

- ▶ Abstract
- ▶ Full Text (PDF)
- ▶ References
- ▶ XML

Research Article

Evaluation of Spatial Parallel Genetic Algorithms for Real Time Routing in Geographic Information System

[R. Shad](#), [A. Shad](#), [D. Molaei](#) and [M.S. Mesgari](#)

ABSTRACT

In this study, two developed shortest path algorithms that run fast on the real large volume road networks have been identified. The first one is heuristic genetic algorithm implemented with approximate buckets in scalar computing environment and the second one is parallel genetic processing which is run in the alternative space. At first, these two algorithms were reviewed and summarized and their data structures and procedures are presented. Continually, in this effort genetic algorithm is used to solve the shortest path problem, because the limitation of traditional optimization methods. Finally, present result demonstrates that parallel heuristic processing can produce better speed up performance for real-time transportation applications.

Services

- [E-mail This Article](#)
- [Related Articles in ASCI](#)
- [Similar Articles in this Jou](#)
- [Search in Google Scholar](#)
- [View Citation](#)
- [Report Citation](#)

How to cite this article:

R. Shad, A. Shad, D. Molaei and M.S. Mesgari, 2009. Evaluation of Spatial Parallel Genetic Algorithms for Real Time Routing Geographic Information System. *Journal of Applied Sciences*, 9: 4179-4182.

DOI: [10.3923/jas.2009.4179.4182](https://doi.org/10.3923/jas.2009.4179.4182)

URL: <http://scialert.net/abstract/?doi=jas.2009.4179.4182>

INTRODUCTION

Real-time transportation applications commonly require real-time processing while demanding best solutions in GIS environme example, today for the police applications or emergency services it is possible to find the fastest route and dispatch agents usir Recently, optimum path problems are discussed in computational geometry, graph algorithms, geographical information and ro Over many years researchers faced the problem of routing to (Maheshwari *et al.*, 2000; Mitchell, 2000) achieve optimum solut fast performance. Mitchell and Papadimitriou (1991) provide an approximation algorithm to compute a weighted short path. La *al.* (1997) described the cost of the approximation is no more than the shortest path cost plus a factor of W_j . Max-Planck Instit Computer Science is explained if ϵ is 1/100 then the number of Steinerpoints is reduced by a factor of 1/10. For two polyhedra with n nodes, Baltzan and Sharir (Baltzan and Sharir, 1988) presented an $O(n^3 \log n)$ time shortest path algorithm. Hershberge Suri (1995) introduced a linear time algorithm.

Subsequently, multiple goals, factors and constraints in the large volume networks caused different heuristic and probabilistic that no always guarantee the optimal solutions are presented. A number of these algorithms in the literatures are reported (Zf Noon, 1998; Goldberg and Radzik, 1993). Dial (1969) was the first one who implements the Dijkstra algorithm using buckets a heuristic method. Dial's original implementation (DKB) requires $nC+1$ buckets in the worst case, where C is the maximum arc l network (Ahuja *et al.*, 1993). Continually, metaheuristic Genetic algorithm (Diaz *et al.*, 1996) is developed as a robust, flexible adaptive tool with optimal designing and programming of networks. Genetic algorithm can be successfully performed in the nor problems and also it is appropriate to face the noisy combinatorial solutions associated to the real networks.

The mentioned efforts have resulted in two categories. The first category contains algorithms which give optimal solutions and good performance (such as Dijkstra). The other ones are very fast and cannot guarantee best solutions (such as heuristic algor

These algorithms can produce optimum real time solutions using serial analysis that execute in scalar environment. Then f or implementing algorithms which be able to guarantee optimal solutions with fast performance, alternative environments need to be explored. For this purpose, there are already algorithms for computing optimum paths in network in parallel. These algorithms need a large number of processors (Ramarao and Venkatesan, 1992; Adamson and Tick, 1991; Bertsekas *et al.*, 1996). The ma problem of these kinds of algorithms is that they are run using parallel processors that are categorized under supercomputers. supercomputers are too expensive and are not accessible for all clients. Also, users require special skills for working with them alternative approach, which caused supercomputers are fewer complexes to use is the utilization of clusters of workstations or clusters, workstations or PCs are the most cost-effective parallel computing environments. Consequently, In this study, we eva illustrate how our algorithm in scalar and alternative environment can overcome the mentioned problems.

GENETIC ALGORITHM FOR SHORTEST PATH

Genetic algorithm is a family of computational methods inspired by evolution. Different solutions with genetic algorithms have been studied (Gen and Cheng, 1997). The shortest path problem can be performed on a given network, finding the optimum path (le from one or some source node to all other nodes or to a subset of nodes). The possible optimum paths in each generation corre: the population. Fitness function compares each path with the others and determine if a path has possibility of survives to next generation or not. This function is achieved based on the sum of products of the route lengths and their related costs. It works pseudo-point system where points are awarded for the solution ending near the target end point, starting near the start point : low-cost. The score in the first areas can be negative if the solution is exceptionally bad. We do not consider the cost of the sol the solution gets near both the start and end points to avoid premature convergence. Crossing over solutions that come within of intersecting dramatically increases the chances of finding a viable pair. The probability is multiplied by the average number from each node. In a representative run of the programs, the probability increased from 1 to 5.5%. The selection operator ext allocate elements of the sorted population. The previous generation is randomly chosen the cross points are randomly selected path and then the marked fragments of the paths are extracted and interchanged in both paths. Mutation operator makes the : space fully accessible. Because the crossover can only combine existing solutions, it cannot cause new links on the graph to be It is used in a similar way of crossing one. A path and its mutation points are randomly chosen.

Control parameters of the model are defined as generations number, generation paths number, path length, finishing criteria, mutation and crossing rates. The individual amount to be generated should be given according to the problem settings and the of nodes contained.

This algorithm allows an initial path population by random generation. Each path has a fitness function which is differentiated i others and then through genetic operators participate in the next generation for producing optimum paths (Fig. 1).

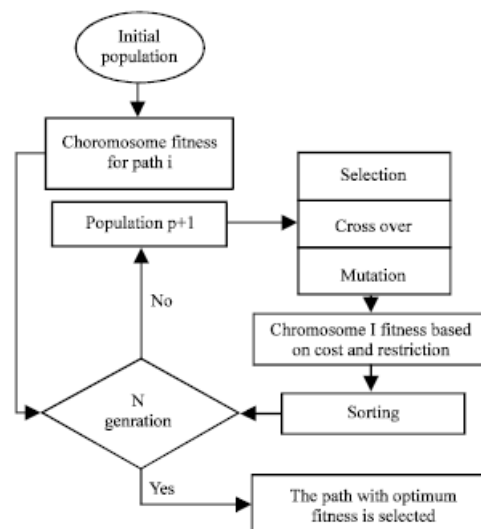


Fig. 1: Flow diagram of genetic algorithm for shortest path

THE GENETICS ALGORITHM IMPLEMENTED WITH APPROXIMATE BUCKETS

In genetic algorithm, different paths in each generation are treated as a not ordered list. This is equivalent to treating the prio Q in the above general procedure for shortest path tree construction as a not ordered list. This is of course a bottleneck operat because all paths in Q have to be visited at each generation in order to select the better path using fitness function for next ge

The bucket data structure is one of data structures which can arrange paths in a sorted fashion. Bucket stores all temporarily paths whose fitness functions fall within a certain range. Paths contained in each bucket can be represented with a doubly linked list. linked list only requires $O(1)$ time to complete an operation in each update in the bucket data structure (Dial, 1969). It can be the memory requirement in this method can be large when both number of nodes and generations are increased. However, we reduce the memory requirement using either the overflow bag implementation or the approximate buckets implementation. The algorithm implemented using approximate buckets contains temporarily paths with labels within the range of $[i*b, (i+1)*b-1]$ is a chosen constant. Therefore, this method can be run for increasing speed on the large volume data in real time applications

PARALLEL PROCESSING

In the high time-complexities and the large size problems parallel algorithms are so attended for shortest path computing. A review of the literatures reveals that parallel processing is used in two categories. First is contained decomposition of shortest path algorithm on multiple data stream computers (Ding *et al.*, 1992). The second one is contained parallel routing algorithms for aerial path planning on the terrain (Vezina *et al.*, 1994). The parallel algorithm used in this work is designed for a distributed memory architecture and decomposition of genetic algorithm for maximizing the utilization of the processors. Then our transportation network should be decomposed on some subnetworks which are equal to processors (PE). In asynchronous operation each processor processes the subnetwork independently and connects its subnetwork to an adjacent subnetwork through common boundary nodes. Each processor creates an extra network which its arcs and nodes consist of the resulted shortest path computing between all boundary nodes. After this computing the extra networks are aggregated together and create a total network which can be processed using one processor. This algorithm can be described for the special case as one-to-all, all-to-one and any-to-any.

EVALUATION OF PERFORMANCE

A cluster of workstations was used for evaluating the parallel heuristic genetic algorithm performance. Transportation network with different number of nodes and links was categorized to subnetworks. For driving subnetworks we use known domain partition structures such as quadtree and octree. One of the main advantages of the quadtree scheme is that it helps to have homogenous subnetworks.

Heuristic genetic algorithm was run on the PEs using MATLAB and C# object oriented programming language. The population-size equation was written based on the gambler ruin model (Ahn and Ramakrishna, 2002). Chromosomes (strings) with variable length were used for encoding the problem. The crossover operation exchanges spatial paths and the mutation maintains diversity of population. The total average of selection, crossing and mutation rates for our model were found as 39, 47 and 14% of the population paths each subnetwork, these rates are varied based on the size and volume. The solutions are heuristically found using approximate bucket and two link lists. Testing results approximately are shown, the best solution fitness and average fitness is converged after 130 generations.

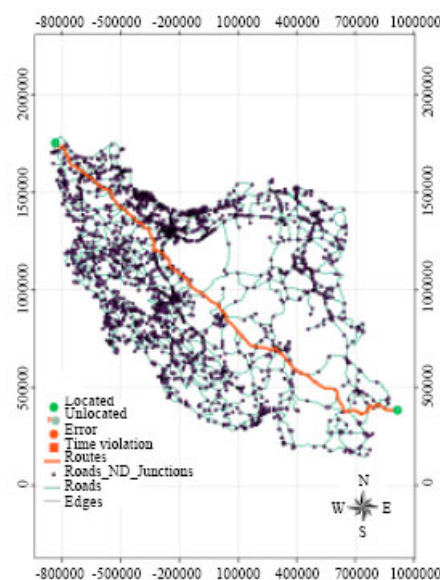


Fig. 2: The transportation networks which is used as our study area

Table 1: Execution time of heuristic genetic algorithm in serial and alternative environment

No. of nodes	300	600	1200	2000	2500
Serial (1 PE)	7.03 (sec)	12.25	23.6	40.6	59.3
Parallel (2 PEs)	5.46	9.1	18.5	34.7	52.8
Parallel (3 PEs)	4.03	8.2	15.3	25.2	40.1
Parallel (4 PEs)	4.03	8.3	14.5	23.0	28.6

Table 2: Speed up of the parallel algorithm

No. of nodes	1200	2000	2500
Serial (1 PE)	1.00 (Tp/Ts)	1.00	1.00
Parallel (2 PEs)	0.78	1.17	1.12
Parallel (3 PEs)	1.74	1.61	1.47
Parallel (4 PEs)	1.74	1.76	2.07

In [Table 1](#) the timing of the heuristic genetic algorithm for different number of PEs in serial and alternative environment is shown.

It is necessary to consider the real-time performance of a parallel algorithm depends on different parameters determined by the algorithm, data, machine and implementation. As we can see from [Table 1](#), as the number of processors increases, the execution time decreases and for small number of points parallel processing is not so useful.

In this study, we measure the speedup as T_p/T_s , where T_p is the time it takes to solve the problem using the parallel algorithm and T_s is the execution time of serial algorithm (Bertsekas *et al.*, 1996). As we can see in [Table 2](#), the speed up of parallel algorithm is in the range of 0.78 to 2.07 for the large size of transportation network.

CONCLUSIONS

In this study, a heuristic genetic algorithm has been used to optimize and evaluate real time routing in serial and alternative environment. In this evaluation each processor is overloaded by partitioned data for reducing its performance time in alternative environment. The results show parallel computing environments are better for producing best routes in large size and complex transportation networks. Comparing with traditional methods, the proposed one, is cost-effective and faster to handle decreasing the total processing time and allowing almost any real conditions.

REFERENCES

- Adamson, P. and E. Tick, 1991. Greedy partitioned algorithms for the shortest-path problem. *Int. J. Parallel Program*, 20: 271-287. [CrossRef](#) | [Direct Link](#) |
- Ahn, C.W. and R.S. Ramakrishna, 2002. A genetic algorithm for shortest path routing problem and the sizing of populations. *IEE Proc. Comput. Vis. Commun. Technol.*, 6: 566-579. [CrossRef](#) | [Direct Link](#) |
- Ahuja, R.K., T.L. Magnanti and J.B. Orlin, 1993. *Network Flows: Theory, Algorithms and Applications*. Prentice Hall, Englewood Cliffs, NJ.
- Baltsan, A. and M. Sharir, 1988. On the shortest paths between two convex polyhedra. *J. ACM.*, 35: 267-287. [CrossRef](#) | [Direct Link](#) |
- Bertsekas, D.P., F. Guerriero and R. Musmanno, 1996. Parallel asynchronous label-correcting methods for shortest paths. *J. Optim. Theory Appl.*, 88: 297-320.
- Dial, R.B., 1969. Algorithm 360: Shortest path forest with topological ordering. *Commun. ACM*, 12: 632-633. [CrossRef](#) | [Direct Link](#) |
- Diaz, A., F. Glover, H.M. Ghaziri, J.L. González, M. Laguna, P.Y. Moscato and F.T. Tseng, 1996. *Heuristic Optimization and Networks in Operations Management and Engineering*. 1st Edn., Editorial Paraninfo S.A., Madrid.
- Ding, Y., P.J. Densham and M.P. Armstrong, 1992. Parallel processing for network analysis: Decomposing shortest path algorithm for MIMD computers. *Proceedings of the 5th International Symposium on Data Handling*, Aug. 3-7, Charleston, SC., pp: 682-691.
- Gen, M. and R. Cheng, 1997. *Genetic Algorithms and Engineering Design*. 1st Edn., John Wiley and Sons, New York, ISBN: 0-471-12741-8.
- Goldberg, A.V. and T. Radzik, 1993. A Heuristic Improvement of the Bellman-Ford Algorithm. *Applied math. Lett.*, 6: 3-6. [CrossRef](#) | [Direct Link](#) |
- Hershberger, J. and S. Suri, 1995. Practical methods for approximating shortest paths on a convex polytope in R. *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, Feb. 5, San Francisco, California, pp: 447-456.
- Lanthier, M., A. Maheshwari and J.R. Sack, 1997. Approximating weighted shortest paths on polyhedral surfaces. *Proceedings of the Annual Symposium on Computational Geometry*, Jun. 4-6, Nice, France, pp: 274-283.
- Maheshwari, A., J.R. Sack and H. Djidjev, 2000. Link Distance Problems. In: *Handbook on Computational Geometry*, Sack, J.R. (Ed.), pp: 1-12.

[Translate](#)

Urrutia (Eds.). Elsevier Science, USA., pp: 519-558.

Mitchell, J.S.B. and C.H. Papadimitriou, 1991. The weighted region problem: Finding shortest paths through a weigh planar sut. J. ACM., 38: 18-73.

[CrossRef](#) |

Mitchell, J.S.B., 2000. Geometric Shortest Paths and Network Optimization. In: Handbook on Computational Geometry, Sack, J Urrutia (Eds.). Elsevier Science, USA., pp: 633-702.

Ramarao, K.V.S. and S. Venkatesan, 1992. On finding and updating shortest paths distributively. J. Algorithms, 13: 235-257.

[CrossRef](#) |

Vezina, G., G. Ratzler and V. Vandongen, 1994. Parallel spatial analysis and interactive visualization software. Proceedings of International Canadian Conference on GIS, June 6-10, Ottawa, pp: 1048-1055.

Zhan, F.B. and C.E. Noon, 1998. Shortest path algorithms: An evaluation using real road networks. Transportation Sci, 32: 65

[CrossRef](#) |