POLITECNICO DI MILANO





Model Identification for Energy-Aware Management of Web Service Systems

Mara Tanelli[1], **Danilo Ardagna**[1], Marco Lovera[1], Li Zhang[2]

[1]Politecnico di Milano, Dipartimento di Elettronica e Informazione, Italy

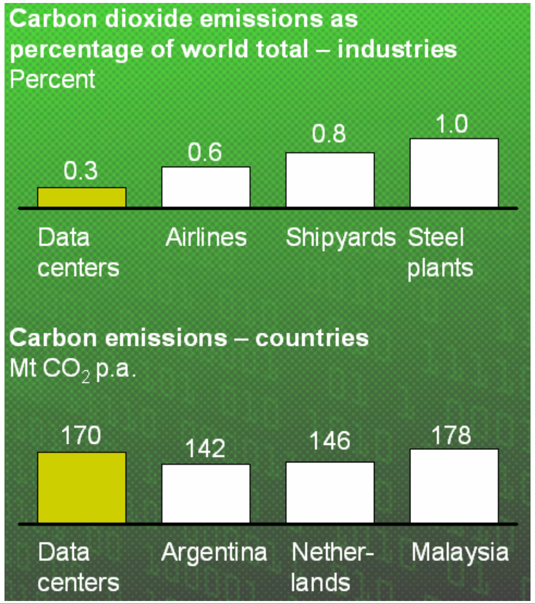
[2]IBM Research, T.J. Watson Research Center, NY

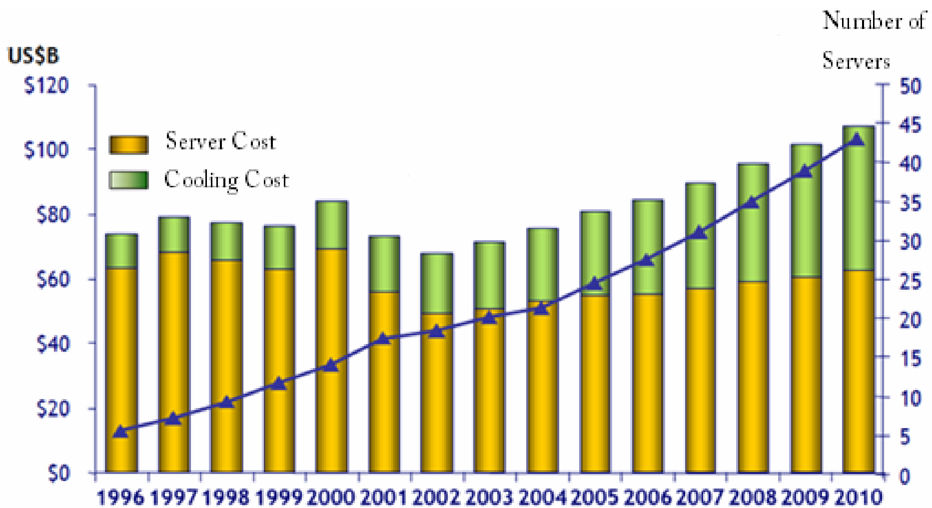
Sydney, December 4 2008

Energy Management in Service Centers

Data Center issues

- Energy consumption
  - 2% of CO2 emission
  - By 2012 energy costs will be 40% of TCO
    - Related costs: cooling, UPS, …
- QoS guarantees and workload variability
- Dynamic resource managment

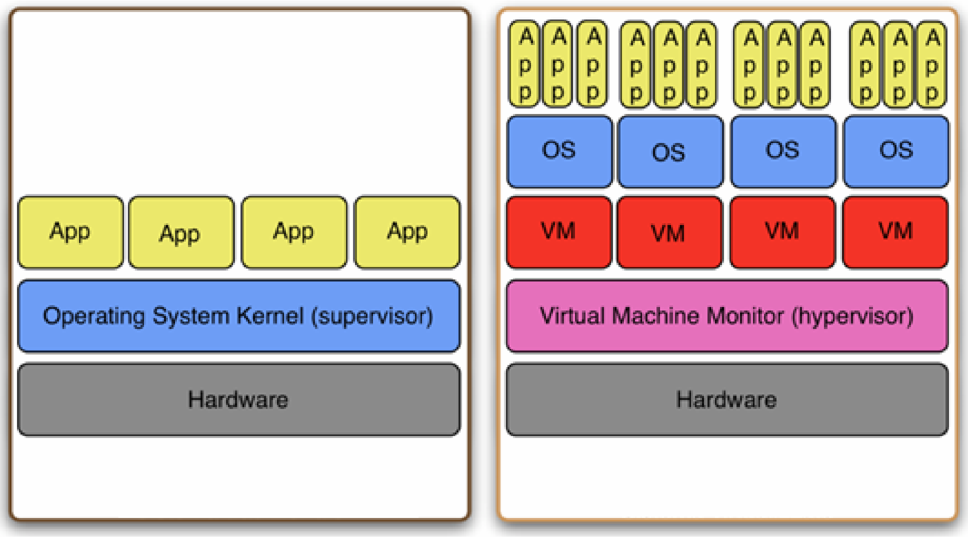




Virtualization

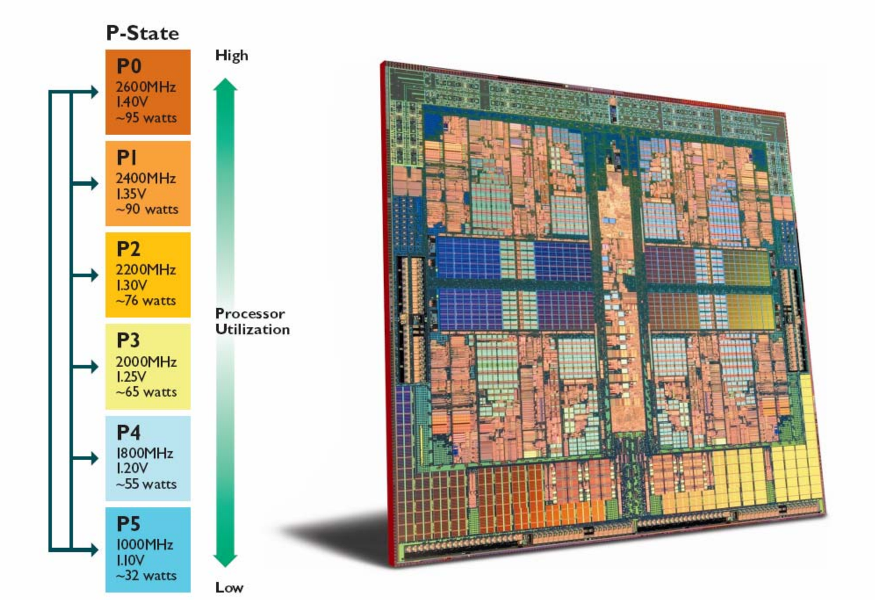
- Hardware resources (CPU, RAM, ecc...) are partitioned and shared among multiple **virtual machines** (VMs)
- The virtual machine monitor (VMM) governs the access to the physical resources among running VMs
- Performance isolation and security

A single OS



VMs run possibly different OSs

Enabling technologies



Dynamic Frequency Scaling (DFS)

- Modern CPUs can work in multiple **p-states** (performance-state) characterized by a given value of voltage and clock frequency
- A p-state transition implies a CPU clock update and, hence, different cost and performance
- Reduced overheads

Autonomic self-management techniques

- Utility Based Approach: Queueing Network model + Optimization framework (e.g., IBM’s Tivoli)
  - Multiple decision variables
  - Long term time horizon (several minutes)
  - Steady state assumption

- Control Theory Approach
  - Short time frame (minutes, seconds)
  - System identification used to develop models for:
    - Capturing system transients
    - Taking into account workload variability
  - Advanced control design techniques used to:
    - Ensure closed-loop stability
    - Guarantee performance (QoS) levels *a priori*

System Identification: problem statement

- Use experimental data to construct dynamical models for performance control of Web services

- Single class Web server with FIFO scheduling
  - § λ_k: requests arrival rate
  - § s_k: service time, CPU time required to serve a single request
  - § R_k: response time, overall time a request stays in the system
  - § X_k: system throughput, requests service rate

- Dynamic frequency scaling modeling:

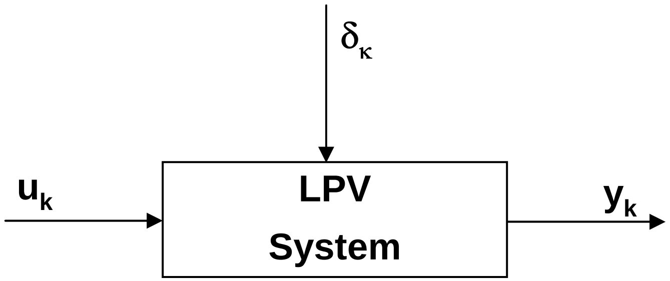
  $s_{u,k}=s_k/u_k$ effective service time

LPV state-space models

- Linear Parameter Varying systems are a class of time-varying systems

- In discrete-time state space form:

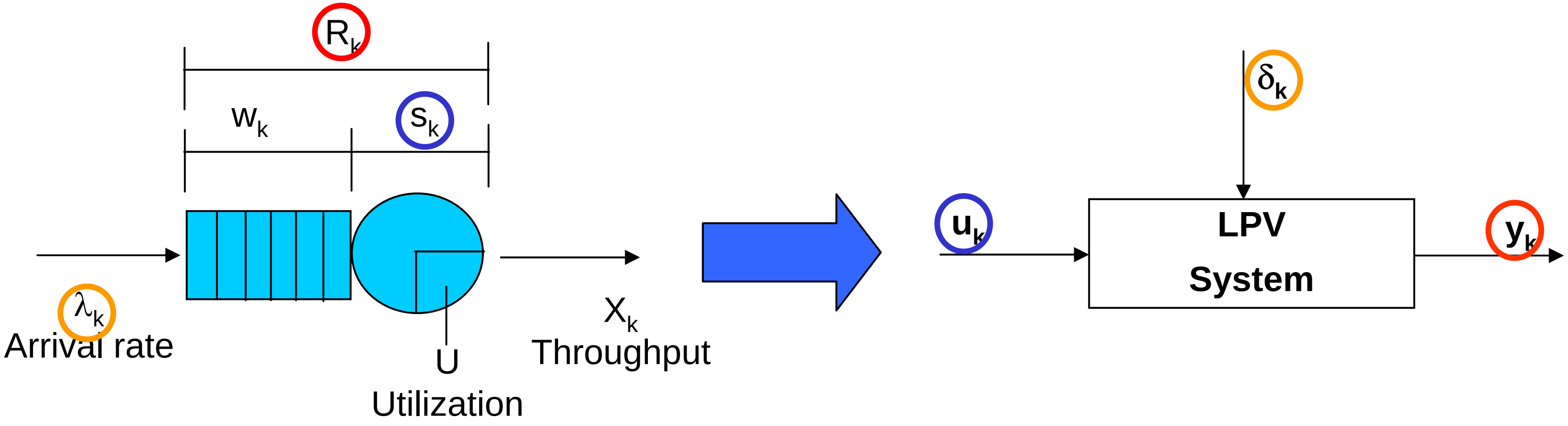
$$\begin{aligned} x_{k+1} &= A(\delta_k)x_k + B(\delta_k)u_k \\ y_k &= C(\delta_k)x_k + D(\delta_k)u_k \end{aligned}$$



- “Time varying systems, the dynamics of which are functions of a measurable, time varying parameter vector δ.”

- Models for LTV systems or linearizations of non linear systems along the trajectory of δ , gain scheduling control problems

LPV state-space models for web server dynamics



We use models with:

- Affine parameter dependence (LPV-A), that is

$$A(\delta_k) = A_0 + A_1\delta_{1,k} + \ldots + A_s\delta_{s,k}$$

and similarly for the B, C and D matrices

- Input-Affine (LPV-IA) parameter dependence, i.e., only the B and D matrices are parametrically varying

- The problem is set up as in the classical output error minimization framework

- The system is described by a set of parameters θ, identification is perfomed minimising the cost function

$$V_N(\theta) := \sum_{k=1}^{N} ||y_k - \widehat{y}_k(\theta)||_2^2$$

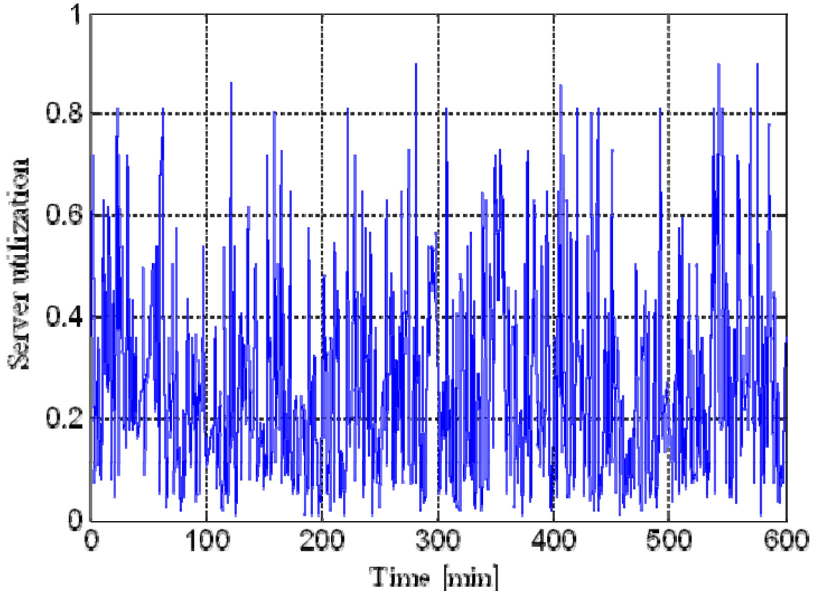
with respect to θ

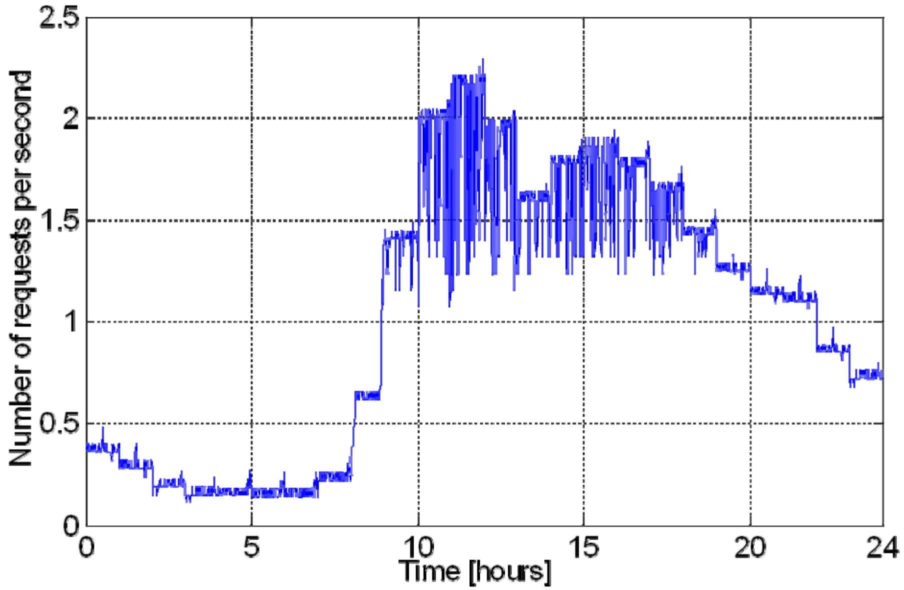
- Minimization carried out via a gradient search method (Levenberg-Marquardt algorithm)

Experimental setting

- A workload generator:
  - Apache JMeter custom extension
- Micro benchmarking web application
  - CPU service time generated according to deterministic (identification), exponential, lognormal, Pareto (validation) distributions
- Application instrumentation (otherwise, ARM API or kernel-based measurement)
- Validation: synthetic workload inspired by a real-world usage (Politecnico di Milano Web site, 24 hours)

Workload and performance metrics





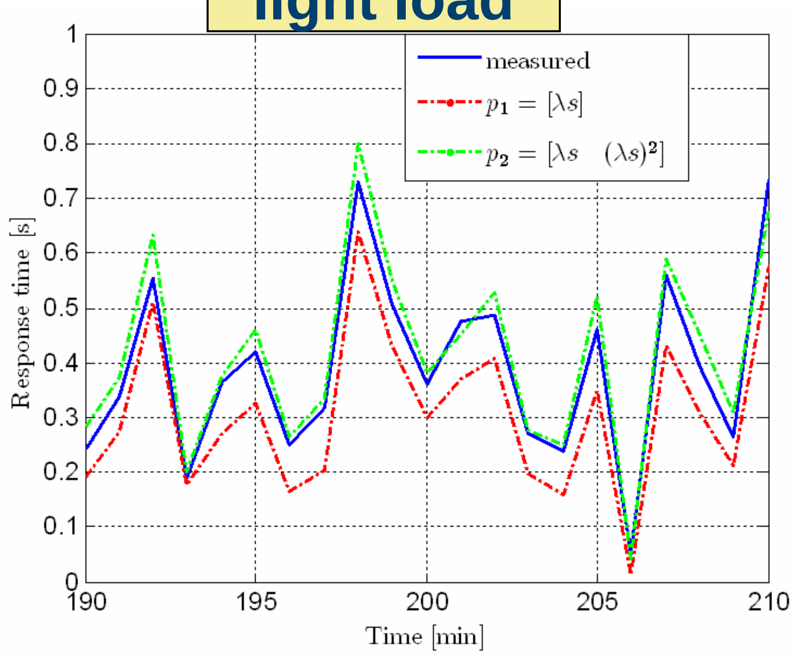
Performance metrics:

- Variance accounted for (VAF) $$VAF = 100\left(1 - \frac{Var[y_k - y_{sim,k}]}{Var[y(k)]}\right)$$

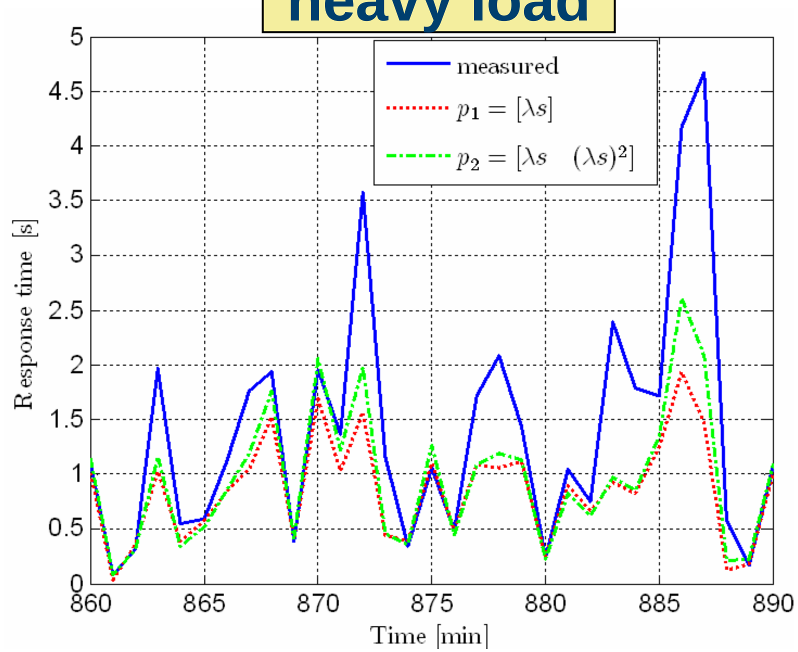
- Average simulation error (e_{avg}) $$e_{avg} = 100\left(\frac{E\left[|y_k - y_{sim,k}|\right]}{E\left[|y_k|\right]}\right)$$

Results: validation data, sampling time 1 min

light load



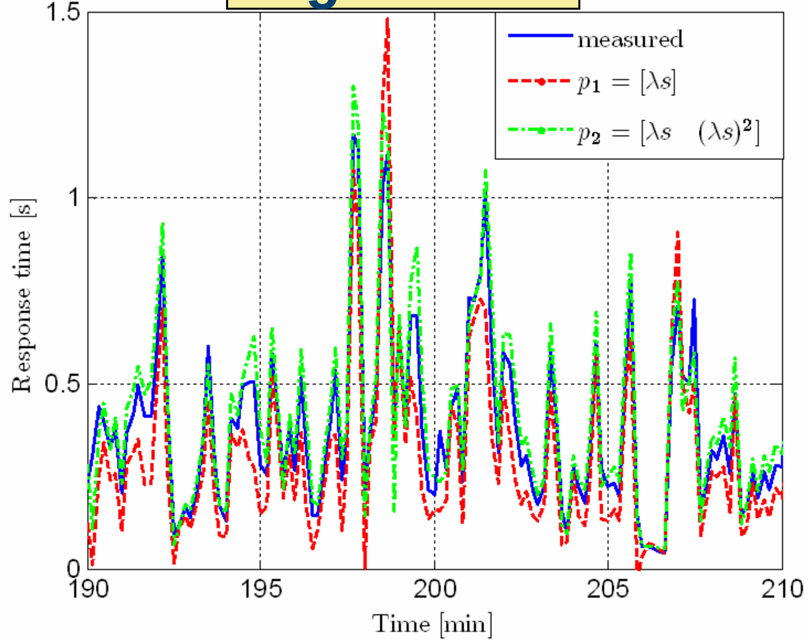
heavy load



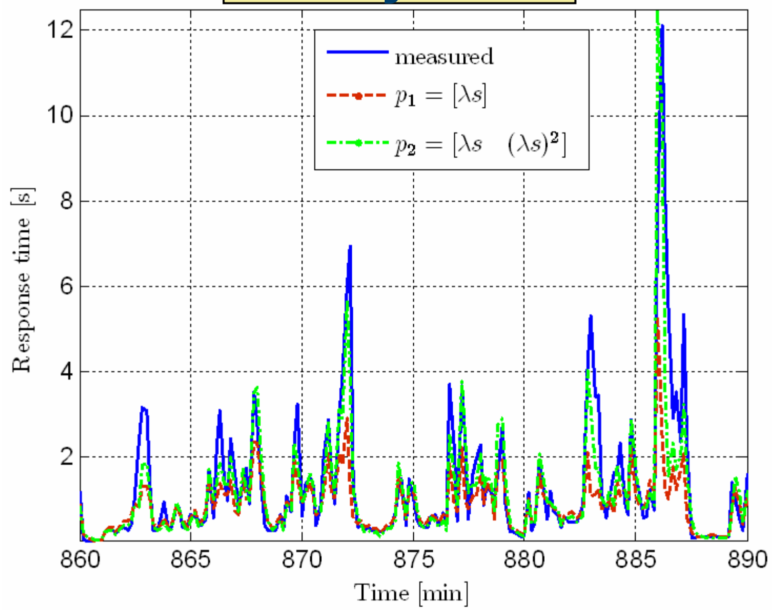
Valid. Performance $\Delta t = 1$ min	LPV-IA(p_1)	LPV-IA(p_2)
VAF on 24h	58.14%	65.18%
VAF light load (1-8)h	91.7%	89.8%
VAF heavy load (9-20)h	52.4%	60.1%
e_{avg} on 24h	25.7%	19.37%
e_{avg} light load (1-8)h	20.9%	10.5%
e_{avg} heavy load (9-20)h	29.2%	24.17%

Results: validation data, sampling time 10s

light load



heavy load



Valid. Performance $\Delta t = 10\,\mathrm{s}$	LPV-IA(p_1)	LPV-IA(p_2)
VAF on 24h	54.01%	71.5%
VAF light load (1-8)h	78.6%	80.2%
VAF heavy load (9-20)h	48.5%	67.1%
e_{avg} on 24h	20.3%	7.4%
e_{avg} light load (1-8)h	20.02%	2.5%
e_{avg} heavy load (9-20)h	22.5%	9.25%

Conclusions and future work

- Framework for modelling Web Services application performance at very fine grained time scales and in transient conditions
- Validation with application benchmarks
- Control design for single-class systems
- Extension to multi-class virtualized environments (MIMO systems)