Using Scientific Workflows for Science and Engineering Optimisation

David Abramson

Director, Research Computing Centre
Professor of Computer Science
University of Queensland

With recent help from Hoang Nguyen, Timos Kipouros, Zane Van Iperen
Introduction

• Some Engineering Optimization Problems
• The Nimrod tool family
• Scientific Workflows
• Execution Engines
  – Clusters & Grids & Clouds
• Interacting with designs
• Conclusions
Some Engineering Optimization Problems
Air pollution

Cope, Victorian EPA (in 1990)
Wildfires
Lynch, Beringer, Uotila  Monash U, AU
Aerofoil Design
Kipouros, Cambridge, UK

• Geometry management using Free Form Deformation – 8 design variables
• Evaluation of the aerodynamic characteristics, Cl, Cd, and Cm coefficients using Xfoil
• Investigation of the lift to drag trade-off subject to hard geometrical constraints to the thickness of the airfoil at 25% and 50% of the chord (in order to maintain practical significance to the design problem)
Heart disease still leading cause of death
Understanding the underlying physiological mechanisms is cheaper and faster when experimental studies are performed together with mathematical models & computer simulations
Studying pathologies
Developing & Testing drugs
Micromixer optimization
Kipouros, Cambridge, UK

• Microfluidics
  – $10^{-9}$ to $10^{-18}$ litres amounts of fluids
  – Gaining importance in various fields
• Micromixer deals with mixing fluids in the smallest scale
• Active vs. passive
Electro-chemistry
Bond, Gavaghan: Monash, Oxford
The Nimrod Tools Family
Nimrod supporting “real” science

• A full parameter sweep is the cross product of all the parameters (Nimrod/G)
• An optimization run minimizes some output metric and returns parameter combinations that do this (Nimrod/O)
• Design of experiments limits number of combinations (Nimrod/E)
• Workflows (Nimrod/K)
Legacy Nimrod family

Plan File

Nimrod Portal

Nimrod/G

Nimrod/O

Nimrod/E

Actuators

Grid Middleware

### Nimrod Portal

#### Parameter Section

1. **Parameter: pressure**
   - Type: float
   - Range: from 5000 to 6000
   - Points: 4

2. **Parameter: concentration**
   - Type: float
   - Range: from 0.002 to 0.005
   - Points: 2

3. **Parameter: material**
   - Type: text
   - Select: any of "Fe", "Al"

#### Tasks Section

1. **Task: main**
   - Copy: `compModel`
     - Node: `compModel`

2. **Task: inputFile.skel**
   - Copy: `inputFile.skel`
     - Node: `inputFile.skel`
   - Node: substitute `inputFile.skel` with `inputFile`

3. **Task: execute**
   - Node: `./compModel < inputFile > results`
     - Copy: `results.$jobname`

**Tasks**

- `rootrootest`
- `rootnodestart`
- `rootmain`
- `rootmodelfinish`
- `rootrootfinish`

**Add a new parameter**
**Add a comment**

**Save**
**Cancel and Reload**
**Text mode**
Nimrod Development Cycle

Prepare Jobs using Portal

Jobs Scheduled Executed Dynamically

Sent to available machines

Results displayed & interpreted
Scientific Workflows and Nimrod
Nimrod/K Workflows

- Nimrod/K integrates Kepler with
  - Massively parallel execution mechanism
  - Special purpose function of Nimrod/G/O/E
  - General purpose workflows from Kepler
Workflow Threading

- Nimrod parameter combinations can be viewed as threads
- Multi-threaded workflows allow independent sequences in a workflow to run concurrently
  - This might be the whole workflow, or part of the workflow
- Tokens in different threads do not interact with each other in the workflow
Complete Parameter Sweep

- Using a MATLAB actor provided by Kepler
- Local spawn
  - Multiple thread ran concurrently on a computer with 8 cores (2 x quads)
  - Workflow execution was just under 8 times faster
- Remote Spawn
  - 100’s – 1000’s of remote processes
Nimrod/EK Actors

- Actors for generating and analyzing designs
- Leverage concurrent infrastructure
Optimization with Nimrod/OK
Nimrod/OK Workflows

• Nimrod/K supports parallel execution
• General template for search
  – Built from key components
• Can mix and match optimization algorithms
Using different optimisers
Hybrid Optimization
Execution Engines
Nimrod over Clusters

Jobs / Nimrod experiment

Nimrod

Actuator, e.g., SGE, PBS, LSF, Condor

Local Batch System
Nimrod over Grids

• Advantages
  – Wide area elastic computing
  – Portal based point-of-presence independent of location of computational resources
  – Grid level security
  – Computational economy proposed
    • New scheduling and data challenges
  – Virtualization proposed (Based on .NET!)

• Leveraged Grid middleware
  – Globus, Legion, ad-hoc standards
Leveraging Cloud Infrastructure

• Centralisation is easier
  – (Clusters vs Grid)

• Virtualisation improves interoperability and scalability
  – Build once, run everywhere

• Computational economy, for real
  – Deadline driven
    • “I need this finished by Monday morning!”
  – Budget driven
    • “Here’s my credit card, do this as quickly and cheaply as possible.”

• Cloud bursting
  – Scale-out to supplement locally and nationally available resources
Nimrod’s Grid Economy is Cloud bursting

<table>
<thead>
<tr>
<th>Resource</th>
<th>#jobs completed</th>
<th>Total job time (h:m:s)</th>
<th>μ / σ Job runtime (mins)</th>
</tr>
</thead>
<tbody>
<tr>
<td>East</td>
<td>818</td>
<td>1245:37:23</td>
<td>91/5.7</td>
</tr>
<tr>
<td>EC2</td>
<td>613</td>
<td>683:34:05</td>
<td>67/14.2</td>
</tr>
</tbody>
</table>
Workways

Interacting with Workflows
WorkWays

• Ease of use of Science Gateway
• Workflows as service
• IO through portlets
• Extensibility
  – Different IO mechanisms, protocols and topology
  – Different UI clients
• Currently Kepler as the workflow engine
Motivation

• Support human-in-the-loop workflows
  – Ability to perform IO operations with a continuously running workflow

• Benefits
  – Insights into workflow execution
  – Steer the execution
WorkWays

• Leverage various existing technologies
  – Kepler workflow
  – Nimrod family toolkit
  – Liferay portal
• Virtual desktops
• AAF
• Various Web-based visualization tools
  – Parallel coordinates
  – Para-view Web
IOActor

- Generic actor
  - Simplify the creation of (new) IO actors
  - Instantiate an IOActor & provide the actor definition

- IOActor definition
  - Actor name
  - Number of (supported) clients
  - Operation: input/output/inout
  - Additional information

```json
{
    "actor": "ImageInOutActor",
    "operation": "inout",
    "input_type": "text",
    "prompt": "Choose an area in the image",
    "output_type": "binary",
    "display_type": "image",
    "action": "subarea",
    "wait_client": true,
    "wait_for_input": true
}
```
IOPortlet

- Web UI client
- Vaadin framework
  - framework for building rich Web applications
- JSR-286 portlet
  - UI elements generated based on requests from connected IOActor
  - Limited UI elements
Micro-mixer design
Optimal Topology
Parallel Coordinates
Micromixer Optimization
Micromixer optimization Workflow
Airfoil Design
2D airfoil optimization
2D airfoil optimization
Conclusions

• Workflows are useful for scripting complex computational science and engineering problems
• Conceptually easy to add optimization
• User interaction requires new workflow actors
• Integration to a Science Gateway allows very powerful workflows to be exposed to wider communities.