A Journey Among Pairs of Vertices: Computing Robots' Paths for Performing Joint Measurements

Robotics Track

Alessandro Riva*
Politecnico di Milano
Milan, Italy
alessandro.riva@polimi.it

Jacopo Banfi
Politecnico di Milano
Milan, Italy
jacopo.banfi@polimi.it

Carlo Fanton
Politecnico di Milano
Milan, Italy
carlo.fanton@mail.polimi.it

Nicola Basilico
Università degli Studi di Milano
Milan, Italy
nicola.basilico@unimi.it

Francesco Amigoni
Politecnico di Milano
Milan, Italy
francesco.amigoni@polimi.it

ABSTRACT

The problem of performing joint measurements recurs in many robotic applications, like constructing communication maps from signal strength samples gathered on the field. In spite of this, a theory supporting efficient algorithms has not been yet developed and *ad hoc* methods are usually employed. In this paper, we consider an environment represented by a metric graph and prove that the problem of *jointly* performing measurements from given vertices is NP-hard when either the total traveled distance or the task completion time have to be minimized. Given the difficulty of finding optimal paths in an efficient way, we propose a greedy randomized approach able to cope with both the optimization objectives. In settings for which joint measurements must be taken for all pairs of vertices, we prove that a deterministic greedy algorithm achieves an $O(m \log n)$ approximation factor for the traveled distance objective, where m is the number of robots and n the number of vertices, and an $O(m^2 \log n)$ approximation factor for the completion time. Experiments in simulation show that our algorithms perform well in practice, also when compared to an *ad hoc* method taken from the literature.

KEYWORDS

multirobot systems; joint measurements

ACM Reference Format:
Alessandro Riva, Jacopo Banfi, Carlo Fanton, Nicola Basilico, and Francesco Amigoni. 2018. A Journey Among Pairs of Vertices: Computing Robots' Paths for Performing Joint Measurements. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 10 pages.

1 INTRODUCTION

Multirobot systems (MRSs) represent a major sub-field of mobile robotics whose challenges have received a growing attention from researchers and practitioners in the last few years [23]. Sensing-constrained planning is central to MRSs. It can be described as the problem of planning the optimal execution of a task where some constraints, like limited range, affect or are imposed to the robot's sensing capabilities. For example, consider coverage, where robots are often required to sense all the free area of the environment [10], or patrolling, where robots might need to check for the presence of threats at some locations with a given frequency [9]. Exploration, surveillance, and target search are also domains that, very often, exhibit the need for robots to take coordinated decisions accounting for sensing requirements [17]. In these settings, robots typically take sequences of measurements from locations that are determined in order to optimize some objective function related to the traveled distance or to the time taken to complete the task.

In this work, we consider a scenario where a team of robots has to perform a given pre-specified set of *joint measurements* in a graph-represented environment. While a measurement is usually defined as a data-acquisition operation performed by a single robot at some location, in this work, we consider a joint measurement as a pairwise operation performed by two robots that occupy two different locations at the same time. We address the off-line resolution of one fundamental problem emerging from this scenario: finding joint paths to perform all the required measurements at minimum cost.

Optimally planning pairwise joint measurements poses additional difficulties with respect to the case in which measurements are performed by single robots. Indeed, optimal solutions might exhibit intricate synchronization patterns, which can be difficult to capture in a systematic algorithmic framework. We tackle this problem by leveraging some peculiar features of the robotic setting. This allows us to derive complexity and approximation results as well to provide an empirical evaluation of our algorithms tailored to MRSs.

Our planning problem (Section 3) unfolds on a discretization of the environment that we represent with a graph. The vertices correspond to locations of interest that can be occupied by a single robot, while the weighted edges represent the shortest paths between such locations. On this graph, a subset of edges defines the measurements to be performed. A plan encodes a joint walk for the robots to perform all the required measurements. We evaluate plans according to two objective functions: the total cumulative

*Corresponding author

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden. © 2018 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

distance and the mission completion time. In this paper, we provide the following contributions:

- we prove that finding the optimal plan is NP-hard with both objective functions (Section 4);
- we propose a randomized greedy algorithm able to cope with both the optimization objectives (Section 5);
- we consider settings, with n vertices and m robots, where measurements must be taken for all pairs of vertices and we prove that a deterministic greedy algorithm achieves an $O(m \log n)$ approximation factor for the traveled distance objective and an $O(m^2 \log n)$ approximation factor for the completion time (Section 5);
- we perform extensive experiments that show that our algorithms perform well in practice, also when compared against an *ad hoc* method taken from the literature (Section 6).

2 RELATED WORK

At a low level of abstraction, limited-range sensing poses constraints to several robot applications. Multirobot exploration is one notable example. In such a domain, limited visibility ranges have an impact on how robots can optimally conduct exploration [15]. At a more abstract level, sensing constraints can take the general form of requirements imposed to locations in the environment. In order to cope with such requirements, robots need to combine the "where to go?" decision with the "what to sense?" one. Think, for example, of a team of robots that is required to collect temperature samples from particular spots in the environment.

When requirements are posed to single robots, vehicle routing [20] or scheduling and sequencing [5] formulations are typically considered. However, in this paper, we abandon the traditional formulation according to which a sensing requirement corresponds to a single location in the environment. Our joint measurements are associated to pairs of locations and require the simultaneous presence of two robots in the two locations in order to be performed. We exploit a specific problem formulation that allows us to devise a practical resolution approach for MRSs and to derive complexity and approximation results. What motivates us is also the fact that this joint sensing constrained planning problem has received much less attention in the literature than the single-robot counterpart, even though it can be found in many applications.

One first example, from which our work took initial inspiration, is the construction of communication maps [2, 12]. A communication map provides estimates of the radio signal strength between pairs of locations in the environment. In such a setting, a measurement is performed by two robots at two different locations that exchange some polling data to acquire a signal strength sample. An efficient planning of the measurements sequence can decrease map-construction costs.

Localization and positioning systems represent another application domain where joint measurements performed by robots are involved. Robot-to-robot mutual pose estimation allows robots to determine their global locations from mutual distance samples [24]. Efficiently sequencing pairwise (mutual) distance measurements while executing some mission can help the robots to localize themselves in a distributed fashion [21].

Analogous problems can be encountered in the Wireless Sensor Networks (WSNs) field, especially when nodes are mobile units [13]. Examples can be found in multilateration-based settings [6] where optimal sequencing of pairwise measurements can speedup the localization of an external entity, a feature particularly critical when such an entity does not exhibit a cooperative behavior [3, 14].

3 PROBLEM FORMULATION

Let $G = (V, E)$ be a complete graph defined on n vertices, and let $c : E \to \mathbb{Z}^+$ be an edge cost function satisfying the triangle inequality. A team A of m robotic agents is deployed on G. They have homogeneous locomotion capabilities and can move between the vertices of G by traveling along its edges at uniform speed, implying that costs can represent either distances or traveling times between vertices.

The agents have to perform a set of coordinated pairwise sensing actions, called *measurements*, from a set $M \subseteq E$ of selected pairs of vertices. A single measurement is considered completed as soon as two agents occupy the pair of vertices at the same time (one agent in each vertex). All the agents start from a common depot $d \in V$ and must come back to it once all measurements have been performed. Note that, since G is a complete metric graph, we can assume without loss of generality that each vertex in V, with the exception of the depot, will always be part of at least one measurement in M.

We represent the execution of a joint measurement task with an *ordered sequence* $S = [s_1, \ldots, s_{|M|}]$, where each element s_k is called *assignment* and associates a pair of agents to a pair of target vertices from which the measurement is performed (each s_k is associated to a unique pair of vertices in M). During the exection of S, each agent $i \in A$ remains still on its current position, until a new vertex is scheduled to i by means of an assignment. Given S, let us define i_k^S as the vertex position of the agent $i \in A$ after the ordered execution of all the assignments up to (and including) the k-th one. These agent positions, initially set to d, capture the evolution of the system while running through S. In accordance with that, and with a slight overload of notation, we can define the cost of an assignment $s_k \in S$ as:

$$c(s_k) = \sum_{i \in A} c(i_{k-1}^S, i_k^S).$$

We define a first objective function capturing the *distance cumulatively traveled* by the team of agents and we denote it as SUMDIST$(\cdot)$:

$$\text{SUMDIST}(S) = \sum_{s_k \in S} c(s_k) + \sum_{i \in A} c(i_{|M|}^S, d). \tag{1}$$

We now introduce the notion of time for an ordered sequence of assignments. We denote $t_j^S(k)$ the time accumulated by an agent j executing the assignments up to (and including) the k-th one. Such a value can be defined recursively. In particular, if the assignment s_k does not schedule any vertex to the agent j, then $t_j^S(k)$ remains unchanged and equal to the value assumed in the previous step. Otherwise, the baseline value is increased by:

$$t(s_k) = \max_{i \in A} \ c(i_{k-1}^S, i_k^S).$$

We can now define a second objective function as the *mission completion time*, namely the latest time at which a robot ends its

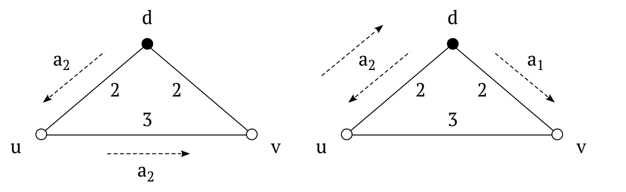


Figure 1: An instance in which the SUMDIST(·) **objective (left) and the** MAXTIME(·) **objective (right) cannot be optimized simultaneously.**

duty arriving at the depot. We call it MAXTIME(·):

$$\text{MAXTIME}(S) = \max_{i \in A} \left\{ t_i^S(|M|) + c(i^S_{|M|}, d) \right\}. \quad (2)$$

Our cumulative distance and completion time objectives represent, in our problem setting, natural and widely adopted cost metrics for MRSs applications.

3.1 Incompatibility of the Two Objectives

We now show that the two objective functions we just introduced are conflicting. We do this with the help of an example, also providing an intuitive explanation on how our encoding for solutions S maps to a multirobot graph walk.

Proposition 3.1. *The* SUMDIST(·) *and* MAXTIME(·) *objectives cannot be always simultaneously optimized, even when $m = 2$.*

Proof. Consider the simple graph with $V = \{d, u, v\}$ depicted in Figure 1. Two agents a_1 and a_2 initially placed at the depot d have to perform two joint measurements defined by the set $M = \{(d, u), (d, v)\}$. By inspection, we can see that a solution S_D^* minimizing the SUMDIST(·) objective is

$$S_D^* = [\langle a_1 \to d, a_2 \to u \rangle, \langle a_1 \to d, a_2 \to v \rangle],$$

with SUMDIST(S_D^*) = 7 (note that the return to the depot is not explicitly represented in ordered sequences S). Here, agent a_1 remains fixed at d while a_2 moves to both u and v (eventually returning at d). In this case, MAXTIME(S_D^*) = 7. Focusing on the MAXTIME(·) objective, instead, we see (again by inspection) that an optimal solution S_T^* is

$$S_T^* = [\langle a_1 \to d, a_2 \to u \rangle, \langle a_1 \to v, a_2 \to d \rangle],$$

with MAXTIME(S_T^*) = 6 and SUMDIST(S_T^*) = 8. To optimize the latter objective, no agent remains fixed at the depot, at the expenses of an increase in the total traveled distance. □

4 NP-HARDNESS

We now prove the following theorem.

Theorem 4.1. *The two optimization problems related to the minimization of the* SUMDIST(·) *and the* MAXTIME(·) *objectives are NP-hard, even when $m = 2$ and each $v \in V$ appears at most once in M.*

Proof. To prove the theorem, it is sufficient to show that the corresponding decision problems, in which we ask for the existence of a solution S with SUMDIST(S) $\leq D$ and MAXTIME(S) $\leq T$, respectively, are NP-complete. In the following, we call such decision problems SUMDIST-D and MAXTIME-D.

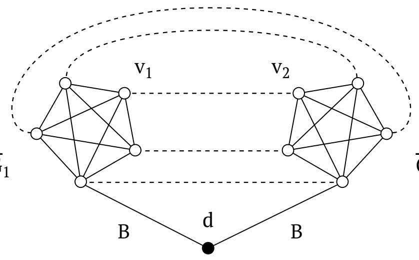


Figure 2: A reduction from a METRIC TSP instance with five vertices.

The NP membership of SUMDIST-D and MAXTIME-D directly follows from the solution encoding outlined in the previous section, together with the fact that the SUMDIST(·) and MAXTIME(·) objectives for a given S can be computed in polynomial time. To prove the NP-hardness of both problems, we provide a reduction from the well-known metric Traveling Salesman Problem (TSP) [11], formally defined below.

METRIC TSP
INSTANCE: Complete graph $\overline{G} = (\overline{V}, \overline{E})$, distance function $d : \overline{E} \to \mathbb{Z}^+$ satisfying the triangle inequality, positive integer B.
QUESTION: Does $\overline{G}$ contain a Hamiltonian cycle[1] with total distance B or less?

Let us consider first SUMDIST-D. From a generic instance of METRIC TSP, we construct a particular instance of SUMDIST-D with $m = 2$ and a graph $G = (V, E)$ obtained as the metric closure[2] of the graph depicted in Figure 2. In the figure, the original METRIC TSP graph is replicated twice in two subgraphs $\overline{G}_1$ and $\overline{G}_2$ which are connected to a depot d through the same vertex copy (chosen arbitrarily) by means of an edge with cost B. In the following, we denote by v_1, v_2 the two vertices obtained by replicating twice a generic vertex $v \in \overline{V}$. The set of measurements is defined as $M = \{(v_1, v_2)\ \forall v \in \overline{V}\}$, that is, the set of measurements is composed of all the pairs of vertices copies. The reduction is then completed by setting $D = 6B$.

Let us focus on the agents' movements in G (which is a complete graph) as happening in the underlying (non-complete) graph depicted in Figure 2 (this is equivalent, since G is obtained as its metric closure).

If the METRIC TSP instance has answer *yes*, there also exists a solution S of SUMDIST-D with SUMDIST(S) $\leq 6B$ in which agent a_1 and a_2 initially reach $\overline{G}_1$ and $\overline{G}_2$, respectively (spending $B + B$), then visit the vertices copies in the order defined by the METRIC TSP solution (spending *at most* $B + B$), and finally travel back to the depot (spending $B + B$). Hence, the total cost is not greater than $6B$.

[1]A Hamiltonian cycle is a closed loop in a graph that visits each node exactly once.
[2]The metric closure of a weighted graph is a complete graph with the same vertices and in which edges are weighted by the shortest path distances between corresponding vertices in the original graph.

On the contrary, consider a SUMDIST-D instance admitting *yes* answer. Necessarily, the two agents must perform measurements while each one always remains confined in one of the two portions of graphs derived from the METRIC TSP instance (excluded the edge needed to travel from/to the depot). Indeed, the existence in S of two subsequent assignments of the form $\langle a_1 \to v_1, a_2 \to v_2 \rangle$ and $\langle a_1 \to u_2, a_2 \to u_1 \rangle$ would immediately imply a total cost of at least $8B$ (with each agent spending $2B$ to travel from/to the depot the first time and again $2B$ for the second time). Consider now any solution S in which the measurement associated with the pair of vertices attached to the depot (by means of the edge with cost B) is not the first one performed. Since $\overline{G}$ (and hence G) is metric, S can always be turned into a solution in which such a measurement is the first one performed without increasing the total solution cost. Therefore, from such a solution, we can immediately derive the existence of a METRIC TSP solution with total distance at most B by examining the order in which the measurements are made. This concludes the proof for SUMDIST-D. The proof for MAXTIME-D follows exactly the same reasoning as above (the only difference is that we need to set $T = 3B$) and is omitted. □

5 ALGORITHMS

From the results of the previous section it is clear that, unless P=NP, the existence of polynomial-time optimal algorithms for any of the two objectives is unlikely. However, for the special case $m = 2$, there exist two simple polynomial-time transformations from our optimization problems to particular TSP instances, for which advanced solvers exist [1]. (This parallelism was firstly noted in [12] for the SUMDIST($\cdot$) objective.)

We first present the transformation procedures for the two-agent case, and then we devise a *greedy randomized* algorithm for general instances of the problems defined in Section 3.

5.1 An Optimal Algorithm for the Case $m = 2$

Consider a complete graph G_M, in which vertices correspond to elements of M. It is immediate to see that, for our purposes, the position of a salesman on G_M univocally identifies the position of a pair of agents on G, and each edge traversal on G_M represents a joint movement of two agents on G. In order to devise a cost function on G_M able to reflect the cost of moving two agents on G, it is sufficient to distinguish among our two objectives. In case of SUMDIST($\cdot$) minimization, given two vertices $e = (u, v)$ and $\tilde{e} = (\tilde{u}, \tilde{v})$ of the G_M graph, the cost between e and $\tilde{e}$ is defined as

$$\min\{c(u, \tilde{u}) + c(v, \tilde{v}), c(u, \tilde{v}) + c(v, \tilde{u})\}.$$

In case of MAXTIME($\cdot$), the cost is defined similarly but substituting the sum with the max operator. Each of these cost functions equals the corresponding cost of moving two agents in the original problem. Finally, since each vertex in G_M corresponds to a measurement that has to be performed, a TSP solution on G_M – after a proper cost function selection – can be easily turned into an optimal solution for the corresponding optimization problem. Furthermore, it can be easily shown that the TSP instances obtained with the above method are metric as well. This means that solving

Algorithm 1: Greedy Randomized

```
1  function greedyRandomized ($M$, $\tau$)
2      while a termination condition is not satisfied do
3          $S \leftarrow \emptyset$
4          $Q \leftarrow M$
5          while $Q$ is not empty do
6              rank the assignments according to a cost function
7              pick an assignment $s$ according to scheme (3)
8              remove from $Q$ the pair of vertices scheduled by $s$
9              append $s$ to $S$
10         update incumbent with $S$ if needed
11     return incumbent
```

the TSP instance with an α-approximation algorithm[3] allows to obtain the same approximation factor on our original problems in polynomial time.

The transformation outlined above is possible since the presence of exactly two agents univocally identifies the "optimal" distance functions of the corresponding TSPs. Indeed, both the state space and the measurements space are defined over pair of vertices. Clearly, with three or more agents, the same reasoning does not apply, and devising efficient algorithms with performance guarantees becomes significantly more complex.

5.2 A Greedy Randomized Algorithm

Randomization is a well known technique aimed at improving the performance of a plain greedy algorithm [4]. At every step, all the possible assignments are ranked according to a (greedy) cost function and a weight is associated at each rank. The probability of selecting an assignment is obtained by normalizing the weights.

We consider a polynomial scheme, which associates a weight $\omega(s)$ to an assignment s as follow:

$$\omega(s) = r(s)^{-\tau}, \tag{3}$$

where $r(s)$ is the rank order of s and the parameter τ is tuned to control how much the random selection is biased towards a greedy choice. As τ is set to higher values, weights and selection probability get more concentrated towards the most cost-efficient assignments.

The whole procedure is then run many times and the solution returned is the best solution found over all the runs. Algorithm 1 summarizes the greedy randomized algorithm, regardless of the specific cost function. Assuming that an assignment can be evaluated in a constant number of steps, each greedy randomized solution can be found in $O(m^2|M|^2 \log m|M|)$ steps, implying that, if the number of runs is polynomial in the input size, the whole algorithm has a polynomial running time.

In the following, we will make use of two different (greedy) cost functions, suitable for the two objectives given in Section 3. Specifically, the two costs of an assignment s are defined as marginal gains, namely the increments produced in the objectives SUMDIST($\cdot$) and MAXTIME($\cdot$) if s is appended to the current partial solution. We refer to the algorithm using the former cost as Algorithm 1_D and to the algorithm using the latter cost as Algorithm 1_T.

[3]Currently, the best approximation algorithm for the metric TSP is Christofides' 3/2-approximation algorithm [8].

Note that, thanks to the random selection based on (3), every possible assignment has a non-zero probability of being chosen at each step. This implies that, for any finite instance and any finite value of τ, both the algorithms are asymptotically optimal for the respective objectives. In the special case of $\tau \to \infty$ the algorithms become purely greedy and the solution found will be not guaranteed optimal. In this case, if the tie breaking is deterministic, e.g., lexicographic, the solution produced is deterministic as well, and the algorithm does not need to be run more than once.

An interesting question is how much a greedy solution can be worse than an optimal one. We partially answer this question by investigating the case in which $M = E$ for both the problem objectives. In particular we show that, for $\tau \to \infty$ (that is, by using a pure greedy algorithm), non-trivial approximation guarantees can be obtained.

5.3 Approximation Factors

We first provide some results that will be useful in proving the approximation factor for Algorithm 1_D. We start by putting in relation the maximum cost of a greedy assignment and the cost of an optimal solution.

Lemma 5.1. *Let S_D be a solution found by Algorithm 1_D and let* OPT_D *be the cost of an optimal solution minimizing the* SUMDIST$(\cdot)$ *objective function. Then,*

$$\max_{s \in S_D} c(s) \leq 2\,\mathrm{OPT}_D\,.$$

Proof. Consider a *single depot multiple traveling salesmen problem* (m-TSP) on G, with m salesmen and a depot d. In this problem, the goal is to find a set of m tours, each starting from d and visiting at least once every vertex in G, such that the sum of the tour costs is minimized. Let $C_{\text{m-TSP}}$ be the cost of the m-TSP optimal solution. Since in our problem each vertex must be visited at least once, it follows that $C_{\text{m-TSP}} \leq \mathrm{OPT}_D$. Then, let C_{MST} be the cost of a minimum spanning tree[4] on G. We have $C_{\mathrm{MST}} \leq C_{\text{m-TSP}}$, because from any m-TSP solution a spanning tree can be obtained by removing the last tour edges (the ones returning to the depot). Finally, the cost between any pair of vertices in G cannot be greater than C_{MST}, because of the triangle inequality. Reconstructing the chain of inequalities we have that the sum of two costs can never exceed twice OPT_D. □

At this point, the reader may have already noticed that a solution encoded as a sequence of assignments may contain some elements, whose contribution in the objective function is zero. These "zero-cost assignments" occur whenever a pair of vertices is measured without changing the positions of the agents on the graph. In order to prove the approximation factor, we need to pay particular attention to the number of zero-cost assignments occurring in an optimal solution. Specifically, we prove the following.

Lemma 5.2. *Consider a problem in which all pairs of h vertices have to be measured with m agents. Then, there always exists an optimal solution minimizing* SUMDIST$(\cdot)$ *and containing at least $\frac{h(h-1)}{2m}$ non-zero-cost assignments.*

[4]A minimum spanning tree is a least cost tree covering all the vertices in a graph.

Proof. Consider an optimal solution S_D^* and let us assume that, without loss of generality, all the zero-cost assignments are scheduled as soon as possible. Suppose that an assignment s moves at least one agent. Since at most two agents can be moved between two successive assignments, at least $m-2$ agent positions (i.e., vertices) are maintained unchanged. All the pairs involving those $m-2$ vertices must be already scheduled before s, according to the above assumption. Each zero-cost assignment following s, thus, must involve at least one of the new agent positions. As a consequence, the maximum number of consecutive zero-cost assignments is $2m-3$, which, considering the total number of pairs to measure $h(h-1)/2$, is slightly above our statement.

In order to achieve the result, let us further constraint S_D^* as follows. If an assignment moves exactly two agents i and j, we assume that both the new agent positions have been already measured with any of the other $m-2$ agent positions (those remained unchanged). Indeed, if this condition does not hold, any two-agent move can be split in two subsequent assignments, moving one agent each (the cost of moving two agents is equal to the cost of moving one agent at a time). The latter assignment can be the same as the original one, while the former can schedule one of the pairs to be measured among i or j and any other agents position. Notice that this "new" assignment would be necessarily scheduled later on in the sequence, but can be anticipated here without any increment in the cost of the solution. In fact, these two-split assignments can be interleaved by some zero-cost assignments, as we assume they are scheduled as soon as possible. The result is an optimal solution whose maximum number of consecutive zero-cost assignments is bounded by $m-2$ and we proved the lemma. □

We are now ready to formulate our main result. In particular, we state that, under specific conditions, Algorithm 1_D approximates an optimal solution (for the distance cumulatively traveled) within a factor that depends on the number of robots m and the number of vertices n.

Theorem 5.3. *Let S_D be a solution found by Algorithm 1_D with $M = E$ and $\tau \to \infty$, and let* OPT_D *be the cost of an optimal solution minimizing the* SUMDIST$(\cdot)$ *objective function. Then,*

$$\frac{\mathrm{SUMDIST}(S_D)}{\mathrm{OPT}_D} \leq 1 + 16m\log n.$$

Proof. We start by rearranging the computation of a solution cost. To do that, let us define λ_l as the number of assignments $s \in S_D$ such that $c(s) \geq l$, plus the number of depot returns whose cost is at least l. Then, the cost of the solution can be alternatively computed as

$$\mathrm{SUMDIST}(S_D) = \sum_{l \geq 1} \lambda_l. \tag{4}$$

Exploiting this new formulation, we can focus on bounding λ_l. Let us leave aside the return to the depot for a while – we will be back on it later – and let us start concentrating on the paths generated by Algorithm 1_D.

Consider a partition $\Pi(r)$ of the set of vertices V, defined for any positive value r and obtained as follows. Set initially $\Pi(r) = \{\}$, then (1) denote $T = V \setminus \{P \in \Pi(r)\}$ and pick (at random) a vertex o from T, (2) create a new subset $P = \{v \in T \mid c(o,v) < r\}$ and let o

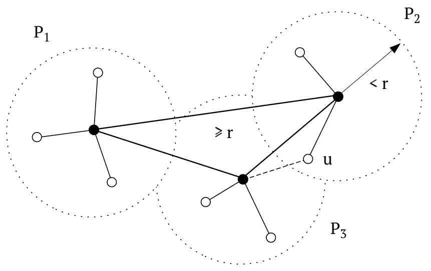


Figure 3: A vertices partition $\Pi(r) = \{P_1, P_2, P_3\}$. Due to the specific picking order, vertex u lies in P_2 instead of P_3.

be the *center* of P, (3) insert P in $\Pi(r)$ and repeat from (1) until T is empty. An example of partition is shown in Figure 3.

For each pair of vertices u, v belonging to the same subset P, it follows that $c(u, v) < 2r$, because of the triangle inequality. Consequently, if a greedy assignment of cost $4r$ or more is scheduled for a pair of agents, at least one of the two agents must leave its current subset. Furthermore, all the pairs of vertices, among the two agents' current subsets, must be mapped, as their costs are strictly lower than $4r$. It follows that the number of assignments of cost $4r$ or more is at most the number of transitions between different subset pairs (note that a subset can be paired with itself). Let us denote $h_r = |\Pi(r)|$, then, for any integer h_r:

$$\lambda_{4r} \leq \binom{h_r}{2} + h_r - 1 \leq h_r(h_r - 1). \tag{5}$$

Consider now an optimal solution measuring only the pairs of center vertices in $\Pi(r)$. Without loss of generality, we assume that depot d is the center of a subset (this can be easily enforced at the beginning of the partitioning process). The cost of an optimal solution, measuring only some pairs in E has cost not greater than OPT_D. Furthermore, thanks to Lemma 5.2, we can assume that in such an optimal solution the number of non-zero-cost assignments is bounded by a function of the number of agents m and the number of centers h_r. Also, each center is at least at distance r from any other center, meaning that any non-zero-cost assignment moves the agents of at least r. Thus, the following bound on the optimal cost holds:

$$\mathrm{OPT}_D \geq r\,\frac{h_r(h_r - 1)}{2m}. \tag{6}$$

From (5) and (6) we obtain:

$$\lambda_r \leq \frac{8m}{r}\,\mathrm{OPT}_D\,. \tag{7}$$

Before we go any further, we take into account the return to depot d in the λ_r computation. In particular, the bound in (5) has to be incremented by m as, in principle, any agent could spend more than $4r$ to come back to d. However, the lowerbound in (6) can be lifted too and, more precisely, it can be incremented by $r/2$. Indeed, if h_r is strictly greater than 1, at least one agent must spend an additional $r \geq r/2$ to come back to d. Conversely, if h_r is exactly equal to 1, the right-hand side term in inequality (6) becomes zero and by adding $r/2 \leq \mathrm{OPT}_D$ (with a view to the use of Lemma 5.1) the whole inequality still holds. Given these new two bounds, the result of inequality (7) follows straightforwardly.

Now we are ready to devise the approximation factor. In order to simplify the notation, let us define $x = \mathrm{OPT}_D / \binom{n}{2} + 1$. Leveraging Lemma 5.1, we decompose the sum in equation (4) as follow:

$$\begin{aligned}\mathrm{SUMDIST}(S_D) &= \sum_{l \geq 1} \lambda_l \\ &= \sum_{l=1}^{x-1} \lambda_l + \sum_{l=x}^{2\,\mathrm{OPT}_D} \lambda_l.\end{aligned}$$

The former term can be simplified noting that $\lambda_l \leq \binom{n}{2}$ holds for any value of l, while, in the latter term, we can make use of (7). To achieve the final result we only need some additional math:

$$\begin{aligned}\mathrm{SUMDIST}(S_D) &\leq \mathrm{OPT}_D \left[1 + 8m \sum_{l=x}^{2\,\mathrm{OPT}_D} \frac{1}{l}\right] \\ &\leq \mathrm{OPT}_D \left[1 + 8m \log \frac{2\,\mathrm{OPT}_D}{x - 1}\right] \\ &\leq \mathrm{OPT}_D \left[1 + 16m \log n\right].\end{aligned}$$

where in the last inequality we substituted the definition of x. □

Our greedy algorithm has thus a logarithmic approximation factor in the number of vertices, similarly to many other greedy algorithms for different applications (e.g., TSP [18] and set cover [19]) and depends linearly on the number of agents. This linear dependency becomes crucial when the number of agents is significantly large. However, in case of many robotics applications, $m \ll n$ is a reasonable assumption, strengthening the contribution of the theoretical result. Of course, being the approximation factor based on a worst-case analysis, the experimental performance can be better, as we will see in Section 6.

It can be also shown that a solution found by Algorithm 1_D has a non-trivial approximation factor even when evaluated w.r.t. the $\mathrm{MAXTIME}(\cdot)$ objective function. Indeed, starting from Theorem 5.3, the following bound can be easily derived.

Theorem 5.4. *Let S_D be a solution found by Algorithm 1_D with $M = E$ and $\tau \to \infty$, and let OPT_T be the cost of an optimal solution minimizing the $\mathrm{MAXTIME}(\cdot)$ objective function. Then,*

$$\frac{\mathrm{MAXTIME}(S_D)}{\mathrm{OPT}_T} \leq m + 16m^2 \log n.$$

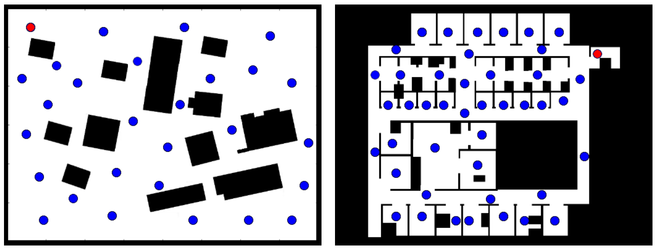
Proof. We can notice that, for any solution S (found regardless the objective function), we have the following bounds:

$$\frac{1}{m}\,\mathrm{SUMDIST}(S) \leq \mathrm{MAXTIME}(S) \leq \mathrm{SUMDIST}(S).$$

Hence, starting from the result of Theorem 5.3, we lowerbound the numerator and upperbound the denominator in the left-hand side of the approximation-factor inequality. After some straightforward math, we obtain an overall upperbound of the approximation factor for the $\mathrm{MAXTIME}(\cdot)$ objective function, as stated by the theorem. □

6 EXPERIMENTS

In order to assess the performance of the proposed algorithms in practical settings, we consider the real environments depicted in Figure 4. We manually spawned vertices on positions of interest (e.g., rooms in the office) over the bitmaps and derived edge costs as



(a) Open **(b) Office**

Figure 4: The bitmaps of the environments used in the experiments. The red vertex represents the depot.

shortest paths among pairs of vertices. In order to specify different sets of required measurements M, we consider different measurement *ranges* defined as numbers of pixels on the corresponding 800×600 bitmap. Given a range, we populate M with all the pairs of vertices (u, v) whose line-of-sight distance is not larger than the range (disregarding obstacles). In our experiments, we use the following three range values: 250 px, 500 px and 1000 px, obtaining sets M with different cardinalities (small, medium, and large).

In the first set of the experiments, we investigate the impact of the parameter τ (see (3)) in the solutions found by Algorithms 1_D and 1_T for their respective objective functions. To do that, we fix the number of agents and the range to intermediate values, that is, 5 agents and 500 px. Figure 5 shows the distributions of the solution costs, obtained in both the environments, for $\tau \in \{3, 5, 7, 9\}$ and over 200 runs. Each curve represents the probability of finding a solution cost less than or equal to a certain value, at each run of the corresponding algorithm. For example, in the open environment, the probability that the algorithm for $\tau = 5$ finds a solution S, such that $\text{SUMDIST}(S) \leq 2600$, is about 0.45 at each run. Although the set of tested values is limited, it is fairly easy to infer the general trend of the curves w.r.t. the τ parameter: increasing τ the distribution tends to fit the step distribution of a pure greedy algorithm. This is quite predictable, thinking about the parameter meaning. Using the pure greedy performance as reference, the results highlight that the curves of $\tau = 3$ and $\tau = 7$ are dominated in all the four scenarios. In particular, $\tau = 5$ achieves overall good performance, while $\tau = 9$ behaves generally bad.

In the second set of experiments, we vary the number of agents and we compare the results obtained by a deterministic pure greedy algorithm ($\tau = \infty$) to those obtained by a greedy randomized algorithm, for $\tau = 5$ and 50 runs for each setting. The optimal solution cost is computed for $m = 2$ agents only, exploiting the transformation of Section 5.1. For the case of $m = 3$ agents, we report the solution cost found by a state-of-the-art algorithm [12], here extended to cope with weighted graphs, and named HKT. At each step of the original HKT algorithm, all the next candidate positions of the agents are evaluated according to a greedy criterion. Such a criterion can consider (a) the traveled distance (the completion time), (b) the number of new joint measurements performed at the new agents positions, or a combination of the two. In some preliminary experiments, HKT algorithm obtained the best results when minimizing (a) for the $\text{SUMDIST}(\cdot)$ objective and when minimizing the ratio between (a) and (b) for the $\text{MAXTIME}(\cdot)$ objective (as often done in information-gathering applications, e.g., [7, 16, 22]). The algorithm proceeds by taking greedy decisions until all the measurements are completed. Notice that HKT cannot be extended straightforwardly to cope with a generic number of agents in polynomial time. Indeed, the number of next candidate positions at each step grows exponentially in the number of agents.

In Table 1, the costs of the solutions found by all the algorithms are shown for the open environment. Algorithm 1_D with $\tau = 5$ consistently outperforms the pure greedy version in all the problem instances. This, of course, comes at the expense of a greater computational burden. The results obtained by the greedy randomized algorithm are comparable even with those obtained by the optimal algorithm for $m = 2$. Also, note that the gap could be further reduced by increasing the number of runs. In case of $m = 3$, the solution costs obtained by $\tau = \infty$ are very similar those obtained by the HKT algorithm, which, however, is outperformed by $\tau = 5$ in all the considered instances.

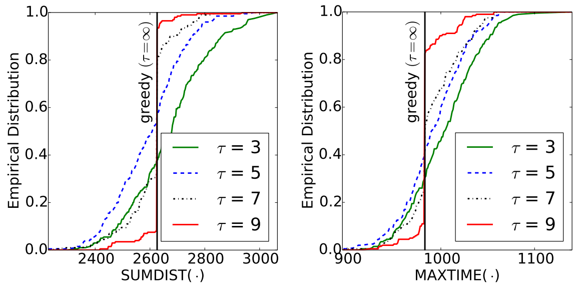
It is interesting to notice that, for the $\text{SUMDIST}(\cdot)$ objective and for range 250 px, the solution cost does not always decrease, as the number of agents increases. This is due to the greedy nature of Algorithm 1_D, which does not take into account the final returns to the depot. The same behavior seems to appear for range 500 px and large number of agents, suggesting the presence of a local minimum. As a consequence, the minimization of the total traveled distance by means of greedy algorithms requires additional attention in the selection of the number of agents.

In the last set of experiments, we investigate the practical dependency between the optimization of our objectives. While Proposition 3.1 ensures that, in general, both the problems cannot be optimized at the same time, Theorem 5.4 tells us that their joint performance is somehow related. Figure 6 shows the solutions found by the two pure greedy algorithms, evaluated according to the same objective. Specifically, we run Algorithms 1_D and 1_T (with $\tau = \infty$) and measure the quality of the solutions they produce according to *both* $\text{SUMDIST}(\cdot)$ and $\text{MAXTIME}(\cdot)$ objectives in the open and office environments. Even if the algorithms perform generally better in their respective objectives, the difference is relatively small. In particular, in case of the $\text{MAXTIME}(\cdot)$ objective the gap remains limited even when increasing the number of agents. This suggests that, in general, it may be possible to obtain good solutions w.r.t. both the objectives.

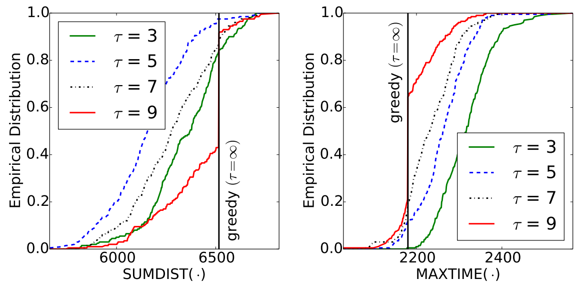
7 CONCLUSIONS

Computing a set of robot paths allowing to efficiently perform pairwise joint measurements is a recurrent problem in the multirobot systems literature. In this paper, we have addressed a systematic study of a graph-theoretical version of this problem, providing complexity results and efficient algorithms (with worst-case bounds on solution quality) suitable for practical settings.

An immediate avenue for future work is related to a deeper theoretical investigation of the combinatorial structure of the problem and, in particular, of the approximation guarantees that can be obtained. For instance, it would be interesting to extend the worst-case bounds of our greedy algorithm to settings where not all the possible pairs of vertices must be jointly visited and to further



(a) Open environment

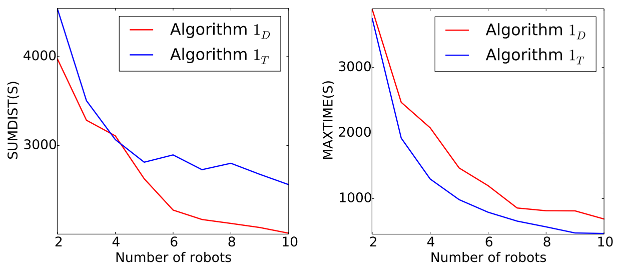


(b) Office environment

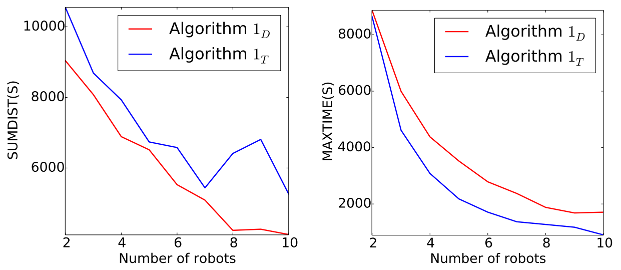
Figure 5: Empirical distributions of the solution costs found by Algorithms 1_D and 1_T.

	Algorithm 1_D						Algorithm 1_T					
	Range: 250 px		Range: 500 px		Range: 1000 px		Range: 250 px		Range: 500 px		Range: 1000 px	
Agents (m)	$\tau = \infty$	$\tau = 5$	$\tau = \infty$	$\tau = 5$	$\tau = \infty$	$\tau = 5$	$\tau = \infty$	$\tau = 5$	$\tau = \infty$	$\tau = 5$	$\tau = \infty$	$\tau = 5$
2	1710	1458	3970	3902	5074	5020	1368	1350	3745	3656	4793	4760
3	1658	**1362**	3284	**3264**	4418	**4138**	814	**729**	1922	**1865**	2491	**2416**
4	1520	**1296**	3108	**2714**	3444	**3274**	589	**544**	1298	**1224**	1662	**1598**
5	1488	**1278**	2626	**2356**	2892	**2840**	472	**394**	983	**924**	1228	**1159**
6	1398	**1344**	2274	**2036**	2636	**2516**	380	**339**	791	**712**	914	**871**
7	1400	**1360**	2168	**1932**	2430	**2328**	379	**318**	656	**593**	857	**739**
8	1666	**1452**	2124	**1928**	2326	**2188**	317	**287**	568	**524**	729	**653**
9	1600	**1542**	2078	**1950**	2300	**2166**	311	**285**	475	**465**	634	**565**
10	1686	**1596**	2014	**1902**	2346	**2148**	286	**278**	467	**424**	584	**525**
	Optimum for SUMDIST(·)						Optimum for MAXTIME(·)					
2	**1346**		**3498**		**4546**		**1263**		**3433**		**4506**	
	HKT for SUMDIST(·)						HKT for MAXTIME(·)					
3	1552		3524		4358		1022		1897		2484	

Table 1: Costs of solutions found by the algorithms in the open environment. Bold indicates the best results.



(a) Open environment



(b) Office environment

Figure 6: Solutions found by the pure greedy algorithms, evaluated according to both the objectives.

improve such bounds by devising constant-factor approximation algorithms.

From a practical standpoint, it could be worth focusing on a particular robotic application requiring to perform joint measurements and studying the relationship between online approaches and the proposed offline resolution scheme. For example, in a context of communication map building where the environment is fully known in advance, it would be interesting to compare the quality of the maps obtained with the online approach of [2] to the quality of those obtained with the help of the algorithms presented in this paper.

REFERENCES

[1] D. Applegate, R. Bixby, V. Chvatal, and W. Cook. 2011. *The traveling salesman problem: a computational study.* Princeton University Press.

[2] J. Banfi, A. Quattrini Li, N. Basilico, I. Rekleitis, and F. Amigoni. 2017. Multirobot online construction of communication maps. In *Proc. ICRA*. 2577–2583.

[3] N. Basilico, N. Gatti, M. Monga, and S. Sicari. 2014. Security games for node localization through verifiable multilateration. *IEEE T Depend Secure* 11, 1 (2014), 72–85.

[4] J. Bresina. 1996. Heuristic-biased stochastic sampling. In *Proc. AAAI*. 271–278.

[5] P. Brucker and P. Brucker. 2007. *Scheduling algorithms.* Vol. 3. Springer.

[6] S. Capkun and J. Hubaux. 2006. Secure positioning in wireless networks. *IEEE J Sel Area Comm* 24, 2 (2006), 221–232.

[7] B. Charrow, S. Liu, V. Kumar, and N. Michael. 2015. Information-theoretic mapping using cauchy-schwarz quadratic mutual information. In *Proc. ICRA*. 4791–4798.

[8] N. Christofides. 1976. *Worst-case analysis of a new heuristic for the travelling salesman problem.* Technical Report 388. Carnegie Mellon University.

[9] Y. Elmaliach, N. Agmon, and G. Kaminka. 2009. Multi-robot area patrol under frequency constraints. *Ann Math Artif Intel* 57, 3-4 (2009), 293–320.

[10] E. Galceran and M. Carreras. 2013. A survey on coverage path planning for robotics. *Robot Auton Syst* 61, 12 (2013), 1258–1276.

[11] M. Garey and D. Johnson. 1979. *Computers and intractability: A guide to the theory of NP-completeness.* W. H. Freeman.

[12] M. Hsieh, V. Kumar, and C. Taylor. 2004. Constructing radio signal strength maps with multiple robots. In *Proc. ICRA*. 4184–4189.

[13] L. Hu and D. Evans. 2004. Localization for mobile sensor networks. In *Proc. MobiCom*. 45–57.

[14] C. Kim, D. Song, Y. Xu, J. Yi, and X. Wu. 2014. Cooperative search of multiple unknown transient radio sources using multiple paired mobile robots. *IEEE Trans Robot* 30, 5 (2014), 1161–1173.

[15] A. Quattrini Li, F. Amigoni, and N. Basilico. 2012. Searching for optimal off-line exploration paths in grid environments for a robot with limited visibility. In *Proc. AAAI*. 2060–2066.

[16] A. Riva and F. Amigoni. 2017. A GRASP Metaheuristic for the Coverage of Grid Environments with Limited-Footprint Tools. In *Proc. AAMAS*. 484–491.

[17] C. Robin and S. Lacroix. 2016. Multi-robot target detection and tracking: taxonomy and survey. *Auton Robot* 40, 4 (2016), 729–760.

[18] D. Rosenkrantz, R. Stearns, and P. Lewis II. 1977. An analysis of several heuristics for the traveling salesman problem. *SIAM J Comput* 6, 3 (1977), 563–581.

[19] P. Slavík. 1996. A tight analysis of the greedy algorithm for set cover. In *Proc. STOC*. 435–441.

[20] P. Toth and D. Vigo. 2014. *Vehicle routing: problems, methods, and applications.* SIAM.

[21] N. Trawny and S. Roumeliotis. 2010. On the global optimum of planar, range-based robot-to-robot relative pose estimation. In *Proc. ICRA*. 3200–3206.

[22] A. Visser and B. Slamet. 2008. Balancing the information gain against the movement cost for multi-robot frontier exploration. In *Proc. EUROS*. 43–52.

[23] Z. Yan, N. Jouandeau, and A. Cherif. 2013. A survey and analysis of multi-robot coordination. *Int J Adv Robot Syst* 10, 12 (2013), 399.

[24] X. Zhou and S. Roumeliotis. 2008. Robot-to-robot relative pose estimation from range measurements. *IEEE Trans Robot* 24, 6 (2008), 1379–1393.