WORKSHOP

ScalPerf'04: SCALABLE APPROACHES to HIGH PERFORMANCE and HIGH PRODUCTIVITY COMPUTING

September 19-23, 2004

Bertinoro International Center for Informatics
`http://www.cs.unibo.it/bici/`

ABSTRACTS

STAPL pContainers: Efficient, Scalable, Portable and Easy to Use Distributed Data Structures

Nancy Amato
Texas A&M University

The Standard Template Adaptive Paralel Library (STAPL) is a collection of generic data structures and algorithms that provides a high productivity parallel programming infrastructure with an approach that draws heavily in design from the C++ Standard Template Library (STL). By abstracting much of the complexities of parallelism from the end user, STAPL provides a high productivity platform by enabling the user to focus on algorithmic design instead of lower level parallel implementation issues.

One of the key components of STAPL is its pContainer infrastructure for distributed data structures. The pContainer offers the user a "shared object" view so that user code can be written as if for a single address space, while the pContainer itself is responsible for efficiently identifying and performing actions on remote elements. In this talk, we describe the design and basic features of the STAPL pContainer infrastructure including its methods for data storage, lookup, retrieval and redistribution. Currently, STAPL has pContainer counterparts of all STL containers and some additional data structures such as graphs. As time allows, we will describe how a pGraph is used in some important scientific applications and will present performance results on several platforms, including IBM's BlueGene/L.

Is there a General Purpose Approach to Latency Hiding?

Giangranco Bilardi
University of Padova

Since memory latency increases with system size, an important (unanswered) question is the feasibility of uniprocessor systems of arbitrary size that can execute any program in time proportional to the number of executed instructions.

In this talk, we explore the pipelining of memory hierarchies as an approach to hiding memory latencies. We propose hierarchical memory organizations that afford full pipelinability without

sacrifying the latency of individual accesses. We also explore processor organizations capable of efficiently driving the proposed memories.

In this context, it appears that different processor organizations may exhibit substantially different performance on different programs, raising the question whether there is one processor that is optimal for all programs, even to within a constant factor.

(The work reported in this talk is joint with K. Ekanadham and P. Pattnaik.)

Deployment of High Performance Computing: real solutions in the scale-up of applications

Sigismondo Boschi
CINECA

Our computing center owns two large HPC machines: an IBM SP Cluster 1600, with 512 processor Power4 and an IBM Linux Cluster 1350, made of 512 processors Xeon-Pentium IV.

With these two systems we tackle every kind of applications, that generally use only a part of each system: typical runs use up to 64, and rarely 128 processors. This is the typical use of HPC.

On the other hand it happens that some problems need extreme resources to be resolved, either in terms of CPUs or of memory of of disk space/bandwidth.

We will show the general approaches used to tackle this kind of problems, and what kind of technical solution are needed to solve a typical kind of problems, like the solution of very large equation systems (N=200000). Also the main differences between the two systems, in terms of capability to solve extreme problems, will be pointed out.

Performance Modeling of Extreme-Scale Parallel Systems and Applications

Adolfy Hoisie
Los Alamos National Laboratory

In this talk we will present novel techniques for performance modeling developed at Los Alamos, including a variety of applications of modeling.

We will describe novel methodologies for performance analysis, modeling and prediction applicable to extreme-scale parallel architectures and applications. We will present ongoing and planned future performance projects in PAL at Los Alamos National Laboratory, based on the new methods that can be applied to next generation parallel systems – 100 T-Ops and beyond.

In addition to the methodology for performance modeling, the applications of the performance models will be presented, including architectural design, workload characterization, application optimization and system diagnostics.

We will conclude by summarizing a number of factors that in our view will significantly impact the development and performance of future generation parallel systems.

FFTW: Towards a Minimal Compositional Framework for High-performance FFTs

Steven G. Johnson
Massachusetts Institute of Technology

The demands of scientific and engineering computation have continued to outpace the performance of computers, so that extracting every ounce of speed from the hardware continues to be desirable in many applications. However, modern architectures have made it increasingly difficult to perform this optimization: complicated cache and CPU pipeline behaviors mean that old heuristics, such as minimizing the number of operations performed, no longer even come close to the optimum. Moreover, thanks to the rapid range of change in the hardware arena, performance tuning of code is a never-ending process.

Instead, we believe that a more stable solution may be possible by changing the question: instead of asking what is the best algorithm, one should ask what is the smallest collection of simple algorithmic fragments whose composition spans the optimal algorithm on as many computer architectures as possible. We describe an implementation of this strategy—combining self-optimization, code generation, and recursive cache-oblivious algorithms—in the context of FFTW, our free and widely used fast Fourier transform (FFT) library (www.fftw.org).

We found that the key question was not so much what set of algorithmic steps to compose, but rather to identify what computational problem these steps should solve: an FFT works by decomposing a DFT into sub-problems, but in what form should these problems be described? One cannot implement algorithmic steps to solve a computational sub-problem that is not described, and conversely the problem should be simple enough that the corresponding algorithmic steps are simple and can be combined without restriction. By decomposing any DFT recursively in terms of a multi-dimensional ”loop” of multi-dimensional transforms and corresponding data strides, we found it possible to define a few algorithmic steps whose compositions span a surprisingly wide variety of FFT algorithms.

Utilizing Reconfigurable Computing Platforms for Computational Fluid Dynamics

Gerhard Lienhart
University of Mannheim

In this talk the approach of utilizing FPGA-based reconfigurable computing platforms for the acceleration of CFD algorithms will be presented. The talk will concentrate on the SPH method to simulate fluid dynamics which is widely-used for numerical astrophysical simulations. The capabilities and limits of the approach are presented and the future prospects will be outlined.

Hydra: the Distributed, FPGA Accelerated Chess Program

Ulf Lorenz
Paderborn University

In times when the FPGA technology has reached maturity such that complex designs are possible, the boarder between hard- and software vanishes. It is now possible to develop fine grained parallel applications without the long-lasting chip design cycles. Simultanously, by the help of message passing libraries like MPI it is easy to write coarse grained parallel applications. The chess program Hydra is a high level application which profits from both worlds.

When playing board games like chess, checkers, othello etc., computers use game tree search algorithms to evaluate a position. The most spectacular success of a game playing program so far, has been the victory of the chess machine 'Deep Blue' vs. G. Kasparov, the best human chess player in the world, at that time. The world has now reached the point that a chess program can not only win one match against a human top player, but they are just getting stronger than the best human players.

We describe the design philosophy, general architechture and performance of the chess program Hydra, which has won an International Grandmaster Tournament with seven wins and 4 draws. Hydra is split up in a hardware- and software engine. The hardware (indeed in FPGA simulated hardware) calculates the time critical part of the search tree near the leaves. The part near the root is done in software, where it is much easier to to develop sophisticated algorithms. The FPGA cards allows the implementation of sophisticated chess knowledge without decreasing the computational speed. Thus, Hydra follows its famous predecessors Belle and Deep Blue.

Special Purpose vs. General Purpose: A case study with BlueGene/L

Jose E. Moreira
IBM Thomas J. Watson Research Center

BlueGene/L is a new massively parallel machine that will soon become the fastest computer in the world. The flagship BlueGene/L installation at Lawrence Livermore National Laboratory will have 65,536 dual-processor compute nodes and achieve 360 Tflops of peak computing power. BlueGene/L will also display unprecedent levels of performance density and performance/power efficiency. We will review what are the key technologies that have allowed these breakthroughs with BlueGene/L. Historically, BlueGene/L is an evolutionary architecture, descending from special purpose machines designed to solve QCD problems. We will discuss the architectural and software issues that make BlueGene/L a machine with broader applicability than its predecessors. We will also discuss what are the characteristics that make applications either suitable or nonsuitable for BlueGene/L.

On the Compilation, Debugging, and Parallelization of Recursive Divide and Conquer Algorithms

Alex Nicolau
University of California, Irvine

The development of divide and conquer (D&C) algorithms for matrix computations has led to the widespread use of high-performance scientific applications and libraries.

In turn, D&C algorithms can be implemented using loop nests or recursion. Recursion is extremely appealing because it is an intuitive means for the deployment of top-down techniques, which exploit data locality and parallelism naturally. However, recursion has been considered impractical for high-performance codes, mostly because of the inherent overhead of the division process into small subproblems.

In this work, we develop techniques to model the behavior of recursive algorithms in a way suitable for use by a compiler in estimating and reducing the division process overheads. We

describe these techniques and JuliusC, a (lite) C compiler, which we developed to exploit them. JuliusC unfolds the application call graph (partially) and extracts the relations among function calls. As a final result, it produces a directed acyclic graph (DAG) modeling the function calls concisely. The approach is a combination of compile-time and run-time analysis and both have negligible complexity.

We illustrate the applicability of our approach by studying 6 test cases. We present the analysis results and we show how our (optimizing) compiler can use these results to increase the efficiency of the division process between 14 to 20 million times, for our codes.

Overview of the HARWEST Project (HARdWare and software co-dEsign for computing sySTems)

Paolo Palazzari
ENEA - C.R. Casaccia

High Level Synthesis techniques, as well as hardware/software co-design, play a fundamental role in the development of heterogeneous computing systems, based both on commodity and dedicated devices. Dedicated devices are to be used to boost computationally expensive kernels, while commodity parts are in charge of executing the other portions of the code (control or I/O dominated). Referring to this topic, the talk will present the goals and the main ideas of the HARWEST project, a three years research project just launched and carried out by Ylichron, a spin-off company of ENEA, the Italian Agency for the New Technologies, the Energy and the Environment. The talk will review the mapping methodology which has been developed in ENEA: such a methodology allows to automatically produce the synthesizable VHDL representing a dedicated parallel architectures which supports algorithms expressed by means of the System of Affine Recurrence Equations (SARE) computational model. Based on such an High Level Synthesis background, the HARWEST project is aimed at broadening the set of computational problems to be dealt, using jointly both the SARE and the Control Data Flow Graphs models to express the computations. Starting from the specifics expressed by SARE+CDFG, through a "correct-by-construction" procedure, the parallel hardware architecture and the software will be automatically generated through an optimization process which will search for the (sub-) optimal HW/SW architecture supporting the original computation.

Challanges in Contemporary Processor Design

Pratap Pattnaik
IBM Thomas J. Watson Research Center

The prediction of Gordon Moore in 1965, only six years after the integrated circuits were invented, that the exponential growth in the number of transistors in an IC will continue, has been true for past four decades. This success in continued miniaturization in the transistor size has contributed to

- Significant increase in the processor clock speed,
- Enormous opportunity to integrated various functions on the same IC as the processor,

- Increase in the power density,
- Increase in the power consumption due to leakage currents,
- Opportunity for the Software to primarily focus on developer productivity, rather than extracting the performance from the hardware,
- A data centric society.

This talk will describe the design alternatives and challenges faced by todays micro architects in the areas of Cache Coherence protocols, micro partitioning technologies, CMP, SMT, Reliability and Serviceability and incorporation of the special purpose functions on the CPU IC. The talk will als ogive some perspective from our own experience with PowerPC systems over the past decade.

Bandwidth-Memory-Processing Tradeoffs for Lattice QCD Computations

Andrea Pietracaprina
University of Padova

ABSTRACT: We study of the resource requirements of key algorithms for Lattice Quantum Chromo Dynamics (LQCD) computations. The main goal is the identification of massively parallel and hierarchical machine structures optimized for the execution of these algorithms, with particular focus on the technological scenarios and the problem sizes likely to be tackled in the next 5-10 years.

CO-AUTHORS: Gianfranco Bilardi, Geppino Pucci, Raffaele Tripiccione.

Think Globally, Search Locally

Keshav Pingali
Cornell University

A major difference between compilers, and library generators such as ATLAS or FFTW is that compilers use simple architectural models to determine values for code optimization parameters such as tile sizes, whereas library generators use global search over the space of parameter values. Both ATLAS and FFTW produce better code than state-of-the-art compilers, and it is commonly believed that this is because the architectural abstractions used in compilers are too simplistic to permit accurate estimation of optimal values for code optimization parameters.

Recent work in the compiler community has shown that in fact, relatively simple models can be used to compute near-optimal values for these parameters. While this is adequate for most situations, a compiler that produces near-optimal code may not be good enough for generating libraries that must be highly tuned because they are used extensively by a large user community. In this paper, we argue that the right solution for this context is to use a combination of model refinement and local search. We demonstrate this by showing experimentally that a modified ATLAS system using model refinement and local search can produce BLAS code competitive with the code produced by the unmodified ATLAS system using global search.

Translating Submachine Locality into Locality of Reference

Geppino Pucci
University of Padova

In this talk, we provide evidence that the structured type of parallelism exposed by submachine locality can be automatically turned into locality of reference on arbitrarily deep hierarchies. Specifically, we develop efficient schemes to simulate parallel programs written for the Decomposable BSP (a BSP variant which captures submachine locality) on the sequential Hierarchical Memory Model (HMM), which rewards the exploitation of temporal locality, and on its extension with block transfer, the BT model, which also rewards the exploitation of spatial locality. The simulations yields good hierarchy-conscious sequential algorithms from parallel ones. We also discuss a generalization of the HMM result to the self-simulation of D-BSP augmented with hierarchical memory modules, which yields an interesting analogue of Brent's lemma, thus proving that the enhanced model features a seamless integration of memory and network hierarchies.

SPIRAL Code Generation for DSP Algorithms: Overview and Future Directions

Markus Püschel
Carnegie Mellon University

Fast changing, increasingly complex, and diverse computing platforms pose central problems in scientific computing: How to achieve, with reasonable effort, portable optimal performance? We present SPIRAL that considers this problem for the performance-critical domain of linear digital signal processing (DSP) transforms. For a specified transform, SPIRAL automatically generates high performance code that is tuned to the given platform. SPIRAL formulates the tuning as an optimization problem, and exploits the domain-specific mathematical structure of transform algorithms to implement a feedback-driven optimizer. Similar to a human expert, for a specified transform, SPIRAL "intelligently" generates and explores algorithmic and implementation choices to find the best match to the computer's microarchitecture. The "intelligence" is provided by search and learning techniques that exploit the structure of the algorithm and implementation space to guide the exploration and optimization. SPIRAL generates high performance code for a broad set of DSP transforms including the discrete Fourier transform, other trigonometric transforms, filter transforms, and discrete wavelet transforms. Experimental results show that the code generated by SPIRAL competes with, and sometimes outperforms, the best available human tuned transform library code.

This talk gives an overview of SPIRAL including current and future research directions.

SmartApps: Adaptive Applications for High Productivity / High Performance

Lawrence Rauchwerger
Texas A&M University

The performance of current parallel and distributed systems continues to disappoint. This is due partially to the inability of applications, compilers and computer systems to adapt to their own dynamic changes. It is also due to the current compartmentalized approach to optimization:

applications, compilers, OS and hardware configurations are designed and optimized in isolation. No global optimization is attempted and the needs of each running application does not constitute the primary optimization consideration.

To address these problem we propose application-centric computing, or Smart Applications (SmartApps). In the SmartApps executable, the compiler embeds most run-time system services, and a optimizing feedback loop that monitors the application's performance and adaptively reconfigures the application and the OS/system platform. At run-time, after incorporating the code's input and determining the system's resources and state, the SmartApps performs an instance specific optimization, which is more tractable than a global generic optimization between application, OS and system. The overriding philosophy of SmartApps is "measure, compare, and adapt if beneficial."

SmartApps is being developed in the STAPL infrastructure. STAPL (the Standard Template Adaptive Parallel Library) is a framework for developing highly-optimizable, adaptable, and portable parallel and distributed applications. It consists of a relatively new and still evolving collection of generic parallel algorithms and distributed containers and a run-time system (RTS) through which the application and compiler interact with the OS and hardware.

SmartApps are developed for several important computational science applications including a discrete-ordinates code simulating subatomic particle transport, a molecular dynamics code and a program using a novel method to simulate protein folding and ligand binding. We use current ASCI shared and distributed machines, networks of workstations that run under Linux and K42 operating systems and, as of late, the Blue Gene machine.

Experiences with the Design of Cache-Aware Numerical Algorithms

Ulrich Rüde
University of Erlangen-Nuernberg

We will present strategies and techniques for tuning the performance of numerical algorithms for cache-based systems. One example will be the multigrid method which is an asymptotically optimal method for solving sparse linear systems arising from the discretization of partial differential equations. The second example will be the Lattice Boltzmann method, an algorithm based on cellular automata for simulating fluid flow. Both classes of applications are fundamentally limited by bandwidth and latency restrictions of modern computer architectures. One of the difficulties is the interference between the CPU micro architecture and the memory system design that makes it very difficult to predict the observed performance and consequently to design truly cache-oblivious algorithms in practice.

Dedicated computing for Lattice Chromo-Dynamics: State-of-the-Art and future Prospects

Raffaele Tripiccione
University of Ferrara

In this talk I briefly outline the basic numerical algorithms used in Lattice Quantum Chromodynamics (LQCD) and the corresponding computational requirements, stressing the specific features that make dedicated computing a viable and effective option for this field. I then describe several

LQCD-optimized architectures developed in the recent past, trying to identify the impact that these developments have had in the field. I then present a survey of the latest generation LQCD engines. The talk ends with an attempt to identify possible future lines of development for effective LQCD computing architectures in the near future.

X-Ray : Automatic Measurement of Hardware Parameters

Kamen Yotov
Cornell University

There is growing interest in autonomic, self-tuning software that can optimize itself automatically on new platforms without manual intervention. Optimization requires detailed knowledge of the target platform such as the latency and throughput of instructions, the numbers of registers, and the organization of the memory hierarchy. An autonomic system needs to determine this kind of platform-specific information on its own.

In this paper, we describe the design and implementation of X-Ray, which is a tool that autonomically measures a large number of such platform-specific parameters. For some of these parameters, we also describe novel algorithms, which are more robust than existing ones. X-Ray is written in C for maximum portability, and it is based on accurate timing of a number of carefully designed micro-benchmarks. A novel feature of X-Ray is that it is easily extensible because it provides simple infrastructure and a code generator that can be used to produce the large number of micro-benchmarks needed for such measurements.

There are very few existing tools that address this problem. Our experiments show that X-Ray produces more accurate and more complete results than any of them.