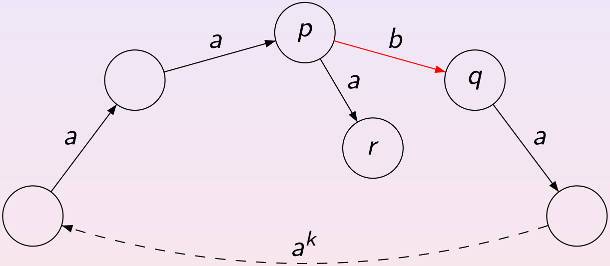
Induction Basis

Thus, let p be a state which is not a bunch, let $q = p \, . \, a$ and let $b \neq a$ be such that $r = p \, . \, b \neq q$. We exchange the labels of the edges $p \xrightarrow{a} q$ and $p \xrightarrow{b} r$.



It is clear that in the new coloring there is only one state of maximal level w.r.t. a, namely q. Thus, the induction basis is verified.

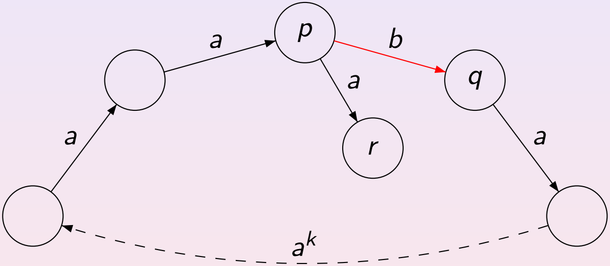
Thus, let p be a state which is not a bunch, let $q = p\,.\,a$ and let $b \neq a$ be such that $r = p\,.\,b \neq q$. We exchange the labels of the edges $p \xrightarrow{a} q$ and $p \xrightarrow{b} r$.



It is clear that in the new coloring there is only one state of maximal level w.r.t. a, namely q. Thus, the induction basis is verified.

Induction Basis

Thus, let p be a state which is not a bunch, let $q = p\,.\,a$ and let $b \neq a$ be such that $r = p\,.\,b \neq q$. We exchange the labels of the edges $p \xrightarrow{a} q$ and $p \xrightarrow{b} r$.



It is clear that in the new coloring there is only one state of maximal level w.r.t. a, namely q. Thus, the induction basis is verified.

Induction Step

Now let $N > 0$. We denote by L the maximum level of the states w.r.t. a in the initial coloring. Observe that $N > 0$ implies $L > 0$. Let p be a state of level L. Since Γ is strongly connected, there is an edge $p' \to p$ with $p' \neq p$, and by the choice of p, the label of this edge is $b \neq a$. Let $t = p' \,.\, a$. One has $t \neq p$. Let $r = p \,.\, a^L$ and let C be the a-cycle on which r lies.

The following considerations split in several cases. In each case except one we can recolor Γ by swapping the labels of two edges such the new coloring either satisfies the premise of Lemma 2 (all states of maximal level w.r.t. a belong to the same tree) or has more states on the a-cycles (and the induction assumption applies). The remaining case turns out to be easy.

Induction Step

Now let $N > 0$. We denote by L the maximum level of the states w.r.t. a in the initial coloring. Observe that $N > 0$ implies $L > 0$. Let p be a state of level L. Since Γ is strongly connected, there is an edge $p' \to p$ with $p' \neq p$, and by the choice of p, the label of this edge is $b \neq a$. Let $t = p' \,.\, a$. One has $t \neq p$. Let $r = p \,.\, a^L$ and let C be the a-cycle on which r lies.

The following considerations split in several cases. In each case except one we can recolor Γ by swapping the labels of two edges such the new coloring either satisfies the premise of Lemma 2 (all states of maximal level w.r.t. a belong to the same tree) or has more states on the a-cycles (and the induction assumption applies). The remaining case turns out to be easy.

Now let $N > 0$. We denote by L the maximum level of the states w.r.t. a in the initial coloring. Observe that $N > 0$ implies $L > 0$. Let p be a state of level L. Since Γ is strongly connected, there is an edge $p' \to p$ with $p' \neq p$, and by the choice of p, the label of this edge is $b \neq a$. Let $t = p' \,.\, a$. One has $t \neq p$. Let $r = p \,.\, a^L$ and let C be the a-cycle on which r lies.

The following considerations split in several cases. In each case except one we can recolor Γ by swapping the labels of two edges such the new coloring either satisfies the premise of Lemma 2 (all states of maximal level w.r.t. a belong to the same tree) or has more states on the a-cycles (and the induction assumption applies). The remaining case turns out to be easy.

Induction Step

Now let $N > 0$. We denote by L the maximum level of the states w.r.t. a in the initial coloring. Observe that $N > 0$ implies $L > 0$. Let p be a state of level L. Since Γ is strongly connected, there is an edge $p' \to p$ with $p' \neq p$, and by the choice of p, the label of this edge is $b \neq a$. Let $t = p' \, . \, a$. One has $t \neq p$. Let $r = p \, . \, a^L$ and let C be the a-cycle on which r lies.

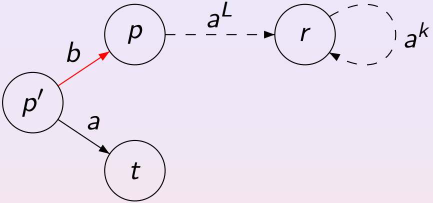
The following considerations split in several cases. In each case except one we can recolor Γ by swapping the labels of two edges such the new coloring either satisfies the premise of Lemma 2 (all states of maximal level w.r.t. a belong to the same tree) or has more states on the a-cycles (and the induction assumption applies). The remaining case turns out to be easy.

Induction Step

Now let $N > 0$. We denote by L the maximum level of the states w.r.t. a in the initial coloring. Observe that $N > 0$ implies $L > 0$. Let p be a state of level L. Since Γ is strongly connected, there is an edge $p' \to p$ with $p' \neq p$, and by the choice of p, the label of this edge is $b \neq a$. Let $t = p' \,.\, a$. One has $t \neq p$. Let $r = p \,.\, a^L$ and let C be the a-cycle on which r lies.

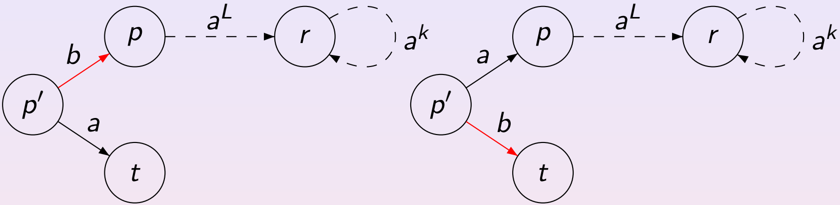
The following considerations split in several cases. In each case except one we can recolor Γ by swapping the labels of two edges such the new coloring either satisfies the premise of Lemma 2 (all states of maximal level w.r.t. a belong to the same tree) or has more states on the a-cycles (and the induction assumption applies). The remaining case turns out to be easy.

Case 1: p' is not on C.



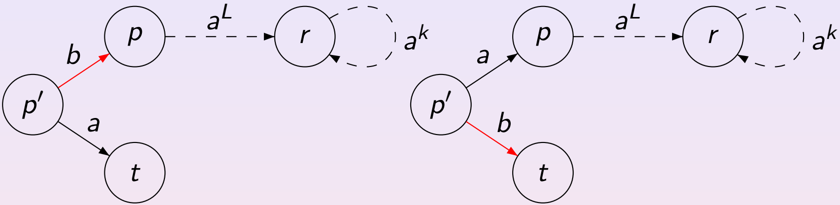
We swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If p' was on the a-path from p to r, then the swapping creates a new a-cycle increasing the number of states on the a-cycles. If p' was not on the a-path from p to r, then the level of p' w.r.t. a becomes $L+1$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of p' and thus belong to the same tree.

Case 1: p' is not on C.



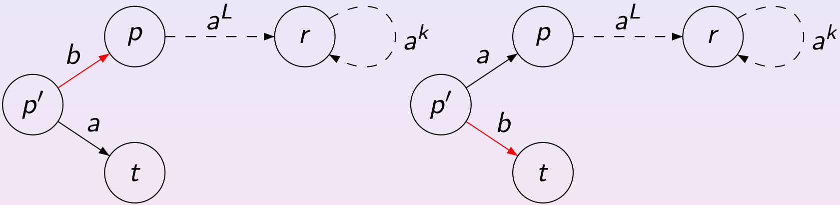
We swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If p' was on the a-path from p to r, then the swapping creates a new a-cycle increasing the number of states on the a-cycles. If p' was not on the a-path from p to r, then the level of p' w.r.t. a becomes $L + 1$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of p' and thus belong to the same tree.

Case 1: p' is not on C.



We swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If p' was on the a-path from p to r, then the swapping creates a new a-cycle increasing the number of states on the a-cycles. If p' was not on the a-path from p to r, then the level of p' w.r.t. a becomes $L+1$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of p' and thus belong to the same tree.

Case 1: p' is not on C.



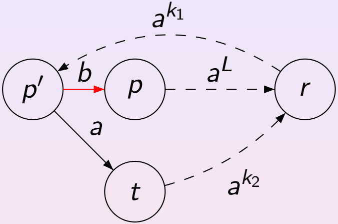
We swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If p' was on the a-path from p to r, then the swapping creates a new a-cycle increasing the number of states on the a-cycles. If p' was not on the a-path from p to r, then the level of p' w.r.t. a becomes $L+1$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of p' and thus belong to the same tree.

Case 2: p' is on C. Let k_1 be the least integer such that $r \,.\, a^{k_1} = p'$. The state $t = p' \,.\, a$ is also on C. Let k_2 be the least integer such that $t \,.\, a^{k_2} = r$. Then the length of C is $k_1 + k_2 + 1$.

Subcase 2.1: $k_2 \neq L$. Again, we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If $k_2 < L$, then the swapping creates an a-cycle of length $k_1 + L + 1 > k_1 + k_2 + 1$ increasing the number of states on the a-cycles. If $k_2 > L$, then the level of t w.r.t. a becomes k_2 whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of t and thus belong to the same tree.

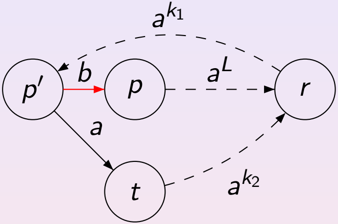
Induction Step: Subcase 2.1

Case 2: p' is on C. Let k_1 be the least integer such that $r\,.\,a^{k_1} = p'$. The state $t = p'\,.\,a$ is also on C. Let k_2 be the least integer such that $t\,.\,a^{k_2} = r$. Then the length of C is $k_1 + k_2 + 1$.



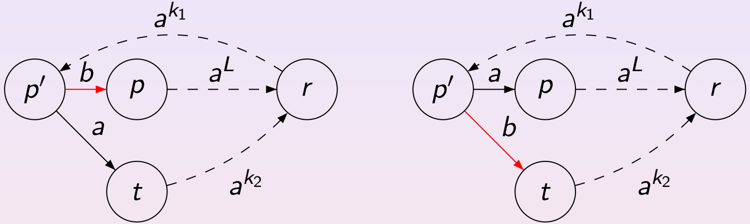
Subcase 2.1: $k_2 \neq L$. Again, we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If $k_2 < L$, then the swapping creates an a-cycle of length $k_1 + L + 1 > k_1 + k_2 + 1$ increasing the number of states on the a-cycles. If $k_2 > L$, then the level of t w.r.t. a becomes k_2 whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of t and thus belong to the same tree.

Case 2: p' is on C. Let k_1 be the least integer such that $r\,.\,a^{k_1} = p'$. The state $t = p'\,.\,a$ is also on C. Let k_2 be the least integer such that $t\,.\,a^{k_2} = r$. Then the length of C is $k_1 + k_2 + 1$.



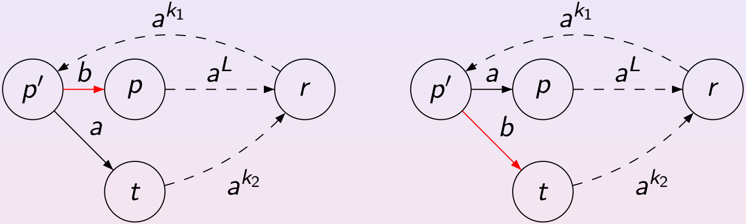
Subcase 2.1: $k_2 \neq L$. Again, we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If $k_2 < L$, then the swapping creates an a-cycle of length $k_1 + L + 1 > k_1 + k_2 + 1$ increasing the number of states on the a-cycles. If $k_2 > L$, then the level of t w.r.t. a becomes k_2 whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of t and thus belong to the same tree.

Case 2: p' is on C. Let k_1 be the least integer such that $r \,.\, a^{k_1} = p'$. The state $t = p' \,.\, a$ is also on C. Let k_2 be the least integer such that $t \,.\, a^{k_2} = r$. Then the length of C is $k_1 + k_2 + 1$.



Subcase 2.1: $k_2 \neq L$. Again, we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If $k_2 < L$, then the swapping creates an a-cycle of length $k_1 + L + 1 > k_1 + k_2 + 1$ increasing the number of states on the a-cycles. If $k_2 > L$, then the level of t w.r.t. a becomes k_2 whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of t and thus belong to the same tree.

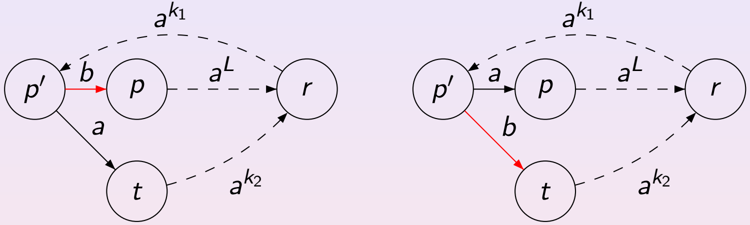
Case 2: p' is on C. Let k_1 be the least integer such that $r\,.\,a^{k_1} = p'$. The state $t = p'\,.\,a$ is also on C. Let k_2 be the least integer such that $t\,.\,a^{k_2} = r$. Then the length of C is $k_1 + k_2 + 1$.



Subcase 2.1: $k_2 \neq L$. Again, we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If $k_2 < L$, then the swapping creates an a-cycle of length $k_1 + L + 1 > k_1 + k_2 + 1$ increasing the number of states on the a-cycles. If $k_2 > L$, then the level of t w.r.t. a becomes k_2 whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of t and thus belong to the same tree.

Induction Step: Subcase 2.1

Case 2: p' is on C. Let k_1 be the least integer such that $r \,.\, a^{k_1} = p'$. The state $t = p' \,.\, a$ is also on C. Let k_2 be the least integer such that $t \,.\, a^{k_2} = r$. Then the length of C is $k_1 + k_2 + 1$.



Subcase 2.1: $k_2 \neq L$. Again, we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$. If $k_2 < L$, then the swapping creates an a-cycle of length $k_1 + L + 1 > k_1 + k_2 + 1$ increasing the number of states on the a-cycles. If $k_2 > L$, then the level of t w.r.t. a becomes k_2 whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of t and thus belong to the same tree.

Induction Step: Subcase 2.2

Let s be the state of C such that $s\,.\,a = r$.

Subcase 2.2: $k_2 = L$ and s is not a bunch. Since s is not a bunch, there is a letter c such that $s' = s\,.\,c \neq r$.

We swap the labels of $s \xrightarrow{c} s'$ and $s \xrightarrow{a} r$. If r still lies on an a-cycle, then the length of the a-cycle is at least $k_1 + k_2 + 2$ and the number of states on the a-cycles increases. Otherwise, the level of r w.r.t. a becomes at least $k_1 + k_2 + 1 > L$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of r and belong to the same tree.

Induction Step: Subcase 2.2

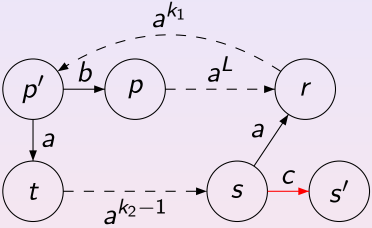
Let s be the state of C such that $s \,.\, a = r$.

Subcase 2.2: $k_2 = L$ and s is not a bunch. Since s is not a bunch, there is a letter c such that $s' = s \,.\, c \neq r$.

We swap the labels of $s \xrightarrow{c} s'$ and $s \xrightarrow{a} r$. If r still lies on an a-cycle, then the length of the a-cycle is at least $k_1 + k_2 + 2$ and the number of states on the a-cycles increases. Otherwise, the level of r w.r.t. a becomes at least $k_1 + k_2 + 1 > L$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of r and belong to the same tree.

Let s be the state of C such that $s\,.\,a = r$.

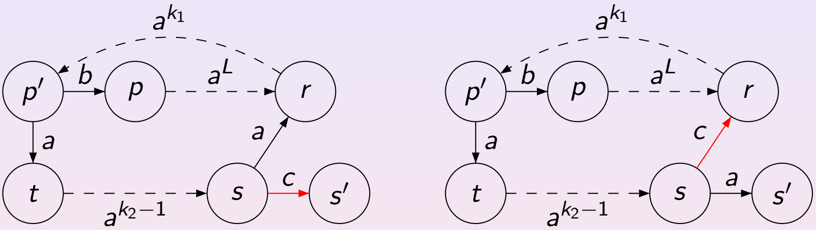
Subcase 2.2: $k_2 = L$ and s is not a bunch. Since s is not a bunch, there is a letter c such that $s' = s\,.\,c \neq r$.



We swap the labels of $s \xrightarrow{c} s'$ and $s \xrightarrow{a} r$. If r still lies on an a-cycle, then the length of the a-cycle is at least $k_1 + k_2 + 2$ and the number of states on the a-cycles increases. Otherwise, the level of r w.r.t. a becomes at least $k_1 + k_2 + 1 > L$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of r and belong to the same tree.

Induction Step: Subcase 2.2

Let s be the state of C such that $s\,.\,a = r$.
Subcase 2.2: $k_2 = L$ and s is not a bunch. Since s is not a bunch, there is a letter c such that $s' = s\,.\,c \neq r$.

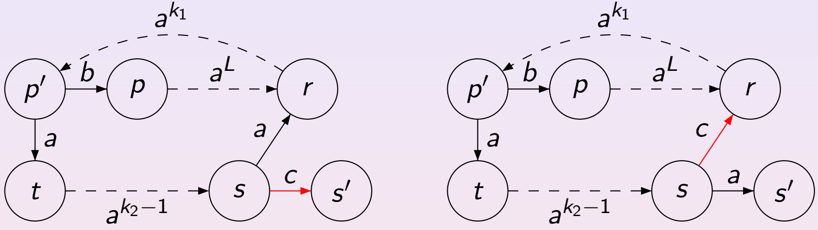


We swap the labels of $s \xrightarrow{c} s'$ and $s \xrightarrow{a} r$. If r still lies on an a-cycle, then the length of the a-cycle is at least $k_1 + k_2 + 2$ and the number of states on the a-cycles increases. Otherwise, the level of r w.r.t. a becomes at least $k_1 + k_2 + 1 > L$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of r and belong to the same tree.

Induction Step: Subcase 2.2

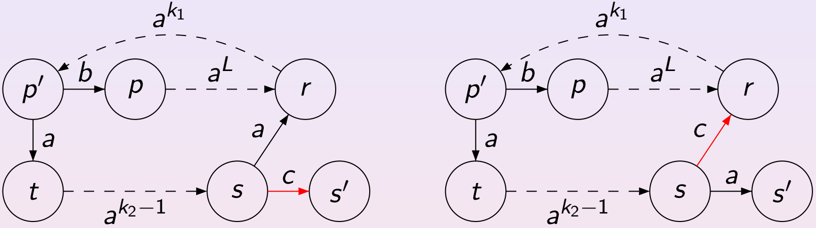
Let s be the state of C such that $s \,.\, a = r$.

Subcase 2.2: $k_2 = L$ and s is not a bunch. Since s is not a bunch, there is a letter c such that $s' = s \,.\, c \neq r$.



We swap the labels of $s \xrightarrow{c} s'$ and $s \xrightarrow{a} r$. If r still lies on an a-cycle, then the length of the a-cycle is at least $k_1 + k_2 + 2$ and the number of states on the a-cycles increases. Otherwise, the level of r w.r.t. a becomes at least $k_1 + k_2 + 1 > L$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of r and belong to the same tree.

Let s be the state of C such that $s\,.\,a = r$.
Subcase 2.2: $k_2 = L$ **and** s **is not a bunch.** Since s is not a bunch, there is a letter c such that $s' = s\,.\,c \neq r$.



We swap the labels of $s \xrightarrow{c} s'$ and $s \xrightarrow{a} r$. If r still lies on an a-cycle, then the length of the a-cycle is at least $k_1 + k_2 + 2$ and the number of states on the a-cycles increases. Otherwise, the level of r w.r.t. a becomes at least $k_1 + k_2 + 1 > L$ whence all states of maximal level w.r.t. a in the new automaton are a-ascendants of r and belong to the same tree.

Induction Step: Subcase 2.3

Let q be the state on the a-path from p to r such that $q \, . \, a = r$.

Subcase 2.3: $k_2 = L$ and q is not a bunch. Since q is not a bunch, there is a letter c such that $q' = q \, . \, c \neq r$.

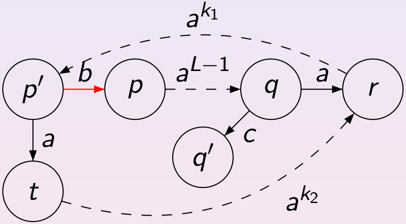
If we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$, we find ourselves in the conditions of Subcase 2.2 (with q and q' playing the roles of s and s' respectively).

Let q be the state on the a-path from p to r such that $q \,.\, a = r$.
Subcase 2.3: $k_2 = L$ **and** q **is not a bunch.** Since q is not a bunch, there is a letter c such that $q' = q \,.\, c \neq r$.

If we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$, we find ourselves in the conditions of Subcase 2.2 (with q and q' playing the roles of s and s' respectively).

Induction Step: Subcase 2.3

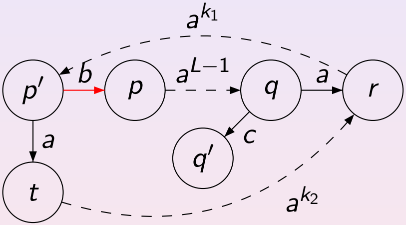
Let q be the state on the a-path from p to r such that $q\,.\,a = r$. **Subcase 2.3: $k_2 = L$ and q is not a bunch.** Since q is not a bunch, there is a letter c such that $q' = q\,.\,c \neq r$.



If we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$, we find ourselves in the conditions of Subcase 2.2 (with q and q' playing the roles of s and s' respectively).

Induction Step: Subcase 2.3

Let q be the state on the a-path from p to r such that $q \, . \, a = r$. **Subcase 2.3:** $k_2 = L$ **and** q **is not a bunch.** Since q is not a bunch, there is a letter c such that $q' = q \, . \, c \neq r$.



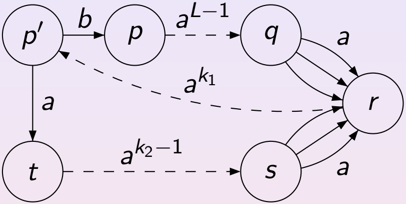
If we swap the labels of $p' \xrightarrow{b} p$ and $p' \xrightarrow{a} t$, we find ourselves in the conditions of Subcase 2.2 (with q and q' playing the roles of s and s' respectively).

Subcase 2.4: $k_2 = L$ and both s and q are bunches.

In this case it is clear that q and s form a stable pair. This completes the proof.

The proof can be 'unfolded' to a quadratic (in $|V|$) algorithm to find a synchronizing coloring of a given admissible digraph $\Gamma = (V, E)$ – Marie-Pierre Béal and Dominique Perrin, 2008.

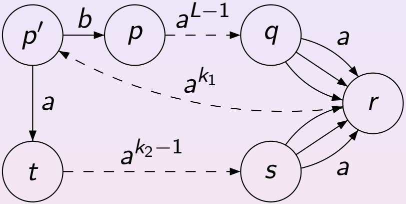
Subcase 2.4: $k_2 = L$ **and both** s **and** q **are bunches.**



In this case it is clear that q and s form a stable pair. This completes the proof.

The proof can be 'unfolded' to a quadratic (in $|V|$) algorithm to find a synchronizing coloring of a given admissible digraph $\Gamma = (V, E)$ – Marie-Pierre Béal and Dominique Perrin, 2008.

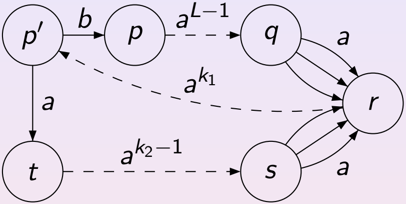
Subcase 2.4: $k_2 = L$ and both s and q are bunches.



In this case it is clear that q and s form a stable pair. This completes the proof.

The proof can be 'unfolded' to a quadratic (in $|V|$) algorithm to find a synchronizing coloring of a given admissible digraph $\Gamma = (V, E)$ – Marie-Pierre Béal and Dominique Perrin, 2008.

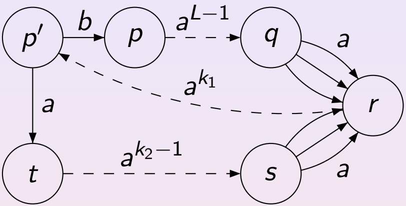
Subcase 2.4: $k_2 = L$ and both s and q are bunches.



In this case it is clear that q and s form a stable pair. This completes the proof.

The proof can be 'unfolded' to a quadratic (in $|V|$) algorithm to find a synchronizing coloring of a given admissible digraph $\Gamma = (V, E)$ – Marie-Pierre Béal and Dominique Perrin, 2008.

Subcase 2.4: $k_2 = L$ **and both** s **and** q **are bunches.**



In this case it is clear that q and s form a stable pair. This completes the proof.

The proof can be 'unfolded' to a quadratic (in $|V|$) algorithm to find a synchronizing coloring of a given admissible digraph $\Gamma = (V, E)$ – Marie-Pierre Béal and Dominique Perrin, 2008.

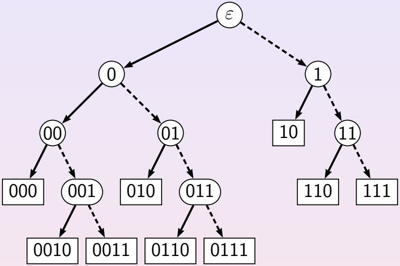
An Application

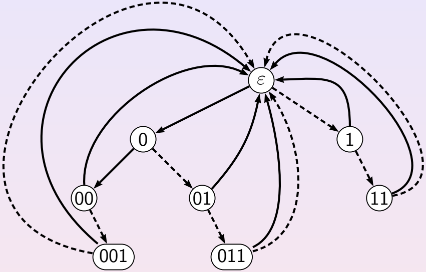
Recall the connection between maximal prefix codes and automata discussed in Lecture 1.

Simple cycles in the automaton on the right correspond to codewords. Thus, if a finite maximal prefix code is such that the g.c.d. of lengths of its codewords is 1, then there exists a synchronized code with the same lengths of codewords (equivalently, with the same tree.) This was proved by Dominique Perrin and Marcel-Paul Schützenberger in 1992.

An Application

Recall the connection between maximal prefix codes and automata discussed in Lecture 1.

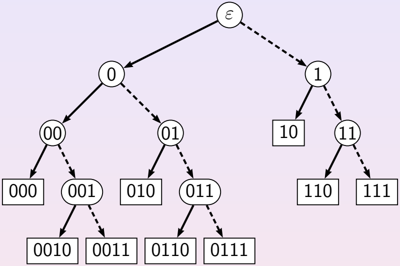


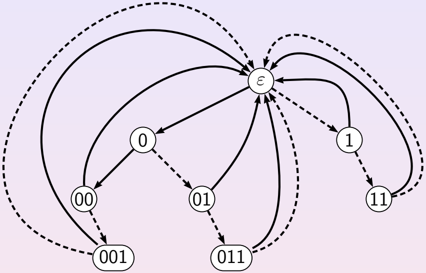


Simple cycles in the automaton on the right correspond to codewords. Thus, if a finite maximal prefix code is such that the g.c.d. of lengths of its codewords is 1, then there exists a synchronized code with the same lengths of codewords (equivalently, with the same tree.) This was proved by Dominique Perrin and Marcel-Paul Schützenberger in 1992.

An Application

Recall the connection between maximal prefix codes and automata discussed in Lecture 1.

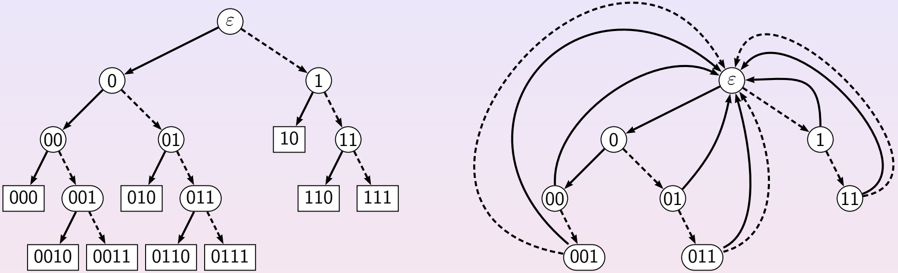




Simple cycles in the automaton on the right correspond to codewords. Thus, if a finite maximal prefix code is such that the g.c.d. of lengths of its codewords is 1, then there exists a synchronized code with the same lengths of codewords (equivalently, with the same tree.) This was proved by Dominique Perrin and Marcel-Paul Schützenberger in 1992.

An Application

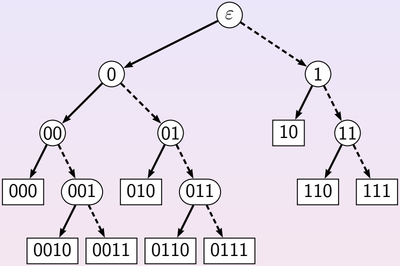
Recall the connection between maximal prefix codes and automata discussed in Lecture 1.

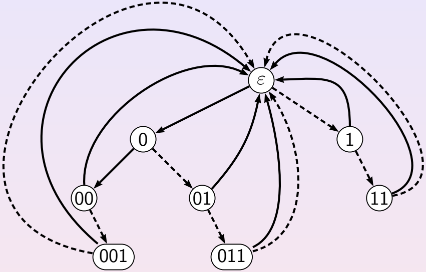


Simple cycles in the automaton on the right correspond to codewords. Thus, if a finite maximal prefix code is such that the g.c.d. of lengths of its codewords is 1, then there exists a synchronized code with the same lengths of codewords (equivalently, with the same tree.) This was proved by Dominique Perrin and Marcel-Paul Schützenberger in 1992.

An Application

Recall the connection between maximal prefix codes and automata discussed in Lecture 1.





Simple cycles in the automaton on the right correspond to codewords. Thus, if a finite maximal prefix code is such that the g.c.d. of lengths of its codewords is 1, then there exists a synchronized code with the same lengths of codewords (equivalently, with the same tree.) This was proved by Dominique Perrin and Marcel-Paul Schützenberger in 1992.