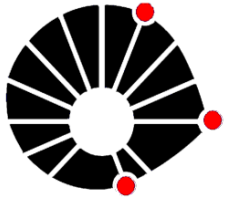
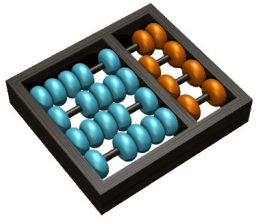
Workflow Scheduling for SaaS / PaaS Cloud Providers Considering Two SLA Levels

Thiago A. L. Genez, Luiz F. Bittencourt, Edmundo R. M. Madeira

Institute of Computing (IC)
University of Campinas (UNICAMP)

April 16, 2012





Outline

Introduction I

Cloud Computing overview:

- Provides resources as general utilities
- Resources can be leased and released on-demand
  - Virtualization layer
- *Pay-per-use* basis
- Split into three general models
  - IaaS – infrastructure as a service
  - PaaS – platform as a service
  - SaaS – software as a service

Introduction II

SaaS (or PaaS) Provider: our focus

- Provides a workflow execution service to its customers through service level agreements (SLA)
  - Must deal with seasonality of customer requests
  - Must be able to provide a good quality-of-service (QoS)
  - Must be prepared to attend a peak of demand
  - Must meet the execution deadline stipulated for each workflow
- Leases resource (VMs) from IaaS provider also through SLA contracts
  - Uses Multiple IaaS providers
  - Can lower its maintenance costs
  - Do not have to deal with peculiarities of hardware
  - Brings *elasticity* to its computational power

Introduction III

Main problems:

- **Workflow scheduling**: How to distribute the dependent services on the virtual machines leased by the SaaS provider?
  - NP-Complete problem
- **Financial issue**: How to minimize the monetary costs involved with leasing virtual machines?
- **SLA levels:** How to manage two SLA levels around the SaaS provider?
  - *First level:* SLAs between the SaaS provider and each customer
  - *Second level:* SLAs between the SaaS provider and each IaaS provider.

Introduction IV

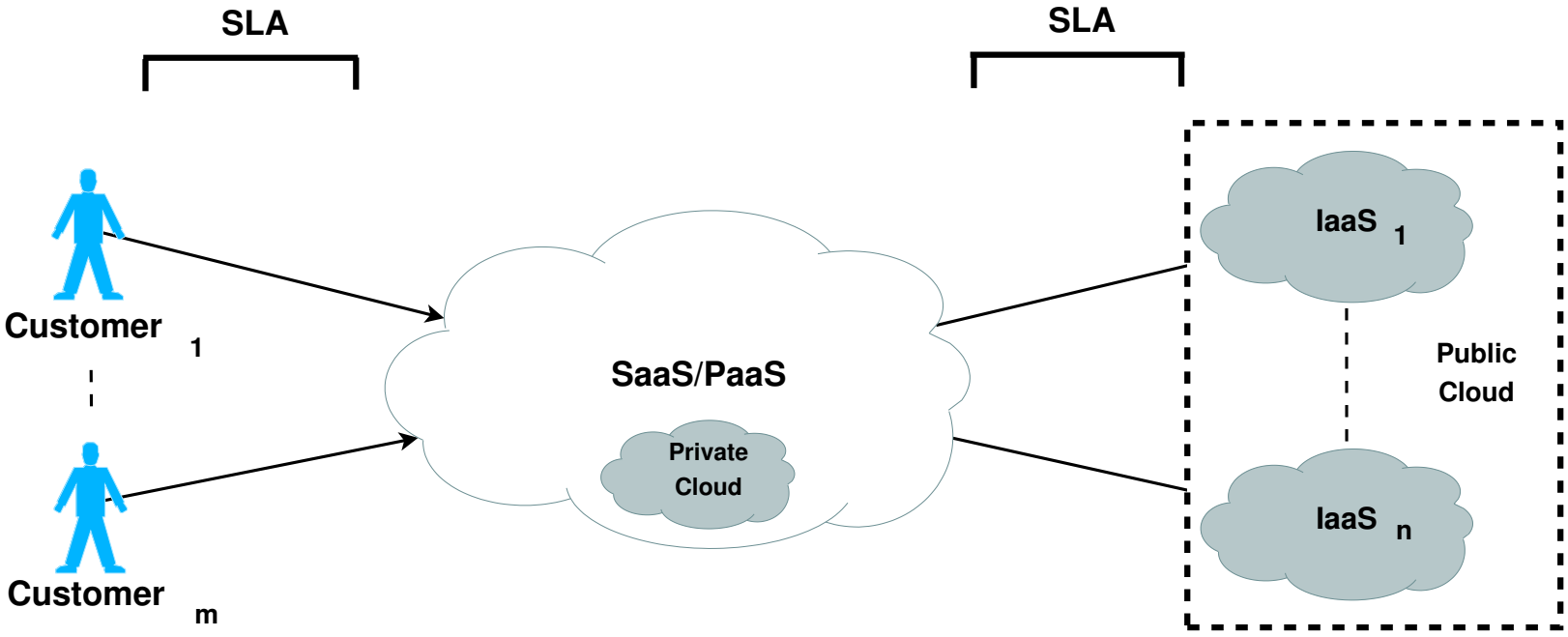
Purpose of this work: The SaaS provider goals are

- Execute the customers workflow within the deadline using VMs leased from multiple IaaS providers
- Minimize the monetary cost involved
- Indirectly, enable maximization of its profit

The Cloud Scenario I

Two SLA levels

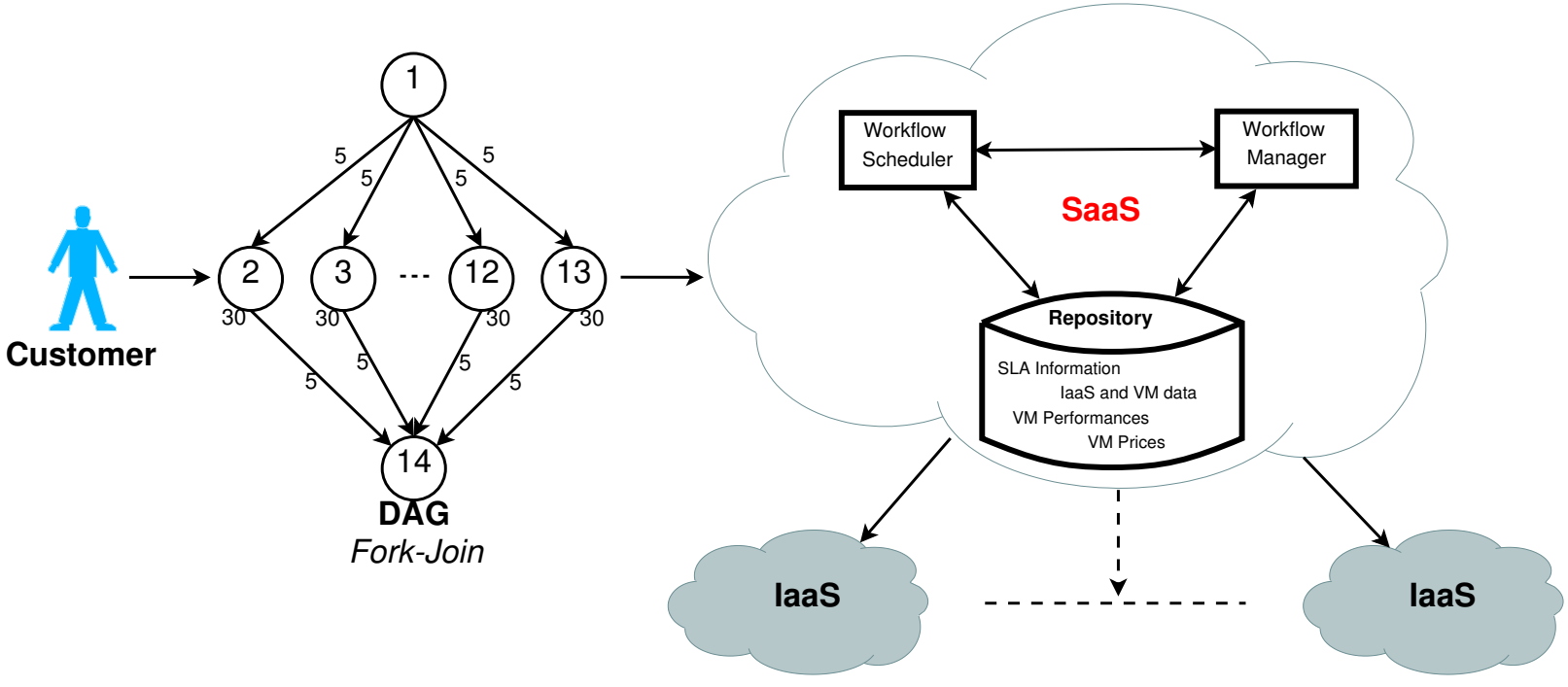
- The scenario considered for serving SaaS cloud customers:



The Cloud Scenario II

Two SLA levels

- Workflow: represented by a DAG $\mathcal{G} = \{\mathcal{U}, \mathcal{E}\}$, where:
  - each node $u_i \in \mathcal{U}$ represents a service
  - each edge $e_{i,j} \in \mathcal{E}$ is a data dependency between services i and j



Problem Modeling

The scheduling cost-optimization problem can be stated as:

"*Find a feasible mapping M between the nodes of the DAG* $\mathcal{G}$ *and the VMs from multiple IaaS providers, such that* **the sum of the monetary computation cost** *for all nodes* $u \in \mathcal{U}$ *on a VM* $v \in \mathcal{V}$ **is minimal**, *the* **dependencies** *among nodes* **are not violated**, *and* **the total execution time** *of the mapping M (makespan)* $\mathcal{M}_{\mathcal{G}}$ *is* **at most equal to the deadline** *required by user, i.e.,* $\mathcal{M}_{\mathcal{G}} \leq \mathcal{D}_{\mathcal{G}}$".

Methods used:

- Integer linear program (ILP)
- Heuristics to obtain integer solutions from the relaxed version of the proposed ILP

Formulation of the Integer Linear Program (ILP) I

Variables and constant used in the integer linear program:

- $x_{u,t,v}$: binary variable that assumes the value 1 if the node u finishes at time t in the VM v; otherwise this variable assumes the value 0;
- $y_{t,v}$: binary variable that assumes the value 1 if the VM v is being used at time t; otherwise this variable assumes the value 0;
- $\mathcal{C}_v$: constant that assumes the cost per time unit for using the VM v.

Linear objective function:

Minimize $\sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} y_{t,v} \times \mathcal{C}_v$

Formulation of the Integer Linear Program (ILP) II

Constraints:

(C1) $$\sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} x_{u,t,v} = 1; \quad \forall\ u \in \mathcal{U};$$

Specifies that a DAG node must be executed only once and in a single VM

(C2) $$\sum_{u \in \mathcal{U}} \sum_{v \in \mathcal{V}} \sum_{t=1}^{\lceil w_u \times \mathcal{J}_v \rceil} x_{u,t,v} = 0;$$

Establishes that a DAG node u can't be set as finished until it has been executed in a VM v

(C3) $$\sum_{s=1}^{t-\lceil w_z \times \mathcal{J}_r + f_{u,z} \times \mathcal{L}_{i,j} \rceil} x_{u,s,v} \geq \sum_{s=1}^{t} x_{z,s,r}$$
$$\forall\ z \in \mathcal{U},\ \forall\ u \in \mathcal{H}(z),\ \forall\ r, v \in \mathcal{V},$$
$$\forall\ t \in \mathcal{T},\ \forall\ i, j \in \mathcal{I} \mid \mathcal{B}_{i,v} = 1,\ \mathcal{B}_{j,r} = 1$$

Determines that a DAG node z cannot begin its execution until all preceding nodes have finished their processing and the resulting data has arrived at the VM that will run z

Formulation of the Integer Linear Program (ILP) III

Constraints:

$$\text{(C4)} \quad \sum_{u \in \mathcal{U}} \sum_{s=t:t \leq \mathcal{D}_\mathcal{G} - \lceil w_u \times \mathcal{J}_v \rceil}^{t+\lceil w_u \times \mathcal{J}_v \rceil - 1} x_{u,s,v} \leq \mathcal{P}_v$$
$$\forall\ v \in \mathcal{V},\ \forall\ t \in \mathcal{T}$$

Specifies that the number of DAG nodes executing on a VM v at a given time t cannot exceed the number of processing cores of v

$$\text{(C5)} \quad \sum_{s=t-\lceil w_u \times \mathcal{J}_v \rceil + 1}^{t} y_{s,v} \geq x_{u,t,v} \times \left(\lceil w_u \times \mathcal{J}_v \rceil \right)$$
$$\forall\ u \in \mathcal{U},\ \forall\ v \in \mathcal{V},\ \forall\ t \in \{\lceil w_u \times \mathcal{J}_v \rceil, \ldots, \mathcal{D}_\mathcal{G}\}$$

Determines that a VM must stay active while it is executing the node which requires it

Formulation of the Integer Linear Program (ILP) IV

Constraints:

(C6) $$\sum_{v \in \mathcal{V}} y_{t,v} \leq \delta_i \quad \forall\ i \in \mathcal{I},\ \forall\ t \in \mathcal{T} \mid \mathcal{B}_{i,v} = 1$$

Specifies that the number of reserved VMs plus the number of on-demand VMs cannot exceed the maximum number allowed by each IaaS provider

(C7) $$\sum_{v \in \mathcal{V}} y_{t,v} \leq \alpha_s \quad \forall\ s \in \zeta,\ \forall\ t \in \mathcal{T} \mid \mathcal{K}_{s,v} = 1$$

Establishes that the amount of VMs being used cannot exceed the limit stipulated in the SLA

(C8) $$x_{u,t,v} \in \{0,1\} \quad \forall\ u \in \mathcal{U},\ \forall\ t \in \mathcal{T},\ \forall\ v \in \mathcal{V}$$

(C9) $$y_{t,v} \in \{0,1\} \quad \forall\ t \in \mathcal{T},\ \forall\ v \in \mathcal{V}$$

Specifies that the variables of this ILP will only assume the binary values 0 or 1

Approaches to solve the ILP I

NP-Completeness

- The workflow scheduling problem is NP-Complete
- Time to solve the ILP increases exponentially with the input size
- We considered three different manners to cope with this:
  - Optimal approach
  - First solution approach
  - Relaxed approach

ILP Relaxation

- $\{0, 1\} \Rightarrow [0, 1]$ in constraints (C8) and (C9)
- Returns real numbers in the variable $x_{u,t,v}$
- Splits a node among virtual machines, however, each node in the DAG is indivisible

Approaches to solve the ILP II

Heuristics

- Iterative method to obtain an integer solution from a relaxed ILP
  - Tries to solve the problem $k \leq |\mathcal{U}| = n$ times
- Heuristics choose a variable $x_{u,t,v}$ to be set to 1, defining its scheduling
- Two heuristics have been developed:
  - *begin-minimum end-maximum times* (BMEMT)
    - Interchangeably gets one node from the beginning and one from the end of the DAG
    - sets the variabale $x_{u,t,v}$ such that t is minimum (or maximum if node is from the end).
  - *begin-minimum time* (BMT)
    - only ready tasks from the beginning of the DAG
    - sets the variable $x_{u,t,v}$ such that t is minimum

Approaches to solve the ILP III

General iterative algorithm

Require: DAG $\mathcal{G} = \{\mathcal{U}, \mathcal{E}\}$, deadline $\mathcal{D}_{\mathcal{G}}$, set of IaaSs $\mathcal{V}$
Ensure: Schedule of $\mathcal{G}$ in $\mathcal{V}$

1: Call relaxed ILP solver
2: **while** $\exists\ u \in \mathcal{U}$ such that $\sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} x_{u,t,v} \neq 1$ **do**
3: Choose a node $u_i \in \mathcal{U}$ according to the heuristic
4: Choose a resulting variable x_{u_i,t_a,v_x} according to the heuristic
5: Add new constraint $x_{u_i,t_a,v_x} = 1$ to the ILP
6: Call relaxed solver for the new ILP
7: **end while**
8: Return the solution from the last solver call

Evaluation I

Details of the simulation:

- Java and IBM ILOG CPLEX Optimizer
- First round:
  - BMEMT and BMT
  - Both with time limit of 1800 seconds on each solver iteration
  - Store the running times RT_{BMT} and RT_{BMEMT}
- Second round:
  - *Optimal solution* and *First solution*
  - Both with time limit equal to RT_{BMT} and RT_{BMEMT}
- The evaluated metrics are:
  - Monetary cost of the schedule
  - The workflow makespan
  - The solve time of the algorithm
  - The number of unfeasible solutions

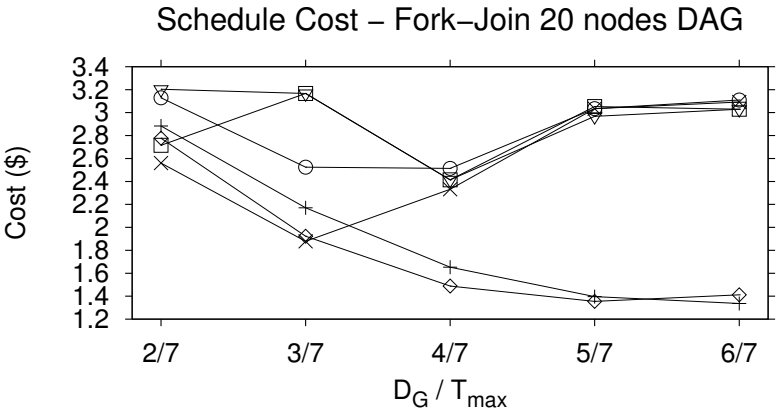
Evaluation II

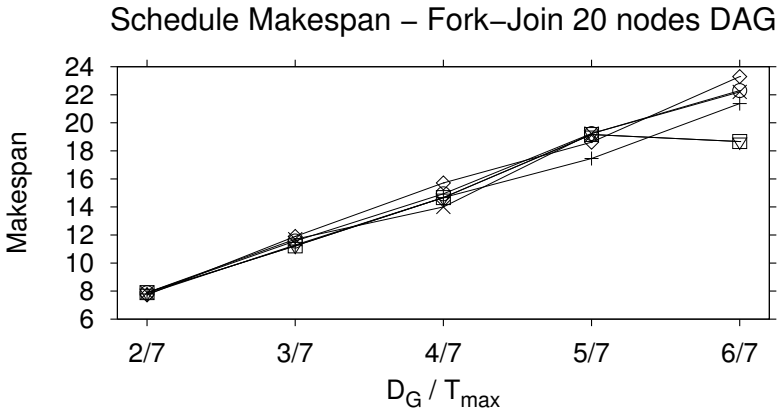
Simulation Configurations

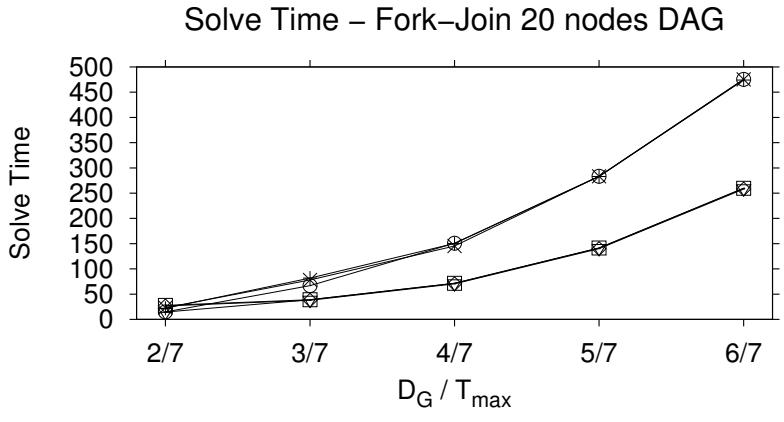
- We used 3 IaaS providers in our simulations
- Each SaaS had its own configurations and prices for reserved and on-demand VMs
- 1 to 8 cores
- Heterogeneous resources performance (processors and links)

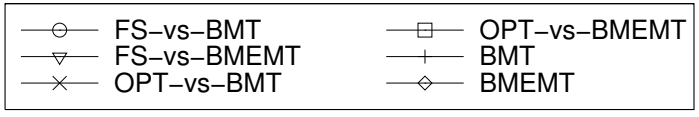
Evaluation III

- Results for the Fork-Join DAG with 20 nodes



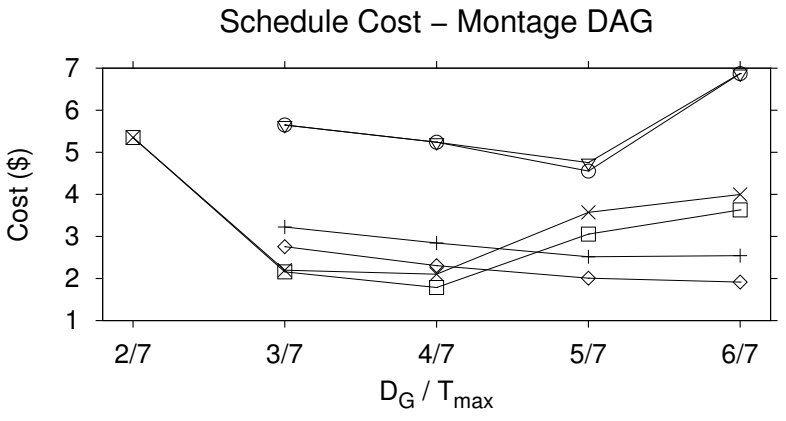


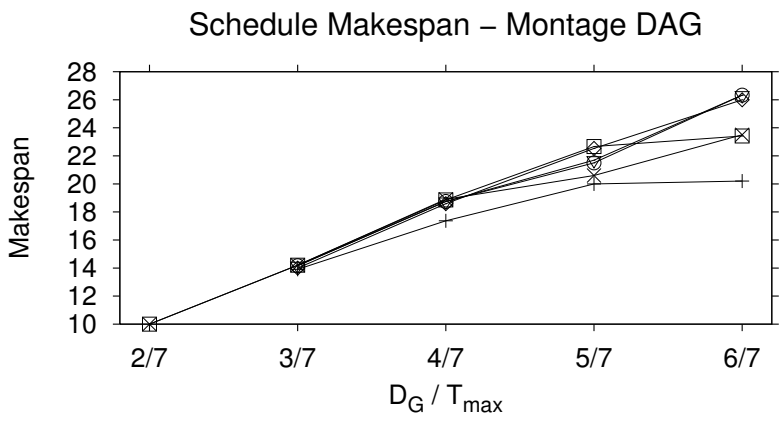


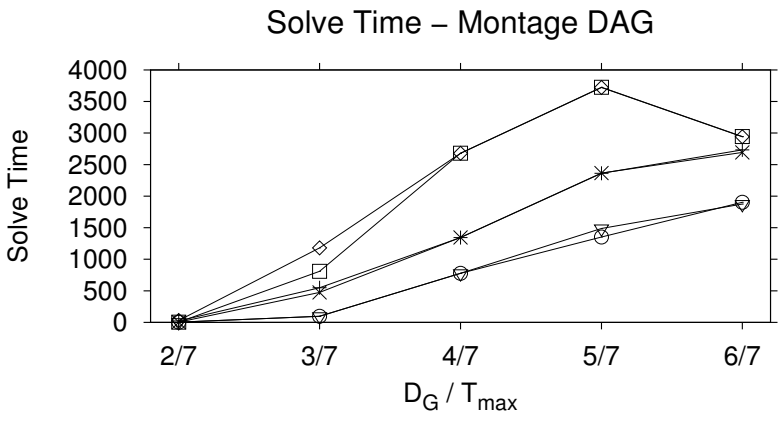


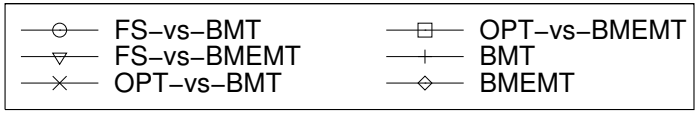
Evaluation IV

- Results for the Montage DAG.









Evaluation V

- Number of unfeasible solutions.

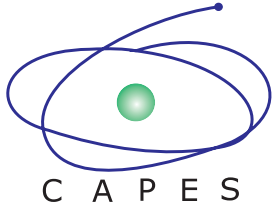
DAG	$\frac{\mathcal{D}_\mathcal{G}}{T_{max}}$	FS vs BMT	FS vs BMEMT	OPT vs BMT	OPT vs BMEMT	BMT	BMEMT
FJ-20	2/7	5	5	0	0	44	38
	3/7	0	57	4	57	0	9
	4/7	28	85	23	85	0	0
	5/7	19	71	19	71	4	0
	6/7	44	83	44	83	11	5
Mon.	2/7	100	100	94	94	100	100
	3/7	42	42	4	4	47	38
	4/7	4	4	0	0	9	19
	5/7	23	23	4	14	14	28
	6/7	50	50	33	33	44	50

Conclusion and Future works

- Customers want their jobs to be executed within an expected execution time
- The SaaS or PaaS clouds want to maximize its profit.
- We presented an ILP to solve the workflow scheduling problem in SaaS or PaaS clouds with two levels of SLA
- Two heuristics were presented (BMT and BMEMT) to find feasible integer solutions over the relaxed runs
- Simulation results shown that:
  - The optimal can generate low-cost solutions with shorter deadlines
  - The heuristics are effective to find low-cost solutions for larger deadlines
- Future works include:
  - Non-iterative methods (constraint programming, non-linear programming, ...)
  - Multiple workflow scheduling in the same set of resources

Thank You!
Questions?

Acknowledgment:



Simulation Configurations

- We used 3 IaaS providers in our simulations
- Each one with its own prices for reserved and on-demand VMs
- Maximum number of VMs that can be leased from each IaaS was:
  - $\delta_A = 4$, $\delta_B = 7$, $\delta_C = 2$.
- External links (links between IaaS providers):
  - Is taken randomly in the $[2, 3]$ interval
- Internal links (links between VMs inside the same IaaS provider):
  - Is taken randomly from the $[0.1, 0.2]$ interval
- Simulations with DAGs of real world applications:
  - Montage and Fork-join DAG with 20 nodes
- $\mathcal{D}_{\mathcal{G}}$ varying from $T_{max} \times 2/7$ to $T_{max} \times 6/7$ in $1/7$ steps
- Intel® Core™ 2 Quad CPU Q6700 2.66GHz and $8GB$ of RAM

Table : IaaS Provider A

Type	Core	Performance Per Core	On-demand Prices	Reserved Prices
Small	1	1.5	$0.13	$0.045
Medium	2	1.5	$0.20	$0.070

Table : IaaS Provider B

Type	Core	Performance Per Core	On-demand Prices	Reserved Prices
Small	1	2	$0.17	$0.045
Medium	2	2	$0.30	$0.059
Large	3	2	$0.40	$0.140
Extra-Large	4	2	$0.52	$0.183
Double Extra-Large	8	2	$0.90	$0.316

Table : IaaS Provider C

Type	**Core**	**Performance Per Core**	**On-demand Prices**	**Reserved Prices**
Small	1	2	$0.15	$0.052
Medium	2	2	$0.25	$0.088
Large	4	2.5	$0.50	$0.176
Extra-large	8	2.5	$0.80	$0.281

Table : SaaS SLAs for reserved VMs

Type	IaaS	VM	Number
Reserved	A	Small	1
Reserved	A	Medium	1
Reserved	B	Small	1
Reserved	B	Medium	1