Algorithm and Library Software Design – Challenges for Tera, Peta, and Future Exascale Computing

Bo Kågström

Department of Computing Science and High Performance Computing Center North (HPC2N) Umeå University, Sweden

eSSENCE Academy 2012 – Sigtuna, Oct. 16-17
Outline

• Parallel computing
  – what, why and how?
• Parallel computer systems
  – today and the future
• European Exascale Software Initiative (ESSI)
• Algorithm and software design - critical issues
  – Aims: scalability, efficiency, portability, robustness
Can all problems be solved in parallel?

Dig a hole:

Can be done in parallel? Yes ☑ No ☐

Dig a ditch:

Can be done in parallel? Yes ☑ No ☐

Brick a wall:

Can a stone be placed Anywhere at anytime? Yes ☑ No ☐
Parallel computing

A collection of processors (nodes) that communicate and cooperate to solve a large problem fast and reliably.

Communication media

- Where is the data placed?
- Is the workload balanced?
- Is the allocation and execution of work tasks optimal?
Parallel algorithm design

• Data locality
  Where is the data placed?
  Should be close to "processor" that needs it!

• Load balance of computational work
  Is the workload balanced?
  All processors should do same amount of work!

• Schedule to minimize idle time
  Are the work tasks done in an optimal order?
  Remove redundant synchronization overhead!

In general, NP-complete!
Parallelism everywhere – new great challenges!

• Parallel architectures
  – From laptops to supercomputers
  – Paradigms: SM (e.g., multicore), DM, hybrid, graphics processing units (GPU)

• Great increasing demand for methods, tools, algorithms, languages and (library) software which support massive parallelism effectively!

Applications demand unlimited amount of resources (flops, bytes):

\[ Tera = 10^{12} \quad \rightarrow \quad Peta = 10^{15} \quad \rightarrow \quad Exa = 10^{18} \]
Key to performance?

To understand the algorithm and architecture interaction!
Parallel system of today

- Multi-level parallelism in architecture designs
- Multi-level memory hierarchies

Parallelism at processor level:

Source: ICL at UT
Parallel system of today

**AMD OPTERON™ 6200 SERIES PROCESSOR ("INTERLAGOS")**

Same platform as AMD Opteron™ 6100 Series processor.

Source: AMD

© HPC2N 2012

eSSENCE Academy 2012
Parallel system of today

Node Board:

• Several chips/sockets per node
• Accelerators

Source: ICL at UT
Parallel system of today

Source: ICL at UT
Parallel system of today

Combination of **shared memory** and **distributed memory programming**

Source: ICL at UT
Future parallel systems?

Different Classes of Chips
- Home
- Games / Graphics
- Business
- Scientific

Source: ICL at UT

© HPC2N 2012

eSSENCE Academy 2012
European Exascale Software Initiative

• **Objective:**
  To build a European vision and roadmap to address the challenges of the new generation of massively parallel systems composed of millions of heterogeneous cores which will provide multi-Petaflop performances in the next few years and Exaflop performances in 2020.

• Co-funded by the European Commission.

• **IESP - International Exascale Software Project**
  co-funded by DOE and NSF in USA.
**WP3: Application Grand Challenges**  
WP Chair: Stéphane Requena (GENCI)

<table>
<thead>
<tr>
<th>WG 3.1</th>
<th>Industrial and Engineering Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>WG 3.2</td>
<td>Weather, Climatology and Earth Sciences</td>
</tr>
<tr>
<td>WG 3.3</td>
<td>Fundamental Sciences (Chemistry, Physics)</td>
</tr>
<tr>
<td>WG 3.4</td>
<td>Life Science and Health</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chair</th>
</tr>
</thead>
<tbody>
<tr>
<td>Philippe Ricoux (TOTAL)</td>
</tr>
<tr>
<td>Giovanni Aloisio (ENES-CMCC)</td>
</tr>
<tr>
<td>Godehard Sutmann (CECAM)</td>
</tr>
<tr>
<td>Modesto Orozco (BSC)</td>
</tr>
</tbody>
</table>

**WP4: Enabling Technologies for Exaflop Computing**  
WP Chair: Bernd Mohr (Jülich)

<table>
<thead>
<tr>
<th>WG 4.1</th>
<th>Hardware Roadmaps, Links with Vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>WG 4.2</td>
<td>Software Eco-system</td>
</tr>
<tr>
<td>WG 4.3</td>
<td>Numerical Libraries, Software and Algorithms</td>
</tr>
<tr>
<td>WG 4.4</td>
<td>Scientific Software Engineering</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chair</th>
</tr>
</thead>
<tbody>
<tr>
<td>Herbert Huber (STRATOS-LRZ)</td>
</tr>
<tr>
<td>Franck Cappello (INRIA-UIUC)</td>
</tr>
<tr>
<td>Iain Duff (STFC-RAL and CERFACS)</td>
</tr>
<tr>
<td>Mike Ashworth (STFC-DL)</td>
</tr>
</tbody>
</table>
Numerical Libraries, Software and Algorithms:

- Dense linear algebra
- Graph and hypergraph partitioning
- Sparse direct methods
- Iterative methods for sparse matrices
- Eigenvalue problems, model reduction
- Optimization
- Control of complex systems
- Structured and unstructured grids

Much interdependence between areas.

Importance of also working at Tera- and Petascale levels!
Algorithm and software design – critical issues

• Reduce synchronization overhead
  – Dynamic scheduling and load balancing
• Hide and avoid communication and data movement
  – Blocking and remapping of data
• Use of mixed precision arithmetic
  – Refinement techniques
  – 2x speed of ops and 2x speed for data movement
• Reproducibility of results
  – Can not in general be guaranteed!
  – Error estimation of results
Algorithm and software design – critical issues

• Fault resilience (tolerance)
  – Recover from HW failures
  – Checkpointing, recomputation, redundant computation
  – Effect on accuracy and performance (speed) – a general trade-off issue!

• Autotuning and performance optimization
  – Build intelligence into software to adapt to hardware

• Energy aware algorithms
  – Frequence of cores can be controlled to save energy
  – Dynamic Voltage and Frequency Scaling (DVFS)
Application software using legacy HPC software

- Column-major (CM) and row-major (RM) storage formats are typically used by compilers
- BLAS, LAPACK, ScaLAPACK, etc. assume that inputs are in CM format
  - Blocks are scattered in memory!
  - Remedy: Use blocked data layouts internally!
Blocked storage formats

Standard formats
- **CM** Column-Major
- **RM** Row-Major
- Inefficient block access

Blocked formats
- **CCRB** Column-Column RB
- **CRRB** Column-Row RB
- **RCRB** Row-Column RB
- **RRRB** Row-Row RB
- **Blocks** are stored contiguously in memory

(RB = Rectangular Block)
Application with mixed use of HPC library routines

- CM → CCRB
- CCRB → RRRB
- RRRB → CM
Library design methodology

- Globally: explicit blocking and message passing for 2D block-cyclic data layouts

- Locally: explicit or recursive blocking and multi-threading for SMP/multicore nodes
Sample framework

- Global level: Static distribution of data and work
- Node level: Dynamic scheduling of work
Welcome to visit our poster!

Efficient and Reliable HPC Algorithms for Matrix Computations in Applications

Sample projects

Björn Adlerborn, Andrii Dmytryshyn, Stefan Johansson, Lars Karlsson
Carl Christian Kjelgaard Mikkelsen, Bo Kågström (eSENCE-PI), Meiyue Shao

Björn
Andrii
Stefan
Lars
Carl Christian
Bo
Meiyue
Thank you!
The HPC2N molecule

From macro- to micro- and further to nano-scale using Density Functional Theory!

DFT computation, semi-stable, binding energy 15eV; Sven Öberg, LTU