The Seventeenth International Symposium on Artificial Life and Robotics 2012 (AROB 17th '12),
B-Con Plaza, Beppu, Oita, Japan, January 19-21, 2012

Multi-Objective Optimal Path Selection in the Electric Vehicles

Umair F. Siddiqi[1], Yoichi Shiraichi[1], and Sadiq M. Sait[2]

[1] Department of Production Science & Technology, Gunma University, Ota, Japan
[2] Department of Computer Engineering, King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia
(Tel & Fax: +81-276-50-2532)
({umair, siraisi}@emb.cs.gunma-u.ac.jp, sadiq@kfupm.edu.sa)

Abstract: Car navigation systems of the modern vehicles are equipped with an optimal path selection (OPS) unit. The OPS unit is responsible for finding the shortest paths between any source and destination nodes in the road network. In Electric Vehicles (EVs), the purpose of the OPS unit is to find multi-objective shortest paths (MOSP) w.r.t.: (i) Recharging Time, (ii) Distance, and (iii) Travelling Time. This work presents a memory efficient Simulated Evolution (SimE) based algorithm for solving the MOSP problem in EVs. The proposed algorithm uses innovative representation of the solution, and problem-specific goodness and allocation operations. Two different techinques for selecting the recharging stations are also proposed. The performance of the proposed algorithm is compared with NSGA-II, which is a popular population based heuristic for solving the multi-objective optimization problem. The comparsion results show that the proposed algorithm achieves the performance equal to NSGA-II while it requires 2.22 times lesser memory than any population based heuristic.

Keywords: Multi-Objective Optimization, Electric Vehicles, Simulated Evolution

1 INTRODUCTION

Car navigation systems of the modern vehicles are equipped with an optimal path selection unit. The optimal path selection unit is responsible for finding the shortest paths between any source and destination nodes in a road network. The optimization objectives in the internal combustion engine (ICE) based vehicles are: minimizing the distance, travelling time, traffic, etc. The Electric Vehicles (EVs) are also gaining popularity [1]. The optimal path selection in EVs should include minimization of the recharging time. The recharging time of the EVs can generally vary from 10 minutes to 30 minutes and therefore, it is an important factor to be considered in optimal path selection.

The main obstacles in the popularity of the EVs are: (i) Limited battery life, the EVs can travel until the time duration which is equal to the time limit of their batteries without recharging. A general value for the time limit is 2 hours. However, the time limits of the EVs are very small as compared to the ICE based vehicles. (ii) Lack of the infrastructure for the recharging stations. The recharging stations should be placed at regular intervals to ensure that the EVs can travel without any recharging problem. (iii) The recharging time of the EVs is also important. The recharging of the EVs should be performed in minimum possible time like 10 minutes. Many researchers have investigated the feasibility and positioning of the recharging stations [1, 2, 3, 4]. The results of the previous research showed that portable or fixed recharging stations could be placed at suitable places along the road, which ensured that the majority of the EVs could travel without any problem [7]. The rapid recharging technology has enabled full recharging of the EV in 10 minutes. Besides rapid recharging, battery swapping technology is also available which has recharging time as low as 5 minutes. However, battery swapping is expensive to implement than recharging.

The problem of finding optimal path between a source and destination node in the road network is a multi-objective shortest path (MOSP) problem. In this work, the MOSP problem has the following objectives: (i) Minimizing the recharging time, (ii) Minimizing the travelling distance, and (iii) Minimizing the travelling time. The MOSP is an NP-hard discrete optimization problem [5, 6]. The objectives in any multi-objective optimization problem are often contradictory therefore, no one solution can be optimal in all objectives. Therefore, a set of Pareto optimal solutions is found for the multi-objective optimization problem. The Pareto optimal set contains the non-dominated solutions. Evolutionary computation (EC) algorithms have been predominately used to solve multi-objective optimization problems. In many EC algorithms the calculation in any iterations does not depends on the previous iteration. Therefore, they are robust to dynamic changes in the parameter values of the road network like changes in traffic, waiting delay at the recharging stations, etc. The performance of any EC is measured by the number of Pareto optimal solutions it returns and the diversity of the solutions. For practical applications other properties of the EC algorithms like computation time and memory requirements are also very important. The research work in this paper, focused on developing a memory and time efficient EC algorithm for solving the MOSP problems in Elec-

tric Vehicles. The objectives of the MOSP problem, can be in contradiction with each other based on the traffic situation or the queues at the recharging stations. For example, when the shortest paths becomes congested then the travelling time on them can exceed the other paths. The proposed algorithm can find MOSP despite the changing traffic conditions. Simulated Evolution (SimE) [9] is a popular EC algorithm which works on only one solution at a time and is therefore, memory efficient than the other population based algorithms like Genetic Algorithms (GA), etc.

This paper proposed a SimE based multi-objectives optimization algorithm for solving the MOSP problem in EVs. The proposed SimE based algorithm has innovative goodness and allocation operations which are tailored for solving the MOSP problem. The performance of the proposed algorithm is compared with a famous heuristic: Non-dominated Sorting Genetic Algorithm (NSGA) -II [8]. The NSGA-II is a population-based algorithm for solving the multi-objective optimization problems. The simulation results show that the proposed algorithm can provide results equal to or better than the NSGA-II while it requires lesser memory than NSGA-II.

This paper is organized as follows: Second section shows the formal description of the problem. Third section contains the detail of the proposed algorithm. Fourth section contains the simulation results and comparison of the proposed algorithm with the other algorithms. The last section contains the conclusion.

2 DESCRIPTION OF THE PROBLEM

2.1 System Overview

In the car navigation system, the driver selects the source and destination nodes of his/her journey. The navigation system finds one or more optimal paths between the source and destination nodes. The navigation systems of the modern vehicles comprise of an embedded system which has interface to the GPS (Global Positioning System). The selection of the optimal path can either be performed independently by each vehicle or the optimal path selection is performed at any centralized station for a group of vehicles.

This work assumes that the optimal path selection is performed independently by the vehicles. Therefore, optimal path selection is performed in the embedded system of the car navigation system and it uses the updated information related to the road network from the GPS. The GPS system can provide the information related to the location of recharging stations and their recharging times to the EVs. The block diagram of the Optimal Path Selection (OPS) unit of an EV is shown in Fig. 1. The figure shows that the OPS unit receives inputs from the GPS and source and destination nodes from the driver. The OPS searches the optimal path based on the provided information and yields one or more optimal paths to the display unit of the car navigation system.

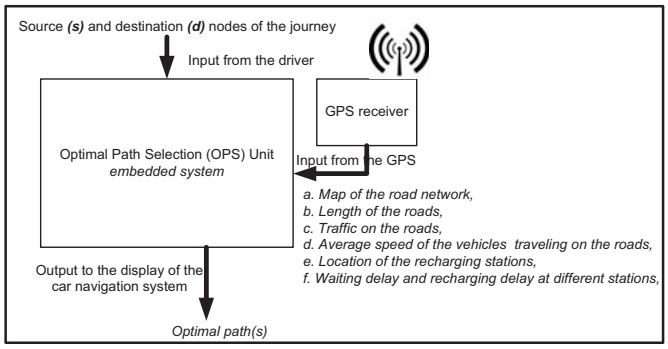

Fig. 1: Block diagram of the Optimal Path Selection (OPS) Unit of the EVs.

Table 1: Properties associated with edges, recharging stations and electric vehicles

Element	Symbol	Description
edge ($e_i \in E$)	$e_i.l$	Length of the edge
	$e_i.S$	Average speed of the EVs
recharging station ($r_j \in R$)	$r_j.R_{delay}$	Recharging time per EV
	$r_j.W_{delay}$	Waiting delay per EV due to queuing in it
	$r_j.e$	$e \in E$, s.t. r_j lies along the edge e
	$r_j.p$	Distance of the r_j from the start of $r_j.e$
	$r_j.D$	Travelling time of r_j from the source node
electric vehicle (EV)	t_{LIMIT}	The maximum time which the EV can travel without recharging
	s	Source node of the journey
	d	Destination node of the journey
	B_s	The battery level at the source node

2.2 Description of the Road Network

Let us consider that in the OPS unit, the electric vehicle is represented as EV and the road network is represented by an undirected graph (G), which is represented as: $G = (V, E, R)$, where V is the set of all vertices or intersections in the road network, E contains the road segments which join the intersections and R contains the recharging stations which exist in the road network. When the road network contains a total of N nodes and M edges, then $V = \{n_0, n_1, ..., n_{N-1}\}$ and $E = \{e_0, e_1, ..e_{M-1}\}$. Any edge $e_i \in E$, $e_i = (n_x, n_y)$, where $n_x \neq n_y$ and $n_x, n_y \in V$. n_x and n_y are the starting and ending nodes of the edge e_i. When the road network contains a total of m recharging stations, then the set R is represented as: $R = \{r_0, r_1, ..., r_{m-1}\}$. The properties associated with the edges, recharging stations and the electric vehicle are shown in Table. 1.

2.3 Description of the Multi-objective Optimization Problem

Let us consider that the solution in the OPS unit is represented as: $P = \{P^A, P^B\}$. P^A stores the edges from E which form a path from the source to the destination node and is represented as: $P^A = \{e_i, e_j, .., e_k\}$, $e_i = (s, n_x)$ and

Input: nodes: u, & v, G=(V,E), N_e= Number of elements in E
Output: Q: A path from u to v nodes.
1: $W = random(M)$
2: Q= Apply Dijkstra's Algorithm (u, v)
3: **return** Q

Fig. 2: Method to find paths: $form\_path(u, v)$.

$e_k = (n_y, d)$, and $s, n_x, n_y, d \in V$. P^B stores the recharging stations from R on which EV should be recharged in order to complete its journey on the path P^A.
The optimization performed in the OPS unit aims to find the paths which are optimized w.r.t. three functions: (i) $f_1(P^B)$, which corresponds to the total recharging time. (ii) $f_2(P^A)$, which corresponds to the total distance and (iii) $f_3(P^A)$, which is the total travelling time. The values of the functions can be computed as follows:

$$f_1(P^B) = \sum_{r_j \in P^B} (r_j.R_{delay} + r_j.W_{delay}) \quad (1)$$

$$f_2(P^A) = \sum_{e_i \in P^A} e_i.l \quad (2)$$

$$f_3(P^A) = \sum_{e_i \in P^A} \frac{e_i.l}{e_i.S} \quad (3)$$

The objective function can be represented as: $Obj(P)$= $Minimize(f_1(P^B), f_2(P^A), f_3(P^A))$. The minimization is performed through the proposed SimE-based algorithm.

3 PROPOSED ALGORITHM

Simulated Evolution (SimE) [9] is a general search strategy for solving a variety of combinatorial optimization problems. The SimE optimization loop consists of the following steps: (i) Evaluation, in which goodness of each element in the population is evaluated. (ii) Selection, in which up-to $n_s\%$ elements in the current solution or P^A having lower goodness values are selected into the set S. (iii) Allocation, in this step the elements in the set S will undergo through the allocation process. The optimization iterations are continued until the stopping criteria which can be the maximum time is reached. This work proposes a design of SimE which has innovative goodness and allocation operations which are tailored for the proposed multi-objective optimization problem. The proposed algorithm stores all non-dominated solutions found during the optimization iterations, which forms its set of Pareto optimal solutions. However, storing only one non-dominated solution which has the maximum value of the hypervolume is sufficient for the proper execution of the proposed algorithm.

In the following the proposed goodness and allocation operations are described in detail. First a method of building a random path from nodes u to v is shown which will be used to build a random path between any two nodes. The algorithm is shown in Fig. 2 and is represented by the function $form\_path(u, v)$. In line 1, W stores M randomly generated integers between $[1, 1000]$. The elements in W are used to assign weights to the edges in E. The Dijkstra's Algorithm finds shortest path w.r.t. the weights assigned to edges in line 1.

3.1 Goodness Function

In SimE, the goodness is defined as the ratio of the estimated minimum cost to the actual cost of the elements. In this work, the solution is represented as $P = \{P^A, P^B\}$. In the Evaluation step, the goodness of all elements in the set P^A is determined. The P^A comprises of a set of edges, and the starting node of the first edge in it is the source node and the ending node of the last edge in it is the destination node. Before goodness calculation is performed, the weights (w_1, w_2, w_3) are assigned to all edges in E. The procedure to assign weights to any edge $e_i \in E$ is shown in Fig. 3.

The weight w_1 corresponds to the average of the recharging and waiting delays of all the recharging stations which exists along the edge e_i. The term in the denominator indicates the total number of recharging stations on any edge [7], which is used to calculate the average. The weight w_2 corresponds to the length of the edge and w_3 corresponds to the travelling time on the edge. The proposed procedure to find the goodness of any edge is shown in Fig. 4. In Fig. 4, the goodness of the edge $e_i \in P^A$ is calculated. The variable r_1 holds the number of edges in P^A which have w_1 value lesser than $e_i.w_1$. Similarly, the variables r_2 and r_3 holds the number of edges in P^A which have w_2 and w_3 values lesser than $e_i.w_2$ and $e_i.w_3$ respectively. At the end, the average of the r_1, r_2 and r_3 variables is calculated.

Input: Any edge $e_i \in P^A$, $P = \{P^A, P^B\}$
Output: $e_i.w_1$, $e_i.w_2$, $e_i.w_3$
1: $e_i.w_1 = \frac{\sum_{r_j \in R \& r_j.e = e_i} (r_j.R_{delay} + r_j.W_{delay})}{0.05 \times e_i.l}$
2: $e_i.w_2 = e_i.l$
3: $e_i.w_3 = \frac{e_i.l}{e_i.S}$
4: **return** $\{e_i.w_1, e_i.w_2, e_i.w_3\}$

Fig. 3: Procedure for the assignment of the weights to any edge $e_i \in P^A$.

3.2 Allocation

The selection operation populates the set S by selecting up-to $n_s \times s$ elements from P^A. $n_s \in [0, 1] \in R^+$. The selection operation selects the elements from P^A which are not already selected and have lowest goodness value with probability P_s and any randomly selected element with probability

Input: Any edge $e_i \in P^A$, $P = \{P^A, P^B\}$
Output: $g(e_i)$: goodness of the edge e_i

$r_1 = 0, r_2 = 0, r_3 = 0$
n =Number of elements in P^A
for $j = 0; j < n; j++$ **do**
  if $P^A[j].w_1 > e_i.w_1$ **then**
    r_1++
  end if
  if $P^A[j].w_2 > e_i.w_2$ **then**
    r_2++
  end if
  if $P^A[j].w_3 > e_i.w_3$ **then**
    r_3++
  end if-[]
end for
$g(e_i) = \frac{r_1+r_2+r_3}{3}$
return $g(e_i)$

Fig. 4: Goodness Function $g(e_i)$, $(e_i \in P^A)$.

$P_s - 1$. The allocation operation is applied after the selection operation and it uses the elements in S.

The proposed allocation operation is shown in Fig. 5 using the pseudo code. The variable P_{min} is initialized to P^A. As shown, the allocation operation consists of doubly nested for loops. The outer-most loop executes for N_s number of times and the inner loop executes for each element in S. In the inner loop, an element from S is selected in each iteration. A random sub-path which exists from the starting node of the selected edge to the destination node is created. Then the sub-path is concatenated with the portion of the original solution (i.e., P^A) from the first edge to the edge which lies just before the edge selected from S. The resultant of the sum of the weights (or the square root of the sum of the squares of the three weights) of the new path is calculated and if the value is lesser than the P_{min} then P_{min} is updated to the new solution. At the end of the doubly nested for loops, the path which has minimum value of the resultant of the sum of the weights is returned and P^A is updated.

3.3 Mutation

The mutation operation is applied with probability M_b, In the mutation operation, the complete path which is stored in P^A is replaced by another path from source to destination. The new path is formed using the $form\_path$ function.

3.4 Selection of the Recharging Stations

The travelling limit (t_{LIMIT}) of the EV is the maximum time until which the EV can travel without recharging. Therefore, the EV requires periodic recharging during its journey. The set P^B in P stores the recharging stations on which the EV should be recharged. This subsection

Input: $S = \{x, y, ..., z\}$, $P = \{P^A, P^B\}$, $N_s \in Z^+$
Output: P^A (which is the path after the allocation operation)

s = Number of elements in S, $P_{min} = P^A$
n = Number of elements in P^A
for $i = 0; i < N_s; i++$ **do**
  for $j = 0; j < s; j++$ **do**
    $snode = P^A[S[j]].startnode$, $enode = P^A[n-1].endnode$
    $t1$ = $FormPath(snode, enode)$
    t = $concatenate\{P^A[0...S[j]-1], t_1\}$
    $v_1 = \sum_{e_i \in t} e_i.w_1$, $v_2 = \sum_{e_i \in t} e_i.w_2$, $v_3 = \sum_{e_i \in t} e_i.w_3$
    $R_1 = \sqrt{v_1^2 + v_2^2 + v_3^2}$
    $v_4 = \sum_{e_i \in P_{min}} e_i.w_1$, $v_5 = \sum_{e_i \in P_{min}} e_i.w_2$, $v_6 = \sum_{e_i \in P_{min}} e_i.w_3$
    $R_2 = \sqrt{v_4^2 + v_5^2 + v_6^2}$
    if $R_1 < R_2$ **then**
      $P_{min} = t$
    end if
  end for
end for
$P^A = P_{min}$
return P^A

Fig. 5: Allocation Operation.

shows the methods of selecting recharging stations for P^B. The proposed methods assumed that P^A is already formed and is not null. Before the recharging stations are selected, the travelling time of all recharging stations which lie along the edges in P^A is determined. For any recharging station $r_j \in P^B$, its travelling time from the source node (s) is stored in $r_j.D$. Equation (4) shows the calculation of the $r_j.D$ value. The equations assumes that r_j lies along e_i and function $pos(e_x)$ returns the position of any edge in the set P^A. $pos(e_x) < pos(e_i)$, if e_x lies before e_i in P^A.

Let us consider: $r_j.e = e_i$, then

$$r_j.D = \sum_{e_x \in P^A \& pos(e_x) < pos(e_i)} \frac{e_x.l}{e_x.S} + \frac{r_j.p}{e_i.S} \quad (4)$$

The proposed work uses two different approaches to select the recharging stations. The first approach is the Last Station Strategy (LSS) and the second approach is the Random Selection Strategy (RSS). In LSS, the recharging station which has travelling time (or D value) closest but not greater than the time which the EV can travel without recharging is selected. The LSS approach is shown using flow chart in Fig. 6(a). In the RSS approach, a recharging station is randomly selected from among the candidate recharging stations. The RSS approach is shown using flow chart in Fig. 6(b). To

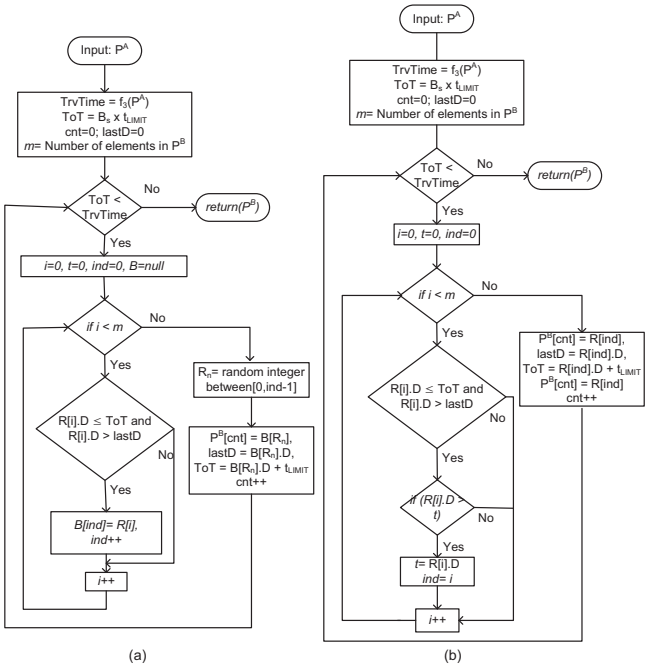

Fig. 6: Selection of recharging stations

populate the set P^B, the proposed algorithm applies both the LSS and RSS techniques. Let us consider that the result from the LSS is P^B and the result from the RSS is P^B.

3.5 Estimation of the Memory Requirements

The SimE works on only one solution. The proposed algorithm creates additional paths which include: Storing a solution which has minimum value of the product: $f_1(P^B) \times f_2(P^A) \times f_3(P^A)$. The allocation operation creates two new paths (t and P_{min}). In the techniques for the selection of recharging stations, the LSS creates one path and RSS creates two paths. The calculation of the number of paths which should be stored in the memory is as follows:
Let us consider that P^A or P^B requires δ units of memory. The maximum memory required is: $mem_{proposed}$= Memory required by the current solution (2δ) + Memory required by the allocation operation (2δ) + Memory required in storing a non-dominated solution (2δ) + memory required in selecting the recharging stations (3δ). Therefore, the total memory requirement is equal to 9δ. For comparison purposes, the minimum memory required by any GA (Genetic Algorithm) is also calculated. The GA stores a population of N elements, and therefore requires: mem_{GA}= $N \times 2\delta$ units of memory. The ratio $\frac{mem_{GA}}{mem_{proposed}} = \frac{2N}{9}$. In GA, the population size i.e. N ≥ 10 [8].

4 SIMULATIONS

The proposed algorithm is implemented using Java and its performance is compared with Non-dominated Sorting Genetic Algorithm - II (NSGA-II). The NSGA-II also optimizes

Table 2: Results of the Wilcoxon Rank-Sum tests.

Road Network	p	h	Remarks
$RG1$	0.9031	0	The Hypervolume distributions are same.
$RG2$	0.9676	0	The Hypervolume distributions are same.
$RG3$	0.4567	0	The Hypervolume distributions are same.
$RG4$	0.0858	0	The Hypervolume distributions are same.

paths w.r.t. the recharging time, distance and travelling time. NSGA-II uses the LSS scheme for selecting the recharging stations. The NSGA-II was implemented with population size (N) equal to 10. The ratio $\frac{mem_{GA}}{mem_{proposed}}$= 2.22. In NSGA-II, the population size is 2N, therefore, the ratio becomes 4.44. The proposed algorithm was implemented with parameter values: N_s= 5, i.e., the loop im the Allocation operations executes for five times, and M_b, which is the mutation probability is set to 0.60. The maximum time for the optimization loop is set to 10 seconds in both the proposed algorithm and in NSGA-II. Four road networks $(RG1, RG2, RG3, RG4)$ of different complexities are generated using a research tool [10]. The complexity of the road networks are as: RG1 having 1800 edges and 600 nodes, RG2 having 900 edges and 300 nodes, RG3 having 3400 edges an 1000 nodes, and RG4 having 4700 edges and 1400 nodes. The values to road lengths are randomly assigned to integers between [1, 400] km. The average speed on the edges are assigned to randomly selected integers between [40,120]. The waiting and recharging delay at the recharging stations varies randomly between $[10, 20] \in R$. The maximum time which the EV can travel without recharging is considered to be 2 hours. In any test case the source and destination nodes are randomly selected and the algorithm are used to find the optimum paths between them. On each graph, up-to 20 test problems are executed. In each test problem, the Pareto sets are obtained from both algorithms.

The 3D Hypervolume calculation is performed by using the tool [11], which was proposed by Carlos Fonseca et al.. The tool calculates the Hypervolume for the minimization problems. In the Hypervolume calculation, the bounds for the maximum values are considered more than the values in the Pareto Optimal sets. The larger Hypervolume value represents that the Pareto optimal set consists of better solutions. The results are shown using Box-and-Whisker plots in Fig. 7. The box-and-whisker plots, the whiskers are drawn between the minimum and maximum values. The central mark is the median, the edges of the boxes are at the 25th and 75th percentiles. The Wilcoxon Rank Sum test [12] is used to test the null hypothesis, which shows that the Hypervolume distributions obtained from the two algorithms are same. The rank sum test returns p and h values. If $h = 0$, which means that the two distribution are same. p indicates the probability that an element from the first distribution is equal to

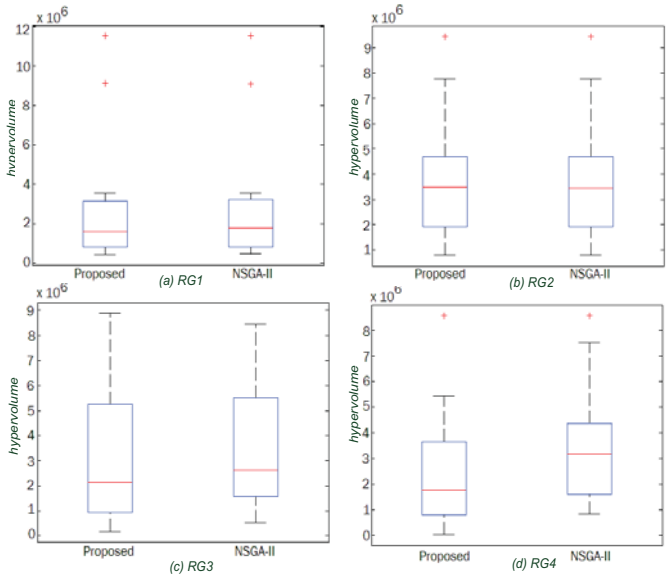

Fig. 7: Box-and-Whisker Plots of the Hypervolumes on different graphs.

the element from the second distribution. The MATLAB's $ranksum$ function is used to apply the rank sum test. The results are shown in Table 2, which shows that the Hypervolumes obtained fromthe proposed algorithm and NSGA-II on different graphs are same. Therefore, the Pareto Optimal solutions obtained from the proposed algorithm are as good as obtained from the NSGA-II.

5 CONCLUSION

In this work, the multi-objective route optimization problem for the Electric Vehicles (EVs) is solved by using the Simulated Evolution (SimE) based algorithm. The optimization objectives are: recharging time, distance and travelling time. Innovative and problem specific Goodness and Allocation operations are being used to achieve better performance. The comparison of the proposed algorithm with NSGA-II, which is a popular GA based heuristic for the multi-optimization problems shows that the proposed algorithm achieves same results as NSGA-II. However, the proposed algorithm requires memory which is 2.2 times lesser than any GA or population based algorithm. This memory size reduction is effective for implementing an embedded hardware.

REFERENCES

[1] Ahmed Y. Saber & Ganesh K. Venayagamoorthy, "One Million Plug-in Electric Vehicles on the Road by 2015," Proceedings of the 12^{th} International IEEE Conference on Intelligent Transportation Systems, St. Lious, MO, USA, Oct. 3-7, 2009.

[2] Medhi E.-Amoli, Kent Choma, & Jason Stefani,"Rapid-Charge Electric-Vehicle Stations," IEEE Transactions of Power Delivery, Vol. 25, No. 3, pp. 1883-1887, July 2010.

[3] Zheng Li, Zafer Sahinoglu, Zhifeng Tao, & K. H. Teo, "Electric Vehicles Network with Nomadic Portable Charging Stations," 2010 IEEE 72^{th} Conference on Vehicular Technology Fall, VTC-2010 Fall, Ottawa, ON, USA, Sept. 6-9, 2010.

[4] Olle Sundstorm & Carl Binding, "Planning Electric-Drive Vehicle Charging under Constrained Grid Conditions," 2010 Internal Conference on Power System Technology, (POWERCON), Zhejiang, China, Oct. 24-28, 2010.

[5] Zbigniew TARAPATA, "Selected Multicriteria Shortest Path Problems: AN Analysis of Complexity, Models and Adoption of Standard Algorithms," Int. J. Appl. Math. Comput. Sci., 2007, Vol. 17, No. 2, 269-287.

[6] M.R. Garey and D. S. Johnson: *Computers and Intractability: A Guide to the Theory of NP-Completeness.*, W. H. Freeman and Co., 1997

[7] Zheng Li, Zafer Sahinoglu, Zhifeng Tao, & K. H. Teo, "Electric Vehicles Network with Nomadic Portable Charging Stations," 2010 IEEE 72^{th} Conference on Vehicular Technology Fall, VTC-2010 Fall, Ottawa, ON, USA, Sept. 6-9, 2010.

[8] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," IEEE Trans. Evolutionary Computation, vol. 6, No. 2, April 2002.

[9] S.M. Sait & H. Youssef, Iterative Computer Algorithms with Applications in Engineering, IEEE Computer Society Press, 1999.

[10] Fabien Viger, Matthieu Latapy, "Efficient and simple generation of random simple connected graphs with prescribed degree sequence," Proc. 11^{th} Conference of Computing and Combinatoric (COCOON 2005), LNCS 3595, pp. 440-449, 2005.

[11] Carlos M. Fonseca, Lus Paquete, and Manuel Lpez-Ibez,"An improved dimension - sweep algorithm for the hypervolume indicator," Proc. 2006 IEEE Congress on Evolutionary Computation (CEC'06), pp. 11571163, Piscataway, NJ, July 2006.

[12] Hollander, M., and D. A. Wolfe, $Nonparametric\ Statistical\ Methods$, John Wiley & Sons, Inc., 1999.