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

Motivation

Queuing Models for Graph Programs

Results, Analytical Modelling and Characterization

Conclusions and Future Work
Motivation

Queuing Models for Graph Programs

Results, Analytical Modelling and Characterization

Conclusions and Future Work
Motivation

Speedup of the Galois GPU compiler on 7 graph algorithms

\[1\] 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 $O(n^2)$
  - $\delta$-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
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
Besides Operator, all other stages are independent of BFS.
The Operator Machine is a multistage queuing network model for graph programs:

- **Input**
  - `WLINPUT`, `ALLNODESINPUT`, `ALLEGESINPUT`

- **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

Motivation

Queuing Models for Graph Programs

Results, Analytical Modelling and Characterization

Conclusions and Future Work
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
Assuming graph data-structure uses CSR layout

```plaintext
    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

![Graph showing Throughput (workitems/s) for different worklist sizes and kernels. The x-axis represents the round, and the y-axis represents the throughput in workitems/s. The graph includes bars for bfs_kernel/rmat22, ubkernel, memory, and loop. The legend indicates the colors used for each kernel: ubkernel (red), memory (blue), and loop (green). The graph shows the throughput increasing with an increase in worklist size for each kernel.]
After sorting the worklist

![Graph showing throughput and worklist size for different strategies over rounds. The y-axis represents throughput in workitems per second (workitems/s), while the x-axis represents round numbers. The graph compares different strategies: memory, loop, sorted+memory, and sorted+loop. Each strategy is represented by different colored bars, with memory in blue, loop in green, sorted+memory in red, and sorted+loop in cyan. The legend is located at the top right of the graph. The x-axis ranges from 0 to 13 rounds, and the y-axis ranges from $10^5$ to $10^{11}$ for throughput and $10^0$ to $10^7$ for worklist size.]
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

Motivation

Queuing Models for Graph Programs

Results, Analytical Modelling and Characterization

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