

Graph-Based Flow Visualization Algorithms

Han-Wei Shen
Computer Science and Engineering
The Ohio State University



SDAV

Scalable Data Management, Analysis, and Visualization



The Challenges in Computational Sciences

- In a few years, exaFLOPs supercomputers will become a reality (exa = 1,000,000,000,000,000,000)
 - Number of cores per processor will increase
 - Memory per core will decrease
- The speed and capacity of memory and I/O devices cannot keep pace with the increase of compute power
 - Cost of moving data will increase
- It will be very difficult for scientists to store and analyze even a small portion of their simulation output

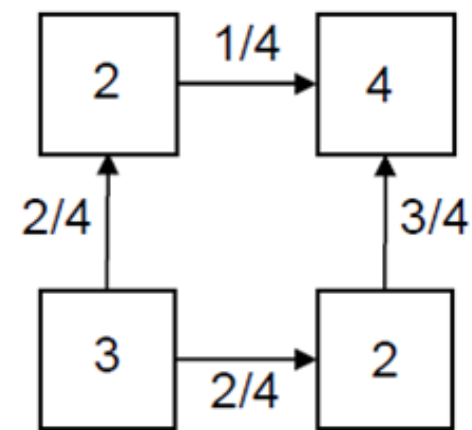
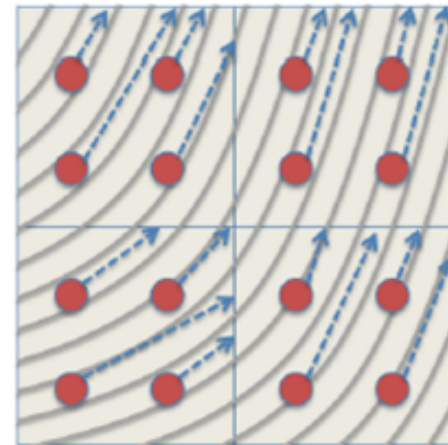


Model Global Flows from Local Computation

- Global structures of flow fields are more expensive to compute because of high data movement cost
 - Computation and bundling of long streamlines
 - How different local regions are connected by the flow
 - Where uniformly distributed particles will go
- There is still a need to reconstruct the global information, ideally from local computation
- Global flow directions can be used to:
 - Workload estimation
 - Data access pattern analysis
 - Guide user interaction

Flow Graphs

- Estimate global flow patterns using only local computation
- Flow graphs:
 - Each block is a node
 - Neighboring blocks have edges between them
 - Weight of edge is probability that a seed goes from one block to the other
 - A transition matrix can be constructed from the graph





SDAV

Scalable Data Management, Analysis, and Visualization

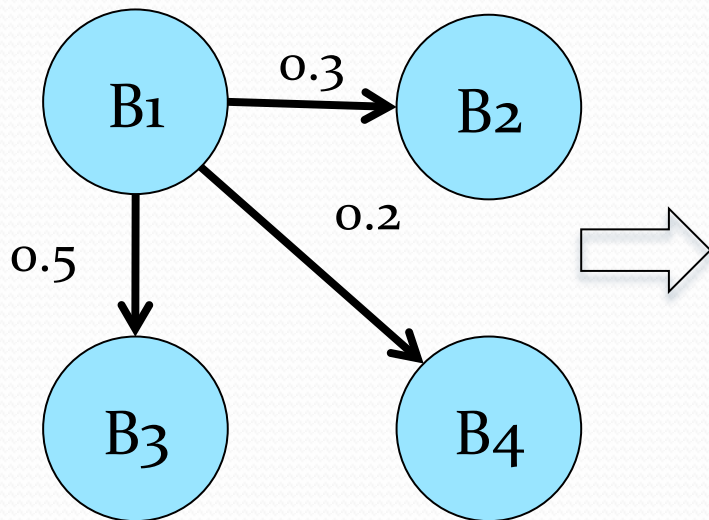


High Performance Flow Line Computation via Flow Graphs

- Flow graphs can be used to facilitate efficient flow lines computation
 - Parallel streamline computation
 - Out-of-core streamline computation
- Parallel flow lines computation
 - Workload estimation for load balancing
- Out-of-core flow lines computation
 - Data access pattern analysis for effective pre-fetching

Transition Matrix from Flow Graphs

- Remember each edge in the graph represents the probability (percentage) of particles from one node to the other
- A transition matrix M can be constructed from the graph



$$\begin{matrix} & \begin{matrix} B1 & B2 & B3 & B4 \end{matrix} \\ \begin{matrix} B1 \\ B2 \\ B3 \\ B4 \end{matrix} & \left[\begin{array}{cccc} - & 0.3 & 0.5 & 0.2 \\ \dots & - & \dots & \dots \\ \dots & \dots & - & \dots \\ \dots & \dots & \dots & - \end{array} \right] \end{matrix} = M$$

Estimate the Particle Distribution

- Assuming initially the particles are distributed among the blocks $B_1 \dots B_m$ as $PL = [L_1, L_2, L_3, L_4, \dots L_m]$
- Then the numbers of particles in each block after K rounds are equal to $PL' = PL \times M^K$

One round example

$$PL' = [L_1, L_2, L_3, L_4, \dots L_m] * \begin{matrix} & \begin{matrix} B_1 & B_2 & B_3 & B_4 \end{matrix} \\ \begin{matrix} B_1 \\ B_2 \\ B_3 \\ B_4 \end{matrix} & \begin{bmatrix} - & 0.3 & 0.5 & 0.2 \\ .. & - & ... & ... \\ .. & ... & - & ... \\ .. & ... & ... & - \end{bmatrix} \end{matrix}$$

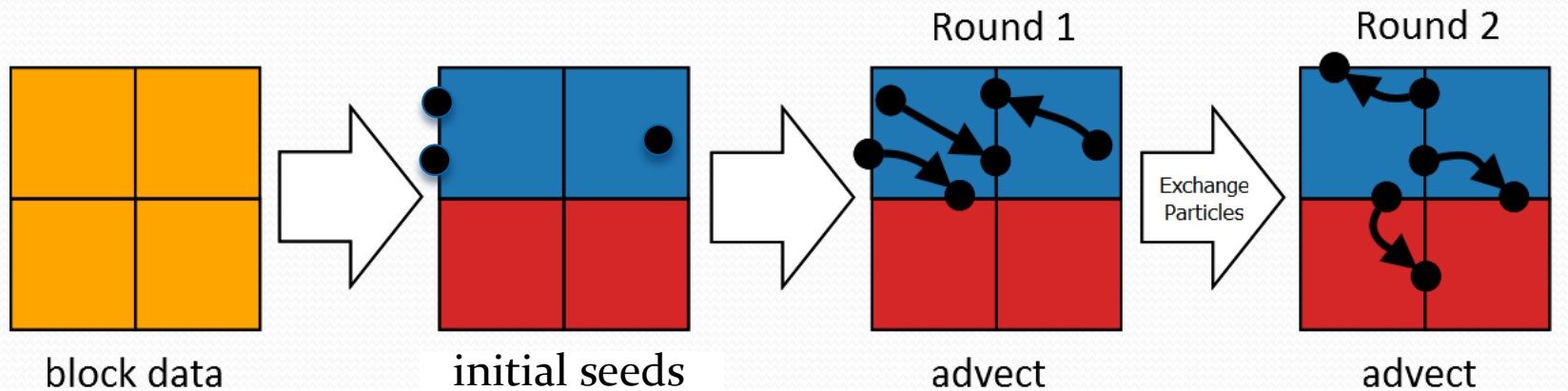


SDAV

Scalable Data Management, Analysis, and Visualization

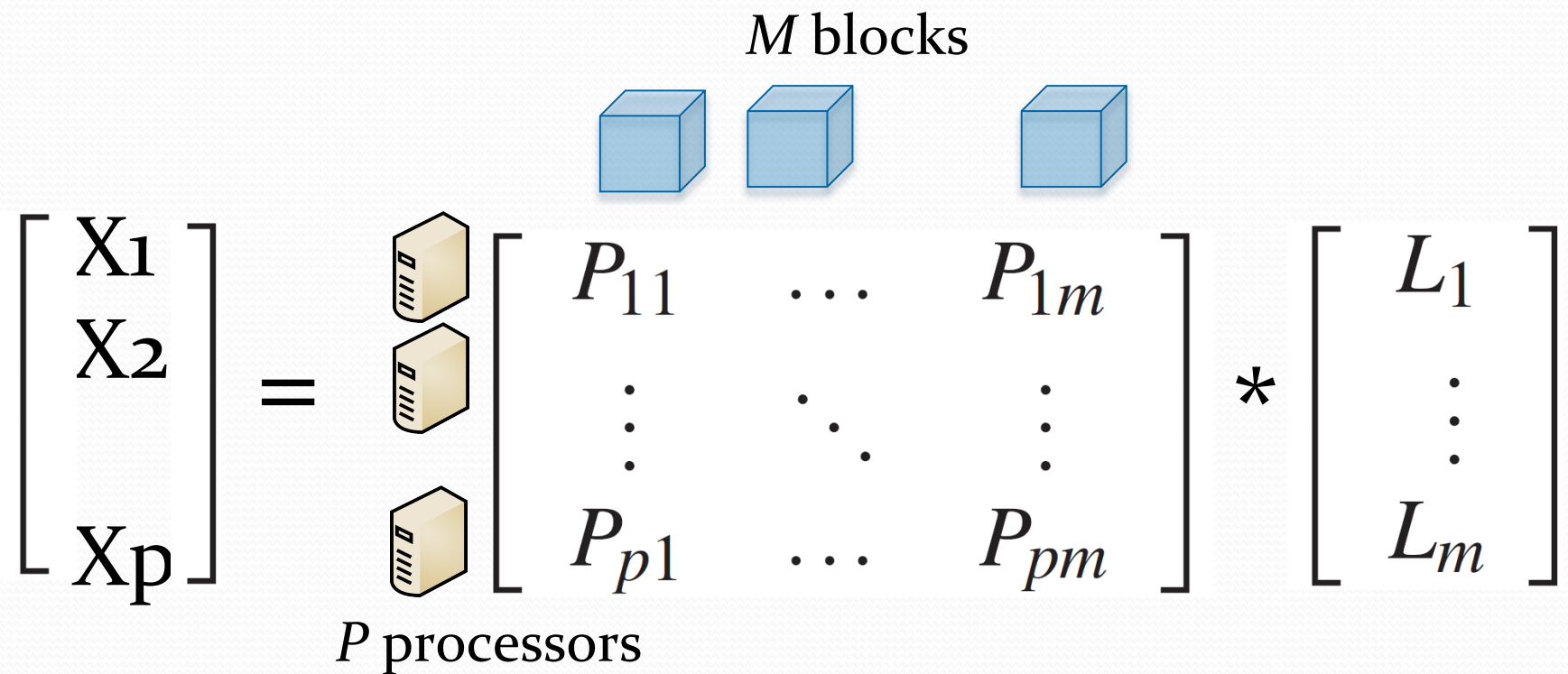


Parallel Particle Tracing in Rounds



Partition Matrix and Load Distribution

M blocks



$$\begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_p \end{bmatrix} = \begin{matrix} \text{P}_1 \text{ processor} \\ \text{P}_2 \text{ processor} \\ \vdots \\ \text{P}_p \text{ processor} \end{matrix} \begin{bmatrix} P_{11} & \dots & P_{1m} \\ \vdots & \ddots & \vdots \\ P_{p1} & \dots & P_{pm} \end{bmatrix} * \begin{bmatrix} L_1 \\ \vdots \\ L_m \end{bmatrix}$$

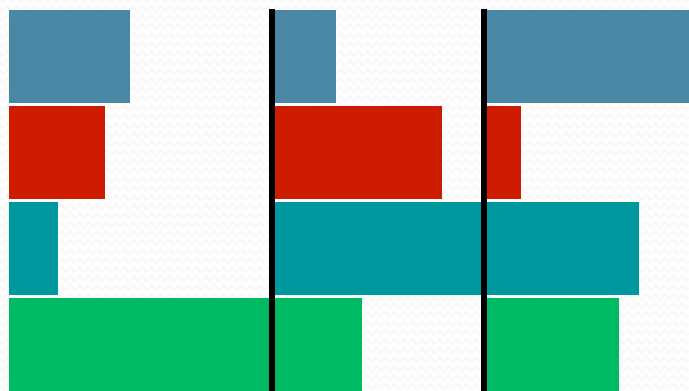
P processors

workload ratio P_{ki} : percentage of work in block i done by processor k

Load Imbalance Cost Function

- Cost function
 - Based on the *variance* of the process workloads
- Why variance?
 - A quadratic form of the unknowns

$$\text{VAR}(x_1 \dots x_p) \frac{1}{p} \sum_{k=1}^p x_k^2 - \left(\frac{1}{p} \sum_{k=1}^p x_k \right)^2$$

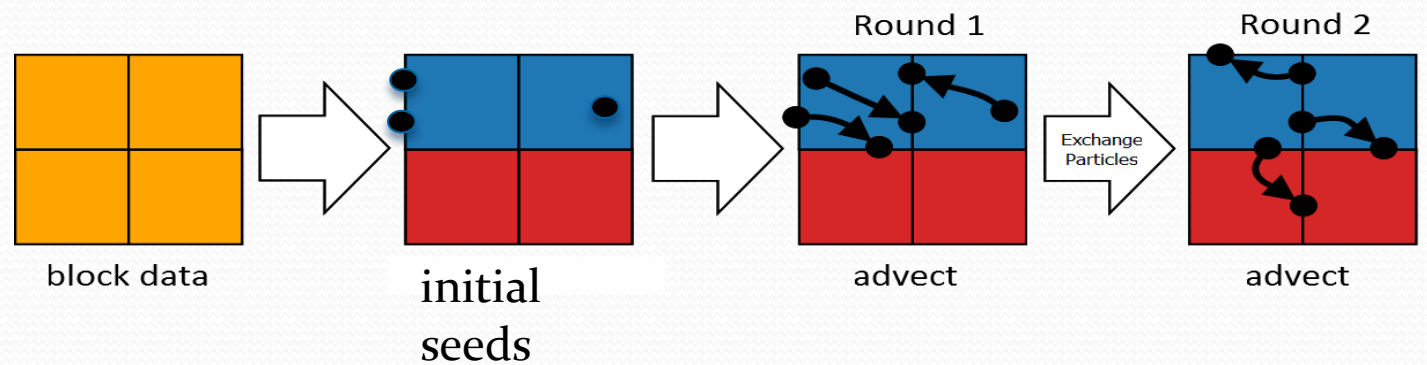


$$\text{VAR}(R_1) + \text{VAR}(R_2) + \text{VAR}(R_3) + \dots$$

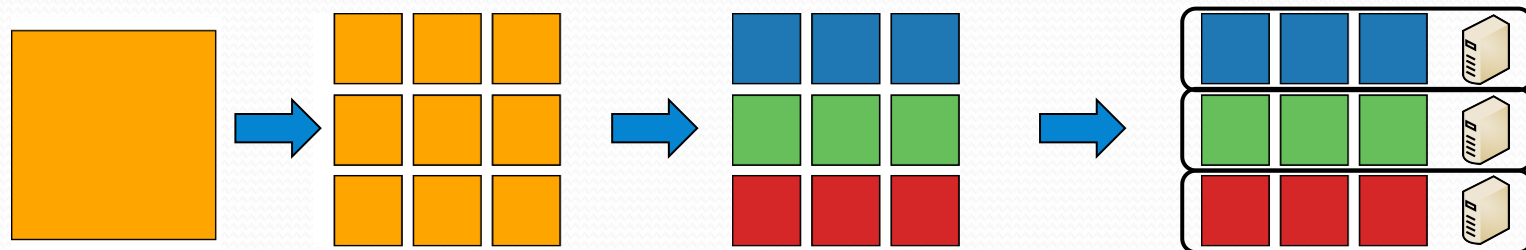


$$\text{VAR}(R_1) + \text{VAR}(R_2) + \text{VAR}(R_3) + \dots$$

Parallel Particle Tracing in Rounds



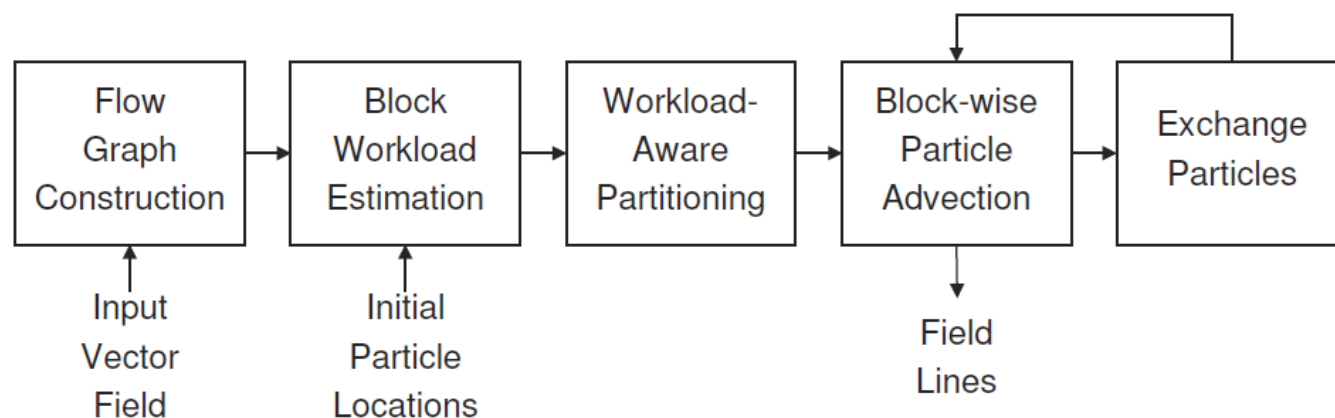
↓ Solve P_{Ki}



Divide the data into the blocks, and partition the blocks into processors

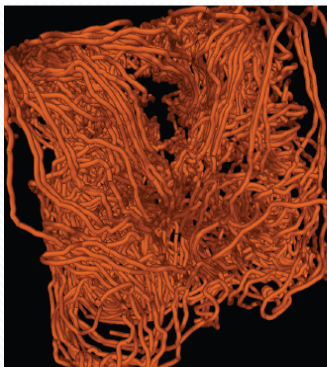
Load Balancing for Parallel Flow line Computation

- Based on the workload estimation, divide the data blocks into N processors such that:
 - The workload variance among the processors in each round of the particles keeps as small as possible
 - Minimize the sum of the workload variance in all rounds
- A quadratic programming problem to get P_{ki}

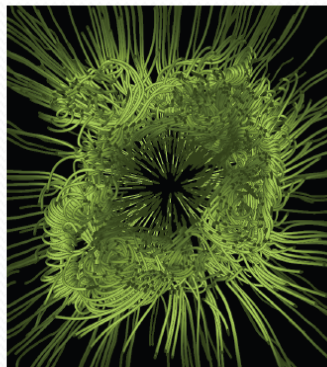




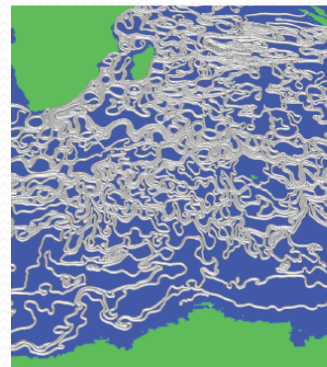
Timing Tests Setup



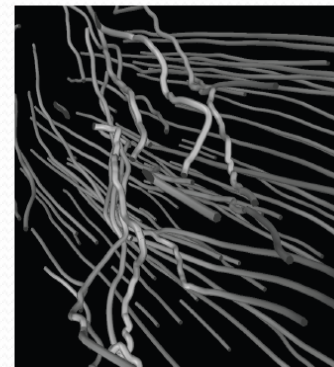
Nek



Plume



Ocean



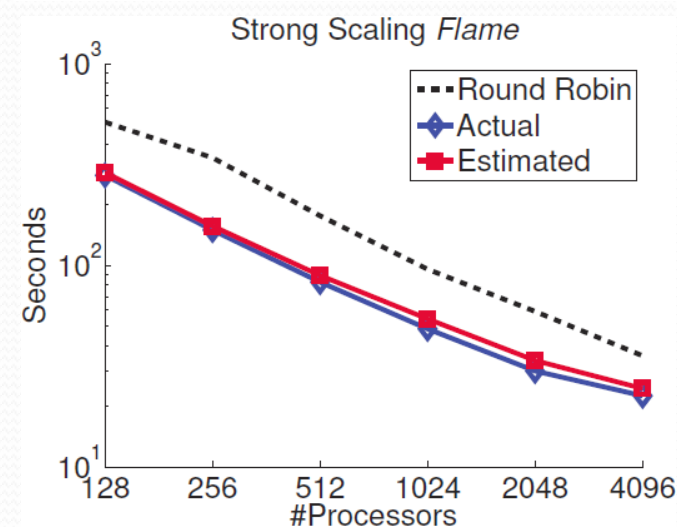
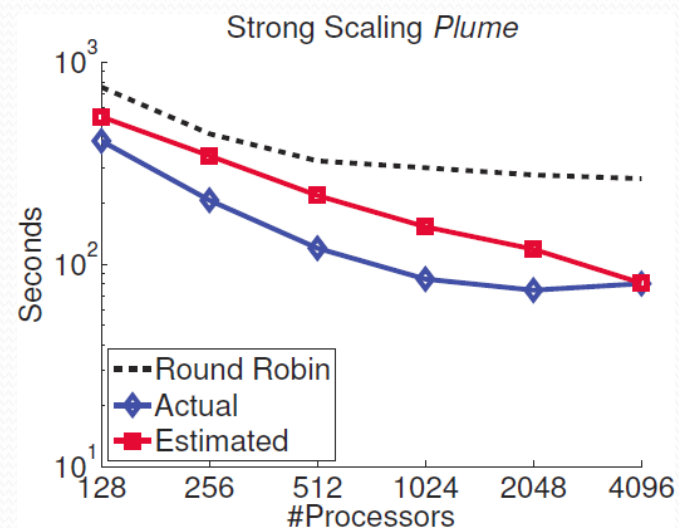
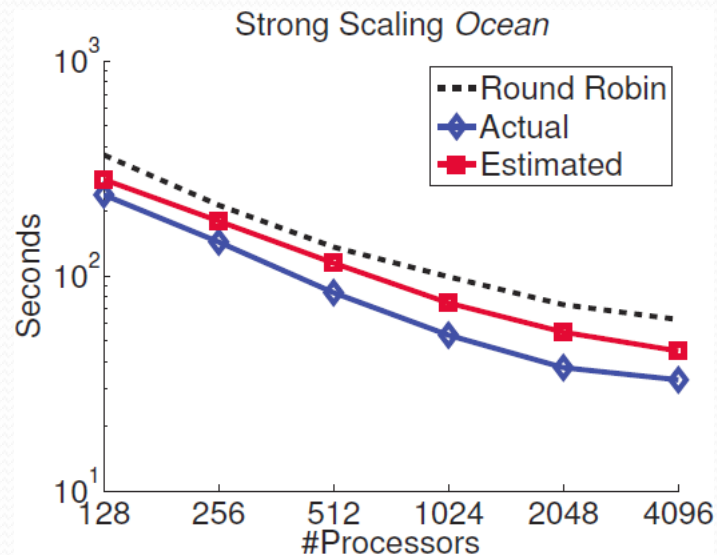
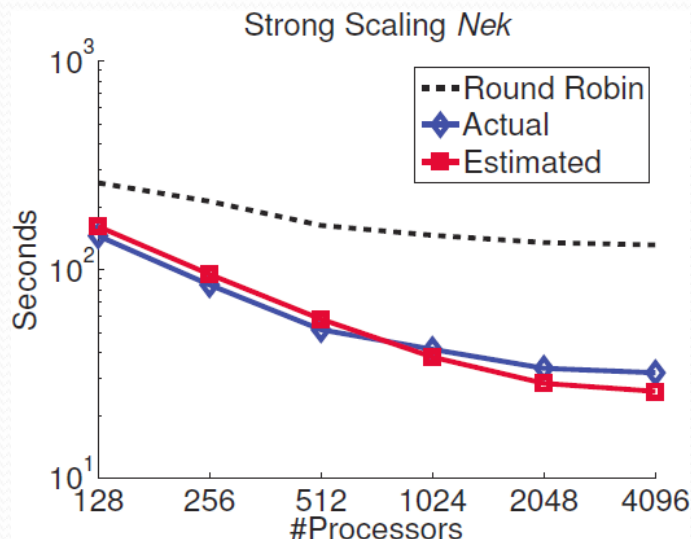
Flame

- Test Machine: *Surveyor*
 - IBM Blue Gene/P
 - 4096 cores
- Strong scaling tests
- Seed evenly throughout domain
- Compare with actual workload and round robin

Dataset	# Particles	
	Size (GB)	(Strong)
<i>Nek</i>	12	256K
<i>Plume</i>	5.81	256K
<i>Ocean</i>	3.86	256K
<i>Flame</i>	18.69	128K



Strong Scaling Results





SDAV

Scalable Data Management, Analysis, and Visualization



Flow Graphs Query

Provide an abstract view of the flow fields

- Explore and filter nodes based on flow attributes and complexity measures
 - Velocity magnitude, entropy, fractal dimension, and/or other scalar quantities, degree, centrality, etc
- Explore the relations between the graph nodes
 - Edge weight
 - Subgraphs connected to a node
 - Clusters of graph nodes

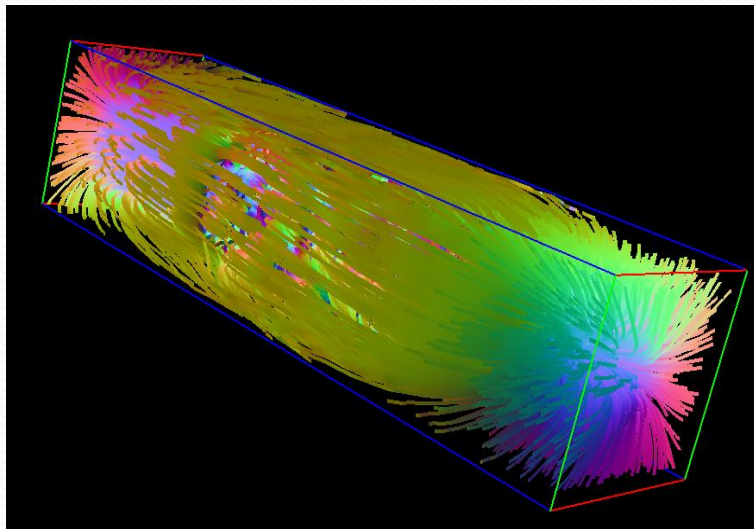


SDAV

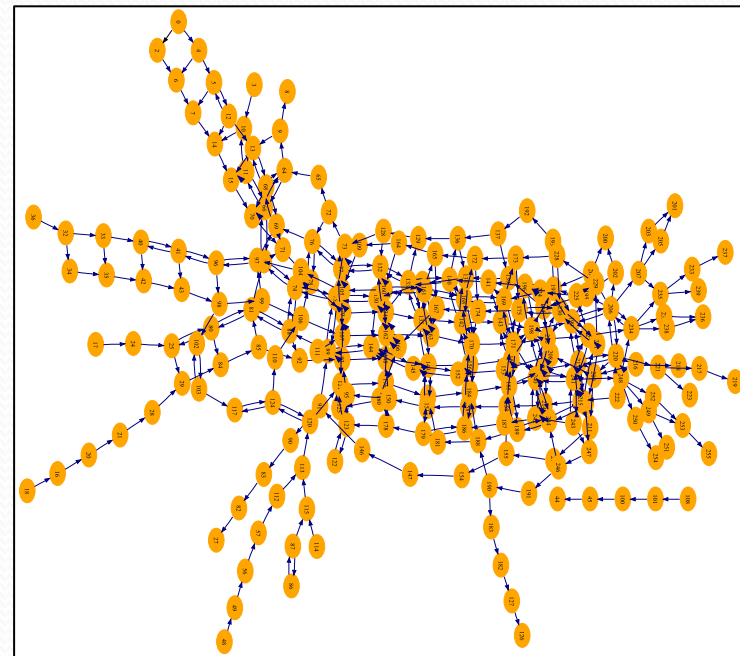
Scalable Data Management, Analysis, and Visualization



Flow Graph Example



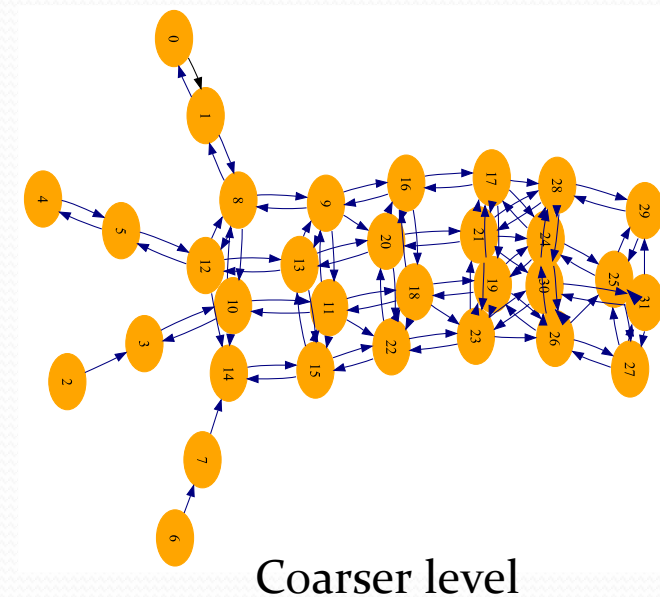
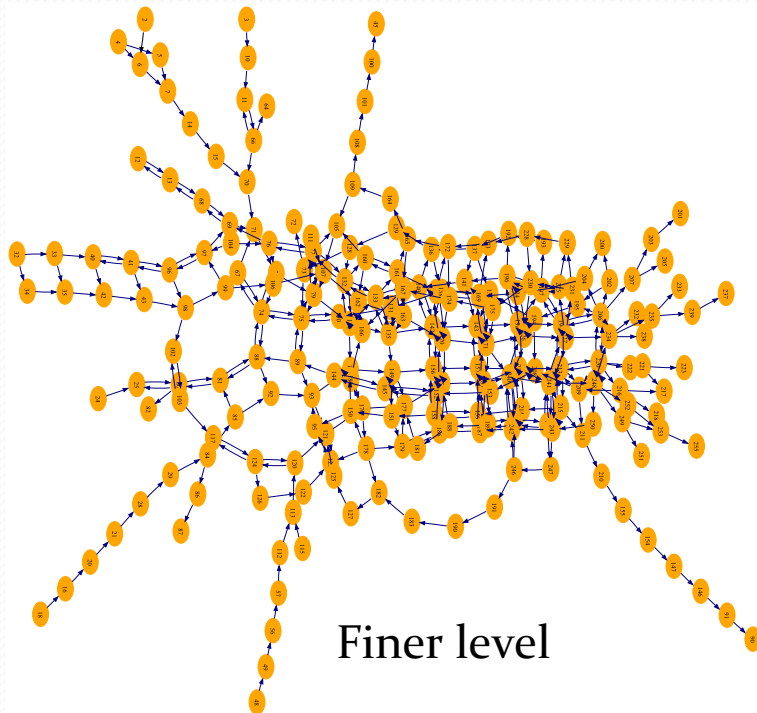
Streamline View



The Flow Graph

Flow Graph Hierarchy

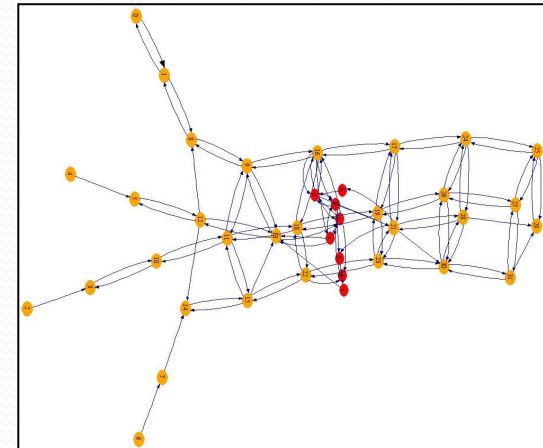
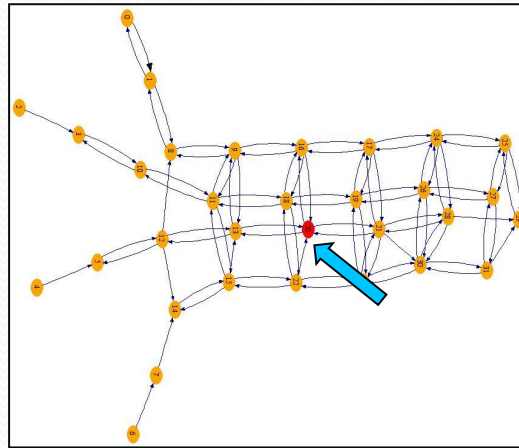
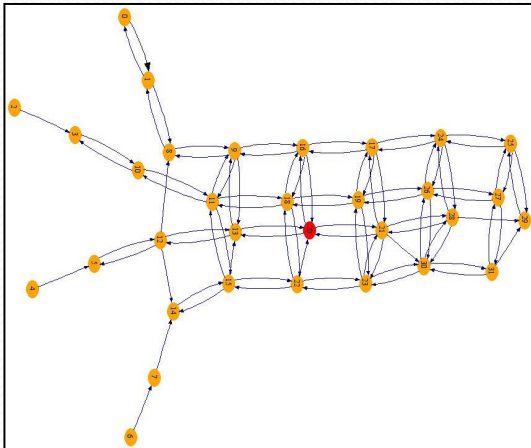
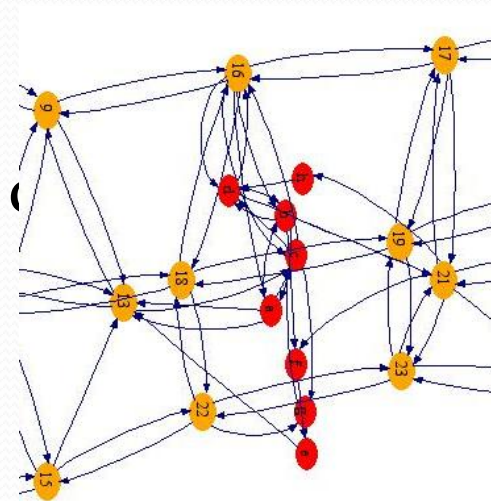
- Hierarchical viewing by merging higher resolution nodes



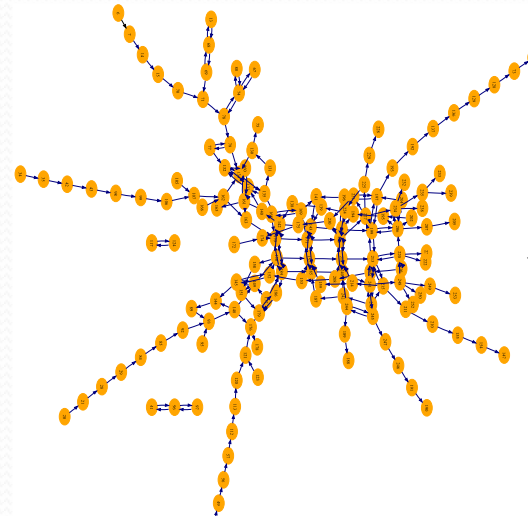
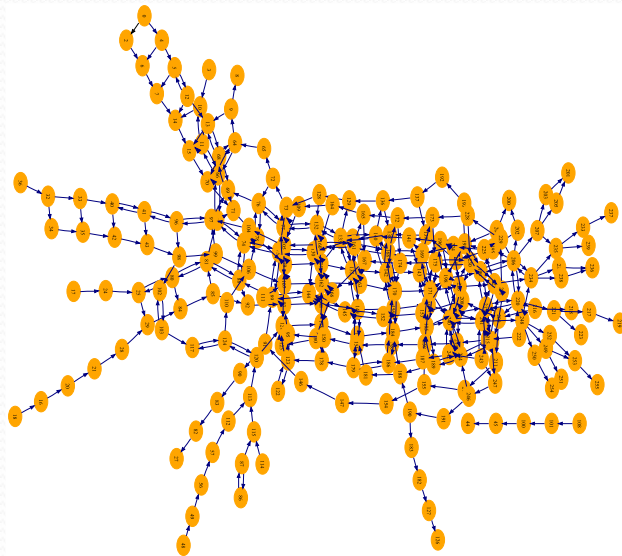
Edge weights are summed together

Flow Graph Hierarchy

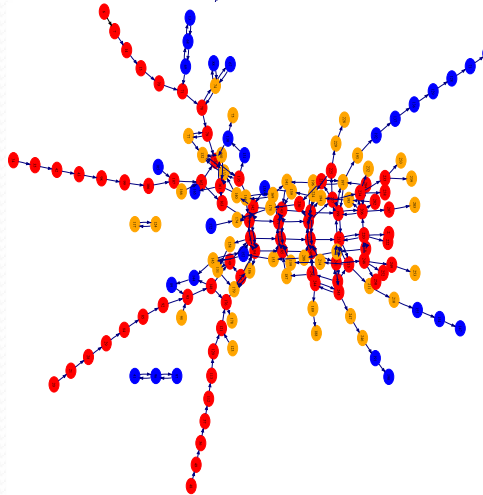
- Interactive manipulation – open and close graph nodes at different resolutions



Query by Velocity Magnitude



Filter by
velocity magnitude



Color by
velocity magnitude

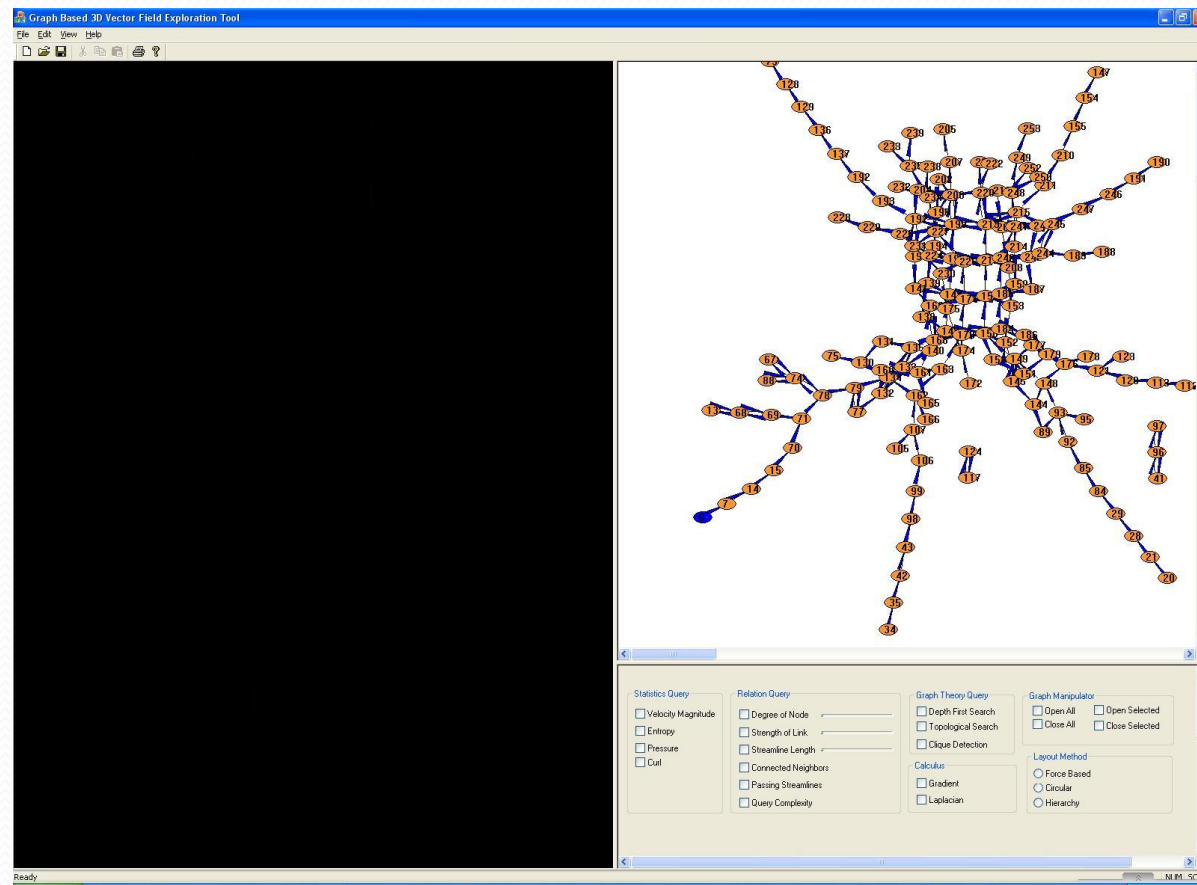


SDAV

Scalable Data Management, Analysis, and Visualization

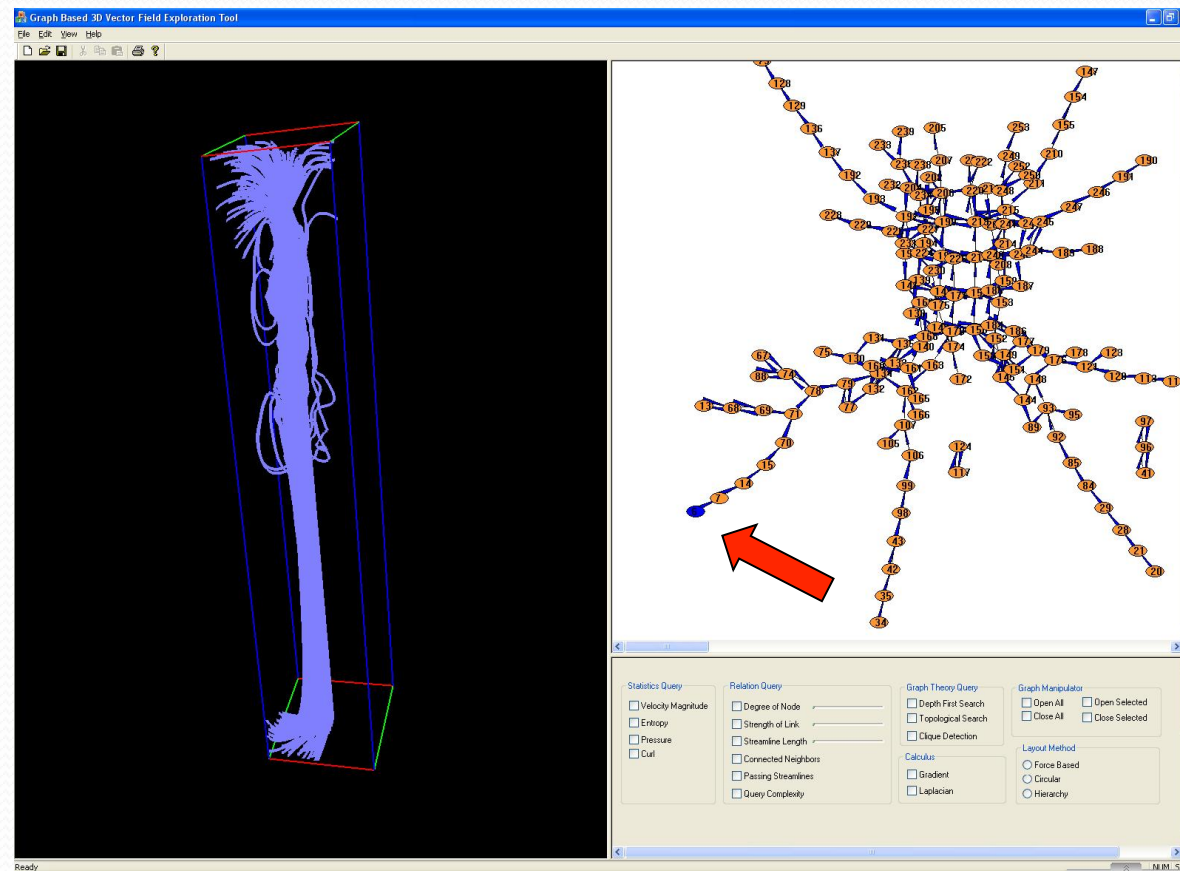


Streamlines Query





Streamlines Query



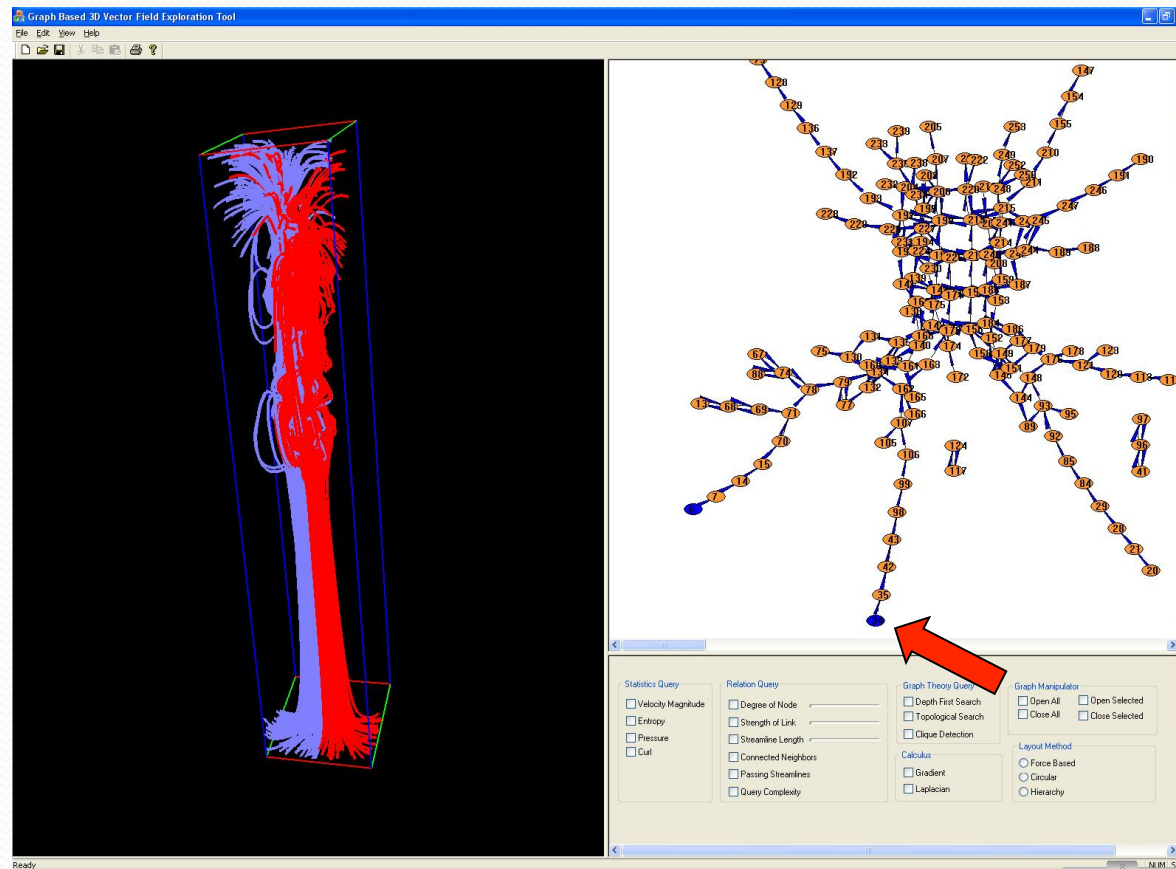


SDAV

Scalable Data Management, Analysis, and Visualization

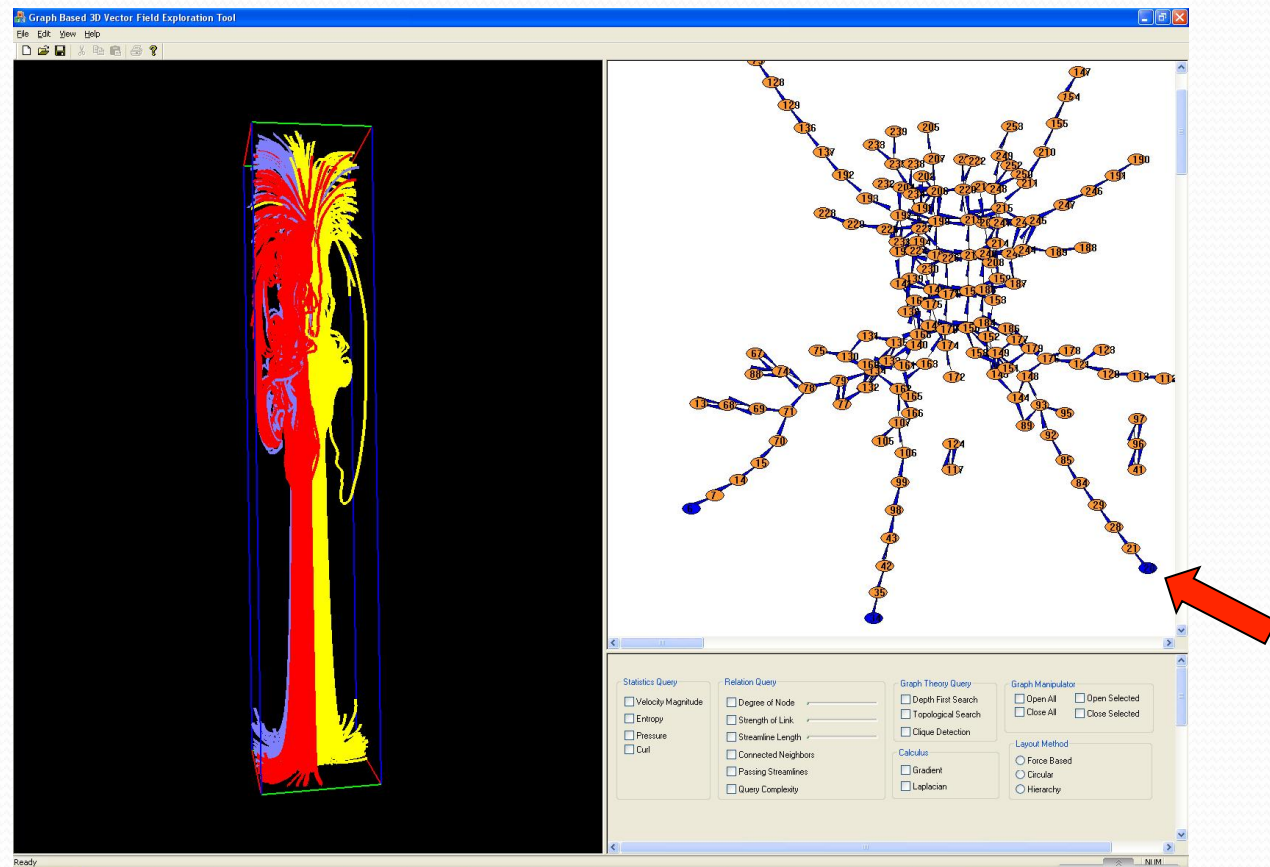


Streamlines Query





Streamlines Query





SDAV

Scalable Data Management, Analysis, and Visualization



I/O Optimization via Flow Graphs

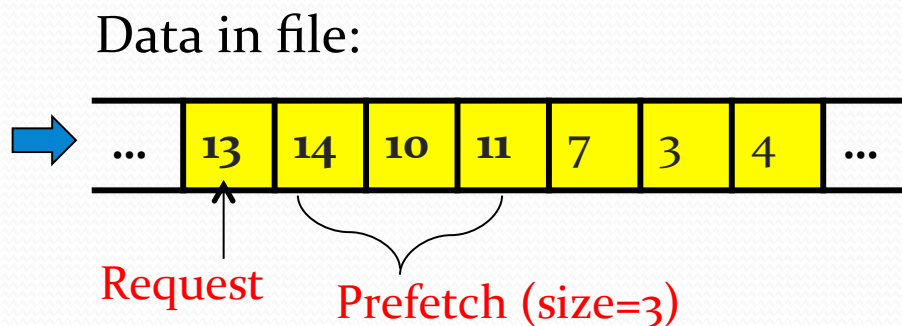
- Disk I/O can become a major bottleneck for large scale out of core flow line computation
- To reduce the I/O overhead
 - Minimize the latency
 - Increase data reference locality
 - Hide the data transfer time
- Approach
 - Divide the data into small blocks
 - Follow the access pattern and organize the blocks into the file
 - Perform pre-pretching



Data Pre-fetching for Flow Line Computation

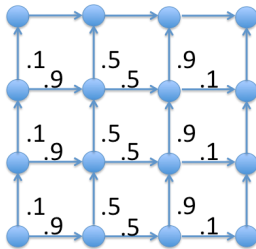
- Prefetch
 - Load multiple blocks at a time from the file
 - Overlap data I/O with flow line computation
- Research goal
 - Find a file layout that best facilitates data prefetching
 - Layout: a linear order of blocks stored in the file
 - Minimize the miss rate: the ratio of needed data blocks not in the prefetched pool

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16



Layout Algorithm Overview

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

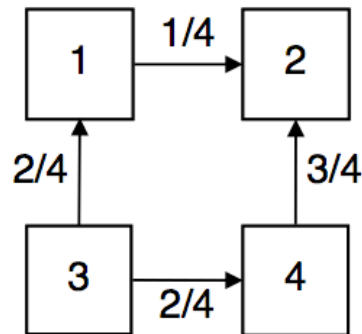
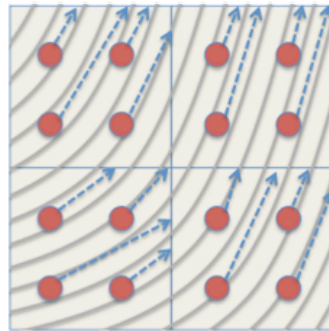


f



1. Divide the flow field into small blocks
2. Analyze the access dependencies of blocks using a graph representation of the flow field
3. Formulate the cost function using the graph
4. Optimize the file layout

Access Dependency Graph

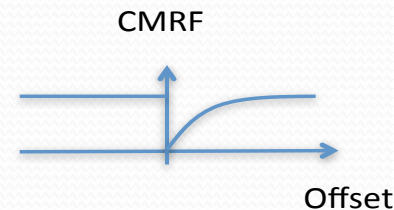


Model the runtime dependencies as a directed graph

- **Node:** data block
- **Directed edge:** seeds moving from one block to a neighboring block
- **Edge weight:** conditional probability: $w(u \rightarrow v) = \Pr(v|u)$

Evaluating File Layout

- Input
 - access dependency graph $G(V,E)$
 - Block layout $L(u)$: position of block U in a file
- Prefetch miss rate if block u and v are needed in sequence
 - $\text{Miss_rate}(u,v) = \Pr(v|u) * \text{CMRF}(L(v)-L(u))$
 - CMRF: cache miss rate function



- Cost of a layout:
$$\text{Cost}(L,G) = \frac{1}{|V|} \sum_{(u,v) \in G} \text{Miss_rate}(u,v)$$



SDAV

Scalable Data Management, Analysis, and Visualization

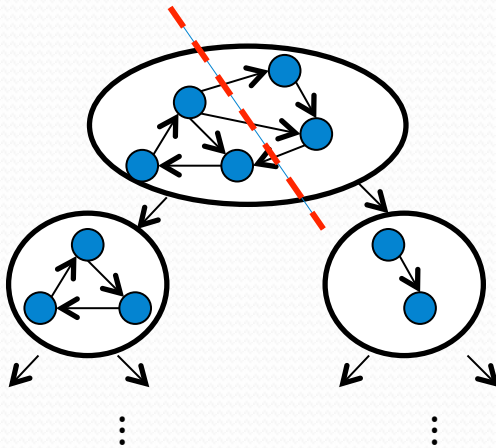


Layout Optimization

- Cost function: family of **minimum linear arrangement** problem
 - NP-Hard
- Recursive approximation algorithm:

Layout Optimization

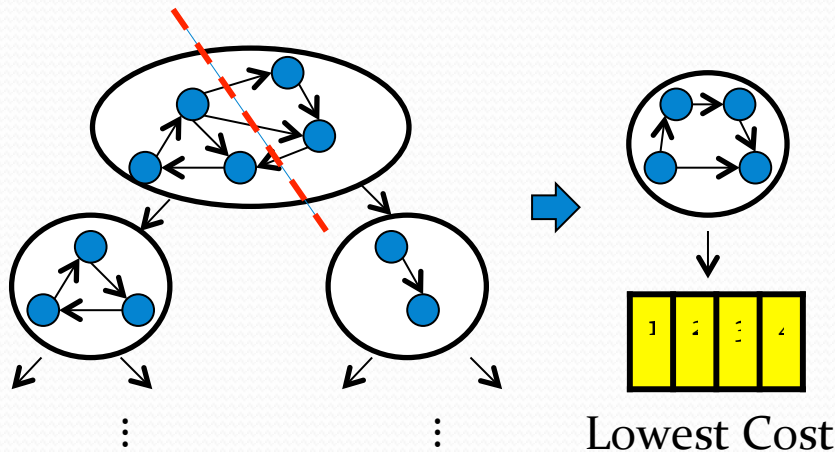
- Cost function: family of **minimum linear arrangement** problem
 - NP-Hard
- Recursive approximation algorithm:



1. Recursively find the balanced min-cut of the graph

Layout Optimization

- Cost function: family of **minimum linear arrangement** problem
 - NP-Hard
- Recursive approximation algorithm:

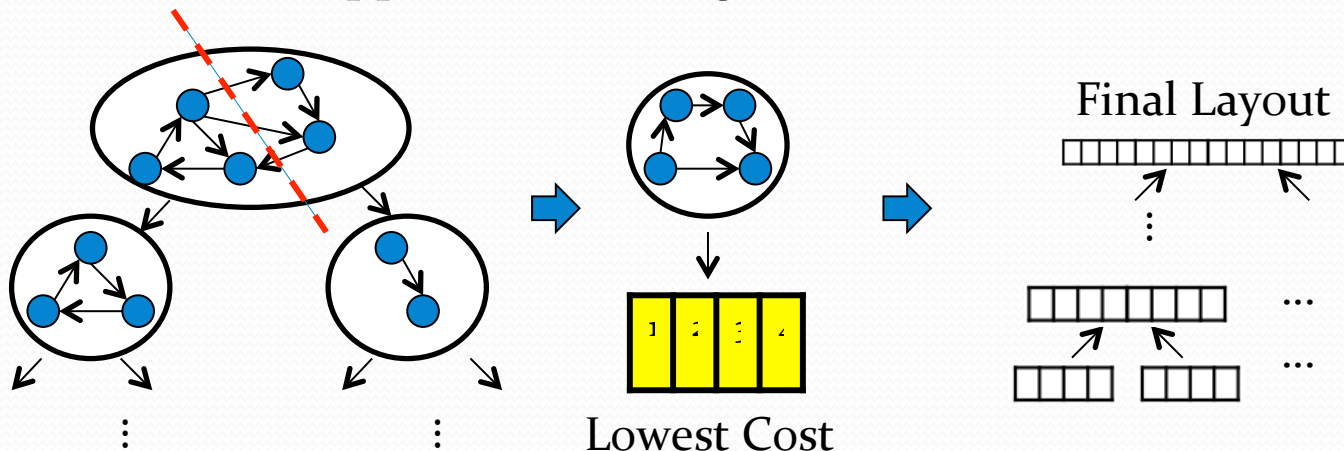


1. Recursively find the balanced min-cut of the graph

2. Exhaustively search for an optimal layout when the subgraph has ≤ 4 nodes

Layout Optimization

- Cost function: family of **minimum linear arrangement** problem
 - NP-Hard
- Recursive approximation algorithm:



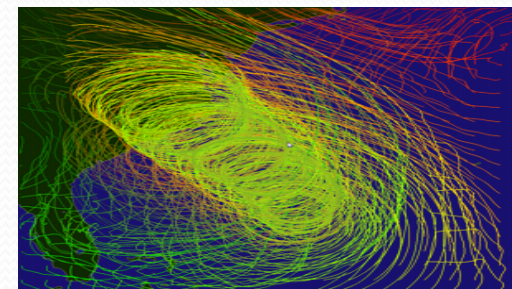
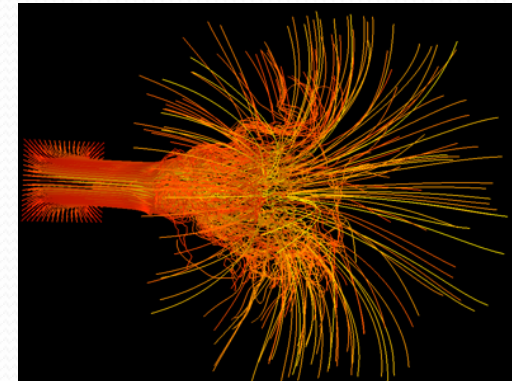
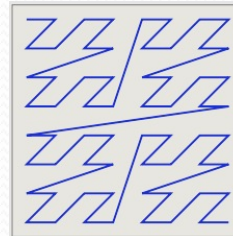
1. Recursively find the balanced min-cut of the graph

2. Exhaustively search for an optimal layout when the subgraph has ≤ 4 nodes

3. Bottom up: Concatenate two partial layouts with lower cost

Performance Evaluation

- Experiment platform: a multithreaded pathline computation system on a 4-core desktop computer with 2GB of memory
- Comparison: Z space-filling-curve layout
- Our layouts vs. Z-curve layout
 - Ours have smaller miss rate
 - Ours require less I/O time
 - **40%** time reduction (compared to Z-curve layout)
- By increasing the prefetch size,
 - The number of disk accesses decreases
 - The I/O time decreases first but then increases
 - ✓ Prefetch size 3 - 7 blocks works better





SDAV

Scalable Data Management, Analysis, and Visualization



Publications

- Flow Graphs for Interaction
 - Lijie Xu and Han-Wei Shen. Flow web: A graph based user interface for 3d flow field exploration. In ***Proceedings of IS&T/ SPIE Visualization and Data Analysis 2010***
- Parallel Flow Line Computation
 - Boonthanome Nouanesengsy, Teng-Yok Lee, Kewei Lu, Han-Wei Shen, Tom Peterka, Parallel Particle Advection and FTLE Computation for Time-Varying Flow Fields, **ACM SC'12**
 - Nouanesengsy, Boonthanome; Lee, Teng-Yok; Shen, Han-Wei, Load-Balanced Parallel Streamline Generation on Large Scale Vector Fields, In: **IEEE Visualization 2011**



SDAV

Scalable Data Management, Analysis, and Visualization



Publications

- Out-of-Core Flow Line Computation with I/O optimization
 - Chun-Ming Chen, Lijie Xu, Teng-Yok Lee, Han-Wei Shen, A Flow Guided Layout for Out-of-Core Streamline Computation, IEEE Pacific Visualization 2012
 - Chun-Ming Chen, Boonthanome Nouanesengsy, Teng-Yok Lee, Han-Wei Shen, Flow-Guided File Layout for Out-of-core Pathline Computation, IEEE symposium on Large Data Analysis and Visualization (LDAV) 2012
 - Chun-Ming Chen and Han-Wei Shen, Graph-based Seed Scheduling for Out-of-core FTLE and Pathline Computation, **IEEE Symposium on Large Data Analysis and Visualization 2013**



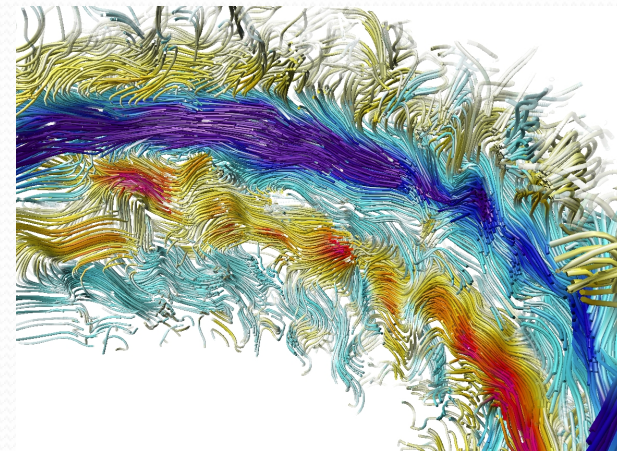
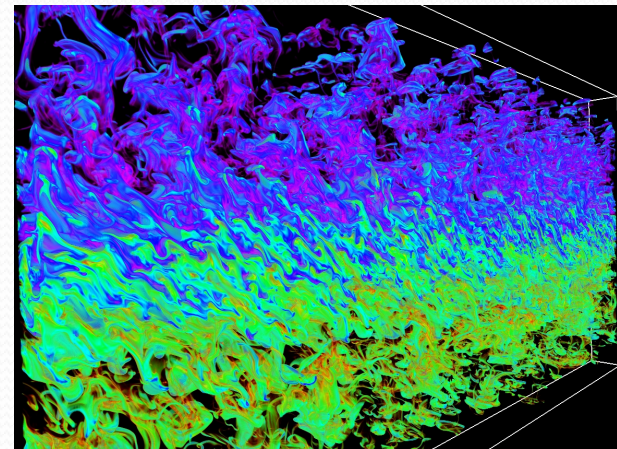
SDAV

Scalable Data Management, Analysis, and Visualization



OSUFlow Library

- As a result of an NSF grant, OSU began to construct a flow visualization library since 2004
- In 2005, the core of OSUFlow Vis library was adopted by NCAR's visualization software VAPOR and released to the turbulence research community via SourceForge (more than 1000 downloads so far)
- In 2008, OSU and Argonne began to extend the library to run on DOE's leadership computing facility
- Currently more than 25,000 lines of code





SDAV

Scalable Data Management, Analysis, and Visualization



Current Capability

- 3D static and time-varying flow lines (streamlines, pathlines, streaklines, etc) generation
- Regular, Irregular, AMR grid
 - Irregular and AMR development are on-going
- Domain decomposition that allows particles to be traced independently in each sub-domain
- Parallel particle tracing in time-varying fields on DOE's leadership computing facility
- Collective I/O optimization
- Allow researchers to experiment different optimization ideas related to large scale parallel processing



SDAV

Scalable Data Management, Analysis, and Visualization



Software

- OSUFlow Software Repo:

<https://svn.mcs.anl.gov/repos/osufLOW>

- Questions?



SDAV

Scalable Data Management, Analysis, and Visualization



Thank You!