Cactus: Advanced Topics

Kranc

Kranc

Kranc Assembles Numerical Code

- A tool for generating thorns
- Original version by S. Husa, I. Hinder, C. Lechner
- Used for many scientific papers
- Allows user to specify continuum problem without regard for implementation

Kranc

Kranc

Kranc Assembles Numerical Code

- A tool for generating thorns
- Original version by S. Husa, I. Hinder, C. Lechner
- Used for many scientific papers
- Allows user to specify continuum problem without regard for implementation

Kranc

Kranc

Kranc Assembles Numerical Code

- A tool for generating thorns
- Original version by S. Husa, I. Hinder, C. Lechner
- Used for many scientific papers
- Allows user to specify continuum problem without regard for implementation

Kranc

Kranc

Kranc Assembles Numerical Code

- A tool for generating thorns
- Original version by S. Husa, I. Hinder, C. Lechner
- Used for many scientific papers
- Allows user to specify continuum problem without regard for implementation

Kranc

Kranc

Kranc Assembles Numerical Code

- Minimally, a Kranc script contains
  - Cactus Group Definitions
  - A set of Calculations
- A calculation in Kranc is written using Mathematica (a symbolic programming language)
  - Example: dot[h[la,lb]] $-$ > -2 alpha K[la,lb]
  - dot means "time derivative"
  - h[la,lb] and K[la,lb] are matrices

Kranc

Kranc

Kranc Assembles Numerical Code

- Minimally, a Kranc script contains
  - Cactus Group Definitions
  - A set of Calculations
- A calculation in Kranc is written using Mathematica (a symbolic programming language)
  - Example: dot[h[la,lb]] $-$ $>$ -2 alpha K[la,lb]
  - dot means "time derivative"
  - h[la,lb] and K[la,lb] are matrices

Kranc

Kranc

Kranc Assembles Numerical Code

- Minimally, a Kranc script contains
  - Cactus Group Definitions
  - A set of Calculations
- A calculation in Kranc is written using Mathematica (a symbolic programming language)
  - Example: dot[h[la,lb]] $->$ -2 alpha K[la,lb]
  - dot means "time derivative"
  - h[la,lb] and K[la,lb] are matrices

Kranc

Kranc

Kranc Assembles Numerical Code

- Minimally, a Kranc script contains
  - Cactus Group Definitions
  - A set of Calculations
- A calculation in Kranc is written using Mathematica (a symbolic programming language)
  - Example: dot[h[la,lb]] $->$ -2 alpha K[la,lb]
  - dot means "time derivative"
  - h[la,lb] and K[la,lb] are matrices

Kranc

Kranc

Kranc Assembles Numerical Code

- Minimally, a Kranc script contains
  - Cactus Group Definitions
  - A set of Calculations
- A calculation in Kranc is written using Mathematica (a symbolic programming language)
  - Example: dot[h[la,lb]] $->$ -2 alpha K[la,lb]
  - dot means "time derivative"
  - h[la,lb] and K[la,lb] are matrices

Kranc

Kranc

Kranc Assembles Numerical Code

- Minimally, a Kranc script contains
  - Cactus Group Definitions
  - A set of Calculations
- A calculation in Kranc is written using Mathematica (a symbolic programming language)
  - Example: dot[h[la,lb]] $->$ -2 alpha K[la,lb]
  - dot means "time derivative"
  - h[la,lb] and K[la,lb] are matrices

Kranc

Kranc

Kranc Assembles Numerical Code

- Included in Cactus-OS distribution: `Cactus/repos/Kranc`.
- The state-of-the art numerical relativity code, produced by Kranc: `Cactus/arrangements/McLachlan`.
- For more, see: `http://kranccode.org`

Kranc

Kranc

Kranc Assembles Numerical Code

- Included in Cactus-OS distribution: `Cactus/repos/Kranc`.
- The state-of-the art numerical relativity code, produced by Kranc: `Cactus/arrangements/McLachlan`.
- For more, see: `http://kranccode.org`

Kranc

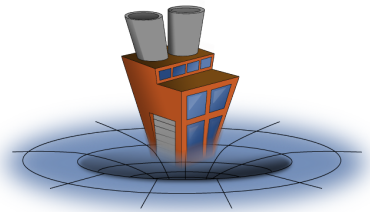
Kranc

Kranc Assembles Numerical Code

- Included in Cactus-OS distribution: `Cactus/repos/Kranc`.
- The state-of-the art numerical relativity code, produced by Kranc: `Cactus/arrangements/McLachlan`.
- For more, see: `http://kranccode.org`

Simfactory

The Simulation Factory:



- Command-line tools for setting up Cactus distribution and managing simulations on a variety of supercomputers, including most of the TeraGrid machines.
- Captures the best practices of experienced users, ensuring repeatable and well-documented scientific results.
- Developed by Erik Schnetter and Michael Thomas.
- Included in Cactus-OS distro: `Cactus/simfactory`.
- For more, see: `http://simfactory.org`

Einstein Toolkit

Open-source code for the numerial relativity and relativistic astrophysics.

Capabilities:

- Accurate evolutions of vacuum spacetimes using Einstein equations.
- Relativistic hydrodynamics, based on the public version of Whisky code.

- Initial data solvers: single and bindary BHs and relativistic stars.
- Analysis: finds BH horizon, calculates spacetime curvature, extracts gravitational waves.
- More information: `http://einsteintoolkit.org`

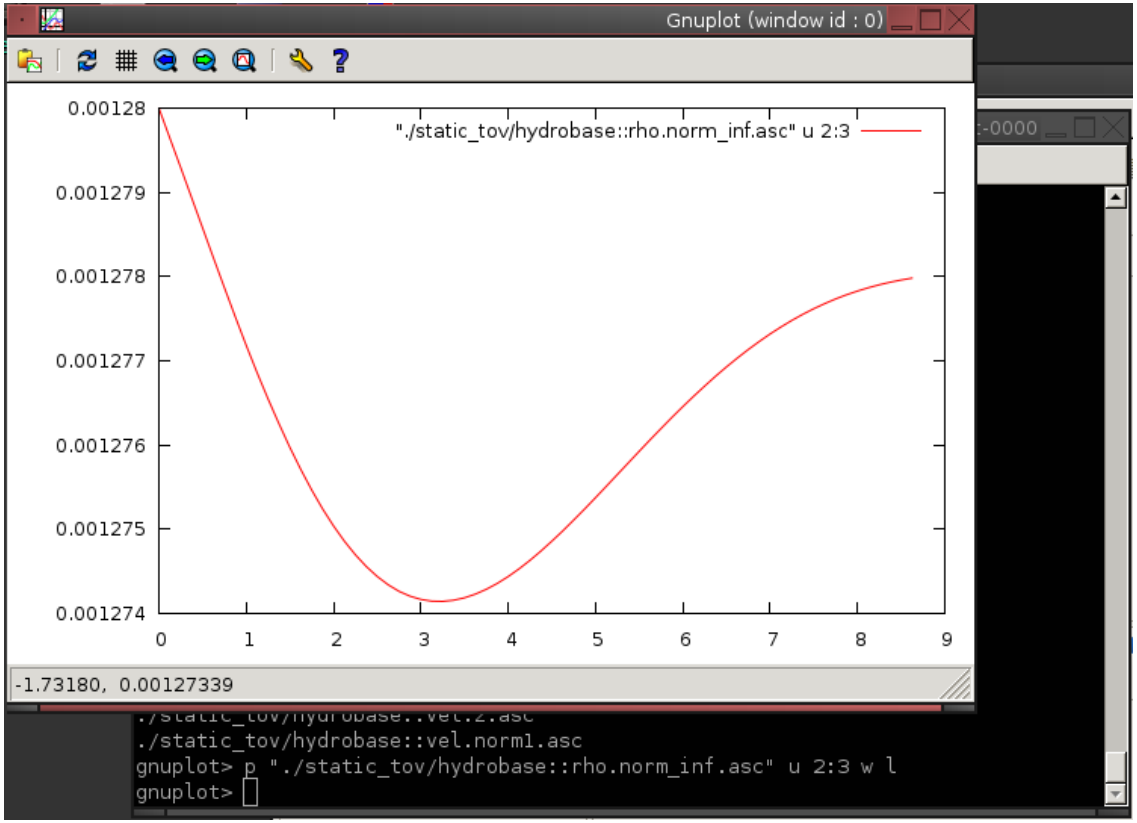
Einstein Toolkit on your virtual box

In Eclipse, open "Mojave > Edit variables..." and set parameters for precompiled version of the Einstein toolkit and a parameter file:

Name	**Value**
par	`par/static_tov.par`
thornlist	`thornlists/einsteintoolkit.th`
config	`et`
sim.name	`tov_star1`

Select "Mojave > Create", then "Mojave > Run" to create and run an evolution of a relativistic TOV star!

Simple visualization with gnuplot




```
cd ~/simulations/tov_star/output-0000
gnuplot> p "static_tov/hydrobase::rho.maximum.asc" u 2:3 w l
gnuplot> p "static_tov/admbase::alpha.x.asc" u 10:13 i 0 w l
```

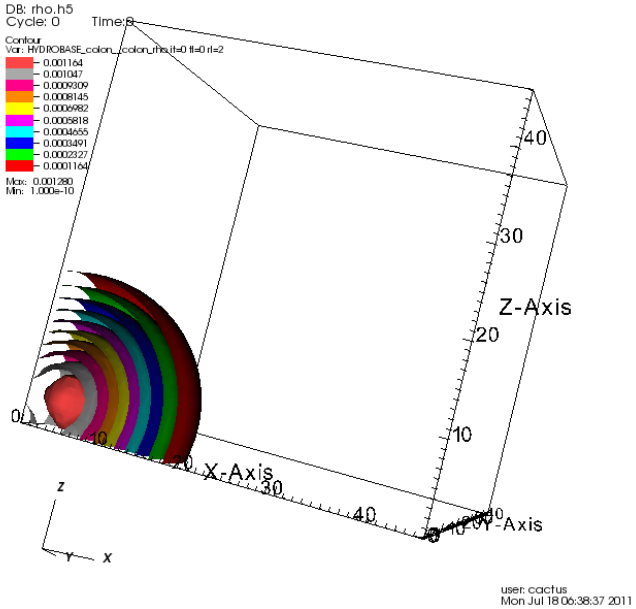
3D output and visualization

To enable 3D output, copy `static_tov.par` to a new parameter file (say, `static_tov_h5.par`) and append the following lines, then use it to create and run a new simulation:

```
IOHDF5::out_criterion = time
IOHDF5::out_dt        = 100.0
IOHDF5::out_vars = "
  grid::coordinates{out_every=10000000}
  HydroBase::rho
"
```

Using VisIt

VisIt is an interactive 3D visualization tool.



- Open VisIt
- Open "File > Open File...", navigate to the output of your simulation, or use:

  `~/simulations/tov_star/output-0000/s`

- Select filter: `*.h5`
- Select the star density: `rho.h5`

- Add contour plot for the second refinement level (`HYDROBASE::rho it=0 tl=0 rl=2`)
- Press "Draw" button.

VisIt website: `https://wci.llnl.gov/codes/visit/`

Thorn HTTPD

Remote steering/monitoring tool

- Mini web server implemented as a thorn
- Connect to a running simulation with a web browser
- View and steer simulation parameters
- Basic remote visualization

Adding HTTPD to your configuration

Using HTTPD is simple:

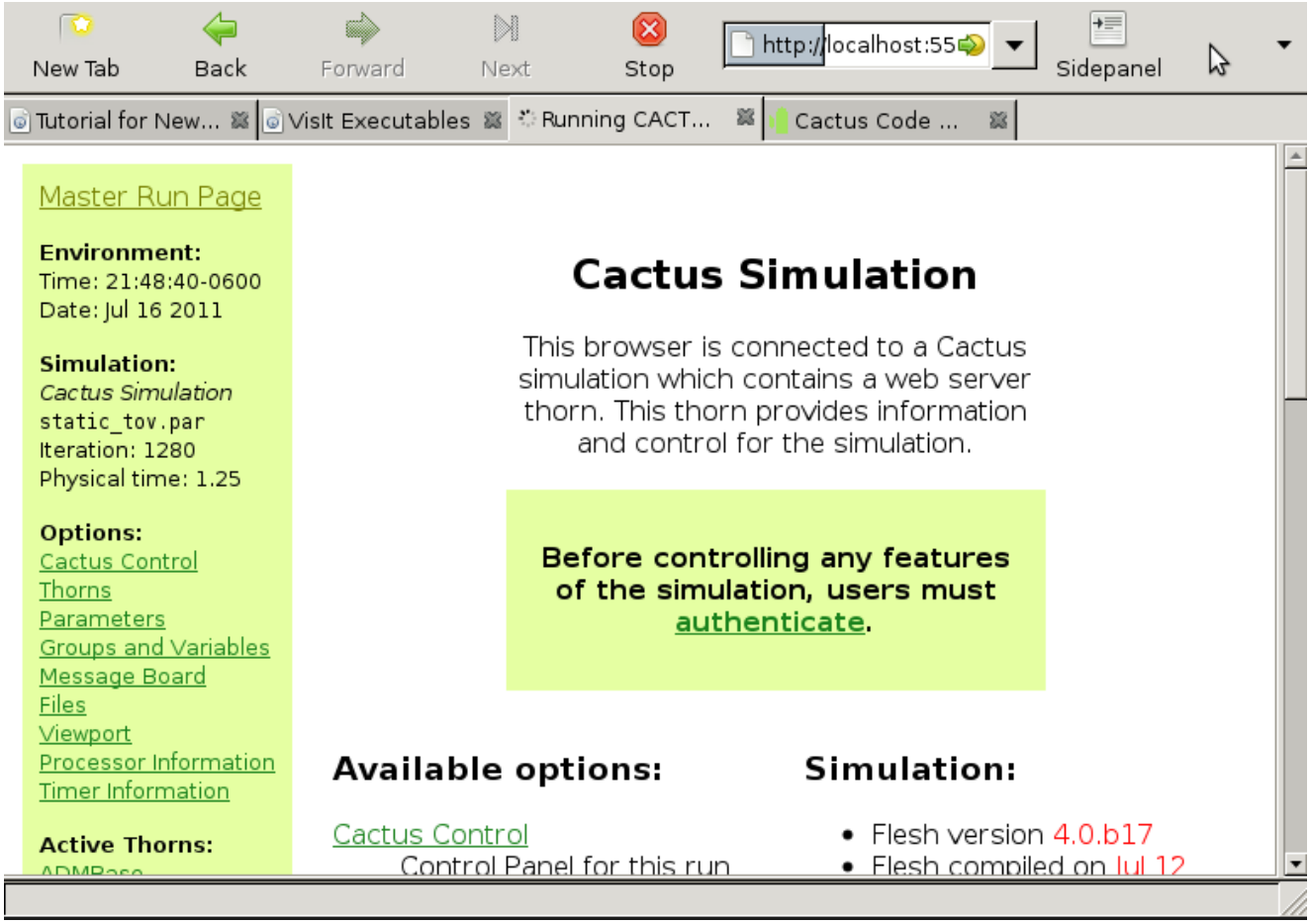
- Add the following thorns to your thornlist (file `~/Cactus/thornlists/BadWaves.th`):

```
CactusConnect/HTTPD
CactusConnect/HTTPDExtra
CactusConnect/Socket
```

- Enable these thorns in your parfile (for example, in `~/Cactus/pars/BadWaveAMR.par`):

```
ActiveThorns = "HTTPD HTTPDExtra Socket"
```

Sample HTTPD session



Sample HTTPD session

Master Run Page

Environment:
Time: 21:58:27-0600
Date: Jul 16 2011

Simulation:
Cactus Simulation
`static_tov.par`
Iteration: 7168
Physical time: 7.00

Options:
Cactus Control
Thorns
Parameters
Groups and Variables
Message Board
Files
Viewport
Processor Information
Timer Information

Control and Status Page

This page is the control center for interacting with the current simulation. It is possible to steer certain parameters, as well as pause, restart, or terminate the simulation.

Run Control

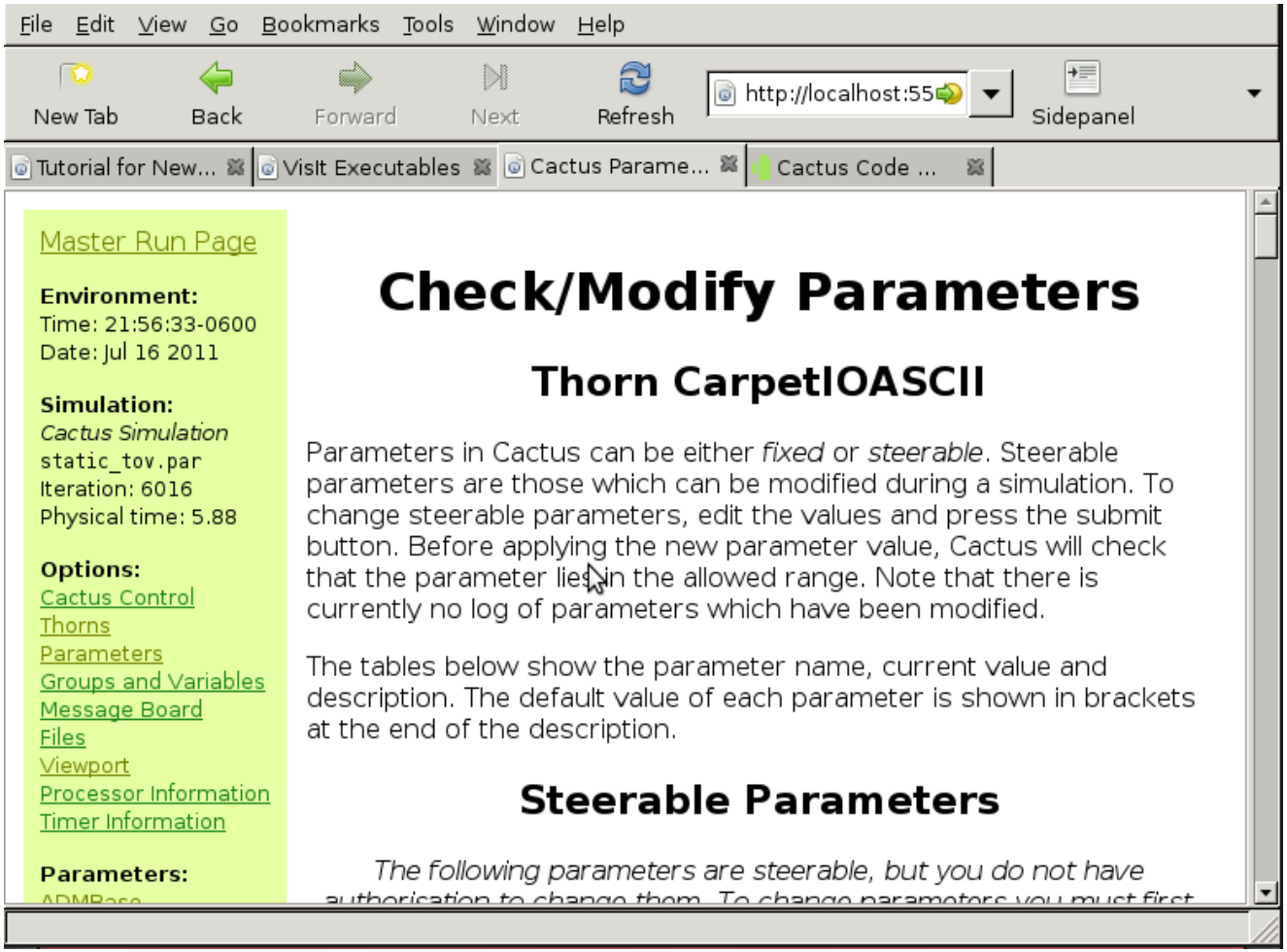
Select if the run should be paused, running normally, or terminated.
You may also single step to the next iteration.

○ PAUSE ◉ RUN ○ TERMINATE STEP

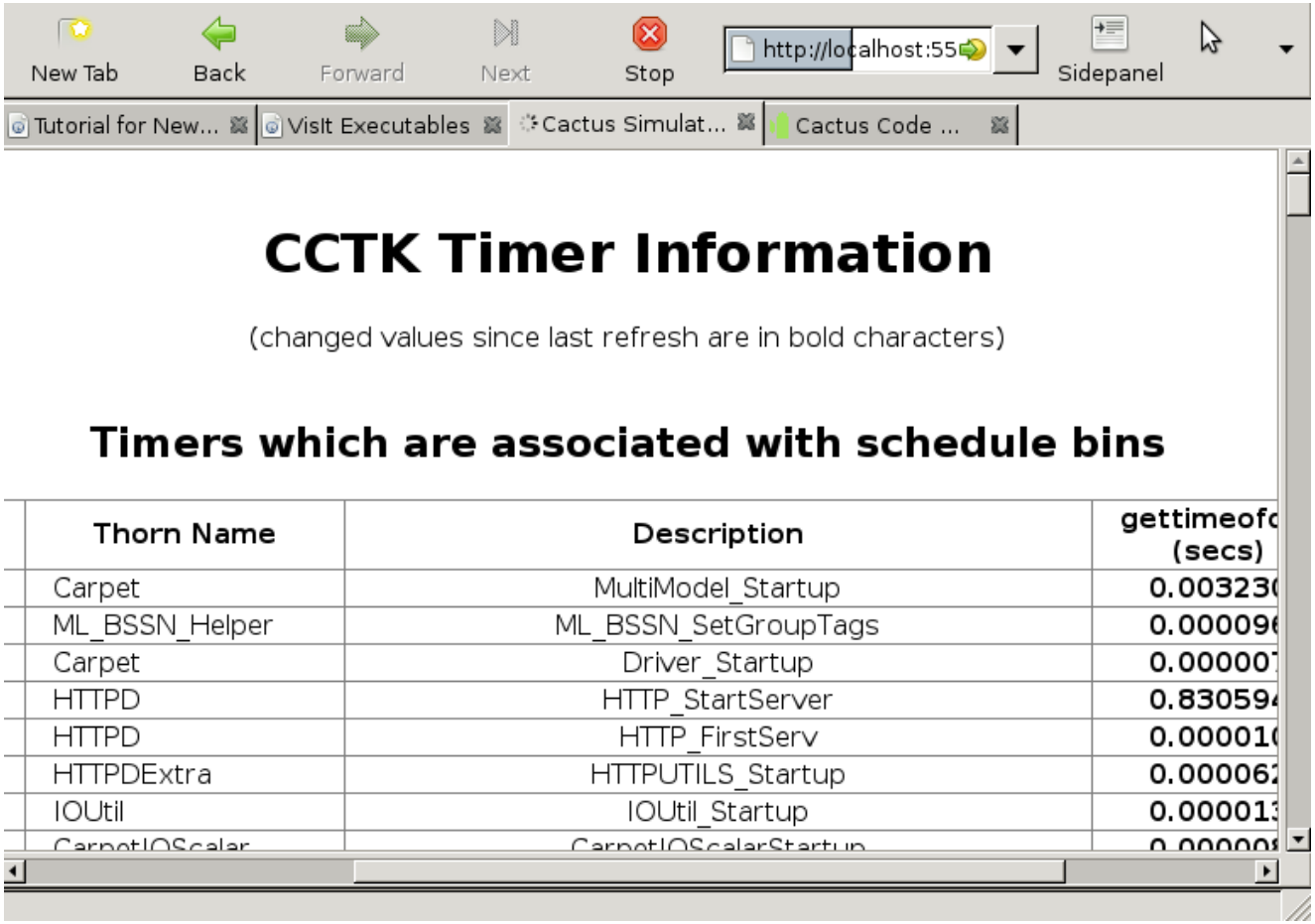
OK Reset

Run Until

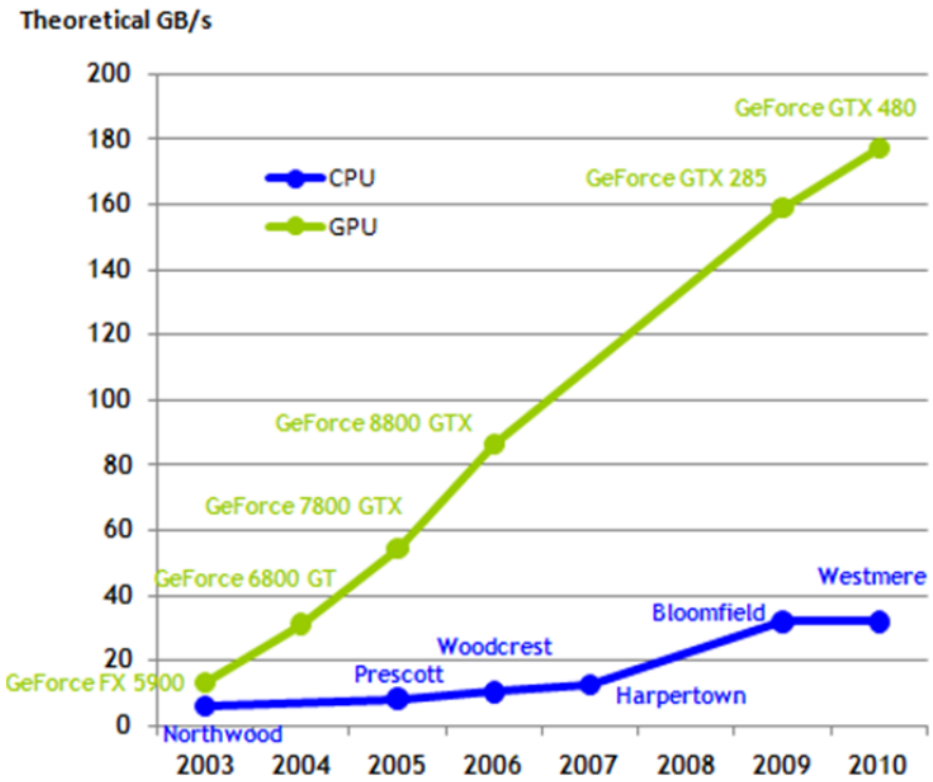
Sample HTTPD session



Sample HTTPD session

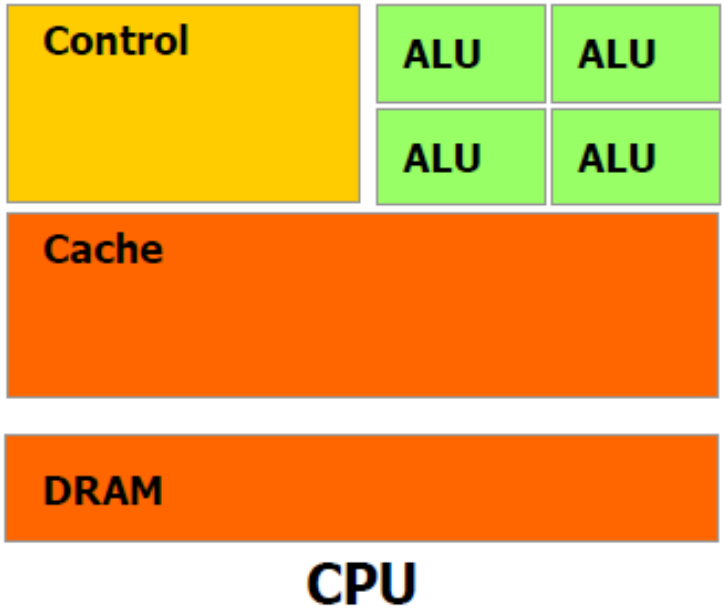


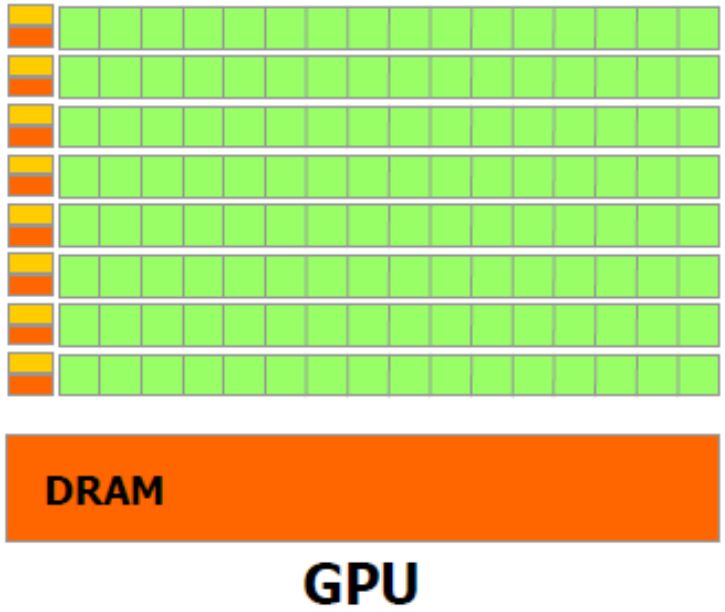
Motivations



- General Purpose Graphics Processing Units (GPGPUs) are high-performance graphical processors that can be used to accelerate a wide range of applications. (CUDA C Programming Guide Version 3.2)

GPU vs. CPU

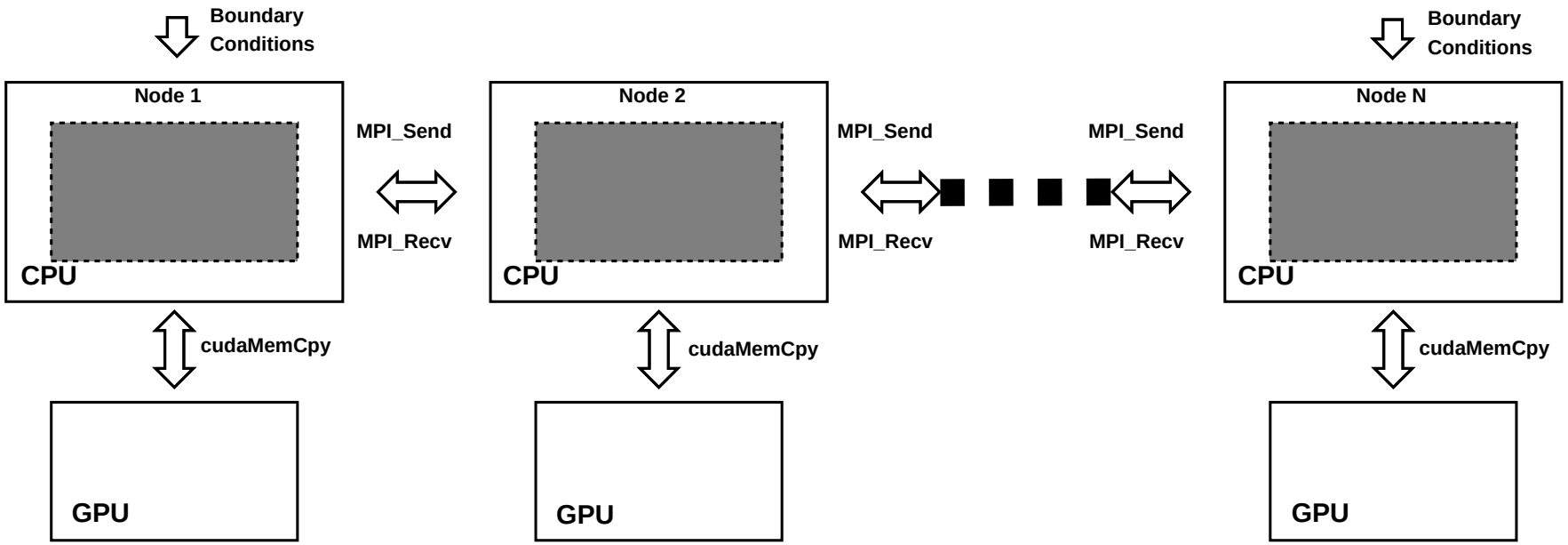




- Compared to CPU, GPU devotes more transistors to data processing rather than data caching and flow control (CUDA C Programming Guide Version 3.2)

Cactus CUDA Toolkit(under development)

- MPI + CUDA programming in Cactus is made simple.



CaCUDAViz

- The volume rendering of a scalar wave spreading from the center. using CUDA on a GPU, while the rendering parameters are steerable at runtime via the Cactus web server.