Caractérisation logique de la complexité d'un automate cellulaire

Théo Grente
Travaux en collaboration avec Étienne Grandjean

GREYC - Université de Caen Normandie

8 mars 2019

Objectifs

Obtenir une caractérisation d'un modèle de calcul (complexité calculatoire) indépendante des machines (complexité descriptive).

Exemple : Langages reconnaissables sur automate fini
= Langages réguliers
= Langages définissables dans la logique MSO [Büchi,1960]

Modèle de calcul étudié : les automates cellulaires.

Motivations

Pourquoi s'intéresser aux automates cellulaires ?

- Ils sont le modèle par excellence du calcul parallèle et local.
- Ils permettent de définir des classes de petite complexité qui capturent des problèmes naturels comme le produit d'entiers ou les langages linéaires algébriques.

Motivations

Pourquoi caractériser en logique les classes de complexité ?

- Pour définir naturellement ces classes de façon intrinsèque/indépendante des calculs.
- Afin de construire par la logique des programmes vérifiés et de complexité déterminée.

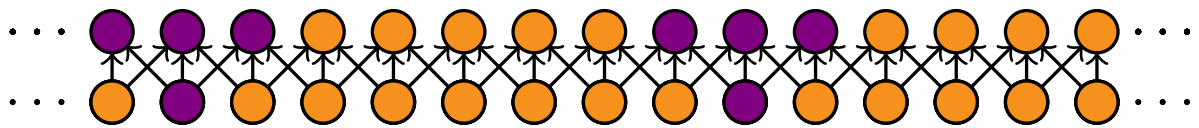
Plan de l'exposé

Plan de l'exposé

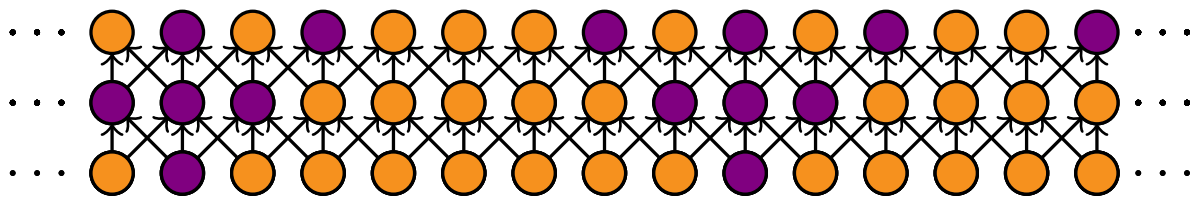
- Automate cellulaire de dimension 1 : ruban de cellules potentiellement infini.

- Automate cellulaire de dimension 1 : ruban de cellules potentiellement infini.
- Chaque cellule se trouve dans un état donné, évoluant (ou non) au cours du temps.

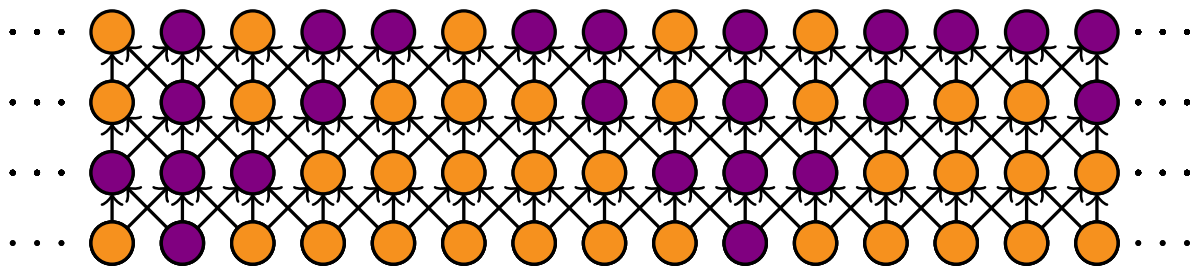
- Automate cellulaire de dimension 1 : ruban de cellules potentiellement infini.
- Chaque cellule se trouve dans un état donné, évoluant (ou non) au cours du temps.
- Modèle de calcul parallèle et local : une fonction de transition locale est appliquée de façon synchrone à chaque cellule et à chaque pas de temps.



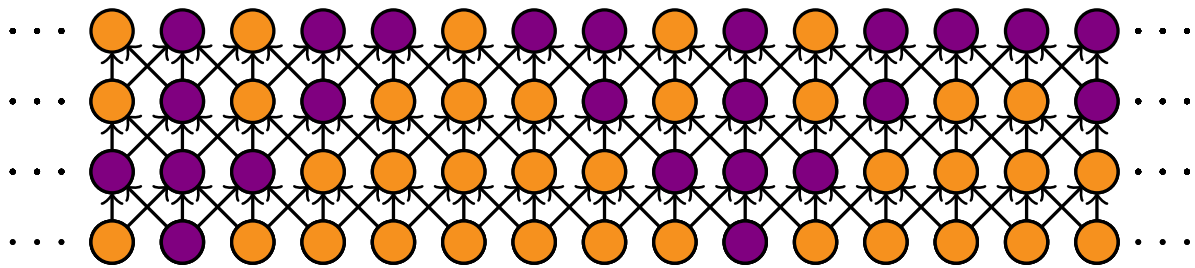
- Automate cellulaire de dimension 1 : ruban de cellules potentiellement infini.
- Chaque cellule se trouve dans un état donné, évoluant (ou non) au cours du temps.
- Modèle de calcul parallèle et local : une fonction de transition locale est appliquée de façon synchrone à chaque cellule et à chaque pas de temps.



- Automate cellulaire de dimension 1 : ruban de cellules potentiellement infini.
- Chaque cellule se trouve dans un état donné, évoluant (ou non) au cours du temps.
- Modèle de calcul parallèle et local : une fonction de transition locale est appliquée de façon synchrone à chaque cellule et à chaque pas de temps.

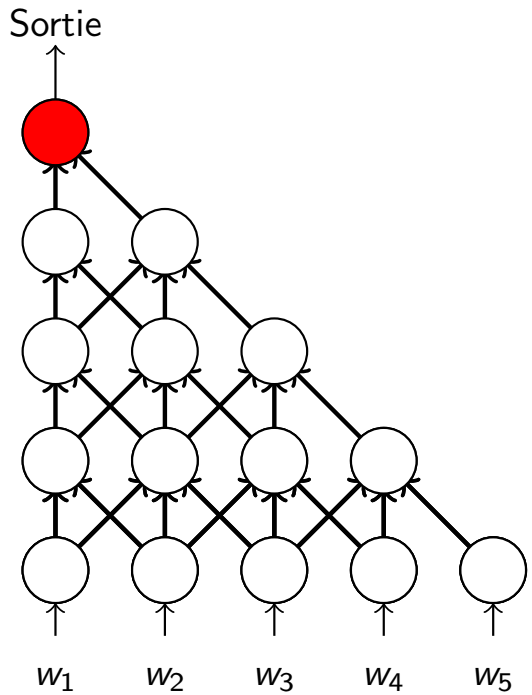


- Automate cellulaire de dimension 1 : ruban de cellules potentiellement infini.
- Chaque cellule se trouve dans un état donné, évoluant (ou non) au cours du temps.
- Modèle de calcul parallèle et local : une fonction de transition locale est appliquée de façon synchrone à chaque cellule et à chaque pas de temps.



Temps minimal et reconnaissance de langages

- L'automate est initialisé suivant le mot w testé : l'état initial de la cellule i dépend uniquement du i^{eme} bit de w.
- Un langage L est reconnu en temps réel par un automate cellulaire $\mathcal{A}$ si et seulement si pour tout mot $w \in L$, la cellule de sortie de $\mathcal{A}$ entre dans un état acceptant juste après avoir eu accès à toute l'information de w.



$\texttt{RealTime}_{\texttt{CA}}$

Plan de l'exposé

Contexte logique

Les formules de Horn sont une conjonction de clauses de Horn de la forme :

$$\alpha_1 \wedge \cdots \wedge \alpha_k \rightarrow \beta$$

ou

$$\alpha_1 \wedge \cdots \wedge \alpha_k \rightarrow \bot$$

où les α_i et β sont des atomes et $\bot$ le faux.

Exemple : calcul de la fermeture transitive R d'une relation binaire E.

$$\begin{cases} E(x,y) \rightarrow R(x,y) \\ R(x,y) \wedge E(y,z) \rightarrow R(x,z) \end{cases}$$

Logique du prédécesseur sur des structures mots

Structure utilisée pour représenter un mot $w = w_0 \ldots w_{n-1} \in \Sigma^*$:

$$\langle w \rangle = ([1, n]; (Q_s)_{s \in \Sigma}, \mathtt{min}, \mathtt{max}, \mathtt{pred})$$

- $Q_s(x) \Leftrightarrow w_x = s$
- $\mathtt{pred}(x) = x - 1$
- $min = \{1\}$
- $max = \{n\}$

Logique du prédécesseur : formules de Horn de la forme $\Phi := \exists \mathbf{R} \forall x \forall y \psi(x, y)$ où $\mathbf{R}$ est une liste de prédicats binaires et ψ est une conjonction de clauses de Horn sur les variables x, y n'utilisant que l'opération de prédécesseur.

pred-ESO-Horn : la classe des formules du prédécesseur et par abus de notation des langages qu'elles définissent.

La logique du prédécesseur sur un exemple : le langage notBordered

$w \in \texttt{notBordered} \iff w$ n'a pas de préfixe propre égal à un suffixe propre.

notBordered est définissable dans la logique du prédécesseur par la formule :
$\Phi_{\texttt{notBordered}} := \exists \texttt{Border} \forall x \forall y \psi$
Où ψ est la conjonction des clauses :

- $\texttt{min}(y) \wedge \neg\texttt{min}(x) \wedge Q_s(x) \wedge Q_s(y) \rightarrow \texttt{Border}(x, y)$
- $\neg\texttt{min}(x) \wedge \neg\texttt{min}(y) \wedge \texttt{Border}(x-1, y-1) \wedge Q_s(x) \wedge Q_s(y) \rightarrow \texttt{Border}(x, y)$, pour tout $s \in \Sigma$;
- $\texttt{max}(x) \wedge \texttt{Border}(x, y) \rightarrow \bot$.

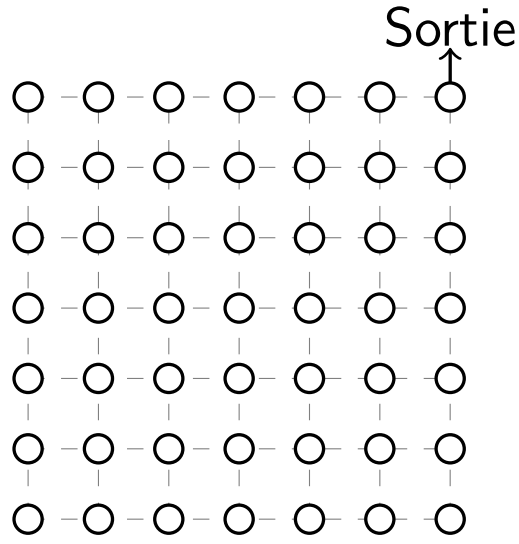
Avec $\texttt{Border}(x, y)$ vrai $\iff w_1 \dots w_y = w_{x-y+1} \dots w_x$

Plan de l'exposé

La logique du prédécesseur normalisée

Toute formule de *pred-ESO-Horn* peut être transformée en une formule équivalente dite normalisée imitant un calcul sur un circuit-grille orienté.

La formule $\Phi_{\texttt{notBordered}}$ normalisée :

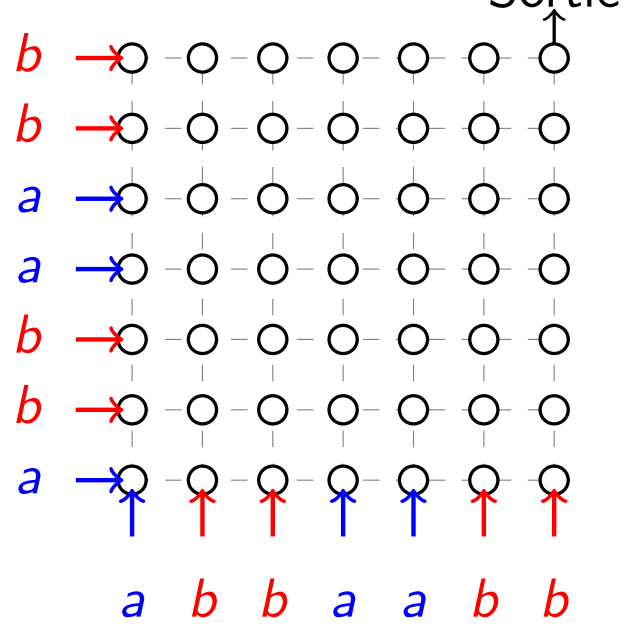


La logique du prédécesseur normalisée

Toute formule de *pred-ESO-Horn* peut être transformée en une formule équivalente dite normalisée imitant un calcul sur un circuit-grille orienté.

La formule $\Phi_{\texttt{notBordered}}$ normalisée :

- $\texttt{min}(y) \wedge Q_s(x) \rightarrow \texttt{Input}_s^x(x, y)$
  $\texttt{min}(x) \wedge Q_s(y) \rightarrow \texttt{Input}_s^y(x, y)$

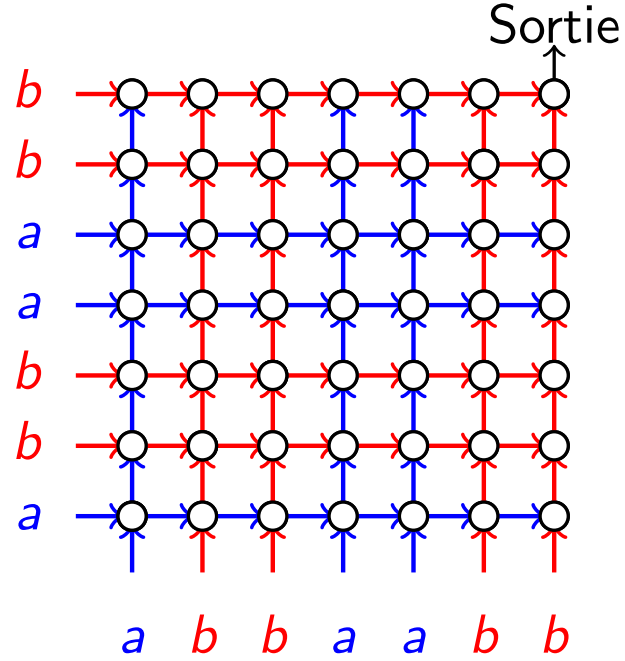


La logique du prédécesseur normalisée

Toute formule de *pred-ESO-Horn* peut être transformée en une formule équivalente dite normalisée imitant un calcul sur un circuit-grille orienté.

La formule $\Phi_{\texttt{notBordered}}$ normalisée :

- $\texttt{min}(y) \wedge Q_s(x) \rightarrow \texttt{Input}_s^x(x, y)$
  $\texttt{min}(x) \wedge Q_s(y) \rightarrow \texttt{Input}_s^y(x, y)$
- $\neg\texttt{min}(y) \wedge \texttt{Input}_s^x(x, y-1) \rightarrow \texttt{Input}_s^x(x, y)$
  $\neg\texttt{min}(x) \wedge \texttt{Input}_s^y(x-1, y) \rightarrow \texttt{Input}_s^y(x, y)$

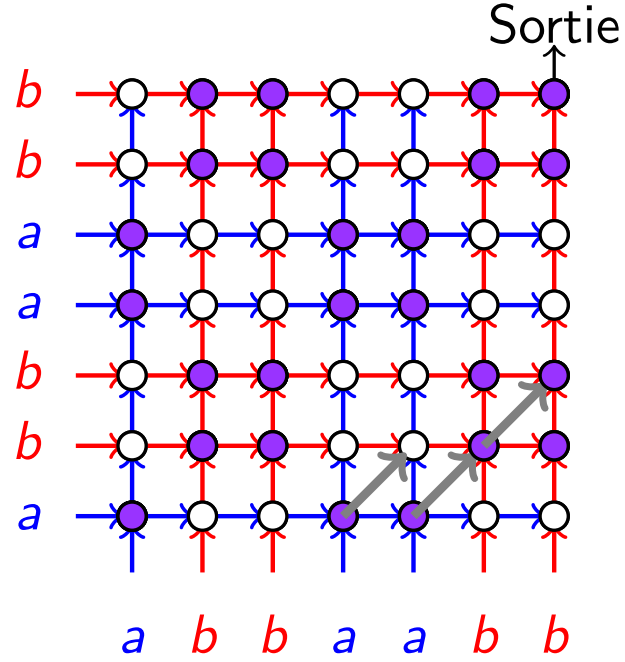


La logique du prédécesseur normalisée

Toute formule de *pred-ESO-Horn* peut être transformée en une formule équivalente dite normalisée imitant un calcul sur un circuit-grille orienté.

La formule $\Phi_{\texttt{notBordered}}$ normalisée :

- $\texttt{min}(y) \wedge Q_s(x) \rightarrow \texttt{Input}_s^x(x,y)$
  $\texttt{min}(x) \wedge Q_s(y) \rightarrow \texttt{Input}_s^y(x,y)$
- $\neg\texttt{min}(y) \wedge \texttt{Input}_s^x(x,y-1) \rightarrow \texttt{Input}_s^x(x,y)$
  $\neg\texttt{min}(x) \wedge \texttt{Input}_s^y(x-1,y) \rightarrow \texttt{Input}_s^y(x,y)$
- $\texttt{min}(y) \wedge \neg\texttt{min}(x) \wedge$
  $\texttt{Input}_s^x(x,y) \wedge \texttt{Input}_s^y(x,y) \rightarrow \texttt{Border}(x,y)$
  $\neg\texttt{min}(x) \wedge \texttt{Border}(x-1,y) \rightarrow \texttt{Border}'(x,y)$
  $\neg\texttt{min}(y) \wedge \texttt{Border}'(x,y-1) \wedge$
  $\texttt{Input}_s^x(x,y) \wedge \texttt{Input}_s^y(x,y) \rightarrow \texttt{Border}(x,y)$

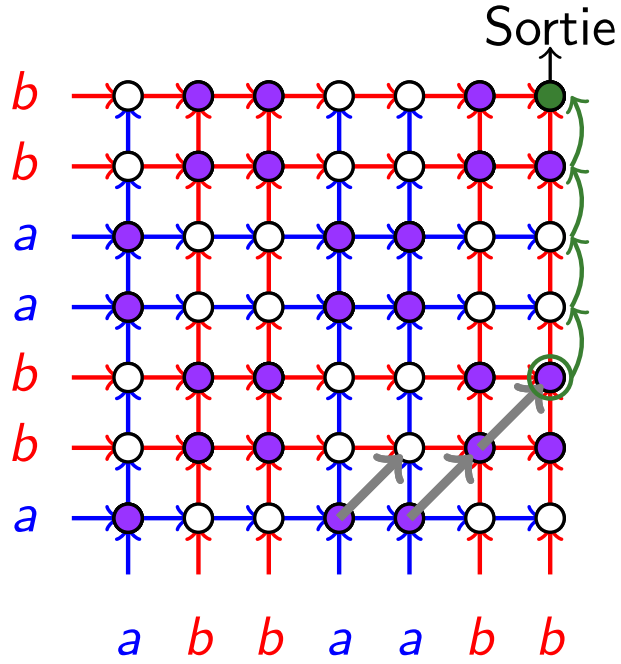


La logique du prédécesseur normalisée

Toute formule de *pred-ESO-Horn* peut être transformée en une formule équivalente dite normalisée imitant un calcul sur un circuit-grille orienté.

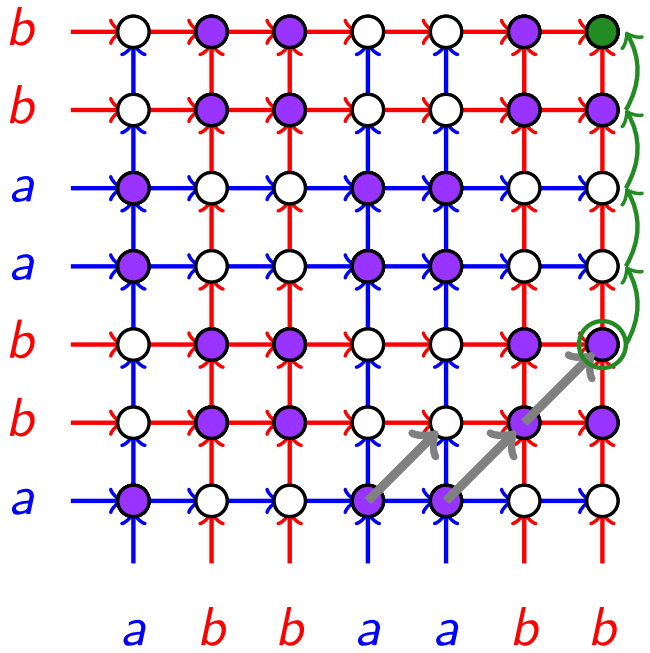
La formule $\Phi_{\texttt{notBordered}}$ normalisée :

- $\texttt{min}(y) \wedge Q_s(x) \rightarrow \texttt{Input}_s^x(x,y)$
  $\texttt{min}(x) \wedge Q_s(y) \rightarrow \texttt{Input}_s^y(x,y)$
- $\neg\texttt{min}(y) \wedge \texttt{Input}_s^x(x,y-1) \rightarrow \texttt{Input}_s^x(x,y)$
  $\neg\texttt{min}(x) \wedge \texttt{Input}_s^y(x-1,y) \rightarrow \texttt{Input}_s^y(x,y)$
- $\texttt{min}(y) \wedge \neg\texttt{min}(x) \wedge$
  $\texttt{Input}_s^x(x,y) \wedge \texttt{Input}_s^y(x,y) \rightarrow \texttt{Border}(x,y)$
  $\neg\texttt{min}(x) \wedge \texttt{Border}(x-1,y) \rightarrow \texttt{Border}'(x,y)$
  $\neg\texttt{min}(y) \wedge \texttt{Border}'(x,y-1) \wedge$
  $\texttt{Input}_s^x(x,y) \wedge \texttt{Input}_s^y(x,y) \rightarrow \texttt{Border}(x,y)$
- $\texttt{max}(x) \wedge \texttt{Border}(x,y) \rightarrow \texttt{Contr}(x,y)$
  $\neg\texttt{min}(y) \wedge \texttt{Contr}(x,y-1) \rightarrow \texttt{Contr}(x,y)$
  $\texttt{max}(x) \wedge \texttt{max}(y) \wedge \texttt{Contr}(x,y) \rightarrow \bot$

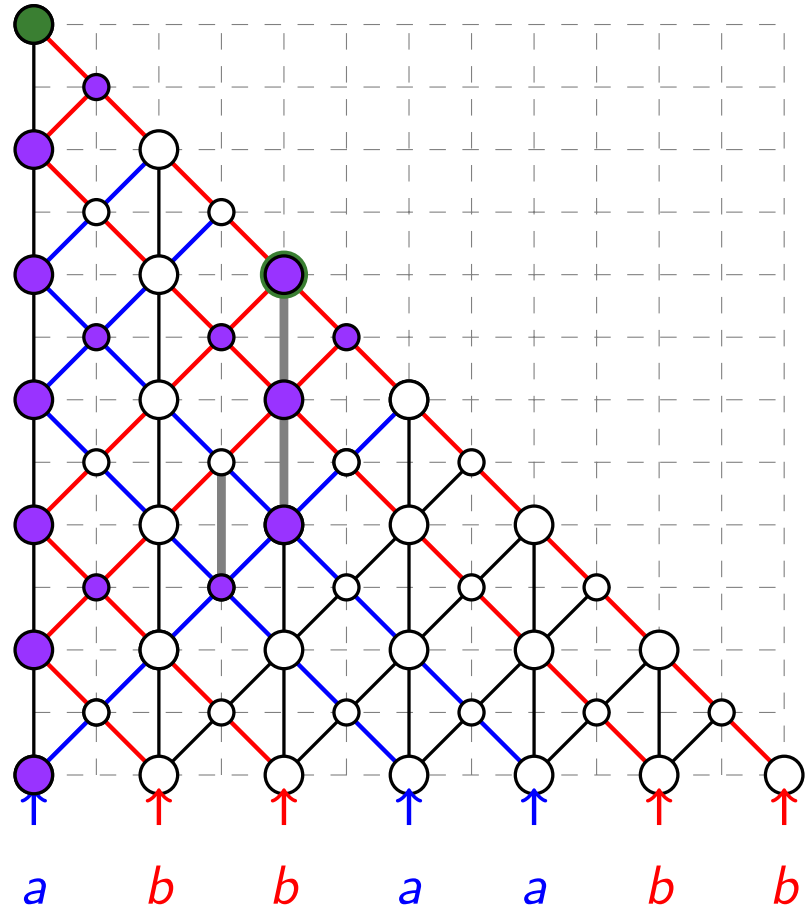


Bijection entre deux modèles de calcul

Calcul de circuit-grille



Temps réel 2-way



En résumé : une équivalence.

Reconnaissable en temps réel par un automate cellulaire

$\Updownarrow$

Reconnaissable par un circuit-grille orienté

$\Updownarrow$

Définissable dans la logique de Horn du prédécesseur