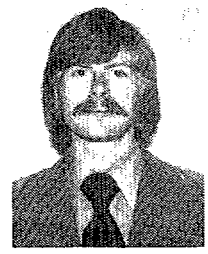
blur removal," in *Evaluation of Motor Degraded Images*, NASA Tech. Rep. SP-193, M. Nagel, Ed. Washington, DC: 1968, pp. 139–148.
[6] M. M. Sondhi, "Image restoration: The removal of spatially invariant degradations," *Proc. IEEE*, vol. 60, pp. 842–853, July 1972.
[7] T. S. Huang, W. F. Schreiber, and O. J. Tretiak, "Image processing," *Proc. IEEE*, vol. 59, pp. 1586–1609, Nov. 1971.
[8] A. A. Sawchuk, "Space-variant image motion degradation and restoration," *Proc. IEEE*, vol. 60, pp. 854–861, July 1972.
[9] ——, "Space-variant image restoration by coordinate transformations," *J. Opt. Soc. Amer.*, vol. 64, pp. 0124–0130, Feb. 1974.
[10] N. D. A. Mascarenhas and W. K. Pratt, "Digital image restoration under a regression model," *IEEE Trans. Circuits Syst.*, vol. CAS-22, pp. 252–266, Mar. 1975.
[11] L. M. Silverman, "Realization of linear dynamical systems," *IEEE Trans. Automat. Contr.*, vol. AC-16, pp. 554–567, Dec. 1971.
[12] A. O. Aboutalib, "Restoration of images degraded by motion," Ph.D. dissertation, Univ. Southern California, Los Angeles, June 1974.
[13] R. W. Brockett, "Poles, zeros and feedback: State space interpretation," *IEEE Trans. Automat. Contr.*, vol. AC-10, pp. 129–135, Apr. 1965.
[14] A. O. Aboutalib and L. M. Silverman, "Restoration of images degraded by curvilinear motion," presented at the 2nd Int. Joint Conf. Pattern Recognition, Copenhagen, Denmark, Aug. 1974.
[15] N. E. Nahi and C. Franco, "Recursive image enhancement vector processing," *IEEE Trans. Commun.*, vol. COM-21, pp. 305–311, Apr. 1973.
[16] H. L. Van Trees, *Detection, Estimation and Modulation Theory*. New York: Wiley, 1971, part II.

A. Omar Aboutalib (M'75) received the B.S. degree in electrical engineering from Ain Shams University, Cairo, Egypt, and the M.S. and Ph.D. degrees in electrical engineering (systems) from the University of Southern California, Los Angeles, in 1966, 1972, and 1974, respectively.

From June 1974 to August 1975, he was a Senior Engineer at Hoffman Electronics Company, El Monte, CA, working on system analysis and software design of avionic systems. From September 1975 to April 1976, he was a Senior Engineer with the Singer Company, Librascope Division, Glendale, CA, working in the target detection of underwater weapons systems. Since April 1976 he has been employed by the Hughes Aircraft Company, Ground Systems Group, Fullerton, CA, working in the development of underwater tracking systems. He also teaches courses in communications and control theory at California State Polytechnic University, Pomona and is involved in research activities at the Image Processing Institute at the University of Southern California. His interests encompass control theory, estimation, and digital image processing.

Dr. Aboutalib is a member of Eta Kappa Nu.



Michael S. Murphy (S'72–M'76) was born in New Orleans, LA, in 1951. He received the B.S.E.E. degree from Louisiana State University, Baton Rouge, and the M.S.E.E. degree from the University of Southern California, Los Angeles, in 1973 and 1975, respectively. He is currently completing degree requirements for the Ph.D. in electrical engineering in the area of control systems at the University of Southern California.

While engaged in graduate studies, he worked as a Teaching Assistant from 1973 to 1975, a Lecturer in 1976, and a Research Assistant from 1973 to 1977, all in the Department of Electrical Engineering–Systems, University of Southern California, Los Angeles.

Mr. Murphy is a member of Sigma Xi, Tau Beta Pi, Eta Kappa Nu, and Phi Kappa Phi.

L. M. Silverman (S'60–M'66–SM'77), for a photograph and biography see page 35 of the February 1977 issue of this TRANSACTIONS.

Application of 0-1 Integer Programming to Multitarget Tracking Problems

CHARLES L. MOREFIELD, MEMBER, IEEE

Abstract—**This paper presents a new approach to the solution of multitarget tracking problems. 0-1 integer programming methods are used to alleviate the combinatorial computing difficulties that accompany any but the smallest of such problems. Multitarget tracking is approached as an unsupervised pattern recognition problem. A multiple-hypothesis test is performed to determine which particular combination of the many feasible tracks is most likely to represent actual targets. This multiple hypothesis test is shown to have the computational structure of the set packing and set partitioning problems of 0-1 integer programming. Multitarget tracking problems that are translated into this form can be rapidly solved, using well-known discrete optimization techniques such as implicit enumeration.**

Manuscript received April 21, 1975; revised November 14, 1975 and June 22, 1976. Paper recommended by E. J. Davison, Past Chairman of the IEEE S-CS Computational Methods and Discrete Systems Committee. This work was accomplished in part while the author was on the staff of the Aerospace Corporation, El Segundo, CA and was supported in part by the Office of Naval Research under Contract N00014-77-C-0296.

The author is with the ORINCON Corporation, La Jolla, CA 92037.

I. INTRODUCTION

THIS paper describes a new algorithm for multitarget tracking based on the use of 0-1 integer programming [1]. The multitarget tracking problem [2]–[12] may be briefly stated as follows: given a large number of (spa-

tially) close measurements, determine trajectory estimates for any targets that may be present. Since it is difficult to determine precisely which target (if any) corresponds to each of the closely-spaced measurements, some targets may go undetected, while others may have inaccurate trajectories attributed to them. For example, an air traffic controller at a busy airport may incorrectly decide that a new return on his radar display corresponds to an aircraft already being tracked, rather than correctly recognizing the appearance of a new aircraft. As another example, a sonar operator may decide, hearing a number of echoes from a given sector, that several ships are present, but be unable to accurately separate the echoes into individual ships' tracks. Although trajectory estimation problems have been well studied in the past, much of this previous work assumes that the particular target corresponding to each observation is known. In multitarget situations, the correspondence between targets and observations may be lost for any number of reasons: crossing trajectories, poor sensor resolution, poor viewing geometry, large number of false alarms, etc. *Ad hoc* attempts to apply standard estimation algorithms under these circumstances generally meet with disappointment.

We approach multitarget tracking as an unsupervised pattern recognition problem, in which it is necessary to estimate from unclassified data [13] both the number of tracks (data clusters) present and the parameters of individual trajectories. Although trajectory and noise models are assumed to be known, it is not known which target is being observed by the sensor (or sensors) as each measurement is obtained.

The integer program described in the paper is essentially a decision-directed pattern recognition algorithm [13] that classifies measurements on the basis of Bayesian decision theory, arranging them into the set of multiple tracks that best fit the given trajectory models. Although decision-directed multitarget tracking was mentioned as early as 1964 [11], it has not been pursued in the literature because of the large amount of computation required. To appreciate the computational difficulties that are involved, consider what happens when the pieces of several jigsaw puzzles are placed in a common pile and randomly mixed together. To solve all of the puzzles, we are obliged to build up many alternative combinations of pieces, deciding at each step of the way whether to continue the current partial solution, or to backtrack and follow a more promising path. In the multitarget tracking problem, the pieces of the puzzles are the measurements, and the puzzles themselves are the unknown tracks of individual targets. The algorithm described in this paper makes it possible to sift through various combinations of data until a globally optimal picture is formed of the surveillance area. The measure of optimality for the algorithm is how well the selected tracks fit the trajectory models that are given.

It is generally recognized [9], [11], [13] that decision-directed classification results in large combinatorial optimization problems. Since n measurements can be partitioned into m individual tracks in as many as $1/m!\sum_{i=1}^{m}\binom{m}{i}(-1)^{m-i}i^n$ different ways [13], even a moderate-sized data base presents a substantial problem. The main concern of much of the pattern recognition and multitarget tracking literature is with avoiding such combinatorial difficulties. Either suboptimal partitions are accepted or the partitioning process is avoided entirely. ISODATA [13] is a well-known example of a decision-directed pattern recognition algorithm that avoids exhaustive enumeration at the expense of global optimality. In the multitarget tracking literature, the recent papers by Singer *et al.* [9]–[10] are devoted primarily to the error analysis of certain locally optimal tracking filters. Rather than partition data into individual tracks, the multitarget tracking algorithms derived by Alspach [2] and Bar-Shalom [3] form trajectory estimates using data from several contiguous tracks.

In contrast, this paper directly faces the combinatorial problems that plague multitarget tracking, and provides for the first time an algorithm based on Bayesian decision theory that is useful for large data bases. Section II states the multitarget tracking problem as an m-hypothesis decision problem and Section III illustrates how feasible tracks (potential data clusters) are constructed for the various hypotheses. The unique part of the paper is Section IV, where it is shown that the multitarget tracking problem belongs to a class of well studied discrete optimization problems: the set partitioning and set packing problems of 0-1 integer programming [14]. The discussion of solution techniques in Section V assumes the use of a conventional computer, although quick processing of extremely large data bases may require the use of parallel processors [7]. Section VI discusses a numerical example; concluding remarks are made in Section VII.

II. Decision-Directed Approach to Multitarget Tracking

In a decision-directed approach to multitarget tracking, a multiple hypothesis test is performed, and individual trajectory estimates are made on the basis of a hypothetically correct partition of the data into subsets due to single targets. Most of the work in estimation theory pertains to situations where *all* the observations $Z=\{z_1,z_2,\cdots,z_n\}$ are due to a single target. An obvious example is the stochastic linear system

$$x_{k+1}=Ax_k+Bu_k, \qquad k=0,1,\cdots,n \tag{1}$$

$$z_k=Cx_k+w_k, \qquad k=1,\cdots,n \tag{2}$$

with states $\{x_k\}\subset R^x$, observations $\{z_k\}\subset R^z$, process noise $\{u_k\}\subset R^u$, and measurement noise $\{w_k\}\subset R^w$. A,B,C are matrices of appropriate dimension that may vary with time. The initial state x_0 is a Gaussian random vector with covariance P_0, independent of the processes $\{u_k\}$ and $\{w_k\}$, which are themselves zero mean white Gaussian noise with covariances $\{Q_k\}$ and $\{R_k\}$. Under these assumptions, the well-known Kalman equations [15]

provide minimum variance unbiased estimates $\{\hat{x}_k\}$ of the states based on all past data. Implicit in the often used model (1),(2) is the assumption that each measurement $z_k \in Z$ is a sample drawn from a single conditional probability density $p(z_k|x_k)$.

Suppose now that s targets are present, and each target is represented by a separate stochastic process identical in overall form to the one described above:

$$x_{k+1}^i = A^i x_k + B^i u_k^i, \qquad k=0,1,\cdots,n, \quad i=1,2,\cdots,s. \tag{3}$$

The subscript indices k denote sample times as before, while the superscripts i index the various targets (ships, aircraft, etc.). Denoting the number of measurements at the kth sample by $m(k)$, a set of $\sum_{k=1}^{n} m(k)$ measurements $Z=\{z_k^j, j=1,2,\cdots,m(k)\}_{k=1}^{n}$ is available. The indices j do *not* correspond to the target indices i, but are the indices of randomly reordered data. This notation includes the possibility that some measurements may be false alarms, and that the detection probability for target observations may be less than 1. For simplicity we assume only one observation equation is necessary so that target x_k^i is related to measurement z_k^j by the equation

$$z_k^j = Cx_k^i + w_k^j. \tag{4}$$

Thus, in the s-target case, each individual measurement z_k^j of an actual target is drawn from a mixture density of the form

$$p(z_k|x_k^1, x_k^2, \cdots, x_k^s) = \sum_{i=1}^{s} P_i p(z_k|x_k^i) \tag{5}$$

with unknown priors denoted P_i. It is the essence of multitarget tracking problems that there is a large amount of overlap among the component densities of (5) so that the target-to-measurement correspondence $x_k^i \leftrightarrow z_k^j$ is difficult to obtain. A number of factors may contribute to this difficulty, including closely-spaced targets, poor viewing geometry, high intensity sensor noise, inadequate sensor sample rate, a large number of false alarms, etc.

A two-dimensional example of such a situation is illustrated by the following figures. Fig. 1 is the data Z (e.g., radar ranges and bearings) at the sample times $k=1,2,3$. The figure also represents the null hypothesis that no targets are present.

Momentarily assuming that (except for random disturbances) targets follow straight lines in the measurement space, the lines drawn in Fig. 2 illustrate the correspondence $x_k^i \leftrightarrow z_k^j$ assumed in one reasonable partition of Z. Fig. 2 represents the hypothesis that two tracks and three false alarms are present.

More generally, the set of all measurements Z can be partitioned into m tracks $\lambda^i \subset Z$

$$Z = \lambda^1 \cup \lambda^2 \cup \cdots \cup \lambda^m \tag{6}$$

where

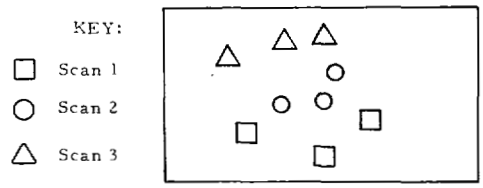


Fig. 1. The null hypothesis.

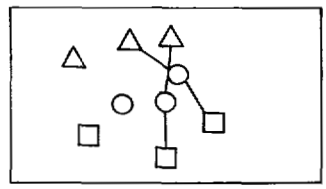


Fig. 2. A track formation hypothesis.

$$\lambda^i \cap \lambda^j = 0, \qquad i \neq j. \tag{7}$$

One of the tracks (say λ^m) may consist entirely of false alarms, as in Fig. 2. With this notation, any hypothesis τ concerning measurements made by the surveillance system (the number of targets present, which data points belong to which target) can be defined as a family of subsets $\lambda^j \subset Z$. It should be noted that the number of targets may vary from hypothesis to hypothesis. For example, if τ^1 is Fig. 1, τ^2 Fig. 2, and τ^3 Fig. 3, then each of the hypotheses $\{\tau^i\}_{i=1}^{3}$ has a different number of tracks.

Throughout the paper it will be assumed that $\tau=\{\lambda^j\}$ represents an actual partition, i.e., that (6) and (7) are satisfied. Thus, the situation shown in Fig. 4 is not a permissible hypothesis if it represents two separate tracks λ^1 and λ^2, since $\lambda^1 \cap \lambda^2 \neq 0$. However, it may be correct in cases of poor sensor resolution to permit some data points to be shared by more than one track. For example, the measurement $z=\lambda^1 \cap \lambda^2$ may consist of two sensor returns combined into one by poor sensor resolution. This can be trivially accounted for by replacing λ^1 by $\lambda^1 \cup \lambda^2$ and eliminating λ^2 as a separate track, so that the resulting hypothesis consists of just one track.

The defining characteristic of multitarget tracking problems is that a number of partitions can be found, each one composed of tracks λ^j that appear feasible from the standpoint of one or more of the models (3), (4). $S=\{\tau^i\}_{i=1}^{\omega}$ will denote the set of all such partitions. The posterior density for any partition τ is given by $p(\tau|Z)=p(Z|\tau)P(\tau)/p(Z)$. Since S is a finite set, $p(\tau|Z)$ can be regarded as a discrete posterior distribution and we can interchangeably write $p(\tau|Z)$ or $P(\tau|Z)$. An assumption fundamental to the derivation of linear integer programs is that the data in each track λ is independent from that in other tracks,

$$p(Z|\tau) = \prod_{j=1}^{m} p(\lambda^j|\tau). \tag{8}$$

This assumption is the motivation behind (7) and is used throughout the paper.

In multitarget tracking problems the mode of $p(\tau|Z)$ is not immediately obvious, i.e., a number of alternative

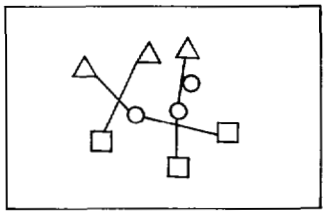


Fig. 3. Alternative track formation hypothesis.

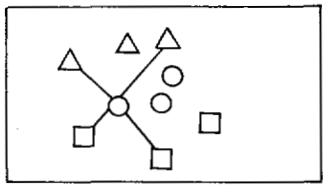


Fig. 4. Unallowed hypothesis.

partitions τ^i appear reasonable. If we regard Z as a sample from the mixture density $p(Z)=\Sigma_{\tau\in S}p(Z|\tau)P(\tau)$, a number of the component densities $p(Z|\tau)$ are significant. This suggests the definition of the multitarget tracking problem adopted in this paper, the discrete optimization problem

$$\max_{\tau\in S}\prod_{j=1}^{m}p(\lambda^j|\tau)P(\tau) \tag{9}$$

where the normalizing constant $p(Z)$ is ignored as usual, and ties are broken arbitrarily. It defines multitarget tracking as a Bayesian decision process that minimizes the Bayes risk incurred in choosing a partition of Z. The solution $\tau^*\in S$ is chosen for every data set Z satisfying

$$Z\in\left\{Z|P(\tau^*|Z)\geqslant P(\tau^j|Z)\ \forall\tau^j\neq\tau^*\right\}.$$

We will follow conventional methodology, replacing the actual posterior densities with approximate densities based upon the trajectory estimates for $\{\lambda^j\}$.

Throughout the paper we will deal with the general tracking problem, in which the data Z is collected at several sample times and the number of targets is unknown. It should be noted, however, that (9) can be restricted to the case where Z is the data from one sampling time and the number of targets is known. This leads to recursive estimation of the mode of $p(\tau|Z)$ [8], or the mean of $p(Z)$, using some or all of the component densities $p(Z|\tau)$ [2], [3].

The main point of this paper is to demonstrate that problem (9) can be converted to a 0-1 integer program, thereby alleviating the computational difficulties that would result from a brute force enumeration of every possible partition of Z.

III. Feasible Track Construction

In general, a hypothesis $\tau\in S$ consists of both false alarms and tracks that appear feasible from the standpoint of the trajectory model. For example, if these models satisfy the assumptions of the Kalman equations a real track can be detected on the basis of the known probability density function of the innovations sequence [16]. It is inefficient to first form a hypothesis τ and then check the feasibility of its tracks, since the same track $\lambda^j\in\tau$ may appear in another hypothesis (the same innovations sequence applies in both cases). An example of this is given in Figs. 2 and 3, which have one track in common. This observation, together with the independence of individual tracks (8), leads to an algorithm that consist of two parts: feasible track construction, followed by solution of (9) as shown in Fig. 5. *Feasible track construction* is a clustering procedure that detects the tracks in Z that are reasonable to incorporate in hypotheses and stores them in a feasible track set F. The subsequent Bayesian decision process is then restricted to hypotheses formed using the tracks in F, so any hypothesis τ is just a subset $\{\lambda^j\}_{i=1}^m$ of F.

The set of false alarms (say λ^m) is defined by the relation

$$\lambda^m=Z-\bigcup_{j=1}^{m-1}\lambda^j. \tag{10}$$

Each track $\lambda\in F$ is the result of a hypothesis test that uses the track likelihood function $p(\lambda)$ determined from the trajectory models. Since the density function of the alternative hypothesis (that λ is not a track) is unknown, the decision rule is simply

$$\begin{aligned}&\ln p\left(\lambda|\{\hat{x}_k\}_{k=1}^n\right)\geqslant\alpha_n\Rightarrow\lambda\in F\\&\ln p\left(\lambda|\{\hat{x}_k\}_{k=1}^n\right)<\alpha_n\Rightarrow\lambda\notin F.\end{aligned} \tag{11}$$

Based on the log likelihood decision function, the feasible track set is

$$F=\left\{\lambda|\lambda\subset Z,\ \ln p\left(\lambda|\{\hat{x}_k\}_{k=1}^n\right)\geqslant\alpha_n\right\}.$$

Since our ultimate objective is to determine a Bayesian decision function (9) that results in a linear integer program, the independence of tracks (8) is a critical assumption. However, the precise form of $p(\lambda)$ is not crucial. For example, in nonlinear problems $p(\lambda)$ might be an accurate approximation [17], [18] to an arbitrary density function.

For trajectory models (3) and (4), it is natural to compute the likelihood function based on the Kalman filter state estimates, basing the hypothesis test (11) on the innovations sequence [16]

$$\delta_k=z_k-CA\hat{x}_{k-1},\qquad k=1,2,\cdots,n. \tag{12}$$

The (negative) log-likelihood function is given in this case by

$$c(\lambda)=\frac{1}{2}n\dim(z)\ln 2\pi+\frac{1}{2}\sum_{k=1}^{n}\ln|V_k|+\frac{1}{2}\sum_{k=1}^{n}\delta_k^tV_k^{-1}\delta_k \tag{13}$$

where V_k is the covariance of δ_k and $\dim(z)$ is the dimension of z. Since V_k can be computed off line, the only random component of $c(\lambda)$ is $\Sigma\delta^tV^{-1}\delta$, which for real

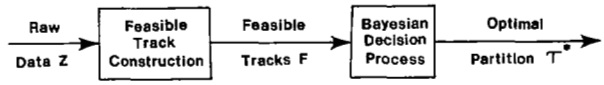


Fig. 5. Overview of the complete multitarget tracking algorithm.

tracks is a chi-squared random variable with $n \cdot \dim(z)$ degress of freedom. Therefore, error probabilities can easily be computed for the hypothesis test (11) to predict the accuracy of feasible track construction. This has a critical impact on the ultimate accuracy of the tracking algorithm, since a real track mistakenly excluded from F cannot be used in the subsequent Bayesian decision process (9).

When sequential trajectory models are used, feasible track construction is best accomplished using a backtracking algorithm [19] of some type. Before any new point is added to a partial track, it must pass a coarse test [9],

$$\|\delta_k\|_\infty \leq \beta_k, \tag{14}$$

a fine test [3], [9],

$$\delta_k' V_k^{-1} \delta_k \leq \gamma_k, \tag{15}$$

and finally the likelihood test (11) [12]. Test (14) checks the magnitude of the maximum component of the vector $\delta_k \in R^z$ against β_k, and is included because it is computationally cheaper to perform than (15). The constants β_k, γ_k can be chosen so that

$$\{z_k | \delta_k' V_k^{-1} \delta_k \leq \gamma_k\} \subset \{z_k | \|\delta_k\|_\infty \leq \beta_k\}. \tag{16}$$

If a depth-first backtracking algorithm [19] is used, as many points as possible are added to a potential track before backtracking occurs. When backtracking occurs at point z_{k+1} because (14), (15), or (11) is violated, only the last state $\hat{x}_k$ must be recomputed before a new branch of the search tree is examined.

After every subset of Z has been examined, the r feasible tracks $F = \{\lambda^j\}_{j=1}^r$ are retained, together with the r-dimensional vector c defined by

$$c_j = c(\lambda^j), \qquad j = 1, 2, \cdots, r. \tag{17}$$

The vector c and set F are used to define the integer program discussed in Section IV.

The above discussion of feasible track formation has suggested rather simple ways of forming F. Operational tracking algorithms are typically based on a number of *ad hoc* ideas about feasible track motion, particularly in cases where measurement or trajectory nonlinearities are present so that convenient models such as (3) and (4) are no longer adequate. The integer program discussed below could be used in these cases also, so long as the vector c and set F are based on reasonable clustering criteria.

Because of the way F is constructed, some tracks may differ from others only by the interchange of a few data points, so that a number of similar trajectory estimates and scores $\{c(\lambda^j)\}$ may be computed. This, together with the potentially large size of F, has led to algorithms that either directly eliminate or average similar tracks [12]. This approach is not taken in this paper since it cannot be determined on an individual basis precisely how tracks should be eliminated. Instead, we use a Bayesian decision process to select tracks on a global basis, and the computational efficiency of 0-1 integer programming is relied upon to alleviate the combinatorial difficulties this approach presents.

IV. An Integer Program for Multitarget Tracking

The Bayesian problem described above can be recast in a more useful computational form. By translating the decision process (9) to a 0-1 integer program [1], efficient computer algorithms are made available for large scale tracking problems. In particular, we will find that multitarget tracking problems are examples of the set partitioning and set packing problems of integer programming [1], [14].

Recall that any hypothesis $\tau \in S$ divides the data Z into tracks $\{\lambda^i\} \subset F$, one of which may consist just of false alarms. The assumed structure of the algorithm is such that every feasible track λ^i is constructed prior to entering the decision logic (Fig. 5). This allows us to put τ in a useful mathematical form.

Suppose there are r tracks in F, and the family of subsets $\tau = \{\lambda^i\} \subset F$ is reinterpreted as an r-dimensional vector of 0's and 1's. If τ_i, the ith component of τ, corresponds to the ith track in F, then any subset of F can be specified by simply setting the appropriate components of τ to 1, while the other components remain 0. To illustrate this, assume that the cardinality of F is 5. Then the τ-vector corresponding to the subset $\{\lambda^1, \lambda^3, \lambda^5\} \subset F$ is

$$\tau = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

The τ-vector defines a complete partitioning of Z, since the set of false alarms λ^m is given by (10).

A. Linear Decision Functions

The reader can get a preliminary idea of our objectives by looking ahead to (29)–(31), which define a discrete linear optimization problem equivalent to (9). Reduction of (9) to this set packing problem [14] will require that each hypothesis be evaluated using a linear functional

$$c'\tau, \tag{18}$$

where the ith component of c is the contribution of track λ^i to $p(Z|\tau)P(\tau)$. This is easily done when no false alarms are present and $P(\tau) = \text{constant}$, since in this case the definition of the r-dimensional vector c given in (13) makes the problem

$$\min_{\tau \in S} c'\tau \tag{19}$$

equivalent to (9).

If false alarms are present, and are equally likely anywhere in the sensor field of view, the density function for the subset of false alarms λ^m is

$$p(\lambda^m|\tau) = \left(\frac{1}{Y}\right)^f \tag{20}$$

where f is the number of points included in λ^m and Y is the size of the surveillance area. If the (negative) likelihoods of individual tracks are as given in (13), then to include false alarms we must have $c'\tau$ equal to [11]

$$\sum_{j=1}^{m-1}\left(\frac{1}{2}n_j \dim(z)\ln 2\pi + \frac{1}{2}\sum_{k=1}^{n_j}\ln|V_k| + \frac{1}{2}\sum_{k=1}^{n_j}\delta_k' V_k^{-1}\delta_k\right) + f\ln Y \tag{21}$$

where the first summation extends over the $m-1$ feasible tracks, and n_j is the number of points in track λ^j. For simplicity we have ignored the obvious dependence of δ_k and V_k on the track index j. If we redefine c_j as

$$c_j = \frac{1}{2}n_j \dim(z)\ln 2\pi + \frac{1}{2}\sum_{k=1}^{n_j}\ln|V_k| + \frac{1}{2}\sum_{k=1}^{n_j}\delta_k V_k^{-1}\delta_k - n_j \ln Y, \tag{22}$$

then $c'\tau$ differs from (21) only by the constant term $\sum_{k=1}^{n} m(k)\ln Y$, which together with $P(\tau)$ can be disregarded. Minimizing $c'\tau$ is then equivalent to minimizing $-\ln p(Z|\tau)$.

Although the above algebra illustrates one decision formula that can be written as a linear function of the binary vector τ, operational tracking systems generally involve more complex data structures than $P(\tau) =$ constant. Often the prior distribution $P(\tau)$ is given as a product of *a priori* information terms such as [3], [9], [11]

$$P(\tau) = \exp(-\mu Y)(\mu Y)^f / f! \tag{23}$$

where μ is the expected number of false alarms in Z per unit of surveillance area. Models of track length may also be available [11], for example, the exponential model

$$P(\tau) = \prod_{j=1}^{m-1}\exp(-n_j/L) \tag{24}$$

where L is the expected track length. Incorporating such terms in c is generally a tedious, straightforward process we will not pursue, except to note that the decomposition

$$\ln P(\tau) \sim \sum_{i=1}^{m-1} g_i(\lambda^i)$$

must hold for some functions g_i if we are to use the linear decision function (18).

B. Linear Constraints

Any hypothesis τ belongs to the finite set S. To reduce (9) to a completely linear problem, we will define a matrix A such that

$$S = \{\tau \text{ binary} \,|\, A\tau \leqslant \mathbf{1}\} \tag{25}$$

where "**1**" is a vector composed entirely of ones. In order to do this, we first develop a binary representation of the tracks $\lambda^i \subset Z$ that is analogous to the one for τ. Recall that Z is composed of $\sum_{k=1}^{n} m(k)$ measurements. If λ is a $\sum_{k=1}^{n} m(k)$-dimensional vector of zeros and ones, whose ith component corresponds to the ith element of Z, then any subset of Z can be represented as a binary vector with the appropriate components set to 1.

The constraints that define S have obvious parallels for the binary track vectors corresponding to any hypothesis τ. Since the separate tracks λ^i and λ^j in a partition $\tau \in S$ must satisfy (7), we have the equivalent binary constraint

$$\lambda^i + \lambda^j \leqslant \mathbf{1}, \qquad i \neq j \tag{26}$$

where "**1**" is defined as before as vector of 1's.

To form a complete set of such inequalities, define a matrix A with columns taken from the feasible track set F:

$$A = (\lambda^1, \lambda^2, \cdots, \lambda^r), \tag{27}$$

so that (26) becomes

$$A\tau \leqslant \mathbf{1}. \tag{28}$$

The binary matrix A has row dimension $\sum_{k=1}^{n} m(k)$ and column dimension r. Substitution of (27) into (25) completely specifies S.

C. 0-1 Integer Programs

Combining the above results yields the complete multitarget tracking problem

$$\text{minimize } c'\tau \tag{29}$$

$$\text{subject to } A\tau \leqslant \mathbf{1} \tag{30}$$

$$\tau \text{ binary.} \tag{31}$$

Problem (29)–(31) is by definition a set packing problem [14], which as we noted previously allows the possibility that false alarms are contained in the data Z. From a computational standpoint it is a much more meaningful version of the multitarget tracking problem (9), since it represents a problem structure often used in combinatorial optimization.

If no false alarms are present (a rather unrealistic assumption for many problems), the constraints (30) can be tightened to

$$A\tau = \mathbf{1} \tag{32}$$

resulting in a set partitioning problem [1]. This is a rather

tricky version of the problem in many applications, since it requires every measurement in Z to be explicitly assigned to a track. If F is not large enough, it can happen that (32) has no solution, so that more tracks would have to be added to F by loosening the feasible track tests, followed by another attempt to solve (29), (32), (31).

Although $\tau=0$ (the null hypothesis that no tracks are present) is a feasible solution of the constraints (30), it is not optimal when there exists a component of c satisfying

$$c_j<0. \tag{33}$$

Relations such as these can be used to define the constraints for feasible track construction. For example, if c_j is defined as in (22), this results in the constraint

$$\sum_{k=1}^{n_j}\delta_k^t V_k^{-1}\delta_k<2n_j\left(\ln Y-\frac{1}{2}\dim(z)\ln 2\pi\right)-\sum_{k=1}^{n_j}\ln|V_k|. \tag{34}$$

This constraint could be used by itself, or to guide the selection of γ_k in (15) according to

$$\sum_{k=1}^{n_j}\gamma_k=2n_j\left(\ln Y-\frac{1}{2}\dim(z)\ln 2\pi\right)-\sum_{k=1}^{n_j}\ln|V_k| \tag{35}$$

which gives a recursive definition of $\{\gamma_k\}_{k=1}^{n_j}$, starting with $\gamma_1=2\ln Y-\dim(z)\ln 2\pi-\ln|V_1|$.

Use of constraints that are defined in this manner is justified by the observation that any track λ^j with associated score $c_j>0$ would never be selected as part of the solution of (29)–(31). Since it will never be chosen by the integer program, there is no reason to incorporate it in F.

All of the above results carry over to sequential decision processes. The partitioning of Z (which now consists just of data from one sample time k) is slightly more detailed, however. Assume that there are currently m tracks on file, and the problem is simply to update existing tracks, start new tracks, and properly classify false alarms. Z is partitioned according to

$$Z=\lambda^1\cup\lambda^2\cup\cdots\cup\lambda^m\cup I\cup R$$

where λ^i is the data point added to track i, I are the data points which initiate new tracks, and R is the subset of false alarms. Any of these sets may be empty.

The matrix A is again constructed by forming a set of binary vectors λ, where each feasible update of every current track is recorded as a vector λ. These vectors have only one nonzero element.

A binary vector is also accorded each potential track initiator. In this case, however, "look-back points" may be included in the vector. This occurs in many operational systems, where track initiation is deferred until a data pattern characterizing a potential track is observed. Thus, data from at least one previous sample time $(k-1)$ is necessary for track initiation to occur on scan k, and the associated λ-vector will have at least two nonzero elements. To be completely consistent with the constraints, any data used on previous scans cannot be incorporated in a current track.

The resulting problem structure is precisely that of (29)–(31). Note that if we set $I=0$, then the solutions of (30) and (31) correspond to the solutions of Bar-Shalom's validation matrix [3]. However, we do not find every solution of the constraints, merely one which maximizes $p(Z|\tau)P(\tau)$. This algorithm (previously mentioned by Sea [8]) finds the mode of $p(\tau|Z)$, while Bar-Shalom's algorithm finds the mean of $p(x_k^i|Z)$, $i=1,2,\cdots,s$.

V. Solution Techniques

We must be somewhat sophisticated in our approach to solving the integer programs developed above when large amounts of data are involved, since a brute force enumeration will result in an algorithm whose run time increases exponentially with problem size. For example, if there are 1000 feasible tracks, then brute force enumeration would require the formation of 2^{1000} τ-vectors. A better (but still inefficient) approach would be to find every solution of (30) or (32), i.e., explicitly construct S. Each element of S would then be scored and the best value of $c^t\tau$ would determine a solution.

In solving (9), however, we are able to use a combination of cost function and constraint information to *implicitly* enumerate large numbers of feasible τ-vectors, thereby significantly speeding the solution of the problem. Although several algorithms are available for this type of problem [1], [14], [22], the numerical results presented in Section VI are primarily based on Pierce's version of the implicit enumeration method [20],[21].

We will give only a brief description of the algorithm, which in overview proceeds by constructing binary solutions of a sequence of inequalities

$$c^t\tau^i<c^t\tau^{i-1} \tag{36}$$

$$A\tau^i\leqslant\mathbf{1} \tag{37}$$

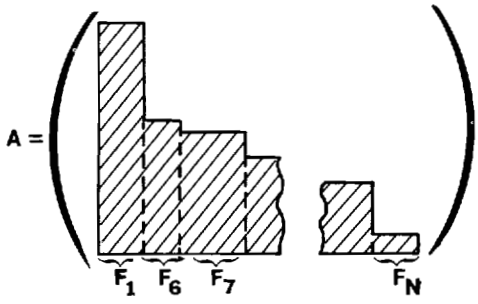
where τ^i is to be determined and τ^{i-1} is a last best solution ($\tau^0=0$). If (36) and (37) cannot be solved, then τ^{i-1} is an optimal feasible solution.

We assume that any possible decompositions of the problem have been made (see Section VI), so that the problem has minimal size. "A" is usually a sparse matrix. For example, if there are 50 targets visible to the sensor, and all track lengths are similar, then one might expect about two percent nonzero elements in A. Accordingly, it is useful to "pack" the data in binary form, one bit of a computer word corresponding to one element of A. Such an approach is useful also for checking constraint violations in the course of the computations. Thus (26) is equivalent to

$$\lambda^i\cdot\text{AND}\cdot\lambda^j=0, \qquad i\neq j \tag{38}$$

in Fortran, where λ^i and λ^j are binary words whose bit length is at least $\sum_{k=1}^{n}m(k)$.

Preprocessing can markedly speed any algorithm. The matrix A is rearranged so that it has the following form [20]:



(39)

A is partitioned into subsets of columns $F_1, \cdots, F_N$, where the first nonzero element of $\lambda \in F_i$ is λ_i. Note that many of the subsets F_i may be empty. It is important that the subsets be ordered spatially across the sensor field of view, the reason for this being that we proceed by selecting at most one track from each subset F_i. As in solving a jigsaw puzzle, we want to proceed "across the picture," always choosing a next piece of the puzzle which is spatially close to our current partial solution. This ordering is easy to accomplish in (39) when we note that each row of A corresponds to one measurement from Z. The vector $z \in Z$ may be composed for example of an elevation, azimuth, and range measurement. Selecting one component (say elevation), we simply permute the rows and columns of A as required to get form (39), where the data defining the partitions of F_i are monotonic in elevation. As a final item, the tracks within each subset F_i are ordered in terms of increasing scores.

An initial upper bound J_{UB} is computed for $c'\tau$ based on the measurement model. For example, if $c'\tau$ is chi-squared distributed, $P(c'\tau \leqslant J_{UB})$ is easily evaluated, and we can select a tight upperbound for any desired probability level.

The algorithm selects one element from each set F_i in turn, until (1) any of several necessary conditions are violated (we say that our partial solution is "fathomed"), or until (2) an element has been selected from each set F_i (a better feasible solution is found). In case (1) backtracking immediately occurs to the previous set F_{i-1}, where a new element is chosen and the algorithm proceeds as before. In case (2), precisely the same action is taken after first setting $J_{UB} = c'\tau^*$, where τ^* is the solution just obtained.

The numerical success of the algorithm depends upon the tightness of the necessary conditions mentioned above. Before we can discuss these, a few remarks must be made. After an element has been chosen from F_{i-1}, a binary subproblem remains of the form

$$\text{minimize} \sum_{j=s}^{r} c_j \tau_j \tag{40}$$

$$\sum_{j=s}^{r} \lambda^j \tau_j = \mathbf{1} - \sum_{j=1}^{s-1} \lambda^j \tau_j \tag{41}$$

where s is the index of the first feasible track in subset F_i. Define

$$\varphi_i = \min \{ c_j | \lambda^j \in F_i \} \tag{42}$$

$$\sigma_i = \sum_{j=i}^{N} \varphi_j \tag{43}$$

and

$$\bar{\lambda}^a = \lambda^a \oplus \lambda^{a+1} \oplus \cdots \oplus \lambda^r \tag{44}$$

where "$\oplus$" denotes the logical "or" operator and "a" is the index of the first feasible track in set F_{i+1}.

The necessary conditions to be satisfied by a selection λ^q from F_i include the following (constraint (46) can only be used with set partitioning problems):

$$\lambda^q \leqslant \mathbf{1} - \sum_{j=1}^{s-1} \lambda^j \tau_j \tag{45}$$

$$\lambda^q \geqslant \mathbf{1} - \left(\sum_{j=1}^{s-1} \lambda^j \tau_j \right) \oplus \bar{\lambda}^a \tag{46}$$

and

$$c_q \leqslant J_{UB} - \sum_{j=1}^{s-1} c_j \tau_j - \sigma_{i+1}. \tag{47}$$

Equation (45) states that track λ^q cannot contain any sensor return previously selected in the current partial solution. Equation (46) specifies that we must be able to select a (possibly infeasible) completion of the current partial solution which uses any currently unassigned data points. Equation (47) states that the best possible completion of the current partial solution cannot exceed J_{UB}. Note that the rhs of (47) is the exact solution of a knapsack subproblem due to Pierce [20].

There are several other details of the algorithm which we will not discuss. Our intent in this section is simply to illustrate the intuitively natural way that various partitions are examined using an implicit enumeration algorithm. One nice feature of the set packing problem should be noted, however, even partial solutions are feasible, so that if time constraints require stopping the computations, a feasible suboptimal solution is available.

VI. Numerical Example

The numerical example we have chosen is a moderately dense simulation with $s = 20$ targets. The actual track motion is described by quadratic polynomials, with random noise affecting only the measurements

$$x_{k+1}^i = \text{diag}(1) x_k^i, \qquad k = 0, 1, \cdots, 4, \quad i = 1, 2, \cdots, 20 \tag{48}$$

where diag(1) is a unit matrix of appropriate size, and

$$z_k^j = \begin{bmatrix} 1 & 10(k-1) & 100(k-1)^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 10(k-1) & 100(k-1)^2 \end{bmatrix} x_k^i + w_k^j, \qquad k = 1, 2, 3, 4 \tag{49}$$

where $\{w_k\}$ is $WGN(0,3)$. Starting points $\{x_0^j\}_{j=1}^{20}$ for the tracks were chosen randomly, no false alarms were added, and every target was detected at every sampling time. The resulting set Z of 80 measurements is shown in Fig. 6, where lines are drawn between points to illustrate how the data should be partitioned. It can be seen that the targets essentially move in straight lines in the sensor field of view (except for the random disturbances $\{w_k\}$).

The feasible track set F was computed using the techniques described previously, resulting in 20 actual tracks and 23 spurious tracks. Since only 4-point tracks were formed, and the models were the same for each track, the c-vector was computed using only the innovations sequences:

$$
c = \begin{bmatrix} 8.000 \\ 53.650 \\ 27.700 \\ 92.500 \\ 5.000 \\ 13.250 \\ 52.650 \\ 102.500 \\ 17.300 \\ 39.850 \\ 91.600 \\ 47.050 \\ .850 \\ 53.000 \\ 102.050 \\ 60.650 \\ 102.050 \\ 4.000 \\ 60.100 \\ 62.800 \\ 20.000 \\ 9.800 \\ 43.600 \\ 66.100 \\ 65.650 \\ 4.250 \\ 102.050 \\ 47.700 \\ 74.650 \\ 98.600 \\ 40.050 \\ 103.450 \\ 23.050 \\ 13.050 \\ 41.000 \\ 102.500 \\ 85.850 \\ 24.050 \\ 8.000 \\ 3.600 \\ 1.000 \\ 31.450 \\ 4.000 \end{bmatrix} \tag{50}
$$

The constraints for the set partitioning problem are as follows:

$$
\begin{bmatrix} \text{[illegible]} \end{bmatrix}^T = \begin{bmatrix} \vdots \end{bmatrix}. \tag{51}
$$

(Recall that the columns of this matrix represent the feasible track set.)

The algorithm achieved perfect accuracy when the set partitioning problem was solved. The total computation time was 1.5 s (CDC 7600), with 0.2 s to form F and 1.3 s used by the integer program.

A close examination of (51) reveals that the problem can be decomposed into a set of independent subproblems. To see this, define a symmetric incidence matrix W, whose elements are

$$
W_{ij} = \begin{cases} 0, & \text{if } (\lambda^i)^t(\lambda^j) = 0 \\ 1, & \text{if } (\lambda^i)^t(\lambda^j) \neq 0. \end{cases} \tag{52}
$$

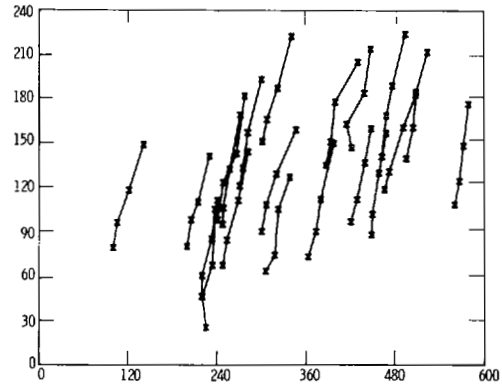


Fig. 6. Target simulation with measurements at four sample times.

The W matrix for this numerical example is as follows:

$$W = \left[\begin{array}{c} \text{(53)} \end{array}\right]$$

Note that, in some cases, single tracks are decoupled (e.g., consider the last row and last column of W).

In general, decoupling is possible when W is reducible, i.e., when a permutation matrix P exists such that

$$PWP^t = \operatorname{diag}(\tilde{W}_1, \tilde{W}_2, \cdots, \tilde{W}_L). \tag{54}$$

Determination of P decouples the original multitarget tracking problem into L independent subproblems. The decoupling procedure is obvious when we note that the tracks associated with $\tilde{W}_i$ and $\tilde{W}_j$ have no data points in common. As a result we can form a set of L subproblems whose solutions do not affect one another. The decoupling procedure is carried out after F is formed, and before the integer program is solved.

Problem size can be further reduced for set partitioning problems (32) by examining the matrix A for redundant constraints. Let a^i be the ith row of A. Then if

$$a^i \geqslant a^j, \tag{55}$$

the row a^i may be deleted. This result follows immediately from the form of the constraints (32). If the jth constraint is satisfied, then (55) guarantees that the ith constraint is also satisfied [1].

When we delete row i, we must also delete every column k such that

$$A_{ik} = 1 \quad \text{and} \quad A_{jk} = 0. \tag{56}$$

Again, this result follows from (32): allowing column k to remain after a^i is eliminated could result in a constraint violation. No information is lost in this process, since the constraint $a^j\tau = 1$ guarantees that the kth feasible track cannot be used [1].

Application of such methods can markedly decrease computation time, which depends exponentially on r and $\sum_{k=1}^{n} m(k)$.

VII. Conclusions

This paper has discussed a computationally efficient form of the multitarget tracking problem. Combinatorial difficulties that might otherwise prevent the use of a Bayesian decision process have been alleviated through the application of well-known 0-1 integer programming techniques. Our brief remarks on solving 0-1 integer programs only scratch the surface of a large body of literature. Other techniques include the use of cutting planes [1] and group theoretic methods [22], [23]. The Pierce implicit enumeration algorithm used in the above numerical example has worked quite well for moderate size multitarget tracking problems ($r < 10^3$, $\sum_{k=1}^{n} m(k) < 10^2$).

If feasible track construction is viewed as a type of detection process, then a large number of the feasible tracks that are found in dense tracking situations are false detections, composed of measurements from more than one track. The particular structure of the tracking problem allows us to accurately cull out these false tracks

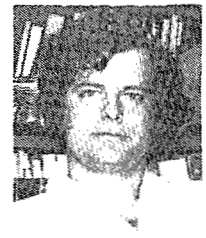
using set partitioning and set packing algorithms. Future work along these lines might investigate the more restricted structure of a sequential decision process. With a little effort, the A matrix in (30) can be made unimodular [1] for sequential tracking, so that a number of graph theoretic algorithms can be applied [1].

ACKNOWLEDGMENT

The author wishes to thank Dr. J. D. Gilchrist and R. M. Terasaki for their helpful suggestions.

REFERENCES

[1] R. S. Garfinkel and G. L. Nemhauser, *Integer Programming*. New York: Wiley, 1972.
[2] D. L. Alspach, "A Gaussian sum approach to the multitarget identification-tracking problem," *Automatica*, vol. 11, pp. 285–296, May 1975.
[3] Y. Bar-Shalom, "Extension of the probabilistic data association filter to multitarget tracking," in *Proc. 5th Symp. Nonlinear Estimation Theory and Its Applications*, Sept. 1974, pp. 16–21.
[4] C. L. Morefield, "Solution of multiple choice estimation problems via 0-1 integer programming," in *Proc. IEEE Conf. Decision and Control*, Nov. 1974, pp. 753–754.
[5] ——, "Efficient computational forms for Bayesian multitarget tracking," in *Proc. 6th Symp. Nonlinear Estimation Theory and Its Applications*, Sept. 1975, pp. 208–216 (based on Aerospace Corporation Rep. TOR/0076 (6085-50)-1, July 7, 1975).
[6] ——, "Application of 0-1 integer programming to a track assembly problem," in *Proc. IEEE Conf. Decision and Control*, Dec. 1975, pp. 428–433 (based on Aerospace Corporation Rep. TR-0075 (5085-10)-1, Apr. 16, 1975).
[7] ——, "Application of the ILLIAC IV parallel processor to multitarget tracking problems," in *Proc. 10th Annu. Asilomar Conf. Circuits, Systems, and Computers*, Nov. 1976.
[8] R. G. Sea, "Optimal correlation of sensor data with tracks in surveillance systems," in *Proc. 6th Int. Conf. Systems Science*, Jan. 1973, pp. 424–426.
[9] R. Singer and R. G. Sea, "New results in optimizing surveillance system tracking and data correlation performance in dense multitarget environments," *IEEE Trans. Automat. Contr.*, vol AC-18, pp. 571–581, Dec. 1973.
[10] R. A. Singer, R. G. Sea, and K. B. Housewright, "Derivation and evaluation of improved tracking filters for use in dense multitarget environments," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 423–432, July 1974.
[11] R. W. Sittler, "An optimal data association problem in surveillance theory," *IEEE Trans. Mil. Elec.*, vol. MIL-8, pp. 125–139, Apr. 1964.
[12] P. Smith and G. Buechler, "A branching algorithm for discriminating and tracking multiple objects," *IEEE Trans. Automat. Contr.*, vol. AC-20, pp. 101–104, Feb. 1975.
[13] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.
[14] R. S. Garfinkel and G. L. Nemhauser, "A survey of integer programming emphasizing computation and relations among models," in T. C. Ho and S. M. Robinson, Eds., *Mathematical Programming: Proceedings*. New York: Academic, 1973, pp. 75–155.
[15] R. E. Kalman, "A new approach to linear filtering and prediction theory," *J. Basic Eng.*, vol. 82D, pp. 35–45, Mar. 1960.
[16] J. S. Meditch, *Stochastic Optimal Linear Estimation and Control*. New York: McGraw-Hill, 1969.
[17] D. L. Alspach, "Gaussian sum approximation in nonlinear filtering and control," *Inform. Sci.*, vol. 7, pp. 271–290, Fall 1974.
[18] H. W. Sorenson, "On the development of practical nonlinear filters," *Inform. Sci.*, vol. 7, pp. 253–270, Fall 1974.
[19] M. B. Wells, *Elements of Combinatorial Computing*. New York: Pergamon, 1971.
[20] J. F. Pierce, "Application of combinatorial programming to a class of all-zero-one integer programming problems," *Management Sci.*, vol. 15, pp. 191–209, Nov. 1968.
[21] J. F. Pierce and J. S. Lasky, "Improved combinatorial programming algorithms for a class of all-zero-one integer programming problems," *Management Sci.*, vol. 19, pp. 528–543, Jan. 1973.
[22] R. Marsten, "An algorithm for large set partitioning problems," *Management Sci.*, vol. 20, pp. 774–787, Jan. 1974.
[23] H. Thiriez, "Airline crew scheduling: A group theoretic approach," Ph. D. dissertation, Massachusetts Inst. Technol., Cambridge, 1969.



Charles L. Morefield (S'70–M'75) was born in Fayetteville, TN, on December 30, 1942. He received the B. S. degree in physics from Tulane University of Louisiana, New Orleans, and the Ph.D. degree in engineering physics from the University of California, San Diego, in 1964 and 1973, respectively.

From 1964 to 1966 he served in the United States Navy as an Operations Officer aboard a small combatant. He was on the technical staff of Chrysler Corporation from 1966 to 1967, and TRW Systems Group from 1967 to 1969. From 1969 to 1973 he worked on his Ph.D. dissertation at the University of California, where he held the position of Research Assistant. He was on the technical staff of Aerospace Corporation from 1973 to 1975 before joining ORINCON Corporation. At ORINCON, he is Principal Investigator on research contracts in the areas of computer communication networks and multitarget tracking. His current interests lie primarily in the solution of large-scale estimation and control problems.

Dr. Morefield is a member of the Association for Computing Machinery.

Dynamic Equations in Descriptor Form

DAVID G. LUENBERGER, FELLOW, IEEE

***Abstract*—This paper studies a general form of sets of equations that is often the product of problem formulation in large-scale systems, especially when the equations are expressed in terms of the natural describing variables of the system. Such equations represent a broad class of time-evolutionary phenomena, and include as special cases ordinary static equations of arbitrary dimension, ordinary state-space equations, combinations of static and dynamic equations, and noncausal systems.**

The main thrust of the paper is to show (for sets of linear equations) that familiar concepts of dynamic system theory can be extended to this more general class, although sometimes with significant modification. Two new (and essentially dual) concepts, that of solvable and conditionable sets of equations, are found to be fundamental to the study of equations of this form. The notion of initial conditions, although not directly related to a state, is used as a general solution method for equations of this type. In

Manuscript received May 24, 1976; revised October 21, 1976. Paper recommended by E. J. Davison, Past Chairman of the IEEE S-CS Computational Methods and Discrete Systems Committee. This work was supported in part by National Science Foundation Grant GK 41481, and in part by the Division of Electric Energy Systems, Energy Research and Development Administration under Grant E(49-18)-2090.

The author is with the Department of Engineering-Economic Systems, Stanford University, Stanford, CA 94305.