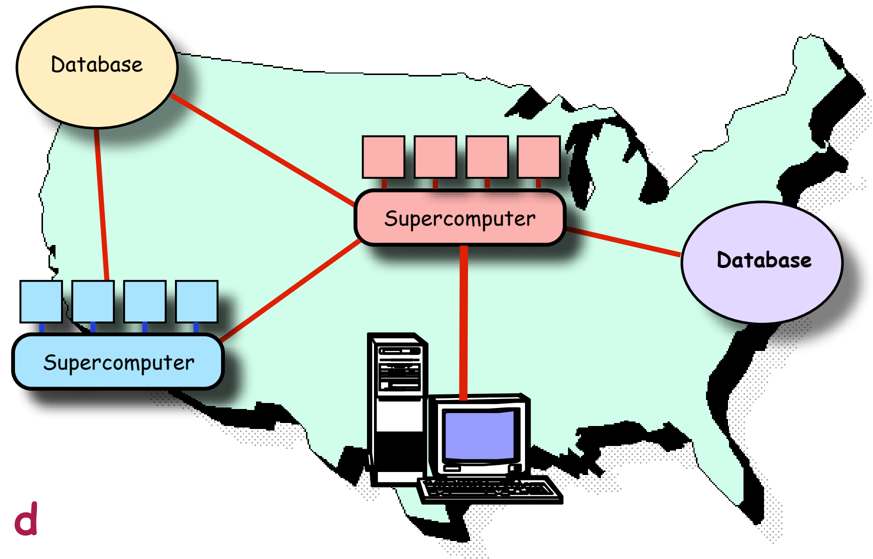
Why Performance Models Matter for Grid Computing

Ken Kennedy
Center for High Performance Software
Rice University

http://vgrads.rice.edu/WoCo9Overview07-06.pdf

Vision: Global Distributed Problem Solving

- Where We Want To Be
  - Transparent Grid computing
    - Submit job
    - Find & schedule resources
    - Execute efficiently
- Where We Are
  - Low-level hand programming
  - Programmer must manage:
    - Heterogeneous resources
    - Scheduling of computation and d
    - Fault tolerance and performance adaptation



- What Do We Propose as A Solution?
  - Separate application development from resource management
    - Through an abstraction called the Virtual Grid
  - Provide tools to bridge the gap between conventional and Grid computation
    - Scheduling, resource management, distributed launch, simple programming models, fault tolerance, grid economies

The VGrADS Team

- **VGrADS is an NSF-funded Information Technology Research project**

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Dan Reed

RICE

Keith Cooper
Ken Kennedy
Charles Koelbel
Richard Tapia
Linda Torczon

THE UNIVERSITY of TENNESSEE

Jack Dongarra

ISI

Carl Kesselman

UCSD

Fran Berman
Andrew Chien
Henri Casanova

UCSB

Rich Wolski

UH

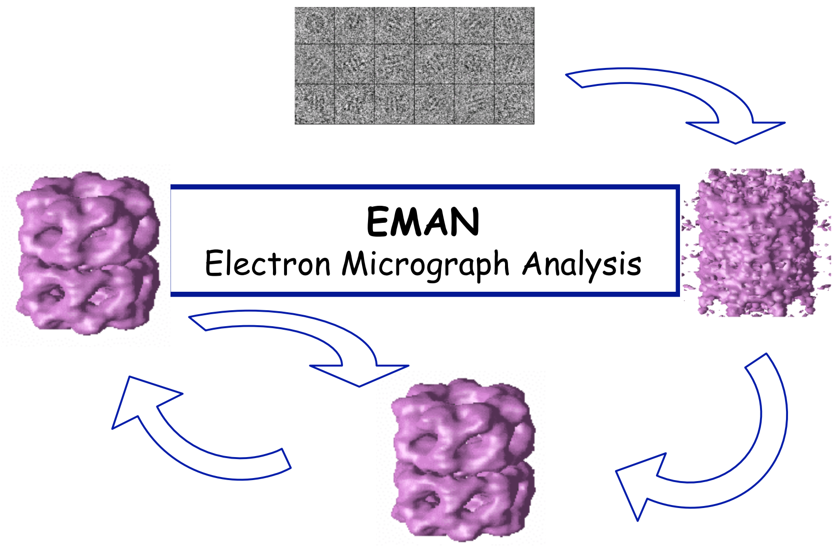
Lennart Johnsson

- **Plus many graduate students, postdocs, and technical staff!**

VGrADS Project Vision

- Virtual Grid Abstraction
  - Separation of Concerns
    - Resource location, reservation, and management
    - Application development and resource requirement specification
  - Permits true scalability and control of resources
- Tools for Application Development
  - Easy application scheduling, launch, and management
    - Off-line scheduling of workflow graphs
    - Automatic construction of performance models
  - Abstract programming interfaces
- Support for Fault Tolerance and Rescheduling/Migration
  - Collaboration between application and virtual grid execution system
- Research Driven by Real Application Needs
  - EMAN, LEAD, GridSAT, Montage

VGrADS Application Collaborations



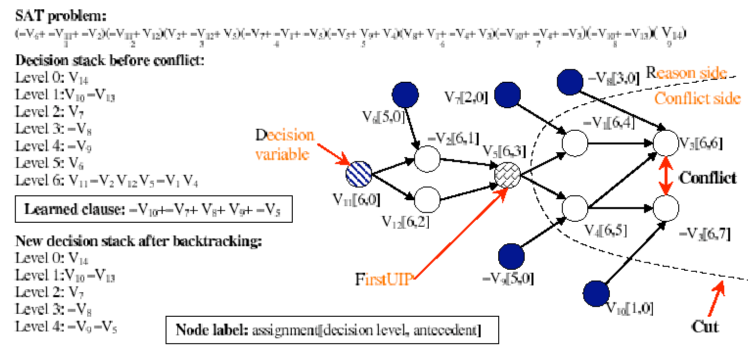
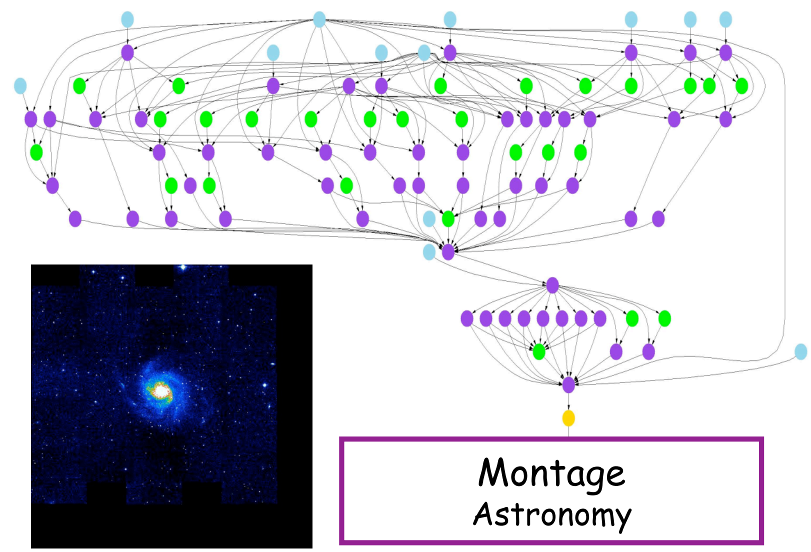
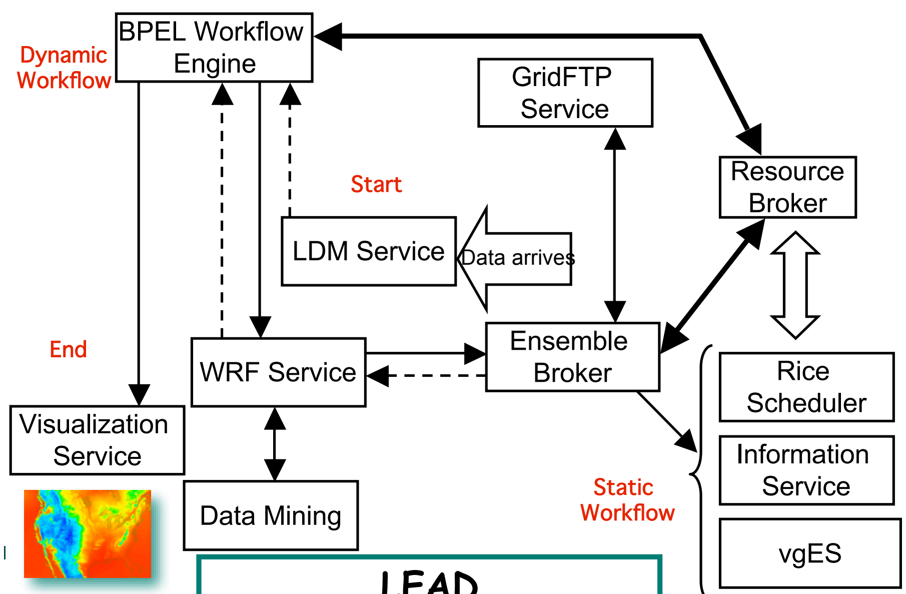


Figure 1: Example of conflict analysis with learning and non-chronological backtracking

GridSAT
Boolean Satisfiability



Montage
Astronomy



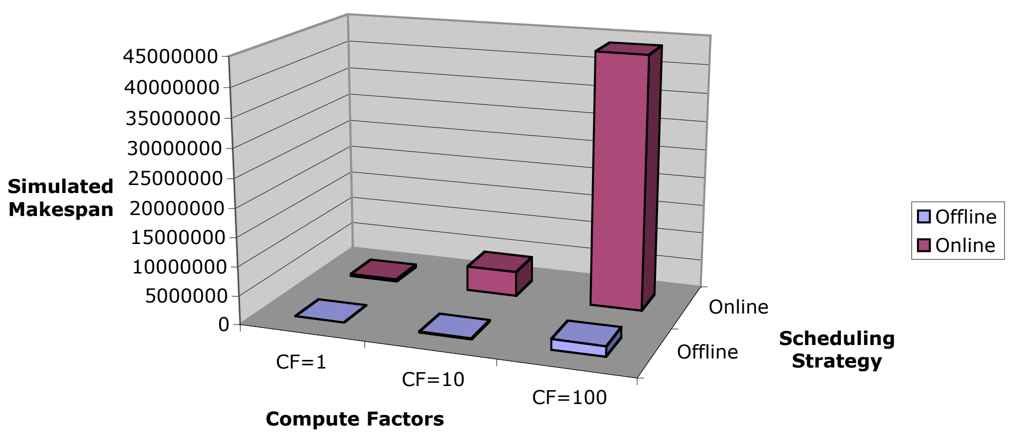
LEAD
Atmospheric Science

VGrADS Big Ideas

- Virtualization of Resources
  - Application specifies required resources in Virtual Grid Definition language (vgDL)
    - Give me a loose bag of 1000 processors, with 1 Gb memory per processor, with the fastest possible processors
    - Give me a tight bag of as many Opterons as possible
  - Virtual Grid Execution System (vgES) produces specific virtual grid matching specification
  - Avoids need for scheduling against the entire space of global resources
- Generic In-Advance Scheduling of Application Workflows
  - Application includes performance models for all workflow nodes
    - Performance models automatically constructed
  - Software schedules applications onto virtual Grid, minimizing total makespan
    - Including both computation and data movement times

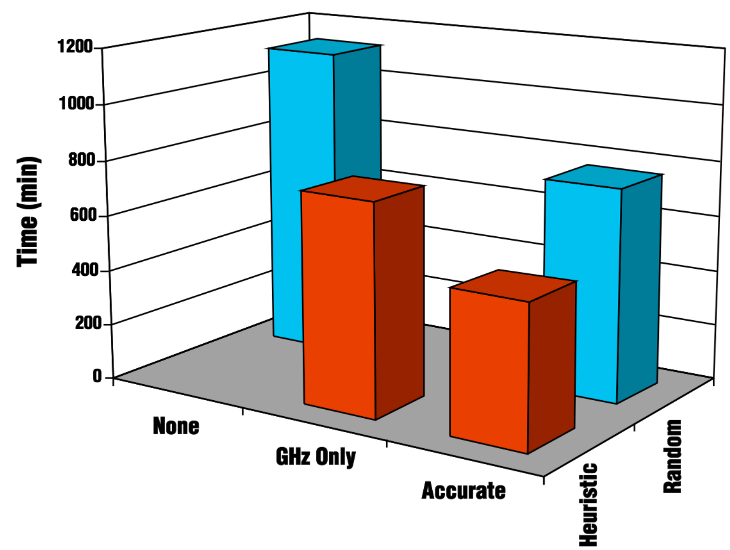
Workflow Scheduling Results

Dramatic makespan reduction of *offline* scheduling over *online* scheduling – Application: **Montage**



"Resource Allocation Strategies for Workflows in Grids"
CCGrid'05

Value of *performance models* and *heuristics* for offline scheduling – Application: **EMAN**



"Scheduling Strategies for Mapping Application Workflows onto the Grid"
HPDC'05

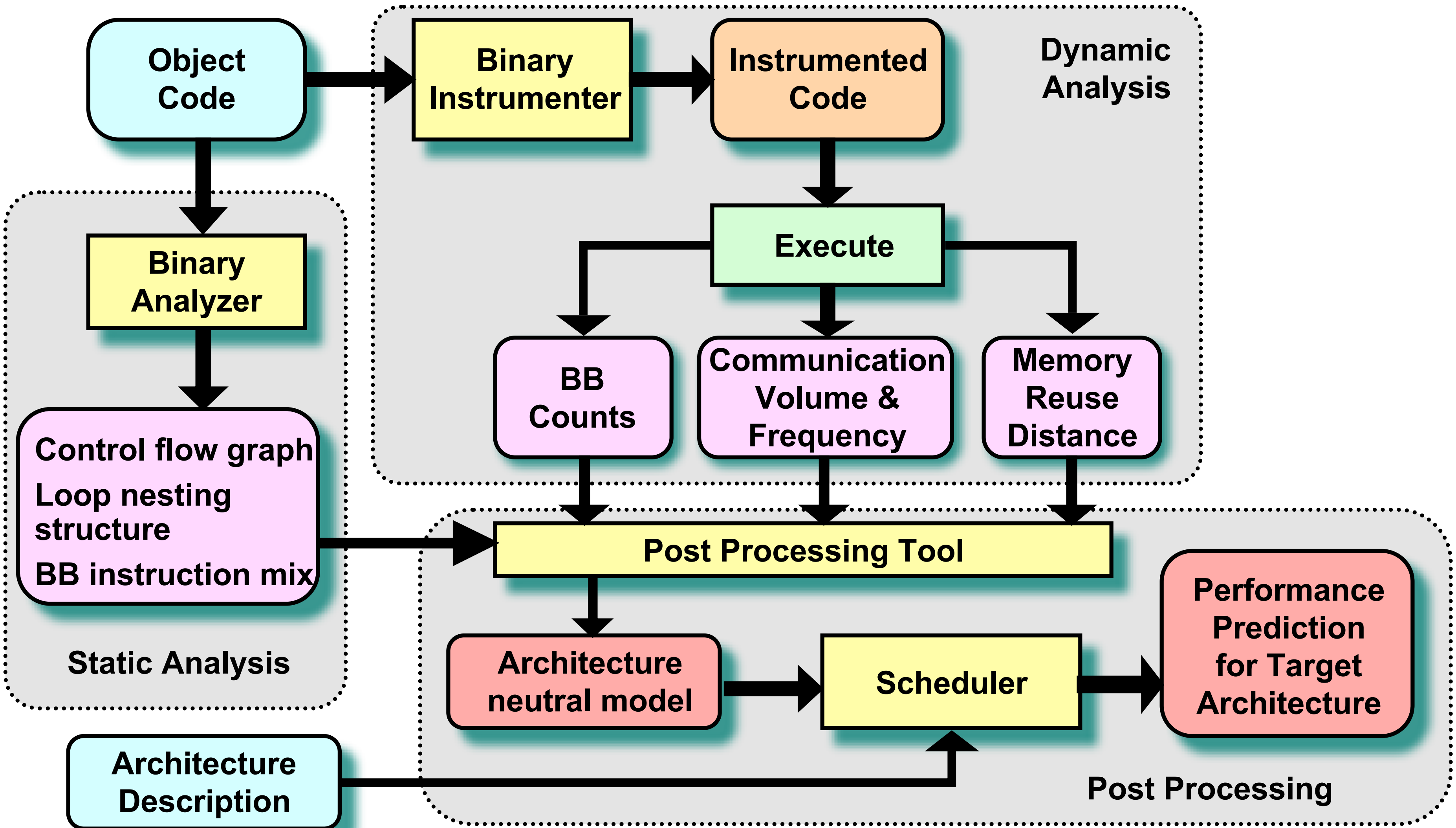
Virtual Grid Results

- Virtual Grid prescreening of resources produces schedules dramatically faster without significant loss of schedule quality
  - Huang, Casanova and Chien. Using Virtual Grids to Simplify Application Scheduling. IPDPS 2006.
  - Zhang, Mandal, Casanova, Chien, Kee, Kennedy and Koelbel. Scalable Grid Application Scheduling via Decoupled Resource Selection and Scheduling. CCGrid06.
- Current VGrADS heuristics are quadratic in number of distinct resource classes
  - Two-phase approach brings this much closer to linear time

Performance Model Construction

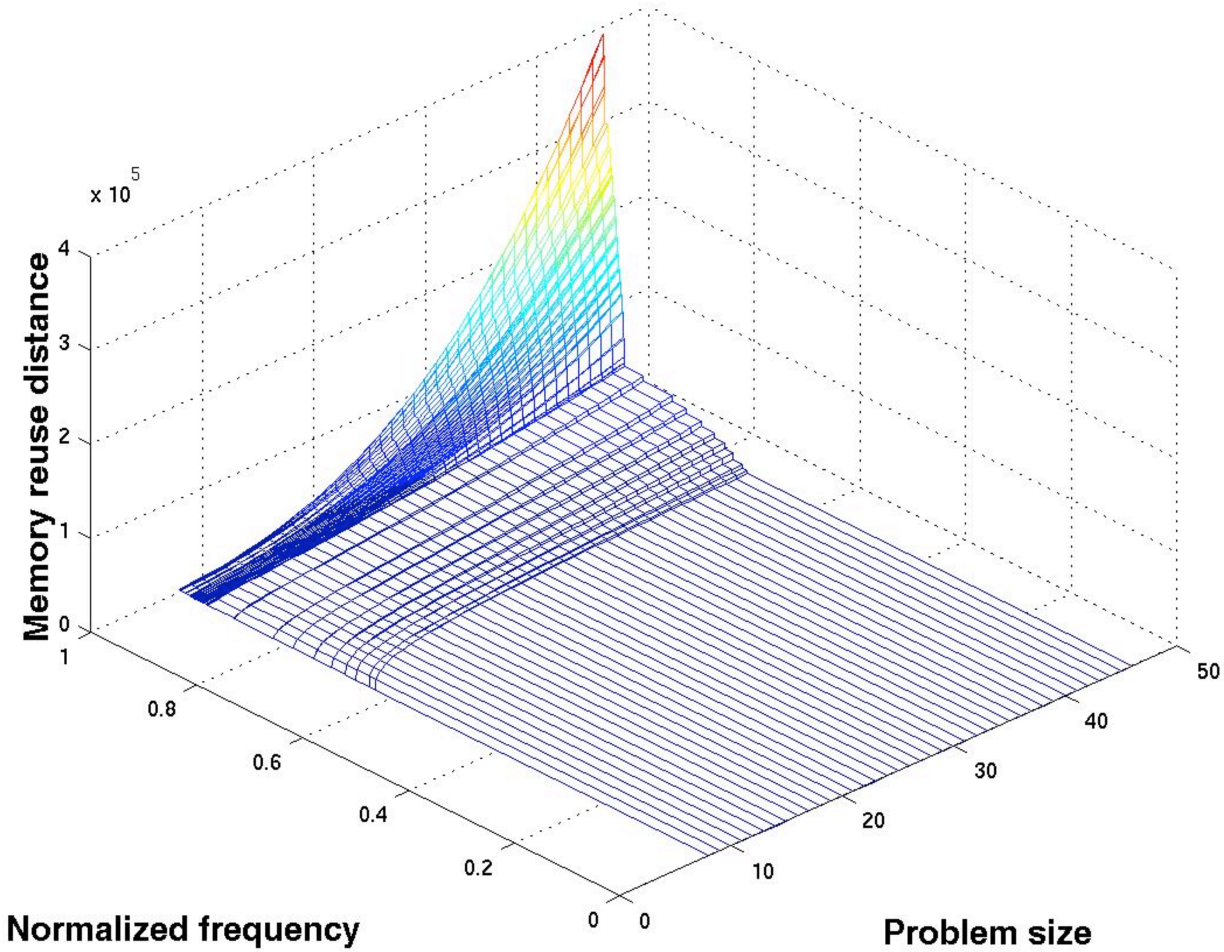
- Problem: Performance models are difficult to construct
  - By-hand and models take a great deal of time
  - Accuracy is often poor
- Solution (Mellor-Crummey and Marin): Construct performance models automatically
  - From binary for a single resource and execution profiles
  - Generate a distinct model for each target resource
- Current results
  - Uniprocessor modeling
    - Can be extended to parallel MPI steps
  - Memory hierarchy behavior
  - Models for instruction mix
    - Application-specific models
    - Scheduling using delays provided as machine specification

Performance Prediction Overview



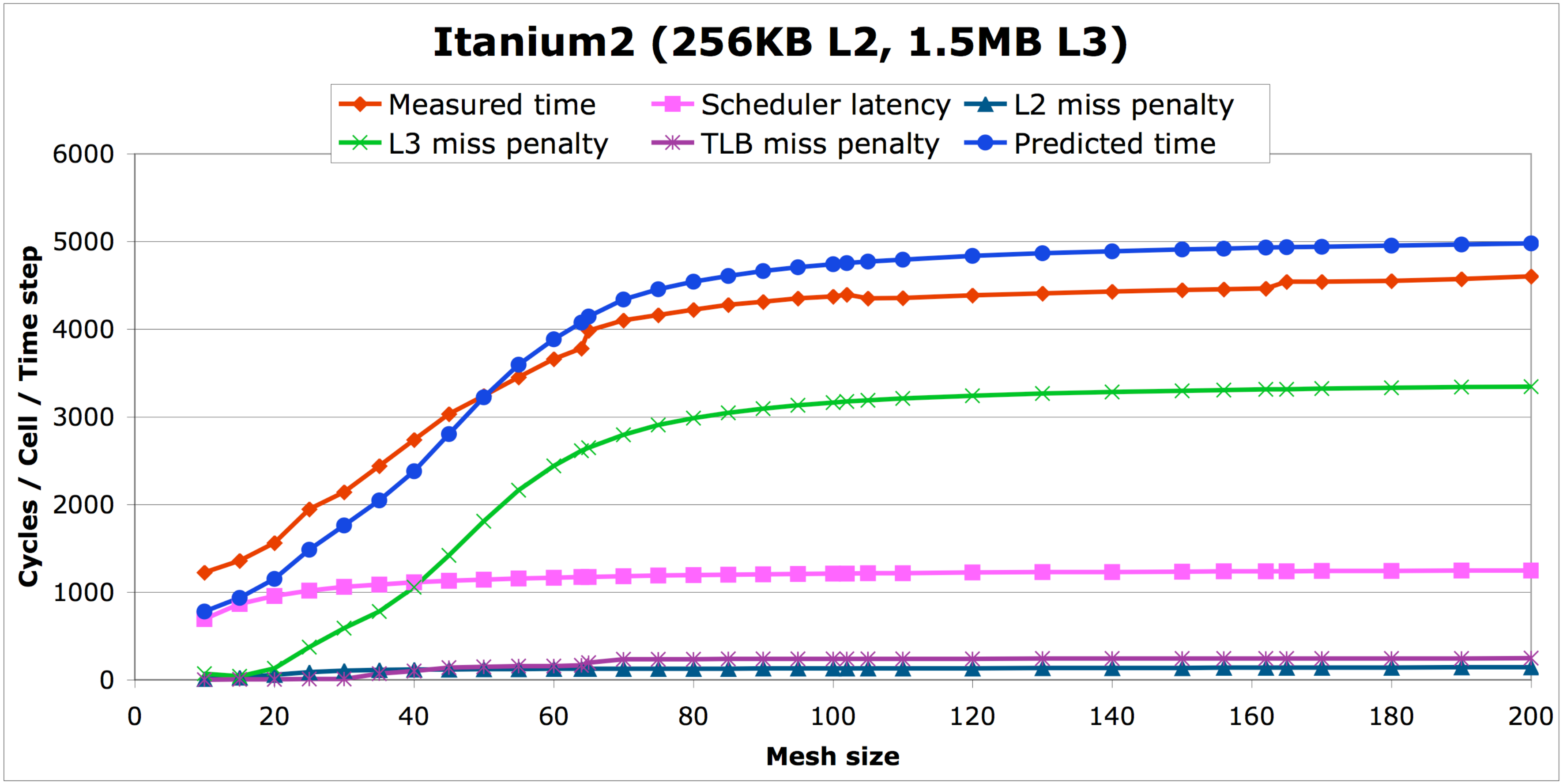
VGrADS
Virtual Grid Application Development Software Project

Modeling Memory Reuse Distance



VGrADS
Virtual Grid Application Development Software Project

Execution Behavior: NAS LU 2D 3.0



Value of Performance Models

- True Grid Economy
  - All resources have a "rate of exchange"
    - Example: 1 CPU unit Opteron = 2 units Itanium (at same Ghz)
  - Rate of exchange determined by averaging over all applications
    - Weighted by application global resource usage percentage
- Improved Global Resource Utilization
  - A particular application may be able to take advantage of performance models to reduce costs
    - Example: For EMAN, 1 Opteron unit = 3 Itanium units
    - Thus, EMAN should always favor Opterons when available
  - If all applications do this, the total system resources will be used far more efficiently

Performance Models and Batch Scheduling

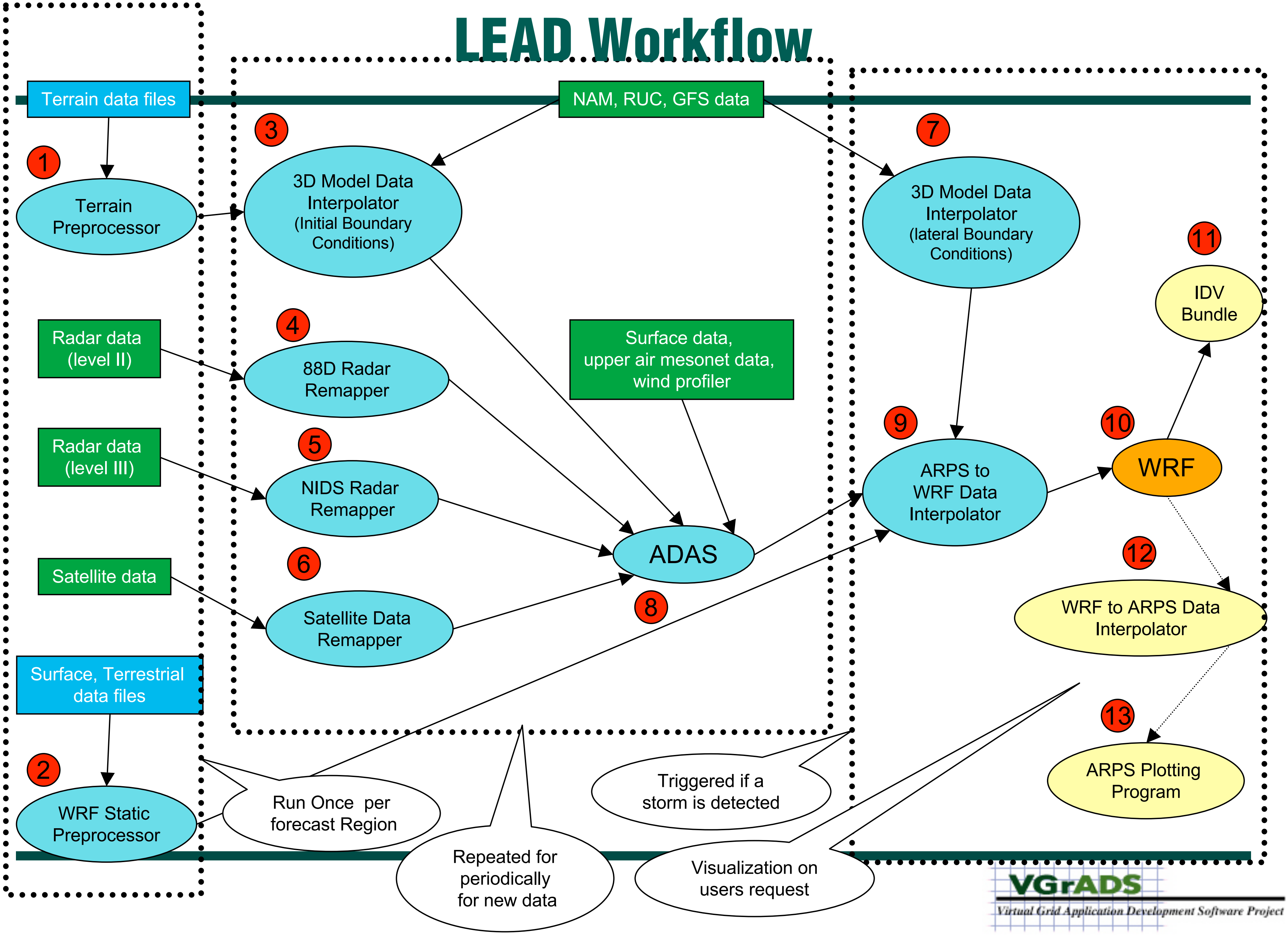
- Currently VGrADS supports scheduling using estimated batch queue waiting times
  - Batch queue estimates are factored into communication time
    - E.g., the delay in moving from one resource to another is data movement time + estimated batch queue waiting time
  - Unfortunately, estimates can have large standard deviations
- Next phase: limiting variability through two strategies:
  - Resource reservations: partially supported on the TeraGrid and other schedulers
  - In advance queue insertion: submit jobs before data arrives based on estimates
    - Can be used to simulate advance reservations
- Exploiting this requires a preliminary schedule indicating when the resources are needed
  - Problem: how to build an accurate schedule when exact resource types are unknown

Solution to Preliminary Scheduling Problem

- Use performance models to specify alternative resources
  - For step B, I need the equivalent of 200 Opterons, where 1 Opteron = 3 Itanium = 1.3 Power 5
    - Equivalence from performance model
- This permits an accurate preliminary schedule because the performance model standardizes the time for each step
  - Scheduling can then proceed with accurate estimates of when each resource collection will be needed
  - Makes advance reservations more accurate
    - Data will arrive neither too early or too late
- It may provide a mixture to meet the computational requirments, if the specification permits
  - Give me a loose bag of tight bags containing the equivalent of 200 Opterons, minimize the number of tight bags and the overall cost
    - Solution might be 150 Opterons in one cluster and 150 Itaniums in another

Scheduling to a Deadline

- LEAD Project has a feedback loop
  - After each preliminary workflow, results are used to adjust doppler radar configurations to get more accurate information in the next phase
  - This must be done on a tight time constraint
- Performance models make this scheduling possible
  - What happens if the first schedule misses the deadline?
    - The time must be reduced, either by doing less computation or by using more resources
    - But how many more resources?
  - Suppose we can differentiate the model to compute for each step, the sensitivity of running time to more resources
    - Automatic differentiation technologies exist, but other strategies may also make this possible
  - Derivatives can be used to predict resources needed to meet the deadline along the critical path


LEAD Workflow
Terrain data files
NAM, RUC, GFS data
1
Terrain Preprocessor
3
3D Model Data Interpolator (Initial Boundary Conditions)
7
3D Model Data Interpolator (lateral Boundary Conditions)
11
IDV Bundle
Radar data (level II)
4
88D Radar Remapper
Surface data, upper air mesonet data, wind profiler
Radar data (level III)
5
NIDS Radar Remapper
9
ARPS to WRF Data Interpolator
10
WRF
ADAS
Satellite data
6
Satellite Data Remapper
8
12
WRF to ARPS Data Interpolator
Surface, Terrestrial data files
13
ARPS Plotting Program
2
WRF Static Preprocessor
Run Once per forecast Region
Triggered if a storm is detected
Repeated for periodically for new data
Visualization on users request
VGrADS
Virtual Grid Application Development Software Project

Summary

- VGrADS Project Uses Two Mechanisms for Scheduling
  - Abstract resource specification to produce preliminary collection of resources
  - More precise scheduling on abstract collection using performance models
    - Combined strategy produces excellent schedules in practice
    - Performance models can be constructed automatically
- New Challenges Require Sophisticated Methods
  - Minimizing cost of application execution
  - Scheduling with batch queues and advanced reservations
  - Scheduling to deadlines
- Current Driving Application
  - LEAD: Linked Environments for Atmospheric Discovery
    - Requires deadlines and efficent resource utilization

Relationship to Future Architectures

- Future supercomputers will have heterogeneous components
  - Cray, Cell, Intel, CPU + coprocessor (GPU), …
- Scheduling subcomputations onto units while managing data movment costs will be the key to effective usage of such systems
  - Adapt VGrADS strategy
  - Difference: must consider alternative compilations of the same computation, then model performance of the result
- Could be used to tailor systems or chips to particular users' application mixes