Analysis of Heuristics for the Freeze-Tag Problem

Marcelo O. Sztainberg[1], Esther M. Arkin[1], Michael A. Bender[2], and Joseph S. B. Mitchell[1]

[1] Dept. of Applied Mathematics and Statistics, SUNY, Stony Brook, NY 11794, USA
{mos,estie,jsbm}@ams.sunysb.edu
[2] Dept. of Computer Science, SUNY, Stony Brook, NY 11794, USA
bender@cs.sunysb.edu

Abstract. In the Freeze Tag Problem (FTP) we are given a swarm of n asleep (*frozen* or *inactive*) robots and a single awake (*active*) robot, and we want to awaken all robots in the shortest possible time. A robot is awakened when an active robot "touches" it. The goal is to compute an optimal *awakening schedule* such that all robots are awake by time t^*, for the smallest possible value of t^*. We devise and test heuristic strategies on geometric and network datasets. Our experiments show that all of the strategies perform well, with the simple greedy strategy performing particularly well. A theoretical analysis of the greedy strategy gives a tight approximation bound of $\Theta(\sqrt{\log n})$ for points in the plane. We show more generally that the (tight) performance bound is $\Theta((\log n)^{1-1/d})$ in d dimensions. This is in contrast to general metric spaces, where greedy is known to have a $\Theta(\log n)$ approximation factor, and no method is known to achieve an approximation bound of $o(\log n)$.

1 Introduction

We consider a natural problem that arises in the study of *swarm robotics*. Consider a set of n *robots*, modeled as points in some metric space. There is one "awake" *source* robot; all other robots are *asleep* (inactive). In order to awaken a sleeping robot, an active robot travels to it and touches it; then, that robot can assist the set of active robots in awakening other asleep robots. Our goal is to activate (wake up) all of the robots as quickly as possible; i.e., we want to minimize the *makespan*, which is the time when the last robot is awakened.

This problem has been coined the *freeze-tag problem* (FTP) [1] because of its similarity with the children's game of "freeze-tag." In the game, the person who is "it" tags a player, who becomes "frozen" until another player (who is not "it" and not "frozen") tags him to unfreeze him. The FTP arises when there are a large number of players that are frozen, and one (not "it") unfrozen player, whose goal it is to unfreeze the rest of the players as quickly as possible. Once a player gets unfrozen, s/he is available to assist in unfreezing other frozen players, who can then assist, etc. Other applications of the FTP arise in the context of distributing data (or some other commodity), where physical proximity is required for distribution. This proximity may be necessary because wireless communication has too high a bandwidth cost or security risk. How does one propagate the data to the entire set of participants in the most efficient manner? Prior work on the dissemination of data in a graph includes the *minimum broadcast time problem*, the *multicast problem*, and the related *minimum gossip time problem*; see [4] for a survey and [2, 6] for recent approximation results.

The FTP is expressed as a combinatorial optimization problem as follows: Given a set of points in a metric space, find an arboresence (*awakening tree*) of minimum height where every node has out-degree at most two.

What makes the freeze-tag problem particularly intriguing is that *any* reasonable ("nonlazy") strategy yields an $O(\log n)$-approximation (Proposition 1.1 of [1]), whereas no strategy is known for general metric spaces that yields a $o(\log n)$-approximation. Arkin et al. [1] show that even very simple versions of the problem (e.g., on star metrics) are NP-complete. They give an efficient polynomial-time approximation scheme (PTAS) for geometric instances on a set of points in any constant dimension d. They also give results on star metrics, where $O(1)$-approximation is possible, and an $o(\log n)$-approximation for the special case of *ultrametrics*.

In this paper, our main results include the following:

(1) A proof that the natural greedy heuristic applied to geometric instances gives an $O((\log n)^{1-1/d})$-approximation in d dimensions. Thus, in one dimension, the greedy heuristic yields an $O(1)$-approximation, and in the plane the greedy heuristic yields an $O(\sqrt{\log n})$-approximation. We prove that this analysis is tight by supplying matching lower bounds.

(2) We perform an experimental investigation of heuristic strategies for the FTP, comparing the different choices of greedy strategies and comparing these greedy strategies with other heuristics.

Notation. We let S denote the *swarm*, the set of n points in a metric space (often Euclidean d-space, denoted $\Re^d$) where the initially asleep robots are located. We let s denote the *source point* where an initially active source robot is placed. We assume that any active robot in motion travels with unit speed. We let R denote the *radius* of the swarm S with respect to s; i.e., $R = \max_{p \in S} \operatorname{dist}(s, p)$. We let $D = \max_{p,q \in S \cup \{s\}} \operatorname{dist}(p, q)$ denote the diameter of the set of robots. We let t^* denote the minimum makespan. Note that, trivially, $t^* \geq R$ (since robots move with unit speed).

2 Wakeup Strategies for the FTP

2.1 Greedy Strategies

A natural strategy for the FTP is the greedy strategy, where an awake robot chooses the nearest asleep robot to awaken. The motivation for awakening nearby robots first is that parallelism is generated early on: the first robot awakens its closest neighbors, these newly awakened robots awaken their closest sleeping neighbors; thus there is rapid exponential growth in the number of awake robots.

In fact the greedy strategy is not a fully defined heuristic. What remains to be specified is how conflicts among robots are resolved, since two robots may have the same closest neighbor. We now describe three methods for resolving these conflicts: claims, refresh, and delayed target choice.

Claims. In order to guarantee that work is not duplicated, an awake robot *claims* the sleeping robot that it intends to awaken. Once a sleeping robot is claimed, no other robot is allowed to claim or awaken it.

Refresh. It may be beneficial for newly awakened robots to "renegotiate" the claims, since the set of awake robots changes. We refer to the ability to reassign active robots to asleep robots as the option to *refresh* claims.

We first consider the case in which claims are *not* adjusted. Thus, once an awake robot A claims an asleep robot B, A awakens B before making any other algorithmic decisions. The algorithm is now well defined because, without loss of generality, at most one robot is awakened at a time, and each time a robot is awakened it claims the nearest asleep robot.

Consider now the case in which claims *are* renegotiated, or *refreshed*, each time a new robot awakens. For motivation, consider the following scenario. An awake robot A is heading toward a sleeping robot B, which A has claimed. Before A reaches B, another robot C awakens. Now, both A and C would like to claim B, but since B is closer to C than to A, C takes over the responsibility of awakening B.

In our experiments, we assign claims by finding a matching between the awake robots and the asleep robots. We use a (potentially suboptimal) greedy strategy to compute a matching, rather than applying a more complex optimization algorithm. We order, by length, the potential matching edges between the set of currently awake and currently sleeping robots. We iteratively add the shortest edge e to the matching, and remove from consideration those edges that are incident to either of e's endpoints. The resulting matching has the property of giving priority to the short matching edges, which is faithful to the greedy heuristic.

Delayed Target Choice. Refresh introduces several anomalies. In particular, an (awake) robot may repeatedly change directions and oscillate without awakening any robots. This happens when an (awake) robot chooses an asleep target, but before the robot reaches its target, another robot claims the target. With the *delayed target choice* option we avoid these oscillations by making the solution "more off-line". Specifically, we avoid committing to the direction a robot is heading until that robot has traveled far enough to awaken its target, at which point the target (and its position) is fully determined.

Lower Bounds on the Performance of the Greedy Heuristic Our lower bounds hold for all variations of the greedy heuristic described above, since, in our lower bound examples, all awake robots will travel together in a "pack".

Theorem 1. *For any $\epsilon > 0$, there exists an instance of the FTP for points $S \subset \Re^1$ on a line ($d = 1$) for which the greedy heuristic results in a makespan that is at least $4 - \epsilon$ times optimal.*

Theorem 2. *The greedy heuristic is an $\Omega(\sqrt{\log n})$-approximation to the FTP in the plane.*

Proof: We arrange $\log n$ disks, each of radius 1, along a "zig-zag" path having $\sqrt{\log n}$ rows each having $\sqrt{\log n}$ disks, with disks touching but not overlapping along the path. Refer to Figure 1. The source robot is at the center, s, of the first disk. There is one asleep robot at the center of the second disk, then two at the center of the next, then four, etc, with the number of asleep robots at the center of each disk doubling as we advance along the path of touching disks. The last disk has (about) $\frac{n}{2}$ robots. When the greedy strategy is applied to this example, the awakening happens in sequence along the zig-zag path, with all of the newly awakened robots at the center of one disk targeting the robots at the center of the next disk. An optimal strategy, however, sends the source robot vertically upwards, awakening one cluster of robots at the center of each disk it passes along the way. These robots are then available to travel horizontally, awakening the robots along each row. This strategy yields makespan $t^* = O(\sqrt{\log n})$. Hence, greedy is an $\Omega(\sqrt{\log n})$-approximation. □

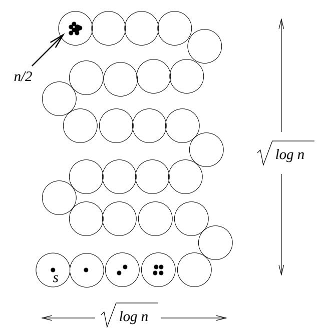

Fig. 1. The lower bound construction in $\Re^2$.

Theorem 3. *The greedy strategy is an $\Omega((\log n)^{1-1/d})$-approximation to the FTP in $\Re^d$.*

Upper Bounds for the Greedy Heuristic We first show that the lower bound of Theorem 1 is essentially tight for the one-dimensional case.

Theorem 4. *The greedy heuristic for a swarm $S \subset \Re$ of points on a line ($d = 1$) is a 4-approximation.*

The following theorems show that, in contrast with arbitrary metric spaces (where greedy gives an $O(\log n)$-approximation for the FTP), greedy is an $o(\log n)$-approximation for geometric instances. Combined with our lower bounds, we get a *tight* analysis of the approximation factor for greedy in all dimensions $d \geq 1$.

Theorem 5. *The greedy strategy is an $O(\sqrt{\log n})$-approximation to the FTP in the plane. One can compute the greedy awakening tree in time $O(n \log^{O(1)} n)$.*

Theorem 6. *The Greedy strategy is an $O((\log n)^{1-1/d})$ approximation to the FTP in d dimensions. One can compute the greedy awakening tree in time $O(n \log^{O(1)} n)$.*

In both theorems, the algorithmic complexity follows from applying dynamic methods for nearest neighbor search (or approximate nearest neighbor search [3, 5]) in order to identify which asleep robot is closest to a newly awakened robot. The nearest neighbor search requires deletions, since robots get deleted from the set of candidate targets when they awaken.

We concentrate on proving the approximation bound for the 2-dimensional case (Theorem 5); the d-dimensional case is a fairly direct generalization.

Suppose that we are given a greedy awakening tree for a swarm of size $|S| = n$. We can convert this tree into an awakening tree (at no increase in makespan) having the following property: on any root-to-leaf path there is at most one non-leaf node having out-degree 1 (rather than 2). Based on this property, and a simple counting argument, there exists a root-to-leaf path, $P = (p_0, p_1, \ldots, p_K)$, having $K \leq \log n$ edges of lengths $r_i = \text{dist}(p_i, p_{i+1})$. At the time t when the last node p_K is reached, all of the remaining asleep robots are robots that will be awakened by other branches of the awakening tree (by definition of the awakening tree). The makespan, therefore, must be at most $t + D \leq t + 2R$. We will show that $t = O(\sqrt{\log n} \cdot t^*)$, implying that the makespan is $O(\sqrt{\log n} \cdot t^*)$

Fixing attention now on one root-to-leaf path, P, in the awakening tree, we define the *outer circle* C_i to be the circle centered at p_i with radius $r_i = \text{dist}(p_i, p_{i+1})$; the *inner circle* c_i is the circle centered at p_i with radius $\frac{1}{3} r_i$. The properties of the greedy heuristic ensure the following claim:

Claim **1** *None of the points $p_{i+1}, \ldots, p_K$ lie inside circle C_i.*

The proof of Theorem 5 is based on an *area-covering argument*. Specifically, we provide a bound on the area covered by the circles $C_0, C_1, \ldots C_{K-1}$, showing that $\sum_{i=0}^{K-1} r_i^2 = O(R^2)$, where R is the radius of the swarm. The subtlety of the proof is that neither the outer circles $C_0, \ldots, C_{K-1}$ nor even the inner circles $c_0, \ldots, c_{K-1}$ are disjoint. The proof is based on the following lemma:

Lemma 1. *The combined area covered by the (inner or outer) circles is $O(R^2)$;*

$$\sum_{i=0}^{K-1} \mathit{area}(C_i) = O\left(\sum_{i=0}^{K-1} \mathit{area}(c_i)\right) = O\left(\sum_{i=0}^{K-1} r_i^2\right) = O(R^2).$$

Proof of Theorem 5: The length, L, of the root-to-leaf path P is simply $L = \sum_{i=0}^{K-1} r_i$. By the Cauchy-Schwartz inequality, the fact that $K \leq \log n$, and Lemma 1 we get

$$L = \sum_{i=0}^{K-1} r_i \leq \sqrt{K} \sqrt{\sum_{i=0}^{K-1} r_i^2} = O(R\sqrt{\log n}).$$

Since this result holds for any root-to-leaf path in the awakening tree, the approximation bound follows. □

Proof of Lemma 1: Each (outer,inner) circle pair, (C_i, c_i), is assigned to a *circle class* according to its radius. Without loss of generality, let the smallest outer circle have radius 1. Define *class i* to be the set of (outer,inner) circle pairs such that the radius of the outer circle has radius r, with $2^{i-1} \leq r < 2^i$.

While pairs of outer circles (and pairs of inner circles) may overlap, using the property of circle classes, we show the following claim:

Claim **2** *Any pair of inner circles belonging to the same class are disjoint.*

Let area A_i be the area covered by inner circles of class i. We claim that

Claim **3** *In order for the inner circles of class i associated with path P to cover area A_i, the corresponding edges of P (associated with circle pairs of class i) must have total length ℓ_i satisfying $\frac{9A_i}{2^{i+1}\pi} \leq \ell_i \leq \frac{9A_i}{2^{i-2}\pi}$.*

Since along path P of the awakening tree we have circles of different classes appearing, not necessarily in order of size, we may have inner circles overlapping, thus not covering "new" area. We need the following claim:

Claim **4** *The length of P that does* not *correspond to edges whose inner circles cover new area is at most a constant fraction of the length that corresponds to edges whose inner circles do cover new area.*

Thus, by Claim 4, the distance traveled along P in which no new area is covered is of the order of the distance traveled in which new area is covered. Since the total area covered is $O(R^2)$, the claim of Lemma 1 follows. □

2.2 Other Heuristic Wakeup Strategies

The greedy strategy has the following weakness: it may be preferable for a robot to travel a longer distance to obtain a better payoff. In this section we examine alternative strategies in an attempt to overcome this weakness.

We design our alternative strategies while keeping in mind the actual application that motivated our study: the need to activate a swarm of small experimental robots, each equipped with certain sensors. The sensors on the actual robots in our project (joint with HRL Labs) have the feature that they sense other robots (or obstacles) in each of k sectors, evenly distributed around the (circular) robot. (Our robots have $k = 8$, implying 45-degree sectors.) Within each of its sectors, a robot can detect only the closest other robot. Thus, there are at most k options facing a robot once it is activated: Which sector should be selected? Once the sector is selected, the robot heads for the closest asleep robot it has sensed in that sector. A greedy strategy that uses sensor sectors will select the sector whose closest robot is closest.
Bang-for-the-Buck. A natural strategy, which we call "bang-for-the-buck", is to choose the next target to awaken based on maximizing the ratio of "value" ("bang") to "cost" ("buck"). There are a variety of heuristic measures of potential value; a particularly simple one is to consider the value of a sector to be the number of currently asleep robots in the sector. The cost of a sector is chosen to be the distance to the nearest asleep robot in the sector.

Random Sector Selection. The goal of this strategy is to "mix up" at random the directions in which robots head to awaken other robots. When a robot awakens, it selects, at random, a sector and chooses its next target to be the closest asleep robot in that sector.

Opposite Cone. In this strategy the goal is to enforce a certain amount of "mixing up" of directions that robots head, by sending a newly awakened robot in a nearly opposite direction from that of the robot that awakened it. In particular, suppose robot A heads due east to awaken robot B at point p; then, the next target that robot A selects will be chosen to be the closest asleep robot in a cone centered on a vector to the west, while the target for robot B to awaken next will be selected from a cone centered on a vector to the east.

3 Experiments

3.1 Experimental Setup

Our experiments are based on a Java simulation of our various strategies. All tests were performed on a PC running Linux OS. The graphical user interface permits the user to select the choice of strategy, the parameters, and the input dataset, and it optionally shows a graphical animation of the simulation.

Datasets. We tested our strategies on both geometric and non-geometric datasets. We investigated four classes of geometric datasets: (1) uniform over a large square (600-by-600); (2) cluster ($\sqrt{n}$ clusters, each of a random size, generated uniformly over a square of side length $c = 2\sqrt{n}$, whose upper left corner is uniformly distributed over the square); (3) square grid; and (4) a regular hexagonal grid. Since our non-greedy strategies are specified geometrically, they were applied only to the geometric data.

The greedy strategies were applied to *non*-geometric datasets, including: (1) Star metrics: the $n-1$ asleep robots are at the leaves of a star (spoke lengths are chosen uniformly between 1 and n), the source robot is at the root. There is either one asleep robot per leaf (case "1-1") or many asleep robots per leaf (case "1-m": we generate $O(\sqrt{n})$ spokes and randomly assign m robots to each spoke, with m chosen uniformly between 1 and $O(\sqrt{n})$). (2) TSPLIB: symmetric traveling salesman files, with data of type MATRIX (14 instances).

Performance Measures. We maintain performance statistics, including: (1) *total time* of the simulation; (2) *total distance* traveled by all robots; and (3) *average distance* traveled by robots that do any traveling. We found that total distance and average distance were tightly correlated with the total time of the simulation; thus, here we report results only on the total time.

Parameter Choices. For each of the strategies, we considered each possible setting of the set of parameter choices (Claims, Refresh, or Delayed Target Choice) discussed in Section 2.1.

3.2 Experimental Results

The experiments on synthetically generated datasets were conducted as follows. For each choice of wakeup strategy, parameters, and dataset, a set of 100 runs was performed, with 10 runs for each value of $n \in \{100, 200, 300, \ldots, 1000\}$.

The experiments on datasets from the TSPLIB (EUC_2D or MATRIX) were done once per dataset, with one robot per point.

We performed runs for the following combinations of strategies and datasets: Greedy was run on all datasets (Uniform, Cluster, Grid, Hexagonal Grid, Stars 1-1, Stars 1-m, TSPLIB (EUC_2D and MATRIX)), while Bang-for-the-Buck, Random Sector Selection, and Opposite Cone were run on only the geometric instances (Uniform, Cluster, Grid, Hexagonal Grid, TSPLIB (EUC_2D)).

In our plots, the horizontal axis corresponds to the swarm size n, the vertical axis to the ratio of the makespan to the lower bound on makespan (the radius, R, except in some star-metric cases). Thus, the vertical axis shows an upper bound on the approximation ratio.

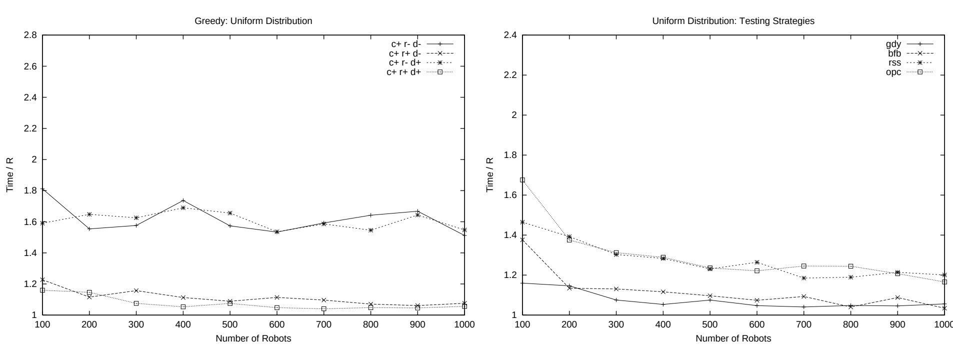

Fig. 2. Left: Greedy on uniform distributions: Choices of parameters for claims (c), refresh (r), and delayed target choice (d). A "+" ("-") indicates the parameter is set to *true* (*false*). **Right:** Greedy (gdy), Bang-for-the-Buck (bfb), Random Sector Selection (rss), and Opposite Cone (opc) tested on uniformly distributed swarms.

Parameter Choices. We experimented with various parameter choices over a common dataset. The claims option is very significant; without it the robots travel in groups rather than spreading out. There is a significant advantage in using the refresh option. While less significant, the delayed target choice is also advantageous. Figure 2 shows the result of running greedy on uniformly distributed points; other datasets yield similar results, with the delayed target choice showing a more pronounced advantage in the case of cluster datasets.

Wakeup Strategy Comparison. The main conclusion we draw from our experiments is that the greedy strategy is a very good heuristic, most often outperforming the other strategies in our comparisons. See Figure 3. As the size (n) of the swarm increases, the approximation ratios tend to stay about the same or decrease; we suspect this is because R becomes a better lower bound for larger swarms. The upper bounds (Time/R) on approximation factors in the geometric instances are between 1.0 and 1.5. For star metrics, the ratio Time/R is significantly higher (between 2 and 3), but this is due to the fact that R is a poor lower bound in the case of stars. In order to verify this, we computed an alternative lower bound specifically for star metrics having one robot per leaf. (The lower bound is $2L_{min}(\lceil \log(n+1) \rceil - 1) + L_{max}$, where L_{min} (resp., L_{max}) is the length of the shortest (resp., longest) spoke of the star.) Figure 4(right) shows the results of greedy on stars 1-1 datasets (of various spoke length distributions) using this lower bound in computing the approximation ratio; we see that there is a striking improvement over using the lower bound of R (which is particularly poor for star metrics). We also give tables (Tables 1,2) showing the percentage of wins for each strategy, and, finally, report in Table 3 the results for greedy on the non-geometric TSPLIB instances.

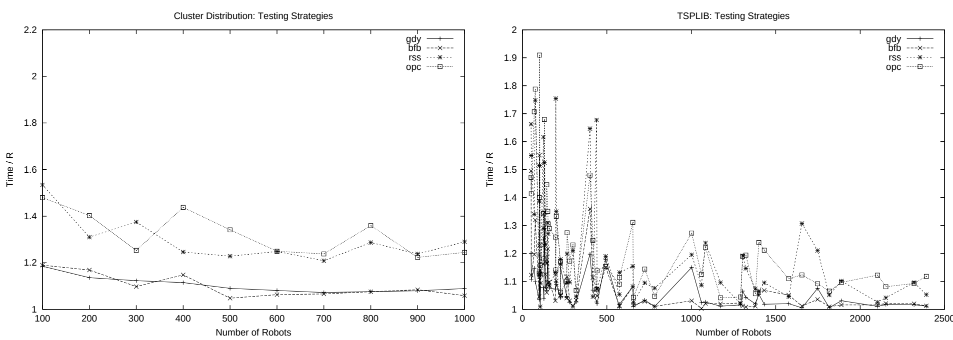

Fig. 3. Strategy comparison on cluster data (left) and TSPLIB EUC_2D data (right).

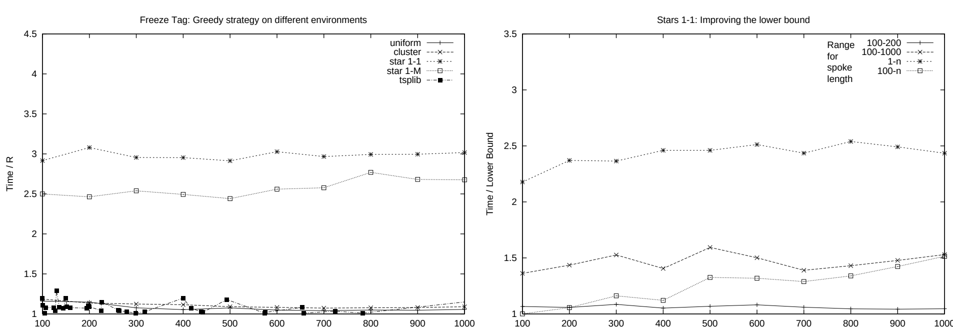

Fig. 4. Left: Comparing greedy on five different datasets. **Right**: Comparing greedy on star 1-1 datasets, *using the improved lower bound* of $2L_{min}(\lceil \log(n+1) \rceil - 1) + L_{max}$. Each curve corresponds to a randomly generated dataset having spoke lengths uniformly generated in the indicated interval.

Algorithm	Wins	%
GDY	45	66.20
BFB	22	32.35
RSS	1	1.45
OPC	0	0.00

Algorithm	Approx. Rate			Winning %		
	Best	Worst	Avg	Max	Min	Avg
GDY	1.00	1.29	1.06	41.13	0.01	6.20
BFB	1.00	1.36	1.07	18.71	2.24	3.97
RSS	1.04	1.04	1.04	2.24	2.24	2.24

Table 1. Left: Comparing strategies on the 68 TSPLIB (EUC_2D) datasets. Right: Winning strategies for the TSPLIB (EUC_2D) datasets: For those runs in which a strategy outperformed the others, we compute the maximum, minimum and average approximation factor and percent by which it lead over the second-place strategy.

Algorithm	Win %
GDY	62
BFB	37
RSS	0
OPC	1

Algorithm	Approx. Rate			Winning %		
	Best	Worst	Avg	Max	Min	Avg
GDY	1.01	1.31	1.06	57.32	0.33	8.83
BFB	1.00	1.16	1.04	27.23	0.13	4.04
OPC	1.23	1.23	1.23	5.25	5.25	5.25

Table 2. Comparing strategies on uniform datasets.

# of Robots	17	21	24	26	42	42	48	48	58	120	175	535	561	1032
Approx. Rate	1.06	1.04	1.16	1.36	1.08	1.12	1.03	1.24	1.19	1.12	3.26	3.01	1.17	3.19

Table 3. Results of greedy strategy for TSPLIB MATRIX (non-geometric) instances.

Acknowledgements. We thank Doug Gage for discussions motivating this research. This research was partially supported by HRL Labs (DARPA subcontract), NASA Ames Research, NSF, Sandia, and the U.S.-Israel Binational Science Foundation.

References

1. E. M. Arkin, M. A. Bender, S. P. Fekete, J. S. B. Mitchell, and M. Skutella. The freeze-tag problem: How to wake up a swarm of robots. In *Proc. 13th ACM-SIAM Sympos. Discrete Algorithms*, pp. 568–577, 2002.
2. A. Bar-Noy, S. Guha, J. Naor, and B. Schieber. Multicasting in heterogeneous networks. In *Proc. 30th ACM Sympos. Theory of Comput.*, pp. 448–453, 1998.
3. S. N. Bespamyatnikh. Dynamic algorithms for approximate neighbor searching. In *Proc. 8th Canad. Conf. Comput. Geom.*, pp. 252–257, 1996.
4. S. M. Hedetniemi, T. Hedetniemi, and A. L. Liestman. A Survey of Gossiping and Broadcasting in Communication Networks. *NETWORKS*, 18:319–349, 1988.
5. S. Kapoor and M. Smid. New techniques for exact and approximate dynamic closest-point problems. *SIAM J. Comput.*, 25:775–796, 1996.
6. R. Ravi. Rapid rumor ramification: Approximating the minimum broadcast time. *Proc. 35th Ann. Sympos. on Foundations of Computer Sci.*, pp. 202–213, 1994.
7. `http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/`.