Multi-objective Optimization of Sparse Array Computations

*Una-May O’Reilly*

MIT Computer Science and Artificial Intelligence Laboratory

*Nadya Bliss, Sanjeev Mohindra, Julie Mullen, Eric Robinson*

MIT Lincoln Laboratory

September 22\(^{nd}\), 2009

This work is sponsored by the Department of the Air Force under Air Force contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.
• Problem Context
  – Performance gap exists for graph algorithms that enable knowledge extraction in decision support systems

• Problem Definition
  – Performance optimization of sparse algebra matrix computations (for graph algorithms)
  – Sparse Mapping and Routing Toolbox

• Solution Methodology
  – multi-objective genetic algorithm to optimize
  – Second objective complements first: find ideal balance of operations for nodes in architecture.
    Discernable from dependency graph

• Preliminary Results
• Future Work and Summary
Emerging Decision Support Trends

- Enormous growth in data size coupled with multi-modalities
- Increasing relevance in relationships between data/objects/entities
- Increasing algorithm & environment complexities
- Asymmetric & fast-evolving warfare
- Increasing need for knowledge processing

Focus on Top of the Pyramid: Knowledge Extraction and Intelligence
Knowledge Extraction Applications

**NETWORK DETECTION**
- Graph analysis for identifying interesting sub-networks within large noisy graphs*

**DATA FUSION**
- Bayesian networks for fusing imagery and ladar for better on board tracking

**TOPOLOGICAL DATA ANALYSIS**
- Higher dimension graph analysis to determine sensor net coverage

---

<table>
<thead>
<tr>
<th>APPLICATION</th>
<th>KEY ALGORITHM</th>
<th>KEY KERNEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network detection</td>
<td>Edge Betweenness Centrality</td>
<td>MATRIX MULT: A +.* B</td>
</tr>
<tr>
<td>Feature aided 2D/3D fusion</td>
<td>Bayesian belief propagation</td>
<td>MATRIX MULT: A +.* B</td>
</tr>
<tr>
<td>Dimensionality reduction</td>
<td>Minimal Spanning Trees</td>
<td>MATRIX MULT: X +.* A +.* Xᵀ</td>
</tr>
<tr>
<td>Finding cycles on complexes</td>
<td>Single source shortest path</td>
<td>D min.+ A</td>
</tr>
</tbody>
</table>

Many knowledge extraction algorithms are based on graph algorithms

*email network from http://www.mailchimp.com/blog/tag/social-networks/

*MIT Lincoln Laboratory*
Many graph algorithms can be expressed as *sparse array* computations.

**Graph preliminaries**

A graph $G = (V,E)$ where
- $V =$ set of vertices
- $E =$ set of edges

**Adjacency matrix representation:**
- Non-zeros entry $A(i,j)$ where there exists an edge between vertices $i$ and $j$

Example operation:
- Vertices reachable from vertex $v$ in $N$ or less steps can be computed by taking $A$ to the $N$th power and multiplying by a vector representing $v$.
The Graph Processing Performance Gap

- Current technologies do not provide *performance* or *power efficiency* for knowledge extraction applications.
- Emerging application trends require closing the performance gap.

- Gap arises due to *sparse and irregular* graph data.
- Mapping can be computed *ahead of algorithm deployment*.

Efficient data mapping will help close gap.
Outline

- Problem Context
- **Problem Definition**
- Solution Methodology
- Preliminary Results
- Future Work and Summary
SMaRT
Sparse Mapping and Routing Toolbox

HARDWARE ABSTRACTION
Detailed, topology-true hardware model
Fine-grained dependency analysis

PROGRAM ANALYSIS
while $f \neq 0$
do
  $d = d + 1$
  $p = p + f$
  $S(d, :) = f$
  $f - fA \times p$

MAPPPING ALGORITHM
GA proposes maps
GA does selection on maps depending on time
GA recombines maps to form new maps

FITNESS EVALUATION

GA proposes routes
GA does selection on route choices depending on time
GA recombines route choices to form new route choices

OUTPUT MAPS
A map for an array is an assignment of blocks of data to processing nodes

SUPPORT FOR IRREGULAR DATA DISTRIBUTIONS

PROGRAM ANALYSIS OUTPUT MAPS

HPEC-09-8
U.M. O'Reilly 10/1/2009

MIT Lincoln Laboratory
The Mapping Optimization Problem

**Given**

\[
\text{function } \text{bc} = \text{vertexBCbatch}(\text{roots, numRoots}) \\
\text{nspl} = 1 : \text{numRoots}, \text{roots} = 1; \\
\text{depth} = 0; \\
\text{fringe} = \text{nspl} .* \text{A}; \\
\text{while } \text{nnz}(\text{fringe}) > 0 \\
\text{depth} = \text{depth} + 1; \\
\text{nspl} = \text{nspl} + \text{fringe}; \\
\text{bfs}[\text{depth}] = \text{fringe} > 0; \\
\text{fringe} = \text{fringe} + *\text{A} .* (\text{nspl} .xor 1); \\
\text{bcu}(:, :) = 0; \\
\text{for } \text{depth} = \text{depth} - 1.2 \\
\text{w} = \text{bfs}[\text{depth}] ./ \text{nspl} .* (\text{bcu} + 1); \\
\text{w} = \text{A} .* \text{w}; \\
\text{w} = \text{w} .* \text{bfs}[\text{depth} - 1] .* \text{nspl}; \\
\text{bcu} = \text{bcu} .+ \text{w}; \\
\text{bc} = \text{bc} .+ (+\text{bcu}); \\
\]

**PROGRAM ANALYSIS**

**Find**

\[
\arg \min_M f(T, H, M) \\
\text{Sample objectives, } \\
\text{Execution latency or FLOPS} \\
\text{Power (maximize operations/Watt)} \\
\text{Efficiency, etc} \\
\]

**Evaluation of the objective function requires performance prediction**

MIT Lincoln Laboratory
Mapping Optimization Challenges

Mapping is NP-complete

Network Coding $\leq_p$ Mapping
with Muriel Médard, MIT EECS

K-Clique $\leq_p$ Mapping
with Ben Miller, LL Gr 102

The search space of maps is extremely large:

Size of the mapping search space: $S_M = N_P^B$

The objective function is a simulation: values are discrete and presumably non-convex

A global search technique (such as a genetic algorithm) is well-suited to mapping
Outline

• Problem Context
• Problem Definition
• Solution Methodology
• Preliminary Results
• Future Work and Summary
**Neo-darwinian evolution**

- Population adaptation to an environment
- Through biased selection based upon fitness of organism
- Through genetic inheritance, random recombination and variation

**Evolution is a search-based optimization process**

- Organism is a candidate solution to the environment
- Fitness of organism expresses performance on objective
- Adaptation is a search process that exploits and explores
- The search proceeds in parallel via the population
Genetic Algorithm for Map Optimization

Mapping Optimization Algorithm

- GENETIC ALGORITHM
- GA PROPOSES MAPS
- FITNESS EVALUATION
- GA SELECTS BETTER MAPS
- GA COMBINES AND VARIES MAPS

Performance = Operations or Execution Latency

Mapping space: arbitrary maps with fixed minimum block size

Routing space: all-pairs all-paths

Recombination

Before 11011101
After 11011101
Variation

MIT Lincoln Laboratory

HPEC-09-13
U.M. O'Reilly 10/1/2009
Dependency graph (DG) is input to simulator and expresses where the data is mapped, how the data is routed between processors, what computations execute on each processor. Topological sort of DG indicates what operations can proceed in parallel. DG is complete specification of computation on the studied architecture.

Dependency graph is tightly coupled with performance.
Outline

• Problem Context
• Problem Definition
• Solution Methodology
• Preliminary Results
• Future Work and Summary
Analysis of Dependency Graph Characteristics

Performance is strongly related to DG

Knowledge of parallelization suggests a knee in the curve at certain degree of complexity

Ways to Define Balance
• Balance of CPU operations on nodes
• Balance of memory operations on nodes
• Average degree of concurrency
• Distribution of degree of concurrency

A multi-objective genetic algorithm can co-optimize map performance and balance
Co-optimization: Pareto Dominance

Better: $A > B$
Map $A$ performs faster
imbalance of $A$ is lower

“A dominates $B$”
$A$’s map and balance
are both better than $B$’s

Non Dominated
$A$’s map is better but
$B$’s balance is better

Or $B$’s map is better but
$A$’s balance is better

No solution is better on
both map and balance

Co-optimization front also known
as estimated pareto front

Comparison of each population member
Complexity $O(mN^2)$
Using comparison info to sort the fronts
Complexity $O(N^2)$
$N$=population size, $m$ = number of objectives
Experimental Setup

**Algorithm**

- Scrambled Powerlaw
- Scrambled Powerlaw

Hybrid Inner-Outer Product

**Architecture**

- Network Latency: 50e-9 seconds
- Network Bandwidth: 5e9 bytes/sec
- Memory Latency: 50e-9 seconds
- Memory Bandwidth: 12e9 bytes/sec
- CPU Rate: 5e9 ops/sec

4x4x4 Torus Topology

**Mappers**

- Baseline
- Anti-Diagonal Block Cyclic
- XO/Mutation Rate

- Multi-Objective Genetic Algorithm

  **Parameters:**
  - Population: 100
  - Generations: 30
  - Selection: 1/5 Pop.

  **Objectives:**
  - Performance
  - Memory Balance

- Random Sample

  Operation Balance
  Varied Grids
Baseline ADBC mapping is outperformed by Multi-Objective Genetic Algorithm.
Co-optimization (MOGA) Results

Best solution is rightmost on performance (x-axis)

Over the run, the non-dominated front migrates toward solutions with better memory balance and performance

Non-dominated front never becomes singular indicating co-optimization is beneficial

Mean memory imbalance decreases over time under co-optimization objectives (while performance improves)

Complexity of best map fluctuates
Network bandwidth parameters:
- Bandwidth $[10^{-1}, 10^0, 10^1]$
- Hardware model affects the characteristics of the objective function

<table>
<thead>
<tr>
<th>Hardware Model</th>
<th>FLOPS Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$ Network Model</td>
<td>34.1%</td>
</tr>
<tr>
<td>$10^0$ Network Model</td>
<td>26.4%</td>
</tr>
<tr>
<td>$10^1$ Network Model</td>
<td>13.0%</td>
</tr>
</tbody>
</table>
Future Work

• Co-optimization objective should reflect relation between algorithm and structure of architecture
  – Knowledge-based analysis: Consider metrics of parallelism of program or graph
  – Statistical Analysis: Regress relationship between properties and performance from a sample of maps on the architecture

• Power co-optimization (in conflict with FLOPS) via the multi-objective, pareto-based Genetic Algorithm
Summary

- Graph algorithms expressed in linear algebra expose a map optimization problem
  - Map optimization can be improved by co-optimizing the performance and algorithm complexity with a multi-objective GA
- Better maps close the performance gap of graph algorithms
- Improved performance of graph algorithms addresses challenges of rapid knowledge extraction
- Rapid knowledge extraction enables effective decision support
END