Implémentation d'automates de pomsets

Présentations de fin de semestre

Encadré par Amazigh Amrane

3 Juillet 2024

Motivation

- Apprentissage actif

Motivation

- Apprentissage actif $\rightarrow$ minimisation, modélisation de système, vérification, ...

Motivation

- Apprentissage actif $\rightarrow$ minimisation, modélisation de système, vérification, ...
- Pomsets & leurs machines

Motivation

- Apprentissage actif $\rightarrow$ minimisation, modélisation de système, vérification, ...

- Pomsets & leurs machines $\rightarrow$ modélisation de systèmes concurrents/non déterministes.

Motivation

- Apprentissage actif $\rightarrow$ minimisation, modélisation de système, vérification, ...
- Pomsets & leurs machines $\rightarrow$ modélisation de systèmes concurrents/non déterministes.
- **Étendre les méthodes d'apprentissage actif vers les machines de pomsets**

Motivation

- Apprentissage actif $\rightarrow$ minimisation, modélisation de système, vérification, ...
- Pomsets & leurs machines $\rightarrow$ modélisation de systèmes concurrents/non déterministes.
- **Étendre les méthodes d'apprentissage actif vers les machines de pomsets** $\rightarrow$ analyser les programmes concurrents.

Ordres partiels

- **Ordre total :** réflexif, transitif, antisymétrique et fortement connecté.

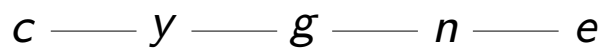
  Mots, ex :

$$c \text{ —— } y \text{ —— } g \text{ —— } n \text{ —— } e$$

Ordres partiels

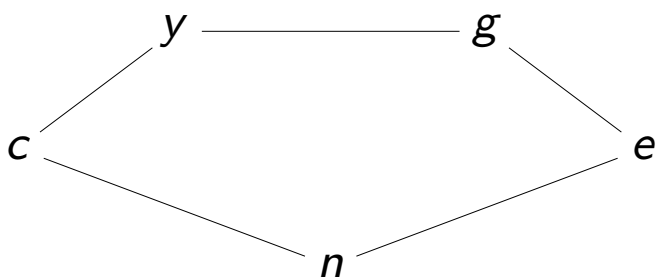
- **Ordre total :** réflexif, transitif, antisymétrique et fortement connecté.

  Mots, ex :



- **Ordre partiel :** réflexif, transitif et antisymétrique.

  Posets, ex :



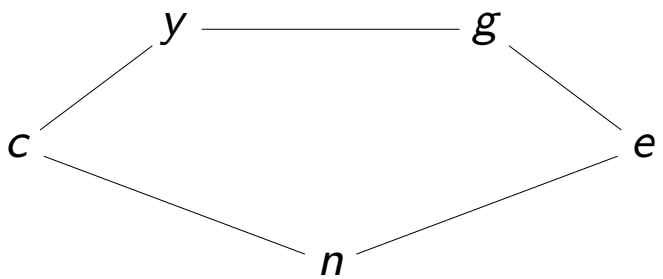
Pomsets

- **Poset** : $(S, <, i)$

Pomsets

- **Poset** : $(S, <, i)$

  ex : c((yg)||n)e :

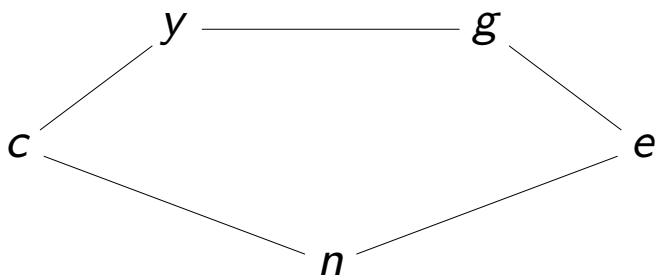


- On note yg||n.

Pomsets

- **Poset** : $(S, <, i)$

  ex : c((yg)||n)e :

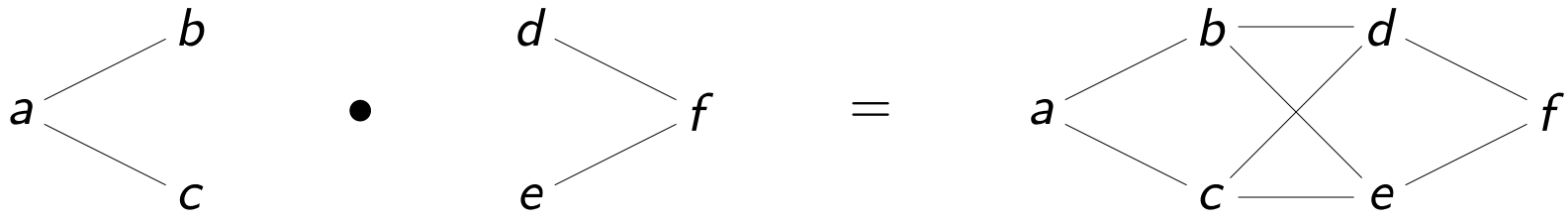


- On note yg||n.

- **Pomset** : classe d'isomorphisme de posets à étiquetage près.

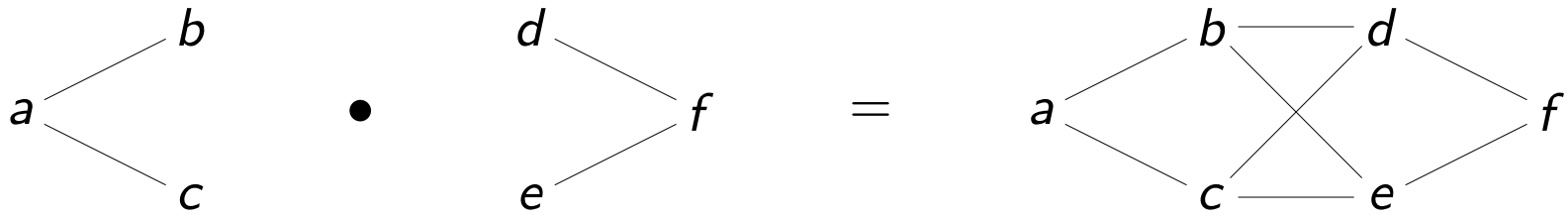
Pomsets - opérations rationnelles

- produit séquentiel (concaténation) :

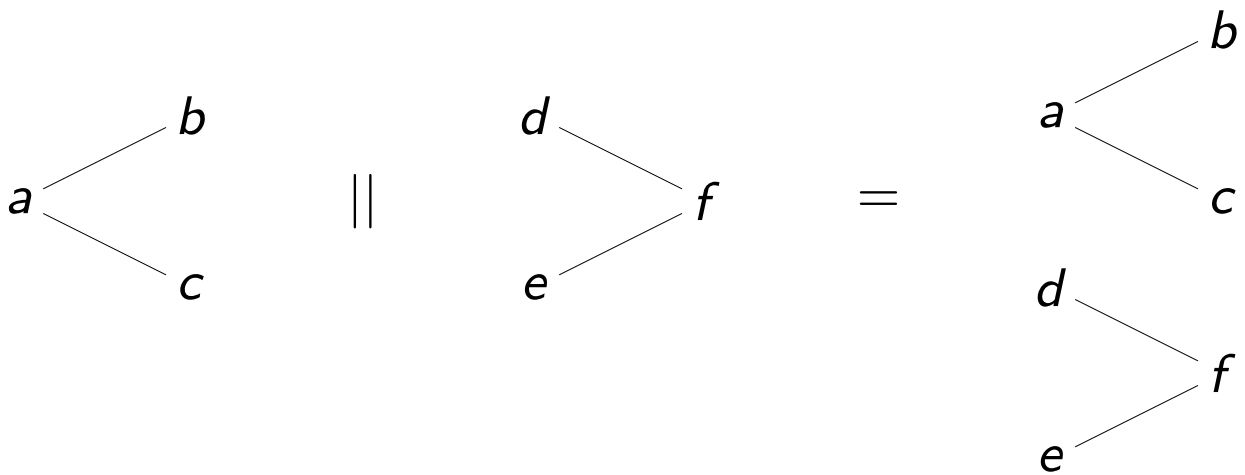


Pomsets - opérations rationnelles

- produit séquentiel (concaténation) :

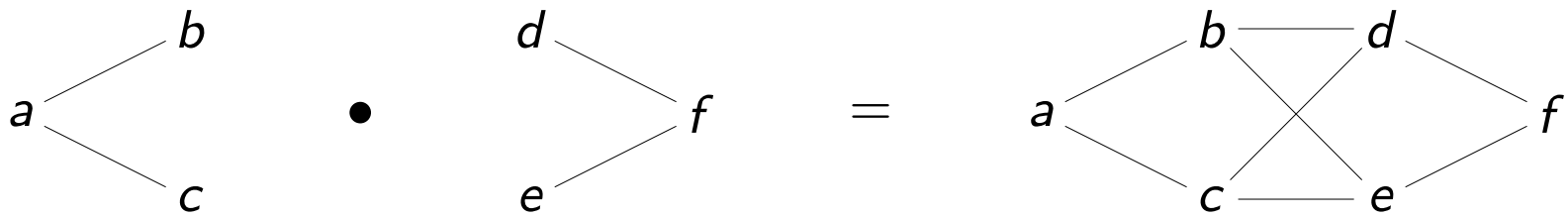


- produit parallèle :

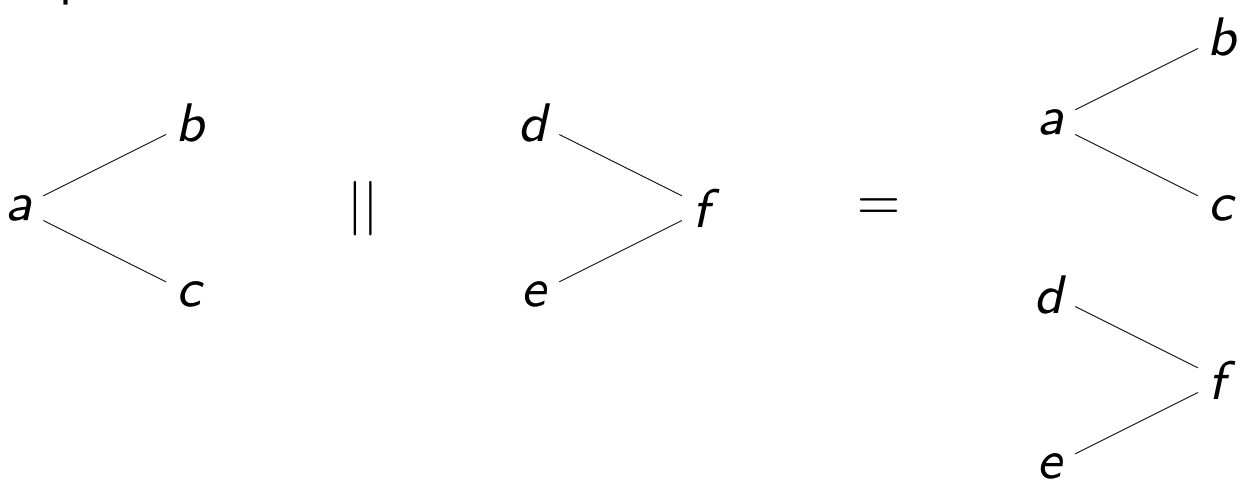


Pomsets - opérations rationnelles

- produit séquentiel (concaténation) :



- produit parallèle :



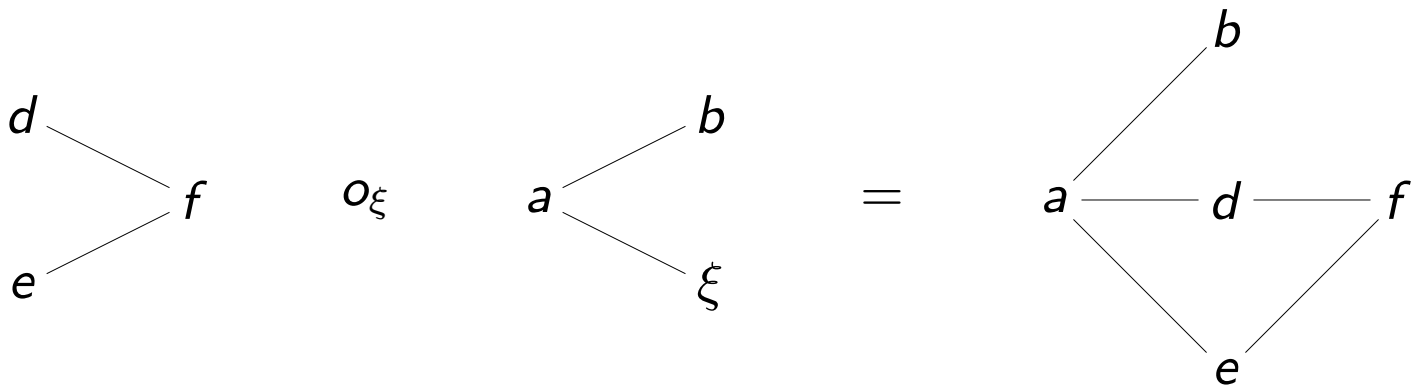
- Préservation de l'ordre et de l'étiquetage des opérandes.

Pomsets - substitution

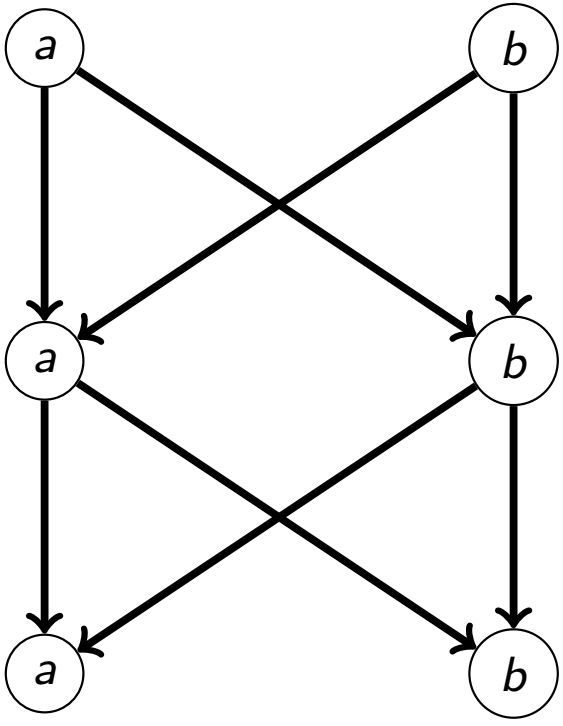
- $p2\ o_\xi\ p1$.
- Remplacer toutes les occurrences de ξ dans $p1$ par $p2$.

Pomsets - substitution

- $p2\ o_\xi\ p1$.
- Remplacer toutes les occurrences de ξ dans $p1$ par $p2$.

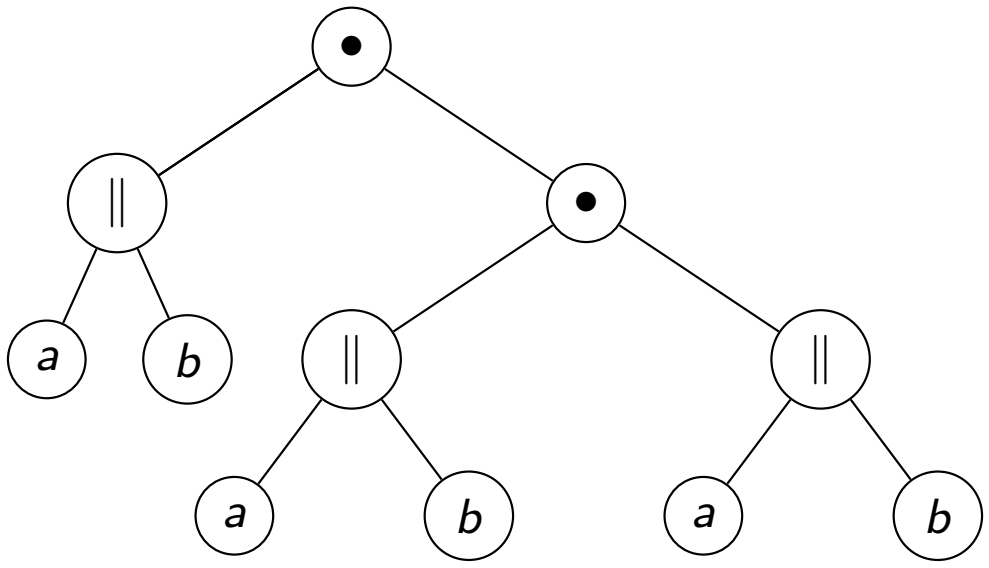


Pomsets - DAG



n noeuds

Pomsets - AST



2n-1 noeuds mais moindre complexité de traitement.

Relation de Myhill-Nerode

- Un mot c est un distingueur des mots w_1 et w_2 de L ssi :

$$w_1 \bullet c \in L \neq w_2 \bullet c \in L$$

Relation de Myhill-Nerode

- Un mot c est un distingueur des mots w_1 et w_2 de L ssi :

$$w_1 \bullet c \in L \neq w_2 \bullet c \in L$$

- **Relation de Myhill-Nerode :**

$$\forall (w_1, w_2) \in L^2, w_1 \sim_L w_2 \iff \nexists c \in \Sigma^*, w_1 \bullet c \in L \neq w_2 \bullet c \in L$$

Relation de Myhill-Nerode

- Un mot c est un distingueur des mots w_1 et w_2 de L ssi :

$$w_1 \bullet c \in L \neq w_2 \bullet c \in L$$

- **Relation de Myhill-Nerode :**

$$\forall (w_1, w_2) \in L^2, w_1 \sim_L w_2 \iff \nexists c \in \Sigma^*, w_1 \bullet c \in L \neq w_2 \bullet c \in L$$

- Partitionne les langages rationnels en un nombre fini de classes d'équivalence. (th. de Myhill-Nerode)

Relation de Myhill-Nerode

- Un mot c est un distingueur des mots w_1 et w_2 de L ssi :

$$w_1 \bullet c \in L \neq w_2 \bullet c \in L$$

- **Relation de Myhill-Nerode :**

$$\forall (w_1, w_2) \in L^2, w_1 \sim_L w_2 \iff \nexists c \in \Sigma^*, w_1 \bullet c \in L \neq w_2 \bullet c \in L$$

- Partitionne les langages rationnels en un nombre fini de classes d'équivalence. (th. de Myhill-Nerode)

- Condition nécessaire et suffisante pour l'apprentissage actif.

Relation à la Myhill-Nerode - pomsets

- Un pomset c contenant une seule occurrence de ξ est un contexte distingueur des pomsets p_1 et p_2 de L ssi :

$$p_1 \ o_\xi \ c \in L \neq p_2 \ o_\xi \ c \in L$$

Relation à la Myhill-Nerode - pomsets

- Un pomset c contenant une seule occurrence de ξ est un contexte distingueur des pomsets p_1 et p_2 de L ssi :

$$p_1 \ o_\xi \ c \in L \neq p_2 \ o_\xi \ c \in L$$

- **Relation à la Myhill-Nerode** :

$$\forall (p_1, p_2) \in L^2, p_1 \sim_L p_2 \iff \nexists c \in C, p_1 \ o_\xi \ c \in L \neq p_2 \ o_\xi \ c \in L$$

Relation à la Myhill-Nerode - pomsets

- Un pomset c contenant une seule occurrence de ξ est un contexte distingueur des pomsets p_1 et p_2 de L ssi :

$$p_1 \ o_\xi \ c \in L \neq p_2 \ o_\xi \ c \in L$$

- **Relation à la Myhill-Nerode** :

$$\forall (p_1, p_2) \in L^2, p_1 \sim_L p_2 \iff \nexists c \in C, p_1 \ o_\xi \ c \in L \neq p_2 \ o_\xi \ c \in L$$

- Partitionne les langages de pomsets reconnaissables en un nombre fini de classes d'équivalence.

Relation à la Myhill-Nerode - pomsets

- Un pomset c contenant une seule occurrence de ξ est un contexte distingueur des pomsets p_1 et p_2 de L ssi :

$$p_1 \ o_\xi \ c \in L \neq p_2 \ o_\xi \ c \in L$$

- **Relation à la Myhill-Nerode** :

$$\forall (p_1, p_2) \in L^2, p_1 \sim_L p_2 \iff \nexists c \in C, p_1 \ o_\xi \ c \in L \neq p_2 \ o_\xi \ c \in L$$

- Partitionne les langages de pomsets reconnaissables en un nombre fini de classes d'équivalence.

- On peut donc appliquer des méthodes d'apprentissage actif sur les langages de pomsets.

Pomset recogniser

- **Bimonoïde :** $(M, \bullet, ||, \mathbf{1})$

Pomset recogniser

- **Bimonoïde :** $(M, \bullet, ||, \mathbf{1})$
- **Pomset recogniser :** $PR = (M, \bullet, ||, \mathbf{1}, i, f)$

Pomset recogniser

- **Bimonoïde :** $(M, \bullet, ||, \mathbf{1})$
- **Pomset recogniser :** $PR = (M, \bullet, ||, \mathbf{1}, i, f)$

ex :

$\bullet$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_a
q_b	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	q_1	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$

$\|\|$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	q_1	$q_\perp$	$q_\perp$	q_a
q_b	q_1	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$

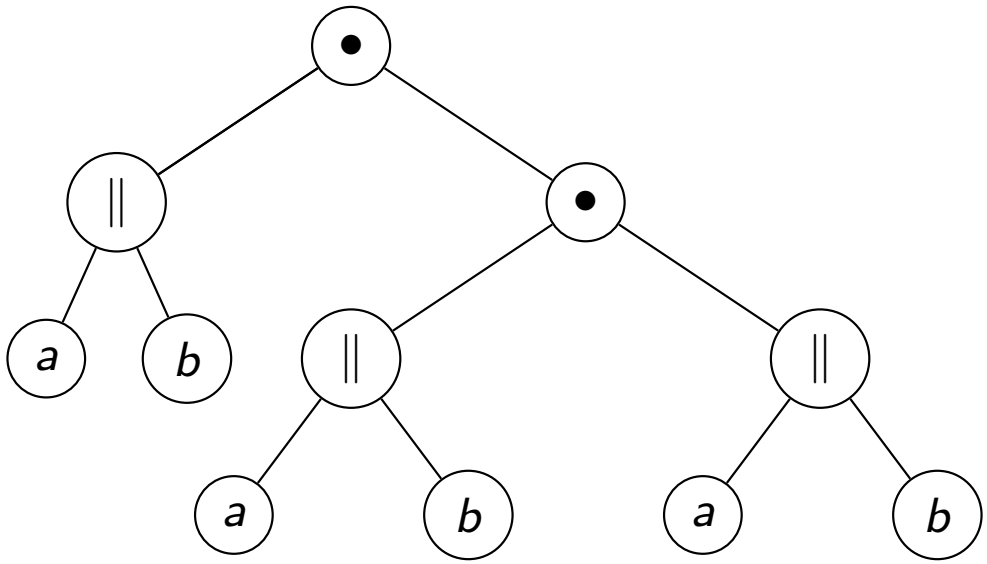
$f = \{q_1, \mathbf{1}\}$
$i(a) = q_a,\ i(b) = q_b$

Reconnaît $L = (a||b)^* = \{\epsilon, a||b, (a||b)(a||b), (a||b)(a||b)(a||b), ...\}$.

Requête d'appartenance

$\bullet$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_a
q_b	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	q_1	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$

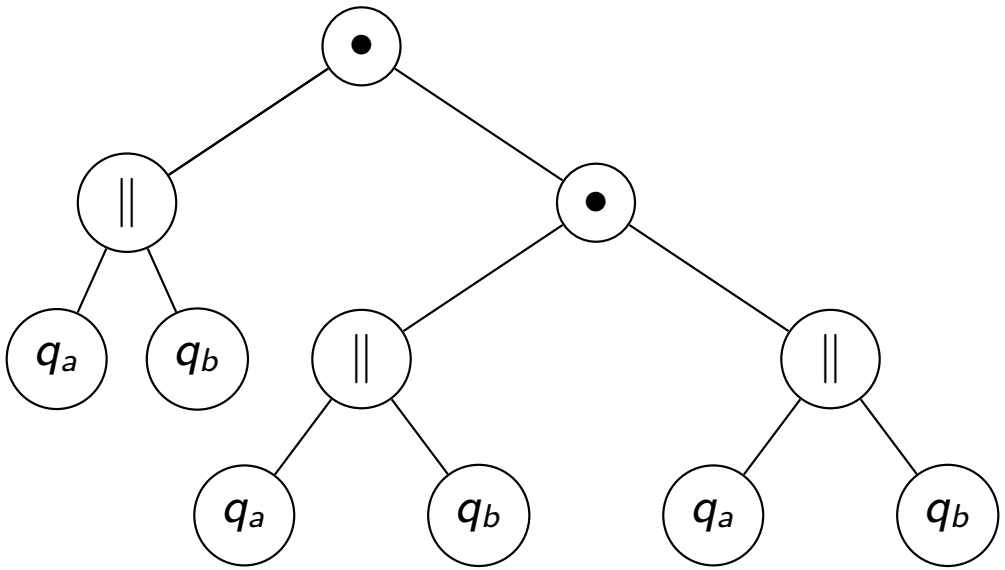
$\|\|$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	q_1	$q_\perp$	$q_\perp$	q_a
q_b	q_1	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$



Requête d'appartenance

$\bullet$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_a
q_b	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	q_1	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$

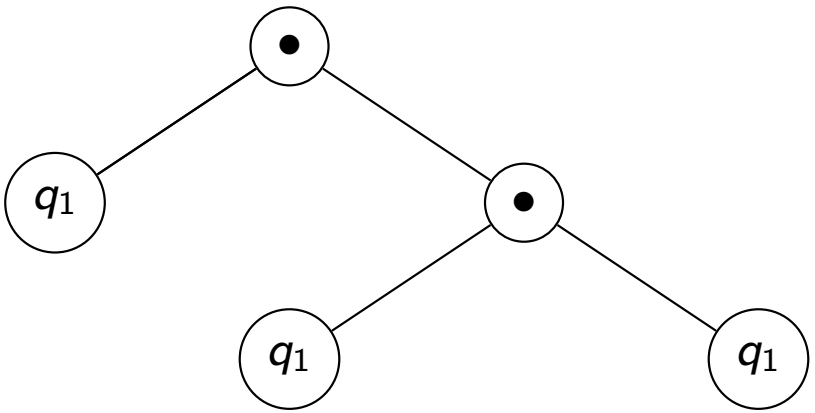
$\|\|$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	q_1	$q_\perp$	$q_\perp$	q_a
q_b	q_1	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$



Requête d'appartenance

$\bullet$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_a
q_b	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	q_1	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$

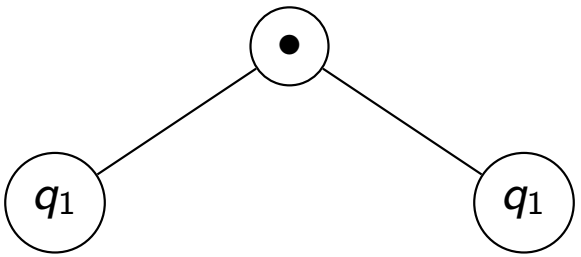
$\|\|$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	q_1	$q_\perp$	$q_\perp$	q_a
q_b	q_1	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$



Requête d'appartenance

$\bullet$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_a
q_b	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	q_1	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$

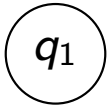
$\|\|$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	q_1	$q_\perp$	$q_\perp$	q_a
q_b	q_1	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$



Requête d'appartenance

$\bullet$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_a
q_b	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	q_1	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$

$\|\|$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$
q_a	$q_\perp$	q_1	$q_\perp$	$q_\perp$	q_a
q_b	q_1	$q_\perp$	$q_\perp$	$q_\perp$	q_b
q_1	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	q_1
$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$	$q_\perp$
$\mathbf{1}$	q_a	q_b	q_1	$q_\perp$	$\mathbf{1}$



Pomset recogniser

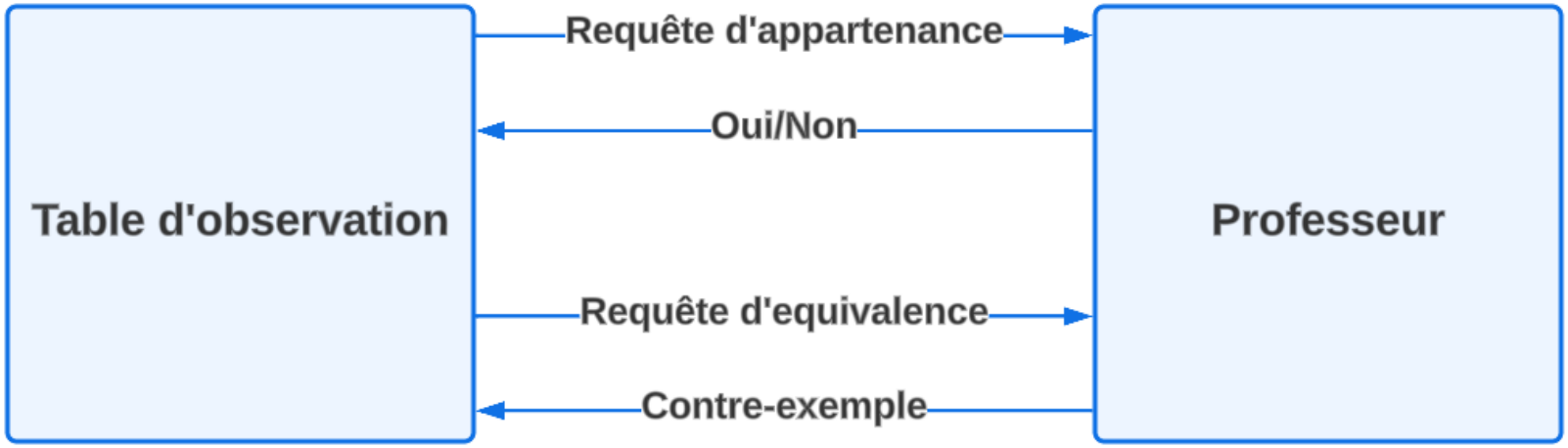
- Cas particulier des **automates d'arbres**.
- L(automates branchants acycliques) $\subset$ L(pomset recognisers) $\subset$ L(automates branchants généraux).
- Respecte le th. de Myhill-Nerode.
- Requête d'appartenance par simple évaluation d'AST.

L^* - principe

- Construit un DFA minimal équivalent à un modèle en un nombre de requêtes polynomial.

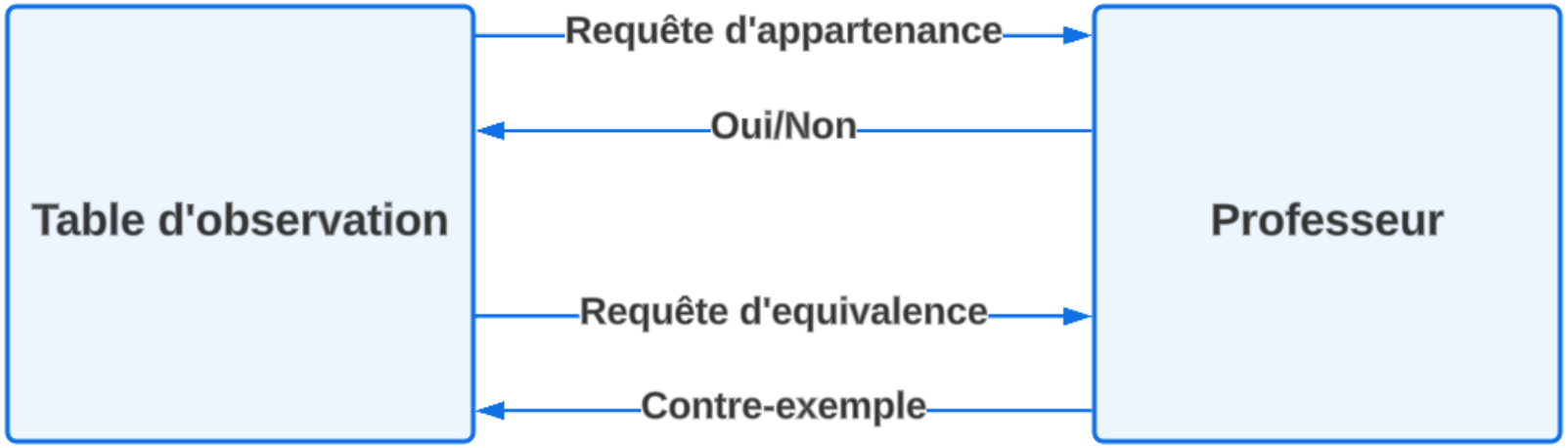
L^* - principe

- Construit un DFA minimal équivalent à un modèle en un nombre de requêtes polynomial.



L^* - principe

- Construit un DFA minimal équivalent à un modèle en un nombre de requêtes polynomial.



- L_p^* apprend un pomset recogniser.

L_p^* - table d'observation

- **Table d'observation :** (S, S^+, E, T)

L_p^* - table d'observation

- **Table d'observation :** (S, S^+, E, T)

  ex :

	ξ	$\xi \| \| b$	$a \| \| \xi$	$(a \bullet \xi) \| \| b$
ϵ	0	0	0	1
a	0	1	0	0
b	0	0	1	0
b||a	1	0	0	0
a+a	0	0	0	0
a||a	0	0	0	0
b+a	0	0	0	0
...	...	...	...	...

Table d'observation correspondant au langage $(a||b)^*$

L_p^* - requête d'équivalence

- Isomorphismes de bimonoïdes.

L_p^* - requête d'équivalence

- Isomorphismes de bimonoïdes.
- Méthode naïve : force brute.
- Il est impératif de trouver de meilleures méthodes. $\rightarrow$ probabiliste, algébrique, w-méthode ...

L_p^* - déroulé

Algorithm 1 $L_p^*()$:

$S = \{\epsilon\}$
$E = \{\xi\}$
while true **do**
  while !*closed or* !*associative* **do**
    make_closed()
    make_associative($\bullet$)
    make_associative($||$)
  $H = create\_hypothesis()$
  $c = find\_counter\_example(H)$
  if $c ==$ *null* **then**
    return H
  $c = handle\_counter\_example(c)$
  $E.add(c)$

$O(n^3 + nm + kn)$

L_p^* - déroulé

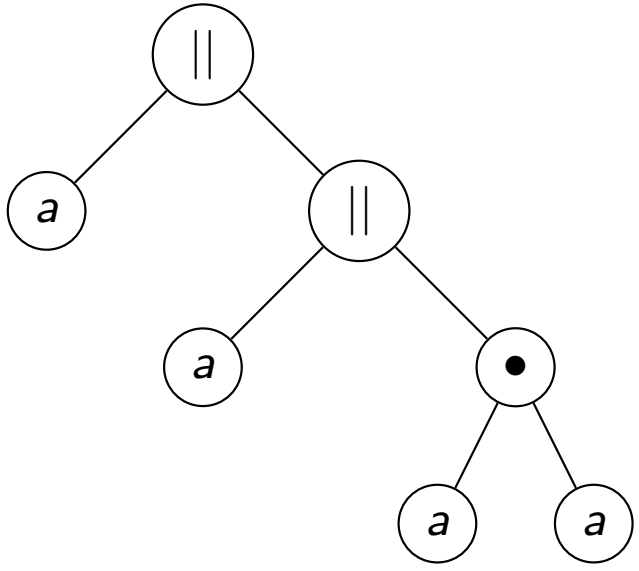
On apprend $L = \{a, aa, a||a\}$:

	ξ
ϵ	0
a	1
aa	1
a||a	1

L_p^* - déroulé

On apprend $L = \{a, aa, a||a\}$:

	ξ
ϵ	0
a	1
aa	1
a||a	1

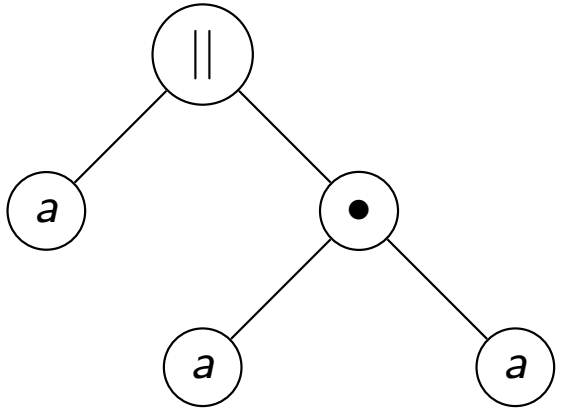


contre-exemple: $a||a||(a \bullet a) = a||a \; o_\xi \; \xi||(a \bullet a)$

L_p^* - déroulé

On apprend $L = \{a, aa, a||a\}$:

	ξ
ϵ	0
a	1
aa	1
a||a	1

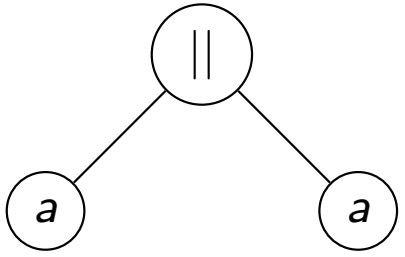


$a||(a \bullet a) = (a \bullet a)\ o_\xi\ \xi||a$

L_p^* - déroulé

On apprend $L = \{a, aa, a||a\}$:

	ξ
ϵ	0
a	1
aa	1
a||a	1

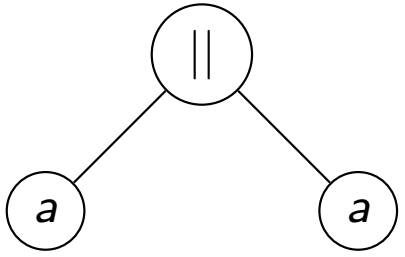


$a||a = a\ o_\xi\ a||\xi$

L_p^* - déroulé

On apprend $L = \{a, aa, a||a\}$:

	ξ	$a\|\|\xi$
ϵ	0	1
a	1	1
aa	1	0
a||a	1	0

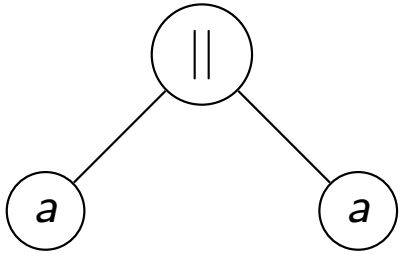


$a||a = a\ o_\xi\ a||\xi$

L_p^* - déroulé

On apprend $L = \{a, aa, a||a\}$:

	ξ	$a\|\|\xi$
ϵ	0	1
a	1	1
aa	1	0
a||a	1	0



$a||a = a\ o_\xi\ a||\xi$

- **Meilleur algorithme : L^λ ?**

Ce que nous voulions faire

- Proposer une implementation optimale des pomsets.
- Implémenter les PR et leurs algorithmes.
- Implémenter une extension de L^* sur les pomset recognisers en c++.
- Proposer une extension similaire de L^λ.
- Trouver une manière optimale de comparer les bimonoides.
- Décrire un générateur aléatoire de PR.
- Implémenter la conversion des PR en automates branchants.

Ce que nous avons fait

- Proposer une implementation optimale des pomsets.
- Implémenter les PR et leurs algorithmes.
- Implémenter une extension de L^* sur les pomset recognisers en c++.
- Proposer une extension similaire de L^λ.
- Trouver une manière optimale de comparer les bimonoides.
- Décrire un générateur aléatoire de PR.
- Implémenter la conversion des PR en automates branchants.

1. Dana Angluin. **Learning regular sets from queries and counterexamples**. Information and Computation, 75(2) :87–106, 1987.
2. A. Nerode. **Linear automaton transformations**. Proceedings of the American Mathematical Society, 9(4) :541–544, 1958.
3. Kamal Lodaya and Pascal Weil. **A kleene iteration for parallelism**. In Foundations of Software Technology and Theoretical Computer Science, 1998.
4. Tobias Kappé, Paul Brunet, Bas Luttik, Alexandra Silva, and Fabio Zanasi. **On series-parallel pomset languages : Rationality, context-freeness and automata**. Journal of Logical and Algebraic Methods in Programming, 103 :130–153, 2019.
5. Gerco van Heerdt, Tobias Kappé, Jurriaan Rot, and Alexandra Silva. **Learning Pomset Automata**, page 510–530. Springer International Publishing, 2021.
6. Hubert Comon, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, and Marc Tommasi. **Tree Automata Techniques and Applications**. 2008.
7. Kamal Lodaya and Pascal Weil. **A kleene iteration for parallelism**. In Vikraman Arvind and Ramaswamy Ramanujam, editors, Foundations of Software Technology and Theoretical Computer Science, 18th Conference, Chennai, India, December 17-19, 1998, Proceedings, volume 1530 of Lecture Notes in Computer Science, pages 355–366. Springer, 1998.

8. Falk Howar and Bernhard Steffen. **Active automata learning as black-box search and lazy partition refinement**. In Nils Jansen, Mariëlle Stoelinga, and Petra van den Bos, editors, A Journey from Process Algebra via Timed Automata to Model Learning - Essays Dedicated to Frits Vaandrager on the Occasion of His 60th Birthday, volume 13560 of Lecture Notes in Computer.
9. Vaughan Pratt. **Modelling concurrency with partial orders**. Internationaljournal of Parallel Programming.
10. Kamal Lodaya and Pascal Weil. **Rationality in algebras with a series operation**, 2001.
11. Kamal Lodaya and Pascal Weil. **Series-parallel languages and the bounded-width property**, 2000.
12. J. Ian Munro and Patrick K. Nicholson. **Succinct posets**.
13. Joel Jones. **Abstract syntax tree implementation idioms**. Pattern Languages of Program Design, 2003.
14. J.-M. Maranda. **Formal categories**. Canadian Journal of Mathematics, 17 :758–801, 1965.
15. Hassler Whitney. **2-Isomorphic Graphs**, pages 125–134. Birkhäuser Boston, Boston, MA, 1992.
16. Thomas Hanneforth, Andreas Maletti, and Daniel Quernheim. **Random generation of nondeterministic finite-state tree automata**. Electronic Proceedings in Theoretical Computer Science, 134 :11–16, November 2013.
17. D. E. Knuth and P. B. Bendix. **Simple Word Problems in Universal Algebras**, pages 342–376. Springer Berlin Heidelberg, Berlin, Heidelberg, 1983.