Scalable Graph Construction using Apache™ Hadoop™

GraphBuilder

Acknowledgements:
Carlos Guestrin et al. (CMU/UW)
(Collaboration through Intel Science and Technology Center)

Nilesh Jain
Intel Corporation

October 24, 2012
Data Analytics and Machine Learning Framework
“Perhaps even more than to the link between mankind and nature, graph theory owes to the link of human beings between each other.”

English translation of “Vielleicht noch mehr als der Berührung der Menschheit mit der Natur verdankt die Graphentheorie der Berührung der Menschen untereinander.”

From 1st Graph Theory Book (1936)
Bridges of Königsberg

Frank Harary's bibliography Wordle

Social links

1736  1850  1936  1969  1985+  21st Century

Leonhard Euler  Francis Guthrie  William Rowan Hamilton  Dénes König  Frank Harary  Computing Revolution  Internet
Natural graphs

Graphs derived from natural phenomena.
Don’t have idealized structure
Don’t have regularized structure
Overlapping communities
Natural graphs follow preferential attachments
They grow with time
“Rich get Richer”
Power-Law Degree Distribution

Twitter Follow Graph

More than $10^6$ vertices have one neighbor.

Image source: [Wikipedia] [cmu.edu/~pegasus]
Graphs are omnipresent!

- **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

Big in size and rich in metadata
Graphs are Essential to Data Mining and Machine Learning

- Identify influential people and information
- Find communities
- Understand people’s shared interests
- Model complex data dependencies
Identifying influential people

Construct a graph

Feature Extraction

Graph Construction

Graph

Social Networking Data

Graph Data-Parallel

Image source: [Wikipedia]
Need a Model (Algorithm)

PageRank:

Definition 1 Let $E(u)$ be some vector over the Web pages that corresponds to a source of rank. Then, the PageRank of a set of Web pages is an assignment, $R'$, to the Web pages which satisfies

$$R'(u) = c \sum_{v \in B_u} \frac{R'(v)}{N_v} + cE(u)$$

such that $c$ is maximized and $\|R'\|_1 = 1$ ($\|R'\|_1$ denotes the $L_1$ norm of $R'$).

[Page et al. 1998]
How many people are pointing to you and what’s their relative importance?

Loops in graph - Must iterate!

What’s the rank of this user?

Depends on rank of who follows her

Depends on rank of who follows them...

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

Dependency Graph

Local Update

Iterative Computations

Similar properties for many other problems!

Graphics source: [Joseph Gonzalez (CMU)]
How do we program graph computation?

“Think like a Vertex”

Malewicz et al. [SIGMOD’10]
The Graph-Parallel Abstraction

A user-defined Vertex-Program runs on each vertex

Graph constrains interaction along edges
  Using messages
  Through shared state

Parallelism: Run multiple vertex programs simultaneously
Distributed Graph Analytics System

Data

Graph Construction
(Feature Extraction, Graph formation)

Graph Ingress
mostly data-parallel

Structured Machine Learning or Data Mining
(Identify influential person)

Graph-Structured Computation
graph-parallel

Efficient design requires balanced utilization

Value
System-Level Challenges

- **Too Big to fit**
  (in system memory)

- **Balanced cut**
  (Power-law graphs are difficult to cut)

- **Work imbalance**
  (Execution on vertex is proportional to degree of vertex)
Difficult to Partition

Power-Law graphs do not have low-cost balanced cuts [Leskovec et al. 08, Lang 04]
Partitioning Approaches
Impact on system performance

Edge cut:

Any edge cut can directly construct a vertex cut which requires strictly less communications and storage. [Gonzalez et al. 2012]

Percolation theory suggests that power law graphs have good vertex cuts. [Albert et al. 2000]
Distributed Graph Analytics Environment

Program For This

Split High-Degree vertices

Run on This

Machine 1
- Master

Machine 2
- Slave

Graphics source: [Joseph Gonzalez (CMU)]
Graph-Structured Computational Frameworks

Pregel - Malewicz et al.  
[PODC’09, SIGMOD’10]

Apache Giraph 2011

Carlos Guestrin et al.  
[UAI’10, OSDI’12]

Others: Kineograph, Stanford GPS, Dryad, BoostPGL, Pegasus, Microsoft Trinity, and Signal-Collect
PageRank Performance

Twitter Graph $|V| 41M$ $|E| 1.4$ Billion

Hadoop

13.3 hrs

GraphLab

14 min

57x

Not a natural fit for Graph-Parallel Abstraction
Must store graph’s state after every iteration

Twitter Graph $|V| 41M$ $|E| 1.4$ Billion

8-node Intel Sandy Bridge E3-1280 Cluster, 16GB/node, 10GbE, 2x SSDs (550 MB/s each)
Graph Construction
“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
Building Graphs for Practical Apps

- Raw Data
  - Influential Person
  - Social Networking
- Pre-processing
  - Extract User and Relationship
  - Extract Doc & Words
- Graph Formation
  - Directed Graph
  - Bipartite (Doc, Words)
  - Bipartite (User, item)
- Add Network Information
  - N/A
  - Word Frequency or TFIDF
  - Rating

- Activity Logs
- XML Docs
- Hidden Topic analysis
- Recommendation System
And, in practice and at scale we must:

- Minimize the use of system resources, like memory, storage, etc.
- Natural Graph partitioning to ensure computational effort is load balanced
- Do our best to ensure the graph we generated is the one we intended to

but the Data Scientist shouldn’t be responsible for this domain expertise!
Graph Abstraction Library: **GraphBuilder**

- Offloads domain expertise
- Written in Java for easy use in Hadoop MapReduce and applications
- Completes Graph Analytics pipeline
GraphBuilder Data flow

**Extract**
Graph formation from data source(s)
- HDFS
- DB
- XML Docs
- Feature Extraction
- Tabulation

**Transform**
Apply cleaning and transformation
- Graph Checks
- and Transformation

**Load**
Prepare for graph analytics
- Graph Compression
- Partitioning
- Serialization

App-Specific Code

GraphBuilder Library

Intel
Extract - Graph Formation

Extract features from data to construct relationship

Optional: Reduce (● → f(x))

Read vertex object

Feature Extraction

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

\[ tf(t, d) = \frac{f(t,d)}{\max\{f(w,d): w \in d\}} \]
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

Steering function
We can save memory if we normalize it (e.g., compress Link graph by 60%).

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

A simple, scalable solution is to “shard” ordered lists:

Dictionary
- (Aaron,0)
- (AMD,4)
- (Brad,1)
- (CMU,2)
- (Dan,5)
- (Dave,3)
- (IBM,6)
- (Intel,7)

Dictionary Shard 1
- (Aaron,0)
- (AMD,4)
- (Brad,1)
- (CMU,2)

Unconverted Edge List
- (Aaron,IBM)
- (Brad,Intel)

M1
- (AMD,5)
- (CMU,3)

Converted Edge List
- (5,4)
- (3,2)
- (0,6)
- (1,7)

Dictionary Shard 2
- (Dan,5)
- (Dave,3)
- (IBM,6)
- (Intel,7)

Source Sorted
- (Dan,AMD)
- (Dave,CMU)

M2
- (IBM,0)
- (Intel,1)

Dest Sorted
Load - Graph Partitioning

Minimize communications by minimizing the number of machines vertex spans

Place about the same number of edges on each machine

*Traditional graph-partitioning algorithms perform poorly on Power-Law Graphs. [Abou-Rjeili et al. 06]*
Heuristic-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
Greedy Algorithm

Place edges on machines which already have the vertices on that edge while ensuring balance loading.
Partitioning Quality

Twitter Graph: 41M vertices, 1.4B edges

Greedy yields a quality cut, but what is the effect on performance?

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

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

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

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

- Built-in Parser/Tabulator
- Custom Parser/Tabulator
- Extract
- Transform
- Load
- Hadoop/Map-Reduce
- Distributed Graph
- Hadoop/HDFS
- Linux Cluster Services (Amazon AWS)
- Private Linux Cluster
Speed of Graph Construction

Wikipedia Graphs

<table>
<thead>
<tr>
<th>Graph</th>
<th>Compression</th>
<th>Custom plug-in code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link</td>
<td>60%</td>
<td>100 lines</td>
</tr>
<tr>
<td>Word-Doc</td>
<td>5%</td>
<td>130 lines</td>
</tr>
</tbody>
</table>

Execution time $\alpha O(|V|)$

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
Summary

- Graphs are essential for structured ML and DM
- High-performing Graph-Analytics pipelines requires careful system design
- *GraphBuilder* solves the Graph Analytics ingress challenge
Going forward…

Available soon (mid-Nov)!

Intel Open-Source Portal http://www.01.org
GraphLab is available at http://graphlab.org
Apache 2 license

Interested in collaboration – Would like to hear from you!
(Office hours: Thursday 10:50am)

Intel Booth (#27) – Real time Analytics, Hadoop Benchmarking
www.intel.com/bigdata
Legal Notices

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.

- Intel may make changes to specifications and product descriptions at any time, without notice.
- All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.
- Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- Code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel’s internal code names is at the sole risk of the user.
- Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.
- Intel, Intel Inside, and the Intel logo are trademarks of Intel Corporation in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
- Copyright © 2012 Intel Corporation.