

Experiences with Shared Memory Programming in Scientific Computing

Gudula Runger

Department of Computer Science



CHEMNITZ UNIVERSITY
OF TECHNOLOGY

Dagstuhl-Seminar 07361, September 2007
Programming Models for Ubiquitous Parallelism



Parallel computing at Chemnitz

- Technical University:
Mechanical and electrical engineering, computer science, physics, mathematics
- one of the main research areas:
 - ▶ scientific computing and high performance computing
 - ▶ parallel and distributed systems (Dept. of computer science)
 - ▶ Cluster systems
- Interdisciplinary research projects in parallelism and scientific computing (including a collaborative research SFB 393; several smaller project groups)



Outline

- 1 Programming Environments and Applications
- 2 Some experiments
 - Mixed Parallelism in ODE methods
 - Communication between MPI processes
 - Task Pool Teams
- 3 Conclusion



Irregular and adaptive Applications

- Irregular applications:
 - ▶ depend on the input (e.g. sparse problems or some ODE methods)
 - ▶ unpredictable irregular dependencies between tasks
 - ▶ hierarchically growing data structures (e.g. FMM)
 - ▶ adaptive algorithms (adaptive FEM)
- Problems:
 - ▶ load imbalances
 - ▶ asynchronous communication requirements/interactions (communication requirements or partners are not known in advance)



Applications

- fast multipole method (FMM], Comp. Chemistry
- adaptive finite elements methods (FEM) , Math. Dept.
- HRA
- dynamical systems , Physics Dept.
- anomalous diffusion in porous media (Sierpinski carpets), Comp. Physics
- mixed parallelism (for ODE solver)

—→

shared memory programming was the choice in many cases but also **message passing** or **mixed parallelism** if required

Support for parallel applications

- Start with the application:
(sequential or parallel programs grown over many years)
- Analysis or specific problems
- Task oriented decomposition, top-down-design
- Suitable programming model for the application
- Tools, libraries, models and methodologies to support program development for parallel platforms

Programming Environments for mixed parallelism

• Multiprocessor-Task Programming:

coarse-grain task-graph

- ▶ **compiler tool TwoL** (two level parallelism):
specification, **static** scheduling of M-Tasks, transformation
Advantage: fixed data distribution and redistribution → data distribution **cost** are known for scheduling
- ▶ **runtime library Tlib:**
executable specification; **dynamic** scheduling of M-Task
Advantage: recursive, hierarchical M-task structure;
divide&conquer algorithms
- ▶ **runtime system TGrid:** Task-graph execution on heterogenous systems

• Task Pool Teams:

Task Pools for shared memory combined with communication

Multi-core

- All programs work and give the correct results;
(models, languages, libraries, programs)
 - ▶ MPI is ported to shared memory or heterogeneous machines
 - ▶ Shared memory (Pthreads or OpenMP)
- people from scientific computing just see more parallelism;
they do not consider multi-core as a new programming paradigm
- **but:** the performance behavior can be different and more diverse

Outline

1 Programming Environments and Applications

2 **Some experiments**

- Mixed Parallelism in ODE methods
- Communication between MPI processes
- Task Pool Teams

3 Conclusion

General linear method

Computation of **k stage values** in time step n at time t_n

$$Y_n = \underbrace{(y_{n1}, \dots, y_{nk})}_{\text{stage values}} \quad \text{vector of size } \mathbf{d} \cdot \mathbf{k}$$

Stage value y_{ni} is the approximation of $y(t_n + a_i h)$
 Computation in each time step

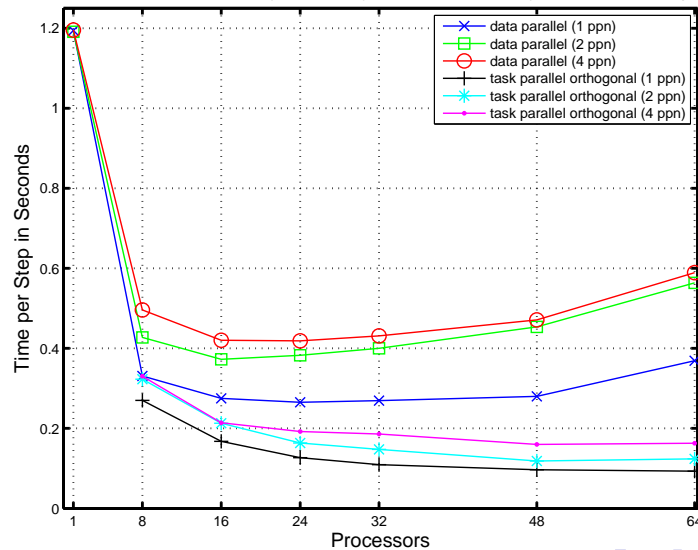
$$Y_{n+1} = \underbrace{(R \otimes I)}_{\text{Kronecker}} Y_n + h(S \otimes I)F(Y_n) + h(T \otimes I)F(Y_{n+1})$$

with R, S, T $\mathbf{k} \times \mathbf{k}$ matrices and I $\mathbf{d} \times \mathbf{d}$ matrix
 $F(Y_n) = (f(y_{n1}), \dots, f(y_{nk}))$ vector of size $\mathbf{d} \cdot \mathbf{k}$

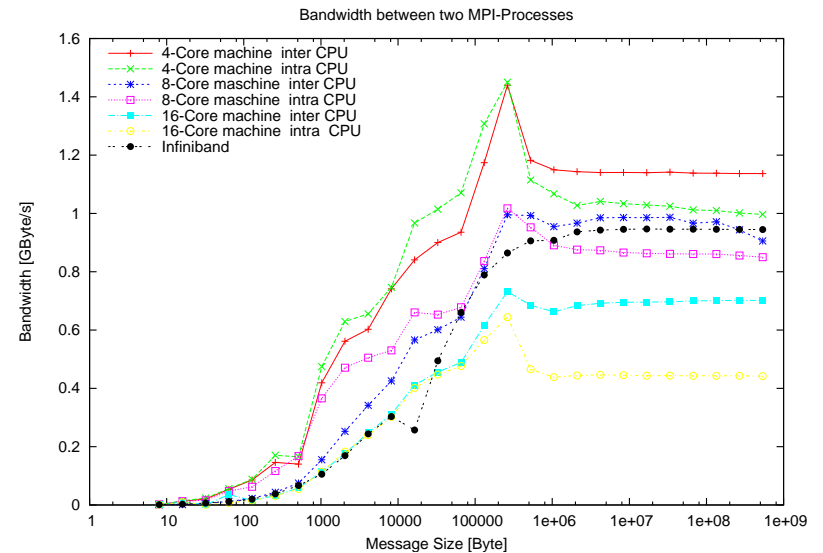
MVAPICH2 on Cluster with Infiniband

- CHIC – 538 dual SMP, with Opteron 8218 Dual-Core @2,6GHz

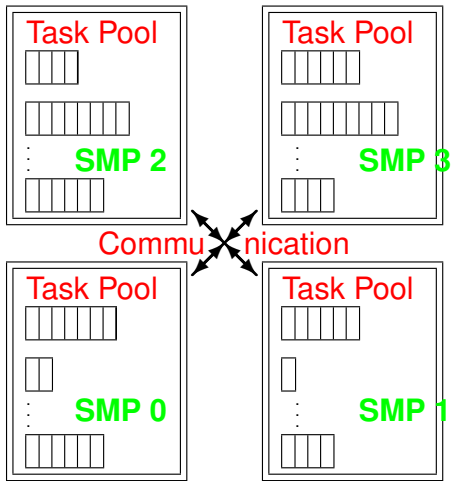
PABM–method with Brusselator(n=500000) for K=8 on CHIC (same number of processes)



Varying communication costs



Hybrid Programming model



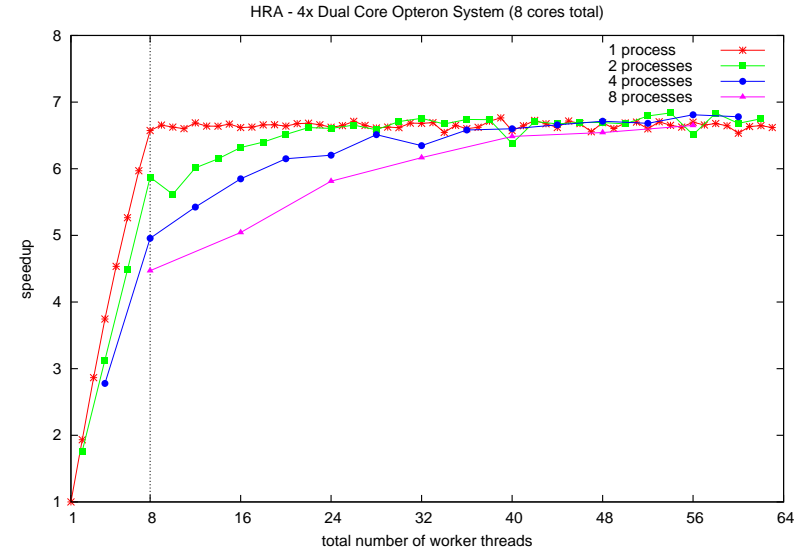
Multithreaded programming with Task Pools (Pthreads)

Message Passing between nodes with MPI

Outline

- 1 Programming Environments and Applications
- 2 Some experiments
 - Mixed Parallelism in ODE methods
 - Communication between MPI processes
 - Task Pool Teams
- 3 **Conclusion**

HRA on SMP cluster – 4X2 cores



More/new interest in parallel programming

- Multi-cores are no big change for scientific computing
 - research concerning performance issues
- Other application areas:
 - ▶ e-Government (long running processes)
 - ▶ e-business (legacy problems in business software)
 - ▶ software in engineering /digital manufacturing
 - interest in parallel programming/multi-core programming
 - new opportunities, e.g. additional functionalities without delay for the user
- Teaching:
 - Parallel programming course (additional course in the third year)
 - attracts many more students

For more information

ruenger@informatik.tu-chemnitz.de

www.tu-chemnitz.de/informatik/PI

