High-Performance Outlier Detection Algorithm for Finding Blob-Filaments in Plasma

Lingfei Wu\textsuperscript{1}, Kesheng Wu\textsuperscript{2}, Alex Sim\textsuperscript{2}, Michael Churchill\textsuperscript{3}, Jong Y. Choi\textsuperscript{4}, Andreas Stathopoulos\textsuperscript{1}, CS Chang\textsuperscript{3}, and Scott Klasky\textsuperscript{4}

\textsuperscript{1}College of William and Mary  
\textsuperscript{2}Lawrence Berkeley National laboratory  
\textsuperscript{3}Princeton Plasma Physics Laboratory  
\textsuperscript{4}Oak Ridge National Laboratory
Outline

- Introduction
- Related work
- Blob detection
- Hybrid parallel
- Evaluations
- Conclusion
What is an outlier?

An outlier is a data object that deviates significantly from the rest of the objects, as if it were generated by a different mechanism.¹

- Outliers could be errors or noise to be eliminated
- Outliers can lead to the discovery of important information in data

Outlier detection is employed in a variety of applications:

- fraud detection
- time-series monitoring
- medical care
- public safety and security

¹Jiawei Han and Micheline Kamber, *Data Mining, Southeast Asia Edition: Concepts and Techniques*, Morgan kaufmann, 2006.
Our goal

Outlier detection is an important task in many safety critical environments.

- An outlier demands to be detected in real-time
- A suitable feedback is provided to alarm the control system
- The size of data sets need fast and scalable outlier detection methods

Our goal: apply the outlier detection techniques to effectively tackle the fusion blob detection problem on extremely large parallel machines

- Massive amounts of data are generated from fusion experiments / simulations
- Near real-time understanding of data is needed to predict performance
Blobs in fusion

What is fusion & Why fusion?

- Fusion is viable energy source for the future
- Fossil fuels will run out soon; Solar and wind have limited potential
- Advantages of fusion: inexhaustible, clear and safe
Blobs in fusion

Blobs are intermittent bursts of particles near the edge of the confined plasma
⇒ Driven by turbulence

Blobs are bad for fusion performance because they:

- Transport heat and particles away from the confined plasma
- May damage the main chamber wall
- Lead to increased levels of neutrals and impurities, bypassing control mechanisms

Blob detection is a very important task!
Big data challenges in fusion energy

Fusion experiments generate massive amounts of data:

- Diagnostics measuring lasts from a few to several hundred seconds generating large amounts of data, $\sim$ Gigabytes to Terabytes!
- Large-scale fusion simulation generates $\sim$ a few tens of Terabytes per second!
Difficulties in large-scale data analysis:

- Existing data analysis is often a single-threaded, slow, and only for post-run analysis
- Fusion experiments demand real-time data analysis
- E.g. ICEE aims to apply blob detection for monitoring health of fusion experiments in KSTAR

Real-time blob detection is a very challenging task!
Three approaches for blob detection

- **Single threshold & conditional averaging**
  - The exact criterion varies
  - Averaging may destroy important information

- **Image analysis techniques**
  - Very sensitive to the setting of parameters
  - Hard to use generic method for all images

- **Contouring method & thresholding**
  - Can not be a real-time blob detection
  - May miss detecting blobs at the edge
  - Is still post-run-analysis
An efficient blob detection approach

- **Our approach:** an outlier detection algorithm for efficiently finding blobs in fusion simulations / experiments
  - Two-step outlier detection with various criteria after normalizing the local intensity
  - Leverage a fast connected component labeling method to find blob components based on a refined triangular mesh

- **Contributions:**
  - A new method not missing detection of blobs in the edge of the region of interests compared to contouring method
  - Targeting for more challenging in-shot-analysis and between-shot-analysis
  - The first research work to achieve blob detection in a few milliseconds
Outlier detection algorithm for finding blobs

Sketch the proposed outlier detection algorithm:

- Loads raw data and normalize
- Refine mesh in region of interests
- Two-step outlier detection to identify blob candidates
- Apply minimum density criteria
- Connected component labeling to compute blobs
- Blobs if median > minimum median density criteria

Outline
- Introduction
- Related work
- Blob detection
  - Our approach
  - The sketch
  - Refine mesh
  - Two-step detection
  - Fast CCL
- Hybrid parallel
- Evaluations
- Conclusion
Refine mesh in the region of interests

- Compute 4 times more triangles by creating new vertexes with the three middle points of original edges
- Apply recursively until reaching the desired resolution
- Depend on specified data set and demanded resolution
Motivation for two-step outlier detection for finding blobs:

A contour plot in the region of interests
Two-step outlier detection to identify blobs

Apply exploratory data analysis to analyze the underlying distribution of the local normalized density:

(a) Extreme Value Distribution

(b) Log Normal Distribution

\[
\begin{align*}
N(r_i, z_i, t) - \mu > \alpha \ast \sigma, \forall (r_i, z_i) \in \Gamma, \\
N(r_i, z_i, t) - \mu_2 > \beta \ast \sigma_2, \forall (r_i, z_i) \in \Gamma_2.
\end{align*}
\]
A fast connected component labeling algorithm

We apply an efficient connected component labeling algorithm on a refined triangular mesh to find blob components:

- This is a two-pass approach and each triangle is scanned firstly
- Reduce unnecessary memory access if any vertex in a triangle is found to be connected with others
- After the label array is filled full, we need flatten the union and find tree
- Second pass is performed to correct labels and all blob candidate components are found
Parallelization of blob detection approach

A hybrid MPI/OpenMP parallelization on many-core processor architecture:

- High-level: use MPI to allocate $n$ processes to process each time frame
- Low-level: use OpenMP to accelerate the computations with $m$ threads
Results: same time frame + four planes
Results: same plane + four time frames

- Outline
  - Introduction
  - Related work
  - Blob detection
  - Hybrid parallel
  - Evaluations
    - Results I
    - Results II
    - Results III
  - Conclusion
Complete blob detection in around 2 ms with MPI/OpenMP using 4096 cores and in 3 ms with MPI using 1024 cores.

- MPI/OpenMP is two times faster than MPI.
- Linear time speedup in blob detection time and slightly more in I/O time.
Conclusion and future work

We present for the first time a real time blob detection method for finding blob-filaments in real fusion experiments or numerical simulations.

● Key components:
  ○ Two-step outlier detection with various criteria
  ○ A fast connected component labeling method
  ○ Hybrid MPI/OpenMP parallelization

● Future work:
  ○ Test the detection algorithm to experimental measurement data from operating fusion devices
  ○ Develop a blob tracking algorithm