Jül-4351

Institute for Advanced Simulation (IAS)
Jülich Supercomputing Centre (JSC)

High-Level Support Initiative of the JSC Simulation Laboratories 2011

P. Gibbon, L. Arnold, D. Brömmel, V. Chihaia, S. Grießbach, R. Halver, L. Hoffmann, A. Karmakar, J. Meinke, S. Mohanty, Th. Müller, A. Schiller, G. Sutmann, O. Zimmermann

Mitglied der Helmholtz-Gemeinschaft

JÜLICH
FORSCHUNGSZENTRUM

High-Level Support Initiative of the JSC Simulation Laboratories 2011

P. Gibbon, L. Arnold, D. Brömmel, V. Chihaia, S. Grießbach, R. Halver, L. Hoffmann, A. Karmakar, J. Meinke, S. Mohanty, Th. Müller, A. Schiller, G. Sutmann, O. Zimmermann

Berichte des Forschungszentrums Jülich; 4351
ISSN 0944-2952
Institute for Advanced Simulation (IAS)
Jülich Supercomputing Centre (JSC)
Jül-4351

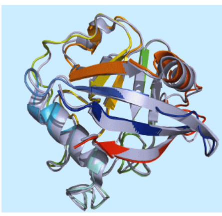
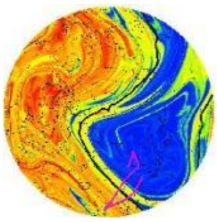
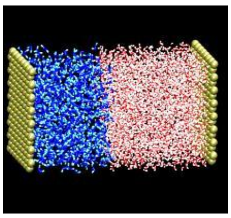
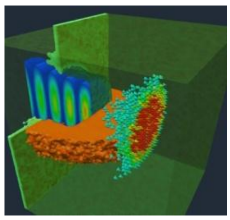
Vollständig frei verfügbar im Internet auf dem Jülicher Open Access Server (JUWEL)
unter http://www.fz-juelich.de/zb/juwel

Zu beziehen durch: Forschungszentrum Jülich GmbH · Zentralbibliothek, Verlag
D-52425 Jülich · Bundesrepublik Deutschland
☎ 02461 61-5220 · Telefax: 02461 61-6103 · e-mail: zb-publikation@fz-juelich.de

High-Level Support Initiative of the JSC Simulation Laboratories 2011

P. Gibbon, L. Arnold, D. Brömmel, V. Chihaia, S. Grießbach, R. Halver, L. Hoffmann, A. Karmakar, J. Meinke, S. Mohanty, Th. Müller, A. Schiller, G. Sutmann, O. Zimmermann

Institute for Advanced Simulation, Jülich Supercomputing Centre, Forschungszentrum Jülich

Contents

1. Introduction: SimLab Support Initiative 2011

A central component of the Helmholtz Programme Supercomputing (Topic 1: Computational Science and Mathematical Methods) is the development of advanced software support through the recently established Simulation Laboratories (SimLabs) at JSC. This was motivated by the realisation that application software is lagging behind HPC hardware developments, which are now at the Petaflop-scale and beyond: it is becoming increasingly difficult to fully exploit the potential of these machines with single applications.

Four such units have now been established at JSC in the fields of Computational Biology, Molecular Systems, Plasma Physics and Climate Science, which came on stream in early 2011. All four labs have been actively engaged with user groups from their respective communities over the past two years, through various workshops, informal cooperations and 3rd-party projects. To channel these activities into a more formal structure, a „Support Call“ was issued in September 2010, inviting current and potential users of the supercomputers in Jülich from these four research communities to apply for high-level support from the Simulation Labs (see Appendix A).

Expertise offered by SimLab staff included:

1. (Re)design of computational methods needed to exploit highly parallel architectures
2. Performance analysis and scaling improvement of application codes and workflows
3. Porting of new codes to the JUROPA and JUGENE systems

In this pilot phase, proposed work packages were permitted to take up to two person-months of SimLab staff resources. The applicant or members of his/her group were also expected to contribute an equivalent amount of manpower to the project, particularly where the work involved substantial code and algorithm redevelopment.

Altogether 21 proposals were received, five of which were either postponed for lack of manpower (the Climate Science Lab comprised only 1 member at that time), or rejected as being outside the scope of the SimLabs' expertise or mandate. Selection decisions were taken in a plenary session involving all four SimLabs, which ensured that proposals which fell outside the expertise of the lab to which it was directed could potentially be transferred to another SimLab, or passed on to one of the cross-sectional teams at JSC. Due to the relatively low number of proposals per staff member (on average each lab had two full-time staff members as of January 2011), it was decided not to request external reviews and ranking for this round. A summary of all proposals can be found in the table on page 5.

After the proposal assignment, there followed a series of face-to-face consultations between the SimLabs and principle investigators of each proposal (PIs) to establish a work plan and time-frame for each project. This was largely completed by December 2010 so that work could begin in earnest in early 2011.

2. Project summary

Project #	Short Title	Code(s)	Task(s)	Scaling	Goal	PM request	PM spent
SLBIO1101	Parallel NGS de-novo assembly	Velvet, Abyss, Ray	Parallelisation	1	>200	2.0	3.0
SLBIO1102	Multi-million atom MD simulations	GROMACS	Scaling; electrostatics	1k	>10k	0.5	0.5
SLBIO1103	Molecular dynamics simulations at constant pH	GROMACS	Scaling, redesign	1k	100k	1.0	0.0
SLBIO1104	Conformational preferences of p53 peptide	PROFASI	Module design	32k		0.5	0.5
						4.0	**4.0**
SLCS1101	Stochastic atmospheric chemistry simulations	EURAD-IM	parallel scaling	200	20k	2.0	0.5
SLCS1102	CLaMS chemistry solver upgrade	CLaMS	algorithm redesign	128	>128	2.0	0.5
SLCS1103	Water vapor retrievals from SCIAMACHY measurements	SCIATRAN	basic parallelisation	1	8	0.5	0.5
SLCS1104	Optimisation of 2-D PREMIER retrievals	JuRaSSiC	single core tuning	256	256	2.0	2.0
						6.5	**3.5**
SLMS1101	Simulations of large ionic molecular systems		Electrostatics, benchmarking	N/A	4k	2.0	1.0
SLMS1102	Parallel Monte Carlo Initial Structure Generation	EMC	Parallelisation	1	>8	2.0	0.5
SLMS1103	IBIsCO Optimisation	IBIsCO	Scaling	<64	>64	2.0	3.0
						6.0	**4.5**
SLPP1101	Performance analysis of the PIC code EPOCH	EPOCH	Scaling, I/O, load balancing	1k	10k	1.0	1.0
SLPP1102	Scaling of the PIC code PICLS on JUGENE	PICLS	Scaling, I/O, 3D	1k	4k	2.0	1.0
SLPP1103	Scaling of adaptive Multigrid solver racoon	racoon	Scaling, BG/P optimization	100	10k	1.0	1.5
SLPP1104	Hybrid parallelization of the transport code RALEF	RALEF-2D	Parallelisation	8	>8	1.0	0.5
SLPP1105	Optimization of the code system EIRENE-B2	B2/B2.5	Parallelisation	1	64	1.0	2.0
						6.0	**6.0**
SLMS1104	Hydrodynamic Modeling of Macromolecules	UltraScan	Grid enabling (UNICORE)	10k	JSC	2.0	
SLPP1106	Port of AMR code racoon to GPUs	racoon	Porting GPU	N/A	hybrid	1.0	
SLCS1105	Characterization of river-aquifer interaction	EnKF3d-SPRING	Parallelisation	1	1k	1.0	
SLCS1106	2D Crosshole GPR Full-waveform inversion	2D_CH_GPR_FWI	Scaling	50	150	1.0	
SLCS1107	3D On-ground GPR full-waveform inversion	3D_OG_ GPR_FWI	Parallelisation	1	250	1.0	

3. Reports

In this section summary accounts of the projects are given as of December 2011. By this point most of the projects had either been completed or had reached a point where a second proposal would be a more appropriate way of continuing.

3.1. Biology

In the year 2011 the Simulation Laboratory Biology processed the following four support call projects:

SLBIO1101	Prof. Peter Nürnberg, Dr. Peter Frommolt	Cologne Center for Genomics	Parallel NGS de-novo assembly
SLBIO1102	Dr. Carsten Kutzner	MPI Göttingen	Multi-million atom MD simulations
SLBIO1103	Dr. Gerrit Groenhoff	MPI Göttingen	Molecular dynamics simulations at constant pH
SLBIO1104	Prof. Stefan Wallin	Lund University, Sweden	Conformational preferences of p53 peptide

SLBIO1101: Parallel NGS de-novo assembly based on de Bruijn graphs

This project was aimed at enabling assembly of large eukaryote genomes from DNA short fragments typical for current Next Generation Sequencing (NGS) machines on commodity clusters that do not feature large amounts of shared memory. In particular a new genome project on a nematode genome should be supported by such a tool. One of the fastest high-quality assemblers is Velvet. It is based on de Bruijn graph data structures but works only in serial mode and requires large memory. It was planned to test and analyze the only open-source MPI parallel de-Bruijn-based assembler Abyss, and to transfer their implementation of parallel de Bruijn graphs to Velvet. A NIC computing grant was available for the project.

Download of test data of a human genome from the trace archive server took almost two weeks. While the source code of Velvet could be analyzed easily, the Abyss analysis proved to be much harder: first, the available version of Scalasca did not work with the code due to renaming of system libraries. The Scalasca developers from the JSC Performance Analysis group fixed this problem within a few days. The job submission system on JUROPA had a hard-wired limit on the command line length that was not sufficient for large assembly commands. Wrapping the call into a script did not circumvent the problem as Scalasca cannot analyze calls within scripts. (NB: the command-line limit problem was solved a few months after the project). Test runs on larger genomes revealed an unsatisfactory scaling behavior regarding memory and run-time. As a consequence only 2-4 threads could be used per JUROPA-node which also resulted in much higher number of node-hours than anticipated. Even with 256 nodes, assembly of a eucaryote genome could not be completed within 24 hours (=max. wallclock time for JUROPA) neither for Romanomermis (nematode genome, ~200 Mbp, low coverage) nor SRA000271 (human genome, ~3 Gbp, better coverage).

Analysis of Abyss using Scalasca revealed e.g. highly unbalanced workload (due to several k-mers that represent highly repetitive sequences and were assigned to the same node e.g. poly-A k-mers). Furthermore, by code analysis it became clear that the datatypes used in Abyss were incompatible with those in Velvet even though both used splay tables for lookup instead of hashing. As the necessary reimplementation was beyond the scope of the support project, and a new assembler (Ray), with more balanced workload but complex messaging structure had been recently introduced,

it was decided to set up a larger research project and apply for funding at BMBF. Funding for this project (NGSforHPC) has since been granted. The project started July 1, 2011 and the Simlab Biology core staff will contribute 19 PM, focusing on parallel assembly algorithms, in particular on the Intel MIC architecture. In total 3 PM were spent on the support project, including 1 PM for the BMBF application.

SLBIO1102: Multi-million atom molecular dynamics simulations

Gromacs is a widely used open-source code for performing molecular dynamics (MD) simulations. One of its distinguishing features has always been its good single-processor performance. A number of kernels have been implemented using hand-optimized assembler code for a variety of architectures including Intel processors and BlueGene's PowerPC 450. Until version 4 its parallel scaling was very weak and even current versions do not scale as well as other MD codes although Gromacs often remains competitive. The goal for this support project was to analyze the scaling behavior of Gromacs for a system with millions of atoms, and to determine cause of the bottleneck for large numbers of processors.

During a video conference with Carsten Kutzner and Gerrit Groenhoff (see Project SLBIO1103) in November, we decided to start with the scaling analysis. Carsten Kutzner would provide the data for a virus capsule with several million atoms and we would instrument Gromacs to enable the analysis of the scaling using Scalasca. We would then perform short runs on JUROPA to analyze the program behaviour.

Instrumenting a program for an analysis with Scalasca is often quite simple but for Gromacs it took some extra work due to the low level optimization. An analysis of the threaded version was not possible with Scalasca since Gromacs does not use OpenMP. This was not really a big issue since the emphasis of this project was on large number of processors and Gromacs does yet not support hybrid programming models with threads and MPI. Some test runs were performed including the new instrumentation. Unfortunately, prior commitments of the PI and his team meant that the necessary data for the virus capsule could not be prepared during the project timespan.

SLBIO1103: Molecular dynamics simulations at constant pH

The protonation state of a given amino acid depends on the pH of the solvent. For an MD simulation the protonation state is usually fixed during the setup of the simulation. For some amino acids the choice is not obvious. Histidine, for example, can have two different protonation states at physiological pH. Gerrit Groenhoff and co-workers developed a simulation method that allows the state to vary depending on the local electrostatic environment.

The goal of this project was to parallelize the calculation of the titration gradients. One PM was requested for this project. During the video conference in November 2010 (c.f. Project SLBIO1102) it was decided that the PI would port his code to the Gromacs 4.5.x code base while the SimLab Biology would first focus on the scaling analysis.

Unfortunately, more pressing commitments of the MPI Göttingen group meant that the necessary modifications to this version of the Gromacs code could not be completed during 2011, so (as with SLBIO1102) this project was shelved for a future call.

SLBIO1104: Conformational preferences of a p53 peptide

The aim of this project was to enable the Monte Carlo code ProFASi to simulate the binding process of a 15-residue peptide from the C-terminus of p53 with 4 different proteins. The conformational preferences of this peptide are interesting because it can bind to 4 different target proteins, thereby

assuming 4 different “native” structures. The software package ProFASi was chosen for this project because the force field implemented in ProFASi has been shown to successfully capture the folding and thermodynamic behaviour of many short peptide chains, through high statistics Monte Carlo simulations. Further, at least for short peptides, ProFASi is known to be relatively unbiased towards either helical or beta sheet secondary structures. These qualities are expected to be important for the simulation under consideration.

As the p53 peptide contains an acetylated Lysine residue, not present in the ProFASi model, support was needed to enable post-translational modifications in ProFASi, in particular Lysine acetylation.

The ProFASi code was extended to enable post-translational modifications. This involved redesigning the module named AA in ProFASi which contains a set of classes and functions for handling amino acids and N- and C-terminal capping groups. The new design permits post-translational modifications of amino acid side chains to be conveniently introduced as needed without touching the rest of ProFASi. For Acetylated-Lysine, ALY, geometrical parameters consistent with the bond-angle and bond-length approximations used in ProFASi were determined. A new ALY class was written, and it was tested with real simulations where ALY was part of a test peptide.

In addition an interaction potential was calibrated for acetylated Lysine. The calibration started with assigning some Lysine interaction strengths to Acetylated Lysine. ALY is uncharged, and hence dropped out of charged side-chain side-chain interactions. Hydrophobic strength was slightly increased relative to Lysine and the C atom in the new Acetyl group was introduced as one of the reference atoms for the contact based hydrophobicity term of ProFASi.

Simulations using modified code are currently running in Lund, Sweden and a joint paper is in preparation. The planned 0.5 PM were fully spent.

Summary

None of the 4 projects ran according to expectations. Stefan Wallin's project SLBIO1104 had a successful outcome but was not strictly supercomputing-related. The NGS project SLBIO1101 could not be completed as planned but did lead to a 3rd-party project that is a) likely to result in publications and b) is about parallelization and library design for HPC. Despite the setbacks with both Gromacs projects SLBIO1102/-1103, there is considerably interest in improving the overall scalability of this highly versatile, open-source code, so that the cooperation between JSC and MPI Göttingen is likely to be continued on a longer term basis.

The number of applicants in a subsequent call of the same type is likely to remain at a low level for Biology. As we see in the NIC proposals, most (and in Jülich almost all) biology-related HPC projects try to simulate larger protein structures via MD, using ready-made packages like Amber, Gromacs or NAMD. All these programs already have a host of dedicated developers with HPC experience. The NGS project originated from personal contacts, so did the ProFASi project.

For the SimLab Biology we see the following near time potential avenues to serve the Bio Community while increasing the potential for generating publications from support projects:

- Bundling compute time grants to support from the SimLab
- Developing ProFASi into a community code.
- Offering a fuller range of support for ‘wet-labs’ that do not develop code themselves but have proprietary experimental data and problems that can only be solved using HPC.
- Soliciting Bio HPC projects that need to be implemented or parallelized and look for fitting partners. This may become a necessity for a Data Lifecycle Support Lab and will also be interesting for projects that cannot be run on NIC/VSR grants, e.g. porting to accellerators.

3.2. Climate Science

In the year 2011 the following four support call projects were executed by the Simulation Laboratory Climate Science:

SLCS1101	Dr. Hendrick Elbern	University of Cologne	Stochastic atmospheric chemistry simulations
SLCS1102	Dr. Jens-Uwe Grooß	FZJ/IEK-7	CLaMS chemistry solver upgrade
SLCS1103	Dr. Katja Weigel	University of Bremen	Water vapor retrievals from SCIAMACHY limb measurements
SLCS1104	Prof. Dr. Martin Riese	FZJ/IEK-7	Optimisation of 2-D PREMIER retrievals

Three additional proposals were submitted but could not be supported.

SLCS1101: Parallel template for stochastic atmospheric chemistry simulations

Numerical simulations of the atmosphere rarely show error estimates. While meteorological forecasts of leading weather centres provide ensemble forecasts to predict simulation skills, the more fundamental step from deterministic modelling to the solution of stochastic partial differential equations is still in its infancy, partly because of a lack of computing power. Model parameters perturbed along our uncertainty of knowledge will allow Monte Carlo simulations to be performed which approximate the solution of the Fokker-Planck equation. This technique provides an indicator of the robustness or reliability of simulations.

The objective of the project was to provide a parallel kernel for ensemble modelling with the EURAD-IM atmospheric chemistry system, an approach used to obtain such error estimates. It is planned to develop a new software package for atmospheric ensemble simulations on JUGENE.

For this project 2 PMs were requested and 0.5 PMs were needed. The work carried out by the SimLab was limited to initial discussion and project planning. Further work was postponed due to lack of manpower at the University of Cologne for essential kernel development, however the work may be continued in 2012.

SLCS1102: CLaMS chemistry solver upgrade

The Chemical Lagrangian Model of the Stratosphere (CLaMS) has been developed in Jülich over the last decade. Its Lagrangian nature allows atmospheric transport processes and mixing of air masses to be studied with better accuracy than similar models based on an Eulerian concept. The Lagrangian concept of CLaMS comprises calculations for so-called air parcels that are independent of each other for most of the time. Therefore the simulations can be performed well on a parallel computer architecture like JUROPA. Even though CLaMS is already ported to this platform, significant performance improvements were to be expected within this project.

In the chemistry module *chem* of CLaMS stiff ordinary differential equations (ODEs) have to be solved. The chemistry module currently offers two possible choices of numerical solvers. The family approximation IMPACT is fast, but has the disadvantage of not converging for some air parcels. The solver SVODE evaluates the stiff ODEs within a given accuracy, but it needs about a factor of 600 more CPU-time. The goal of this project is to replace the solver SVODE by a state-of-the-art solver. With this, the divergences occurring rarely when using the solver IMPACT could be overcome. Thus a more stable simulation and more reliable simulation results are expected.

The proposed work includes a feasibility study, which of the two solvers, the Rosenbrock Method 4th order RODAS or the so-called multi-rate method for stiff ODEs that has been developed at the

university of Wuppertal in the group of Prof. M. Günther, would be better suited or easier to implement into the module. The chosen solver should then be implemented into the module. For a test simulation the results and CPU-time of all methods should be compared. Furthermore, a method for improving the load balance and memory usage should also be implemented. If the memory reduction is significant, the module could also be run on JUGENE.

For this project 2 PMs were requested and 0.5 PMs needed. As an initial step, a detailed literature study was carried out. A performance comparison of SVODE and RODAS can be found in Sandu et al., "Benchmarking Stiff ODE Solvers For Atmospheric Chemistry Problems II: Rosenbrock Solvers", Atmospheric Environment, Vol. 31, No. 20, 3499 – 3472, 1997. This paper shows that for three digit accuracy RODAS is about 1.5 times faster than SVODE, for 4 digit accuracy both are equally fast. At this point it was decided that no further efforts should be made to implement other stiff ODE solvers in CLaMS. SVODE is still compatible with state of the art schemes. Current explicit solvers like RODAS or SVODE will not be able to provide a performance comparable to the IMPACT scheme.

SLCS1103: Water vapour profiles from SCIAMACHY limb measurements

The objective of the project is to provide a new data series of water vapour in the upper troposphere and lower stratosphere region. Water vapour is one of the most important trace gases in the atmosphere: It has been shown in several recent model studies that the water vapour content in the stratosphere has an influence on the climate at the earth surface.

The water vapour profiles in the upper troposphere and lower stratosphere are retrieved from SCIAMACHY (Scanning Imaging Absorption Spectrometer for Atmospheric CHartographY) limb measurements. SCIAMACHY is a spectrometer on board the ESA Environmental Satellite (Envisat), in operation since autumn 2002. The measurements are expected to be on-going for several years. Usually more than 1000 new limb profiles are measured each day. Already more than 2 million profiles have been obtained to date: eight years of measurement data are to be evaluated within the DFG-project SHARP (Stratospheric Change and its Role for Climate Prediction, see http://www.fu-berlin.de/sharp), which aims to improve the understanding of global climate change and the accuracy of climate change predictions.

The applicants requested support with an OpenMP parallelization for FORTRAN as well as single core tuning of their code. The Simlab provided support for basic parallelization (OpenMP) and single core tuning of SCIATRAN for JUROPA. In addition, new batch scripts for distribution of sequential retrieval jobs via MPI on JUROPA were developed. For this project 0.5 PMs were requested and needed. A successful VSR/NIC proposal followed in May 2011.

SLCS1104: Optimization of 2-D PREMIER retrievals

The Juelich Rapid Spectral Simulation Code (JURASSIC) is a fast radiative transfer model for the Infrared spectral region. It is used to simulate radiance observations made by satellite- or air-borne remote sensing instruments. The code is also used to retrieve atmospheric data (pressure, temperature, or trace gas concentrations) from remote sensing observations. At FZJ/IEK-7 and JSC the JURASSIC model is currently used to carry out feasibility studies for a new satellite mission proposed to the European Space Agency. Due to new instrumental techniques, the amount of measurement data to be provided by the PREMIER mission will exceed the data rates of current experiments by a factor 100. Feasibility studies indicate that the single core performance, as well as the parallelization scheme used by the JURASSIC model, both need to be improved to cope with the large amount of measurement data provided by future satellite missions like PREMIER.

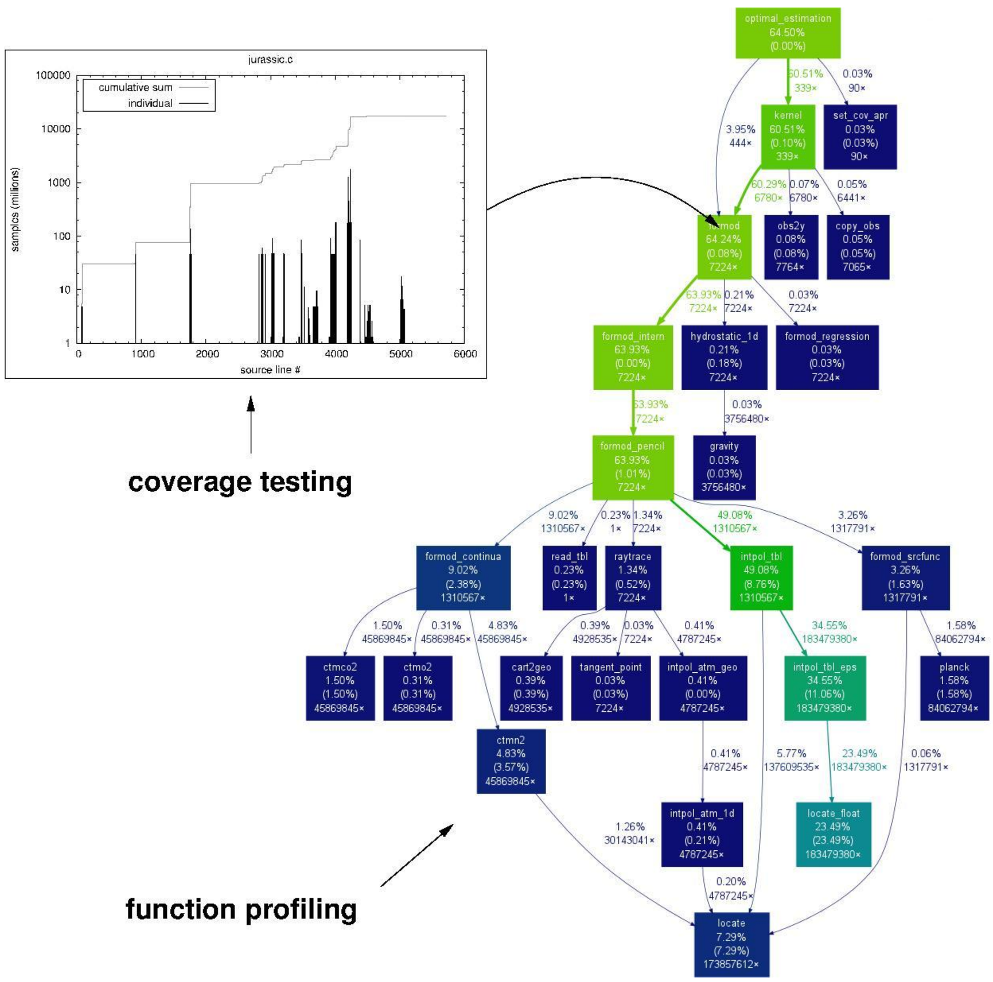
Within the project the JURASSIC model was ported and optimized for usage on the supercomputer JUROPA. Single core performance and parallelization strategies applied in JURASSIC were measured

and tuned based on the existing, well-tested application of PREMIER 2-D retrievals. Figure 1 illustrates outputs of function profiling and coverage testing as an example.

For this project 2 PMs were requested and 2 PMs were needed. The project goals were met and no problems were encountered. Outcomes of the performance analyses were highly valued input to European Space Agency (ESA) project reports. As a result of this work, the SimLab has been offered funding to participate in a follow-up ESA study during 2012.

Summary

The projects SLCS1103: 'Water vapor retrievals from SCIAMACHY limb measurements' and SLCS1104: 'Optimization of 2-D PREMIER retrievals' were completed successfully. An initial literature study for the project SLCS1102: 'CLaMS chemistry solver update' showed that the proposed solvers cannot provide the aspired performance improvements. The project SLCS1101 was postponed on request of the PI due to missing manpower.



[redacted]

3.3. Molecular Systems

The following three projects were carried out by the Simulation Laboratory Molecular Systems:

SLMS1101	Prof. Maxim Fedorov	MPI Leipzig & University of Strathclyde	Simulations of large ionic molecular systems
SLMS1102	Dr. Pieter J. in 't Veld	BASF Ludwigshafen	Parallel Monte Carlo initial structure generation for EMC
SLMS1103	Prof. Dr. Florian Müller-Plathe	TU Darmstadt	Extending the code IBIsCO to model surface interactions of polymers

SLMS1101: Simulations of large ionic moleulcar systems

The group from MPI Leipzig has a strong background in simulations of complex molecular systems and has particular interest in simulations of ionic liquids. These systems usually have very large viscosity, which means that they require very long simulation times to reach equilibration. In addition, these systems are of interest in the presence of surfaces and within capacitors in order to study the capacitance of the latter. The project was split into two main activities, i.e. A) development of a fast method for the evaluation of mutual interactions between charged particles in the presence of walls with a given surface potential and B) development of a fast electrostatics method for Monte Carlo simulations.

A) Recently it has been of interest to study molecular systems in the presence of external electric fields and between walls of a fixed potential. This problem may be tackled with introducing pseudo-charges, guaranteeing the given potential on the surface. However, it is a time consuming iterative process which extends the already long simulation runs. In addition, introducing rough surfaces, the problem tends to get even more complex. After an initial consultation with the PI it was proposed to treat the problem with the multigrid method, where the constant surface potential can be introduced via a Dirichlet boundary condition. At present, roughness of the surface is introduced as a superposition of harmonic functions, guaranteeing smoothness on the surface without introducing singular behaviour in the field solution. The multigrid solver splits the computation into regularized near- and far-field parts, where near-field contributions are calculated as direct particle pair-interactions. Up to now the roughness of surfaces was modelled and first results for test systems of charge distributions were obtained on the basis of a relaxation scheme. Full implementation of the multigrid solver as well as its parallelization has still to be done and will form the basis for a follow-up project in 2012.

B) The same group from Leipzig is also active in performing Monte Carlo simulations of complex systems. The problem here is that if the electrostatic potential energy landscape cannot be analyzed in terms of single particle contributions, then local changes of particle coordinates require the full reconstruction of the far field, in order to decide acceptance or rejection according to the Metropolis algorithm. A method which does this job is the plain Ewald summation method, as it can be written as single particle contributions in both the near- and far-field. However, Ewald summation has a complexity of $O(N^{3/2})$; local updates require $O(N^{1/2})$ operations. Therefore, a method with $O(N)$ complexity is desirable, which reduces local updates to $O(1)$ complexity. It was suggested by the Leipzig group to introduce an approximation of the $1/r$ in terms of Gaussian sums, which in principle allows fast Gauss transforms to be applied. The SimLab Molecular Systems was asked to contribute to the development of such a formalism and parallel implementation of the method. Different approaches were followed which use expansions of Hermite polynomials, Taylor expansions or Fourier techniques, in order to factorize the Gaussians into independent particle contributions.

Currently the overhead of the method is still too large to be more efficient than the Ewald summation so it is not yet clear whether the new technique will bring a real speed advantage. This question may be tackled in a follow-up project.

SLMS1102: Parallel Monte Carlo Initial Structure Generation for the EMC code

The Extensible Monte-Carlo code EMC has been developed by the PI and is used for initial structure generation within the framework of an EU project "Multi-Scale Modelling of Nano-Structured Polymeric Materials". The main purpose of EMC is to overcome the expensive initial structure generation and equilibration problem common to all molecular dynamics (MD) simulations. Conceptually, EMC is a very powerful approach in the context of mesoscale (coarse-grain) and atomistic MD simulations in terms of an initial structure generator as well as in terms of a stand-alone MC code.

The main features of EMC are:

- simple extendibility with respect to the supported force fields
- multiple chemical components
- support for a variety of multiple spatial distributions of the components
- support for coarse-grain force-fields
- standard displacement moves as well as re-bridging and end-bridging moves (polymers)
- flexible scripting language for input description

The code is written in C, partially in the spirit of object oriented encapsulation of data and data access mechanisms, with extensive use of pre-processing and dynamic memory allocation. To ensure extensibility, the code is highly modularized and all quantities are stored in well-structured linked lists so that the code can support arbitrary particle sizes and complicated spatial structures. Since the data are primarily held in terms of (recursive) structs of pointers, conventional debugging is challenging. However, some explicit diagnostic support is available. Although the data structures themselves can ultimately be located in the source code, and the code itself is well-structured, the organisation of data actually resident in memory and its access not transparent – factors which can have a large impact on performance.

#flop per	Number of threads			
interaction	1	2	4	8
20	12.9	8.8	6.2	5.9
50	23.9	14.0	8.9	6.8
100	32.8	18.9	11.2	8.6
200	51.2	28.4	15.9	10.7
400	87.1	47.1	25.3	16.5

Table 1: Simplified model for 700 interactions per move on JUROPA; timings in µs with an anticipated number of floating point operations (add-multiply) equivalent per interaction.

Initially the force and energy update of a single MC move were parallelized via pthreads (by Intveld), requiring a minor code modification, only. This turns out to be not too successful on today's architecture with an execution time of the order 20 µs per MC move limiting parallelization to at most a single node. Larger tasks are required. Parallel computation of multiple trial moves (20-30) in advance does not substantially improve the situation with these execution times because of insufficient task granularity, but again would be easy to incorporate into the code. In absence of long-range interactions, spatial decomposition of the total simulation volume would permit multiple, parallel and independent MC moves (e.g. checkerboard partitioning) with task sizes, that can be

chosen almost arbitrary large and thus furnish a basis for multi-node usage. In addition, parallel tempering would also give more opportunities for parallelism.
Incorporation of such approaches, however, would have to harmonize well with the data flow as well as data and program structure. Therefore, rather than pursuing this project further, it was agreed that the PI would first supply the necessary documentation on the data structures together with annotated input examples.

SLMS1103: Extending the code IBIsCO to model surface interactions of polymers

Tuning the code IBIsCO

The IBIsCO model uses coarse-grain molecular dynamics to simulate molecular liquids and polymers, discriminating between interphase and bulk properties and the simulation of polymerization processes (R-IBIsCO). Coarse-grained potentials are derived from radial distribution functions using the iterative Boltzmann inversion method.

We removed the reordering scheme in the subroutines *writetrj* and *average,* two routines that were very time consuming. The subroutines *takepos* and *takeen* are now called only once for the case that the subroutines *writetrj* and *average* are called in the main program in the same MD step. The communication scheme of the data between the processors was changed by assembling the different components of the positions and velocities, as well as the atom and cell indices in the communication buffers DATA*n*_EXP or DATA*n*_IMP (where *n*=3, 5 or 8). Consequently, the subroutines *communicate* and *boundary* corresponding to the three simulation schemes (Leapfrog, DPD and LA), as well as the subroutine *trans* were changed. Some redundant calls of MPI_Barrier were also removed. The subroutine *boundaryv* is not called in the code and it was excluded from compilation.

The performance enhancement of the modified code (input files reading and data preparation are not considered) compared to the original version supplied by the TU Darmstadt group for the two sets of simulations are presented in Figure 2, which shows a 4X improvement in the larger run on 512 cores. For both simulation types the list-cell is updated at each 10 MD time steps.

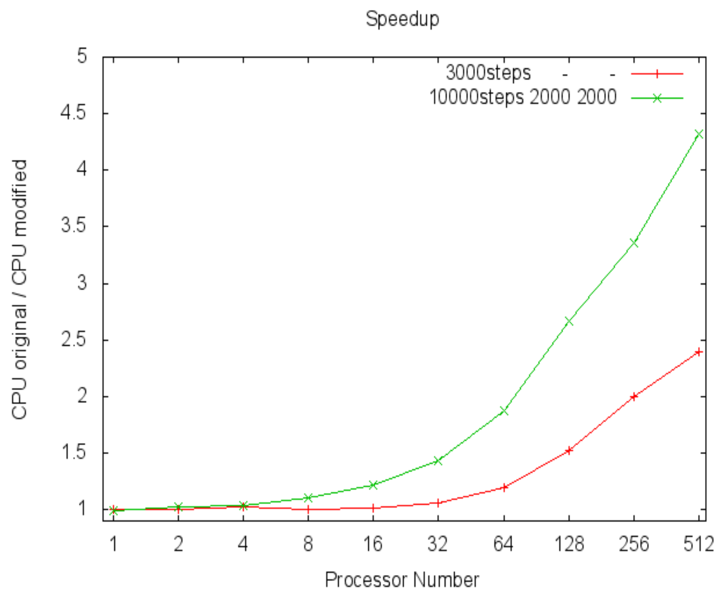
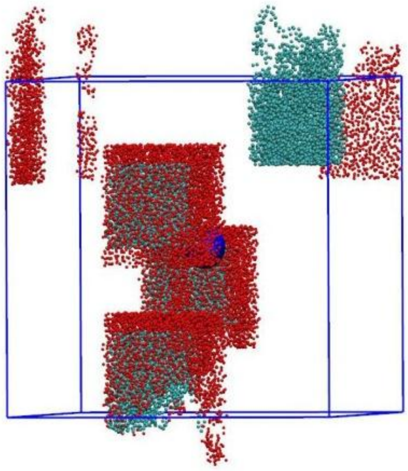


Figure 2: CPU time speed-up per MD step, for 3000 MD steps without trajectory writing and averaging (red curve); for 10000 MD steps with trajectory writing and averaging at each 2000 step (green curve).

The adaption of the code IBIsCO to surface calculations

To adapt the code to the surface calculations we chose to keep the parallel communication scheme and to modify the particle administration within the processor domains and in the link-cell list. In the current implementation, the surface slab is created only along the z direction, but in principle this scheme can be generalized to all other directions. The z direction was chosen because it is used in some other slab-type calculations that are implemented in the original version of the IBIsCO code.



[redacted]

The particles are assigned or updated to the domains in the subroutines *division* and *trans*. To reduce the unbalanced distribution of the particles to the domains, the code can be adapted to work with a non-uniform size of the domains, but this scheme will increase the computing time because of the additional effort required to localize the particles to the domains. Because the average numbers of the particles assigned to the processor domains are very similar after the slab z-shift, we did not adapt the code to the non-uniform domains. For example, in a 4x4x4 processor scheme, each processor domain contains on average 2500 particles with a standard deviation of 227 particles – Figure 3.

3.4. Plasma Physics

The following five support call projects were carried out by the Simulation Laboratory Plasma Physics:

SLPP1101	Dr. Tony Arber,	University of Warwick	Performance analysis of the PIC code EPOCH
SLPP1102	Dr. Michael Bussmann	Helmholtz Zentrum Dresden-Rossendorf	Scaling of the PIC code PICLS on JUGENE
SLPP1103	Dr. Jürgen Dreher	University of Bochum	Scaling of adaptive Multigrid solver racoon
SLPP1104	Dr. Anna Tauschwitz	University of Frankfurt	Hybrid parallelization of the transport code RALEF
SLPP1105	Dr. Sven Wiesen	FZJ/IEK-4	Optimization of the code system EIRENE-B2

SLPP1101: Performance analysis of the Particle-in-Cell code EPOCH

EPOCH is an open source Particle-in-cell (PIC) code for high energy density physics, in particular for interactions of high-intensity laser radiation with plasmas. The code is part of the UK Collaborative Computational Physics (CCP) software stack and serves a community of over 20 active users, including a number of groups in Germany. EPOCH employs MPI-parallelised explicit 2nd order relativistic electromagnetic (EM) field and particle solvers, including a dynamic MPI load-balancing option. MPI-IO based output allows restarts from checkpoint data on arbitrary number of processors.

Control of setup and runs of EPOCH is done through a customisable input deck. The main objectives of this support project were:

- To port the code on to JUGENE and JUROPA machines.
- Performance analysis on JSC machines using various benchmark problem setups.
- Analyses of current I/O performance.
- Test large scale problem capabilities and scaling.

Roughly one PM in total was spent by the SimLab Plasma Physics members, from which the major objectives were met, establishing a fruitful working relationship with the EPOCH development team at the University of Warwick. Among the completed tasks were:

- Scaling of the benchmark version of EPOCH:
  - Performance analysis test performed on JUROPA using a moderate scale problem set up (Figure 4).
  - Performance analysis test performed on JUGENE using high and low resolution problem set up (Figure 5). The scalability of the code up to 16k JUGENE cores is found to be satisfactory. However, we need to investigate further why it runs faster than the ideal scaling on JUGENE architecture.
  - Performance analysis test performed on the Minerva cluster (U. Warwick) using the whole nodes - Figure 6.
- Proper mapping on JUGENE torus network was tried for performance improvement but this did not alter the scaling behaviour significantly.

The current I/O performance measurement was performed on the Lustre file system using different object storage target numbers and MPI I/O tweaks (e.g. ROM I/O).

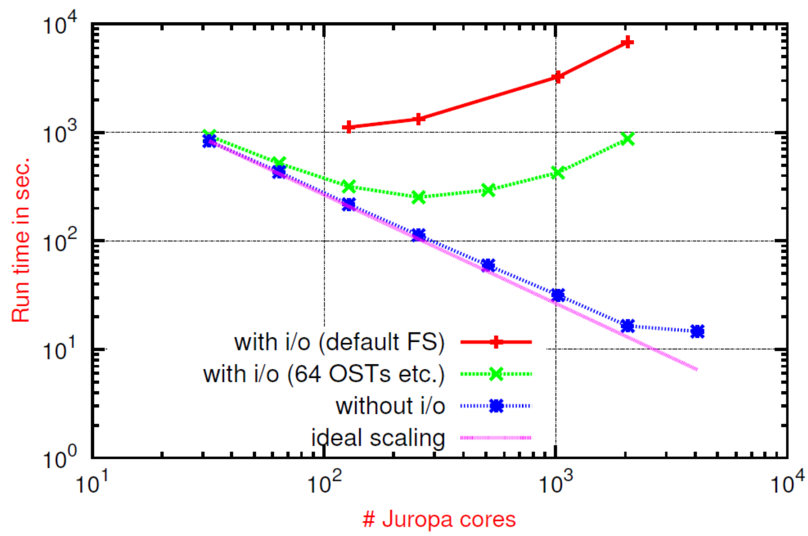


Figure 4: Strong scaling of EPOCH2D on JUROPA with 17 Million particles running for 3350 timesteps with a grid mesh: 1024 x 512. The I/O runs had totaled 80 file dumps.

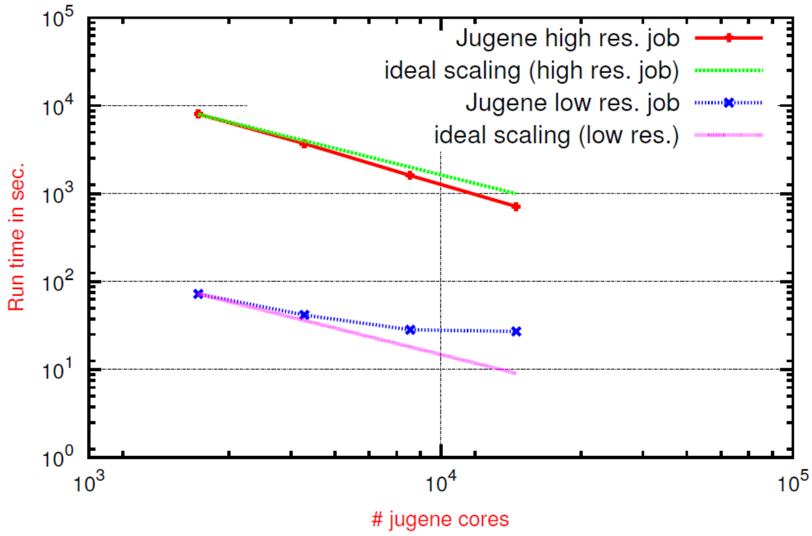


Figure 5: Strong scaling EPOCH2D on JUGENE with a high resolution (0.54 billion particles, 13400 timesteps, grid mesh of 4096_2048) and low resolution (17 Million particles, 3350 timesteps and grid mesh of 1024 x 512)

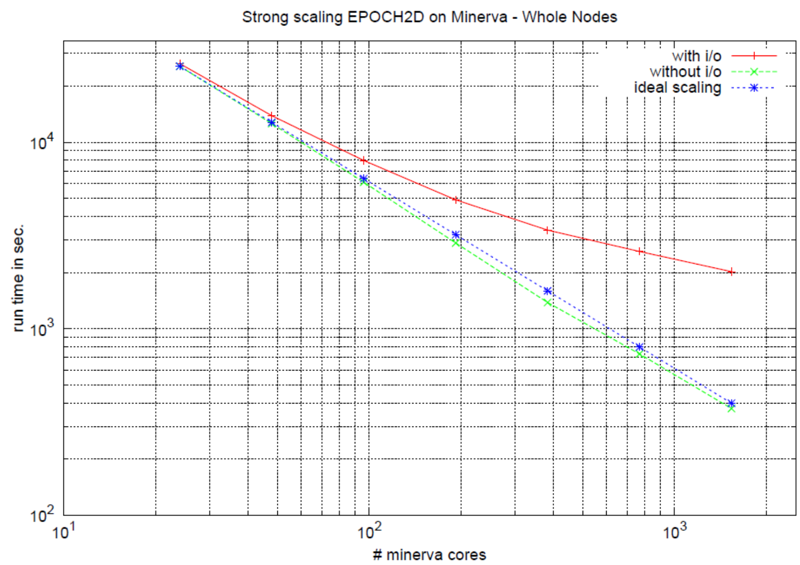


Figure 6: Performance test on Minerva (U. Warwick) using a moderate scale problem set up.

SLPP1102: Scaling of the Particle-in-Cell code PICLS on JUGENE

PICLS is a 2D Particle-in-Cell code written for distributed memory machines using purely MPI. It is aimed at realistic simulations of laser-driven ion acceleration. The proposal's version already ran on a maximum of 1024 cores on JUGENE. The main goal was to enable PICLS to run large-scale simulations on JUROPA and JUGENE using several hundred macro-particles per cell and large cell numbers. The long-term stability of the code should reach 100000 time steps. The work plan agreed upon during a video-meeting consisted of several points. The first was to analyse and improve the scaling capabilities on JUROPA and JUGENE, with a focus on the domain decomposition (possibly using the torus network) and an improved load-balancing scheme. Another objective related to this was a new parallel I/O routine to increase writing speed and thus scaling. The new I/O should be capable of 3D problems and make use of SIONlib. The Rossendorf group had also previously encountered memory problems on JUGENE of unclear origin. One task thus was to identify its source and help eliminate it.

The final step then was to use realistic parameters for a large scale simulation and show that PICLS is capable to handle such a simulation. Out of the two PMs planned for this work, only one PM was actually spent.

Good scaling could be achieved for up to 4k cores on JUGENE (probably not limited to this value but this was the largest partition tested) with a somewhat rudimentary large-scale test problem. The main computational parts of PICLS seemed to perform well: the only problematic areas were related to I/O. Disabling all I/O (data, diagnostics, etc., almost all in plain ASCII) lead to satisfactory results – as seen in Figure 7.

This shows that PICLS will scale once I/O is improved. Even though I/O has been identified as a bottleneck, no changes have been made to the code to improve the situation. The JSC internal Parallel I/O Challenge (PIOC) led to a few recommendations, but the PICLS developers (some of whom are based in the US) first wanted to define a data format and also maintain compatibility with their other codes.

Another important factor in achieving good scaling that has been pointed out to the developers is an improved domain decomposition. The currently implemented scheme is 1D and does not reflect the load on each node. In order to have reasonable initialisation times of the code, the problem size (in the dimension that is divided) had to be chosen to be a multiple of the number of nodes. Otherwise the outcome may be badly unbalanced – see Figure 8.

The memory problems on JUGENE could not be reproduced (albeit lacking an input file that showed the issues), so no real reason was identified. Some hints were given to the developers to limit continuous (de)allocation of work arrays to limit possible memory fragmentation. Memory leaks on the other hand seem very unlikely.

The performance analysis showed some subroutines that could potentially be improved to increase the speed-up. The most likely problem within those subroutines: MPI communication (non-blocking, many synchronisations). We also made some general suggestions to enhance the general code quality and that improvements are possible with respect to unused variables, implicit variables and redundant function calls.

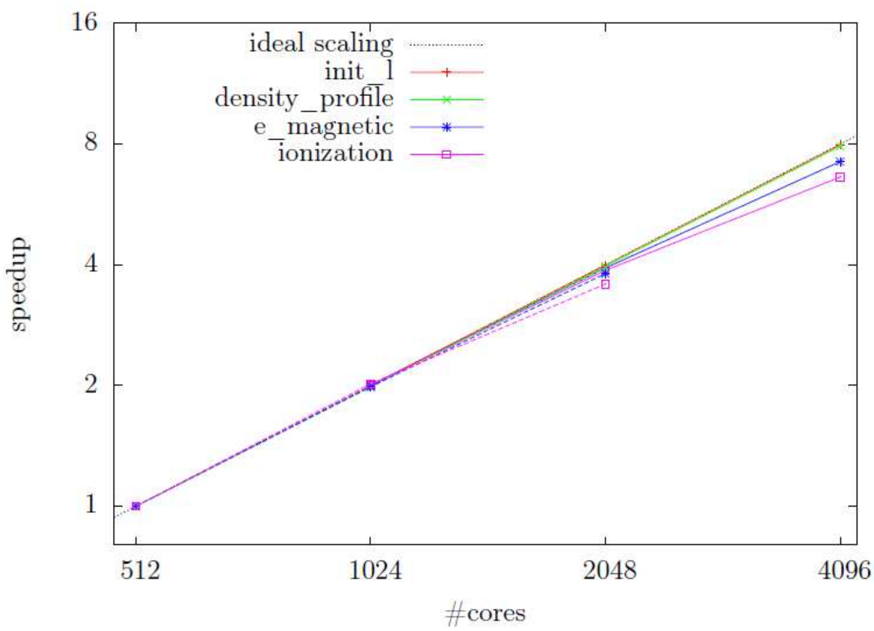


Figure 7: Scaling on JUGENE for PICLS with two problem sizes, divided into the main subroutines

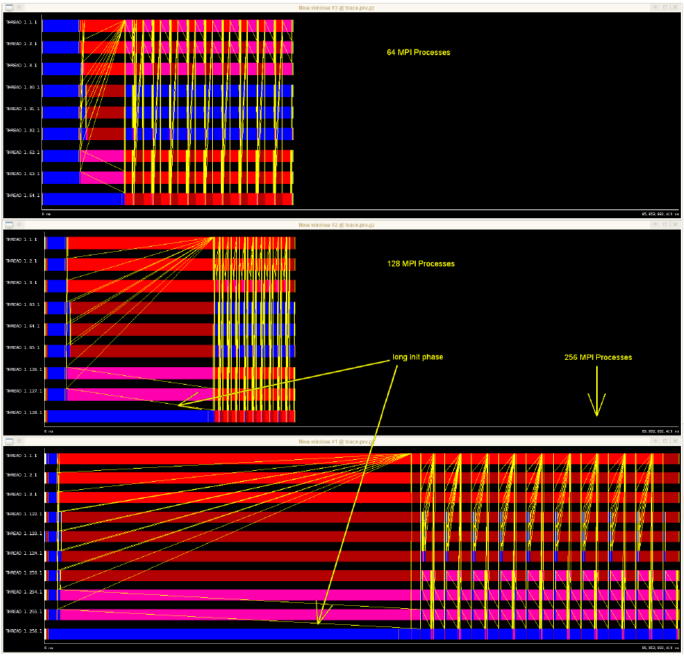


Figure 8: Examples of the bad load-balancing when the problem size does not team up with the number of nodes. The problem size is kept fixed while the number of nodes is increased. The last rank gets more and more cells to compute.

SLPP1103: Scaling of multi-grid solver racoon

The adaptive mesh refinement code racoon is developed at the Institute for Theoretical Physics I (Ruhr-University Bochum). Mainly it is used to study fundamental problems in plasma physics that require resolution of disparate scales like MHD turbulence, current sheet formation in ideal plasmas and magnetic reconnection. It has been successfully run on the JSC machines and yields formidable parallel scaling behaviour for hyperbolic systems like the compressible MHD equations on up to 256k cores. In order to simulate incompressible systems and implement implicit algorithms, a generic adaptive multi-grid (MG) solver was developed and is currently under re-design to be able to operate in staggered mesh arrangements. The original version of this elliptic solver scales to less than 1k cores.

The key goals of this support activity were to:

- analyse the MPI parallel multi-grid solver racoon
- improve scalability and execution performance
- achieve simulations on more than 2 racks (4k cores)

We planned with 1PM, however 1.5PM were invested.

An extended performance analysis, using the Scalasca tool, showed a couple of bottlenecks in the application, with large optimization potential. The first optimization aimed the volume exchange in the restriction phase of the MG solver. The application uses internal communication buffers to combine small messages into large ones and therefore achieve, in general, a better communication performance. However on the very large scales (128k cores) these buffers are much larger than the information needed to be sent and therefore there existed a large overhead by transmitting nearly empty buffers. A proper adjustment of the buffer sizes resolved this bottleneck.

The second task was to optimize the communication scheme and computational performance of the smoother. Here, again the buffer sizes had to be adjusted, as well as the way of boundary exchange on the coarsest level. The application used an overloaded strategy to solve the equations on the coarsest level, and even invoked MPI, although all computations were serial. Together with the racoon developers, we managed to achieve a simple and fast coarse grid solver for racoon. Finally, the solver was OpenMP-parallelized to reach even higher scaling.

The last optimization task aimed a proper MPI rank placement on JUGENE's torus network. Here, the ranks were placed along a Hilbert curve, according to the domain decomposition in the application.

The results of the SimLab work are summarised in Figure 9, which displays the successive reduction of the execution time (wall clock time, 128k cores) of the code after each of the above improvements had been applied.

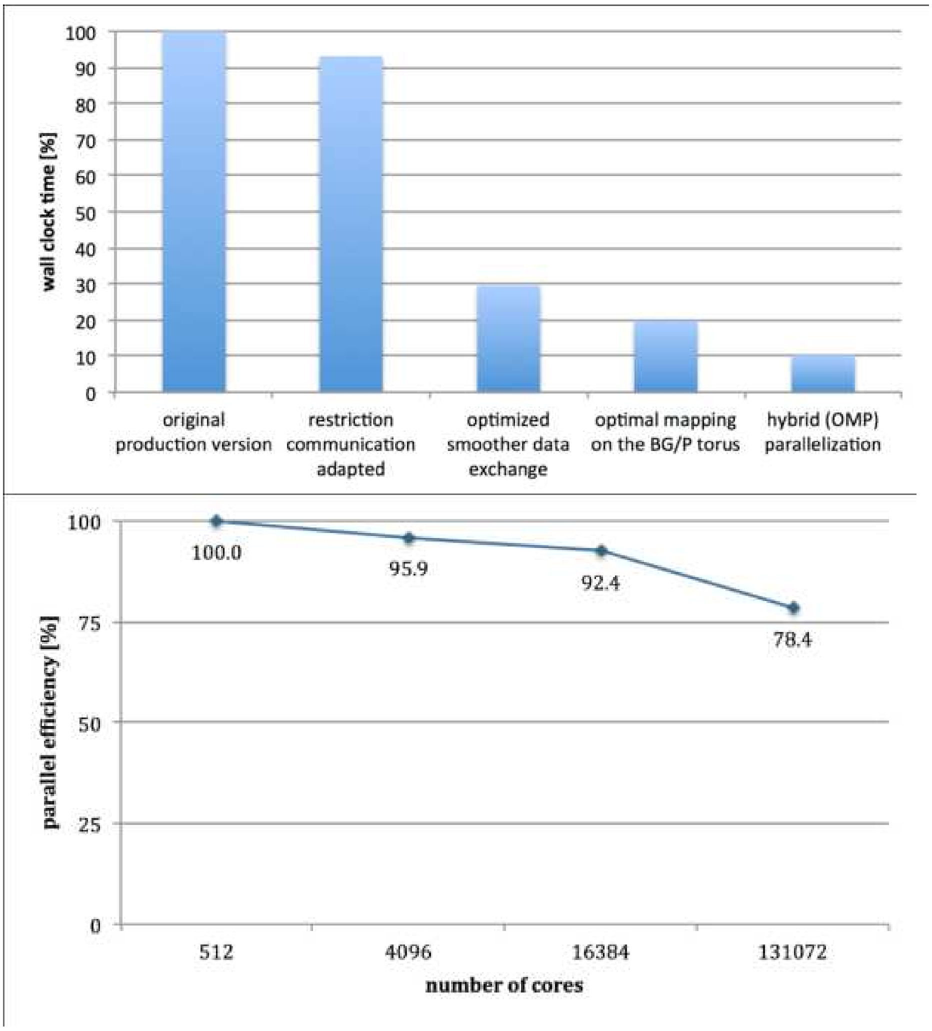


Figure 9: Top: reduction of the execution time on 128k cores BG/P due to various changes in the communication scheme. Bottom: parallel efficiency of the upgraded version.

As a lot of effort was spent on optimizing the communication, as well as the development of an hybrid parallelization strategy, the parallel efficiency has been boosted. The largest simulation scale of the original version was 4k cores, and we managed to run racoon efficiently on up to 128k cores. The following graph shows the parallel efficiency in a weak scaling benchmark.

In summary, we fulfilled all objectives in a very efficient collaborative work with racoon's developers, mainly Tobias Grafke. The only problems arose during the inclusion of OpenMP and SIMD instructions. A combination of both was not possible, as the IBM XL compiler did not produce proper SIMD instructions. As there are no fundamental reasons against the usage of both techniques at the same time, we assume that this is not possible with the IBM XL compiler. This issue was not followed further, as IBM reduced its compiler support for BG/P. We decided to use the OpenMP version, as it provides the slightly better performance.

Additional outcomes of this support activity are

- a NIC project with 4.8 rack months granted
- a publication on the benefits of an adequate MPI rank placement on torus networks (L. Arnold, Folding applications into high dimensional torus networks, International Conference on Parallel Computing, ParCo 2011 Ghent, Belgium.)

SLPP1104: MPI/OpenMP hybrid parallelization of the radiation transport code RALEF

The RALEF-2D code has been developed as part of a collaboration between GSI, the University of Frankfurt and the Institute for Theoretical and Experimental Physics, Moscow to model radiation-dominated plasmas in two dimensions. RALEF-2D is used within the Plasma Physics section of the Extreme Matter Institute (EMMI – a Helmholtz Alliance coordinated by GSI) to provide modelling support for the experimental high-energy density plasmas program at the combined laser/heavy ion beam facility at GSI. The code combines 2D hydrodynamics with frequency-resolved radiation transport equations. The hydrodynamic part was adopted from the Caveat code originally developed at LANL.

Key aspects of the SLPP support included:

- provide MPI and HPC support for the developers of RALEF
- advise on the development of a MPI/OpenMP hybrid parallelization strategy
- achieve RALEF production runs on more than 8 cores

We planned with 1PM, however we invested only 0.5PM since the developers (mainly Steffen Faik) did all of the coding and we acted as consultants, as planned.

The purely OpenMP parallelized code was successfully upgraded, by using MPI, to run on multiple nodes. The maximal number of nodes, which can be used, is limited by the problem size, as the work can be decomposed into only 24 parts. The current scalability is shown in Figure 10 below and shows a reasonable wall-clock time speedup on up to 8 nodes.

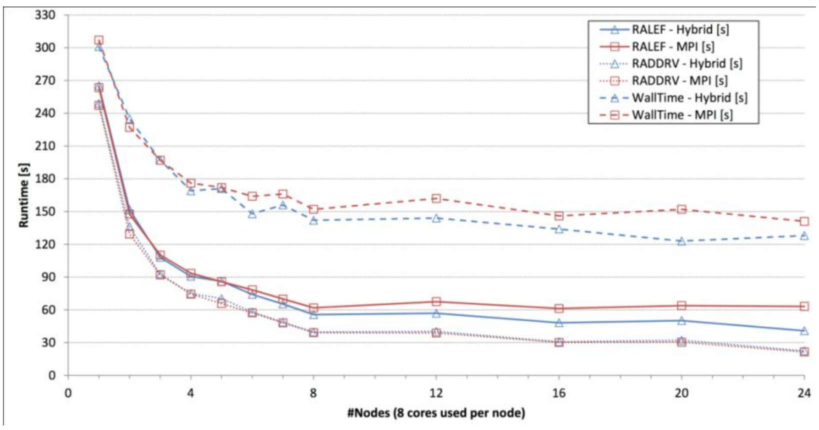


Figure 10: Performance of RALEF for pure MPI and hybrid OpenMP-MPI modes

Up to now, the developers are satisfied with the scalability and did not have a closer look into the bottlenecks which are apparent beyond 8 nodes. The next step in a future project would involve a detailed scalability and performance analysis. An approved NIC computing time application followed this support activity.

SLPP1105: Performance analysis and optimization of the code system EIRENE-B2

B2/B2.5 is a multi-fluid plasma code widely used in the interpretative and predictive modelling of fusion edge plasma scenarios (eg Textor, Asdex-Upgrade, JET, ITER). In case of a code-coupling between B2 and the Monte-Carlo neutrals code EIRENE (which forms the code suite B2-EIRENE or SOLPS), the latter is normally the bottleneck in typical fusion device simulations. But with EIRENE having been recently parallelized with MPI, the plasma fluid part B2/B2.5 now consumes much more of the computation time. Specifically, when including high-Z impurity species in the model (eg tungsten as necessary for ITER reference scenarios) the kernel fluid-solver in B2 can slow down the simulation significantly.

The goals of this project were to:

- parallelize the linear system solver in B2
- analyse the performance of the application coupling EIRENE-B2
- set up a large-scale simulation scenario

So far, we used 1PM of the 1PM planned. However, the activity has not yet been concluded.

We have provided the developers a FORTRAN module, which invokes the parallel sparse matrix solver Watson Sparse Matrix Package (WSMP). First tests, based on the matrices provided by IEK-4, showed a reduction of computation time by a factor of 3 to 4 on a full JUROPA node. No more speedup could be expected for this problem size; the solver time is already in the order of 50 µs. The developer team has already included the new module into the production version of EIRENE-B2.

First parallel performance tests showed that during production, B2 is not, as expected, the main limiting factor in scalability: it turns out that parts of the parallel EIRENE code are still heavily sequential. Although the poor scalability of the B2 will eventually be a bottleneck, parallelizing the sequential parts of EIRENE will have a major short-term impact on the time to solution. However, this parallelization task was outside the scope of this project and will be postponed to the next call.

No large-scale scenario was set up, although it was recommended in intermediate meetings. The need for high-resolution simulations is not currently an urgent issue in the collaborating group, but could become more pressing in the near future. The partner group successfully applied for VSR computing time.

4. Conclusions and Recommendations

4.1. General comments

Selection process

In many cases the effective total effort required by SimLab staff was difficult to assess accurately for various reasons:

- Undocumented or idiosyncratic code structure and/or implementation
- Suitability of the overall code layout for parallelization unknown a priori
- Lack of familiarity (within partner groups) of standard alternative algorithms/techniques
- Inaccurate or lacking performance benchmarks

To be fair, the filter applied when selecting projects in this first round was quite broad, ranging from non-HPC issues such as incorporating new physics modules (SLBIO1104), to full-blown scalability improvements on JUGENE (SLPP1103). With hindsight the less fruitful projects could have deselected (or perhaps reformulated) at an earlier stage if more stringent acceptance criteria had been applied. Basic prerequisites such as available manpower and feasibility need to be evaluated by the SimLabs, even if an external review is favourable.

Project duration and scheduling

In nearly all cases, the effort expended by JSC staff exceeded that invested by the project partners, despite the initial intention of having a more equal division of labour. Due to the various other commitments and duties of SimLab members and delays in coordinating with partners, the time required for project completion varied considerably (2-12 months). The absence of a definite completion deadline made it difficult to decide whether to pursue further work (where progress was better than expected), or to postpone SimLab activity until approval of a continuation project. At least 4 projects fall into this category.

Another limitation experienced by all labs was the lack of flexibility in dealing with 'post-deadline' applications: requests for assistance arising through on-going outreach activities (workshops, conferences and 3rd party projects), which could not be postponed until the next regular call for applications. A good example of this is the work on the CESM code now undertaken by the SimLab Climate Science with the G8 project 'Enabling Climate Simulations at Extreme Scale' (ECS), which would have fit in well with the scope of the 2010 call, but whose approval from the G8 council came in December 2010.

Dissemination

Out of the 16 projects carried out in 2011, six have already led to new NIC or VSR applications for computing time and three have resulted in longer term 3rd-party-funded activity – see Table 1 below. So far only 2 support actions look likely to culminate in a joint publication, though this may increase during the course of the partners' supercomputing projects, which began in May and November 2011 respectively. The low expected publication yield is primarily due to the modest level (2-4 person-weeks) of SimLab staff commitment actually required to meet the present crop of project goals. Also, the absence of any formal agreement or obligation on the part of the project partners

means that follow-up work has been largely decided on a case-by-case basis. Generally one can conclude that short-term (< 2 PM) actions are unlikely to result either in joint publications or 3rd-party funding – two of the main performance indicators for the HGF reports and evaluation. A direct *measurable impact* of SimLab activity through this support instrument will only be achieved by carefully selected longer-term (eg 6 PM) actions, in which the project partners also make a substantial commitment of their own. Indirect benefits will of course be felt through new or upgraded computing time proposals, and improvements to widely-used community codes.

Table 2: Project dissemination results. Filled circles represent awarded grants or published papers; open circles pending activity.

Project #	Short Title	PM spent	VRS/NIC Grant	3rd-party grant	Publication
SLBIO1101	Parallel NGS de-novo assembly	3.0	•	•	
SLBIO1102	Multi-million atom MD simulations	0.5		o	
SLBIO1103	Molecular dynamics simulations at constant pH	0.0			
SLBIO1104	Conformational preferences of p53 peptide	0.5			o
SLCS1101	Stochastic atmospheric chemistry simulations	0.5			
SLCS1102	CLaMS chemistry solver upgrade	0.5			
SLCS1103	Water vapor retrievals from SCIAMACHY measurements	0.5	•		
SLCS1104	Optimisation of 2-D PREMIER retrievals	2.0		•	•
SLMS1101	Simulations of large ionic molecular systems	1.0			
SLMS1102	Parallel Monte Carlo Initial Structure Generation	0.5			
SLMS1103	IBIsCO Optimisation	3.0	•		o
SLPP1101	Performance analysis of the PIC code EPOCH	1.0			
SLPP1102	Scaling of the PIC code PICLS on JUGENE	1.0			
SLPP1103	Scaling of adaptive Multigrid solver racoon	1.5	•		•
SLPP1104	Hybrid parallelization of the transport code RALEF	0.5	•		
SLPP1105	Optimization of the code system EIRENE-B2	2.0	•		

4.2. Recommendations

1. The SimLab high-level support activity should be divided into two separate instruments:
    a) Short-term code-enhancement/enabling measures targeted at ***new or continuing NIC/VSR projects***. The call for this action will be synchronised with the computing time call in August 2012, in a similar fashion to 'preparatory phase' support offered to PRACE projects. Priority will be given to applications for time on the newly installed BlueGene/Q. Assessment of feasibility will be included in the existing technical review by JSC staff; NIC reviewers will be informed of this new option and invited to comment on the suitability of the projects for such SimLab support. Issues which still need clarification are:
        - How much of SimLab resources can feasibly devoted to NIC/VSR projects?
        - *A priori* involvement of cross-sectional team members (Performance Analysis, Application Support, Methods and Algorithms)
        - Involvement of NIC commission + reviewers
        - Other support modes for approved/running projects: target PIs (especially during BG/Q introduction phase)

    b) Long-term, high-impact (up to 6PM SimLab effort) measures with a broader outreach (HPC users in DE, EU, and possibly beyond where there is a strategic or community benefit; emphasis on Exascale). These proposals will be reviewed by a ***SimLab Advisory Board*** before final selection by a panel of JSC staff.

2. Further labs involved should, if possible, include:
    - JARA labs Fluid & Solid Engineering and Ab Initio Methods (for FZJ and RWTH projects)
    - Nuclear & Particle Physics – potential conflicts of interest need addressing here
    - KIT SimLabs

3. Relevant Dates & Next Steps:

29 Feb 2012	Final SimLab Report 2011
29 Feb	Deadline for VSR/JARA-HPC proposals on BG/Q, JUROPA, Bull (period May-Oct)
1 April	2nd Call for advanced SimLab support (open; selection round every 3-6 months)
19-20 April	NIC executive meeting (discuss proposal for SL-preparatory phase support)
24 April	VSR/JARA-HPC commission for projects beginning May 2012
May-July	Projects running on P to be targeted for assistance porting to Q during 'migration phase' from Aug-Oct.
31 July	Call for NIC/VSR/JARA-HPC projects on BlueGene/Q and JUROPA: support template
Aug-Oct	Test accounts on Q for NIC-users Nov-Oct

Appendix A: Proposal template

Proposal for the use of Simulation Laboratory resources

Project Title								
Name of code								
Name of project @ JSC								
Simulation Lab (select one)	*Biology*		*Climate*		*Mol. Systems*		*Plasma*	
Type of work requested	*Module/algorithm redesign Application profiling* *Basic parallelisation* *Single core tuning* *Parallel scaling*				*Other (please specify)*			
Support level (person-months, max=2)								

Project coordinator:

Name	
Institution	
Street Address	
City	
Country	
Email	
Phone	

Collaborating team member(s):

Name	
Email	

Description of application/code in its present form

	current	goal
Algorithm type		
Maximum number of cores		
Memory requirements		
Storage requirements		
Pure MPI or mixed communication (OpenMP+MPI)		
Own code / 3rd party code		
Code publicly available (yes/no)?		
Library requirements		
Current architecture(s) and site name(s) used		
Target architecture(s)		

Summary:

Give a short description of the scientific background and motivation for using the proposed algorithm/code.

Scientific Goals:

Describe the expected outcome of the proposed work package. Indicate the benefits this support activity will have on the software and modelling capabilities within existing or planned HPC projects at JSC (and, if relevant, elsewhere).

Work Program (max. 1 page)

Describe the nature of the optimisation work stated on the cover sheet in more detail.

Confidentiality:

If the application is successful the above stated information and results from the projects will be used for further dissemination by JSC. However, if the application or the project's results must be treated confidentially, please declare your reasons here.

Jül-4351
Mai 2012
ISSN 0944-2952

JÜLICH
FORSCHUNGSZENTRUM