Strategies for Scalable Data Analysis and Visualization

Bernd Hentschel, RWTH Aachen University
Agenda

• Why visualization?
• Why scalable?
• “Scalable” by example – Particle Tracing
• Summary
Introduction
“The purpose of computing is insight, not numbers!"

R. W. Hamming, 1962
The Situation

Computing

Numbers
Terabytes-Petabytes-Exabytes

JUQUEEN JSC,
Forschungszentrum Jülich GmbH

1101111011010110
0100101010010101
0100101010101001
0101110101011010
111100...

0100101010010101
0100101010101001
0101110101011010
111100...

Visual Computing Institute | Prof. Dr. Torsten W. Kuhlen
Virtual Reality & Immersive Visualization
B. Hentschel | Strategies for Scalable Data Analysis and Visualization | 30.11.2016
The Situation

Computing → Numbers: Terabytes-Petabytes-Exabytes → Visualization → Insight

Numbers:

\[
\begin{align*}
0010100101001111 & \\
0100101010010101 & \\
0100101010101001 & \\
0101110101011010 & \\
111100 & \\
0100101010010101 & \\
0100101010101001 & \\
0101110101011010 & \\
0101001010101001 & \\
0101110101011010 & \\
011100 & \\
\end{align*}
\]

Insight
The Situation

Simulation

Post-processing

Computing

Numbers
Terabytes-Petabytes-Exabytes

Visualization

Insight

JUQUEEN JSC, Forschungszentrum Jülich GmbH

0101111010101101 0101010010011110 0100101010010101 0101110101011010 11100...

Numbers Terabytes-Petabytes-Exabytes

Computing

Visualization

Insight
Simulation-Based Research Results @ RWTH

**Turbulent/Non-Turbulent Interface Detection**

**Liquid Sheet Breakup**

**Simulation of a Helicopter Jet Engine**

**Simulation of Chromatography Columns**


[Göbbert, Bode, Pitsch; Proposal for Computing Time on the JARA-HPC, 2014]

[Püttmann, Nicolai, Behr, von Lieres; Finite Elements in Analysis and Design, 2014]

[Cetin, Meinke Schröder; STAB, 2014]
Simulation-Based Research Results @ RWTH

Turbulent/Non-Turbulent Interface Detection

Liquid Sheet Breakup

Simulation of a Helicopter Jet Engine

Simulation of Chromatography Columns
Simulation-Based Research Results @ RWTH

**Turbulent/Non-Turbulent Interface Detection**
- Analysis based on 1,024 x 768 x 768 grid
- 18 GB per time step
- 3 time steps used in paper
  ➔ **54 GB used for analysis**
- 4,096 x 2,048 x 1,536 available
- **384 GB per time step**

**Liquid Sheet Breakup**
- 150 mio. gridpts
- 14.6 GB per time step
- 3,000 time steps
  ➔ **~43.7 TB of raw data**

**Simulation of a Helicopter Jet Engine**
- 330 mio. gridpts
- 41 GB per time step
- 3,000 time steps
  ➔ **~120 TB of raw data**

**Simulation of Chromatography Columns**
- 25 mio. degrees of freedom
- 20 mio tetrahedral cells
- ~1GB raw data per time step
- 354 time steps
  ➔ **354 GB of raw data**
The Common Post-Processing Approach

HPC Center

HPC-System

HPC File System

Analysis$_1$

Simulation

Local Systems

Analysis$_2$

Visualization

…

Generate TB of data!

Write TB of data!

Read TB of data!

Transfer TB of data!
In-Situ Processing of Large Simulation Data

HPC Center

HPC-System

Simulation

In-situ processing

HPC File System

Local Systems

Analysis$_1$

Visualization

Analysis$_2$

...

Generate TB of data!

Write GB/MB of data!

Read GB/MB of data!

Transfer GB/MB of data!

Vis needs to scale!
Towards Scalable Particle Tracing
Example – Streamsurfaces

[Hummel et al. “IRIS: Illustrative Rendering of Integral Surfaces”, 2012]
Example – Monte Carlo FTLE

MCFTLE, 5.2k iterations, $1.9 \cdot 10^{11}$ particle integrations, 117.6h compute

Particle Tracing Basics

\[ \mathcal{L}(t_0) = x_0 \]

\[ \frac{d\mathcal{L}}{dt} = v(\mathcal{L}, t) \]
**Particle Tracing Algorithm**

Seed new particle trace at initial position $s_0$

While $s_i$ inside domain

- Determine cell which contains $s_i$
- Interpolate vector value $v$ at $s_i$
- Integrate $s_{i+1} = \text{Step}(v, s_i)$  //may require additional interpolations
- Add line segment $(s_i, s_{i+1})$ to output
Particle Tracing Challenges

Data size

Seed set size

Seed distribution

Vector field complexity

[D. Pugmire et al. “Scalable Computation of Streamlines on Very Large Datasets“, 2009]
Basic Parallelization

Parallelize over Seeds

Parallelize over Blocks

[D. Pugmire et al. “Scalable Computation of Streamlines on Very Large Datasets”, 2009]
[D. Camp et al. “Streamline Integration Using MPI-Hybrid Parallelism on a Large Multicore Architecture”, 2011]
Hybrid Parallelization – MPI / OpenMP

Hybrid POS

Hybrid POB

Increase per node parallelism!

[D. Camp et al. “Streamline Integration Using MPI-HybridParallelism on a Large Multicore Architecture”, 2011]
Hybrid Parallelization – Results

Parallelize over Seeds
Improvement MPI-Hybrid over MPI-only ($T_{total}$)

128 x 1 vs. 32 x 4

[D. Camp et al. “Streamline Integration Using MPI-HybridParallelism on a Large Multicore Architecture”, 2011]
Packet-Oriented Particle Tracing

[B. Hentschel et al “Packet-Oriented Streamline Tracing on Modern SIMD Architectures”, 2015]
Packet-Oriented Particle Tracing – Auto Vectorization

[B. Hentschel et al “Packet-Oriented Streamline Tracing on Modern SIMD Architectures”, 2015]
Packet-Oriented Particle Tracing – Optimized Results

5.6x on 8-way SIMD

[B. Hentschel et al “Packet-Oriented Streamline Tracing on Modern SIMD Architectures”, 2015]
Work Requesting – Method

[C. Mueller et al “Distributed Parallel Particle Advection using Work Requesting”, 2013]
Work Requesting – Method

[C. Mueller et al “Distributed Parallel Particle Advection using Work Requesting”, 2013]
Work Requesting – Method

[C. Mueller et al “Distributed Parallel Particle Advection using Work Requesting”, 2013]
Work Requesting – Method

[C. Mueller et al “Distributed Parallel Particle Advection using Work Requesting”, 2013]
[C. Mueller et al “Distributed Parallel Particle Advection using Work Requesting”, 2013]
Work Requesting – Method

[C. Mueller et al “Distributed Parallel Particle Advection using Work Requesting”, 2013]
Work Requesting – Results

Work Requesting

Standard POS

[C. Mueller et al “Distributed Parallel Particle Advection using Work Requesting”, 2013]
Work Requesting – Results

[C. Mueller et al “Distributed Parallel Particle Advection using Work Requesting”, 2013]
Wrap-Up
Summary

- Scalable simulation-based workflows require scalable vis.
- Scaling vis is a challenge on multiple levels.
- Tightly integrating “in situ” will make things even more interesting.
Ongoing Work – The In Situ Terminology Project

Community effort to describe the in situ design space with a unified vocabulary

The In Situ Terminology Project

Hank Childs, Andrew Bauer, E. Wes Bethel, Joseph Cottam, Matthieu Dorier, Thomas Haller, Tim Eyre, Berk Geveci, Bernd Hentschel, Joseph Insley, Aaron Knoll, Ira Mihov, Mike Murphy, Jay Lofström, Jeremy Meredith, Kenneth Moreland, Manish Parashar, Steve Petritzka, Doug Ritzhaupt

Abstract
The term “in situ processing” has become widely used for processing data and an umbrella term for many industry-related terms. To address this problem, a group of researchers at the In Situ Terminology Project created a unified vocabulary for in situ visualization and analysis science. This paper summarizes the findings of the project and presents the project’s work. An important finding from this group was that there are three key aspects to in situ processing: integration type, proximity, access, division of execution, and operation controls. The paper discusses these aspects, evaluates existing systems within the in situ processing design space, and terms relate to the axes.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation
Acknowledgements

• Christoph Garth, University of Kaiserslautern
• Hank Childs, University of Oregon
• Cornelius Müller, (University of Kaiserslautern)
• Dave Camp, LBNL
• Michael Klemm (Intel)
• The Virtual Reality and Immersive Visualization Group
Thank you for your attention!

hentschel@vr.rwth-aachen.de
Packet-Oriented Particles – Data Layout

Seed positions $s_i$ in SOA

Vector field $\mathbf{v}$ in AOS

Swizzle to SOA

Linear Interpolation in SOA

$+ \Delta t \times$

Internal advection step in SOA

$X_0 \quad X_1 \quad X_2 \quad \ldots$

$Y_0 \quad Y_1 \quad Y_2 \quad \ldots$

$Z_0 \quad Z_1 \quad Z_2 \quad \ldots$

$t_0 \quad t_1 \quad t_2 \quad \ldots$

$U_0 \quad V_0 \quad W_0 \quad U_1 \quad V_1 \quad W_1 \quad \ldots$

$8 \times \text{simd-width}$

vector values
Packet-Oriented Particles – Optimized Memory Access (AVX2)

1. SIMD loads

2. Rearrange

Two velocity tuples

256 bit
Packet-Oriented Particles – Diverging Particles

\[ x_0^{(0)} \quad x_0^{(1)} \quad x_0^{(2)} \quad x_0^{(3)} \]

\[ x_{k_2}^{(0)} \quad x_{k_2}^{(1)} \quad x_{k_2}^{(2)} \quad x_{k_2}^{(3)} \]

\[ x_{k_3}^{(0)} \quad x_{k_3}^{(1)} \quad x_{k_3}^{(2)} \quad x_{k_3}^{(3)} \]

\[ x_{k_1}^{(0)} \quad x_{k_1}^{(1)} \quad x_{k_1}^{(2)} \quad x_{k_1}^{(3)} \]
Packet-Oriented Particles – Divergence in the Wild

SIMD efficiency

channel flow
shear flow
isotropic

normalized integration step

Visual Computing Institute | Prof. Dr. Torsten W. Kuhlen
Virtual Reality & Immersive Visualization
B. Hentschel | Strategies for Scalable Data Analysis and Visualization | 30.11.2016
Packet-Oriented Particles – Repacking
Results – Repacking: Channel

![Graph showing integration step vs. integrations/s and SIMD efficiency over normalized integration step]
Results – Repacking: Shear Flow
Results – Repacking: Isotropic Turbulence

![Graph showing integration step vs. SIMD efficiency for isotropic turbulence.](image-url)