Parnass2: A cluster of Dual-Processor PCs

Marc Alexander Schweitzer[†‡], Gerhard Zumbusch[†]and Michael Griebel[†]

Abstract

We report on a cluster with 96 CPUs (48 Dual-Processor PCs running Linux 2.1.127) at our department, called Parnass2[1] [15]. The computing nodes are connected by a Myrinet, a Gigabit per second switched LAN, and additionally by a Fast-Ethernet. A comparison of different message passing (MPI) libraries for the Myrinet is given. Furthermore, we present the performance results of Parnass2 for some of the parallel codes developed at our department, namely a sparse grid code for PDEs, a particle code for molecular dynamics, a finite difference code for the Navier–Stokes equation, and a parallel adaptive finite element / finite difference multigrid code. We compare these results with the performance of these codes running on a Cray T3E-1200, a Cray T3E-600, a SGI Origin 200/2000 and Parnass [8], a cluster of SGI O2 workstations.

1 Introduction

A few years ago high performance computing power could only be achieved by medium grained parallel computers based on propriety components. The dramatic performance increase of mass-produced microprocessors over the last few years made the dedicated high-end processor development too expensive (and dispensable). Todays supercomputers are parallel computers consisting of cheap, standard, yet very fast microprocessors usually connected via a propriety network. The next step in this development will be the replacement of these propriety networks with standard networks, provided that they are similarly efficient (> 1 GBit/s bandwidth, < 5 μs latency). Among the standard networks ATM, HIPPI, Fibre Channel, GigaBit Ethernet, SCI and Myrinet, the Myrinet is presently the cheapest Gigabit per second LAN/SAN available ($\sim$ \$1500 per node).

In §2 we give a short discussion on the Myrinet and its technical specifications; different programming models for clusters of multi-processor nodes are discussed in §3. In §4 we report on the hardware and software of our cluster Parnass2, which currently consists of 96 CPUs (48 dual-processor nodes) and has a peak performance of 38.4 Gflops/s. We benchmarked Parnass2 with several codes

[†]Institut für Angewandte Mathematik, Universität Bonn, Wegelerstr. 6, D-53115 Bonn, {schweitz, zumbusch, griebel}@iam.uni-bonn.de

[‡]In part supported by the Sonderforschungsbereich 256 *Nonlinear Partial Differential Equations* at the University of Bonn.

[1] Parnass2 is the successor project to Parnass [8], a cluster of workstations. Being the successor and consisting of Dual-Processor nodes, the cluster was named Parnass2. The project is supported in part by the Sonderforschungsbereich 256 *Nonlinear Partial Differential Equations* at the University of Bonn.

which were developed at our department. The results of these experiments are presented in §5. Here we also compare these performance results with those of a Cray T3E-1200, a Cray T3E-600, a SGI Origin 200/2000 and Parnass running these codes.

2 Myrinet

The Myrinet system is a low cost, high speed network at local (LAN) or system area (SAN). The network may be set up using a set of programmable PCI-Bus or S-Bus network interface cards (NIC) and 8-port (or the new 16-port) crossbar cut-through switches. The components are connected by copper cables at a raw bandwidth of 1280 MBit/s full duplex. The network topology and the communication protocol are completely arbitrary.
The Myrinet adapter contains a custom RISC processor, static 0.5 or 1 MByte RAM (SRAM) for the program and for the data, a field programmable gate array (FPGA), a network interface with two network DMA engines, and a bus interface to the workstation with a DMA engine, see Figure 1.

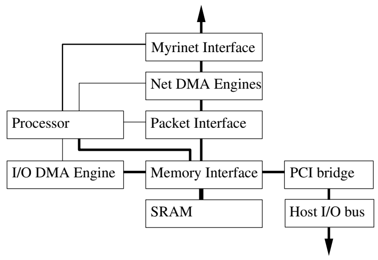

Figure 1: Layout of a Myrinet network interface card (NIC).

Myrinet is intended as local area network or system area network for the construction of parallel computers based on message passing. The resulting parallel computer is a two-level system: The NICs can be programmed to implement the message passing protocol and perform the data transfers, while the workstations serve as compute nodes attached to the NIC, see Figure 2.

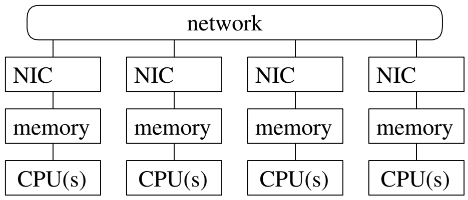

Figure 2: Structure of a two-level multicomputer.

There are multicomputers build of Myrinet components with different kinds of computing nodes. Computing nodes can be signal processors, FPGAs (e.g. for pattern recognition), VME single board computers, (multi-processor) PCs and (multi-processor) workstations. The strategy is to combine a high performance network by one vendor with high performance processors by another vendor, which suit the applications best. Instead of developing both components at once, one can participate in the advances of processing technology and of networking technology independently. Using mass-produced components gives the best overall price-performance ratio. For a more detailed discussion of the Myrinet, its hardware and technical specifications see [8] and the references within.

3 Programming Models

Software is the key to efficiently combine a high performance network like the Myrinet with high speed computing nodes. Furthermore, it is essential to provide efficient standard programming interfaces like MPI (or PVM) in order to allow for portability onto/from other platforms of developed codes. Thus it is an issue of great importance, that (efficiently implemented) message passing libraries exist, which exploit the Myrinet's $\sim$ 1 GBit/s bandwidth and $\sim 5\ \mu$s latency.
Currently there are many efforts to develop message passing libraries for the Myrinet: BIP-MPI, GM-MPICH on GM, GM-MPICH on BIP, HPVM/FM, MPICH on BullDog and MPICH-PM. Furthermore, there are a lot of message passing libraries, which operate on TCP/IP over Myrinet (MPICH, LAM, PVM, ...). We conducted experiments with all of the implementations available to us on a reference configuration of Parnass2, see §4.3. Here it turned out, that most of the libraries (as of now) do not support multi-processor computing nodes.
Some of the libraries may be safely used in a cluster of multi-processor nodes (CLUMP), but they do not exploit the multi-processor performance since they only allow for the execution of a single job which locks the NIC. Hence, these libraries constrain the programmer to a two-level parallel programming paradigm like the use of OpenMP for the parallelization of each message passing job on a single multi-processor computing node. This programming model is far more complex, therefore more prone for design and programming errors. Furthermore, in many applications, the first level parallelization with the message passing interface will lead to intrinsically serial algorithms for each process, where multithreaded shared memory parallelization with OpenMP may not be efficiently applied. Still the programmer may also complete the second level parallelization by hand. This will then lead to an even more complex programming model and non-portable code and is therefore inadvisable.

4 Parnass2

Parnass2 is a dedicated cluster of personal computers (PCs) used for research in scientific computing at the Department of Applied Mathematics at the University of Bonn. This resource is used as a supercomputer to carry out computational tasks which are too large for conventional workstations or servers. The cluster utilizes off-the-shelf components (Intel Pentium-II, Linux, Fast Ethernet), hence it can achieve multi-Gflops performance for around $100K. But more importantly, on real applications it competes surprisingly well with systems costing 10–100 times more. Because the system is inexpensive (about the cost of a high-end workstation), it can be used as a *personal supercomputer*, rather than a resource that must be shared by many people.

4.1 Hardware

In the current installation Parnass2 consists of 96 CPUs (48 Dual-Pentium-II 400 MHz). These dual-processor computing nodes (256 MB RAM, 8.4 GB HD) are connected by a Myrinet in a fat-tree like topology (see Figure 4) and a Fast-Ethernet. Furthermore, there is an additional designated login server (Dual-Pentium-II 400 MHz, 512 MB RAM, 8.4 GB HD, 2 $\times$ Fast-Ethernet) and an additional NFS server (Dual-Pentium-II 400 MHz, 512 MB RAM, 3 $\times$ 8.4 GB HD, Fast-Ethernet), see Figure 3. Therefore the computing nodes are fully symmetric, giving Parnass2 a peak performance of 38.4 Gflops/s and 12 GByte main memory. In a Linpack benchmark we were already able to measure a performance of 18.3 Gflops/s. In the final installation Parnass2 will employ 128 CPUs (64 Dual-Pentium-II, 512 MB RAM, 8.4 GB HD). The computing nodes will be connected in a fat-tree Myrinet topology (see Figure 5). The peak performance of Parnass2 will be 51.2 Gflops/s and the cluster will have 32 GByte main memory. The overall prize of this final installation is \$280K, giving an anticipated prize-performance ratio of \$5.500 per peak Gflops/s and \$12K per sustained Gflops/s.

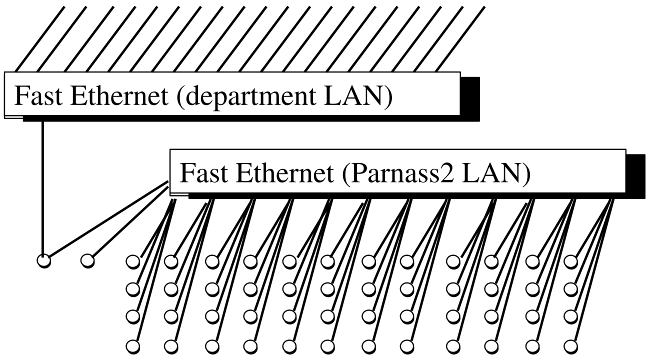

Figure 3: Ethernet configuration of Parnass2.

4.2 Network

The current Myrinet installation of Parnass2 employs three switches M2M-OCT-SW8, three switches M2FM-SW8 (as interface adapter LAN to SAN), 12 Myrinet LAN adapters and 36 Myrinet SAN adapters. These components are connected in a two-level fat-tree like topology (Figure 4), giving a full bisection bandwidth of 61 GBit/s network performance. In its final installation of 64 computing nodes the network (Figure 5) will have a full bisection bandwidth of 82 GBit/s.

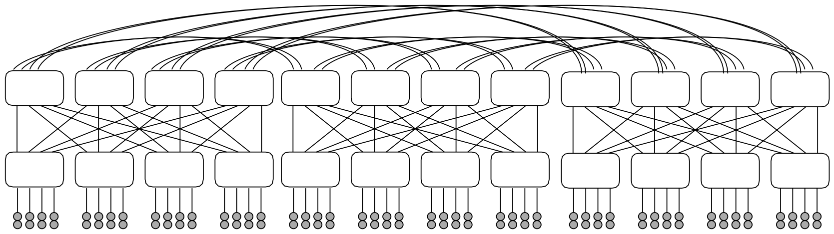

Figure 4: Two level fat-tree like topology of 96 CPUs (48 nodes).

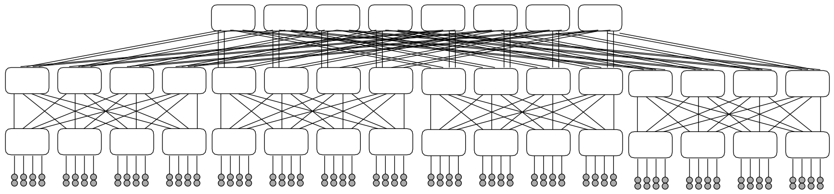

Figure 5: Three level fat-tree topology of 128 CPUs (64 nodes).

4.3 Software

The operating system of Parnass2 is Linux (RedHat 5.1). Currently we use the 2.1.127 kernel with SMP support. Compilers installed on the Parnass2 login server are the GNU compilers 2.8.1 `gcc`, `g++`, `g77` and the experimental GNU compilers (egcs) 1.1b. Other development tools and libraries installed are the Gnu Debugger 4.17 `gdb`, STL, Lapack, Scalapack, Blas, Blacs and PBlacs.
To determine the primary message passing environment to be used on Parnass2, we conducted a series of experiments with all message passing implementations available to us: Myricom TCP, GM, BIP, HPVM, BullDog, and Score. The first observation was that at the time we performed these experiments (10/1998) almost all of the non-TCP/IP based message passing interfaces did not (fully) support multi-processor computing nodes. The BIP library allowed for the use of a multi-processor computing node but did not exploit the multi-processors feature of executing multiple processes, since a single job running on a computing node monopolizes the Myrinet adapter. HPVM and Score 2.4 are currently the only message passing environments (not based on TCP/IP) that allow for multiple jobs on a single computing node. So the experiment which could be performed with most of the systems was just a one-to-one inter-node (Figure 6) bandwidth test (Table 1). We also performed a one-to-one intra-node (Figure 6) bandwidth test (Table 1) with the libraries based on TCP/IP and the libraries providing their own MPICH device capable of multi-processing in a node.
These results clearly show that the bandwidth of the TCP/IP based libraries for the Myrinet reduce the raw bandwidth of the Myrinet (1280 MBit/s) to a level close to the bandwidth of a Fast-Ethernet. From the non-TCP/IP based message passing systems, only Score 2.4/MPICH-PM [14] shows a bandwidth performance close to the anticipated > 1 GBit/s in the inter-node communication benchmark (in spite of the results published by the developing research groups themselves). Furthermore, Score 2.4/MPICH-PM is currently the only

Figure 6: one-to-one communication benchmark (inter-node on the left, intra-node on the right).

Software distribution	MPICH device	Bandwidth (MBit/s) inter-node	intra-node
TCP Myricom	chameleon	80	245
TCP on GM 0.2	chameleon	u/d	u/d
BIP-IP 0.95b	chameleon	180	245
BullDog/BDM	myri	300	n/a
BIP-MPI 0.95b	gmpi	350	n/a
GM-MPICH on BIP 0.95	GM	-	u/d
GM-MPICH on GM 0.2	GM	u/d	u/d
HPVM/FM 1.0a	FM	550	220
Score 2.2/MPICH-PM	score	500	n/a
Score 2.4/MPICH-PM	score-pmvm	850	1150

Table 1: Performance of different communications libraries in a one-to-one communication benchmark.

system that is able to (fully) exploit the fast process communication inside a multi-processor node. With the HPVM/FM it seems, that one processor is sending its data to the Myrinet adapter where the second processor will then collect the data, hence reducing the bandwidth even further.
Overall Score 2.4/MPICH-PM fulfills our demand for multiple jobs per computing node and it is the fastest message passing system on the Myrinet at the present point in time, hence its use as the primary message passing environment on Parnass2 is quite clear.

5 Applications

The main purpose of Parnass2 is to serve as a platform for our activities in high performance scientific computing. Several parallel applications for the solution of partial differential equations are already running on Parnass2, some others are still under development. We present the performance results of Parnass2 for the following codes:

- A sparse grid finite difference code [13]. Here, a space filling curve is used for dynamic load balancing, data is stored in a parallel hash table.
- An adaptive finite element / finite difference code with a multi-grid solver [9, 10]. We also use the space filling curve heuristic for dynamic load balancing in this code and data is also stored in a parallel hash table.
- MolGrid: A linked cell code for the simulation of molecular dynamics [2], i.e. Hamiltonian systems. We use classical domain decomposition for the parallelization of this code.

- NaSt3D\GP: A finite difference fluid dynamics code [5]. Here, again, classical domain decomposition is used for the parallelization of the code.

These results are compared to the performance of a Cray T3E-1200, a Cray T3E-600, an SGI Origin 200/2000 and Parnass, a cluster of SGI O2 workstations [8] running the same code.

Sparse Grid Finite Difference PDE Solver for Convection-Diffusion-Problems. The first benchmark problem we present is a three-dimensional convection-diffusion problem discretized using finite differences on a uniform sparse grid [12]. In Tables 2, 3 we display the execution times on Parnass2 and a Cray-T3E-600 (300 MHz Alpha CPUs).

time		processors						
nodes	1/h	1	2	4	8	16	32	64
81	4	0.03	0.03	0.07	0.08	0.11		
225	8	0.12	0.09	0.09	0.11	0.16	0.20	0.20
593	16	0.63	0.41	0.33	0.32	0.38	0.44	0.53
1505	32	3.78	2.29	1.60	1.34	1.26	1.33	1.53
3713	64	22.1	13.3	8.79	6.39	5.17	4.47	4.47
8961	128	68.1	40.7	24.8	16.2	11.9	8.89	7.56
21249	256	201	119	66.1	40.1	28.0	18.6	13.5
49665	512	575	379	169	106	71.6		28.0
114689	1024	1630			275	179		62.6

Table 2: Parallel execution times for the solution of a 3D convection-diffusion problem on sparse grids (maximum of 100 iterations) on Parnass2.

time		processors				
nodes	1/h	1	4	16	32	64
81	4	0.04	0.06	0.10	0.14	0.16
225	8	0.25	0.20	0.38	0.54	0.57
593	16	1.31	0.71	0.89	1.24	1.53
1505	32	10.0	3.66	3.03	3.67	5.75
3713	64	52.5	20.1	12.8	11.9	13.1
8961	128	158	58.2	29.6	22.6	20.9
21249	256	473	192	66.6	44.9	67.6
49665	512	1426	396	156	95.4	62.0
114689	1024	4112				125

Table 3: Parallel execution times for the solution of a 3D convection-diffusion problem on sparse grids (maximum of 100 iterations) on a Cray T3e-600.

A direct comparison of the execution times reveals that the 300 MHz Alpha processors in the Cray T3E-600 configuration perform roughly a factor 2.5 slower than the Parnass2 nodes, but the highly sophisticated T3E network scales slightly better than the Parnass2 Myrinet network.

Adaptive Finite Elements and Multi-Grid Solver for Linear Elasticity Problems. The Lamé equation in the displacement formulation

$$\begin{array}{rcll} \mu\vec{\Delta}\vec{u} + (\lambda+\mu)\nabla(\nabla\cdot\vec{u}) & = & \vec{f} & \text{in } \Omega \subset \mathbb{R}^3 \\ \vec{u} & = & \vec{0} & \text{on } \Gamma_D \subset \partial\Omega \\ \sigma(\vec{u})\cdot\vec{n} & = & \vec{0} & \text{on } \Gamma_N = \partial\Omega \setminus \Gamma_D \end{array} , \qquad (1)$$

time		processors						
nodes	dof	1	2	4	8	16	32	64
125	375	0.10	0.12	0.11	1.50	2.91		
450	1350	1.44	0.99	0.80	1.35	0.50	0.39	1.05
1155	3465	4.14	2.48	1.71	1.32	1.00	0.70	2.74
4412	13236	19.0	10.3	6.09	5.23	3.07	1.89	1.21
18890	56670	98.6	50.3	28.1	20.6	11.6	6.35	3.70
93021	279063	582	294	157	102	54.8	28.2	15.1
506620	1519860				556	306	155	78.1
3178218	9534654							494

Table 4: Parallel execution times (in seconds) for the solution of Lamé's equation in 3D, Parnass2.

time		processors								
nodes	dof	1	4	16	64	128	256	512	768	1024
35937	107811	162	34.1	9.11	2.23	1.23	0.75	0.55	0.56	0.52
109873	329619	435	108	29.6	7.20	3.57	1.87	1.13	0.91	0.80
410546	1231638			114	28.6	14.2	7.02	3.51	2.48	1.94
1857030	5571090				133	67.1	33.3	16.5	11.0	
9619175	28857525					351				

Table 5: Parallel execution times (in seconds) for the solution of Lamé's equation in 3D, CrayT3E-1200.

with Lamé's constants

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \qquad \mu = \frac{E}{2(1+\nu)}$$

and linear strain $\epsilon(\vec{u}) = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i})$, $\lambda > 0$, $\mu > 0$ and $E > 0$, was considered for our performance test with the adaptive finite element multi-grid solver. The initial grids on the two machines had to be chosen differently to allow scaling up to 1024 processors on the Cray. Therefore the number of unknowns per level and the corresponding amount of work for the multi-grid cycle are not the same on the two machines. We would have to complete a detailed analysis of the numerics to really compare these numbers in detail. However we learn already from these numbers that the code scales as well on Parnass2 as it does on the Cray.

Molecular Dynamics. The third code we used for benchmarking Parnass2 was a classical linked cell molecular dynamics code, called MolGrid [2]. The test problem was the super-cooling and crystallization of Argon, the potential used in this simulation is the short-range Lennard–Jones potential. The results displayed in Table 6 clearly show the superior performance of a single Parnass2 computing node (i.e. 2 CPUs) to a pair of MIPS R10000 CPU of an SGI Origin 200/2000 or two nodes of Parnass (i.e. a pair of O2s connected by Fast-Ethernet). Furthermore, the Parnass2 network shows a performance similar to the shared memory performance of an SGI Origin 200/2000.

Three-dimensional Fluid Dynamics. As a final performance test for Parnass2 we used our conventional finite difference Navier–Stokes code and simulated the 3D backward-facing-step (see page 117 in [5]) and the 3D channel-flow

platform	time particles	processors 2	4	8	16	32	48	64
Parnass2	35e2	0.04	0.18	0.02	0.01	0.09		
	30e3	0.49	0.26	0.15	0.10	0.16	0.12	0.11
	25e4	4.35	2.42	1.28	0.84	0.54	0.38	0.35
	20e5	36.68	18.14	9.76	5.23	2.61	1.93	1.86
	16e6					69.85		
Parnass	25e4	59.67	29.94	15.19	8.93			
Origin 200/2000	25e4	13.37	7.51					

Table 6: Parallel execution times (in seconds) for the super-cooling of Argon, Parnass2, Parnass and Origin 200/2000.

problems. Here, again, the insight we get from Table 7 is the fact that the

time platform	cells	processors 1	2	4	8	16	32	64
Parnass2	4.5e5	65	37.75	18.5	10.45	5.25	2.25	1
Origin 200/2000	4.5e5	81	46	23	–	–	–	–
Parnass2	3.5e6	–	1800*	80	42	21	11	5.5

Table 7: Parallel execution times (in seconds) for the backward facing step, Parnass2 and Origin 200/2000. Parallel execution times (in seconds) for the channel flow, (* due to swapping) Parnass2

code does scale as well on Parnass2 as on the shared memory machine Origin 200/2000.

Outlook. Currently there are several other parallel applications under development at our department, namely a hierarchical adaptive tree-code (TreeMol) for molecular dynamics [2], a smoothed particle hydrodynamics code [1] for astrophysics, an eigenvalue solver for Schrödinger's equation [3] based on the sparse grid combination technique [6], a wavelet based Navier–Stokes solver [11] and a parallel algebraic multi-grid (AMG) solver [16, 4, 7].

Acknowledgements. The computing time on the Cray T3E-1200 and T3E-600 was donated to the department by Cray Research.

References

[1] A. Caglar, *Multilevel-SPH höherer Ordnung*, tech. rep., Sonderforschungsbereich 256, 1999. in preparation.

[2] A. Caglar and M. Griebel, *Fast Parallel Algorithms for Molecular Dynamics*, Journal of Molecular Liquids, (1999). to appear.

[3] J. Garcke, *Berechnung von Eigenwerten der stationären Schrödingergleichung mit der Kombinationstechnik*, Master's thesis, Institut für Angewandte Mathematik, Universität Bonn, 1998.

[4] T. Grauschopf, M. Griebel, and H. Regler, *Additive multilevel-preconditioners based on bilinear interpolation, matrix dependent geometric coarsening and algebraic multigrid coarsening for second order elliptic pdes*, Applied Numerical Mathematics, 23 (1997). also as SFB Report 342/02/96A, Institut für Informatik, TU München, 1996.

[5] M. Griebel, T. Dornseifer, and T. Neunhoeffer, *Numerical Simulation in Fluid Dynamics, A Practical Introduction*, SIAM, Philadelphia, 1998.

[6] M. Griebel, M. Schneider, and C. Zenger, *A combination technique for the solution of sparse grid problems*, in Iterative Methods in Linear Algebra, P. de Groen and R. Beauwens, eds., Elsevier, 1992, pp. 263–281.

[7] M. Griebel and T. N. und H. Regler, *Algebraic multigrid methods for the solution of the navier–stokes equations in complicated domains*, Int. J. Numer. Methods for Heat and Fluid Flow, 26 (1998), pp. 281–301. also as SFB report 342/1/96A, Institut für Informatik, TU München, 1996.

[8] M. Griebel and G. Zumbusch, Parnass*: Porting Gigabit-LAN components to a workstation cluster*, in 1. Workshop Cluster Computing 1997, TU Chemnitz, 1997. also as Tech. Rep. 19, Sonderforschungsbereich 256, Institut für Angewandte Mathematik, Universität Bonn.

[9] ——, *Parallel multi-grid in an adaptive PDE solver based on hashing*, in ParCo'97, E. D'Hollander, G. R. Joubert, F. J. Peters, and U. Trottenberg, eds., Elsevier, 1998, pp. 589–599.

[10] ——, *Parallel Adaptive Subspace Correction Schemes with Applications to Elasticity*, CMAME, (1999). submitted.

[11] F. Koster, *Turbulence Simulation using Wavelets*, tech. rep., Sonderforschungsbereich 256, 1999. in preparation.

[12] T. Schiekofer, *Die Methode der Finiten Differenzen auf dünnen Gittern zur Lösung elliptischer und parabolischer partieller Differentialgleichungen*, PhD thesis, Institut für Angewandte Mathematik, Universität Bonn, 1998.

[13] G. Zumbusch, *A Sparse Grid PDE Solver*, in Modern Software Tools in Scientific Computing (Proceedings SciTools' 98), Basel, 1999, Birkhäuser.

[14] http://www.rwcp.or.jp/lab/pdslab/, http://www.rwcp.or.jp/lab/pdslab/score/manuals.html.

[15] http://wwwwissrech.iam.uni-bonn.de/research/parnass2.

[16] http://wwwwissrech.iam.uni-bonn.de/research/projects/schweitz/amg.html.