ParaViz3D: MPI Trace Visualization with 3D Animation

Jean-Yves Verhaeghe, NHR@FAU

23.02.2026

Section 1

Theoretical Problem

Summary

What will be discussed in this presentation

Introduction

1. Trace data and its traditional display
2. Limits of this display, and how we propose to improve it
3. Results of our visualization method
4. This is a work in progress

Summary

What will be discussed in this presentation

Introduction

1. Trace data and its traditional display
2. Limits of this display, and how we propose to improve it
3. Results of our visualization method
4. This is a work in progress

Used programs

1. Jacobi codes
2. Idle waves
3. Different run conditions and different domains

Theoretical Problem

Small introduction about trace data

What is it, how and why is it used?

- Systematic log of events of a program's execution
- Essential to understand, compare, investigate programs
- Obfuscated list of timestamps : hard to picture
- Need to display it for easier comprehension
- Used tool: **Score-P** (OTF2 format)

Theoretical Problem
The traditional display

List display

- Horizontal axis for time
- Vertical axis for processes
- Color coded rectangles for each region (function)

How does it help?

- Visual representation
- Seeing event synchronicity
- Time-related phenomenons

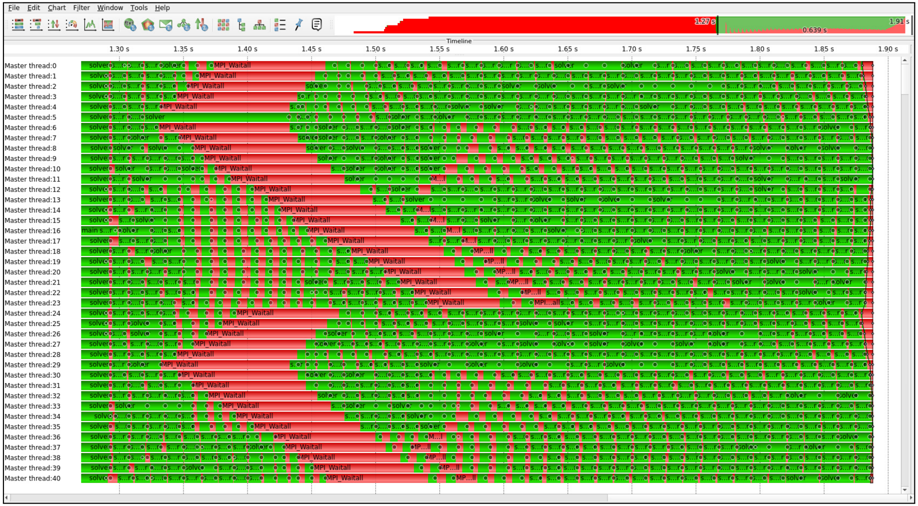


Figure: Vampir output example

Theoretical Problem

The traditional display

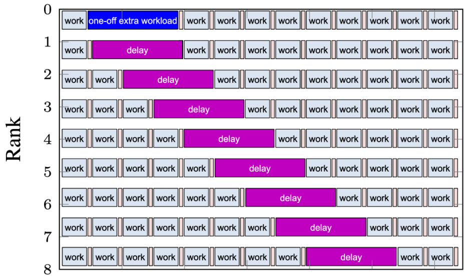
Limits

- Many programs adapt to their problem's geometry
- Line data is ok, but what about grids or spheres?
- Contiguous ranks in the domain and in the communication scheme are often separated
- Unable to display too many ranks
- 2 axis displays can only elegantly show 1D communication
- Many other subtleties (examples later)

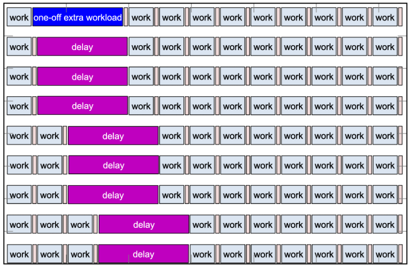
Theoretical Problem
Used example

Which program was used for our development?

- Jacobi codes that regularly need to communicate with direct neighbors
- Extra workload on rank 5
- Its neighbor need to wait for the lagger's communication
- In turn, their neighbor wait, idling processes pile up
- Rank 5 finishes and communicates
- In turn neighboring ranks resume computation and communicate
- This creates a ripple effect of idling processes called **idle wave**



(a) $d = \pm(1)$ (slow idle wave)



(b) $d = \pm(1, 2)$ (three times faster idle wave)

Figure: Idle wave example

Theoretical Problem

What kind of tool did we explore?

Requirements for the display

- Generates videos/animations: frees up the time axis
- Display in 3D: straightforward representation of data in up to 3 dimensions

Theoretical Problem

How to display idle waves?

Making a tool to visualize idle waves

- We initially used Blender to generate this display (later: browser display)
- We created a Python script to parse the OTF2 traces
- We explored smart ways to display traces in the most understandable way (colors, transparency, geometry, etc.)

Section 2

Testbed

Testbed

Idle waves generated through Fritz nodes

The Fritz computer nodes

- CPUs: 2x Intel Xeon Platinum 8360Y
- Nb of cores: 2 x 36 = 72
- Frequency: 2.4 GHz
- L3 cache: 2 x 54 MB = 108 MB
- NUMA LDs: 4
- Memory: 256GB DDR4-3200
- SMT is disabled

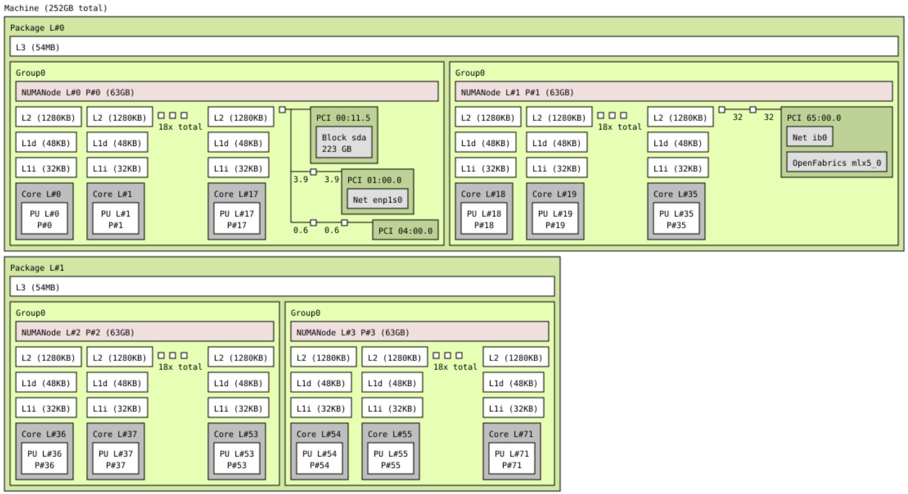


Figure: Fritz topology

Section 3

Tool usage

Python script

How to go from a Score-P trace to a ParaViz3D file?

Key Steps

- A provided python script reads through the trace
- Sorts accordingly and agressively filters
- Output a text file 100 times smaller
- This scheme allows for big traces to be handled

Python script

How to go from a Score-P trace to a ParaViz3D file?

Script demo

Section 4

Results

Blender output demo

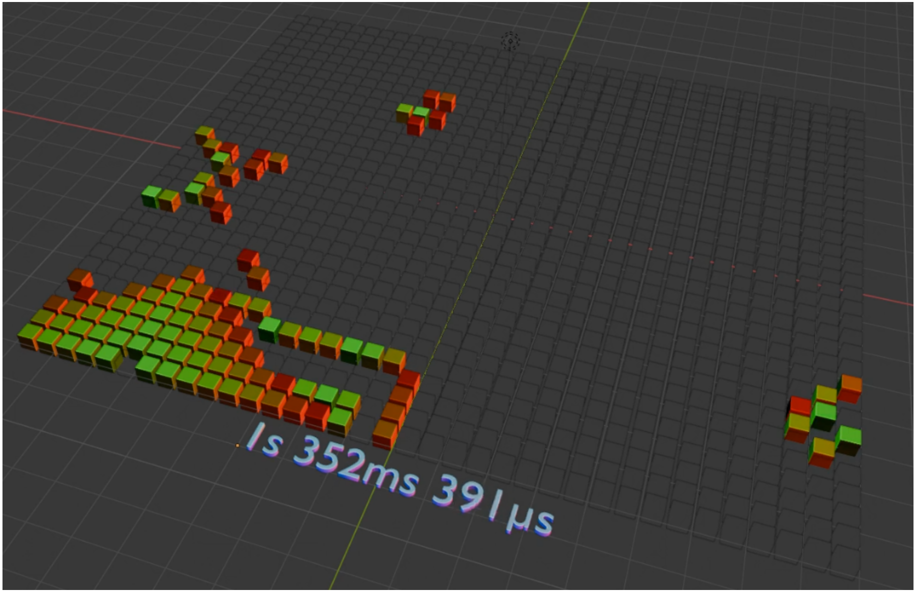


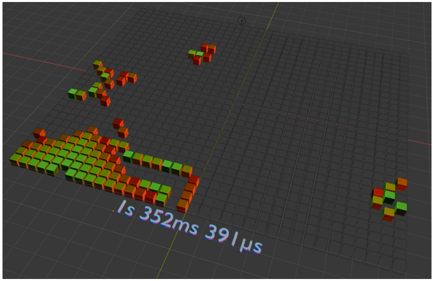
Figure: Idle wave example in 2D grid

Results

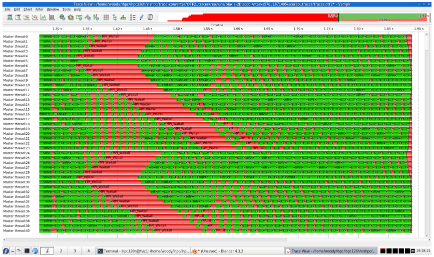
Screenshots of the tool

Blender vs Vampir

- Screenshot of the view



(a) Blender output



(b) Vampir output

Figure: biaxis 2D Jacobi

Results

Visualization steps

Display experiments and choices

- Red for MPI, blue for computation?

Results

Visualization steps

Display experiments and choices

- Red for MPI, blue for computation?
- Red for MPI, transparency for computation (+sparsity in 3D grid)?

Results

Visualization steps

Display experiments and choices

- Red for MPI, blue for computation?
- Red for MPI, transparency for computation (+sparsity in 3D grid)?
- Selected default: Color gradient, progressively going from red to green over the span of MPI regions

Results

Visualization steps

Display experiments and choices

- Red for MPI, blue for computation?
- Red for MPI, transparency for computation (+sparsity in 3D grid)?
- Selected default: Color gradient, progressively going from red to green over the span of MPI regions
- Adapt to user selection

Results

3D output

Tool advantages

- 2D an 3D domains accurately displayed

Results
3D output

Tool advantages

- 2D an 3D domains accurately displayed
- Straightforward and intuitive presentation of the trace

Results
3D output

Tool advantages

- 2D an 3D domains accurately displayed
- Straightforward and intuitive presentation of the trace
- Faster and finer comprehension of the program

Results
3D output

How can we improve it further?

- Develop additional metrics and statistics shown in 3D

Results

3D output

How can we improve it further?

- Develop additional metrics and statistics shown in 3D
- Making them customizable by the user, preferably on-the-fly

Results
3D output

How can we improve it further?

- Develop additional metrics and statistics shown in 3D
- Making them customizable by the user, preferably on-the-fly
- Support custom shapes, and meshes, including unstructured

Results
3D output

How can we improve it further?

- Develop additional metrics and statistics shown in 3D
- Making them customizable by the user, preferably on-the-fly
- Support custom shapes, and meshes, including unstructured
- Using VR for real 3D and live rotate, zoom, pan

Results
3D output

How can we improve it further?

- Develop additional metrics and statistics shown in 3D
- Making them customizable by the user, preferably on-the-fly
- Support custom shapes, and meshes, including unstructured
- Using VR for real 3D and live rotate, zoom, pan
- Making it lightweight and scalable

Results
3D output

How can we improve it further?

- Develop additional metrics and statistics shown in 3D
- Making them customizable by the user, preferably on-the-fly
- Support custom shapes, and meshes, including unstructured
- Using VR for real 3D and live rotate, zoom, pan
- Making it lightweight and scalable
- Using JavaScript to display over the browser

Section 5

Current work: browser output

Browser output

Browser output demo

Q&A Session

Special thanks to Georg Hager, and Ayesha Afzal!