



Parallelization of ECEPP/3 in SMMP

J. H. Meinke, U. H. E. Hansmann

published in

From Computational Biophysics to Systems Biology (CBSB07),
Proceedings of the NIC Workshop 2007,
Ulrich H. E. Hansmann, Jan Meinke, Sandipan Mohanty,
Olav Zimmermann (Editors),
John von Neumann Institute for Computing, Jülich,
NIC Series, Vol. 36, ISBN 978-3-9810843-2-0, pp. 219-222, 2007.

© 2007 by John von Neumann Institute for Computing
Permission to make digital or hard copies of portions of this work for
personal or classroom use is granted provided that the copies are not
made or distributed for profit or commercial advantage and that copies
bear this notice and the full citation on the first page. To copy otherwise
requires prior specific permission by the publisher mentioned above.

<http://www.fz-juelich.de/nic-series/volume36>

Parallelization of ECEPP/3 in SMMP

Jan H. Meinke¹ and Ulrich H. E. Hansmann^{1,2}

¹ John von Neumann Institute for Computing,
Research Centre Jülich, 52425 Jülich, Germany
E-mail: {j.meinke, u.hansmann}@fz-juelich.de

² Department of Physics,
Michigan Technological University,
Houghton, MI 49931, USA

The power consumption of modern processors makes it difficult to increase their clock speed further. Even in the PC market CPU manufacturers now include multiple compute cores on a single chip to improve performance and keep up with Moore's law, a trend likely to continue. This is even more important in high performance computing, where cooling and electricity bills are becoming a large issue. The compactness and low power consumption of an IBM BlueGene, e.g. is only possible because of CPUs with moderate clock rates.

Protein folding is computationally hard. To take advantage of the increasing number of compute cores and reduce the time to solution, we need to parallelize our codes, especially the time intensive calculation of the energy function, and minimize the serial portions of the code. We are presenting the parallelization implemented in SMMP for the ECEPP/3 force and the resulting scaling behavior on various platforms including BlueGene/L. Combining the parallel energy function with parallel tempering, simulations scale up to thousands of processors.

1 Introduction

Moore predicted that the number of transistors on the cheapest chips doubles every 24 months.¹ This prediction has held for over 40 years. Combined with increasing clock speeds, this development led to ever faster CPUs, faster-running serial codes, and larger power consumption. One way to decrease the power consumption is to run the processor at a lower clockspeed. Using the same chip, doubling the clock speed results in an approximately 8 times higher power consumption. In the mid 1990s, an Intel Pentium I running at 133 MHz used about 11 W. Modern processors can use more than 100 W. With the introduction of multi-core processor to the consumer PC market in 2005, Intel moved away from providing processors that run serial code faster and forced many application developers to think about parallel programming. In high-performance computing, where the number of processors can go into the thousands, power consumption and cooling are important factors. With the development of BlueGene/L, IBM decided to use comparatively slow processors — with a correspondingly low power consumption — combined with fast networking, to build a very compact system that scales to more than a 100000 processors and a peak performance of more than 350 TFLOPS. This high performance comes at a price. Comparing the single processor performance of a BG/L with an AMD Opteron at 2.4 GHz, we find that our application runs more than 7 times faster on the Opteron than BG/L. To make up for this loss in performance and take advantage of our BlueGene/L JUBL and the increasing size of our smaller PC clusters, we parallelized the calculation of the energy in our protein simulation package SMMP.^{2,3} In combination with parallel tempering, we now regularly run our simulation on 4096 processors. Here, we

present the application scaling on 4 different platforms and give some pointers for efficient parallelization on BlueGene/L.

2 Distributing the Work

For these measurements we used the ECEPP/3 force field.⁴ The energy function that needs to be calculated is

$$E_{\text{ECEPP/3}} = \sum_{(i,j)} \frac{332q_iq_j}{\epsilon r_{ij}} + \sum_{(i,j)} \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^6} \right) + \sum_{(i,j)} \left(\frac{C_{ij}}{r_{ij}^{12}} - \frac{D_{ij}}{r_{ij}^{10}} \right) + \sum_l U_l (1 \pm \cos(n_l \xi_l)) , \quad (1)$$

where i and j are indices of atoms, r_{ij} is the distance between atom i and atom j , and l is the index of a dihedral angle in the protein chain. All other variables (q_i , A_{ij} , B_{ij} , C_{ij} , and D_{ij}) are parameters of the force field.

In SMMP, every atom is associated with a dihedral angle. We used this relation to distribute the interactions as evenly across processors as possible without regard to spatial proximity.

3 Setup and Scaling

We ran our benchmark on 4 different platforms: JUMP, an IBM p690 cluster with 32 processors and 112 GB of shared memory per node; JUBL, an IBM BlueGene/L with 8 racks and a total of 16384 Power4 processor at 700 MHz; JULI a PC cluster using dual-core PowerPC 970MP processors at 2.5 GHz with an InfiniPath network and NICOLE, an Opteron based PC cluster with a clock speed of 2.4 GHz using Infiniband networking. Except for the setup of the communicators used for the energy calculation on BG/L, we used the same source code for all measurements. We performed 50 sweeps of a Monte Carlo simulation of the designed protein TOP7⁵ starting from a stretched chain. Data was written to disk every 10 sweeps. On JUBL, we used multiple replicas in parallel with the indicated number of processors per replica to fill a half plane (512 processors).

Figure 1 shows walltime and scaling for the various machines. The execution time on a single processor ranges from about 18 min on JULI to about 2 h on JUBL. The lowest execution time ranges from 81 s on JUMP to 269 s on JUBL with 64 processors per replica. The maximum speedup is 25 on JUBL with 64 processors.

For JUMP, JULI, and NICOLE we used MPI's default processor assignment. On JUBL, however, this approach leads to a sub-optimal distribution of the processors. BG/L has a cubic geometry. By default, the rank of a processors increases first along x, than y, and finally z. This leads to a planar distribution of processors (cf. left picture in figure 2). Instead of the default, one should make communicators as cubic as possible (see right picture in figure 2) unless the problem geometry suggests a different approach.

4 Conclusions

For protein simulations, distributing the calculation of the interactions evenly across a number of processors can be done efficiently without worrying about the spatial distribution of

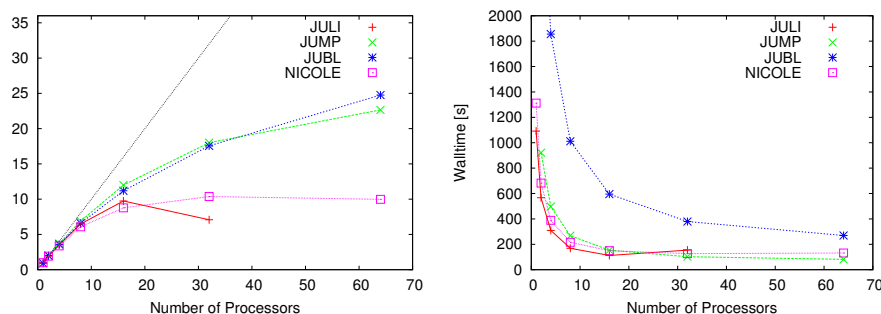


Figure 1. Strong scaling behavior. This figure shows the walltime and corresponding parallel scaling vs. number of processors on JULI, JUMP, and JUBL. For the benchmark, we performed 50 sweeps of a Monte Carlo simulation of Top7,⁵ a 92 residue protein with 1477 atoms. Data and configurations were written to disk after every 10 sweeps. On JUBL, we used multiple replicas in parallel with the indicated number of processors per replica to fill a half plane.

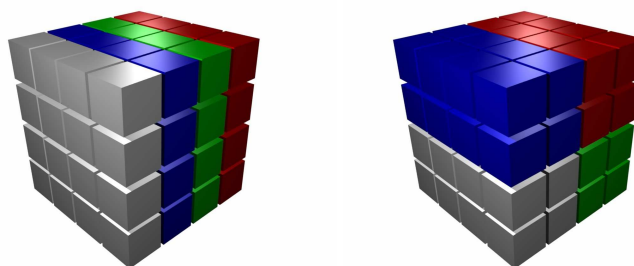


Figure 2. Two processor arrangements on BG/L. To combine parallel tempering with a parallel calculation of the energy for each replica, we define our own communicators for the energy calculation. On the left-hand side, we show the default arrangement for a consecutive number of processors on a $4 \times 4 \times 4$ partition with four replicas. Each replica is assigned to a plane of the partition. On the right, we show the most cubic arrangement possible. The cubic arrangement scales significantly better (up to $25\times$) than the planar one (up to $18\times$).

the atoms involved. The low cost per processor makes BlueGene/L an attractive platform for protein simulations. Using a cubic arrangement of 64 processors, we achieve a speedup of up to $25\times$ on BG/L. With the large number of processors available on JUBL, we can run simulations with 64 replicas at a quarter of the cost and at the same speed as on JUMP.

Acknowledgments

Development, computations and measurements were performed on computers of the John von Neumann Institute for Computing (NIC). U.H. received support from the National Science Foundation (USA) under grant number CHE-0313618.

References

1. Gordon E. Moore, *Cramming more components onto integrated circuits*, Electronics, **38**, no. 8, 114–117, 1965.
2. Frank Eisenmenger, Ulrich H. E. Hansmann, Shura Hayryan, and Chin-Kun Hu, *[SMMP] A modern package for simulation of proteins*, Comput. Phys. Commun., **138**, 192–212, aug 2001.
3. Frank. Eisenmenger, Ulrich. H. E. Hansmann, Shura Hayryan, and Chin-Kun Hu, *An enhanced version of SMMP—An Open-Source Software Package for Simulation of Proteins*, Comput. Phys. Commun., **174**, 422–429, 2006.
4. G. Nemethy, K. D. Gibson, K. A. Palmer, C. N. Yoon, G. Paterlini, A. Zagari, S. Rumsey, and H. A. Scheraga, *Energy parameters in polypeptides. 10. Improved geometrical parameters and nonbonded interactions for use in the ECEPP/3 algorithm, with application to proline-containing peptides*, J. Phys. Chem., **96**, no. 15, 6472 – 6484, 1992.
5. B. Kuhlman, G. Dantas, G. C. Ireton, G. Varani, B. L. Stoddard, and D. Baker, *Design of a Novel Globular Protein Fold with Atomic-Level Accuracy*, Science, **302**, no. 5649, 1364 – 1368, 2003.