Market Driven Multi Resource Allocation

Liran Funaro
Under the supervision of Prof. Assaf Schuster and Dr. Orna Agmon Ben-Yehuda

Department of Computer Science

Cost Efficient Scaling
Jan. 26, 2020
One of the main challenges of public and private cloud providers needs to serve all the clients on each server according to their service-level-agreement (SLA). This affects utilization, hence, affects the number of clients per server. Thus, it affects the provider’s operation cost per client.
Resource Allocation

- One of the main challenges of public and private cloud providers
- Needs to serve all the clients on each server according to their service-level-agreement (SLA)
- Affects utilization
  - Hence, affects the number of clients per server
  - Thus, affects the provider’s operation cost per client
- Can reduce the provider’s operation costs
- Coupled with a fitting pricing scheme, it can increase the provider’s profits
Our Goals

- Explore designs for such resource allocation schemes
  - Increase the resource utilization
  - Taking into account the financial needs of providers and clients
Our Goals

▶ Explore designs for such resource allocation schemes
  ▶ Increase the resource utilization
  ▶ Taking into account the financial needs of providers and clients
  ▶ That is, design a market driven resource allocation scheme
Our Goals

- Explore designs for such resource allocation schemes
  - Increase the resource utilization
  - Taking into account the financial needs of providers and clients
  - That is, design a **market driven resource allocation** scheme

- What is the gap between the current resource utilization to an optimal one?
- What is the origin of the gap?
- What are the provider’s economic requirements?
- What are the clients’ economic requirements?
The Problem: Fixed Resource Bundles

- Resources in the cloud are underutilized
- The main cause of resource underutilization is fixed performance bundles

- Clients rent the resources to sustain their highest workload
  - But they do not use the resources all the time
- The provider guarantees with good probability that the clients will be able to use their rented resources at any given time
- It must reserve these resources
  - It cannot resell them or use them to other purposes
Our Approach

- Incentivizing clients to reduce their fixed reserved resource requirements
  - With an option to add resources on the fly on demand
Incentivizing clients to reduce their fixed reserved resource requirements
  ▶ With an option to add resources on the fly on demand
▶ How? By designing an allocation mechanisms that incorporates a smart pricing scheme
Our Mechanisms

Two different mechanisms, each is suitable for different goals

▶ **Auction-based mechanism**: optimizes the clients’ economic benefit


▶ **Stochastic allocation mechanism**: allocate a stochastic amount of resources alongside a fixed, reserved, amount


Both mechanisms improve hardware utilization by using some kind of economic mechanism that incentivize clients to reduce the fixed, reserved, portion of their bundle.
Auction-Based Mechanism

- Auction mechanism that optimize the *social welfare*
  - The aggregated value all the clients draw from the cloud
- Each client rents a base resource bundle that is reserved for it
Auction-Based Mechanism

- Auction mechanism that optimize the *social welfare*
  - The aggregated value all the clients draw from the cloud
- Each client rents a base resource bundle that is reserved for it
- Then...
The host announces an auction every few seconds.
Auction Protocol

The host announces an auction every few seconds

Each guest bids with a valuation for each quantity of additional resource — how much it is worth, subjectively
Auction Protocol

The host announces an auction every few seconds.

Each guest bids with a valuation for each quantity of additional resource — how much it is worth, subjectively.

The host solves an optimization problem: finding the allocation that maximize the social welfare.
Auction Protocol

The host announces an auction every few seconds

Each guest bids with a valuation for each quantity of additional resource — how much it is worth, subjectively

The host solves an optimization problem: finding the allocation that maximize the social welfare

The host informs the guests of their allocation and charges them according to the exclusion-compensation principle
Exclusion-Compensation Principle: Each guest pays for the damage it inflicted on the other guests in the system

If the demand is low, clients can rent the additional resources in a very low price, which is financially beneficial to them

It incentivizes clients to rent a smaller bundle because the client can bid for additional resources at a lower price on average compared to the reservation price
First introduced by Orna Agmon Ben-Yehuda for RAM allocation

We extended this mechanism to last-level-cache (LLC) allocation
First introduced by Orna Agmon Ben-Yehuda for RAM allocation

We extended this mechanism to last-level-cache (LLC) allocation

- Our mechanism can improve the aggregate benefit of the clients in a single physical machine
- Guests can utilize their cache fast enough to allow rapid changes in the allocation
New Challenges

High computational complexity

Memory elastic applications
Finding the optimal allocation has a high computational complexity
- Forces a long time period between auctions—more than an hour
- For multi resource: RAM, LLC, CPU, BW, etc.
Finding the optimal allocation has a high computational complexity
- Forces a long time period between auctions—more than an hour
- For multi resource: RAM, LLC, CPU, BW, etc.

We introduced a new efficient multi-resource auction algorithm with a pseudo near linear complexity

Allow a multi-resource auction every two minutes for up to 256 clients
- On a single core
- Embarrassingly parallel
New Challenges

- High computational complexity
- Memory elastic applications
Memory Elastic Applications

- Resource elastic applications performance is proportional to the resource availability
- For example, cache elasticity
  - Looks similar for CPU and BW

![Application Performance Graph]

- Memory elastic applications are scarce
- Usually looks like this (thrashing)
Resource elastic applications performance is proportional to the resource availability.

For example, cache elasticity.
- Looks similar for CPU and BW.

Memory elastic applications are scarce.
Usually looks like this (thrashing).

---

L. Funaro (Technion)

Market Driven Multi Resource Allocation

Cost Efficient Scaling
Why do developers toil towards making performance scale nicely with the CPU and bandwidth, but neglect doing this for memory?
A proof that memory-elastic applications exist or can be created is essential to break this circular dependency.
Mechanisms that were designed to allow trade-off between memory and other resources can be used to provide memory elasticity.
Memory as Cache

- Some applications use the RAM to cache computation results, network traffic, and so on (e.g., using memcached)
- They can seamlessly improve their performance when more memory is available to the operating system
Intermediate Calculations

- Applications that handles data that is larger than the available memory
  - E.g., databases, Spark (map/reduce), deep neural network
- Can use larger memory buffers to reduce secondary memory access and speed up temporarily data-heavy operations
  - E.g., sorting, large matrix multiplication, deep neural network data propagation
Applications with automatic memory management (e.g., Java applications) may need fewer garbage-collection cycles with a larger heap, and improve their performance as depicted in Figure 3.
Some applications have multiple short-lived jobs, each with different memory requirements.

- For example, web servers might require a certain memory to handle each session.
- They may be able to handle more concurrent sessions when more memory is available.

To deal with lack of memory, they can cap the number of concurrent sessions.

- They trade off memory for latency and throughput.
New Challenges

- High computational complexity
- Memory elastic applications
Auction Drawbacks

- Requires a large and constant amount of computational power
  - Could have been allocated to the clients
- Does not take the provider’s profits into account
Under the SA mechanism, the provider offers clients a combination:
- an amount of reserved resources
- with a choice of a stochastic allocation class

- The provider posts fixed unit-prices for both goods
- And periodically publishes statistics on resource availability for each SA class
- Each client may choose to rent reserved and/or stochastic resources
Conclusions

▶ We developed new market-driven resource allocation schemes
▶ We showed how they can improve
  ▶ Financial properties: social welfare and profits
  ▶ Technical properties: resource utilization and the number of clients per server
▶ We designed, developed and implemented as open source frameworks
  ▶ Algorithms to support these mechanisms
  ▶ Rigorous evaluation methodologies
▶ Our work has demonstrated how sophisticated allocation and pricing mechanisms can improve hardware—and thus energy—efficiency in the cloud significantly
▶ We believe that more research on resource allocation mechanisms in the cloud would help cloud providers immensely


Liran Funaro, Orna Agmon Ben-Yehuda, and Assaf Schuster. “Memory Elasticity Benchmark”. In: 2019

Questions?

Liran Funaro: funaro@cs.technion.ac.il