RunView: One-Glance, Zero-Effort
Representation of Program Efficiency within the Cactus Computational Framework

Anna Neshyba
Mentors: Dr. David Koppelman, Dr. Steven Brandt
The Scenario:

Consider this situation: a physicist is developing new simulation models.

- Focused on correctness, and discovery
- May not notice inefficiencies, may ignore them
The Project:

Create a one glance, zero-effort visual analysis of code performance and efficiency. Building a tool that:

1. understands certain code performance data and,
2. arranges it into a visually tractable format that compels the domain expert to fix the problem

This tool targets programs within the Cactus Computational Framework.
The Background

Hardware has certain peak computational capabilities:

- floating-point bandwidth
- instruction execution bandwidth
- data transfer bandwidth
The Data

- Uses embedded timers in Cactus
- Uses Performance Application Programming Interface (PAPI)
The Presentation

• Uses Elements of Art

• Uses SVG/HTML/JavaScript/jQuery
Prior Work


- Gprof
  - collects information for specially compiled programs

Prior Work

Not a new concept:


- HPCviewer & HPCtraceview

Prior Work

- Require knowledge of HPC
- Require effort
The Benefits:

The benefits of this tool for Cactus users:

• Should not require expertise or interaction

The benefits of developing this tool for Cactus:

• Can take advantage of embedded timers
The Features

• Routine Graphic

• Timeline Graphic
  • True time-line
  • Pattern Finder
Routine Graphic

Organized by:

- hierarchy of procedure calls
- aggregate run time.
Routine Graphic Interactive Features

**Zoom**: when clicked on,

- the graphic shifts to the left
- all of the routine’s descendants are scaled
Bubbles

- Scaled to L3 cache misses
- Represent disruption in flow
**Color**

<table>
<thead>
<tr>
<th>CallF</th>
<th>CallFu</th>
<th>CallFunction</th>
</tr>
</thead>
</table>

**Color**: the colors are dependent on two characteristics:

- Percentage of time spent running at full speed (full)
- Percentage of time spent running at full speed while not stalled (fullns).
Time-line Graphic

Organized by:

- time executed
- Shared horizontal coordinates are happening simultaneously

Provides a visual conception of how the program is running
Time-line Graphic

Pattern Finder:

• Identifies repeating sequences
• Important for inefficiency mitigation
Discovery:

- Allows domain experts to discover inefficiencies
Future Work

• Envision and incorporate characteristics for other forms of inefficiencies
• Test on domain experts
• Further interactive features
Thank you!

Questions?

Contact: annaneshyba@gmail.com