PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs

Paper review by Sean Parker
Structure

- Background/motivation
- Key contributions
- Results/analysis
- Review
- Developments
Background

- Graph-parallel computation
- Pregel (Google)
  - Bulk synchronous parallel (BSP) model
- GraphLab
  - Asynchronous distributed shared-memory abstraction
- Power-law graphs
Motivation

• Natural graphs have a structure limiting performance
• Communication asymmetry in some graphs
• Difficulty partitioning natural graphs
• High storage costs
Natural graphs

Graphs derived from natural phenomena

- Pregel & GraphLab not suited for natural graphs
- Challenges of high degree vertices
- Low quality partitioning
Partitioning Natural Graphs

- Minimise communication
- Balance communication and storage

- Pregel & GraphLab partition on edges

\[
\mathbb{E}
\left[
\frac{|Edges\ Cut|}{|E|}
\right] = 1 - \frac{1}{p}.
\]
Key contributions

• GAS program abstraction
• Delta caching
• Efficient vertex cuts
• PowerGraph implementation
• Performance evaluation
GAS Abstraction

• Three stages:
  • Gather
  • Apply
  • Scatter

```java
interface GASVertexProgram(u) {
  // Run on gather_nbrs(u)
  gather(D_u, D_{(u,v)}, D_v) → Accum
  sum(Accum left, Accum right) → Accum
  apply(D_u, Accum) → D^{new}_u
  // Run on scatter_nbrs(u)
  scatter(D^{new}_u, D_{(u,v)}, D_v) → (D^{new}_{(u,v)}, Accum)
}
```
Efficient vertex cuts

- PowerGraph investigated three methods for vertex cuts:
  - Random
  - Greedy (Oblivious)
  - Greedy (Coordinated)
More features!

• Delta caching

• Fault tolerance (checkpointing)

• Synchronous, asynchronous, async+serialisable execution
Review & discussion

• Natural graphs are difficult to process efficiently
• GAS abstraction makes it simple to write graph-parallel algorithms
• Vertex-cut algorithm major contribution
• Is PowerGraph only useful for natural graphs?
  • Been used for MLDM applications (Collaborative Filtering, Computer Vision etc)
Since publication

- PowerGraph project became part of GraphLab v2.1
- GraphLab was renamed to Turi (Turi Create)
- Turi purchased by Apple in 2016
  - Became a Python framework for ML applications (classifiers, detection, clustering etc)

https://github.com/apple/turicreate