HPC-related Cloud Computing Research at the ANU

Peter Strazdins
Computer Systems Group,
School of Computer Science,
The Australian National University

Taiwan Universities Delegation,
ANU, 03 November 2011

1 Talk Overview

- optimizing live migration and communication performance of Xen
- a dynamic scheduling framework for heterogeneous clouds
- a resilient and heterogeneity-oblvious HPC programming paradigm
2 Optimizing Migration & Communication Performance of Xen

- new Xen Bridge architectures for communication using multiple NICs
- hybrid shared-separate bridge configuration much faster than default
  - almost as good as under native Linux (XenHPC/EurSys’09 paper)
- communication overhead much less if one core available for hypervisor
  - ⇒ (future) heterogeneous multicore design to support virtualization

- reduce VM live migration cost for running HPC applications
- optimize by reducing the number of page copying cycles from 30 (minimizes downtime) to 2 (minimizes walltime stretch) (NPC’09 paper)
3 Dynamic Scheduling for Heterogeneous Compute Farms

- **ARRIVE-F**: a open-source profiling/dynamic resource scheduling framework for (Xen) virtualized heterogeneous compute farms
- Muhammad Atif’s PhD thesis, Nov’10
- profiles execution of each (MPI) parallel job (using PMPI & Oprofile), and migrates jobs to more appropriate sub-clusters
  - note: binary compatibility across sub-clusters is required
4 Dynamic Scheduling Framework: Results

- performance estimation model has > 80% accuracy
- overall throughput improvement of 25% on a 32 VM compute farm comprised of homogeneous sub-clusters
  - 2–3 GHz CPUs, 100Mb or 1Gb network
- workloads of 200 jobs from the NAS Parallel Benchmarks
- advantages: no prior knowledge of job characteristics required, transparent to users

![Diagram of typical migration scenario: swap jobs \(j\) and \(k\)]
5 A Dynamic Scheduling Framework for Heterogeneous Clouds

- problem: compute resources within (and across) cloud providers tend to be heterogeneous
  - workloads may not be allocated to the most appropriate resources
- opportunity: develop Arrive-F for clouds!
  - within a cloud provider (also generalize for use in data centers)
  - between cloud providers. A motivating scenario:
    - initially run and profile job on NCI/NF cloud (like the current vayu cluster)
    - migrate jobs with “undesirable’ profiles to (cheap) cloud providers
    - requires a Lustre-like implementation on a cloud
- issues to be addressed:
  - take VM creation into account; generalize supporting infrastructure
  - take energy into account; also use dynamic power management?
  - migration decisions directed/constrained by Service Agreements
  - potentially longer data transfer times / latencies
A Resilient and Heterogeneity-oblivious HPC Programming Paradigm

Motivation:
- it is hard to write traditional parallel algorithms (e.g. in MPI) for heterogeneous systems
  - harder still in presence of faults, dynamically changing loads!
- this problem has been solved for (embarrassingly-parallel) grid applications, using a ‘service-oriented’ approach (e.g. in computational finance)
  - can this be applied to general HPC scientific applications?

Our approach (with Jaison Mulerikkal (PhD) and Platform Computing):
- extend the Platform Symphony API for HPC workloads (ANU-SOAM)
- with integrated compute and data services; communication provided by the latter
7 ANU-SOAM: Architecture

client task:
  initialize common data
  repeat
    spawn a set of service tasks
    synchronize common data
    extract results from common data

service task:
  get portion of common data
  compute on common data
  put updated common data

- put of common data not visible until synchronize
- Service Instance process executes a number service tasks in each set
  - enables (dynamic) load balancing with heterogeneous compute speeds
8 Extending ANU-SOAM for Resilience, the Cloud

- improve existing infrastructure (rest of Jaison’s PhD thesis)
  - add concept of local and partially replicated data
    - retained on Service Instance processes for the next set of tasks (needs notion of ‘affinity’)
  - refine scheduling
- fault tolerance: simply restart last set of service tasks
  - needs also redundancy in common data service
- investigate effect on clouds
  - higher latencies, higher variability in performance
  - challenge: keep the overheads of the infrastructure low!