Ginseng: Market-Driven Memory Allocation

Orna Agmon Ben-Yehuda[1] Eyal Posener[1]
Muli Ben-Yehuda[1,2] Assaf Schuster[1] Ahuva Mu'alem[1,3]

[1]Department of Computer Science
Technion — Israel Institute of Technology
and
[2]Stratoscale
and
[3]Ort Braude

VEE, March 2014

What is the future of the IaaS Cloud?

- Fine resource granularity
- Fine time granularity
- Market-driven resource pricing
- Tiered service levels (like spot instances)

We predicted that these trends will culminate in a new cloud model, the Resource-as-a-Service (RaaS) Cloud.
More details in:

- *The Resource-as-a-Service (RaaS) Cloud*. Orna Agmon Ben-Yehuda, Muli Ben-Yehuda, Assaf Schuster, and Dan Tsafrir. HotCloud, June 2012.
- *The rise of RaaS: the Resource-as-Service Cloud*. Orna Agmon Ben-Yehuda, Muli Ben-Yehuda, Assaf Schuster, Dan Tsafrir. Communications of the ACM. Forthcoming, July 2014.

Designing a Resource Allocation Mechanism for the RaaS Cloud

- Maximize social welfare (sum of guest valuations=benefit)
  - Prefer production over debugging
  - On-line trading—prefer guests that have more expensive deals done
- Pareto-Efficient (any guest's allotment cannot be improved without hurting another guest)

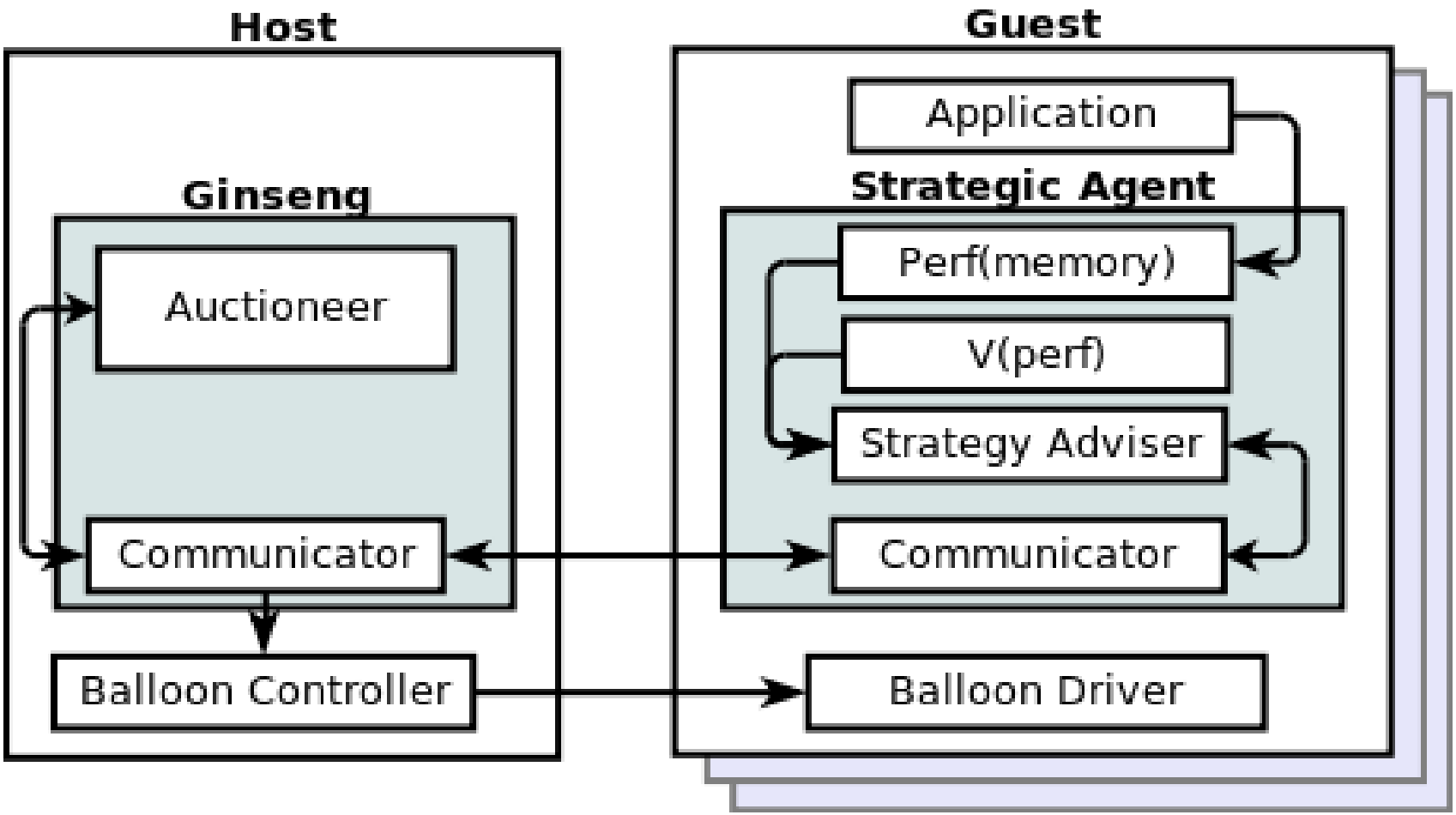
We focus on the hardest resource of all—memory.

Allocating Memory for Virtual Machines

In a real commercial cloud the host and guests belong to different economic entities.

- **White box** approaches cannot work in a real commercial cloud.
  - What is the guest doing? What should be measured? How?
  - How much is the performance worth to the client?
  - Whose fault is it that the guest's performance is low? Maybe the software is inefficient?
- **Black box** approaches cannot work in a real commercial cloud.
  - Host measurements: memory stress can be faked to induce the host to allocate more memory.
  - Guest measurements: results can be mis-reported.
- Ginseng uses an **economic mechanism** that incentivizes guests to reveal how much memory is worth to them, so it can optimize the social welfare.

Ginseng


Host
Guest
Application
Ginseng
Strategic Agent
Auctioneer
Perf(memory)
V(perf)
Strategy Adviser
Communicator
Communicator
Balloon Controller
Balloon Driver

Designing an Allocation Mechanism

- VCG (second price): Vickrey (1961), Clarke (1971), Groves (1973)
- Each guest bids with a type (a valuation of the good - how much it is worth, subjectively)
- The auctioneer finds the allocation that maximizes the social welfare.
- The auctioneer charges guests according to the **exclusion-compensation** principle: the difference between the social welfare when they exist to the social welfare when they do not exist.
- Prices are NOT uniform. Prices may drop to a minimal price (possibly zero) if there are enough resources.

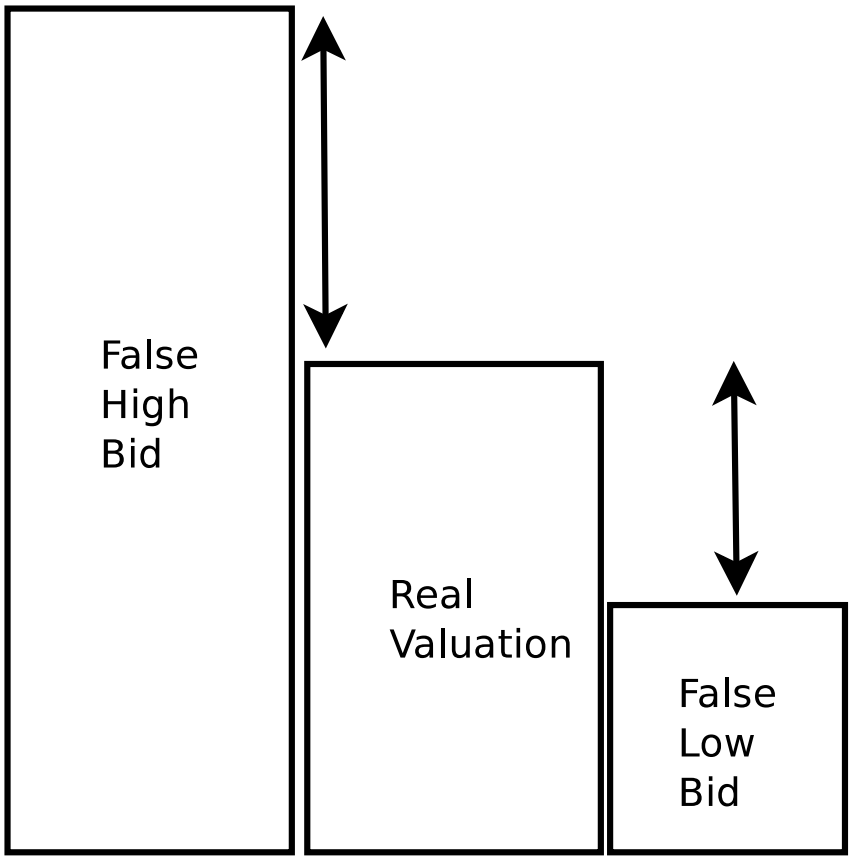
A VCG auction is **truthful**: guests bid their real types, no matter what other guests do.

Example: Second Price Auction

Example: Second Price Auction

Example: Second Price Auction

How much should one bid in a second price auction?



The second price auction is truthful

Designing a Bandwidth Allocation Mechanism

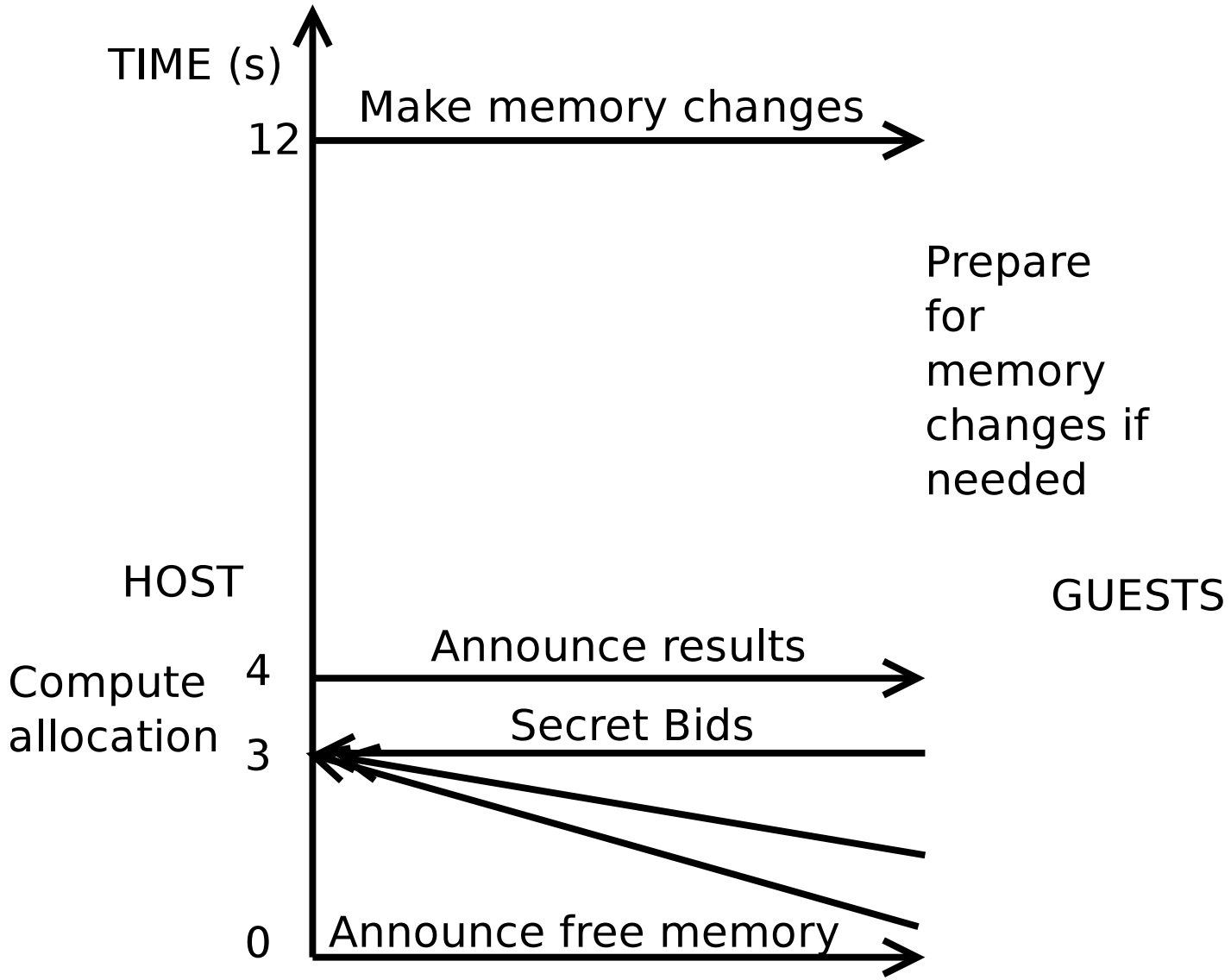
Progressive Second Price (PSP) auction, Lazar and Semret (1999):

- Each guest bids with a required quantity and a unit-price (not with their full types).
- The highest unit-price bidders are allocated with resources at desired quantities.
- Guests are charged by the exclusion-compensation principle.
- Guests hear each other's bids, change their requested quantities and converge to an equilibrium.

Challenges in Designing a Memory Allocation Mechanism for the Cloud

- Maintaining privacy. (spoiler: our **Memory Progressive Second Price (MPSP)** protocol is based on secret bids.)
- Keeping memory from changing hands too often.
- Supporting real-world valuation and performance profiles.

A Round in The MPSP Auction Protocol



PSP is not Ideal for Memory Allocation: Cache Warmup

Memory is only beneficial if you use it long enough (e.g. allowing cache warmup)

- In case of a tie between guests, none of the PSP guests wins the good.
- In the MPSP auction, ties are broken in favor of the guest currently holding the memory, as well as by a random order, such that memory does not go unused if it is needed.

PSP is not Ideal for Memory Allocation: Concavity

- PSP is for concave monotonically rising valuation functions, and for a divisible good only.
- Clients may sometimes value memory thus, in smart applications:

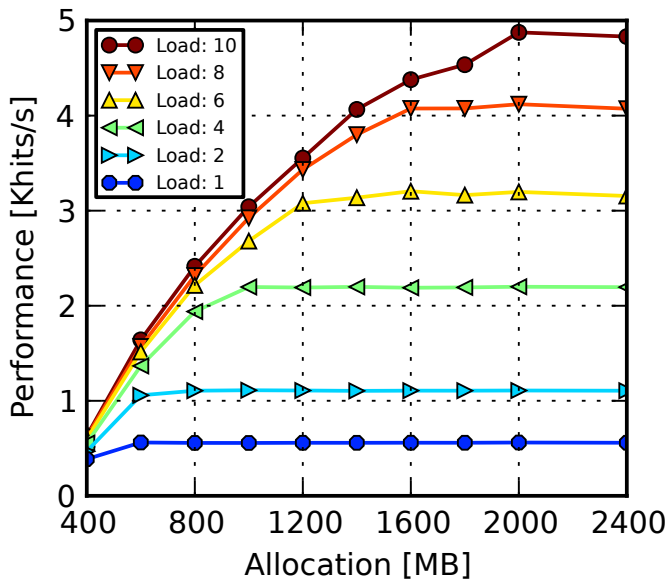


Figure: Elastic-memory memcached: `https://github.com/ladypine/memcached`

PSP is not Ideal for Memory Allocation: Concavity

- PSP is for concave monotonically rising valuation functions, and for a divisible good only.
- But legacy applications tend to have a step function performance graph, which is not concave:

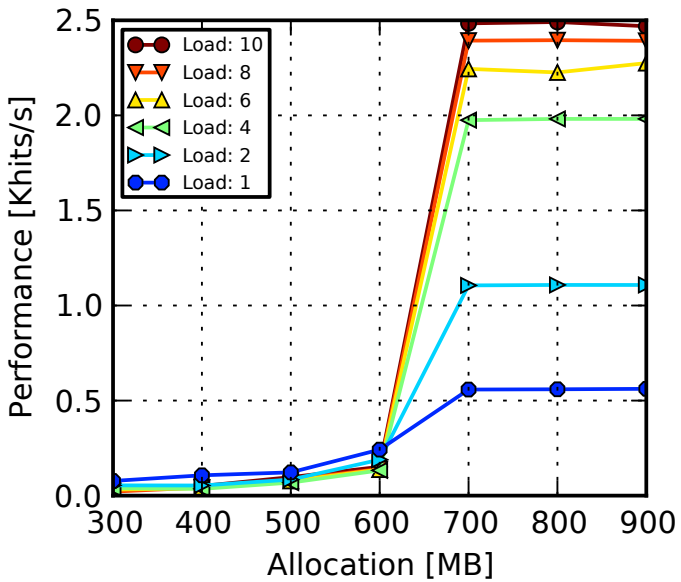


Figure: Off-the-Shelf memcached

PSP is not Ideal for Memory Allocation: Non-Smooth Online Measurements

- PSP is for concave monotonically rising valuation functions, and for a divisible good only.
- Sometimes, especially when performance is measured on-line, the performance is not even monotonicaly rising

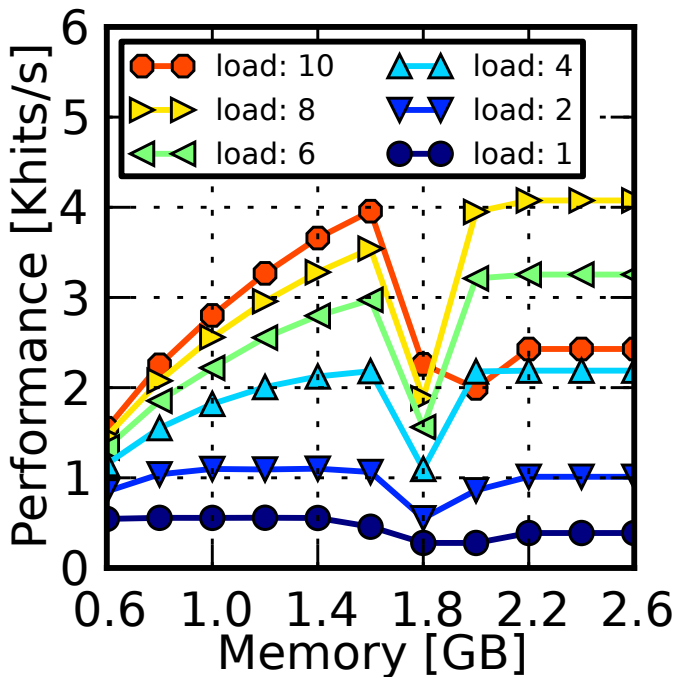
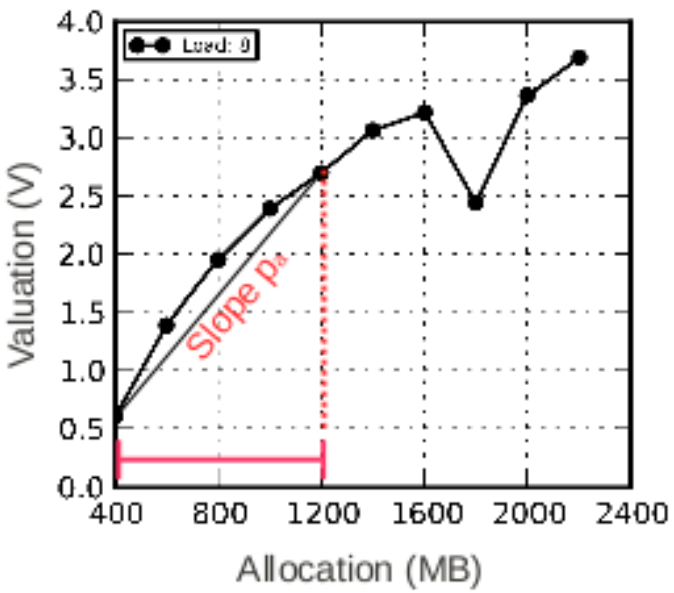


Figure: Elastic Memory memcached, in a non-optimal environment

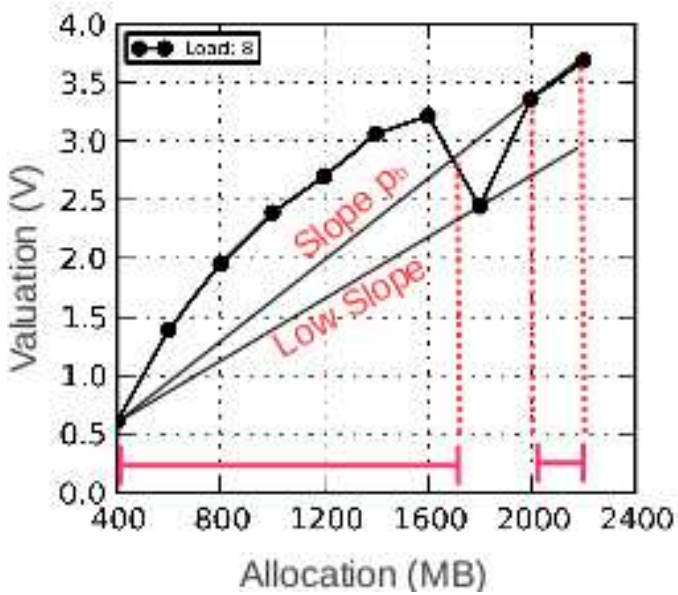
MPSP supports non-concave, non-monotonically rising functions

The MPSP auction supports memory ranges:

- A bid is composed of a single unit-price and multiple memory ranges.



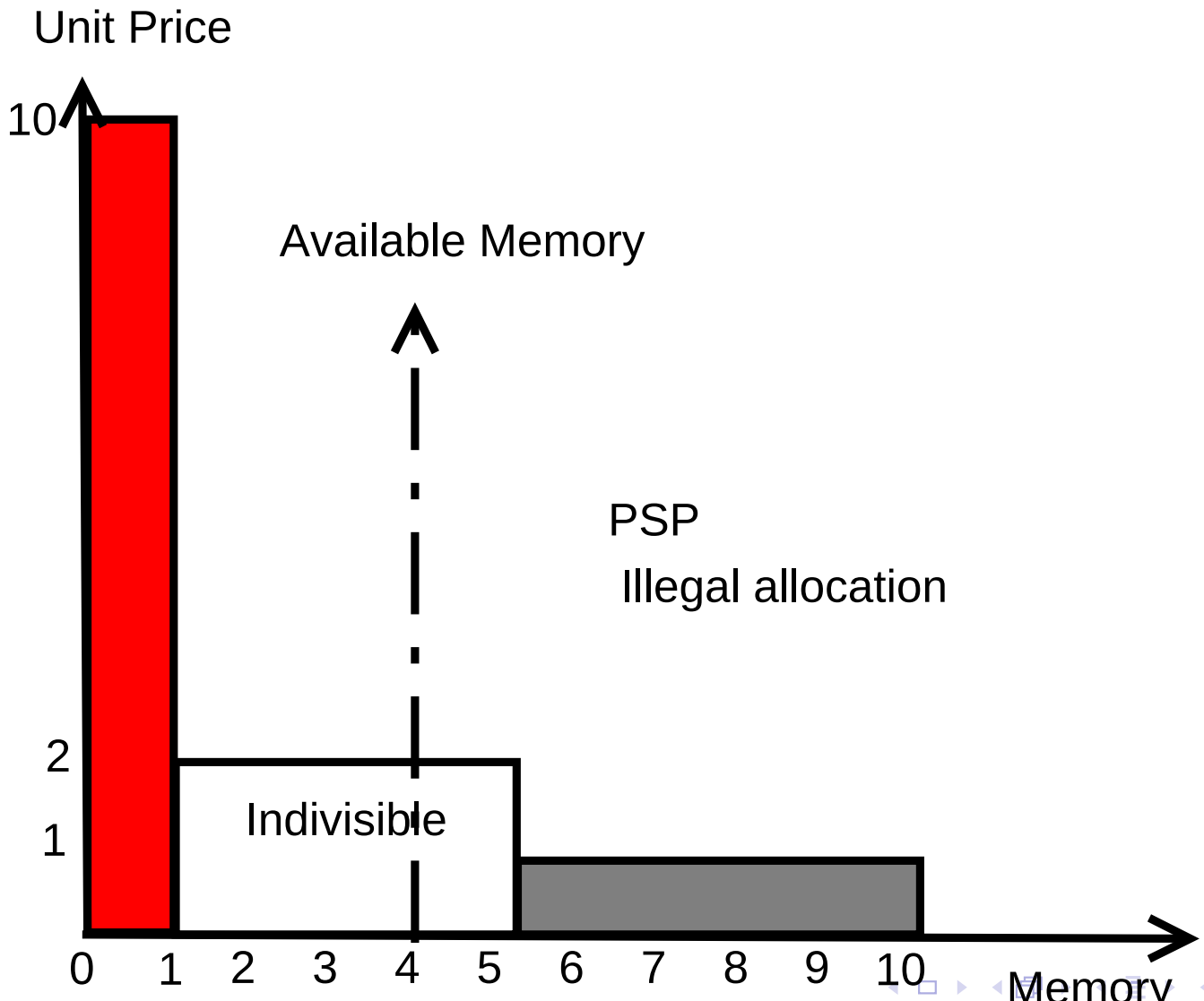
(a) *Single range*



(b) *Multiple range*

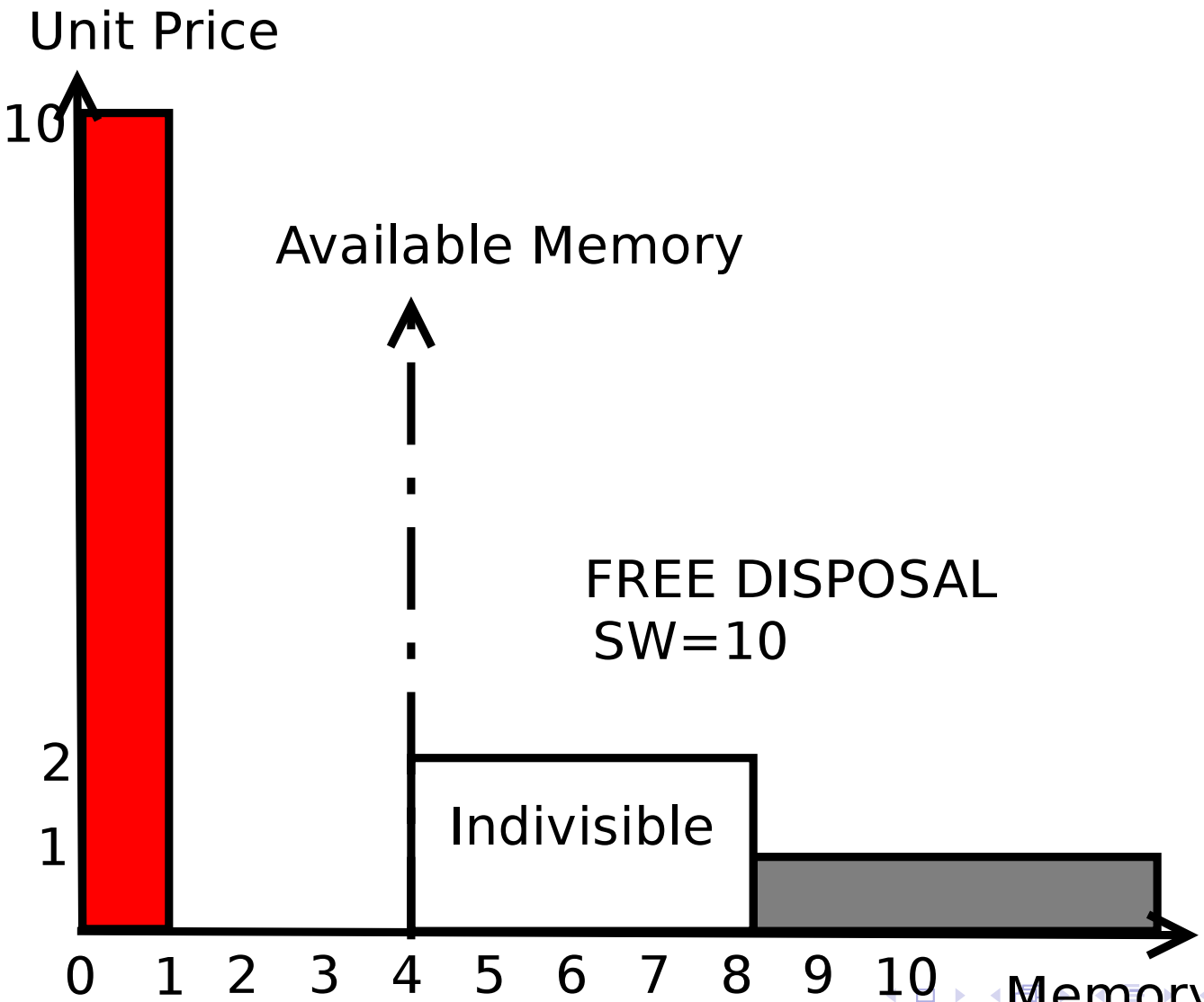
The MPSP Allocation Choice

The PSP allocation prefers higher unit prices, assuming there are no forbidden ranges:



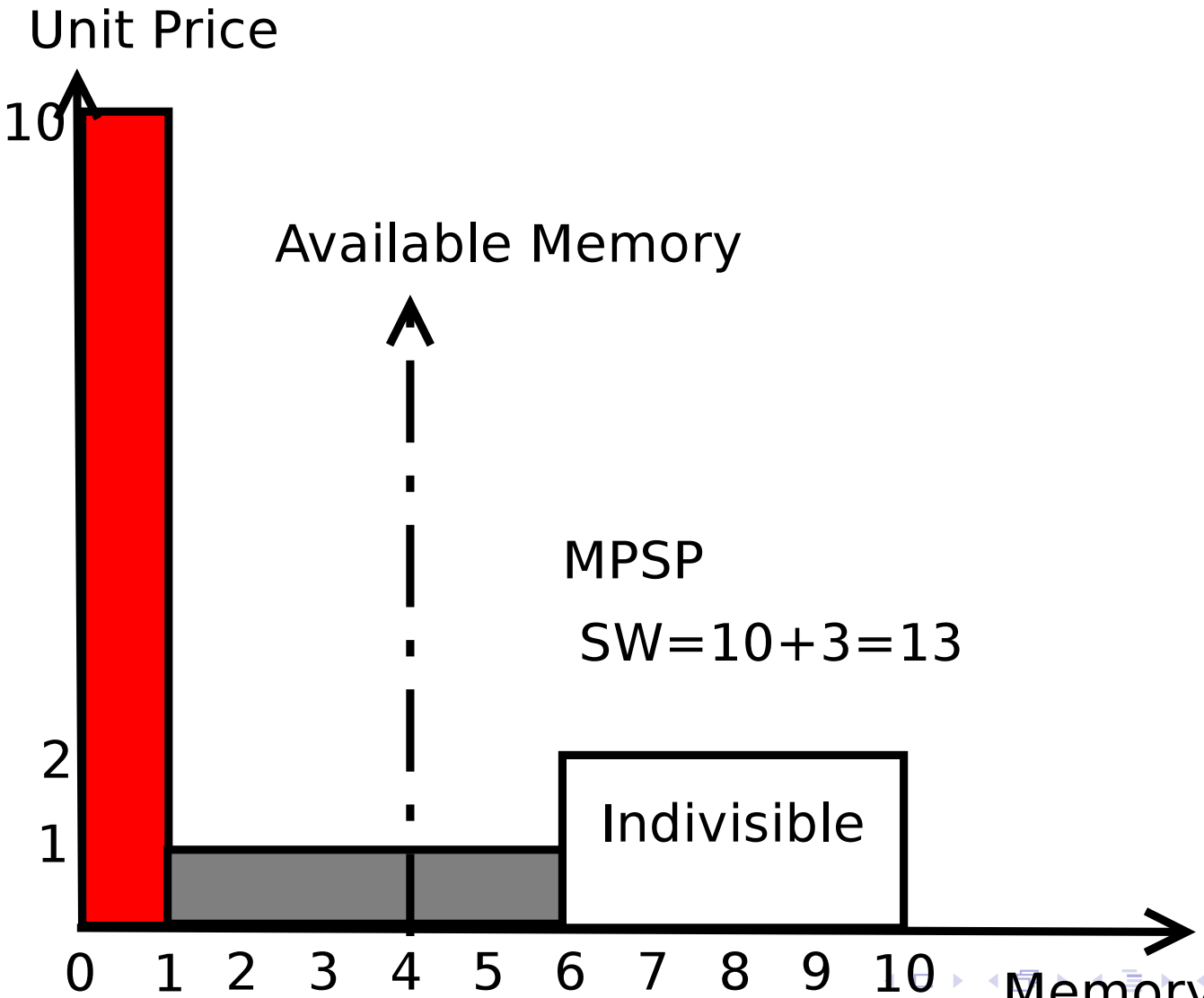
The MPSP Allocation Choice

Free disposal of auction results supports forbidden ranged, but is inefficient:



The MPSP Allocation Choice

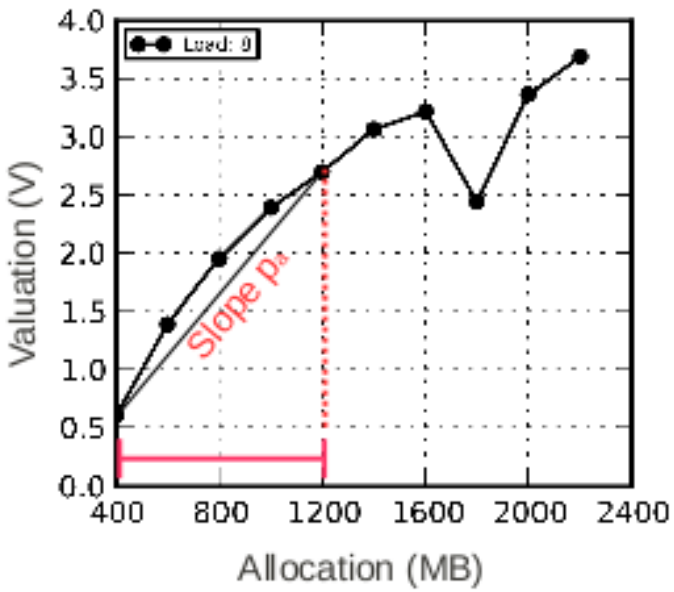
The MPSP allocation finds the allocation with the highest social welfare under the forbidden ranges constraints:



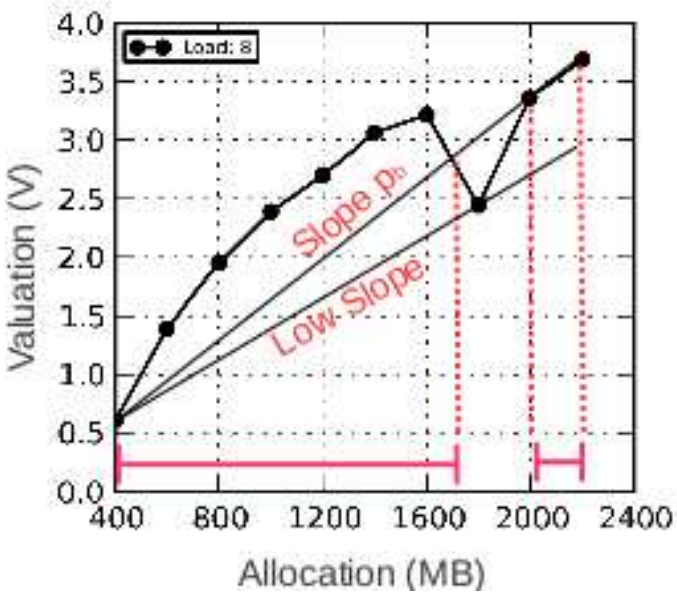
Guests in the MPSP are close to being truthful

Bidding the true valuation of a memory quantity is the best course of action when:

- The guest asks for a specific quantity (not a range), or
- The valuation function is concave monotonically rising, or
- The system is at a steady state.



(c) *Single range*



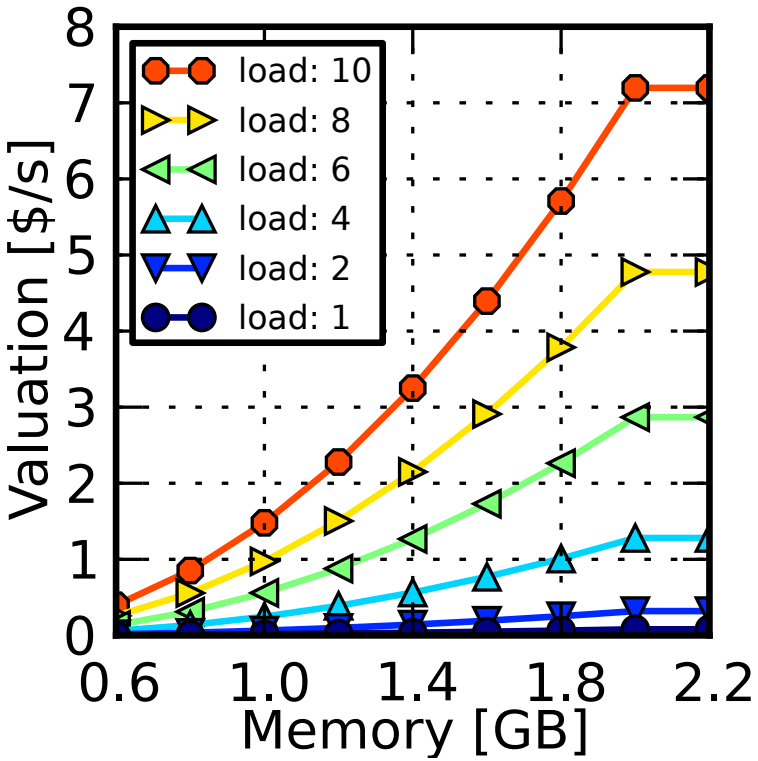
(d) *Multiple range*

Positive Results

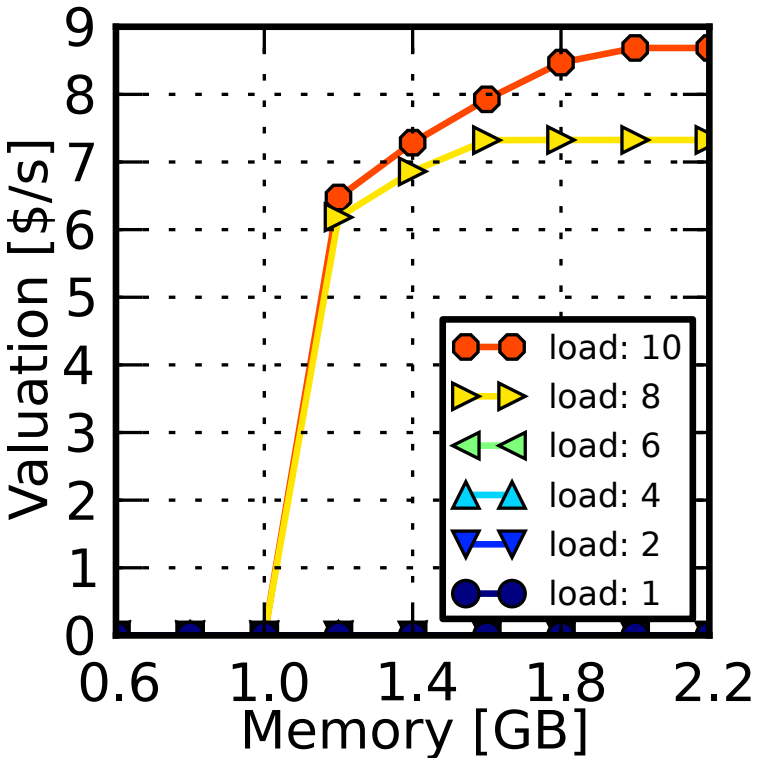
- MPSP maximizes the *Social Welfare* given the bids, even for non-concave non-monotonically rising valuations.
- Guests are almost always *p-truthful*.
- Guests can learn on-line and bid for a quantity that improves their profit, thus taking into effect the exclusion compensation payments (more in the paper...).

As a result, Ginseng can optimize the social welfare.

Experimental Evaluation: Non-Concave Valuation Functions



(e) ***MemoryConsumer (square of performance)***



(f) ***Dynamic Memcached (partially linear)***

Figure: Valuation functions for different loads

Experimental Evaluation: Comparison to Other Methods on a Dedicated Dynamic-Memory Benchmark

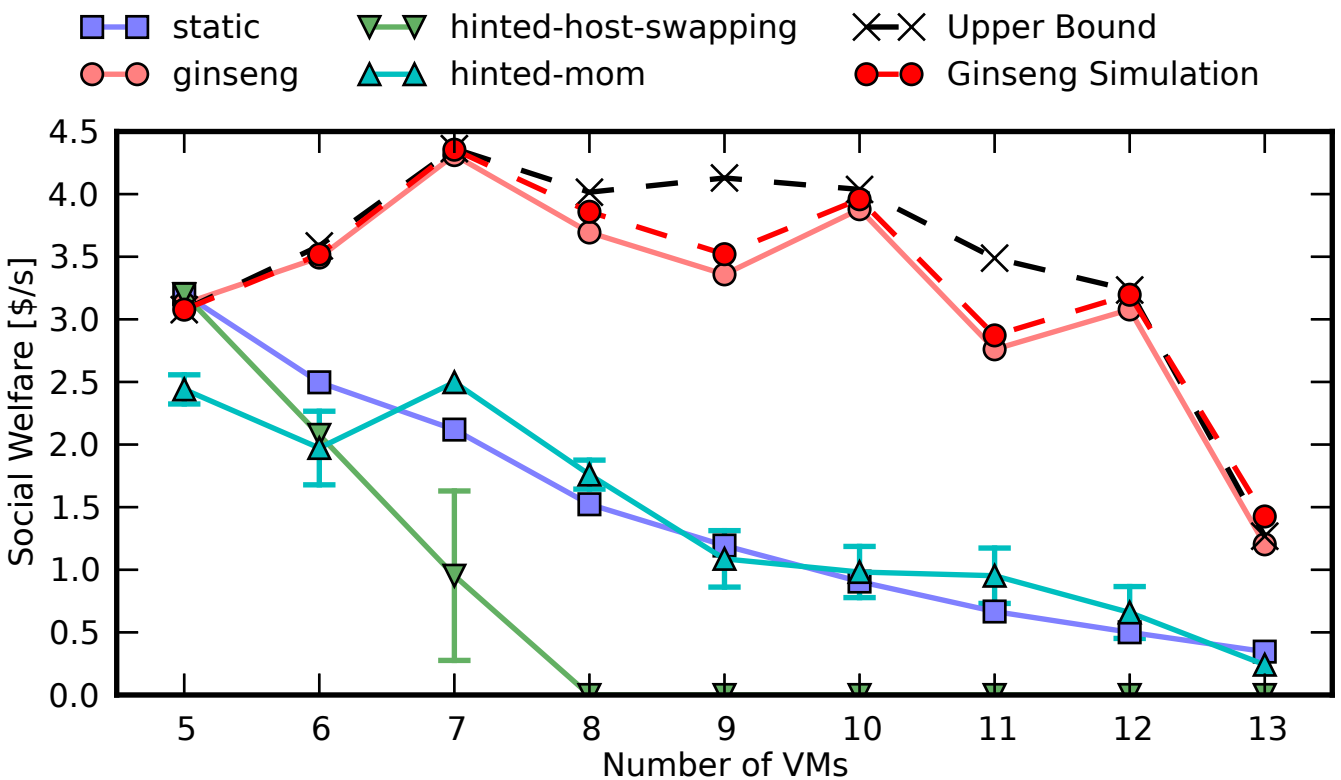


Figure: MemoryConsumer, valuation is square of performance

$\times 6.2$ improvement!

Experimental Evaluation: Comparison to Other Methods on a Dynamic-Memory Memcached

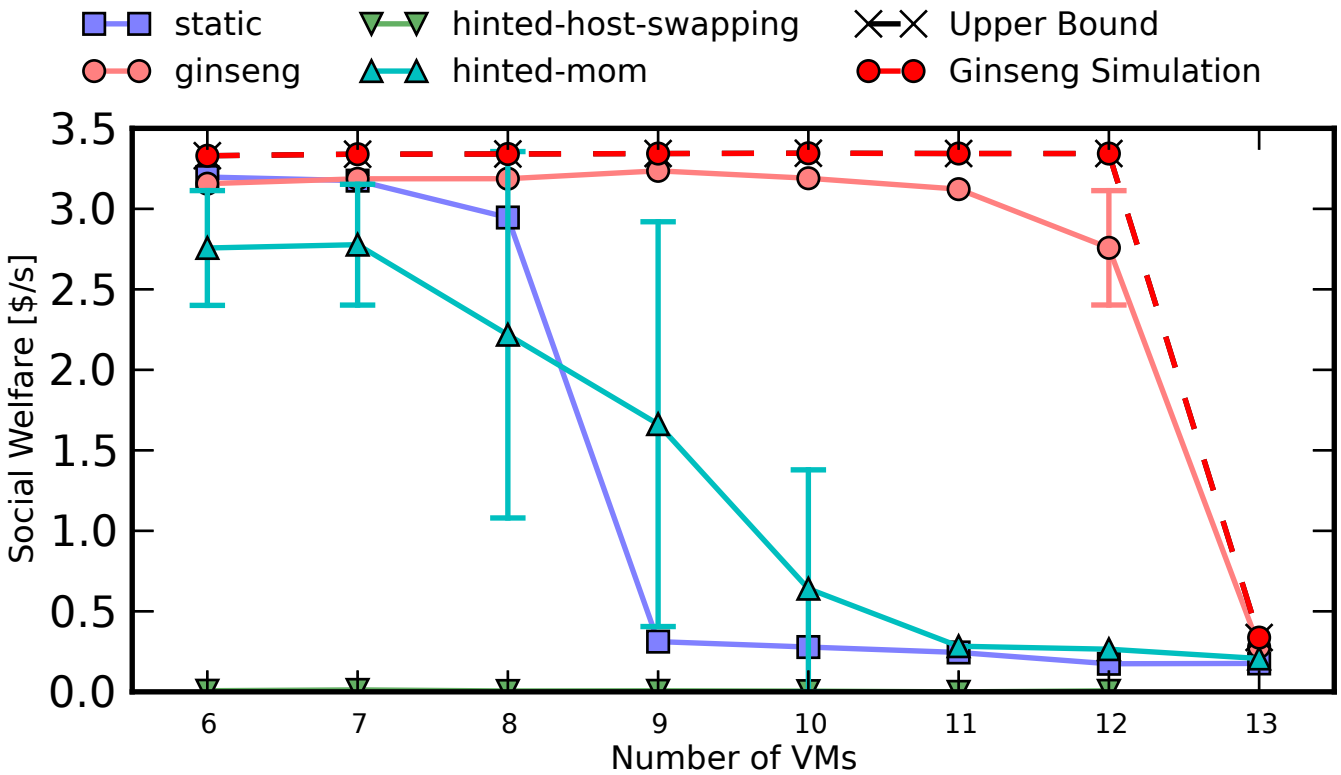


Figure: Memcached, first guest valuation is piecewise linear

$\times 15.8$ improvement!

Conclusion

- Money (economic valuations) is the best way to give priorities, especially in the cloud.
- Black-box approaches are limited and error-prone.
- White-box is not feasible.
- But Ginseng, a selfishness-based mechanism, does the trick!

Thank You for Listening!

Contact us at:
{ladypine, posener, muli, assaf, ahumu} at cs.technion.ac.il

Experimental Evaluation: Performance is comparable

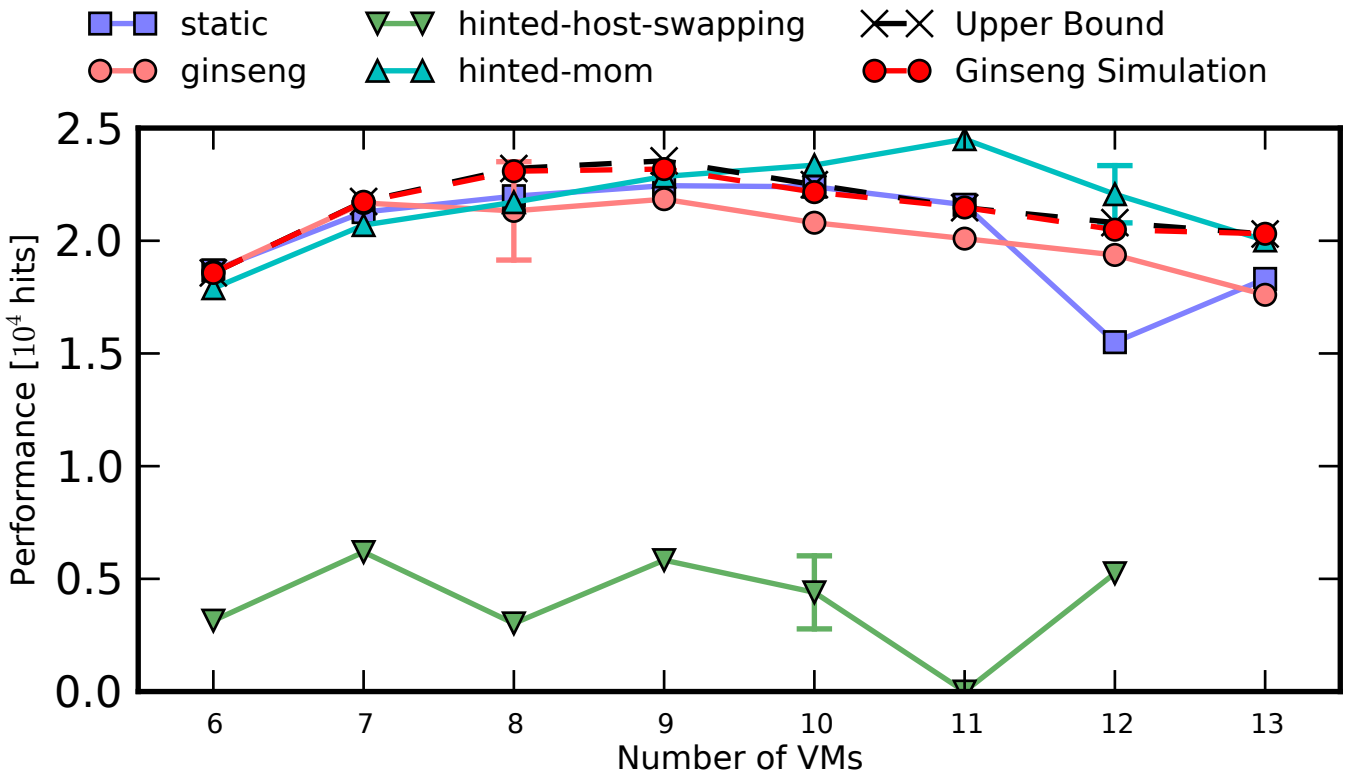


Figure: Memcached, first guest valuation is piecewise linear

Offline profiling is good enough

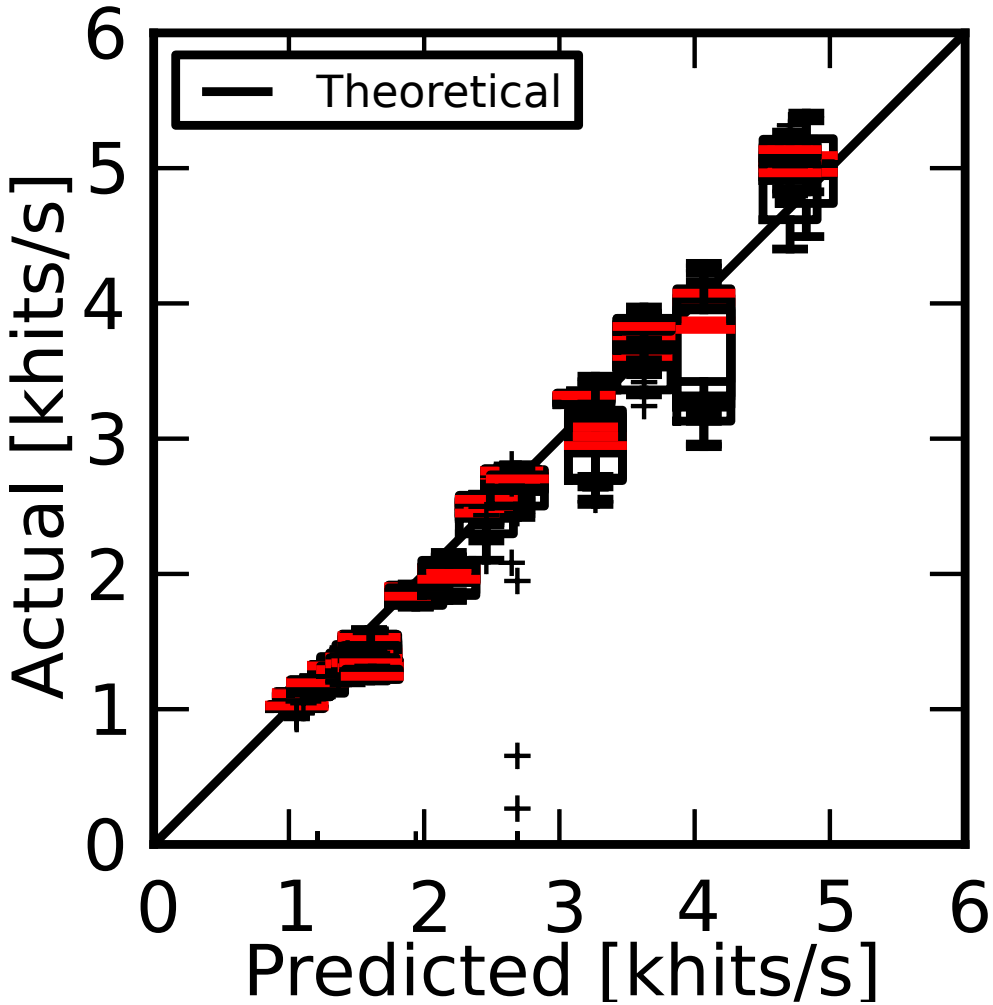


Figure: Memcached