Exploring adaptation to support dynamic applications on hybrid grids-clouds infrastructure

Hyunjoo Kim[1], Yaakoub el-Khamra[2], Shantenu Jha[3], Manish Parashar[1]

[1]NSF Center for Autonomic Computing, Dept. of Elect. & Computer Eng., Rutgers, The State Univ. of New Jersey, USA

[2]Texas Advanced Computing Center, Austin, Texas, USA

[3]Center for Computation and Tech. and Dept. of Computer Science, Louisiana State Univ., USA

Center for Autonomic Computing

Motivation

- Clouds support a different although complementary usage model to more traditional HPC grids
  - Cross-over applications from the legacy & cluster world
  - Cyclone – SGI's latest cloud offering

- Imperative questions
  - Application types and capabilities that can be supported by clouds?
  - Can the addition of clouds enable scientific applications and usage modes, that are not possible otherwise?
  - What abstractions and systems are essential to support these advanced applications on different hybrid grid-cloud platforms?

Objectives

- To address motivated questions in the context of dynamic applications
  - Dynamic applications are able to adapt at the application level (dynamic formulation) and at the infrastructure utilization level (dynamic execution)
- To establish and analyze an interesting case study for a sophisticated scientific application that benefits from a hybrid HPC grid-cloud execution environment
- To investigate application-infrastructure adaptivity, how it can be supported on hybrid infrastructure and the subsequent performance advantages

Background

- Autonomic objectives for the integration of HPC grids and clouds (eScience 2009)
  - *Acceleration*: explores how clouds can be used as accelerators to improve the application time-to-completion
  - *Conservation*: investigates how clouds can be used to conserve HPC grid allocations
  - *Resilience*: investigates how clouds can be used to handle unexpected situations such as an unanticipated HPC grid downtime, inadequate allocations or unanticipated queue delays

Exploring adaptation

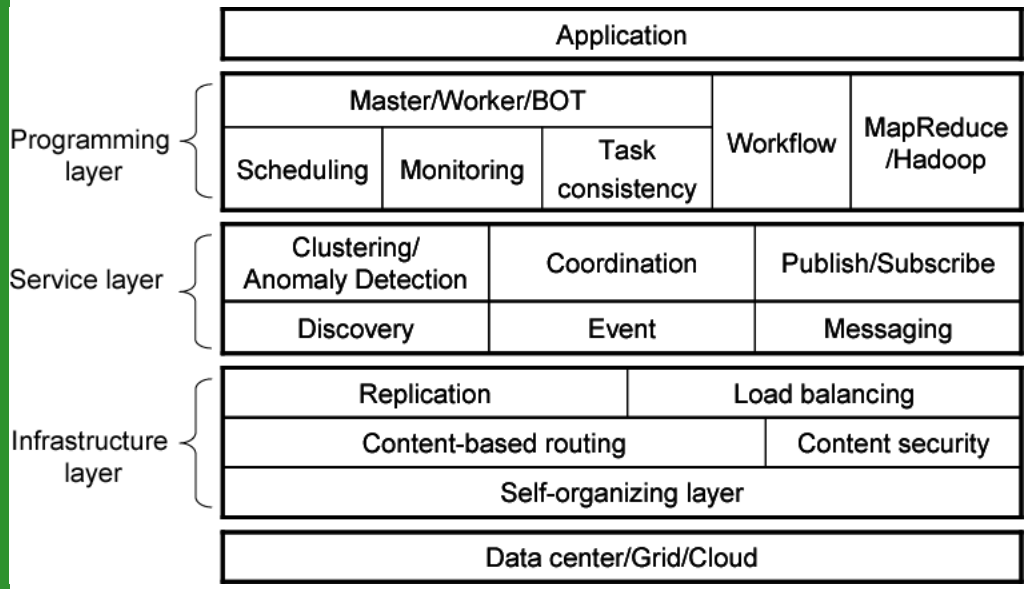
- Three approaches to adapting computational science applications based on acceleration objective
  - Track1: selection and adaptivity of infrastructure
  - Track2: tuning of applications
  - Track3: adaptivity for both infrastructure and application
- Infrastructure adaptivity
  - Explores a richer infrastructure space and selects appropriate numbers of types (e.g., the number and type of virtual machines)
- Application adaptivity
  - Involves adapting the structure and behavior of the applications based on application/system characteristics (e.g., the size of ensemble members, problem size and application configuration) and runtime state

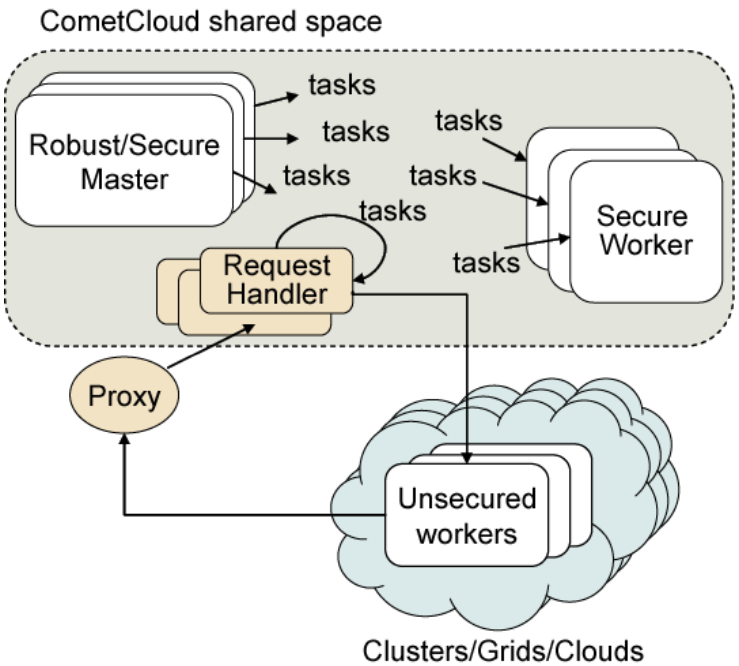
Application characterization

- The Reservoir Simulator
  - Solves the equations for multiphase fluid flow through porous media, allowing us to simulate the movement of oil and gas in subsurface formations. It is based on the Portable Extensible Toolkit for Scientific Computing: PETSc
- The Ensemble Kalman filter (EnKF)
  - The parallel EnKF computes the Kalman gain matrix and updates the model parameters of the ensembles. The Kalman filter uses production data from the reservoir for it to update the reservoir models in real-time, and launch the subsequent long-term forecast, enhanced oil recovery and CO_2 sequestration studies.

CometCloud

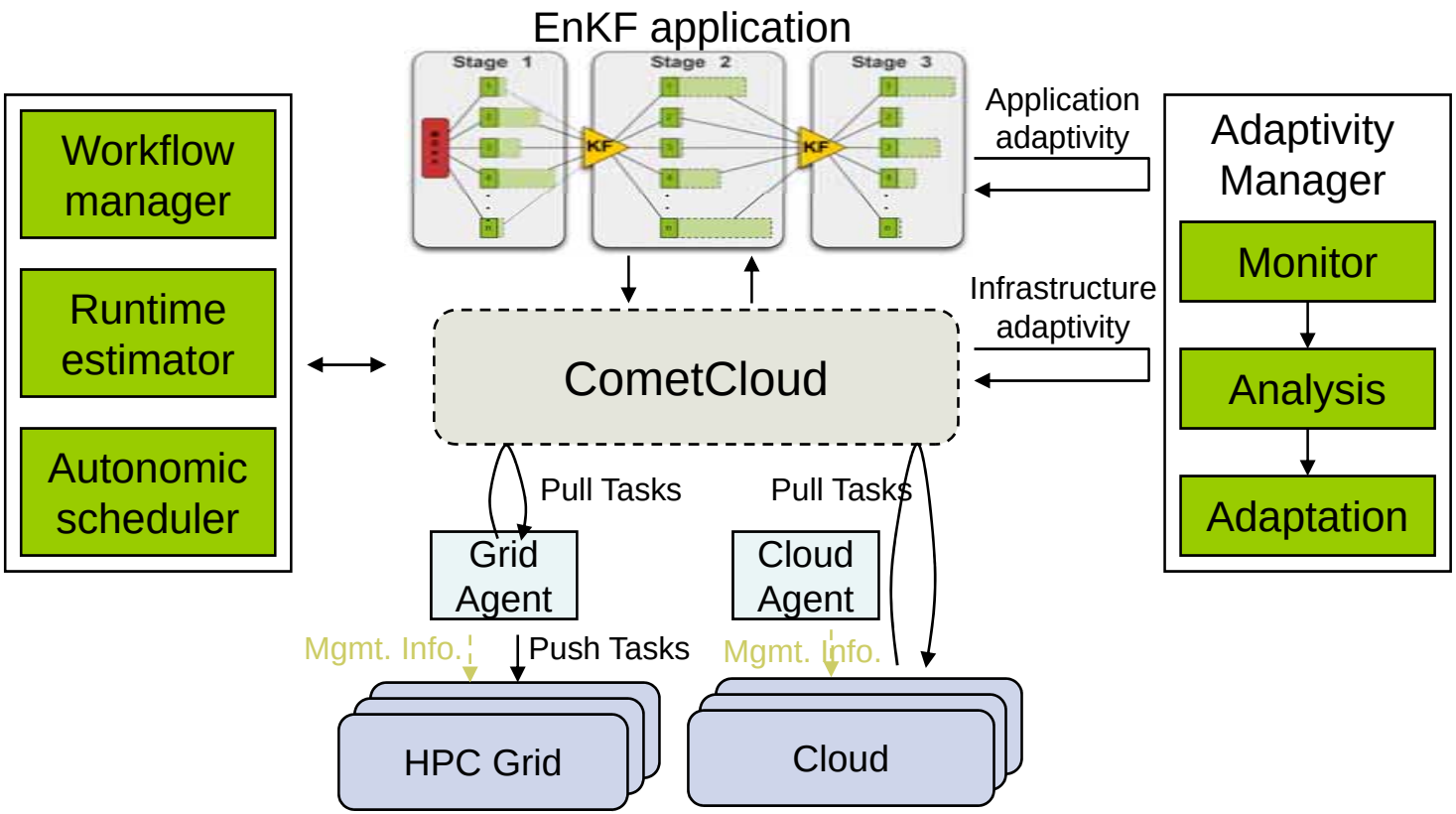
- An autonomic computing engine over hybrid computing infrastructure.
- Support on-demand bridging of public/private clouds and grids as well as autonomic cloudbursts





Dynamic execution using CometCloud

- Autonomic architecture for adaptivity



Experimental environments

- Three stages of the EnKF workflow with 20x20x20 problem size and 128 ensemble members with heterogeneous computational requirement
- Deploy EnKF on TG (16 cores) and several instance types of EC2 (MPI enabled)

Instance type	Cores	Memory(GB)	Platform(bit)	cost($/hour)
m1.small	1	1.7	32	0.1
m1.large	2	7.5	64	0.34
m1.xlarge	4	15	64	0.68
c1.medium	2	1.7	32	0.17
c1.xlarge	8	7	64	0.68

TABLE I: EC2 instance types used in experiments

Results – Baseline

- No adaptation is applied
- Resource classes: TeraGrid, c1.medium

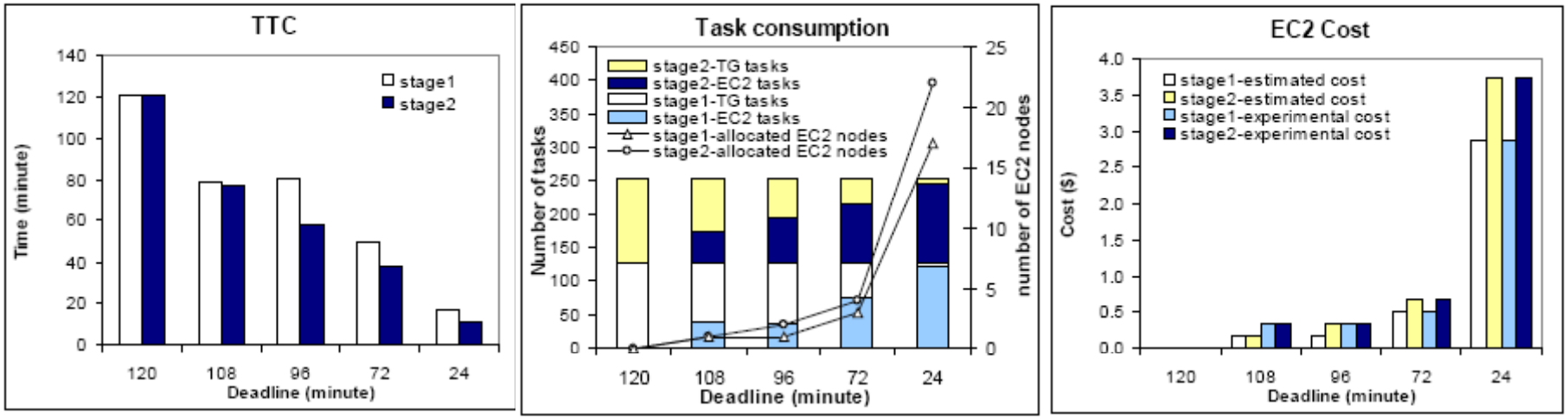


Figure 1: Baseline experiment without adaptivity with a deadline policy. Tasks are completed within a given deadline. The shorter the deadline, the more EC2 nodes are allocated.

Center for Autonomic Computing

RUTGERS THE STATE UNIVERSITY OF NEW JERSEY

THE UNIVERSITY OF ARIZONA

UF UNIVERSITY of FLORIDA

Results - Track 1

- Infrastructure adaptivity
- Resource classes: TeraGrid, 5 instance types of EC2

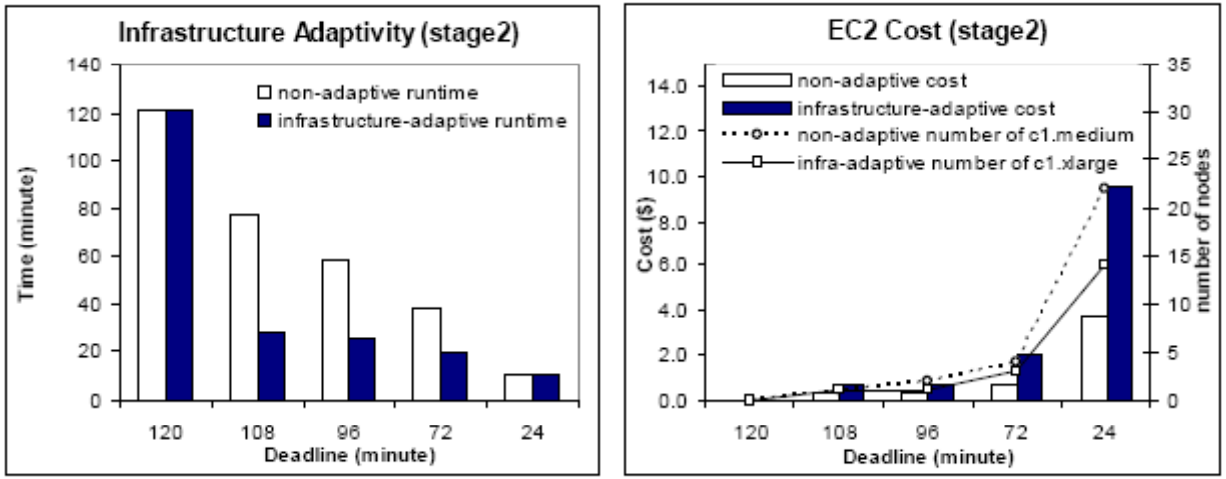


Figure 2: Experiments with infrastructure adaptivity. The TTC is reduced with infrastructure adaptivity at additional cost.

Center for Autonomic Computing

RUTGERS THE STATE UNIVERSITY OF NEW JERSEY

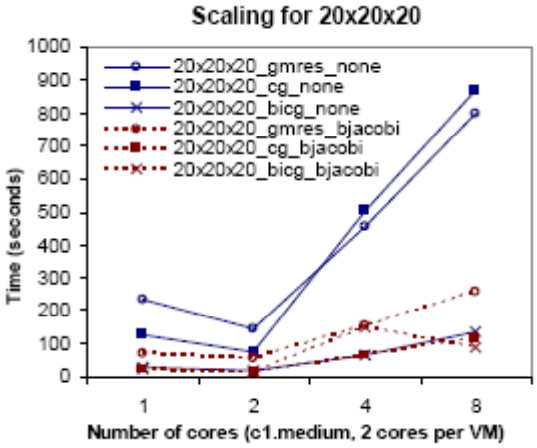
THE UNIVERSITY OF ARIZONA

UF UNIVERSITY of FLORIDA

Results – Track 2

- Application adaptivity
- Resource classes: TeraGrid, 5 instance types of EC2

Figure 3: TTC for simulations of 20x20x20 with different solvers (GMRES, CG, BiCG) and block-Jacobi preconditioner. Benchmarks ran on EC2 nodes with MPI



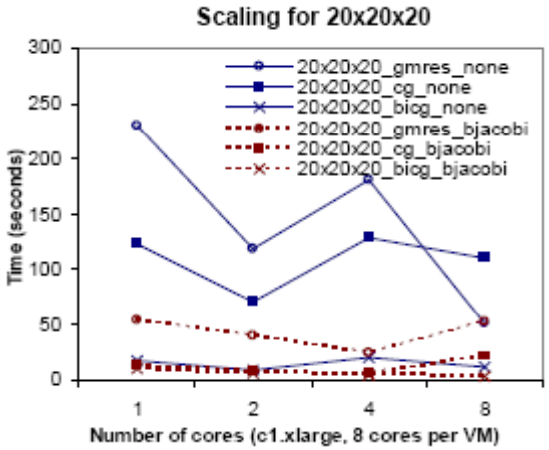
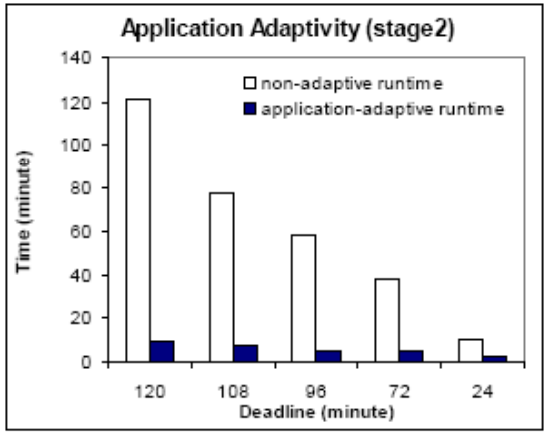
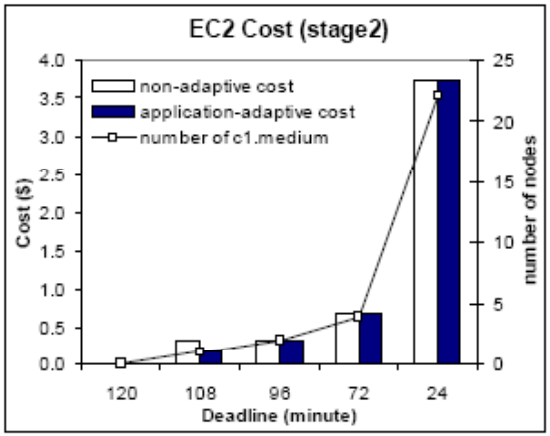


Figure 4: Experiments with application adaptivity. The TTC is reduced with application adaptivity for equivalent or slightly less cost.

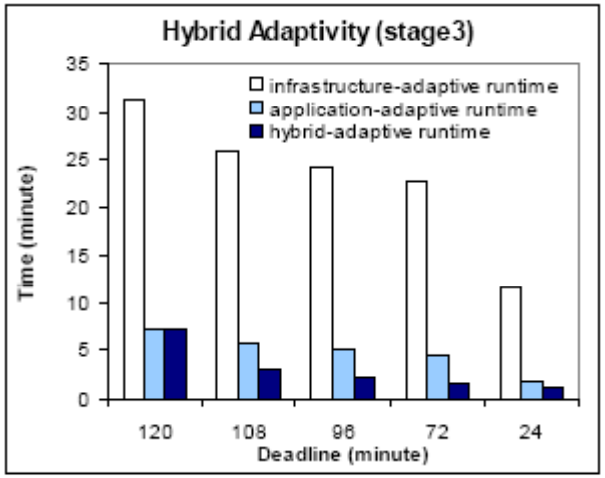




Center for Autonomic Computing

Results – Track 3

- Both infrastructure/application adaptivities
- Resource classes: TeraGrid, 5 instance types of EC2



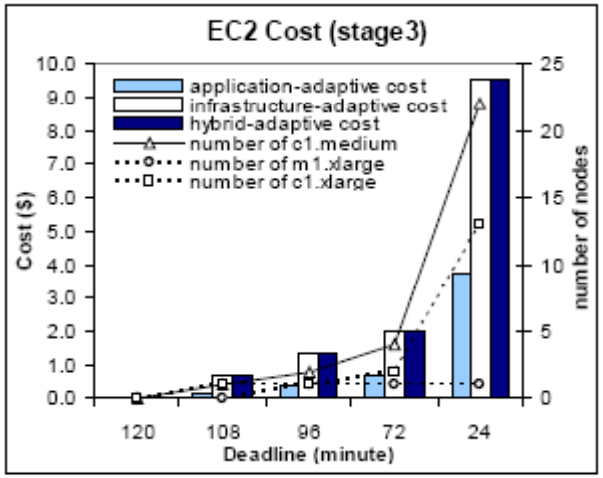


Figure 5: Experiment with adaptivity applied for both infrastructure and application. The TTC is reduced further than with application or infrastructure adaptivity on its own. The cost is similar to that in infrastructure adaptivity for durations less than one hour since EC2 usage is billed hourly with a one hour minimum.

Conclusion

- We established two objectives
  - 1) to build and analyze an interesting case study for a sophisticated scientific application that benefits from a hybrid HPC grids-clouds execution modes by exploring richer infrastructure as well as application space
  - 2) to investigate application or/and infrastructure adaptivity and how those adaptivities affect performance in terms of time-to-completion and cost.
- Experimental results show that TTC decreases even applying on adaptivity and more decreases applying both adaptivities with reasonable EC2 costs.