T U M

I N S T I T U T   F Ü R   I N F O R M A T I K

Parallel Communication on Workstation Networks with Complex Topologies

Alexander Pfaffinger

TUM-I9514
Mai 1995

TECHNISCHE UNIVERSITÄT MÜNCHEN

TUM-INFO-05-1995-I9514-/.-FI

Druck: Mathematisches Institut und
Institut für Informatik der
Technischen Universität München

Parallel Communication on Workstation Networks with Complex Topologies

Alexander Pfaffinger *

Institut für Informatik
Technische Universität München
Arcisstraße 21, D-80290 München 2

May 11, 1995

Abstract

Workstation clusters offer a cheap and powerful alternative to parallel processors. For communication intensive distributed applications the single Ethernet connection is a global resource and soon becomes a bottleneck that prohibits scalability. Our suggested solution is to build complex multiple-bus topologies with a second Ethernet link per workstation. This paper introduces two classes of bus networks for workstation clusters: a tree-like topology for divide-and-conquer algorithms and a hypercube-like network for a broader class of applications. We present some graph-theoretic aspects, simulation results, and numerical test data on a network with 16 workstations.

1 Introduction

In the field of parallel high performance computing, there is a trend towards workstation clusters, which are usually interconnected via Ethernet buses.

*This work was supported by the Siemens AG.

Networked workstations are already available at many sites and typically their computing capacity is not fully exploited. Therefore, they offer a cost-effective alternative to dedicated parallel computers.
Recently, a number of machine independent parallel platforms like MPI, PVM, and LINDA has been developed. They support the distributed programming of a workstation cluster. These systems are now widely used in parallel numerical computing.
With increasing demand for communication between the parallel tasks, however, the bus-like nature of the Ethernet soon becomes a bottleneck. Hence, only few computing nodes can be used efficiently. The system is not scalable with respect to the number of processors.
By introducing additional buses and a more complex interconnection topology of the workstations, the communication bandwidth can be significantly improved. The required hardware investment are just some additional LAN adapter cards and Ethernet cables. Depending on the network topology, pairs of workstations may now communicate in parallel.
This matter has already been studied intensively for tightly coupled systems (see e.g. [6]), but not yet in the specific context of workstation networks. In particular the characteristics of an Ethernet, the high process generation costs under UNIX, and the network administration must be considered.
In this paper we will describe two network topologies which both need only two I/O-ports per workstation. In section 2 we will sketch out a tree-like structure which is optimally adapted to divide-and-conquer algorithms. In section 3 we will present a more general hypercube-like topology. Section 4 gives some simulation results.

2 A Hierarchical Topology

A common model for parallel computing is the divide-and-conquer paradigm: a task is split into two (or more) subtasks that can be executed in parallel. Each of these subtasks can be divided in subsubtasks and so on.
This leads to a tree-like dependence graph. In the case of exactly two subtasks per node we obtain a binary tree. This graph also reflects the communication scheme of the parallel tasks. A task needs to talk only to its parent and its children.
When we consider the distribution of the jobs onto the computing nodes, we

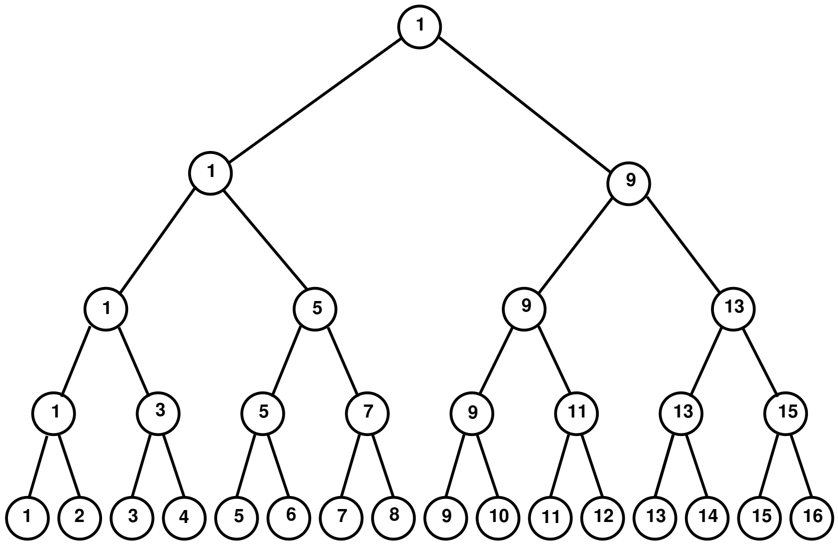

Figure 1: Binary tree with 16 computing nodes: dependence and communication graph of a parallel divide-and-conquer application. Each interior node is identified with its leftmost descendent.

get a slightly different graph. The parallel algorithm starts at node one. At the point of division two new tasks are created, which should be placed at different nodes. Since the parent is idle when the two subjobs are calculating, one subtask can remain on the initial node. This leads to a "squashed" binary tree. Figure 1 shows a divide-and-conquer graph with 16 computing nodes. Each left child of a node is identified with its parent.

We will discuss the communication pattern in more detail. Most importantly, we must distinguish between the cheap communication within a node and the expensive external communication between different nodes. The first one arises between parent and left child in Figure 1. Only the communication of each parent with its right child is external.

If we assume that all tasks at one layer of the tree need quite the same computation time, there are no simultaneous communications at different (edge-) levels of the tree. Therefore, it is desirable that all communications on one level are done in parallel, i.e. on different buses. The problem is to find a minimal bus topology in which at each level all nodes can communicate with their right child via different buses.

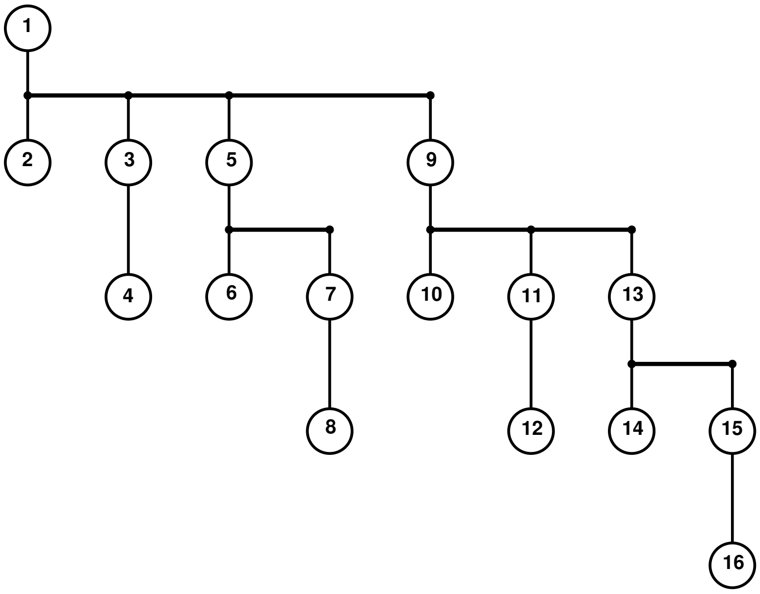

Figure 2: Tree-like Ethernet topology: all communication between parent and (right) child at one level of Figure 1 can be done in parallel via different buses.

Of course, this could be achieved by a complete interconnection, where each node is connected to any other node via a different bus (or link). This would mean that for n processing elements each node needs to have $n-1$ I/O-ports. But real hardware will always show a constant maximum. In this paper we will examine topologies where only two I/O ports per workstation are needed.

Figure 2 shows an appropriate tree-like solution with 8 buses that is obviously minimal in respect to the number of buses (we have 8 parallel communication at the deepest level). Its hierarchical structure reflects the recursive definition of the corresponding squashed binary tree.

The next higher network, which includes 32 nodes and 16 buses, would be constructed as follows: take a copy of the 16-node-network, add 16 to all of its node labels, and put the least labeled node (17 here) with an additional link into the bus that connects 1 and 2. This topology corresponds perfectly to the complete squashed binary tree with 32 leaf nodes.

In general, for a complete binary tree with 2^d leaves the corresponding bus network would consist of 2^d computing nodes and 2^{d-1} buses. Each node is connected to at most 2 buses and each bus is assigned to at most $d+1$ nodes. For each level in the binary tree all data transfer at that level from parent to child (or vice versa) can be done in parallel. This is a prerequisite of scalability.

Numerical Applications

The architecture shown in Figure 2 has been implemented with 16 HP-720 workstations. Each odd-numbered node has been provided with an additional LAN adapter card. Node 1 is used as the gateway to outside networks.
On this installation several applications with a divide and conquer strategy have been parallelized. Besides the complex chip placement algorithm GORDIAN [8] this is the parallel Finite Element application ARESO [5, 9] which we will describe somewhat more in detail below.
ARESO is a solver for partial differential equations that is based on domain decomposition and recursive substructuring. In the case of a square domain ARESO starts on top level with a certain problem size N, typically a power of 2, that describes the number of grid points on the borderlines.
When the problem is split up in two parts, the size decreases by the factor $1/\sqrt{2}$. (More accurately, N is halved every second step.) At level l we, therefore, get $N_l = N/\sqrt{2}^l$. The computation time per node on level l is of order $O(N_l^3)$ while the communication amount per node is $O(N_l^2)$. This means that the message size per node decreases with increasing level l, but the number of communicating nodes increases. With a single Ethernet this leads to high collision rates.
Figure 3 compares the runtime of ARESO for a fixed problem size on the single bus and on the tree topology. At 4 nodes the tree structure has only a slight advantage over the single bus. But involving 8 and 16 computers the single bus is overloaded so that no speed up is obtained. The runtime is even longer than for 4 nodes. For the tree topology, in contrast, the algorithm still shows a speed up.
It is interesting that the advantage of multiple buses is not only restricted to divide-and-conquer algorithms. In [11] the runtime behavior of a parallel unstructured matrix application on the different topologies was compared. It concerns the distributed solution of a big sparse block matrix system.

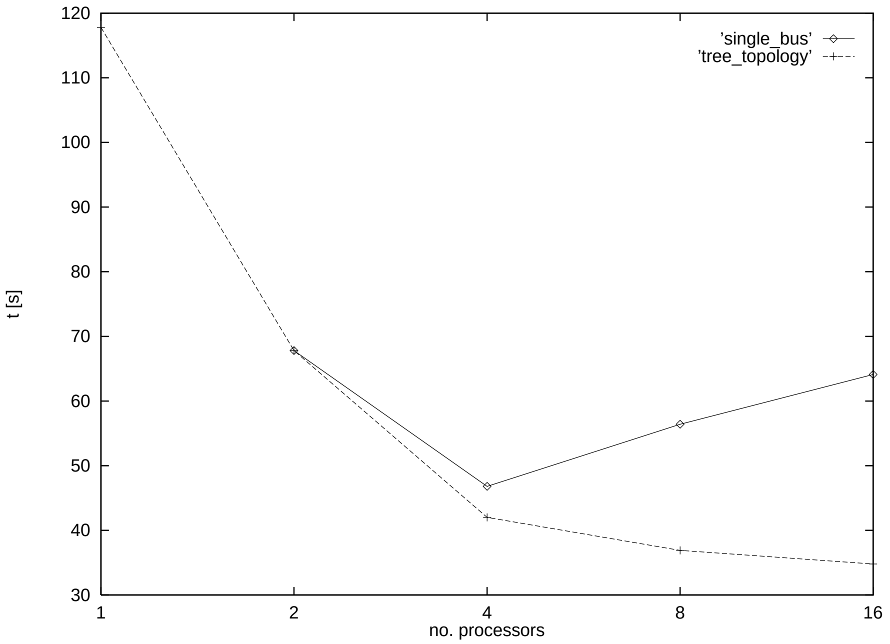

Figure 3: ARESO runtimes on the single bus and the hierarchical tree-like topology.

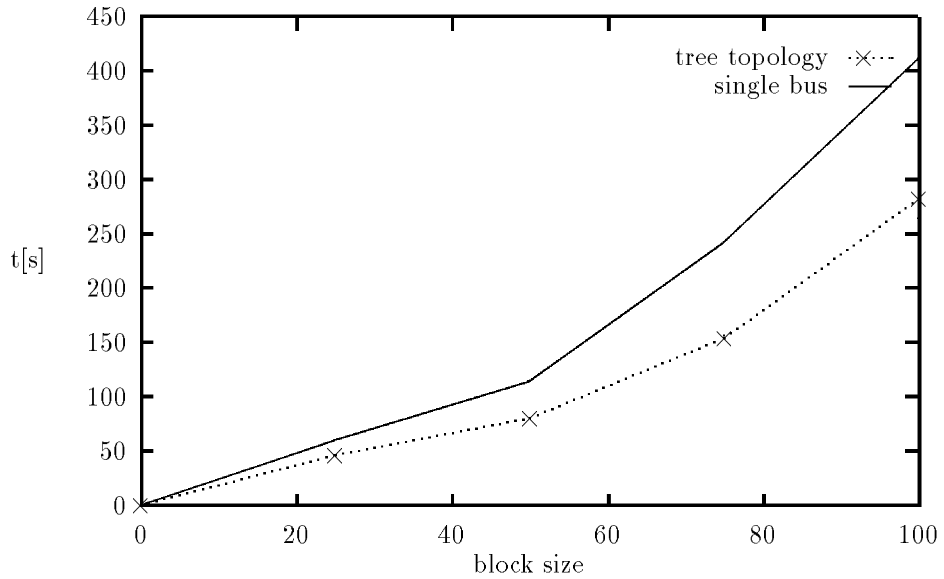

Figure 4: Runtime comparison of a distributed block matrix solver on the single bus and on the hierarchical tree network.

The matrix is split into different block rows that are distributed among the processors. Eliminated rows must be transferred to those processors who need the specific row for elimination of one of its own rows. A detailed discussion of the numerical problem and the algorithm can be found in [10, 9, 11].

Figure 4 shows the runtimes on the two topologies for different block sizes. Again, the multiple bus system is significantly faster than the single bus. Nevertheless, the introduced tree topology is not very convenient for unstructured dependence graphs. We will sketch out a better solution in the next section.

3 The Dual hypercube

The tree-like topology in the previous section is tailored to applications where the process graph is a complete binary tree that arises from the divide-and-conquer paradigm. However, for more general applications several inherent disadvantages and difficulties may arise.

If the dependence graph is less structured, e.g. even for divide-and-conquer with adaptivity, it is not clear how the processes should be mapped to the tree. Simple and convenient embedding schemes may result in a large amount of long-distance communication and an overload of the top level bus.
Another problem is fault tolerance. If workstation no. 9 in Figure 2 crashes, the half of the 16 computers is unreachable. A more symmetric topology with several communication paths between different nodes seems to be more comfortable.
A general topology would be the hypercube. But two main problems arise. First, the direct links between two nodes do not fit the bus-like nature of Ethernet. Secondly, we would need a logarithmic number of connections per workstation. In practice only a fixed number of connections per workstation is supported.
Both difficulties vanish if we exchange the roles of nodes and edges in the hypercube graph. Then, each edge represents a computing node with exactly two connections to buses, which are in their turn represented by the graph vertices. Because we changed the role of nodes and edges, we call this family of topologies *dual hypercube*. Figure 5 shows an example of dimension $d = 3$. For dimension d we get $d2^{d-1}$ nodes and 2^d buses. Thus, we have $d/2$ as many nodes as buses, enough to provide a logarithmic diameter d and high throughput of the network. Since d is only the logarithm of the graph size, the amount of communication in each bus is expected to increase slowly. Each bus is assigned to exactly d workstations and each node needs to have only two I/O-ports. Some other basic properties can be found in [1, 3] where similar types of graphs were introduced.
Due to its proximity to the hypercube and cube-connected-cycles, the dual hypercube seems to be well suited for a broad class of algorithms. If we want to compare the dual hypercube directly with the hierarchical topology of the previous section, we must restrict ourselves to dimensions $d = 2^n$ of the cube since only then the number of nodes is a power of 2 (namely 2^{n+2^n-1}). The number of buses then equals 2^{2^n} for the dual hypercube and 2^{n+2^n-2} for the tree. In [7] we discuss the embedding of binary trees and product graphs of binary trees in dual hypercubes.

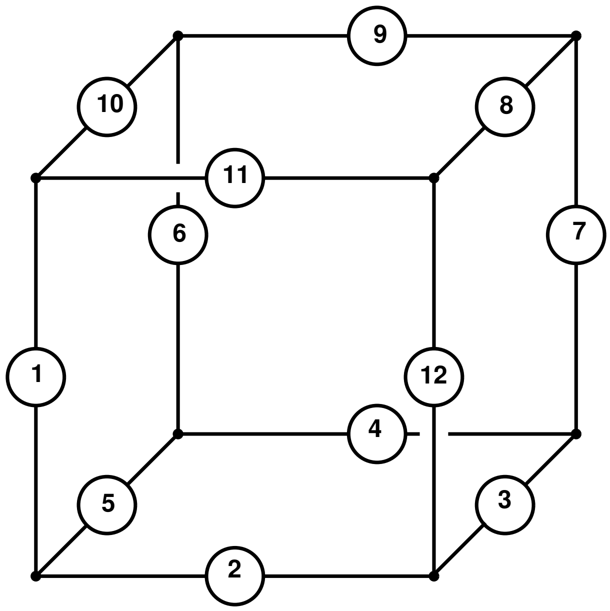

Figure 5: Dual hypercube of dimension 3.

3.1 Hardware Realization and Numerical Tests

An interesting relation between the hierarchical topology in Figure 2 and the dual hypercube is shown in Figure 6. It describes the way in which we built an extended 3-dimensional dual hypercube (which is actually a faulty 5-dimensional dual hypercube) involving the former tree network by simply adding four additional buses.

Using different routing mechanisms we could directly compare the performance of the two distinct topologies. While the routing on the tree could be done statically by standard UNIX software, it was a nontrivial issue to realize a reasonable routing scheme for the cube. Dynamic routers like *gated* do not allow to change the path from one node to another in short intervals (e.g. between two messages or packets). Parallel platforms like PVM don't support multiple paths between two workstations at all.

Therefore, a new routing daemon MRouter [4] (based on the TCP protocol) was developed, which runs a random path strategy: before each message is sent from one node to another the route is determined uniformly random among all optimal (i.e. shortest) paths.

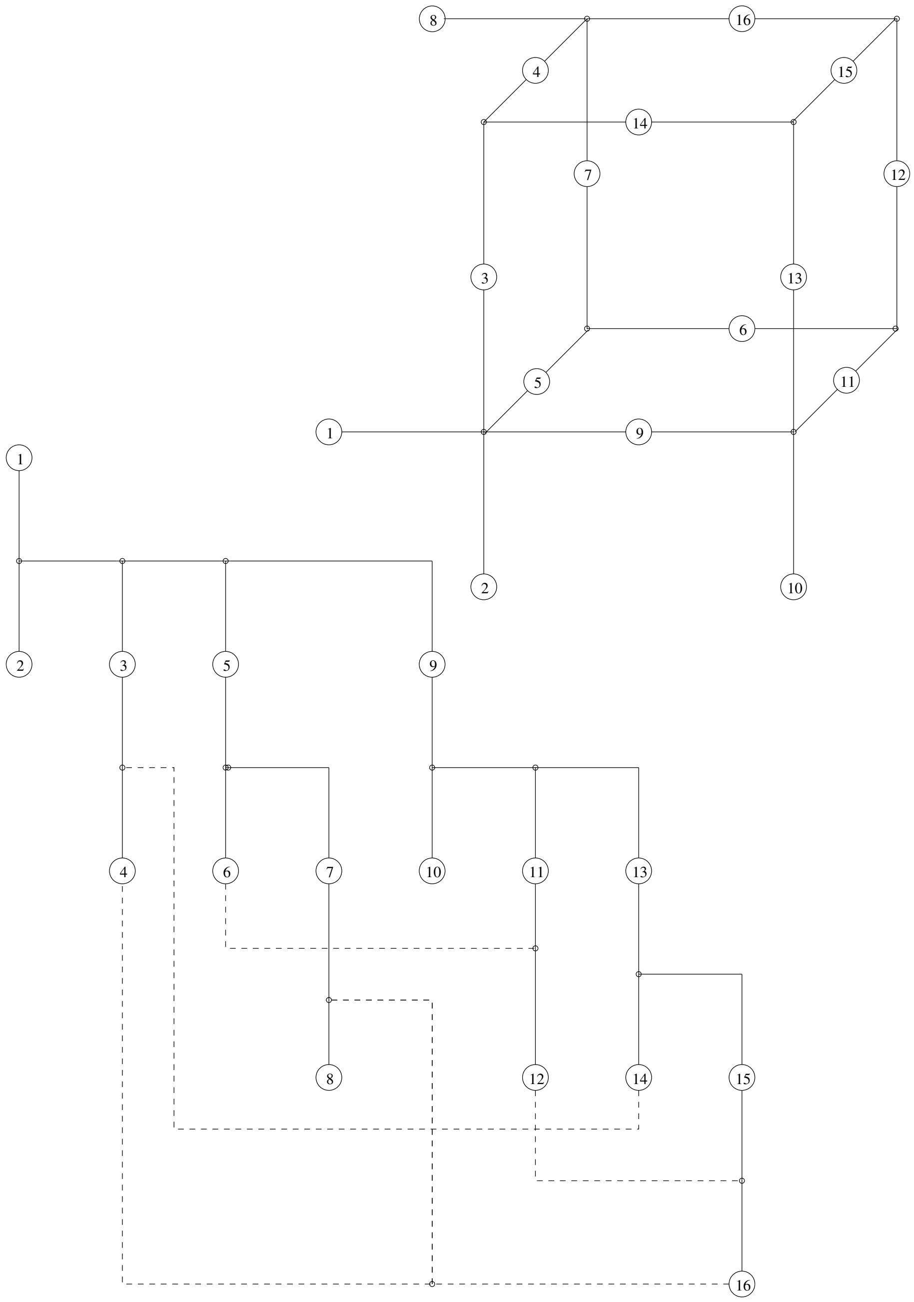

Figure 6: Embedding of the tree-like bus network with 16 nodes into the 5-dimensional dual hypercube.

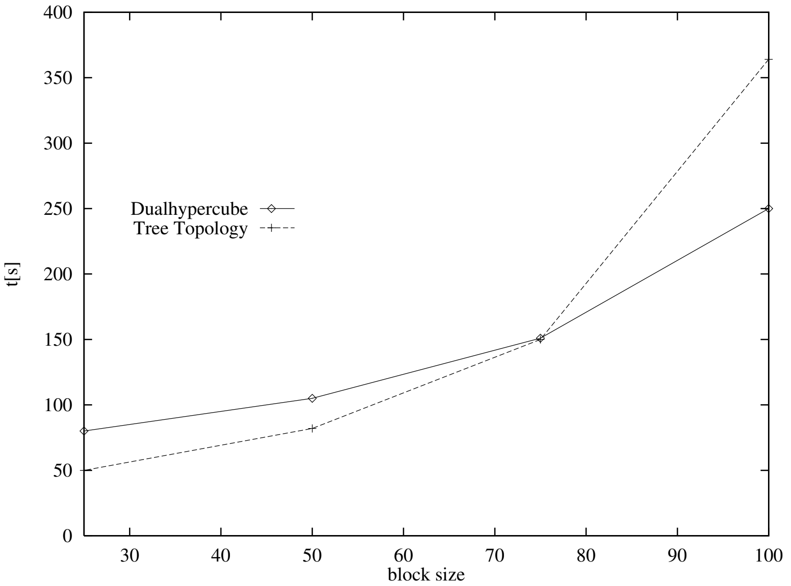

Figure 7: Runtimes of the sparse block matrix solver on the dual hypercube and the hierarchical topology.

We adapted the sparse block matrix solver mentioned in section 2 to MRouter and compared it to the PVM version on the tree-like network using the 12 HP-720 workstations that belong to the three-dimensional dual hypercube. The runtime behavior on the different topologies is shown in Figure 7.
At low block sizes (up to 75) the version on the tree-like topology is faster due to the minor overhead of the UDP protocol used by PVM. When the size of the matrix blocks grows, however, the hypercubic network is more scalable.

4 Simulation Results

In [2] M. Backschat describes the object oriented network simulator DAPOS++[1] that simulates in detail the communication of parallel applications on a complex workstation cluster. The topology of the cluster, the characteristics of the underlying hardware, the routing strategy, and the distributed algorithm

[1]Distributed-Algorithm-based Process-Oriented Simulation in C++

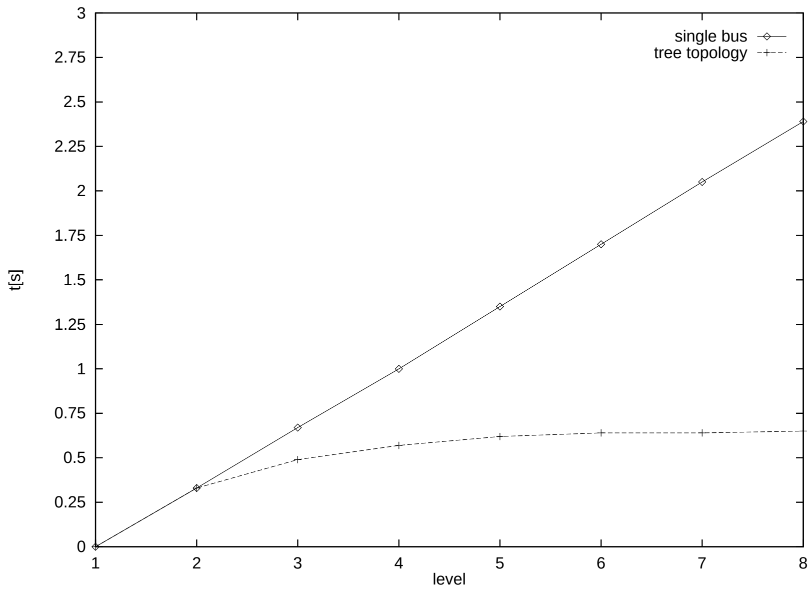

Figure 8: Simulated ARESO communication time on the shared bus and the tree topology.

can be described in a flexible manner by the user.
In contrast to many other products DAPOS++ does not use statistic data or models, but simulates the communication from application level down to packet transmission. Of course, we used statistic data for validation.
With DAPOS++ it was possible to study the divide-and-conquer communication scheme for various topologies and for large order of magnitude. In all examples that follow we choose the ARESO application introduced in section 2 as object of simulation.
In Figure 8 we compared the pure communication times for constant problem size, but increasing computing nodes (1 to 128). The time on the tree topology remains nearly constant whereas on the single bus the time grows linearly due to the increasing collision rates (in our simulation up to 90 percent). The Ethernet does not crash here even for 128 nodes because in the ARESO application the transmitted packets become very small when we use more and more workstations.
In order to include the dual hypercube network we have to focus on dimensions that are powers of 2 (as mentioned in section 3). In Figure 9

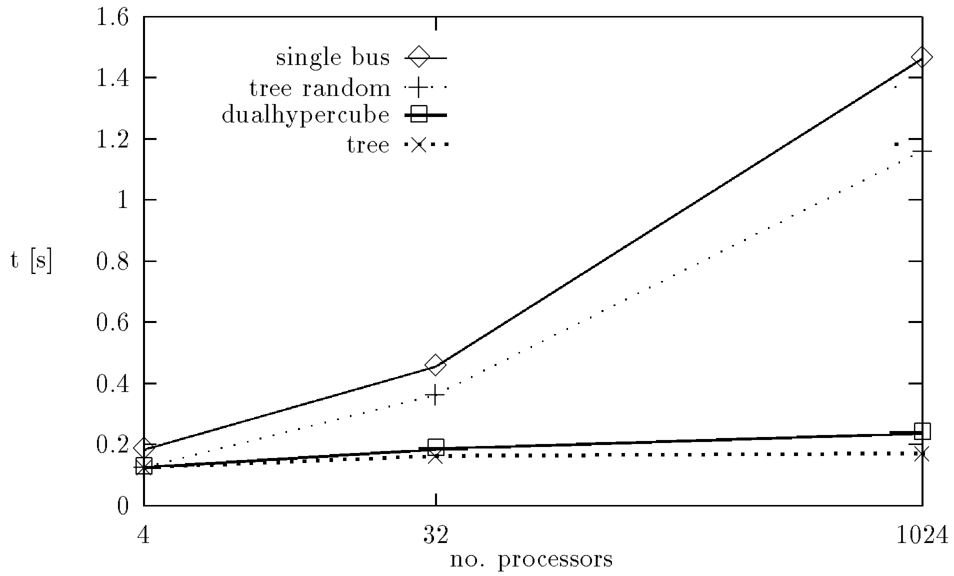

Figure 9: Simulated ARESO communication time on the shared bus, the tree topology, and the dual hypercube. On the tree-like network, we compare an optimal and a random mapping of the nodes.

we, therefore, simulated the communication time on 4, 32, and 1024 workstations — dimension 2, 4, and 8 of the dual hypercube — on the three different topologies. The problem size was divided by 2 in comparison with the former simulation.
As expected, the single bus showed the worst performance. But it is interesting that the dual hypercube performs almost as well as the corresponding tree network, though we chose the nodes on the cube randomly whereas the embedding onto the hierarchical network was optimal.
In the adaptive case, however, the dual hypercube turns out to be more flexible than the tree. As pointed out in section 2 the mapping of an unbalanced binary tree on the hierarchical network may lead to an inconvenient embedding. In this case the performance of the tree-like network deteriorates significantly. For demonstration we chose a random mapping of the complete squashed binary tree onto the 2^d nodes of the hierarchical network and compared its throughput against the optimal embedding. We integrated the corresponding results in Figure 9. The simulation clearly confirms the

superiority of the dual hypercube.

5 Conclusions

We presented two classes of bus topologies for workstation clusters. The first one, a hierarchical tree-like network, is shown to be best suited for divide-and-conquer parallelization. This theoretical consideration is confirmed by simulation results and real applications running on 16 HP-720 computers connected via Ethernet cables.
The second class of topologies, the dual hypercube, is better suited for more general algorithms requiring more flexibility. This includes adaptive divide-and-conquer applications and totally unstructured (random) communication dependencies. This is also illustrated with simulation results and actual runtimes of distributed numerical applications.

6 Acknoledgements

I would like to thank Prof. Chr. Zenger and Dr. Ulrich Rüde for many stimulating discussions. I am also grateful to Reiner Hüttl for providing plenty of test data concerning the ARESO project. I am thankful to Martin Backschat who helped me to prepare the simulator, and Andreas Paul who helped me to build and administer the network.

References

[1] D.P. Agrawal and V.K. Janakiram. Evaluating the Performance of Multicomputer Configurations. *Computers*, May 1986.

[2] Martin Backschat. Simulation von Divide&Conquer-Algorithmen auf verschiedenen Netztopologien in C++. *Technische Universität München, Fortgeschrittenpraktikum*, 1994.

[3] L.N. Bhuyan and D.P. Agrawal. Generalized Hypercube and Hyperbus Structures for a Computer Network. *IEEE Transactions on Computers*, April 1984.

[4] M. Eckart. Parallelisierung auf hypercubeartigen Workstationnetzen. Master's thesis, Technische Universität München, 1994.

[5] R. Hüttl and M. Schneider. Parallel Adaptive Numerical Simulation. SFB-Bericht 342/01/94 A, Technische Universität München, 1994.

[6] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures, Arrays, Trees, Hypercubes.* Morgan Kaufmann Publishers, San Mateo, 1992.

[7] A. Pfaffinger. Dual Hypercubes. *Technische Universität München, SFB-Bericht*, to appear.

[8] H. Regler and U. Rüde. Layout optimization with algebraic multigrid methods (AMG). In *Proceedings of the Sixth Copper Mountain Conference on Multigrid Methods, Copper Mountain, April 4-9, 1993*, Conference Publication. NASA, 1993.

[9] M. Schneider. *Verteilte adaptive numerische Simulation auf der Basis der Finite-Elemente-Methode.* PhD thesis, Technische Universität München, 1994.

[10] M. Schneider, U. Wever, and Q. Zheng. Solving large and sparse linear equations in analog circuit simulation on a cluster of workstations. *The Computer Journal*, 36(8):685–689, 1993.

[11] B. Weininger. Lösen großer, dünnbesetzter linearer Gleichungssysteme auf einem Netz von Workstations. Master's thesis, Technische Universität München, 1994.

SFB 342: Methoden und Werkzeuge für die Nutzung paralleler Rechnerarchitekturen

bisher erschienen :

Reihe A

342/1/90 A	Robert Gold, Walter Vogler: Quality Criteria for Partial Order Semantics of Place/Transition-Nets, Januar 1990
342/2/90 A	Reinhard Fößmeier: Die Rolle der Lastverteilung bei der numerischen Parallelprogrammierung, Februar 1990
342/3/90 A	Klaus-Jörn Lange, Peter Rossmanith: Two Results on Unambiguous Circuits, Februar 1990
342/4/90 A	Michael Griebel: Zur Lösung von Finite-Differenzen- und Finite-Element-Gleichungen mittels der Hierarchischen Transformations-Mehrgitter-Methode
342/5/90 A	Reinhold Letz, Johann Schumann, Stephan Bayerl, Wolfgang Bibel: SETHEO: A High-Performance Theorem Prover
342/6/90 A	Johann Schumann, Reinhold Letz: PARTHEO: A High Performance Parallel Theorem Prover
342/7/90 A	Johann Schumann, Norbert Trapp, Martin van der Koelen: SETHEO/PARTHEO Users Manual
342/8/90 A	Christian Suttner, Wolfgang Ertel: Using Connectionist Networks for Guiding the Search of a Theorem Prover
342/9/90 A	Hans-Jörg Beier, Thomas Bemmerl, Arndt Bode, Hubert Ertl, Olav Hansen, Josef Haunerdinger, Paul Hofstetter, Jaroslav Kremenek, Robert Lindhof, Thomas Ludwig, Peter Luksch, Thomas Treml: TOPSYS, Tools for Parallel Systems (Artikelsammlung)
342/10/90 A	Walter Vogler: Bisimulation and Action Refinement
342/11/90 A	Jörg Desel, Javier Esparza: Reachability in Reversible Free- Choice Systems
342/12/90 A	Rob van Glabbeek, Ursula Goltz: Equivalences and Refinement
342/13/90 A	Rob van Glabbeek: The Linear Time - Branching Time Spectrum
342/14/90 A	Johannes Bauer, Thomas Bemmerl, Thomas Treml: Leistungsanalyse von verteilten Beobachtungs- und Bewertungswerkzeugen
342/15/90 A	Peter Rossmanith: The Owner Concept for PRAMs
342/16/90 A	G. Böckle, S. Trosch: A Simulator for VLIW-Architectures
342/17/90 A	P. Slavkovsky, U. Rüde: Schnellere Berechnung klassischer Matrix-Multiplikationen
342/18/90 A	Christoph Zenger: SPARSE GRIDS

Reihe A

342/19/90 A	Michael Griebel, Michael Schneider, Christoph Zenger: A combination technique for the solution of sparse grid problems
342/20/90 A	Michael Griebel: A Parallelizable and Vectorizable Multi- Level-Algorithm on Sparse Grids
342/21/90 A	V. Diekert, E. Ochmanski, K. Reinhardt: On confluent semi-commutations-decidability and complexity results
342/22/90 A	Manfred Broy, Claus Dendorfer: Functional Modelling of Operating System Structures by Timed Higher Order Stream Processing Functions
342/23/90 A	Rob van Glabbeek, Ursula Goltz: A Deadlock-sensitive Congruence for Action Refinement
342/24/90 A	Manfred Broy: On the Design and Verification of a Simple Distributed Spanning Tree Algorithm
342/25/90 A	Thomas Bemmerl, Arndt Bode, Peter Braun, Olav Hansen, Peter Luksch, Roland Wismüller: TOPSYS - Tools for Parallel Systems (User's Overview and User's Manuals)
342/26/90 A	Thomas Bemmerl, Arndt Bode, Thomas Ludwig, Stefan Tritscher: MMK - Multiprocessor Multitasking Kernel (User's Guide and User's Reference Manual)
342/27/90 A	Wolfgang Ertel: Random Competition: A Simple, but Efficient Method for Parallelizing Inference Systems
342/28/90 A	Rob van Glabbeek, Frits Vaandrager: Modular Specification of Process Algebras
342/29/90 A	Rob van Glabbeek, Peter Weijland: Branching Time and Abstraction in Bisimulation Semantics
342/30/90 A	Michael Griebel: Parallel Multigrid Methods on Sparse Grids
342/31/90 A	Rolf Niedermeier, Peter Rossmanith: Unambiguous Simulations of Auxiliary Pushdown Automata and Circuits
342/32/90 A	Inga Niepel, Peter Rossmanith: Uniform Circuits and Exclusive Read PRAMs
342/33/90 A	Dr. Hermann Hellwagner: A Survey of Virtually Shared Memory Schemes
342/1/91 A	Walter Vogler: Is Partial Order Semantics Necessary for Action Refinement?
342/2/91 A	Manfred Broy, Frank Dederichs, Claus Dendorfer, Rainer Weber: Characterizing the Behaviour of Reactive Systems by Trace Sets
342/3/91 A	Ulrich Furbach, Christian Suttner, Bertram Fronhöfer: Massively Parallel Inference Systems
342/4/91 A	Rudolf Bayer: Non-deterministic Computing, Transactions and Recursive Atomicity
342/5/91 A	Robert Gold: Dataflow semantics for Petri nets

Reihe A

342/6/91 A	A. Heise; C. Dimitrovici: Transformation und Komposition von P/T-Netzen unter Erhaltung wesentlicher Eigenschaften
342/7/91 A	Walter Vogler: Asynchronous Communication of Petri Nets and the Refinement of Transitions
342/8/91 A	Walter Vogler: Generalized OM-Bisimulation
342/9/91 A	Christoph Zenger, Klaus Hallatschek: Fouriertransformation auf dünnen Gittern mit hierarchischen Basen
342/10/91 A	Erwin Loibl, Hans Obermaier, Markus Pawlowski: Towards Parallelism in a Relational Database System
342/11/91 A	Michael Werner: Implementierung von Algorithmen zur Kompaktifizierung von Programmen für VLIW-Architekturen
342/12/91 A	Reiner Müller: Implementierung von Algorithmen zur Optimierung von Schleifen mit Hilfe von Software-Pipelining Techniken
342/13/91 A	Sally Baker, Hans-Jörg Beier, Thomas Bemmerl, Arndt Bode, Hubert Ertl, Udo Graf, Olav Hansen, Josef Haunerdinger, Paul Hofstetter, Rainer Knödlseder, Jaroslav Kremenek, Siegfried Langenbuch, Robert Lindhof, Thomas Ludwig, Peter Luksch, Roy Milner, Bernhard Ries, Thomas Treml: TOPSYS - Tools for Parallel Systems (Artikelsammlung); 2., erweiterte Auflage
342/14/91 A	Michael Griebel: The combination technique for the sparse grid solution of PDE's on multiprocessor machines
342/15/91 A	Thomas F. Gritzner, Manfred Broy: A Link Between Process Algebras and Abstract Relation Algebras?
342/16/91 A	Thomas Bemmerl, Arndt Bode, Peter Braun, Olav Hansen, Thomas Treml, Roland Wismüller: The Design and Implementation of TOPSYS
342/17/91 A	Ulrich Furbach: Answers for disjunctive logic programs
342/18/91 A	Ulrich Furbach: Splitting as a source of parallelism in disjunctive logic programs
342/19/91 A	Gerhard W. Zumbusch: Adaptive parallele Multilevel-Methoden zur Lösung elliptischer Randwertprobleme
342/20/91 A	M. Jobmann, J. Schumann: Modelling and Performance Analysis of a Parallel Theorem Prover
342/21/91 A	Hans-Joachim Bungartz: An Adaptive Poisson Solver Using Hierarchical Bases and Sparse Grids
342/22/91 A	Wolfgang Ertel, Theodor Gemenis, Johann M. Ph. Schumann, Christian B. Suttner, Rainer Weber, Zongyan Qiu: Formalisms and Languages for Specifying Parallel Inference Systems
342/23/91 A	Astrid Kiehn: Local and Global Causes
342/24/91 A	Johann M.Ph. Schumann: Parallelization of Inference Systems by using an Abstract Machine

Reihe A

342/25/91 A Eike Jessen: Speedup Analysis by Hierarchical Load Decomposition

342/26/91 A Thomas F. Gritzner: A Simple Toy Example of a Distributed System: On the Design of a Connecting Switch

342/27/91 A Thomas Schnekenburger, Andreas Weininger, Michael Friedrich: Introduction to the Parallel and Distributed Programming Language ParMod-C

342/28/91 A Claus Dendorfer: Funktionale Modellierung eines Postsystems

342/29/91 A Michael Griebel: Multilevel algorithms considered as iterative methods on indefinite systems

342/30/91 A W. Reisig: Parallel Composition of Liveness

342/31/91 A Thomas Bemmerl, Christian Kasperbauer, Martin Mairandres, Bernhard Ries: Programming Tools for Distributed Multiprocessor Computing Environments

342/32/91 A Frank Leßke: On constructive specifications of abstract data types using temporal logic

342/1/92 A L. Kanal, C.B. Suttner (Editors): Informal Proceedings of the Workshop on Parallel Processing for AI

342/2/92 A Manfred Broy, Frank Dederichs, Claus Dendorfer, Max Fuchs, Thomas F. Gritzner, Rainer Weber: The Design of Distributed Systems - An Introduction to FOCUS

342/2-2/92 A Manfred Broy, Frank Dederichs, Claus Dendorfer, Max Fuchs, Thomas F. Gritzner, Rainer Weber: The Design of Distributed Systems - An Introduction to FOCUS - Revised Version (erschienen im Januar 1993)

342/3/92 A Manfred Broy, Frank Dederichs, Claus Dendorfer, Max Fuchs, Thomas F. Gritzner, Rainer Weber: Summary of Case Studies in FOCUS - a Design Method for Distributed Systems

342/4/92 A Claus Dendorfer, Rainer Weber: Development and Implementation of a Communication Protocol - An Exercise in FOCUS

342/5/92 A Michael Friedrich: Sprachmittel und Werkzeuge zur Unterstüt- zung paralleler und verteilter Programmierung

342/6/92 A Thomas F. Gritzner: The Action Graph Model as a Link between Abstract Relation Algebras and Process-Algebraic Specifications

342/7/92 A Sergei Gorlatch: Parallel Program Development for a Recursive Numerical Algorithm: a Case Study

342/8/92 A Henning Spruth, Georg Sigl, Frank Johannes: Parallel Algorithms for Slicing Based Final Placement

342/9/92 A Herbert Bauer, Christian Sporrer, Thomas Krodel: On Distributed Logic Simulation Using Time Warp

342/10/92 A H. Bungartz, M. Griebel, U. Rüde: Extrapolation, Combination and Sparse Grid Techniques for Elliptic Boundary Value Problems

Reihe A

342/11/92 A	M. Griebel, W. Huber, U. Rüde, T. Störtkuhl: The Combination Technique for Parallel Sparse-Grid-Preconditioning and -Solution of PDEs on Multiprocessor Machines and Workstation Networks
342/12/92 A	Rolf Niedermeier, Peter Rossmanith: Optimal Parallel Algorithms for Computing Recursively Defined Functions
342/13/92 A	Rainer Weber: Eine Methodik für die formale Anforderungsspezifkation verteilter Systeme
342/14/92 A	Michael Griebel: Grid– and point–oriented multilevel algorithms
342/15/92 A	M. Griebel, C. Zenger, S. Zimmer: Improved multilevel algorithms for full and sparse grid problems
342/16/92 A	J. Desel, D. Gomm, E. Kindler, B. Paech, R. Walter: Bausteine eines kompositionalen Beweiskalküls für netzmodellierte Systeme
342/17/92 A	Frank Dederichs: Transformation verteilter Systeme: Von applikativen zu prozeduralen Darstellungen
342/18/92 A	Andreas Listl, Markus Pawlowski: Parallel Cache Management of a RDBMS
342/19/92 A	Erwin Loibl, Markus Pawlowski, Christian Roth: PART: A Parallel Relational Toolbox as Basis for the Optimization and Interpretation of Parallel Queries
342/20/92 A	Jörg Desel, Wolfgang Reisig: The Synthesis Problem of Petri Nets
342/21/92 A	Robert Balder, Christoph Zenger: The d-dimensional Helmholtz equation on sparse Grids
342/22/92 A	Ilko Michler: Neuronale Netzwerk-Paradigmen zum Erlernen von Heuristiken
342/23/92 A	Wolfgang Reisig: Elements of a Temporal Logic. Coping with Concurrency
342/24/92 A	T. Störtkuhl, Chr. Zenger, S. Zimmer: An asymptotic solution for the singularity at the angular point of the lid driven cavity
342/25/92 A	Ekkart Kindler: Invariants, Compositionality and Substitution
342/26/92 A	Thomas Bonk, Ulrich Rüde: Performance Analysis and Optimization of Numerically Intensive Programs
342/1/93 A	M. Griebel, V. Thurner: The Efficient Solution of Fluid Dynamics Problems by the Combination Technique
342/2/93 A	Ketil Stølen, Frank Dederichs, Rainer Weber: Assumption / Commitment Rules for Networks of Asynchronously Communicating Agents
342/3/93 A	Thomas Schnekenburger: A Definition of Efficiency of Parallel Programs in Multi-Tasking Environments
342/4/93 A	Hans-Joachim Bungartz, Michael Griebel, Dierk Röschke, Christoph Zenger: A Proof of Convergence for the Combination Technique for the Laplace Equation Using Tools of Symbolic Computation

Reihe A

342/5/93 A	Manfred Kunde, Rolf Niedermeier, Peter Rossmanith: Faster Sorting and Routing on Grids with Diagonals
342/6/93 A	Michael Griebel, Peter Oswald: Remarks on the Abstract Theory of Additive and Multiplicative Schwarz Algorithms
342/7/93 A	Christian Sporrer, Herbert Bauer: Corolla Partitioning for Distributed Logic Simulation of VLSI Circuits
342/8/93 A	Herbert Bauer, Christian Sporrer: Reducing Rollback Overhead in Time-Warp Based Distributed Simulation with Optimized Incremental State Saving
342/9/93 A	Peter Slavkovsky: The Visibility Problem for Single-Valued Surface (z = f(x,y)): The Analysis and the Parallelization of Algorithms
342/10/93 A	Ulrich Rüde: Multilevel, Extrapolation, and Sparse Grid Methods
342/11/93 A	Hans Regler, Ulrich Rüde: Layout Optimization with Algebraic Multigrid Methods
342/12/93 A	Dieter Barnard, Angelika Mader: Model Checking for the Modal Mu-Calculus using Gauß Elimination
342/13/93 A	Christoph Pflaum, Ulrich Rüde: Gauß' Adaptive Relaxation for the Multilevel Solution of Partial Differential Equations on Sparse Grids
342/14/93 A	Christoph Pflaum: Convergence of the Combination Technique for the Finite Element Solution of Poisson's Equation
342/15/93 A	Michael Luby, Wolfgang Ertel: Optimal Parallelization of Las Vegas Algorithms
342/16/93 A	Hans-Joachim Bungartz, Michael Griebel, Dierk Röschke, Christoph Zenger: Pointwise Convergence of the Combination Technique for Laplace's Equation
342/17/93 A	Georg Stellner, Matthias Schumann, Stefan Lamberts, Thomas Ludwig, Arndt Bode, Martin Kiehl und Rainer Mehlhorn: Developing Multicomputer Applications on Networks of Workstations Using NXLib
342/18/93 A	Max Fuchs, Ketil Stølen: Development of a Distributed Min/Max Component
342/19/93 A	Johann K. Obermaier: Recovery and Transaction Management in Write-optimized Database Systems
342/20/93 A	Sergej Gorlatch: Deriving Efficient Parallel Programs by Systemating Coarsing Specification Parallelism
342/01/94 A	Reiner Hüttl, Michael Schneider: Parallel Adaptive Numerical Simulation
342/02/94 A	Henning Spruth, Frank Johannes: Parallel Routing of VLSI Circuits Based on Net Independency
342/03/94 A	Henning Spruth, Frank Johannes, Kurt Antreich: PHIroute: A Parallel Hierarchical Sea-of-Gates Router

Reihe A

342/04/94 A	Martin Kiehl, Rainer Mehlhorn, Matthias Schumann: Parallel Multiple Shooting for Optimal Control Problems Under NX/2
342/05/94 A	Christian Suttner, Christoph Goller, Peter Krauss, Klaus-Jörn Lange, Ludwig Thomas, Thomas Schnekenburger: Heuristic Optimization of Parallel Computations
342/06/94 A	Andreas Listl: Using Subpages for Cache Coherency Control in Parallel Database Systems
342/07/94 A	Manfred Broy, Ketil Stølen: Specification and Refinement of Finite Dataflow Networks - a Relational Approach
342/08/94 A	Katharina Spies: Funktionale Spezifikation eines Kommunikationsprotokolls
342/09/94 A	Peter A. Krauss: Applying a New Search Space Partitioning Method to Parallel Test Generation for Sequential Circuits
342/10/94 A	Manfred Broy: A Functional Rephrasing of the Assumption/Commitment Specification Style
342/11/94 A	Eckhardt Holz, Ketil Stølen: An Attempt to Embed a Restricted Version of SDL as a Target Language in Focus
342/12/94 A	Christoph Pflaum: A Multi-Level-Algorithm for the Finite-Element-Solution of General Second Order Elliptic Differential Equations on Adaptive Sparse Grids
342/13/94 A	Manfred Broy, Max Fuchs, Thomas F. Gritzner, Bernhard Schätz, Katharina Spies, Ketil Stølen: Summary of Case Studies in FOCUS - a Design Method for Distributed Systems
342/14/94 A	Maximilian Fuchs: Technologieabhängigkeit von Spezifikationen digitaler Hardware
342/15/94 A	M. Griebel, P. Oswald: Tensor Product Type Subspace Splittings And Multilevel Iterative Methods For Anisotropic Problems
342/16/94 A	Gheorghe Ştefănescu: Algebra of Flownomials
342/17/94 A	Ketil Stølen: A Refinement Relation Supporting the Transition from Unbounded to Bounded Communication Buffers
342/18/94 A	Michael Griebel, Tilman Neuhoeffer: A Domain-Oriented Multilevel Algorithm-Implementation and Parallelization
342/19/94 A	Michael Griebel, Walter Huber: Turbulence Simulation on Sparse Grids Using the Combination Method
342/20/94 A	Johann Schumann: Using the Theorem Prover SETHEO for verifying the development of a Communication Protocol in FOCUS - A Case Study -
342/01/95 A	Hans-Joachim Bungartz: Higher Order Finite Elements on Sparse Grids

SFB 342: Methoden und Werkzeuge für die Nutzung paralleler Rechnerarchitekturen

bisher erschienen :

Reihe B

342/1/90 B	Wolfgang Reisig: Petri Nets and Algebraic Specifications
342/2/90 B	Jörg Desel: On Abstraction of Nets
342/3/90 B	Jörg Desel: Reduction and Design of Well-behaved Free-choice Systems
342/4/90 B	Franz Abstreiter, Michael Friedrich, Hans-Jürgen Plewan: Das Werkzeug runtime zur Beobachtung verteilter und paralleler Programme
342/1/91 B	Barbara Paech1: Concurrency as a Modality
342/2/91 B	Birgit Kandler, Markus Pawlowski: SAM: Eine Sortier- Toolbox -Anwenderbeschreibung
342/3/91 B	Erwin Loibl, Hans Obermaier, Markus Pawlowski: 2. Workshop über Parallelisierung von Datenbanksystemen
342/4/91 B	Werner Pohlmann: A Limitation of Distributed Simulation Methods
342/5/91 B	Dominik Gomm, Ekkart Kindler: A Weakly Coherent Virtually Shared Memory Scheme: Formal Specification and Analysis
342/6/91 B	Dominik Gomm, Ekkart Kindler: Causality Based Specification and Correctness Proof of a Virtually Shared Memory Scheme
342/7/91 B	W. Reisig: Concurrent Temporal Logic
342/1/92 B	Malte Grosse, Christian B. Suttner: A Parallel Algorithm for Set-of-Support Christian B. Suttner: Parallel Computation of Multiple Sets-of-Support
342/2/92 B	Arndt Bode, Hartmut Wedekind: Parallelrechner: Theorie, Hardware, Software, Anwendungen
342/1/93 B	Max Fuchs: Funktionale Spezifikation einer Geschwindigkeitsregelung
342/2/93 B	Ekkart Kindler: Sicherheits- und Lebendigkeitseigenschaften: Ein Literaturüberblick
342/1/94 B	Andreas Listl; Thomas Schnekenburger; Michael Friedrich: Zum Entwurf eines Prototypen für MIDAS