"In Cloud (Markets) We Trust"

Towards a Trustworthy Marketplace for Cloud Resources

Azer Bestavros

Computer Science Department
Boston University

In collaboration with

Vatche Ishakian (BU), Jorge Londono (BU→U Pontificia Bolivariana), Ray Sweha (BU), and Shanghua Teng (BU→USC)

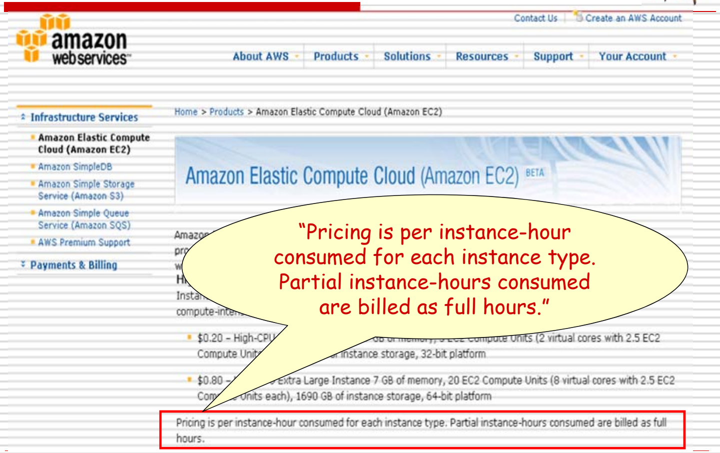
http://www.cs.bu.edu/groups/wing

DIMACS Workshop on Systems and Networking Advances in Cloud Computing
December 9, 2011

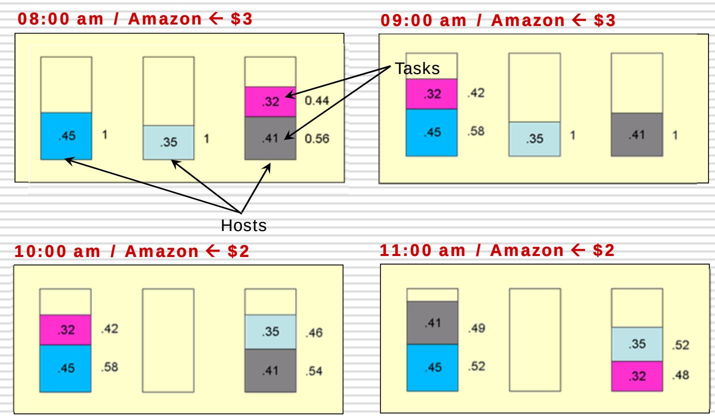
Do you trust that "the price is right"?

- Holistic system (social) view is passé
  - Tenants make resource acquisition/control decisions; no incentive to optimize for, or be fair/friendly to others – it's a marketplace
  - Infrastructure owners have no incentive to minimize cost for tenants; they only react to marketplace pressure
- Economic utility as a dimension of trust
  - Challenge is to design the mechanisms that engender trust in the cloud marketplace

Current IaaS Practice: Fixed Pricing



Marketplace Implications?



(Cloud) Colocation Games

- IaaS cloud providers offer fixed-sized instances for a fixed price
- Provider's profit = number of instances sold; no incentive to colocate customers
- Virtualization enables colocation to reduce costs without QoS compromises
- Customers' selfishness reduces the colocation process to a strategic game

Colocation Games: Questions

- Does it reach equilibrium?
- If so, how fast?
- If so, at what price (of anarchy)?
- How about multi-resource jobs/hosts?
- How about multi-job tasks?
- How about job/host dependencies?
- How could it be implemented?
- How would it perform in practice?
- ...

Colocation Game: Model

- A hosting graph $G = (V,E)$
  - V & E labeled by capacity vector R and fixed price P
- Workloads as task graphs $T_i = (V_i, E_i)$
  - V_i & E_i labeled by a utilization vector W
- Valid mappings
  - $V_i \rightarrow V$ & $E_i \rightarrow E$: $\Sigma W \leq R$; supply meets demand
- Shapley Cost function
  - Cost P of a resource is split among workloads mapped to it *in proportion to use*

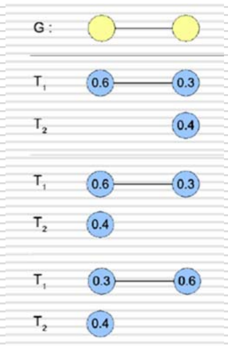
$$c_M(T_i) = \sum_{j \in \{V_i, E_i\}} P_j \frac{w_{ij}}{U_j}$$

The General Colocation Game (GCG)

- GCG is a pure strategies game:
  Each workload is able to make a (better response) "move" from a valid mapping M into another M' so as to minimize its own cost
- Example applications:
  - Overlay reservation, e.g., on PlanetLab
  - CDN colocation, e.g., on CloudFront

General Colocation Game: Properties

- GCG may not converge to a Nash equilibrium
- Theorem:
  Determining whether a GCG has a Nash Equilibrium is NP-Complete (by reduction to 3-SAT problem)
- Need more structure to ensure convergence



Colocation Games: Variants

- Process Colocation Game (PCG):
  Each workload consists of a single vertex representing an independent process that needs to be assigned to a single host with only one capacitated resource
- Multidimensional PCG (MPCG):
  Same as PCG but with multi capacitated resources
- Example applications:
  - VM colocation, e.g., on a Eucalyptus cluster
  - Streaming server colocation

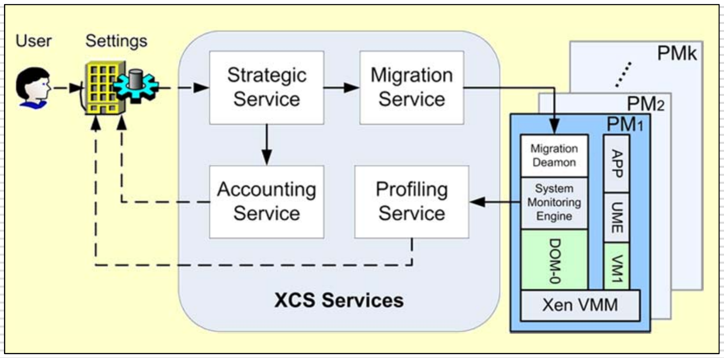
Colocation Games: Variants

- Parallel PCG (PPCG):
  Task graph consists of a set of disconnected vertices (independent processes), each with multidimensional resource utilization needs
- Uniform PPCG:
  Same as PPCG but with identical resource utilization for all processes
- Example applications:
  - Map-Reduce paradigm
  - MPI scientific computing paradigm

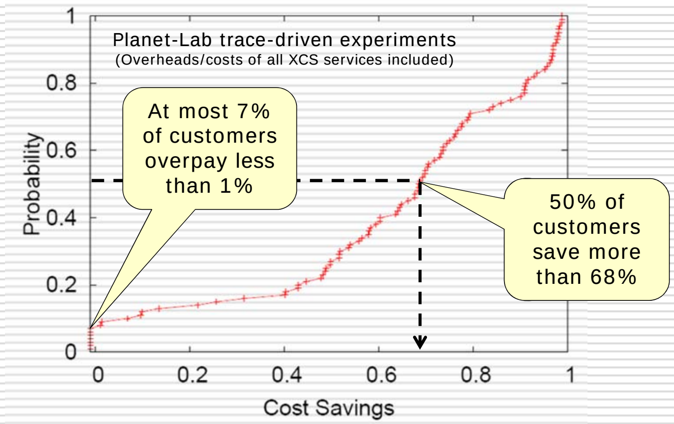
Colocation Games: Theoretical results

- PCG converges to a Nash Equilibrium under better-response dynamics
- PCG converges to a Nash Equilibrium in $O(n^2)$ better-response moves, where $n = |V|$
- Price of Anarchy for PCG is 3/2 when hosting graph is homogeneous and 2 otherwise
- MPCG converges to a Nash equilibrium under better-response dynamics
- Uniform PPCG converges to a Nash equilibrium under better-response dynamics
- …

CloudCommons: Architecture



CloudCommons: Benefit to Customers



Can we think of a better mechanism?

- Customer cost should be a function of supply and demand
  - Supply may vary over time
  - Supplier's cost may vary over time
  - Demand may vary over time
  - Demand may exhibit structure, and may be subject to malleable constraints
- Need language to specify supply and demand (and act as basis for SLAs)

Resource Supply/Demand Model

- Supply/demand SLA types: (C, T, D, W)
  - C ~ amount available or consumed
  - T ~ allocation period
  - D ~ tolerable number of missed allocations in W
  - W ~ window of >= 1 allocation intervals
- Examples
  - SLA type $(2,5,0,1)$
    2 resource units supplied/consumed every 5 seconds with no missed allocations allowed
  - SLA type $(3,30,2,5)$
    3 resource units supplied/consumed every 30 seconds with no more than 2 out of 5 missed allocations

SLA Calculus

- Models various patterns of allocation and consumption (e.g., RR, GPS, LB, …)
- SLA types define type hierarchies
  - $(1, N, 0, 1) < (k, k * N, 1, 0)$
  - $(C, T, D, W) < (C, T, D', W)$, if $D < D'$
  - …
- Possible to transform SLAs from one form to another (safer) form

Using SLA Calculus for Colocation

- Not possible to colocate

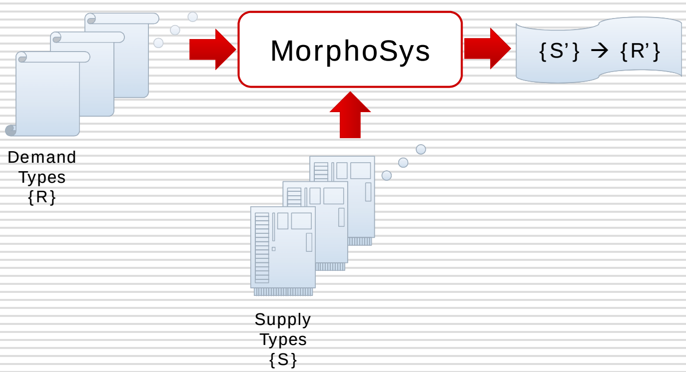
	Job 1	Job 2	Job 3	Job 4	Job 5
C	1	2	3	4	5
T	4	9	17	34	67

- Possible to colocate

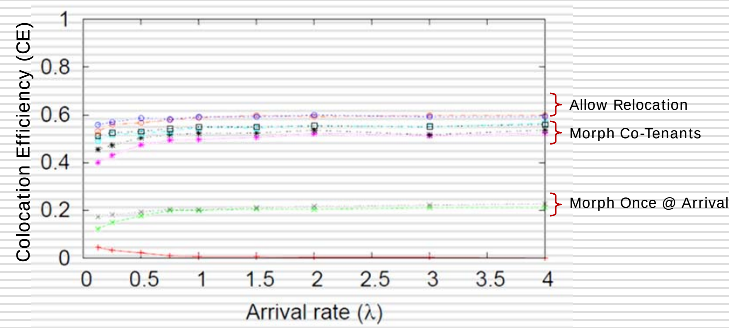
	Job 1	Job 2	Job 3	Job 4	Job 5
C	1	2	3	4	5
T	4	8	16	32	64

- SLA types and calculus provide a notion of supply & demand elasticity

Morphing SLAs for Efficiency



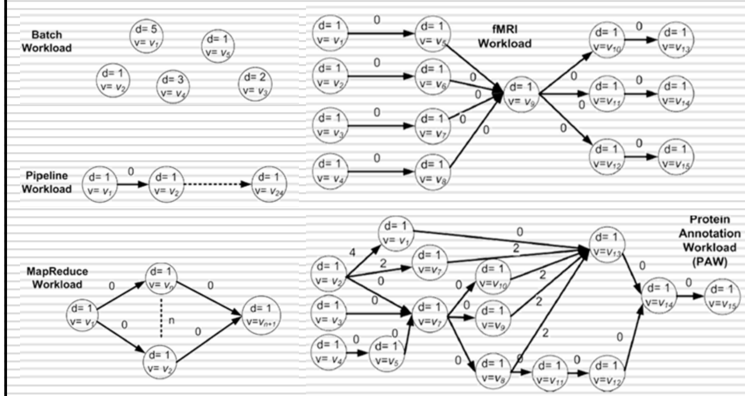
MorphoSys: Performance



Beyond Simple Types

- A workload is a set of requests (tasks), each with its SLA, subject to constraints:
  - Temporal dependencies between tasks
  - Start and end times
- Flexibilities might exist; another source of elasticity:
  - Min and max delays between tasks
  - Deadline slacks

Workload = DAG of SLA types



The Customer's Perspective

- Why should customers expose the elasticity of their workloads?
- Current IaaS (fixed) pricing mechanisms do not provide proper incentives
- Implications:
  - Less efficient workload management
  - Customers (should) game the marketplace

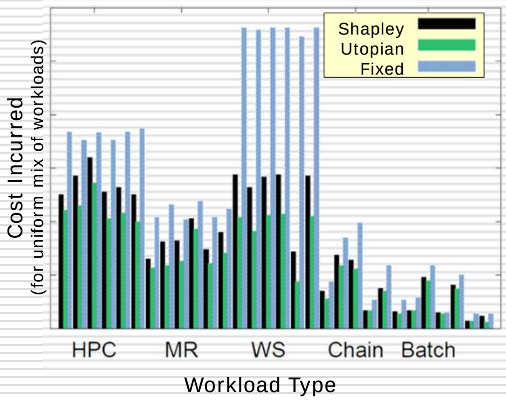
Dynamic Pricing: Shapley Value

- Well defined concept for fair cost sharing from coalitional game theory
  - Marginal contribution to the total cost, averaged over every permutation, e.g., for 3 workloads

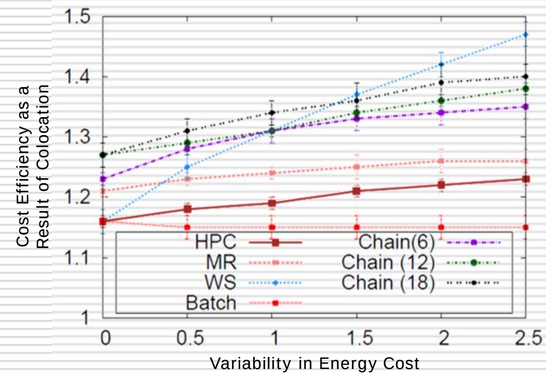
$$s(w_1) = \frac{1}{6}\left(\begin{array}{l} 2\,c(w_1) + [c(w_2 w_1) - c(w_2)] + [c(w_3 w_1) - c(w_3)] + \\ [c(w_2 w_3 w_1) - c(w_2 w_3)] + [c(w_3 w_2 w_1) - c(w_3 w_2)] \end{array} \right)$$

  - Impractical to calculate
  - Estimate by sampling random permutations

Workload Elasticity = Savings



Workload Elasticity = Savings



Conclusion

- Resource management must be seen in an economics context
- By setting up the right mechanisms, one can engender trust in the cloud marketplace
- Supply elasticity meets demand elasticity for an efficient marketplace
- New services needed to support strategic and operational aspects of new mechanisms

CloudCommons: http://csr.bu.edu/cc

Supported by a NSF Large CyberTrust Award (in collaboration with Brown and UCI)