Tracking of Multiple Targets Using Optimal Number of UAVs

ARUN DAS
SHAHRZAD SHIRAZIPOURAZAD
Arizona State University, Tempe, AZ USA

DAVID HAY
Hebrew University of Jerusalem, Jerusalem, Israel

ARUNABHA SEN
Arizona State University, Tempe, AZ USA

In this paper, we study the problem of mobile target tracking using the fewest number of mobile trackers for two different types of targets. Given the target trajectories and the period of observation, we propose techniques to compute the *minimum number of trackers* and their *trajectories* required to track all mobile targets. Two classes of mobile targets are considered in this paper: 1) targets that need tracking for the *entire duration of observation* and 2) targets that need tracking *at least once* during the period of observation. We show that even when target trajectories are known in advance, the problem is computationally hard, i.e., NP-complete. We formulate the problem as a network flow problem and propose algorithms for its solution. We evaluate the performance of our algorithms through simulation and study the impact of parameters such as the speed and sensing range of the trackers.

Manuscript received September 1, 2017; revised November 30, 2017 and July 20, 2018; released for publication September 7, 2018. Date of publication October 24, 2018; date of current version August 7, 2019.

DOI. No. 10.1109/TAES.2018.2875573

Refereeing of this contribution was handled by Z. Davis.

This work was supported in part by the U.S. Defense Threat Reduction Agency under Grants HDTRA1-09-1-0032 and HDTRA1-14-C-0015, in part by the U.S. Air Force Office of Scientific Research under Grant FA9550-09-1-0120, in part by the NSF under Grant 1441214, and in part by the Israeli Centers of Research Excellence (I-CORE) program (Center No. 4/11).

Authors' addresses: A. Das, S. Shirazipourazad, and A. Sen are with the School of Computing, Informatics, and Decision System Engineering, Arizona State University, Tempe, AZ 85287 USA, E-mail: (arun.das@asu.edu; sshiraz1@asu.edu; asen@asu.edu); D. Hay is with the School of Computer Science and Engineering, Hebrew University of Jerusalem, Jerusalem 91904, Israel, E-mail: (dhay@cs.huji.ac.il). *(Corresponding author: A. Das.)*

0018-9251 © 2018 IEEE

I. INTRODUCTION

Target tracking problems (TTPs) address the need of recording or monitoring a set of (mobile) targets of interest for a period of time with the best use of the availability and capabilities of trackers. Consequently, the solution needs of this class of problems usually involve the aspects of tracker allocation and path planning to satisfy the tracking requirements of the target. Although TTPs have been studied fairly extensively by researchers in the recent past, the problem of continuous tracking of mobile targets using the fewest number of mobile trackers, even when the trajectories of all the targets are known in advance, has not received much attention. In this paper, we study such a TTP where the objective is to find 1) the optimal number of mobile trackers (with limited speed) and 2) their trajectories to track a set of targets moving on known trajectories. It may be noted that although in this problem the target trajectories are known, there is no restriction on the nature of such trajectories. In other words, the targets are free to move in any direction at any point of time with any speed.

TTPs with known target trajectories have several applications in both military and civilian environments. For instance, in military environments, unmanned aerial vehicles (UAVs), or trackers, are frequently used for *convoy protection*, such as when providing continuous aerial coverage to a group of ground vehicles, or targets, moving on predetermined paths [1]. In the civilian space, with the recent popularity of rotary-wing UAVs (commonly known as *drones*), such as quadrotor, hexarotor, and octarotor UAVs, several applications of TTPs with known target trajectories have evolved in agriculture, warehousing, distribution, and retail sectors. For instance, modern warehouse infrastructures, such as the ones owned by Amazon, contain more than 15 miles of conveyor belts to move inventory [2] and commercial drones have been proposed as a viable option for inventory management of items (targets) in such large warehouses [3]. Target tracking applications have been proposed in the retail sector as well for optimally managing the use of shelf space in stores. For example, Cleveland *et al.* [4] propose target tracking techniques using mobile robots to address the problem of visual inspection of shelves in pharmacies and grocery stores to allow for better management of a store's retail space.

From the above scenarios, it may be observed that not only does the duration of time for which a target needs to be tracked depends on the application, but also depending upon the environment, the target trajectories may be known in advance. For instance, in a convoy protection exercise, the ground vehicles moving on a predetermined route may need to be tracked for the entire duration of the journey from the source to the destination. Also, to track warehouse inventory moving on conveyor belts, a tracker, such as a drone-mounted radio-frequency identification (RFID) tag reader, may just need to perform a fly-by of a target to record the item and move on to the next piece of inventory. In light of the different tracking requirements of different classes of targets, in this paper we consider two target classes: one

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

requiring tracking for the entire period of observation (Class I type targets), and the other requiring tracking at least once during the entire period of observation (Class II type targets). With this target classification, we formulate two TTPs in an operating environment where target trajectories are known in advance. In this setting, the first problem is formulated for an environment where all mobile targets are of Class I, while the second problem is formulated for an environment where all mobile targets are of Class II.

It may be noted that in our problem domain, the number of targets and their trajectories are not restricted, i.e., there are no restrictions on the number of targets in the deployment space, or the target speed, or the target movement trajectories. The objective of both the problems studied in this paper is to find 1) the minimum number of trackers with limited speed and 2) the trajectories of these trackers necessary to track all targets for the duration of time as determined by the class of the target. To the best of our knowledge no existing studies have addressed the TTP with the same objective as in this paper.

The rest of the paper is organized as follows. In Section II a summary of related works is presented, in Section III the TTPs are formulated and their complexity is proved. In Section IV we present the solution techniques to the problem, and in Section V we present our experimental results. Finally, we conclude the paper in Section VI.

II. RELATED WORK

Motivated by the importance of target tracking in military and civilian environments, considerable research has been conducted on TTPs using UAVs and mobile sensors [1], [5]–[11]. Most of the TTPs studied using sensor networks consider sensor nodes (trackers) to be static and the issues for path planning of trackers are not considered. For instance, Fu *et al.* [12] formulate the multi-TTP using wireless sensor networks as a sensor assignment problem where static sensors are dynamically selected and allocated to track each of the targets. A survey of such studies can be found in [13].

Typically, solving the TTP using mobile trackers (UAVs) involves two components: 1) estimation of target positions using sensor data and 2) mobility management of trackers (sensors). Most of the studies on target tracking using mobile sensors focus on the quality of detection of the mobile targets with a given set of UAVs (mobile sensors). In such studies, one or more mobile trackers are used to track single or multiple targets. Zou and Chakrabarty [9] study the distributed mobility management of a given set of mobile sensors for better tracking of a single target. Enyang *et al.* [8] study a scenario when there is one mobile sensor, a set of static sensors, and a mobile sensor controller. The controller receives the data from the sensors and estimates the location of the target and directs the mobility of the sensors. Oh *et al.* [11] propose an active sensing and guidance algorithm in a multitarget, multitracker environment to maximize target estimation accuracy while factoring in the physical constraints of the system such as the turning radius, speed, and sensing range of a UAV. Adamey and Ozguner [14] study multitarget tracking using multiple UAVs, and develop a decentralized approach for target location estimation and UAV mobility management. Farmani *et al.* [7] study a decentralized multitarget tracking system for cooperative UAVs that have limited sensing capabilities and use extended Kalman filters to estimate the location of mobile targets. Oliveira *et al.* [6] propose a path-generation algorithm for the study of the moving path following problem that has applications in target tracking of single or multiple ground-based targets using an UAV flying at a constant altitude.

In spite of extensive studies on TTPs, there exists only a handful of studies, such as in [1], [5], [15], and [16], that address the problem of tracking mobile targets with an optimal number of mobile trackers. Ding *et al.* [1] study the problem of convoy protection, or providing continuous aerial coverage to a group of unmanned ground vehicles (UGVs) using UAVs where the UAV's kinematics are modeled as Dubins vehicles [17]. The authors consider a scenario where the UGVs are mobile on a straight line with constant speed, and using a single UAV, strategies are proposed to ensure periodic "meet-ups" with the UGVs and establish the bounds on the speed of the UAV that guarantees continuous coverage of the mobile UGVs. Zorbas *et al.* [5] study energy efficiency issues related to mobile target tracking and provide a power efficient target tracking solution by adjusting the UAVs position (altitude). Srinivas *et al.* [15] study a problem in which their goal is to find the smallest set of mobile backbone nodes such that the mobile regular nodes are always under coverage of at least one backbone node. However, in the studies of Zorbas *et al.* [5] and Srinivas *et al.* [15] they assume that trackers have the capability of moving at infinite speeds. This assumption of unbounded speed may be considered unrealistic as the physical limitations of trackers are ignored in this case. Radhakrishnan and Saripalli [16] study the problem of finding the optimal set of UAVs, assuming that the UAVs have finite speed. They propose an algorithm in which they first compute the solution to the coverage problem for each time instance (a static disk coverage problem) using a greedy algorithm. The authors then propose a motion assignment algorithm that determines the movement of the UAVs from their current positions to their next positions. This two-step greedy, and motion assignment algorithm proposed by the authors does not guarantee the optimality of the overall solution to the problem.

As outlined in the previous section, in this paper, we study the TTP where the objective is to find 1) the minimum number of trackers with limited speed and 2) the trajectories of these trackers necessary to track all targets for the duration of time determined by the class of the target. To the best of our knowledge no existing studies have addressed this problem. It may be noted that preliminary results of this study were presented by Hay *et al.* [18].

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

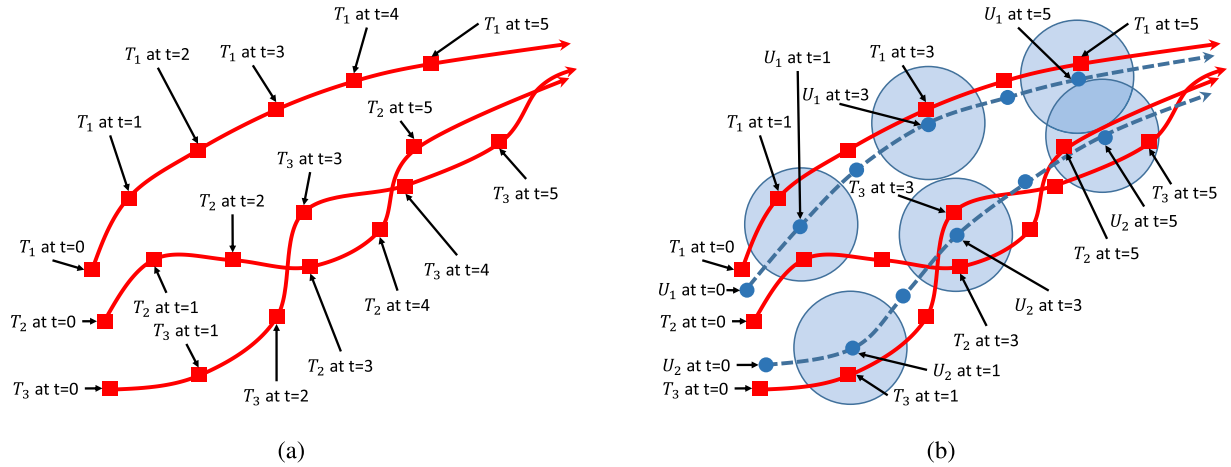


Fig. 1. Problem instance of TTP_1. (a) Targets and their trajectories. (b) Trackers and their trajectories for tracking all the targets of Fig. 1(a) during the entire period of observation [0–5]. Reprinted by permission from Springer Nature: Springer International Publishing, in *Proc. Int. Conf. Combinatorial Optimization and Applications* [18] © 2014.

III. PROBLEM FORMULATION

The goal of the TTP is to find the fewest number of mobile trackers and their trajectories that are sufficient to track all mobile targets within a specified period of time. The input to the TTP is as follows:
GIVEN:

1) n targets and their trajectories in a two (three) dimensional 2-D (3-D) space,
2) the maximum speed of the trackers, denoted by d,
3) the uniform sensing range of the trackers, denoted by r,
4) the period of observation [0–$\mathcal{T}$].

We assume that a tracker can track (sense) a target if the target is within the sensing range r of the tracker. For this paper, we consider two classes of mobile targets: 1) targets that require tracking for the *entire duration of observation* (Class I) and 2) targets that are required to be tracked *at least once* during the period of observation (Class II). Depending on the class of the target, we study two different versions of the TTP:
TTP_1: We refer to the TTP problem where all targets belong to Class I as TTP_1. The objective of TTP_1 is to find the minimum number of mobile trackers and their trajectories required to track all the targets during the entire period of observation.
TTP_2: We refer to the TTP problem where all targets belong to Class II as TTP_2. The objective of TTP_2 is to find the minimum number of mobile trackers and their trajectories required to track all the targets at least once during the period of observation.

Fig. 1 elucidates the objective of TTP_1 with an example. In Fig. 1(a), the trajectories of the targets are shown with red lines, and the location of the targets at specific instances of time are shown with red squares on the trajectories. The velocities of the targets across their trajectories (which may not be uniform) are known, and the period of observation for this example is [0–5]. As shown in Fig. 1(b), only two trackers (UAVs) are required to track all three targets for the entire observation period. In Fig. 1(b), the tracker locations are shown with blue circles, tracker trajectories are shown with blue dashed lines, and the sensing ranges of the trackers are shown with the larger blue circles (for clarity of the figure, not all sensing ranges of the trackers are shown).

Similarly, the objective of TTP_2 is elaborated with the help of an example. Fig. 2(a) shows the trajectories of six mobile targets (shown with thin colored lines) on a 2-D deployment area during the observation period [0–14]. For this example, it is assumed that 1) the maximum speed of the available trackers are 1 unit/s ($d = 1$), 2) the tracker sensing range is 1 unit ($r = 1$), i.e., a target can be tracked only when the distance between a tracker and the target is at most 1 unit, and 3) all targets must be tracked by a tracker at least once within the period of observation [0–14] ($\mathcal{T} = 14$). As shown in Fig. 2(b), for this example, only one mobile tracker (UAV) is sufficient to track all the targets of the constructed TTP_2 instance. The trajectory of the tracker is shown with a thick red line in Fig. 2(b). The locations where a target is tracked (the *rendezvous points*) are shown with black rectangles in the figure. For instance, at time $t = 0$, when tracker U_1 is at location $(3, 7)$, U_1 tracks target T_1 that is at location $(4, 7)$ at time $t = 0$. Similarly, when U_1 is at location $(7, 9)$ at time $t = 14$, U_1 tracks T_6 which is at location $(8, 9)$ at time $t = 14$.

THEOREM 1 Both versions of TTP are NP-complete.

PROOF Given a set of points in the plane, the geometric disk cover (GDC) problem asks to identify a minimally sized set of disks (of prescribed radius) that covers all the points. In both TTP_1 and TTP_2, the period of observation is given to be [0–$\mathcal{T}$]. If we consider a special case of these problems where $\mathcal{T} = 0$, then the period of observation becomes a single instance of time and at this instance of time, all targets can be viewed as being stationary (points on a plane). In this

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

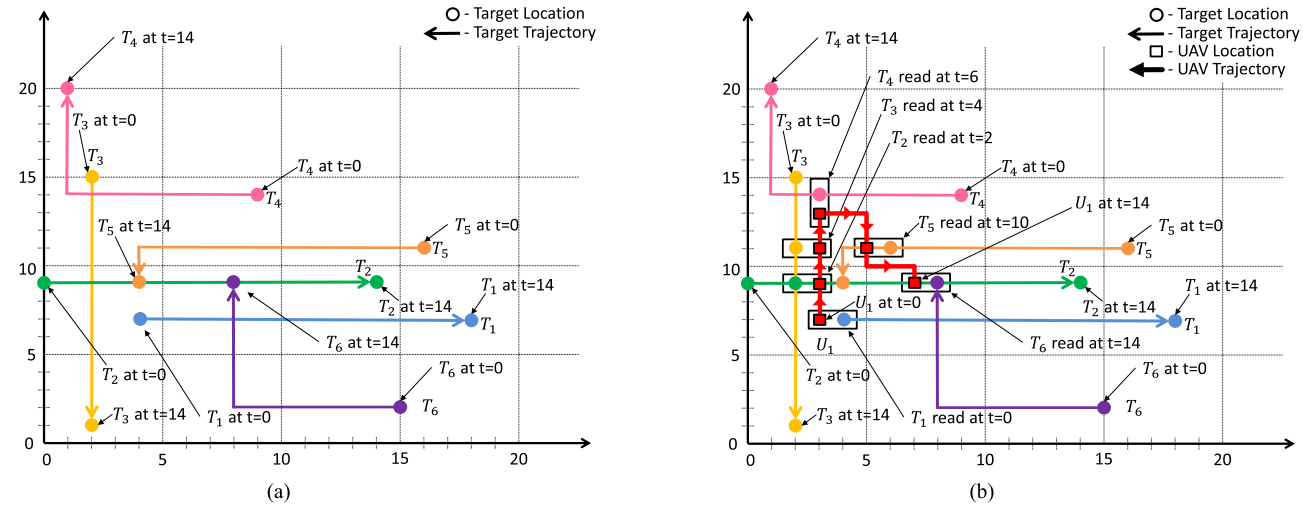


Fig. 2. Problem instance of TTP_2. (a) Location of targets $T_1, \ldots, T_6$ and their trajectories during the observation period [0–14]. (b) Single tracker (U_1) is sufficient to track all targets of Fig. 2(a) at least once during the observation period [0–14]. The trajectory of U_1 is shown with a thick red line and the *rendezvous points* where the target is tracked are shown in black rectangles.

scenario, the trackers can also be viewed as being stationary. Since the sensing range of the tracker is circular with radius r, the goal of the TTP problem now becomes equivalent to that of the GDC problem. Since the GDC problem is a known NP-complete problem [19], we can conclude that both TTP_1 and TTP_2 are NP-complete problems. □

It is only conceivable that when both the targets and trackers are mobile, computation of the minimum number of trackers and their required trajectories will be considerably harder than the scenario when both targets and trackers are stationary.

IV. SOLUTION TECHNIQUE

In this section, we propose our techniques to compute the solution of both versions of the TTP. In the interest of simplicity, we focus our discussion on movement of the targets and trackers on a 2-D (1-D) space. However, the techniques presented can be extended to higher dimensions.

To begin, let $|p_1 \sim p_2|$ denote the Euclidean distance between two points p_1 and p_2 in the 2-D (1-D) space. In TTP_1, the goal is to find the smallest set of trackers $\mathcal{B}$, such that the following constraints are satisfied:

1) *Coverage constraint*$_1$: For any target a_i, and for all time instances $t, t = 0, \ldots, \mathcal{T}$, there is a tracker $b_j \in \mathcal{B}$ whose location at time t, denoted by $p(b_j, t)$ is within distance r from $p(a_i, t)$; i.e., $|p(a_i, t) \sim p(b_j, t)| \leq r$, and

2) *Mobility constraint*$_1$: For any tracker b_j with specified velocity d and any time step $t \geq 1$, $|p(b_j, t-1) \sim p(b_j, t)| \leq d$.

Similarly, for TTP_2 the goal is also to find the smallest set of trackers $\mathcal{B}$, such that:

1) *Coverage constraint*$_2$: Corresponding to any target a_i, there exists at least an instance of time t_i within the period of observation [0–$\mathcal{T}$] ($0 \leq t_i \leq \mathcal{T}$) and a tracker b_j, such that the distance between a_i and b_j at time instance t_i is at most r, i.e., $|p(a_i, t_i) \sim p(b_j, t_i)| \leq r$.

2) *Mobility constraint*$_2$: For any tracker b_j with specified velocity d and any time step $t \geq 1$, $|p(b_j, t-1) \sim p(b_j, t)| \leq d$.

To solve both versions of the TTP problem we follow a four-step process. An overview of these steps is highlighted below:

1) *Step 1*: We discretize the observation time into equal intervals of length δ, and the observation space into equal units of size ε. It may be noted that in a continuous setting there are an infinite number of points in the observation space where a tracker can be present as time varies from the start to the end of the observation period. With this discretization, the number of locations where a tracker can be present is made finite. Although trackers are restricted to be present in one of these finite locations, no such restriction is imposed on the targets.

2) *Step 2*: From this discretized space, given the target trajectories and the tracker's sensing radius r, for each time step, a set of *potential points* is computed for each target. At time t, the set of potential points for a target a_i that is at location p is the set of all points in the discretized space that is within distance r to p, i.e., the set of all points in the discretized space where if a tracker were to be present at time t it would be able to track a_i.

3) *Step 3*: We construct a TTP graph where the potential points for every target, for every time step, are represented as nodes of the graph. In addition to including two additional nodes (source and destination nodes), edges are introduced in the graph in accordance to the mobility (speed) available to the trackers.

4) *Step 4*: A generalized version of the minimum flow problem (MFP) [20] is solved on the constructed graph to obtain the solution to the TTP.

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

In subsequent sections, we elaborate on each of the above steps in further detail. As the generalization of the MFP as well as the graph construction mechanism are different for TTP_1 and TTP_2, we describe the graph construction and the generalization process in section IV-B (IV-E) and IV-C (IV-F), respectively for TTP_1 (TTP_2). As both the generalizations are based on the classical minimum flow problem, for the sake of completeness, we state the problem as follows:

Definition *MFP:* Given a capacitated network $G = \langle V, E\rangle$ where a nonnegative capacity $c(i, j)$, and a nonnegative lower bound $l(i, j)$ is associated with each edge (i, j), and two special nodes—a source node S and a sink node D, a flow is defined to be a function $f : E \to \mathbb{R}^+$ satisfying the following conditions:

$$\sum_{j\in V} f(i,j) - \sum_{j\in V} f(j,i) = \begin{cases} F, & i = S \\ 0, & i \neq S, D \\ -F, & i = D \end{cases}$$

$$l(i,j) \leq f(i,j) \leq c(i,j) \tag{1}$$

for some $F \geq 0$ where F is the value of the flow f.

The MFP finds a flow f which minimizes F.

A. Discretization of the Problem Instance

Although the TTP is a *continuous time domain* problem (as the mobile targets and trackers can be anywhere in the deployment area at a given time), our approach to solving the TTP is to discretize both time and space. In our technique, time is discretized into equal intervals of length δ, and space into equal *units* of size ε. Depending on the spatial dimension of the problem instance, i.e., 1-D, 2-D, or 3-D, the unit will imply an *interval* of length ε, a *square* with length of each side ε, and a cube with length of each side ε, respectively. It may be noted that although in the following sections we focus our discussion on a 2-D (1-D) space, the proposed techniques are applicable to higher dimensions as well.

Let $\mathcal{A} = \{a_0, \ldots, a_{n-1}\}$ be a set of n targets moving on a 2-D space over time instances $t = 0, \ldots, \mathcal{T}$. Also, let $p(a_i, t) = (x(a_i, t), y(a_i, t))$ be the location of target $a_i \in \mathcal{A}$ at time instance t, where $x(a_i, t)$ and $y(a_i, t)$ denote the x-coordinate and y-coordinate of a_i at time t. We assume that a target is covered by a tracker b_j if the distance between a_i and b_j is less than the sensing radius r.

In the graph construction step, for TTP_1 (TTP_2) discussed below, we construct a directed graph $G_1 = \langle V_1, E_1\rangle$ ($G_2 = \langle V_2, E_2\rangle$) where the vertex set V_1 (V_2) represents the locations from where the targets can be tracked, and the edge set E_1 (E_2) represents the possible tracker movements or tracker presence. As shown in Fig. 3(a), for the 2-D space, we discretize the deployment space into a *grid* of squares where the length of each side of a square is ε, and restrict the locations where a tracker can be present to the intersection points of the constructed grid. It may be noted that the possible locations a tracker can be present depends on the granularity of the grid determined by the space discretization parameter ε. Let $\mathcal{N}$ be the set of all points in the grid for the 2-D space, namely, $\mathcal{N} = \{i \cdot \varepsilon, j \cdot \varepsilon \mid i, j \in \mathbb{Z}\}$. We note that for a target a_i to be tracked by a tracker at time instance t, there should be at least one tracker in the disk of radius r centered at $p(a_i, t)$. Let $D(p(a_i, t), r)$ denote that disk. As shown in Fig. 3(a), the red point represents the location of target a_i at time t, while the blue points show the $\mathcal{N} \cap D(p(a_i, t), r)$ *potential points* in disk $D(p(a_i, t), r)$ where a tracker can be located to track target a_i at time t. Let $\mathcal{P}_{a_i,t}$ denote the set of potential points within radius r for target a_i at time t, i.e., $\mathcal{P}_{a_i,t} = \mathcal{N} \cap D(p(a_i, t), r)$.

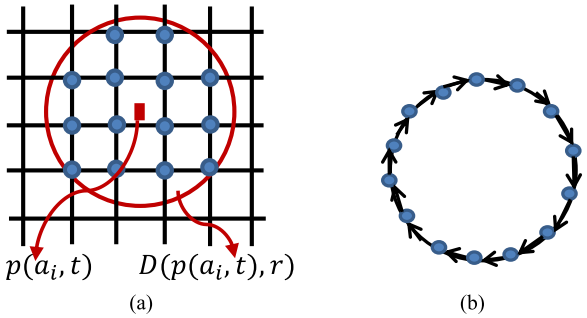


Fig. 3. (a) Disk centered at the location of target a_i at time t. (b) Ring corresponding to discrete points in the disk shown in (a). Reprinted by permission from Springer Nature: Springer International Publishing, in *Proc. Int. Conf. Combinatorial Optimization and Applications* [18] © 2014.

Using the set of all potential points for every target at every time step, in the following sections we discuss the detailed graph construction steps for TTP_1 and TTP_2 to model the problems as flow networks. We then find the solution to the flow networks by solving generalized versions of the minimum flow problem on the constructed graphs.

B. TTP_1 Graph Construction

To construct the edge-capacitated directed graph $G_1 = \langle V_1, E_1\rangle$, for every potential point $p_j \in \mathcal{P}_{a_i,t}$, for target $a_i \in \mathcal{A}$, and for each time step $t = 0, \ldots, \mathcal{T}$, we add a vertex to V_1 that is represented by the triplet $\langle location, target\ id, time\ step\rangle$, where *location* corresponds to the (x, y) coordinate of the point p_j, *target id* corresponds to a_i, and *time step* corresponds to t. Note that this construction allows the same (x, y) coordinate to be added multiple times for different targets and/or time steps. In addition to these vertices, we also add one supersource vertex S and one supersink vertex D. Thus, the set of vertices in G_1 is

$$V_1 = \{S, D\} \cup \bigcup_{p_j \in \mathcal{P}_{a_i,t}} \langle p_j, a_i, t\rangle$$
$$\forall t = 0, \ldots, \mathcal{T}, i = 0, \ldots, n-1. \tag{2}$$

To construct the edge set E_1, four types of edges are introduced in G_1:

1) *Intratarget edges:* For every target a_i and time step t we construct a *directed ring* connecting all vertices represented by $\langle location, target\ id, time\ step\rangle$, that have target

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

id a_i and time step t [see Fig. 3(b)]. In other words, a directed ring is constructed with all the vertices introduced in G_1 that were representative of the set of potential points $\mathcal{P}_{a_i,t}$ of disk $D(p(a_i,t),r)$. We note that the order of the nodes in the ring is arbitrary. For all such intratarget edges, the demand and capacity of these edges are set to 1.

2) *Mobility edges:* The mobility edges represent whether a tracker at point p_b at time step t can be at (move to) point p_c at time step t' ($t \le t' \le \mathcal{T}$). Note that whether the tracker can be at point p_c from point p_b by time t' depends on the available tracker speed d. Formally, the edge $(\langle p_b, a_i, t\rangle \rightarrow \langle p_c, a_j, t'\rangle)$ is introduced in E_1, iff $t' \ge t$ and $|p_c \sim p_b| \le d \times (t' - t)$. It may be noted that a_i and a_j may refer to the same or different targets. For each such edge the capacity is set to 1 and demand set to 0. The direction of these edges is oriented from the node with the lower time step to the node with the higher time step, and in case of a tie it is broken by the nodes' target ids.

3) *Supersource edges:* The supersource edges connect the supersource node S to all vertices represented by $\langle location, target\,id, time\,step\rangle$ in $G_1 = \langle V_1, E_1\rangle$. The capacity of these edges are set to 1 and demand is set to 0. Note that these edges originate at the supersource.

4) *Supersink edges:* The supersink edges connect all vertices represented by $\langle location, target\,id, time\,step\rangle$ to the supersink node D in $G_1 = \langle V_1, E_1\rangle$. The capacities of these edges are set to 1 and demand set to 0. Note that these edges terminate at the supersink.

C. Generalized Minimum Flow Problem for TTP_1

To solve TTP_1, we modify the minimum flow problem (Section IV) such that the lower bound constraints on the intratarget edges in graph $G_1 = \langle V_1, E_1\rangle$ are satisfied iff the flow starts from source node S. Therefore, we add additional constraints to the minimum flow problem to ensure there exists a path from S to every node in $V_1 \setminus \{S, D\}$ such that the flow f on the edges of this path is 1. Let g_u be a flow function defined with respect to node $u \in V_1 \setminus \{S, D\}$. It may be noted that if there is a path from S to a node in the constructed ring, then there is also a path to all nodes in that ring. Hence, for every ring in G_1, corresponding to a target a_i at time instance t, an arbitrary node u of the form $\langle p_j, a_i, t\rangle$ is selected and the following constraints are added to the minimum flow problem:

$$\sum_{j\in V_1} g_u(i,j) - \sum_{j\in V_1} g_u(j,i) = \begin{cases} 1, & i = S \\ 0, & i \ne S, D \\ -1, & i = D \end{cases}$$

$$g_u(i,j) \le f(i,j). \tag{3}$$

D. Solution to TTP_1

In this section, we first provide an example of the TTP_1 graph construction process (Section IV-B). We then show how solving the generalized minimum flow problem (Section IV-C) on the constructed graph finds the solution to the TTP_1. For the sake of simplicity, we explain the solution process with the help of an instance of TTP_1 (as shown in Fig. 4) where all the three targets are restricted to move in a 1-D space (line).

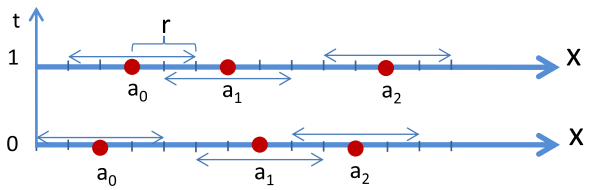


Fig. 4. Locations of three targets (a_0, a_1, a_2) over two time steps in a 1-D space. Reprinted by permission from Springer Nature: Springer International Publishing, in *Proc. Int. Conf. Combinatorial Optimization and Applications* [18] © 2014.

Fig. 4 shows the locations of the targets on the x-axis for two time steps, i.e., $\delta = 1$, and the observation period is [0–1]. Also, in this example $r = 2$, $\varepsilon = 1$, and $d = 1$. Fig. 5 shows the graph $G_1 = \langle V_1, E_1\rangle$ constructed following the rules given in Section IV-B. For clarity of the figure, not all edges of E_1 are shown in Fig. 5. The number on each vertex of G_1 shows a potential point on the x-axis.

Next, we explain how the solution of the generalized minimum flow problem on the constructed graph $G_1 = \langle V_1, E_1\rangle$ finds the solution to the TTP_1.

It may be noted that a tracker covers a target a_i at time step t, only if the tracker is located at a potential point p_j, such that $p_j \in \mathcal{P}_{a_i,t}$, $\forall t, 0 \le t \le \mathcal{T}$. We prove that 1) the minimum number of trackers required to cover all targets at all instances of time is equal to the value of generalized minimum flow in graph $G_1 = \langle V_1, E_1\rangle$ and 2) the trajectories of the trackers can be computed from the solution of the generalized minimum flow problem.

For the TTP_1 example of Fig. 4, the blue edges in Fig. 5 show the edges with a flow value of 1 computed using the generalized minimum flow solution (see Section IV-C). It can be seen that for this example, the value of the generalized minimum flow is 2 implying that two trackers are sufficient to track the three targets of example Fig. 4 for two time steps. It can be interpreted from the flow that the first tracker at time step $t = 0$ is at location $x = 3$ ($x(a_0, 0) + \varepsilon = 3$) covering target a_0, and moves to $x = 4$ at time step $t = 1$ covering both a_0 and a_1 at ($x(a_0, 1) + \varepsilon = 4$ and $x(a_1, 1) - 2\varepsilon = 4$). The second tracker at time step $t = 0$ is at location $x = 9$ covering targets a_1 and a_2, where $x(a_1, 0) + 2\varepsilon = 9$ and $x(a_2, 0) - \varepsilon = 9$, and moves to $x = 10$, ($x(a_2, t_1) - \varepsilon = 10$) at time step $t = 1$ covering a_2.

LEMMA 1 Any valid flow with value F on the constructed graph G_1 can be decomposed to a valid assignment of F trackers that solves the TTP_1.

PROOF The decomposition proceeds by first assigning a tracker to a supersource edge with flow 1. Then, for the node $\langle p_j, a_i, t\rangle$ to which this supersource edge is connected, a tracker b is added to $\mathcal{B}$ and is assigned location p_j at time step t. If the intratarget edges of target a_i at time step t have a flow of value 1, the intratarget edges are traversed, which can be interpreted as tracker b tracking target a_i at

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

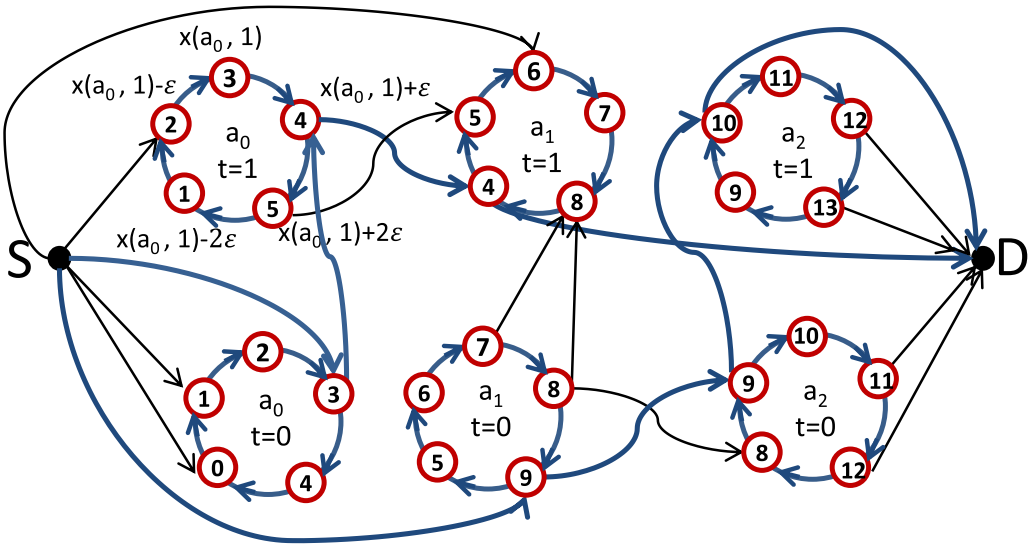


Fig. 5. Solution to the TTP_1 for generalized minimum flow problem of the example in Fig. 4. Reprinted by permission from Springer Nature: Springer International Publishing, in *Proc. Int. Conf. Combinatorial Optimization and Applications* [18] © 2014.

time step t. As the intraedges of target a_i at time step t are constructed as a ring, the flow traverses all the vertices on the ring and returns to $\langle p_j, a_i, t\rangle$. Next, from the outgoing edges of vertex $\langle p_j, a_i, t\rangle$ the edge with flow 1 is selected for traversal. Such an edge from $\langle p_j, a_i, t\rangle$ necessarily exists due to the flow conservation rules of the generalized minimum flow (3). Let the destination of this outgoing edge be $\langle p_{j'}, a_{i'}, t'\rangle$. Depending on t' there are two cases. 1) If $t' = t$, then $p_{j'} = p_j$, but $i \neq i'$, implying that the tracker b at location p_j and time t not only covers (tracks) target a_i but also target $a_{i'}$; hence, $a_{i'}$ is added to the coverage of b at time t and the traversal progresses similarly from vertex $\langle p_{j'}, a_{i'}, t'\rangle$ and the intratarget edges of target $a_{i'}$ at time t'. 2) If $t' > t$, this implies that tracker b moves from position p_j at time t to position $p_{j'}$ at time step t'. This movement is a legal move for the tracker and is enforced by the mobility edges added to the graph adhering to $|p_{j'} \sim p_j| \leq d \times (t' - t)$. The edge traversal then progresses similarly from $\langle p_{j'}, a_{i'}, t'\rangle$ and the intratarget edges of target $a_{i'}$ at time t' until the supersink node D is reached. Upon this traversal from S to D, the trajectory of tracker b can be constructed and the targets that b covers can also be mapped. To repeat the process for other required trackers, the flows that b covers in the graph are removed and the decomposition process continues until there are no outgoing flows from S. At the end of this process, it is ensured that every target in $\mathcal{A}$ is covered at all time steps as the valid flows from S to D implies that all intratarget edges have a flow of 1 that meets the demand of all targets. This decomposition thus ensures that the trajectories of all the required trackers will be computed that will satisfy the coverage and mobility constraints of TTP_1, and every target will be tracked for every time step by a tracker in the decomposition of F. Since one tracker is assigned to each supersource edge of flow 1, the tracker assignment will have exactly F trackers. □

LEMMA 2 Let F represent the solution to the generalized minimum flow problem on the graph constructed from the instance of TTP_1. Then, any valid assignment $\mathcal{B}$ of F trackers with maximum velocity $d - \sqrt{2}\varepsilon$ and sensing radius $r - \frac{\sqrt{2}}{2}\varepsilon$ in the 2-D space can be represented as a valid flow with value F on the graph constructed from the instance of TTP_1.[1]

PROOF In the continuous setting, every tracker b_m starts its trajectory at some time step t, at location p_j, covering a set of targets within distance r from p_j. Let this set of targets be $A_{t,j,m}$. In the discretized space, let $p_{j'}$ be the closest grid point to p_j, i.e., in the 2-D plane, $p_{j'}$ is at most distance $\frac{\sqrt{2}}{2}\varepsilon$ of p_j. Thus, as all targets in $A_{t,j,m}$ are within distance $r - \frac{\sqrt{2}}{2}\varepsilon$ of p_j, they are within distance r of $p_{j'}$. Next, tracker b_m moves to a new location p_k at time step $t', t' > t$ to cover a set of targets $A_{t',k,m}$. In the discretized space, let $p_{k'}$ be the closest grid point to p_k. We can represent the trajectory of b_m by a flow of one unit in G_1 in the following way. Corresponding to the tracker b_m, we add a flow of one unit to G_1 leaving S to the node $\langle p_{j'}, a_i, t\rangle$ where $a_i \in A_{t,j,m}$ and a_i has the lowest target id among other targets in $A_{t,j,m}$. $\langle p_{j'}, a_i, t\rangle$ exists in V_1 as a_i is within distance r of $p_{j'}$ and $p_{j'} \in \mathcal{N}$. The flow traverses the ring corresponding to a_i at time step t satisfying the demand on the intratarget edges and it leaves the ring from the same node $\langle p_{j'}, a_i, t\rangle$. Next, the flow enters the node $\langle p_{j'}, a_{i'}, t\rangle$ in the ring corresponding to the next target $a'_i \in A_{t,j}$, where target id $i' > i$. The flow traverses the ring and this process continues till all the rings corresponding to the targets in $A_{t,j,m}$ at time step t are traversed by this flow (all the edges in these rings have a flow of one unit). We note that if a target is covered by more than one tracker at a time instance, just one of them is considered to be in charge of covering

[1]This result can be extended to higher dimensional spaces.

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

the target at that time instance. In other words, a flow of one unit is selected to traverse every ring. Hence, the flow value on the intratarget edges does not exceed the capacity. We also note that for every two targets $a_i, a_{i'} \in A_{t,j,m}$ at time t, there is a mobility edge from $\langle p_{j'}, a_i, t\rangle$ to $\langle p_{j'}, a_{i'}, t\rangle$ (due to the graph construction rules), where the flow goes from one ring to the next ring through this edge. Next, through a mobility edge, the flow enters the node $\langle p_{k'}, a_q, t'\rangle$, where $a_q \in A_{t,k,m}$. Since, the tracker is able to move from location p_j to p_k between time steps t and t' and its velocity is bounded by $d - \sqrt{2}\varepsilon$, the distance between p_j and p_k is at most $(d - \sqrt{2}\varepsilon) \times (t' - t)$ and hence the distance between $p_{j'}$ and $p_{k'}$ is at most $\frac{\sqrt{2}}{2}\varepsilon + (d - \sqrt{2}\varepsilon) \times (t' - t) + \frac{\sqrt{2}}{2}\varepsilon \leq d(t' - t)$, implying that there is a mobility edge $(\langle p_{j'}, a_{i'}, t\rangle \rightarrow \langle p_{k'}, a_q, t'\rangle)$ in G_1. Then, the flow continues similarly based on the tracker trajectory, and at the time that tracker b_m is not required to track any further targets, the flow proceeds toward the supersink node D. It can be seen that the flow constraints on all the edges are satisfied. Also, since we add a flow of one unit to the graph from S to D for every tracker, the total flow will exactly be F units.□

Lemmas 1 and 2 establish that the solution of the generalized minimum flow (see Section IV-C) on the constructed graph G_1 yields the minimum number of trackers required to track all targets. Moreover, the trajectories of all the trackers can be computed from the solution as it provides the location of each of the trackers at different instances of time.

E. TTP_2 Graph Construction

To model the TTP_2 as a flow network we construct an edge-capacitated directed graph $G_2 = \langle V_2, E_2\rangle$ from the set of all potential points $\mathcal{P}_{a_i,t}, a_i \in \mathcal{A}, t = 0, \ldots, \mathcal{T}$. It may be recalled that a potential point $p_j \in \mathcal{P}_{a_i,t}$ represents a (x, y) point on a 2-D plane where a tracker can be present at time t to cover (track) target a_i. Thus, from every potential point $p_j \in \mathcal{P}_{a_i,t}$, a tuple of the form $\langle location, time\rangle$ can be constructed that represents a location and time where a tracker can be present to track target a_i. Let the set LT_i denote the set of all such $\langle location, time\rangle$ tuples for target a_i, formally as follows:

$$LT_i = \bigcup_{p_j \in \mathcal{P}_{a_i,t}} \langle p_j, t\rangle, \forall t = 0, \ldots \mathcal{T}. \tag{4}$$

It may be noted that LT_i and $LT_{i'}, i \neq i'$ may not be disjoint, i.e., two different targets a_i and $a_{i'}$ at time t may share a potential point. In other words, they may produce the same $\langle location, time\rangle$ tuple. This implies that both targets a_i and $a_{i'}$ can be tracked from the same location and time as specified by the $\langle location, time\rangle$ tuple. Let the set $\mathcal{LT}$ be the set of all distinct $\langle location, time\rangle$ tuples across all targets, i.e., $\mathcal{LT} = \bigcup_{i=0}^{n-1} LT_i$.

To construct graph $G_2 = \langle V_2, E_2\rangle$, two types of vertices are introduced in the following way. For every $\langle location, time\rangle$ tuple in $\mathcal{LT}$, a *location* type node identified by $L_{\text{location,time}}$, and a *time* type node identified by $T_{\text{location,time}}$ is introduced. In addition to these vertices, we also add one supersource vertex S and one supersink vertex D. Thus, the set of vertices in $G_2 = \langle V_2, E_2\rangle$ is

$$\forall \langle i, j\rangle \in \mathcal{LT}, V_2 = \{S, D\} \cup L_{i,j} \cup T_{i,j}. \tag{5}$$

To construct the edge set E_2, four types of edges are introduced in $G_2 = \langle V_2, E_2\rangle$:

1) *Location-time edges:* For every location node $L_{i,j}$ and time node $T_{i,j}$, a directed edge of the form $(L_{i,j} \rightarrow T_{i,j})$ is introduced in $G_2 = \langle V_2, E_2\rangle$. It may be noted that the number of such edges will be equal to the cardinality of $\mathcal{LT}$, as for each distinct $\langle location, time\rangle$ tuple in $\mathcal{LT}$, a location node $L_{\text{location,time}}$, a time node $T_{\text{location,time}}$, and an edge $L_{\text{location,time}} \rightarrow T_{\text{location,time}}$ is introduced in G_2. The capacity of these edges is set to 1.

2) *Mobility edges:* For every time node $T_{i,j} \in V_2$, and location node $L_{i',j'} \in V_2$, where i, i' represent locations on the plane, and j, j' represent time steps, a directed edge of the form $(T_{i,j} \rightarrow L_{i',j'})$ is introduced in G_2 iff a) $j' \geq j$, and b) $|i \sim i'| \leq d \times (j' - j)$, where $|i \sim i'|$ represents the distance between locations i and i'. That is, given tracker speed d, a directed edge is introduced from every time node to every location node if the tracker can move to location i' from location i in $j' - j$ time steps where $j' \geq j$.

3) *Supersource edges:* From the supersource node S, for every location node $L_{i,j} \in V_2$, a directed edge of the form $(S \rightarrow L_{i,j})$ is introduced in G_2. The capacity of these edges is set to 1.

4) *Supersink edges:* From every time node $T_{i,j} \in V_2$, a directed edge of the form $(T_{i,j} \rightarrow D)$ to the supersink node D is introduced in G_2. The capacity of these edges are set to 1.

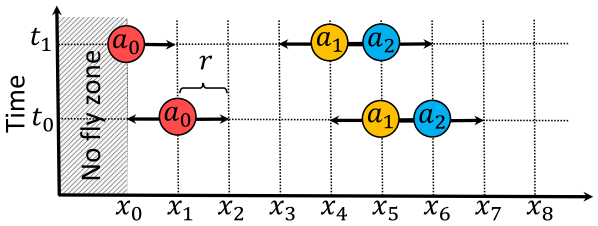


Fig. 6. Locations of three targets on 1-D space (line) at two different instances of time. Interval of time $\delta = t_1 - t_0$ and interval of space $\varepsilon = x_{i+1} - x_i, \forall 0 \leq i \leq 7$.

F. Generalized Minimum Flow Problem for TTP_2

In this section, we explain the G_2 graph construction process with the help of an example, where the movements of the targets are restricted to a 1-D space. It may be noted that although in this example the targets are restricted to move in a 1-D space, the graph construction process described in Section IV-E is also valid for 2-D or 3-D spaces. As shown in Fig. 6, this example has three targets and the observation period comprises of two time steps t_0 and t_1. Target a_0 is at location x_1 at time t_0 and in location x_0 at time t_1. Similarly, target a_1 is in location x_5 at t_0 and in

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

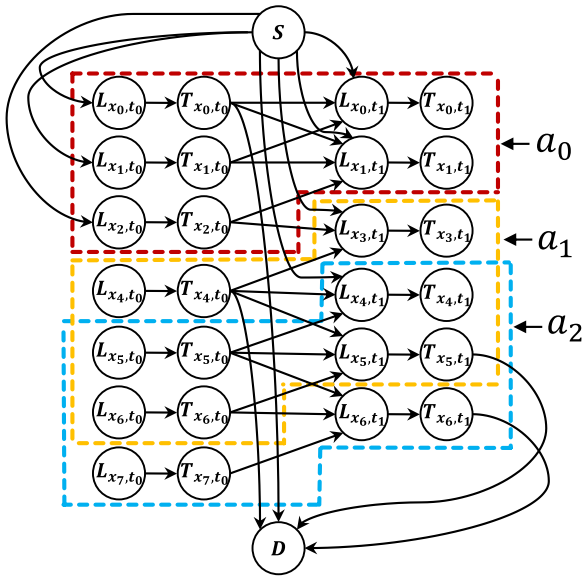


Fig. 7. Graph G_2 constructed from the TTP_2 problem instance of Fig. 6.

location x_4 at t_1, and the target a_2 is at location x_6 at t_0 and in location x_5 at t_1.

In the example of Fig. 6, it is assumed that tracker sensing range $r = \varepsilon$. The LT_i sets containing the $\langle location, time\rangle$ tuples for each target $a_i, i = 0, 1, 2$ is constructed as follows:

$$\begin{aligned} LT_0 &= \{\langle x_0, t_0\rangle, \langle x_1, t_0\rangle, \langle x_2, t_0\rangle, \langle x_0, t_1\rangle, \langle x_1, t_1\rangle\}, \\ LT_1 &= \{\langle x_4, t_0\rangle, \langle x_5, t_0\rangle, \langle x_6, t_0\rangle, \langle x_3, t_1\rangle, \langle x_4, t_1\rangle, \langle x_5, t_1\rangle\}, \\ LT_2 &= \{\langle x_5, t_0\rangle, \langle x_6, t_0\rangle, \langle x_7, t_0\rangle, \langle x_4, t_1\rangle, \langle x_5, t_1\rangle, \langle x_6, t_1\rangle\}. \end{aligned} \tag{6}$$

$$\begin{aligned} \text{The set } \mathcal{LT} &= \bigcup_{i=0}^{2} LT_i \\ &= \{\langle x_0, t_0\rangle, \langle x_1, t_0\rangle, \langle x_2, t_0\rangle, \langle x_4, t_0\rangle, \langle x_5, t_0\rangle, \\ &\quad \langle x_6, t_0\rangle, \langle x_7, t_0\rangle, \langle x_0, t_1\rangle, \langle x_1, t_1\rangle, \\ &\quad \langle x_3, t_1\rangle, \langle x_4, t_1\rangle, \langle x_5, t_1\rangle, \langle x_6, t_1\rangle\}. \end{aligned} \tag{7}$$

For this example, following the graph construction rules of Section IV-E, 13 location nodes $L_{i,j}$ and 13 time nodes $T_{i,j}$ are created in G_2 from every $\langle location, time\rangle$ tuple $\langle i, j\rangle \in \mathcal{LT}$. Additionally, the supersource and supersink nodes S and D are added to G_2. Finally, the edges are introduced in G_2 following the rules from Section IV-E. Fig. 7 shows graph $G_2 = \langle V_2, E_2\rangle$ constructed from the example of Fig. 6. For clarity of the figure, not all edges of E_2 are shown in Fig. 7.

To solve the TTP_2 problem instance of Fig. 6 every target a_0, a_1, and a_2 must be tracked by a tracker at least once at time step t_0 or t_1. From the construction of $\langle location, time\rangle$ tuples for each target, it can be observed that in order to track target a_0, there must be a tracker at $\langle x_0, t_0\rangle$ or $\langle x_1, t_0\rangle$ or $\langle x_2, t_0\rangle$ or $\langle x_0, t_1\rangle$ or $\langle x_1, t_1\rangle$, where $\langle x_i, t_j\rangle$ indicates location x_i at time t_j. From these $\langle location, time\rangle$ tuples, the corresponding $(L_{i,j} \to T_{i,j})$ edges, $(x_0 \to t_0)$, $(x_1 \to t_0)$, $(x_2 \to t_0)$, $(x_0 \to t_1)$, and $(x_1 \to t_1)$, are created in G_2 and are shown within a red bounding box in Fig. 7. These set of $(L_{i,j} \to T_{i,j})$ edges created from the $\langle location, time\rangle$ tuples for each target is labeled as an *edge group*.

Using similar reasoning it can be concluded that in order to track a_1, there must be a tracker at $\langle x_4, t_0\rangle$ or $\langle x_5, t_0\rangle$ or $\langle x_6, t_0\rangle$ or $\langle x_3, t_1\rangle$ or $\langle x_4, t_1\rangle$ or $\langle x_5, t_1\rangle$. The corresponding $(L_{i,j} \to T_{i,j})$ edge group created for a_1 is shown within a yellow bounding box in Fig. 7. Finally, in order to track a_2, there must be a tracker at $\langle x_5, t_0\rangle$ or $\langle x_6, t_0\rangle$ or $\langle x_7, t_0\rangle$ or $\langle x_4, t_1\rangle$ or $\langle x_5, t_1\rangle$ or $\langle x_6, t_1\rangle$. The corresponding $(L_{i,j} \to T_{i,j})$ edge group created for a_2 is shown within a blue bounding box in Fig. 7.

We interpret a flow of one unit on an edge $(L_{i,j} \to T_{i,j})$ as presence of a tracker in location i at time j. Thus, to satisfy the coverage constraint of TTP_2, every edge group of G_2 must have at least one unit of flow. For instance in the constructed graph of Fig. 7, if one unit of flow exists between $(L_{x_0,t_0} \to T_{x_0,t_0})$ and $(L_{x_5,t_0} \to T_{x_5,t_0})$, it can be interpreted that a tracker is present at location x_0 at time t_0 that is able to track a_0, and another tracker at location x_5 at time t_0 that is able to track a_1 and a_2, thus satisfying the coverage constraint of tracking all three targets. As the general minimum flow problem [20] specifies a lower bound on every edge and not on a group of edges, we define a generalized minimum flow problem as follows:

DEFINITION *Generalized minimum flow problem (GMFP):* In the generalized minimum flow problem the lower bound on the flow requirement $l(i, j)$ is no longer associated with an edge (i, j), but associated with a set (group) of edges $E_k \subseteq E$ of the graph $G = \langle V, E\rangle$, and is denoted by l_k. Formally, the problem is stated as follows.

Given: 1) a capacitated network $G = \langle V, E\rangle$ with a nonnegative capacity $c(i, j)$ associated with each edge (i, j), 2) a set of subsets $\mathcal{E}'$ of the edge set E (i.e., $\mathcal{E}' = \{\mathcal{E}_0, \ldots, \mathcal{E}_{n-1}\}$, $\mathcal{E}_k \subseteq E, 0 \le k \le n-1$), 3) a lower bound on the flow requirement l_k associated with each $\mathcal{E}_k, 0 \le k \le n-1$, and 4) two special nodes, a source node S and a sink node D. A flow is defined to be a function $f : E \to \mathbb{R}^+$ satisfying the following conditions:

$$\sum_{j\in V} f(i, j) - \sum_{j\in V} f(j, i) = \begin{cases} F, & i = S \\ 0, & i \ne S, D \\ -F, & i = D \end{cases}$$

$\forall \mathcal{E}_k, 0 \le k \le n-1, \exists l_k$, a lower bound of flow in $\mathcal{E}_k$, implying that there must exist at least one edge $(i, j) \in \mathcal{E}_k$ such that $l_k \le f(i, j) \le c(i, j)$ (8)

for some $F \ge 0$ where F is the value of the flow f. The generalized minimum flow problem is to determine a flow f for which F is minimized. It may be noted that when $|\mathcal{E}_k| = 1, 0 \le k \le n-1$, the GMFP reduces to the minimum flow problem of [20].

G. Solution to TTP_2

We now show how solving the GMFP on the G_2 graph solves TTP_2. First, for the graph $G_2 = \langle V_2, E_2\rangle$ constructed in Section IV-E, the capacity $c(i, j)$ for all edges $(i, j) \in E_2$

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

is set to 1. Second, as noted above, an instance of the GMFP has a set of subsets $\mathcal{E}'$ of the edge set E (i.e., $\mathcal{E}' = \{\mathcal{E}_0, \ldots, \mathcal{E}_{n-1}\}$), where $\mathcal{E}_k \subseteq E, 0 \leq k \leq n-1$, with a lower bound on the flow requirement l_k associated with each $\mathcal{E}_k$. If l_k is the lower bound of flow in $\mathcal{E}_k$, the GMFP requires that there must exist at least one edge $(i, j) \in \mathcal{E}_k$ such that $l_k \leq f(i, j) \leq c(i, j)$. In the graph $G_2 = \langle V_2, E_2 \rangle$, $\mathcal{E}_k$ is set as follows:

$$\mathcal{E}_k = \bigcup_{(L_{i,j} \to T_{i,j}) \in E_2} (L_{i,j} \to T_{i,j}),$$
$$\forall \langle i, j \rangle \in LT_k, 0 \leq k \leq n-1. \tag{9}$$

For the example of Fig. 6, for the set LT_0 given in (6), $\mathcal{E}_1 = \{(L_{x_0,t_0} \to T_{x_0,t_0}), (L_{x_1,t_0} \to T_{x_1,t_0}), (L_{x_2,t_0} \to T_{x_2,t_0}), (L_{x_0,t_1} \to T_{x_0,t_1}), (L_{x_1,t_1} \to T_{x_1,t_1})\}$. The lower bound of flow requirement in $\mathcal{E}_k, 0 \leq k \leq n-1$ is set to 1, i.e., $l_k = 1$. In this example, at least one edge in the edge set $\{(L_{x_0,t_0} \to T_{x_0,t_0}), (L_{x_1,t_0} \to T_{x_1,t_0}), (L_{x_2,t_0} \to T_{x_2,t_0}), (L_{x_0,t_1} \to T_{x_0,t_1}), (L_{x_1,t_1} \to T_{x_1,t_1})\}$ must have a flow of one unit. It is assumed that two trackers cannot be at the same location at the same time. In Fig. 7, the directed edge set $\mathcal{E}_0, \mathcal{E}_1$, and $\mathcal{E}_2$ corresponding to three tags, a_0, a_1, and a_2 are shown enclosed in three rectangular boxes, colored red, yellow, and blue, respectively.

LEMMA 3 Any valid flow of the GMFP of (8) on G_2 provides 1) the minimum number of trackers needed to track all targets at least once during the observation period and 2) provides the required tracker trajectories.

PROOF If the minimum number of trackers needed to track all targets at least once during the observation period is m, $|\mathcal{B}| = m$, then there exists exactly m flows (paths) from the supersource node to the supersink node in G_2 that will satisfy the requirements of (8) for the GMFP. Suppose that the trajectory of tracker $U_i \in \mathcal{B}$ is given by the location–time pairs $(L_{p_1,q_1}, T_{p_1,q_1}), (L_{p_2,q_2}, T_{p_2,q_2}), \ldots, (L_{p_k,q_k}, T_{p_k,q_k})$ denoting that tracker U_i started its traversal at location p_1 at time q_1, then moved to location p_2 at time q_2 and so on, finally to end its traversal at location p_k at time q_k. In the constructed G_2 graph, a flow of one unit (or a path) originating from the supersource node S and terminating at the supersink node D can be constructed in the following way: $S \to L_{p_1,q_1} \to T_{p_1,q_1} \to L_{p_2,q_2} \to T_{p_2,q_2} \to \cdots \to L_{p_k,q_k} \to T_{p_k,q_k} \to D$. Since such a path from S to D can be constructed for every tracker $U_i \in \mathcal{B}$, exactly m unit flows can be constructed from S to D. As the m trackers are sufficient to track all targets at least once, at least one edge in every targets edge-group $\mathcal{E}_k, k = 0, \ldots, n-1$ would have a flow of one unit thus satisfying the requirements of the GMFP.

Conversely, if the solution to the GMFP of (8) has m unit flows from S to D, then exactly m trackers are sufficient to track all targets at least once and their trajectories are deducible from the m unit flows. Due to the construction rules of G_2, each unit flow in G_2 corresponds to a path from S to D where the intermediate nodes are of the form $L_{p,q}$ and $T_{p,q}$, and intermediate edges are either of the form $L_{p,q} \to T_{p,q}$ or $T_{p,q} \to L_{p',q'}$. From each such flow, a tracker trajectory can be constructed as follows. Suppose that there exists a flow in G_2 from S to D of the form $S \to L_{a,b} \to T_{a,b} \to L_{c,d} \to T_{c,d} \to L_{e,f} \to T_{e,f} \to D$, a trajectory for tracker U_i can be constructed from this flow where the tracker begins its traversal from location a at time step b, then moves to location c at time step d, then finally to location e at time step f, where it ends its traversal. Since m flows are sufficient to satisfy the lower bound constraints imposed on graph G_2 by (8), it can be concluded that m trackers are sufficient to track all targets at least once in the given observation period. □

In the following, we show how the GMFP of (8) on the graph $G_2 = \langle V_2, E_2 \rangle$ can be computed using integer linear programming (ILP). The objective of the ILP is as follows:

$$\text{minimize } F$$

subject to

$$\sum_{j \in V_2} f(i, j) - \sum_{j \in V_2} f(j, i) = \begin{cases} F, & i = S \\ 0, & i \neq S, D \\ -F, & i = D \end{cases} \tag{10}$$
$$\forall \mathcal{E}_k, 0 \leq k \leq n-1, \text{ if the edge } (i, j) \in \mathcal{E}_k$$
$$\sum f(i, j) \geq 1 \tag{11}$$
$$\forall (i, j) \in E_2, f(i, j) \leq c(i, j) \tag{12}$$
$$\forall f(i, j) = 0/1 \tag{13}$$

H. Extension to Other Classes of Targets

Beyond the two classes of targets discussed in the preceding sections that required targets to be tracked either continuously (TTP_1), or for a single unit of time during the period of observation (TTP_2), other classes of targets exist that have different tracking requirements. For instance, a target may require *variable time* tracking, i.e., during the observation period $[0\text{–}\mathcal{T}]$ a target may require to be tracked for t units of time, where $0 \leq t \leq \mathcal{T}$. If this class of targets is denoted by TTP_3, several subclasses of this target class may also exist. For instance, a target subclass may require that the t units of tracking time must be *continuous* (TTP_{3A}), or that the t units of tracking can be *fragmented* over the period of observation (TTP_{3B}). Several other variations of target classes may also exist beyond these classes, such as targets requiring periodic tracking, or targets that need to be tracked for t units of time every t' time units.

Although several target classes exist that are beyond the scope of this paper, some of the proposed solution techniques in this paper can be modified to solve for problem instances for other target classes. For instance, the solution for the TTP_2 problem proposed in this paper can be extended to solve the TTP_{3B} type of target subclass as follows. The same graph construction steps for the solution of TTP_2 (Section IV-G) can be used for TTP_{3B}. Also, for the TTP_{3B} target class for each edge set $\mathcal{E}_k \in \mathcal{E}'$, a lower bound l_k exists where $0 \leq l_k \leq \mathcal{T}$ that denotes the tracking time requirement for each target of the TTP_{3B} problem instance.

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

Correspondingly, the constraint outlined in (11) for TTP_2 can be modified to $\sum f(i, j) \geq l_k$ to reflect the tracking requirement for each target of the TTP_{3B} problem instance. A preliminary study of target tracking for TTP_{3A} and TTP_{3B} types of target classes has been presented by Das *et al.* [21] in the context of mobile data collection using data mules.

I. Impact of Discretization

It may be noted that although the TTP is a *continuous time domain* problem (as the mobile targets and trackers can be anywhere in the deployment area at a given time), the proposed solution techniques discretize both time and space. Time is discretized into equal intervals of length δ and space into equal intervals of length ε. This discretization allows the possibility of degradation of the quality of the solution, in the sense that the absolute minimum number of trackers needed to track all targets may not be found. Accuracy of the solution can be improved if smaller values of the discretization parameters δ and ε are chosen. However, it may also be noted that smaller values of δ and ε will also increase the number of nodes and edges of the constructed graphs, thereby increasing the complexity of finding the solution to the generalizations of the minimum flow problem in the constructed graphs. This is particularly true if the solutions are computed using an ILP. Thus, the proposed techniques in the preceding sections offer a direct tradeoff between the *quality of the solution* (accuracy) and the *cost of the solution* (computation time) by allowing separate inputs for the discretization parameters δ and ε.

V. EXPERIMENTAL RESULTS

In this section, we present our experimental results for the techniques proposed in the preceding sections. This section is intended to give an insight on two primary aspects of the proposed techniques. Firstly, as the proposed techniques are primarily integer linear programs that compute the optimal solution for computationally hard problems, this section first provides an insight on the average execution times required for computing the optimal solution for different problem instances. Second, as the proposed techniques factor in several variables (time/space discretization parameters, speed of trackers, tracker sensing range, etc.), the choice of variables not only impact the computation time, but also impact the solution itself, i.e., the minimum number of required trackers. To this effect, the second half of this section deals with the impact of such variables on the computed solution.

A. Impact of Problem Instance Size on Computation Time

To provide insights on the impact of the size of the problem instance to the computation time required, we constructed two problem instance environments and created individual problem instances by varying the number of targets in that environment. For these set of experiments, we set $\varepsilon = 1$, $\delta = 1$, and considered two 2-D spatial grid environments of 10×10 and 25×25 units, where targets (trackers) can be located at any of the intersection points of the grid at a given time. For each of these environments, we varied the number of targets from 5 to 20 and generated random target trajectories within the bounds of the environment over the observation period [0–10], ($\mathcal{T} = 10$), to create individual problem instances. For each number of targets considered, we created five problem instances for that target count, recorded the execution time for each one of them, and averaged the computation time. For each instance, we set the sensing range of the tracker $r = 1$, and maximum available speed of the tracker $d = 2$. For the purpose of providing a broader insight into the execution times, for each generated problem instance we ran our solution techniques for TTP_1 (continuous tracking), TTP_2 (one-time tracking), and also for TTP_{3B} (variable-time tracking, with allowed fragmented tracking) after creating random variable tracking time requirements for each target of the problem instance. As outlined in Section IV-H, the techniques to compute TTP_2 were modified for computing the solution for TTP_{3B}. In the constructed network flow graphs for the 10×10 environment, the number of nodes ranged from 394 to 1676 for TTP_1, and 786 to 3350 for TTP_2 (TTP_{3B}) instances, while the number of edges ranged from 44 828 to 834 106 for TTP_1 and 13 512 to 212 741 for TTP_2 (TTP_{3B}) instances. Similarly, for the 25×25 environment, the number of nodes in the network flow graph ranged from 429 to 1760 for TTP_1, and 854 to 3518 for TTP_2 (TTP_{3B}) instances, while the number of edges ranged from 24 586 to 348 242 for TTP_1, and 8688 to 62 580 for TTP_2 (TTP_{3B}) instances. A total of approximately 500 experiments were carried out by solving the problem instances using the IBM CPLEX Optimizer 12.5 on an Intel i5 2.8 Ghz Quad-core processor with 8GB memory running a Windows 8 environment. Fig. 8(a) and (b) shows the average execution times of the constructed instances.

B. Impact of Instance Variables on the Computed Solution

In this section, we investigate the effect of different parameters such as the tracker sensing radius (r) and the speed of trackers (d), on the number of trackers needed to track all targets for both TTP_1 and TTP_2. We also conduct experiments to examine the effects of the granularity of the discretization parameters ε and δ on the total number of trackers required to track all the targets.

For this set of experiments, a problem instance with five mobile targets in a 2-D deployment area over the time interval [0–9] are individually considered for TTP_1 and TTP_2. Fig. 9(a) and (b) shows the target trajectories considered for TTP_1 and TTP_2, respectively. The trajectories considered for each of the five targets in the two problem instances were given by different parametric equations. Thus, the speed of the targets were not constant or uniform. The same hardware and software environment used in the experiments of the previous section were also used to solve these two instances. We present the results of these experiments next.

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

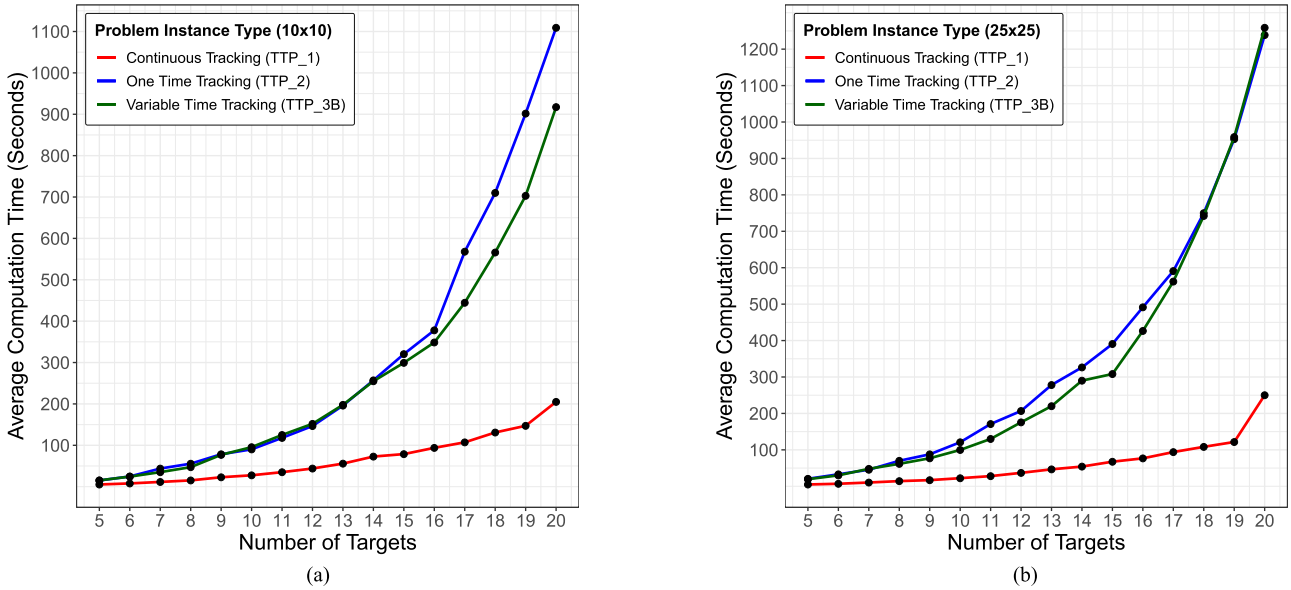


Fig. 8. Average execution times (in seconds) for problem instances created in (a) 10×10 and (b) 25×25 2-D spatial grids. Computation times averaged over five instances for each number of targets.

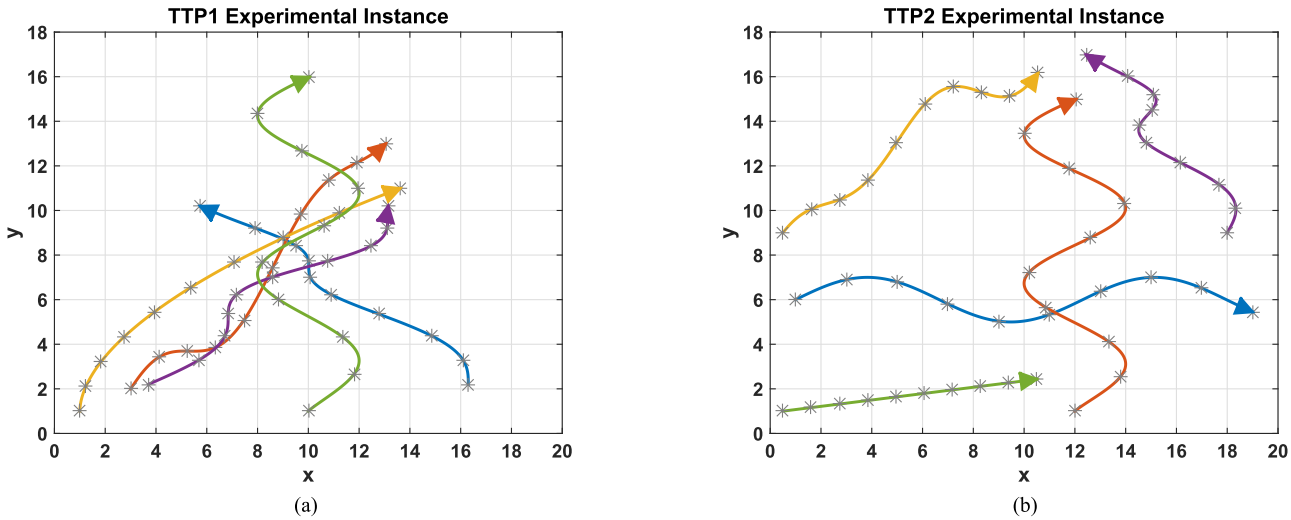
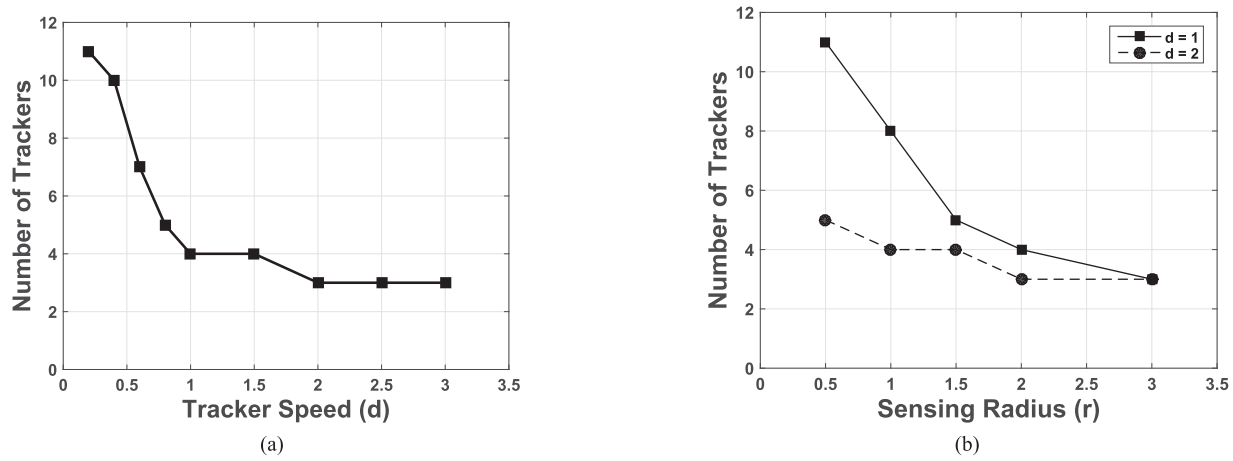


Fig. 9. (a) Trajectories of a TTP_1 problem instance with five targets moving on a 2-D plane over the time interval [0–9]. (b) Trajectories of a TTP_2 problem instance with five targets moving on a 2-D plane over the time interval [0–9]. The gray asterisks on the trajectories denote the locations of the target for each time step from 0 to 9, where the asterisk at the tail of the arrow denotes the target location at time step 0, while the asterisk at the head of the arrow denotes the location of the target at time step 9.

1) *Experimental Results for TTP_1:* In the first set of the TTP_1 experiments, we compute the minimum number of trackers required to track all targets at every (discrete) time step for different values of tracker speed d. In these experiments, we set $r = 2$, $\varepsilon = 1$, $\delta = 2$, and the observation period considered was from [0–9]. Fig. 10(a) illustrates the results of these experiments and the following observations can be made from these results. 1) When the tracker speed d is less than the average speed of the targets, the number of trackers needed to cover all targets in every discrete time step is more than the total number of targets. 2) By increasing the tracker speed d, the number of trackers decreases initially, but when $d \geq 2$ which is close to or greater than the average speed of the targets per δ unit of time, three trackers are sufficient to track all five targets at every discrete time step with an interval $\delta = 2$ between [0–9]. We note that the speed of targets in these experiments is not constant or uniform since the parametric equations for their trajectories are different.

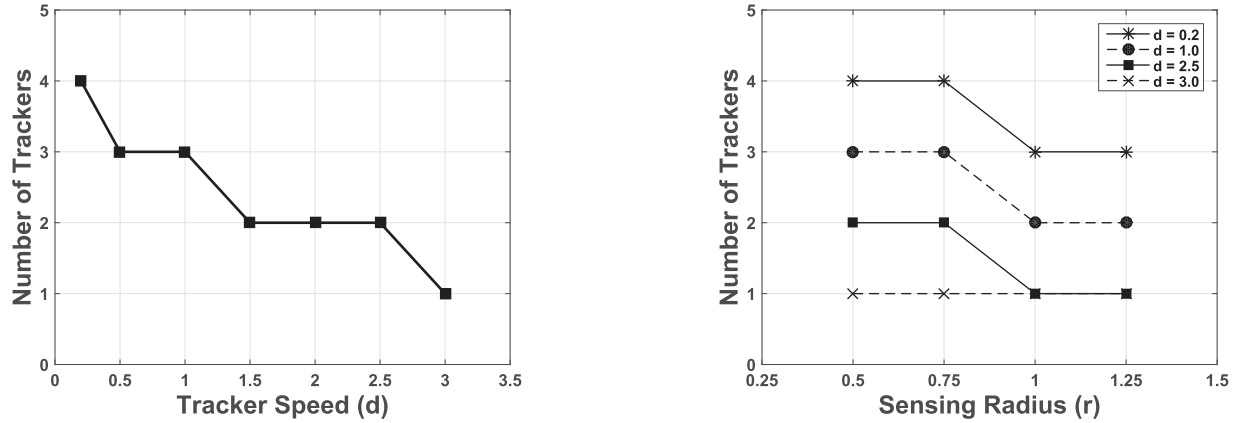
In the second set of experiments, we examine the effect of the sensing radius of trackers r on the solution of TTP_1. In these experiments, we set $\varepsilon = 0.5$, $\delta = 2$, and the observation period considered was from [0–9]. Fig. 10(b) shows the results of these experiments for two different values of tracker speed d. It can be observed from the experiments that 1) when the speed of trackers is large enough (in this example we set $d = 2$), the number of trackers needed to cover the targets does not exceed the number of targets,

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.



(a) (b)

Fig. 10. Experimental results for a TTP_1 problem instance of Fig. 9(a). (a) Number of trackers versus tracker speed (d) in time interval [0–9], where tracker sensing radius $r = 2$, and discretization parameters $\varepsilon = 1$ and $\delta = 2$. (b) Number of trackers versus tracker sensing radius (r) in time interval [0–9], where tracker speed $d = 1$, 2, and discretization parameters $\varepsilon = 0.5$, and $\delta = 2$.



(a) (b)

Fig. 11. Experimental results for a TTP_2 problem instance of Fig. 9(b). (a) Number of trackers versus tracker speed (d) in time interval [0–9], where tracker sensing radius $r = 2$, and discretization parameters $\varepsilon = 1$ and $\delta = 2$. (b) Number of trackers versus sensing radius (r) with varying tracker speed d in the time interval [0–9], $\varepsilon = 0.5$, $\delta = 1.0$.

even when the sensing radius r is very small, and 2) when the sensing radius increases, the number of trackers for both values of d starts to get closer. It may be noted that when the value of r is large enough, only one tracker will be sufficient to track all targets of the problem instance of Fig. 9(a).

2) *Experimental Results for TTP_2:* For the TTP_2 problem instance of Fig. 9(b), we varied the tracker speed d and computed the minimum number of trackers required to track all targets at least once during the observation period [0–9]. In these experiments, we set $r = 0.75$, $\varepsilon = 0.5$, and $\delta = 1$. Fig. 11(a) shows the results of these experiments. It can be observed from these experiments that when the sensing radius $r = 0.75$, with the increase of tracker speed d, the number of trackers required keeps decreasing and eventually becomes 1 when $d = 3$, which is more than twice the average speed of all targets per δ unit of time.

In the second set of experiments for TTP_2, we investigated the impact of sensing radius r on the total number of trackers for different speed of targets d. The discretization parameters were set to $\varepsilon = 0.5$, $\delta = 1.0$, and the observation period was [0–9]. The tracker speed d was varied from 0.2 to 3.0, and the sensing range r from 0.5 to 1.25. The results of these experiments are shown in Fig. 11(b). It can be observed from these results that 1) when the speed of trackers is sufficiently large (in this case when $d = 3$ which is more than twice the average speed of the targets), even when the sensing radius r is very small ($r = 0.5$), just one tracker is sufficient to track all targets at least once during the period [0–9], and 2) when the sensing radius increases, the number of trackers for the different values of d starts to get closer. It may be noted that for sufficiently large values of r, only one tracker will be sufficient to track all targets of the problem instance of Fig. 9(b) even when tracker speed varies from 0.2 to 3.0.

3) *Observed Impact of Discretization Parameters δ and ε:* In both the TTP_1 and TTP_2 experiments, we observed the impact of different values for δ and ε on the number of trackers required to track all the targets. Observations

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

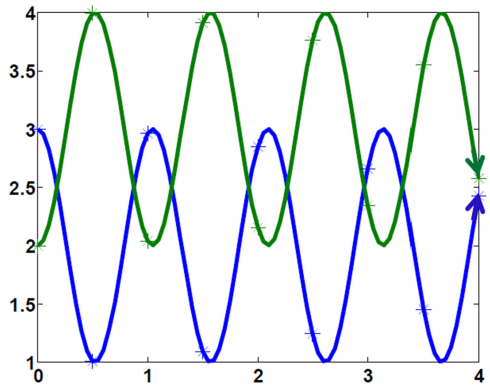


Fig. 12. Trajectories of two targets in time interval [0–4]. Reprinted by permission from Springer Nature: Springer International Publishing, in *Proc. Int. Conf. Combinatorial Optimization and Applications* [18] © 2014.

from these experiments showed that smaller values of δ and ε allowed the proposed solution techniques for TTP_1 and TTP_2 to yield better results, i.e., a lesser number of trackers may be required to track all targets depending on the values of δ and ε. A smaller value of ε increases the granularity of the deployment space, i.e., for every target a_i at time step t, more potential points per unit of time t will be considered by the solution technique when ε is small. This increase in the number of potential points can result in a fewer number of required trackers as the solution technique will now consider more candidate locations in the deployment space where a tracker can be present. A similar argument can be made for smaller values of the time discretization parameter δ as well. A smaller value of δ enables the proposed techniques to consider more potential points within the observation period [0–$\mathcal{T}$]. This may also lead to a fewer number of required trackers to track all targets. Although smaller values of δ and ε have the potential to increase the accuracy of the solution (by decreasing the number of required trackers), the cost of computation also increases considerably.

For instance, for the TTP_1 problem instance depicted in Fig. 9(a), we examined different values of $\delta = 1, 2, 4, 8$ and $\varepsilon = 0.5, 1$, where $r = 2, d = 2$ in the time interval [0–8]. For this set of target trajectories, the change of δ and ε did not effect the solution of TTP_1 and three trackers were sufficient to track all the targets for each (discrete) time step in [0–8]. However, it may be noted that when target trajectories are different than the ones considered in Fig. 9(a), the solution of TTP_1 may change considerably depending on the values of δ and ε. For instance, when the target trajectories come very close to each other and then diverge in short time intervals. Fig. 12 depicts two such target trajectories in the time interval [0–4]. Let $\varepsilon = 0.5$, $r = 1$, and $d = 2$. For this example, we computed the solution of TTP_1 for different values of $\delta = 0.5, 1, 2, 4$. When $\delta = 0.5$, the solution of TTP_1 was 2 while for the rest of values the solution of TTP_1 was 1.

Similar experiments were also conducted for the TTP_2 problem instance. For the example of Fig. 9(b), different values of δ and ε were examined, specifically $\delta = 0.5, 1, 2, 4, 8$ and $\varepsilon = 0.5, 1$ with $r = 1, d = 1$ in the time interval [0–8] were observed. For the set of target trajectories of Fig. 9(b), the change of values for δ and ε had no impact on the minimum number of trackers required to track all targets at least once in the period of observation [0–8]. However, the computation time significantly increased when smaller values of δ and ε were used for the experiments.

VI. CONCLUSION

In this paper, we studied the mobile TTP using the fewest number of mobile trackers for two different classes of targets. Based on practical applications of TTPs, such as in convoy protection of military vehicles and mobile RFID tag tracking in warehouses, we envisaged two different classes of targets: 1) targets that need tracking for the *entire duration of observation* (Class I) and 2) targets that need tracking *at least once* during the period of observation (Class II). We formulated the TTP where the objective is to find the fewest number of mobile trackers and their trajectories required to track all mobile targets during the period of observation. Depending on the target class, we studied two different versions of the TTP, namely: 1) TTP_1—where all targets belong to Class I and 2) TTP_2—where all targets belong to Class II. We proved that both versions of the TTP problem are computationally hard. We then proceeded to formulate the TTP_1 and TTP_2 problems as network flow problems and proposed algorithms for their solution. We evaluated the performance of our algorithms through simulation and studied the impact of the problem size to the execution time, and the impact of parameters such as speed and sensing range of the trackers on the minimum number of trackers needed to track all targets.

REFERENCES

[1] X. C. Ding, A. R. Rahmani, and M. Egerstedt
Multi-uav convoy protection: An optimal approach to path planning and coordination
IEEE Trans. Robot., vol. 26, no. 2, pp. 256–268, Apr. 2010.

[2] N. Sherman
Amazon showcases robots at massive new warehouse
2015. [Online]. Available: http://www.baltimoresun.com/business/bs-bz-amazon-grand-opening-2015092 8-story.html

[3] E. Ackerman
Flying inventory assistants are a good use for drones
2014. [Online]. Available: http://spectrum.ieee.org/automaton/robotics/aerial-robots/flying-invent ory-assistants-are-a-good-use-for-drones

[4] J. Cleveland *et al.*
An automated system for semantic object labeling with soft object recognition and dynamic programming segmentation
in *Proc. IEEE Int. Conf. Automat. Sci. Eng.*, 2015, pp. 683–690.

[5] D. Zorbas, T. Razafindralambo, D. P. P. Luigi, and F. Guerriero
Energy efficient mobile target tracking using flying drones
Procedia Comput. Sci., vol. 19, pp. 80–87, 2013.

[6] T. Oliveira, A. P. Aguiar, and P. Encarnação
Moving path following for unmanned aerial vehicles with applications to single and multiple target tracking problems
IEEE Trans. Robot., vol. 32, no. 5, pp. 1062–1078, Oct. 2016.

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

[7] N. Farmani, L. Sun, and D. Pack
A scalable multi-target tracking system for cooperative unmanned aerial vehicles
IEEE Trans. Aerosp. Electron. Syst., vol. 53, no. 4, pp. 1947–1961, Aug. 2017.

[8] E. Xu, Z. Ding, and S. Dasgupta
Target tracking and mobile sensor navigation in wireless sensor networks
IEEE Trans. Mobile Comput., vol. 12, no. 1, pp. 177–186, Jan. 2013.

[9] Y. Zou and K. Chakrabarty
Distributed mobility management for target tracking in mobile sensor networks
IEEE Trans. Mobile Comput., vol. 6, no. 8, pp. 872–887, Aug. 2007.

[10] H. Chen, K. Chang, and C. S. Agate
UAV path planning with tangent-plus-Lyapunov vector field guidance and obstacle avoidance
IEEE Trans. Aerosp. Electron. Syst., vol. 49, no. 2, pp. 840–856, Apr. 2013.

[11] H. Oh, S. Kim, H.-s. Shin, and A. Tsourdos
Coordinated standoff tracking of moving target groups using multiple UAVs
IEEE Trans. Aerosp. Electron. Syst., vol. 51, no. 2, pp. 1501–1514, Apr. 2015.

[12] Y. Fu, Q. Ling, and Z. Tian
Distributed sensor allocation for multi-target tracking in wireless sensor networks
IEEE Trans. Aerosp. Electron. Syst., vol. 48, no. 4, pp. 3538–3553, Oct. 2012.

[13] M. Naderan, M. Dehghan, and H. Pedram
Mobile object tracking techniques in wireless sensor networks
In *Proc. Int. Conf. Ultra Modern Telecommun. Workshops*, Oct. 2009, pp. 1–8.

[14] E. Adamey and U. Ozguner
A decentralized approach for multi-UAV multitarget tracking and surveillance
In *Proc. SPIE*, vol. 8389, May 2010, Art. no. 838915.

[15] A. Srinivas, G. Zussman, and E. Modiano
Construction and maintenance of wireless mobile backbone networks
IEEE/ACM Trans. Netw., vol. 17, no. 1, pp. 239–252, Feb. 2009.

[16] G. Radhakrishnan and S. Saripalli
Target tracking with communication constraints: An aerial perspective
In *Proc. IEEE Int. Workshop Robot. Sensors Environ.*, Oct. 2010, pp. 1–6.

[17] L. E. Dubins
On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents
Amer. J. Math., vol. 79, no. 3, pp. 497–516, 1957.

[18] D. Hay, S. Shirazipourazad, and A. Sen
Optimal tracking of multiple targets using UAVs
In *Proc. Int. Conf. Combinatorial Optim. Appl.*, 2014, pp. 750–763.

[19] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto
Optimal packing and covering in the plane are NP-complete
Inf. Process. Lett., vol. 12, no. 3, pp. 133–137, 1981.

[20] S. Even
Graph Algorithms. Cambrdige, U.K.: Cambridge Univ. Press, 2011.

[21] A. Das, A. Mazumder, A. Sen, and N. Mitton
On mobile sensor data collection using data mules
In *Proc. Int. Conf. Comput., Netw. Commun.*, 2016, pp. 1–7.

Arun Das received the Ph.D. in computer science from Arizona State University, Tempe, AZ, USA, in 2016. As a graduate student, his research focused on combinatorial optimization techniques for efficient resource allocation in cyber physical systems and critical infrastructure networks.

Since 2017, he has been a Staff Software Engineer with Visa Inc., Foster City, CA, USA, where his work involves applying network theory, graph theory, and optimization techniques to develop efficient fraud detection, fraud resolution, and risk mitigation techniques for Visa's Card-Linked Offers platform. His research interests include the design and analysis of algorithms, combinatorial optimization, computer networks, complex networks, and network tools.

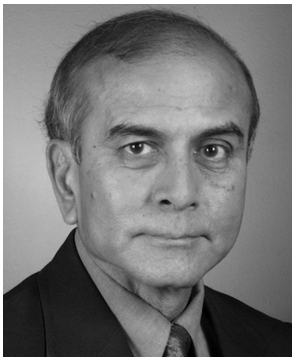
Shahrzad Shirazipourazad received the B.Sc. degree in software engineering and the M.Sc. degree in computer networks from the Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, in 2007 and 2009, respectively, and the Ph.D. degree in computer science from Arizona State University, Tempe, AZ, USA, in 2014.

In 2014, she continued her research as a Postdoctoral Researcher with Arizona State University. In October 2014, she joined PayPal as a Senior Software Engineer. Her research interests include resource optimization in communication and social networks, network robustness analysis, and design and analysis of algorithms.

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.

David Hay received the B.A. (summa cum laude) and Ph.D. degrees in computer science from the Technion—Israel Institute of Technology, Haifa, Israel, in 2001 and 2007, respectively.

He was with IBM Haifa Research Labs, Cisco Systems, San Jose, CA, USA, with the Electronic Department, Politecnico di Torino, Turin, Italy, and with the Electrical Engineering Department, Columbia University, New York, NY, USA. He is currently an Associate Professor with the Rachel and Selim Benin School of Computer Science and Engineering, Hebrew University of Jerusalem, Jerusalem, Israel. In 2010, he co-founded (with Prof. Brembler-Barr) the DEEPNESS lab, focusing on deep packet inspection in next generation network devices. His research interests include computer networks—in particular, network algorithmics, packet classification, deep packet inspection, network survivability, and software-defined networking.



Arunabha (Arun) Sen received the bachelor's degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, and the Ph.D. degree in computer science from the University of South Carolina, Columbia, SC, USA.

He was a Fulbright Fellow with INRIA, Lille, France, and a visiting scientist with the Wroclaw University of Technology, Wroclaw, Poland and with the University of Passau, Passau, Germany. He is currently a Professor of computer science and engineering with Arizona State University, Tempe, AZ, USA. He spent his sabbaticals at the University of California, San Diego, CA, USA and the University of California, Berkeley, CA, USA. His research interests include the areas of resource optimization problems in networks, including airborne, wireless, optical, power grid, transportation, social networks, and the newly emerging discipline of network science. He primarily studies the algorithmic issues related to the problems in these domains utilizing graph theoretic, game theoretic and combinatorial optimization techniques.

Authorized licensed use limited to: Hebrew University. Downloaded on June 16,2023 at 11:20:35 UTC from IEEE Xplore. Restrictions apply.