Using Genetic Algorithms to solve the Minimum Labeling Spanning Tree Problem

Final Presentation

Oliver Rourke, oliverr@umd.edu

Advisor: Dr Bruce L. Golden, bgolden@rhsmith.umd.edu
R. H. Smith School of Business

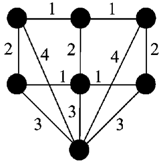
Introduction to Minimum Labelling Spanning Tree Problem (MLST)

- Combinatorial optimization problem first proposed in 1996 [Chang:1996]
- Connected Graph - set of vertices and edges.
- Each edge has a label
- Find the smallest set of labels which gives a connected sub-graph

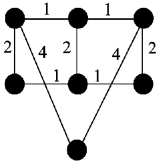
An example of a labelled spanning tree, and subgraphs

Question: What is the smallest set of labels which induces a connected (sub-) graph?

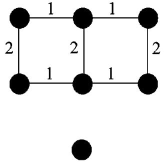
Complete Graph G



Subgraph induced by {1, 2, 4} - Connected



Subgraph induced by {1, 2} - Not Connected



More about MLST

- NP-complete - 'perfect' algorithm impossible (?)
- Many heuristics have been used including:
  - Variable Neighborhood Search (VNS) - Best
  - Simulated Annealing
  - Pilot Method
  - Reactive Tabu Search

Introduction to Genetic Algorithms (GAs)

- Evolutionary-inspired heuristic for optimization problems

Introduction to Genetic Algorithms (GAs)

- Evolutionary-inspired heuristic for optimization problems
- Population = set of (valid) solutions
- Select, Breed, Replace

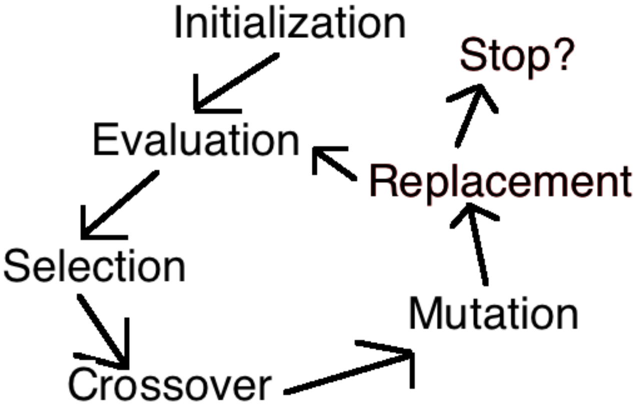
Introduction to Genetic Algorithms (GAs)

- Evolutionary-inspired heuristic for optimization problems
- Population = set of (valid) solutions
- Select, Breed, Replace
- Advantages:
  - Flexible and adaptable
  - Robust performance at global search
  - Simple to parallelize

One Parameter GA for MLST - Serial

- From Xiong, 2005
- Designed to be simple - no fine tuning
- One parameter - p, population size
- Solution: List of labels (gives connected sub-graph)
- Gene: Label in the list
- Modified Genetic Algorithm (MGA), Xiong, 2006 - more intelligent crossover operator

GA: Overview


Initialization
Stop?
Evaluation
Replacement
Selection
Mutation
Crossover

GA Improvements

- 1: Coin toss: Make crossover/mutation stochastic
- 2: Keep equally fit offspring over parents
- 3: Favor mutation: Encourage retention of new material

Databases

Two distinct databases:

- 36 sets of instances (10 instances per set) from Cerulli et al. [2005], smaller graphs. Used for validation only.
- 5 sets of 100 randomly generated larger graphs (using technique from Xiong, 2005). Each set has 100 nodes, 0.2 edge density and either 25, 50, 100, 250 or 500 labels.

Serial Testing

- Conducted on Genome cluster at UMD
- All experiments using one processor, instances run sequentially
- Tested on own generated databases (same as above) with N = 100, p = 0.2, L = 25, 50, 100, 250, 500
- Run 10 times with different random number seeds
- Stop: Max running time (L*20ms per instance) vs. Max generation count

Serial GA changes - results

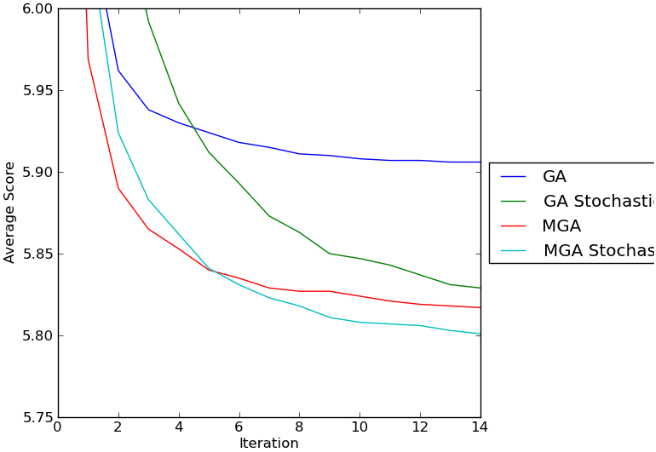
- GA + variants:

Algorithm	% Above BKS	% Above Optimum (if known)	Time (s)
Original GA	6.97	3.79	129.3
Crossover Coin toss	4.7	2.3	139.8
Mutation Coin Toss	5.1	3.0	130.5
Keep Equal	8.3	5.1	123.1
Favor Mutation	5.1	2.4	130.11
Everything	3.9	2.4	139.7

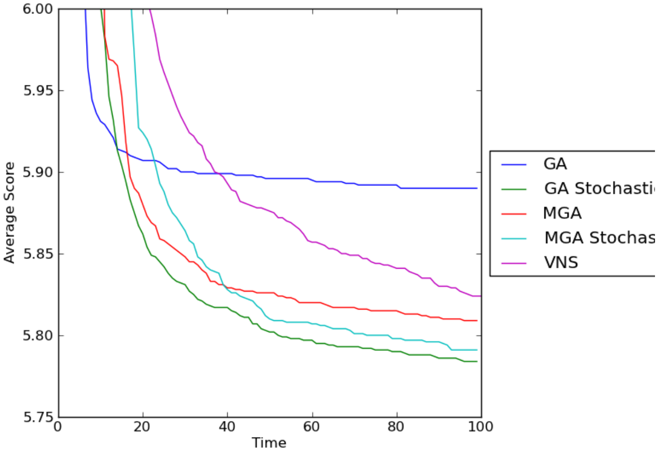
- Xiong's Modified GA (MGA) + variants

Algorithm	% Above BKS	% Above Optimum (if known)	Time (s)
MGA	4.1	2.2	474.8
MGA with Everything	2.9	1.5	421.1

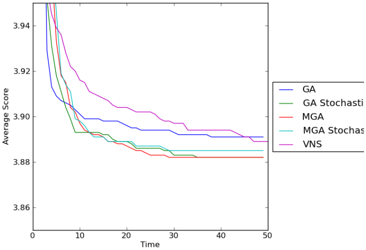
Results vs Iteration count (L=50)



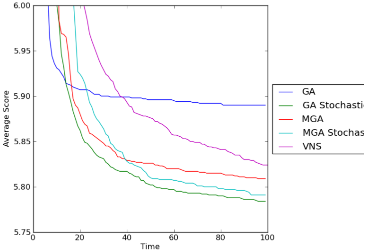
Results vs Time (L=50)



All Results vs Time



(a) L=25



(b) L=50

All Results vs Time

Genetic Algorithm for the MLST

Oliver Rourke

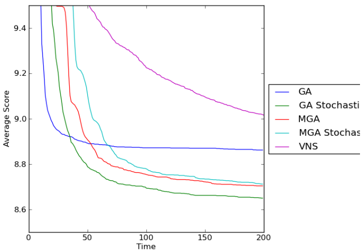
The MLST

Genetic Algorithms

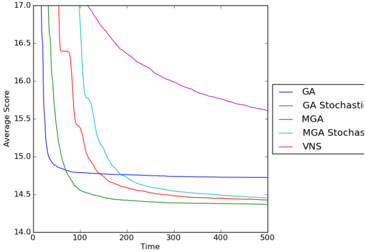
My Serial GAs

Parallel GAs

Software/ Validation etc...

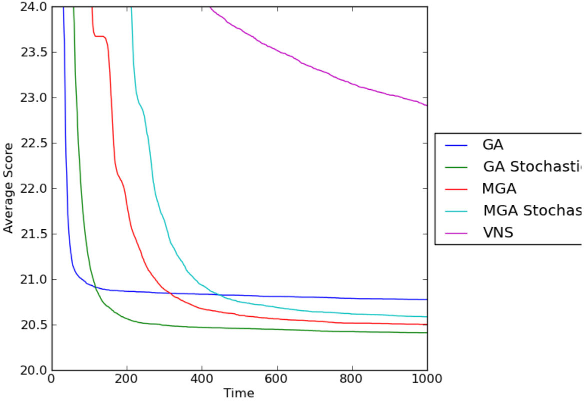


(c) L=100



(d) L=250

All Results vs Time



(e) L=500

Serial Algorithm $->$ Parallel Algorithm

- Why?
  - Speedup
  - Larger Problems
  - Effectively use all available computational resources

Dividing up the Population

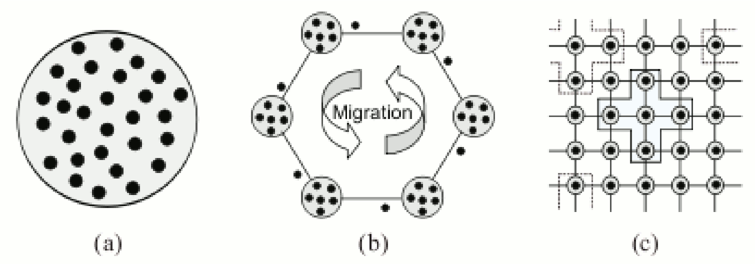


Figure: Three different types of GAs showing interaction between individuals (black dots) in the population. a)Panmictic b) Distributed c) Cellular [Alba:2008]

Distributed GA - results

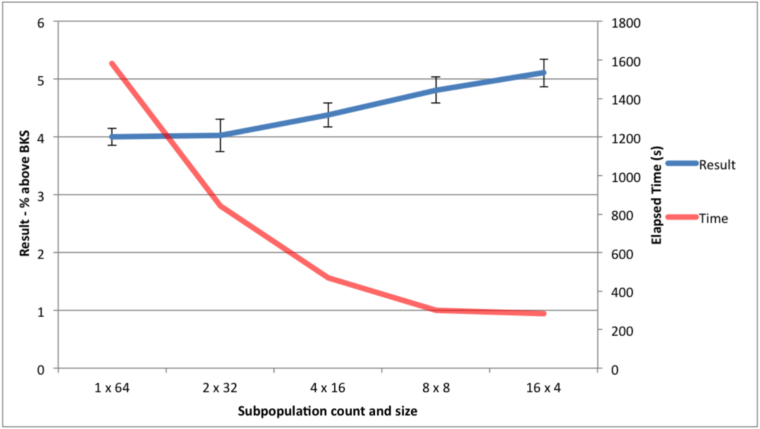


Figure: % above BKS and computational time for a variety of island sizes - no migration (run on 2.2GHz quad-core Intel Core i7)

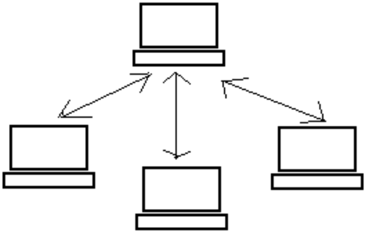
Serial Algorithm $- >$ Parallel Algorithm

- Allocate different subpopulations to different processors
- Communication between subpopulations - better results?

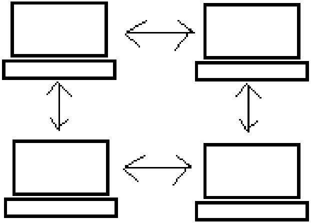
Serial Algorithm $->$ Parallel Algorithm

- Allocate different subpopulations to different processors
- Communication between subpopulations - better results?
- Master-slave versus Direct communication (Who?)
- Message Passing versus Shared Memory (How?)

Parallel Structures (Who?)



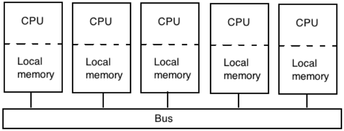
(a) Master-slave



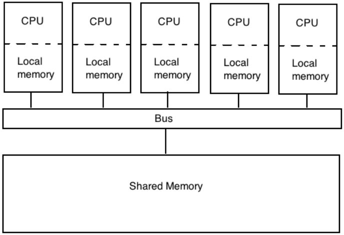
(b) Direct communication

Figure: Different approaches to parallel programming

Communication Schemes(How?)



(a) Message Passing



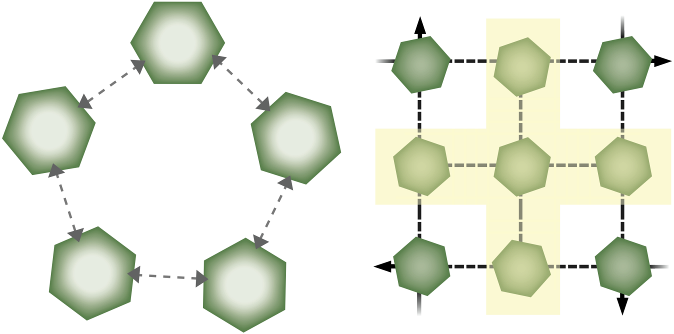
(b) Shared Memory

Figure: Different approaches to inter-processor communication

Local communication

- Arrange subpopulations on grid, define neighborhood on grid [Scharrenbroich: 'CGA-inspired' distributed GA]
- Carry strongest individuals between 'neighboring' subpopulations at certain points in algorithm

Mesh diagrams



(a) One dimensional

(b) Two dimensional

Figure: Different mesh arrangement with one possible neighborhood definition in (b)

Local communication

- Arrange subpopulations on grid, define neighborhood on grid [Scharrenbroich: 'CGA-inspired' distributed GA]
- Carry strongest individuals between 'neighboring' subpopulations at certain points in algorithm
- How?
  - Replace weakest individuals locally?
  - Place in 'waiting room' where they can be accessed, not directly replacing...
- When?
  - Regular intervals?
  - When population stagnates

Direct Communication results

- At best: negligible improvement
- Why? Problem, population size, number of processors...

Direct Communication results

- At best: negligible improvement
- Why? Problem, population size, number of processors...
- OR Straight up bad idea (in this case)

Global Communication

- All subpopulations connected through common 'vault'
- Strongest unique solutions found to date stored in vault

Vault Diagram

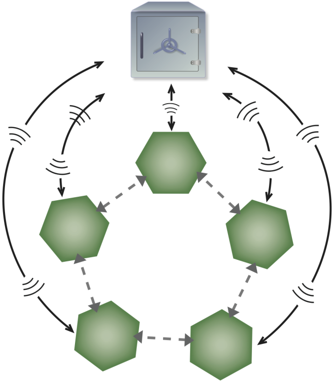


Figure: 1D Mesh with vault included

Vault implications

- Communication from subpopulations to vault - best, unique individuals

Vault implications

- Communication from subpopulations to vault - best, unique individuals
- Communication out of vault - occasionally breed with a randomly selected individual out of vault (modify selection)

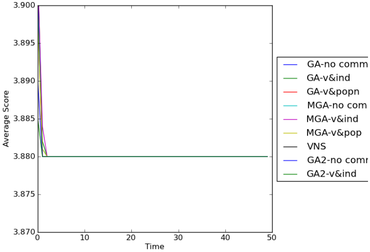
Vault implications

- Communication from subpopulations to vault - best, unique individuals
- Communication out of vault - occasionally breed with a randomly selected individual out of vault (modify selection)
- Evolution can lose local optima - vault will maintain global optima
  - Simulated Annealing type selection
  - Respawning (Individual?Subpopulation?)

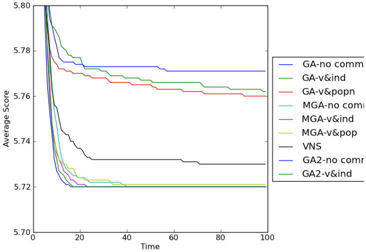
Parallel Testing

- Conducted on Genome cluster at UMD
- All experiments involving 32 threads with 32 processors and 32 subpopulations (or separate VNS trials)
- Tested on own generated databases (same as above) with $N = 100$, $p = 0.2$, $L = 25, 50, 100, 250, 500$
- Run 10 times with different random number seeds
- Max running time = L*20ms per instance (for each processor)

All Results vs Time

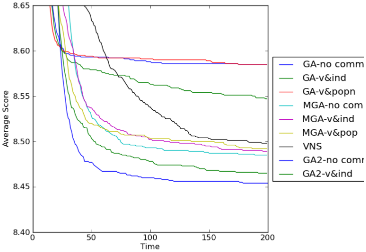


(a) L=25

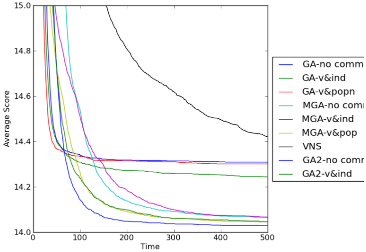


(b) L=50

All Results vs Time



(c) L=100



(d) L=250

All Results vs Time

Genetic Algorithm for the MLST

Oliver Rourke

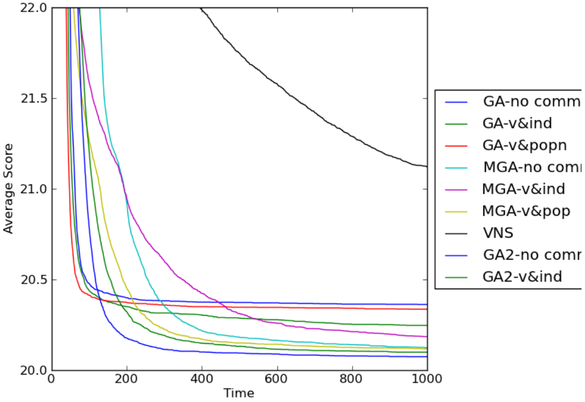
The MLST

Genetic Algorithms

My Serial GAs

Parallel GAs

Software/ Validation etc...



(e) L=500

Vault extensions

- Vault evolution (separate exploration of search space with investigation of interesting areas)

Vault extensions

- Vault evolution (separate exploration of search space with investigation of interesting areas)
- Replace unique with different enough (Hamming or Levenshtein distance...)

Software/Hardware

- C++ with pthreads
- Run on Genome cluster at UMD - Quad-Core AMD Opteron® Processor 8382 (2.6GHz)

Serial Validation

- Genetic Algorithm: Compared with original code from Xiong et al. [2005]. When random seed set correctly and random numbers sampled in the same order, returned the same results.
- VNS: Results compared with the results reported in Consoli (2009). Similar results achieved (no statistically significant difference)

Parallel Validation

- Remove all inter-processor communication, record results from each processor individually. Check similar to serial code.
- Verify the sending and receiving for each type of message on both ends
- Investigate speed-up (on 32 processor machine, 32 subpopulations for parallel code):

Instance Set	L=100	L=500
Time for serial (s)	2.38	5.42
Parallel-No Comm (s)	3.75	8.25
Parallel-Synchronized Iterations (s)	4.35	10.89
Parallel-Synchronized+Vault (s)	4.51	12.03

Goals

- Create own competitive, efficient, serial GA code

Goals

- Create own competitive, efficient, serial GA code
- Convert to an efficient parallel GA, first synchronous and later asynchronous.

Goals

- Create own competitive, efficient, serial GA code
- Convert to an efficient parallel GA, first synchronous and later asynchronous.
- Fine tune parallel GA (investigate migration operators)

Goals

- Create own competitive, efficient, serial GA code
- Convert to an efficient parallel GA, first synchronous and later asynchronous.
- Fine tune parallel GA (investigate migration operators)
- Run optimized code on large array of processors

Deliverables

- Efficient, competitive serial GA code for the MLST
- Efficient, asynchronous and synchronous parallel GA code for the MLST
- Results from running code on appropriate machines
- Report, presentation

Thanks

A sincere thank you to Dr Golden for so much invaluable advice, patient listening and watching over the whole project

Thanks also go to Drs Ide and Balan for help with presenting work and other advice, to Dr Zimin for help with the Genome cluster and many pointers to do with parallel computing, and to all of you for listening and for your questions

Bibliography

- Alba, E. and Dorronsoro, B., Cellular Genetic Algorithms, Springer, NY, 2008
- Back, T., Hammel., U and Schwefel, H., Evolutionary Computation: Comments on the History and Current State, IEEE Transactions on evolutionary computation, Vo. 1, No 1, 1997
- Canyurt, O. And Hajela, P., Cellular Genetic algorithm technique for the multicriterion design optimization, Struct. Multidisc Optim 20, 2010
- Chang, R. and Leu, S., The minimum labeling spanning trees, Information Processing Letters 63, 1997
- Cerulli, R. et al., Metaheuristics comparison for the minimum labelling spanning tree problem in The Next Wave in Computing, Optimization, and Decision Technologies, vol. 29 of Operations Research/Computer Science Interfaces Series, NY, NY, USA, 2005
- Consoli, S. et al., Greedy Randomized Adaptive Search and Variable Neighbourhood Search for the minimum labelling spanning tree problem, European Journal of Operational Research, vol. 196 pp. 440-449, 2009
- Drummond, L., Ochi, L. and Vianna, D., An Asynchronous parallel metaheuristic for the period vehicle routing problem, Future Generation Computer Systems 17, 2001
- Groer, C., Golden, B. and Wasil, E., A Parallel Algorithm for the Vehicle Routing Problem, INFORMS Journal on Computing, 2010
- Fujimoto, N. and Tsutsui, S., A Highly-Parallel TSP Solver for a GPU Computing Platform, LNCS 6046, 2011
- Huy, N. et al., Adaptive Cellular Memttic Algorithms, Evolutionary Computation 17(2), 2009
- Karova, M., Smarkov, V. and Penev, S, Genetic operators crossover and mutation in solving the TSP problem, International conference on computer systems and technologies, 2005. Katayama, Hirabayashi, Naruhusa, Performance Analysis for Crossover Operators of Genetic Algorithm, Systems and Computers in Japan, Vol 30., No 2., 1999
- Papaioannou, G. and Wilson, J., The evolution of cell formation problem methodologies based on recent studies (1997-2008): Review and directions for future research, European Journal of Operational Research 206, 2010

Bibliography (cont.)

- Paszkowicz, W., Properties of a Genetic algorithm extended by a random self-learning operator and asymmetric mutations: A convergence study for a task of powder-pattern indexing, Analytics Chimica Acta 566 (2006)
- Sarma J., and De Jong, K., An analysis of the effect of the neighborhood size and shape on local selection algorithms. In H.M. Voigt, W. Ebeling, I. Rechenberg, and H.P. Schwefel, editors, Proc. of the International Confer- ence on Parallel Problem Solving from Nature IV (PPSN-IV), volume 1141 of Lecture Notes in Computer Science (LNCS), pages 236244. Springer-Verlag, Heidelberg, 1996.
- Simoncini D., et al., From Cells to Islands: An Unified Model of Cellular Parallel Genetic Algorithms, ACRI, 2006.
- Seredynski, F. and Zomaya, A., Sequential and Parallel Automata-Based Scheduling Algorithms, IEE Transactions on Parallel and Distributed Systems, Vol 13, No 10, 2002
- Serpell, M. and Smith, E., Self-Adaptation of Mutation Operator and Probability for Permutation Representation in Genetic Algorithms, Evolutionary Computation 18(3), 2010
- Vidal, P. and Alba, E., A Multi-GPU Implementation of a Cellular Genetic Algorithm, IEEE 2010
- Xiong, Y., Golden, B. and Wasil, E., A One-Parameter Genetic Algorithm for the Minimum labeling Spanning Tree problem, IEEE Transactions on evolutionary computation, Vol. 9, No. 1, 2005
- Xiong, Y., Golden, B., and Wasil, E., Improved Heuristic for the Minimum Label Spanning Tree Problem, IEEE Transactions on evolutionary computing, Vol 10., No. 6, 2006