RFSA via Functors

Quentin Schroeder

Université Paris Cité, France

Word automata

deterministic automata $\quad 1 \longrightarrow Q \longrightarrow 2 \quad$ in Set

$$\begin{tikzcd} 1 \arrow[r] & Q \arrow[loop above, "a"] \arrow[r] & 2 \end{tikzcd}$$

Word automata

deterministic automata
$$\begin{tikzcd} 1 \arrow[r] & Q \arrow[loop above, "a"] \arrow[r] & 2 \end{tikzcd}$$
in Set

non-deterministic automata
$$\begin{tikzcd} 1 \arrow[r, "\shortmid" marking] & Q \arrow[loop above, "a", "\shortmid" marking] \arrow[r, "\shortmid" marking] & 1 \end{tikzcd}$$
in Rel

Word automata

deterministic automata
$$\begin{tikzcd} 1 \arrow[r] & Q \arrow[r] \arrow["a", loop above] & 2 \end{tikzcd}$$
in Set

non-deterministic automata
$$\begin{tikzcd} 1 \arrow[r, "|" marking] & Q \arrow[r, "|" marking] \arrow["a", loop above, "|" marking] & 1 \end{tikzcd}$$
in Rel

Idea:

- Automata = Functors
- Minimization works once the ouput category is well behaved!
  (Colcombet and Petrişan,2021)

The problem with minimization of NFAs

- Known problem: no unique minimal NFA

The problem with minimization of NFAs

- Known problem: no unique minimal NFA
- Functorial framework explains this: Rel is not well behaved!

The problem with minimization of NFAs

- Known problem: no unique minimal NFA
- Functorial framework explains this: Rel is not well behaved!
- Several approaches exist: we consider the one of Denis et al.

Word automata as functors

Word automata on A^* can be seen by interpreting the vertices and edges in the graph below in some output category $\mathcal{C}$ of "effects".

$$\begin{tikzcd} in \arrow[r, "\triangleright"] & st \arrow[loop above, "a(a \in A)"] \arrow[r, "\triangleleft"] & out \end{tikzcd}$$

Word automata as functors

Word automata on A^* can be seen by interpreting the vertices and edges in the graph below in some output category $\mathcal{C}$ of "effects".

$$\begin{tikzcd} in \arrow[r, "\triangleright"] & st \arrow[loop above, "a (a \in A)"] \arrow[r, "\triangleleft"] & out \end{tikzcd}$$

This amounts to defining a functor – a composition preserving mapping – from the free category $\mathcal{I}$ to $\mathcal{C}$.

Automata and languages as functors

An automaton $\mathcal{A}$ accepts a language $\mathcal{L}$ when the next diagram commutes

$$
\begin{tikzcd}
in \arrow[rr, "\triangleright w \triangleleft \,:\, w \in A^*"] & & out \\
in \arrow[r, "\triangleright"] & st \arrow[loop above, "a"] \arrow[r, "\triangleleft"] & out
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\mathcal{O} \arrow[r, "\mathcal{L}"] \arrow[d, hook] & \mathcal{C} \\
\mathcal{I} \arrow[ur, "\mathcal{A}"'] &
\end{tikzcd}
$$

Automata and languages as functors

An automaton $\mathcal{A}$ accepts a language $\mathcal{L}$ when the next diagram commutes

$$\begin{tikzcd}
in \arrow[rr, "\triangleright w \triangleleft \,:\, w \in A^*"] & & out \\
in \arrow[r, "\triangleright"] & st \arrow[r, "\triangleleft"] \arrow[loop above, "a"] & out
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathcal{O} \arrow[r, "\mathcal{L}"] \arrow[d, hook] & \mathcal{C} \\
\mathcal{I} \arrow[ur, "\mathcal{A}"'] &
\end{tikzcd}$$

For every language $\mathcal{L} \colon \mathcal{O} \to \mathcal{C}$ we consider a category $\mathrm{Auto}_{\mathcal{L}}$ of automata accepting $\mathcal{L}$.

$\mathcal{O}$ can be seen as an “observation” subcategory of $\mathcal{I}$.
$\mathcal{I}$ is like a “template” for the internal state of the automaton.

Functorial Minimization

Minimization of $\mathcal{C}$-automata

- What does it mean for a DFA to be minimal?

Minimization of $\mathcal{C}$-automata

- Answer 1: DFA is minimal if it is state minimal

Minimization of $\mathcal{C}$-automata

- Answer 2: A DFA is minimal if it divides any other automaton accepting its language. Here divides $=$ «is a quotient of a sub-automaton of»

Minimization of $\mathcal{C}$-automata

- Answer 2: A DFA is minimal if it divides any other automaton accepting its language. Here divides $=$ «is a quotient of a sub-automaton of»
- To generalize this we need a factorization system, for which we need a notion of quotient (think: surjection) and of sub-object (think: injection)

Sufficient condition for minimization

When does a ‘minimal’ automaton accepting a language $\mathcal{L}$ exist?

$$\begin{tikzcd} \mathcal{O} \arrow[r, "\mathcal{L}"] \arrow[d, hook] & \mathcal{C} \\ \mathcal{I} \arrow[ur, dashed, "\mathtt{Min}(\mathcal{L})?" description] & \end{tikzcd}$$

Sufficient condition for minimization

When does a 'minimal' automaton accepting a language $\mathcal{L}$ exist?

$$\begin{tikzcd}
\mathcal{O} \arrow[rr, "\mathcal{L}"] \arrow[dd, hook] & & \mathcal{C} \\
& \mathcal{A}_{\mathtt{init}}(\mathcal{L}) \arrow[ur] & \\
\mathcal{I} \arrow[ur] & &
\end{tikzcd}$$

If the category of automata accepting $\mathcal{L}$ has

- an initial object $\mathcal{A}_{\mathtt{init}}(\mathcal{L})$,

Sufficient condition for minimization

When does a ‘minimal’ automaton accepting a language $\mathcal{L}$ exist?

$$\begin{tikzcd}
\mathcal{O} \arrow[rr, "\mathcal{L}"] \arrow[dd, hook] & & \mathcal{C} \\
& \mathcal{A}_{\mathtt{init}}(\mathcal{L}) \arrow[ur] & \\
\mathcal{I} \arrow[ur] \arrow[r] & \mathcal{A}_{\mathtt{final}}(\mathcal{L}) \arrow[uur] &
\end{tikzcd}$$

If the category of automata accepting $\mathcal{L}$ has

- an initial object $\mathcal{A}_{\mathtt{init}}(\mathcal{L})$,
- a final object $\mathcal{A}_{\mathtt{final}}(\mathcal{L})$, and,

Sufficient condition for minimization

When does a ‘minimal’ automaton accepting a language $\mathcal{L}$ exist?

$$\begin{tikzcd}
\mathcal{O} \arrow[r, "\mathcal{L}"] \arrow[d, hook] & \mathcal{C} \\
\mathcal{I} \arrow[ur, bend left=40, "\mathcal{A}_{\mathtt{init}}(\mathcal{L})"] \arrow[ur, dashed, "\mathtt{Min}(\mathcal{L})"] \arrow[ur, bend right=40, "\mathcal{A}_{\mathtt{final}}(\mathcal{L})"'] &
\end{tikzcd}$$

If the category of automata accepting $\mathcal{L}$ has

- an initial object $\mathcal{A}_{\mathtt{init}}(\mathcal{L})$,
- a final object $\mathcal{A}_{\mathtt{final}}(\mathcal{L})$, and,
- a factorization system

then $\mathtt{Min}(\mathcal{L})$ is obtained as the factorization

$$\mathcal{A}_{\mathtt{init}}(\mathcal{L}) \twoheadrightarrow \mathtt{Min}(\mathcal{L}) \rightarrowtail \mathcal{A}_{\mathtt{final}}(\mathcal{L})\,.$$

Sufficient condition for minimization

When does a ‘minimal’ automaton accepting a language $\mathcal{L}$ exist?

$$\begin{tikzcd}
\mathcal{O} \arrow[r, "\mathcal{L}"] \arrow[d, hook] & \mathcal{C} \\
\mathcal{I} \arrow[ur, bend left=40, "\mathcal{A}_{\mathtt{init}}(\mathcal{L})"] \arrow[ur, dashed, "\mathtt{Min}(\mathcal{L})" description] \arrow[ur, bend right=40, "\mathcal{A}_{\mathtt{final}}(\mathcal{L})"'] &
\end{tikzcd}$$

If the category of automata accepting $\mathcal{L}$ has

- an initial object $\mathcal{A}_{\mathtt{init}}(\mathcal{L})$, ✓ when $\mathcal{C}$ has copowers
- a final object $\mathcal{A}_{\mathtt{final}}(\mathcal{L})$, and, ✓ when $\mathcal{C}$ has powers
- a factorization system ✓ when $\mathcal{C}$ has one

then $\mathtt{Min}(\mathcal{L})$ is obtained as the factorization

$$\mathcal{A}_{\mathtt{init}}(\mathcal{L}) \twoheadrightarrow \mathtt{Min}(\mathcal{L}) \rightarrowtail \mathcal{A}_{\mathtt{final}}(\mathcal{L})\,.$$

Example: minimizing DFAs

The initial automaton $\mathcal{A}_{\mathtt{init}}$ for Set-automata accepting a language L is the following :

$$\begin{tikzcd} 1 \arrow[r, "\varepsilon"] & A^* \arrow[loop above, "w \mapsto wa"] \arrow[r, "L?"] & 2 \end{tikzcd}$$

Example: minimizing DFAs

The initial automaton $\mathcal{A}_{\texttt{init}}$ for Set-automata accepting a language L is the following :

$$\begin{tikzcd} 1 \arrow[r, "\varepsilon"] & A^* \arrow[loop above, "w \mapsto wa"] \arrow[r, "L?"] & 2 \end{tikzcd}$$

The final automaton $\mathcal{A}_{\texttt{final}}$ for Set-automata accepting a language L is the following :

$$\begin{tikzcd} 1 \arrow[r, "L"] & 2^{A^*} \arrow[loop above, "K \mapsto a^{-1}K"] \arrow[r, "\varepsilon?"] & 2 \end{tikzcd}$$

Example: minimizing DFAs

The unique map from the initial to the final automaton is given by $! \colon A^* \to 2^{A^*}$, defined by $w \mapsto w^{-1}L$.

$$
\begin{tikzcd}
 & A^* \arrow[dr, bend left, "\mathcal{L}?"] \arrow[dd, dotted, "!"] & \\
1 \arrow[ur, bend left, "\varepsilon"] \arrow[dr, bend right, "\mathcal{L}"'] & & 2 \\
 & 2^{A^*} \arrow[ur, bend right, "\varepsilon?"'] &
\end{tikzcd}
$$

Example: minimizing DFAs

The unique map from the initial to the final automaton is given by $!\colon A^* \to 2^{A^*}$, defined by $w \mapsto w^{-1}L$.

$$
\begin{tikzcd}
 & A^* \arrow[d, two heads, "reachablestates"] \arrow[dr, bend left, "\mathcal{L}?"] & \\
1 \arrow[ur, bend left, "\varepsilon"] \arrow[dr, bend right, "\mathcal{L}"'] & \mathtt{Min}(\mathcal{L}) \arrow[d, tail, "\mathsf{acceptedLanguage}"] & 2 \\
 & 2^{A^*} \arrow[ur, bend right, "\varepsilon?"'] &
\end{tikzcd}
$$

Example: minimizing DFAs

The unique map from the initial to the final automaton is given by $! \colon A^* \to 2^{A^*}$, defined by $w \mapsto w^{-1}L$.

$$\begin{tikzcd}
 & A^* \arrow[d, two heads, "\mathit{reachablestates}" description] \arrow[dr, bend left, "\mathcal{L}?"] & \\
1 \arrow[ur, bend left, "\varepsilon"] \arrow[r, "\mathcal{L}"] \arrow[dr, bend right, "\mathcal{L}"'] & \mathtt{Min}(\mathcal{L}) \arrow[r, "\varepsilon?"] \arrow[d, tail, "\text{acceptedLanguage}" description] & 2 \\
 & 2^{A^*} \arrow[ur, bend right, "\varepsilon?"'] &
\end{tikzcd}$$

Non-Example: minimizing NFAs

The initial automaton $\mathcal{A}_{\mathtt{init}}$ for Rel-automata accepting a language L is the following :

$$\begin{tikzcd} 1 \arrow[r, "\varepsilon", "|" marking] & A^* \arrow[loop above, "\mathbf{w},\mathbf{wa}", "|" marking] \arrow[r, "L", "|" marking] & 1 \end{tikzcd}$$

Non-Example: minimizing NFAs

The initial automaton $\mathcal{A}_{\mathtt{init}}$ for Rel-automata accepting a language L is the following :

$$\begin{tikzcd} 1 \arrow[r, "\varepsilon", "|"'] & A^* \arrow[loop above, "\mathbf{w},\mathbf{wa}", "|"'] \arrow[r, "L", "|"'] & 1 \end{tikzcd}$$

The final automaton $\mathcal{A}_{\mathtt{final}}$ for Rel-automata accepting a language L is the following:

$$\begin{tikzcd} 1 \arrow[r, "L", "|"'] & A^* \arrow[loop above, "\mathbf{aw},\mathbf{w}", "|"'] \arrow[r, "\varepsilon?", "|"'] & 1 \end{tikzcd}$$

Non-example: minimizing NFAs

The unique map from the initial to the final automaton is given by $! \colon A^* \to A^*$, defined by (w, v) `iff` $v \in w^{-1}L$.

$$\begin{tikzcd}
 & A^* \arrow[d, "\text{reachedState}"] \arrow[dr, bend left, "\mathcal{L}"] & \\
1 \arrow[ur, bend left, "\varepsilon"] \arrow[r, "i"] \arrow[dr, bend right, "\mathcal{L}"'] & ??? \arrow[r, "f"] \arrow[d, "\text{acceptedLanguage}"] & 1 \\
 & A^* \arrow[ur, bend right, "\varepsilon"'] &
\end{tikzcd}$$

Strategy for minimization

$\mathcal{A}$

$$\mathsf{NFA} \xrightarrow{\textit{model}} \mathsf{Rel}$$

Strategy for minimization

$$\begin{tikzcd}
\mathcal{A} \arrow[d, mapsto] \\
\mathcal{A}^{\mathcal{C}}
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\text{NFA} \arrow[r, dotted, no head, "\textit{model}"] \arrow[d, tail, "\textit{embed}"'] & \text{Rel} \arrow[d, tail, "\textit{embed}"'] \\
\mathcal{C}\text{-auto} \arrow[r, dotted, no head] & \mathcal{C}
\end{tikzcd}$$

Strategy for minimization

$$
\begin{tikzcd}
\mathcal{A} \arrow[d, mapsto] \\
\mathcal{A}^{\mathcal{C}} \arrow[d, mapsto] \\
\mathtt{Min}_{\mathcal{C}\text{-auto}}(\mathcal{A}^{\mathcal{C}})
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\mathsf{NFA} \arrow[r, dotted, no head, "\textit{model}"] \arrow[d, tail, "\textit{embed}"'] & \mathsf{Rel} \arrow[d, tail, "\textit{embed}"'] \\
\mathcal{C}\text{-auto} \arrow[r, dotted, no head] \arrow[d, "\textit{min}"'] & \mathcal{C} \\
\mathcal{C}\text{-auto} \arrow[r, dotted, no head] & \mathcal{C}
\end{tikzcd}
$$

Strategy for minimization

$$\begin{tikzcd}
\mathcal{A} \arrow[d, mapsto] \\
\mathcal{A}^{\mathcal{C}} \arrow[d, mapsto] \\
\mathtt{Min}_{\mathcal{C}\text{-auto}}(\mathcal{A}^{\mathcal{C}}) \arrow[d, mapsto] \\
\mathit{convert}(\mathtt{Min}_{\mathcal{C}\text{-auto}}(\mathcal{A}^{\mathcal{C}}))
\end{tikzcd}
\qquad
\begin{tikzcd}
\text{NFA} \arrow[r, dotted, no head, "\mathit{model}"] \arrow[d, tail, "\mathit{embed}"'] & \text{Rel} \arrow[d, tail, "\mathit{embed}"'] \\
\mathcal{C}\text{-auto} \arrow[r, dotted, no head] \arrow[d, "\mathit{min}"'] & \mathcal{C} \\
\mathcal{C}\text{-auto} \arrow[r, dotted, no head] \arrow[d, "\mathit{convert}"'] & \mathcal{C} \arrow[d, "\mathit{convert}"'] \\
\text{NFA} \arrow[r, dotted, no head] & \text{Rel}
\end{tikzcd}$$

The right choice for $\mathcal{C}$

- The category of (complete) join-semi lattices JSL, i.e. ordered sets where each subset has a least upper bound.

The right choice for $\mathcal{C}$

- The category of (complete) join-semi lattices JSL, i.e. ordered sets where each subset has a least upper bound.
- Example: For a set S, $(\mathcal{P}(S), \subseteq, \bigcup)$ is a JSL.

The right choice for $\mathcal{C}$

- The category of (complete) join-semi lattices JSL, i.e. ordered sets where each subset has a least upper bound.
- Example: For a set S, $(\mathcal{P}(S), \subseteq, \bigcup)$ is a JSL.
- Embedding:

$$\begin{aligned}\mathcal{K} : \mathrm{Rel} &\to \mathrm{JSL} \\ X &\mapsto \mathcal{P}(X) \\ (R : X \nrightarrow Y) &\mapsto R[-] : \mathcal{P}(X) \to \mathcal{P}(Y)\end{aligned}$$

Strategy for minimization

$$
\begin{tikzcd}
A \arrow[d, mapsto] \\
A^{\mathcal{C}} \arrow[d, mapsto] \\
\mathtt{Min}_{\mathcal{C}\text{-auto}}(A^{\mathcal{C}})
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\mathsf{NFA} \arrow[r, dotted, no head, "\textit{model}"] \arrow[d, tail, "\textit{embed}"'] & \mathsf{Rel} \arrow[d, tail, "\textit{embed}"'] \\
\mathcal{C}\text{-auto} \arrow[r, dotted, no head] \arrow[d, "\textit{min}"'] & \mathcal{C} \\
\mathcal{C}\text{-auto} \arrow[r, dotted, no head] & \mathcal{C}
\end{tikzcd}
$$

Strategy for minimization

$$
\begin{tikzcd}
A \arrow[d, mapsto] \\
A^{det} \arrow[d, mapsto] \\
\mathtt{Min}_{\mathsf{JSLA}}(A^{det})
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\mathsf{NFA} \arrow[r, dotted, no head, "model"] \arrow[d, tail, "det_{\mathsf{JSL}}"'] & \mathsf{Rel} \arrow[d, tail, "\mathcal{K}"'] \\
\mathsf{JSLA} \arrow[r, dotted, no head] \arrow[d, "min"'] & \mathsf{JSL} \\
\mathsf{JSLA} \arrow[r, dotted, no head] & \mathsf{JSL}
\end{tikzcd}
$$

Minimizing join-semi lattice automata accepting L

The initial automaton $\mathcal{A}_{\mathtt{init}}$ for JSL-automata (JSLA) accepting a language L is the following :

$$
\begin{tikzcd}
2 \arrow[r, "\{\varepsilon\}"] & 2^{A^*} \arrow[loop above, "S \mapsto Sa"] \arrow[r, "L \cap ?"] & 2
\end{tikzcd}
$$

Minimizing join-semi lattice automata accepting L

The initial automaton $\mathcal{A}_{\mathtt{init}}$ for JSL-automata (JSLA) accepting a language L is the following :

$$\begin{tikzcd} 2 \arrow[r, "\{\varepsilon\}"] & 2^{A^*} \arrow[loop above, "S \mapsto Sa"] \arrow[r, "L \cap ?"] & 2 \end{tikzcd}$$

The final automaton $\mathcal{A}_{\mathtt{final}}$ for JSL-automata accepting a language L is the following:

$$\begin{tikzcd} 2 \arrow[r, "L"] & 2^{A^*} \arrow[loop above, "S \mapsto a^{-1}S"] \arrow[r, "\varepsilon ?"] & 2 \end{tikzcd}$$

Minimizing join-semi lattice automata accepting L

The unique map from the initial to the final automaton is given by $! \colon 2^{A^*} \to 2^{A^*}$, defined by $S \mapsto \bigcup_{w \in S} w^{-1}L$.

$$
\begin{tikzcd}
 & 2^{A^*} \arrow[dr, bend left, "\mathcal{L}\cap ?"] \arrow[d, two heads, "\text{reachable}" description] & \\
2 \arrow[ur, bend left, "\{\varepsilon\}"] \arrow[r, "i"] \arrow[dr, bend right, "\mathcal{L}"'] & \mathtt{Min}(\mathcal{L}) \arrow[r, "f"] \arrow[d, tail, "\text{accepted}" description] & 2 \\
 & 2^{A^*} \arrow[ur, bend right, "\varepsilon ?"'] &
\end{tikzcd}
$$

Going back to Rel

Note: $Q_{min} := \mathtt{Min}_{JSL}(\mathcal{L})$ consists unions of sets $w^{-1}L$.

- Reduce a join-semi lattice to its basic components: the join irreducible elements.

$$
\begin{tikzcd}
\mathcal{A} \arrow[d, mapsto] \\
\mathcal{A}^{det} \arrow[d, mapsto] \\
min_{\mathsf{JSLA}}(\mathcal{A}^{det}) \arrow[d, mapsto] \\
\mathcal{J}^{gen}(min_{\mathsf{JSLA}}(\mathcal{A}^{\mathsf{JSL}}))
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathsf{NFA} \arrow[r, dash, dotted, "model"] \arrow[d, tail, "det_{jsl}"'] & \mathsf{Rel} \arrow[d, tail, "\mathcal{K}"'] \\
\mathsf{JSLA} \arrow[r, dash, dotted] \arrow[d, "min"'] & \mathsf{JSL} \\
\mathsf{JSLA} \arrow[r, dash, dotted] \arrow[d, dotted, "\mathcal{J}^{gen}"'] & \mathsf{JSL} \arrow[d, "\mathcal{J}"'] \\
\mathsf{NFA} \arrow[r, dash, dotted] & \mathsf{Rel}
\end{tikzcd}
$$

Going back to Rel

Note: $Q_{min} := \mathtt{Min}_{JSL}(\mathcal{L})$ consists unions of sets $w^{-1}L$.

- Reduce a join-semi lattice to its basic components: the join irreducible elements.
- Join irreducible elements $\mathcal{J}(L)$ are elements that are $\neq \perp$ and that aren't joins of other elements, e.g. singletons in a powerset lattice.

$$\begin{tikzcd}
\mathcal{A} \arrow[d, mapsto] \\
\mathcal{A}^{det} \arrow[d, mapsto] \\
min_{\mathsf{JSLA}}(\mathcal{A}^{det}) \arrow[d, mapsto] \\
\mathcal{J}^{gen}(min_{\mathsf{JSLA}}(\mathcal{A}^{\mathsf{JSL}}))
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathsf{NFA} \arrow[r, dotted, no head, "model"] \arrow[d, tail, "det_{jsl}"'] & \mathsf{Rel} \arrow[d, tail, "\mathcal{K}"'] \\
\mathsf{JSLA} \arrow[r, dotted, no head] \arrow[d, "min"'] & \mathsf{JSL} \\
\mathsf{JSLA} \arrow[r, dotted, no head] \arrow[d, dotted, "\mathcal{J}^{gen}"'] & \mathsf{JSL} \arrow[d, "\mathcal{J}"'] \\
\mathsf{NFA} \arrow[r, dotted, no head] & \mathsf{Rel}
\end{tikzcd}$$

Going back to Rel

Note: $Q_{min} := \mathtt{Min}_{JSL}(\mathcal{L})$ consists unions of sets $w^{-1}L$.

- Reduce a join-semi lattice to its basic components: the join irreducible elements.
- Join irreducible elements $\mathcal{J}(L)$ are elements that are $\neq \perp$ and that aren't joins of other elements, e.g. singletons in a powerset lattice.

$$
\begin{tikzcd}
\mathcal{A} \arrow[d, mapsto] \\
\mathcal{A}^{det} \arrow[d, mapsto] \\
min_{\mathsf{JSLA}}(\mathcal{A}^{det}) \arrow[d, mapsto] \\
\mathtt{min}_{RFSA}(\mathcal{A})
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathsf{NFA} \arrow[r, dotted, dash, "model"] \arrow[d, tail, "det_{jsl}"'] & \mathsf{Rel} \arrow[d, tail, "\mathcal{K}"'] \\
\mathsf{JSLA} \arrow[r, dotted, dash] \arrow[d, "min"'] & \mathsf{JSL} \\
\mathsf{JSLA} \arrow[r, dotted, dash] \arrow[d, dotted, "\mathcal{J}^{gen}"'] & \mathsf{JSL} \arrow[d, "\mathcal{J}"'] \\
\mathsf{NFA} \arrow[r, dotted, dash] & \mathsf{Rel}
\end{tikzcd}
$$

Canonical RFSA

For a regular language $\mathcal{L}$ the canonical RFSA of Denis et al. is

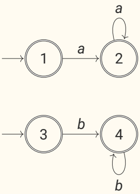
$$min_{RFSA}(\mathcal{L}) = (Q, I, F, \delta)$$

where

$$\begin{aligned}
Q &= \{w^{-1}L \ \texttt{join-irreducible}\} \\
I &= \{w^{-1}L \ \texttt{join-irreducible} \subseteq L\} \\
F &= \{w^{-1}L \ \texttt{join-irreducible} \mid w \in L\} \\
\delta_a &= \{(p, q) \mid \quad q \subseteq a^{-1}p\}
\end{aligned}$$

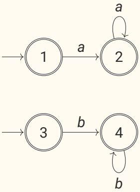
This definition coincides with the automaton generated by the image of $\mathcal{J}$ of the minimal JSLA accepting $\mathcal{L}$.

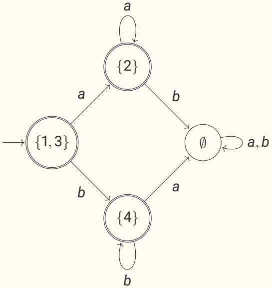
Example



A non-minimal NFA for $L = a^* + b^*$.

Example

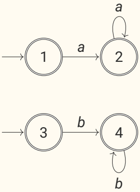


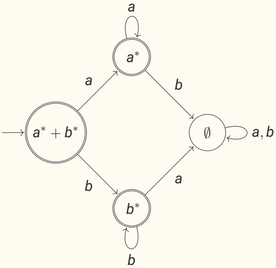


A non-minimal NFA for $L = a^* + b^*$.

1. Determinize and keep only reachable states:

Example

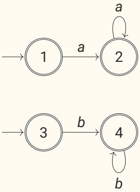


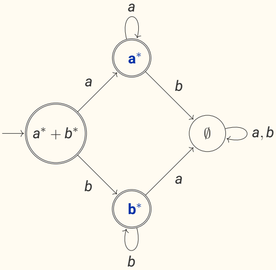


A non-minimal NFA for $L = a^* + b^*$.

1. Determinize and keep only reachable states:
2. Minimize by identifying states accepting the same quotients:

Example

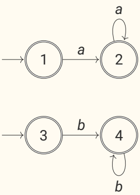


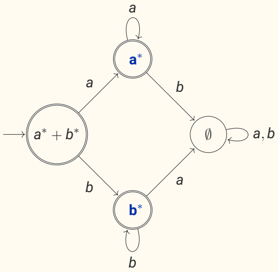


A non-minimal NFA for $L = a^* + b^*$.

1. Determinize and keep only reachable states:
2. Minimize by identifying states accepting the same quotients:
3. Find irreducible states and check if $\subseteq L$:

Example





A non-minimal NFA for $L = a^* + b^*$.

1. Determinize and keep only reachable states:
2. Minimize by identifying states accepting the same quotients:
3. Find irreducible states and check if $\subseteq L$:
4. Construct the NFA on irreducible quotients: