Parallel Computation Of 2D Morse-Smale Complexes
Nithin Shivashankar, Senthilnathan Maadaswamy and Vijay Natarajan
Department Of Computer Science And Automation, IISc, Bangalore

Objective
Parallel computation of the Morse-Smale complex designed for multi-core and GPU environments.
Computation for datasets that don’t fit in Memory.

The Morse-Smale Complex
Input: Scalar (real valued) function
Example: 2D plane \( \sin(x) + \sin(y) \).

Gradient curves: Curves that trace the direction of steepest descent
Critical Points: Points of origin/destination of gradient curves
Morse-Smale complex: Partition based on origin/destination of gradient curves.

Parallel Morse-Smale(MS) Complex Algorithm

INPUT: Domain as a Cell complex and a scalar function sampled at vertices.
1. Compute discrete gradient
   KEY RESULT: Computing discrete gradient is independent of order. Can be done in parallel.
2. Traverse the gradient field to compute the MS complex.

Large Data: Divide and conquer
KEY RESULT: Order of merging does not affect the result

Evaluation
Experiments with synthetic and real world datasets on 2D grids.
Observed near linear scaling with data size and number of cores.

Reference

Contact and Acknowledgements
nithin@csa.iisc.ernet.in
senthilnathan.m@intel.com
vijayn@csa.iisc.ernet.in
Work supported by Intel and DST.

Parallel Computation Of 2D Morse-Smale Complexes
Nithin Shivashankar, Senthilnathan Maadaswamy and Vijay Natarajan
Department Of Computer Science And Automation, IISc, Bangalore

Objective
Parallel computation of the Morse-Smale complex designed for multi-core and GPU environments.
Computation for datasets that don’t fit in Memory.

The Morse-Smale Complex
Input: Scalar (real valued) function
Example: 2D plane \( \sin(x) + \sin(y) \).

Gradient curves: Curves that trace the direction of steepest descent
Critical Points: Points of origin/destination of gradient curves
Morse-Smale complex: Partition based on origin/destination of gradient curves.

Parallel Morse-Smale(MS) Complex Algorithm

INPUT: Domain as a Cell complex and a scalar function sampled at vertices.
1. Compute discrete gradient
   KEY RESULT: Computing discrete gradient is independent of order. Can be done in parallel.
2. Traverse the gradient field to compute the MS complex.

Large Data: Divide and conquer
KEY RESULT: Order of merging does not affect the result

Evaluation
Experiments with synthetic and real world datasets on 2D grids.
Observed near linear scaling with data size and number of cores.

Reference

Contact and Acknowledgements
nithin@csa.iisc.ernet.in
senthilnathan.m@intel.com
vijayn@csa.iisc.ernet.in
Work supported by Intel and DST.
The Reeb graph of a scalar function is obtained by mapping each connected component of its level sets to a point.

**The ReCon Algorithm**

1. Identify the loop saddles of the input
2. Split the input at a function value infinitesimally above that of the loop saddles to obtain a set of simply connected interval volumes
3. Compute the contour trees for each interval volume
4. Construct the Reeb graph by computing the union of these contour trees

**Properties**

1. Efficient: has a running time of $O(n \log n + s n)$
   - $n$ - # triangles
   - $s$ - # saddles
2. Generic: works without any modifications on $d$-manifolds and non-manifolds
3. Easy to implement
4. Handles data that do not fit in memory
5. At least an order of magnitude faster than existing generic algorithms

**References**

- Pascucci et al. ACM Trans. Graph 2007 (Online)
- Doraiswamy et al. IEEE TVCG 2011 (OS)
- Tierny et al. IEEE TVCG 2009 (LS)
- Harvey et al. SCG 2010 (Rand)

**Applications**

- Topology based shape matching. Hilaga et al. SIGGRAPH 2001
- Transfer function design. Weber et al. TVCG 2007
- User Interface. Bajaj et al. Vis 1997

**Handling Large data**

- David 56M
  - Time taken: $x = 3.6m, y = 3.8m, z = 3.2m$ (4.7m, 4.8m, 16.6m)
- St. Matthew 372M
  - Time taken: $x = 26.9m, y = 26.7m, z = 25.2m$ (40m, 42hrs, 41m)
- Atlas 507M
  - Time taken: $x = 41.5m, y = 38.5m, z = 42.6m$ (a, b, c)

**Acknowledgements**

Harish Doraiswamy was supported by Microsoft Corporation and Microsoft Research India under the Microsoft Research India PhD Fellowship Award. This work was supported by the Department of Science and Technology, India, under Grant SR/S3/EECE/048/2007
Efficient Online Visualization and Simulations for Large-scale Applications

**Problem Statement**

Develop an adaptive integrated steering framework that
- allows simultaneous simulation and online visualization
- spawns high-resolution simulation dynamically over desired region-of-interest
- supports optimal processor allocation for simulation
- supports optimal frequency of output for visualization

**Simulation-Visualization Lag**

Illustration of increasing visualization lag

**Framework for Adaptive Simulation and Online Visualization**

Decision Algorithm

APPLICATION MANAGER

# Processors Output Frequency

FRAME SELECTION ALGORITHM

Output Frames

SIMULATION

Stall if no disk space

Storage

ADAPTIVE FRAME SENDER

Network

FRAME RECEIVER

**Frame Selection Algorithms**

- Most-recent
  - Transfer the most-recently simulated frame
- Auto-clustering
  - Modified k-means for temporal clustering to select the most-representative frames
- Adaptive
  - Transfer full or reduced frames within acceptable lag bound

**Results**

Lag reduced by 90%

http://garl.serc.iisc.ernet.in/
http://vgl.serc.iisc.ernet.in/
Symmetry in Scalar Field Topology

Dilip Thomas

dilip@csa.iisc.ernet.in

Vijay Natarajan

vijayn@csa.iisc.ernet.in

Symmetric structures in EM image of RuBisCO molecule

Four types of symmetric regions identified

Classify subtrees of the contour tree

Motivation

Scalar fields contain repeating patterns

Provide insights on scientific phenomena

Technique

Classify subtrees of the contour tree

Similarity measure to compare subtrees

Applications

Symmetry-aware isosurface extraction

Symmetry-aware transfer function design