Genetic Algorithm Based Route Planner for Large Urban Street Networks

Suranga Chandima Nanayakkara*, Dipti Srinivasan † Lai Wei Lup † Xavier German † Elizabeth Taylor * and S.H. Ong †

*Marine Mammal Research Laboratory, Tropical Marine Science Institute, National University of Singapore, 14 Kent Ridge Road, Singapore 119223
scn@nus.edu.sg, mdcohe@leonis.nus.edu.sg
†Department of Electrical and Computer Engineering, National University of Singapore, 9 Engineering Drive 1, Singapore 117576
{elesd, u0307643, g0501075, eleongsh}@nus.edu.sg

Abstract—Finding the shortest path from a given source to a given destination is a well known and widely applicable problem. Most of the work done in the area have used static route planning algorithms such as A*, Dijkstra’s, Bellman-Ford algorithm etc. Although these algorithms are said to be optimum, they are not capable of dealing with certain real life scenarios. For example, most of these single objective optimizations fail to find the equally good solutions when there is more than one optimum (shortest distance path, least congested path). We believe that the Genetic Algorithm (GA) based route planning algorithm proposed in this paper has the ability to tackle the above problems. In this paper, the proposed GA based route planning algorithm is successfully tested on the entire Singapore map with more than 10,000 nodes. Performance of the proposed GA is compared with an ant based path planning algorithm. Simulation results demonstrate the effectiveness of the proposed algorithm over ant based algorithm. Moreover, the proposed GA may be used as a basis for developing an intelligent route planning system.

I. INTRODUCTION

There have been many attempts to use genetic algorithm (GA) based techniques for route planning problems [1], [2], [3], [4], [5]. For example, Chang and Ramakrishna 2002 [1] have used a GA with variable-length chromosomes and have obtained very good results. Chakraborty 2004 [2] developed a GA that is able to find dissimilar multiple sub-optimal paths. A similar problem was tackled by Yinzhen et al., 2005 [3] taking the fuzziness of the meaning of ‘dissimilar paths’ into account. However, almost all these algorithms were tested on small and medium size networks, typically with 10-150 nodes.

Recently, Zhu et al. [6] have used Particle Swarm Optimization (PSO) technique to solve Vehicle Routing Problem with Time Windows (VRPTW), making use of the various constraints as well as ‘punishment coefficients’ to evaluate the fitness function. Jun [7], has proposed an Ant Colony Optimization (ACO) based approach for a single vehicle route planning system. A hybrid between Dijkstra’s algorithm with Ant algorithm is used to improve the results over a pure ACO. Both pure and hybrid algorithms have been tested on the Singapore Map. In this paper, we have compared the results of this hybrid algorithm and the results of our proposed GA based algorithm.

On the other hand, Bing and Tay, 1995 [8] combined conventional shortest path algorithms (Dijkstra’s and A*) with explicit knowledge about the road network, to make the path planning algorithm much faster. In addition, they successfully tested their algorithm on the entire Singapore map (with over 10,000 nodes). However, in most real life scenarios, the shortest path may not be the best path for the user. In fact, different users may prefer different routes depending on the traffic congestion, road quality and other time varying quantities. Therefore, there is a need to develop a multi-objective algorithm that can handle many different objectives simultaneously.

The objective of this paper is to develop a GA based route-planning algorithm that can work on a huge network with thousands of nodes. The proposed GA uses only distance information. However, it is easily extendable to a multi-objective optimization algorithm and thus can be used as the basis for an intelligent route planning system. The map of Singapore which has more than ten thousand nodes is used as the test case. Each of the links (connecting two nodes) is assigned with a cost that corresponds to the distance between the nodes. The proposed algorithm is able to find the shortest path from one location to another. The results are compared with a pure ant-based route planning algorithm proposed by Xavier [9] and a modified version of ant-based algorithm proposed by Jun [7].

The rest of this paper is organized as follows. Proposed GA is described in section 2 and implementation details are given in the subsequent section. Section 4 contains the results and comparison of the different approaches. The last section of the paper consists of our conclusions and suggestions for future work.
II. DESCRIPTION OF THE ALGORITHM

The main challenge of this work is that the GA has to work with a huge network with over 10,000 nodes. To illustrate this, consider the size of the cost matrix $C = [c_{ij}]$, where $c_{ij}$ is the cost of link $(i,j)$. Finding a feasible path from the source to destination becomes even more difficult since each node (location) is connected to a maximum of 6 other nodes and some links (roads) are directional in the sense that one can travel only in one direction. Therefore, GA operators, especially the initialization, were designed carefully (see Figure 1 for the pseudo-code) to tackle the given network.

Pseudo-code of the overall GA is given in Figure 2 and each of the GA components are explained in the following subsections.

A. Representation

A chromosome consists of sequence of positive integers that represents the IDs of nodes through which a path passes. Each locus of the chromosome represents an order of a node in a routing path. A chromosome has a variable length between 2 to $N$ depending on the path, where $N$ is the total number of nodes in the network. This is because it never needs more than $N$ number of nodes to form a routing path. Figure 3 shows an example of chromosome encoding form node $S$ to $D$.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{chromosome_example.png}
\caption{Example of a Chromosome}
\end{figure}

B. Initialization

Since any given location is connected to a maximum of six other locations, finding a possible path from source to destination can be very difficult. For example, a typical path (say from Hougang Avenue to Bukit Panjang Ring Road) can go through more than 50 locations (nodes). One possible initialization method is to start from the source, randomly pick a node connected to source and continue this procedure until the destination is reached. This method takes a very long time to find a path and it sometimes never reaches the destination. This is because this scheme allows the loops within a path. Moreover, if we try to modify the above scheme by not allowing loops, in most cases a path gets trapped in a node where there are no more possible links (i.e. all the nodes connected have already been visited). Therefore, the following rules were used in order to make the initialization faster and avoid getting stuck at a node.

- Start from the source and build a path (path one) towards the destination and at the same time build another path (path two) starting from the destination towards the source.
- Stop building the path if path one reaches the destination or path two reaches the source or if the path size reaches a maximum length.
- If either path one or path two is successful, remove the loops (using the repair function described in the following section) from the path and add the successful path to the population. If unsuccessful, check path one and path two to see if there is a common node. If so, form a feasible path from the source to destination, combining two paths at the common node. Remove the loops and add the successful path to the population. If there is no common node in path one and two, this is taken as an unsuccessful attempt and this starts a new search.

Pseudo-code of the initialization algorithm is shown in Figure 1.

C. Repair Function

The objective of the repair function is to repair the unfeasible chromosomes (invalid paths) so that they become feasible. Traditionally, there are many strategies that deal with infeasible chromosomes [10]. Two most common methods are to repair the infeasible chromosome and the other is to impose a penalty [11]. However, designing an appropriate penalty function is not an easy task [1]. Moreover, penalty function may sacrifice some feasible chromosomes as well.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{example.png}
\caption{Example of a Chromosome}
\end{figure}
because the unfeasible chromosomes might continue to be reproduced in the proposed GA. Therefore, a simple repair function is implemented to remove the lethal genes that form a loop in a particular path. This simple method can correct all the unfeasible chromosomes without adding too much computational cost. Figure 4 shows an example of the proposed repair function.

D. Population size

Population size has a huge impact on the execution time of the proposed GA. This is because the initialization can be very slow. Therefore, having a smaller population size will lead to a shorter execution time. On the contrary, it is desirable to use larger population sizes so that the population diversity can be maintained during the GA. These two objectives are conflicting and after many experimental trials, it was observed that the suitable population size depends on the complexity of the route. For shorter routes, a population size of 2 was adequate to give good results and for longer routes a population size of 20 seemed to be a reasonable choice.

E. Fitness Evaluation

The fitness function of the proposed GA is rather straightforward. This is because the current GA is a single objective optimization and the objective is to minimize the total cost (distance). The total cost for a particular chromosome can be calculated by adding up the link costs between the nodes encoded in that chromosome. The following equation shows the fitness of the $i^{th}$ chromosome.

$$f_i = \frac{\alpha}{\sum_{j=1}^{l_i-1} C_{g_i(j),g_i(j+1)} }$$  
(1)

Where $f_i$ represents the fitness of the $i^{th}$ chromosome, $l_i$ is the length of the $i^{th}$ chromosome, $g_i(j)$ is the gene of the $j^{th}$ locus in the $i^{th}$ chromosome, $C$ is the link cost between two nodes. $\alpha$ is a scaling parameter. This approach ensures that the function has a higher value when the fitness characteristic of the chromosome is better than that of others.

F. Variation Operators

The variation operators are implemented in such a way that they always yield valid chromosomes except for possible loops. The repair function is used to get rid of the loops.

1) Mutation: Mutation was implemented by picking up a sub-path from the chromosome and replacing it with a new path, with a probability $P_m$. Two loci ($p_1$ and $p_2$ such that $p_1 < p_2$) are picked randomly form the chromosome and the path from the gene at locus $p_1$ and locus $p_2$ is taken as the sub-path to be replaced. This is illustrated in Figure 5. The initialization algorithm is used to create a new path from the source of the sub-path to the destination of the sub-path. If the initialization algorithm manages to find a valid path easily (this means that the initialization algorithm finds a path within few iterations), the sub-path is replaced by the newly created path, otherwise the chromosome does not get mutated. The repair function is applied to the mutated chromosome to eliminate possible loops in the path.

2) Crossover: The crossover implementation used in this project is different from the conventional one point crossover but is similar to the approach used in [1]. Crossover we proposed works as follows: Firstly, two parents are selected and they should have at least one common gene (node) except for source and destination nodes in order to apply the crossover operation. One possible crossover point (common node) is randomly picked whenever there are many possible crossover points. Once the crossover point is decided, the usual method of one point crossover operation is applied and finally the repair function is applied to the created offspring to remove possible loops. The crossover operation is illustrated in Figure 6.

G. Parent Selection

Three different selection schemes are implemented so that the user can choose a selection scheme and compare the performance. Roulette wheel selection, rank-based selection and tournament selection (with a tournament size of 2) are available in the current version of the software. From the
Fig. 6. Example of the Crossover Operation

experimental results, it was concluded that a tournament
selection scheme (with tournament size of 2) is the best
parent selection scheme for the proposed GA.

H. Survivor Selection

This method picks up some of the best individuals (user
can define the number) in the previous generation and
replaces the worst in the current population. It is ensured
that the preserved individuals are dissimilar. This is desirable
since we can ensure good population diversity by avoiding
a situation where one individual dominates the entire popu-
lation. Best results were obtained with 2 elites.

I. Termination

The GA terminates after a fixed number of generations
specified by the user or the user has the option to terminate
it whenever he is satisfied with the best solution evolved so
far.

III. IMPLEMENTATION

The proposed GA based route planner is implemented
in LabVIEW programming environment as an interactive
programme. LabVIEW is a multi-threaded programming
platform hence it allows running multiple loops simultane-
ously. This makes the run time of the GA in LabVIEW much
faster. In addition, LabVIEW is a graphical programming
language and this makes it very easy to understand the
way in which GA concepts have been implemented. The
LabVIEW programme is compiled into an executable file
with an installer so that it is able to run without the LabVIEW
environment. This allows it to be installed and run from any
computer. The current version of the programme allows the
user to change the GA parameters (population size, selection
scheme, crossover rate, mutation type and rate, number of
elitists) interactively. In addition to the best routing path
evolved by the programme and the cost of that path, the
output of the programme includes: A real-time graph that
shows the variation of the best, worst and average fitness
values for every generation. An indicator that shows the best
fitness value and the chromosome that has that fitness for
each generation, a real-time graph that shows the variation
of the standard deviation of the population fitness for every
generation and a graph to show how the mutation rate
changes with the generation number.

IV. RESULTS

The Proposed GA is tested for routes within the Singapore
city area and for longer routes across the country. Each test
case is simulated for 30 independent runs and the average
value obtained is recorded. Table I shows the results obtained
for long routes and Figure 7 shows the shortest path from
Pasir Ris Road to Clementi Road (Test case 1 in Table I).

<table>
<thead>
<tr>
<th>Case</th>
<th>Location 1</th>
<th>Location 2</th>
<th>Path * (km)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pasir Ris Road</td>
<td>Clementi Road</td>
<td>27.57 (0.78)</td>
</tr>
<tr>
<td>2</td>
<td>Woodlands Road</td>
<td>Orchard Boulevard</td>
<td>20.43 (0.71)</td>
</tr>
<tr>
<td>3</td>
<td>Tampines Avenue 3</td>
<td>Upper Bukit Timah Road</td>
<td>22.38 (0.43)</td>
</tr>
<tr>
<td>4</td>
<td>Hougang Avenue 4</td>
<td>Blk Panjang Ring Road</td>
<td>19.70 (0.56)</td>
</tr>
<tr>
<td>5</td>
<td>Woodlands Centre Road</td>
<td>East Coast Parkway</td>
<td>30.37 (0.42)</td>
</tr>
<tr>
<td>6</td>
<td>Jurong Port Road</td>
<td>Simei Street 2</td>
<td>33.16 (0.40)</td>
</tr>
<tr>
<td>7</td>
<td>Ubi Avenue 3</td>
<td>Sunset Way</td>
<td>17.78 (0.36)</td>
</tr>
<tr>
<td>8</td>
<td>Punggol Road</td>
<td>Sembawang Road</td>
<td>20.20 (0.20)</td>
</tr>
</tbody>
</table>

Table II shows the different test cases for short routes and
the distance of the best path obtained using the proposed
GA. Since most of the roads in the city area are one way
roads, cases 2a, 2b, 3a, 3b, 5a and 5b were tried out the
see the effect of this (Table II). Figure 8 shows the shortest
path generated from New Bridge Road (A) to Orchard Rd.

Table II

<table>
<thead>
<tr>
<th>Case</th>
<th>Location 1</th>
<th>Location 2</th>
<th>Path * (km)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pasir Ris Road</td>
<td>Clementi Road</td>
<td>27.57 (0.78)</td>
</tr>
<tr>
<td>2</td>
<td>Woodlands Road</td>
<td>Orchard Boulevard</td>
<td>20.43 (0.71)</td>
</tr>
<tr>
<td>3</td>
<td>Tampines Avenue 3</td>
<td>Upper Bukit Timah Road</td>
<td>22.38 (0.43)</td>
</tr>
<tr>
<td>4</td>
<td>Hougang Avenue 4</td>
<td>Blk Panjang Ring Road</td>
<td>19.70 (0.56)</td>
</tr>
<tr>
<td>5</td>
<td>Woodlands Centre Road</td>
<td>East Coast Parkway</td>
<td>30.37 (0.42)</td>
</tr>
<tr>
<td>6</td>
<td>Jurong Port Road</td>
<td>Simei Street 2</td>
<td>33.16 (0.40)</td>
</tr>
<tr>
<td>7</td>
<td>Ubi Avenue 3</td>
<td>Sunset Way</td>
<td>17.78 (0.36)</td>
</tr>
<tr>
<td>8</td>
<td>Punggol Road</td>
<td>Sembawang Road</td>
<td>20.20 (0.20)</td>
</tr>
</tbody>
</table>

Fig. 7. Shortest path for the test case 1 given in Table I. Blue color lines in
this figure represent the roads and the thick red line represents the shortest
path obtained from the proposed GA.

Fig. 8. Shortest path generated from New Bridge Road (A) to Orchard Rd.
(B) in thick red line and return path (from B to A) in thick green line. Two paths are clearly different. This is because most of the roads in the city area are one way. The proposed GA is able to handle these situations effectively.

The Singapore map has more than ten thousand nodes and hence initialization of one possible path (a chromosome) usually takes a long time. Therefore, the population size is limited to a maximum of 20. In addition, once a population is initialized for a particular route, the user has the option to reuse the created initial population. In other words, the user can choose to skip the initialization step. This option reduces the GA execution time and enables users to try out different GA parameter sets for a particular path within a short time. A fixed mutation probability of 1 gave the best performance. However, this does not mean that every gene gets mutated. This is because even if all the chromosomes are selected and a random sub-path in the chromosome is picked for mutation, the chosen sub-path is replaced only when the mutation operator finds an obvious alternative path.

### Table II

<table>
<thead>
<tr>
<th>Case</th>
<th>Source</th>
<th>Destination</th>
<th>Shortest Path (km)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Raffles Boulevard</td>
<td>Clarke Quay</td>
<td>3.03 (0.13)</td>
</tr>
<tr>
<td>2a</td>
<td>New Bridge Road</td>
<td>Orchard Rd</td>
<td>4.89 (0.12)</td>
</tr>
<tr>
<td>2b</td>
<td>Orchard Road</td>
<td>New Bridge Road</td>
<td>4.08 (0.14)</td>
</tr>
<tr>
<td>3a</td>
<td>Scotts Road</td>
<td>Newton Circus</td>
<td>1.13 (0.05)</td>
</tr>
<tr>
<td>3b</td>
<td>Newton Circus</td>
<td>Scotts Road</td>
<td>1.13 (0.06)</td>
</tr>
<tr>
<td>4</td>
<td>Victoria Street</td>
<td>Novena Terrace</td>
<td>3.20 (0.09)</td>
</tr>
<tr>
<td>5a</td>
<td>Somerset Road</td>
<td>Hong Kong Street</td>
<td>2.49 (0.08)</td>
</tr>
<tr>
<td>5b</td>
<td>Hong Kong Street</td>
<td>Somerset Road</td>
<td>2.24 (0.07)</td>
</tr>
<tr>
<td>6</td>
<td>Cashew Road</td>
<td>Kent Ridge Crescent</td>
<td>9.72 (1.18)</td>
</tr>
</tbody>
</table>

**Fig. 8.** Shortest path for the test cases 2a and 2b given in Table II. Blue color lines in this figure represent the roads in the Singapore city area. Thick red line represents the shortest path from New Bridge Road (A) to Orchard Road (B) and thick green line represents the shortest path for return journey. Two paths are different since the most of the roads in city area are one way.

### A. Proposed GA vs Ant Based Algorithm

We have compare the performance of the proposed GA with a pure ant-based algorithm proposed by Xavier [9] and with an ant-based algorithm with smart initialization proposed by [7]. Since longer routes are much more challenging, we tested the performance of the three different algorithms using test cases given in Table I. Each test case is simulated for 30 independent runs and average value was obtained. In addition, a t-test was conducted to statistically analyze the differences in the average distances. The t-test assesses whether the means of two groups are statistically different from each other. We calculated the t values using

\[
t = \frac{\bar{x}_T - \bar{x}_C}{\sqrt{\frac{\sigma_T^2}{n_T} + \frac{\sigma_C^2}{n_C}}},
\]

where \(\bar{x}_T\) is the value we tested (obtained from the proposed GA), \(\bar{x}_C\) is the value we tested (obtained from the ant algorithm), \(\sigma_T\) denote the variances and \(n_T, n_C\) are the number of samples. In Table III, value within the parentheses shows the values obtained by the t-test. For example, -2.57 in row one column three shows the t value between the values obtained for proposed GA and pure ant algorithms. In this case, -2.57 is higher than the value given in the standard t-table with 95% confidence and therefore, the difference between the two mean values are statistically significant. From the results summarized in Table III and Figure 9, it is clear that the proposed GA is able to find a better path than the pure ant algorithm in 6 cases out of 8. In fact, when the calculated t-values are compared with a standard t-table, we can conclude, with 95% confidence, that the differences in means are statistically significant. On the other hand, modified ant based algorithm (with smart initialization) given in [7], gives equally good results compared to the proposed GA. However, ant algorithm with smart initialization is somewhat biased since it runs the modified Dijkstra’s algorithm to find the absolute shortest route from the source node to the destination and uses this information to initialize the algorithm. This method is unconventional since it modifies the solution (the one obtained with Dijkstra’s algorithm) and then tries to get back the original using the ant based algorithm. In contrast, the GA initialization we propose is much more general.

### V. Conclusions

The computation time of the GA seems to depend a lot on the initialization. For the smaller network with 6 nodes,
it took only about 0.6 seconds to initialize a population size of 10. Conversely, for the larger network with over 10,000 nodes, it took about one minute just to initialize one candidate solution. This is expected when considering the size of the network. However, reducing the initialization time still remains a major challenge.

One obvious way to reduce the initialization time is to cut down the population size. This was actually tried and it was observed that, for complex routes, the accuracy of the GA decreases as the population size decreases. In contrast, a population size of 2 was adequate to get good enough results for shorter routes (routes within the city area). Therefore, we believe that it is desirable to modify the GA so that it can do a fast analysis to check the complexity of the path and decide a suitable population. Moreover, it is possible to use niching methods such as fitness sharing [12] or crowding [13] to promote the formation of stable sub-populations which enables us to find the all optimal routes.

In addition, a hierarchical approach can be adopted to improve the algorithm’s performance. A hierarchy can be introduced by splitting traffic networks into several smaller and less complex networks. The proposed GA described in this paper is much simpler than the ant based algorithm to solve a similar problem. Nevertheless, after doing intensive simulations on a network with more than 10000 nodes we observe that the proposed GA in the paper produces better results than ant based algorithm. In addition, it is relatively easy to modify the proposed single objective GA to a multi-objective GA that can do intelligent route planning. For example, if real-time information about the traffic situation is available (for example being updated on a server), the proposed GA can be modified to find a different set of optimal (or sub-optimal) routes (least congested, shortest, route along major roads, etc.). In other words, this proposed GA could be used as a starting point for developing an intelligent route planning system.

REFERENCES