On Cyclic Solutions to the Min-Max Latency Multi-Robot Patrolling Problem

Peyman Afshani
Department of Computer Science, Aarhus University, Denmark

Mark de Berg
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Kevin Buchin
Department of Computer Science, TU Dortmund, Germany

Jie Gao
Department of Computer Science, Rutgers University, New Brunswick, NJ, USA

Maarten Löffler
Department of Information and Computing Sciences, Utrecht University, The Netherlands

Amir Nayyeri
School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, USA

Benjamin Raichel
Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA

Rik Sarkar
School of Informatics, University of Edinburgh, UK

Haotian Wang
Department of Computer Science, Rutgers University, New Brunswick, NJ, USA

Hao-Tsung Yang
School of Informatics, University of Edinburgh, UK

Abstract

We consider the following surveillance problem: Given a set P of n sites in a metric space and a set R of k robots with the same maximum speed, compute a *patrol schedule* of minimum latency for the robots. Here a patrol schedule specifies for each robot an infinite sequence of sites to visit (in the given order) and the latency L of a schedule is the maximum latency of any site, where the latency of a site s is the supremum of the lengths of the time intervals between consecutive visits to s.

When $k = 1$ the problem is equivalent to the travelling salesman problem (TSP) and thus it is NP-hard. For $k \geqslant 2$ (which is the version we are interested in) the problem becomes even more challenging; for example, it is not even clear if the decision version of the problem is decidable, in particular in the Euclidean case.

We have two main results. We consider *cyclic solutions* in which the set of sites must be partitioned into ℓ groups, for some $\ell \leqslant k$, and each group is assigned a subset of the robots that move along the travelling salesman tour of the group at equal distance from each other. Our first main result is that approximating the optimal latency of the class of cyclic solutions can be reduced to approximating the optimal travelling salesman tour on some input, with only a $1 + \varepsilon$ factor loss in the approximation factor and an $O\left((k/\varepsilon)^k\right)$ factor loss in the runtime, for any $\varepsilon > 0$. Our second main result shows that an optimal cyclic solution is a $2(1 - 1/k)$-approximation of the overall optimal solution. Note that for $k = 2$ this implies that an optimal cyclic solution is optimal overall. We conjecture that this is true for $k \geqslant 3$ as well.

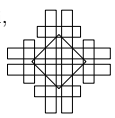
The results have a number of consequences. For the Euclidean version of the problem, for instance, combining our results with known results on Euclidean TSP, yields a PTAS for approximating an optimal cyclic solution, and it yields a $(2(1 - 1/k) + \varepsilon)$-approximation of the optimal unrestricted (not necessarily cyclic) solution. If the conjecture mentioned above is true, then our algorithm is actually a PTAS for the general problem in the Euclidean setting. Similar results can be obtained by combining our results with other known TSP algorithms in non-Euclidean metrics.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Approximation, Motion Planning, Scheduling

Digital Object Identifier 10.4230/LIPIcs.SoCG.2022.2

Related Version *Full Version*: http://arxiv.org/abs/2203.07280



38th International Symposium on Computational Geometry (SoCG 2022).
Editors: Xavier Goaoc and Michael Kerber; Article No. 2; pp. 2:1–2:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Funding *Mark de Berg*: Supported by the Dutch Research Council (NWO) through Gravitation-grant NETWORKS-024.002.003.
Jie Gao: This work is supported by NSF OAC-1939459, CCF-2118953 and CCF-1934924.
Benjamin Raichel: Partially supported by NSF CAREER Award 1750780.

1 Introduction

We study the following problem, motivated by the problem of monitoring a fixed set of locations using autonomous robots: We are given a set $P = \{s_1, \cdots, s_n\}$ of n sites in a metric space as well as a set $R = \{r_1, \cdots, r_k\}$ of k robots. We assume the robots have the same maximum speed, called the *unit speed*, and their task is to repeatedly visit (i.e., survey) the sites such that the maximum time during which any site is left unmonitored is minimized. More precisely, we wish to compute a *patrol schedule*; that is, an infinite sequence of sites to visit for each robot, of minimum *latency*. Here the latency of a site s_i is the supremum of the length of the time intervals between consecutive visits of s_i, and the latency of the patrol schedule is the maximum latency over all the sites.

Related Work. For $k = 1$, the problem reduces to the Traveling Salesman Problem. To see this, consider the time interval $[0, 3L]$, where L is the optimal latency, and observe that every site is visited at least twice by the robot in this time interval. Let $L' \leqslant L$ be the maximum length of time between two consecutive visits of a site. Then there exists a site that is visited at times t_0 and $t_0 + L'$ and all other sites are visited at least once in the time interval $(t_0, t_0 + L')$. Hence, if an optimal solution has latency L, there is a TSP tour of length at most L. The converse is clearly true as well – by repeatedly traversing a TSP tour of length L we obtain a patrol schedule of latency L – and so the TSP problem is equivalent to the patrol problem for a single robot. Since TSP is NP-hard even in the Euclidean case [16] we will focus on approximation algorithms. There are efficient approximation algorithms for TSP and, hence, for the patrolling problem for $k = 1$. In particular, there is a $(3/2)$-approximation for metric TSP [5] (which was slightly improved very recently [10]) and a PTAS for Euclidean TSP [4, 15]. However, it seems difficult to generalize these solutions to the case $k \geqslant 2$, because it seems non-trivial to get a grip on the structure of optimal solutions in this case. We will mention some of the major challenges shortly.

There has been a lot of work on such surveillance problems in the robotics community [7, 9, 14, 21, 17, 18]. Most previous work, however, focused on either practical settings or aspects of the problems other than finding the best approximation factor. There are two papers that provide theoretical guarantees for the weighted version of the problem, where sites of higher weight require more frequent patrols. Alamdari et al. [2] provided a $O(\log n)$-approximation algorithm for the weighted problem for $k = 1$. (Due to existence of weights, a TSP tour may no longer be optimal for $k = 1$.) Afshani et al. [1] studied the problem for $k \geqslant 1$ and they present an $O(k^2 \log \frac{w_{\max}}{w_{\min}})$-approximation algorithm, where $w_{\max}$ and $w_{\min}$ are the maximum and the minimum weights of the sites.

Related Problems. As already mentioned, the TSP problem can be viewed as a special case of the problem for unweighted sites and for $k = 1$. Another related problem is the *k-path cover* problem where we want to find k paths that cover the vertices of an edge-weighted graph such that the maximum length of the paths is minimized. This problem has a 4-approximation algorithm [3]. Another problem is the problem of covering all the sites with k trees that minimize the maximum length of the trees; this problem is known as the *min-max tree cover*

problem and it has constant-factor approximation algorithms [3, 13] with 8/3 being the current record [20]. The *k-cycle cover* problem is similar, except that we want to use k cycles (instead of paths or trees); again constant-factor approximation algorithms are known, with 16/3 being the current record [20]. If the goal is to minimize the sum of all cycle lengths, there is a 2-approximation for the metric setting and a PTAS in the Euclidean setting [11, 12]. Our problem is also related to (but different from) the *vehicle routing problem* (VRP) [6], which asks for k tours, starting from a given depot, that minimize the total transportation cost under various constraints; see the surveys by Golden et al. [8] or Tóth and Vigo [19].

Our Results. All covering problems mentioned above are obviously decidable. The question of decidability for the patrolling problem seems non-trivial. However, since patrol schedules are infinite sequences and thus it is not even clear how to guess a solution[1]. To tackle this issue, we consider the class of *cyclic solutions*. In a cyclic solution the set P of sites is partitioned into $\ell \leqslant k$ subsets $P_1, \cdots, P_\ell$, and each subset P_i is assigned k_i robots, where $\sum_{i=1}^{\ell} k_i = k$. The k_i robots are then distributed evenly along a TSP tour of P_i, and they traverse the tour at maximum speed. Thus, the latency of the sites in P_i equals $\|T_i\|/k_i$, where $\|T_i\|$ is the length of the TSP tour of P_i.

The significance of this definition is that in Section 3 we prove that (in any metric space) the best cyclic solution is a $2(1-1/k)$-approximation of the optimal solution in terms of maximum latency. We do this by transforming an optimal solution to a cyclic one, with only a $2(1-1/k)$ factor loss in the approximation ratio. This proof is highly non-trivial and involves a number of graph-theoretic arguments and carefully inspecting the coordinated motion of the k robots, cutting them up at proper locations, and re-gluing the pieces together to form a cyclic solution. In combination with this, in Section 4 we prove that, given a γ-approximation algorithm for TSP, for any fixed k and $\varepsilon > 0$, we can obtain a $(1+\varepsilon)\gamma$-approximation of the best cyclic schedule in polynomial time. Therefore, in the Euclidean setting, we can use a known PTAS to obtain a $(1+\varepsilon)$-approximation to the *best cyclic* solution and in the general metric setting, we can use known approximation algorithms for TSP [10] to get a 1.5-approximation to the *best cyclic* solution. Together with the results in Section 3 these lead to a $(2-2/k+\varepsilon)$-approximation algorithm for the Euclidean case, and a $(3-3/k)$-approximation for general metrics.

We conjecture that the best cyclic solution is in fact the best overall solution. If this is true, then our algorithm in Section 4 already gives a PTAS in the Euclidean setting. Observe that a corollary of our result in Section 3 is that the conjecture holds for $k = 2$. We remark that there is an easy proof showing the existence of a cyclic 2-approximation solution (See Section 2.2). Our new bound $2(1-1/k)$ is a significant improvement when k is a small constant. For example, for $k = 3$, we get that a cyclic 4/3 approximate solution exists, and for $k = 2$ –as mentioned above– that there is a cyclic optimal solution.

2 Challenges, Notation, and Problem Statement

2.1 Notation and Problem Statement

Let (P, d) be a metric space on a set P of n sites, where the distance between two sites $s_i, s_j \in P$ is denoted by $d(s_i, s_j)$. Following Afshani et al. [1], we model the metric space in the following way. We take the undirected complete graph $G = (P, P \times P)$, and we view each edge

[1] If we assume that all distances are integers and we want to decide whether the latency is at most a given integer ℓ, then we can guess a solution. These assumptions, however, do not hold in the Euclidean case, even if the coordinates of sites are rational.

$(s_i, s_j) \in P \times P$ as an interval (that is, a continuous 1-dimensional space) of length $d(s_i, s_j)$ in which the robot can travel. This transforms the discrete matric space (P, d) into a continuous metric space $C(P, d)$. From now on, and with a slight abuse of terminology, when we talk about the metric space (P, d) we actually mean the continuous metric space $C(P, d)$.

We allow the robots to "stay" on a site for any amount of time. This implies it never helps if a robot moves slower than the maximum speed: indeed, the robot may as well move at maximum speed towards the next site and stay a bit longer at that site. Also, it does not help to have a robot start at time $t = 0$ "in the middle" of an edge, so we can assume all robots start at some sites at the beginning. A *schedule* of a robot r_j is defined as a continuous function $f_j : \mathbb{R}^{\geqslant 0} \to C(P, d)$, where $f_j(t)$ specifies the position of r_j at time t. The unit-speed constraint implies that a valid schedule must satisfy $d(f_j(t_1), f_j(t_2)) \leqslant |t_1 - t_2|$ for all t_1, t_2. A *schedule for the collection R of robots*, denoted by $\sigma(R)$, is a collection of schedules f_j, one for each robot $r_j \in R$. Note that we allow robots to be at the same location at the same time.

We say that a site $s_i \in P$ is *visited* at time t if $f_j(t) = s_i$ for some robot r_j. Given a schedule $\sigma(R)$, the *latency* L_i of a site s_i is defined as follows.

$$L_i = \sup_{0 \leqslant t_1 < t_2} \{|t_2 - t_1| : s_i \text{ is not visited during the time interval } (t_1, t_2)\}$$

We only consider schedules where the latency of each site is finite. Clearly such schedules exist; e.g., a robot can repeatedly traverse a TSP tour of the sites. Given a metric space (P, d) and a collection R of k robots, the *(multi-robot) patrol-scheduling problem* is to find a schedule $\sigma(R)$ minimizing the latency $L := \max_i L_i$, the maximum latency of any site.

2.2 Challenges

The problem of scheduling multiple robots is quite challenging and involves several subtleties, caused by the fact that patrol schedules are infinite sequences. For example, the time intervals between consecutive visits of any given site might increase continuously, and so we have to define the latency of a site using the notion of supremum rather than maximum. Moreover, for $k > 1$, it is not even clear if the problem is decidable: Given a set of n points in the Euclidean plane, an integer $k > 1$, and a value L, is it decidable if there exists a patrol schedule for the k robots such that the maximum latency is bounded by L? As already mentioned, a corollary of our results in Section 3 is that for $k = 2$ there exists an optimal cyclic solution and thus for $k = 2$ the answer to the above question is yes.

A severe challenge is that, since patrol schedules are infinite sequences, it is difficult to rule out chaotic solutions where the robots visit the sites in a way that avoids any sort of repeated pattern. Indeed, optimal solutions can behave so chaotically that they require an infinite sequence of bits to describe. For instance, consider the left situation in Figure 1,

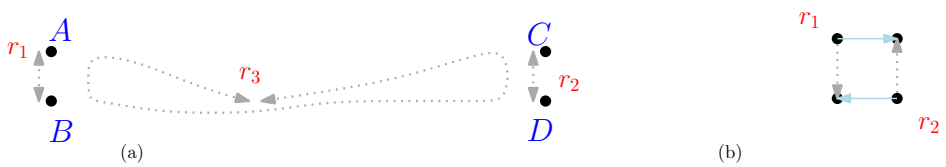


Figure 1 (a) Four points A, B, C, and D form a short and wide rectangle. Robots r_1, r_2, and r_3 can have infinite "unpredictable" optimal patrol schedules. (b) Robots r_1 and r_2 can move to the other diagonal in two different ways.

where we have three robots and four points A, B, C, D that are the vertices of a thin rectangle. To obtain the optimal latency, it suffices that r_1 moves back and forth between A and B, and r_2 moves back and forth between C and D. Since r_3 cannot be used to decrease the latency – it will take r_3 too much time to go from A, B to C, D – it can behave as chaotically as it wants, thus causing the description of the patrol schedule to be arbitrarily complicated. This is even possible using only two robots: Consider four sites that form a unit square and two robots placed on opposite corners of the square; see the right situation in Figure 1. An optimal schedule is then an infinite sequence of steps, where in each step both robots move counterclockwise or both move clockwise. Such a schedule need not be cyclic and, hence, may require an infinite sequence of bits to describe. Of course, in both cases we know optimal cyclic solutions exist, and such solutions can be described using finitely many bits. We conjecture that this should be true in general:

▶ **Conjecture 1.** *For the k-robot patrolling problem with min-max latency, there is a cyclic solution that is optimal.*

It is easy to see that there exists a cyclic solution that is a 2-approximation: take an optimal schedule with latency L, and at time L move the robots back to their respective starting positions at time 0, and repeat. The challenge lies in getting an approximation factor smaller than 2, which we achieve in Section 3 where we show that there is a cyclic solution that is a $2(1 - 1/k)$ approximation.

3 Turning an Optimal Solution into a Cyclic Solution

The main goal of this section is to prove the following theorem.

▶ **Theorem 2.** *Let L be the latency in an optimal solution to the k-robot patrol-scheduling problem in a metric space (P, d). There is a cyclic solution with latency at most $2(1 - 1/k)L$.*

We prove the theorem by considering an optimal (potentially "chaotic") solution and turning it into a cyclic solution. This is done by first identifying a certain set of "bottleneck" sites within a time interval of length L, then cutting the schedules into smaller pieces, and then gluing them together to obtain the final cyclic solution. This will require some graph-theoretic tools as well as several new ideas (see Appendix in the full-version paper).

Below we sketch the main ideas of the proof; the full proof can be found in Appendix of the full-version paper.

3.1 Bidirectional sweep to find "bottleneck" sites

Consider an optimal patrol schedule with latency L, and consider a time interval $\mathcal{I} := [t_0, t_0 + L]$ for an arbitrary $t_0 > 2L$. By our assumptions, every site is visited at least once within this time interval. We assign a time interval $I_i \subseteq \mathcal{I}$ to every robot r_i. Initially $I_i = \mathcal{I}$. To identify the important sites that are visited by the robots, using a process that we will describe shortly, we will *shrink* each I_i. Shrinking is done by moving the left and right endpoints of I_i "inward" at the same speed. This will be done in multiple stages and at the end of each stage, an endpoint of some intervals could become *fixed*; a fixed endpoint does not move anymore during the following stages. When both endpoints of I_i are fixed, we have found the final shrunken interval for r_i. Initially, all the endpoints are *unfixed*. For an interval $I_i = [t_i, t'_i]$, shrinking I_i by some value $\varepsilon \geqslant 0$ yields the interval $[t_i + \varepsilon\varphi_1, t'_i - \varepsilon\varphi_2]$ where φ_1 (resp. φ_2) is 1 if the left (resp. right) endpoint of I_i is unfixed, otherwise it is 0. Note that an interval $[x, y]$ with $x > y$ is considered *empty* (i.e., an empty set) and thus shrinking an interval by a large enough value will yield an empty interval (assuming at least one endpoint is unfixed).

The invariant. We maintain the invariant that at the beginning of each stage of the shrinking process, all the sites are visited during the shrunken intervals, i.e., for every site $s \in P$, there exists a robot r_i, and a value $t \in I_i$ such that $f_i(t) = s$. Observe that the invariant holds at the beginning of the first stage of the shrinking process.

The shrinking process. Consider the j-th stage of the shrinking process. Let $\varepsilon_j \geqslant 0$ be the largest (supremum) number such that shrinking all the intervals by ε_j respects the invariant. If ε_j is unbounded, then this is the last stage; every interval I_i that has an unfixed endpoint is reduced to an *empty interval* and we are done with shrinking, meaning, the shrinking process has yielded some $k' \leqslant k$ intervals with both endpoints fixed, and $k - k'$ empty intervals. Otherwise, ε_j is bounded and well-defined as the invariant holds for $\varepsilon_j = 0$. With a slight abuse of the notation, let $I_1, \cdots, I_k$ be the intervals shrunken by ε_j. See Figure 2(left).

Since ε_j is the largest value that respects our invariant, it follows that there must be at least one interval I_{i_j} and at least one of its endpoints t_j such that at time t_j, the robot r_{i_j} visited the site $f_{i_j}(t_j)$ and this site is not visited by any other robot in the interior of their time intervals. Now this endpoint of I_{i_j} is marked as fixed and we continue to the next stage.

For a fixed endpoint A, let $\ell(A)$ be the distance of A to the corresponding boundary of the unshrunk interval. More precisely, if A is a left endpoint then the position of A on the time axis is $t_0 + \ell(A)$, and if A is a right endpoint then this position is $t_0 + L - \ell(A)$. With our notation, if A was discovered at stage j, then $\ell(A) = \varepsilon_1 + \cdots + \varepsilon_j$.

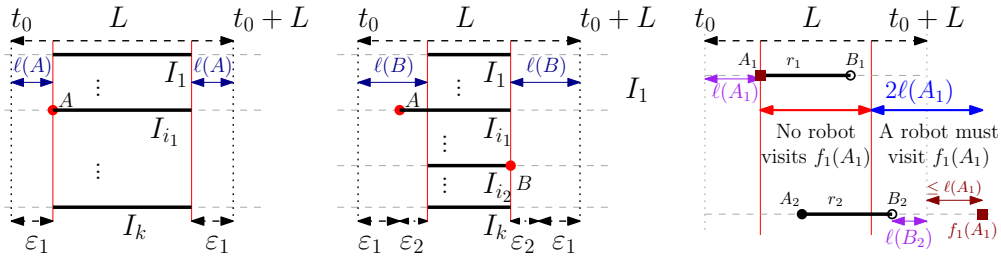


Figure 2 (left) A is fixed at stage 1. (middle) B is fixed at stage 2. (right) By the property of the shrinking process, the site visited at A_1 is not visited by any robot within the red time interval but since the site has latency at most L, it must be visited by some robot in the blue interval.

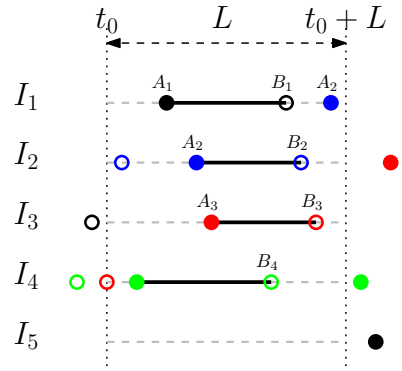
3.2 Patrol graph, shortcut graph, and bag graph

Shortcutting idea. Figure 2 (right) explains the crucial property of our shrinking process: Robot r_1 visits the site $p = f_1(A_1)$ at time A_1 (which corresponds to the left endpoint of the interval I_1) but to keep the latency of p at most L, p must be visited by another robot, say r_2, sometime in the interval $[t_0 + L - \ell(A_1), t_0 + L + \ell(A_1)]$, shown in blue in the figure. For the moment, assume the right endpoint of the interval of r_2 is a fixed point B_2 and r_2 visits a site $p' = f_2(B_2)$ at time B_2. This implies that the distance between p and p' is at most $\ell(A_1) + \ell(B_2)$. Now observe that we can view this as a "shortcut" between endpoints p and p': for example, r_2 can follow its own route from A_2 to B_2, then take the shortcut to A_1, and then follow r_1's route to B_1. The extra cost of taking the shortcut, which is $\ell(A_1) + \ell(B_2)$, can also be charged to the two "shrunken" pieces of the two intervals (the purple intervals in the picture). Our main challenge is to show that these shortcuts can be used to create a cyclic solution with only a small increase in the latency.

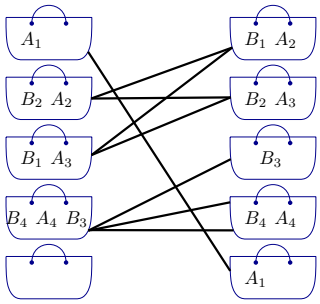
To do that, we will define a number of graphs associated with the shrunken intervals. We define a *patrol graph* $\mathcal{P}$, a *bag graph* $\mathcal{B}$ and a *shortcut graph* $\mathcal{S}$. The first two are multigraphs, whereas the shortcut graph is a simple graph.

For examples see Figure 3 on page 7 and its discussion on page 8.

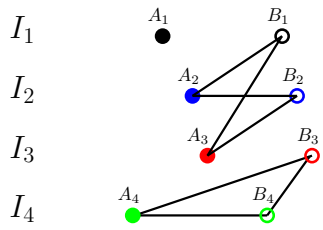
We start with the bag graph and the shortcut graph. We first shrink the intervals as described previously. To define these graphs, consider $2k$ conceptual *bags*, two for each interval (including the empty intervals). More precisely, for each interval we have one *left bag* and one *right bag*. The bags are the vertices of the bag graph $\mathcal{B}$ (Figure 3(b)). The vertices of the shortcut graph $\mathcal{S}$ are the endpoints of the non-empty intervals. To define the edges of the two graphs, we use *placements*. We will present the details below but basically, every endpoint of a non-empty interval will be placed in two bags, one in some left bag β_1 and another time in some right bag β_2. After this placement, we add an edge in the bag graph between β_1 and β_2. Once all the endpoints have been placed, we add edges in the shortcut graph between every two endpoints that have been placed in the same bag (Figure 3(c)).



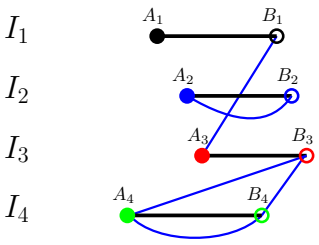
(a) After the shrinking process



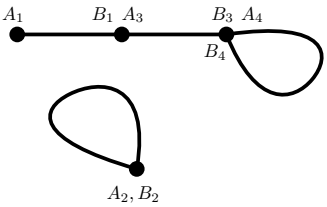
(b) The bag graph



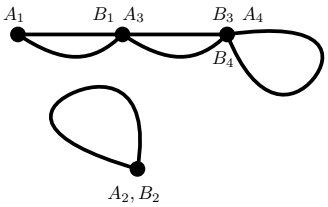
(c) The shortcut graph



(d) The patrol graph



(e) After contracting the blue edges



(f) Eulerized

Figure 3 Examples of bag, shortcut and patrol graph.

An example of a bag and shortcut graphs. An example is shown in Figure 3. In part (a), we have four non-empty intervals $I_1 = [A_1, B_1], I_2 = [A_2, B_2], I_3 = [A_3, B_3], I_4 = [A_4, B_4]$ and an empty interval I_5 (we will later explain the second appearance of each endpoint in this picture and for now the reader can ignore the "floating" endpoints). An example of a bag graph is shown in Figure 3(b): Every endpoint is placed twice (once in some left bag and one is some right bag). E.g., A_1 is placed in the top-left bag and the bottom-right bag and thus the two bags are connected in the bag graph. Similarly, B_1 is placed in two bags, once at the top-right bag and the other time at the mid-left bag. In part (c) of the figure, one can see the shortcut graph in which two endpoints are connected if and only if they are placed in the same bag. Also, this is a simple graph and despite the fact that A_4 and B_4 are placed together in two different bags, they are still connected once in the shortcut graph.

Initially, all the bags are empty. For every non-empty interval $I_1 = [A_1, B_1]$, we place the left endpoint of I_1 in its own left bag and the right endpoint of I_1 in its own right bag. This is the first placement. For the second placement, consider a non-empty interval I_{i_1} and its left endpoint A. The position of A on the time interval is $t = t_0 + \ell(A)$. See Figure 2(right). By our assumptions, the robot r_{i_1} visits the site $p = f_{i_1}(t)$ at time t. Consider the stage of our shrinking process when A gets fixed. For this to happen, the site p cannot be visited by any robot in the time interval $(t_0 + \ell(A), t_0 + L - \ell(A))$ (the red interval in Figure 2(right)), as otherwise, we could either shrink all the intervals by an infinitesimal additional amount or some other endpoint would have been fixed. On the other hand, this site has latency at most L, so it must be visited by another robot in the time interval $(t, t + L] = (t_0 + \ell(A), t_0 + \ell(A) + L]$. This means that the robot r_j that visits p earliest in this interval must do so within the time interval $[t_0 + L - \ell(A), t_0 + L + \ell(A)]$ (the blue interval in Figure 2(right)). Note that r_j could be any of the robots, including r_{i_1} itself. We now place A in the right bag of I_j.

A very similar strategy is applied to the right end point of I_1; for details see Appendix of the full-version paper, where we also prove the following properties.

▶ **Lemma 3.** *The bag graph $\mathcal{B}$ and the shortcut graph $\mathcal{S}$ have the following properties.*

(a) *$\mathcal{B}$ is a bipartite graph.*

(b) *$\mathcal{S}$ is isomorphic to the line graph of $\mathcal{B}$*

(c) *Let $\mathcal{B}'$ be a connected component of $\mathcal{B}$. If none of the vertices of $\mathcal{B}'$ belong to empty-intervals, then the number of vertices of $\mathcal{B}'$ is equal to its number of edges.*

(d) *Let $\mathcal{S}'$ be a connected component of $\mathcal{S}$. If $\mathcal{S}'$ has a vertex v of degree one, then it must be the case that v corresponds to an endpoint of a non-empty interval that has been placed (alone) in a bag of an empty interval.*

The patrol graph. The above lemma, combined with the graph theoretical tools that we outline in Appendix allows us to define the patrol graph $\mathcal{P}$. Here, we only give an outline, and for the full details see Appendix of the full version paper. An example of a patrol graph is shown in Figure 3(d). Initially, the patrol graph, $\mathcal{P}$, consists of k' isolated black edges, one for each non-empty interval. Observe that both $\mathcal{P}$ and the shortcut graph $\mathcal{S}$ have the same vertex set (endpoints of the non-empty intervals). We add a subset of the edges of the shortcut graph to $\mathcal{P}$. Let us consider an "easy" case to illustrate the main idea.

An easy case. Assume $\mathcal{B}$ is connected and that it has an even number of edges. In this case, we can in fact prove that an optimal cyclic solution exists. Recall that $\mathcal{S}$ is the line graph of $\mathcal{B}$ and it is known that the line graph of a connected graph with even number of edges, has a perfect matching. Thus, we can find a perfect matching M as a subset of edges of $\mathcal{S}$. Add M to $\mathcal{P}$ as "blue" edges. Now, every vertex of $\mathcal{P}$ is adjacent to a blue and a black edge and thus

$\mathcal{P}$ decomposes into a set of "bichromatic" cycles, i.e., cycles with alternating black-blue edges. With a careful accounting argument, we can show that this indeed yields a cyclic solution without increasing the latency of any of the sites. We have already mentioned the main idea under the "shortcutting idea" paragraph, at the beginning of the section. Specifically, we will use the following lemma.

▶ **Lemma 4.** *Consider two adjacent vertices v and w in the shortcut graph. This means that there are two non-empty intervals I_1 and I_2 such that v corresponds to an endpoint A of I_1 and w corresponds to an end point B of I_2 and A and B are placed in the same bag. Let s_1 be the site visited at A during I_1 and s_2 be the site visited at B on I_2. Then, we have $d(s_1, s_2) \leqslant \ell(A) + \ell(B)$.*

Black edges represent the routes of the robots, and blue edges are the shortcuts that connect one route to another. So in this easy case, once the patrol graph has decomposed into bichromatic cycles, we turn each cycle into one closed route (i.e., cycle) using the shortcuts. All the robots that correspond to the black edges are placed evenly on this cycle. Since by our invariant all the sites are visited at some time on the black edges, it follows that the robots visit all the sites. A careful accounting argument shows that the cost of taking the shortcuts is upper bounded by the value $\ell(\cdot)$ of the end points involved. Since the values $\ell(\cdot)$ correspond to the length of the sub-paths of the robots that is missing due to our shrinking process, one can show that the latency does not increase at all.

Unfortunately, $\mathcal{B}$ can have connected components with odd number of edges. Nonetheless, in all cases we can build a particular patrol graph, $\mathcal{P}$, with the following properties.

▶ **Lemma 5.** *The patrol graph $\mathcal{P}$ consists of k' pairwise non-adjacent black edges and a number of blue edges. Any blue edge (v, w) in $\mathcal{P}$ corresponds to an edge in the shortcut graph $\mathcal{S}$. Furthermore, the set of blue edges can be decomposed into a matching and a number of triangles. In addition, any vertex of $\mathcal{P}$ that is not adjacent to a blue edge can be charged to a bag of an empty interval.*

The idea covered in the above "easy case" works because it covers the black edges of $\mathcal{P}$ with bichromatic edge-disjoint cycles and each cycle becomes a cyclic route. Unfortunately, in general $\mathcal{P}$ might not have the structure that would allow us to do this. Here, we only outline the steps we need to overcome this: we consider each connected component, $\mathcal{P}_i$, of $\mathcal{P}$. We first contract blue edges of $\mathcal{P}_i$ to obtain a contracted patrol graph, $\mathcal{P}_i^c$, (Figure 3(e)) then we eulerize it (Figure 3(f)), meaning, we duplicate a number of black edges such that the resulting graph is Eulerian. This yields us an Eularized contracted patrol graph, $\mathcal{P}_i^{Ec}$ (Figure 4(a)). Next, we put the contracted blue edges back in $\mathcal{P}_i^{Ec}$ which gives us the final patrol graph (Figure 4(b)). In this final graph, we can show that we can cover the black edges using bichromatic edge disjoint cycles where each connected component of the final graph turns into one cycle (Figure 4(c)); this yields us a cyclic solution. However, the duplicated black edges represent routes of robots that need to be traversed twice to obtain the cyclic solution. This leads us to the final challenge: how to allocate the robots to the resulting cycles to minimize the latency. With some careful accounting and considering a few cases, we can show that this can be done in such a way that the resulting cyclic solution has latency at most $2L(1 - 1/k)$. We do this in Appendix of the full-version paper, proving Theorem 2.

4 Cyclic Solutions

In this section we show how to approximate an optimal cyclic solution to the patrol scheduling problem for k robots in a metric space (P, d). We start with some notation and basic observations.

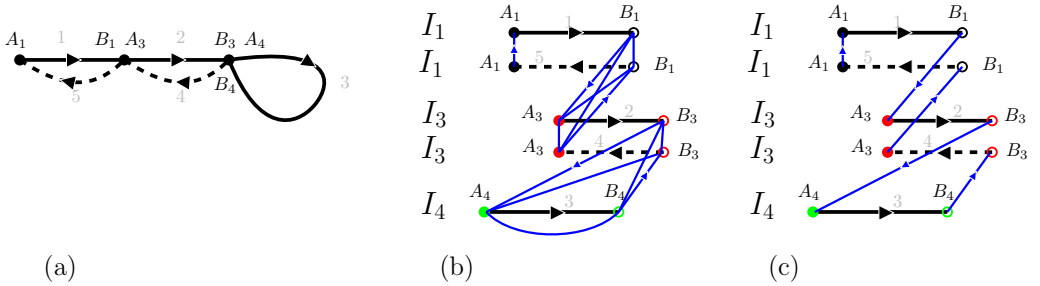


Figure 4 (a) An Eularized contracted patrol graph (ECPG). Duplicated edges are drawn with dashed lines. The edges are directed and numbered according to an Euler tour. (b) The same ECPG after "uncontracting" the blue edges. The duplicated routes (i.e., the routes that will be traversed twice) are shown with dashed lines. The corresponding Euler tour is marked and numbered. The shortcuts are taken in the correct direction. (c) The bichromatic cycle gives us a cyclic route. It shows how the robots can travel along it.

For a subset $Q \subseteq P$, let $\mathrm{TSP}(Q)$ denote an optimal TSP tour of Q and let $tsp(Q)$ denote its total length. Let $\mathrm{MST}(Q)$ denote a minimum spanning tree of Q. Now consider a partition $\Pi = \{P_1, \ldots, P_t\}$ of P, where each subset P_i is assigned k_i robots such that $\sum_{i=1}^{t} k_i = k$. A *cyclic solution* for this partition and distribution of robots is defined as follows. For each P_i there is a cycle C_i such that the k_i robots assigned to P_i start evenly spaced along C_i and then traverse C_i at maximum speed in the same direction. Hence, the latency L of such a cyclic solution satisfies $L \geqslant \max_i(tsp(P_i)/k_i)$, with equality if $C_i = \mathrm{TSP}(P_i)$ for all i.

To prove the main theorem of this section we need several helper lemmas. Let $\Pi = (P_1, \ldots, P_t)$ be a partition of P and let $E \subseteq P \times P$ be a set of edges. The *coarsening of* Π *with respect to* E is the partition Π' of P given by the connected components of the graph $(\bigcup_i \mathrm{MST}(P_i)) \cup E$.

▶ **Lemma 6.** *Let S be a cyclic solution with partition $\Pi = (P_1, \ldots, P_t)$ and latency L. Let $\Pi' = (P'_1, \ldots, P'_{t'})$ be the coarsening of Π with respect to an edge set E of total length ℓ. Then there is a cyclic solution S' with partition Π' and latency L' such that $L' \leqslant L + \ell$.*

Proof. Let $C_1, \ldots, C_t$ be the cycles used in S. Consider a subset $P'_i \in \Pi'$, and assume without loss of generality that P'_i is the union of the subsets $P_1, \ldots, P_s$ from Π. Then there is a set $E_i \subseteq E$ of $k-1$ edges such that $\left(\bigcup_{j=1}^{s} C_j\right) \cup E_i$ is connected. Moreover, there is a cycle C'_i covering all sites in P'_i traversing the edges of each C_j once and the edges of E_i twice. Hence,

$$\|C'_i\| = \sum_{j=1}^{s} \|C_j\| + 2 \cdot \|E_i\|,$$

where $\|\cdot\|$ denotes the total length of a set of edges. Since the latency in S is L, we know that $\|C_j\| \leqslant k_j L$. Hence, using $\sum_{j=1}^{s} k_j \geqslant 2$ robots for the cycle C'_i, the latency for the sites in P'_i is at most

$$\frac{\|C'_i\|}{\sum_{j=1}^{s} k_j} = \frac{\sum_{j=1}^{s} \|C_j\| + 2 \cdot \|E_i\|}{\sum_{j=1}^{s} k_j} \leqslant \frac{\sum_{j=1}^{s} k_j L + 2 \cdot \|E_i\|}{\sum_{j=1}^{s} k_j} \leqslant L + \|E_i\| \leqslant L + \ell.$$

Thus the latency for any subset $P'_i \in \Pi'$ is at most $L + \ell$. ◀

▶ **Lemma 7.** *Let L^* be the latency of an optimal cyclic solution. For any $\varepsilon > 0$, there exists a cyclic solution with partition $\Pi = (P_1, \ldots, P_t)$ and latency $L < (1+\varepsilon)L^*$ such that for any pair $i \neq j$ we have $d(P_i, P_j) > \varepsilon \cdot L^*/k$, where $d(P_i, P_j) := \min\{d(x,y) : x \in P_i \text{ and } y \in P_j\}$.*

Proof. Let S^* be an optimal solution with partition $\Pi^* = (P_1^*, \ldots, P_q^*)$, where $q \leqslant k$. Let E_{short} be the set of all edges of the complete graph of the metric space with length at most $\varepsilon L^*/k$. Let $\Pi = (P_1, \ldots, P_t)$ be the partition obtained by coarsening Π^* with respect to E_{short}, and let $E^* \subseteq E_{\text{short}}$ be a minimal subset such that coarsening Π^* with E^* gives the same partition Π. Observe that as $q \leqslant k$, we have $|E^*| \leqslant k-1$. Lemma 6 implies that there is a cyclic solution S with partition Π and latency at most

$$L^* + |E^*| \cdot (\varepsilon L^*/k) < (1+\varepsilon)L^*.$$

Moreover, since Π is a coarsening of Π^* with respect to E_{short}, the pairwise distance between any two sets of Π is larger than $\varepsilon L^*/k$. ◀

▶ **Lemma 8.** *Suppose there is a cyclic solution of latency L for a given metric space (P, d) and k robots. Then $\text{MST}(P)$ has fewer than $k(1+1/\alpha)$ edges of length more than αL, for any $0 < \alpha \leqslant 1$.*

Proof. Let $C_1, \ldots, C_q$ be the cycles in the given cyclic solution of latency L, let k_i denote the number of robots assigned to C_i, and let $P_i \subset P$ be the sites in C_i. Let E be a subset of $q-1 \leqslant k-1$ edges from $\text{MST}(P)$ such that $(\bigcup_{i=1}^{q} C_i) \cup E$ is connected. Then

$$\sum_{i=1}^{q} \|C_i\| > \|\text{MST}(P)\| - \|E\| = \|MST(P) \setminus E\|$$

Since $\|C_i\| \leqslant k_i L$, we have $\sum_{i=1}^{q} \|C_i\| \leqslant kL$. Hence, $\|\text{MST}(P) \setminus E\| < kL$, which implies that $\text{MST}(P) \setminus E$ contains less than k/α edges of length more than αL. Including the edges in E, we thus know that $\text{MST}(P)$ has less than $k(1+1/\alpha)$ edges of length more than αL. ◀

▶ **Theorem 9.** *Suppose we have a γ-approximation algorithm for TSP in a metric space (P, d), with running time $\tau_\gamma(n)$, and an algorithm for computing an MST that runs in time $T'(n)$. Then there is a $(1+\varepsilon)\gamma$-approximation algorithm for finding a minimum-latency cyclic patrol schedule with k robots that runs in $T'(n) + (O(k/\varepsilon))^k \cdot \tau_\gamma(n)$ time.*

Proof. Let L^* be the latency in an optimal cyclic solution. By Lemma 7 there is a solution S with latency $(1+\varepsilon)L^*$ and partition $\Pi = \{P_1, \ldots, P_t\}$ such that $d(P_i, P_j) > \varepsilon L^*/k$ for all $i \neq j$. Let E be the set of edges of $\text{MST}(P)$ with length more than $\varepsilon L^*/k$, and let $T_1, \ldots, T_z$ be the forest obtained from $\text{MST}(P)$ by removing E. Let $V(T_j)$ denote the sites in T_j. For any j we have $V(T_j) \subseteq P_i$ for some i. Otherwise, there would exist two sites $p, q \in V(T_j)$ that are neighbors in T_j but stay in different sets in Π. This would lead to a contradiction: the former implies $d(p,q) \leqslant \varepsilon L^*/k$ while the later implies $d(p,q) > \varepsilon L^*/k$. Thus Π is a coarsening of $\{V(T_1), \ldots, V(T_z)\}$ with respect to some subset of E.

By Lemma 8, the number of edges of $\text{MST}(P)$ longer than $\varepsilon L^*/k$ is at most $k\left(1+\frac{k}{\varepsilon}\right)$. That is, the heaviest $k\left(1+\frac{k}{\varepsilon}\right)$ edges of $\text{MST}(P)$ are a superset of the set E from above. Thus we can find the partition Π from above by first computing $\text{MST}(P)$, removing the heaviest $k\left(1+\frac{k}{\varepsilon}\right)$ edges, and then trying all coarsenings determined by subsets of the removed edges. Given a γ-approximation for TSP, below we argue how to get a γ-approximation to the optimal cyclic solution for a given partition Π. Running this subroutine for each of the above determined partitions and taking the best solution found will thus give latency at most $(1+\varepsilon)\gamma L^*$.

Observe that the optimal cyclic solution on a given partition $\Pi = \{P_1, \ldots, P_t\}$ uses cycles determined by $\mathrm{TSP}(P_i)$, and chooses k_i, the number of robots assigned to P_i, so as to minimize $\max_i tsp(P_i)/k_i$. Thus we can compute a γ-approximation of the optimal solution on Π by first computing a γ-approximation to $\mathrm{TSP}(P_i)$ for all i, where $\overline{tsp}(P_i)$ denotes its corresponding value, and then selecting $k'_1, \ldots, k'_t$ so as to minimize $\max_i \overline{tsp}(P_i)/k'_i$. The latter step of determining the k'_i can be done in $O(k \log k)$ time by initially assigning one robot to each P_i, and then iteratively assigning each next robot to whichever set of the partition currently has the largest ratio. The latency of the solution we find for Π is thus

$$\max_i \left\{ \frac{\overline{tsp}(P_i)}{k'_i} \right\} \leqslant \max_i \left\{ \frac{\overline{tsp}(P_i)}{k_i} \right\} \leqslant \max_i \left\{ \frac{\gamma \cdot tsp(P_i)}{k_i} \right\} = \gamma \cdot [\text{optimal cyclic latency for } \Pi],$$

where the last inequality follows from the fact that $\overline{tsp}(P_i) \leqslant \gamma \cdot tsp(P_i)$ for all i.

It remains to bound the running time. For each partition Π we approximate $\mathrm{TSP}(P_i)$ for all i, and then run an $O(k \log k)$ time algorithm to determine the robot assignment. Thus the time per partition is bounded by $\tau_\gamma(n)$, where n is the total number of sites. Here we assume that $\tau_\gamma(n) = \Omega(n \log n)$ and that $\tau_\gamma(n)$ upper bounds the time for the initial $\mathrm{MST}(P)$ computation.

The number of partitions we consider is determined by the number of subsets of size at most k of the longest $k(1 + k/\varepsilon)$ edges of $\mathrm{MST}(P)$, which is bounded by

$$\binom{k(1+k/\varepsilon)}{k} \cdot 2^k,$$

as the first term bounds the number of subsets of size exactly k, and for each subset the second term accounts for the number of ways in which we can pick at most k edges from that subset. We have the following standard upper bound on binomial coefficients.

$$\binom{N}{K} \leqslant \left(\frac{N \cdot e}{K} \right)^K.$$

Therefore, the total number of partitions we consider is at most

$$\binom{k(1+k/\varepsilon)}{k} \cdot 2^k \leqslant \left(\frac{k(1+k/\varepsilon) \cdot e}{k} \right)^k \cdot 2^k = (2e(1+k/\varepsilon))^k = (O(k/\varepsilon))^k.$$

Thus the total running time is $(O(k/\varepsilon))^k \cdot \tau_\gamma(n)$ as claimed. ◀

Recently, Karlin et al. [10] presented a $(3/2 - \delta)$-approximation algorithm for metric TSP, where $\delta > 10^{-36}$ is a constant, thus slightly improving the classic $(3/2)$-approximation by Christofides [5]. Furthermore TSP in $\mathbb{R}^d$ admits a PTAS [4, 15]. Thus we have the following.

▶ **Corollary 10.** *For any fixed k, there is polynomial-time $(3/2)$-approximation algorithm for finding a minimum-latency cyclic patrol schedule with k robots in arbitrary metric spaces, and there is a PTAS in $\mathbb{R}^d$ for any fixed constant d.*

Theorem 2 in Section 3 and Corollary 10 together imply the following.

▶ **Theorem 11.** *For any fixed k and $\varepsilon > 0$, there is a polynomial-time $(3(1 - 1/k) + \varepsilon)$-approximation algorithm for the k-robot patrol-scheduling problem in arbitrary metric spaces, and a polynomial-time $(2(1 - 1/k) + \varepsilon)$-approximation algorithm in $\mathbb{R}^d$ (for fixed d).*

5 Conclusion and Future Work

This is the first paper that presents rigorous analysis and approximation algorithms for multi-robot patrol scheduling problem in general metric spaces. There are several challenging open problems. The first and foremost is to prove or disprove the conjecture that there is always a cyclic solution that is optimal overall. Proving this conjecture will immediately provide a PTAS for the Euclidean multi-robot patrol-scheduling problem. It would also imply that the decision problem is decidable. Another direction for future research is to extend the results to the weighted setting. As has been shown for the 1-dimensional problem [1], the weighted setting is considerably harder.

References

1 Peyman Afshani, Mark de Berg, Kevin Buchin, Jie Gao, Maarten Löffler, Amir Nayyeri, Benjamin Raichel, Rik Sarkar, Haotian Wang, and Hao-Tsung Yang. Approximation algorithms for multi-robot patrol-scheduling with min-max latency. In *Algorithmic Foundations of Robotics XIV*, pages 107–123, 2021.

2 Soroush Alamdari, Elaheh Fata, and Stephen L Smith. Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations. *The International Journal of Robotics Research*, 33(1):138–154, 2014.

3 Esther M Arkin, Refael Hassin, and Asaf Levin. Approximations for minimum and min-max vehicle routing problems. *Journal of Algorithms*, 59(1):1–18, 2006.

4 Sanjeev Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM (JACM)*, 45(5):753–782, 1998.

5 Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie-Mellon Univ., Pittsburgh, 1976.

6 George B Dantzig and John H Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

7 Yehuda Elmaliach, Asaf Shiloni, and Gal A. Kaminka. A realistic model of frequency-based multi-robot polyline patrolling. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '08, pages 63–70, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

8 Bruce L Golden, Subramanian Raghavan, and Edward A Wasil. *The vehicle routing problem: latest advances and new challenges*, volume 43. Springer Science & Business Media, 2008.

9 L. Iocchi, L. Marchetti, and D. Nardi. Multi-robot patrolling with coordinated behaviours in realistic environments. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2796–2801, September 2011. doi:10.1109/IROS.2011.6094844.

10 Anna R. Karlin, Nathan Klein, and Shayan Oveis Gharan. A (slightly) improved approximation algorithm for metric TSP. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 32–45, 2021.

11 M Yu Khachai and ED Neznakhina. A polynomial-time approximation scheme for the Euclidean problem on a cycle cover of a graph. *Proceedings of the Steklov Institute of Mathematics*, 289(1):111–125, 2015.

12 Michael Khachay and Katherine Neznakhina. Polynomial time approximation scheme for the minimum-weight k-size cycle cover problem in Euclidean space of an arbitrary fixed dimension. *IFAC-PapersOnLine*, 49(12):6–10, 2016.

13 M Reza Khani and Mohammad R Salavatipour. Improved approximation algorithms for the min-max tree cover and bounded tree cover problems. *Algorithmica*, 69(2):443–460, 2014.

14 Kin Sum Liu, Tyler Mayer, Hao-Tsung Yang, Esther Arkin, Jie Gao, Mayank Goswami, Matthew P. Johnson, Nirman Kumar, and Shan Lin. Joint sensing duty cycle scheduling for heterogeneous coverage guarantee. In *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*, pages 1–9. IEEE, 2017.

15 Joseph SB Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM Journal on computing*, 28(4):1298–1309, 1999.

16 Christos H Papadimitriou. The Euclidean travelling salesman problem is NP-complete. *Theoretical computer science*, 4(3):237–244, 1977.

17 D. Portugal and R. P. Rocha. On the performance and scalability of multi-robot patrolling algorithms. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 50–55, November 2011. doi:10.1109/SSRR.2011.6106761.

18 E. Stump and N. Michael. Multi-robot persistent surveillance planning as a vehicle routing problem. In *Automation Science and Engineering (CASE), 2011 IEEE Conference on*, pages 569–575, August 2011. doi:10.1109/CASE.2011.6042503.

19 Paolo Toth and Daniele Vigo. *The vehicle routing problem*. SIAM, 2002.

20 Wenzheng Xu, Weifa Liang, and Xiaola Lin. Approximation algorithms for min-max cycle cover problems. *IEEE Transactions on Computers*, 64(3):600–613, 2013.

21 Hao-Tsung Yang, Shih-Yu Tsai, Kin Sum Liu, Shan Lin, and Jie Gao. Patrol scheduling against adversaries with varying attack durations. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1179–1188, 2019.