Unified Model Performance on the NEC SX-6

Paul Selwood
Introduction
- National Weather Service
  - Global and Local Area

- Climate Prediction (Hadley Centre)

- Operational and Research activities

- Relocated to Exeter from Bracknell 2003/4

- 150th Anniversary in 2004
Supercomputers

- 1996-2004: Cray T3E
- 2003: NEC SX-6
  - Operational May 2004
  - Currently 2 x 15 node SX-6 systems
  - May 2005 - additional 15 node SX-8
Single code used for NWP forecast and climate prediction
- N216L38 Global (40km in 2005)
- 20km European (12 km in 2005)
- 12km UK model (4km in 2005)

Submodels (atmosphere, ocean ...)

Grid-point model (regular lat-long)
- non-hydrostatic, semi-implicit, Semi-Lagrangian dynamics
- Arakawa C-grid, Charney-Philips vertical staggering
- Route to disk depends on packet size
  - < 64KB nfs (slow)
  - > 64KB GFS (fast)

- System buffering only available for Fortran I/O – Unified Model uses C.
Solution
- User buffering of output data.
- Removal of unnecessary opens and closes.

Subsequent buffering of headers increase I/O rate to > 140 MB/s.

Considering use of locally attached disk in future (> 1GB/s seen).

<table>
<thead>
<tr>
<th>STASH version</th>
<th>Time (s)</th>
<th>Rate (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>212</td>
<td>40</td>
</tr>
<tr>
<td>Buffered</td>
<td>114</td>
<td>80</td>
</tr>
<tr>
<td>Removed Open/Close</td>
<td>69</td>
<td>110</td>
</tr>
</tbody>
</table>
Semi-Lagrangian Advection
Semi-Lagrangian advection demonstrating load balance problems

<table>
<thead>
<tr>
<th>Routine</th>
<th>Max (s)</th>
<th>Min (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theta departure points</td>
<td>29.67</td>
<td>18.46</td>
</tr>
<tr>
<td>Wind departure points</td>
<td>57.55</td>
<td>35.84</td>
</tr>
</tbody>
</table>

N216L38, 2 day forecast
Interpolation of Departure points

- Need to calculate departure points for variables held at θ, u and v points.
- Currently call the departure point routine 3 times, once for each variable type.
- The departure point routine is expensive and poorly load balanced as extra calculations are done for latitudes >80°.
- Can improve runtime and load balance by calculating the departure point for a θ point and then interpolating to the u and v points.
- Approximation only works for higher resolutions.
### Interpolation of Departure Points

- Much simpler algorithm is generally cheaper
- Perfect load balance for wind calculations
- Load balance remains a (hard) problem for theta point calculations

<table>
<thead>
<tr>
<th>Routine</th>
<th>Max (s)</th>
<th>Min (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wind departure points (original)</td>
<td>57.55</td>
<td>35.84</td>
</tr>
<tr>
<td>Wind departure points (new)</td>
<td>14.25</td>
<td>14.23</td>
</tr>
</tbody>
</table>
Convection
Convection load balance is an old problem which we had a solution for on the T3E.

- Needed reworking as coded directly in SHMEM
- Segment based dynamic load balancing
  - Having enough segments to give good balance would harm vector performance
  - Data in segments could be sparse – data moved that wasn’t used
- Moved too much data – approach wouldn’t be appropriate with SX-6 compute/communicate ratio.
Convection – Deep and Shallow
- Load balance separately for deep/shallow
- Calculations per point assumed to be equal
- Tuneable threshold for data sizes to move
- Compress data to active points
- Communications use one-sided MPI
Shallow and Deep Convection Load Balance

![Graph 1](image1.png)

![Graph 2](image2.png)

Original vs. Balanced Load Distribution

Time (s) vs. CPU
Short Wave Radiation
Load Balancing – Short Wave Radiation

- Similar approach to convection
- Short Wave flux calculations take 90% of time so only this is balanced
- Computation is relatively more expensive than for convection
One sided MPI communications have advantages on the SX-6

- No need to explicitly schedule communications (MPI_Get from the under-loaded CPU)
- Speed! - Example is processor pairs exchanging 1000 words

<table>
<thead>
<tr>
<th>Method</th>
<th>Time on 2 Nodes (16 CPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI_Get (global memory)</td>
<td>32.3 msec</td>
</tr>
<tr>
<td>Buffered Send/Recv</td>
<td>6.2 msec</td>
</tr>
<tr>
<td>Individual Send/Recv</td>
<td>21.7 msec</td>
</tr>
<tr>
<td>Buffered MPI_Get (global memory)</td>
<td>5.8 msec</td>
</tr>
</tbody>
</table>
Many unnecessary barriers in code
  - T3E relics!
  - Easily removed for immediate benefit.

Gathering/Scattering 3D fields level-by-level
  - Optimised by copying into temporary buffers and doing one communication per CPU pair
  - Halves cost of these communications

>6000 halo exchanges in 6 hour forecast
  - Can we amalgamate any?
  - Can we remove any?
Questions?