Freeze-Tag Remains NP-hard on Binary and Ternary Trees[*]

Lehilton Lelis Chaves Pedrosa,
Lucas de Oliveira Silva[†]

[1] Institute of Computing
State University of Campinas (Unicamp)
Campinas, SP – Brazil

lehilton@ic.unicamp.br,l220715@dac.unicamp.br

Abstract. *The Freeze-Tag Problem (FTP) is a scheduling-like problem motivated by robot swarm activation. The input consists of the locations of a set of mobile robots in some metric space. One robot is initially active, while the others are initially frozen. Active robots can move at unit speed, and upon reaching the location of a frozen robot, the latter is activated. The goal is to activate all the robots within the minimum time, i.e., minimizing the time the last frozen robot is activated, the so-called makespan of the schedule.*

Arkin et al. proved that FTP is strongly NP-hard even if we restrict the problem to metric spaces arising from the metric closure of an edge-weighted star graph, where a frozen robot is placed on each leaf, and the active robot is placed at the center of this star [Arkin et al. 2002]. In this work, we continue to explore the complexity of FTP and show that it keeps its hardness even if further restricted to binary unweighted rooted trees with frozen robots only at leaves and the active robot on its root. Additionally, we prove that a generalized version, whose domain includes ternary weighted trees, remains hard, even if we require that every non-root node has precisely one frozen robot.

1. Introduction

The Freeze-Tag Problem (FTP) was introduced in 2002 by Arkin et al. to model the automatic awakening of a robot swarm that is started by manually turning on a single robot [Arkin et al. 2002]. The input consists of a set of points in some metric space representing the location of each mobile robot, as well as a special point in this set representing the starting robot. The starting robot is initially "active" (also known as "on", "unfrozen" or "awake") and is called the source, while the other robots are initially "frozen" (also "off" or "asleep"). Frozen robots become active when an active one reaches its location. Once activated, a robot can move at unit speed and help unfreeze the remaining ones. The goal is to minimize the time the last frozen robot is activated, the so-called makespan of the schedule. In the decision version, one is also given a time limit, and the goal is to decide if all the robots can be activated within this limit.

Frequently, the points correspond to nodes of some underlying graph, and the distance between two points is the length of a shortest path connecting them. In this work,

[*]Supported by São Paulo Research Foundation (FAPESP) grant #2022/13435-4 and National Council for Scientific and Technological Development (CNPq) grant #312186/2020-7.

[†]Corresponding author.

we consider instances whose corresponding metric spaces arise from the node distances in some, possibly weighted, tree and assume that such a tree is given in the input.

Depending on the choice of metric, the problem takes different forms. For example, Arkin et al. showed that FTP admits a polynomial-time approximation scheme (PTAS) for Euclidean spaces but has no polynomial time $(5/3 - \epsilon)$-approximation for weighted metric graphs in general [Arkin et al. 2002]. Moreover, Bender et al. showed that FTP is hard on unweighted general graphs, even with precisely one robot per node [Arkin et al. 2003].

In the same line, Arkin et al. revealed a dichotomy for the hardness of FTP in stars graphs that lie in the edge weights' presence. The problem is NP-hard with them but becomes polynomially solvable if restricted to unweighted starts [Arkin et al. 2002]. The algorithm they presented to the unweighted case is a greedy one, which is relatively straightforward to prove optimal: Each active robot currently at the root claims a leaf with the maximum number of (frozen) robots and goes there to activate them (and then all the robots at said leaf return simultaneously to the root).

Regarding Euclidean spaces, Abel et al. in 2017 were the first to establish the complexity of FTP in some Euclidean metric space by proving that it's NP-hard in the plane with L_2 distance [Abel et al. 2017]. This result was followed by a paper by Demaine and Rudoy, who proved that the problem is also hard in 3D Euclidean space for any L_p metric with $p > 1$ [Demaine and Rudoy 2017].

In this work, we complement the results of Arkin et al. [Arkin et al. 2006] by proving that FTP remains hard if restricted to degree-bounded trees. Namely, we show that FTP is NP-hard on binary unweighted or ternary weighted rooted trees. Furthermore, the latter holds in the strong sense even if we require that every non-root node has one frozen robot.

2. Problem Definition

Consider a metric space over some set V, which is the *domain* of the problem, with a corresponding distance function dist : $V \times V \to \mathbb{R}_{\geq 0}$. In this work, we assume that the domain corresponds to the nodes of some graph $G = (V, E)$, with non-negative edge weights, and the dist(u, v) corresponds to the length of a shortest path between vertices u and v. An FTP instance consists of a set of robots R and a special robot $v_0 \in R$, called the *source*. Each robot $v_i \in R$ is identified with a location $v_i \in V$. Thus R is a subset of V, possibly with repetition. We assume that robot v_0 is initially active and the other robots are initially frozen.

A solution for an instance (R, v_0) is a rooted binary tree $\mathcal{T}$, which is called *schedule* or *wake-up tree*. The nodes of this tree correspond to R, and the root is v_0. The children of a node v_i in $\mathcal{T}$ represent the next destinations of the two robots at v_i that will be available once an active robot first reaches it. If one of these robots stops moving, it will have no corresponding destination node, and v_i will be a leaf or have only one child. The weight of an edge of $\mathcal{T}$ corresponds to the distance between the two corresponding locations. The length of a longest path between the root and a leaf is the *makespan* of the scheduling, which is denoted by cost$(\mathcal{T})$. The objective of FTP is finding a scheduling $\mathcal{T}$ with minimum makespan. In the decision version, one is also given a time limit of L, and the goal is to decide whether some scheduling achieves this bound.

In this work, we consider only the offline version of the problem, where the algorithm can query the whole input simultaneously to build a schedule. In the context of the robot swarm application, this means that each active robot has position information of all the others and that the active robots can coordinate their movements.

Although not addressed here, some online versions have already been tackled. In 2006, Hammar et al. studied the FTP from the perspective of online algorithms [Hammar et al. 2006]. In this version of the problem, instead of receiving all the input at once, an algorithm receives its input serially in pieces. In this case, the performance is usually measured as the worst-case competitiveness ratio of the online algorithm, that is, the ratio between the algorithm's solution value and the one achievable by an optimal offline algorithm. More recently, in 2019, Brunner and Wellman presented a $(1 + \sqrt{2})$-competitive strategy with the makespan as the value. They also provided an optimality guarantee for metric domains [Brunner and Wellman 2019].

3. Complexity Results

3.1. Binary Trees

In this subsection, we consider the FTP with domain arising from an unweighted binary input tree $T = (V, E)$ rooted at its source node v_0. Furthermore, there is precisely one frozen robot on each leaf and no other robot. Below, we show that even under these restrictions, the problem remains hard.

Theorem 1. *FTP is* NP-hard *for the case of unweighted binary rooted trees with a single frozen robot on each leaf and the source on its root.*

The proof is based on a reduction from FTP on edge-weighted star graphs with a frozen robot on each leaf and source in its center, which is known to be strongly *NP-hard* [Arkin et al. 2002], and is illustrated in Figure 1. The main idea employed is to covert the center node of the star, which is the only one of high degree, into a binary tree and substitute the weighted edges for very long paths. We only require that these new paths outweigh the overhead introduced by the center node conversion, ensuring just a multiplicative factor in the final makespan.

3.2. Ternary Trees

We now consider a generalization that includes edge weights and relaxes the maximum degree of nodes to four but restricts the instances such that every node has precisely one robot. We show that this version is strongly *NP-hard*.

Theorem 2. *FTP is strongly* NP-hard *for the case of weighted ternary rooted trees with precisely one frozen robot on each non-root node and the source on its root.*

Again the base problem used here is the FTP in edge-weighted star graphs with a frozen robot on each leaf and the source in its center. The construction looks very much like the one of Theorem 2 (see Figure 2 for an illustration), but with an additional node attached to every non-root node of the binary tree using an edge of huge weight. Therefore each new node, with its corresponding robot, will "occupy" all tree robots and simplify the whole instance into a star-like domain.

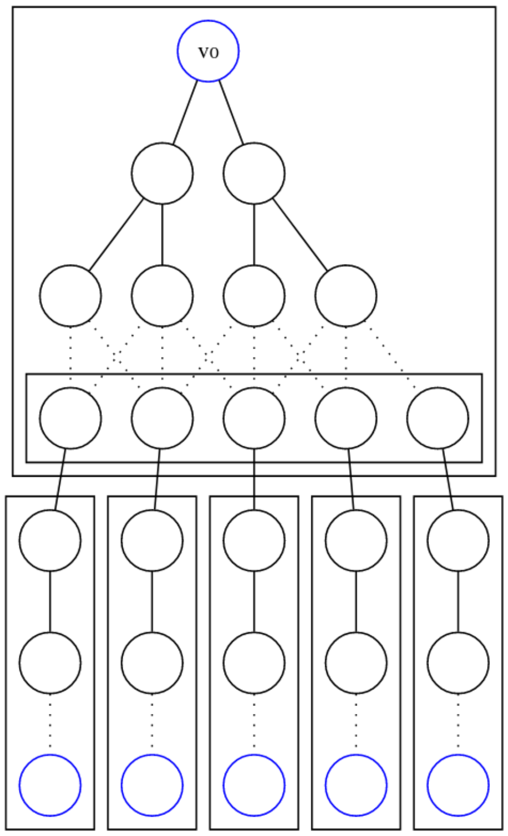

Figure 1. The FTP instance produced by the reduction of Theorem 1.

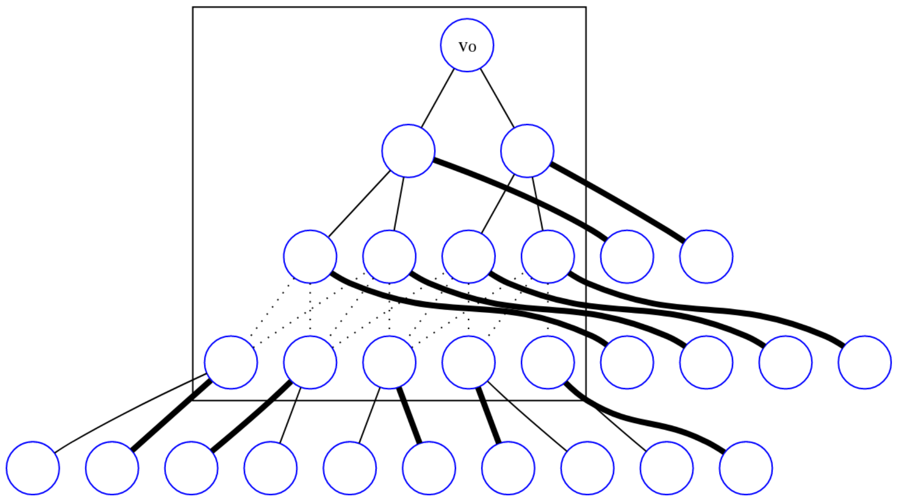

Figure 2. The FTP instance produced by the reduction of Theorem 2 (thick lines represent edges of high weight).

4. Conclusion

To further delimit the hardness of FTP in trees, one could consider unweighted or weighted binary rooted trees with precisely one frozen robot on each non-root node and the source on its root. We believe that they are both polynomial problems.

If our claim is valid for the unweighted case, a possible next step would be to generalize the found algorithm to any tree. And upon success, generalize it even further by constructing a parameterized algorithm for FTP in general unweighted graphs with the treewidth as a parameter.

In contrast, by Arkin et al., we know that FTP is *NP-hard* for general unweighted graphs with precisely one frozen robot on each non-source node [Arkin et al. 2003]. So we would like to find a broader class of polynomially solvable graph domains where each node starts with a robot.

References

Abel, Z., Akitaya, H. A., and Yu, J. (2017). Freeze Tag Awakening in 2D is NP-Hard. *In Abstracts from the 27th Fall Workshop on Computational Geometry*, pages 105–107.

Arkin, E. M., Bender, M. A., Fekete, S. P., Mitchell, J. S. B., and Skutella, M. (2002). The Freeze-Tag Problem: How to Wake up a Swarm of Robots. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, page 568–577. Society for Industrial and Applied Mathematics.

Arkin, E. M., Bender, M. A., Fekete, S. P., Mitchell, J. S. B., and Skutella, M. (2006). The Freeze-Tag Problem: How to Wake Up a Swarm of Robots. *Algorithmica*, 46(2):193–221.

Arkin, E. M., Bender, M. A., and Ge, D. (2003). Improved Approximation Algorithms for the Freeze-Tag Problem. In *Proceedings of the Fifteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 295–303. Association for Computing Machinery.

Brunner, J. and Wellman, J. (2019). An Optimal Algorithm for Online Freeze-Tag. In Farach-Colton, M., Prencipe, G., and Uehara, R., editors, *10th International Conference on Fun with Algorithms (FUN 2021)*, volume 157 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 8:1–8:11. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

Demaine, E. D. and Rudoy, M. (2017). Freeze Tag is Hard in 3D. *In Abstracts from the 27th Fall Workshop on Computational Geometry*, pages 108–110.

Hammar, M., Nilsson, B. J., and Persson, M. (2006). The online freeze-tag problem. In *LATIN 2006: Theoretical Informatics*, pages 569–579. Springer Berlin Heidelberg.