Dynamically Adaptable *m*-Best 2-D Assignment Algorithm and Multilevel Parallelization

ROBERT L. POPP, Member, IEEE
ALPHATECH, Inc.

KRISHNA R. PATTIPATI, Fellow, IEEE

YAAKOV BAR-SHALOM, Fellow, IEEE
University of Connecticut

In recent years, there has been considerable interest within the tracking community in an approach to data association based on the *m*-best two-dimensional (2-D) assignment algorithm. Much of the interest has been spurred by its ability to provide various efficient data association solutions, including joint probabilistic data association (JPDA) and multiple hypothesis tracking (MHT).

The focus of this work is to describe several recent improvements to the *m*-best 2-D assignment algorithm. One improvement is to utilize a nonintrusive 2-D assignment algorithm switching mechanism, based on a problem sparsity threshold. Dynamic switching between two different 2-D assignment algorithms, highly suited for sparse and dense problems, respectively, enables more efficient solutions to the numerous 2-D assignment problems generated in the *m*-best 2-D assignment framework. Another improvement is to utilize a multilevel parallelization enabling many independent and highly parallelizable tasks to be executed concurrently, including 1) solving the multiple 2-D assignment problems via a parallelization of the *m*-best partitioning task, and 2) calculating the numerous gating tests, state estimates, covariance calculations, and likelihood function evaluations (used as cost coefficients in the 2-D assignment problem) via a parallelization of the data association interface task. Using both simulated data and an air traffic surveillance (ATS) problem based on data from two Federal Aviation Administration (FAA) air traffic control radars, we demonstrate that efficient solutions to the data association problem are obtainable using our improvements in the *m*-best 2-D assignment algorithm.

Manuscript received May 16, 1997; revised July 10, 1998 and March 18, 1999.

IEEE Log No. T-AES/35/4/09313.

Research was conducted at the University of Connecticut and sponsored by Grants AFOSR F409620-93-1-0399 and ONR/BMDO N00014-91-J-1950.

Authors' addresses: R. L. Popp, ALPHATECH, Inc., Information Technology, 50 Mall Rd., Burlington, MA 01803, E-mail: (rpopp@alphatech.com); K. R. Pattipati and Y. Bar-Shalom, University of Connecticut, Electrical and Systems Engineering, 260 Glenbrook Rd., Storrs, CT 06269.

I. INTRODUCTION

A. Motivation

The problem of data association, namely, partitioning measurements across lists (e.g., sensor scans) into tracks and false alarms so that accurate estimates of true tracks can be recovered, has been extensively studied for many years. For a given multitarget tracking problem, the decision as to which algorithm to employ in solving the data association problem is typically motivated by several factors, including: 1) tracking accuracy requirements (e.g., rms error in position and velocity, track purity, track continuity, etc.), often determined by the characteristics of the multitarget scenario in the surveillance environment (e.g., target density), and 2) computational requirements, often determined by choices made with respect to 1) above and the computational resources available. Although in no way the only one, the sparsity (or density) of the surveillance environment is one very important consideration. For example, some characteristics that determine the sparsity/density of a surveillance environment include clutter, noise, and target density (e.g., urban environments typically contain denser environments in terms of ground targets than nonurban environments). Moreover, air traffic surveillance (ATS) applications typically deal with sparse air target environments, whereas military ground target tracking applications often times must deal with dense environments.

For sparse environments, multiple hypothesis tracking (MHT), the multidimensional S-D ($S \geq 3$)[1] assignment, and joint probabilistic data association (JPDA) are typically not necessary for data association because little is gained in accuracy at the expense of large computational overheads associated with these methods. On the other hand, a single scan processing approach to data association (i.e., 2-D assignment) is often practical and has been shown to be fairly efficient and accurate. For dense environments, however, the conventional wisdom is that the single scan processing approach to data association does not provide reliable performance [12, 22]. The finality of 2-D assignment decisions may, and often does, lead to loss of track and improperly partitioned measurements into tracks and false alarms. Alternatively, the more complex statistical estimation techniques, such as MHT and S-D assignment, have proven to be fairly accurate and reliable approaches to data association in dense environments. Postponing final decisions pertaining to difficult data association situations may be the prudent thing to do until more information, such as the next scan(s) of data, are received.

[1] For simplicity, unless otherwise stated, assume ($S \geq 3$) in all references to S-D.

In recent years, there has been considerable interest in an efficient and robust approach to data association based on an m-best 2-D assignment formulation [7–10, 19, 31]. With an appropriate modification of a cost matrix and by solving a series of modified copies of the initial problem, an algorithm, first due to Murty [18], can be used to find the m-best (ranked) solutions to not only the 2-D assignment data association problem, but, in general, to many other classical optimization problems as well. Determining m-best solutions (as opposed to only the best one) becomes especially important for assignment-based approaches to data association since the *hard* irrevocable decisions that such approaches make can be mitigated.

B. Related Research

The problem of data association has been extensively studied for many years in multitarget tracking problems, with a wealth of research identifying numerous well-known approaches proposed over the years, e.g., (in order of decreasing complexity) MHT [4, 15, 30], S-D assignment [11, 12, 20–24], JPDA [4], and numerous 2-D assignment algorithms [1, 2, 5, 13, 14, 25, 32]. The MHT algorithm evaluates the probabilities of feasible joint association hypotheses (within a time window and/or a pruned set to limit its otherwise exploding requirements).[2] The JPDA algorithm averages over all the hypotheses from the latest scan (a window of size 1), whereas the 2-D assignment picks the best (optimal) set of assignments from the latest scan.

S-D assignment is a discrete mathematical optimization formulation of the data association problem that systematically resembles an MHT within a window of length $(S-1)$. However, the main challenge to overcome in the S-D assignment problem is that of solving the ensuing NP-hard multidimensional assignment problem. In particular, an algorithm that determines the *optimal* solution is not only arduous, but also, impractical for even fairly small-sized problems; however, satisfactory tracking and computational performance can be realized utilizing existing S-D assignment algorithms that provide good *suboptimal* solutions, of quantifiable accuracy, and in pseudopolynomial time. Interested readers can find a wealth of material on the S-D assignment approaches to tracking in [11, 12, 20–24].

For 2-D assignment algorithms, although the best algorithms proposed over the years have polynomial time complexity, i.e., $O(N^3)$ in N tracks, they often have vastly different performances. For example, Pattipati, et al. [20] investigated the computational efficiency of three well-known assignment algorithms, namely, a generalized auction algorithm, a relaxation algorithm for network flows, and the signature method. Based on extensive experimental results, the generalized auction algorithm outperformed the other two methods by a factor of $4-5$. For sparse and dense problems, extensive research [5, 6, 9, 13, 14, 20, 32] shows: 1) the auction algorithm [5] is highly-suited and one of the most efficient algorithms for sparse problems, 2) the Jonker–Volgenant–Castanon (JVC) algorithm [13, 14] is highly suited and efficient for dense problems, and 3) the auction algorithm significantly outperforms the JVC algorithm for sparse problems, and vice versa for dense problems. Also, Jonker and Volgenant [14], Drummond, et al. [13], and Cox, et al. [9] each empirically showed the JVC algorithm to be uniformly faster than numerous 2-D assignment algorithms (not including the auction algorithm) for both sparse and dense problems. Tsaknakis [32], however, showed the auction algorithm significantly outperforming the JVC algorithm for an array of sparse problems (as we also show in this work).

Murty [18] was the first to recognize the utility of calculating not only the best (optimal) solution, but also the 2nd, 3rd, and, in general, the mth best solution to the 2-D assignment problem and various other classical optimization problems. Miller, et al. [16] proposed several optimizations to Murty's m-best 2-D assignment algorithm that substantially reduced its complexity from $O(mn^4)$ to $O(mn^3)$. For applications to multitarget tracking, Miller [16] and Cox [8, 9] were the first to recognize the utility of specifically utilizing Murty's m-best 2-D assignment algorithm for data association. However, the manner in which the m-best 2-D solutions are processed allows for several data association approaches to be approximated [7–10, 19, 31]. Recent research has suggested that efficient MHT and JPDA solutions can be obtained when using an m-best 2-D assignment formulation of the data association problem [8–10]. Contrary to the claims in [8–10], because it lacks the depth in lists (sensor scans) processed, i.e., 1 list processed at-a-time, an m-best 2-D is *at best* a special-case (1-scan) approximation of MHT with very limited utility and benefit. However, a (1-scan) JPDA can be approximated using an m-best 2-D assignment algorithm. Alternatively, an m-best S-D, as proposed in a companion paper [28], processes over S lists and is, in the authors' view, the correct and most efficient way to approximate an $(S-1)$-scan MHT in the assignment framework. Moreover, an m-best 2-D is subsumed by an m-best S-D assignment algorithm.

C. Scope and Organization of Paper

However promising it may be for data association, the m-best 2-D assignment algorithm still has

[2] We are assuming Reid's classical measurement-oriented MHT approach [30] and not a suboptimal, yet computationally practical, track-oriented approach as described in [15].

computational complexity issues to contend with. For example, in generating a JPDA or even an approximated MHT solution, at each scan, the m-best descendents for each of the existing hypotheses derived from the previous scan need to be determined, not simply the m-best 2-D assignment solutions to a single problem. Hence, even though it has polynomial time complexity across two scans, the complexity (in space and time) of the m-best 2-D assignment algorithm is exponential in the number of scans S processed over time, i.e., $O(m^S N^3)$, assuming N tracks and m hypotheses. To mitigate its computational complexity, several improvements to the m-best 2-D assignment algorithm can be pursued, including 1) suboptimal solutions, where a reasonable approximation to MHT can be obtained by retaining at most the m-best 2-D assignments (hypotheses) at each scan, pruning the remaining hypotheses which, one hopes, have negligible effect due to very small probabilities, 2) parallelization, where many independent and highly parallelizable tasks can be executed concurrently, and 3) efficient solution to the numerous 2-D assignment problems generated in the m-best 2-D assignment framework. The scope of the present effort pertains to 2 and 3.

We begin by providing in Section II a general overview of the various aspects of the data association algorithm developed in this work (termed m-best-2-D in the sequel). In particular, we discuss: 1) the 2-D assignment problem, 2) the m-best 2-D assignment algorithm, including improvements by Miller, et al. [16], and 3) a discussion of how the m-best 2-D assignment algorithm can provide solutions to various data association approaches. In Section III, we describe our dynamically adaptive 2-D assignment algorithm switching scheme that we incorporated into the m-best 2-D assignment approach, enabling a much more efficient solution to the data association problem, especially in dynamic environments. In Section IV, we present a multilevel parallelization of the m-best partitioning task and the data association interface task. Using both simulated data and an ATS problem based on data from two FAA air traffic control radars, in Section V, we provide results demonstrating that more efficient solutions to the data association problem are possible when our proposed improvements are incorporated into the m-best 2-D assignment framework. In Section VI, we provide concluding remarks and future directions for this and related research.

II. DATA ASSOCIATION

In general, data association is the decision process of *linking* measurements (from successive scans) of a common origin (i.e., a target or false alarm) such that each measurement is associated with at most one origin. In multitarget tracking, a probabilistic formulation is given to measurement-to-track data association. Consequently, the assignment of measurements to tracks and the update of tracks themselves may be done in a variety of ways, each having specific probabilistic interpretation. In this section, after providing an overview of the classical 2-D assignment problem and the m-best 2-D assignment algorithm, we discuss how various data association solutions can be obtained using this algorithm.

A. 2-D Assignment Problem Formulation

In m-best 2-D, we cast the 2-D assignment problem as follows. M measurements from the latest scan are to be assigned to the N most likely tracks from the previous scans using a global cost minimization function [3, 4] (typically based on a likelihood function). Note that $N \neq M$ is typically the case. Specifically, let $y = 0,\ldots,N$ denote a particular track from the set of existing tracks (including a *dummy* track $y = 0$), and $z = 0,\ldots,M$ denote a particular measurement from the latest set (scan) of measurements (including a *dummy* measurement $z = 0$). By using dummy tracks and measurements, it is possible to handle all data association contingencies, namely, false alarms, newly initiated tracks, track maintenance, track termination, and the case where the track and measurement list sizes are unequal (i.e., rectangular 2-D assignment problems). Define the binary *assignment* variable

$$\chi_{yz} = \begin{cases} 1 & \text{if measurement } z \text{ is assigned to track } y \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Note that $\chi_{y0} = 1$ implies that track y is unassociated and has missed a detection in the latest scan. Furthermore, $\chi_{0z} = 1$ implies that measurement z is unassociated, that is, not assigned to any of the N previously established tracks, but, instead, assigned to the dummy track (false alarm or new track initiation). Since measurement errors within a scan are independent of each other, maximizing the negative log-likelihood ratio, consisting of the joint probability density function (pdf) and probability [3, 4] of measurements given their origins and the corresponding detection events, over the set of feasible assignments can be cast into the following 2-D assignment problem:

$$\min \sum_{y=0}^{N} \sum_{z=0}^{M} c_{yz} \chi_{yz} \tag{2}$$

subject to

$$\sum_{z=0}^{M} \chi_{yz} = 1 \qquad y = 1,\ldots,N$$
$$\sum_{y=0}^{N} \chi_{yz} = 1 \qquad z = 1,\ldots,M \tag{3}$$

where the cost of assigning measurement z to track y is

$$c_{yz} = \begin{cases} 0 & \text{if} \quad y = 0 \text{ or } z = 0 \\ -\log\left(\dfrac{\Lambda(y,z)}{\Lambda(0,z)}\right) & \text{if} \quad -\log(\cdot) < 0 \\ \infty & \text{otherwise} \end{cases}. \tag{4}$$

The numerator in the above $-\log(\cdot)$ expression (see [25, 27] for the derivation), based on a likelihood function calculation $\Lambda(\cdot)$ from the state estimator, is the likelihood that measurement z at scan S originated from track y, and the denominator is the likelihood that measurement z corresponds to none of the existing tracks (i.e., a false alarm). The likelihood of false alarms, i.e., $\Lambda(0,z)$, is assumed uniformly probable over the sensor's field of view [4].

B. Review of *m*-Best 2-D Assignment Algorithm

In this section, because it serves as the foundation of *m*-best 2-D, we provide an overview of the *m*-best 2-D assignment algorithm. Suppose we express the 2-D assignment problem, say P, with N tracks and M measurements, as a weighted bipartite graph, represented by a list of triplets $\langle y,z,c_{yz}\rangle$, where each y represents a hypothesized track, each z represents a measurement from the current scan, and c_{yz}, the cost of assigning measurement z to track y, is based on the cost coefficient calculation as specified in (4). A feasible solution, or assignment, say a_i, is a set of triplets in which each y and z appear exactly once (except for dummy tracks and dummy measurements which may appear multiple times), i.e.,

$$a_i = \{\langle y,z,c_{yz}\rangle\} \cup \{\langle 0,d_z,0\rangle\} \cup \{\langle d_y,0,0\rangle\} \tag{5}$$

where

$$1 \le y, \quad d_y \le N, \qquad 1 \le z, \quad d_z \le M, \qquad y \ne d_y, \qquad z \ne d_z$$
$$\{y\} \cup \{d_y\} = \{1,\ldots,N\}, \qquad \{z\} \cup \{d_z\} = \{1,\ldots,M\}. \tag{6}$$

The feasible solution space A can then be expressed as

$$A = \bigcup \{a_i\}, \qquad \{a_i\} \ne \{a_j\}, \qquad i \ne j \tag{7}$$

where the size of the feasible solution space (i.e., the number of data association assignments) as derived in [31] is

$$|A| = \sum_{j=0}^{\min\{N,M\}} \frac{M!N!}{j!(M-j)!(N-j)!}. \tag{8}$$

The cost (or likelihood) of an assignment, denoted by $C(a_i)$, can be found by summing the individual costs (negative log-likelihoods) in the triplets, i.e.,

$$C(a_i) = \sum_{\langle y,z,c_{yz}\rangle \in a_i} c_{yz}. \tag{9}$$

Determining the single best (most likely, optimal) assignment $A^*_{(1)}$ to P then is a matter of determining the assignment that minimizes this sum, which can be solved using any number of well-known 2-D assignment algorithms (e.g., auction, JVC). The *m*-best 2-D assignments to P, i.e., $A^*_{(1)},\ldots,A^*_{(m)}$, are the m assignments $a_i \in A$ with the m least costs, i.e.,

$$A^*_{(1)} = \arg\min_{a_i \in A} \{C(a_i)\} \tag{10}$$

$$A^*_{(2)} = \arg \min_{a_i \in A \setminus A^*_{(1)}} \{C(a_i)\} \tag{11}$$

$$\vdots$$

$$A^*_{(m)} = \arg \min_{\substack{a_i \in A \setminus A^*_{(k)} \\ k=1,\ldots,m-1}} \{C(a_i)\}. \tag{12}$$

To determine the *m*-best 2-D assignments to P, we can use Murty's algorithm to rank, in polynomial time, the 2-D assignment problem solutions in order of increasing cost. Specifically, Murty's algorithm solves a series of 2-D assignment problems where the original problem P is *partitioned* into a number of subproblems, say P_n, $n = 1,\ldots,N$, having solution spaces A_n, $A_n \subset A$. The partitioning task in Murty's algorithm maintains the following two constraints:

$$\bigcup_{n=1}^{N} A_n = A - A^*_{(1)} \tag{13}$$

$$A_j \cap A_k = \emptyset, \qquad \text{for} \quad j,k = 1,\ldots,N, \quad j \ne k. \tag{14}$$

To create subproblem P_1, we first copy P to P_1, and then remove from P_1 the 1st triplet in the best (optimal) assignment $A^*_{(1)}$ of P, i.e., we remove $\langle y_1,z_1,c_{y_1z_1}\rangle \in A^*_{(1)}$ from P_1. Hence, subproblem P_1 is P less $\langle y_1,z_1,c_{y_1z_1}\rangle$, which implies that no solution to P_1 will ever contain this 1st triplet in its solution space, i.e.,

$$A_1 = \{a_i \in A : \langle y_1,z_1,c_{y_1z_1}\rangle \notin a_i\}. \tag{15}$$

In general, in creating subproblem P_n, $2 \le n \le N$, we first copy P to P_n, and then do the following two things. First, we remove from P_n the nth triplet in the best assignment $A^*_{(1)}$ of P, i.e., we remove $\langle y_n,z_n,c_{y_nz_n}\rangle \in A^*_{(1)}$ from P_n. This implies that no solution to P_n will ever contain this nth triplet in its solution space. Second, we force the 1st,$\ldots$,$(n-1)$st triplets, i.e., $\langle y_j,z_j,c_{y_jz_j}\rangle \in A^*_{(1)}$, $j = 1,\ldots,(n-1)$, in the best assignment of P to be in all solutions to P_n. We enforce this by removing in P_n all triplets incident

to y_j and z_j in P, except for the triplets $\langle y_j, z_j, c_{y_j z_j} \rangle$ themselves, which implies that every solution to P_n will contain the 1st,…,$(n-1)$st triplets in its solution space. Hence, for $j = 1,\ldots,(n-1)$, subproblem P_n is P less $\langle y_n, z_n, c_{y_n z_n} \rangle$, less all triplets $\langle y_j, k, c_{y_j k} \rangle \in P$, $0 \le k \le M$, and $\langle l, z_j, c_{l z_j} \rangle \in P$, $0 \le l \le N$, except for the triplets $\langle y_j, z_j, c_{y_j z_j} \rangle$. Thus,

$$\begin{aligned} A_n = \{ a_i \in A : \langle y_j, z_j, c_{y_j z_j} \rangle \in a_i,\ \langle y_n, z_n, c_{y_n z_n} \rangle, \\ \langle y_j, k, c_{y_j k} \rangle,\ \langle l, z_j, c_{l z_j} \rangle \notin a_i, \\ j = 1,\ldots,(n-1),\ 0 \le k \le M,\ 0 \le l \le N \}. \end{aligned} \tag{16}$$

Note that solution spaces A_n for subproblems P_n, $n = 1,\ldots,N$, are disjoint and their union will be exactly the solution space to P less its optimal assignment (i.e., $A - A^*_{(1)}$).

After partitioning P according to its optimal assignment $A^*_{(1)}$, we solve each subproblem P_n, $1 \le n \le N$, and pair it together with its optimal solution A^*_n, and place each pairing (P_n, A^*_n) on a queue, say Q. The 2nd best assignment to P, say $A^*_{(2)}$, is the optimal assignment A^*_n corresponding to subproblem P_n on the queue Q having minimum cost (or maximum likelihood), i.e.,

$$A^*_{(2)} = \arg \min_{A^*_n \in Q} \{ C(A^*_n) \} \tag{17}$$

which is equivalent to (11). To find the 3rd best assignment in P, we simply replace subproblem P_n corresponding to the 2nd best assignment $A^*_{(2)}$ by its partitioning in the queue Q. The optimal assignment $A^*_{(3)}$ corresponding to subproblem P_k, $1 \le k \le 2N$, in the queue Q that has minimum cost after partitioning P_n would then be the 3rd best assignment to P, and so on.

In the m-best 2-D method, since we perform one partitioning task for each of the m-best 2-D assignments determined, in the worst case, each partitioning creates N new problems. This creates up to $O(mN)$ 2-D assignment problems to be solved and insertions of (problem, solution) pairings on the queue Q. Solving each 2-D assignment problem has worst case complexity $O(N^3)$, and each insertion step has worst case complexity $O(mN)$. Hence, the worst case complexity of m-best 2-D is $O(mN(N^3 + mN)) = O(mN^4)$.

C. Improvements to m-Best 2-D

Miller, et al. [16] proposed three optimizations to improve Murty's method, which can be summarized as follows. 1) During the partitioning process, have the created subproblems inherit dual variables and partial solutions from the initial problem. 2) After the partitioning process, sort the subproblems by lower cost bounds before solving the assignment problems. 3) Partition in an order based on lower cost bounds. The first optimization, inheriting dual variables and partial solutions during partitioning, is a well-known optimization technique which reduces the worst case complexity of Murty's algorithm from $O(mN^4)$ to $O(mN^3)$. The other two optimizations do not improve the worst case complexity, but are practical because they improve the average case complexity substantially. For example, when a problem is partitioned, lower cost bounds on the cost of the best solutions to its subproblems can be determined. These bounds can then be used to: 1) avoid solving subproblems that are unlikely to produce the next best assignment, and 2) change the order in which the triplets are used for partitioning, thereby increasing the probability that smaller problems are more likely to contain the best solutions. These optimziations have been incorporated into m-best 2-D, and we refer interested readers to [16] for more details.

D. Various Data Association Approaches via m-Best 2-D

In this section, we provide a discussion (although in no way meant to be comprehensive) of how to use m-best 2-D to get solutions to various data association approaches. In general, in addition to requiring an appropriate strategy for track initiation and maintenance, the key differentiator of various data association approaches using the m-best 2-D assignment algorithm is in the proper formulation of assignment (hypothesis) costs.

In terms of track initiation, new tracks could be created based on the set of unassociated measurements resulting from solving the 2-D assignment problems. Initializing (forming) tracks based on a single measurement requires *a priori* velocity estimates/covariances or uncertainty regions.

For a description of how an m-best 2-D assignment algorithm can be cast in such a way as to provide solutions to an MHT approach, see Cox, et al. [8, 9] and/or Danchick, et al. [10]. While they describe the m-best 2-D assignment as a solution to MHT with a window of 1 scan, the m-best S-D assignment can provide the solution to an MHT with a window of $S-1$ scans. Similarly, the m-best 2-D assignment algorithm can also provide the JPDA solution by acting as a preprocessor to a JPDA filter (i.e., joint event generator/extractor) for any number of measurements and targets [4, 31, 34].

In general, for MHT and JPDA solutions, the problem at hand is the efficient (nonenumerative) generation/extraction of the feasible joint events (assignments) of measurements and tracks for each hypothesis at a particular scan and the derivation of their conditional probabilities. In the m-best 2-D assignment framework, for a particular hypothesis, the m feasible joint events correspond to the m-best

2-D assignments generated, i.e., $A_{(1)},\ldots,A_{(m)}$. For a particular track, say y, the m feasible joint events determine the measurements assigned to it via the following triplets: $\langle y, z_{i_j}, c_{yz_{i_j}} \rangle$, $0 \le i_j \le M$, $j = 1,\ldots,m$ (note that it is not necessarily the case that $z_{i_j} \neq z_{i_k}$, $0 \le i_k \le M$, $j \neq k$). The normalized cost of a feasible joint association is the probability of the corresponding feasible joint event. The marginal probability that a track is associated to the same measurement needed in the JPDA is determined by summing the joint probabilities of the feasible events in which this pair appears. The normalization is such that the sum of the joint probabilities over all possible feasible events is unity. However, the normalization term is constant and is of importance only when absolute values of probabilities matter. Since finding the global m-best 2-D assignments (hypotheses) requires only relative values, we can ignore the normalization constant, which implies an enumeration of all feasible joint events is not necessary. The enumeration could be limited to the m-best feasible joint events.

Getting 2-D assignment (single scan processing) solutions using the m-best 2-D assignment algorithm is rather trivial, i.e., simply set $m = 1$. In this case, (4) provides the association cost (based on the negative log-likelihood ratio), and the associated measurements in the 2-D assignment problem are used to extend existing tracks, i.e., the corresponding track state vector is updated with the new measurement via a standard update equation in a state estimator. Tracks that go unassociated (i.e., not assigned a measurement) in the 2-D assignment problem for a specified threshold of consecutive scans are terminated (dropped). Getting m-best S-D assignment solutions using the m-best 2-D assignment algorithm is not a straightforward extension of the 2-D case, and is the subject of a forthcoming paper.

How the set of tracks get combined/pruned based on the m-best 2-D assignment solutions can also be done in numerous ways. For example, one option is to never keep more than m track branches after solving the data association problem based on the m-best 2-D assignment solutions, i.e., keep only the m-best solutions at all times. Another option is to never keep more than some specified number of track branches after solving the data association problem, where the specified number can be based on the likelihood scores of the various track branches. Yet another option is to let the track branches grow (exponentially) by m for each scan of measurements processed, and to invoke pruning mechanisms after solving the data association problem to limit its otherwise exploding growth. Which approach is best is not the focus of the present work and is the subject of a forthcoming paper.

III. DYNAMICALLY ADAPTIVE m-BEST 2-D ASSIGNMENT ALGORITHM SWITCHING

In this and the next section, we describe a nonintrusive dynamic 2-D assignment algorithm switching mechanism, based on a problem sparsity threshold and multilevel parallelization. When incorporated into the m-best 2-D assignment algorithm, these improvements enable the data association problem to be solved much more efficiently than the "best" m-best 2-D assignment algorithm found in the literature (see Cox, et al. [9] and Miller [16]), especially in a dynamic environment.

Recall that the m-best 2-D assignment algorithm is independent of the 2-D assignment algorithm used for solving the numerous 2-D assignment problems generated during the partitioning task. Also recall that extensive research [5, 6, 13, 14, 20] shows the auction algorithm to be highly suited and one of the most efficient algorithms for sparse problems, and vice versa for the JVC algorithm and dense problems. Furthermore, Tsaknakis [32] has shown (as well as we do in this paper) that the auction algorithm can significantly outperform the JVC algorithm for sparse problems, and vice versa for dense problems. Hence, providing there is little overhead incurred, it would seem logical that switching between the two 2-D assignment algorithms for sparse and dense problems, respectively, whichever is applicable at the time, would enable more efficient solutions to the data association problem, especially in dynamic environments. Consequently, we developed an improvement to the m-best 2-D assignment algorithm by utilizing a nonintrusive 2-D assignment algorithm switching mechanism, based on a problem sparsity threshold parameter, i.e.,

$$\text{Sparsity} = \frac{|\mathcal{L}(S)|}{|\mathcal{C}(S)|} \tag{18}$$

where $|\mathcal{C}(S)| = NM$ denotes the maximum size of the set of candidate associations, and $|\mathcal{L}(S)|$ denotes those associations participating in the 2-D assignment problem.

Specifically, the task of interfacing with the 2-D assignment problem in m-best 2-D requires the creation of several data structures (e.g., vectors corresponding to the forward star graph representation of the 2-D assignment problem) to hold only the relevant information required by the 2-D assignment algorithms. The forward star graph representation is well known and commonly used for graph problems that are considered sparse and/or irregular (e.g., rectangular 2-D assignment problems). Note that the obvious *unassignments* (i.e., track having an empty gate and measurement not falling within a track gate) were preprocessed out and do not participate in the 2-D assignment problem. This was handled consistently in both the auction and JVC algorithms

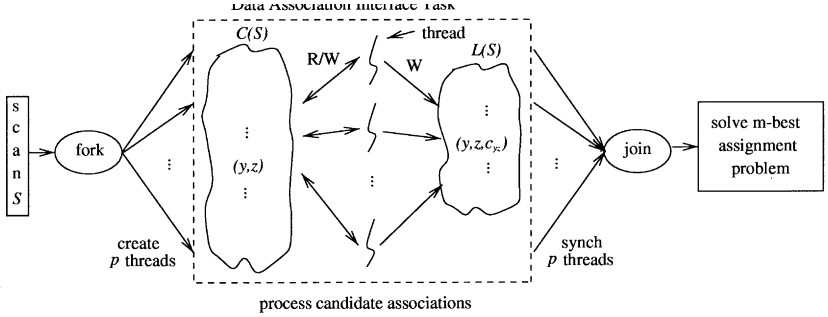


Fig. 1. Shared-memory parallelization of data association interface task in m-best 2-D.

in both stand-alone mode comparisons and when used within m-best 2-D. Similarly, the obvious *assignments* (i.e., track having a single measurement falling within its gate) were not preprocessed out and do participate in the 2-D assignment problem. This also was handled consistently in both the auction and JVC algorithms in both stand alone mode comparisons and when used within m-best 2-D.

Once the 2-D assignment problem data structures were created, the sparsity of the problem can be determined. Since the interface to the auction algorithm and the JVC algorithm is nearly identical, m-best 2-D can use the value of Sparsity to dynamically switch between these two algorithms with negligible overhead incurred; consequently, the switching mechanism is nonintrusive. Dynamic switching enables more efficient solutions to the 2-D assignment problem, and, when numerous problems require solution due to the partitioning task, this greater efficiency is compounded than when using a single static 2-D assignment algorithm.

IV. MULTILEVEL PARALLELIZATION

To mitigate the computational complexity issues, we also developed a multilevel parallelization in m-best 2-D targeted for shared-memory multiprocessor systems. A multilevel parallelization enables many independent and highly parallelizable tasks to be executed concurrently, including: 1) multiple 2-D assignment problems via a parallelization of the partitioning task, and 2) the numerous gating tests, state estimates, covariance calculations, and likelihood function evaluations (used as cost coefficients in the 2-D assignment problem) via a parallelization of the data association interface task.

A. Data Association Interface Task Parallelization

Even though it has been a historical and widely held belief that the most computationally intensive aspect of multitarget tracking has been the task of solving the data association problem, as we have shown in previous research [26, 27], contrary to conventional wisdom, the *interface* to the data association problem also comprises a significant fraction of the workload. Consequently, as illustrated in Fig. 1, in m-best 2-D, like in previous work, we developed a coarse-grained shared-memory parallelization of the interface to the data association problem. In particular, based on the supervisor/worker model, a supervisor thread[3] initially *forks* a specified number of worker threads, say p, to process the set of candidate associations, i.e., $\mathcal{C}(k)$. Once forked, the supervisor awaits processing of $\mathcal{C}(k)$ to be completed by the p worker threads via a *join* operation. Worker threads, asynchronously and in parallel, process a specified number of candidate associations per serialized critical section access across mutually exclusive track and measurement data. The processing of a candidate association primarily consists of computing the numerous *independent* gating tests (which consists of a coarse maximum velocity gating test and a fine Kalman filter elliptical gating test [26, 27]), state estimates, covariance calculations, and likelihood function evaluations used as assignment cost coefficients in the 2-D assignment problem. Since the processing cost corresponding to each candidate association is not uniform (depends on the results of gating), dynamic scheduling of candidate associations across threads is employed. In this way, because candidate associations are dynamically scheduled, maximum load balancing is achieved [26]. Upon processing of $\mathcal{C}(k)$ by the worker threads, the supervisor can then solve the data association problem via the auction or JVC algorithm.

B. m-Best 2-D Partitioning Task Parallelization

The shared-memory parallelization of the partitioning task is also coarse-grained and much like the supervisor/worker model previously described (see Fig. 2). Recall that after determining the optimal solution to an initial 2-D assignment problem,

[3]Note that a thread refers to a light weight process using the Solaris multithreading parallel processing interface [17, 29] (see Section VA).

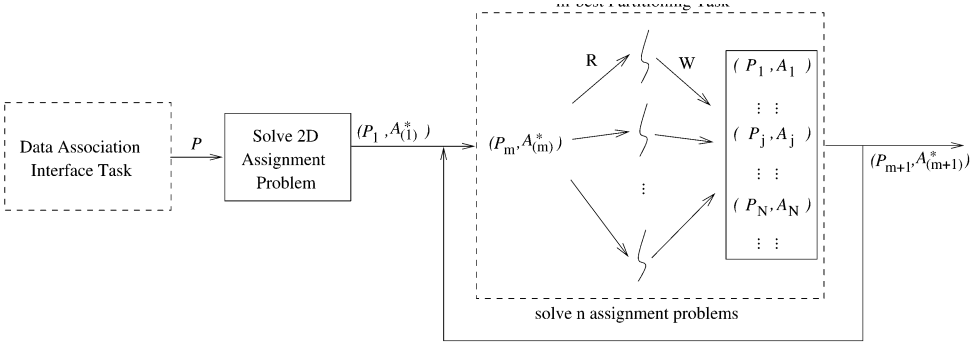


Fig. 2. Shared-memory parallelization of partitioning task in m-best 2-D.

denoted as $(P, A^*_{(1)})$, the partitioning task consists of creating N subproblems, say $P_1, \ldots, P_N$ (where N is the assumed track list size), and determining their optimal solutions, say $A^*_1, \ldots, A^*_N$, respectively. Since each of the N subproblems are independent of one another, they can be processed (i.e., created and solved) in parallel. The supervisor thread creates a specified number of worker threads, say $p \leq N$, to process the N subproblems $P_1, \ldots, P_N$, and awaits processing of the N subproblems to be completed before determining the next best assignment. Each worker thread, asynchronously and in parallel, creates its respective subproblem(s) and determines the (their) optimal solution(s) via the auction or JVC algorithm; collectively the worker threads generate (P_n, A^*_n), $1 \leq n \leq N$. Since the processing cost corresponding to each subproblem P_n is not uniform (depends on the number of triplets that are fixed and/or removed based on the partitioning algorithm), dynamic scheduling of subproblems across threads is employed. Again, in this way, because subproblems are dynamically scheduled, maximum load balancing is achieved. Upon processing of the N subproblems by the p worker threads, the supervisor can then determine the 2nd best assignment and its corresponding subproblem, say, $(P_2, A^*_{(2)})$. To find the 3rd best assignment, we simply repeat this loop, replacing $(P, A^*_{(1)})$ with $(P_2, A^*_{(2)})$, and so on.

V. RESULTS

In this section, after describing both the implementation environment and characteristics of the simulated data and the ATS problem used in this work, we provide the following results: 1) consistent with the works of other researchers, a comparison of the auction algorithm and the JVC algorithm shows the superior performance of the former over the latter for sparse problems, and vice versa for dense problems, and 2) plots show the improved performance of m-best 2-D (utilizing 2-D assignment algorithm switching and multilevel parallelization) over the "best" m-best 2-D assignment algorithm found in the literature (i.e., the m-best 2-D assignment algorithm due to Cox, et al. [9] with the optimizations developed by Miller [16] incorporated). To make the comparisons fair, we also incorporated Miller's optimizations in m-best 2-D. However, it should be noted that the benefits of Miller's optimizations are noticeably evident only when solving very large-scale problems, i.e., 2-D assignment problems several orders of magnitude greater than those solved in this work (track and measurement list sizes in the tens of thousands as opposed to in the hundreds).

A. Implementation Environment

The parallel computing environment used in this work consisted of a 4-processor SPARCstation 20—a MIMD shared-memory multiprocessor system. A simple model of this architecture is illustrated in Fig. 3 along with various hardware specifications. The software utilized in this work consisted of the Solaris 2.4.2 development environment, which includes the SunOS 5.4.2 UNIX operating system (OS) and the multithreaded system architecture, which we used as our parallel processing interface. Because of the numerous performance advantages offered by the *threads* model, we chose multithreading, as opposed to multiprocessing, as our avenue to exploit the multiple processor resources available in our multiprocessor system. Multithreading separates a UNIX process into some number of lightweight independent threads, each of which (concurrently) executes a sequence of the process's instructions. Unlike a process, a thread requires very little interaction with the OS. Consequently, threads can be processed quickly (i.e., created, maintained, destroyed, blocked, activated, etc.), thousands can be present at one time, and synchronization and context switching between them can be accomplished rapidly. As depicted in Fig. 3, threads are dispatched across the processor set indirectly via a two-level scheduling hierarchy. Threads implicitly communicate via shared memory. Consequently, synchronization mechanisms (e.g., mutual exclusion) must be supported to allow threads to cooperate in accessing shared data. More details concerning multithreading can be found in [17, 29].

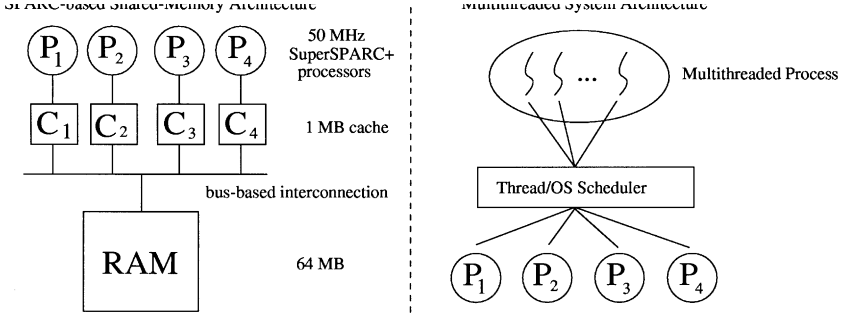


Fig. 3. Model of shared-memory architecture.

As a preface to the remainder of this section, since the 4-processor SPARCstation 20 used in this work is a time-shared multiprocessor system, the execution times of any tests performed depended, in part, on random events such as the system load and thread schedule order. Consequently, Monte Carlo simulations were performed, and all results presented represent the means of those simulations with standard errors less than 3%.

B. Simulated Data Description

In this section, we provide a brief overview of the simulated data used in this work. The purpose of using simulated data was to demonstrate the performance differences between the auction algorithm and the JVC algorithm on 2-D assignment problems of different sizes and sparsities. In Fig. 4, we provide a pseudocode description of the routine used to generate the various 2-D assignment problems. The basic idea in *Generate_2D_Assignment_Problem*( ) is to create a problem P in terms of its triplets $\langle y,z,c_{yz}\rangle$. Each track y is a uniform random number between $[1,N]$, each measurement z is a uniform random number between $[1,M]$, the cost (or likelihood) associated with y and z is c_{yz} and is a uniform integer random number[4] between $[LB,UB]$, where LB and UB correspond to a user specified lower and upper bound, respectively ($[1,1000]$ in this work), and the number of triplets in P is based on a user specified sparsity parameter, i.e., the sparsity of the problem P is: $Sparsity*N*M$ (see (18)).

```
Generate_2D_Assignment_Problem(N,M,Sparsity,LB,UB)
  Matrix m[N][M]
  2D_Assign_Prob P
  Num_Edges = Sparsity * N * M
  i = 1
  while (i ≤ Num_Edges)
      y = Uniform_Random_Number(1, N)
      z = Uniform_Random_Number(1, M)
      if (m[y][z] ≠ true)
          m[y][z] = true
          c_yz = −1.0 * Uniform_Random_Number(LB, UB)
          P.triplets[i] = ⟨y, z, c_yz⟩
          i = i + 1
  return P
```

Fig. 4. Pseudocode for 2-D assignment problem generator.

C. 2-D Assignment Algorithm Performance

The overall objective of this section is simply to provide motivation for our proposed improvement to the m-best 2-D assignment algorithm using the dynamic 2-D assignment algorithm switching mechanism in m-best 2-D. In doing so, we compare the auction algorithm[5] with the JVC algorithm, and demonstrate the superior performance of the former over the latter for sparse 2-D assignment problems, and vice versa for dense problems. As a consequence, more efficient solutions to the numerous 2-D assignment problems generated by the m-best partitioning task are possible. Our intent here is not to provide comprehensive results and analyses as it pertains to the performance characteristics of these two algorithms. On the contrary, our results are based on fairly limited testing. However, with respect to the validity of our claims concerning the performance characteristics of the auction and JVC algorithms, we rely heavily on the consistency of our results with those obtained in numerous other research efforts [5, 6, 13, 14, 20, 32].

The routine *Generate_2D_Assignment_Problem*( ) was used to create various 2-D assignment problems of different sizes and sparsities. As previously described, this routine will randomly generate (using a uniform distribution) the tracks, measurements, candidate measurement-to-track associations, and costs for those candidate associations. In Fig. 5, we provide plots comparing the auction algorithm and

[4]The assignment cost was limited to integer costs for all testing because this is a limitation of the auction algorithm. This is important to note since the JVC algorithm used in this work was designed for real costs and better performance may have been obtained in the JVC case if it had been suitably modified to take advantage of integer costs.

[5]We implemented the generalized auction algorithm as opposed to the forward-reverse auction algorithm in this work because it was found that the latter has worse performance than the former for sparse problems (consistent with other research efforts [32]), and worse performance than the JVC algorithm for dense problems.

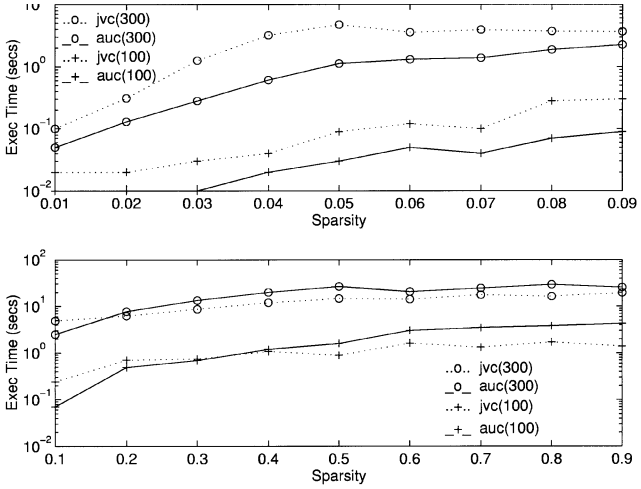


Fig. 5. Comparison of auction algorithm and JVC algorithm for $n \times n$ ($n = 100, 300$) assignment problems across a range of sparsities.

the JVC algorithm for problems of size 100×100 and 300×300, respectively. Furthermore, we divided the plots into two subplots based on the sparsity of the problem. The upper subplot compares the auction algorithm and the JVC algorithm for problems considered to be fairly sparse, i.e., sparsity ranging from $[.01, .09]$.[6] The lower subplot, on the other hand, compares the auction algorithm and the JVC algorithm for problems whose sparsity ranges from mildly sparse to fairly dense, i.e., sparsity ranging from $[0.1, 0.9]$.

In general, as can be seen in each of the plots, the auction algorithm has fairly consistent behavior for sparse problems in comparison to the JVC algorithm, and vice versa for dense problems. This is entirely consistent with the results of other researchers [9, 13, 14, 20, 32]. In particular, the auction algorithm outperforms the JVC algorithm for problems that would be considered fairly sparse (the upper subplot). At approximately 0.2 ± 0.1 sparsity, the auction algorithm and the JVC algorithm have comparable performance, whereas, for dense problems, the JVC algorithm noticeably outperforms the auction algorithm. As the density of the problem increases, so too does the likelihood that the level of contention amongst the tracks for particular measurements increases. The auction algorithm, benefiting from an inherent rapid convergence property when determining solutions to sparse problems, suffers from internal price wars as the level of contention increases, and, as a result, experiences degraded performance for dense problems. However, most practical ATS multitarget tracking problems have sparsity below 0.1 [4];

[6]Note that under conditions where sparsity is ≥ 0.02, there would be on average no less than 6 measurements per track gate, which would render tracking performance impractical and meaningless for a 2-D assignment single-scan processing approach.

however, note that this is not the case in general for multitarget tracking.

Recall that dummy tracks and dummy measurements were used to ensure feasibility for the 2-D assignment problem. Also, for both the auction and JVC algorithms, sparsification techniques were used to represent the 2-D assignment problem. As a result, $N \times M$ matrices were not used (where N is the number of tracks and M the number of measurements). Instead, for the auction algorithm, the various vectors needed to represent the sparse 2-D assignment problem were either of size $N + 1$, $M + 1$, or, in the case of the cost vector, in the range $[N, NM]$. For the JVC algorithm, a similar scheme was used as was described in [13], i.e., for each target, an additional dummy measurement was added. Many of the same vectors were processed as in the auction algorithm and were of the same size, e.g., $N + 1$. However, because N dummy measurements needed to be added (one per target) to the measurement list (originally of size M), the cost vector size was in the range $[N + M, (N + M)(N + M)]$. In terms of the performance comparision between the auction and JVC algorithms described later on, two cases were considered: 1) Given an N target and M measurement problem, compare auction and JVC directly (which implies that the two algorithms process over different size vectors), and 2) Compare an N_1 target and M_1 measurement auction algorithm problem with an N_2 target and M_2 measurement JVC algorithm problem, where $N_1 = M_1 = (N_2 + M_2)$. Both cases were tested with negligible differences evident. Given the fairly small size problems considered in this work (i.e., $100 = N, M = 300$), and since the JVC algorithm typically disregards (jumps over) most target's dummy measurements, the direct comparison of 1 was chosen and in the authors' view fair.

D. ATS Problem Description

In this section, we provide a simple overview of the ATS problem used in this work, and refer interested readers to [33] for a more thorough presentation. The surveillance system is at a location referred to as the fusion center. The sensors in this surveillance system, two L-band FAA air traffic control radars located at Remsen and Dansville, NY, respectively, asynchronously transmit scans of data, approximately every 10 s apart, to the fusion center. The full measurement database consists of 210 scans (98 from Remsen and 112 from Dansville), while a modified version (for which we report results for in this work) consists of 55 scans (26 from Remsen and 29 from Dansville). A particular scan received at the fusion center consists of some number of measurements (detection reports) consisting of a number of primary radar (skin) returns (i.e., slant range, azimuth angle) and secondary (beacon) returns

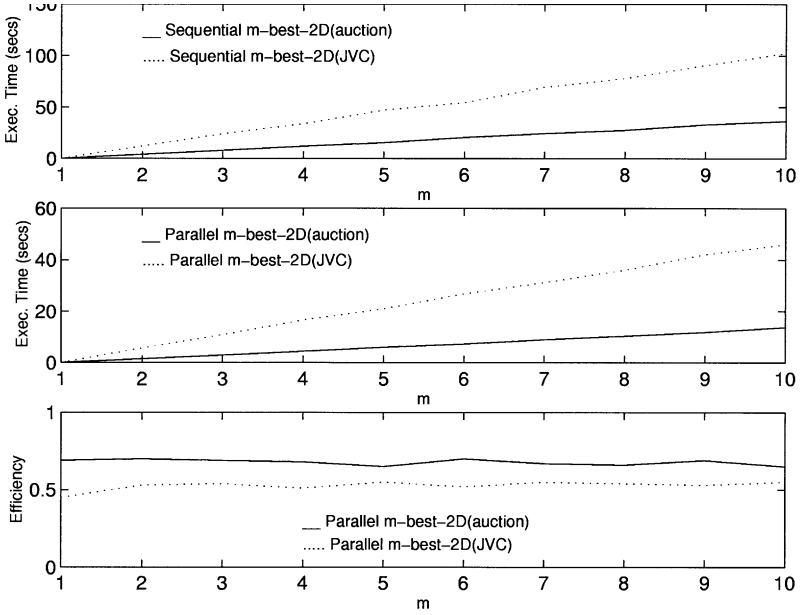


Fig. 6. Sequential and parallel performance of m-best 2-D on 4-processor SPARCstation 20.

(i.e., slant range, azimuth angle, altitude). Because state estimation is done in a three-dimensional (3-D) Cartesian coordinate system, conversion from a radar's polar frame of reference was carried out [4, 33].

E. m-Best 2-D Assignment Algorithm Performance

In this section, using both simulated data and the actual FAA data from the ATS problem as described in Section VD, we demonstrate that more efficient solutions to the data association problem are possible when incorporating our proposed improvements into the m-best 2-D assignment approach to data association.

1) *Performance Results: Multilevel Parallelization*: In providing parallel performance metrics, we use standard definitions for speedup and efficiency, i.e.,

$$\text{speedup} = \frac{\tau_1}{\tau_p}, \qquad \text{efficiency} = \frac{\text{speedup}}{p} \tag{19}$$

where τ_1 denotes the sequential execution time utilizing 1 processor, and τ_p denotes the parallel execution time utilizing p processors.

In Fig. 6, we provide various plots showing the sequential and parallel performance of m-best 2-D using actual FAA data from the ATS problem on the 4-processor SPARCstation 20. Note that because the ATS problem is sparse throughout, an m-best 2-D configured with a static auction algorithm to solve the 2-D assignment problem is equivalent to an m-best 2-D configured with the dynamic 2-D assignment algorithm switching mechanism as proposed in this work, i.e., no switching occurs.

Since all the 2-D assignment problems contained within the ATS problem are sparse, m-best 2-D configured with the auction algorithm outperformed m-best 2-D configured with the JVC algorithm in terms of execution time by a factor of (≈ 1.5) using 2 processors and a factor of (≈ 3) using 4 processors.[7] However, notice the apparent improvement in terms of parallel efficiency of m-best 2-D configured with the auction algorithm versus the JVC algorithm. The explanation for this is as follows. In determining a single 2-D assignment problem solution, the JVC algorithm requires more internal data structures (i.e., memory space) than does the auction algorithm. Consequently, the JVC algorithm suffers from a greater system-level overhead burden (i.e., the OS creating and managing the memory space) than does the auction algorithm. When m-best 2-D sequentially solves the numerous 2-D assignment problems generated as a result of the partitioning task, it can reuse (as opposed to recreate) the additional memory space. However, this is not the case when m-best 2-D solves in parallel the numerous 2-D assignment problems generated, since each of the subproblems created requires its own instance of this additional memory space. Consequently, the system-level overhead burden is compounded when solving the numerous 2-D assignment problems in parallel, and, as a result, worse parallel efficiency (and

[7]Note that the assignment costs in the 2-D assignment problems were initially real numbers based on a negative log likelihood function. To be converted to integer cost (which was necessary), the real cost was multiplied by a scale term and then quantized. The scale term used in this work was sufficiently large, i.e., 1000, and resulted in assignment costs in the range of $[-100000, 100000]$. The size of the scale term has impacts on execution time and the impact of different scale term multipliers was not pursued as part of this work.

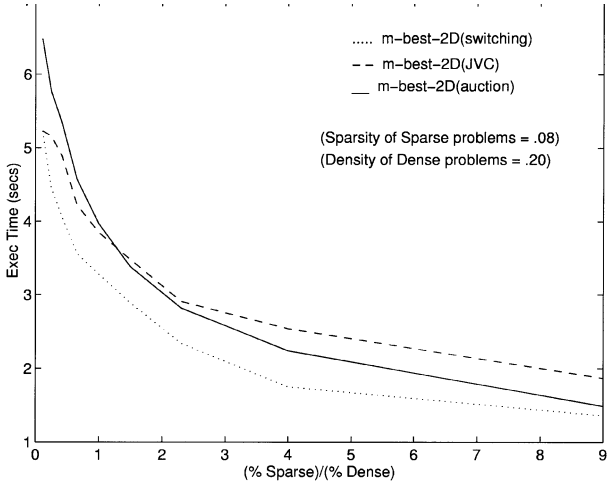


Fig. 7. Comparison of m-best 2-D statically configured with auction algorithm, JVC algorithm, and utilizing dynamic 2-D assignment algorithm switching mechanism in simulated ATS environment (with sparse and dense 2-D assignment problems that are not very sparse and dense, respectively). Horizontal axis represents percentage (fraction) of simulated ATS environment that is sparse relative to dense.

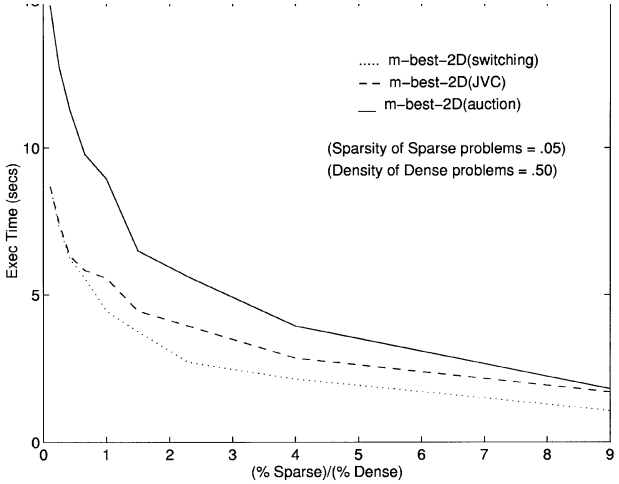


Fig. 8. Comparison of m-best 2-D statically configured with auction algorithm, JVC algorithm, and utilizing dynamic 2-D assignment algorithm switching mechanism in simulated ATS environment (with sparse and dense 2-D assignment problems that are sparse and dense, respectively). Horizontal axis represents percentage (fraction) of simulated ATS environment that is sparse relative to dense.

speedup) ensues in m-best 2-D configured with the JVC algorithm.

2) *Performance Results: Dynamic 2-D Assignment Algorithm Switching*: Next, we want to show the benefit, in terms of performance improvement, of using the dynamic 2-D assignment algorithm switching mechanism in m-best 2-D, as opposed to using m-best 2-D statically configured with a single 2-D assignment algorithm like auction or JVC. These performance improvements get compounded in an environment that is dynamic (i.e., sparse and dense) considering the numerous 2-D assignment problems generated due to the partitioning task.

In order to demonstrate m-best 2-D in dynamic environments, we used several of the 2-D assignment problems created using the routine *Generate_2D_Assignment_Problem*( ) to simulate various environments having different degrees of "dynamicness". In Figs. 7–9, we provide various plots showing the performance of m-best 2-D statically configured with the auction algorithm, the JVC algorithm, and utilizing the dynamic 2-D assignment algorithm switching mechanism in these simulated environments. The horizontal axis represents the ratio of the simulated environments that are sparse relative to dense. So, for example, at value 1 on the horizontal axis, half of the 2-D assignment problems in the environment are sparse, and the other half are dense, which, in this case, would represent a highly dynamic environment. As we approach value 0 (9) on the horizontal axis, the environment becomes more dense (sparse), and, consequently, less dynamic. Note that in this work, we used Sparsity = 0.10 as the problem sparsity threshold value to induce the 2-D assignment algorithm switching.

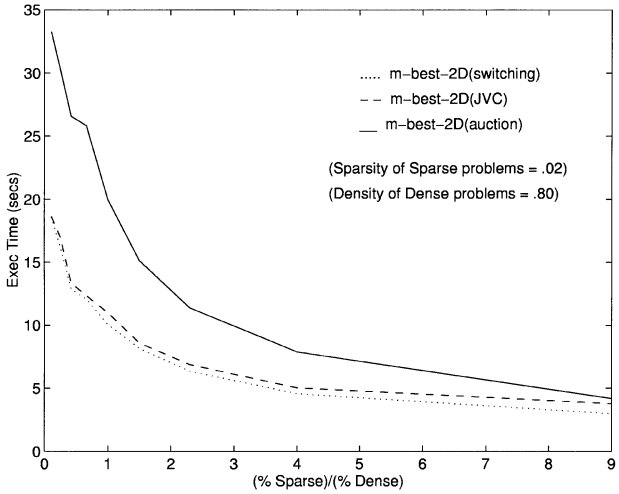


Fig. 9. Comparison of m-best 2-D statically configured with auction algorithm, JVC algorithm, and utilizing dynamic 2-D assignment algorithm switching mechanism in simulated ATS environment (with sparse and dense 2-D assignment problems that are very sparse and dense, respectively). Horizontal axis represents percentage (fraction) of simulated ATS environment that is sparse relative to dense.

Besides the fraction of the environments that is sparse and dense, another dimension that needed to be considered is the "degree" of sparsity and density of the sparse and dense problems in the ATS environment, respectively. So, for example, in Fig. 7, the sparsity of the sparse problems is 0.08, whereas the density of the dense problems is 0.20. In this case, the sparse problems are not very sparse, and the dense problems are not very dense. In Figs. 8 and 9, we increased the sparsity of the sparse problems and the density of the dense problems. In these cases,

the sparse problems are very sparse, whereas the dense problems are very dense, respectively. We used these three examples as boundary cases to analyze the performance of *m*-best 2-D utilizing the dynamic 2-D assignment algorithm switching mechanism, and statically configured with the auction and JVC algorithms, respectively.

In general, several observations can be made with respect to each of the plots. First, *m*-best 2-D utilizing the dynamic 2-D assignment algorithm switching mechanism outperformed *m*-best 2-D statically configured with the auction and JVC algorithms, respectively. The reason for this is that the former is able to switch, incurring negligible overhead in doing so, between the auction algorithm and the JVC algorithm, and thereby apply the most highly suited 2-D assignment algorithm for the sparse and dense problems contained in the simulated environments.

Another observation is that one may notice the gap widening between *m*-best 2-D statically configured with auction versus *m*-best 2-D statically configured with JVC as the density of the dense problems in the environment increases, i.e., as we go from Fig. 7, to Fig. 8, and then to Fig. 9. The reason for this is that as the density of the dense problems increases, so too does the benefits of using the JVC algorithm on the dense problems versus using the auction algorithm on the sparse problems, i.e., the performance gains of using the auction algorithm for the sparse problems are overshadowed by the performance gains of using the JVC algorithm on the dense problems. This same rationale explains why in Fig. 9 the plots of *m*-best 2-D statically configured with the JVC algorithm performs almost equally as well as *m*-best 2-D utilizing the dynamic 2-D assignment algorithm switching mechanism.

And lastly, as is evidenced by the plots, the 2-D assignment algorithm switching mechanism in *m*-best 2-D is not only beneficial in dynamic environments, but also static ones as well. The *m*-best 2-D utilizing dynamic 2-D assignment algorithm switching outperforms the *m*-best 2-D statically configured with the auction and JVC algorithms at both ends of the horizontal axis in each of the plots, where, in these areas, the simulated environment is considered fairly static. As the plots correctly show, the performance of *m*-best 2-D utilizing dynamic 2-D assignment algorithm switching should approach *m*-best-2-D statically configured with auction as the fraction of the sparse problems in the ATS environment increases, and likewise for the JVC algorithm as the fraction of the dense problems in the environment increases.

VI. CONCLUSION

Spurred on by recent interest in the *m*-best 2-D assignment algorithm for the data association problem, in this paper we described several improvements that enable much more efficient solutions to the data association problem, especially in dynamic environments. One improvement was to utilize a nonintrusive 2-D assignment algorithm switching mechanism, based on a problem sparsity threshold, enabling the auction and JVC 2-D assignment algorithms, each highly suited for sparse and dense problems, respectively, to efficiently solve the numerous 2-D assignment problems generated as part of the partitioning task. Another improvement was to utilize a multilevel parallelization of the data association problem, i.e., we parallelized the partitioning task and the data association interface task. This parallelization enables not only multiple 2-D assignment problems to be executed in parallel, but also the numerous gating tests, state estimates, covariance calculations, and likelihood function evaluations (used as cost coefficients in the 2-D assignment problem). Finally, using both simulated data and an ATS problem based on two FAA air traffic control radars, we demonstrated that more efficient solutions to the data association problem are possible when using the *m*-best 2-D assignment algorithm with our improvements incorporated.

REFERENCES

[1] Balas, E., Miller, D., Pekny, J., and Toth, P. (1991)
A parallel shortest augmenting path algorithm for the assignment problem.
Journal of the ACM, **38**, 4 (Oct. 1991), 985–1004.

[2] Balinski, M. (1985)
Signature methods for the assignment problem.
Operations Research, **33**, 3 (May–June 1985), 527–536.

[3] Bar-Shalom, Y., and Li, X. R. (1993)
Estimation and Tracking: Principles, Techniques and Software.
Boston, MA: Artech House, 1993.

[4] Bar-Shalom, Y., and Li, X. R. (1995)
Multitarget-Multisensor Tracking: Principles and Techniques.
Storrs, CT: YBS Publishing, 1995.

[5] Bertsekas, D. (1988)
The auction algorithm: A distributed relaxation method for the assignment problem.
Annals of Operations Research: Special Issue on Parallel Optimization, 1988.

[6] Bertsekas, D., and Castanon, D. (1991)
Parallel synchronous and asynchronous implementations of the auction algorithm.
Parallel Computing, **17** (1991), 707–732.

[7] Brogan, W. (1989)
Algorithm for ranked assignments with applications to multiobject tracking.
Journal of Guidance, **12**, 3 (May 1989), 357–364.

[8] Cox, I., and Miller, M. (1995)
On finding ranked assignments with application to multitarget tracking and motion correspondence.
IEEE Transactions Aerospace and Electronic Systems, **32**, 1 (Jan. 1995), 486–489.

[9] Cox, I., Miller, M., Danchick, R., and Newnam, G. (1997)
A comparison of two algorithms for determining ranked assignments with application to multitarget tracking and motion correspondence.
IEEE Transactions on Aerospace and Electronic Systems, **33**, 1 (Jan. 1997), 295–300.

[10] Danchick, R., and Newman, G. (1993)
A fast method for finding the exact N-best hypotheses for multitarget tracking.
IEEE Transactions on Aerospace and Electronic Systems, **29**, 2 (Apr. 1993), 555–560.

[11] Deb, S., Pattipati, K. R., and Bar-Shalom, Y. (1993)
A multisensor-multitarget data association algorithm for heterogeneous sensors.
IEEE Transactions on Aerospace and Electronic Systems, **29**, 2 (Apr. 1993), 560–568.

[12] Deb, S., Yeddanapudi, M., Pattipati, K., and Bar-Shalom, Y. (1997)
A generalized S-D assignment algorithm for multisensor-multitarget state estimation.
IEEE Transactions on Aerospace and Electronic Systems, **33**, 2 (Apr. 1997), 523–538.

[13] Drummond, O., Castanon, D., and Bellovin, M. (1990)
Comparison of 2-D assignment algorithms for sparse, rectangular, floating point, cost matrices.
Journal of SDI Panels on Tracking (Institute for Defense Analyses, Alexandria, VA), 4/1990 (Dec. 1990), 81–97.

[14] Jonker, R., and Volgenant, A. (1987)
A shortest augmenting path algorithm for dense and sparse linear assignment problems.
Journal of Computing, **38** (1987), 325–340.

[15] Kurien, T., Castanon, D., and Lublin, L. (1997)
Surveillance and sensor management algorithms for rapid detection of time-critical targets.
Technical report TR-794, Alphatech, Inc., 1997.

[16] Miller, M., Stone, H., and Cox, I. (1995)
Optimization to Murty's algorithm for determining a ranked set of assignments.
Technical report, NEC Research Institute, 1995.

[17] — (1994)
Multithreaded Programming Guide.
Sun Microsystems, Inc., Mountain View, CA., 1994.

[18] Murty, K. (1968)
An algorithm for ranking all the assignments in order of increasing cost.
Operations Research, **16** (1968), 682–687.

[19] Nagarajan, V., Chidambara, M., and Sharma, R. (1987)
Combinatorial problems in multitarget tracking—A comprehensive survey.
IEE Proceedings, **134**, 1 (1987), 113–118.

[20] Pattipati, K., and Deb, S. (1989)
Comparison of assignment algorithms with applications to the passive sensor data association problem.
In *Proceedings IEEE International Conference on Control and Application*, Jerusalem, Israel, 1989.

[21] Pattipati, K., Deb, S., Bar-Shalom, Y., and Washburn, R. (1992)
A new relaxation algorithm and passive sensor data association.
IEEE Transactions on Automatic Control, **37**, 2 (Feb. 1992), 197–213.

[22] Poore, A., and Rijavec, N. (1991)
Multitarget tracking, multidimensional assignment problems, and Lagrangian relaxation.
Proceedings of SDI Panels on Tracking (Aug. 1991), 51–74.

[23] Poore, A. B., and Rijavec, N. (1993)
A Lagrangian relaxation algorithm for multidimensional assignment problems arising from multitarget tracking.
SIAM Journal of Optimization, **3**, 3 (Aug. 1993), 544–563.

[24] Poore, A. B., and Rijavec, N. (1995)
Partitioning multiple data sets via multidimensional assignments and Lagrangian relaxation.
In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*.
Providence, RI: American Mathematical Society, 1995.

[25] Popp, R. L., Pattipati, K. R., Bar-Shalom, Y., and Ammar, R. A. (1997)
Shared-memory parallelization of the data association problem in multitarget tracking.
IEEE Transactions on Parallel and Distributed Systems, **8**, 10 (Oct. 1997), 993–1005.

[26] Popp, R., Pattipati, K., Bar-Shalom, Y., and Yeddanapudi, M. (1997)
Parallelization of a multiple model multitarget tracking algorithm with superlinear speedups.
IEEE Transactions on Aerospace and Electronic Systems, **33**, 1 (Jan. 1997), 281–290.

[27] Popp, R., Pattipati, K., and Bar-Shalom, Y. (1998)
Multitarget tracking algorithm parallelizations for air traffic surveillance.
Annals of Operations Research: Special Issue on Parallel Optimization, 1999.

[28] Popp, R., Pattipati, K., Bar-Shalom, Y., and Gassner, R. (1998)
An adaptive m-best S-D assignment algorithm and parallelization for multitarget tracking.
Presented at the IEEE Aerospace Conference, Snowmass, CO, Mar. 1998.

[29] Powell, M., Kleiman, S., Barton, S., Shah, D., Stein, D., and Weeks, M. (1991)
SunOS multi-thread architecture.
In *Proceedings of the USENIX '91 Conference*, Dallas, TX, 1991, 1–14.

[30] Reid, D. (1979)
An algorithm for tracking multiple targets.
IEEE Transactions on Automatic Control, **AC-24**, 6 (Dec. 1979), 423–432.

[31] Roecker, J. (1994)
A class of near optimal JPDA algorithms.
IEEE Transactions on Aerospace and Electronic Systems, **30**, 2 (Apr. 1994), 504–510.

[32] Tsaknakis, H. (1987)
Multiassignment algorithms for CSO tracking.
Alphatech, Inc., Burlington, MA, 1987.

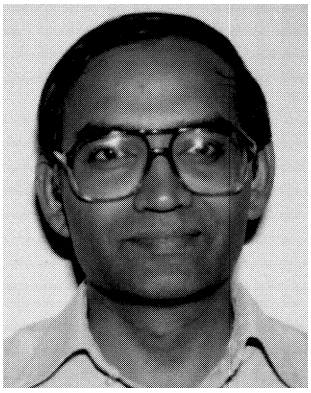
[33] Yeddanapudi, M., Bar-Shalom, Y., and Pattipati, K. R. (1997)
IMM estimation for multitarget-multisensor air traffic surveillance.
IEEE Proceedings, **85**, 1 (Jan. 1997), 80–94.

[34] Zhou, B., and Bose, N. (1993)
Multitarget tracking in clutter: Fast algorithms for data association.
IEEE Transactions on Aerospace and Electronic Systems, **29**, 2 (Apr. 1993), 352–363.

Robert Popp (S'93—M'94) received his B.A. and M.A. degrees in computer science from Boston University, Boston, MA (summa cum laude, distinction) in 1992, and his Ph.D degree in electrical engineering from the University of Connecticut, Storrs, in 1997.

He is presently a Senior Research Scientist and Associate Division Manager of the Fusion Technology and Systems Division of ALPHATECH, Inc. Dr. Popp is also an Adjunct Professor in the Computer Science and Engineering Department at the University of Connecticut.

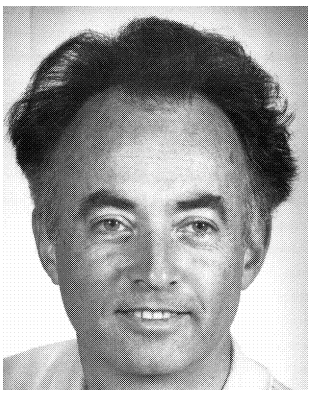
Dr. Popp has authored over 60 technical papers, reports, and proposals, and presently holds two patents. His expertise and research interests are in surveillance, multisensor multitarget tracking, parallel and distributed systems, adaptive computing systems, and computer networking. He is a member of various honorary and technical societies, including Phi Beta Kappa and Upsilon Pi Epsilon. Dr. Popp is listed in *Who's Who in the East*. He is presently serving as Associate Editor for the *IEEE Transactions on Systems, Man, and Cybernetics* (SMC), and as Chairman of the IEEE SMC Society's Adaptive Computing Systems (ACS) technical committee. He served as Session Chairman in the IEEE SMC '98 and HPC '99 Conferences, and is presently serving on the Program Committees for the IEEE Aerospace 2000 and HPC 2000 Conferences. He is also organizing and serving as Program Chairman for the IEEE SMC Society "Agent-based Technologies" workshop to be held in 2000.



Krishna R. Pattipati (S'77—M'80—SM'91—F'95) received the B.Tech degree in electrical engineering with highest honors from the Indian Institute of Technology, Kharagpur, in 1975, and the M.S. and Ph.D. degrees in systems engineering from the University of Connecticut, Storrs, in 1977 and 1980, respectively.

From 1980–1986 he was employed by ALPHATECH, Inc., Burlington, MA, where he supervised and performed research on fault-tolerant computer architecture modeling, human decision modeling, multi-target tracking, queuing networks, systems testing and diagnosis, and large-scale mixed-integer optimization. Since 1986, he has been with the University of Connecticut, where he is a Professor of Electrical and Systems Engineering.

Dr. Pattipati has published over 190 articles, primarily in the application of systems theory and optimization (continuous and discrete) techniques to large-scale systems. He has served as a consultant to ALPHATECH, Inc., IBM Research and Development, and Qualtech Systems, Inc. He was selected by the IEEE Systems, Man, and Cybernetics Society as the Outstanding Young Engineer of 1984, and received the Centennial Key to the Future award. He was elected a Fellow of the IEEE in 1995 for his contributions to discrete-optimization algorithms for large-scale systems and team decision making. He won the best technical paper awards at the 1985, 1990, and 1994 IEEE AUTOTEST Conferences, and at the 1997 Command and Control Conference. Dr. Pattipati is currently Editor of the *IEEE Transactions on SMC–Cybernetics* (Part B), and is the Vice-President for Technical Activities of the IEEE SMC Society.



Yaakov Bar-Shalom (S'63—M'66—SM'80—F'84) was born on May 11, 1941. He received the B.S. and M.S. degrees from the Technion, Israel Institute of Technology, in 1963 and 1967 and the Ph.D. degree from Princeton University, Princeton, NJ, in 1970, all in electrical engineering.

From 1970 to 1976 he was with Systems Control, Inc., Palo Alto, CA. Currently he is School of Engineering Distinguished Professor in the Dept. of Electrical and Systems Engineering and Director of the ESP Lab (Estimation and Signal Processing) at the University of Connecticut. His research interests are in estimation theory and stochastic adaptive control and he has published over 230 papers in these areas. In view of the causality principle between the given name of a person (in this case, "(he) will track", in the modern version of the original language of the Bible) and the profession of this person, his interests have focused on tracking.

He coauthored the monograph *Tracking and Data Association* (Academic Press, 1988), the graduate text *Estimation and Tracking: Principles, Techniques and Software* (Artech House, 1993), the text *Multitarget-Multisensor Tracking: Principles and Techniques* (YBS Publishing, 1995), and edited the books *Multitarget-Multisensor Tracking: Applications and Advances* (Artech House, Vol. I 1990; Vol. II 1992). He has been elected Fellow of IEEE for "contributions to the theory of stochastic systems and of multitarget tracking." He has been consulting to numerous companies, and originated the series of Multitarget-Multisensor Tracking short courses offered via UCLA Extension, at Government Laboratories, private companies, and overseas. He has also developed the commercially available interactive software packages MULTIDAT TM for automatic track formation and tracking of maneuvering or splitting targets in clutter, PASSDAT TM for data association from multiple passive sensors, BEARDAT TM for target localization from bearing and frequency measurements in clutter, IMDAT TM for image segmentation and target centroid tracking and FUSEDAT TM for fusion of possibly heterogeneous multisensor data for tracking. During 1976 and 1977 he served as Associate Editor of the *IEEE Transactions on Automatic Control* and from 1978 to 1981 as Associate Editor of *Automatica*. He was Program Chairman of the 1982 American Control Conference, General Chairman of the 1985 ACC, and Co-Chairman of the 1989 IEEE International Conference on Control and Applications. During 1983–1987 he served as Chairman of the Conference Activities Board of the IEEE Control Systems Society and during 1987–1989 was a member of the Board of Governors of the IEEE CSS. Currently he a member of the Board of Directors of the International Society of Information Fusion. In 1987 he received the IEEE CSS Distinguished Member Award. Since 1995 he is a Distinguished Lecturer of the IEEE AESS. He is co-recipient of the M. Barry Carlton Award for the best paper in the *IEEE Transactions on Aerospace and Electronic Systems* in 1995, and the 1998 University of Connecticut AAUP Excellence Award for Research.