REGAL: une bibliothèque pour la génération d'automates déterministes

Julien David

Mardi 30 octobre 2007

Générateurs

Motivations

Un générateur d'un ensemble d'objet permet de tester :

- Si un algorithme fonctionne sur tous les objets d'un ensemble où s'il existe des cas particulier.
- la distribution de la complexité d'un algorithme
- la proportion d'objets possédant certaines propriétés.
- ...

Problème : On doit cependant garantir l'**uniformité** parmi les objets d'une même taille

Méthode de Boltzmann (Duchon, Flajolet, Louchard, Schaeffer 2004)

Permet d'engendrer des objets de grandes tailles, prend peu de place en mémoire.

- Méthode générique
- N'engendre pas d'objets de taille fixe.
- $Pr(\gamma)$ est proportionelle à $x^{|\gamma|}$
  $\forall \gamma_1, \gamma_2$ tel que $|\gamma_1| = |\gamma_2|$, $Pr(\gamma_1) = Pr(\gamma_2)$
- Le paramètre x peut être déterminé en fonction de la taille moyenne que l'on souhaite obtenir.
- Nécessite peu de précalcul.
- Générateur de taille exacte : méthode par rejet.

Méthode par rejet

Soit deux ensembles A et E tels que $A \subset E$.
On engendre un objet γ dans E.

- si $\gamma \in A$, on garde l'objet
- sinon on le rejette et on en engendre un nouveau.

L'efficacité de cette méthode dépend de la proportion d'objets de A dans E.

Automates

Automate

Un automate est un quintuplet $\mathcal{A} =< \mathcal{Q}, A, \tau, I, F >$ où

- $\mathcal{Q}$ est l'ensemble des états
- A est l'alphabet.
- τ est l'ensemble des transitions de l'automate
- I est l'ensemble des états initiaux
- F est l'ensemble des états finals.

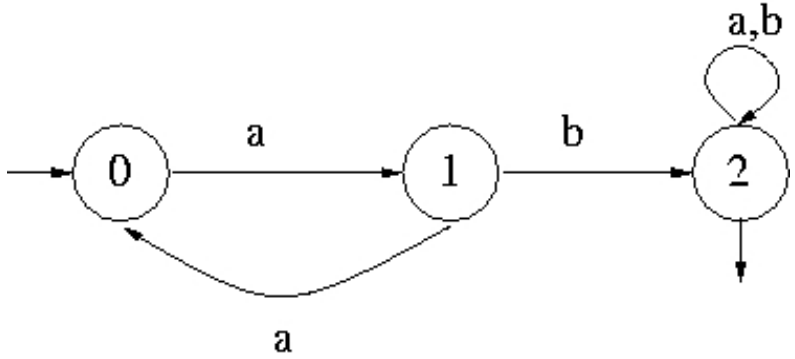


FIG.: Automate

Le langage reconnu est $\mathcal{L} = (aa)^* ab(a + b)^*$

Automate déterministe :

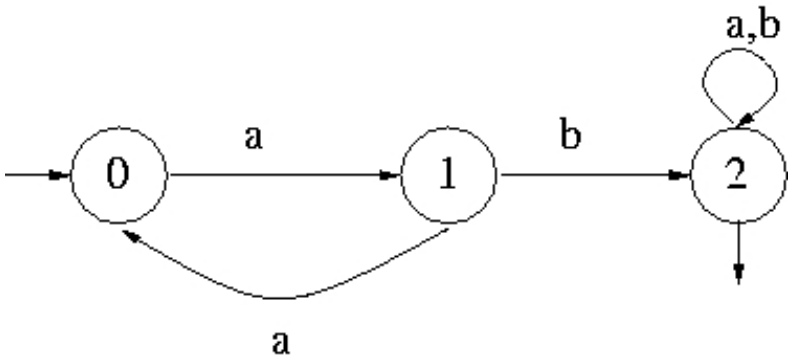


FIG.: Automate déterministe

Automate complet :

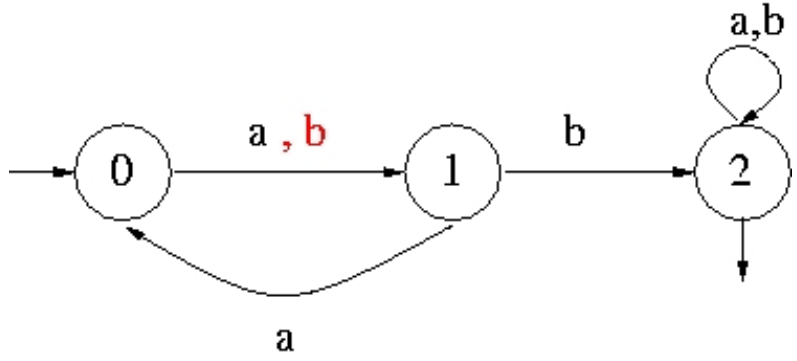


FIG.: Automate complet

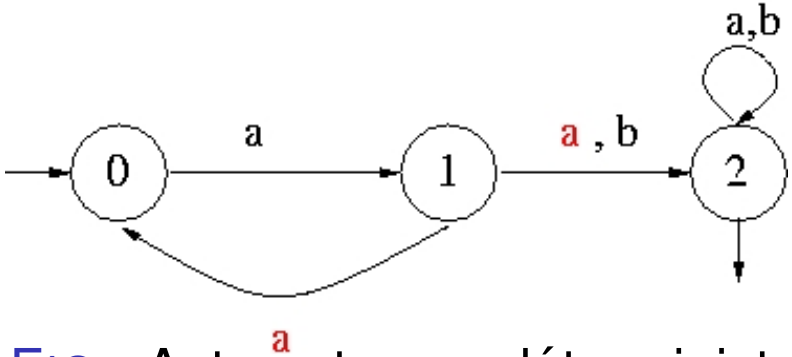


FIG.: Automate non déterministe

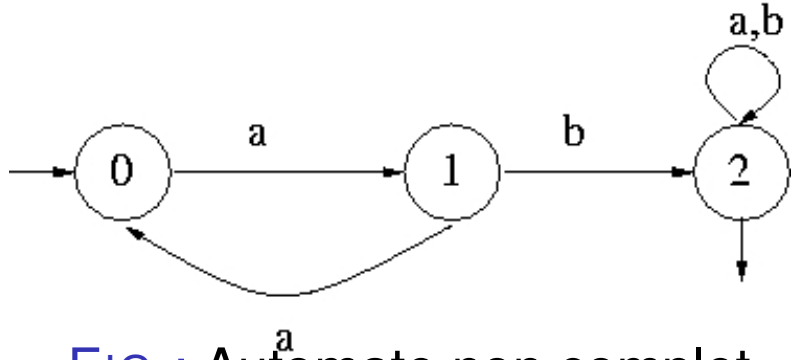


FIG.: Automate non complet

Automate accessible :

- Pour tout état $q \in \mathcal{Q}$, il existe un chemin allant d'un état initial jusqu'à q.

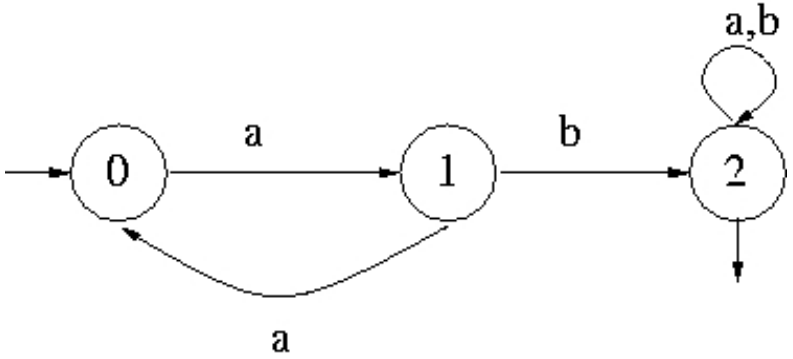


FIG.: Automate accessible

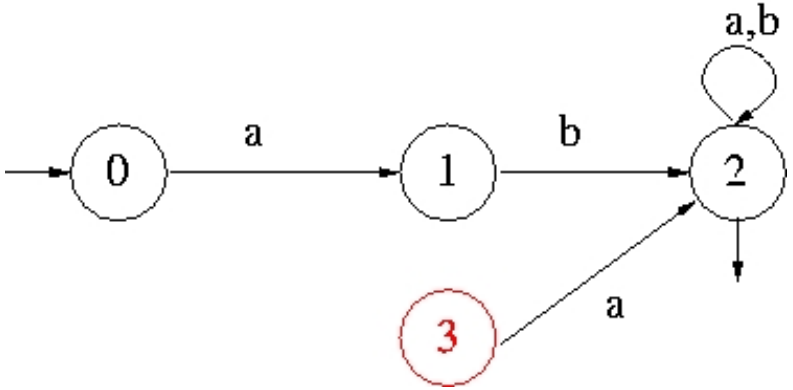


FIG.: Automate non accessible

Automate minimal :

- Un automate est minimal si on ne peut supprimer d'état sans modifier le langage reconnu par l'automate

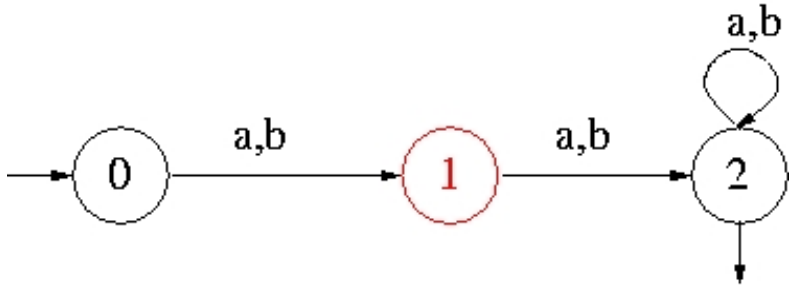


FIG.: Automate minimal

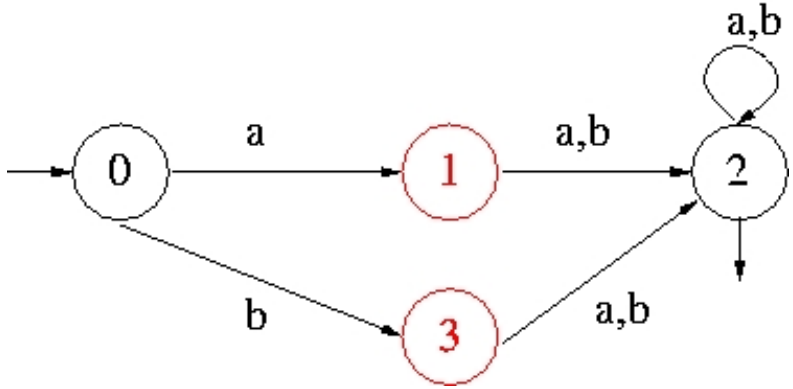


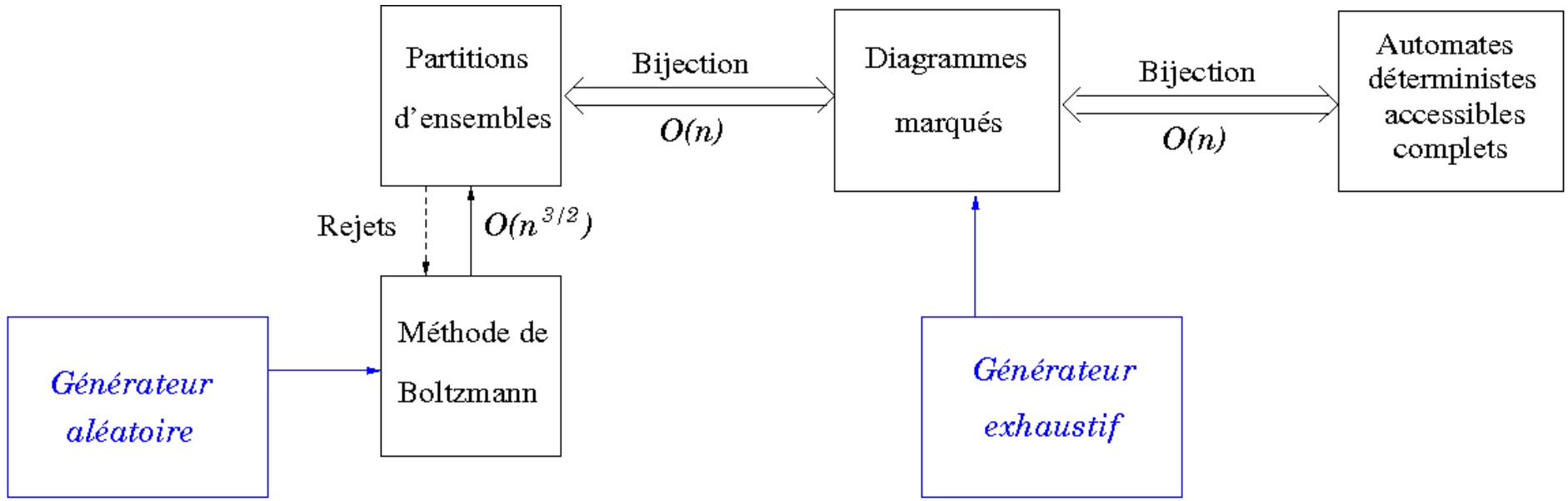
FIG.: Automate non minimal

Génération aléatoire : Problématique

On souhaite générer uniformément des automates déterministes, accessibles, complets de taille fixée.
On pourra ainsi :

- Étudier des propriétés sur les automates.
- Comparer différentes implantations d'un même algorithme.
- Étudier la complexité d'algorithmes en moyenne.

Algorithmes



Bassino–Nicaud[2006]

Nicaud[2000]
Champarnaud–Paranthoen[2005]

REGAL

Random and Exhaustive Generators for Automata - Library

REGAL

Bibliothèque en C++ de génération d'automates.
Téléchargeable à l'adresse : **http ://regal.univ-mlv.fr**

- L'utilisateur a la liberté de définir sa propre implantation d'automate.
- Facilement interfaçable avec une autre bibliothèque.
- Les générateurs sont rapides.
- La prise en main de la librairie est relativement simple.

Rapidité de la génération

Un million d'automates à 5 états $\sim$ 17 secondes.
Temps moyen de génération d'un automate de grande taille :

Taille de l'automate	1 000	10 000	100 000
Temps Moyen	0.04 s	1.43 s	110.23 s

Tests effectués sur un Intel $2,8$ Ghz

Générateur exhaustif :

- Automates déterministes accessibles
  48, 1 728, 83 968, 5 141 600, 379 618 560, 32 791 333 120
- Automates minimaux
  24, 1 028, 56 014, 3 705 306, 286 717 796, 25 493 886 852
- Automates fortement connexes
  18, 1 184, 55 888, 3 405 792, 252 853 248, 22 057 601 792

Générateur aléatoire :

- Automates déterministes accessibles
- Automates minimaux [efficacité non démontrée]

Taille	100	500	1 000	5 000
Automates minimaux (%)	85.06	85.32	85.09	85.32

- Automates co-accessibles [efficacité non démontrée]

Co-accessibles (%)	99.70	99.92	99.98	100.00

- Automates fortement connexes [Korshunov '78]

Fortement connexes (%)	78.85	79.57	79.71	79.77

Moore vs Hopcroft

Moore et de Hopcroft sont des algorithmes de minimisation d'automates qui fonctionnent sur le même principe
Complexité dans le pire des cas :

- Moore (1956) : $\mathcal{O}(n^2)$
- Hopcroft (1971) : $\mathcal{O}(n \log n)$

Moore vs Hopcroft

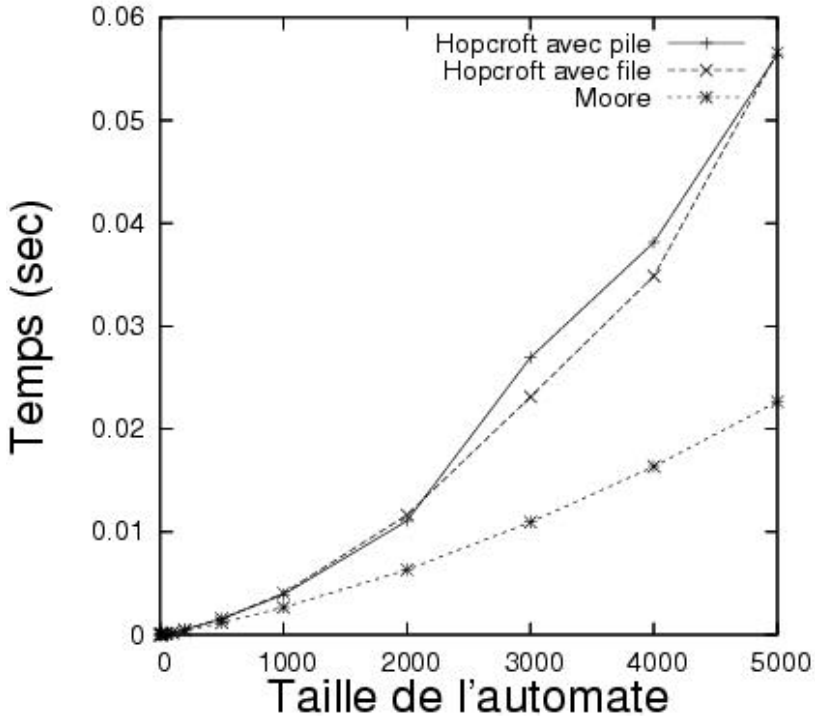


FIG.: Complexité en temps des algorithmes de Moore et Hopcroft

Conclusion et perspectives

- REGAL peut générer rapidement des automates de grandes tailles et en grande quantité.
- Son utilisation permet d'obtenir facilement des nouvelles informations sur les propriétés des automates et les algorithmes qui s'y appliquent.

Perspectives :

- Étudier, à l'aide de REGAL, d'autres algorithmes sur les automates.
- De nouveaux générateurs pour REGAL.