Modeling Performance of Graph Programs on GPUs in a Compiler

Sreepathi Pai   Keshav Pingali

August 11, 2016

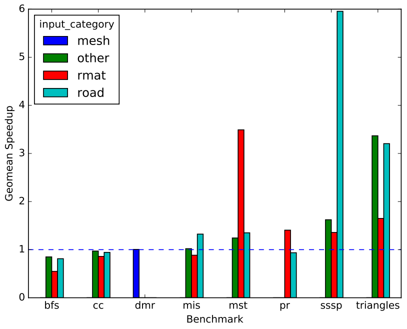
The University of Texas at Austin
ModSim 2016

Outline

Outline

Motivation

Speedup of the Galois GPU compiler on 7 graph algorithms[1]



[1] S. Pai and K. Pingali, ``A compiler for throughput optimization of graph algorithms on GPUs'', in *OOPSLA 2016*.

Performance of Graph Programs

- Algorithm
  - BFS is $O(|V+E|)$, but many implementations are $O(n^2)$
  - δ-stepping SSSP is an order of magnitude faster than naive
- Graph Input
  - Road networks are uniform, high-diameter, and exhibit locality
  - Social network graphs are non-uniform, low-diameter and have little locality
- Software (Runtime)
  - Data structure memory layout
  - Data structure implementation
- Hardware
  - Memory bandwidth (?)
  - Atomic instruction performance

The Problem

- No performance model exists for graph programs on GPUs
    - Must manually tease out performance effects
- No sound methodology exists to guide effort
    - *Ad hoc* techniques lead to incorrect generalizations
- No useful characterization to drive algorithms, runtimes, architecture
    - Can we ever achieve peak performance?

Outline

Motivation

Queuing Models for Graph Programs

Results, Analytical Modelling and Characterization

Conclusions and Future Work

Breadth-First Search

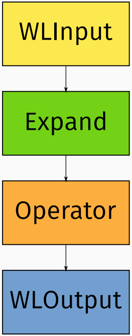
Level-by-Level Breadth-First Search (BFS)

```
Kernel BFS(graph, LEVEL)
   ForAll(N in Worklist)
       ForAll(e in graph.edges(N))
           If(e.dst.level == INF)
               e.dst.level = LEVEL
               Worklist.push(e.dst)
```

- Worklist contains source node initially
- Worklists are bulk-synchronous

BFS as a Queuing Model

```
ForAll(N in Worklist)
  ForAll(e in graph.edges(N))
    If(e.dst.level == INF)
      e.dst.level = LEVEL
      Worklist.push(e.dst)
```



Except for Operator, all other stages are independent of BFS

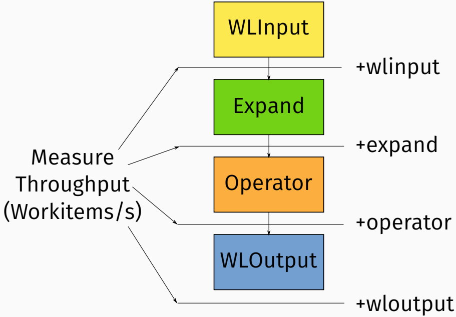
The Operator Machine

The Operator Machine is a multistage queuing network model for graph programs:

- Input
    - WLInput, AllNodesInput, AllEdgesInput
- Expansion (optional)
    - XSerial, XThreadBlock, XWarp, ...
- Operator
    - NodeOp, EdgeOp
- Output (optional)
    - WLOutput

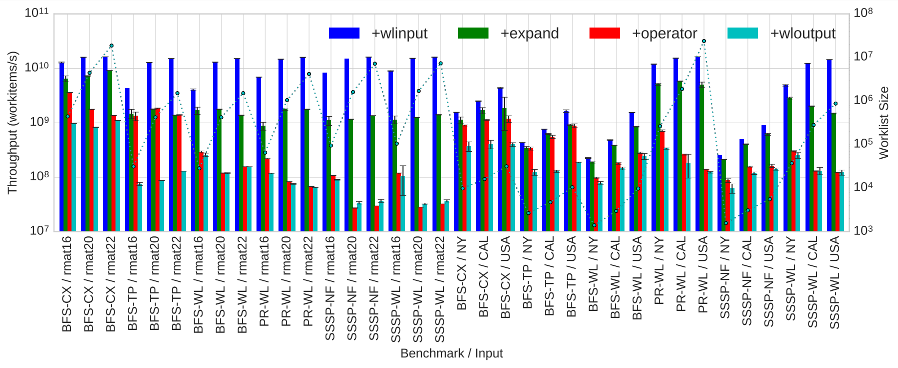
Measuring Peak Throughput of an Operator Machine

- "Cumulative" benchmarks for each stage
- Requires checkpoints from full executions
    - Compiler-assisted
- Yields peak throughputs



Outline

Initial Results: Peak Throughputs



Input Performance

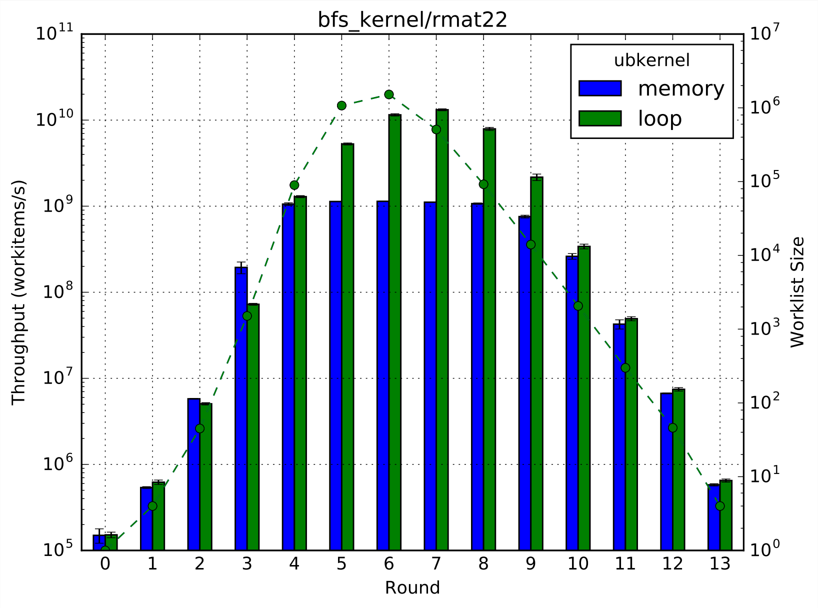
- WLInput - Read a worklist
- Peaks out at 56GByte/s
- Depends on:
    - Size of worklist
    - Number of concurrent reads / thread
- Worklists are large for Social Network Graphs
- Worklists are usually small for Road Networks
    - In BFS and SSSP
    - Not in PageRank, Minimum Spanning Tree, Connected Components

Expansion Performance

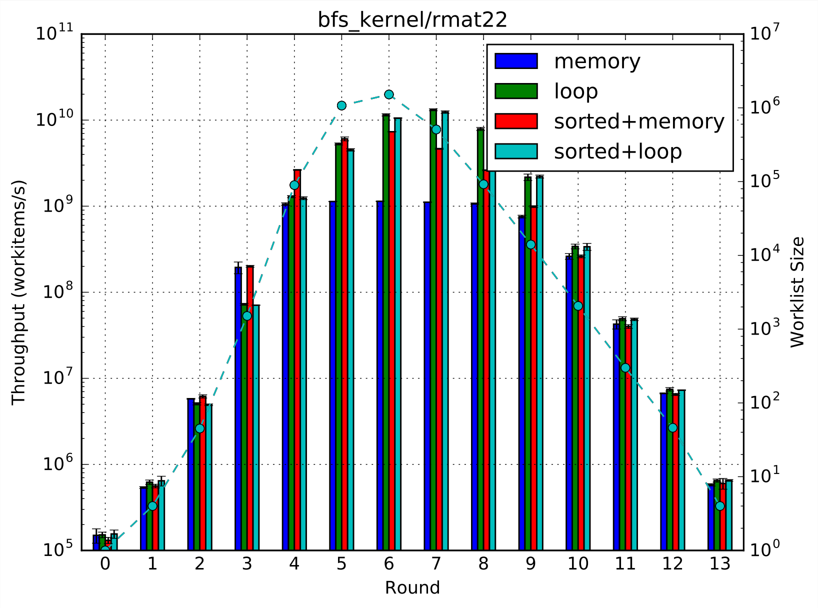
Assuming graph data-structure uses CSR layout

```
N = Worklist[tid]

// indirect memory accesses
start = graph.row_offset[N];
end = graph.row_offset[N+1];

// irregular loop
for(i = start; i < end; i++) {
     // empty
}
```

Microbenchmark Performance for Expansion



After sorting the worklist



Expansion Characterization

- Indirect Memory Access performance is dictated by:
  - Number of 32-byte lines spanned per GPU warp
  - TLB hit rate
- Loop performance is dictated by:
  - Maximum number of edges
  - Branch performance

Outline

Conclusions and Future Work

- Operator Machine is a queuing network model for graph programs on GPUs
  - Allows us to drill down into performance
  - Generalizes well
  - Yields sound conclusions
- TLB Miss Throughput is critical for random graphs
- Compiler integration in progress to guide profile-based optimizations