Proceedings of the Seventeenth IASTED International Conference
APPLIED INFORMATICS
February 15-18, 1999, Innsbruck, Austria

AMIGOS A RAPID PROTOTYPING SYSTEM

M. RADI and S. SELBERHERR
Institute for Microelectronics, TU Vienna
Gusshausstrasse 27-29,A-1040 Vienna, AUSTRIA
e-mail: radi@iue.tuwien.ac.at
http://www.iue.tuwien.ac.at

Abstract

To accurately simulate physically based technology problems, a simulation tool must include a variety of prototyping capabilities and numerical methods. Increasingly complex physical formulations are required to account for effects that were not important in simulating previous generations of technology. Thus flexibility in definition of models as well as appropriate numerical solving methods are highly desirable. We applied an object-oriented approach to implement a dimension independent solver which uses an analytical input interface highly optimized for numerical calculations of partial differential equations on complex simulation domains. It is equipped with a numerical solver that supports direct and iterative solution methods, as well as an adaptive hierarchical grid adaptation algorithm which can be controlled by the analytical input language.

General Overview

AMIGOS is a problem independent simulation system which can handle a wide range of nonlinear partial differential equation systems in time and space in either one, two or three dimension(s). It is designed to automatically generate optimized numerical models from a simple mathematical input language, so that no significant speed loss in comparison to 'hand coded' standard simulation tools occurs.

In difference to similar algorithms based on the 'operator on demand' concept [1], AMIGOS is completely independent of the of discretization since the model developer can formulate any discretization of choice. There are no restrictions whether using scalar, field or tensor quantities within a model, and, if desired, any derived field quantity can be calculated, too. Furthermore, the user can influence the numerical behavior of the differential equation system by complete control of the residual vector and its derivative (e.g. punishing terms, damping terms, etc.). Even interpolation and grid-adaptation formulations can be used within a developed model and can thus be very well fitted to a special problem.

AMIGOS is equipped with three layers of access to serve the needs of the variety of users (Fig. 1) but in contrast with previous generations of software just the *platform developer* requires access to and modification of the source code. Even during model development the analytical user input will be interpreted, optimized, transformed and solved on any complex simulation domain at once without the necessity of time consuming recompilations (*one-pass concept*) supporting a variety of several testing and debugging features. After finishing the test and calibration phase the user can switch to the *two-pass* concept where all modifications are translated to C-code and are linked to a model library for high performance calculations on large simulation domains in the standard user-mode.

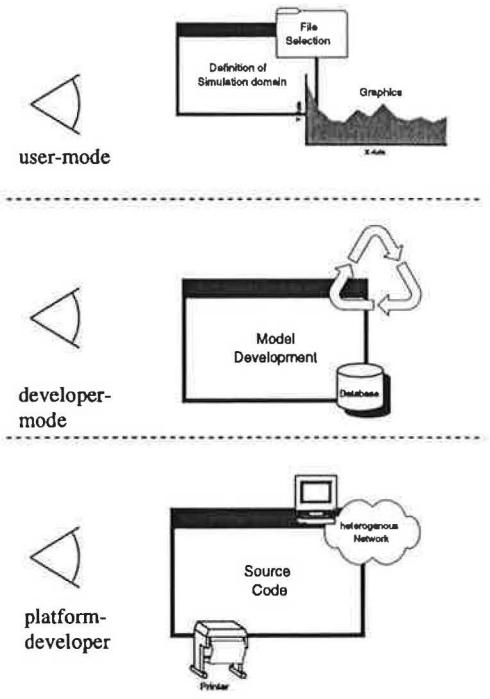

Figure 1: Several different user perspectives

Analytical Modeling Interface

AMI is a kind of precompiler which transforms the analytical user input of any partial differential equation system into an optimized numerical model. To achieve a simple user interface we have developed an interpreter in the manner of programs like Mathematica, Mathcad, Matlab, etc., which is tailor-made for optimized matrix operations as well as for other mathematical expressions (e.g., +, -, *, /, sin, sqrt, etc.) and for operations which are necessary to solve numerical problems. To minimize the evaluation time for a model every once calculated operation is reused, zero operations are eliminated, and, furthermore, an analytical optimizer simplifies the input to reduce the number of operations even further. Especially, in use with matrix operations it reduces the number of operations by an average of about two third depending on the complexity of the analytical model being optimized.

AMI supports all standard discretization methods (e.g., finite elements,finite boxes and finite differences) and poses no restrictions for use of numerical tricks (e.g., penalty terms, etc.) which are often utilized to improve the convergence of nonlinear problems.

As a result AMI generates a residual vector and the Jacobian of a single grid element for use in our nonlinear General Object-oriented Solver (GOS). Furthermore, derived quantities (scalars, field quantities) and parameters can be defined and will be automatically calculated if necessary. External user functions implemented in C can be linked and used within the analytical input interface.

General Object-oriented Solver

GOS includes all necessary functions to solve a discretized numerical problem (e.g., Newton iterator, time-step control, etc.) on a given simulation domain as well as complete grid management in one, two and three space dimensions. It holds the complete information about the simulation domain and uses AMI 's output to build the global stiffness matrix. Furthermore it has a built in Equation System Analyzer (ESA) that detects the couplings between quantities inside and between several segments of the simulation domain so that the global stiffness matrix can be cut into smaller pieces which leads to best conditions for subsequent memory expensive and time consuming solution steps.

To simplify the mapping of all abstract variables defined in AMI to the real simulation domain GOS is equipped with a simple grid and boundary description language for the definition of all necessary boundary and volume instructions as well as for selecting different simulation modes.

Three-Dimensional Temperature Distribution

The following example is a study of the performance and memory management of AMIGOS against a hand-coded finite element solver (STAP – SMART THERMAL ANALYSIS PROGRAM [2]). Although a point problem comparison against a single simulator may not be indicative of the relative efficiency of AMIGOS against every specially optimized limited-purpose PDE-based simulator, this example shows that the performance of AMIGOS is competitive with that of a hand-coded simulator.

A two layer interconnect structure has been chosen as benchmark example where temperature and potential distributions are calculated. The model for the numerical calculation of joule self heating effects is based on two partial differential equations. First, the heat-conduction equation

$$c_p \rho \frac{\partial T}{\partial t} - div\,(\gamma_T \cdot grad(T)) = p \qquad (1)$$

has to be solved to calculate the temperature distribution T in conductor and insulator domains, where c_p is the specific heat and ρ is the mass density of the material. γ_T represents the material specific thermal conductivity, t is the time, and the term on the right side p is the electrical power loss density.

For stationary problems the heat-conduction equation can be simplified to

$$div\,(\gamma_T \cdot grad(T)) = -p \qquad (2)$$

The power loss density p is derived from the electric potential φ

$$p = \gamma_E \cdot (grad(\varphi))^2 \qquad (3)$$

that is calculated by solving the Euler Equation

$$div\,(\gamma \cdot grad(\varphi)) = 0 \qquad (4)$$

The *electrical* conductivity which in this case is a function of the temperature will be modeled by

$$\gamma(T) = \gamma_0 \cdot \frac{1}{1 + \alpha\,(T - T_0)} \qquad (5)$$

In this model the dependence of γ on the temperature T is determined by the coefficient α. γ_0 is the conductivity at the reference temperature T_0 of 300K.

The comparisons were made calculating 14863 nodes building 73728 tetrahedrons. The calculated results are shown in Fig. 4 to Fig. 6.

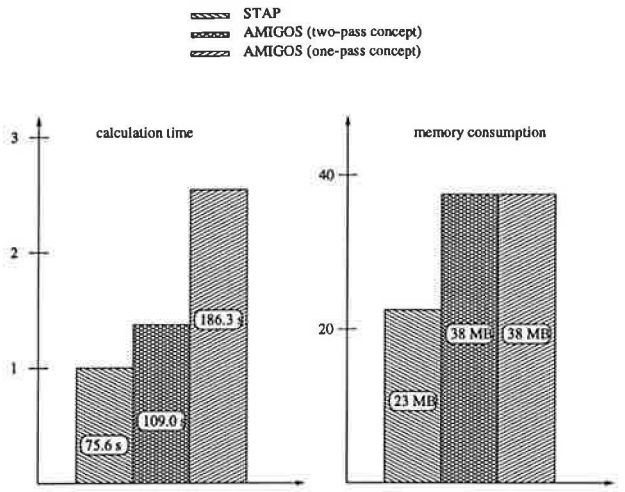

Figure 2: Performance comparison of calculation time and memory consumption

Whereas Fig. 2 shows that the STAP system is quite faster than AMIGOS using less memory since it exploits the symmetry of the resulting sparse matrix for compression methods that can not be used by AMIGOS for the sake of flexibility. Fig. 3 shows that AMIGOS rapid prototyping abilities are unbeatable when developing a new model (in this case a transient electric simulation which was in scope of feasibility for the STAP system).

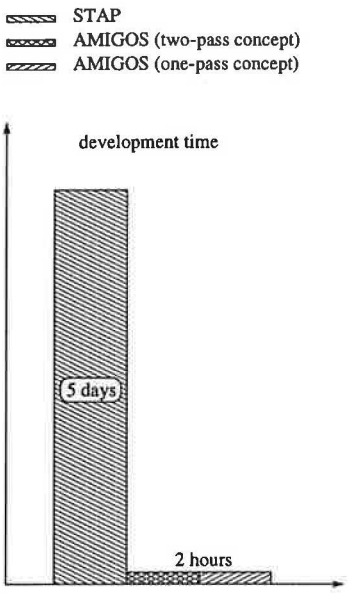

Figure 3: Performance comparison of development time

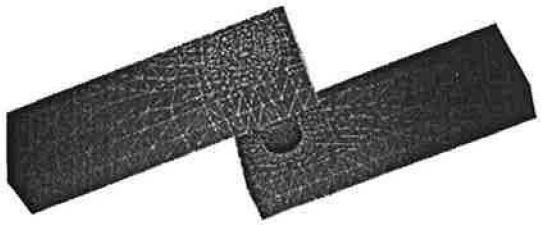

Figure 4: Calculated potential distribution showing the mesh of discretization

Figure 5: Calculated temperature distribution within the two layered interconnected structure

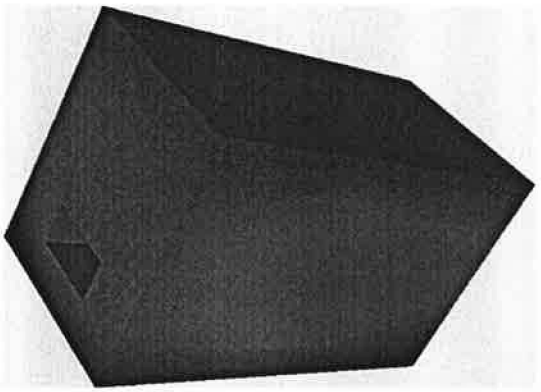

Figure 6: Calculated temperature distribution within the surrounding insulator

References

[1] D.W. Yergeau, E.C. Kan, M.J. Gander, and R.W. Dutton. Alamode: A layered model developement environment. In H. Ryssel and P. Pichler, editors, *Simulation of Semiconductor Devices and Processes*, volume 6, pages 66–69, Wien, 1995. Springer.

[2] R. Sabelka, R. Martins, and S. Selberherr. Accurate layout-based interconnect analysis. In H.De Mayer and S.Biesemans, editors, *Simulation of Semiconductor Devices and Processes*, pages 336–339, Wien, 1998. Springer.