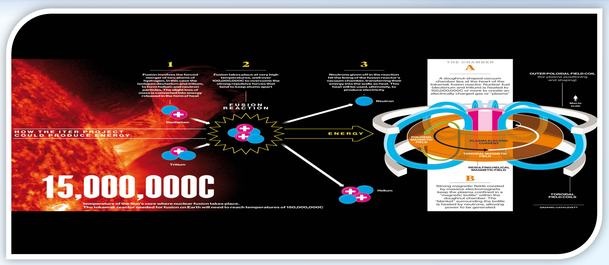
Real-Time Outlier Detection Algorithm for Finding Blob-Filaments in Plasma

Lingfei Wu, Kesheng Wu(Advisor)*, Alex Sim(Advisor)*, Andreas Stathopoulos(Advisor)
College of William and Mary, Williamsburg, Virginia, USA
*Lawrence Berkeley National Laboratory, Berkeley, California, USA

Introduction



Reprinted image from http://www.independent.co.uk/

❖ Magnetic Fusion

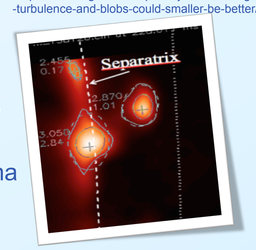
❶ What is fusion & why fusion?

✧ Fusion is a viable energy source for the future
✧ Fossil fuels will run out soon; Solar and wind have limited potential
✧ Advantages of fusion: Inexhaustible, clean, and safe

❷ Tokamak and Blobs

✧ Tokamak demands steady-state plasma confinement challenged by blobs
✧ Blobs carry high energy and plasma outside the magnetic confinement towards the wall
✧ Blobs result in loss of heat, degrading plasma confinement and erosion of the wall.

Reprinted image from http://why-sci.com/edge-turbulence-and-blobs-could-smaller-be-better/



★Blob detection is a very important task!

Motivation

❖ Fusion experiments generate massive data

✧ Three types of analysis: In-shot-analysis, between-shot-analysis, and post-run-analysis
✧ A shot lasts from a few seconds to several hundred seconds, generating ~ a few Gigabytes to a few Terabytes data sets !
✧ Large-scale simulation generates ~ 15 Terabytes per second !

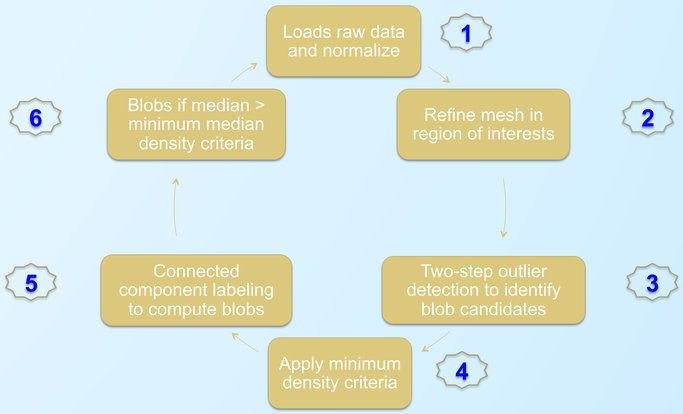
❖ Difficulty in large-scale data analysis

✧ Existing data analysis approaches are often a single-threaded, only for post-run analysis and take a long time to produce results
✧ Real fusion experiments demand real-time data analysis
✧ Example: ICEE, developed by researchers in ORNL, LBNL and PPPL, builds an in transit data processing framework for near real-time scientific applications. It aims to provide safety critical functions (such as blob detection) in KSTAR to monitor health of fusion experiments.

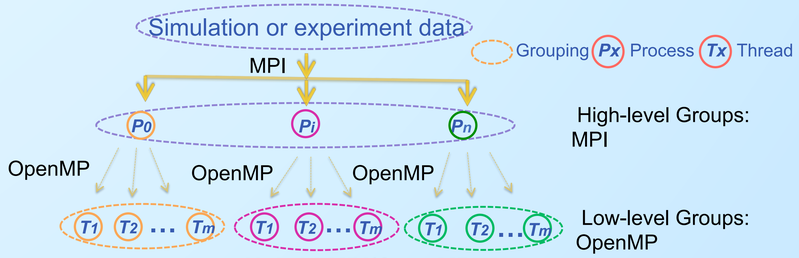
★Real-time blob detection is a very challenging task!

An Efficient Blob Detection Approach

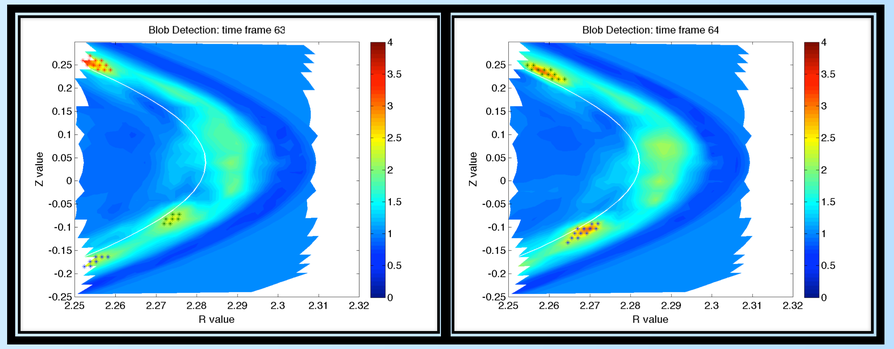
❖ Our approach: develop a real-time outlier detection algorithm for efficiently finding blobs in numerical simulations and fusion experiments



Outlier detection algorithm



Hybrid MPI/OpenMP



Example blob in two continuous time frames

❖ Outline of the new blob detection algorithm:

❶ Loads raw simulation or experimental data in each time frame and computes normalized density
❷ Refine the triangular mesh in the region of interests
❸ Apply two-step outlier detection to Identify blob candidates with appropriately chosen confidence level in the region of interests
❹ Compare the normalized density of blob candidates with the minimum density criteria to filter out unwanted candidates
❺ Apply connected component labeling to compute different blobs
❻ A blob is found if its median satisfies minimum median density criteria

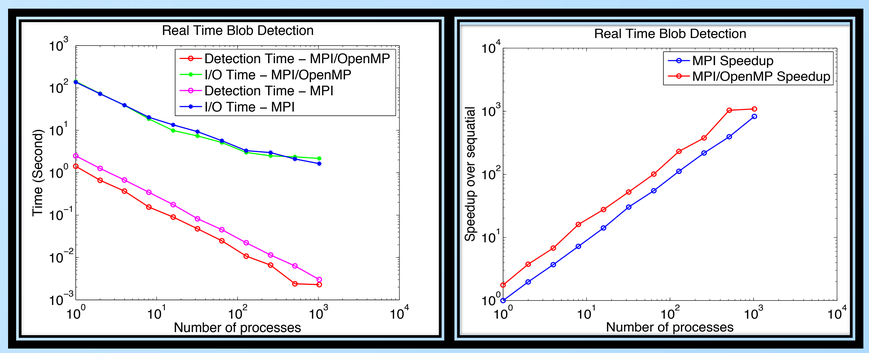
❖ A real-time blob detection approach using hybrid MPI/OpenMP:

❶ High-level: use MPI to allocate n processes to process each time frame
❷ Low-level: use OpenMP to accelerate the computations with m threads

Preliminary Results

❖ Results:

✧ We implement the proposed algorithm in C and test our implementation on NERSC supercomputer Edison with MPI and OpenMP
✧ Our data sets (33GB) is from the XGC1 simulation containing 1024 time frames which last around 2.5 milliseconds (ms)
✧ Most Encouraging results: complete blob detection in around 2 ms with MPI/OpenMP using 4096 cores and in 3 ms with MPI using 1024 cores
✧ MPI/OpenMP implementation is two times faster than MPI implementation
✧ Linear time scalability in blob detection time and slightly more in I/O time
✧ MPI and MPI/OpenMP achieve speedup over serial up to 800x and 1200x
✧ We are integrating it into ICEE and plan to test it in the KSTAR experiments
✧ We also plan to develop a blob tracking algorithm based on this work.



Blob detection performance

This work was supported by DOE under Contract No. DE-AC02-05CH11231, and partially supported by NSF under a grant No. CCF 1218349 and by DOE under a grant No. DE-FC02-12ER41890.

I thank Suren Byna, Houjun Tang, Xiaocheng (Chris) Zou from LBNL, and collaborators from PPPL and ORNL for their help and feedbacks.