Foundations of Reversibility in Finite-State Devices

Martin Kutrib

Institut für Informatik, Universität Giessen, Germany

Handout for a Course at the International Training School
on Reversible Computation in Toruń
August 2017

Contents

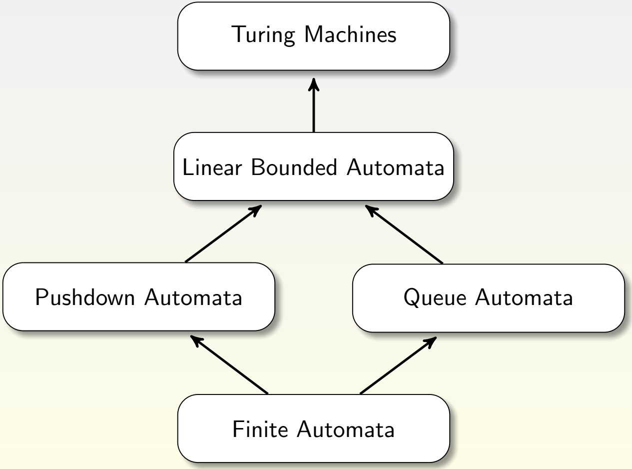
Very Basics on Finite-State Devices

Starting Point

Deterministic devices with a finite number of discrete internal states. The machines have a read-only input tape, may be equipped with further resources, and evolve in discrete time. Given a configuration representing the complete "global state" of a device, the transition function is used to compute the successor configuration.

Very Basics on Finite-State Devices

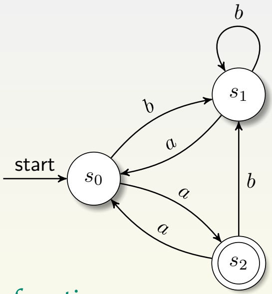
Recall:



Very Basics on Finite-State Devices

Example: deterministic finite automaton (DFA)

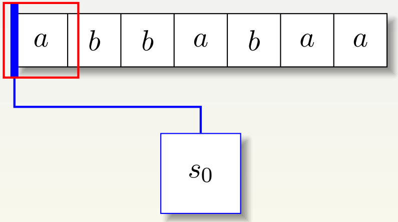
$M = \langle S, \Sigma, \delta, s_0, F \rangle$

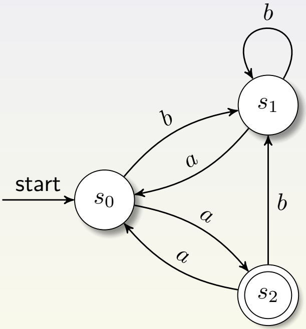


- S is the finite set of internal states
- $s_0 \in S$ is the initial state
- $F \subseteq S$ is the set of accepting states
- Σ is the finite set of input symbols
- $\delta : S \times \Sigma \to S$ is the partial transition function

Very Basics on Finite-State Devices

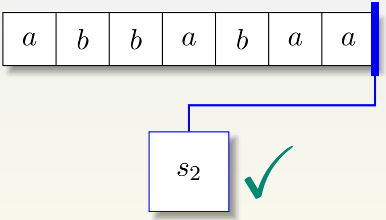
Example: deterministic finite automaton (DFA)

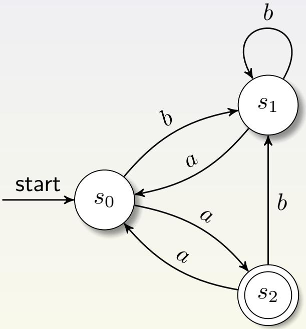




Very Basics on Finite-State Devices

Example: deterministic finite automaton (DFA)





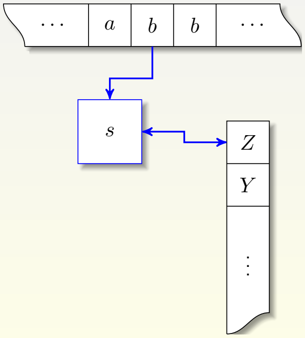
Language accepted by the DFA: the set of all accepted strings.

Very Basics on Finite-State Devices

Example: deterministic pushdown automaton (DPDA)

$$M = \langle S, \Sigma, \Gamma, \delta, s_0, F, \perp \rangle$$

- S is the finite set of internal states
- $s_0 \in S$ is the initial state
- $F \subseteq S$ is the set of accepting states

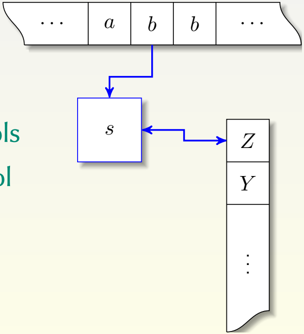


Very Basics on Finite-State Devices

Example: deterministic pushdown automaton (DPDA)

$$M = \langle S, \Sigma, \Gamma, \delta, s_0, F, \perp \rangle$$

- Γ is the finite set of pushdown symbols
- $\perp \notin \Gamma$ is the empty pushdown symbol

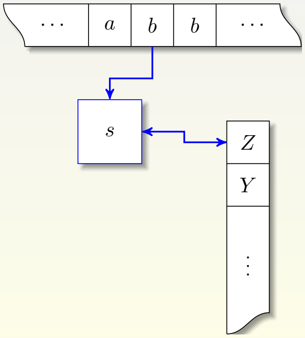


Very Basics on Finite-State Devices

Example: deterministic pushdown automaton (DPDA)

$$M = \langle S, \Sigma, \Gamma, \delta, s_0, F, \perp \rangle$$

➜ Σ is the finite set of input symbols

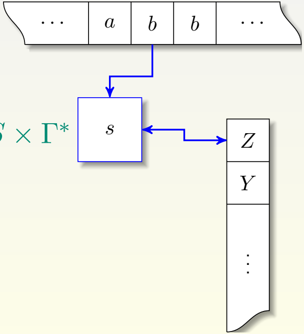


Very Basics on Finite-State Devices

Example: deterministic pushdown automaton (DPDA)

$M = \langle S, \Sigma, \Gamma, \delta, s_0, F, \perp \rangle$

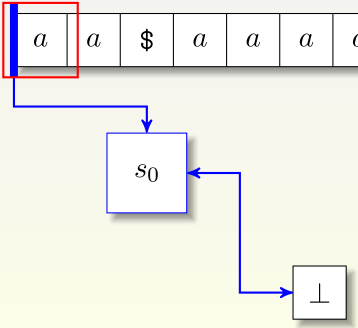
➜ $\delta : S \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\perp\}) \to S \times \Gamma^*$
is the partial transition function



Very Basics on Finite-State Devices

Example: deterministic pushdown automaton (DPDA)

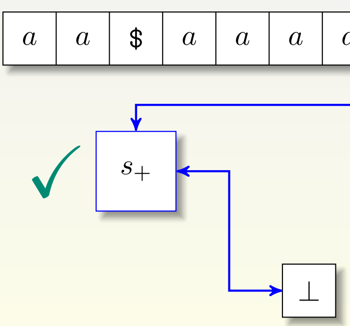
The language $\{\, a^n \$ a^{2n} \mid n \geq 0 \,\}$ is accepted by some DPDA.



Very Basics on Finite-State Devices

Example: deterministic pushdown automaton (DPDA)

The language $\{\, a^n\$a^{2n} \mid n \geq 0 \,\}$ is accepted by some DPDA.



Very Basics on Finite-State Devices

Reversible Computations

- In essence, every configuration has at most one unique successor configuration and at most one unique predecessor configuration.
- Reversibility is meant with respect to the possibility of stepping the computation back and forth.

What Does Reversibility Actually Mean?

What does reversibility mean?

- Basically, the definition of logical reversibility requires that the device is deterministic and that any configuration must have at most one predecessor.

But:

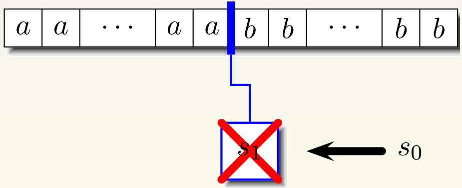
- In which way is the predecessor configuration computed?
- Do we have to consider all possible configurations?

What Does Reversibility Actually Mean?

In which way is the predecessor configuration computed?

- May we use a universal device? Do we have to use a device of the same type? Or else a device with the same computational power?

Example An irreversible DFA accepting the language a^*bb^*.

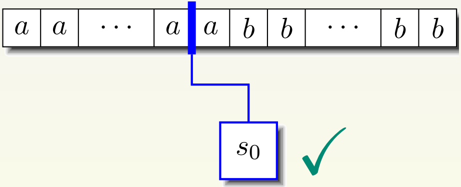


What Does Reversibility Actually Mean?

In which way is the predecessor configuration computed?

➜ May we use a universal device? Do we have to use a device of the same type? Or else a device with the same computational power?

Example An irreversible DFA accepting the language a^*bb^*.



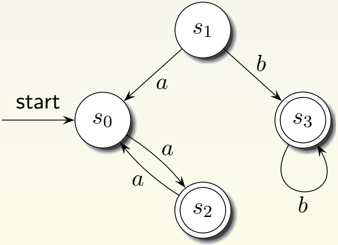
An equivalent DFA with lookahead two.

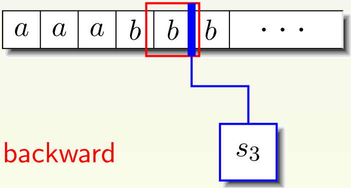
What Does Reversibility Actually Mean?

Do we have to consider all possible configurations?

➜ Or only configurations that are reachable from some initial configurations, that is, configurations that actually occur in computations?

Example A DFA and an unreachable configuration.





(Minimal) Reversible Finite Automata

Recall Minimization:

- A finite automaton is said to be minimal if its number of states is minimal with respect to the accepted language.
- For a given n-state DFA one can efficiently compute an equivalent minimal automaton in $O(n \log n)$ time.
- The minimal DFA accepting a given regular language is unique up to isomorphism.

(Minimal) Reversible Finite Automata

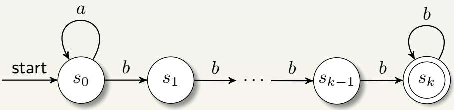
Problems

- Is any regular language accepted by some reversible DFA?
- If not, is it decidable whether a regular language is accepted by a reversible DFA?
- Is the minimal reversible DFA unique?
- Can the minimal reversible DFA be constructed?
- How about the size of a minimal reversible DFA compared with the size of a minimal DFA?

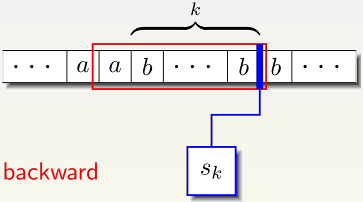
(Minimal) Reversible Finite Automata

Is any regular language accepted by some reversible DFA?

- Languages: $a^*b^kb^*$, $k \geq 1$.



irreversible
for lookahead k



reversible
for lookahead $k+1$

Theorem [Angluin 1982; K., Worsch 2014]

For any $k \geq 1$, $\mathscr{L}(\text{REV}(k+1)\text{-DFA}) \supset \mathscr{L}(\text{REV}(k)\text{-DFA})$.

(Minimal) Reversible Finite Automata

Is any regular language accepted by some reversible DFA with a certain lookahead?

Theorem [Angluin 1982; K., Worsch 2014]

$$\mathrm{REG} \supset \bigcup_{k\geq 1} \mathscr{L}(\mathrm{REV}(k)\text{-DFA}).$$

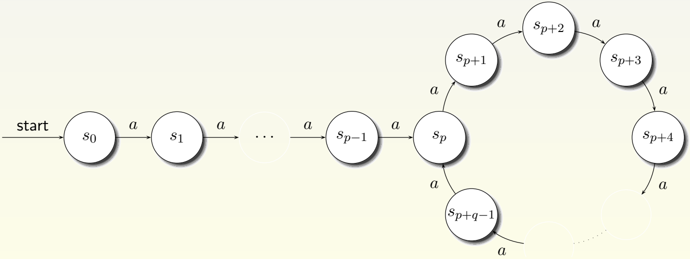
- Let $L \subseteq \Sigma^*$ be a language from $\mathscr{L}(\mathrm{REV}(2)\text{-DFA}) \setminus \mathscr{L}(\mathrm{REV}(1)\text{-DFA})$.
- Define a regular substitution by $s(a) = a\#^*$, for $a \in \Sigma$.
- Language $s(L)$ is regular.
- For any $k \geq 1$, language $s(L)$ contains all words from $s(L) \cap (\Sigma\#^k)^*$.
- If $s(L)$ is accepted by some $\mathrm{REV}(k)$-DFA, it is accepted by some $\mathrm{REV}(1)$-DFA, a contradiction.

(Minimal) Reversible Finite Automata

Is any regular language accepted by some reversible DFA with a certain lookahead?

Theorem **[K., Worsch 2014]**

$$\text{unaryREG} = \bigcup_{k \geq 1} \mathscr{L}(\text{unaryREV}(k)\text{-DFA}).$$



(Minimal) Reversible Finite Automata

Is any regular language accepted by some reversible DFA with a certain lookahead?

Theorem **[K., Worsch 2014]**

$$\text{unaryREG} = \bigcup_{k\geq 1} \mathscr{L}(\text{unaryREV}(k)\text{-DFA}).$$

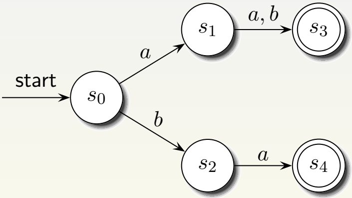
Theorem **[Kondacs, Watrous 1997]**

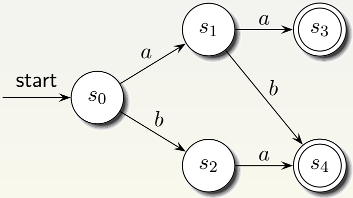
Every regular language is accepted by a reversible two-way DFA.

(Minimal) Reversible Finite Automata

Is the minimal reversible DFA unique?

➜ Language: $\{aa, ab, ba\}$.

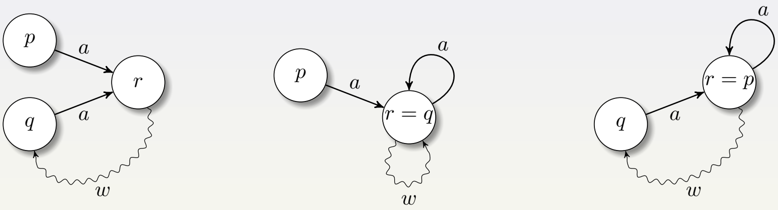




No: Let L be a regular language accepted by some reversible DFA. Then a minimal reversible DFA accepting L is not necessarily unique, even not up to isomorphism.

(Minimal) Reversible Finite Automata

Decidability – Key tool.

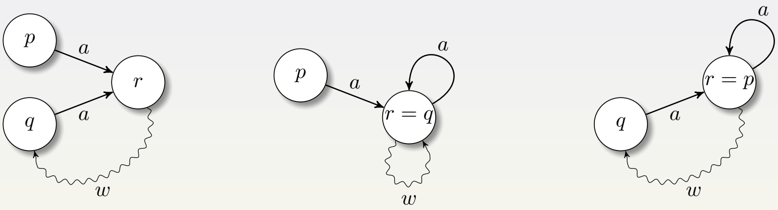


Theorem (Forbidden Patterns) **[Holzer, Jakobi, K. 2015]**

Let $M = \langle S, \Sigma, \delta, s_0, F \rangle$ be a minimal deterministic finite automaton. The language $L(M)$ can be accepted by a reversible DFA if and only if there do not exist useful states $p, q \in S$, a letter $a \in \Sigma$, and a word $w \in \Sigma^*$ such that $p \neq q$, $\delta(p, a) = \delta(q, a)$, and $\delta(q, aw) = q$.

(Minimal) Reversible Finite Automata

Decidability – Key tool.



Example

- Both languages a^*ba^* and b^*ab^* are accepted by reversible DFA,
- but their union is not.

(Minimal) Reversible Finite Automata

Decidability.

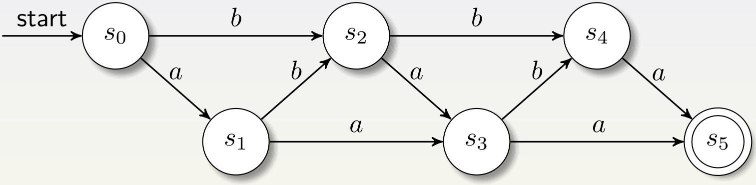
- The problem to decide whether a given DFA is reversible is trivial by inspection of the transition function.

- The regular language reversibility problem, that is, given a DFA M, decide whether $L(M)$ is accepted by any reversible DFA, is NL-complete.

- The problem to decide whether a given DFA is already a minimal reversible DFA is NL-complete.

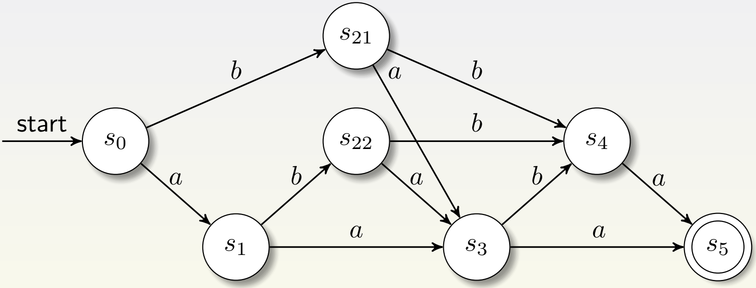
(Minimal) Reversible Finite Automata

Construction of a minimal reversible DFA.



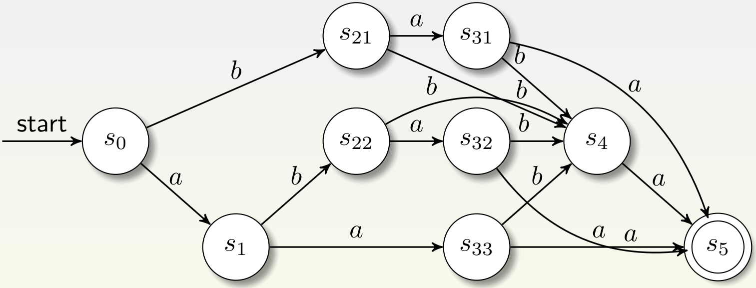
(Minimal) Reversible Finite Automata

Construction of a minimal reversible DFA.



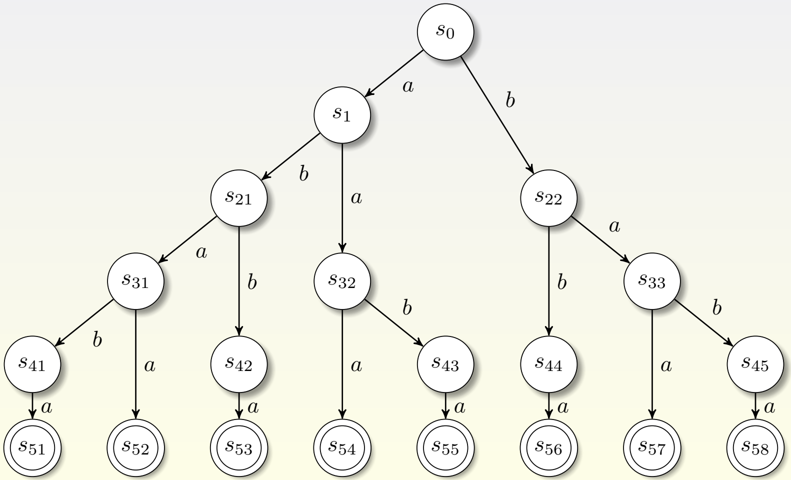
(Minimal) Reversible Finite Automata

Construction of a minimal reversible DFA.



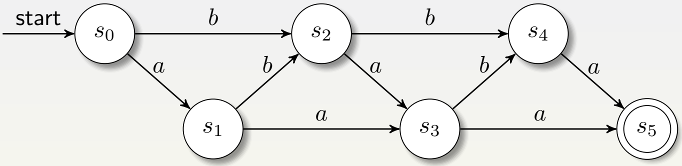
(Minimal) Reversible Finite Automata

Construction of a minimal reversible DFA.



(Minimal) Reversible Finite Automata

Size trade-off between minimal and minimal reversible DFA.



Lower bound:

$$\sum_{i=1}^{n} F_i = F_{n+2} - 1 \in \Omega\left(\frac{1+\sqrt{5}}{2}\right)^n = \Omega(\Phi^n)$$

that is, $\sim 1.618^n$, with $\Phi = (1+\sqrt{5})/2$, the golden ratio.

(Minimal) Reversible Finite Automata

Size trade-off between minimal and minimal reversible DFA.

Lower bound:

$$\sum_{i=1}^{n} F_i = F_{n+2} - 1 \in \Omega\left(\frac{1+\sqrt{5}}{2}\right)^n = \Omega(\Phi^n) \sim 1.618^n.$$

Theorem (Upper Bound) **[Holzer, Jakobi, K. 2015]**

Let M be a minimal n-state DFA that accepts a 'reversible language'. Then a minimal reversible DFA for $L(M)$ has at most 2^{n-1} states.

(Minimal) Reversible Finite Automata

Size trade-off between minimal and minimal reversible DFA.

Theorem (Lower Bound) **[Holzer, Jakobi, K. 2015]**

For every $n \geq 3$ there is an n-state DFA M_n over a k-ary input alphabet accepting a 'reversible language', such that any equivalent reversible DFA needs at least

$$\sum_{i=1}^{n} F_{n-1} + F_{n-2} + \cdots + F_{n-k}$$

states.
For $k = 3$ the lower bound is of order 1.839^n and for $k = 4$ it is of order 1.927^n. For growing alphabet sizes the bound asymptotically tends to 2^{n-1}, that is, $\Omega(2^{n-1})$.

Pushdown Stores and Queues

Reversible pushdown automata (REV-PDA).

- Transition function of the form

$$\delta : S \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\perp\}) \to S \times \Gamma^*$$

- Reverse transition function $\delta^{\leftarrow}$ of the same form.

A DPDA M with transition function δ is said to be reversible (REV-DPDA), if there exists a reverse transition function $\delta^{\leftarrow}$ inducing a relation $\vdash^{\leftarrow}$ from one configuration to the next, so that

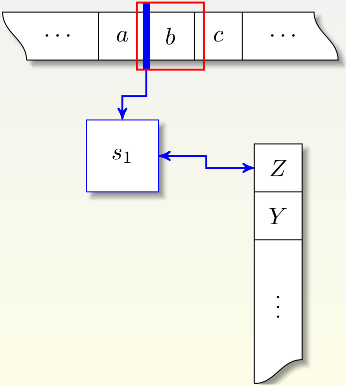
$$c' \vdash^{\leftarrow} c \text{ if and only if } c \vdash c',$$

for any two configurations c, c' of M.

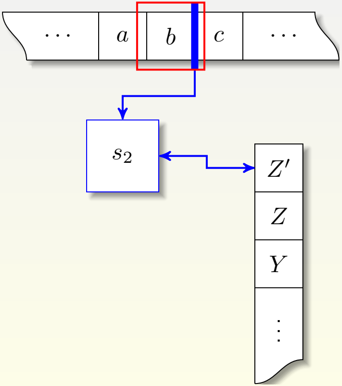
Pushdown Stores and Queues

Example

$\delta(s_1, b, Z) = (s_2, Z'Z)$



$\delta^{\leftarrow}(s_2, b, Z') = (s_1, \lambda)$



Pushdown Stores and Queues

Transition structure of reversible pushdown automata.

push: $\delta(s_1, a, Z) = (s_2, YZ) \implies \delta^{\leftarrow}(s_2, a, Y) = (s_1, \lambda)$

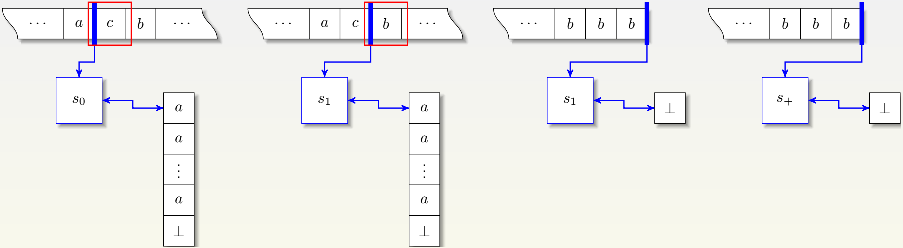
change top: $\delta(s_1, a, Z) = (s_2, Y) \implies \delta^{\leftarrow}(s_2, a, Y) = (s_1, Z)$

pop: $\delta(s_1, a, Z) = (s_2, \lambda) \implies$
for all $X \in \Gamma \cup \{\perp\}$: $\quad \delta^{\leftarrow}(s_2, a, X) = (s_1, ZX)$

Pushdown Stores and Queues

Example A reversible REV-PDA accepting the languages

$$\{\, wcw^R \mid w \in \{a,b\}^+ \,\} \text{ or } \{\, a^n c b^n \mid n \geq 1 \,\}.$$



$$\begin{array}{rcl}
\delta(s_0, a, \perp) & = & (s_0, a\perp) \\
\delta(s_0, a, a) & = & (s_0, aa) \\
\delta(s_0, c, a) & = & (s_1, a) \\
\delta(s_1, b, a) & = & (s_1, \lambda) \\
\delta(s_1, \lambda, \perp) & = & (s_+, \perp)
\end{array}$$

$$\begin{array}{rcl}
\delta^{\leftarrow}(s_0, a, a) & = & (s_0, \lambda) \\
\delta^{\leftarrow}(s_1, c, a) & = & (s_0, a) \\
\delta^{\leftarrow}(s_1, b, a) & = & (s_1, ba) \\
\delta^{\leftarrow}(s_1, b, \perp) & = & (s_1, a\perp) \\
\delta^{\leftarrow}(s_+, \lambda, \perp) & = & (s_1, \perp)
\end{array}$$

Pushdown Stores and Queues

Transitions on empty input.

- For any REV-PDA, the number of consecutive transitions on empty input are bounded by some constant.
- For any REV-PDA an equivalent realtime REV-PDA can effectively be constructed.

Theorem

Every language not accepted by some realtime deterministic push-down automaton is not accepted by REV-PDA.

Example $\{ a^m e b^n c a^m \mid m, n \geq 0 \} \cup \{ a^m e b^n d a^n \mid m, n \geq 0 \}$

Pushdown Stores and Queues

Deterministic realtime pushdown automata:
Can we simulate them reversibly?

Example

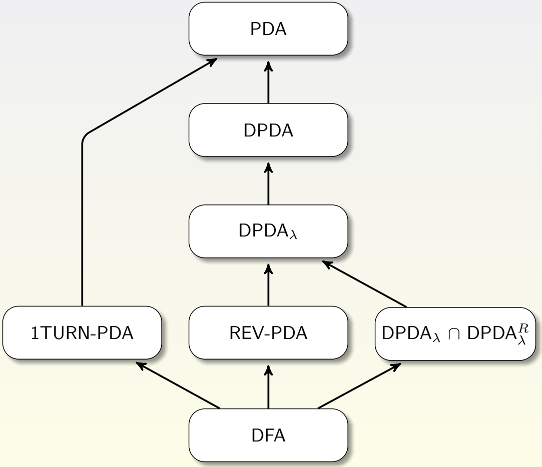
- The language $\{ a^n b^n \mid n \geq 0 \}$ is accepted by some deterministic realtime pushdown automaton.
- It is not accepted by any REV-PDA.

Theorem

There are deterministic realtime pushdown automata that cannot be simulated reversibly.

Pushdown Stores and Queues

Computational capacity.



Pushdown Stores and Queues

Decidability of basic properties.

- Finiteness, infiniteness, emptiness, universality, equivalence, and regularity are decidable for REV-PDA.
- Inclusion is undecidable for REV-PDA.

Pushdown Stores and Queues

Decidability of reversibility – machines.

- The problem to decide whether a given pushdown automaton is reversible on all configurations is trivial by inspection of the transition function.

- The problem to decide whether a given pushdown automaton is reversible on all reachable configurations is P-complete.

Pushdown Stores and Queues

Decidability of reversibility – languages.

- ➜ It is undecidable whether the language accepted by a nondeterministic pushdown automaton can be accepted by a REV-PDA.
- ➜ The same problem for deterministic pushdown automata is open.

Pushdown Stores and Queues

Reachable versus unreachable configurations.

Theorem

Given a pushdown automaton that is reversible on all reachable configurations, an equivalent pushdown automaton that is reversible on all configurations can effectively be constructed.

Pushdown Stores and Queues

Reversible queue automata (REV-QA).

- Transition function of the form

$$\delta : S \times (\Sigma \cup \{\lambda\}) \times ((\Gamma \times \Gamma) \cup (\{\bot\} \times \{\bot\})) \to S \times (\Gamma \cup \{\lambda\}) \times \{\texttt{keep}, \texttt{remove}\}$$

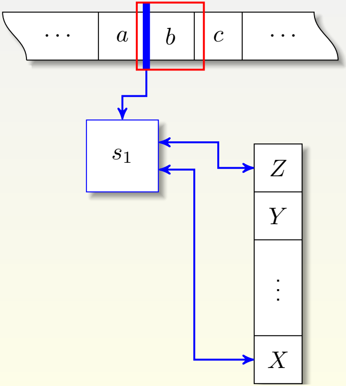
- Reverse transition function $\delta^{\leftarrow}$ of the same form.

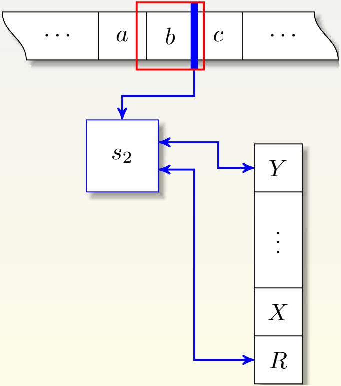
Pushdown Stores and Queues

Example

$\delta(s_1, b, Z, X) = (s_2, R, \texttt{remove})$

$\delta^{\leftarrow}(s_2, b, Y, R) = (s_1, Z, \texttt{remove})$





Pushdown Stores and Queues

Computational capacity of general queue automata.

Theorem

Given a deterministic queue automaton, an equivalent reversible queue automaton can effectively be constructed.

Theorem

Every recursively enumerable language is accepted by some reversible queue automaton.

Pushdown Stores and Queues

Quasi realtime reversible queue automata.

A DQA is said to work in quasi realtime, if the number of consecutive λ-transitions is bounded by a constant.

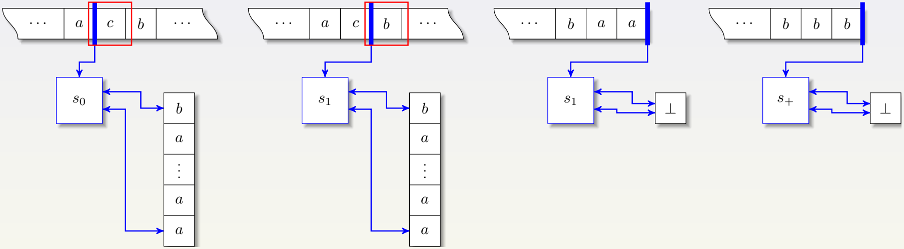
Theorem

For every quasi realtime reversible DQA an equivalent realtime reversible DQA can effectively be constructed.

➜ It is sufficient and natural to consider realtime reversible DQA.

Pushdown Stores and Queues

Example A reversible REV-QA accepting $\{\, wcw \mid w \in \{a,b\}^+ \,\}$.



$$
\begin{aligned}
\delta(s_0, x, \bot, \bot) &= (s_0, x, \texttt{keep})\\
\delta(s_0, x, y, z) &= (s_0, x, \texttt{keep})\\
\delta(s_0, c, y, z) &= (s_1, \lambda, \texttt{keep})\\
\delta(s_1, x, x, y) &= (s_1, \lambda, \texttt{remove})\\
\delta(s_1, \lambda, \bot, \bot) &= (s_+, \lambda, \texttt{keep})
\end{aligned}
$$

$$
\begin{aligned}
\delta^{\leftarrow}(s_0, x, y, z) &= (s_0, \lambda, \texttt{remove})\\
\delta^{\leftarrow}(s_1, c, y, z) &= (s_0, \lambda, \texttt{remove})\\
\delta^{\leftarrow}(s_1, x, y, z) &= (s_1, x, \texttt{keep})\\
\delta^{\leftarrow}(s_1, x, \bot, \bot) &= (s_1, x, \texttt{keep})\\
\delta^{\leftarrow}(s_+, \lambda, \bot, \bot) &= (s_1, \lambda, \texttt{keep})
\end{aligned}
$$

$$x, y, z \in \{a, b\}$$

Pushdown Stores and Queues

Deterministic realtime queue automata:
Can we simulate them reversibly?

Witness $L_{bin} = ((aa + a)(bb + b))^+$.

$$L_{mcp} = \{\, p\$w_1\$w_1\$w_2\$w_2\$ \cdots \$w_n\$w_n \mid p \in L_{bin}, n \geq 0, w_i \in \{a, b\}^* \,\}$$

Example

- The language L_{mcp} is accepted by some deterministic realtime queue automaton.
- It is not accepted by any realtime REV-QA.

Theorem

There are deterministic realtime queue automata that cannot be simulated reversibly.

Pushdown Stores and Queues

Reversible pushdown automata versus realtime reversible queue automata.

- The mirror language $\{ wcw^R \mid w \in \{a, b\}^+ \}$ is accepted by a reversible pushdown automaton, but not by any even irreversible quasi realtime queue automaton.
- The non-context-free copy language $\{ wcw \mid w \in \{a, b\}^+ \}$ is accepted by a reversible realtime queue automaton.

Theorem

The families of languages accepted by realtime reversible queue automata and by reversible pushdown automata are incomparable.

Pushdown Stores and Queues

Decidability of basic properties.

➜ Finiteness, infiniteness, emptiness, universality, equivalence, inclusion, and regularity are not semidecidable for realtime REV-QA.

Decidability of reversibility – machines.

- The problem to decide whether a given realtime queue automaton is reversible on all configurations is trivial by inspection of the transition function.
- The problem to decide whether a given realtime queue automaton is reversible on all reachable configurations is not semidecidable.

Pushdown Stores and Queues

Reachable versus unreachable configurations.

Theorem

Given a realtime queue automaton that is reversible on all reachable configurations, there exists an equivalent realtime queue automaton that is reversible on all configurations.

Time Symmetry

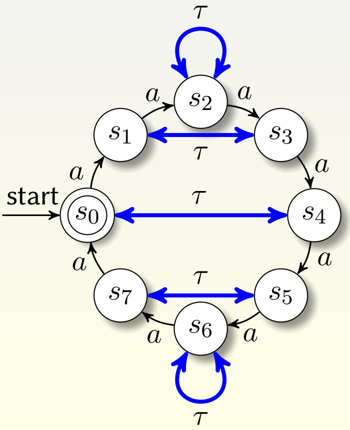
- For example, in Newtonian mechanics, relativity, or quantum mechanics one can go back and forth in time by applying the same dynamics,
- provided that the sense of time direction is changed by a specific transformation of the phase-space.
- For Newtonian mechanics, the transformation leaves masses and positions unchanged but reverses the sign of the momenta.
- Here we consider weak transformations τ, that is, involutions having the property $\tau \circ \tau = \mathrm{id}$.
- Time-symmetric machines themselves cannot distinguish whether they run forward or backward in time.

Time Symmetry

Example A reversible DFA is time symmetric if and only if there is an involution τ that maps states to states such that

$$\delta_x^{-1} = \tau \circ \delta_x \circ \tau$$

holds for all input symbols x, where δ_x is the transition function for symbol x.

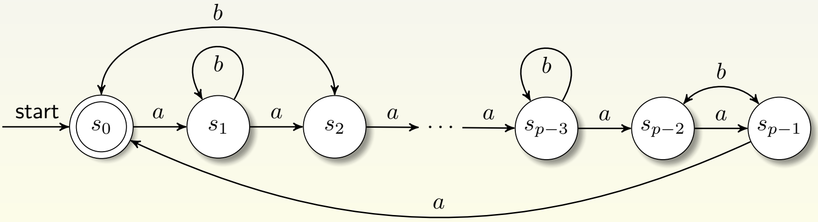


Time Symmetry

Theorem

There are infinitely many reversible DFA which are not time symmetric.

Example A reversible DFA that is not time symmetric.

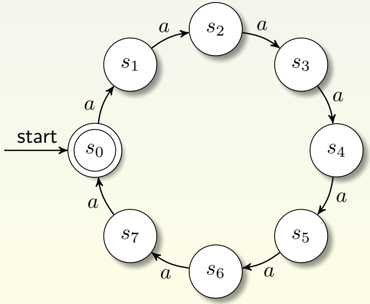


Time Symmetry

Theorem

Each reversible unary DFA is time symmetric.

Example Choose two arbitrary states s_i and s_j and set $\tau(s_i) = s_j$.



Time Symmetry

Theorem

Any reversible DFA can be simulated by some time-symmetric DFA.

Time Symmetry

Time-symmetric pushdown automata.

- A reversible REV-PDA is time symmetric if and only if there is an involution τ that maps states and stack symbols to states and stack symbols such that

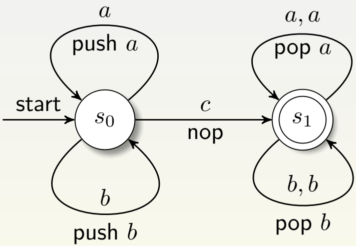
$$\hat{\delta}_x^{-1} = \tau \circ \hat{\delta}_x \circ \tau$$

  holds for all input symbols x, where $\hat{\delta}_x$ is the transition function for symbol x.

Time Symmetry

Example A time-symmetric pushdown automaton accepting the linear context-free language

$$\{\, wcv \mid w \in \{a,b\}^*, w^R = vu, 0 \le |u| \le |w| \,\}.$$



➜ Set $\tau(s_0, \gamma) = (s_1, \gamma)$, for all $\gamma \in \Gamma^*$.

$$(\tau \circ \hat{\delta}_b \circ \tau)(s_0, b\gamma) = (\tau \circ \hat{\delta}_b)(s_1, b\gamma) =$$

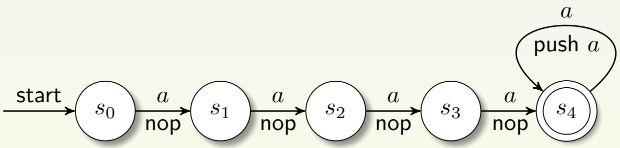
$$\tau(s_1, \gamma) = (s_0, \gamma) = \hat{\delta}_b^{-1}(s_0, b\gamma)$$

Time Symmetry

Theorem

There are infinitely many reversible REV-PDA which are not time symmetric.

Example A reversible but not time-symmetric REV-PDA accepting the language a^4a^*.



Theorem

Any reversible REV-PDA can be simulated by some time-symmetric REV-PDA.