GraphOps: A Dataflow Library for Graph Analytics Acceleration

Tayo Oguntebi*
Google, Inc.

Kunle Olukotun
Pervasive Parallelism Laboratory
Stanford University

* Work done while the author was at Stanford University

22 February 2016
FPGA 2016
Outline

- The GraphOps Library
- Locality-Optimized Graph Representation
- Results and Conclusions
The GraphOps Library

```c
// BFS order iteration from s
InBFS(y; G, Nodes, From, s) {
    Do {
        if (y == s) {
            Din = Sum(u:G.Nodes) (u.member == num) {
                u.Degree()
            },
            Dout = Sum(u:G.Nodes) (u.member != num) {
                u.Degree()
            },
            Cross = Sum(u:G.Nodes) (u.member == num) {
                Count(j:u.Nbrs) (j.member != num)
            },
        } else {
            // More code here
        }
    } While (/* condition */) 
}
```
Running Example: PageRank

Procedure pagerank()
{
    Double diff;
    Int cnt = 0;
    Double N = G.NumNodes();
    G.pg_rank = 1 / N;
    Do {
        diff = 0.0;
        Foreach (t: G.Nodes) {
            Double val = (1-d) / N + d* Sum(w: t.InNbrs) {
                w.pg_rank / w.OutDegree();
            };
            diff += | val - t.pg_rank |;
            t.pg_rank <= val @ t;
        }
        cnt++;
    } While ((diff > e) && (cnt < max));
}
The GraphOps Library

ForAllPropRdr ➔ NbrPropRed ➔ ElemUpdate

DRAM Interface
The GraphOps Library

DATA

- ForAllPropRdr
- AllNodePropRdr
- SetWriter
- VertexReader

CONTROL

- NbrPropRed
- NbrPropRdr
- NbrPropFilter
- NbrSetReader

UTILITY

- ElemUpdate
- SetReader
- GlobNbrRed
The GraphOps Library

DATA

CONTROL

UTILITY

Data Readers

- ForAllPropRdr
- AllNodePropRdr
- VertexReader
- NbrPropRdr
- NbrSetReader

Reduction

- GlobNbrRed
- NbrPropRed

Set Manipulation

- SetReader
- SetWriter

Property Filtering

- NbrPropFilter

Mutation

- ElemUpdate
The GraphOps Library

- Set of optimized hardware blocks for executing common graph processing functions
  - High-level: Easy to use
  - Composable: Flexible enough to compose different applications
  - Extensible and parameterizable
  - Pre-verified: Low-level implementation details built-in to the design

Problem: Poor Locality $\Rightarrow$ Poor Performance!
Outline

- The GraphOps Library
- Locality-Optimized Graph Representation
- Results and Conclusions
Rethinking the Graph Representation

Conventional Form: Compressed Sparse Row (Adjacency Lists)

No locality!
Rethinking the Graph Representation

Locality-Optimized Form

Node Indices

Node Array

Edge Array

Property Array
(e.g. Pagerank scores)

Locality-Optimized Array

Trades off compactness for locality... Space for time
Pre-Processing the Layout

- We have locality...now need to restore consistency

- ProcessGraphLayout(): *Scatter* operation
  - Performed on the host

“The cheapest decent memory controller that you can buy is still an Intel Xeon CPU...” – Prof. Christos Kozyrakis
Programming Model

Graph* g;
GenerateGraph(g);

PreprocessGraphLayout(); // Prepare locality-optimized form

do {
    WriteToDeviceMem();
    Run();
    ReadFromDeviceMem();
    ProcessGraphLayout(); // i.e. scatter
} while (not converged);
Outline

- The GraphOps Library
- Locality-Optimized Graph Representation
- Results and Conclusions
Energy Efficiency (Throughput / Watt)

Graph Size (N)

Efficiency: SpMV

- GraphOps
- SW 1
- SW 8
- GraphOps+Scatter

Uniform graph. Avg degree 8.
Thank You

• Details and full results in the paper

• Questions: Find me during the break / poster session.

• Complete library open-sourced (MIT License) and available at:
  https://github.com/tayo/GraphOps
Supplementary Material
ForAll Property Reader
**Evaluation Platforms**

- Intel Xeon 5650 @ 2.7GHz
  - 2 sockets, 12 cores, 24 threads
  - Bandwidth: 32 GB/s per socket
  - 3 Memory Channels
- FPGA: Xilinx Virtex-6 (150MHz)
  - Connected to host via PCIe x8 Gen 2
  - Bandwidth: 38.4 GB/s

**Effective performance is about 1/6 of what bandwidth allows**

**Constraining Factors**

**Locality**
- Optimal: Sequential access
- Using: Alternating reads to the different arrays. All units operating simultaneously

**Packet size**
- Optimal: 384 bytes x 4
- Using: 192 bytes x 2

**Burst size**
- Optimal: as large as possible (max 256)
- Using: usually 1-2 (enough for a nbr set)

---

**PageRank System**

- ForAllPropRdr
- NbrPropRed
- ElemUpdate

**Vertices** (Fully-Used)

**PageRank scores** (Not Fully Used)

**Updated Page ranks** (Fully-Used)

**DRAM**

---

**PageRank Bandwidth Usage**

- Prop Array (Write): 2%
- Prop Array (Read): 93%
- L-O Array (Read): 2%
- Node Array (Read): 2%
The GraphOps Library: Utility Blocks

ForAllPropRdr \rightarrow NbrPropRed \rightarrow ElemUpdate

Done \rightarrow Done \rightarrow Done

MemUnit \rightarrow MemUnit

DRAM Interface
Neighbor Property Reducer
Element Update

Diagram showing the process of element update with nodes labeled ArrayCtrl, UpdQueueSM Control, Data Packet Composition Registers, Pipelined Logic, and DataArray. Arrows indicate the flow of data and control signals, including Result to Write, Array Index, Stall, Read Request, Write Request, and Done.
X-Stream: Streaming Graphs on CPUs

- Graph processing system using commodity hardware
- Sequentially streams entire edge lists, generates updates on active edges
- Designed to take advantage of sequential memory – absolutely no memory lookups necessary
# X-Stream Comparison: Datasets

<table>
<thead>
<tr>
<th>Name</th>
<th>Nodes</th>
<th>Edges</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>amazon0601</td>
<td>475K</td>
<td>3.4M</td>
<td>Amazon product co-purchasing network from June 1 2003</td>
</tr>
<tr>
<td>cit-Patents</td>
<td>3.8M</td>
<td>16.5M</td>
<td>Citation network among US Patents</td>
</tr>
<tr>
<td>wiki-Talk</td>
<td>2.4M</td>
<td>5M</td>
<td>Wikipedia talk (communication) network</td>
</tr>
<tr>
<td>web-BerkStan</td>
<td>685K</td>
<td>7.6M</td>
<td>Web graph of Berkeley and Stanford</td>
</tr>
<tr>
<td>soc-Pokec</td>
<td>1.6M</td>
<td>30.6M</td>
<td>Pokec online social network</td>
</tr>
</tbody>
</table>

Datasets are courtesy of the Stanford SNAP project. [snap.stanford.edu](http://snap.stanford.edu)
X-Stream Comparison

**Power Comparison**

- **X-Stream**: 190 W (2 sockets, TDP)
- **GraphOps**: ~25 W
Potential Future Work

• Higher level synthesis tool to target the GraphOps library
• Hide data transfer latency with double buffering and asynchronous execution
• Investigate locality-optimized storage for other sparse domains, e.g. machine learning
• Batch updates for host-side application
• Multi-FPGA
• Dynamic Graphs
Memory Consistency

- Single writer per array
- If a GraphOps block is modifying an array, only that block may be simultaneously reading from the array
- Replicated arrays are read-only. Updates are made to the standard property array.
  - Use a SCATTER operation at the end of the computation
Locality-Optimized Format: on CPUs

- PageRank
  - 2M nodes, 16M edges
  - OMP-C++, 4-thread
    - Current run-time: 3040
    - With replicated arrays: 1610

- Advantage was erased with the scatter
Locality-Optimized Format: on CPUs

- Colleague (Chris) has been working on graph storage formats
- He attempted to implement my idea as part of a CPU run-time
- The scatter nullifies the advantage of the coalesced memory accesses
Bandwidth Study

**Streaming Architecture: Page Rank Accelerator**

- **Vertex Reader**
- **Nbr Reducer**
- **Elem Updater**
- Vertices ➔ Pagerank data (replicated) ➔ Updated Pageranks ➔ DRAM

**Memory Layout**

- **Node Array**
  - 0 1 3 6 10 13 16 18
- **Edge Array** (not used)
  - 2 3 4 0 5 6 1 4 5 6 1 3 5 2 3 4 2 3
- **Property Array** (Pagerank)
  - p0 p1 p2 p3 p4 p5 p6 p7
- **Replicated Array** (Pagerank)
  - p2 p3 p4 p0 p5 p6 p1 p4 p5 p6 p1 p3 p5 p2 p3 p4 p2 p3

**Evaluation Platforms**

- **Intel Xeon 5650 @ 2.7GHz**
  - 2 sockets, 12 cores, 24 threads
  - **Bandwidth**: 32 GB/s per socket
  - **3 Memory Channels**
- **FPGA**: Xilinx Virtex-6 (150MHz)
  - Connected to host via PCIe x8 Gen 2
  - **Bandwidth**: 38.4 GB/s

**Bandwidth Usage**

- **Page Rank**
  - Prop Array (Write) 2%
  - Prop Array (Read) 93%
  - Rep Array (Read) 2%
  - Node Array (Read) 2%

**Methodology**

- Instrumented memory interface units with counters

**Line Bandwidth**: 6.4 GB/s

**Constraining Factors**:

- **Locality**
  - Optimal: Sequential access
  - Using: Alternating reads to the different arrays. All units operating simultaneously

- **Packet size**
  - Optimal: 384 bytes x 4
  - Using: 192 bytes x 2

- **Burst size**
  - Optimal: as large as possible (max 256)
  - Using: usually 1-2 (enough for nbr set)
Usage Calculations

- Replicated array requests (number of bursts):
  - 1, 1, 1, 2, 1, 1, 2, ...
  - Average number of bursts per request: 1.22
    - Divided instrumented value of repl data divided by number of nodes
- Average number of bursts used per request: Assuming uniform with average degree of 8: 8 nbs is 0.25 bursts. So usage rate is: 0.25/1.22 = 0.205
- Expected nbr bandwidth is: 6.4 GB/s * 0.205 = 1.312 GB/s
- Peak performance of PageRank is: 36 MEPS == 216 MB/s
  - About a factor of 1/6 of the expected performance

- Cause of performance being dropped on the floor: Single Memory Channel
  - Queuing/Switching: Nbr Reducer has to wait on the other requests using the memory channel concurrently and pay the cost of switching the active bank/rank/columns etc
- Ideally: multiple memory channels. One of them dedicated to Replicated data for streaming.
Limitations of the GraphOps Library

• Limited expressability
• Limited portability
• Requires coalesced data for efficiency\(^1\)
  – Common graph formats lead to highly inefficient memory behavior

\(^1\) Efficient Parallel Graph Exploration on Multi-core CPU and GPU. Hong, Oguntebi, et al.
Streaming Processors

• Multiple “functional units” execute simultaneously
• Each function performs a different task on the data stream flowing through it
• GraphOps blocks are implemented as coarse-grained functions
• More simple approach for end user: higher level building blocks
Disadvantages to Graph Replication
Additional GraphOps diagrams
Figures from FPGA Paper
GraphOps are Parameterizable
Edge Properties

• A logical way of describing the locality-optimized format
  – Think of the data as being associated with an edge instead of a vertex
Approach

• Initially started with a domain-specific HLS approach
• Was hoping to build full applications on the FPGA
  – Sensitive control was difficult / time-consuming to generate automatically in hardware
  – Especially without an ISA and full architecture
  – Memory behavior was bad anyway
• Converted to an accelerator-based approach
Real-world Dataset properties

• (from snap website)
How Different from GPUs and CPU Vector Machines
Scatter/Gather Options in HW
About Us

We develop brand name with individual creative solutions and help our customers to earn money

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged.
Team Work Sample

We develop brand name with individual creative solutions and help our customers to **earn money**

Marcus Lopez  
*Designer*

María Castro  
*Marketing*

Carlos Perez  
*Animation*

Antonio Ruiz  
*Sales Rep*

Architecto beatae vitae dicta sunt explicabo nemo enim ipsam voluptatem.
Contrary to popular. It has roots in a piece of classical Latin \textit{our process}, from start to finish.

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown.
Services List Sample

We develop brand name with individual creative solutions and help our customers to ear money

INVESTMENT
CONSULTING
Architecto beatae vitae dicta sunt explicabo nemo enim ipsam voluptatem. Architecto beatae vitae dicta.

TAX
STRATEGIES
Architecto beatae vitae dicta sunt explicabo nemo enim ipsam voluptatem. Architecto beatae vitae dicta.

BROKER
COMPARISION
Architecto beatae vitae dicta sunt explicabo nemo enim ipsam voluptatem. Architecto beatae vitae dicta.

BUSINESS
ANALYTICS
Architecto beatae vitae dicta sunt explicabo nemo enim ipsam voluptatem. Architecto beatae vitae dicta.
We develop brand name with individual creative solutions and help our customers to **earn money**.

*Lorem Ipsum* is simply dummy text of the printing and typesetting industry. *Lorem Ipsum* has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged.

*Lorem Ipsum* is simply dummy text of the printing and typesetting industry. *Lorem Ipsum* has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.
2 Columns Sample

We develop brand name with individual creative solutions and help our customers to earn money.

**Lorem Ipsum** is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged.

**Lorem Ipsum** is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged.

**Lorem Ipsum** is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.