Evaluation de performances

Master 2R SL module EP

Jean-Marc Vincent, Arnaud Legrand et Bruno Gaujal[1]

[1]Laboratory ID-IMAG
MESCAL Project
Universities of Grenoble
{Jean-Marc.Vincent,Arnaud.Legrand,Bruno.Gaujal}@imag.fr
http://www-id.imag.fr/Laboratoire/Membres/Legrand_Arnaud/teaching/index.php

Outline

Outline

Research activities in performance evaluation

Teams in Grenoble

- Mescal project : large systems (clusters and grids)
- Moais project : interactive parallel systems
- Drakkar team : networking
- Sardes : middleware
- Hadas : data-bases
- etc

Industrial collaborations

- France-Télécom R & D : load injectors, performances of middlewares
- HP-Labs : cluster computing, benchmarking
- Bull : benchmarking, performances analysis

Research activities in performance evaluation

Teams in Grenoble

- Mescal project : large systems (clusters and grids)
- Moais project : interactive parallel systems
- Drakkar team : networking
- Sardes : middleware
- Hadas : data-bases
- etc

Industrial collaborations

- France-Télécom R & D : load injectors, performances of middlewares
- HP-Labs : cluster computing, benchmarking
- Bull : benchmarking, performances analysis

Application context (1)

Complexity of computer systems

- **hierarchy** : level decomposition : OS / Middleware / Application
- **distribution** : asynchronous resources : memory, CPU, network
- **dynamicity** : architecture and environment (reliability, mobility,...)
- **scalability** : number of components (autonomous management)

Typical problems

- Minimize losses in routing policies
- Minimize active waiting in threads scheduling
- Maximize cache hits
- Optimise block sizes in parallel applications
- Maximize troughput of communication systems
- Fix time-outs, reemission periods, ...
- Fix the granularity : pages, blocks, tables, message sizes...

Application context (1)

Complexity of computer systems

- **hierarchy** : level decomposition : OS / Middleware / Application
- **distribution** : asynchronous resources : memory, CPU, network
- **dynamicity** : architecture and environment (reliability, mobility,...)
- **scalability** : number of components (autonomous management)

Typical problems

- Minimize losses in routing policies
- Minimize active waiting in threads scheduling
- Maximize cache hits
- Optimise block sizes in parallel applications
- Maximize troughput of communication systems
- Fix time-outs, reemission periods, ...
- Fix the granularity : pages, blocks, tables, message sizes...

Application context (2)

Typical “hot” applications

- **Peer to peer systems** : dimensionning, control
- **Mobile networks** : ad-hoc networking, reactivity, coherence
- **Grids** : resources utilization, scheduling
- etc

Other application domains

- production systems : production lines, logistic,...
- embedded systems
- modelling of complex systems : biology, sociology,...
- etc

Application context (2)

Typical “hot” applications

- **Peer to peer systems** : dimensionning, control
- **Mobile networks** : ad-hoc networking, reactivity, coherence
- **Grids** : resources utilization, scheduling
- etc

Other application domains

- production systems : production lines, logistic,...
- embedded systems
- modelling of complex systems : biology, sociology,...
- etc

Outline

Development of parallel/distributed aplications

- **Qualitative specifications :** Is the result correct ?
  - properties verifications : formal/automatic proofs
  - testing : critical dataset
- **Quantitative specifications :**Is the result obtain in an acceptable time ?
  - performance model
  - performance measurements
- **Problem identification**
  - debugging, log analysis
  - performance statistical analysis
- **Modification**
  - source code / libraries / OS / architecture
  - parameters of the system : dimensionning
  - control algorithms : tuning

Development of parallel/distributed aplications

- **Qualitative specifications :** Is the result correct ?
  - properties verifications : formal/automatic proofs
  - testing : critical dataset
- **Quantitative specifications :**Is the result obtain in an acceptable time ?
  - performance model
  - performance measurements
- **Problem identification**
  - debugging, log analysis
  - performance statistical analysis
- **Modification**
  - source code / libraries / OS / architecture
  - parameters of the system : dimensionning
  - control algorithms : tuning

Development of parallel/distributed aplications

- **Qualitative specifications :** Is the result correct ?
  - properties verifications : formal/automatic proofs
  - testing : critical dataset
- **Quantitative specifications :**Is the result obtain in an acceptable time ?
  - performance model
  - performance measurements
- **Problem identification**
  - debugging, log analysis
  - performance statistical analysis
- **Modification**
  - source code / libraries / OS / architecture
  - parameters of the system : dimensionning
  - control algorithms : tuning

Development of parallel/distributed aplications

- **Qualitative specifications :** Is the result correct ?
  - properties verifications : formal/automatic proofs
  - testing : critical dataset
- **Quantitative specifications :**Is the result obtain in an acceptable time ?
  - performance model
  - performance measurements
- **Problem identification**
  - debugging, log analysis
  - performance statistical analysis
- **Modification**
  - source code / libraries / OS / architecture
  - parameters of the system : dimensionning
  - control algorithms : tuning

Dual analysis

Understand the behavior of a distributed application

1. identification of distributed patterns, states of the system
2. pattern verification
3. time evaluation
4. global analysis of the execution and performance synthesis
5. system monitoring
6. global cost evaluation for the application user

Understand resources utilization

1. hierarchical model of resources
2. evaluation of utilization at :
   application level; executive runtime;
   operating system; hardware architecture
3. global cost evaluation for the resources manager

Dual analysis

Understand the behavior of a distributed application

1. identification of distributed patterns, states of the system
2. pattern verification
3. time evaluation
4. global analysis of the execution and performance synthesis
5. system monitoring
6. global cost evaluation for the application user

Understand resources utilization

1. hierarchical model of resources
2. evaluation of utilization at :
   application level; executive runtime;
   operating system; hardware architecture
3. global cost evaluation for the resources manager

Evaluation methods

From abstraction to physical reality

Model **Method**

Remarks :

Hybrid methods (emulation, load injectors, synthetic programs,...)
Dynamical process of evaluation
Experimentation $\Rightarrow$ Planning experiments methodology

Evaluation methods

From abstraction to physical reality

Model	**Method**
Mathematical $\longrightarrow$	Analysis (formal, numerical, approximation)

Remarks :

Hybrid methods (emulation, load injectors, synthetic programs,...)

Dynamical process of evaluation

Experimentation $\Rightarrow$ Planning experiments methodology

Evaluation methods

From abstraction to physical reality

Model		**Method**
Mathematical	$\longrightarrow$	Analysis (formal, numerical, approximation)
Software	$\longrightarrow$	Simulation (discrete events)

Remarks :

Hybrid methods (emulation, load injectors, synthetic programs,...)
Dynamical process of evaluation
Experimentation $\Rightarrow$ Planning experiments methodology

Evaluation methods

From abstraction to physical reality

Model		**Method**
Mathematical	$\longrightarrow$	Analysis (formal, numerical, approximation)
Software	$\longrightarrow$	Simulation (discrete events)
Prototype	$\longrightarrow$	Observation (measures)

Remarks :

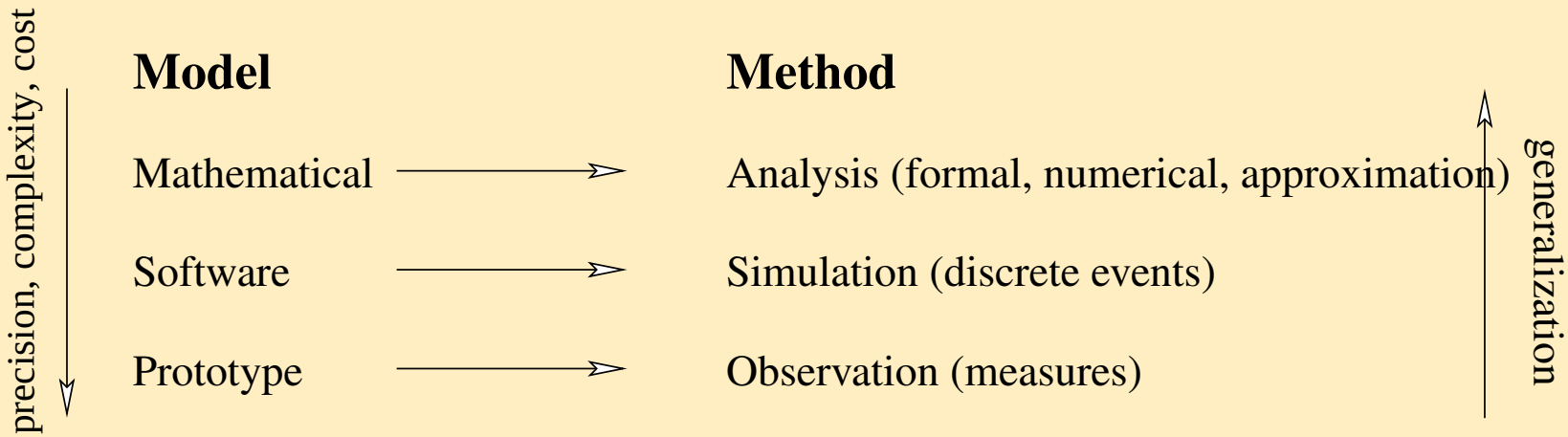
Hybrid methods (emulation, load injectors, synthetic programs,...)

Dynamical process of evaluation

Experimentation $\Rightarrow$ Planning experiments methodology

Evaluation methods

From abstraction to physical reality



Remarks :

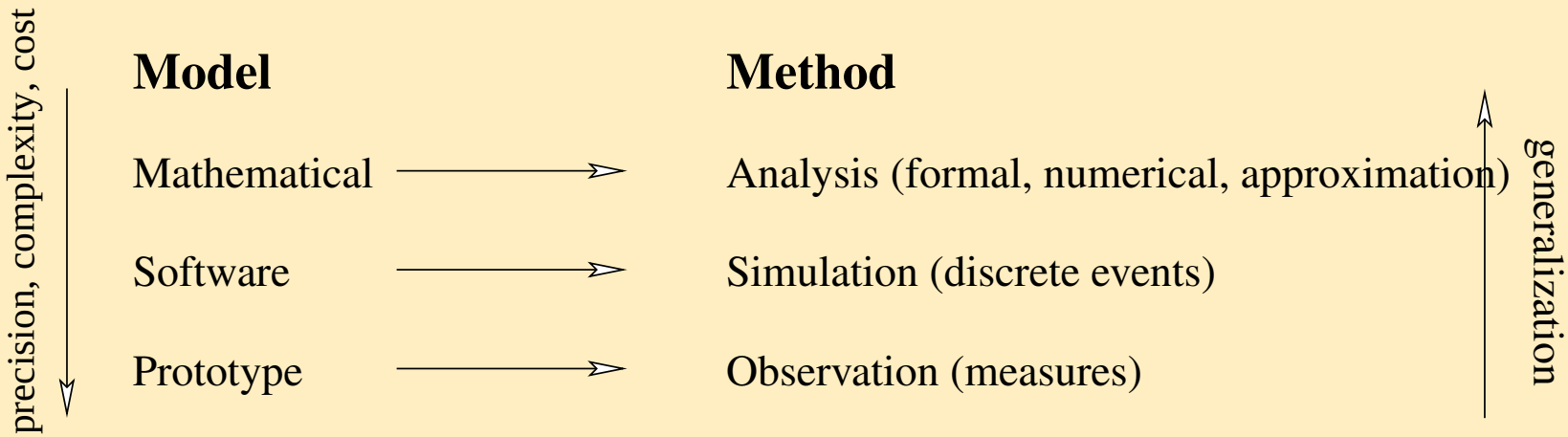
Hybrid methods (emulation, load injectors, synthetic programs,...)

Dynamical process of evaluation

Experimentation ⇒ Planning experiments methodology

Evaluation methods

From abstraction to physical reality



Remarks :

Hybrid methods (emulation, load injectors, synthetic programs,...)
Dynamical process of evaluation
Experimentation $\Rightarrow$ Planning experiments methodology

Steps for a Performance Evaluation Study (Jain)

1. State the goals of the study : level of decision, investment, optimization, technical,...
2. Define system boundaries.
3. List system services and possible outcomes.
4. Select performance metrics.
5. List system and workload parameters
6. Select factors and their values.
7. Select evaluation techniques.
8. Select the workload.
9. Design the experiments.
10. Analyze and interpret the data.
11. Present the results. Start over, if necessary.

Outline

Aim of the course

Objective

1. Be able to analyze and predict performances of parallel/distributed systems
2. Be able to build a software environment that produces the performances indexes.

Methods

1. Specification and identification of problems : modelling
2. Analysis of quantitative models : formal, numerical, simulation
3. Experimentation and statistical data analysis.

Aim of the course

Objective

1. Be able to analyze and predict performances of parallel/distributed systems
2. Be able to build a software environment that produces the performances indexes.

Methods

1. Specification and identification of problems : modelling
2. Analysis of quantitative models : formal, numerical, simulation
3. Experimentation and statistical data analysis.

Organisation of the course

Practical evaluation of systems 8 lectures 1h30

1. Performances of computer systems : quality of service, performance indexes,...
2. Analysis and visualization of univariate statistical data
3. Performance measurments of computer systems : tools and analysis
4. Trace and analysis of distributed applications
5. Random generation of workload
6. Generation of complex structures
7. Simulation of computer systems
8. Software environments for performance evaluation

Organisation of the course (2)

Modelling of computer systems 8 lectures 1h30

1. Deterministic models and quality of service
   Network Calculus (3-4 lectures)
2. Stochastic models of automata
3. Steady-state analysis of systems
4. Trafic modelling
5. Contention modelling

Applications : networking, distributed applications, parallel/grid computing

References : text books

- **The Art of Computer Systems Performance Analysis : Techniques for Experimental Design, Measurment, Simulation and Modeling.** Raj Jain *Wiley 1991 (nouvelles versions)*
  Covers the content of the course, a complete book
- **Performance Evaluation** Jean-Yves Le Boudec EPFL electronic book http://ica1www.epfl.ch/perfeval/lectureNotes.htm
  Covers the statistical part of the course
- **Measuring Computer Performance: A Practitioner's Guide**
  David J. Lilja *Cambridge University press* 2000
  Covers the practical part of measurement and benchmarking
- **Discrete-Event System Simulation** Jerry Banks, John Carson, Barry L. Nelson, David Nicol, *Prentice Hall*, 2004
  Covers the part on simulation

References : journals and conferences

- **General**: JACM, ACM Comp. Surv., JOR, IEEE TSE,...
- **Specialized**: Performance Evaluation, Operation research, MOR, ACM TOMACS, Queueing Systems, DEDS, ...
- **Application**: IEEE TPDS, TC, TN, TAC, Networks,...
- **Theoretical**: Annals of Probability, of Appl. Prob, JAP, Adv. Appl. Prob,...
- **Conferences on performances**: Performance, ACM-SIGMETRICS, TOOLS, MASCOT, INFORMS, ...
- **Conferences on an application domain**: ITC, Europar, IPDPS, Renpar, ...
- **National seminars**: Atelier d'évaluation de performances,...

Evaluation

Homework

- Research article reading (summary)
- Software environment study (install, test, analyse)
- Comparison of software environments

Final exam

- Theoretical aspects
- Articles

Evaluation

Homework

- Research article reading (summary)
- Software environment study (install, test, analyse)
- Comparison of software environments

Final exam

- Theoretical aspects
- Articles

Outline

Networking

Flow performance

- latency, waiting time, response time
- loss probability
- jitter

Operator performance

- bandwidth utilisation
- achievable throughput
- loss rate

Quality of service

contract between user and provider
service guarantees
tradeoff between utilization and QoS

Networking

Flow performance

- latency, waiting time, response time
- loss probability
- jitter

Operator performance

- bandwidth utilisation
- achievable throughput
- loss rate

Quality of service

contract between user and provider
service guarantees
tradeoff between utilization and QoS

Networking

Flow performance

- latency, waiting time, response time
- loss probability
- jitter

Operator performance

- bandwidth utilisation
- achievable throughput
- loss rate

Quality of service

contract between user and provider
service guarantees
tradeoff between utilization and QoS

Parallel processing

Program execution

- makespan, critical path
- speedup, efficiency
- active waiting, communication overlapping
- throughput

System utilization

- cpu utilization, idle time
- memory occupancy
- communication throughput

Parallel programming and scheduling

granularity of the application

tradeoff between utilization and makespan

Parallel processing

Program execution

- makespan, critical path
- speedup, efficiency
- active waiting, communication overlapping
- throughput

System utilization

- cpu utilization, idle time
- memory occupancy
- communication throughput

Parallel programming and scheduling

granularity of the application
tradeoff between utilization and makespan

Parallel processing

Program execution

- makespan, critical path
- speedup, efficiency
- active waiting, communication overlapping
- throughput

System utilization

- cpu utilization, idle time
- memory occupancy
- communication throughput

Parallel programming and scheduling

granularity of the application

tradeoff between utilization and makespan

Distributed applications

Application

- response time
- reactivity
- throughput (number of processed requests/unit time)
- streaming rate

System utilization

- service availability
- resource utilization
- communication throughput

System security

- reliability (error-free period)
- availability

Distributed applications

Application

- response time
- reactivity
- throughput (number of processed requests/unit time)
- streaming rate

System utilization

- service availability
- resource utilization
- communication throughput

System security

- reliability (error-free period)
- availability

Distributed applications

Application

- response time
- reactivity
- throughput (number of processed requests/unit time)
- streaming rate

System utilization

- service availability
- resource utilization
- communication throughput

System security

- reliability (error-free period)
- availability

Synthesis

User point of view

optimize its own performance
- get the maximum amount of resources for its own purpose
- guarantee the higher quality of service

Resource point of view

Contract between users and resources:
- guarantee of “equity”
- optimize the use of resources
- minimize costs by identifying performance bottlenecks

Tradeoff Performance - Cost

Synthesis

User point of view

optimize its own performance
- get the maximum amount of resources for its own purpose
- guarantee the higher quality of service

Resource point of view

Contract between users and resources:
- guarantee of “equity”
- optimize the use of resources
- minimize costs by identifying performance bottlenecks

Tradeoff Performance - Cost

Synthesis

User point of view

optimize its own performance
- get the maximum amount of resources for its own purpose
- guarantee the higher quality of service

Resource point of view

Contract between users and resources:
- guarantee of “equity”
- optimize the use of resources
- minimize costs by identifying performance bottlenecks

Tradeoff Performance - Cost