VGrADS Programming Tools Overview

VGrADS Knoxville Workshop 2004

Ken Kennedy

Center for High Performance Software Research

(HiPerSoft)

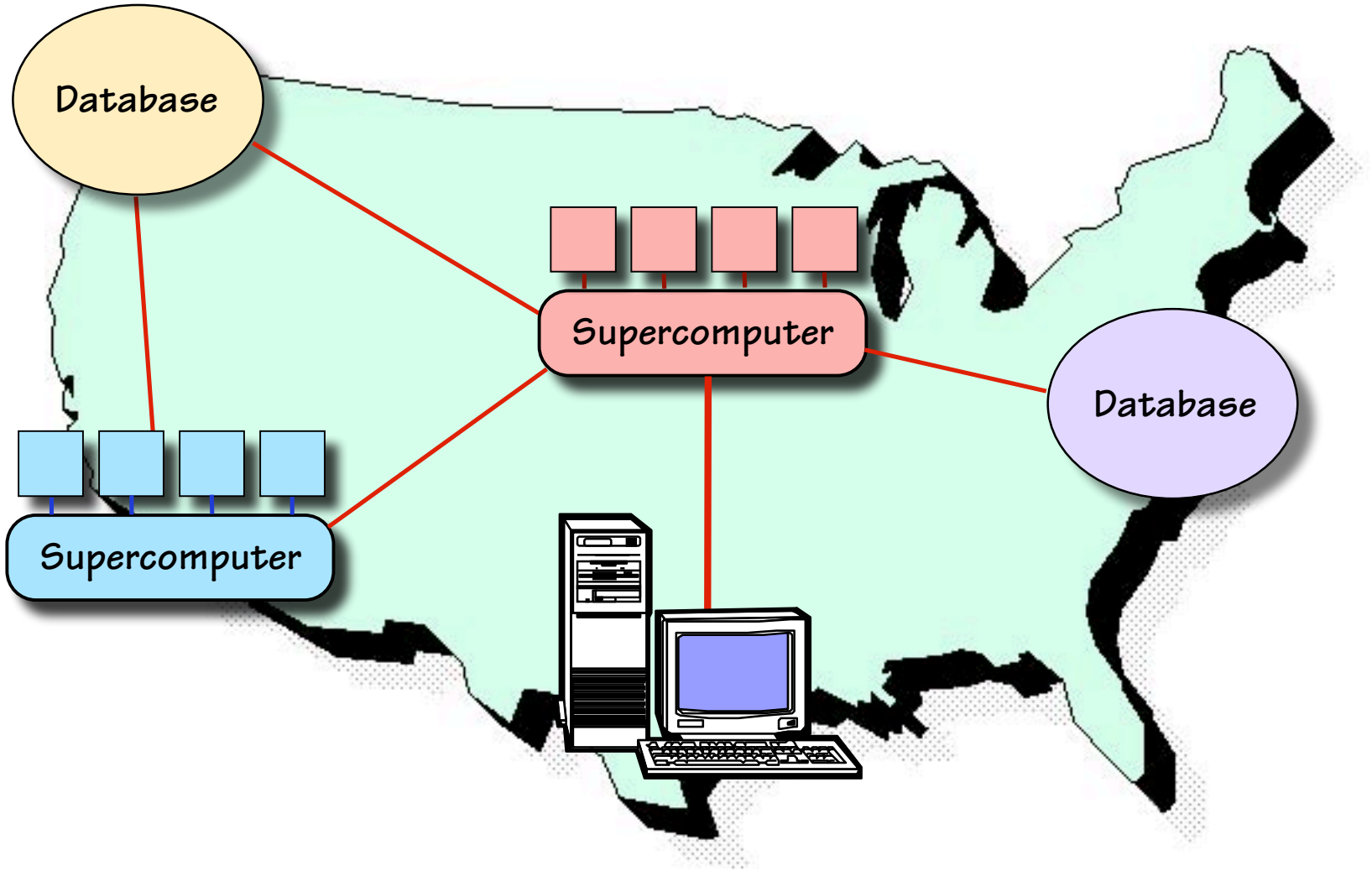
Rice University

http://www.hipersoft.rice.edu/vgrads/

GrADS

Grid Application Development Software Project

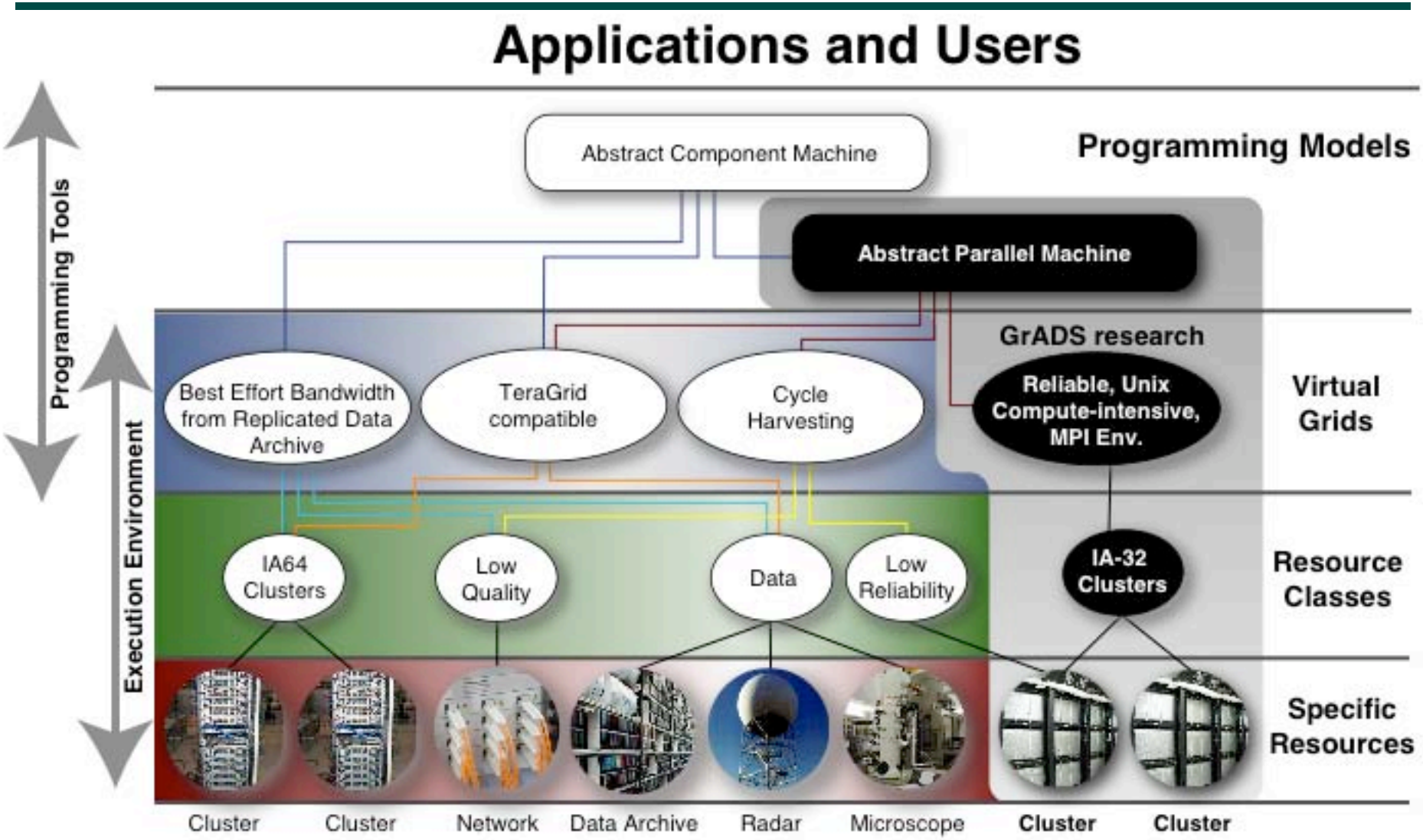
National Distributed Problem Solving



VGrADS Vision

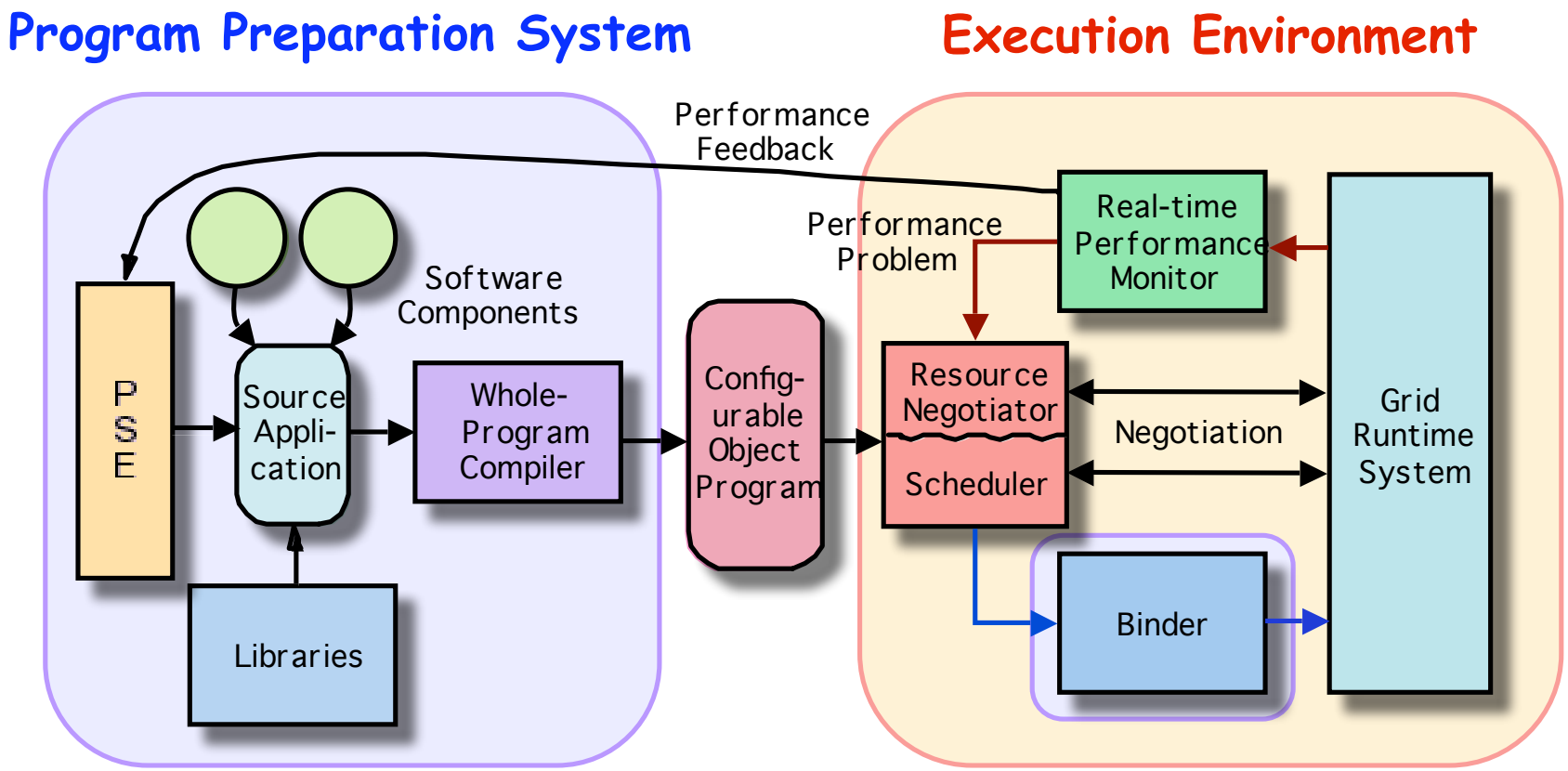
- Build a National Problem-Solving System on the Grid
  - Transparent to the user, who sees a problem-solving system
- Why don't we have this today?
  - Complex application development
    - Dynamic resources require adaptivity
    - Unreliable resources require fault tolerance
    - Uncoordinated resources require management
  - Weak programming tools and models
    - Tied to physical resources
    - If programming is hard, the Grid will not not reach its potential
- What do we propose as a solution?
  - Virtual Grids (vgrids) raise level of abstraction
  - Tools exploit vgrids, provide better user interface

VGrADS Vision



GrADS
Grid Application Development Software Project

GrADSoft Architecture



Lessons from GrADS

- Mapping for MPI Jobs is Hard
  - Much easier to schedule to single clusters
    - Although we were able to do some interesting experiments
- Performance Model construction is hard
  - Hybrid static/dynamic schemes are best
  - Difficult for application developers to do by hand
- Heterogeneity Adds Complexity
  - We completely revised the Binder mechanisms to support this
  - Scheduling is more critical
- Rescheduling/Migration is Hard
  - N-N is possible with pre-allocation
  - N-M rescheduling requires application collaboration (generalized checkpointing)
  - Both require performance models to determine when it is profitable

Programming Tools

- Integration of Grid Applications from Abstract Descriptions
  - Abstract Component Machine
    - Components preinstalled on various resources
    - Applications defined by scripting languages
      - EMAN: Python
    - Conversion of scripts to workflows
- Workflow Scheduling
  - Based on performance models and data movement costs
  - Full graph scheduling
    - Avoid getting stuck at certain resources
- Automatic Construction of Performance Models
  - Work of Gabriel Marin, John Mellor-Crummey and Bo Liu

Workflow Applications

- Application consists of a collection of components to be executed in a certain partial order for successful execution
  - Ordering typically based on data dependences
- Workflow applications can be represented by a DAG (Directed Acyclic Graph)
  - Nodes in the DAG denote application components
    - Types: Sequential, parameter sweep, embarrassingly ||, tightly-coupled etc.
    - May access/update datasets and databases, control nodes for data movements
  - Edges represent data and control dependencies
    - Data dependences typically involve file transfer

Grid Applications

LIGO

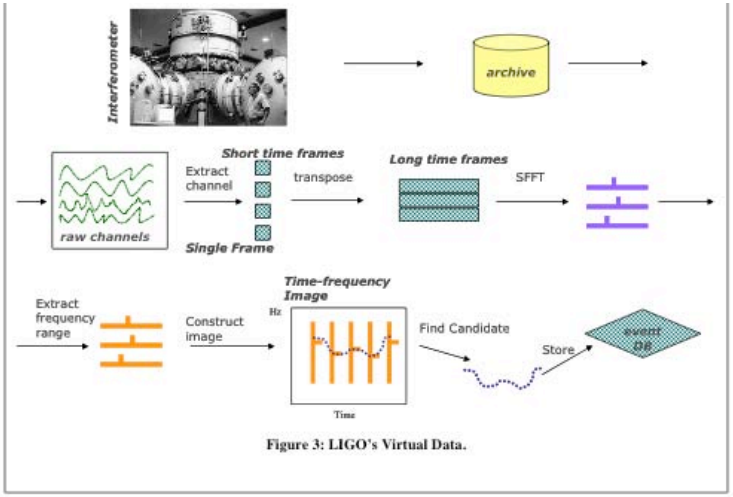
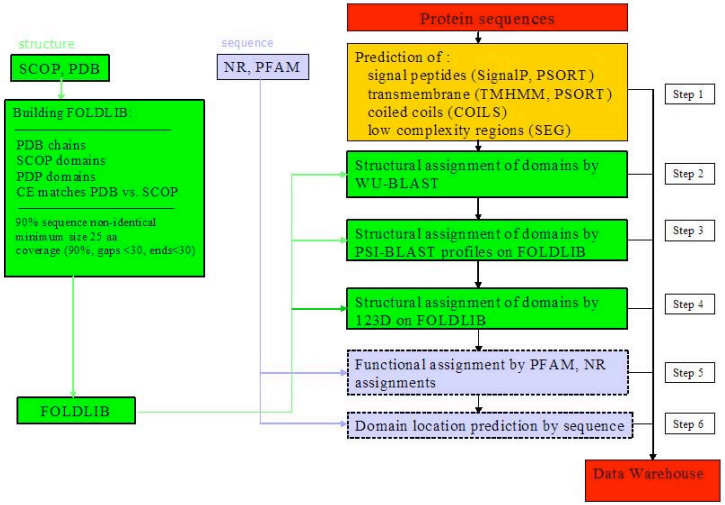
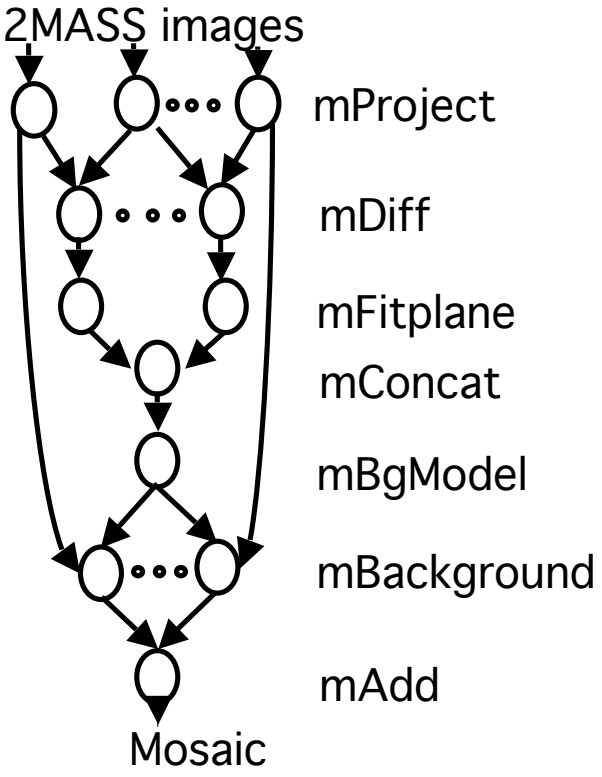


Figure 3: LIGO's Virtual Data.

EOL

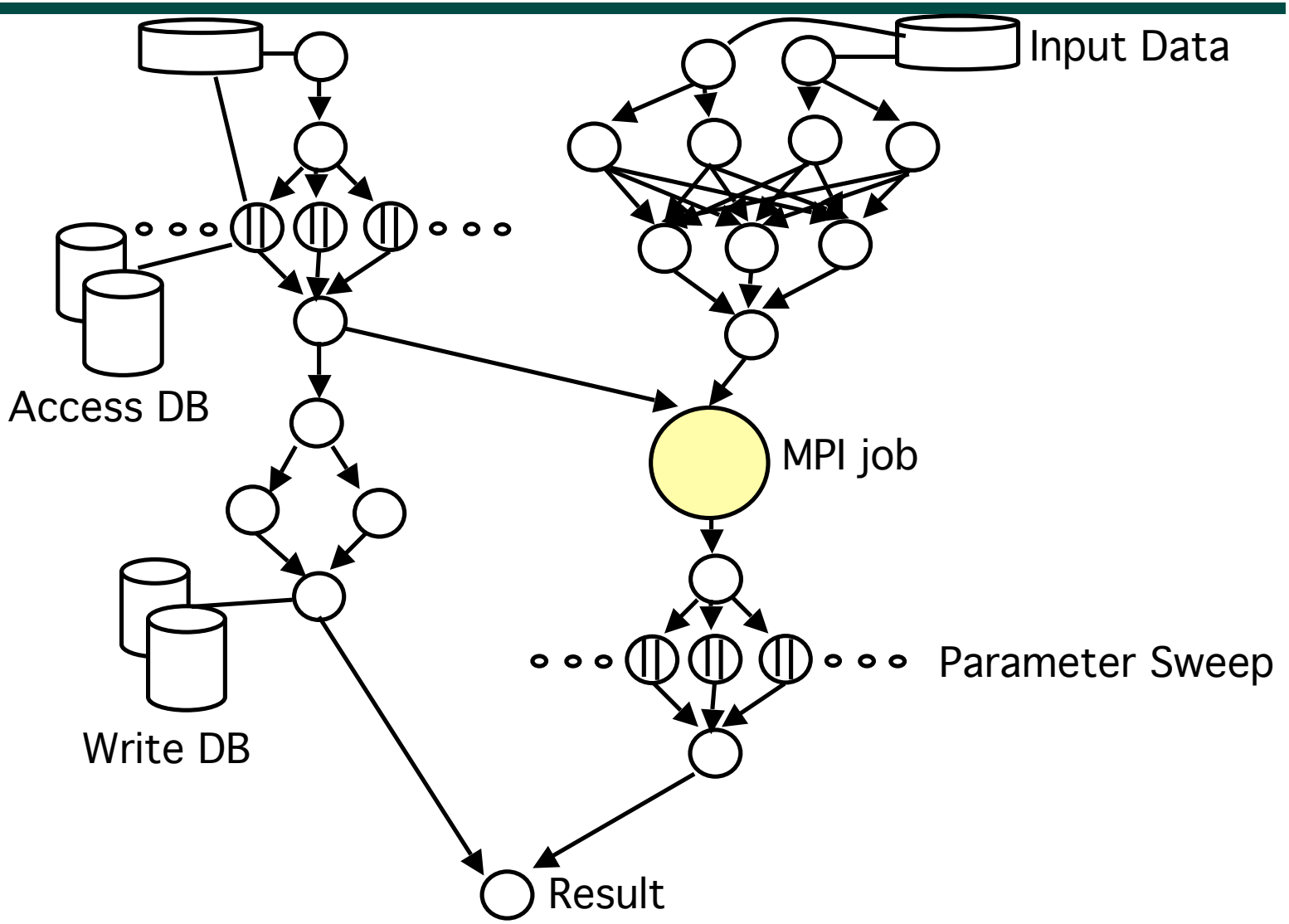


Montage



GrADS
Grid Application Development Software Project

Grid Workflow Application Representation

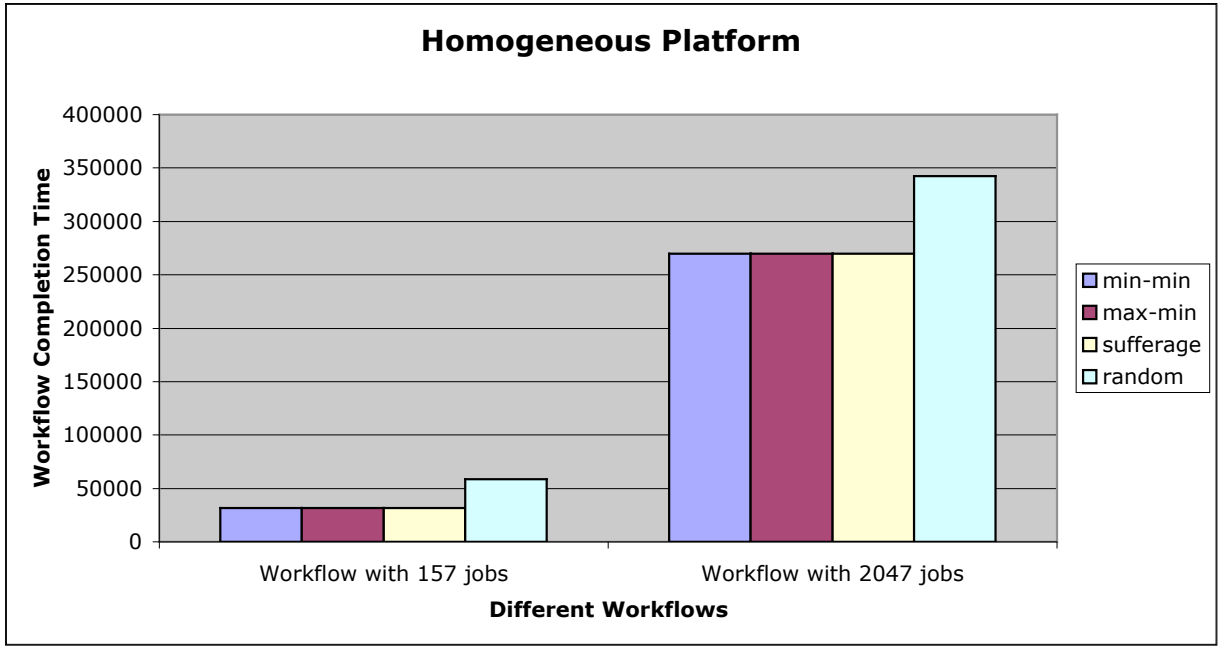


Workflow Scheduling Strategy

- Base Work on Performance Model Construction
  - —Estimate: operation counts and memory delays
    - Operation counts based on profiling
    - Memory estimates based on construction of reuse-distance estimation function
- Look at Whole DAG
  - —Use performance models as surrogates for execution
  - —Apply heuristics to match resources to workflow steps
    - Both computation and data movement costs considered
    - Selection of most critical requests first

      Work backward and forward from those
  - —Preprocess workflow to make scheduling easier
    - Fusing adjacent vertices that have high data transfer volume

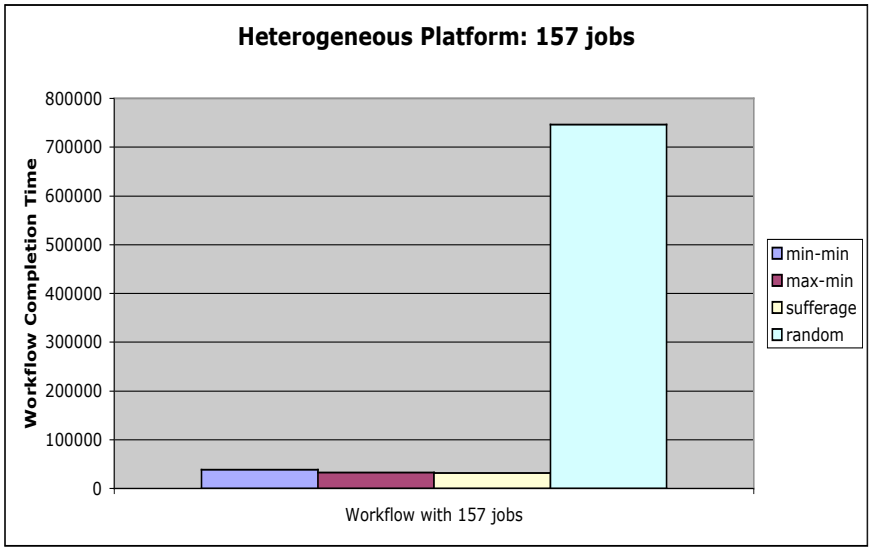
Heuristic Workflow Scheduling: Results

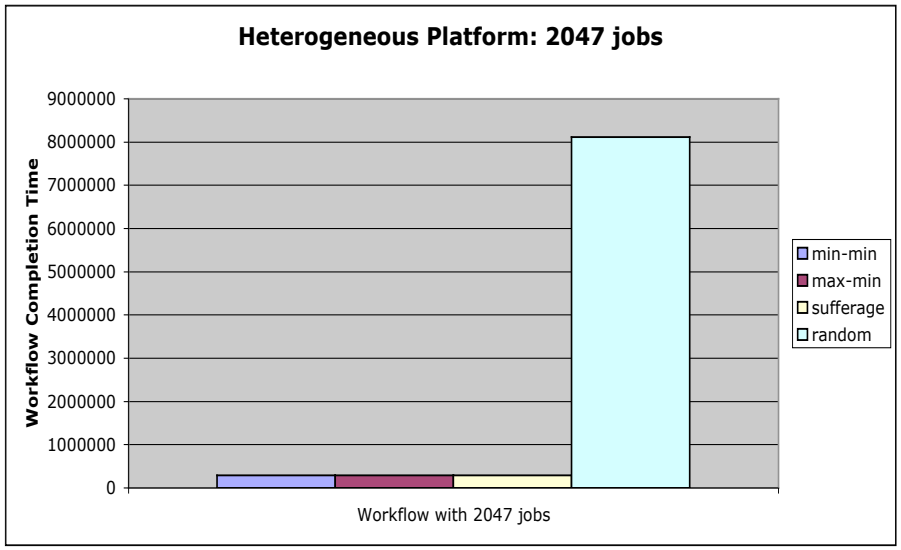


- Simulation results for workflow completion times for different "Montage" workflows
- Improvement of >20% for homogeneous platform

Preliminary results from joint work with Ewa Deelman *et. al.* at USC ISI

Heuristic Workflow Scheduling: Results





- By using heuristic workflow scheduling, workflow completion times improve by an order of magnitude[>20 times] over random scheduling for heterogeneous platform
- Workflow completion time is within 10% of that using a very expensive AI scheduler that doesn't scale to 2047 jobs

Issues

- Division of Labor between Vgrid and Programming Tools
  - Rescheduling in GrADS
    - Work illustrated the difficulties associated with rescheduling
    - Should it be done at all?
  - Scheduling under uncertainty
    - Current scheduler: assumes stable resources
    - Problems: varying load on processors

      Makes scheduling inaccurate
    - How can Vgrid and Ptools collaborate on this?
- Making Peace with Community Efforts
  - Do we really need Globus?
  - What about Grid services?

Issues

- Dynamic versus Static Workflow Scheduling
  - Current Condor/DAGMan: dynamic
    - Schedule when all inputs are ready
    - Problem: computations stuck at inappropriate resources
  - Current GrADS/VGrADS: completely static for a single DAG
    - Schedule entire DAG using performance models in place of actual executions
    - Problems: accurate performance models, load variability
  - Hybrid Scheme: adjust static schedules dynamically
    - Monitor accuracy of performance prediction and loads
    - Reschedule remaining parts of DAG based thereon
    - Do we need system support for milestone notification?

Programming Tools Futures

- Translation from High-Level Abstractions
  - Construction of workflows from Python
    - Future: other scripting languages
  - What are the scientific problems?
- Scheduling
  - Global workflow scheduling
    - Future: partial dynamic adaptation
  - Validation of new scheduling algorithms
  - Scheduling in the presence of load
    - Dependence on prediction
- Performance Model Construction
  - Fully automatic methods
  - Composition of performance models from components
  - Performance models for parallel applications (LACSI)

Other Issues

- Fault Tolerance
  - Fault tolerance in DAGS
    - Should be easier: duplicate copies of files and rescheduling
  - Support for checkpoint construction
- Scheduling to minimize variability of makespans
  - Application: real-time

Education, Outreach, and Training

- Started activity at Rice
  - Lectures at CS CAMP (high school girls, their teachers & principals)
  - Will sponsor student(s) traveling to Grace Hopper Celebration of Women in Computing (Chicago, October 6-9)
- Working to recruit/identify Summer 2005 students
  - AGEP students (undergraduate Juniors/Seniors, mostly underrepresented groups) to work on grid computing projects
  - Add more content for CS CAMP 2005
    - Seeking funding for future CS CAMP activities
- Germ of idea: grid computing experiment for high school
  - Leverage CS CAMP participants' experience
  - Provide canned software to schools, run an "interesting" application
  - Could be great project to build on or glorious failure