The Annals of Statistics
1977, Vol. 5, No. 2, 237–255

OPTIMAL STRATEGIES FOR A CLASS OF CONSTRAINED SEQUENTIAL PROBLEMS[1]

BY JOSEPH B. KADANE AND HERBERT A. SIMON

Carnegie-Mellon University

This paper considers and unifies two sequential problems which have been extensively discussed. A class of sequential problems is proposed that includes both. An arbitrary partial ordering constraint is permitted to restrict possible strategies. An algorithm is proposed for finding the optimal strategy, and we prove that a strategy is optimal for the class of problems if and only if it can be found by the algorithm. The main tool is a set of functional equations in strategy space.

1. Two important sequential problems. The first sequential problem considers a single object hidden in the kth of n boxes with probability p_k. Without loss of generality, suppose $\sum p_k = 1$. A *search strategy* for finding the object is a permutation of a subset of the first n integers saying what to do next if the object has not yet been found. Thus $(9, 2, 3, \cdots)$ is interpreted to mean that box 9 is to be searched first; if the object is not found then box 2 is searched, etc. In this section we consider the simplified model in which a search of a box containing the object is sure to be successful, although this assumption is later relaxed. A search of box k costs c_k if it is unsuccessful and x_k if it is successful.

There are at least two kinds of such searches. In a *detection* search, the goal is to find an object in some search of some box. In a *whereabouts* search, the goal is to state correctly at the end of a search which box contains an object. This can be accomplished either by finding an object in the search, as in the detection case, or alternatively, by guessing correctly at the end of an unsuccessful search which box contains an object. See Kadane (1971) for a treatment of optimal whereabouts search.

In this paper, the first sequential problem is to determine a search strategy that includes each of the boxes and minimizes the expected cost of a detection search. An earlier paper (Kadane (1968)) deals with maximizing the probability of a successful detection search spending no more than some budget B (when $x_k \leqq c_k$ for all k).

Received March 1975; revised July 1976.

[1] Kadane was supported in part by the Office of Naval Research under contract numbers N00014-67-A-0314-0022 and N00014-75C-0516; Simon was supported in part by Research Grant MH-07722 from the National Institute of Mental Health and in part by the Advanced Research Projects Agency of the Office of the Secretary of Defense (F44620-73-C-0074) which is monitored by the Air Force Office of Scientific Research. The authors are grateful to Michael Garey, Richard Mickelson and Victor Mizel for helpful comments.

AMS 1970 *subject classifications.* Primary 90B40; Secondary 49C15.

Key words and phrases. Search theory, detection search, constrained search, functional equations in strategy space, sequencing, scheduling, job shop problem, job sequencing, single-machine job sequencing.

Let σ_i $(i = 1, 2)$ be disjoint strategies. Then $\sigma = \sigma_1\sigma_2$ is a strategy which looks first at the boxes specified by σ_1, in the order specified by σ_1, and then at the boxes specified by σ_2, in the order specified by σ_2, until the object is found or σ is exhausted. Also let $\sigma_k{}^*$ be the strategy consisting of a search at box k only.

For a strategy σ, let $X(\sigma)$ be the expected cost of σ, $P(\sigma)$ be the probability that σ is successful and $C(\sigma)$ be the conditional expected cost of σ, given that σ is unsuccessful. Then we have the initial conditions

$$\begin{aligned} X(\sigma_k{}^*) &= p_k x_k + (1 - p_k)c_k \, , \\ C(\sigma_k{}^*) &= c_k \, , \\ P(\sigma_k{}^*) &= p_k \, , \end{aligned} \tag{1.1}$$

and the recurrence relations

$$\begin{aligned} X(\sigma_1\sigma_2) &= X(\sigma_1) + X(\sigma_2) - P(\sigma_1)C(\sigma_2) \, , \\ C(\sigma_1\sigma_2) &= C(\sigma_1) + C(\sigma_2) \, , \\ P(\sigma_1\sigma_2) &= P(\sigma_1) + P(\sigma_2) \, . \end{aligned} \tag{1.2}$$

The first equation in (1.2) arises because $X(\sigma_1) + X(\sigma_2)$ is the cost of going ahead with σ_2 even if the object was found using σ_1. The probability of its being found in σ_1 is $P(\sigma_1)$; and if it was, it is sure not to be found in σ_2, so $C(\sigma_2)$ is the appropriate cost.

For consistency, if Λ is the empty strategy, define

$$C(\Lambda) = P(\Lambda) = X(\Lambda) = 0 \, . \tag{1.3}$$

Using these definitions, C, X and P are associative and C and P are commutative. Problems of this type are considered by Adophson and Hu (1973), Bellman (1957), Black (1965), Blackwell (n.d., see Matula (1964)), Denby (1967), Greenberg (1964), Hall (1976), Horn (1972), Kadane (1968), Matula (1964), Sidney (1975), Staroverov (1963), and Stone (1975, pages 110–114), among others.

The second sequential problem can also be stated as a search problem. In this problem there may be any number of objects, from zero to n, hidden. The event E_k that an object is hidden in the kth of n boxes again has probability p_k, but E_k is now independent of $E_{k'}$ $(k \neq k')$, where in the first problem $\{E_k, k = 1, \cdots, n\}$ is *disjoint*. Again a search of box k costs c_k if it is unsuccessful and x_k if it is successful. Once again we assume for simplicity that a search of a box is sure to be successful if an object is there, although this assumption will be relaxed later. Search continues until the first object is found or until all boxes have been searched. We seek a search strategy having minimum expected cost.

For the strategy σ, let $V(\sigma)$ be the expected cost of using σ and $S(\sigma)$ be the probability that the strategy is *not* successful in finding an object. Then we have the initial conditions

$$\begin{aligned} V(\sigma_k{}^*) &= p_k x_k + (1 - p_k)c_k \, , \\ S(\sigma_k{}^*) &= 1 - p_k \, , \end{aligned} \tag{1.4}$$

and the recurrence relations

$$(1.5) \qquad V(\sigma_1\sigma_2) = V(\sigma_1) + S(\sigma_1)V(\sigma_2)\,, \qquad S(\sigma_1\sigma_2) = S(\sigma_1)S(\sigma_2)\,.$$

For consistency, if Λ is the empty strategy, define

$$(1.6) \qquad V(\Lambda) = 0\,, \qquad S(\Lambda) = 1\,.$$

Problems of this type are considered by Bellman (1957), Dean (1966), Garey (1973), Joyce (1971), Kadane (1969), Mitten (1960), Simon and Kadane (1975), and Sweat (1970).

2. A convenient class of problems embracing both search problems. This section owes a large, but not transparent, debt to the paper of Rau (1971). The class of problems proposed below is a proper subclass of the class proposed by Rau (see Kadane (1975)), which in turn is a proper subclass of the class proposed by Smith (1956). Suppose that three functions f, F and G are defined on strategies $\sigma_k{}^*$ consisting of a single search of box k. Suppose also that f, F and G are extended to arbitrary strategies by the recurrence relations

$$(2.1) \qquad F(\sigma_1\sigma_2) = F(\sigma_1) + F(\sigma_2) + G(\sigma_1)f(\sigma_2)$$

$$(2.2) \qquad f(\sigma_1\sigma_2) = f(\sigma_1) + [1 + mG(\sigma_1)] \cdot f(\sigma_2)$$

$$(2.3) \qquad G(\sigma_1\sigma_2) = G(\sigma_1) + [1 + mG(\sigma_1)]G(\sigma_2)$$

where m is a fixed number. For an empty strategy Λ, we take

$$(2.4) \qquad F(\Lambda) = f(\Lambda) = G(\Lambda) = 0\,.$$

First we establish a basic theorem about the system 2.1-3:

THEOREM 1. *With the above definitions, f, F and G are defined consistently on strings of arbitrary length. In particular*

$$(2.5) \qquad F((ab)c) = F(a(bc))\,,$$

$$(2.6) \qquad f((ab)c) = f(a(bc))\,,$$

$$(2.7) \qquad G((ab)c) = G(a(bc))\,.$$

PROOF.

$$\begin{aligned} F((ab)c) &= F(ab) + F(c) + G(ab)f(c) \\ &= F(a) + F(b) + G(a)f(b) + F(c) + f(c)[G(a) + G(b) + mG(a)G(b)] \\ F(a(bc)) &= F(a) + F(bc) + G(a)f(bc) \\ &= F(a) + F(b) + F(c) + G(b)f(c) + G(a)[f(b) + [1 + mG(b)]f(c)]\,, \end{aligned}$$

thus $F((ab)c) = F(a(bc))$, proving (2.5).

The proofs of (2.6) and (2.7) are similar, and are therefore omitted. □

Next we establish that the two search problems of Section 1 are special cases of the system 2.1-3.

THEOREM 2.

(a) *When $m = 0$, associating f with $-C$, G with P, and F with X, the system* (2.1-3) *yields the recurrence relations* (1.2).

(b) *When $m = -1$, associating F and $-f$ with V, and G with $1 - S$ yields a consistent set of recurrence relations identical with* (1.5).

PROOF.

(a) Let $m = 0$, and make the substitutions indicated. (1.2) is immediate.

(b) Let $m = -1$, and consider (2.3):

$$\begin{aligned} 1 - S(\sigma_1\sigma_2) &= 1 - S(\sigma_1) + S(\sigma_1)[1 - S(\sigma_2)] \\ &= 1 - S(\sigma_1)S(\sigma_2) . \end{aligned}$$

Now the second equation of (1.5) is immediate. Next consider (2.1):

$$\begin{aligned} V(\sigma_1\sigma_2) &= V(\sigma_1) + V(\sigma_2) - G(\sigma_1)V(\sigma_2) \\ &= V(\sigma_1) + S(\sigma_1)V(\sigma_2) \end{aligned}$$

which repoduces the first equation of (1.5).

Finally, consider (2.2):

$$-V(\sigma_1\sigma_2) = -V(\sigma_1) - S(\sigma_1)V(\sigma_2)$$

which again reproduces the first equation of (1.5). This shows that the substitutions yield a consistent set of equations identical with (1.5). □

That the first equation of (1.5) has two (identical) generalizations in (2.1) and (2.2), causes no problem in the sequel.

Thus the system (2.1) to (2.3) is a class of sequential problems including both search problems proposed in Section 1. Kadane (1975) shows that, allowing for a possible rescaling of f and G, any two nonzero values of m yield equivalent problems. Thus $m = 0$ and $m = -1$ are in this sense the only two essentially different values of m in (2.1) to (2.3). Hence, without loss of generality, we may assume $m \leqq 0$ for the remainder of this paper. However, when $m = -1$, the two equations (2.1) and (2.2) need not collapse into one as they do for the second example above, so there are really three essentially different special cases of the system (2.1) to (2.3), of which two are the examples of Section 1.

3. Constraints. Reconsider the first search problem of Section 1 where now there is a probability $\alpha_{j,k}$ of overlooking the object in the jth search of box k, given that it is in box k and has not been found before the jth search of box k. Then the unconditional probability $p_{j,k}$ that the jth search of box k is successful (if it is in the search strategy) satisfies

$$p_{j,k} = p_k(1 - \alpha_{j,k}) \textstyle\prod_{0<j'<j} \alpha_{j',k} . \tag{3.1}$$

Additionally the jth search of box k can be supposed to cost some amount $c_{j,k}$ if it is unsuccessful and $x_{j,k}$ if it is successful. The notation can be simplified by denoting the jth search of box k by a single index, say i. Thus p_i is the probability of success, c_i the cost if unsuccessful and x_i the cost if successful, of some search. If the object is found in the jth search of box k, it is found in no other search of any box. Hence the events $E_{j,k}$ that the object is found in the jth search of box k are disjoint. In effect this observation allows $\alpha_{j,k} = 0$ without loss of generality, at the cost of introducing a constraint on the optimal strategy. A strategy is called *feasible* if the jth search of box k is preceded by the $(j - 1)$st search of box k for every k and every $j > 1$. Clearly feasible strategies are the only ones which make sense.

Constraints of this type are called "parallel" because they can be graphed as n parallel lines, one for each box, indicating that the jth search of box k must be preceded by the $(j - 1)$st of box k and must precede the $(j + 1)$st search of box k.

A similar generalization of the second problem would have the jth search of box k cost $c_{j,k}$ if unsuccessful, $x_{j,k}$ if successful, and have probability $p_{j,k}$ of success. In order for this to be a valid generalization of the second problem, the event $E_{j,k}$ must be independent of $E_{j',k'}$ provided $(j, k) \neq (j', k')$. Again only feasible strategies are interesting. See Kadane (1969) for a discussion.

More generally, suppose S is a set of searches and C is a set of constraints, or arcs, a subset of $S \times S$. Thus if $c = (s_1, s_2) \in C$, then search s_1 must be conducted before search s_2. The pair (S, C) form a graph. See Berge (1962, 1973). The *transitive closure* C^* of C is the subset of $S \times S$ such that $(s_1, s) \in C^*$ iff there exist $s_1, s_2, \cdots, s$ such that $(s_1, s_2) \in C$, $(s_2, s_3) \in C$, $\cdots$. Thus (S, C^*) is again a graph, and has all the constraints implied by C^* and transitivity. If $(s_1, s_2) \in C^*$ then s_1 is a *predecessor* of s_2 and s_2 is a *successor* of s_1.

We now restrict the discussion to graphs such that, if $(s_1, s) \in C^*$, there is a *finite* sequence $(s_1, s_2, \cdots, s_r, s)$ such that $(s_1, s_2) \in C$, $(s_2, s_3) \in C$, $\cdots$, $(s_r, s) \in C$. Notice that in the case of parallel constraints above this restriction is satisfied. A case where it would not be satsfied is where all searches of box 1 had to be completed before any searches of box 2 could be undertaken. With this restriction, if s_1 is a predecessor of s_2, and no other predecessor of s_2 is a successor of s_1, then s_1 is an *immediate predecessor* of s_2 and s_2 is an *immediate successor* of s_1. The *immediate graph* C^- is formed by $(s_1, s_2) \in C^-$ iff s_1 is an immediate predecessor of s_2.

The case of parallel constraints is then seen to satisfy the restriction that every search has no more than one immediate predecessor and no more than one immediate successor.

A *cycle* is a sequence of arcs

$$\mu = (u_1, \cdots, u_q)$$

such that

(1) each arc u_k; $1 < k < q$, has one endpoint in common with u_{k-1} and the other endpoint in common with u_{k+1};

(2) the same arc does not appear twice;

(3) the endpoint u_1 does not share with u_2 is the same as the endpoint u_q does not share with u_{q-1}.

A *chain* satisfies the first condition above only. Notice that in a cycle the endpoint u_k shares with u_{k+1} need not be the successor in u_k. Thus $(s_1, s_2)(s_3, s_2)(s_3, s_1)$ is a cycle.

A *connected* graph is a graph which contains, for every two nodes x and y, a chain from x to y. Since the relation, $x = y$ or there is a chain from x to y, is an equivalence relation, the equivalence classes divide S into *connected components*. Finally a *tree* is a connected graph without cycles, and a *forest* is a graph without cycles, i.e., a graph whose connected components are trees.

A forest is thus a more general structure than parallel constraints. The theory of Sections 4 and 5 applies to an arbitrary graph of constraints on S. However the Garey reduction algorithm of Section 5 applies especially well to finite forests. Further details about graph theory may be found in many books, for example those of Berge (1962, 1973).

4. Search over a partially ordered set. In this section we present several algorithms and prove that a strategy is optimal if and only if it can be found by any of the algorithms for every member of the class of problems introduced in Section 2 under arbitrary partial ordering constraints. This section generalizes the main results of Sidney (1975) and Simon and Kadane (1975). Since we are dealing with algorithms we assume that the basic set S is finite, although many of our results hold more generally.

Before stating the first algorithm, a few lemmas are convenient.

LEMMA 1.

$$G(ab) = G(ba) .$$

PROOF.

$$\begin{aligned} G(ab) &= G(a) + [1 + mG(a)]G(b) \\ &= G(a) + G(b) + mG(a)G(b) \\ &= G(b) + [1 + mG(b)]G(a) \\ &= G(ba) . \end{aligned}$$

□

LEMMA 2.

$$F(abcd) - F(acbd) = [1 + mG(a)]\{f(c)G(b) - f(b)G(c)\} .$$

PROOF.

$$\begin{aligned} &F(abcd) - F(acbd) \\ &\quad = F(abc) + F(d) + G(abc)f(d) - F(acb) - F(d) - G(acb)f(d) \\ &\quad = F(abc) - F(acb) \\ &\quad = F(a) + F(bc) + G(a)f(bc) - F(a) - F(cb) - G(a)f(cb) \\ &\quad = F(bc) - F(cb) + G(a)[f(bc) - f(cb)] \end{aligned}$$

$$= F(b) + F(c) + G(b)f(c) - F(b) - F(c) - G(c)f(b)$$
$$+ G(a)[f(b) + f(c) + mG(b)f(c) - f(b) - f(c) - mG(c)f(b)]$$
$$= G(b)f(c) - G(c)f(b) + mG(a)[G(b)f(c) - G(c)f(b)]$$
$$= [1 + mG(a)][G(b)f(c) - G(c)f(b)]\,,$$

where Lemma 1 has been used several times. □

A feasible strategy on a set of nodes S is any strategy of the nodes of S that satisfies the order constraints C on those nodes. An optimal strategy for (S, C) is a feasible strategy on S that minimizes F over feasible strategies on S.

For the rest of the paper, assume $1 + mG(a) > 0$ for all a, and $G(a) > 0$ for all nonempty a. For every nonempty a, let $\varnothing(a) = f(a)/G(a)$.

THEOREM 3. *Suppose $(abcd)$ and $(acbd)$ are feasible strategies, b and c are nonempty, and $\varnothing(c) > \varnothing(b)$.*
Then $F(acbd) < F(abcd)$, so $(abcd)$ is not optimal.

PROOF. Using Lemma 2,

$$F(abcd) - F(acbd) = [1 + mG(a)]\{f(c)G(b) - f(b)G(c)\}$$
$$= [1 + mG(a)]G(b)G(c)[\varnothing(c) - \varnothing(b)] > 0\,. \qquad \square$$

Let A be a partially ordered set of nodes, and let it contain B and $D = A - B$. We say B is an *initial subset* of A and D is a *terminal subset* of A iff there exist feasible strategies b on B and d on D such that $a = (bd)$ is a feasible strategy on A. If B is initial for A and D is terminal for A, and if b' is a feasible strategy for B and d' is a feasible strategy for D, then $a' = (b'd')$ is a feasible strategy for A.

If a is a strategy, the *set* of a, denoted $T(a)$, is the set of nodes ordered by a. A strategy a is $\varnothing^*$-maximal for the set N iff

(i) $T(a)$ is initial for N;
(ii) $\varnothing(a) \geqq \varnothing(a')$ for all a' such that $T(a')$ is initial for N;
(iii) a has no substrategies satisfying (i) and (ii).

ALGORITHM 1.

0. Set the current initial set N equal to S, set $\alpha = \Lambda$.
1. Let a' be a $\varnothing^*$-maximal strategy for N.
2. Let a be a feasible strategy minimizing F over all feasible, $\varnothing^*$-maximal strategies on $T(a')$.
3. Append a to the end of α.
4. Replace N by $N - T(a)$.
5. If N is empty, stop. α is the strategy found by Algorithm 1. Otherwise return to 1.

We now prove a series of lemmas leading to a proof of the following theorem:

THEOREM 6. *A strategy is optimal if and only if it can be generated by Algorithm* 1.

The following function and lemma are useful in the sequel. Let $R(\alpha) = f(\alpha) - mF(\alpha)$.

LEMMA 3.

$$R(\alpha_1\alpha_2) = R(\alpha_1) + R(\alpha_2) .$$

PROOF.

$$\begin{aligned} R(\alpha_1\alpha_2) &= f(\alpha_1\alpha_2) - mF(\alpha_1\alpha_2) \\ &= f(\alpha_1) + [1 + mG(\alpha_1)]f(\alpha_2) - m(F(\alpha_1) + F(\alpha_2) + G(\alpha_1)f(\alpha_2)) \\ &= f(\alpha_1) - mF(\alpha_1) + f(\alpha_2) - mF(\alpha_2) \\ &= R(\alpha_1) + R(\alpha_2) . \end{aligned}$$ □

Therefore R is symmetric. Let $H(\alpha_2; \alpha_1) = F(\alpha_2) + G(\alpha_1)f(\alpha_2)$.

LEMMA 4. *Let α_1 be fixed. α_2 minimizes $F(\cdot)$ over $T(\alpha_2)$* iff *α_2 minimizes $H(\cdot; \alpha_1)$ over $T(\alpha_2)$.*

PROOF. α_2 minimizes $H(\cdot; \alpha_1)$ over $T(\alpha_2)$ iff α_2 minimizes

$$\begin{aligned} H(\cdot; \alpha_1) - G(\alpha_1)R(\cdot) &= F(\cdot) + G(\alpha_1)f(\cdot) - G(\alpha_1)f(\cdot) + mG(\alpha_1)F(\cdot) \\ &= [1 + mG(\alpha_1)]F(\cdot) \end{aligned}$$

over $T(\alpha_2)$ iff α_2 minimizes $F(\cdot)$ over $T(\alpha_2)$. Note that the symmetry of R proved in Lemma 3 is used in the first statement above. □

LEMMA 5. *Suppose an optimal strategy β has an initial set I_1 so that $\beta = (\beta_1, \beta_2)$ where $T(\beta_1) = I_1$ and let $I_2 = T(\beta_2)$. Then a feasible strategy of the form $\alpha = (\alpha_1\alpha_2)$ where $T(\alpha_1) = I_1$ and $T(\alpha_2) = I_2$ is optimal if and only if*

(i) *α_1 is optimal for I_1*
(ii) *α_2 is optimal for I_2.*

PROOF. In view of Lemma 4, the following can be substituted for (ii) above: (ii′) α_2 minimizes $H(\ ;\alpha_1)$ over I_2.

Suppose α_1 is optimal for I_1, α_2 is optimal for I_2 and α_2 minimizes $H(\ ;\alpha_1)$ over I_2. Then

$$F(\beta_1) \geqq F(\alpha_1), \quad F(\beta_2) \geqq F(\alpha_2) \qquad \text{and} \qquad H(\beta_2; \alpha_1) \geqq H(\alpha_2; \alpha_1) .$$

By Lemma 1, G is symmetric so $G(\alpha_1) = G(\beta_1)$, so $H(\ ;\alpha_1) = H(\ ;\beta_1)$. Then

$$\begin{aligned} F(\beta) &= F(\beta_1) + H(\beta_2; \beta_1) \\ &\geqq F(\alpha_1) + H(\alpha_2; \alpha_1) \\ &= F(\alpha) . \end{aligned}$$

Optimality of β now implies optimality of α.

Conversely now suppose that α is optimal, and suppose, contrary to the hypothesis, that α_1 is not optimal for I_1, i.e., suppose α^* satisfies

$$F(\alpha^*) < F(\alpha_1) \qquad \text{and} \qquad T(\alpha^*) = T(\alpha_1) = I_1 .$$

Since G is symmetric, $G(\alpha^*) = G(\alpha_1)$. Then

$$\begin{aligned} F(\alpha^*\alpha_2) &= F(\alpha^*) + F(\alpha_2) + G(\alpha^*)f(\alpha_2) \\ &< F(\alpha_1) + F(\alpha_2) + G(\alpha_1)f(\alpha_2) \\ &= F(\alpha)\,, \end{aligned}$$

which contradicts the optimality of α. Hence α_1 is optimal for I_1.

Now suppose that α is optimal and suppose, contrary to the hypothesis, that α_2 does not minimize $H(\ ;\alpha_1)$ over I_2, i.e., suppose α^* satisfies

$$H(\alpha^*;\alpha_1) < H(\alpha_2;\alpha_1) \qquad \text{and} \qquad T(\alpha^*) = T(\alpha_2) = I_2\,.$$

Then

$$\begin{aligned} F(\alpha_1\alpha^*) &= F(\alpha_1) + H(\alpha^*;\alpha_1) \\ &< F(\alpha_1) + H(\alpha_2;\alpha_1) \\ &= F(\alpha)\,. \end{aligned}$$

Then α is not optimal, which is a contradiction. Thus α_2 minimizes $H(\ ;\alpha_1)$ over I_2. □

LEMMA 6. *Let a and b be nonempty strategies. Then*

$$\varnothing(ab) = \varnothing(a)\left(\frac{G(a)}{G(ab)}\right) + \varnothing(b)(1 - G(a)/G(ab))\,.$$

PROOF.

$$\begin{aligned} \varnothing(ab) &= \frac{f(ab)}{G(ab)} = \frac{f(a) + [1 + mG(a)]f(b)}{G(a) + [1 + mG(a)]G(b)} \\ &= \frac{f(a)}{G(a)} \cdot \frac{G(a)}{G(ab)} + \frac{f(b)}{G(b)} \cdot \frac{G(b)[1 + mG(a)]}{G(ab)} \\ &= \varnothing(a)\left(\frac{G(a)}{G(ab)}\right) + \varnothing(b)(1 - G(a)/G(ab))\,. \end{aligned}$$

□

LEMMA 7. *If s is $\varnothing^*$-maximal, then $T(s)$ occurs as an uninterrupted string in every optimal strategy.*

PROOF. To show this, consider a related problem in which the constraint set C is reduced by eliminating all constraints in which a member of $T(s)$ is required to precede something not in $T(s)$. Thus we consider the constraint set

$$C' = C - \{(a, b) \mid (a, b) \in C, a \in T(s), b \notin T(s)\}\,.$$

Thus the feasible strategies for C are feasible for C'. We shall show (1) $T(s)$ must occur as an uninterrupted string in any optimal strategy for C' and (2) at least one optimal strategy of C' is feasible (and hence optimal) for C. Hence every optimal strategy for C is optimal for C' and therefore $T(s)$ occurs as an uninterrupted string in it.

To prove (1) assume the contrary, i.e., that there exists an optimal strategy α for C' such that

$$\alpha = g_1h_1g_2h_2, \cdots, g_rh_rg_{r+1}$$

where $r > 1$, $T(s) = \bigcup T(h_j)$, $(T(s))^c = \bigcup T(g_j)$ and all strategies g_j and h_j are nonempty except possibly g_1 and g_{r+1}. Let $k = \min\{j \mid \varnothing(h_j) \geqq \varnothing(h_m)\}$ for $m = 1, 2, \cdots, r$. Hence $\varnothing(h_k) \geqq \varnothing(h_j)$ for all j, and

$$\varnothing(h_k) > \varnothing(h_j) \qquad \text{for} \quad j < k . \tag{*}$$

Observe that $k \neq 1$, since if $k = 1$, h_1 would violate the $\varnothing^*$-maximality of s using Lemma 5.

Since α is optimal under C' and since, by construction of C', any g_i can be interchanged with an adjacent h_j without disturbing C' feasibility, we conclude by Theorem 3 that $\varnothing(g_k) \geqq \varnothing(h_k)$. By (*), $\varnothing(h_{k-1}) < \varnothing(h_k)$. Hence, exchanging h_{k-1} with g_k will yield a strategy α' which is feasible for C' and such that $F(\alpha') < F(\alpha)$, which contradicts the assumption that α is optimal for C'. Thus $T(s)$ occurs in an uninterrupted string in any optimal strategy for C', proving (1).

It remains to show that at least one optimal strategy of C' is feasible (and hence optimal) for C.

Let $\alpha' = \alpha_1 s' \alpha_2$ be optimal for C', where $T(s') = T(s)$. Clearly $\alpha = \alpha_1 s \alpha_2$ is feasible for C', and

$$\begin{aligned} F(\alpha') - F(\alpha) &= F(\alpha_1) + H(s'\alpha_2; \alpha_1) - F(\alpha_1) - H(s\alpha_2; \alpha_1) \\ &= H(s'\alpha_2; \alpha_1) - H(s\alpha_2; \alpha_1) \\ &= (H(s'\alpha_2; \alpha_1) - G(\alpha_1)R(s'\alpha_2)) - (H(s\alpha_2; \alpha_1) - G(\alpha_1)R(s\alpha_2)) \\ &= (1 + mG(\alpha_1))F(s'\alpha_2) - (1 + mG(\alpha_1))F(s\alpha_2) \\ &= (1 + mG(\alpha_1))(F(s'\alpha_2) - F(s\alpha_2)) , \quad \text{using Lemma 3.} \end{aligned}$$

Now

$$\begin{aligned} F(s'\alpha_2) - F(s\alpha_2) &= F(s') + G(s')f(\alpha_2) - F(s) - G(s)f(\alpha_2) \\ &= F(s') - F(s) , \quad \text{using Lemma 1.} \end{aligned}$$

Thus

$$F(\alpha') - F(\alpha) = (1 + mG(\alpha_1))(F(s') - F(s)) .$$

Optimality of α' for C' and feasibility of α for C' imply $F(\alpha') \leqq F(\alpha)$, so $F(s') \leqq F(s)$.

Now using the symmetry of R proved in Lemma 3, $R(s) = R(s')$, i.e., $f(s) - mF(s) = f(s') - mF(s')$. Then

$$m(F(s') - F(s)) = f(s') - f(s) = G(s)(\varnothing(s') - \varnothing(s)) .$$

Now $\varnothing^*$-optimality of s implies $\varnothing(s') \leqq \varnothing(s)$, and $G(s) > 0$, so either $m = 0$ or $F(s') \geqq F(s)$.

If $m \neq 0$, then $F(s') = F(s)$. Then $\varnothing(s') = \varnothing(s')$. On the other hand, if $m = 0$, then $f(s') = f(s)$, so $\varnothing(s') = \varnothing(s)$. Then in both cases, s' is $\varnothing^*$-optimal as well. •

Now consider $\alpha^* = s'\alpha_1\alpha_2$.

$$F(\alpha') - F(\alpha^*) = F(\alpha_1 s') - F(s'\alpha_1) = G(\alpha_1)G(s')(\varnothing(s') - \varnothing(\alpha_1)) = 0 .$$

Therefore α^* is optimal for C' and feasible for C. □

Lemma 7 is proved for the second example of Section 1 in Simon and Kadane (1975). The proof of Lemma 7 yields the following:

LEMMA 8. *If* s *is* $\varnothing^*$*-maximal, then there exists an optimal strategy* α *of the form* $\alpha = s'u$ *where* $T(s) = T(s')$ *and* $\varnothing(s) = \varnothing(s')$.

LEMMA 9. *If* $m \neq 0$, *and if* s *is* $\varnothing^*$*-maximal then there exists an optimal strategy of the form* $\alpha = su$.

PROOF. $\alpha^{**} = s\alpha_1\alpha_2$ suffices.

It may seem to the reader that Lemma 9 should also hold for $m = 0$. Consider, however, the following example: Suppose $S = \{a, b, c\}$ and $C = \{(a, c), (b, c)\}$ so that both a and b must be done before c. Thus abc and bac are the only feasible strategies including all of S, and the only feasible initial strategies are a, ab, abc, b, ba and bac. Now suppose $F(a) = F(b) = F(c) = 0$, $G(a) = G(b) = G(c) = \frac{1}{2}$ and $f(a) = 1$, $f(b) = 2$ and $f(c) = 6$. A simple computation shows that if $m = 0$, bac is $\varnothing^*$-maximal (as is abc) but not optimal. However if $m = -1$, bac is (uniquely) $\varnothing^*$-maximal and optimal.

THEOREM 4. *Algorithm* 1 *generates only optimal strategies.*

PROOF. By induction on $|N|$. If $|N| = 1$, the theorem is trivial. Suppose it is true for $|N| = 1, 2, \cdots, n - 1$, and assume $|N| = n$.

Let α_1 be the $\varnothing^*$-maximal subset selected during the first iteration. Then Algorithm 1 will ultimately produce an algorithm of the form $\alpha = \alpha_1\alpha_2$ (where α_2 might be empty).

Lemma 8 guarantees the existence of an optimal strategy α^* of the form $\alpha^* = \alpha_1'u$ where $T(\alpha_1) = T(\alpha_1')$ and $\varnothing(\alpha_1) = \varnothing(\alpha_1')$. Then α_1' is also $\varnothing^*$-maximal. By operation of step 2 of the algorithm, $F(\alpha_1) \leqq F(\alpha_1')$. Now Lemma 5 applies, and says that for $\alpha_1'u$ to be optimal, α_1' must minimize F over $T(\alpha_1')$. Then $F(\alpha_1') \leqq F(\alpha_1)$, so $F(\alpha_1') = F(\alpha_1)$. Then α_1 also minimizes F over $T(\alpha_1')$. The inductive hypothesis implies that α_2 minimizes F over $T(\alpha_2)$. Then Lemma 5 applies again, and implies that α is optimal. □

LEMMA 10. *Suppose* β *is an optimal strategy of the form* $\beta = s_1ss_2$, *where* s_1 *or* s_2 *or both may be empty. Suppose* β' *is the strategy* $\beta' = s_1s's_2$, *so* $T(s) = T(s')$. *Then* $\varnothing(s) \geqq \varnothing(s')$.

PROOF. If $m = 0$, $\varnothing(s) = \varnothing(s')$ for all s' such that $T(s) = T(s')$, so the lemma is easy. Suppose then $m < 0$. Then by Lemma 3,

$$mF(s) - f(s') = mF(s') - f(s')\,, \quad \text{i.e.,}$$
$$m[F(s) - F(s')] = f(s) - f(s') = G(s)(\varnothing(s) - \varnothing(s'))\,.$$

Now

$$0 \geqq F(\beta) - F(\beta') = F(s_1ss_2) - F(s_1s's_2) = F(s_1s) - F(s_1s')$$
$$= H(s; s_1) - H(s'; s_1)$$

$$= [H(s; s_1) - G(s_1)R(s)] - [H(s'; s_1) - G(s_1)R(s')]$$
$$= [1 + mG(s_1)][F(s) - F(s')]$$
$$= [1 + mG(s_1)]G(s)(\varnothing(s) - \varnothing(s'))/m .$$

Then

$$[1 + mG(s_1)] > 0,\ G(s) > 0,\ m < 0 \Rightarrow \varnothing(s) \geqq \varnothing(s') . \qquad \square$$

LEMMA 11. *Suppose a_1 and a_2 are distinct $\varnothing$*-maximal strategies. Then $T(a_1) = T(a_2)$ or $T(a_1) \cap T(a_2) = \wedge$.*

PROOF. Let β be an optimal strategy. By Lemma 7, a_1 and a_2 must occur as uninterrupted strings in β. If they overlap, the only way this might occur is for β to be of the form

$$\beta = s_1 a_1' x a_2' s_2$$

where $T(a_1) = T(a_1'x)$, $T(a_2) = T(xa_2')$ and $T(a_1) \cap T(a_2) = T(x)$. Suppose $T(x) \neq \wedge$. Since $T(a_2)$ is initial and xa_2' is feasible, x must be feasible and initial. Lemma 10 implies that $\varnothing(a_1) = \varnothing(a_1'x)$ and $\varnothing(a_2) = \varnothing(xa_2')$. Now if a_1' were empty, then $\varnothing(a_1) = \varnothing(x) = \varnothing(a_2)$, so x would be an initial substrategy of a_2 satisfying (i) and (ii) of the definition of $\varnothing$*-maximality of a_2, and therefore contradicting the assumption of $\varnothing$*-maximality of a_2. Therefore a_1' is not empty.

Since a_1' is not empty, and is an initial substrategy of a_1, $\varnothing(a_1') < \varnothing(a_1)$. Now Lemma 5 displays $\varnothing(a_1) = \varnothing(a_1'x)$ as a convex combination of $\varnothing(a_1')$ and $\varnothing(x)$. Then $\varnothing(x) > \varnothing(a_1)$. But x is initial and not empty, so this contradicts the $\varnothing$*-maximality of a_1. Hence x is empty. $\square$

LEMMA 12. *If α is an optimal strategy, then there is a $\varnothing$*-maximal strategy α_1 such that $\alpha = \alpha_1\alpha_2$, and such that α_1 minimizes F over all feasible, $\varnothing$*-maximal strategies on $T(\alpha_1)$.*

PROOF. Suppose there are k nonoverlapping $\varnothing$*-maximal strategies $s_1, s_2, \cdots, s_k$ and suppose

$$\alpha = \beta_1 s_1' \beta_2 s_2' \cdots \beta_k s_k' \beta_{k+1}$$

where $T(s_i') = T(s_i)$, $i = 1, \cdots, k$. Lemmas 7 and 11 allow α to be written in this form. Suppose β_1 is not empty. By construction, β_1 is not $\varnothing$*-maximal, nor is any substrategy of β_1 $\varnothing$*-maximal. Lemma 10 implies $\varnothing(s_i') = \varnothing(s_i)$, $i = 2, \cdots, k$.

Now $\varnothing(\beta_1) > \varnothing(s_1') = \varnothing(s_1)$ contradicts in $\varnothing$*-maximality of s_1. $\varnothing(\beta_1) = \varnothing(s_1)$ and β_1 being initial implies that β_1 or some substrategy of β_1 is $\varnothing$*-maximal, which contradicts the construction. Then $\varnothing(\beta_1) < \varnothing(s_1)$. By Theorem 3, α is not optimal, which again is a contradiction. Hence β_1 is empty.

Now suppose α_1 does not minimize F over all feasible, $\varnothing$*-maximal strategies on $T(\alpha_1)$. Then let α_1' be a feasible strategy such that $T(\alpha_1) = T(\alpha_1')$, $\varnothing(\alpha_1) = \varnothing(\alpha_1')$ and $F(\alpha_1) > F(\alpha_1')$. $\alpha_1'\alpha_2$ is a feasible strategy for S. Then

$$F(\alpha_1'\alpha_2) - F(\alpha_1\alpha_2) = F(\alpha_1') + G(\alpha_1')f(\alpha_2) - F(\alpha_1) - G(\alpha_1)f(\alpha_2)$$
$$= F(\alpha_1') - F(\alpha_1) < 0 .$$

This contradicts the optimality of α. Hence α_1 minimizes F over all feasible $\varnothing^*$-maximal strategies on $T(\alpha_1)$. □

THEOREM 5. *If α is optimal, it can be generated by Algorithm* 1.

PROOF. The theorem is trivial if $|N| = 1$. Suppose it is true for $|N| = 1, 2, \cdots, n-1$ and assume $|N| = n$. Let α be an optimal strategy on N. By Lemma 12, there is a $\varnothing^*$-maximal strategy α_1 that minimizes F over all feasible, $\varnothing^*$-maximal strategies on $T(\alpha_1)$ such that $\alpha = \alpha_1\alpha_2$. Then Algorithm 1 might choose α_1 in its first iteration. This leaves a problem on $N = N - T(\alpha_1)$ with $|N - T(\alpha_1)| < n$. Now Lemma 5 applies to show that α_2 is optimal on $T(\alpha_2) = N - T(\alpha_1)$. And the inductive hypothesis applies to show that α_2 can be generated by Algorithm 1 for $|N - T(\alpha_1)| = |T(\alpha_2)| < n$. Hence α can be generated by Algorithm 1. □

THEOREM 6. *A strategy α is optimal if and only if α can be generated by Algorithm* 1.

PROOF. If: Theorem 4; only if: Theorem 5. □

We note that if $m \neq 0$, step 2 of Algorithm 1 is unnecessary, by application of Lemma 9. The example following Lemma 9 shows that without step 2, Algorithm 1 could generate suboptimal strategies when $m = 0$.

LEMMA 13. *Suppose the sequence of $\varnothing^*$-maximal strategies generated by two applications of Algorithm* 1 *are $(s_1, s_2, \cdots, s_a)$ and $(t_1, t_2, \cdots, t_b)$ where the strategies are ordered in order of their generation by the algorithm. Then $a = b$ and there is a permutation π of $\{1, \cdots, a\}$ such that $T(s_i) = T(t_{\pi(i)})$ for $i = 1, \cdots, a$.*

PROOF. By induction on $|S|$. The proof is a straightforward extension of Lemma 10 of Sidney (1975). □

Then the class of sets of strategies $E = T(e)$ such that e occurs in some sequence of $\varnothing^*$-maximal subsets is well defined, independent of the particular realizations of steps 1 and 2 of Algorithm 1. Consider the restrictions defined on sets E by $\tilde{\mathscr{C}} = \{(E_i, E_j) \mid (e_i, e_j) \in C^*$ for some $e_i \in E_i$, $e_j \in E_j$ or $\varnothing(e_i) > \varnothing(e_j)$ where $\varnothing(e_i) \geqq \varnothing(e_i')$ for all strategies e_i on E_i, and $\varnothing(e_j) \geqq \varnothing(e_j')$ for all strategies e_j on $E_j\}$.

Lemma 13 can now be rephrased as

THEOREM 7. *A strategy α is optimal* iff *it is of the form $\alpha = \alpha_{i(1)}\alpha_{i(2)} \cdots \alpha_{i(a)}$ where*

(1) *$T(\alpha_{i(k)}) = E_{i(k)}$ and $\alpha_{i(k)}$ is optimal for $E_{i(k)}$;*
(2) *$E_{i(1)}, \cdots, E_{i(a)}$ is consistent with $\tilde{\mathscr{C}}$.*

A strategy a is $\varnothing$-maximal for N iff

(i) $T(a)$ is initial for N;
(ii) $\varnothing(a) \geqq \varnothing(a')$ for all a' such that $T(a')$ is initial for N.

Theorem 7 implies the following corollary:

Any $\emptyset$-maximal strategy is a concatenation of optimal strategies on E-sets.

Thus step 1 of Algorithm 1 may be replaced with:

1*: Let a' be a $\emptyset$-maximal strategy for N.

The resulting algorithm, 1*, satisfies

THEOREM 8. *A strategy is optimal if and only if it can be generated by Algorithm* 1*.

Sidney correctly observed that the key to Algorithms 1 and 1* is steps 1 and 2. He gives algorithms which are specializations of Algorithms 1 and 1* to constraints in the form of parallel chain networks, parallel networks, job modules and rooted trees, all of which generalize readily to the class of problems considered in this paper. However, he does not give a computational method for accomplishing steps 1 and 2 when the constraints form an arbitrary partial ordering. It is to this topic that we now turn.

The following algorithm, 1′, is an implementation of Algorithm 1*. In Algorithm 1′, N is the current initial set, K is the set of $\emptyset$-maximal strategies on N, U is the value for $\emptyset$ on K, H is the subset of K having the same set as a', and α is the strategy being constructed.

ALGORITHM 1′.

0. Set $N = S$, $\alpha = \wedge$, $K = \wedge$, $H = \wedge$ and $U = 0$.
1. If $N = \wedge$ and $K = \wedge$, stop. α is the strategy found by Algorithm 1′.
2. If $N = \wedge$ and $K \neq \wedge$, go to 6.
3. Order N according to $\emptyset$, highest first. Let $n \in N$ have the largest $\emptyset$ in N.
4. If n is feasible,
   a) If $K = \wedge$, $K \to \{n\}$, $U = \emptyset(n)$, $N \to N - \{n\}$, go to 1.
   b) If $K \neq \wedge$, if $\emptyset(n) = U$, $K \to K \cup \{n\}$, $N \to N - \{n\}$, go to 1 if $\emptyset(n) < U$, go to 6.
5. If n is not feasible, let $B \subset N$ be the set of immediate predecessors to n. $N \to N - \{n\} \cup \{bs \mid b \in B\}$. Go to 1.
6. Take any element $a' \in K$. $H \to \{a'\}$, $K \to K - \{a'\}$, $U = 0$.
7. a. If $K = \wedge$, go to 8.
   b. Choose $f \in K$.
   c. If $T(f) = T(a')$, then $H \to H \cup \{f\}$.
   d. $K \to K - \{f\}$. Return to 7a.
8. Order H according to F, lowest first. Let a satisfy $F(a) \leqq F(a^*)$ for all $a^* \in H$.
9. Append a to the end of α.
10. $N \to N - T(a)$. Return to 1.

To see the connection between Algorithm 1′ and Algorithm 1, note that steps

2, 3, 4, 5 and 6 implement step 1 of Algorithm 1, and that steps 7 and 8 implement step 2. The crucial step is 5, which is justified by Lemma 5.

Since Algorithm 1′ is only an implementation of Algorithm 1, we have

THEOREM 9. *A strategy is optimal if and only if it can be generated by Algorithm* 1′.

5. Garey reduction theorems and a hybrid algorithm. While Algorithms 1, 1* and 1′ have the comforting property that a strategy is optimal if and only if it can be found by each algorithm, this does not guarantee that these algorithms are quick to find the optimal strategy. In particular the material of Section 4 is predicated on searching over all strategies, of which there would be $n!$ if they all were feasible. If there were no constraints, Theorem 3 suggests that a very simple sort on $\varnothing$ over searches would yield an optimal strategy, and in fact every optimal strategy. This suggests that some preliminary algorithm might be used to process parts of the problem which have a simple topology, before turning the problem over to the all-purpose, but probably slower, Algorithm 1.

One approach to this end is given in a recent paper by Garey (1973). Garey's treatment is limited to the second example of Section 1. He gives an algorithm which finds an optimal strategy, and is easily modified to find every optimal strategy, for problems where the partial ordering graph has an immediate graph C^- which form a forest. Additionally Garey's algorithm reduces every problem that has a general partial ordering restriction by eliminating constraints and by requiring certain searches to be conducted in a particular order, thus allowing them to be treated as a unit in finding the optimal strategy. The purpose of this section is to show that Garey's reduction theorems and reduction algorithm apply to the whole class of problems developed in Section 2.

A search is called *terminal* iff it has no successors, and *initial* if it has no predecessors. A search s is a *maximal successor* of search s_1 iff it is an immediate successor of s_1 and satisfies, if s' is any immediate successor of s_1, $\varnothing(s) \geqq \varnothing(s')$. [For readers comparing this treatment with Gareys, note that Garey's R satisfies $R(s) = -\varnothing(s)$.] A search is a *minimal predecessor* of search s_1 iff it is an immediate predecessor and satisfies, if s' is any immediate predecessor of s_1, $\varnothing(s') \geqq \varnothing(s)$.

THEOREM 10. *For any problem of the class considered here that has an optimal strategy, let t_i be a nonterminal search having only terminal successors. If t_j is a maximal successor of t_i satisfying $\varnothing(t_j) \geqq \varnothing(t_i)$ and t_j has no other immediate predecessors, then there is an optimal solution in which the subsequence $t_i t_j$ occurs.*

PROOF. Let σ be an optimal strategy. Since σ contains every search somewhere in σ, and each of the successors of t_i, say $t_1^i t_2^i, \cdots$, including t_j occur in σ after t_i, let, without loss of generality

$$\sigma = a_0 t_i\, a_1 t_1^i\, a_2 t_2^i \cdots a_r t_j a_{r+1}$$

where every a_k except possibly a_{r+1}, contains no successor of t_i. Then every nonempty a_k, except a_{r+1} and a_0, is interchangeable with t^i_{k-1} and $t_k{}^i$. Therefore, using the optimality of σ if a_k is nonempty $\varnothing(t^i_{k-1}) \geqq \varnothing(a_k) \geqq \varnothing(t_k{}^i)$. If a_k is empty, t^i_{k-1} and $t_k{}^i$ are exchangeable and again by the optimality of σ $\varnothing(t^i_{k-1}) \geqq \varnothing(t_k{}^i)$. Therefore

$$\varnothing(t_i{}^1) \geqq \varnothing(a_2) \geqq \varnothing(t_2{}^i) \geqq \cdots \geqq \varnothing(t_j)$$

(where empty a_i's can be dropped from the above string of inequalities).

Since t_j is maximal among successors to t_i,

$$\varnothing(t_j) \geqq \varnothing(t_1{}^i)\,,$$

so equality obtains throughout the above expression.

$$\sigma' = a_0 t_i\, a_1 t_j t_1{}^i\, a_2 t_2{}^i \cdots a_r a_{r+1}$$

is a strategy, and Theorem 2 implies

$$F(\sigma) = F(\sigma')\,.$$

Now if a_1 is empty the theorem is proved. If not, it is exchangeable with both t_i and t_j. Then

$$\varnothing(t_i) \geqq \varnothing(a_1) \geqq \varnothing(t_j)$$

by Theorem 2. Now $\varnothing(t_j) \geqq \varnothing(t_i)$ by assumption, so equality obtains in the above. Hence

$$F(\sigma) = F(a_0\, a_1 t_i t_j t_1{}^i\, a_2 t_2{}^i \cdots a_r a_{r+1})$$

and the theorem is proved by optimality of σ. □

THEOREM 11. *Let t_j be a terminal search having an immediate predecessor t_i such that $\varnothing(t_i) > \varnothing(t_j)$. Consider the modified problem which is identical to the given problem except that the constraint graph C of the modified problem is formed from the original constraint graph by replacing the constraint from t_i to t_j by a constraint from each immediate predecessor of t_i to t_j. Then every optimal solution to the modified problem is also an optimal solution to the original problem.*

PROOF. Let σ be an optimal solution to the modified problem. Suppose that t_j precedes t_i in σ. Then we can write $\sigma = (a_0 t_j a_1 t_i a_2)$, where a_i's may be empty for $i = 0, 1, 2$. Suppose a_1 is not empty. All predecessors of t_j must be in a_0 since σ is a solution. Hence all predecessors of t_i are in a_0, also. Finally, since t_j is terminal, all predecessors of a_1 are in a_0. Hence t_j and a_1 are interchangeable, and a_1 and t_i are interchangeable. Then

$$\varnothing(t_j) \geqq \varnothing(a_1) \geqq \varnothing(t_i)\,.$$

If a_1 is empty, t_j and t_i are interchangeable, leading to

$$\varnothing(t_j) \geqq \varnothing(t_i)$$

by the optimality of σ. But these inequalities are impossible by the assumption of the theorem that $\varnothing(t_j) < \varnothing(t_i)$. Hence t_i precedes t_j in σ. Hence σ is also a solution to the original problem.

Let σ_R be an optimal solution to the original problem. Since the original problem is the more restricted,

$$F(\sigma_R) \geqq F(\sigma) .$$

Then $F(\sigma) = F(\sigma_R)$ and σ is optimal for the original problem. □

The following theorems are duals to Theorems 10 and 11.

THEOREM 12. *For any problem of the class considered here which has an optimal strategy, let t_j be a noninitial search having only initial predecessors. If t_i is a minimal predecessor of t_j satisfying $\emptyset(t_i) \leqq \emptyset(t_j)$ and t_i has no other immediate successors, then there is an optimal strategy in which the strategy $t_i t_j$ occurs.*

THEOREM 13. *Let t_j be a terminal search having an immediate predecessor t_i such that $\emptyset(t_i) > \emptyset(t_j)$. Consider the modified problem which is identical to the given problem except that the constraint graph C of the modified problem is formed from the original constraint graph by replacing the constraint that t_i precede t_j by constraints that t_i precede each immediate successor of t_j. Then every optimal solution for the modified problem is also a solution to the original problem.*

The proofs of Theorems 12 and 13 are the same as those of Theorems 10 and 11, respectively, with the sense of each constraint reversed, each inequality reversed, and each strategy reversed.

Garey then proposes a reduction algorithm which includes steps (a) to (g) and (i) below.

ALGORITHM 2.

Step (a). Select a connected component, containing at least one constraint from the current reduced precedence graph. If none exists, go to step (i).

Step (b). Choose any nonterminal task t_i', having only terminal immediate successors, from the current reduced version of the component under consideration. If no such task exists, go to step (e).

Step (c). Find a maximal successor t_j' of t_i'. If $\emptyset(t_j') < \emptyset(t_i')$, go to step (d). Otherwise reduce the component by deleting t_j' and the constraint from t_i' to t_j', and replace t_i' by a new strategy $[t_i', t_j']$. If the new task is terminal, go to step (b). Otherwise repeat step (c).

Step (d). For each immediate successor t_k' of t_i', replace the constraint t_i' to t_k' by a constraint from the immediate predecessor of t_i' to t_k'. Go to step (b).

Step (e). Choose any noninitial task t_j', having only initial immediate predecessors, from the current reduced version of the component under consideration. If no such task exists, go to step (h).

Step (f). Find a minimal predecessor t_i' of t_j'. If $\emptyset(t_j') < \emptyset(t_i')$, go to step (g). Otherwise reduce the component by deleting t_i' and the constraint from t_i' and t_j' and replace t_i' by a new strategy $[t_i', t_j']$. If the new strategy is initial, go to step (e). Otherwise repeat step (f) with $[t_i', t_j']$ acting as the strategy t_j'.

Step (g). For each immediate predecessor t_k' of t_j', replace the arc from t_k' to t_j' by an arc from t_k' to the immediate successor of t_j'. Go to step (e).

Step (h). Perform Algorithm 1′ on the resulting reduced component, but replace step 9 of Algorithm 1 with the following:

9′; Bracket a and treat it as a single strategy.

Step (i). Let $t_1', t_2', \cdots, t_m'$ denote the remaining strategies in the completely reduced precedence graph. Order them as $t'_{k_1}, t'_{k_2}, \cdots, t'_{k_m}$ so that $\varnothing(t'_{k_i}) \geqq \varnothing(t'_{k_{i+1}})$, for all i, $1 \leqq i \leqq m-1$. Removing the brackets from this sequence results in an optimal solution to the original problem.

This hybrid algorithm completely reduces a forest without using step (h) and reduces the work of Algorithm 1′ in an arbitrary partially ordered graph. Garey concludes his investigation by saying that in the partially-ordered case "the proper choice may depend somehow on the overall likelihood of success for the complete set of tasks or certain large subsets thereof, a nonlocal property which may be difficult to use in an efficient algorithm" (Garey (1973), page 55). We believe that the material of Section 4 constitutes the nonlocal results sought by Garey.

6. Conclusion. The results of Section 4 through Theorem 8, but excluding Algorithm 1′, are available already for the first example in the paper of Sidney (1975). A development leading to Lemma 7 is given for the second example in Simon and Kadane (1975). Similarly the results of Section 5 are available in the paper of Garey (1973) for the second example. This paper then unifies the theory, and adds the application of each of these results to the other members of the class proposed in Section 2.

No monotonicity constraints, such as those discussed by Kadane (1968, 1969) are required for the first example. Because of that, this paper suggests an algorithm for finding the best strategy in the parallel constraint case, by dividing the searches of each box into strategies on E-sets as follows: ($\mathscr{S}$ is the current set of strategies on E-sets, and $\sigma_{i,j}$ is the jth search of box i).

(a) Set $\mathscr{S} = (\sigma_{i,1}, \sigma_{i,2})$, $h = 3$.

(b) Suppose $\mathscr{S} = (B_1, B_2, \cdots, B_k)$. If $\varnothing(B_{k-1}) \geqq \varnothing(B_k)$ then go to step (c). If not then join B_{k-1} and B_k, and return to the start of (b). If $\mathscr{S}$ has only one search, go to step (c).

(c) Set $\mathscr{S} = (\mathscr{S}, \sigma_{i,h})$, $h = h + 1$, return to (b).

Because each box can be searched infinitely many times, some conditions must be imposed to ensure that the algorithm above will terminate. Once each box has been treated with this algorithm, the best search is found by ordering resulting strategies according to $\varnothing$, highest first. The resulting search will be feasible. The regularity conditions imposed in Kadane (1968), that p_{ij}/c_{ij} be nonincreasing in j for each i, is thus seen to be the condition that each E-set consist of only a single element.

In this sense Theorem 4 generalizes the result proposed, but not proved, in Kadane (1968).

REFERENCES

[1] ADOPHSON, D. and HU, T. C. (1973). Optimal linear ordering. *SIAM J. Appl. Math.* **25** 403–423.

[2] BELLMAN, R. (1957). *Dynamic Programming.* Princeton Univ. Press.

[3] BERGE, C. (1962). *The Theory of Graphs and Its Applications.* Wiley, New York. (Translation published by Methuen, London.)

[4] BERGE, CLAUDE (1973). *Graphs and Hypergraphs.* North-Holland, Amsterdam.

[5] BLACK, W. (1965). Discrete sequential search. *Information and Control* **8** 156–162.

[6] DEAN, B. V. (1966). Stochastic networks in research planning. In *Research Program Effectiveness.* (Yovits et al., eds.) Chapter 12. Gordon and Breach, New York.

[7] DENBY, D. C. (1967). Minimum downtime as a function of reliability and priority in a component repair. *J. Indust. Engrg.* **18** 436–439.

[8] GAREY, M. R. (1973). Optimal task sequencing with precedence constraints. *Discrete Mathematics* **4** 37–56.

[9] GREENBERG, H. (1964). Optimum test procedure under stress. *Operations Res.* **12** 689–692.

[10] HALL, G. J., JR. (1976). Sequential search with random overlook probabilities. *Ann. Statist.* **4** 807–816.

[11] HORN, W. A. (1972). Single-machine job sequencing with treelike precedence ordering and linear delay penalties. *SIAM J. Appl. Math.* **23** 189–202.

[12] JOYCE, W. B. (1971). Organization of unsuccessful R. and D. programs. *IEEE Trans. Engrg. Management* **18** 57–65.

[13] KADANE, JOSEPH B. (1968). Discrete search and the Neyman-Pearson lemma. *J. Math. Anal. Appl.* **22** 156–171.

[14] KADANE, JOSEPH B. (1969). Quiz show problems. *J. Math. Anal. Appl.* **27** 609–623.

[15] KADANE, JOSEPH B. (1971). Optimal whereabouts search. *Operations Res.* **19** 894–904.

[16] KADANE, JOSEPH B. (1975). A characterization of the Rau class of sequential problems. Unpublished mimeo, Carnegie-Mellon Univ.

[17] MATULA, D. (1964). A periodic optimal search. *Amer. Math. Monthly* **71** 15–21.

[18] MITTEN, L. G. (1960). An analytic solution to the least cost testing sequence problem. *J. Indust. Engrg.* **11** 17.

[19] RAU, JOHN G. (1971). Minimizing a function of permutations of n integers. *Operations Res.* **19** 237–240.

[20] SIDNEY, J. B. (1975). Decomposition algorithms for single-machine sequencing with preference relations and deferral costs. *Operations Res.* **23** 283–298.

[21] SIMON, H. and KADANE, J. B. (1975). Optimal problem-solving search: all or none solutions. *Artificial Intelligence* **6** 235–247.

[22] SMITH, W. E. (1956). Various optimizers for single-stage production. *Naval Res. Logist. Quart.* **3** 59–66.

[23] STAROVEROV, O. (1963). On a searching problem. *Theory. Probability Appl.* **8** 184–187.

[24] STONE, L. D. (1975). *Theory of Optimal Search.* Academic Press, New York.

DEPARTMENT OF STATISTICS
CARNEGIE-MELLON UNIVERSITY
SCHENLEY PARK
PITTSBURGH, PENNSYLVANIA 15213