Computation Institute

Abstractions for Clouds

Daniel S. Katz (dsk@ci.uchicago.edu)
Senior Fellow, Computation Institute (University of Chicago & Argonne National Laboratory)
Associate Faculty, Center for Computation & Technology, Louisiana State University
Adjunct Associate Professor, Electrical and Computer Engineering, LSU

My experience (in brief)

- Thought about a EE-CS double major at Northwestern – took all but 3 CS classes – learned Unix, C
- Wrote parallel (message passing) electromagnetics code in 1987, on 16 node Intel 286 hypercube (iPSC/d4) (w/ 2 types of message passing, host-node and node-node), learned Fortran as a by-product
- Then (~1990-93) worked at Cray as an intern
  - Parallelism was vector processing, and up to 8 processors simultaneously working on shared memory; learned about performance and optimization
- Then (~1993) worked at JPL for Cray on T3D/E
  - MPI was standardized, really learned message passing
- At JPL (~2000), worked on grid computing (NASA IPG) applications, learned distributed computing
  - … Worked on spaceborne systems, learned more about fault tolerance
- Clouds
  - Worked on at LSU and U Chicago, funded projects at NSF

Clouds

- Thesis
  - Clouds are somewhere between parallel (cluster) and distributed systems (grid)
- Want to
  - Develop applications, deploy and execute applications
- While optimizing
  - Human effort, application (and system) performance
- Applications are data intensive, distributed, dynamic (D3)
  - All cloud applications aren't D3, but these are the most general
- Platforms are heterogeneous, distributed, not reliable
  - All platforms don't have all these characteristics, but these platforms are the most general

Issues

- Development
  - Expressing affinity (compute to resources, compute to data, data to resources, data to compute, data to data, compute to compute, resources to resources, etc.)
- Performance
  - Mapping compute and data components of application to resources
  - Respecting affinity
  - Responding to resource utilization (compute, storage, network)
  - Fault tolerance & spatial/temporal redundancy
- Challenge
  - How to think about clouds?
  - What are the right abstractions?

Abstractions Challenge

- Role of cloud abstractions
  - Encapsulate application characteristics &
  - Encapsulate platform characteristics
  - to:
    - Decrease development effort
    - Increase performance ye support flexibility
    - Distinguish education from training
- How do we abstract the set of distributed systems to allow this?
- What middleware and tools are needed?

Acknowledgements

- AIMES project (DOE-funded)
  - Shantenu Jha, Mark Santcroos, Matteo Turilli (Rutgers)
  - Jon Weissman (Minnesota)
  - Zhao Zhang (Chicago)
- ExM project (DOE-funded)
  - Mike Wilde, Justin Wozniak, Ketan Maheshwari, Rusty Lusk, Ian Foster (Argonne)
  - Zhao Zhang, Tim Armstrong (Chicago)
  - Matei Ripeanu, Samer Al-Kiswany, Emalayan Vairavanathan (British Columbia)
- DPA & 3DPAS e-SI research themes (e-SI & NSF funded)
  - Shantenu Jha, Omer Rana, Manish Parashar, Jon Weissman, Neil Chue Hong, Simon Dobson, Andre Luckow, Yogesh Simmhan
- Boilerplate
  - Some work was supported by the National Science Foundation while working at the Foundation. Any opinion, finding, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.