Journal of Physics: Conference Series **78** (2007) 012034 doi:10.1088/1742-6596/78/1/012034

SciDAC-2 Software Infrastructure for Lattice QCD

Bálint Joó

Jefferson Lab, 12000 Jefferson Avenue, Newport News, VA 23606, USA

E-mail: bjoo@jlab.org

Abstract. We present work carried out by the USQCD Collaboration on Software Infrastructure funded under SciDAC 2. We present successes of the software from the original SciDAC 1 project as well as ongoing and future work. We outline the various scientific collaborations SciDAC-2 has created.

1. Introduction

In this contribution we summarize the work our collaboration is carrying out in the Lattice Quantum Chromodynamics (QCD) Application area of SciDAC-2. The article is organized as follows: In section 2 we elaborate on the software infrastructure developed under the original SciDAC-1 project. In sections 3 and 4 we discuss how this software stack allowed us to effectively exploit significant new resources, and we elaborate on the ongoing work of re-optimizing the software for the Petascale era. In section 5 we outline some issues for exploiting multi-core architectures, while in section 6 we present recent result in our research effort into lattice algorithms. We close our article with discuss the potential use of workflow technologies in our computational infrastructure in section 7. We summarize and conclude in section 8.

2. Building on the Success of Software from SciDAC-1

In the SciDAC-1 project, a wealth of lattice QCD software has been developed [1]. This software; being easy to deploy and having reasonable efficiency out of the box; allowed the efficient exploitation of computing resources available to our community at the time of SciDAC-1, namely: the QCD clusters at the Jefferson Lab (JLab) and the Fermi National Accelerator Laboratory (FNAL), and the QCDOC supercomputer at the Brookhaven National Laboratory (BNL).

The crucial feature of the software stack was its layered design, which is illustrated in figure 1. The layers are as follows:

Level 1 - Message Passing and Linear Algebra: This layer consists of the QCD Message Passing (QMP) and QCD Linear Algebra (QLA) application programming interfaces (APIs). In particular, QMP is a lightweight API which encapsulates the communications needs of lattice QCD. It can be implemented efficiently, both in terms of the MPI Message Passing Interface standard, or directly over low level hardware drivers for individual architectures.

Level 2 - Data Parallel Layer: This layer presents a virtual data parallel computer to the user, on which applications may be built. The layer includes lattice wide arithmetic operations, shifts and global reductions through the QCD Data Parallel APIs (QDP/C and QDP++) as well as a standardized I/O interface in the QCD I/O (QIO) module.

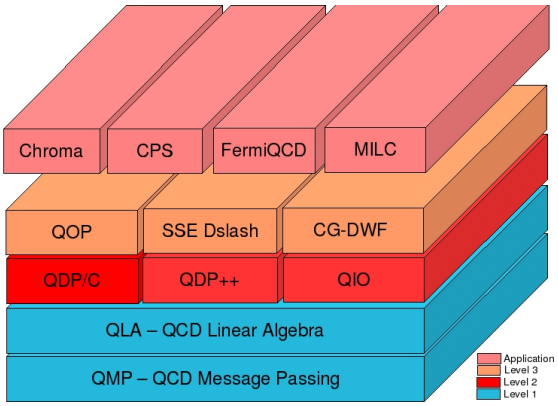

Figure 1. The SciDAC Software Layers

Level 3 - Optimized Components: Going through the lower 2 layers is not always efficient, and it is worthwhile to optimize some kernels in a way that cuts through one or more more layers. Examples of such kernels are linear operators, linear system solvers and molecular dynamics force terms. Level 3 provides a place for such components.

Application Layer: The modules in the lower layers need to be assembled into executable applications. There are several application suites available which have been built on, or retrofitted to use the SciDAC modules, such as: MILC, The Columbia Physics System, Chroma and FermiQCD (c.f. links from [1]). Individual researchers and smaller collaborations could either use these codes, or could build their own using the available software modules.

3. Enabling INCITE-ful Research

Apart from the successful exploitation of hardware dedicated to QCD the flexibility and good performance of the SciDAC modules, especially on SSE and PowerPC based architectures allowed us to make a successful bid under the INCITE program of the Department Of Energy (DOE). We are now performing production calculations on the Cray XT3/4 facility at Oak Ridge National Laboratory (ORNL). In figure 2 we show the performance of the Conjugate Gradients linear

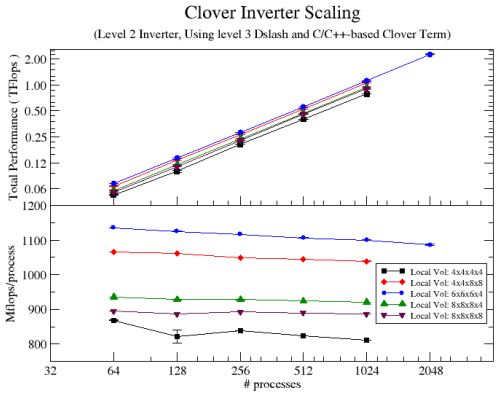

Figure 2. The Cray XT3/4 Machine at ORNL (left). The Scaling of the Chroma Conjugate Gradients solver on the Cray XT3 at ORNL (right)

system solver in the Chroma software system on the Cray XT3 architecture. We can see almost linear scaling in performance for up to 2048 cores.

4. Optimizing Not Just For the Petascale

While our applications perform reasonably well on new hardware, architecture specific optimizations can potentially yield even further benefits. This is particularly important to reach the petascale era at leadership computing facilities. Re-optimizing our key kernels for the SSE3 and SSE4 architectures should yield significant speed gains on Opteron and Intel processors. Opteron improvements would benefit performance on the Cray XT series of computers as well as Opteron based clusters that have been installed at the US Lattice QCD National Facility sites. Another target architecture for us is the BlueGene, especially its forthcoming generations. Our colleagues in USQCD are working with IBM to produce optimized QCD code for these machines.

A crucial ingredient of optimization is performance analysis, and this has resulted in a successful collaboration with the Renaissance Computing Institute (RENCI). In the initial phases of the work, RENCI targeted their performance analysis software on the MILC application suite, and identified several potential spots for optimization. Under SciDAC-2, they are looking at the Chroma code, and carrying out research in the performance analysis of C++ codes.

5. Threading our way towards Multi-core

Current trends in hardware point to multi-core systems as the place where performance growth is most likely to occur. Dual core hardware is already commonplace even in laptops. Several lattice QCD clusters already have dual core CPUs with some considering upgrades to 4 core CPUs. The Cray XT3/4 system at ORNL is currently comprised of dual core CPUs and is to be upgraded to 4 core CPUs in December 2007.

In terms of QCD code, the approach of running threaded code on a multi-core CPU, with message passing between such CPUs can significantly cut down on the memory copies one would face if one considered each core as a separate MPI (QMP) process communicating through memory. Some QCD codes are already capable of testing this approach, having originally been written for clustered symmetric multiprocessor (SMP) systems.

As part of our work in SciDAC-2 we have developed a lightweight threading library called QMT (QCD Multi-Threaded). This API allows the sites of a lattice to be divided between several threads. A novel feature of this library is its use of lightweight barrier synchronization protocols, especially for Opteron architectures.

There is, however, more to multi-core than just multi-threading. Performance can be impacted by several other aspects of multi-core architectures:

- How many cores can be kept productive before exhausting memory bandwidth?
- How to deal with processor, memory and cache affinity?
- Is one dealing with a flat or non uniform memory architecture (NUMA)?
- Does the hardware have a shared cache? Does it automatically enforce cache coherency? If so at what price?

Since one can envisage systems, with different answers to each of these questions it may be that designing a unified general multi-core API can become quite challenging.

6. Developing New Algorithms

A complementary activity to code optimization is the development of improved algorithms. The U.S. QCD community has a strong history of algorithmic development. Under SciDAC-2, several collaborations have been formed with other groups, such as numerical analysts and overseas lattice QCD researchers. USQCD collaborates with TOPS on applying multigrid algorithms to lattice QCD. A fruitful collaboration has developed between applied mathematicians in the

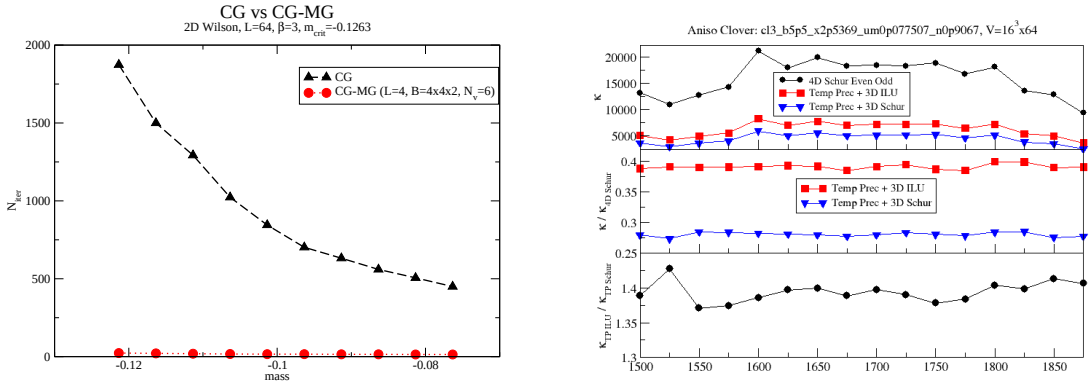

Figure 3. Multigrid Preconditioning eliminates critical slowing down in the Schwinger Model (left). Temporal preconditioning improves the condition number of the lattice Dirac equation for an anisotropic lattice QCD action (right).

college of William and Mary on improved linear system and eigensystem solvers. A long term collaboration between USQCD and Trinity College in Dublin, Ireland, has resulted in a new preconditioning technique for anisotropic lattice actions.

Some preliminary results of some of our algorithmic research are illustrated in figure 3. In the left panel we show improvements brought to the solution of the lattice Dirac equation by multigrid preconditioning. The dark triangles show the number of iterations required to solve the equation using a conventional Conjugate Gradients algorithm, as a function of the mass of the quarks employed. As the mass gets smaller, the number of iterations can be seen to diverge, a phenomenon known as critical slowing down. With the multigrid preconditioning (red squares) this divergence is completely eliminated. These tests were carried out in the two dimensional Schwinger Model – a toy model that is often used as a testbed for lattice QCD algorithms. Work is currently underway to extend these results to full QCD.

In the right panel of figure 3 we show improvements brought to the solution of the lattice Dirac equation by the technique of temporal preconditioning. In the top panel, we show the condition number of the linear operator in the equation using conventional 4D Schur style even-odd preconditioning (black circles) and two different kinds of temporal preconditioning (TP). We can see a major improvement over the conventional 4D Schur case for both varieties of TP. The middle panel shows the ratios of our two candidate TP Schemes to the 4D Schur case. We can see that the TP systems have condition numbers that are between 27% and 39% of the conventionally preconditioned systems. This translates into over a factor of 2 in the number of iterations required to solve the equation and to an as yet unknown factor in the reduction of the fermion force in a Molecular Dynamics integration. We are currently working on integrating the temporal preconditioning technique into our molecular dynamics codes, to quantify this gain.

7. Improving Infrastructure - Managing Complexity

As the computational tasks of lattice QCD become increasingly elaborate, infrastructure improvements are required to manage this complexity. There are several areas, where complexity becomes limiting, e.g.:

Data generation and analysis: An analysis task may involve fetching a file from storage, queuing the computation, archiving the results. The task may depend on other previously computed results, and may serve as an input for downstream computations. Complexity

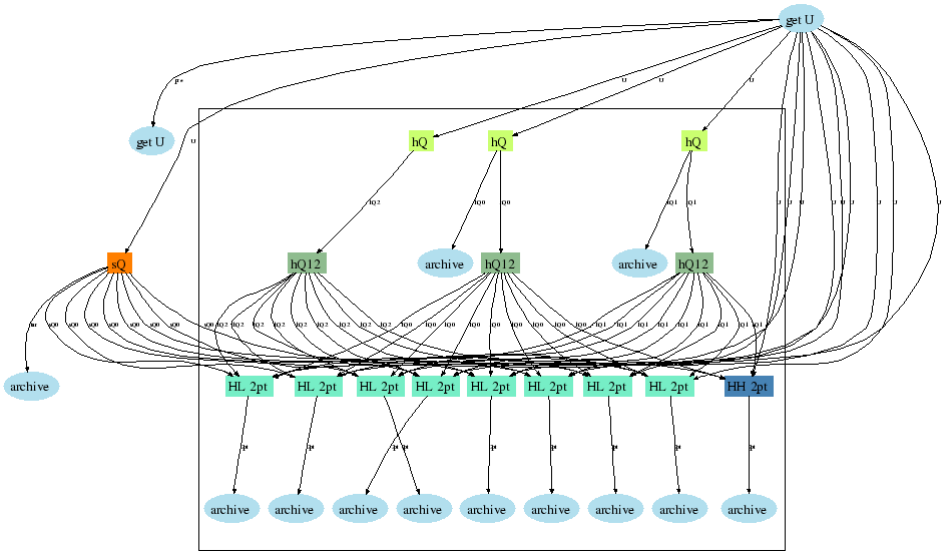

Figure 4. A workflow for computing heavy-light mesonic two point functions

arises in managing data latency from storage, queues, file transfers, task dependencies and the enormous amount of data to analyse.

Hardware Monitoring: With clusters of increasing size failures become more likely. Failures need to be detected and appropriate actions need to be taken. All actions should be logged. A particular challenge, is to ensure that healthy hardware is not slowed down by the monitoring process. Complexity can come from the types of failure, the actions taken, and their repercussions throughout the system.

Building Software: With layered software, whose components can come from various parts of the Internet, building an application becomes a nontrivial process. The software must be downloaded, configured, built and installed in an appropriate order for a platform. Different platforms may involve a different selection of modules. Complexity comes from the number of modules and the combinatorial number of ways in which they can be assembled.

The common feature of the above examples is that they involve a potentially complex network of interdependent tasks. Such processes are referred to as workflows and tools have recently been developed, especially in the Grid community, to automate their execution (Kepler, SWIFT, Triana etc). Under SciDAC-2, USQCD researchers are collaborating with workflow tool developers, to enable the leveraging of these tools in our computational infrastructure and to tame the ever increasing complexity.

8. Summary and Conclusions

The great success of the software infrastructure in SciDAC-1, has enabled the effective exploitation of the hardware of the US National Lattice Facility centers, as well as targets of opportunity such as Cray XT3/4 and BlueGene systems. Under SciDAC-2 we continue to re target our software to meet new challenges in the field, such as optimizing for new architectures or having to deal with multi-core technologies. We continue to collaborate with diverse research communities both within and outside of SciDAC to further the development of algorithms and to improve our computational infrastructure.

References

[1] USQCD Web Site: `http://www.usqcd.org`