Random generation of deterministic automata

Frédérique Bassino

Institut Gaspard-Monge
Université de Marne-la-Vallée

Joint work with Cyril Nicaud

- Finite automata
- Uniform random generation
  - Bijections to transform deterministic automata into set partitions
  - Boltzmann samplers to generate set partitions
  - Complexity
- Experimental results and open problems

- Finite automata : models of decision algorithms that require a finite memory.
- Examples :
  - To test whether a binary number is a multiple of 3 or not.
  - But to test whether a word can be decomposed as 1^n0^n requires to remember the numbers of 0's and 1's already red.
- In practice
  - Pattern matching
  - Lexical analysis of a text

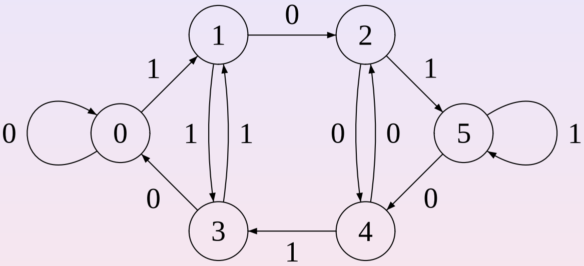
Finite automata

A finite automaton $\mathcal{A}$ is

- a directed finite graph
- whose edges are labelled on a finite alphabet
- with a set I of initial states (or vertices)
- and a set F of final states

- The language recognized by a finite automaton is the set of the labels of the paths from any initial state to any final state.
- Regular languages are the languages recognized by a finite automaton (the sets of words that label the successfull paths in a finite automaton).

Example



An automaton for the binary expansions of the multiples of 6.

- The state 0 is the initial and final state.
- Expansions are red most signicant digit first.

Regular languages and minimal automata

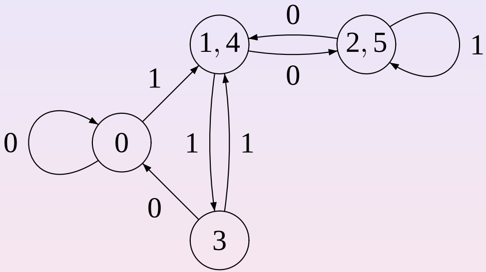
To each regular language, one can associate in a unique way its minimal automaton.

An automaton is deterministic and complete

- if it has only one initial state
- and if for any state q and for any letter ℓ, there exists exactly one an edge labelled ℓ starting from q.

The minimal automaton of a regular language is the complete and deterministic automaton with the minimal number of states that recognizes this language.

The minimal automaton of the multiples of 6



Minimal automaton of the binary expansions of the multiples of 6.

- The state 0 is the initial and final state.
- Expansions are red most significant digit first.

Problem

Enumeration and random generation of regular languages counted by the size of their minimal automaton.

Goal

To analyze the average space complexity of algorithms handling regular languages, the space complexity of a regular language being the number of states of its minimal automaton.

For example, estimate the average size of the intersection of two regular languages.

Accessible complete and deterministic automata

Problem

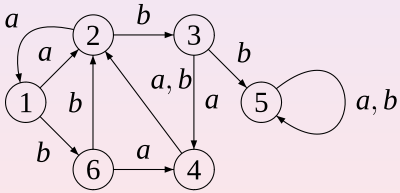
Uniform random generation of accessible complete and deterministic automata with n states (on a finite alphabet).

- An automaton is accessible (or initially connected) if any state can be reached from an initial state.
- Experimentally,
  - 85% of accessible automata on a 2-letter alphabet are minimal,
  - this proportion grows fast with the size of the alphabet.
- *Conjecture* : Asymptotically a constant proportion of accessible complete and deterministic automata are minimal.

From automata to transition structures

An accessible complete and deterministic automaton is transformed into a transition structure by

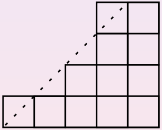
- not taking into account the final states
- labelling the states using a depth first algorithm with respect to the lexicographical order.



A complete and deterministic transition structure corresponds to 2^n (choice of final states) non-isomorphic automata with n states.

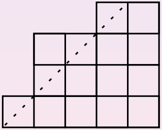
k-Dyck boxed diagrams

- A diagram of width m and height n is a sequence $(x_1, \ldots, x_m)$ of weakly increasing nonnegative integers such that $x_m = n$.
- A k-Dyck diagram of size n is a diagram of width $(k-1)n + 1$ and height n such that $x_i \geq \lceil i/(k-1) \rceil$ for each $i \leq (k-1)n$.



(1,1,2,4,4)

Diagram of width 5 and height 4

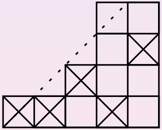


(1,3,3,4,4)

2-Dyck diagram of size 4

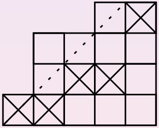
k-Dyck boxed diagrams

- A boxed diagram is a pair of sequences $((x_1, \dots, x_m), (y_1, \dots, y_m))$ where $(x_1, \dots, x_m)$ is a diagram and for each $i \in [\![1..m]\!]$, the y_ith box of the column i of the diagram is marked.
- A diagram gives rise to $\prod_{i=1}^{m} x_i$ boxed diagrams.



(1,1,2,4,4)
(1,1,2,1,3)

A boxed diagram



(1,3,3,4,4)
(1,1,2,2,4)

A 2-Dyck boxed diagram

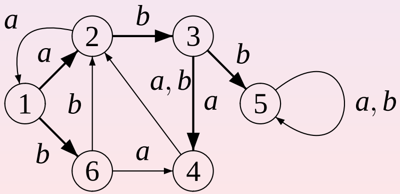
Transition structures and k-Dyck boxed diagrams

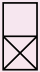
Theorem

The set of accessible, complete and deterministic transition structures of size n on a k-letter alphabet is in bijection with the set $\mathcal{D}_n$ of k-Dyck boxed diagrams of size n.

From transition structures to k-Dyck boxed diagrams

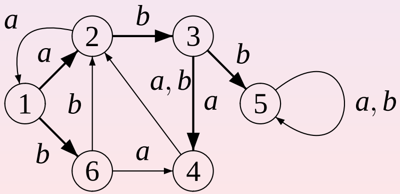
- Build from the initial state a spanning tree using a depth first algorithm with respect to the lexicographical order,
- Encode each transition which is not in the tree as a column
  - whose height is equal to the number of states of the automaton that are already in the tree
  - whose marked box corresponds to the state in which arrives this transition.

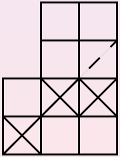




From transition structures to k-Dyck boxed diagrams

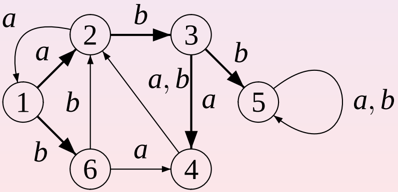
- Build from the initial state a spanning tree using a depth first algorithm with respect to the lexicographical order,
- Encode each transition which is not in the tree as a column
    - whose height is equal to the number of states of the automaton that are already in the tree
    - whose marked box corresponds to the state in which arrives this transition.

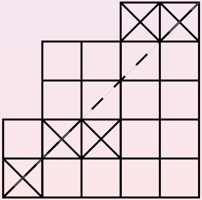




From transition structures to k-Dyck boxed diagrams

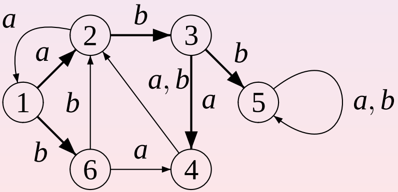
- Build from the initial state a spanning tree using a depth first algorithm with respect to the lexicographical order,
- Encode each transition which is not in the tree as a column
  - whose height is equal to the number of states of the automaton that are already in the tree
  - whose marked box corresponds to the state in which arrives this transition.

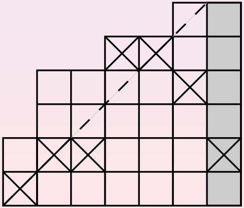




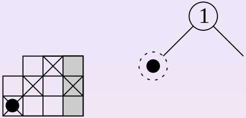
From transition structures to k-Dyck boxed diagrams

- Build from the initial state a spanning tree using a depth first algorithm with respect to the lexicographical order,
- Encode each transition which is not in the tree as a column
  - whose height is equal to the number of states of the automaton that are already in the tree
  - whose marked box corresponds to the state in which arrives this transition.



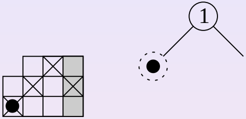


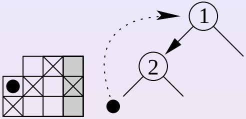
From k-Dyck boxed diagrams to transition structure



Create the initial state
$cpt < x_1$, create a state

From k-Dyck boxed diagrams to transition structure



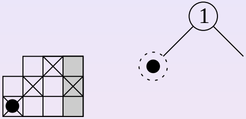


Create the initial state

$cpt < x_1$, create a state

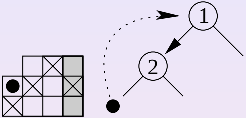
$cpt = x_1$, create an edge

From k-Dyck boxed diagrams to transition structure

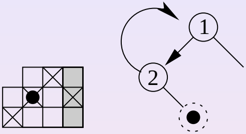


Create the initial state

$cpt < x_1$, create a state

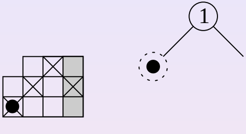


$cpt = x_1$, create an edge



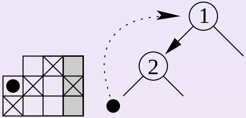
$cpt < x_2$, create a state

From k-Dyck boxed diagrams to transition structure

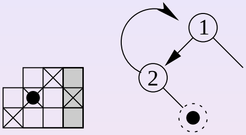


Create the initial state

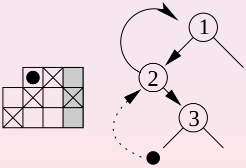
$cpt < x_1$, create a state



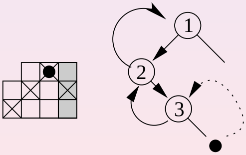
$cpt = x_1$, create an edge



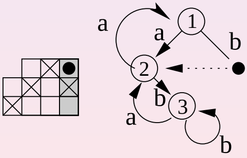
$cpt < x_2$, create a state



$cpt = x_2$, create an edge



$cpt = x_3$, create an edge



$cpt = x_4$, create an edge

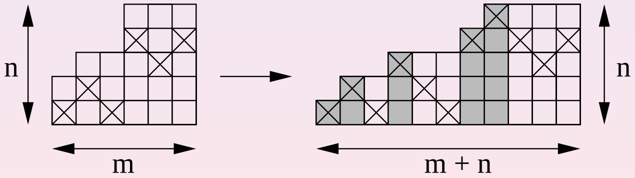
Theorem

The set of boxed diagrams of width m and height n is in bijection with the set of set partitions of $n + m$ elements into n non-empty subsets.

Theorem

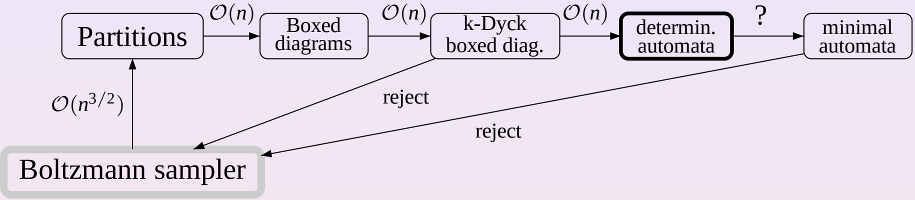
The set of boxed diagrams of width m and height n is in bijection with the set of set partitions of $n + m$ elements into n non-empty subsets.

- Add n boxed columns $(c_i)_{1 \leq i \leq n}$ of height i at the left most position that satisfies the weakly increasing condition
- Mark their highest box



From a boxed diagram to the set partition
$\{\{1,3,6\},\{2,5\},\{4,10\},\{7,9,11\},\{8\}\}$

Random generation



Theorem (Bassino, Nicaud 2007)

The average time complexity of the uniform generation of complete deterministic and accessible automaton with n states using a Boltzmann sampler is $\mathcal{O}(n^{3/2})$.

Boxed diagrams and k-Dyck boxed diagrams

Theorem (Korshunov 1978)

The number of accessible complete and deterministic automata with n states on a k-letter alphabet is asymptotically equals to

$$C_k \, n \, 2^n \left\{ {kn \atop n} \right\} \quad \textit{where} \quad \frac{1}{2} < C_k < 1.$$

Corollary

The probability for a boxed diagram of width $(k-1)n$ and height n to satisfy the k-Dyck condition is asymptotically equal to C_k.
The average number of rejects is $1/C_k$ (less than 2).

Boltzmann samplers

(Duchon, Flajolet, Louchard and Schaeffer 2004)

- A Boltzmann sampler generates objects with a probabilty distribution $\mathbb{P}_x(\gamma) = C\frac{x^{|\gamma|}}{|\gamma|!}$ or $(Cx^{|\gamma|})$
- The generated objects do not have a fixed size, but two objects of the same size have the same probability to be generated.
- The parameter x is chosen depending upon the average size required.
- A rejection algorithm can be used to generate objects of fixed size.
- Almost no precalculus, small memory space used.

Boltzmann samplers

Goal

To uniformly generate at random set partitions of a set with kn elements into n nonempty subsets.

- Partition into n non-empty subsets = set of n non-empty sets
- Exponential generating function counting non-empty sets according to their cardinality : $N(z) = e^z - 1$.
- In the Boltzmann model, the size of each of the n subsets follows a Poisson law of parameter x : $\mathbb{P}_x(|\gamma| = s) = \frac{1}{(e^x - 1)} \frac{x^s}{s!}$.
- The average size of the generated partition is

$$\mathbb{E}_x(\text{size of partitions}) = n\, x \frac{e^x}{e^x - 1}.$$

- $\mathbb{E}_x(\text{size of partitions}) = kn$ for $x = \zeta_k$. (saddle point of $\left\{ {kn \atop n} \right\}$)

Boltzmann samplers

- Generate the size of each of the n subsets following a Poisson law of parameter $x = \zeta_k$ (linear complexity).
- The probability for the generated partition to be of size exactly kn is asymptotically $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$.
  The average number of rejects is $\mathcal{O}(\sqrt{n})$.
- Draw a random partition of $\{1, \ldots, kn\}$ to label the struture (linear complexity)

The average time complexity is $\mathcal{O}(n^{3/2})$

Minimal automata

Proportion of minimal automata

Taille	100	500	1 000	2 000	5 000
minimaux	85.06 %	85.32 %	85.09 %	85.42 %	85.32 %

- Tests made with the C++ library REGAL.
- Tests made with 20 000 automata of each size and a binary alphabet.
- The proportion of minimal automata grows with the cardinality of the alphabet.
- Random generation algorithm for minimal automata using a rejection algorithm.

Open problem

Counting minimal automata

Average time complexity of minimization algorithms

The worst-case complexity of Hopcroft's algorithm is $\Theta(n \log n)$ and the one of Moore's algoritm is $\Theta(n^2)$. But what are their average time complexities ?

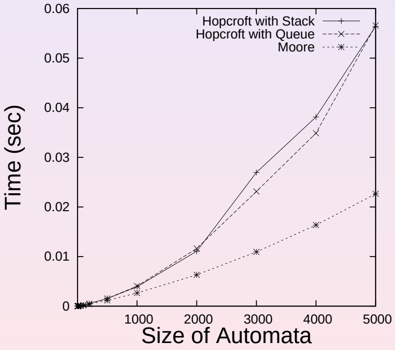


FIG.: Average time complexity of Moore's and Hopcroft's algorithms

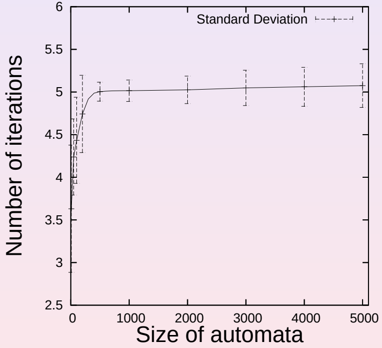


FIG.: Number of iterations in the main loop of Moore's algorithm