VENUS: Vertex-Centric Streamlined Graph Computation on a Single PC

Jiefeng Cheng¹, Qin Liu², Zhenguo Li³, Wei Fan¹, John C.S. Lui², Cheng He¹

¹Huawei Noah's Ark Lab, ²The Chinese University of Hong Kong

Why Single Machine System?

Motivations
We have large graphs:
• Web graph, Social graph, User-login ratings graph, ...

We need to do intensive computation on graphs:
• PageRank, community detection, alternating least squares for collaborative filtering, shortest path, ...

Two Solutions
Systems for graph computation: distributed or single-machine?

Distributed systems: Pregel, GraphLab, Giraph, ...
• Expensive clusters
• Complex setup & configuration
• Writing buggy distributed programs

Single-machine systems: GraphChi[1], X-Stream[2], TurboGraph, FlashGraph, ...
• Store graphs on disk or SSDs
• Graph computation on a commodity PC (cheap, easy to program)

We can achieve competitive results over distributed systems:
PageRank on a Twitter graph (41M nodes, 1.4B edges)
• Spark: 8.1min with 50 machines (each with 2 CPUs, 7.5G RAM)[3]
• VENUS: 8 min on a single machine with quad-core CPU, 16G RAM

Exiting Systems

Vertex-centric programming model: used by Pregel, GraphLab, GraphChi, ...
• Each vertex updates itself based on its neighborhood
A seminar work, GraphChi[1]

for each iteration
  for each vertex v
    update(v)

void update(v)
  fetch data from each in-edge
  update data on v
  spread data to each out-edge

• Updated data on each vertex must be propagated to its neighbors through disk
• Extensive disk I/O

Our new system, VENUS:
• Only store mutable values on vertices

void update(v)
  fetch data from each in-neighbor
  update data on v
  spread data to each out-edge

• Much less data access
• Enable streamlined processing
• Sacrifice little expressiveness

VENUS: Vertex-Centric Streamlined Graph Computation

Main Ideas
Disk storage (offline)
• Split partition onto shards and execute each in memory
• Separate edge data and vertex data
Computing model (online)
• Cache vertex data
• Load edge data sequentially
• Execute update functions in parallel

Architecture

Offline Storage
Each shard corresponds to an interval of vertices:
G-shard: in-edges of nodes in the interval (immutable)
V-shard: vertex values of all vertices in the shard (mutable)

Online Computing Model
Vertex-centric streamlined processing
• V-shards are cached
• G-shards stream in
• Execute $update(v)$ in parallel

Load and Update v-shards
Two I/O efficient algorithms:
1. An extension of PSW[1];
2. A merge-join between value table and v-shard
Detailed in our paper[4].

Experimental Results

Clueweb12: web scale graph
• 978 million nodes, 42.5 billion edges
• 402 GB on disk
• 2 iterations of PageRank

References
[1] Kyrola, A., Bliachko, G., & Guettstrin, C. (2012). GraphChi: Large-Scale Graph Computation on Just a PC. In OSDI.
Contact: Qin Liu (qliu@cse.cuhk.edu.hk)