POLITECNICO DI MILANO





A Game Theoretic Formulation of the Service Provisioning Problem in Cloud Systems

Danilo Ardagna[1], Barbara Panicucci[1], Mauro Passacantando[2]

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

[2]Università di Pisa, Dipartimento di Matematica Applicata, Italy

Basque Center for Applied Mathematics, Bilbao, 27th May 2011

Agenda

- Cloud computing preliminaries
- Game theoretical formulation for the service provisioning problem
- Solution methods
- Experimental results
- Conclusions and future work

What is Cloud Computing?

- A coherent, **large-scale**, **publicly accessible** collection of compute, storage, and networking **resources**

- Available via Web service calls **through the Internet**
- Short- or long-term access **on a pay-per-use basis**

Motivation for Cloud Computing

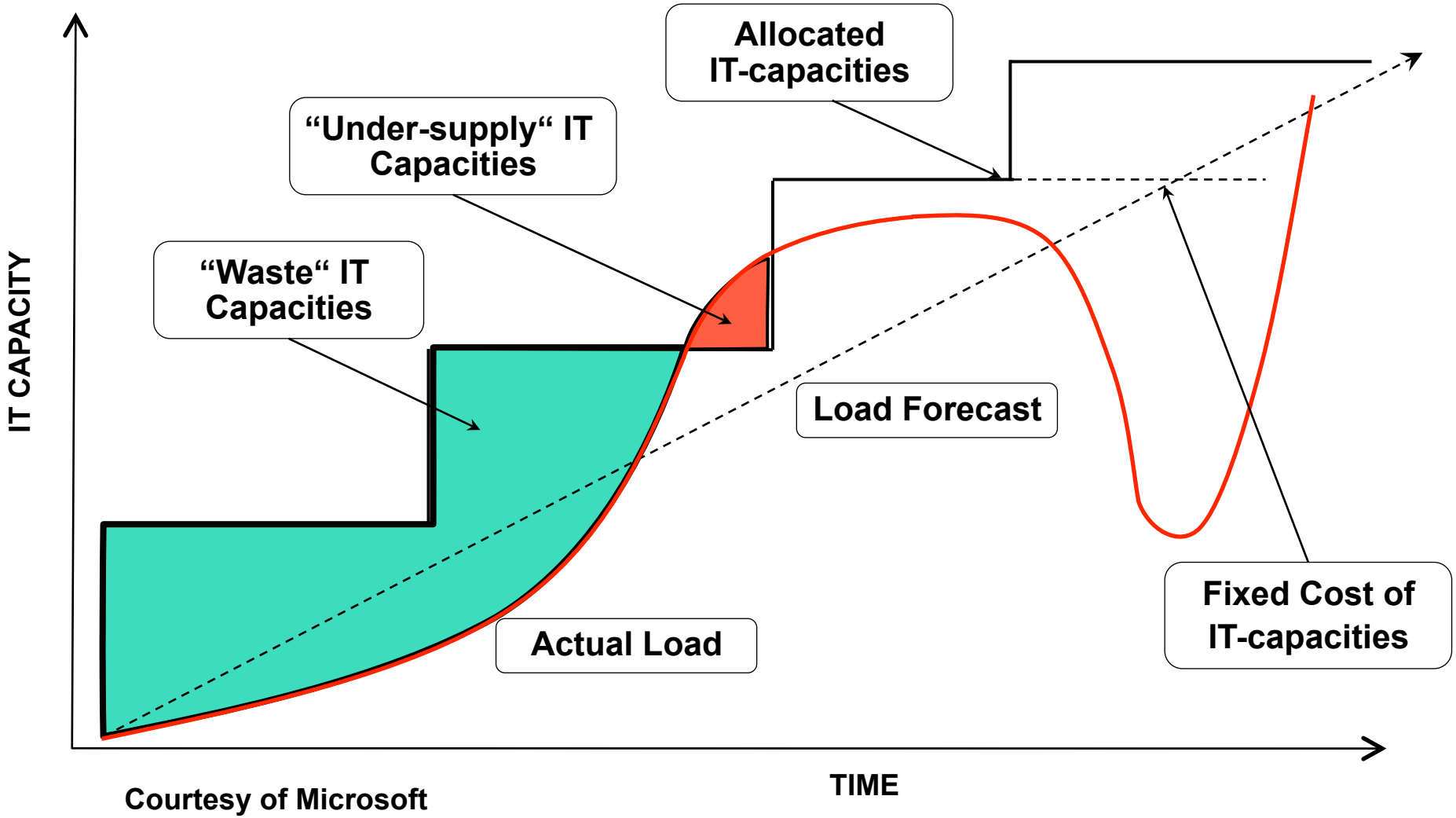
- Internet-based service **over-provisioning**
- Application deployment is **non-trivial**
- Global **financial crisis**

- Cloud computing as a cost-effective alternative to **cut down IT costs**

Cloud General Characteristics

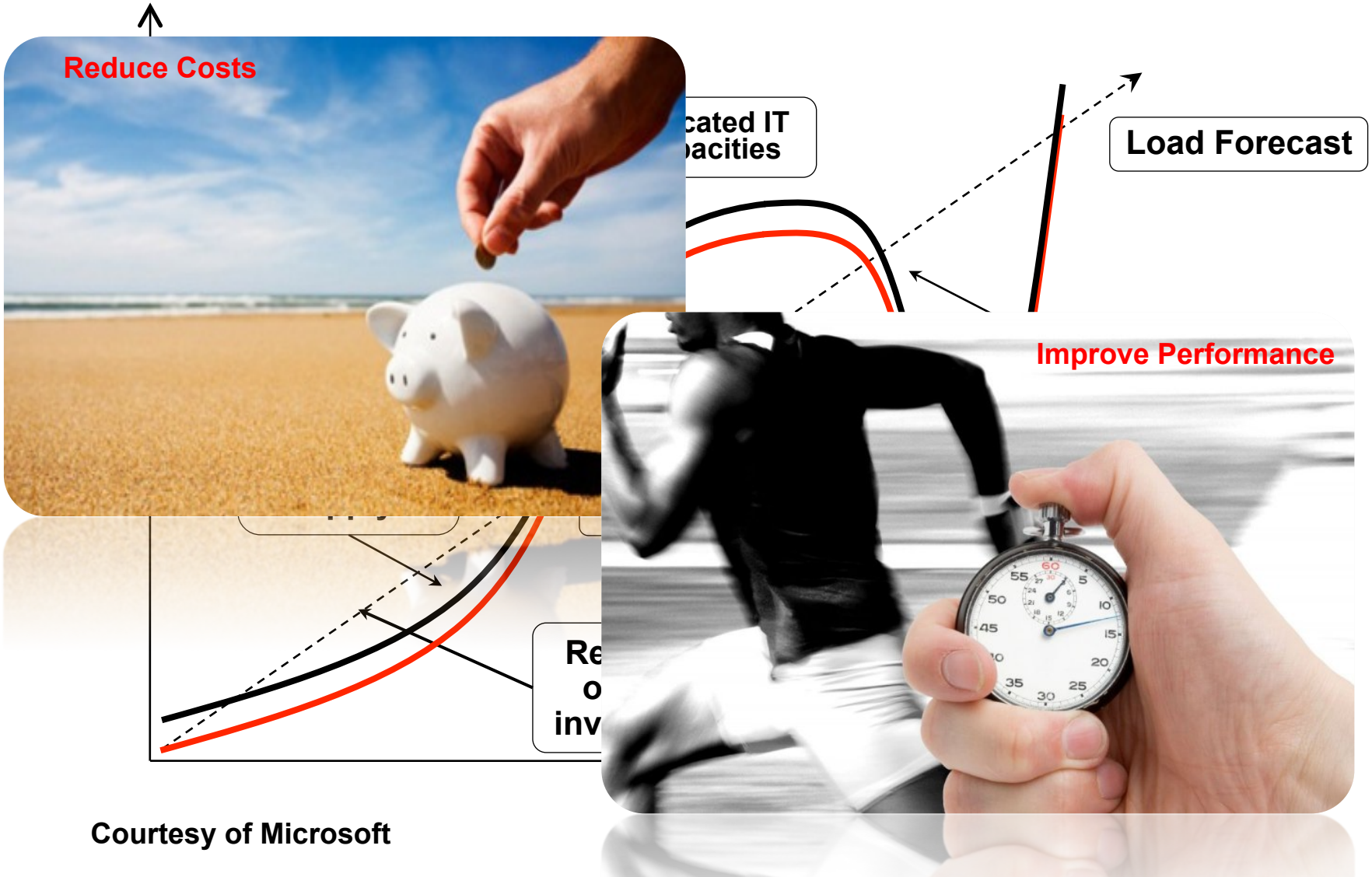
- **Scaling up and scaling down** of resource usage **as-needed**
- **Economies of scale**: The cloud provider can procure real estate, power, cooling, bandwidth, and hardware at the best possible prices
- **Pay-as-you-go**: Resource allocation decisions have an immediate effect on resource consumption and the level of overall costs

Over-provisioning – Out of Cloud



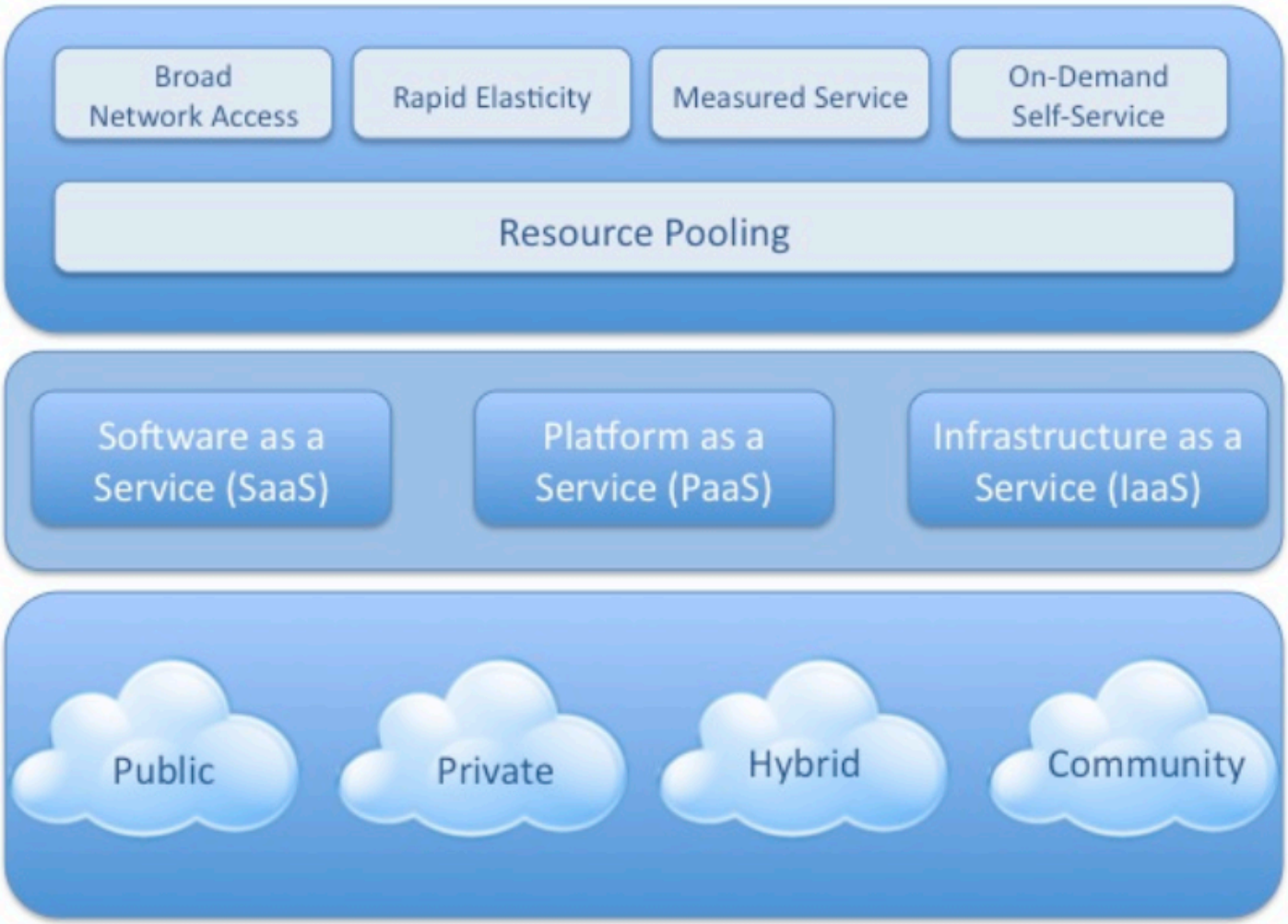
Courtesy of Microsoft

Cloud-provisioning



Courtesy of Microsoft

NIST Cloud Models



Service Models

Deployment Models

Service Models

- **Software as a Service**:
  - On demand applications over the Internet
  - Examples: Google Docs, Salesforce.com, Rackspace, and SAP Business ByDesign
- **Platform as a Service**:
  - Platform layer resources, including operating system support and software development frameworks
  - Examples: Google App Engine, Microsoft Windows Azure, and Force.com
- **Infrastructure as a Service**:
  - On-demand provisioning of infrastructural resources, usually in terms of Virtual machines (VMs)
  - Examples: Amazon EC2, GoGrid, and Flexiscale

Cloud Computing: Research issues

- Modern Clouds live in an open world characterized by **continuous changes** which occur **autonomously** and **unpredictably**

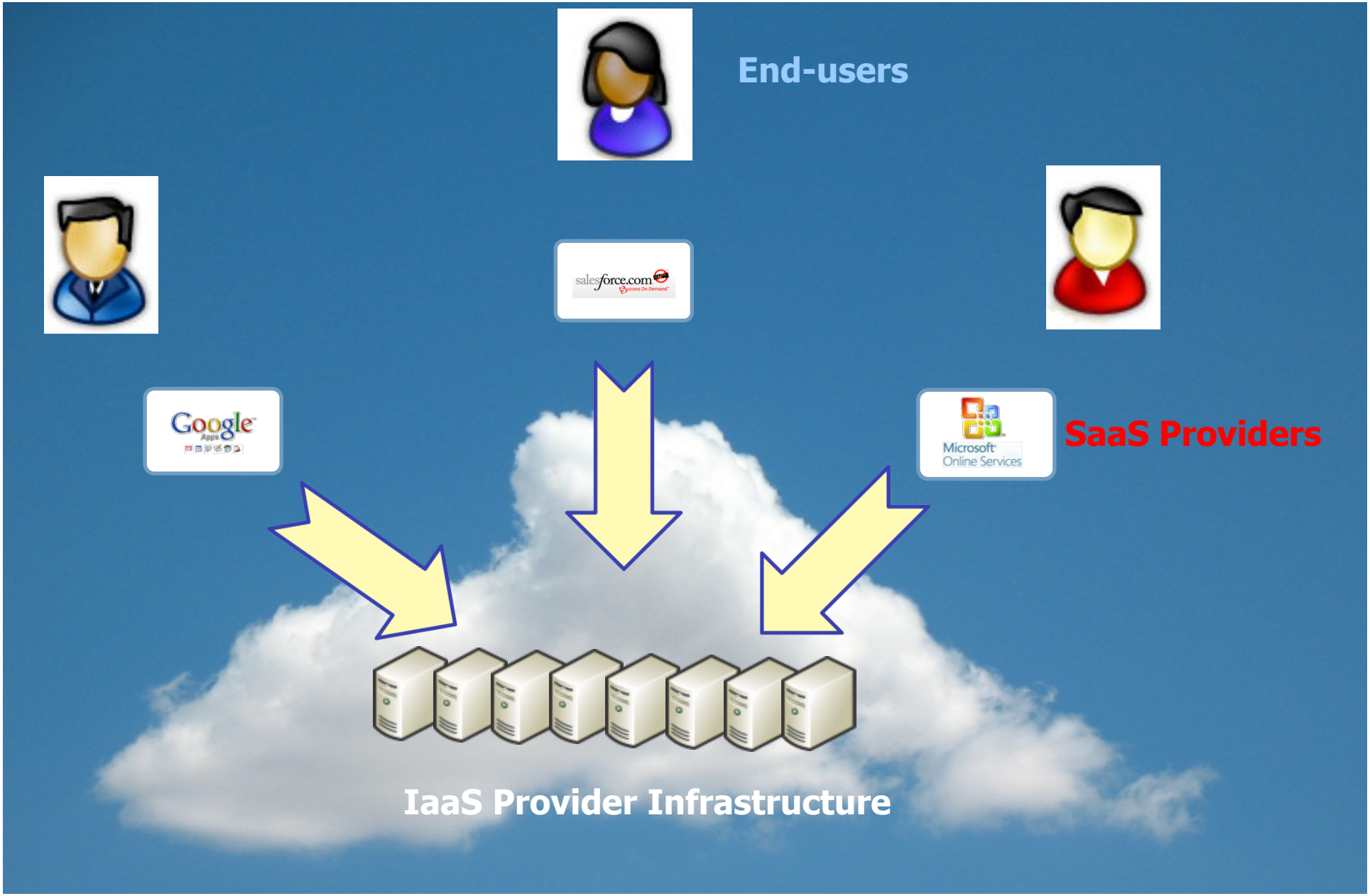
- Cloud provider **charges for used resources** even if they are idle: **Automatically scale** quickly up and down

- Development of **efficient service provisioning** policies

Our contribution

- **Game theoretic approach** to gain an in-depth analytical understanding of the service provisioning problem
- **Nash Equilibrium**: No player can benefit by **changing** his/her **strategy** while the other players **keep** their **strategies unchanged**
- **Perspective of SaaS providers** hosting their applications at an IaaS provider
- **Generalized Nash game** model and **efficient algorithm** for the **run-time allocation** of IaaS resources to competing SaaSs

Problem statement



SLA among SaaS and end-users

Design assumptions

- **Applications** are hosted **in VMs** which are dynamically instantiated by the IaaS provider
- **Each VM** hosts **a single** WS application
- Multiple **homogeneous** VMs implementing the same WS application can **run in parallel**

- IaaS provider **charges** software providers on an **hourly basis**
- Each **WS class** hosted in a VM modeled as an **M/G/1-PS queue**

IaaS pricing model

- IaaS provider **offers**:
  - *flat* VMs (ϕ time-unit cost)
  - *on demand* VMs (δ time-unit cost)
  - *on spot* VMs (σ_k time-unit cost)
- On spot **cost** σ_k periodically **fluctuates** depending on:
  - IaaS provider **time of the day** energy costs
  - **demand** from SaaS for on spot VMs
- SaaS providers **compete** for the use of on spot VMs and specify the maximum cost σ_k^U they are willing to pay
- With the current pricing models $\delta > \phi$ and we assume $\delta > \sigma_k^U$

Problem formulation – System parameters

N	number of SaaS providers
$\mathcal{A}_p$	Set of applications of the p SaaS provider
$\mathcal{A}$	Set of applications of all the SaaS providers
f_p^U	Maximum number of flat computational resources IaaS can provide for provider p
s^U	Maximum number of on spot computational resources IaaS can provide for all the SaaS providers
C	Capacity of computational resources
Λ_k	Prediction of the arrival rate for application k
μ_k	Maximum service rate of a capacity 1 server for executing class k application
m_k	Application k utility function slope
ϕ	Time unit cost for *flat* VMs
δ	Time unit cost for *on demand* VMs
σ_k^L	Minimum time unit cost for *on spot* VMs used for application k, set by the IaaS provider
σ_k^U	Maximum time unit cost for *on spot* VMs used for application k, set by the SaaS provider

Problem formulation – Decision variables

f_k	Number of *flat* VMs used for application k
d_k	Number of *on demand* VMs used for application k
s_k	Number of *on spot* VMs used for application k
σ_k	Time unit cost for *on spot* VMs used for application k
$x^p = (f_k, d_k, s_k)_{k \in \mathcal{A}_p}$	Strategies vector of SaaS provider p
$x^{-p} = (x^i)_{i=1, i \neq p}^{N}$,	Strategies vector of all SaaS providers different from p

SaaS problem

- The goal of SaaS provider *p* is to:
  - determine the **number** of flat f_k, on demand d_k, and on spot s_k VMs to be devoted for the execution of all WS applications *k*
  - **satisfy** the prediction Λ_k for the arrival rate of the WS application *k*
  - **maximize** its profits

$$E[R_k] = \frac{1}{C\mu_k - \frac{\Lambda_k}{f_k + d_k + s_k}}$$

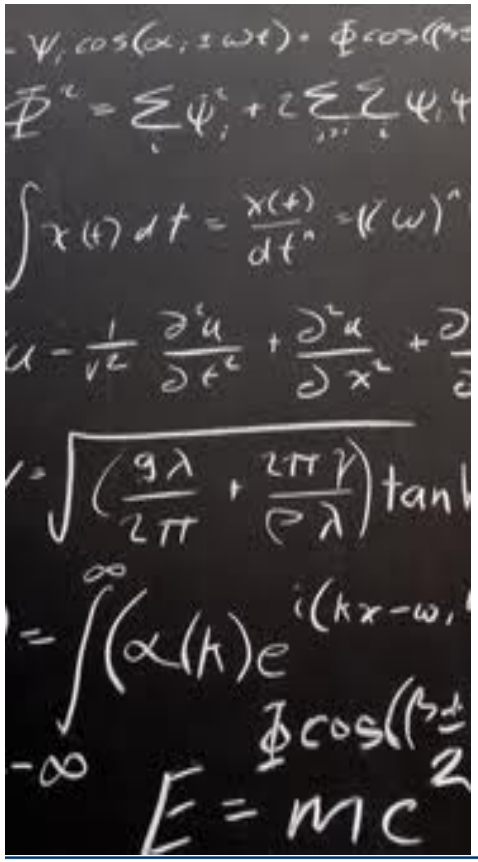
$$\mathcal{V}_k(E[R_k])\,\Lambda_k = \nu_k\,\Lambda_k \; + \frac{m_k\,\Lambda_k(f_k + d_k + s_k)}{C\,\mu_k\,(f_k + d_k + s_k) - \Lambda_k}$$

SaaS problem

$$\max \ \Theta_p = \sum_{k \in \mathcal{A}_p} \frac{m_k \, \Lambda_k (f_k + d_k + s_k)}{C \, \mu_k \, (f_k + d_k + s_k) - \Lambda_k} - \sum_{k \in \mathcal{A}_p} \phi \, f_k - \sum_{k \in \mathcal{A}_p} \delta \, d_k - \sum_{k \in \mathcal{A}_p} \sigma_k \, s_k$$

Revenues from SLA

Infrastructural costs



$$\sum_{k \in \mathcal{A}_p} f_k \leq f_p^U$$

Flat VMs ≤ the ones available

$$f_k + d_k + s_k > \frac{\Lambda_k}{C \, \mu_k} \qquad \forall \ k \in \mathcal{A}_p$$

Resources are not saturated

$$\sum_{k \in \mathcal{A}} s_k \leq s^U$$

On spot VMs ≤ the ones available

$$f_k, d_k, s_k \geq 0 \qquad \forall \ k \in \mathcal{A}_p$$

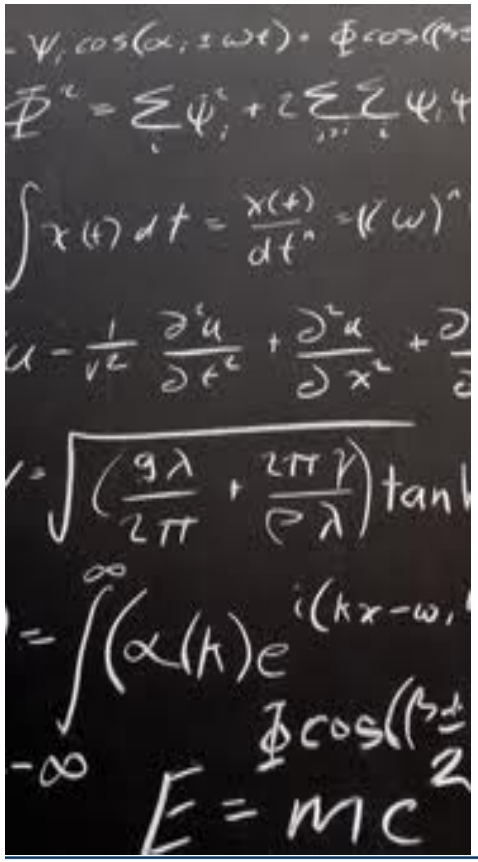
IaaS problem

$$\max \ \Theta_I = \sum_{k \in \mathcal{A}} (\phi \, f_k + \delta \, d_k + \sigma_k \, s_k)$$

Revenues from VM instances

$$\sigma_k^L \leq \sigma_k \leq \sigma_k^U \qquad \forall \ k \in \mathcal{A}$$

On spot costs within the bounds



- The on spot instance cost lower bound σ_k^L is **fixed by the IaaS** provider according to the time of the day

Problem analysis

- SaaS providers and the IaaS provider are **making decisions at the same time**, and the decisions of a SaaS depend on those of the others SaaS and the IaaS
- IaaS objective function **depends on** SaaS decisions
- Decisions **can not be analyzed in isolation**:
  - determining what a SaaS would do requires taking into account the IaaS and other SaaSs decisions
- **Generalized** Nash game

Problem analysis

- Generalized Nash Equilibrium Problem (GNEP):
  Not only the objective functions of each player depend upon the strategies chosen by all the other players**, but also each player's strategy set** may depend on the rival players' strategies
- In our setting the constraint of each problem involving other player's variables (**joint constraint**) comes from:

$$\sum_{k \in \mathcal{A}} s_k \leq s^U$$

Problem analysis

- IaaS has **a dominant strategy**: $\sigma_k = \sigma^U_k$

- The derived GNEP satisfies the **Convexity Assumption**:
  - the payoff functions of both SaaS providers and IaaS, are concave in its own variables
  - the set of strategies are convex

- SaaS decisions depend on the decisions of the other SaaSs and the IaaS, the **only constraint** involving other player's variables, **is the same** for all players

This special class of GNEP is known as jointly convex GNEP

Existence of equilibria

Theorem 1. *There exists at least one generalized Nash equilibrium for the game*

Proof.

- Based on:
  - Equivalence between generalized Nash equilibria and fixed points of the best-response mapping
  - Kakutani's fixed point theorem

Note that in general multiple GNE exist

A single application case study

- **No upper bound s^U** on the number of on spot VMs, the join constraint is relaxed

- Each player's strategy belong to a set which is fixed and does not depend on the rival player' strategies:
  - The **GNEP reduces to a NEP** which is much more simple to solve

- An **analytic study** can be obtained writing down the **KKT conditions** for the SaaS and the IaaS optimization problems and concatenating them

A single application case study

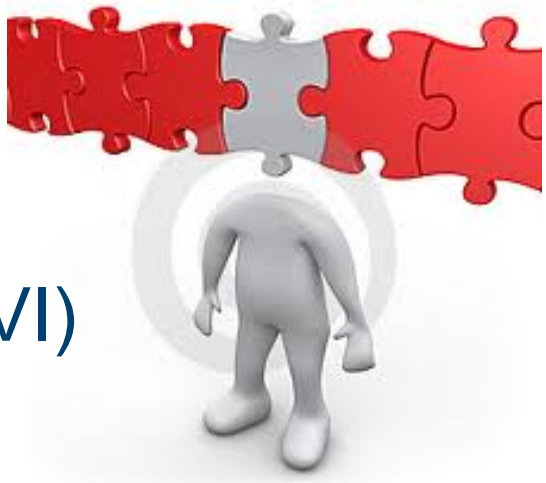
Conditions	SaaS equilibrium and value	IaaS equilibrium and value
$\phi > \sigma^U$	$f = 0 \quad d = 0 \quad s = \frac{\Lambda}{C\,\mu}\left(1 + \sqrt{\frac{-m}{\sigma^U}}\right)$ $\Theta_S = -\frac{\Lambda}{C\,\mu}(\sqrt{\sigma^U} + \sqrt{-m})^2$	$\sigma = \sigma^U$ $\Theta_I = \frac{\Lambda}{C\,\mu}(\sqrt{-m\,\sigma^U} + \sigma^U)$
$\phi = \sigma^U$	$0 \le f \le f^U \quad d = 0 \quad s \ge 0$ $f + s = \frac{\Lambda}{C\,\mu}\left(1 + \sqrt{\frac{-m}{\sigma^U}}\right)$ $\Theta_S = -\frac{\Lambda}{C\,\mu}(\sqrt{\sigma^U} + \sqrt{-m})^2$	$\sigma = \sigma^U$ $\Theta_I = \frac{\Lambda}{C\,\mu}(\sqrt{-m\,\sigma^U} + \sigma^U)$
$\phi < \sigma^U$ $f^U \le \frac{\Lambda}{C\,\mu}$	$f = f^U \quad d = 0 \quad s = \frac{\Lambda}{C\,\mu}\left(1 + \sqrt{\frac{-m}{\sigma^U}}\right) - f^U$ $\Theta_S = -\frac{\Lambda}{C\,\mu}(\sqrt{\sigma^U} + \sqrt{-m})^2 + f^U\,(\sigma^U - \phi)$	$\sigma = \sigma^U$ $\Theta_I = \frac{\Lambda}{C\,\mu}(\sqrt{-m\sigma^U} + \sigma^U) + f^U\,(\phi - \sigma^U)$

In general, there is no a unique Nash equilibrium, however, if multiple equilibria exist they are equivalent

A single application case study

Conditions	SaaS equilibrium and value	IaaS equilibrium and value
$\phi < \sigma^U$ $f^U > \frac{\Lambda}{C\,\mu}$ $\sigma^U < \frac{-m\,\Lambda^2}{(C\,\mu\,f^U-\Lambda)^2}$	$f = f^U \quad d = 0 \quad s = \frac{\Lambda}{C\,\mu}\left(1+\sqrt{\frac{-m}{\sigma^U}}\right) - f^U$ $\Theta_S = -\frac{\Lambda}{C\,\mu}(\sqrt{\sigma^U}+\sqrt{-m})^2 + f^U\,(\sigma^U-\phi)$	$\sigma = \sigma^U$ $\Theta_I = \frac{\Lambda}{C\,\mu}(\sqrt{-m\sigma^U}+\sigma^U) + f^U$
$\phi < \sigma^U$ $f^U > \frac{\Lambda}{C\,\mu}$ $\frac{-m\,\Lambda^2}{(C\,\mu\,f^U-\Lambda)^2} < \phi$	$f = \frac{\Lambda}{C\,\mu}\left(1+\sqrt{\frac{-m}{\phi}}\right) \quad d = 0 \quad s = 0$ $\Theta_S = -\frac{\Lambda}{C\,\mu}(\sqrt{\phi}+\sqrt{-m})^2$	$\max\{\sigma^L,\phi\} \le \sigma \le \sigma^U$ $\Theta_I = \frac{\Lambda}{C\,\mu}(\sqrt{-m\,\phi}+\phi)$
$\phi < \sigma^U$ $f^U > \frac{\Lambda}{C\,\mu}$ $\phi \le \frac{-m\,\Lambda^2}{(C\,\mu\,f^U-\Lambda)^2} \le \sigma^U$	$f = f^U \quad d = 0 \quad s = 0$ $\Theta_S = \frac{m\,\Lambda\,f^U}{C\,\mu\,f^U-\Lambda} - \phi\,f^U$	$\max\{\sigma^L, \frac{-m\,\Lambda^2}{(C\,\mu\,f^U-\Lambda)^2}\} \le \sigma \le \sigma$ $\Theta_I = \phi\,f^U$

Variational inequality reformulation



- A jointly convex **GNEP** can be **solved** introducing a **variational inequality** (VI)

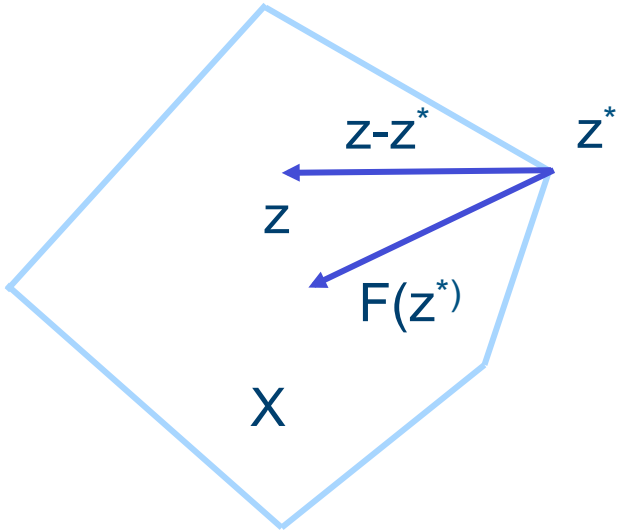
- VI: Given $X \subseteq R^n$ and $F: R^n \rightarrow R^n$, the VI(X,F) problem consists in finding a vector $z^* \in X$:

$$\langle F(z^*), z - z^* \rangle \geq 0, \text{for all } z \in X$$

VI basics

- In geometric terms: The **angle** between $F(z^*)$ and **any feasible direction** is **≤ 90°**

- This formulation is particularly convenient: A **unified framework** for **equilibrium** and **optimization** problems



VI basics

- Let z* be a solution of:

$$(P) \qquad \min f(z)$$
$$z \in X$$

- where $f \in C^1(X)$, and X is closed and convex. Then z* **is also a solution** of the VI$(X, \nabla f)$

VI basics

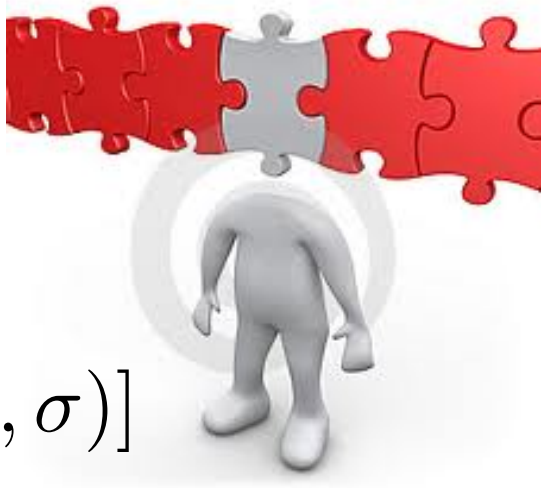
- Let be z* a solution of:

$$(P) \qquad \min f(z) \quad z \in X$$

- where $f \in C^1(X)$, and X is closed and convex. Then z* **is also a solution** of the VI$(X, \nabla f)$

- If f(z) is a **convex function** and z* is a solution of the VI$(X, \nabla f)$, then z^* **is also a solution** of (P)

General solution method



- Given a jointly convex GNEP, then every solution of the VI*(X,F)*, where:

$$F = -[(\nabla_{x_p}\Theta_p(x,\sigma)_{p=1}^{N}, \nabla_{\sigma}\Theta_I(x,\sigma)]$$
$$X := X_1 \times \ldots X_N \times X_I$$

is also a solution of the jointly convex GNEP

General solution method

- GNEP has usually multiple or even infinitely many solutions and **it is not true** that any solution of the jointly convex GNEP is also a solution of the VI

- A solution of the jointly convex GNEP which is also a solution of the VI is called a **variational equilibrium**

- Among all the equilibria, calculate a **variational equilibrium** which is **more socially stable**

General solution method

$$F = -\begin{bmatrix} \dfrac{\partial \Theta_1}{\partial f_1} \\ \dfrac{\partial \Theta_1}{\partial d_1} \\ \dfrac{\partial \Theta_1}{\partial s_1} \\ \cdots \\ \cdots \\ \dfrac{\partial \Theta_I}{\partial \sigma_1} \\ \cdots \end{bmatrix} = \begin{bmatrix} \dfrac{m_1 \Lambda_1^2}{(C \mu_1 (f_1 + d_1 + s_1) - \Lambda_1)^2} + \phi \\ \dfrac{m_1 \Lambda_1^2}{(C \mu_1 (f_1 + d_1 + s_1) - \Lambda_1)^2} + \delta \\ \dfrac{m_1 \Lambda_1^2}{(C \mu_1 (f_1 + d_1 + s_1) - \Lambda_1)^2} + \sigma_1 \\ \cdots \\ \cdots \\ -s_1 \\ \cdots \end{bmatrix}$$

General solution method

$$JF = \left[\begin{array}{ccc|c} \begin{bmatrix} a_1 & a_1 & a_1 \\ a_1 & a_1 & a_1 \\ a_1 & a_1 & a_1 \end{bmatrix} & & 0 & \\ & \ddots & & B \\ 0 & & \begin{bmatrix} a_{|\mathcal{A}|} & a_{|\mathcal{A}|} & a_{|\mathcal{A}|} \\ a_{|\mathcal{A}|} & a_{|\mathcal{A}|} & a_{|\mathcal{A}|} \\ a_{|\mathcal{A}|} & a_{|\mathcal{A}|} & a_{|\mathcal{A}|} \end{bmatrix} & \\ \hline & -B^T & & 0 \end{array}\right]$$

$$a_k = -\frac{2\, m_k\, \Lambda_k^2\, C\, \mu_k}{(C\, \mu_k\, (f_k + d_k + s_k) - \Lambda_k)^3} \; > \; 0$$

General solution method

$$\begin{bmatrix} \begin{bmatrix} a_1 & a_1 & a_1 \\ a_1 & a_1 & a_1 \\ a_1 & a_1 & a_1 \end{bmatrix} & & 0 & \\ & \ddots & & 0 \\ 0 & & \begin{bmatrix} a_{|\mathcal{A}|} & a_{|\mathcal{A}|} & a_{|\mathcal{A}|} \\ a_{|\mathcal{A}|} & a_{|\mathcal{A}|} & a_{|\mathcal{A}|} \\ a_{|\mathcal{A}|} & a_{|\mathcal{A}|} & a_{|\mathcal{A}|} \end{bmatrix} & \\ 0 & & 0 & 0 \end{bmatrix}$$

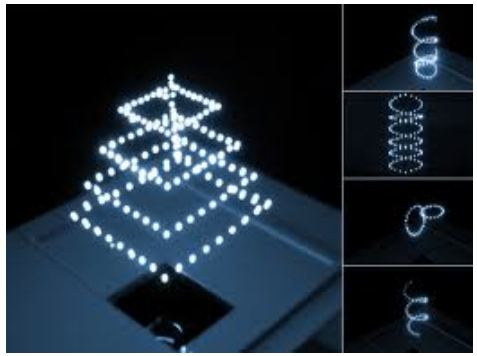
In the feasible set *X*, the symmetric part non-zero eigenvalues (i.e., $3a_1,\ldots, 3a_{|A|}$) are positive, ***F*** results to be **monotone** (not strictly):

$$\langle F(w) - F(z), w - z \rangle \geq 0, \text{ for any } w,z \in X$$

General solution method

- The **VI** associated with our GNEP is **monotone**
- We implemented the **hyperplane projection** method by Iusem and Svaiter 1997
- **Monotonicity** guarantees that the hyperplane **method converges**

Hyperplane projection method

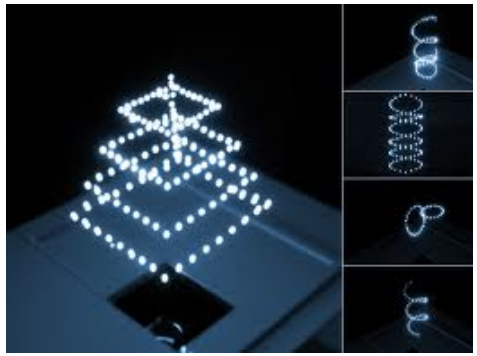


- If X is closed and convex, then $z^* \in X$ **is a solution** of the variational inequality problem VI(F,X) **iff** for any $\gamma>0$:

$$z^* = P_X(z^* - \gamma F(z^*))$$

- Iterative method performing **two projections per iteration**

Hyperplane projection method

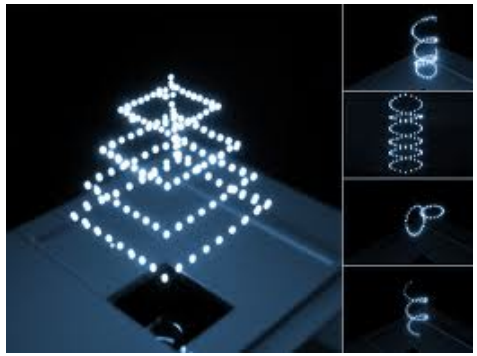


- Let $\boldsymbol{z^t}$, be the current approximation to the VI(F, X) solution
- First, compute the point $\boldsymbol{P_X\,[z^t - F\,(z^t)]}$
- Next, **search** the line segment between z^t and $P_X\,[z^t - F\,(z^t)]$ for a point $\boldsymbol{w^t}$ such that the hyperplane

$$\partial H^t := \{z \in R^n \mid \langle F\,(w^t),\, z - w^t \rangle = 0\}$$

strictly separates z^t from any solution z^* of the problem

Hyperplane projection method



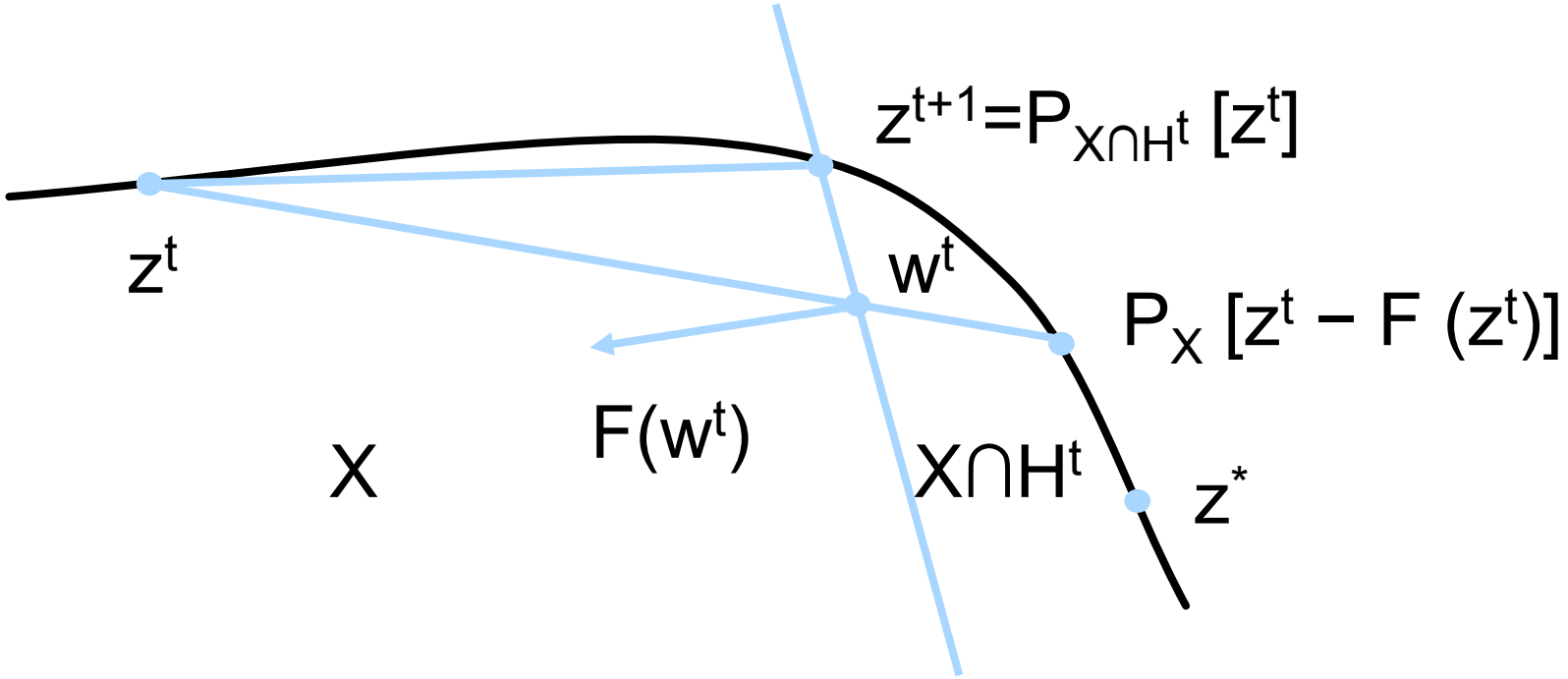
- w^t found by a computationally inexpensive **Armijo-type** procedure
- The **next iterate z^{t+1}** is computed by projecting z^t onto the **intersection** of the feasible set X with:

$$H^t := \{x \in R^n \mid \langle F(w^t), z - w^t \rangle \leq 0\}$$

- At each iteration:
  - One projection **onto the set X** (to construct the separating hyperplane H^t)
  - One projection **onto the intersection $X \cap H^t$**

Hyperplane projection method

$$\partial H^t := \{z \in R^n \mid \langle F(w^t), z - w^t \rangle = 0\}$$

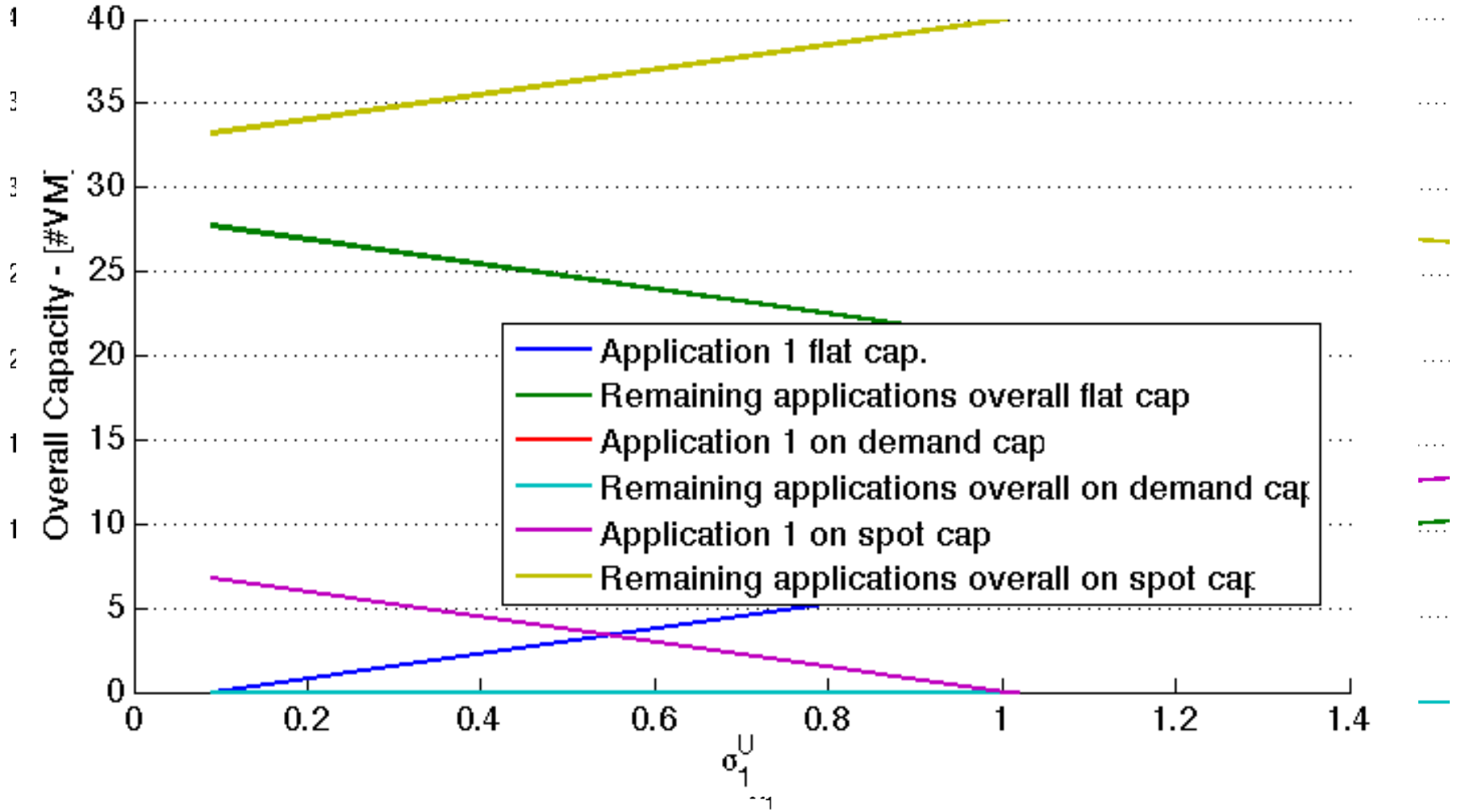


Experimental results – Scalability analysis

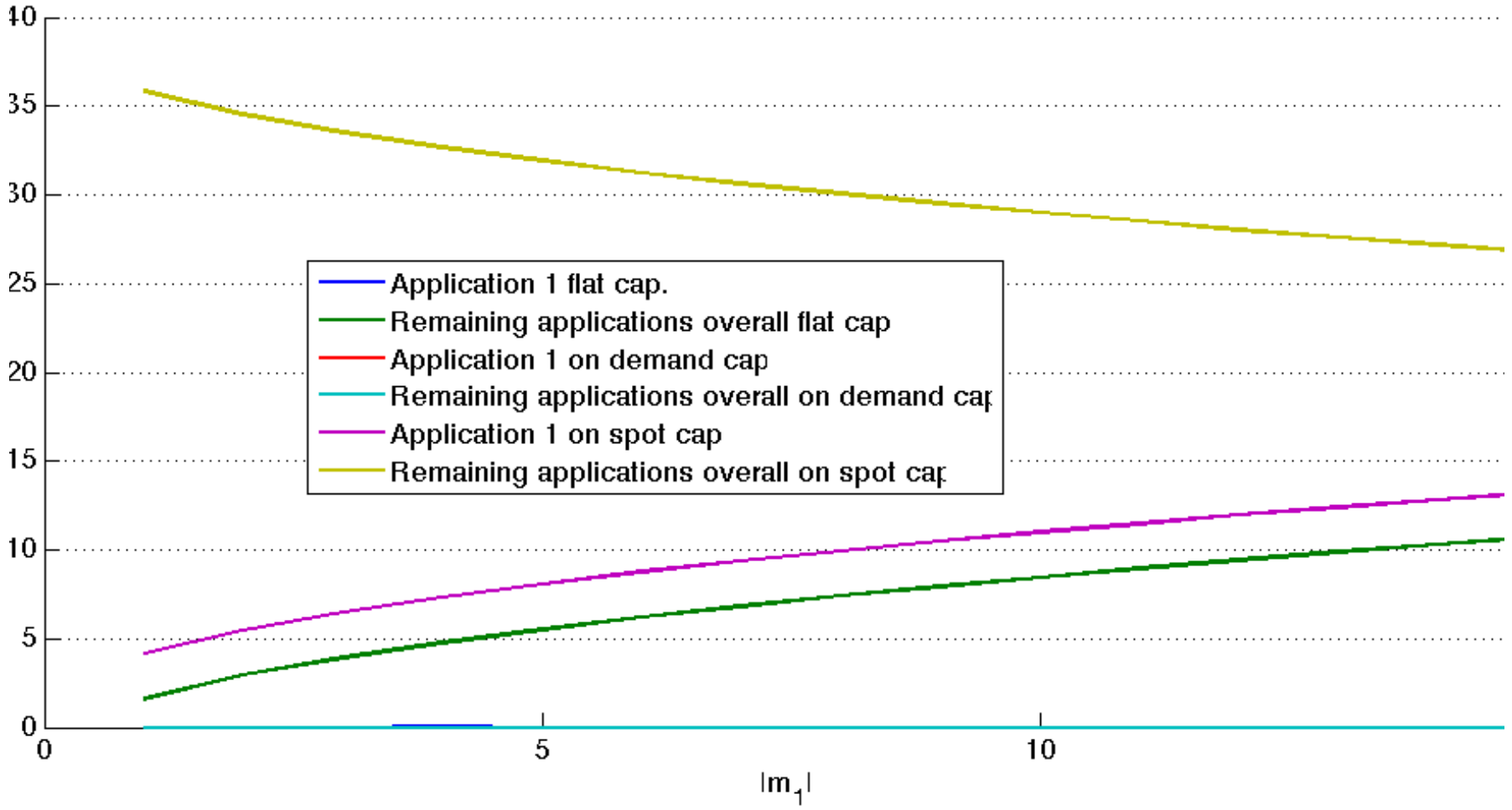
$N, \lvert\mathcal{A}\rvert$	Exe. Time (s)	$N, \lvert\mathcal{A}\rvert$	Exe. Time (s)
10,1000	18.9	60,6000	125.7
20,2000	133.8	70,7000	259.2
30,3000	299.4	80,8000	298.5
40,4000	115.5	90,9000	186.4
50,5000	116.4	100,10000	258.0

- All tests have been performed on a **VMWare VM running** on an Intel Nehalem dual socket quad-core system with 32 GB of RAM
- The VM has **a physical core** dedicated with guaranteed performance and 4 GB of memory reserved
- **KNITRO 7.0** has been used as nonlinear optimization solver

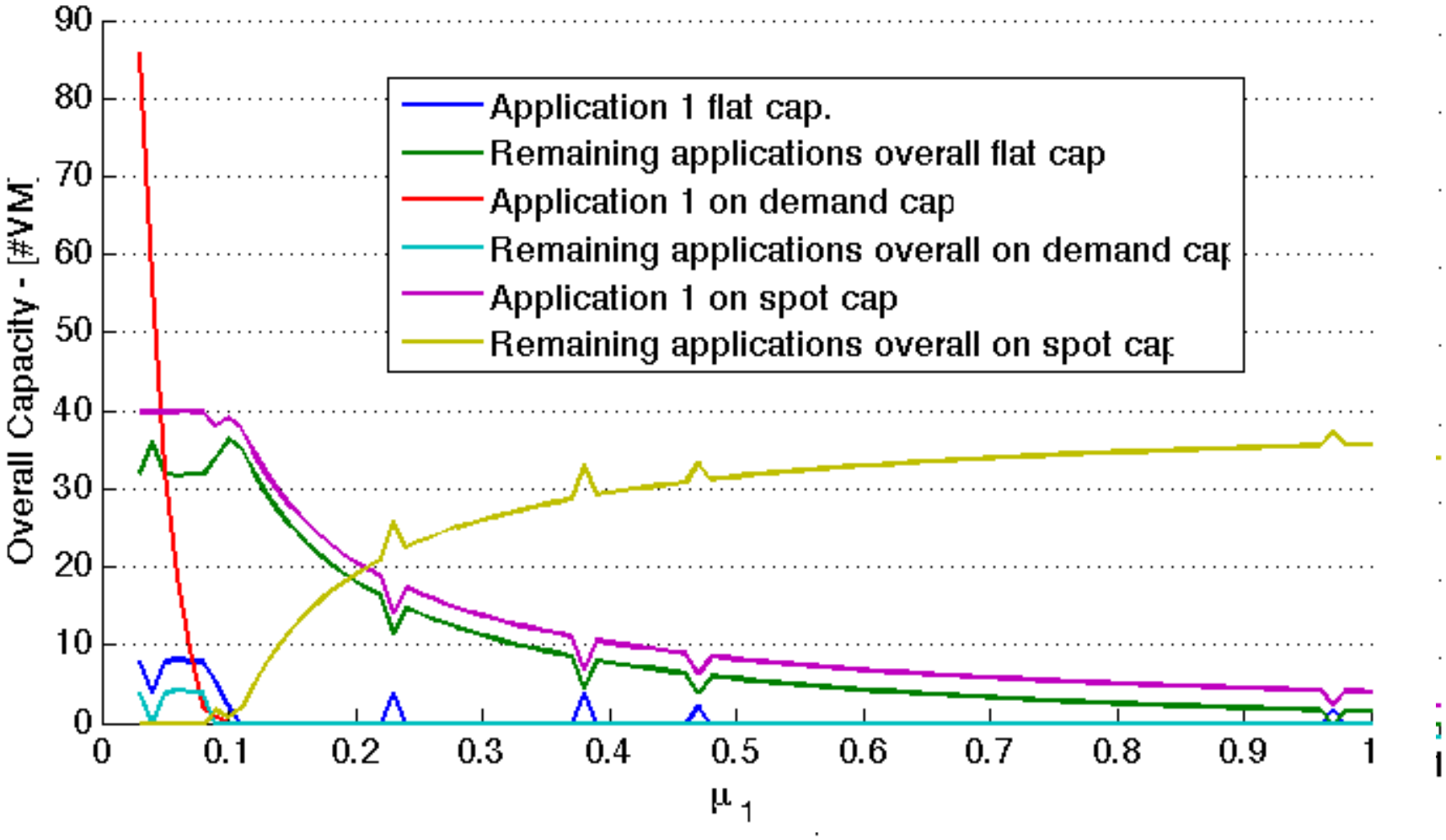
Experimental results – Equilibria sharing analysis



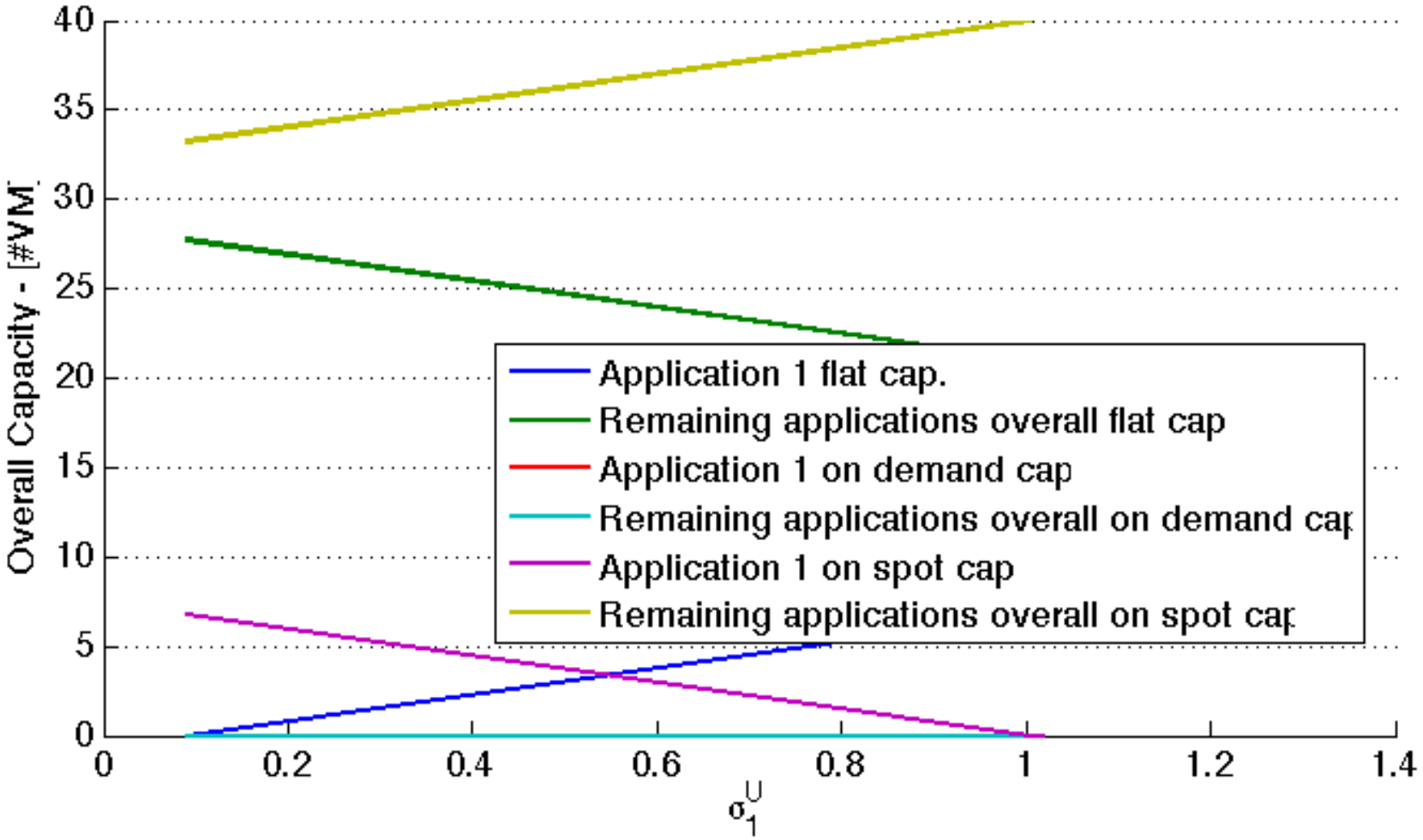
Experimental results – Equilibria sharing analysis



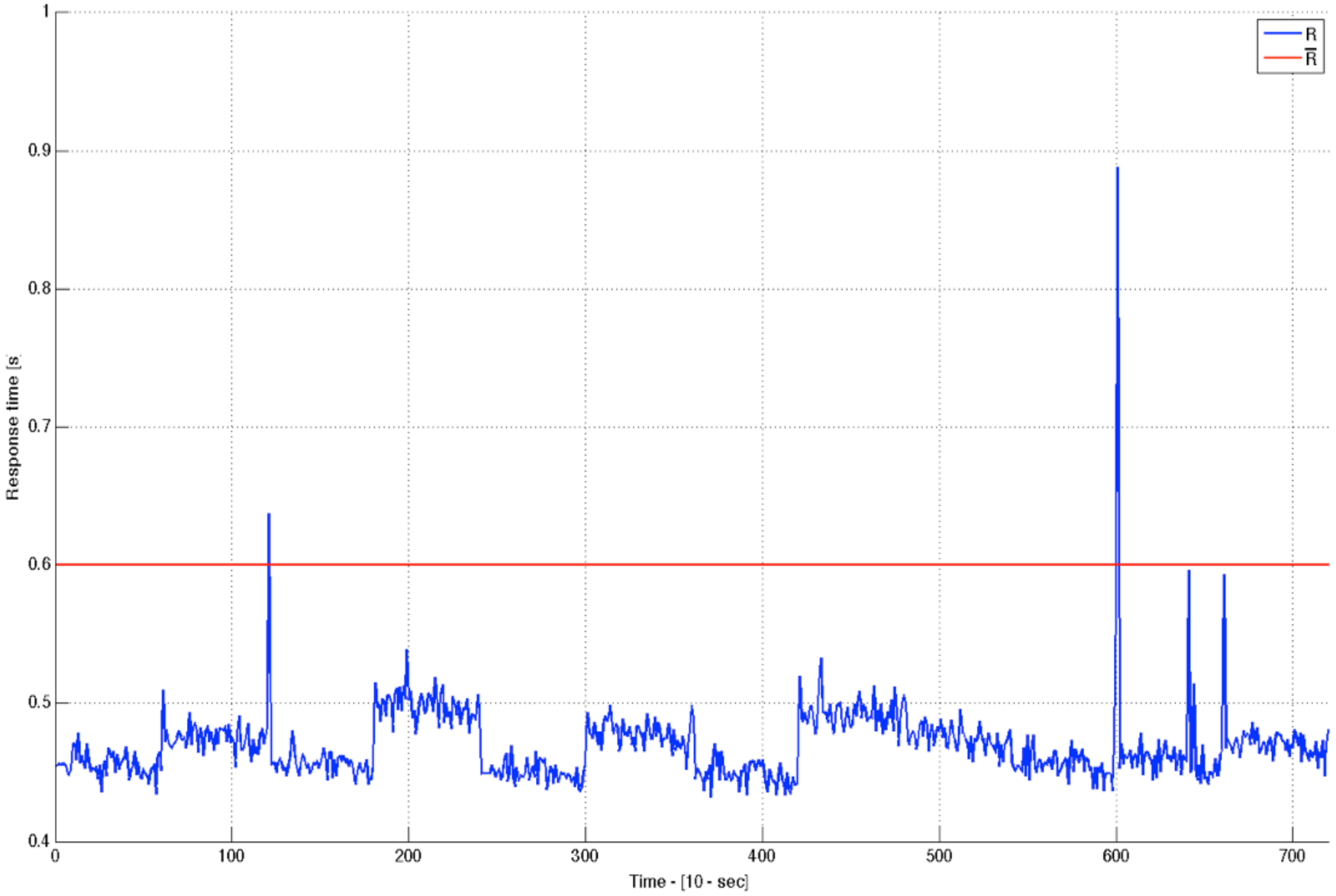
Experimental results – Equilibria sharing analysis



Experimental results – Equilibria sharing analysis



Ongoing work – Validation on Amazon EC2



Conclusions and future work

- **Game theory based approach** for the run time management of a IaaS provider capacity among multiple competing SaaSs
- The model includes **infrastructural costs and revenues** depending on the achieved level of performance of individual requests
- The model will be extended to include **hybrid clouds**
- Comparison with the **heuristics adopted in practice** by SaaS and IaaS providers


Thank you!
Questions?

References

- D. Ardagna, B. Panicucci, M. Passacantando **A Game Theoretic Formulation of the Service Provisioning Problem in Cloud Systems**. *WWW 2011 Proceedings*. 177-186. Hyderabad, India.