

# Simulated annealing for multi-robot hierarchical task allocation with MinMax objective

Alejandro R. Mosteo and Luis Montano

*Departamento de Informática e Ingeniería de Sistemas. Universidad de Zaragoza*

*Instituto de Investigación en Ingeniería de Aragón, I3A*

*c/ María de Luna 1, E 50018 Zaragoza, Spain*

*{amosteo, montano}@unizar.es*

**Abstract**—In this paper we study algorithms for minimizing the worst-case cost of any agent in a multi-robot team in time-critical missions. We propose a generalized model for flexible mission planning, using hierarchical task networks as the planning framework, and the multiple traveling salesman problem as the cost model for task allocation. Two approximated solutions are provided and compared for this NP-Hard problem, one based in current research in market-based techniques, and another one based in the optimization technique known as simulated annealing. We provide simulation results which back the model described and the proposed algorithms.

## I. INTRODUCTION

Cooperative robotics is one of the hottest topics in current research, as shown by the growing number of articles published every year. Amongst all the fields which can benefit from multi-robot cooperation, the aid in civil tasks is a promising one (USAR<sup>1</sup> being a field on its own). Examples of work in this respect are exploration in structured [1] or unstructured [2] environments, autonomous patrolling of known [3] or unknown areas [4], victim detection [5] and sensor networks [6] for quick response in emergency situations.

The EXPRES<sup>2</sup> project is framed around the USAR principles and aims to provide robotic help in distress situations where human intervention is dangerous. The objective is to build a cooperative robot team which can perform complex tasks in civil aid operations with a high degree of autonomy. Examples of missions that are being studied in the EXPRES project are victim location in road tunnel accidents (with emphasis in maintaining a completely connected mobile MANET<sup>3</sup> [7]) and the one which will be examined in this paper: complete exploration of a parking or urban area for quick location of interesting vehicles. The motivation for such a task is that it is a common tactic of some terrorist groups in our country to advertise the planting of a bomb car at some location, often supplying critical data about the vehicle such as the plaque or the model. The rationale seems to be that this *modus operandi* creates distress amongst the civil population, but only police lives are put at risk in the deactivation mission.

Since we are dealing with potentially time-critical interventions, within this project we are interested in task allocation

methods that improve the mission finalization time over other metrics such as resource consumption or individual performance. Hence the use of the *MinMax* criterion:

$$A_{\text{Opt}} = \arg \min_A (\max (C_r(A))), r \in \{r_1, \dots, r_n\} \quad (1)$$

where  $A$  is a certain allocation of tasks to robots and  $C_r$  is the cost (time in our case) incurred by a robot  $r$  of the robot team  $R$  given the task allocation.

The paper is structured as follows. We start with a background section where we will comment related bibliography and the techniques being used for our solution. We then propose our problem and solution model and two different implementations to conclude with simulation results and future work statement.

## II. BACKGROUND

The problem of planning and task allocation in multi-robot teams is well expressed in [8] by the two questions:

- What do we do?
- Who does what?

For clarity, the first question refers to *planning*, while the second refers to *allocation*. The introduction of the cited paper provides a more in-depth description of the importance of such questions, and some explanations of why adopting a solution that isolates each one or that couples them both can have a substantial impact on the quality of the solution.

### A. Hierarchical Task Networks

HTNs have been proposed for more flexible and powerful planning in numerous occasions [8]–[11] as an answer for the planning problem. Their expressive power allows high-level thinking and visualization, which is also useful for humans, and representation of logical relations [11], amongst other advantages.

HTNs are trees of tasks (Fig. 1), where a node represent a task to be performed or a logical relation. Leaves are *primitive* tasks, that is, tasks that a robot can directly perform. Inner nodes represent tasks that robots do not know how to execute. These tasks, often called *compound*, *abstract* or *inner* tasks, must be recursively decomposed into lower level tasks, until primitive tasks are found.

<sup>1</sup>Urban Search And Rescue

<sup>2</sup>EXPLORATION & RESCUE. This project is partially funded by the Spanish MCYT-FEDER project DPI2003-07986.

<sup>3</sup>Mobile ad-hoc network

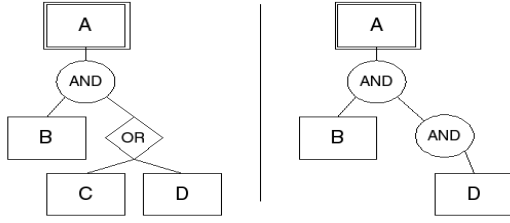


Fig. 1. Left: An expanded HTN. Right: one of the two possible executable plans.

AND nodes are fulfilled when all its children are completed, whereas OR nodes need only a branch to be executed. Hence, the root node represents the whole mission. An *executable plan* is a fully expanded task tree where OR relations have been removed by choosing one of its branches.

The use of OR nodes introduces an exponential problem [11], since the number of solutions is  $d_1 \times d_2 \times \dots \times d_o \geq 2^o$ , being  $d_i$  the number of children of each one of the  $o$  OR nodes. If we have no further constraints [12] that can aid in the pruning of solutions we are facing a new challenge, since the random choosing of an executable plan can be suboptimal.

### B. The Multiple Traveling Salesmen Problem (MTSP)

The traveling salesman problem (TSP) aims to find the shortest route that visits without repetition all of a set of cities. As we will later explain, we use the MTSP problem as the underlying model of our allocation strategy. This is a generalization of the TSP with multiple travelers. When the costs are not homogenous for all the travelers, this is called the multiple cost (m-cost) MTSP [13]. The MTSP allows two optimization criteria: minimizing the sum of costs of travelers [14], also known as *TotalSum* criterion, or minimizing the worst of the costs, or *MinMax*, in which we are interested. This was used in [15] for route planning but not as a general task framework.

The *MinMax* criterion has been studied in research unrelated to mobile robotics in [16], albeit not for the m-cost variation. In [13] solutions for two travelers were given and is noted that there seems to be not prior research in the m-cost variant.

### C. Coverage

We will leverage past work on provably complete coverage [17], [18] for the generation of coverage paths composed of straight segments for some of our simulation scenarios. More general forms of complete coverage have been recently addressed in, for example, [19], [20]. For the last of our simulation scenarios we will use an urban city area, where complete coverage is obtained when all streets have been traversed by some robot.

## III. PROBLEM STATEMENT

The general problem we want to solve is the generation of plans and allocation of tasks to robots in such a fashion that the solution is *MinMax* optimal; that is, the cost of the worst robot is minimized, bearing in mind that

- 1) The plan generation can involve an exponential number of executable plans.
- 2) The optimal assignment of any of these plans is an NP-Hard problem.

The particular problem used as simulation example is the complete exploration of a parking or urban area in minimum time by a robotic team, including to find the best entry point in the area to be explored for each one of the available robots.

The approach we propose in this paper is the use of *Hierarchical Task Networks* (HTN) for the modelling of the high level problems and generation of executable plans, the *Multiple Traveling Salesman Problem* problem as the cost model, and the use of simulated annealing to perform the minimization in real time. We will also provide a market-based solution for comparison. Our solutions, like those in [8], [11], aim to solve both problems of plan generation and task allocation simultaneously.

For the problem at hand, we are making the following assumptions:

- We have a floorplan of the area to be explored. This could be obtained from an aerial photography, a database of critical targets or the national body in charge of buildings and roads.
- The robots are always connected to the MANET, so messages can reach any robot. We can note here that another goal of the EXPRES project is the building of a low-frequency modem which would enable communications through usually isolating materials —e.g. tunnel or basement walls.
- The robots are capable of localization and navigation and carry appropriate sensors for the mission at hand.

### A. Hierarchical tasks

In figure 2 we show an schema of the tasks used. We have a first set of tasks related to the exploration problem:

- *Main exploration*: This represents the exploration of the entire area. This task is expanded using the cited algorithms in a set of straight paths or, in the case of streets, each street will suppose a task. The rationale is that a robot, while walking down such a straight path, is capable of detecting with its sensors some relevant features of a nearby vehicle.

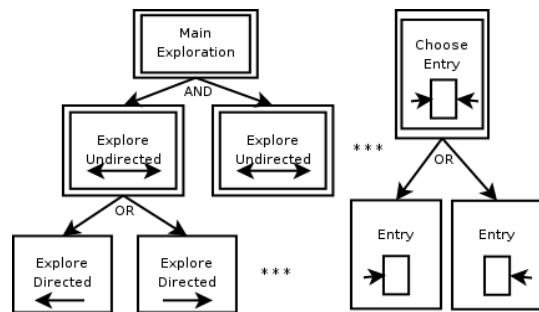


Fig. 2. Example of partial mission task tree.

- *Explore Undirected Segment*: These are each of the segments result of the decomposition. Since we do not want to constraint our possibilities, we assume that a robot can explore them in any of the two ways. Each of these tasks are expanded as two OR-related *Directed Segment* tasks.
- *Explore Directed Segment*: This primitive task means that a robot must cover the segment in an explicit direction.

We also have tasks whose purpose is to find the best entry point for the robots if the area has several of them, so they can be deployed optimally.

- *Choose Entry Point*: This compound task is a collection of all the allowed entry points. It is trivially expanded as several OR-related nodes of the following class. There are as many as robots.
- *Use Entry Point*: This primitive task simply means that the robot has to be deployed at the given Entry Point.

The entry tasks carry a constraint so they must be scheduled as the first one in every robot. In the market-based approach these tasks are not used, since the robots need to know where they are to be able to bid.

### B. The cost model

In a sequential task execution model, the analogy with the MTSP problem is as follows: the cost of traveling from a city to another is the cost of performing a task having previously performed another one. Solving this MTSP problem gives in effect the solution to the task allocation problem. To define the problem, the robots have to broadcast a cost-matrix of size  $|T_p|^2$  to the solvers, being  $T_p$  the number of primitive tasks. This is enough to evaluate any allocation. We are thus using the most general variety of the problem: m-cost (since robots can be heterogeneous or have different sensed data) with asymmetric costs.

Our most relevant cost is that of covering a segment, which is computed as the time required to travel the segment plus the time needed to arrive to the starting pose.

## IV. MARKET-BASED SOLUTION

There are plenty of results backing the use of market-based solutions in multi-robot teams ([2], [21], [22]). Furthermore, there's also a body of knowledge on the use of HTNs coupled with auction based solutions to increase the quality of solutions [11], [12]. These solutions, while performing better than more simplistic approaches, can still be suboptimal since in essence the problem to be solved is NP-Hard [14] and some compromises have to be made.

We will use a solution akin to that in [11] for comparison with our annealing proposal, with a notable modification. We have seen that many market solutions say that a robot, by trying to maximize its profits, is maximizing the whole team profit [2], [11]. In practice, this means that the *sum* of profits is maximized. Speaking in term of costs, the sum of costs is minimized (*TotalSum* criterion). This manifest itself in situations like the one shown in Fig. 3 where two robots are visiting spots of interest. Let us suppose R1 is already

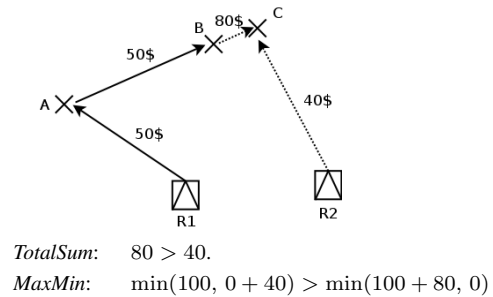


Fig. 3. The *TotalSum* criterion would give task C to R1, even if R2 is idle, whereas the *MaxMin* criterion will award C to R2. Note that the *MaxMin* criterion is the equivalent in terms of profit to the *MinMax* in terms of cost.

committed to tasks A, B when C is auctioned. Even if R2 is idle, R1 profit for C is greater than that of R2, and R1 will win the task. This effectively serves to optimize evaluation criteria such as the one used in [2] favored by a *TotalSum* criterion, but will not provide the minimization we are interested in.

We thus propose a modification for the auctioning algorithm which, in practice, turns the marketplace into an heuristic for the *MinMax* criterion. We give it in terms of costs: Now, a robot starts an auction broadcasting the tuple  $\{t, C_a\}$  where  $t$  is the auctioned task and  $C_a$  is the auctioneer plan cost *without* removing  $t$  from its plan. Listening robots will compute his best plan cost  $C_r$  which includes  $t$ . Robots having  $C_r < C_a$  will reply using  $C_r$  as bid. Finally, the auction is awarded to the bidder with minimum  $C_r$ . Since  $C_r < C_a$  and also  $C_a$  will decrease after removing  $t$  from the auctioneer plan, in practice we are reducing the maximum cost.

In our implementation, robots auction subtrees entirely owned by them. Since in this particular problem all OR nodes consist of single primitive tasks, the auctions will always be for OR nodes. A bidder will consider each decomposition of the OR task, and bid on the best one for his current plan. The strategies tried to insert a new task are best insertion point, optimal TSP solution when the plan has less than 20 tasks, and complete greedy replanning with decomposition and selection of the next best task. Finally, the *Choose Entry Point* subtrees are not used in this approach, since robots need to know where they are to properly bid. Hence, this solution does not address this part of the problem.

## V. SIMULATED ANNEALING SOLUTION

Simulated annealing [23] is an optimization technique commonly used for problems which are intractable by traditional means and which fits well with combinatorial problems. It has been successfully used in the solving of traditional NP-Hard problems like the Traveling Salesman Problem [24] and also in robotic path planners [15].

Simulated annealing mimics the metallurgy process of annealing, which involves heating and cooling cycles. In simulation, the current solution is replaced by some variation of it, which will be accepted or rejected with a certain probability  $p(C_{old}, C_{new}, temperature)$ . This probability depends on the

difference of quality of the solutions and on the current temperature. When temperature reaches zero, only better solutions are accepted (greedy behavior), while during hot stages worse solutions can replace the current one.

A fundamental element of the annealing problem is the generation of neighbor solutions. We can leverage this point not only to solve the MTSP problem, but also to explore alternative plan expansions, hence we are simultaneously solving the planning and allocation problems. As a consequence, this is also the key ingredient of our approach to determine the best entry point for each robot. Another advantage of the annealing approach is that we can address the most complex and general m-cost asymmetric MTSP model with arbitrary constraints on the task ordering or execution time window. This is naturally handled avoiding the generation of invalid solutions when feasible or, failing that, by giving infinite cost to any solution violating constraints.

While the annealing process is centralized in principle, we think that in our current project this can be used to our advantage by running an identical algorithm with different random seeds in every robot. The required information for the algorithm can be propagated to the control center and to every robot using the MANET. (Results in geographic routing [25] suggest that this can be achieved with latency costs  $O(n \log n)$  and resource costs  $O(n)$  being  $n$  the number of nodes.) Furthermore, since it is critical to keep track of the global mission state at the control center, it is moot not to take advantage of a centralized algorithm there.

In our approach, the execution of the mission is divided in two stages: a startup stage, where an initial solution is computed during a reasonably short period compared to the expected total mission time, and the execution stage.

#### A. Startup stage

While we can count on a precalculated database of relevant urban locations (for example, public parkings in principal cities), in case the actual location is not available the startup stage will account for this lack of precalculated data, once the relevant blueprints are obtained. The human operator must provide the possible entry points for the robots. Meanwhile, a set of segments will be computed. The task tree is transmitted to the robots, which in turn answer with the cost-matrix for the primitive tasks. With this information, the planning can begin and the algorithm is run in every computing device available.

#### B. Execution stage

Once a prudential time has elapsed (in our experiments we give one minute for the startup stage), robots are committed to the best solution found and start task execution. However, the annealing algorithm can continue, taking into account the current robot commitments. In this way, better plans can be found at any moment. We can also benefit from the decreasing size of the remaining problem, which means that the final stages will be probably solved to optimality. Also any deviation in the sensed environment from the *a priori* data can be integrated and taken into account.

In essence, each robot and connected resource can continually work in the search of a better solution. It is not critical to communicate complete solutions until a robot is about to finish its current task, so network usage can be minimized by just broadcasting the cost of the best solution known.

#### C. Neighborhood generation

In our implementation we have several kinds of solution modifications that produce a new neighbor, as follows:

- Random reinsertion.  
This movement will randomly choose a task from any agent, remove it from the owner plan and insert it again at a random point in the plan of a random agent.
- Random plan re-expansions.  
In this movement, an OR branch is chosen and replaced randomly by one of its siblings.
- Guided movements.  
Very much like the previous movements, but in this case we choose a task or branch from the most costly agent plan, in a guided attempt to improve the worst case.
- Heuristics.  
In this class of movements we have two more expensive TSP heuristics, based in greedy insertion [26].

We have found that this mix of purely random movements and heuristic movements works well. In one hand, pure random movements allow the exploration of far apart areas in the solution space. In the other hand, guided and heuristic movements quickly find good solutions in a given local area. Heuristics are expensive, so we give them a very small probability of being performed. We have aimed for quickly computable movements, since this allows the exploration of a larger solution space. Our implementation using the Gnat Ada [27] compiler explores around 2000-6000 solutions per second in a standard PC a year old, depending on the problem size. Since this computation could be run in parallel in every robot, we can benefit of replicated computation.

## VI. EXPERIMENTS

In this section we will present the different results obtained by the two proposed solutions. We have used three simulated scenarios of different complexity. In every one of them, the tasks for the market planner simulator are initially auctioned in random order. After completing the auctioning, enough full rounds of auctions are run by each robot until no plan improvements occur. In the annealing planner, we simulate the complete running period, taking into account that a committed task cannot be removed from the robot that is executing it. Multiple planners are not run in parallel, but a single one is used for our simulations. The cooling schedule is an exponential one, which is cyclically applied over the complete simulation time. The initial solution is constructed using a greedy allocation after a random plan expansion.

The first scenario is a small parking roughly inspired in the one of our workplace, which has 12 lanes to explore as seen in Fig. 4. This gives  $2^{12}$ , that is 4096, possible executable plans. Two entry points are considered for this parking, one

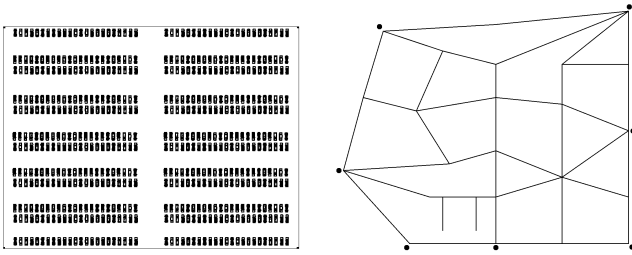


Fig. 4. Left: The small parking scenario. Right: The city schema used for the large scenario.

at the center top and another one at the center bottom. The second scenario is a bigger model with the same structure, with a total of  $2^{32}$  possible plans, which we consider big enough to be regarded as intractable by full enumeration of solutions. The third scenario (Fig. 4) is a model of the streets in a small area of our city, with a total of  $2^{43}$  possible plans, and some additional difficulties like dead-ends and a more irregular disposition of the segment tasks. This scenario has seven possible entry points. We shall call these three scenarios *small*, *medium* and *large* for short. Note that even if the two parking scenarios have apparent symmetries in the disposition of tasks, there are small differences that suppress them. We have done this in purpose to avoid multiplicity of solutions and problem simplification, although on a real implementation we would of course want to leverage these when possible.

Fig. 5 shows the minimum, maximum, median and quartile costs over 10 runs of each algorithm with a team of four robots, which is the size of our real robotic team. The *SA* column gives the cost computed by the annealing planner having to chose entry points; the *SA<sub>Fix</sub>* gives the cost computed by the annealing planner using from start the best entry points previously found. The *Market* column gives the *MinMax* cost computed by the market simulation with the same entry points as *SA<sub>Fix</sub>*. Finally, the *Opt* column gives the optimal solution cost, which we know for the parking scenarios. These costs do not include the time given for the startup stage or the auction rounds, but are the solutions computed by the algorithms.

As we can see, the simulated annealing solutions are consistently better than the market-based ones, which is to be

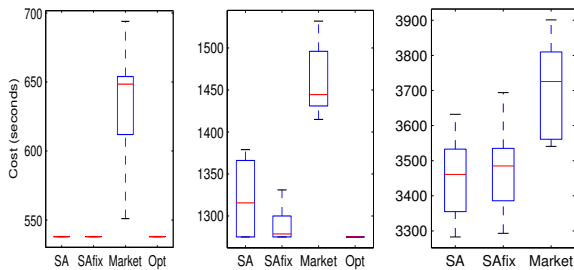


Fig. 5. Costs of solutions for the 12-, 32- and 43-lane problems.

expected from a centralized theoretically complete<sup>4</sup> algorithm over an heuristic one. *SA* solutions are within a 0%-4% of the optimal known solution, whereas the market ones are about 15%-18% off which is, nonetheless, a good result, specially for problems of this complexity. Note that the small scenario has been solved to optimality in all runs by the annealing algorithm, while only a few times in the medium scenario and probably none in the large one.

Since we don't know the optimal solution of the urban scenario, we can not give assurances on quality solution, but again we see a better performance from the annealing solver. In table I we give the ratio mean/best for each algorithm and problem size. These ratios also indicate a decrease in the quality of solutions when the problem grows in size for the annealing solver, whereas the gap with the market-based solution shrinks. This could indicate that for really big problems it can be enough the use of a greedy heuristic, or that the annealing solver does not really perform much better than such an heuristic.

TABLE I  
RATIO OF MEAN TO BEST SOLUTION

Scenario	SA	SA <sub>Fix</sub>	Market
Small	1.00	1.00	1.18
Medium	1.04	1.02	1.15
Large	1.05	1.06	1.13

Table II shows the quality of solutions for the different scenarios when considering different number of robots for the *SA* algorithm. The values are the ratio average/best solution cost. Columns show the already noted decrease in quality of solutions as the problem gets larger, whereas rows highlight that adding robots increases the number of solutions and so the performance degrades over a larger solution space.

TABLE II  
TEAM SIZE & SA ALGORITHM

Scenario	2 robots	4 robots	8 robots
Small	1.00	1.00	1.00
Medium	1.00	1.04	1.05
Large	1.03	1.05	1.06

In Fig. 6 we can see the progress of the best solution cost obtained by the *SA* algorithm in a typical run. We observe that in the first seconds there is a quick evolution, which is to be expected since the first plan was chosen at random and can be highly suboptimal. After that, slow improvements can be obtained over all the mission duration, and specially when the problem size reduces in the final period.

## VII. CONCLUSIONS AND FUTURE WORK

The proposed simulated annealing algorithm simultaneously solves the planning and task allocation problems. It performs well in all tried examples, often finding the optimal solution for small-medium sized problems. This is not so common

<sup>4</sup>with infinite running time.

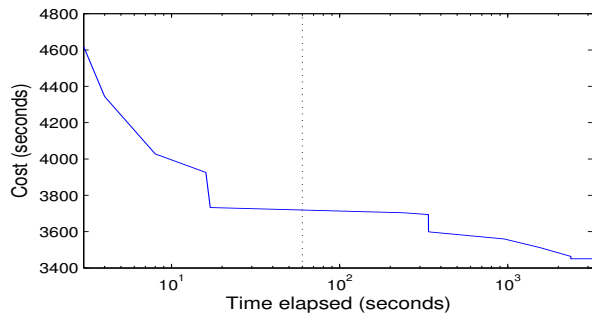


Fig. 6. Evolution of the best solution found over a complete run of the algorithm in the large parking problem with 4 robots. The vertical line indicates execution stage start at second 60. Better solutions found after this point can still be applied.

in larger problems, but it still outperforms the market-based solution for the examples studied. This is to be expected from a centralized algorithm. We believe that a replicated centralized algorithm is desirable in a critical mission, where we can not afford to distribute the information in the system so that losing a part of it would mean an irreparable loss —and failure.

We have also shown that the annealing solver is well fit for tackling the most general MTSP variety, including complex constrains in the task model, such as tasks that need to be performed at certain points of the mission. This is naturally handled in the neighbor solution generation by means of infinite costs or avoiding the generation of bad solutions when feasible.

Our market-based implementation illustrates that the *Min-Max* criterion on auction-based solutions is a viable alternative when we want to retain the advantages of market approaches. This kind of solutions can be used when we are constrained on resources due to other computations being carried in the robots, or for very quick obtention of plans when there is little time to react, since the annealing algorithm is slower in giving its best solutions, or when true distribution is desired.

Future plans include the investigation of the *MinMax* criterion in problems with reduced a priori information, the annealing performance with more complex plan trees and larger numbers of primitive tasks and also with tightly constrained plans such as those needed in coordinated surveillance missions. Finally, experimental validation on our team of robots in collaboration with the MANET research group at our university is also in the near horizon.

#### ACKNOWLEDGMENT

The authors thank Norbert Ascheuer and Christoph Helmberg for his insightful help with the MTSP problem.

#### REFERENCES

- [1] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 376–378, 2005.
- [2] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," in *Proceedings of the International Conference on Robotics and Automation*, 2002.

- [3] B. P. Gerkey, S. Thrun, and G. Gordon, "Visibility-based pursuit-evasion with limited field of view," in *National Conference on Artificial Intelligence (AAAI 2004)*, July 2004.
- [4] S. Sachs, S. LaValle, and S. Rajko, "Visibility-based pursuit-evasion in an unknown planar environment," *The International Journal of Robotics Research*, vol. 23, no. 1, pp. 3–26, January 2004.
- [5] S. Burion, "Human detection for robotic urban search and rescue," Master's thesis, Carnegie Mellon University, 2004.
- [6] V. Kumar, D. Rus, and S. Singh, "Robot and sensor networks for first responders," *IEEE Pervasive Computing*, pp. 24–33, October 2004.
- [7] I. Chlamtac, M. Conti, and J. J.-N. Liu, "Mobile ad hoc networking: imperatives and challenges," *Ad Hoc Networks*, vol. 1, no. 1, pp. 13–64, 2003.
- [8] R. M. Zlot and A. T. Stentz, "Complex task allocation for multiple robots," in *Proceedings of the International Conference on Robotics and Automation*. IEEE, April 2005.
- [9] P. Stone and M. M. Veloso, "Task decomposition and dynamic role assignment for real-time strategic teamwork," in *Agent Theories, Architectures, and Languages*, 1998, pp. 293–308.
- [10] T. Belker, M. Hammel, and J. Hertzberg, "Learning to optimize mobile robot navigation based on htn plans," 2003.
- [11] R. M. Zlot and A. T. Stentz, "Market-based multirobot coordination using task abstraction," in *International Conference on Field and Service Robotics*, July 2003.
- [12] L. Hunsberger and B. J. Grosz, "A combinatorial auction for collaborative planning," in *Proceedings of the Fourth International Conference on MultiAgent Systems (ICMAS'00)*. Washington, DC, USA: IEEE Computer Society, 2000, p. 151.
- [13] C. Helmberg, "The m-cost ATSP," in *7th International IPCO Conference on Integer Programming and Combinatorial Optimization*. London, UK: Springer-Verlag, 1999, pp. 242–258.
- [14] M. Lagoudakis, M. Berhault, S. Koenig, P. Keskinocak, and A. J. Kleywegt, "Simple auctions with performance guarantees for multi-robot task allocation," in *IROS'04, IEEE/RSJ, Ed.*, vol. 1, 2004, pp. 698–705.
- [15] B. Brummit and A. T. Stentz, "Grammps: A generalized mission planner for multiple mobile robots," in *Proceedings of the IEEE International Conference on Robotics and Automation*, May 1998.
- [16] P. M. França, M. Gendreau, G. Laporte, and F. M. Müller, "The m-traveling salesman problem with MinMax objective," *Transportation Research*, vol. 29, pp. 267–275, 1995.
- [17] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon decomposition," in *International Conference on Field and Service Robotics*, 1997.
- [18] D. T. Latimer, IV, S. Srinivasa, V. L. Shue, S. Sonne, H. Choset, and A. Hurst, "Towards sensor based coverage with robot teams," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, May 2002.
- [19] S. S. Ge and C.-H. Fua, "Complete multi-robot coverage of unknown environments with minimum repeated coverage," in *Proceedings of the IEEE International Conference on Robotics and Automation*, IEEE, Ed., 2005, pp. 727–732.
- [20] F. Zhang, W. Chen, and Y. Xi, "Improving collaboration through fusion of bid information for market-based multi-robot exploration," in *Proceedings of the IEEE International Conference on Robotics and Automation*, IEEE, Ed., 2005, pp. 1169–1174.
- [21] B. P. Gerkey, "On multi-robot task allocation," Ph.D. dissertation, University of Southern California, August 2003.
- [22] B. P. Gerkey and M. J. Matarić, "Sold!: Auction methods for multi-robot coordination," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, Oct. 2002.
- [23] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 4598, no. 220, pp. 671–680, May 1983.
- [24] V. Cerny, "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm," *Journal of Optimization Theory and Applications*, vol. 45, no. 1, pp. 41–51, 1985.
- [25] I. Stojmenovic, "Geocasting with guaranteed delivery in sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 29–37, December 2004.
- [26] G. Reinelt, *The traveling salesman: computational solutions for TSP applications*, ser. Lecture notes in computer science. Springer, 1994, vol. 840.
- [27] "Ada libre site." [Online]. Available: <https://libre2.adacore.com/>