Graph Applications

Social Network Analysis

Recommendations

Speech Recognition

Bioinformatics

CAD

Webpage Layout
Research Ongoing At All Levels

Applications

Algorithms

Implementation

Platform
Need More Informative Metrics

- **Time**
  - is often the the most important
  - requires other parameters matched

- **Traversed edges per second (TEPS)**
  - a rate, so can compare different inputs
  - confusion about what counts as a TE
Need More Informative Metrics

- Time
  - is often the the most important

Time & TEPS only quantify which is fastest, no insight into why

- a rate, so can compare different inputs

- confusion about what counts as a TE
Graph Performance Factors

1. Algorithms - how much work to do
2. Cache utility - how much data to move
3. Memory bandwidth - how fast data moves

For measurements: [Beamer, IISWC, 2015]
Graph Algorithm Iron Law (GAIL)

\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{time}}{\text{mem. req.}}
\]
annotate code to count edges traversed

\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{time}}{\text{mem. req.}}
\]

use performance counters to access total # of memory requests
Graph Algorithm Iron Law (GAIL)

Role: algorithm designer

\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{time}}{\text{mem. req.}}
\]

Metric: algorithmic intensity
Graph Algorithm Iron Law (GAIL)

**Role:** implementor

\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{mem. req.}}{\text{time}}
\]

**Metric:** cache utility
Graph Algorithm Iron Law (GAIL)

**Role:**

\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{time}}{\text{mem. req.}}
\]

**Metric:**

DRAM BW utilization

**System Designer**
Comparing BFS Implementations

- 3 BFS Approaches
  - **Naive** - classic top-down
  - **Bitmap** - uses bitmaps to reduce communication
  - **Direction-optimizing** - algorithmically does less

- Time doesn’t explain speedup

Kronecker SCALE=27, 32 threads, Ivy Bridge
BFS Analyzed by GAIL

\[
\text{time}_{\text{kernel}} = \frac{\text{edges}_{\text{kernel}}}{\text{mem. req.}_{\text{edge}}} \times \frac{\text{mem. req.}_{\text{mem. req.}}}{\text{time}_{\text{mem. req.}}}
\]
BFS Strong Scaling Analyzed by GAIL

Kronecker

USA Roads

Metric Normalized

Cores

Cores

Time  Memory Requests / Edge  ns / Memory Request
Delta-Stepping Analyzed by GAIL

Single-source shortest paths algorithm

~Dijkstra
Work Efficiency

1

$\Delta$ Parameter

∞

~Bellman-Ford
Parallelism

tradeoff
Delta-Stepping Analyzed by GAIL

USA roads, 8 threads, Ivy Bridge
GAP Benchmark Suite

- Benchmark Specifications (technical report)
  - standardize input graphs and rules
  - allows other implementations to compare
- Portable, high-quality baseline code
  - Only requirement is C++11 & OpenMP
  - Built in testing to verify results

gap.cs.berkeley.edu
\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{time}}{\text{mem. req.}}
\]
\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{time}}{\text{mem. req.}}
\]

- GAIL concisely breaks down performance
  - useful as a starting point for **introspection**
  - useful as simple model to weigh **tradeoffs**

gap.cs.berkeley.edu
Acknowledgements

Research partially funded by DARPA Award Number HR0011-12-2-0016, the Center for Future Architecture Research, a member of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA, and ASPIRE Lab industrial sponsors and affiliates Intel, Google, Huawei, Nokia, NVIDIA, Oracle, and Samsung. Any opinions, findings, conclusions, or recommendations in this paper are solely those of the authors and does not necessarily reflect the position or the policy of the sponsors.
What Do GAIL Results Represent?

- GAIL results are for a particular execution
  - fixed: input, platform, implementation
  - changing any of above will change results
- Focused on single-server shared-memory
- GAIL requirements
  - measure: runtime, traversed edges, memory requests
  - algorithm has notion of “traversing” edge
Formal complexity analysis is helpful, but…

- Many algorithms’ runtime is input graph topology-dependent, but often difficult to model real-world graphs
- Hides many improvements to platform or implementation optimizations
- Can be overly pessimistic. Many algorithms with a slower worst-case performance much faster in practice
What About Other Platforms?

- GAIL is for single-server shared memory
- For other platforms, replace memory requests with equivalent bottleneck metric
  - Clusters: packets or bytes transmitted
  - Flash/HD: blocks read from storage
  - Cache-less (XMT): memory requests OK
Iron Law Reapplied

For CPUs:

\[
\frac{\text{time}}{\text{program}} = \frac{\text{insts.}}{\text{program}} \times \frac{\text{cycles}}{\text{inst.}} \times \frac{\text{time}}{\text{cycle}}
\]

For Graph Algorithms:

\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{time}}{\text{mem. req.}}
\]
Graph Algorithm Iron Law (GAIL)

\[
\frac{\text{time}}{\text{kernel}} = \frac{\text{edges}}{\text{kernel}} \times \frac{\text{mem. req.}}{\text{edge}} \times \frac{\text{time}}{\text{mem. req.}}
\]

\[
\frac{\text{mem. req.}}{\text{kernel}} = \frac{\text{data transferred}}{\text{cache line size}}
\]

\[
\frac{\text{time}}{\text{edge}} = \frac{1}{\text{TEPS}}
\]
## Evaluation Setup

<table>
<thead>
<tr>
<th>Graph</th>
<th># Vertices</th>
<th># Edges</th>
<th>Degree</th>
<th>Diameter</th>
<th>Degree Dist.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roads of USA</td>
<td>23.9M</td>
<td>58.3M</td>
<td>2.4</td>
<td>High</td>
<td>const</td>
</tr>
<tr>
<td>Web Crawl of .sk Domain</td>
<td>50.6M</td>
<td>1949.4M</td>
<td>38.5</td>
<td>Medium</td>
<td>power</td>
</tr>
<tr>
<td>Kronecker Synthetic Graph</td>
<td>128.0M</td>
<td>2048.0M</td>
<td>16.0</td>
<td>Low</td>
<td>power</td>
</tr>
</tbody>
</table>

- **Target Platform**
  - Dual-socket Intel Ivy Bridge 3.3 GHz
  - Socket: 8 cores with 25MB L3 cache
  - DRAM: 128 GB DDR3-1600