Beachcombing on strips and islands[⋆]

Evangelos Bampas[1], Jurek Czyzowicz[2], David Ilcinkas[3], and Ralf Klasing[3]

[1] LIS, Aix-Marseille University and CNRS, Marseille, France
`evangelos.bampas@gmail.com`
[2] Département d'informatique, Université du Québec en Outaouais, Canada
`Jurek.Czyzowicz@uqo.ca`
[3] LaBRI, CNRS & Univ. Bordeaux, France
`{david.ilcinkas,ralf.klasing}@labri.fr`

Abstract. A group of mobile robots (beachcombers) have to search collectively every point of a given domain. At any given moment, each robot can be in *walking mode* or in *searching mode*. It is assumed that each robot's maximum allowed searching speed is strictly smaller than its maximum allowed walking speed. A point of the domain is searched if at least one of the robots visits it in searching mode. The Beachcombers' Problem consists in developing efficient *schedules* (algorithms) for the robots which collectively search all the points of the given domain as fast as possible. We consider searching schedules in the following one-dimensional geometric domains: the cycle of a known circumference L, the finite straight line segment of a known length L, and the semi-infinite line $[0, +\infty)$.

We first consider the *online* Beachcombers' Problem (i.e. the scenario when the robots do not know in advance the length of the segment to be searched), where the robots are initially collocated at the origin of a semi-infinite line. It is sought to design a schedule A with maximum *speed* S, defined as $S = \inf_\ell \frac{\ell}{t_A(\ell)}$, where $t_A(\ell)$ denotes the time when the search of the segment $[0, \ell]$ is completed under A. We consider a *discrete* and a *continuous* version of the problem, depending on whether the infimum is taken over $\ell \in \mathbb{N}^*$ or $\ell \geq 1$. We prove that the `LeapFrog` algorithm, which was proposed in [Czyzowicz et al., SIROCCO 2014, LNCS 8576, pp. 23–36 (2014)], is in fact optimal in the discrete case. This settles in the affirmative a conjecture from that paper. We also show how to extend this result to the more general continuous online setting.

For the *offline* version of the Beachcombers' Problem (i.e. the scenario when the robots know in advance the length of the segment to be searched), we consider the *t-source* Beachcombers' Problem (i.e. all robots start from a fixed number $t \geq 1$ of starting positions) on the cycle and on the finite segment. For the *t-source* Beachcombers' Problem on the cycle, we show that the structure of the optimal solutions is

⋆ This work appears in preliminary form in the Proceedings of the 11th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, LNCS vol. 9536, pp. 155–168, Springer (2015).

BY-NC-ND © 2019. This accepted manuscript is licensed under the CC BY-NC-ND 4.0 license: https://creativecommons.org/licenses/by-nc-nd/4.0/. The final publication is available on ScienceDirect: https://doi.org/10.1016/j.tcs.2019.04.001.

identical to the structure of the optimal solutions to the $2t$-source Beachcombers' Problem on a finite segment. In consequence, by using results from [Czyzowicz et al., ALGOSENSORS 2014, LNCS 8847, pp. 3–21 (2014)], we prove that the *1-source* Beachcombers' Problem on the cycle is NP-hard, and we derive approximation algorithms for the problem. For the *t-source* variant of the Beachcombers' Problem on the cycle and on the finite segment, we also derive efficient approximation algorithms.

One important contribution of our work is that, in all variants of the offline Beachcombers' Problem that we discuss, we allow the robots to *change direction of movement* and search points of the domain on both sides of their respective starting positions. This represents a significant generalization compared to the model considered in [Czyzowicz et al., ALGOSENSORS 2014, LNCS 8847, pp. 3–21 (2014)], in which each robot had a fixed direction of movement that was specified as part of the solution to the problem. We manage to prove that changes of direction do not help the robots achieve optimality.

1 Introduction

A group of n mobile robots have to explore collectively a given one-dimensional domain. The robots may be initially collocated or dispersed in the domain. At every moment of time, a robot can be either in *walking mode* or in *searching mode*. A robot in walking mode traverses the domain with a speed not exceeding its maximal *walking speed*. A robot in searching mode can travel using at most its maximal *searching speed*, which is strictly smaller than its walking speed, reflecting the fact that a searching activity is more time-consuming. Different robots may have distinct maximal walking and searching speeds. A robot can change mode, speed, and direction of movement instantaneously. There is no communication between the robots during the execution of the algorithm. In the Beachcombers' Problem, the goal is to design a schedule for the movement of all robots so that the domain is *searched* as fast as possible. A domain is said to be searched under a given schedule, if every point of the domain is visited by at least one robot in searching mode.

As pointed out in [12], where the Beachcombers' Problem was introduced, there are numerous examples in quite diverse domains in which exploration using *two-speed* robots arises as a natural model for the underlying processes. For example, *foraging* or *harvesting* a field may take longer than inadvertent walking. In computer science, *web page indexing* or *code inspection* require a more involved investigation. A common feature of these examples is that the activity of searching, or other action to be performed on the territory, takes more time than casual territory traversal. The analogy to *beachcombers* has been introduced in [12] to bring out that, e.g., a beachcomber looking for things of value performs a meticulous search of the beach, which takes significantly more time than simply walking from one point of the beach to another. Further motivation for the two-speed model can be found in [12, 13].

Preliminaries and notation. We consider searching schedules using two-speed robots in the following one-dimensional geometric domains: the cycle of a known circumference L, the finite straight line segment of a known length L, and the semi-infinite line $[0, +\infty)$.

A *schedule* $\mathcal{S}$ for the robots is defined by a strictly increasing sequence of times $t_0, t_1, \ldots$, as well as, for every robot i and every interval $[t_j, t_{j+1}]$, for $j \geq 0$, a *mode* (walking or searching), a *speed* (respecting the maximum speed of the chosen mode for robot i), and a *direction* of movement. For any fixed robot r, we refer to the individual schedule of robot r under $\mathcal{S}$ as the *trajectory* of robot r. A schedule is *finite* if the sequence of times $t_0, t_1, \ldots$ has a last element t_f. A schedule is *correct* if, for every point p of the domain, there exists a time moment at which p is visited by a robot in searching mode. For a finite schedule, its *completion time* (or *finishing time*) is t_f.

The efficiency of the search in the first two domains (i.e., the cycle of a known circumference L and the finite straight line segment of a known length L) is measured in terms of the completion time t_f (the smaller the better) or, equivalently, in terms of the *speed* L/t_f of the schedule (the larger the better). We are only interested in finite schedules for those domains. In the third domain (i.e., the semi-infinite line $[0, +\infty)$), the efficiency is better expressed by the *speed* of the search, represented by $\inf_\ell \frac{\ell}{t_{\mathcal{S}}(\ell)}$, where $t_{\mathcal{S}}(\ell)$ denotes the time when the search of the segment $[0, \ell]$ is completed under schedule $\mathcal{S}$. In the *discrete* version of the problem, the infimum $\inf_\ell \frac{\ell}{t_{\mathcal{S}}(\ell)}$ is taken over $\ell \in \mathbb{N}^*$. On the other hand, in the *continuous* setting, the infimum $\inf_\ell \frac{\ell}{t_{\mathcal{S}}(\ell)}$ is taken over $\ell \geq 1$. Alternatively, the Beachcombers' Problem on the semi-infinite line can be seen as the Beachcombers' Problem on the finite straight line segment of *unknown* length ℓ. Then, the term $\inf_\ell \frac{\ell}{t_{\mathcal{S}}(\ell)}$ can be seen as the worst-case speed of the search over the different possible segment lengths ℓ.

In the sequel, we will use the term *offline setting* for instances of the problem in which the size of the domain is known (segment of known length and cycle of known circumference), and the term *online setting* for instances of the problem in which the size of the domain is unknown (semi-infinite line or, equivalently, segment of unknown length).

In all cases, a correct schedule $\mathcal{S}$ is *optimal* if there does not exist any other correct schedule with speed larger than that of $\mathcal{S}$. Equivalently, in the offline setting, a correct schedule is optimal if it has minimum completion time.

Previous work. The Beachcombers' Problem was introduced and studied in [12]. An optimal (offline) algorithm was presented for the problem in which all robots are initially located on one endpoint of a finite segment of known length. Furthermore, a 2-competitive (online) algorithm was presented for the case where all robots are initially collocated on the origin of a semi-infinite line. In [13], the Beachcombers' Problem was studied for the case of more than one starting positions on a finite segment of known length. For a fixed number $t \geq 2$ of starting positions, the t-source Beachcombers' Problem on a finite segment asks to find t starting points on the segment, an assignment of the robots to the starting points, and a search schedule which concludes the search of the finite segment as quickly as possible. It was shown in [13] that this problem is NP-hard for $t = 2$, even when all robots have the same walking speed, that the optimal solution can be computed efficiently when all robots have the same searching speed, and that there exist a deterministic approximation algorithm for $t = 2$ and a randomized approximation algorithm for general t. In [11], the authors study the single-source problem on a finite segment, when the starting point is given as part of the input and can be anywhere on the segment.

Our contributions. In Section 2, we study the online Beachcombers' Problem on the semi-infinite line $[0, +\infty)$. We prove that the 2-competitive `LeapFrog` algorithm, which was proposed in [12], is in fact optimal in the discrete case. This settles in the affirmative a conjecture from [12]. We also show how to extend this result to the more general continuous online setting.

As regards the offline Beachcombers' Problem, we consider in Section 3 the *t-source Beachcombers' Problem on the cycle with zigzags* (t-$\mathsf{SBPC_z}$), where the possibility for zigzags means that robots may change direction of movement during their trajectories. We show that the structure of the optimal solutions to t-$\mathsf{SBPC_z}$ is identical to the structure of the optimal solutions to the $2t$-source Beachcombers' Problem on a finite segment, as defined in [13]. This implies that the results from [13] are carried over to the case of the cycle with zigzags, yielding an NP-hardness result for 1-$\mathsf{SBPC_z}$ as well as the existence of efficient exact and approximation algorithms for certain special cases of the problem. In particular, the NP-hardness of 1-$\mathsf{SBPC_z}$ seems at first somewhat surprising, considering that in [12] the authors presented an efficient algorithm generating optimal schedules for the single-source problem on a finite segment without zigzags. Our results for the cycle topology provide a partial answer to an open question posed in [12] and [13], concerning the study of the problem in different domain topologies.

Furthermore, in Section 4, we provide additional arguments to show that our results for t-$\mathsf{SBPC_z}$ carry over to the *t-source Beachcombers' Problem on the line with zigzags* (t-$\mathsf{SBPL_z}$).

An important contribution of our work is that, in both variants of the offline Beachcombers' Problem that we discuss, we allow the robots to *zigzag*, i.e., to change direction of movement and search points of the domain on both sides of their respective starting positions. This represents a significant generalization compared to the model considered in [13], in which each robot had a fixed direction of movement that was specified as part of the solution to the problem. On an intuitive level, allowing the robots to zigzag should not result in a faster schedule. However, no proof of this intuition had been found until now. We manage to *prove* that changes of direction do not help the robots achieve optimality.

Related work. Searching and exploration have been studied in numerous papers considering graphs or geometric environments (e.g. [1, 4, 5, 7, 8, 15, 17–19, 22]). The performance of the searching or exploration is typically expressed by the trajectory length or the time used by the mobile robot.

Many searching and exploration algorithms are studied in the *online* setting, i.e., the target position or sometimes other parameters of the environment are *a priori* unknown (cf. [2, 3, 9, 15, 17, 20, 21]). Efficiency of such algorithms is typically measured by the *competitive ratio*, i.e., the ratio of the time spent by the online algorithm with respect to the time of the optimal offline algorithm.

Most of the papers studying searching and exploration concern single robots. Sets of collaborating mobile robots were studied, e.g., in [10, 16, 23, 24]. Tradeoffs between the number of robots and the time of exploration were derived in [20].

The majority of the research on mobile robots concerns robots having the same mobile speed. Robots with distinct speeds were considered in the context of sensor energy efficiency [26], for designing fast converging population protocols [6], and for patrolling the boundary of an environment [14, 25].

2 The online beachcombers' problem on the semi-infinite line

In this section, we consider two variants of the online beachcombers' problem on the semi-infinite line (given in Definitions 1 and 2). Definition 1 corresponds to the online problem presented in [12]. Definition 2 is new, and generalizes Definition 1.

Definition 1 (Discrete Online Beachcombers' problem). *Given n robots with walking speeds w_i and searching speeds $s_i < w_i$, for $1 \le i \le n$, initially collocated at the origin of a semi-infinite line $[0, +\infty)$, the problem consists in finding an optimal correct searching schedule for this semi-infinite line. The* discrete online speed *of a schedule A is defined as*

$$\inf_{\ell \in \mathbb{N}^*} \frac{\ell}{t_A(\ell)}$$

where $t_A(\ell)$ denotes the time when the search of the segment $[0, \ell]$ is completed.[1]

Definition 2 (Continuous Online Beachcombers' problem). *Given n robots with walking speeds w_i and searching speeds $s_i < w_i$, for $1 \le i \le n$, initially collocated at the origin of a semi-infinite line $[0, +\infty)$, the problem consists in finding an optimal correct searching schedule for this semi-infinite line. The* continuous online speed *of a schedule A is defined as*

$$\inf_{\ell \ge 1} \frac{\ell}{t_A(\ell)}$$

where $t_A(\ell)$ denotes the time when the search of the segment $[0, \ell]$ is completed.

[1] Note that all points in the segment $[0, \ell]$ have to be searched, not only certain discrete points.

Note that the infimum is taken over $\ell \geq 1$ and not over $\ell > 0$. Indeed, our definition of a schedule excludes the possibility to perform infinitesimally small steps at the origin of the semi-infinite line. Therefore, given any schedule, it is always possible to find a sufficiently small ℓ such that the whole segment $[0, \ell]$ is searched by a single robot. As a consequence, if the infimum was taken over $\ell > 0$, then the continuous online speed of any schedule would be at most the maximum searching speed of any of the robots. This degeneration of the problem when considering arbitrarily small segments is very similar to the case of the linear search problem (a.k.a. the cow path problem). In this latter problem, the online competitive ratio of any search strategy is unbounded if the strategy cannot perform infinitesimally small moves and if the target can lie arbitrarily close to the origin.

The main purpose of this section is to prove the optimality of the 2-competitive online algorithm `LeapFrog` described in [12]. Our first step toward this goal is to restrict ourselves to particular schedules, which are much simpler to analyze but are nevertheless at least as efficient (in terms of online speeds) as general ones. The following simple lemma, of a flavor similar to Proposition 1 in Section 3 and some results in [12], holds both for the discrete and the continuous cases.

Lemma 1. *For every correct schedule $\mathcal{S}$, there exists a correct schedule $\mathcal{S}'$ whose discrete and continuous online speeds are not smaller than the respective ones of $\mathcal{S}$, and such that:*

1. *The interiors of the segments searched by the different robots do not overlap (i.e., the searched segments may only intersect at their endpoints).*
2. *Every moving robot always moves in the initial direction at the full speed permitted by its current mode.*

Proof. First observe that any searching schedule may be converted to another one, which has the property that all sub-segments which were being searched (during some time intervals $[t_j, t_{j+1}]$ by some robots) have pairwise disjoint interiors. Indeed, if some sub-segment is being searched by two different robots (or twice by the same robot), the second searching may be replaced by the walk through it by the robot involved. Since the walking speed of any robot is always larger than its searching speed, the speed of such converted schedule is not smaller than the original one.

We can thus assume that the schedule $\mathcal{S}$ satisfies Condition 1.

Now in this schedule, each robot i searches a sequence of segments $([a^i_k, b^i_k])_{k\in\mathbb{N}}$ with $a^i_k < b^i_k$ and $b^i_k < a^i_{k+1}$, but the robot may not perform the search in that particular order. We now define a schedule $\mathcal{S}'$ in which each robot i always moves in the initial direction from the origin. Whenever i is in a segment $[0, a^i_0)$ or (b^i_k, a^i_{k+1}), it walks at full speed, whereas whenever i is in a segment $[a^i_k, b^i_k]$, it searches at full speed. If the sequence of segments $([a^i_k, b^i_k])$ is in fact finite, the robot i stops after searching the last segment.

We claim that this new schedule satisfies both conditions. Indeed, since the regions searched by each robot are the same in $\mathcal{S}$ and $\mathcal{S}'$, $\mathcal{S}'$ is a correct schedule, and since $\mathcal{S}$ satisfies Condition 1, then so does $\mathcal{S}'$. By construction, $\mathcal{S}'$ also satisfies Condition 2. We thus just have to check that the discrete and continuous online speeds of $\mathcal{S}'$ are not smaller than the respective ones of $\mathcal{S}$.

Suppose that this is not the case. Then there exists a segment $[0, \ell]$, with $\ell \geq 1$, that $\mathcal{S}$ searches faster that $\mathcal{S}'$. This further implies that one robot i searches $[0, \ell] \cap (\cup_{k \in \mathbb{N}}[a_k^i, b_k^i]) = (\cup_{k=0}^{p-1}[a_k^i, b_k^i]) \cup [a_p^i, b_p'^i]$, with $a_p^i < b_p'^i \leq b_p^i$, faster in $\mathcal{S}$ than in $\mathcal{S}'$. But to search a segment of length d, i needs time at least d/s_i, and to go from b_k^i to a_{k+1}^i, i needs time at least $(a_{k+1}^i - b_k^i)/w_i$. Thus $\mathcal{S}$ needs at least $(a_0^i + \sum_{k=0}^{p-1}(a_{k+1}^i - b_k^i))/w_i + (\sum_{k=0}^{p-1}(b_k^i - a_k^i) + b_p'^i - a_p^i)/s_i$, which is precisely the time required by i to search the same region in $\mathcal{S}'$, a contradiction. The schedule $\mathcal{S}'$ thus satisfies the properties stated in the Lemma. □

The idea of the `LeapFrog` algorithm is to make all sufficiently fast robots, forming the so-called swarm of the algorithm, meet at some regular intervals, every unit of distance in [12]. For this purpose, each robot of the swarm is assigned a specific fraction of such unit segment that it has to search. The robot walks the rest of the segment.

More precisely, if the k robots $r_1, \ldots, r_k$ are ordered such that $w_1 \geq \cdots \geq w_k$, the `LeapFrog` algorithm is defined as follows. First, let us define the speed S_j, for $1 \leq j \leq k$, as $S_j = \frac{\sum_{i=1}^{j} \frac{1}{\delta_i}}{1 + \sum_{i=1}^{j} \frac{1}{w_i \delta_i}}$, with $\delta_i = \frac{1}{s_i} - \frac{1}{w_i}$. The discrete online speed of Algorithm `LeapFrog`, denoted LF, is defined as S_{j^*}, where j^* is either the smallest j such that $S_j \geq w_{j+1}$, or $j^* = k$ if there is no such j. Then, all robots r_i with $i > j^*$ stop at the origin forever. The other robots constitute the swarm of the algorithm and each such robot r_i searches at speed s_i a proportion $c_i = \frac{1}{\mathrm{LF} \cdot \delta_i} - \frac{1}{w_i \delta_i}$ of each unit segment (and walks at speed w_i the rest of the time).

The key properties of the algorithm that we will use are, first, that the robots constituting the swarm are exactly those robots having a walking speed larger than LF, second, that the algorithm satisfies the regularity properties from Lemma 1, and third, that all robots of the swarm meet at the same time at the end of each unit segment.

Before proving that Algorithm `LeapFrog` is optimal in terms of *discrete* online speed, we prove the slightly weaker result that no correct schedule can have a *continuous* online speed larger than LF.

Lemma 2. *The* continuous *online speed of any correct schedule is at most* LF.

Proof. For a contradiction, let us assume that there exists a correct schedule $\mathcal{S}$ whose continuous online speed is larger than LF, say with speed s. Without loss of generality, we assume that this schedule satisfies the regularity properties from Lemma 1. Let R_i (resp. R_f), be the set of robots which search (resp. do not search) infinitely often in $\mathcal{S}$, i.e., arbitrarily far from the origin. If the set R_f is non-empty, then let d be the maximum distance from the origin of a point searched by any such robot. Otherwise, let $d = 0$. Now, let d' be some distance larger than 1 such that every robot from the set R_i has searched a segment of positive length between distance 1 and distance d' from the origin.

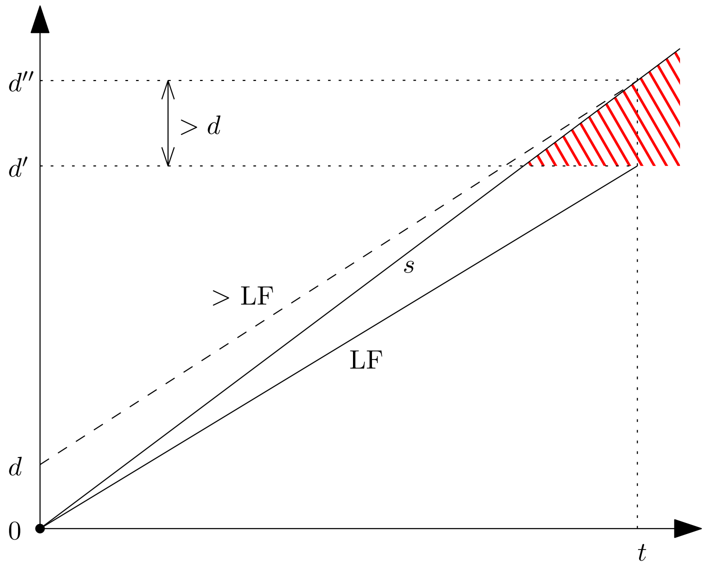

Fig. 1. Time-space diagram representing the key elements of the proof of Lemma 2. Beyond distance d from the origin, only robots from R_i, and thus from the `LeapFrog`'s swarm, are searching. Moreover, beyond distance d', the robots from R_i cannot be in the hatched region of the diagram. Therefore, in time t, these robots have to traverse and collectively search the segment $[d, d'']$, which is longer than what the `LeapFrog`'s swarm is able to traverse and collectively search in the same time t, leading to a contradiction.

Finally, consider a sufficiently large distance $d'' > \max(d', d \cdot \frac{s}{s-\mathrm{LF}})$, such that the distance $\mathrm{LF} \cdot (d''/s)$ is an integer, and let $t = d''/s$. Note that $d'' - d > \mathrm{LF} \cdot t$ by definition of d''.

In the remainder of the proof (see Figure 1), we will prove that every robot from the set R_i belongs to the swarm of robots active in Algorithm `LeapFrog`. We will also prove that the robots belonging to R_i will, in time t, traverse and collectively search in $\mathcal{S}$ the segment $[d, d'']$, which is longer than what the `LeapFrog`'s swarm is able to traverse and collectively search in the same time t, leading to a contradiction.

Fix any robot j belonging to the set R_i, with walking speed w_j and searching speed $s_j < w_j$. Since robot j searches infinitely often in $\mathcal{S}$, this robot must have a walking speed larger than s (by regularity of $\mathcal{S}$), and thus larger than LF, and therefore it must also belong to the `LeapFrog`'s swarm (by the first key property of Algorithm `LeapFrog`). By the second and third key properties of `LeapFrog`, robot j in Algorithm `LeapFrog` is exactly at distance $\mathrm{LF} \cdot t$ from the origin at time t. On the other hand, consider the last segment $[d_1, d_2]$ searched by robot j in $\mathcal{S}$ before reaching distance d'' from the origin (one may have $d_2 = d''$). By definition of d' and d'', we have $d_2 > 1$. By definition of s, robot j is at distance d_2 from the origin at time at most d_2/s. Since $w_j > s$, robot j is at time $t = d''/s$ at a distance larger than or equal to d'' from the origin, and thus larger than $\mathrm{LF} \cdot t$. Differently speaking, at time t, robot j is further from the origin in $\mathcal{S}$ than in `LeapFrog`. By regularity of both schedules, it follows that, in the time interval $[0, t]$, robot j has searched less (i.e., has searched during a smaller proportion of the time interval) in $\mathcal{S}$ than in Algorithm `LeapFrog`.

We can now summarize the situation as follows. The robots of the set R_i are also in the `LeapFrog`'s swarm and, in the time interval $[0, t]$, they collectively search a smaller distance in $\mathcal{S}$ than in Algorithm `LeapFrog`. This contradicts the fact that, until time t, they searched in $\mathcal{S}$ the whole segment $[d, d'']$, whose length is larger than $\mathrm{LF} \cdot t$, without the help of the robots from R_f. $\square$

We are now ready to prove the main result of this section.

Theorem 1. *The* discrete *online speed of any correct schedule is at most* LF.

Proof. For a contradiction, suppose that there exists a correct schedule $\mathcal{S}$ whose discrete online speed is $s = \mathrm{LF} + 2\epsilon$, for some $\epsilon > 0$. For any positive integer N, we define the average online speed of schedule $\mathcal{S}$ in the segment $[N, N+1]$ as $s_N = \inf_{\ell \in [N,N+1]} \frac{\ell}{t_A(\ell)}$.

Note that, for any positive integer N, $s_N \geq \frac{N}{N+1} \cdot s$. Indeed, for any $\ell \in [N, N+1]$, we have $\frac{\ell}{t_A(\ell)} \geq \frac{N}{t_A(N+1)} \geq \frac{N}{N+1} \cdot s$, where the first inequality follows from the fact that $t_A(\cdot)$ is an increasing function and the second inequality follows from the definition of s. Therefore, for a sufficiently large integer N_0, we have that $s_N \geq s - \epsilon$ for all integers $N \geq N_0$.

We are interested in the continuous online speed of $\mathcal{S}$ for distances larger than N_0, which is given by $\inf_{\ell \geq N_0} \frac{\ell}{t_{\mathcal{S}}(\ell)}$. Since $[N_0, +\infty) = \cup_{N \in \mathbb{N}: N \geq N_0} [N, N+1]$, we conclude that

$$\inf_{\ell \geq N_0} \frac{\ell}{t_{\mathcal{S}}(\ell)} = \inf_{N \in \mathbb{N}: N \geq N_0} s_N \geq s - \epsilon > \mathrm{LF} \;.$$

Therefore, the continuous online speed of $\mathcal{S}$ is strictly larger than LF for distances larger than N_0. Consequently, there exists a correct schedule $\mathcal{S}'$ with continuous online speed strictly larger than LF, which is obtained from $\mathcal{S}$ by scaling the unit of distance and the unit of time by a factor of N_0. This contradicts Lemma 2. $\square$

Concerning the continuous online speed metrics, it is possible to obtain a slightly more precise result than the one of Lemma 2.

Lemma 3. *If there are at least two robots and Algorithm* `LeapFrog` *uses all the robots, then the continuous online speed of any correct schedule is strictly smaller than* LF.

Proof. For a contradiction, let us assume that there exists a correct schedule $\mathcal{S}$ whose continuous online speed is at least LF, and thus, by Lemma 2, exactly LF. Without loss of generality, we assume that this schedule satisfies the regularity properties from Lemma 1. Similarly as in the proof of Lemma 2, let d be a distance larger than 1 such that every robot has searched a segment of positive length between distance 1 and distance d from the origin. For the same reasons as in Lemma 2, at time d/LF, every robot is at distance at least d from the origin, and the whole segment $[0, d]$ has been searched. The former property implies that every robot searched a proportion of the segment not larger than in the case of the original `LeapFrog` algorithm. Combined with the latter property, this implies that every robot is in fact exactly at distance d from the origin at time d/LF. Let $[d, d']$ be the next segment which is continuously searched by one of the robots after that time d/LF. Since there are more than one robots in the swarm, the searching speed of the robot searching $[d, d']$ must be smaller than LF. This implies that $d'/t_{\mathcal{S}}(d')$ must be strictly smaller than LF for this distance $d' \geq 1$, a contradiction. □

It turns out that simple variations of Algorithm `LeapFrog` can match the bounds given in Lemmas 2 and 3. More precisely, we have the following positive result.

Lemma 4. *For any $\epsilon > 0$, there exists an algorithm whose continuous online speed is at least* LF $- \epsilon$. *Moreover, in the case when there is only one robot or Algorithm* `LeapFrog` *does not use all of the robots, there exists an (optimal) algorithm whose continuous online speed is exactly* LF.

Proof. In Algorithm `LeapFrog`, all robots participating in the swarm are synchronized at every integer distance from the origin, that is, they all arrive at the same time at the end of each unit segment. For any positive integer N, we denote by $\texttt{LeapFrog}_N$ the variant of `LeapFrog` in which the robots participating in the swarm synchronize every $1/N$ units of distance, instead of every unit as in the original Algorithm `LeapFrog`. It is easy to check that the continuous online speed of Algorithm $\texttt{LeapFrog}_N$ tends to LF as N tends toward infinity. The family of algorithms $\{\texttt{LeapFrog}_N\}_{N\in\mathbb{N}^*}$ is thus optimal in the case when there are at least two robots and Algorithm `LeapFrog` uses all the robots (cf. Lemma 3).

If there is only one robot, then the only reasonable algorithm is the one in which the single robot always searches at its maximal speed. This algorithm is in fact Algorithm `LeapFrog`, and its continuous online speed is equal, in this special case, to its discrete online speed LF. Lemma 2 thus shows that Algorithm `LeapFrog` is optimal also in this case.

The remaining case is when there are at least two robots, but the swarm of Algorithm `LeapFrog` does not use all the robots. In this particular case, we consider the following adaptation `LeapFrog′` of Algorithm `LeapFrog`. Let r, with searching speed s, be some robot not participating in the swarm in Algorithm `LeapFrog`. In our adaptation `LeapFrog′`, this robot r searches the semi-infinite line from its beginning at its maximum searching speed s during $1/\mathrm{LF}$ time units before stopping forever. Let p be the point at which r stops. (Note that p is at distance $s/\mathrm{LF} \leq 1$ from the origin.) All the robots of the swarm walk at the walking speed of the slowest walker among them until reaching point p. Since this walking speed is strictly larger than LF, by definition of the swarm, the robots reach point p at time $p/\mathrm{LF} - \epsilon/\mathrm{LF}$, for some positive ϵ. At this point, all swarm robots execute Algorithm $\texttt{LeapFrog}_N$ as if p was the origin of the semi-infinite line, with $N = \lceil 1/\epsilon \rceil$. The integer N is chosen sufficiently large so that, at any time at least $1/\text{LF}$, the swarm has always searched one segment of length $1/N$ ahead of the normal Algorithm $\texttt{LeapFrog}_N$. If $\mathcal{S}$ is the schedule obtained by $\texttt{LeapFrog}'$, this implies that $\frac{\ell}{t_{\mathcal{S}}(\ell)} \geq \text{LF}$ for any $\ell \geq 1$, and thus that the continuous online speed of Algorithm $\texttt{LeapFrog}'$ is equal to LF, which is optimal by Lemma 2. □

3 Multi-source beachcombers on the cycle

In this section we show that the structure of the optimal solutions to the offline multi-source Beachcombers' Problem on the cycle with zigzags is identical to the structure of the optimal solutions to the multi-source Beachcombers' Problem on a finite segment, as defined in [13]. This implies that the results from [13] are carried over to the case of the cycle, even if the robots are allowed to *zigzag* (i.e., to change direction of movement and search points of the domain on both sides of their respective starting positions). The (offline) t-source Beachcombers' Problem on the cycle with zigzags is defined as follows.

Definition 3 (t-$\mathsf{SBPC_z}$ – t-Source Beachcombers' Problem on the Cycle with zigzags). *Consider a cycle C_L of circumference $L > 0$*[2] *and n robots $r_1, \ldots, r_n$, each robot r_i having searching speed s_i and walking speed $w_i > s_i$. For an integer $t \geq 1$, t-$\mathsf{SBPC_z}$ consists in (a) dividing the robots into at most t groups, (b) choosing a particular starting point on the cycle for each group (the source of that group), and (c) finding an optimal correct finite searching schedule for C_L.*

We will refer to the source point of a group of robots in a particular solution as the *origin* or the *origin point* of the robots in that group. Also, note that it is not immediately obvious that every instance of t-$\mathsf{SBPC_z}$ has an optimal solution, because the number of feasible solutions is infinite. We will prove that an optimal solution always exists in Subsection 3.2.

The (offline) t-source Beachcombers' Problem on the segment was defined in [13] as follows.

Definition 4 (t-SBP – t-Source Beachcombers' Problem [13]). *Consider a line segment $I_L = [0, L]$ and n robots $r_1, \ldots, r_n$, each robot r_i having searching speed s_i and walking speed $w_i > s_i$. For an integer $t \geq 1$, the t-Source Beachcombers' Problem consists in (a) dividing the robots into at most t groups, (b) choosing, for each group, a particular starting point on the segment (the source of that group) and a fixed direction of movement, and (c) finding an optimal correct finite searching schedule for I_L respecting these directions of movement.*

[2] Note that one could normalize the parameter $L = 1$. However, we prefer to keep the parameter $L > 0$ general for the sake of consistency with the definition of the t-Source Beachcombers' Problem in [13].

Note that the model of [13] precludes by definition any change of direction of movement for the robots, and in fact each group of robots has a fixed direction of movement which is specified as part of the solution to the t-SBP problem. On the other hand, in our model for t-$\mathsf{SBPC_z}$, there is no fixed direction associated with each robot group (i.e., robots from the same group may go in different directions) and, in fact, we allow solutions in which robots can actually change direction of movement.

In this section, we prove that optimal solutions for t-$\mathsf{SBPC_z}$ do not involve robots changing their direction of movement (cf. Lemma 11 in Subsection 3.3). This implies that, in an optimal solution for t-$\mathsf{SBPC_z}$, robots from the same source are divided into robots that move only clockwise and robots that move only counterclockwise. Intuitively, *one* robot source in an optimal t-$\mathsf{SBPC_z}$ solution serves as *two* robot sources that obey the t-SBP restriction of each group of robots having a fixed direction of movement specified as part of the solution. In Subsection 3.4, we formalize this tight connection between the optimal solutions of t-$\mathsf{SBPC_z}$ and $2t$-SBP in Lemmas 12 and 13, and we use it to show that the results from [13] are carried over to the t-$\mathsf{SBPC_z}$ problem.

In the rest of this section, we will mainly refer to schedules for t-$\mathsf{SBPC_z}$, unless it is explicitly stated otherwise. In Subsection 3.1, we introduce the notion of *normal schedule* (cf. Definition 6), and show how to transform any correct schedule into a normal schedule with at most the same completion time. In Subsection 3.2, we show that every t-$\mathsf{SBPC_z}$ instance admits an optimal and normal schedule. Then, in Subsection 3.3, we show a specific structure of optimal normal schedules, which we use in Subsection 3.4 in order to derive hardness and approximability results for t-$\mathsf{SBPC_z}$ from the corresponding results for t-SBP. Finally, in Subsection 3.5, we show a stronger structural property of optimal t-$\mathsf{SBPC_z}$ schedules.

3.1 Normalization of t-$\mathsf{SBPC_z}$ schedules

Proposition 1. *For every correct schedule $\mathcal{S}$, there exists a correct schedule $\mathcal{S}'$, with the same robot partition, whose completion time is not greater than that of $\mathcal{S}$ and which additionally satisfies the following properties:*

1. *Every pair of arcs searched by the robots under $\mathcal{S}'$ have disjoint interiors.*
2. *During every time interval of $\mathcal{S}'$, every robot i is either stopped (i.e., "walking" or "searching" at a speed of 0) or it moves at the maximum speed w_i or s_i, according to its chosen mode during that interval.*

Proof. 1. If two robots search overlapping arcs under $\mathcal{S}$, we alter the trajectory of one of the robots so that it walks over the part that was searched by both robots. Since the maximum walking speed of a robot is always greater than its maximum searching speed, this results in a correct schedule with completion time at most equal to that of $\mathcal{S}$.
2. If a robot moves, either walking or searching, at a speed that is lower than the maximum allowed speed for its chosen mode, we alter its trajectory so

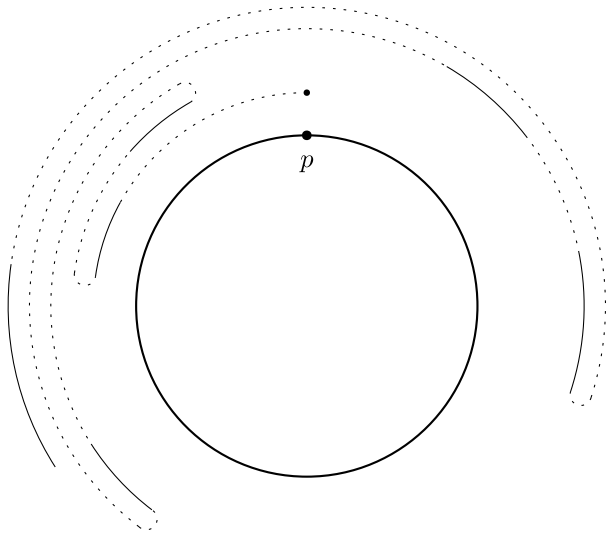

Fig. 2. An example of the trajectory of a single robot, starting from its origin point p. Dotted lines correspond to walking at maximum walking speed and solid lines correspond to searching at maximum searching speed. Changes of direction, speed, and mode are instantaneous.

that it concludes the corresponding arc at the maximum allowed speed. This modification can only reduce the completion time of the new schedule. □

Observation 1. If $\mathcal{S}$ does not satisfy the properties of Proposition 1, then, in the new schedule $\mathcal{S}'$ that we obtain by applying Proposition 1, the completion time of at least one robot is strictly smaller than the completion time of the same robot under $\mathcal{S}$. Indeed, in both kinds of modifications that we have in the proof of Proposition 1, a part of the trajectory of some robot is performed at strictly higher speed, resulting in strictly smaller completion time.

We will use this observation in the proof of Lemma 14 in Subsection 3.5.

In view of Proposition 1, we will assume in the following that the trajectory of each robot i is characterized by a sequence of arcs and, for each arc, a mode (searching or walking) and a direction (clockwise or counterclockwise), such that in each arc the robot is moving at the maximum allowed speed (Figure 2).

Note that an arc may correspond to one or more consecutive time intervals of the schedule.

Before we can state and prove the next lemma, we need to introduce the notion of a *journey* of a robot and some related terminology.

Definition 5. *Given a schedule, a* clockwise *(resp.* counterclockwise*)* journey *of a robot r with origin p is a partial trajectory of r, either between two successive visits to p or after its last visit to p and until the time it definitively stops, that starts with r leaving p in the clockwise (resp. counterclockwise) direction. A* *journey is* simple *if it traverses each point of the cycle at most twice. A clockwise (resp. counterclockwise) journey is a* loop *if it returns to p while moving in the clockwise (resp. counterclockwise) direction. An* excursion *is a journey that is not a loop. A clockwise (resp. counterclockwise) excursion is* quasi-regular *if it is simple and, in addition, r does not walk past the clockwise (resp. counterclockwise) farthest searched arc. A clockwise (resp. counterclockwise) excursion is* regular *if it is quasi-regular and, in addition, r does not search any arcs while moving in the counterclockwise (resp. clockwise) direction.*

Note that Definition 5 implies that a simple loop traverses each point of the domain (except p itself) exactly once.

Lemma 5. *For every correct schedule $\mathcal{S}$, there exists a correct schedule $\mathcal{S}'$, with the same robot partition, whose completion time is not greater than that of $\mathcal{S}$ and in which every robot r with origin p satisfies all of the following:*

1. *The trajectory of r consists either of a single simple loop, or of at most one clockwise regular excursion and at most one counterclockwise regular excursion that do not overlap.*
2. *Robot r only and definitively stops just after it searches a non-empty arc for the last time.*

Proof. We make the following modifications to the trajectory of each robot r with origin p, independently of the other robots. Let d^+ (resp. d^-) be the farthest clockwise (resp. counterclockwise) distance from p that r reaches during a clockwise (resp. counterclockwise) journey. We have $0 \leq d^- \leq L$ and $0 \leq d^+ \leq L$, where L is the circumference of the cycle.

If $d^- + d^+ \geq L$, then the whole trajectory of r can be replaced by a single simple loop (clockwise or counterclockwise), during which r searches all of the arcs which are searched by r under $\mathcal{S}$, in order of increasing distance from p in the chosen direction for the loop, stopping immediately after traversing the last such arc. Note that some of the searched arcs may be traversed in the reverse direction with respect to $\mathcal{S}$. The direction of the loop is clockwise if r searches an arc immediately clockwise adjacent to p, otherwise the loop is performed in the counterclockwise direction.

If $d^- + d^+ < L$, then $\mathcal{S}'$ will consist of two regular excursions in opposite directions. We assume for the moment that the last journey of r under $\mathcal{S}$ is a counterclockwise journey and we modify the trajectory of r as follows (Figure 3 contains an illustration):

– We replace all of the clockwise journeys of r by a single clockwise regular excursion that goes at most up to clockwise distance d^+, searching on the way all of the arcs that were searched by r in clockwise journeys under $\mathcal{S}$, in order of increasing clockwise distance from p, and then returns to p by walking counterclockwise immediately after searching the last such arc.
– We also replace all of the counterclockwise journeys of r by a single counterclockwise regular excursion that goes at most up to counterclockwise distance d^-, searching on the way all of the arcs that were searched by r in

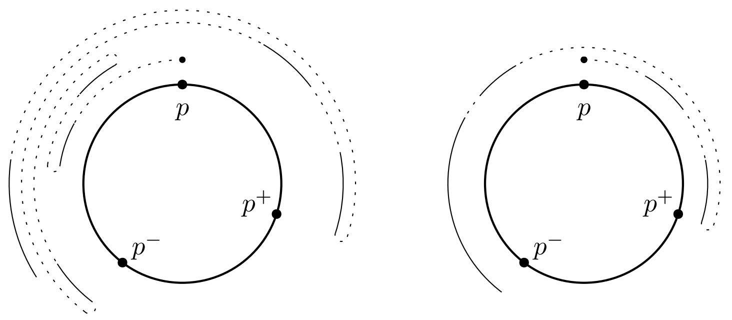

Fig. 3. An illustration of the proof of Lemma 5, when $d^- + d^+ < L$. *Left:* The trajectory of a single robot that consists of a counterclockwise journey up to point p^-, followed by a clockwise journey up to point p^+, followed by a counterclockwise journey up to the robot's stopping point. *Right:* The robot's trajectory is replaced by a single clockwise journey up to p^+ followed by a single counterclockwise journey up to p^-. Note that, in the new schedule, the robot searches the same arcs but it walks less.

counterclockwise journeys under $\mathcal{S}$, in order of increasing counterclockwise distance from p, and then stops immediately after searching the last such arc.

- Robot r executes first the above clockwise regular excursion and, subsequently, it executes the above counterclockwise regular excursion.

On the other hand, if the last journey of r under $\mathcal{S}$ is a clockwise journey, then we perform the same modifications as above, except that r returns to p during the counterclockwise regular excursion (which is now executed first) and stops after searching the farthest searched arc during the clockwise regular excursion.

In both cases, the new trajectory of r satisfies the two required properties, and the new overall schedule $\mathcal{S}'$ remains correct because r searches the same arcs as under $\mathcal{S}$, whereas the trajectories of all other robots remain unchanged. Moreover, the time required for r may clearly only decrease as a result of the modifications, therefore the completion time of $\mathcal{S}'$ is not greater than that of $\mathcal{S}$. □

Observation 2. Given a schedule $\mathcal{S}$, let r be a robot with origin p whose trajectory does not satisfy the properties of Lemma 5 and let $\mathcal{S}'$ be the new schedule that we obtain by applying the modifications described in the proof of Lemma 5 for robot r. Furthermore, assume that the trajectory of r in $\mathcal{S}$ does not consist of a simple loop starting and ending with a searched arc, or a (possibly empty) quasi-regular excursion followed by a regular excursion in the opposite direction. We claim that the completion time of r is strictly smaller than its completion time under $\mathcal{S}$.

Indeed, if $d^- + d^+ \geq L$, then, by construction of the trajectory of r in $\mathcal{S}'$, the total distance traversed by r in walking mode is strictly smaller in $\mathcal{S}'$ than under $\mathcal{S}$, whereas the total length of the searched arcs remains the same.

If $d^- + d^+ < L$, then r does not perform a loop under $\mathcal{S}$ and we can assume that the trajectory of r in $\mathcal{S}$ satisfies the second property, and contains at most one clockwise simple excursion and at most one counterclockwise simple excursion. Otherwise, r clearly walks strictly less under $\mathcal{S}'$ than under $\mathcal{S}$ and searches the same total distance as under $\mathcal{S}$. We can also assume that, of these two simple excursions, the one that is executed last is regular, otherwise r clearly walks strictly less in $\mathcal{S}'$. Therefore, the excursion that is executed first must not be quasi-regular, which implies that r walks past the farthest searched arc during that excursion. By construction of $\mathcal{S}'$ this extra useless walking is eliminated, therefore r walks strictly less under $\mathcal{S}'$.

We will use this observation in the proof of Lemma 14 in Subsection 3.5.

Lemma 6. *For every correct schedule $\mathcal{S}$, there exists a correct schedule $\mathcal{S}'$, with the same robot partition (but not necessarily the same sources), whose completion time is not greater than that of $\mathcal{S}$ and in which the following properties are satisfied:*

1. *The given cycle C_L is covered by at most t arcs with non-overlapping interiors, such that the robots originating from each robot source only move within its associated arc. If $t = 1$, the arc simply encompasses the whole cycle.*
2. *For each robot source p, a clockwise traversal of C_L starting from p first encounters all the arcs that are searched in clockwise journeys by robots originating from p, and then it encounters all the arcs that are searched in counterclockwise journeys by robots originating from p.*

Proof. Without loss of generality, we assume that the schedule $\mathcal{S}$ satisfies the properties of Lemma 5. We will show how to modify $\mathcal{S}$ so as to obtain $\mathcal{S}'$ with the required properties.

Fix a source p under $\mathcal{S}$ such that not all of its robots have empty trajectories. By Lemma 5, every robot originating from p performs either a single simple loop, or at most one clockwise regular excursion and at most one counterclockwise regular excursion that do not overlap. Let $R(p)$ (resp. $L(p)$) denote the sum of the lengths of all arcs that are searched in clockwise (resp. counterclockwise) journeys by robots originating from p. Consider a new arc of length $L(p)+R(p)$. This new arc can be searched completely by the group of robots that originate from p under $\mathcal{S}$, as follows: The source is a point p' at distance $L(p)$ from one endpoint of the new arc. Each robot follows exactly the same trajectory as under $\mathcal{S}$, except that during its clockwise (resp. counterclockwise) journey, the parts of its trajectory that are searched by robots not originating from p under $\mathcal{S}$, or that are searched during counterclockwise (resp. clockwise) journeys of robots originating from p, are reduced to zero-length segments (or “chopped off,” see Figure 4 for an illustration). Note that, under this schedule for the new arc of length $L(p)+R(p)$, the completion time of each robot is not greater than its completion time under $\mathcal{S}$.

Let $p_1, \ldots, p_{t'}$, where $t' \leq t$, be an arbitrary enumeration of the robot sources that contain robots with nonempty trajectories under $\mathcal{S}$. Note that

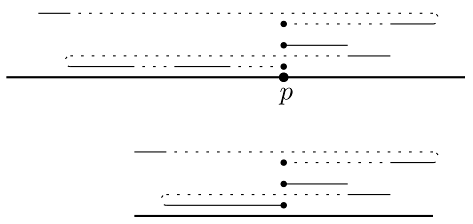

Fig. 4. An illustration of the proof of Lemma 6. *Top:* Schedule $\mathcal{S}$. A group of robots originating from point p search collectively a non-continuous subset of the arcs. The gaps are searched by robots originating from different robot sources (not shown in the figure). *Bottom:* Schedule $\mathcal{S}'$. The same group of robots follow the same trajectories (with the gaps having been eliminated) and they search collectively a continuous region of the same length as the non-continuous region that they searched under $\mathcal{S}$. The distance walked by each robot is clearly not increased compared to the distance walked by the same robot under $\mathcal{S}$, and there exists at least one robot that walks strictly less than under $\mathcal{S}$.

$\sum_{i=1}^{t'} (L(p_i) + R(p_i)) = L$. We construct a new schedule $\mathcal{S}'$ as follows. We partition C_L into t' arcs, where the i-th arc has length $L(p_i) + R(p_i)$. The i-th arc is searched completely by the robots that originated from p_i under $\mathcal{S}$, as explained in the previous paragraph.

The new schedule clearly satisfies property 1. If $t' \geq 2$, then property 1 implies property 2 in view of Lemma 5. If $t' = 1$, property 2 is guaranteed by construction of the new trajectories of the robots. $\square$

Observation 3. If $\mathcal{S}$ satisfies the properties of Lemma 5 but it does not satisfy those of Lemma 6, then in the new schedule $\mathcal{S}'$ that we obtain by applying Lemma 6, the completion time of at least one robot is strictly smaller than the completion time of the same robot under $\mathcal{S}$.

Indeed, if the trajectories of robots from two different sources overlap, then at least one of them will have part of its trajectory that is searched by a robot from a different source. That part of its trajectory will be reduced to a zero-length segment (chopped off) in $\mathcal{S}'$, therefore its completion time will be strictly reduced.

On the other hand, if there is only one robot source p under $\mathcal{S}$ whose robots have nonempty trajectories, and a clockwise traversal from p encounters an arc searched in a counterclockwise journey (say by robot r_1) before an arc searched in a clockwise journey (say by robot r_2), then both of r_1, r_2 will have parts of their trajectories reduced to zero-length segments.

We will use this observation in the proof of Lemma 14 in Subsection 3.5.

Lemma 7. *For every correct schedule $\mathcal{S}$, there exists a correct schedule $\mathcal{S}'$, with the same robot partition, whose completion time is not greater than that of $\mathcal{S}$, and in which each journey of the trajectory of each robot contains exactly one searched arc.*

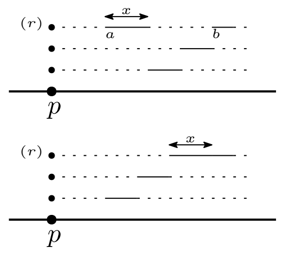

Fig. 5. An illustration of the proof of Lemma 7 (the clockwise direction is to the right in the figure). *Top:* Schedule $\mathcal{S}$. Robot r performs a clockwise journey that contains at least two searched arcs a and b. Two other robots search the part of the cycle between a and b. *Bottom:* Schedule $\mathcal{S}'$. Robot r searches and walks the same total lengths as under $\mathcal{S}$, but a and b are merged into a single searched arc. The searched arcs of the two other robots that lie between a and b have been shifted by x toward the robot source p.

Proof. Without loss of generality, we assume that the schedule $\mathcal{S}$ satisfies the properties of Lemmas 5 and 6. We will show how to modify $\mathcal{S}$ so as to obtain $\mathcal{S}'$ with the required property.

Let p be a robot source in $\mathcal{S}$ and let r be any robot starting from p that performs a clockwise (without loss of generality) journey that contains at least two searched arcs. Let a and b be two such arcs that are searched successively by r, of which a is closest to p. Finally, let x be the length of a. We modify the trajectory of r so that it no longer searches a, but it starts searching at distance x before reaching the beginning of b. Furthermore, for every robot r' that searches in $\mathcal{S}$ one or more arcs between a and b, we displace all such arcs by a distance of x counterclockwise. Figure 5 contains an illustration. As a result of this modification, we obtain a schedule in which r finishes at the same time because it walks and searches over the same distances as before, whereas every other robot whose trajectory was modified finishes at the same time or possibly earlier, because some of its searched arcs were moved closer to its origin. Therefore, the completion time of the new schedule is not greater than that of the original schedule.

Successive applications, for all robot sources p, of this modification (or its symmetric one for counterclockwise journeys) reduce by at least one the total number of searched arcs in every new schedule, up to the point where this process terminates with the desired schedule $\mathcal{S}'$ in which each journey of each robot contains one searched arc. $\square$

Observation 4. If $\mathcal{S}$ satisfies the properties of Lemmas 5 and 6 but it does not satisfy that of Lemma 7, then in the new schedule $\mathcal{S}'$ that we obtain by applying Lemma 7, the completion time of at least one robot is strictly smaller than the completion time of the same robot under $\mathcal{S}$.

Indeed, fix a robot source p and consider the last time that the modification described in the proof of Lemma 7 is applied to a robot starting from p. Then, *there must exist at least one searched arc c between a and b, which is the only arc searched by some robot r', and which is displaced, in the new schedule, by a distance of x closer to the origin p of r'. Therefore, r' walks less in $\mathcal{S}'$ before reaching c and its completion time is strictly decreased.*

We will use this observation in the proof of Lemma 14 in Subsection 3.5.

We call a schedule that satisfies the properties guaranteed by Proposition 1 and Lemmas 5, 6, and 7 *normal*:

Definition 6 (Normal schedule). *A schedule $\mathcal{S}$ is called* normal *if it satisfies the following properties:*

1. *Every pair of searched arcs (not necessarily by the same robot) have disjoint interiors.*
2. *During every time interval, every robot is either stopped or it moves at the maximum walking or searching speed, according to its chosen mode during that interval.*
3. *The trajectory of every robot consists either of a single simple loop, or of at most one clockwise regular excursion and at most one counterclockwise regular excursion that do not overlap.*
4. *Every robot only and definitively stops just after it searches a non-empty arc for the last time.*
5. *The given cycle C_L can be covered by at most t arcs with non-overlapping interiors, such that the robots originating from each robot source only move within its associated arc.*
6. *For each robot source p, a clockwise traversal of C_L starting from p first encounters all the arcs that are searched in clockwise journeys by robots originating from p, and then it encounters all the arcs that are searched in counterclockwise journeys by robots originating from p.*
7. *Each journey of the trajectory of each robot contains exactly one searched arc.*

It follows from the proofs of Proposition 1 and Lemmas 5, 6, and 7 that, for every correct schedule $\mathcal{S}$ that is not normal, there exists a correct normal schedule $\mathcal{S}'$ that has smaller or equal completion time. In other words, we can guarantee all of the properties ensured by Proposition 1 and Lemmas 5, 6, and 7 simultaneously. Moreover, in all cases, the new schedule $\mathcal{S}'$ can be computed in polynomial time.

We thus have the following:

Lemma 8. *For every non-normal correct schedule $\mathcal{S}$, there exists a normal correct schedule $\mathcal{S}'$, with the same robot partition, whose completion time is not greater than that of $\mathcal{S}$. The normal schedule $\mathcal{S}'$ can be computed from $\mathcal{S}$ in polynomial time.*

3.2 Existence of optimal and normal t-$\mathsf{SBPC_z}$ schedules

Given the statement of the t-$\mathsf{SBPC_z}$ problem in Definition 3, it is not immediately obvious that there exists an optimal solution. Indeed, since the set of feasible solutions in a given instance is clearly infinite, it might be the case that there exist infinite sequences of solutions with strictly decreasing completion times, converging to the infimum completion time over all feasible solutions, without there being a solution that actually attains that infimum. In this subsection, we prove that there exists, in fact, an optimal solution, which is also normal in the sense of Definition 6.

Note that, in view of Lemma 8, for every infinite sequence of solutions with strictly decreasing completion times that converge to some limit $\mathfrak{c}$, there exists an infinite sequence of *normal* solutions with strictly decreasing completion times, which converge to some limit $\mathfrak{c}' \leq \mathfrak{c}$. It suffices, therefore, to restrict our attention to the set of normal solutions $\mathcal{N}$ to a given instance and prove that it contains one that is optimal.

In view of Definition 6, a normal solution for t-$\mathsf{SBPC_z}$ can be specified (up to the actual cyclic order of the robot group starting points) by choosing the following parameters:

1. The number of robot groups $t' \leq t$.
2. The partition of robots into t' groups.
3. For each robot, the number of journeys in its trajectory (one or two).
4. For each robot, the direction of its first journey (clockwise or counterclockwise).
5. For each robot group, two orders of the robots of the group. A robot r precedes a robot r' in the first (resp. second) order if the arc searched by r during its clockwise (resp. counterclockwise) journey is closer to p than the arc searched by r' during its clockwise (resp. counterclockwise) journey, where p is the yet unspecified origin of the group.
6. For each robot r, the length of the arc that it searches during its clockwise journey and the length of the arc that it searches during its counterclockwise journey.

Fixing the above parameters also fixes the length of the arc that is associated to each group (cf. Definition 6, item 5), as well as the starting point of the group within its associated arc. Different cyclic orders of the robot group origins yield different solutions, but we will consider these as equivalent for our purposes.

We next observe that, once parameters 1-5 above are fixed, the completion time of the trajectory of each robot is a linear function of the lengths of the searched arcs of all robots during their clockwise and counterclockwise journeys (parameter 6). One can then formulate a linear program that seeks the lengths of the searched arcs of the robots so as to minimize the maximum completion time over all robot trajectories, subject to the constraint that the lengths of all searched arcs are nonnegative and their sum is equal to L.

Let $\mathcal{P}$ denote the set of all possible (joint) choices for parameters 1-5, and let $T(y)$ be the optimal value of the linear program that corresponds to the choice of parameters $y \in \mathcal{P}$. Note that the size of $\mathcal{P}$ is finite, since it is bounded by a function of n and t. Therefore, $\min_{y \in \mathcal{P}} T(y)$ is well defined and it is the completion time of the optimal schedule in $\mathcal{N}$. We thus proved the following:

Theorem 2. *Every instance of t-$\mathsf{SBPC_z}$ has an optimal and normal solution.*

3.3 Properties of optimal and normal t-$\mathsf{SBPC_z}$ schedules

Definition 7. *Given a normal schedule, we use the term* clockwise *(resp.* counterclockwise*)* leg *of a robot to refer to the clockwise (resp. counterclockwise) part of a clockwise (resp. counterclockwise) journey of that robot.*

The leg of a journey is, in a sense, the "interesting" part of the journey, i.e., the part of the trajectory from the origin of the robot to the farthest point reached in that journey, during which the robot searches at least one arc.

In this subsection, we show that in every optimal and normal schedule, the following assertions hold:

1. All robots terminate their trajectories simultaneously (Corollary 1).
2. The trajectory of each robot contains exactly one leg (Lemma 11).

Lemma 11 establishes that, in an optimal solution, robots do not change direction of movement.

With every fixed normal schedule $\mathcal{S}$, we associate the corresponding partition of the circle into pairwise interior-disjoint arcs, each of which is searched by a single robot that is moving in the same direction over a continuous time interval. In view of Lemma 5, we may assume that no robot source is in the interior of any of the arcs.

Definition 8. *Let $\mathcal{S}$ be a normal schedule for t-$\mathsf{SBPC_z}$. We denote by $\mathcal{A}^+_\mathcal{S}$ (resp. $\mathcal{A}^-_\mathcal{S}$) the set of searched arcs that belong to clockwise (resp. counterclockwise) legs of robots. If p is a source under $\mathcal{S}$, then $\mathcal{A}^+_\mathcal{S}(p)$ (resp. $\mathcal{A}^-_\mathcal{S}(p)$) denotes the set of arcs searched in the clockwise (resp. counterclockwise) direction by robots originating from p. For $a, b \in \mathcal{A}^+_\mathcal{S} \cup \mathcal{A}^-_\mathcal{S}$ and p a point on C_L, we write $a \prec_p b$ if a clockwise traversal from p encounters first a and then b.*

For the purpose of stating the next lemma, given a normal schedule $\mathcal{S}$ with completion time T, we will denote by $\mathcal{I}(\mathcal{S})$ the inclusion-maximal set of searched arcs that satisfies the property that each arc in $\mathcal{I}(\mathcal{S})$ is searched by a robot that stops strictly earlier than T and, for each robot source p, the union of all arcs in $\mathcal{I}(\mathcal{S}) \cap (\mathcal{A}^+_\mathcal{S}(p) \cup \mathcal{A}^-_\mathcal{S}(p))$ is a continuous arc that contains p. We will denote by $R(\mathcal{S})$ the number of distinct robots that search the arcs in $\mathcal{I}(\mathcal{S})$.

Lemma 9. *Let $\mathcal{S}$ be a normal correct schedule for t-$\mathsf{SBPC_z}$ with completion time T, such that at least one robot stops at time $T - \epsilon$, for some $\epsilon > 0$. Then, there exists a normal correct schedule $\mathcal{S}'$ such that, either its completion time is strictly smaller than T, or its completion time is equal to T but $R(\mathcal{S}') > R(\mathcal{S})$.*

Proof. By assumption, there exist not necessarily distinct robot sources p and q such that some robot originating from p terminates strictly earlier than T and some robot originating from q terminates at time T, under $\mathcal{S}$. We distinguish two cases:

Case 1: All robots originating from p terminate strictly earlier than T under $\mathcal{S}$.

If all robots originating from p terminate strictly earlier than T under $\mathcal{S}$, then there must exist robot sources p' and q' such that all robots originating from p' terminate strictly earlier than T, at least one robot originating from q' terminates at time T, and the arcs associated to p' and q' (cf. Definition 6, item 5) are adjacent.

Observation 5. Let α be a positive real. For a given set of robots, if there exists a correct single-source schedule with completion time T for a line segment of length L, then there exists a correct single-source schedule with completion time αT for a line segment of length αL.

Now, the idea is to slightly extend the arc searched collectively by robots originating from p' (while making sure that all of these robots still terminate strictly earlier than T) and correspondingly contract the arc searched by robots originating from q', which results in all of these robots terminating strictly earlier than T. The extension and contraction of the arcs searched by robots originating from p' and q', respectively, are possible due to Observation 5.

The new schedule $\mathcal{S}'$ has completion time at most T. If the completion time of $\mathcal{S}'$ remains T, then all robots that were contributing to $R(\mathcal{S})$ still contribute to $R(\mathcal{S}')$, and there is at least one robot (namely, the one originating from q' that terminates at time T under $\mathcal{S}$) that contributes to $R(\mathcal{S}')$ but does not contribute to $R(\mathcal{S})$. Therefore, $R(\mathcal{S}') > R(\mathcal{S})$.

Case 2: There exists a robot originating from p that terminates at time T under $\mathcal{S}$.

In this case, let a be the minimum element of $(\mathcal{A}^+_{\mathcal{S}}(p) \cup \mathcal{A}^-_{\mathcal{S}}(p)) \setminus \mathcal{I}(\mathcal{S})$ under $\prec_p$ and let b be the maximum element of $(\mathcal{A}^+_{\mathcal{S}}(p) \cup \mathcal{A}^-_{\mathcal{S}}(p)) \setminus \mathcal{I}(\mathcal{S})$ under $\prec_p$ (each of a and b is adjacent to some arc in $\mathcal{I}(\mathcal{S})$ or to the origin p). Since $\mathcal{S}$ is normal, we have $a \in \mathcal{A}^+_{\mathcal{S}}(p)$ or $b \in \mathcal{A}^-_{\mathcal{S}}(p)$. We assume without loss of generality that $a \in \mathcal{A}^+_{\mathcal{S}}$ and that a is searched by some robot j. Moreover, robot j must be terminating its trajectory at time T, otherwise arc a would be in $\mathcal{I}(\mathcal{S})$. Furthermore, by minimality of a, there exists some robot $i \neq j$ that terminates at time $T - \epsilon$, for some $\epsilon > 0$, and that visits the clockwise starting extremity u of a. Note that if this point u is not the same as p, then robot i must visit u during its clockwise leg.

We distinguish three subcases, which are illustrated in Figure 6. If the trajectory of robot i is such that the robot goes clockwise after its first visit of u (therefore it walks over the entire length of arc a), then we modify it so that it searches a sub-arc c of a of length $\min(\frac{\epsilon}{2} \cdot \frac{w_i s_i}{w_i - s_i}, \frac{|a|}{2})$, instead of walking over c. If the trajectory of robot i is such that the robot goes counterclockwise after its first visit of u, then we modify it so that it searches a sub-arc c of a of

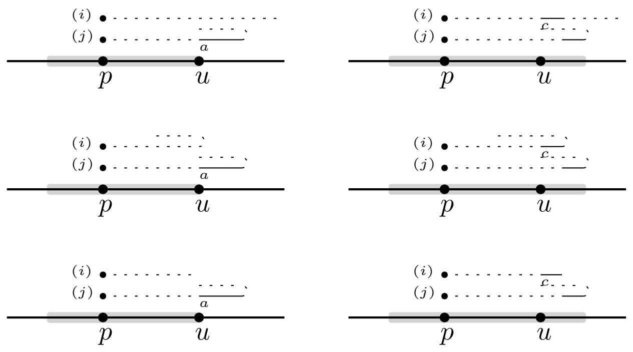

Fig. 6. An illustration of the modifications to schedule $\mathcal{S}$ in Case 2 of the proof of Lemma 9 (the clockwise direction is to the right in the figure). Assuming that j is not the only robot that terminates at time T under $\mathcal{S}$, the shaded part around p in the figures on the left represents $\mathcal{I}(\mathcal{S}) \cap (\mathcal{A}^+_{\mathcal{S}}(p) \cup \mathcal{A}^-_{\mathcal{S}}(p))$ and the shaded part in the figures on the right represents $\mathcal{I}(\mathcal{S}') \cap (\mathcal{A}^+_{\mathcal{S}'}(p) \cup \mathcal{A}^-_{\mathcal{S}'}(p))$. In all three cases, robot j contributes to $R(\mathcal{S}')$ but not to $R(\mathcal{S})$, therefore $R(\mathcal{S}') > R(\mathcal{S})$. *Top:* Subcase 1: i continues clockwise after its first visit of u. *Middle:* Subcase 2: i goes counterclockwise after its first visit of u. *Bottom:* Subcase 3: i stops upon its first visit of u.

length $\min(\frac{\epsilon}{2} \cdot \frac{w_i s_i}{w_i + s_i}, \frac{|a|}{2})$, then walks counterclockwise over c, and then resumes its previous trajectory. Finally, if robot i never moves after its first visit of u (this includes the case when u is the same as p and robot i stops immediately at the source), then we extend its trajectory so that it searches a sub-arc c of a of length $\min(\frac{\epsilon}{2} \cdot s_i, \frac{|a|}{2})$ and then stops.

In all cases, we alter the trajectory of robot j so that it walks over c and then continues with searching the remaining part of arc a and then with the rest of its previous trajectory. In the new schedule $\mathcal{S}'$, both i and j terminate strictly before time T, while the trajectories of other robots remain the same.

If j is the only robot to finish at time T under $\mathcal{S}$, then the completion time of $\mathcal{S}'$ is strictly smaller than T. Otherwise, $\mathcal{S}'$ finishes at time T and $R(\mathcal{S}') > R(\mathcal{S})$. □

From Lemma 9, we obtain the following important corollaries:

Corollary 1. *In every optimal and normal schedule for t-$\mathsf{SBPC_z}$, all robots terminate their trajectories simultaneously.*

Proof. For the purpose of contradiction, let $\mathcal{S}$ be an optimal and normal schedule for a given set of n robots with completion time T, in which at least one robot terminates earlier than T, and which maximizes $R(\mathcal{S})$ (which is at most $n-1$ by definition). Lemma 9 should apply, but the completion time cannot decrease because of $\mathcal{S}$ being optimal, and $R(\mathcal{S})$ cannot increase because of its maximality, which leads to the desired contradiction. □

Corollary 2. *In every optimal and normal schedule for t-$\mathsf{SBPC_z}$, the trajectory of each robot contains at least one leg.*

We are now ready to further restrict the structure of optimal and normal schedules. We first show that there are no *crossing robots* (Lemma 10, cf. Definition 9), and then that each robot performs only one leg (Lemma 11).

Definition 9 (Crossing robots). *Let $\mathcal{S}$ be a normal schedule. We say that a pair of robots i, j originating from the same robot source p* cross *under $\mathcal{S}$ if robot i searches arcs $a_i^+ \in \mathcal{A}_\mathcal{S}^+(p)$ and $a_i^- \in \mathcal{A}_\mathcal{S}^-(p)$ (in any order), robot j searches arcs $a_j^+ \in \mathcal{A}_\mathcal{S}^+(p)$ and $a_j^- \in \mathcal{A}_\mathcal{S}^-(p)$ (in any order), and $a_i^+ \prec_p a_j^+ \prec_p a_i^- \prec_p a_j^-$ or $a_j^+ \prec_p a_i^+ \prec_p a_j^- \prec_p a_i^-$.*

An example of crossing robots is shown in Figure 7 (left).

Lemma 10. *No optimal and normal schedule contains a pair of crossing robots.*

Proof. Let $\mathcal{S}$ be an optimal and normal schedule in which robot i, originating from robot source p, searches arcs $a_i^+ \in \mathcal{A}_\mathcal{S}^+(p)$ and $a_i^- \in \mathcal{A}_\mathcal{S}^-(p)$ (in any order) and robot j, originating from the same robot source p, searches arcs $a_j^+ \in \mathcal{A}_\mathcal{S}^+(p)$ and $a_j^- \in \mathcal{A}_\mathcal{S}^-(p)$ (in any order). By item 7 of Definition 6, these are the only arcs searched by robots i and j. By Corollary 1, all robots terminate their trajectories simultaneously. For a contradiction, suppose that robots i and j cross under $\mathcal{S}$, i.e., $a_i^+ \prec_p a_j^+ \prec_p a_i^- \prec_p a_j^-$ (Figure 7, left). Fix an ϵ in the range $0 < \epsilon < \min(|a_i^+|, |a_j^-|)$. Let b be the sub-arc of length ϵ of a_j^- that lies at its counterclockwise end and let c be the sub-arc of length ϵ of a_i^+ that lies at its clockwise end. By assumption, under $\mathcal{S}$, robot i walks over b on its way to a_i^- and robot j walks over c on its way to a_j^+.

We modify the trajectory of robot i so that it searches b and a_i^- in its counterclockwise leg, and $a_i^+ \setminus c$ in its clockwise leg. Similarly, we modify the trajectory of robot j so that it searches c and a_j^+ in its clockwise leg, and $a_j^- \setminus b$ in its counterclockwise leg. Figure 7 illustrates the modifications. These modifications result in a correct and normal schedule $\mathcal{S}'$, in which the distances walked by robot i and by robot j are shortened by at least ϵ with respect to $\mathcal{S}$, whereas the distance searched by each robot remains the same. Therefore, under $\mathcal{S}'$, both robots conclude their trajectory strictly earlier than under $\mathcal{S}$. Since no robot increases the completion time of its own trajectory, either $\mathcal{S}'$ takes less time than $\mathcal{S}$, which contradicts the optimality of $\mathcal{S}$, or $\mathcal{S}'$ is also optimal and some robots terminate their trajectories earlier than others. This contradicts Corollary 1. □

Lemma 11. *In every optimal and normal schedule, the trajectory of each robot contains exactly one leg.*

Proof. Let $\mathcal{S}$ be an optimal and normal schedule in which at least one robot has a trajectory with a clockwise leg and a counterclockwise leg. By Lemma 10, there exists a robot $i^\star$ with a two-leg trajectory that originates from some robot source p and searches arcs $a_{i^\star}^+ \in \mathcal{A}_\mathcal{S}^+(p)$ and $a_{i^\star}^- \in \mathcal{A}_\mathcal{S}^-(p)$, such that $a_{i^\star}^+$ is greater under $\prec_p$ than every other arc in $\mathcal{A}_\mathcal{S}^+(p)$ that is searched by a robot that originates from p and whose trajectory has two legs, and $a_{i^\star}^-(p)$ is smaller under $\prec_p$

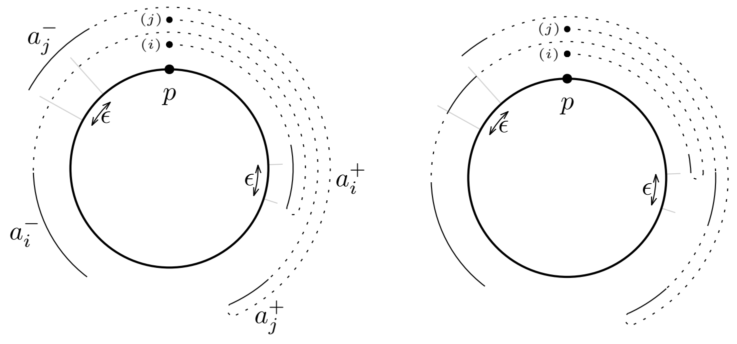

Fig. 7. An illustration of the proof of Lemma 10. *Left:* Schedule $\mathcal{S}$. Robots i and j form a pair of crossing robots. *Right:* Schedule $\mathcal{S}'$. Both robots search the same total lengths as under $\mathcal{S}$. Robot i walks a distance of 2ϵ less and robot j walks a distance of ϵ less than under $\mathcal{S}$.

than every other arc in $\mathcal{A}^-_{\mathcal{S}}(p)$ that is searched by a robot that originates from p and whose trajectory has two legs. In particular, for every arc $z \in \mathcal{A}^+_{\mathcal{S}}(p) \cup \mathcal{A}^-_{\mathcal{S}}(p)$ such that $a^+_{i^\star} \prec_p z \prec_p a^-_{i^\star}$, z is searched by a robot whose trajectory has only one leg. Without loss of generality, we assume that robot $i^\star$ first performs its clockwise leg and then walks back to the origin and performs its counterclockwise leg, the other case being handled by a completely symmetric argument. Let $r_1, \ldots, r_{\rho^-}$ be the robots that search the arcs in $\{z : a^+_{i^\star} \prec_p z \prec_p a^-_{i^\star}\} \cap \mathcal{A}^-_{\mathcal{S}}(p)$, ordered so that $a_{r_{\rho^-}} \prec_p \cdots \prec_p a_{r_1} \prec a^-_{i^\star}$, where a_{r_j} is the arc searched by robot r_j. Similarly, let $q_1, \ldots, q_{\rho^+}$ be the robots that search the arcs in $\{z : a^+_{i^\star} \prec_p z \prec_p a^-_{i^\star}\} \cap \mathcal{A}^+_{\mathcal{S}}(p)$, ordered so that $a^+_{i^\star} \prec_p a_{q_1} \prec_p \cdots \prec_p a_{q_{\rho^+}}$, where a_{q_j} is the arc searched by robot q_j (Figure 8, bottom).

Our goal is to modify the lengths of the segments $a^+_{i^\star}$, $a^-_{i^\star}$, a_{r_j} for $1 \le j \le \rho^-$, and a_{q_j} for $1 \le j \le \rho^+$, without changing their relative order, their assignment to robots, or their total length, so that the time required for robot $i^\star$ to conclude its trajectory is reduced, whereas the time required for all other robots remains the same. Note that the robot source p may need to be shifted to a new point p', in order to ensure that the completion times of the robots that search the segments $\{z : z \prec_p a^+_{i^\star} \vee a^-_{i^\star} \prec_p z\}$ remain the same as under $\mathcal{S}$ (Figure 8, top). Although the robot source p is displaced in the new schedule, the arc associated to the robot source p' under the new schedule is at the same position on the cycle and has the same length as the arc associated to p under $\mathcal{S}$. In this way, we obtain a schedule for the group of robots originating from p' in which: (a) the robots originating from p' cover collectively the same sub-segment as under $\mathcal{S}$ in at most the same time and (b) one robot terminates strictly earlier than the others. This contradicts Corollary 1.

To this end, we introduce the following variables:

- δ^+: the amount by which $a^+_{i^\star}$ is lengthened.
- δ^-: the amount by which $a^-_{i^\star}$ is lengthened.
- ϵ_{q_j}, for $1 \le j \le \rho^+$: the amount by which a_{q_j} is lengthened.

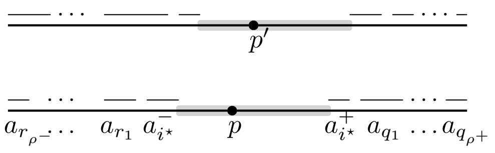

Fig. 8. An illustration of the modifications to schedule $\mathcal{S}$ in the proof of Lemma 11 (the clockwise direction is to the right in the figure). *Bottom:* The cycle arc corresponding to robot source p under $\mathcal{S}$. The shaded part around p is covered by robots whose trajectories are not modified in $\mathcal{S}'$. *Top:* Under $\mathcal{S}'$, the robot source p is displaced to p'. The trajectories of the robots that covered the shaded part under $\mathcal{S}$ are not modified, and therefore they cover the corresponding shaded part under $\mathcal{S}'$.

- ϵ_{r_j}, for $1 \le j \le \rho^-$: the amount by which a_{r_j} is lengthened.

All of these variables may be associated with positive or negative values. The modified searched arcs of robots r_j $(1 \le j \le \rho^-)$ and q_j $(1 \le j \le \rho^+)$ must still cover the same total length of the cycle as they did under $\mathcal{S}$. This yields the following constraint:

$$\sum_{\lambda=1}^{\rho^-} \epsilon_{r_\lambda} + \sum_{\lambda=1}^{\rho^+} \epsilon_{q_\lambda} + \delta^- + \delta^+ = 0 \tag{1}$$

We demand that, in the new schedule, robot $i^\star$ complete its trajectory strictly earlier than in $\mathcal{S}$. Given the modifications to $a^+_{i^\star}$ and $a^-_{i^\star}$, robot $i^\star$ walks δ^+ more than in $\mathcal{S}$ and searches $\delta^+ + \delta^-$ more than in $\mathcal{S}$. Therefore, we must have $\frac{\delta^+}{w_{i^\star}} + \frac{\delta^+ + \delta^-}{s_{i^\star}} < 0$ or, slightly rearranged:

$$\delta^- < -\delta^+ \cdot \left(1 + \frac{s_{i^\star}}{w_{i^\star}}\right) \tag{2}$$

We also demand that, in the new schedule, each robot r_j and each robot q_j conclude its trajectory at the same time as in the original schedule. Robot r_j walks $\sum_{\lambda=1}^{j-1} \epsilon_{r_\lambda} + \delta^-$ more than in $\mathcal{S}$ and searches ϵ_{r_j} more than in $\mathcal{S}$. Similarly, robot q_j walks $\sum_{\lambda=1}^{j-1} \epsilon_{q_\lambda} + \delta^+$ more than in $\mathcal{S}$ and searches ϵ_{q_j} more than in $\mathcal{S}$. We have, therefore, the following sets of constraints:

$$\frac{\sum_{\lambda=1}^{j-1} \epsilon_{r_\lambda} + \delta^-}{w_{r_j}} + \frac{\epsilon_{r_j}}{s_{r_j}} = 0 \text{ , for } j = 1, \dots, \rho^- \tag{3}$$

$$\frac{\sum_{\lambda=1}^{j-1} \epsilon_{q_\lambda} + \delta^+}{w_{q_j}} + \frac{\epsilon_{q_j}}{s_{q_j}} = 0 \text{ , for } j = 1, \dots, \rho^+ \tag{4}$$

By Eq. 3 and 4, we determine the values of ϵ_{r_j} $(1 \le j \le \rho^-)$ and ϵ_{q_j} $(1 \le j \le \rho^+)$ in terms of δ^+ and δ^- as follows:

$$\epsilon_{r_j} = -\delta^- \cdot \frac{s_{r_j}}{w_{r_j}} \cdot \prod_{\lambda=1}^{j-1} \left(1 - \frac{s_{r_\lambda}}{w_{r_\lambda}}\right) \text{ , for } j = 1, \dots, \rho^- \tag{5}$$

$$\epsilon_{q_j} = -\delta^+ \cdot \frac{s_{q_j}}{w_{q_j}} \cdot \prod_{\lambda=1}^{j-1} \left(1 - \frac{s_{q_\lambda}}{w_{q_\lambda}}\right) \text{ , for } j = 1, \dots, \rho^+ \tag{6}$$

If we plug Eq. 5 and 6 into Eq. 1, we obtain:

$$\delta^- \cdot (1 - R) + \delta^+ \cdot (1 - Q) = 0 \tag{7}$$

where R and Q are defined as follows:

$$R = \sum_{j=1}^{\rho^-} \left(\frac{s_{r_j}}{w_{r_j}} \cdot \prod_{\lambda=1}^{j-1} \left(1 - \frac{s_{r_\lambda}}{w_{r_\lambda}}\right)\right) , \qquad Q = \sum_{j=1}^{\rho^+} \left(\frac{s_{q_j}}{w_{q_j}} \cdot \prod_{\lambda=1}^{j-1} \left(1 - \frac{s_{q_\lambda}}{w_{q_\lambda}}\right)\right) \tag{8}$$

These equations can be transformed into the following ones:

$$R = 1 - \prod_{\lambda=1}^{\rho^-} \left(1 - \frac{s_{r_\lambda}}{w_{r_\lambda}}\right) , \qquad Q = 1 - \prod_{\lambda=1}^{\rho^+} \left(1 - \frac{s_{q_\lambda}}{w_{q_\lambda}}\right) \tag{9}$$

Due to the fact that, for every robot i, $0 < s_i < w_i$, we conclude from Eq. 9 that $0 \leq R < 1$ and $0 \leq Q < 1$, and, in particular, $R = 0$ (resp. $Q = 0$) if and only if $\rho^- = 0$ (resp. $\rho^+ = 0$).

In order to choose appropriate values for δ^- and δ^+, we argue as follows.

Case 1: $\frac{1-Q}{1-R} > 1 + \frac{s_{i^\star}}{w_{i^\star}}$. By Eq. 7, we have $\delta^- = -\delta^+ \cdot \frac{1-Q}{1-R}$. We choose, then, a positive value for δ^+ and we obtain $\delta^- = -\delta^+ \cdot \frac{1-Q}{1-R} < -\delta^+ \cdot \left(1 + \frac{s_{i^\star}}{w_{i^\star}}\right)$, thus Eq. 2 is satisfied. The exact value of δ^+ should be small enough that none of the shortened arcs ($a^-_{i^\star}$ and a_{q_j}, $j = 1, \dots, \rho^+$) is shortened to a negative length, i.e.:

$$0 < \delta^+ \leq \min \left(\left|a^-_{i^\star}\right| \cdot \frac{1-R}{1-Q}, \min_{1 \leq j \leq \rho^+} \frac{\left|a_{q_j}\right|}{\frac{s_{q_j}}{w_{q_j}} \cdot \prod_{\lambda=1}^{j-1} \left(1 - \frac{s_{q_\lambda}}{w_{q_\lambda}}\right)} \right) \tag{10}$$

The new schedule contradicts Corollary 1.

Case 2: $\frac{1-Q}{1-R} < 1 + \frac{s_{i^\star}}{w_{i^\star}}$. By Eq. 7, we have $\delta^- = -\delta^+ \cdot \frac{1-Q}{1-R}$. We choose, then, a negative value for δ^+ and we obtain $\delta^- = -\delta^+ \cdot \frac{1-Q}{1-R} < -\delta^+ \cdot \left(1 + \frac{s_{i^\star}}{w_{i^\star}}\right)$, thus Eq. 2 is satisfied. Again, the exact value of δ^+ should be small enough (in absolute value) that none of the shortened arcs ($a^+_{i^\star}$ and a_{r_j}, $j = 1, \dots, \rho^-$) is shortened to a negative length, i.e.:

$$0 > \delta^+ \geq -\min \left(\left|a^+_{i^\star}\right| , \min_{1 \leq j \leq \rho^-} \frac{\left|a_{r_j}\right|}{\frac{s_{r_j}}{w_{r_j}} \cdot \prod_{\lambda=1}^{j-1} \left(1 - \frac{s_{r_\lambda}}{w_{r_\lambda}}\right)} \right) \tag{11}$$

The new schedule contradicts Corollary 1.

Case 3: $\frac{1-Q}{1-R} = 1 + \frac{s_{i^\star}}{w_{i^\star}}$. In this case, for all values of δ^- and δ^+ that satisfy Eq. 7, Eq. 2 holds with equality and thus the completion time of the trajectory of robot $i^\star$ remains the same as in the original schedule $\mathcal{S}$. However, if we set δ^+

to the maximum value allowed by Eq. 10, we are guaranteed that either $a_{i^\star}^-$ or some arc a_{q_j} is reduced to length 0. Then, we can replace this schedule by a schedule in which the corresponding robot does not execute at all the leg that contains the searched arc whose length is reduced to zero. If that robot is $i^\star$, then it will execute only its clockwise leg, whereas if that robot is some robot q_j, then it will stop at its origin at time 0. In both cases, the new schedule contradicts Corollary 1. □

3.4 Hardness and approximability of t-$\mathsf{SBPC_z}$

With Lemmas 8 and 11, we have the necessary tools in order to connect the solutions for t-$\mathsf{SBPC_z}$ instances to those for $2t$-SBP instances:

Lemma 12. *For a fixed $t \geq 1$, let $\mathcal{I}$ be an instance of $2t$-SBP on a finite segment of length L and let $\mathcal{I}'$ be the corresponding instance of t-$\mathsf{SBPC_z}$ with the same set of robots on a cycle of circumference L. Every solution for $\mathcal{I}$ can be transformed in polynomial time to a solution for $\mathcal{I}'$ with smaller or equal cost.*

Proof. Let $\mathcal{S}$ be a schedule for $\mathcal{I}$, with $t' \leq 2t$ robot sources. The same schedule $\mathcal{S}$ can also be seen as a $2t$-$\mathsf{SBPC_z}$ schedule for C_L (the cycle of circumference L), if we imagine that the finite segment of length L is bent into a cycle by identifying its two endpoints. By Lemma 8, we can compute in polynomial time a normal schedule $\mathcal{S}'$ for C_L, with the same number t' of robot sources, and with smaller or equal cost.

By inspection of the proofs of Proposition 1 and Lemmas 5, 6, and 7, the schedule $\mathcal{S}'$ retains the property of $\mathcal{S}$ that robots do not change direction of movement and, additionally, all robots that belong to the same robot group move in the same direction. Moreover, in view of this property, the arc associated to each robot source p (cf. Definition 6, item 5) has p at one of its extremities. We show now how to convert $\mathcal{S}'$ into a t-$\mathsf{SBPC_z}$ schedule, and therefore into a schedule for $\mathcal{I}'$.

We call *flipping* a group of robots that start from a source p and search collectively an arc s the operation whereby the source of the group is moved to the other endpoint of s and the trajectories of the robots are executed in the inverse direction with respect to the original schedule. Note that successive flips of groups of robots starting from schedule $\mathcal{S}'$ do not affect the correctness or the completion time of the schedule. Our goal is to flip certain groups of robots, so as to end up with a schedule $\mathcal{S}''$ in which at most $\lceil \frac{t'}{2} \rceil \leq t$ distinct points serve as robot sources. Then, the desired t-$\mathsf{SBPC_z}$ schedule for $\mathcal{I}'$ is a schedule which is identical to $\mathcal{S}''$, except that, for each of those distinct points, all robots that start from that point are merged into a single group of robots.

Let $p_1, \ldots, p_{t'}$ be an enumeration of the robot sources in $\mathcal{S}'$, starting from an arbitrary robot source p_1 and proceeding clockwise, breaking ties arbitrarily. For $i = 1, \ldots, \lfloor \frac{t'}{2} \rfloor$, we look at p_{2i-1} and p_{2i}. If, for both of these robot sources, the direction of movement of the robots is the same, then we flip one of the two groups so that their sources coincide. On the other hand, if the two groups of robots move toward each other, then we flip both groups, which also results in both groups having the same point as source. Note that, if the two groups move away from each other, then the two robot sources must already be identical, or the interior of the clockwise arc $p_{2i-1} \rightsquigarrow p_{2i}$ would not be searched by any robot. The resulting schedule $\mathcal{S}''$ has $\lceil \frac{t'}{2} \rceil \leq t$ distinct points that serve as robot sources, as desired. □

Lemma 13. *For a fixed $t \geq 1$, let $\mathcal{I}$ be an instance of t-$\mathsf{SBPC_z}$ on a cycle of circumference L and let $\mathcal{I}'$ be the corresponding instance of $2t$-SBP with the same set of robots on a finite segment of length L. Every optimal solution for $\mathcal{I}$ can be transformed in polynomial time to a solution for $\mathcal{I}'$ with equal cost.*

Proof. Let $\mathcal{S}$ be an optimal schedule for $\mathcal{I}$ with $t' \leq t$ robot sources. By Lemma 8, we can compute from $\mathcal{S}$ in polynomial time an optimal and normal schedule $\mathcal{S}'$ for $\mathcal{I}$ with t' robot sources.

By Lemma 11, for each of the t' robot sources in $\mathcal{S}'$, some of the robots assigned to that point move clockwise without changing direction and the rest move counterclockwise without changing direction. Now, we choose an arbitrary robot source and we cut the cycle at one of the extremities of its associated arc (cf. Definition 6, item 5), thus obtaining a segment of length L. For each robot source, we divide its robots into two groups moving in opposite directions. These robot groups are now separate, but they happen to have the same source point. Since each group from $\mathcal{S}'$ is now split into at most two new groups, and all the robots from each new group move in the same direction, we obtain a valid solution with the same completion time for $\mathcal{I}'$, with at most $2t' \leq 2t$ robot groups. □

Corollary 3. *For a fixed $t \geq 1$, let $\mathcal{I}$ be an instance of t-$\mathsf{SBPC_z}$ on a cycle of circumference L and let $\mathcal{I}'$ be the corresponding instance of $2t$-SBP with the same set of robots on a finite segment of length L. The optimal solutions for $\mathcal{I}$ and $\mathcal{I}'$ have the same cost.*

In view of Lemmas 12 and 13 and Corollary 3, the results in [13] for $2t$-SBP carry over to t-$\mathsf{SBPC_z}$. In particular, it is shown in [13] that 2-SBP is NP-hard even when all robots have the same walking speed ([13, Theorem 5]), and that 2-SBP admits a 0.5569-approximation algorithm that runs in polynomial time ([13, Lemma 4]). We thus obtain the following for 1-$\mathsf{SBPC_z}$:

Theorem 3. 1-$\mathsf{SBPC_z}$ *is* NP*-hard, even when all robots have the same walking speed.*

Theorem 4. 1-$\mathsf{SBPC_z}$ *admits a* 0.5569*-approximation algorithm that runs in polynomial time.*

Moreover, it is shown in [13] that t-SBP can be solved optimally in polynomial time if all robots have the same search speed ([13, Theorem 2]) and that t-SBP admits a randomized algorithm which achieves an expected approximation ratio of $1 - \left(1 - \frac{1}{t}\right)^t$, needs $\mathcal{O}(n \log t)$ random bits, and runs in polynomial time ([13, Theorem 3]). We thus obtain the following for t-$\mathsf{SBPC_z}$:

Theorem 5. *t-$\mathsf{SBPC_z}$ instances in which all robots have the same search speed can be solved optimally in polynomial time.*

Theorem 6. *t-$\mathsf{SBPC_z}$ admits a randomized algorithm which achieves an expected approximation ratio of $1 - \left(1 - \frac{1}{2t}\right)^{2t}$, needs $\mathcal{O}(n \log t)$ random bits, and runs in polynomial time.*

3.5 A stronger structural property of optimal t-$\mathsf{SBPC_z}$ schedules

Although not necessary for the purposes of this work, we can actually prove the stronger property that every optimal schedule is, in fact, a normal schedule in which the trajectory of every robot contains exactly one leg. In view of Observations 1, 2, 3, and 4, in almost all cases, the modification of a non-normal schedule $\mathcal{S}$ to a normal schedule $\mathcal{S}'$ strictly decreases the completion time of at least one robot.

The only exception is in Lemma 5, when the trajectory of the robot consists of a quasi-regular excursion which is not regular, followed by a regular excursion in the opposite direction. Let us call a trajectory of this kind *unusual* and let us call a schedule *quasi-normal* if it satisfies Definition 6, except that robots may also have unusual trajectories.

If a schedule $\mathcal{S}$ is optimal and not quasi-normal, then, by the above observations, there is an optimal and normal schedule $\mathcal{S}'$ in which not all robots finish their trajectories simultaneously. This contradicts Corollary 1. Therefore, every optimal schedule is quasi-normal.

Furthermore, a quasi-normal schedule $\mathcal{S}$ that is not normal must have at least one robot with an unusual trajectory. The quasi-regular excursion of each such robot can be trivially transformed into a regular excursion, simply by having the robot search its assigned arcs on its way from its origin p to the farthest point of the excursion, instead of on its way back to p. Note that the resulting schedule $\mathcal{S}'$ has the same completion time as $\mathcal{S}$, and it has at least one robot whose trajectory contains two legs, since an unusual trajectory consists of two journeys.

We conclude that, if an optimal schedule is quasi-normal and not normal, then it can be transformed into an optimal and normal schedule in which at least one robot performs two legs. This contradicts Lemma 11. We have, therefore, the following more precise result:

Lemma 14. *Every optimal schedule is a normal schedule in which the trajectory of each robot contains exactly one leg.*

4 Multi-source beachcombers on the line

We now show that our results from the previous section also hold for the multi-source version of the beachcombers' problem on the line, in which robots are allowed to change direction as in t-$\mathsf{SBPC_z}$ (cf. Definition 3). Note that, in contrast to t-SBP (cf. Definition 4), the robots are allowed to change direction of movement and, in particular, to search segments on both sides of their respective starting points. We define the problem t-$\mathsf{SBPL_z}$:

Definition 10 **(t-$\mathsf{SBPL_z}$ – t-Source Beachcombers' Problem on the Line with zigzags).** *Consider a line segment $I_L = [0, L]$ and n robots $r_1, \ldots, r_n$, each robot r_i having searching speed s_i and walking speed $w_i > s_i$. For an integer $t \geq 1$, t-$\mathsf{SBPL_z}$ consists in (a) dividing the robots into at most t groups, (b) choosing a particular starting point on the segment for each group (the source of that group), and (c) finding an optimal correct searching schedule for I_L.*

Given a t-$\mathsf{SBPL_z}$ instance, it is convenient to imagine that the given segment of length L is bent into a loop so that its endpoints coincide, and additionally that an impassable barrier is placed at the point where the endpoints coincide. With this transformation, every correct t-$\mathsf{SBPL_z}$ schedule can be regarded as a correct t-$\mathsf{SBPC_z}$ schedule with the same completion time. We thus have the following analogue of Lemma 12:

Proposition 2. *For a fixed $t \geq 1$, let $\mathcal{I}$ be an instance of t-$\mathsf{SBPL_z}$ on a finite segment of length L and let $\mathcal{I}'$ be the corresponding instance of t-$\mathsf{SBPC_z}$ with the same set of robots on a cycle of circumference L. Every solution for $\mathcal{I}$ is also a solution for $\mathcal{I}'$ with equal cost.*

We say that a t-$\mathsf{SBPL_z}$ schedule is *normal* if its corresponding t-$\mathsf{SBPC_z}$ schedule for the cycle-with-barrier instance is normal according to Definition 6. In view of the following proposition, we can regard normal correct t-$\mathsf{SBPC_z}$ schedules as normal correct t-$\mathsf{SBPL_z}$ schedules with the same completion time:

Proposition 3. *Every normal correct t-$\mathsf{SBPC_z}$ schedule can be regarded as a normal correct t-$\mathsf{SBPL_z}$ schedule with the same completion time.*

Proof. It suffices to place the barrier on the cycle so that it does not affect the correctness of the given normal t-$\mathsf{SBPC_z}$ schedule $\mathcal{S}$. If $t \geq 2$, we place the barrier at one of the endpoints of the arc associated with one of the robot sources (cf. Definition 6, item 5). If $t = 1$, we place the barrier at the point where, during a clockwise traversal from the origin, we reach the end of the last arc that is searched in the clockwise direction (cf. Definition 6, item 6). With the barrier in place, $\mathcal{S}$ is a normal correct t-$\mathsf{SBPL_z}$ schedule. □

The existence of optimal and normal solutions for t-$\mathsf{SBPL_z}$ follows immediately from Theorem 2 and Propositions 2 and 3. Moreover, with Proposition 3, we can prove the following analogue of Lemma 13:

Proposition 4. *For a fixed $t \geq 1$, let $\mathcal{I}$ be an instance of t-$\mathsf{SBPC_z}$ on a cycle of circumference L and let $\mathcal{I}'$ be the corresponding instance of t-$\mathsf{SBPL_z}$ with the same set of robots on a finite segment of length L. Every optimal solution for $\mathcal{I}$ can be transformed in polynomial time to a solution for $\mathcal{I}'$ with equal cost.*

Proof. Let $\mathcal{S}$ be an optimal schedule for $\mathcal{I}$. By Lemma 8, there exists an optimal and normal schedule $\mathcal{S}'$ for $\mathcal{I}$ (which can be computed in polynomial time). By Proposition 3, $\mathcal{S}'$ can be regarded as a normal correct schedule for $\mathcal{I}'$, with the same completion time. □

Corollary 4. *For a fixed $t \geq 1$, let $\mathcal{I}$ be an instance of t-$\mathsf{SBPC_z}$ on a cycle of circumference L and let $\mathcal{I}'$ be the corresponding instance of t-$\mathsf{SBPL_z}$ with the same set of robots on a finite segment of length L. The optimal solutions for $\mathcal{I}$ and $\mathcal{I}'$ have the same cost.*

In view of Proposition 2 and 4 and Corollary 4, the results we obtained for t-$\mathsf{SBPC_z}$ in Section 3.4 hold also for t-$\mathsf{SBPL_z}$:

Theorem 7. 1-$\mathsf{SBPL_z}$ *is* NP*-hard, even when all robots have the same walking speed.*

Theorem 8. 1-$\mathsf{SBPL_z}$ *admits a* 0.5569*-approximation algorithm that runs in polynomial time.*

Theorem 9. t-$\mathsf{SBPL_z}$ *instances in which all robots have the same search speed can be solved optimally in polynomial time.*

Theorem 10. t-$\mathsf{SBPL_z}$ *admits a randomized algorithm which achieves an expected approximation ratio of* $1 - \left(1 - \frac{1}{2t}\right)^{2t}$*, needs* $\mathcal{O}(n \log t)$ *random bits, and runs in polynomial time.*

Finally, note that the equivalent of Lemma 14 holds for t-$\mathsf{SBPL_z}$ as well.

Lemma 15. *Every optimal t-$\mathsf{SBPL_z}$ schedule is a normal schedule in which the trajectory of each robot contains exactly one leg.*

Proof. Let $\mathcal{S}$ be an optimal t-$\mathsf{SBPL_z}$ schedule. By Proposition 2 and Corollary 4, $\mathcal{S}$ is also an optimal t-$\mathsf{SBPC_z}$ schedule. By Lemma 14, $\mathcal{S}$ is a normal t-$\mathsf{SBPC_z}$ schedule in which the trajectory of each robot contains exactly one leg. By Proposition 3, $\mathcal{S}$ can be regarded as a normal t-$\mathsf{SBPL_z}$ schedule. □

5 Concluding remarks

There are several directions in which the study of the search and exploration using two-speed robots may continue. An obvious one is to improve the approximation ratio for the versions of the problem that are NP-hard. In this respect, we should investigate whether zigzags may help to obtain approximate solutions, at least for particular combinations of searching and walking speeds of the robots (note that we know from the present paper that zigzags never help to obtain *optimal* solutions). Another direction is to study the configurations of robots' speeds and/or environments for which optimal solutions can be computed efficiently. Finally, it is worthwhile to consider different and more general search domains, such as non-simple closed or open curves.

Acknowledgements. We thank the anonymous referees for their careful reading and valuable comments, which have induced a significant improvement to the presentation of the paper.

Part of this work was done while Jurek Czyzowicz was visiting the LaBRI as a guest professor of the University of Bordeaux. This work was partially funded by the ANR project DISPLEXITY (ANR-11-BS02-014). This study has been carried out in the frame of "the Investments for the future" Programme IdEx Bordeaux – CPU (ANR-10-IDEX-03-02).

References

1. Albers, S., Henzinger, M.R.: Exploring unknown environments. SIAM J. Comput. 29(4): 1164–1188 (2000)
2. Albers, S.: Online algorithms: a survey. Math. Program. 97(1-2): 3–26 (2003)
3. Albers, S., Schmelzer, S.: Online algorithms - what is it worth to know the future? In: Algorithms Unplugged, pp. 361–366. Springer (2011)
4. Alpern, S., Gal, S.: The theory of search games and rendezvous. Kluwer Academic Publishers (2003)
5. Baeza-Yates, R.A., Culberson, J.C., Rawlins, G.J.E.: Searching in the plane. Information and Computation 106: 234–234 (1993)
6. Beauquier, J., Burman, J., Clement, J., Kutten, S.: On utilizing speed in networks of mobile agents. In: ACM SIGACT-SIGOPS 2010, pp. 305–314. ACM (2010)
7. Beck, A.: On the linear search problem. Israel Journal of Mathematics 2(4): 221–228 (1964)
8. Bellman, R.: An optimal search problem. Bull. Am. Math. Soc. p. 270 (1963)
9. Berman, P.: On-line searching and navigation. In: Fiat, A., Woeginger, G. (eds.) Online Algorithms The State of the Art, pp. 232–241. Springer (1998)
10. Chalopin, J., Flocchini, P., Mans, B., Santoro, N.: Network exploration by silent and oblivious robots. In: Graph-Theoretic Concepts in Computer Science, WG 2010, LNCS 6410, pp. 208–219. Springer (2010)
11. Chen, Y., Deng, X., Ji, Z., Liao, C.: The beachcombers' problem: Walking and searching from an inner point of a line. In: Language and Automata Theory and Applications, LATA 2016, LNCS 9618, pp. 270–282. Springer (2016)
12. Czyzowicz, J., Gasieniec, L., Georgiou, K., Kranakis, E., MacQuarrie, F.: The beachcombers' problem: Walking and searching with mobile robots. Theor. Comput. Sci. 608: 201-218 (2015)
13. Czyzowicz, J., Gasieniec, L., Georgiou, K., Kranakis, E., MacQuarrie, F.: The multi-source beachcombers' problem. In: Algorithms for Sensor Systems, ALGOSENSORS 2014, Revised Selected Papers, LNCS 8847, pp. 3–21. Springer (2014)
14. Czyzowicz, J., Gasieniec, L., Kosowski, A., Kranakis, E.: Boundary patrolling by mobile agents with distinct maximal speeds. In: Algorithms, ESA 2011, LNCS 6942, pp. 701–712. Springer (2011)
15. Czyzowicz, J., Ilcinkas, D., Labourel, A., Pelc, A.: Worst-case optimal exploration of terrains with obstacles. Inf. Comput. 225: 16–28 (2013)
16. Das, S., Flocchini, P., Kutten, S., Nayak, A., Santoro, N.: Map construction of unknown graphs by multiple agents. Theor. Comput. Sci. 385(1-3): 34–48 (2007)
17. Demaine, E.D., Fekete, S.P., Gal, S.: Online searching with turn cost. Theoretical Computer Science 361(2): 342–355 (2006)

18. Deng, X., Papadimitriou, C.H.: Exploring an unknown graph. In: Foundations of Computer Science, FOCS 1990, pp. 355–361. IEEE (1990)
19. Deng, X., Kameda, T., Papadimitriou, C.H.: How to learn an unknown environment (extended abstract). In: Foundations of Computer Science, FOCS 1991, pp. 298–303. IEEE (1991)
20. Dereniowski, D., Disser, Y., Kosowski, A., Pajak, D., Uznanski, P.: Fast collaborative graph exploration. Inf. Comput. 243: 37-49 (2015)
21. Fleischer, R., Kamphans, T., Klein, R., Langetepe, E., Trippen, G.: Competitive online approximation of the optimal search ratio. SIAM J. Comput. 38(3): 881–898 (2008)
22. Fomin, F.V., Thilikos, D.M.: An annotated bibliography on guaranteed graph searching. Theor. Comput. Sci. 399(3): 236–245 (2008)
23. Fraigniaud, P., Gasieniec, L., Kowalski, D.R., Pelc, A.: Collective tree exploration. Networks 48(3): 166–177 (2006)
24. Higashikawa, Y., Katoh, N., Langerman, S., Tanigawa, S.: Online graph exploration algorithms for cycles and trees by multiple searchers. J. Comb. Optim. 28(2): 480-495 (2014)
25. Kawamura, A., Kobayashi, Y.: Fence patrolling by mobile agents with distinct speeds. Distributed Computing 28(2): 147-154 (2015)
26. Wang, G., Irwin, M.J., Fu, H., Berman, P., Zhang, W., Porta, T.L.: Optimizing sensor movement planning for energy efficiency. ACM Transactions on Sensor Networks 7(4): 33 (2011)