Turk J Elec Engin, VOL.15, NO.2 2007, © TÜBİTAK

A VRP-Based Route Planning for a Mobile Robot Group

Osman PARLAKTUNA[1], **Aydın SİPAHİOĞLU**[2], **Ahmet YAZICI**[3]
[1]*Electrical Engineering Department, Eskişehir Osmangazi University, Eskişehir-TURKEY*
e-mail: oparlak@ogu.edu.tr
[2]*Industrial Engineering Department, Eskişehir Osmangazi University, Eskişehir-TURKEY*
e-mail: asipahi@ogu.edu.tr
[3]*Computer Engineering Department, Eskişehir Osmangazi University, Eskişehir-TURKEY*
e-mail: ayazici@ogu.edu.tr

Abstract

In this study, a vehicle routing problem-based approach is presented to construct non-intersecting routes for the members of a mobile robot team. It is assumed that each robot starts from a central location such as the charging point, completes its route and returns to the starting location. The proposed method consists of three algorithms: a sweep algorithm determines the position of each node in clockwise (or counter clockwise) manner with respect to the starting location; savings algorithm calculates the saving obtained by adding a node to the route of a robot; Dijkstra's shortest path algorithm is used to calculate the shortest distance from any node to another one when the network is sparse. Simulations are performed using some benchmark VRP problems and results are compared with the optimal solution of the same problems. It is shown that our approach constructs routes significantly fast with near optimal energy consumption.

1. Introduction

Many missions in autonomous mobile-robot systems depend on navigating in a known or an unknown environment and performing some tasks, like land-mine exploration, post-office automation, cleaning, transporting load from one node to another, or assisting rescue after disasters. A multi-robot cooperative system appears to be more effective and adaptive to accomplish various such kinds of complex tasks, relative to a single robot approach, and most of these missions may be performed more efficiently by a collaboratively working multi-robot team [1].

In many multi-robot applications, there exists a need to coordinate the actions of robots to achieve a goal. When many robots operate in the same environment, a motion planning is required for the robots to reach their goals while avoiding collisions among themselves. Besides, with the development of cooperative robotics in the recent years, the problem of path planning for robot groups is receiving attention increasingly [2]. Somhom et al., [3] introduced an algorithm in competition-based network to solve the min-max multiple traveling salesmen problem (m-TSP) which needs the maximum distance among all salesmen to be minimum. Yu et al. [2] described in their work how to apply a Genetic Algorithm in finding a globally sub-optimal path for the robot group working under certain tasks. In [4], path planning of cooperative multi-mobile robot systems is discussed with the proposal of a novel Cooperative Co-evolutionary Adaptive Genetic Algorithm (CCAGA). Guo and Parker [5] proposed a distributed and optimal motion-planning algorithm for multiple robots. They decomposed the problem into two modules: path planning and velocity planning. Zu et al. [6] proposed a path-planning algorithm for multi-robot system, which finds an optimal path that satisfies some performance criteria, based on the depth-first search idea. In [7] a motion-planning framework is presented that enables multiple mobile robots with limited ranges of sensing and communication to maneuver and achieve goals safely in dynamic environments. In [8] a solution to the multi-robot motion-planning problem based on a decoupled analysis in the space domain and in the time domain is described. It investigates the practical use of the notion of motion plan quality and of the motion-plan robustness measures for computing safe motions. An approach to fuel-optimal path planning of multiple vehicles using a combination of linear and integer programming is presented in [9]. The approaches above and some others for motion coordination and motion planning problems in multi robot systems are summarized in [10,11].

A mission in multi-robot applications could consist of a series of locations to be visited in any order by any robot, or perhaps each robot should visit a series of locations before returning to its recharging area. In the literature, this problem is known as m-traveling salesmen problem (m-TSP). The aim of the m-TSP is to find tours for travelers such that the total distance traveled is minimized and all nodes are visited exactly once. In real applications, there are some constraints for travelers such as all travelers have certain capacities, all tasks should be performed in a specified time interval, each traveler should perform a certain number of tasks, or tour distance for each traveler should be less than a certain value. If capacity contraints exists, the problem is called vehicle routing problem (VRP). The aim of the VRP is to find a sequence of visiting places for each traveler to minimize the total distance while the capacity of each traveler is not exceeded.

Robot's energy capacity is an important constraint in task planning for a mobile robot. Robot consumes energy while it travels and/or performs the given tasks. Task planning algorithm should construct a tour where the required energy does not exceed energy capacity of the robot. This problem resembles the VRP problem, but it is slightly different because the robot consumes energy not only at the visited places but also during the travel between the places. This problem is solved by extending VRP to include the energy dissipated by the robot during the travel. In multi-robot applications, interaction between the robots may also cause some problems. To overcome this problem, a sweep algorithm is used. Therefore, in this paper, the extended VRP and the sweep algorithm are combined to construct nonintersecting tours for multi-robot group with energy capacities.

Remaining of the paper is organized as follows. In section II, a brief explanation of the route finding problem and algorithms used in the proposed method are given. Section III explains the proposed method and gives computation load of the proposed method with respect to the number of nodes and the number of robots. In section IV, simulations and experimental results are given. Finally, section V concludes the paper.

2. Preliminaries

TSP and VRP are of the most widely studied combinatorial optimization problems [12]. It is believed that the report by Menger (1932) is the first published work on TSP [13]. However, a breakthrough in this area was done by Dantzig, Fulkerson and Johnson in 1954. To the best knowledge of the authors, the first study about m-TSP was done by Svestka and Huckfield in 1973 [14], and a good survey for m-TSP formulations and solution procedures is studied by Bektas [15]. One of the first studies for VRP was done by Dantzig and Ramser in 1959 [16]. The books edited by Lawler et al. [17] and edited by Gutin and Punnen [13] provide detailed and comprehensive survey about TSP, its solution techniques, and applications. Another book edited by Toth and Vigo [18] provides detailed survey about VRP and its variants. TSPLIB [19] is also an important web site since it includes many test problems in this area.

There are many exact-solution methods and mathematical models for the TSP and the VRP. However, finding the optimal solution is not easy due to the NP-Hard nature of these problems. Therefore, heuristics are preferred instead of exact-solution methods. Although heuristics couldn't find the optimal solution, they find a near-optimal solution in a very short time. Developed heuristics for the TSP can be used for the VRP by making some minor modifications. Some definitions and heuristic algorithms for the TSP and the VRP are explained below.

A network can be represented as a graph G = (N, A) where N is a set of n nodes (vertices) and A is a set of arcs (edges). Nodes describe to be visited places and arcs describe links between nodes. A network can be classified as directed or undirected depending on the availability and direction of connections. If none of the arcs have a direction, this network is called as an undirected network. The matrix of distances between nodes is called the distance matrix and it can be symmetric ($d_{ij}=d_{ji}$) or asymmetric ($d_{ij} \neq d_{ji}$). Asymmetric matrix can occur if the network has at least one directed arc. Arcs may be bidirectional ($d_{ij} = d_{ji} > 0$), unidirectional ($d_{ij} > 0, d_{ji} = 0$), or closed to traffic ($d_{ij} = d_{ji} = 0$). The distance matrix can also be complete (each node is connected with the others) or sparse. In TSP, collection of nodes visited by the salesman is called a tour where the salesman starts and finishes at the same node. There are (n-1)!/2 possible tours in an n-node complete network. Number of possible tours increases exponentially by n which is the reason why solving the TSP or the VRP is difficult.

In our proposed method, three different algorithms are used. First, a sweep algorithm is used to cluster the nodes, then a Savings algorithm constructs the tours, and finally Dijkstra's [23] shortest path algorithm finds the shortest path between any pair of nodes if the network is sparse.

The Sweep algorithm used in this study is proposed by Wren and Holliday in 1971 [18]. The Sweep algorithm is simple and can be defined in 2 steps:

Step 1- The polar coordinates of every node are calculated and sorted in increasing polar angle order.

Step 2- Nodes are clustered such that the total capacity of each cluster does not exceed the capacity of the traveler.

In the algorithm any point can be used as the starting point of the sweep. Starting from the node having the smallest polar angle, nodes are assigned to the current traveler until the capacity of the traveler is not exceeded. The Sweep algorithm may be combined with other heuristics, i.e., a Savings Algorithm, to find the minimum total distance between the node clusters.

The Savings algorithm used in this paper is a typical tour construction algorithm and developed by Clarke & Wright in 1963. The aim of the algorithm is to maximize the distance saved over a previous tour at each step. Algorithm's steps are as follows [21]:

Step 1: Select any node as the origin and denote this as node 0.

Step 2: Produce different tours such that (0, i, 0), as initial configuration for all i.

Step 3: Calculate the savings $s_{ij} = d_{0i} + d_{j0} - d_{ij} \geq 0$ for all pairs of i and j.

Step 4: Sort the savings in non-increasing order.

Step 5: Find the first feasible arc (i, j) in the savings list where, i and j are on different routes and both i and j are either the first or the last visited node on their respective routes. Add arc (i, j) to the current tour and delete arcs (0, i) and (j, 0). Delete arc (i, j) from the savings list.

Step 6: Repeat step 5, until no (0, i, 0) routes remain.

An additional savings calculation method has been offered by Ballou [22]. This method permits the nodes of i and j not to be the first or last stop. For example, if a point C is inserted between stops A and B which are on the same route, then the savings can be calculated as s = $d_{0C}+d_{C0}+d_{AB}-d_{AC}-d_{CB}$. In this study, Ballou's calculation method is used.

If the network is complete, the distance matrix is formed the distance values between each pair of nodes of the network. However, if the network is sparse, there may be two or more path between two nodes. In this case, a new distance matrix should be calculated using the shortest distance between each pair of nodes in the network. In this study, Dijkstra's shortest path algorithm is used for this purpose.

Dijkstra's algorithm is commonly used to find the shortest paths from the starting node s to all other nodes in the network. Dijkstra's algorithm maintains a distance label d(i) for each node i, which is an upper bound for the shortest path length to node i. At any intermediate step, algorithm divides the nodes into two groups as permanent and temporary. The distance label to any permanent node represents the shortest distance from the source to that node [23].

Parameter definitions and steps as pseudo code of the Dijkstra's algorithm are given below.

N: Set of nodes

S: Set of permanent nodes

$\underline{S}$: Set of temporary nodes

d(i): minimum distance from s to i.

Pred(j): predecessor indices

```
Algorithm Dijkstra:
begin
    $S : \Phi$; $\underline{S} = N$;
    d(i): $= \infty$ for each node i $\in$ N;
    d(s): = 0 and pred (s):=0;
    while |S| < n do
    begin
        let i $\in \underline{S}$ be a node for which d(i) = min{d(j): j $\in \underline{S}$};
        S: = S $\cup$ {i};
        $\underline{S}$ : = $\underline{S}$ − {i};
        for each (i,j) $\in$ A(i) do
        if d(j) > d(i) + $c_{ij}$ then d(j):= d(i) + $c_{ij}$ and pred(j):= i
    end;
end;
```

Depending on the network structure, algorithms explained above are applied to tour construction problem for a multi-robot team.

3. Proposed Method

The proposed method considers tour construction problem for m-robot team, which requires visit of each node of an $n-$node network by at least one member of the team. Intersecting tours for the robots may produce undesirable effects on the performance of the team. Robots will consider each other as obstacles and try to avoid each other resulting longer traveling distances, loosing their tracks, etc. Therefore, the main objective of this study is constructing non-intersecting routes with near-minimum total energy consumption while considering energy capacities.

It is assumed that the robots know global positions, relative distances, and connections of the nodes. The task is completed when all the nodes are visited and the robots return to their home location (or charge location). Assume that E_{ij} denotes energy consumption during the travel from node i to node j, and E_i, $i = 1, \dots, n$ denotes required energy in performing its task at the node. Robots (R_k, $k = 1, \dots, m$) have limited capacity of energy (E_{k-cap}, $k = 1, \dots, m$) and must return their home location before consuming their full energy.

Proposed Algorithm:

Initialization Step:

i- Obtain all the information about the environment: Number of nodes (n), their coordinates(x_i, y_i), $i = 1, \dots, n$, required energy E_i, $i = 1, \dots, n$, status of the arcs (open-closed-unidirectional), distances between nodes (d_{ij} values), and required energy $E_{ij} = f(d_{ij})$ as a function of traveling distancesd_{ij}.

ii- Obtain all the information about the robots: Number of available robots m. Energy capacity (E_{k-cap}, $k = 1, \dots, m$) of robots.

iii- If there exist any unidirectional or closed arcs between any pair of nodes, use Dijkstra's algorithm to find minimum paths between that pair of nodes. Using these minimum paths, update distance matrix D such that all $d_{ij} > 0$.

iv- Set the consumed energy by each robot $E_{k-total\_load} = 0$ and overload flag of each robot$R_{k-Notfull} = 1$, $k = 1, \dots, m$. Assign all n-1 nodes except the starting node to an array P.

v- Set robot index k = 1.

Main Step: 1- Use the sweep algorithm to find a node. Assign it to a temporary node t_n. Using this temporary node, construct a temporary tour for the robotR_k by using Ballou's Savings algorithm. Calculate$E_{k-total_load} = E_i + E_{ij}$. If $E_{k-cap} \geq E_{k-total\_load}$ include this node in theR_k's tourR_{k_tour}, and remove this node from the array P, otherwise set $R_{k-Notfull} = 0$ and k = k+1.

2- If $\sum_{k=1}^{m} R_{k-Notfull} > 0$ and there are some remaining nodes in P go to step 1, otherwise STOP.

3.1. Complexity of the proposed approach

At each iteration of the savings algorithm, savings values are calculated and one of the eligible nodes is added to the route. Therefore, the required number of iterations is equal to number of the nodes. In classical savings algorithms, eligible node is added as the first or the last node of the tour. Thus, at each iteration, two savings calculations are required for each eligible node. However, the eligible node may be added between any two nodes in Ballou's savings algorithm. For that reason, savings calculations must be performed for all eligible nodes for each possible position. This situation increases the computational load of the algorithm. Since the Ballou's Savings algorithm is used in this study, the computational load of the algorithm for a single robot with unlimited capacity is calculated by the following equation:

$$fs(n) = \sum_{i=1}^{n-2} i + \sum_{i=1}^{n-3} i \cdot (n-i) = \sum_{i=1}^{n-2} i + n \cdot \sum_{i=1}^{n-3} i - \sum_{i=1}^{n-3} i^2.$$

where n is the number of nodes. This equation can be re-written as follows:

$$\begin{aligned} fs(n) &= \tfrac{(n-1)(n-2)}{2} + \tfrac{n(n-3)(n-2)}{2} + \tfrac{(n-3)(n-2)(2n-5)}{6} \\ &= 0.8333n^3 - 4.5n^2 + 7.6667n - 4 \end{aligned}$$

As seen from the equation, complexity of the algorithm is $O(n^3)$. When the robot has limited capacity, the proposed algorithm calculates savings obtained by the addition of each node to the route of the robot. If the capacity of the robot is exceeded by the addition of a node, the algorithm stops. Assuming that the robot has a capacity to visit all the nodes (worst case scenario for the calculation load), then the calculation load of the algorithm becomes

$$\begin{aligned} fs\_capacity(n) &= fs(1) + fs(2) + fs(3) + \cdots + fs(n-1) + fs(n) \\ &= 0.8333 \sum_{k=1}^{n} k^3 - 4.5 \sum_{k=1}^{n} k^2 + 7.6667 \sum_{k=1}^{n} k - 4n \\ &= 0.2083n^4 - 1.0833n^3 + 1.7917n^2 - 0.91667n \end{aligned}$$

Therefore, the complexity of the algorithm for a single capacitated robot in an n-node network is $O(n^4)$.

Multi robot case:

Considering m equal-capacity robots, work load may be distributed equally among the robots, then the calculations for the unconstrained case will be equal to m*fs(n_m) where n_m= n/m (the number of nodes each robot visits). Similarly, for the limited-capacity robots, number of calculations becomes m*fs_capacity(n_m). In both cases, number of calculations will be less than the corresponding number of calculations for the single robot case.

4. Application

Proposed method is applied in laboratory environment and results were presented in [24]. However, in this study, we rather concentrated on the analysis of the proposed method in terms of algorithm complexity, relation between solution time and number of nodes and/or robots, and compared the results of the proposed algorithm with the results of exact-solution methods. For this purpose, first, simulations are performed on an Eil22 VRP test problem for different number of robots. Then, solution times and values of the proposed approach are compared with the results of exact-solution methods using Eil22 and Eil33 VRP test problems. In order to find the optimal solutions of these test problems, the method proposed by Kara et. al. [20] is used.

4.1. Simulation Results of Eil22 VRP test problem

Eil22 VRP test problem is used in simulations for different number of robots (1-4). Nodes and required energy to accomplish a task in a given node are taken from the VRP test problem as follows [NodeNo E_i] = [1 0; 2 1100;3 700; 4 800;5 1400; 6 2100; 7 400; 8 800; 9 100; 10 500; 11 600; 12 1200; 13 1300; 14 1300; 15 300; 16 900; 17 2100; 18 1000; 19 900; 20 2500; 21 1800; 22 700]; For the simulations, energy consumed during the travel between the nodes is calculated as a function of traveled distance d_{ij} as$f(d_{ij}) = 10d_{ij}$ (note that for $f(d_{ij}) = 0$ problem turns to a classical VRP problem). To determine approximate required energy for this problem, Ballou's Savings algorithm is used for only one robot with infinite energy capacity. The constructed tour for the single robot is shown in Figure 1. In this case, the total energy to complete the task is calculated as 25382 units. Note that 88.65 percent of this energy is consumed during performing the tasks in the nodes and remaining 11.35 percent is consumed during the travel. In this simulation, the robot travels a distance of 288.2088 units.

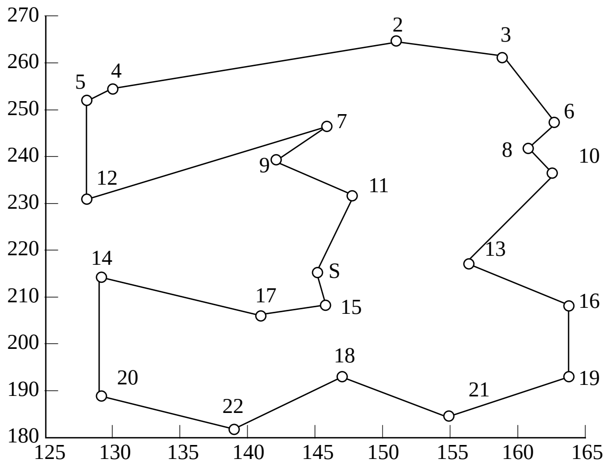

Figure 1. Constructed tour for a single robot in Eil22 VRP test problem.

As the second simulation, 2 robots with selected energy capacities of $E_{k-cap} = 13000$ units for $k = 1, 2$ are used to construct tours for the Eil22 VRP problem. Note that these energy capacities are selected as approximately half capacity of the single robot case. Figure 2 shows the constructed tours for these energy capacities. The total consumed energy is 25576 units and the maximum distance traveled by one of the robots is 171.3058 units.

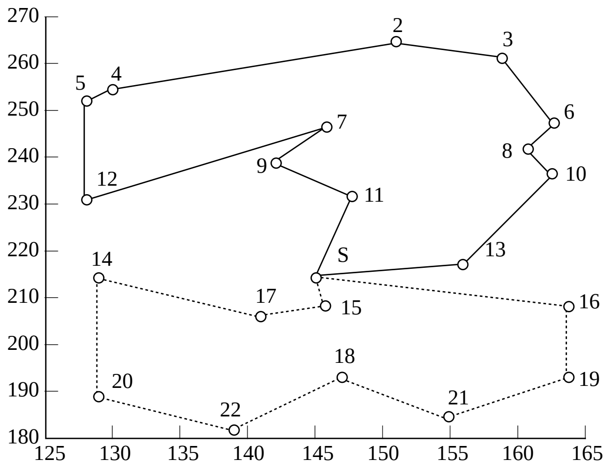

Figure 2. Constructed tours for 2 robots in Eil22 VRP problem.

As the third simulation, 3 robots with selected energy capacities of $E_{k-cap} = 9000$ units for $k = 1, 2, 3$ are used to construct tours for the Eil22 VRP problem. Figure 3 shows constructed tours. Total consumed energy is 26116 units and the maximum distance traveled by one of the robots is 138.1717 units.

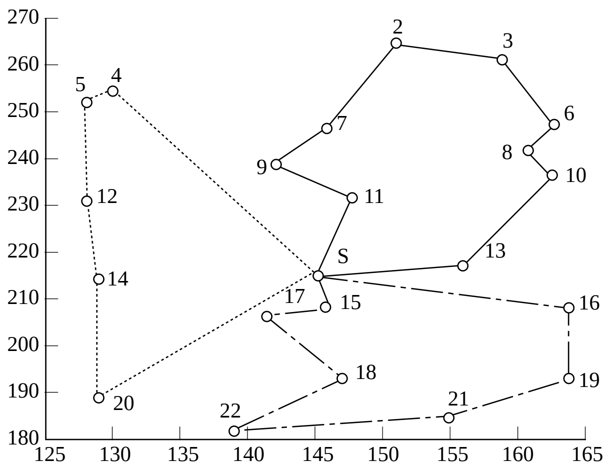

Figure 3. Constructed tours for 3 robots in Eil22 VRP problem.

As the last simulation, 4 robots with selected energy capacities of $E_{k-cap} = 7000$ units for $k = 1, ..., 4$ are used to construct tours for the Eil22 VRP problem. Fig. 4 shows constructed tours for these energy capacities. Total consumed energy is 26504 units and the maximum distance traveled by one of the robots is 123.6591 units.

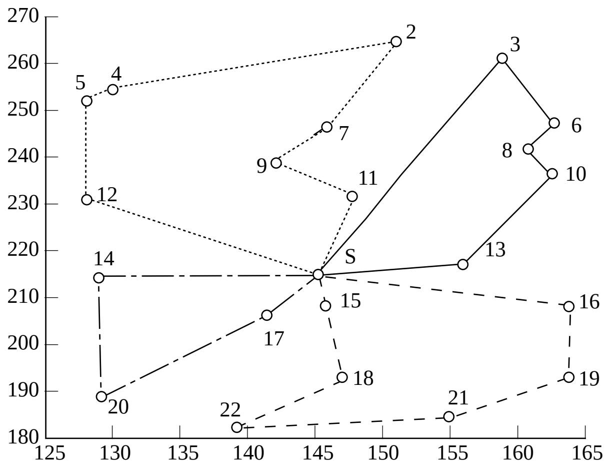

Figure 4. Constructed tours for 4 robots in Eil22 VRP problem.

As seen from the figures 2-4, the constructed tours are non-intersecting for all multi-robot cases. Therefore, the robots do not block each other during the tours. Total consumed energies change slightly depending on the number of robots. As the number of robot increases, maximum traveling distance of robots decreases. Therefore, if each robot is assumed to have constant speed, total time to complete the tours decreases as the number of robots increases.

4.2. Comparisons of heuristic and exact-solution method results

Since the exact-solution methods for standard VRP consider the energy spent during the travel to be zero, in order to compare results between the proposed method and the exact-solution method, the energy spent during travel is assumed zero. In this study integer linear programming is used to solve capacitated VRP as an exact-solution method. Models are solved by using Lingo 9.0 software in HP6000 workstation.

To make a comparison between the classical VRP method and the proposed method, two different test problems are chosen from TSPLIB: Eil22 and Eil33. The results are compared with respect to solution time and optimum solution. Eil22 and Eil33 test problem results are shown in Table 1 and Table 2, respectively. Both of the test problems are solved for different number of robots and different robot capacities.

Table 1. Results for Eil22 test problem.

Number of robots	Robot Capacity	z*	Time (hh:mm:ss)	z-savings	Time savings (seconds)	Performance difference (percent)
1	24000	278.437	00:00:23	288.20	0.92	-3.51 %
2	12000	289.8801	00:00:33	307.59	0.2	-6.11 %
3	8000	342.402	03:08:15	361.58	0.1	-5.6 %
4	6000	375.28	08:49:14	400.40	0.06	-6.7 %

In Table 1, different robot numbers and robot capacities are shown in the first and second columns. z* denotes the optimal solution value by using integer linear programming. Time values show the solution times for the optimal solution. In the fifth and sixth columns, the values obtained with proposed method are given. The last column shows the difference between solution of the proposed method and the optimal solution in percentage. It can be seen that, the performance of the proposed method is slightly worse than the optimal solution. However, the difference of the solution times is too big to ignore, and the difference between solution times increases as the number of robots increase. For instance, for four robots, it takes approximately 9 hours to find the optimal solution whereas the proposed method reaches the near optimal solution in only 0.06 seconds. Since the solution time is very important for real time tour constructing problems, the proposed method is preferable even if it can not find the optimal solution.

Table 2. Results for Eil33 test problem.

Number of robots	Robot Capacity	z	Time (hours)	z-savings	Time savings (seconds)	Performance difference (percent)
1	30000	463.638	9	451.0785	4.48	2.71 %
2	15000	771.518	9	570.2888	1.122	26.08 %
3	12000	797.392	9	685.9633	0.39	13.97 %
4	8500	664.278	9	878.6755	0.31	11.40 %

Since the optimal solution cannot be obtained in 9 hours for the Eil33 test problem, instead of z*, term z is used in column three. This value shows the best solution (not optimal) value of the problem after 9 hours. Of course, the proposed algorithm's results are not optimal solution. However, the results of the proposed algorithm are better than the exact-solution method. Furthermore, the solution times are too short to compare with solution times of the exact-solution method. For instance, for two robots, it takes only 1.122 seconds to obtain a good solution, and the objective value is 26.08% better than the exact-solution method.

Also, notice that there is a decrease in solution time of the proposed method when the number of robots increases, as seen Table 1 and Table 2. This is an expected result of the proposed method. For example, consider the Eil33 test problem. For a single robot, calculation load of the algorithm is $\mathit{fs\_capacity}(33) = 210047$. For 3 robots (assuming each robot visits equal number of nodes, 11) calculation load of the algorithm is $3{*}\mathit{fs\_capacity}(11) = 5471$.

5. Conclusions and Future Work

In this paper, a tour-construction algorithm that considers capacity of the robots is proposed for a multi-robot team. As shown by simulations, the proposed method determines non-intersecting routes for the members of the team in a known and static environment.

Due to NP-hard nature of the VRP, solution produced by the exact-solution methods takes longer time as the number of nodes increases. There is also an increase in the solution time when the number of robots increases. Compared to exact-solution method, the algorithm reaches the solution in a very short time with an acceptable solution value. Moreover, the solution time of the proposed algorithm decreases as the number of robots increases.

In this paper, only the energy capacity of the robots is considered. However, other types of capacity problems such as each traveler should perform a certain number of tasks or tour distance for each traveler should be less than a certain value may be easily solved by the proposed method.

The heuristic used here has also been applied to single robot case in dynamic environment [25]. Therefore, the method of this paper may also be expanded to multi-robot problems in dynamically changing environments.

Acknowledgements

This work is partially supported by State Planning Organization of Republic of Turkey under the project "Accomplishing a given task by a homogeneous mobile robot group using dynamic role assignment". 2005-continues.

References

[1] R. R. Murphy, Introduction to AI Robotics, The MIT Pres, 2000.

[2] Z. Yu, L. Jinhai, G. Guochang, Z. Rubo, Y. Haiyan, "An implementation of evolutionary computation for path planning of cooperative mobile robots," in *Proc 4th World Congress on Intelligent Control and Automation*, pp. 1798-1802, 2002.

[3] S. Somhom, A. Modares, T. Enkawa, "Competition-based neural network for the multiple traveling salesmen problem with min-max objective," *Computers & Operations Research,* vol. 26, pp. 395-407, 1999.

[4] Z.X. Cai, Z.H. Peng, "Cooperative Co-evolutionary Adaptive Genetic Algorithm in Path Planning of Cooperative Multi-Mobile Robot Systems", Journal of Intelligent & Robotic Systems, vol.33 (1), pp.61-71, 2002.

[5] Y. Guo, L. E. Parker, "A distributed and optimal motion planning approach for multiple mobile robots," in *Proc IEEE International Conference on Robotics & Automation* Washington DC, pp. 2612-2619, May 2002.

[6] L. Zu , Y. Tian, J. Liu, "Algorithms of task-allocation and cooperation in multi mobile robot system," in *Proc 5th World Congress on Intelligent Control and Automation*, Hangzhou, P.R. China, pp. 2841-2845, 2004.

[7] M.C. Clark, S.M. Rock, J.C. Latombe, "Motion Planning For Multiple Mobile Robot Systems Using Dynamic Network", Proceedings of the 2003 IEEE International Conference on Robotics & Automation, pp.4222-4227, 2003.

[8] C. Ferrari, E. Pagello, J. Ota , T. Arai, "Multi-robot Motion Coordination in Space and Time", Robotics and Autonomous Systems, vol. 25 (3-4), pp.219-229, 1998.

[9] T. Schouwenaars, B.D. Moor, E. Feron, J. How, "Mixed Integer Programming For Multi-Vehicle Path Planning", European Control Conference, pp. 2603-2608, 2001.

[10] T. Arai, E. Pagello, L.E. Parker, "Advances in Multi-Robot Systems", IEEE Transactions on Robotics and Automation, vol. 18 (5), pp.655-661, 2002.

[11] J. Ota, "Multi-Agent Robot Systems as Distributed Autonomous Systems", Advanced Engineering Informatics, vol.20(1), pp.59-70, 2006.

[12] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *EJOR*, Vol.59, pp. 231-247, 1992.

[13] A. Punnen, "The traveling salesman problem applications, formulations and variations," in *The traveling salesman problem and its variations*, edited by Gutin G., Punnen A.P. Kluwer Academic Publishers, 2002.

[14] R.V. Kulkarni, P.R. Bhave, "Integer programming formulations of vehicle routing problem," *EJOR*, Vol. 20, pp. 58-67, 1985.

[15] T. Bektas, "The multiple traveling salesman problem: An overview of formulations and solutions procedures", *Omega-International Journal of Management Science*, vol.34, no.3, pp.209-219, 2006.

[16] G. Laporte, "The vehicle routing problem: An overview of exact and approximate algorithms," *EJOR*, Vol. 59, pp. 345-358, 1992.

[17] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy-Kan, D.B. Shmoys, *The traveling salesman problem*, John Wiley & Sons, Newyork, 1985.

[18] G. Laporte, F. Semet, "Classical heuristics for the capacitated VRP," in *The vehicle routing problem* edited by Toth P., Vigo D., SIAM, 2002.

[19] TSPLIB, 2006, Available: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

[20] I. Kara, G. Laporte, T. Bektaş, "A Note on the Lifted Miller-Tucker Zemlin Sub Tour Elimination Constraints for the Capacitated Vehicle Routing Problem" European Journal of Operational Research, vol.158, pp.793-795, 2004.

[21] B.L. Golden, W.L. Stewart, "Empirical analysis of heuristics", in *The traveling salesman problem*, edited by Lawler et al., John Wiley & Sons, New York, 1985.

[22] R.H. Ballou, Business logistics / supply chain management, 5th ed., Pearson and Prentice Hall, 2004.

[23] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, "Network flows, theory algorithms, and applications," Prentice Hall, 1993.

[24] A. Yazıcı, A. Sipahioğlu, O. Parlaktuna, U. Gürel, "A Heuristic-Based Route Planning Approach for a Homogeneous Multi-robot Team", Proceedings of the 2006 IEEE International Symposium on Intelligent Control, Munich, Germany, pp.1237-1242, October 2006.

[25] O. Parlaktuna, A. Sipahioğlu, A. Yazıcı, U. Gürel, "Tsp Approach For Mobile Robot Dynamic Path Planning," The 1st International Conference on Control and Optimization with Industrial Application, Abstracts: pg.79, Baku (Azerbaijan), 2005.