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

Danilo Ardagna¹, Barbara Panicucci¹, Mauro Passacantando²

¹Politecnico di Milano, Dipartimento di Elettronica e Informazione, Italy
²Università 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

- Actual Load
- Allocated IT-capacities
- "Waste" IT Capacities
- "Under-supply" IT Capacities

Load Forecast

Fixed Cost of IT-capacities

Courtesy of Microsoft
Cloud-provisioning

- Reduce Costs
- Reduce initial investments
- No "under-supply"
- Possible reduction of IT-capacities in case of reduced load
- Reduce Costs
- Improve Performance

Load Forecast

Actual Load

Load Forecast

Courtesy of Microsoft
NIST Cloud Models

Service Models
- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

Deployment Models
- Public
- Private
- Hybrid
- Community

Characteristics:
- Broad Network Access
- Rapid Elasticity
- Measured Service
- On-Demand Self-Service

Resource Pooling
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

IaaS Provider Infrastructure

End-users

SaaS Providers

IaaS Provider Infrastructure
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 ($\phi$ time-unit cost)
  - on demand VMs ($\delta$ time-unit cost)
  - on spot VMs ($\sigma_k$ time-unit cost)

- On spot cost $\sigma_k$ 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 $\sigma_k^U$ they are willing to pay

- With the current pricing models $\delta > \phi$ and we assume $\delta > \sigma_k^U$
### Problem formulation – System parameters

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>number of SaaS providers</td>
</tr>
<tr>
<td>$\mathcal{A}_p$</td>
<td>Set of applications of the $p$ SaaS provider</td>
</tr>
<tr>
<td>$\mathcal{A}$</td>
<td>Set of applications of all the SaaS providers</td>
</tr>
<tr>
<td>$f^U_U$</td>
<td>Maximum number of flat computational resources IaaS can provide for provider $p$</td>
</tr>
<tr>
<td>$s^U_U$</td>
<td>Maximum number of on spot computational resources IaaS can provide for all the SaaS providers</td>
</tr>
<tr>
<td>$C$</td>
<td>Capacity of computational resources</td>
</tr>
<tr>
<td>$\Lambda_k$</td>
<td>Prediction of the arrival rate for application $k$</td>
</tr>
<tr>
<td>$\mu_k$</td>
<td>Maximum service rate of a capacity 1 server for executing class $k$ application</td>
</tr>
<tr>
<td>$m_k$</td>
<td>Application $k$ utility function slope</td>
</tr>
<tr>
<td>$\phi$</td>
<td>Time unit cost for flat VMs</td>
</tr>
<tr>
<td>$\delta$</td>
<td>Time unit cost for on demand VMs</td>
</tr>
<tr>
<td>$\sigma^L_k$</td>
<td>Minimum time unit cost for on spot VMs used for application $k$, set by the IaaS provider</td>
</tr>
<tr>
<td>$\sigma^U_k$</td>
<td>Maximum time unit cost for on spot VMs used for application $k$, set by the SaaS provider</td>
</tr>
</tbody>
</table>
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 $k$
- $\sigma_k$: Time unit cost for on spot VMs used for application $k$
- $x^p = (f_k, d_k, s_k)_{k \in A_p}$: Strategies vector of SaaS provider $p$
- $x^{-p} = (x^i)_{i=1, i \neq p}$: 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 $\Lambda_k$ for the arrival rate of the WS application $k$
  - maximize its profits

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

$$\nu_k(E[R_k]) \Lambda_k = \nu_k \Lambda_k + \frac{m_k \Lambda_k (f_k + d_k + s_k)}{C \mu_k (f_k + d_k + s_k) - \Lambda_k}$$
SaaS problem

\[
\max \ \Theta_p = \sum_{k \in A_p} \frac{m_k \Lambda_k(f_k + d_k + s_k)}{C \mu_k (f_k + d_k + s_k) - \Lambda_k} - \sum_{k \in A_p} \phi f_k - \sum_{k \in A_p} \delta d_k - \sum_{k \in A_p} \sigma_k s_k
\]
IaaS problem

\[ \max \Theta_I = \sum_{k \in A} (\phi f_k + \delta d_k + \sigma_k s_k) \]

\[ \sigma_k^L \leq \sigma_k \leq \sigma_k^U \quad \forall k \in A \]

- The on spot instance cost lower bound \( \sigma_k^L \) 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 cannot 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:

\[
\sum_{k \in A} s_k \leq s^U
\]
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
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

<table>
<thead>
<tr>
<th>Conditions</th>
<th>SaaS equilibrium and value</th>
<th>IaaS equilibrium and value</th>
</tr>
</thead>
</table>
| $\phi > \sigma^U$ | $f = 0 \quad d = 0 \quad s = \frac{\Lambda}{C_\mu} \left(1 + \sqrt{-m \sigma^U}\right)$ | $\sigma = \sigma^U$
\[\Theta_S = -\frac{\Lambda}{C_\mu} (\sqrt{\sigma^U} + \sqrt{-m})^2\] | $\Theta_I = \frac{\Lambda}{C_\mu} (\sqrt{-m \sigma^U} + \sigma^U)$ |
| $\phi = \sigma^U$ | $0 \leq f \leq f^U \quad d = 0 \quad s \geq 0$
\[f + s = \frac{\Lambda}{C_\mu} \left(1 + \sqrt{-m \sigma^U}\right)\]
\[\Theta_S = -\frac{\Lambda}{C_\mu} (\sqrt{\sigma^U} + \sqrt{-m})^2\] | $\sigma = \sigma^U$
\[\Theta_I = \frac{\Lambda}{C_\mu} (\sqrt{-m \sigma^U} + \sigma^U)\]
| $\phi < \sigma^U$ | $f = f^U \quad d = 0 \quad s = \frac{\Lambda}{C_\mu} \left(1 + \sqrt{-m \sigma^U}\right) - f^U$
\[\Theta_S = -\frac{\Lambda}{C_\mu} (\sqrt{\sigma^U} + \sqrt{-m})^2 + f^U (\sigma^U - \phi)\] | $\sigma = \sigma^U$
\[\Theta_I = \frac{\Lambda}{C_\mu} (\sqrt{-m \sigma^U} + \sigma^U) + f^U (\phi - \sigma^U)\]

In general, there is no a unique Nash equilibrium, however, if multiple equilibria exist they are equivalent.
## A single application case study

<table>
<thead>
<tr>
<th>Conditions</th>
<th>SaaS equilibrium and value</th>
<th>IaaS equilibrium and value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi &lt; \sigma^U$</td>
<td>$f = f^U \quad d = 0 \quad s = \frac{\Lambda}{C\mu} \left(1 + \sqrt{-m/\sigma^U}\right) - f^U$</td>
<td>$\sigma = \sigma^U$</td>
</tr>
<tr>
<td>$f^U &gt; \frac{\Lambda}{C\mu}$</td>
<td>$\Theta_S = -\frac{\Lambda}{C\mu} (\sqrt{\sigma^U} + \sqrt{-m})^2 + f^U (\sigma^U - \phi)$</td>
<td>$\Theta_I = \frac{\Lambda}{C\mu} (\sqrt{-m\sigma^U} + \sigma^U) + f^U$</td>
</tr>
<tr>
<td>$\sigma^U &lt; \frac{-m\Lambda^2}{(C\mu f^U - \Lambda)^2}$</td>
<td>$f = \frac{\Lambda}{C\mu} \left(1 + \sqrt{-m/\phi}\right) \quad d = 0 \quad s = 0$</td>
<td>max{\sigma^L, \phi} $\leq \sigma \leq \sigma^U$</td>
</tr>
<tr>
<td>$\Theta_S = -\frac{\Lambda}{C\mu} (\sqrt{\phi} + \sqrt{-m})^2$</td>
<td>$\Theta_I = \frac{\Lambda}{C\mu} (\sqrt{-m\phi} + \phi)$</td>
<td></td>
</tr>
<tr>
<td>$\phi &lt; \sigma^U$</td>
<td>$f = f^U \quad d = 0 \quad s = 0$</td>
<td>max{\sigma^L, \frac{-m\Lambda^2}{(C\mu f^U - \Lambda)^2}} $\leq \sigma \leq \sigma^U$</td>
</tr>
<tr>
<td>$f^U &gt; \frac{\Lambda}{C\mu}$</td>
<td>$\Theta_S = \frac{m\Lambda f^U}{C\mu f^U - \Lambda} - \phi f^U$</td>
<td>$\Theta_I = \phi f^U$</td>
</tr>
<tr>
<td>$\phi \leq \frac{-m\Lambda^2}{(C\mu f^U - \Lambda)^2} \leq \sigma^U$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Variational inequality reformulation

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

• VI: Given $X \subseteq \mathbb{R}^n$ and $F: \mathbb{R}^n \rightarrow \mathbb{R}^n$, 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 $\leq 90^\circ$

• This formulation is particularly convenient: A unified framework for equilibrium and optimization problems
• Let $z^*$ be a solution of:

$\begin{align*}
(P) \quad \min & \ f(z) \\
\text{s.t. } & \ z \in X
\end{align*}$

• 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:

\[
\begin{align*}
(P) & \quad \min f(z) \\
& \quad z \in X
\end{align*}
\]

• 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 \textbf{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:

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

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

\[
F = - \begin{bmatrix}
\frac{\partial \Theta_1}{\partial f_1} \\
\frac{\partial d_1}{\partial \Theta_1} \\
\frac{\partial s_1}{\partial \Theta_1} \\
\vdots \\
\frac{\partial \Theta_I}{\partial \sigma_1} \\
\vdots
\end{bmatrix} = \begin{bmatrix}
\frac{m_1 \Lambda_1^2}{(C \mu_1 (f_1 + d_1 + s_1) - \Lambda_1)^2} + \phi \\
\frac{m_1 \Lambda_1^2}{(C \mu_1 (f_1 + d_1 + s_1) - \Lambda_1)^2} + \delta \\
\frac{m_1 \Lambda_1^2}{(C \mu_1 (f_1 + d_1 + s_1) - \Lambda_1)^2} + \sigma_1 \\
\vdots \\
-s_1 \\
\vdots
\end{bmatrix}
\]
General solution method

\[ JF = \begin{bmatrix}
    a_1 & a_1 & a_1 \\
    a_1 & a_1 & a_1 \\
    a_1 & a_1 & a_1 \\
\end{bmatrix}
\]

\[ -B^T \]

\[ a_k = -\frac{2 m_k \Lambda_k^2 C \mu_k}{(C \mu_k (f_k + d_k + s_k) - \Lambda_k)^3} > 0 \]
In the feasible set $X$, the symmetric part non-zero eigenvalues (i.e., $3a_1, \ldots, 3|A|$) 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 $z^t$, be the current approximation to the $\text{VI}(F, X)$ solution

• First, compute the point $P_X [z^t - F(z^t)]$

• Next, search the line segment between $z^t$ and $P_X [z^t - F(z^t)]$ for a point $w^t$ such that the hyperplane

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

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:

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

- At each iteration:
  - One projection onto the set \( X \) (to construct the separating hyperplane \( H^t \))
  - One projection onto the intersection \( X \cap H^t \)
\[ \partial H^t := \{ z \in \mathbb{R}^n \mid \langle F(w^t), z - w^t \rangle = 0 \} \]
## Experimental results – Scalability analysis

| $N, |\mathcal{A}|$ | Exe. Time (s) | $N, |\mathcal{A}|$ | 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

Overall Capacity - [#VM] vs. $o^U_{1}$

- Application 1 flat cap.
- Remaining applications overall flat cap
- Application 1 on demand cap
- Remaining applications overall on demand cap
- Application 1 on spot cap
- Remaining applications overall on spot cap
Experimental results – Equilibria sharing analysis
Experimental results – Equilibria sharing analysis

![Graph showing experimental results for equilibria sharing analysis]

- **Overall Capacity - [#VM]**
- **μ₁**

- **Application 1 flat cap.**
- **Remaining applications overall flat cap**
- **Application 1 on demand cap**
- **Remaining applications overall on demand cap**
- **Application 1 on spot cap**
- **Remaining applications overall on spot cap**
Experimental results – Equilibria sharing analysis

Graph showing the relationship between overall capacity and the number of VMs for different applications and capacity types.
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