Markov Chains and Computer Science

A not so Short Introduction

Jean-Marc Vincent

LIG laboratory, Team Inria-Polaris
University Grenoble-Alpes
Jean-Marc.Vincent@univ-grenoble-alpes.fr

November 2024

Outline

1. **Markov Chain**
   - History
   - Approaches
2. Formalisation
3. Long run behavior
4. Cache modeling
5. Synthesis

History (Andreï Markov)

This study investigates a text excerpt containing 20,000 Russian letters of the alphabet, excluding Ь and Ъ,[2] from Pushkin's novel *Eugene Onegin* – the entire first chapter and sixteen stanzas of the second.

This sequence provides us with 20,000 connected trials, which are either a vowel or a consonant.

Accordingly, we assume the existence of an unknown constant probability p that the observed letter is a vowel. We determine the approximate value of p by observation, by counting all the vowels and consonants. Apart from p, we shall find – also through observation – the approximate values of two numbers p_1 and p_0, and four numbers $p_{1,1}$, $p_{1,0}$, $p_{0,1}$, and $p_{0,0}$. They represent the following probabilities: p_1 – a vowel follows another vowel; p_0 – a vowel follows a consonant; $p_{1,1}$ – a vowel follows two vowels; $p_{1,0}$ – a vowel follows a consonant that is preceded by a vowel; $p_{0,1}$ – a vowel follows a vowel that is preceded by a consonant; and, finally, $p_{0,0}$ – a vowel follows two consonants.

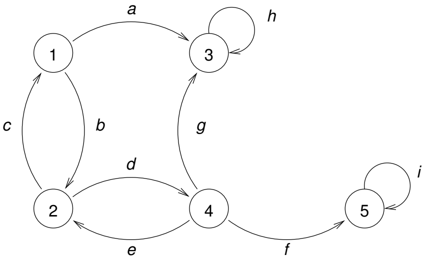
The indices follow the same system that I introduced in my paper "On a Case of Samples Connected in Complex Chain" [Markov 1911b]; with reference to my other paper, "Investigation of a Remarkable Case of Dependent Samples" [Markov 1907a], however, $p_0 = p_2$. We denote the opposite probabilities for consonants with q and indices that follow the same pattern.

If we seek the value of p, we first find 200 approximate values from which we can determine the arithmetic mean. To be precise, we divide the entire sequence of 20,000 letters into 200 separate sequences of 100 letters, and count how many vowels there are in each 100: we obtain 200 numbers, which, when divided by 100, yield 200 approximate values of p.

An example of statistical investigation in the text of "Eugene Onegin" illustrating coupling of "tests" in chains.
(1913) In Proceedings of Academic Scientific St. Petersburg, VI, pages 153-162.

1856-1922

Graphs and Paths



Random Walks

Path in a graph:
X_n n-th visited node
path : $i_0, i_1, \cdots, i_n$
normalized weight : arc $(i,j) \longrightarrow p_{i,j}$

concatenation : . $\longrightarrow \times$
$\mathcal{P}(i_0, i_1, \cdots, i_n) = p_{i_0,i_1} p_{i_1,i_2} \cdots p_{i_{n-1},i_n}$

disjoint union : $\cup \longrightarrow +$
$\mathcal{P}(i_0 \rightsquigarrow i_n) = \sum_{i_1,\cdots,i_{n-1}} p_{i_0,i_1} p_{i_1,i_2} \cdots p_{i_{n-1},i_n}$

automaton : state/transitions randomized (language)

Dynamical Systems

Figure 3. A fern drawn by a Markov chain

Diaconis-Freedman 99

Evolution Operator

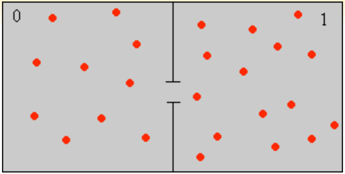
Initial value : X_0
Recurrence equation : $X_{n+1} = \Phi(X_n, \xi_{n+1})$

Innovation at step $n+1$: ξ_{n+1}
Finite set of innovations : $\{\phi_1, \phi_2, \cdots, \phi_K\}$

Random function (chosen with a given probability)

Randomized Iterated Systems

Measure Approach



Ehrenfest's Urn (1907)

Paul Ehrenfest (1880-1933)

Distribution of K particles

Initial State $X_0 = 0$
State = nb of particles in 0
Dynamic : uniform choice of a particle and jump to the other side

$$
\begin{aligned}
\pi_n(i) &= \mathbb{P}(X_n = i | X_0 = 0) \\
&= \pi_{n-1}(i-1).\frac{K-i+1}{K} \\
&\quad + \pi_{n-1}(i+1).\frac{i+1}{K}
\end{aligned}
$$

$$\pi_n = \pi_{n-1}.P$$

Iterated product of matrices

Algorithmic Interpretation

```
int minimum (T,K)
min= +∞
cpt=0;
for (k=0; k < K; k++) do
  if (T[i]< min) then
    min = T[k];
    process(min);
    cpt++;
  end if
end for
return(cpt)
```

Worst case K;
Best case 1;
on average ?

Number of processing min

State : $X_n =$ rank of the n^{th} processing

$$\mathbb{P}(X_{n+1} = j | X_n = i, X_{n-1} = i_{k-1}, \cdots, X_0 = i_0) = \mathbb{P}(X_{n+1} = j | X_n = i)$$

$$\mathbb{P}(X_{n+1} = j | X_n = i) = \begin{cases} \frac{1}{K-i+1} & \text{si } j < i; \\ 0 & \text{sinon.} \end{cases}$$

All the information of for the step $n+1$ is contained in the state at step n

$$\tau = \min\{n; \quad X_n = 1\}$$

Correlation of length 1

Outline

Formal definition

Let $\{X_n\}_{n\in\mathbb{N}}$ a random sequence of variables in a discrete state-space $\mathcal{X}$

$\{X_n\}_{n\in\mathbb{N}}$ is a **Markov chain** with initial law $\pi(0)$ iff

- $X_0 \sim \pi(0)$ and
- for all $n \in \mathbb{N}$ and for all $(j, i, i_{n-1}, \cdots, i_0) \in \mathcal{X}^{n+2}$

$$\mathbb{P}(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \cdots, X_0 = i_0) = \mathbb{P}(X_{n+1} = j | X_n = i).$$

$\{X_n\}_{n\in\mathbb{N}}$ is a **homogeneous** Markov chain iff

- for all $n \in \mathbb{N}$ and for all $(j, i) \in \mathcal{X}^2$

$$\mathbb{P}(X_{n+1} = j | X_n = i) = \mathbb{P}(X_1 = j | X_0 = i) \stackrel{def}{=} p_{i,j}.$$

(invariance during time of probability transition)

Formal definition

Let $\{X_n\}_{n\in\mathbb{N}}$ a random sequence of variables in a discrete state-space $\mathcal{X}$

$\{X_n\}_{n\in\mathbb{N}}$ is a **Markov chain** with initial law $\pi(0)$ iff

- $X_0 \sim \pi(0)$ and
- for all $n \in \mathbb{N}$ and for all $(j, i, i_{n-1}, \cdots, i_0) \in \mathcal{X}^{n+2}$

$$\mathbb{P}(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \cdots, X_0 = i_0) = \mathbb{P}(X_{n+1} = j | X_n = i).$$

$\{X_n\}_{n\in\mathbb{N}}$ is a **homogeneous** Markov chain iff

- for all $n \in \mathbb{N}$ and for all $(j, i) \in \mathcal{X}^2$

$$\mathbb{P}(X_{n+1} = j | X_n = i) = \mathbb{P}(X_1 = j | X_0 = i) \stackrel{def}{=} p_{i,j}.$$

(invariance during time of probability transition)

Algebraic representation

$P = ((p_{i,j}))$ is the **transition matrix** of the chain

- P is a **stochastic matrix**

$$p_{i,j} \geqslant 0; \quad \sum_j p_{i,j} = 1.$$

Linear recurrence equation $\pi_i(n) = \mathbb{P}(X_n = i)$

$$\pi_n = \pi_{n-1} P.$$

- Equation of **Chapman-Kolmogorov** (homogeneous): $P^n = ((p_{i,j}^{(n)}))$

$$p_{i,j}^{(n)} = \mathbb{P}(X_n = j | X_0 = i); \quad P^{n+m} = P^n . P^m;$$

$$\begin{aligned}
\mathbb{P}(X_{n+m} = j | X_0 = i) &= \sum_k \mathbb{P}(X_{n+m} = j | X_m = k)\mathbb{P}(X_m = k | X_0 = i); \\
&= \sum_k \mathbb{P}(X_n = j | X_0 = k)\mathbb{P}(X_m = k | X_0 = i).
\end{aligned}$$

Interpretation: decomposition of the set of paths with length $n + m$ from i to j.

Problems

Finite horizon

- Estimation of $\pi(n)$
- Estimation of stopping times

$$\tau_A = \inf\{n \geqslant 0;\ X_n \in A\}$$

- ...

Infinite horizon

- Convergence properties
- Estimation of the asymptotics
- Estimation speed of convergence
- ...

Problems

Finite horizon

- Estimation of $\pi(n)$
- Estimation of stopping times

$$\tau_A = \inf\{n \geqslant 0;\ X_n \in A\}$$

- ...

Infinite horizon

- Convergence properties
- Estimation of the asymptotics
- Estimation speed of convergence
- ...

Applications in computer science

Applications in most of scientific domains ...
In computer science :

Markov chain : an algorithmic tool

- Numerical methods (Monte-Carlo methods)
- Randomized algorithms (ex: TCP, searching, pageRank...)
- Learning machines (hidden Markov chains)
-. . .

Markov chains : a modeling tool

- Performance evaluation (quantification and dimensionning)
- Stochastic control
- Program verification
-. . .

Applications in computer science

Applications in most of scientific domains ...
In computer science :

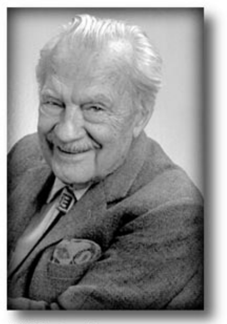
Markov chain : an algorithmic tool

- Numerical methods (Monte-Carlo methods)
- Randomized algorithms (ex: TCP, searching, pageRank...)
- Learning machines (hidden Markov chains)
-. . .

Markov chains : a modeling tool

- Performance evaluation (quantification and dimensionning)
- Stochastic control
- Program verification
-. . .

Nicholas Metropolis (1915-1999)



Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

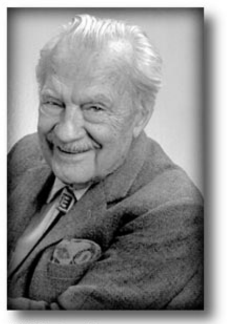
Simulated annealing

Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$

$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Nicholas Metropolis (1915-1999)



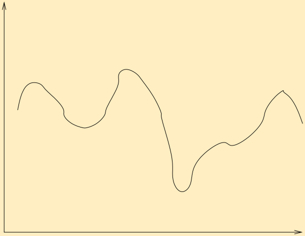
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

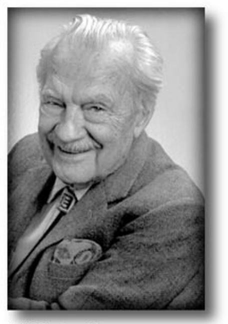
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Nicholas Metropolis (1915-1999)



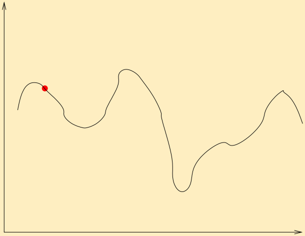
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

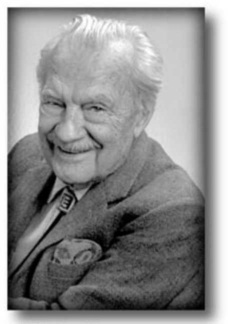
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Nicholas Metropolis (1915-1999)



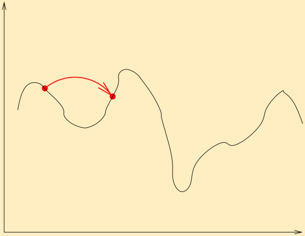
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

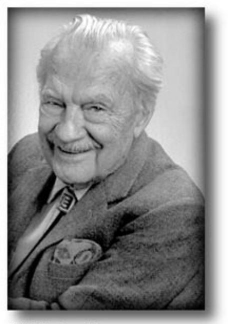
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Nicholas Metropolis (1915-1999)



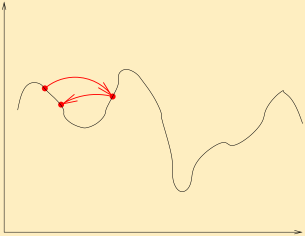
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

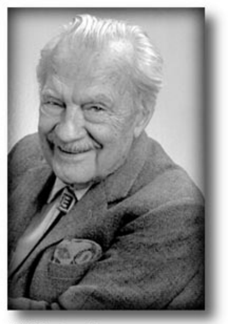
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Nicholas Metropolis (1915-1999)



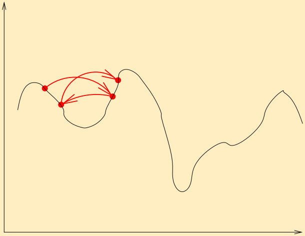
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

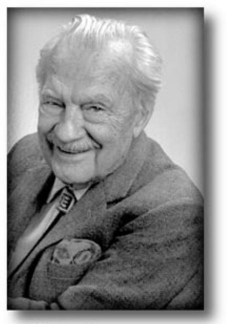
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Nicholas Metropolis (1915-1999)



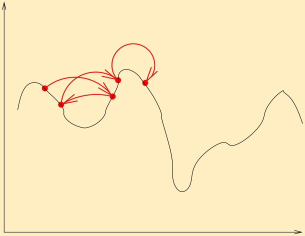
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

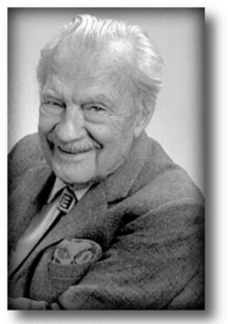
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \overset{n\to\infty}{\longrightarrow} 0.$$

Nicholas Metropolis (1915-1999)



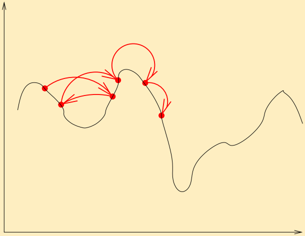
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

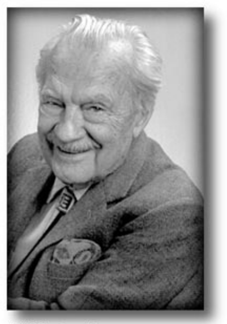
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \overset{n\to\infty}{\longrightarrow} 0.$$

Nicholas Metropolis (1915-1999)



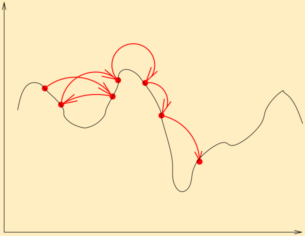
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

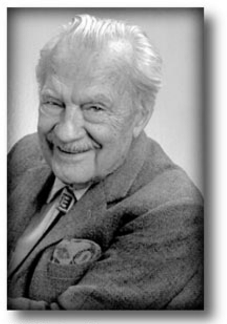
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Nicholas Metropolis (1915-1999)



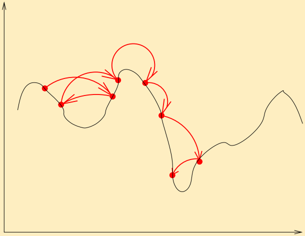
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

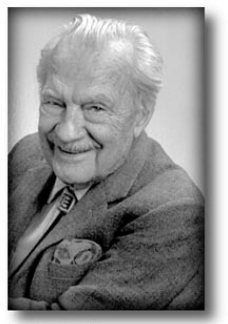
Convergence to a global minimum by a stochastic gradient scheme.

$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Nicholas Metropolis (1915-1999)



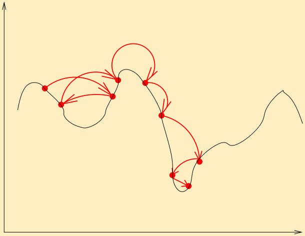
Nick Metropolis

Metropolis contributed several original ideas to mathematics and physics. Perhaps the most widely known is the Monte Carlo method. Also, in 1953 Metropolis co-authored the first paper on a technique that was central to the method known now as simulated annealing. He also developed an algorithm (the Metropolis algorithm or Metropolis-Hastings algorithm) for generating samples from the Boltzmann distribution, later generalized by W.K. Hastings.

Simulated annealing

Convergence to a global minimum by a stochastic gradient scheme.

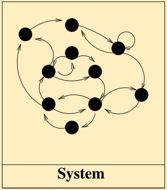
$$X_{n+1} = X_n - \vec{grad}\Phi(X_n)\Delta_n(Random).$$



$$\Delta_n(random) \xrightarrow{n\to\infty} 0.$$

Modeling and Analysis of Computer Systems

Complex system



Basic model assumptions

System :
- automaton (discrete state space)
- **discrete** or continuous time

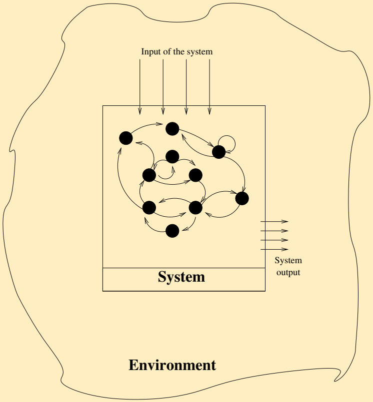
Environment : non deterministic
- time homogeneous
- stochastically regular

Problem

Understand "typical" states
- steady-state estimation
- ergodic simulation
- state space exploring techniques

Modeling and Analysis of Computer Systems

Complex system



Basic model assumptions

System :
- automaton (discrete state space)
- **discrete** or continuous time

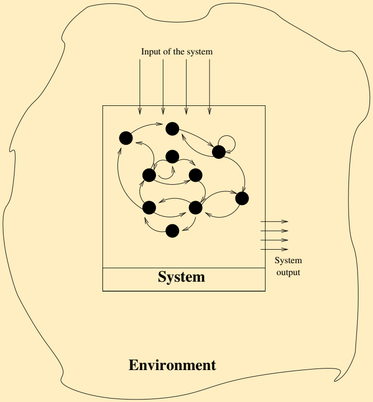
Environment : non deterministic
- time homogeneous
- stochastically regular

Problem

Understand "typical" states
- steady-state estimation
- ergodic simulation
- state space exploring techniques

Modeling and Analysis of Computer Systems

Complex system



Basic model assumptions

System :
- automaton (discrete state space)
- **discrete** or continuous time

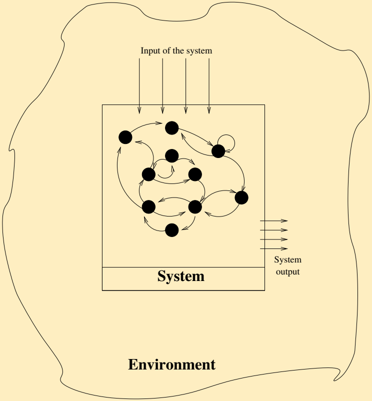
Environment : non deterministic
- time homogeneous
- stochastically regular

Problem

Understand "typical" states
- steady-state estimation
- ergodic simulation
- state space exploring techniques

Modeling and Analysis of Computer Systems

Complex system



Basic model assumptions

System :
- automaton (discrete state space)
- **discrete** or continuous time

Environment : non deterministic
- time homogeneous
- stochastically regular

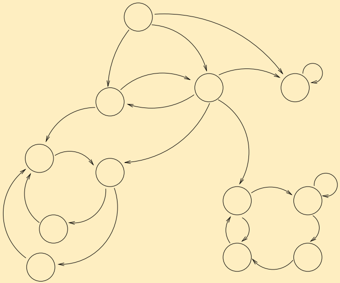
Problem

Understand "typical" states
- steady-state estimation
- ergodic simulation
- state space exploring techniques

Outline

States classification

Graph analysis



Irreducible class

Strongly connected components

i and j are in the same component if there exist a path from i to j and a path from j to i with a positive probability

Leaves of the tree of strongly connected components are **irreducible** classes

States in irreducible classes are called **recurrent**

Other states are called **transient**

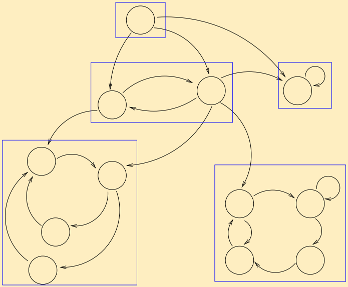
Periodicity

An irreducible class is **aperiodic** if the gcd of length of all cycles is 1

A Markov chain is **irreducible** if there is only one class.
Each state is reachable from any other state with a positive probability path.

States classification

Graph analysis



Irreducible class

Strongly connected components
i and *j* are in the same component if there exist a path from *i* to *j* and a path from *j* to *i* with a positive probability
Leaves of the tree of strongly connected components are **irreducible** classes
States in irreducible classes are called **recurrent**

Other states are called **transient**

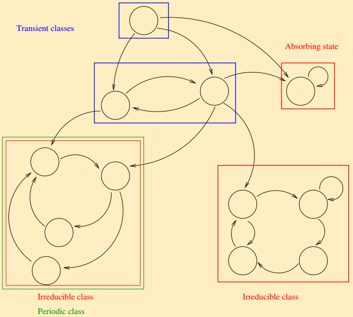
Periodicity

An irreducible class is **aperiodic** if the gcd of length of all cycles is 1

A Markov chain is **irreducible** if there is only one class.
Each state is reachable from any other state with a positive probability path.

States classification

Graph analysis



Irreducible class

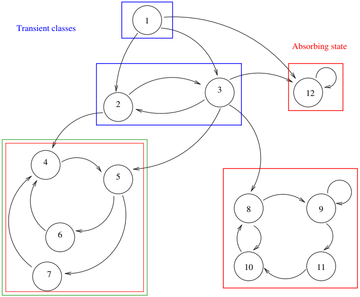
Strongly connected components
i and j are in the same component if there exist a path from i to j and a path from j to i with a positive probability
Leaves of the tree of strongly connected components are **irreducible** classes
States in irreducible classes are called **recurrent**
Other states are called **transient**

Periodicity

An irreducible class is **aperiodic** if the gcd of length of all cycles is 1

A Markov chain is **irreducible** if there is only one class.
Each state is reachable from any other state with a positive probability path.

States classification : matrix form

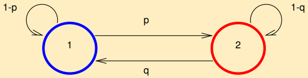


	1	2	3	4	5	6	7	8	9	10	11	12
1		*	*									*
2			*	*								
3					*			*				*
4					*							
5						*	*					
6				*								
7				*								
8									*	*		
9									*		*	
10								*				
11										*		
12												1

Automaton Flip-flop

ON-OFF system

Two states model :
- communication line
- processor activity
- ...



Parameters :
- proportion of transitions : p, q
- mean sojourn time in state 1 : $\frac{1}{p}$
- mean sojourn time in state 2 : $\frac{1}{q}$

Trajectory

X_n state of the automaton at time n.

Transient distribution

$$\pi_n(1) = \mathbb{P}(X_n = 1);$$

$$\pi_n(2) = \mathbb{P}(X_n = 2)$$

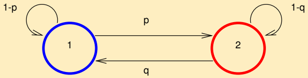
Problem

Estimation of π_n : state prevision, resource utilization

Automaton Flip-flop

ON-OFF system

Two states model :
- communication line
- processor activity
- ...



Parameters :
- proportion of transitions : p, q
- mean sojourn time in state 1 : $\frac{1}{p}$
- mean sojourn time in state 2 : $\frac{1}{q}$

Trajectory

X_n state of the automaton at time n.

Transient distribution

$$\pi_n(1) = \mathbb{P}(X_n = 1);$$

$$\pi_n(2) = \mathbb{P}(X_n = 2)$$

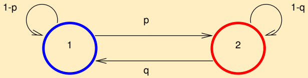
Problem

Estimation of π_n : state prevision, resource utilization

Automaton Flip-flop

ON-OFF system

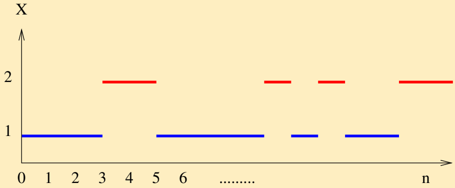
Two states model :
- communication line
- processor activity
- ...



Parameters :
- proportion of transitions : p, q
- mean sojourn time in state 1 : $\frac{1}{p}$
- mean sojourn time in state 2 : $\frac{1}{q}$

Trajectory

X_n state of the automaton at time n.



Transient distribution

$$\pi_n(1) = \mathbb{P}(X_n = 1);$$

$$\pi_n(2) = \mathbb{P}(X_n = 2)$$

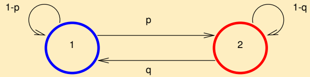
Problem

Estimation of π_n : state prevision, resource utilization

Automaton Flip-flop

ON-OFF system

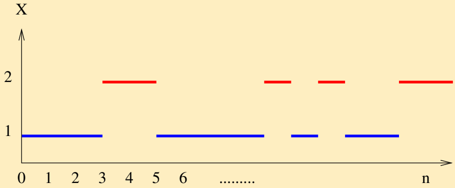
Two states model :
- communication line
- processor activity
- ...



Parameters :
- proportion of transitions : p, q
- mean sojourn time in state 1 : $\frac{1}{p}$
- mean sojourn time in state 2 : $\frac{1}{q}$

Trajectory

X_n state of the automaton at time n.



Transient distribution

$$\pi_n(1) = \mathbb{P}(X_n = 1);$$

$$\pi_n(2) = \mathbb{P}(X_n = 2)$$

Problem

Estimation of π_n : state prevision, resource utilization

Mathematical model

Transition probabilities

$$P = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix}$$

$$\begin{aligned}
\mathbb{P}(X_{n+1} = 1 | X_n = 1) &= 1-p; \\
\mathbb{P}(X_{n+1} = 2 | X_n = 1) &= p; \\
\mathbb{P}(X_{n+1} = 1 | X_n = 2) &= q; \\
\mathbb{P}(X_{n+1} = 2 | X_n = 2) &= 1-q.
\end{aligned}$$

$$\begin{cases} \pi_{n+1}(1) = \pi_n(1)(1-p) + \pi_n(2)q; \\ \pi_{n+1}(2) = \pi_n(1)p + \pi_n(2)(1-q); \end{cases}$$

$$\pi_{n+1} = \pi_n P$$

Linear iterations

Spectrum of P (eigenvalues)

$Sp = \{1, 1-p-q\}$

System resolution

$|1-p-q| < 1$ Non pathologic case

$$\begin{cases} \pi_n(1) = \frac{q}{p+q} + \left(\pi_0(1) - \frac{q}{p+q}\right)(1-p-q)^n; \\ \pi_n(2) = \frac{p}{p+q} + \left(\pi_0(2) - \frac{p}{p+q}\right)(1-p-q)^n; \end{cases}$$

$1-p-q = 1$ $p = q = 0$ Reducible behavior

$1-p-q = -1$ $p = q = 1$ Periodic behavior

Mathematical model

Transition probabilities

$$P = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix}$$

$$\begin{aligned}
\mathbb{P}(X_{n+1} = 1 | X_n = 1) &= 1-p;\\
\mathbb{P}(X_{n+1} = 2 | X_n = 1) &= p;\\
\mathbb{P}(X_{n+1} = 1 | X_n = 2) &= q;\\
\mathbb{P}(X_{n+1} = 2 | X_n = 2) &= 1-q.
\end{aligned}$$

$$\begin{cases} \pi_{n+1}(1) = \pi_n(1)(1-p) + \pi_n(2)q;\\ \pi_{n+1}(2) = \pi_n(1)p + \pi_n(2)(1-q); \end{cases}$$

$$\pi_{n+1} = \pi_n P$$

Linear iterations

Spectrum of P (eigenvalues)

$\mathcal{S}p = \{1, 1-p-q\}$

System resolution

$|1-p-q| < 1$ Non pathologic case

$$\begin{cases} \pi_n(1) = \frac{q}{p+q} + \left(\pi_0(1) - \frac{q}{p+q}\right)(1-p-q)^n;\\ \pi_n(2) = \frac{p}{p+q} + \left(\pi_0(2) - \frac{p}{p+q}\right)(1-p-q)^n; \end{cases}$$

$1-p-q = 1$ $p = q = 0$ Reducible behavior

$1-p-q = -1$ $p = q = 1$ Periodic behavior

Mathematical model

Transition probabilities

$$P = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix}$$

$$\begin{aligned} \mathbb{P}(X_{n+1} = 1 | X_n = 1) &= 1-p; \\ \mathbb{P}(X_{n+1} = 2 | X_n = 1) &= p; \\ \mathbb{P}(X_{n+1} = 1 | X_n = 2) &= q; \\ \mathbb{P}(X_{n+1} = 2 | X_n = 2) &= 1-q. \end{aligned}$$

$$\begin{cases} \pi_{n+1}(1) = \pi_n(1)(1-p) + \pi_n(2)q; \\ \pi_{n+1}(2) = \pi_n(1)p + \pi_n(2)(1-q); \end{cases}$$

$$\pi_{n+1} = \pi_n P$$

Linear iterations

Spectrum of P (eigenvalues)

$\mathcal{S}p = \{1, 1-p-q\}$

System resolution

$|1-p-q| < 1$ Non pathologic case

$$\begin{cases} \pi_n(1) = \frac{q}{p+q} + \left(\pi_0(1) - \frac{q}{p+q}\right)(1-p-q)^n; \\ \pi_n(2) = \frac{p}{p+q} + \left(\pi_0(2) - \frac{p}{p+q}\right)(1-p-q)^n; \end{cases}$$

$1-p-q = 1$ $p = q = 0$ Reducible behavior

$1-p-q = -1$ $p = q = 1$ Periodic behavior

Mathematical model

Transition probabilities

$$P = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix}$$

$$\begin{aligned}
\mathbb{P}(X_{n+1} = 1 | X_n = 1) &= 1-p; \\
\mathbb{P}(X_{n+1} = 2 | X_n = 1) &= p; \\
\mathbb{P}(X_{n+1} = 1 | X_n = 2) &= q; \\
\mathbb{P}(X_{n+1} = 2 | X_n = 2) &= 1-q.
\end{aligned}$$

$$\begin{cases} \pi_{n+1}(1) = \pi_n(1)(1-p) + \pi_n(2)q; \\ \pi_{n+1}(2) = \pi_n(1)p + \pi_n(2)(1-q); \end{cases}$$

$$\pi_{n+1} = \pi_n P$$

Linear iterations

Spectrum of P (eigenvalues)

$\mathcal{S}p = \{1, 1-p-q\}$

System resolution

$|1-p-q| < 1$ Non pathologic case

$$\begin{cases} \pi_n(1) = \frac{q}{p+q} + \left(\pi_0(1) - \frac{q}{p+q}\right)(1-p-q)^n; \\ \pi_n(2) = \frac{p}{p+q} + \left(\pi_0(2) - \frac{p}{p+q}\right)(1-p-q)^n; \end{cases}$$

$1-p-q = 1$ $p = q = 0$ Reducible behavior

$1-p-q = -1$ $p = q = 1$ Periodic behavior

Mathematical model

Transition probabilities

$$P = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix}$$

$$\begin{aligned}
\mathbb{P}(X_{n+1} = 1 | X_n = 1) &= 1-p;\\
\mathbb{P}(X_{n+1} = 2 | X_n = 1) &= p;\\
\mathbb{P}(X_{n+1} = 1 | X_n = 2) &= q;\\
\mathbb{P}(X_{n+1} = 2 | X_n = 2) &= 1-q.
\end{aligned}$$

$$\left\{ \begin{array}{l} \pi_{n+1}(1) = \pi_n(1)(1-p) + \pi_n(2)q; \\ \pi_{n+1}(2) = \pi_n(1)p + \pi_n(2)(1-q); \end{array} \right.$$

$$\pi_{n+1} = \pi_n P$$

Linear iterations
Spectrum of P (eigenvalues)
$\mathcal{S}p = \{1, 1-p-q\}$

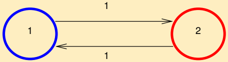
System resolution

$|1-p-q| < 1$ Non pathologic case

$$\left\{ \begin{array}{l} \pi_n(1) = \frac{q}{p+q} + \left(\pi_0(1) - \frac{q}{p+q}\right)(1-p-q)^n; \\ \pi_n(2) = \frac{p}{p+q} + \left(\pi_0(2) - \frac{p}{p+q}\right)(1-p-q)^n; \end{array} \right.$$

$1-p-q = 1$ $p = q = 0$ Reducible behavior

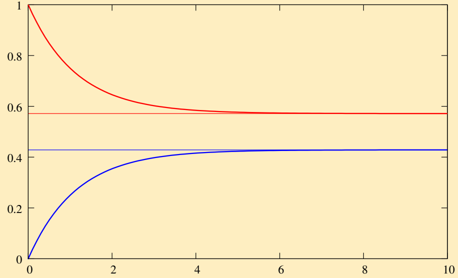
$1-p-q = -1$ $p = q = 1$ Periodic behavior



Recurrent behavior

Numerical example

$p = \frac{1}{4}$, $q = \frac{1}{3}$



Rapid convergence
(exponential rate)

Steady state behavior

$$\begin{cases} \pi_\infty(1) = \frac{q}{p+q}; \\ \pi_\infty(2) = \frac{p}{p+q}. \end{cases}$$

π_∞ unique probability vector solution

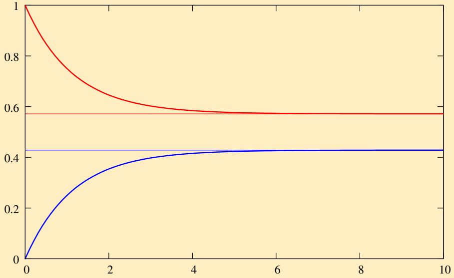
$$\pi_\infty = \pi_\infty P.$$

If $\pi_0 = \pi_\infty$ then $\pi_n = \pi_\infty$ for all n
stationary behavior

Recurrent behavior

Numerical example

$p = \frac{1}{4}$, $q = \frac{1}{3}$



Rapid convergence
(exponential rate)

Steady state behavior

$$\begin{cases} \pi_\infty(1) = \frac{q}{p+q}; \\ \pi_\infty(2) = \frac{p}{p+q}. \end{cases}$$

π_∞ unique probability vector solution

$$\pi_\infty = \pi_\infty P.$$

If $\pi_0 = \pi_\infty$ then $\pi_n = \pi_\infty$ for all n
stationary behavior

Convergence In Law

Let $\{X_n\}_{n\in\mathbb{N}}$ a homogeneous, irreducible and aperiodic Markov chain taking values in a discrete state $\mathcal{X}$ then

- The following limits exist (and do not depend on i)

$$\lim_{n\to+\infty} \mathbb{P}(X_n = j | X_0 = i) = \pi_j;$$

- π is the unique probability vector invariant by P

$$\pi P = \pi;$$

- The convergence is rapid (geometric); there is $C > 0$ and $0 < \alpha < 1$ such that

$$||\mathbb{P}(X_n = j | X_0 = i) - \pi_j|| \leqslant C.\alpha^n.$$

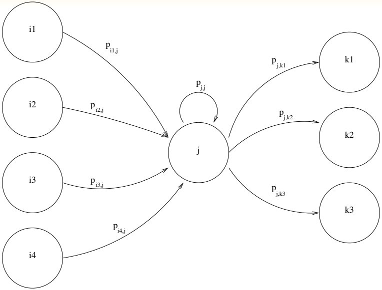
Denote

$$X_n \xrightarrow{\mathcal{L}} X_\infty;$$

with X_∞ with law π
π is the **steady-state probability** associated to the chain

Interpretation

Equilibrium equation



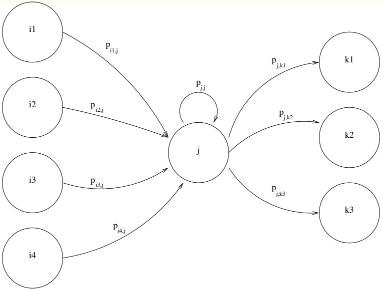
Probability to enter j =probability to exit j
balance equation

$$\sum_{i \neq j} \pi_i p_{i,j} = \sum_{k \neq j} \pi_j p_{j,k} = \pi_j \sum_{k \neq j} p_{j,k} = \pi_j (1 - p_{j,j})$$

$\pi \stackrel{def}{=}$ steady-state.
If $\pi_0 = \pi$ the process is stationary ($\pi_n = \pi$)

Interpretation

Equilibrium equation



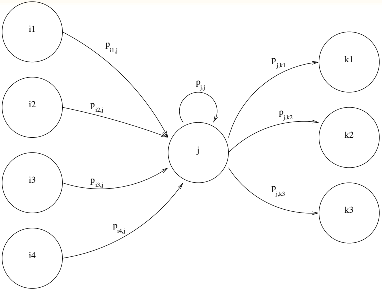
Probability to enter j =probability to exit j
balance equation

$$\sum_{i \neq j} \pi_i p_{i,j} = \sum_{k \neq j} \pi_j p_{j,k} = \pi_j \sum_{k \neq j} p_{j,k} = \pi_j (1 - p_{j,j})$$

$\pi \stackrel{def}{=}$ steady-state.
If $\pi_0 = \pi$ the process is stationary ($\pi_n = \pi$)

Interpretation

Equilibrium equation



Probability to enter j =probability to exit j
balance equation

$$\sum_{i \neq j} \pi_i p_{i,j} = \sum_{k \neq j} \pi_j p_{j,k} = \pi_j \sum_{k \neq j} p_{j,k} = \pi_j (1 - p_{j,j})$$

$\pi \overset{def}{=}$ steady-state.
If $\pi_0 = \pi$ the process is stationary ($\pi_n = \pi$)

Proof 1 : Finite state space algebraic approach

Positive matrix $P > 0$

contraction $\max_i p_{i,j}^{(n)} - \min_i p_{i,j}^{(n)}$

Perron-Froebenius $P > 0$

P is positive and stochastic then the spectral radius $\rho = 1$ is an eigenvalue with multiplicity 1, the corresponding eigenvector is positive and the other eigenvalues have module < 1.

Case $P \geqslant 0$

Aperiodique and irreducible $\Rightarrow$ there is k such that $P^k > 0$ and apply the above result.

Proof 1 : details $P > 0$

Soit x et $y = Px$, $\omega = \min_{i,j} p_{i,j}$

$$\overline{x} = \max_i x_i, \ \underline{x} = \min_i x_i.$$

$$y_i = \sum_j p_{i,j} x_j$$

Property of centroid :

$$(1-\omega)\underline{x} + \omega\overline{x} \leqslant y_i \leqslant (1-\omega)\overline{x} + \omega\underline{x}$$

$$0 \leqslant \overline{y} - \underline{y} \leqslant (1-2\omega)(\overline{x} - \underline{x})$$

$$P^n x \longrightarrow s(x)(1, 1, \cdots, 1)^t$$

Then P^n converges to a matrix where all lines are identical.

Proof 2 : Return time

$$\tau_i^+ = \inf\{n \geqslant 1; \quad X_n = i | X_0 = i\}.$$

then $\frac{1}{\mathbb{E}\tau_i^+}$ is an invariant probability (Kac's lemma)

1914-1984

Proof :

1. $\mathbb{E}\tau_i^+ < \infty$
2. Study on a regeneration interval (Strong Markov property)
3. Uniqueness by harmonic functions

Proof 3 : Coupling

Let $\{X_n\}_{n\in\mathbb{N}}$ a homogeneous aperiodic and irreducible Markov chain with initial law $\pi(0)$ and steady-state probability π.

Let $\left\{\tilde{X}_n\right\}_{n\in\mathbb{N}}$ another Markov chain $\tilde{\pi}(0)$ with the same transition matrix as $\{X_n\}$

$\{X_n\}$ et $\left\{\tilde{X}_n\right\}$ independent

- $Z_n = (X_n, \tilde{X}_n)$ is a homogeneous Markov chain
- if $\{X_n\}$ is aperiodic and irreducible, so it is for Z_n

Let τ be the hitting time of the diagonal, $\tau < \infty$ P-a.s. then

$$|\mathbb{P}(X_n = i) - \mathbb{P}(\tilde{X}_n = i)| < 2\mathbb{P}(\tau > n)$$

$$|\mathbb{P}(X_n = i) - \pi(i)| < 2\mathbb{P}(\tau > n) \longrightarrow 0.$$

Ergodic Theorem

Let $\{X_n\}_{n\in\mathbb{N}}$ a homogeneous aperiodic and irreducible Markov chain on $\mathcal{X}$ with steady-state probability π then
- for all function f satisfying $\mathbb{E}_\pi|f| < +\infty$

$$\frac{1}{N}\sum_{n=1}^{N} f(X_n) \xrightarrow{P-p.s.} \mathbb{E}_\pi f.$$

generalization of the *strong law of large numbers*

- If $\mathbb{E}_\pi f = 0$ then there exist σ such that

$$\frac{1}{\sigma\sqrt{N}}\sum_{n=1}^{N} f(X_n) \xrightarrow{\mathcal{L}} \mathcal{N}(0,1).$$

generalization of the *central limit theorem*

Fundamental question

Given a function f (cost, reward, performance,...) estimate

$$\mathbb{E}_\pi f$$

and give the quality of this estimation.

Solving methods

Solving $\pi = \pi P$

- Analytical/approximation methods
- Formal methods $N \leqslant 50$
  Maple, Sage,...
- Direct numerical methods $N \leqslant 1000$
  Mathematica, Scilab,...
- Iterative methods with preconditioning $N \leqslant 100,000$
  Marca,...
- Adapted methods (structured Markov chains) $N \leqslant 1,000,000$
  PEPS,...
- Monte-Carlo simulation $N \geqslant 10^7$

Postprocessing of the stationary distribution

Computation of rewards (expected stationary functions)
Utilization, response time,...

Ergodic Sampling(1)

Ergodic sampling algorithm

Representation : **transition fonction**

$$X_{n+1} = \Phi(X_n, e_{n+1}).$$

$x \leftarrow x_0$
{choice of the initial state at time =0}
$n = 0$;
repeat
  $n \leftarrow n + 1$;
  $e \leftarrow Random\_event()$;
  $x \leftarrow \Phi(x, e)$;
  Store x
  {computation of the next state X_{n+1}}
until some empirical criteria
return the trajectory

Problem : Stopping criteria

Ergodic Sampling(2)

Start-up

Convergence to stationary behavior

$$\lim_{n\to+\infty} \mathbb{P}(X_n = x) = \pi_x.$$

Warm-up period : Avoid initial state dependence
Estimation error :

$$||\mathbb{P}(X_n = x) - \pi_x|| \leqslant C\lambda_2^n.$$

λ_2 second greatest eigenvalue of the transition matrix
- bounds on C and λ_2 (spectral gap)
- cut-off phenomena

λ_2 and C non reachable in practice
(complexity equivalent to the computation of π)
some known results (Birth and Death processes)

Ergodic Sampling(3)

Estimation quality

Ergodic theorem :

$$\lim_{n\to+\infty} \frac{1}{n} \sum_{i=1}^{n} f(X_i) = \mathbb{E}_\pi f.$$

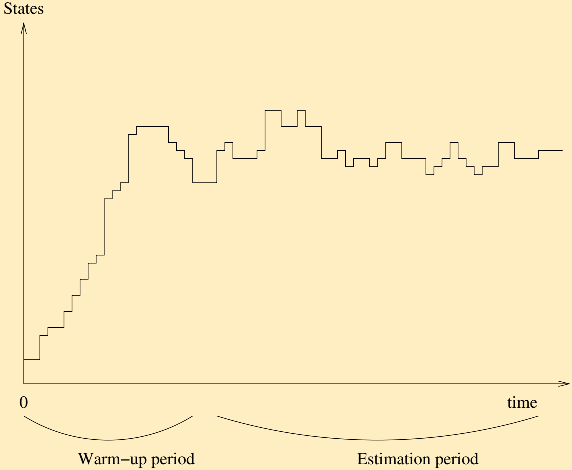
Length of the sampling : Error control (CLT theorem)

Complexity

Complexity of the transition function evaluation (computation of $\Phi(x, .)$)
Related to the stabilization period + Estimation time

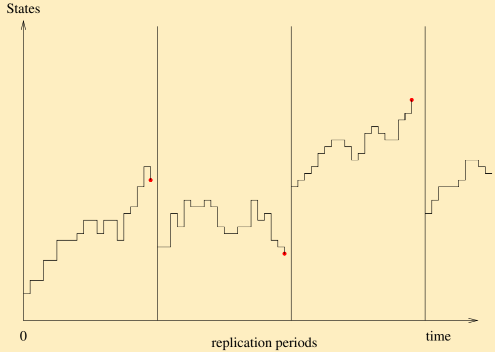
Ergodic sampling(4)

Typical trajectory



Replication Method

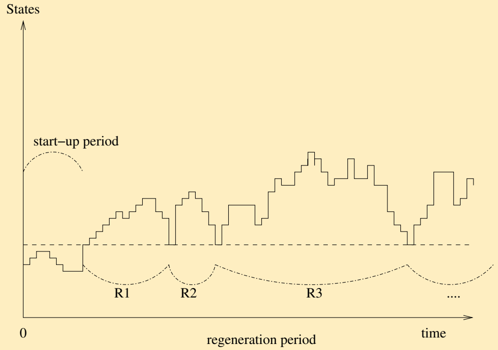
Typical trajectory



Sample of independent states
Drawback : length of the replication period (dependence from initial state)

Regeneration Method

Typical trajectory



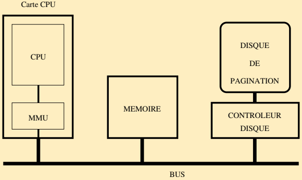
Sample of independent trajectories
Drawback : length of the regeneration period (choice of the regenerative state)

Outline

Cache modelling

Virtual memory

Paging in OS



- cache hierarchy (processor)
- data caches (databases)
- proxy-web (internet)
- routing tables (networking)
- ...

State of the system : Page position

Huge number of pages, small memory capacity

Move-to-front strategy

Least recently used (LRU)

	Virtual memory								
	Memory			Disque					
Adress	1	2	3	4	5	6	7	8	State
Pages	P_3	P_7	P_2	P_6	P_5	P_1	P_8	P_4	E
Pages	P_5	P_3	P_7	P_2	P_6	P_1	P_8	P_4	E_1

Move-ahead strategy

Ranking algorithm

	Virtual memory								
	Memory			Disk					
Adress	1	2	3	4	5	6	7	8	State
Pages	P_3	P_7	P_2	P_6	P_5	P_1	P_8	P_4	E
Pages	P_3	P_7	P_2	P_5	P_6	P_1	P_8	P_4	E_2

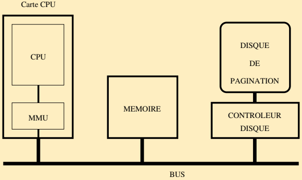
Problem

Performance : mean response time (memory access << disk access)
Choose the strategy that achieves the best long-term performance

Cache modelling

Virtual memory

Paging in OS



- cache hierarchy (processor)
- data caches (databases)
- proxy-web (internet)
- routing tables (networking)
- ...

State of the system : Page position

Huge number of pages, small memory capacity

Move-to-front strategy

Least recently used (LRU)

	Virtual memory								
	Memory			Disque					
Adress	1	2	3	4	5	6	7	8	State
Pages	P_3	P_7	P_2	P_6	P_5	P_1	P_8	P_4	E
Pages	P_5	P_3	P_7	P_2	P_6	P_1	P_8	P_4	E_1

Move-ahead strategy

Ranking algorithm

	Virtual memory								
	Memory			Disk					
Adress	1	2	3	4	5	6	7	8	State
Pages	P_3	P_7	P_2	P_6	P_5	P_1	P_8	P_4	E
Pages	P_3	P_7	P_2	P_5	P_6	P_1	P_8	P_4	E_2

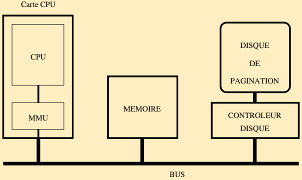
Problem

Performance : mean response time (memory access << disk access)
Choose the strategy that achieves the best long-term performance

Cache modelling

Virtual memory

Paging in OS



- cache hierarchy (processor)
- data caches (databases)
- proxy-web (internet)
- routing tables (networking)
- ...

State of the system : Page position

Huge number of pages, small memory capacity

Move-to-front strategy

Least recently used (LRU)

	Virtual memory								
	Memory			Disque					
Adress	1	2	3	4	5	6	7	8	State
Pages	P_3	P_7	P_2	P_6	P_5	P_1	P_8	P_4	E
Pages	P_5	P_3	P_7	P_2	P_6	P_1	P_8	P_4	E_1

Move-ahead strategy

Ranking algorithm

	Virtual memory								
	Memory			Disk					
Adress	1	2	3	4	5	6	7	8	State
Pages	P_3	P_7	P_2	P_6	P_5	P_1	P_8	P_4	E
Pages	P_3	P_7	P_2	P_5	P_6	P_1	P_8	P_4	E_2

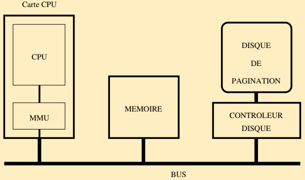
Problem

Performance : mean response time (memory access $<<$ disk access)
Choose the strategy that achieves the best long-term performance

Cache modelling

Virtual memory

Paging in OS



- cache hierarchy (processor)
- data caches (databases)
- proxy-web (internet)
- routing tables (networking)
- ...

State of the system : Page position

Huge number of pages, small memory capacity

Move-to-front strategy

Least recently used (LRU)

	Virtual memory								
	Memory			Disque					
Adress	1	2	3	4	5	6	7	8	State
Pages	P_3	P_7	P_2	P_6	P_5	P_1	P_8	P_4	E
Pages	P_5	P_3	P_7	P_2	P_6	P_1	P_8	P_4	E_1

Move-ahead strategy

Ranking algorithm

	Virtual memory								
	Memory			Disk					
Adress	1	2	3	4	5	6	7	8	State
Pages	P_3	P_7	P_2	P_6	P_5	P_1	P_8	P_4	E
Pages	P_3	P_7	P_2	P_5	P_6	P_1	P_8	P_4	E_2

Problem

Performance : mean response time (memory access $<<$ disk access)
Choose the strategy that achieves the best long-term performance

Modelling

State of the system

N = number of pages
State = permutation of $\{1, \cdots, N\}$
Size of the state space = $N!$
$\Longrightarrow$ numerically untractable

Example : Linux system
- Size of page = $4kb$
- Memory size = $1Gb$
- Swap disk size = $1Gb$

Size of the state space = 500000!
exercise : compute the order of magnitude

Flow modelling

Requests are random
Request have the same probability distributions
Requests are stochastically independent
$\{R_n\}_{n\in\mathbb{N}}$ random sequence of i.i.d. requests

State space reduction

P_A = More frequent page
All other pages have the same frequency.

$$a = \mathbb{P}(R_n = P_A), \quad b = \mathbb{P}(R_n = P_i),$$

$$a > b, \quad a + (N-1)b = 1.$$

$\{X_n\}_{n\in\mathbb{N}}$ position of page P_A at time n.
State space = $\{1, \cdots, N\}$ (size reduction)
Markov chain (state dependent policy)

Modelling

State of the system

N = number of pages
State = permutation of $\{1, \cdots, N\}$
Size of the state space = $N!$
$\Longrightarrow$ numerically untractable

Example : Linux system
- Size of page = $4kb$
- Memory size = $1Gb$
- Swap disk size = $1Gb$

Size of the state space = 500000!
exercise : compute the order of magnitude

Flow modelling

Requests are random
Request have the same probability distributions
Requests are stochastically independent
$\{R_n\}_{n\in\mathbb{N}}$ random sequence of i.i.d. requests

State space reduction

P_A = More frequent page
All other pages have the same frequency.

$$a = \mathbb{P}(R_n = P_A), \quad b = \mathbb{P}(R_n = P_i),$$

$$a > b, \quad a + (N-1)b = 1.$$

$\{X_n\}_{n\in\mathbb{N}}$ position of page P_A at time n.
State space = $\{1, \cdots, N\}$ (size reduction)
Markov chain (state dependent policy)

Modelling

State of the system

N = number of pages
State = permutation of $\{1, \cdots, N\}$
Size of the state space = $N!$
$\Longrightarrow$ numerically untractable

Example : Linux system
- Size of page = $4kb$
- Memory size = $1Gb$
- Swap disk size = $1Gb$

Size of the state space = $500000!$
exercise : compute the order of magnitude

Flow modelling

Requests are random
Request have the same probability distributions
Requests are stochastically independent
$\{R_n\}_{n \in \mathbb{N}}$ random sequence of i.i.d. requests

State space reduction

P_A = More frequent page
All other pages have the same frequency.

$$a = \mathbb{P}(R_n = P_A), \quad b = \mathbb{P}(R_n = P_i),$$

$$a > b, \quad a + (N-1)b = 1.$$

$\{X_n\}_{n \in \mathbb{N}}$ position of page P_A at time n.
State space = $\{1, \cdots, N\}$ (size reduction)
Markov chain (state dependent policy)

Modelling

State of the system

N = number of pages
State = permutation of $\{1, \cdots, N\}$
Size of the state space = $N!$
$\Longrightarrow$ numerically untractable

Example : Linux system
- Size of page = $4kb$
- Memory size = $1Gb$
- Swap disk size = $1Gb$

Size of the state space = 500000!
exercise : compute the order of magnitude

Flow modelling

Requests are random
Request have the same probability distributions
Requests are stochastically independent
$\{R_n\}_{n \in \mathbb{N}}$ random sequence of i.i.d. requests

State space reduction

P_A = More frequent page
All other pages have the same frequency.

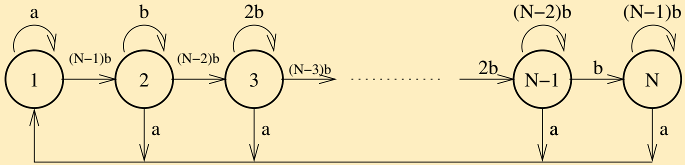
$$a = \mathbb{P}(R_n = P_A), \quad b = \mathbb{P}(R_n = P_i),$$

$$a > b, \quad a + (N-1)b = 1.$$

$\{X_n\}_{n \in \mathbb{N}}$ position of page P_A at time n.
State space = $\{1, \cdots, N\}$ (size reduction)
Markov chain (state dependent policy)

Move to front analysis

Markov chain graph



Transition matrix

$$\begin{bmatrix} a & (N-1)b & 0 & \cdots & \cdots & 0 \\ a & b & (N-2)b & \ddots & & \vdots \\ \vdots & 0 & 2b & (N-3)b & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & & \ddots & (N-2)b & b \\ a & 0 & \cdots & \cdots & 0 & (N-1)b \end{bmatrix}.$$

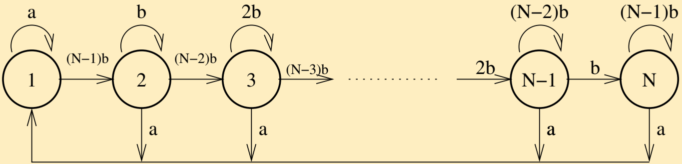
Example

$N = 8$, $a = 0.3$ and $b = 0.1$

$$\pi = \begin{bmatrix} 0.30 \\ 0.23 \\ 0.18 \\ 0.12 \\ 0.08 \\ 0.05 \\ 0.03 \\ 0.01 \end{bmatrix}.$$

Move to front analysis

Markov chain graph



Transition matrix

$$\begin{bmatrix} a & (N-1)b & 0 & \cdots & \cdots & 0 \\ a & b & (N-2)b & \ddots & & \vdots \\ \vdots & 0 & 2b & (N-3)b & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & & \ddots & (N-2)b & b \\ a & 0 & \cdots & \cdots & 0 & (N-1)b \end{bmatrix}.$$

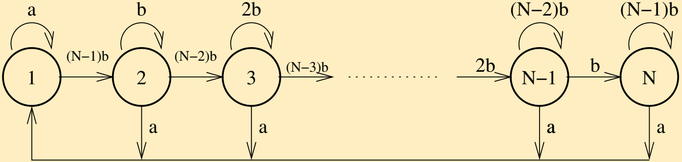
Example

$N = 8$, $a = 0.3$ and $b = 0.1$

$$\pi = \begin{bmatrix} 0.30 \\ 0.23 \\ 0.18 \\ 0.12 \\ 0.08 \\ 0.05 \\ 0.03 \\ 0.01 \end{bmatrix}.$$

Move to front analysis

Markov chain graph



Transition matrix

$$\begin{bmatrix} a & (N-1)b & 0 & \cdots & \cdots & 0 \\ a & b & (N-2)b & \ddots & & \vdots \\ \vdots & 0 & 2b & (N-3)b & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & & \ddots & (N-2)b & b \\ a & 0 & \cdots & \cdots & 0 & (N-1)b \end{bmatrix}.$$

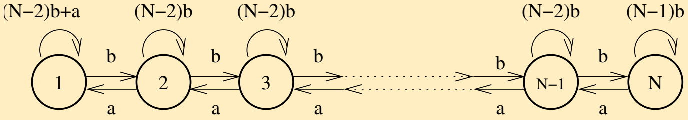
Example

$N = 8$, $a = 0.3$ and $b = 0.1$

$$\pi = \begin{bmatrix} 0.30 \\ 0.23 \\ 0.18 \\ 0.12 \\ 0.08 \\ 0.05 \\ 0.03 \\ 0.01 \end{bmatrix}.$$

Move ahead analysis

Markov chain graph



Transition matrix

$$\begin{bmatrix} a+(N-2)b & b & 0 & \cdots & \cdots & 0 \\ a & (N-2)b & b & \ddots & & \vdots \\ 0 & a & (N-2)b & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & (N-2)b & b \\ 0 & 0 & \cdots & 0 & a & (N-1)b \end{bmatrix}.$$

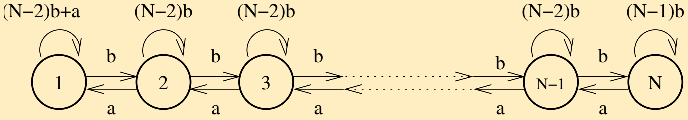
Example

$N = 8$, $a = 0.3$ and $b = 0.1$

$$\pi = \begin{bmatrix} 0.67 \\ 0.22 \\ 0.07 \\ 0.02 \\ 0.01 \\ 0.01 \\ 0.00 \\ 0.00 \end{bmatrix}.$$

Move ahead analysis

Markov chain graph



Transition matrix

$$\begin{bmatrix} a+(N-2)b & b & 0 & \cdots & \cdots & 0 \\ a & (N-2)b & b & \ddots & & \vdots \\ 0 & a & (N-2)b & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & (N-2)b & b \\ 0 & 0 & \cdots & 0 & a & (N-1)b \end{bmatrix}.$$

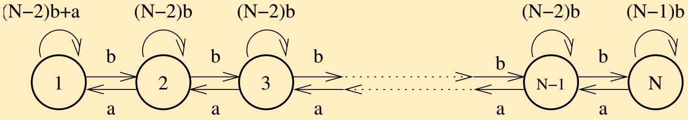
Example

$N = 8$, $a = 0.3$ and $b = 0.1$

$$\pi = \begin{bmatrix} 0.67 \\ 0.22 \\ 0.07 \\ 0.02 \\ 0.01 \\ 0.01 \\ 0.00 \\ 0.00 \end{bmatrix}.$$

Move ahead analysis

Markov chain graph



Transition matrix

$$\begin{bmatrix} a+(N-2)b & b & 0 & \cdots & \cdots & 0 \\ a & (N-2)b & b & \ddots & & \vdots \\ 0 & a & (N-2)b & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & (N-2)b & b \\ 0 & 0 & \cdots & 0 & a & (N-1)b \end{bmatrix}.$$

Example

$N = 8$, $a = 0.3$ and $b = 0.1$

$$\pi = \begin{bmatrix} 0.67 \\ 0.22 \\ 0.07 \\ 0.02 \\ 0.01 \\ 0.01 \\ 0.00 \\ 0.00 \end{bmatrix}.$$

Performances

Steady state

$$MF = \begin{bmatrix} 0.30 \\ 0.23 \\ 0.18 \\ 0.12 \\ 0.08 \\ 0.05 \\ 0.03 \\ 0.01 \end{bmatrix} MA = \begin{bmatrix} 0.67 \\ 0.22 \\ 0.07 \\ 0.02 \\ 0.01 \\ 0.01 \\ 0.00 \\ 0.00 \end{bmatrix}.$$

Move to front

$$\pi(i) = \frac{(N-1-i)\cdots(N-2)(N-1)b^{i-1}}{(a+(N-i)b)\cdots(a+(N-2)b)(a+(N-1)b)}\pi_1.$$

Move ahead

$$\pi_i = (\frac{b}{a})^{i-1}\frac{1-\frac{b}{a}}{1-(\frac{b}{a})^N}.$$

Cache miss

Memory size	Move to front	Move Ahead
0	1.00	1.00
1	0.70	0.33
2	0.47	0.11
3	0.28	0.04
4	0.17	0.02
5	0.09	0.01
6	0.04	0.00
7	0.01	0.00
8	0.00	0.00

Best strategy :
Move ahead

Comments

Self-ordering protocol : decreasing probability
Convergence speed to steady state :
Move to front : 0.7^n Move ahead : 0.92^n
Tradeoff between "stabilization" and long term performance
Depends on the input flow of requests

Performances

Steady state

$$MF = \begin{bmatrix} 0.30 \\ 0.23 \\ 0.18 \\ 0.12 \\ 0.08 \\ 0.05 \\ 0.03 \\ 0.01 \end{bmatrix} MA = \begin{bmatrix} 0.67 \\ 0.22 \\ 0.07 \\ 0.02 \\ 0.01 \\ 0.01 \\ 0.00 \\ 0.00 \end{bmatrix}.$$

Move to front

$$\pi(i) = \frac{(N-1-i)\cdots(N-2)(N-1)b^{i-1}}{(a+(N-i)b)\cdots(a+(N-2)b)(a+(N-1)b)}\pi_1.$$

Move ahead

$$\pi_i = (\frac{b}{a})^{i-1}\frac{1-\frac{b}{a}}{1-(\frac{b}{a})^N}.$$

Cache miss

Memory size	Move to front	Move Ahead
0	1.00	1.00
1	0.70	0.33
2	0.47	0.11
3	0.28	0.04
4	0.17	0.02
5	0.09	0.01
6	0.04	0.00
7	0.01	0.00
8	0.00	0.00

Best strategy :
Move ahead

Comments

Self-ordering protocol : decreasing probability
Convergence speed to steady state :
Move to front : 0.7^n Move ahead : 0.92^n
Tradeoff between “stabilization” and long term performance
Depends on the input flow of requests

Performances

Steady state

$$MF = \begin{bmatrix} 0.30 \\ 0.23 \\ 0.18 \\ 0.12 \\ 0.08 \\ 0.05 \\ 0.03 \\ 0.01 \end{bmatrix} MA = \begin{bmatrix} 0.67 \\ 0.22 \\ 0.07 \\ 0.02 \\ 0.01 \\ 0.01 \\ 0.00 \\ 0.00 \end{bmatrix}.$$

Move to front

$$\pi(i) = \frac{(N-1-i)\cdots(N-2)(N-1)b^{i-1}}{(a+(N-i)b)\cdots(a+(N-2)b)(a+(N-1)b)}\pi_1.$$

Move ahead

$$\pi_i = (\frac{b}{a})^{i-1}\frac{1-\frac{b}{a}}{1-(\frac{b}{a})^N}.$$

Cache miss

Memory size	Move to front	Move Ahead
0	1.00	1.00
1	0.70	0.33
2	0.47	0.11
3	0.28	0.04
4	0.17	0.02
5	0.09	0.01
6	0.04	0.00
7	0.01	0.00
8	0.00	0.00

Best strategy :
Move ahead

Comments

Self-ordering protocol : decreasing probability
Convergence speed to steady state :
Move to front : 0.7^n Move ahead : 0.92^n
Tradeoff between “stabilization” and long term performance
Depends on the input flow of requests

Performances

Steady state

$$MF = \begin{bmatrix} 0.30 \\ 0.23 \\ 0.18 \\ 0.12 \\ 0.08 \\ 0.05 \\ 0.03 \\ 0.01 \end{bmatrix} MA = \begin{bmatrix} 0.67 \\ 0.22 \\ 0.07 \\ 0.02 \\ 0.01 \\ 0.01 \\ 0.00 \\ 0.00 \end{bmatrix}.$$

Move to front

$$\pi(i) = \frac{(N-1-i)\cdots(N-2)(N-1)b^{i-1}}{(a+(N-i)b)\cdots(a+(N-2)b)(a+(N-1)b)}\pi_1.$$

Move ahead

$$\pi_i = (\frac{b}{a})^{i-1}\frac{1-\frac{b}{a}}{1-(\frac{b}{a})^N}.$$

Cache miss

Memory size	Move to front	Move Ahead
0	1.00	1.00
1	0.70	0.33
2	0.47	0.11
3	0.28	0.04
4	0.17	0.02
5	0.09	0.01
6	0.04	0.00
7	0.01	0.00
8	0.00	0.00

Best strategy :
Move ahead

Comments

Self-ordering protocol : decreasing probability
Convergence speed to steady state :
Move to front : 0.7^n Move ahead : 0.92^n
Tradeoff between “stabilization” and long term performance
Depends on the input flow of requests

Outline

Synthesis : Modelling and Performance

Methodology

1. Identify states of the system
2. Estimate transition parameters, build the Markov chain (verify properties)
3. Specify performances as a function of steady-state
4. Compute steady-state distribution and steady-state performance
5. Analyse performances as a function of input parameters

Classical methods to compute the steady state

1. Analytical formulae : structure of the Markov chain (closed form)
2. Formal computation ($N < 50$)
3. Direct numerical computation (classical linear algebra kernels) ($N < 1000$)
4. Iterative numerical computation (classical linear algebra kernels) ($N < 100.000$)
5. Model adapted numerical computation ($N < 10.000.000$)
6. Simulation of random trajectories (sampling)

Synthesis : Modelling and Performance

Methodology

1. Identify states of the system
2. Estimate transition parameters, build the Markov chain (verify properties)
3. Specify performances as a function of steady-state
4. Compute steady-state distribution and steady-state performance
5. Analyse performances as a function of input parameters

Classical methods to compute the steady state

1. Analytical formulae : structure of the Markov chain (closed form)
2. Formal computation ($N < 50$)
3. Direct numerical computation (classical linear algebra kernels) ($N < 1000$)
4. Iterative numerical computation (classical linear algebra kernels) ($N < 100.000$)
5. Model adapted numerical computation ($N < 10.000.000$)
6. Simulation of random trajectories (sampling)

Bibliography

Books

- O. Haggstrom. **Finite Markov Chains and algorithmic applications**. Cambridge University Press. 2001.
- P. Brémaud **Markov chains: Gibbs fields, Monte Carlo Simulation and Queues**. Springer-Verlag, 1999
- D. A. Levin, Y. Peres, et E. L. Wilmer **Markov Chains and Mixing Times**, American Mathematical Society,2009.

Websites

- Virtual Laboratories in Probability and Statistics
  `http://www.math.uah.edu/stat/`
- The MacTutor History of Mathematics archive (photos)
  `http://www-history.mcs.st-and.ac.uk/`