Generic parallelization strategies for data assimilation

Nils van Velzen, Martin Verlaan
Outline

- OpenDA
- Coupling of model to DA method
- Parallelism of model and EnKF
- Parallelism for black box models
- Conclusions
OpenDA

- What is OpenDA?
  - A generic toolbox for data-assimilation
  - Set of interfaces that define interactions between components
  - Library of data-assimilation algorithms
  - Open source

- Why OpenDA?
  - More efficient than development for each application
  - Shared knowledge between applications
  - Development of algorithms with e.g. universities
  - Easier to test
OpenDA

- Object oriented design
  - Classes, software building blocks
  - Interface (set of functions suitable for all models, observations, etc)
• Formal form of a model
\[ \frac{dx}{dt} = M(x(t), u(t), p, w(t)) \]

• State of model instance \( x(t), u(t), p, w(t) \)

• Instance state cannot be directly changed only through the methods like:

  GetState, AxpyState, Compute…

• Algorithm has no knowledge on model internals
Generic parallelization strategies for data assimilation
EnKF semi parallel

- Only parallelize model steps:
  - Not scalable, often sufficient

\[
\xi_i^f(t_k) = M(\xi_i^a(t_{k-1})) + w_i(t_k)
\]

\[
x_i^f(t_k) = \frac{1}{N} \xi_i^f(t_k)
\]

\[
E_i^f(t_k) = \begin{bmatrix}
\xi_1^f(t_k) - x_1^f(t_k), \\
\xi_2^f(t_k) - x_2^f(t_k), \\
\vdots \\
\xi_N^f(t_k) - x_N^f(t_k)
\end{bmatrix}
\]

\[
\xi_i^a(t_k) = \xi_i^f(t_k) + K(t_k)[y(t_k) - H(t_k)\xi_i^f(t_k) + v_i(t_k)]
\]
EnKF semi parallel

- Lotos-euros air quality model
EnKF semi parallel

- Generic semi parallel due to OO concepts
Generics parallelization strategies for data assimilation
## EnKF parallel

### Column wise distribution

<table>
<thead>
<tr>
<th></th>
<th>Separate</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{nN}{p}$</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>$n \log_2(p)$</td>
<td></td>
<td>$n \log_2(p)$</td>
</tr>
<tr>
<td>$\frac{nN}{p} + nN \log_2(p-1)$</td>
<td></td>
<td>$nN \log_2(p-1)$</td>
</tr>
</tbody>
</table>

### Row wise distribution

<table>
<thead>
<tr>
<th></th>
<th>Separate</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_m + \frac{nN}{p}$</td>
<td></td>
<td>$C_m$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\frac{nN}{p}$</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
### Column wise distribution

<table>
<thead>
<tr>
<th>Separate</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{nN}{p} )</td>
<td>0</td>
</tr>
<tr>
<td>( n \log_2(p) )</td>
<td>( n \log_2(p) )</td>
</tr>
<tr>
<td>( \frac{nN}{p} + n \log_2(p-1) )</td>
<td>( nN \log_2(p-1) )</td>
</tr>
</tbody>
</table>

### Row wise distribution

<table>
<thead>
<tr>
<th>Separate</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C_m + \frac{nN}{p} )</td>
<td>( C_m )</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \frac{nN}{p} )</td>
<td>0</td>
</tr>
</tbody>
</table>
EnKF parallel

• WAQUA shallow water model
• Comparison of parallelization strategies for RRSQRT (Roest et al.)
EnKF parallel

- Full parallel using OO concepts:
  - All filters run same code
  - State vectors are distributed (parallel vector)
  - Operations on parallel vectors by a parallel vector implementation
  - NO need to change the model and filter code
  - All complexities hidden in generic support layers/implementations
Parallel computing with black box models

- Black box models
  - No change to model code
  - Data exchange using files
  - Note: disk can be slow/more data written than needed
Parallel computing with black box models

- Swan model for wind generated waves
  - Operational model + DA for the north sea
  - Black box model
  - 1 hour, 8 cpu's 10 min
    50% IO
  - EnKF implementation (?)
  - Parallelization of filter
Parallel computing with black box models

- Black box model is normal model
  - Semi parallel+full parallel
  - Note disk speed,
  - Use local disks:
    - Faster
    - No sequential bottleneck
Conclusions

• Generic parallelization strategies due to object oriented programming concepts.
• Single filter implementation for sequential as parallel computing
• EnKF like algorithms need combination of parallel strategies
• Black box models and IO can be parallelized as well