

AFRL-IF-RS-TR-2002-178
Final Technical Report
August 2002



AGILE CONTROL OF MILITARY OPERATIONS JOINT FORCE AIR COMPONENT COMMANDER (JFACC)

ALPHATECH, Incorporated

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. J107

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

**AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK**

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

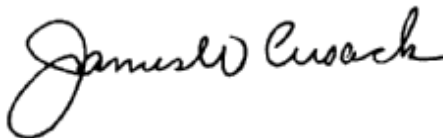
AFRL-IF-RS-TR-2002-178 has been reviewed and is approved for publication

APPROVED:

A handwritten signature in black ink, appearing to read 'Carl A. DeFranco', with a stylized, cursive script.

CARL A. DEFRANCO
Project Engineer

FOR THE DIRECTOR:

A handwritten signature in black ink, appearing to read 'James W. Cusack', with a stylized, cursive script.

JAMES W. CUSACK, Chief
Information Systems Division
Information Directorate

REPORT DOCUMENTATION PAGE			<i>Form Approved</i> OMB No. 074-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE AUGUST 2002	3. REPORT TYPE AND DATES COVERED Final Aug 99 – Oct 01	
4. TITLE AND SUBTITLE AGILE CONTROL OF MILITARY OPERATIONS JOINT FORCE AIR COMPONENT COMMANDER (JFACC)			5. FUNDING NUMBERS C - F30602-99-C-0203 PE - 63760E PR - J107 TA - 00 WU - 01	
6. AUTHOR(S) D. A. Logan, J. M. Wohletz, D. A. Castañon, M. L. Burry, and B. Bank				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) ALPHATECH, Incorporated 50 Mall Road Burlington Massachusetts 01803			8. PERFORMING ORGANIZATION REPORT NUMBER TR-1048	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Advanced Research Projects Agency AFRL/IFSA 3701 North Fairfax Drive 26 Electronic Parkway Arlington Virginia 22203-1714 Rome New York 13441-4514			10. SPONSORING / MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2002-178	
11. SUPPLEMENTARY NOTES AFRL Project Engineer: Carl A. DeFranco/IFSA/(315) 330-3096/ Carl.DeFranco@rl.af.mil				
12a. DISTRIBUTION / AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.				12b. DISTRIBUTION CODE
13. ABSTRACT (Maximum 200 Words) This research focused on the problem of providing military commanders with real-time, optimal control of military air-to-ground operations for a 24-hour segment of a Joint Air Operations (JAO) campaign. In particular, we focused on developing control algorithms that anticipate possible air-to-ground mission modifications due to uncertain future events, thereby generating missions that can be readily adapted in the presence of contingencies. The primary benefit of this technology is agile and stable control of distributed and dynamic military operations conducted in inherently uncertain, hostile, and rapidly changing environments. The control methodology developed combines Approximate Dynamic Programming (ADP) and statistical hybrid state modeling techniques. Accordingly, a novel hybrid, multi-rate control architecture that tailors the control strategy for different battlespace situations was developed. For this JAO problem, a key concern was the scalability of the control methodology to larger scenarios. As a result, we investigated a broad spectrum of ADP control strategies. The solution quality and computation performance of these algorithms was tested and verified in a JAO simulator. It was shown through experimentation that the ADP strategies were able to produce operationally consistent control strategies that anticipated likely contingencies and positioned assets for opportunities of recourse in near real-time.				
14. SUBJECT TERMS Joint Air Operations, Approximate Dynamic Programming, Control, Battlespace, Mutli-Rate Control, Statistical Hybrid, Agile Control			15. NUMBER OF PAGES 113	
			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	

TABLE OF CONTENTS

<i>List of Figures</i>	<i>iii</i>
<i>1 Executive Summary</i>	<i>1</i>
<i>2 Introduction</i>	<i>3</i>
2.1 Background	3
2.2 Problem Description.....	5
2.3 Theoretical Technique.....	5
2.4 Value to Military	6
2.5 Report Layout.....	7
<i>3 JAO Plant Dyanmics</i>	<i>8</i>
3.1 JOA Air-to-GrouND Problem Description	8
3.2 Hybrid Dynamical Model Formulation	11
3.2.1 Battlespace Objects.....	11
3.2.2 Battlespace Dynamics.....	16
3.3 Hybrid Dynamical Model Implementation	24
3.3.1 Discrete Event Simulation	26
3.3.2 Plant/Controller Interface.....	26
3.3.3 Distributed Processing	27
<i>4 JAO Controller Formulation</i>	<i>29</i>
4.1 Control Framework	29
4.2 Proof of Concept Experiments and Research Framework	33
4.2.1 Proof of Concept Experimental Results.....	33
4.2.2 Control Research Framework	36
4.3 Combinatorial Optimization Algorithms.....	39
4.3.1 Maximum Marginal Return	41
4.3.2 Combinatorial Rollout	44
4.3.3 Surrogate Method.....	45
4.4 Design Model Approach	47
4.5 Combinatorial Optimization Algorithms Implemented.....	49
4.5.1 Retasker using Combinatorial Rollout.....	49
4.5.2 Aborter using Combinatorial Rollout	50
4.5.3 Target Tasking using Maximum Marginal Return	51
4.5.4 AOR Tasking using Maximum Marginal Return	53
4.5.5 Target Tasking using Surrogate Method.....	53
4.5.6 AOR Tasking using Surrogate Method.....	54
4.5.7 Target Tasking using Combinatorial Rollout	54
4.6 Design Models Implemented.....	55
4.6.1 1-Stage/1-Wave Model	55
4.6.2 2-Stage/1-Wave Model with Retasking.....	56
4.6.3 2-Stage/2-Wave Model	58
4.6.4 4-Stage/2-Wave Model with Retasking.....	62
4.6.5 2-Stage/1-Wave Model with AOR Tasking.....	63
4.6.6 2-Stage/1-Wave Model with AOR Tasking and ISR Collection.....	69
4.7 JAO Scalability Assessment.....	75

5	<i>Experimentation Results</i>	78
5.1	Demonstration Scenario	78
5.2	1-Stage/1-Wave Problem	80
5.3	2-Stage/1-Wave with Retasking Problem	82
5.4	2-Stage/2-Wave Problem	86
5.5	2-Stage/1-Wave Model AOR Tasking Under Uncertainty Problem.....	92
6	<i>Conclusions</i>	100
7	<i>References</i>	103
8	<i>Appendices</i>	105
8.1	Types of Control.....	105
8.2	AEC 2000	105
8.3	AEC 1999	105
8.4	ACC 2001	105
8.5	SPIE 2001	105

LIST OF FIGURES

Figure 1 Key Steps in the Current JAO Process Limit Agility and Responsiveness.....	3
Figure 2 Autonomous Jumps Represent Interactions with Air Packages	12
Figure 3 Autonomous and Controlled Jumps Associated with Targets.....	13
Figure 4 Autonomous and Controlled Jumps Associated with Threats.....	14
Figure 5 SAM Engagement Model Geometry.	23
Figure 6 ALPHATECH's BMC ³ Development Environment GUI.....	25
Figure 7 Discrete Event Simulation Framework	26
Figure 8 Distribution of Controller Processing Across Multiple Platforms	28
Figure 9 SDP Recursion.....	31
Figure 10 NDP Solution Structure.....	32
Figure 11 Reduced-Order JAO Scenario Used for Proof of Concept Experimentation	34
Figure 12 Proof of Concept Performance ADP for Reduced-Order JAO Scenario	35
Figure 13 Scalability Assessment of Proof of Concept ADP Implementation.....	35
Figure 14 Hybrid, Multi-Rate Control Architecture.....	37
Figure 15 ADP Complexity Mitigation Approach	38
Figure 16 Enabling Technologies Investigated to Mitigate ADP Complexities for JAO Problem	38
Figure 17 ADP Research Framework in the Context of the Hybrid, Multi-Rate Control Architecture.....	39
Figure 18 TCT Retasking Radius	50
Figure 19 1-Stage Prediction	56
Figure 20 2-Stage Retasking Prediction	57
Figure 21 Retasking Approximations	58
Figure 22 2-Stage/2-Wave Predictor	59
Figure 23 Random Sampling for 2-Stage/2-Wave Prediction	60
Figure 24 Certainty Equivalent Approximation for 2-Stage/2-Wave Prediction	61
Figure 25 Aggregate Open-loop Approximation for 2-Stage/2-Wave Prediction.....	62
Figure 26 4-Stage/2-Wave Prediction.....	63
Figure 27 2-Stage AOR Predictor.....	65
Figure 28 Random Sampling for AOR Prediction.....	66

Figure 29 Certainty Equivalent Approximation for AOR Prediction	67
Figure 30 Partial Open-Loop Approximation for AOR Prediction	68
Figure 31 Information Dynamics.....	70
Figure 32 AOR Prediction with Partial Information.....	71
Figure 33 Random Sampling for AOR Prediction with ISR Uncertainty	72
Figure 34 Certainty Equivalent Approximation for AOR Prediction with ISR Uncertainty.....	73
Figure 35 Partial Open-loop Approximation for AOR Prediction with ISR Uncertainty	74
Figure 36 ADP Algorithms Developed and Implemented in Hybrid, Multi-Rate Control Architecture.....	75
Figure 37 Scalability Assessment of ADP Algorithms Developed and Implemented for Hybrid, Multi-Rate Control Architecture (Single CPU)	76
Figure 38 Scalability Assessment of ADP Algorithms Developed and Implemented for Hybrid, Multi-Rate Control Architecture (125 CPUs).....	77
Figure 39 Demonstration Scenario Used for Experimental Evaluation.....	79
Figure 40 Battlespace State Propagation Diagram for Known Initial State	81
Figure 41 Battlespace State Propagation Diagram for Unknown Initial State	81
Figure 42 Battlespace State Propagation for 2-Stage/1-Wave Retasking Problem	82
Figure 43 Prediction Accuracy of Different Design Models for the 2-Stage/1-Wave Retasking Problem with Known Initial State x_0	83
Figure 44 Prediction Accuracy of Different Design Models for the 2-Stage/1-Wave Retasking Problem with Unknown Initial State x_0	83
Figure 45 Behavioral Comparison of Proactive Versus Reactive Control Strategy for 2-Stage/1- Wave Retasking Problem.....	84
Figure 46 Control Performance for the 2-Stage/1-Wave Retasking Problem	85
Figure 47 Controller Computational Performance for the 2-Stage/1-Wave Retasking Problem	86
Figure 48 Battlespace State Propagation for 2-Stage/2-Wave Problem	88
Figure 49 Prediction Accuracy of Different Design Models for the 2-Stage/2-Wave Problem	88
Figure 50 Behavioral Comparison of Proactive Versus Reactive Control Strategy for 2-Stage/2- Wave Problem.....	90
Figure 51 Control Performance for the 2-Stage/2-Wave Problem	90
Figure 52 Controller Computational Performance for the 2-Stage/2-Wave Problem	91

Figure 53 Modified Demonstration Scenario Used for AOR Tasking Under Uncertainty Problem	93
Figure 54 Markov Transient Response of Known SAM Site Status Distribution.....	94
Figure 55 Battlespace State Propagation for 2-Stage/1-Wave AOR Tasking Problem with ISR Collection and Degradation	95
Figure 56 Prediction Accuracy of Different Design Models for the 2-Stage/1-Wave AOR Tasking Under Uncertainty Problem	96
Figure 57 Behavioral Comparison of Proactive Versus Reactive Control Strategy for 2-Stage/1- Wave AOR Tasking Under Uncertainty Problem	97
Figure 58 Control Performance for the 2-Stage/1-Wave AOR Tasking Under Uncertainty Problem	98
Figure 59 Controller Computational Performance for the 2-Stage/1-Wave AOR Tasking Under Uncertainty Problem	98

1 EXECUTIVE SUMMARY

The current process for planning of missions in Joint Air Operations (JAO) is slow and manually intensive. This research, performed by ALPHATECH, focused on the problem of providing military commanders with real-time, optimal control of military air-to-ground operations through the use of fast, near optimal mission replanning, using control algorithms that anticipate possible mission modifications due to uncertain future events for a 24-hour segment of a JAO campaign. The primary benefit of this technology is agile and stable control of distributed, dynamic military operations conducted in inherently uncertain, hostile, and rapidly changing environments. The goal of the JAO controller is to achieve specified Joint Force Air Component Commander (JFACC) objectives while minimizing the friendly asset losses. The controller generates or updates mission definitions for both base assets and airborne assets. The mission definition includes a target, mission waypoints, strike package composition, weapon composition, and desired time-on-target. Key features of the JAO environment include risk and reward that are dependent on package composition. The outcome of JAO events — target destruction, threat destruction, friendly asset attrition, emerging targets and threats — are uncertain, and thus missions must be adapted based on the observed outcomes. Due to the potential scarcity of resources, efficient resource utilization and adaptation to emerging battlespace conditions are paramount to assure a successful mission.

In order to develop controllers which address the uncertain, dynamic nature of the JAO statement, we adopted a framework for dynamic decision making under uncertainty, known as Markov Decision problems. In this framework, optimal decisions are chosen based on the most recent information, and the selected decisions must hedge against possible future contingencies. This explicit modeling of future contingencies results in *proactive* versus *reactive* control behaviors. This proactive attribute is desirable for stable and agile control of the JAO enterprise because future information arrival and control opportunities are dependent on stringent spatial, temporal, and coordination constraints.

The principal approach for control design using Markov Decision problems is the Stochastic Dynamic Programming (SDP) algorithm. However, it is well known that this approach suffers from the *curse of dimensionality* and is intractable (lacks scalability) for realistic sized problems. Thus, our investigations focused on developing Approximate Dynamics Programming (ADP) strategies that provide the desirable *proactive* control behaviors but can be

computed in near real-time for realistic JAO scenarios. The control design technology is based on combining hybrid state modeling techniques for developing statistical dynamical models relating mission decisions to evolution of objects in the battlespace, together with ADP control design techniques that have demonstrated real-time, proactive performance for other relevant military problems. In our investigations, we developed a broad spectrum of ADP control techniques for the JAO problem, and evaluated their relative performance and scalability. The technical accomplishments of this research are summarized below:

- Translated the JAO Control Enterprise into a Dynamical Hybrid State, Discrete Event, Stochastic Decision Making Problem
- Integrated Emerging ADP Technologies into JAO Feedback Controllers
- Experimentally Demonstrated the Benefits of Feedback Control
- Experimentally Demonstrated Benefits of Approximate Optimal Control
- Developed Innovative Hybrid, Multi-Rate Control Architecture
- Developed Computationally Efficient Control Algorithms that Produce Operationally Consistent Behaviors
- Extended Control Algorithms to Accommodate Hierarchical Mission Tasking and ISR Information Collection

The control algorithms developed in these investigations achieve the desired research objective of automating military operations planning to provide real-time, near-optimal control strategies that achieve operational objectives while minimizing asset losses. Adopting a hybrid, multi-rate control architecture permitted the tailored application of control to the operational situation at hand; faster control was used when actions had relatively local effects (e.g. a mission detour or divert), followed by slower modifications to address overall mission strategy. Given this architecture, a spectrum of ADP control strategies were developed and implemented that produce immediate retasking or abort decisions to preplanned multiple wave tasking. The solution quality and computation performance of these algorithms was tested and verified in a JAO simulator. It was shown through experimentation that the ADP strategies were able to produce operationally consistent, proactive control strategies that anticipated likely contingencies and positioned assets for opportunities of recourse all in either real-time or near real-time. Furthermore, a scalability assessment illustrated that many of the ADP controllers could provide near real-time performance for scenarios with 250 targets with some modest parallel computatio.

2 INTRODUCTION

2.1 BACKGROUND

The current process for JAO planning and execution—based on the steps of Strategy Development, GAT (Guidance, Apportionment, and Targeting), MAAP (Master Air Attack Planning), and Air Tasking Order (ATO) Production—is illustrated in Figure 1. The product is published every 24 hours. End-to-end development time typically requires 72 hours.

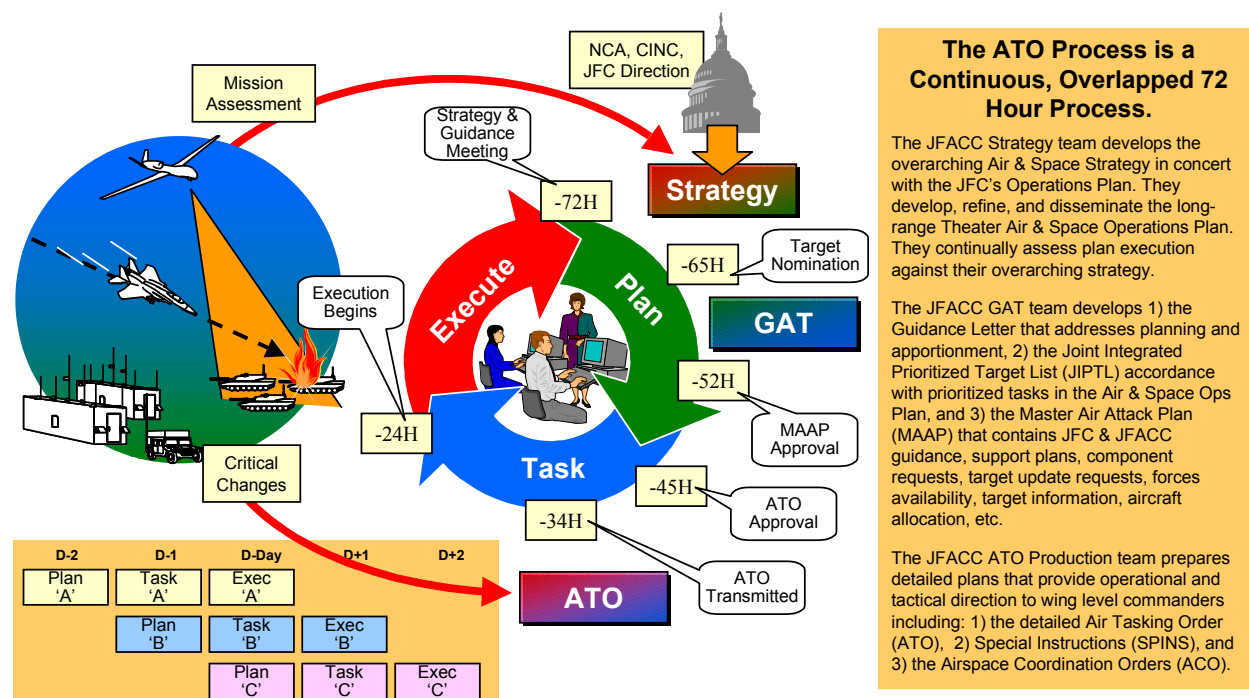


Figure 1 Key Steps in the Current JAO Process Limit Agility and Responsiveness

The heavily sequential nature of the current JAO process hinders the JFACC's ability to operate within the decision cycles of our adversaries. Moreover, brute force attempts to adapt the current process (e.g., diverting resources to engage time-critical targets) often lead to unstable operations. Given the realities of the dynamic problem above, the current process suffers from the following limitations.

Lack of Agility: Today's JAO planning tools tend to produce tightly woven plans. Given the problem as stated by the user, planners recognize the dependencies between resources, and generate solutions that maximize effectiveness by pushing resources to limits. Alas, unanticipated changes to the plan can generate significant ripple effects due to strong dependencies. These dependencies are intrinsic to the JAO control problem: the delivery of a

missile to a target requires the coordination of multiple resources 1) to locate and identify the target, 2) to launch the weapon, 3) to provide safe ingress and egress, 4) to provide sufficient fuel to airborne assets and 5) to assess bomb damage.

Planners attempt to avoid ripple effects by: 1) scheduling redundancy into the plan; or when changes are needed, by 2) terminating execution of all portions of the plan related to the breakpoint, or 3) ignoring the dependencies and allow the other portions of the plan to continue. Redundancy implies inefficient use of assets, termination of a portion of a plan implies ineffective use of assets, and ignoring dependencies implies risk of instability, i.e., mission failure.

Ad Hoc Stability: Current systems rely on human operators to serve as a stabilizing force by assessing the likely impact of unanticipated events. Although human operators are *very* good at adapting to new situations, their performance degrades dramatically when they are overloaded with information and tasks. Since the effectiveness of the assessment depends vitally on the operator's insight into the plan, and on the time available to make a decision, the drawback to this approach is the highly subjective and non-comprehensive nature of the operator's assessment. In addition, this task often becomes a pacing task, preventing the approach from scaling to highly dynamic environments.

Ineffective Feedback: Current systems are ineffective at providing feedback to guide the use of JAO resources. Fortunately, more sensors will soon supply more timely information on the state of the battlespace. Given sufficient agility in the attack and sensor platforms, the challenge is to reengineer the JAO process to incorporate this information in a disciplined manner. This also requires the JAO planning process to actively guide the information collection process in support of mission execution by providing timely information need specifications.

Ineffective Use of Assets and Resources: Current systems tend to build redundancy into the plans in order to provide a degree of agility (e.g., place aircraft on "SCUD CAP" in case a time-critical target (TCT) appears). Newer concepts focus on diverting ongoing missions to deal with significant changes in the situation. Unfortunately, this agility often comes at the expense of major disruptions to the execution of the remainder of the plan. For example, the diversion of an electronic countermeasures mission to support a TCT kill mission may result in the cancellation of several planned missions—again, leading to ineffective use of our resources.

2.2 PROBLEM DESCRIPTION

Within the Joint Air Operations Enterprise model, ALPHATECH's research is focused on generating real-time, optimal control strategies for a 24-hour segment on a JAO campaign. It is assumed that some form of higher level decomposition of the battle space has been performed, and that an Area of Responsibility (AOR) has been defined that includes approximately 100 targets. The goal of the controller is to achieve the specified JFACC objective for the AOR while minimizing the friendly asset losses. The controller will generate/update mission definitions for both base assets and airborne. The mission definition includes a target, high-level waypoints, strike package composition, weapon composition, and desired time-on-target. Inputs to the controller include a known target list, available assets, known threats list, and some indication of the likelihood of emerging target and/or threats. Through the continuous gathering of Intelligence, Surveillance, and Reconnaissance (ISR) data, the control system will generate these mission definitions on a time scale of approximately 15 minutes. The updates can be as benign as continue current plan or as drastic as reroll some packages, abort others, or launch new packages.

Key features of the JAO environment of interest include risk and reward that are dependent on package composition and weaponeering. The outcome of JAO events — target destruction, threat destruction, friendly asset attrition, emerging targets and threats — are uncertain in realistic JAO environments. Finally, limited resources and dealing with emerging and threats are paramount to the successful execution of the JFACC objective.

2.3 THEORETICAL TECHNIQUE

To address the limitations of the current JAO control process, ALPHATECH will investigate the utility of a comprehensive new approach based upon modern control theory. Because of its complexity and the time critical nature of the decisions that must be made, automated decision aids must support JAO operators. Past attempts at developing such decision aids have been based upon planning technology. Unfortunately, the plans produced by these decision aids are often rendered obsolete by unforeseen events. This difficulty can be addressed by periodic replanning, either in full or (more commonly) by partial modification of the plan.

However, planning technology provides little guidance on 1) how this replanning should be conducted, nor 2) how plans can be made robust to the need for replanning and repair.

In contrast, control theory uses the idea of continuous feedback to minimize the impact of uncertainty. Uncertainty is explicitly represented in both dynamics and observation models. Feedback control laws select current decisions as a function of the current (estimated) state of the system. These control laws are selected to optimize a quantitative objective criterion, subject to constraints on controls, dynamics, and observability. Control laws for current decisions explicitly consider the fact that future decisions will be made in the same optimal manner, based upon state information available in the future [B96]. Thus control theory provides an extensive conceptual foundation for continuous JAO.

2.4 VALUE TO MILITARY

The goal of this research is to provide real-time dynamic control of military air operations via near optimal mission assignments, which are robust under replanning. The primary benefit of this technology is agile and stable control of distributed and dynamic military operations conducted in inherently uncertain, hostile, and rapidly changing environments.

Utilizing the available real-time information from the battle space, the propose algorithm generates mission assignments resembling an ATO to achieve a desired JFACC objective. The fact of using feedback to create or update mission assignment desensitizes the desired outcome to modeling error and uncertainties that are inherently associated with military air operations. Furthermore, by formulating the control problem as an optimal control problem, the recommendations will anticipate key uncertainties and provide opportunities for recourse. In fact, the optimal solution will achieve the desired JFACC objective by minimizing the operational cost and risk to our assets. As an example, the optimal solution can identify a critical communication linkage that if destroyed can achieve air superiority without having to destroy every component of an Integrated Air Defense System (IADS). Given the progression towards more Unmanned Air Vehicles (AUVs), the proposed system could be used to automatically provide the UAV with its mission. The benefits of this technology to the military is summarized below using the taxonomy presented in the BAA:

Increased Agility: Approximate optimal control techniques generate solution that permit opportunities of recourse for key uncertain JAO outcomes; thus, increasing the agility of JAO operations by proactively (versus reactively), consistently, and efficiently responding to changes in the environment.

Flexibility: This technology is applicable to a wide spectrum of military conflicts.

Stability: Feedback control provides an automated system to stabilize the JAO environment; thus, eliminating the need for ad hoc stabilization via human operators.

Effective Feedback: Feedback control provides an natural framework to fuse large volumes of data to produce a coherent control strategies.

Effective Use of Assets and Resources: Optimal control generates control solution that effectively use the available resources.

Automated Operations: Envisioned is a prototype system that fits into existing C2 AOC. This system would monitor the progression of the battle space and generate real-time control strategies. The system could automatically manage autonomous assets.

2.5 REPORT LAYOUT

This report has three primary sections. In Section 3, the details of the JAO plant dynamics are presented. Next, Section 4 contains a detailed discussion of the JAO controllers formulation and implementation. Finally, Section 5 summarizes the key experimental results for the dynamics and controllers that were implemented. In addition to these primary sections, a series of appendices are included to compliment the main body of the report.

3 JAO PLANT DYNAMICS

In this section, we develop the JAO plant dynamics in which air package composition and tasking are represented in the context of a Joint Force Air Component Commander (JFACC) air-to-ground problem. Our model provides a flexible representation of the JFACC problem that “matches” the fidelity of our control approach (and has evolved accordingly), allowing for efficient experimentation. The presentation of the JAO plant dynamics begins with a general discussion of the JAO air-to-ground problem. This discussion is followed by the presentation of the dynamics that were implemented for this research. Finally, the details of the JAO plant dynamics implementation will be presented.

3.1 JOA AIR-TO-GROUND PROBLEM DESCRIPTION

In this section, we describe the general JFACC air-to-ground problem from which we will distill the salient JAO plant dynamics in subsequent sections.

The objective of the JFACC strike mission planning problem is to maximize damage to enemy targets while minimizing losses of our own aircraft. Strike missions involve targets, air defense units, strike aircraft, and Suppression of Enemy Air Defense (SEAD) aircraft. We discuss the roles of each below.

Targets are objects the JFACC wishes to damage. The JFACC objectives may specify a desired effect such as destroying, temporarily disabling, or disrupting the target. Targets are vulnerable to strike aircraft, which are discussed below. Multiple strike aircraft or special tactics may be required to achieve the desired effect when targets are geographically dispersed (i.e., have multiple aim points). Targets may also be coupled (physically or logically), and the strike mission may need to achieve some threshold damage before attaining the JFACC's objective. Collateral damage is always a concern, and application of strike force above some threshold may lead to undesirable outcomes. Multiple strike missions may also need to be coordinated in time (i.e., sequenced or synchronized) to achieve the desired effects.

The true state of the target (including location, identity, function, and value) is dynamic and may not be known with certainty. Targets may move or hide, and their functions may change, making their prosecution more or less desirable over time. Strike mission planners do not necessarily know all potential targets beforehand, and they may elect to keep some strike forces in reserve to attack targets that were heretofore unknown. These dynamics imply that the targets vulnerability and value vary with time. A time is typically specified for which a specific

strike mission is likely to achieve the desired effect. In general, strike missions will need to be tailored depending on when and where the target will be prosecuted. Adversaries protect targets using a combination of passive and active defense measures. Passive measures make targets harder to kill and/or prosecute; they include hardening, cover, concealment, and deception (HCCD). Active defense directly attacks aircraft in the strike mission. We discuss Air Defense units next.

Air Defense (AD) units defend targets. The effectiveness of an AD unit depends on the type of AD unit, the type attacking aircraft, the tactics of both the aircraft and the AD unit, and a variety of external influences such as weather. AD units are vulnerable to various type of suppression, which may be available on the strike aircraft but are concentrated typically on SEAD aircraft (discussed below). Suppression may be in the form of munitions that destroy critical components of the AD unit. Although this is typically permanent, AD units may be repaired given enough time. Other types of suppression such as jamming, chaff, and decoys, temporally disable or deceive AD units, which make them less effective. The AD unit's effectiveness may be enhanced by strategic placement, such as protecting multiple targets or overlapping to protect a single target. AD units may also be connected or coupled which tend to improve their effectiveness and reduce their vulnerability.

Similar to targets, the true state of the AD units is dynamic and may not be known with certainty. The AD units may be known or unknown, move or hide, they may choose not to expose themselves to suppression aircraft, or may operate at a lower effectiveness to reduce vulnerability.

Strike aircraft prosecute targets. Prosecution encompasses a range of effects, from permanently destroying the target, to temporarily disabling it, to simply disrupting it's operation. However, strike aircraft are vulnerable to enemy Air Defense (AD). Therefore, the aircraft's effectiveness is judged by its ability to circumvent AD and achieve the desired effect on the target. To accomplish these objectives, the aircraft is configured with munitions and the ability to suppress enemy air defense.

Each Strike aircraft is configured with a limited number of munitions, which are non-renewable and may only be replenished at an airbase (provided the supply exists). Selection of the munitions, as well as the tactics used to deploy them, determine the effect on the target (i.e., destroy, disable, or disrupt) and depend on several external influences including terrain, type of

target, air defenses, potential for collateral damage, and weather. Much of the strike aircraft's ability to suppress enemy air defense also comes from nonrenewable resources such as chaff, flares, and decoys. Strike aircraft may also be equipped with self-protection jamming equipment; this equipment is not depletable, but has power and duty cycle constraints. We discuss renewable resources for suppressing enemy air defense further when we introduce Weasel aircraft below. Another important nonrenewable resource is fuel. Fuel may be replenished at appropriate facilities, including air bases and orbiting tanker stations. SEAD aircraft help the strike aircraft through the enemy's air defenses. Some SEAD aircraft are configured to physically attack air defense units; these are Weasel aircraft. Other SEAD aircraft are configured to jam air defense units (primarily air defense radars); these are Jammers.

Weasel aircraft are configured with munitions; these are depletable resources. Jammers are equipped with an array of jamming pods; these are not depletable, but have power and duty-cycle constraints.

SEAD aircraft are packaged with strike aircraft to support a particular mission or placed on CAP to support a group of missions in a particular area. Jammer aircraft are slower than strike and weasel aircraft, but their effect may cover a wide geographical area. Therefore, these aircraft are ideal for supporting multiple strike missions in a particular geographical area.

To achieve the JFACC objectives, air packages, i.e. team of aircraft that coordinate their activities and fly in either a loose or tight formation, must be composed, tasked, and retasked such that the maximum number of targets are prosecuted with minimal resource losses. In general, air package configuration number and type of aircraft and resources. Resources include munitions type and quantity, SEAD capability (including self-protection pods), SEAD configuration (e.g., frequency range), and extra fuel tanks. Air package tasking includes target designation, course routing information, and potential opportunities for future retasking. Retasking involves updating an air package tasking in flight, which is limited by a variety of constraints but may be valuable to high value, fleeting targets.

In the following, we develop our hybrid dynamical model for Joint Air Operations (JAO). We start with a general discussion of hybrid dynamical models. Based on this general discussion, we develop hybrid dynamical models for each of the objects in the JAO environment. Finally, we introduce a composite hybrid dynamical model that facilitates interaction among JAO

objects. The hierarchical structure of the composite hybrid dynamical model simplifies design and analysis, as well as permitting a more lucid exposition of this complex environment.

3.2 HYBRID DYNAMICAL MODEL FORMULATION

Consider the following hybrid dynamical system based on [Bran95]

$$H = (Q, \Sigma, A, G, V, C, F)$$

where Q is a discrete state space. Each state (or mode) $q \in Q$ is associated with a controlled dynamical system, $\Sigma_q \in \Sigma$, an autonomous jump set $A_q \in A$, and a controlled jump set $C_q \in C$.

The controlled dynamical system is given by $\Sigma_q = [X_q, \Gamma_q, \phi_q, U_q]$ where X_q is the state space, Γ_q is an appropriately defined timing map, ϕ_q represents the dynamics, and U_q is the continuous control set. If the state of the dynamical system enters $A_q \subset X_q$, the system jumps autonomously to some new mode $p \neq q$ according to a control mapping G_q , which is parameterized by the control set V_q . Alternatively, if the state enters $C_q \subset X_q$, the system may be instructed to jump to some new mode $p \neq q$ such that $X_p \subset F_q$.

3.2.1 Battlespace Objects

In this section, we develop the hybrid dynamical models for air packages, targets, and threats.

3.2.1.1 Air Packages

An air package dynamics will be using the hybrid dynamical representation in the above section. The discrete state

$$q \in Z_+^2 \times \{Ingress, Egress, Base\}$$

where Z_+^2 specifies the number of Strike and Weasel aircraft and $\{Ingress, Egress, Base\}$ identifies the current operating conditions of the air package; *Ingress* indicates that the air package is fully loaded and may be retasked, *Egress* indicates that the air package has already engaged its target and is not eligible for retask, and *Base* indicates that the air package's mission is completed and the associated aircraft are available for assembly into new air packages. The continuous dynamics of an air package are independent of the discrete state and are given by

$$\Sigma = [X, \Gamma, \phi, \mu(r)]$$

where $X \in \mathbb{R}^2$ is the (x, y) location of the air package, Γ is an appropriately defined timing map, ϕ is a constant velocity kinematic model, and $\mu(r): X \rightarrow U$ is a flight controller that maps the current state into a continuous control input $u \in U$ given the next waypoint r . Disturbances are neglected in the continuous dynamics. Waypoints may be introduced through either autonomous jumps via the control set V (e.g., last waypoint has been reach, pick-up the next waypoint or loiter) or through controlled jumps specified in C (e.g., retask or abort mission). We neglect disturbance with respect to waypoint selection, i.e., no navigational error.

The autonomous jump set A is similarly defined for all q , including waypoint arrival, which instigates subsequent actions. Autonomous jumps also represent various interactions when considered within a composite model; these jumps are summarized in Figure 2 and will be discussed in detail in Section 3.2.2 below. From this figure we see that *threat engagements* may diminish the aircraft in the air package, and a single engagement has the potential to destroy all aircraft; and *Target engagements* transition the air package from ingress to egress.

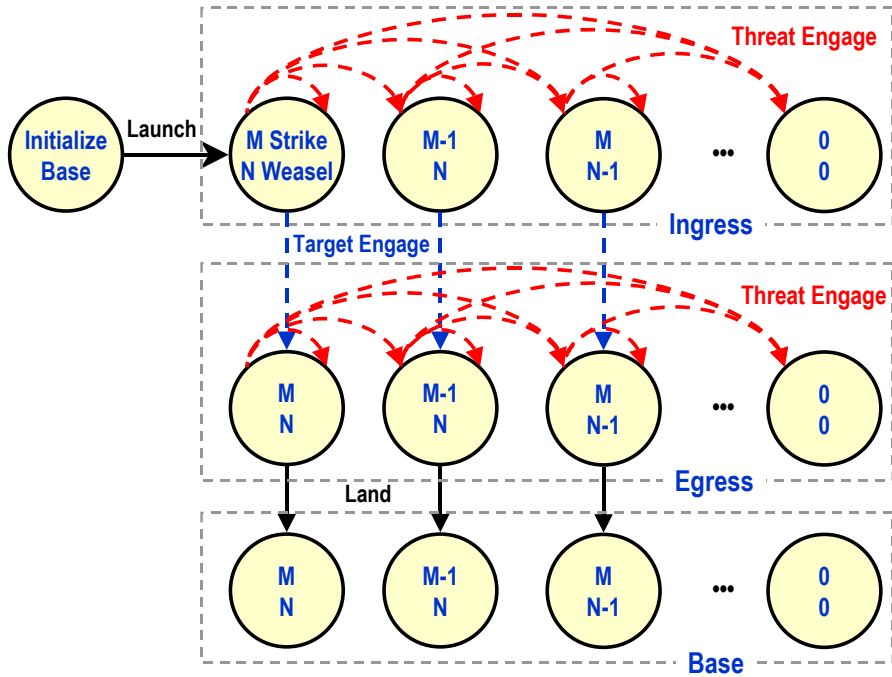


Figure 2 Autonomous Jumps Represent Interactions with Air Packages

The controlled jump set $C \subset X$ enables changes in tasking and/or configuration of air packages. The mapping $F: C \rightarrow 2^S$ where $S = X \times Q$ specifies a set of feasible jumps for each state $x \in C$ (e.g., reconfiguration at the base). We obtain the familiar control theoretic notation

by recognizing that in the absence of uncertainty, $F(x)$ is the set of possible control actions. In a stochastic setting, $F(x)$ is the set of all possible outcomes.

3.2.1.2 Observation Platforms

In the final experiments, which are run with partial information, observation platforms are used. These platforms are similar to the air package, but only interact with other objects through detection. Detection provides perfect information regarding specific objects in the JAO environment. We typically consider a global information model in which information is maintained independent of individual battlespace objects. However, the current implementation allows for a more general representation that accounts for distributed information and communication in which information models similar to the global information model are associated with each battlespace object.

3.2.1.3 Targets

Targets are also represented using the hybrid dynamical representation in introduction to Section 3.2. In general, targets have the following discrete state: $q \in \{Known, Unknown, Dead\}$, and may be initialized as known or unknown. The continuous state, for our purposes, is a fixed location, $X = \{x_{loc}\}$ for all q . Figure 3 illustrates the autonomous and controlled jumps associated with targets.

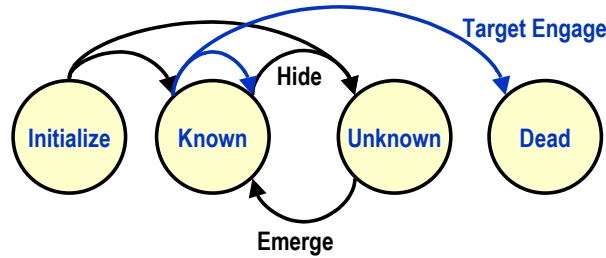


Figure 3 Autonomous and Controlled Jumps Associated with Targets

Autonomous jumps (blue) are only used to represent interactions within a composite model namely engagements with air packages. The controlled jumps (black) are characterized by the controlled jump sets, C_q , and the controlled jump transition maps, F_q . These controlled transitions are governed by a Poisson process, which may be considered a stochastic model of the adversary, allowing the targets to autonomously hide or emerge.

3.2.1.4 Threats

Using the hybrid dynamical representation, threats may be in one of five discrete states, $q \in \{Active, Inactive, Unknown, Repairing, Dead\}$, and may be initialized as active, inactive, or unknown. As with targets, the continuous state is a fixed location, $X = \{x_{loc}\}$ for all q . Figure 4 illustrates the autonomous and controlled jumps associated with threats.

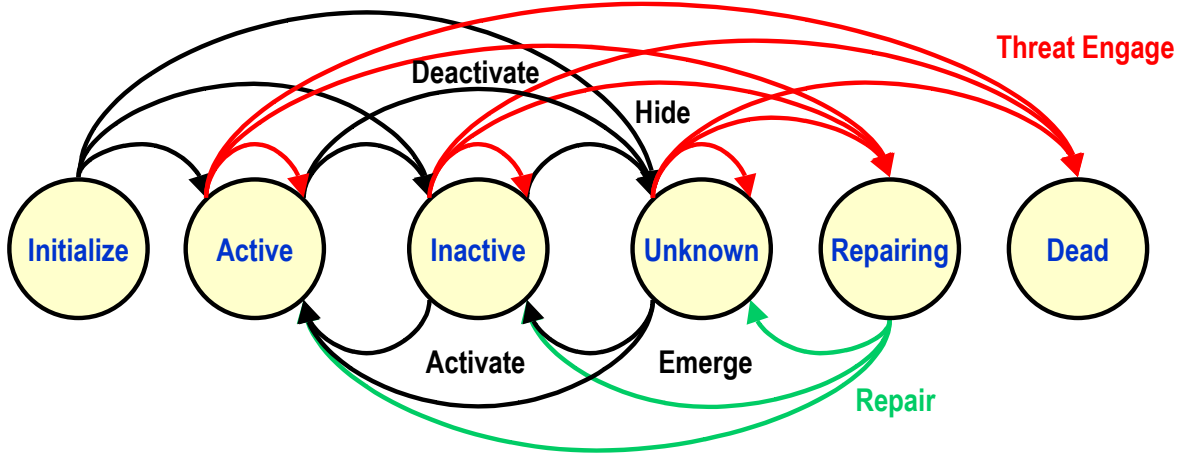


Figure 4 Autonomous and Controlled Jumps Associated with Threats

Autonomous jumps (red) are only used to represent interactions within a composite model namely engagements with air packages. The controlled jumps (black and green) are characterized by the controlled jump sets, C_q , and the controlled jump transition maps, F_q . These transitions are governed by a Poisson process, which may be considered a stochastic model of the adversary, allowing the threats to autonomously active, deactivate, hide, emerge, and repair. In addition, we consider the attack, i.e. SAM launch, transient state during engagement with an air package. We now characterize the autonomous behavior of a threat.

The state transitions that can occur between engagements correspond to *Emerge*, *Hide*, *Activate*, *Deactivate*, and *Repair*. Note that some transitions are only possible during an engagement (e.g., transition to *REPAIRABLE* or *DEAD* from any other state) while other transition are only possible between engagements (e.g., from *REPAIRABLE*). The autonomous transition dynamics are modeled as a stochastic timed automaton, equivalently a continuous time Markov model. Accordingly, a matrix that specifies the rate at which state transitions occur is defined. These rates depend on the time spent in each state and the transition probability, and are represented concisely in the following matrix:

	<i>Active</i>	<i>Inactive</i>	<i>Repairable</i>	<i>Unknown</i>	<i>Dead</i>
<i>Active</i>	•	$\mu_{Inactive Active}$	0	$\mu_{Unknown Active}$	0
<i>Inactive</i>	$\mu_{Active Inactive}$	•	0	$\mu_{Unknown Inactive}$	0
<i>Repairable</i>	$\mu_{Active Repairable}$	$\mu_{Inactive Repairable}$	•	$\mu_{Unknown Repairable}$	0
<i>Unknown</i>	$\mu_{Active Unknown}$	0	0	•	0
<i>Dead</i>	0	0	0	0	0

where • is defined so that the rows sum to 0. $\mu_{Inactive|Active}$ is the rate at which an active threat deactivates (i.e., observable but not radiating). $\mu_{Unknown|Active}$ is the rate at which an active threat hides (i.e., becomes unobservable). $\mu_{Active|Inactive}$ is the rate at which an inactive threat activates (i.e., starts radiating). $\mu_{Unknown|Inactive}$ is the rate at which an inactive threat hides. $\mu_{Active|Unknown}$ is the rate at which an unknown threat activates. The transition rate from *REPAIRED* are functions of the repair rate $\gamma_{REPAIRED}$:

$$\mu_{Active|Repairable} = P_{ACTIVE|REPAIRED} \cdot \gamma_{REPAIR}$$

$$\mu_{Inactive|Repairable} = P_{INACTIVE|REPAIRED} \cdot \gamma_{REPAIR}$$

$$\mu_{Unknown|Repairable} = P_{UNKNOWN|REPAIRED} \cdot \gamma_{REPAIR}$$

where the repair rate is weighted by the probability that the threat will immediately transition into the associated state. These probabilities sum to one

$$P_{ACTIVE|REPAIRED} + P_{INACTIVE|REPAIRED} + P_{UNKNOWN|REPAIRED} = 1$$

indicating that a repaired threat will become *ACTIVE*, *INACTIVE*, or *UNKNOWN*. The transition probabilities are derived from the rate matrix given the time interval Δt_{Inter} using the following matrix equation.

$$P_{Inter}(\Delta t_{Inter}) = e^{Q \cdot \Delta t_{Inter}}$$

where Q is the rate matrix and Δt_{Inter} is the time interval. For a given initial probability distribution over the state of the threat, denoted in vector form by σ_0^T , the distribution after Δt_{Inter} , denoted σ_F^T , is given by

$$\sigma_F^T = \sigma_0^T \cdot P_{Inter}(\Delta t_{Inter})$$

where Δt_{Inter} is the time interval between σ_0^T and σ_F^T . For large enough Δt_{Inter} , σ_F^T will achieve a steady state distribution, which is useful to characterize initial information regarding the threat (i.e., has not been observed or interacted with in a long time).

In the following, we derive the steady state distribution for a threat with the rate matrix Q discussed above. The continuous Markov model is given by

$$\dot{\sigma}^T = \sigma^T \cdot Q$$

with the constraint that σ^T is a proper distribution and the elements sum to one

$$\sum_i \sigma_i = 1$$

Steady state is achieved when $\dot{\sigma}^T = 0$. To compute the steady state probabilities, we define $b^T = [0 \ 0 \ 0 \ 0 \ 0 \ | \ 1]$ and $\tilde{Q} = [Q \ 1]$. The steady state distribution is then given by

$$\bar{\sigma} = (\tilde{Q} \cdot \tilde{Q}^T)^{-1} \tilde{Q} \cdot b$$

3.2.2 Battlespace Dynamics

When the individual battlespace objects are brought together into a setting where interactions are possible, we form a composite hybrid dynamical model with the same form as the individual objects. However, the composite JAO model subsumes the individual dynamical models and introduces interaction dynamics that depend on two or more objects within the JAO environment. For our purpose, interactions involving more than two participants are decomposed into a sequence of pairwise interactions.

Consider the composite hybrid dynamical model

$$H_c = (Q_c, \Sigma_c, A_c, G_c, V_c)$$

The discrete state space of the composite model is defined as the composition of individual discrete state spaces $Q_c = \prod_i Q_i$, where the subscript i denotes the i th battlespace object. Each state (or mode) of the composite model $q_c \in Q_c$ is associated with a composite dynamical system, $\Sigma_{q_c} \in \Sigma_c = \prod_i \Sigma_i$, a composite autonomous jump set $A_{q_c} \in A_c = \prod_i A_i$, and a controlled jump set $C_{q_c} \in C_c = \prod_i C_i$. For our purposes, the composite dynamical system Σ_{q_c} is

a non-interacting parallel composition with the exception of $X_{q_c} \in X_c = \prod_i X_i$, which is the composite state space on which the autonomous jumps sets $A_{q_c} \subset X_{q_c}$ are defined. Therefore, if the composite state of the dynamical system enters $A_{q_c} \subset X_{q_c}$ (an interaction), the system jumps autonomously to some new mode $p_c \neq q_c$ according to a control mapping G_{q_c} , which is parameterized by the control set V_q . These interactions effect one or more of the battlespace objects.

This hierarchical paradigm simplifies the modeling process by relating directly to the physical reality (i.e., objects and interactions) of the JAO environment. In the following sections, we discuss the interactions between air packages and threats, denoted threat engagement, and interactions air packages and targets, denoted target engagements.

3.2.2.1 Detection Interaction

A detection interaction occurs when an observation platform i is within sensor range of an observable object j

$$\|x_i - x_j\|_2 \leq R_i^2$$

where $x_i \in X_{q_c}$ is the location of an observation platform, $x_j \in X_{q_c}$ is the location of an observable object, and R_i is the sensor range associated with the observation platform i . The result of a detection interaction is a discrete change in the available information regarding the battlespace, but does not otherwise effect the objects. We assume perfect observations.

3.2.2.2 Target Engagement

A target engagement occurs when an air package i arrives at the location of target j

$$\|x_i - x_j\|_2 = 0$$

where $x_i \in X_{q_c}$ is the location of an air package and $x_j \in X_{q_c}$ is the location of a target. The result of a target engagement may only change the discrete state of the target, and does change continuous state of either object, i.e., $X_{q_c} = X_{p_c}$ if q_c is the discrete state prior to the engagement and p_c the discrete state after the engagement. The control set V_{q_c} controls the transition based on the attributes of the interacting objects (i.e., aim points and salvo) and the random component that determines the outcome.

The current model of the target engagement was proposed as part of the enterprise modeling effort. This is a static, analytic model that provides the probability of destroying the target as a function of the number of Strike aircraft in the air package, the number of aimpoint associated with the target and the strike salvo (i.e., the number of munitions). The probability that air package i destroys target j is given by:

$$P_{ij} = \left(1 - \exp\left(-\text{numStrike}_i / \text{aimPoints}_j\right)\right) \cdot \left(1 - \left(1 - P_D^{\text{Target}_j}\right)^{\text{strikeSalvo}}\right)$$

where aimPoints_j is an attribute of the Target representing the number of points that need to be hit on a particular target, thus influencing the effectiveness of the air package. $P_D^{\text{Target}_j}$ is the effectiveness of the aircraft's munition. strikeSalvo is number of munitions launched by the strike aircraft at the target. This simple static model is sufficient under for the current modeling assumptions (i.e., targets do not defend themselves).

3.2.2.3 Threat Engagement

A threat engagement occurs when an air package i is within range of threat j

$$\|x_i - x_j\|_2 \leq R_j^2$$

where $x_i \in X_{q_c}$ is the location of an air package, $x_j \in X_{q_c}$ is the location of a threat, and R_j is the range associated with threat j . The result of a threat engagement may change the discrete state of either or both participating objects, but does not change their continuous state, i.e., $X_{q_c} = X_{p_c}$ if q_c is the discrete state prior to the engagement and p_c the discrete state after the engagement. The control set V_{q_c} controls the transition based on the duration of the engagement and the random component that determines the outcome.

The threat engagement is broken into three parts, pre-engagement, engagement, and post-engagement. Pre-engagement transitions account for emergence or activation of a threat with the intent of attacking. Engagement transitions account for the interactive dynamics of an air package with an active and attacking threat. Post-engagement transitions account for hiding or shoot-and-scoot behaviors. Finally, we combine these segments in a single set of transition dynamics and adopt a concise matrix representation.

Pre-engagement transition accounts for emergence with the intent of attacking. The probability that a threat will attack is given by the probability of attacking from any given state (except *DEAD* and *REPAIRABLE*) times the probability of being in that state. Note, in order to attack, a threat in the *REPAIRABLE* state must have transitioned into another state prior to engagement, otherwise there is no guarantee that sufficient time has elapsed for the “repair” to take place. Therefore, the probability of attack is given by

$$P_{ATTACK} = P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}$$

This formulation assumes that there is some form of sensing available (possibly visual) that cues threat activation for subsequent attack from *INACTIVE* and *UNKNOWN* states. We also assume that an “attacking” threat is identifiable (e.g., by tracking radar, etc.), thus only threats that attacking are engaged. Therefore, if the threat does not attack, there are no pre-engagement transitions.

$$\begin{aligned} P_{ACTIVE}^{Pre} &= (1 - P_{ATTACK|ACTIVE}) \cdot P_{ACTIVE} \\ P_{INACTIVE}^{Pre} &= (1 - P_{ATTACK|INACTIVE}) \cdot P_{INACTIVE} \\ P_{REPAIRABLE}^{Pre} &= P_{REPAIRABLE} \\ P_{UNKNOWN}^{Pre} &= (1 - P_{ATTACK|UNKNOWN}) \cdot P_{UNKNOWN} \\ P_{DEAD}^{Pre} &= P_{DEAD} \end{aligned}$$

The engagement transition accounts for uncertainty associated with the interaction. In general, engagements may be considered *suppression* or *lethal*. Suppression engagements disable the threat, thus transitioning into the *REPAIRABLE* state while lethal engagements destroy the threat, thus transitioning into the *DEAD* state. We distinguish lethal engagements probabilistically with $P_{DEAD|SUCCESS}$ given a successful attack on the threat. This indicates that sufficient damage was done to transition the threat to *DEAD*. The threat must be *ACTIVE* after the pre-engagement transition to engage, and will transition to either *ACTIVE*, *REPAIRABLE*, or *DEAD* as a result of the engagement.

$$\begin{aligned} P_{ACTIVE}^{Engage} &= P_{FAIL|ATTACK} \cdot P_{ATTACK} + P_{ACTIVE}^{Pre} \\ P_{REPAIRABLE}^{Engage} &= (1 - P_{DEAD|SUCCESS}) \cdot P_{SUCCESS|ATTACK} \cdot P_{ATTACK} + P_{REPAIRABLE}^{Pre} \\ P_{DEAD}^{Engage} &= P_{DEAD|SUCCESS} \cdot P_{SUCCESS|ATTACK} \cdot P_{ATTACK} + P_{REPAIRABLE}^{Pre} \end{aligned}$$

The engagement dynamics are characterized by $P_{FAIL|ATTACK}$ and $P_{SUCCESS|ATTACK} = 1 - P_{FAIL|ATTACK}$, which depend on the interacting air package. Threat that are initially *INACTIVE*, *UNKNOWN*, or *DEAD* do not take part in the engagement.

$$\begin{aligned} P_{INACTIVE}^{Engage} &= P_{INACTIVE}^{Pre} \\ P_{UNKNOWN}^{Engage} &= P_{UNKNOWN}^{Pre} \\ P_{DEAD}^{Engage} &= P_{DEAD}^{Pre} \end{aligned}$$

Post-engagement transition accounts for deactivating/hiding behaviors after engagement.

$$\begin{aligned} P_{ACTIVE}^{Post} &= P_{ACTIVE}^{Engage} - (P_{DEACTIVATE} + P_{HIDE}) \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK} \\ P_{INACTIVE}^{Post} &= P_{DEACTIVATE} \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK} + P_{INACTIVE}^{Engage} \\ P_{UNKNOWN}^{Post} &= P_{HIDE} \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK} + P_{UNKNOWN}^{Engage} \end{aligned}$$

where $P_{DEACTIVATE} + P_{HIDE} \leq 1$.

If the threat is *UNKNOWN*, *REPAIRABLE*, or *DEAD*, there is no post-engagement transition.

$$\begin{aligned} P_{REPAIRABLE}^{Post} &= P_{REPAIRABLE}^{Engage} \\ P_{DEAD}^{Post} &= P_{DEAD}^{Engage} \end{aligned}$$

We combine pre-engagement, engagement, and post-engagement transition probabilities into a concise matrix representation. First, the transition probabilities are aggregated.

$$\begin{aligned}
P'_{ACTIVE} &= (1 - P_{DEACTIVATE} - P_{HIDE}) \cdot P_{FAIL|ATTACK} \cdot \left(\frac{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}}{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}} \right) + (1 - P_{ATTACK|ACTIVE}) \cdot P_{ACTIVE} \\
P'_{INACTIVE} &= P_{DEACTIVATE} \cdot P_{FAIL|ATTACK} \cdot \left(\frac{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}}{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}} \right) + (1 - P_{ATTACK|INACTIVE}) \cdot P_{INACTIVE} \\
P'_{REPAIRABLE} &= (1 - P_{DEAD|SUCCESS}) \cdot P_{SUCCESS|ATTACK} \cdot \left(\frac{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}}{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}} \right) + P_{REPAIRABLE} \\
P'_{UNKNOWN} &= P_{HIDE} \cdot P_{FAIL|ATTACK} \cdot \left(\frac{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}}{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}} \right) + (1 - P_{ATTACK|UNKNOWN}) \cdot P_{UNKNOWN} \\
P'_{DEAD} &= P_{DEAD|SUCCESS} \cdot P_{SUCCESS|ATTACK} \cdot \left(\frac{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}}{P_{ATTACK|ACTIVE} \cdot P_{ACTIVE} + P_{ATTACK|INACTIVE} \cdot P_{INACTIVE} + P_{ATTACK|UNKNOWN} \cdot P_{UNKNOWN}} \right) + P_{DEAD}
\end{aligned}$$

This set of equations may be concisely represented in matrix form as follows

$$\begin{bmatrix} P'_{ACTIVE} \\ P'_{INACTIVE} \\ P'_{REPAIRABLE} \\ P'_{UNKNOWN} \\ P'_{DEAD} \end{bmatrix}^T = \begin{bmatrix} P_{ACTIVE} \\ P_{INACTIVE} \\ P_{REPAIRABLE} \\ P_{UNKNOWN} \\ P_{DEAD} \end{bmatrix}^T \cdot \begin{bmatrix} P_{Active|Active} & P_{Inactive|Active} & P_{Repairable|Active} & P_{Unknown|Active} & P_{Dead|Active} \\ P_{Active|Inactive} & P_{Inactive|Inactive} & P_{Repairable|Inactive} & P_{Unknown|Inactive} & P_{Dead|Inactive} \\ 0 & 0 & P_{Repairable|Repairable} & 0 & 0 \\ P_{Active|Unknown} & P_{Inactive|Unknown} & P_{Repairable|Unknown} & P_{Unknown|Unknown} & P_{Dead|Unknown} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where the individual terms are defined as follows:

$$P_{Active|Active} = (1 - P_{DEACTIVATE} - P_{HIDE}) \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|ACTIVE} + (1 - P_{ATTACK|ACTIVE})$$

$$P_{Inactive|Active} = P_{DEACTIVATE} \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|ACTIVE}$$

$$P_{Repairable|Active} = (1 - P_{DEAD|SUCCESS}) \cdot P_{SUCCESS|ATTACK} \cdot P_{ATTACK|ACTIVE}$$

$$P_{Unknown|Active} = P_{HIDE} \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|ACTIVE}$$

$$P_{Dead|Active} = P_{DEAD|SUCCESS} \cdot P_{SUCCESS|ATTACK} \cdot P_{ATTACK|ACTIVE}$$

$$\begin{aligned}
P_{Active|Inactive} &= (1 - P_{DEACTIVATE} - P_{HIDE}) \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|INACTIVE} \\
P_{Inactive|Inactive} &= P_{DEACTIVATE} \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|INACTIVE} + (1 - P_{ATTACK|INACTIVE}) \\
P_{Unknown|Inactive} &= P_{HIDE} \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|INACTIVE} \\
P_{Repairable|Inactive} &= (1 - P_{DEAD|SUCCESS}) \cdot P_{SUCCESS|ATTACK} \cdot P_{ATTACK|INACTIVE} \\
P_{Dead|Inactive} &= P_{DEAD|SUCCESS} \cdot P_{SUCCESS|ATTACK} \cdot P_{ATTACK|INACTIVE} \\
P_{Repairable|Repairable} &= 1 \\
P_{Active|Unknown} &= (1 - P_{DEACTIVATE} - P_{HIDE}) \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|UNKNOWN} \\
P_{Inactive|Unknown} &= P_{DEACTIVATE} \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|UNKNOWN} \\
P_{Repairable|Unknown} &= (1 - P_{DEAD|SUCCESS}) \cdot P_{SUCCESS|ATTACK} \cdot P_{ATTACK|UNKNOWN} \\
P_{Unknown|Unknown} &= P_{HIDE} \cdot P_{FAIL|ATTACK} \cdot P_{ATTACK|UNKNOWN} + (1 - P_{ATTACK|UNKNOWN}) \\
P_{Dead|Unknown} &= P_{DEAD|SUCCESS} \cdot P_{SUCCESS|ATTACK} \cdot P_{ATTACK|UNKNOWN}
\end{aligned}$$

Alternately, $P_{Active|Active}$, $P_{Inactive|Inactive}$, $P_{Repairable|Repairable}$, and $P_{Unknown|Unknown}$ may be defined based on the other terms such that each row sums to 1. The parameters of this model are as follows:

- $P_{DEACTIVATE}$ is the probability that an “attacking” threat will deactivate after the engagement;
- P_{HIDE} is the probability that an “attacking” threat will hide (become unknown) after the engagement;
- $P_{FAIL|ATTACK}(\pi, \Delta t_{Engage})$ is the probability that the threat survived (i.e., suppression failed) the engagement and is a function of the engagement time, Δt_{Engage} , and the probabilistic state of the associated air package π (Discussed further in following paragraphs);
- $P_{SUCCESS|ATTACK}(\pi, \Delta t_{Engage}) = 1 - P_{FAIL|ATTACK}(\pi, \Delta t_{Engage})$;
- $P_{DEAD|SUCCESS}$ is the probability that a successful threat engagement will destroy the threat;
- $P_{ATTACK|ACTIVE}$ is the probability that an active threat will attack;
- $P_{ATTACK|INACTIVE}$ is the probability that an inactive threat will attack; and
- $P_{ATTACK|UNKNOWN}$ is the probability that an unknown threat will attack.

Consider $P_{FAIL|ATTACK}(\pi, \Delta t_{Engage})$, which depends on the state of the associated air package π and the engagement time Δt_{Engage} . This represents the actual engagement dynamics and is

based on an aggregate model of the shoot-look-shoot behavior of both the air package and the threat. In this model, the uncertainty associated with the threat engagement depends on the composition of the air package, the state of the threat (as a result of previous missions), and the duration of the engagement (determined from the geometry). The simplified geometry of this interaction is shown in Figure 5. The circle represents the footprint of the SAM's lethal range. Any route through the lethal range of the threat will result in an exposure time that indicates the duration of the engagement, ΔT . The state prior to the engagement is denoted π_0 , and the final state is denoted π_F . The engagement dynamics are defined within a composite model of the interacting air package and threat.

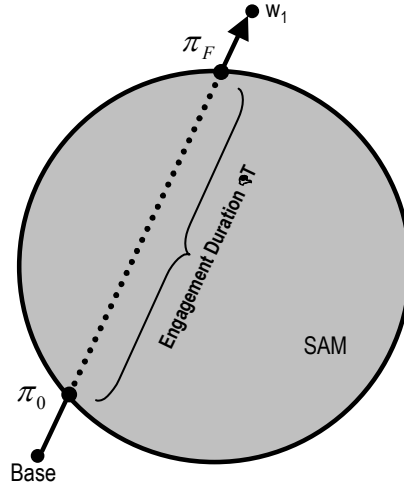


Figure 5 SAM Engagement Model Geometry.

The underlying dynamics of the threat engagement are represented by a set of event (e.g., SAM fires, Weasel fires) that are related by a continuous-time Markov chain, which may be solved analytically using

$$\pi_F = \pi_0 e^{Q\Delta T}$$

where Q is a transition rate matrix describing the engagement dynamics. Q is constructed from a set of parameters that describe the effectiveness of the SAM against each of the aircraft in the package, and the effectiveness of the weasel aircraft against the SAM. The following table summarizes these parameters

Parameter	Symbol	Notional Value
Weasel firing rate	γ_{weasel}	1 shot every 6 min (shoot-look-shoot)

system which may be easily extended or modified without changing the existing simulation framework.

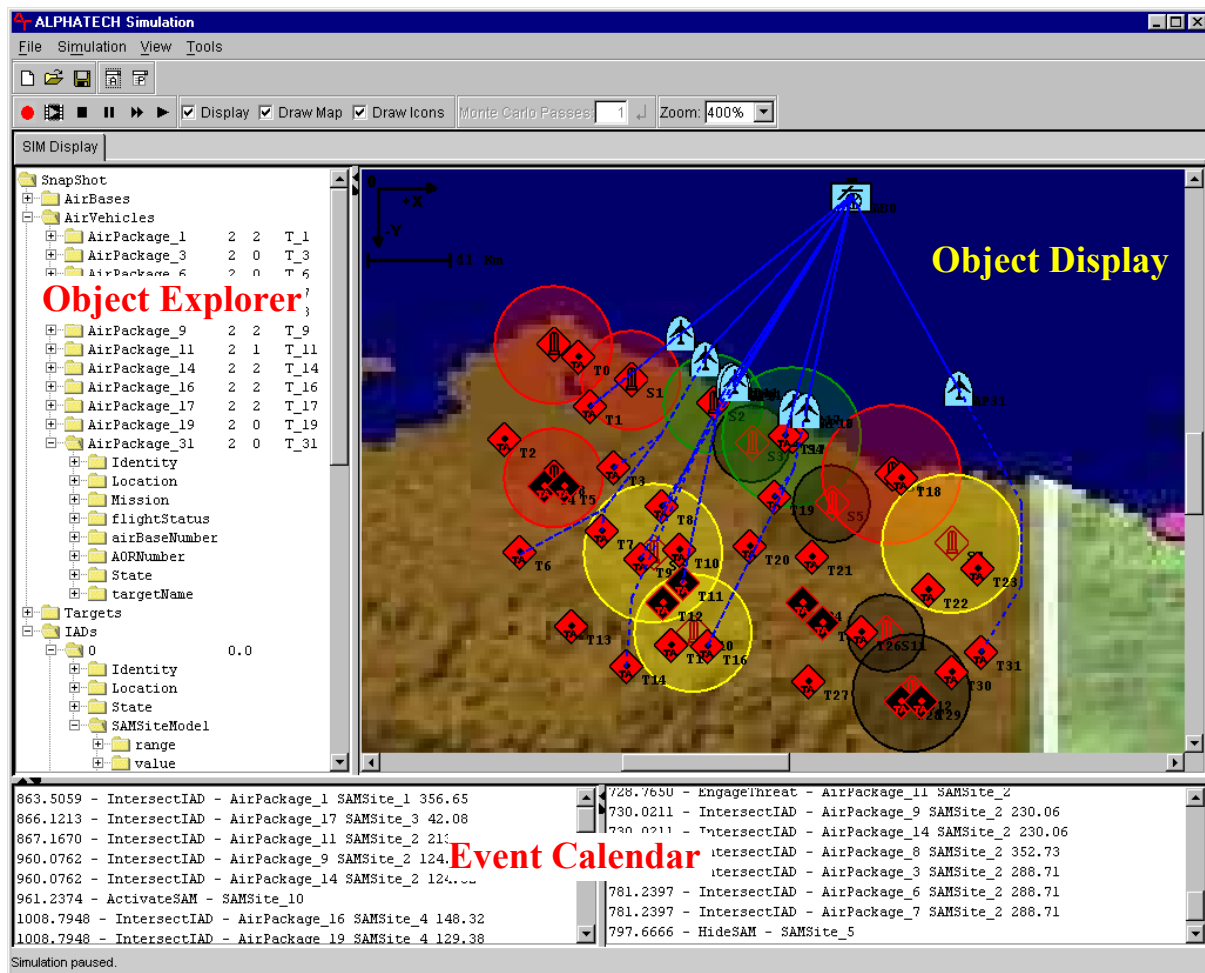


Figure 6 ALPHATECH's BMC³ Development Environment GUI

3.3.1 Discrete Event Simulation

ALPHATECH's BMC³ Development Environment is based on the discrete event architecture illustrated in Figure 7 [CassLaf99]. At the heart of the simulation is the event calendar, a time-sorted list of events waiting to be executed. The simulation is driven ahead in time by

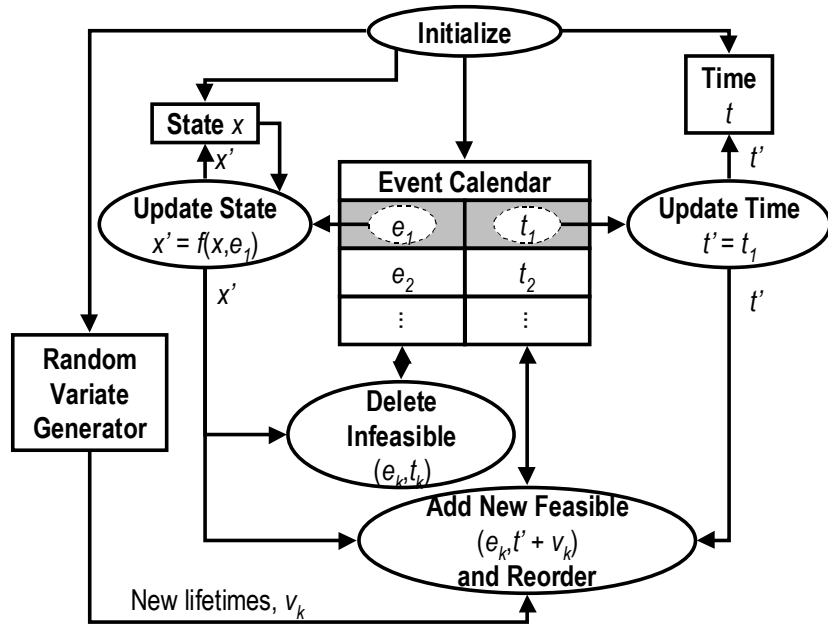


Figure 7 Discrete Event Simulation Framework
(copied from Reference[CassLaf99])

continually removing the next event from the calendar, updating the simulation time to that of the event by updating any continuous variables (such as position of air packages), and finally processing the event. The event is created with the knowledge of which simulation objects are involved in it, and any other information that may dynamically affect its behavior. Therefore, when the simulator instructs it to do so, the event can execute itself according to its inherent dynamics, using the current state of the simulation. Depending on the specific event's execution, it may update the state of the simulation by changing the state of any relevant simulation objects, adding new objects to the simulation, removing objects, or by adding new events to the calendar. This process continues until the event calendar is empty or any user defined stop conditions are met.

3.3.2 Plant/Controller Interface

When an event is executed that requires guidance from the controller, the plant accomplishes this via an interface to the controller. This interface between the plant and the controller maintains the integrity of each as separate software entities. When a control decision is needed, it is the plant's responsibility to collect the current known or estimated state of all simulation objects into a data structure required as input to the controller, which is sent via the

interface. Depending on arguments specified by the user, the information passed to the controller may be collected by the plant in two different ways. Most experiments performed assume perfect state information about all objects in the simulation. In this case, the plant extracts the true state of all objects from the simulation to create the data structure sent to the controller. When modeling information collection, however, a data structure is maintained with the current estimated or observed state of all simulation objects, which gets updated at the appropriate times by all surveillance aircraft. In this case, a new deep copy of this data structure is made by the plant and passed via the interface to the controller. It is then the controller's responsibility to return its guidance to the plant via the interface, using the same type of data structure it was passed. However, since the controller only has the ability to affect the actions of air packages, only that type of object is returned to the plant. If the controller decides to configure and task new air packages, objects are returned to instruct the plant how to do so. When it needs to modify the missions of existing air packages, any changes will be reflected in the corresponding air package objects' missions returned as guidance. The plant must then incorporate those changes into the real air packages. The controller does not have the ability to modify or create any of the simulation objects itself, but rather instructs the plant how to do so.

As described above, the plant and controller have a well defined client/server relationship. The plant's simulation runs as a client, which calls the controller as a server when it needs guidance. Each is a separate software entity, although they do not have to be run as separate processes. In fact, the plant and controller almost always run in the same process when performing experiments to save the overhead of communicating over a network. The interface described above makes it easy to maintain this relationship, as represents the network between the client and server.

3.3.3 Distributed Processing

Some of the controller algorithms implemented require a great deal of computation time. No matter how streamlined and efficiently the code is written, it is just infeasible to use only one process to get some control decisions due to the computational requirements. This becomes even more of a problem as the size of the problem being solved grows. One way of dealing with these issues of computation time and problem scalability is to distribute the controller's processing

duties among multiple CPUs, thereby decreasing the overall time it takes to close the loop. This can be done in a variety of ways, depending on how a specific controller works.

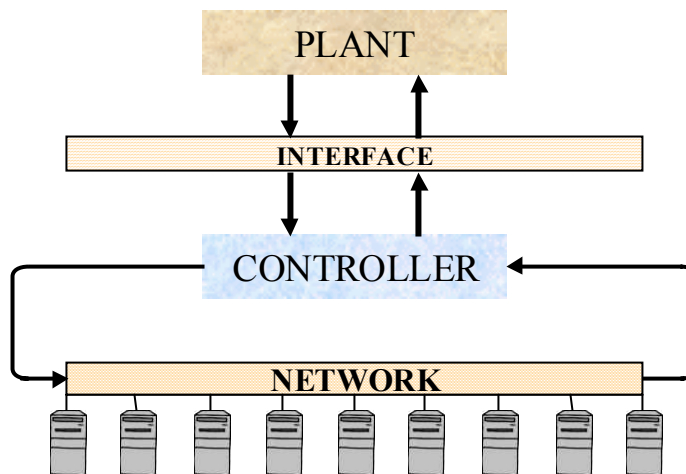


Figure 8 Distribution of Controller Processing Across Multiple Platforms

One example of a controller that benefits from distributing its processing is the Maximum Marginal Return (MMR) algorithm (see Section 4.3.1). Before allocating each aircraft, the MMR must evaluate the outcome of adding it to every air package in the current mission queue to determine how the asset can best be used. Rather than calling its predictor object to estimate the performance expectations one option (asset to air package assignment) at a time, the MMR can evaluate each option in parallel by distributing its predictor calls over multiple processors. In this case, if twenty options needed to be evaluated and ten machines were available, distributing the processing should be ten times faster than evaluating all twenty options in a single process.

4 JAO CONTROLLER FORMULATION

In this chapter, the JAO controller formulation details will be presented. The discussion begins with a general discussion of the control framework used for this research. Then, some proof of concept experimental results that guided our controller research framework will be presented. Having established the control and research framework, the details of the controller formulations and implementations will be presented. At the end of this chapter, a scalability analysis will be presented to assess whether near real-time computation performance is achieved.

4.1 CONTROL FRAMEWORK

The JAO environment is a uncertain dynamical system that has the following attributes: control decisions made over time; probabilistic transition from one state to the next, which is dependent on the choice of control; and rewards/costs that are accumulated during each transition, which is dependent on control and state transition outcome. Thus, the tasking of air packages in a JAO environment can be viewed as a sequential decision problem where each decision is based on the observations of certain discrete events.

This class of problems can be formulated as a Markov decision problem [B96]. The principal approach for solving such problems is Stochastic Dynamic Programming (SDP). Using the SDP formulation, an optimal control solution is computed off-line, and on-line computation is reduced to feedback rule evaluation or table lookup interpolation. However, it is well known that this approach suffers from the *curse of dimensionality* and is intractable for realistic sized JAO problems.

A subtle but significant attribute of the SDP formulation is that it produces control strategies that anticipate the effects of future contingencies, and evaluates the possible actions at all future states. The algorithm accomplishes this by modeling the future information arrival and control decisions. It is this fact that produces *proactive* versus *reactive* control behaviors. This proactive attribute is imperative for stable and agile control of the JAO enterprise because future information arrival and control opportunities are dependent on stringent spatial, temporal, and coordination constraints.

Given the strengths and weakness of the SDP formulation, there has been a great deal of research on Approximate Dynamic Programming (ADP) methods in recent years. These methods generally maintain the SDP structure, but use a variety of techniques to approximate the optimal cost-to-go. Accordingly, ADP algorithms have been applied to a variety of dynamic

decision problems [B99], [BTW97], [Patek99], [BC98], [BC99]. The goal of this research is to extend these ADP algorithms to the JAO context. In the following paragraphs, the general SDP and ADP formulations will be presented.

Consider a discrete-time version of a dynamic decision problem,

$$x_{k+1} = f_k(x_k, u_k, w_k)$$

where x_k is the state taking values in some set X_k , u_k is the control to be selected from a finite set $U_k(x_k)$, w_k is a random disturbance, and f_k is a given function. We assume that the disturbance w_k , $k=0,1,\dots$ has a given distribution that depends explicitly only on the current state and control.

Define a control policy, which is a sequence of feedback functions that map each state x_k to control u_k :

$$\pi_k = \{\mu_k(x_k), \mu_{k+1}(x_{k+1}), \dots, \mu_{k+N-1}(x_{k+N-1})\}$$

thus, the control at time k is $u_k = \mu_k(x_k) \in U_k(x_k)$. In the N -stage horizon problems considered herein, the single-stage cost function is denoted by $g_k(x_k, \mu_k(x_k), w_k)$ and the terminal cost function is denoted by $G_{k+N}(x_{k+N})$. The cost-to-go for policy π starting from state x_k at time k can be computed as follows:

$$J_k^\pi(x_k) = E \left\{ G_{k+N}(x_{k+N}) + \sum_{i=k}^{k+N-1} g_i(x_i, \mu_i(x_i), w_i) \right\}$$

and can be represented in the SDP recursion format as follows

$$J_k^\pi(x_k) = E \{ g_k(x_k, \mu_k(x_k), w_k) + J_{k+1}^\pi(f_k(x_k, \mu_k(x_k), w_k)) \}$$

for all k and with the initial condition

$$J_{k+N}^\pi(x_{k+N}) = E \{ G_{k+N}(x_{k+N}) \}$$

The N -stage, SDP solution is as follows

$$\pi_k^* = \arg \min_{\pi_k, u_k \in U_k(x_k)} E \{ g_k(x_k, \mu_k(x_k), w_k) + J_{k+1}^\pi(f_k(x_k, \mu_k(x_k), w_k)) \}$$

The computational feasibility of the SDP recursion depends on the number of future state realizations required to describe the system. Figure 9 provides a graphic illustration of the SDP recursion and tree structure of possible future state realizations. To illustrate the number of states required, assume that there are N targets, M air packages, and that we simplify physical position descriptions to describe only the N positions of the targets. Then, the number of possible combinations of positions is M^N , and the number of possible uncollected target sets at a given time is 2^N , resulting in numbers of states $(2M)^N$. For modest numbers of assets and targets, the number of states far exceeds our capability for computing and/or storing the resulting optimal decision rules.

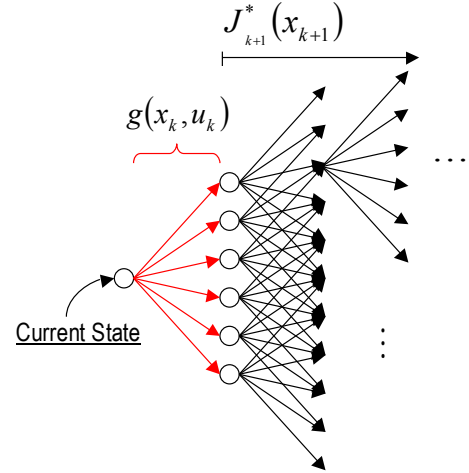


Figure 9 SDP Recursion

In an attempt to overcome the SDP *curse of dimensionality*, the ADP algorithm replaces the control mapping for times $k+1 \rightarrow k+N-1$ with some approximate mapping $\bar{\mu}_i(x_i)$. Additionally, the ADP algorithm is solved forward in time, and is computed at the actual state x_k versus all possible states at time k . As Appendix 8.1 presents, there are a variety of approaches to approximating future control maps. The approach adopted for this research is to generate an approximate control policy that maps a subset of future state realizations to a subset of control actions.

Thus, the ADP algorithm has the following policy:

$$\pi_k^{ADP} = \{u_k(x_k), \bar{\mu}_{k+1}(x_{k+1}), \dots, \bar{\mu}_{k+N-1}(x_{k+N-1})\}$$

Using this policy, the approximate optimal control solution at time k is

$$u_k^{ADP} = \arg \min_{u_k \in U_i(x_i)} E \left\{ g_k(x_k, u_k(x_k), w_k) + J_{k+1}^{\pi^{ADP}}(f_k(x_k, u_k(x_k), w_k)) \right\}$$

Thus, the ADP policy is a one step-look-ahead policy with the optimal cost-to-go approximated by the cost-to-go of the base policy. The ADP algorithm computes the best control at the current state x_k at time k by balancing the current cost with an approximate cost-to-go using approximations to model future control decisions. A graphical illustration of the ADP approach is illustrated in Figure 10 where the tree structure of the cost-to-go has been replaced with an approximate cost-to-go.

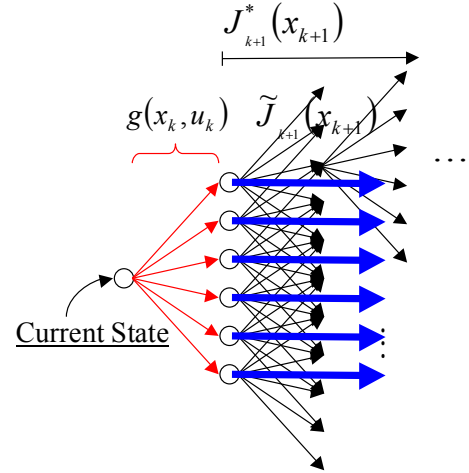


Figure 10 NDP Solution Structure

By approximating the future control maps and by solving the problem forward in time for the actual state x_k , the ADP algorithm significantly reduces the computational complexity of the SDP framework. Again, SDP considers all of the possible states and computes a tentative decision for each possible state, whereas ADP only computes decisions for states that actually occur in the scenario. Thus, the number of states considered by ADP is much smaller; however, the drawback of this approach is the solution must be determined in real-time.

Having defined the ADP framework, the difficulty remains of computing the expectation in the above optimization. Given the complexity of the JAO problem and the fact that the control solution will have to be computed in real-time, only an estimate of the expectation can be computed. Accordingly, the Q -factor is introduced:

$$Q(x_k, u_k) = E\{g_k(x_k, u_k, w_k) + J_{k+1}^{ADP}(f_k(x_k, u_k, w_k))\}$$

For the reasons stated above, only an estimate for the Q -factor $\hat{Q}(x_k, u_i)$ is obtained.

Thus, given the estimate Q -factor $\hat{Q}(x_k, u_i)$ corresponding to each candidate control

$u_k \in U_k(x_k)$, the ADP control at time k for state x_k is

$$u_k^{ADP} = \arg \min_{u_k \in U_k(x_k)} E\{Q(x_k, u_k(x_k))\}$$

In summary, the ADP algorithm provides the control framework for this research. This technique has been shown to illustrate operationally consistent, proactive behaviors for relevant military applications. The focus of this research is to extend the ADP method to the problem of

JAO, and in doing so, develop ADP algorithms that exhibit optimal behavior in real-time or near real-time for realistic JAO scenarios. As a final note, given the approximation illustrated above, the ADP algorithm is not as ambitious as the SDP, and only provides modest guarantees of near-optimality [BTW97]; in fact, it is an intermediate methodology between Model Predictive Control (MPC) and SDP.

4.2 PROOF OF CONCEPT EXPERIMENTS AND RESEARCH FRAMEWORK

In this section, the proof of concept experiments that guided the development of ADP control techniques for the JAO problem will be presented. Following the proof of concept experiment discussion, the research framework adopted for this program will be presented.

4.2.1 Proof of Concept Experimental Results

At the beginning of this research program, proof of concept experiments were performed for the purpose of identifying key technology gaps in the state-of-the-art of ADP with respect to the JAO domain. By identifying the key technology barriers, the research and development was focused on mitigating these barriers so as to satisfy the research objective of providing military commanders with real-time, near optimal control strategies for air-to-ground operations. In this section, the proof of concept experimental results that guided the development of ADP control techniques for the JAO problem will be presented. Since a majority of these initial experimental results have been documented in conference publications, most of the details are contained in the Appendices and the summary of results and lessons learned will be summarized here.

The approach used to establish the proof of concept experiments was to apply ADP techniques that have been applied to other relevant military problems. As part of AFOSR's New World Vistas (NWV) program, ALPHATECH performed basic research on ADP control techniques for the problem of sensor asset scheduling. Under this program, ADP control techniques were developed that optimize the collection of data by multiple sensor platforms based on requests of multiple end-users. The controller dynamically replans the paths of sensor platforms as the result of dynamic requests for data, failure of individual sensors (thereby providing fault tolerance), and failure of sensors to collect individual pieces of data due to unpredictable obscuration effects such as weather. Many of the results from this program are contained in two DARPA Advances in Enterprise Control papers that appear in Appendices 8.2 and 8.3.

Given this successful application of ADP techniques to military relevant problems, the next logical step was to apply the ADP techniques developed under the AFOSR program to the problem of orchestrating a 24 hour air-to-ground campaign in a risky environment. However, the multi-vehicle scheduling problem does not map one-to-one with the air-to-ground problem because this problem is much larger and contains a richer set of dynamics. For one, this problem requires the formation and tasking of air packages in an environment where risk and reward depend on the air package composition. Furthermore, since air packages pose a risk to enemy assets and enemy assets pose a risk to air packages, considerable coupling exists between battlespace assets. Given the fact that multiple turns of the aircraft will be required to achieve the operational objectives, this coupling remains dominant through the air-to-ground campaign.

For the proof of concept experiments, the rollout algorithm [B99], [BTW97] was chosen for implementation on a reduce-order JAO problem. The rollout algorithm—which has been used for a wide variety of dynamic decision problems [B99], [Patek99], [BC99], [BC98], [BC99]—is a technique that exploits knowledge of a suboptimal decision rule to obtain an approximate cost-to-go for use in ADP framework. The rollout algorithm approximates control mapping for times $k+1 \rightarrow k+N-1$ with a baseline heuristic $\bar{\mu}(x_i)$. Thus, the rollout algorithm computes the best control at the current state x_k at time k by balancing the current cost with an approximate cost-to-go using a baseline heuristic to model future control decisions. To generate the estimate of Q -factor, Monte Carlo evaluations were performed by simulating the base policy in real-time, i.e. simulation-in-the-loop.

The rollout algorithm is applied to a small JAO scenario, Figure 11, that includes limited assets, risk/reward that is dependent on package composition, basic threat avoidance routing, and multiple targets, some of which are fleeting and emerging. Simulation results illustrate the benefits of the approximate optimal control strategy. It is shown that the

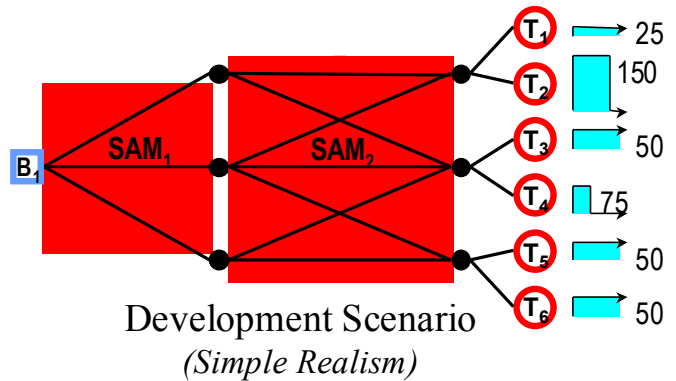


Figure 11 Reduced-Order JAO Scenario Used for Proof of Concept Experimentation

rollout strategy provides statistically significant performance improvements over strategies that do not anticipate future information arrival and control decisions. The performance improvements were attributed to the fact that the rollout algorithm is able to learn near-optimal behaviors—establishing combat air patrol over time critical areas, staging packages and opening attack corridors to manage friendly asset attrition, aggressively prosecuting fleeting targets, and reserving assets for contingencies that are not modeled in the baseline heuristic. The details of this experiment and the results obtained are contained in Appendix 8.4.

Having shown that ADP strategies can produce operationally consistent, proactive control solutions, the question remains whether this current ADP implementation using rollout is feasible for a realistic sized JAO scenario. Figure 13 illustrates the scalability assessment performed on this ADP implementation. From this figure, it is seen that that real-time computation performance is not feasible for larger scenarios. Furthermore, when considering a richer set of dynamics, it is expected that the computation complexity will grow by several orders of magnitude.

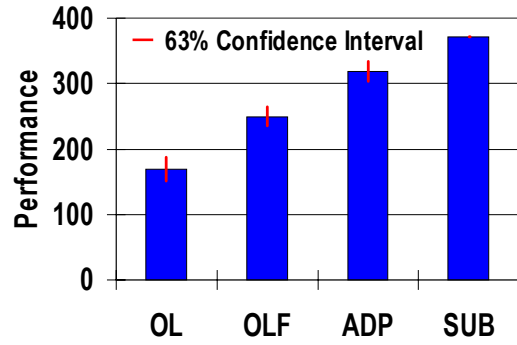


Figure 12 Proof of Concept Performance ADP for Reduced-Order JAO Scenario

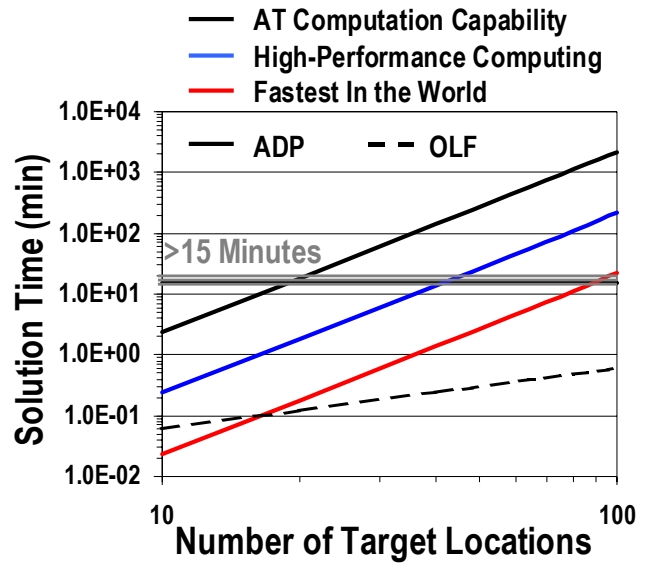


Figure 13 Scalability Assessment of Proof of Concept ADP Implementation

From this proof of concept experiment, the following lessons were learned relative to JAO control:

- **Proactive and Reactive Control** provides near-optimal performance in situations with **abundant opportunity and time** to react to **uncertain future information** arrival;
- **Proactive Control** provides near-optimal performance in situations with **limited opportunity and/or time** to react to **uncertain future information** arrival:
 - High attrition environment
 - Control response delays, i.e. inertia, $\geq$ significant event time-scales
 - Information delay
- **Key Proactive JAO Behaviors:** *Positioning assets now for future opportunity*
 - Preparing battlespace
 - Reserving assets
 - Geographically positioning assets
- **Computational Performance:**
 - ADP using combinatorial rollout to search control space and temporal rollout, i.e. simulation-in-the-loop, for prediction is **infeasible** for 100 DMPI scenario
 - The reactive controller implemented provides **rapid replanning**, and is **feasible** for 100 DMPI scenario and has considerable margin

In summary, ADP technique known as rollout, which was developed under the AFOSR NWV program, was applied to a reduced-order JAO scenario that includes limited assets, risk/reward that is dependent on mission composition, basic threat avoidance routing, and multiple targets, some of which are fleeting and emerging. Simulation results illustrate the benefits of the ADP control strategy. It is shown that the proactive ADP strategy provides statistically significant performance improvements over a reactive feedback strategy by developing operationally consistent control strategies that anticipate likely contingencies and position assets for opportunities of recourse. However promising these results are, the current implementation of the rollout algorithm is not computationally feasible for realistic JAO scenarios.

4.2.2 Control Research Framework

The proof of concept experiments in the above section highlighted ADP performance both in terms of behaviors and computation complexity. It was shown that ADP control strategies could produce proactive, operationally consistent behaviors but scalability remains the key technical barrier of using this technique for realistic sized JAO problems. As a result, the

bulk of this research was devoted toward reducing the computational complexity of the ADP approach while maintaining the operationally consistent behaviors. Based on the lessons learned, a two pronged approach for reducing the problem complexity was pursued:

1. Reduce control problem size where appropriate, and
2. Improved the efficiency of the ADP algorithms.

As discussed in the previous section, one of the lessons learned was that both reactive and proactive control provides near-optimal control strategies in situations where there is abundant opportunity and time to react to uncertain future information arrival. However, the reactive control approach can produce this solution in a fraction of the time that it takes the proactive controller approach. Thus, in situations where there is abundant opportunity and time to react to uncertain future information arrival, it makes sense to use a reactive versus proactive control approach. The lessons learned also identified situations in which a proactive control approach is required in order to proved near-optimal control strategies. These situations include environments that exhibit high attrition and significant control response and information delays relative to the battlespace time scale. With this intuition into the JAO control problem, a hybrid, multi-rate control architecture was adopted that tailors the application of control, i.e. reactive or proactive, at a time scale that is appropriate to the battlespace situation at hand.

Figure 14 illustrates the hybrid, multi-rate architecture chosen for this research. To implement this architecture, three types of information must be specified *a priori*. First, the significant events at which control loop closures are to occur must be defined. For each significant event, a set of control algorithms must be defined, and finally, the loop closure rate for each significant event must be defined. In this figure, event-based loop closures are denoted by **E** where temporal-based loop closures are denoted by **T**. By adopting this architecture, the control problem complexity is substantially reduced for situations that do not require advanced ADP algorithms.

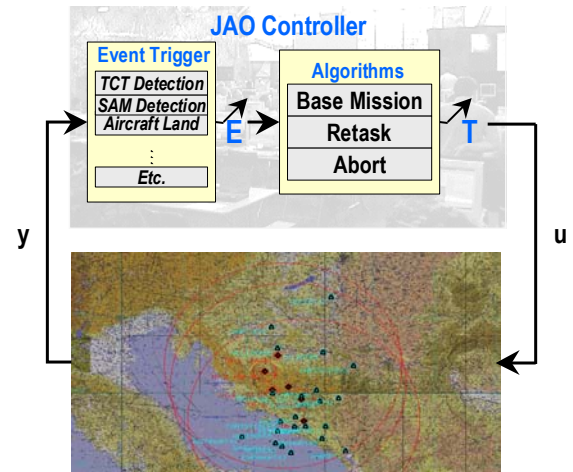


Figure 14 Hybrid, Multi-Rate Control Architecture

The implementation of this control architecture provided an immediate reduction in the control problem complexity for certain situations. However, as noted in the lessons learned, situations with limited opportunity and/or time to react to uncertain future information arrival require proactive control strategies to achieve optimal performance. Thus, for these situations, the complexity reduction must be achieved by developing fast and efficient ADP algorithms that still exhibit the desired operationally consistent behaviors. The approach adopted for developing efficient ADP algorithms was to exploit the natural decomposition, which is illustrated in Figure 15,

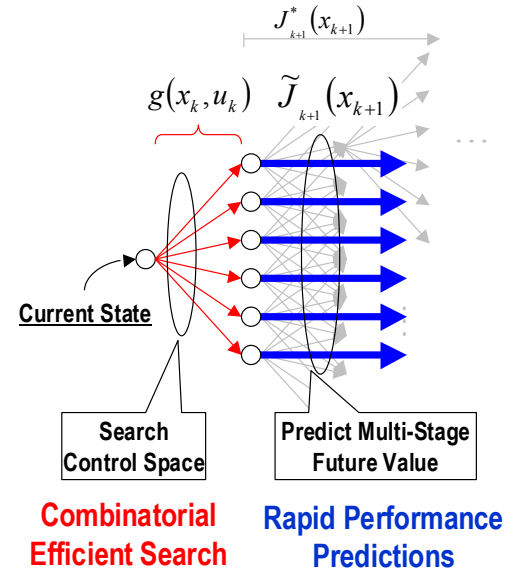


Figure 15 ADP Complexity Mitigation Approach

between the control space search and the performance prediction problems of the ADP framework. Given this natural decomposition, parallel research initiatives that focused on reducing the complexities of these problems were conducted. Figure 16 highlights some of the technologies that were investigated as part of the complexity mitigation.

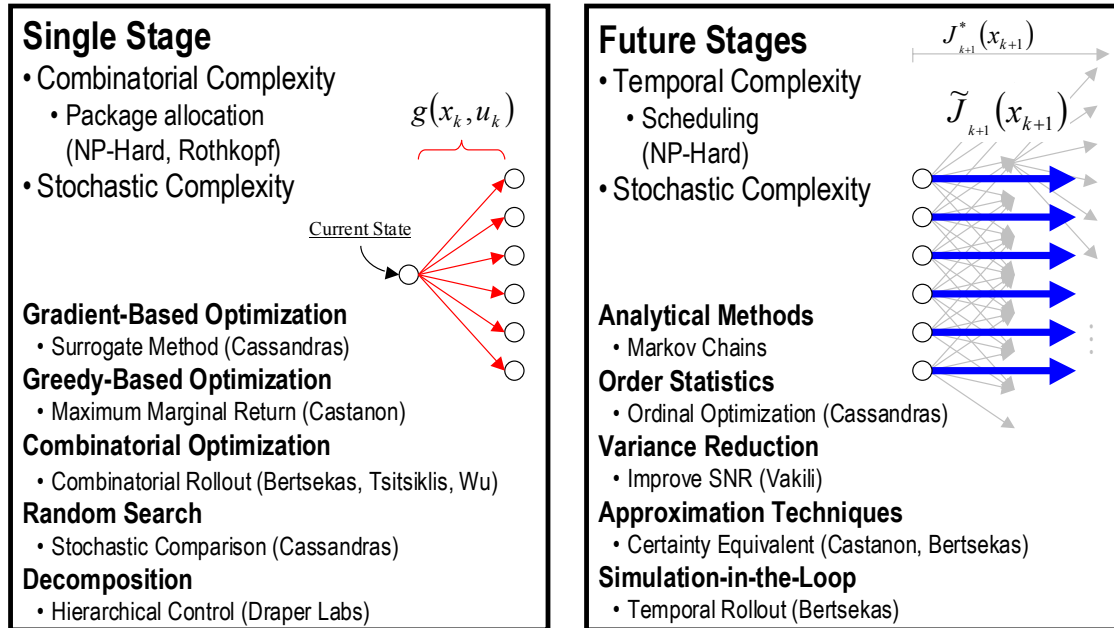


Figure 16 Enabling Technologies Investigated to Mitigate ADP Complexities for JAO Problem

In summary, the proof of concept experiments showed that ADP control strategies could produce proactive, operationally consistent behaviors but scalability remains the key technical barrier of using this technique for realistic sized JAO problems. As a result, the bulk of this research was devoted towards reducing the computational complexity of the ADP approach while maintaining the operationally consistent behaviors. One immediate complexity reduction was achieved by adopting a hybrid, multi-rate control architecture that tailors the application of control, i.e. reactive or proactive, for the battlespace situation at hand; however, additional complexity reduction is required. Thus, the research was focused on developing fast and efficient combinatorial assignment and prediction models that form the foundation of the ADP algorithm. Figure 17 illustrates the control architecture in the context of the ADP decomposition, i.e. assignment and prediction. It is the goal of this research to develop a spectrum of ADP control strategies that can be mixed and matched to provide a broad range of performance and computational complexity. The details of the different combinatorial assignment and analytic prediction models developed under this research follows in the subsequent sections.

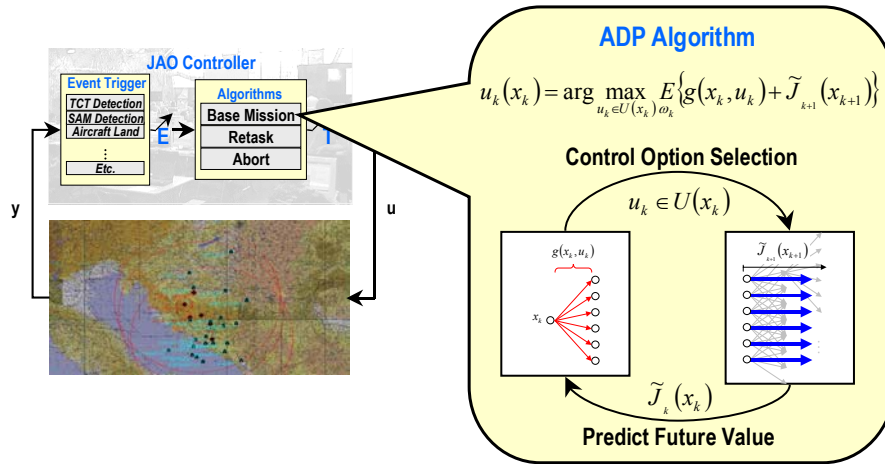


Figure 17 ADP Research Framework in the Context of the Hybrid, Multi-Rate Control Architecture

4.3 COMBINATORIAL OPTIMIZATION ALGORITHMS

In this section, the combinatorial optimization algorithms that were incorporated into our hybrid, multi-rate control architecture are presented. These algorithms were selected based on a trade study analysis of different combinatorial optimization approaches for a 1-stage problem.

Details of this trade study appear in Appendix 8.5. From this trade study, three approaches were chosen: combinatorial rollout, maximum marginal return, and the surrogate method. The formulation of each of these techniques will be presented in the subsequent sections. We first begin with a general discussion of combinatorial optimization.

As noted in Section 4.1, optimization of the Q -factor is fundamental to our control formulation. In a standard SDP implementation, the control space is enumerated, thus guaranteeing an optimal solution. However, in the JAO problem, the complexity of the control space and the requirements for near real-time computation make explicit enumeration infeasible for problems of interest. Therefore, in the course of our proof-of-concept experiments, we investigated several alternatives to direct enumeration. These alternatives are combinatorial optimization techniques, adapted to the structure of the JAO problem, which are approximate optimization techniques. Our rationale for using approximate combinatorial optimization is that the JAO problem, even in the absence of dynamics, is provably NP-Hard, as 3-D matching can be reduced in polynomial time to instances of the JAO problem. We consider three specific approximate combinatorial techniques:

- A greedy algorithm, based on maximum marginal return, using resource by resource decomposition;
- Combinatorial rollout, which is an approximate technique for incrementally building a solution;
- Surrogate method, which embeds the combinatorial optimization in a larger continuous space optimization.

We describe each of these techniques in greater detail below.

To better define the combinatorial problem, consider the problem of optimizing a known function $Q(x_k, u_k)$, over a set of feasible controls $u_k \in U(x_k)$ for a given state x_k . In the JAO case, each control corresponds to a set of resource allocation pairs $(N_i^{Strike}, N_i^{Weasel})$ to possible missions i ; that is,

$$u_k = \{(N_0^{Strike}, N_0^{Weasel}), (N_1^{Strike}, N_1^{Weasel}), \dots, (N_T^{Strike}, N_T^{Weasel})\}$$

in which N_i^{Strike} strike aircraft and N_i^{Weasel} weasel aircraft are assigned to the i -th target for all i .

Due to the potential coupling in effectiveness across missions which fly across similar air defenses, function Q cannot be decomposed as an additive sum of effectiveness across missions.

The combinatorial optimization problem becomes

$$\max_{u_i \in U(x_i)} Q(x_k, u_k)$$

subject to resource constraints on the total availability of strike and weasel aircraft, which may be written as

$$\begin{aligned} \sum_i N_i^{Strike} &\leq N^{Strike}(x_k) \\ \sum_i N_i^{Weasel} &\leq N^{Weasel}(x_k) \end{aligned}$$

where $N^{Strike}(x_k)$ and $N^{Weasel}(x_k)$ correspond to the availability of each aircraft type at a given state x_k .

4.3.1 Maximum Marginal Return

The first algorithm we discuss is the Maximum Marginal Return algorithm (MMR). The basis for this algorithm is the following optimization problem, which is a simplified mathematical model of the JAO problem that provides explicit approximations for the function $Q(x, u)$ for a single wave scenario.

Assume that we have present $k = 1, K$, K SAM sites in the scenario, $t = 1, K$, T targets, $w = 1, K$, W weasels, and $s = 1, K$, S strike aircraft in the scenario. The simplified model assumes that trajectories for each target t are known, and have a sequence of SAMs associated with each trajectory. When a weasel interacts with SAM k while headed for target t , the probability that the SAM is destroyed is given by q_{kt} . We make the probabilistic assumption that interactions between SAMs and weasels are independent events. In this case, given m weasels headed for target t , the probability that SAM k survives is given by:

$$P_s(k) = (1 - q_{kt})^m$$

Similarly, given a full set of missions, with m_t weasels headed for target t , the overall probability that SAM k survives is given by

$$P_s(k) = \prod_t (1 - q_{kt})^{m_t}$$

This simple equation requires the additional assumption that risk to weasels is negligible in these interactions.

The second part of the model represents the interactions between SAMs and strike aircraft, and between strike aircraft and targets. Let p_i denote the probability that a strike

aircraft which reaches target t destroys the target. Let r_{kt} denote the probability that if SAM k is still alive, it will destroy a strike aircraft headed for target t . Note the dependence on the target, which represents how strongly a specific SAM can interact with the route headed for target t . Assuming again that interaction events between SAMs and strike aircraft, between strike aircraft and targets, and between weasel aircraft and SAMs are mutually independent, we obtain the following expressions: Let n_t denote the number of strike aircraft allocated to target t . Then, the probability that a strike aircraft headed for target t reaches target t is given by $P(t)$:

$$P(t) = \prod_k (1 - r_{kt} \prod_{t'} (1 - q_{kt'})^{m_{t'}})$$

and the probability that target t survives is given by $P^S(t)$:

$$P^S(t) = (1 - P(t))^{n_t}$$

Given target values V_t the combinatorial optimization problem of interest is:

$$\begin{aligned} \max_{(m_t, n_t)} \quad & J(\{(m_t, n_t)\}) = \sum_t V_t (1 - P^S(t)) \\ \text{subject to} \quad & \sum_t n_t \leq S, \quad \sum_t m_t \leq W \end{aligned}$$

The above objective function exhibits the coupling between packages headed for different targets affected by common SAMs, plus the coupling between weasels and strike aircraft in packages. Consideration of this objective function reveals that if the weasel allocations to each package are fixed, then the objective function becomes a separable objective function over the strike aircraft content, where each separable component is concave. That is, if we fix m_t , the objective function becomes

$$J(\{(m_t, n_t)\}) = \sum_t J_t(n_t)$$

where

$$J_t(n_t) = V_t (1 - (1 - P^S(t))^{n_t})$$

has a concave envelope. This means that finding the optimal strike aircraft allocation is a simple optimization problem if the weasel allocation is known. This problem is solvable optimally using a maximum marginal return algorithm, which assigns strike aircraft incrementally to the target that offers the greatest increase in performance per strike aircraft assigned. This approach

suggests an alternating procedure, which alternates between selecting weasel aircraft assignments and strike aircraft assignments. Unfortunately, fixing the strike aircraft assignment does not decouple the weasel assignment problem, which still remains a combinatorially hard problem.

The MMR algorithm which we developed uses the alternating decomposition, while applying an incremental marginal return algorithm for both the weasel and strike aircraft allocation to packages. It also uses the more detailed objective function Q which arises from our ADP approaches. The algorithm is outlined below:

- Initially, allocate one strike aircraft per mission. That is, let $N_i^{Strike} = 1$ for all i .
- Set $N_i^{Weasel} = 0$ for all i . Determine weasel aircraft allocations per package as follows:
 - For each package i , compute the marginal return $MR(i)$ as follows: Let $Q_i^{+wk}(\{(N_i^{Strike}, N_i^{Weasel})\})$ denote the performance achieved by adding k weasel aircraft to package i , for $k = 1, 2$. Then,

$$MR(i) = \max_k \{Q_i^{+wk}(\{(N_i^{Strike}, N_i^{Weasel})\}) - Q(\{(N_i^{Strike}, N_i^{Weasel})\})/k\}$$
 - Select the package i with largest $MR(i)$, and increase $N_i^{Weasel} = N_i^{Weasel} + 1$
 - Repeat until no unallocated weasel aircraft remain.
- Set $N_i^{Strike} = 0$ for all i . Determine strike aircraft allocations per package as follows:
 - For each package i , compute the marginal return $MR(i)$ as follows: Let $Q_i^{+sk}(\{(N_i^{Strike}, N_i^{Weasel})\})$ denote the performance achieved by adding k strike aircraft to package i , for $k = 1, 2$. Then,

$$MR(i) = \max_k \{Q_i^{+sk}(\{(N_i^{Strike}, N_i^{Weasel})\}) - Q(\{(N_i^{Strike}, N_i^{Weasel})\})/k\}$$
 - Select the package i with largest $MR(i)$, and increase $N_i^{Strike} = N_i^{Strike} + 1$.
 - Repeat until no unallocated strike aircraft remain.
- Repeat iteration between assignment of weasel and strike aircraft until convergence or a fixed number of iterations are performed.

The above algorithm incorporates several important extensions to address the use of the objective function Q instead of the simpler model. In particular, to deal with possible regions where the function is not concave, we use two increments ($k = 1, 2$) to compute the maximum

marginal return. Note that the above algorithm is an approximate algorithm, as the true objective function for a multistage problem will not be separable or concave. It does provide a fast, approximate algorithm, which can be used in combination with other algorithms such as combinatorial rollout, which we consider next.

4.3.2 Combinatorial Rollout

The Combinatorial Rollout algorithm is a recent algorithm developed by Bertsekas et al [BTW97]. The basic idea of the algorithm is to improve on the performance of a baseline algorithm, which in our case is the MMR algorithm described above. These incremental improvements are related to the policy improvement step in standard policy iteration algorithms for dynamic programming.

In combinatorial rollout, we solve the optimization problem, one package at a time, as follows:

- Order the package indices $i = 1, K, T$. Let the current index $i' = 1$.
- Assume that packages $(N_i^{Strike}, N_i^{Weasel})$ are fixed for $i < i'$. Determine the package allocation to target i' as follows:
 - Enumerate all possible package allocations to target i' . For each package allocation $(N_i^{Strike}, N_i^{Weasel})$, allocate remaining strike aircraft and weasel aircraft (not already allocated to packages $i < i'$ or allocated to i') using MMR algorithm to packages $i > i'$, and evaluate the performance of the composite assignments.
 - Select $(N_i^{Strike}, N_i^{Weasel})$ as the allocation which gives the maximum performance in the previous substep.
- If $i' < T$, increment $i' = i' + 1$; else, the algorithm is complete.

In a setting where the performance function is evaluated exactly, combinatorial rollout is guaranteed to perform no worse than the baseline algorithm, provided the baseline algorithm satisfies a mild condition of sequential consistency [BTW97] which our MMR algorithm satisfies. However, when the performance function is evaluated only approximately, as in stochastic settings, this performance improvement is not guaranteed. In particular, the incremental nature of the algorithm makes it difficult to distinguish among package allocations where the difference in performance is of the same order of magnitude as the evaluation error. In

the next section, we describe a combinatorial algorithm, which is explicitly designed for optimization of functions with uncertainty in performance evaluation.

4.3.3 Surrogate Method

The Surrogate Method [GoCass00] is a gradient-based approach for searching the control space (i.e., sets of feasible missions). This approach constructs a continuous "surrogate" objective function that is used to generate gradient information that guides a search through the discrete space of mission allocations. It also uses a stochastic approximation technique, which allows for uncertain gradient information evaluation. The gradient information is obtained by selecting a set of neighbor points to the current allocation, and evaluating the function at these neighbor points.

The principal idea of the surrogate method is to embed the combinatorial optimization problem into a continuous optimization problem. Let $\{(N_i^{Strike}, N_i^{Weasel})\}$ denote the combinatorial decision variables. The surrogate method instead optimizes over variables $\{(x_i^{Strike}, x_i^{Weasel})\}$ where x denotes a continuous allocation. Let Q denote the combinatorial performance index; the key problem is that Q is only defined on feasible nonnegative integer package assignments. The algorithm extends the function Q to a surrogate function $R\{(x_i^{Strike}, x_i^{Weasel})\}$ defined on continuous package assignments as follows:

- Given $\{(x_i^{Strike}, x_i^{Weasel})\}$, find the closest integer assignment $\{(N_i^{Strike}, N_i^{Weasel})\}$, and evaluate the performance $Q(\{(N_i^{Strike}, N_i^{Weasel})\})$.
- Find $2T$ neighbors of $\{(N_i^{Strike}, N_i^{Weasel})\}$ by modifying the number of strike or weasel aircraft to each package one at a time by one aircraft, and evaluate the performance Q for each neighbor.
- Use the $2T + 1$ values to evaluate $R\{(x_i^{Strike}, x_i^{Weasel})\}$ as a linear interpolation of the corner values.

Note that the above approximation has $R\{(x_i^{Strike}, x_i^{Weasel})\} = Q(\{(N_i^{Strike}, N_i^{Weasel})\})$ at nonnegative integer values of the allocations $\{(x_i^{Strike}, x_i^{Weasel})\}$. Since R is now defined over continuous variables, one can compute the gradient, which is piecewise constant over regions where the closest corner and the neighbors are constant. Note, however, that the continuous function is not

differentiable at integer values, because these values are at the intersection of different piecewise linear approximations (i.e. an integer point is a corner for many regions).

The surrogate algorithm is summarized as follows:

- Initialize a feasible guess at the package allocations $\{(N_i^{Strike}, N_i^{Weasel})\}$ across all targets.
- Initialize the step size $a_1 = 1$, and the fractional package allocations $\{(x_i^{Strike}, x_i^{Weasel})\}$ by perturbing the integer assignments $\{(N_i^{Strike}, N_i^{Weasel})\}$. Initialize the iteration index to $k = 1$.
- Perform a gradient iteration as follows:
 - Compute the $2T + 1$ neighbors of the fractional package allocations $\{(x_i^{Strike}, x_i^{Weasel})\}$, evaluate the Q function at the neighbors, and evaluate the gradient $g\{(x_i^{Strike}, x_i^{Weasel})\}$.
 - Modify the fractional allocation as

$$\{(x_i^{Strike}, x_i^{Weasel})\} = \{(N_i^{Strike}, N_i^{Weasel})\} + a_k \cdot g\{(x_i^{Strike}, x_i^{Weasel})\}$$
 - If the new allocation $\{(x_i^{Strike}, x_i^{Weasel})\}$ is infeasible, project it inside the feasible region by reducing each allocation by the same proportionality constant.
- Increase the iteration index $k = k + 1$, reduce the step size as $a_k = 1/k$.
- Compute nearest feasible integer allocation $\{(N_i^{Strike}, N_i^{Weasel})\}$ and evaluate its performance.
- Repeat iterations for specified number of iterations.

Because of the piecewise linear nature of the approximation, the optimal fractional solution is at an integer value; thus, if the optimization finds the optimal allocation for the surrogate cost function $R\{(x_i^{Strike}, x_i^{Weasel})\}$, it also finds the optimal allocation for the original objective function $Q\{(N_i^{Strike}, N_i^{Weasel})\}$ [GoCass00]. Furthermore, the slowly-decreasing step size guarantees convergence to a local optimum even if the evaluations of the function Q are noisy. However, as a gradient descent algorithm, the surrogate optimization method often converges to a local instead of a global optimum. To overcome this limitation, we implement a couple of steps: First, we initialize the algorithm with the MMR solution described previously. Second, we also perform several repetitions of the algorithm from randomly selected assignments, and select the best of the resulting allocations.

4.4 DESIGN MODEL APPROACH

In this section, we develop a design model for the system dynamics, which are describe in substantial detail in Section 3.2. Similar to the hybrid dynamical model, our design model exploits the fact that interactions between JAO objects are sparse and involve relatively few objects in each event, in order to achieve compact and efficient evaluation methods. In the following, we outline our implementation for the JAO environment.

The design model is based on a small set of dynamical models and their associated interaction dynamics. These models correspond to physical objects, such as air packages, threats, and targets. This direct mapping simplifies construction of the design model. Based on the physical objects, the design models may be distilled from the more detailed hybrid dynamical model, as follows.

Parallel composition of individual object models provides a concise representation of the JAO dynamics in the absence of interactions. When objects interact, e.g., threat and target engagements, a product composition of the associated objects concisely represents the interaction. To represent these interactions, a set of composite models representing pairs of interacting objects was developed. These models, based on individual exchanges in an underlying Markov process, capture the complex interaction dynamics, e.g., weasel suppression, target acquisition, etc. The first of these models is a transition model between an air package i and an air defense SAM j . Let π_i denote the discrete state associated with the air package, consisting of the number of aircraft of each type which remain alive in the package, and let s_j denote the state of the SAM. In our SAM models, the SAM can be in one of five states, as described in our hybrid dynamics. Given the hybrid state trajectory of the air package and the capabilities of the SAM, we compute the transition kernel $P(\pi_i(+), s_j(+) | \pi_i(-), s_j(-))$, which is a matrix indexed by possible package contents into and out of the engagement, and SAM status into and out of the engagement.

The second model is a transition model between an air package i attacking target k . Targets can be in one of two states, alive or dead. As before, these dynamics are lifted from the detailed hybrid model, and represented as a transition kernel between the joint states, as $P(\pi_i(+), t_k(+) | \pi_i(-), t_k(-))$. Other models represent independent transition dynamics in SAM states, as SAMs turn on and off, or are repaired after incurring damage. By factoring the joint

probabilities at the end of each stage into products of marginal probabilities for each object, one obtains an efficient prediction of the distribution at the end of a wave of activity, which can be used to compute the performance statistics associated with any given strategy.

The above prediction approach is based on propagating through a pre-specified sequence of interaction events. An important extension that we considered in this work is closed-loop prediction, where the particular sequence of interaction events depends on the specific states that are observed as outcomes. For instance, after a specific interaction between an air package and a SAM, the air package may abort its mission if the number of surviving weasel aircraft and strike aircraft falls below required quantities. Similarly, the missions selected in the second wave of an attack depend on the relative success of the first wave missions in eliminating targets, and the number of aircraft surviving the first wave.

We focus on the problem of two-stage prediction, where information is collected at the end of a stage or wave, and the next set of missions is then adapted to the results of the first wave. Closed-loop prediction depends on the arrival of information. We assume that the state at the end of the first wave is observed, and that the strategy for the second wave is then computed. Our evaluation approach is based on computing an analytical approximation to the distribution of this state as before, sampling this distribution to generate a finite number of representative scenarios. For each sample scenario, we use a combinatorial algorithm using a single wave analytical approximation of the performance criteria to determine both the desired sequence of missions and their expected performance. The performance achieved for these samples is then averaged to obtain estimates of the two-wave performance.

Another extension that we considered was a model of partial information arrival, where ISR sensors were scheduled over the battle space. In this case, the observations are perfect, but occur over time, and are localized around the ISR sensors. The localized observations result in partial information regarding the battlespace. These observations are projected forward to the current time using the same probabilistic transition models discussed previously. The result is a probabilistic state estimated at the decision point from which decisions must be made.

4.5 COMBINATORIAL OPTIMIZATION ALGORITHMS IMPLEMENTED

This section describes how the plant implements the different controller algorithms described in Section 4.3, including when and how they are used. Further details about each algorithm are also given. As it will be evident in the sections below, the controllers may be split up into two distinct groups depending on their function. One type is used to allocate resources (strike and weasel aircraft) at the airbase to newly constructed air package objects, which the controller outputs. In addition to configuring the air packages, the controller also sets their initial missions. The second type of controller is used to modify previously created air packages' missions while in flight. No controller has the ability to reconfigure air packages while in flight, such as moving assets between packages.

4.5.1 Retasker using Combinatorial Rollout

This controller may be called by the plant when a given TCT emerges. It has a very specific role of finding the air packages that qualify for retasking to the TCT, determining the best one (if any) to divert to it, and modifying its mission accordingly. In order for an air package to qualify for retasking, it must meet the following two criteria: 1) it must be currently flying ingress to a normal target, and 2) its ingress mission route must intersect a circle of a given radius around the TCT, as illustrated in Figure 18 below. The radius, or retask range, used in all experiments was 50 km. If an air package is tasked to an AOR, it automatically qualifies for retasking if the TCT is within the same AOR. The reason for implementing a localized retasking, via the intersection range and AOR groupings, is to avoid drastic geographic changes to an air package's mission, which would have a greater disturbance on the highly coupled missions of the other aircraft. Only allowing the packages that were already passing near the TCT to divert to it should minimize the effect on coupled activities of the previously configured packages, such as in the coordinated suppression of enemy air defense. This is very important since only one air package is diverted to the TCT, and the other packages are not retasked to account for the change in the mission queue. The Retasker was implemented in this specific, localized manner in order to provide real-time control upon TCT emergence, and also to avoid the potential snowball effect that allowing other air packages to divert to newly unassigned targets (resulting from previous retaskings) could have.

When using the Retasker, the plant will request guidance immediately following any TCT emerge event (it is part of the event's execution logic), passing the controller the name of the TCT, in addition to the required estimated state of the world. If an air package is already assigned to the TCT, then the retask call is unnecessary and will terminate. Otherwise, the Retasker first determines which air packages are valid candidates as explained above. It then uses the combinatorial rollout algorithm discussed in Section 4.1 to find the best retask

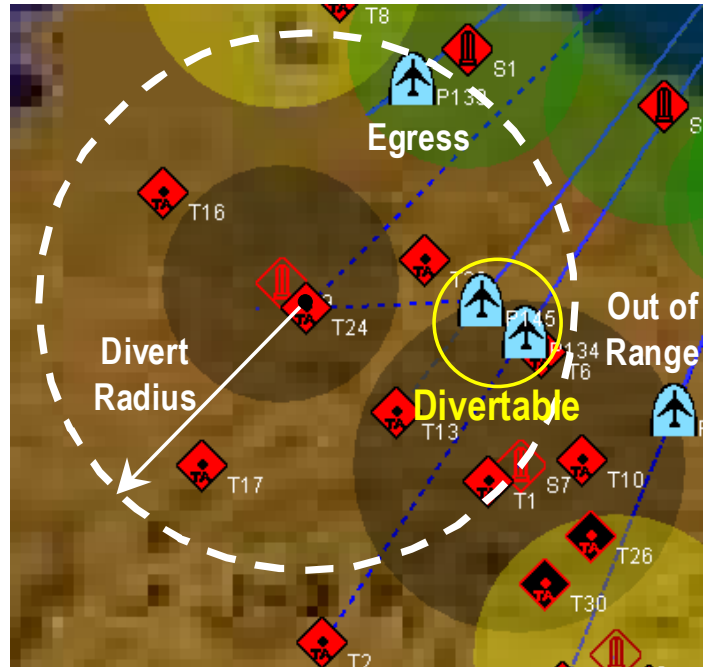


Figure 18 TCT Retasking Radius

option. This is done by changing one of the valid air package's missions at a time, and evaluating the effect on the entire mission queue using the one-wave, one-stage predictor (see Section 4.6.1). The option that gives the best performance expectation is selected, and if its predicted value improves that of the original mission queue, the corresponding air package is retasked by modifying its mission and returning it to the plant as guidance.

4.5.2 Aborter using Combinatorial Rollout

This specialized controller is used in a variety of circumstances to decide whether one or more ingress air packages should abort their current missions and return to the airbase. The ability to abort missions is useful to avoid attrition to air packages, especially in an uncertain hostile environment. Depending on the current state of the world, an air package may be aborted whenever the expected gain of prosecuting its assigned target (and value of supporting other aircraft) is outweighed by the potential for further attrition. This allows the ability to save resources that might otherwise be ineffective and/or destroyed.

The Aborter functions differently depending on how the plant uses it. One way the plant may employ this controller is by calling it automatically at periodic time intervals. In this case, the Aborter performs a “full abort,” giving each air package the opportunity to abort its mission, by using the combinatorial rollout algorithm discussed in Section 4.1. This works by aborting

each ingress air package one at a time, evaluating the effect on the entire mission queue using the one-wave, one-stage predictor (see Section 4.6.1). The option that gives the best performance expectation is selected, and if its predicted value improves that of the current mission queue, the corresponding air package is aborted by modifying its mission to return to the airbase immediately. This process continues using the remaining air packages, evaluating each option in a mission queue that includes any previously aborted packages, until the mission queue cannot be improved by aborting any air packages. All aborted packages are then returned to the plant as guidance.

Another way the plant uses the Aborter is when a SAMSite emerges from an unknown (hiding), inactive, or repairing state. If a SAMSite activates in response to the intersection of an air package with its range of lethality (i.e. missile range) in order to engage the package, the Aborter is called after the engagement and any possible post-engagement SAMSite transitions occur. This is useful for responding to an uncertain engagement, which may have resulted in a loss of aircraft that could compromise the success of both the attrited air package's mission and any other missions with which it is coupled. In this case, the Aborter first decides whether or not the just-engaged air package should abort its mission by comparing the respective performance expectations using the one-wave, one-stage predictor. If so, a full abort is performed (as explained above) to account for the potential effect on any other air packages. The initial fixation on the air package involved in the engagement is done to speed up the computation time, and also as an attempt to localize the abort.

If the Aborter is called in response to a SAMSite emergence that occurred stochastically, not due to a specific simulation event, the Aborter functions slightly differently. It first decides if any ingress air packages currently routed through the emerged SAMSite's range of lethality should be aborted. This is done very similarly as a full abort, except the abort candidates are limited to this subset of air packages. If any of these packages were retasked to the airbase, a full abort is then performed on the remaining air packages to account for the possible coupling across missions. Similarly as above, the first step is performed to both minimize computation time and to localize the Aborter's effect to just those air packages involved with the specific SAMSite.

4.5.3 Target Tasking using Maximum Marginal Return

The MMR controller is used to construct and configure new air packages from resources at the airbase, and task them to targets. First, the controller must create new air packages, one

for each unassigned target in the estimated world state provided by the plant. Each package is initialized with zero weasel and one strike aircraft. As detailed in Section 4.3.1, the next step is to allocate weasel aircraft to the set of packages. This is done by temporarily incrementing the number of weasels in each package one at a time, evaluating each option within the current mission queue configuration using any of the possible predictors (see Section 4.6), and permanently adding the weasel aircraft to the package with the best performance expectation. This process continues until either the airbase runs out of weasel aircraft to allocate, or adding weasels to any package does not improve the mission queue's predicted value. The algorithm may be tuned by changing the maximum number of weasel aircraft allowed in an air package, changing the increment used when allocating weasels (how many to allocate at a time), or even by stepping multiple increments ahead when searching the control space for the best predicted value. To clarify the latter option, take the example of allocating two weasels at a time. If we allow three steps into the control space, then the number of mission queues to evaluate would equal the number of air packages times three, each being incremented by two, four, or six weasel aircraft. The one option that gives the best performance expectation would have its corresponding air package's weasel count incremented by two weasels, regardless of how many were allocated when testing that option.

Next the strike aircraft are allocated using a similar process. Initially, the previous weasel assignment is untouched, but the strike aircraft in each package are cleared to zero. The strike aircraft at the base are then allocated incrementally just as the weasels were above. The only difference is in how the control space may be "stepped into." If strike aircraft were used in the previous weasel allocation example, options with four or six aircraft would only be evaluated if no options with two or four aircraft, respectively, improved the mission queue's expected performance. These tunings of the weasel and strike assignment were used to overcome specific problems where the MMR would exit prematurely before allocating many of the aircraft, depending on how the algorithm was being used. The weasel and strike allocation cycles may be repeated as desired, although one or two iterations of the algorithm is usually sufficient. By clearing the particular type of aircraft being assigned from the current mission queue, it may be possible to improve that aircraft's allocation using the current allocation of the other types of aircraft.

Like any other controller, the plant may employ the MMR whenever it deems it useful. In the MMR's case, this would be any time air packages may be formed and tasked to targets, whether the resources are at an airbase or an AOR. Commonly, the MMR would be used at the start of the simulation and at the end of each consequent wave of attack, that is, when all air packages return to base after executing their missions. Alternately, the MMR could be called when each air package returns to base, a "gorilla" air package arrives at its AOR waypoint, or even at periodic time intervals.

4.5.4 AOR Tasking using Maximum Marginal Return

This controller is a variation of the MMR algorithm used to task air packages to targets. Its main function is still to allocate weasel and strike aircraft to air packages. But instead of tasking them to targets, their missions are to AOR waypoints, at which a target tasking controller is used. There are two ways of using this controller, depending on how many stages the user wants to model. The traditional, more accurate way of using the controller is to form a gorilla package for each AOR, and allocate resources just like in the target tasking MMR (see Section 4.5.3), but using a two-stage predictor to evaluate mission queues of gorilla air packages (see Section 4.6.5). The first stage represents the gorilla packages flying to their AOR waypoints, and the second stage is the tasking of smaller air packages from the AORs to targets and back to the airbase. The other way of using this controller is in a one-stage context. This case works just like the target tasking MMR, actually configuring air packages tasked to unassigned targets. The only difference is that they are routed through the AOR waypoint instead of directly to the target. After the aircraft allocation is complete, the aircraft in all air packages tasked to targets in each AOR are conglomerated into larger gorilla air packages tasked to the respective AORs. The only reason to use this one-stage method in lieu of using the better two-stage predictor is to save computation time. This controller is implemented by the plant at the same times as the target tasking MMR, except it can only be used to task aircraft at the airbase, not when they arrive at an AOR waypoint.

4.5.5 Target Tasking using Surrogate Method

The surrogate method is another controller used to construct and configure new air packages from resources at the airbase, and task them to targets. It is implemented by the plant in the same way as the MMR (see the end of Section 4.5.3). First, the controller must create unconfigured air packages, one for each unassigned target in the estimated world state provided

by the plant. A gradient-based approach then searches the control space to allocate strike and weasel aircraft to the different packages, as detailed in Section 4.3.3. One addition to the algorithm's description above is an option that was added to speed it up. Rather than restarting the algorithm multiple times with random air package configurations (or target assignments), it was useful to seed the surrogate with the MMR's solution, and then iterate over that to try to improve upon it by moving aircraft between the packages.

4.5.6 AOR Tasking using Surrogate Method

This controller is a variation of the surrogate method used to task air packages to targets (see Section 4.3.3). Its main function is still to allocate weasel and strike aircraft to air packages. But instead of tasking them to targets, their missions are to AOR waypoints, at which a target tasking controller is used. After forming one "gorilla" package for each AOR, resources are allocated using the same gradient-based approach detailed in Section 4.3.3, but using a two-stage predictor to evaluate mission queues of gorilla air packages (see Section 4.6.5). The first stage represents the gorilla packages flying to their AOR waypoints, and the second stage is the tasking of smaller air packages from the AORs to targets and back to the airbase. This controller is implemented by the plant at the same times as the target tasking surrogate method, except it can only be used to task aircraft at the airbase, not when they arrive at an AOR waypoint.

4.5.7 Target Tasking using Combinatorial Rollout

The combinatorial rollout controller is used to construct and configure new air packages from resources at the airbase, and task them to targets, based on the algorithm in Section 4.3.2. First, the controller must create new air packages, one for each unassigned target in the estimated world state provided by the plant. The control space is then directly enumerated with every possible air package configuration to each target, taking into consideration any constraints on maximum numbers of aircraft per package and incremental assignments of aircraft (i.e. two weasel aircraft at a time). Each option is selected and added to a mission queue to evaluate. Any resources remaining at the airbase (not in the mission queue) are used to form air packages to task to unassigned targets, using a greedy heuristic. This mission queue is then evaluated using any of the possible predictors (see Section 4.6). The air package option whose heuristically completed mission queue gives the best performance expectation is then added to the final

mission queue. Any other options in the enumeration tasked to that package's target are removed from the possible candidates. In successive iterations of this option selection, the mission queue evaluated includes not only the new candidate, but also any previously selected packages, which decreases the resources available to the heuristic when filling out the rest of the mission queue. This process continues until either all resources have been exhausted, or the current mission queue's predicted value cannot be improved by adding any option to it.

This controller is implemented by the plant in the same way as the other target tasking controllers (see the end of Section 4.5.3), except it can only be used to task aircraft at the airbase, not when they arrive at an AOR waypoint.

4.6 DESIGN MODELS IMPLEMENTED

The control architecture described Section 4.2.2 executes a measured response, i.e., trading off computation and performance to specific “trigger” events. In order to achieve “closed-loop” behaviors, control decisions must explicitly account for subsequent decisions. In this section, we highlight several multi-stage controller designs, typically for the allocation of base resources, which account for subsequent decisions. In each case, what distinguishes these controllers is the associated prediction model.

4.6.1 1-Stage/1-Wave Model

The 1-Stage/1-Wave controller determines a set of missions over a single wave. Each wave begins when the air packages are launched from base and ends when they return.

This controller solves the combinatorial optimization problem discussed above (Section 4.3), and therefore any of the methods may be used. Each of these methods evaluates candidate control option using our design model (Section 4.4) over a single wave. This is depicted in Figure 19.

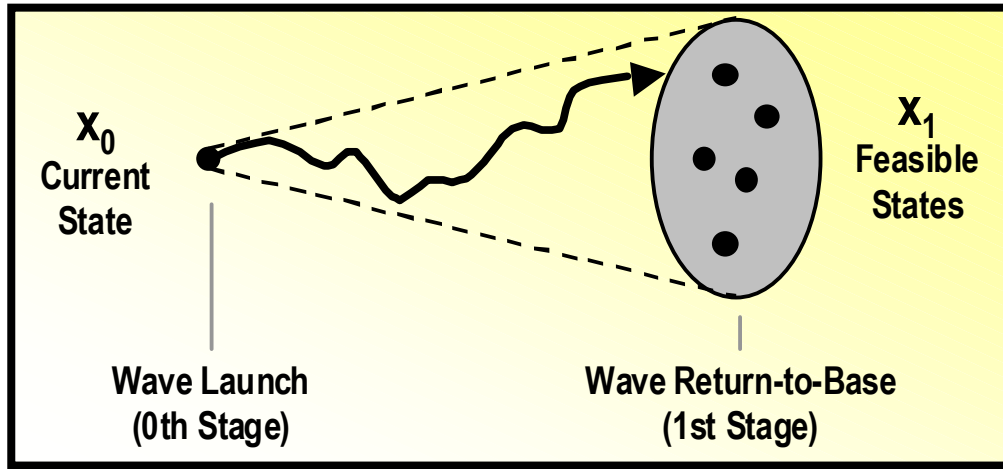


Figure 19 1-Stage Prediction

This is the baseline controller, which is also the basis for subsequent controller implementations. Due to relative performance with respect to computation, the MMR algorithm is better suited for the 1 Stage/1 Wave case.

4.6.2 2-Stage/1-Wave Model with Retasking

This controller also considers a 1-wave horizon, but accounts for potential retasking of air packages in response to emerging TCTs. TCT emergence is a random event during the execution of a wave. When a TCT emerges, divertable air packages (i.e., loaded and within range) are considered for retasking, and the best option is selected (see Section 4.5.1). Therefore, we expect the controller to assign missions from base that anticipate potential retasking. This type of proactive control significantly increases computational complexity. The difficulty is in predicting the value over 2 stages, i.e., through an intermediate decision point that corresponds to a retasking decision. We illustrate this in Figure 20. A set of missions is launched from base. When a TCT emerges, a control decision is required, which in general will map the current state x_1 at the time of the TCT emergence to an appropriate retasking control u_1 . Given a specific state x_1 and control u_1 , we are able to complete the 1-wave prediction. This is difficult even for a single TCT since the emergence of the TCT occurs at a random point in time and the control decision is state dependent.

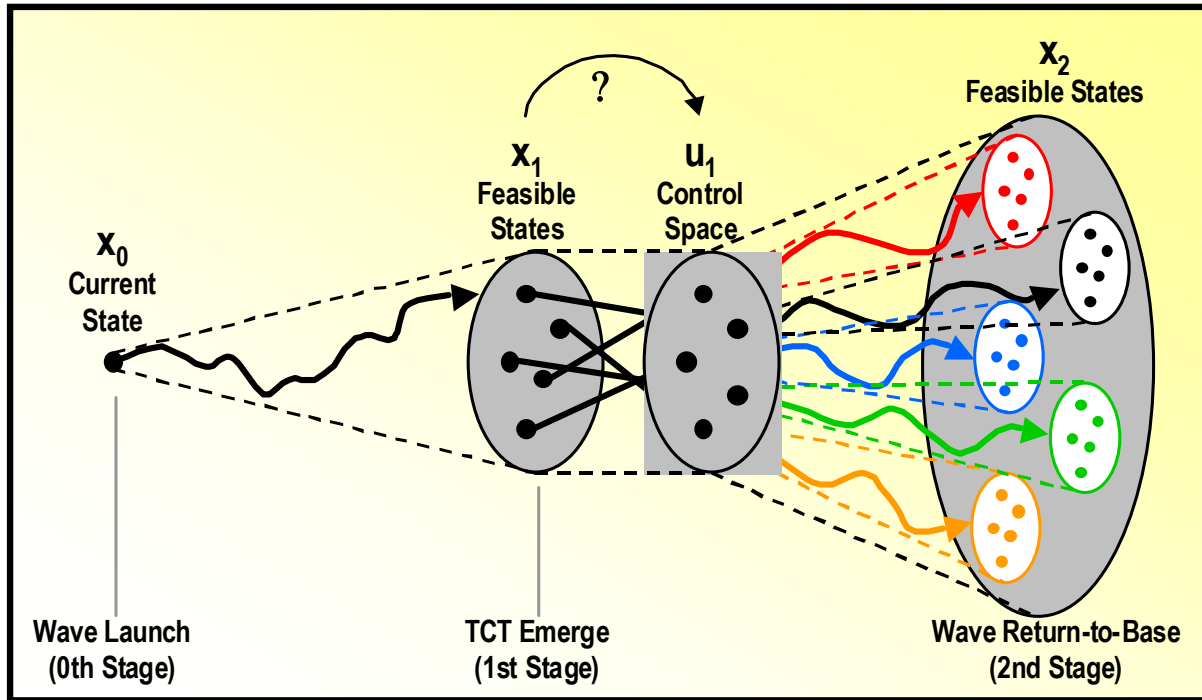


Figure 20 2-Stage Retasking Prediction

To simplify this problem, we assume that retasking depends only on the arrival probability of the TCT (rather than the precise state of all air packages, targets, and threats) and that the retask decisions are synchronized across missions. This is illustrated in Figure 21, where the first air package to enter a TCT divert range triggers a divert decision for all air packages.

With these assumptions, we are able to use the same combinatorial algorithms with a specialized predictor. Consider Figure 20, our baseline predictor is used up until the retasking point (TCT Emerge in the figure). At this point the retasking controller is used to identify a candidate retask option, which is implemented probabilistically depending on the probability that the TCT has emerged. Therefore, for a given retask option, the predicted value is given by a weighted average of either retasking or not retasking the individual air packages. Note, that the retask control algorithm needs to be solved for each evaluation of this 2-stage controller.

This controller generates an open loop control decision that hedges against the probabilistic arrival of TCTs, effectively inflating the value of missions which could be diverted while encouraging an allocation of resources appropriate for prosecution of potential TCT as well as original targets.

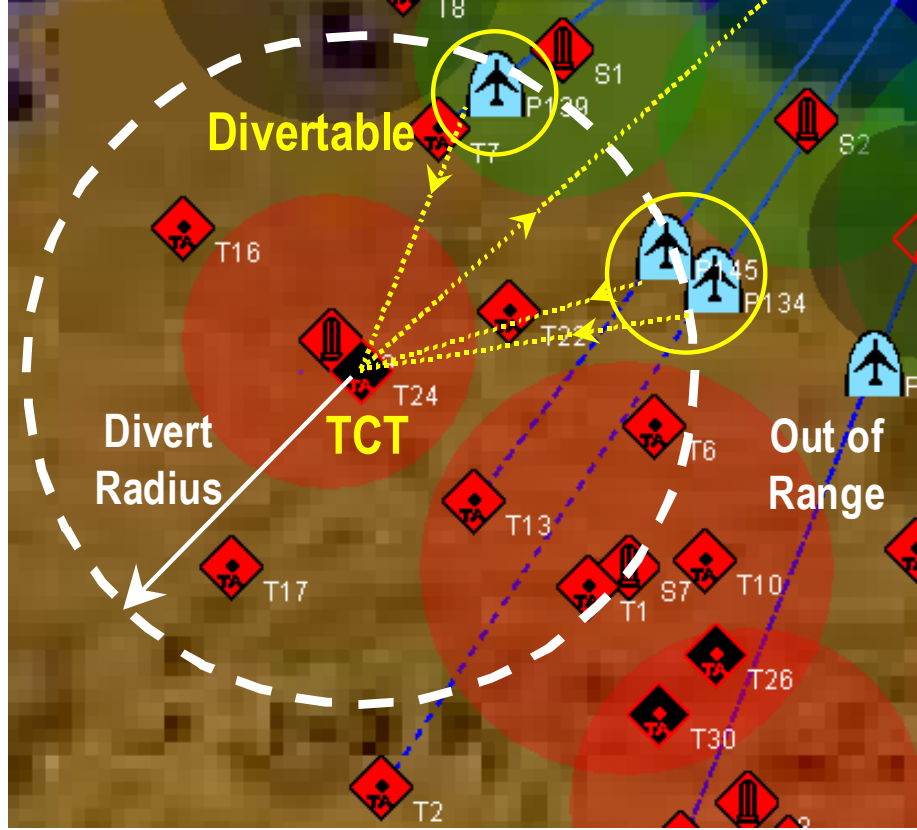


Figure 21 Retasking Approximations

4.6.3 2-Stage/2-Wave Model

In this case, we consider two waves without retasking. The controller determines a set of mission for the first wave, while explicitly accounting for second-wave control decisions. Selection of the first-wave control uses the same combinatorial algorithms as in the previous cases, however a prediction is required that considers two waves. This is depicted in Figure 22.

The set of missions is launched from base. When the air packages return to base, a control decision is required, which in general will map the current state x_1 at the beginning of the second wave to an appropriate second wave control (i.e., set of missions) u_1 . Given a specific state x_1 and control u_1 , we are able to compute the 2-wave prediction as a function of the x_1 .

We use our baseline predictor for the first wave, which generates a probabilistic distribution $f(x_1|x_0, u_0)$ over the state x_1 . The control $u_1 = \mu(x_1)$ is determined from a given state using one of the same combinatorial algorithms that were available for the first stage. To evaluate the second stage, we compute the expected value $J[f(x_2|x_0, u_0)]$ at the end of the

second stage by averaging over the state x_1 at the end of the first stage. To compute this expectation, we enumerate the feasible states x_1 at the end of the first stage, determine the second wave control $u_1 = \mu(x_1)$, and compute the value $J[f(x_2|x_1, u_1, x_0, u_0)]$ for the given state x_1 . The sum of these values, weighted by the probability of the associated state x_1 , provides the second stage value $J[f(x_2|x_0, u_0)]$. However, this approach is only feasible when the state space of x_1 is small. In the following, we discuss several approximations of the second stage evaluation. Specifically, we consider random sampling, certainty equivalence approximations, and an open-loop approximation.

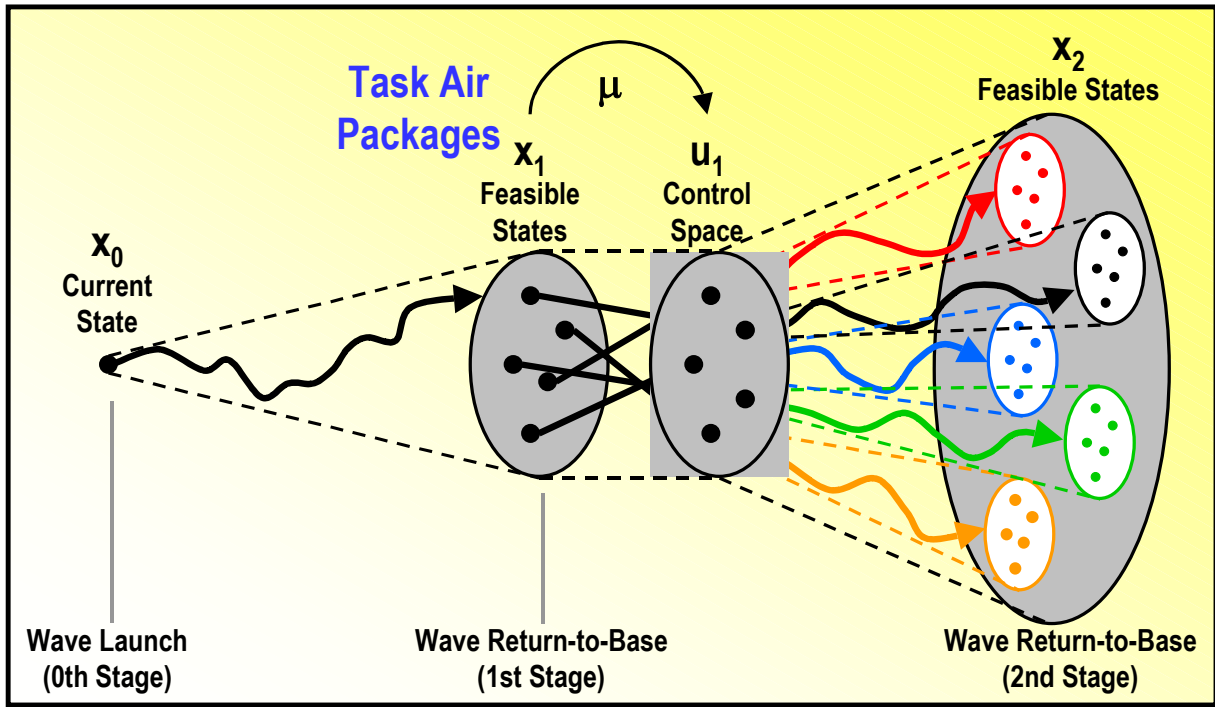


Figure 22 2-Stage/2-Wave Predictor

4.6.3.1 Random Sampling

In Figure 23, we illustrate the random sampling approach used to predict the performance of the second wave. Our design model is used to predict the value of the first wave, and it also provides the (approximate) distribution $f(x_1|x_0, u_0)$ over the state x_1 at the end of the first wave. In this approach, Monte Carlo integration is used to estimate the second stage performance $J[f(x_2|x_0, u_0)]$. A state x_1 is selected randomly according to the known distribution $f(x_1|x_0, u_0)$ at the end of the first wave. The control $u_1 = \mu_{MMR}(x_1)$ is determined based on that state x_1 .

Given the state x_1 and the control u_1 , our design model is used to compute the value of the second stage $J[f(x_2|x_1, u_1, x_0, u_0)]$ given the intermediate state x_1 , from which we estimate the second stage performance

$$\hat{J}[f_2(x_2 | x_0, u_0)] = \frac{1}{N_{samples}} \sum_{i=1}^{N_{samples}} J[f(x_2|x_i, u_i, x_0, u_0)]$$

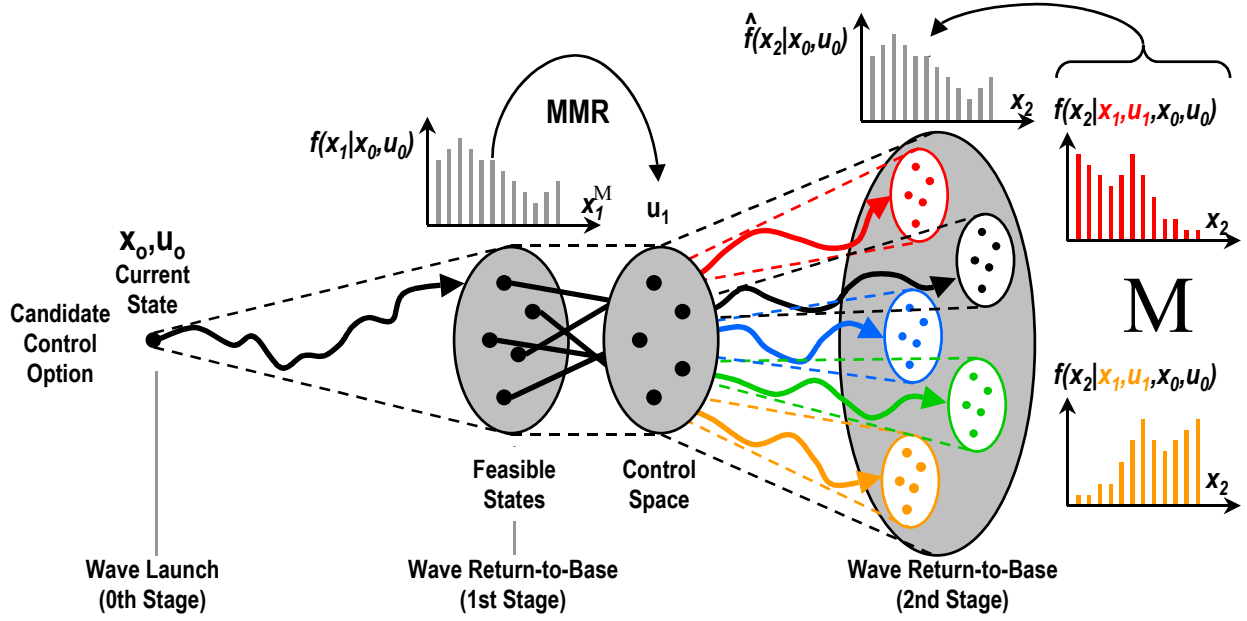


Figure 23 Random Sampling for 2-Stage/2-Wave Prediction

Using the MMR algorithm, this approach requires $O(N_{samples}(N_{TL} + 1)^2(N_s + N_w)^2)$ single stage evaluations using our design model. $N_{samples}$ is the number of Monte Carlo samples. N_{TL} is the number of target locations. N_s is the total number of strike aircraft. N_w is the total number of weasel aircraft. Methods such as importance sampling could theoretically improve performance. However, preliminary experiments demonstrated only a minimal improvement in computation with comparable performance.

4.6.3.2 Certainty Equivalent Approximation

In Figure 24, we illustrate a certainty equivalent approach used to predict the performance of the second wave. Our design model is used to predict the value of the first wave and the associated (approximate) distribution $f(x_1|x_0, u_0)$ over the state x_1 at the end of the first wave. In this approach, a certainty equivalent state $\bar{x}_1$ is selected to represent the entire

distribution. In our experiments, we consider the mean and mode of the distribution. Given the certainty equivalent state $\bar{x}_1$, a control $\bar{u}_1 = \mu_{MMR}(x_1)$ is determined using the MMR algorithm. Given the state $\bar{x}_1$ and the control $\bar{u}_1$, our design model is used to compute the value of the second stage, which is our estimate of the overall second stage performance

$$\hat{J}[f(x_2 | x_0, u_0)] = J[f(x_2 | \bar{x}_1, \bar{u}_1, x_0, u_0)]$$

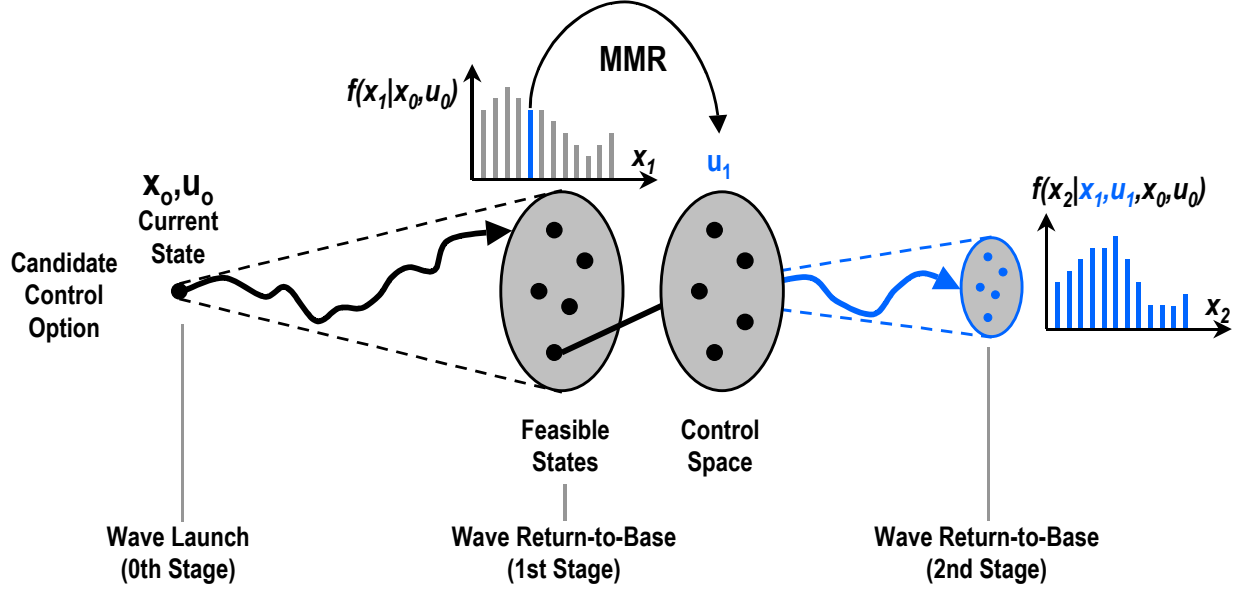


Figure 24 Certainty Equivalent Approximation for 2-Stage/2-Wave Prediction

Using the MMR algorithm, this approach requires $O((N_{TL} + 1)^2 (N_s + N_w)^2)$ single stage evaluations using our design model. Other certainty equivalent states could also be used in this context.

4.6.3.3 Aggregate Open-Loop Approximation

In Figure 25, we illustrate an aggregate open-loop approximation that is used to predict performance in the second wave. In this case, we augment the single stage controller by doubling the number of strike aircraft N_s that are available. The additional aircraft represent the reuse of aircraft in the second stage. Given the increased resources, we determine the initial control u_0 . Due to the additional resources, this control is infeasible, i.e., there are not enough strike aircraft at base to perform all the missions. Therefore, we prune the control decision to make it feasible. Two methods are used. First, we truncate the number of missions (retaining the highest value missions) and reassign weasel aircraft. An alternate approach sends all the missions at half the

strike aircraft. Our design model is used to compute the value of the first wave $\hat{J}[f^{2N_s}(x_1^{2N_s} | x_0^{2N_s}, u_0^{2N_s})]$ with the additional aircraft. The additional aircraft provide a representation of the two wave performance.

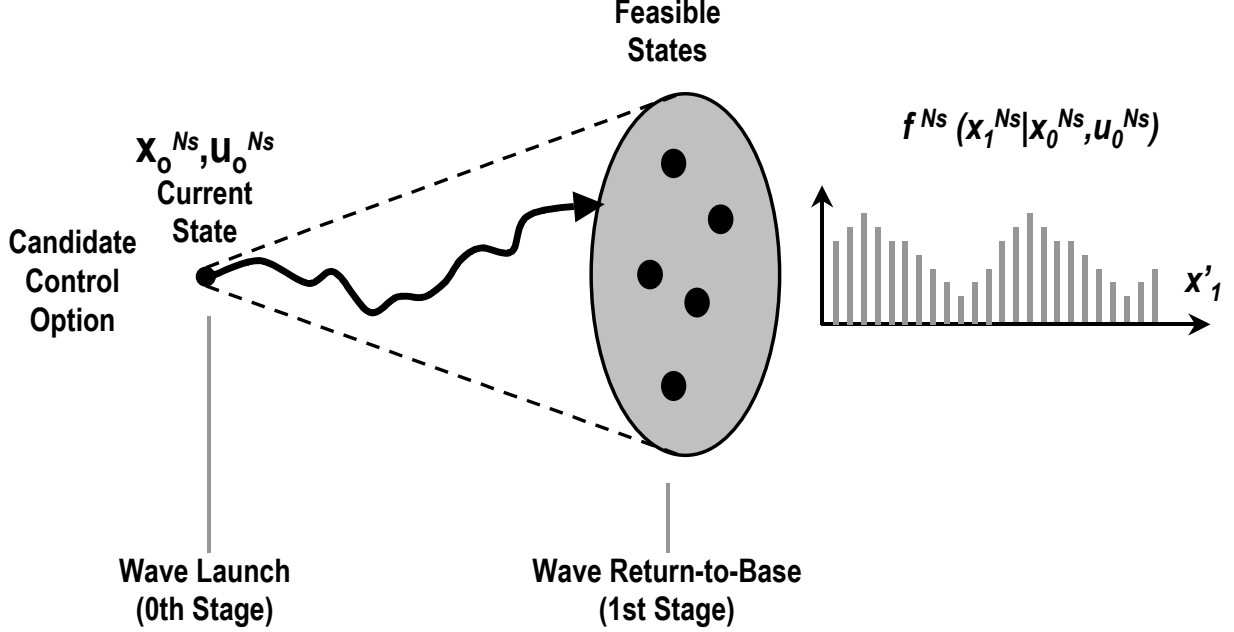


Figure 25 Aggregate Open-loop Approximation for 2-Stage/2-Wave Prediction

Using the MMR algorithm, this approach requires $O((N_{TL} + 1)(2N_s + N_w))$ single stage evaluations using our design model. Similar open-loop approximation could also be used in this context.

4.6.4 4-Stage/2-Wave Model with Retasking

This controller combines the two previous controllers to further extend the control horizon. We consider two waves with retasking. The controller determines a set of missions for the first wave, while explicitly accounting for retasking in the first wave, a second-wave control decision and retasking in the second wave. Selection of the first-wave control uses the same combinatorial algorithms, however a prediction is required that considers two waves with retasking. Two of the four stages are depicted in Figure 26.

We use our retasking predictor (Figure 20) for the first wave, which results in a probabilistic distribution over the state x_2 . The control $u_2 = \mu(x_2)$ is determined from a given state using the same combinatorial algorithms that were used for a single stage. To evaluate the

second stage, we average performance, again using the retasking predictor, over the state x_2 . This is implemented using one of the approximation techniques discussed in the previous section, i.e., random sampling, certainty equivalence approximations, and an open-loop approximation.

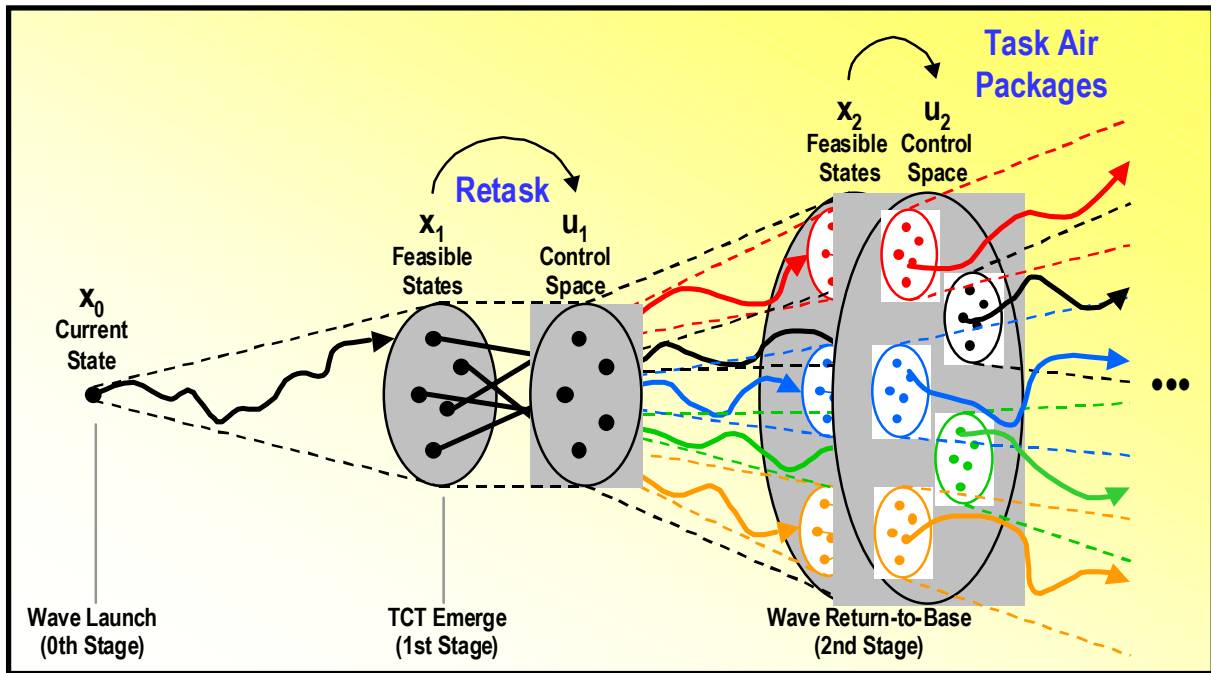


Figure 26 4-Stage/2-Wave Prediction

4.6.5 2-Stage/1-Wave Model with AOR Tasking

Given a hierarchical decomposition of the JAO environment, we formulate an alternate 2-stage problem, which first allocates "gorrilla" air packages to specific Areas of Responsibility (AORs), and then upon arrival to the AOR, tasks regular air packages to specific targets. This effectively decomposes the 1-wave problem into smaller sub-problems associated with individual AORs. This begins to address scalability; sophisticated algorithms may be scaled to realistically sized problems or efficient algorithms may be scaled to larger problems. In this case, the base controller determines a set of missions to AOR points, while explicitly accounting for detailed tasking that occurs at the AOR points. We expect that missions from base will be constructed to provide sufficient aircraft at each AOR point to perform local tasking. This problem is complicated by the potential coupling associated with air defenses, which may occur on ingress, egress, and among AORs. We explicitly account for the ingress coupling, but to

simplify this problem we neglect egress coupling (i.e., all air packages assume full responsibility for air defenses encountered during egress) and the coupling among AORs (similarly, each AOR assumes full responsibility for air defenses in the region).

The set of missions is launched from base to specific AORs. When the air packages arrive at the AOR points, a control decision is required, which in general will map the current state x_1 to an appropriate control (i.e., set of missions within an AOR) $u_1 = \mu(x_1)$. At each AOR point, one of the combinatorial algorithms is used with a single stage predictor modified to account for fact that aircraft do not begin at base and that only a subset of targets are available. Given a specific state x_1 and control u_1 , we are able to complete the 2-stage AOR prediction.

Neglecting coupling among AORs and that due to threats during egress, the base controller uses one of the combinatorial algorithms to assign aircraft to AOR point. The predictor is modified to account for the subsequent tasking of aircraft to targets when they arrive at an AOR. This is illustrated in Figure 27. Our baseline predictor is used to determine the value (lost) during ingress to the AOR points, as well as the probabilistic distribution $f(x_1|x_0, u_0)$ over the state x_1 upon arrival at the AOR points. For a given state, we are able to determine the control $u_1 = \mu_{MMR}(x_1)$, which assigns aircraft to specific targets at the AOR point. To evaluate the second stage in each AOR, we average the performance over the state x_1 similar to the 2-wave/2-stage case. The performance from each AOR is then combined to form the overall performance of the second stage. In the following, we discuss several approximations used to evaluate the second stage in each AOR. Specifically, we consider random sampling and certainty equivalence approximations as before, as well as a partial open-loop approximation.

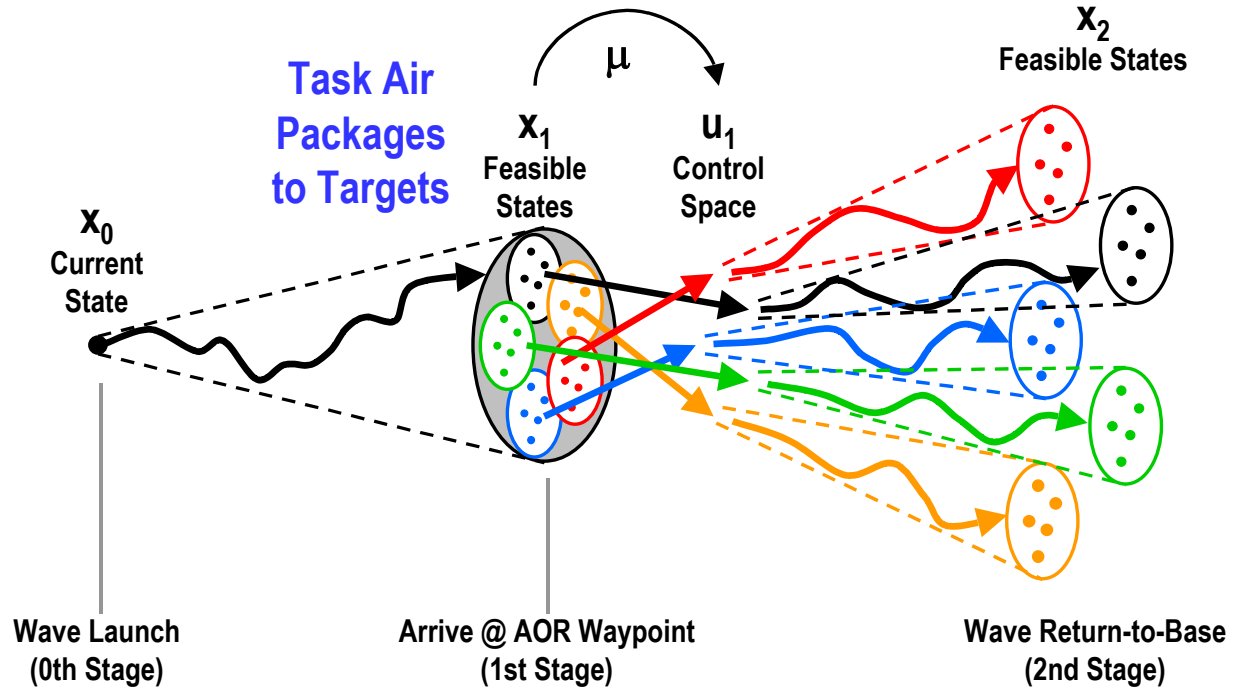


Figure 27 2-Stage AOR Predictor

4.6.5.1 Random Sampling

In Figure 28, we illustrate the random sampling approach used to predict the performance within each AOR. Our design model is used to predict the value igress to the AOR and provides the (approximate) distribution $f(x_1|x_0, u_0)$ over the state x_1 . At this point, we assume that each AOR is independent. For each AOR, Monte Carlo integration is used to estimate the performance $J[f(x_2|x_0, u_0)]$. A state x_1 is selected randomly according to the known distribution $f(x_1|x_0, u_0)$ in each AOR. The control $u_1 = \mu_{MMR}(x_1)$ is determined based on that state x_1 . Given the state x_1 and the control u_1 , our design model is used to compute the value in each AOR, $J_i[f(x_2|x_1, u_1, x_0, u_0)]$ given the intermediate state x_1 , from which we estimate the second stage performance in AOR i .

$$\hat{J}_i[f(x_2 | x_0, u_0)] = \frac{1}{N_{samples}} \sum_{i=1}^{N_{sample}} J_i[f(x_2|x_1, u_1, x_0, u_0)]$$

The overall performance of the second stage is determined by combining the value of the individual AORs.

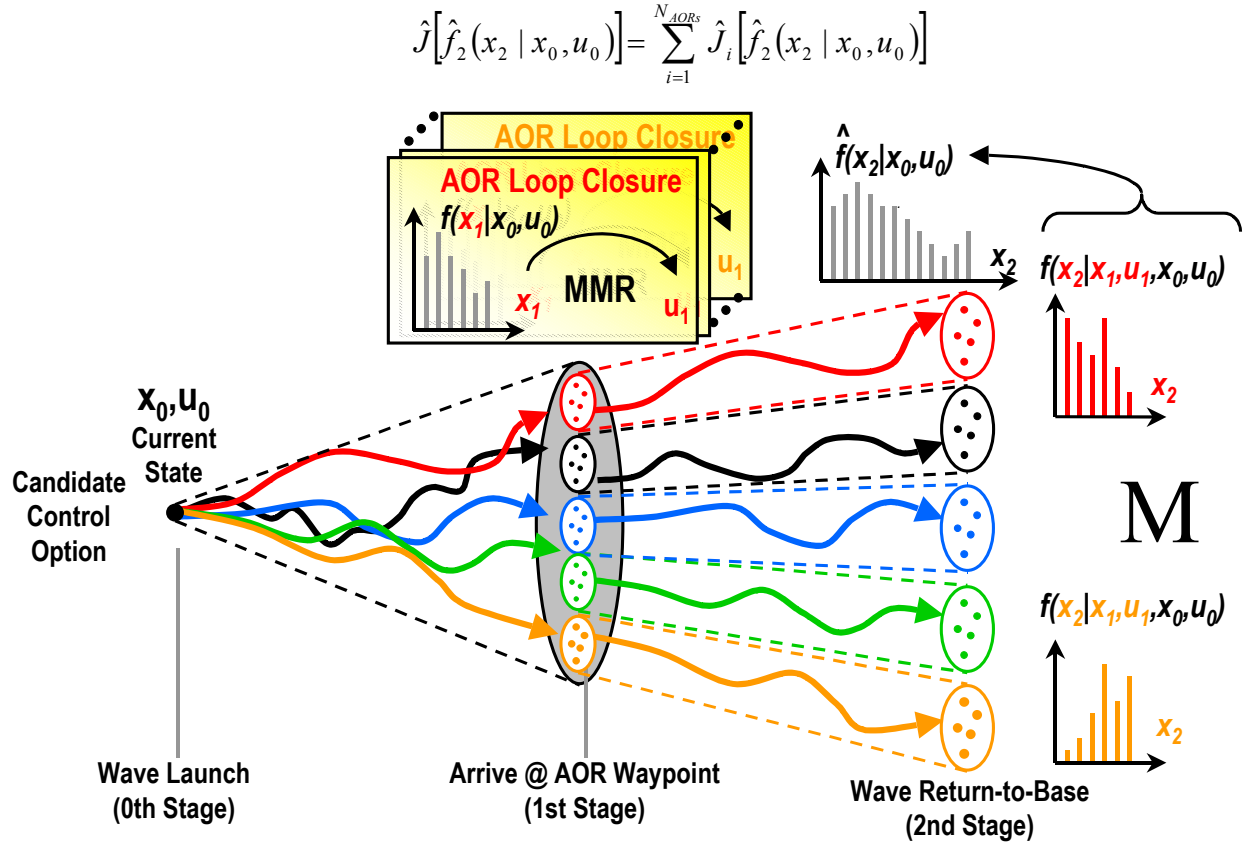


Figure 28 Random Sampling for AOR Prediction

Using the Surrogate Method as a base controller and the MMR as an AOR controller, this approach requires $O(N_{Asset_Types} \cdot N_{Destinations} \cdot N_{Iterations} \cdot N_{Samples} \cdot N_{AOR} \cdot MMR_{AOR})$ single stage evaluations using our design model. N_{Asset_Types} is the number of asset types, i.e., 2 in this case. $N_{Destinations}$ is the number of destinations assignable from base. $N_{Iterations}$ is the number of iterations the Surrogate Method is allowed to converge. N_{AOR} is the number of AORs. MMR_{AOR} is the computational complexity of the MMR assignment at each AOR point.

4.6.5.2 Certainty Equivalent Approximation

In Figure 29, we illustrate a certainty equivalent approach used to predict the performance within each AOR. Our design model is used to predict the value ingress to the AOR and provides the (approximate) distribution $f(x_1 | x_0, u_0)$ over the state x_1 . Assuming that each AOR is independent, a certainty equivalent state $\bar{x}_1$ is selected to represent the entire distribution. In our experiments, we consider the mean and mode of the distribution in each AOR. Given the

certainty equivalent state $\bar{x}_1$, a control $\bar{u}_1 = \mu_{MMR}(\bar{x}_1)$ is determined using the MMR algorithm. Given the state $\bar{x}_1$ and the control $\bar{u}_1$, our design model is used to compute the estimated value of the second stage in AOR i .

$$\hat{J}_i[f(x_2|x_0, u_0)] = J_i[f(x_2|x_1, u_1, x_0, u_0)]$$

The overall performance of the second stage is determined by combining the value of individual AORs.

$$\hat{J}[f(x_2|x_0, u_0)] = \sum_{i=1}^{N_{AORs}} \hat{J}_i[f(x_2|x_0, u_0)]$$

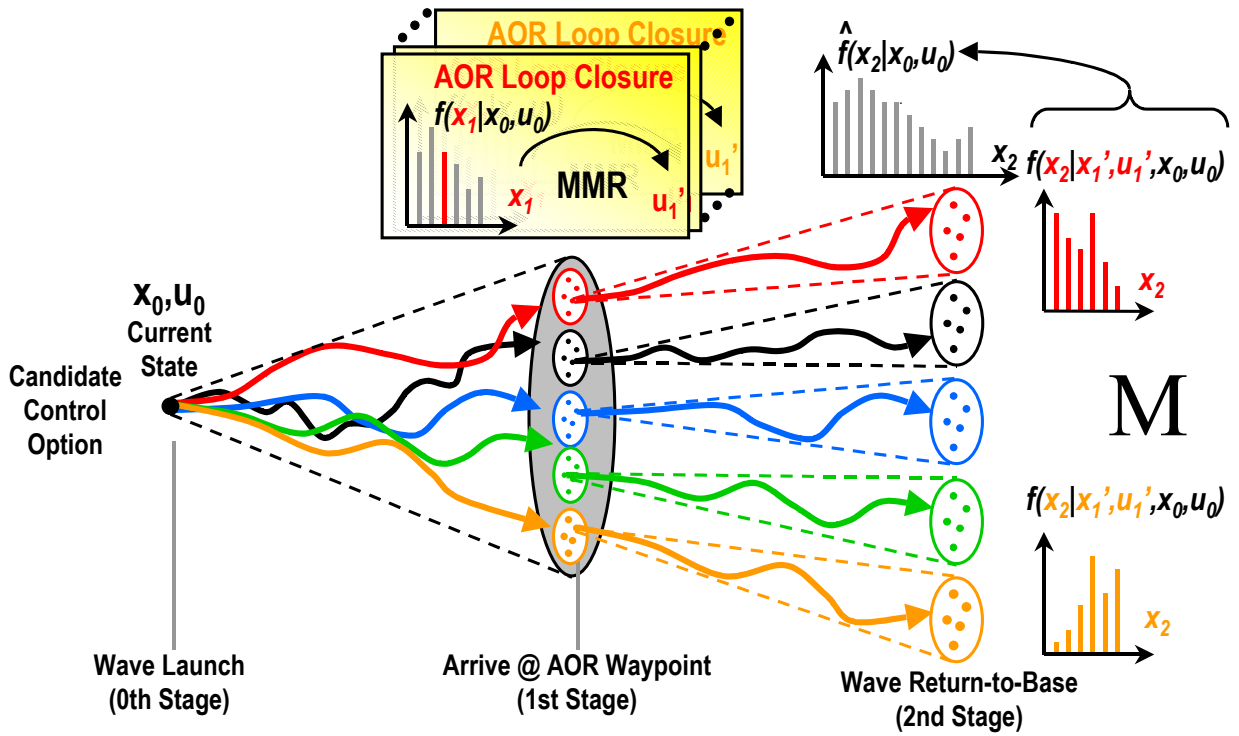


Figure 29 Certainty Equivalent Approximation for AOR Prediction

Using the Surrogate Method as a base controller and the MMR as an AOR controller, this approach requires $O(N_{Asset_Types} \cdot N_{Destinations} \cdot N_{Iterations} \cdot N_{AOR} \cdot MMR_{AOR})$ single stage evaluations using our design model.

4.6.5.3 Partial Open-loop Approximation

In Figure 30, we illustrate a partial open-loop approach to predict the performance within each AOR. Our design model is used to predict the value ingress to the AOR and provides the

(approximate) distribution $f(x_1|x_0, u_0)$ over the state x_1 . Assuming that each AOR is independent, we consider the k th state of “gorilla” air package $x_1^{AP_k}$ and adopt a certainty equivalent state $\bar{x}_1^{Enemy}$ for the targets and threats.

The “gorilla” air package state $x_1^{AP_k}$ and the certainty equivalent state $\bar{x}_1^{Enemy}$ of the targets and threats are combined to establish the state x_1 . A control $u_1 = \mu(x_1)$ is determined using the MMR algorithm. Our design model is used to compute the value in AOR i ,

$J_i[f(x_2|x_1^{AP_i}, \bar{x}_1^{Enemy}, u_1, x_0, u_0)]$ as a function of the “gorilla” air package state. The estimated value on the second stage in AOR i is given by

$$J_i[f(x_2|x_0, u_0)] = \sum J_i[f(x_2|x_1^{AP_i}, \bar{x}_1^{Enemy}, u_1, x_0, u_0)] \cdot P(x_1^{AP_i})$$

The overall performance of the second stage value is determined by combine the value of individual AORs

$$\hat{J}[f(x_2 | x_0, u_0)] = \sum_i^N J_i[f(x_2 | x_0, u_0)].$$

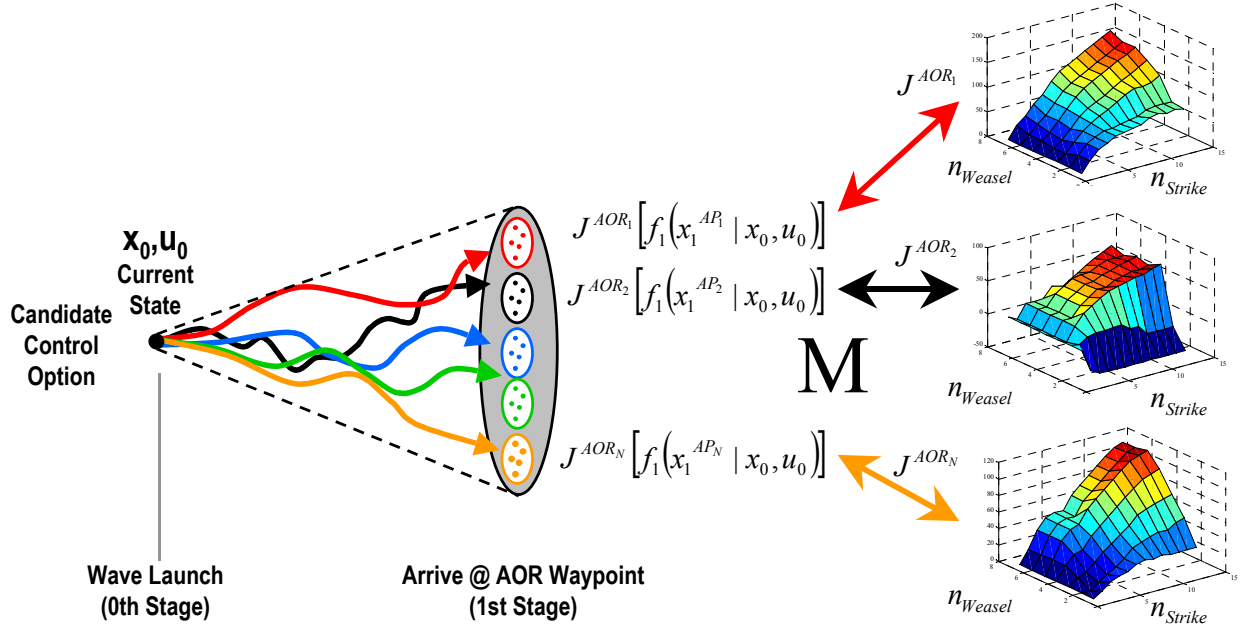


Figure 30 Partial Open-Loop Approximation for AOR Prediction

Using the Surrogate Method as a base controller and the MMR as an AOR controller, this approach requires $O(AOR_{Strike} \cdot AOR_{Weasel} \cdot N_{AOR} \cdot MMR_{AOR})$ single stage evaluations using our

design model. AOR_{Strike} is the number of strike aircraft per AOR. AOR_{Weasel} is the number of weasel aircraft per AOR.

4.6.6 2-Stage/1-Wave Model with AOR Tasking and ISR Collection

Similar to the previous AOR section, this controller allocates air packages to AORs, which are subsequently tasked to specific targets. However in this case, we consider only partial information, i.e., not all objects are observed prior to tasking within individual AORs. Similar to the previous section, this addresses scalability by decomposing the 1-wave problem into sub-problems, but also begins to address partial observations and imperfect information at the decision point.

Information dynamics describe how information evolves over time. These dynamics are captured in the probabilistic models associated with each battlespace object, as well as observation dynamics. Up to this point, all objects have been perfectly observed at each decision point. In this case, only some objects will be observed, while the probabilistic state of others evolves over time. In the absence of observations (and interactions), the probabilistic state will eventually converge to a steady state distribution which may be computed directly from the probabilistic model associated with the object. In Figure 31, we illustrate the information dynamics associated with a single object. Assuming no previous observations, the initial information state correspond to the steady state distribution $P_{Active} = 0.4$ (red),

$P_{Inactive} = 0.35$ (yellow), and $P_{Unknown} = 0.25$. There is no change in the information state until an observation occurs at which point we observe the object perfectly, $P_{Active} = 1.0$. At that point (or once the object is no longer observed), information starts to degrade (i.e., we are less certain what state the object is in). If a decision needs to be made regarding this object, we note that there is less ambiguity and thus more information during or immediately following an observation. We expect this behavior to affect the types of decisions that the base controller makes. Namely, that aircraft should be sent to AORs that have more information (less ambiguity regarding the state of the associated objects).

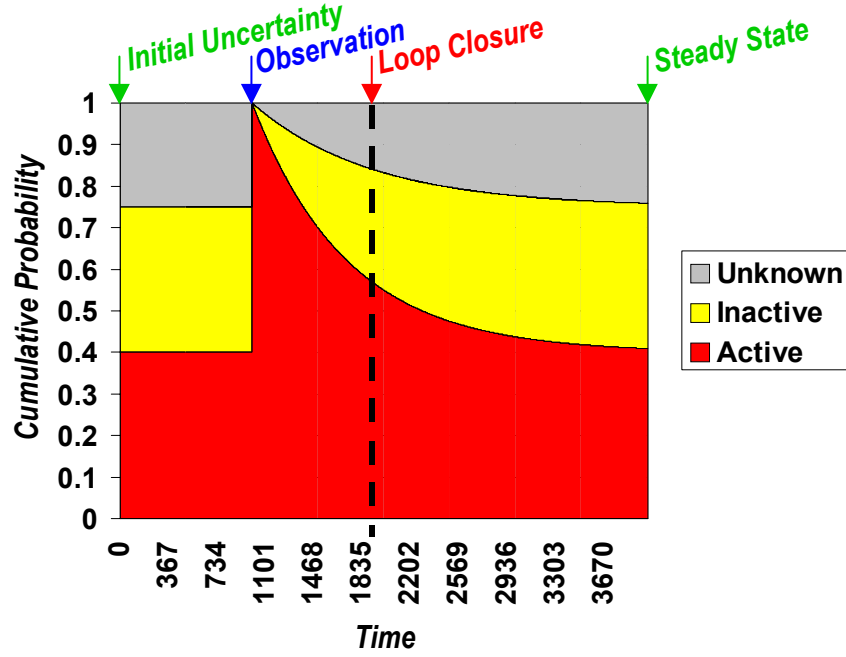


Figure 31 Information Dynamics

In this case, the AOR control problem is the same as in the previous section, except that the information available at the AOR points will be based on previous partial observations. Depending on the amount of time that has passed since the last observation, information regarding targets and air defenses will degraded. Knowing where and when observations will occur, we expect that the base controller will send aircraft to the AORs that will have better information at the associated decision point. As in the previous case, we neglect the coupling among AORs and that due to threats during egress. The base controller uses one of the combinatorial algorithms to assign aircraft to AOR points, and the predictor is modified to account for the subsequent tasking of aircraft to targets when they arrive at an AOR. This is illustrated in Figure 32. Our baseline predictor is used to determine the value during ingress to the AOR points, but also determines the distribution $f(I_1|I_0, u_0)$ from which the observations $I_1 = \{z_1, z_2, K\}$ are drawn. Observations that occur during ingress are projected forward (i.e., the information is degraded) to the decision point so that a current estimate $\hat{x}_1 = P(x_1|I_1)$ of the state may be used to determine the control decision $u_1 = \mu_{MMR}(\hat{x}_1)$ for the AOR. Given the control decision u_1 and the estimate of the current state $\hat{x}_1$, we evaluate the second stage in each AOR, $J_i[f(x_1|I_1, u_1, I_0, u_0)]$. We use the same methods that were used in the previous section to

estimate the second stage performance, except that the random sampling and certainty equivalent approximations are taken in terms of the observations rather than the current state. Since the partial open-loop approximation enumerates the air package state $x_1^{AP_k}$, which we assume to be perfectly observable, only certainty equivalent $\bar{x}_1^{Enemy}$ is considered in terms of the observations.

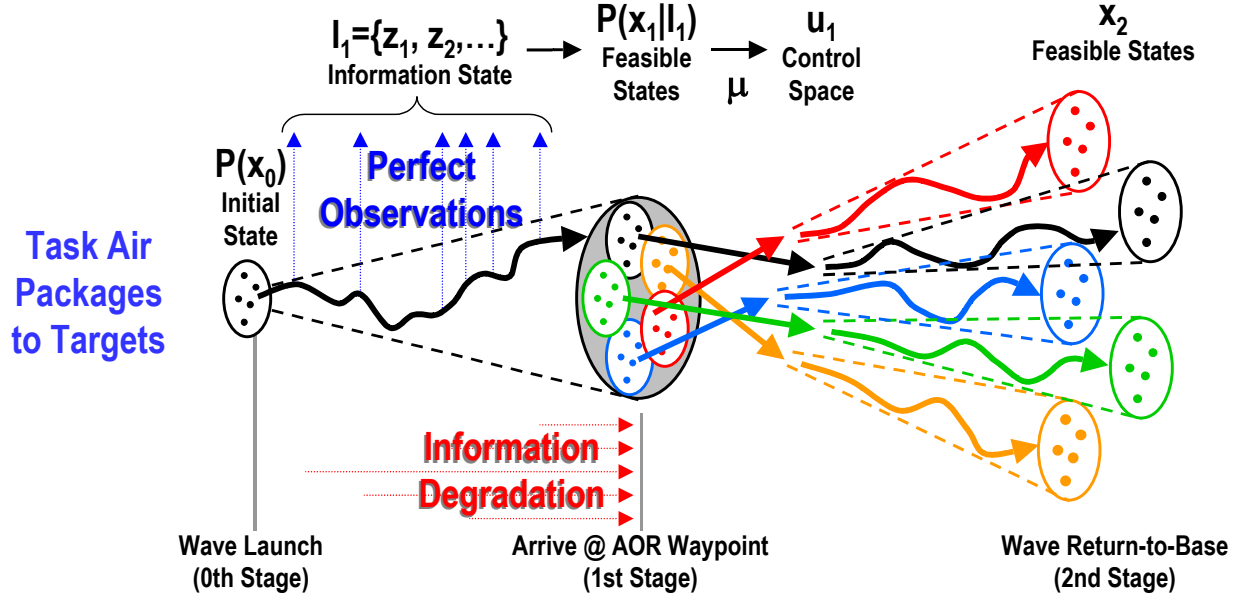


Figure 32 AOR Prediction with Partial Information

4.6.6.1 Random Sampling

In Figure 33, we illustrate the random sampling approach used to predict the performance within each AOR. Our design model is used to predict the value ingress to the AOR and provides the distribution $f(I_1|I_0, u_0)$ from which the observations $I_1 = \{z_1, z_2, K\}$ are drawn. The observations are selected randomly according to the known distribution $f(\mathcal{G}_1|I_0, u_0)$ and projected forward to the decision point, providing the current state estimate $\bar{x}_1 = P(\mathcal{G}_1|I_1)$. Based on the state estimate $\bar{x}_1$, a control $u_1 = \mu_{MMR}(\bar{x}_1)$ is determined for each AOR. Given the state estimate $\bar{x}_1$ and the control u_1 , our design model is used to compute the value in each AOR, $J_i[f(\mathcal{G}_2|I_1, u_1, I_0, u_0)]$ given the intermediate state estimate $\bar{x}_1$, from which we estimate the second stage performance in AOR i .

$$\bar{J}_i[f(\mathcal{G}_2|I_0, u_0)] = \frac{1}{N_{samples}} \sum_{i=1}^{N_{samples}} J_i[f(\mathcal{G}_2|I_1, u_1, I_0, u_0)]$$

The overall performance of the second stage is determined by combining the value of the individual AORs.

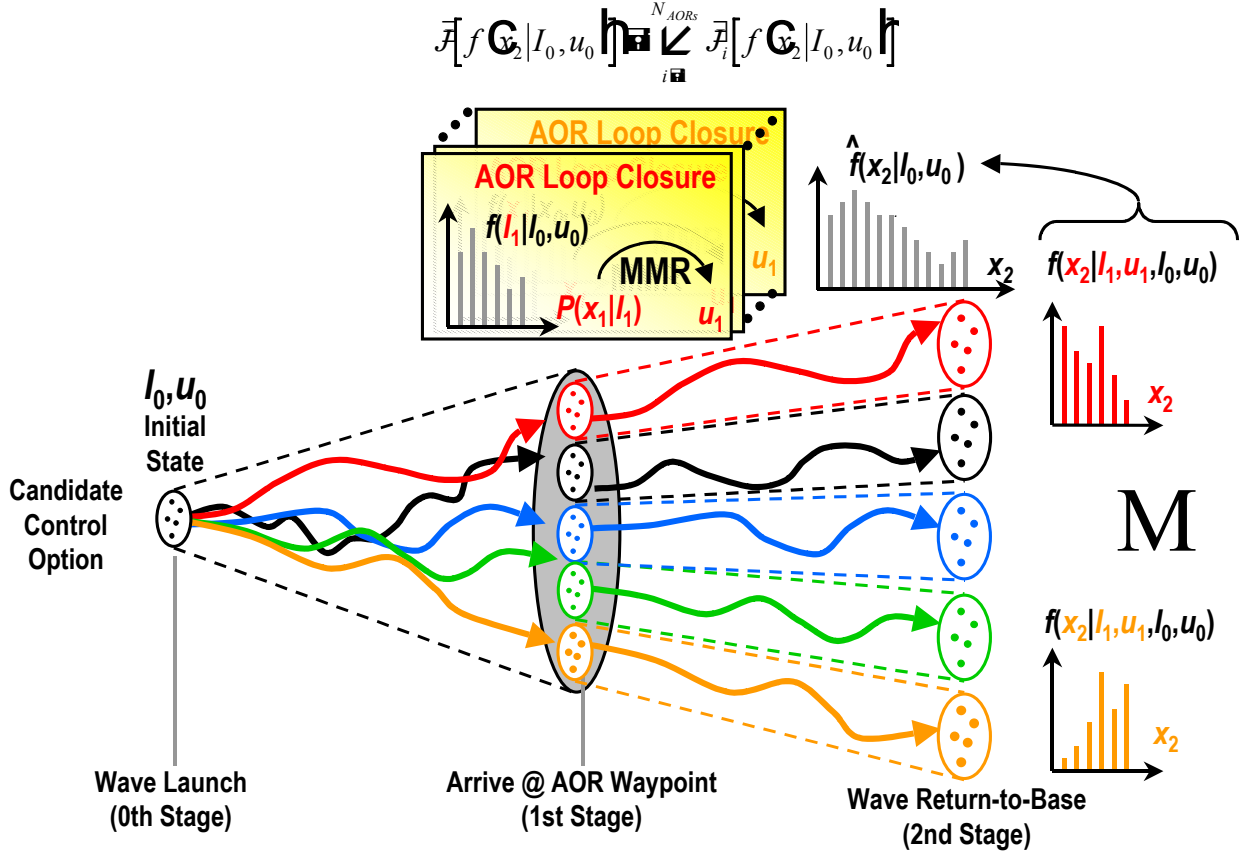


Figure 33 Random Sampling for AOR Prediction with ISR Uncertainty

Using the Surrogate Method as a base controller and the MMR as an AOR controller, this approach requires $O(N_{Asset_Types} \cdot N_{Destinations} \cdot N_{Iterations} \cdot N_{Samples} \cdot N_{AOR} \cdot MMR_{AOR})$ single stage evaluations using our design model.

4.6.6.2 Certainty Equivalent Approximation

In Figure 34, we illustrate the certainty equivalent approach used to predict the performance within each AOR. Our design model is used to predict the value ingress to the AOR and provides the distribution $f(I_1|I_0, u_0)$ from which the observations $I_1 = \{z_1, z_2, K\}$ are drawn. The certainty equivalent observation $\bar{I}_1$ is selected to represent the entire distribution and projected forward to the decision point, providing the current state estimate $\hat{x}_1 = P(x_1|\bar{I}_1)$. Based

on the state estimate $\hat{x}_1$, a control $u_1 = \mu(\hat{x}_1)$ is determined for each AOR. Given the state estimate $\hat{x}_1$ and the control u_1 , our design model is used to compute the value in each AOR i .

$$\hat{J}_i[f(x_2|I_0, u_0)] = J_i[f(x_2|\bar{I}_1, u_1, I_0, u_0)]$$

The overall performance of the second stage is determined by combining the value of the individual AORs.

$$\hat{J}[f(x_2|I_0, u_0)] = \sum_{i=1}^{N_{AORs}} \hat{J}_i[f(x_2|I_0, u_0)]$$

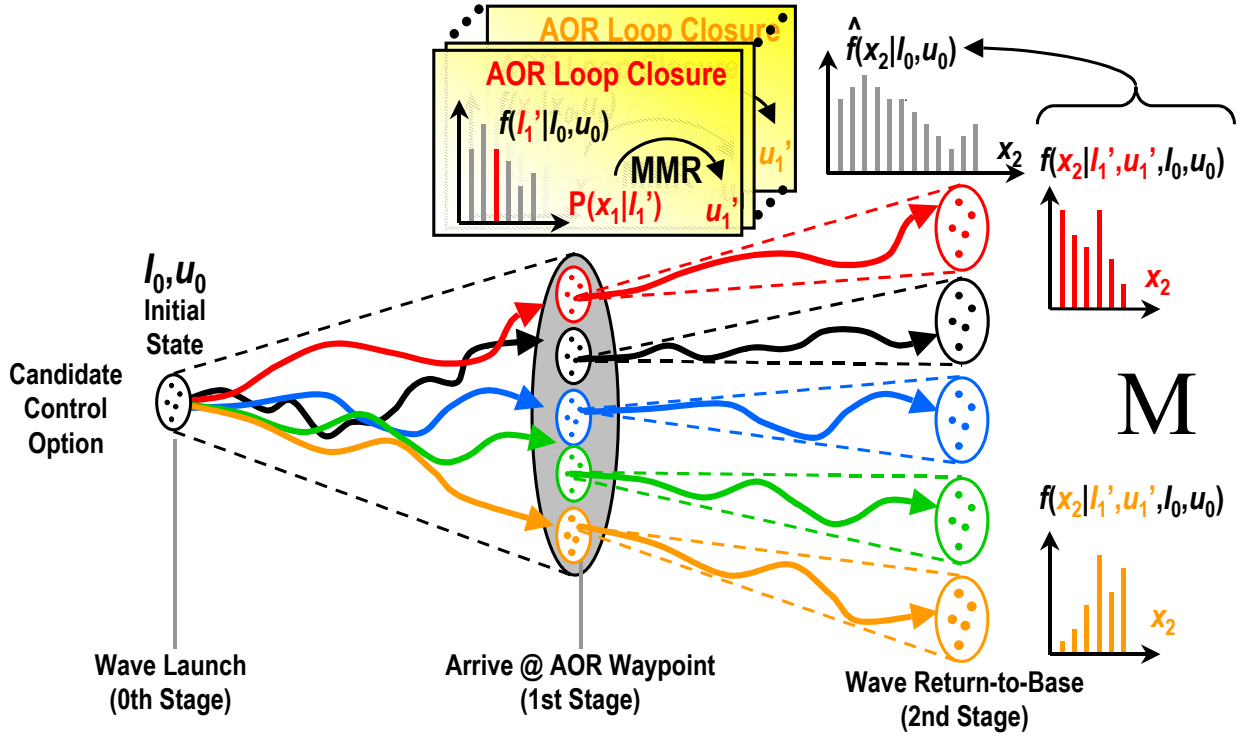


Figure 34 Certainty Equivalent Approximation for AOR Prediction with ISR Uncertainty

Using the Surrogate Method as a base controller and the MMR as an AOR controller, this approach requires $O(N_{Asset_Types} \cdot N_{Destinations} \cdot N_{Iterations} \cdot N_{AOR} \cdot MMR_{AOR})$ single stage evaluations using our design model.

4.6.6.3 Partial Open-loop Approximation

In Figure 35, we illustrate a partial open-loop approach to predict the performance within each AOR. Our design model is used to predict the value ingress to the AOR and provides the (approximate) distribution $f(I_1|I_0, u_0)$ from which the observations $I_1 = \{z_1, z_2, K\}$ are drawn.

We consider the k th state of the “gorilla” air package $x_i^{AP_k}$ and adopt a certainty equivalent observation $\bar{I}_1^{Enemy}$ for the targets and threats.

The “gorilla” air package state $x_1^{AP_k}$ and the certainty equivalent observation $\bar{I}_1^{Enemy}$ of the targets and threats are combined to estimate the current state $\hat{x}_1 = P(x_1 | x_1^{AP_i}, \bar{I}_1^{Enemy})$. Based on the state estimate $\hat{x}_1$, a control $u_1 = \mu(\hat{x}_1)$ is determined for each AOR. Given the state estimate $\hat{x}_1$ and the control u_1 , our design model is used to compute a value in AOR i , $J_i[f(x_2 | x_1^{AP_i}, \bar{I}_1^{Enemy}, u_1, I_0, u_0)]$ as a function of the “gorilla” air package state. The estimated value of the second stage in each AOR i is given by

$$\hat{J}_i[f(x_2 | I_0, u_0)] = \sum_k J_i[f(x_2 | x_1^{AP_i}, \bar{I}_1^{Enemy}, u_1, I_0, u_0)] \cdot P(x_1^{AP_i})$$

The overall performance of the second stage value is determined by combining the value of individual AORs

$$\hat{J}[f(x_2 | I_0, u_0)] = \sum_{i=1}^{N_{AORs}} \hat{J}_i[f(x_2 | I_0, u_0)].$$

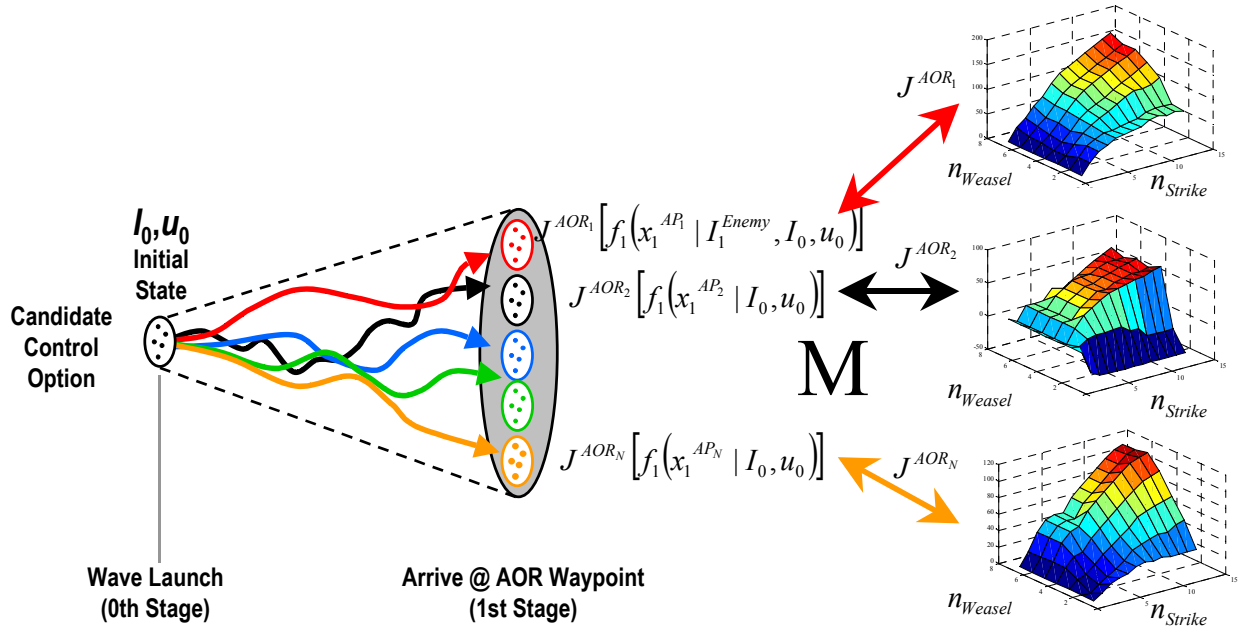


Figure 35 Partial Open-loop Approximation for AOR Prediction with ISR Uncertainty

Using the Surrogate Method as a base controller and MMR as an AOR controller, this approach requires $O(AOR_{Strike} \cdot AOR_{Weasel} \cdot N_{AOR} \cdot MMR_{AOR})$ single stage evaluations using our design model.

4.7 JAO SCALABILITY ASSESSMENT

As presented in Section 4.2, the goal of this research was to develop ADP algorithms that produce operationally consistent behaviors for realistic sized JAO scenarios. One immediate complexity reduction was achieved by adopting a hybrid, multi-rate control architecture that tailors the application of control, i.e. reactive or proactive, for the battlespace situation at hand; however, additional complexity reduction was required. Thus, the research was focused on developing fast and efficient combinatorial assignment and prediction models that form the foundation of the ADP algorithm. In this end, it was the goal of this research to develop a spectrum of ADP control strategies that can be mixed and matched to provide a broad range of performance and computational complexity. This was achieved. In the previous four sections, the details of the efficient combinatorial assignment algorithms along with the fast analytic prediction models were presented. Accordingly, Figure 36 summarizes the ADP algorithms that were developed and implemented as part of this research. Again, depending on the battlespace situation, different assignment algorithms and prediction models can be combined to produce a tailored application of control.

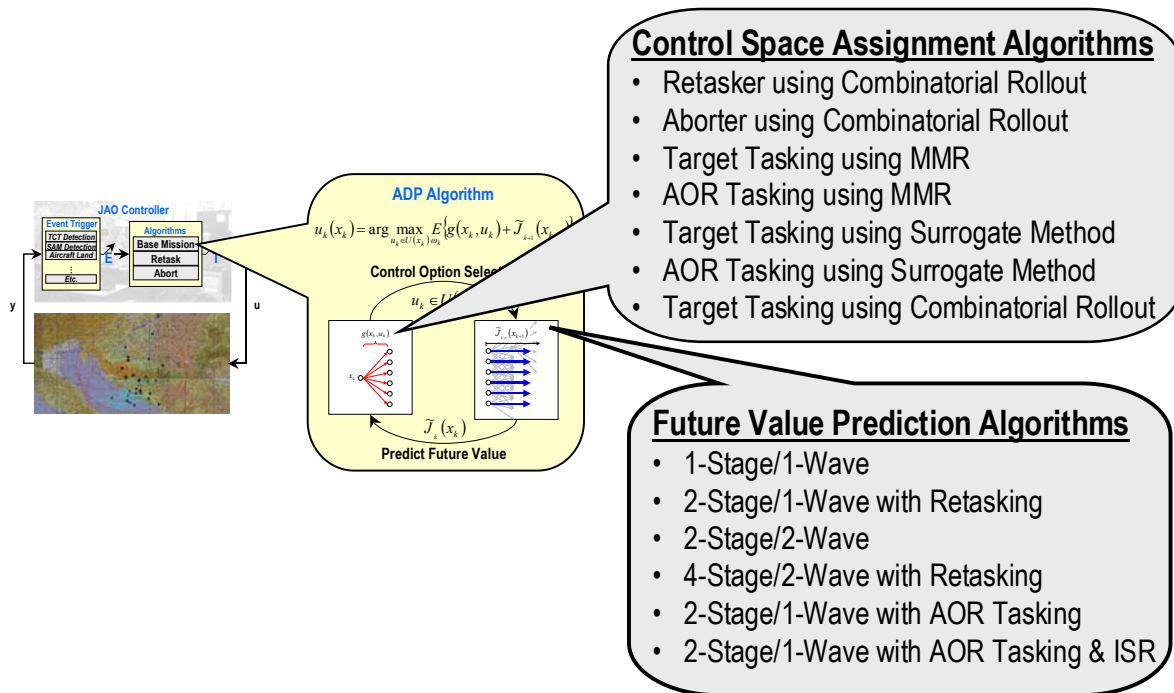


Figure 36 ADP Algorithms Developed and Implemented in Hybrid, Multi-Rate Control Architecture

Having developed and implemented these algorithms, the question remains. Do these ADP algorithms produce proactive, operationally consistent behaviors in real-time or near real-time for realistic sized JAO scenarios? The behavioral part of this question is the topic of the next chapter. Here we present the scalability assessment for the different ADP combinations implemented for generating a base mission queue without AOR tasking. Figure 37 presents the computation complexity of the base mission controllers which represent an upper bound on the computation complexity of the ADP implemented in ALPHATECH's BMC3 Development

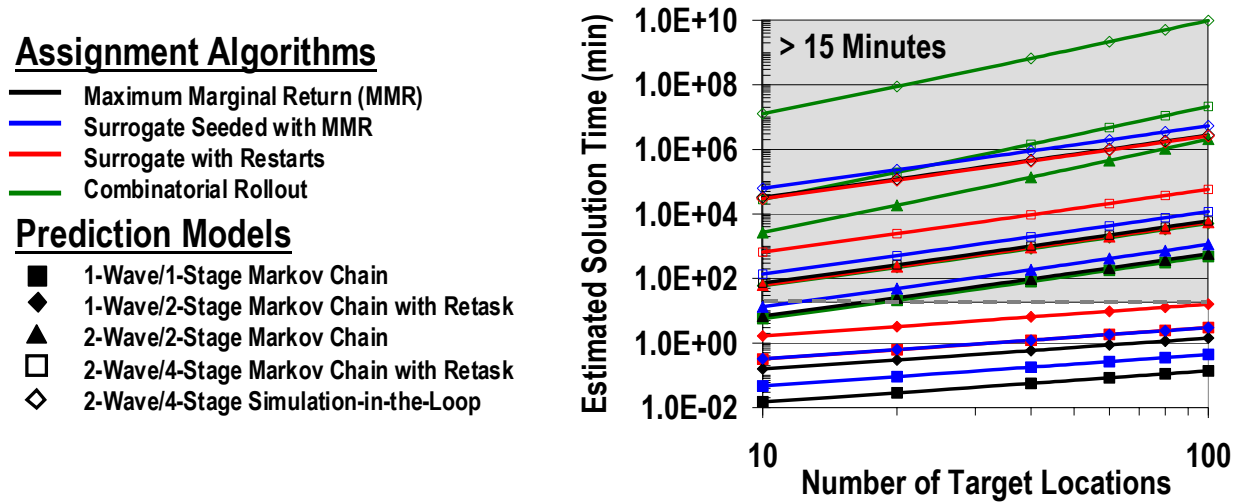


Figure 37 Scalability Assessment of ADP Algorithms Developed and Implemented for Hybrid, Multi-Rate Control Architecture (Single CPU)

Environment. Note, the shaded area represents control solutions that require more than 15 minutes to compute. Furthermore, a 100-target scenario corresponds to approximately 30 target locations.

The assumptions for this assessment are as follows:

- Base Mission Controller Generating Mission Queue
- Base Resources: 24 Strike and 12 Weasel Aircraft
- Maximum Package (6 Strike, 2 Weasel)
- 3.57 DMPIs per Target Location
- MMR for Future Base Loop Closures
- CE Analytical Predictors
- Performance on 850 MHz CPU
- Distributed on 125 CPU Array

It is seen from this figure that there is true a spectrum of solution approaches in terms of computation complexity. Figure 38 illustrates the computation complexity if is assumed that the AOC has distributed computational capability.

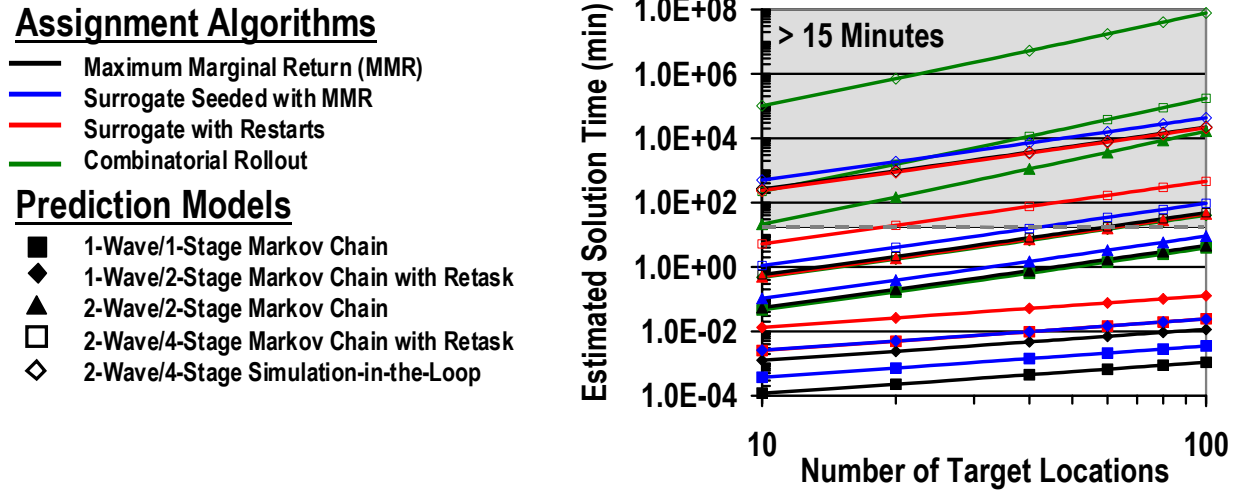


Figure 38 Scalability Assessment of ADP Algorithms Developed and Implemented for Hybrid, Multi-Rate Control Architecture (125 CPUs)

Thus, it is seen from the two figures above that there is a variety of base mission controller that can generate mission queues in near real-time for realistic sized JAO scenarios. In particular, the distributed MMR with analytical 4-stage/2-wave predictors provides near real-time computation performance, for scenarios with approximately 60 target locations. Likewise, the distributed Surrogate Method with analytical 2-stage/2-wave predictors provides near real-time computation performance, for scenarios with approximately 60 target locations.

In summary, this scalability assessment illustrated that real time or near real-time computational performance is achievable for most of the ADP algorithms developed and implemented as part of this research. Furthermore, this scalability assessment indicated that many of the ADP controllers could provide near real-time performance for scenarios with 250 targets if some modest parallel computation capability was available in an AOC.

5 EXPERIMENTATION RESULTS

In this section, experimental results that illustrate proactive, operationally consistent control strategies for the ADP algorithms developed in the previous section will be presented. The primary emphasis of this experimentation is to demonstrate the benefits of proactive versus reactive control strategies for relevant JAO problems. In this end, control behaviors and empirical simulation results will be presented to highlight the differences between the two control paradigms. All experimental results were generated using ALPHATECH's BMC³ Development Environment.

The presentation of these experimental results is organized such that we begin with simple JAO problems and then build upon them in both terms of scenario complexity and controller complexity.

5.1 DEMONSTRATION SCENARIO

The scenario used for this assessment is based on the Cyberland Scenario provided by the DARPA/JFACC Program Office. For this scenario, it has been assumed that a higher level of decomposition, both spatial and temporal, of the JFACC objectives has been performed. The enclosed scenario represents a 24-hour segment of the air campaign and the Area of Responsibility (AOR) is the Northern Air Defense (AD) District in West Cyberland. Key features of this scenario include approximately 100 targets and threats and 36 air vehicle assets of which there are 24 generic strike aircraft and 12 generic weasel aircraft. As noted in Section 3, targets may be known, unknown (hidden), time critical, or destroyed. Threats may be active, inactive, unknown, repairing, or destroyed. There is also uncertainty in the interaction between enemy and friendly assets that depends on the characteristics of the target or threat and the composition of the air package involved, i.e. the number of strike and weasel aircraft, and also on the geometry of the interaction.

Figure 39 illustrates the target/threat laydown used for this experimental evaluation. It is seen from this figure that there are 24 normal target locations, which are represented by . A normal target is a generic target that is known for all time and has 4 Designated Mean Points of Impact (DMPIs). The scenario also contains 4 TCT regions, each containing 2 emerge locations. In this context, TCTs are static, and are characterized by a single DMPI per emerge location. The TCTs emerge and hide based on stochastic processes; through the combination of temporal

and event-based stochastic transitions, the TCTs are on average only vulnerable for approximately 15 minutes in this scenario. When the TCT is hiding, it is represented by , and when vulnerable, it is represented by . It is also seen from this figure that the scenario contains 13 SAM sites of varying size. The position of the SAM site in this scenario is represented by , and the ring around the SAM represents its lethal range. The color of this ring represents the status of the SAM sites; the color scheme is as follows: red represents radar on, yellow represents radar off, green represents under repair, and black represents unknown.

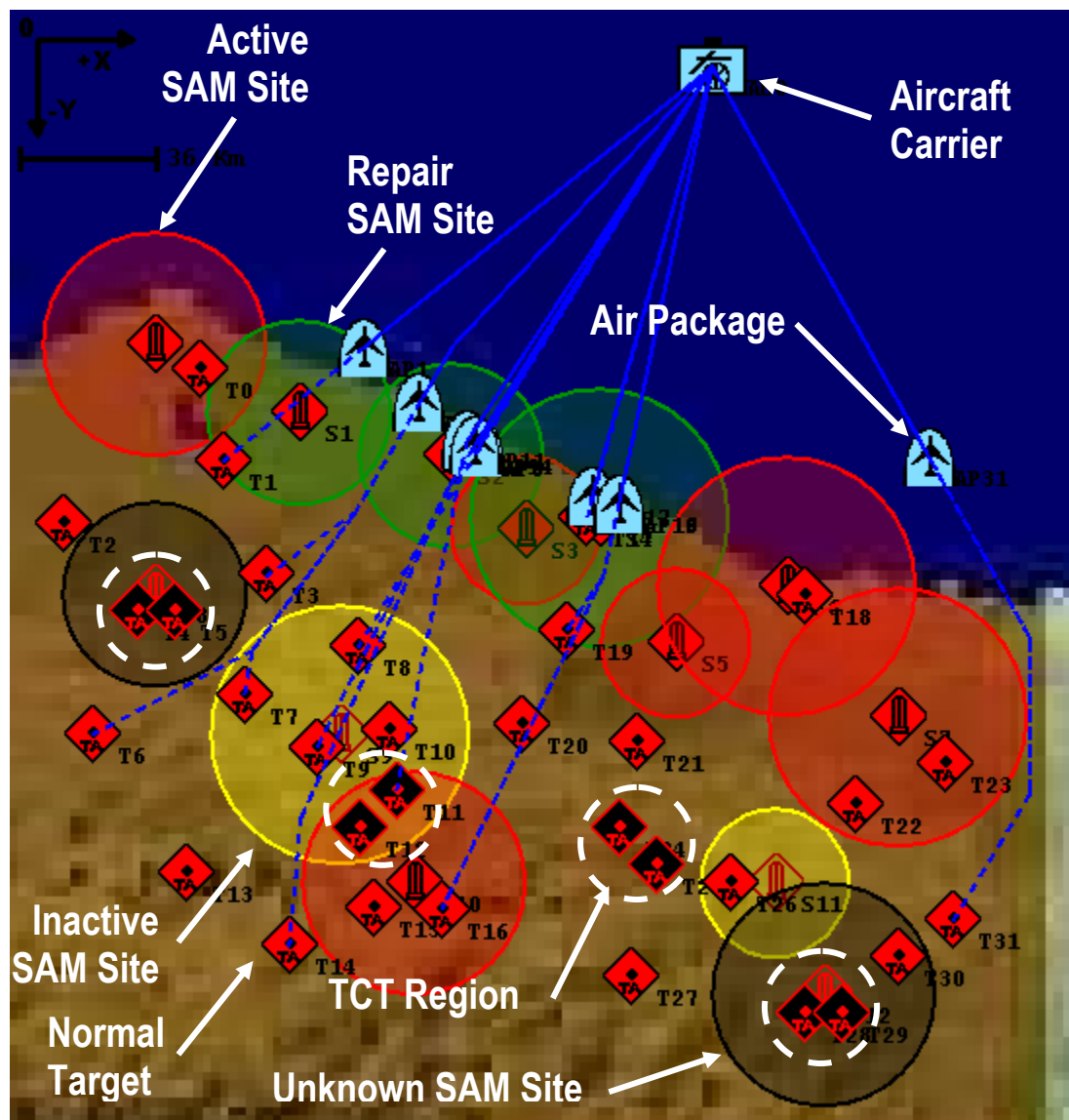


Figure 39 Demonstration Scenario Used for Experimental Evaluation

In terms of the friendly assets, there is an aircraft carrier positioned off the northern coast, which contains half of a squadron of generic strike and weasel aircraft available for this JFACC objective. There is a total of 24 strike aircraft and 8 weasel aircraft. As mentioned in the previous section, the controller composes and tasks air packages to target locations. In this scenario, an air package is represented by , and its mission is denoted by . It is assumed in this scenario that the maximum air package size is 6 strike and 2 weasels. It is also assumed that aircraft are assigned to air packages in increments of 2. Given the air package composition and tasking, the controller has the capacity to define a risk avoidance route. For some experiments, this route is determined *a priori*, and in others, the route selection is part of the control space. Finally, it is assumed in this scenario that air vehicles do not have adequate range to circumvent this defense posture.

Finally, as noted in Section 4, a relative valuation scheme is required to distinguish control options. For this scenario, all normal targets are valued at 40 points and TCTs are valued at 400 points. In terms of the airborne assets, both strike and weasel aircraft are valued at 40 points each. Note, SAM sites have no explicit value, however, they clearly have implicit value given that they affect the attrition of aircraft. Given this valuation scheme, the performance metric used for the control optimization is the sum of the target value destroyed minus the aircraft lost.

In summary, Figure 39 illustrates the baseline scenario that was used for a series of experiments that illustrate proactive, near real-time control performance of the controllers presented in Section 4. Again, the presentation of these experimental results is organized such that we begin with simple JAO problems and then build upon them in both terms of scenario complexity controller complexity. As the complexity increases, minor changes to the baseline scenario were required and will be called out in the appropriate sections.

5.2 1-STAGE/1-WAVE PROBLEM

The first set of experiments were performed to assess the accuracy of the analytic predictors over a set of control decisions. The details of this analytic prediction model are contained in Section 4.6.1. To perform this assessment, a one-wave problem was set up where the initial mission queue was defined *a priori* and no loop closures were permitted during the execution of the wave, i.e. no retasks or aborts. In this situation, the wave begins when all air

packages are launched at time $t=0$, and the wave ends when all air packages return to base. Results will be presented for the cases when the initial status of the enemy assets is either known or is specified by a distribution. Finally, straight line routing was used for all air packages since this condition results in higher attrition, and thus will highlight the prediction accuracy of the baseline analytic prediction model presented in Section 4.6.1.

The first experiment that was run to make this assessment was for the case when the initial status of the enemy assets was known *a priori*. This situation is illustrated in Figure 40 where the initial state is known at $t=0$, and the battlespace state is propagated to the wave return-to-base. Using the prediction model presented in Section 4.6.1, the estimated

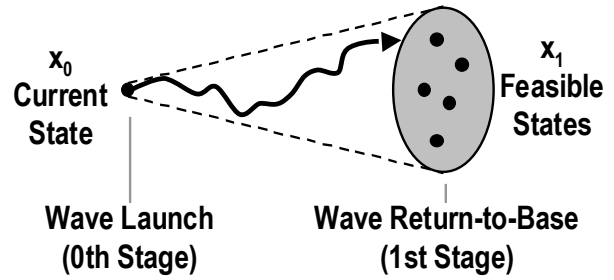


Figure 40 Battlespace State Propagation Diagram for Known Initial State

performance of the air package assignment was obtained and compared to simulation runs. With 10,000 Monte Carlo performance runs, the performance prediction using the 1-stage/1-wave prediction model was statistically equivalent to the simulated performance.

Given the experimentally demonstrated accuracy of the 1-stage prediction model given a known initial state, the next step was to assess the accuracy of the prediction model for an uncertain initial state of the enemy. As discussed in Section 4.6, many of the multiple state prediction models require the propagation of the battlespace state given uncertain enemy status. As in the previous evaluation, the air package composition and tasking is specified as part of the initial state x_0 . Figure 41 illustrates the 1-stage prediction problem for the case where the initial state is defined by a distribution.

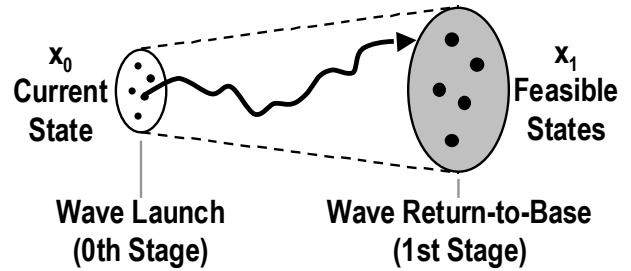


Figure 41 Battlespace State Propagation Diagram for Unknown Initial State

As in the previous evaluation, the air package composition and tasking is specified as part of the initial state x_0 . Figure 41 illustrates the 1-stage prediction problem for the case where the initial state is defined by a distribution. Given the air packages and enemy status distributions at $t=0$, the battlespace state was propagated to the return-to-base without any retasking or aborts. Based on the return-to-base distribution obtained from the prediction model, the expected performance was computed. For the empirical evaluation, 2,000,000 Monte Carlo evaluations were obtained by first generating 200 realizations of enemy status at $t=0$ and then for each sample, obtaining 10,000 samples of the

expected performance. Based on these evaluations, it was determined that the 1-stage prediction model was statistically optimistic by 6%.

5.3 2-STAGE/1-WAVE WITH RETASKING PROBLEM

The next set of experiments were performed to assess the performance of the 2-Stage/1-Wave Retasking model in both terms of prediction quality and control solution quality. The details of this analytic prediction model are contained in Section 4.6.2. For this 1-wave problem, all air packages launch at $t=0$, and when a TCT emerges, the retasker controller outlined in Section 4.5.1 is called to divert ingress air packages to the TCT. As before, the wave ends when all air packages return to base. As noted in Section 4.6.2, the difficulties of modeling the future retasking is the loop closure is dependent on the TCT emerge event which is stochastic based and multiple TCT emerge events can occur during a single wave. The results of this assessment will be presented below.

To perform the prediction accuracy assessment, the one-wave problem was set up where the initial mission queue was defined *a priori* and this mission queue was executed. Figure 48 illustrates the 2-Stage/1-Wave retasking prediction problem where the initial state includes the initial mission queue and the enemy status at $t=0$. Results will

be presented for the cases when the initial status of the enemy assets is either known or is specified by a distribution. To perform this assessment, the analytic model with retasking was compared to experimental evaluation and to the prediction model that does not model the retask loop closure, i.e. reactive control strategy.

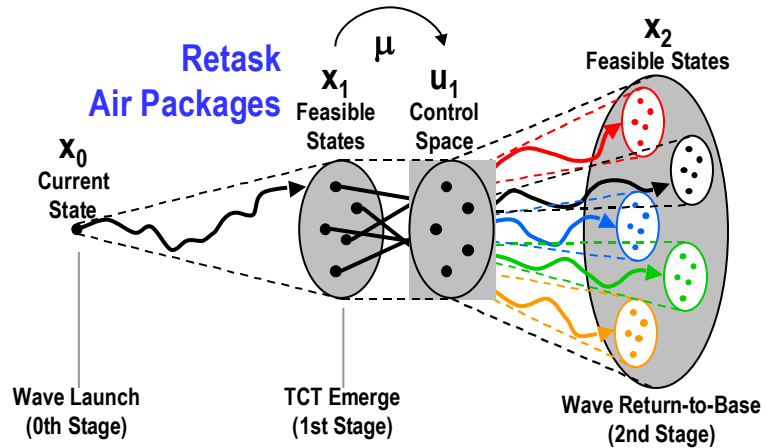


Figure 42 Battlespace State Propagation for 2-Stage/1-Wave Retasking Problem

Figure 43 illustrates the prediction accuracy of the two analytic prediction models compared to empirical evaluation. For the empirical evaluation, 10,000 Monte Carlo evaluations were performed to generate the empirical performance prediction. It is seen from the figure that the analytic model that accounts for the retasking loop closure is within 11% of the true expected performance. Furthermore, the analytic model that does not account for the retasking loop closure is only within 50% of the true expected performance.

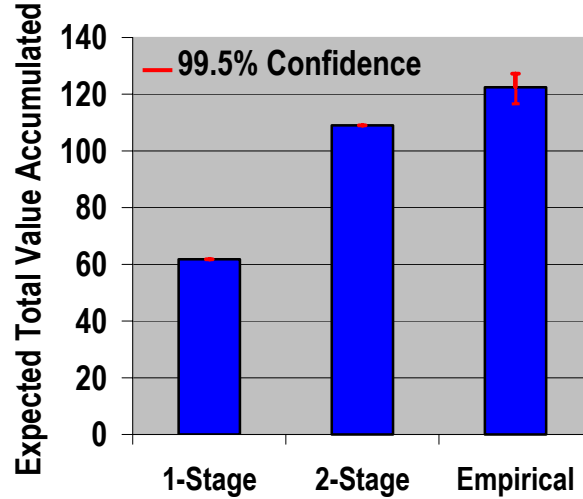


Figure 43 Prediction Accuracy of Different Design Models for the 2-Stage/1-Wave Retasking Problem with Known Initial State x_0

Next, the prediction accuracy assessment was performed for the situation where the initial status of the enemy is uncertain and is specified by distributions. Given the air packages and enemy status distributions at $t=0$, the battlespace state was propagated to the return-to-base with retask loop closures for TCT emerge events. As for the known enemy status case, results were obtained from the 2-Stage/1-Wave Retasking Predictor, 1-Stage/1-Wave Predictor, and

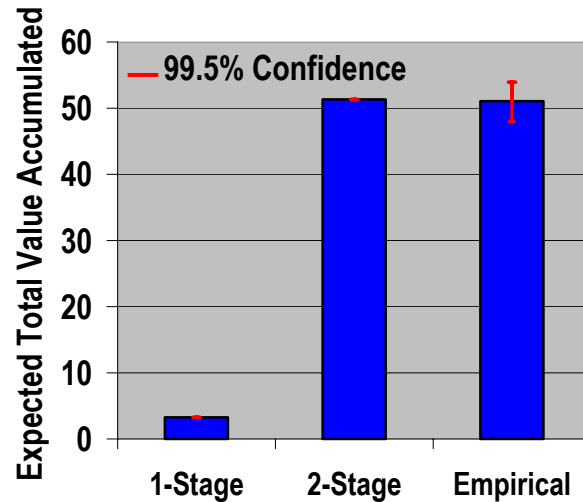


Figure 44 Prediction Accuracy of Different Design Models for the 2-Stage/1-Wave Retasking Problem with Unknown Initial State x_0

experimental evaluation. For the empirical evaluation, 33,500 Monte Carlo evaluations were obtained by first generating 335 realizations of enemy status at $t=0$ and then for each sample, obtaining 1,000 samples of the expected performance. Figure 44 illustrates the prediction accuracy of the two analytic prediction models compared to the empirical results. It is seen from

the figure that the analytic model that accounts for the retasking loop closure is statistically equivalent to the true expected performance. Furthermore, the analytic model that does not account for the retasking loop closure provides a poor prediction and is only 6% of the true expected performance. Thus, it is seen from this and the previous assessment that 2-stage/1-wave retasking analytic prediction model does provide a good approximation to the true expected performance. Furthermore, not unexpectedly, the 1-Stage/1-Wave analytic prediction model does not provide a good approximation to the true expectation.

Having presented the prediction accuracy, the question remains whether the higher fidelity, i.e. proactive, prediction model that anticipates the future retasking loop closures improves the base mission control solution. To perform this assessment, two mission queues were generated: one using the 2-stage analytic predictor that anticipates the retasking loop closure and one using the 1-stage analytic predictor that neglects the retasking loop closure. In both cases, the Surrogate Method, which was presented in Section 4.3.3, was used to search the control space, and the initial status of the enemy and thus the initial state x_0 is known. Figure 45 illustrates the two mission queues produced.

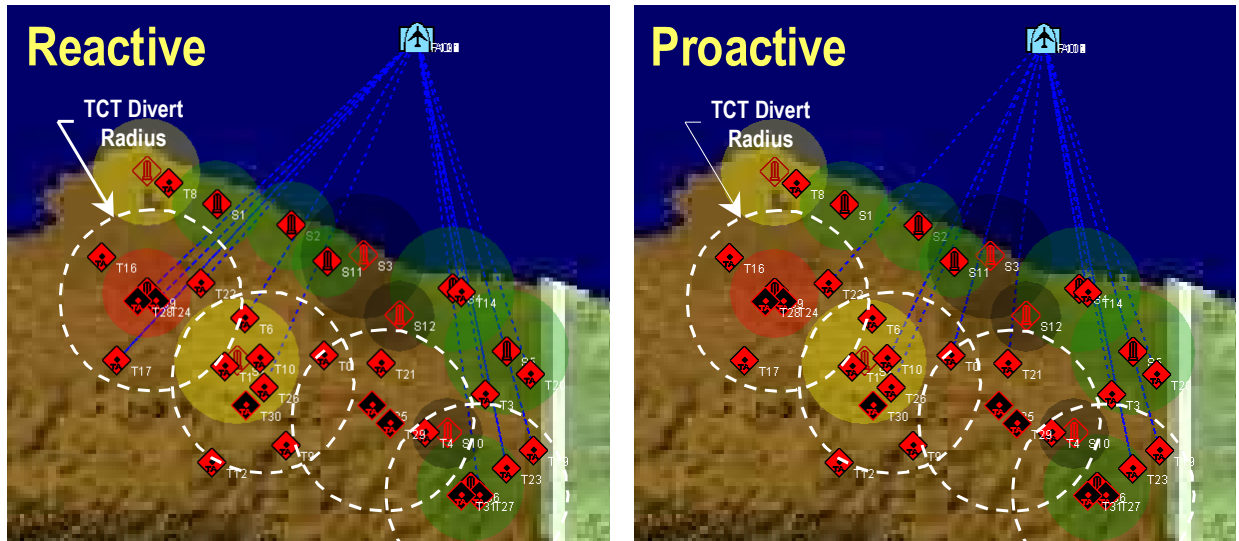


Figure 45 Behavioral Comparison of Proactive Versus Reactive Control Strategy for 2-Stage/1-Wave Retasking Problem

It is seen from this figure that the mission queue generated using the reactive prediction model assigns air packages to TCT locations. In comparison, the mission queue generated using the proactive prediction model does not assign air packages to any TCT locations, but instead strategically locates air packages such that each TCT has a minimum of two retask options. The

significance of this differing mission assignment is that the 2-stage model anticipates the fact that air packages in the vicinity of the TCT regions can be retasked in the event that a TCT emerges; if no TCT emerges, the air packages strike normal targets. Thus, for the 1-stage mission queue, air packages fly to TCT location and if no TCT emerges, the air package cannot achieve any positive value. On the other hand, for the 2-stage solution, air packages fly to normal targets in the vicinity of TCTs. If a TCT emerges during ingress, a retasking solution exists since an air package will be in range; if no TCT emerges, the air packages proceed to their normal targets and achieve positive value. Thus, by anticipating the TCT emerge event and the subsequent loop closure, resources can be more effectively tasked.

Having illustrated the behavioral differences between the base mission solutions, the performance numbers will now be presented. Figure 46 illustrates the performance and empirical performance prediction for the two mission queues.

Note, empirical results were obtained using 10,000 Monte Carlo samples. It is seen from this figure that 2-stage control solution exhibits a statistically significant

performance improvement over the 1-stage control solution. Furthermore, as expected,

the predicted performance of the two solutions is pessimistic. In the case of the 1-stage solution, the predicted performance does not account for any retasks, whereas the empirical solution does permit reactive retasking to a TCT emerge event. On the other hand, the lower predicted performance of the 2-stage model was identified in previous paragraphs to be related to the approximations relative to the uncertain loop closure time.

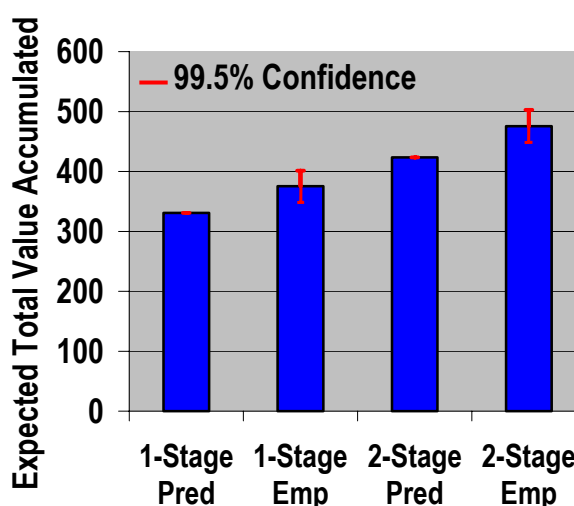


Figure 46 Control Performance for the 2-Stage/1-Wave Retasking Problem

The final piece to the performance story is to compare the proactive versus reactive controller complexity for this particular scenario. Figure 47 illustrates the number of 1-stage prediction function calls required to produce a control solution for the different control approaches. Note, as highlighted in Section 4.6.3, a single stage prediction model may require hundreds of one-stage prediction function calls to produce the estimated performance for a control option. It is seen from this figure that the 2-stage prediction algorithm only

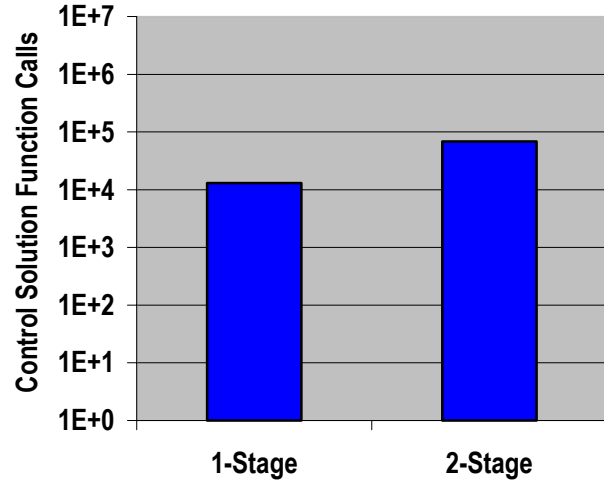


Figure 47 Controller Computational Performance for the 2-Stage/1-Wave Retasking Problem

requires a marginal 0.5 orders of magnitude more function calls. In terms of clock time, the reactive, 1-stage algorithm computes a base mission queue in ~1 minutes, whereas the proactive, 2-stage algorithm computes a higher quality mission queue in ~5 minutes.

To summarize these results, the 2-Stage/1-Wave with Retasking problem illustrates the benefits of anticipating future TCT emerge events and subsequent retasking controller loop closures. It was shown that the analytic prediction model that approximates the future loop closure does provide an accurate prediction for a given control option both for the cases where the initial enemy status is deterministic or defined by distributions. Furthermore, it was shown that using the analytic prediction model that anticipates the retasking loop closure produces proactive control solutions that strategically position air packages near TCT regions. Finally, experimental results show that the anticipatory approach provides a statically significant performance improvement over control solution that only reacts to the TCT emerge event.

5.4 2-STAGE/2-WAVE PROBLEM

In the previous section, the benefits of proactive versus reactive control were illustrated for the case when the prediction horizon was 1-wave but included the potential for retasking loop closures. In this assessment, the complexity of the problem is increased by extending the prediction horizon to include the second wave, without the potential for retasking loop closures.

Thus, this experiment will highlight the benefits of performing proactive control over a 2-wave problem. Accordingly, the 2-Stage/2-Wave prediction model presented in Section 4.6.3 will be used for this experiment.

To show this benefit, the scenario was constructed such that all normal targets have terminal time constraints, i.e. expiration deadlines, on their value, and these constraints were established such that they become active during the second wave execution. Additionally, routing risk was added to the control space; in this context, the controller could choose either a high or low risk route for the entire wave. The combination of target value deadlines and routing provides a design trade of managing risk and execution time—through either target tasking or route selection—to maximize the two-wave performance. Thus, for this two-wave problem, the controller determines the initial mission queue and routing option using a two-wave prediction model. Then, the air packages launch at $t=0$ and ingress to their respective targets, deploy their munitions, and egress to base. When all air packages return to base, the controller then determines the second wave mission queue and routing option using a 1-wave prediction model. Again the air packages launch from base, ingress to their respective targets, deploy their munitions, and return to base. The second wave—and the experiment—is complete when all air packages return to base. For this experiment, all mission queues were generated using the Maximum Marginal Return assignment algorithm discussed in Section 4.5.3.

As noted in Section 4.6.3, the difficulty of a 2-wave prediction is modeling the loop closure and subsequent mission generation at the end of the first wave. This is a difficult problem because there is a combinatorially large number of possible states at the end of the first wave. As a further complication, all of the assignment algorithms outlined in Section 4.3 produce a discrete mission queue for a discrete number of resources; thus, there is no control algorithm that maps a surviving aircraft distribution to a mission queue distribution. Given these complexities, a variety of 2-stage/2-wave prediction models will be assessed in this section. As in the previous sections, an assessment of the prediction quality and control performance will be made for each of these approaches.

To perform the prediction accuracy assessment, the two-wave problem was set up where the initial mission queue was defined *a priori*; thus, the only loop closure occurs when the first wave ends and the base controller determines the second wave mission queue. Figure 48 illustrates the 2-stage/2-wave prediction problem where the initial state includes the initial

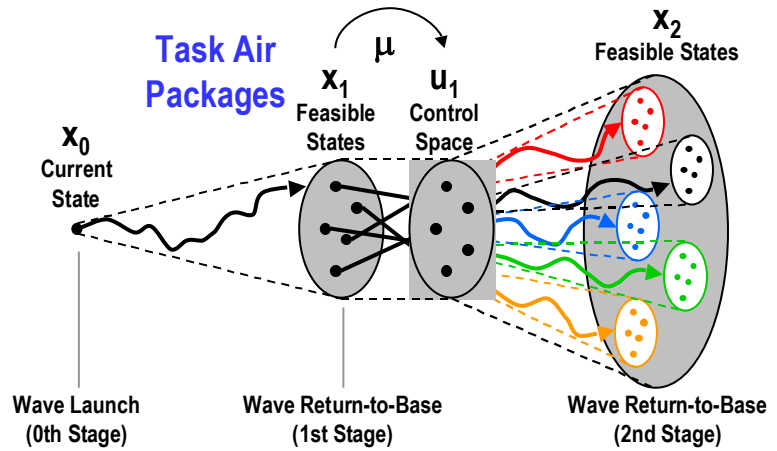


Figure 48 Battlespace State Propagation for 2-Stage/2-Wave Problem

mission queue and the enemy status at $t=0$. The battlespace state is propagated to the 1st wave return to base, and based on some approximation, the 2nd wave mission queue is modeled. Given the 2nd wave mission queue and the distribution over the enemy status, the battlespace state is propagated to the end of the 2nd wave. Based on the distribution of the battlespace at the end of the second wave, the performance metric is computed.

To perform this assessment, the 2-wave prediction models presented in Section 4.6.3, which include random sampling, certainty equivalence, and aggregated approximation, are compared to experimentally generated expected performance. Figure 49 illustrates the prediction accuracy of the different two-wave prediction models compared to empirical evaluation. For the empirical evaluation, 10,000 Monte Carlo evaluations were performed to generate the empirical performance prediction. It is seen from this figure that the

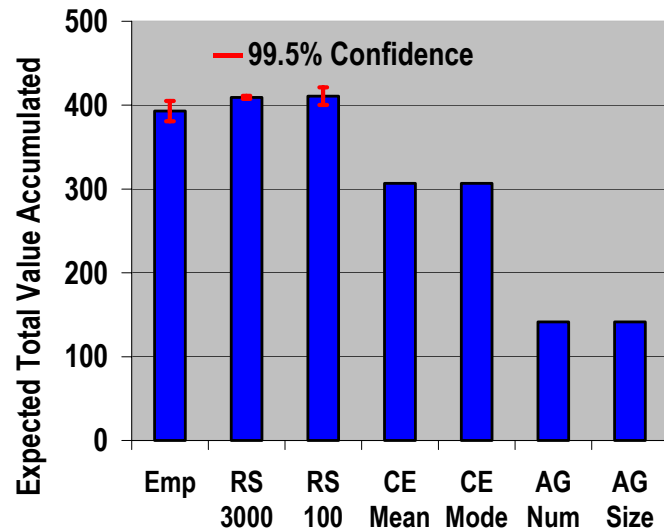


Figure 49 Prediction Accuracy of Different Design Models for the 2-Stage/2-Wave Problem

two random sampling approaches are statistically equivalent to the empirical expected performance. Note, RS # represents random sampling using # realizations of the battlespace state at the end of the first wave; for each realization, the mission controller is called to produce the second wave mission queue. It is also seen from this figure that the two certainty equivalence approaches are within 21% of the true expected performance. Note, CE Mean/Mode represents certainty equivalence using the mean/mode of the battlespace state at the end of the first wave to generate a single 2nd wave mission queue. Finally, it is seen from this figure that the aggregate approximation approaches are within 64% of the true expected performance. Note, AG Num represents the aggregation approach that reduces the number of air packages that are launched and AG Size represents the approach that reduces the size of the air packages launched. Thus, a spectrum of 2-wave prediction models with varying accuracy, randomness, and computation complexity exist. We now direct our attention to how well these models support the proactive control decision being made during the first wave mission generation.

Having presented the prediction accuracy, the question remains whether the higher fidelity prediction models improve the initial base mission control solution. To perform this assessment, initial mission queues were generated using all of the 2-wave prediction models presented above. Again, the MMR assignment algorithm was used to search the control space. To provide a comparison between proactive and reactive control techniques, mission queues were generated using the 1-stage/1-wave prediction model and the 2-stage/2-wave prediction models. In both cases, the initial status of the enemy and thus the initial state x_0 is known. Figure 50 illustrates the initial mission queues produced by the reactive control technique and a representative proactive control technique using one of the 2-stage/2-wave prediction models. It is seen from this figure that the proactive control approach chooses a low risk route for the initial wave but chooses not to attack the furthest distance targets. In contrast, the reactive control solution also chooses low-risk routes, but chooses to strike targets that require longer ingress and egress times. The net impact of further targets and low risk routes increases the 1-wave control solution expected execution time by 30% over that of the two-wave control solution. This 30% increase in expected wave execution time severely limits the target opportunities during the second wave.

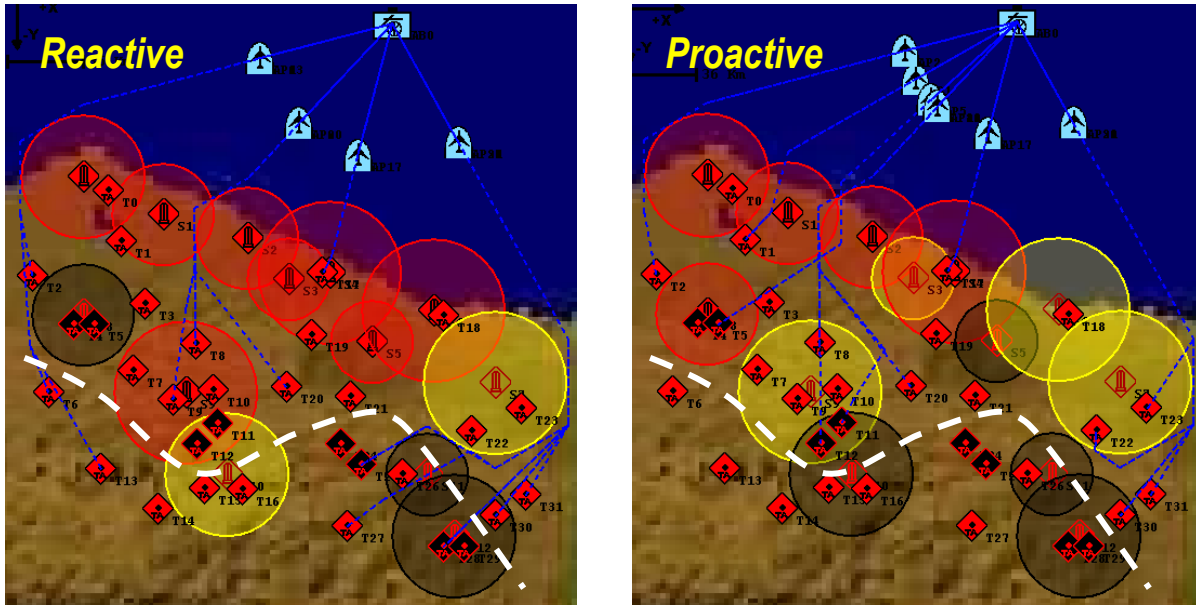


Figure 50 Behavioral Comparison of Proactive Versus Reactive Control Strategy for 2-Stage/2-Wave Problem

Having illustrated the behavioral differences between the base mission solutions, the performance numbers will now be presented. Figure 51 illustrates the empirical performance for the different mission queues generated using 1-stage and 2-stage prediction models. Note, empirical results were obtained using 10,000 Monte Carlo samples. It is seen from this figure that the performance of the 2-stage control solutions is mixed. From the prediction assessment above, it is known that the random sampling approach provides an accurate, albeit noisy, estimate of the expected performance. Thus, this prediction model serves as a baseline to highlight the

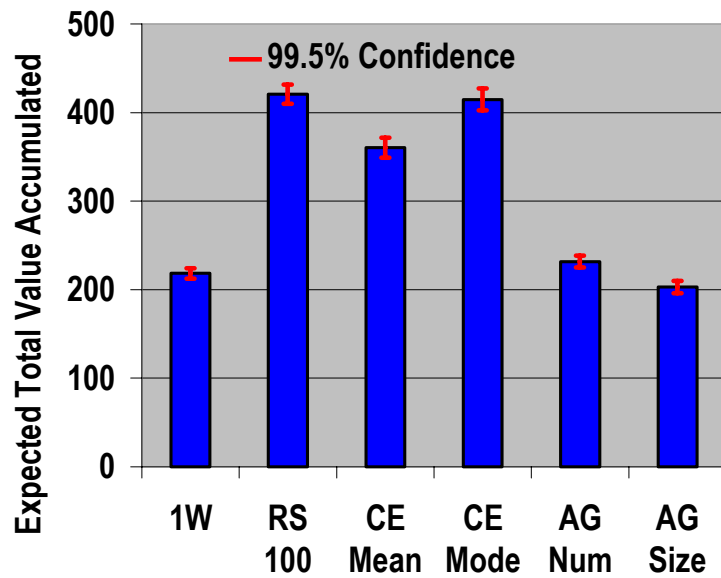


Figure 51 Control Performance for the 2-Stage/2-Wave Problem

achievable performance improvement over the 1-stage approach. Furthermore, the random sampling approach provides a baseline to compare the different 2-wave approaches. It is seen from this figure that the baseline proactive control approach provides a 93% improvement over the expected performance of the reactive control approach. Relative to the other 2-wave approaches, it is seen that the certainty equivalence using the mode is statistically equivalent to the baseline, the certainty equivalence using the mean is within 14% of the baseline, the aggregate approach reducing the number of 1st wave air packages is within 45% of the baseline, and the aggregate approach reducing the size of the 1st wave air packages is within 52% of the baseline. Thus, the certainty equivalent approaches provide superior performance over the aggregate approaches. Furthermore, it is believed that the certainty equivalence using the mode provides superior performance over the certainty equivalence using the mean because the mode amplifies the differences between the good and the bad control options.

The final piece to the performance story of the different proactive, 2-wave approaches is to view the controller complexity for this particular scenario. Figure 52 illustrates the number of one-stage prediction function calls required to produce a control solution for the different control approaches. Note, as highlighted in Section 4.6, a single 2-stage prediction model may require hundreds of one-stage prediction function calls to produce

the estimated performance for a control option. It is seen from this figure that the baseline solution requires approximately 5.0 orders of magnitude more function calls than the 1-wave control solution. Relative to the 2-wave control solution, the certainty equivalence approaches require approximately 2.0 orders of magnitude less function call than the baseline, and the aggregate approaches require approximately 4.5 order of magnitude fewer function calls.

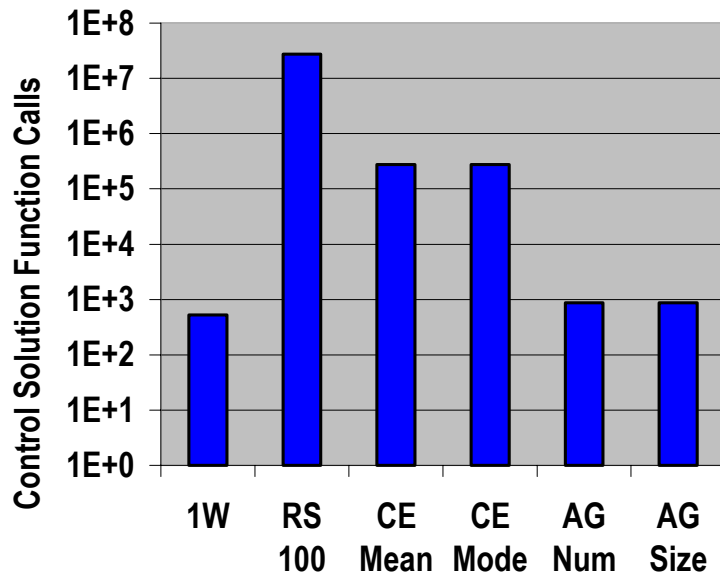


Figure 52 Controller Computational Performance for the 2-Stage/2-Wave Problem

In summary, this assessment highlighted the benefits of performing proactive versus reactive control over a 2-wave problem. As highlighted in Section 4.6.3, there are a variety of ways to approximate the loop closure that generates the 2nd wave mission queue, and we have chosen to illustrate random sampling, certainty equivalence, and aggregate approaches where the random sampling approach was used as a baseline to compare prediction accuracy and control solution quality. From this assessment, it was determined that proactive control, which anticipated the deadlines in the second wave and determined 1st wave missions to minimize risk and execution time, provides a substantial performance improvement over non-anticipatory, i.e. reactive, control strategies. Furthermore, it was shown that the certainty equivalence control approach does provide an accurate prediction of the expected 2-wave performance and does produce high-quality control solutions at a substantial reduction in computation complexity when compared to the baseline. Finally, it was shown that the aggregated approaches neither provide an accurate prediction of the 2-wave predicted performance nor provide quality control solutions.

5.5 2-STAGE/1-WAVE MODEL AOR TASKING UNDER UNCERTAINTY PROBLEM

In the previous section, the benefits of proactive versus reactive control were illustrated for a 2-wave problem with stringent terminal constraints. In this assessment, the complexity of the problem will be increased to include AOR tasking under uncertainty. Thus, this experiment will highlight the benefits of anticipating the arrival of information and closing the loop in the context of AOR tasking. Accordingly, the 2-Stage/1-Wave AOR Tasking prediction model presented in Section 4.6.6 will be used for the experiment.

To show this benefit, the standard scenario presented in Section 5.1 was modified by making all targets of the TCT type and by including AOR points and ISR collection assets. Given that ISR collection is explicitly modeled in this scenario, perfect state information about the enemy status is not assumed for this experiment during execution. Note that in previous experiments, the battlespace state at loop closures was always perfectly observed, and uncertain state information was only captured within the prediction model for future loop closures. Figure 53

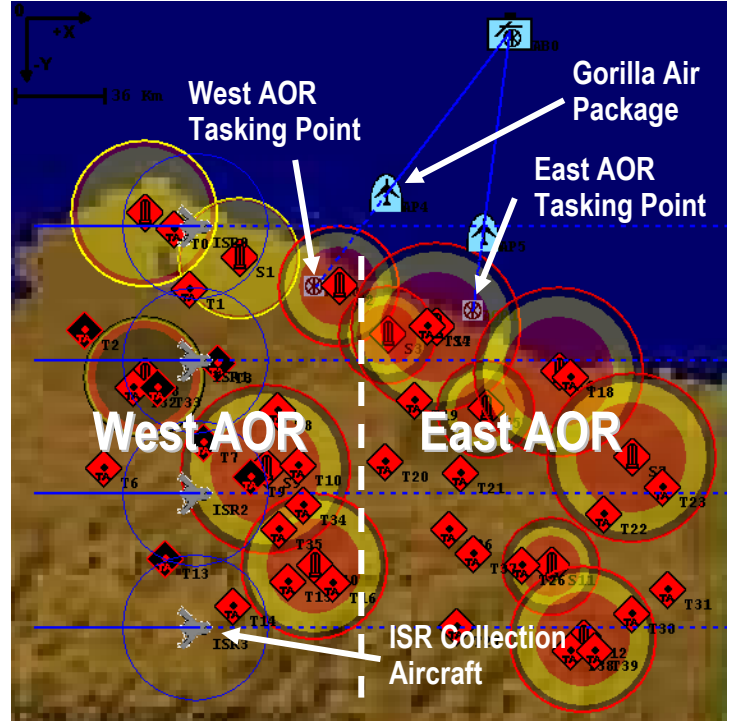


Figure 53 Modified Demonstration Scenario Used for AOR Tasking Under Uncertainty Problem

illustrates the modified scenario used for this experiment. It is seen in this figure that the scenario now includes two AORs and ISR assets that have *a priori* defined missions to fly from left to right. Since we only have perfect state information when an ISR asset is within range of an enemy asset, our knowledge of the enemy state is represented by a distribution at any given time. This knowledge is represented by the rings in the SAM's lethal range where the different colors represent different modes. Thus, the area of the ring represents the probability of being in a particular state. If an ISR asset is within range of the SAM, the color of the SAM's range is solid to reflect perfect observation. Once the ISR asset is out of range of the SAM site, the status information begins to decay according to a transient Markov process, and eventually achieves a steady state distribution. Figure 54 illustrates this transient behavior.

Having no information about the enemy status at $t=0$, it is reasonable to assume that all assets are in a steady state condition. For this experiment, the steady state distribution for the targets is $P_{known}=0.5$, $P_{unknown}=0.5$, and $P_{dead}=0$. For the SAMs, the steady state distribution is such that $P_{active}=0.4$, $P_{inactive}=0.35$, $P_{unknown}=0.25$, $P_{repair}=0$, and $P_{dead}=0$.

The execution sequence for this scenario is as follows:

- Based on uncertain initial information, it is assumed that enemy status at $t=0$, i.e. x_0 , is governed by steady state distributions.
- Given this information, the controller composes and tasks gorilla air packages to the AORs.
- The gorilla air packages launch at $t=0$ in ingress to their respective AORs using straight line routes.
- During ingress, the ISR assets along with the gorilla air packages collect observations about the enemy status.
- Upon each gorilla air package's arrival to its AOR, a controller decomposes it into smaller air packages and assigns them to particular target locations based on the information collection.
- Air packages ingress via straight line routes to their respective target locations, deploy munition, and egress back to base.
- Wave ends when all air packages return to base.

Given these modifications to the base scenario and the execution sequence above, the proactive base mission controller must anticipate the ISR collection, information degradation, and the AOR loop closure mapping in order to make optimal gorilla package composition and tasking. Note that for all experimental results presented in this section, the base mission queues, i.e. gorilla air package composition and tasking, are generated using the Surrogate Method discussed in Section 4.5.6. Likewise, all AOR taskings, i.e. decomposition of gorilla air package and air package composition and tasking, are generated using the Maximum Marginal Return assignment algorithm discussed in Section 4.5.3.

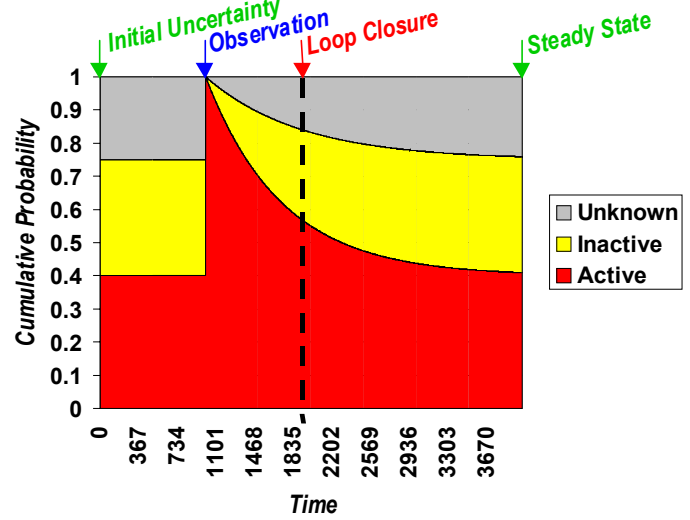


Figure 54 Markov Transient Response of Known SAM Site Status Distribution

As noted in Section 4.6.6, the difficulty of a 2-stage/1-wave AOR tasking under uncertainty prediction model is modeling information arrival and degradation and the loop closure upon arrival to the AOR. This is a difficult problem because there is a combinatorial large number of possible states upon arrival to the AOR due to gorilla package attrition and information arrival and degradation. As a further complication, all of the assignment algorithms outlined in Section 4.3 produce a discrete mission queue for a discrete number of resources; thus, there are no control algorithms that map a surviving aircraft distribution to a mission queue distribution. Given these complexities, a variety of AOR tasking prediction models will be assessed in this section. As in the previous sections, an assessment of the prediction quality and control performance will be made for each of these approaches.

To perform the prediction accuracy assessment, the 1-wave AOR tasking problem was set up where the initial gorilla mission queue was defined *a priori*. Thus, the only loop closures occur when the gorilla air packages arrive at the AORs; upon arrival, the AOR tasking controller

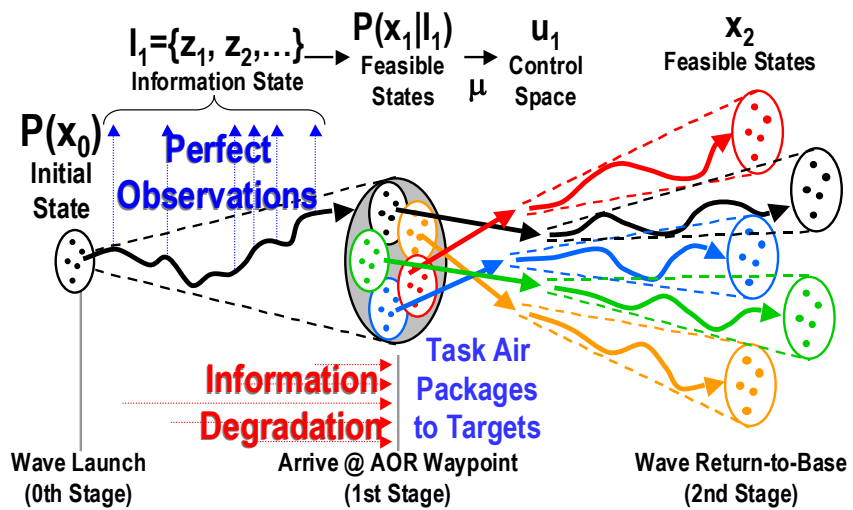


Figure 55 Battlespace State Propagation for 2-Stage/1-Wave AOR Tasking Problem with ISR Collection and Degradation

determines the low level air package composition and tasking to target locations. Figure 55 illustrates the 2-stage/1-wave AOR tasking under uncertainty prediction problem where the initial state includes the initial mission queue and the enemy status at $t=0$. The battlespace state is propagated to the AOR rendezvous point, and based on some approximation, the AOR tasking is modeled. Given the AOR tasking and the distribution over the enemy status, the battlespace state is propagated to the end of the 1st wave. Based on the distribution of the battlespace at the end of the second wave, the performance metric is computed.

To perform this assessment, the 2-stage AOR prediction models presented in Section 4.6.6, which include random sampling, certainty equivalence, and partial OLF, are compared to experimentally generated expected performance. As an additional comparison, the 1-stage prediction model that does not anticipate future information arrival and control decisions is included. Figure 56 illustrates the prediction accuracy of the different two-wave, AOR prediction models compared to empirical evaluation.

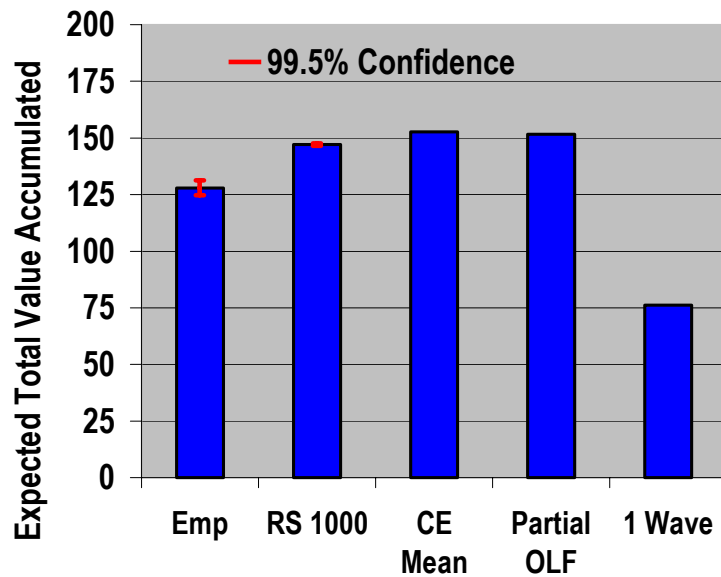


Figure 56 Prediction Accuracy of Different Design Models for the 2-Stage/1-Wave AOR Tasking Under Uncertainty Problem

For the empirical evaluation, 23,500 Monte Carlo evaluations were performed to generate the empirical performance prediction. It is seen from this figure that the 2-stage AOR prediction algorithms are consistently optimistic by approximately 15%. As discussed in the previous section, the random sampling approach should provide an accurate, albeit noisy, estimate of the expected performance. However, as seen in the above figure, the random sampling approach is producing an optimistic estimate of the expected performance. From a thorough evaluation of prediction model, it was determined that there is a model mismatch between the prediction model and the simulator. In the simulator, information degradation begins when the ISR collection asset flies out of range of the enemy asset; however, in the prediction model, information degradation begins immediately after detection, i.e. when the asset first comes into range. Thus, the prediction model is overestimating the transient time for information degradation. Finally, it is seen in the above figure that the 1-stage prediction model, which does not account for information arrival and subsequent AOR loop closures, is only within 40% of the true expected performance.

Having presented the prediction accuracy, the question remains whether the higher fidelity prediction models improve the initial gorilla air package assignment. To perform this

assessment, initial mission queues were generated using all of the 2-stage AOR prediction models presented above. To provide a comparison between proactive and reactive control techniques, mission queues were generated using both the 1-stage/1-wave and 2-stage/1-wave prediction models. Figure 57 illustrates the initial mission queues produced by the proactive and reactive control techniques. It is seen from this figure that the proactive control approach chooses to send all assets to the AOR for which information becomes available. This solution is attained by all the 2-stage controllers, i.e random sampling, certainty equivalent, and partial OLF. On the other hand, the 1-stage controller, not able to recognize the arrival of information, let alone its benefit, selects a uniform allocation of resources to AORs. These results are consistent with our expectations.

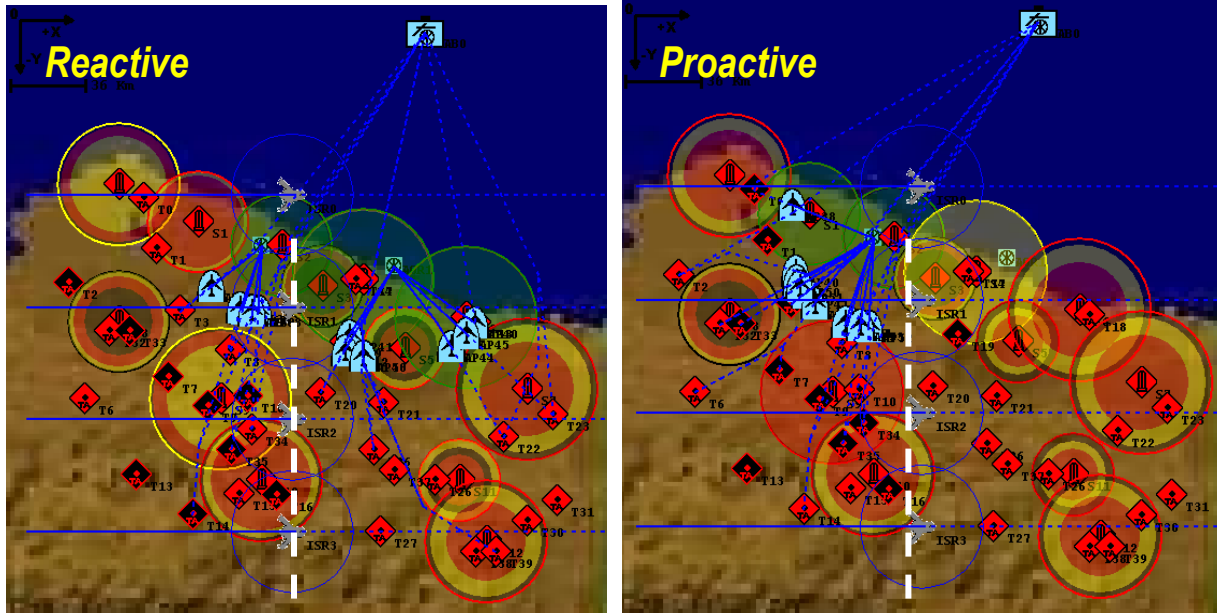


Figure 57 Behavioral Comparison of Proactive Versus Reactive Control Strategy for 2-Stage/1-Wave AOR Tasking Under Uncertainty Problem

Having illustrated the behavioral differences between the base mission solutions, the performance numbers will now be presented. Figure 58 illustrates the empirical performance for the different mission queues generated using 1-stage and 2-stage prediction models. Note, empirical results were obtained using 20,400 Monte Carlo samples. It is seen from this figure that the performance of the 2-stage control solutions are all identical, which is not surprising since their mission queues were identical. In comparison to the reactive control approach, the proactive controllers produced a 106% improvement in expected performance.

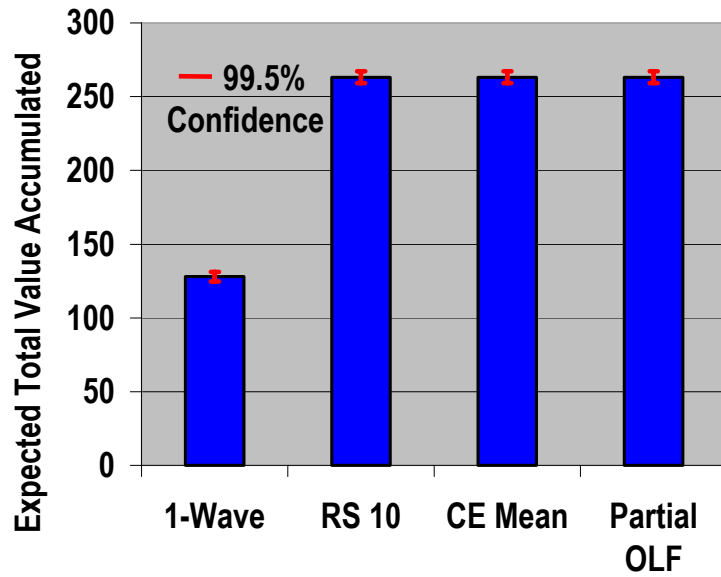


Figure 58 Control Performance for the 2-Stage/1-Wave AOR Tasking Under Uncertainty Problem

The final piece to the performance story of the different proactive, 2-stage approaches is to view the controller complexity for this particular scenario. Figure 59 illustrates the number of one-stage prediction function calls required to produce a control solution for the different control approaches. Note, as highlighted in Section 4.6, a 2-stage prediction model may require hundreds of one-stage prediction function calls to produce the estimated performance for a

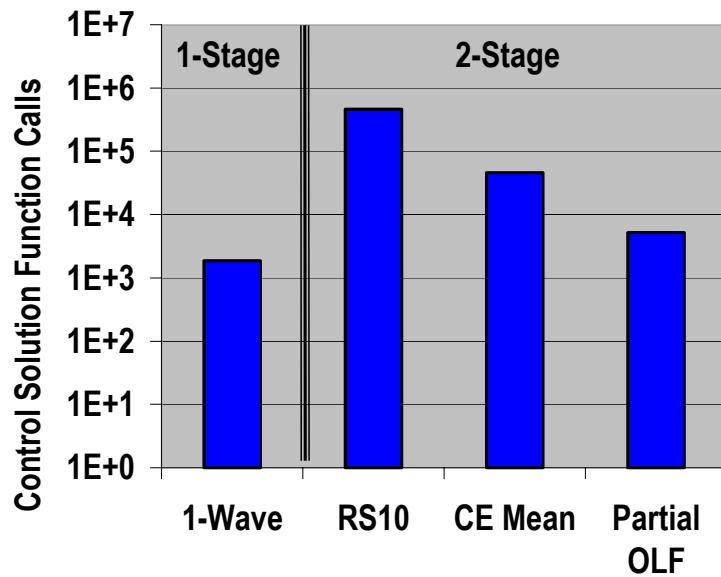


Figure 59 Controller Computational Performance for the 2-Stage/1-Wave AOR Tasking Under Uncertainty Problem

control option. It is seen from this figure that the baseline solution requires approximately 2.5

orders of magnitude more function calls than the 1-wave control solution. Relative to the 2-wave control solution, the certainty equivalence approaches require approximately 1.0 orders of magnitude less function call than the baseline, and the aggregate approaches require approximately 2.0 order of magnitude fewer function calls.

In summary, this assessment highlighted the benefits of performing proactive versus reactive control for a 1-wave AOR tasking under uncertainty. It was shown that control strategies that anticipate future information collection, degradation, and loop closures can provide a substantial performance improvement over control strategies that only react to future information.

6 CONCLUSIONS

This research, performed by ALPHATECH, focused on providing military commanders with the ability to perform real-time dynamic control of military air operations using near optimal mission replanning for a 24-hour segment of a JAO campaign using control algorithms that anticipate possible mission modifications due to uncertain future events. The near real-time, near optimal control decisions being produced consist of the generation/modification of mission definitions for both assets at base and airborne with the performance goal of achieving the specified JFACC objective while minimizing the friendly asset losses. The mission definition includes assignment of resources to targets, high-level routing (by specification of waypoints), strike package composition, weapon composition, and desired time-on-target. The primary benefit of this technology is agile and stable control of distributed and dynamic military operations conducted in inherently uncertain, hostile, and rapidly changing environments.

The JAO problem size investigated includes approximately 100 targets/threats and a mixture of 50 airborne assets taken from two generic airborne asset types: strike and weasel aircraft. Strike aircraft attack targets and weasel aircraft strike surface-to-air threats. Risk to the air packages is introduced via threats such as were surface to air missiles. The size of operation we chose for our study assumes that some form of geographic decomposition of the battle space has been specified, and that we are concerned primarily with achieving the specified missions for a 24 hour period. Hence, there is implicit value in saving assets for future operations beyond the specified horizon.

For the above JAO problem, there are many interesting dynamics that make it challenging. The scarcity of aircraft resources forces multiple “turns” of the aircraft in order to service all of the targets. There are also multiple sources of uncertainty in the problem. There is uncertainty in the status and location of enemy assets. Targets may be known, unknown, hiding, emerging, time critical, or destroyed. Threats may be active, inactive, unknown, repairing, or destroyed. There is also uncertainty in the interaction between enemy and friendly assets that depends on the characteristics of the target/threat and the relative position and composition of the air package, i.e. number and position of strike and weasel.

Given this highly uncertain and rapidly changing environment, the JAO control problem can be viewed as a dynamic decision problem under uncertainty. This class of problems can be formulated as a Markov decision problem. Exact techniques for control design using this

approach, such as Stochastic Dynamic Programming (SDP), are computationally expensive, and do not scale up to the size of the JAO problem of interest. A subtle but significant attribute of the Markov decision problem formulation is that it produces control strategies that anticipate the effects of future contingencies, and evaluates the possible actions over all possible future states, by modeling the future information arrival and control decisions. It is this fact that produces *proactive* versus *reactive* control behaviors. This proactive attribute is desirable for stable and agile control of the JAO enterprise because future information arrival and control opportunities are dependent on stringent spatial, temporal, and coordination constraints.

Given the strengths and weaknesses of the SDP algorithm, this research focused on developing Approximate Dynamics Programming (ADP) strategies that provide the desirable *proactive* control behaviors but with near real-time computation effort. The control design technology is based on combining hybrid state modeling techniques for developing statistical dynamical models relating mission decisions to evolution of objects in the battlespace, together with ADP control design techniques that have demonstrated real-time, proactive performance for other relevant military problems. Accordingly, a spectrum of ADP control techniques were developed for the JAO problem; these techniques were developed in discrete event simulations of JAO scenarios. The major accomplishments of the research were:

- Translated the JAO Control Enterprise into a Dynamical Hybrid State, Discrete Event, Stochastic Decision Making Problem
- Integrated Emerging ADP Technologies into JAO Feedback Controllers
- Experimentally Demonstrated the Benefits of Feedback Control
- Experimentally Demonstrated Benefits of Approximate Optimal Control
- Developed Innovative Hybrid, Multi-Rate Control Architecture
- Developed Computationally Efficient Control Algorithms that Produce Operationally Consistent Behaviors
- Extended Control Algorithms to Accommodate Hierarchical Mission Tasking and ISR Information Collection

In summary, our investigations demonstrated the feasibility of automating military operations planning to provide real-time, near-optimal control strategies that achieve operational objectives while minimizing asset losses. By adopting a hybrid, multi-rate control architecture, we were able to tailor the application of control at a time scale appropriate to the operational situation at hand. Within the proposed multi-rate architecture, we developed a spectrum of ADP

control strategies that produce a range of control decisions, ranging from immediate retasking or abort decisions to preplanned multiple wave tasking. The solution quality and computation performance of these algorithms was tested and verified in a JAO discrete event simulator. Our experiments show that the ADP strategies were able to produce operationally consistent, proactive control strategies that anticipated likely contingencies and positioned assets for opportunities of recourse all in either real-time or near real-time. Furthermore, a scalability assessment indicated that many of the ADP controllers could provide near real-time performance for scenarios with 250 targets with some modest parallel computation.

The results of this investigation can be extended in several important directions. First, the algorithms developed in this investigation can be extended to include further modeling details of a JAO environment, such as detailed weaponeering, additional platform types and missions. In this manner, the algorithms could then form the basis for a decision aid for an Air Operations Center (AOC). This decision aid would assist operators in rapidly replanning missions in the presence of contingencies, and help to generate robust Air Tasking Orders (ATO). Second, the technology developed in this work can be extended to design robust autonomous controllers for automated vehicles conducting uncertain missions.

7 REFERENCES

- [B96] Bertsekas, D. P., Dynamic Programming and Optimal Control, Vols. I-II, Athena Scientific, Belmont, MA 1995.
- [B99] Bertsekas, D.P., "Rollout Algorithms: An Overview," Proceedings of the 38th Conference on Decision & Control, Phoenix, Arizona, December, 1999.
- [BT96] Bertsekas, and Tsitsiklis, J. N, Neuro-Dynamic Programming, Athena Scientific, Belmont, MA 1996.
- [BTW97] Bertsekas D.P., Tsitsiklis, J.N., and Wu, C., "Rollout Algorithms for Combinatorial Optimization," Journal of Heuristics, Vol. 3, Num. 3, November, 1997, pp.245-262
- [SB98] Sutton, R., and Barto, A. G., Reinforcement Learning, MIT Press, Cambridge, MA, 1998.
- [BC98] Bertsekas, D. P., and Castañon, D. A., "Rollout Algorithms for Stochastic Scheduling Problems," *Journal of Heuristics*, Vol. 5, 1999.
- [CaPa99] Cassandras, C. G. and C. Panayiotou, "Concurrent Sample Path Estimation for Discrete Event Systems," to appear in Journal of Discrete Event Dynamic Systems, 1999.
- [CSSA89] D. A. Castañon, N. Sandell, W. Stonestreet and M. Athans, Advanced Weapon Target Assignment Algorithms Program: Final Report, ALPHATECH TR-440 on Army Strategic Defense Command Contract DASG60-86-C-0050, July 1989.
- [Dai97] Dai, L. and Chen, C. H., "Rates of Convergence of Ordinal Comparison for Dependent Discrete Event Dynamical Systems," Journal on Optimization Theory and Applications, V. 94, 1997.
- [Vak91] Vakili, P., "A Standard Clock Technique for Efficient Simulation," Operations Research Letters, Vol. 10, pp. 445-452, 1991.
- [Glas91] Glasserman, P., Gradient Estimation Via Perturbation Analysis. Boston, MA: Kluwer, 1991.
- [HoCao91] Ho, Y.C., and X.R. Cao, Perturbation Analysis of Discrete Event Dynamic Systems. Kluwer Publishing, 1991.
- [Cass93] Cassandras, C.G., Discrete Event Systems: Modeling and Performance Analysis. Boston, Irwin, Inc., 1993.
- [CassLaf99] Cassandras, C.G., and S. Lafortune, *Introduction to Discrete Event Systems*, Kluwer Academic Publisher, 1999.
- [HoCass97] Ho, Y.C., and Cassandras, C.G., "Perturbation Analysis for Control and Optimization of Queueing Systems: An Overview and the State of the Art", in "Frontiers in Queueing" (J. Dshalalow, Ed.), CRC Press, pp. 395-420, 1997.
- [HoCass99] Ho, Y.C., Cassandras, C.G., Chen, C.H., Dai, L., "Ordinal Optimization and Simulation," submitted to special issue on simulation to be published by INFORMS 1999.
- [Ro83] Ross, S. M., *Introduction to Stochastic Dynamic Programming*, Academic Press, N.Y., 1983.
- [BC99] Bertsekas, D.P., Castañon, D. A. and et al, "Adaptive Multi-platform Scheduling in a Risky Environment," DARPA-JFACC Advances in Enterprise Control (AEC) Symposium, 1999.

- [BC00] Bertsekas, D. P., Castañon, D. A. and et al, "Dynamic Programming Methods for Adaptive Multi-Platform Scheduling in a Risky Environment," DARPA-JFACC Advances in Enterprise Control (AEC) Symposium, 2000.
- [GHZ99] Gong, W., Ho, Y., and Zhai W., "Stochastic Comparison Algorithm for Discrete Optimization with Estimation," *SIAM Journal of Optimization*, Vol. 10, No. 2, pp. 384-404, 1999.
- [GoCass00] Gokbayrak, K., and Cassandras, C.C., "An On-Line 'Surrogate Problem' Methodology for Discrete Stochastic Resource Allocation Problems, " to appear in *Journal of Optimization Theory and Applications*, 2000.
- [Bran95] Branicky, M.S., "Studies in Hybrid Systems: Modeling, Analysis, and Control," PhD Dissertation, Laboratory of Information and Decision Systems, LIDS-TH-2304, Massachusetts Institute of Technology, June 1995.
- [Patek99] Patek, S. D., D. A. Logan and D. A. Castañon, "Approximate Dynamic Programming for the Solution of Multiplatform Path Planning Problems," *Proceedings IEEE SMC '99 Conference*, Richmond, VA, 11/1999.

8 APPENDICES

As noted in previous sections, some of the results that were documented in technical memorandums and conference proceedings are being included to complement the body of this report.

8.1 TYPES OF CONTROL

See attached

Wohletz, J.M., "Optimal Control Solutions for Stochastic Systems," ALPHATECH TM-572, Burlington, MA, 2000.

8.2 AEC 2000

See attached

Bertsekas, D.P., D.A. Castañon, et.al., "Dynamic Programming Methods for Adaptive Multi-Platform Scheduling in a Risky Environment," *DARPA AEC Symposium*, , Minneapolis, MN, July 10-11, 2000.

8.3 AEC 1999

See attached

Bertsekas, D.P., Castañon, D. A. and et al, "Adaptive Multi-platform Scheduling in a Risky Environment," DARPA-JFACC Advances in Enterprise Control (AEC) Symposium, 1999.

8.4 ACC 2001

See attached

Wohletz, J.M., D.A. Castañon, and M.L. Curry, "Closed-Loop Control for Joint Air Operations," *IEEE American Control Conference*, ACC01-INV3501, Arlington, VA, 2001.

8.5 SPIE 2001

See attached

Cassandras, C.G., K. Gokbayrak, D. Castañon, J. Wohletz, M. Curry, and M. Gates, "Modeling and Agile Control for a Joint Air Operation Environment," *Proceedings of SPIE 15th Annual Intl. Symposium*, April 2001.