Mathematical models for computer systems behaviour

Goals: predict computer system behaviours
- performances measurements,
- comparison of systems,
- dimensioning,

Methodology:
- modelling environment (stochastic process)
- modelling system (automaton)
- behaviour = reaction of automaton on stochastic stimuli
Organisation

Automata + probabilistic transitions:
  Discrete time Markov chains

Automata + probabilistic transitions + time:
  Poisson processes, continuous time Markov chains

State space structure:
  Simple queues, product form queuing networks
  Stochastic automata networks

Simulation of Markov chains:
  Direct simulation/perfect simulation
Applications

- Operating systems
- Networks protocols
- Manufacturing systems
- Production lines
- Middlewares
- ...
Scientific domains

Applied mathematics:
- stochastic processes, ergodic theory,...
- Markov processes

References:
- R. Nelson Probability theory with...
- R. Jain The art of computer systems performance analysis
A first example: Flip-flop

Transition matrix

\[
P = \begin{pmatrix} 1-p & p \\ q & 1-q \end{pmatrix}
\]

\[
P(X_{n+1} = B | X_n = A) = p
\]

\[
P(X_{n+1} = A | X_n = A) = 1 - p
\]

\[
P(X_{n+1} = A | X_n = B) = q
\]

\[
P(X_{n+1} = B | X_n = B) = 1 - q
\]
Long run behaviour

$$\pi_n = P(X_n = A)$$

$$\pi_{n+1} = \pi_n (1 - p) + (1 - \pi_n) q = q + (1 - p - q) \pi_n$$

Linear recurrence equation

Case 1: $$|1 - p - q| < 1$$

$$\pi_n = \frac{q}{p + q} + \left( \pi_0 - \frac{q}{p + q} \right) (1 - p - q)^n$$

Case 2: $$1 - p - q = 1 \Rightarrow p = q = 0$$

Case 3: $$1 - p - q = -1 \Rightarrow p = q = 1$$
Results

Convergence

\[ \pi_A = \lim_{n \to \infty} P(X_n = A) = \frac{q}{p + q}; \]
\[ \pi_B = \lim_{n \to \infty} P(X_n = B) = \frac{p}{p + q}; \]

Geometric

\[ (1 - p - q)^n \]

Asymptotic satisfies

\[ \pi_A = \pi_A (1 - p) + \pi_B q \]
\[ \pi_B = \pi_A p + \pi_B (1 - q) \]

Lack of memory
Ergodic convergence

\[
\pi_A = \lim P(X_n = A) = \lim \frac{1}{n} \sum_{i=1}^{n} 1(X_i = A);
\]

\[
\pi_B = \lim P(X_n = B) = \lim \frac{1}{n} \sum_{i=1}^{n} 1(X_i = A)
\]

Performances indexes:
link utilisation
communication delays
Discrete time Markov chains

\[ \{X_n\}_{n \in \mathbb{N}} \quad \text{Trajectory of the system} \]
\[ \text{Discrete state space} \]
\[ \text{Step by step evolution} \]

\[ P(X_{n+1} = j \mid X_n = i, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) = P(X_{n+1} = j \mid X_n = i) \]

Lack of memory

Homogeneity (in time)

\[ P(X_{n+1} = j \mid X_n = i) = P(X_1 = j \mid X_0 = i) = p_{i,j} \]

Transition matrix (stochastic)

\[ P = \begin{pmatrix} p_{i,j} \end{pmatrix} \quad p_{i,j} \geq 0, \quad \sum_j p_{i,j} = 1. \]
Chapman-Kolmogorov equations

\[ P(X_2 = j \mid X_0 = i) = \sum_k P(X_2 = j \mid X_1 = k) P(X_1 = k \mid X_0 = i) = \sum_k p_{i,k} p_{k,j} \]

Iteration ---> product of matrices

Asymptotic behaviour

\[ \lim_{n \to \infty} P^n = ?? \]
Classification of states

Absorbing state

Transient states

Aperiodic recurrent states: irreducible class

Periodic recurrent states
Convergence theorem

\( \left\{ X_n \right\}_{n \in \mathbb{N}} \) Homogeneous, aperiodic, irreducible Markov chain (finite state space)

\[
\lim_{n \to \infty} P(X_n = j \mid X_0 = i) = \pi_j
\]

\( \pi = (\pi_1, \pi_2, \ldots, \pi_N) \) Steady-state vector

Unique solution of the linear system \( \pi = \pi P \)

Geometric convergence (module of the second eigenvalue)
Ergodic theorem

\( \{X_n\}_{n \in \mathbb{N}} \) Homogeneous, aperiodic, irreducible Markov chain (finite state space)

For any function \( f \) (cost function)

\[
\lim_{n \to \infty} \frac{1}{n} \sum_{k=0}^{n-1} f(X_k) = \sum_{i} \pi_i f(i)
\]

Estimation of the steady-state distribution by
-> simulation

\[
\lim_{n \to \infty} \frac{1}{n} \sum_{k=0}^{n-1} 1_{X_k=i} = \pi_i
\]
Equilibrium equations

Interpretation:

$$\sum_i \pi_i p_{i,j} = \pi_j \sum_k p_{j,k} = \pi_j$$

$$\iff \pi = \pi P$$
Example: cache management

Cache space Memory space

Optimisation: Time to access cache $\ll$ Time to access memory

Cache replacement policy

$M =$ cache size

$N =$ global memory size
LRU policy

Cache hit

Cache miss
Move ahead policy

Cache hit

Cache miss
Environment model

Program = sequence of memory accesses

hypothesis 1:
    independent sequence
    same distribution

State space: permutations of (1,...,N)
    size: N!
State space reduction

Hypothesis 2:

one reference A is more frequent
others are equally distributed
(uniform)

\[ a = P(\text{reference A}), \]
\[ b = P(\text{reference other than A}), \]
\[ a + (N - 1)b = 1. \]
Markov model

\[ \{X_n\}_{n \in \mathbb{N}} \quad X_n = \text{position of reference A at step } n \]

Markov chain: homogeneous, aperiodic, irreducible

=> convergence and ergodicity

=> computation of the steady-state
\[
P_{\text{LRU}} = \begin{bmatrix}
    a & (N-1)b & 0 & 0 & 0 \\
    a & b & (N-2)b & 0 & 0 \\
    a & 0 & 2b & (N-3)b & 0 \\
    a & 0 & 0 & 0 & 0 \\
    a & 0 & 0 & 0 & 0 \\
    a & 0 & 0 & (N-2)b & b \\
    a & 0 & 0 & 0 & (N-1)b
\end{bmatrix}
\]

\[
\pi_i \cdot (N - i)b = \pi_{i+1} \cdot (a + (N - i - 1)b)
\]
Move ahead

\[
P_{MA} = \begin{bmatrix}
1-b & b & 0 & 0 \\
a & 1-a-b & b & 0 \\
0 & a & 1-a-b & b \\
0 & 0 & a & 1-a-b \\
0 & 0 & 0 & a \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
\pi_i = C \left( \frac{b}{a} \right)^i
\]

\[
\pi_i (a + b) = \pi_{i+1} a + \pi_{i-1} b \quad \Rightarrow \quad \pi_i b = \pi_{i+1} a
\]
Numerical example

Toy example  N=8  a=0.3   b=0.1

Steady-state probability vectors

LRU [0.30  0.23  0.18  0.12  0.08  0.05  0.03  0.01 ]

MA   [0.67  0.22  0.07  0.02  0.01  0.01  0.00  0.00 ]

Self optimising algorithm

Rapid decreasing
## Cache miss evaluation

<table>
<thead>
<tr>
<th></th>
<th>Page A</th>
<th></th>
<th>Global</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M LRU</td>
<td>MA</td>
<td>M LRU</td>
</tr>
<tr>
<td>0</td>
<td>1.00</td>
<td>1.00</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0.70</td>
<td>0.33</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0.47</td>
<td>0.11</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0.29</td>
<td>0.04</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0.17</td>
<td>0.02</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>0.09</td>
<td>0.01</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>0.04</td>
<td>0.00</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>0.01</td>
<td>0.00</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>0.00</td>
<td>0.00</td>
<td>8</td>
</tr>
</tbody>
</table>
Example: conclusion

Self adapting algorithm,
“Minimise” cache miss
Move-ahead better than LRU

Speed of convergence ?

2nd module of eigenvalue
  LRU : 0.7
  MA  : 0.96
LRU reaches stationary regime more quickly than MA !!!
Example: conclusion

Hypothesis 2:
relaxing uniformity (ex Zipf law)

==> same behaviour (simulation)

Hypothesis 1:
model of evolution of probability of references

==> depends on the “variability” of the process
compromise between rate of convergence and speed of evolution!
Generalizations

- Infinite state space
  Same behaviour (if non recurrent null)
- Transient analysis
  study of powers of P
- Generalized Markov Processes
  Timed states
  - computation of steady-state
  - weighted probabilities
Solving Markov chains

N< 50  Formal methods Maple
N<500 Classical numerical methods
   (Gaussian elimination,…) Mathematica, Lapack…
N<100000 Iterative methods, preconditionning,
N< 10000000 Specific numerical algorithms
   (sparse matrices, …) Marca, Peps,…
N>100000000 Simulation

Over … Approximations and analytical techniques
Links in computer science

Common formalisms with verification tools

Queuing networks

Petri nets

Process algebra

Automata networks

==> state space construction,
behaviours specifications