Advanced Models, Applications, and Software Systems for High Performance Computing – Application in Microelectronics

E. Langer and S. Selberherr

Institute for Microelectronics, Technische Universität Wien, 1040 Wien, Austria

Abstract This contribution deals with the Austrian research project AURORA and the application of high performance computing (HPC) in the field of microelectronics. In the first part the 'Spezialforschungsbereich' AURORA – Advanced Models, Applications, and Software Systems for High Performance Computing – is presented which is funded by the Austrian 'Fonds zur Förderung der wissenschaftlichen Forschung'. Seven research groups belonging to different institutes of the 'Universität Wien' and the 'Technische Universität Wien' are participating in AURORA, thus covering the fields computer science, statistics and operations research, numerical mathematics, electrochemistry, and microelectronics. The second part deals with the activities concerning the application of HPC to the simulation of the behaviour of microelectronic devices and their technological process steps in order to intensify the research capabilities of the simulation tools.

1 The Interdisciplinary Research Project AURORA

'Advanced Models, Applications, and Software Systems for High Performance Computing' is a national research project funded by the Austrian 'Fonds zur Förderung wissenschaftlicher Forschung'. The project has been started in April 1997 with the first period of three years; the maximum duration of such a *Special Research Field* ('Spezialforschungsbereich') is ten years on condition that the intended evaluations yield a positive result. The funds for all participating institutions amount to approximately 8.5 million ATS per year.

1.1 Scientific Concept and Long-Term Goals

High Performance Computing Systems (HPCs) represent the leading edge of computing technology. Such systems are dominated by architectural concurrency and take many different forms, including distributed-memory multiprocessors, symmetric shared-memory machines (SMPs), workstation clusters, and networks of SMPs with hybrid parallelism. Furthermore, the improvement of global communication links has led to the possibility of computing in geographically distant heterogeneous distributed networks. HPCs have been applied to many problems in science, engineering, and business over the past decade. In particular, their use for the simulation and optimization of complex processes occurring in nature, industry, or scientific experiments, has been very successful.

However, current HPCs suffer from a lack of high-level programming support. This has resulted in a still relatively small user base largely characterized by experts willing to deal with the idiosyncrasies of machines in order to exploit their capabilities to the fullest extent. The research agenda of AURORA focuses on high-level software for HPCs, with the related research issues covering the range from models, applications, and algorithms to languages, compilers, and programming environments. The activities are part of a worldwide effort towards developing the models, theories, and practical foundations to make HPCs easily and efficiently accessible to a broad range of users. They consolidate, extend and strengthen work in those areas where the participants have already made important contributions. The major goals of AURORA include

- pushing the state-of-the-art in high-level paradigms, languages, and programming environments for HPCs,
- the study and development of new models, applications, and algorithms for high performance computing,
- the selection of a set of advanced benchmark applications from the domains of both scientific computation and information management,
- the parallel implementation of these benchmarks across a range of HPCs, with a view towards achieving the highest possible performance,
- the design and implementation of techniques for managing large amounts of data in parallel file systems, and for accessing heterogeneous and distributed data,
- integration of research with educational activities, and
- active contributions to international standardization efforts.

AURORA has a distinctive interdisciplinary character, its unique feature being the synergy achieved by the coordinated cooperation of its subprojects towards the common goal of pushing the state-of-the-art in the field of software for HPCs. The institutions taking part in AURORA are involved in a large number of projects with European, American, and Japanese partners from universities, research establishments, and industrial companies. They participate in the European Commission's Research Programme; cooperation with leading hardware manufacturers supplies the project with first-hand knowledge about future architectural trends. These connections will help to broaden the scale of the overall research, understand the requirements of industry, and react quickly to new trends in architectures and networks.

1.2 Subprojects and Participating Institutions

AURORA consists of eight subprojects; five of them belong to three institutes of the 'Universität Wien', and three are carried by institutes of the

'Technische Universität Wien'. The following list shows all subprojects with the name and affiliation of the principle investigator:

- *Coordination Project*
  H. Zima, Institute for Software Technology and Parallel Systems, Universität Wien
- *Languages and Compilers for Scientific Computation*
  H. Zima, Institute for Software Technology and Parallel Systems, Universität Wien
- *Tools*
  T. Fahringer, Institute for Software Technology and Parallel Systems, Universität Wien
- *Services and Systems*
  B. Chapman, European Centre of Excellence for Parallel Computing at Vienna (VCPC), Universität Wien
- *Numerical Algorithms and Software for High Performance Computing*
  C. Überhuber, Institute for Applied and Numerical Mathematics, Technische Universität Wien
- *Parallel Algorithms for Dynamic Stochastic Optimization in Financial Planning*
  G. Pflug, Institute of Statistics, Operations Research, and Computer Science, Universität Wien
- *Parallelization of Program Package WIEN95 Used for Quantum Mechanical Calculations of Solids*
  K. Schwarz, Institute of Technical Electrochemistry, Technische Universität
  Wien
- *Parallelization of Program Packages for the Simulation of Semiconductor Processes and Devices*
  E. Langer, Institute for Microelectronics, Technische Universität Wien

The major rationale behind the initiative of the partners to establish a *Special Research Field* in the area of HPC was the prospect of the synergy arising from the interdisciplinary cooperation between language, compiler, and tool designers as well as developers of numerical algorithms on the one hand, and designers of complex, state-of-the-art applications on the other hand. Fig. 1 schematically shows the synergetic cooperation between the different subprojects.

1.3 Languages and Compilers for Scientific Computation

Research for high-level programming support, conducted in the past decade at a variety of scientific institutions, resulted in the development of a number of research compilers, based upon the data parallel *Single-Program-Multiple-Data* (SPMD) paradigm. These systems perform a source-to-source translation from Fortran, annotated by a user-supplied data distribution specification, into explicitly parallel message passing code. This transformation,

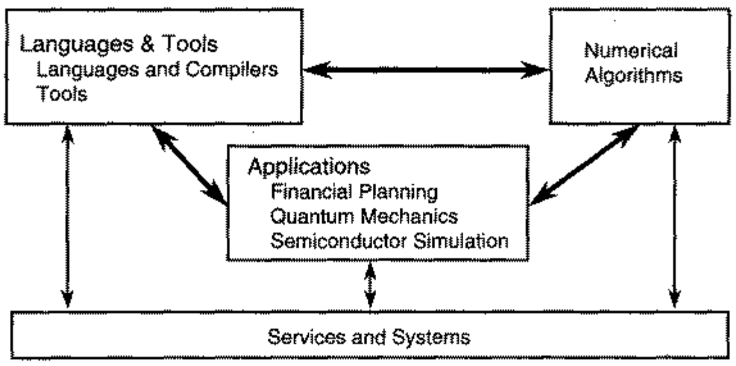

Fig. 1. A coarse grain view of synergy in AURORA

which is guided by the data distribution and by analysis information collected by the compiler, relieves the programmer from having to deal explicitly with message passing. The success of these projects paved the way for the development of *High Performance Fortran* (HPF), a de-facto standard developed by a consortium with participants from industry, academia, and research laboratories. HPF-1, the current version of HPF, extends Fortran 90 by high-level constructs for regular data distribution and alignment; it is based on concepts in languages such as Vienna Fortran [1], Fortran D, and Connection Machine Fortran. Typically, these languages allow the explicit specification of processor arrays to define the set of abstract processors used to execute a program. Distributions map data arrays to processor sets; the establishment of an alignment relation between communication if such elements are jointly used in a computation.

However, science as well as industry is interested in modeling the real world which usually does not consist of perfectly regular structures. Many advanced scientific applications including those in the application projects of AURORA are at the leading edge of simulation technology. Such applications, characterized by features like multiblock codes, unstructured meshes, adaptive grids, or sparse matrix computations, cannot be expressed in HPF-1 without incurring significant overheads with respect to memory or execution time.

Therefore, one major task of this project is the development of HPF+, an essential extension to the HPF-1 standard, which shall contribute to a new standard HPF-2. Further activities are the implementation of the *Vienna Fortran Compilation System* (VFCS) and support for irregular applications, task parallelism, parallel I/O, and interoperability.

1.4 Tools

Although rapid advances in HPCs are bringing teraflops performance within grasp, the software infrastructure for massive parallelism has not kept pace. Substantial advances in the field of parallel languages enable the programmer to write parallel programs at a machine-independent level. However, as parallelization of programs is far from being automated, there is a clear need for useful, efficient and accurate tools to support this process.

Current tools do not respond to the specific needs of users. Performance and/or debugging information are rarely supported at the HPF level but at a level much closer to the hardware. Furthermore, many languages require the programmer to explicitly specify a data and work distribution and insert calls to optimized libraries. As a consequence users must become quasi-computer scientists to understand and control these tools.

This project has been established in response to these needs. A variety of tools will be developed to support performance analysis [2] and debugging at the HPF+ level as well as translation from sequential to HPF+ programs. All tools will be integrated with the *Aurora Compilation Environment* (ACE) and focus on those features that are needed for parallelization of the AURORA applications. The following tools will be extended or developed:

- A high level debugger will be developed which enables HPF+ programmers to observe the behaviour of their programs at the level at which the programs have been developed. A followed approach is referred to as *sequential view of parallel execution*; the real parallel code is executed, but a corresponding source code level interface is presented to the programmer.
- SCALA, a performance tool for the behavioural analysis of parallelized programs will be developed. It comprises performance prediction techniques, and post-execution and scalability analysis to compute performance indices that reflect the behaviour of parallel programs.
- P^3T is a prototype performance estimator for parallel programs which has been implemented on the basis of the *Vienna Fortran Compilation System*. It statically estimates the outcome of communication, computation, load balance and cache behaviour based on distributed memory multiprocessor systems at compile time. In the context of this project P^3T will be substantially recoded and tuned for the Meiko CS-2 which is one of the target machines for AURORA.
- MIGRATOR, a reverse engineering tool supporting the translation of sequential Fortran into HPF+ will be developed. It will combine techniques for automatic algorithmic recognition and automatic data distribution.
- A graphical user interface will be developed based on object oriented user interface design principles that will provide a unique look and feel for the components of ACE.

1.5 Services and Systems

The work on software for high performance computing systems which is performed by all subprojects necessarily requires target architectures for tool and application development and an appropriate level of system support [3]. The software development projects in AURORA require HPC platforms as the target for their efforts and as installation platforms for their evaluations by the application developers. The individual application projects in AURORA require not only access to project platforms, but also to professional compilers and tools for much of their development work, in particular during the first phases of the project. In addition, they need a measure of training and support in order to successfully perform their work.

This project makes advanced computing systems available to the members of AURORA and provides state-of-the-art industrial compilers and tools as well expertise in their use to the application developers. It provides an installation and test environment for the research tools developed in the projects described in Sects. 1.3 and 1.4. Professional consulting staff with a background in numerical computing, application development as well as in programming languages and techniques, advice on application parallelization as well as on the commercial tools and languages available. Through this project the tool developers also have access to professional products and will therefore have the opportunity to evaluate their tools and techniques in comparison.

Experiences of the application developers in AURORA with professional language implementations and the accompanying tools, in particular with respect to their functionality, will provide both background and motivation for their input to research and development in these areas.

It is not possible to take up work in this area without appropriate training in the languages, tools, and techniques. This project provides the necessary instruction in the use of machines, the usage of software, and in parallel programming.

1.6 Numerical Algorithms and Software for High Performance Computers

Both the simulation and optimization of processes in science, engineering or economics involve numerical techniques with a huge number of computational steps which are to be performed within a given time limit. For complex large-scale computations there has always been a demand for higher performance. Hardware developers and manufacturers have indeed been able to meet this demand to a high degree. However, these achievements have brought up a phenomenon which troubles the users of high performance computers: The gap between hardware peak performance and the actual performance achieved by application programs is becoming wider and wider. It is increasingly difficult for application programmers to exploit the available hardware resources to a satisfactory degree. Even for simple algorithms like matrix-matrix multiplication, exploiting the hardware potential is anything but trivial [4].

Increased peak performance due to innovative computer architecture can be exploited only if application programs are restructured and modified. Analyzing and optimizing the performance of numerical programs, supported by using tools and techniques developed by projects in Sects. 1.3 and 1.4 is a main goal of this project. Simultaneously, the other main task is to provide expertise in mathematical modeling as well as in effective numerical and parallel programming to the application projects (Sects. 1.7–2). Feedback from and interaction with applications problems is essential for theoretical research in numerical and applied mathematics.

Portable programs can only run efficiently on different high performance (parallel) systems if they are adjusted to the computer system they run on. The chosen approach to this problem deals with algorithms that are able to adapt themselves automatically to the peculiarities of different computer architectures (*Architecture Adaptive Algorithms*). In particular, it is planned to attack the following three prototypical numerical problems:

- hierarchical blocked algorithms
- FFT algorithms
- numerical integration with dynamic load balancing

1.7 Parallel Algorithms for Dynamic Stochastic Optimization in Financial Planning

Over the last decade there has been a tremendous growth in financial modeling that utilizes inputs from finance theory, operations research/management science and mathematical as well as computational techniques. Financial planning problems typically concern the allocation of financial resources to achieve specific goals. First, any re-balancing of funds must accommodate the consumption needs of an investor and anticipated future liabilities/deposits must be taken into account. Second, transfers of funds are associated with transactions costs that have to be included in the decision making process. Third, any financial planning is governed by uncertainties of future returns, interest rates, withdrawal streams, etc. Thus, a multiperiod stochastic financial optimization approach becomes necessary [5].

Since the early work of Markowitz stochastic optimization techniques have been applied to portfolio planning problems. In such a problem an investor faces the choice of designing an efficient portfolio based on risk and expected returns measures. Portfolio selection problems have since become a standard approach in theoretical and empirical finance and meanwhile are used by many banks and firms from the financial services industries. Although the Markowitz model constitutes the classical approach to hedge unsystematic risk it suffers from several shortcomings. Investors' preferences are modeled in an ad hoc fashion, it does not properly take into account uncertainty over all stages of the planning horizon, and it sticks to the assumption of a single period framework. A more general approach has been proposed in the literature that is now referred to as financial optimization [6]. Inspired by recent advances in the theory of dynamic stochastic optimization techniques (in particular multistage stochastic dynamic programming) financial economists have only recently begun to utilize these techniques for asset allocation decisions, portfolio immunization strategies and the optimal design of financial securities. The goal of this project is now the development of a generally applicable multiperiod stochastic financial model which – due to its complexity – can only be realized within a high performance computing environment.

1.8 Parallelization of Program Package WIEN95 Used for Quantum Mechanical Calculations of Solids

Over the last fourteen years the members of the *Computational Quantum Theory Group* within the Institute of Electrochemistry have developed a sophisticated program [7], that allows to compute the electronic and magnetic properties of solids based on the *linearized augmented plane wave* (LAPW) method using density functional theory. This first principal scheme does not rely on experimental data but requires only the knowledge of the atomic numbers of the constituent atoms. LAPW is one among the most accurate methods to investigate high technology materials for which quantum mechanics is crucial for their understanding, as for example the new high temperature super-conductors, phase transitions, surface catalysis, intercalation compounds (e.g. for new Li batteries), magnetic structures, electric field gradients, etc. Such applications demand high precision and a reliable computer code that can run – even for weeks – to produce results. The LAPW program package whose latest version is called WIEN95 is written in Fortran 77 and runs on various Unix platforms from workstations to vector computers, where the handling is controlled by C-shell scripts.

The kernel of the computation consists of the following steps: the translational symmetry of a solid leads to Bloch states which are labeled by the k-vector within the unit cell of the reciprocal space (Brillouin zone). For a given crystal potential (with fixed nuclei) one must solve a set of Schrödinger type equations for each of these k-vectors to derive the wave functions $\psi_{n,k}$ for all states n, k. However, the potential itself depends on the electron density that is determined from $\psi^*\psi$ of all occupied states. Consequently such a problem can only be solved iteratively in the so-called self-consistent field scheme. The wave functions are expanded in basis sets (LAPWs) and the coefficients are determined by the variational principle according to which a generalized eigenvalue problem must be solved. The corresponding size of the matrix is related to the accuracy of the calculation and thus to the number of plane waves used in the basis, where about 50–100 plane waves are needed per atom in the unit cell. For systems containing up to 100 atoms per unit cell a matrix size of 5000 and more must be handled.

Since the solutions of material science problems require higher computational resources than presently available on a single processor machine, the main objective of this project is the parallelization of WIEN95 with the additional condition to maintain portability.

2 Application of HPC in Microelectronics

Since the technologies in all areas of electronics continue to evolve, numerical simulation of the technological steps in the fabrication of semiconductor devices [8] as well as the simulation of their electrical behaviour [9] has become eminently important. Neither an improvement of an existing fabrication process nor the development of a new integrated semiconductor device nowadays can be performed efficiently without simulation. One main direction for ongoing research is the extension of the simulation tools to three spatial dimensions because the shrinking geometries of the devices lead to physical phenomena which are not taken into account by two-dimensional simulations. Nevertheless, three-dimensional simulations actually suffer from a very high demand on computing power and can, therefore, only be applied for investigations of special effects. In order to utilize these tools for process development and improvement in conventional computing environments, an acceleration of the most time consuming program parts is necessary. Parallelization seems to be the unique way to achieve an effective decrease of the real time consumption.

Various program packages for the simulation of semiconductor processes and devices available at the Institute for Microelectronics have been developed, extended and improved by members of the institute over almost two decades. All these programs – the most prominent one is MINIMOS [10] – are worldwide used by national and international industrial partners as well as by universities. The relative high diversification of existing programs is not only given by the three main topics (process simulation, device simulation, and simulation of interconnect capacitances) but also due to the fact that different parts of these themes have to be treated by separate modules because of the underlying physics and/or geometry (different process steps such as ion implantation, diffusion, oxidation, etching, and deposition or different devices such as MOS transistors, bipolar structures, BiCMOS devices). During the last years, the *Viennese Integrated System for Technology CAD Applications* (VISTA) [11] has been created – a framework putting all developed tools together.

The simulation of semiconductor processes and devices is based, on the one hand, on the numerical time-dependent solution of a set of partial differential equations in two or three spatial dimensions, and, on the other hand, on Monte Carlo or cell based algorithms. Monte Carlo techniques actually are applied for the simulation of ion implantation and for a special carrier transport model within device simulation. During the first period of the AURORA project *Parallelization of Program Packages for the Simulation of*

Semiconductor Processes and Devices the main effort is laid on a decrease of the turn around time of the Monte Carlo simulation of ion implantation. The improvement shall be achieved by creating a parallelized version running on a heterogeneous workstation cluster. A very important goal is to maintain the portability of the program code which efficiently executes on various platforms because the parallelized version should run also at all sites of our cooperating partners.

2.1 Monte Carlo Simulation of Ion Implantation

Ion implantation is currently the most important technique for introducing dopants into semiconductors. Fig. 2 illustrates the physical process: After an acceleration with a high electric field, the ions hit the surface of the target with a certain energy E_0 and a given direction (defined by the tilt angle α) and enter the substrate material (e.g., silicon). Within the material the ions are colliding with substrate atoms thus loosing a portion of their energy. Between two collisions of the same ion under investigation another braking mechanism takes effect: the so-called electronic stopping power [8]. After coming to rest due to the energy loss the ion contributes to the whole doping concentration which basically estimates the electrical properties of the semiconductor structure.

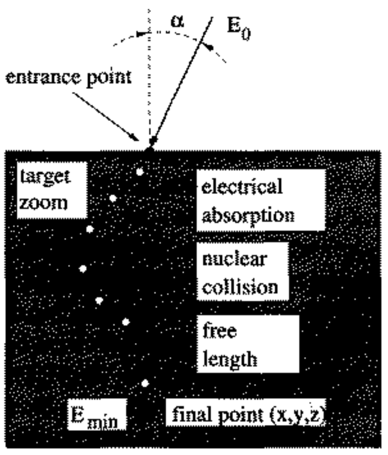

Fig. 2. Ion trajectory within the substrate

Out of the possible simulation methods, i.e. analytic description, solution of the Boltzmann transport equation, and the Monte Carlo method, the latter has been turned out to be the favourite choice for simulations in three spatial dimensions. The Monte Carlo simulation exactly follows the physical process by sequentially calculating a high number of ion paths (trajectories) and finally the position of each ion within the substrate. Depending on the number of spatial dimensions to be regarded up to some millions of trajectories must be calculated in order to obtain a sufficient accurate statistical information. Although the so-called *trajectory split method* [12] reduces the computing effort about a factor of five, a further increase of the throughput – by means of parallelization – is an absolute necessity.

It should be noted that the assumption holds that one ion does not influence directly another ion, therefore, the validity of the sequential calculation is justified. Furthermore, in addition to the electronic stopping only collisions between one ion and one substrate atom have to be taken into account (*two body problem*).

The kind of structure of the target material has an important influence on the simulation: An amorphous target can be treated easily because there do not exist preferred directions, that means, that the ions are scattered randomly after the collisions and one trajectory has no influence on the global geometrical information. The case of ion implantation into a crystalline target (in practice this is the most common case) is much more complicated because here preferred directions for ion propagation exist (*channeling* of ions). Furthermore, the ion bombardment causes damages of the crystal structure and an accumulation of such *point defects* leads to a locally amorphous behaviour of the material: the local degree of amorphization and, therefore, the global geometric information changes (in principle) from one particle simulation to another; nevertheless, spatially separated trajectories can be calculated in parallel even in the crystalline case. The chosen strategy of parallelization can be summarized as follows:

- The simulation space including the list of ions to be calculated is distributed to the available nodes.
- A master task holds information of all processes.
- Child tasks hold all local data needed for damage calculation.
- When an ion leaves the current subspace the trajectory data are transferred to the corresponding processor which continues the calculation.
- At certain time steps all processors must be synchronized in order to include the transient effect of amorphization. At these points where all nodes have calculated a certain number (a few hundred) of trajectories a load balancing can take place by redistributing the simulation space.

This strategy ensures that only few information per ion has to be sent over the network. The current state within AURORA is a message passing version of the sequential Monte Carlo code without the planned dynamic load balancing mechanism. In cooperation with the technology providing projects (Sects. 1.3 and 1.4) the use of HPF+ for parallelizing the calculation of the few hundred trajectories within each subspace between the mentioned time steps is investigated.

2.2 Photolithography Simulation

Among all technologies photolithography holds the leading position in pattern transfer in today's semiconductor industry. The reduction of the lithographic feature sizes towards or even beyond the used wavelength and the increasing nonplanarity of the devices place considerable demands onto the lithography process and, in consequence, onto the modeling because a three-dimensional simulation becomes necessary.

During the last years an overall three-dimensional photolithography simulator consisting of three different modules has been developed [13] whereby each module accounts for one of the fundamental processes of photolithography: imaging (illumination of the photomask), exposure/bleaching, and development. Because from the simulation point of view the exposure/bleaching reaction is by far the most demanding problem within photolithography simulation, only this module is focused in the following.

The exposure/bleaching module simulates the chemical reaction of the photosensitive resist. Thereby the light propagation within the optically nonlinear resist and electromagnetic (EM) scattering effects due a nonplanar topography have to be modeled. The exposure state of the photoresist is described by the concentration of the photoactive compound (PAC) $M(\boldsymbol{x},t)$ which constitutes the latent bulk image. During the image transfer into the photoresist by light absorption, the PAC is dissolved and the optical properties, e.g., the refractive index $n(\boldsymbol{x},t)$, are changed. As usually this reaction is modeled by Dill's 'ABC'-model [14]

$$\frac{\partial M(\boldsymbol{x},t)}{\partial t} = -CI(\boldsymbol{x},t)M(\boldsymbol{x},t) \tag{1}$$

$$n(\boldsymbol{x},t) = n_0 + j\frac{\lambda}{4\pi}\left(AM(\boldsymbol{x},t) + B\right) \ , \tag{2}$$

where $I(\boldsymbol{x},t)$ is the exposing light intensity, and A, B, and C are Dill's parameters. Consequently, the EM field inside the nonlinear photoresist must be calculated. Because the bleaching rate is small compared to the frequency of the EM field, the quasi-static assumption

$$M(\boldsymbol{x},t_{k+1}) = M(\boldsymbol{x},t_k)e^{-CI(\boldsymbol{x},t_k)(t_{k+1}-t_k)} \tag{3}$$

is applied, where the initial PAC distribution is homogeneous, $M(\boldsymbol{x},t_0) \equiv 1$. Furthermore, a steady-state field distribution within a time step $t_k \leqq t < t_{k+1}$ is assumed. Therefore, the EM field is time-harmonic and obeys the Maxwell equations in the form of

$$\begin{aligned} \operatorname{curl} \boldsymbol{H}_k(\boldsymbol{x}) &= -j\omega_0\varepsilon_0\varepsilon_k(\boldsymbol{x})\boldsymbol{E}_k(\boldsymbol{x}) \\ \operatorname{curl} \boldsymbol{E}_k(\boldsymbol{x}) &= j\omega_0\mu_0\boldsymbol{H}_k(\boldsymbol{x}) \, . \end{aligned} \tag{4}$$

The complex permittivity $\varepsilon_k(\boldsymbol{x})$ is related to the refractive index by Maxwell's law (5), and the exposing light intensity is given by (6) according to [15]

$$\varepsilon_k(\boldsymbol{x}) = n^2(\boldsymbol{x}, t_k) \tag{5}$$

$$I(\boldsymbol{x}, t_k) = \frac{1}{2}\sqrt{\frac{\varepsilon_0}{\mu_0}} n_0 \|\boldsymbol{E}_k(\boldsymbol{x})\|^2 . \tag{6}$$

The equation set (2) to (6) represents the simulation model for the exposure/bleaching reaction, whereby an efficient solution of the inhomogeneous but linear Maxwell equations (4) is the crucial point for the application of the model.

One of the possible solution strategies is based on the so-called *time-domain finite-difference* (TDFD) method which leads to a typical HPC application. This method uses an equally spaced ortho-tensor-product-grid and thus avoids the meshing problem. Although the required computing resources are extremely high, the full power of parallel computers can be exploited because the equations are predisposed for a parallel implementation: The TDFD mesh is divided into N equal-sized subspaces, and the N nodes calculate independently the field in the subspaces. At each time step, the field values at the dividing planes are exchanged. Fig. 3 illustrates the performance gain with increasing number of grid points G.

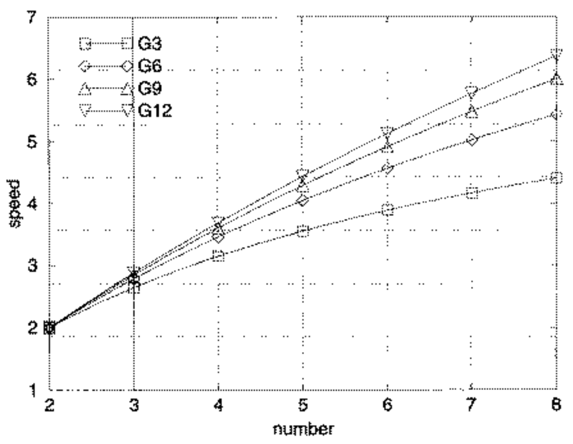

Fig. 3. Performance gain by means of parallelization

2.3 Optimization of MOSFET Doping Profiles

High performance ULSI technology requires devices with optimized characteristics. In the past various device structures have been reported for MOS-

FETs with different aims, e.g. improving short-channel effects, drive current, leakage current, gate delay, etc. Low-power applications became more and more important because of the growing portable electronics market. Supply voltage will continuously be reduced for future device generations in order to reduce the power consumption and to enable single-battery operation. Very low off-state currents are needed to meet the standby power requirements.

Optimizations performed by hand or manually controlled simulations are no longer suitable for complex performance goals. Therefore, a fast self-contained device optimization process is required [16] which, due to its required computing effort, represents a HPC application. Parameters like device structure, supply voltage, or allowed drain-source leakage current have a strong influence on the optimization results. General dependencies on these parameters can be found to give a guideline for the design of optimal doping profiles.

As an example the criteria for an optimization of an n-MOSFET doping profile shall be discussed. The goal is to achieve maximum drive current I_{on} for a supply voltage $V_{\mathrm{dd}} = 1.5\,\mathrm{V}$ while at the same time keeping the drain-source leakage current I_{off} below $1\,\mathrm{pA}$. This constraint must be combined with the goal to the global optimization target

$$\mathrm{target}(I_{\mathrm{on}}, I_{\mathrm{off}}) = \mathrm{penalty}(I_{\mathrm{off}}) - I_{\mathrm{on}}\,, \tag{7}$$

where a half-parabolic penalty function for I_{off} is used:

$$\mathrm{penalty}(I_{\mathrm{off}}) = \begin{cases} \frac{10^7}{1\,\mathrm{pA}} \cdot (I_{\mathrm{off}} - 1\,\mathrm{pA})^2 & : \quad I_{\mathrm{off}} > 1\,\mathrm{pA} \\ 0 & : \quad I_{\mathrm{off}} \leqq 1\,\mathrm{pA}\,. \end{cases} \tag{8}$$

The global optimization target (7) is minimized using a nonlinear optimizer. As illustrated by Fig. 4 one optimization step is described as follows:

- the optimizer requests an evaluation with a set of doping parameters
- the device description (geometry, doping, and simulation-grid) is produced by an analytical device generator using this parameter set
- the required device simulations are carried out by the simulator
- the global optimization target is calculated and returned to the optimizer

2.4 Parallel TCAD Simulations Using Dynamic Load Balancing

As already outlined at the beginning of this section, the use of the simulation of semiconductor process and devices at a large scale suffers from the high required computing times. Therefore, maximum parallelism has to be exploited and the available computation resources have to be used optimally in order to reduce the overall simulation time to a minimum. One way is to distribute the simulation tools themselves on several CPUs on several hosts as discussed for the Monte Carlo simulation of ion implantation in Sect. 2.1.

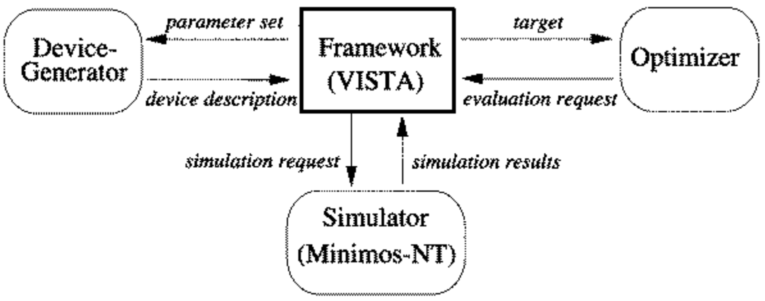

Fig. 4. Optimization process

A convenient additional effort lies in the parallelization of multiple instances of each simulation tool, as far as this is suitable [17]. For many situations like optimization or split run experiments the latter approach offers excellent performance.

Fig. 5 shows the components of the VISTA framework [11]. An optimizer is utilized to search for an optimum set of system parameters with respect to certain system characteristics. This typically requires the computation of Jacobian matrices, which needs large numbers of independent model evaluations that can be executed in parallel. The process model delivers its jobs to the *queue manager*, which maintains two queues of waiting and executing jobs, respectively. The *queue manager* is permanently looking for hosts which are able to process these jobs while the system load of these hosts is periodically monitored.

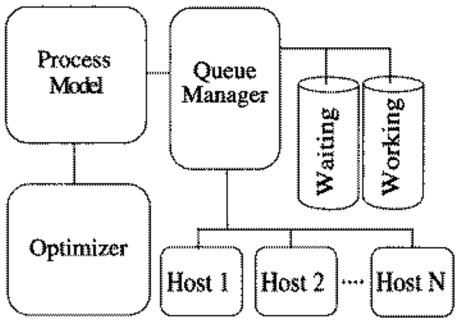

Fig. 5. Architecture of the VISTA framework

In order to find the optimum host, it is necessary to make the decision based on an estimation of the current system load. Time delays of the load reported by the Unix operating system would lead to overloading of machines, since it typically takes five seconds up to one minute until a job is reflected in the system load. Therefore, VISTA keeps track of jobs that are executed by itself on each host and calculates an estimated effective load based on this information and the load reported by the Unix operating system. For a host with N jobs running and M jobs recently finished, the effective load is estimated by

$$l_{\text{eff}} = l_{\text{os}} + \underbrace{\sum_{i=1}^{N} e^{-\frac{t-t_i^{\text{start}}}{\tau}}}_{\text{running jobs}} - \underbrace{\sum_{i=1}^{M} \left[\left(1 - e^{-\frac{t_i^{\text{stop}}-t_i^{\text{start}}}{\tau}} \right) \cdot e^{-\frac{t-t_i^{\text{stop}}}{\tau}} \right]}_{\text{finished jobs}} \tag{9}$$

as depicted in Fig. 6. In this formula τ is an estimate for the time by which the reported operating system load is delayed. The correction term for running jobs prevents hosts from being overloaded due to time delays. The contribution for finished jobs prevents hosts from being considered busy while the operating system still reports too much system load.

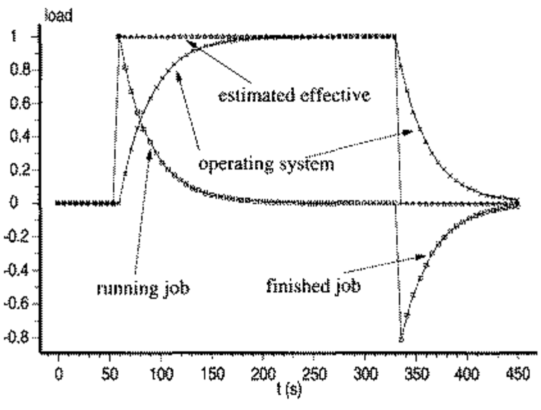

Fig. 6. Running and finished jobs for effective load estimation

Without such an estimation, both load balancing and load limit based disabling of hosts would give poor results. Either for large numbers of requested jobs a host would hopelessly be overloaded due to time delays, or if one limited the number of jobs on a host, inactive jobs, which are for some reason sleeping, would block hosts. On the other hand the operating system reports an unnecessary high system load for hosts which recently finished jobs, and therefore computation resources are unused until the system load drops down to the effective value.

3 Conclusion

Nowadays, high performance computing plays an important role in many disciplines and its importance will continuously increase. Because of the rapid progress in microelectronic technology, massive parallel computer architectures will be available for reasonable costs within the near future whereby the accessibility to HPC environments will be considerably improved. Doubtless the *Special Research Field* AURORA will push essentially the simulation of semiconductor devices and their technological process steps and, therefore, contribute to an accelerated development of high performance computing systems.

References

1. Zima, H. P., Brezany, P., Chapman, B. M.: SUPERB and Vienna Fortran. Parallel Computing **20** (1994) 1487–1517
2. Fahringer T.: Estimating and Optimizing Performance for Parallel Programs. IEEE Computer **28(11)** (1995) 47–56
3. Chapman, B. M., Pantano, M., Zima, H. P.: Supercompilers for Massively Parallel Architectures. Proc. Aizu Inter. Symposium on Parallel Algorithms/Architecture Synthesis (1995) 315–322
4. Überhuber, C. W.: Computer-Numerik I. Springer-Verlag, Berlin Heidelberg New York Tokyo (1995)
5. Pflug, G. Ch.: Optimization of Stochastic Models. Kluwer Academic Group, Boston (1996)
6. Zenios S. A.: Massively Parallel Computations for Financial Modeling Under Uncertainty. Mesirov, J. (ed.), Very Large Scale Computing in the 21st Century. SIAM (1991) 273–294
7. Schwarz, K.: Quantum Mechanical Calculations Based on Density Functional Theory. Phase Transitions **52** (1994) 109–122
8. Langer, E., Selberherr, S.: Prozeßsimulation: Stand der Technik. In: Festkörperprobleme / Advances in Solid State Physics **36**, Vieweg (1996) 203–243
9. Kosina, H., Langer, E., Selberherr, S.: Device Modeling for the 1990s. Microelectronics Journal **26** (1995) 217–233
10. Fischer, C, Habaš, P., Heinreichsberger, O., Kosina, H., Lindorfer, Ph., Pichler, P., Pötzl, H., Sala, C., Schütz, A., Selberherr, S., Stiftinger, M.,Thurner, M.: MINIMOS 6 User's Guide, Institut für Mikroelektronik, Technische Universität Wien, Austria (1994)
11. Halama, S., Fasching, F., Fischer, C., Kosina, H., Leitner, E., Pichler, C., Pimingstorfer, H., Puchner, H., Rieger, G., Schrom, G., Simlinger, T., Stiftinger, M., Stippel, H., Strasser, E., Tuppa, W., Wimmer, K., Selberherr, S.: The Viennese Integrated System for Technology CAD Applications. In: Technology CAD Systems (Fasching, F., Halama, S., Selberherr, S., eds.), Springer (1993) 197–236
12. Bohmayr, W., Burenkov, A., Lorenz, J., Ryssel, H., Selberherr, S.: Statistical Accuracy and CPU Time Characteristic of Three Trajectory Split Methods for Monte Carlo Simulation of Ion Implantation. In: Simulation of Semiconductor Devices and Processes – SISDEP **6** (1995) 492–495

13. Kirchauer, H., Selberherr, S.: Three-Dimensional Photolithography Simulation. IEEE Trans. Semiconductor Technology Modeling and Simulation **6** (1997) http://www.ieee.org/journal/tcad/accepted/kirchauer-jun97/
14. Dill, F. H.: Optical Lithography. IEEE Trans. Electron Devices **ED-22(7)** (1975) 440–444
15. Bernard, D. A.: Simulation of Focus Effects in Photolithography. IEEE Trans. Semiconductor Manufacturing **1(3)** (1988) 85–97
16. Stockinger, M., Strasser, R., Plasun, R., Wild, A., Selberherr, S.: A Qualitative Study on Optimized MOSFET Doping Profiles. Proc. SISPAD'98 – Inter. Conf. on Simulation of Semiconductor Processes and Devices (to appear)
17. Pichler, C., Plasun, R., Strasser, R., Selberherr, S.: High-Level TCAD Task Representation and Automation. IEEE Trans. Semiconductor Technology Modeling and Simulation **5** (1997)