This paper describes an extension to an approach patented by QinetiQ [1]. Implementing this approach without a licence will infringe this patent. A free evaluation licence for a MATLAB implementation of the patent can be obtained by emailing (ehm@signal.qinetiq.com).

Abstract - An assignment problem is considered with the constraint that the same hypothesis cannot be applied to more than one object. We desire efficiency without approximation. Multiple target tracking methods such as the Joint Probabilistic Association Filter (JPDAF) motivate us. Methods of solving this assignment problem involving enumerating all possible joint assignments are infeasible except for small problems. A recent approach circumvents this combinatorial explosion by representing the structure of the target hypotheses in a ‘net’ which exploits redundancy in an ordered list of objects used to describe the problem. Here, we generalize this approach to process the objects in a tree structure; this exploits conditional independence between subsets of the objects. This gives a substantial computational saving and allows us to consider scenarios which were previously impractical. In particular, we show the feasibility of using an exact JPDAF implementation to track 400 targets.

Keywords: Multiple target tracking, mutual exclusion, data association, estimation, assignment.

1 Introduction

We consider the problem of assigning a hypothesis to each of a set of objects, where we wish to enforce the constraint that the same hypothesis cannot be assigned to more than one object. For example, in multiple target tracking it is usually assumed that a measurement cannot be generated by more than one target. When computing the assignment probability of each measurement to each target, enforcing the constraint that each measurement can be assigned to at most one target is commonly known as Mutual Exclusion [2, 3] in the particle filtering literature, as well as the Hidden Markov Model literature [4]. A system in the Kalman filtering literature that imposes this constraint is the Joint Probabilistic Data Association Filter [5].

Due to computational limitations, the mutual exclusion constraint is often not fully enforced. Instead, either a single most likely joint assignment is found, as in Global Nearest Neighbour (GNN) [6], the assignments treated as independent, as in the Probabilistic Data Association Filter (PDAF) [6], or approximations employed [7, 8]. GNN can have difficulties recovering if an incorrect assignment is made, whereas PDAF tends to result in closely spaced targets sharing the same measurements and can therefore cause track coalescence. Using JPDAF prevents track coalescence provided that a sufficient number of mixture components for each target’s distribution is maintained [9]. It is therefore desirable to tackle the scenarios for which enforcing the mutual exclusion constraint is feasible.

Another application of an assignment algorithm which enforces this constraint is in using of contextual information to improve target identification performance. This motivates the authors’ interest and involves assigning targets to groups with the constraint that a target can belong to at most one group.

The precise statement of the problem of interest is given in Section 2. Previously, solving this problem was commonly performed by summing over all joint assignments which do not violate the constraint. However, the number of possible joint assignments grows exponentially with the number of objects, making this impractical except when only a small number of objects is considered.

In [10], the authors present a fast mutual exclusion algorithm known as Efficient Hypothesis Management, or EHM. EHM provides a solution which is practical for scenarios involving a large number of objects. This approach works by noting that we only need to remember at each stage which of the hypotheses have already been used. Furthermore, if a hypothesis cannot be assigned to any of the objects yet to be processed then it is not necessary to remember that it has been used.

This means that it is often the case that in the as-
assignment tree several nodes contain the same relevant information. By replacing the assignment tree by a
graph or ‘net’ which represents these nodes as a single
node, we can dramatically reduce the exponential
growth in complexity.

In this paper, we improve on the EHM algorithm by
making the following observation. Suppose that after
processing a subset A of the objects, we can partition
the remaining objects into two or more sets $T_1, \ldots, T_n$
such that none of the sets $\{T_i\}_{i=1}^n$ have any hypotheses
in common with any of the others. Then the assignment
probabilities of each object set $T_i$ depend on the assignment
probabilities of the others only through the
assignment probabilities of $A$. We therefore process
each set $T_i$ as a separate branch of a tree. We call this
enhanced algorithm EHM 2.

The paper is organized as follows. In Section 2, we
give an abstract mathematical statement of the problem.
In Section 3, we motivate this problem by showing
how to reduce the multiple target tracking problem to
it, where the only approximation made is to consider
the joint probability distribution of the target states as
a product of the marginal distributions. In Section 4,
we describe EHM 2. In Sections 5 and 6, we give some
results comparing EHM 2 to EHM for scenarios involving
large numbers of targets.

## 2 Problem statement

We have a sequence of independent discrete random
variables $a_1, \ldots, a_T$. For each $t$, the variable $a_t$
can take values in the set $Z_t \subseteq \{1, \ldots, M\}$, where $M$
is the total number of hypotheses. The probability
distributions $P(a_t)$ are known. We wish to calculate the conditional probabilities
$$P(a_t|A) \quad \text{for } t = 1, \ldots, T$$
where $A$ is the event that all the $a_t$ are distinct.

### 2.1 Null hypotheses

We often have an additional null hypothesis which can
be assigned to more than one object, and is often given
the index 0. For example, in multiple target tracking,
a target may not be detected and therefore the null hypothesis represents the possibility that the target is not
assigned to any of the measurements. By representing
the null hypothesis by an extra dummy hypothesis $M+1$ for object $t$, we see that this assignment problem is equivalent to the one above. Since the null hypothesis
for each object has a distinct index, we can assign
it more than once.

## 3 Multiple target tracking

Suppose we have $T$ targets with states represented by
$x_{1:T} = (x_1, \ldots, x_T)$. Each target independently
produces a measurement with probability $P_d$. A measure-
ment $y$ produced by a target at state $x$ has distribution
$p_T(y|x)$. In addition, $M_C$ clutter measurements
are produced, where $M_C$ has distribution $P_{M_C}$.

A clutter measurement $y$ has distribution $p_C(y)$. Let
$\tilde{y}_{1:M} = (\tilde{y}_1, \ldots, \tilde{y}_M)$ be this set of measurements in
order, with the target generated measurements first,
followed by the clutter measurements. Here, $M$ is the
total number of measurements.

The measurements we receive are the measurements
$\tilde{y}_{1:M}$ shuffled by some random (and unknown) permu-
tation $\gamma : \{1, \ldots, M\} \rightarrow \{1, \ldots, M\}$. Hence we receive
measurements $y_{1:M} = (y_1, \ldots, y_M)$, where $y_i = \tilde{y}_{\gamma(i)}$.

We assume that each permutation has the same prior probability $P(\gamma|M) = 1/M$.

We assume that the joint prior distribution on the
target states can be approximated by the product of
the marginal distributions:
$$p(x_{1:T}) \approx \prod_{t=1}^T p(x_t).$$

This is a standard assumption in multiple target tracking since we are generally interested in the target
marginal distributions and storing the joint distribution is computationally prohibitive. However, this
means that we throw away potentially useful information about the joint distributions.

Our aim is to calculate the marginal probability distributions $p(x_t|y_{1:M})$ for each target $t = 1, \ldots, T$, conditional on the received measurements. Let $a_t$ be
the measurement generated by target $t$ (or 0 if target $t$
generates no measurement). Then we can write the marginal distributions as the following mixture.
$$p(x_t|y_{1:M}) = p(x_t|a_t = 0|y_{1:M}) + \sum_{a_t \neq 0} p(x_t|a_t, y_a) P(a_t|y_{1:M}).$$

Since $p(x_t|a_t, y_a)$ can be computed by standard
methods (such as the Kalman filter), it remains to calculate
the marginal association probabilities $P(a_t|y_{1:M})$. We
do this by first calculating the joint association prob-
babilities $P(a_{1:T}|y_{1:M})$. By Bayes’ Theorem,
$$P(a_{1:T}|y_{1:M}) = \frac{P(a_{1:T}, M|y_{1:M})}{P(y_{1:M})} \times \int p(y_{1:M}|a_{1:T}, x_{1:T}, M)p(x_{1:T}) dx_{1:T}. \quad (4)$$

For $t = 1, \ldots, T$, let $d_t$ be 1 if target $t$ is detected
and 0 otherwise, and let $d_{1:T} = (d_1, \ldots, d_T)$. Then
$M_T = \sum_{t=1}^T d_t$ is the number of target generated measures-
ments, and $M = M_T + M_C$.

$M$ can be deduced from $\gamma$ since it is the size of
the domain, and $M_C = M - M_T = M - \sum_{t=1}^T d_t$.

Let $c_{1:M_C} = (c_1, \ldots, c_{M_C})$ be the measurement indices corresponding to clutter, in the order that they appear in $\tilde{y}_{1:M}$. Then $a_t$ and $c_t$ can be written as
$$a_t = \begin{cases} 0 & \text{if } d_t = 0 \\ \gamma^{-1} \left( \sum_{j=1}^t d_j \right) & \text{if } d_t = 1 \end{cases}$$
$$c_t = \gamma^{-1} \left( t + \sum_{j=1}^t d_j \right). \quad (6)$$

Hence we can deduce the measurement to target associ-
ations from $(d_{1:T}, \gamma)$. Since we can deduce $M_T$ and
where we take the convention that \(p_T(y_t|x) = 1\), so these terms vanish from the product. We assume a uniform clutter distribution, i.e. \(p_C(\cdot) = V^{-1}\) where \(V\) is the volume of the observation region \(\Gamma\). Then

\[
\int p(y_1, M|d_1, \gamma, x_1, T) p(x_1, T) \, dx_1, T
\]

\[
= \int p(y_1, M|d_1, \gamma, x_1, T) p(x_1, T) \, dx_1, T
\]

\[
= \frac{1}{V^{MC}} \prod_{t=1}^{T} \left[ \int p_T(y_t, x_t) p(x_t) \, dx_t \right]
\]

\[
= \frac{1}{V^{MC}} \prod_{t=1}^{T} p(t; y_t)
\]  

(8)

where \(p_t(\cdot)\) denotes the distribution of a measurement originating from target \(t\). Also,

\[
P(d_1, T, \gamma) = P(\gamma|M) P(M|d_1, T) P(d_1, T)
\]

\[
= \frac{1}{M!} P^{MC}(M_C) \prod_{t=1}^{T} P_d^t (1 - P_d)^{1 - d_t}
\]  

(9)

We assume that the number of clutter points is Poisson distributed, i.e.

\[
P_{MC}(m) = \frac{\lambda^{MV} (MV)^m}{m!}
\]  

(10)

where \(\lambda\) is a parameter representing the mean number of clutter points in a unit volume. If a prior value of \(\lambda\) is not available, it is commonly estimated as the number of measurements in the gated region divided by the volume of the gated region. Hence

\[
P(d_1, T, \gamma) = \frac{e^{-\lambda V} \lambda^{MV} M_C}{M! MC!} \prod_{t=1}^{T} P_d^t (1 - P_d)^{1 - d_t}
\]  

(11)

Note that for each joint association hypothesis \(a_{1:T}\), there are \(MC!\) permutations \(\gamma\) which give rise to it (since permuting the clutter measurements has no effect on \(a_{1:T}\)). So

\[
P(a_{1:T}, M) = \frac{e^{-\lambda V} \lambda^{MV} M_C}{M!} \prod_{t=1}^{T} P_d^t (1 - P_d)^{1 - d_t}
\]  

(12)

Here, the number of clutter permutations \(MC!\) cancels with the \(MC!\) term from the Poisson distribution.

Substituting (8) and (12) into (4) gives

\[
P(a_{1:T}|y_{1:M}) = \frac{\lambda^{MC} e^{-\lambda V} T}{M! p(y_{1:M})} \prod_{t=1}^{T} P_d^t (1 - P_d)^{1 - d_t} p_t(y_{a_t})
\]

\[
= \frac{\lambda^{MC} e^{-\lambda V} T}{M! p(y_{1:M})} \prod_{t=1}^{T} P_d^t (1 - P_d)^{1 - d_t} p_t(y_{a_t})
\]

\[
= \frac{\lambda^{MC} e^{-\lambda V} T}{M! p(y_{1:M})} \prod_{t=1}^{T} P_d^t (1 - P_d)^{1 - d_t} p_t(y_{a_t})
\]

\[
= \alpha \prod_{t=1}^{T} \beta_{t,a_t}
\]  

(13)

where \(\alpha\) is a normalizing constant and

\[
\beta_{t,j} = \left\{ \begin{array}{ll}
\lambda (1 - P_d) & \text{if } j = 0 \\
P_d p_t(y_j) & \text{if } j \neq 0.
\end{array} \right.
\]  

(14)

By performing gating, we typically find that many assignments of measurements to targets are highly unlikely, and set \(\beta_{t,j}\) to be zero for these. For each target \(t\), the values of \(j\) such that \(\beta_{t,j} > 0\) corresponds to the set \(Z_t\) in Section 2. Gating thus creates redundancy in the problem which is critical to an efficient solution.

Given the formula (13), we can reduce the problem of finding the marginal distributions to that in Section 2 as follows. Let \(\tilde{a}_1, \ldots, \tilde{a}_T\) be independent random variables with the distributions

\[
P(\tilde{a}_t = j) = \frac{\beta_{t,j}}{\sum_{k=1}^{M} \beta_{t,k}}
\]  

(15)

for \(t = 1, \ldots, T\) and \(j = 0, \ldots, M\). Let \(\Lambda \subseteq \{0, 1, \ldots, M\}^N\) be the set of valid associations, i.e. those where the nonzero values are distinct. Then

\[
P(\tilde{a}_t = j \mid \tilde{a}_{1:T} \in \Lambda) \propto \sum_{v \in \Lambda, v_t = j} \left( \prod_{k=1}^{T} \beta_{k,v_k} \right) P(a_t = j).
\]  

(16)

Since the left and right sides of (16) are probability distributions and therefore both sum to 1 over \(j\),

\[
P(a_t = j) = P(\tilde{a}_t = j \mid \tilde{a} \in \Lambda).
\]  

(17)

Hence we can calculate the marginal distributions of \(a_{1:T}\) by considering the independent variables \(\tilde{a}_{1:T}\).

### 4 Faster mutual exclusion

We solve the problem in Section 2 in two parts. In Subsection 4.1 we describe how to calculate the conditional probabilities when the objects are processed in a tree structure. This differs from the original EHM algorithm [10] where objects are processed sequentially. The algorithm is essentially smoothing on a tree. The exposition is similar to [11], which describes smoothing on chains, generalized to trees. Smoothing on trees is also covered in [4]. Pearl's algorithm [12] also covers
probabilistic inference on trees. In Subsection 4.2 we reduce the computational complexity of the algorithm by grouping together nodes which contain the same information about future possible assignments.

4.1 Calculating the conditional probabilities using trees

By following the construction in Subsection 4.3, we obtain a disjoint set of rooted trees on the nodes $1, \ldots, T$ with the following property. For each node $t$, let $M_t$ be the set of possible assignments for $t$ and its descendants, i.e.

$$M_t = Z_t \cup \left( \bigcup_{s \in \text{de}(t)} Z_s \right)$$

(18)

where $\text{de}(t)$ denotes the set of descendants of $t$. Then for all $s, t \in \{1, \ldots, T\}$ such that $s \neq t$, $s \notin \text{de}(t)$ and $t \notin \text{de}(s)$ we have

$$M_s \cap M_t = \emptyset.$$  

(19)

Hence separate branches have mutually exclusive possible assignments. Separate trees correspond to sets of objects which have no hypotheses in common and can therefore be processed separately, so we assume without loss of generality that we have a single tree.

It is worth noting that the original EHM algorithm is the special case of EHM 2 where we have a single non-branching tree.

Let $an(t)$ denote the set of ancestors of $t$, $\overline{\text{de}}(t) = \text{de}(t) \cup \{t\}$ and $\overline{\text{par}}(t) = \text{par}(t) \cup \{t\}$. We will also use $\text{nde}(t)$ to denote all nodes other than $t$ and $\text{de}(t)$, $\text{pa}(t)$ to denote the parent of $t$, $\text{ch}(t)$ to denote the set of children of $t$ and $\text{sib}(t) = \text{ch}(\text{pa}(t)) \setminus \{t\}$ to denote the set of siblings of $t$. We denote the root node by $rt$.

Let $A_t$ denote the event that $a_t$ does not conflict with $a_i$ for any ancestor $i$ of $t$, i.e.

$$A_t = \{a_t \notin \{a_i : i \in an(t)\}\}.$$ 

(20)

Given a subset $S \subseteq \{1, \ldots, T\}$, we also define the event $A_S = \bigcap_{t \in S} A_t$, taking the convention that $A_\emptyset = \Omega$, the certain event. It is clear from (19) that $a_S$ can only equal $a_t$ if $s \in an(t)$, $t \in an(s)$ or $s = t$, so $A = A_{\{1, \ldots, T\}}$. We also define $a_S$ to be the vector $(a_t : t \in S)$. Marginalizing the joint density $(a_t, an(t))$ and decomposing $\{1, \ldots, T\} = \{t\} \cup \text{de}(t) \cup \text{nde}(t)$ gives

$$P(a_t, A) = \sum_{an(t)} P(a_t, an(t), A_t, A_{\text{de}(t)}, A_{\text{nde}(t)}).$$

(21)

Let $V(a_t)$ be the set of $an(t)$ such that the elements of $an(t)$ are distinct and none of them are equal to $a_t$. If $t = rt$, and therefore $an(t) = \emptyset$, let $V(a_t) = \emptyset$. It is clear that $P(a_t, an(t), A) = 0$ for $an(t) \notin V(a_t)$ so we need not sum over these terms. For $an(t) \in V(a_t)$,

$$P(a_t, an(t), A_t, A_{\text{de}(t)}, A_{\text{nde}(t)})$$

$$= P(a_t, an(t), A_{\text{de}(t)}, A_{\text{nde}(t)})$$

$$= P(A_{\text{de}(t)})P(an(t)|A_{\text{par}(t)})P(a_t|an(t), A_{\text{nde}(t)})$$

$$\times P(an(t), A_{\text{nde}(t)}).$$

(22)

By conditional independence

$$P(a_t|an(t), A_{\text{nde}(t)}) = P(a_t)$$

(23)

$$P(A_{\text{de}(t)}|an(t), A_{\text{nde}(t)}) = P(A_{\text{de}(t)}|\overline{\text{an}(t)})$$

$$= \prod_{i \in \text{ch}(t)} P(A_{\overline{\text{par}(i)}}|an(i))$$

(24)

so

$$P(a_t|A) = \frac{P(a_t)}{P(A)} \sum_{an(t) \in V(a_t)} P(an(t), A_{\text{de}(t)}) \times \left( \prod_{i \in \text{ch}(t)} P(A_{\overline{\text{par}(i)}}|an(i)) \right)^{a_t}.$$ 

(25)

The conditional probabilities are calculated by means of forwards and backwards weights as in the original EHM algorithm [10] and the forward-backward algorithm used in Hidden Markov Models [13]. We define the backward and forward weights

$$w_B(t, an(t)) \triangleq P(A_{\overline{\text{par}}(t)}|an(t)),$$ 

(26)

$$w_F(t, an(t)) \triangleq P(a_t, an(t), A_{\text{nde}(t)}),$$

(27)

for distinct $(a_t, i \in an(t))$, which we shortly show how to calculate recursively. Then we can write (25) as

$$P(a_t|A) = \frac{P(a_t)}{P(A)} \sum_{an(t) \in V(a_t)} w_F(t, an(t)) \times \left( \prod_{i \in \text{ch}(t)} w_B(i, an(i)) \right).$$ 

(28)

To compute $P(a_t|A)$, it remains for us to derive recursive formulae for the forward and backward weights since $P(A)$ is just a normalising constant.

$$w_B(t, an(t)) = \sum_{an(t)} P(A_t, A_{\text{de}(t)}, a_t|an(t))$$

$$= \sum_{an(t)} P(A_t, A_{\text{de}(t)}|a_t, an(t))P(a_t|an(t))$$

$$= \sum_{an(t)} P(A_t|a_t, an(t))P(A_{\text{de}(t)}|a_t, an(t))P(a_t).$$

(29)

Letting $\overline{V}(an(t))$ denote the set of $a_t$ such that $a_t \notin \{a_i : i \in an(t)\}$, we have

$$w_B(t, an(t)) = \sum_{a_t \in \overline{V}(an(t))} P(A_{\text{de}(t)}|an(t))P(a_t).$$ 

(30)

Applying (24) to this gives

$$w_B(t, an(t)) = \sum_{a_t \in \overline{V}(an(t))} \left( \prod_{i \in \text{ch}(t)} w_B(i, an(i)) \right)P(a_t).$$

(31)

These weights can be calculated recursively up the tree.
By definition, \( w_F(t, \emptyset) = 1 \) for \( t = rt \). To obtain the forward weight recursion for \( t \neq rt \), note that \( \text{nde}(t) \) can be decomposed as
\[
\text{nde}(t) = \{pa(t)\} \cup \text{nde}(pa(t)) \cup \bigcup_{i \in \text{sub}(t)} \text{de}(i)
\]
and so
\[
w_F(t, a_{an(t)}) = P(a_{an(t)}, A_{pa(t)}, A_{nde(pa(t))}, \{A_{\text{de}(i)}\}_{i \in \text{sub}(t)}) \times
P(a_{an(t)}, A_{pa(t)}, A_{nde(pa(t))}).
\]
(33)

By conditional independence,
\[
P(\{A_{\text{de}(i)}\}_{i \in \text{sub}(t)} | a_{an(t)}) = \prod_{i \in \text{sub}(t)} w_B(i, a_{an(i)}).
\]
(34)

Also
\[
P(a_{an(t)}, A_{pa(t)}, A_{nde(pa(t))}) = P(A_{pa(t)}) P(a_{an(t)}, A_{nde(pa(t))}) = P(A_{pa(t)}) P(a_{an(t)}) w_F(pa(t), a_{an(pa(t))}).
\]
(35)

Since \( \{a_i | i \in an(t)\} \) are distinct, \( P(A_{pa(t)} | a_{an(t)}) = 1 \). Hence, by plugging (34) and (35) into (33), we obtain
\[
w_F(t, a_{an(t)}) = P(a_{pa(t)}) \left( \prod_{i \in \text{sub}(t)} w_B(i, a_{an(i)}) \right) \times
w_F(pa(t), a_{an(pa(t))}).
\]
(36)

4.2 Reducing the number of nodes

The technique in Subsection 4.1 gives a marginal improvement in efficiency over the standard JPDAF algorithm, but note that we still have to sum over the exponentially growing number of terms \( a_{an(t)} \) in (28). We can improve on this by exploiting further redundancy in the problem in a similar manner to the original EHM algorithm. For \( t \in \{1, \ldots, T\} \) and values for \( a_{an(t)} \), we define \( \chi(t, a_{an(t)}) \) by
\[
\chi(t, a_{an(t)}) = (t, \{a_i : i \in an(t)\} \cap M_t)
\]
(37)
i.e. the set of previous indices which could be assigned in the future, labelled with \( t \) as well. Note that \( \chi(t, a_{an(t)}) \) is a sufficient statistic of \( a_{an(t)} \) in the sense that it encapsulates all the information about \( a_{an(t)} \) to determine whether the events \( \{A_i\}_{i \in \text{de}(t)} \) are true or false. In particular, for \( i \in \text{de}(t) \), the assignment \( a_i \) conflicts with those of \( a_{an(t)} \) if and only if \( a_i \notin \chi(t, a_{an(t)}) \).\(^2\) Let
\[
\Theta_t = \{\chi(t, a_{an(t)}) : a_i \in Z_t \text{ for } i \in an(t) \text{ are distinct}\}
\]
(38)
be the set of such sufficient statistics for assignment \( t \) which can be reached by valid assignments to \( an(t) \). For \( \theta_t \in \Theta_t \), let
\[
\chi(\theta_t) = \{a_{an(t)} : a_{an(t)} \text{ are distinct and } \chi(t, a_{an(t)}) = \theta_t\}
\]
(39)
be the set of valid assignments of \( a_{an(t)} \) which give rise to \( \theta_t \). Note that \( \chi((t, \emptyset)) = \{\emptyset\} \) for \( t = rt \). Let
\[
\bar{W}(\theta_t) = \{a_i \in Z_t : a_i \notin \theta_t\}
\]
(40)
be the set of valid assignments \( a_t \) given \( \theta_t \). For \( a_t \in Z_t \), let
\[
W(a_t) = \{\theta_t \in \Theta_t : a_t \notin \theta_t\}
\]
(41)
be the set of \( \theta_t \) which allow the valid assignment \( a_t \).

We can think of \( \bigcup_{t=1}^T \Theta_t \) as a set of nodes in a ‘net’ on which we impose the following parent and child structure. Given \( \theta_t \in \Theta_t \), \( i \in ch(t) \) and a value of \( a_t \in \bar{W}(\theta_t) \), we define the child \( ch(\theta_t; i, a_t) \in \Theta_i \) by
\[
ch(\theta_t; i, a_t) \triangleq \chi(i, (a_{an(t)}, a_i))
\]
(42)
where \( a_{an(t)} \in \chi(\theta_t) \). This can easily be seen to be well defined, i.e. independent of the choice of \( a_{an(t)} \). Conversely, we define the set of parents \( pa(\theta_t; a_{pa(t)}) \) of \( \theta_t \) given \( a_{pa(t)} \in Z_{pa(t)} \) by
\[
\phi(\theta_t; a_{pa(t)}) = \{\theta \in \Theta_{pa(t)} : ch(\theta; t, a_{pa(t)}) = \theta_t\}.
\]
(43)

We now define \( w_F \) and \( w_B \) on the set of net nodes. For \( \theta_t \in \Theta_t \), let
\[
w_F(\theta_t) \triangleq w_B(t, a_{an(t)}) \text{ for } a_{an(t)} \in \chi(\theta_t),
\]
(44)
\[
w_F(\theta_t) \triangleq \sum_{a_{an(t)} \in \chi(\theta_t)} w_F(t, a_{an(t)}).
\]
(45)

It follows from (26) and the fact that \( \theta_t \) is a sufficient statistic for future assignments that \( w_B(\theta_t) \) is independent of the choice of \( a_{an(t)} \) used in the definition, and therefore well defined.

By applying these definitions to the recursion equations (28), (31) and (36), it can be seen (details are omitted to conserve space) that the marginal probabilities can be calculated using the following recursion.

\[
P(a_t | A) = \frac{1}{P(A)} \sum_{\theta_t \in W(\theta_t)} \left( \prod_{i \in ch(t)} w_B(ch(\theta_t; i, a_t)) \right)
\times w_F(\theta_t) P(a_t),
\]
(46)
\[
w_B(\theta_t) = \sum_{a_{pa(t)} \in \bar{W}(\theta_t)} \left( \prod_{i \in ch(t)} w_B(ch(\theta_t; i, a_i)) \right) P(\theta_t)
\text{ for } \theta_t \in \Theta_t,
\]
(47)
\[
w_F((t, \emptyset)) = \text{ 1 for } t = rt,
\]
(48)
\[
w_F(\theta_t) = \sum_{a_{pa(t)}} P(a_{pa(t)}) \sum_{\theta_t \in \Theta_{pa(t)}} \sum_{\theta \in \Theta_{pa(t)}} \left( \prod_{i \in \text{sub}(t)} w_B(ch(\theta; i, a_{pa(t)})) \right) p(\theta)
\text{ for } \theta_t \in \Theta_t, \ t \neq rt.
\]
(49)

4.3 Constructing the trees

We initialize the set of trees to be empty. Each object \( t \in \{1, \ldots, T\} \) to be assigned is considered in turn.
A node is created for that object and the children of that node are defined to be the roots $j$ such that $Z_t \cap M_t \neq \emptyset$. It is convenient to process the objects in reverse order so that the tree roots correspond to earlier objects in the list.

Note that if the objects decompose into clusters (as described in [14]), the above algorithm will generate several trees, one for each cluster. Since each cluster does not interact with the other clusters, each of these trees can be dealt with separately. This automatic clustering is a useful property of the algorithm.

4.3.1 Ordering of assignments

Clearly, the ordering of the objects is important so as to make the most effective use of conditional independence. The ordering approach suggested in [10] gives reasonable results in that for fairly large problems EHM 2 gives a substantial improvement over EHM. However, this ordering scheme was designed without conditional independencies in mind. It seems likely that a modified ordering algorithm which specifically looks for conditional independencies will allow us to improve the results still further. This should be an avenue of future research.

4.4 Example

To illustrate the algorithm, we consider the following small example representing a multiple target tracking scenario. The process of gating has been conducted and the null hypothesis is given an index of 0. Table 1 summarizes the valid associations for each target. Figure 2 shows the resulting net using EHM 2. Each target is a circular region centred on the target with diameter 0.1. We analyse the largest resulting cluster, which consists of 45 targets and 45 measurements. Figure 4(a) illustrates the gating arrangement, with the target gates represented by circles and measurements represented by crosses. Figure 4(b) illustrates the measurement interdependencies of the targets in this cluster. Here, the nodes represent the centres of the target gates and two nodes are connected by an edge if the corresponding targets share a common measurement hypothesis. The targets are numbered in the order they are processed.

We order the targets using the approach in [10]. The target tree resulting from this ordering is shown in Figure 5. Figure 6 shows the nets resulting from running EHM and EHM2. The EHM net contains 1365 nodes, whereas the EHM 2 net contains only 238 nodes. This results in a substantial computational saving. By comparison, the number of valid joint assignments in this problem is $5.379 \times 10^{20}$, so expanding the complete assignment tree is hopelessly impractical.

We note in particular that EHM 2 has successfully exploited the conditional independence of the target structure to remove much of the repetitive nature in the EHM net.

<table>
<thead>
<tr>
<th>Target number</th>
<th>Gated measurement indices</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0, 1</td>
</tr>
<tr>
<td>2</td>
<td>0, 1, 2, 3</td>
</tr>
<tr>
<td>3</td>
<td>0, 1, 2</td>
</tr>
<tr>
<td>4</td>
<td>0, 3, 4</td>
</tr>
</tbody>
</table>

Table 1: Associations for each target for the example in Section 4.4

Figure 1: The net of association hypotheses using EHM for the example presented in Section 4.4

Figure 2: The net of association hypotheses using EHM 2 for the example presented in Section 4.4
5.2 Scenario 2

To test the limits of EHM, we consider a larger example with 320 targets and 320 measurements. These are uniformly distributed over \((0, 2) \times (0, 2)\), so the measurement and target densities are the same as in the previous scenario. We keep the same gate size and again analyse the largest cluster.

The largest cluster has 68 targets and 68 measurements. EHM 2 takes just over a second on a 3 GHz PC and the net has 9346 nodes. By contrast, the EHM net has 710,874 nodes and it takes 5 min 25 sec, which is clearly impractical for most real time tracking applications. This scenario involves \(1.528 \times 10^{32}\) valid joint assignments. Hence we see that in this example EHM 2 can easily deal with a problem that pushes EHM to the limit of practicability. The figures that result are not particularly informative and are not shown for brevity.

6 Tracking Results

In this section, we test EHM 2 on a real tracking scenario. The target state space is taken to be \((x, \dot{x}, y, \dot{y})\), with the targets following a nearly constant velocity model \(x_{k+1} = Ax_k + \nu_k\). The process noise terms \(\nu_k\) are independent with covariance matrix \(Q\).

\[
A = \begin{pmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad Q = q \begin{pmatrix} \frac{\nu_1^2}{2} & \frac{\nu_2^2}{2} & 0 & 0 \\ \frac{\nu_3^2}{2} & T & 0 & 0 \\ 0 & 0 & \frac{\nu_4^2}{2} & \frac{\nu_5^2}{2} \\ 0 & 0 & \frac{\nu_6^2}{2} & T \end{pmatrix}
\]

where \(T = 1.0\) is the time between measurement epochs and \(q = 1.0\) is the power spectral density of the process noise. The initial position of each of the 400 targets is uniformly distributed in \([-2000, 2000]^2\), with each initial velocity component having distribution \(\mathcal{N}(0, 10^2)\). The targets are tracked for 20 measurement epochs.

Measurements are taken of the targets’ positions. Each position measurement has an independent error which has distribution \(\mathcal{N}(0, 20^2)\). We assume for this scenario that all targets are detected and there are no clutter measurements. Measurements are gated, with measurements within 3 standard deviations of the predicted measurement of a track being considered as a possible valid association for that track. New tracks are initiated on measurements which fall outside all targets’ gates.

We note again performing mutual exclusion by expanding the whole assignment tree, using EHM and using EHM 2 are mathematically equivalent and so will give the same tracking results (up to machine precision). Clearly, expanding the assignment tree is infeasible for a problem of this size. We compare the time required to complete the tracking for 5 randomly generated runs using EHM and EHM 2. We also compare the number of nodes in the largest net in each

Figure 3: The tree of valid associations for the example presented in Section 4.4

Figure 4: (a) Illustration of the (a) gating arrangement and (b) measurement interdependencies for Scenario 1. (See main text for details.)

Figure 5: Tree of how the targets are processed for Scenario 1. The numbers correspond to the numbering in Figure 4(b).
Figure 6: Nets of Scenario 1 using (a) EHM and (b) EHM 2. Note in particular the repetition in structure of (a).

case; this gives a measure of the amount of memory required.

The results are given in Table 2. The N/As in the table denote the runs where the mutual exclusion algorithm encounters problems due to memory limitations. For such a large number of targets, there may occasionally be insufficient redundancy in the problem for EHM 2 to exploit. In these cases, we may be forced to use approximation schemes. However, using EHM 2 considerably expands the range of problems we can solve analytically.

Table 2: Comparison of the time taken to track 400 targets with EHM and EHM 2, and the number of nodes in the largest net, for 5 runs using the scenario described in Section 6.

<table>
<thead>
<tr>
<th>Run number</th>
<th>Time (sec)</th>
<th>Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EHM</td>
<td>EHM 2</td>
</tr>
<tr>
<td>1</td>
<td>N/A</td>
<td>126s</td>
</tr>
<tr>
<td>2</td>
<td>318s</td>
<td>102s</td>
</tr>
<tr>
<td>3</td>
<td>324s</td>
<td>110s</td>
</tr>
<tr>
<td>4</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>5</td>
<td>640s</td>
<td>291s</td>
</tr>
</tbody>
</table>

7 Conclusions

A previously published efficient method of performing mutual exclusion for large assignment problems has been further improved by exploiting conditional independence between groups of objects. This improvement is shown to result in a substantial computational saving in large problems. Furthermore, an example problem is given which can easily be solved with EHM 2 but is impractical to solve with EHM, and we test this algorithm on a tracking scenario involving hundreds of targets.

8 Acknowledgements

The authors would like to acknowledge UK Ministry of Defence funding through the Detection and Identification of Difficult Targets Major Programme Area.

References