Outline

• What is dynamic consolidation?
  • Background and challenge
• Why is postcopy live migration promising?
  • Comparison between postcopy/precopy
  • Our postcopy live migration implementation
• Reactive consolidation system
  • Overall design
  • Packing algorithm
• Evaluation
• Conclusion
Dynamic Consolidation

- Dynamically optimize VM locations in response to VM load changes in order to
  - Eliminate excessive power
  - Assure VM performance
Dynamic Consolidation

When datacenter load becomes small,

- Move running VMs into fewer hosts
- Suspend idle physical hosts
- Eliminate excessive power

Reduce power consumption
Dynamic Consolidation

When datacenter load becomes high,

- Distribute running VMs into other hosts
- Power up new physical hosts

Assure VM Performance

The CPU usage of physical machines

Remove overload
Challenge

• Be transparent to IaaS customers
  • IaaS providers present performance criteria
    • EC2 Small Instance: 1.0-1.2GHz 2007 Opteron
  • Live migration incurs
    • Long duration until completed
    • CPU Overheads

Hide what’s going on in the background as much as possible!
Contribution

- **Reactive consolidation by postcopy live migration**
  - Develop postcopy live migration for Qemu/KVM
    - Presented in CCGrid2010
  - Develop a reactive VM consolidation system
    - Optimize VM locations in response to load changes
    - Exploit postcopy live migration for quick load balance
    - Presented in this talk

- **Assure VM performance in a higher degree**
  - Reduce performance loss on VM repacking
    - 50% better in a randomly-generated scenario
Outline

- What is dynamic consolidation?
  - Background and challenge
- Why is postcopy live migration promising?
  - Comparison between postcopy/precopy
  - Our postcopy live migration implementation
- Reactive consolidation system
  - Overall design
  - Packing algorithm
- Evaluation
- Conclusion
Precopy v.s. Postcopy

- **Precopy live migration**
  - Copy VM memory *before* switching the execution host
  - Widely used in VMMs

- **Postcopy live migration**
  - Copy VM memory *after* switching the execution host
  - No publicly-available implementation
    - But, we developed it!
Precopy Live Migration (1)

*Copy VM memory before relocation*

- Ex. VM with 1 GB RAM
  - Takes 10 seconds at least with GbE
  - May take more and more.

1. Copy all memory pages to destination
2. Copy memory pages updated during the previous copy again
3. Repeat the 2\textsuperscript{nd} step until the rest of memory pages are enough small
4. Stop VM
5. Copy CPU registers, device states, and the rest of memory pages.
6. Resume VM at destination
Precopy Live Migration (2)

*Copy VM memory before relocation*

- Ex. VM with 1 GB RAM
  - Takes 10 seconds at least with GbE
  - May take more and more.

1. Copy all memory pages to destination
2. Copy memory pages updated during the previous copy again
3. Repeat the 2\(^{nd}\) step until the rest of memory pages are enough small
4. Stop VM
5. Copy CPU registers, device states, and the rest of memory pages.
6. Resume VM at destination

\[
\text{Migration time} = \frac{\text{RAM size}}{\text{Network speed}} + \alpha
\]
Precopy Live Migration (3)

*Copy VM memory before relocation*

- Ex. VM with 1 GB RAM
  - Takes 10 seconds at least with GbE
  - May take more and more.

1. Copy all memory pages to destination
2. Copy memory pages updated during the previous copy again
3. Repeat the 2\textsuperscript{nd} step until the rest of memory pages are enough small
4. Stop VM
5. Copy CPU registers, device states, and the rest of memory pages.
6. Resume VM at destination

\[
\text{Migration time} = \frac{\text{RAM size}}{\text{Network speed}} + \alpha
\]

- It takes long time to switch location.
- It is difficult to estimate how long it takes.
Postcopy Live Migration (1)

Copy VM memory after relocation

1. Stop VM
2. Copy CPU and device states to destination
3. Resume VM at destination
4. Copy memory pages
Postcopy Live Migration (2)

Copy VM memory after relocation

1. Stop VM
2. Copy CPU and device states to destination
3. Resume VM at destination
4. Copy memory pages

Copy CPU and device states
Only 256KB w/o VGA
=> Less than 1 sec for relocation
Postcopy Live Migration (3)

*Copy VM memory after relocation*

1. Stop VM
2. Copy CPU and device states to destination
3. Resume VM at destination
4. Copy memory pages
Postcopy Live Migration (4)

Copy VM memory after relocation

1. Stop VM
2. Copy CPU and device states to destination
3. Resume VM at destination
4. Copy memory pages

Copy memory pages
- On-demand
- Background
Demo (Precopy v.s. Postcopy)

Precopy Live Migration

Postcopy Live Migration
Problems in Prior Consolidation Studies

- All prior studies are based on precopy live migration.
  - To tackle a long migration time, prior studies use load prediction techniques.
  - Repacking timescale ~ hour/day (e.g., business hour).
- However, IaaS datacenters allow users to run any kinds of workloads at any time.
  - Precise prediction is difficult because we cannot use workload specific algorithms.
  - Nobody cannot predict sudden load changes.
Reactive Consolidation

- Exploit postcopy live migration
- Reactively optimize VM locations in response to load changes.
- Repacking timescale ~ 10 seconds

Overloaded → Detect → Switch the execution host (1 sec) → Complete (T = Ramsize / Bandwidth)
Packing Algorithm (1)

- Realistic design
  - Lightweight, near-optimal
- Define two server types
  - Shared Server (with large RAM, expensive)
    - Consolidate many idle VMs
    - Always power on
  - Dedicated Server
    - Host an actively-running VM
    - Suspend when unused
Packing Algorithm (2)

If overloaded, an active VM pops out from Shared Server. Pop-Out Threshold = over 90% CPU usage of Shared Server.
Packing Algorithm (3)

If a VM becomes idle and Shared Server has space for it, the VM returns to Shared Server. Idle Threshold = under 50% usage of Dedicated Server.
Evaluation

- Experiments
  - Simple load change scenario
    - Pure CPU-intensive workload
    - Memory-intensive workload
  - Compound load change scenario
  - A new benchmark program
    - Metric for performance assurance
    - Generate a target CPU load with a specified memory update intensity
    - Measure achieved operations per second
      - Failed ops = Target ops – Achieved ops
Simple Load Change Scenario

- 6VMs
- VM0 and VM1 become active.
• How it basically works.
  1. Detect overloading and optimize locations.
     • VM0 is moved to Dedicated Server 4
  2. Detect idle state and optimize location.
     • VM0 is moved back to Shared Server
Using Postcopy

Overload is removed in less than 10 seconds.

Using Precopy

Overload is removed in more than 20 seconds.

Shared Server’s CPU Usage (%)

VMo cannot get enough CPU resource. This results in Failed operations.
Failed Operations per Second
(Red: VM0, Green: VM1)

Using Postcopy

Using Precopy

- Postcopy alleviates the number of failed operations.
  - 20000(ops) -> 5000 (ops)
  - Note that 4000 (ops) is detection overhead (avg. 5sec).
- In Poscopy, performance loss until completed is small, i.e., 1000 (ops); this workload is pure CPU-intensive 😊
Failed Operation per Second v.s. Memory Update Intensity

Using precopy incurs large performance penalty for memory-intensive workloads.

Using postcopy greatly contributes to reducing packing overheads.

Detection overhead

1 GB/s at 100% CPU
Compound Load Change Scenarios (1)

- **One-hour, randomly-generated load changes**
  - Emulate race-to-halt workloads
    - Mostly idle, but sometimes active

- An active VM consumes a random CPU load between 80% and 100%.
- An idle VM consumes a random CPU load between 0% and 20%.

- A new state continues for a random duration between 60 and 300 seconds.
- Memory update intensity is 0.6 (600MB/s on 100% CPU usage).
By using postcopy migration, failed operations are reduced to the half level of using precopy.
Compound Load Change Scenarios (3)

The number of live migrations during each scenario

The consolidation system using postcopy frequently optimizes VM locations.
Related Work

- Postcopy live migration
  - SnowFlock (live VM cloning), EuroSys2009
  - Postcopy live migration (Xen), VEE2009
  - Postcopy live migration (Qemu/KVM), CCGrid2010
    - By me 😊
- VM Consolidation System
  - Black-box v.s. Gray-box, NSDI2007
    - Need workload-specific information for better prediction
  - Genetic-Algorithm, DCAI2009
    - Quickly find near-optimal locations
    - By us
Conclusion

• Reactive VM consolidation System
  • Keeps performance criteria of VMs as much as possible in IaaS datacenters
  • Does not reply on load prediction
  • Reactively optimizes VM locations in response to load changes
  • Exploits postcopy live migration for quick load balance

• Evaluation
  • better performance assurance than using precopy live migration, especially for memory-intensive workloads
“Yabusame (流鏑馬) is a type of mounted archery in traditional Japanese archery. An archer on a running horse shoots three special turnip-headed arrows successively at three wooden targets.” from wikipedia.

Photo: © Yuki Shimazu 2011
http://www.flickr.com/photos/shimazu/5631324478/