LARGE SCALE PROBLEM SOLVING USING AUTOMATIC CODE GENERATION AND DISTRIBUTED VISUALIZATION

G. Allen[12], W. Benger[1] , E. Bentivegna[1], P. Diener[13], J. Ge[1], A. Hutanu[12], Robert Kooima[1] , O. Korobkin[13], K. Liu[12] , C. Toole [12], R. Paruchuri[1], E. Schnetter[13], J. Tao[1], A. Yates[1]

{[1] Center for Computation and Technology, [2] Department of Computer Science, [3] Department of Physics and Astronomy} Louisiana State University

SCALABILITY CHALLENGES

Programming Productivity

The programming productivity has long been a concern in the computational science community. In addition to possible human errors, the limit for code writing and the ever-growing complexity of many scientific codes make the development and maintenance of many large scale scientific applications an intimidating task. In addressing these issues, we present our latest work on generic methods for generating code that solves a set of coupled nonlinear partial differential equations using the **Kranc** code generation package. Our work greatly benefits from the modular design of the **Cactus** framework.

Scalability to Large Number of Processors

The ever growing complexity in developing highly scalable and efficient parallel scientific applications always leaves a gap for many application developers to cross. We need a bridge, a computational infrastructure, which can not only hide the hardware complexity, but also provide a user friendly interface for scientific application developers to speed up scientific discoveries. We present a highly efficient computational infrastructure that is based on the **Cactus** framework and the **Carpet** adaptive mesh refinement library.

I/O Bandwidth

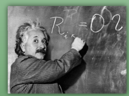
We are faced with difficult challenges in moving data when dealing with large datasets, challenges that arise from I/O architecture, network protocols and hardware resources. I/O architectures that do not use a non-blocking approach are fundamentally limiting the I/O performance Standard network protocols such as **TCP** cannot utilize the bandwidth available in emerging optical networks and cannot be used efficiently on wide-area networks. Single disks, or workstations are not able to saturate high-capacity network links. We build a system that combines an efficient pipeline-based architecture, can take advantage of non-standard high-speed data transport protocols such as **UDT** and use distributed grid resources to increase the I/O throughput.

Interactive Visualization of Large Data

Bringing efficient visualization and data analysis power to the end users' desktop while visualizing large data and maintain interactiveness, by as having the ability to control and steer the visualization by the user is a major challenge for visualization applications today. We are looking at the case where sufficiently powerful visualization resources are not available at either the location where the data was generated or at the location where the user is located, and we use visualization clusters in the network to interactively visualize large amounts of data.

SPECIFIC APPLICATION GRAVITATIONAL WAVE MODELING

Programming productivity Translating complex science into code




R Marenfeld, NOAO/AURA/NSF

Kranc
Automated code generation through Mathematica

Cactus
Portable, modular, extensible scientific software framework

Scalability to large number of processors Efficient utilization of high-concurrency machines

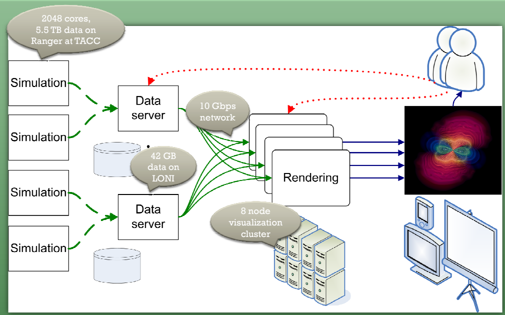
Cactus
Computational framework

Carpet
Adaptive mesh refinement

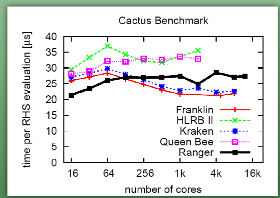
I/O Bandwidth and interactive visualization Moving and analyzing large datasets

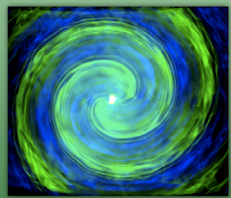
We carried out a demonstration run simulating the evolution of a binary black hole system for the Second **IEEE** International Scalable Computing Challenge (**SCALE 2009**) at **CCGrid** 2009.

The run used **2048 cores** on Ranger at TACC that generated **5.5 TB** data, where **42 GB** data was staged across the **LONI** distributed environment. The data was held in memory on two **LONI** clusters, and served across **10 Gpbs** networks to an 8 node visualization cluster at **LSU** where it was rendered and streamed using **SAGE** to the final display. The final visualization system achieved the goals of being interactive (**2** second response, **5** frames per second), collaborative (control via tangible interaction devices) and scalable.



Weak scaling benchmark results on different supercomputer systems and volume rendering of gravitational waves





CENTER FOR COMPUTATION & TECHNOLOGY
TeraGrid
TACC
www.loni.org
ALPACA
CyberTools
XiRel