RELIABILITY IN MODERN CLOUD SYSTEMS

Summer 2025

LOGISTICS

ASSIGNMENT 2

- Grades were released last Wednesday

ASSIGNMENT 3

- Grades to be released by next Wednesday

ASSIGNMENT 4

- Check-In #2 on Monday
- Assigned Time Slots for Efficiency:
  - 2:00-2:10: Lukas + Bastien
  - 2:10-2:20: Paritosh
  - 2:20-2:30: Jinhao + Bekhrouz
  - 2:30-2:40: Ali Fahad + Zawayar
  - 2:40-2:50: Aiman + Asim
  - 2:50-3:00: Felix + Marius
  - 3:00-3:10: Talal + Umair

TESTING DISCUSSION

PAPER SUMMARY #1

PAPER SUMMARY #1

- Implement microservices with formal guarantees
- Deploys the service on serverless framework
  - Uses persistent state for objects
- Adds 2 primitives:
  - Transactions
  - Async service invocations

PAPER SUMMARY #2

PAPER SUMMARY #2

- Uses P# to verify the Azure Batch PoolManager
  - Wrote a P# specification
  - Liveness and Safety Properties specified
  - Nothing is merged into master unless all tests pass
- Missed bugs were caused by:
  - Incomplete behavior when mocking dependencies
  - Missing tests
- P# didn’t scale well for industry use-cases
  - They had to make improvements

DISCUSSION THEMES

- If you are building a system, which testing techniques are useful?
- If a model checker, for a given model of a system, does not find any property violations, does this mean that there are no property violations in the system implementation?
- How can developers combine formal methods with actual runtime behaviour of implementations?

DISCUSSION THEMES

- If a model checker, for a given model of a system, does not find any property violations, does this mean that there are no property violations in the system implementation?

DISCUSSION THEMES

- If a model checker, for a given model of a system, does not find any property violations, does this mean that there are no property violations in the system implementation?

Not necessarily. Model may not capture all behaviours. Model Checker may not be able to explore all the states in a bounded time.

DISCUSSION THEMES

- How can developers combine formal methods with actual runtime behaviour of implementations?

DISCUSSION THEMES

- How can developers combine formal methods with actual runtime behaviour of implementations?

- Apply formal techniques directly on the implementation
  - Model check an implementation directly (MODIST, SAMC, FlyMC)
  - Even harder than regular model checking
- Use logs for checking if the execution adheres to the formal model
  - Check if the properties hold on the execution logs (PObserve at Amazon AWS)

DISCUSSION THEMES

- How can developers combine formal methods with actual runtime behaviour of implementations?

- Apply formal techniques directly on the implementation
  - M… …C)
  - Ev…
- Use lo… …l
  - Ch… …t Amazon AWS)

USE RUST!

DISCUSSION THEMES

- If you are building a system, which testing techniques are useful?

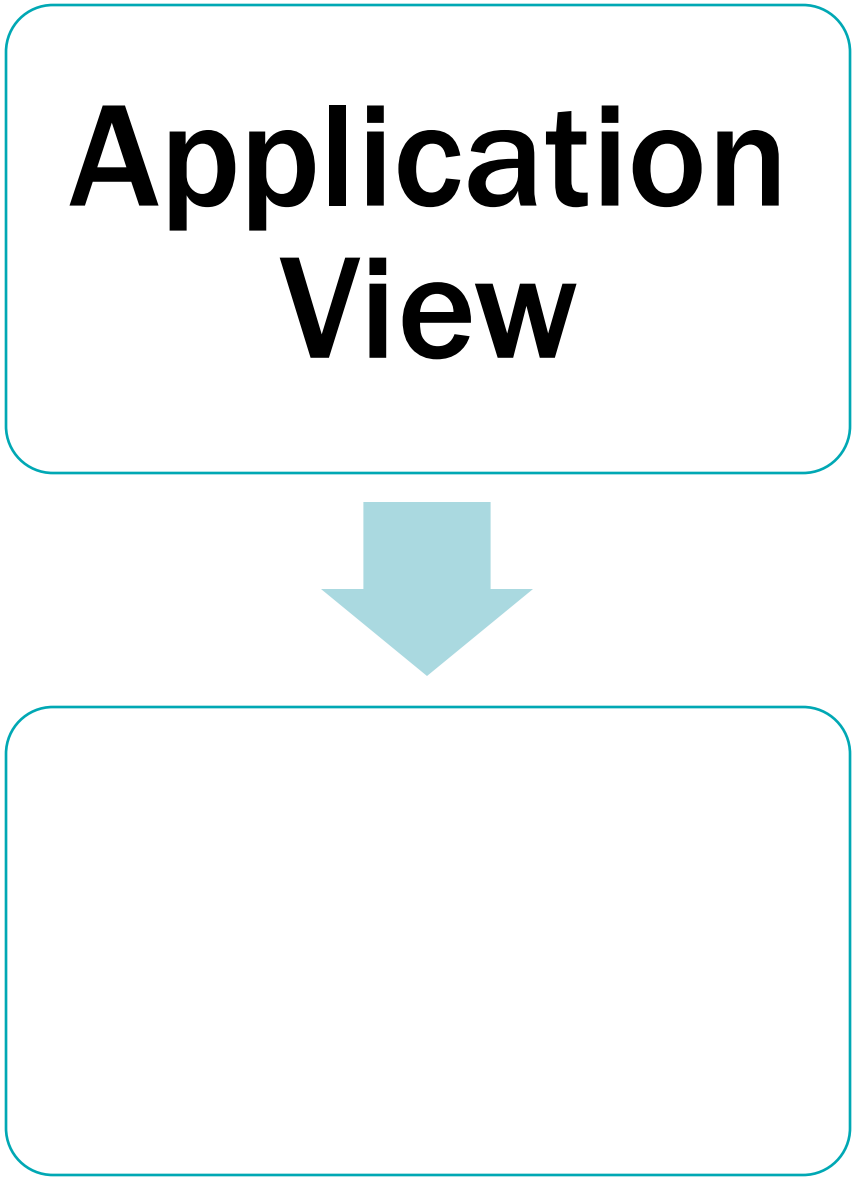
DISCUSSION THEMES

- If you are building a system, which testing techniques are useful?

All testing techniques are useful as long as they are used in the right stage of the system development

CLOUD INFRASTRUCTURE EFFICIENCY

...COURSE SO FAR

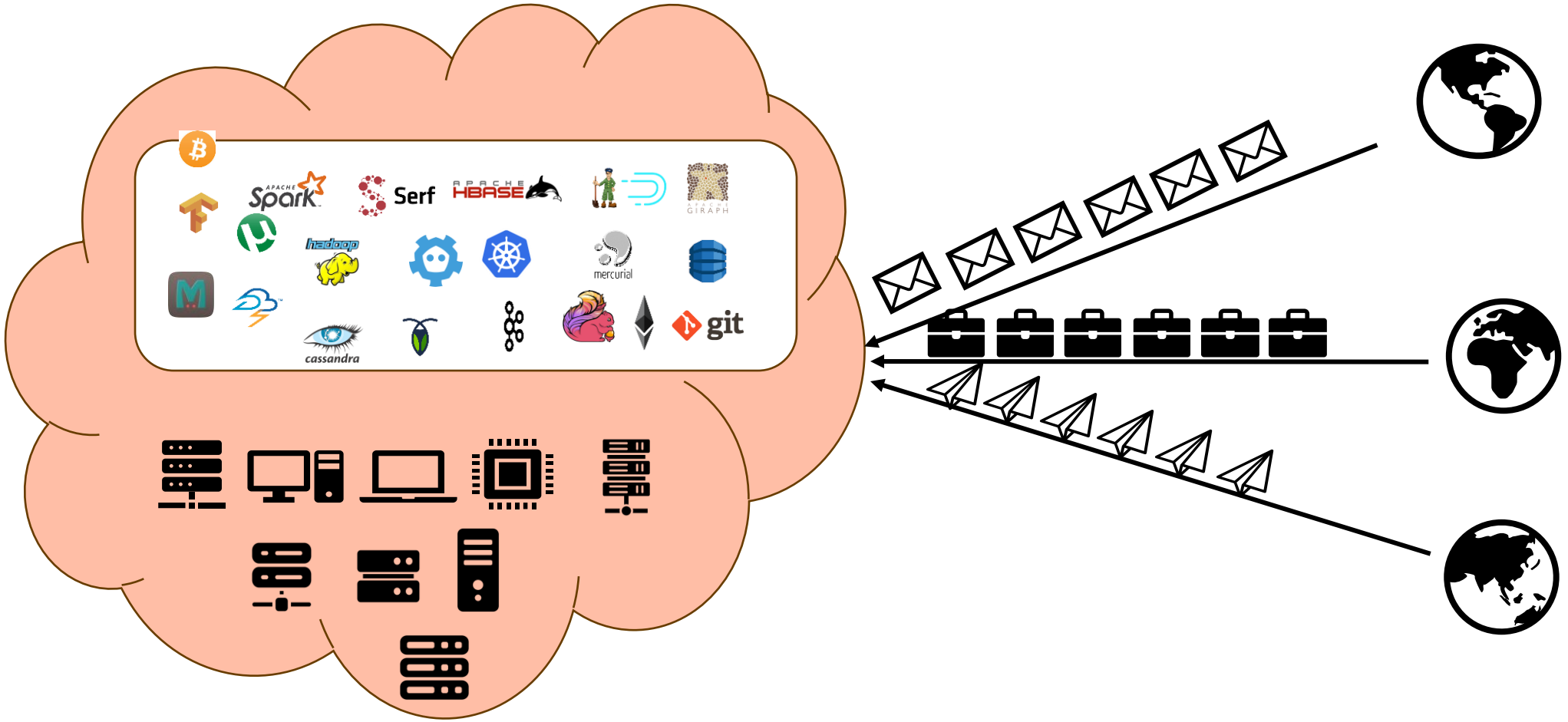


LAST 3 LECTURES

Application View

Cloud Provider View

CLOUD TODAY



CLOUD POV: APPLICATION IS THE WORKLOAD

- Applications are deployed as a set of VMs or maybe containers
- Require resources
  - CPUs
  - Memory
  - Bandwidth
  - Accelerators (GPUs, TPUs, ASICs, FPGAs)

CLOUD POV: APPLICATION IS THE WORKLOAD

- Applications are deployed as a set of VMs or maybe containers
- Require resources
  - CPUs
  - Memory
  - Bandwidth
  - Accelerators (GPUs, TPUs, ASICs, FPGAs)

How to allocate resources for an application?

RESOURCE ASSIGNMENT + ALLOCATION

- Resource Allocation: How many units of a resource does an application need
  - Eg: 20 vCPUs, 4GB RAM
- Resource Assignment: Which physical servers will service this workload
  - Eg: the 20 vCPUs will be deployed on Server A

RESOURCE UTILIZATION

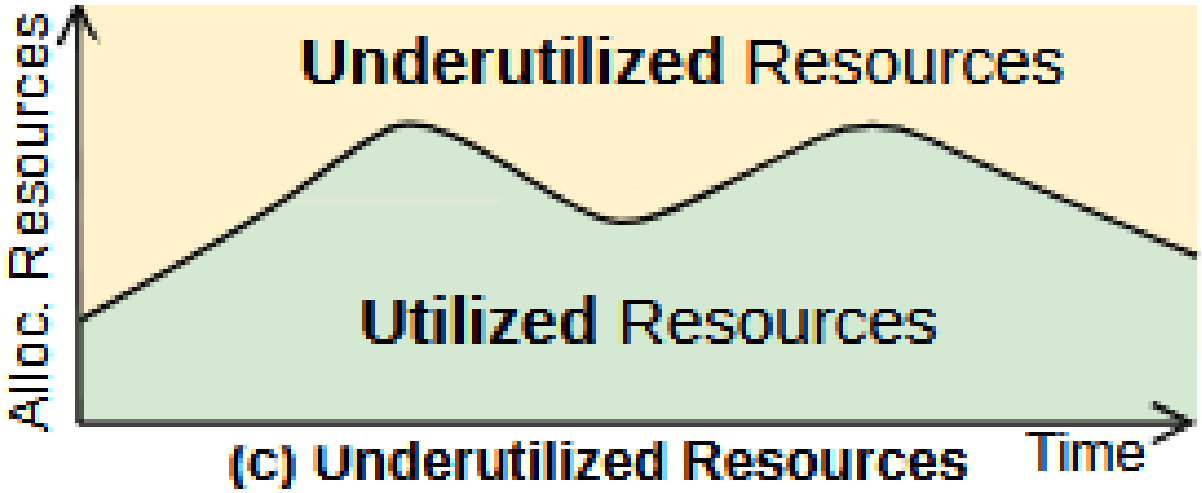
Goal: Infrastructure providers want to maximize the utilization of their resources

RESOURCE UTILIZATION

Goal: Infrastructure providers want to maximize the utilization of their resources

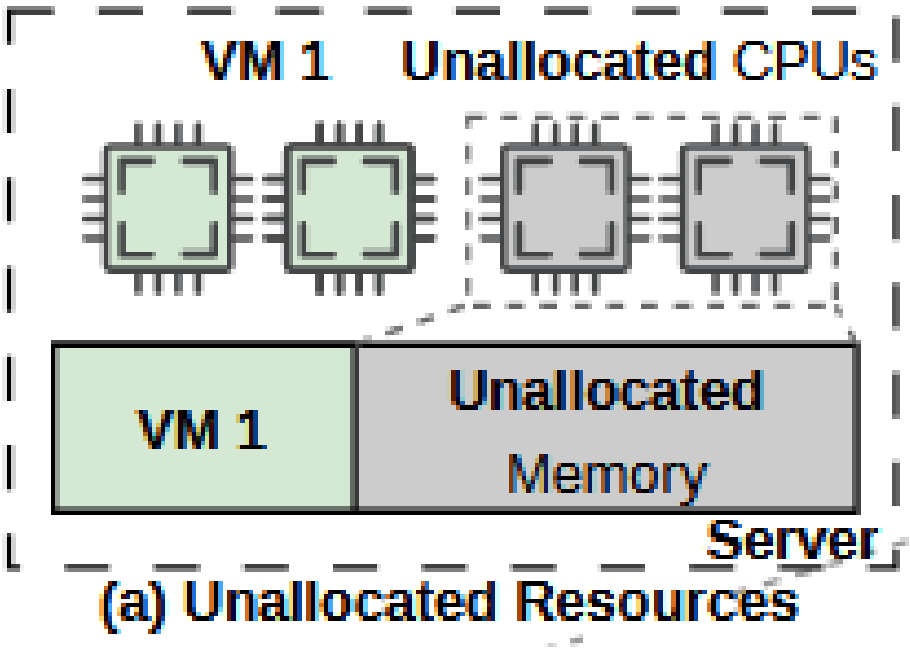
- Fit more workloads onto the infrastructure
- Improves Efficiency of the infrastructure
- Better overall performance
- Improved energy efficiency and sustainabilit
- Financially better for the cloud

RESOURCES ARE UNDERUTILIZED



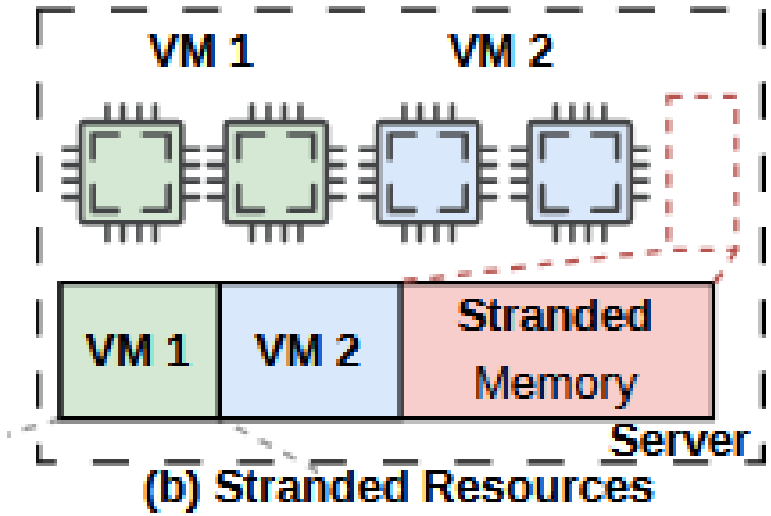
UNALLOCATED FOR PERFORMANCE

- No resource contention
- Best for applications as they don't have to share resources



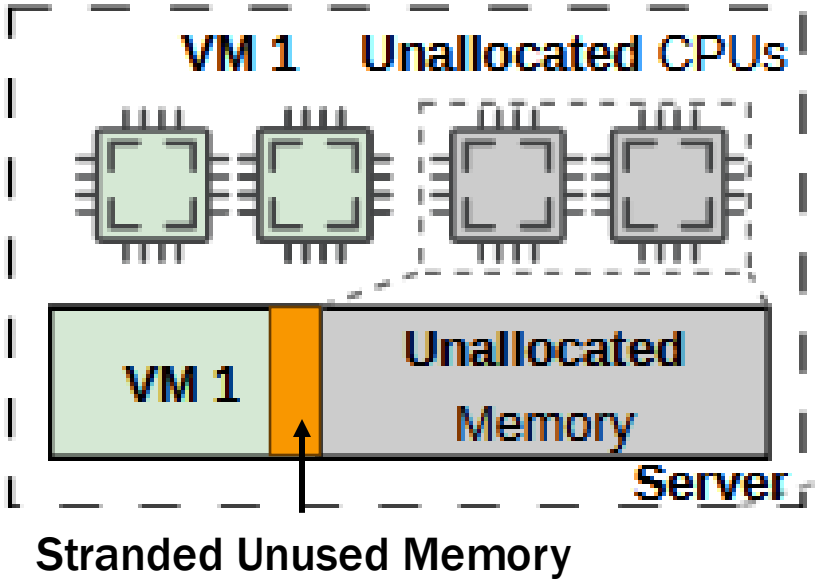
STRANDED RESOURCES I

- One resource on a machine maybe max utilized
- Other resource maybe stranded as it can't be allocated



STRANDED RESOURCES II

- Application are conservative in their resource requests
- They usually ask for more than they need;
  - Extra resources are unused but can't be reclaimed



RESOURCE UTILIZATION OPTIMIZATION TECHNIQUES

AUTO-SCALING

- Start the application with minimum amount of resources
- Only allocate and assign more resources based on the workload

AUTO-SCALING

- Start the application with minimum amount of resources
- Only allocate and assign more resources based on the workload
- Two Types:
  - Vertical Autoscaling: Add more resources (eg: more CPU cores or memory)
  - Horizontal Autoscaling: Scale the number of replicas for a service

RIGHTSIZING

- Clouds continuously monitor the utilization
- They can find VMs/services that are underutilized and recommend a smaller type of the VM/services that takes less resources
- Possible way for reclaiming the unused stranded resources
  - Combined with autoscaling

OVERSUBSCRIPTION

- Assign more virtual resources to a server than it can physically handle
- Key Idea: Not all applications will want their max capacity at the same time
- Eg: Server A has 48 CPUs, 16GB RAM
  - App A is assigned 24 vCPUs, 4GB RAM
  - App B is assigned 24 vCPUs, 8GB RAM
  - App C is assigned 12 vCPUs, 4 GB RAM

OVERSUBSCRIPTION

- Eg:: Server A has 48 CPUs, 16GB RAM
  - App A is assigned 24 vCPUs, 4GB RAM
  - App B is assigned 24 vCPUs, 8GB RAM
  - App C is assigned 12 vCPUs, 4 GB RAM
- We assigned apps totalling 60 vCPUs when there are only 48 physical CPU cores available -> Server A is oversubscribed
- Eviction if all apps peak at the same time

COACH VMS

- New type of VM that has a guaranteed capacity + an oversubscribed capacity
- Oversubscribed capacity comes from a sharing pool

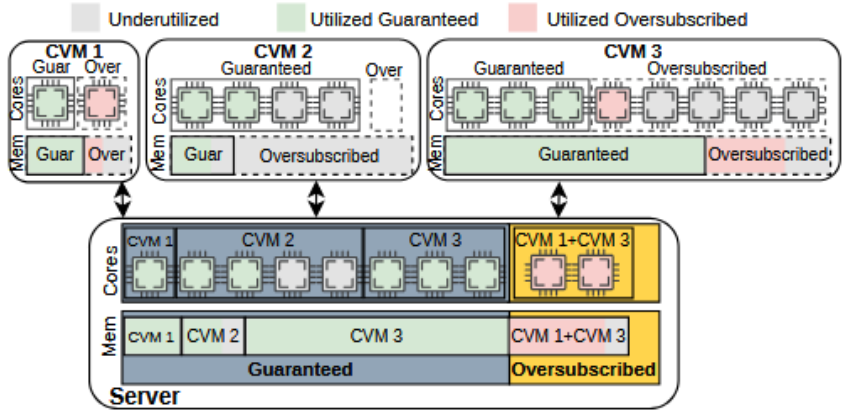


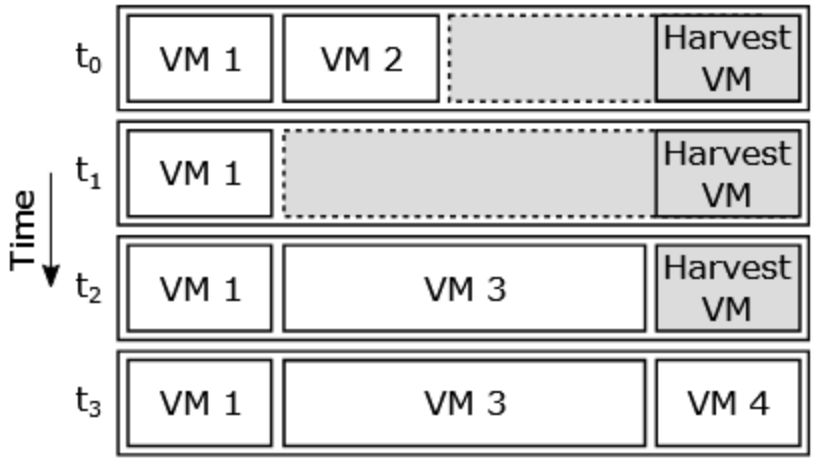
Figure 14. Three CoachVMs in a server showing the guaranteed and oversubscribed CPU and memory.

SPOT VMS

- Clouds have unallocated and unused resources
  - The servers are already running and drawing a large amount of power
  - Leaving them idle is bad
- Offer these resources at discounted rates through fixed size VMs
- Improve the utilization and gets cloud some more money
  - Evictable if a regular VM/app needs to be deployed

HARVEST VMS

- Like Spot VMs but they don't have a fixed size
  - Capacity changes over time
- Instead of using a fixed amount of resources, they flexibly use all the available unused resources in the machine



DISCUSSION THEMES

- Resource utilization is one key goal of cloud infrastructure providers? What are some other goals that cloud providers care about?

- How to decide the resource assignment for different workloads?