42nd International Symposium on Theoretical Aspects of Computer Science
(STACS 2025)

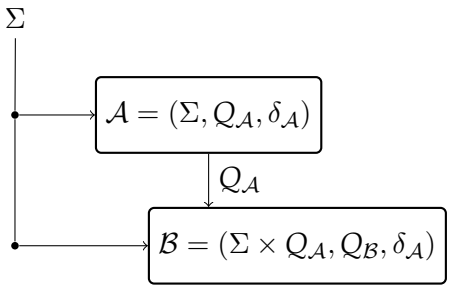
On Cascades of Reset Automata

Roberto Borelli – University of Udine
Luca Geatti – University of Udine
Marco Montali – Free University of Bozen-Bolzano
Angelo Montanari – University of Udine

Jena, March 04-07, 2025

BACKGROUND

Cascades of semiautomata

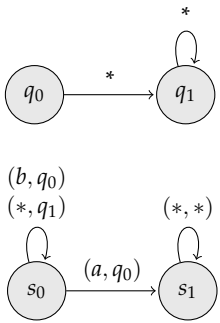


Definition

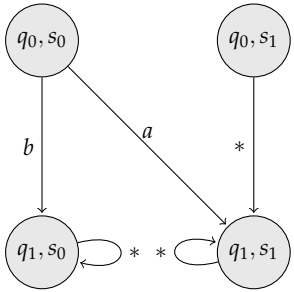
The cascade product between $\mathcal{A}$ and $\mathcal{B}$, denoted with $\mathcal{A} \circ \mathcal{B}$, is the semiautomaton $(\Sigma, Q_\mathcal{A} \times Q_\mathcal{B}, \delta)$ such that, for all $(q_\mathcal{A}, q_\mathcal{B}) \in Q_\mathcal{A} \times Q_\mathcal{B}$ and for all $a \in \Sigma$:

$$\delta((q_\mathcal{A}, q_\mathcal{B}), a) = (\delta_\mathcal{A}(q_\mathcal{A}, a), \delta_\mathcal{B}(q_\mathcal{B}, (a, q_\mathcal{A})))$$

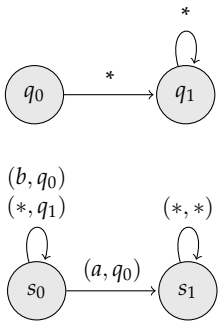
The cascade product is a generalization of the classical direct product.



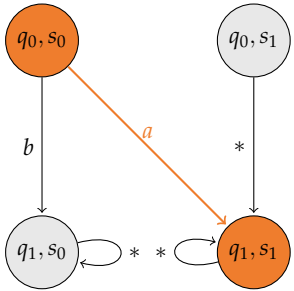
Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



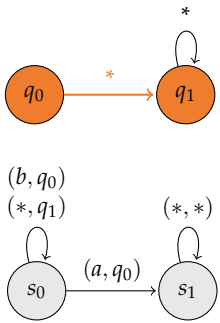
The cascade product $\mathcal{A} \circ \mathcal{B}$.



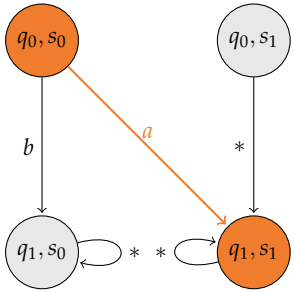
Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



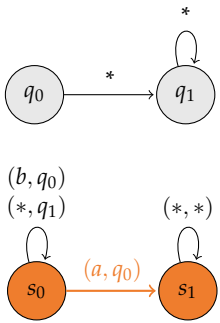
The cascade product $\mathcal{A} \circ \mathcal{B}$.



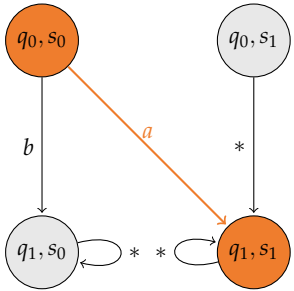
Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



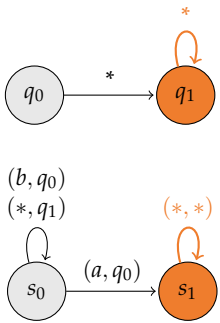
The cascade product $\mathcal{A} \circ \mathcal{B}$.



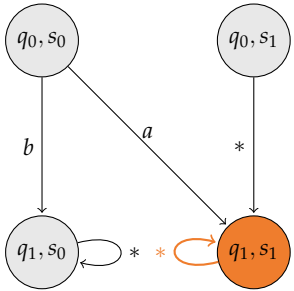
Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



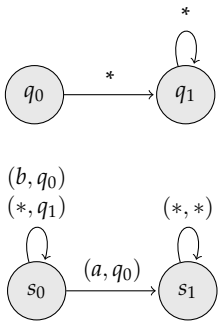
The cascade product $\mathcal{A} \circ \mathcal{B}$.



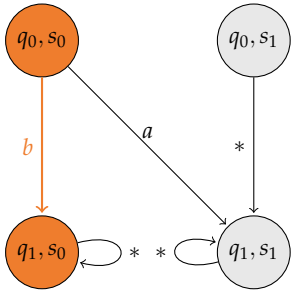
Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



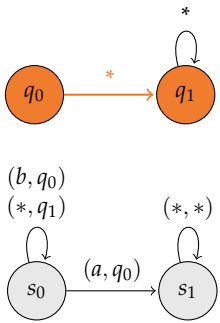
The cascade product $\mathcal{A} \circ \mathcal{B}$.



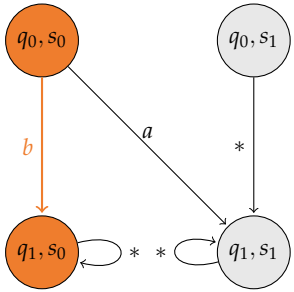
Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



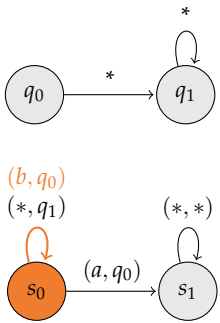
The cascade product $\mathcal{A} \circ \mathcal{B}$.



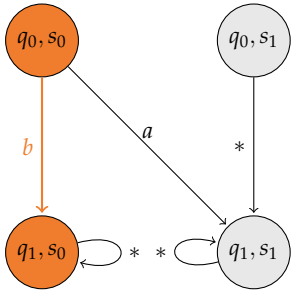
Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



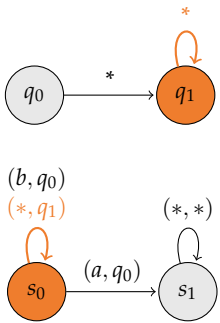
The cascade product $\mathcal{A} \circ \mathcal{B}$.



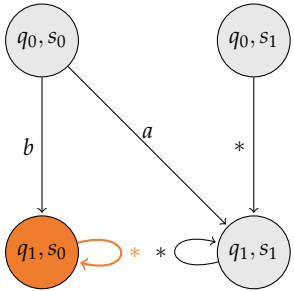
Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



The cascade product $\mathcal{A} \circ \mathcal{B}$.



Semiautomata $\mathcal{A}$ and $\mathcal{B}$.



The cascade product $\mathcal{A} \circ \mathcal{B}$.

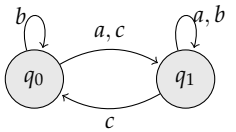
Function induced by a symbol

Definition

Let $\mathcal{A} = (\Sigma, Q, \delta)$ be a semiautomaton. For each symbol $a \in \Sigma$, we define the function induced by a in $\mathcal{A}$, denoted by τ_a, as the transformation

$$\tau_a : Q \to Q$$

such that, for all $q \in Q$, it holds $\tau_a(q) = q'$ iff $\delta(q, a) = q'$.



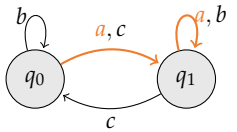
Function induced by a symbol

Definition

Let $\mathcal{A} = (\Sigma, Q, \delta)$ be a semiautomaton. For each symbol $a \in \Sigma$, we define the function induced by a in $\mathcal{A}$, denoted by τ_a, as the transformation

$$\tau_a : Q \to Q$$

such that, for all $q \in Q$, it holds $\tau_a(q) = q'$ iff $\delta(q, a) = q'$.



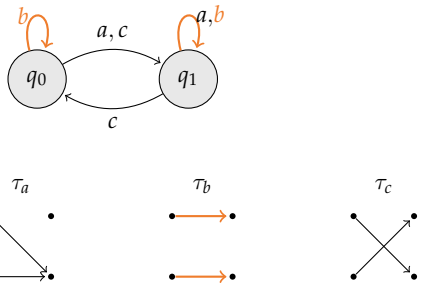
Function induced by a symbol

Definition

Let $\mathcal{A} = (\Sigma, Q, \delta)$ be a semiautomaton. For each symbol $a \in \Sigma$, we define the function induced by a in $\mathcal{A}$, denoted by τ_a, as the transformation

$$\tau_a : Q \to Q$$

such that, for all $q \in Q$, it holds $\tau_a(q) = q'$ iff $\delta(q, a) = q'$.



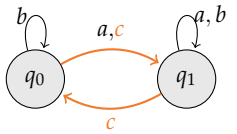
Function induced by a symbol

Definition

Let $\mathcal{A} = (\Sigma, Q, \delta)$ be a semiautomaton. For each symbol $a \in \Sigma$, we define the function induced by a in $\mathcal{A}$, denoted by τ_a, as the transformation

$$\tau_a : Q \to Q$$

such that, for all $q \in Q$, it holds $\tau_a(q) = q'$ iff $\delta(q, a) = q'$.

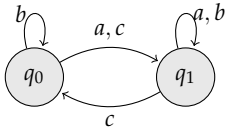


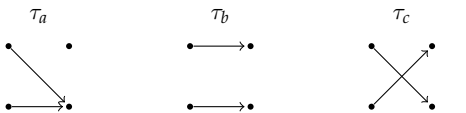
Reset and Permutations

Definition

Let $\tau : Q \to Q$. We say that τ is a

- reset function iff there exists $q' \in Q$ such that $\tau(q) = q'$, for all $q \in Q$. In this case, we say that τ is a *reset on* q'.
- permutation iff $\tau : Q \to Q$ is a bijection.





reset

permutation (identity)

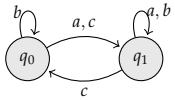
permutation

Reset and Permutations Semiautomata

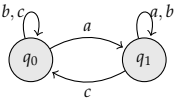
Definition

Let $\mathcal{A} = (\Sigma, Q, \delta)$ be a semiautomaton. We say that $\mathcal{A}$ is:

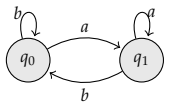
- a permutation-reset semiautomaton iff, for each $a \in \Sigma$, τ_a is either a permutation or a reset.
- a reset semiautomaton iff, for each $a \in \Sigma$, τ_a is either the *identity* function or a reset function;
- a pure-reset semiautomaton iff, for each $a \in \Sigma$, τ_a is a reset function.



(a) A permutation-reset semiautomaton.



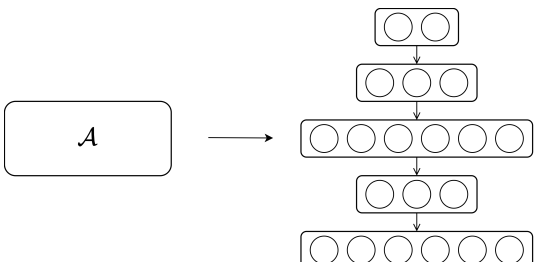
(b) A reset semiautomaton.



(c) A pure-reset semiautomaton.

In its automata-theoretic version, the Krohn-Rhodes theorem (1962) states that:

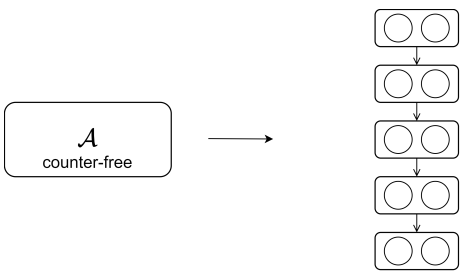
Any semiautomaton $\mathcal{A}$ can be decomposed into a cascade $\mathcal{C}$ of permutation-reset semiautomata such that there exists an homomorphism from the $\mathcal{C}$ to $\mathcal{A}$.



In its automata-theoretic version, the Krohn-Rhodes theorem (1962) states that:

Any semiautomaton $\mathcal{A}$ can be decomposed into a cascade $\mathcal{C}$ of permutation-reset semiautomata such that there exists an homomorphism from the $\mathcal{C}$ to $\mathcal{A}$.

If $\mathcal{A}$ comes from an LTL formula, then each component in $\mathcal{C}$ is a two-state reset semiautomaton.

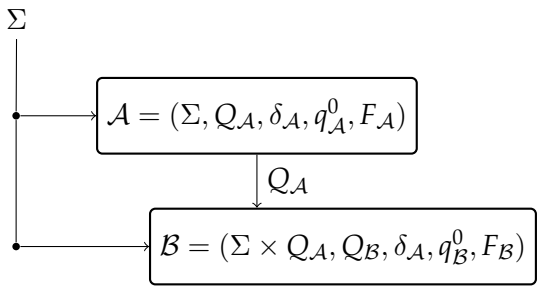


Our Contributions

1. **Cascades of Automata**
   - We define cascades of automata, and from this point onward, we will focus on cascades of reset automata.
   - Regular expressions associated with cascades.
2. **Expressiveness Results**
   - *Hierarchy Lemmas*: What happens if we increase the number of components? And what about the number of states in the first component?
   - *Narrowing Lemma*: What happens if we restrict ourselves to two-state cascades?
3. **Efficient Closure Properties**
   - Intersection, union, complementation, and left concatenation over Σ.

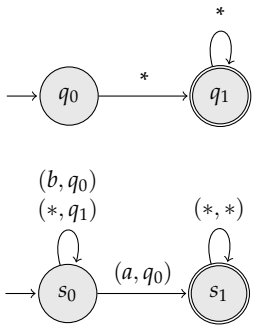
CASCADES OF AUTOMATA

We generalize the notion of cascade product of semiautomata to automata.

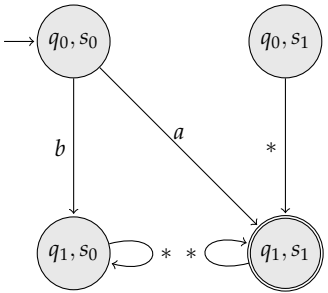


Definition

The cascade product between $\mathcal{A}$ and $\mathcal{B}$, is the automaton $(\Sigma, Q_\mathcal{A} \times Q_\mathcal{B}, \delta, (q^0_\mathcal{A}, q^0_\mathcal{B}), F_\mathcal{A} \times F_\mathcal{B})$ where δ is defined as in the case of cascades of semiautomata.

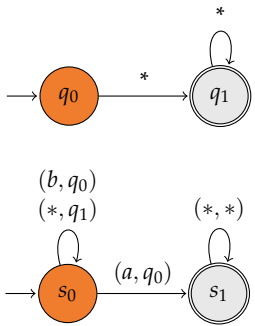


Automata $\mathcal{A}$ and $\mathcal{B}$.

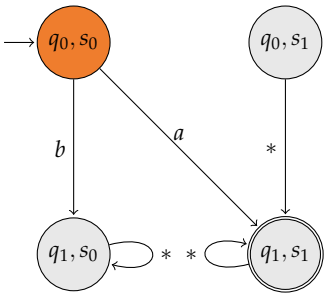


The cascade product $\mathcal{A} \circ \mathcal{B}$.

$$\mathcal{L}(\mathcal{A} \circ \mathcal{B}) = a \cdot \Sigma^*$$

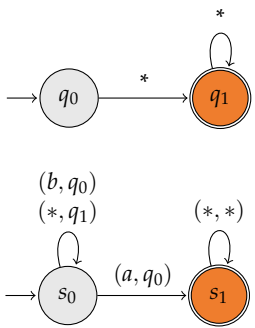


Automata $\mathcal{A}$ and $\mathcal{B}$.

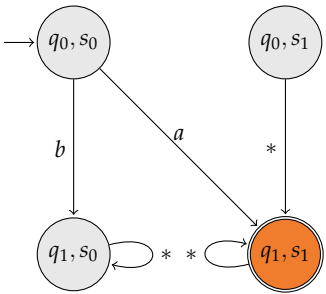


The cascade product $\mathcal{A} \circ \mathcal{B}$.

$$\mathcal{L}(\mathcal{A} \circ \mathcal{B}) = a \cdot \Sigma^*$$



Automata $\mathcal{A}$ and $\mathcal{B}$.



The cascade product $\mathcal{A} \circ \mathcal{B}$.

$$\mathcal{L}(\mathcal{A} \circ \mathcal{B}) = a \cdot \Sigma^*$$

Language of a cascade of height 1

Theorem

Let Σ be a finite alphabet. A language $\mathcal{L} \subseteq \Sigma^*$ is recognized by a reset automaton $\mathcal{A}$ if and only if

$$\mathcal{L} = J \cup (\Sigma^* \cdot R \cdot I^*)$$

for some $I, R \subseteq \Sigma$ such that $I \cap R = \varnothing$ and either $J = I^*$ or $J = \varnothing$.

Accepting runs of $\mathcal{A}$:

Language of a cascade of height 1

Theorem

Let Σ be a finite alphabet. A language $\mathcal{L} \subseteq \Sigma^*$ is recognized by a reset automaton $\mathcal{A}$ if and only if

$$\mathcal{L} = J \cup (\Sigma^* \cdot R \cdot I^*)$$

for some $I, R \subseteq \Sigma$ such that $I \cap R = \varnothing$ and either $J = I^*$ or $J = \varnothing$.

Accepting runs of $\mathcal{A}$:

1. The automaton can be in any state.

Language of a cascade of height 1

Theorem

Let Σ be a finite alphabet. A language $\mathcal{L} \subseteq \Sigma^*$ is recognized by a reset automaton $\mathcal{A}$ if and only if

$$\mathcal{L} = J \cup (\Sigma^* \cdot R \cdot I^*)$$

for some $I, R \subseteq \Sigma$ such that $I \cap R = \varnothing$ and either $J = I^*$ or $J = \varnothing$.

Accepting runs of $\mathcal{A}$:

1. The automaton can be in any state.
2. The automaton follows a *reset transition* to a final state.

Language of a cascade of height 1

Theorem

Let Σ be a finite alphabet. A language $\mathcal{L} \subseteq \Sigma^*$ is recognized by a reset automaton $\mathcal{A}$ if and only if

$$\mathcal{L} = J \cup (\Sigma^* \cdot R \cdot I^*)$$

for some $I, R \subseteq \Sigma$ such that $I \cap R = \varnothing$ and either $J = I^*$ or $J = \varnothing$.

Accepting runs of $\mathcal{A}$:

1. The automaton can be in any state.
2. The automaton follows a *reset transition* to a final state.
3. The automaton remains in the final state through *identity transitions*.

Language of a cascade of height 1

Theorem

Let Σ be a finite alphabet. A language $\mathcal{L} \subseteq \Sigma^*$ is recognized by a reset automaton $\mathcal{A}$ if and only if

$$\mathcal{L} = \boxed{J} \cup (\Sigma^* \cdot R \cdot I^*)$$

for some $I, R \subseteq \Sigma$ such that $I \cap R = \varnothing$ and either $\boxed{J = I^*}$ or $J = \varnothing$.

Accepting runs of $\mathcal{A}$:

1. The automaton can be in any state.
2. The automaton follows a *reset transition* to a final state.
3. The automaton remains in the final state through *identity transitions*.
4. If the initial state is also final, the automaton can accept words just by staying in its initial state following *identity transitions*.

Language of a cascade of height 1

Theorem

Let Σ be a finite alphabet. A language $\mathcal{L} \subseteq \Sigma^*$ is recognized by a reset automaton $\mathcal{A}$ if and only if

$$\mathcal{L} = J \cup (\Sigma^* \cdot R \cdot I^*)$$

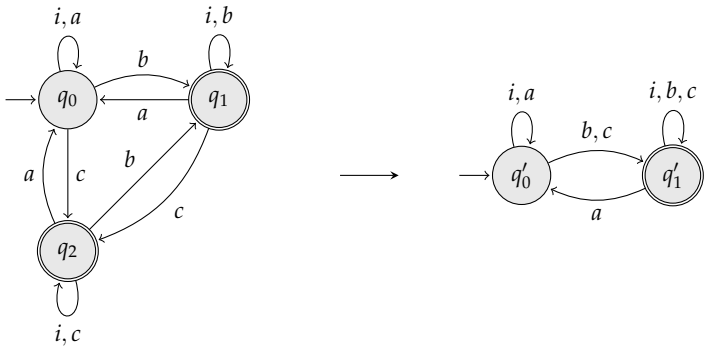
for some $I, R \subseteq \Sigma$ such that $I \cap R = \varnothing$ and either $J = I^*$ or $J = \varnothing$.

Accepting runs of $\mathcal{A}$:

1. The automaton can be in any state.
2. The automaton follows a *reset transition* to a final state.
3. The automaton remains in the final state through *identity transitions*.
4. If the initial state is also final, the automaton can accept words just by staying in its initial state following *identity transitions*.

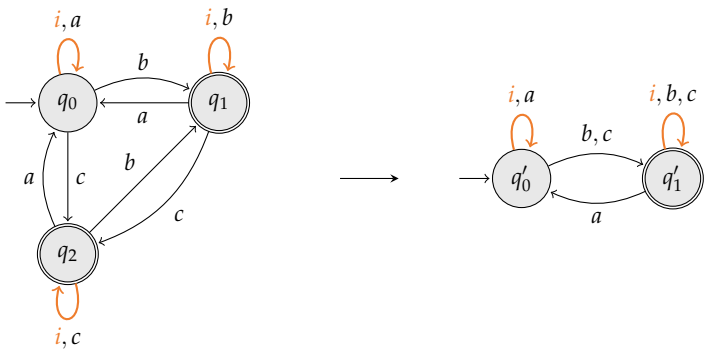
Corollary

For every reset automaton, there exists an equivalent one with two states, exactly one of which is final.



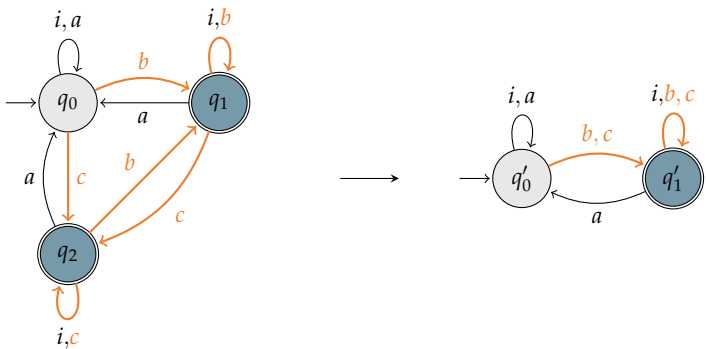
Corollary

For every reset automaton, there exists an equivalent one with two states, exactly one of which is final.



Corollary

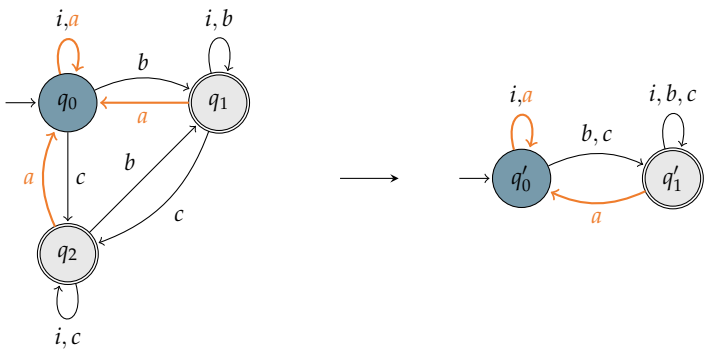
For every reset automaton, there exists an equivalent one with two states, exactly one of which is final.



Language of a cascade of height 1

Corollary

For every reset automaton, there exists an equivalent one with two states, exactly one of which is final.



Cascades of height 1 for LTL_f formulas

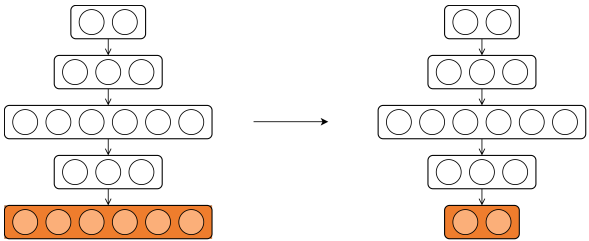
Reset cascade decomposition is relevant in problems such as pastification and normalization of temporal logic formulas. It is crucial to study which LTL_f formulas can be expressed with cascades of resets of a specific height.

LTL_f **formula**	**Regular Expression**	**Definable by a reset cascade of height 1**
$\mathsf{F}p$	$\Sigma^* \cdot \vec{p}$	✓
p	$\vec{p} \cdot \Sigma^*$	✗
$p \,\mathsf{U}\, q$	$(\vec{p})^* \cdot \vec{q} \cdot \Sigma^*$	✗

On the final automaton of a cascade

Lemma

The last automaton of a reset cascade can always be restricted to be a two-state reset automaton with exactly one final state.



Language of a cascade of unbounded Height

Theorem

Let $\mathcal{A} = \langle \Sigma, Q, \delta_A, q_0, F_A \rangle$ be an automaton and let $\mathcal{B} = \langle \Sigma \times Q, \{s_0, s_1\}, \delta, s_0, \{s_f\} \rangle$, with $s_f \in \{s_0, s_1\}$, be a two-state reset automaton with one final state. It holds that:

$$\mathcal{L}(\mathcal{A} \circ \mathcal{B}) = M \cup \bigcup_{(\sigma,q) \in R_{s_f}} \mathcal{L}_q(\mathcal{A}) \cdot \sigma \cdot \mathcal{L}\left(\mathcal{A} \downarrow_{I_B}^{\delta_A(q,\sigma)}\right)$$

where R_{s_f} is the set of symbols in that induce a reset function on state s_f, and $M \coloneqq \mathcal{L}(\mathcal{A} \downarrow_{I_B}^{q_0})$ if $s_0 = s_f$ or $M \coloneqq \varnothing$ otherwise.

Language of a cascade of unbounded Height

Theorem

Let $\mathcal{A} = \langle \Sigma, Q, \delta_A, q_0, F_A \rangle$ be an automaton and let $\mathcal{B} = \langle \Sigma \times Q, \{s_0, s_1\}, \delta, s_0, \{s_f\} \rangle$, with $s_f \in \{s_0, s_1\}$, be a two-state reset automaton with one final state. It holds that:

$$\mathcal{L}(\mathcal{A} \circ \mathcal{B}) = M \cup \bigcup_{(\sigma, q) \in R_{s_f}} \mathcal{L}_q(\mathcal{A}) \cdot \sigma \cdot \mathcal{L}\left(\mathcal{A} \downarrow_{I_B}^{\delta_A(q,\sigma)}\right)$$

where R_{s_f} is the set of symbols in that induce a reset function on state s_f, and $M \coloneqq \mathcal{L}(\mathcal{A} \downarrow_{I_B}^{q_0})$ if $s_0 = s_f$ or $M \coloneqq \varnothing$ otherwise.

The automaton $\mathcal{B}$ takes a reset transition to its final state:

Language of a cascade of unbounded Height

Theorem

Let $\mathcal{A} = \langle \Sigma, Q, \delta_A, q_0, F_A \rangle$ be an automaton and let $\mathcal{B} = \langle \Sigma \times Q, \{s_0, s_1\}, \delta, s_0, \{s_f\} \rangle$, with $s_f \in \{s_0, s_1\}$, be a two-state reset automaton with one final state. It holds that:

$$\mathcal{L}(\mathcal{A} \circ \mathcal{B}) = M \cup \bigcup_{(\sigma, q) \in R_{s_f}} \mathcal{L}_q(\mathcal{A}) \cdot \sigma \cdot \mathcal{L}\left(\mathcal{A} \downarrow_{I_B}^{\delta_A(q,\sigma)}\right)$$

where R_{s_f} is the set of symbols in that induce a reset function on state s_f, and $M \coloneqq \mathcal{L}(\mathcal{A} \downarrow_{I_B}^{q_0})$ if $s_0 = s_f$ or $M \coloneqq \varnothing$ otherwise.

The automaton $\mathcal{B}$ takes a reset transition to its final state: (i) $\mathcal{A}$ must be in state q

Language of a cascade of unbounded Height

Theorem

Let $\mathcal{A} = \langle \Sigma, Q, \delta_A, q_0, F_A \rangle$ be an automaton and let $\mathcal{B} = \langle \Sigma \times Q, \{s_0, s_1\}, \delta, s_0, \{s_f\} \rangle$, with $s_f \in \{s_0, s_1\}$, be a two-state reset automaton with one final state. It holds that:

$$\mathcal{L}(\mathcal{A} \circ \mathcal{B}) = M \cup \bigcup_{(\sigma, q) \in R_{s_f}} \mathcal{L}_q(\mathcal{A}) \cdot \sigma \cdot \mathcal{L}\left(\mathcal{A} \downarrow_{I_B}^{\delta_A(q,\sigma)}\right)$$

where R_{s_f} is the set of symbols in that induce a reset function on state s_f, and $M \coloneqq \mathcal{L}(\mathcal{A} \downarrow_{I_B}^{q_0})$ if $s_0 = s_f$ or $M \coloneqq \varnothing$ otherwise.

The automaton $\mathcal{B}$ takes a reset transition to its final state: (i) $\mathcal{A}$ must be in state q and (ii) the input symbol must be σ.

Language of a cascade of unbounded Height

Theorem

Let $\mathcal{A} = \langle \Sigma, Q, \delta_A, q_0, F_A \rangle$ be an automaton and let $\mathcal{B} = \langle \Sigma \times Q, \{s_0, s_1\}, \delta, s_0, \{s_f\} \rangle$, with $s_f \in \{s_0, s_1\}$, be a two-state reset automaton with one final state. It holds that:

$$\mathcal{L}(\mathcal{A} \circ \mathcal{B}) = M \cup \bigcup_{(\sigma,q) \in R_{s_f}} \mathcal{L}_q(\mathcal{A}) \cdot \sigma \cdot \mathcal{L}\left(\mathcal{A} \downarrow_{I_B}^{\delta_A(q,\sigma)}\right)$$

where R_{s_f} is the set of symbols in that induce a reset function on state s_f, and $M := \mathcal{L}(\mathcal{A} \downarrow_{I_B}^{q_0})$ if $s_0 = s_f$ or $M := \varnothing$ otherwise.

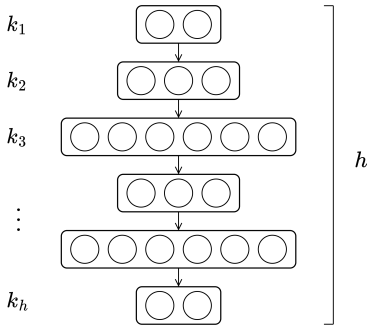
The automaton $\mathcal{A}$ after reading σ, must reach a final state using only certain transitions that allow $\mathcal{B}$ to remain in its final state by following identity transitions.

EXPRESSIVENESS RESULTS

Classes of languages

Definition

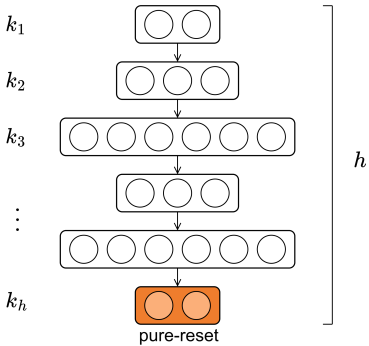
- $\mathcal{R}(k_1, \ldots, k_h)$: languages definable by a cascade $\mathcal{A}_1 \circ \cdots \circ \mathcal{A}_h$ of reset automata such that $\mathcal{A}_i$ has k_i states, for each $1 \leq i \leq h$.



Classes of languages

Definition

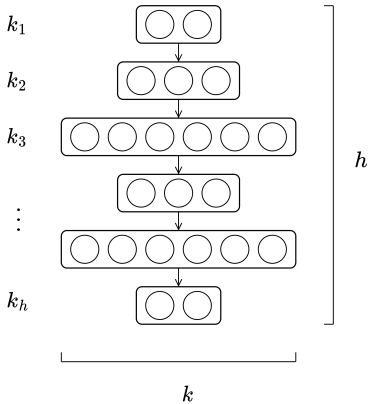
- $\mathcal{R}(k_1, \ldots, k_h)$: languages definable by a cascade $\mathcal{A}_1 \circ \cdots \circ \mathcal{A}_h$ of reset automata such that $\mathcal{A}_i$ has k_i states, for each $1 \leq i \leq h$.
- $\mathcal{RPR}(k_1, \ldots, k_h)$: subclass of $\mathcal{R}(k_1, \ldots, k_h)$ where the last automaton ($\mathcal{A}_h$) is required to be pure-reset.



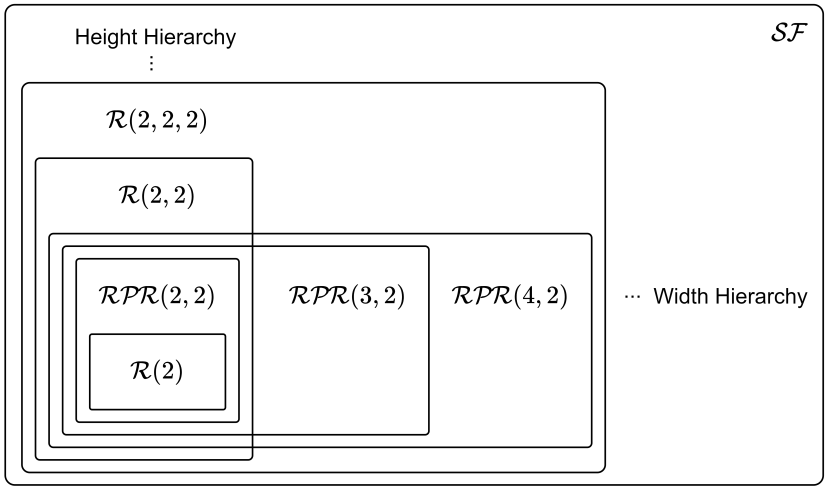
Classes of languages

Definition

- $\mathcal{R}(k_1, \ldots, k_h)$: languages definable by a cascade $\mathcal{A}_1 \circ \cdots \circ \mathcal{A}_h$ of reset automata such that $\mathcal{A}_i$ has k_i states, for each $1 \leq i \leq h$.
- $\mathcal{RPR}(k_1, \ldots, k_h)$: subclass of $\mathcal{R}(k_1, \ldots, k_h)$ where the last automaton ($\mathcal{A}_h$) is required to be pure-reset.
- $\mathcal{R}_k^h$: languages definable by a reset cascade of height h, where each component has $\leq k$ states.



Overall picture of expressiveness



A reset automaton in cascade with a pure-reset layer, is able to recognize the set of words ending with a two-character suffix. This is impossible using a single reset automaton.

Lemma

Let $L = \Sigma^* aa$. It holds that:

1. $L \in \mathcal{RPR}(2,2)$;
2. $L \notin \mathcal{R}(2)$.

Prohibiting identities in the final layer results in a loss of expressive power no matter of the number of states of the first automaton.

Lemma

Let Σ be an alphabet with at least two symbols, let $L = a\Sigma^*$. It holds that:

1. $L \in \mathcal{R}(2, 2)$;
2. $L \notin \mathcal{RPR}(k, 2)$, *for every* $k \geq 2$.

The Width-Hierarchy Lemma

We have seen that the final component of a reset cascade can always be restricted to be two-state. This does not apply to the first component.

Lemma

For each $k > 2$, let $\Sigma = \{\sigma_0, \ldots, \sigma_{k-1}\}$. Let $L_k = \Sigma^* \left(\Sigma^2 \setminus \bigcup_{0 \le i < k} \sigma_i \sigma_i\right) \cup (\Sigma \setminus \sigma_0)$, it holds that:

1. $L_k \in \mathcal{RPR}(k, 2)$;
2. $L_k \notin \mathcal{R}(k-1, 2)$.

The Height-Hierarchy Lemma

We have seen that some languages cannot be expressed by cascades of height 1. For any given height, there exists a language that cannot be captured at that height, provided the components of the cascade are restricted to two-state resets.

Lemma

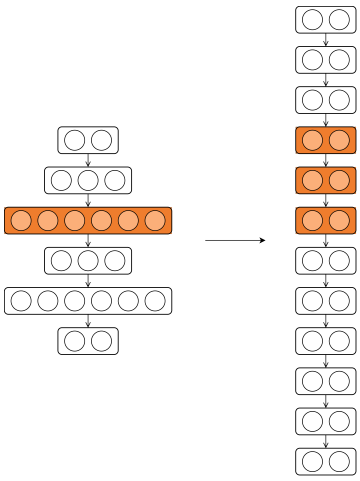
For each $h \geq 2$, let $L_h = \Sigma^{h-2} a \Sigma^*$. It holds that:

1. $L_h \in \mathcal{R}_2^h$;
2. $L_h \notin \mathcal{R}_2^{h-1}$.

The Narrowing Lemma

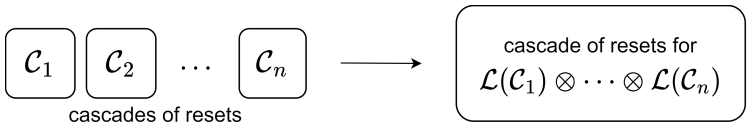
Lemma

A cascade of resets can be transformed into a cascade composed entirely of two-state resets, with a quadratic increase of the height.

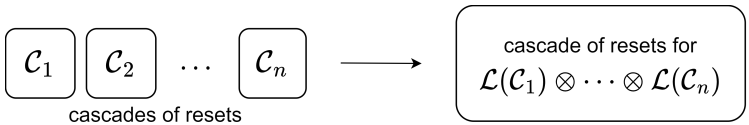


EFFICIENT CLOSURE PROPERTIES

Closure of cascades of resets



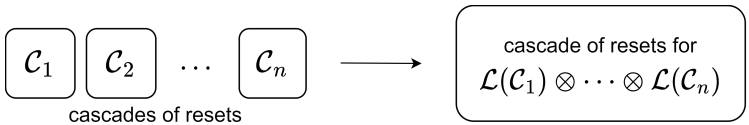
Closure of cascades of resets



We treat the cases where $\otimes$ is:

- Intersection
- Negation
- Union
- Left concatenation of Σ

Closure of cascades of resets



We treat the cases where $\otimes$ is:

- Intersection
- Negation
- Union
- Left concatenation of Σ

We achieve closure properties by:

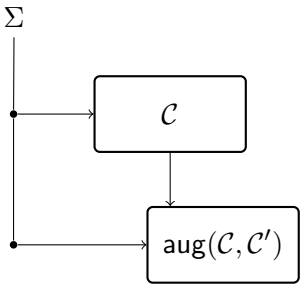
- adding two-states pure-reset components
- modifying original cascades with operations that do not alter the property of being a reset cascade

Intersection

Lemma

Let $\mathcal{C}$ and $\mathcal{C}'$ be two reset cascades of height respectively n and m. There exists a reset cascade for the language $\mathcal{L}(\mathcal{C}) \cap \mathcal{L}(\mathcal{C}')$ of height $n + m$.

The cascade $\boxed{\mathsf{aug}\,(\mathcal{C}, \mathcal{C}')}$ behaves like $\mathcal{C}'$ and ignores the state of $\mathcal{C}$. Furthermore it holds that $\mathcal{C} \circ \mathsf{aug}(\mathcal{C}, \mathcal{C}') = \mathcal{C} \times \mathcal{C}'$.



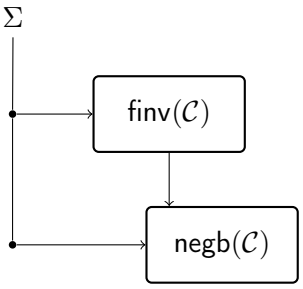
Negation

Lemma

Let $\mathcal{C}$ be a reset cascade of height n. There exists a reset cascade for the language $\overline{\mathcal{L}(\mathcal{C})}$ of height $n+1$.

The cascade $\boxed{\mathsf{finv}\,(\mathcal{C})}$ is the cascade $\mathcal{C}$ with all states set as final.

The block $\boxed{\mathsf{negb}\,(\mathcal{C})}$ reaches a final state whenever $\mathcal{C}$ is in a non final state and viceversa.

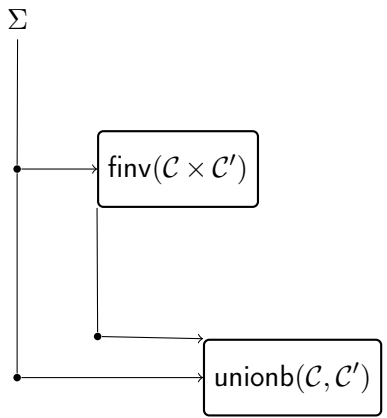


Union

Lemma

Let $\mathcal{C}$ and $\mathcal{C}'$ be two reset cascades of height respectively n and m. There exists a reset cascade for the language $\mathcal{L}(\mathcal{C}) \cup \mathcal{L}(\mathcal{C}')$ of height $n + m + 1$.

The block $\boxed{\mathsf{unionb}\,(\mathcal{C}, \mathcal{C}')}$ reaches a final state whenever either $\mathcal{C}$ is in a final state or $\mathcal{C}'$ is in a final state.

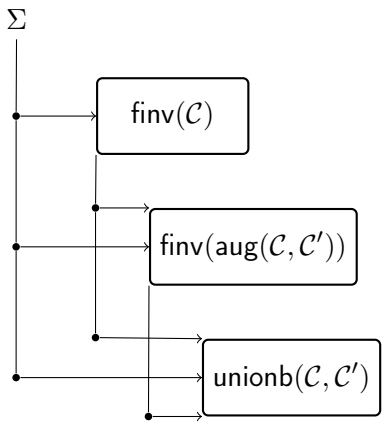


Union

Lemma

Let $\mathcal{C}$ and $\mathcal{C}'$ be two reset cascades of height respectively n and m. There exists a reset cascade for the language $\mathcal{L}(\mathcal{C}) \cup \mathcal{L}(\mathcal{C}')$ of height $n + m + 1$.

The block $\boxed{\mathsf{unionb}\,(\mathcal{C}, \mathcal{C}')}$ reaches a final state whenever either $\mathcal{C}$ is in a final state or $\mathcal{C}'$ is in a final state.



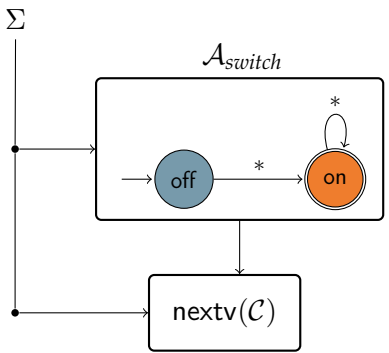
Left concatenation of Σ

Lemma

Let $\mathcal{C}$ be a reset cascade of height n. There exists a reset cascade for the language $\Sigma \cdot \mathcal{L}(\mathcal{C})$ of height $n+1$.

The cascade $\boxed{\mathsf{nextv}\,(\mathcal{C})}$:

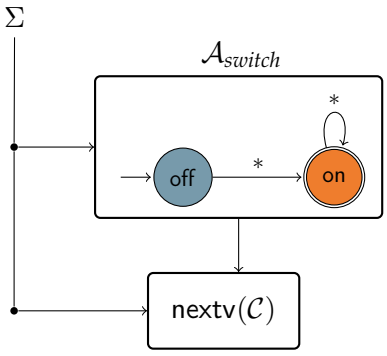
- goes to its initial state (trough a reset transition) whenever it reads off
- operates as $\mathcal{C}$ whenever it reads on



Left concatenation of Σ

Lemma

Let $\mathcal{C}$ be a reset cascade of height n. There exists a reset cascade for the language $\Sigma \cdot \mathcal{L}(\mathcal{C})$ of height $n+1$.



The cascade $\boxed{\mathsf{nextv}\,(\mathcal{C})}$:

- goes to its initial state (trough a reset transition) whenever it reads off
- operates as $\mathcal{C}$ whenever it reads on

Given a cascade of resets of height h for an LTL_f formula ϕ, we can construct a cascades of resets for $\mathsf{X}(\phi)$ of height $h+1$.

CONCLUSIONS

Summary of the results and future works

Summary of the results:

- We studied *cascades of reset automata* instead of semiautomata, deriving regular expressions and expressiveness bounds.
- Reset cascades are closed under fundamental operations with minimal structural modifications.

Future works:

- Closure under concatenation. This would enable efficient decomposition into reset cascades for full LTL_f.
- Extend the study to cascades of permutation automata.

THANK YOU!
ANY QUESTION?

REFERENCES

Bibliography I

Kenneth Krohn and John Rhodes (1965). "Algebraic theory of machines. I. Prime decomposition theorem for finite semigroups and machines". In: *Transactions of the American Mathematical Society* 116, pp. 450–464.

Oded Maler and Amir Pnueli (1990). "Tight Bounds on the Complexity of Cascaded Decomposition of Automata". In: *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*. IEEE Computer Society, pp. 672–682. DOI: 10.1109/FSCS.1990.89589. URL: https://doi.org/10.1109/FSCS.1990.89589.

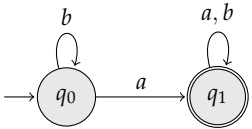
Oded Maler (2010). "On the Krohn-Rhodes Cascaded Decomposition Theorem". In: *Time for Verification, Essays in Memory of Amir Pnueli*. Ed. by Zohar Manna and Doron A. Peled. Vol. 6200. Lecture Notes in Computer Science. Springer, pp. 260–278. DOI: 10.1007/978-3-642-13754-9_12. URL: https://doi.org/10.1007/978-3-642-13754-9%5C_12.

Bibliography II

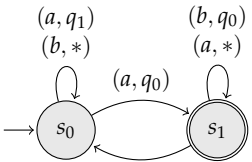
Marcus Gelderie (2011). “Classifying regular languages via cascade products of automata”. In: *Language and Automata Theory and Applications: 5th International Conference, LATA 2011, Tarragona, Spain, May 26-31, 2011. Proceedings 5*. Springer, pp. 286–297.

Karl-Heinz Zimmermann (2020). “On Krohn-Rhodes theory for semiautomata”. In: *CoRR* abs/2010.16235. arXiv: 2010.16235. URL: https://arxiv.org/abs/2010.16235.

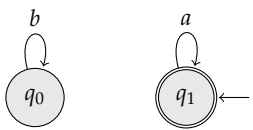
APPENDIX



(a) The automaton $\mathcal{A}$.



(b) The automaton $\mathcal{B}$.



(c) The automaton $\mathcal{A} \downarrow_{I_B}^{s_1}$ where $I_B = \{(a, q_1), (b, q_0)\}$.

$$\begin{aligned}\mathcal{L}(\mathcal{A} \circ \mathcal{B}) &= \mathcal{L}_{q_0}(\mathcal{A}) \cdot a \cdot \mathcal{L}(\mathcal{A} \downarrow_{I_B}^{s_1}) \\ &= b^* \cdot a \cdot a^* = b^* \cdot a^+\end{aligned}$$