Proceedings of the 1998 Winter Simulation Conference
D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan, eds.

REACTIVE TABU SEARCH IN UNMANNED AERIAL RECONNAISSANCE SIMULATIONS

Joel L. Ryan, T. Glenn Bailey, James T. Moore

Department of Operational Sciences
Air Force Institute of Technology
2950 P Street, Bldg 640 (AFIT/ENS)
Wright-Patterson AFB, Ohio 45433-7765, U.S.A.

William B. Carlton

Department of Systems Engineering
Mahan Hall, Bldg 752
United States Military Academy
West Point, New York 10996-1775, U.S.A.

ABSTRACT

We apply a Reactive Tabu Search (RTS) heuristic within a discrete-event simulation to solve routing problems for Unmanned Aerial Vehicles (UAVs). Our formulation represents this problem as a multiple Traveling Salesman Problem with time windows (mTSPTW), with the objective of maximizing expected target coverage. Incorporating weather and probability of UAV survival at each target as random inputs, the RTS heuristic in the simulation searches for the best solution in each realization of the problem scenario in order to identify those routes that are robust to variations in weather, threat, or target service times. We present an object-oriented implementation of this approach using CACI's simulation language MODSIM.

1 INTRODUCTION

We present a continuation of the research begun by Carlton (1995) into the effectiveness of Reactive Tabu Search (RTS) on the multiple traveling salesman problem with time window constraints (mTSPTW), and on how this approach can be used to model unmanned aerial vehicle (UAV) applications (Sisson 1997). UAV problems differ from those traditionally found in the General Vehicle Routing Problem (GVRP) literature because they include unique stochastic inputs, such as random winds and service times. The advantages of object-oriented simulation are added to these earlier works to provide a mechanism for extensive exploration of problems within the GVRP family. This paper demonstrates an application that identifies UAV routes that are persistent throughout a simulation's state space.

We begin by noting that *tabu search* (TS) (Glover 1990, Glover and Laguna 1997) is a heuristic for providing excellent solutions to hard combinatorial problems by moving from one solution to another in a way that avoids becoming trapped in local optimal solutions. (TS records and returns the best solutions discovered during the search, and often these solutions are optimal. It is important to note, however, that TS does not guarantee finding an optimal solution, nor will it recognize an optimal solution if it encounters one.) Through the use of flexible memory, mechanisms for either constraining or relaxing the criteria used in the search process, and intensification and diversification, TS represents a logical application of adaptive, memory-based search strategies. The literature identifies TS – and a variant called *reactive tabu search* (RTS) – as powerful heuristics for the GVRP (see Laporte 1992, Battiti and Tecchiolli 1994, and Battiti 1996). Since this paper focuses on TS applications in a simulation context, we will assume a working knowledge of TS/RTS procedures; otherwise, we refer the interested reader to the above references.

2 UAV PROBLEM FORMULATION

We begin our formulation of the UAV problem (UAVP) using a mTSPTW baseline. Following Carlton (1995), our RTS seeks "near optimal" solutions to a mTSPTW with *nc* customers, indexed by *i* or *j*, each requiring a service time s_i. (In the context of the UAVP, the terms "target" or "target node" represent a "customer".) The starting depot is designated 0; the terminal depot by *nc*. Given *nv* vehicles, if no feasible solutions are found after a reasonable search we increase *nv* and restart the search. The time window for each customer *i*'s pick up is (e_i, l_i), where e_i is the earliest possible arrival time and l_i is the latest. The early arrival time is treated as a "soft" constraint; i.e., vehicles arriving before e_i may wait until e_i is reached. W_i is the wait time at customer *i*. The parameter $t_{i,j}$ is the travel time from customer *i* to customer *j*. The binary decision variable $X^v_{i,j}$ equals 1 if vehicle *v* travels on the arc between customers *i* and *j*; otherwise it is 0. Tour schedule variables A_i and T_i indicate the time a vehicle arrives at customer *i* and the time service starts at customer *i*, respectively. The time windows, times between nodes, and service times are constrained to be integer for computational efficiency. Formally, we express the mTSPTW as

$$\text{MIN } Z_t = \sum_{i=1}^{nc} \sum_{j=1}^{nc} \sum_{v=1}^{nv} X^v{}_{i,j} \cdot t_{i,j}$$

Subject To:

$$\sum_{i=0}^{nc} \sum_{v=1}^{nv} X^v{}_{i,j} = 1 \quad \forall\ j = 1..nc$$

{One vehicle enters per customer}

$$\sum_{j=1}^{nc} \sum_{v=1}^{nv} X^v{}_{i,j} = 1 \quad \forall\ i = 0..nc$$

{One vehicle leaving per customer}

$$X^v{}_{i,j} = 1 \Rightarrow T_i + s_i + t_{i,j} + W_j = T_j$$

{Time precedence}

$$e_i \le T_i \le l_i \quad \forall\ i = 1..nc$$

{Time windows}

$$W_i = T_i - A_i \quad \forall\ i = 1..nc$$

{Waiting times}

{Subtour breaking constraints are not shown, but are included in the model.}

$$\sum_{i=0}^{nc} X^v{}_{i,i} = 0 \quad \forall\ v = 1..nv$$

{Vehicle must serve different adjacent nodes}

$$\sum_{i=0}^{nc} X^v{}_{i,j} = \sum_{\substack{k=0 \\ k \ne i}}^{nc} X^v{}_{j,k} \quad \forall\ j = 1..nc,\ \forall\ v = 1..nv$$

{Same vehicle entering a node must exit}
{Routes cannot terminate at a customer}

The UAV problem modifies this formulation by including vehicle-related route length constraints and changing the objective function. Given T^v as the maximum time a vehicle can be used, route length constraints are defined as

$$\sum_{i=0}^{nc} \sum_{j=1}^{nc} X^v{}_{i,j} \cdot s_j + \sum_{i=0}^{nc} \sum_{j=1}^{nc} X^v{}_{i,j} \cdot W_j + \sum_{i=0}^{nc} \sum_{j=0}^{nc} X^v{}_{i,j} \cdot t_{i,j} \le T^v \quad \forall\ v = 1..nv \tag{1}$$

The objective function is replaced by an expected coverage function. Formally, coverage is defined as the number of targets that will be visited; therefore, the expected coverage of any single target equals the probability of surviving that target. Notationally, for target node n_i^v (the i^{th} target node visited in the route of vehicle *v*) the expected coverage is given by

$$\prod_{i=a^v}^{n_i^v} Ps(i)$$

where a^v is the starting node of vehicle *v*'s tour, and *Ps(i)* is the probability of survival at target node *i*. For instance, assuming a UAV travels from target 1 to 2 to 3, and *Ps*(1) = 0.9, *Ps*(2) = 0.8, and *Ps*(3) = 0.7, target 1's coverage is 0.9, target 2's is 0.9*0.8=0.72, and target 3's is 0.90*0.80*0.70=0.50.

The expected number of nodes covered along the route of vehicle *v* is given by the sum of the individual node coverages; i.e.,

$$\sum_{n_i^v = a^v}^{b^v} \prod_{i=a^v}^{n_i^v} Ps(i) \tag{2}$$

where b^v is the ending node of vehicle v's tour and $a^v \le n_i^v \le b^v$. Thus, for the three node example above, the expected number of nodes covered is 0.90 + 0.72 + 0.50 = 2.12. The inclusion of (1) to the constraints of the mTSPTW formulation, along with the substitution of its objective function with the objective of maximizing (2), defines the UAVP.

3 IMPLEMENTATION

Object-oriented programming languages facilitate the inheritance and reuse of existing object definitions and methods (Kassou and Pecuchet 1994). Our paper makes full use of this approach by using CACI's object-oriented language MODSIM (CACI 1997). In MODSIM, an object contains its own fields and routines (methods). While the contents of an object's fields can only be modified by its own methods, it can share those values with any other part of the program. Through inheritance, new object types arise from existing types by inheriting the fields and objects of the existing type. New objects can then redefine (or override) the inherited methods to behave differently, as well as add original fields and methods.

This code encapsulation is useful in solving the GVRP in that it allows different objective functions to be efficiently introduced to a RTS solver. Such inheritance and reuse advantages motivate our creation of an RTS *object* for the UAVP by translating Carlton's (1995) C language code into a set of MODSIM libraries and objects. These objects provide a "core" solver for the mTSP and mTSPTW instances of the GVRP family, and with very minor adjustments can solve UAVP problems as well.

Table 1 depicts the MODSIM structure of libraries and objects designed to solve mTSPTW problems. The pseudocode corresponds to the OBJECT, METHOD, and PROCEDURE columns in a hierarchical fashion similar to a path name. The heading ("main") indicates the implementation code can be found in the main module. In all cases, one follows the path to find the physical location of the code in the right-most nonblank space. If the code is not in the main module, the library listed refers to the library in which the right-most nonblank identifier lies. Dark gray spaces indicate that depth in the hierarchy is unneeded to specify the location.

The libraries provide a general framework for categorizing code into areas of similarity. Here, "tabuMod" contains code for use in GVRP-related tabu heuristics. The modules of "tsptwMod" contain code tailored for the mTSPTW and UAV problems, and "hashMod" holds the code for the creation and use of the hashing structure. As noted by Carlton (1995), many different objective functions can be used for GVRP problems, so "bestSolnMod" separates the code determining the best solution visited. Finally, we verified our RTS object by stepping line-by-line through the translated code with a 4-city TSP problem to ensure accuracy, then compared the heuristic's results to a problem from Reinelts' TSPLIB (1991).

Using the "portable" quality of our UAV object, we embed it in a Monte-Carlo simulation that seeks to model the inherent variability of the operational environment's parameters. We accomplish this by creating a scenario at each replication of the simulation that has a unique realization of wind magnitude and direction, target survival rates, and service times. Then, beginning from an arbitrary solution, the RTS object finds the best routing solution for that particular scenario. When the RTS for the current replication's scenario ends (i.e., the specified number of tabu search iterations have been accomplished), a new set of realizations of the random variables are generated for the scenario used in the simulation's next replication. The RTS then begins from the best solution of the just-completed scenario; in this manner the previous routing solution serves as a naïve forecast of the next one. When the search completes the last scenario, the frequencies of routes used in the feasible solutions are summed in a route frequency matrix. The feasible solution whose routes are most persistent (i.e., those whose sum of route frequencies is the greatest) is termed the "Robust Tour."

We demonstrate this technique using Sisson's (1997) notional Nari dataset (Table 2). The coordinates are stated in miles from a fixed point. In this case the Nari dataset is essentially a TSP with a route length constraint of 24 hours. The service times are stochastic and a range of service times is possible at each target. The minimum service time is chosen unless a uniform random draw between 0 and 1 results in a value less than a predetermined probability (P_l) that the UAV may need to loiter at the target. If the first draw determines that an extended service time is required, a second uniform random draw between the minimum and maximum service times determines the amount of time the UAV will loiter over the target.

We conducted a 21-day simulation of the Nari scenario, where the winds vary between 205 and 245 degrees in orientation at a magnitude ranging between 0 to 20 knots. The mean expected probabilities of survival (P_s) for the target nodes are set to either 0.8 or 0.9. Within the simulation, each target node is given a 0.5 probability (P_l) of its service (i.e., loiter) time increasing above its minimum level, and eleven vehicles are made available for use. In each replication, the RTS is limited to 500 iterations.

Figure 1 graphically depicts the tours chosen for each day of the 21-day simulation (the depot lies above the first tick mark on the horizontal axis, where we see the eleven vehicle tours converge). Although the shapes of the tours do not readily yield to a visual examination, based on a frequency analysis Day 16 contains those tours that appeared most often.

4 CONCLUSIONS

This paper extends reactive tabu search to unmanned aerial vehicle routing through discrete-event simulations that incorporate the stochastic nature of real-world UAV scenarios. The use of RTS objects that use proven heuristic methods within a simulation provides good solutions to individual realizations of the target environment, which in turn can form the basis for identifying routing assignments that are robust to variations in wind, loiter times, and probability of survival. Additionally, the work described here can be applied towards evaluating the military worth of new and innovative concepts that attempt to improve UAV mission performance.

Table 1: Main Module Diagram, mTSPTW.

	Mtsptw (ModSim main module)			
mTSPTW Reactive Tabu Search Pseudocode	**SOURCE**	**OBJECT**	**METHOD**	**PROCEDURE**
0. Intialize: Structures, vectors, parameters....................................	(main)			
1. Input problem instance:..	tsptwMod	timeMatrix	readCarlton	
a. Number of iterations = niters..	(main)			
b. Compute time/distance matrix..	tsptwMod	timeMatrix	timeMatrix	
2. Select the starting tour..	tsptwMod	startTour	startTour	
a. Compute initial schedule..	tabuMod	"	"	tourSched
b. Compute initial tour penalties..	tabuMod	"	startPenBest	compPens
c. Given penalties, compute initial tour cost....................................	tabuMod	"	"	tsptwPen
d. Compute the initial hashing values: f(T) and thv(T).....................	tabuMod	"	"	tourHVwz
e. Save as initial best solution..	bestSolnMod	"	"	twBestTT
3. While (k <= niters)..	tsptwMod	reacTabuObj	search	
a. Look for the incumbent tour in the hashing structure..................	hashMod	"	"	lookfor
1) If found, update the iteration when found, and increase the tabu length, if applicable..	tabuMod	"	"	cycle
2) If not found, add to the hashing structure, and decrease the tabu length, if applicable..	tabuMod	"	"	nocycle
b. Perform "later" insertions: I(i,d) for i = 1 to n-1, d >= 1.................	tabuMod	reacTabuObj	search	SwapNode
1) Calculate the penalties associated with an insertion..............	tabuMod	"	"	compPens
2) Calculate the value of making this insertion......................	tabuMod	"	"	moveValTT
c. Evaluate all "earlier" insertions: I(i,d) for i = 3 to n, d <= -2..........	"	"	"	"
d. Move to the non-tabu neighbor according to an appropriate decision criteria. If all tours are tabu, move to the neighbor with the smallest move value, and reduce the tabu length.........	tabuMod	reacTabuObj	search	insert
e. Update the search parameters:				
1) Incumbent tour schedule..	tabuMod	"	"	tourSched
2) Incumbent tour hashing value..	tabuMod	"	"	tourHVwz
3) Retain the best feasible solution found and the tour with the smallest tour cost regardless of feasiblity..........................	bestSolnMod	"	"	twBestTT
f. Increase iteration count: k = k + 1..	tsptwMod	reacTabuObj	search	
4. Output results..	tabuMod			twLoadToFile

Directions: To find where a portion of the pseudocode is executed, one can read the OBJECT, METHOD, and PROCEDURE columns like a hierarchial path name. The heading "(main)" indicates the implementation code can be found in the main module. Dark gray spaces indicate that space's depth in the hiearchy is unneeded to specify the location and " indicates the reference is identical to the last entry above it.

Table 2: Nari Dataset.

	Coordinates (in miles) X	Y	Early Arrival	Late Arrival	Service Time Ranges (in hours)		Probability of Survival
0*	100.286	64.286	0	24	0	0	1
1	7.714	381.429	0	24	1	5	0.9
2	55.714	6	0	24	1	5	0.8
3	81.429	351.429	0	24	1	5	0.9
4	58.286	342.857	0	24	1	5	0.8
5	65.143	325.714	0	24	1	5	0.9
6	34.286	327.429	0	24	1	5	0.8
7	70.286	296.571	0	24	1	5	0.9
8	27.429	291.429	0	24	1	5	0.8
9	93.429	297.429	0	24	1	5	0.9
10	48	280.286	0	24	1	5	0.8
11	76.286	269.143	0	24	1	5	0.9
12	120	274.286	0	24	1	5	0.8
13	160.286	291.429	0	24	1	5	0.9
14	100.286	251.143	0	24	1	5	0.8
15	114	216	0	24	1	5	0.9
16	205.714	234	0	24	1	5	0.8
17	104.571	219.429	0	24	1	5	0.9
18	144	220.286	0	24	1	5	0.8
19	126.857	203.143	0	24	1	5	0.9
20	231.429	217.714	0	24	1	5	0.8
21	292.286	191.143	0	24	1	5	0.9
22	181.714	145.714	0	24	1	5	0.8
23	200.571	140.571	0	24	1	5	0.9
24	291.429	137.143	0	24	1	5	0.8
25	214.286	121.714	0	24	1	5	0.9
26	248.571	92.571	0	24	1	5	0.8
27	274.286	82.286	0	24	1	5	0.9
28	291.429	78.857	0	24	1	5	0.8
29	332.571	82.286	0	24	1	5	0.9
30	349.714	80.571	0	24	1	5	0.8
31	377.143	84	0	24	1	5	0.9
32	375.429	99.429	0	24	1	5	0.8
33	385.714	111.429	0	24	1	5	0.9
34	402.857	115.714	0	24	1	5	0.8
35	404.571	106.286	0	24	1	5	0.9
36	396	94.286	0	24	1	5	0.8
37	432	92.571	0	24	1	5	0.9
38	437.143	70.286	0	24	1	5	0.8
39	447.429	43.714	0	24	1	5	0.9
40	472.286	33.429	0	24	1	5	0.8

* Denotes the depot.

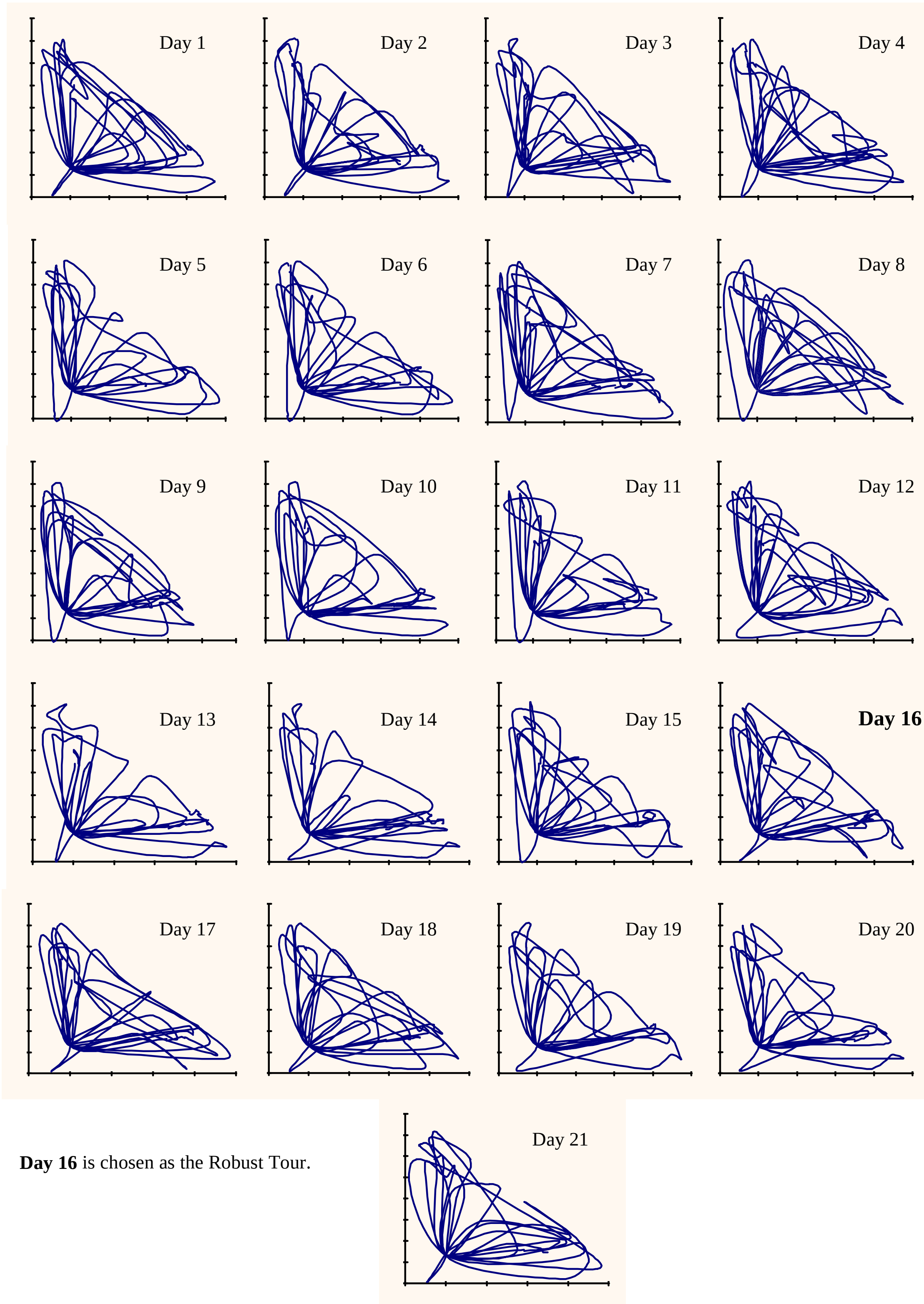

Figure 1: Tours Chosen for Nari Scenarios**.**

Our research also contributes to the practical application of RTS with the creation of the MODSIM libraries. Using these libraries, future code can be quickly tailored to specific members of the GVRP family. Even if the programmer is not using MODSIM, the libraries provide a straightforward translation given the “strongly typed” nature of MODSIM and the strict adherence to code encapsulation they embody. Their use can reduce the up-front coding time so often required in GVRP research.

REFERENCES

Battiti, R. 1996. Reactive Search: Toward Self-Tuning Heuristics. *Modern Heuristic Search Methods.* Ed. V.J. Rayward-Smith, I.H. Osman, C.R. Reeves and G.D. Smith, New York: John Wiley and Sons, Inc.

Battiti, R. and G. Tecchiolli. 1994. The Reactive Tabu Search. *ORSA Journal on Computing,* 6 (2), 126-140.

CACI Products Company. 1997. La Jolla, California.

Carlton, W. B. 1995. A Tabu Search Approach to the General Vehicle Routing Problem. Ph.D. dissertation, Department of Mechanical Engineering, University of Texas, Austin, Texas.

Glover, F. 1990. Tabu Search: A Tutorial. *Interfaces, 20*: 74-94.

Glover, F. and M. Laguna. 1997. *Tabu Search.* Boston: Kluwer Academic Publishers.

Kassou, I. and J. Pecuchet. 1994. Use of simulation within a general framework of optimization in job-shop scheduling. In *Proceedings of CISS - First Joint Conference of International Simulation Societies.* SCS: San Diego, California.

Laporte, G. 1992. The Vehicle Routing Problem: An overview of exact and approximate algorithms. *European Journal of Operational Research, 59*: 345-358.

Reinelt, G. 1991. TSPLIB - A TSP Library. *ORSA Journal on Computing. 3*: 376-384.

Sisson, M. R. 1997. Applying Tabu Heuristic to Wind Influenced, Minimum Risk and Maximum Expected Coverage Routes. Master’s Thesis, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio.

AUTHOR BIOGRAPHIES

JOEL L. RYAN, Captain, USAF, received his B.S. from the U.S. Air Force Academy in 1993 and his M.S. in operations research from the Air Force Institute of Technology in 1998. He is currently assigned to the Studies and Analyses squadron at HQ, Air Combat Command, Langley AFB, VA.

T. GLENN BAILEY, Lieutenant Colonel, USAF, is an assistant professor in the Department of Operational Sciences at the Air Force Institute of Technology. A senior pilot and 1978 graduate of the U.S. Air Force Academy, Lt Col Bailey received his M.S. in operations research from AFIT in 1988, and his Ph.D. from the University of Texas at Austin in 1995. He is a member of InfORMS, MORS, and SIAM.

JAMES T. MOORE is an Associate Professor in the Department of Operational Sciences at the Air Force Institute of Technology. A 1974 graduate of the University of Colorado, Dr. Moore received a M.S. in operations research from AFIT in 1981 and his Ph.D. from the University of Texas at Austin in 1988. His research interests include optimization, scheduling, and heuristics.

WILLIAM B. CARLTON, Lieutenant Colonel, USA, is the Director of the Operations Research center at the United States Military Academy. A 1976 graduate of Presbyterian College, LTC Carlton received a M.S. in operations research from Georgia Tech, and his Ph.D. from the University of Texas at Austin. He is a registered professional engineer in the Commonwealth of Virginia, and a member of the MORS and the Army Operations Research Society (AORS).