Cluster-Wide Context Switch of Virtualized Jobs

Fabien Hermenier, Adrien Lèbre, Jean-Marc Menaud

VTDC’10, 22 June 2010

ASCOLA Team

ECOLE DES MINES DE NANTES

INRIA
Agenda

Motivation

Global Design
  Architecture
  Implementation

Proof of concept
  A sample scheduler
  Experiment on a cluster

Conclusion
Motivation

Agenda

Motivation

Global Design
  Architecture
  Implementation

Proof of concept
  A sample scheduler
  Experiment on a cluster

Conclusion
Motivation

Clusters

- large infrastructures to execute various jobs

Resource Management System (RMS)

- manage the execution of jobs
- resources are allocated to jobs according to their description
- scheduling: which jobs to execute, and where?
Motivation

Jobs schedulers

Usually
A coarse-grain exploitation of resources:
- static allocation of resources
- execution to completion

Dynamic schedulers exist
Based on mechanisms that manipulate the jobs dynamically (migration, preemption, dynamic allocation of resources, ...). BUT
- mechanisms are complex to implement
- mechanisms are complex to use efficiently
Motivation

Virtual Machines (VMs) as a backend for dynamic schedulers

▶ each component is embedded into its VM
▶ VMMs provide migration, preemption
▶ still complex to use efficiently

A cutting-edge building block

dynamic consolidation, best-effort jobs, ... 

▶ various policies, but common concepts to perform the changes
▶ each provides an ad-hoc solution to handle several common issues:
  ▶ dependencies between actions
  ▶ correctness
  ▶ reactivity
Proposition

Performing the changes should not be a primary concern for developers

▶ a generic cluster-wide context switch based on VMs
▶ developers only focus on the algorithm to select the jobs to run
▶ the cluster-wide context switch takes care of the rest
   ▶ detects the changes to perform
   ▶ ensures the correctness of the transition
   ▶ computes the fastest possible transition

The implementation leverages the consolidation manager Entropy
Agenda

Motivation

Global Design
   Architecture
   Implementation

Proof of concept
   A sample scheduler
   Experiment on a cluster

Conclusion
From jobs to **virtualized Jobs**

- a vjob encapsulates one or several VMs
- to change the state of a vjob, actions (except migrate) are executed on each VMs

**Figure:** The life cycle of a vjob
Configuration

- describes the assignment of the running VMs to working nodes
- nodes provide CPU and memory resources
- running VMs require CPU and memory resources to run at peak level

(a) Non-viable configuration

(b) Viable configuration
The control loop of Entropy

Monitor

- extract the current configuration: VM position, CPU/memory consumption
- adaptable to a specific monitoring system (currently Ganglia)
The control loop of Entropy

Scheduling policy
- an algorithm to select the vjobs to run wrt. the current configuration
- provided by a developer
The control loop of Entropy

- Scheduling policy
- CW Context switch module
- Monitoring
- Exécution

▶ selects a position for each VM to run
▶ infers the actions that make the transition w. the current configuration
▶ computes the fastest plan that ensure the correctness of the process

The cluster-wide context switch module
The control loop of Entropy

Execution

- associate each action of the plan with a driver that performs the action
  adaptable to specific environments.
  Currently support Xen VMM (XML-RPC) or shell command
Role of the CW context switch

- detects the actions to perform
- selects a position for each VM to run
- plans the actions to guarantee the correctness of the process
- computes the fastest possible plan
Plan the actions
Plan the actions

current configuration

Step 1
Plan the actions

Current configuration

Step 1

Step 2
Plan the actions

The reconfiguration plan

- a protocol to execute actions
- actions feasible in parallel are grouped into a same step
- steps are executed sequentially
Suspending/Resuming a vjob

- inter-connected VMs should be continuously in the same state
- coordination to ensure that distributed applications will not fail
Suspending/Resuming a vjob

- inter-connected VMs should be continuously in the same state
- coordination to ensure that distributed applications will not fail

- actions are grouped into a same step
- synchronization between the pause/unpause actions
Reducing the duration of a cluster-wide context switch

- the duration of an action depends on its context
- a function estimates the cost of a whole CW context switch
Reducing the duration of a CW context switch

An approach based on constraint programing

Entropy computes a new configuration that
- is viable
- respects the scheduling policy
- implies the minimal cost

In practice
- actions are performed asap.
- prefer moving VMs with small memory requirements
- avoid migrations and remote resumes
Proof of concept

Agenda

Motivation

Global Design
  Architecture
  Implementation

Proof of concept
  A sample scheduler
  Experiment on a cluster

Conclusion
A sample scheduler

Principle

- a FIFO queue
- VMs are assigned to nodes using a First Fit Decrease heuristic
- priority between jobs to prevent starvation

Example

![Diagram showing VM allocation on nodes with CPU and memory capacity axes.](image-url)
A sample scheduler

Principle

- a FIFO queue
- VMs are assigned to nodes using a First Fit Decrease heuristic
- priority between jobs to prevent starvation

Example
A sample scheduler

Principle

- a FIFO queue
- VMs are assigned to nodes using a First Fit Decrease heuristic
- priority between jobs to prevent starvation

Benefits using CW context switch

- dynamic allocation of resources
- preemption
- migration of VMs
Environment

Hardware

- 11 working nodes
- 3 storage nodes share VM images
- 1 service node is running Entropy

Protocol

- a queue of 8 vjobs (NASGrid benchmarks)
- each vjob uses 9 VMs
- comparison with regards to FCFS
  - resources usage
  - completion time
Experiment on a cluster

Benefits

▶ improve resource usage
▶ suspend/resume transparent for the developer

Resources usage
Experiment on a cluster

Benefits

▶ improve resource usage
▶ suspend/resume transparent for the developer
▶ reduce the completion time

Cumulated execution time

▶ FCFS: 250 minutes
▶ Entropy: 150 minutes
Agenda

Motivation

Global Design
  Architecture
  Implementation

Proof of concept
  A sample scheduler
  Experiment on a cluster

Conclusion
Conclusion

RMSs start to manage VMs instead of process

- VMMs provide mechanisms to implement dynamic schedulers
- Manipulate VMs is tedious and may be non cost-effective
- Various scheduling policies but common concepts to perform the context switch

A generic cluster-wide context switch

- Make the implementation of dynamic schedulers easier
- The context switch is outside the scheduling algorithm
- An implementation in Entropy with a sample algorithm

http://entropy.gforge.inria.fr
version 1.2 (LGPL)
I’m looking for a postdoc position

- fond of - virtualization, distributed systems, autonomic computing, ...
- dislike - tomatoes