Algorithm for Ranked Assignments with Applications to Multiobject Tracking

William L. Brogan
University of Nebraska, Lincoln, Nebraska

A sequential algorithm is presented for obtaining a cost-ranked set of solutions to the assignment problem. Concurrently, a conservative bound can be calculated that indicates how many of the ranked set are better than other potential assignments that may have been missed. Applications to two important strategic defense initiative measurement-assignment problems, namely the cycle-to-cycle and the sensor-to-sensor problems, are demonstrated. The procedure is useful for initiating new object tracks as well as for assigning incoming measurements to established tracking filters. Knowledge of the ranked set of assignments, as opposed to a single optimum, is important because of measurement uncertainties. The prudent course may be to initiate several tentative tracks in certain close-call situations.

Introduction

CONSIDER two sets of quantities, $Y = \{y_i, i = 1,m\}$ and $Z = \{z_j, j = 1,n\}$. Assume without loss of generality that $n \geq m$. The assignment problem is the process of assigning each y_i to some z_j, with the stipulation that at most one y_i is assigned to each z_j. In nonsquare problems, $n - m$ of the z_j will have no y_i assigned to them. In the *optimal* assignment problem, there is a known cost d_{ij} associated with assigning y_i to z_j. Let these costs form the elements of an $m \times n$ matrix D. The problem is to make i,j assignments in such a way as to minimize the sum of the m costs. That is, for each row i of D, a unique column j is selected such that the sum of the m selected d_{ij} elements is minimized.

A succinct statement of the optimal assignment problem can be written in terms of the binary variables x_{ij}. If y_i is assigned to z_j, then x_{ij} is 1; otherwise, it is 0. Select the nm values of x_{ij} to minimize

$$\sum_{j}^{n} \sum_{i}^{m} x_{ij} d_{ij}$$

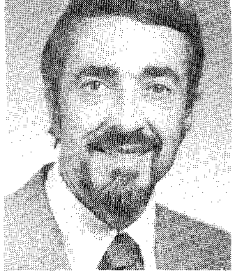
subject to

$$\sum_{j=1}^{n} x_{ij} = 1 \text{ for each } i, \qquad 1 \leq i \leq m$$

$$\sum_{i=1}^{m} x_{ij} \leq 1 \text{ for each } j, \qquad 1 \leq j \leq n$$

Munkres[1] developed an algorithm for solving the square ($n = m$) problem. Silver[2] provided an efficient implementation of the Munkres algorithm. Bourgeois and LaSalle[3] provided a generalization for nonsquare problems. A sparse matrix version of Munkres algorithm has been presented by Stephens and Krupa.[4] Wong[5] also presents results on improving the computational efficiency of Munkres. Integer programming has been applied to multiobject tracking.[6,7] Integer programming, combined with branch-and-bound techniques, has been applied to a wide class of assignment problems.[8] Although solutions to the optimal assignment problem are thus known, these results are not entirely satisfactory in all cases.

The 0–1 integer programming approach involves 2^{nm} possible assignment combinations, although most of these will not satisfy the constraints. Some integer programming approaches require that all feasible assignment sequences be explicitly enumerated in the problem setup stage. This is unwieldy in large problems, and hence this approach is not pursued here. Other methods require the implicit evaluation of all combinations[7] by methods usually referred to as branch-and-bound algorithms.[9] The method of this paper may be categorized as a variation of the branch-and-bound approach. However, it actually evolved out of a dynamic programming view of the problem. The integer programming and the Munkres approaches seek a single "optimal" answer. No indication is given as to what the second, third, and subsequent best choices might be, or how close these are to the first in terms of cost. In many situations where noisy measurements are involved in determining the sets Y and/or Z, such secondary information is very useful if not essential. In stochastic assignment problems, the costs are random variables and the question of best assignment is actually a probabilistic decision process.



William L. Brogan is a professor in the Department of Electrical Engineering at the University of Nebraska in Lincoln. Dr. Brogan received a B.S.M.E. degree from the State University of Iowa, and the M.S. and Ph.D. in Engineering from UCLA. His professional career has been divided approximately equally between the aerospace industry and academia. He is the author of *Modern Control Theory* (Prentice-Hall, 1985) and a chapter in *Advances in Control Systems*, Vol. 6, (Academic, 1968). Dr. Brogan is active in presenting continuing education short courses and has served as a consultant to several industrial firms.

Received May 21, 1987; revision received Jan. 19, 1988. Copyright © American Institute of Aeronautics and Astronautics, Inc., 1988. All rights reserved.

The approach to the optimal assignment problem that is presented here determines a ranked set of p assignments, along with their associated costs. This is a sequential process in which only the best p possibilities are kept at each step of algorithm. Since there are $n!/(n-m)!$ feasible i,j assignment combinations, it is possible to miss certain good final combinations. A sufficient condition is provided that determines an integer $q \le p$ such that the first q assignments are guaranteed to be optimal. A good deal of empirical testing of the approach to realistic strategic defense initiative (SDI) track-initiation problems has shown the procedure to be very effective in a dense target environment.

Several versions of the SDI problem can be defined and solved using the same basic algorithm. Differences exist only in setting up the cost matrix D. For example, the set Y may be a set of measurements from a given sensor to m targets at time t_k. The set Z could be the set of measurements made by the same sensor at time t_{k+1}. This is referred to here as the cycle-to-cycle assignment problem. The number n could be larger than m because of targets just arriving in the field of view or they could be false returns because of clutter or other factors. If the number in the second measurement cycle is smaller because of data dropout or some other reason, the transpose of the D matrix can be used in place of D. The costs in D would be the sum-squared miss distances, in measurement space, between the two sets of data after correction for the motion during the cycle period $t_{k+1} - t_k$. Several methods of accounting for intercycle motion have been found effective for the SDI problem and will be discussed. The most attractive ones require no a priori knowledge of the dynamics of the targets being tracked.

If two sensors observe a common group of targets, it is of interest to identify a common object seen by each, tagged i by the first sensor and j by the second. This is the sensor-to-sensor assignment problem also analyzed in this paper. It is not necessary to assume time synchronization of the measurements.

In a third version of the problem, the elements y_i are expected measurements as calculated in m tracking filters. The actual measurements are the z_j elements, which must be assigned to the correct filters.

The basic assignment algorithm is presented first, followed by the establishment of sufficient conditions for optimality. The remainder of the paper applies these results to the SDI measurement-assignment and track-initiation problems.

Recursive Assignment Algorithm

The algorithm is recursive in that each item y_i is, in turn tentatively assigned to some z_j that is as yet unassigned. The $m \times n$ cost matrix D and a specified buffer size $p \ge n$ are given at the outset. Define a linear array of p real numbers $C_k(j)$, $j = 1, \ldots, p$ and a growing $k \times p$ array of integers $A(i,j)$, $i = 1, \ldots, k$; $j = 1, \ldots, p$. The array C_k will contain the p smallest scores found after assigning the first k items y_i to selected items z_j. After the kth stage, the array A will contain the sets of i,j assignments that have yielded the scores in C_k. That is, column 1 of A, $\boldsymbol{a}_1$, will contain the set of integer values for j that have been paired with $i = 1, 2, \ldots, k$, respectively, to yield the best score $C_k(1)$. Column 2 of A will contain the j values that gave the second best score $C_k(2)$ and so on. The array A grows longer by one row at each stage of the assignment process. The final $m \times p$ A matrix contains the desired p best assignments. Note that the buffer width p can be larger than n.

The array C is initialized as $C_0 = [d_{1,j}, \text{BigVal}]$ where BigVal represents a padding, if necessary, from $d_{1,n}$ out to width p with fictitious costs that are large enough to guarantee they will never be selected. The first stage of the algorithm determines C_1 and the first row of A. This amounts to ranking the elements in C_0 from smallest to largest. The column indices j of $a_{1,j}$ that give this ranked order are stored in row 1 of A. For example, consider the 4×5 cost matrix

$$D = \begin{bmatrix} 4 & 1 & 10 & 3 & 6 \\ 1 & 3 & 4 & 6 & 10 \\ 2 & 0.1 & 17 & 5 & 11 \\ 12 & 5 & 2.5 & 8 & 6 \end{bmatrix} = [d_{ij}]$$

Here, $n = 5$ and $m = 4$. The buffer width p is selected arbitrarily as 7 for this example. As p increases, so does computer burden, but the chance of missing good assignments decreases as will be seen in the next section. In general, the required buffer size is problem-dependent. The algorithm starts with

$$C_1 = [1 \quad 3 \quad 4 \quad 6 \quad 10 \quad 1.E6 \quad 1.E6]$$

and

$$A(1,j) = [2 \quad 4 \quad 1 \quad 5 \quad 3 \quad 6 \quad 7]$$

The size used for BigVal is 10^6 in this example. For all stages from $k = 1$ onward, the general algorithm procedure is as follows:

1) Compute $b_{i,j} = C_k(j) + d_{k+1,i}$, provided that $i \notin a_j$ (all feasible combinations of i with all unused j).

2) Form C_{k+1} by sorting the p smallest values of $b_{i,j}$.

3) Append the values of i to the column a_j and rearrange the new a columns to put them in correspondence with the ranked scores produced.

The recursion can be stated in equation form as

$$C_{k+1}(b) = \min_{\substack{i,j \\ i \notin a_j}}{}^{b}\{C_k(j) + d_{k+1,i}\}, \qquad \begin{array}{l} i = 1, \ldots, n \\ j = 1, \ldots, p \\ b = 1, \ldots, p \end{array} \tag{1}$$

where $\min^b\{\ \}$ means the bth smallest. That is, the true minimum is $\min^1$, the second smallest is $\min^2$, and so on. If the values of i and j that give the bth minimum are called $i(b)$ and $j(b)$, then the corresponding A matrix after the $k+1$th stage would be

$$A = \begin{bmatrix} a_{j(1)} & a_{j(2)} & \cdots & a_{j(p)} \\ i(1) & i(2) & \cdots & i(p) \end{bmatrix}$$

Using this procedure to continue the previous example gives, after stage $k = 2$:

$$C_2 = [2 \quad 4 \quad 5 \quad 6 \quad 7 \quad 7 \quad 7]$$

$$A = \begin{bmatrix} 2 & 4 & 2 & 4 & 1 & 5 & 4 \\ 1 & 1 & 3 & 2 & 2 & 1 & 3 \end{bmatrix}$$

after stage $k = 3$:

$$C_3 = [4.1 \quad 7 \quad 7 \quad 7.1 \quad 7.1 \quad 8 \quad 9.0]$$

$$A = \begin{bmatrix} 4 & 2 & 2 & 4 & 5 & 4 & 4 \\ 1 & 1 & 3 & 3 & 1 & 2 & 3 \\ 2 & 4 & 1 & 2 & 2 & 1 & 1 \end{bmatrix}$$

after the final stage $k = 4$:

$$C_4 = [6.6 \quad 9.5 \quad 9.6 \quad 10.1 \quad 10.5 \quad 13.0 \quad 13.0]$$

$$A = \begin{bmatrix} 4 & 2 & 5 & 4 & 4 & 2 & 2 \\ 1 & 1 & 1 & 1 & 2 & 3 & 1 \\ 2 & 4 & 2 & 2 & 1 & 1 & 4 \\ 3 & 3 & 3 & 5 & 3 & 5 & 5 \end{bmatrix}$$

This result states that the minimum score found was 6.6 and it was achieved by the assignment pairing

y_i	1	2	3	4
z_j	4	1	2	3

The second best score is 9.5 and is achieved by using column 2 of A for the selection of z_j and so on.

There are a total of $n!/(n-m)! = 120$ pairings for this example. These choices have not all been exhaustively checked. It can be shown that the first five of the assignment sets are optimal, but two choices were missed that give costs between 10.5 and 13.0. Sufficient conditions are established next that give an integer $q \le p$ such that at least the first q solutions are optimal. The sufficiency conditions have been found to be overly conservative in many numerical tests.

Sufficient Conditions for Optimality

At each step in the algorithm, only p cost terms are retained in C_k. Other $b_{i,j}$ terms are evaluated but not retained. Still other combinations are never evaluated because of terms discarded in earlier steps. Thus, at each stage k, all of the potential combinations fall into one of three sets

$S_k = \{p$ saved cost terms$\}$; when S_k is rank-ordered, it becomes C_k.
$T_K = \{t_i$, those $b_{i,j}$ terms computed but truncated or discarded$\}$.
$U_k = \{u_i$, those uncomputed and unknown cost combinations$\}$.

It is known at each step that for all $t_i \in T_k$, $t_i \ge C_k(p)$, that is, all truncated terms are at least as large as the maximum saved term. To establish optimality of the algorithm, it must be established that there are no unknown terms $u_i \in U_k$ that are smaller than $C_k[q(k)]$ for some positive integer $q(k)$. For stage $k = 1$, $S_1 = \{d_{1,j}\}$ (elements of row 1 of D, perhaps padded by BigVal), and $T_1 = \{\emptyset\}$, $U_1 = \{\emptyset\}$, where $\{\emptyset\}$ is the empty set. For stage $k = 2$, $\min T_2 \ge \max S_2 = C_2(p)$ and $U_2 = \{\emptyset\}$. Stage $k = 3$ is the first one for which U_k is nonempty. For all $u_i \in U_3$,

$$u_i = t_v + d_{3,j} \qquad \text{for some} \qquad t_v \in T_2 \tag{2}$$

Since every $t_i \in T_2$ satisfies $t_v \ge \max S_2 = C_2(p)$, it is clear that

$$\min U_3 \ge C_2(p) + \min_j\{d_{3,j}\} \tag{3}$$

Call this lower bound on the unknown U terms $Q(3)$. If for some positive integer $q(3)$, it is true that $C_3[q(3)] \le Q(3)$, then the first $q(3)$ cost terms in C_3 are indeed optimal. The general $k+1$ stage is slightly more involved since the unknown terms could derive from unknown terms on previous stages as well as truncated terms from one stage back. That is, the set of unconsidered terms at stage $k+1$ is U_{k+1}, and each member u_i of this set is one of two possible types

$$u_i = \{t_v + d_{k+1,j}\} \qquad t_v \in T_k$$
$$u_i = \{u_r + d_{k+1,j}\} \qquad u_r \in U_k \tag{4}$$

Therefore, $\min\{u_i\} = \min\{\min[t_v, u_r] + \min[d_{k+1,j}]\}$. Using the previously established bounds $\min T_k \ge C_k(p)$ and $\min U_k \ge C_k[q(k)]$ gives the new lower bound

$$\min U_{k+1} \ge C_k[q(k)] + \min\{d_{k+1,j}\} = Q(k+1) \tag{5}$$

Thus, the number of guaranteed optimal terms in the cost array C_{k+1} and the allocation matrix A is given by

$$q(k+1) = \max_j\{j \mid C_{k+1}(j) \le Q(k+1)\} \tag{6}$$

The values of $q(k)$ are easily computed at each stage of the algorithm and can not be allowed to increase at any step. At step 1, $q(1) = p$ (the full buffer width) since nothing is lost until at least the third step. Applying Eqs. (5) and (6) to the previous example gives $Q(2) = \text{BigVal}$ and $q(2) = 7$. On cycle 3, Eqs. (5) and (6) give $Q(3) = 7.1$ and $q(3) = 5$. Finally, $Q(4) = 9.6$ and $q(4) = 3$, meaning only the first three assignment sets are known with certainty to be optimal.

It is easy to establish that the foregoing results will depend on the order in which rows of D are processed. Some orders may produce nonpositive final q values, meaning no guarantee of optimality can be made. Other orders may produce relatively large q values and hence strong guarantees that the optimal results have been found. It is conjectured, based on empirical results, that a good ordering is one having the rows arranged with maximum to minimum ranking of

$$\Delta d_k = \max_j\{d_{k,j}\} - \min_j\{d_{k,j}\} \tag{7}$$

Bringing the D matrix into this row ordering is not essential to the validity of the previous sufficiency conditions. However, it has been found empirically that this order usually leads to a stronger result [larger $q(m)$]. When the previous example is reworked with this row order, the final buffer full of costs are $C_4 = [6.6\ 9.5\ 9.6\ 10.1\ 10.5\ 12.6\ 13.1]$ and $q(4) = 4$. Thus, although the first five solutions are the same as before, the row ordering gives more confidence to the answers.

Figure 1 gives an interpretation to Eqs. (5) and (6) and is useful in understanding the origin of the row-ordering suggestion. Figure 1 assumes that at the kth stage, the values of the array $C_k(j)$ and $q(k)$ have been found. All considered $b_{i,j}$ values are obtained by adding d_{ij} increments to C_k. Those that exceed the lower bound for unknown terms $\min U_{k+1} = Q(k+1)$ are potentially larger than some ignored terms. Hence, the optimality of any b_{ij} term in the cross-hatched region cannot be assured. The further to the right that the $C_k(j) + \max\{d_{k+1,j}\}$ curve crosses the $Q(k+1)$ bound, the fewer $b_{i,j}$ terms that will be at risk. Note that this implies the smaller the Δd_{k+1}, the better for a given size ΔC_k. For the contrived example sketched in Fig. 1, the actual nonlinear upper-bound curve shows $q = 3$ as the maximum integer that can be guaranteed optimal. The exact curvature of the upper-bound curve $C_k(j) + \max\{d_{k+1,j}\}$ is unknown. Consider the linear approximate curve that crosses the min U_{k+1} bound at q'. The largest integer less than q', namely 2, would be the estimated optimal bound with the straight-line approximation. This approximation is never used except as a means of explaining the relationship between Δd_{k+1} and q. From similar triangles,

$$q(k) - q'(k+1) = \frac{\Delta d_{k+1}}{\Delta C_k}[q(k) - 1]$$
$$q'(k+1) = q(k) - \frac{\Delta d_{k+1}}{\Delta C_k}[q(k) - 1] \tag{8}$$

Define the ratio $\Delta d_{k+1}/\Delta C_k = \mathscr{R}$. If $\mathscr{R} = 1$, then $q'(k+1) = 1$.

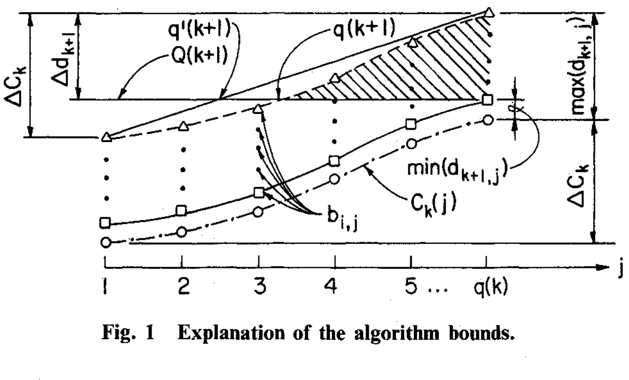


Fig. 1 Explanation of the algorithm bounds.

If $\mathscr{R} < 1$, then $1 < q'(k+1) < q(k)$. Finally, if $\mathscr{R} > 1$, then $q'(k+1) < 1$, that is, no optimality can be claimed. Since $\Delta d_1 = \Delta C_1$, it is reasonable to require that $\Delta d_2 / \Delta d_1 < 1$. It is further suggested that this rank ordering be continued to give

$$\Delta d_1 > \Delta d_2 > \Delta d_3 > \cdots > \Delta d_n \tag{9}$$

This ordering has given good empirical results in many cases, including the previous numerical example. Overly conservative estimates are often obtained, however. In the previous numerical example, a value of $q = 4$ was obtained, whereas in truth the first five solutions are globally optimum. The reason for the conservatism is clear. It is possible that at least $q(k)$ of the considered b_{ij} values can be found below the min U_{k+1} bound. If so, no loss of optimality has occurred at this step so that $q(k+1) = q(k)$ and no solutions have been overlooked that are better than the first $q(k+1)$ values.

Although it would be highly desirable to be able to compute the foregoing bounds on an a priori basis and thus use the results to select the buffer size, a method for doing this is not known. The concurrent monitoring process just presented is useful in that it raises a flag whenever the value of $q(k)$ falls below the acceptable value. If this happens, a larger buffer size, or in some cases a different row ordering, must be used in a new solution attempt.

SDI Applications

Cycle-to-Cycle Measurement-Association Problem

Consider one cycle of a tracking sensor with m line-of-sight (LOS) unit vector measurements to targets. Let each unit vector define a point y_i in an inertially referenced plane. For illustration, let each point have two components, "azimuth" and "elevation" angles. On a second cycle, measurements produce the points z_j in the same plane. The cycle-to-cycle measurement-assignment problem consists of selecting i, j pairings that cause the best overall connection between points i in cycle 1 with points j in cycle 2. Both sensors and objects being tracked are in motion. The between-cycle motion is the vector $\boldsymbol{m}_i$, so that

$$z_j = y_i + \boldsymbol{m}_i \tag{10}$$

The subscripts i and j do not necessarily agree even when a single point is being discussed, because points on the two cycles are numbered independently of each other. It is not known which i goes with which j. If all the points in the cluster of targets are moving with a high degree of coherence, as in a typical SDI boost-phase scenario, all of the between-cycle motion vectors $\boldsymbol{m}_i$ are largely the same, differing only by small and more or less random amounts. That is,

$$\boldsymbol{m}_i = \boldsymbol{m} + \delta\boldsymbol{m}_i \qquad \text{for all} \qquad i \tag{11}$$

When random observation noise $\boldsymbol{v}_j$ is considered, each cycle 2 point can be written as

$$z_j = z_{j(\text{true})} + \boldsymbol{v}_j \tag{12}$$

Let the measurement noise terms in cycle 1 be $\boldsymbol{u}_i$ and $\hat{\boldsymbol{m}}$ be an estimate of $\boldsymbol{m}$. Any errors in the estimate can be included in the deviation $\delta\boldsymbol{m}_i$. Then, an estimate of where a point on cycle 1 should be on cycle 2 is

$$y_i = y_{i(\text{true})} + \boldsymbol{u}_i + \hat{\boldsymbol{m}} \tag{13}$$

The vector difference can be expressed as

$$\begin{aligned} \boldsymbol{e}_{ij} &= y_i - z_j \\ &= y_{i(\text{true})} - y_{j(\text{true})} + (\boldsymbol{u}_i - \boldsymbol{v}_j) - \delta\boldsymbol{m}_j \end{aligned} \tag{14}$$

The subscript number i is inherited from cycle 1, whereas j is inherited from cycle 2. If i and j happen to correspond to the same point, then $\boldsymbol{e}_{ij}$ is due entirely to noise and random motion and hence should be small. The measurement-assignment algorithm developed earlier is applied to the cost matrix D whose elements are

$$d_{ij} = (\boldsymbol{e}_{ij})^T(\boldsymbol{e}_{ij}) \tag{15}$$

In actual operation, D is formed using the cycle 2 measured points directly, along with equivalent points computed from cycle 1 measurements. This involves estimating the mean motion vector $\boldsymbol{m}$. Three methods of doing this have been considered.

1) *Motion model.* If good object tracks have already been established, model extrapolation from cycle 1 to 2 could be used. The $\boldsymbol{e}_{ij}$ are exactly the measurement residuals that would be formed by comparing each z_j measurement with each of the (extended) Kalman filters being used to maintain the existing tracks through points y_i. Rather than Eq. (15), the inverse covariance-weighted residuals may be used to form d_{ij}. This method has proven very effective once good tracks have been established, but it is not applicable to the track-initiation problem.

2) *Maximum correlation.* Even though the mean motion $\boldsymbol{m}$ may be large, complicated, and unknown, the two sensor scan images will look very similar as long as the noise and random individual motions are relatively small. Thus, the mean motion can be determined by shifting the cycle 1 pattern until maximum correlation with the cycle 2 pattern is located. In simulation tests, this has been done as follows. The points in cycle 1 are normalized onto the unit square by making use of the maximum and minimum azimuth and elevation angles appearing in that cycle

$$y_i' = \begin{bmatrix} (Az_i - Az_{\min})/(Az_{\max} - Az_{\min}) \\ (El_i - El_{\min})/(El_{\max} - El_{\min}) \end{bmatrix} \tag{16}$$

The unit square is then quantized into a rectangular grid (typically 20 × 20 pixels). A matrix A is formed whose values a_{ij} are the number of targets that fall within that bin. The same-size pixels are used to quantize some region of the sensor space large enough to capture all the cycle 2 points. A known bias can be used to position this second grid roughly in the area of the cluster of objects seen on cycle 2. The same min and max values of Eq. (16) are used to normalize all cycle 2 points (with the known bias taken into account). In the typical SDI scenario, with long viewing ranges and typical cycle times, the second quantized region might be twice the size of the first to allow for uncertain motions and noise. A matrix B is formed, with b_{ij} being the number of cycle 2 targets falling within the ijth bin. Then, the matrix A is

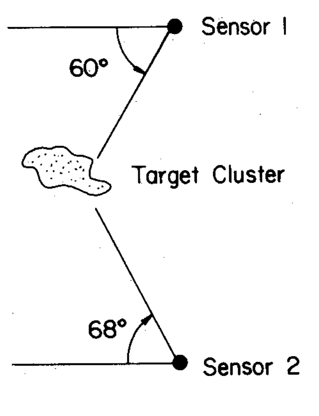


Fig. 2 Viewing geometry.

Table 1 Cycle-to-cycle assignment errors for satellite 1 (better geometry)

Noise level	Cycle time, s					
μrad	5	10	15	20	25	30
0	(1, 2) 0 1	(1, 3) 0 1	(1, 4) 0 0.733	(1, 5) 0 0.667	(1, 6) 0 0.667	(1, 7) 0 0.667
10		(1, 3) 0 0.467		(1, 5) 0 0.467		
30	(1, 2) 0 0.412	(1, 3) 0 0.333	(1, 4) 0 0.333	(1, 5) 2 0.176	(1, 6) 0 0.2	(1, 7) 0 0.4
	(2, 3) 0 0.267	(2, 4) 0 0.267	(2, 6) 0 0.267			
		(3, 5) 2 0.176		(3, 7) 2 0.267		
50		(1, 3) 2 0.2		(1, 5) 2 0.133		

Table 2 Cycle-to-cycle assignment errors for satellite 2 (weaker geometry)

Noise level	Cycle time, s					
μrad	5	10	15	20	25	30
0	(1, 2) 0 0.857	(1, 3) 0 0.858	(1, 4) 0 0.857	(1, 5) 0 0.762	(1, 6) 0 0.762	(1, 7) 0 0.619
30	(1, 2) 2 0.558	(1, 3) 2 0.421	(1, 4) 5 0.235	(1, 5) 10 0.235	(1, 6) 5 0.522	(1, 7) 2 0.353
	(2, 3) 4 0.4	(2, 4) 5 0.294	(2, 6) 4 0.263			
		(3, 5) 5 0.467		(3, 7) 2 0.368		

correlated with the matrix B for all possible Δi, Δj shifts

$$C(\Delta i,\Delta j) = \frac{\sum_i \sum_j a_{ij} b_{i+\Delta i, j+\Delta j}}{\sum_i \sum_j a_{ij} a_{ij}} \tag{17}$$

The shift $\Delta i_m, \Delta j_m$ that maximizes the correlation is the fine-tuning portion of the estimated shift $\boldsymbol{m}$ that would be added to the roughly selected bias terms mentioned earlier. Once $\boldsymbol{m}$ has been determined, all cycle 1 points are used to calculate effective cycle 2 positions using Eq. (13). Then, the cost matrix D is formed following Eq. (15) and the assignment algorithm then applied.

The advantage of this approach is that it does not require any knowledge of a motion model. All that is required is that the sensor be able to recognize the same cluster on the second cycle. This method is capable of accurately determining a mean motion in a dense target environment. With a lesser number of targets, a potential problem can arise. If the bin sizes are too coarse, the resolution of the determined shift is poor ($\pm$half a bin size). If the bin sizes are made very small, then each one will contain few if any targets and multiple solutions arise for the location of the maximum (but small) correlation.

3) *Simple normalization.* This method bypasses the explicit solution for $\boldsymbol{m}$. Both cycles 1 and 2 are normalized onto unit squares, with Eq. (16), but with different normalization constants appropriate for each cycle. The difference vectors between these normalized points $\boldsymbol{y}_i'$ and $\boldsymbol{z}_j'$ are then used directly to form $\boldsymbol{e}_{ij}$. The D matrix is still given by Eq. (15) and the assignment algorithm proceeds as before. The advantages of this scheme are the computational savings of not computing the multitude of correlations and the improved ability to correctly associate points in patterns that are shrinking or growing between cycles.

Note that no matter which of the aforementioned approaches is used, correctly solving the minimization problem posed at the beginning of this paper does not assure that the correct measurement assignments have been made. Consideration of Eq. (14) shows that d_{ij} is a chi-squared random variable if the measurement and motion errors are Gaussian. (Justification for a Gaussian motion error model may not be compelling, but the analytical advantages gained are substantial.) Further, d_{ij} is a central chi-square for a correct ij pairing and a noncentral chi-square otherwise. See Ref. 10 for a discussion of the probability of error, which obviously depends on the true separation between points j_1 and j_2, the statistics of the measurement noises and the errors or scatter in the between-cycle motions $\delta \boldsymbol{m}_i$.

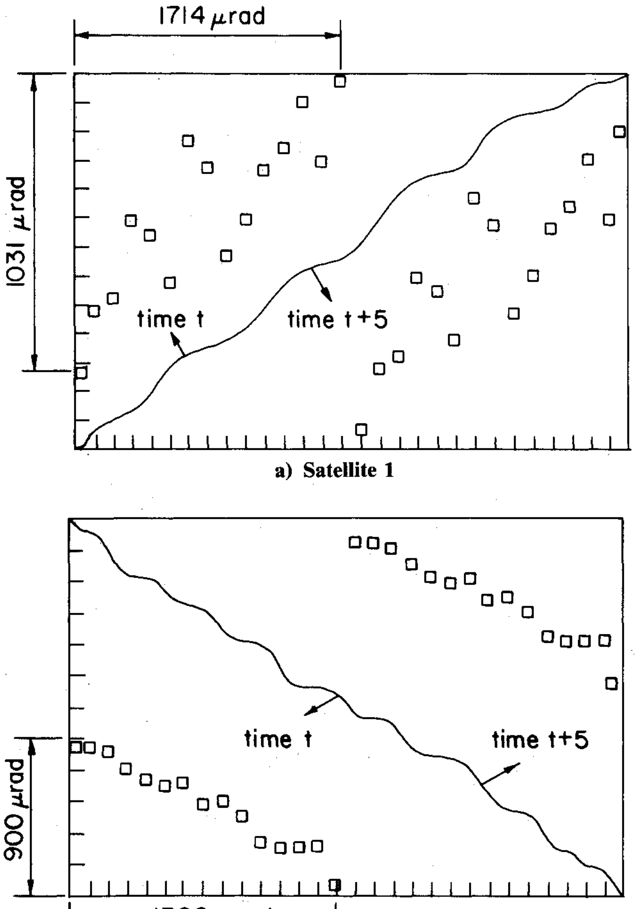
Numerical Results for the Cycle-to-Cycle Assignment Problem

Realistic trajectories were simulated for a cluster of 15 boosters during the early minutes of flight. Two observer satellites were in orbits appropriate to the SDI problem. Figure 2 shows the relative azimuth angles between the sensors and the cluster at a typical instant. The out-of-plane elevation angles are less than 5 deg. Seven cycles of noise-free unit vectors from sensors to targets were generated, one every 5 s. From these, azimuth and elevation angles were computed. The noise-free view of the cluster on two consecutive cycles is shown in Figs. 3a and 3b for sensors 1 and 2, respectively. Various levels of random noise were added to the measurements and the cycle-to-cycle assignment procedure applied. Booster motion was simulated without major doglegs at staging, so the $\delta \boldsymbol{m}_i$ random motion effects were rather small. The larger apparent separations seen by satellite 1 yield better assignment results. Cycle time and measurement noise levels were varied, resulting in Table 1 for satellite 1 and Table 2 for satellite 2. These tables show cycle pairs (in parenthesis), followed by the number of assignment errors and maximum correlation found using Eq. (17). Both methods 2 and 3 of accounting for intercycle motion were tested, with similar results. Tables 1 and 2 report method 2. The results of Tables 1 and 2 are for single trials under each of the stipulated conditions. Random noise will cause these results to vary somewhat. A set of 5 Monte Carlo repetitions were made using satellite 1 with a 10 s cycle period. Figure 4 shows the maximum, minimum, and average number of errors vs noise level. Although 5 replications do not represent a large statistical sample, Fig. 4 does give an indication of the performance to be expected. The results become unreliable at low correlations, say, 0.4 or less. Low correlations can be caused by high measurement noise or large unpredicted motions δm_i compared with the true separation distances as foreshortened by viewing geometry. Many of the errors were "pair flips," which are defined in detail later.

Sensor-to-Sensor Measurement-Association Problem

At any particular time, the locations of two sensors s_1 and s_2 are known. The unit vector that defines the direction from sensor 1 to target i is r_{1i}. The unit vector from sensor 2 to target j is r_{2j}. If these unit vectors are measured without error, it is possible to determine whether both sensors are pointing at the same point, and if so, the location of that point. The lack of measurement-time synchronization is corrected for by least-squares fitting a cubic polynomial to each component of a time sequence of noisy LOS measurements. This fitting process also provides some noise smoothing. Note that this assumes that the cycle-to-cycle association problem has already been solved. Each component of each LOS vector is approximated by an equation of the form

$$x(t) = c_0 + c_1(t - t_0) + c_2(t - t_0)^2 + c_3(t - t_0)^3 \tag{18}$$



a) Satellite 1

b) Satellite 2

Fig. 3 Two cycles of measurement data.

where t_0 is the mid-data span time. At least four cycles of data are required to solve for the c_i coefficients. After determining all the fit coefficients, reconstructed vectors r'_{1i} and r'_{2j} can be computed at some set of synchronized times t_k for all i and j targets. In the numerical results to follow, these times were selected as the actual measurement times of one of the sensors. The reconstructed vectors are no longer unit vectors, so they must be renormalized to give the final set of time-synchronized unit LOS vectors

$$\begin{aligned} r_{1i}(t_k) &= r'_{1i}(t_k)/\|r'_{1i}(t_k)\| \\ r_{2j}(t_k) &= r'_{2j}(t_k)/\|r'_{2j}(t_k)\| \end{aligned} \tag{19}$$

In the following discussion, these time-synchronized measurements will be used. The position of the ith target pointed to by sensor 1 is

$$p_{1i} = s_1 + \alpha r_{1i} \tag{20}$$

and the jth target pointed to by sensor 2 is located at

$$p_{2j} = s_2 + \beta r_{2j} \tag{21}$$

where α and β are as yet unknown ranges to the targets. The vector distance between these two targets is e_{ij}. If, in fact, the two targets are the same, e_{ij} would be zero in the absence of all errors. In general, e_{ij} will not be zero, but the ranges α and β will be computed to the point of closest approach between the two rays. That is, α and β will be selected to minimize

$$e_{ij}^T e_{ij} = \|s_i - s_2\|^2 + \alpha^2 + \beta^2 + 2\alpha p - 2\beta q + 2\alpha\beta a \tag{22}$$

where three scalar product terms

$$a = r_{1i}^T r_{2j}, \quad p = (s_1 - s_2)^T r_{1i}, \quad q = (s_1 - s_2)^T r_{2j} \tag{23}$$

have been defined. The minimization is accomplished by setting the partial derivatives with respect to α and β to zero and solving two simultaneous linear equations, giving

$$\alpha = (aq - p)/(1 - a^2) \quad \text{and} \quad \beta = (q - ap)/(1 - a^2) \tag{24}$$

After determining values for α and β, the squared magnitudes of all e_{ij} vectors of Eq. (22) can be calculated. These terms are used to make up the cost matrix D to which the assignment algorithm will be applied. After the assignment decisions have been made, the position of each target can be initiated by using the appropriate ranges α and β along with the corresponding LOS vectors in Eqs. (20) or (21). A se-

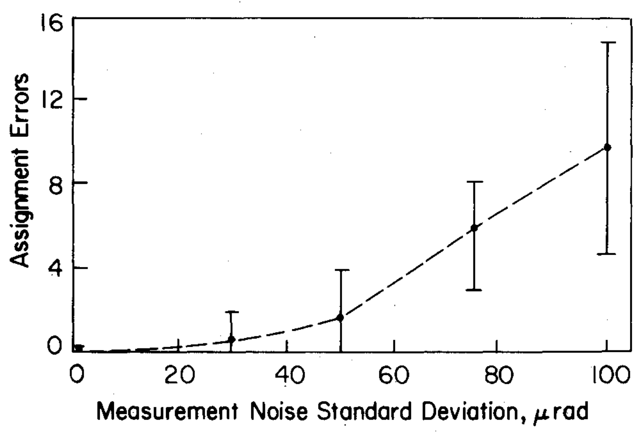


Fig. 4 Cycle-to-cycle assignment errors vs measurement noise.

quence of these position estimates can be used to initialize each track velocity, and if desired, the accelerations as well.

In principle, the point of closest approach procedure can be applied to a single pair of synchronized scans. As is to be expected from sensitivity considerations, a more robust assignment solution is obtained by simultaneously using a time sequence of measurements from each sensor. Once again, this assumes that the cycle-to-cycle association problem has been solved. The cost matrix D to be used is made up of squared errors summed over K cycles, that is,

$$d_{ij} = \left[\sum_{k=1}^{K} e_{ij}(t_k)^T e_{ij}(t_k) \right]^{\frac{1}{2}} \tag{25}$$

Table 3 Sensor-to-sensor assignment results (satellites 1 and 2)

Measurement noise, μrad	Number of cycles	Results (out of 15 targets) Correct	Pair flips	Other errors
10	6	13	1	0
	9	13	1	0
	15	15	0	0
15	6	13	1	0
	9	15	0	0
	15	15	0	0
30	6	11	2	0
	9	15	0	0
	15	15	0	0
50	6	11	2	0
	9	13	1	0
	15	13	1	0
75	6	11	2	0
	9	10	1	3
	15	13	1	0

Table 4 Sensor-to-sensor assignment results (satellites 1 and 3 and 2 and 3 with 15 data cycles)

Measurement noise, μrad	Satellite pair	Results (out of 15 targets) Correct	Pair flips	Other errors
10	1,3	13	1	0
	2,3	15	0	0
20	1,3	13	1	0
	2,3	11	2	0
30	1,3	12	0	3
	2,3	8	2	3

Some limited numerical work was done with d_{ij} defined without the square root shown in Eq. (25). The choice of cost matrix is somewhat subjective. It was found that at least the first- and second-best assignment pairing were the same under both definitions for the cases tested. However, the third and higher choices tended to differ somewhat. In retrospect, the squared version is probably preferable because it saves some square-root calculations, makes the more familiar chi-square theory applicable, and also seems to give a sharper demarcation between the first, second, etc. ranked assignment pairings. Nevertheless, the simulation studies reported next all used a D matrix defined by Eq. (25).

Numerical Results for the Sensor-to-Sensor Assignment Problem

The key parameters that affect performance are the viewing geometry, the number, density and motion of the targets, the number of measurement cycles used, the cycle time, the measurement accuracy, sensor location errors, and interpolation errors. The number, density, and motion of the targets are the same as in earlier examples, typified by Fig. 3. The duration of observations is now 15 cycles with a 5 s period. A third satellite is added to those of Fig. 2, so that three different pairs can be considered. Measurement noise levels up to 75 μrad are considered. No sensor location errors are considered. Interpolation errors introduced by the cubic data fitting are inherently in the results to follow, but no detailed study of these errors is included. It is known that these errors increase with interpolation time, and the maximum synchronization adjustment required here was about 2.5 s.

Tables 3 and 4 present assignment error results for different data spans, measurement accuracies, and satellite pairs. In both of these tables, the number of correct sensor-to-sensor assignments (out of 15) are listed, as well as the number of pair flips and other errors. A pair flip is a special type of error. It is characterized by having two objects interchanged in the first-choice assignment and correctly paired in the second-choice assignment. Usually, the cost differences between first and second choice are small, and a great deal of indecision is obvious in the algorithm results regarding this closely contested pair. Table 5 lists an example of the algorithm-assignment pairings and associated costs. Although 15 ranked sets of assignments are tabulated, typically only the first 3 to 6 can be guaranteed as being best, as discussed earlier. This particular example illustrates a pair flip. The best assignment is actually all correct except for rows 9 and 15. (Row numbers are the target numbers as attached by the first sensor.) Correct results for these two rows are (9,12) and (15,14), and these choices show up in the second-best pairings. Note also the 14, 12,14, . . . alternating pattern throughout all choices in these

Table 5 Example assignment matrix and best costs

Row i (sensor 1)	Assigned column j (target number on sensor 2)														
1	7	7	11	11	7	7	11	11	7	7	7	7	7	11	11
2	4	4	4	4	5	5	5	5	4	4	9	9	9	9	9
3	8	8	8	8	8	8	8	8	11	11	8	8	8	8	8
4	5	5	5	5	4	4	4	4	5	5	4	5	4	4	4
5	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
6	11	11	7	7	11	11	7	7	8	8	11	11	11	7	7
7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
8	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
9	14	12	14	12	14	12	14	12	14	12	14	14	12	14	12
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
12	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
13	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
14	9	9	9	9	9	9	9	9	9	9	5	4	5	5	5
15	12	14	12	14	12	14	12	14	12	14	12	12	14	12	14

$C_{15} = [16.54 \quad 16.70 \quad 16.97 \quad 17.13 \quad 17.13 \quad 17.30 \quad 17.56 \quad 17.73 \quad \ldots]$

Table 6 Monte Carlo sensor-to-sensor assignment results (50 trials with satellite pair 1,2)

Measurement noise, μrad	Number of scans	Number of occurrences out of 50		
		All correct	1 Pair flip	2 Pair flip
15	8 at 10 s	34	11	5
30	8 at 10 s	17	26	7
	15 at 5 s	18	25	7

two rows. This indecision on the part of the algorithm is typical of what is called a pair flip. Its occurrence is obvious even when the correct answers are not known. The associated costs are listed at the bottom of Table 5. Differences for pair flips are typically small. These scores are also useful in rejecting obviously bad assignments.

The major advantages of the algorithm presented here, as compared with the Munkres approach, lie in the ability to see the pattern of first, second, etc. assignments and their scores. In a case like Table 5, it may be wise to initiate an extra tentative track and let later data determine which one is actually most feasible. This opportunity does not exist with the Munkres approach.

The previous results are representative samples of random processes. Higher confidence can be placed in Table 6 results, which give 3 sets of 50 Monte Carlo repetitions within the major parameter range of interest. The 15 μrad cases resulted in all 15 targets being correctly paired together on 34 of the 50 trials. Eleven times out of 50 a single pair flip occurred and five times there were two pair flips. Similar interpretations apply to the other tabulated results. Notice that the triangulation process inherent in this scheme is improved by having as much diversity as possible in the viewing geometry. Thus, 8 cycles, each 10 s apart, and 15 cycles, each 5 s apart, both give an end-to-end time of 70 s of change. Assignment accuracies are essentially the same for these two cases and are better than would result from 8 cycles at a 5 s spacing. The ability to correctly assign targets is obviously a direct function of the measurement accuracy levels and viewing geometry as well.

Conclusions

An algorithm for solving the assignment problem has been presented. Not only the best assignment pairings, but several of the next best choices are determined, along with their costs. Sufficiency conditions for the optimality of the procedure have been established.

The assignment algorithm has been used as the central module in studying several interesting strategic defense initiative track-initiation and measurement-assignment problems. The key parameters affecting performance in these problems have been investigated and simulation results presented. If the uncertainty caused by measurement error is small compared with the true target separations as projected at the sensors, the assignment problem is an easy one. When these ratios increase due to noise or the apparent foreshortening due to viewing geometry, the problem becomes more difficult or even impossible to accomplish correctly. Results presented here show that reasonable noise levels and realistic geometries can be dealt with successfully. (Average true separation to a noise standard-deviation ratio of approximately 2, with minimum ratios near 1.) An important feature of this assignment algorithm is its ability to provide clear signals when ambiguous or close-call situations arise. In these cases, it may be prudent to initialize a small number of extra tracks. Also, obviously bad assignments can be rejected by simple thresholding of the assignment costs.

The computational burden of the approach has been studied in some detail. Although those details go beyond the scope of this paper, a review of the algorithm reveals a triple nesting of do loops. Thus, aside from certain logic and branching decisions, the computer burden goes predominantly as the third power of the number of targets being assigned. This is competitive with other approaches such as the Munkres algorithm.

References

[1]Munkres, J., "Algorithm for the Assignment and Transportation Problems," *Journal of the Society for Industrial and Applied Mathematics*, Vol. 5, March 1957, pp. 32–38.

[2]Silver, R., "An Algorithm for the Assignment Problem," *Communications of the ACM*, Vol. 3, Nov. 1960, pp. 605, 606.

[3]Bourgeouis, F. and Lassalle, J.-C., "An Extension of the Munkres Algorithm for the Assignment Problem to Rectangular Matrices," *Communications of the ACM*, Vol. 14, Dec. 1971, pp. 802–804.

[4]Stephens, P. A. and Krupa, N. R., "A Sparse Matrix Technique for the Munkres Algorithm," *Proceedings of the 1979 Summer Computer Simulation Conference*, American Federation of Information Processing Societies, Toronto, Canada, July 1979, pp. 44–47.

[5]Wong, J. K., "A New Implementation of an Algorithm for the Optimal Assignment Problem: An Improved Version of Munkres' Algorithm," *BIT* (Sweden), Vol. 19, No. 3, 1979, pp. 418–424.

[6]Morefield, C. L., "Application of 0–1 Integer Programming to a Track Assembly Problem," Aerospace Corp., El Segundo, CA, Rept. SAMSO-TR-75-186, April 1975.

[7]Morefield, C. L., "Application of 0–1 Integer Programming to Multitarget Tracking Problems," *IEEE Transactions on Automatic Control*, Vol. AC-22, No. 3, 1977, pp. 302–312.

[8]Taha, H., *Integer Programming: Theory, Applications and Computations*, Academic Press, New York, 1975.

[9]Shapiro, J. F., *Mathematical Programming*, Wiley, New York, 1979.

[10]Bhattacharyya, A., "On a Measure of Divergence Between Two Statistical Populations Defined by Their Probability Distributions," *Bulletin of Calcutta Mathematical Society*, Vol. 35, Sept. 1943, pp. 99–109.