Automating the Development of Parallel Multidisciplinary Scientific Applications

Jian Tao

Center for Computation & Technology
Louisiana State University
jtao@cct.lsu.edu

ALCF Seminar

June 11, 2009 ANL

CENTER FOR COMPUTATION & TECHNOLOGY

Gravitational Wave & γ-Ray Burst Modeling

- The modeling of gravitational waves and γ-ray bursts is among the most complex and challenging problems of astrophysics today.
- Einstein field equations (EFE) play a central role.
- Only numerical solutions to the EFE are possible in most realistic cases.

Grand Challenges in Computational Science

- Solving the EFE, as well as many other complex equations numerically requires powerful computers.
- We also need highly scalable and efficient parallel applications that can fully leverage the existing computational resources.
- Unfortunately, the ever growing system level complexity, as well as lines of code makes the development and maintenance of such applications more and more difficult.

Our Approach

We are building a problem solving environment that can

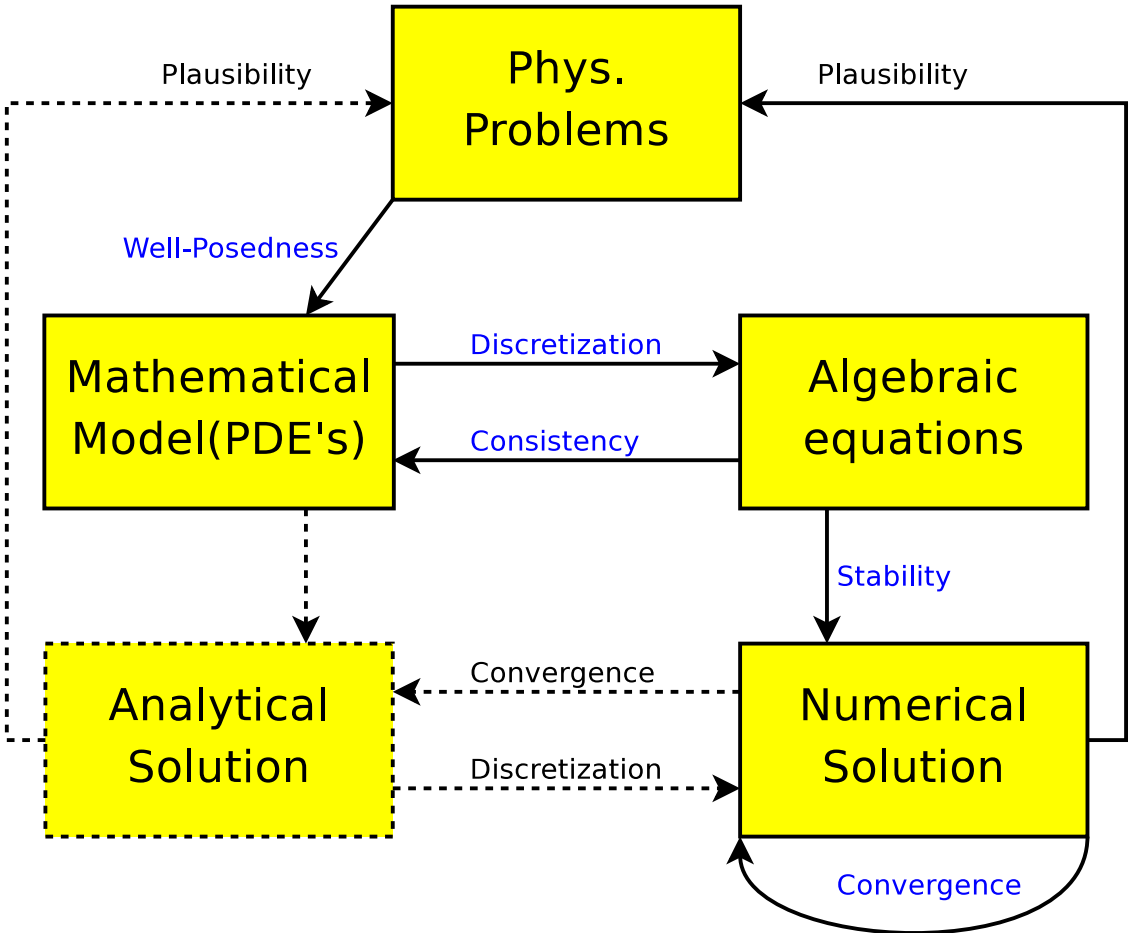
- hide the system level complexity and provide a user friendly interface to speed up large scale parallel code development via a computational infrastructure.
- hide the application level complexity and enable domain experts to build and verify scientific models easily via automatic code generation from a high level set of mathematical equations.

Our work is based on the Cactus-Carpet computational infrastructure and the Kranc code generation package.

CENTER FOR COMPUTATION & TECHNOLOGY

Ultimate Goal

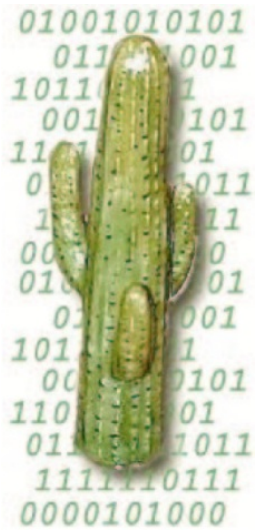
The ultimate goal is to automate the procedure to get the numerical solution to a scientific problem together with automatic convergence, consistence, and stability tests.



Cactus Computational Framework

Cactus is

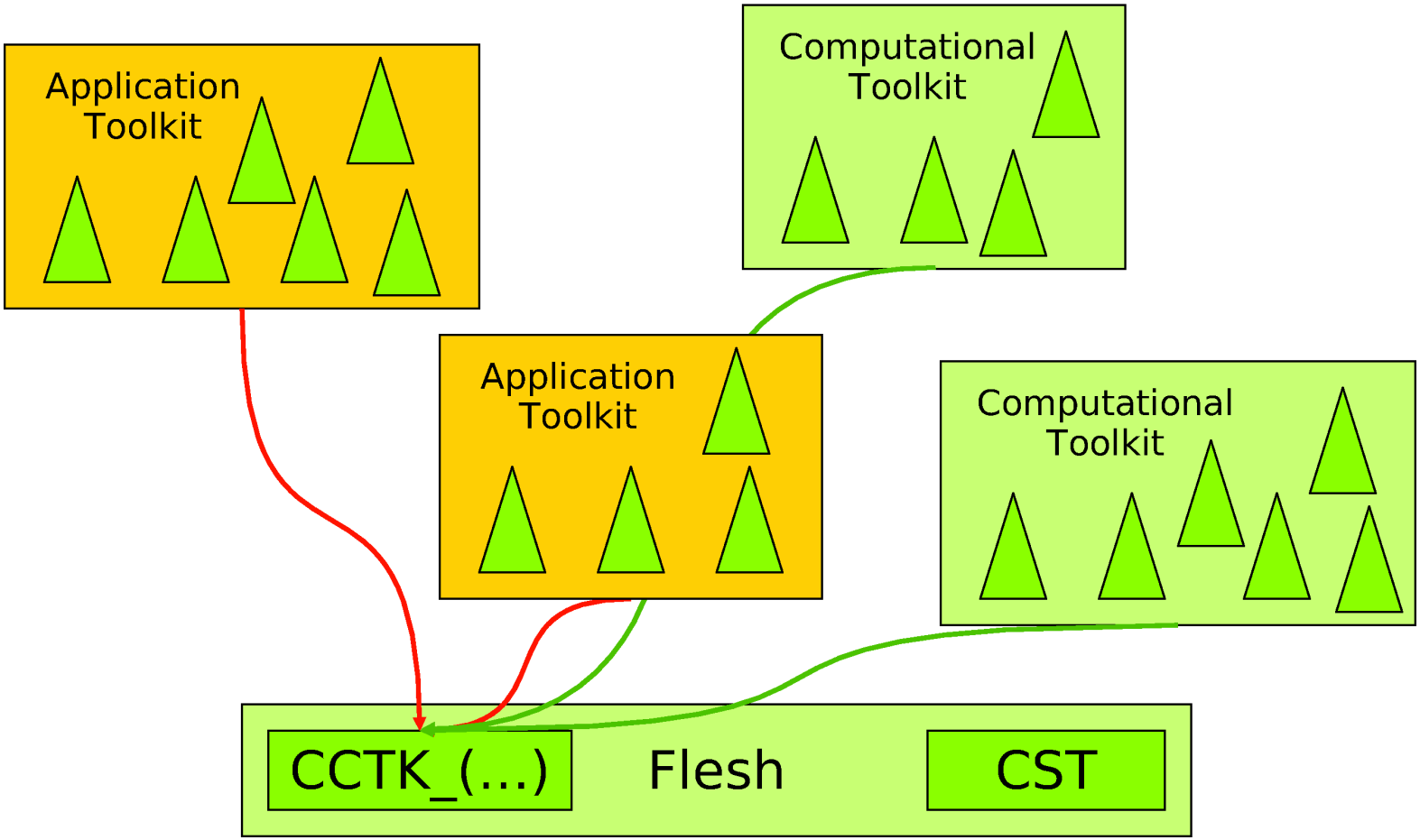
- a framework
  for developing portable, modular applications
  solving partial differential equations.
- focusing, although not exclusively,
  on high-performance simulation codes.
- designed to allow
  domain experts in one field to develop modules
  that are transparent to experts in other fields.



CENTER FOR COMPUTATION & TECHNOLOGY

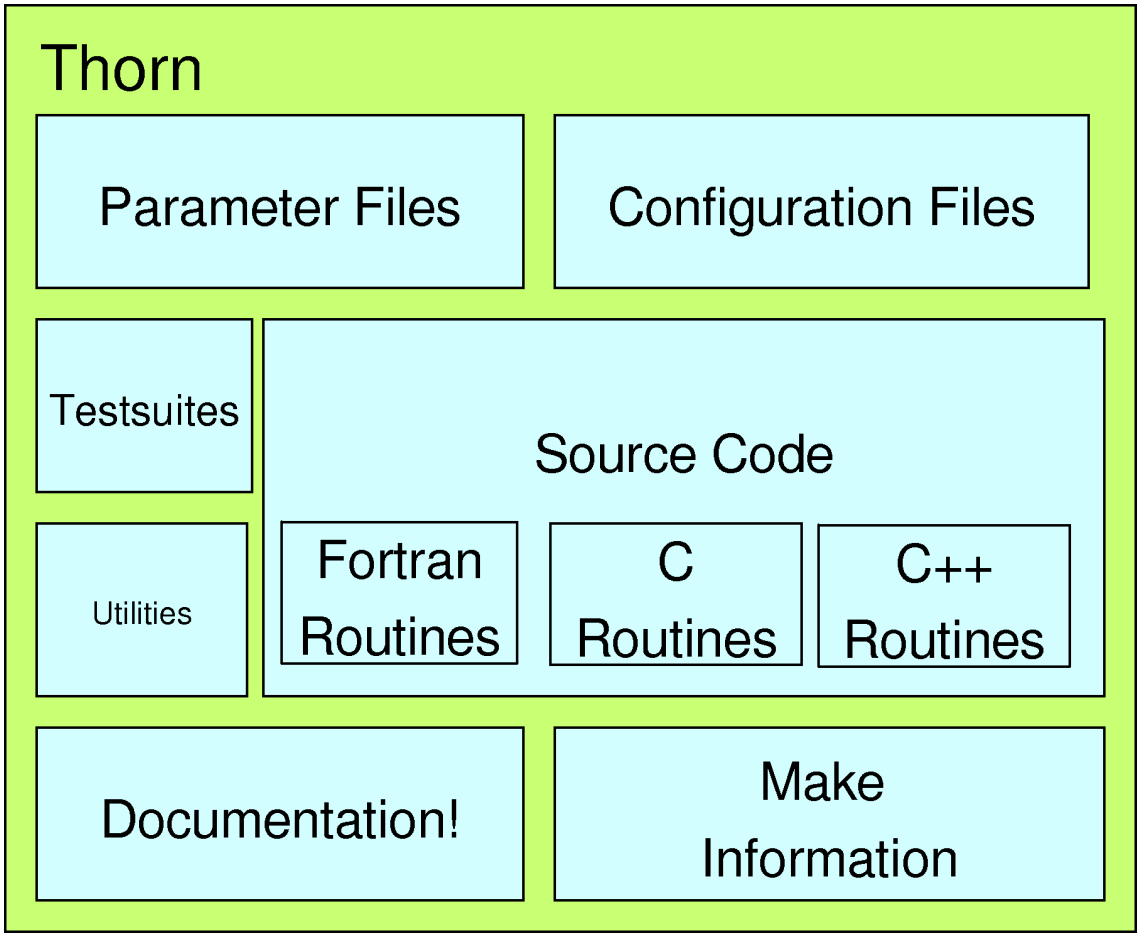
Application View

The structure of an application that is built upon the Cactus computational framework



Thorn Architecture

Inside view of a plug-in module, or thorn for Cactus



Philosophy & History

Philosophy

- Open code base to encourage community contributions
- Strict quality control for base framework
- Development always driven by real application developers
- Support and develop for a wide range of application domains

History

- 1995, Paul Walker, Joan Masso, Ed Seidel, and John Shalf: Cactus 1.0 for numerical relativity.
- In July 1999, Cactus 4.0 Beta 1 was release, mainly by Tom Goodale, Joan Masso, Gabrielle Allen, Gerd Lanfermann, and John Shalf.
- Cactus 4.0 Beta 16 was released on Feb 2nd, 2009.

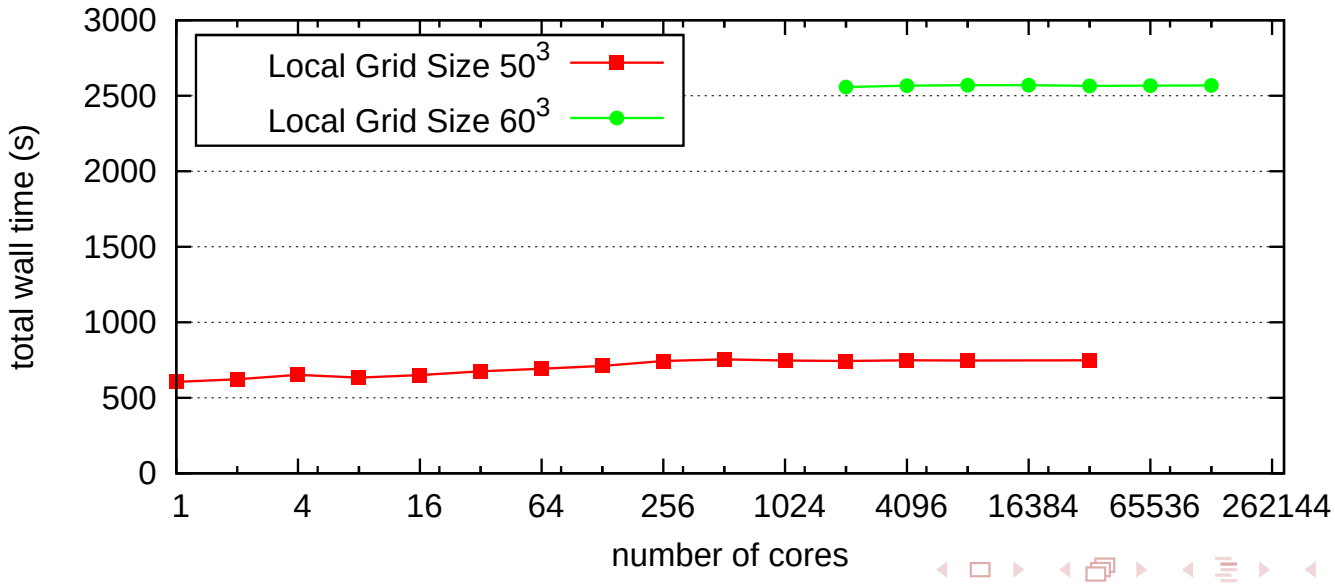
Portability & Scalability

Portability

- Cactus runs on all variants of the Unix operating system, Windows platform, Xbox, etc..

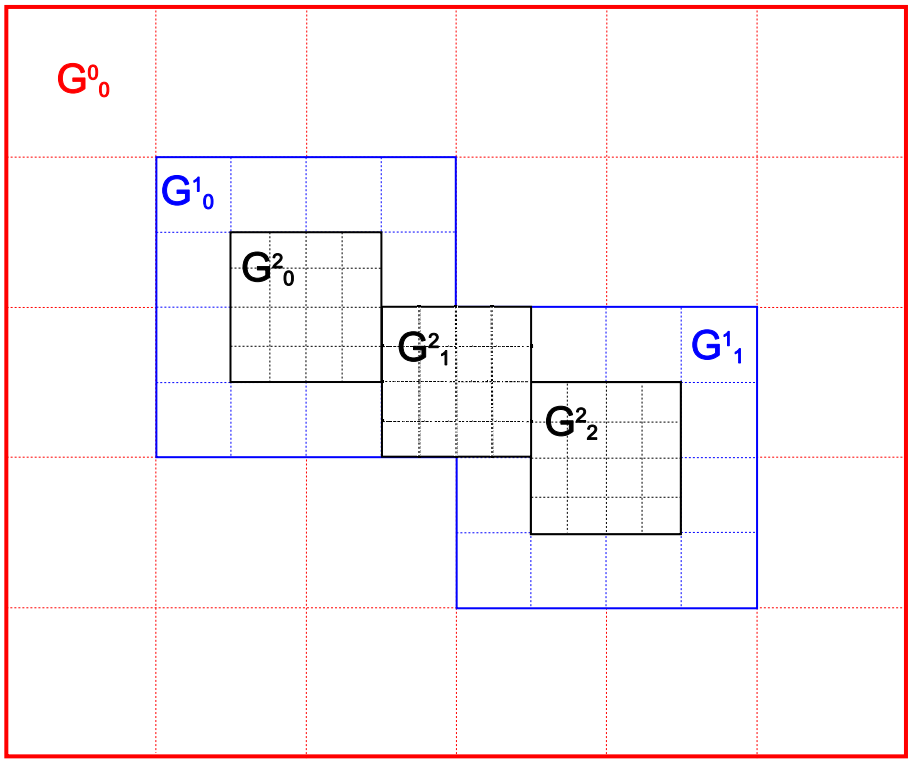
Scalability

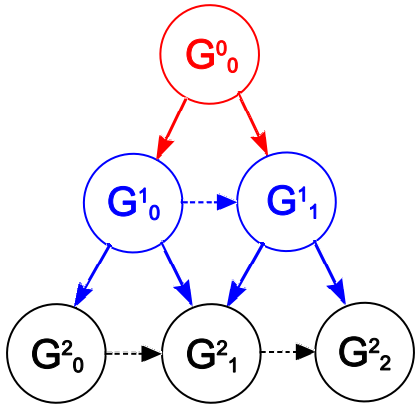
- Cactus scales to 131,072 out of 163,840 cores on Intrepid (Blue Gene/P) at ANL with a uniform grid driver.



Adaptive Mesh Refinement Method

The B&O AMR algorithm proposed by Berger and Oliger in 1984 is built upon a nested grid hierarchy of rectangular grids with increasing resolution.

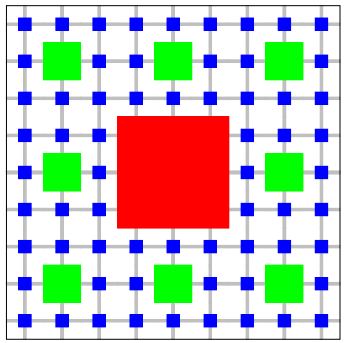




Carpet Adaptive Mesh Refinement Library

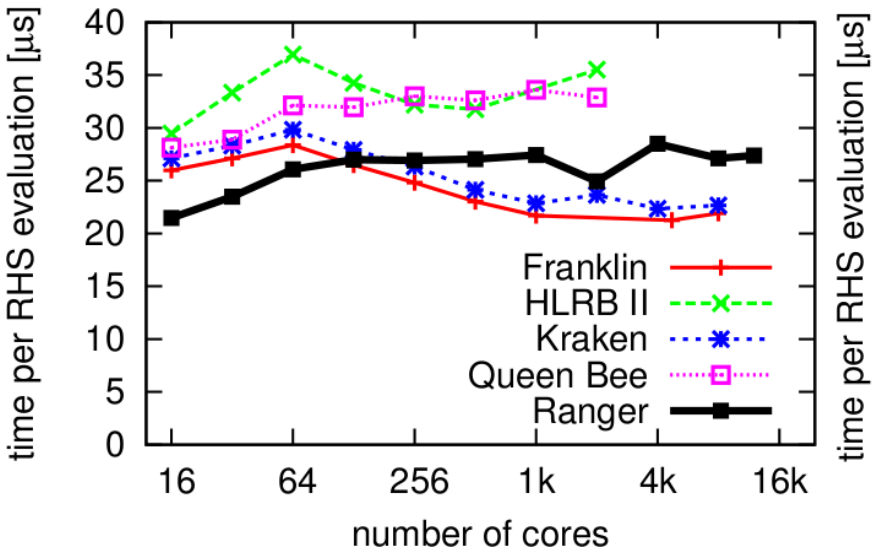
Carpet

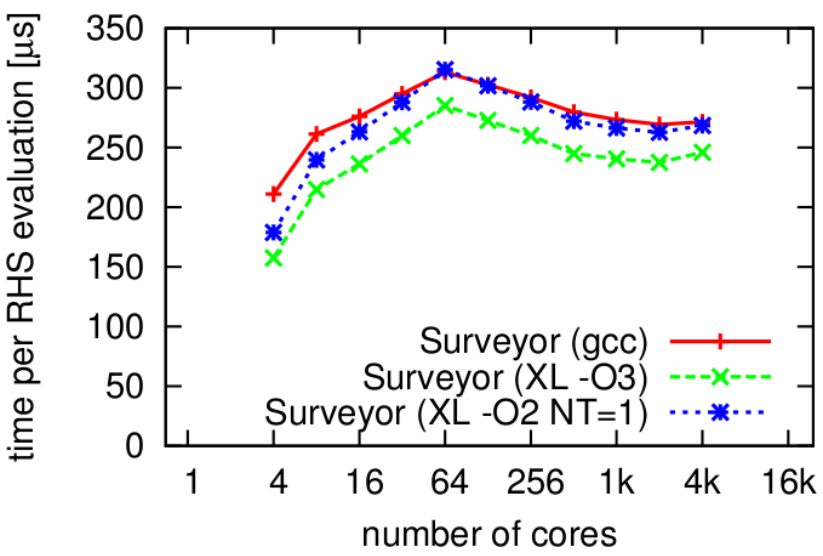
- is a driver layer of Cactus providing adaptive mesh refinement, multi-patch capability, as well as parallelization and efficient I/O.
- is written primarily in C++.
- was created in 2001 by Erik Schnetter at the Theoretische Astrophysik Tübingen.
- is currently maintained at the Center for Computation & Technology at LSU.



Scalability of Carpet

Carpet has been actively developed, and it is now used by several numerical relativity groups around the world for their production simulations of binary compact objects.





Hybrid Parallel Programming with LoopControl

LoopControl is a Cactus thorn that

- provides iterators over grid points.
- allows additional loop-level optimizations, such as cache-aware loop tiling or OpenMP parallelization on multicore machines.
- helps to minimize changes to the existing codes to support OpenMP.

Kranc Code Generation Package

Kranc is

- a suite of Mathematica packages with a computer algebra toolbox for numerical relativists.
- a prototyping system for physicists or mathematicians handling very complicated systems of partial differential equations.

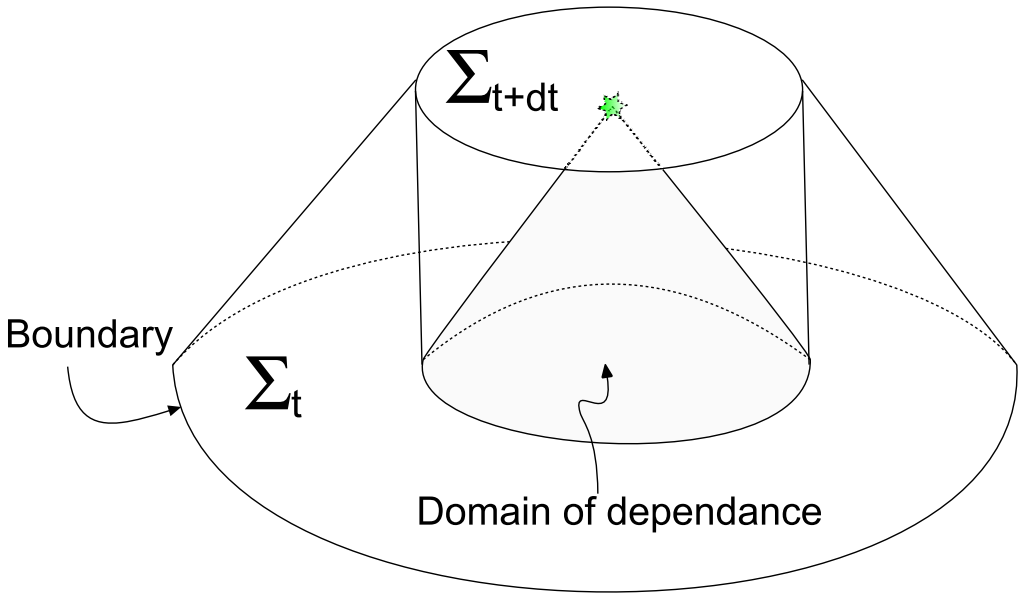
Kranc can generate entire Cactus based codes starting from a high level set of partial differential equations.

CENTER FOR COMPUTATION & TECHNOLOGY

Potential Applications

Various kinds of initial value boundary problems:

$$\begin{aligned} \partial_t u^a &= f(u^a), \\ u^a|_{t=0} &= g(u^a), \\ u^a|_{\partial\Sigma} &= h(u^a) \end{aligned}$$



Kranc in Action: Wave Equation (I)

Mathematical expression:

$$\partial_t^2 u = \delta^{ab} \partial_b \partial_a u$$

Rewritten in 1st order in time:

$$\begin{aligned} \partial_t u &= \rho, \\ \partial_t \rho &= \delta^{ab} \partial_b \partial_a u \end{aligned}$$

Input to Kranc:

$$\begin{aligned} \text{dot[u]} &\rightarrow \text{rho}, \\ \text{dot[rho]} &\rightarrow \text{KD[ua, ub]PD[u[la], lb]} \end{aligned}$$

CENTER FOR COMPUTATION & TECHNOLOGY

Kranc in Action: Wave Equation (II)

Generated C code to calculate the RHS:

```
/* Precompute derivatives (new style) */
   PDstandardNth11u = PDstandardNth11(u, i, j, k);
   PDstandardNth22u = PDstandardNth22(u, i, j, k);
   PDstandardNth33u = PDstandardNth33(u, i, j, k);
/* Calculate temporaries and grid functions */
   urhsL = rhoL;
   rhorhsL = PDstandardNth11u + PDstandardNth22u
           + PDstandardNth33u;
/* Copy local copies back to grid functions */
   rhorhs[index] = rhorhsL;
   urhs[index] = urhsL;
```

CENTER FOR COMPUTATION & TECHNOLOGY

McLachlan Code

The McLachlan code is

- created in the XiRel project to solve the EFE in vacuum, namely
$$G_{\mu\nu} \equiv R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R = 0.$$
- automatically generated using Kranc.
- coupled with LoopControl for hybrid MPI + OpenMP parallel programming.

$\tilde{R}_{ij}$ in McLachlan (I)

In the BSSN (Baumgarte-Shapiro-Shibata-Nakamura) $3+1$ formalism used in McLachlan, we need to calculate the 3D conformal part of Ricci tensor $\tilde{R}_{ij}$.

Mathematical expression:

$$\begin{aligned}\tilde{R}_{ij} &= -\frac{1}{2}\tilde{\gamma}^{lm}\tilde{\gamma}_{ij,lm} + \tilde{\gamma}_{k(i}\partial_{j)}\tilde{\Gamma}^{k} + \tilde{\Gamma}^{k}\tilde{\Gamma}_{(ij)k} \\ &+ \tilde{\gamma}^{lm}\left(2\tilde{\Gamma}^{k}_{l(i}\tilde{\Gamma}_{j)km} + \tilde{\Gamma}^{k}_{im}\tilde{\Gamma}_{klj}\right)\end{aligned}$$

CENTER FOR COMPUTATION & TECHNOLOGY

$\tilde{R}_{ij}$ in McLachlan (II)

Input to Kranc:

$$
\begin{aligned}
\text{Rt[li, lj]} \quad \rightarrow \quad & -(1/2)\text{gtu[ul, um]PD[gt[li, lj], ll, lm]} \\
+ \quad & (1/2)\text{gt[lk, li]PD[Xt[uk], lj]} \\
+ \quad & (1/2)\text{gt[lk, lj]PD[Xt[uk], li]} \\
+ \quad & (1/2)\text{Xtn[uk]gt[li, ln]Gt[un, lj, lk]} \\
+ \quad & (1/2)\text{Xtn[uk]gt[lj, ln]Gt[un, li, lk]} \\
+ \quad & \text{gtu[ul, um]}(+\text{Gt[uk, ll, li]gt[lj, ln]Gt[un, lk, lm]} \\
+ \quad & \text{Gt[uk, ll, lj]gt[li, ln]Gt[un, lk, lm]} \\
+ \quad & \text{Gt[uk, li, lm]gt[lk, ln]Gt[un, ll, lj]})
\end{aligned}
$$

CENTER FOR COMPUTATION & TECHNOLOGY

$\tilde{R}_{ij}$ in McLachlan (III)

Generated C code to calculate $\tilde{R}_{11}$ (5 more not shown)

```
Rt11  =  -(gtu11*khalf*PDstandardNth11gt11) + gtu21*
    (2*Gt211*Gt212*gt22L + 4*Gt112*gt13L*Gt311 + 2*Gt113*gt11L*Gt312 + 2*gt13L*Gt312*Gt313 + 2*gt13L*Gt211*Gt322 +
      2*gt13L*Gt311*Gt323 + 2*Gt311*Gt312*gt33L - PDstandardNth12gt11) - gtu31*PDstandardNth13gt11 +
   gt11L*PDstandardNth1Xt1 + gt12L*(4*Gt111*Gt212*gtu21 + 2*Gt211*Gt222*gtu21 + 2*Gt212*Gt222*gtu22 +
      4*Gt113*Gt211*gtu31 + 4*Gt113*Gt212*gtu32 + 4*Gt113*Gt213*gtu33 + PDstandardNth1Xt2) +
   gt13L*(4*Gt111*Gt312*gtu21 + 2*Gt212*Gt312*gtu21 + 4*Gt112*Gt312*gtu22 + 4*Gt113*Gt311*gtu31 +
      4*Gt113*Gt312*gtu32 + 4*Gt113*Gt313*gtu33 + PDstandardNth1Xt3) - gtu22*khalf*PDstandardNth22gt11 -
   gtu32*PDstandardNth23gt11 - gtu33*khalf*PDstandardNth33gt11 +
   Gt111*(6*Gt113*gt11L*gtu31 + 4*gt12L*Gt213*gtu31 + gt11L*Xtn1) +
   Gt211*(2*Gt112*gt11L*gtu11 + 4*Gt111*gt12L*gtu11 + 2*gt11L*Gt122*gtu21 + 2*gt11L*Gt123*gtu31 + gt12L*Xtn1) +
   Gt311*(4*Gt111*gt13L*gtu11 + 2*gt12L*Gt213*gtu11 + 2*gt13L*Gt313*gtu11 + 2*gt11L*Gt123*gtu21 +
      2*gt11L*Gt133*gtu31 + gt13L*Xtn1) + gt12L*Gt212*Xtn2 + gt13L*Gt312*Xtn2 +
   Gt112*(6*Gt111*gt11L*gtu21 + 4*gt12L*Gt211*gtu21 + 4*gt12L*Gt212*gtu22 + 2*gt11L*Gt213*gtu31 +
      6*Gt113*gt11L*gtu32 + gt11L*Xtn2) + Gt113*gt11L*Xtn3 +
   Gt213*(2*gt11L*Gt122*gtu32 + 4*Gt112*gt12L*gtu32 + 2*gt11L*Gt123*gtu33 + gt12L*Xtn3) +
   Gt313*(4*Gt111*gt13L*gtu31 + 2*gt12L*Gt213*gtu31 + 2*gt11L*Gt123*gtu32 + 4*Gt112*gt13L*gtu32 +
      2*gt12L*Gt223*gtu32 + 2*gt11L*Gt133*gtu33 + gt13L*Xtn3) + 3*gt11L*gtu11*SQR(Gt111) + 3*gt11L*gtu22*SQR(Gt112) +
   3*gt11L*gtu33*SQR(Gt113) + gt22L*gtu11*SQR(Gt211) + gt22L*gtu22*SQR(Gt212) +
   2*(gt12L*Gt211*Gt212*gtu11 + Gt113*gt11L*Gt311*gtu11 + Gt211*gt23L*Gt311*gtu11 + gt13L*Gt211*Gt312*gtu11 +
      Gt112*gt11L*Gt212*gtu21 + gt12L*Gt223*Gt311*gtu21 + Gt212*gt23L*Gt311*gtu21 + gt12L*Gt213*Gt312*gtu21 +
      Gt211*gt23L*Gt312*gtu21 + gt11L*Gt122*Gt212*gtu22 + gt11L*Gt123*Gt312*gtu22 + gt12L*Gt223*Gt312*gtu22 +
      Gt212*gt23L*Gt312*gtu22 + gt13L*Gt212*Gt322*gtu22 + gt13L*Gt312*Gt323*gtu22 + gt12L*Gt212*Gt213*gtu31 +
      gt12L*Gt211*Gt223*gtu31 + Gt211*Gt213*gt22L*gtu31 + gt12L*Gt233*Gt311*gtu31 + Gt213*gt23L*Gt311*gtu31 +
      gt13L*Gt213*Gt312*gtu31 + Gt113*gt11L*Gt313*gtu31 + Gt211*gt23L*Gt313*gtu31 + gt13L*Gt211*Gt323*gtu31 +
      gt13L*Gt311*Gt333*gtu31 + Gt311*Gt313*gt33L*gtu31 + gt11L*Gt123*Gt212*gtu32 + gt12L*Gt213*Gt222*gtu32 +
      gt12L*Gt212*Gt223*gtu32 + Gt212*Gt213*gt22L*gtu32 + gt11L*Gt133*Gt312*gtu32 + gt12L*Gt233*Gt312*gtu32 +
      Gt213*gt23L*Gt312*gtu32 + Gt212*gt23L*Gt313*gtu32 + gt13L*Gt213*Gt322*gtu32 + gt13L*Gt212*Gt323*gtu32 +
      gt13L*Gt313*Gt323*gtu32 + gt13L*Gt312*Gt333*gtu32 + Gt312*Gt313*gt33L*gtu32 + gt12L*Gt213*Gt223*gtu33 +
      gt12L*Gt233*Gt313*gtu33 + Gt213*gt23L*Gt313*gtu33 + gt13L*Gt213*Gt323*gtu33 + gt13L*Gt313*Gt333*gtu33 +
      gt12L*gtu21*SQR(Gt212)) + gt22L*gtu33*SQR(Gt213) + gt33L*gtu11*SQR(Gt311) + gt33L*gtu22*SQR(Gt312) +
   2*gt13L*gtu31*SQR(Gt313) + gt33L*gtu33*SQR(Gt313);
```

CENTER FOR COMPUTATION & TECHNOLOGY

McLachlan in Action

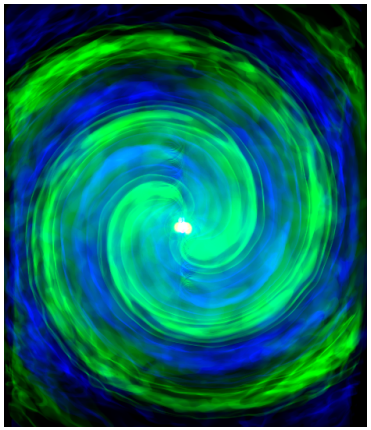
McLachlan is also

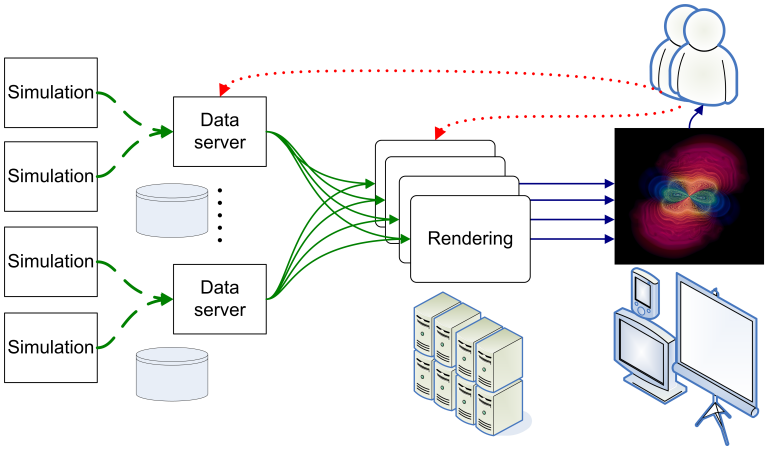
- the numerical kernel of the new Cactus benchmark submitted to the SPEC CPU Benchmark Search Program.
- adopted by the LSU team to compete in the Second IEEE International Scalable Computing Challenge (SCALE 2009) at CCGrid 2009.
- a Canadian musician, singer and songwriter.

CENTER FOR COMPUTATION & TECHNOLOGY

LSU IEEE SCALE 2009 Demo

LSU black hole demo using Cactus-Carpet computational infrastructure and the McLachlan code generated with Kranc won the first prize at the SCALE 2009 challenge at CCGrid09.





LSU
CENTER FOR COMPUTATION & TECHNOLOGY

Two Roads One Goal

Computer Science

- a toolkit in Kranc to provide a more user friendly interface
- modules in Cactus for supporting applications in other fields
- highly scalable and efficient infrastructural code

CENTER FOR COMPUTATION & TECHNOLOGY

Two Roads One Goal

Computer Science

- a toolkit in Kranc to provide a more user friendly interface
- modules in Cactus for supporting applications in other fields
- highly scalable and efficient infrastructural code

Physics

- a GR-Hydro code to solve the EFE with matter
- applications in fields other than numerical relativity

Two Roads One Goal

Computer Science

- a toolkit in Kranc to provide a more user friendly interface
- modules in Cactus for supporting applications in other fields
- highly scalable and efficient infrastructural code

Physics

- a GR-Hydro code to solve the EFE with matter
- applications in fields other than numerical relativity

The goal

- speeding up scientific discoveries with the aid of computers

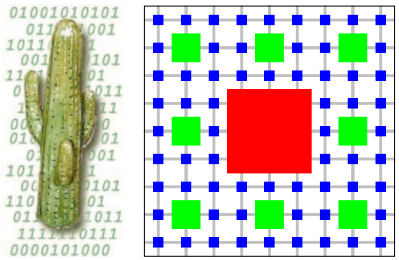
Acknowledgements

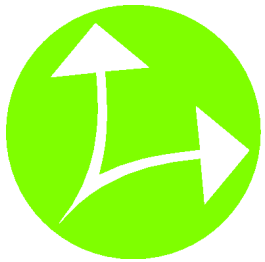
This work is supported by the XiRel project (NSF 0701566).

My thanks go to

- Gabrielle Allen, Eloisa Bentivegna, Peter Diener, Ian Hinder, Frank Löffler, and Erik Schnetter.
- all developers of Cactus, Carpet, Kranc, and many open source software projects.

We used the computational resources at LONI, Teragrid, and ALCF.






LSU
CENTER FOR COMPUTATION & TECHNOLOGY