A Fully Automated Distributed Multiple-Target Tracking and Identity Management Algorithm

Songhwai Oh *
Univ. of California, Berkeley, CA 94720, U.S.A.

Inseok Hwang †
Purdue University, West Lafayette, IN 47906, U.S.A.

Kaushik Roy ‡
Stanford University, Stanford, CA 94305 U.S.A.

Shankar Sastry §
Univ. of California, Berkeley, CA 94720, U.S.A.

In this paper, we consider the problem of tracking multiple targets and managing their identities in sensor networks. Each sensor is assumed to be able to track multiple targets, manage the identities of targets within its surveillance region, and communicate with its neighboring sensors. The problem is complicated by the fact that the number of targets within the surveillance region of a sensor changes over time. We propose a scalable distributed multiple-target tracking and identity management (DMTIM) algorithm that can track multiple targets and manage their identities efficiently in a distributed sensor network environment. DMTIM finds a globally consistent solution by maintaining local consistency among neighboring sensors. DMTIM consists of data association, multiple-target tracking, identity management, and information fusion. The data association and multiple-target tracking problems are efficiently solved by Markov chain Monte Carlo data association (MCMCDA) which can track an unknown number of targets. DMTIM manages identities of targets by incorporating local information and maintains local consistency among neighboring sensors via information fusion.

I. Introduction

Recent advances in sensor technology and wireless communication have led to the concept of a sensor network. A sensor network is a network of local sensor nodes which have sensing, processing, and communication capabilities. Sensor networks have received growing interest in a variety of applications that include battlefield surveillance and enemy tracking in military applications, and habitat monitoring, environment observation, and traffic surveillance in civilian applications (see Ref. 1,2 and references therein). To fully exploit the capability of sensor networks, algorithms for sensor networks should be *scalable*, *i.e.*, adding/deleting sensors or targets into a sensor network can be handled efficiently, and *distributed*, *i.e.*, the algorithm can be implemented in individual sensors. In this paper, we propose a scalable distributed multiple-target tracking and identity management (DMTIM) algorithm which can keep track of multiple maneuvering targets and their identities in a sensor network.

In Ref. 3, the authors have developed a scalable distributed multiple-target identity management (DMIM) algorithm which can maintain the identities of multiple maneuvering targets in sensor networks. The proposed identity management algorithm has a capability to reduce the uncertainty of the overall system by incorporating local information about the identity of a target collected by each sensor. Different information fusion algorithms have been investigated to get the global information of the system from information provided by local sensors under a variety of sensor network scenarios. However, the identity management algorithm works for the cases in which the number of targets in a sensor network is known and constant and their trajectories are available to local sensors. In this paper, we relax the above assumptions and extend the algorithm in Ref. 3 to a distributed multiple-target tracking and identity management (DMTIM) algorithm which can track unknown, time-varying numbers of maneuvering targets and their identities in a sensor network.

Each sensor monitors its surveillance region and tracks multiple targets in a cluttered environment. However, the traditional multiple-target tracking algorithms such as the joint probabilistic data association (JPDA) filter[4] and multiple hypothesis tracker (MHT)[5] are not suitable for sensor networks since the track initiation and termination

*Dept. of EECS, Univ. of California, Berkeley. sho@eecs.berkeley.edu.
†School of Aeronautics and Astronautics, Purdue University. ihwang@purdue.edu.
‡Department of Electrical Engineering, Stanford University. kroy1@stanford.edu.
§Dept. of EECS, Univ. of California, Berkeley. sastry@eecs.berkeley.edu.

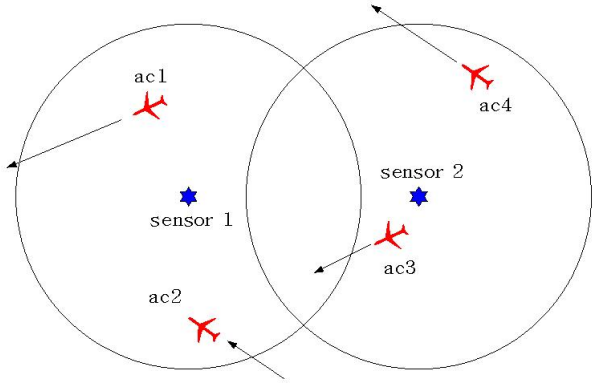

Figure 1. A distributed multiple-target tracking and identity management scenario for a two-sensor network.

are difficult with JPDA and both JPDA and MHT require large memory and computation cycles (the time and space complexities of MHT are higher than those of JPDA). Since MHT can initiate and terminate tracks, the tracking task can be easily distributed in a network of sensors, whereas this is difficult with JPDA. In Ref. 6, a distributed tracking algorithm based on MHT is developed for multiple sensors. But the approach demands large computational power and large amount of memory on each sensor.

In Ref. 7, Markov chain Monte Carlo data association (MCMCDA) is presented. MCMCDA can track an unknown number of targets in real-time and is an approximation to the optimal Bayesian filter. It has been shown that MCMCDA is computationally efficient compared to MHT and outperforms MHT under extreme conditions, such as a large number of targets in a dense environment, low detection probabilities, and high false alarm rates.[7] MCMCDA is suitable for sensor networks since it can autonomously initiate and terminate tracks and track an unknown number of targets. Also, it has been shown that MCMCDA is robust against communication delays, *i.e.*, out-of-sequence measurements.[8] Furthermore, the single-scan version of MCMCDA finds an approximate solution to JPDA in polynomial time.[9] Since the exact calculation of association probabilities in JPDA at each stage is NP-hard,[10] MCMCDA is well-suited for computing the mixing matrix of association probabilities in DMTIM when the number of targets is large. In DMTIM, each sensor can efficiently track an unknown number of targets using MCMCDA and the identities of detected targets are managed using DMIM in a distributed manner.

This paper is organized as follows: The overview of distributed multiple-target tracking and identity management (DMTIM) is presented in Section II. In Section III, we formally state the multiple-target tracking problem and its probabilistic model. The Markov chain Monte Carlo data association (MCMCDA) algorithm, which is an essential part of DMTIM, is described in Section IV. Then the components of DMTIM including data association, multiple-target tracking, identity management and information fusion are presented in Section V. We demonstrate and evaluate the DMTIM algorithm in simulation in Section VI.

II. Distributed Multiple-Target Tracking and Identity Management (DMTIM)

The main focus of this paper is the problem of tracking multiple targets and managing their identities in sensor networks. Each sensor is assumed to have its own surveillance region, and an ability to communicate with its neighboring sensors. A simple two-sensor example is shown in Figure 1 in which the circles represent the surveillance regions of the sensors. Each sensor is assumed to have the capability to track multiple targets and manage the identities of targets within its surveillance region. The problem gets complicated since the number of targets within the surveillance region of a sensor changes over time. For example, some targets may come from the surveillance regions of neighboring sensors, some targets may have not yet been registered into the identity management system, and some targets may leave the surveillance region of the current sensor. For a large network, a centralized approach is not feasible and we must seek for a scalable distributed solution which maintains local consistency among neighboring sensors, leading to a globally consistent solution.

In this paper, we propose a scalable distributed multiple-target tracking and identity management (DMTIM) algorithm that can track an unknown and time-varying number of maneuvering targets and manage their identities efficiently in a distributed sensor network. The structure of DMTIM is shown in Figure 2. At each sensor, *Multiple-Target Tracking (Data Association)* in *MTIM* estimates the number of targets and tracks of all detected targets in its

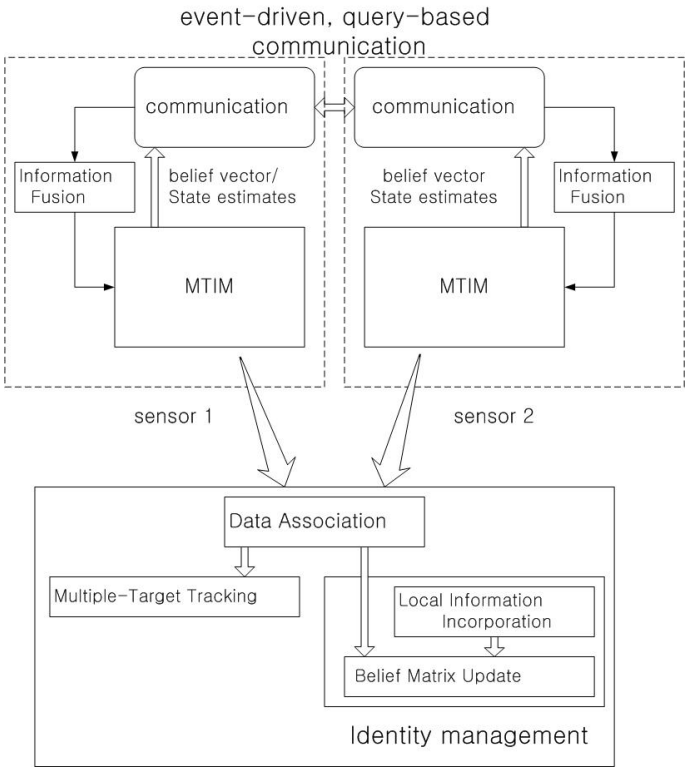

Figure 2. The structure of the distributed multiple-target tracking and identity management (DMTIM) algorithm for a two-sensor example.

survaillance region, as well as the mixing matrix and local information that are later used by *Identity Management* to update the sensor's belief matrix. Then the neighboring sensors exchange local state estimates and belief matrix by communication with each other and local consistency is maintained by *Information Fusion*, leading to global consistency.

In the remainder of this paper, the building blocks of DMTIM are described in detail. But, first, we describe the multiple-target tracking problem and Makov chain Monte Carlo data association (MCMCDA).

III. Multiple-Target Tracking

The *Multiple-Target Tracking (Data Association)* block of DMTIM needs to be able to track an unknown number of targets, since the number of targets in each sensor's surveillance region changes over time. Hence, we can not apply the conventional multiple-target tracking formulation such as JPDA, in which a fixed number of targets is assumed. In this section, we describe a general formulation which allows the uncertainties in the number of targets and appearance and disappearance times of targets.

A. Problem Formulation

Let $T \in \mathbb{Z}^+$ be the duration of surveillance. Let K be the number of objects that appear in the surveillance region $\mathcal{R}$ during the surveillance period. Each object k moves in $\mathcal{R}$ for some duration $[t_{\rm i}^k, t_{\rm f}^k] \subset [1, T]$. Notice that the exact values of K and $\{t_{\rm i}^k, t_{\rm f}^k\}$ are unknown. Each object arises at a random position in $\mathcal{R}$ at $t_{\rm i}^k$, moves independently around $\mathcal{R}$ until $t_{\rm f}^k$ and disappears. At each time, an existing target persists with probability $1 - p_{\rm z}$ and disppears with probability $p_{\rm z}$. The number of objects arising at each time over $\mathcal{R}$ has a Poisson distribution with a parameter $\lambda_{\rm b} V$ where $\lambda_{\rm b}$ is the birth rate of new objects per unit time, per unit volume, and V is the volume of $\mathcal{R}$. The initial position of a new object is uniformly distributed over $\mathcal{R}$.

Let $F^k : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ be the discrete-time dynamics of the object k, where n_x is the dimension of the state variable, and let $x^k(t) \in \mathbb{R}^{n_x}$ be the state of the object k at time t. The object k moves according to

$$x^k(t+1) = F^k(x^k(t)) + w^k(t), \qquad \text{for } t = t_{\rm i}^k, t_{\rm i}^k + 1 \ldots, t_{\rm f}^k - 1, \tag{1}$$

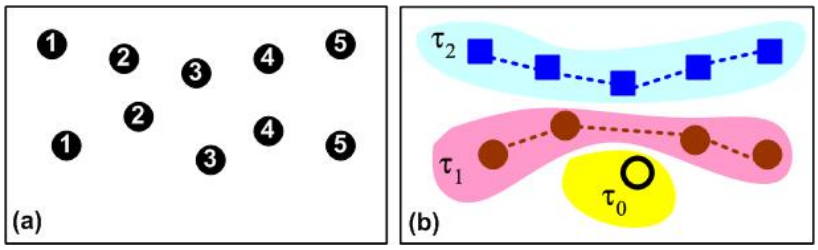

Figure 3. (a) An example of measurements Y (each circle represents a measurement and numbers represent measurement times); (b) an example of a partition ω of Y

where $w^k(t) \in \mathbb{R}^{n_x}$ are white noise processes. The white noise process is included to model non-rectilinear motions of targets. The noisy observation (or measurement[a]) of the state of the object is measured with a detection probability p_d. Notice that, with probability $1-p_\mathrm{d}$, the object is not detected and we call this a missing observation. There are also false alarms and the number of false alarms has a Poisson distribution with a parameter $\lambda_\mathrm{f} V$ where λ_f is the false alarm rate per unit time, per unit volume. Let $n(t)$ be the number of observations at time t, including both noisy observations and false alarms. Let $y^j(t) \in \mathbb{R}^{n_y}$ be the j-th observation at time t for $j = 1, \ldots, n(t)$, where n_y is the dimension of each observation vector. Each object generates a unique observation at each sampling time if it is detected. Let $H^j : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ be the observation model. Then the observations are generated as follows:

$$y^j(t) = \begin{cases} H^j(x^k(t)) + v^j(t) & \text{if } j\text{-th measurement is from } x^k(t) \\ u(t) & \text{otherwise,} \end{cases} \tag{2}$$

where $v^j(t) \in \mathbb{R}^{n_y}$ are white noise processes and $u(t) \sim \mathrm{Unif}(\mathcal{R})$ is a random process for false alarms. We assume that targets are indistinguishable in this paper, but if observations include target type or attribute information, the state variable can be extended to include target type information. The multiple-target tracking problem is to estimate K, $\{t_\mathrm{i}^k, t_\mathrm{f}^k\}$ and $\{x^k(t) : t_\mathrm{i}^k \le t \le t_\mathrm{f}^k\}$, for $k = 1, \ldots, K$, from observations.

B. Solutions to the Multiple-Target Tracking Problem

Let $y(t) = \{y^j(t) : j = 1, \ldots, n(t)\}$ be all measurements at time t and $Y = \{y(t) : 1 \le t \le T\}$ be all measurements from $t = 1$ to $t = T$. Let Ω be a collection of partitions of Y such that, for $\omega \in \Omega$,

1. $\omega = \{\tau_0, \tau_1, \ldots, \tau_K\}$;
2. $\bigcup_{k=0}^{K} \tau_k = Y$ and $\tau_i \cap \tau_j = \emptyset$ for $i \neq j$;
3. τ_0 is a set of false alarms;
4. $|\tau_k \cap y(t)| \le 1$ for $k = 1, \ldots, K$ & $t = 1, \ldots, T$; and
5. $|\tau_k| \ge 2$ for $k = 1, \ldots, K$.

An example of a partition is shown in Figure 3 and ω is also known as a *joint association event* in literature. Here, K is the number of tracks for the given partition $\omega \in \Omega$ and $|\tau_k|$ denotes the cardinality of the set τ_k. We call τ_k a track when there is no confusion although the actual track is the set of estimated states from the observations τ_k. However, we assume there is a deterministic function that returns a set of estimated states given a set of observations, so no distinction is required. The fourth requirement says that a track can have at most one observation at each time, but, in the case of multiple sensors with overlapping sensing regions, we can easily relax this requirement to allow multiple observations per track. A track is assumed to contain at least two observations since we cannot distinguish a track with a single observation from a false alarm, assuming $\lambda_\mathrm{f} > 0$. For special cases, in which $p_\mathrm{d} = 1$ or $\lambda_\mathrm{f} = 0$, the definition of Ω can be adjusted accordingly.

Let $e(t-1)$ be the number of targets from time $t-1$, $z(t)$ be the number of targets terminated at time t and $c(t) = e(t-1) - z(t)$ be the number of targets from time $t-1$ that have not terminated at time t. Let $a(t)$ be the number of new targets at time t, $d(t)$ be the number of actual target detections at time t and $g(t) = c(t) + a(t) - d(t)$

[a]Note that the terms *observation* and *measurement* are used interchangeably in this paper.

be the number of undetected targets. Finally, let $f(t) = n(t) - d(t)$ be the number of false alarms. It can be shown that the posterior of ω is:

$$P(\omega|Y) \propto P(Y|\omega) \prod_{t=1}^{T} p_{\mathrm{z}}^{z(t)} (1-p_{\mathrm{z}})^{c(t)} p_{\mathrm{d}}^{d(t)} (1-p_{\mathrm{d}})^{g(t)} \lambda_{\mathrm{b}}^{a(t)} \lambda_{\mathrm{f}}^{f(t)} \tag{3}$$

where $P(Y|\omega)$ is the likelihood of measurements Y given ω, which can be computed based on the chosen dynamic and measurement models.

There are two major approaches to solve the multiple-target tracking problem:[11] *maximum a posteriori* (MAP) and Bayesian (or *minimum mean square error* (MMSE)) approaches. The MAP approach finds a partition of observations such that $P(\omega|Y)$ is maximized and estimates states of targets based on the partition which maximizes $P(\omega|Y)$. The MMSE approach seeks the conditional expectations such as $\mathbb{E}(x_t^k|Y)$ to minimize the expected (square) error. However, when the number of targets is not fixed, a unique labeling of each target is required to find $\mathbb{E}(x_t^k|Y)$ under the MMSE approach. In this paper, we take the MAP approach to the multiple-target tracking problem for its convenience.

IV. Markov Chain Monte Carlo Data Association (MCMCDA)

This section presents an algorithm for solving the multiple-target tracking problem described in Section III. The algorithm described in this section is the computational engine of the *Multiple-Target Tracking (Data Association)* block of DMTIM.

A. Markov chain Monte Carlo

Markov chain Monte Carlo (MCMC) plays a significant role in many fields such as physics, statistics, economics, and engineering.[12] In some cases, MCMC is the only known general algorithm that finds a good approximate solution to a complex problem in polynomial time.[13] MCMC techniques have been applied to complex probability distribution integration problems, counting problems, and combinatorial optimization problems.[12,13]

MCMC is a general method to generate samples from a distribution π on a space Ω by constructing a Markov chain $\mathcal{M}$ with states $\omega \in \Omega$ and stationary distribution $\pi(\omega)$. We now describe an MCMC algorithm known as the Metropolis-Hastings algorithm. If we are at state $\omega \in \Omega$, we propose $\omega' \in \Omega$ following the proposal distribution $q(\omega, \omega')$. The move is accepted with an acceptance probability $A(\omega, \omega')$ where

$$A(\omega, \omega') = \min\left(1, \frac{\pi(\omega')q(\omega', \omega)}{\pi(\omega)q(\omega, \omega')}\right), \tag{4}$$

otherwise the sampler stays at ω. With this construction, the detailed balance condition is satisfied, *i.e.*, for all $\omega, \omega' \in \Omega$ with $\omega' \neq \omega$,

$$Q(\omega, \omega') = \pi(\omega)P(\omega, \omega') = \pi(\omega')P(\omega, \omega'), \tag{5}$$

where $P(\omega, \omega') = q(\omega, \omega')A(\omega, \omega')$ is the transition probability from ω to ω'.

If $\mathcal{M}$ is irreducible and aperiodic, then $\mathcal{M}$ converges to its stationary distribution by the ergodic theorem.[14] Hence, for a given bounded function $f : \Omega \to \mathbb{R}^m$, the sample mean $\hat{f} = \frac{1}{N}\sum_{n=1}^{N} f(\omega_n)$, where ω_n is the state of $\mathcal{M}$ at time t, converges to $\mathbb{E}_\pi f(\omega)$ as $N \to \infty$. Notice that (4) requires only the ability to compute the ratio $\pi(\omega')/\pi(\omega)$, avoiding the need to normalize π.

B. MCMCDA

The MCMC data association (MCMCDA) algorithm is described in Algorithm 1. MCMCDA is an MCMC algorithm whose state space is Ω described in Section III-B and whose stationary distribution is the posterior (3). The proposal distribution for MCMCDA consists of five types of moves (a total of eight moves). They are

1. birth/death move pair;
2. split/merge move pair;
3. extension/reduction move pair;
4. track update move; and

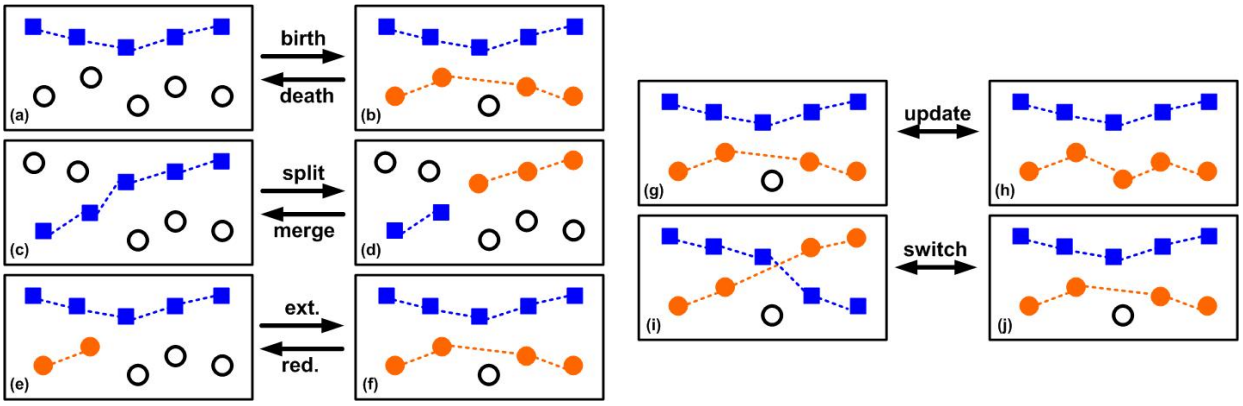

Figure 4. Graphical illustration of MCMCDA moves (associations are indicated by dotted lines and hollow circles are false alarms)

5. track switch move.

The MCMCDA moves are graphically illustrated in Figure 4. For detail description of each move, see Ref. 7. The inputs for MCMCDA are the set of all observations Y, the number of samples n_{mc}, the initial state ω_{init}, and a bounded function $X : \Omega \to \mathbb{R}^m$. At each step of the algorithm, ω is the current state of the Markov chain. The acceptance probability $A(\omega, \omega')$ is defined in (4) where $\pi(\omega) = P(\omega|Y)$ from (3). The output $\hat{X}$ approximates the MMSE estimate $\mathbb{E}_\pi X$ and $\hat{\omega}$ approximates the MAP estimate $\arg\max P(\omega|Y)$. The computation of $\hat{\omega}$ can be considered as simulated annealing at a constant temperature. Notice that MCMCDA can provide both MAP and MMSE solutions to the multiple-target tracking problem.

The Markov chain $\mathcal{M}$ designed by Algorithm 1 is irreducible and aperiodic.[7] In addition, the transitions described in Algorithm 1 satisfy the detailed balance condition since it uses the Metropolis-Hastings kernel (4). Hence, by the ergodic theorem,[14] the chain converges to its stationary distribution.[11] It has been shown that MCMCDA is computationally efficient compared to MHT and outperforms MHT under extreme conditions, such as a large number of targets in a dense environment, low detection probabilities, and high false alarm rates.[7]

Algorithm 1 MCMCDA

Input: $Y, n_{\text{mc}}, \omega_{\text{init}}, X : \Omega \to \mathbb{R}^m$
Output: $\hat{\omega}, \hat{X}$
$\omega = \omega_{\text{init}}$; $\hat{\omega} = \omega_{\text{init}}$; $\hat{X} = 0$
for $n = 1$ to n_{mc} **do**
propose ω' based on ω (for details, see Ref. 7)
sample U from Unif$[0, 1]$
$\omega = \omega'$ if $U < A(\omega, \omega')$
$\hat{\omega} = \omega$ if $p(\omega|Y)/p(\hat{\omega}|Y) > 1$
$\hat{X} = \frac{n}{n+1}\hat{X} + \frac{1}{n+1}X(\omega)$
end for

V. Components of DMTIM

We now describe in detail the components of distributed multiple-target tracking and identity management (DMTIM) described in Section II. We use a *belief vector* to represent the identity of a target probabilistically. For multiple targets, we have a *belief matrix* $B(t)$ whose columns are belief vectors of the targets. Thus, entry $B_{ij}(t)$ represents the probability that target j can be identified as label (or name) i at time t.

A. Multiple-Target Tracking (Data Association)

The *Multiple-Target Tracking (Data Association)* block of DMTIM takes in sensor measurements and outputs a mixing matrix, state estimates, and local information. The computations of a mixing matrix, state estimates, and local information are described below.

1. Mixing Matrix

Suppose there are K targets and K identities, for example, K aircraft with idenitites {piper, cherokee, cessna, $\cdots$ }, in the surveillance region of the current sensor. Then, the problem of managing identities of multiple targets is to match each target to its identity over time. For this, we use the idea of the Identity-Mass-Flow.[15] The idea of the Identity-Mass-Flow is that an identity is treated as a unit mass assigned to a target. These masses cannot be destroyed or created, and flow from a target into another through the *mixing matrix*, $M(t)$ at time t. The mixing matrix is a $K \times K$ matrix whose element $M_{ij}(t)$ represents the probability that target i at time $t-1$ has become target j at time t. Thus, the mixing matrix is a doubly stochastic matrix; that is, its column sums and row sums are equal to 1.

The exact computation of mixing matrix is NP-hard. More generally, the exact computation of association probabilities in JPDA is NP-hard[10] since the related problem of finding the permanent of a 0-1 matrix is #P-complete.[16] Hence, for a large problem, *i.e.*, when the number of targets is large, we need to seek for an approximation algorithm. While the heuristic approaches do not guarantee asymptotic optimality and may fail in some situations, the single-scan version of MCMCDA can approximate the mixing matrix $M = [M_{ij}]$ (the time index is suppressed) in polynomial time.[9] Hence, MCMCDA can efficiently approximate a mixing matrix with guaranteed error bounds.

2. State Estimate

The online MCMCDA[7] is used for multiple-target tracking. As mentioned earlier, MCMCDA can track an unknown number of targets and can initiate and terminate tracks. Hence, MCMCDA is suitable for tracking targets within the surveillance region of a sensor as the number of targets changes over time. At each measurement sampling time step, measurements are combined with measurements from previous time steps and construct the measurement set Y. MCMCDA finds the partition $\hat{\omega}$ which approximates the MAP estimate of the multiple-target tracking problem and state estimates for all tracks in $\hat{\omega}$. For each track $\tau \in \omega$, we compare it with the tracks of previously identified targets. If τ does not share any measurements with the tracks of previously identified targets, we declare τ as a new target. Then the current sensor makes a query about the identity of τ to its neighboring sensors. If the identity of τ is known to the neighboring sensors, its identity is copied to the current sensor. Otherwise, a new identity is created for τ. The identity of a target is deleted when the track of the target is terminated. In Section V-B, we describe how the belief matrix is updated upon changes in the number of identities.

3. Local Information

Local information has the form of a belief vector. In Ref. 17, MHT is used to generate local information. However, due to high time and space complexities of MHT, the method does not scale to larger problems. MCMCDA described in Section IV allows an efficient way to compute local information from both latest and past measurements. Another benefit of MCMCDA is that local information can be computed simultaneously while the number of targets and tracks of all targets are estimated. For identity k, let N_{jk} be the number of times the j-th latest observation is associated with the initial observation identified by k after the initial n_{bi} samples while running Algorithm 1, where n_{bi} is the number of initial burn-in samples. When Algorithm 1 terminates, we compute $\gamma^k = (\gamma^k_1, \ldots, \gamma^k_{n(t)})^T$ for identity k, where $\gamma^k_j = N_{jk}/(n_{\text{mc}} - n_{\text{bi}})$. Then we form local information from γ^k by resizing the vector according to the latest observations assigned in state estimates and normalizing the resized vector.

B. Identity Management

The *Identity Management* block consists of *Belief Matrix Update* and *Local Information Incorporation*. The mixing matrix and local information from *Multiple-Target Tracking (Data Association)* are used to update the belief matrix.

1. Belief Matrix Update

The *Belief Matrix Update* block maintains identity information stored in a $K \times K$ belief matrix $B(t)$ over time. The evolution of this belief matrix is governed by the equation:[15]

$$B(t) = B(t-1)M(t) \tag{6}$$

We can show that (6) keeps row and column sums of the belief matrix constant when the numbers of targets and identities are the same. However, this is not the case for distributed identity management since the number of the targets within the surveillance region of individual sensors may change over time. There are two possible cases: a

target leaves or enters the surveillance region of a sensor. When a target leaves, we delete the corresponding column in the belief matrix managed by the sensor. When a target enters the surveillance region of a sensor, there are two possible cases: (i) the target comes from the surveillance region of another sensor, which may be queried, or (ii) the target comes from the outside of the surveillance region of a sensor network. For these cases, we propose Algorithm 2, a scalable, event-driven, query-based belief matrix update algorithm.

Algorithm 2 Event-driven, query-based Belief Matrix Update

- For sensor i and target k

if target k leaves the surveillance region of sensor i. **then**
  delete the corresponding column in the belief matrix.
end if
if a target enters the surveillance region of sensor i. **then**
  send a query about the identity of target k.
  if there is an answer "yes" and receive the belief vector of target k, **then**
    augment the belief matrix with the belief vector received.
  else
    augment the belief matrix with a belief vector with a new identity assigned to the target.
  end if
end if

For distributed identity management, a belief matrix managed by each sensor may not be a square matrix but might more likely be a skinny matrix which has more rows than columns. The belief matrix may not be a doubly stochastic matrix, but it should be a stochastic matrix with column sums equal to one. Its row sums remain constant because an identity mass cannot be destroyed or created. It also has the property that the sum of column sums is equal to the sum of row sums; that is, even though the number of targets in the surveillance region of each sensor changes, the identity mass is conserved in the surveillance region. Since the evolution of the belief matrix is governed by (6), these characteristics of the belief matrix are preserved over time.

2. Local Information Incorporation

When local information is available, we use local information to decrease the uncertainty of the belief matrix measured by entropy. The entropy (Shannon information) of an $L \times K$ belief matrix is defined as

$$H(B(t)) \triangleq -\sum_{i=1}^{L}\sum_{j=1}^{K} B_{ij}(t)\log B_{ij}(t). \tag{7}$$

Then, the problem is how to incorporate this information to the belief matrix. From the idea of the Identity-Mass-Flow and the characteristics of (6), we know that the belief matrix should have the following properties: its column sums are equal to one, its row sums remain constant, and the sum of row sums and the sum of column sums are equal. However, if we replace a column in the belief matrix with local information, it is not guaranteed that the new belief matrix has the above properties. For a nonnegative square matrix, the Sinkhorn algorithm[18] can be used to scale a matrix to achieve specified row and column sums;[19,20] that is, we scale a new belief matrix so that its row and column sums remain the same as those of the belief matrix before local information incorporation. However, since the belief matrix is, in general, a nonnegative rectangular matrix, such iteration may not converge.[19,20,21] But the question whether a given matrix is *almost* scalable can be decided in polynomial time[21] and we refer the readers to Ref. 21 for more detail. Thus, we can efficiently check whether the available local information can be incorporated. Thus, local information can be incorporated when it makes the new belief matrix almost scalable. Even though the new belief matrix is almost scalable, local information incorporated may not necessarily decrease the uncertainty (entropy) of the belief matrix. Therefore, local information is incorporated only when it reduces the uncertainty of the belief matrix. The local information incorporation algorithm is described in Algorithm 3.

C. Information Fusion

For DMTIM, information fusion is crucial to compute the global information of the system from information provided by local sensors. In this section, we consider the problem of combining state estimates and identity belief vectors of the same target from different sensors.

Algorithm 3 Local Information Incorporation

- **Given:** local information (belief vector) of a target and a belief matrix $B(t)$.
- Make a matrix $B'(t)$ by replacing the column corresponding to the target in $B(t)$ with the local information.
- Operator S represents the matrix scaling process in Ref. 21.

if $B'(t)$ is almost scalable **then**
  $B_{new}(t) := \mathsf{S}(B'(t))$
  if $H(B_{new}(t)) \leq H(B(t))$ **then**
    $B(t) := B_{new}(t)$
  else
    $B(t) := B(t)$
  end if
else
  $B(t) := B(t)$
end if

1. Identity Information Fusion

Identity information (belief vectors) fusion can be formulated as an optimization problem. Three different cost functions, Shannon information, Chernoff information, and the sum of Kullback-Leibler distances, representing different performance criteria, are proposed in Ref. 3. The Shannon information method finds the global estimate as the belief vector that has the least uncertainty measured by entropy (Shannon information). However, the identity fusion algorithms using Chernoff information and the sum of Kullback-Leibler distances between the local estimates and the global estimate, are to compute the global estimate as the average of the given local estimates either geometrically or arithmetically. Therefore, the Shannon information method would be useful when the performance and/or fidelity of sensors is high, while the other fusion algorithms would be useful when good *a prior* information about a system is not available. In this paper, the Shannon information method is used because we consider scenarios in which all sensors are cooperative.

Suppose we have two belief vectors, $b_i \in [0,1]^n$ ($\sum_{j=1}^{n} b_i(j) = 1$ for $i \in \{1,2\}$), provided by local sensors. The Shannon information method computes a fused belief vector as a *convex combination* of two belief vectors:

$$b_{new} = wb_1 + (1-w)b_2, \qquad \text{where} \quad \sum_{j=1}^{n} b_{new}(j) = 1 \tag{8}$$

with a weight, $w = \frac{H(b_1)^{-1}}{H(b_1)^{-1}+H(b_2)^{-1}}$, which is the normalized inverse of Shannon information of a belief vector. Therefore, this fusion algorithm puts a larger weight on the belief vector which has smaller Shannon information than the other. When $H(b_1) = H(b_2) = 0$, we set $w = \frac{1}{2}$. $w = 0$ if $H(b_2) = 0$ (no uncertainty in b_2) and $w = 1$ when $H(b_1) = 0$ (no uncertainty in b_1). In these cases, the fused belief vector computed by the proposed fusion algorithm is a belief vector which has no uncertainty.

2. State Estimate Fusion

Since each sensor maintains its own set of tracks, there can be multiple tracks from the same target maintained by different sensors. In order to resolve this inconsistency, we do track-level data association to combine tracks from different sensors as described in Ref. 8. Let ω_i be the set of tracks maintained by sensor i and NB_i be a set of neighboring sensors around i, including i itself. Let $Y' = \{\tau_k(t) : \tau_k \in \omega_j, 1 \leq t \leq T, 1 \leq k \leq |\omega_j|, j \in \text{NB}_i\}$ be a set of observations of all identified targets. We form a set of combined observations Y from Y' by combining observations made from overlapping survaillance regions and keeping the remaining observations. We then form a new set of tracks ω_{init} from $\{\tau \in \omega_j : j \in \text{NB}_i\}$ while making sure that constraints defined in Section III-B are satisfied. Then we run Algorithm 1 on the set of combined observations Y with the initial state ω_{init} to find locally consistent tracks.

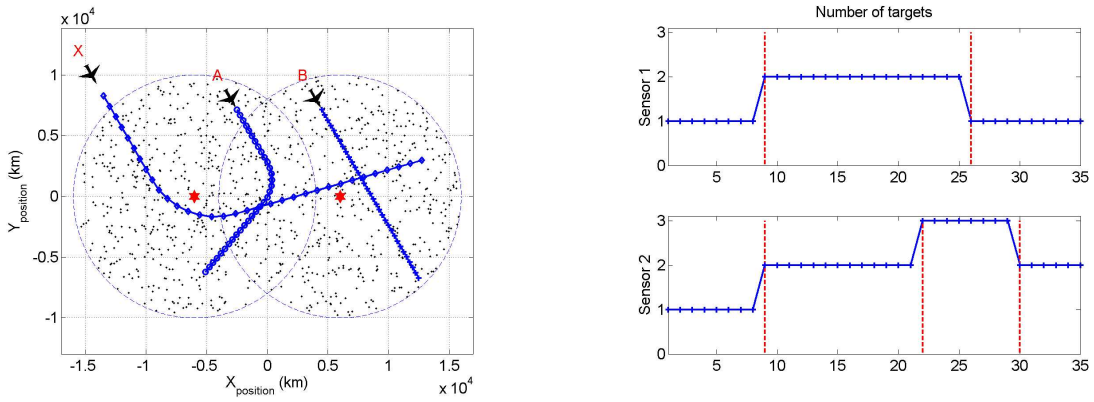

Figure 5. (left) Aircraft trajectories for three-aircraft, two-sensor scenario, superimposed with accumulated measurements (dots). The sensor positions are marked by $\star$. (right) Estimated number of targets by each sensor.

VI. Simulation Results

In this section, we present a simple scenario to illustrate the performance and capabilities of the DMTIM algorithm. There are two stationary sensors, *e.g.*, air traffic control radars, tracking multiple aircrafts through two-dimensional speace. The sensing range of each sensor is assumed to be circular with a radius of 10 km and a pair of sensors can communicate if they are within the communication radius of 20 km. The scenario with three aircrafts is shown in Figure 5 (left). The aircrafts labeled A and B are previously registered and aircraft labeled X is unknown to the identity management system. The sensor on the left is denoted by sensor 1 and the sensor on the right is denoted by sensor 2.

The *Multiple-Target Tracking (Data Association)* block in each sensor estimates the number of targets and estimates tracks of each detected targets as shown in Figure 5 (right) and Figure 6, respectively. In Figure 5 (right), the events, in which the number of targets changes, are indicated by dotted vertical lines. The belief vector for each target, *i.e.*, a column of the belief matrix, computed by the *Identity Management* block is shown in Figure 7. At time 1, sensor 1 knows about target 1 and its belief vector is $(b^1_{A,1}, b^1_{B,1})^T = (0.8, 0.2)^T$, where $b^i_{j,k}$ is the probability that target k of sensor i can be identified as label j, while sensor 2 knows about its target 1 and its belief vector is $(b^2_{A,1}, b^2_{B,1})^T = (0.2, 0.8)^T$. At time 9, sensor 1 detects a new target (target 2 of sensor 1) and assigns a new identity (X) since the target is unknown to its neighboring sensors. The updated belief vectors are shown in Figure 7 (left). At the same time, sensor 2 detects a new target (target 2 of sensor 2) and its identity and state estimate information is transferred from sensor 1, since its track is recognized as target 1 of sensor 1. The updated belief vectors are shown in Figure 7 (right). At time 22, sensor 2 detects a new target (target 3 of sensor 2) and its identity and state estimate information is transferred from sensor 1, since its track is recognized as target 2 of sensor 1. At time 26, target 2 of sensor 1 leaves the surveillance region of sensor 1 and information about target 2 is removed from sensor 1. At time 30, target 2 of sensor 2 leaves the surveillance region of sensor 2 and information about target 2 is removed from sensor 2.

For illustration purpose, Figure 7 (left) and (right) are showing the local belief vectors at each sensor before *Information Fusion*. At time 21, aircraft A and aircraft X cross one another and the uncertainty about identity is increased as shown in Figure 7 (left). For example, the belief that target 1 of sensor 1 can be identified with aircraft A is reduced from 0.8 to 0.45. However, *Information Fusion* can reduce this uncertainty by fusing the belief vector of target 1 of sensor 1 and the belief vector of target 2 of sensor 2. We use Shannon information for *Information Fusion* since we are considering a cooperative situation and it has been shown that Shannon information is superior against the other criteria in terms of cooperative efficiency.[3] The fused belief vector is shown in Figure 8 (left), in which the belief vectors from time 9 to 29 are shown. When target 3 of sensor 2 appears at time 23, *i.e.*, one time step after its detection, the identity uncertainty is reduced by *Information Fusion*. For example, the (fused) belief that target 1 of sensor 1 can be identified with aircraft A is increased from 0.45 to 0.64 as shown in the bottom plot of Figure 8 (left). Lastly, the tracks estimated by each sensor in a distributed manner are fused by *State Estimate Fusion* and the fused tracks are shown in Figure 8 (right).

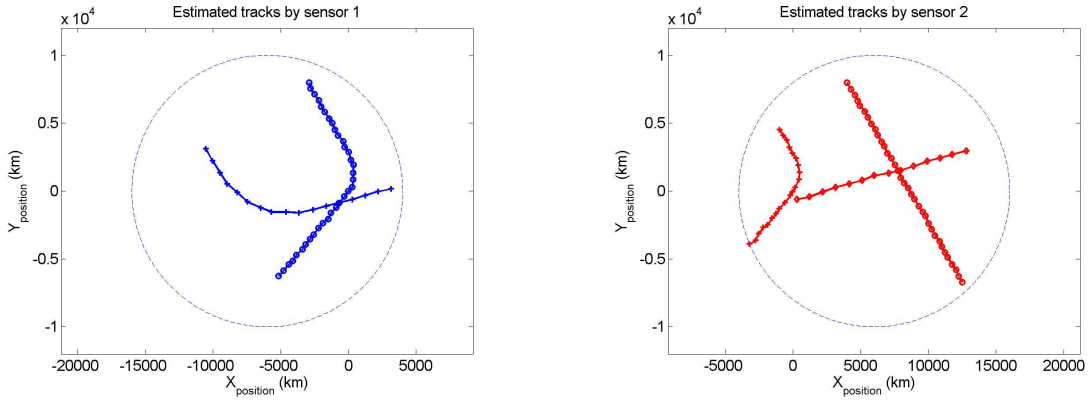

Figure 6. (Left) Estimated tracks by sensor 1. (Right) Estimated tracks by sensor 2 (a track of a target is shown after its detection).

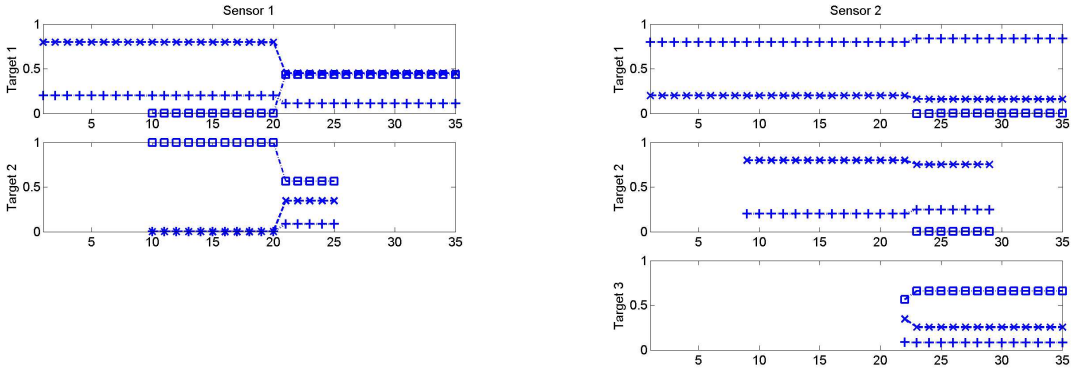

Figure 7. (left) Local belief vectors computed by sensor 1. (right) Local belief vectors computed by sensor 2. (The symbols $\times$, $+$, and $\square$ denote aircraft A, aircraft B, and aircraft X, respectively).

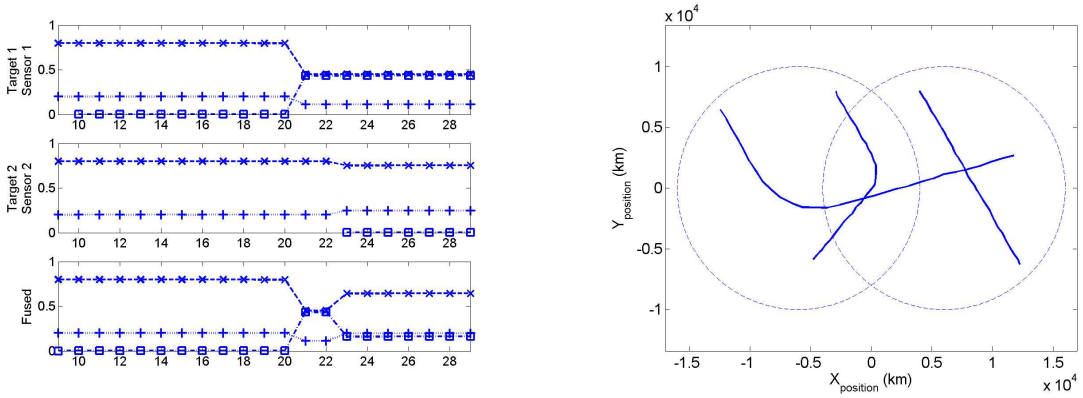

Figure 8. (left) Fused belief vector between target 1 of sensor 1 and target 2 of sensor 2 (The symbols $\times$, $+$, and $\square$ denote aircraft A, aircraft B, and aircraft X, respectively). (right) Fused tracks

VII. Conclusions

We have proposed a scalable distributed multiple-target tracking and identity management (DMTIM) algorithm that can track multiple targets and manage their identities efficiently in a distributed sensor network environment.

DMTIM consists of data association, multiple-target tracking, identity management and information fusion. The data association and multiple-target tracking problems are efficiently solved by Markov chain Monte Carlo data association (MCMCDA) which can track an unknown and time-varying number of targets. DMTIM efficiently incorporates local information about the identity of a target to reduce the uncertainty in the system and maintains local consistency among neighboring sensors via information fusion.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. EIA-0122599.

References

[1]Akyidliz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E., "A survey on sensor networks," *IEEE Communications Magazine*, Vol. 40, No. 8, August 2002, pp. 102–114.

[2]Xu, N., "A Survey of Sensor Network Applications," http://enl.usc.edu/ ningxu/papers.

[3]Hwang, I., Roy, K., Balakrishnan, H., and Tomlin, C., "A Distributed Multiple-Target Identity Management Algorithm in Sensor Networks," *Proceedings of the 43nd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, December 2004.

[4]Bar-Shalom, Y. and Fortmann, T., *Tracking and Data Association*, Academic Press, San Diego, CA, 1988.

[5]Reid, D., "An algorithm for tracking multiple targets," *IEEE Transaction on Automatic Control*, Vol. 24, No. 6, December 1979, pp. 843–854.

[6]Chong, C., Mori, S., and Chang, K., "Distributed Multitarget Multisensor Tracking," *Multitarget-Multisensor Tracking: Advanced Applications*, edited by Y. Bar-Shalom, Artech House: Norwood, MA, 1990, pp. 247–295.

[7]Oh, S., Russell, S., and Sastry, S., "Markov Chain Monte Carlo Data Association for General Multiple-Target Tracking Problems," *Proc. of the 43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, Dec. 2004.

[8]Oh, S., Schenato, L., and Sastry, S., "A Hierarchical Multiple-Target Tracking Algorithm for Sensor Networks," *Proc. of the International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

[9]Oh, S. and Sastry, S., "A Polynomial-Time Approximation Algorithm for Joint Probabilistic Data Association," *Proc. of the American Control Conference*, Portland, OR, June 2005.

[10]Collins, J. and Uhlmann, J., "Efficient gating in data association with multivariate distributed states," *IEEE Trans. Aerospace and Electronic Systems*, Vol. 28, No. 3, July 1992, pp. 909–916.

[11]Oh, S., Russell, S., and Sastry, S., "Markov Chain Monte Carlo Data Association for Multiple-Target Tracking," Tech. Rep. UCB//ERL M05/19, Univ. of California, Berkeley, 2005.

[12]Beichl, I. and Sullivan, F., "The Metropolis Algorithm," *Computing in Science and Engineering*, Vol. 2, No. 1, 2000, pp. 65–69.

[13]Jerrum, M. and Sinclair, A., "The Markov Chain Monte Carlo Method: An Approach to Approximate Counting and Integration," *Approximations for NP-hard Problems*, edited by D. Hochbaum, PWS Publishing, Boston, MA, 1996.

[14]Roberts, G., "Markov chain concepts related to sampling algorithms," *Markov Chain Monte Carlo in Practice*, edited by W. Gilks, S. Richardson, and D. Spiegelhalter, Interdisciplinary Statistics Series, Chapman and Hall, 1996.

[15]Shin, J., Guibas, L., and Zhao, F., "A Distributed Algorithm for Managing Multi-Target Identities in Wireless Ad-hoc Sensor Networks," *Information Processing in Sensor Networks*, edited by F. Zhao and L. Guibas, Lecture Notes in Computer Science 2654, Palo Alto, CA, April 2003, pp. 223–238.

[16]Valiant, L., "The complexity of computing the permanent," *Theoretical Computer Science*, Vol. 8, 1979, pp. 189–201.

[17]Hwang, I., Balakrishnan, H., Roy, K., and Tomlin, C., "Multiple-Target Tracking and Identity Management Algorithm In clutter, with Application to Aircraft Tracking," *Proceedings of the AACC American Control Conference*, Boston, MA, June 2004.

[18]Sinkhorn, R., "Diagonal equivalence to matrices with prescribed row and column sums," *American Mathematical Monthly*, Vol. 74, 1967, pp. 402–405.

[19]Rothblum, U. and Schneider, H., "Scaling of Matrices which Have Prescribed Row Sums and Column Sums via Optimization," *Linear Algebra and Its Applications*, Vol. 114/115, 1989, pp. 737–764.

[20]Linial, N., Samorodnitsky, A., and Wigderson, A., "A deterministic strongly polynomial algorithm for matrix scaling and approximate permanents," *Combinatorica*, Vol. 20, 2000, pp. 545–568.

[21]Balakrishnan, H., Hwang, I., and Tomlin, C., "Polynomial Approximation Algorithms for Belief Matrix Maintenance in Identity Management," *Proceedings of the 43nd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, December 2004.