ARTICLE IN PRESS

Theoretical Computer Science ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Optimal strategies for maintaining a chain of relays between an explorer and a base camp

Jarosław Kutyłowski [a], Friedhelm Meyer auf der Heide [b,*]

[a] International Graduate School of Dynamic Intelligent Systems, University of Paderborn, Germany
[b] Heinz Nixdorf Institute, University of Paderborn, Germany

ARTICLE INFO

Keywords:
Algorithms
Distributed
Network communications

ABSTRACT

We envision a scenario with robots moving on a terrain represented by a plane. A mobile robot, called *explorer* is connected by a communication chain to a stationary *base camp*. The chain is expected to pass communication messages between the explorer and the base camp. It is composed of simple, mobile robots, called *relays*.

We are investigating strategies for organizing and maintaining the chain, so that the number of relays employed is minimized and nevertheless the distance between neighbored relays in the chain remains bounded. We are looking for local and distributed strategies employed by restricted relays that have to base their decision ("Where should I go?") solely on the relative positions of its neighbors in the chain.

We present the *Manhattan–Hopper* and the *Hopper* strategy which improve the performance of all known solutions to this problem significantly. They are the first such strategies that are optimal in this setting, i.e., that allow the explorer to move with constant speed, independent of the length of the chain, and keep this length minimum up to a constant factor.

In this paper we will deal with the question on how to organize a chain of relays connecting two points on a plane. One of the points, the *base camp*, is stationary while the other is mobile and is called *explorer*. We have no influence on the movement of the explorer; the strategies we design control movement of the relays. We design strategies for the relays that keep the chain short and maintain connectivity despite movement of the explorer. The main challenge lies in the restriction that the relays have to decide locally, basing their decisions only on very limited information.

Up to now we have informally spoken of a *chain* of relays. Hereby we mean an ordered sequence of relays, denoted by $v_1, \ldots, v_n$. For technical reasons we include the explorer and the base camp into the chain, as respectively its first and last element; v_0 describes the explorer and v_{n+1} the base camp. Elements of the chain will be referred to as *stations*, meaning either a relay, the explorer or the base camp. The main restriction on the chain is that we require the distance between stations neighbored in the chain not to exceed the transmission distance of 1. Thanks to this, the chain can forward communication messages between the base camp and the explorer.

We want to design strategies for the relays which guide their movement on the plane. Each of the relays is assumed to execute its own copy of this strategy and is able to sense the environment. We will go deeper into detail on the computational and sensing abilities of relays when describing the relay model later.

The greatest challenge for this problem is that we are looking for strategies which operate locally and with very simple logic. Therefore, we have to design a solution in which each relay computes its position on its own, where communication is not used at all, and relays possess no memory. We are interested to see whether one can design efficient strategies solving this problem within this simple model.

* Corresponding author. Tel.: +49 5251 606480.
E-mail addresses: jaroslaw@kutylowski.de (J. Kutyłowski), fmadh@uni-paderborn.de (F. Meyer auf der Heide).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.04.010

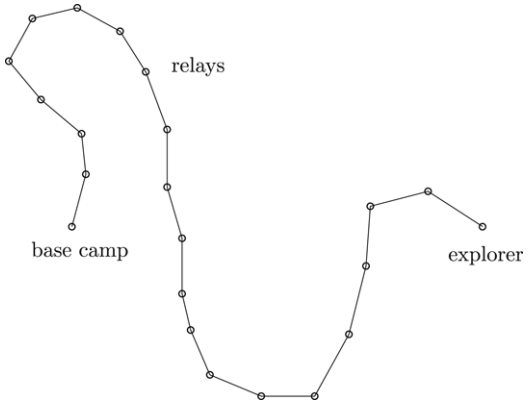


Fig. 1. A long winding chain.

Time model

We divide time into discrete time steps. The time steps are synchronized among all stations, i.e. their duration and starting/ending times are the same for all stations.

We require the execution of a strategy in a time step to fit into the *LCM*-model (as in [7]). In this model, the execution of each step is divided into three operations. These operations are: *Look*, *Compute*, *Move*. In the first operation of the step the relay gathers new information about its environment by scanning it with its sensors. In the second operation, the sensoric input is analyzed and a decision is made about the behavior of the robot within the current step. During the *Move* operation, the robot moves to a position precomputed in the *Compute* operation.

Examining strategies in the *LCM*-model implicitly assumes that one regards a *LCM*-step as an atomic time unit and determines the performance of strategies based on the number of such steps necessary to reach a certain condition. This implies that a *Look*-operation accounts for a comparable amount of time as a *Move*-operation. In order to keep this assumption realistic, we will be restricting the maximum distance a relay can move during one time step. On the other hand, in contrast to our intuition, the *Move* operation is not necessarily the one which takes the longest time. Very often, gathering the sensoric input is even more time-consuming, if for example laser scans of the surrounding must be taken. An example of such a scenario can be found in [12].

Relay model

The relays are required to be able to execute sequential runs of our strategies. A run encompasses the execution of the strategy by all relays in the chain, such that v_i executes the strategy only after v_{i-1} has finished its execution of the given run. Therefore, relays have to be able to determine that their predecessor has finished executing the strategy to start their own action.

We assume that the robots are oblivious and base their decisions in a time step only on the current sensory input. The sensor input is very restricted: v_i only sees v_{i-1}, v_{i+1}, i.e. v_i knows the distances to v_{i-1} and v_{i+1} and the angle at v_i formed by the line segments $\langle v_{i-1}, v_i\rangle$ and $\langle v_i, v_{i+1}\rangle$. We assume that measurements are precise.

Performance measures

Analysis of the strategies shown in this paper will be performed in two settings: either static or dynamic. In the static case we will be given a chain of relays at the beginning. This chain can be very winding (as shown in Fig. 1) and therefore significantly longer than the distance between the explorer and the base camp. We even assume that the chain may be self-intersecting and spiral around the base camp, as far as a relay is still able to distinguish its chain neighbors from relays comprising other parts of the chain.

While the explorer won't be moving, we will investigate the time needed for relays to become arranged close to the line segment between the base camp and the explorer, approximately minimizing the length of the chain. The performance of a strategy in this scenario is given by the number of time steps it requires for all relays to get appropriately close to their optimal positions on the line segment, starting with a worst-case chain arrangement.

In the dynamic scenario, we will start with some well-defined chain configurations and let the explorer move. We will then examine the performance of the strategy in dealing with the explorer's movements. Although the dynamic scenario is of more importance for practical reasons, results from the static scenario can be used for comparing the performance between different strategies and in order to understand weaknesses of the strategies.

Within the dynamic scenario we are looking at the guaranteed speed the explorer is able to move, so that the strategy is still able to catch up with its movement without disconnecting the chain, no matter how the actual movement pattern of

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

the explorer is. Then, the performance of a strategy is measured as a ratio of this guaranteed speed to the speed the relays are allowed to move.

Related work

The problem of maintaining a chain of relays between a base camp and an explorer has been already discussed in literature. In [16,17] it has been investigated from the engineering point of view, providing results on a an experimental scenario with real robots. In [10] we show the most intuitive strategy for controlling the movement of relays and investigate its performance on a plane. This strategy, called *Go-To-The-Middle*, moves each each relay directly into the middle of the line segment between its chain neighbors in each time step. While the strategy is very simple, local and requires no special abilities from the relays, it sacrifices performance. In the static case mentioned earlier, the strategy requires $\Omega(n^2)$ time steps in order to bring the chain length near to the distance between of the explorer and base camp, assuming that n is the number of relays in the chain at the beginning. In the dynamic case, relays are only able to maintain the chain connected if the explorer is moving with a speed inversely proportional to its distance to the base camp. In other words, the explorer has to decrease its speed if moving further away from the base camp. Another disadvantage of the *Go-To-The-Middle* strategy is that it requires the explorer to know the distance to the base camp in order to control the number of relays in the chain.

In [11] we show a strategy which takes advantage of a more powerful relay model. Here we equip the relays with the ability to align their local coordinate systems with respect to a global coordinate systems using a GPS-like positioning system. Allowing the relays to store coordinates of the base camp in their coordinate system, the *Chase-Explorer* strategy is able to maintain a chain of nearly optimal length in the dynamic case without restricting the speed of the explorer. The requirement on the relays to posses a GPS positioning scheme can be relaxed by giving the relays the ability to align their local coordinate systems to the global one using a compass and communication among the relays. We furthermore show how the *Chase-Explorer* strategy can be applied on a terrain with obstacles.

A wide variety of problems similar to the maintenance of a communication chain have been investigated in the area of swarm robotics. We will shortly survey results which deal with ordering robots to form some geometric figures on the plane.

One of the basic problems investigated in this setting is the gathering problem [15]. A dispersed group of robots, represented by points on a plane, is required to gather at a point in the plane. This point can be chosen freely by the robots, it is only important that all robots finally gather at one point. It turns out that in an asynchronous model, in which the duration and the starting/ending of time steps may vary between robots, it is impossible for deterministic robots to gather at one point in a finite time [5]. Allowing more than 5 robots to detect whether multiple robots occupy one point allows gathering at one point in finite time [3]. The problem may be also considered in a setting with robot failures [2]. The problem turns out to be slightly easier in the semi-synchronous model, in which time steps are synchronized among robots but robots may be put to sleep during any time step. Gathering is then possible with 3 robots in finite time [19]. Recently the problem has been evaluated in an asynchronous setting with robots equipped with imprecise compasses [14].

The convergence problem is an easier variant of the gathering problem, in that we only require all robots to converge in infinite time to one point. The gravitational algorithm which allows robots to move towards their center of gravity converges in synchronous and semi-synchronous models [7] and also in an asynchronous model [8]. The problem can also be solved in a semi-synchronous model when the sensoric input and the odometry of robots is imprecise [6]. A different model with robots having only limited visibility of the plane has been shown to be solvable in [1].

The problem of pattern formation is slightly more involved, as robots are required to form a pattern on a plane, rather than only moving to one point. Asynchronous robots can form in advance any known pattern if they know the orientation of two axes, while it can be only solved for an odd number of robots when only the orientation of one axis is known [13]. Certain patterns can be formed even without knowing a common orientation, depending on the starting positions of the robots [18]. In [9] and [4] algorithms for forming a circle by robots in the semi-synchronous model are evaluated.

Outline

For a better understanding of the concepts underlying the *Hopper* strategy, we will first discuss the *Manhattan–Hopper* strategy, which is very similar to *Hopper* but works with a slightly simplified model. In that model we assume all stations to move on a grid and not on a continuous plane.

We start our analysis of the *Manhattan–Hopper* strategy in Section 2 by analyzing its performance in a static scenario. We will show, that the *Manhattan–Hopper* strategy reduces the chain to its optimal length in $4n$ steps, assuming that we start with a chain with n relays.

Afterwards, in Section 3, we establish the performance of the *Manhattan–Hopper* strategy in a dynamic scenario, with the explorer moving. We will show that the *Manhattan–Hopper* strategy allows the explorer to move with a speed of $1/2$ per round, in comparison to relays which have to move with a speed of $\frac{1}{2}(1+\sqrt{2})$ per time step, where a round consists of 6 time steps. With this restriction on movement speed of the explorer, and assuming that the explorer moves on a discrete grid only, we maintain a chain with at most $3d_t+2$ relays in time step t. Here d_t denotes the distance between the explorer and the base camp in time step t.

We eventually present the *Hopper* strategy in Section 4 and perform an analysis of its performance in the static scenario in Section 5. Although the *Hopper* strategy works in a continuous plane, it requires only $8n+1$ time steps to reduce the chain

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

length to use at most $2\sqrt{2}d + 1$ relays, where d is the distance between the explorer and the base camp. This implies that the *Hopper* strategy, while using the same restrictions of the relays, is by $\Omega(n)$ better than the *Go-To-The-Middle* strategy described in [10].

The analysis of the *Hopper* strategy is extended in Section 6 to a dynamic case. We show that the explorer is still able to move with a speed only by a constant factor slower than the speed of the relays. In addition, the length of the chain is guaranteed to be linear in the distance between the explorer and the base camp during the whole strategy.

The following table summarizes the performance of the strategies described in this paper and the strategies *Go-To-The-Middle* and *Chase-Explorer*. Recall that in the static scenario, we define performance as the time necessary to bring the chain near to an optimal length, starting with a chain of n relays. In the dynamic scenario we are comparing the guaranteed speed of the explorer, compared to the the maximum allowed speed for the relays. In this case n is a value which corresponds to the number of relays in the chain at a given point of time, which is always proportional to the distance of the explorer to the base camp with the strategies investigated.

Strategy	Static scenario	Dynamic scenario
Go-To-The-Middle	$\mathcal{O}(n^2 \log n)$	$\mathcal{O}(1/n)$
Chase-Explorer	$\mathcal{R}n$	$\mathcal{O}(1)$
Manhattan–Hopper	$4n$	$1 + \sqrt{2}$
Hopper	$25n + 1$	$\mathcal{O}(1)$

Note that in the static scenario, a relay has potentially to move for a distance $\Omega(n)$ in order to get to its destination on the line segment between the explorer and the base camp. Therefore both the *Manhattan–Hopper* and *Hopper* strategies introduced in this paper have a performance which is optimal up to constant factors in the static scenario. The same applies to the dynamic scenario, where it is obvious that there exists no strategy with the ratio of explorer's speed to the relays' speed greater than 1.

Notation

Recall that we denote stations in the chain by $v_0, \ldots, v_{k+1}$, where v_0 denotes the explorer and v_{k+1} the base camp. The distance between the explorer v_0 and the base camp v_{k+1} in a time step t is denoted by d_t. In static analysis, where this distance is constant, we denote it by d.

We will denote a line segment defined by two points p, q on the plane by $\langle p, q\rangle$. The notation $|x|$ denotes the absolute value of x, whereas $|p - q|_2$ is the Euclidean distance between points p and q.

Spatial vectors as used in this paper have a magnitude and a direction and are represented by a pair of coordinates $[x, y]$. A vector has no fixed initial and terminating point, although it can be used to define the relation between two points on the plane. Two vectors $\vec{u} = [x_u, y_u]$, $\vec{a} = [x_a, y_a]$ can be added so that $\vec{u} + \vec{a} = [x_u + x_a, y_u + y_a]$.

1. The Manhattan–Hopper strategy

For the *Manhattan–Hopper* strategy to operate properly, all stations have to assume positions on grid points. The mesh size of the grid is fixed to $\frac{1}{2}$. We assume that in the first time step, the positions of all stations are on grid points and that the explorer may only move between grid points. Our strategy will only move relays between grid points, so that all stations are on grid points in all time steps.

Strategy description

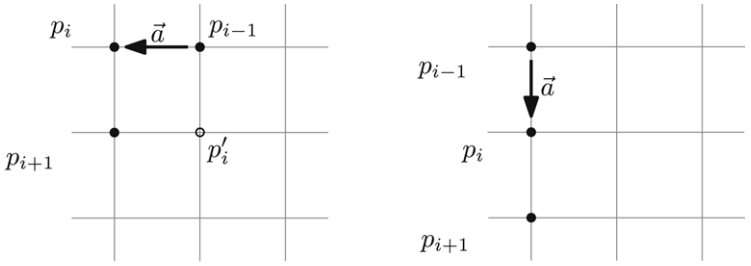
The *Manhattan–Hopper* strategy is executed in *runs*. In each run, a relay v_i executes its action only after relay v_{i-1} has finished its own. We will see later that such runs can be pipelined, so that a new run can be started every three steps. For the sake of analysis, it is sufficient and more comfortable to assume that a new run is executed only after the previous one has finished. We will adhere to that assumption for the rest of the chapter.

We say that the chain is in *proper condition* if each two neighboring stations in the chain are positioned in neighboring points of the grid. For this discrete scenario it is natural to measure the length of the chain in terms of its manhattan length. We say that the chain has optimal length if its length is equal to the manhattan distance between the explorer and base camp.

Let us first describe the *Manhattan–Hopper* strategy for setting the immobile explorer. We will extend it to accommodate the explorer's movement in Section 3. As already mentioned, the *Manhattan–Hopper* strategy is executed in runs. For the sake of our description, let us fix a specific run r. Let p_i denote the position of v_i at the beginning of this run, while p'_i is its position after it has executed the *Manhattan–Hopper* strategy.

In a run, the relay v_i moves only after v_{i-1} has finished its movement and prior to the movement of v_{i+1}. Therefore, the positions of v_i's neighbors are p'_{i-1} and p_{i+1} when v_i decides on its action. Typically, the relay v_i executes the *hop-operation*, defined as follows. Let $\vec{a}$ be a vector such that $p'_{i-1} + \vec{a} = p_i$ as shown in Fig. 2. Then the hop-operation executed by relay v_i moves it to position p'_i, such that $p'_i = p_{i+1} - \vec{a}$. This implies that, if v_{i-1}, v_i, v_{i+1} lie in-line, then station v_i does not move. This

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010



(a) Stations not in line, v_i moves

(b) Stations in line, v_i does not move

Fig. 2. Hop-operation in *Manhattan–Hopper* strategy.

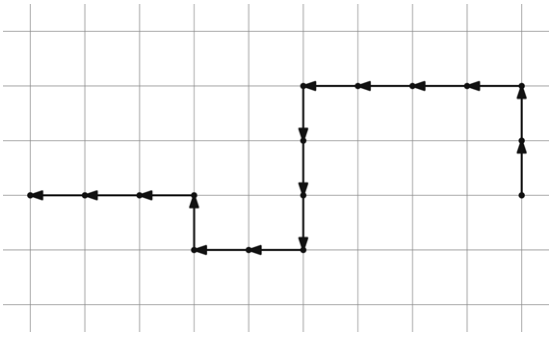


Fig. 3. Chain before executing the *Manhattan–Hopper* strategy.

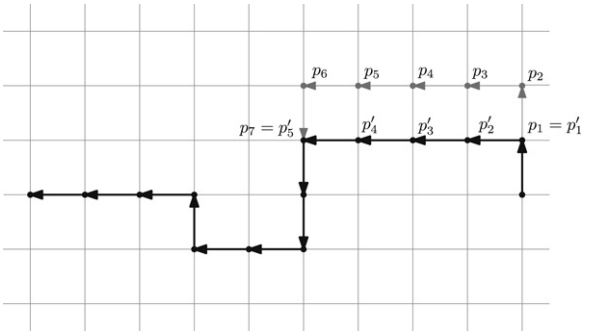


Fig. 4. Chain after the execution of a run of the *Manhattan–Hopper* strategy. Relays v_6 and v_7 are removed.

situation is depicted in Fig. 2(b). The provided definition of the hop-operation is equivalent to mirroring v_i's position w.r.t. the line through p'_{i-1} and p_{i+1}.

A special case occurs when relay v_i moves to a point on the grid occupied by v_{i+2}. Then a *remove*-operation is executed, i.e., v_{i+1} and v_{i+2} are removed from the chain and the run stops. After this removal, we index the chain once again, so that the relays are described by $v_1, \ldots, v_{n_{k'}}$. This is shown on Figs. 3 and 4 where relays v_6 and v_7 are removed as $p'_5 = p_7$.

Thus, a run of the *Manhattan–Hopper* strategy either removes two relays from the chain, or is executed by all relays. Removing relays in this way is the only way how the *Manhattan–Hopper* strategy shortens the length of the chain. In particular, if v_i moves to a point already occupied by a relay v_j such that $j \neq i+2$, no relays are removed and the execution of the run proceeds as earlier. This may sound counterproductive as the chain part forming a loop between v_i and v_j might be removed. In our analysis, we assume that such loops are not removed, because this would require information exchange among the relays, those in the loop have to get to know that they are obsolete. It is easy to see that the same results as we show also hold in the stronger model, where long loops may be removed.

A reasonable performance of the *Manhattan–Hopper* strategy can only be attained if the runs are not executed one after another but pipelined in a fast manner. This is possible without problems due to the sequential nature of runs of the *Manhattan–Hopper* strategy. Run r starting execution in time step t may be followed by run $r+1$ starting execution in time step $t+3$. Thus, the execution of m runs starting with a chain with n relays takes at most $n+3m$ time steps.

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

2. Performance of Manhattan–Hopper in the static scenario

We first investigate the performance of the *Manhattan–Hopper* strategy in the static case and eventually show the following theorem.

Theorem 1. *Starting with a chain in a proper condition with n relays, the Manhattan–Hopper strategy ensures that*

(a) *the chain remains in proper condition,*
(b) *the relays move at most distance* $\frac{1}{2}\sqrt{2}$ *per round, i.e., every third step, and*
(c) *after n runs, i.e., 4n steps, the chain has optimal length.*

Property (b) follows directly from the definition of the strategy. Also property (a) is easy to see: If a run does not execute a removal, then the manhattan distances between neighbors do not change. Whenever stations v_{i+1} and v_{i+2} are removed from the chain, relay v_i is on the position of v_{i+2} and therefore has manhattan distance $\frac{1}{2}$ to v_{i+3}.

The remaining part of this section will be devoted to showing property (c), by showing that n runs suffice to reduce the chain to its optimal length. As we can start a new run every 3 steps, this implies (c).

Let d denote the (manhattan) distance between the explorer and the base camp. Fix some run and let p_i and p'_i denote the positions of the station v_i before and after this run, respectively.

We can describe the positions of the relays with respect to the explorer by using the spatial unit vectors $\vec{e}_0 = \left[\frac{1}{2}, 0\right]$, $\vec{e}_1 = \left[-\frac{1}{2}, 0\right]$, $\vec{e}_2 = \left[0, \frac{1}{2}\right]$, $\vec{e}_3 = \left[0, -\frac{1}{2}\right]$. $\vec{e}_0$ and $\vec{e}_1$ are both horizontal vectors, with opposite directions. Analogously, $\vec{e}_2$ and $\vec{e}_3$ are vertical. We will call such pairs of vectors *oppositional*. For our run, we can define a sequence of spatial unit vectors $C = (\vec{u}_0, \vec{u}_1, \ldots, \vec{u}_k)$ such that $p_i = p_0 + \vec{u}_0 + \cdots + \vec{u}_{i-1}$. This sequence will be called *configuration* and it describes the positions of all relays in the chain in relation to the explorer by using unit vectors.

We want to describe the behavior of *Manhattan–Hopper* in terms of operations on configurations. The following two lemmas describe how one run of the *Manhattan–Hopper* strategy can modify the configuration.

Lemma 2. *Let* $C = (\vec{u}_0, \vec{u}_1, \ldots, \vec{u}_k)$ *be the configuration at the beginning of the run. Assume the run finishes without removing any relays from the chain. Then the configuration at the beginning of the next run is* $C' = (\vec{u}_1, \ldots, \vec{u}_k, \vec{u}_0)$. *Furthermore, for each* $j = 0, \ldots, k$, *the vectors* $\vec{u}_0$ *and* $\vec{u}_j$ *are not oppositional.*

Proof. We will show that the invariant $p'_i = p_{i+1} - \vec{u}_0$ holds for all $i = 0, \ldots, k$ by induction. The induction basis is clearly given, as we have by definition of vector $\vec{u}_0$ that $p'_0 + \vec{u}_0 = p_0 + \vec{u}_0 = p_1$.

Assume now that the invariant holds for all $i \leq j$. Then we have $p'_j = p_{j+1} - \vec{u}_0$. Consider the hop-operation performed by relay v_{j+1}. As the position p'_{j+1} depends on the vector between p'_j and p_{j+1} and the position of v_{j+2} we have from the definition of the hop-operation $p'_{j+1} = p_{j+2} - \vec{u}_0$. This proves the invariant.

The positions of all relays v_i for $i = 1, \ldots, k$ at the beginning of the run are, by the definition of the configurations, given by

$$p_i = p_0 + \vec{u}_0 + \cdots + \vec{u}_{i-1}.$$

After the run we have by our invariant for all relays

$$p'_i = p_{i+1} - \vec{u}_0 = \vec{u}_1 + \cdots + \vec{u}_i.$$

Obviously the base camp does not move and therefore we have $p'_{k+1} = p_{k+1}$. Thus,

$$p'_{k+1} = p_k + \vec{u}_0 = \vec{u}_1 + \cdots + \vec{u}_k + \vec{u}_0.$$

The final configuration is thus given by $C_{r+1} = (\vec{u}_1, \ldots, \vec{u}_k, \vec{u}_0)$.

We will proceed with proving the second statement of the lemma. Assume for the sake of contradiction that there exists a vector $\vec{u}_j$ which is oppositional to $\vec{u}_0$, i.e. $\vec{u}_j = -\vec{u}_0$. By our earlier observation we have $p'_{j-1} = p_j - \vec{u}_0$ after the run strategy has been executed by v_j. Therefore $p'_{j-1} = p_j - \vec{u}_0 = p_j + \vec{u}_j = p_{j+1}$, as by its definition vector $\vec{u}_j$ connects v_j to v_{j+1}. In that case, the run would remove relays v_j and v_{j+1} from the chain, contradicting the assumptions of our lemma. □

Lemma 3. *Let* $C = (\vec{u}_0, \vec{u}_1, \ldots, \vec{u}_k)$ *be the configuration at the beginning of the run. The run finishes with removing* v_{i+1} *and* v_{i+2} *from the chain if and only if the vector* $\vec{u}_{i+1}$ *is the first vector in C oppositional to* $\vec{u}_0$. *The configuration at the beginning of the next run is then given by* $C' = (\vec{u}_1, \ldots, \vec{u}_i, \vec{u}_{i+2}, \ldots, \vec{u}_k)$.

Proof. We will first show that, if v_{i+1} and v_{i+2} are removed from the chain, then $\vec{u}_{i+1}$ is oppositional to $\vec{u}_0$. Since all relays prior to v_i execute the hop-operation without removing any relays from the chain, the execution of the run for these relays is the same as the execution described in Lemma 2. Therefore, for all $j \leq i$ it holds that

$$p'_j = \vec{u}_1 + \cdots + \vec{u}_j.$$

As v_{i+1} and v_{i+2} have been removed, it must hold $p'_i = p_{i+2}$. Therefore

$$p'_i = \vec{u}_1 + \cdots + \vec{u}_i = \vec{u}_0 + \vec{u}_1 + \cdots + \vec{u}_{i+1} = p_{i+2},$$

so that $\vec{u}_0 = -\vec{u}_{i+1}$.

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

For the proof in the other direction assume that $\vec{u}_{i+1}$ is the first vector in C oppositional to $\vec{u}_0$. By the previous argument, no relay pair v_{j+1}, v_{j+2} with $j < i$ can be removed, as then $\vec{u}_{j+1}$ would be oppositional to $\vec{u}_0$. Therefore all relays v_j for $j \leq i$ have executed the hop-operation, so that

$$p'_i = \vec{u}_1 + \cdots + \vec{u}_i.$$

As $\vec{u}_0 = -\vec{u}_{i+1}$ it holds

$$p'_i = \vec{u}_1 + \cdots + \vec{u}_i = \vec{u}_0 + \vec{u}_1 + \cdots + \vec{u}_{i+1} = p_{i+2},$$

and v_{i+1} and v_{i+2} are removed by the *Manhattan–Hopper* strategy.

Assume now that v_{i+1} and v_{i+2} are removed from the chain in the current run. From our previous reasoning we have for any $j \leq i$

$$p'_j = \vec{u}_1 + \cdots + \vec{u}_j$$

and for all $j \geq i + 3$

$$p'_j = \vec{u}_1 + \cdots + \vec{u}_i + \vec{u}_{i+2} + \cdots + \vec{u}_{j-1}.$$

The configuration after the completed run is therefore given by

$$C_{r+1} = (\vec{u}_1, \ldots, \vec{u}_i, \vec{u}_{i+2}, \ldots, \vec{u}_k). \quad \square$$

Let $C_1 = (\vec{a}_0, \ldots, \vec{a}_k)$ be the configuration describing the chain at the beginning of the first run. Each of the vectors $\vec{a}_0, \ldots, \vec{a}_k$ is of the type of one of the unit vectors, nevertheless we can keep track of each of them by assigning them unique identification numbers. The two former lemmas show that each run of *Manhattan–Hopper* either removes vectors from this configuration or the configuration is shifted cyclically. Particularly, no vector is added to the configuration by the *Manhattan–Hopper* strategy. Therefore, we can describe each configuration occurring in the future as some arrangement of a subset of the vectors $\vec{a}_0, \ldots, \vec{a}_k$.

Thus, we can follow the trace of each $\vec{a}_j$, whose initial position in the start configuration is j. Assume that, in run r, it has position l. Then it is either removed (by Lemma 3), or its position is reduced by 1 (in case of the shift in Lemma 2 or in case of Lemma 3, if the removed relay is v_i with $i > l$) or its position is reduced by 2 (in case of Lemma 2, if the removed relay is v_i with $i < l$).

Now assume that, after n runs, there still exists an oppositional pair of vectors $\vec{u}_p, \vec{u}_q$, with $p < q$. Then, by the above, at most p runs earlier, $\vec{u}_p$ was at position 0. As also $\vec{u}_q$ is in this configuration, $\vec{u}_p$ will be removed in the next run by Lemma 3, contradicting the assumption that $\vec{u}_p$ is part of the configuration after n runs.

Thus, there are no oppositional vectors in the configuration after n runs. This implies that the length of the path after n runs is minimal, namely the manhattan distance between the explorer and the base camp. This proves part (c) of the theorem.

3. Performance of Manhattan–Hopper in the dynamic scenario

In order to adapt *Manhattan–Hopper* to work in a dynamic scenario, we have to deal with the movement of the explorer. For that purpose, we allow the explorer to move only to a grid point adjacent to its current position at once. The *Dynamic-Manhattan–Hopper* strategy groups the movement of the explorer and the execution of two runs into one round. The first run is a *follow*-run in which relays execute the *follow*-operation. This allows them to catch up with the movement of the explorer. After a run of the *Manhattan–Hopper* strategy, a so called *hopper*-run is executed. At this point of time we assume that the two runs are executed one after another and the round only finishes when the runs are ready. We will elaborate more on the pipelining of the runs later.

The follow-operation is applied by relay v_i whenever v_{i-1} increases its distance to v_i to 1. Then relay v_i moves to the old position of the station v_{i-1}, i.e. $p'_i = p_{i-1}$. Thereby, the chain shifts itself in the direction of the explorer by one grid point, so that finally the last relay is in manhattan distance 1 to the base camp. At this point of time, the base camp inserts a new relay at the old position p_k of the relay v_k. During execution of the follow-operation, the maximum distance between neighbored stations is 1 and therefore the chain remains connected. If the chain was in a proper condition before the follow-run, it is so afterwards, too.

While the follow-run ensures that the manhattan distance between two relays is exactly $\frac{1}{2}$, the hopper-run is meant to decrease the length of the chain if necessary. We first show that these two runs between every movement of the explorer are sufficient to maintain an optimal chain length.

Lemma 4. *Let the chain have optimal length prior to the explorer's movement. Then after the explorer's movement the follow-run and the hopper-run bring the chain to an optimal length.*

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

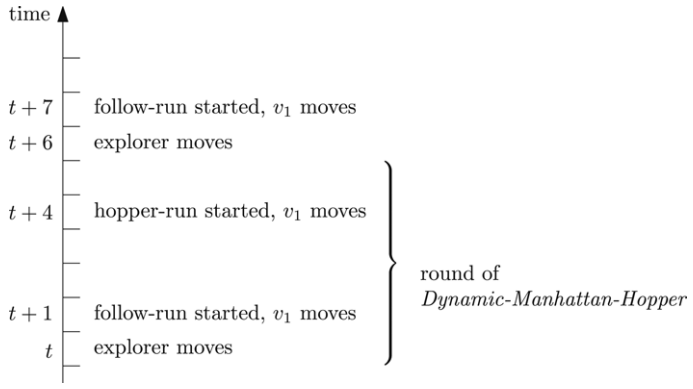


Fig. 5. One round of the execution of *Dynamic-Manhattan–Hopper*.

Proof. Let $C = (\vec{u}_0, \ldots, \vec{u}_k)$ be the configuration of the chain prior to the explorer's movement. Let the explorer move by $\vec{u}$, i.e. $p'_0 = p_0 + \vec{u}$. Then, after the explorer's movement and the first run of the *Manhattan–Hopper* strategy we have $C' = (\vec{u}, \vec{u}_0, \ldots, \vec{u}_k)$. The manhattan length of the chain defined by C' increases with respect to C by $\frac{1}{2}$.

Recall that we are assuming that the length of the chain defined by C is optimal before the explorer's movement. When moving the explorer by $\vec{u}$ increases its manhattan distance to the base camp, then the length of the chain after applying the follow-run is already optimal and the hopper-run is actually unnecessary.

In case the explorer's movement decreases its manhattan distance to the base camp, this decrease is equal to $\frac{1}{2}$. Then, the follow-run increases the chain's length additionally by $\frac{1}{2}$. We have to show that the hopper-run reduces the length of the chain by the same amount. Observe that, since the length of the chain described by C' is not optimal, there must exist a pair of oppositional vectors in C'. On the other hand, there was no such pair in C. This implies, that there exists a vector $\vec{u}_j \in C$ which is oppositional to $\vec{u}$. By Lemma 3 the existence of a vector oppositional to the first vector of the configuration leads to a hopper-run which reduces the manhattan length of the chain by 1. □

Pipelined execution

The runs of the *Dynamic-Manhattan–Hopper* strategy are executed in the discrete, synchronized time model. In order to pipeline them, we have to make sure that consecutive runs do not interfere. This is guaranteed if the follow-run starts directly after movement of the explorer, but two idle steps are inserted before the hopper-run and between the hopper run and the next movement, see Fig. 5. Thus the explorer may move every 6 time steps. Such a sequence of 6 time steps starting with a movement of the explorer is called a *round*.

Let d_r denote the distance between the explorer and the base camp at the beginning of round r. We are now ready to show the following theorem.

Theorem 5. *Assume we start with an optimal chain. Then, the chain maintained by the Dynamic-Manhattan–Hopper strategy has the following properties before each round r.*

(a) *The chain remains connected,*
(b) *the explorer may move a distance of* $\frac{1}{2}$ *every round, i.e. every* 6*th time step,*
(c) *relays move at most distance* $\frac{1}{2}(1+\sqrt{2})$ *per round, and*
(d) *the number of relays used in the chain is at most* $3d_r + 2$.

Proof. Property (a) follows easily from the fact that the follow-run decreases the distances between neighbored relays to $\frac{1}{2}$. During a round, a relay executes the follow-run, moving for at most $\frac{1}{2}$ and a hopper-run, moving for at most $\frac{1}{\sqrt{2}}$. Property (c) follows. The rest of the proof is devoted to showing property (d).

In case the runs are pipelined, the decrease of chain length corresponding to movement of the explorer is delayed until the run started after the movement is finished. In the worst case this can take as many time steps as relays have been in the chain at the moment the explorer moved. This can mean that the distance of the explorer to the base camp decreases during some timespan, while the chain is still executing various runs which will eventually decrease its length. Therefore, we have to investigate how large the difference between chain length and the explorer's distance may become.

Assume the chain has optimal length at the beginning of time step 1. Runs of the *Manhattan–Hopper* strategy are pipelined with an appropriate time separation and thus there is no difference for run $r+1$ whether it is started 3 time steps after the start of run r or after a full completion of run r. Therefore and by Lemma 4 a hopper-run started in round r works on a chain of length at most d_r, having $2d_r$ relays. Thus, the hopper-run lasts for at most $2d_r + 2$ timesteps. Let us fix round r. The number of relays in the chain at the beginning of round r is bounded from above by $2d_r$ plus the number of unfinished hopper-runs times 2. Therefore, we will look at how many hopper-runs are still unfinished at the beginning of round r. Take a round

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

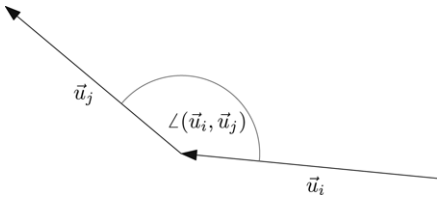


Fig. 6. Angle between vectors $\vec{u}_i$ and $\vec{u}_j$.

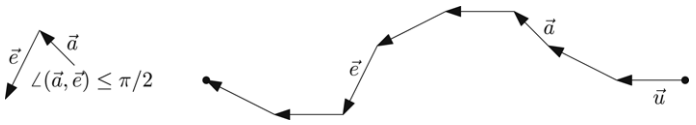


Fig. 7. Vector $\vec{u}$ is non-conflicting, while $\vec{a}$ and $\vec{e}$ have a conflict.

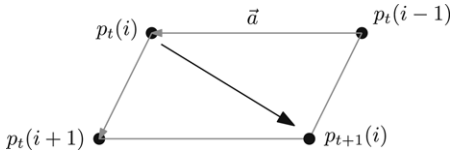


Fig. 8. Relay v_i executing the hop-operation in time step t.

$z < r - \frac{d_r+1}{2}$. The hopper-run started in round z will last for at most $2d_z + 2$ time steps and thereby will finish not later than at round $z + \frac{2d_z+2}{6}$. As during the rounds between z and r the explorer might have moved for a distance of at most $(r - z)/2$, we have $d_z \leq d_r + (r - z)/2$. Then,

$$z + \frac{d_z + 1}{3} \leq z - \frac{z}{6} + \frac{r}{6} + \frac{d_r + 1}{3} < r - \frac{5}{12}(d_r + 1) + \frac{1}{3}(d_r + 1) \leq r.$$

Therefore a hopper-run started in round z is finished before round r. There are at most $\frac{d_r+1}{2}$ runs still unfinished in round r and thereby the number of relays in the chain is at most $2d_r + (d_r + 1)$. This proves last property (d). □

4. Hopper strategy

The *Hopper* strategy is an extension of the *Manhattan–Hopper* strategy, designed in such a way that it does not require stations to be positioned on a discrete grid. We still require relays to have precise local coordinate systems, nevertheless we do not use any fixed grid structure in order to organize the relays. Furthermore, we allow the explorer to move freely, without being restricted to grid points.

As for the *Manhattan–Hopper* strategy, we will describe the chain at the beginning of a run by the configuration $C = (\vec{u}_0, \ldots, \vec{u}_k)$. Note that, since stations are not necessarily aligned to grid points, the vectors $\vec{u}_i$ are no longer restricted to unit vectors as in the *Manhattan–Hopper* strategy. For two vectors $\vec{u}_i$ and $\vec{u}_j$ we define the angle $\angle(\vec{u}_i, \vec{u}_j)$ as the smaller of the two angles created by anchoring $\vec{u}_j$ at the terminal point of $\vec{u}_i$, as shown on Fig. 6. Note that the $\angle$ function is symmetric, i.e. $\angle(\vec{u}_i, \vec{u}_j) = \angle(\vec{u}_j, \vec{u}_i)$ and thereby $\angle(\vec{u}_i, \vec{u}_j) \leq \pi$.

We will say that a vector $\vec{u}$ has a conflict with a vector $\vec{a}$ if $\angle(\vec{u}, \vec{a}) \leq \pi/2$. A vector which has an angle larger than $\pi/2$ to all vectors from a configuration is called non-conflicting. The concept is depicted in Fig. 7, where vector $\vec{u}$ is non-conflicting and vectors $\vec{a}$ and $\vec{e}$ are clearly conflicting as the angle $\angle(\vec{a}, \vec{e}) \leq \pi/2$.

The *Hopper* strategy is executed, exactly as *Manhattan–Hopper*, in sequential runs. Once again, let us fix a run. Then p_i and p'_i will denote the positions of the station v_i before and after this run, respectively.

The behavior of relay v_i in the run depends on whether the distance between p'_{i-1} and p_{i+1} is larger than 1 or not. If the distance is smaller or equal to 1 then obviously v_i is no longer necessary in the chain. In this case the relay removes itself, hereby executing the *remove-operation*. Executing the remove-operation finishes the run of the *Hopper* strategy.

In case the distance between p'_{i-1} and p_{i+1} is larger than 1, the action of v_i depends on the angle between the line segments $\langle p'_{i-1}, p_i\rangle$ and $\langle p_i, p_{i+1}\rangle$. The hop-operation is invoked if the angle is larger than $\pi/2$, otherwise the shorten-operation is used.

Hop-operation. The hop-operation is exactly the same as used previously in *Manhattan–Hopper*. Let $\vec{a}$ be a spatial vector such that $p_{i-1} + \vec{a} = p_i$. Then the hop-operation executed by relay v_i moves it to position $p'_i = p_{i+} - \vec{a}$. This is depicted in Fig. 8. After the hop-operation has been executed, the sequential run proceeds at the next relay.

Shorten-operation. In the shorten-operation the relay v_i moves to the middle of the line segment between p'_{i-1} and p_{i+1}, just as it did in the *Go-To-The-Middle* strategy. Executing the shorten-operation finishes the run of the *Hopper* strategy.

Again, we can pipeline runs, so that a new run starts in every third step.

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

5. Performance of Hopper in the static scenario

We again denote by n the number of relays at the beginning of the first run and by d the distance between the explorer and the base camp.

Analysis of the performance of the *Hopper* strategy will proceed in a way similar to analysis of the *Manhattan–Hopper* strategy. Unfortunately, the *Hopper* strategy not only removes or moves vectors in the configuration but also adds new vectors. In the course of the analysis we will show how these new vectors are formed. Eventually we will show that, after a linear number of steps, all pairs of vectors in the configuration form an angle larger than $\pi/2$. Our last lemma will show that a chain described by such a configuration is short. This yields the following theorem.

Theorem 6. *Starting with a chain with n relays, the Hopper strategy ensures that*

(a) *the chain remains connected,*
(b) *relays move at most distance* $\sqrt{2}$ *per run,*
(c) *after* $8n+1$ *runs, i.e.,* $25n+1$ *steps, the chain uses at most* $2\sqrt{2}d+1$ *relays.*

The parts (a) and (b) follow immediately from the strategy definition. The rest of this chapter is devoted to proving the bound on the number of runs stated in part (c).

Changes in configurations. We start with showing how a run can influence configurations.

Lemma 7. *Let* $C=(\vec{u}_0,\vec{u}_1,\ldots,\vec{u}_k)$ *be the configuration at the beginning of a run. Assume the run finishes without executing a remove-operation or a shorten-operation. Then the configuration after the run has completed is* $C'=(\vec{u}_1,\ldots,\vec{u}_k,\vec{u}_0)$. *Furthermore, for each* $i=0,\ldots,k$, $\angle(\vec{u}_0,\vec{u}_i)>\pi/2$ *holds.*

Proof. In the case of a run finishing without removing any relays from the chain and without executing a shorten-operation, all relays in the chain have been executing the hop-operation only. The hop-operation is executed by relay v_i only if the angle between $\langle p'_{i-1},p_i\rangle$ and $\langle p_i,p_{i+1}\rangle$ is larger than $\pi/2$. As $p'_{i-1}=p_i-\vec{u}_0$ when all relays prior to i have executed the hop-operation, the angle considered is equal to $\angle(\vec{u}_0,\vec{u}_i)$. This implies $\angle(\vec{u}_0,\vec{u}_i)>\pi/2$ for all $i=0,\ldots,k$. □

Lemma 8. *Let* $C=(\vec{u}_0,\vec{u}_1,\ldots,\vec{u}_k)$ *be the configuration at the beginning of the run. Assume the run finishes with* v_i *executing a remove-operation or a shorten-operation. Let* $C'=(\vec{a}_0,\vec{a}_1,\ldots,\vec{a}_m)$ *be the configuration of the chain at the beginning of the next run. Then each vector* $\vec{a}_i$ *is a positive linear combination of the vectors* $\vec{u}_0,\ldots,\vec{u}_k$. *More precisely,* $C'=(\vec{u}_1,\ldots,\vec{u}_{i-1},\frac{1}{2}(\vec{u}_0+\vec{u}_i),\frac{1}{2}(\vec{u}_0+\vec{u}_i),\vec{u}_{i+1},\ldots,\vec{u}_k)$ *in case of a shorten-operation, and* $C'=(\vec{u}_1,\ldots,\vec{u}_{i-1},\vec{u}_0+\vec{u}_i,\vec{u}_{i+1},\ldots,\vec{u}_k)$ *in case of a remove operation.*

Proof. Assume that relay v_i executes a remove- or a shorten-operation. Then all relays with indices smaller than i execute the usual hop-operation in the run. We can adapt the invariant introduced for the analysis of the *Manhattan–Hopper* strategy and state that $p'_j=p_{j+1}-\vec{u}_0$ holds for all $j<i$.

Let us first assume that the run has ended with a shorten-operation. The configuration of the chain prior to executing a shorten-operation by v_i is $(\vec{u}_1,\ldots,\vec{u}_{i-1},\vec{u}_0,\vec{u}_i,\ldots,\vec{u}_k)$. The shorten-operation moves relay v_i into the middle of the line segment between v_{i-1} and v_{i+1}. As v_{i-1} and v_{i+1} are connected through v_i, it holds $p'_{i-1}+\vec{u}_0+\vec{u}_i=p_{i+1}$. Therefore the shorten-operation replaces the vectors $\vec{u}_0,\vec{u}_i$ in the configuration by two vectors $\frac{1}{2}(\vec{u}_0+\vec{u}_i)$ and $\frac{1}{2}(\vec{u}_0+\vec{u}_i)$. Thus

$$C'=\left(\vec{u}_1,\ldots,\vec{u}_{i-1},\frac{1}{2}(\vec{u}_0+\vec{u}_i),\frac{1}{2}(\vec{u}_0+\vec{u}_i),\vec{u}_{i+1},\ldots,\vec{u}_k\right).$$

In the case the run ends with a relay being removed, the same argument yields

$$C'=(\vec{u}_1,\ldots,\vec{u}_{i-1},\vec{u}_0+\vec{u}_i,\vec{u}_{i+1},\ldots,\vec{u}_k).\quad\square$$

Shorten-operations. If the run ends with removing a relay we are certainly improving the chain by reducing its length. We want to show that executing a shorten-operation also reduces the chain length by a significant amount, if the involved spatial vectors have a minimum length.

Let the length of a chain C be defined by

$$|C|=\sum_{i=0,\ldots,k}|\vec{u}_i|.$$

Lemma 9. *Let* $C=(\vec{u}_0,\vec{u}_1,\ldots,\vec{u}_k)$ *be the configuration at the beginning of the run. Assume the run finishes with* v_i *executing a shorten-operation, yielding configuration* C'. *If* $|\vec{u}_0|\geq\frac{1}{2}$ *and* $|\vec{u}_i|\geq\frac{1}{2}$ *then* $|C'|\leq|C|-\frac{1}{3}$.

Proof. The configuration of the chain prior to executing a shorten-operation by v_i is given by $(\vec{u}_1,\ldots,\vec{u}_{i-1},\vec{u}_0,\vec{u}_i,\ldots,\vec{u}_k)$. For the sake of concise notation denote

$$a:=|\langle p_i,p'_{i-1}\rangle|=|\vec{u}_0|,\qquad b:=|\langle p_i,p_{i+1}\rangle|=|\vec{u}_i|,\qquad c:=|\langle p'_{i-1},p_{i+1}\rangle|=|\vec{u}_0+\vec{u}_i|.$$

As v_i moves to the middle of the line segment $\langle p'_{i-1},p_{i+1}\rangle$ the length of the chain decreases with the movement of v_i by $a+b-c$. Denote by γ the angle between $\langle p_i,p'_{i-1}\rangle$ and $\langle p_i,p_{i+1}\rangle$.

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

Note that by the Law of Cosines $c = \sqrt{a^2 + b^2 - 2ab\cos\gamma}$. Therefore, for fixed a, b, the value of c is maximized for $\gamma = \pi/2$. So, $a + b - c \geq a + b - \sqrt{a^2 + b^2}$. Using border conditions $\frac{1}{2} \leq a, b \leq 1$ and $1 \leq c \leq \sqrt{a^2 + b^2}$ the function $a + b - \sqrt{a^2 + b^2}$ is minimized for $a = \frac{1}{2}$, $b = \frac{1}{2}\sqrt{3}$, with $a + b - \sqrt{a^2 + b^2} \geq \frac{1}{2}\left(\sqrt{3} - 1\right) \geq \frac{1}{3}$. □

Unfortunately we cannot be sure that all vectors in the configuration have a length of at least $\frac{1}{2}$. Therefore there are some executions of the shorten-operation which shorten the path by an amount less than $\frac{1}{3}$.

Lemma 10. *There are at most $2n+1$ executions of shorten-operations such that one of the participating vectors has length smaller than $\frac{1}{2}$.*

Proof. There are at most $n + 1$ vectors with length smaller than $\frac{1}{2}$ at the beginning. The only possibility that a vector of length smaller than $\frac{1}{2}$ is created during the hopper-strategy is by a remove-operation. As such an operation can be executed at most n times, the lemma follows. □

Non-conflicting suffix. We now introduce a technical lemma, which is helpful in establishing the fact that a vector which becomes non-conflicting remains so for the whole future.

Lemma 11. *Let $\vec{u}$ be a spatial vector and $V = \vec{u}_1, \ldots, \vec{u}_k$ a set of spatial vectors. If $\angle(\vec{u}, \vec{u}_i) > \pi/2$ for all $i = 1, \ldots, k$, then, for any positive linear combination $\vec{s}$ of vectors from V, $\angle(\vec{u}, \vec{s}) > \pi/2$ holds.*

Proof. Without loss of generality let $\vec{u} = (1, 0)$. Then for any vector $\vec{u}_i$, the condition $\angle(\vec{u}, \vec{u}_i) > \pi/2$ implies $\vec{u}_i = (a_i, b_i)$ with $a_i > 0$. Thus, each positive linear combination $\vec{u}'$ of the vectors from V is of the form $\vec{u}' = (a', b')$ with $a' > 0$. This implies that $\angle(\vec{u}, \vec{u}') > \pi/2$. □

A non-conflicting suffix of a configuration C is a suffix of C consisting of non-conflicting vectors. Let S be the longest non-conflicting suffix of C. Then by Lemma 7, a run without remove- and shorten-operation makes S longer, because $\vec{u}_0$ is added to it.

Lemma 12. *The remove-operation and the shorten-operation do not decrease the size of the longest non-conflicting suffix.*

Proof. A shorten- or remove-operation can only shorten the non-conflicting suffix S, if it replaces $\vec{u}_0$ and some $\vec{u}_i$ contained in S by $\vec{u}_0 + \vec{u}_i$, or two copies of $\frac{1}{2}(\vec{u}_0 + \vec{u}_i)$. We will show that, in this situation, $\vec{u}_0 + \vec{u}_i$ is non-conflicting in the new configuration C'.

As $\vec{u}_i$ is included in S, it is non-conflicting in C. Furthermore, earlier in this run, a shorten-operation would have been executed if $\vec{u}_0$ had a conflict with a vector $\vec{u}_j$ with $j < i$. Therefore, also $\vec{u}_0$ is non-conflicting in C. This implies that $\vec{u}_0 + \vec{u}_i$ is by Lemma 11 non-conflicting and thus a proper replacement for $\vec{u}_i$ in S. Thus the size of the longest non-conflicting suffix does not decrease. □

Taking together the results established so far we can describe the following possible effects of a run

(1) it removes a relay,
(2) it reduces the path length by at least $\frac{1}{3}$,
(3) it reduces the path length by less than $\frac{1}{3}$,
(4) it makes the non-conflicting suffix longer.

Case 1 can only happen at most n times, Case 2 at most $3n$ times. By Lemma 10, Case 3 can occur only $2n + 1$ times. Case 4 occurs at most n times, because the length of the non-conflicting suffix can not exceed n, and it will never be shortened by Lemma 12.

Thus, after at most $6n + 1$ runs in total, the configuration is non-conflicting.

Chain length. What is still to be shown is that a non-conflicting configuration has a small length. The following lemma proves this fact.

Lemma 13. *Let $C = (\vec{u}_0, \ldots, \vec{u}_k)$ be a non-conflicting configuration. Then*

$$|\vec{u}_0| + \cdots + |\vec{u}_k| \leq \sqrt{2}d.$$

Proof. Sort all the vectors from C ascending w.r.t. their angle to the line through the explorer and base camp, construct a configuration $C' = (\vec{a}_0, \ldots, \vec{a}_k)$ out of this order, and rearrange the relays so that their positions reflect the configuration C'. Obviously $|C| = |C'|$ although the shapes of the corresponding chains may differ significantly.

Due to sorting of vectors, the polygon defined by the configuration C' and the line segment between the explorer and the base camp is convex.

Take now $\vec{a}_0$ and $\vec{a}_k$ and enlarge both of them so that they form a triangle together with the line segment between the explorer and the base camp, as shown on Fig. 9. The length of C' is not larger than $a + b$ as C' is convex.

As $\angle(\vec{a}_0, \vec{a}_k) > \pi/2$, the angle γ is at least $\pi/2$. Note that for a fixed d, the sum $a + b$, expressed as a function of α is decreasing. Therefore, the sum $a + b$ is maximized for $\gamma = \pi/2$. Maximizing $a + b$ with the condition $\sqrt{a^2 + b^2} = d$ we obtain $a + b \leq \sqrt{2}d$. □

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

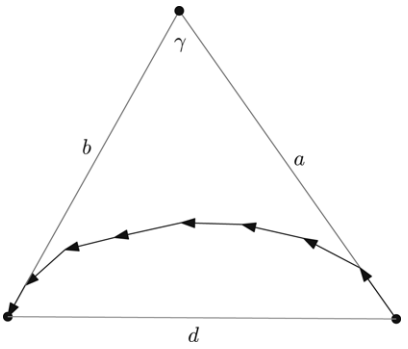


Fig. 9. The convex chain C' is shorter than $a + b$.

Number of relays. The former lemma shows only that the length of the chain after $8n + 1$ runs is bounded by $\sqrt{2}d$. We nevertheless have no bound on the number of relays used in this chain.

Lemma 14. *Let all k vectors be non-conflicting at the beginning of run r. Then after $2k$ runs at most one vector has length smaller than $\frac{1}{2}$.*

Proof. Since all vectors are non-conflicting, the shorten-operation is never executed after run r. Therefore, each run either ends with a remove-operation or it shifts a vector to the end of the configuration.

Assume now a run $r' \geq r$ with $C_{r'} = (\vec{u}_0, \ldots, \vec{u}_k)$. If $|\vec{u}_0| < \frac{1}{2}$ and $|\vec{u}_i| < \frac{1}{2}$ and no remove-operation has been executed by the run prior to the movement of v_{i+1}, then the strategy we have by our invariant is $p'_i = p_{i+1} - \vec{u}_0$.

By the triangle inequality, the distance between p'_i and p_{i+2} is smaller than 1, as $|\vec{u}_0| < \frac{1}{2}$ and $|\vec{u}_i| < \frac{1}{2}$. Then, the relay v_{i+1} executes the remove-operation.

On the contrary, if $|\vec{u}_0| \geq \frac{1}{2}$ and $|\vec{u}_i| \geq \frac{1}{2}$, then the relay v_{i+1} cannot execute the remove-operation, as by $\angle(\vec{u}_0, \vec{u}_i) > \pi/2$ the distance $|p'_i - p_{i+2}|_2 > 1$.

Denote by S_r the longest suffix of the configuration containing only vectors with length larger than $\frac{1}{2}$. Assume now that before run r the vector $\vec{u}_j$ is the first vector in S_r and that there is more than one vector with a size smaller than $\frac{1}{2}$ in the configuration. If $|\vec{u}_0| < \frac{1}{2}$ then the remove-operation will be executed either by relay v_i, such that $|\vec{u}_{i-1}| < \frac{1}{2}$ or by an earlier relay, so that S_r is left untouched. If $|\vec{u}_0| \geq \frac{1}{2}$ then by our earlier observation the remove-operation cannot influence S_r as all vectors in S_r have length at least $\frac{1}{2}$.

Therefore, after at most k shifts, there is at most one vector with size $\frac{1}{2}$. As there may be at most k remove-operations, there are k shifts in $2k$ runs. □

Let C be the configuration after $8n + 1$ runs. Using the former lemma, we can conclude that there are at most $2|C| + 1$ relays in the chain after this time. With the bound on $|C|$ from Lemma 13, part (c) of Theorem 6 follows, concluding the proof of this theorem.

6. Performance of Hopper in the dynamic scenario

The basic idea used for transforming the *Hopper* strategy into *Dynamic-Hopper* is very similar to that used for the *Dynamic-Manhattan–Hopper* strategy. The *Dynamic-Hopper* strategy is executed in rounds, such that at the beginning of the round the explorer moves, then a follow-run and a series of α hopper-runs are executed. The constant α will be defined later. We assume that all runs are pipelined, as shown on Fig. 10. Whenever we will be talking of runs, we mean hopper-runs.

We assume that each movement of the explorer has a distance of at least $\frac{1}{2}$ and at most 1. We assume that at the beginning of the first round, the length of the chain and the number of vectors are appropriately bounded by the distance between the base camp and the explorer. The exact restrictions on the chain are given later.

Theorem 15. *There exists a constant α such that for the chain maintained by the Dynamic-Hopper strategy the following holds.*

(a) *The chain remains connected,*
(b) *the explorer may move a distance of $\frac{1}{2}$ every round, i.e. every $3(\alpha + 1)$ time steps,*
(c) *relays move at most distance $\alpha\sqrt{2}$ per round,*
(d) *the number of relays used in the chain is $\mathcal{O}(d_r)$ at the beginning of round r.*

Parts (a)–(c) are obvious from the definition of the strategy. In the rest of the chapter we will prove part (d), thereby specifying the constant α. For this purpose we have to introduce notions of virtual length and infected vectors, as well as divide the execution time into phases. After this introduction we will discuss the outline of the remaining analysis in more detail.

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

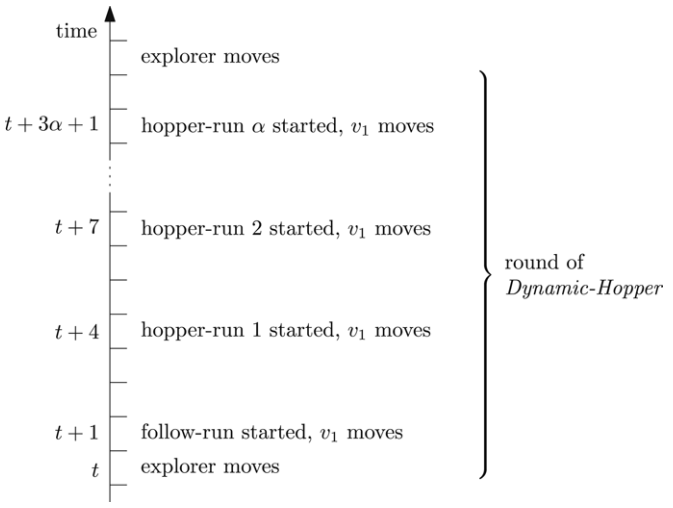


Fig. 10. Round of *Dynamic-Hopper* strategy.

Virtual length. We can define the virtual length of a spatial vector to be

$$\|\vec{u}\| = \begin{cases} |\vec{u}| & \text{if } |\vec{u}| \geq \frac{1}{2} \\ |\vec{u}| + \frac{1}{2} & \text{otherwise.} \end{cases}$$

This notion has the advantage that a vector has always virtual length of at least $\frac{1}{2}$. The virtual length of a configuration is obviously the sum of the virtual lengths of all vectors belonging to this configuration. In contrast, the length $|\vec{u}|$ will be called the real length.

We want to investigate how the operations performed by hopper-runs influence the virtual length. For the remove-operation, note that the sum of the participating vectors' virtual length is at least 1. On the other hand, the vector created by the remove-operation has a virtual length of at most 1. Therefore, the remove-operation does not increase the virtual length of the chain.

For the shorten-operation, either both of the participating vectors have a real length of at least $\frac{1}{2}$ and then by Lemma 9 the operation reduces the virtual chain length by at least $\frac{1}{3}$. Now assume that one of the vectors $\vec{u}_0$ and $\vec{u}_1$ participating in the shorten-operation has a real length smaller than $\frac{1}{2}$. W.l.o.g. let it be $\vec{u}_0$. For the real length of the resulting vector, it holds $|\vec{u}_0 + \vec{u}_1| \leq |\vec{u}_0| + |\vec{u}_1|$. By definition of the virtual length we have $|\vec{u}_0| + |\vec{u}_1| \leq \|\vec{u}_0\| - \frac{1}{2} + |\vec{u}_1|$. This implies that the shorten operation reduces the virtual length of the chain by at least $\frac{1}{2} > \frac{1}{3}$.

Fact 16. *Every shorten-operation reduces the virtual length of the chain by at least* $\frac{1}{3}$.

As each relay has a virtual length of at least $\frac{1}{2}$, virtual length can be used to bound the number of relays in the chain.

Fact 17. *Let l be the virtual length of the chain. Then the number of relays is bounded from above by $2l$.*

Phases. We will divide the runtime of the *Dynamic-Hopper* strategy in phases. The first phase starts in the first round. Let n denote the number of relays in the chain, l be its virtual length and d the distance between the base camp and the explorer in the beginning of the phase. Similarly, n', l' and d' denote the same at the end of the phase.

Throughout the analysis we use three constants. By α we denote the number of hopper-runs in a round, γn is the number of rounds in a phase. Furthermore, we assume that at the beginning of the phase $l \leq \beta d$ holds. This is also the restriction we lay upon the chain at the beginning of the first round. The rationale for this initial assumption is that the *Dynamic-Hopper* strategy is designed to maintain a steady quality of the chain all the time and therefore should start its work on a proper chain, too.

We require $\beta \geq 2\sqrt{2} + 1$ and the constants γ and α to fulfill

$$\gamma \leq \frac{1}{2\sqrt{2}(10\beta + 14)},$$

and $\alpha \geq \frac{8}{\gamma} + 8$.

Infected vectors. At the beginning of a phase all vectors are marked as non-infected. Every new vector introduced by movement of the explorer to the configuration is called infected. Every time a remove-operation or a shorten-operation removes at least one infected vector, the vectors created by the operation are marked as infected. Intuitively, the marking of infected vectors allows us to distinguish vectors which have been influenced by the movement of the explorer from the current phase.

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

A non-infected vector is said to be non-conflicting if it has no conflict to any non-infected vectors.

Outline of the analysis. We will first show that at the end of the phase, non-infected vectors have a real length of at least $\frac{1}{2}$ and that they all have angles larger than $\pi/2$ with respect to each other. Afterwards, we will distinguish two cases: either at the end of the phase there are more than $(3\beta+4)\gamma n$ infected vectors or not. In the first case we will show that so many infected vectors could have been only created by many shorten-operations which also decreased the length of the chain significantly. In the second case we will be able to show that thanks to the majority of vectors being non-infected, the chain length is linear in the distance d_r. In both cases we will show that $l' \leq \beta\, d'$.

Lemma 18. *At the end of a phase all non-infected vectors are non-conflicting and all but at most one have real length at least* $\frac{1}{2}$.

Proof. Let us follow the changes in position of a non-infected vector in configurations. For that purpose, whenever a vector is replaced by a shorten-operation, we assign its identification to the new vector created. In case of a remove-operation, we assign the new vector the identification of one of the removed vectors.

Whenever a vector is not at the first position in the configuration, its position can change in the following manner

(1) decrease by 1 if the run shifts a vector to the end of the configuration,
(2) decrease by 1 or without changes in case of a remove-operation or shorten-operation,
(3) increase by 1 if the explorer has moved and a new vectors has been added.

As there are at most $3(n+\gamma n)+(n+\gamma n)$ runs ending in Case 2 and at most $n+\gamma n$ occurrences of Case 3, after $7(n+\gamma n)$ runs each non-infected vector has been at the first position of the configuration and has been either shifted as non-conflicting to the end, or removed. Therefore, after $7(n+\gamma n)$ runs all non-infected vectors are non-conflicting. After at most $8(n+\gamma n)$ runs each vector has been on the first position for at least two times, and then following the argumentation from Lemma 14 all but one vectors have a real length of at least $\frac{1}{2}$.

By definition of α, we have $\alpha\gamma n \geq 8(n+\gamma n)$ and the lemma follows. □

Lemma 19. *Let there be at least* $(3\beta+4)\gamma n$ *infected vectors at the end of the phase. Then* $l' \leq \beta d'$.

Proof. The number of infected vectors in a phase generated by movement of the explorer is at most γn. Note that only the shorten-operation is able to generate two new infected vectors out of one, thereby increasing the number of infected vectors in the configuration. Therefore, in order to have $(3\beta+4)\gamma n$ infected vectors at the end of the phase, at least

$$(3\beta+4)\gamma n - \gamma n \geq 3(\beta+1)\gamma n$$

shorten-operations must have been performed during this phase.

Note that each of the shorten-operations decreases the virtual length of the chain by at least $\frac{1}{3}$ and the only operations enlarging the chain are movements of the explorer. So, we have $l' \leq l - (\beta+1)\gamma n + \gamma n = l - \beta\gamma n$. Since the explorer decreased its distance to the base camp by at most γn it follows that $d' \geq d - \gamma n$. From the definition of γ it follows $\gamma < \frac{1}{2\beta}$ which, thanks to $n \leq 2l \leq 2\beta d$, implies $d - \gamma > 0$. We obtain

$$\frac{l'}{d'} \leq \frac{l-\beta\gamma n}{d-\gamma n} \leq \frac{l}{d} \leq \beta.$$

The last inequality holds because $l \leq \beta d$. This implies the desired bound on l'. □

Lemma 20. *Let there be at most* $(3\beta+4)\gamma n$ *infected vectors at the end of the phase. Then* $l' \leq \beta d'$.

Proof. Sum up the non-infected vectors to a vector $\vec{u}$ and the infected vectors to a vector $\vec{a}$. Obviously, $p'_0 + \vec{u} + \vec{a} = p'_{k+1}$ where p'_{k+1} is the position of the base camp v_{k+1}. Define the point $\rho = p'_0 + \vec{u}$. Since all non-infected vectors have an angle larger than $\pi/2$ with respect to each other, by Lemma 13 the actual length of non-infected vectors is at most $\sqrt{2}\,|\rho - p_0|_2$. Furthermore, by Lemma 18 all but one non-infected vectors have a real length equal to its virtual length. Thus the virtual length of non-infected vectors L' is bounded by $L' \leq \sqrt{2}\,|\rho - p_0|_2 + \frac{1}{2}$. The virtual length of the chain is bounded by $l' \leq L' + |\vec{a}|$ and since $|\vec{a}| \leq (3\beta+4)\gamma n$ we have $L' \geq l' - (3\beta+4)\gamma n$.

Therefore,

$$\begin{aligned} d' &= |p'_0 - p'_{k+1}|_2 \geq |\vec{u}| - |\vec{a}| \geq \frac{1}{\sqrt{2}}L' - \frac{1}{2\sqrt{2}} - (3\beta+4)\gamma n \\ &\geq \frac{1}{\sqrt{2}}l' - (5\beta+7)\gamma n - \frac{1}{2\sqrt{2}}. \end{aligned} \tag{1}$$

As used earlier we have $n \leq 2l \leq 2\beta d$. On the other hand $d' \geq d - \gamma n$. Combining both bounds we obtain $n \leq d'\,\frac{2\beta}{1-2\beta\gamma}$.

Plugging in the bound on n into Eq. (1) and assuming $d' \geq 1$ we obtain

$$\frac{l'}{d'} \leq \sqrt{2}\left(1 + \frac{1}{2\sqrt{2}} + \frac{2\beta\gamma(5\beta+7)}{1-2\beta\gamma}\right) \leq \beta,$$

as $\gamma \leq \frac{1}{2\sqrt{2}(10\beta+14)}$ is sufficiently small. □

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010

Applying both Lemmas 19 and 20, we have $l \leq \beta d$ at the beginning of each phase. During the phase the number of vectors may increase by at most γn and the distance between the explorer and the base camp may decrease by at most γn. Denote by n_t the number of relays in the chain in time step n_t and the distance between explorer and base camp by d_t. Then we have

$$\frac{n_t}{d_t} \leq \frac{n+n\gamma}{d-n\gamma} \leq \frac{2\beta(1+\gamma)}{1-2\beta\gamma},$$

as $d \geq \frac{n}{2\beta}$. For the defined value of γ we have $\frac{2\beta(1+\gamma)}{1-2\beta\gamma} \leq \beta^2$ and therefore the number of relays in the chain at any point of time is at most $\beta^2 d_t$. This concludes the proof of part (d) of Theorem 15.

7. Conclusions

The presented *Hopper* strategy presents a major improvement in performance over earlier known strategies maintaining chains or relays stations. In comparison to the *Go-To-The-Middle* strategy, its behavior differs in the following points

- the execution is organized in sequential runs and not in parallel steps performed by all relays,
- relays are removed from the chain as soon as they are not necessary any more,
- the *Hopper* strategy lets relays oscillate rather than slowly converge as *Go-To-The-Middle*.

It is worth noting that the concept of sequential runs does not help on its own to improve the performance of *Go-To-The-Middle*, which is basically the same if changed from parallel to sequential. The authors believe rather that the combination of sequential runs together with oscillation helps the *Hopper* strategy to find situations s.t. relays can be safely removed. This is best seen in the example of *Manhattan–Hopper* strategy, where during a sequential run the strategy can get two relays to the same place and decide to remove one. The whole set of operations and technical tricks used by the *Hopper* strategy is only intended to mimic this behavior of *Manhattan–Hopper* and allow for the same effects without using a fixed grid.

Interesting problems subject to future research lie in maintaining the chain of relays on a plane with obstacles, and in generalizing the strategies to multiple explorers. Here we would have to deal with Steiner trees instead of chains.

References

[1] H. Ando, Y. Oasa, I. Suzuki, M. Yamashita, Distributed memoryless point convergence algorithm for mobile robots with limited visibility, IEEE Transactions on Robotics and Automation 15 (1999).
[2] N. Agmon, D. Peleg, Fault-tolerant gathering algorithms for autonomous mobile robots, in: Proc. of the 15th ACM-SIAM Symp. on Discrete Algorithms, SODA, 2004, pp. 1063–1071.
[3] M. Cieliebak, P. Flocchini, G. Prencipe, N. Santoro, Solving the robots gathering problem, in: Proc. of the 30th Int. Colloq. on Automata, Languages and Programming, ICALP, 2003, pp. 1181–1196.
[4] I. Chatzigiannakis, M. Markou, S. Nikoletseas, Distributed circle formation for anonymous oblivious robots, in: Proc. of the 3rd Workshop on Efficient and Experimental Algorithms, WEA, in: Lecture Notes in Computer Science, vol. 3059, 2004, pp. 159–174.
[5] M. Cieliebak, G. Prencipe, Gathering autonomous mobile robots, in: Proc. of the 9th Int. Colloq. on Structural Information and Communication Complexity, SIROCCO, in: Proceedings in Informatics, vol. 13, Carleton Scientific, 2002, pp. 57–72.
[6] R. Cohen, D. Peleg, Convergence of autonomous mobile robots with inaccurate sensors and movements, Technical Report MCS04-08, The Weizmann Institute of Science, 2004.
[7] R. Cohen, D. Peleg, Robot convergence via center-of-gravity algorithms, in: Proc. of the 11th Int. Colloq. on Structural Information and Communication Complexity, SIROCCO, in: Lecture Notes in Computer Science, vol. 3104, 2004, pp. 79–88.
[8] R. Cohen, D. Peleg, Convergence properties of the gravitational algorithm in asynchronous robot systems, SIAM Journal on Computing 34 (6) (2005) 1516–1528.
[9] X. Défago, A. Konagaya, Circle formation for oblivious anonymous mobile robots with no common sense of orientation, in: Proc. of the 2nd ACM International Workshop on Principles of Mobile Computing, POMC, ACM Press, 2002, pp. 97–104.
[10] M. Dynia, J. Kutylowski, P. Lorek, F. Meyer auf der Heide, Maintaining communication between an explorer and a base station, in: Proc. of the 1st IFIP Int. Conf. on Biologically Inspired Cooperative Computing, BICC, IFIP, Springer-Verlag, Berlin, 2006, pp. 137–146.
[11] M. Dynia, J. Kutylowski, F. Meyer auf der Heide, J. Schrieb, Local strategies for maintaining a chain of relay stations between an explorer and a base station, in: Proc. of the 19th ACM Symp. on Parallelism in Algorithms and Architectures, SPAA, ACM Press, 2007, pp. 260–269.
[12] S.P. Fekete, R. Klein, A. Nüchter, Online searching with an autonomous robot, Computational Geometry: Theory and Applications 34 (2006) 102–115.
[13] P. Flocchini, G. Prencipe, N. Santoro, P. Widmayer, Hard tasks for weak robots: The role of common knowledge in pattern formation by autonomous mobile robots, in: Proc. of the 10th Int. Symp. on Algorithms and Computation, ISAAC, 1999, pp. 93–102.
[14] Y. Katayama, Y. Tomida, H. Imazu, N. Inuzuka, K. Wada, Dynamic compass models and gathering algorithms for autonomous mobile robots, in: Proc. of the 14th Int. Colloq. on Structural Information and Communication Complexity, SIROCCO, in: Lecture Notes in Computer Science, vol. 4474, 2007, pp. 274–288.
[15] M. Mataric, Designing emergent behaviors: From local interactions to collective intelligence, in: Proc. of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats, vol. 2, 1992, pp. 432–441.
[16] H.G. Nguyen, N. Farrington, N. Pezeshkian, A. Gupta, J.M. Spector, Autonomous communication relays for tactical robots, in: Proc. of the 11th International Conference on Advanced Robotics, ICAR, 2003, pp. 35–40.
[17] H.G. Nguyen, N. Pezeshkian, A. Gupta, N. Farrington, Maintaining communication link for a robot operating in a hazardous environment, in: Proc. of the 10th Int. Conf. on Robotics and Remote Systems for Hazardous Environments, American Nuclear Society, 2004.
[18] I. Suzuki, M. Yamashita, Distributed anonymous mobile robots — formation and agreement problems, in: Proc. of the 3rd Int. Colloq. on Structural Information and Communication Complexity, SIROCCO, in: International Informatics Series, vol. 6, Carleton University Press, 1996, pp. 313–330.
[19] I. Suzuki, M. Yamashita, Distributed anonymous mobile robots: Formation of geometric patterns, SIAM Journal on Computing 28 (4) (1999) 1347–1363.

Please cite this article in press as: J. Kutyłowski, F. Meyer auf der Heide, Optimal strategies for maintaining a chain of relays between an explorer and a base camp, Theoretical Computer Science (2008), doi:10.1016/j.tcs.2008.04.010