A Novel Framework for Tracking Groups of Objects

Ron Zohar and Dan Geiger
Department of Computer Science, Technion, Israel
ronron@tx.technion.ac.il, dang@cs.technion.ac.il

May 30, 2002

Abstract

We describe a novel framework for group tracking and identification termed flow conservation group tracking. Our framework integrates local kinematic measurements with flow conservation constraints so as to yield a robust estimation of the positions and types of targets across an arena. The framework is based on three components each of which is novel. The measurement-to-track component uses a reduction to minimum cost flow problems. The estimation of the number of objects on each track is based on estimation using flow networks, and the assessment of the types of vehicles is reduced to hierarchy matching algorithms.

Keywords: Flow networks, Gaussian networks, Group tracking, Hierarchies, Target Identification.

1 Introduction

Group tracking is an approach to tracking where groups of objects are being monitored as a single entity rather than monitoring each object individually. Group tracking is applied to tracking closely spaced objects with similar state vectors [4]. For such closely spaced objects, it is often impossible to track each object separately due to sensors’ limitations. Furthermore, in some applications, such as tracking a convoy of ground vehicles, maintaining individual tracks of each vehicle often presents an insurmountable computational burden.

On a battlefield, groups of vehicles may split or merge, and it is the responsibility of the group tracking system to continuously maintain the amount and types of targets within each group. An up-to-date text book of this field [4] comments: "Despite the intuitive appeal of group tracking, little development has been reported in the tracking literature since the classic work of Taezer [15] and the overview discussion given in Chapter 11 of [3]. It appears that the implementation problems resulting from splitting and merging of groups have discouraged the further development of group tracking methods. However, it seems clear that some form of group tracking is the best approach for the tracking environment of many closely spaced targets."

Standard tracking algorithms for single objects [2, 4] associate measurements at sample time $t_i$ with tracks estimated at time $t_{i-1}$ (measurement-to-track algorithms). This association is mostly based on kinematic models, consisting of location, velocity, and sometimes acceleration. A well known approach for extending this methodology to tracking groups is centroid group tracking [3, 4]. Using this approach, only a centroid generated for the group is tracked where a centroid is a virtual object which summarizes the kinematics of the group. This virtual centroid object is treated in the tracking system as if it were a single real object.

The principle observation motivating this work is that centroid group tracking is a mis-generalization of the tracking framework from single objects to tracking groups of objects because it places the main emphasize on kinematics rather than on the principle of matter conservation, namely, that the number of objects before and after a split is not changed. In single object tracking, matter is implicitly conserved by the mere existence or inexistence of a track; The association of a
single observation to each existing track guarantees that objects disappear from the tracking system only when an explicit decision is made to drop a track. On the other hand, in centroid group tracking, each observation and each track may represent numerous objects, and the actual number of objects within each group may only be estimated. Consequently, a group tracking framework which does not insure that matter is conserved through time may yield erroneous decisions. This problem worsens when the frequency of sensor readings is low, yielding kinematics information that is not sufficiently accurate.

When groups of objects sampled at time $t_i$ are associated with established tracks, one should use an estimation of the number of objects within the group; There is no point associating a large group of objects with a previously established track that contains a few objects just because the kinematic information matches. Similarly, when a group splits into several subgroups it is clear that the number of objects in the subgroups equals the number of objects in the original group. However, this important structural information is being neglected in current group tracking schemes, where the number of vehicles in each group and their type are ignored or estimated in a local fashion.

In this paper we describe a novel framework for group tracking and identification termed *flow conservation group tracking*, which we believe to be a preferred extension of the methodology from tracking single objects to tracking groups of objects. Our framework integrates local kinematic measurements with flow conservation constraints so as to yield a more robust estimation of the position and types of objects across an arena. We base our framework to a large extent on the ability to estimate and match the number of targets in each group, in addition to kinematics.

An important component of our framework is the estimation of the types of objects in the arena given reports received at a group level. We present the notion of *hierarchical identification* where each report may be a list of labels of various specificity. For example (T55, T88, T88, Tank) is a report about four objects. We develop a framework and polynomial algorithms for the fusion of consistent and inconsistent such reports into a single summary report and for assessing the quality of the given reports.

The rest of this paper proceeds as follows: In Section 2 we describe the components of the flow conservation group tracking framework. In Sections 3 and 4 we describe in some detail two of the three components of our framework and conclude with a discussion of ongoing research (Section 5).

2 The Framework

As in classic frameworks for multi target tracking systems [e.g. [4]] we assume iterative processing is being used so that tracks that have been formed on a previous scan are being updated at the next scan. Incoming observations are being associated with existing tracks using an association process which integrates kinematic estimation with flow conservation. Associated observations are used to update track parameters and may cause the splitting or the merging of tracks. Similarly, observations not assigned to existing tracks may initiate new tentative tracks. The tracks are updated and stored. Then, a flow estimation component uses the updated tracks information to estimate the updated number of objects on each track. Finally, an identification component uses the tracked information and the estimated number and types of objects on each track for the fusion of tracking reports into a single consistent summary report. This framework is depicted in Figure 1. In this paper we describe the guidelines of the first component and focus on the second and third component.

The first component in our system performs *Flow Conserving Measurements-to-Track Association*. This component receives clusters from an external information source which samples groups in the arena at time $t_i$. The information provided for each reported cluster includes its approximate location and velocity, an estimated number of vehicles with error characterization and (possibly) aggregated information of the types of objects in the cluster. The measurements-to-track association procedure associates the clusters received at time $t_i$ with tracks existing at time $t_{i-1}$ where the information stored for each track includes the above mentioned tracking information (location,
The flow-conserving measurements-to-track association component has been formalized and efficiently solved as a minimum cost flow problem, where the association process is modeled as the flow of vehicles from tracks existing at time $t_{i-1}$ to observations at time $t_i$. In particular, a bi-partite graph is generated with track-nodes for each track that existed at time $t_{i-1}$ and with cluster-nodes for each cluster received at time $t_i$. Directed edges connect every track-node with every cluster-node to which it may be associated. Each edge $(T_j, C_k)$ connecting track $T_j$ with cluster $C_k$ is assigned a value which represents the cost of flow through the edge. We select cost values that are proportional to the log probability for an object belonging to track $T_j$ at time $t_{i-1}$ to be detected at cluster $C_k$ at time $t_i$. This probability may be determined using prediction of the track’s location at time $t_i$, which can be done using existing approaches (e.g. KF, EKF, IMM) and as proposed for centroid group tracking. Finally, a minimum cost flow algorithm is applied to this bi-partite graph yielding for each edge $(T_j, C_k)$ the amount of vehicles from track $T_j$ that are accounted for by cluster $C_k$. It should be noted that the suggested approach is capable of addressing the issues of low probability of detection and false detections, as well as arrival or departure of objects to and from the arena by a slightly more sophisticated modeling of the flow graph.

This approach for measurement-to-track association, which is currently being developed with Mark Matusevich, differs significantly from the component with a similar name used in classic frameworks for multi target tracking systems. First, this component generates many-to-many association relations, where previous components typically generate one-to-one relations. Second, our component uses the estimation of number of objects to assure matter conservation.

The second component in our framework estimates the number of vehicles on each track. The accuracy of the estimation directly affects the performance of the measurement-to-track association component. In [3] it is suggested to estimate the number of targets in a group by averaging the number of observations contained in several group detections associated with the track. Indeed, when a track is continuously maintained for a long period of time, this approach is appropriate. However, in general, the track splits and merges repeatedly such that the number of observations associated with the latest track does not provide sufficient information for number-of-objects estimation. It should be stressed that in reality, the merging and splitting of tracks may occur very frequently mainly due to clustering artifacts rather than actual operational maneuvers. Consequently, the samples received on the latest track alone may not suffice for reliable estimation of the number of objects on that track. This motivates our Flow Conserving Flow Estimation component.
When a group in the arena splits into several subgroups it is clear that the number of objects in the subgroups equals the number of objects in the original group. Our flow conserving measurement-to-track association component ensures that this is also the case for the generated tracks. Consequently, it is possible to create a graph from the tracking information where edges represent tracks, internal nodes correspond to the merger or separation of groups of objects, such that the number of objects on each track is constant and the number of objects that enter an internal node equals the number of objects that leave it. Given a set of measurements of the number of objects on each edge the problem we address, which we call the Most Probable Flow Estimation problem (MPFE), is to estimate the most probable assignment of flow for every edge such that the conservation constraint is maintained. This problem is further described in Section 3.

Finally, the third component of our system performs Flow Conserving Type Identification. This component is applicable to systems which receive aggregated type information about groups in the arena. Once again, the frequent splitting and merging of groups requires incorporation of global information via flow conservation constraints on each type in order to yield a more accurate identification. In Section 4 we develop a framework and polynomial algorithms for the fusion of consistent and inconsistent reports on types of objects into a single summary report for each track separately. We present the notion of hierarchical identification where each report may be a list of labels of various specificity.

3 Flow Estimation

We now focus on the problem of accurately estimating the number of targets in an arena within our flow conservation group tracking framework. The estimated number of vehicles is an integral component in the measurement to track association procedure and therefore directly affects the quality of the group tracking system. When a group in the arena splits into several subgroups, it is clear that the number of targets in the subgroups equals the number of targets in the original group. Consequently, it is possible, given a set of measurements of the number of targets on each track, to globally estimate the most probable number of vehicles on each track by utilizing these flow conservation constraints. We model this problem using the notion of flow networks.

A flow network is a weighted directed graph $G = (V, E, w)$ where $V$ is a set of nodes, $E$ is a set of $m$ directed edges and $w$ is a function $w : E \rightarrow \mathbb{R}$ which assigns a flow $w(l)$ for every edge $l$ such that the following flow conservation constraint is maintained: the sum of flows into an internal node equals the sum of flows going out of an internal node. A Hidden Flow Model (HFM) is a family of conditional distributions of the measured flow for the edges in a flow network given the true flow in the network.

Given a set of measurements of the flow on each edge and assuming an independent Gaussian measurement error for each edge, the problem we address, which we call the Most Probable Flow Estimation problem (MPFE), is to estimate the most probable assignment of flow for every edge such that the conservation constraint is maintained. In group tracking, edges represent tracks, and internal nodes correspond to the merger or separation of groups of objects. The number of objects on each track is constant and the number of objects that enter an internal node equals the number of objects that leave it. Various sensors are used to track the scene. Each sensor provides an estimate for the number of vehicles on each track. The goal is to improve the estimation of the number of vehicles on each track by utilizing the flow constraints on internal nodes.

Similar problems arise whenever an estimation is required in problems that can be modeled as a Hidden Flow Model. For example, estimation of currents in electrical networks, traffic estimation in roads, and throughput estimation in computer networks. Most probable flow estimation provides a mechanism for improving the estimation obtained by individual independent sensors by utilizing flow conservation constraints.

In this section we present polynomial algorithms for the assignment of flow to each edge, such
that the likelihood of this assignment is maximized given individual measurements with normally distributed measurement errors. An $O(m)$ algorithm is provided when the underlying undirected graph of $G$ is a tree where $m$ is the number of edges. Notably, a solution of this problem using Linear Minimum Variance Unbiased estimation (LMVU) [9] or Conditional Gaussian (CG) networks [10, 5] yields an $O(m^3)$ algorithm.

The rest of this section is organized as follows. We first provide a formal description of the flow estimation problem (Section 3.1) and discuss known solutions (Section 3.2). Then, we describe an $O(m)$ algorithm for the special case of binary polytrees, and analyze the algorithm (Section 3.3).

In a companion technical report [16] we extend the algorithm to deal with general polytrees in time complexity of $O(m)$ and with unknown precision of the measuring devices and we describe several experiments that demonstrate the estimation improvement, and a sensitivity analysis of the assumption to normally distributed measurement errors.

### 3.1 Problem Formulation

A flow network is a weighted directed graph $G = (V, E, w)$ where $V$ is a set of nodes, $E \subseteq V \times V$ is a set of $m$ directed edges, and $w : E \rightarrow \mathbb{R}$ is a flow function that assigns a flow $w(l)$ to edge $l$. An internal node is a node that has some edges coming into it and some that are leaving it. A source is a node with a single outgoing edge and no incoming edges and a sink is a node with a single incoming edge and no outgoing edges. Let $A(v)$ and $B(v)$ denote the sets of edges into node $v$ and out of node $v$, respectively.

For every internal node $v \in V$, the flow conservation constraint is given by

$$0 = \sum_{l \in A(v)} w(l) - \sum_{l \in B(v)} w(l)$$  \hspace{1cm} (3.1)

These constraints are linear and can be represented using an $n \times m$ constraint matrix $R$ via the equation $RX = 0$, where $n$ is the number of rows (one per internal node), $m = |E|$ is the number of columns (one per edge), $X_j = w(l_j)$ is the flow on edge $l_j$, and $X = (X_1, ..., X_m)$. In particular, for every node $v_i$ and edge $l_j$, we set $R_{ij} = -1$ if edge $l_j$ points to $v_i$, $R_{ij} = 1$ if edge $l_j$ points away from $v_i$, and $R_{ij} = 0$ otherwise.

We define the Most Probable Flow Estimation (MPFE) problem as follows:

**Instance:** Let $G = (V, E, w)$ be a flow network constrained by $RX = 0$, with $X$ the flow on each edge and $R$ a constraint matrix of the flows. Let $\epsilon_i$ be the measured flow on the $i$-th edge.

**Query:** Compute

$$X^\text{max} = \arg \max_X \prod_{i=1}^m P(\epsilon_i | X_i),$$  \hspace{1cm} (3.2)

where $P(\epsilon_i | X_i) = N(\epsilon_i | X_i, \pi_i)$ is a normal distribution function given by

$$N(\epsilon_i | X_i, \pi_i) \equiv \sqrt{\pi_i / 2\pi} \exp \left\{ -\frac{\pi_i (X_i - \epsilon_i)^2}{2} \right\}$$

with $\pi_i \triangleq \frac{1}{\sigma_i^2}$ being the precision of the measurement device on the $i$-th edge.

Our formulation of the problem is based on several assumptions, which we now explicate. Let $\epsilon = \{\epsilon_1, ..., \epsilon_m\}$ where $\epsilon_i$ is the measured value on the $i$-th edge. To find the most probable flow we compute using Bayes’ rule

$$X^\text{max} = \arg \max_X P(X | \epsilon) = \arg \max_X \frac{P(\epsilon | X) \cdot P(X)}{P(\epsilon)}$$
Assuming $P(X)$ is constant for all the consistent configurations of $X$ and since $P(\epsilon)$ is also constant, we obtain

$$X^{\text{max}} = \arg \max_X P(\epsilon | X)$$  \hfill (3.3)

Assuming the measurement on each edge is unbiased and independent of measurements on other edges given the true flow on that edge, we obtain Eq. 3.2.

### 3.2 Various Solutions of MPFE

The MPFE problem can be solved for arbitrary flow networks in $O(m^3)$ steps by various well known techniques which we now explicate.

One technique immediately applicable for this problem is that of Linear Minimum Variance Unbiased estimation (LMVU). The following theorem from [9] provides a unified approach for the formulation and solution of diverse problems including MPFE as a special case:

**Theorem 3.1** If the data can be modeled as

$$x = H\theta + w$$

where $x$ is an $N \times 1$ vector of observations, $H$ is a known $N \times t$ observation matrix $(N > t)$ of rank $t$, $\theta$ is a $t \times 1$ vector of parameters to be estimated, and $w$ is an $N \times 1$ noise vector with pdf $N(0, C)$ then the LMVU estimator is

$$\hat{\theta} = (H^T C^{-1} H)^{-1} H^T C^{-1} x$$

and the covariance matrix is

$$C_\theta = (H^T C^{-1} H)^{-1}$$

For the MPFE problem, the $\theta$ vector represents a set of $t = m - n$ independent flows. Since the flow conservation constraints are linear, the flow on each edge is represented using $H\theta$. Since the computation of $\hat{\theta}$ and $C_\theta$ requires matrix inversions, it requires $O(m^3)$ time complexity and $O(m^2)$ space complexity. The main disadvantage of this solution for MPFE and other graph-based problems (such as inference in Bayesian networks) is the fact that the solution does not use the topology of the graph and is therefore less intuitive and, in some cases, less efficient, as we will demonstrate herein.

A second approach is to translate a given flow network into a Gaussian network such that each edge in the flow network becomes a variable in the Gaussian network and each flow constraint becomes a clique. Recall that in Gaussian networks each variable $v$ is distributed normal with a mean that is a linear function of the vertices that point into it and with a conditional variance that is fixed given the variables that point into $v$ [5, 10, 14]. This framework can be used in the limit to represent flow conservation constraints because when a variance of a variable tends to zero, the variable is simply a linear function of the variables that point into it. However, by transforming the flow network into a Gaussian network some independence assertions are lost and therefore the clique-tree algorithm [8] yields an $O(m^3)$ algorithm even in cases where an $O(m)$ algorithm can be found.

A third relevant approach is a reduction of the MPFE problem to the Minimal Cost Flow problem [1]. The input of a Minimal Cost Flow problem is a directed graph with a cost function $G_e(f_e)$ of the flow $f_e$ on each edge $e$. The output is an assignment of flows that minimizes the total cost $\sum G_e(f_e)$ and satisfies the flow conservation constraints. The MPFE problem is a special case of the Minimal Cost Flow problem because the cost function can be selected for edge $f_e$ to be $-\log P(\epsilon | X_i)$. Also this approach does not yield an $O(m)$ algorithm for the problem we address herein. Furthermore, this approach does not support the computation of expected precision of estimation, or the convergence rate to the true flow values, or the estimation of the conditional distribution functions from data.
Our approach is that of bucket elimination \[6\]. It is also of \(O(m^3)\) time complexity for general graphs. However, for polytrees we were able to develop an \(O(m)\) algorithm. The principle ideas of this algorithm are developed in the next section, which can be skipped without loss of continuity. More details of our algorithm can be found in \[16\].

### 3.3 Solving MPFE on Binary Polytrees

A polytree is a directed graph \(G\) such that the underlying undirected graph of \(G\), namely, the graph where each directed edge is replaced with an undirected edge, is a tree. A binary polytree is a polytree such that each internal node has at most two outgoing or incoming edges. In this section we develop an \(O(m)\) algorithm for the flow estimation problem, where the flow network \(G\) is a binary polytree.

#### 3.3.1 Foundations

We start with several lemmas describing relevant properties of a normal distribution \(N(e_i|X_i, \pi_i)\).

The first lemma restates a known property of normal distributions; Its one-line proof is brought here for completeness. The lemma shows that to maximize Eq. 3.2 one can maximize an equivalent function in which several measurements with various precisions of the true flow \(X\) on a link are replaced with a single equivalent measurement of \(X\). In the sequel we will always replace a set of measurement on an edge with its single equivalent measurement. It is convenient to denote the normal distribution \(N(e_i|X_i, \pi_i)\) with \(f_{e_i, \pi_i}(X_i)\).

**Lemma 3.2**

\[
\arg \max_X \prod_{j=1}^{k} f_{e_j, \tau_j}(X) = \arg \max_X f_{\mathcal{E}, \tau}(X)
\]

and where \(\mathcal{E} = \sum_{j=1}^{k} \tau_j e_j\) and \(\tau = \sum_{j=1}^{k} \tau_j\).

**Proof:**

\[
\prod_{j=1}^{k} f_{e_j, \tau_j}(X) = C_1 \prod_{j=1}^{k} \exp \left\{ -\frac{1}{2} \tau_j (X - e_j)^2 \right\} =
\]

\[C_2 \exp \left\{ -\frac{1}{2} \left( \sum_{j=1}^{k} \tau_j \right) \left( X^2 - 2X \sum_{j=1}^{k} \frac{\tau_j e_j}{\tau_j} \right) \right\} = C_3 \exp \left\{ -\frac{1}{2} \tau (X - \mathcal{E})^2 \right\} = C f_{\mathcal{E}, \tau}(X)
\]

(Where \(C_1\) through \(C_3\) are constants).

The second lemma provides a closed form maximum likelihood solution of the flow through an internal node which is adjacent to exactly three edges.

**Lemma 3.3** The maximum value for the product \(f_{\mu, \tau_i}(X_i) f_{\mu, \tau_r}(X_r)\) subject to

\[
(-1)^{d_i} X_i + (-1)^{d_r} X_r = X_p
\]

\(d_i, d_r \in \{0, 1\}\) (3.4)

equals

\[C f_{(-1)^{d_i} \rho_i + (-1)^{d_r} \rho_r, \tau_{d_i + d_r}}(X_p)\]

for every \(X_p\) where \(C, d_i\) and \(d_r\) are constants.
Proof: It suffices to compute $L(X_p)$ where

$$L(X_p) = \max_{X_1} f_{\mu_l, \tau_l}(X_l) f_{\mu_r, \tau_r}(X_r \mid X_l)$$

The value $X^*_p$ which maximizes $L(X_p)$ is given by:

$$X^*_p = \min_{X_1} \tau_l (X_l - \mu_l)^2 + \tau_r \left( (-1)^{d_r} X_p - (-1)^{d_l+1} X_l - \mu_r \right)^2.$$  

(3.5)

Differentiating with respect to $X_l$ and equating to zero yields

$$X^*_l = \frac{\tau_l \mu_l + (-1)^{d_l} \tau_r X_p - (-1)^{d_l+1} \tau_r \mu_r}{\tau_l + \tau_r}$$

(3.6)

which is a global minimum because 3.5 is quadratic. Consequently,

$$f_{\mu_l, \tau_l}(X^*_l) = C_l \exp \left\{ \frac{-\tau_l \mu_l^2}{2(\tau_l + \tau_r)} \left( X_p - \left((-1)^{d_l} \mu_r + (-1)^{d_l} \mu_l \right)^2 \right\}$$

and

$$f_{\mu_r, \tau_r}\left( (-1)^{d_r} \left( X_p - (-1)^{d_l} X^*_l \right) \right) = C_r f_{-1}^{d_r} \mu_r + (-1)^{d_l} \mu_l, \frac{\tau_r + \tau_l}{2(\tau_l + \tau_r)} \left( X_p \right).$$

Lemma 3.2 yields

$$L(X_p) = C f_{-1}^{d_l} \mu_l + (-1)^{d_r} \mu_r, \frac{\tau_r + \tau_l}{2(\tau_l + \tau_r)} \left( X_p \right).$$

We now demonstrate the importance of these lemmas for solving the MPFE problem for the simple binary tree in Figure 2. Our algorithms generalize this example.

![Figure 2: A Simple Binary Tree](image)

Let $X_p$, $X_r$, and $X_l$ be flows such that $X_p = X_r + X_l$. Let $e_p$, $e_r$, $e_l$ be normally-distributed measurements of the flows on these edges with known precisions $\tau_p$, $\tau_r$, and $\tau_l$, respectively. Using Eq 3.2, the most probable flow is given by:

$$\{X^*_p, X^*_r, X^*_l\} = \arg \max_{\{X_p, X_r, X_l\} \text{ s.t. } X_p = X_r + X_l} \prod_{i} f_{e_i, \tau_i}(X_i) \prod_{j} f_{\mu_j, \tau_j}(X_j)$$

in order to find $X^*_p$ it suffices to compute

$$X^*_p = \arg \max_{X_p} \prod_{i} f_{e_i, \tau_i}(X_i) \max_{X_r, X_l} f_{\mu_r, \tau_r}(X_r) f_{\mu_l, \tau_l}(X_l).$$  

(3.7)
using Lemma 3.3, Eq 3.7 becomes:

$$X_p^* = \arg \max_{X_p} f_{e_p, \tau_p} (X_p) f_{e_1 + e_r, \tau_1 + \tau_r} (X_p)$$

using Lemma 3.2,

$$X_p^* = \arg \max_{X_p} f_{e_p, \tau_p + (\tau_p + \tau_r)} f_{e_1 + e_r, \tau_1 + \tau_r} (X_p).$$

Therefore,

$$X_p^* = \frac{\epsilon_p \tau_p + (\epsilon_1 + \epsilon_r) \tau_1 \tau_r}{\tau_p + \frac{\tau_1 \tau_r}{\tau_1 + \tau_r}},$$

and the precision \(\tau_p^*\) of the estimator \(X_p^*\) is given by

$$\tau_p^* = \tau_p + \frac{\tau_r \tau_1}{\tau_1 + \tau_r}.$$

After \(X_p^*\) is found, the optimal value for \(X_1\) is available from Eq 3.6 and subsequently the optimal value for \(X_r\) is given by Eq. 3.4. Alternatively, we use Lemmas 3.2 and 3.3 to obtain

$$X_1^* = \arg \max_{X_1} f_{e_1, \tau_1 + \tau_r} f_{e_1 + e_r, \tau_1 + \tau_r} (X_1)$$

and therefore \(\tau_1^* = \tau_1 + \frac{\tau_r \tau_p}{\tau_p + \tau_r}\),

and similarly, obtain \(X_r^*\) and \(\tau_r^*\).

### 3.3.2 The Algorithm

In this section we develop an \(O(m)\) algorithm for MPFE on binary polytrees termed \textsc{bintree-mpfe}.

**Definition 1** Let \(G = (V, E, w)\) be a flow network and let \(e \in E\) be the edge connected to a source or a sink in \(G\). We call \(e\) an \(E\)-leaf of \(G\).

Recall that there are \(n\) linear constraints (one for each internal node). The algorithm consists of \(n\) steps. At each step, the algorithm uses Lemma 3.3 to eliminate two \(E\)-leaves with a common node \(r\) by adjusting the measurement and precision of measurement received on the third edge adjacent to \(r\). The process iterates until a single edge remains for which the optimal flow estimation as well as its precision are given by the adjusted measurement and precision of measurement for that edge.

Suppose that \(l_j, l_k\) are \(E\)-leaves with a common node, and are also adjacent to \(l_h\) such that \(X_j + X_k = X_h\). Let \(\epsilon_h, \epsilon_j, \epsilon_k\) and \(\tau_h, \tau_j, \tau_k\) be the measurements and the precisions of measurements on the edges \(l_h, l_j, l_k\), respectively. The elimination process consists of adjusting \(\epsilon_h\) and \(\tau_h\) as follows:

$$\epsilon'_h \leftarrow \frac{\epsilon_h \tau_h + (\epsilon_j + \epsilon_k) \tau_j \tau_k}{\tau_h + \frac{\tau_j \tau_k}{\tau_j + \tau_k}}, \quad \tau'_h \leftarrow \frac{\tau_h \tau_j}{\tau_j + \tau_k}.$$

The interpretation is that in order to find the optimal flow values for the rest of the graph, \(l_j\) and \(l_k\) may be eliminated by providing an additional measurement for \(l_h\) equal to \((\epsilon_j + \epsilon_k)\) with precision \(\frac{\tau_j \tau_k}{\tau_j + \tau_k}\). This interpretation holds due to Lemma 3.3. The entire algorithm, which we call \textsc{bintree-mpfe}, is summarized in Figure 3.

The next theorem proves the correctness of \textsc{bintree-mpfe} and analyzes its complexity.

**Theorem 3.4** Let \(G(V, E, w)\) be an instance of the MPFE problem where \(G\) is a binary polytree. \textsc{bintree-mpfe} correctly computes the most probable flow in time and space complexity of \(O(|E|)\).

**Proof:** Since \(G\) is a binary polytree, it is possible to find two \(E\)-leaves \(l_i\) and \(l_r\) adjacent to a third edge \(l_p\) such that \((-1)^{d_i} X_i + (-1)^{d_r} X_r = X_p\) with \(d_i, d_r \in \{0, 1\}\). Equation 3.2 may then be rewritten as:

$$X^\text{max} = \arg \max_{X \in \mathbb{R}} f_{e_1, \tau_1} (X_1) f_{e_r, \tau_r} (X_r) \prod_{i \neq l_i, l_r} f_{e_i, \tau_i} (X_i)$$
**Algorithm bintree-mpfe**

Input: An Instance of the MPFE problem. A queried edge $l_q$.
Output: The most probable flow on $l_q$ and its precision (i.e. $X^*_q$, $\tau^*_q$)

**Initialization:**
1. For each edge $l_i$, let $\epsilon_i$ be the measurement of the flow on that edge and let $\tau_i$ be the precision of measurement.
2. Let $G^f \leftarrow G$

**Repeat**

Select two E-leaves, $l_i, l_r$, adjacent to a third edge, $l_q$, such that $l, r \neq q$.
Select $d_l, d_r \in \{0, 1\}$ such that $(-1)^{d_l} X_l + (-1)^{d_r} X_r = X_p$.

$$
\epsilon_{lr} = (-1)^{d_l} \epsilon_l + (-1)^{d_r} \epsilon_r
$$

$$
\tau_{lr} = \frac{\tau_l + \tau_r + \epsilon_{lr}}{\tau_l + \tau_r}
$$

$$
\tau'_{lr} = \tau_l + \tau_r
\text{ } G' = G^f \setminus l_i, l_r
$$

**Until** a single edge, $l_q$, remains in $G'$

$X^*_q \leftarrow \epsilon'_{l_q}$

$\tau^*_q \leftarrow \tau'_{l_q}$

---

Figure 3: Algorithm bintree-mpfe

Noticing that $(-1)^{d_l} X_l + (-1)^{d_r} X_r = X_p$ is the only constraint in $RX = 0$ that relates $X_i$ with $X_r$, we write

$$
X^{\max} \setminus \{X_i, X_r\} = \arg\max_{s.t. R'X = 0} \prod_{i \notin \{l, r\}} f_{e_i, \tau_i} (X_i) \max_{s.t. (-1)^{d_l} X_l + (-1)^{d_r} X_r = X_p} f_{e_{lr}, \tau_{lr}} (X_l) f_{e_r, \tau_r} (X_r)
$$

where $R'$ is the constraints matrix excluding the constraint $(-1)^{d_l} X_l + (-1)^{d_r} X_r = X_p$. Using Lemma 3.3 we have:

$$
X^{\max} \setminus \{X_i, X_r\} = \arg\max_{s.t. R'X = 0} f_{e_r, \tau_r} (X_r) \prod_{i \notin \{l, r\}} f_{e_i, \tau_i} (X_i)
$$

with $\tau_{lr} = \frac{\tau_l + \tau_r + \epsilon_{lr}}{\tau_l + \tau_r}$ and $\epsilon_{lr} = (-1)^{d_l} \epsilon_l + (-1)^{d_r} \epsilon_r$. Using Lemma 3.2 the two terms depending on $X_p$ can be merged yielding,

$$
X^{\max} \setminus \{X_i, X_r\} = \arg\max_{s.t. R'X = 0} f_{e_{lr}, \tau_{lr}} (X_p) \prod_{i \notin \{l, r\}} f_{e_i, \tau_i} (X_i).
$$

Eq. 3.9 means that in order to find the optimal flow values for $X \setminus \{X_i, X_r\}$, it is sufficient to consider a smaller equivalent tree by removing the two E-leaves $X_i, X_r$ (along with the sources/sinks connected to these edges), and by updating the parameters of $f_{e_r} (X_p)$ as specified by Eq. 3.9 and by bintree-mpfe.

Algorithm bintree-mpfe consists of $n$ iterations. At each iteration the algorithm reduces the graph by two E-leaves $l_i, l_r$ and updates the parameters of one remaining edge $l_q$, which becomes an E-leaf (due to the elimination of its adjacent edges). After each iteration the graph remains a binary polytree. Therefore, this process may be iterated until a single edge remains. At each iteration the edges $l_i, l_r$ are selected to be different than $l_q$, the queried edge. Consequently, after $n$ iterations the
remaining edge is \( l_q \). The optimal flow value for this edge and its precision, \( X_q^*, \tau_q^* \) are simply given by the parameters of the remaining distribution \( f_{\epsilon_q^*,\tau_q^*}(X_q) \) as specified in \texttt{bintree-mpfe}.

It remains to analyze time and space complexity. Algorithm \texttt{bintree-mpfe} has \( n \) iterations. Each iteration requires \( O(1) \) operations. Since \( n \leq |E|/3 \) the time complexity is \( O(|E|) \). Space is used for storing \( 2n \) parameters, yielding space complexity \( O(|E|) \).

In order to find the most probable flow and its precision on every edge in the flow network, it is possible to run the above algorithm, each time for a different queried edge \( l_q \). This results in time complexity of \( O(|E|^2) \). However, an \( O(|E|) \) algorithm is obtained with some additional bookkeeping and using a message-passing framework similar to the one suggested for inference in Bayesian networks in [11]. This modified algorithm defines for each internal node \( \tau \) two buckets for each of its adjacent edges (6 buckets altogether). The 'Up' bucket for an edge \( l \) saves the best estimation for the edge's flow and its precision using measurements from all the edges contained in the subtree rooted at \( l \) and not containing other edges adjacent to \( \tau \). The 'Down' bucket saves the best estimation for \( l \) using the measurements from all the other edges.

Initially, all the buckets are empty except for the 'Up' buckets of internal nodes adjacent to E-leaves. These buckets are filled with the measurements received for the E-leaves and with their precisions. The algorithm proceeds in a distributed manner as follows: Each internal node monitors its 'Up' buckets. For each pair of 'Up' buckets that gets filled, the node fills the 'Down' bucket of the third adjacent edge with the flow and precision of flow using Lemma 4.2. Independently, each non E-leaf edge monitors the 'Down' buckets of its adjacent nodes. Once a 'Down' bucket gets filled, the edge combines the values from that bucket with the measurements received on the edge, using Lemma 4.1, and places the result on the 'Up' bucket of the other node adjacent to the edge. Eventually, all the buckets get filled. The optimal values for each edge is then readily given by combining the values on the corresponding 'Up' and 'Down' buckets using Lemma 4.1.

Since each internal node performs at most 3 operations and each edge performs at most 2 operations, and since the final computation is \( O(1) \) per edge, the time complexity of the algorithm is \( O(|E|) \). Since there are less than \( 4|E| \) buckets, the space complexity of the algorithm is \( O(|E|) \) as well.

### 4 Hierarchical Identification

We now develop a framework and efficient algorithms for the conservative fusion of consistent and inconsistent reports received for a group of objects into a single summary report. We present the notion of \textit{hierarchical identification} where each report may be a list of labels of various specificity. The number of vehicles on each track is assumed to have been estimated (say, as done in Section 3). These algorithms are developed as part of the flow-conserving identification component for the flow-conservation group tracking framework. However, unlike in Section 3, as a first step, the analysis in this section assumes that each track is solved independently of other tracks. A needed extension of this work which utilizes flow conservation constraints for each type along with an error model is currently being developed (see Section 5).

We consider a group of objects where each object can be described using labels of various specificity. Labels are arranged in a hierarchy according to their specificity. For example, Tank, IFV, Artillery Vehicle are labels for objects, and T88, T54 are more specific labels than Tank. An example for such a hierarchy is shown in Figure 4. A report is a list of labels associated with a group of objects, as reported by some identification device. For example, (T55, T88, T88, Artillery Vehicle) is a report about four objects. Reports do not associate which object in the group has been given a specific label. In our example, the report does not specify which object is a T55 and which is an Artillery Vehicle. The labels used in each report may be given in various levels of specificity. Furthermore, the information in the reports may be erroneous.
In this section, we first provide a formal description of the Hierarchy Matching problem (Section 4.1). Then we describe an intuitive $O(lm^2n)$ algorithm (Section 4.2) where $m$ is the number of objects, $n$ is the number of reports and $l$ is the number of labels in the hierarchy.

In a companion technical report [17], we provide a more efficient $O(nl + m)$ algorithm for this problem, present the Minimal Modification Hierarchy Matching problem, and provide an algorithm for its solution for star hierarchies.

### 4.1 Problem formulation

Suppose $L$ is a finite set with a partial order $>$. Each element $l_i \in L$ is called a label and $l_1 > l_2$ is interpreted as $l_2$ is more specific than $l_1$, or equivalently, $l_1$ is more general than $l_2$. For example, $l_1 =$ Tank is more general than $l_2 =$ T88.

**Definition 2** A hierarchy $H$ is a directed tree $(L, E)$ where each node in $L$ is a label, and if $(l_i, l_j)$ is an edge in $E$, then label $l_2$ is more specific than label $l_1$.

For example, consider the military-domain hierarchy in Figure 4. Tank, IFV, Artillery Vehicle are labels for objects, and T88, T54 are more specific labels than Tank.

We use the term multiset to denote a set with repetitions. A multiset $A$ is compatible wrt a hierarchy $H = (L, E)$ if all elements of $A$ belong to $L$ and reside on a single directed path in $H$.

We define the Hierarchy Matching problem as follows.

**Instance:** Let $H = (L, E)$ be a hierarchy. Let $R_j = \{l_{j1}, ..., l_{jm}\}$, $j = 1, ..., n$, be multisets with elements from $L$ and let $R$ be the union $R_1 \cup R_2 \cup ... R_n$ (with repetitions) of the multisets $R_j$. In other words, the multiset $R$ contains $mn$ elements from $L$, not necessarily distinct.

**Query:** Partition $R$ into disjoint multisets $A_1, ..., A_m$ each containing exactly one element from each $R_j$, $j = 1, ..., n$, such that for $1 \leq i \leq m$, $A_i$ is compatible wrt $H$; or output a valid statement that no such partition exists.

When such a partition exists, the multiset $R$ is said to be consistent wrt $H$ and $A_1, ..., A_m$ is called a consistent partition of $R$. Otherwise it is inconsistent wrt $H$.

The Hierarchy Matching problem is motivated by the following interpretation of its components. Consider a group of objects, in which the number of objects $m$ is known. Each object can be described using a label. The set of possible labels is denoted by $L$. Labels are arranged in a hierarchy tree as in Figure 4. A report $R_j$ is a list of labels associated with a group of $m$ objects, as reported by some identification device. For example, (T55, T88, T88, Artillery Vehicle) is a report about four objects. Reports do not associate which object in the group has been given a specific label. In our example the report does not specify which object is a T55 and which is an Artillery Vehicle. The labels used in each report may be given in various levels of specificity.

The Hierarchy Matching problem seeks a consistent interpretation of the reports $R_1, ..., R_n$ if such exists by matching the labels of each object across the $n$ reports. If the reports are consistent, it is possible to summarize the information they convey using a consensus report.

**Definition 3** Given a consistent partition $A_1, ..., A_m$ we say that the consensus report is the multiset constructed of the most specific labels in each of the multisets $A_i$, $i = 1, ..., m$.

Consider using Figure 4 the reports: $R_1 = (Equipment, Tank)$, $R_2 = (Military, IFV)$. It is readily seen that these reports may be partitioned into the consistent partition: $A_1 = (Military, Tank)$, $A_2 = (Equipment, IFV)$. In this case, the consensus report is the most specific label from $A_1$ and the most specific label from $A_2$ that is (Tank, IFV).

Now consider a third report $R_3 = (Military, Truck)$. Although each pair of the reports $R_1, R_2, R_3$ is consistent, no consistent partition exists for the three reports. Consequently the reports are inconsistent and a consensus report does not exist.

The Hierarchy Matching problem is a restricted version of an N-Dimensional Matching problem, defined as follows. Let $G = (V, E)$ be a graph and let $V_1, ..., V_n$ be a partition of $V$ into disjoint sets each containing exactly $m$ nodes. The problem is to partition $V$ into $m$ disjoint sets $A_1, ..., A_m$ such that each $A_i$ contains exactly one node from each set $V_j$ (namely, $n$ nodes) and for
Figure 4: A Hierarchy of Labels

every two nodes u, v in A; there exists an edge (u, v) in E. When n = 2, this becomes the bi-partite graph matching problem. When n > 2, the decision version of this problem is NP complete [7]. However, for the Hierarchy Matching problem, which can be viewed as a restricted version of the n-dimensional matching problem, we developed a polynomial algorithm with time complexity $O(n(l + m))$.

4.2 Algorithm HIERARCHY-MATCHING

The algorithm described in this section is based on the fact that due to the constraint implied by a hierarchy the desired multisets $A_i$, $1 \leq i \leq m$ can be found sequentially in a greedy manner.

**Definition 4** Let $R$ be a multiset with elements from a set $L$ with partial order $>$. An element $l_1$ is called a least element of $R$ if $l_1 \in R$ and there is no other label $l_2 \in R$ such that $l_1 > l_2$.

**Definition 5** Let $l, l_1$ be elements of a set $L$ with partial order $>$, such that $l \rhd l_1$, and let $R$ be a multiset with elements from $L$. The element $l_1$ is called a least element of $R$ wrt $l$ if $l_1 \in R$ and there is no other element $l_2 \in R$ such that $l_1 > l_2 > l$.

**Definition 6** Let $(H, R_1, ..., R_n)$ be an instance of the Hierarchy Matching problem and let the multiset $R$ be the union $R_1 \cup R_2 \cup ... \cup R_n$ (with repetitions). Let $A = \{a_1, ..., a_n\}$ be a multiset with elements from $R$ such that: (a) The multiset $A$ is compatible wrt $H$ (b) There is an index $k \in \{1, ..., n\}$ such that $a_k$ is a least element of $R$ (c) For $j = 1, ..., n$, label $a_j$ is a least element of $R_j$ wrt $a_k$. Then $A$ is said to be a reducible multiset wrt to $(H, R_1, ..., R_n)$.

**Lemma 4.1** Let $(H, R_1, ..., R_n)$ be an instance of the Hierarchy Matching problem where $R$ is the union $R_1 \cup R_2 \cup ... \cup R_n$ (with repetitions). Let $A = \{a_1, ..., a_n\}$ be a reducible multiset wrt $(H, R_1, ..., R_n)$. Then $R$ is consistent iff $R^* \equiv R \setminus A$ is consistent.

**Proof:** First assume that the multiset $R^*$ is consistent. Consequently it may be partitioned into disjoint compatible multisets wrt $H$. Since $A$ is a reducible multiset it is also compatible wrt $H$ by requirement (a) of Definition 6. Since $R = R^* \cup A$ with repetitions, the multiset $R$ must be consistent as well.

Now assume that $R$ is consistent. We will prove that $R^*$ may be partitioned into $m - 1$ disjoint compatible multisets. By definition, $R$ may be partitioned into $m$ disjoint compatible multisets $A_1, ..., A_m$ each containing exactly one element from each $R_j$, $j = 1, ..., n$. Clearly, if one of the
multisets $A_1, \ldots, A_m$ is identical to the reducible multiset $A$ then the proof is complete. However, in general, this may not be the case. We address this difficulty in the following.

Since $A = \{a_1, \ldots, a_n\}$ is a reducible multiset, then by requirement (b) of Definition 6 there is an index $k \in [1, \ldots, n]$ such that $a_k$ is a least element of $R$. Since $R$ is partitioned into the multisets $A_1, \ldots, A_m$, one of these multisets must also contain the label $a_k$. Denote this multiset $A^* = \{a_1^*, \ldots, a_n^*\}$.

Apart from the label $a_k$ which resides in both multisets, $A^*$ and $A$, the labels $a_j, j \neq k$ need not equal the corresponding $a_j^*$ labels. However, both $A^*$ and $A$ are compatible multisets containing $a_k$, meaning that all the labels $\{a_1, \ldots, a_n\}$ and $\{a_1^*, \ldots, a_n^*\}$ must reside on the same directed path in $H$. Furthermore, requirement (c) of Definition 6 yields $a_j^* \geq a_j$ for $j = 1, \ldots, n$. Since $a_j^*$ is more general than $a_j$ it may replace $a_j$ in any multiset without affecting this multiset compatibility. Consequently it is possible to locate for $j = 1, \ldots, n$ the label $a_j$ in the reports $A_1, \ldots, A_m$ and swap it with $a_j^*$. The result is a consistent partition of $R$ into $m$ compatible multisets of which one is identical to $A$.

Lemma 4.1 implies that given an instance of the hierarchy matching problem it is possible to pursue equivalent problems with reduced size by removing reducible-multisets. The following Lemma proves that reducible-multisets can always be found in a consistent set of reports.

**Lemma 4.2** Let $(H, R_1, \ldots, R_n)$ be an instance of the hierarchy matching problem where $R$ is the union $R_1 \cup R_2 \cup \ldots R_n$ (with repetitions). Let $a$ be a least element of $R$. Then if $R$ is consistent, there exists a reducible multiset $A = \{a_1, \ldots, a_n\}$ wrt $(H, R_1, \ldots, R_n)$ such that $a \in A$.

**Proof:** Since $R$ is consistent it may be partitioned into $m$ disjoint compatible multisets $A_1, \ldots, A_m$, each containing exactly one element from each $R_j, j = 1, \ldots, n$. Since $R$ is partitioned into the multisets $A_1, \ldots, A_m$, one of these multisets must contain the label $a$. Denote this multiset $A^* = \{a_1^*, \ldots, a_n^*\}$. The multiset $A^*$ is compatible and contains a least element of $R$. Indeed, it is possible that for some report $R_j$, $a_j^*$ is not a least element wrt $a$. However, this may be easily fixed by replacing $a_j^*$ with the least element of $R_j$ wrt $a$.

Lemmas 4.1 and 4.2 yield the following algorithm for hierarchy matching: At each iteration the algorithm finds a reducible-multiset. This can be done if the set of reports is consistent (Lemma 4.2). By Lemma 4.1 it is valid to remove the reducible multiset if it exists. This process is iterated. After $m$ iterations either a consistent partition is found or such a partition does not exist and therefore inconsistency is declared. This algorithm is shown in Figure 5.

**Theorem 4.3** Let $(H, R_1, \ldots, R_n)$ be an instance of the Hierarchy Matching problem where $R$ is the union $R_1 \cup R_2 \cup \ldots R_n$ with repetitions. Then, algorithm hierarchy-matching solves the problem in time complexity $O(m^3 n)$ and space complexity $O(l + mn)$.

**Proof:** Denote by $R^{(i)}$ the set of reports before the $i$th iteration. At each iteration the algorithm attempts to find a reducible multiset in $R^{(i)}$. Recall that if $R^{(i)}$ is consistent it is always possible to find a reducible-multiset by Lemma 4.2. Therefore, if a reducible-multiset is not found, the multiset $R^{(i)}$ is inconsistent. By Lemma 4.1, if $R^{(i-1)}$ is consistent than so must be $R^{(i)}$. Consequently, the inconsistency of $R^{(i)}$ yields inconsistency of $R^{(i-1)}$ and so on until we obtain that $R^{(1)} = R$ is inconsistent as declared by the algorithm.

Consider now the case when the algorithm does not output a statement that $R$ is consistent. This case occurs if $R^{(m)}$ is found to be consistent which implies that $R^{(m-1)}$ is consistent and therefore that $R^{(m-2)}$ is consistent and thus $R^{(1)} = R$ is consistent as well. Furthermore, a consistent partition to reducible multisets is readily given.

We now analyze time and space complexity. At each iteration of FindReducibleMultiset the algorithm finds a least element in $R$ and then a least element in each of the remaining $n-1$ reports. Consequently, a naive implementation for FindReducibleMultiset yields time complexity $O(l mn)$ (where it is assumed that the query; given labels $l_1, l_2$ is $l_1 > l_2$? is $O(l)$). This function is called (at most) $m$ times hence the entire algorithm has time complexity $O(l m^3 n)$. Space is needed to store $R$ and the multisets $A_i$ as well as the hierarchy labels which yields a space complexity of $O(l + mn)$. //
5 Discussion and Future Work

In this paper we described a novel framework for group tracking and identification termed flow conservation group tracking, which we believe to be a preferred extension of the methodology for tracking single targets to tracking groups. Our framework integrates local kinematic measurements with flow conservation constraints and is based to a large extent on the ability to estimate and match the number and types of targets in each group, in addition to kinematics.

We envision several major improvements to our framework. First, we expect the framework to be augmented to support multiple hypothesis tracking. However, unlike the framework suggested in [11], each hypothesis must also include the estimation of number of vehicles on each of the tracks in addition to kinematics. Second, we believe that a tighter integration may be obtained between the measurement-to-track and the flow-estimation components. Finally, since flow conservation group tracking may be applied in several resolutions (depending on the clustering attributes) it will be interesting to characterize, given an arena, optimal attributes for clustering so as to provide a more accurate tracking procedure.

Each of the framework’s components may also be improved independently. An improvement to the flow estimation component may be the development of a methodology that is not based on normal measurement errors. Our initial results reported in [16] are very encouraging showing that our algorithm works even when data is generated from various distributions. Our initial experiments show that flow conservation provides a constant factor asymptotic improvement vs. sole local estimates regardless of sample size. Characterizing this improvement as a function of the topology may also be interesting. Also, it seems possible to improve the flow estimation algorithm for special graph topologies. We have already developed an algorithm that solves the MPFE problem in time complexity $O(|E|^3)$ provided that the flow network is a polytree and we believe it may be possible to extend this improvement to other network topologies as well for which our algorithm's current complexity is $O(|E|^3)$. This improvement may be helpful in the application of flow conservation group tracking in large scale real-time scenarios.

The main improvement we envision to the identification module is related to the exploration of both the Hierarchy Matching and the Min Modification Hierarchy Matching problems
in the context of global estimation in flow networks. Instead of providing a summary report that is solely based on measurements received on a single edge, future algorithms will generate a summary report per edge in a global fashion that uses reports on other edges in the flow network to correct its estimates, analogous to the way we estimate the number of objects in MFPE. Another improvement includes introducing an error model and a modified algorithm that provides the most likely summary report per edge in the flow network - both locally and globally.

Acknowledgment

The authors thank David Reinitz for useful conversations and Mark Matusevich for developing with us the use of minimum cost flow algorithms for the measurement-to-track component.

References