The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Concurrent Tree Traversals for Improved Mission Performance under Limited Communication Range

Alejandro R. Mosteo, Luis Montano

Abstract— **In previous work we presented a multi-robot strategy for routing missions in large scenarios where network connectivity must be explicitly preserved. This strategy is founded on the traversal of path trees in such a way that connectivity to a static control center is always maintained, while ensuring that any target that is reachable by a chain consisting of all robots is eventually visited. In this work we improve the strategy performance by extending its sequential one-task-at-a-time execution approach with concurrent execution of tasks. We demonstrate that the general problem is NP-hard and offer several heuristic approaches to tackle it. We study the improvements that these heuristics can offer in regard to several important variables like network range and clustering of targets, and finally compare their performance over the optimal solutions for small problem instances. In summary, we offer a complete characterization of the new concurrent capabilities of the** CONNECTTREE **strategy.**

I. INTRODUCTION

Many application domains involve tasks of visiting places of interest using robots. The problem of routing a team over appropriate paths with the goal of visiting all target locations (corresponding to specific tasks) is known as *multi-robot routing*. It is known that solving multi-robot routing optimally is an NP-hard problem for many interesting team objectives [1]: minimizing total mission cost (energy), minimizing total mission time (makespan), and minimizing average service time (latency).

In many interesting real-world applications network connectivity must be explicitly considered and preserved. While some domains may tolerate temporary losses of communication between team members, in others it is mandatory to maintain connectivity at all times: monitored surveillance, rescue missions, tightly coordinated cooperation, etc. These include domains where uninterrupted contact to a stationary monitoring base or command center is commonly required.

Mobile robots typically use an ad-hoc wireless connection to communicate. Network integrity is maintained as long as any robot can reach any other robot either directly or by relaying messages through some other robot(s). Adding such limited communication constraints to multi-robot routing makes the problem even harder. Furthermore, real signals in populated scenarios exhibit a complex behavior caused by reflections, multi-paths, scattering and interferences, among other factors [2]. This makes necessary the use of careful deployment strategies that measure signal quality instead of relying on expected ranges of operation. We presented one such strategy in [3] under the name of CONNECTTREE, which is based on the traversal of a tree of paths with the control base at the root and targets at the leaves.

In this paper we enhance this strategy by analyzing the implications of concurrent tree traversal while preserving the advantageous features of the basic sequential algorithm, which only attempted to visit one target at a time. To do so, in Section II we review related work and revisit the most relevant principles of CONNECTTREE past work. Section III provides a formal definition of the multi-robot routing problem. Section IV studies the properties of the new problem. Section V describes algorithms to approximate this complex optimization challenge, while statistical results are shown and discussed in Section VI. Finally, we discuss future work and conclude in Section VII.

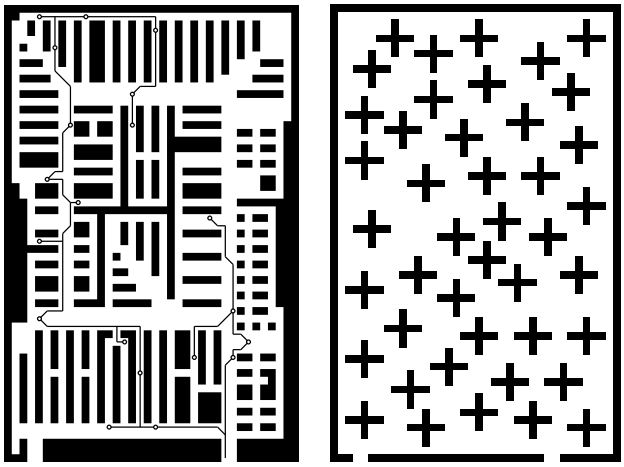
II. LIMITED CONNECTIVITY

One common approach is to include link quality in the motion [4] or goal generation functions. This idea is not readily applicable to general-purpose service robotics, because the team tasks may be totally unrelated to those generated in order to preserve communication. As a consequence, they suffer from local minima problems where some robots become trapped by obstacles, threatening mission completion. Some proposals [5] thus require temporary network splits to allow for escape routes, and therefore do not guarantee uninterrupted connectivity.

Closely related are self-deploying mobile sensor networks. In [6], a team of mobile relays maximizes network route qualities by means of algorithms with varying degrees of decentralization. In addition to locality problems, these approaches do not directly address the problem of mission performance or execution, since they instead provide a networked environment for other robots.

We aim our interest at approaches that can be used on routing problems in scenarios with complex obstacles while providing uninterrupted connectivity. Some of our early work [7] takes first steps in this direction, adapting task allocation algorithms to operate on top of a signal-constrained navigation model; however this work cannot be readily applied to complex environments. In [8], the Hoplites framework is applied using a line-of-sight constraint, but actual link quality is not used. Another proposal [9] experiments with a robot that deploys stationary relays when necessary to extend its range of operation in out-of-sight scenarios, focusing mainly on hardware issues. Finally, in [3], we used a similar idea of robots that stop to act as relays with a focus on mission cost optimality.

A. R. Mosteo and L. Montano are with the Instituto de Investigación en Ingeniería de Aragón, Universidad de Zaragoza, 50018 Zaragoza, Spain {amosteo,montano}@unizar.es



a) Mall b) Random

Fig. 1. a) Shopping mall scenario with overlaid path tree. Twenty targets are shown as knots in the tree. b) Randomly generated scenario.

A. *Sequential* CONNECTTREE *strategy*

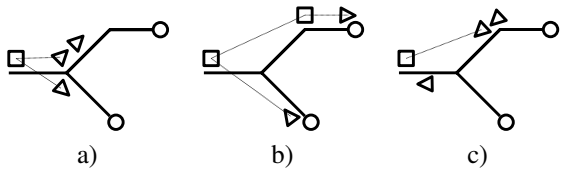
Our generic strategy for multi-robot routing under limited communication range in dense environments is based on the following ideas (which are fully detailed in [3]). A tree of paths is constructed rooted at the base and containing all targets (Fig. 1a). Robots move together as a whole (in practice in single column formation) towards a target, monitoring the network link to the base; once this link quality falls under some predefined safety threshold, the tail robot is stopped at that point to act as a relay, and the rest continue advancing while monitoring now their link to the stopped relay robot[1]. Using this pattern repeatedly, the robots can reach far away from the base without ever breaking the multi-hop link to the base.

Once the target is visited, robots retreat from the target along the same path they used to reach it, deliberately mirroring the deployment: a relay can start retreating only when all moving robots reach it back. Paths to consecutive targets may share a portion in the upper levels of the tree. Therefore, after visiting a target, robots do not need to retreat all the way to the base to move on to the next target. Instead, it is sufficient to retreat up to a common ancestor point between the two paths. Thus, during the entire process, critical network links lie within the tree and the relative robot ordering is always maintained. Besides other advantages, never two robots have to negotiate in a narrow passage.

B. *Concurrent* CONNECTTREE *strategy*

In the sequential model, only the head robot is used to visit targets. We now want to visit several targets simultaneously. To do so we allow the mobile pack of robots to *split* at any branching point of the tree. The split pack divides thus in two (or more) packs of some number of robots, each of these packs monitoring now their link to the previous common relay (or base). Each pack then proceeds down the tree towards a different target, and robots in the pack may stop to act as relays as needed, like in the sequential case, exclusively for the sake of connectedness of its own branch. This split operation may happen recursively as long as there are robots available.

[1]We presume that link quality between static endpoints remains reasonably bounded as not to break the link. This is true in our practical experience when using adequate safety margings. Parallel research in reactive spring-damper motion models such as found in [7] is in the pipeline of future work for use when this assumption does not hold.



a) b) c)

Fig. 2. An example with four robots. In a), one robot has already stopped to act as relay (square). At the branching point, it has been decided that two robots will continue along the upper branch and one along the lower branch (triangles). Their links to the previous relay are represented by thin lines. In b), the robot below has reached its target (circle) and it is starting its way back. Meanwhile, above, another relay has been stopped. In c), the small pack of one robot is waiting for the other pack of two to come back, in order to merge and continue up the tree and towards the base as a four-pack.

When one pack retreats after visiting its target, it must stop and wait at the common relay for the other pack(s) to come back. When all packs sharing a relay are reunited at the relay point, a conceptual *merging* occurs, and the original pack is reformed. This pack can now resume its way back to the upper areas of the tree. See Fig. 2 for an example.

The need for all packs to wait at the deepest common relay for a merge is due to the same considerations detailed in [3], which allow CONNECTTREE to provide cost bounds and mission completeness.

In consequence we can see that, from the point of view of robots going down a branch, nothing has changed in regard to the sequential case: they move within the tree, monitoring its link to the relay immediately above in the tree. Also, like in the sequential case, the retreat phase requires all robots below a relay to be reunited before this relay can return to its mobile status in order to move up, even if now robots come back in different packs.

With the introduction of the split action, we now have several decisions to make at each branching point: Should the pack split here? If so, how many robots would go down each branch? We will study these questions in Section V.

III. MULTI-ROBOT ROUTING

The considered *multi-robot routing* problem is formally specified by a set of robots, $R = \{r_1, r_2, \ldots, r_n\}$, a set of targets, $T = \{t_1, t_2, \ldots, t_m\}$, the locations of both robots and targets on the two-dimensional plane, and a non-negative cost function $c(i, j)$, $i, j \in R \cup T$, which denotes some abstract cost of moving between locations i and j (e.g. distance, energy, time, etc.). We assume that the robots are identical; therefore the same cost function applies to all robots. Typical cost measures are travel distance, travel time, or energy consumption between locations. In this work, we additionally assume that the cost function satisfies the triangle inequality. This assumption holds as long as the

robots follow the cheapest obstacle-free path when moving between locations.

The mission begins with all robots located at some base location and completes when all targets are visited and the robots return to base. The problem of multi-robot routing is to find an allocation of targets to robots and a path for each robot that begins at the base, visits all targets allocated to it, and ends back to the base, so that the mission is completed and a team objective is minimized. Three intuitive team objectives [1], corresponding to minimizing energy, makespan, and latency respectively, are:

MINSUM: Minimize the sum of the robot path costs over all robots.
MINMAX: Minimize the maximum robot path cost over all robots.
MINAVE: Minimize the average target path cost over all targets.

The *robot path cost* of a robot is the sum of the costs along its entire path. The *target path cost* of a target t is the total cost of the path traversed by the designated robot from the base up to target t along its path. The three team objectives above can be formally expressed as

$$\begin{aligned} \text{MINSUM} &: \min_A \sum_j RPC(r_j, A_j), \\ \text{MINMAX} &: \min_A \max_j RPC(r_j, A_j), \\ \text{MINAVE} &: \min_A \frac{1}{m} \sum_j CTPC(r_j, A_j) \ , \end{aligned}$$

where $A = \{A_1, A_2, \ldots, A_n\}$ is a partition of the set of targets, so that targets in A_i are allocated to robot r_i, $RPC(r_i, A_i)$ denotes the *robot path cost* for robot r_i to visit all targets in A_i starting from and finishing to the base, and $CTPC(r_i, A_i)$ denotes the *cumulative target path cost* of all targets in A_i, again, if robot r_i visits all targets in A_i starting from the base.

IV. PROBLEM CHARACTERIZATION

In this section we address the algorithmic complexity of obtaining optimal solutions for each of these team objectives. In order to analytically evaluate the cost of the obtained solutions, we assume a known and fixed value for the maximum length L of the link between two robots. This is the simplest most used model for wireless links, which is more tractable for a first approach. Promising results would grant a more sophisticated stochastic model with detailed simulation of wireless propagation as, for example, in [2].

Our previous work was devoted to the building of trees, which is an instance of the Steiner problem [10] (also NP-hard). Thus, in this work we start from an already constructed fully traversable tree and try to minimize the objectives for that tree. However, concurrency cannot in any case improve MINSUM cost since the total weight of a tree is fixed and the optimal (and polynomial on the size of the tree) solution is any sequential depth-first traversal. Therefore, for the remainder of this paper, we focus exclusively on the other two objectives.

Counterexamples in Fig. 3 show that depth-first traversals are no longer always optimal. This particular problem of concurrent traversal of a tree has ties with optimal covers (for the selection of compatible branches that can be visited concurrently) and scheduling (because of the timing aspects of concurrent execution). We next show, using reduction, that the problem is at least NP-hard for the MINMAX objective.

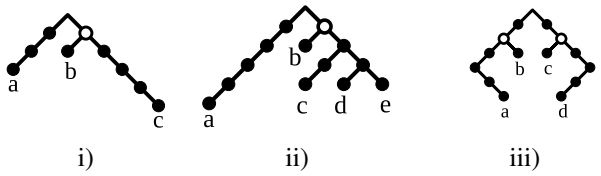


Fig. 3. Let it be the circles the places were relays stop and, for evaluation purposes, L=1. Root is at top and targets are located at the labeled leaves. White circles represent nodes that violate the depth-first property in the optimal solution. In i), with $n = 5$, the optimal solution would be to visit concurrently a and b, merge at the root, visit c and return to base, for a MINMAX cost of 16. The best depth-first solution requires a cost of 18, since b cannot be visited concurrently with any other target. A similar reasoning gives optimal cost of 18 for ii) with $n = 7$ and 30 with $n = 6$ for iii), respectively.

We start from the NP-hard problem of *multiprocessor scheduling (SS8)* in [11]. Its inputs are a set T of tasks, and for each task t in T its execution time C_t, and a global deadline D. The question asked is, *is there a 2-processor non-preemptive schedule for the tasks that completes within deadline D?* That is, is there a way of mapping the tasks to start times in such a way that no more than two tasks overlap and all of them finish before D?

We can transform this problem into one instance of our concurrent tree traversal this way: we set $n = 2$ (two robots that take the role of processors), $L = \infty$ (no practical network restriction), and construct a tree shaped as a star (every leaf connected to the root) where each leaf contains a target t from T, with edge length set as $\frac{C_t}{2}$ (so visiting each leaf and coming back takes exactly C_t). By solving this instance, we could give a solution to the original problem, since each departure of a robot from the root is equivalent to the scheduling time of a task. The transformation from optimization to decision problem is trivial. For the second part of the proof, given a solution to our problem, we can verify in polynomial time (by simply simulating it) that the constraints and deadline D are honored, thus completing the proof of NP-hardness. ■

A conservative upper bound of the brute force algorithm would be: at each branching point, we can divide our robots in 2^n subsets; the tree contains at worst $t-1$ internal nodes (branching points). Each internal node may merit at most t splitting operations, if only one target is visited per split. This gives a pessimistic upper bound of $O(t^2 2^n)$. Indeed, the subject of determining a tighter bound for the brute force case is not trivial and has been recently studied for a related problem in [12].

V. ALGORITHMS

A. Deterministic heuristics

Given the previous result, it is reasonable to find ideas that could provide good heuristics. Most obvious, perhaps,

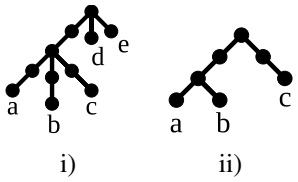


Fig. 4. In i), with $n = 6$, the optimal solution involves sending all robots down the leftmost branch, in order to visit e.g. a and b concurrently (late split), then c and finally d and e concurrently, for a MINMAX cost of 14. If we attempted an early split at the root, the four robots remaining for the leftmost branch would require 16 time units to return. Contrarily, in ii) with $n = 5$ an early split gives the optimal cost of 8 against the 10 units of a late split in the leftmost branch to visit a and b concurrently.

is when to split a robot pack. Two possible options are: a) to split as soon as we know that we can reach the farthest targets in all descendant branches concurrently (we refer to this as *early* split), even if this implies visiting sequentially targets within each of these branches; or b) keep together until we can visit all remaining targets below the current split point concurrently (we refer to this as *late* split). Again, there is no clear winner for these strategies. See optimal solutions for each case in Fig. 4.

Another idea is which branch to visit next when there is no split. Inspired on TSP heuristics [13], we may choose the next farthest target or the next closest target. The combination of these two ideas gives the four next combinations: farthest-first with early split (FAREARLY in the figures), closest-first with early split (NEAREARLY), farthest-first with late split (FARLATE), and closest-first with late split (NEARLATE).

One particularly interesting aspect of the sequential CONNECTTREE strategy is that it offers cost bounds over the optimal unconstrained TSP solution. We can retain these bounds in our heuristics if we impose some restrictions on the solutions attempted. As we have seen in previous examples (Figs. 3 and 4), cost gains are obtained when some two (or more) branches are executed concurrently. If we thus allow only depth-first traversals as in [3], while opportunistically looking for some other branches that can be *concurrently executed in less time* than the current depth-first branch, it follows that we cannot do worse than the original traversal. Hence, any bounds that it had are retained. This group of heuristics implements this property. Also, they run in $O(m)$ time, with no perceptible delay for the amounts of tasks tested.

For comparison, the basic sequential depth-first, closest-first traversal (SEQDF) from [3] is used here as baseline for measuring improvements in the evaluation runs.

B. Randomized traversal with timeout

RANDOM10/RANDOM60: This heuristic has the potential of finding the optimal solution at the price of sacrificing any cost guarantees. At each split point, we randomly choose the next branch and how many robots to send, between the minimum needed for the closest leaf target (early split), and the maximum robots available (a late split). This process is repeated iteratively with any remaining robots in the same split point, unless a coin toss dictates to leave them unused.

In other words, at each juncture we randomly choose one of many possible actions. Thus, with sufficient running time, this algorithm could theoretically explore a large portion of the solution space. The best known cost serves to discard the complete exploration of solutions that are definitely known to be suboptimal.

The number in the name is the timeout in seconds for the algorithm. The best solution found on timeout is reported.

C. Optimal solution by brute-force

OPTIMAL: In our effort to characterize the results obtained, we implemented a brute-force algorithm in order to obtain some optimal solutions, even if for small problem instances. This algorithm explores every possible choice, given enough time and memory. A bread-first search is performed in which every known state is ordered by either current MINMAX or MINAVE cost, depending on the objective. When the earliest state is final, it corresponds to the optimal solution. In practice, we were able to solve problems ranging from 12 to 17 targets, depending on the complexity of the solution path tree.

VI. EVALUATION

A. Analytical setup

In order to implement and evaluate these heuristics, we constructed an exact grid simulator. The parameters of the test missions, unless explicitly stated otherwise, were as follows. The simulation map is a $135m \times 86m$ realistic scenario (Fig. 1), modeled after a shopping mall close to our lab, where obstacles are the product stands, and in which we hope to conduct future real experimentation. Visiting randomly distributed targets in such a scenario may represent tasks of object inspection or delivery, service requests, or sample collection missions, for example.

There were $n = 8$ mobile robots and $m = 50$ targets in the mission. The targets were uniformly randomly distributed over the free cells of the map. The communication range was modeled as the branch length between robots, being, by default, $L = 35$, which is not far from the minimum needed to reach the farthest points in the map using all robots. Ten runs were performed for each algorithm, using a different seed for each run; nevertheless, the same set of seeds was used across the different algorithms, so that they are compared on the same basis.

With these starting points, in each test we modified one variable of interest, as detailed in the following subsections.

B. Network range influence

This test was set up to explore the influence of network range on mission performance. It is to be expected that requiring more robots for the farthest targets would reduce the opportunities for concurrency. Hence, the link length L was set to the values of $L = 25m$ and $L = 50m$ (typical of Wi-Fi transmitters). The shortest range is barely enough to reach the farthest places in the simulation map. The longest range, on the contrary, is quite generous, thus allowing us to

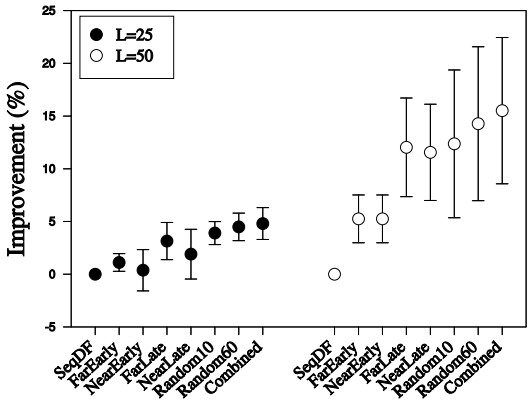


Fig. 5. Mean and 95% confidence interval of cost reduction for the MINMAX objective over the sequential depth-first closest-first traversal.

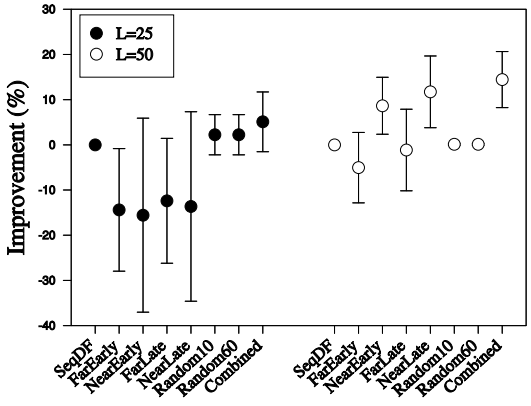


Fig. 6. Mean and 95% confidence interval of cost reduction for the MINAVE objective over the sequential depth-first closest-first traversal.

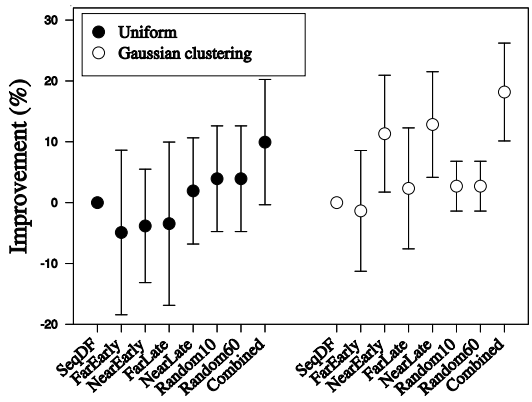


Fig. 7. Mean and 95% confidence interval of cost reduction for the MINAVE objective over the sequential depth-first closest-first traversal with both non-clustered and clustered random targets.

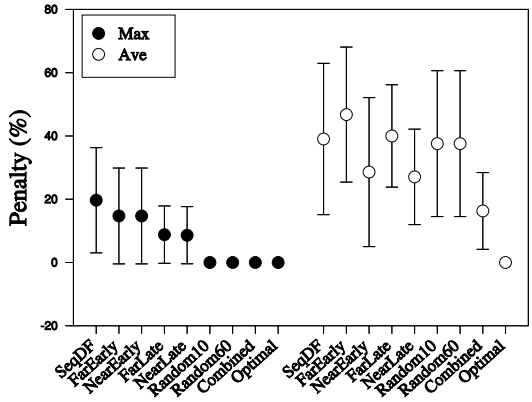


Fig. 8. Mean and 95% confidence interval of cost overhead over the optimal solution found by brute force, for both cost objectives.

find which improvements can be expected in such favorable conditions.

This test was performed for both the MINMAX and MINAVE objectives. The results are, respectively, in Figs. 5 and 6. In these and similar figures, the COMBINED values were obtained by taking the best result in each run obtained by any heuristic. Therefore, they represent our best realistic expectations of improvement over the sequential strategy.

Fig. 5 already shows some interesting results: even in such tight conditions, some modest improvements are possible. However, it is with $L = 50$ that we see to which extent we can expect improvements, with instances running close to 20%. Furthermore, it made patently clear that the *late split* strategy is better than the *early split* one. This is also interesting because in future real experiments, with the possibility of requiring replannings (see [3]), late splits will minimize the penalty for a replanning.

Turning to Fig. 6 for the task averages we see less optimistic results. This is not entirely surprising, since we are not, in truth, attempting any MINAVE specific optimizations, besides when greedily going for the nearest targets. This is patent in the $L = 50$ portion of the figure, where both near-first heuristics fared better. Another point of interest is that the random heuristic seemed lost when given a larger solution space to choose from, and in fact could not perform better than the sequential approach.

C. Clustering influence

Clustering of targets has a definite importance for MINAVE costs. In effect, algorithms that fail to detect target clusters (e.g. greedy strategies) can be dramatically outperformed by more potent optimizers. In this test we tried to ascertain if, particularly the random strategy, could take advantage of the wider solution space explored. Tasks were either uniformly distributed or following a Gaussian random distribution around a random but small (no less than one, no more than 4) number of clusters. By looking at Fig. 7 we can see that, again, the short time allotted to the random heuristics was not enough to observe any particular advantage for them. However, the existence of clusters did augment the chances of improving the sequential results, because late splits resulted in large amounts of targets visited in short periods of time.

D. A peek into optimality

For the sake of our brute force solver, we set in this trial $m = 12$, allowing us to obtain truly optimal mission costs for both objectives. Albeit with a reduced solution space,

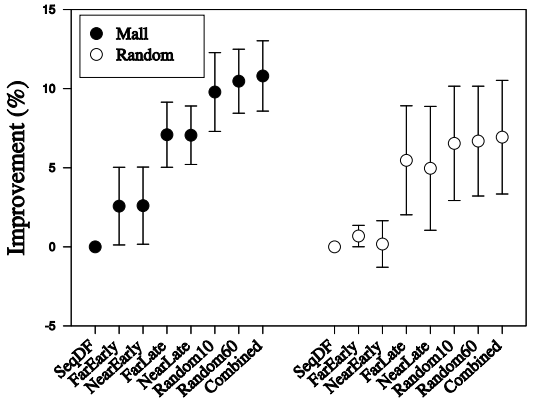


Fig. 9. Mean and 95% confidence interval of cost reduction for the MINMAX objective over the sequential depth-first closest-first traversal in both scenarios.

the results are quite revealing (Fig. 8). We confirmed the tendencies seen in Section VI-B that favor late split for the MINMAX case. Also, we see that the small problem size allows the randomized solver, just like in the $L = 25$ case, to come ahead of the other heuristics. Results are worse in the MINAVE case but, still, by using the combined results of all heuristics, we can get acceptable results given the irregular performance of the heuristics.

These results point that the work in MINAVE specific algorithms is granted, since average task latency is arguably as potentially useful for service robotics as minimum mission time, and there is a larger gap to close.

E. Random scenarios

In this final test, we tried to rule out any evident artifacts in the results due to the particular simulation scenario used. We constructed random scenarios to this end as the one shown in Fig. 1b), randomly placing crosses into an empty area of the same size as the mall, until it was saturated. The results shown in Fig. 9, albeit less generous, illustrate that the same trends can be observed in both kinds of scenario.

F. Final remarks

After these batteries of tests, we can extract several concluding remarks. Firstly, we see that by using the combined results of all algorithms, we can expect to obtain noticeable improvements over the sequential strategy, which is an encouraging result. However, we also saw that there is room for improvement, specially in regard to the MINAVE objective. We also observed that late splitting is, in general, a more successful strategy. This may be of use for the design of improved heuristics, and is definitely advantageous for the real case in which network range is not known.

VII. CONCLUSION

In this work we studied the concurrent traversal of trees with multi-robot teams under limited communication range, in order to improve our past work with the CONNECTTREE strategy. While the initial design of this strategy only visited targets in a sequential way, we have now set the foundations to use concurrency whenever possible in order to reduce mission costs. We have done so in such a way that the cost bounds that were demonstrated for the sequential strategy can be retained in some cases for the concurrent strategy.

As part of our approach to concurrency, we studied the algorithmic complexity qualities of the problem at hand, showing that it is NP-hard and thus ruling out the immediate availability of an optimal algorithm for the general problem. Instead, we described several heuristic approaches and studied the improvements that these heuristics can offer. We saw that noticeable improvements can be obtained for both makespan and latency objectives, and to some extent quantified, by comparing to optimal solutions for small problem instances, the gap still to cover.

Future work will focus on the implementation of such improvements in our team of real robots, in order to evaluate the influence of the jump from simulation to experimentation. Also, theoretical research into the latency optimization objective is needed in light of the results presented herein.

VIII. ACKNOWLEDGMENTS

Thanks to Michail G. Lagoudakis for his support and insightful comments. This work was partially supported by the Spanish project DPI2006-07928 and the European project IST-1-045062-URUS-STP.

REFERENCES

[1] M. G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, and S. Jain, "Auction-based multi-robot routing," in *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[2] V. Sridhara and S. Bohacek, "Realistic propagation simulation of urban mesh networks," *Computer Networks*, vol. 51, no. 12, pp. 3392–3412, August 2007.

[3] A. R. Mosteo, L. Montano, and M. G. Lagoudakis, *Distributed Autonomous Robotic Systems 8*. Springer Berlin Heidelberg, 2009, ch. Guaranteed-Performance Multi-robot Routing under Limited Communication Range, pp. 491–502.

[4] E. Stump, A. Jadbabaie, and V. Kumar, "Connectivity management in mobile robot teams," in *ICRA*. IEEE, 2008, pp. 1525–1530.

[5] M. N. Rooker and A. Birk, "Communicative exploration with robot packs," *Lecture Notes in Artificial Intelligence*, pp. 267–278, 2006.

[6] B. P. Gerkey, R. Mailler, and B. Morisset, "Commbots: Distributed control of mobile communication relays," in *AAAI Workshop on Auction Mechanisms for Robot Coordination (AuctionBots)*, Boston, Massachusetts, July 2006, p. 51–57.

[7] A. R. Mosteo, L. Montano, and M. G. Lagoudakis, "Multi-robot routing under limited communication range," in *International Conference on Robotics and Automation*, 2008.

[8] N. Kalra, D. Ferguson, and A. Stentz, "A generalized framework for solving tightly-coupled multirobot planning problems," in *ICRA*, 2007, pp. 3359–3364.

[9] N. Pezeshkian, H. G. Nguyen, and A. Burmeister, "Unmanned ground vehicle radio relay deployment system for non-line-of-sight operations," in *IASTED Int. Conf. on Robotics and Applications*. Germany: DARPA/ITO, August 2007.

[10] F. K. Hwang, D. S. Richards, and P. Winter, *The Steiner Tree Problem*. Elsevier, 1992.

[11] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[12] T. P. Baker and M. Cirinei, *Principles of Distributed Systems*. Springer Berlin / Heidelberg, 2007, ch. Brute-Force Determination of Multiprocessor Schedulability for Sets of Sporadic Hard-Deadline Tasks, pp. 62–75.

[13] G. Reinelt, *The traveling salesman: computational solutions for TSP applications*, ser. Lecture notes in computer science. Springer, 1994, vol. 840.