Faculty of Engineering and Materials Science
German University in Cairo

Utilization of a Market-Based Approach for Solving the Multi-Robot Task Allocation Problem

A thesis submitted in partial fulfilment of the requirements for the degree of Bachelor of Science in Mechatronics Engineering

By

Mahmoud Mostafa Elbarawy

Supervised by

Prof. Elsayed Ibrahim Imam Morgan

Mechatronics Department
Engineering and Materials Science Faculty
German University in Cairo

This is to certify that:

(i) the thesis comprises only my original work toward the Bachelor Degree of Science (B.Sc.) at the German University in Cairo (GUC),

(ii) due acknowledgment has been made in the text to all other material used

---

Mahmoud Mostafa Elbarawy
February 26, 2022

Acknowledgments

I would like to thank God for having such parents .they always supports me.also thanks God for giving me the opportunity to learn from our beloved Dr's and TA's. special thanks for Ali Bahaa for his efforts with me in my study and for tolerating me during this journey i would like to thank my dear Friends Wessam Shoman, Youssef Ashraf and Motaz for being a part of this team.I would like to thank all MRS lab family.I would like to thank Dr. Omar M.Shehata for always being a part of motivating the team and always giving a helpful feedback about the work.I Also Would like to thank our beloved Prof. Elsayed Ibrahim Imam Morgan.

Abstract

It is a well known fact that Multi-Robot Systems have been used all over the world in Warehouse management as they are smart,cheap and more Reliable than manpower . For example Amazon company is using $KivaSystems$ which consists of mobile robots (kiva-robots) for order fulfillment in the warehouse.The power of such an systems does not lie in the capabilities of the single Robot but in the capabilities of whole coordination of the robotic system . One of the most important and complex problems that this systems face is $TaskAllocationProblemforMulti-Robotsystems$.that is aims to answer this question

"Which Robot is responsible to handle which task such that the overall system performance of the team of Robots is optimized ?"

As a fact researchers all over the world tended to make smart algorithms for solving that type of tasks for the system of mobile robots this problem is a well known problem in the scientific society which is Task Allocation Problem for Multi-Robot systems. In this thesis we propose a Market-Based approach ($CusterCombinatorialAuction$) for solving such a problem also we introduce a new approach which is $IterativeMarket$ solution. The results of the work was really powerful for solving such a problem for the system also we demonstrate our work using a path planing technique and there were simulated results for our work using Pure Pursuit Controller to create robotic systems and simulate it to avoid obstacles in the warehouse. As a conclusion the algorithms performance was really powerful and manage to solve the $TaskAllocationProblem$ efficiently in big scale . we made a comparative study between $ClusterCombinatorialAuction$ and $IterativeMarket$,the performance of $ClusterCombinatorialAuction$ in computational time Outperformed $IterativeMarket$.however, $IterativeMarket$ gives better Cost.

Contents

List of Abbreviations

MRSs	Multi-Robot Systems
MRTAs	Multi-Robot Task allocation
OAPs	Optimal Assignment Problem
AEPs	ALLIANCE Efficiency Problem
MTSPs	Multiple Traveling Salesman Problem
TSPs	Traveling Salesman Problem
C.C.A	Cluster Combinatorial Auction
IT	Iterative market

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Robotic Systems: Multi-robot systems (MRS)

Multi-robot systems (MRS) are a group of robots that are designed aiming to perform some behave like disposal of land mines, search, rescue,and map building and other things . By this behavior, some goals that are impossible for a single robot to achieve become feasible and attainable. individual robots have different capabilities. These differences may be in hardware; for instance, robots can be different in terms of their size, mobility (driving versus flying), sensors, or computational power. Robots can also be different in software, such that they have different available behaviors or perceptual algorithms. Robots in a team often have different sensors. To complete a task, the sensors distributed across a robot team should be brought to where they are most needed. Two domains in which robots with sensors may need to be recruited to achieve team objectives are error handling and distributed sensing. In terms of error handling, a robot may experience a sensor failure or become stuck and require another robots external viewpoint for diagnosis and recovery . For distributed sensing, an autonomous aerial vehicle may detect a suspicious object on the ground and employ a ground vehicle to investigate more closely One of the reasons that the topic has become more popular is the various benefits of MRS compared to single robot systems.are Resolving task complexity: some tasks may be quite complex for a single robot to do or even it might be impossible. Increasing the performance: task will take no time in multi robot system compared to single robot system Increasing reliability: increasing the system reliability when having multiple robots doing a task and one fails, others could still do the job.

1.2 Robotic Systems: Single-robot Versus Multi-robot

A single-robot system contains only one individual robot that is able to model itself, the environment and their interaction [1]. Several individual robots are well known, such as RHINO [2], ASIMO [3], MER-A [4], BigDog [5], NAO [22] and PR2 [6]. The robot in a single-robot system is often designed to deal with a task on its own account. Such robots are usually integrated with multiple sensors, which themselves need a complex mechanism and an advanced intelligent control system. Although a single-robot system give have a relatively strong performance, some tasks may be inherently too complex or even impossible for it to perform, such as spatially separate tasks. For example, Dudek et al. [7] gave an example of a missile launch task that requires some sort of synchronization: there are two keys separated by a large distance in space that need to be activated simultaneously. Hence, an inherent restriction to the single-robot system is that it is spatially limited.

A MRS contains more than one individual robot, whether group homogeneous or heterogeneous. Using a MRS can have several potential advantages over a single-robot system:

A MRS has a better spatial distribution.

A MRS can achieve better overall system performance. The performance metrics could be the total time required to complete a task [23] or the energy consumption of the robots [8].

A MRS introduces robustness that can benefit from data fusion and information sharing among the robots, and fault-tolerance that can benefit from information redundancy. For example, multiple robots can localize themselves more efficiently if they exchange information about their position whenever they sense each other [9] [10] [11] .

A MRS can have a lower cost. Using a number of simple robots can be simpler (to program), cheaper (to build) than using a single powerful robot (that is complex and expensive) to accomplish a task.

A MRS can exhibit better system reliability, flexibility, scalability [12] and versatility. Robots with diverse abilities can be combined together to deal with complex task, and one or several robots may fail without affecting the task completion.

1.3 Multi-Robot Environment

MRS can be classified according to their collective behaviour in the environment. MRS can either have a cooperative behaviour or a competitive behaviour [13]. 1-Cooperative behavior means that robot work and coordinate with each other in order to achieve a certain goal or complete a certain task. Examples of cooperative behavior are area exploration and search and rescue missions. 2-Competitive behaviour on the other side is the opposite of cooperative where robots com-pete against each other to accomplish a self desired goal.An example of competitive be-havior is competition like robocup.

1.4 Robot Locomotion

To make the robots move in the environment it need a locomotion mechanism . Locomotion means that the robots utilizes its actuators to exert force on the environment to move on it . there are several Locomotion types that will be discussed in this section

Wheeled Robot: Wheeled robots are robots that use wheels in their movements . There are many types of wheeled robots according to their number of wheels which can start from a single wheel robot.The most common is the 3 or 4 wheel robots where the 3 wheel robot uses 2 wheels and a caster wheel. Their disadvantage is that they may lose traction

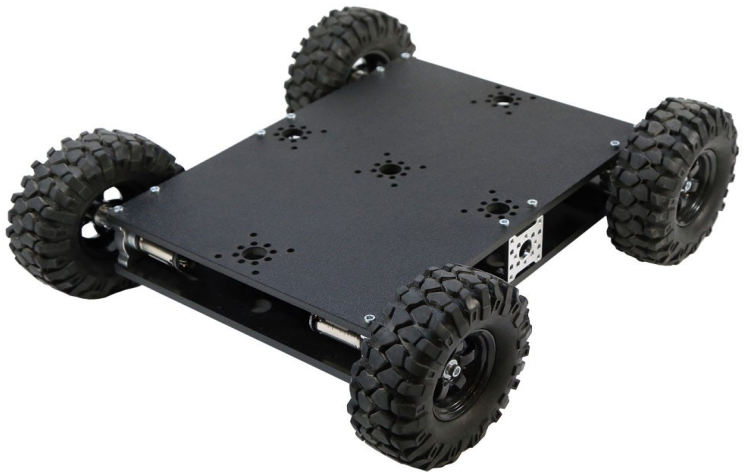

Figure 1.1: Four wheeled robot

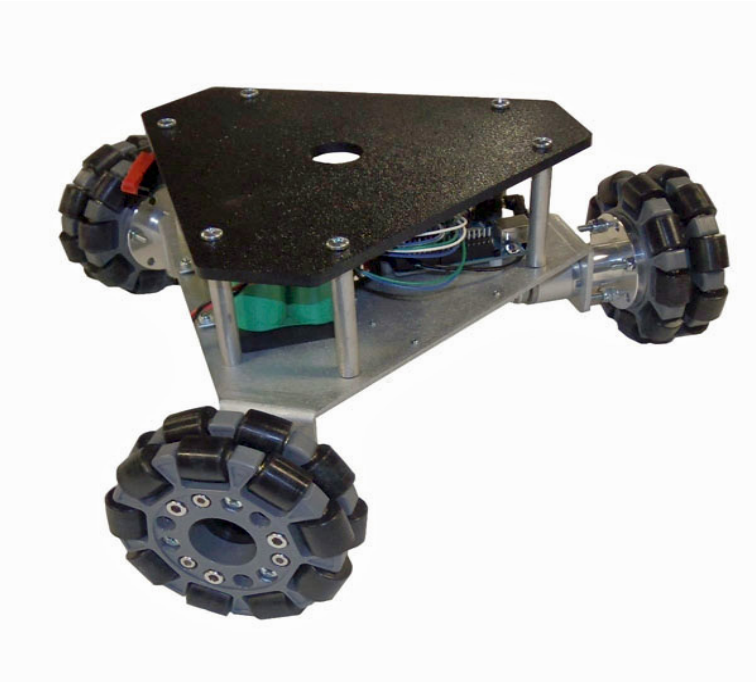

Figure 1.2: Three wheeled robot

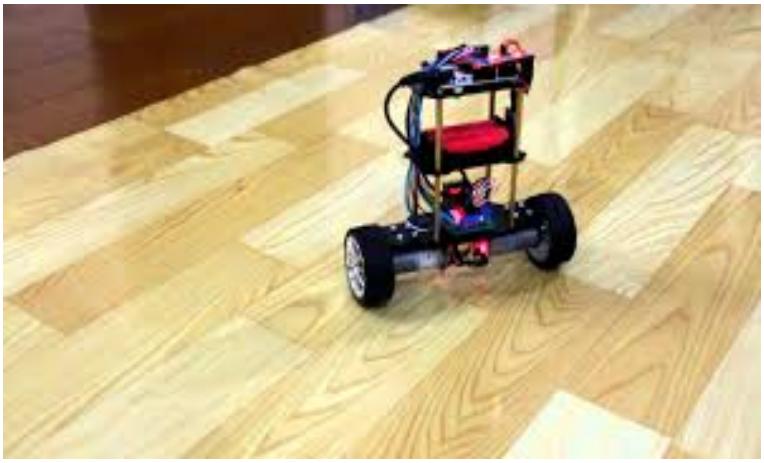

Figure 1.3: two wheeled robot

Tracked Robot: Tracked robots uses tracks like the ones used in tanks. Their driving mechanism is similar to the wheeled robots. Their main advantage is that they decrease the slip and give more traction for the robot especially when used on smooth surfaces such as sand. Their disadvantage is the wear of the tracks due to the side way forces and also its increased complexity

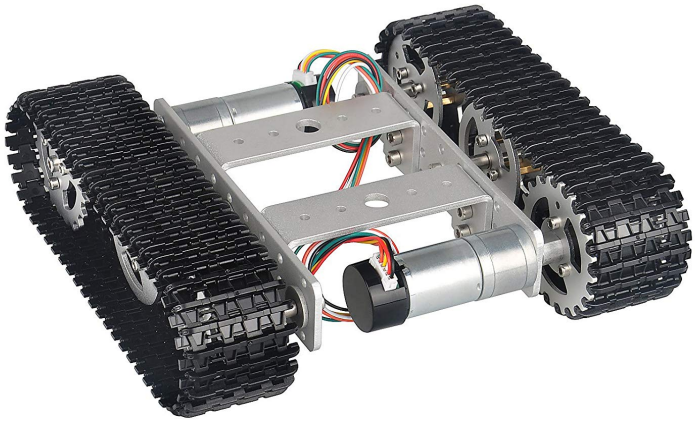

Figure 1.4: two wheeled robot

Stationary Robot: Stationary robots are robots that have a fixed base while the robot actuators (end effector) are moving.There are several types of stationary robots such as kuka robot, SCARA robot and puma robot.

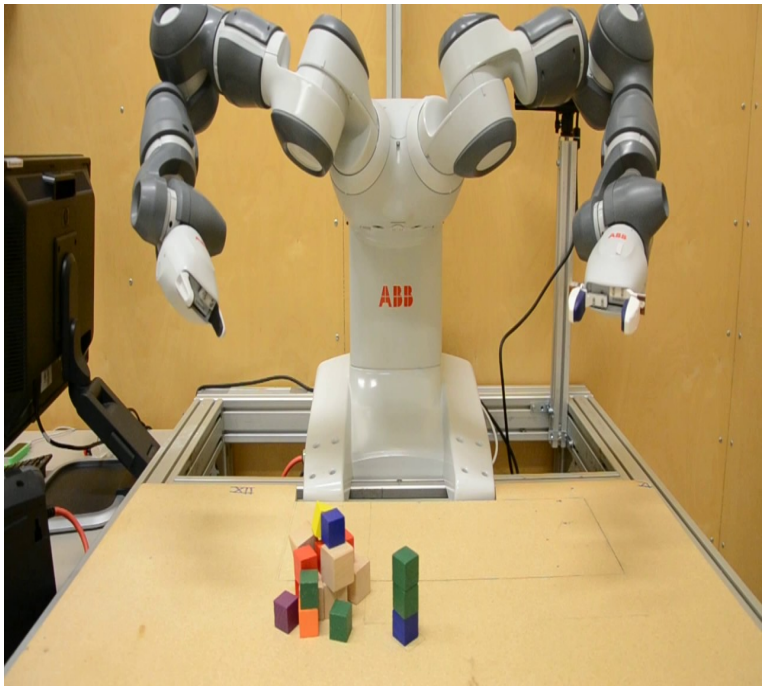

Figure 1.5: ABB Stationary Robot

Legged Robot: Legged robots are robots that utilization legs for their movement. This sort of movement is more refined than wheeled. this kind of movement is favored over wheeled robots in harsh landscapes. As in wheeled robots.

Figure 1.6: Two Legged Robot

Swarm Robots: The use of several independent robots is swarm robotics to carry out a job. Robot swarms decentralizedly co-ordinate the behavior of many comparatively easy robots. The growth of cooperative artificial intelligence (AI) performs an significant part in Swarm Robotics. Current applications of robot swarms include research and salvation, precise farming, production chain management (SCM) and army recognition. Swarm technology seeks to build on the methods used by social animals, such as insects, to accomplish complicated duties that go beyond the capacity of any individual. For instance, Swarm Robotics scientists could research how ants label paths to geographical places by pheromones.The investigators could then reproduce the same conduct with robots using bee filters.

Figure 1.7: Swarm Robots

Soft robots: Soft robots are produced of smooth and flexible components and give distinctive possibilities in fields where standard stiff robots are not feasible, e.g. for the distribution of drugs or non-invasive surgery, such as aids, prothesis or artificial bodies. The evolving sector of soft computing includes material researchers, mechanical and biomedical technicians, and smart systems researchers. Many smooth robotic devices are designed with natural inspiration. This problem includes contributions by researchers in main fields of this multi-disciplinary sector, with papers on biomicrobots, the three-dimensional printing and origami folding of smooth robots, the use of smooth machines and biomedical apps. In addition, problems, conceptual layout and milestones are addressed for clinical translation.

Figure 1.8: Soft robots

1.5 MRS challenging problems

In order to apply (MRS) in real word applications there are a challenging problems like Task allocation, Group formation, Cooperative object detection and tracking, Communication relaying and self-organization.

1.6 Problem Statement

Task allocation in Multi-Robot Systems (MRSs)is one of the a very hard problem that need to be solved in an optimized way there are two main approaches for solving such a problem

- optimization approach
- Market-based approach

This thesis is focus in solving Task allocation problem using Market-based approaches and solving a problem of path planing for the system.

1.7 Organization of Thesis

This thesis is organized in such a way that chapter one is introduction about the field of our work and the problem that we will tackle in our study.

Second chapter is the literature review that shows the results of previous studies that were done in the same field and the methods that the researchers used and which of them are the successful methods to complete their work and get up with a better result at the end.

In the third chapter which is the methodology, I will demonstrate my work and techniques I used to perform task allocation.

The fifth chapter which is the experiments and results, I will show the results validation of the implemented algorithms for Task allocation .

Sixth chapter which is conclusion will conclude this study starting from the aim of the study passing by the approach of working till the results of our experiments.

Last but not least, seventh chapter is the future work will be about the recommendations about the forthcoming work to get a better results

Chapter 2

Literature Review

2.1 Multi-Robot Task allocation (MRTAs) problem

MRTAs problem addresses the question of finding the task-to-robot assignments so as to accomplish the general framework objectives this issue can be separated into to two sub problems first how a set of tasks is assigned to a set of robots. Second, how the behavior of the robot team is coordinated in order to achieve the cooperative tasks efficiently and reliably To solve this problem we need first to model it according to [14] there are four main modeling technique for this problem.

2.1.1 Model 1 Discrete Fair Division

The MRTAs issue can be viewed for instance of a Fair Division Problem [15]. Given a set of N robots (r1,r2, rNr1,r2, rN) and a set of errands S. It is required to separate S into N shares (s1,s2, sNs1,s2, sN) so every robot gets a decent amount of S. A decent amount is an offer that, in the supposition of the robot accepting it, is worth 1/N of the all out estimation of S Fair division problems can be characterized relying upon the idea of the arrangement of offers S into two sorts: Indivisible tasks, such that each item should be given entirely to a single robot. Divisible tasks, which are often modeled as a subset of a real space. Additionally, the set to be divided may be homogeneous, or heterogeneous. Thus the set of the dividable tasks should be given to homogeneous or heterogeneous robot team Two distinct plans have been accounted for in the literature to manage discrete fair division issues. The primary plan is known as the technique for sealed bid at which every bidder presents a secret sealed bid. The bid are kept private until the end of the biding time frame. After the Auction shuts, the offers are opened by the salesperson and the bartering winner is resolved. The winner will be the one with the most elevated offer cost The technique of marker is the second arrangement for a separate

reasonable division. The assignments can be organized linearly with this technique. This could be the case if a lot of small tasks have to be shared. The N robots accessible show their view of a reasonable division by putting markers N1N1 and agree to recognize any portion of the assignments between any couple of their successive markers. The following phase is to locate the leftmost marker in the following. The proprietor of this sign gets the first section (from the left end to the target) and all other signs of that machine are withdrawn. This stage will be repeated until all robots have what they think is sensible. In reasonable separation, MRTA is set in [16]. This approach deals only with the task of one heterogeneous robot globally.

2.1.2 Model 2 Optimal Assignment Problem (OAPs)

An instance of an ideal allocation issue can be considered as MRTAs. When a collection of robots R takes place in this sort of problem, [17], a number of assignments T are to increase the benefit W(rt) of assigning robot r to the assignment t. By adding virtual tasks or zero-profit robots, R and T are presumed to be n with the same dimension as R= r1,r2,...rN1,r2... rN, and T= t1,t2,... tNt1,t2,... tN. Such a correspondence between R and T is called optimal. The goal is to assign the set of robots R to the taskset T in relation to the allocation of multi robots to maximize the profit [18]

2.1.3 Model 3 ALLIANCE Efficiency Problem (AEPs)

ALLIANCE Efficiency Problem is a calculation of the mono-target enhancement used first to resolve [19] NP issue. It has been combined to deal with all issues related to mono-target simplification and used to address issues in artificial intelligence and robotics. In this algorithm, several clans, with certain abilities, attempt to win over an atmosphere which provides the required funds for their two characteristics: the abilities and resources required for their survival.

2.1.4 Model 4 Multiple Traveling Salesman Problem(MTSPs)

In the MTSPs there are m salesmen travelling a set of n cities, and each salesman is defined to start and end at the same city. In simulations, each city must be visited exactly once by only one salesman and its objective is to minimum total distances travelled by all the salesmen. As the number of salesmen is not fixed, The cost could be distance or time. A comprehensive study of 32 formulations for the multiple traveling salesman problem is investigated in [20] considering their relative performances A number of variations of the original MTSPs were introduced by different researchers to accommodate the MTSPs to their problems. These variations included the following [21]: 1-Salesmen starting node: all the salesmen may start from a single depot

node and then all of them must return back to the same node or every salesman can start from a certain node, and thus each salesman must return back to his starting node 2-Number of salesmen: the number of salesmen used in different applications varies according to the type and requirements of the application itself. In some applications, the number of salesmen is dynamic such that after each iteration the number of salesman may or may not change. 3-City time frame: in some applications the task of the salesman is not only to visit the city, but also to stay in the city for a certain time in order to move to the next city. 4-Fair division of salesmen: another variation of the general MTSPs is the addition of constraints that specify the maximum number of cities or the maximum distance that can be traveled by a single salesman. This variation can be used in applications that are concerned with the fair division of the available resources (salesmen) The MRTAs problem is modeled as a multi-traveling salesman problem considering that robots play the roles of the salesmen and the tasks are the same as cities.

2.2 Heuristic techniques for solving Traveling Salesman Problem (TSPs)

In the 19th century, the problem of finding a Hamiltonian cycle on a graph was mathematically formulated by Hamilton [22]. The Hamiltonian cycle is a well-known reduction of the TSP. Some of the first techniques to be used to solve the TSP include Branch and Bound [23], the Christofides algorithm [24] and the 2-Opt [25].

2.3 Market based approaches in solving MTSP

The market-based solution to a number of problems has been widely used, including the MTSP. In [26], a clustering technique with an auction process was employed. The goals of all robots are to minimize the distance and to balance the workload between the robots equally. The first step is to break down N tasks into n groups, to minimize the distance in each cluster. The cost of visiting n clusters for each robot is calculated. Finally, each cluster is assigned to the lowest offer robot in the auction phase.We noticed that the algorithm is quite complex because all possible combinations are considered for the assignment of clusters to robots. This means that only a very small number of clusters can apply the approach.The total cost used for the assignment is equal to the sum of the cost of visiting the tasks in the cluster and the idle cost (i.e., sum of the difference in cost of travel between any two robots). Two scenarios were considered: one with two clusters and the other with three clusters. We noticed that the scenarios that were used are not sufficient to prove the effectiveness of the algorithm.

A new market-based clustering strategy (CM-MTSP) was recently introduced [27] to fix the multi-target MTSP. The algorithm is that objectives are grouped into groups and then allocated to the finest robot for each cluster. The authors assumed in that work that the number of clusters was the same as the number of robots. The findings of the study show that the CM-MTSP balances contradictory goals and lowers implementation time relative to a greedy market alternative.

In [28], a clustering algorithm was suggested to minimize both the total range covered by each machine as well as the amount of the range covered by all the machines. The algorithm goes like this. First, a list of assigned duties is presumed to be available to all robots. If a robot hits the job place, it gives a message to the other robots so that an auction can be carried out. The robot re-plans its route and proceeds to the next assignment after all auction bids are finished.If a robot receives an auction message, a group of new clusters is established to perform its assigned tasks, and the auction process for the newly created clusters begins, except for the cluster that contains its initial task currently. If an auction signal is given to a cluster by the robot, this cluster is bidden. Finally, the cluster wins the robot with the highest offer. Compared to the final task, the assessment of results showed the proportion of enhancement in the original task.

MTSP was solved through market-based approaches. The authors suggested a new approach called movement and improvement in [29]. The approach includes four steps: initial allocation targets, tour building, removal of conflicting objectives and improvement of solutions. Two metres: the total cost and the maximum cost were used to measure the efficiency of the algorithm. The results showed the superiority of movement in comparison with centralized GA and improved an algorithm.

The solution proposed in [30] includes four phases: auction of the market, trade between agents, the switch of agents and the abandonment of agents. The quality of the solution, the number of iterations necessary for the solution, and the execution time were considered for 3 performance criteria. The approach has been shown to produce better solutions than other suboptimal solutions.

The market-based strategy involves an auction for the goals. The highest goal is won by all robots. More precisely, each robot sends an offer to the central machine, selecting the best objective (i.e. the target which has the lowest local cost). Local expenses are described as the weighted amount of the objective robot feature expenses. For instance, the robotR1 uses the LinKernighan heuristic TSP solver[31] to offer for Target T1. For instance it will use its trip price including T1. The offer includes the chosen destination and the associated expenses for each purpose feature. Note that, regardless of the others, each robot offers.More exactly, the goal can vary from robot to robot. For instance, R1 offers for T1, while R2 concurrently offers for T2.Once the various robots have offers received, the main computer calculates the overall price of each respective tender and then sets its respective robot to its highest goal. The greatest

goal is the minimum worldwide price target. In contrast to local costs, the total cost takes into account all tour costs, such as the amount of the whole tour length, the maximum tour length and the deviation rate for the tour length. Robots proceed the bidding cycle until all objectives have been allocated

2.4 Auctions

Several various processes of auctioning have been used in multi-robot teams for job distribution. At a sequential auction, in order chosen by the auctioneer, the auctioneer sells a sequence of items, one item at a time. After map updates, Simmons et al. [32] have robots offering to visit border stations. Wei et al. [33] are offering robots to find unvisited target points and to carry objectives towards an objective. Rekleitis et al. are using a series of one-round auctions.[34] by Vail and Veloso [35] for assigning positions in robot tennis, and by Pippin and Christensen [36] for assigning the target detection duties for separating a Boustrophedon multi-robot coverage sweep. Vail and Veloso [35] have a one-round auction series circulated to exchange data on robots, and each auction is calculated on all robots individually.

MURDOCH utilizes a comparable series of one-around auctions to assign a wide range of duties including boxing, centuries ' task and the monitoring of objects (Mataric and Sukhatme [37], Gerkey and Mataric [38]. Nanjanath and Gini [39] use several simultaneous sequence auctions and replay auctions during work performance. Guerrero and Oliver [40] offer robots to participate in a range of multi-robot duties, and when a robot leaves its job an auction is made when needed.

The only type of auction guaranteed to find an excellent solution is to combine auctures in the absence of independent item valuation. Every agent submits an offer for each possible subset of items and the auctioneer awards the agents with a set of items, so as to ensure the best overall allocation for agents. Combination auction is difficult to fix and it is exponentially complicated in Sandholm [41].Combinatory auctions for Hunsberger and Grosz [42] multi-robot tasks are used. In particular, a study of combination of auction methods and apps is conducted in Parkes and Ungar [43] and Andersson et al. [44] and Sandholm [45] and De Vries and Vohra [46]. In Andersson et al. [47], we present the blended integral programming method that we use to achieve the minimum summed route distance goal. An open iterative combinatorial auction in Parkes and Hungary [48] has been shown to approach optimism as the minimum offer increase approaches 0, with fair tender.

2.5 Proposition

This thesis focus in finding the answer of

"Which Robot is responsible to handle which task such that the overall system performance of the team of Robots is optimized ?"

using a Market approach to solve such a problem which is **Task allocation** our proposal approach is to use innovative market auction type which is Cluster Combinatorial Auction (C.C.A) Cluster Combinatorial Auction with a 2-opt heuristics solver for TSPs and proposed a new Iterative market (IT) solution for such a problem. A comparative study will be done between the two Algorithms to find which is better
A path planning technique was also implemented in this study and there is a obstacle avoidance in this path planing technique.

Chapter 3

Methodology

3.1 Market Methodology

According to [49] the main goal in free-markets is the maximization of the overall system profit. By making each participant in the market tries to maximize its profit, as a result of this, the overall profit for the system is expected to increase. The idea of the market-driven method for multi-robot teams is based on the interaction of the robots among themselves . In general, there is a main goal of the team . Some entity outside of the team is assumed to offer a payoff for that goal. The main goal of the system is decomposed into smaller tasks and an auction is performed for each of these tasks. In each auction, the participant robots (who are able to communicate among themselves) calculate their estimated cost for accomplishing that task and offer a price to the auctioneer. At the end of the auction, the bidder with the lowest offered price will be given the right of execution of the task and receives its income on behalf of the auctioneer. There are many possible actions that can be taken. A robot may open another auction for selling a task that it won from another auction, two or more robots may cooperatively work and get a task which is hard to accomplish by a single robot, or for a heterogeneous system, robots with different sensors/actuators may cooperate by resource sharing (for example, a small robot with a camera may guide a large robot without a vision system for carrying a heavy load). In order to implement the strategy, a cost function is defined for mapping a set of resources (required energy, required time, etc.) to a real number and the net profit is calculated by subtracting the estimated cost for accomplishing the task from the income of the task.

3.2 Auctions

Generally, any protocol that allows agents to indicate their interest in one or more resources or tasks is considered an auction moreover, auctions provide a general theoretical structure for understanding resource allocation among self-interested agents. Since auctions are simply mechanisms for allocating goods, there are various types of auction that can achieve this goal.

form literature review we found that Combinatorial Auctions is a really power full technique that we may implement but we introduce some modifications for it.

3.3 Combinatorial Auction

Combinatorial Auction:bids on a set of items (Tasks) and there is a winner where the winner is one that utilizes the objective function. The advantage of this technique according to [50] is that it runs only once as the bidding is on a group of tasks rather than task by task. .

3.4 Implemented Model

To implement the Market approach we first need to choose a modeling technique for the problem in this thesis we used Multiple Traveling Salesman Problem MTSPs as our model At start we need problem parameters each Agent (robot) have (ID,Velocity,energy level,max wight to left,and position) Each Task had (ID,position,and nature). at the beginning we need a model symmetric (Distance) matrix which is the distance between each task and anther and each agent and anther assuming that there is no obstacles in the environment then we implement a simple Auction

3.5 Implemented Algorithm 1

At the algorithm of simple market we just bid on each task one by one and select the winner form the robots then its position is updated so on the next bid the position will not the same the biding at this algorithm means calculating the distance between the robot and the task and the nearest robot will win the task and so on.

Algorithm 1 Simple Auction

INPUT1: list of available tasks($available - tasks$)
INPUT2: list of available Robots ($available - Agents$)

OUTPUT1: list of assign Tasks to robot ($cities - winner$)

1: **for each** task $T \in (available - tasks)$ **do**
2: **for each** agent $R \in (available - agents)$ **do**
3: bidingList $\leftarrow$ bides(T, R)
4: **end for**
5: min(minAgent, biddingList) $\leftarrow T$
6: updatePosition(minAgent)
7: **end for**

3.5.1 Clustering Technique

Clustering is Grouping we used k-means clustering technique an implemented function on MATLAB and we get the output of such function of both centroid of the cluster and cluster data set points.

3.5.2 k-Means

The k-means algorithm is used to partition a given data into k clusters. The algorithm starts with a random set of k center-points (μ). During each update step, all points(cities) x are assigned to their nearest center-point.
Afterwards, the center-points are re-positioned by calculating the mean of the assigned point to the respective center-points.The update process continue until all the K centroid's move a small tolerance compared to the previous iteration .

K-means ' s main problem is dependency on the initially chosen centroids. The centroids could end up in splitting common data points while other, separated points get grouped together if some of the centroids are more attracted by outliers. This points will get pulled to the same group of data points. to solve such a problem we run iterations on k-means function

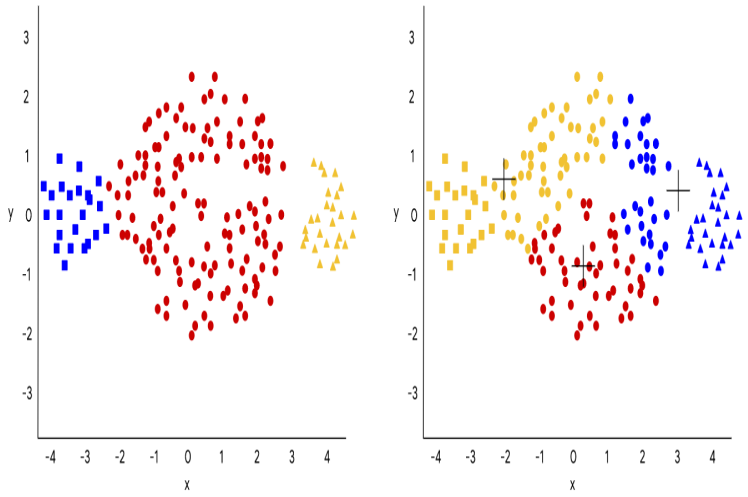

Figure 3.1: K-means

3.5.3 Cluster Combinatorial Auction On (k-Centroid's)

At this section i will demonstrate how we used k-means in combinatorial auction.
first we Give the Algorithm its inputs as it's mentioned in the pseudo code (Tasks position matrix, Robot position matrix ,and Distance matrix,) then the Algorithm creates all Tasks and Robots withe the desired Attributes if needed to change some attributes like Energy level and Task nature it is doable however for simplicity we just take the essential Attributes to start our Model and start the Auction.

Then we call K-means clustering Technique as we rived form literature that Combinatorial Auction is the type of Auctions were we bid not only on one Good but a set of Goods so to create this set of Goods we group them by K-means.

After that we get form k-means the Groups($clusterindx$) of Tasks and the position of each centroid of k-means($Clusterspos$)

Once we get the centriod's position we start the Auction we make each robot bid on the centroid of each cluster At this Algorithm centriod's of each Group represent the Goods of the Auction. The $Bidding - process$ at this Algorithm will be demonstrated in the next section .After $Bidding - Process$ finished the winner is determined with the lowest $Cost$ then they are assigned to there Group.

Then we Solve each agent with it's cluster As a single Travel Sales Man problem TSPs we get an already implemented TSPs Solver[51] but we made some modifications on it ; will be illustrated in TSP section.

Algorithm 2 Algorithm Cluster Combinatorial Auction

INPUT1: n tasks, $n \times 2$ tasksLocation
INPUT2: m agents, $m \times 2$ agentsLocation

```
1: if m > n then
2:     K ← n
3: else
4:     K ← m
5: end if
6: clusters ← apply K-means(K, tasksLocation+agentsLocation)
7: BiddingProcessOnCentroid(clusters, agents)
8: for each agent do
9:     apply 2-opt(agent)
10: end for
```

3.5.4 Bidding Process

Bidding process On the previous algorithm was in the centroid of clusters as mentioned in the Pseudo code and we get the min cost for each cluster by the robots then we pop from available robots list and available cluster list.

Algorithm 3 Bidding ON Cluster Centroid

INPUT1: clusters
INPUT2: Agents

OUTPUT1: updated agents

```
1: for each cluster do
2:     for each agent ∈ available agents do
3:         BiddingList ← cost(cluster, agent)
4:     end for
5:     min(winnerAgent, BiddingList) ← cluster
6:     remove(winnerAgent, available agents)
7: end for
```

3.5.5 Travel Sales Man TSPs Solver

According to [51] $TSPSEARCH$ Heuristic method for Traveling Salesman Problem TSPs,Generate a nearest neighbour tours are from randomly selected starting points. Each tour is improved by 2-opt heuristics and the best result is selected.

EXCHANGE2 Improve tour by 2-opt heuristics (pairwise exchange of edges).The basic operation is to exchange the edge pair (a b,c d) with the pair (a c,b d). The algorithm examines all possible edge pairs in the tour and applies the best exchange. This procedure continues as long as the tour length decreases. The resulting tour is called 2-optimal.

The modification done on this algorithm are: we propose to start the Tour from the robot as it was randomly chosen from the cluster point the other modification was the $cost$,$TSPSEARCH$ Heuristic method was solving a closed Tour cost which means the start and the end of the robot is the same however it is not a constrain in the real life application the robots may start from a location and end up in anther different one so we mad the cost only for traveled cities and the order is not a closed one.

3.5.6 Cluster Combinatorial Auction (TSP) on Bidding

This algorithm is the same as Algorithm 1 however we modified Algorithm 1 bidding process by tanking the biding not only on the centroid's of K-means but also the Tour it's self was taken in consideration from the beginning of bidding process it gives a better solution and made the code more efficient and more realistic

Algorithm 4 Algorithm Cluster Combinatorial Auction TSP on Bidding

INPUT1: n tasks, $n \times 2$ tasksLocation
INPUT2: m agents, $m \times 2$ agentsLocation
OUTPUT3: Task Allocation
OUTPUT4: cost

1: **if** m $>$ n **then**
2: K $\leftarrow$ n
3: **else**
4: K $\leftarrow$ m
5: **end if**
6: clusters $\leftarrow$ **apply** K-means(K, tasksLocation+agentsLocation)
7: BiddingOnAllTour(clusters, agents)

3.5.7 Bidding In All Tour

Algorithm 5 Bidding ON All Tour

INPUT1: clusters
INPUT2: agents

OUTPUT1: updated agents

```
1: for each cluster do
2:    for each agent ∈ available agents do
3:       apply BiddingList ← 2-opt(cluster+agent)
4:    end for
5:    min(winnerAgent, BiddingList) ← cluster
6:    remove(winnerAgent, available agents)
7: end for
```

3.5.8 Iterative Market

Iterative Market is a really power full technique that we porpoised and implemented .the idea in iterative market is simple as mention before in k-means section that the disadvantage of k-means is the dependency of centriod's on its initial random position the centroids could end up in splitting common data points while other, separated points get grouped together if some of the centroids are more attracted by outliers. This points will get pulled to the same group of data points.So to over come this problem of Grouping technique we made an iteration on the(2-Cluster Combinatorial Auction (TSP) on Bidding) for certain number and we comber the over all cost of the market with the previous one if it gives better solution keep it,if not $ignore$ and continue iteration this technique is power full on large scale for a large number of tasks and robots it gives a stunning solution.

Algorithm 6 Iterative Cluster Combinatorial Auction

INPUT1: n tasks, $n \times 2$ tasksLocation
INPUT2: m agents, $m \times 2$ agentsLocation

OUTPUT1: AuctionOpt

```
I = n×m
Temp=inf
for I do
    (TaskAllocation, cost) ← ClusterCombinatorialAuctionTSPOnBidding(tasks,agents)
    if current cost < Temp then
        AuctionOpt = TaskAllocation
        Temp = current cost
    end if
end for
```

3.6 MAP

After implementing the algorithm we know need to make it more realistic since we implemented it in an empty platform we used an implemented package in matlab called Robotic Tool box we used $BinaryOccupancyGrid$ to create obstacles
Occupancy grids are used to represent a robot work space as a discrete grid. Information about the environment can be loaded from prior knowledge.

Occupancy grids are used in robotics algorithms such as path planning. They are used in mapping applications for integrating sensor information in a discrete map, in path planning for finding collision-free paths, and for localizing robots in a known environment . You can create maps with different sizes and resolutions to fit your specific application.
In order to avoid obstacles we need to use path planning we used a $shortestpath$ function on this tool and we create a two matrices for our path planing technique we create a nod matrix and an obstacle matrix however we meet a real challenge on converting from matrix axis at the top left to plot axis at the bottom right we used an equation of $x = j$, $y = Maxsiz - i$
i: is the counter in the $-y$-axis from the left top
j: is the counter in the x-axis from the left top

3.7 Simulated experiments

In order to test our Algorithm and our path planing technique we need to simulate it in a moving robot we used Pure Pursuit Controller in our our experiment Pure Pursuit Controller:Pure pursuit is an algorithm for tracking that calculates the curvature that moves a car from the position at the moment to some destination. The whole point of the algorithm is to select a target position, which is a little further away from the robot along the way. The pure pursuit is the name derived from the analogy used to describe the process. We tend to think of the robot as a distance from a point on the path-it is following that point. We tend to look a little far before the robot and go to that place.
Theoretical Derivation:Pure pursuit is a method by which the curvature can be geometrically determined and the robot reaches a chosen pathway, called the destination point. This goal is a point on the road that looks at the current position of the vehicle. An arc is built that connects the present point and the target point. The string length of this arc is the distance to the lookout and serves as the third limit when a unique arc is determined and connected to the two points.

Chapter 4

Results

The previously proposed approaches was implemented and simulated on Matlab and simulink version 2018a. The task allocation algorithm was implemented on Matlab while the path planning was simulated using the Robotics tool box. The PC used runs a 64-bit Windows operating system with a 1.6 GHz processor and 8 GB ram.

4.1 Proposed scenario's

In this study w proposed a different scenario's to test the efficiency of the Algorithms we tested for :
Robots $<$ **Tasks.**
Robots $=$ **Tasks.**
Robots $>$ **Tasks.**
in each scenario we tested the Algorithm in small and large scale.

4.2 Performance matrix

In the different simulated scenarios we measure our algorithm performance according to some criteria.

- Computation time.
- Number of tasks compared to Number of Robots (Agents).
- Total cost which is total traveled distance for the Robots.
- Maximum cost which represent the maximum distance traveled by a Robot in the system.

4.3 Results Output:

In the up coming sections we will view the output for each and every algorithm the (X or Empty circle O) represent the position of Robots the colored Dots represent the position of Tasks the line between the Dots is the traveled path with calculated distance.
The Tested results of (The Computation time, Number of tasks compared to Number of Robots (Agents),Total cost which is total traveled distance for the Robots,and Maximum cost which represent the maximum distance traveled by a Robot in the system) for each simulated experiment is included under each 2-D view of the Map.

For the tables output each and every experiment have a two tables

1. Table one for Tasks contain :
    - ID's for the Task.
    - Position of each and every Task in the 2-D map.
2. Table two for Robots contain :
    - ID's for the Robots
    - Position for each and every Robot in the 2-D map.
    - Assign Tasks in the ordered manner.
    - Cost for Robot represent the the distance traveled by the agent.

4.4 simple market results:

As discussed in section 3.5 the algorithm is implemented and test on MATLAB. a simple MTSP problem of 6 agents and 20 tasks were conducted and solved using the simple market algorithm as seen in Figure 4.1.

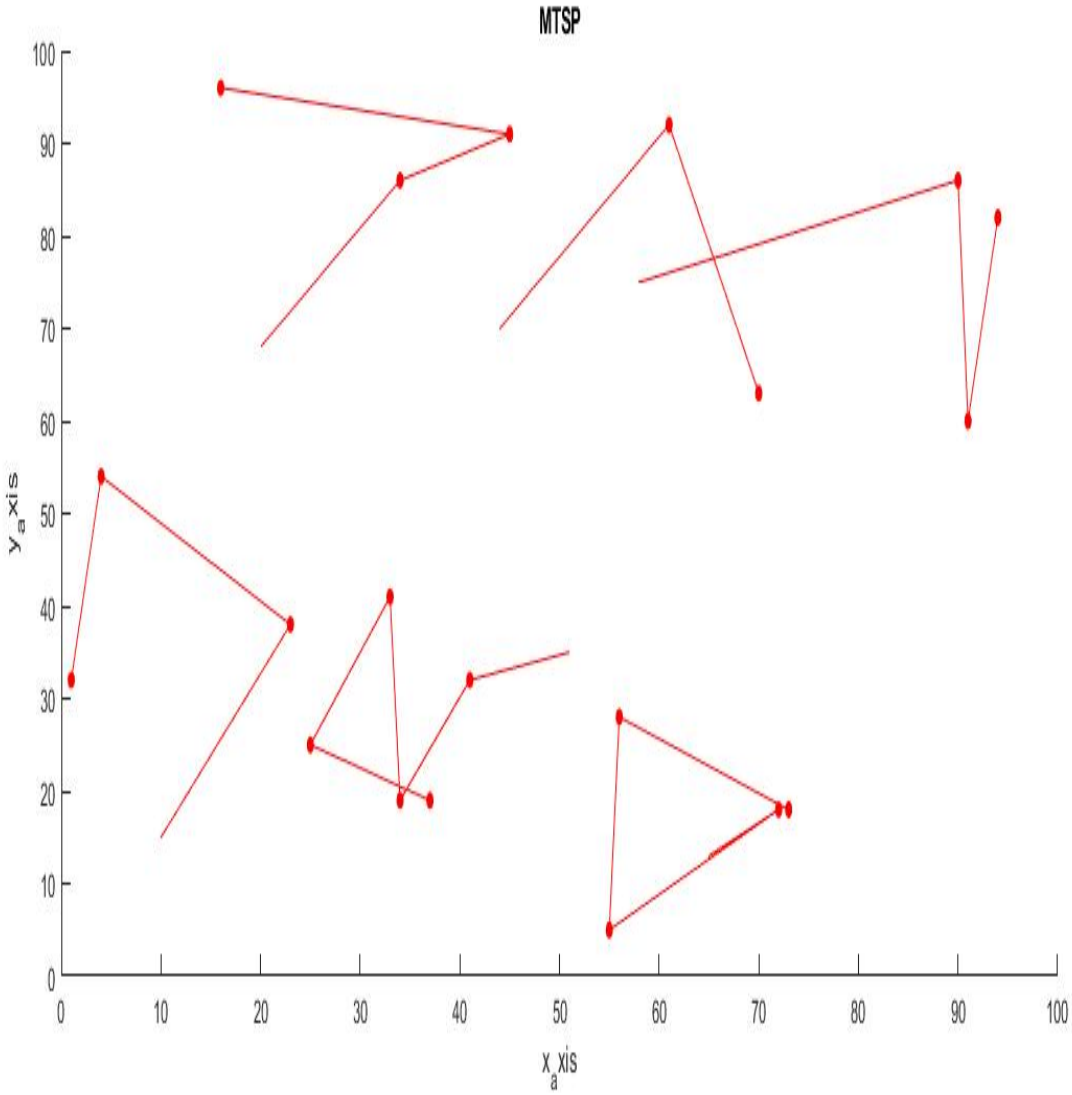

Figure 4.1: Time:0.65099 sec Tasks:20 Agents:6

as we can notice that this results is not the best one as there is crosses (X) in the order of motion so we tried to implement another algorithm as it will be discussed in the flowing sections.

4.5 Cluster combinatorial auction results

In this result we bid on the centroids of clusters as mentioned in section 3.5.3 and in Algorithm 2 the start of solving such a Multiple Traveling Salesman Problem (MTSPs) was from that centroid the agents start there tour form the X centroid of cluster .

4.5.1 Result in small scale :

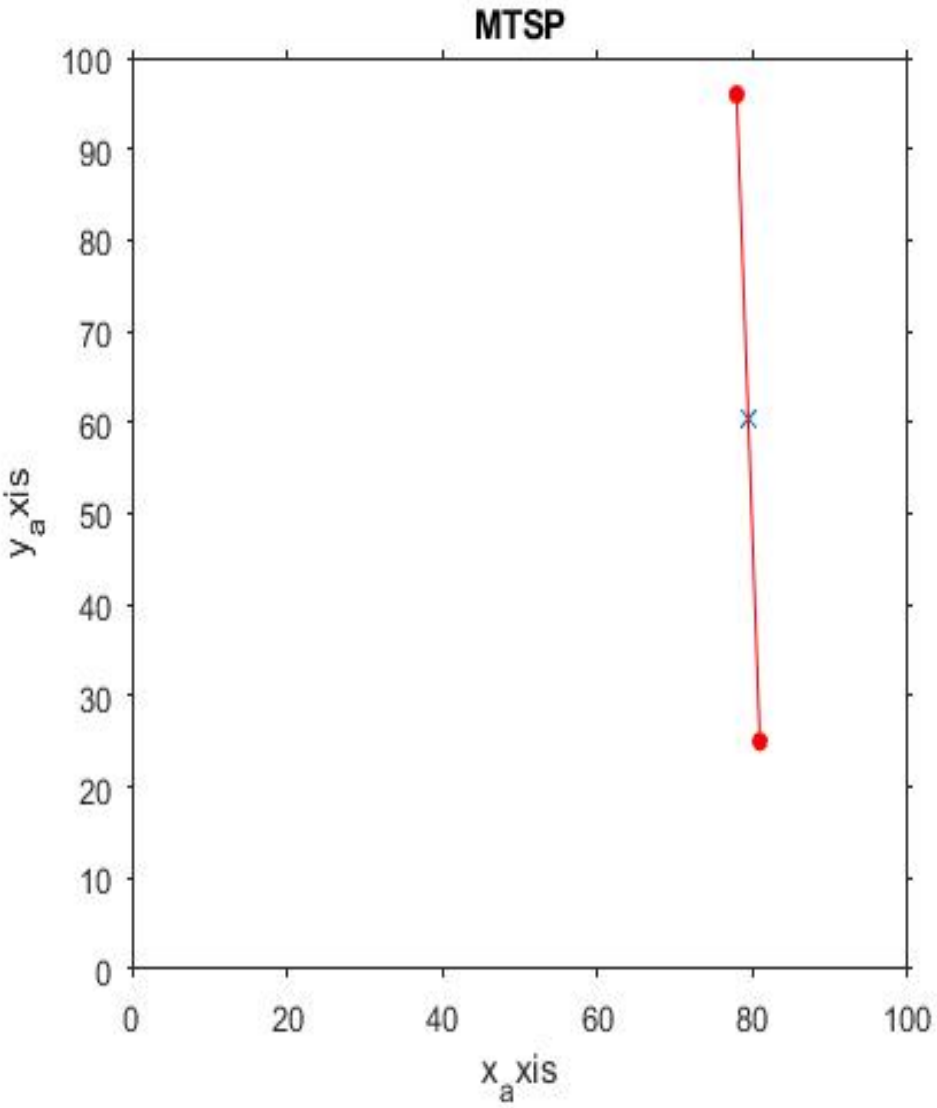

Figure 4.2: Time:0.174057 sec Tasks:2 Agents:1

4.5.2 Result in Large scale :

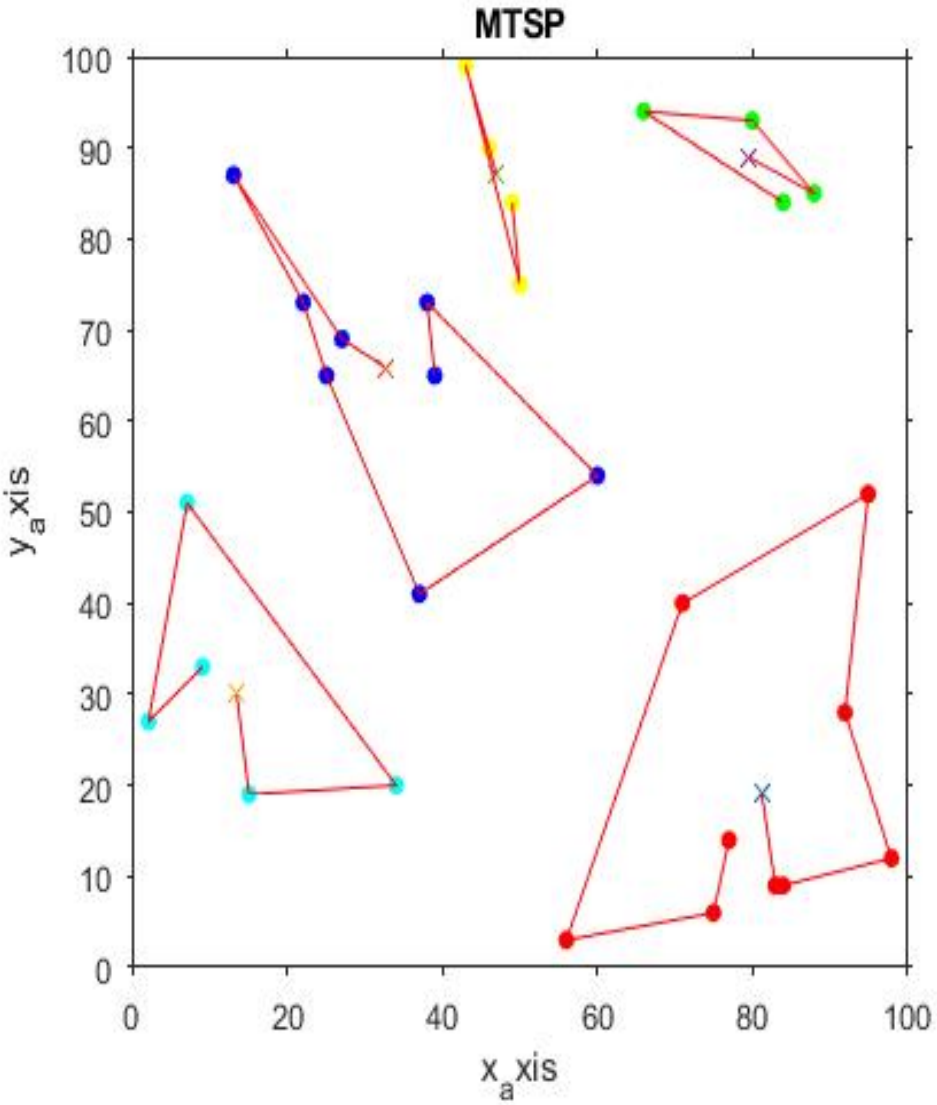

Figure 4.3: Time:0.214457 sec Tasks:30 Agents:5

AS we can notice from this result that the solution is all ways start from the center of the cluster so we wanted to improve the solution more than that.

4.5.3 Result in Extra large scale :

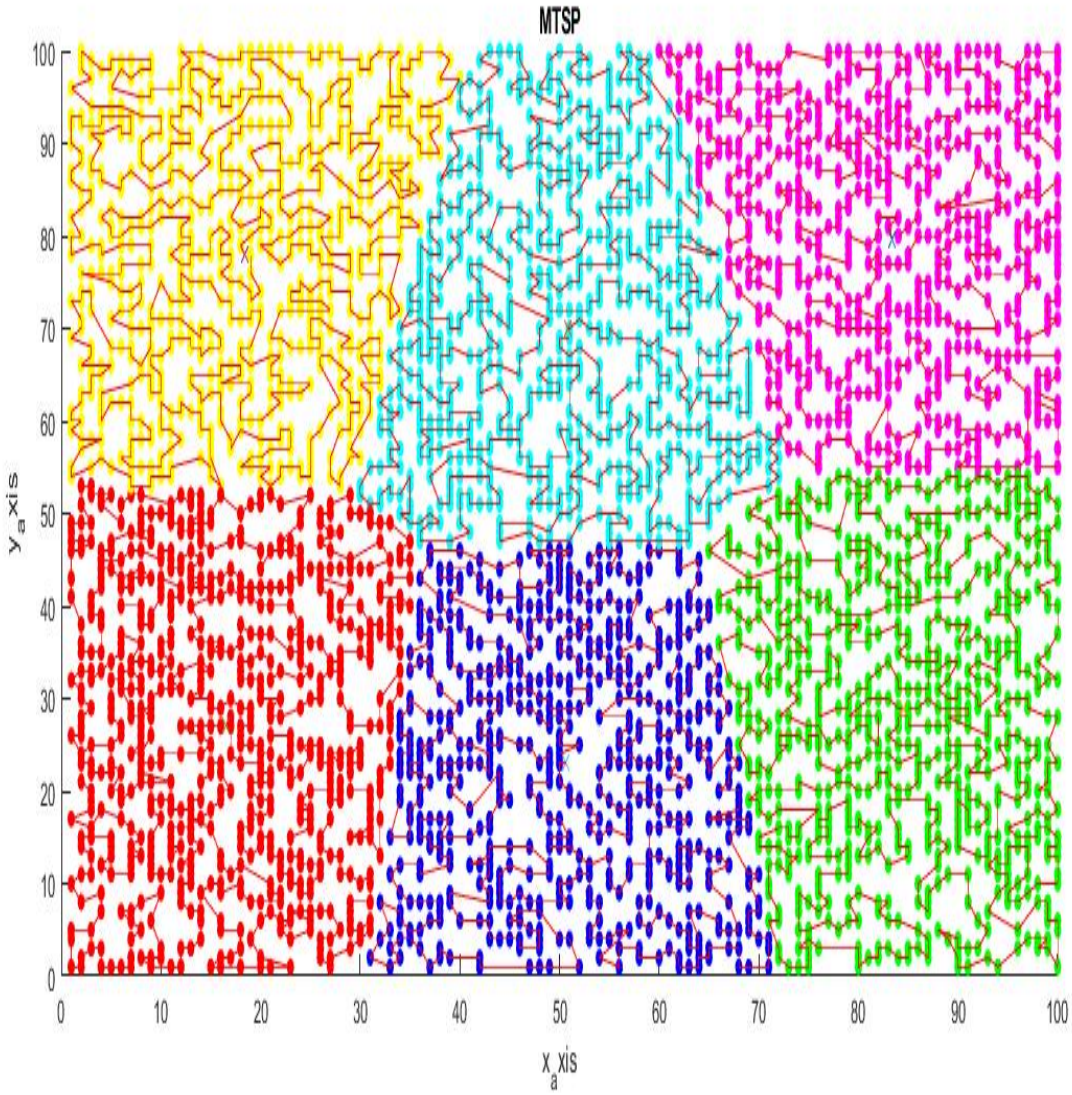

Figure 4.4: Time:13.210116 sec Tasks:5000 Agents:6

As we can notice that cluster appears clear as Crystal at this result.and this show how the algorithm is really powerful at large scale.

4.6 Cluster Combinatorial Auction (TSP) on Bidding

At this result we implemented a path planing for our algorithm we used shortest path between two point function in Matlab . in the next sections we will discuss different scenarios for managing the warehouse. As stated in section 3.5.6 and in Algorithm 4 there is a winner where the winner is one that utilizes the objective function.The advantage of this technique according to [52] is that it runs only once as the bidding is on a group of tasks rather than task by task. To group Tasks, $K - means$ are again used. the idea of clustering is to group a set of data points according to there location and how this location is related to the locations of the other tasks.

4.6.1 Robots < Tasks

4.6.2 Small scale problems :

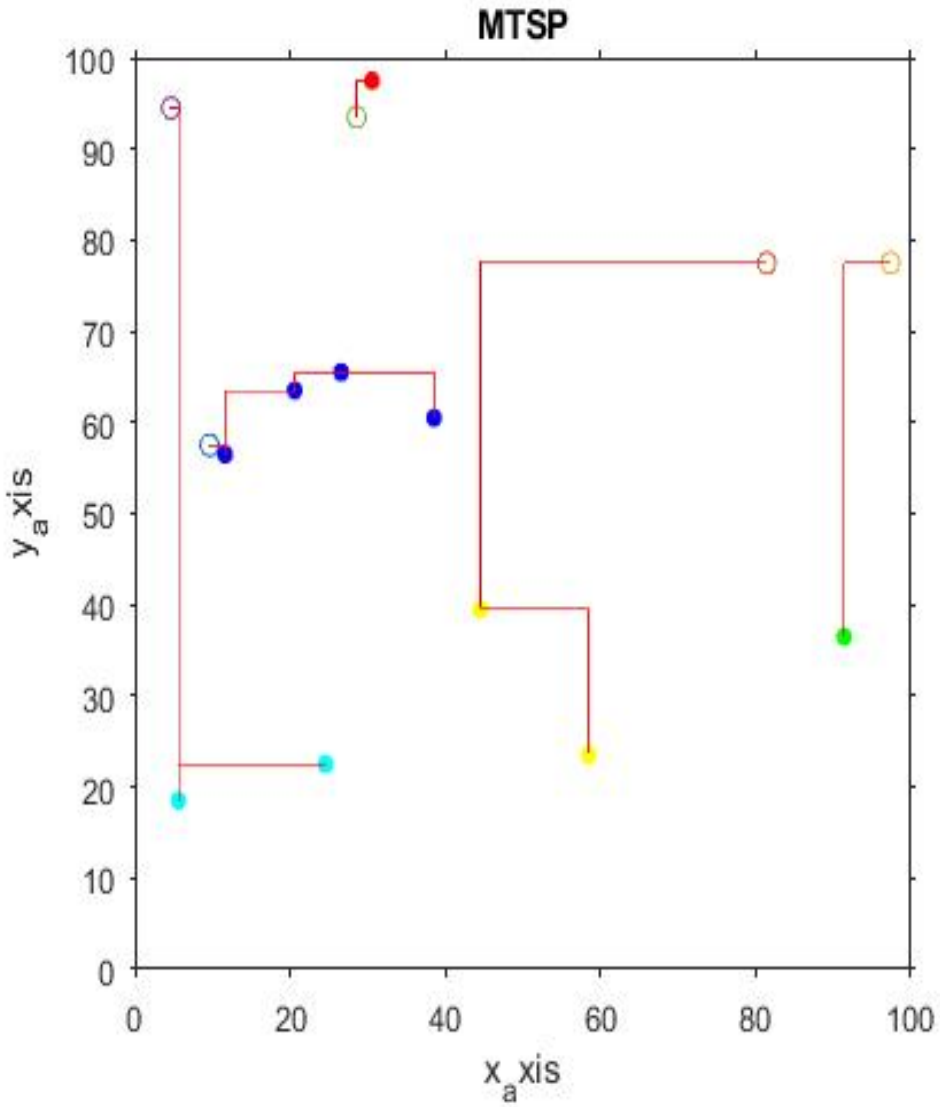

Figure 4.5: Time:0.3413 sec Tasks:10 Agents:5 Max cost:164.733 Total cost:442.173

Table 4.1: Tasks positions Small scale problem (R<T)

Tasks ID	X	Y
1	91.5000000000000	36.5000000000000
2	44.5000000000000	39.5000000000000
3	38.5000000000000	60.5000000000000
4	58.5000000000000	23.5000000000000
5	26.5000000000000	65.5000000000000
6	20.5000000000000	63.5000000000000
7	11.5000000000000	56.5000000000000
8	30.5000000000000	97.5000000000000
9	5.50000000000000	18.5000000000000
10	24.5000000000000	22.5000000000000

Table 4.2: Agents Small scale problem (R<T)

Agents.ID	position	Assign Tasks	Cost
1	[9.50000000000000;57.5000000000000]	[7,6,5,3]	44
2	[81.5000000000000;77.5000000000000]	[2,4]	158.486451477014
3	[97.5000000000000;77.5000000000000]	1	66.4680374315355
4	[4.50000000000000;94.5000000000000]	[9,10]	164.733689857734
5	[28.5000000000000;93.5000000000000]	8	8.48528137423857

4.6.3 Large scale problems:

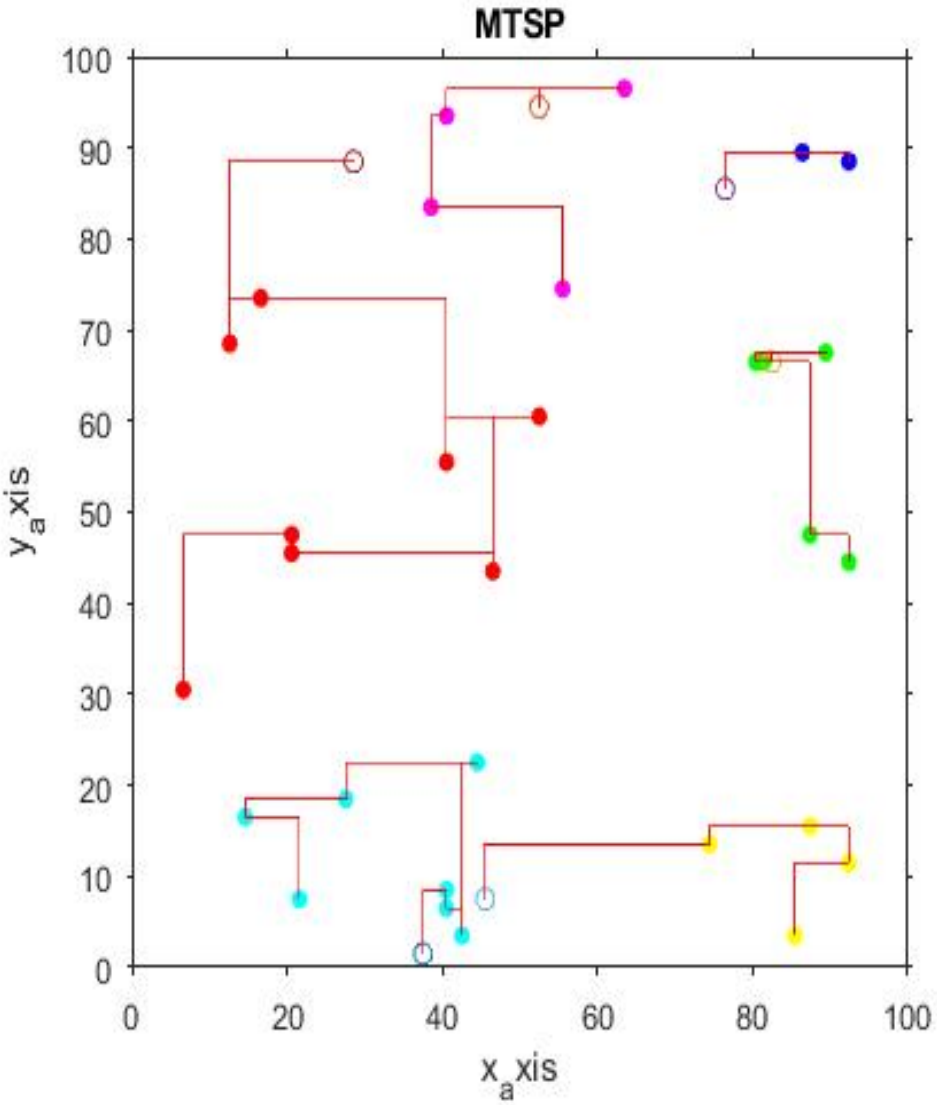

Figure 4.6: Time:0.392535 sec Tasks:30 Agents:6 Max cost:188 Total cost:515.24221

Table 4.3: Tasks positions Large scale problems (R<T)

Tasks ID	X	Y
1	16.50000000000000	73.50000000000000
2	14.50000000000000	16.50000000000000
3	89.50000000000000	67.50000000000000
4	92.50000000000000	11.50000000000000
5	20.50000000000000	47.50000000000000
6	40.50000000000000	6.500000000000000
7	6.500000000000000	30.50000000000000
8	92.50000000000000	88.50000000000000
9	81.50000000000000	66.50000000000000
10	20.50000000000000	45.50000000000000
11	38.50000000000000	83.50000000000000
12	86.50000000000000	89.50000000000000
13	12.50000000000000	68.50000000000000
14	74.50000000000000	13.50000000000000
15	46.50000000000000	43.50000000000000
16	42.50000000000000	3.500000000000000
17	40.50000000000000	8.500000000000000
18	27.50000000000000	18.50000000000000
19	63.50000000000000	96.50000000000000
20	87.50000000000000	15.50000000000000
21	40.50000000000000	93.50000000000000
22	92.50000000000000	44.50000000000000
23	21.50000000000000	7.500000000000000
24	40.50000000000000	55.50000000000000
25	52.50000000000000	60.50000000000000
26	55.50000000000000	74.50000000000000
27	87.50000000000000	47.50000000000000
28	85.50000000000000	3.500000000000000
29	80.50000000000000	66.50000000000000
30	44.50000000000000	22.50000000000000

Table 4.4: Agents Large scale problems (R<T)

Agents.ID	position	Assign Tasks	Cost
1	[45.50000000000000;7.500000000000000]	[14,20,4,28]	74
2	[28.50000000000000;88.50000000000000]	[13,1,24,25,15,10,5,7]	188
3	[37.50000000000000;1.500000000000000]	[17,6,16,30,18,2,23]	90
4	[52.50000000000000;94.50000000000000]	[19,21,11,26]	77
5	[82.50000000000000;66.50000000000000]	[3,29,9,27,22]	52
6	[76.50000000000000;85.50000000000000]	[12,8]	34.2422103119899

4.6.4 Robots=Tasks

4.6.5 Small scale :

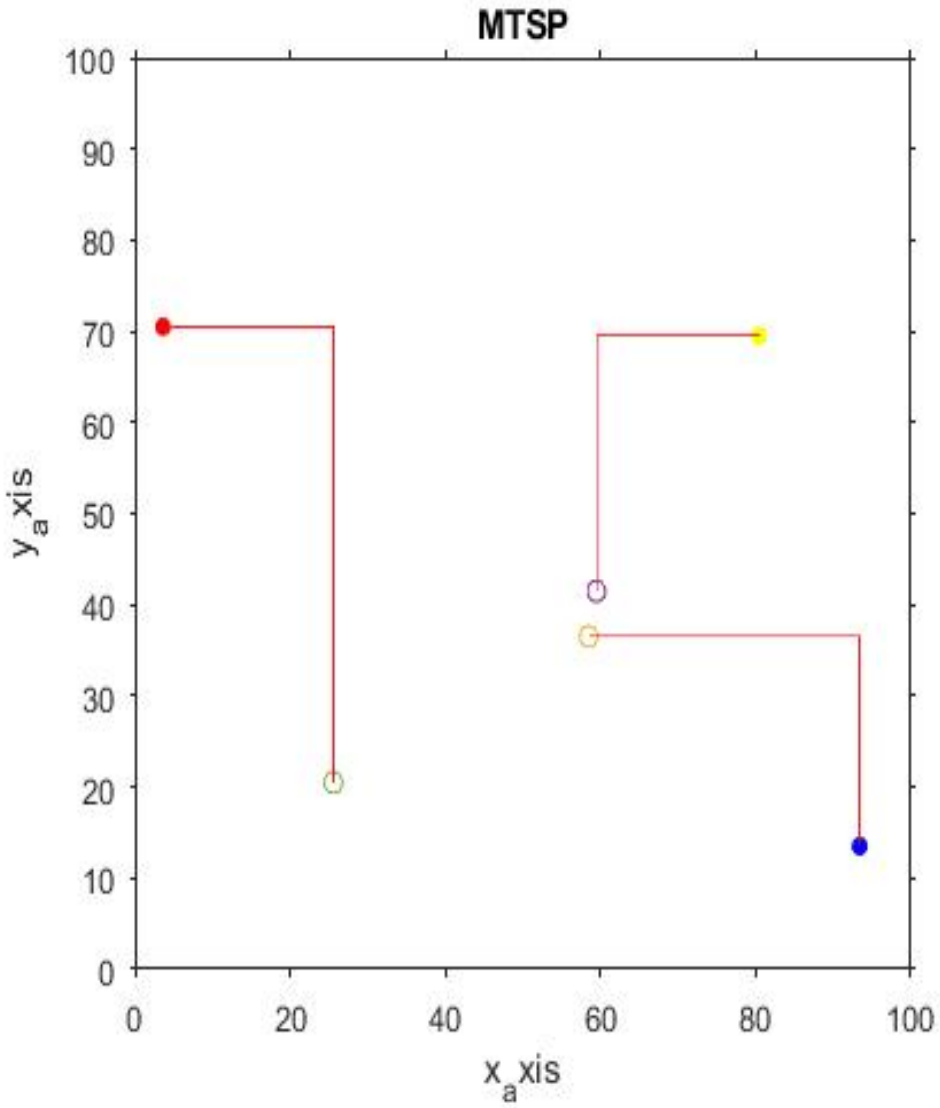

Figure 4.7: Time:0.281657 sec Tasks:3 Agents:3 Max cost:101.82337 Total cost:253.14422

Table 4.5: Tasks positions Small scale problem (R=T)

Tasks ID	X	Y
1	93.5000000000000	13.5000000000000
2	3.50000000000000	70.5000000000000
3	80.5000000000000	69.5000000000000

Table 4.6: Agents Small scale problem (R=T)

Agent ID	position	Assign task	Cost
1	[58.5000000000000;36.5000000000000]	1	82.0243866176395
2	[59.5000000000000;41.5000000000000]	3	69.2964645562817
3	[25.5000000000000;20.5000000000000]	2	101.823376490863

4.6.6 Large scale:

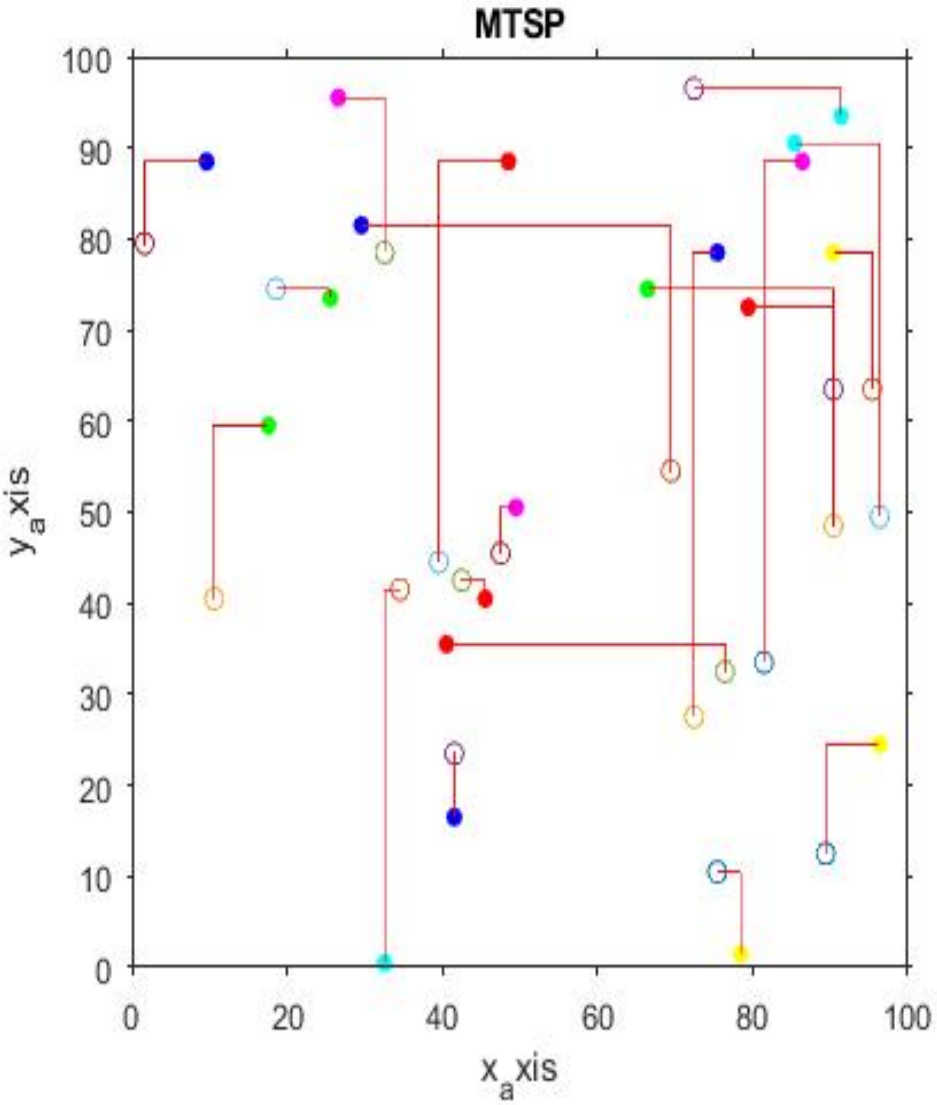

Figure 4.8: Time:0.34698 sec Tasks:20 Agents:20 Max cost:94.752308 Total cost:854.184991

Table 4.7: Tasks positions Large scale (R=T)

Tasks ID	X	Y
1	49.50000000000000	50.50000000000000
2	48.50000000000000	88.50000000000000
3	75.50000000000000	78.50000000000000
4	29.50000000000000	81.50000000000000
5	40.50000000000000	35.50000000000000
6	79.50000000000000	72.50000000000000
7	85.50000000000000	90.50000000000000
8	78.50000000000000	1.500000000000000
9	41.50000000000000	16.50000000000000
10	86.50000000000000	88.50000000000000
11	96.50000000000000	24.50000000000000
12	66.50000000000000	74.50000000000000
13	25.50000000000000	73.50000000000000
14	32.50000000000000	0.500000000000000
15	9.500000000000000	88.50000000000000
16	91.50000000000000	93.50000000000000
17	26.50000000000000	95.50000000000000
18	17.50000000000000	59.50000000000000
19	90.50000000000000	78.50000000000000
20	45.50000000000000	40.50000000000000

Table 4.8: Agents Large scale (R=T)

Agent ID	position	Assign task	Cost
1	[89.5000000000000;12.5000000000000]	11	26.8700576850888
2	[95.5000000000000;63.5000000000000]	19	28.2842712474619
3	[10.5000000000000;40.5000000000000]	18	36.7695526217005
4	[90.5000000000000;63.5000000000000]	6	28.2842712474619
5	[76.5000000000000;32.5000000000000]	5	55.1543289325507
6	[39.5000000000000;44.5000000000000]	2	74.9533188057740
7	[47.5000000000000;45.5000000000000]	1	9.89949493661167
8	[75.5000000000000;10.5000000000000]	8	16.9705627484771
9	[34.5000000000000;41.5000000000000]	14	60.8111831820431
10	[72.5000000000000;27.5000000000000]	3	76.3675323681471
11	[72.5000000000000;96.5000000000000]	16	31.1126983722081
12	[42.5000000000000;42.5000000000000]	20	7.07106781186548
13	[18.5000000000000;74.5000000000000]	13	11.3137084989848
14	[1.50000000000000;79.5000000000000]	15	24.0416305603426
15	[81.5000000000000;33.5000000000000]	10	84.8528137423857
16	[69.5000000000000;54.5000000000000]	4	94.7523086789974
17	[90.5000000000000;48.5000000000000]	12	70.7106781186548
18	[41.5000000000000;23.5000000000000]	9	9.89949493661167
19	[32.5000000000000;78.5000000000000]	17	32.5269119345812
20	[96.5000000000000;49.5000000000000]	7	73.5391052434009

4.7 Robots > Tasks

4.7.1 Small scale:

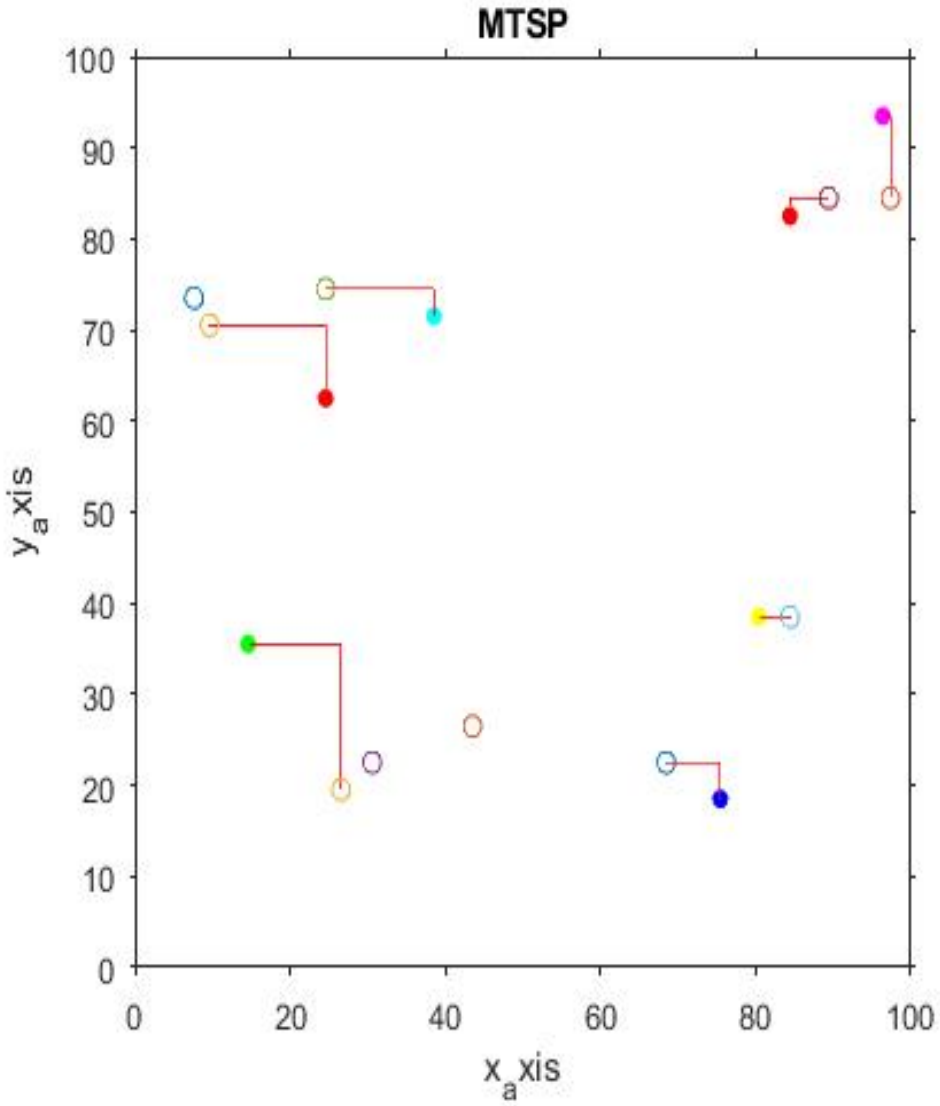

Figure 4.9: Time:0.238791 sec Tasks:7 Agents:10 Max cost:39.59797 Total cost:141.42135

Table 4.9: Tasks positions Small scale problem (R>T)

Tasks ID	X	Y
1	84.5000000000000	82.5000000000000
2	24.5000000000000	62.5000000000000
3	14.5000000000000	35.5000000000000
4	80.5000000000000	38.5000000000000
5	38.5000000000000	71.5000000000000
6	75.5000000000000	18.5000000000000
7	96.5000000000000	93.5000000000000

Table 4.10: Agents Small scale problem (R>T)

Agent ID	position	Assign task	Cost
1	[68.5000000000000;22.5000000000000]	6	15.5563491861040
2	[43.5000000000000;26.5000000000000]	[]	0
3	[26.5000000000000;19.5000000000000]	3	39.5979797464467
4	[30.5000000000000;22.5000000000000]	[]	0
5	[24.5000000000000;74.5000000000000]	5	24.0416305603426
6	[84.5000000000000;38.5000000000000]	4	5.65685424949238
7	[89.5000000000000;84.5000000000000]	1	9.89949493661167
8	[7.50000000000000;73.5000000000000]	[]	0
9	[97.5000000000000;84.5000000000000]	7	14.1421356237310
10	[9.50000000000000;70.5000000000000]	2	32.5269119345812

4.7.2 Large scale problem:

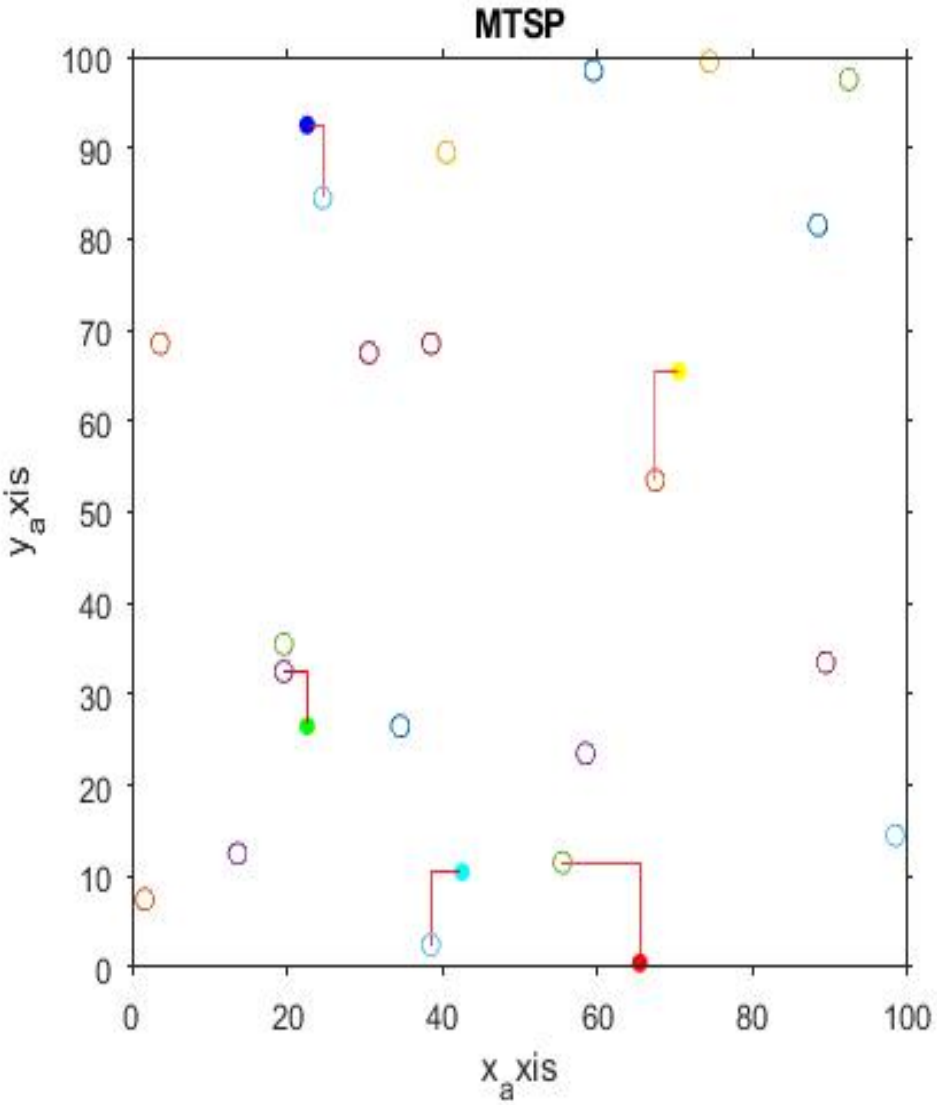

Figure 4.10: Time:0.406710 sec Tasks:5 Agents:20 Max cost:29.6984 Total cost:94.7523

Table 4.11: Tasks positions Large scale problem (R>T)

Tasks ID	X	Y
1	42.5000000000000	10.5000000000000
2	70.5000000000000	65.5000000000000
3	22.5000000000000	92.5000000000000
4	65.5000000000000	0.500000000000000
5	22.5000000000000	26.5000000000000

Table 4.12: Agents Large scale problem (R>T)

Agent ID	position	Assign task	Cost
1	[19.5000000000000;32.5000000000000]	5	12.7279220613579
2	[92.5000000000000;97.5000000000000]	[]	0
3	[24.5000000000000;84.5000000000000]	3	14.1421356237310
4	[30.5000000000000;67.5000000000000]	[]	0
5	[59.5000000000000;98.5000000000000]	[]	0
6	[3.50000000000000;68.5000000000000]	[]	0
7	[40.5000000000000;89.5000000000000]	[]	0
8	[58.5000000000000;23.5000000000000]	[]	0
9	[55.5000000000000;11.5000000000000]	4	29.6984848098350
10	[98.5000000000000;14.5000000000000]	[]	0
11	[38.5000000000000;68.5000000000000]	[]	0
12	[34.5000000000000;26.5000000000000]	[]	0
13	[1.50000000000000;7.50000000000000]	[]	0
14	[74.5000000000000;99.5000000000000]	[]	0
15	[13.5000000000000;12.5000000000000]	[]	0
16	[19.5000000000000;35.5000000000000]	[]	0
17	[38.5000000000000;2.50000000000000]	1	16.9705627484771
18	[89.5000000000000;33.5000000000000]	[]	0
19	[88.5000000000000;81.5000000000000]	[]	0
20	[67.5000000000000;53.5000000000000]	2	21.2132034355964

4.8 Implemented Map

we wanted to make a warehouse 2-D that will simulate the real world environment we used as mentioned in section 3.6 an Occupancy grid to create our map

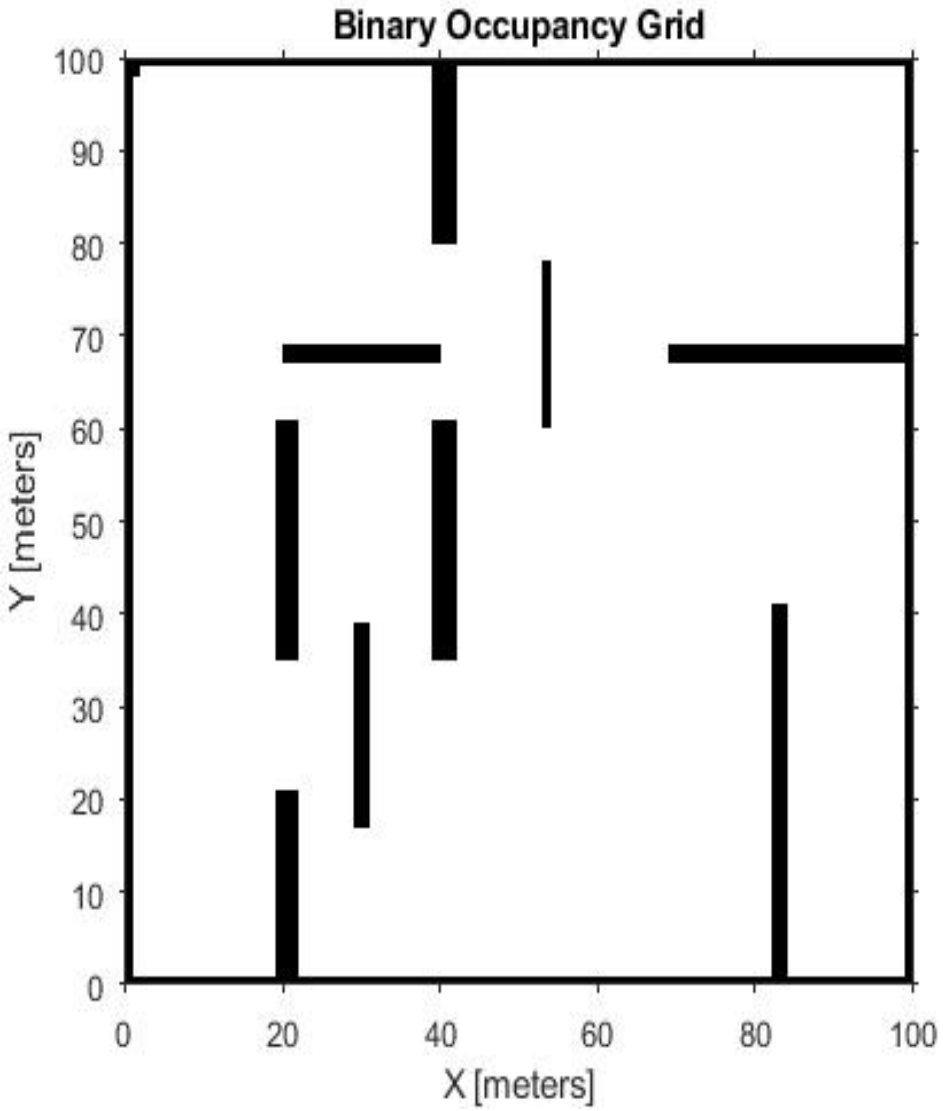

Figure 4.11: Warehouse Map

4.8.1 Robots<Tasks

4.8.2 Small scale problems :

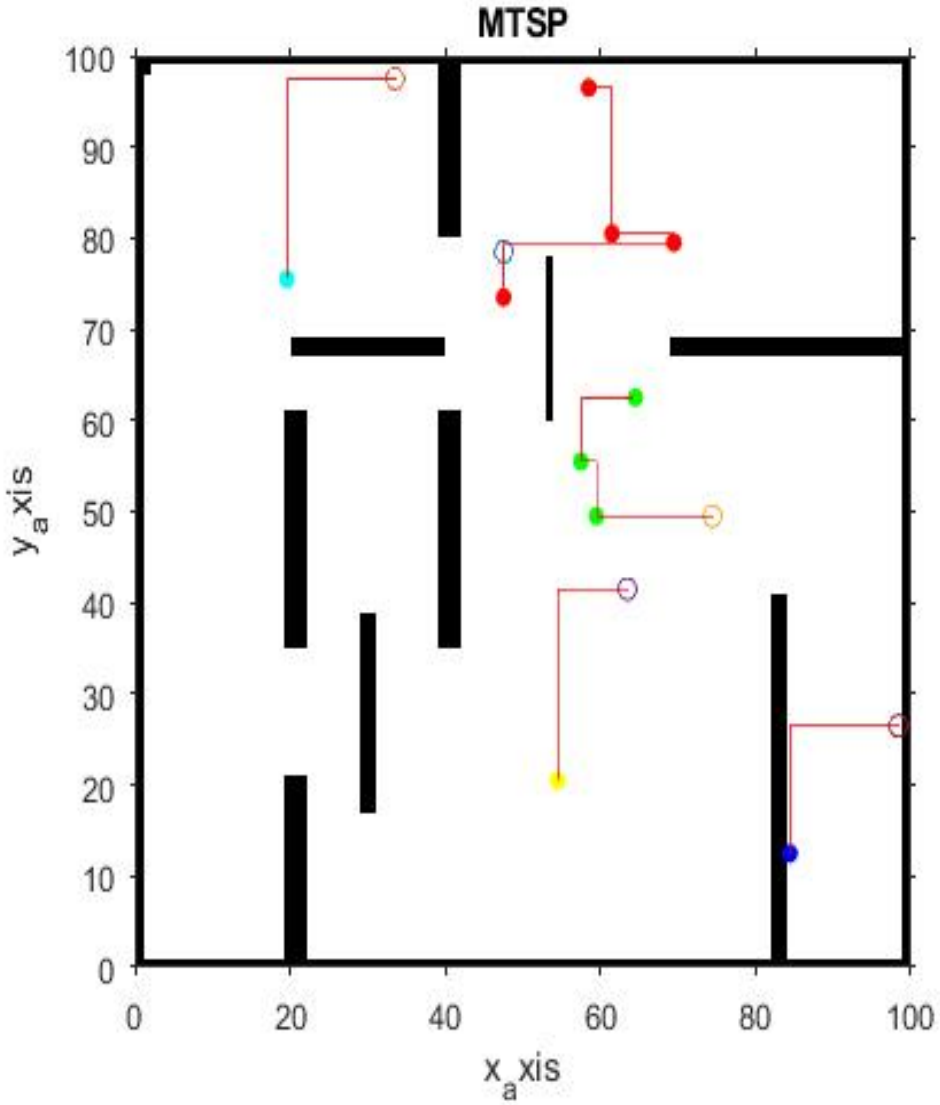

Figure 4.12: Time:0.3326 sec Tasks:10 Agents:5 Max cost:61 Total cost:230.936

Table 4.13: Tasks positions Small scale problems R<T

Tasks ID	X	Y
1	61.5000000000000	80.5000000000000
2	19.5000000000000	75.5000000000000
3	59.5000000000000	49.5000000000000
4	64.5000000000000	62.5000000000000
5	47.5000000000000	73.5000000000000
6	57.5000000000000	55.5000000000000
7	69.5000000000000	79.5000000000000
8	58.5000000000000	96.5000000000000
9	84.5000000000000	12.5000000000000
10	54.5000000000000	20.5000000000000

Table 4.14: Agents Small scale problems R<T

Agents.ID	position	Assign Tasks	Cost
1	[98.5000000000000;26.5000000000000]	9	39.5979797464467
2	[47.5000000000000;78.5000000000000]	[5,7,1,8]	61
3	[33.5000000000000;97.5000000000000]	2	50.9116882454314
4	[74.5000000000000;49.5000000000000]	[3,6,4]	37
5	[63.5000000000000;41.5000000000000]	10	42.4264068711929

4.8.3 Large scale problems:

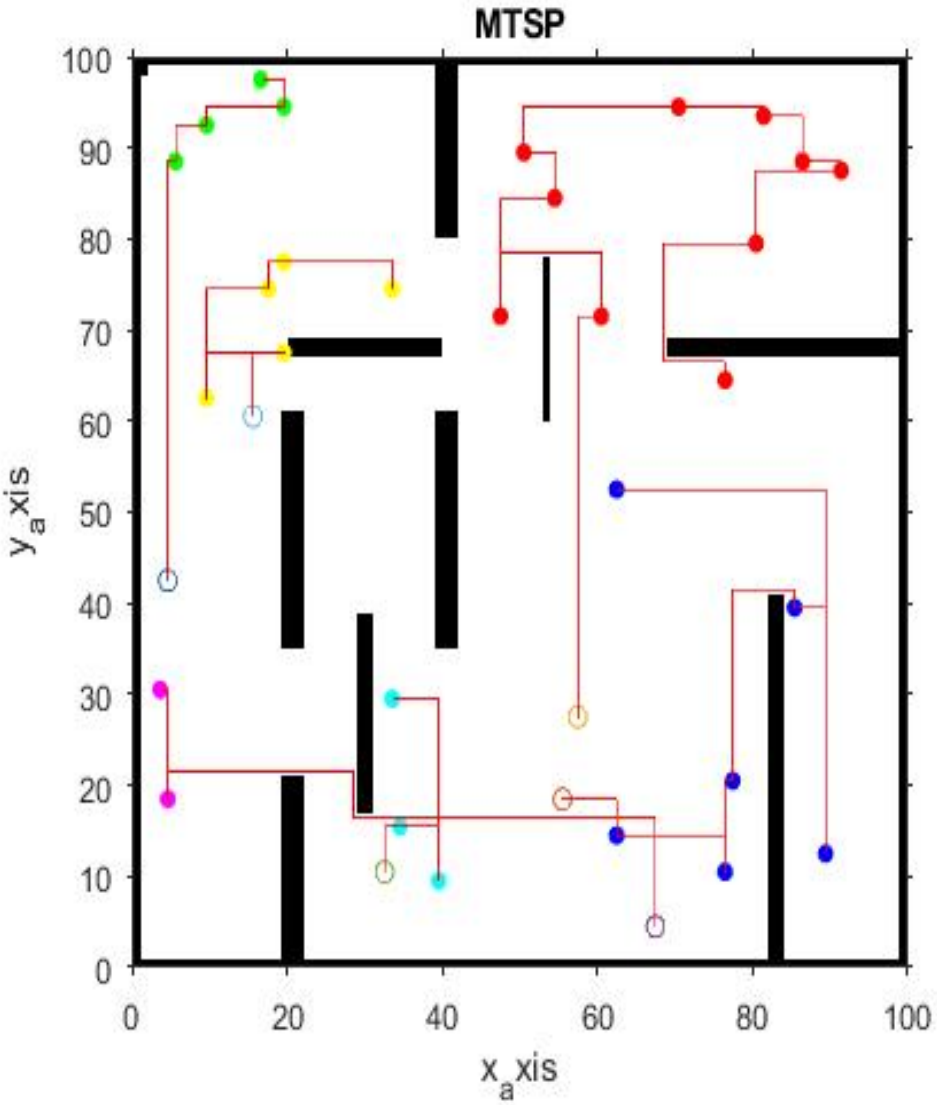

Figure 4.13: Time:0.236 sec Tasks:30 Agents:6 Max cost:210 Total cost:724.0539

Table 4.15: Tasks positions Large scale problems R<T

Tasks ID	X	Y
1	91.50000000000000	87.50000000000000
2	5.500000000000000	88.50000000000000
3	54.50000000000000	84.50000000000000
4	34.50000000000000	15.50000000000000
5	33.50000000000000	29.50000000000000
6	19.50000000000000	94.50000000000000
7	47.50000000000000	71.50000000000000
8	80.50000000000000	79.50000000000000
9	39.50000000000000	9.500000000000000
10	77.50000000000000	20.50000000000000
11	62.50000000000000	14.50000000000000
12	9.500000000000000	62.50000000000000
13	62.50000000000000	52.50000000000000
14	33.50000000000000	74.50000000000000
15	86.50000000000000	88.50000000000000
16	9.500000000000000	92.50000000000000
17	19.50000000000000	77.50000000000000
18	60.50000000000000	71.50000000000000
19	50.50000000000000	89.50000000000000
20	89.50000000000000	12.50000000000000
21	3.500000000000000	30.50000000000000
22	19.50000000000000	67.50000000000000
23	81.50000000000000	93.50000000000000
24	17.50000000000000	74.50000000000000
25	16.50000000000000	97.50000000000000
26	70.50000000000000	94.50000000000000
27	76.50000000000000	10.50000000000000
28	85.50000000000000	39.50000000000000
29	76.50000000000000	64.50000000000000
30	4.500000000000000	18.50000000000000

Table 4.16: Agents Large scale problems R<T

Agents.ID	position	Assign Tasks	Cost
1	[4.50000000000000;42.5000000000000]	[2,16,6,25]	73
2	[55.5000000000000;18.5000000000000]	[11,27,10,28,20,13]	169
3	[57.5000000000000;27.5000000000000]	[18,7,3,19,26,23,15,1,8,29]	210
4	[67.5000000000000;4.50000000000000]	[30,21]	160.053938301809
5	[32.5000000000000;10.5000000000000]	[4,9,5]	44
6	[15.5000000000000;60.5000000000000]	[22,12,24,17,14]	68

4.9 Robots=Tasks

4.9.1 Small scale:

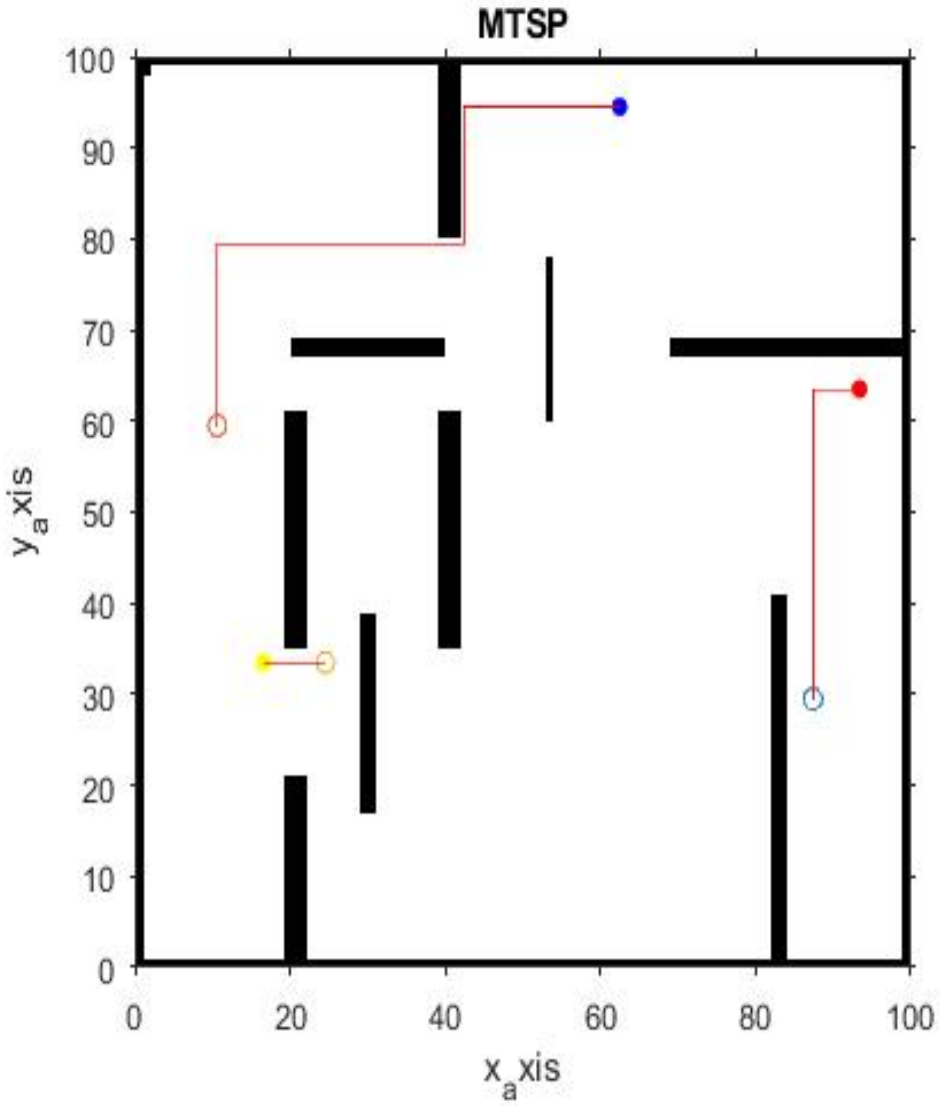

Figure 4.14: Time:0.2752 sec Tasks:3 Agents:3 Max cost:123.0365 Total cost:190.9188

Table 4.18: Agents Small scale problem R= T

Agent ID	position	Assign task	Cost
1	[87.5000000000000;29.5000000000000]	2	56.5685424949238
2	[10.5000000000000;59.5000000000000]	1	123.036579926459
3	[24.5000000000000;33.5000000000000]	3	11.3137084989848

Table 4.17: Tasks positions Small scale problem R= T

Tasks ID	X	Y
1	62.5000000000000	94.5000000000000
2	93.5000000000000	63.5000000000000
3	16.5000000000000	33.5000000000000

4.9.2 Large scale problems:

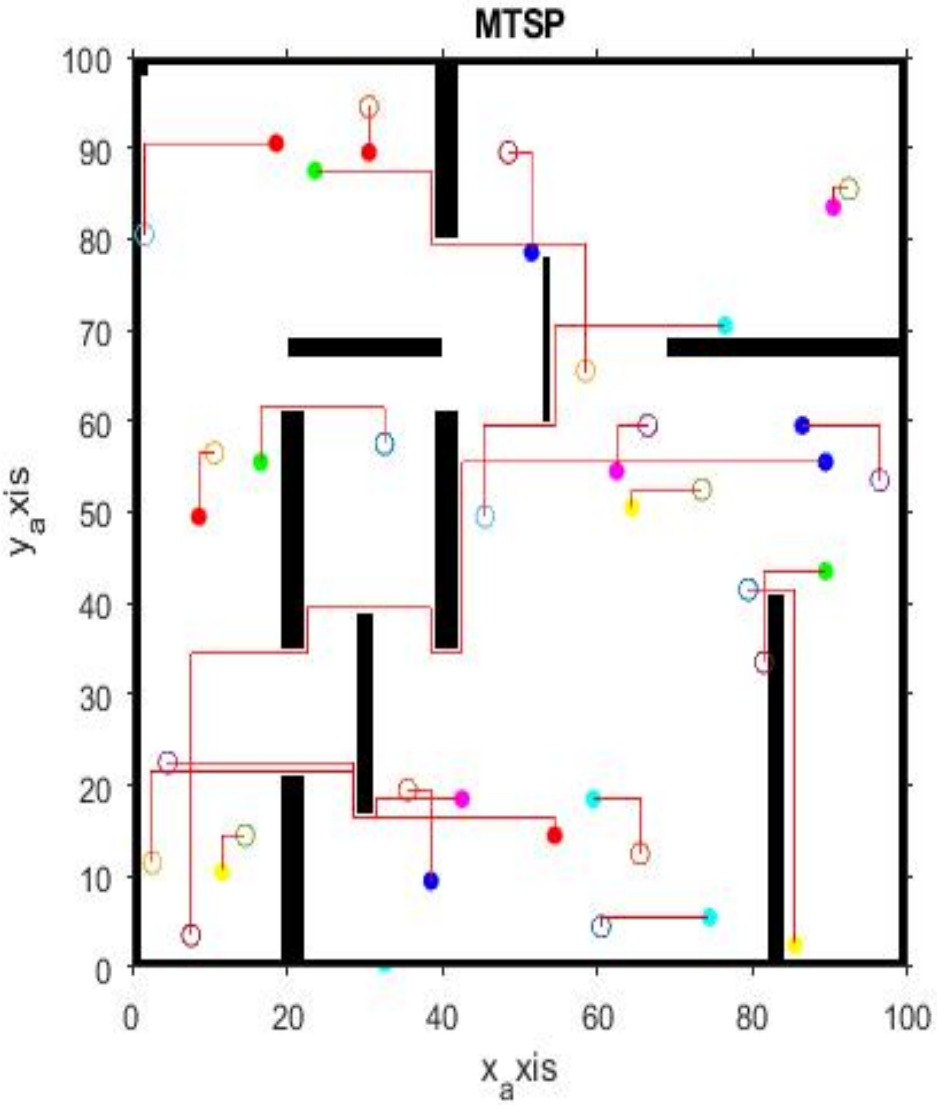

Figure 4.15: Time:0.371498 sec Tasks:20 Agents:20 Max cost:203.646 Total cost:847.11392

Table 4.19: Tasks positions Large scale problem R= T

Tasks ID	X	Y
1	86.50000000000000	59.50000000000000
2	51.50000000000000	78.50000000000000
3	90.50000000000000	83.50000000000000
4	54.50000000000000	14.50000000000000
5	11.50000000000000	10.50000000000000
6	76.50000000000000	70.50000000000000
7	16.50000000000000	55.50000000000000
8	62.50000000000000	54.50000000000000
9	85.50000000000000	2.500000000000000
10	74.50000000000000	5.500000000000000
11	59.50000000000000	18.50000000000000
12	38.50000000000000	9.500000000000000
13	23.50000000000000	87.50000000000000
14	8.500000000000000	49.50000000000000
15	89.50000000000000	43.50000000000000
16	89.50000000000000	55.50000000000000
17	42.50000000000000	18.50000000000000
18	18.50000000000000	90.50000000000000
19	30.50000000000000	89.50000000000000
20	64.50000000000000	50.50000000000000

Table 4.20: Agents Large scale problem R= T

Agent ID	position	Assign task	Cost
1	[48.5000000000000;89.5000000000000]	2	19.7989898732233
2	[32.5000000000000;57.5000000000000]	7	36.7695526217005
3	[35.5000000000000;19.5000000000000]	12	18.3847763108502
4	[58.5000000000000;65.5000000000000]	13	80.6101730552664
5	[96.5000000000000;53.5000000000000]	1	22.6274169979695
6	[73.5000000000000;52.5000000000000]	20	15.5563491861040
7	[1.50000000000000;80.5000000000000]	18	38.1837661840736
8	[81.5000000000000;33.5000000000000]	15	25.4558441227157
9	[60.5000000000000;4.50000000000000]	10	21.2132034355964
10	[30.5000000000000;94.5000000000000]	19	7.07106781186548
11	[2.50000000000000;11.5000000000000]	4	97.5807358037436
12	[66.5000000000000;59.5000000000000]	8	12.7279220613579
13	[14.5000000000000;14.5000000000000]	5	9.89949493661167
14	[45.5000000000000;49.5000000000000]	6	73.5391052434009
15	[7.50000000000000;3.50000000000000]	16	203.646752981726
16	[79.5000000000000;41.5000000000000]	9	63.6396103067893
17	[65.5000000000000;12.5000000000000]	11	16.9705627484771
18	[10.5000000000000;56.5000000000000]	14	12.7279220613579
19	[4.50000000000000;22.5000000000000]	17	65.0538238691624
20	[92.5000000000000;85.5000000000000]	3	5.65685424949238

4.10 Robots > Tasks

4.10.1 Small scale:

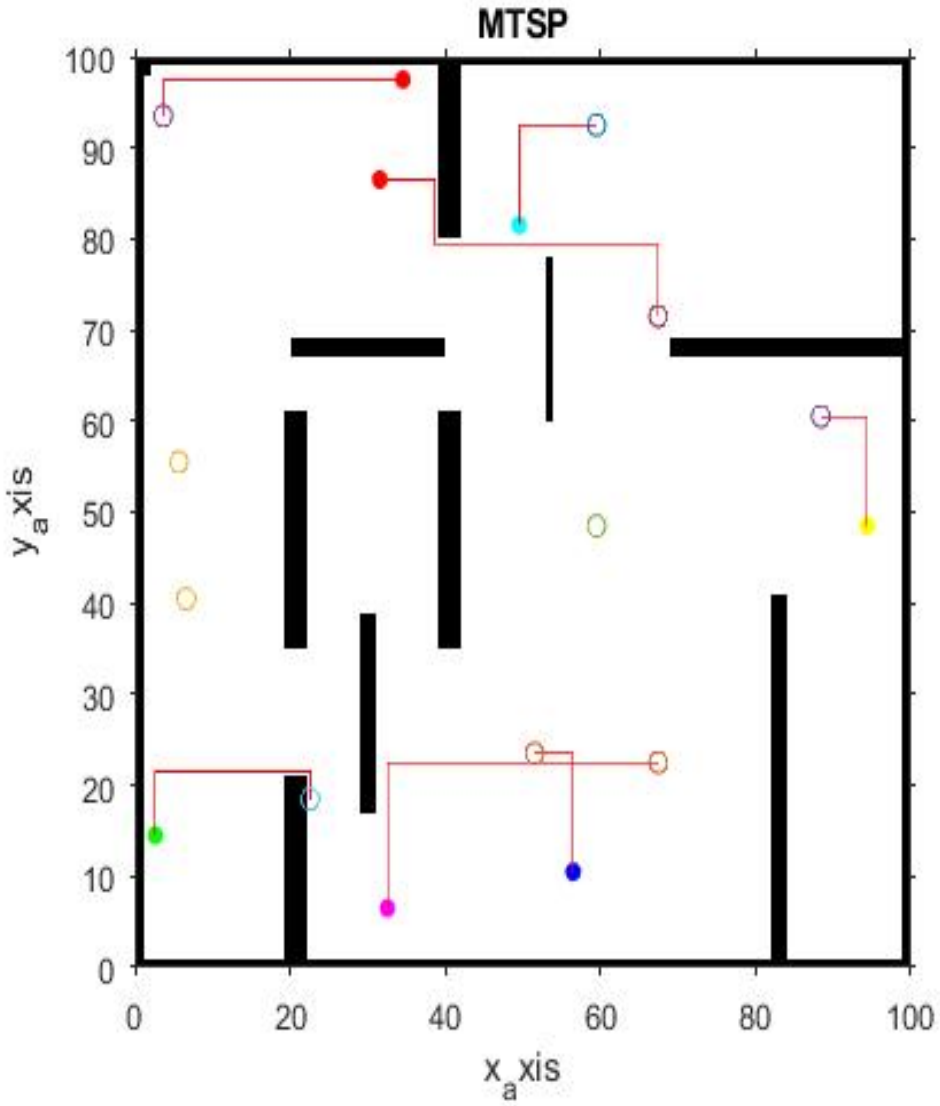

Figure 4.16: Time:0.22055 sec Tasks:7 Agents:10 Max cost:72.12489 Total cost:316.783

Table 4.22: Agents small scale problem R>T

Agent ID	position	Assign task	Cost
1	[67.5000000000000;22.5000000000000]	7	72.1248916810279
2	[5.50000000000000;55.5000000000000]	[]	0
3	[3.50000000000000;93.5000000000000]	2	49.4974746830583
4	[59.5000000000000;48.5000000000000]	[]	0
5	[22.5000000000000;18.5000000000000]	5	42.4264068711929
6	[67.5000000000000;71.5000000000000]	6	72.1248916810279
7	[59.5000000000000;92.5000000000000]	3	29.6984848098350
8	[51.5000000000000;23.5000000000000]	4	25.4558441227157
9	[6.50000000000000;40.5000000000000]	[]	0
10	[88.5000000000000;60.5000000000000]	1	25.4558441227157

Table 4.21: Tasks positions small scale problem R>T

Tasks ID	X	Y
1	94.5000000000000	48.5000000000000
2	34.5000000000000	97.5000000000000
3	49.5000000000000	81.5000000000000
4	56.5000000000000	10.5000000000000
5	2.50000000000000	14.5000000000000
6	31.5000000000000	86.5000000000000
7	32.5000000000000	6.50000000000000

4.10.2 Large scale:

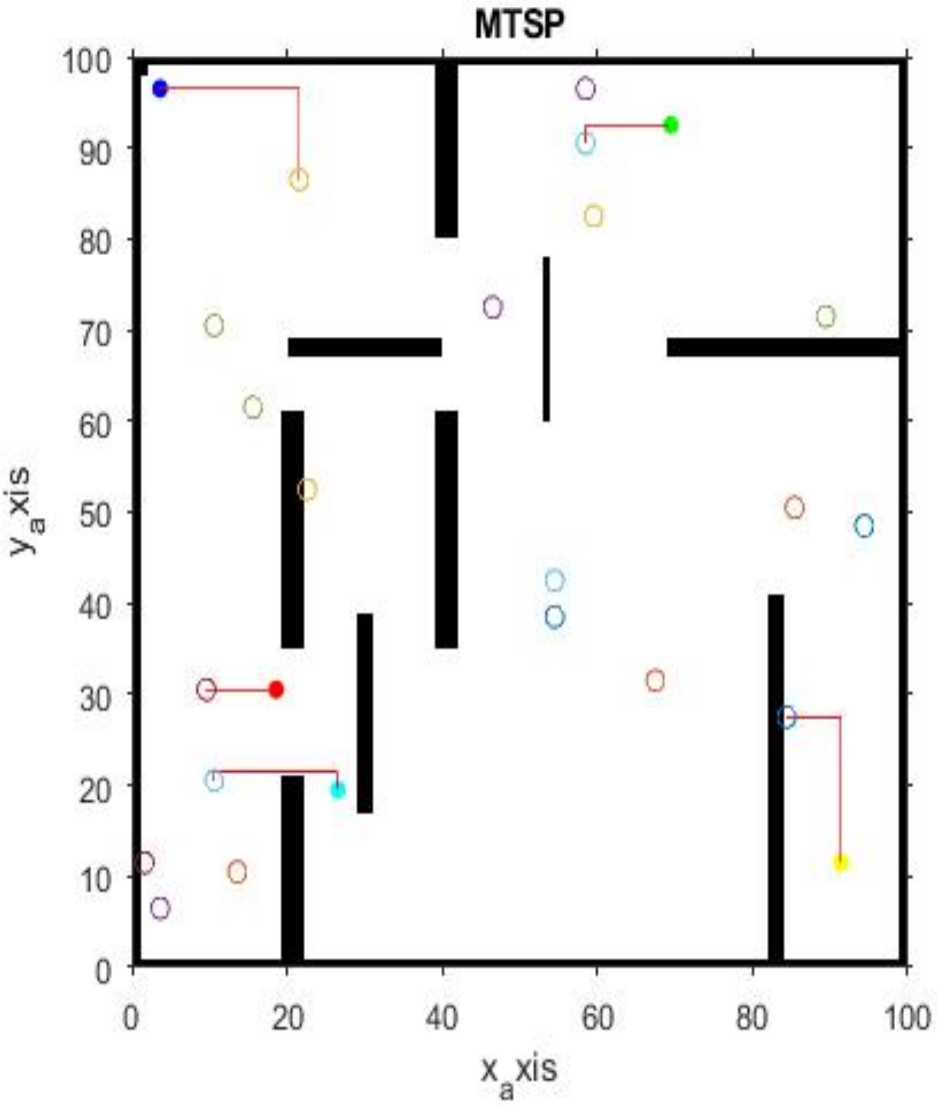

Figure 4.17: Time:0.371068 sec Tasks:5 Agents:20 Max cost:39.5979 Total cost:130.10764

Table 4.23: Tasks positions Large scale problem R>T

Tasks ID	X	Y
1	18.50000000000000	30.50000000000000
2	3.500000000000000	96.50000000000000
3	91.50000000000000	11.50000000000000
4	26.50000000000000	19.50000000000000
5	69.50000000000000	92.50000000000000

Table 4.24: Agents Large scale problem R>T

Agent ID	position	Assign task	Cost
1	[84.50000000000000;27.50000000000000]	3	32.5269119345812
2	[67.50000000000000;31.50000000000000]	[]	0
3	[59.50000000000000;82.50000000000000]	[]	0
4	[46.50000000000000;72.50000000000000]	[]	0
5	[10.50000000000000;70.50000000000000]	[]	0
6	[58.50000000000000;90.50000000000000]	5	18.3847763108502
7	[9.500000000000000;30.50000000000000]	1	12.7279220613579
8	[94.50000000000000;48.50000000000000]	[]	0
9	[85.50000000000000;50.50000000000000]	[]	0
10	[21.50000000000000;86.50000000000000]	2	39.5979797464467
11	[3.500000000000000;6.500000000000000]	[]	0
12	[15.50000000000000;61.50000000000000]	[]	0
13	[54.50000000000000;42.50000000000000]	[]	0
14	[1.500000000000000;11.50000000000000]	[]	0
15	[54.50000000000000;38.50000000000000]	[]	0
16	[13.50000000000000;10.50000000000000]	[]	0
17	[22.50000000000000;52.50000000000000]	[]	0
18	[58.50000000000000;96.50000000000000]	[]	0
19	[89.50000000000000;71.50000000000000]	[]	0
20	[10.50000000000000;20.50000000000000]	4	26.8700576850888

4.11 Comparative results between Combinatorial Auction and Iterative solution

a comparative study was made between Cluster Combinatorial Auction (C.C.A) and Iterative market (IT) to test the efficiency of IT compared with C.C.A for different scenario's in both small and large scale

1. Robots =Tasks
2. Robots >Tasks
3. Robots <Tasks

Table 4.25: C.C.A VS Iterative

scenario	Agents	Tasks	C.C.A Time sec	C.C.A Total Cost	C.C.A MAX Cost	IT Time sec	IT Total Cost	IT MAx Cost
1	6	6	0.116189	490.73210	127.2792	0.366044	541.643	120.2081
2	10	10	0.105823	492.149	98.9949	1.261149	458.205	74.953
3	10	7	0.057960	337.9970	100.4091	1.571339	337.9970	100.4091
4	20	6	0.096097	141.42	49.49	5.756	141.42	49.49
5	6	20	0.074526	599.776	142	1.471771	453.0538	106
6	7	30	0.259659	615	111	2.776726	557.1231	106
7	10	1000	2.053258	1554	204	failure	failure	failure

4.12 Discussion

The results show's that in simple market 4.4 the performance was not that good and the output solution for the tour had crosses on it that means the tour is not the shortest one that can be obtained.
For cluster combinatorial auction 4.5 that bid on the centroid of the group the results was not bad however to solve the tour the robot should start from the center of the group which is not logical how ever the algorithm show's a great ability on solving extra large scale problems 4.5.3
In the edited version of cluster combinatorial auction 4.6 the algorithm was really powerful and we implemented a path planing for it the performance was better than the old version and the path planing was working in the proper manner in both small and large scale . when implementing the Map 4.8 with obstacles the algorithm path planing was working and the obstacle avoidance gives a good results.

The algorithm of Cluster Combinatorial Auction (C.C.A) with edited version was good however the usage of k-means only once have a disadvantage which is the dependency on the first random centroids that is thrown in the map so to overcome such a problem we implemented an Iterative market and we make a comparative study in section 4.11.
The results of 4.11 show's that when Robots = Tasks the Iterative market (IT) outperformed C.C.A in both computation time and cost .
When Robots > Tasks the cost is the same between both algorithms however the computation time in IT is larger than C.C.A .
when the Robots < Tasks the IT outperformed C.C.A in both computation time and cost .
when extra large problems of Robots < Tasks the Iterative market (IT) failed to solve the problem.
From the results we recommend to use C.C.A in extra large problems and when Robots > Tasks.For other types of problems like in the scenario's of the study we recommend using Iterative market (IT)

Chapter 5

Conclusion

This study aims to find a solution for Multi-Robot Task Allocation (MRTA) problems. A Cluster Combinatorial Auction ,and Iterative market Algorithms were implemented in this study. K-means was implemented in both Market approaches for Grouping .A 2-opt heuristic was used in both Market approaches for solving the tour order to fit the objective function. The problem was formulated as instance of MTSPs to find the answer of a very important question

"Which Robot is responsible to handle which task such that the overall system performance of the team of Robots is optimized ?"

Results showed that Cluster Combinatorial Auction (C.C.A) outperformed Iterative Market in computation time however Iterative gives a better cost in the expanses of time.
From the results it is recommend to use Cluster Combinatorial Auction (C.C.A) in extra large problems and when Robots¿Tasks.For other types of problems like in the scenarios of the study presented in the thesis we recommend using Iterative market Iterative market (IT)

Chapter 6

Future Work

It is recommended to investigate different algorithms and include a heterogeneous Robots systems to accommodate more realistic scenarios. Also it is recommended to include the robot dynamics in the model to be more applicable in the real world.We may also implement intersection control in the future work.

Bibliography

[1] Veloso Manuela Stone, Peter. *Multiagent systems: A survey from a machine learning perspective*. Autonomous Robots, 8(3), pp.345-383, 2000.

[2] Burgard Wolfram Cremers Armin B. Fox Dieter Hofmann Thomas Schneider Frank E. Strikos Jiannis Thrun Sebastian Buhmann, Joachim. *The mobile robot RHINO*. AI Magazine, 16(2):3138,, 1995.

[3] Watanabe Ryujin Aoyama Chiaki Matsunaga Shinichi Higaki Nobuo Fujimura Kikuo Sakagami, Yoshiaki. *The intelligent ASIMO: system overview and integration*. In Proceedings of IROS02, pages 24782483, Lausanne, Switzerland, 2002.

[4] *http://marsrover.nasa.gov/home/index.htm*.

[5] M. Raibert. *the Rough-Terrain Quaduped Robot*. Proceedings of the 17th World Congress The International Federation of Automatic Control Seoul, 2008.

[6] Rusu R.B. Jones E.G. Marder-Eppstein E. Pantofaru C. Wise M. Msenlechner L. Meeussen W. Bohren, J. and S. Holzer. *Towards autonomous robotic butlers: Lessons learned with the PR2*. IEEE International Conference on Robotics and Automation (pp. 5568-5575). IEEE, 2011.

[7] Michael R. M. Jenkin Milios-Evangelos Wilkes David Dudek, Gregory. *A taxonomy for multiagent robotics*. Autonomous Robots, 3(4):375397, 1996.

[8] Vaughan Richard T Wawerla, Jens. *A fast and frugal method for team-task allocation in a multirobot transportation system*. In Proceedings of ICRA10, pages 14321437, 2010.

[9] Burgard Wolfram Kruppa Hannes Thrun-Sebastian Fox, Dieter. *A probabilistic approach to collaborative multi-robot localization*. Autonomous Robots, 8(3):325344, 2000.

[10] Bekey George A Roumeliotis, Stergios I. *Distributed multirobot localization*. IEEE Transactions on Robotics and Automation, 18(5):781795, 2002.

[11] Fregene Kingsley Parker Lynne E Madhavan, Raj. *Distributed heterogeneous outdoor multi-robot localization*. In Proceedings of ICRA02, pages 374381, Washington, DC, USA, 2002.

[12] Bahr Alexander Martinoli Alcherio Prorok, Amanda. *Low-cost collaborative localization for large-scale multi-robot systems*. In Proceedings of ICRA12, pages 42364241, 2012.

[13] *A survey and analysis of multi-robot coordination*.

[14] Hussein A. Khamis, A. and A. Elmogy. *Multi-robot task allocation: A review of the state-of-the-art. In Cooperative Robots and Sensor Networks 2015 (pp. 31-51)*. Springer, Cham, 2015.

[15] M De Longueville. *A course in topological combinatorics*. Springer Science Business Media, 2012.

[16] J.C.G. Higuera and G Dudek. *May. Fair subdivision of multi-robot tasks)*. In 2013 IEEE International Conference on Robotics and Automation (pp. 3014-3019). IEEE, 2013.

[17] H.W Kuhn. *The Hungarian method for the assignment problem*. Naval research logistics quarterly, 2(12), pp.83-97, 1955.

[18] C. Nam and D.A. Shell. *May. Assignment algorithms for modeling resource contention and interference in multi-robot task-allocation*. In 2014 IEEE International Conference on Robotics and Automation (ICRA) (pp. 2158-2163). IEEE, 2014.

[19] V. Lattarulo and G.T Parks. *June. A preliminary study of a new multi-objective optimization algorithm*. In 2012 IEEE Congress on Evolutionary Computation (pp. 1-8). IEEE, 2012.

[20] Sherali H.D. Sarin, S.C. and A Bhootra. *New tighter polynomial length formulations for the asymmetric traveling salesman problem with and without precedence constraints*. Operations research letters, 33(1), pp.62-70, 2005.

[21] T. Bektas. *The multiple traveling salesman problem: an overview of formulations and solution procedures* . Omega,34(3), pp.209-219., 2006.

[22] Norman Biggs, E Keith Lloyd, and Robin J Wilson. *Graph Theory, 1736-1936*. Oxford University Press, 1986.

[23] Peter J Kolesar. A branch and bound algorithm for the knapsack problem. *Management science*, 13(9):723–735, 1967.

[24] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[25] Shen Lin. Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44(10):2245–2269, 1965.

[26] Murugappan Elango, Subramanian Nachiappan, and Manoj Kumar Tiwari. Balancing task allocation in multi-robot systems using k-means clustering and auction based mechanisms. *Expert Systems with Applications*, 38(6):6486–6491, 2011.

[27] Sahar Trigui, Anis Koubâa, Omar Cheikhrouhou, Basit Qureshi, and Habib Youssef. A clustering market-based approach for multi-robot emergency response applications. In *2016 International conference on autonomous robot systems and competitions (ICARSC)*, pages 137–143. IEEE, 2016.

[28] Bradford Gregory John Heap and Maurice Pagnucco. Repeated sequential auctions with dynamic task clusters. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[29] Omar Cheikhrouhou, Anis Koubâa, and Hachemi Bennaceur. Move and improve: A distributed multi-robot coordination approach for multiple depots multiple travelling salesmen problem. In *2014 IEEE international conference on autonomous robot systems and competitions (ICARSC)*, pages 28–35. IEEE, 2014.

[30] Elad Kivelevitch, Kelly Cohen, and Manish Kumar. A market-based solution to the multiple traveling salesmen problem. *Journal of Intelligent & Robotic Systems*, 72(1):21–40, 2013.

[31] Keld Helsgaun. Lin-Kernighan heuristic for solving the traveling salesman problem., 2017.

[32] Reid Simmons, David Apfelbaum, Wolfram Burgard, Dieter Fox, Mark Moors, Sebastian Thrun, and Håkan Younes. Coordination for multi-robot exploration and mapping. In *Aaai/Iaai*, pages 852–858, 2000.

[33] Changyun Wei, Koen V Hindriks, and Catholijn M Jonker. Auction-based dynamic task allocation for foraging with a cooperative robot team. In *European Conference on Multi-Agent Systems*, pages 159–174. Springer, 2014.

[34] Ioannis Rekleitis, Ai Peng New, Edward Samuel Rankin, and Howie Choset. Efficient boustrophedon multi-robot coverage: an algorithmic approach. *Annals of Mathematics and Artificial Intelligence*, 52(2-4):109–142, 2008.

[35] Alan C Schultz, Lynne E Parker, and Frank Schneider. *Multi-robot systems: from swarms to intelligent automata*. Springer, 2002.

[36] Charles E Pippin and Henrik Christensen. A bayesian formulation for auction-based task allocation in heterogeneous multi-agent teams. In *Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR II*, volume 8047, page 804710. International Society for Optics and Photonics, 2011.

[37] Maja J Mataric and Gaurav S Sukhatme. Task-allocation and coordination of multiple robots for planetary exploration. In *In Proceedings of the 10th International Conference on Advanced Robotics*. Citeseer, 2001.

[38] Brian P Gerkey and Maja J Mataric. Sold!: Auction methods for multirobot coordination. *IEEE transactions on robotics and automation*, 18(5):758–768, 2002.

[39] Maitreyi Nanjanath and Maria Gini. Repeated auctions for robust task execution by a robot team. *Robotics and Autonomous Systems*, 58(7):900–909, 2010.

[40] José Guerrero and Gabriel Oliver. Multi-robot task allocation strategies using auction-like mechanisms. *Artificial Research and Development in Frontiers in Artificial Intelligence and Applications*, 100:111–122, 2003.

[41] Tuomas Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial intelligence*, 135(1-2):1–54, 2002.

[42] Luke Hunsberger and Barbara J Grosz. A combinatorial auction for collaborative planning. In *Proceedings fourth international conference on multiagent systems*, pages 151–158. IEEE, 2000.

[43] David C Parkes and Lyle H Ungar. Iterative combinatorial auctions: Theory and practice. *AAAI/IAAI*, 7481, 2000.

[44] Arne Andersson, Mattias Tenhunen, and Fredrik Ygge. Integer programming for combinatorial auction winner determination. In *Proceedings Fourth International Conference on MultiAgent Systems*, pages 39–46. IEEE, 2000.

[45] Tuomas Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial intelligence*, 135(1-2):1–54, 2002.

[46] Sven De Vries and Rakesh V Vohra. Combinatorial auctions: A survey. *INFORMS Journal on computing*, 15(3):284–309, 2003.

[47] Arne Andersson, Mattias Tenhunen, and Fredrik Ygge. Integer programming for combinatorial auction winner determination. In *Proceedings Fourth International Conference on MultiAgent Systems*, pages 39–46. IEEE, 2000.

[48] David C Parkes and Lyle H Ungar. Iterative combinatorial auctions: Theory and practice. *AAAI/IAAI*, 7481, 2000.

[49] Vedran Kordic. *Cutting edge robotics*. I-Tech, 2005.

[50] Michael Otte, Michael J Kuhlman, and Donald Sofge. Auctions for multi-robot task allocation in communication limited environments. *Autonomous Robots*, pages 1–38, 2019.

[51] Jonas Lundgren. *TSPSEARCH Heuristic method for Traveling Salesman Problem (TSP)*. https://www.mathworks.com/products/matlab.html, 2012.

[52] Michael Otte, Michael J Kuhlman, and Donald Sofge. Auctions for multi-robot task allocation in communication limited environments. *Autonomous Robots*, pages 1–38, 2019.