CHAPTER 6

HYBRID METHODOLOGIES FOR BMRTA

In this chapter two hybrid methodologies have been proposed to solve the percentage idleness BMRTA model. The first methodology combines the saving matrix with the TSP Convhull algorithm and the second one amalgamates the angular method with the TSP Convhull algorithm. The first part of both the methods group (assigns) selected numbers of task with the robots and the second part determine the shortest path of robot travel. The saving matrix based method tries to balance perfectly tasks allocated to the robots whereas the angular based method tries to minimize the total travel path length. A comparative study is performed for a set of benchmark data to evaluate the strength and weakness of both the methods. The path balancing is achieved by minimizing the percentage idleness between the robots.

This chapter proposes two variants based on the saving matrix, originally known as the Clarke-Wright algorithm (Clarke and Wright 1964) and the angular technique to group the ‘N’ task of a multi-robot system into ‘K’ number of groups so that the number of tasks allocated to each robot is balanced and to minimizes the total path length. The path balancing in the angular method used by Yang et al (2007), which uses the polar coordinates of every node, calculates and sorts out in increasing polar order. The TSP Convhull algorithm is used to find the shortest path between any pair of nodes.
The rest of the chapter is organized as follows: Section 6.1 details the hybrid methodology proposed to solve the percentage idleness BMRTA model problem using, saving matrix technique and angular method. Section 6.2 illustrates calculations for the proposed methodology with examples. Section 6.3 compares the results obtained by the two proposed hybrid methodologies. Section 6.4 concludes with the advantage, limitations and flexibility of the proposed methods, and gives further direction for improvement.

6.1 METHODOLOGIES

The ‘N’ tasks must be partitioned into ‘K’ clusters equivalent to the number of robots using the saving matrix method and the angular method. Equal numbers of tasks i.e. (nearest integer of (N/K)) are allocated to each robot using the saving matrix as follows. Allocation is done based on the maximum cost/distance saving obtained using, the saving matrix to the first robot. The step is repeated until the required number of tasks is allocated to the first robot (equal to the nearest integer of “N/K”). The same procedure is repeated to the entire robot until the allocation of all the tasks is exhausted. Thus the main objective of the method is to balance the number of tasks allocated to individual robots and subsequently attempts to minimize the travel distance for each robot. For the angular method the tasks are sorted out in a decreasing order based on their angular positions until the required number of tasks is allocated to the first robot i.e. nearest integer of (N/K) from the total sorted angular position list. The same procedure is repeated to the entire robot until all the tasks are allocated to all the robots. The detailed explanations of the hybrid methodology are delineated in this section and the outline is shown in Figure 6.1.
Figure 6.1 The proposed hybrid methodologies for balanced task allocation and path minimization for multi-robot systems
6.1.1 Saving Matrix Method

6.1.1.1 Computation of the Distance Matrix

The distance matrix identifies the distance between every pair of location to be visited. The distance is used as a surrogate for the cost of traveling between the pair of locations. If the traveling costs between every pair of location are known, the costs can be used in place of distance. The distance \( \text{Dist} (A, B) \) is the distance between a point A with coordinates \((x_A, y_A)\) and a point B with coordinates \((x_B, y_B)\) is evaluated as follows using Equation (6.8).

\[
\text{Dist} (A, B) = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}
\]  

(6.1)

6.1.1.2 Saving Matrix calculation

The saving matrix represents the savings that accrue on consolidating two tasks on a single robot. Savings may be evaluated in terms of distance, time, or money. The saving can be calculated by the following formula given in Equation (6.2).

\[
S(x, y) = \text{Dist} (\text{OP}, x) + \text{Dist} (\text{OP}, y) - \text{Dist}(x, y)
\]  

(6.2)

From the formula it is noted that the maximum saving occurs when the distance between points \( x \) and \( y \) is minimum. Thus the selection of the saving matrix represents the savings that accrue on consolidating two tasks on a single robot. Savings may be evaluated in terms of distance, time, or money.

From the formula, It is noted that the maximum saving occurs when distance between the points \( x \) and \( y \) is found to be minimum. Thus the selection of the maximum saving will establish the minimum path length.
6.1.3 Assign Tasks to Robots

When assigning tasks to robots, the BMRTA attempts to maximize the savings. An iterative procedure is used to make this assignment. In this assignment the tasks are assigned to one temporary robot, for example if there is ‘N’ tasks and ‘K’ robots then the ratio of the nearest integer of (N/K) will determine the number of tasks allocated to each robot. The first groups of tasks say (nearest integer of (N/K)) are assigned to first robot. In the same manner remaining tasks are assigned to left behind robots.

6.1.2 Angular Method

The angular position ($\theta_i$) of task $i$ with coordinates $(x_i, y_i)$, is the angle made relative to the X axis by the line joining the task $i$ to the origin (Figure 6.2). The angular position of each task is obtained as the inverse tangent of the ratio of its ‘y’ coordinate to the ‘x’ coordinate

$$\theta_i = \tan^{-1}\left(\frac{y_i}{x_i}\right)$$

![Figure 6.2 Angular Position of Task](image)
For the angular method the tasks are sorted in the decreasing order of their angular positions and the required number of tasks are assigned to the first robot i.e. the nearest integer of \( N/K \) from the sorted out angular position list. The same procedure is repeated to the entire robot until all the tasks are allocated to all the robots.

6.1.3 TSP Convhull Algorithm

The TSP Convhull algorithm is used for routing between the assigned tasks of each robot after the assignment phase. It starts from the Convhull and it is possible to find the shortest path of length closest to the exact solution. It is necessary to add the left out points to the Convhull. The adding process must be efficient, so each iteration adds the points that cause minimum growth of the path length. An inbuilt matlab code for TSP Convhull algorithm within the main matlab program is used for sequencing and routing the tasks for each robot in a short span of time.

6.2 NUMERICAL ILLUSTRATIONS

For illustrating the proposed hybrid methodologies a data set is considered as shown in Table 6.1. The data set consists of 13 tasks and 4 robot. The depot point for all robots (origin) is taken as the first node of the data set similar to the condition of the input data for the VRP problem. Observations based on both the methodologies are shown in Tables 6.2 and 6.3 and the paths of robots are shown in Figures 6.3 and 6.4.
Table 6.1 Data sets for task coordinate

<table>
<thead>
<tr>
<th>Tasks</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>15</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>17</td>
<td>-2</td>
</tr>
<tr>
<td>9</td>
<td>7</td>
<td>-4</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>-6</td>
</tr>
<tr>
<td>11</td>
<td>15</td>
<td>-6</td>
</tr>
<tr>
<td>12</td>
<td>20</td>
<td>-7</td>
</tr>
<tr>
<td>13</td>
<td>7</td>
<td>-9</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>-15</td>
</tr>
</tbody>
</table>

Table 6.2 Observations from saving matrix based hybrid methodology

<table>
<thead>
<tr>
<th>Particulars</th>
<th>13 tasks and 4 robots</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Path length</td>
</tr>
<tr>
<td>Robot 1</td>
<td>48.912</td>
</tr>
<tr>
<td>Robot 2</td>
<td>50.972</td>
</tr>
<tr>
<td>Robot 3</td>
<td>57.706</td>
</tr>
<tr>
<td>Robot 4</td>
<td>54.437</td>
</tr>
<tr>
<td>Total distance</td>
<td>212.027</td>
</tr>
<tr>
<td>% idleness</td>
<td>15</td>
</tr>
</tbody>
</table>
Figure 6.3  Path for four robots using saving matrix based hybrid methodology

Table 6.3 Observations from angular method based hybrid methodology

<table>
<thead>
<tr>
<th>Particulars</th>
<th>13 tasks and 4 robots</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Path length</td>
</tr>
<tr>
<td>Robot 1</td>
<td>38.22</td>
</tr>
<tr>
<td>Robot 2</td>
<td>42.86</td>
</tr>
<tr>
<td>Robot 3</td>
<td>44.20</td>
</tr>
<tr>
<td>Robot 4</td>
<td>36.01</td>
</tr>
<tr>
<td>Total distance</td>
<td>161.29</td>
</tr>
<tr>
<td>% idleness</td>
<td>18.5</td>
</tr>
</tbody>
</table>
Figure 6.4  Path for four robots using angular method based hybrid methodology

Table 6.4 Comparison of saving / angular method for path and number of task balancing

<table>
<thead>
<tr>
<th>No of robots</th>
<th>Saving method</th>
<th>Angular method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Robot path length</td>
<td>Number of tasks allocated to each robot</td>
</tr>
<tr>
<td>4</td>
<td>48.91-50.97-57.71-54.44</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Total</td>
<td>212.03</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 6.4 shows that the total path length by the saving method is larger than the angular method path length. There is 31.45 % deviation in the total path length between the saving and the angular methods. So the angular method is suitable when compared with the saving method.
Table 6.5 Percentage deviation of saving / angular method results with minimum global optimal solution

<table>
<thead>
<tr>
<th>No of robots</th>
<th>IP solver (minimum)</th>
<th>% deviation from IP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Robot path length</td>
<td>Number of tasks allocated to each robot</td>
</tr>
<tr>
<td>4</td>
<td>84.40-16.12-13.66-15.62</td>
<td>10</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>13</td>
</tr>
</tbody>
</table>

From Table 6.5 the angular method has 24.26 % deviation form it minimum objective and the saving method have 63.94 % deviation form the minimum team objective value.

6.3 RESULTS AND DISCUSSION

The proposed saving matrix based hybrid methodologies approach for BMRTA with the objective to minimize the total distance travel by all robots for the fixed single depot cases and attempts to minimize the percentage idleness between the robots. Results listed in Table 6.4 show significant advantages for the proposed saving method in terms of minimizing the percentage idleness between the robots. The saving matrix based methodology gives better results in terms of balancing the path length between the robots. However in terms of the minimization of total path length saving matrix based methodology gives no expected results. Furthermore, Figure 6.3 clearly shows the saving matrix based methodology would lead to the collision of path which is not desired for multi-robot coordination. From
the results delineated in Table 6.4, the angular method based hybrid methodology is more suitable in terms of the minimization of the total path length, but in terms of minimizing the percentage idleness between the robot it shows slightly higher percentage i.e. a 3% increase. Nevertheless with this minimum 3% increase in percentage idleness there is a huge reduction in the total path length i.e. from 212.027 to 161.29. It is also observed from the Figure 6.4 there is no collision of path and that is highly required for the multi-robot coordination.

To evaluate the proposed methodology benchmark problem data set is obtained using the URL http://www.branchandcut.org/VRP. The program is coded in Matlab and run on a PC with Pentium dual core processor; the origin is taken as the first node for all robots similar to the condition of the input data for the VRP problem. However, the capacity constraint has not been considered.

**Table 6.6 Performance of saving matrix based hybrid methodology on the data set Eil22. Vrp.**

<table>
<thead>
<tr>
<th>Particulars</th>
<th>22 tasks 3 robots</th>
<th>22 tasks 2 robots</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Path length</td>
<td>No of Task</td>
</tr>
<tr>
<td>Robot 1</td>
<td>171.057</td>
<td>7</td>
</tr>
<tr>
<td>Robot 2</td>
<td>227.76</td>
<td>7</td>
</tr>
<tr>
<td>Robot 3</td>
<td>179.00</td>
<td>8</td>
</tr>
<tr>
<td>Total distance/task</td>
<td>577.817</td>
<td>22</td>
</tr>
<tr>
<td>% idleness</td>
<td>25</td>
<td></td>
</tr>
</tbody>
</table>
Table 6.7 Performance of angular position method based hybrid methodology on the data set Eil22. Vrp

<table>
<thead>
<tr>
<th>Particulars</th>
<th>22 tasks 3 robots</th>
<th>22 tasks 2 robots</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Path length</td>
<td>No of Task</td>
</tr>
<tr>
<td>Robot 1</td>
<td>131.079</td>
<td>7</td>
</tr>
<tr>
<td>Robot 2</td>
<td>160.014</td>
<td>7</td>
</tr>
<tr>
<td>Robot 3</td>
<td>138.472</td>
<td>8</td>
</tr>
<tr>
<td>Total distance/task</td>
<td>429.565</td>
<td>22</td>
</tr>
<tr>
<td>% idleness</td>
<td>18</td>
<td></td>
</tr>
</tbody>
</table>

6.4 SUMMARY

This chapter proposes two varieties of hybrid based methodologies to assign equal number of tasks to robots and simultaneously tries to minimize the total travel length. In one aspect this thesis studies how to coordinate a team of mobile robots to visit several tasks in a known terrain. The proposed hybrid methodologies have two major assignments: allotment of tasks to each robot and sequencing of robots to the assigned tasks. The assignment and sequencing are done with two objectives: to minimize the total distance and to minimize the percentage idleness in robots. In this chapter we have proposed two methods for solving the BMRTA. The saving matrix based methodology is found to be superior in the minimization of the percentage idleness between the robots. However it is meager in minimizing the total travel distance and the path of the robots in some instances intersects with each other and that is not suitable for coordinating the environment between robots. The angular method based hybrid methodology, although it minimizes the percentage idleness between the robots, is found to be slightly higher than the saving matrix based hybrid methodology, whereas the minimizing the total path
length which is directly proportional to the cost of travel is reduced to large extent. Besides, the angular method based hybrid methodology ensures a collision free path for each robot, and that is highly desirable.

The angular method is superior to the saving method since in angular method the task locations are arranged based on the polar angular positions. Then the task locations are clustered by sequentially assigns the task locations to the current robot by considering the order of polar angle with respect to the depot. When the require path length balancing and the number of task balancing is achieved the remains task location are assigned to the remaining robots. Each cluster is solved as separate TSP by Convhull Algorithm to find the route for each robot. Since the task locations are clustered based on their angular positions the routes of the robots never cross each other.

The results listed in Table 6.6 and Table 6.7 highly emphasize this conclusion. In future the same work could be extended to multi depot cases. With some modification in the methodology the same model could also be extended to non identical tasks such as tasks with variable energy level. This chapter uses the saving and angular methods for solving the percentage idleness BMRTA. Out of the two methods, the angular method performs well when compared to the saving method. In the previous chapters the time and space simultaneously taken under static condition to solve the balancing ratio BMRTA problem, idle cost BMRTA problem, and percentage idleness BMRTA problem. In the next chapter the time, space, and task priority are simultaneously considered which is said to be under dynamic condition for allocating tasks to the robots.