Intelligent Optimization of Parallel and Distributed Applications

**USC/ISI:**
Jacqueline Chame
Chun Chen
Ewa Deelman
Yolanda Gil
**Mary Hall**
Kristina Lerman
Yoonju Lee Nelson

**USC (Physics/CS):**
Bhupesh Bansal
Aiichiro Nakano
Priya Vasishta

**OSU:**
Umit Catalyurek
Vijay Kumar
Tahsin Kurc
Joel Saltz
Ashish Sharma

This research has been partially funded by NSF NGS and CSR.
Motivation

Historical Perspective

• Highly tuned applications are too hard to develop, port
• Complexity leads to fragile applications & system software
• Ad hoc approaches, not formalized
• Community knowledge exists in the minds of too few people
• Fragmentation of community, duplication of effort

How to Move Forward

• Systematize the process of constructing and tuning applications from existing components or patterns
• Build tools that form a foundation to which many can contribute and improve
• Organize the community to work together
Key Concepts

• A systematic strategy for composing application components into workflows

• Search for the most appropriate implementation of both components and workflows

• Component optimization
  - Select among implementation variants of the same computation
  - Derive integer values of optimization parameters
  - Only search promising code variants and a restricted parameter space

• Workflow optimization
  - Knowledge-rich representation of workflow properties
Early Project Goals

• Define interfaces
• Combine infrastructures
  - Pegasus + Wings already combined
  - Now want to incorporate DataCutter
• Experiments in each sub-project
  - Compiler-guided component optimization
  - Optimization of workflow intermediate data
• Pairwise experiments
  - Component optimization of MD simulation
  - DataCutter workflow in Wings/Pegasus
System Design

Application (Nakano, Vashishta):
- Scalable Molecular Dynamics Simulation

Existing Tools (Hall):
- ECO Compiler: Model-guided empirical optimization
- Code Isolator

Other Application Tools:
- Express application-level parameters, range, models
- Learning parameter models (Lerman)
Compiler: Matrix-Vector Multiply on Pentium M

Application-Level Parameters: Visualization of MD Simulation

• Explore tradeoff space of two application-level parameters
  – Cell size: granularity of decomposition
  – Cache size: number of neighbors to replicate

Findings:
• Cell size has more impact on performance
• Parameter values sensitive to graph connectivity, number of processors
• Search can be generalized
System Design

Existing Tools:
- **WINGS (Gil)**: Workflow specification and mapping
- **Pegasus (Deelman)**: Workflow scheduling and optimization for distributed platforms
- **DataCutter (Saltz, Kurc, Catalyurek)**: Data management and optimization of data-intensive applications
Workflow Optimization: Reducing Workflow Space Requirements

- **Optimization problem:** Data replicas and intermediate results introduce extensive storage requirements
- **CleanUp:** identify “dead” files and remove

Performance Optimization of Data-intensive Workflows: Integrating Wings and Data Cutter

**Application: Biomedical image analysis**

```
<table>
<thead>
<tr>
<th>z-projection</th>
<th>normalization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>alignment &amp; stitching</td>
<td>warping</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

*Sample workflow for image correction*

Each component is implemented as a DataCutter filter. Our system enables distributed processing of very large, out-of-core image data.

**Integration with Wings**

Extend the core ontology of Wings to support the creation of image analysis workflow templates

**Performance optimization parameters:**
- Tiling of data to fit in core
- Mapping data to processors
- Internal data organization

**Performance/accuracy parameters:**
- Image resolution
- Feature detection accuracy
Concluding Remarks

• Three core technical ideas
  – **Compiler technology:** Modular compilers, systematic approach to optimization, empirical search, *hand-tuned performance*
  – **Components:** Tunable, automatically-generated XML-based interfaces, knowledge representations, more empirical search
  – **Systematic:** Based on machine learning, knowledge representation

• Focus on long-term evolutionary path
• ... And community organization