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

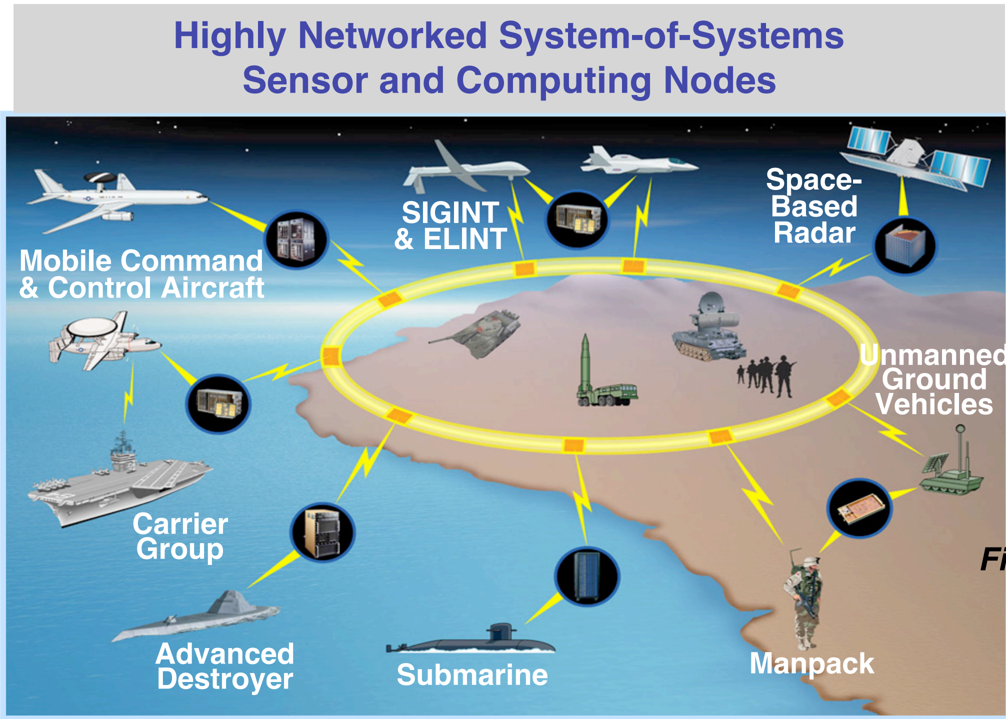
September 22nd, 2009

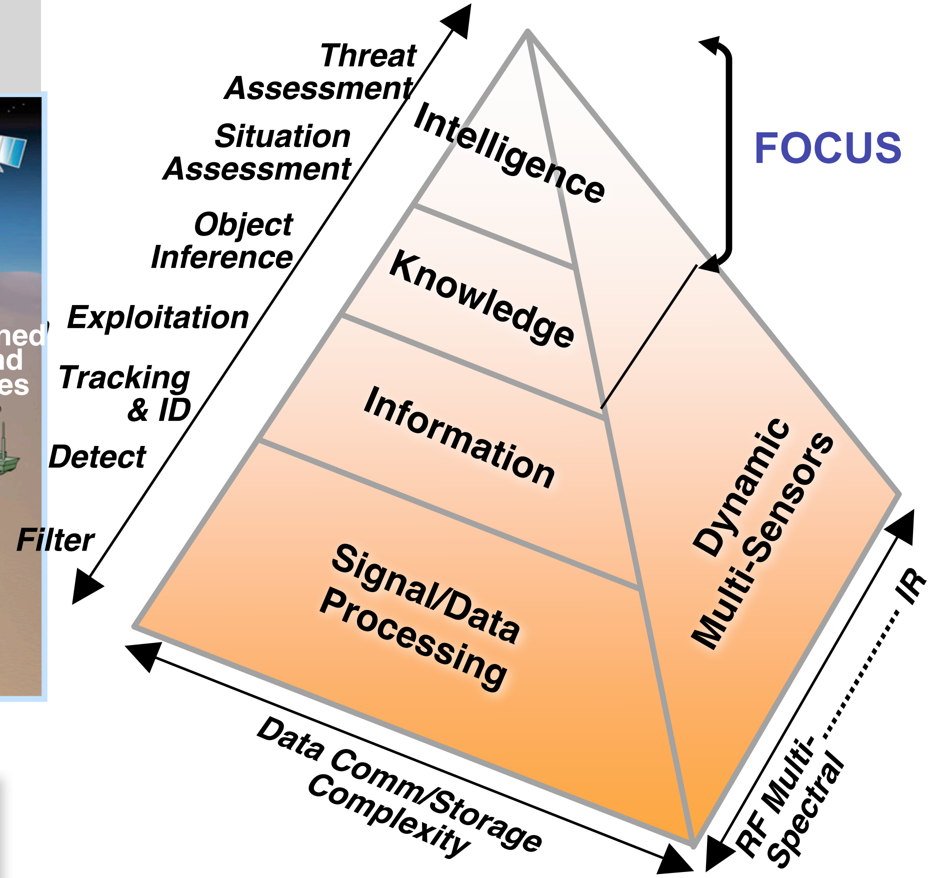
This work is sponsored by the Department of the Air Force under Air Force contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

Outline

- 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





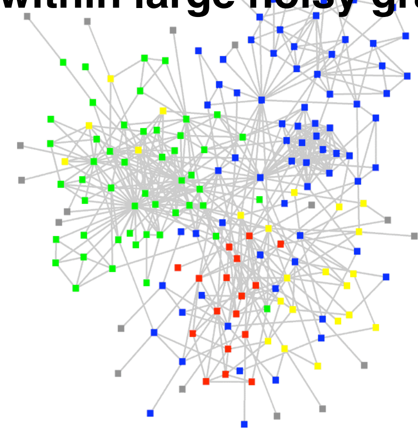
- Enormous growth in data size coupled with multi-modalities
- Increasing relevance in relationships between data/objects/entities
- Increasing algorithm & environment complexities
- Asymmetric & fast-evolving warfare
- Increasing need for knowledge processing

Focus on Top of the Pyramid: Knowledge Extraction and Intelligence

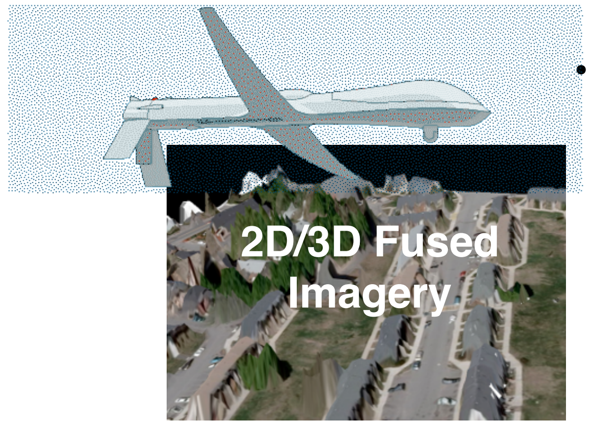
Knowledge Extraction Applications

NETWORK DETECTION

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



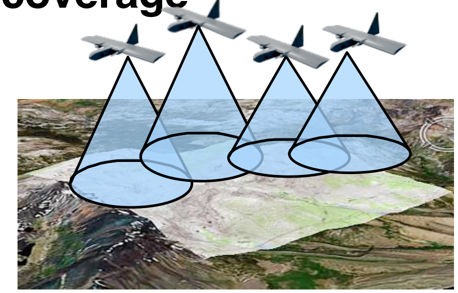
DATA FUSION

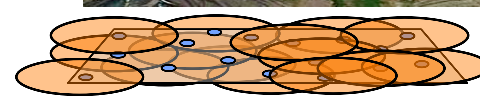


- Bayesian networks for fusing imagery and ladar for better on board tracking

TOPOLOGICAL DATA ANALYSIS

- Higher dimension graph analysis to determine sensor net coverage





*A. Tahbaz Salehi and A. Jadbabaie, *Distributed coverage verification in sensor networks without location information*

APPLICATION →	KEY ALGORITHM →	KEY KERNEL
• Network detection	• Edge Betweenness Centrality	MATRIX MULT: A +.* B
• Feature aided 2D/3D fusion	• Bayesian belief propagation	MATRIX MULT: A +.* B
• Dimensionality reduction	• Minimal Spanning Trees	MATRIX MULT: X +.* A +.* X^T
• Finding cycles on complexes	• Single source shortest path	D min.+ A

Many knowledge extraction algorithms are based on graph algorithms

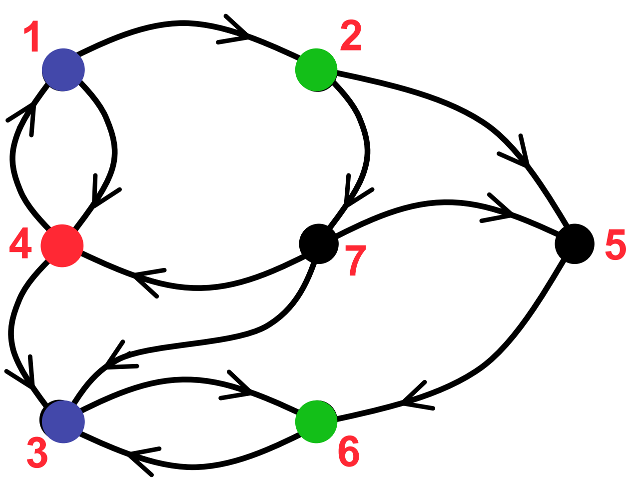
*email network from http://www.mailchimp.com/blog/tag/social-networks/

Fundamental Observation
-Graph-Sparse Matrix Duality-

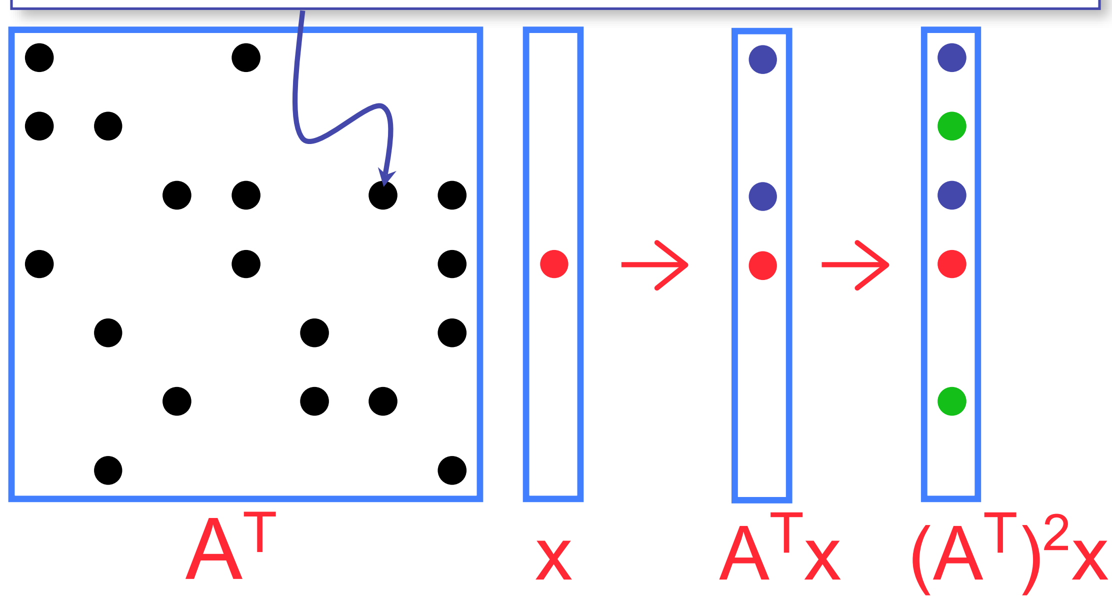
Many graph algorithms can be expressed as *sparse array* computations

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

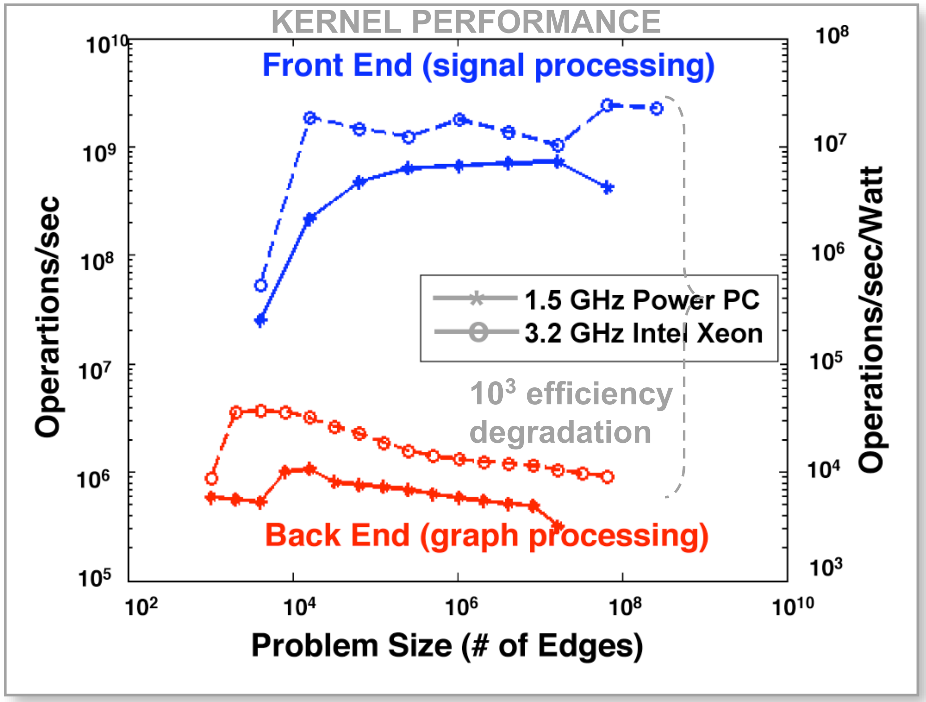


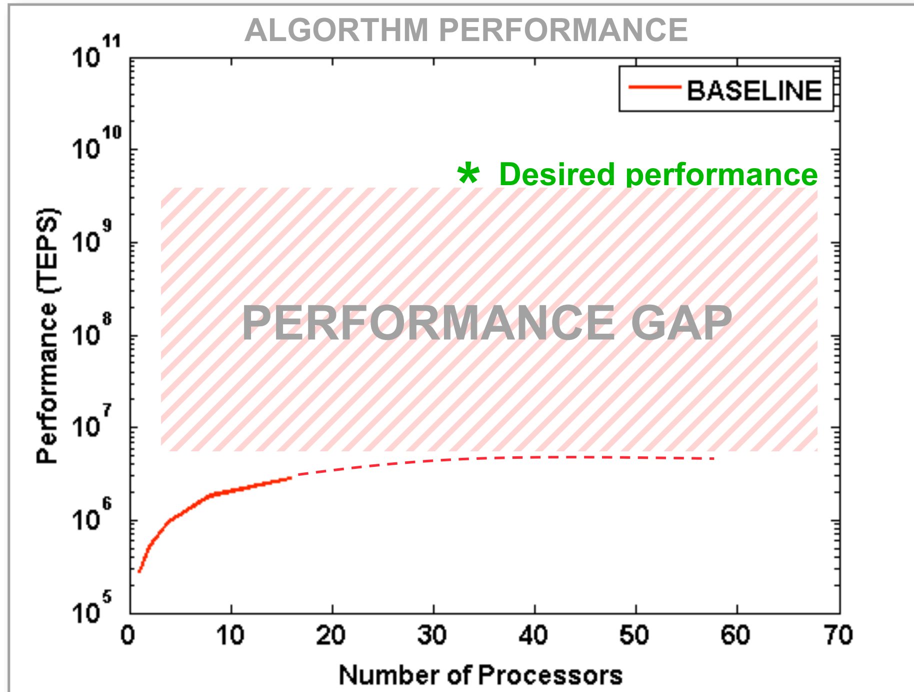
Graph G:



Example operation:
- Vertices reachable from vertex v in N or less steps can be computed by taking A to the Nth 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

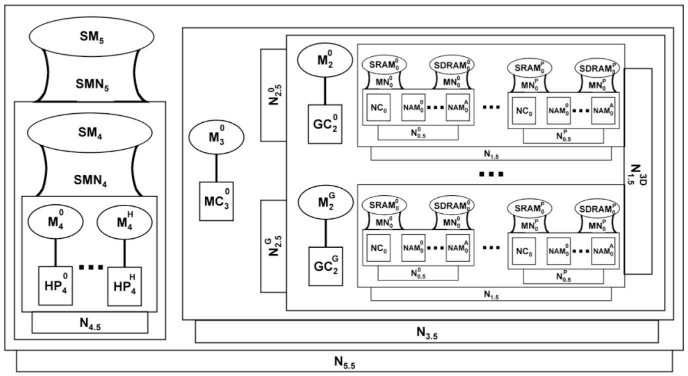
Efficient data mapping will help close gap

Outline

- Problem Context
- **Problem Definition**
- Solution Methodology
- Preliminary Results
- Future Work and Summary

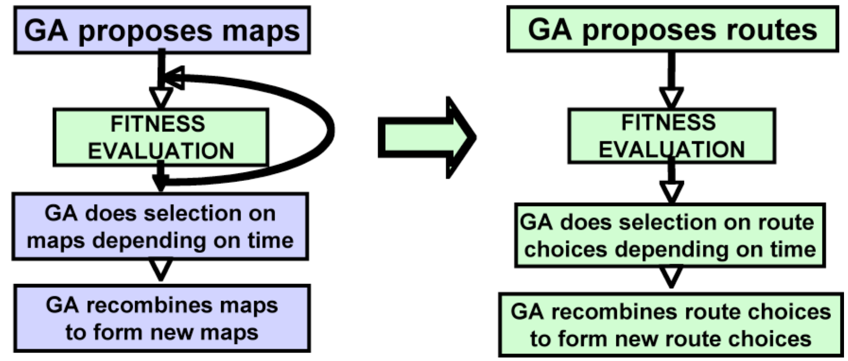
SMaRT
Sparse Mapping and Routing Toolbox

HARDWARE ABSTRACTION

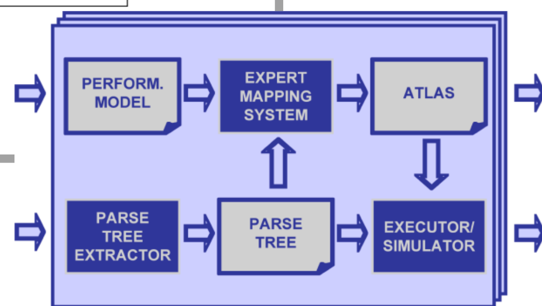


Detailed, topology-true hardware model

MAPPING ALGORITHM



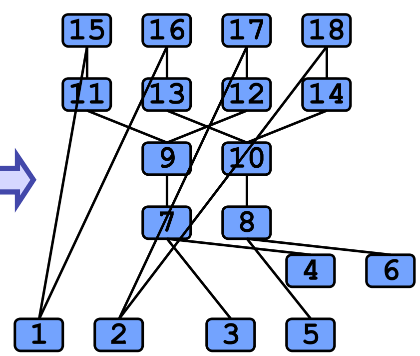
Stochastic search for mapping and routing



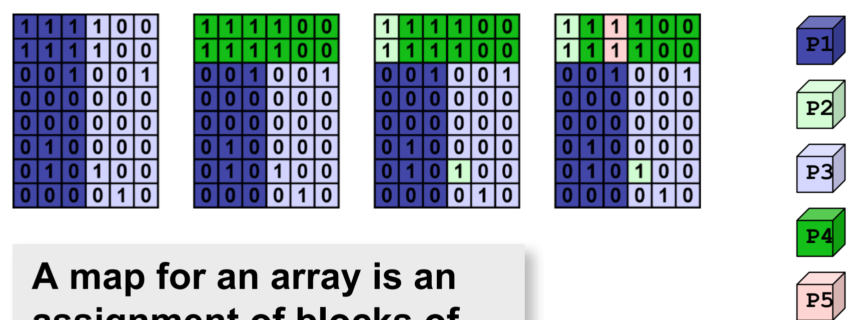
Fine-grained dependency analysis

Support for irregular data distributions

$$
\begin{array}{l}
\text{while } f \neq 0 \\
\quad \text{do} \\
\qquad d = d + 1 \\
\qquad p = p + f \\
\qquad S(d,:) = f \\
\qquad f = fA \times \neg p
\end{array}
$$



PROGRAM ANALYSIS



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

OUTPUT MAPS

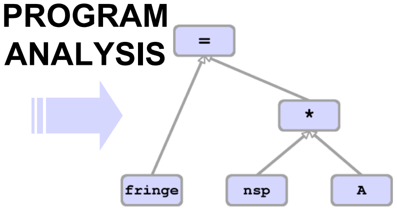
The Mapping Optimization Problem

Given

```
function bc = vertexBCbatch(roots,numRoots)
nsp(1:numRoots,roots) = 1;
depth = 0;
fringe = nsp +.* A;
while nnz(fringe) > 0
   depth = depth + 1;
   nsp = nsp .+ fringe;
   bfs[depth] = fringe .> 0;
   fringe = fringe +.* A .* (nsp .xor 1);
bcu(:,:) = 0;
for depth = depth:-1:2
   w = bfs[depth] ./ nsp .* (bcu .+ 1);
   w = A +.* w;
   w = w .* bfs[depth-1] .* nsp;
   bcu = bcu .+ w;
bc = bc .+ (+.bcu);
```

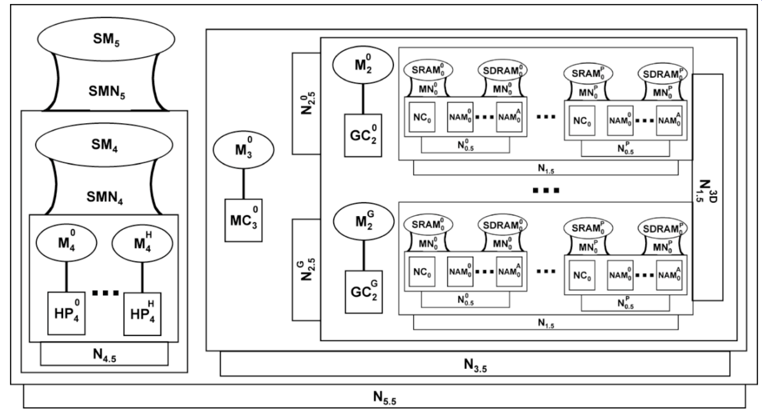
ALGORITHM CODE

PROGRAM ANALYSIS



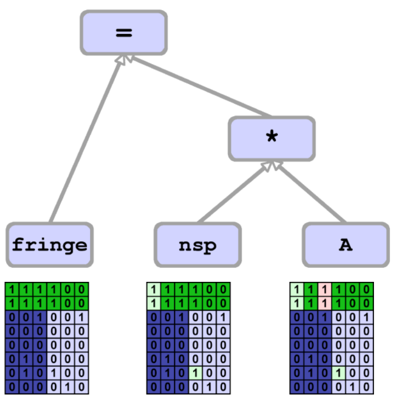
PARSE TREE, T

,



HARDWARE MODEL, H

Find



SET OF MAPS, M

Such that: a performance objective is optimized

$$argmin_M f(T,\ H,\ M)$$

Sample objectives, f

- **Execution latency or FLOPS**
- **Power (maximize operations/Watt)**
- **Efficiency, etc**

Evaluation of the objective function requires performance prediction

Mapping Optimization Challenges

Mapping is NP-complete

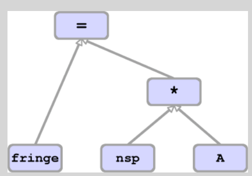
Network Coding $\leq_P$ Mapping *with Muriel Médard, MIT EECS*	K-Clique $\leq_P$ Mapping *with Ben Miller, LL Gr 102*

The search space of maps is extremely large:

Size of the mapping search space: $S_M = N_P{}^{(B)}$

$N_P =$ number of processing nodes

$B =$ number of blocks



$$B = \sum_{i=1}^{N_A} B_i$$

number of arrays in T

number of blocks in array i

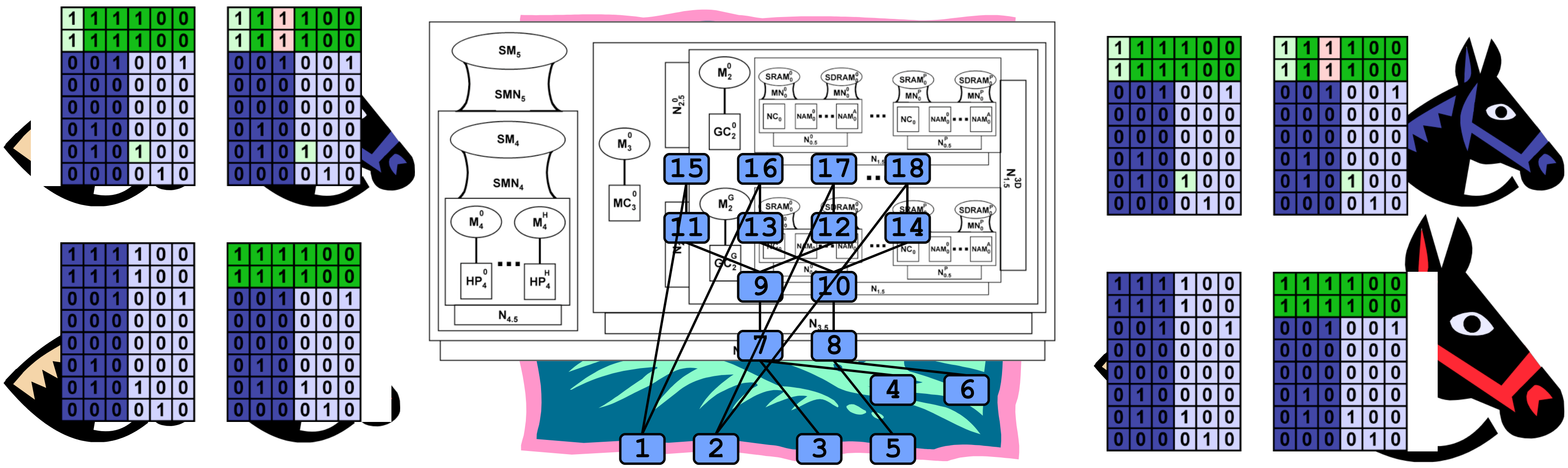
The objective function is a simulation: values are discrete and Presumably non-convex

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

Genetic Algorithm Concepts

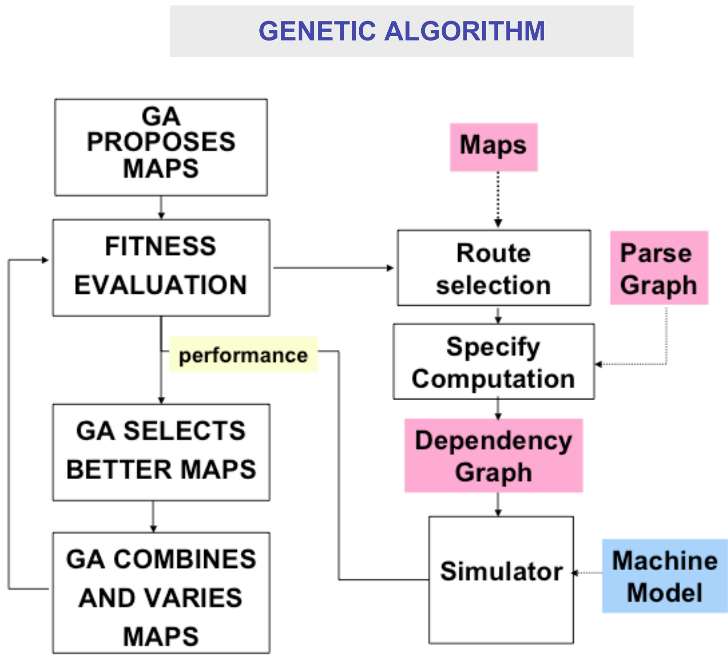


Neo-darwinian evolution
- Population adaptation to an environment
- Through biased selection based upon fitness of organism
- through genetic inheritance, random recombination and variation

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

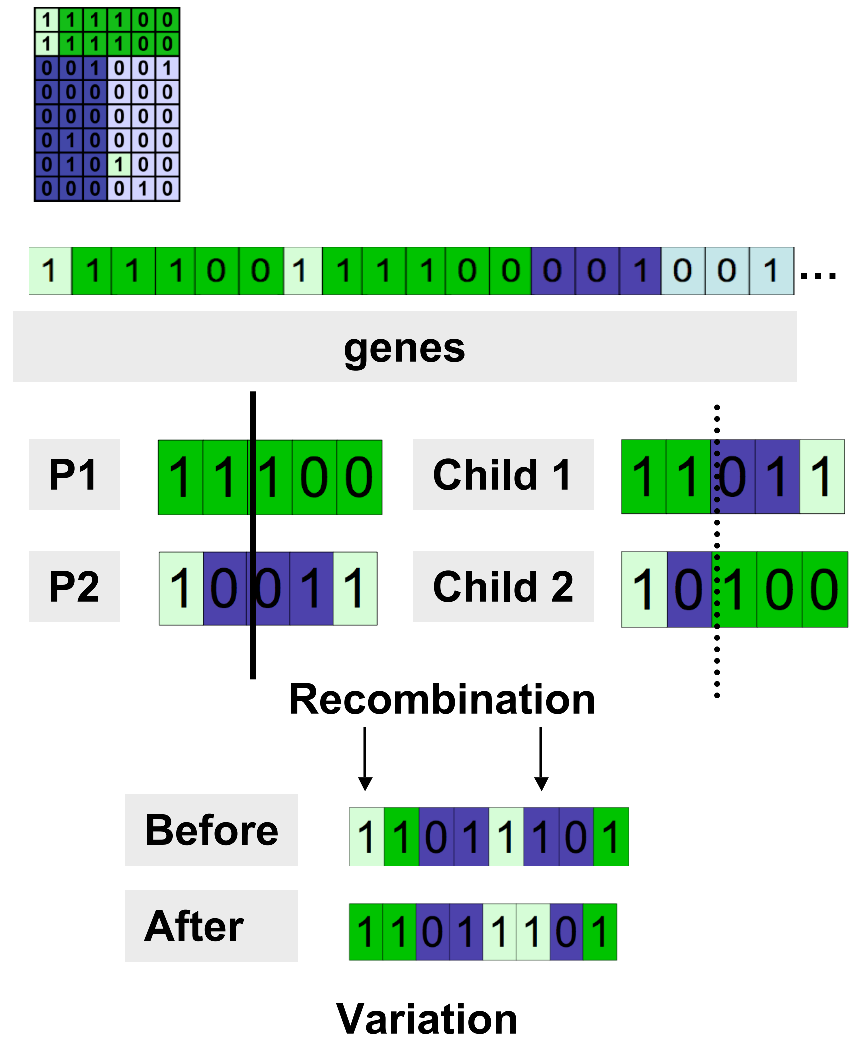
Mapping Optimization Algorithm



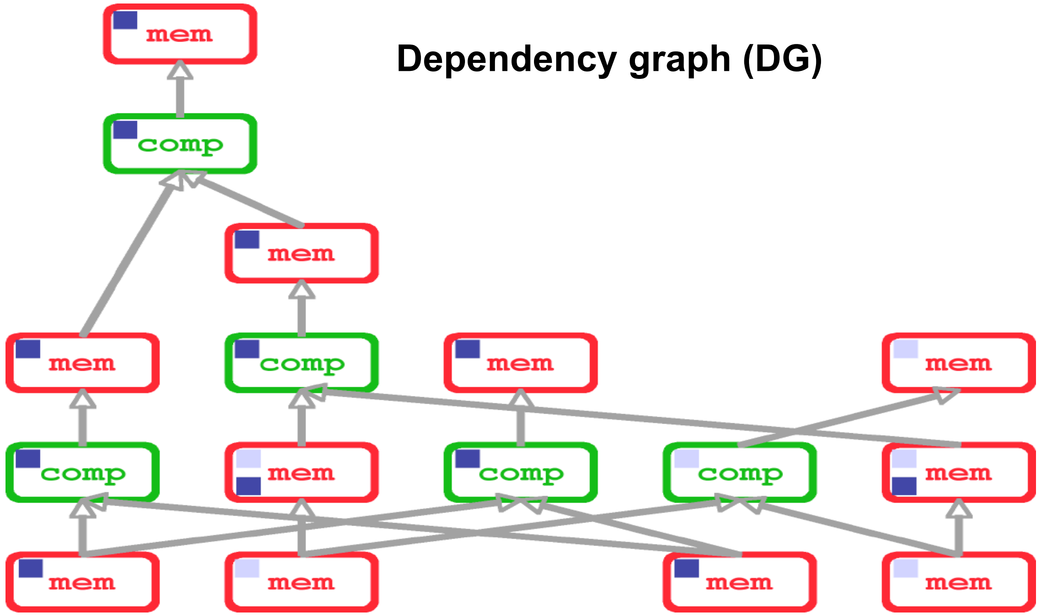
Performance =
Operations or Execution Latency

Mapping space: arbitrary maps with fixed minimum block size

Routing space: all-pairs all-paths



Dependency Graph



Dependency graph (DG)

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

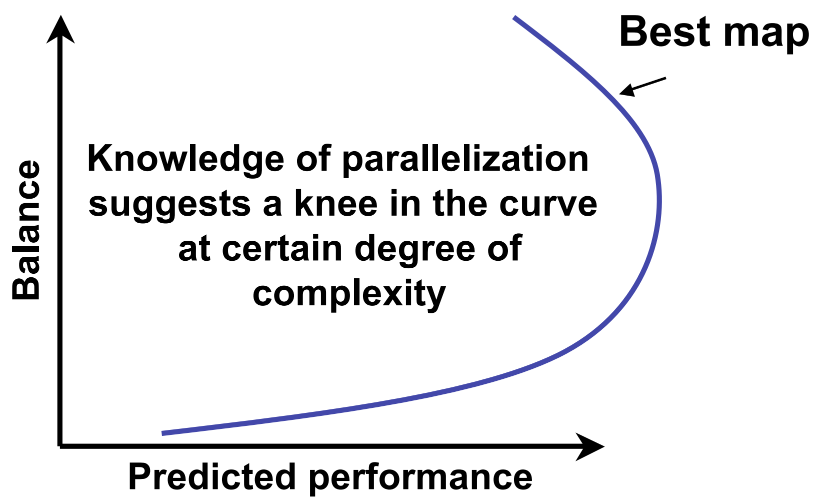
Dependency graph is tightly coupled with performance

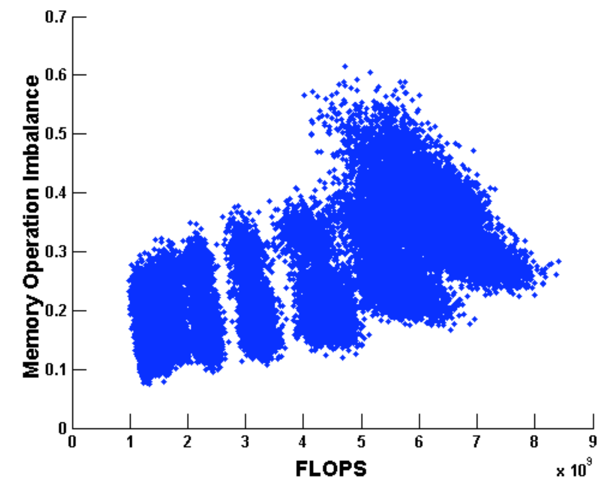
Outline

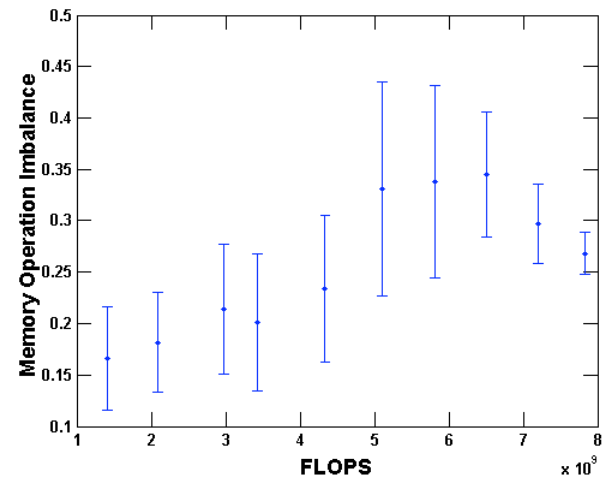
- Problem Context
- Problem Definition
- Solution Methodology
- **Preliminary Results**
- Future Work and Summary

Analysis of Dependency Graph Characteristics

Performance is strongly related to DG







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**

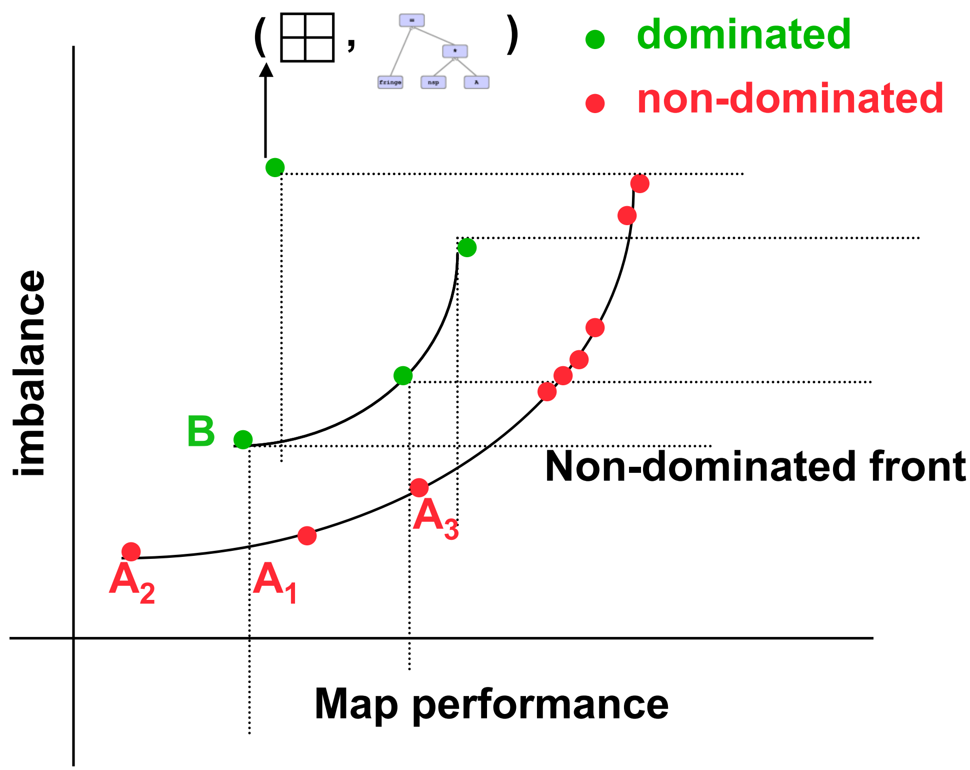
"A dominates B"
A's map and balance
are **both better** than B's

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

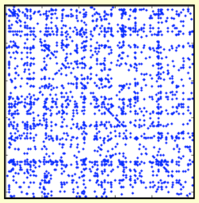
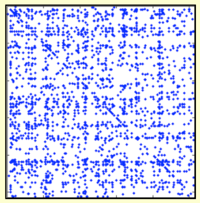


Comparison of each population member
Complexity O(mN^2)
Using comparison info to sort the fronts
Complexity O(N^2)
N=population size, m = number of objectives

Experimental Setup

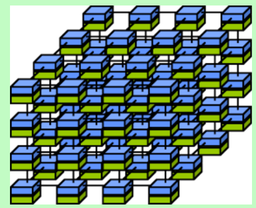
Algorithm

Scrambled Powerlaw X **Scrambled Powerlaw**

Hybrid Inner-Outer Product

Architecture

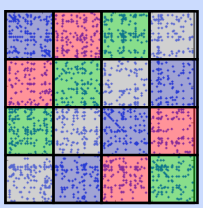


Network Latency	50e-9 seconds
Network Bandwidth	5e9 bytes/sec
Memory Latency	50e-9 seconds
Memory Bandwidth	12e9 bytes/sec
CPU Rate	5e9 ops/sec

4x4x4 Torus Topology

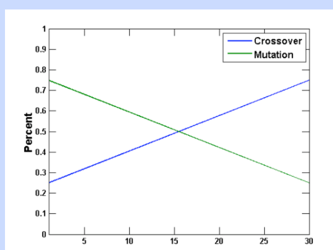
Mappers

Baseline



Anti-Diagonal Block Cyclic

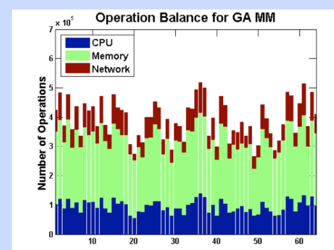
Multi-Objective Genetic Algorithm



XO/Mutation Rate

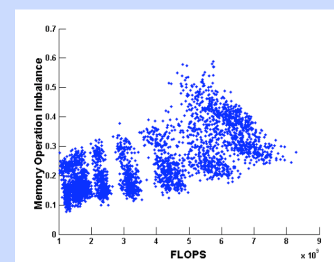
Parameters:
Population: 100
Generations: 30
Selection: 1/5 Pop.

Objectives:
Performance
Memory Balance



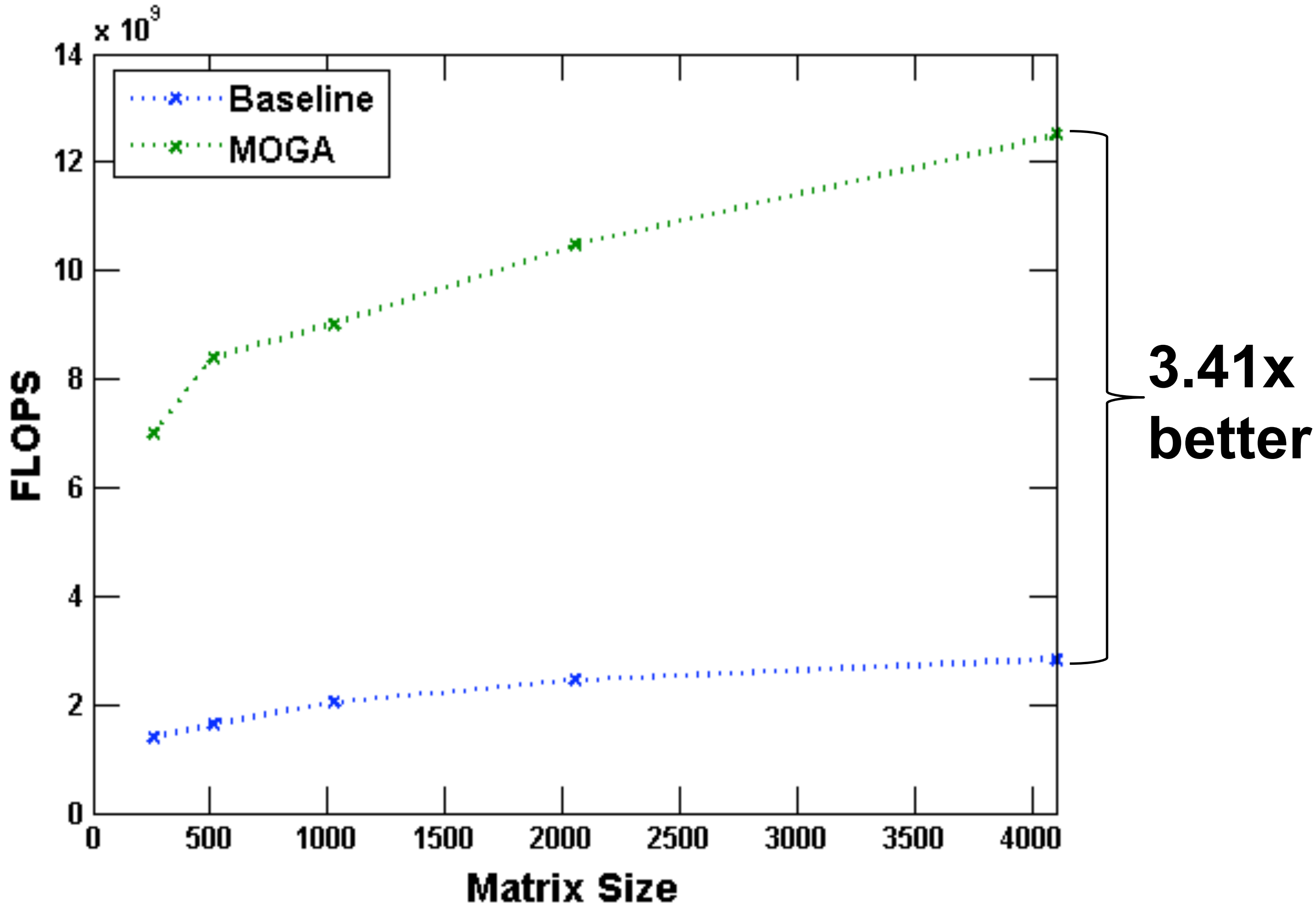
Operation Balance

Random Sample



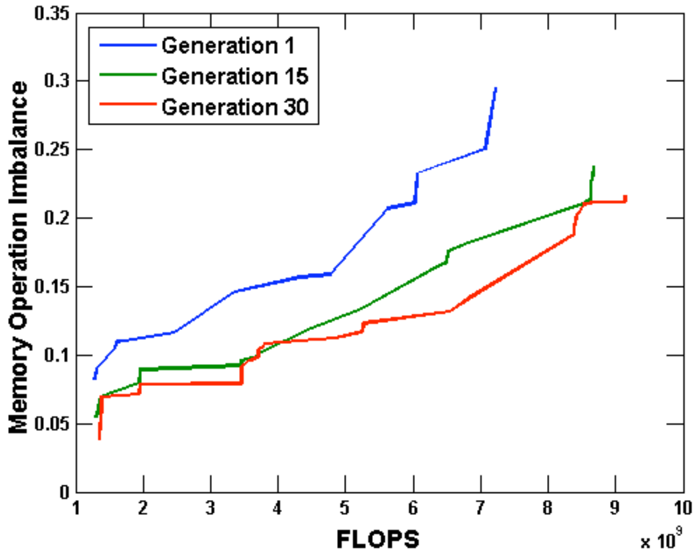
Varied Grids

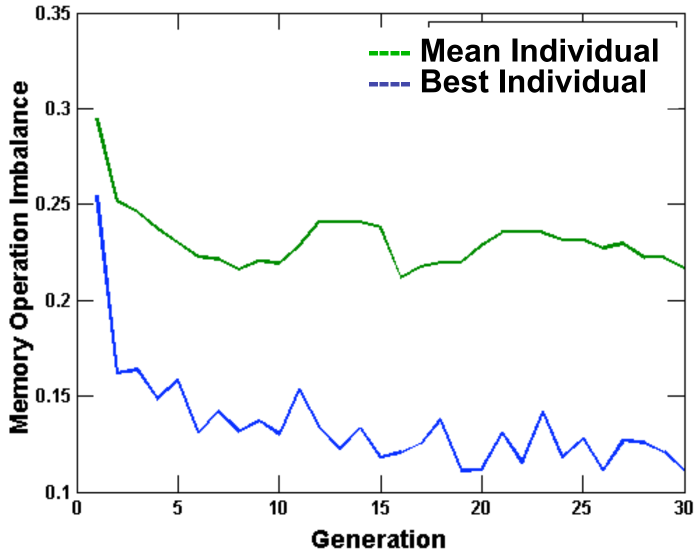
Optimization Algorithm Comparison



Baseline ADBC mapping is outperformed by Multi-Objective Genetic Algorithm

Co-optimization (MOGA) Results





Best solution is rightmost on performance (x-axis)

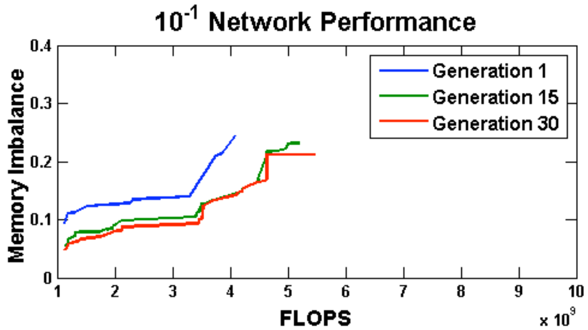
Over the run, the non-dominated front migrates toward solutions with better memory balance and performance

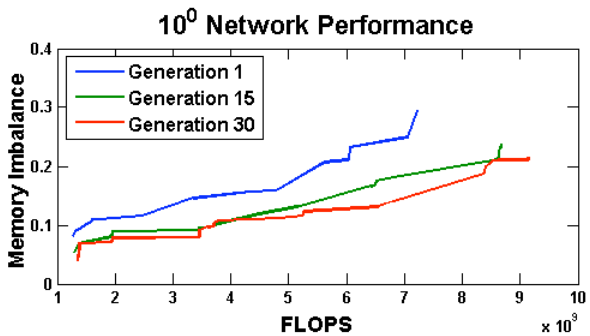
Non-dominated front never becomes singular indicating co-optimization is beneficial

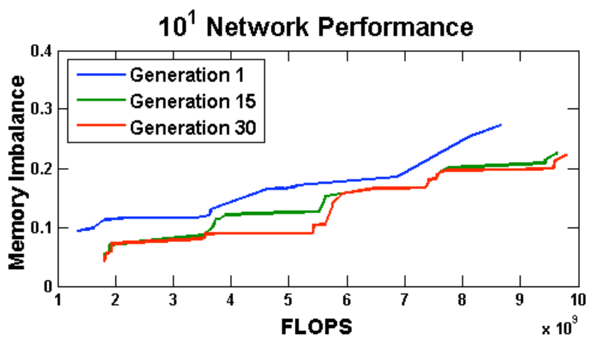
Mean memory imbalance decreases over time under co-optimization objectives (while performance improves)

Complexity of best map fluctuates

Hardware Model Parametric Study

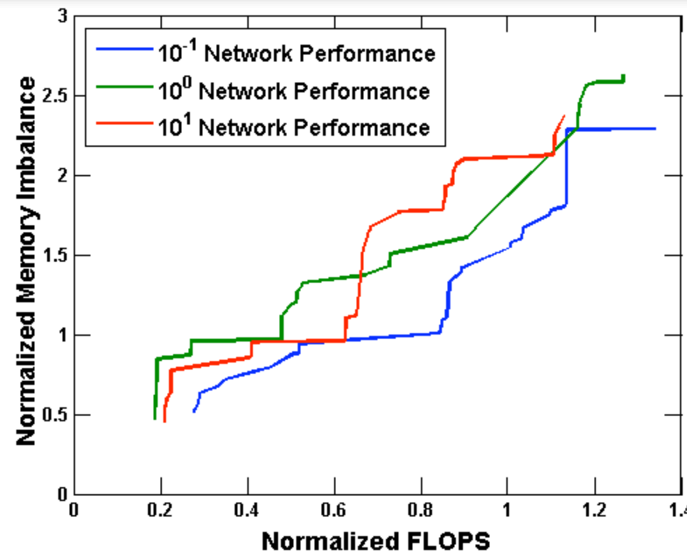






Network bandwidth parameters:

- **Bandwidth*[10^{-1} 10^{0} 10^{1}]**
- **Hardware model affects the characteristics of the objective function**



Hardware Model	FLOPS Improvement
10^{-1} Network Model	34.1%
10^{0} Network Model	26.4%
10^{1} Network Model	13.0%

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