

# Mes histoires d'amour avec les travaux de Maxime Crochemore II

MARIE-PIERRE BÉAL

Institut Gaspard-Monge  
Université Paris-Est



Journées Algorithmique, combinatoire du texte et applications  
en bio-informatique, 26, 27 et 28 septembre 2007

# Content of the talk

- Compression par antidictionnaire
- Codage pour canaux contraints
- Minimisation des automates locaux ou AFT

# Compression par antidictionnaire

M. Crochemore, F. Mignosi, A. Restivo, S. Salemi, 2000

## Compression

*a b a a b a b a a b a a b a b a a b a b a*  
*a b a b a b* , 21

Antidictionnaire pour le texte

$$\mathcal{F} = \{aaa, \\ babab, \\ bb\}.$$

Requête : Étant donné  $v$ , existe-t-il  $u, a$  tel que  $ua \in \mathcal{F}$  ?





# Mots interdits minimaux

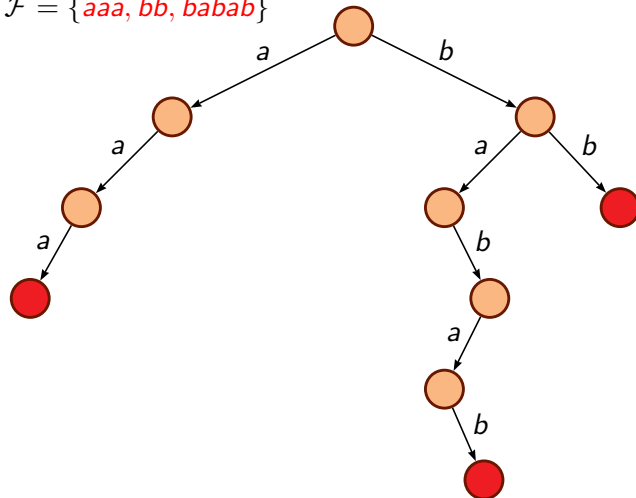
(M.-P. B, M. Crochemore, J. Fayolle, G. Fici, F. Fiorenzi, F. Mignosi, A. Restivo, S. Salemi, M. Sciortino, ...)

<i>b a b a b</i>	interdit
<i>b a b a</i>	autorisé
<i>a b a b</i>	autorisé

Un ensemble de mots interdits minimaux est antifactoriel.

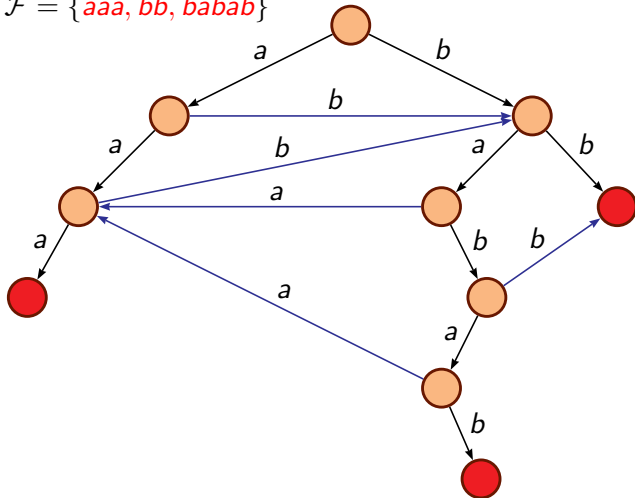
# Le trie des mots interdits minimaux

$\mathcal{F} = \{aaa, bb, babab\}$



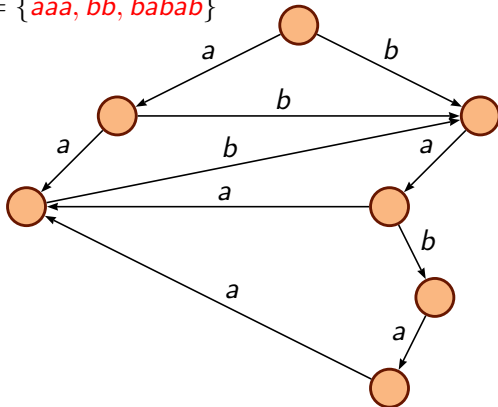
# L'automate $\mathcal{A}(AD)$

$$\mathcal{F} = \{aaa, bb, babab\}$$



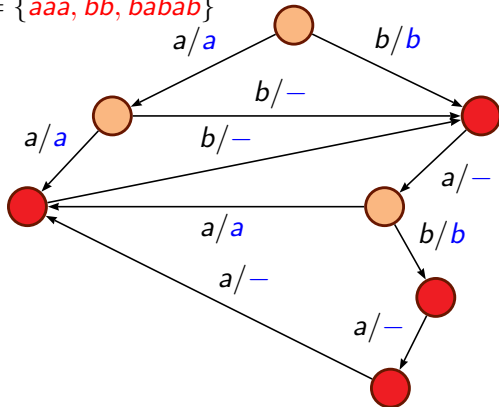
# L'automate $\mathcal{A}(AD)$

$$\mathcal{F} = \{aaa, bb, babab\}$$

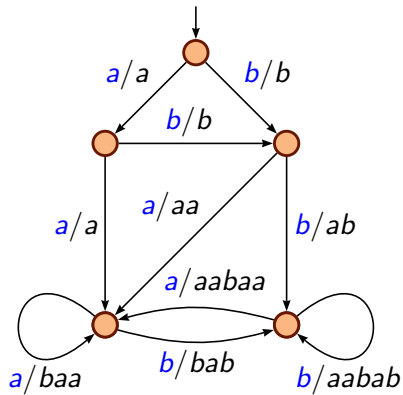
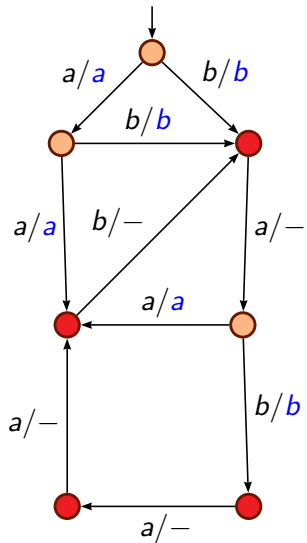


# De l'automate $\mathcal{A}(AD)$ au codeur

$$\mathcal{F} = \{aaa, bb, babab\}$$



# Codeur/Décodeur



- La compression et la décompression sont très rapides;

- La compression et la décompression sont très rapides;
- Si  $AD$  est fini, on construit  $\mathcal{A}(AD)$  en temps linéaire;

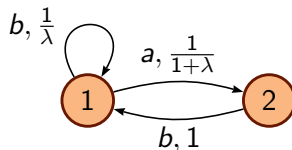
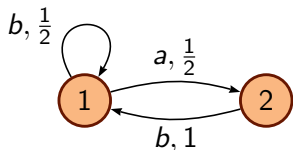
- La compression et la décompression sont très rapides;
- Si  $AD$  est fini, on construit  $\mathcal{A}(AD)$  en temps linéaire;
- Si  $AD$  est fini,  $\mathcal{A}(AD)$  est un automate  $(k - 1)$ -local, où  $k$  est la taille maximal d'un mot de  $AD$ ;

- La compression et la décompression sont très rapides;
- Si  $AD$  est fini, on construit  $\mathcal{A}(AD)$  en temps linéaire;
- Si  $AD$  est fini,  $\mathcal{A}(AD)$  est un automate  $(k - 1)$ -local, où  $k$  est la taille maximal d'un mot de  $AD$ ;
- On peut paralléliser le codage (le codage est à fenêtre glissante);

- La compression et la décompression sont très rapides;
- Si  $AD$  est fini, on construit  $\mathcal{A}(AD)$  en temps linéaire;
- Si  $AD$  est fini,  $\mathcal{A}(AD)$  est un automate  $(k - 1)$ -local, où  $k$  est la taille maximal d'un mot de  $AD$ ;
- On peut paralléliser le codage (le codage est à fenêtre glissante);
- On peut faire des recherches de motifs dans le texte compressé.

# Efficacité du codage

L'efficacité optimale n'est atteinte que pour les sources dites équilibrées (à gauche ci-dessous).



$$\lambda^2 = 1 + \lambda$$

Le codeur fournit  $(x, c(t), |t|)$  où

- $x$  est un codage de l' $AD$ ;
- $c(t)$  est le codage du texte;
- $|t|$  est la taille du texte.

## Contrôle de la taille de l' $AD$

- Pour avoir une bonne compression, il en faut pas prendre tous les mots interdits minimaux.
- Il y a un compromis entre la place prise pour l' $AD$  et le gain en compression.

# Stratégies pour le calcul d'un antidictionnaire

- À partir d'un texte, on peut construire en temps linéaire l'automate des facteurs , puis un automate des mots interdits minimaux .

# Stratégies pour le calcul d'un antidictionnaire

- À partir d'un texte, on peut construire en temps linéaire l'automate des facteurs de longueur  $\leq k$ , puis un automate des mots interdits minimaux de longueur  $\leq k$ .
- On peut couper des branches du trie de l'*AD* pour optimiser le gain en compression.

# Stratégies pour le calcul d'un antidictionnaire

- À partir d'un texte, on peut construire en temps linéaire l'automate des facteurs de longueur  $\leq k$ , puis un automate des mots interdits minimaux de longueur  $\leq k$ .
- On peut couper des branches du trie de l'*AD* pour optimiser le gain en compression.
- On peut auto-comprimer l'*AD*. Si  $v \in AD$ ,  $AD - \{v\}$  est un antidictionnaire pour  $v$ .

# Stratégies pour le calcul d'un antidictionnaire

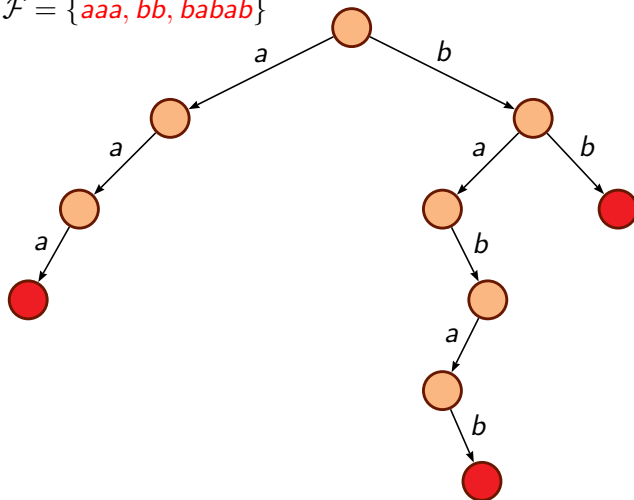
- À partir d'un texte, on peut construire en temps linéaire l'automate des facteurs de longueur  $\leq k$ , puis un automate des mots interdits minimaux de longueur  $\leq k$ .
- On peut couper des branches du trie de l'*AD* pour optimiser le gain en compression.
- On peut auto-comprimer l'*AD*. Si  $v \in AD$ ,  $AD - \{v\}$  est un antidictionnaire pour  $v$ .
  - Version statique : le texte est lu deux fois.

# Stratégies pour le calcul d'un antidictionnaire

- À partir d'un texte, on peut construire en temps linéaire l'automate des facteurs de longueur  $\leq k$ , puis un automate des mots interdits minimaux de longueur  $\leq k$ .
- On peut couper des branches du trie de l'*AD* pour optimiser le gain en compression.
- On peut auto-comprimer l'*AD*. Si  $v \in AD$ ,  $AD - \{v\}$  est un antidictionnaire pour  $v$ .
  - Version statique : le texte est lu deux fois.
  - Version dynamique : l'*AD* et le codage du texte sont calculés à la volée. Les trois optimisations ci-dessus peuvent se faire en même temps.

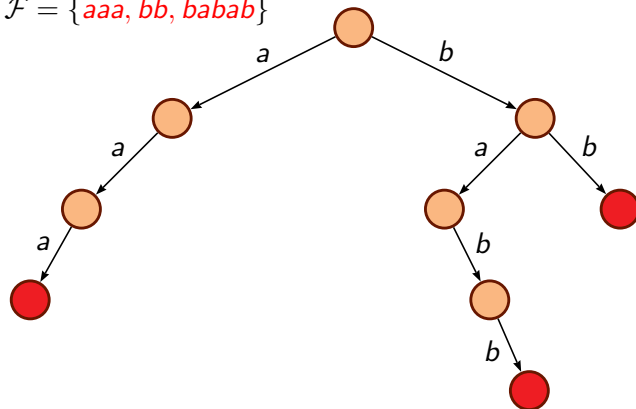
# Exemple d'auto-compression de l'AD

$\mathcal{F} = \{aaa, bb, babab\}$



# Exemple d'auto-compression de l'AD

$\mathcal{F} = \{aaa, bb, babab\}$



# Content of the talk

- Compression par antidictionnaire
- Codage pour canaux contraints
- Minimisation des automates locaux ou AFT

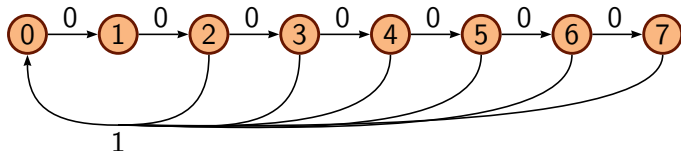
# Modèle de canaux contraints

## Automate déterministe local

Il existe  $k$  tel que deux chemins de même étiquette de longueur  $k$  arrivent toujours sur un même état.

## Système de type fini

Mots (bi-infini) acceptés par un automate local.



La contrainte  $[2, 7]$ ,  $\mathcal{F} = \{11, 101, 00000000\}$

# Canaux contraints avec positions libres

## Canal contraint stable par $U$ -flip

Pour chaque mot du canal (à shift près), l'échange de bits en position  $u \in U$  modulo une période  $p$  donne un mot qui reste dans le canal.

## Exemple

$$p = 3, U = \{1\}$$

0 0 1 0 0 0 0 1 1 0 ...

## Problème

Étant donné un canal de type fini  $S$ , trouver le plus grand sous-canal stable par  $U$ -flip (canal  $T$ ).

Fixer les positions libres à 1 (canal  $T^1$ ).

# Canaux contraints avec positions libres

## Canal contraint stable par $U$ -flip

Pour chaque mot du canal (à shift près), l'échange de bits en position  $u \in U$  modulo une période  $p$  donne un mot qui reste dans le canal.

## Exemple

$$p = 3, U = \{1\}$$

0 0 1 0 0 0 0 1 1 0 ...

## Problème

Étant donné un canal de type fini  $S$ , trouver le plus grand sous-canal stable par  $U$ -flip (canal  $T$ ).

Fixer les positions libres à 1 (canal  $T^1$ ).

# Canaux contraints avec positions libres

## Canal contraint stable par $U$ -flip

Pour chaque mot du canal (à shift près), l'échange de bits en position  $u \in U$  modulo une période  $p$  donne un mot qui reste dans le canal.

## Exemple

$$p = 3, U = \{1\}$$

0 1 1 0 1 0 0 0 1 0 ...

## Problème

Étant donné un canal de type fini  $S$ , trouver le plus grand sous-canal stable par  $U$ -flip (canal  $T$ ).

Fixer les positions libres à 1 (canal  $T^1$ ).

# Canaux périodiques de type fini PFT

Introduits par Moision and Siegel 2001.

La canaux  $T$  et  $T^1$  sont des canaux **periodiques de type fini**.

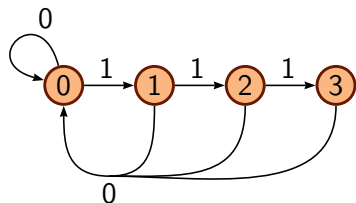
## Definition

Soient

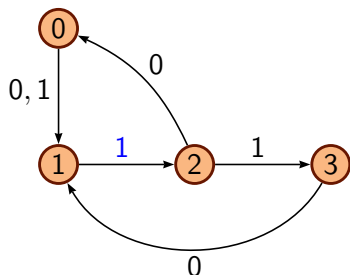
- $p$  la période,
- $U \subset \{0, 1, \dots, p-1\}$ ,
- $\mathcal{G} = (\mathcal{G}_i)_{0 \leq i \leq p-1}$  une suite d'ensembles finis de mots (les mots interdits en phase  $i$  modulo  $p$ ).

Le canal  $X_{\mathcal{G}}$  est l'ensemble des mots bi-infinis  $x$  tels que, à shift près,  $\mathcal{G}_i$  est un ensemble de mots interdits pour  $x$  en position  $i \bmod p$ .

# Exemple



Le canal  $MTR(3)$  de type fini  $X_{\mathcal{F}}$  avec  $\mathcal{F} = \{1111\}$



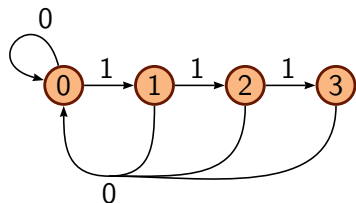
Le canal  $T^1$  pour le canal  $MTR(3)$  avec positions libres,  $p = 3$ ,  $U = \{1\}$ .  
 $T^1 = X_{\mathcal{G}}$  avec

$$\mathcal{G}_0 = \{1x11, x = 0, 1\},$$

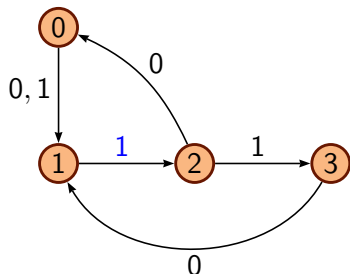
$$\mathcal{G}_1 = \{0, x11y, x, y = 0, 1\},$$

$$\mathcal{G}_2 = \{111x\}.$$

# Exemple



Le canal  $MTR(3)$  de type fini  $X_{\mathcal{F}}$  avec  $\mathcal{F} = \{1111\}$



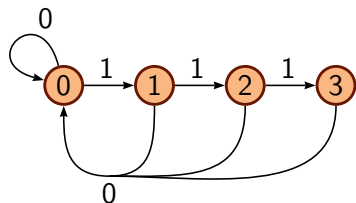
Le canal  $T^1$  pour le canal  $MTR(3)$  avec positions libres,  $p = 3$ ,  $U = \{1\}$ .  
 $T^1 = X_{\mathcal{G}}$  avec

$$\mathcal{G}_0 = \{1111, 1011\},$$

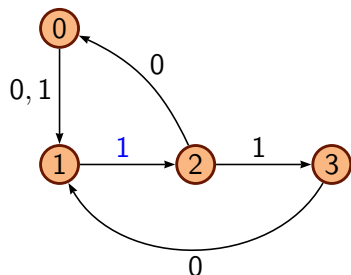
$$\mathcal{G}_1 = \{0, 1111, 0110, 0111\},$$

$$\mathcal{G}_2 = \{1111, 1110\}.$$

# Exemple



Le canal  $MTR(3)$  de type fini  
 $X_{\mathcal{F}}$  avec  $\mathcal{F} = \{1111\}$



Le canal  $T^1$  pour le canal  
 $MTR(3)$  avec positions libres,  
 $p = 3$ ,  $U = \{1\}$ .  
 $T^1 = X_{\mathcal{G}}$  avec

$$\mathcal{G}_0 = \{1111\},$$

$$\mathcal{G}_1 = \{0, 1111\},$$

$$\mathcal{G}_2 = \{1111\}.$$

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input:** un trie des mots interdits  $\mathcal{T}$ ;

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input:** un trie des mots interdits  $\mathcal{T}$ ;
  - **Output:** un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ ;
  - **Output**: un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .
  - $O(|\mathcal{T}|)$ , avec alphabet constant.

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ ;
  - **Output**: un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .
  - $O(|\mathcal{T}|)$ , avec alphabet constant.
- Problem 2 (M.-P. B., M. Crochemore, G. Fici, 2005)

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ ;
  - **Output**: un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .
  - $O(|\mathcal{T}|)$ , avec alphabet constant.
- Problem 2 (M.-P. B., M. Crochemore, G. Fici, 2005)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ , une paire  $p, U$  des positions libres;

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ ;
  - **Output**: un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .
  - $O(|\mathcal{T}|)$ , avec alphabet constant.
- Problem 2 (M.-P. B., M. Crochemore, G. Fici, 2005)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ , une paire  $p, U$  des positions libres;
  - **Output**: un automate déterministe acceptant le plus grand sous-canal stable  $T$  ou  $T^1$ .

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ ;
  - **Output**: un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .
  - $O(|\mathcal{T}|)$ , avec alphabet constant.
- Problem 2 (M.-P. B., M. Crochemore, G. Fici, 2005)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ , une paire  $p, U$  des positions libres;
  - **Output**: un automate déterministe acceptant le plus grand sous-canal stable  $T$  ou  $T^1$ .
  - Deux étapes

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input:** un trie des mots interdits  $\mathcal{T}$ ;
  - **Output:** un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .
  - $O(|\mathcal{T}|)$ , avec alphabet constant.
- Problem 2 (M.-P. B., M. Crochemore, G. Fici, 2005)
  - **Input:** un trie des mots interdits  $\mathcal{T}$ , une paire  $p, U$  des positions libres;
  - **Output:** un automate déterministe acceptant le plus grand sous-canal stable  $T$  ou  $T^1$ .
  - Deux étapes
    1. Construire  $p$  tries des mots interdits périodiques qui définissent  $T^1$ , temps linéaire (M.-P. B., M. Crochemore, L. Gasieniec, 2007);

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ ;
  - **Output**: un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .
  - $O(|\mathcal{T}|)$ , avec alphabet constant.
- Problem 2 (M.-P. B., M. Crochemore, G. Fici, 2005)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ , une paire  $p, U$  des positions libres;
  - **Output**: un automate déterministe acceptant le plus grand sous-canal stable  $T$  ou  $T^1$ .
  - Deux étapes
    1. Construire  $p$  tries des mots interdits périodiques qui définissent  $T^1$ , temps linéaire (M.-P. B., M. Crochemore, L. Gasieniec, 2007);
    2. **Constuction du canal à partir des tries**, temps linéaire.

- Problem 1 (M. Crochemore, F. Mignosi, A. Restivo, 1998)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ ;
  - **Output**: un automate déterministe acceptant le canal  $X_{\mathcal{T}}$ .
  - $O(|\mathcal{T}|)$ , avec alphabet constant.
- Problem 2 (M.-P. B., M. Crochemore, G. Fici, 2005)
  - **Input**: un trie des mots interdits  $\mathcal{T}$ , une paire  $p, U$  des positions libres;
  - **Output**: un automate déterministe acceptant le plus grand sous-canal stable  $T$  ou  $T^1$ .
  - Deux étapes
    1. Construire  $p$  tries des mots interdits périodiques qui définissent  $T^1$ , temps linéaire (M.-P. B., M. Crochemore, L. Gasieniec, 2007);
    2. **Construction du canal à partir des tries**, temps linéaire.
  - Temps linéaire

## Étape 2: des tries au canal PFT

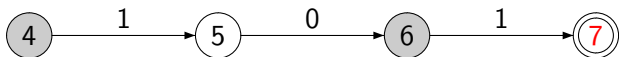
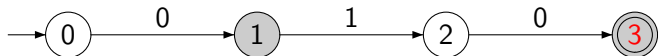
- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .

## Étape 2: des tries au canal PFT

- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .
- Exemple avec  $p = 2$ ,  $U = 1$ , et  $\mathcal{G}_0 = \{010\}$ ,  $\mathcal{G}_1 = \{101\}$ .

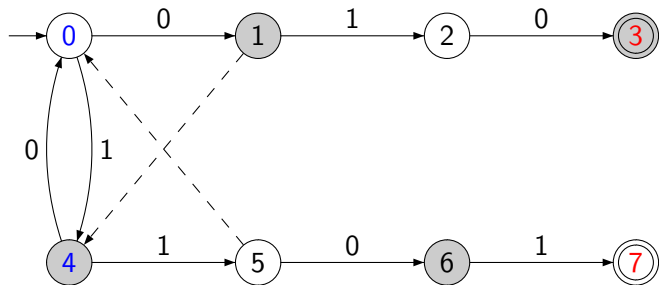
## Étape 2: des tries au canal PFT

- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .
- Exemple avec  $p = 2$ ,  $U = 1$ , et  $\mathcal{G}_0 = \{010\}$ ,  $\mathcal{G}_1 = \{101\}$ .



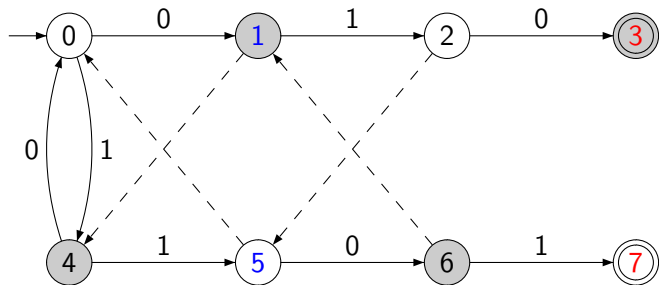
## Étape 2: des tries au canal PFT

- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .
- Exemple avec  $p = 2$ ,  $U = 1$ , et  $\mathcal{G}_0 = \{010\}$ ,  $\mathcal{G}_1 = \{101\}$ .



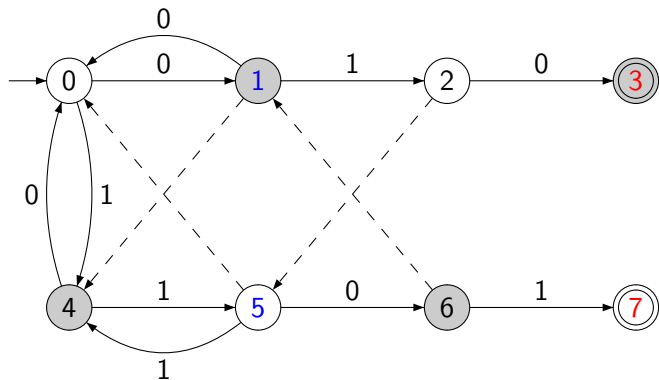
## Étape 2: des tries au canal PFT

- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .
- Exemple avec  $p = 2$ ,  $U = 1$ , et  $\mathcal{G}_0 = \{010\}$ ,  $\mathcal{G}_1 = \{101\}$ .



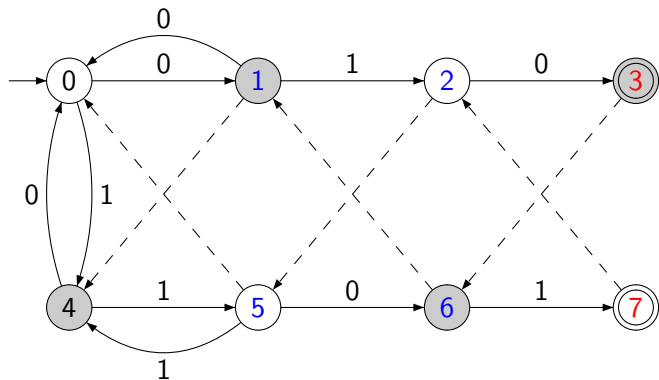
## Étape 2: des tries au canal PFT

- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .
- Exemple avec  $p = 2$ ,  $U = 1$ , et  $\mathcal{G}_0 = \{010\}$ ,  $\mathcal{G}_1 = \{101\}$ .



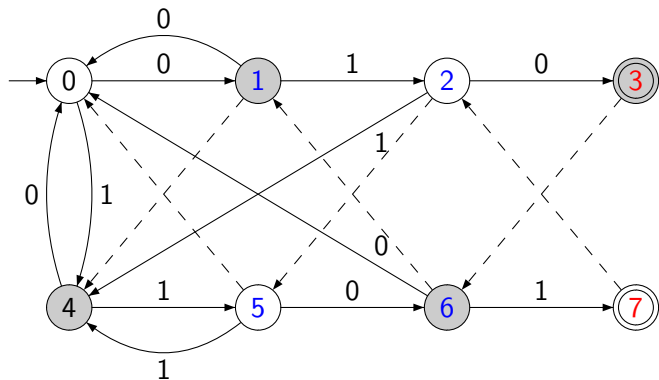
## Étape 2: des tries au canal PFT

- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .
- Exemple avec  $p = 2$ ,  $U = 1$ , et  $\mathcal{G}_0 = \{010\}$ ,  $\mathcal{G}_1 = \{101\}$ .



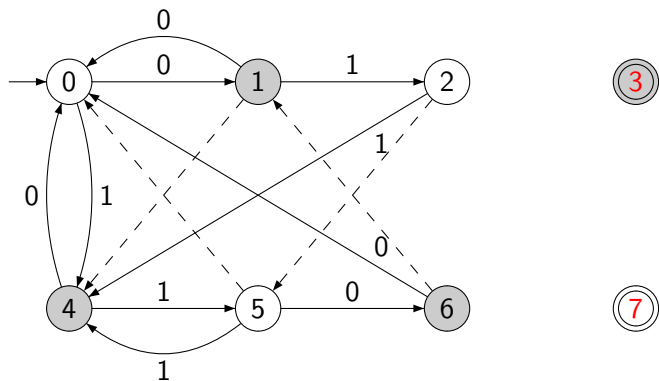
## Étape 2: des tries au canal PFT

- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .
- Exemple avec  $p = 2$ ,  $U = 1$ , et  $\mathcal{G}_0 = \{010\}$ ,  $\mathcal{G}_1 = \{101\}$ .



## Étape 2: des tries au canal PFT

- **Input:**  $p$  tries  $\mathcal{T}_i$ ,  $0 \leq i < p$ , acceptant  $\mathcal{G}_i$ ,  $(p, U)$ .
- Exemple avec  $p = 2$ ,  $U = 1$ , et  $\mathcal{G}_0 = \{010\}$ ,  $\mathcal{G}_1 = \{101\}$ .



# Content of the talk

- Compression par antidictionnaire
- Codage pour canaux contraints
- **Minimisation des automates locaux ou AFT**

# Minimisation des automates AFT

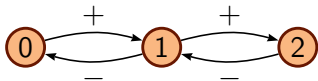
## Automate AFT

Un automate est presque de type fini (AFT) si

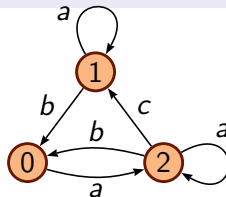
- Il est faiblement déterministe (ou déterministe avec délai fini) et faiblement co-déterministe;
- Il a un mot synchronisant;
- Il est irréductible (graphe fortement connexe).

## Canal AFT

Mots acceptés par un automate AFT.



Contrainte de charge (AFT)



Canal non AFT

## Automate AFT

Un automate est presque de type fini (AFT) si

- Il est faiblement déterministe (ou déterministe avec délai fini) et faiblement co-déterministe;
- Il a un mot synchronisant;
- Il est irréductible (graphe fortement connexe).

## Canal AFT

Mots acceptés par un automate AFT.

$SFT \subsetneq PFT \subsetneq AFT \subsetneq \text{canal sofique}$

# Minimisation des automates AFT

Un algorithme de minimisation (B., Crochemore 2007)

Un algorithme de minimisation en  $O(m \log n)$  pour les automates déterministes AFT (inclut les déterministes locaux).

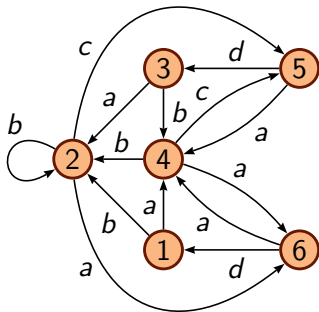
## Caractéristiques

- appartient à la classe des algorithmes de raffinement de partitions;
- procède par fusion (dynamique) d'états. Les parties de la partition sont fusionnées et non coupées comme dans Hopcroft.
- La complexité (avec  $n$  états,  $m \leq kn$  flèches,  $k$  lettres) est

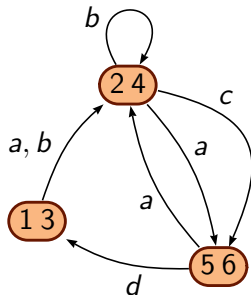
		Minimisation	Test de minimalité
tout automate	Hopcroft	$O(kn \log n)$	$O(kn \log n)$
acyclique	Revuz	$O(m + n + k)$	$O(m + n + k)$
AFT	B., Crochemore	$O(m \log n)$	$O(m)$

# Exemple

Soit  $p$  un état,  $\text{output}(p) = \{(a, q) \mid p \xrightarrow{a} q\}$ .



Un AFT déterministe



Son automate minimal

## Proposition

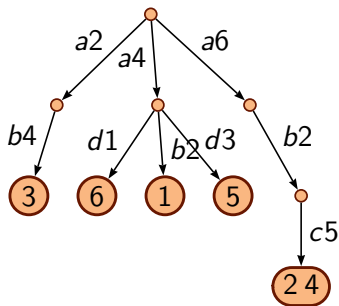
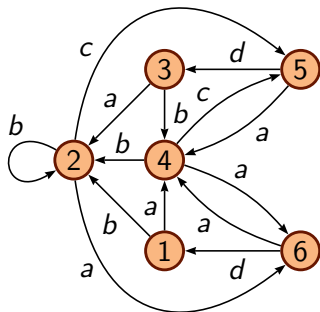
*Si un automate déterministe AFT n'est pas minimal, il existe deux états qui peuvent être fusionnés (qui ont même output).*

# Algorithme : construction et mise à jour d'un trie des "signatures"

Signature de l'état 1:  $\sigma(1) = a4b2$  (en ordre lexicographique)

Nœuds du trie : prefixes d'une signature

Feuilles du trie : ensemble des états ayant cette signature



C'est un algorithme de discrimination pour des multi-ensembles avec mises à jour.