NPS-OR-95-003

NAVAL POSTGRADUATE SCHOOL
Monterey, California

BRANCH AND BOUND METHODS FOR SEARCH PROBLEMS

by

Alan R. Washburn

April 1995

Prepared for: Naval Postgraduate School
Monterey, CA 93943

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943-5000

Rear Admiral T. A. Mercer
Superintendent

Harrison Shull
Provost

This report was prepared for the Naval Postgraduate School, Monterey, CA.

Reproduction of all or part of this report is authorized.

This report was prepared by:

[illegible]

ALAN WASHBURN
Professor of Operations Research

Reviewed by:

[illegible]

PETER PURDUE
Professor and Chairman
Department of Operations Research

Released by:

[illegible]

PAUL J. MARTO
Dean of Research

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE April 1995	3. REPORT TYPE AND DATES COVERED Technical

4. TITLE AND SUBTITLE Branch and Bound Methods for Search Problems	5. FUNDING NUMBERS
6. AUTHOR(S) Alan R. Washburn	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943	8. PERFORMING ORGANIZATION REPORT NUMBER NPS-OR-95-003
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Branch and bound methods for search problems are considered, including hybrid methods that employ multiple bounds. Computational results are reported.

14. SUBJECT TERMS Branch and Bound, Search	15. NUMBER OF PAGES 35
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

Branch and Bound Methods for Search Problems

Alan R. Washburn

Department of Operations Research
Naval Postgraduate School
Monterey, CA 93943

1. Introduction

This paper summarizes computational experience with branch-and-bound (B+B) algorithms for solving two search problems. B+B methods often involve a quality/quantity tradeoff in calculating bounds, an aspect that will be explored. The results may therefore be of some general interest. Additionally, the two search problems are of inherent interest in themselves. The first (section 2) involves search for a moving target by one searcher, and the second (section 3) involves search for a stationary target by multiple searchers with different capabilities.

2. Path-constrained Search for a Moving Target

2.1 Preliminaries

The target track is $X = (X_1, \ldots, X_T)$ where $X_i \in C$ represents the position of the target at time i. C is a finite set of cells, and the single searcher as well as the target must at all times be in one of them. The positive integer T is the fixed amount of time available for search. If the searcher's track is $\sigma = (\sigma_1, \ldots, \sigma_T)$, then the probability of not detecting track X is some given function $\eta(X, \sigma)$, and the searcher's object is to minimize the nondetection probability $\sum_X f(X)\eta(X,\sigma)$, where $f(\ )$ is a given probability mass function. In other words, the probability law governing the target's motion is assumed known.

If there were (say) $|C| = 9$ cells and $T = 10$ detection opportunities, there would in principle be 9^{10} possible tracks, too many to permit even evaluating the objective function, much less optimizing it. Since interesting problems can be even larger, special structures must be imposed to permit optimization. Essentially all work to date has been based on the assumptions that

$$\eta(X, \Psi) = \exp(-Z(X, \Psi)), \tag{1}$$

and that

$$Z(X,\Psi) = \sum_{t=1}^{T} W(X_t,t)\Psi(X_t,t) = \sum_{t=1}^{T} \sum_{x\in C} W(x,t)\Psi(x,t) \tag{2}$$

$\Psi(X_t, t)$ being an indicator function for the event $(X_t = \sigma_t)$; i.e. $\Psi(x, t) = 1$ if and only if $\sigma_t = x$. The Ψ and σ notations for a searcher path are equivalent; each will be used in the sequel when convenient. The search effectiveness function $W(x, t)$ is assumed to be non-negative for all $x\in C$, $1 \le t \le T$. Formulas (1) and (2) include the important instance where $W(\cdot, \cdot)$ is a constant W, in which case $Z(X, \Psi)/W$ is the number of times N that the searcher and the target occupy the same cell. Letting $QS = \exp(-W)$, $\eta(X, \Psi)$ is then $(QS)^N$. Thus QS can be identified as the probability of overlooking the target even when the correct cell is searched. The overlook probability can be made to depend on time and location through the function $W(\cdot, \cdot)$, but the independence assumption is inherent in the exponential detection function.

The usual way of constraining the searcher's path is to fix the searcher's position to be σ_0 at time 0 and then require that a searcher in cell x at time t must proceed to cell in $S(x, t)$ at time $t + 1$. The set $S(x, t)$ is a subset of C corresponding to the "Forward neighbors" of x; $x\in C$, $0 \le t < T$. It is also useful to define $S^*(x, t)$ to be the set of cells at time $t - 1$ from which it is possible to be in x at t, so for $x\in C$ and $0 \le t < T$, $S^*(x, t + 1) =$

$\{y | x \in S(y, t)\}$. For $x \in C$, $y \in C$, and $1 \le t < T$, let $u(x, y, t)$ be 1 if the searcher visits x at t and y at $t + 1$. Then the problem of minimizing the nondetection probability can be stated as the nonlinear programming problem NLP1, with $\tau = 0$:

$$\min E(\eta(X, \Psi))$$

subject to

$$\sum_{x \in S(\sigma_\tau, \tau)} \Psi(x, \tau+1) = 1 \quad \text{and} \quad \Psi(x, \tau+1) = 0 \ \text{ for } x \notin S(\sigma_\tau, \tau) \tag{3}$$

$$\sum_{y \in S(x,t)} u(x,y,t) = \Psi(x,t); \quad \tau < t < T, \quad x \in C \tag{4}$$

$$\sum_{y \in S^*(x,t+1)} u(y,x,t) = \Psi(x,t+1); \quad \tau < t < T, \quad x \in C \tag{5}$$

$$\Psi(x, t) \text{ and } u(x, y, t) = 0 \text{ or } 1, \ x \in C, y \in C, \tau < t < T. \tag{6}$$

Constraint (3) requires the searcher to start at some cell in $S(\sigma_\tau, \tau)$, while constraints (4) – (6) require him to move from one legal cell to another. If $\tau > 0$, it should be understood that $\sigma_0, \ldots, \sigma_\tau$ is specified to be a legal beginning of a searcher track, with only the part after τ to be optimized. The $E(\ )$ in the objective function denotes expected value, so the objective function is a weighted sum of exponentials of the form (1), one term for each potential target track.

2.2 Branch and Bound for NLP1

The following basic B+B algorithm is a slight modification of Stewart's [7]. It is assumed that the bound computed in step 2 reduces to the exact nondetection probability when the searcher's entire path is specified ($\tau = T$). $K(\tau)$ is "the set of continuations yet to be explored," and q^* is "the best nondetection probability yet found."

1) Set $\tau = 0$, initialize q^* to be any number exceeding 1, and let $\sigma_\tau = \sigma_0$.

2) Obtain a lower bound q on the nondetection probability, subject to the searcher's path following $\sigma_0, \ldots, \sigma_\tau$ up to time τ.

3) If $q < q^*$ and $\tau < T$, then Branch; i.e. let $K(\tau + 1) = S(\sigma_\tau, T)$, increment τ, let σ_τ be any cell in $K(\tau)$, and return to 2. Otherwise, the current path has now been fathomed.

4) If $q < q^*$ and $\tau = T$, let $q^* = q$ and save the path $\sigma_1, \ldots, \sigma_T$.

5) If $\tau = 0$, stop. The last saved path is optimal and q^* is its nondetection probability.

6) Delete σ_τ from $K(\tau)$. If $K(\tau)$ is now empty, decrement τ and return to step 4. If not, let σ_τ be any cell in $K(\tau)$ and return to step 2.

If the bound in step 2 is too loose, then no fathoming will occur. The object is to find relaxations of NLP1 that are easy to solve, but still provide sharp bounds. Fortunately NLP1 has several relaxations that considerably simplify it. Three of the most important are described below.

Convex. If binary constraints (6) are replaced by simple non-negativity requirements, NLP1 is a convex program for which Kuhn-Tucker conditions are necessary and sufficient for optimality.

Linear. Since $e^{-Z} > 1 - Z$ for $Z \geq 0$, a lower bound can be obtained by minimizing $1 - E(Z(X, \Psi))$ which is equivalent to maximizing $\sum_{t=1}^{T} E\big(W(X_t,t)\Psi(X_t,t)\big)$. The interpretation is that one is maximizing the mean number of detections. This is a longest route problem where the reward for visiting (x, t) is $W(x, t)\text{Prob}(X_t = x)$, a relatively simple optimization problem.

The bounds obtained by this relaxation will be sharp as long as the maximized objective function is small, so one can expect NLP1 to be easiest to solve when the prospects of detection are slim.

A slight sharpening of the bound achieved by this method is possible if $Z(X, \Psi)$ is always an integer multiple of some quantity $\Delta > 0$, as will be the case if $W(x, t)$ is always a multiple of Δ. In that case let $f = (1 - \exp(-\Delta))/\Delta$, which is smaller than 1, or otherwise let $f = 1$. Then $e^{-Z(X, \Psi)} \geq 1 - fZ(X, \Psi)$. The rewards in the longest route problem can be multiplied by f, which will result in a sharper bound.

<u>Distribution of Effort (DOE)</u>. Constraints (3) – (5) are network constraints, the structure of which can already be exploited, but a further simplification results if the searcher is permitted to occupy any cell at time t that is reachable from the last constrained cell σ_τ. The set S_t of such cells can be obtained from the recurrence $S_{t+1} = \bigcup_{x \in S_t} S(x,t)$; $t > \tau$, with $S_\tau = \{\sigma_\tau\}$. Then (3) – (5) can be replaced by

$$\sum_{x \in S_t} \Psi(x,t) = 1; \quad \tau < t \leq T. \tag{7}$$

Note that (7) is already implied by (3) – (5), so that the replacement is truly a relaxation. This relaxation has the advantage that the u-variables disappear from the formulation. A further relaxation could be obtained by substituting C for S_t, but there seems to be no computational advantage in doing so.

2.3 The Markov Specialization

The three relaxations described in section 2.2 cannot in themselves make B+B a practical technique as long as evaluation of the objective function still requires enumeration of all possible target tracks. Consider a problem with 9 cells and 10 time periods. The Linearity relaxation requires as data only the marginal probabilities $P(X_t = x)$, a total of 9×10 numbers, to determine a candidate solution $\Psi(x, t)$. Evaluating $E(\eta(X, \Psi))$, however, would still require evaluation of a probability and an exponential for each track, a potentially explosive amount of computational effort. If NLP1 is to be solvable as a practical matter, either the number of tracks must be constrained or some structure must be imposed that obviates the need to enumerate them. Tactical decision aids have been based on the idea that target motion can be modeled with a track population on the order of 1000, so proceeding on the former course would be a reasonable choice. Nonetheless, all B+B computational work to date has been based on the Markov specialization of NLP1, a structural assumption that makes it unnecessary to enumerate paths. The rest of this report also concerns that specialization.

The main advantage of the Markov assumption is that it permits the operation of the FAB (Forward and Backward) algorithm. The FAB algorithm uses the two functions

$$P(\Psi, x, t) = \text{Prob}(X_t = x \ \underline{\text{and}} \text{ no detection before } t), \tag{8}$$

and $$Q(\Psi, x, t) = \text{Prob}(\text{no detection after } t, \underline{\text{given}}\ X_t = x), \tag{9}$$

as well as the relation that

$$\text{ND} \equiv \text{Prob(no detection)} = \sum_x P(\Psi,x,t)\exp(-W(x,t)\Psi(x,t))\,Q(\Psi,x,t). \tag{10}$$

Formula (10) is valid for $t = 1, \ldots, T$ even without the Markov assumption, but it is especially useful in the Markov case because

a) $P(\Psi, x, t)$ is easily calculated given $\Psi(y, u)$ for $y \in C$ and $u < t$

b) $Q(\Psi, x, t)$ is easily calculated given $\Psi(y, u)$ for $y \in C$ and $u > t$

c) neither $P(\Psi, x, t)$ nor $Q(\Psi, x, t)$ depends on $\Psi(y, t)$ for any $y \in C$.

The import of c) is that $\Psi(\cdot, t)$ can be changed to increase the objective function as long as the searches before and after t remain feasible, thus permitting an iterative (FAB) algorithm for gradually decreasing the objective function. FAB requires repeated evaluations of $P(\cdot, \cdot, \cdot)$ and $Q(\cdot, \cdot, \cdot)$, so a) and b) are also important. See [2, 10] for the details of these evaluations. The main use of FAB has been in computing an improving sequence of search plans for NLP1 and its various relaxations. The limiting FAB search plans might reasonably be termed "locally optimal", since a certain class of small perturbations cannot improve the objective function.

It turns out that each search plan in the FAB sequence has associated with it a global lower bound on the nondetection probability; this is shown in Washburn [8] for the DOE relaxation of NLP1, and in Appendix A of this report for NLP1 itself. The availability of these lower bounds makes FAB potentially useful in B+B. Specifically, one might use FAB in step 2 to find a locally optimal extension $\sigma_{\tau+1}, \ldots, \sigma_T$, with an associated lower bound q as well as nondetection probability $\hat{q}$. B+B has no use for $\hat{q}$ as stated, but step 2 could be extended as follows: "If $\hat{q} < q^*$, let $q^* = \hat{q}$ and save the path $\sigma_1, \ldots, \sigma_T$." This extension is potentially valuable in making fathoming easier in step 3. Thus FABing could be viewed as a method of producing lower bounds that has the side benefit of reducing q^* quickly. It could also be viewed as redundant effort, since each of the path extensions produced by FAB will eventually be considered (implicitly or explicitly) by the B+B procedure in any case. Plainly the important question is whether locally optimal extensions are worth the computational trouble.

The Markov specialization also permits an effective generalization of the Linear bound discussed earlier. Suppose that the searcher's path up to time τ is fixed, let $Z_\tau = \sum_{t=1}^{\tau} W(X_t,t)\Psi(X_t,t)$, and $Z_+ = \sum_{t=\tau+1}^{T} W(X_t,t)\Psi(X_t,t)$. Then $Z(X, \Psi) = Z_\tau + Z_+$, with Z_τ representing the past and Z_+ the future. Since the target's motion is Markov, Z_τ and Z_+ are independent when X_τ is given, and therefore

$$E\left(e^{-Z_\tau} e^{-Z_+} \middle| X_\tau = x\right) = E\left(e^{-Z_\tau} \middle| X_\tau = x\right) E\left(e^{-Z_+} \middle| X_\tau = x\right). \tag{11}$$

Let
$$P^+(x,\tau) = E\left(e^{-Z_\tau} \middle| X_\tau = x\right) \mathrm{Prob}(X_\tau = x). \tag{12}$$

$P^+(x, \tau)$ can be obtained from the FAB function $P(\Psi, x, \tau)$ by multiplying by $\exp(-W(x, \tau)\Psi(x, \tau))$; the + superscript is intended to convey the idea that the effect of search at time τ is included. Then the nondetection probability is

$$\mathrm{ND} = E\left(e^{-Z_\tau + Z_+}\right) = \sum_{x \in C} P^+(x,\tau) E\left(e^{-Z_+} \middle| X_\tau = x\right). \tag{13}$$

But $e^{-Z_+} \geq 1 - fZ^+$, where $f \leq 1$ is the factor introduced earlier in describing the Linear relaxation. Therefore

$$\text{ND} \geq \sum_{x \in C} P^+(x,\tau) - f \sum_{t=\tau+1}^{T} \sum_{x \in C} E\big(W(X_t,t)\Psi(X_t,t) \big| X_\tau = x\big) P^+(x,\tau). \tag{14}$$

But $$E\big(W(X_t,t)\Psi(X_t,t) \big| X_\tau = x\big) = W(\sigma_t,t)\text{Prob}\big(X_t = \sigma_t \big| X_\tau = x\big). \tag{15}$$

Now let $$R(\sigma_t,t) \equiv f \sum_{x \in C} W(\sigma_t,t)\text{Prob}\big(X_t = \sigma_t \big| X_\tau = x\big) P^+(x,\tau) \tag{16}$$

be the searcher's reward for visiting cell σ_t at time t. A lower bound on ND can now be obtained by minimizing the right hand side of (14), which amounts to finding the search path continuation that maximizes the total reward at times $\tau + 1, \ldots, T$, a relatively easy longest path optimization problem.

Appendix B describes one more bound that is worthy of consideration if the Markov motion is ergodic. This Ergodic bound is based on bounding the factors in (16), thus obviating the need for longest path computations. The Ergodic bound can be expected to be less sharp than the Linear bound obtained from (16), but more easily computed.

Obviously there are a great many ways of bounding NLP1, particularly in the Markov specialization. Some methods that have been used in the past are reviewed in the next section.

2.4 Review of Previous Computational Experience with B+B

All discussions in this section concern the Markov specialization. The Linear, Convex, and DOE relaxations are as described in section 2.2.

Stewart [7] was the first author to consider B+B for NLP1. He considers the Linear relaxation, but finds the resulting bounds "...too weak to be usefully effective...". He is then led to the DOE relaxation, employing the FAB algorithm to "solve" it. He acknowledges that the resulting B+B solutions are potentially non-optimal because FAB solutions of the DOE relaxation are themselves non-optimal, but rejects making the additional Convex relaxation (FAB solutions would then be optimal) because the resulting bounds are again weak. In Stewart [6], computational results are given for a one-dimensional problem where the searcher has two options (right or left) at each time, and $T = 10$. The true optimal solution was found in 101 out of 105 test problems.

Eagle and Yee [3] use the Convex relaxation of NLP1 to obtain bounds. The relaxation has a nonlinear objective function and network constraints. It is solved by an iterative method where at each stage a linear approximation to the objective function is made. An attractive feature of this method is that each of these minimizations results in a solution feasible in NLP1, thus permitting q^* to be quickly reduced. The procedure is tested on a problem where the searcher moves in a 3×3 grid, having four choices in the center cell or a smaller number on the edges, and $T = 10$. Searcher and target start in opposite corners, the idea being that the searcher must transit to the vicinity of the target before starting to search. Solution times are a few minutes. Solution times are also a few minutes in larger 5×5 and 7×7 problems, the surprising lack of a sharp increase being explained by the fact that searcher and target continue to start in opposite corners.

Martins [5] pursues the idea of using bounds that are easily evaluated, rather than sharp. He uses the Linear relaxation in the form of equations (13) – (16), so calculating a bound takes the form of a longest path problem. The resulting procedure does more branching than the Eagle-Yee procedure on the same test problems, but still has run times that are smaller by a factor of about 4 in problems where $W(x, t) = 1$ for all (x, t); i.e. where the overlook probability is $\exp(-1) = .632$. Martins' procedure is even faster with larger overlook probabilities, an expected result because the Linear relaxation is closer to NLP1 in that case.

Evidently there is something to be said for all three of the relaxations described in section 2.2. In addition, no experiments have yet been made using FAB bounds of the type described in the Appendix. These observations prompt the experiments reported in the next section.

2.5 Results of Experiments

All of the experiments reported here are for $C = \{1, \ldots, N\}$, a one-dimensional set of cells. The target moves right and left with probability .3, or remains stationary with probability .4. In boundary cells where motion would take the target outside C, the target remains stationary instead. The target's initial cell is specified, as is the cell that the searcher must examine at time 1. First the searcher examines the given cell, then each party moves to a new position, then a second search is made at time 2, etc., until finally the last search is made at time T. If the searcher's current position is $x \in C$, then the searcher's next position can be any $y \in C$ such that $|x - y| \leq 1$. The overlook probability $QS \equiv \exp(-W(x, t))$ is constant for all $x \in C$, $1 \leq t \leq T$. Problem 1 has $N = 9$, $T = 10$, $QS = .6$, and both parties starting in cell 5, in which case the optimal nondetection probability ND^* is .26639607 and an optimal searcher track is 5555456654. Problem 1 is large enough to be interesting, but small enough to permit extensive experimentation on a 486 (8MHz) PC.

Four lower bounds are considered:

1) <u>ERGO</u>. The random walk chosen for the target has stationary distribution $(1/N, \ldots, 1/N)$, and $W(x, t)$ is constant, so computation of the ergodic bound as outlined in Appendix B is trivial.

2) <u>MEAN</u>. This is the same bound utilized by Martins, and is named after the fact that the mean number of future detections is maximized.

3) <u>FAB</u>. This is the FAB bound for NLP1 with no relaxations. The subroutine computing the lower bound also returns the feasible nondetection probability associated with the FAB path. The function $Q(\Psi, x, t)$ is initialized to 1.0 for all (x, t), but never reset. Thus each call to FAB starts with the leftover $Q(\ )$ from some previous call.

4) <u>FABC</u>. This is the FAB bound for NLP1 with the Convex and DOE relaxations. FABC and FAB both utilize the same function $Q(\ )$ in the same way.

Table 1 shows the performance of the four bounds on Problem 1 for two initial segments. Segment (5) is an example of a hard bounding problem (since only the starting point is specified), and segment (5, 6, 7, 8, 9) is an example of an easy bounding problem (since the searcher seems determined to move *away* from the target's starting position as fast as possible).

TABLE 1. Relative Performance of Four Bounds

	segment = (5)			segment = (5,6,7,8,9)		
	time (sec)	*lower*	*upper*	*time (sec)*	*lower*	*upper*
ERGO	.1	-.26400	—	.1	.27764	—
MEAN	1.9	.06457	—	1.1	.37523	—
FAB	3.4	.19600	.26726	1.9	.37141	.39079
FABC	5.9	.24108	—	2.4	.38658	—

It can be observed in Table 1 that the bounds are listed in order of difficulty of computation. MEAN takes longer than ERGO because it must solve a longest route problem. FAB takes longer than MEAN because it must solve a longest route problem in addition to FAB computations. FABC does not need to solve a longest route problem because of the DOE relaxation, but nonetheless takes longer than FAB because the Convex relaxation implies a need for logarithms and exponentials. FAB and FABC each do only a single FAB iteration; doing two iterations would double the time required and produce only slightly better bounds.

It may also be observed in Table 1 that MEAN produces a better bound than ERGO. This must be true in general because ERGO operates by bounding terms in the sum computed by MEAN. FABC also produces a better bound than either MEAN or FAB in Table 1, but that is not true in general. Table 2 shows the results of an experiment where the bounds were tested on variations of Problem 1. There are 9322 paths with $1 \leq \tau < 10$ in Problem 1. For each of those paths, the number of instances where one bound is greater than another (excepting ERGO) by at least 10^{-6} is shown in columns 2-4 of Table 2, the code being that instances of the reverse of the column heading are shown in ( ). When $QS = .6$, MEAN beats FAB 7431 times, FAB beats MEAN 537 times, and the two are essentially equal (they must be exactly equal when $\tau = 9$) 9322 – 7431 – 537 = 1354 times.

TABLE 2. Relative Performance of Four Bounds (Cont'd.)

QS	MEAN > FAB	FAB > FABC	MEAN > FABC	ERGO	MEAN	FAB	FABC
.1	9277 (45)	185 (9137)	9186 (136)	44	8	27	18
.6	7431 (537)	14 (7762)	5917 (2039)	104	20	57	20
.9	3256 (1836)	0 (3982)	1920 (3166)	31	3	4	1

All four bounds must in every instance be smaller than the feasible nondetection probability returned by FAB. The last four columns show the average excess of this quantity over the bound, times 10,000. The two variations are cases where the single look detection probability is very small ($QS = .9$) or large ($QS = .1$). The overall situation is complicated, but note that the MEAN bound is surprisingly good, with its only real competition coming from FABC. Other than the "byproduct" of a feasible nondetection probability, FAB has little to recommend it.

Table 3 shows the run times in seconds for the EXHAUST method (exhaustion) and for four B+B procedures on the same three variations of Problem 1. All four B+B procedures begin with a single call to FAB to establish a good initial feasible path. ERGO is fastest when $QS = .6$, followed closely by MEAN. This situation is reversed when QS is small. All procedures except exhaustion solve the problem very quickly when $QS = .9$.

TABLE 3. Run Times for Five Procedures (seconds)

QS	EXHAUST	ERGO	MEAN	FAB	FABC
.1	1.4	1.4	1.3	4.0	4.7
.6	1.4	.6	.8	2.5	2.1
.9	1.4	.1	.1	.1	.1

EXHAUST is competitive in Table 3 except when QS is small. The FORTRAN implementation of EXHAUST exploits the Markov assumption just like the four bounds do, hence its quickness.

2.5.1 Hybrid Procedures

When attempting to fathom a starting segment, it often happens that a computed bound is not quite large enough to prevent branching. In that case it is tempting to compute some other bound, especially a sharper one, on the grounds that a little additional effort may prevent the branch. Consider a class of such hybrid procedures where bound B is calculated if and only if bound A comes within δ of fathoming the segment, where $\delta \geq 0$. Bound B will never be evaluated if $\delta = 0$, but its evaluation becomes more likely as δ increases. When δ is sufficiently large, failure of bound A always results in evaluation of bound B.

To test the usefulness of hybrid procedures, several experiments were made on a larger problem (Problem 2) where $N = 15$, $T = 16$, and both parties start in cell 8. For each hybrid procedure (A, B), five values of δ (0, .005, .02, .045, .08) were tested to determine which one gave the lowest run time. In practice .08 is essentially infinite, since further increase does not change the sequence of evaluations. Results are shown in Table 4, using the code that the best value of δ is shown in (   ) if it is not 0.

In Table 4, the fastest B+B procedure when $QS = .6$ is simply ERGO with no backup, since neither (ERGO, MEAN) nor (ERGO, FABC) ever employs the backup bound (the two computer programs are still distinct, however, so the run times are somewhat different). The B+B based on MEAN is a close runner-up. The run time of FABC when not backed up is 533 seconds (not shown), decreasing to 452 seconds when backed up by MEAN at $\delta = .005$. The (FABC, MEAN) hybrid is not a contender, nor is (FAB, FABC), nor is EXHAUST.

TABLE 4. Run Times (seconds) for Hybrid B+B Algorithms on Problem 2

A	B	QS = .6	QS = .9
EXHAUST	—	1305	1305
ERGO	MEAN	157	12
ERGO	FABC	168	5 ($\delta = .08$)
MEAN	FABC	197	4 ($\delta = .08$)
FAB	FABC	821 ($\delta = .005$)	6 ($\delta = .08$)
FABC	MEAN	452 ($\delta = .005$)	4

When $QS = .9$, the sharpness of the FABC bound comes into play; the last four B+B procedures all place heavy reliance on it and produce short solution times. In order to distinguish between them, three of the four were tested on Problem 3, which has $N = 19$, $T = 20$, and both parties starting in cell 10. Results are shown in Table 5.

TABLE 5. Run Times (seconds) on Problem 3

A	B	QS = .9	QS = .93	QS = .95
FABC	—	111	10	1
ERGO	FABC	117	11	1
MEAN	FABC	136	14	1

Pure FABC has the shortest run times, and run times decrease rapidly as QS increases. The decrease in run times is mostly because the increasing sharpness of the FABC bound requires fewer initial segments to be fathomed. In the case of pure FABC, this number decreases from 34512 to 2580 to 132 as QS increases from .9 to .93 to .95. The other two procedures have slightly smaller segment numbers, but with greater fathoming cost per segment.

One might examine the above calculations and conclude that hybrid procedures are not a good idea, since, in all cases examined, a pure procedure is at least as good as any hybrid. However, hybrid procedures exhibit a robustness that pure procedures lack. The author's favorite is MEAN backed up by FABC. MEAN is chosen, rather than ERGO, for reasons not related to the data presented above. The ERGO bound is based on $\max_j P(\Psi, j, \tau)$, regardless of the location of the searcher, whereas the initial term in the MEAN bound will be $P(\Psi, j, \tau)$ for some j near the searcher. The difference is slight if the searcher starts out near the target, as in all results reported here, but the ERGO bound will be almost useless in the "transit" phase if the searcher must first move to the vicinity of the target. For example, consider the problem where $N = 19$, $T = 20$, $QS = .6$, and where searcher and target start at opposite ends of C. The (MEAN, FABC) procedure solves the problem in a small fraction of a second, whereas (ERGO, FABC) takes over 4 seconds. Thus, MEAN is preferred to ERGO for robustness reasons.

Figure 1 shows performance of the (MEAN, ERGO) hybrid B+B procedure on a variety of problems where there are $N = 19$ cells, with searcher and target each starting in cell 10. The number of time periods T ranges from 14 to 25, and the overlook probability ranges from .60 to .93. The quantity graphed is run time R in seconds. It can be seen that the logarithm of R is a linear function of T. The slope is about .35; R increases by a factor of $10^{.35} = 2.24$ when T is incremented by unity. It is neither surprising nor encouraging that R increases exponentially with T, but somewhat encouraging that the factor of increase is smaller than the maximum number of cells to which each segment can branch (3). Figure 1 makes it clear that this factor of increase is nearly independent of QS, at least over the range considered (the slope actually increases slightly with QS).

Run time R has a surprisingly strong dependence on QS. When $T = 20$, the ratio (R when $QS = .6$)/(R when $QS = .93$) is 732. This ratio would be 1.0 if solutions were found by exhaustion, so the B+B procedure has a strong preference for problems where QS is large.

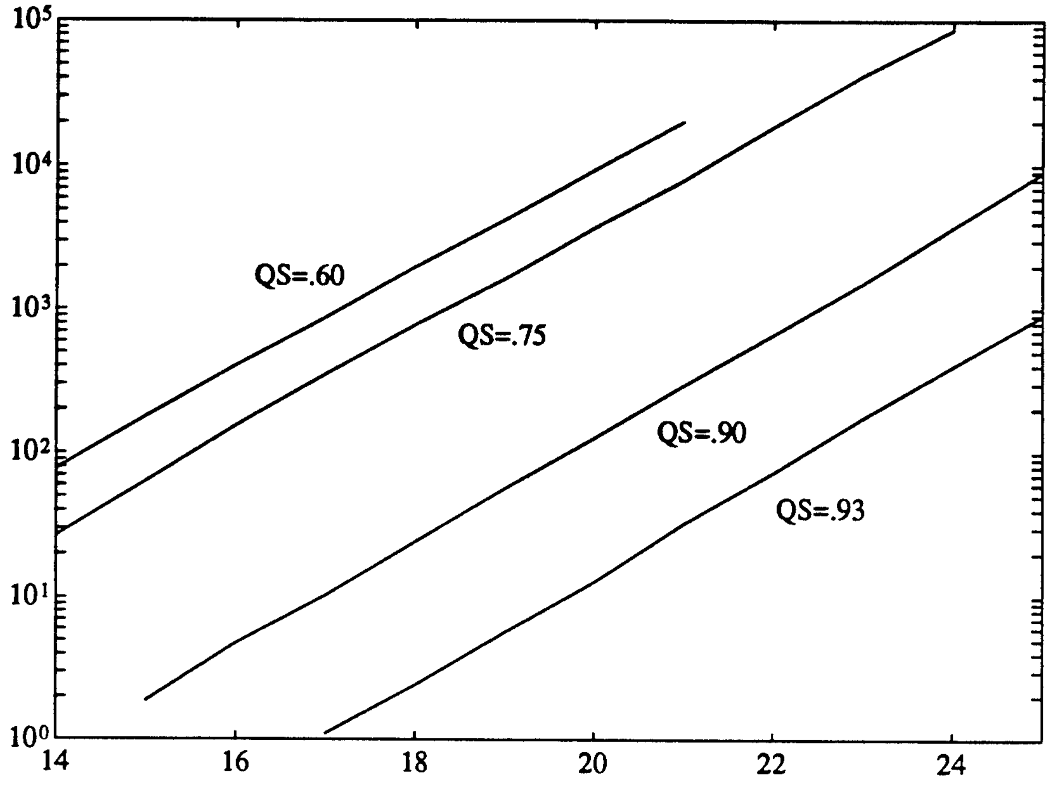

Figure 1. Run time in seconds versus number of time periods for various values of the nondetection probability *QS*

2.5.2 Nearly Optimal Solutions

Any B+B procedure can easily be adapted to find solutions that are nearly optimal, rather than exactly optimal. If step 3 in section 2.2 is modified to begin "If q + d < q^*...", where $d \geq 0$, then the last saved path will be within d of optimality.

Depending on the bounds being employed, permitting d to be positive can make a large reduction in the run time. Figure 2 shows the results of testing the (MEAN, FABC) hybrid under the same circumstances as in Figure 1, except that $QS = .6$ in all cases and d is permitted to be positive. Thus the top curve is the same in both Figures. Very small values of d do not have much effect on run time, but somewhat larger values can reduce it drastically. In the test problem, increasing d from .01 to .02 resulted in decreasing run time by an order of magnitude. Finding solutions that are optimal to within .02 is a much easier problem than finding solutions that are optimal. This tendency continues as d is increased; when $d = .03$, for example, solution requires only 5 seconds (not shown) even when $T = 25$. As d increases, fathoming becomes so easy that only a small number of initial segments needs to be considered.

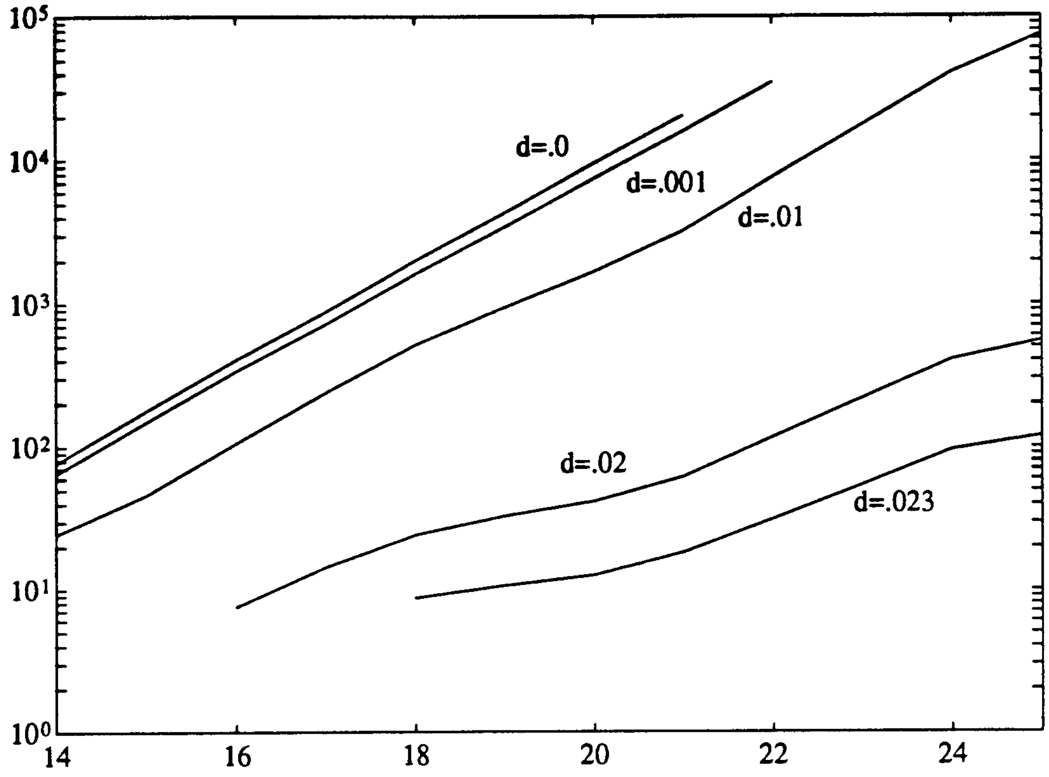

Figure 2. Run time in seconds versus number of time periods for various values of the optimality tolerance *d*

The strong decrease of run time with d is perhaps the best hope for solving search problems that are large enough to be of practical interest. Exact solutions of such problems are unlikely to be possible, but B+B methods can provide solutions that are nearly optimal in a well-defined sense.

3. Search with Multiple Asset Types

3.1 Introduction

It was assumed in section 2 that search was conducted by a single agent who could only be in one place at any given time. Generalization to multiple identical agents is straightforward. Intuitively, multiple agents are equivalent to a single agent with freedom to spread himself out, so one would expect to find that the DOE relaxation is sharp, depending somewhat on starting conditions for the multiple agents. This sanguine view of things should not obscure the accompanying combinatorial explosion, however; the prospect is one of proving that the DOE relaxation is sharp only after considerable computational difficulty.

To avoid this possibility, in this section the multiple agents are assumed to be significantly different. To keep computation times within bounds, only a single time period is considered. Thus the problem considered here is a generalization in one sense, and a restriction in another. To be precise, the problem to be solved is NLP2:

$$\text{minimize} \sum_{j\in C} P_j \exp\left(-y_j\right)$$

where

$$y_j = \sum_{i=1}^{M} W(i,j)x_{ij} \tag{17}$$

$$\sum_{j\in C} x_{ij} \le b_i; \quad i = 1,\ldots,M \tag{18}$$

$$x_{ij} \ge 0 \quad \text{and integer.} \tag{19}$$

The interpretation of variable x_{ij} is "number of type i assets assigned to cell j," with data b_i being the number of type i assets available. Data P_j is the probability that the target is in cell j, and the objective function is the nondetection probability. Use of the exponential function implies that the assets all have independent chances of detection.

NLP2 also has other interpretations. For example, x_{ij} might be "number of assets of type i assigned to kill targets of type j," and the objective function might be "average surviving value in the target set." Alternatively, x_{ij} might be "number of minesweepers of type i assigned to sweeps of type j", and the objective function might be "total number of unswept mines". The essential feature is that assignment of assets to roles (cells) determines a "potential" y_j for each role, with the objective function being a separable, convex function of the potentials.

Although time is not considered explicitly in this section, one could solve NLP2 in period 1, use Bayes Theorem to determine new occupancy probabilities for period 2, conditional on the failure of search in period 1, solve NLP2 again in period 2, etc. The resulting multi-period "myopic" search plan is not necessarily optimal, but is usually not far off. This is the implied mode of operation in Search and Rescue software such as CASIE III (National Association for Search and Rescue (NASAR, PO Box 3709, Fairfax, VA 22038)).

3.2 Bounds

There has been very little work with NLP2. It is an optimization problem with a strictly convex objective function and linear constraints, which would make it relatively easy among nonlinear problems were it not for the requirement that x_{ij} be integral. The B+B strategy employed here will be to fix some of the x_{ij} and optimize with respect to the others. Two bounds will be evaluated:

<u>LINEAR</u>

Let $y_j = f_j + z_j$, where f_j is the fixed part and z_j the free part. Since $\exp(-y_j) \geq \exp(-f_j)\,(1 - z_j)$, a lower bound can be obtained by maximizing $\sum_{j \in C} P'_j z_j$, where $P'_j = P_j \exp(-f_j)$, a linear function of the free variables. All remaining assets of type i

should be assigned to whichever index j maximizes $W(i, j)P'_j$, so evaluation of the bound is trivial.

CONVEX

In this scheme the integer requirement (19) is omitted for free variables, and the resulting convex minimization problem is solved using the method of Washburn [9]. CONVEX bounds are always better than LINEAR bounds, but are harder to compute.

A B+B procedure can be based either on fixing assets or units. In the former case one fixes x_{ij} for all j and certain assets i. In the latter case there are simply $\sum_{i=1}^{M} b_j$ units, each of which is either fixed or not. The asset-based procedure is conceptually simpler because fixed assets are essentially eliminated from the problem, but the unit-based procedure has proved to be superior computationally. Only the unit-based procedure will be described further.

Let $U = \sum_{i=1}^{M} b_i$ be the total number of units present, and order the units from 1 to U, with lower numbered units corresponding to lower numbered asset types. Let τ be a "marker" such that units whose index exceeds τ are free, while the rest are fixed. All are free when $\tau = 0$, and all are fixed when $\tau = U$. If k is a fixed unit, let $F(k)$ be the cell that k is assigned to. $F(k)$ is required to be nondecreasing within each asset type; this requirement implements the assumption that all units of each asset type are identical. For example, suppose $N = 5$, $M = 3$, $b = (2, 3, 4)$, $U = 9$, and $\tau = 4$. Then the "segment" $(F(1), \ldots, F(u)) = (5, 5, 2, 3)$ indicates that both units of the first asset are assigned to cell 5, units of the second asset are assigned to cells 2 and 3, and 5 units, including all 4 of type 3, are unassigned. The next segment will either be (5, 5, 2, 4) or (5, 5, 2, 3, 3), depending on whether (5, 5, 2, 3) is fathomed or not. All B+B procedures, including exhaustion, are implemented within this framework. Each procedure starts with a null segment and terminates when the segment is again null, as in the B+B algorithm of section 2.2. Segments where $\tau = U$ will be called "complete", or "incomplete" if $\tau < U$.

Since the LINEAR bound always results in integer allocations, each attempt to fathom with LINEAR also results in an easily evaluated feasible value that might set a record, as in step 4 of the B+B algorithm in section 2.2. The CONVEX allocations are not necessarily integer, but a feasible value can still be (and is) obtained by rounding them. Thus each attempt to fathom with either LINEAR or CONVEX results in a feasible value, as well as a bound.

3.3 Computational Results

Table 6 shows unit overlook probabilities $q_{ij} \equiv \exp(-W(i,j))$ for the base case. These numbers have been arbitrarily selected in the middle of the interval (0,1). There are 5 cells in the base case with $(P_j) = (30, 40, 100, 10, 100)$. These "probabilities" could be normalized to sum to 1 without affecting the optimal allocation, but are not normed in the results given below. The vector b in the base case is (2, 3, 2, 4, 3) a total of 14 units. The best (x_{ij}) is

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 2 \\ 3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \end{bmatrix}$$

and the associated minimized objective function is 8.38.

TABLE 6. Overlook Probabilities in the Base Case

		cell				
		1	**2**	**3**	**4**	**5**
asset	**1**	.40000	.50000	.90000	.30000	.40000
	2	.20000	.30000	.70000	.30000	.40000
	3	.40000	.30000	.60000	.40000	.40000
	4	.10000	.40000	.60000	.40000	.30000
	5	.50000	.30000	.50000	.50000	.50000

To solve the base case, the B+B procedure employing the CONVEX bound (hereafter simply CONVEX) evaluates 249 segments in a run time of 1 second; only incomplete segments are counted here, since complete segments do not require an approximation. The LINEAR B+B evaluates over 7 million such segments and takes 800 seconds to solve the same problem. Exhaustion evaluates the totality of all 8,505,931 incomplete segments in 600 seconds. The LINEAR procedure turns out to be worse than exhaustion, since at some computational expense it manages to eliminate practically none of the incomplete segments.

The LINEAR procedure is not at its best on problems where $W(i,j)$ is as large as in the base case, since the linear approximation is most accurate when $W(i,j)$ is small. A sequence of variations was therefore constructed using the formula $q_{ij} = 1 - \rho\left(1 - q_{ij}^{0}\right)$, where q_{ij}^{0} is as given in Table 6 and the "shrink factor" ρ is a power of .5.

Results are shown in Table 7. It is evident that LINEAR performs much better on problems where $W(i,j)$ is small, but that the same thing can be said of CONVEX. Experimentation with other starting points has on a few occasions produced problems where LINEAR performs better than CONVEX, but such instances are narrow, rare, and always on problems where the overlook probabilities seem impractically large. Given the availability of a fast way of solving the convex relaxation of NLP2, there seems to be no role for the LINEAR bound. Experience with a hybrid scheme where CONVEX is called only if LINEAR fails to fathom does not change this conclusion. The best B+B method for solving NLP2 appears to be CONVEX.

TABLE 7. Performance of LINEAR and CONVEX on a Sequence of Problems where the Shrink Factor ρ is $(1/2)^i$, $i = 0, \ldots, 6$.

	LINEAR	CONVEX	
i	segments	segments	time (sec)
0	7126977	249	.94
1	1256715	55	.16
2	199480	44	.11
3	17129	99	.22
4	1791	1	0
5	323	1	0
6	91	1	0

Performance of CONVEX on larger problems was tested by increasing b_1 from its baseline value of 2. Figure 3 shows that the run time increases erratically but inexorably as the problem size increases. The largest run time (40 minutes) was with $b_1 = 40$, as expected, but there are several instances where run time decreases dramatically as b_1 is increased. Run time turns out to be much less predictable for the multi-asset problem than for the problem considered in section 2. When $b_1 = 40$ the minimized objective function is only .002, a tiny fraction of the total cell value of 270, so there is little point to increasing b_1 beyond 40. In general, it is hard to make statements about how run time depends on problem size because increasing the number of units will always make the objective function vanish. One might adjust the $W(i, j)$ to prevent this as the number of units increases, but unfortunately the run time depends strongly on the magnitudes of these coefficients, as well as on the number of units.

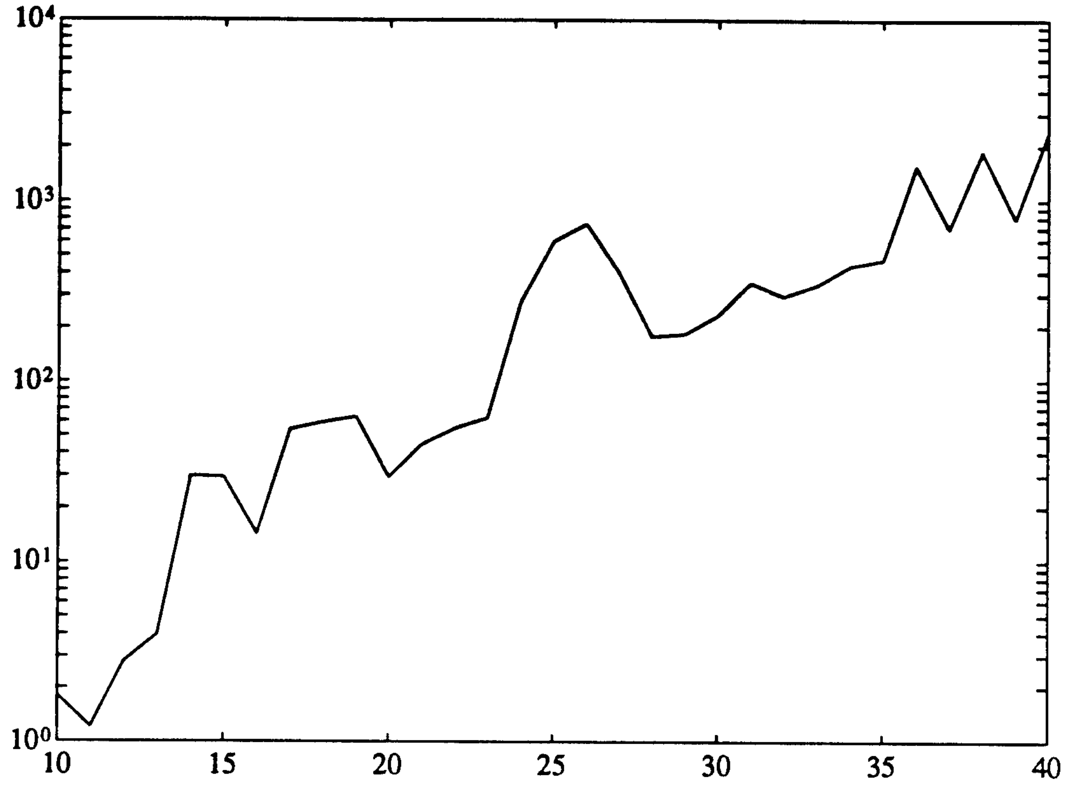

Figure 3. Solution time in seconds versus b_1 for CONVEX

As in section 2, run times can be reduced substantially by calculating solutions that are only within d of optimality, where $d > 0$. The effect is particularly dramatic on large problems. For example, setting $d = .01$ and increasing b_1 as before results in essentially a zero run time for $b_1 \geq 20$, even though the run time for $b_1 = 19$ is about 20 seconds. There are two explanations of this welcome but odd behavior:

- CONVEX obtains its feasible value by rounding, and big numbers are easier to round than small ones.
- A fixed d is a larger percentage of the optimized objective function as b_1 increases.

In any case, the effect on sufficiently large problems is that a single call to CONVEX obtains a feasible solution within d of the bound, thus terminating the procedure.

In conclusion, the best B+B procedure for NLP2 appears to be based on the CONVEX relaxation, rather than a hybrid. Problems having on the order of 20 units are solvable in a few seconds, or considerably larger problems if the $W(i, j)$ are small or if only near optimal solutions are needed.

4. Summary

Efficient Branch and Bound methods require bounds that are both easily evaluated and sharp. In practice there is often a tradeoff between speed of evaluation and sharpness. The moving target problem considered in section 2 is a good example of this; the ERGO bound is fastest, while FABC is sharpest. The best bound to employ turns out to be strongly dependent on problem parameters, particularly the overlook probability. Robustness can be achieved by employing a hybrid bound where a sharp bound is attempted only after the failure of a fast one, the author's favorite hybrid being the employment of FABC after the failure of MEAN. A hybrid procedure is generally slightly slower than a pure procedure based on one of its own components, but the gain in robustness may be worth the sacrifice.

The problem considered in section 3 is simpler in that the fast LINEAR bound is (almost) uniformly dominated by the sharp CONVEX bound. There appears to be no good argument for hybrid bounds in this problem, at least not for the class of parametric variations considered.

B+B procedures do not prevent an exponential growth of solution time as problem size increases, so progress in expanding the size of search problems that can be optimally solved will be slow. A compensating feature is that B+B procedures can be easily adapted to finding solutions that are almost optimal in a well defined sense. For both problems considered, requiring the solution to be only within d of optimality, where $d > 0$, permits much easier fathoming and results in strongly decreased run times.

REFERENCES

[1] Benkoski, S.J., Monticino, M.G., and Weisinger, J.R., "A Survey of the Search Theory Literature," *Naval Research Logistics*, **38**, 469–494 (1991).

[2] Brown, S.S., "Optimal Search for a Moving Target in Discrete Time and Space," *Operations Research*, **28**, 1275–1289 (1980).

[3] Eagle, J.N., and Yee, J.R., "An Optimal Branch-and-Bound Procedure for the Constrained Path, Moving Target Search Problem," *Operations Research*, **38**, 110–114 (1990).

[4] Feller, W., *An Introduction to Probability Theory and Its Applications*, Volume 1, Edition 2, Wiley, New York, p. 373 (1957).

[5] Martins, G., "A New Branch-and-Bound Procedure for Computing Optimal Search Paths," Naval Postgraduate School Master's Thesis (1993).

[6] Stewart, T.J., "Experience With a Branch-and-Bound Method for Constrained Searcher Motion," *Search Theory and Applications* (eds. Haley and Stone), Plenum, New York, (1980).

[7] Stewart, T.J., "Search for a Moving Target When Search Motion is Restricted," *Computations and Operation Research*, **6**, 129–140 (1979).

[8] Washburn, A.R., "An Upper Bound Useful in Optimizing Search for a Moving Target," *Operations Research*, **29**, 1227–1230 (1981).

[9] Washburn, A.R., "Finite Method for a Nonlinear Allocation Problem," to appear in *Journal of Optimization Theory and Applications*, **85**, #3, (1995).

[10] Washburn, A.R., "Search for a Moving Target: The FAB Algorithm," *Operations Research*, **31**, 739–751 (1983).

APPENDIX A: Extension of a Result on Bounding

Consider the minimization of $q(\Psi) \equiv \exp(-Z(X, \Psi))$, where $Z(X, \Psi)$ is given by (2) and $\Psi(x, t), x \in C, 1 \leq t \leq T$ is a feasible search plan. If Ψ and Ψ' are two different search plans, then Washburn [8] shows that $q(\Psi')$ cannot be smaller than $q(\Psi)$ by an amount that exceeds

$$\Delta(\Psi', \Psi) = \sum_{t=1}^{T} \sum_{x \in C} D(\Psi, x, t)[\Psi'(x,t) - \Psi(x,t)], \tag{A1}$$

where $D(\Psi, x, t) = W(x, t)\ P(\Psi, x, t)\ \exp(-W(x, t)\Psi(x, t))\ Q(\Psi, x, t)$, with $P(\Psi, x, t)$ and $Q(\Psi, x, t)$ being as defined in (8) and (9). For every search plan Ψ, there is therefore a global lower bound

$$q(\Psi) - \max_{\Psi'} \Delta(\Psi', \Psi) \tag{A2}$$

on the nondetection probability.

Washburn [8] is concerned with the DOE relaxation of NLP1, but in fact (A1) can be the basis of efficient bounds for NLP1 itself. Consider the case where the search path $\sigma_0, \ldots, \sigma_\tau$ is shared by Ψ and Ψ', with the continuation $\sigma'_{\tau+1}, \ldots, \sigma'_T$ possibly differing from $\sigma_{\tau+1}, \ldots, \sigma_T$. Then

$$\Delta(\Psi', \Psi) = \sum_{t=\tau+1}^{T} \left[D(\Psi, \sigma'_t, t) - D(\Psi, \sigma_t, t) \right].$$

Let $G(x, t) \equiv D(\Psi, x, t) - D(\Psi, \sigma_t, t)$. Then selection of Ψ' to maximize $\Delta(\Psi', \Psi)$ is equivalent to solving a longest route problem starting from (σ_τ, τ), with the reward for visiting (x, t) being $G(x, t)$.

A somewhat sharper bound is sometimes possible. The development above relies on the observation that

$$q(\Psi) - q(\Psi') = E\left(e^{-Z(x,\Psi)} - e^{-Z(x,\Psi')}\right) = E\left(e^{-Z(x,\Psi)}\left[1 - e^{-[Z(x,\Psi') - Z(x,\Psi)]}\right]\right). \tag{A3}$$

Let $Y \equiv Z(x, \Psi') - Z(x, \Psi)$. Then $1 - e^{-Y} \leq fY$, where f is the factor introduced in describing the Linear relaxation in section 2.2. It follows from (A2) that

$$q(\Psi) - q(\Psi') \leq E\left(fYe^{-Z(x,\Psi)}\right) \tag{A4}$$

with the right hand side of (A4) being $f\Delta(\Psi,\Psi')$ as given by (A1). Therefore $D(\Psi, x, t)$ and $G(x, t)$ can also be multiplied by f. This sharpened bound is the one used as the "FAB bound" reported earlier.

It should be noted that (A2) is valid for any search plan that follows $\sigma_0, \ldots, \sigma_\tau$ up to time τ. Now, constraint (3) in NLP1 can clearly be relaxed by replacing = with $\leq$ without effect on the optimized objective function, $\exp(-Z(x, \Psi))$ being a decreasing function of $\Psi(x, t)$ for all $x \in C$, $1 \leq t \leq T$. Therefore the NULL continuation of setting $\Psi(x, t) = 0$ for $x \in C$ and $t > \tau$ is permissible in (A2). In that case $D(\Psi, x, t) = W(x, t)\ P(\Psi, x, t)$, and (A2) is equivalent to the right hand side of (14). In other words, the MEAN bound obtained via (14) is the same as the bound obtained by the policy of no further search in (A2). There may be search policies that, from a standpoint of producing good bounds, are superior to either the FAB or the NULL continuation. However, no means of easily generating such continuations is currently known.

APPENDIX B: An Ergodic Bound

Assume that the target's motion is ergodic with Markov transition matrix $p = (p_{ij})$; $i,j \in C$. Then there is a unique probability distribution π that solves the vector equation $\pi = \pi p$. π is the "stationary distribution" of p. Furthermore, $(q_{ij}) = (\pi(j)p_{ji}/\pi(i))$ is also the transition matrix of a Markov chain — the time reversed chain [4]. Let $u_t(j) = P(X_t = j)$ and let $r_t(j) = u_t(j)/\pi_j$, $j \in C$ and $1 \le t \le T$. Also, let $r(t) = \max_{j \in C} r_t(j)$. Then the extent to which $r(t) \ge 1$ measures the deviation of the distribution $(u_t(j); j \in C)$ from the equilibrium distribution π. Now, since $u_{t+1}(i) = \sum_{j \in C} P_{ji} u_t(j)$, it follows that

$$r(t+1) = \max_i u_{t+1}(i)/\pi(i) = \max_i \sum_{j \in C} P_{ji} u_t(j)/\pi(i) \tag{B1}$$

But $P_{ji}/\pi(i)$ is the same as $q_{ij}/\pi(j)$, and $u_t(j)/\pi(j) \le r(t)$, so

$$r(t+1) \le \max_i \sum_{j \in C} q_{ij} r(t) = r(t). \tag{B2}$$

Since $\sum_{j \in C} q_{ij} = 1$ for all i, every transition brings an ergodic chain closer to equilibrium.

The reward in equation (16) is just

$$R(\sigma_t, t) = fW(\sigma_t, t)\text{Prob}(X_t = \sigma_t \ \underline{\text{and}}\ E_\tau), \tag{B3}$$

where E_τ is the event that searches up to and including time τ all fail to detect the target. Let $u_t(\sigma_t) \equiv \text{Prob}(X_t = \sigma_t \ \underline{\text{and}}\ E_\tau)$, $\tau < t \le T$, and let $u_t = (u_t(j))$; $j \in C$ be the row vector that includes the probability of all possible states. u_t is defective (sums to $\text{Prob}(E_\tau)$, rather than 1), but nonetheless $u_{t+1} = u_t p$ for $\tau < t \le T$. Therefore, if $r = \max_{j \in C} u_{\tau+1}(j)/\pi(j)$, it must be true that $u_t(j) \le r\pi(j)$ for all j and $\tau < t \le T$. Consequently

$$\sum_{t=\tau+1}^{T} R(\sigma_t, t) \le fr \sum_{t=\tau+1}^{T} W(\sigma_t, t)\pi(\sigma_t). \tag{B4}$$

Finally let $A(\tau, \sigma_\tau)$ be the maximized value of the sum on the right hand side of (B4). Finding $A(t, x)$ for $1 \le t \le T$ and $x \in C$ is a longest path problem, but it need only be solved once. Then $frA(\tau, \sigma_\tau)$ is an upper bound on the right hand side of (16).

In operation $u_{\tau+1}(j)$ is just $P(\Psi, j, \tau + 1)$, the same quantity manipulated in the FAB algorithm, so calculating r simply involves a maximization of ratios. In the simplest case $\pi(j) = 1/N$ for $j \in C$ and $W(x, t) = W$ for $j \in C$ and $t > \tau$. In that case the right hand side of (B4) is $fW \max_{j \in C} P(\Psi, j, \tau + 1)$, regardless of the continuation σ.

INITIAL DISTRIBUTION LIST

1.	Research Office (Code 08) Naval Postgraduate School Monterey, CA 93943-5000	1
2.	Dudley Knox Library (Code 52) Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
4.	Prof. Alan R. Washburn (Code OR/Ws) Naval Postgraduate School Monterey, CA 93943-5000	5
5.	Prof. Robert Dell (Code OR/De) Naval Postgraduate School Monterey, CA 93943-5000	1
6.	Prof. James N. Eagle (Code OR/Er) Naval Postgraduate School Monterey, CA 93943-5000	1
7.	Editorial Assistant (Code OR/Bi) Naval Postgraduate School Monterey, CA 93943-5000	1
8.	The Johns Hopkins University Applied Physics Laboratory Attn: L. F. Rogers (7-338) Johns Hopkins Road Laurel, MD 20723-6099	1
9.	EPL Analysis, Inc. Attn: E. P. Loane 2919-A Olney-Sandy Spring Road Olney, MD 20832	1
10.	David Lovelock Math Department University of Tucson Tucson, AZ 85721	1
11.	METRON, Inc. 11911 Freedom Dr., Suite 800 Reston, VA 22090	1

12. Koji Iida 1
National Defense Academy
Hashirimizu 1-10-20
Yokosuka 239 JAPAN

13. Daniel H. Wagner, Ass. 4
Station Square Two
Paoli, PA 19301

14. LCDR Gustavo H. A. Martins 1
Attn: Brazilian Naval Commission
4706 Wisconsin Ave, NW
Washington, D.C. 20016

15. LCDR Almir Garnier Santos.. 1
Attn: Brazilian Naval Commission
4706 Wisconsin Ave, NW
Washington, D.C. 20016