GraphBuilder: Collaborating to Construct Large-Scale Graphs

Ted Willke
Systems Architecture Lab

November 30th, 2012

http://www.istc-cc.cmu.edu/

Intel Science & Technology Center for Cloud Computing
Petascale graphs: The end-to-end challenge

- GraphLab is indeed promising
- But we struggled with feeding it and other practicalities
- Set out to study potential approaches...

Full Internet Map [Lumeta]

Social Graph [Facebook]
But first... a bit on how we got here.
Hadoop Research

• Evaluating new cluster technologies requires solid baselines
• But an exhaustive search for the best Hadoop configuration would take 7,257,600,000 lengthy trials!
• Solve the challenge and accelerate our other work at the same time?

Issues with manual tuning:
• Domain knowledge
• Parameter interaction
• Time consuming

We spend weeks tuning clusters for a few days of experiments.
Our Approach

1. Focus on the most important parameters for each circumstance*
2. Apply generalized search algorithms to efficiently explore the parameter space
3. Model the system to reason about unexplored space

* Future work
Gunther: The Elephant Trainer
An Auto-tuner for Hadoop MapReduce

Key benefits:
• Little domain knowledge required
• Easily adapts to new datasets, workloads, frameworks, & clusters
• More effective and faster than manual approaches
Search + Model

Genetic search algorithm is ~95% effective in <30 trials

SVR model is very accurate but requires hundreds of trials (320 in this case)

Apply model to predict perf and inform future searches.

<table>
<thead>
<tr>
<th>Modeling Approach</th>
<th>SNE Cluster</th>
<th>ZT Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>Q1</td>
</tr>
<tr>
<td>MLR</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>MLR-I</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>MLR-Q</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>MLR-IQ</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>ANN</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>M5Tree</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>SVR</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>
Dimensionality Reduction

Rule out parameters *up front* that primarily affect resources that aren’t likely to bottleneck

**Direction:**

1. Incorporate node- & cluster-level utilization observations ($m$) into model
2. Apply EV-based MV analysis offline to determine *what* params matter when
3. Use 1st run to collect $m$ and apply to search
But tuned MapReduce is still MapReduce.
“Chance favors the connected mind.”
--Steven Johnson

Garth made the connection at the December 2011 ISTC retreat!

Cluster Computing Architecture
Spark*
Fast, Interactive, Language-Integrated Cluster Computing

- Replaces storage operations with in-memory caching (will fall back if necessary)
- Provides in-memory distributed fault-tolerance and can reconstruct partitions

PageRank on 64 DP Xeons

40M Webpages, 1.4 Billion Links

Hadoop

Twister

GraphLab

GraphLab is 41X faster. What’s going on here?

Hadoop results from [Kang et al. '11]
Twister (in-memory MapReduce) [Ekanayake et al. ‘10]
Big graphs are a big deal!

Many problems involve irregular data structures most naturally expressed as graphs, trees, and arbitrary sets (paraphrasing Keshav Pingali)

- **Human Brain**
  - 100B Neuron
  - 100T Relationships

- **Social Network**
  - 1B Users
  - 140B Friendships

- **Internet**
  - 1Trillion Pages
  - 100s T Links

- **e-commerce**
  - Millions of Products & Users

- **Online Services**
  - 27M Users
  - 70K Movies

- **Science**
  - Large Biological Cell Networks

Image source: [Wikipedia](alz.org) [Facebook]
Collaborative Filtering: Mining Relationships

Customers Who Bought This Item Also Bought What?

- Dog Food
- Milo’s Meatball Treats
- Greenies for Teeth

Time to find product similarities is:

- worst case: \(O(P^2)\) days, minutes, seconds
- if algorithm exploits data dependency structure:
- ideally partitioned: \(O(PC/M)\) days, minutes, seconds

\(|IP|\) and \(|IC|\)
Graph processing:
An extremely short history

Data-parallel
  ▼
  Data-parallel + Iterative
  ▼
  Graph-parallel + Iterative
  ▼
  Asynchronous graph-parallel

Ship the **entire** graph structure...
... over and over ...
or, better yet, pass the results ...
... whenever you want.
So we’re done, right?
“I spend more than half of my time integrating, cleansing and transforming data without doing any actual analysis. Most of the time I’m lucky if I get to do any analysis at all.”

Anonymous Data Scientist
from Jeff Heer’s (Stanford) interview study, 2012
Taking a Broader Perspective

I think I know how write a custom script to construct a graph on 1 machine… to run on my *totally awesome* graph processing cluster!!???

Semi-efficient parallel predictions 😞

* Adaptation of GraphLab team material
Many of these challenges are solved for small problems... but what about Internet scale?
Challenges for Emerging Area

1. Few people skilled in the apps and algos
2. App frameworks emerging and evolving rapidly
3. Lack of tools to deploy systems and analyze system behavior

Lightly charted territory offers big opportunities for Intel and other companies.
Parallel Machine Learning (ML):
Joint work with the ISTC for CC (UW/CMU)
Machine Learning Pipeline

Data

Extract Features

Graph Formation

Structured Machine Learning Algorithm

Graph Ingress mostly data-parallel

Graph-Structured Computation graph-parallel

Cluster Computing Architecture
Hadoop for Graph Construction

- Intuitive Map and Reduce programming model (in Java)
- Framework takes care of resource provisioning
- Provides redundant storage and fault recovery
Cluster Computing Architecture

People

- Kushal
- Diana
- Nilesh
- Danny
- Ted
- Frank
- Ivy
- Jay

Interests

- [Image of interests]

Intel
# Building Graphs for Practical Apps

<table>
<thead>
<tr>
<th>Raw Data</th>
<th>Pre-processing</th>
<th>Graph Formation</th>
<th>Add Network Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>What <strong>words</strong> are most associated with what <strong>(hidden) topics</strong>?</td>
<td>XML Docs</td>
<td>Extract Doc Names and Words</td>
<td>Bipartite (Docs, Words)</td>
</tr>
<tr>
<td>What does context tell me about the <strong>type</strong> (person, place, thing) of this noun?</td>
<td>News Feeds</td>
<td>Extract Noun Phrases and Contexts</td>
<td>Bipartite (NP, Context)</td>
</tr>
<tr>
<td>What are the highest <strong>ranked pages</strong>?</td>
<td>Web Pages</td>
<td>Extract Page URLs and Links</td>
<td>Directed Graph</td>
</tr>
</tbody>
</table>

*Cluster Computing Architecture*
And, in practice and at scale we must:

- Minimize the use of system resources, like memory, storage, etc.
- Ensure GL’s computational effort is load balanced for *power-law graphs*
- Do our best to ensure the graph we generated is the one we intended to

... but the application programmer shouldn’t be responsible for this domain expertise!
GraphBuilder makes it easy.
- Fills a hole in the ecosystem
- Written in Java for easy use in Hadoop MR and apps
- Offloads domain expertise
GraphBuilder Data flow

**Extract**  
Graph formation from data source(s)

HDFS  
DB  
XML Docs

Feature Extraction and Tabulation

**Transform**  
Apply cleaning and transformation

Graph Checks and Transformation

**Load**  
Prepare for graph analytics

Graph Compression, Partitioning, and Serialization

App-Specific Code  
Cluster Computing Architecture

GraphBuilder Library
**Extract - Graph Formation**

Extract features from data to construct relationships

---

**Read** → **Tokenize** → **Optional Reduce**

- **Read Records**
  - Write simple data-specific functions.
  - Program sequential, not parallel!

- **Extract Features**
  - `conf.set(XMLInputFormat.START_TAG_KEY, START_TAG);`
  - `conf.set(XMLInputFormat.END_TAG_KEY, END_TAG);`
  - `new XMLRecordReader((FileSplit) split, conf);`
  - `Document doc = builder.parse(new InputSource(new StringReader(s)));`
  - `title = xpath.evaluate("//page/title/text()", doc);`
  - `title = title.replaceAll("\s", "_");`
  - `id = xpath.evaluate("//page/id/text()", doc);`
  - `String text = xpath.evaluate("//page/revision/text/text()", doc);`
  - `parseLinks(text);`
Extract - Tabulation

Built-in tabulation functions for TF, TFIDF, WC, ADD, MUL, DIV. Interface for custom tabulation on source and/or target vertex.

Example: Term Frequency

User Defined:
- Reduce \( (f(x)) \rightarrow f(x) \)
- Apply \( f(x) \rightarrow f(x) \)

\[
_tf (t, d) = \frac{f(t,d)}{\max\{f(w,d) : w \in d\}}
\]
Transform – Graph Transforms & Checks

- Would like the ability to:
  - Optionally filter duplicate, dangling and/or self edges
  - Transform a directed graph into an undirected graph
  - Calculate graph statistics, compute sub-graphs, etc.

- The library provides:
  - Functions to perform self-, dangling- and duplicate-edge removal
  - Directionality transformation

- Solutions are based on a distributed hashing algorithm

Cluster Computing Architecture
We can save memory if we compress/normalize graph

But, seems to call for global lookups in a framework that prefers independent subproblems

A simple, scalable solution is to “shard” ordered lists:
Load - Graph Partitioning
“Cut quality varies inversely with cut balance.” [Kevin Lang, ’04]

- Minimize communications by minimizing the number of machines v spans
- Place about the same number of edges on each machine
Load - Graph Partitioning

“Cut quality varies inversely with cut balance.” [Kevin Lang, ’04]

- Minimize communications by minimizing the number of machines v spans
- Place about the same number of edges on each machine

"Cut quality varies inversely with cut balance." [Kevin Lang, ’04]
Heuristics-Based Partitioning Strategies

• Random edge placement: Edges are placed randomly by each system

• Greedy edge placement: Global coordination for edge placement to minimize the vertex spanned

• Oblivious greedy placement: Implements a local version of the Greedy without global coordination
Oblivious Algorithm

Machine 1

Machine 2

Cluster Computing Architecture
CASE 1: Both end points have never been seen before

→ Randomly assign
**CASE 2:**
Both end points have been seen before on the same partition

→ Assign to a partition which contains both endpoints
CASE 3: Both end points have been seen before but on different partitions

→ Assign to any partition that contains an endpoint
Machine 1’s Shard

CASE 3:
Both end points have been seen before but on different partitions

→ Assign to any partition that contains an endpoint

Cluster Computing Architecture
Machine 1’s Shard

Partition 1

CASE 4:
Only one end point has been seen before

→ Assign to a partition that contains the endpoint

Partition 2
Partitioning Quality

Twitter Graph: 41M vertices, 1.4B edges

Greedy yields a quality cut and the best performance....

*Gonzalez et al., “PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs,” [OSDI’12]
Performance is inversely proportional to replication.

*Gonzalez et al., “PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs,” [OSDI'12]*
Load - Graph Serialization

- Self-describing data format
  - JSON +/- compression
- Extensible
  - Easy to connect with Graph Databases
  - Plug-in Graph Visualizers

```
{  
  "src_id": 34,
  "dest_id": 45
  "e-data": 30
}

{  
  "ver_id": 34,
  "v-data": 56,
  "mirror": [1,2,3],
  "owner": 1
}
```
GraphBuilder Stack

Cluster Computing Architecture
GraphBuilder Demo

WIKIPEDIA → GraphBuilder → Partitioned Bipartite graph → Latent Dirichlet Allocation (LDA) Algorithm → GraphLab

WordCloud Visualizer

Topic Modeling

Knowledge Extraction

Cluster Computing Architecture
Our Wikipedia Graphs

| Graph     | $|V|$   | $|E|$   | $\alpha$ |
|-----------|-------|--------|----------|
| LDA       | 4.9M  | 478M   | 2.23     |
| PageRank  | 9.7M  | 107M   | 2.41     |
Prototype Overview

- **Hardware:** 8 node cluster
  - 1U Dual CPU (Intel SNB) Amazon build ZT systems
  - 64 GB Memory, Four SATA Hard Drives
  - Intel 10G Adapter and Switch

- **Software:**
  - Apache Hadoop 1.0.1
  - GraphLab v2.1
  - GraphBuilder beta
Preliminary Results

<table>
<thead>
<tr>
<th>Graph</th>
<th>Custom plug-in code</th>
<th>Graph Compression</th>
<th>Partitioning Improvement (vs. Random)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PageRank</td>
<td>100 lines</td>
<td>60%</td>
<td>17%</td>
</tr>
<tr>
<td>LDA</td>
<td>130 lines</td>
<td>5%</td>
<td>32%</td>
</tr>
</tbody>
</table>

Hardware: 8 node cluster
1U Dual CPU (Intel SNB) Amazon build ZT systems
64 GB Memory, Four SATA Hard Drives
Intel 10G Adapter and Switch

Software:
Apache Hadoop 1.0.1
GraphLab v2.1
GraphBuilder beta

Cluster Computing Architecture
Scaling Experiment

Cluster Computing Architecture
Collaboration ahead!

Sam Madden  Carlos Guestrin  Ted Willke
All Together Now

Parallel Machine Learning Pipeline

Future areas for ISTC collaboration:
1. Improve usability and data wrangling
2. Research GL fault tolerance and local storage support
3. Advance GB + GL for streaming and time-evolving apps
Launches today!

Intel open source portal at http://www.01.org

GraphLab2 at http://graphlab.org

Both under Apache 2.0 licensing.

Looking for research partners and committers. Contacts:

nilesh.jain@intel.com
theodore.l.willke@intel.com
How many people are pointing to you and what’s their relative importance?

What’s the rank of this user?

Depends on rank of who follows her

Depends on rank of who follows them...

Loops in graph - Must iterate!

Graphics source: [Joseph Gonzalez (CMU)]
Properties of Graph-Structured Computation

Dependency Graph

Local Update

Iterative Computations

Similar properties for many other problems!
Data Dependencies

- MapReduce does not efficiently express data dependencies
  - User must code substantial data transformations
  - Costly data replication

- MapReduce does not efficiently express iterative algorithms
Approaches to Graph-Structured Computation

• **Bulk Synchronous Processing (BSP)**
  – Giraph on Hadoop (Inspired by Google Pregel)
  – Dryad (Microsoft Research)
  – Apache Hama on Hadoop (Twitter)

• **Asynchronous Graph-Parallel**
  – Galois (UT Austin) → Edge partitioning
  – GraphLab (CMU) → Vertex partitioning

GraphLab has an edge.
GraphLab Goals

• Designed specifically for ML
  – Graph dependencies
  – Iterative
  – Asynchronous
  – Dynamic

• Simplifies design of parallel programs
  – Abstracts away hardware issues
  – Automatic data synchronization
  – Addresses multiple hardware architectures
The GraphLab Framework

- **Graph Based Data Representation**
- **Scheduler**
- **Update Functions User Computation**
- **Consistency Model**

*Cluster Computing Architecture*
GraphBuilder makes it easy.

- Fills a hole in the ecosystem
- Written in Java for easy use in Hadoop MapReduce and applications
- Offloads domain expertise
Prototype Overview

• Hardware: 8 node cluster
  – 1U Dual CPU (Intel SNB)
  – Amazon build ZT systems
  – 64 GB Memory, Four SATA Hard Drives
  – Intel 10G Adapter and Switch

• Software:
  – Apache Hadoop 1.0.1
  – GraphLab v2.1
  – GraphBuilder beta