Parallel Feature Identification and Elimination from a CFD Dataset

Jeremy Davis

CSE 260

November 30, 2006

Introduction

- Analysis of scientific data places a high demand on computing resources
  - Computational complexity (processing cost)
  - Large data sets (memory and I/O cost)
- Parallel processing can help
  - Split computation among multiple processors
  - Larger overall memory size

Datasets

- This project uses a set of 3D computational fluid dynamics (CFD) simulation datasets
  - Discrete field data
  - Each point contains flow velocity (X, Y, and Z directions), pressure, and density values
  - Two sizes:
    - 385 x 130 x 194 (370 MB)
    - 642 x 193 x 385 (1830 MB)
  - One file per time increment

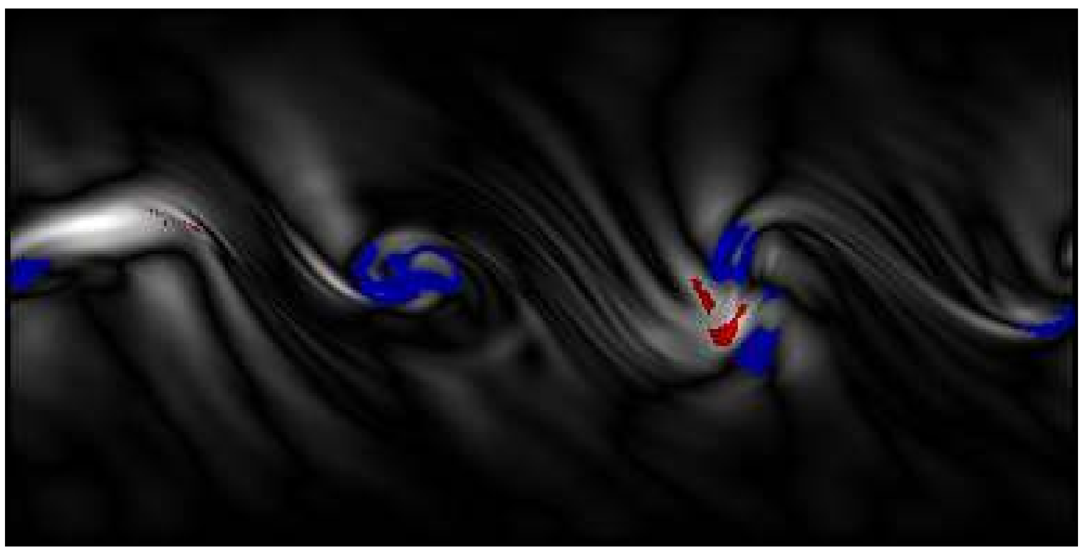
Analysis

- Analysis consisted of calculating vorticity at each point, and identifying features which match certain characteristics
  - Vorticity calculated from flow velocities of nearby points
  - Thresholding used to identify points which qualify as a feature

Vorticity Features

- Columns – Points corresponding to high vertical vorticity and low horizontal vorticity
- Dislocations – Points corresponding to low vertical vorticity and high horizontal vorticity

Y



X

Analysis Steps

- Partition the data to allow parallel computation
- Calculate vorticity
- Organize data points based on vorticity values
- Identify features
- Calculate and plot results

Data Partitioning

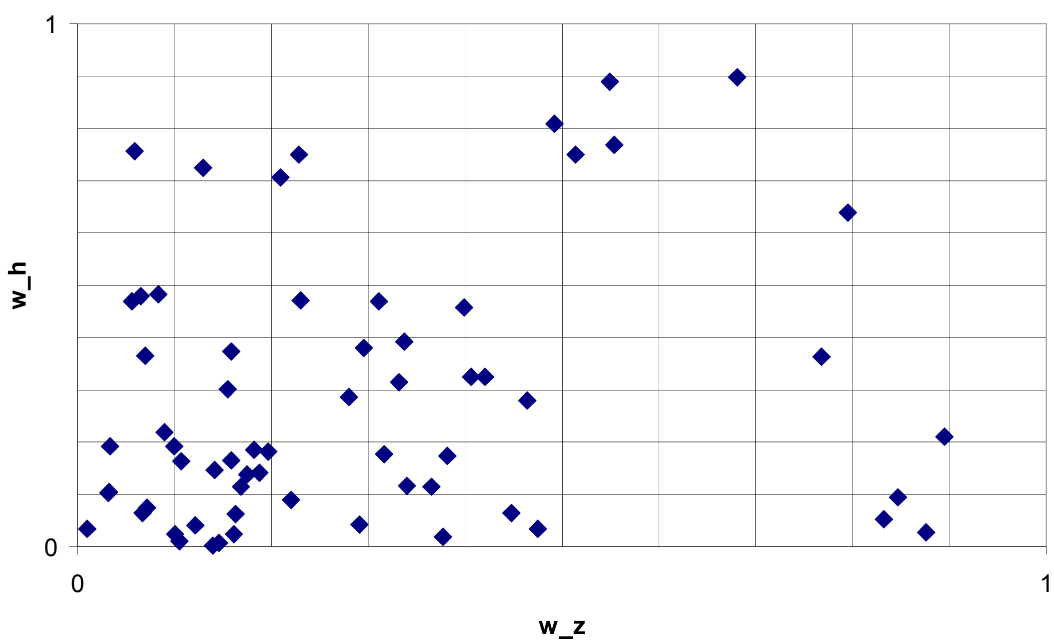
- Dataset is first partitioned into distinct 3D regions
  - Each parallel process will work with a subset of the available regions
  - Some points duplicated at region boundaries to allow independent vorticity calculation
  - I/O intensive
  - Not scalable (device contention)

Vorticity Calculation

- Each process calculates vorticity values for all points within its assigned region(s)
- Highly scalable
  - No communication needed – processes can work independently within their own regions

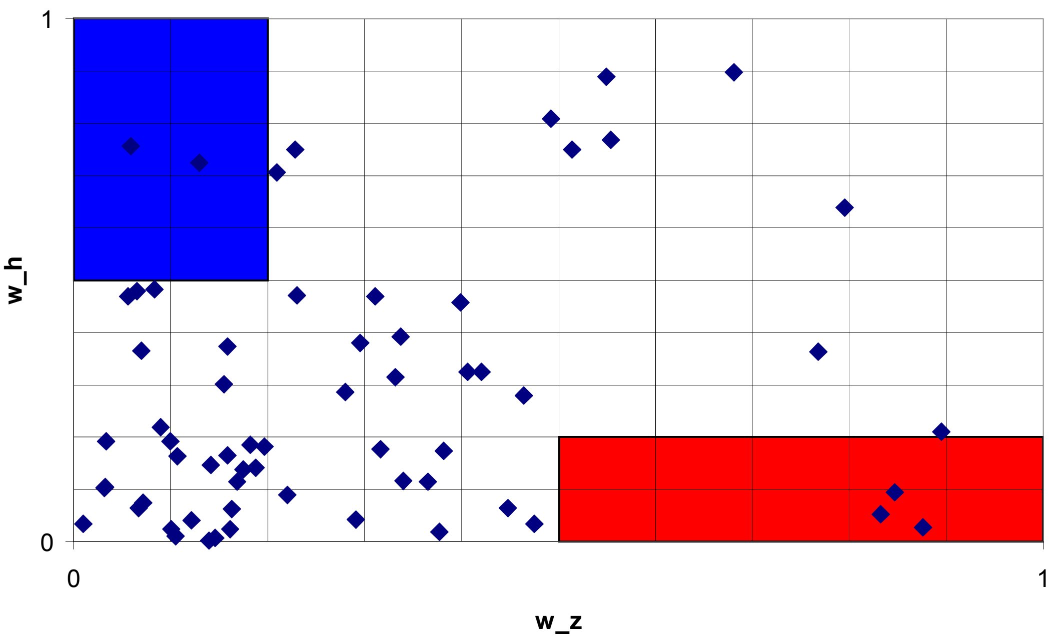
Data Organization

- As vorticity is calculated, identifiers for each point are added to a spatial data structure
  - Horizontal and vertical vorticity determine spatial coordinates



Identify Features

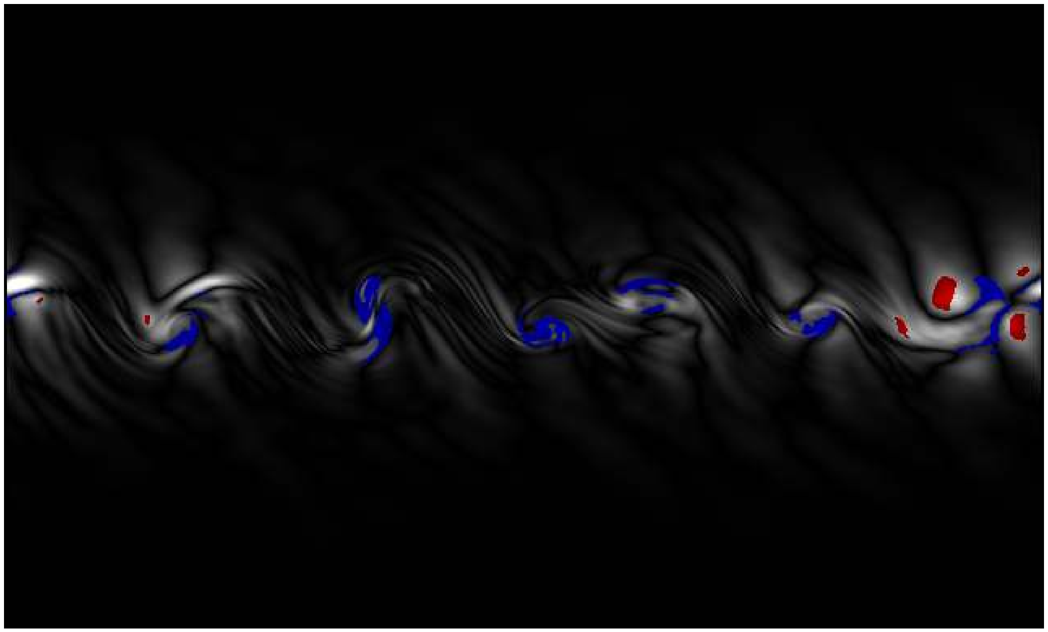
- Points meeting the feature thresholds can be found via a spatial query
  - Only check points that are within or close to the threshold values
  - Incremental queries can be done using prior results



Calculate and Plot Results

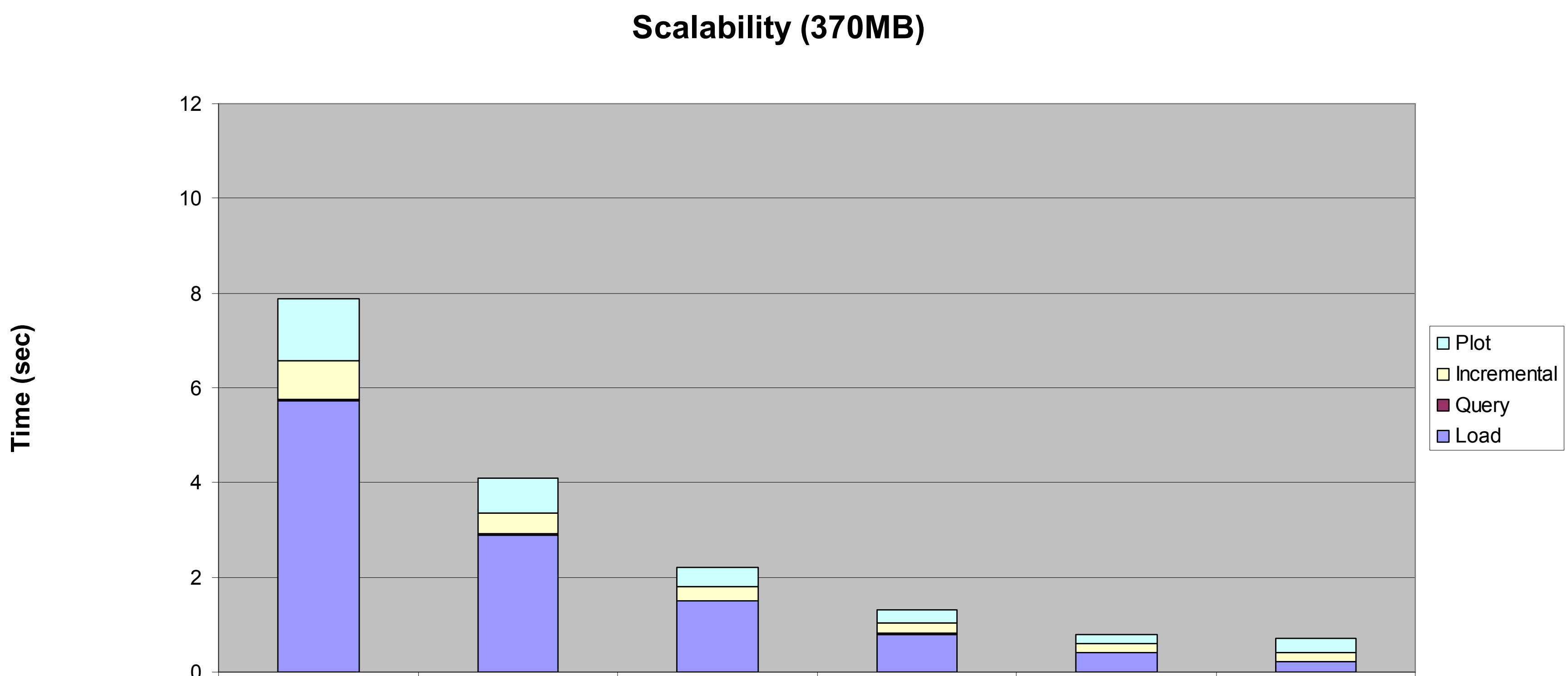
- Once features are identified, the results can be visualized, or further calculations can be performed
  - Aggregate values for feature points, or eliminate features and analyze remaining points

Y



X

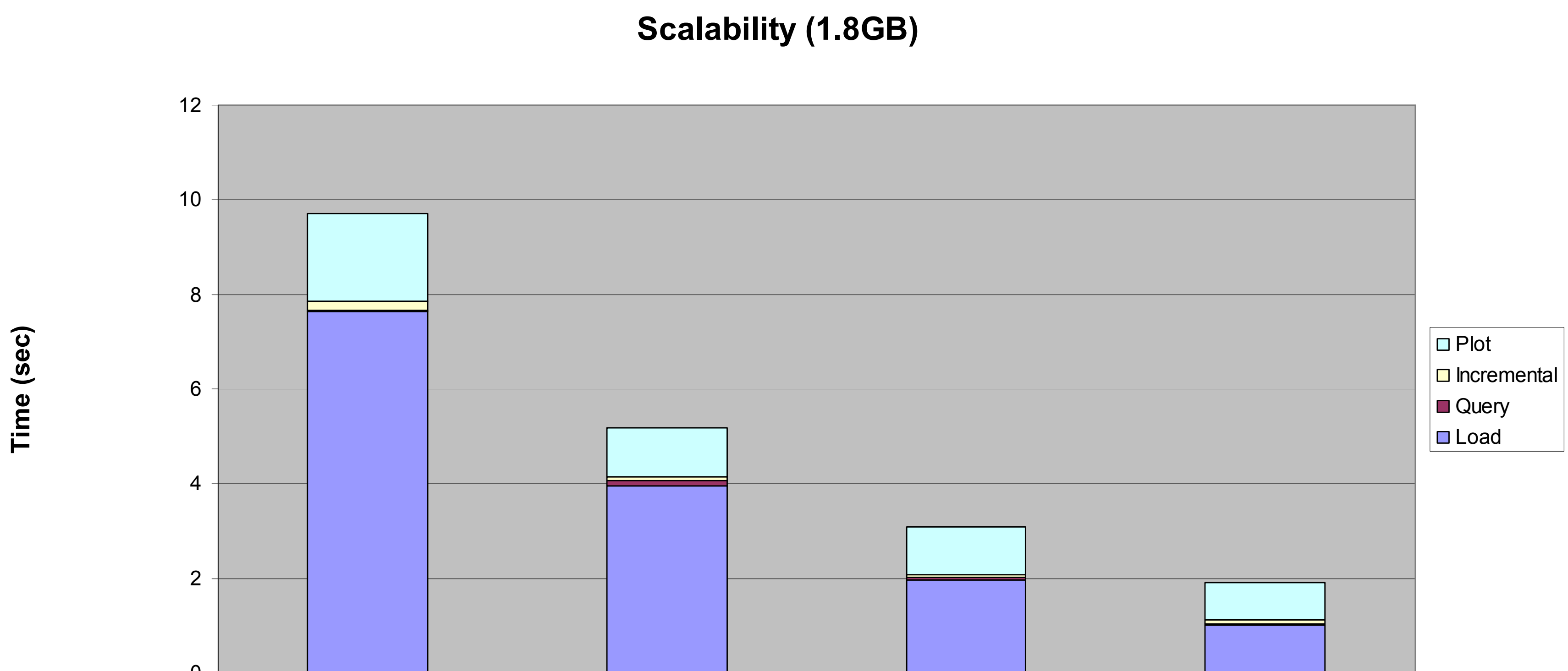
Performance and Scalability



# Processors	1	2	4	8	16	32
Plot	1.333032452	0.71183627	0.420557948	0.263743564	0.192967688	0.308394354
Incremental	0.814430177	0.450360894	0.27885984	0.231188253	0.1833327	0.178096771
Query	0.030633043	0.014426033	0.007339783	0.014834478	0.007126876	0.007275396
Load	5.716965887	2.902121382	1.503670968	0.799210361	0.40865275	0.215583072

Processors

Performance and Scalability



	4	8	16	32
Plot	1.855306729	1.039068092	0.993502174	0.789143051
Incremental	0.174567164	0.10038595	0.062899802	0.062656001
Query	0.046756749	0.105631368	0.04669781	0.042826369
Load	7.627303584	3.949373392	1.971391893	1.002135271

Processors

Conclusions and Future Work

- Analysis can be completed in parallel with good scalability
  - I/O must be considered
- Experiment with other spatial data structures
  - E.g. R-Tree based
- Explore interactive applications