Subway: Minimizing Data Transfer during Out-of-GPU-Memory Graph Processing

Amir Hossein Nodehi Sabet, Zhijia Zhao, Rajiv Gupta

Computer Science and Engineering
UC Riverside

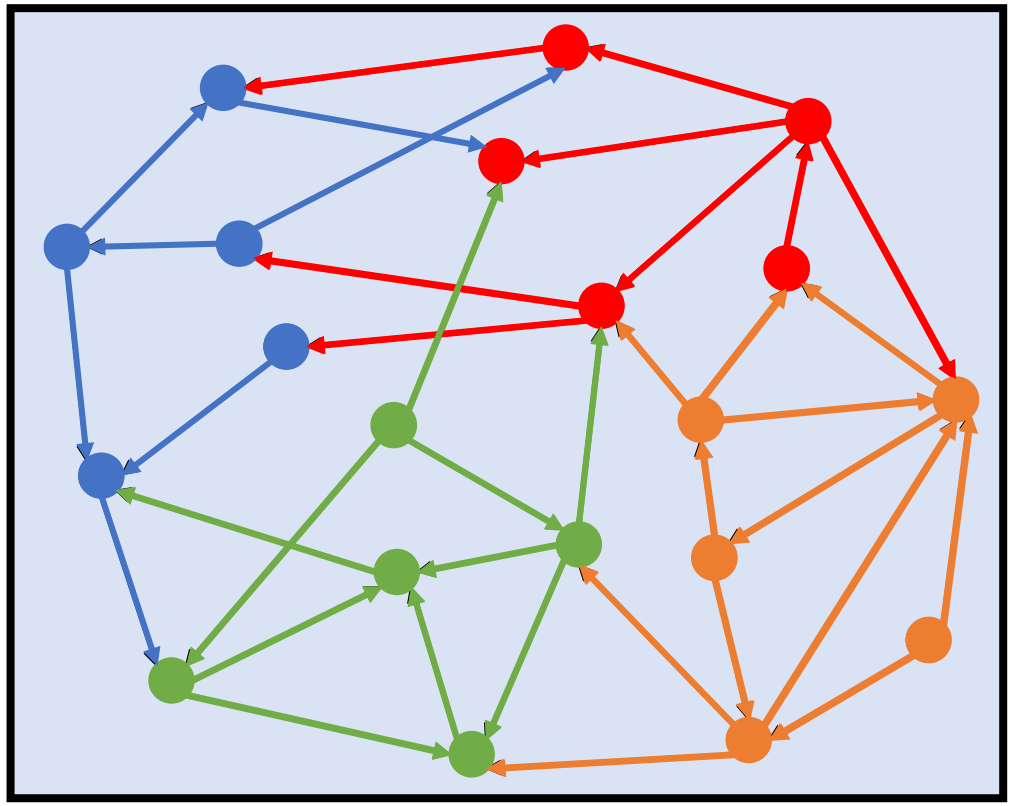
Background and Motivation

- GPUs enable massive parallelism for graph processing
  - CuSha [1]
  - Gunrock [2]
  - Tigr [3]
  - …
- Graphs can be large and tend to grow over time
  - Web graphs
  - Social networks
- But GPU memory is limited!!
  - Out-of-GPU-Memory Graph Processing

[1] Khorasani, Farzad, et al. "CuSha: vertex-centric graph processing on GPUs." HPDC'14
[2] Wang, Yangzihao, et al. "Gunrock: A high-performance graph processing library on the GPU." PPoPP'16
[3] Nodehi Sabet, Amir Hossein, Junqiao Qiu, and Zhijia Zhao. "Tigr: Transforming irregular graphs for gpu-friendly graph processing." ASPLOS'18

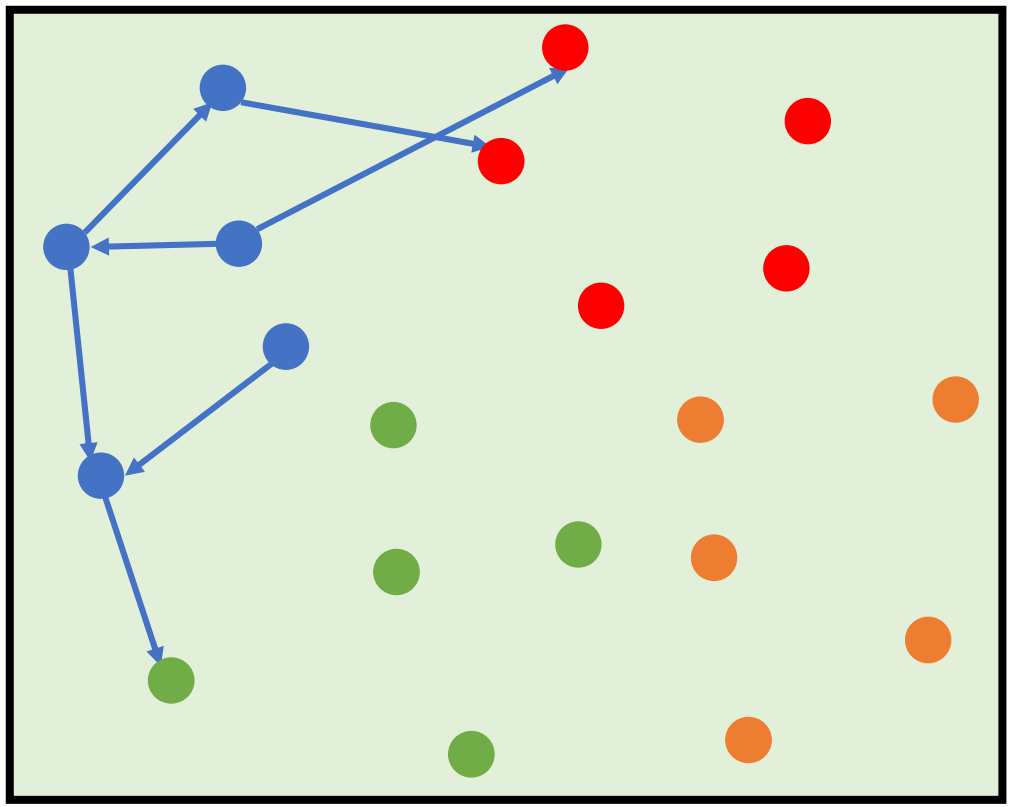
Partition-based Graph Processing

Main Memory



Transferring

GPU Memory



Computation

A Key Observation

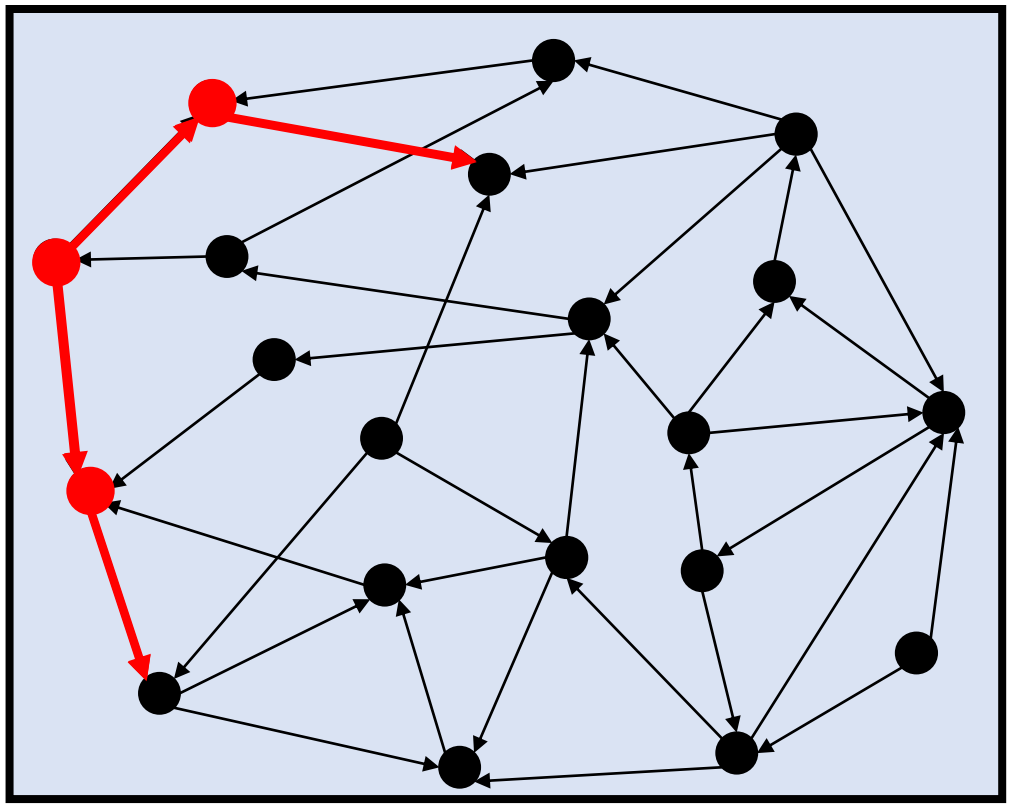
Ratio of active vertices (edges) is often low in most iterations

Average Ratio of Active Edges across Iterations

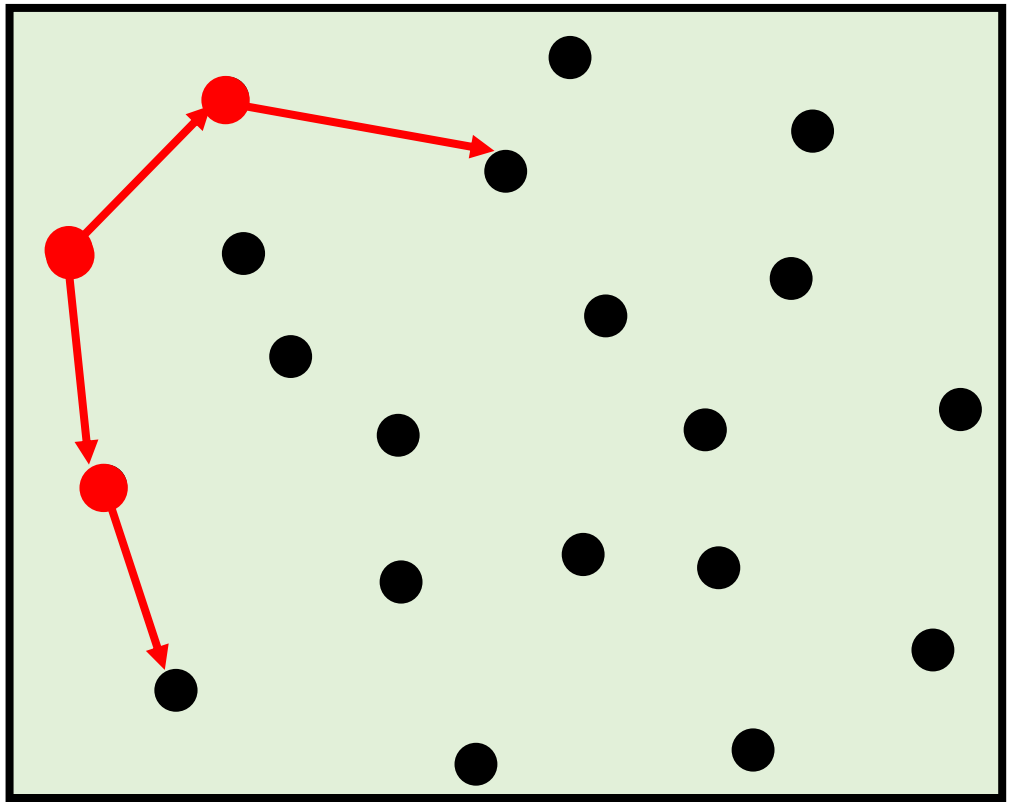
Algo.	friendster	Uk-2007
SSSP	9.1%	5.1%
BFS	4.1%	0.6%
CC	9.8%	3.2%

Only Load Active Edges to GPU?

Main Memory



GPU Memory



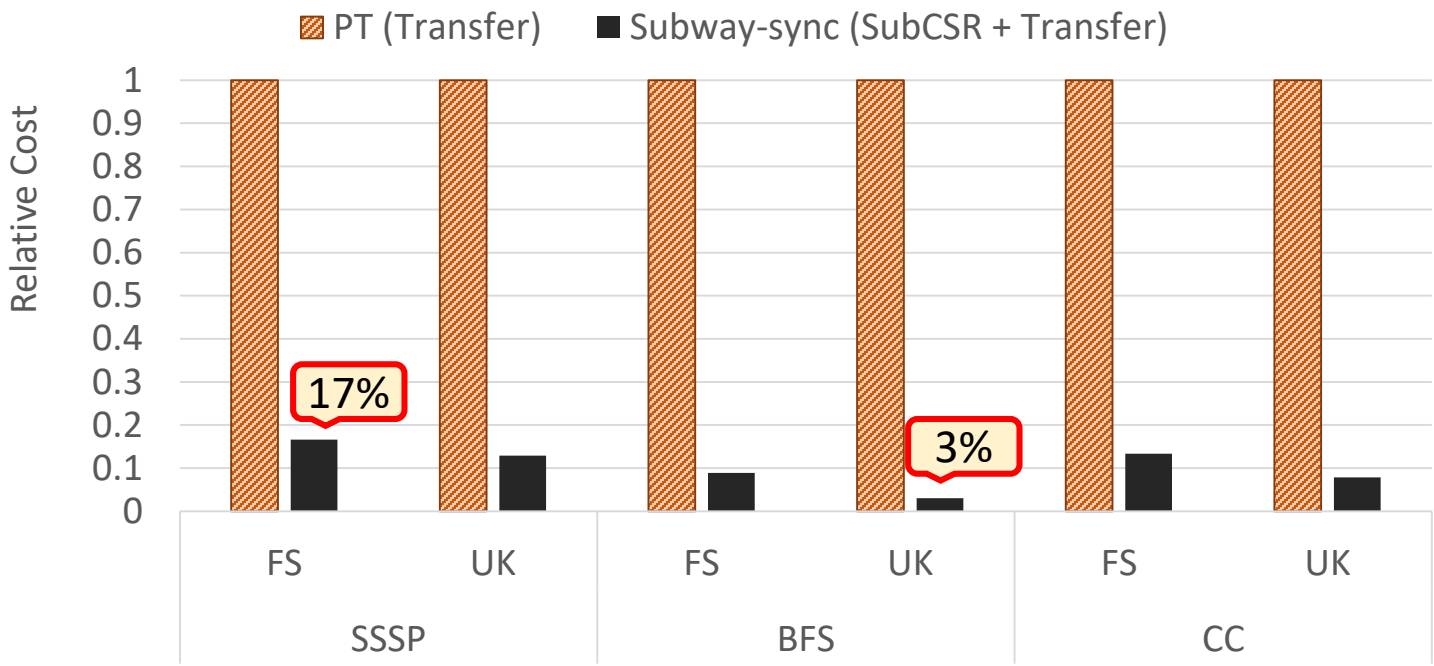
Too expensive to generate ?!

Efficient Subgraph Generation

Subway:

- a concise **subgraph representation**, called SubCSR
- a highly **parallel algorithm** for subgraph generation
- an efficient **GPU-accelerated** implementation

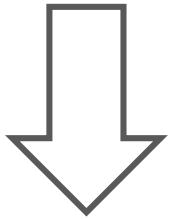
SubCSR Generation Cost



Costs: Partitioning-based vs. Subway (subgraph generation)

Takeaway

~~Too expensive to dynamically generate subgraphs!~~



Subway

Improve performance up to 28X !

Thank you

Amir Nodehi: anode001@ucr.edu or on **Slack**

The source code (to be posted soon): https://github.com/AutomataLab/Subway