# A STOCHASTIC AGENT APPROACH (SAA) FOR MISSION EFFECTIVENESS

A Dissertation
Presented to
The Academic Faculty

By

Seth Edward Gordon

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology

August 2018

Copyright © Seth Edward Gordon 2018

**A STOCHASTIC AGENT APPROACH (SAA) FOR MISSION EFFECTIVENESS**

Approved by:

Dr. Dimitri Mavris, Advisor
School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Daniel Schrage
School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Mariel Borowitz
Sam Nunn School of International Affairs
*Georgia Institute of Technology*

Dr. J. Charles Domercant
Electronic Systems Laboratory
*Georgia Tech Research Institute*

Dr. Michael Steffens
School of Aerospace Engineering
*Georgia Institute of Technology*

Date Approved: April 10, 2018

Do or do not, there is no try.

*Jedi Master Yoda*

To my wife, Kirsten, I could not have done it without your support.

## ACKNOWLEDGEMENTS

This thesis document has been a labor of love for the past two years. However, none of this would have been possible without the dedication and support of my friends, family, colleagues, advisors, and mentors. Foremost among these is my wife, Kirsten, who has been there through my entire graduate career and kept pushing me forward. In close second are my advisor, Professor Dimitri Mavris, and supervising Research Engineers, Dr. J. Charles Domercant and Dr. Michael Steffens. Without their guidance, technical advice, and supervision, I would not have been able to assemble the sources, knowledge base, or other aspects necessary to generate a Doctoral Thesis topic or this dissertation. Thank you all.

**TABLE OF CONTENTS**

## LIST OF TABLES

## LIST OF FIGURES

## ACRONYMS

**6DOF** Six-Degree-of-Freedom. 78, 97, 105

**ABM** Agent-Based Model. 15, 16, 52, 69, 77, 79, 92, 95, 98, 108–112, 114, 115, 119–121, 123, 148, 169, 170, 174, 176, 178–180, 184, 186, 196, 199, 200, 209, 210, 239, 240, 253, 261, 262, 267, 268, 270, 271, 273, 274, 277, 279–281, 283

**ABV** Agent-Based Variable. xx, xxi, xxvi, xxvii, 16, 17, 120–122, 125, 126, 128, 139, 143–145, 148, 156–158, 162, 164–169, 172–174, 176, 180–198, 200, 202–204, 208, 209, 211–214, 216, 222, 223, 227, 228, 231–233, 240–242, 247, 256–260, 267, 268, 270, 271, 273–277, 279, 280, 282, 283

**ACM** Aggregate Combat Model. 11, 85–90, 92–94, 97, 100

**AFSIM** Advanced Framework for Simulation, Integration and Modeling. 54, 92

**AI** Artificial Intelligence. xxvii, 16, 48, 93, 114, 115, 117, 119, 187, 238, 239

**ANN** Artificial Neural Network. 237–240, 245, 247–249, 252, 258, 261, 271, 281

**AoA** Analysis of Alternatives. 20, 40–45

**AR** Aspect Ratio. 107, 230

**ASDL** Aerospace Systems Design Laboratory. 216

**ATLAS** A Tactical, Logistical, and Air Simulation model. 88, 89

**BRAWLER** Engagement Level Air-to-Air Model. 92

**C4ISR** Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance. 29

**CBA** Capability-Based Assessment. 8, 20, 29, 39, 44, 46

**CEM** Combat Evaluation Model. 89

**CONOPS** Concept of Operations. 100

**COTS** Commercial Off-the-Shelf. 21, 44

**CoV** Coefficient of Variance. 220–222, 229

**DAU** Defense Acquisition University. 291

**DoD** United States Department of Defense. xv, 1, 8, 29, 38, 39, 41–43, 286, 288, 291

**DoE** Design of Experiments. 15, 17, 49, 145, 199, 208, 213, 223–229, 261, 274, 276, 281

**DOTmLPF-P** Doctrine, Organization, Training, Materiel, Leadership, Personnel, Facilities, and Policy. 36, 37, 286

**EBCM** Entity-Based Combat Model. 77, 85, 90–94, 97, 100, 108, 170, 233, 267, 272, 273

**EFA** Exploratory Factor Analysis. 231

**EoA** Evaluation of Alternatives. 20, 40–42, 45

**FCC** Flight Control Computer. 179

**FCS** United States Army Future Combat System. 39

**FLOPS** Flight Optimization System. 51, 91, 95

**FPS** First-Person Shooter. 115

**GA** Genetic Algorithm. 136–138, 152, 154, 157

**GAO** Government Accountability Office. xv, 39, 40, 42–44

**GOTS** Government Off-the-Shelf. 21, 44

**GPS** Global Positioning System. 43

**GUI** Graphical User Interface. 247

**HFD** Hierarchical Functional Decomposition. 8

**IDAGAM** Institute for Defense Analysis Gaming Model. 89

**IPPD** Integrated Product and Process Development. 20, 32

**JCIDS** Joint Capabilities Integration and Development System. 8, 20, 36, 38, 39, 41, 42, 44, 280, 286

**JRC** Joint Requirements Council. 29

**KPP** Key Performance Parameters. 4, 291

**M&S** Modeling and Simulation. 12–14, 16, 47, 68, 69, 77, 78, 113, 232, 235, 270, 272

**MANA** Map Aware Non-Uniform Automata. 92

**MBSE** Model-Based Systems Engineering. 8

**MOE** Measure of Effectiveness. xxvii, 13, 107, 108, 120, 199, 200, 207, 208, 215–218, 221, 222, 227, 231, 243–245, 247, 250, 252, 253, 255–259, 270, 277, 279, 281, 291

**MOP** Measure of Performance. 291

**MORS** The Military Operations Research Society. 10

**MOS** Measure of Suitability. 291

**NASA** National Aeronautics and Space Administration. 51, 95

**OEC** Overall Evaluation Criterion. 125, 126, 136, 157, 167, 189

**OneSAF** One Semi-Automated Forces. 92

**P**$_{detect}$ Probability of Detection. 98, 114, 120

**P**$_{kill}$ Probability of Kill. 60, 62, 79, 81, 98, 114, 120, 192, 273

**QE** Quality Engineering. 32

**QFD** Quality Function Deployment. 9

**Re** Reynolds Number. 102

**RFP** Request for Proposal. 21, 35, 36

**RGS** Requirements Generation System. 35, 36, 38

**RIMPAC** Rim of the Pacific Exercise. 51, 65, 67, 74

**RTS** Real Time Strategy. 117, 119

**SAA** Stochastic Agent Approach. xxiii, xxvi, xxvii, 2, 12, 15, 16, 18, 35, 47, 49, 50, 56, 57, 77, 80, 85, 97, 113, 120, 170, 174, 183, 186, 196, 197, 209, 222, 227–229, 234, 235, 237, 253, 260, 266, 267, 269–274, 276–284

**SBD** Small Diameter Bomb. 21, 67

**SE** Systems Engineering. 32

**SME** Subject Matter Expert. 1, 25, 47, 75, 76, 231, 272

**SoS** System of Systems. 9, 10, 46, 61, 73, 79, 94, 100, 287, 289

**TSFC** Thrust Specific Fuel Consumption. 34, 102

**TTP** Tactics, Techniques, and Procedures. 37

**USAF** United States Air Force. 45, 67

**USMC** United States Marine Corps. 45

**USN** United States Navy. 45, 61

**UTE** Unified Tradeoff Environment. 13, 28, 33–35

**VECTOR-2** Vector Research Corporation Aggregate Combat Model 2. 89

**WSARA** Weapon Systems Acquisition Reform Act. 40

**WVR** Within Visual Range. 41

## SUMMARY

Former United States Deputy Secretary of Defense Robert Work advocated for more wargaming-style analysis during his tenure due to the changing geopolitical and technological environment. The demand for higher-level analyses is driven by a burgeoning focus on capability-based methods. The requisite strategic and life-cycle simulations are informed by lower fidelity approximations of combat and mission performance. These estimations are usually derived from a mix of historical data, subject-matter expertise, and full-scale exercises.

However, an analysis capability gap exists when evaluating new assets or the use of systems in an off-design scenario. More specifically, the data to generate the empirical relationships is simply unavailable when the systems have yet to be developed or evaluated. Some assumptions can be made to map known relationships and capabilities between similar systems, but there is no guarantee that such assumptions are valid. This lack of data necessitates analysis at smaller scope but higher fidelity levels to inform new representations of mission-level activities. To generate the requisite data, there is a need to rework how combat mission models are developed and utilized to account for both engineering and decision-making effects.

The **Stochastic Agent Approach (SAA)** was developed to generate new approximations of mission-level performance and capabilities. This first required the creation of a large dataset of mission plans and options to understand the mission solution space. However, a full-resolution variable space, consisting of tracking every asset at every time-step, is too complex to explore in a reasonable amount of time. Thus the main contribution of this research is the simplification of the Mission Design Space using Agent-Based Variables (ABVs).

In this approach, rather than a detailed discretization of the actions (or decision-making logic) undertaken by every simulated asset at every time-step, the variable space

becomes a series of decision-affecting coefficients. Unlike game-focused Artificial Intelligence methods, there is no evaluation, variation, or attempt to optimize the underlying decision options or use deterministic logic. Instead there is probability, represented as a continuous metric between 0.00 and 1.00, of an agent pursuing one action versus another given the same stimuli.

This methodology is applicable to a variety of scenarios and situations. The procedure for generating and leveraging Agent-Based Variables is simulation method- and tool-agnostic and adaptable to most potential scenarios. Furthermore, these variables can often be held in common regardless of the type of simulated asset, assuming appropriate logic is performed. This prevents a variable space explosion if different asset types are considered.

This research consists of three broad stages divided into subsidiary research questions and conclusions. The first objective was to evaluate means of mission simulation to select the most appropriate archetype for data generation. The second objective was to address the mission design space scalability problem with Agent-Based Variables. The final objective was to sample across the new variable space and generate empirical relationships between design variables and Measures of Effectiveness

In summary, the **SAA** illustrates how to craft Agent-Based Variables to achieve a degree of Mission Space Exploration and capability approximation not currently feasible. This enables a superior ability to understand, adapt to, and conduct trades within the Mission Design Space through this inclusion of decision-making affecting parameters.

# CHAPTER 1

# INTRODUCTION

> Military planners and theorists intuitively understood that all these new technologies, systems, and advances would drive new ways of fighting, but they were forced to envision what future battlefields would look like with few clues to go by.
>
> - Robert Work & Gen. Paul Selva [1]

Former Deputy Secretary of Defense Robert Work advocated for additional war games and mission simulation during his tenure in the DoD [1, 2]. He believed superior analyses are necessary not only due to a mix of a changing geopolitical (and thus warfighting) environment but also to ensure that the appropriate procurement, training, and preparation decisions were being made. In his view, the quality of the proposed war games were most at risk from two factors: bias on the part of the simulation creators and an overriding avoidance of failure [1].

The human-centric focus of missions and campaign-level exercises makes them more at-risk of bias than a physics-based model simply because the "rules" are written by Subject Matter Experts (SMEs) or even potential participants. Those biases can prevent impartial understanding of the solution space or predispose certain outcomes and plans. On the topic of "failure," Colonel Work notes that war games should allow participants "the opportunity to make critical mistakes and learn from them — and to perhaps reveal breakthrough strategies and tactics when doing so" [1]. For conducting solution trade-offs, knowledge of success is frequently as useful as knowledge of failure. Consequently, the mix of war game bias and failure avoidance can prevent full-scale understanding of what non-engineering solutions may exist.

Generally, war games and higher-level mission simulations can be improved through

minimizing the number of parameters and simulation "rules" which are rigid or predefined. That desire to enable additional tradeoffs in the mission space is what has inspired the development of the **Stochastic Agent Approach (SAA)** defined in this research.

## 1.1 Historical Problem Context

The **Stochastic Agent Approach** is a process to generate new understanding by incorporating previously defaulted parameters into the solution generation variable space. However, the need for a new approach is best contextualized by understanding the risks inherent in poorly informed and conducted war games and high-level simulations. A historical and well-documented example comes from the Pacific Theater of World War II.

### 1.1.1 The Battle of Midway

The Battle of Midway in July 1942 marked the turning point of World War II in the Pacific Theater. That battle saw thousands of military personnel from the United States and Imperial Japan clash in the skies and seas around the namesake Midway Atoll [3]. While the battle was an astounding strategic victory for the United States, that success was at least partially enabled by two critical planning failures by the Empire of Japan. However, to understand the battle and the preparation that went into it, there is a need to differentiate between "strategy" and "tactics."

Strategic approaches to missions focus on the overall architecture of the solution as well as deployments, maintenance, and infrastructure [4, 5]. Generals and other high-ranking decision-makers operate on the strategic level to avoid micromanaging the units under their command and, for example, would be concerned about the general location of a fleet rather than it's precise coordinates. Meanwhile, tactical concerns are more limited in both scope and scale, whether due to a smaller considered geographic

area or number of involved units [6, 7]. Tactical decisions are made by warfighters or lower-level commanders and can include the maneuvers of individual units or personnel. However, both levels are necessary for an effective fighting force as localized (tactical) success does not guarantee strategic victory and vice versa.

### 1.1.2 Simulations at the Strategic Level

With regard to the Battle of Midway, the first planning failure centered on Imperial Japan's campaign and strategic-level simulation and understanding. In the Pacific, a focus on the aircraft carrier and aviation in general revolutionized military strategic and operational concerns. To address the changes, both sides conducted expansive war games to evaluate their own commanders and tactics. While these simulations (both tabletop and full-scale exercises) were ultimately low-tech, they did result in an emergent understandings of the value of aviation [3].

However, it was a misconception when conducting these war games that handicapped the Imperial Japanese forces during the Battle of Midway. In fact, the American actions at the battle (see Figure 1.1) were predicted by junior members of the Japanese command staff during wargaming exercises in the months prior. Unfortunately for the Japanese, those strategies were dismissed as outlandish by their superiors and subsequently ignored [8]. This blindness at the strategic level was driven because of improper military simulation, or rather the dismissal of some simulation cases despite the results' validity. The campaign and strategic-level simulation, a tabletop wargaming exercise in this case, was completed improperly and thereby failed to properly predict American actions during the battle.

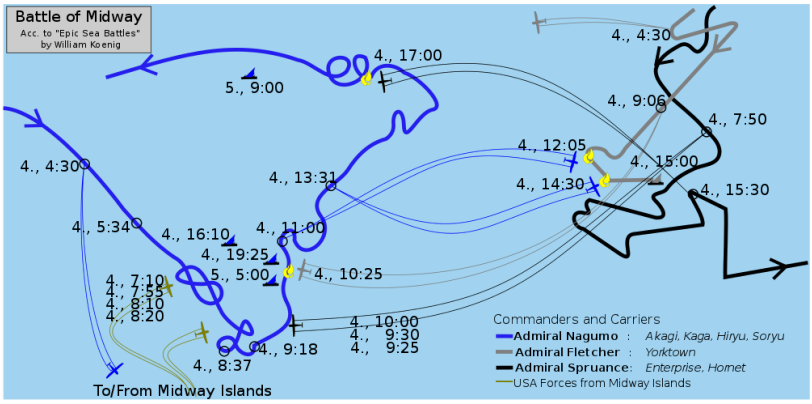



Figure 1.1: Overall Maneuver Map for the Battle of Midway (June 1942) [9]

### 1.1.3 Simulations at the Mission Level

The second failure by the Imperial Japanese military were overly optimistic assumptions regarding the quality of their aircraft and airmen. While a strategic blunder did undeniably occur, that failure was exacerbated by American ingenuity at the tactical and mission-level. Subject matter expertise and past experiences predicted significant mission and engagement level success for Japanese airmen and aircraft. The American *F4F* was inferior to its counterpart, the Japanese *A6M* (or *Zero*) in power, climb, speed, and range, the canonical Key Performance Parameters (KPPs) for air-to-air combat at the time. By performance properties alone, the American pilots should have lost almost every engagement, at least as per conventional air-to-air combat logic [3, 10].

However, the United States critically innovated in the form of better tactics for the supposedly inferior *F4F*. American aviators like Jimmy Thatch discovered that their aircraft could be used to bait their opponents into a disadvantageous position. Collaboration and unusual maneuver preferences could be employed to compensate, or even take advantage of, the American fighters' conventionally inferior performance. A result from these mission simulations was the "Thatch Weave" (see Figure 1.2) which was used by fighters to successfully defend American ships and strike aircraft [3, 10].

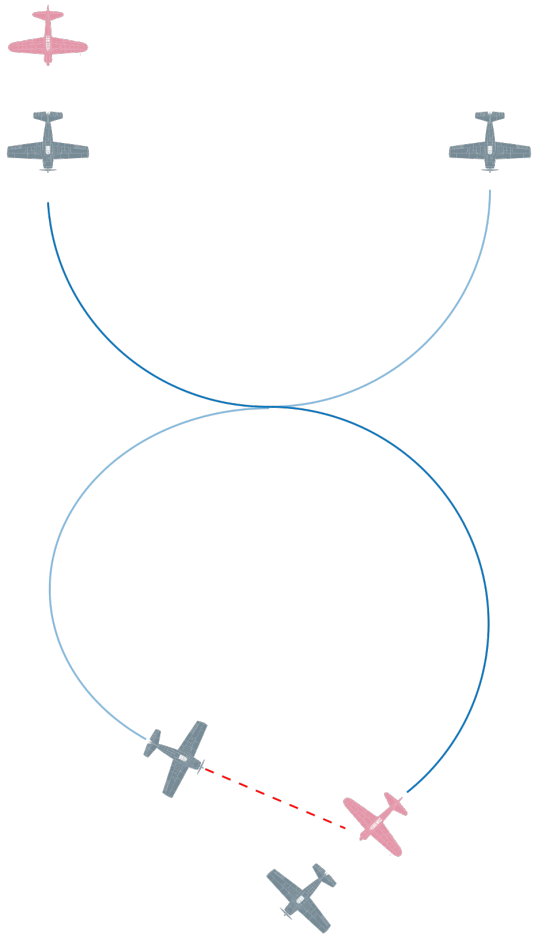


Figure 1.2: Example of the Thatch Weave [9]

### 1.1.4 Observations

These historical simulation missteps at both the strategic and the mission level represent the cost of failing to capture, explore, or articulate novel solutions. Today, the clever application of tactics and strategy is of immense interest for a capability-focused and resource-limited force [1]. However, both tactical and strategic simulation can suffer from limitations at identifying new and emergent capabilities [11]. In fact, these limitations are interrelated as higher-detail simulations and approximations inform those at a more aggregate level. As illustrated in Figure 1.3, each level of the simulation pyra-

mid informs or is approximated to inform the levels above it [12]. Therefore, improperly accounting for lower-level elements can impact the utility and quality of higher-level simulations and considerations.

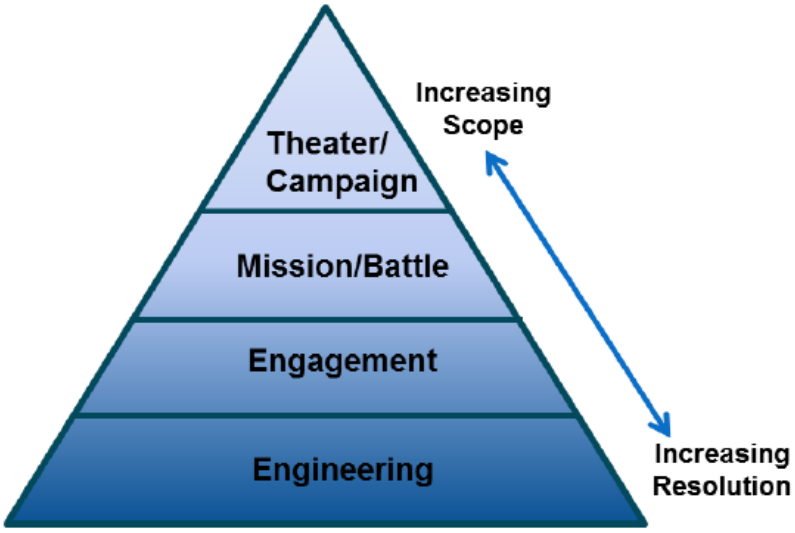



Figure 1.3: The "Pyramid of Simulation" as Defined by the Air Force Matériel Command [12]

The "Thatch Weave" represented an innovation at the Mission and Engagement level, while the use and strategy of Aircraft Carriers represented an innovation at the Campaign level. Still true today, if one were to experiment and identify an optimal Campaign-level strategy, an approximation of the lower-level missions and engagements would be necessary [12]. However, as evidenced by the assumptions made by the Imperial Japanese military command, it would be easy to develop an improper approximation for the lower-level combat between forces. Relying on performance metrics alone can contribute to the wrong formulation for the combat model, most likely a variation of the canon Lanchester Laws [10, 13].

### 1.1.5 Issues with Empirical Mission Approximations

The Lanchester Laws are a paired differential equation representation of combat which posits that the loss rate (attrition) of each side is a function of its opponents' population.

The equations were designed originally for large force-on-force engagements with limited tactical-level considerations by aggregating each side into massive units (e.g. entire army groups). This aggregation thereby assumes combat is a "furball," or relatively unstructured combat between the individuals within each massive unit [14, 15, 16]. Generally, these equations are driven by the populations and the relative effective firepower of each side (modeled as coefficients on those populations). For a more in-depth discussion of the Lanchester Equations, see Chapter 3.

The Lanchester Laws remain popular as a quick approximation of combat capability [13, 17]. More modern approximations use extensive test and mock battle data to inform the various coefficients in the linked equations. However, the results remain imperfect due to both the lack of modern validation data and an inability of historical data to predict off-design or novel uses of technologies [13, 18, 19, 20]. Meanwhile, various tradeoff environments have been developed for singular design problems, but these methods need to be extended to inform campaign-level considerations much like more complex variations of the Lanchester Laws before them [17, 21, 22, 23, 24].

### 1.1.6 Conclusion

The net result is that better understandings of mission capabilities are required for the proper utilization of campaign and life-cycle level simulations. Furthermore, these approximations must rely on a higher fidelity model rather than incomplete historical or test data if they hope to capture off-design or new capabilities. Lastly, any mission or engagement-level approximations require the use of decision-making algorithms to better simulate individual contributions and actions. This is notably distinct from relying on the large force-on-force approximations inherent in models like the Lanchester Laws. Furthermore, this difference is primarily driven by the rise of precision munitions and more capable individual assets as opposed to massed forces [13, 25]. The union of these requirements indicates that a new methodology is required.

## 1.2 Modern Problem Context

The quintessential role of a engineer is to design or develop a solution to an identified problem. The traditional approach to these solutions are technical and asset-based: find a problem, design a tool to solve it, and use that tool. New asset development conventionally requires extensive requirements generation to define the exact form the solution must take. However, Capability-Based Assessment (CBA) requires more life-cycle, campaign-level, and mission analysis to support the desired capability-driven solutions in lieu of conventional performance-driven solutions [23, 24]. Consequently, the DoD introduced the concept of Joint Capabilities Integration and Development System (JCIDS) and more broadly encouraged CBAs to develop new solutions.

Unfortunately, the rate of new asset development remains slow and the apparent rate of new threat emergence is increasing [26, 27, 28, 29]. Changes in the geopolitical framework affect not only the requirements for new capabilities, but also the funding sources and available technologies [26, 29, 30]. This is especially relevant as the primary concerns for the United States shifted from the Cold War, to transnational terrorism, to regional powers, to the rising challenges from state actors [4, 5, 31, 32]. Additional stand-off capabilities by would-be enemies further complicate the operational space and capabilities demanded of the military [32].

### 1.2.1 Capability-Based Solution Generation

Both the United States Department of Defense and commercial decision-makers have pushed into the realm of Capability-Based Solutions. In this method, the conventional approach to detailed requirements is eschewed in favor of focusing on the results rather than its specific form. The end-goal and broader constraints are considered, not deep specifics such as asset parameters and properties. Hierarchical Functional Decomposition (HFD), Model-Based Systems Engineering (MBSE), and various quality engineering

methods, such as Quality Function Deployment (QFD), support the capability focus instead of specific requirements [29]. Adding to the complication, development of both new hardware and applications has accelerated in the commercial realm, enabling the possibility of cross-utility for civilian systems, or novel application of military systems inspired by civilian approaches [33]. New methods like Set-Based Design have endeavored to include System of Systems (SoS)-level parameters and considerations during the design and solution generation processes [34].

Capability-focused approaches require a deeper understanding of the strategic, operational, tactical, and sustainment of the assets available. This is in contrast to the requirements and performance-focused (or traditional engineering) analyses conducted for previous solution-generation approaches [27]. These various forms of simulation fall under the umbrella of "Mission Simulation" wherein the focus is not on estimating the raw performance of a given design or solution, but rather the overall capability, success rate, and SoS elements affected. Capability elements span the gamut in both simulation and analysis complexity and in stakeholder inputs as illustrated in Figure 1.4 [29, 34].

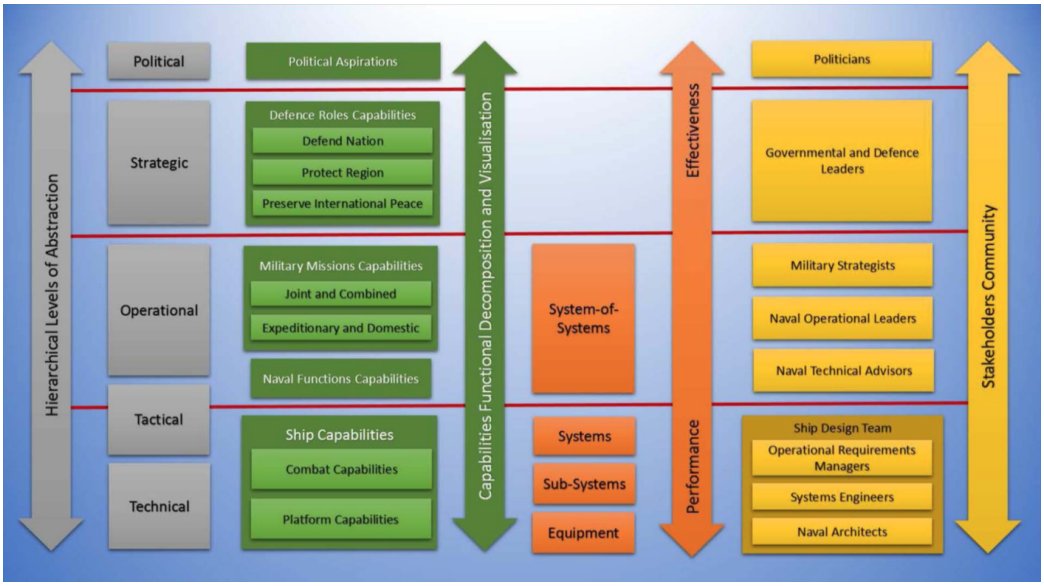



Figure 1.4: Capability-Based and System-of-System Based Analysis for Naval Applications [29]

Furthermore, former Deputy Defense Secretary Robert Work highlighted the need for more methods and mission simulations to inform decision-making, identify promising technologies, and better utilize existing assets. He believed that superior methods of wargaming and mission simulation were necessary to find and leverage these non-traditional solutions, or to better justify and employ traditional or existing systems [2]. The Chairman of The Military Operations Research Society further asserts that wargaming and other more detailed variations of mission simulation should be constructively integrated into the analytic processes supporting decision-making [35]. Changes to the expected risks and opponents also necessitate changes to the methods of operation for United States forces, with a particular focus on capabilities (or missions) rather than specific performance objectives. Efficiency is a worthwhile concern, but force effectiveness and capability is more relevant for accomplishing assigned missions [4].

### 1.2.2 Exploring the Mission Space

The solution to these problems is a renewed focus on capability-based methods and specifically a need for further developments in the realm of mission simulation and wargaming. The analyses needed for these methods will rely on new models, new datasets, or both. On the strategic level, decisions are often made using more aggregate or empirical data [35]. The power of strategic or operational simulations (war games) is to observe interactions and critical System of Systems effects, not to generate high-fidelity models of individual engagements. To still incorporate the lower-level elements, these simulations rely on empirical relationships or more general equations such as the Lanchester Laws [13, 17]. However, the historical data is less applicable to modern systems or revolutionary technologies. As the architecture and form of solutions shift, the previously utilized empirical models and relationships are no longer effective. This is especially relevant when the older model forms are difficult to abstract or connect to physics first-principles [13, 17].

The key aspect of these capability-focused methods is the use of the Mission Design Space. This variable space can vary dramatically in scale and scope depending on the mission addressed and the desired fidelity. On the lower end of the fidelity spectrum, a mission might be solely uniquely defined by the number and type of assets used for the mission. Such an approach would be akin to an Aggregate Combat Model or Lanchester Laws-based problem [13, 14, 17]. These parameters are classified in this document as Mission Architecture Variables in that they affect the overall form of the mission but not detailed tactical or maneuver actions. Mission Architecture parameters are also easily integrated for a Discrete Event Simulation or other forms of modeling which are not as concerned with intermediary actions.

For a high-fidelity mission simulation, the variable space might include every action by every entity at every discretized timestep. Such parameters are classified in this document as Mission Action Variables in that they detail the specific maneuvers, decisions, and activities of the simulated units. This variable space is much more intractable than the Mission Architecture Variables, but is more complete by incorporating not only the start and ending states (in common with a Discrete Event Simulation), but also all intermediary elements.

Changes in warfighting over the last few decades has encouraged the use of precision and high-quality but low-quantity units by Western military forces. As such, the actions by an individual unit has far more influence on a mission's outcome than when thousands of soldiers were marched into combat during World War I. This implies that ignoring individual actions and decisions is an oversimplification of the problem and thus requires superior entity-level models.

### 1.2.3 Level of Simulation

As outlined by the Pyramid of Simulation in Figure 1.3, there are several intervening levels of models between detailed engineering design and higher-level campaign simu-

lations. That breadth of "missions" requires further refinement to scope the problem for this research.

In the context of this work, a "mission" will refer to a small-scale tactical engagement between units. Examples include aerial dogfights, strike missions, and ground unit movements and engagements. At that level, simulated decisions affect singular units while those actions are conventionally aggregated into approximations equations like the Lanchester Laws. Looking at the Midway example, the proposed **SAA** approach would affect the combat between aircraft and between aircraft and warships, but would not seek to develop the larger battle strategy such as between the fleets involved. The fleet-level decisions are more amenable to strategic evaluation which would be facilitated by engagement and mission-level approximations developed in this research. While the method is capable of being extrapolated to that level, the scope of the resulting simulation is beyond a single project.

### 1.2.4 Physics-Based Modeling and Simulation

This focus on integrated capabilities has synergy with a push towards physics-based Modeling and Simulation (M&S) [36]. Previously, computational methods were more empirical in nature and used years or decades of accumulated testing data to generate approximations of various design or engineering decisions [37, 38]. However, physics-based methods return to first-principles and so are more flexible with regard to new designs or new missions.

This development required three elements to achieve higher understanding. The first was sufficiently detailed models to accurately simulate the physics and performance of a design without a reliance on empirical data [38]. The second was the idea of Design Space Exploration to span the available solution or variable space and generate new understandings [38, 39]. The third was the idea of Surrogate Models (or Response Surfaces) which took in the sampled solution space and generated an empirical relationship be-

tween available design variables and the metrics of interest. The resulting faster model (effectively a newly crafted empirical relationship) could then be employed to rapidly consider and evaluate tradeoffs [19, 40, 41]. These ideas were readily applied to the conventional Engineering Design Space, or the physical parameters and technologies of the assets being developed.

For the purposes of this thesis, two terms will be utilized to define the soldiers, vehicles, and other physical components of the military simulation. These two terms are "asset" and "entity." An asset refers to the actual physical unit being simulated. An entity refers to the model of a physical asset (or assets) in the combat simulation. For example, if the simulation is for a tank battle between two opposing forces modeled at the platoon-level, then the assets are the tanks themselves and the entities are the platoons of tanks involved.

### 1.2.5 Tradeoff Environments

However, improvements in computer M&S methods tended to focus on the physics and engineering design of potential solutions. Meanwhile, other efforts have pushed into a more integrated and inclusive solution trade-off environment such as the Unified Tradeoff Environment (UTE) which considers not only engineering design parameters, but also technology infusion and specific requirements [21, 22]. Though the UTE and similar ideas include mission-related parameters indirectly (usually via requirements), there does not seem to be the same problem variation and consideration in the true Mission Design Space. Figure 1.5 illustrates how requirements-based changes to a fixed mission architecture (X-axis) propagate into changes to the general Measures of Effectiveness (MOEs) (Y-axis) like gross weight and various cost metrics. Measures of Effectiveness are parameters which are used for comparing different solution options. Usually they are used to define "how well" a given solution performs its desired function [42, 43].

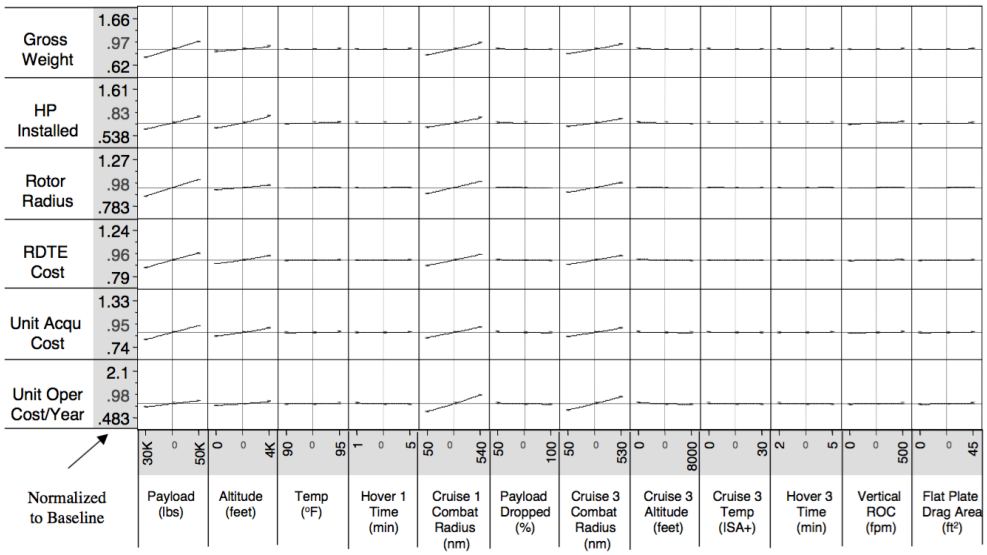



Figure 1.5: The Mission (Architecture) Space of a Unified Tradeoff Environment [21]

The Engineering Design Space is thus contrasted with the Mission Design Space. Consider a notional example of designing a new civilian aircraft. If the actual form of the aircraft (wingspan, engine parameters, material selection, etc.) is the Engineering Design Space, then the specifics of the flight path (payload, route, climb slope, throttle settings, etc.) would constitute the Mission Design Space. Thus, a capability gap is identified in the realm of mission simulation. While new forms of Modeling and Simulation have arisen for the engineering design of systems, similar methods have yet to be applied the mission-based aspects of potential solutions.

As was conducted for the Engineering Design Space, these approximations and tradeoffs should be equally feasible for the Mission Design Space. The necessary steps are for physics-based models to simulate the mission, a Mission Space Exploration to sample across the Mission Design Space, and lastly the application of a Surrogate Model to the sampled variable space to facilitate faster solution investigation and incorporation. A better-informed and more traceable mission or engagement approximation would thus assist with strategic and operational-level Modeling and Simulation by supplementing or replacing existing empirical representations.

## 1.3 Research Overview

This thesis proposes the **Stochastic Agent Approach** for addressing the lack of detailed tradeoffs of mission-level capabilities when comparing them at higher levels. Through the appropriate application of mission simulation, decision-making techniques, and physics-based modeling, it is possible to not only articulate and evaluate a wide variety of missions, but also do so in an adaptable and verifiable manner. This document details the literature review, research, experiments, and proposed sequence associated with the **SAA**.

The primary proposed solution to the weaknesses apparent in solution-generation is the application of a "Mission Space Exploration" in the same manner that "Design Space Exploration" revolutionized asset development. These two explorations both start with a higher fidelity model of a particular system or mission. They then use an intelligently-selected large number of cases (i.e. a Design of Experiments (DoE)) sampled across their variable spaces to generate a broader understanding. Lastly, a Surrogate Model (or other approximation function) is applied over the large dataset. This more computationally efficient representation can then be queried for even more cases to facilitate understanding and integration with a higher-level simulation suite.

### 1.3.1 Summary of Contributions

The end-goal of the **SAA** is to use higher fidelity Agent-Based Model (ABM) to inform mission-level approximations for infusion into a campaign or life-cycle level model.

The major hurdle to developing a mission-level Surrogate Model is the sheer variety of possible mission solutions. If every considered design variable is an action by an entity at a given time, the variable space explodes uncontrollably. The scaling problem is accentuated by the "chaining" behavior implicit in a simulated mission, where early actions affect the utility of later ones. Thus, a standard approach is to use a deterministic

Agent-Based Model while varying various system architecture or capability settings [20].

However, making the ABM deterministic and fixed will by definition miss the potential emergent behavior of new tactics and strategies. One solution to this problem is to leverage a system such as Dynamic Scripting to identify ideal logic-chains for an optimization of entity Artificial Intelligence (AI) [44, 45, 46]. While the optimization approach is incompatible with the exploration required for tradeoff understanding, the same variable breakdown could be considered with binary or Boolean values representing whether a given behavior is included. That variable space is thus more comprehensible than the illogical articulation, tracking, and variation of every agent action at every timestep.

Instead of this binary and large-variable approach, the proposed solution at the core of the **SAA** is ABVs. These parameters in effect convert otherwise binary decision-making present in ABMs into continuous parameters. Rather than a notional stimuli generating the same fixed reaction from an agent, there is instead a probability of one decision over another, hence a "stochastic agent." This method is inspired by stochastic processes like Markov Chains and more flexible optimization routines like Simulated Annealing [38, 47, 48]. The resulting continuous ABVs can be employed in the Mission Space Exploration and the generation of a Surrogate Model as with any other conventional and continuous variable. Preferences for certain actions or maneuvers can then be seen as having an effect on the simulation in a less-binary manner alongside potential synergies between the involved design variables.

In addition to the introduction of Agent-Based Variables, this research also conducted significant literature review, logic exercises, and experiments aimed at addressing supporting research elements. There is an overview of mission simulation methods, with the eventual selection of computer-based M&S (more specifically an Agent-Based Model) as the simulation method of choice. Supporting experiments also identify the necessary replications of each design point to achieve understanding of stochastic ef-

fects and which DoE methods and Surrogate Models are appropriate.

### 1.3.2 Document Overview

For reference, Appendix A includes a glossary of terminology and notes used throughout this document. Some of the terms are slightly irregular while others have multiple possible meanings.

Chapter 2 provides a more in depth background and motivation for this research. It also contextualizes various past approaches in the realm of solution generation, design, and process development.

Chapter 3 introduces the first research question which aims to identify an appropriate simulation method to underpin the approach. The selected method is required to generate the requisite cases for a Mission Space Exploration.

Chapter 4 focuses on the second research question and details efforts to reduce the dimensionality problem that arises when considering the Mission Design Space. This chapter introduces the concept of Agent-Based Variables and illustrates their utility through a notional strike mission example model.

Chapter 5 is built around the third research question and defines and contextualizes the idea of ABVs. A robust identification method is proposed and described in this chapter.

Chapter 6 develops a sampling method for an Agent-Based Variable-involved Mission Design Space. It consists of work related to Research Question 4 and includes efforts to identify the necessary replicate number (to address simulation-level stochasticity) and the appropriate form and size of the employed Design of Experiments.

Chapter 7 describes the development of a Surrogate Model over the ABV-involved Mission Design Space. It encompasses Research Question 5 and considers not only variations in the type of Surrogate Model, but also what trends and conclusions can be drawn from these approximations. This chapter also includes a comparison between

the **SAA** and a Lanchester Laws-based approach which employs a series of differential equations to represent the mission.

Chapter 8 is the conclusion and includes discussion of potential future work inspired by this research as well as possible applications.

# CHAPTER 2

# BACKGROUND AND PREVIOUS RESEARCH

> War games help strip down a strategic, operational, or tactical problem and reduce its complexity in order to identify the few, important factors that constrain us or an opponent.
>
> - Robert Work & Gen. Paul Selva [1]

## 2.1 Overview of Engineering Problem Solving

The value of a Mission Space Exploration (and engagement-level tradeoff analysis) is contextualized by understanding the roles and properties of engineering or military problem solving approaches. Various methodologies exist to generate candidate solutions, but many of them also assume that requirements are fixed ahead of time. Furthermore, most of the solution-generation approaches common in Aerospace Engineering and in the Defense Industry assume that the solution will be matériel in nature [11, 27]. Matériel solutions are ones which are firmly rooted in the design and deployment of physical systems rather than finding new uses for existing assets. The matériel solution assumption effectively constrains the solution space. In summary, the focus on requirements, asset design-based solution generation, and the resulting optimization methods have all impeded the investigation of mission-based tradeoffs.

Historically, the generation of solutions has required a breakdown of the capability gap, a set of requirements for a system to bridge that capability gap, and then the design, development, and testing of that system [49]. A capability gap arises when a given mission cannot be achieved using current equipment (assets) and doctrine. Once that

gap is identified, the requirements and designed solution intend to bridge that gap. Frequently, a thorough comparison and evaluation of candidate solutions is required and was popularized with the advent of more complex engineering approaches [50]. With multiple options on the table, a more refined comparison and requirements-generation methodology was required [30]. The next section thereby details past approaches to solution generation in engineering and the aerospace industry in general.

First though, an overarching element of solution develop is the idea of the Evaluation of Alternatives (EoA), or effectively comparing all possible solution alternatives at a sufficient fidelity to make a decision. Notably, an EoA might be split across multiple processes depending on the specific solution methodology selected. Examples such as the Integrated Product and Process Development (IPPD) system have a single stage of comparison while the Department of Defense's JCIDS has it split between the initial CBA and the later Analysis of Alternatives (AoA) [49, 51]. Regardless of the methodology employed, identifying and evaluating a diverse selection of alternatives is necessary to inform the final decision-maker(s). In general, all solutions to an identified capability gap are composed of asset elements and mission elements, which can be colloquially referred to as the means and ways of the solution.

### 2.1.1 Means and Ways

The conventional solution-generation approach, while focused on an engineered solution, implicitly constrains key elements of that solution: the mission, requirements, and objectives for that design. In short, all solutions have two sides: the means and the ways. Conventional engineering design sets the ways to a logical default while enabling the consideration of various means to achieve that mission. Mission-focused solutions fix the means, or what assets are available, and allows permutations of the ways to find new uses or missions.

*Means-Based Solution*

The first example is for an engineering-focused, means-based solution. In this case, an Request for Proposal (RFP) goes out for a new aircraft. This aircraft must fly a certain speed, have a certain range, and carry a certain payload. Thus, the mission profile and requirements are relatively fixed. An engineering-focused solution is proposed whereby a new (or modified) aircraft is designed and simulated until a final design is selected. That system is deployed and the capability has been achieved.

*Ways-Based Solution*

The second example is for a mission-focused, ways-based solution. In this case, the RFP simply requires that a total payload is moved between two points in a certain amount of time. Given that there is no time to design and build a new solution, the means are fixed to whatever assets are posited as being available. The mission or ways-based solutions thus include various combinations of aircraft, ships, and ground systems which utilize existing units to move the cargo in the required amount of time.

*In Summary*

Traditionally, only the first example is considered by engineering problem-solvers. However, with increasing use of Commercial Off-the-Shelf (COTS) and Government Off-the-Shelf (GOTS) alternatives, ways-based solutions are not only more possible, they are more difficult to predict due to the variety of alternatives [11, 28]. A modern example is the F-22 serving in Syria, where the "thoroughbred" air-supremacy aircraft was adapted for a light bomber role through the inclusion of the Small Diameter Bomb (SBD) [52]. Thus, full-fidelity problem solving is no longer divided between engineers designing a solution and decision-makings developing new missions. A superior alternative should be achieved if two sides combine their efforts synergistically rather than sequentially.

### 2.1.2 Engineering Design Focused Methods

When presented with a capability gap, the conventional approach is to design a solution. During an era of profligate defense spending, the approach was not only common, but frequently encouraged as part of the arms race between the United States and the Soviet Union [11, 28]. The engineering design approach can be broadly divided into two camps, those that focus on developing optimal (usually for a single mission) solutions and those that focus on broader understanding and robustness. These camps are defined by their solution generation archetypes, Design Space Optimization and Design Space Exploration, respectively.

#### *Design Space Optimization*

Design Space Optimization is the more typical form of Engineering Design Space utilization. As in traditional solution-generation methods, the objectives are frequently set *a priori*. Thus, optimization enables a designer or decision maker to evaluate a series of candidate designs such that a "best" option is identified. Optimization is frequently incremental and improving sample to sample [49, 53]. In the case of an evolutionary optimization routine such as a Genetic Algorithm, each subsequent "generation" of solution candidates has (or hopes to have) superior alternatives to its predecessors. Other more continuous optimization approaches might use a function's Hessian or differential to seek and follow "downhill" directions until a minimum is found [38].

Regardless, the incredible non-convexity of a large-variable design space can introduce additional complications through a variety of local minima [53]. Furthermore, the optimal design point does not guarantee a robust design point. As illustrated by Figure 2.1, different design points have different degrees of robustness. An optimal point might provide the best solution, but slight variations or imperfections may rapidly degrade that solution's utility. Conversely, a robust point might be non-optimal, but maintains much of its utility despite variations or imperfections. In graphical terms, an optimal point is

in a valley while a robust point is along a plane.

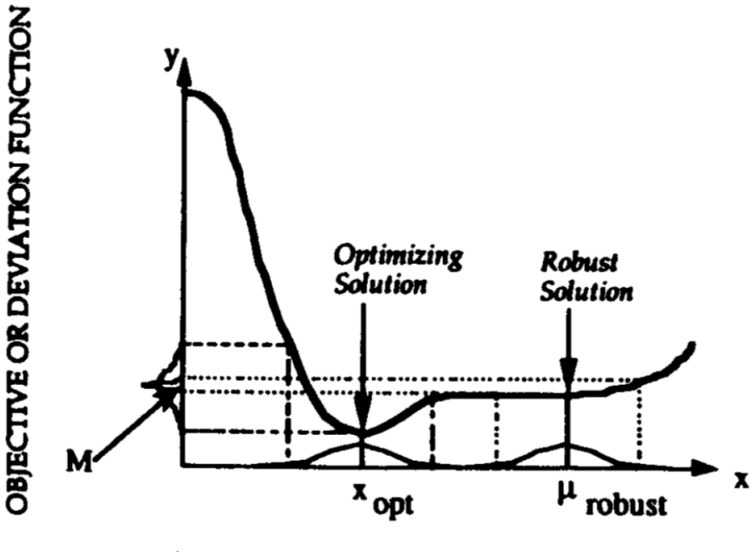



Figure 2.1: Notional Illustration Comparing the Optimal Design Point to a Robust Design Point [39]

*Design Space Exploration*

Design Space Exploration is a general concept whereby a solution space is queried extensively in order to understand the parameters of that space. Such an exploration of a variable space tends to generate a mix of reasonable, impossible, and fanciful designs. However, the key with a Design Space Exploration is that all design options are considered, even if only briefly [39]. The thought is that by conducting this sampling, one can either better justify the elimination of certain options or discover emergent or revolutionary alternatives not previously considered [39, 53]. This is especially enabled by avoiding the narrowing of considered solutions required of optimization methods.

Though a simple idea in principle, Design Space Exploration is a relatively novel concept due to computational and time limitations [53]. When designs were hand-drawn, modeled, developed, and designed, there was limited financial justification of the time and labor involved in investigating so many disparate designs [54]. Incremen-

tal and evolutionary design improvements, rather than revolutionary advances, were also encouraged in the interest of safety [11, 27, 49]. The lack of exploration capability fostered optimization-based solution approaches. Incremental developments gradually improved designs toward a design optimum as defined by customer requirements. Higher, faster, larger, more capable systems were the result. However, these "thoroughbred" assets started to lack flexibility. Though excelling at a specific mission, they failed to be amenable to shifting requirements and mission types [11, 27]. As the development cycle for new assets lengthened, this inflexibility proved to be more and more of an impediment.

It wasn't until improvements in computational models that such a broad-based sampling of designs was feasible or worthwhile. Work researchers such as Mavris, illustrated how these Design Space Explorations, conducted quickly and efficiently through computer models, can help engineers better understand the solution space available [19, 21, 22, 39, 41]. Areas of technology improvement or requirements relaxation became more apparent as other design points were considered on their capability merits rather than their requirements failures [55].

### 2.1.3 Mission Focused Methods

Mission-focused solution generation was traditionally reserved for decision-makers or units in the field. However, improvements in simulation have unlocked a capability to evaluate new missions and ways-based solutions [11, 27, 28]. With more analysis required and requested for these mission-focused methods, there has been a similar bifurcation of the community between those leveraging optimization and those leveraging exploration.

*Mission Space Optimization*

In the realm of mission simulation, computational resources have been largely leveraged to conduct ever improving optimizations of mission plans and architectures. Research by Revello, Kewley, Embrechts, and others illustrate how optimization-focused approaches are popular in this community [56, 57, 58]. Even complex war games aim to develop optimal strategies through the fusion of multiple different models with clear-cut objectives and available parameters [59]. Such approaches would involve routines where the mission is steadily iterated on until a solution is identified.

The favoritism shown toward optimization can be seen in the early implementations of electronic mission simulation. Nuclear war games, simulations seeking to model the exchange and second-strike of nuclear weapons, followed the trend perfectly with a distinct focus on identifying an optimal plan of attack and counter-attack [60]. Furthermore, these simulations made too many systems-of-systems-level assumptions including perfect command and control and rational commanders and personnel [60].

Seemingly simplifying the problem, the Mission Design Space is a variable space where the optimum can often be qualitatively well understood. High safety, high success rate, low cost, and low risk are paramount and so optimizers seem well adapted to the problem. In the defense sphere, maximizing the probability of mission success and minimizing casualties are obvious objectives. However, optimums are misleading because the best on one axis is rarely the best on another. However, the relationships between output parameters can be articulated and compared by SMEs and decision makers thereby ensuring optimization remains an effective approach [21, 22].

By definition, optimization is dramatically affected by changes in requirements and constraints [11]. As these parameters shift, the space of available solutions shifts as well. As the solution space changes, what constitutes the optimum can change nonlinearly [53]. These changes can convert formerly ignored solution points into the optimums while prior optima may become infeasible or violate shifted constraints. The contours

of the solution space are not guaranteed to remain constant, thereby minimizing the utility of the iterative (and relatively narrow) search which constitutes optimization. Consequently, these shifts in requirements necessitate a new optimization to be run, increasing the cost and time elapsed before a solution can be identified.

Optimization efforts at the mission level remain popular for developing guidance routines for missiles or training for combatants. Garcia notably uses such decision-making optimization approaches in evaluating ideal evasion techniques for aircraft or munitions [61, 62]. These simulations are much smaller and usually only involve individual units of each side engaging each other (i.e. one aircraft evading one missile) as illustrated by Figure 2.2. As a result, optimization remains valid because the Mission Design Space for that scenario is small.

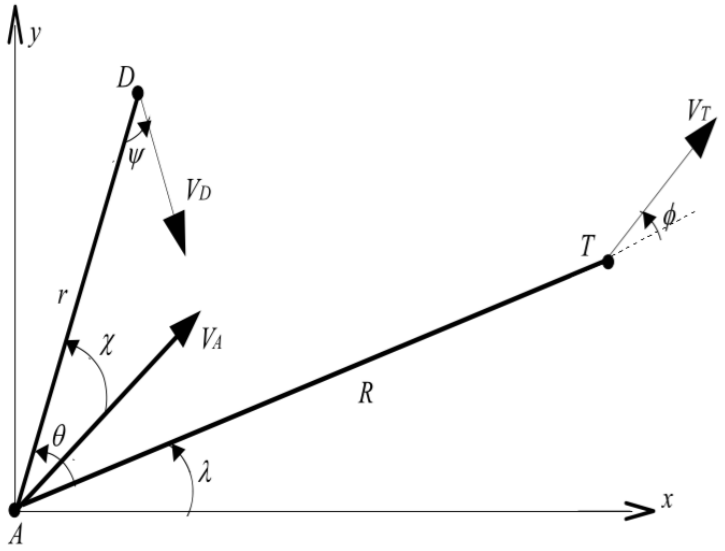



Figure 2.2: Example of the State-Based Maneuvering Engagement Model Used by Garcia [61]

As seen in Figure 2.2, there are only three (3) simulated units in the mission. There are the allied Target (T) and Defender (D), and the opposing Aggressor (A). The Aggressor seeks the Target while the Defender protects the Target. The example is only in two

dimensions and decomposes the problem into a control-based state-space with the goal of optimizing the maneuvers of the units to maximize their individual probabilities of mission success.

Evolutionary optimization techniques are commonly used due to the variability of the solution space and the disparate forms and ranges of the variables used [57, 58]. The Genetic Algorithm is a common example of these evolutionary optimization routines and consists of converting all design points into binary "genes." These genes are then evaluated in the model, rated for their "fitness" (or closeness to optimality). A new population of the design points is generated by "crossing" and "mutating" the binary genes with a preference for maintaining the high fitness genes from the previous population. Thus, each generation of design points is iteratively superior than its predecessor, though there is no guarantee of finding the true optimum [38, 41, 57, 58]. An example of a gene from such an algorithm is provided in Figure 2.3.

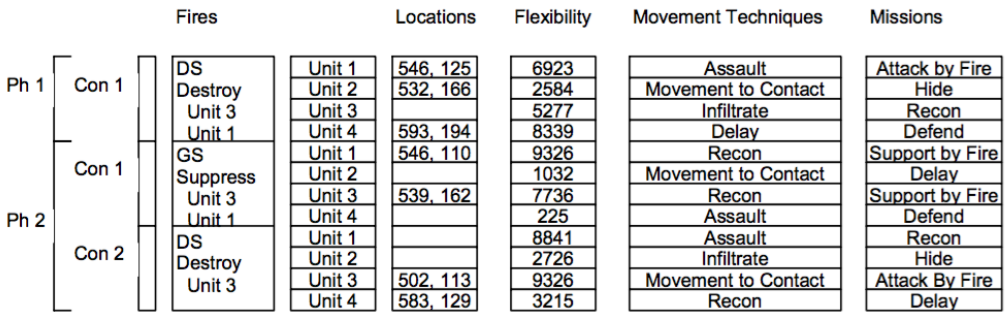



Figure 2.3: Example of a Mission Plan "Gene" for a Genetic Algorithm Mission Optimization Problem [58]

Though these optimizations are useful and relevant for considering mission-focused solutions, the algorithms require some foreknowledge of what constitutes a good or successful mission. Without that knowledge, there is no ability to define optimality or a fitness function, preventing effective comparison of design points or identification of promising parameters. Fortunately, running the same solution through different scenar-

ios can be used to find more robust solutions by comparing and combining the fitness of each solution through each scenario [56]. Unfortunately, the computational overhead of an optimization remains and is multiplied by the need simulate and rate each design point through each considered scenario.

*Mission Space Exploration*

Mission Space Exploration is similar in principle to Design Space Exploration. However, the complexity of the Mission Variable Space makes it more difficult to quantify and articulate enough points to constitute a thorough exploration. Some degree of exploration has already occurred in the context of war games. As Revello discussed, there is a need for robust solutions to a variety of opponent settings as well as addressing the inherent stochasticity of such scenarios [56]. Sometimes though, an exploration is conducted but is in reality a multi-objective optimization problem or a search for a robust but still "optimal" solution [56]. Some efforts have already infused elements of Mission Space Exploration into the solution generation process. These result in variations of the trade-off environments such as the UTE [21, 22]. Figure 1.5 from Chapter 1 illustrates this form of requirements-based Mission Space Exploration.

*Observations*

These optimization-focused approaches are not necessarily helpful when moving beyond a narrowly-scoped mission simulation. When including higher-level strategic and operational considerations, the significant investment of time in optimizing a tactical mission is less favored than an empirical or easy-to-compute relationship [19, 41, 63]. Furthermore, an optimization routine does not function well without well-defined objectives. Lastly, generating a Surrogate Model or approximation of expected mission effects when only optimums are considered in inherently flawed. Missions are variable, noisy, and complex, so a generated approximation of the outcome(s) should include

such potential variations and distributions. Furthermore, the possibility of failure is ever present and as with Design Space Exploration, infeasible or non-viable points are equally illuminating for a decision-maker.

### 2.1.4 Capability-Based Approaches

The introduction of Design Space Exploration facilitated the shift toward the idea of capability-based approaches. The concept of solution exploration and improved computer models both enabled a larger number of considered alternatives and therefore the possibility of more effectively comparing disparate solutions. As these solutions might be of different archetypes, the primary point of comparison was their capabilities, not specific performance metrics [50, 64]. The capability-focused paradigms have propagated throughout the industry and government, and taken a variety of forms depending on the specific organization or application.

Capability-Based Acquisition is the current ideal for DoD solution generation. The objective is to acquire an end-result system that can accomplish a specifically assigned goal while requirements are primarily informed by integration and Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) concerns [50]. The related Capability-Based Planning method similarly aims to achieve a particular goal, but keeps specific requirements at the cost and sustainability level [65]. Additionally, the United States Department of Homeland Security has similarly attempted to employ capability-focused acquisitions and solution-generation methods through the Joint Requirements Council [66].

A focus or attempt to employ capability-based methods requires more mission space simulation and evaluation than previously considered. Unlike the conventional approach, CBA requires the investigation of variable mission parameters rather than only engineering design parameters [11, 27, 29]. Thus new mission models are required, more specifically ones which can cover a wider Mission Design Space and the inclusion

of in-development designs rather than existing systems with historical data to inform empirical models.

## 2.2 Specific Solution Generation Approaches

With the broad categorization of solution generation methods, there are numerous specific forms in use today. Furthermore, it is appropriate to contrast the more modern methods with either their predecessors or more conventional solution-generation approaches.

### 2.2.1 The Generic Design-Based Problem Solving Approach

One of the most generic approaches to engineering problem solving was documented by Finkelstein and Finkelstein in 1983 [67]. In this paper, the authors detail the generic problem solving and design approach as visualized in Figure 2.4.

In summary of their review, the authors described a general five-step process of developing and evaluating possible designs. The process begins with the research and formulation of the value model which sets the priorities of the various capabilities and performance metrics expected of the new design. Once those priorities have been identified and outlined, various competing designs are envisioned and sufficiently approximated so as to allow comparison. The fourth step is the analysis and comparison of these competing designs through whatever methods are available to the designers and decision makers. Finally, a decision is made as to which design to proceed with [67].

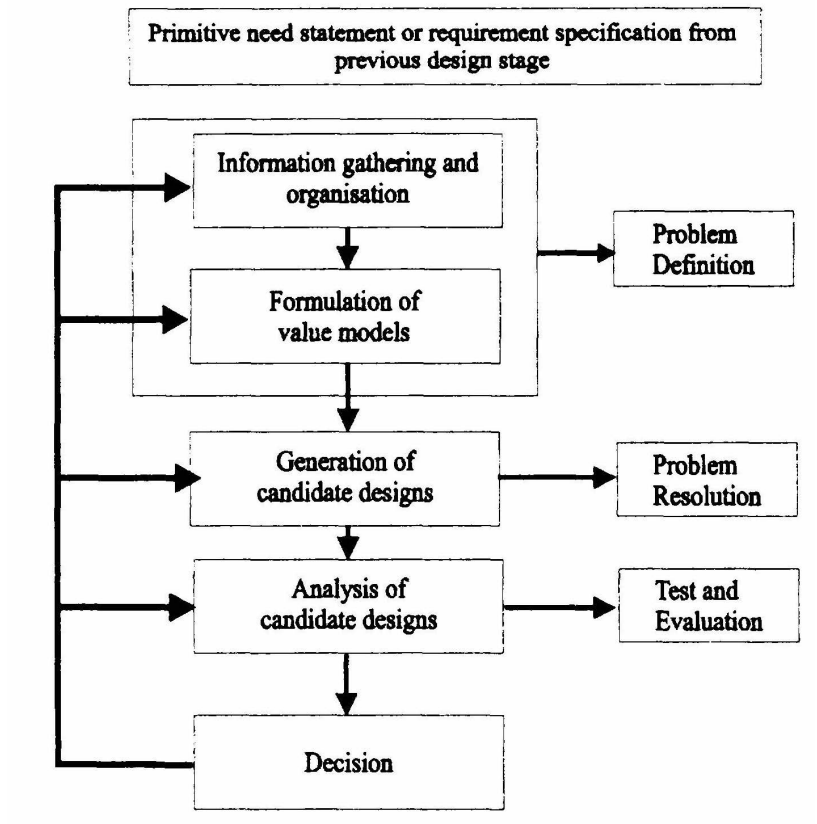



Figure 2.4: The Stages of a Generic Design Process as Outlined by Finkelstein and Finkelstein [67]

In this general model, feedback exists purely from the capstone decision to the previous stages. While more limiting that a broader range of feedback, this approach allows for the adjustment of capability or performance priorities, design attributes, and analysis methods as the decision-maker possesses a progressively more clear picture of the problem and candidate solutions. However, the weakness of the generic Finkelstein approach is that requirements are defined *a priori*, or before significant solution development has been conducted. Furthermore, the simplicity of the method is a handicap in a capability-based context where not all solutions follow the same archetype, making direct comparison more difficult.

### 2.2.2 Integrated Product & Process Development (IPPD)

The IPPD approach is a more modern take on the sequence of events needed for the successful proposal, configuration, and deployment of a design. The approach was developed at the Georgia Institute of Technology as a means to specifically incorporate Quality Engineering (QE), Systems Engineering (SE), and computer-integrated design approaches into one coherent methodology [49]. This approach is illustrated by Figure 2.5.

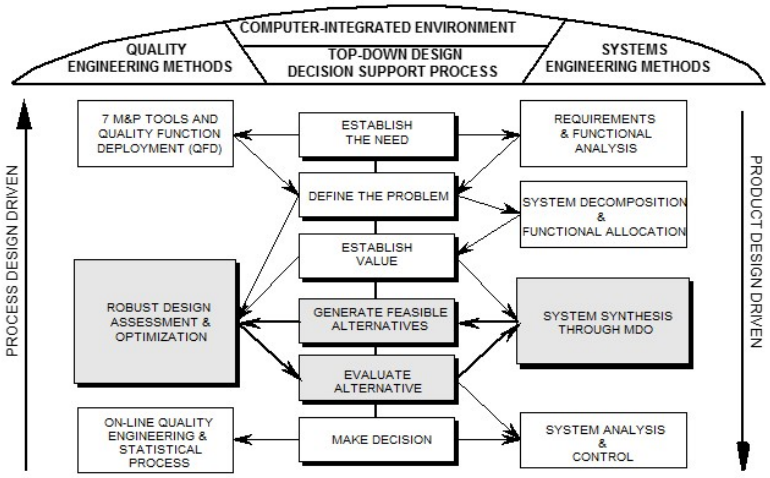



Figure 2.5: The Georgia Tech Integrated Product and Process Development Approach Flowchart [49]

At the core of the IPPD approach is the use of computer models to ensure the development of robust (or resilient) solutions with care taken to address and quantify the risks and complications that inherently arise during the design process. At the core of the IPPD methodology is a variation on the design sequence described by Finkelstein, with the first step of the process now divided between “Establish the Need” and “Define the Problem.” However, the IPPD approach incorporates far more feedback and interactions between both the general stages (center column) and the QE and SE steps taken to ensure the full quantification of necessary elements (left and right columns, respec-

tively) [49].

### 2.2.3 Unified Tradeoff Environment (UTE)

The Unified Tradeoff Environment is a method for illustrating the various methods for concocting a solution by adjusting more than simply the conventional Engineering Design Variables. This approach serves as a measure of inspiration for this research as it aims to expand the traditional Engineering Design Space to include other possible adjustments to the solution space. It is a tradeoff environment because it facilitates the comparison of different adjustments to solution parameters and requirements in a unified manner [21, 22].

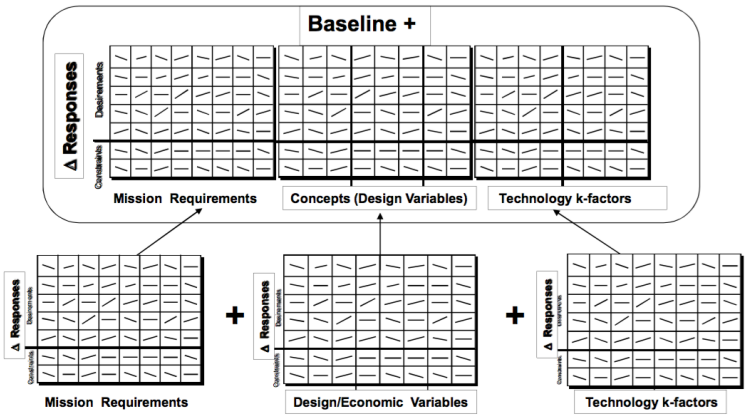



Figure 2.6: An Example of the Unified Tradeoff Environment with all Three Variable Spaces of Interest (Mission, Design, and Technologies) [68]

The UTE identifies three different variables spaces of interest: the design space, the technology space, and the requirements space (illustrated by Figure 2.6). Each one of these variable spaces possess continuous representations of each element. Modifying these variables generate potential solutions either by directly modulating the affecting variables (design and technology variables) or changing the requirements for what constitutes a successful solution (requirements variables) [21, 22, 68].

*Design Space*

The design space in the UTE context is functionally identical to the Engineering Design Space discussed previously. Of note is that this space is driven by more direct design variables and variations on a given system architecture and technology selection [21, 22, 68]. For example, an aircraft's wingspan or other outer mold line properties are design space variables.

*Technology Space*

The technology space in the UTE context refers to the potential infusion of new technology options. Notably, the assumption is that technology selections do not directly affect the aforementioned design space variables [21, 22, 68].

It is worth noting that the technology space might not actually be discrete and individual technologies. Instead, they can consist of so-called K-factors which affect intermediary performance and capability parameters [21, 22, 68]. For example, a new engine architecture might provide superior fuel efficiency which would be represented by an effect on the Thrust Specific Fuel Consumption (TSFC) K-Factor. The new engine capability is an integrated or intermediary effect, rather than a direct design variable change. For example, a new type of engine with similar form factor such as the Geared Turbofan would be an element of technology infusion with minimal effect on overall engineering design properties. Furthermore, other such technologies could be internal elements like material selection or subsystems.

*Requirements Space*

The requirements space in the UTE context refers to the set of objectives for the selected design and solution [21, 22, 68]. For the Unified Tradeoff Environment, the changing requirements ultimately affect the filtering on the wide spectrum of solution combinations generated from the engineering design and technology variables. For example, a

requirement for a certain life-cycle cost is defined. However, no feasible or viable solution is possible. Thus the life-cycle cost requirement is reduced, thereby allowing for the use of solutions which previously failed the requirement.

*Reason for Interest*

The Unified Tradeoff Environment is of interest because it is a methodology that aims to bridge the gap between developments in high-fidelity approaches and the high-computational efficiency methods most relevant and useful to decision makers. The Unified Tradeoff Environment leverages Surrogate Models to generate "accurate enough" relationships between all types of available design variables discussed previously. In a similar manner to how the UTE overlays physics and technology models, the proposed **SAA** seeks to enable the same approach for the mission or engagement space. However, engagement-level surrogates are more readily applicable to strategic and operational war games and life-cycle simulations, rather than the engineering design process affected by the UTE.

### 2.2.4 Requirements Generation Systems (RGS)

The Requirements Generation System (RGS) is the former method of handling acquisitions and bridging capability gaps for the United States military. It is most similar to the Finkelstein approach in execution as the focus was on new matériel solutions and developing the necessary requirements to generate the candidate designs [7]. Most of these problems are proposed through a conventional RFP with clearly enumerated performance or composite design metrics (e.g. speed, weight, payload, outer dimensions) required of any candidate solutions.

A weakness of the RGS approach is that it assumed specific new asset performance properties map directly onto mission effectiveness. This approach was acceptable on older systems without the technological or integration-related concerns that arise in modern assets [69]. The complications associated with new asset development and de-

sign are common across the aerospace industry. However, many of the best examples of such limitations are found in the context of defense acquisitions. For example, overly specific requirements do "not allow for a design that has a max level speed of 540 knots, but has double the range and payload over competing concepts that achieve the speed requirement" [55]. Despite the attempt to focus on specific requirements, one of the "Top 5 Systems Engineering Issues" as identified by the National Defense Industry Association is that "requirements definition, development, and management is not applied consistently and effectively" [70]. Thus the RGS was seen as a solution-generation dead-end and replaced by the JCIDS methodology [71].

### 2.2.5 Joint Capabilities Integration and Development Systems (JCIDS)

JCIDS is the solution-generation approach utilized by the United States Department of Defense (as of 2018) and replaced the Requirements Generation System in 2004 [71]. Its objective is evaluate a wide range of possible solutions, both matériel and non-matériel [72]. The process covers the full spectrum of requirements generation, ranging from initial validation of the perceived capability gap through the specific generation of a RFP for a new procurement [73]. The JCIDS approach is worth considering if only because of the amount of resources dedicated to solution development through this particular method. Furthermore, JCIDS was a deliberate effort to generate more capable, time-efficient, and cost-efficient solutions [55]. In short, JCIDS embraces both means and ways based solutions, not just the means epitomized by RGS

JCIDS consists of two primary solution branches centered on matériel and non-matériel solutions. A matériel solution involves the conventional engineering approaches discussed previously. Meanwhile, non-matériel solutions consist of all efforts to bridge a Capability Gap via existing systems and modifications of non-hardware elements. The acronym DOTmLPF-P is a shorthand reference to the various non-new matériel components that provide capabilities and solutions. The term is usually used in a context that

implies a solution can or is reached using currently available assets, rather than newly required or procured matériel, hence the lower-case “m” [69]. Table 2.1 defines these elements of DOTmLPF-P with definitions paraphrased from sources identified during the course of this research [7, 74].

Table 2.1: Components of DOTmLPF-P [7, 74]

| | |
|---|---|
| Doctrine | The fundamental principles that guide the employment of U.S. military forces in coordinated action toward a common objective. |
| Organization | The way a force organizes to accomplish missions, execute functions, and deliver, support, or sustain warfighting capabilities. |
| Training | Individual, staff, and collective training in doctrine or Tactics, Techniques, and Procedures (TTP). |
| Matériel | All items necessary to equip, operate, maintain, and support military activities without distinction as to its application for administrative or combat purposes (notably refers specifically to currently-fielded matériel). |
| Leadership | Authorities that oversee, coordinate, and command. |
| Personnel | Those individuals required in either a military or civilian capacity to accomplish the assigned mission. |
| Facilities | A real property entity consisting of one or more of the following: a building, a structure, a utility system, pavement, and underlying land. Key facilities include command installations and industrial facilities of primary importance to the support of military operations or military production programs. |
| Policy | Policy can direct, assign tasks, prescribe desired capabilities, and provide guidance for ensuring the Armed Forces of the United States are prepared to perform their assigned roles. |

In short, a non-matériel solution is a mission-focused solution to the problem. Furthermore, it includes a mix of tactical, operational, and strategic possibilities into the realm of solution generation. This is in comparison to the RGS approach which implicitly favored new assets and engineering-based solutions to an identified capability gap.

The other branch of the JCIDS approach is for new asset procurement and, generally speaking, is not dissimilar from a conventional requirements-driven process [72]. Once more, objectives are set and designs are proposed to meet them. The difference is that many of these requirements are capability-focused rather than specific numeric or performance metrics.

Regardless, the existence of a non-matériel solution branch is a major departure for JCIDS relative to the RGS approach. It is that mission-focused non-matériel solution branch of the process that introduces a need and potential improvement in the form of superior mission modeling.

## 2.3 Limitations and Failures of Existing Solution Generation Methods

Despite the introduction of the JCIDS approach, there remain issues regarding the timely and cost-effective acquisition of new DoD assets [71].

### 2.3.1 Solution Cost and Time Overruns

One of the primary issues with any form of DoD solution development is the relative inability of military systems to change dramatically. This inflexibility complicates the procurement process by inherently favoring proven, but perhaps less optimal, designs over less familiar, but perhaps equally capable, designs. For example, the design and procurement of a new ship for the United States Navy often involves a 5-10 year design period and a 2-7 year construction time, for an asset which will have a service life from 25-50 years. As such, it takes on the order of 30 years (if not more) to dramatically shift the major assets available to naval leadership [27]. In the words of the House Armed

Services Committee:

> *The rising costs and lengthening schedules of major defense acquisition programs lead to more expensive platforms fielded in fewer numbers. The committee's concerns extend to all three key components of the Acquisition process including requirements generation, acquisition and contracting, and financial management.* [75]

In short, the increasing development time of defense acquisitions complicates the DoD's mission in two ways. First, these long development cycles often lead to more expensive and fewer assets being procured. Second, the increasing length of time between the identification of a capability gap and the bridging of the gap leaves soldiers and citizens of the United States vulnerable in the meantime. Thus, it becomes necessary to improve the DoD's ability to not only procure the correct assets, but to do so in a timely manner with minimal risk of purchasing a sub-par asset. It was these paired risks that led to the United States Department of Defense implementing the JCIDS approach in 2004 [71].

*Procurement Since JCIDS Implementation*

Despite the focus on capability-based solutions to problems, defense acquisitions since the implementation of JCIDS have not dramatically improved in timeline or cost-effectiveness. Many programs have been canceled outright, including the United States Army Future Combat System (FCS) and a new Marine Corps' amphibious vehicle. While effort had been applied in the form of CBAs and other capability-related analyses, the new defense acquisitions tended to under-perform or otherwise fail to meet the capability goals [28]. The use of JCIDS did not seem to stem the rise in program time and cost as, according to the GAO, the 96 ongoing defense acquisition programs in fiscal year 2008 were $296 billion over-budget and on average 22 months delayed [28, 76]. More recently, the total defense cost overruns have tallied up to $450 billion by Summer 2017 [26].

Thus, affordability has been identified as a primary driving element for new asset and solution development [49]. Costs also tend to be locked in early in the design process, often through decisions involving the architecture and form of the solution [49, 77]. In this regard, costs are frequently "frozen" before the details of the solution are identified (see Figure 2.7), which can frequently be traced to the mission simulations and capability gap analysis done to inform the requirements [77]. Meanwhile, the Weapon Systems Acquisition Reform Act (WSARA) of 2009 was intended to specifically ensure that appropriate accountability and alternative-consideration was implemented for United States Department of Defense acquisitions [78, 79, 80].

*Costs Committed*

The importance of the EoA (or more specifically, the AoA for military acquisitions) is further underscored by a GAO study which found that "three-quarters of a program's total life-cycle cost is influenced by decisions made before it is approved to start development " [28]. Thus the aforementioned locking-in of cost is tied directly to the conceptual design and architecture of the selected solution, rather than complications that may arise during the detailed design or manufacturing (see Figure 2.7). Notably, the Analysis of Alternatives is the final step before the detailed design and development of a new asset. Thus it has a critical role as the final check and confirmation prior to the majority of the financial and time investment of the procurement system.

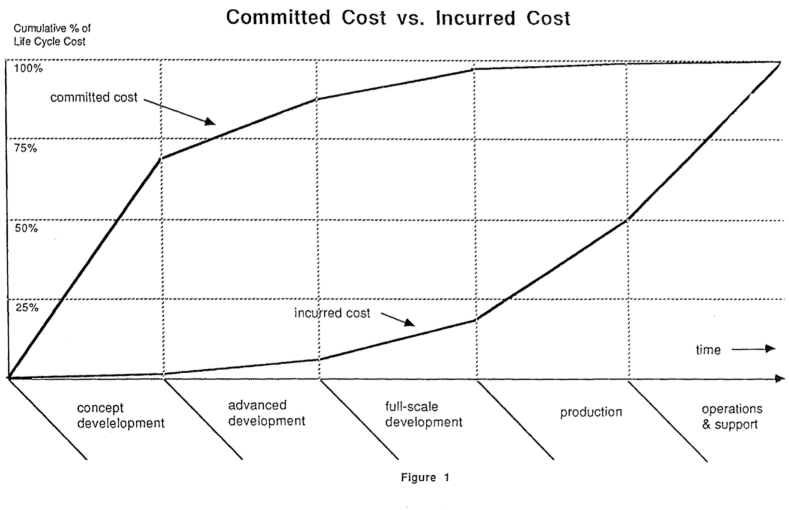



Figure 2.7: Illustration of How Program Expenses Are Committed Versus Incurred (Spent) During a Solution Development Process [77]

### 2.3.2 Incomplete Evaluation of Alternatives (EoA)

DoD processes have a specifically defined stage of JCIDSknown as the AoA which is a variation of the generic EoA [12, 72, 81]. The goal of this stage is the complete comparison of multiple, perhaps dissimilar, solutions to an identified capability gap.

First, issues can arise when comparing assets that will serve similar roles. The most notable recent example is the comparison of light multi-role fighters for the United States Military. The $5^{th}$-generation Lockheed Martin F-35 and its variants will be replacing or supplementing mainstay $4^{th}$-generation aircraft like the Lockheed Martin F-16. However, some tests and mock combat actions between the two aircraft have revealed that the older F-16 can actually outmaneuver and emerge victorious in a Within Visual Range (WVR) dogfight [82, 83]. However, the comparison is arguably flawed because the stealthy F-35 should never allow itself to be engaged in such a form of combat in the first place [82, 84].

Additional considerations also need to be made for evaluating modifications to existing systems, whether currently used by the decision-maker / end-user or not. A major

recent example is the identified need for a new frigate for the United States Navy [26]. In this scenario, there is not only the option to design a new ship from scratch, but also to "Americanize" existing ship designs from allied nations or more thoroughly modernize the ships or designs of the preceding *Oliver Hazard Perry*-class [26]. Attempts have been made to develop more robust mission solutions to identified problems or scenarios, but they are frequently subject to the same issue of shifting requirements [56]. Regardless, improved mission modeling and war gaming could be used to improve the Analysis of Alternatives or any form of Evaluation of Alternatives [1, 2].

Lastly, the solution space becomes even more complex when potential combinations of assets are considered, ranging from a mixed of different manned aircraft, to a fully autonomous or remotely-piloted solution, to the spectrum in-between [26].

*Program Time and Cost Overruns*

During their investigation of DoD procurement under the JCIDS approach, the GAO traced the issues with many of these programs to subpar AoA. Indeed, "programs that had a limited assessment of alternatives tended to have poorer outcomes than those that had more robust AoAs" [28]. As part of the research, the GAO analyzed the 32 defense procurement programs (as of 2008) initiated since the 2004 introduction of the JCIDS approach. What they found was that of those 32 programs, only 22 (68%) conducted an Analysis of Alternatives at all. Of those 22 that conducted AoAs, only 9 (28% of total) conducted what the GAO deemed "sufficient" or "complete" AoAs. After dividing the 32 programs into these three categories based on quality of their Analysis of Alternatives , the GAO determined how delayed or how over-budget these programs were [28]. What they found is summarized in Table 2.2 .

Table 2.2: GAO Study of DoD Acquisition Programs 2003-2009 [28]

| Cost Growth OR Schedule Growth | <10% <7 months | <25% <12 months | >25% >12 months |
|---|---|---|---|
| **Quality of AoA** | | | |
| No AoA | 7 | 0 | 3 |
| "Narrow" AoA | 4 | 1 | 8 |
| "Complete" AoA | 7 | 1 | 1 |

As enumerated in Table 2.2 , the majority of the programs that overran on either cost or time were those that conducted limited Analysis of Alternatives. In comparison, those programs which conducted complete AoA were minimally over-budget or over-schedule. While the conducting of a complete AoA does not guarantee a timely and on-budget procurement process, it does appear to decrease the probability of such overruns. It is of additional note that programs without an AoA often skipped the step due to the limited nature of the acquisition or the existence of significant prior data and analysis. For example, no AoA was conducted during the process of upgrading and replacing Global Positioning System (GPS) satellites, as these systems were proven and the changes to individual systems were of low risk [28]. However, it is also worth noting that in at least three cases, the reasoning behind skipping the Analysis of Alternatives proved faulty as they dramatically ran over-budget or over-schedule.

### 2.3.3 Long Term Effects

The United States Department of Defense has tough decisions to make but the decision process is fragmented and inconsistent [76]. Unfortunately, the DoD has also proven unable to identify appropriate ways of weighting capability importance or in integrat-

ing desired capabilities across joint combat commands [85]. The net result is a clear demand for broad-based capability analyses, but a lack of robust or time-efficient methods to do so.

*Lack of Multi-Mission Considerations*

The weaknesses identified with the AoA contrast with the earlier elements of the JCIDS approach. While the previous acquisition steps encompass a broad-based solution space using current, upgraded, GOTS, or COTS assets alongside non-matériel elements, the AoA is noteworthy for its relatively static and defined mission properties. As noted by Captain Norbert Doerry (USN, ret.) "the problem with this process is that each mission area has its own CBA, yet ships and aircraft are inherently multi-mission" [11]. Additionally, "matériel solutions are developed to specifically address each individual capabilities gap independent of other gaps or consideration of overall fleet or systems architectures."[11]. Similarly, the GAO found that "many of the AoAs that GAO reviewed did not effectively consider a broad range of alternatives for addressing a warfighting need" [28]. Summarizing Captain Doerry's and the GAO's observations, assets are increasingly multi-role and multi-mission, an aspect which is not accounted for during Capability-Based Assessments which seek the solution to specifically identified capability gaps. Thus, even though JCIDS advocates for the use of capability-based approaches, each capability gap is identified and solved in isolation. This has the unintended effect of leading to rigid and poorly arranged use-cases for a complete multi-mission analysis. These narrow use-cases thereby limit the scope of the Analysis of Alternatives by artificially constraining potential competing solutions to a specific architecture.

By failing to properly address the multi-mission nature of new assets or evaluating the assets' robustness to changes in mission, the resulting procurement often generates a suboptimal solution beyond the design mission. Conversely, if the proposed asset is to be multi-mission by definition, then the resulting degree of specificity in the re-

quirements contributes to requirements creep and thus extended design, development, testing, and deployment time and costs [27, 76]. Of these two scenarios, requirements creep is the more common, as evidenced by the compounding delays and cost overruns associated with the Joint Strike Fighter program. In this case, the need to satisfy the disparate requirements of the United States Air Force (USAF), United States Navy (USN), and United States Marine Corps (USMC) all contributed to increasing complexity of the system [82, 84]. In general though, constantly shifting requirements are a major source of cost and time overruns as the system shifted from fixed-price contracts [26]. Changing requirements, potential opponent's actions, and even the laws of war further complicate a decision-maker's ability to find a robust solution [56].

*Timeline*

The weaknesses of the Analysis of Alternatives aside, the ability to conduct such studies effectively are constrained by a mix of short timelines and lack of guidance. A frequently cited problem with defense acquisitions are that program sponsors select the preferred solution too early in the design process. There was also an artificial limiting of the time available to conduct an Analysis of Alternatives to meet deadlines. In fact, many AoAs were conducted concurrently with the design work and other actions that would otherwise provide data for the process [28]. Thus any solution that seeks to improve AoAs must do so in an easily extensible manner which does not overly complicate the procurement process or meaningfully contribute to program timeline.

Once more, improved mission modeling can help address these shortcomings in the AoA and EoA processes. As highly detailed models are often time-intensive to operate, Surrogate Models may hold the key to improved understanding of solutions [19, 41].

## 2.4 Overall Research Objective Flow

Given the overall objectives set forth by Capability-Based Assessment and similar methods, there is an increased demand for mission modeling. Furthermore, these models are not performance-focused, but rather capability-focused. Lastly, the higher-level campaign, System of Systems, and life-cycle models all require information provided by lower-level mission simulations as illustrated by the "Pyramid of Simulation" in Figure 1.3. Mission simulation is the key to not only identifying non-matériel solutions, but also understanding the full ramifications of decisions made during the solution generation process.

These objectives and requirements set forth a clear demand for more informed approximations of simulated missions and asset capabilities. These approximations are in turn inspired by the developments in (Engineering) Design Space Exploration and Surrogate Models which can capture the tradeoffs and effects at a lower computational expense than running a full-fidelity simulation. There is a necessary sequence of research endeavors to enable Mission Space Exploration and the creation of a trade space. These steps revisit principles first detailed in Chapter 1 and articulate the Research Objectives outlined previously.

Overall Research Objective

Develop new approximations of mission-level behaviors, including not only Mission Architecture but also decision-making elements, which can then be propagated and infused into higher-level (campaign or life-cycle) models and simulations

### 2.4.1 Identifying the Simulation Means

When developing model approximations, there is a need for a baseline dataset. Various methods exist to obtain this data, with historical precedent preferring SME estimations, combat data, or information from real-world exercises [13, 17]. Consequently, there is a tension between the historical precedent and more modern computer M&S based methods. This dichotomy directly informs and contextualizes Research Objective 1.

Research Objective 1

Identify the most appropriate means of obtaining the results from a variety of mission variations

This research objective also maps readily onto the first research question posed. For the given objective of sampling and simulating across a mission space, the necessary model must be identified and developed. As the **Stochastic Agent Approach** aims to be code agnostic, it is important to identify the form and architecture of the data generation method, not a specific tool.

### 2.4.2 Exploring the Mission Variable Space

Once an appropriate data-generation method is found, the next major obstacle is generating data from that method. Without an effectively sampled dataset, there is no ability to create a computationally efficient approximation. Thus Research Objective 2 is focused on the Mission Design Space and its resultant complications.

Research Objective 2

Identify the most appropriate means of sampling across the Mission Design Space

This objective aims to address the Mission Design Space directly and seeks to properly control an otherwise expansive and massive variable space. A large variable space

is more difficult to sample satisfactorily and past work on Mission Space Optimization and agent AI learning (and thus optimization) have bypassed the issue through deterministic and constraint-focused methods.

### 2.4.3 Crafting an Approximation for Mission-Level Performance

With a model architecture identified and the Mission Design Space articulated, the final step of the process is to generate the necessary computationally-efficient approximations. Much like the Lanchester Equations or a engineering design tradeoff environment, the approximation is intended to increase the speed of simulation through a simpler representation of the complex system [13, 19, 41]. While the Lanchester Laws are specifically defined differential equations, a general approximation method is more amenable to incorporating the anticipated new interactions.

Surrogate Models in general are mathematical approximations of a higher fidelity model [39]. These equations appear in various forms ranging from polynomial regressions to Artificial Neural Networks [19]. The methods are necessary as it is time-intensive to generate a Mission Space Exploration using a higher fidelity approach (be it a computer model or full-scale exercises). Consequently, a broader understanding of the trade space is obtained through the fitted approximation which can filter for the dominant effects of the various inputs. This desire to generate a superior mission-level approximations informs Research Objective 3. However, there are several stages to the Surrogate Model development process.

Research Objective 3

Develop a higher efficiency approximation of the Mission Design Space to enable effective integration with higher-level models

The first step is to address simulation stochasticity. Higher-fidelity mission models are stochastic by definition and as such, will generate a distribution on the various

output metrics of interest. Consequently, a minimum replicate number is needed to confidently capture those distributions in a reasonable amount of time.

The second step is to find a Design of Experiments of an appropriate form and size for the Mission Design Space. There are multiple DoE forms to consider, each with their own strengths and weaknesses. Furthermore, these samplings vary in size and a certain number of design points will be needed to craft an appropriately accurate Surrogate Model. The main result is a method for selecting an appropriate DoE archetype, not a claim to a universally relevant type.

The final step is to actually craft the Surrogate Model. As would be expected, there are numerous archetypes of both fit and form. The main result is a method for selecting an appropriate Surrogate Model archetype, not a claim to a universally relevant type.

### 2.4.4 Conclusion

This research concludes by generating new observations of the trends and relationships between various model inputs and outputs in a notional context. The resulting approximation equations can effectively function as a "black box" for future users, though the trends and relationships have been informed by a higher fidelity model. "Sanity checks" will be performed on these relationships by observing the influence of various input parameters and comparing to common-sense logic or precedents like the Lanchester Laws.

Overall, the **SAA** describes the means for understanding a complete Mission Design Space from mission identification through final model development. Notional case studies are used to illustrate and contextualize each research question. Though the exact conclusions might be most appropriate only for the case studies considered, the overall method and process to reaching these conclusions should be universal and code agnostic.

# CHAPTER 3

# IDENTIFYING AN APPROPRIATE SIMULATION APPROACH

> The required analysis combines physics-based modeling of the individual war fighting units, realistic cost engineering/estimation, and rigorous operations analysis.
>
> - Capt. Norbert Doerry and Howard Fireman [11]

## 3.1 Major Mission Simulation Approaches

The first research sub-objective is to identify an appropriate means to run the different mission simulations required for the proposed Mission Space Exploration and sampling. Due to the breadth of computational, historical, and proprietary capabilities of mission simulation, evaluating and comparing the various architectures is necessary. This is especially relevant as the **Stochastic Agent Approach** aims to be toolset or code agnostic. Furthermore, there are both different forms of mission simulation and internal variations within those archetypes. As such, the research starts with Research Question 1.1:

> Research Question 1.1
>
> What is the most effective simulation archetype for conducting a Mission Space Exploration?

Mission simulation is a means to evaluate the conduct of and coordination between units. The objective is to test an asset through an approximation of its real-world use [6, 7, 12]. Mission simulation ranges from basic analysis of preliminary aircraft designs to full-scale prototype testing. Some mission simulations are massive international exer-

cises such as “Operation Red Flag” or Rim of the Pacific Exercise (RIMPAC) [86, 87].

Mission simulation varies in fidelity as well. Assumptions can be made in all forms of non-physical simulation while even full-scale physical activities might avoid the hazards of live-fire exercises. This diversity in simulation forms and complexity requires the identification of an appropriate architecture for Mission Space Exploration. The desired Mission Space Exploration and sampling is on the tactical level as that form is the abstracted for strategic and operational analyses. While this doesn't explicitly limit the form of mission models considered, this form of model does have its own set of requirements and desired properties.

### 3.1.1 Active and Passive Representations in Mission Simulation

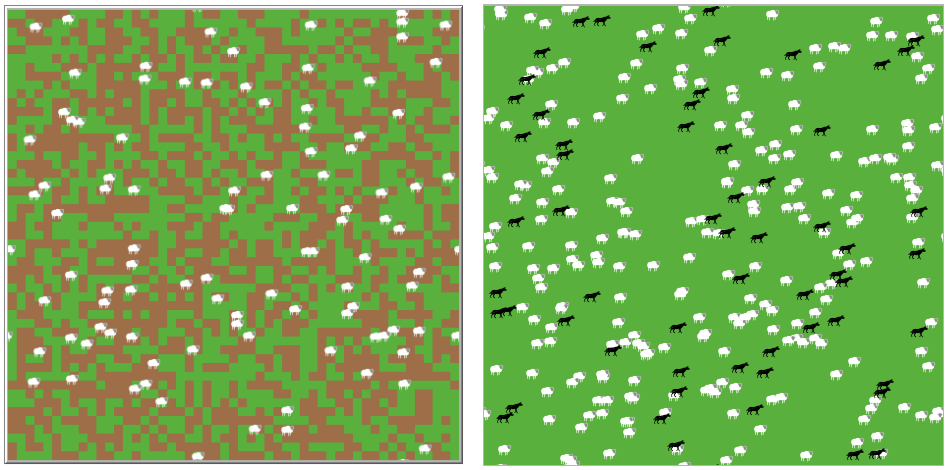
When simulating missions, there is a major consideration of the decision-making fidelity of the units or factions involved. Each side in a simulation can be passive or active with regards to decision-making or intelligence within the simulation. However, the level of entity “activity” does not influence the means of simulating that entity. The same can be said for levels of fidelity or computational intensity. In short, whether a unit or faction is active or passive does not necessitate a particular form of model. Depending on the desired observations and insights, active or passive formulations may be more relevant.

A passive faction is one which acts deterministically and has known and fixed parameters at the start of the simulation. A passive defending faction would be akin to a static air defense system with its assets rooted to particular locations. Many forms of wargaming and human-involved simulations have passive or static opponents. A passive attacking or motive faction would be an asset following defined waypoints such as a simple, deterministic simulation of an aircraft's flight plan. Such a form of passive, motive mission modeling can be seen in National Aeronautics and Space Administration (NASA)'s Flight Optimization System (FLOPS) model where the designed aircraft is

flown through a prescribed mission plan [37].

An active faction is one which acts stochastically (at the simulation level) or with a degree of simulated decision-making. Active factions are often associated with ABMs, discussed later in this chapter. An active defender would be one which would react and redeploy its units in response to stimuli (e.g. send reinforcements towards inbound enemies). An active attacker or motive faction would make decisions and changes to their actions based on the detection of opponents (e.g. maneuver around defenders). For example, a path-finding algorithm through a region would have an active motive unit seeking that optimal path. Various forms of video games have utilized active computer-controlled units which act and react to the players actions [44, 88, 89, 90]. Though perhaps deterministic in their reactions (i.e. if the players does *A* the computer will always react *B*), these are still changes to unit actions in the midst of the simulated mission.

For a comparison example, Figures 3.1a and 3.1b illustrate different predator-prey-resources Agent-Based Model formulations derived from the NetLogo ABM tool [91, 92]. In this case, there are active predator and prey entities moving around and reacting in the model. There are also passive grass entities which have no form of decision-making and simply exist to grow, be eaten by prey, and grow again.



(a) An Example Active-Passive Model with Active "Prey" Interacting with Passive "Grass"

(b) An Example Active-Active Model with Active "Predators" Interacting with Active "Prey"

Figure 3.1: Example Illustrations of Notional Agent-Based Models [91, 92]

In the context of this research, the focus of the simulations will be semi-active scenarios where one faction has active agency in the model. This is intended to avoid the "Curse of Dimensionality" which would arise due to the interactions of two active forces [93]. However, being active or passive does not preclude variations in deployments and capabilities between simulation runs. A passive defense system can still be varied in order to cover the spectrum of possible pre-combat actions. In the case of the simulations used in this research, the attacking faction is the only one with a potential for agency and active maneuvers. Conversely, the defending faction is a passive collection of entities and plans.

It should also be possible to simulate passive-passive scenarios through active-passive models. With an active attacking force probing, optimizing, and evaluating through the complex mission space, the resulting course of action can be mapped into a passive mission plan for future use. However, this requires a truly passive defending force (i.e. identical deployments and paths). Traceability of the actions by an active side enables the create of mission plans for future simulations of a passive version of that same side.

### 3.1.2 Variations on Mission Simulation

Generic mission simulation spans a variety of forms and functions. From the basic Breguet (aircraft) Range Equation to test flights, most forms of mission simulation are "isolated" and conducted for singular systems [37]. These forms of mission simulation rely on the physics of the problem and often avoid interactions between entities in the name of simplicity and scalability. Interactions, should they be simulated, can vary from passive detection and acknowledgment of other entities in the simulation, to active coordination and information sharing. Furthermore, destructive interactions can exist as entities can be damaged or destroyed through engagements with other entities. When destructive interactions are employed, mission simulation is more appropriately labeled combat simulation.

Combat simulation can be broadly summarized as the modeling and simulation of military engagements involving two or more entities. This form of modeling is not simply a physics-based simulation of a vehicle, namely because it often includes both entity decision-making and the simulation of the munitions involved [15, 16]. As such, combat simulation not only has to include the physics of the primary assets involved, but also their reactions to changing situations and the physics of any ammunition, missiles, and other sub-entities involved in the scenario [56]. Numerous combat simulation archetypes exist, such as the Advanced Framework for Simulation, Integration and Modeling (AFSIM) environment seen in Figure 3.2.

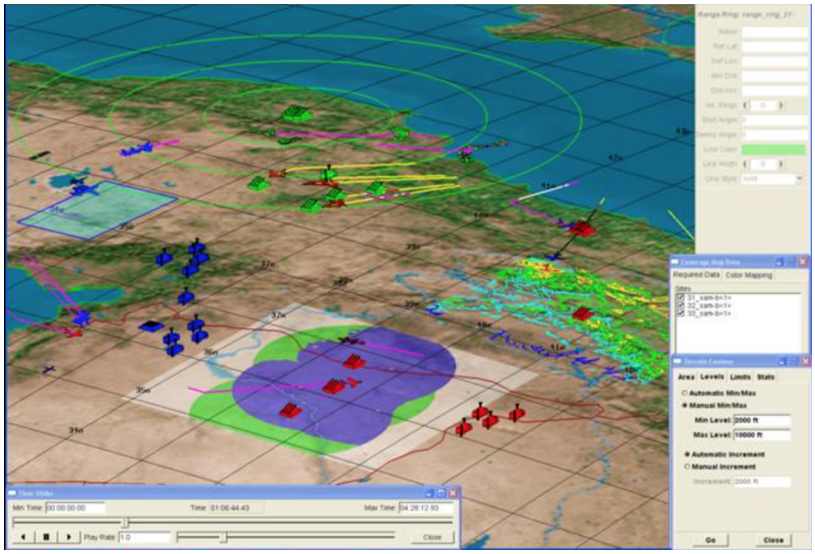


Figure 3.2: Example of the AFSIM Modeling and Simulation Framework [94]

For a military-focused Mission Space Exploration, combat simulation needs to be considered within the broader context of mission simulation. Engagement, attrition, and coordination are fundamental to military missions and as such, any considerations in an exploration requires acknowledgment of the physics, decision-making, and engagement. More generally war games are a form of integrated combat simulation, usually involving human participants to make decision for each side [23, 24, 95, 96, 97].

Wargaming is an activity which is intended to simulate human decision-making and problem solving specifically in the context of combat and warfare [56, 97]. Human players control at least one of the simulated sides and make decisions and actions as though the simulation were real. Observations can then be made to identify which actions led to success, what elements may be lacking, or whether changes need to be made to military systems and deployments. These war games can be generally divided into 6 categories along two axes of concern: the level of structure and the objective, as outlined in Table 3.1 [35].

Table 3.1: Categorization of War Games [35]

| | Creating Knowledge | Conveying Knowledge | Entertainment |
|---|---|---|---|
| Unstructured Problem | Discovery Games | Education Games | Role Playing |
| Structured Problem | Analytic Games | Training Games | Commercial (Board) Games |

For the purposes of this research, the most interesting form of war games are Discovery and Analytic Games. The objective of the **SAA** is to generate knowledge and understanding of mission space options and alternatives, and then generate approximations of that knowledge for faster future query. There is also an element of "conveying knowledge" worth considering because analysis gains additional meaning when it can be applied or used to inform decision-makers. It is worth noting that many of the considerations which go into evaluating war games can be considered independent of the exact means of implementation.

Regardless, wargaming and mission simulation both have issues with validation in that even a well-verified model might miss key features or aspects. There are also accusations that money has been wasted on large and complex simulation suites which suffer from incorrect assumptions [95]. In the end though, the goal of all forms of mission simulation and wargaming is to understand the logic and flow of things rather than fully validated results [23, 24, 95]. Especially in this context of methodology development, there is a need to compare various approaches through the same model rather than claiming perfect model representation of the real world.

From here on, "wargaming" and variations on the term will specifically apply to the human-in-the-loop aggregate-level studies as described in the terminology section. The

focus on aggregate (rather than individual entity behavior) performance and human-involved decision-making are hallmarks of the field dating back to the origins of military tactics and strategy [96]. The addition of computational models and computer-controlled opponents have supplemented and in some cases supplanted the humans involved in wargaming and as such, the field has expanded beyond its original scope. To differentiate the terminology, "mission simulation" will more broadly apply to the field of wargaming which can be conducted through a variety of means (human-in-the-loop, physical testing, computational models, etc.).

### 3.1.3 Discrete Event Simulation

Discrete Event Simulation is a modeling approach where the assumption is that the outcome is not very sensitive or dependent on the precise means used to achieve that outcome [98]. As such, the approach effectively encapsulates how many engagements and missions are represented at the campaign level of simulation. For the highest-level simulations, these mid-level mission approximations consist of "black box" inputs, usually of Mission Architecture Variables, and generate the various metrics of interest such as attrition numbers or resource costs [13, 99].

The main limitation of the approach is the assumption of fixed mission-level behaviors and logic, exemplified by the relatively lack of engagement or mission tradeoffs in a Mission Architecture context. However, improved traceability of the equations which inform the discrete event "black box" is the end-goal when developing the **SAA**. New approximations of these in-mission actions, with tradeoffs possible through new entity-level inputs, are exactly the types of superior understandings desired for capability-based methods. When missions are small-scale, individual decisions are of more influence and knowledge of those tradeoffs is more of interest. Thus representations like the Lanchester Equations lose utility as engagement results cannot be averaged or amortized over such few involved units [14, 17].

### 3.1.4 Terminology

*Human-in-the-Loop*

The United States Department of Defense Modeling & Simulation Coordination Office defines "human-in-the-loop simulation" as a "simulation and simulators that employ one or more human operators in direct control of the simulation / simulator or in some key support function" [6]. More generally, Human-in-the-Loop combat models are those that require a human decision-maker at one or more stages of the simulation [15]. This form of combat model has its advantages by mitigating or removing the uncertainty associated with the inability to precisely model human decision-making with a purely computer model. For example, a wargaming exercise through a video game like interface would constitute a human-in-the-loop combat model while a more conventional (and automated) computer simulation would not.

The primary disadvantage of such a model is the longer runtime of each simulation. Because each iteration requires a human being to make and input decisions, it must also output information about the simulation in an easily understood manner to facilitate such decisions. As such, computation time is spent rendering assets and waiting for the human to respond. Conversely, an automated model could make decisions in a fraction of the time, albeit with cost inherent in its imperfect representation of human decision-making.

*Entities, Assets, and Agents*

For the purposes of this thesis, two terms will be utilized to define the soldiers, vehicles, and other physical components of the military simulation. These two terms are "asset" and "entity." An asset refers to the actual physical unit being simulated. An entity refers to the model of a physical asset (or assets) in the combat simulation. For example, if the simulation is for a tank battle between two opposing forces modeled at the platoon-

level, then the assets are the tanks themselves and the entities are the platoons of tanks involved. The flowchart of this relationship is seen in Figure 3.3.

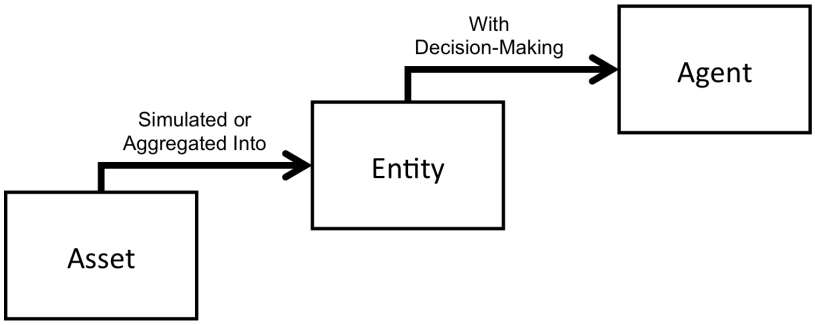



Figure 3.3: Assets Relating to Entities Relating to Agents

The United States Department of Defense Modeling & Simulation Coordination Office defines an "agent" as "a computer system capable of autonomous action, of deciding for itself what behaviors are needed to satisfy its design objectives, and capable of interacting with other agents" [6]. However, "entity" and "agent" are not synonymous. The primary difference being that an entity does not require any form of "intelligence" or decision-making capability while an agent does. That difference is maintained in this document.

In summary, at the highest fidelity levels of physics and decision-making simulation, every agent is assigned to a singular entity which represents a single asset. However, as various levels are abstracted or defaulted, these terms can diverge. At a more aggregate level, every simulated entity might encompass numerous assets (whether homogeneous or heterogeneous). Furthermore, that entity might not have any form of automated decision-making, either because it is directly commanded by a human player or because it acts according to an inflexible pre-programmed routine (e.g. a patrol route).

*Deterministic and Stochastic*

Whether a model is deterministic or stochastic depends on how repeated experiments are handled. For each run of a deterministic model, the resulting simulation is identical.

For each run of a stochastic model, the resulting simulation changes due to the use of random variables.

Deterministic models are exemplified by the aforementioned Lanchester Equations as the simulation is driven by a series of well-defined differential equations. Deterministic combat models require the assumption that each round of combat results in a fixed damage amount (or attrition) to the entities involved given the same series of input conditions [15]. Not all deterministic models are necessarily aggregate however. If an entity-based model has no elements of variability in responses (e.g. purely deterministic decision-making or passively pre-planned travel paths), then it too can be ultimately deterministic.

Stochastic models notably include random variables and thus some form of "dice roll" within the combat simulation. For example, entity $A$ has a 50% chance to kill (Probability of Kill ($P_{kill}$)) a given target. Each time entity $A$ then engages a target, a random number generator determines the outcome based on the relative values of the random number and the $P_{kill}$ value. The use of a stochastic model is more relevant in small-scale engagements where the loss of individual entities has a more dramatic effect on the strength of a given side [15]. Additionally, having a stochastic model requires replication of experiments to ensure the elimination of outliers and to avoid the reporting of incomplete data due to the probabilities involved [14, 100, 101]. In short, using a stochastic model has inherent variability in the outputs of the simulations, which must be reported for a valid discussion of those results.

### 3.1.5 Simulation Form: War Games

As mentioned previously war games and wargaming are an approach to mission simulation whereby human decision-makers are used to simulate the commanders and participants of one or both sides. Frequently these simulations are conducted at the campaign or aggregate level and use approximations to resolve combat between entities [8, 95]. An

example of the view available to the human participants is illustrated by Figure 3.4.

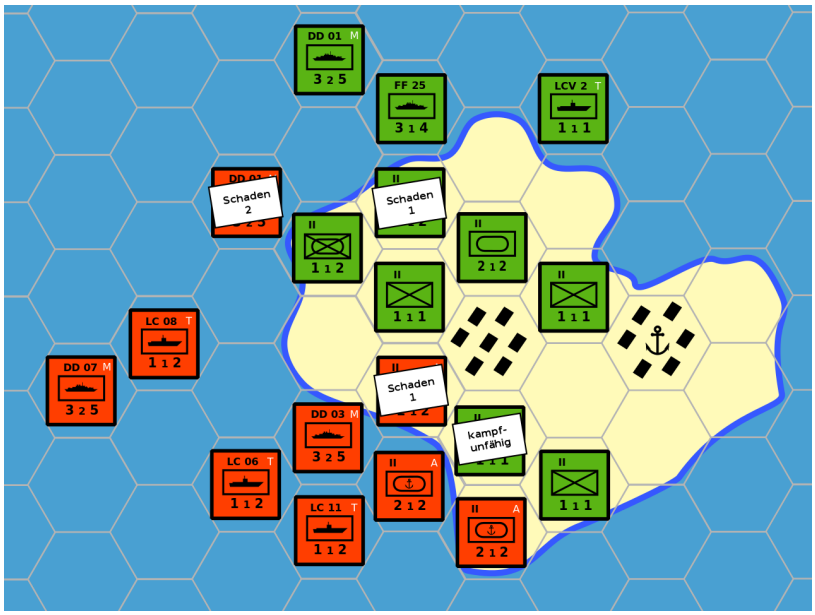



Figure 3.4: Example Illustration of the Map for an Aggregate-Level War Game [9]

*Description*

Wargaming methods involve human participants conducting the majority of the decision-making and moderating. Retired USN Commander Phillip Pournelle defines a war game as a "dynamic representation of conflict or competition in a synthetic environment" [35]. While computer models are now more frequently used to supplement the human players, the participants are the primary solution drivers [8, 95]. This human-in-the-loop simulation method, while accurate with regard to how decisions are made, has a low speed of simulation compared to other approaches [56].

War games usually operate most effectively at a longer timescale (e.g. 1 turn = 1 year) so as to not overburden the participants or delve too deeply into the specifics of individual engagements (see Figure 3.5). Furthermore, most war games utilize subject-matter or decision-making experts for the purposes of generating problem understandings [97, 102]. War games at the strategic level focus on addressing and understanding the forms of uncertainty introduced by System of Systems and policy-level effects. More specifi-

cally, there are the uncertainties associated with conflict-initiation, victory conditions, and the "laws of war" [56, 103]. Figure 3.5 illustrates how the Army War College breaks down the hierarchy of combat simulation with the traditional levels of wargaming outlined in red.

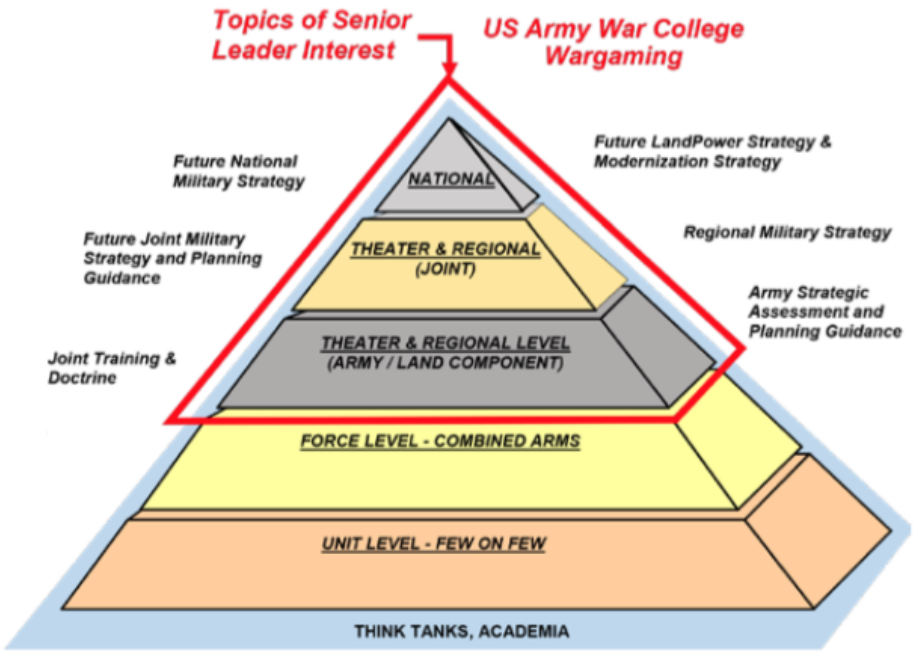



Figure 3.5: The Pyramid of Combat Simulation Levels with Traditional War Games Outlined in Red [97]

Revello, et. al. developed an automated and computerized version of a naval warfare war game which is an good example for the archetype, illustrated by Figure 3.6 In that model, the strategic and operational nature of the war game is apparent with individual unit movements and engagements conducted in 12-hour cycles. Furthermore, asset placement is limited to larger regions rather than a detailed three-dimensional map. The assets are also grouped into higher-level entities, though red assets are individually tracked. Combat was decided through $P_{kill}$ metrics empirically defined based on the involved assets and level of knowledge. Lastly, there is specific concerns for operational

effects such as resupply and maintenance [56].

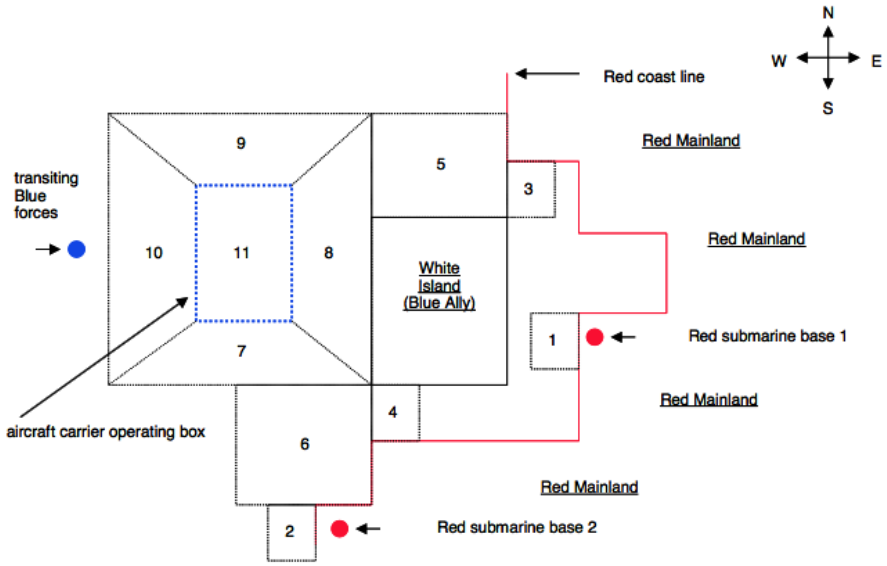



Figure 3.6: The Strategic View and Map for an Example War Game [56]

*Strengths*

War games can cover a variety of fidelities and timescales and by definition have a complete representation of decision-making (at least for the side using human players) [103]. Also, by employing and leveraging human ingenuity, wargaming exercises can identify creative (or emergent) actions and decisions [8]. A notable example of this can be seen in war games conducted in the context of a hypothetical amphibious assault wherein the defending force sorties its submarines not to engage any particular target, but to simply exist as a threat that may encourage an "overreaction" from the attacking force. This tactic, though played out in a course at the United States Naval Academy, inadvertently mirrored the actions taken by an Argentine submarine during the Falklands War of the 1980s [8].

*Weaknesses*

A major issue with war games is that the current methods are less capable of accounting for novel means of combat or new forms of assets [8, 56, 103]. This inflexibility is driven by the required availability of datasets to inform the models necessary to approximate lower-level actions. If no empirical or extrapolated data is available, then the relationships between assets (whether collaborative or combative) is uncertain [103]. A further complication arises when particular strategies are either implicitly or explicitly encouraged or eliminated. This was seen in the Imperial Japanese wargaming leading up to the Battle of Midway when the junior officers playing as the "Americans" were banned from utilizing a series of fleet and aircraft maneuvers, which ironically turned out to perfectly mirror American actions at the later battle [8].

By operating on the strategic and operational levels, war games also include more extensive concerns for resource management, timeline, and out-of-combat actions of assets and personnel [56, 103]. While these complex constraints at the strategic and operational levels affect actions on the mission-level, they are themselves usually not simulated at that fidelity [56].

*Initial Evaluation*

Given that the focus of this research is on mission and engagement-level effects, wargaming does not appear to be a good option for the simulation method. The approach is optimized for human-in-the-loop and strategic situations, while a Mission Space Exploration is much more focused on the tactical level. Furthermore, it would be unreasonable to use an empirical and aggregate model to generate an approximation equation, which is itself empirical and aggregate.

### 3.1.6 Simulation Form: Physical Testing and Exercises

Physical testing is the most realistic simulation method because it involves the direct participation, use, and coordination of assets and personnel. Physical testing of singular vehicles is commonplace and frequently required for full evaluation or official certification of designs [104]. Training of personnel can also be seen as a variation on physical testing where the focus is on rehearsing and preparing for various contingencies while enhancing interoperability and knowledge [86, 87]. The United States Pacific Fleet hosts the semi-annual RIMPAC exercises (see Figure 3.7) and describes these instances of mission "physical testing" as:

> *A unique training opportunity that helps participants foster and sustain the cooperative relationships that are critical to ensuring the safety of sea lanes and security on the world's oceans* [87]

Regardless of the end-goal, all forms of physical testing are related by the use of actual entities, assets, and personnel to conduct and rehearse missions. However, this archetype of mission simulation implicitly reflects empirical data from past missions or engagements.


Figure 3.7: Information and Context for the 2016 RIMPAC Exercises [9]

*Description*

Physical testing involves real assets and personnel conducting actions in keeping with anticipated missions or operations. These activities are frequently oriented around training personnel in the most realistic context possible without actual engagement occurring. Specialized facilities, ranges, or regions are used in order to best enable training and testing as well as provide an approximation of the terrain and environment expected in actual missions [87].

Using experiences from past missions and engagements is a natural evolution of the desires of planners to develop more informed understandings of the scenarios they are faced with. The military theorist Sun Tzu even noted that one should "rouse him, and learn the principle of his activity or inactivity" [105]. In other words, learning from past actions (or failures) should inform future decisions and plans.

Physical testing or field use has also proven critical in identifying new capabilities or particular strengths of assets. As discussed previously, clever maneuvering and coordination was used by the Americans in World War II to compensate for the comparatively poor performance of their fighter aircraft [3, 10]. More recently, the American F-22 has been adapted from its intended Air Supremacy role to include small-scale strike capabilities via the SBD weapon system [52]. Notably, the United States Air Force has used allied military experience to explore new off-design capabilities and mission plans, specifically in the context of the Israeli military [18].

More generally though, large scale training maneuvers are conducted both by nations individually and in joint operations contexts. The RIMPAC and Operation Red Flag exercises are major, regular examples of these mission testing operations [86, 87]. Data from these exercises can in turn be used to populate empirical or aggregate models [13].

*Strengths*

The primary strength of physical testing is that there are minimal approximations compared to other evaluation methods. There are perfect representations of the problem physics and of decision-making by mission-involved personnel. Therefore, any generated approximations are as close to fully validated as possible.

*Weaknesses*

However, physical testing is expensive, dangerous, and can only be run in real-time. Being a real-world approximation of a mission also has a few insurmountable drawbacks. Firstly, the use of physical assets and personnel means that prototype or proposed designs cannot be incorporated. Secondly, the real-time nature of the activities limits decision-makers' abilities to run a multitude of cases in a controlled manner. Lastly, there is limited ability for live-fire rehearsals due to the costs, risks, and complications associated with using live ordinance in the presence of (or perhaps directed at) allies.

*Initial Evaluation*

While physical testing is of immense use in final preparations and evaluations of mission plans and ideas, it is not well suited to collecting the large dataset required for a complete Mission Space Exploration. The time to arrange the required exercises is extensive. Furthermore, the dangers associated with any real-world testing further discourages a large number of iterations due to the increasing cumulative risk of incident. However, the data generated from these activities is helpful for model calibration and validation, as evidenced by the Lanchester Laws themselves.

### 3.1.7 Simulation Form: Computer Modeling and Simulation

Computer Modeling and Simulation is a relatively new approach to simulating mission activities. This development is primarily driven by the recent advent of computer sys-

tems fast and complex enough generate meaningful results [60]. The United States Department of Defense Modeling & Simulation Coordination Office defines a "computer simulation" as "a simulation that is executed on a computer, with some combination of executing code, control / display interface hardware, and, in some cases, interfaces to real world equipment" [6].

*Description*

Computer M&S is using a numerical simulation to approximate real-world actions [96]. The series of models and resulting simulation suite can capture a variety of properties depending on the desired fidelity. Some computer models are specifically tied to particular physical phenomena such as aerodynamics or structural analysis. Other models are much more focused on the decision-making and interactions between simulated entities as seen in computer-assisted war games or Agent-Based Models.

Computer models of missions can also be considered automated, and frequently higher-fidelity, forms of the war games already played with human participants. Automated opponents can be trained and developed using evolutionary and machine-learning algorithms. The key with these computer models is that "emergent behavior" can help illustrate novel solutions or highlight potential strengths and weaknesses of proposed solutions [56]. The United States Department of Defense Modeling & Simulation Coordination Office defines emergent behavior as "a behavior or property that appears when a number of simple entities (agents) operate in an environment, forming more complex behavior as a collective" [6]. Emergent behavior is only possible in automated mission simulation as the large number of cases or elapsed time allows dynamic steady-state or other conditions to emerge.

*Strengths*

The major advantage of a computer simulation is that it usually runs faster than real-time. Furthermore, any form of asset or mission can be approximated within reason by relying on first-principles physics. With sufficient work, new regions and assets can be incorporated into the model even if they only exist as preliminary designs or ideas [38].

It is also easier to program a mission or flight plan into the computer model than to completely retrain personnel, thus enabling a degree of exploration and evaluation not easily possible with human participants. Another strength is that computer models have a broad range of possible fidelities ranging from empirical approximations to high-fidelity physics. They can also scale from aggregate-level understandings (similar to a war game) to detailed individual unit actions (similar to a small-unit, full-scale physical exercise).

Ultimately, the most powerful capability offered by a computer-based simulation is the ability to run a multitude of cases in a reasonable amount of time [56]. This enables an exploration and query of the massive solution space which exists in the context of mission simulation.

*Weaknesses*

The weaknesses of computer models lie primarily in the efforts needed to develop, verify, and validate these models. For a singular mission simulation, both war games and physical testing usually have the data or tools already available for the simulation while a computer model may have to be developed from scratch.

Additionally, computer models are ultimately only approximations of real world physics and behaviors. Though physics can be approximated to a high fidelity, there are effectively no "universal truths" when it comes to simulating human decision-making [44]. Consequently, many efforts for computer models either have relatively simple decision-making routines or are attempts to "teach" a computerized entity or battle manager to

behave optimally or more like a human [45, 88, 90]. Furthermore, computer models incur nonlinear time and complexity increases as additional design variables are introduced [38].

*Initial Evaluation*

Computer models are the most promising simulation archetype because they do not have the same "show-stopping" issues of physical testing or human-in-the-loop war games. The process can be easily automated, which facilitates rapid simulation and evaluation of different missions. Furthermore, computer models are much more adjustable due to being entirely artificial simulations. The decision-maker or simulation operator has deep control over the models and thus can add variables or cases much more easily than in the other methods.

## 3.2 Selecting the Mission Simulation Architecture

### 3.2.1 Requirements

To enable decision-making on the simulation architecture, a set of requirements needs to be articulated. These requirements are identified based on past mission simulation approaches and the needs of a solution space sampling, variable space exploration, and eventual Surrogate Model development.

*Account for Disparate Asset Architectures*

The first of two major requirements for a Mission Space Exploration approach is the ability to simulate a variety of possible assets used in a proposed solution. The simulation cannot be limited to a singular asset type otherwise an exploration cannot occur.

All three simulation archetypes can account for disparate assets to some degree. However, physical testing has direct access to these assets while a computer model can

theoretically include any asset given sufficient data. A war game is more limited in applicability due to the more aggregate nature of its fidelity. While differences between assets can be accounted for, they are often done in a lower fidelity manner than the other approaches.

*Account for Disparate Mission Architectures*

The second major requirement for a Mission Space Exploration approach is the ability to simulate a variety of possible mission architectures. The simulation cannot be limited to a singular mission plan or archetype. Simulated entities need to have variable mission plans, paths, logic, etc. in order to explore the variety of mission possibilities.

All three simulation archetypes can account for disparate mission design points to some degree. A computer model, as before, can theoretically include any mission plan given sufficient data, even if the mission plan is foolish, dangerous, or abnormal. Physical testing is similarly capable, but is limited due to participant training or the potential dangers of trying off-design missions. Furthermore, the hazards of physical testing requires that an evaluated mission remain safe and within the bounds of participant training. A war game is once more limited in applicability due to the more aggregate nature of its fidelity limiting the detailed understanding of engagement-level activities. Similar to physical testing, participant training may skew the results.

*Scaling with System Complexity*

The selected mission simulation form must scale well as simulated assets increase in complexity. If the approach cannot account for variations between assets and their subsystems, then a significant amount of understanding is lost.

Both a computer-based approach and physical testing become more difficult to run efficiently given increased system-level complexity. However, the lack of fidelity in a war game means that it is frequently unable to address at all the more detailed variations or

adjustments in system performance.

*Scaling with System-of-Systems Complexity*

The selected mission simulation form must also scale well as simulated entities interact with one another in a System of Systems context. As almost all missions are conducted in a coordinated manner between systems, accounting for SoS interactions is of paramount importance.

Both a computer-based approach and physical testing become more difficult to run efficiently given increased SoS complexity. However, once again the lack of engagement fidelity in a war game means that it is frequently unable to address more detailed variations in mission execution.

*Account for Existing Asset Performance*

For a basic Mission Space Exploration, all currently available physical assets need to be incorporated. Fortunately, all three simulation approaches can address assets with sufficient data to inform their component models (or physically exist and are available for a full-scale test).

*Account for Proposed Asset Performance*

For a full-scope Mission Space Exploration, near-future physical assets also require inclusion. These systems may also include minor modifications to existing units.

Unlike incorporating currently existing systems, the three simulation archetypes vary dramatically in their ability to include in-development assets. A computer model is the most capable in this department because, as mentioned before, data and estimations can inform a physics-based model which can then sufficiently approximate performance of any system. A war game can somewhat include in-development systems because the lower-fidelity of the approach allows for the approximation necessary when

performance is uncertain. However, physical testing cannot account for such prototype or proposed systems simply because an operational asset is unavailable. A flight test of a proposed mission cannot occur if the system only exists as an engineering drawing.

*Fidelity of Simulated Decision-Making*

For a proper Mission Space Exploration, simulated decision-making is central to legitimacy concerns. Especially with regard to combat modeling, decision-making of both allied and opponent entities is necessary. Furthermore, it is unlikely that a complex mission will be conducted deterministically (in a passive-passive situation) in real-life. Compensation for variations in environmental conditions and asset properties is required to ensure mission success and model verification.

As physical testing and war games both include human participants, there is a perfect simulation of human decision-making. For a computer model, a function is required to approximate these decisions and their effects. However, because system operators often rely on training to inform their decision-making, a decent computer representation of human operators in a mission is more possible than in more nebulous contexts.

*Experiment Setup Time*

There will always be some time required to set up a Mission Space Exploration. However, decreasing this setup time will naturally accelerate the analysis and thus provide necessary information in a more timely manner.

As would be expected, physical testing takes the longest to set up due to the amount of personnel, assets, and logistics required. There is a reason that large exercises like Operation Red Flag and RIMPAC are only on an annual or biannual basis [86, 87]. War games are comparatively easy to arrange as much of the study is predicated on participants making decisions and engaging with one another. As the "script" is developed by

Subject Matter Experts and has a basis in historical data or well-studied potential events, a war game is relatively simple to arrange [97]. Lastly, a computer model requires extensive setup and coding. However, there are minimal physical and logistical requirements, making it easier to arrange than a physical exercise.

*Experiment Run Time*

There will always be some time required to run each case of a Mission Space Exploration. Decreasing this per-case runtime is necessary for the exploration (or a conventional optimization) to be time-effective.

Of the three simulation archetypes, computer models and war games can run consistently at faster than real-time. However, war games become ineffective at engagement-level fidelity even when they are time efficient via a longer traced timescale (e.g. 1 turn = 1 year) [96]. Conversely, full-scale physical exercises happen in real-time by definition.

### 3.2.2 Analysis of Experimental Architecture

To compare the available mission simulation archetypes, a basic Pugh Matrix is used to quantify the qualitative ratings of each approach. A Pugh Matrix operates by assigning a rating to each alternative for each evaluation category. These ratings are either poor, acceptable, or excellent. The better alternative is found by determining which has the most excellent ratings and the fewest poor ratings. Frequently, a poor rating is a "show-stopper" in that it is a near-insurmountable impediment to effectively employing that alternative.

The Pugh Matrix breakdown of the different archetypes can be seen in Table 3.2.

Table 3.2: Qualitative Breakdown of Simulation Archetypes

| Property | War Games | Computer M&S | Physical Testing |
|---|---|---|---|
| Account for Disparate Asset Architectures | Acceptable | Excellent | Excellent |
| Account for Disparate Mission Architectures | Acceptable | Excellent | Acceptable |
| Scaling with System Complexity | Poor | Acceptable | Acceptable |
| Scaling with System-of-Systems Complexity | Poor | Acceptable | Acceptable |
| Account for Existing Asset Performance | Excellent | Excellent | Excellent |
| Account for Proposed Asset Performance | Acceptable | Excellent | Poor |
| Fidelity of Simulated Decision-Making | Excellent | Acceptable | Excellent |
| Experiment Setup Time | Acceptable | Acceptable | Poor |
| Experiment Run Time | Acceptable | Excellent | Poor |

### 3.2.3 Final Architecture Selection

As illustrated by the Pugh Matrix analysis in Table 3.2, a computer-based mission simulation approach is the most applicable to a Mission Space Exploration. It does not have the same "show-stopping" weaknesses as a war game or physical testing, though it is frequently not as good as them in specific instances. For example, decision-focused analyses would be best served by a war game or consulting with SMEs. Similarly, physical

testing is always the highest fidelity experiment of a given plan, mission, or organization. This Pugh Matrix analysis yields the first research conclusion.

Research Conclusion 1.1

A computer model is the most appropriate simulation architecture to generate the dataset required for a Mission Space Exploration and subsequent mission-level approximations.

With the confirmation of Research Conclusion 1.1, the next step is to refine the model selection. Given that a computer simulation tool will be used, a specific form of model must be selected. Given the various fidelities and model architectures available, Research Question 1.2 is simply:

Research Question 1.2

What form of computer model is appropriate for conducting a Mission Space Exploration?

On cursory inspection, there is a single computer M&S form which is appropriate for such detailed simulation. That type is an Entity-Based Combat Model (EBCM) where each individual unit in the mission is simulated independently and with a detailed implementation of physics and interactions. A well-known form of EBCM is the Agent-Based Model where each unit in the simulation acts according to a set of internal rules and logic. As the **SAA** requires the tracing of individual units, their actions, and their decisions, a model which focuses on individual entities seems the most appropriate.

### 3.3 Elements of a Computer Mission Model

Modeling and Simulation is the general methodology by which new designs, ideas, and procedures are evaluated in some form of approximation of its real-life capabilities. M&S can range from high-fidelity detailed analysis of air flows through Computational

Fluid Dynamics to more abstract evaluation of emergent behavior in a model of an ant colony.

An individual model is a base-level approximation of a behavior, physical process, or other element, while a simulation can be generally seen as a fusion of different models to achieve some form of unified understanding [96, 106]. In the context of this document, M&S is assumed to be referring to a computer-based approach to approximating real-world systems and interactions. These computer models come in varying forms of fidelity, integrability, and validation depending on the customer, developer, or requirements.

### 3.3.1 Physics Model

The physics model of a military simulation refers to the applications of Newton's laws and other physical laws to the entities simulated. There are varying levels of fidelity for the physics model. The most basic form often includes only a given entity's position, velocity, and a fixed movement speed. In this example, the entities are either stationary or moving with a constant magnitude velocity. Turns are not tracked as changes to heading occur instantaneously. The position and velocity vectors are used for the decision-making algorithms associated with the entity. This data can also be communicated to allies or "observed" by opposing entities for use in their decision-making routines [15, 16].

The most complex physics models implemented in military simulation include movement in three dimensions and full analysis of forces involved. A common example of a more complex physics model can be seen in the simulation of air-to-air combat. As compared to surface or naval assets, aircraft have full freedom to maneuver in three dimensions translationally alongside rotation (i.e. Six-Degree-of-Freedom (6DOF)). Additionally, aircraft are more likely to approach the physical limits of their design during their engagements (high G-Forces, limits of excess power, etc.) and therefore require

accounting for these subsystem-level effects rather than the system-level metrics alone [107].

In addition to modeling the physics of the battlefield entities, the physics model must also account for any munitions utilized during the simulation. Many combat models eschew the simulation of munition ballistics to save computational time and complexity (i.e. utilize a $P_{kill}$ instead). Other models often include the full simulation of missile and munition trajectories if the performance of these assets is critical to mission success (e.g. missile defense scenarios) [15, 16].

### 3.3.2 Decision-Making Model

The decision-making model of a combat simulation can be seen as the most subjective of the model elements. For the decision-making model, a variety of assumptions must be defined to allow for the attempted modeling of a human thought-process and the simulation of the involved personnel acting and reacting to changes in the environment. One of the more popular approaches to modeling decision-making is an Agent-Based Model wherein each entity in the simulation makes decisions according to a individual and simple set of rules, often phrased as conditional statements. In other cases, the entities themselves lack decision-making authority and are instead directed by a singular decision-making entity (i.e. a "Battle Manager") in a manner reminiscent of the human player in a war game [108, 109].

#### *Game Theory*

Game Theory is a mathematical approach to representing and investigating any problem involving simulated decision-making or interactions. There are diverse problems encapsulated by the area and it was notably investigated by the RAND Corporation to study Nuclear strategy in the 1950s [60, 95]. More recently, the mathematical representations have been applied to System of Systems contexts like a space situational aware-

ness network [99]. The approach has been utilized by Garcia in the context of optimizing evasion behaviors and maneuvers for aerial engagements [61, 62].

The field is most useful when attempting Mission Space Optimization such as detailing optimal maneuvers or decisions by battlefield commanders [99]. However, the biggest complication when considering Game Theory methods and models in **Stochastic Agent Approach** context is the heterogeneous nature of the problems and the desire for a resulting tradeoff environment. Game Theory can be leveraged to investigate problems with few players or of a canonical form, but the specific mathematical models are not as extensible to a generalized context as desired for the **SAA**.

### 3.3.3 Engagement Model

The engagement model is effectively a synthesis of the physics and decision-making models. From the physics side, it is directly related to the complexity of the munition physics (if any) implemented and the fidelity of simulated maneuvers. On the decision-making side, it relates to the organization of the modeled assets into simulated entities and how these units relate to each other during encounters, whether collaborative or combative.

*Lanchester Equations and Laws*

The most simple form of engagement model is the salvo-based attrition model popularized by the Lanchester Equations mentioned previously. For this model, the attrition is the loss to one side or the other due to combat actions. The Lanchester Equations assume that the attrition rate of two opposing forces is a function of the strength of their opponents [14, 15, 16, 17]. This principle can be seen in Equation 3.1 with $A$ and $B$

representing the populations of the two opposing forces.

$$\begin{aligned} \frac{dA}{dt} &= f(B) \\ \frac{dB}{dt} &= f(A) \end{aligned} \tag{3.1}$$

For example, for two opposing forces of differing numbers of units, the Lanchester Equations can be notionally defined as those seen in Equation 3.2.

$$\begin{aligned} \frac{dA}{dt} &= c_1 * B \\ \frac{dB}{dt} &= c_2 * A \end{aligned} \tag{3.2}$$

Here, the two coefficients $c_1$ and $c_2$ reflect a (usually empirical or otherwise based on historical data) relationship between how many of each force exists and their relative firepower or military success against their opponent. For example and in a qualitative case, a tank will have a large Lanchester coefficient against other vehicles but a lower (if not 0) coefficient against aircraft. Oftentimes, this data is informed by large engagement exercises such as Operation Red Flag [13, 86]. These engagements are "salvo based" wherein the actual interactions are not continuous but occur at discrete timesteps with the attrition rate being treated akin to a $P_{kill}$. With the linear differential relationship established and the discrete, stochastic salvo formulation, the Lanchester "square law" is present wherein small force advantages can quickly snowball into victory [14, 17] .

A slight variation on the Lanchester equations is can be used to generate the "linear law" wherein the attrition rates of the two sides is modulated by the population of both [14, 110]. The linear law formulation as seen in Equation 3.3 is formulated to prevent the "runaway" victories which historically have seemed rarer than the Lanchester Laws otherwise indicate.

$$\begin{aligned} \frac{dA}{dt} &= c_1 * B * A \\ \frac{dB}{dt} &= c_1 * A * B \end{aligned} \tag{3.3}$$

Furthermore, the coefficients are not static in a higher-fidelity model [17]. Elements such as the terrain, past maneuvers, and fatigue all affect the implementation of these equations and their coefficients. There might also be proportional or relative thresholds for an engaging force retreating, pressing the advantage, or holding ground [17]. Regardless, the Lanchester equations remain an efficient representation of combat [13, 14, 17]. However, they require the empirical data to justify the coefficient selection beyond various rules of thumb such as the 3:1 ratio which "guarantees" attacker victory [17, 110].

Figure 3.8, Figure 3.9, and Figure 3.10 illustrate example of how the dynamics of an attrition and notional firepower-based combat occurs in a Lanchester Square Law relationship. Each one consists of a different combination of starting populations and relative firepower metrics for two opposing factions (black and green). These three figures illustrate the populations (y-axis) of each side over time (x-axis) with different relative populations and firepowers. In the ratios listed in the title of each figure, the first value represents the "black" faction population and firepower values, while the second value represents the "green" faction population and firepower values. Thus "Population a:b and Firepower x:y" illustrates combat between a black side of population a and relative firepower x, against a green side of population b and relative firepower y. In the basic Lanchester Equations formulation, the relative firepower is the coefficient on the population ($c_1$ or $c_2$ in Equation 3.2).

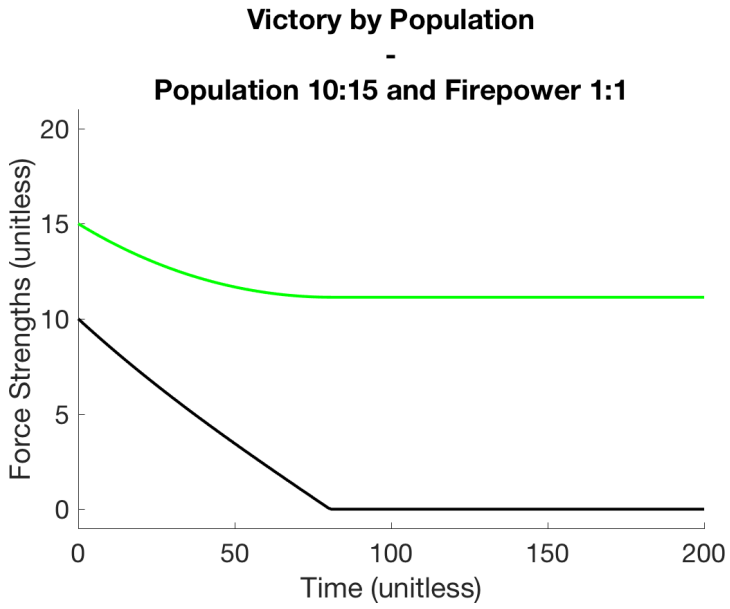



Figure 3.8: Illustration of Faction Population Dominating an Engagement Simulated by the Lanchester Equations

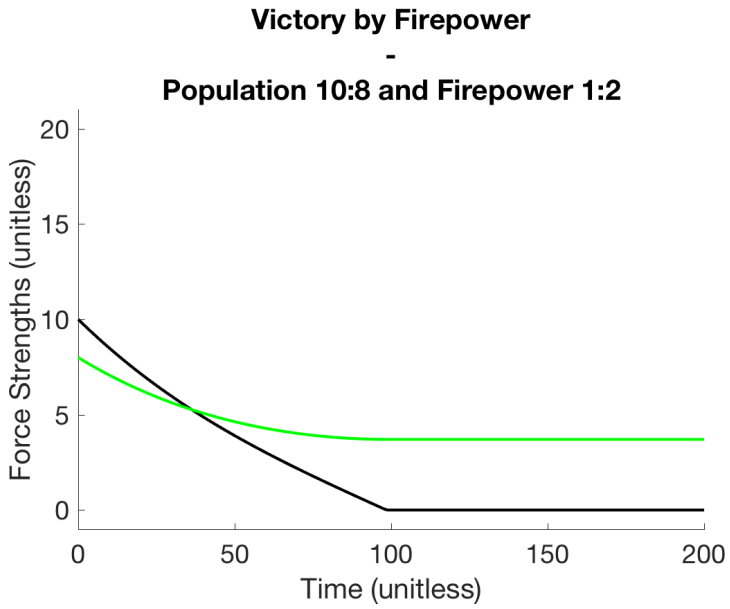



Figure 3.9: Illustration of Relative Firepower Dominating an Engagement Simulated by the Lanchester Equations

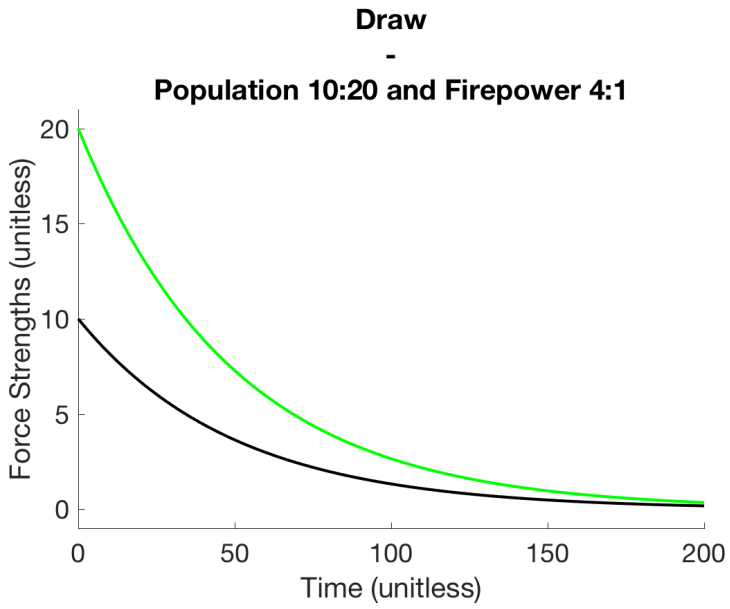



Figure 3.10: Illustration of Population and Relative Firepower Equilibrating an Engagement Simulated by the Lanchester Equations

From these notional models, the aggregating property of the Lanchester formulation is clear in that there is no individual tracking of units. Furthermore, variations in the relatively few parameters available (population and firepower) can result in dramatically different conclusions. Finer discretization of the populations also introduces complication as the models work best on large, continuous values. For large groups, rounding the population is not an appreciable adjustment. However, for smaller engagements, the equations become less useful because "0.5" of a population means a lot more to a force size of 10 than a force size of 10,000.

*Higher Fidelity Engagement Models*

More complex engagement models do not have as clear a universal example. However, such simulations are usually defined by the use of a full physics model for both the engaging assets and their associated munitions. As such, full-fidelity engagement models would track the specific trajectories of the shells, missiles, countermeasures, and other elements through the duration of their missions. These munitions then impact or oth-

erwise damage the opponent once within some radius of interaction. For example, a missile with a fragmentary warhead could be considered a hit if the missile is within some notional damage radius of the target entity [111, 112].

## 3.4 Computer Mission Model Archetypes

Several computer model forms exist. Generally speaking, there is a continuum from the lower-fidelity, higher-level Aggregate Combat Models (ACMs) to higher-fidelity, lower-level Entity-Based Combat Models. These model archetypes are common in numerous different simulation environments, meaning that any one is possible for adoption and use in the **SAA**.

### 3.4.1 Aggregate Combat Models (ACM)

Aggregate Combat Models are the simplest form of combat model. ACMs involve the aggregation of multiple assets on the battlefield into larger modeled entities, often with the assumption that these units fight better in large groups [14, 110]. Additionally, the relationships between entities are often abstracted to a series of empirical relationships like the differential equations that underly the Lanchester Laws [14, 17]. Traditionally, ACMs simulate larger timescales than other models and are thus appropriate for engagements lasting hours or days rather than second-by-second combat sequences. The focus of ACMs is the attrition rates of the sides involved. Experiments involving this model type also consider the possible "inflection points" or necessary force capabilities and numbers to minimize friendly attrition and maximize opponent attrition (see Figure 3.11) [13].

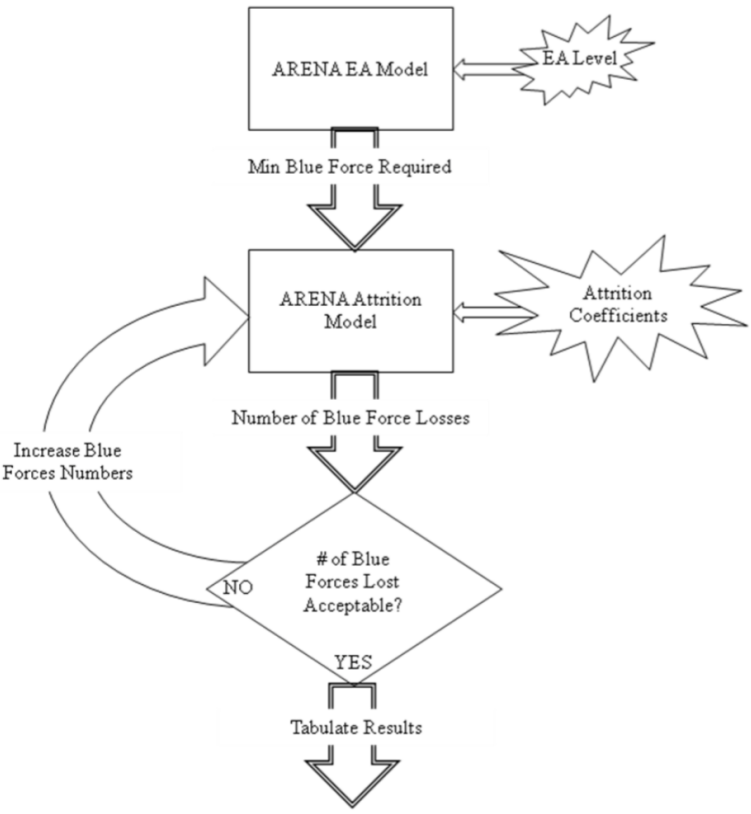



Figure 3.11: Simulation Flowchart for an Air-to-Air Combat Aggregate Combat Model [13]

The major appeal of an ACM is that it can keep simulations within the capability of modern computer architectures and reasonable computation times [15]. Aggregate models assume a degree of homogeneity within a given force, in that the loss of a portion of its fighting strength does not negatively impact the ability of the remaining strength to continue engaging. This homogeneity assumption extends to considering every asset or entity of a given type functionally identical. Aggregate Combat Models are the primary source of various military rules of thumb such as the “3-to-1 rule” [110].

*Process*

The primary feature of an ACM is how it models groups of units, rather than individual assets, as entities [15]. As such, the first step when developing such a model is to de-

cide how to combine these assets into larger groupings or, alternatively, how to separate up larger divisions into smaller groupings if that is the desired fidelity or organization. As such, the United States Army combat unit system is a readily available and sensible structure (see Table 3.3) for an ACM of ground-based assets while similar structures exist for the other branches of the armed services [15].

Table 3.3: United States Army Command Hierarchy [7, 15]

| |
|---|
| Individual combatant |
| Squad |
| Platoon |
| Company |
| Battalion |
| Brigade |
| Division |
| Corps |
| Army |
| Theater |

Depending on the desired fidelity of the ACM, the simulation entity will be a different scale on the command hierarchy. The simulation then treats each entity (e.g. a platoon) as a separate unit that can observe, maneuver, and engage within the battlespace. Because the individual battlefield assets are aggregated, these higher level units must incorporate the general position and nature of its component entities through additional variables such as the area covered or average distance between entities [14, 15]. ACMs frequently utilize the aforementioned Lanchester Equations and engagements are generally considered semi-random forms of combat [14].

In addition to deciding the appropriate level of aggregation in an ACM, the user must

determine whether to simulate homogeneous or heterogeneous units. A homogeneous unit is one whose combat prowess is described by a single metric relative to the total number of remaining entities within the unit, such as the "firepower index" utilized in A Tactical, Logistical, and Air Simulation model (ATLAS) ACM [15]. A heterogeneous unit is one where variations between component entities are tracked and thus its combat ability can be found through a summation of functions describing the combat prowess of its component entities. Additionally, heterogeneous units are useful as they allow for the modeling of effectiveness of certain entities against certain targets (e.g. anti-tank missiles against armored vehicles) [15]. Regardless whether the units are homogeneous or heterogeneous, the groupings move as single force assuming all assets involved can complete the requisite maneuvers.

For a notional example of a heterogeneous ACM, consisting of an infantry unit $A_1$, an allied anti-tank unit $A_2$, and an opposing tank unit $B$. In this scenario, the notional Lanchester Equations may be akin to those in Equation 3.4.

$$\begin{aligned} \frac{dA_1}{dt} &= 5.0 * B \\ \frac{dA_2}{dt} &= 5.0 * B \\ \frac{dB}{dt} &= 0.01 * A_1 + 1.0 * A_2 \end{aligned} \tag{3.4}$$

As seen in Equation 3.4, the attrition rates of the $A$ units is a function of the number of opposing units remaining. Additionally, the nature of the heterogeneous model is apparent in the attrition rate for the tank unit $B$ as the capability of the anti-tank unit $A_2$ is more significant than that of the infantry unit $A_1$.

The Lanchester Laws themselves can be formulated in a more complex manner, with linear and square law formulations being the most common [14]. These two forms of the equations are linked to the types of combat considered for the model. The Lanchester Linear Law is most applicable to simulating "ancient battles" with the assumption that combat is undirected and an entity can only engage opponents close to it [14, 110]. The

Lanchester Square Law is most applicable to "modern battles" with more precise targeting and longer (i.e. cross-battlespace) engagement ranges [14]. However, even these representations could be considered incomplete as they assume full-force commitment to an engagement [110].

*Advantages*

The primary advantage of an ACM is the ability to effectively simulate thousands or more units on the battlefield as a single entity. As a result, an Aggregate Combat Model scales very well with increasing numbers of assets. Many of the most prevalent ACMs are at the theater-level of engagement with examples such as ATLAS, Combat Evaluation Model (CEM), Institute for Defense Analysis Gaming Model (IDAGAM), Vector Research Corporation Aggregate Combat Model 2 (VECTOR-2), and others. In short, for simulating large-scale engagements, an ACM dramatically reduces the runtime without a significant loss of fidelity [15]. An additional benefit of an Aggregate Combat Model is that it is generally simpler to program. Given that the units modeled have been abstracted, the engagements between opposing forces are often described through differential equations or other simple mathematical representations.

*Disadvantages*

The primary weakness of an ACM is its comparative lack of physics implemented in the model. When the individually maneuvering entities are abstracted to the unit level or higher, the physics are usually simplified to velocities, either linear or rotational. Additionally, the use of aggregate attributes ignores any variability between individual units (effectively the Central Limit Theorem from statistics), which might prove important when analyzing specific designs or tactics [15].

An additional complication of an ACM arises with the desire to simulate sortie-based engagements as would be seen with aircraft. Because aerial vehicles need to individu-

ally refuel and rearm after each engagement, the Aggregate Combat Model can break down unless the timescale is sufficient to ignore the time away from the engagement [13]. Many combat models then abstract the aircraft involved to the total population in theater while individual sorties are not simulated. Should each sortie be simulated, the actual engagements follow the standard ACM approaches such as utilizing the Lanchester Equations [13, 15].

Another downside of an Aggregate Combat Model comes from the use of empirical equations to describe combat performance. To generate these relationships, data has to be extracted or derived either from more complex EBCMs or from real-world data. As such, ACMs are best suited for modeling current or past assets where there is easily accessible combat records [13]. Due to the empirical and aggregate representation, there is also an inability to track when specific assets are destroyed or make certain decisions. Thus, there is a loss of traceability.

While operational and strategic level decisions can be observed and affected, Aggregate Combat Models do not function as effectively at the engagement or small-group level. In modern engagements, the number of units involved is often small, especially in comparison to the division-level and historical engagements that ACMs are most commonly used to simulate. Small-scale combat scenarios are also much more beholden to stochastic effects. For example, in a four versus four air-to-air engagement, a single "lucky shot" can remove 25% of the opposing firepower. In comparison, larger-scale engagements are often more robust to stochasticity as a result of the large number of units amortizing most effects.

### 3.4.2 Entity-Based Combat Models (EBCM)

An EBCM is more complex in formulation than an ACM, but can be considered the simpler in principle. Rather than abstracting the combat and other properties as seen in an ACM, an EBCM individually simulates each asset (see Figure 3.12). The individual as-

set models include parameters such as position, detection range, equipment, and target priorities. An EBCM is also often referred to as a "high-resolution combat model" [15]. To clarify the model form, there is no particular code-base, technology, or system associated with an EBCM. Rather it is an approach for how to represent a mission and the actions of the entities within it [108, 109].

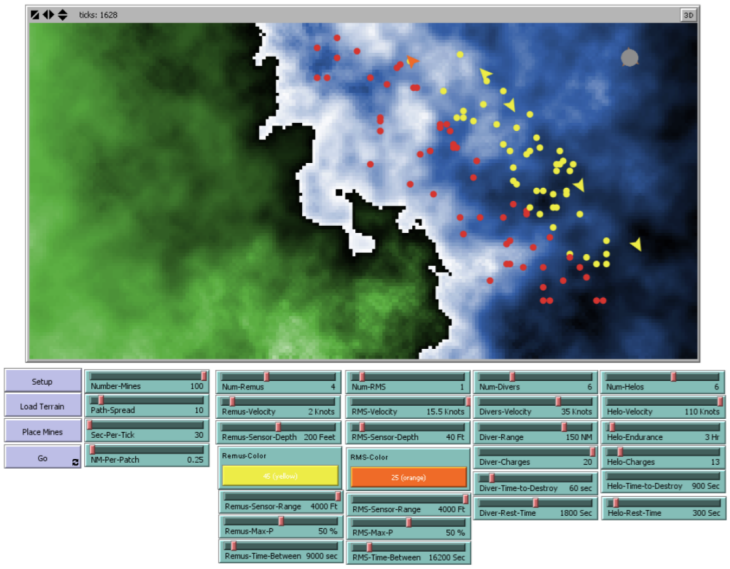



Figure 3.12: An Entity-Based Model for Mine Countermeasures [20]

*Process*

An Entity-Based Combat Model can be broadly divided into three elements: the physical models, the decision-making models, and the overarching combat simulation. Previous sections described the nature of physics models and decision-making models. As such, the process for an EBCM is simply the application of these approaches, at whatever fidelity is deemed appropriate, to individual assets in the battlespace. Notably an EBCM is defined primarily by the fidelity of the physics model and entity tracking, not necessarily any form of decision-making. Higher fidelity decision-making models are commonly used at this level of simulation, but are not required. For example, the model suite FLOPS tracks individual aircraft through their missions, but does not possess any

form of decision-making simulation [37].

An Agent-Based Model is a common implementation of an EBCM, though the method is not limited to the actions of individual units (i.e. you could have an agent-based ACM). While an EBCM tracks individual units, it does not guarantee any degree of autonomous decision-making or simulations to that effect. Meanwhile, an ABM is capable of such feats [108]. Traditionally ABMs are coded in a more decentralized decision-making formulation. However, agents can still "command" and organize themselves in a hierarchy [58]. Through that decentralized formulation, ABMs also are a source of understanding potential emergent behavior as the simplistic interactions between agents and the inherent stochasticity of the simulation propagate through simulated time [108, 109].

For subsequent analysis of an EBCM as an option, an ABM will be considered instead. A deterministic Entity-Based Combat Model generates semi-trivial results as it would represent a passive side in an engagement. Such passivity in maneuvering assets is not representative of real scenarios, though it does simplify the identification of mission plans and alternatives.

*Advantages*

The major advantage of an EBCM is at the engagement level model. With comparatively few assets to individually model, a time-efficient EBCM remains possible with current computer architectures while maintaining an high level of fidelity and traceability. This "audit trail" of effects and capabilities allows for the easier comparison of assets and their parameters [15]. Furthermore, an ABM assumes individual actions are selected from a pre-defined and autonomous decision-making system [108]. Entity-Based Combat Models are also common in more modern mission simulation suites like AFSIM, One Semi-Automated Forces (OneSAF), Map Aware Non-Uniform Automata (MANA), and Engagement Level Air-to-Air Model (BRAWLER) [94, 113, 114, 115].

*Disadvantages*

The primary weaknesses of EBCMs are the computational complexity of their formulation and inherent stochasticity. As a result, EBCMs do not scale well with the number of assets simulated. Depending on the formulation of the Entity-Based Combat Model, stochasticity may require numerous replications [15]. Validation of an EBCM is also difficult though verification is somewhat easy due to the higher-fidelity physics and attempts to simulate human decision-making. Lastly, Entity-Based Combat Models run into the issue of appropriately simulating human behaviors. While an ACM aggregates engagement performance to amortize potential variations and alternatives, the higher-fidelity formulation of the EBCM necessitates additional efforts to simulate decision-making, which are by definition complex as per the difficulties in developing Artificial Intelligence [44, 88].

### 3.4.3 Salvo Model

The Salvo Model focuses not on the minutia of timestep-by-timestep simulation, and instead focuses on large mass-engagements of munitions and interceptors [116]. It is something of a hybrid between an entity-based and aggregate model in that engagements tend to be more aggregated, though individual tracking of assets and munitions occurs. Furthermore, the timescale is more complex as it is built around the discrete events of each salvo rather than continuous-time maneuvering combat or differential-equation effects (as in an EBCM or ACM, respectively) [25].

The Salvo Model is directly derived from the tactics and strategy which shares its name. Mass salvos of munitions are intended to overwhelm defenses with later salvos following up against remaining targets [116]. Such salvo exchanges originated with the modeling of nuclear strikes and retaliations, but now can be broadly applied to waves of precision missile attacks between opponents [60, 95, 116].

*Process*

The Salvo Model can be considered a higher-fidelity representation of the salvo-based Aggregate Combat Model or Lanchester Laws [14, 116]. At each model cycle, the forces (at least those who can engage) launch a salvo of munitions which are then attempted to be intercepted by their opponents. After the munitions are intercepted or reach their targets, damage is assessed, and follow-up salvos are considered. Thus unlike the ACM formulation, the Salvo Model tracks individual munitions and evaluates their effectiveness against targets. Notably, the specific targets of each salvo are tracked and understood for their broader impact on the simulation. Hits against sensors or command-and-control infrastructure have a specific effect on the simulation, much like an EBCM.

*Advantages*

The Salvo Model is superior to an Aggregate Combat Model since it specifically tracks the involved munitions and assets. Furthermore, it accounts for the System of Systems effects of a heterogeneous force, much like an Entity-Based Combat Model. The Salvo Model also runs more efficiently than a full-fidelity EBCM due to the availability of simplifying assumptions of both munitions and entity actions and capabilities.

*Disadvantages*

The Salvo Model does not traditionally include-higher fidelity maneuvering or physics models for the non-munition assets. It is most readily applied to static forces engaging each other, somewhat akin to a nuclear or ballistic missile exchange [60, 95, 116]. Furthermore, the Salvo Model operates in a more discrete-event manner rather than continuous time, thereby being less effective for a complex and continuous engagement such as between large ground forces (when an ACM may be more appropriate) or a small-scale engagement (when an EBCM maybe be more appropriate).

## 3.5 Selection of a Computer Model for Mission Space Exploration

### 3.5.1 Requirements

The requirements for the simulation suite can be divided into two forms, the physics model and the mission model.

The physics model is required because any Mission Space Exploration must be able to (reasonably) accurately simulate the assets required of a given mission architecture. Additional requirements are that the same model can be consistently applied to different asset types. For example, the NASA FLOPS simulation tool is optimally used for civilian commercial aircraft but can be adjusted to simulate general aviation or military aircraft as well. Its ability to estimate performance and capability of these aircraft means that its physics model is quite good. However, that tool cannot easily account for helicopters, ships, ground vehicles, or missiles, and therefore is limited in the scope of mission architectures it can simulate [37].

The mission model is required by definition for a mission simulation. The ability to simulate assets acting, reacting, and communicating with each other is necessary to observe the effects of changing mission parameters on mission success. For example, the open-source NetLogo Agent-Based Model is an excellent tool for simulating entities interacting with each other. However, its physics model is lacking and therefore limits its ability to work alongside other tools to achieve the necessary level of fidelity [91].

Lastly, for developing an approximation atop the higher-fidelity mission simulation cases, there is a weak justification for the Aggregate Combat Model or Salvo Model. Both of these approaches are inherently more empirical than an entity-based approach and so their use in trying to craft a new approximation is less appropriate.

### 3.5.2 Unified Comparison of Simulation Architectures

The Pugh Matrix approach to alternative selection is used to compare and contrast the different mission model architectures available. The same "poor," "acceptable," and "excellent" qualifiers are assigned to the three computer-based simulation approaches. Table 3.4 illustrates the qualitative breakdown.

Table 3.4: Qualitative Breakdown of Computer-Based Simulation Approaches

| Property | Aggregate Model | Entity-Based Model | Salvo Model |
|---|---|---|---|
| Account for Disparate Asset Architectures | Poor | Excellent | Acceptable |
| Account for Disparate Mission Architectures | Poor | Excellent | Poor |
| Scaling with System Complexity | Excellent | Acceptable | Acceptable |
| Scaling with System-of-Systems Complexity | Excellent | Acceptable | Excellent |
| Account for Existing Asset Performance | Excellent | Excellent | Excellent |
| Account for Proposed Asset Performance | Acceptable | Excellent | Acceptable |
| Fidelity of Simulated Decision-Making | Acceptable | Excellent | Excellent |
| Experiment Setup Time | Excellent | Acceptable | Excellent |
| Experiment Run Time | Excellent | Acceptable | Excellent |

### 3.5.3 Selected Simulation Framework

As can be seen from the Pugh Matrix analysis in Table 3.4, an Entity-Based Combat Model approach is the most applicable to a Mission Space Exploration. This conforms with our requirement that the unit-level accountability and fidelity of an EBCM address both the physics and decision-making requirements. Furthermore, it avoids any "poor" ratings which would greatly complicate the utilization of this approach.

Research Conclusion 1.2

An Entity-Based Combat Model will capture the fidelity and variability required for a broader Mission Space Exploration and surrogation

## 3.6 Conclusion

The initial step for developing an approach to sample and approximate a mission space is to identify the method to generate the necessary data points. Based on historical data, past publications, and the presented requirements, a computer-based Entity-Based Combat Model and simulation approach is most appropriate. This method promises to provide insights and understanding in a computationally efficient manner while maintaining a degree of scenario traceability. Various empirically-derived models like the Aggregate Combat Model and war games represent the end-use of developed Surrogate Models and so are inappropriate for generating the required information. Physical testing is complex, dangerous, and expensive in both time and money while the Salvo Model has a more narrow scope of applicability to mission and engagement simulation problems.

With the simulation method identified, the next step of developing the **Stochastic Agent Approach** is to sample across the Mission Design Space. However, that space is expansive by definition due to the dimensionality of the simulated region (e.g. potentially open-ended 6DOF physics and actions). The complications are also related to the

stochasticity of engagements either in the $P_{kill}$, Probability of Detection ($P_{detect}$), or inherent Agent-Based Model variations. Thus the simplifying of the Mission Design Space, or at least the sampled points, is of the utmost importance.

# CHAPTER 4

# METHOD FOR ACCOMPLISHING A MISSION SPACE EXPLORATION

> In view of all that we have said in the foregoing sections, the many obstacles we appear to have surmounted, what casts the pall over our victory celebration? It is the curse of dimensionality, a malediction that has plagued the scientist from earliest days.
>
> - Richard Bellman [93]

## 4.1 Addressing Scalability for a Mission Space Exploration

In traditional combat simulation, there is a lack of variety in the mission parameters beyond the Mission Architecture. The Mission Architecture is simply the overall form of the mission, consisting of properties like the number and type of assets available. This lack of variety can be traced to one of two sources. The first is a desire to fix the asset architecture around some notional configuration (e.g. a fixed-wing aircraft) [28]. The second is to avoid an explosion in simulation time or complexity through defaulting variables to reduce the degrees of freedom [21, 22, 93].

The two problems are linked as the more parameters that can be defaulted to existing values, the simpler the solution space becomes. However, this simplification may result in the loss of potential unconsidered solutions as well as more explicit loss of fidelity. Meanwhile, a full-fidelity simulation with an untenable number of variables is impossible to satisfactorily sample and comprehend in a reasonable amount of time. The "Curse of Dimensionality" also complicates efforts to develop an approximation equation due to an increase both in the number of variables that constitute the model

fit and the number of cases needed to effectively describe the solution space [38].

*The Mission Design Space*

The Mission Design Space can vary dramatically in scale and scope depending on the mission addressed and the desired fidelity. On the lower end of the spectrum, a mission might be solely uniquely defined by the number and type of assets used for the mission. Such an approach would be akin to an Aggregate Combat Model or Lanchester Laws-based problem [13, 14, 17]. With all other mission considerations defaulted due to existing rules or concepts, the actual variable space remains small [11]. This is the traditional solution to a Mission Space Exploration problem: using existing Concept of Operations (CONOPS) to default many of the potential mission variables [49]. However, more variable mission parameters need to be considered and modified in order to appropriately sample the solution space.

For a high-fidelity mission simulation, the variable space might include every action by every entity at every discretized timestep. Consequently, the variable space explodes as the number of expected variables scales multiplicatively with the number of assets, properties of those assets considered, and number of unique actions (or timesteps) tracked in the simulation. The result is a variable space explosion which is unsustainable for larger scale models involving multiple, interacting assets. Such a high-fidelity simulation might be appropriate for a single-entity mission but in the context of a full System of Systems model, such fidelity is not reasonable to achieve.

*Objectives*

The main thrust of this chapter is to address this mission variable scalability problem. Given the selection of an Entity-Based Combat Model for the Mission Space Exploration, is there a means to achieve the fidelity needed for a proper exploration while keeping the variable space manageable? This exploration is necessary because such a broader sam-

pling informs the trade space desired for a more robust mission-level approximation. If optimization were employed, a narrower and less complete understanding would be obtained.

Of secondary concern is the desired traceability of every action in the simulation. A mission model is helpful in every scenario for observing aggregate effects and achievements. This returns to the aforementioned idea that an active-passive mission model can be used to explore passive-passive mission simulations. Unfortunately, models' internal elements and assumptions have become increasingly opaque leading to a lower ability to divine new understandings from modern tools despite calibration from real-world data [60].

However, there has been a more recent push into improved traceability in simulations, namely by delving into the specific actions undertaken by modeled entities [21, 22, 58]. Why did a particular mission go so well? Why did another fail so miserably? These questions can all only be answered by tracking the actions and behaviors of all entities through the simulation. However, it is necessary to distinguish between Mission Action and Mission Design Variables.

The Mission Action Variables can be obtained by simply tracing the time-history of the simulated mission. The primary constraint is the storage capacity of whatever appropriate medium is used (computer hard drive, pilot's notebook, etc.). Meanwhile the Mission Design Variables are those under the express control of the user to define unique scenarios and simulations. These two variables are not necessarily identical. Only when the mission is purely deterministic are the Mission Action Variables and the Mission Design Variables coincident, exemplified when evaluating a civilian aircraft flying a prescribed route [21, 22, 39]. These concerns with the Mission Design Space explosion inform Research Question 2.

Research Question 2

What is an effective means of reducing the scope and complexity of the Mission Design Space without a loss of understanding or fidelity?

### 4.1.1 Variable Types

As alluded to in earlier sections of this document, there are three broad types of variables which can be present in a high fidelity mission model. These three types are continuous, discrete, and categorical variables.

*Continuous Variables*

A continuous variable refers to values that have no breaks along the range of interest. Continuous variables in an aerospace context are best exemplified by physical size variables or non-dimensional values. Examples include wingspan, Reynolds Number (Re), TSFC, and position relative to a point (e.g. coordinates in space).

Continuous variables introduce a degree of complexity to a variable space sampling problem. The infinite number of settings on the continuum of a single variable could dramatically increase the number of necessary simulation cases to understand the solution space. However, continuous variables also simplify some of the physics and other elements of the model by facilitating smooth transitions between simulation cases (e.g. no need for value conditional checks assuming constituent equations remain valid). In short, continuous variables complicate the problem dimensionality but simplify the model implementation.

*Discrete Variables*

A discrete variable refers to values that are numeric and ordered, but do not exist in a continuum. Discrete variables in an aerospace context usually refer to entire subsys-

tems, or more specifically the number of such systems or subsystems involved. Discrete variables are utilized most clearly in cases where fractional values make little physical sense. Examples include number of wings on an aircraft, number of engines on a ship, and number of munitions equipped.

These variables are sometimes the easiest to utilize in a mission context. The limited number of valid variable values reduces the complexity of sampling across the variable space. Meanwhile, the ordinality enables more effective comparison and simulation of different scenarios. Running a mission simulation of an aircraft with two or four missiles equipped is relatively trivial compared to when discrete variables are employed in a more physics-affecting context (e.g. number of engines on a vehicle). In short, for a mission simulation, discrete variables simplify the problem dimensionality and model by reducing the number of considered values within a variable's range.

*Categorical Variables*

A categorical variable refers to values that are non-numeric and thus cannot be ordered or directly compared. Categorical variables in an aerospace context usually refer to configuration decisions of the asset involved. These design choices, while usually critical to the overall form and function of a vehicle, are not themselves ordered or even directly comparable as they can be considered "type" options. Examples include the type of engine utilized, type of sensors on board, and form of locomotion (fixed-wing aircraft, tracked vehicle, boat, etc.) of an asset.

In some ways, categorical variables simplify the variable space of a mission simulation in a similar manner to the discrete variables. With a limited number of possible settings and assuming model adaptability, changing categorical variables is simple. However, that simplification requires a sufficiently adaptable model to accommodate those categorical changes. Frequently, categorical changes can be represented as effects on the continuous and discrete variables (e.g. categorical change of aircraft $A$ to air-

craft $B$ can be represented as changes to the physics and mission profile of that asset). Thus categorical variables simplify the problem dimensionality but complicate model development.

*Overview*

These three variable types are all relevant for a mission simulation. However, it is a hybrid of discrete and continuous variables which are the root of the scalability problem. While each "decision" by a simulated entity is a discrete action, these actions are taken in continuous time and lie along a continuum within each time of action. For example, if the entity flies at a given altitude, the altitude selected is, in its highest-fidelity form, a continuous measure as is the time that maneuver is undertaken.

Thus the Mission Action Space is a massive and complex continuous variable space consisting of a mix of discrete and continuous variables, affecting a discrete number of categorical entities, acting in continuous time. Furthermore, there are variables which do not neatly fit into these categories. Simulated entities communicate, engage, and may be influenced by one another [91, 108, 109]. All these pairwise interactions add up and further complicate the scalability problem.

## 4.2 Current Methods of Variable Space Simplification

When a variable space is well defined, or a large sample set generated from observed data, there are generally two means to reduce problem dimensionality. The first is to directly affect the dimensionality by reducing the number of variables considered. The second is to reduce the fidelity of the sampling along the considered variables [117] .

### 4.2.1 Defaulting Variables

The most straightforward solution to a variable space explosion is to reduce the number of unique design variables available in the solution space [39, 117].

*Advantages*

Reducing problem dimensionality is most common in well-understood problems. In these contexts, the general effects and trends of the design variables are known. This means that the parameters with low influence on the variability of the solution can be defaulted to some known parameter [39].

This simplification is commonly seen in both engineering design and mission planning problems. On the design side, a full-fidelity model would incorporate an understanding and simulation of every individual component of a vehicle. However, the dimensionality (and also fidelity) can be reduced by aggregating these parameters into either subsystems or overall vehicle properties [39]. For example, an aircraft's wingspan might be a design variable instead of the dimensions of every individual component of the wing. In more general data analytics, not every parameter has a significant affect on the metrics of interest [117].

For a mission simulation, various elements can be simplified on both the capability and mission planning sides of the problem. Vehicle physics might be simply aggregated into a ground speed and heading rather than a full 6DOF model. With regard to the overall mission plan, assets might be assumed to operate at a fixed altitude for the entire mission or in some other staged or simplified representation of a path [37, 39, 63].

*Issues and Complications*

Reducing problem dimensionality can be effective but ultimately suffers from issues stemming from a loss of problem and solution space fidelity. In the stochastic and interaction-laden world of mission simulation, it is unknown which variables are important or not *a priori* [20, 117].

Additionally, directly affecting the mission space does not address the desire for a common variable space for all available mission architectures and systems. The literal Mission Action Space of one asset is not guaranteed to be identical to another asset

due to disparate physical capabilities or performance metrics. A quote famously misattributed to Albert Einstein covers this principle by saying that "everyone is a genius, but if you judge a fish by its ability to climb a tree, it will live its whole life believing that it is stupid." In short, using Mission Action Variables to define the available options, and then down-selecting within that variable space, does not actually bridge as many scenarios or architectures as would be desired.

### 4.2.2 Reducing Variable Sampling

Rather than reducing the dimensionality of the problem, one can simply take fewer samples. This can be accomplished either by directly sampling fewer points, or by aggregating the various continuous variables into discrete ones or ranges [117]. On the mission side, the terrain might be discretized into regions and positioning and maneuvering is tracked on that grid rather than continuous space [56].

#### *Advantages*

Lower fidelity variable ranges can hopefully still provide insight into trends and effects without needing to sample as thoroughly across the range of possible parameters. Furthermore, a more discretized mission space greatly simplifies the maneuvering and interaction elements of the model by operating on a grid or other organized arrangement [56, 58].

#### *Issues and Complications*

Simply reducing the fidelity along each variable axis, while simple to implement, has a direct effect on the degree of available understanding of the solution space. The more aggregate the discretized solution points, the higher the possibility of interesting or ideal points being missed during the sampling and analysis [38, 39, 56].

### 4.3 Changing the Variable Space

A tertiary method would be to simply change the considered variables rather than either simplifying or defaulting the existing values. In previous sections, it was made clear that tracing every parameter of every entity in the simulation is a desired capability. Consequently, this full traceability should enable decision-making regarding the performance of a given mission plan. If the complete time-history of the mission is available for every simulated mission, then there is sufficient information (when combined with relevant Measures of Effectiveness) to enable appropriate solution selection. Critically, this means that the selection of points in the Mission Design Space for the exploration is separate from the properties that can be used to define alternatives in that variable space.

In a conventional design variable space, the selected design variable has direct impact upon the algorithms and parameters of the simulation or experiment [38]. A design variable of an aircraft's Aspect Ratio (AR) has known physical effects on its aerodynamics and integration effects on the structures, materials, and other elements of the design [104]. However, what if the aspect ratio was not a direct design variable but rather computed as an intermediary value from other properties? Suddenly, the effects and trends for the AR could be traced, but it is not itself a conventional design variable. This ability to have "direct-impact" variables generated in the midst of evaluation rather than as a direct design variable unlocks new possibility to generate candidate solution points. In short, the question shifts from

> How can one sample across a massive design variable space?

which is frequently a concern in a Mission Space Optimization context, to

> How can one generate a diverse set of alternatives?

which is more in keeping with a Mission Space Exploration.

The questions are slightly different, the key difference being that the form of the in-

puts used to generate the alternatives supplements the conventional design variables, not replacing them. This should facilitate a degree of abstraction from the conventional Mission Action Space discussed previously to a new paradigm which reformulates the Mission Design Space. If a smaller design variable space is employed and thoroughly sampled, perhaps there could be "emergent solutions" amongst the more direct-impact mission variables.

Given the selection of an Entity-Based Combat Model for running a mission space exploration, variations on the model archetype are up for consideration. Foremost among them is the Agent-Based Model which assigns the involved entities a degree of internal agency and decision-making. This naturally allows for a new variable space to emerge, that of parameters which affect the logic and decision-making of the agents rather than specifically dictating their timestep-by-timestep actions [108, 109]. By adjusting these proposed parameters, the inherent stochasticity of the Agent-Based Model might be able to satisfactorily explore the solution space of the MOEs in order to provide the data necessary for a mission approximation. This idea directly informs the Hypothesis for Research Question 2.

Hypothesis 2

Changing the variable space from a direct mission action basis to an agent-based decision-making basis reduces the dimensionality of the design variable space with minimum sacrifice of the ability to explore the Measures of Effectiveness for a given solution space.

### 4.3.1 Agent-Based Models (ABMs)

To help identify a shift in the variable space, it is necessary to understand the inner workings and arrangements for an Agent-Based Model. An ABM is a form of EBCM where individual entities are assigned a series of rules to operate by. These rules are often de-

terministic of the "if-then" form [91, 109].

*Example Predator-Prey Agent-Based Model*

Take a notional predator-prey ABM. In this model, there are two types of agents: predators and prey. The prey simply exist but have a handful of rules to operate on. These rules are articulated in pseudo code seen in Table 4.1 and inspired by an Agent-Based Model tutorial in the NetLogo environment from Northwestern University [92].

Table 4.1: Rules for Prey in a Notional Agent-Based Model

| Sequence Within a Timestep | Condition | Action |
|---|---|---|
| 1 | If random number < probability | Generate new prey ("birth") |
| 2 | (no conditional) | Move in random direction 1 unit |

This table of conditionals can also be illustrated as a Decision Tree wherein the various branches of the tree represent available actions or decisions. A Decision Tree operates by articulating the series of "if-then" statements and conditionals. Every time the decision-making system is queried, the Decision Tree is traversed until an action or decision is made. In the case of the notional prey (seen in Figure 4.1), there are not true decisions so much as two paired random processes.

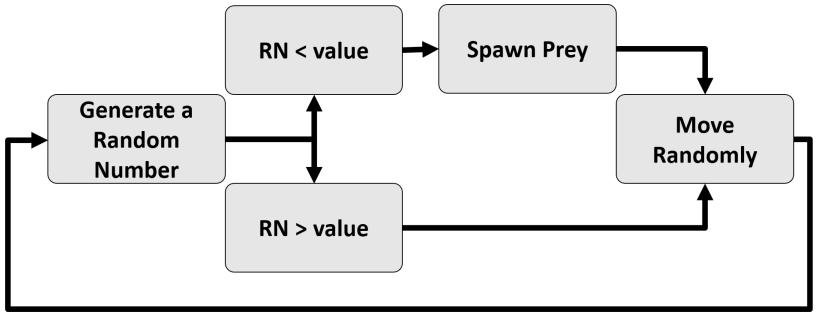



Figure 4.1: Prey Decision Tree Derived from the Notional Predator-Prey Example

The first random process is whether the prey "gives birth" and creates a new independent prey agent. The second random process is which direction the prey then travels for that particular timestep. While represented in Figure 4.1 as a Decision Tree, there are really no decisions happening for the prey agent. Everything is a probability and thus illustrating how stochasticity can affect the resulting agent-based simulation.

Similar to the prey, the predators have a simple rule set as seen in Table 4.2, once more inspired by the ABM tutorial from Northwestern [92].

Table 4.2: Rules for Predators in a Notional Agent-Based Model

| Sequence Within a Timestep | Condition | Action |
|---|---|---|
| 1 | If prey within range | Kill prey, increase energy by 10 units |
| 2 | If random number < probability | Generate new predator ("birth") |
| 3 | (no conditional) | Move in random direction 1 unit, reduce energy by 1 unit |
| 4 | If energy <= 0 | Die |

This table of conditionals can also be illustrated as a Decision Tree for the predator. In this case (seen in Figure 4.1), there are not true decisions so much as two paired random processes and a single deterministic "decision" of whether to consume available prey.

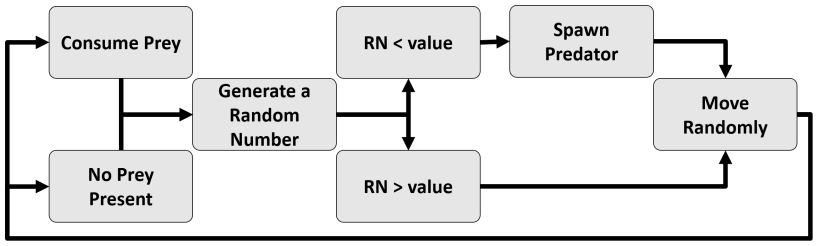



Figure 4.2: Predator Decision Tree Derived from the Notional Predator-Prey Example

The sole decision present in this ABM formulation is whether the predator should consume (kill) an available prey. In this example, the predator will always (and thus deterministically) consume the prey to replenish its energy. As with the prey example, there are also two random processes. The first is whether the predator "gives birth" and creates a new independent predator agent. The second random process is which direction the predator then travels for that particular timestep. While represented in Figure 4.2 as a Decision Tree, once more there are really no "decisions" happening for the predator agent. All elements are either a random process or a deterministic action based on a stimuli. There is also a non-illustrated "death check" wherein if the predator agent reaches an energy level of 0 (having not sufficient replenished it by consuming prey), then the predator agent dies.

*Emergent Behavior*

Agent-Based Models are critically used to identify emergent behavior. The approach is inherently decentralized (each agent acts independently) and so "steady states" or particular macroscopic behavior may become apparent over time [6, 56]. This Agent-Based Model of predator-prey dynamics, though simple, allows for some understanding of expected emergent behaviors, albeit at the aggregate level. Changing the "birth probabilities" of either predator or prey as well as the rates of energy loss and consumption for the predators can provide insight into the relative dynamics of the populations [92]. The model can also be improved through the inclusion of additional resources such as hav-

ing the prey consume “grass” which regenerates on a fixed timescale. The result is an ABM which illustrates the “carrying capacity” of a region with competing species. However, the random movements and deterministic decision-making mean that this form of Agent-Based Model can only provide insights at the aggregate level. There is limited ability to observe what specific actions conducted by a given predator or prey could maximize their lifespan since the overall stochasticity and large Mission Action Space precludes such insights.

*Agent-Based Models and Command Hierarchies*

More complex Agent-Based Models can introduce hierarchies to the problem. Rather than every agent acting independently, a degree of imposed coordination or collaboration can be overlaid. Kewley and Embrechts popularized this approach by considering each level of a command hierarchy as an agent which generated constraints and objectives for their subordinates [57, 58]. A notional example of these constraints and “orders” can be seen in Figure 4.3 where each level of the command hierarchy is simulated as a agent restricting the acceptable “deployment area” of its subordinate agents.

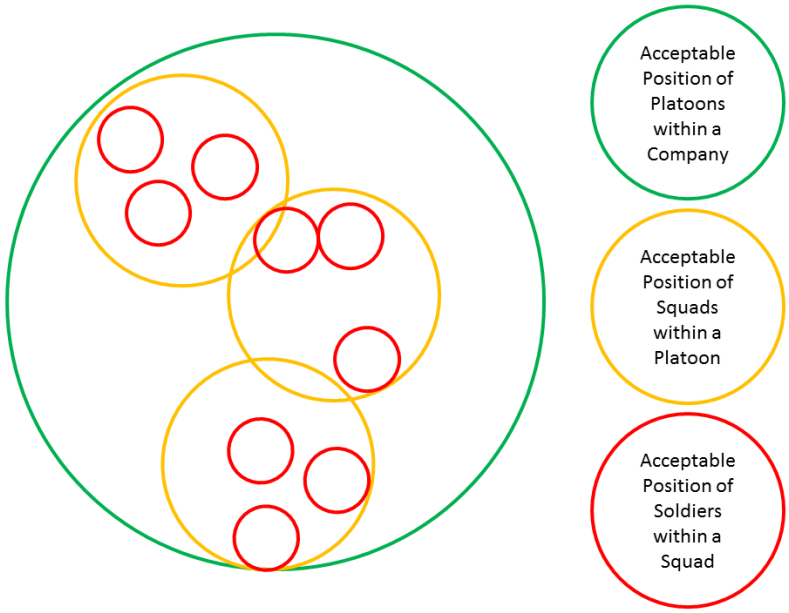



Figure 4.3: Example of Agent-Based Command and Coordination via Hierarchical Constraints

In Figure 4.3, there is a single company commander agent, supervising three platoon commander agents, each supervising three squad commander agents, supervising $n$ individual soldier agents. Each agent has no knowledge of the demands or concerns of its superiors, but is still affected by those considerations via constraints imposed on its decision-making. Thus rather than simulating $9n$ soldier agents with myriad interlaced constraints, there are $9n + 13$ agents with far simpler constraints. Though the variable space is more complex, the actual M&S is more manageable and scalable.

The net result is that for the optimization problem articulated by Kewley and Embrechts, a more realistic operational and tactical-level engagement model was developed [58]. The agents acted less stochastically than in the example predator-prey model and thus the various iterations could incrementally improve the mission plan and architecture until an optimum was reached.

Though optimization is common in mission simulation, the goal of developing the **Stochastic Agent Approach** is to generate superior approximations for engagements.

Utilizing optimization to generate the cases would perhaps lead to overly optimistic results while also increasing the computational cost due the optimization overhead [38, 40].

*Extending the Agent-Based Model*

Regardless, and due to the formulation for an Agent-Based Model, it becomes evident that the "rules" upon which these agents operate can be the origin of a new variable space. Rather than modifying the list of specific actions and decisions undertaken by entities in the simulation, the "library" of actions or preferences for these agents can be modified instead. Then, the inherently stochastic agent-based behavior of the model can simulate numerous variations on the same mission. ABMs can already include stochasticity through aggregate probabilities like $P_{kill}$ and $P_{detect}$ (or random motion as in the predator-prey model) which translate complex interactions into simple probabilities. Furthermore, repetitions of a given engagement can be run with variations in starting conditions [13, 92].

The use of decision-making variables as a design space is not new. However, previous efforts are focused on optimizing or developing new AI for use in video games, war games, or other human-simulation contexts [44, 88].

### 4.3.2 Gaming Artificial Intelligence

There are multiple forms of video games available to consumers today and their development and sale is a $25.1 billion industry. Though such commercial and entertainment products are usually not considered for defense applications, there is an incredible utility to leveraging similar skillsets and experiences between commercial games and military war games [118, 119]. These games rely on computer-controlled opponents to challenge human players during the game. While older systems would have manually (and relatively simply) coded opponent algorithms, more recent examples have embraced

machine-learning and other AI principles to allow the computer to "learn" how to challenge or even beat human players [44, 88, 120].

In the context of discussing video game "AI" it is worth noting that it does not refer to generalized Artificial Intelligence. Rather the term AI is used as something of a catchall for not only "intelligent behaviors" but also path-finding and even some elements of "randomly generated" levels and actions [88]. The most basic computer opponents operated on fixed trajectories such as seen in *Space Invaders*. These fixed behaviors gave way to simple Agent-Based Models such as opponents chasing Mario in his eponymous games. Eventually, more complex approximations of tactics and strategy were introduced as the AI became more centralized and capable, allowing simulated opponents to coordinate and collaborate as seen in various First-Person Shooter (FPS) games like *Battlefield* [88, 120].

*Dynamic Scripting*

Inspired by both video games and Agent-Based Models, Dynamic Scripting is a method of machine learning where the rules by which simulated entities operate changes iteration to iteration. The process starts with a library of possible actions and maneuvers. These elements are then selected to create a given iteration of the AI. The game is played and afterwards, the computer evaluates what elements of its decision-making process yielded better results and maintains those algorithms. Meanwhile, unused algorithms are pruned and replaced to improve performance [88]. Dynamic Scripting can thus effectively create the ABM Decision Tree or logic chain which yields the desired results. Figure 4.4 and Figure 4.5 illustrates different methods for training and developing these AI using training against human and computer-controlled (hard-coded) opponents, respectively.

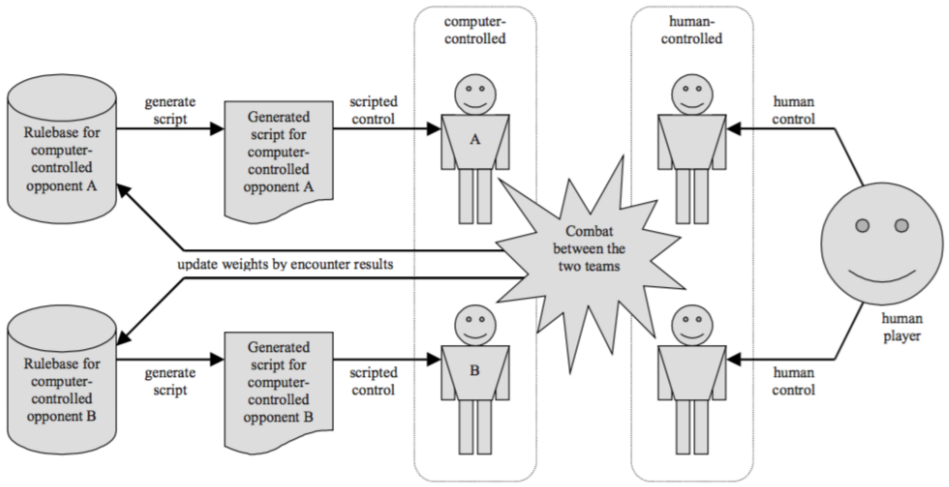



Figure 4.4: Human-Involved Training and Development of a Dynamic Scripting-Based AI [121]

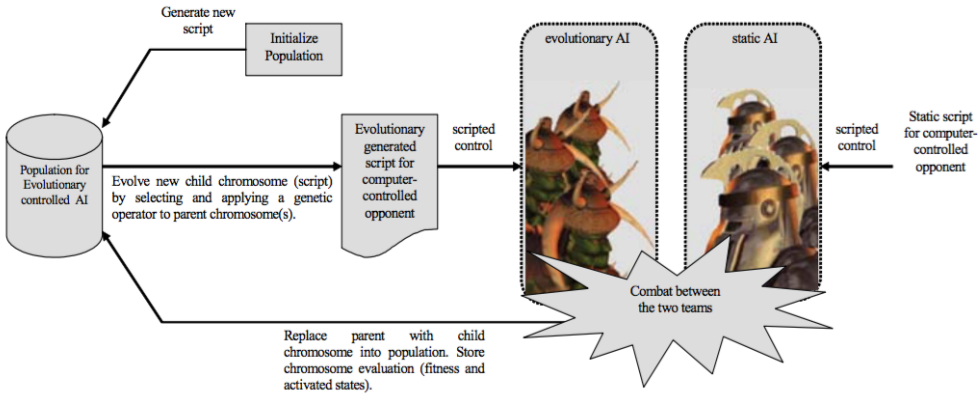



Figure 4.5: Automated and Evolutionarily Optimized Generation of a Dynamic Scripting-Based AI [122]

For Dynamic Scripting, the variable space is effectively a series of Booleans defining which elements of a Decision Tree are maintained. While this space is more tractable than the full Mission Action Space, the agents and battle managers involved still act deterministically. Given the same series of inputs, the actions are identical. This determinism suffers from the same issue plaguing a optimization-focused mission model; namely that there is not a variability in the result to cover a broader mission space.

Furthermore, a focus on optimization for Dynamic Scripting means that there is little focus on an overall heuristic or understanding of decision-making sequences. The most clear example is when this AI generating approach is applied to a more discrete problem. Real Time Strategy (RTS) games are a popular form of entertainment and are a major application of computer-controlled strategic opponents [88, 90]. A key part of RTS games is the "build order" when early in the match there is the need to be judicious in selecting how to leverage limited starting resources. Economic structures provide additional resource income while military structures produce combat units and thus defensive or offensive capabilities. Ponsen notably applied Dynamic Scripting to identify an optimal build order in these games [88]. The resulting build-order Decision Tree is seen in Figure 4.6. Certain buildings act as "gates" to others, hence why the Decision Tree converges and diverges when the "castle" is needed to unlock additional options.

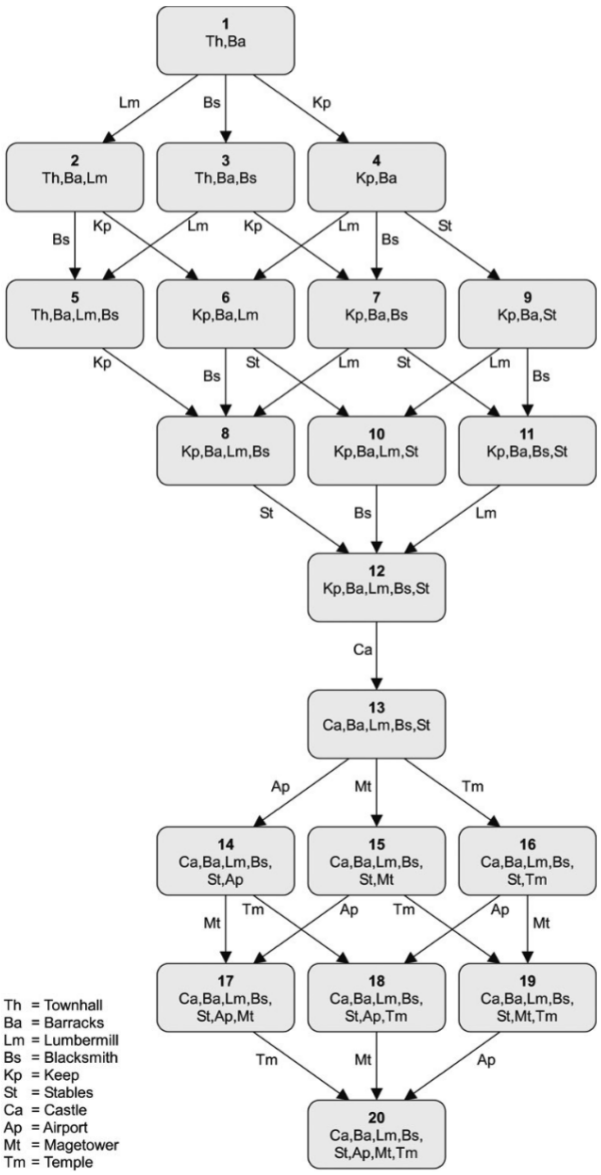



Figure 4.6: Strategy Game Build Order Decision Tree for a Dynamic Scripting-Based AI [88]

As is clear from Figure 4.6, defining different Decision Tree branches is graphically trivial, but computationally expensive. Each layer is dependent on which options were selected previously, thus generating a complex variable space full of conditional checks and discrete or categorical variables.

If the decision-making method were abstracted to individual agents or used in an

exploratory context (rather than a full-factorial search as in Ponsen), then there is no guarantee of sufficient coverage or understanding for a heuristic approximation. Dynamic Scripting works well when developing AI or optimizing them, but lacks the easily articulated variable space which enables tradeoff understandings. Furthermore, the approach is most tractable when developing a singular AI decision-making system, such as a "battle manager" for an RTS game. When modeling individual agents, either relative homogeneity or an acceptance of non-optimal results is similarly necessary [88, 89, 90].

*Deep Learning*

Deep Learning is a newer method of machine learning wherein the computer is given a set of basic rules (such as for a board game, Chess, or Go) and then "teaches itself" how to play. While the initial cases are semi-random, eventually the AI learns and crafts various methods and tactics to use [123].

Once more, the advantage of the method in a solution-generation context negatively impacts its utility in a solution-exploration context. The deterministic nature of the resulting AI logic and the known desire for an optimal both constrain the problem space and prevent effective exploration. To comprehend a solution space, not all solutions need to be feasible or even viable [39]. Failed designs and missions are as enlightening as successful ones, therefore making Deep Learning less useful in the context of this research.

### 4.3.3 Proposal of the Agent-Based Variable (ABV) Approach

While numerous methods exist for coding these intelligent agents and AI routines, they ultimately appear to be deterministic in nature. Furthermore, the development and optimization of ABM routines is fundamentally based on either subject-matter expertise or machine learning. This implies that they suffer from the same complications inherent in an optimization routine: the objectives and priorities must be defined extensively *a*

*priori.* For a Mission Space Exploration, there is neither the time nor the desire to train agents.

Additionally, the **Stochastic Agent Approach** is intended to understand the contours of the solution space, generate a more traceable approximation, and then sample from the distribution on the Measures of Effectiveness to inform higher-level models. Any form of learning by the agents in the model would contaminate the exploration by having early cases influence later ones, thereby preventing effective comparison.

However, stochasticity in the simulation is derived from minor probabilistic elements like $P_{kill}$ and $P_{detect}$ or from a random-walk as seen in the predator-prey example. Thus there is limited "intelligence" behind the actions in a deterministic Agent-Based Model, which is notably not what is desired from a mission simulation for the purposes of exploration and approximation development. Regardless, stochasticity is a promising element to capture and leverage for exploration purposes. Thus the question is how additional stochasticity can be introduced into an Agent-Based Model in order to facilitate exploration. Part of sampling a variety of cases is that no scenario is perfect. If the agents were coded to conduct their mission flawlessly, then there is no understanding of engagement performance since the model has been manipulated to yield "perfect" results.

The proposed solution uses the idea of Agent-Based Variables (ABVs). These are coefficients which define the probability of certain actions being taken by the agents in an Agent-Based Model. Agents will no longer act deterministically, instead many of their decisions will be stochastic and probabilistic. The approach is inspired by Petri Nets, Markov Chains, and State Machines, all of which have either been implemented to improve ABMs or are themselves more stochastic representations of processes [47, 48, 124]. ABVs are convenient to process as they exist on a continuum from 0.00 to 1.00. Furthermore, they have a clearly understood effect for each relevant type of decision available to the agents. The net result is that the categorical decision-making variables are now

continuous and therefore easier to sample and explore across. This idea of ABVs informs Hypothesis 2.

Hypothesis 2 - Refined

Sampling fewer cases but adjusting continuous, agent-decision-affecting variables can accomplish a comparable amount of mission space exploration to a direct full-factorial Monte Carlo approach without losing the insights which come from optimization

#### 4.3.4 Implementing Agent-Based Variables

Unlike in the conventional Agent-Based Model, Agent-Based Variables convert deterministic agent behavior into stochastic. Rather than actions being conducted with certainty, there will only be a probability of an action being taken given a set of inputs. However, not every decision requires this stochasticity. For example, it is unlikely that a "hungry" predator or prey will forgo eating, given the opportunity. Conversely, it is uncertain whether a predator will solely kill a prey for sustenance as opposed to "for fun." This research details a more precise definition and formulation method for Agent-Based Variables in Chapter 5. For the time-being, the focus is on considering whether such a decision-based variable formulation could work for a Mission Space Exploration and sampling.

### 4.4 Experiment for Research Question 2

To experiment with and evaluate the proposed Agent-Based Variables approach, an appropriate case study is necessary. Given that ABVs operate on decisions made by entities in the simulated mission, the case study should include some element of non-trivial decision-making. One option is to have decisions made in a peaceful context like path-finding. Another option is to have some form of simulated combat between agents. Re-

gardless, the ABV formulation must be easily contrasted with a Monte Carlo-like random Mission Space Exploration. This means that any actions taken by entities must be easily quantified into discrete actions or paths through the mission.

As the proposed Mission Space Exploration is intended to assist with combat-based solution generation, an interaction-focused combat model is appropriate. To simplify the case study and allow easier comparison, an ABV scheme will only be employed on one of the two sides in the simulation. To reduce dimensionality, the case study can be two-dimensional with passive opponents who operate more like "impediments" than maneuvering enemies.

Given the set of requirements, the notional case study is inspired by an aircraft strike mission against and through an air defense system. The active allied Attackers seek to destroy a Target located on the far side of a series of Defenders. The distributed air defense systems are functionally equivalent to "barbed wire" in that the entities must maneuver through or around the affected areas. If these defense are not avoided, the Attackers steadily lose health over time. This provides a nontrivial trade space of Attacker health for time needed to reach the Target.

### 4.4.1 Case Study Terminology

**Health** is a numeric representation of the resiliency of an entity. If this value reaches 0, then the entity ceases to function and can no longer be targeted. An entity with 0 health is "dead."

An **Attacker** is an active entity on the *Blue* side tasked with striking its assigned target. For this case study it has perfect knowledge of *Red* entity positions, but only reacts to those within a certain detection range. This entity is the sole motive entity in the simulation and seeks to move from its starting location to its assigned target. The Attacker has a user-specified speed and health among other properties.

A **Defender** is a passive entity on the *Red* side. This entity does not actually make any

actions or decisions. Instead, a Defender's location defines a region where any present Attackers lose health over time. The Defender has an assigned attack radius, within which any present attacker takes damage. The damage value per timestep is set such that an Attacker will lose all of its health if it attempts to cross the full diameter of the circular area of effect from the Defender.

A **Target** is a passive entity on the *Red* side. This entity does not actually make any actions or decisions. Instead, a Target's location is the point that a Attacker is attempting to reach in order to completer its mission.

### 4.4.2 Case Study Architecture and Setup

The case study example is coded in MATLAB. Instead of basing the simulation on an existing Agent-Based Model form, the system is based on a "battle manager" which updates a series of data arrays accordingly. These data arrays contain all the information for the various entities in the simulation. From the values representing each entity, they are updated based on the simulated position and decisions of every entity. The agents all act independently despite the data being processed through the central battle manager.

The simulation tool possesses conditional checks to separate each case depending on formulation. Additionally, the location of the Defenders and Target are fixed simulation-to-simulation to reduce problem dimensionality. The example "map" can be seen in Figure 4.7 with the Defenders marked by black asterisks (and red circles for their range of effect) and the Target marked by a pink cross.

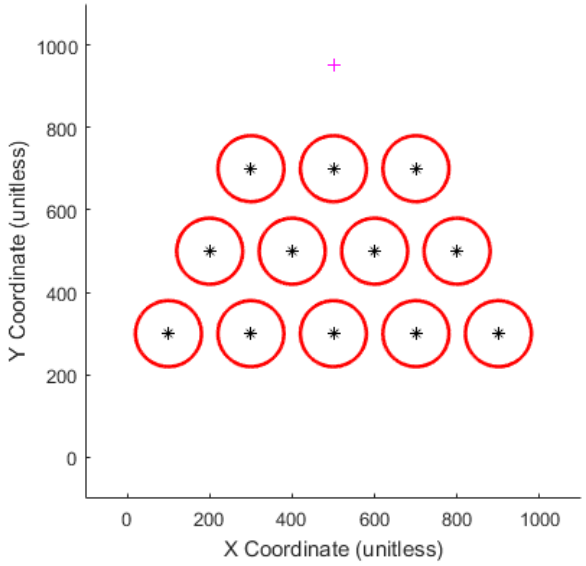



Figure 4.7: Map of the Strike Mission Case Study Showing Locations of the Defenders and Target

The simulation is divided into timesteps. At each timestep, the Attackers move in accordance with either their built-in decision-making or following a prescribed route (depending on the exploration method employed). If an Attacker is within range of a Defender, then it accrues damage. All these various settings are detailed in later sections.

The simulated mission is complete when one of three conditions have been met. First, all the surviving Attackers reach the assigned Target. Second, all the Attackers could be killed. Third, the scenario could run out of simulated time with the time limit set to be approximately three times the minimum possible completion time.

### 4.4.3 Selected Measures of Interest

For this Strike Mission case study, a few metrics of interest were identified *a priori*. Further discussion of identifying these metrics is found in Chapter 7. For the time being and for this simple problem, subject-matter expertise and logic can be used to find values of interest. For any given mission, three metrics of interest are the entity success rates, the entity loss rates, and the time necessary for mission success. In this case study, all three

values are readily obtained by processing the time history of the simulation. The metrics are described in Table 4.3. Notably, the success rate is not listed in Table 4.3 as success is defined as reaching the target (i.e. health > 0 and time < time limit).

Table 4.3: The Metrics of Interest for the Case Study

| Property | Desired Value | Range | Comments |
|---|---|---|---|
| Attacker Remaining Health | Maximize | 0 - 10 | How little damage the Attacker accrued |
| Time of Success | Minimize | 0 - 300 | Time for an Attacker to reach the Target |

These metrics of interest appear logical. In general, an attacking force wishes to maximize its probability of success, lose as few assets as possible, and achieve the mission as quickly as possible. It is noteworthy that unlike a Mission Space Optimization of this problem, there is no relative weighting or development of an Overall Evaluation Criterion (OEC). These metrics are tracked and compared, but they are not directly linked or combined in any manner. The goal is to see if an Agent-Based Variable formulation can cover the same range of these values as a more brute-force or "dumb" exploration with a larger number of cases.

As another point of comparison, an optimization routine will be employed to exemplify a more "perfect" simulation of the mission. If the Agent-Based Variables can get close the results from the optimization method, in a shorter amount of time, then ABVs are a promising approach to the mission variable space. Notably, a successful exploration does not have to capture the true global optima, merely still encapsulate regions of interest and avoid missing relevant solution space contours. As a result, an OEC needs to be introduced to enable an optimization to proceed. The OEC is defined relative to the hypothetical minimum time to target and maximum remaining health. A optimal

path would thus have an OEC value of 2.0 as illustrated by Equation 4.1.

$$
\begin{aligned}
OEC &= \frac{AttackerMaxHealth}{AttackerFinalHealth} + \frac{AttackerSuccessTime}{HypotheticalMinimumSuccessTime} \\
OEC_{maximum} &= \frac{AttackerMaxHealth}{AttackerMaxHealth} + \frac{HypotheticalMinimumSuccessTime}{HypotheticalMinimumSuccessTime} \\
OEC_{maximum} &= 2
\end{aligned}
\tag{4.1}
$$

### 4.4.4 Cases

There are several properties which are held in common between the various cases. These properties include the non-ABV and non-positional properties of all entities. For a full list of entity properties and their assigned values, see Table 4.4. All units are notional and should be interpreted relative to their related properties. For example, an entity's speed is in units of distance per time where the distance and time units are notional, thereby making the speed measure similarly notional.

Table 4.4: Model Properties and Initial Values for the Case Study

| Entity | Property | Value | Comments |
|---|---|---|---|
| - | Attacker $n$ | 10 | - |
| - | Defender $n$ | 11 | - |
| Attackers | Speed | 10 | - |
| Attackers | Max Health | 100 | - |
| Attackers | Attack Range | 1 | - |
| Attackers | Starting X | Varies | - |
| Attackers | Starting Y | Fixed | 50 |
| Attackers | Aggression Coefficient | Varies | ABV-based formulation only |
| Attackers | Path to Target | Varies | Non-ABV formulation only |
| Defenders | Attack Range | 80 | - |
| Defenders | X | Fixed Pattern | - |
| Defenders | Y | Fixed Pattern | - |
| Target | X | Fixed | 500 |
| Target | Y | Fixed | 950 |

Given that much of the variable space is defaulted, an illustration of the resulting mission can be seen in Figure 4.8. In this case, the Attackers are blue circles toward the bottom, the Defenders are black asterisks with red circles denoting their range, and the Target is the pink cross near the top.

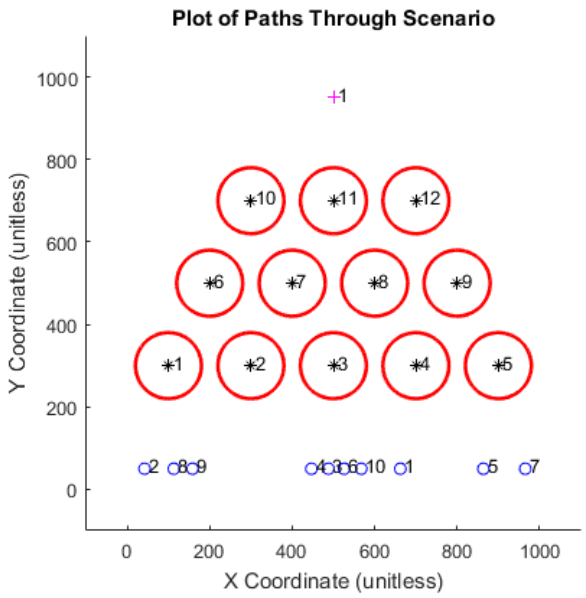



Figure 4.8: Example Starting Entity Arrangement for the Strike Mission Scenario

The overall objective for this experiment is to compare and contrast different Mission Design Space articulation methods and approaches. During the course of the literature review for this research, a total of five generic cases were identified. These methods are not necessarily complete, but do try to span the range of approaches from "brute-force" exploration through detailed optimization. The cases are:

- A Random Walk "brute force" method
- A waypoint-based "brute force" method
- A waypoint-based Genetic Algorithm optimization method
- A specialized optimization routine (the $A^*$ algorithm)
- The proposed ABV-based formulation

Some of the initial properties are set based on how settings are populated. Often, these variables are constant between each of the archetypes while others are unique to a given formulation. All methods employ the same mission simulation components, the

primary difference being how those mission plans are identified or created. The five methods are be described in subsequent sections.

### 4.4.5 Random Walk Method

The most simple form of Mission Action Space exploration is to generate a series of random paths between the Attackers' starting locations and the Target. This means that the resulting variable space has $n * k$ variables where $n$ is the number of Attackers and $k$ is the number of "steps" in an Attacker's path through the region. What constitutes a step in this context varies based on the mission planning formulation. In theory, each Attacker could possess a uniquely defined step at each timestep. Thus the ability to shrink the variable space to be on the order of $n$ is low.

The Random Walk pathing method is intended to illustrate the futility of a "full-resolution" Mission Design Space (or more specifically the Mission Action Space). In this case, the Attackers make pre-defined decisions with a high frequency and with no memory. Furthermore, the paths are defined *a priori* such that the net motion brings each Attacker from its starting point to the Target. However, there is no accounting for Defender placement. As such, the bizarre and random pathing during the mission yields a minuscule mission success rate as Attackers often loiter within range of Defenders and are destroyed rather than reacting.

To simplify the problem, the $y$-coordinate of each Attacker was monotonically increasing. Additionally, diagonal movement was removed for simplicity. This meant that the entire Random Walk consisted of a series of $+x$, $-x$, and $+y$ movements in the two-dimensional Cartesian space.

To generate the random paths for each platform required *a priori* knowledge of both the Attacker and Target locations. Once the net differences in the $x$ and $y$ directions were found, the number of necessary steps was found using Equation 4.2 where $k_1$ is the number of necessary steps, $x$ is the net x motion necessary, $y$ is the net y motion

necessary, $v$ is the speed of the Attackers, and $t$ is the time per timestep.

$$k_1 = \frac{x}{v * t}$$
$$k_2 = \frac{y}{v * t} \tag{4.2}$$

Once $k_1$ and $k_2$ are found, it is subtracted from the maximum number of simulation timesteps allotted (the simulation times out after 300 time units, or 3000 timesteps at the 0.1 timeunit / timestep resolution). The resulting $k_3$ value represents the number of available "useless" steps where the Attacker may deviate from the pseudo-direct path. $k_4 = 0.5 * k_3$ is simply the maximum number of deviations in any one $x$-direction and a random number ($k_5$) is generated between 0 and $k_3$ to find the number of deviations. The result of the pathing is an array with $k_1$ steps in the $x$-direction (positive or negative depends on relative placement), $k_2$ steps in the positive $y$-direction, $k_5$ steps in the positive $x$-direction, and $k_5$ steps in the negative $x$-direction such that $k_1 + k_2 + k_5 + k_5 <= 3000$. Knowing these steps, a vector ($Xdir$) is generated filled with $+1$, $-1$, and 0 depending on the values for $k_1$, $k_2$, and $k_5$. This vector is then randomized to mix up the step order. A paired vector ($Ydir$) is generated such that $Ydir(i) = 1$ when $Xdir(i) = 0$, while $Ydir(i) = 0$ otherwise. The result is a two-dimensional pseudo-random walk between the Attacker's starting point and the Target location.

An example of a Random Walk representation is seen in Figure 4.9.

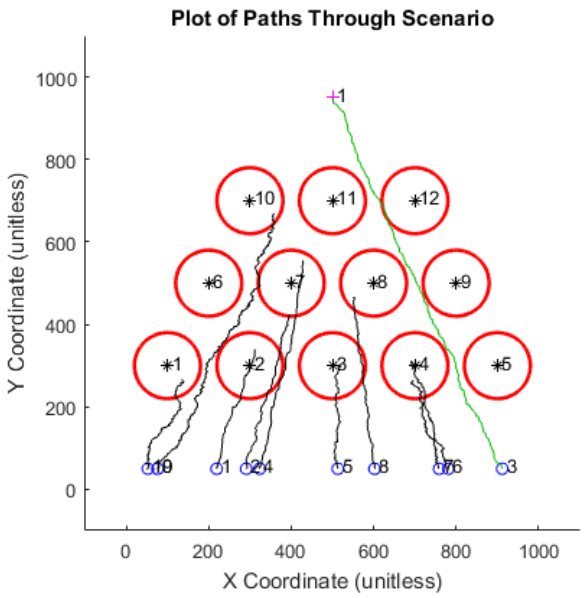



Figure 4.9: Example of Attacker Paths in a High Resolution Random Walk

For an Attacker with the maximum difference in x-coordinate as the Target and using the coordinates defined in Table 4.4, $x = 500$ and $y = 900$. This means that $k_1 = 500$ and $k_2 = 900$ with a maximum of 3000 timesteps. Thus $k_3 = 1600$, $k_4 = 800$, and the maximum number of random deviations $k_5 = 800$. The resulting path vector for $x$ contains 900 0s, 800 1s, and 1300 $-1$s. The number of possible permutations of this vector is found through Equation 4.3 with $N$ being the total number of vector elements and $A$, $B$, and other elements in the denominator are the number of repetitions of each vector element.

$$\begin{aligned} perm &= \frac{N!}{A! * B! \ldots} \\ perm &= \frac{3000!}{900! * 800! * 1300!} \\ perm &= 2.5 * 10^{1398} \end{aligned} \tag{4.3}$$

Thus there are around $10^{1400}$ unique random paths assuming the Attacker starts as far from the Target as possible.

For an Attacker with the same X-coordinate as the Target and using the coordinates

defined in Table 4.4, $x = 0$ and $y = 900$. This means that $k_1 = 0$ and $k_2 = 900$ with a maximum of 3000 timesteps. Thus $k_3 = 2100$, $k_4 = 1050$, and the maximum number of random deviations $k_5 = 1050$. The resulting path vector contains 900 0s, 1050 1s, and 1050 -1s. The number of possible permutations of this vector is found through Equation 4.4 with $N$ being the total number of vector elements and $A$, $B$, and other elements in the denominator are the number of repetitions of each vector element.

$$\begin{aligned} perm &= \frac{N!}{A! * B!...} \\ perm &= \frac{3000!}{900! * 1050! * 1050!} \\ perm &= 3.1 * 10^{1424} \end{aligned} \tag{4.4}$$

Thus there are around $10^{1425}$ unique random paths assuming the Attacker starts as close to the Target as possible. There are more possible variations in this context because there is more leeway to "wander" in the $+x$ and $-x$ directions than in the previous scenario.

Clear from the results of Equations 4.3 and 4.4 is that there is an intractable number of unique Random Walk paths between an Attacker starting location and the fixed Target, even with critical simplifying assumptions (i.e. monotonically increasing $y$). To simplify the problem, unique decisions were made every 50 timesteps rather than every step. The result means that the Attackers move in the same direction for 50 consecutive timesteps before a new direction is set. This less intensive method is illustrated in Figure 4.10.

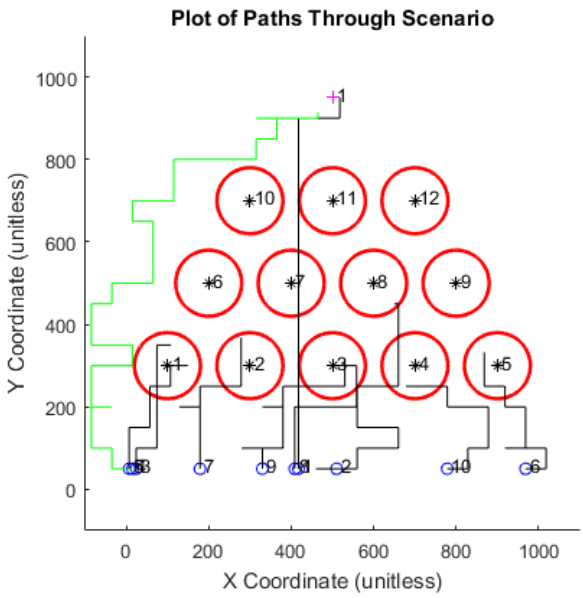



Figure 4.10: Example of Attacker Paths in a Lower Resolution Random Walk

Repeating the mathematical analysis of possible paths through the space effectively decreases the number of steps (and thus Mission Design Variables) by a factor of 50. The resulting path vector contains 18 0s, 16 1s, and 26 -1s for the maximal X-difference case. The number of possible permutations of this vector is found through Equation 4.5 with $N$ being the total number of vector elements and $A$, $B$, and other elements in the denominator are the number of repetitions of each vector element.

$$\begin{aligned} perm &= \frac{N!}{A! * B! \ldots} \\ perm &= \frac{60!}{18! * 16! * 26!} \\ perm &= 1.5 * 10^{26} \end{aligned} \tag{4.5}$$

The resulting path vector contains 18 0s, 21 1s, and 21 -1s for the minimal X-difference case. The number of possible permutations of this vector is found through Equation 4.6 with $N$ being the total number of vector elements and $A$, $B$, and other elements in the

denominator are the number of repetitions of each vector element.

$$\begin{aligned} perm &= \frac{N!}{A! * B! \ldots} \\ perm &= \frac{60!}{18! * 21! * 21!} \\ perm &= 5.0 * 10^{26} \end{aligned} \tag{4.6}$$

With the "50-step" simplification, the number of unique paths has decreased by around 1400 orders of magnitude (factor of $10^{1400}$). Regardless, the number of unique mission paths per Attacker remains intractable. Consequently, it is reasonable to expect the Random Walk method to not only yield poor coverage in a reasonable amount of time, but also to not guarantee even achieving the mission. Thus, the method is a perfect illustration of the futility of *a priori* Mission Action Space sampling and definition.

### 4.4.6 Waypoint Method

The waypoint method aims to achieve a more reasonable representation of a direct exploration of the Mission Action Space. The path for each Attacker consists of a list of waypoints where there is a single waypoint for each y-coordinate and the y-coordinate is monotonically increasing. The result is a more manageable variable space depending on the resolution. An example of the waypoint representation is seen in Figure 4.11.

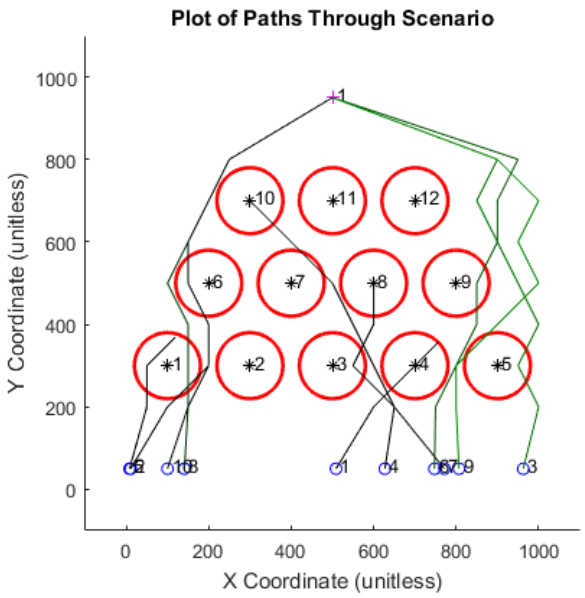



Figure 4.11: Example of Attacker Paths in a Waypoint Formulation

As part of the simplification for the waypoint method, the available waypoints were gridded at a resolution of 50 distance units in both the X and Y directions. The first waypoint is always a random location at $Y = 50$ and the final waypoint is always located at $(500, 950)$, or the location of the Target. The waypoint grid is thus generated along $Y = [100, 200, 300, ...800, 900]$ and $X = [0, 50, 100, 150, ...900, 950, 1000]$. This yields a total of 9 y-coordinates and 21 x-coordinates or 189 total unique waypoints. However, because the assumption will be made that x is monotonically increasing, there are actually 8 decisions of 19 options each, yielding a grand total of $21^8 = 3.78 * 10^{10}$ possible unique path combinations. This number of combinations is more addressable than the Random Walk method, but is still a massive number of options.

To simplify the waypoint method and make it more "logical", the total lateral deviation from waypoint-to-waypoint is limited to 200 distance units in each direction. This converts the 8 decisions of 19 options each to a problem closer to 8 decisions of 9 options each. Thus the problem dimensionality shrinks to $9^8 = 4.30 * 10^7$. Though the number is still massive, it is twenty orders of magnitude better than the 50-step Random Walk

and another three orders of magnitude better than the "full fidelity" waypoint method. Also of note is that making the waypoint pathing more "logical" has the added benefit of improving the various mission metrics of interest. With bounded deviations between waypoints, the Attackers generally do not cross Defenders as much while similarly not wasting time with the lateral motion.

### 4.4.7 Genetic Algorithm Optimization

The Genetic Algorithm (GA) approach is inspired by the waypoint method and the precedent of evolutionary algorithms being used for Mission Space Optimization [58, 125]. This method uses the same discretization of the mission plans as the waypoint method for shared code and simplicity. A Genetic Algorithm is used in this case because the mission space is not particularly continuous and is thus well-adapted to the discretized variable space employed by the algorithm.

The initial population is generated randomly just as in a batch of cases for the Waypoint method. After the population is simulated, a "fitness" metric is defined for each Attacker's path. The objective is to minimize fitness calculated much like a conventional optimization problem [38]. That fitness is proportional to the remaining health of an Attacker, inversely proportional to the time taken to reach the Target, and inversely proportional to the distance remaining to the Target. The actual equation used to find the fitness values is divided into two formulations, one for Attackers which survive the simulation and one for Attackers which do not. This relationship is seen in Equation 4.7 and are derived from the OEC previously defined for the scenario.

$$\begin{aligned} fitness_{alive} &= \frac{AttackerMaxHealth}{AttackerFinalHealth} + \frac{AttackerSuccessTime}{HypotheticalMinimumSuccessTime} \\ fitness_{dead} &= 2 + \frac{DistancetoTarget}{100} \end{aligned} \tag{4.7}$$

The minimum (best) possible fitness value is 2. This corresponds to an Attacker

which completed the mission with no damage and in the shortest hypothetical amount of time (defined by the time taken to traverse the shortest possible distance between the Attackers' starting position options and the Target). The constant in the $fitness_{dead}$ equation is used to ensure that no failed Attacker scores better than a surviving Attacker.

When working through the Genetic Algorithm formulation, example paths can be seen in Figures 4.12 and 4.13. These figures correspond to the first and final generations of the GA, respectively. As evidenced by Figure 4.13, the optimization algorithm narrowed its search to variations on the successful path it identified. However, the nature of the routine also illustrates how it "got stuck" in variations of one particular path rather than exploring other candidates, or in other words, identified a local but not global optima.


Figure 4.12: First Generation of the Genetic Algorithm - Waypoint Optimization

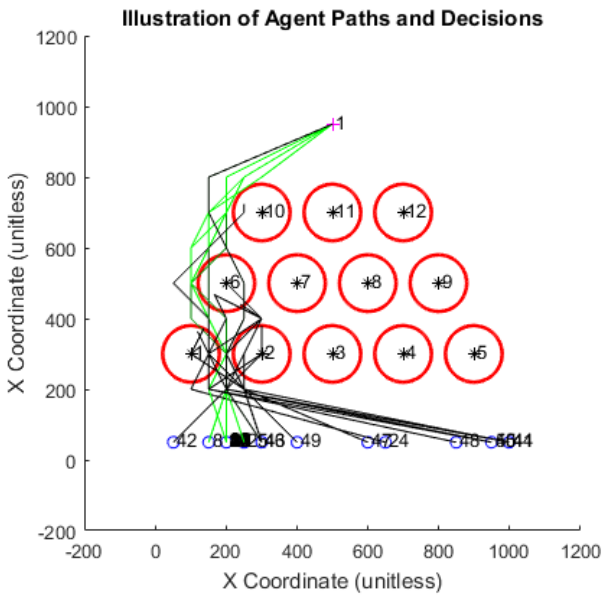



Figure 4.13: Final Generation of the Genetic Algorithm - Waypoint Optimization

The Genetic Algorithm has precedence for working in a mission planning context. However, the optimization routine is only as good and refined as the mission space provided to it. While the waypoint method is more manageable than the Random Walk, it is still a massive and contoured solution space. Similarly, while the GA can provide feasible and satisficing mission plans, it is not expected to find the optimum as easily as a more refined or adapted method.

### 4.4.8 A* Optimization

Given that the two Monte Carlo approaches (Random Walk and Waypoint) and the GA optimization do not show promise as sampling methods, it is necessary to consider a reformulation of the optimization problem. The selected approach was to convert the mission from an interaction-focused model into a terrain-map like region. This conversion would enable the use of graph-based optimization routines which are popular for path-finding applications.

Pre-processing reads in all the Defender locations, and assigns a “cost” to each re-

gion based on their proximity to the Defenders. The original Mission map can be seen in Figure 4.14 while the terrain map conversion of the region is seen in Figure 4.15. In this scenario, the maximum cost was set to be 10 at the location of each Defender, transitioning to a value of 0 at a radius equal to 1.2 times the attack radius of the Defender (the same as the buffer or "sensor range" in the ABV method to be discussed later). To reduce computational complexity of finding location transition (or movement) costs, the region resolution was decreased by an order of magnitude such that the 1000-by-1000 unit region is instead treated as a grid of 100-by-100 10-unit squares.

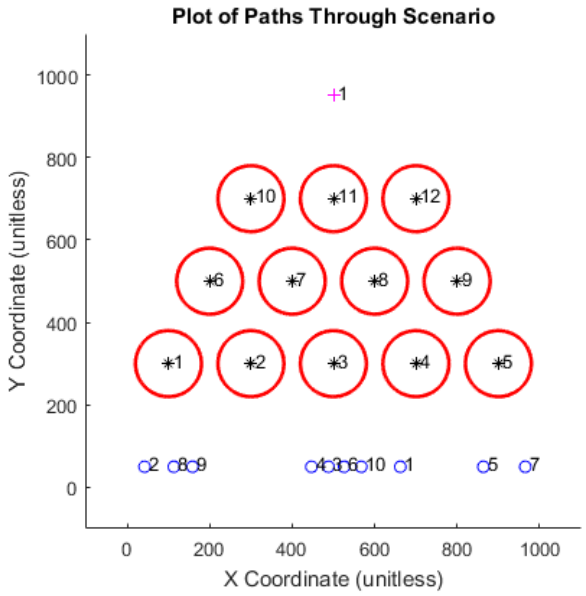



Figure 4.14: Example Entity-Based Representation for the Strike Mission Case Study

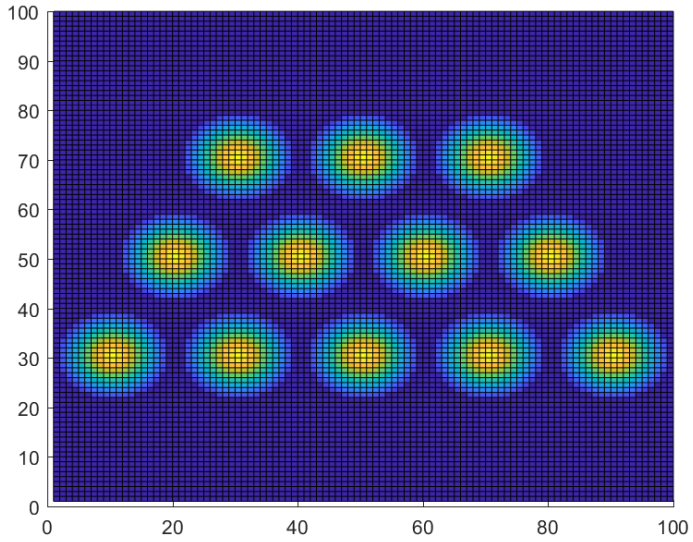



Figure 4.15: Example Terrain Map Representation for the Strike Mission Case Study

The optimization routine employed is known as the $A^*$ algorithm. The approach is a variation on Djikstra's algorithm and is thus a breadth-first search method [126]. This approach entails picking the lowest cost "next step" of the path. However, the assigned cost value is adjusted by an estimated cost that is a function of distance from that point to the destination. As a result, paths which deviate further and further from the destination are eventually penalized to the point of no longer being selected, despite otherwise promising properties [126]. The algorithm is also adaptable in that it is not constrained to pick the lowest cost step based on its immediately previous action. All possible next steps are considered, even if they branch from another explored point (rather than the "tip" of the explored region).

Using the $A^*$ optimization routine generates paths through the space for the Attackers to follow. After the paths are generated, the mission is still simulated to generate the necessary metrics of interest. It is worth noting however that an optimal path does not always guarantee one with a shorter travel time or less damage. The mission environment had to be translated from the "map" to a "terrain" and the optimizer does not

understand the other elements relevant for the mission simulation. As such, some of the optimal paths actually yield "dead" Attackers if the appropriate buffers are not included (hence the 1.2 multiplier on the "terrain" radius relative to the true Defender attack radius). Figure 4.16, Figure 4.17, and Figure 4.18 illustrate examples of these optimized paths.

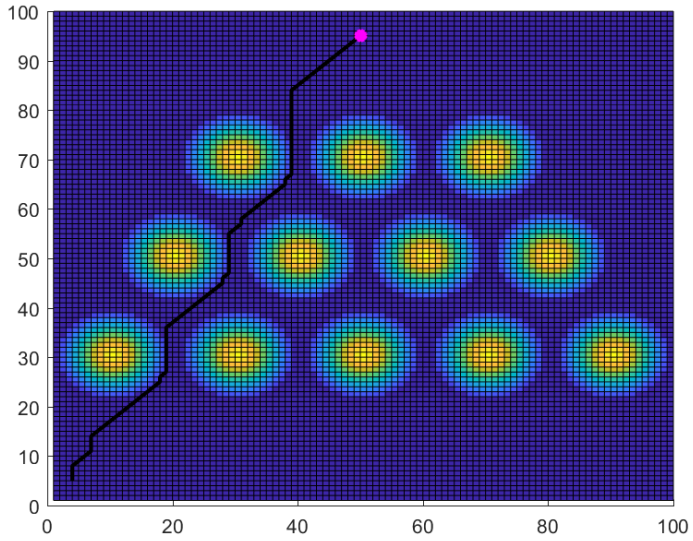



Figure 4.16: Optimal Path for an Attacker Starting at X = 50

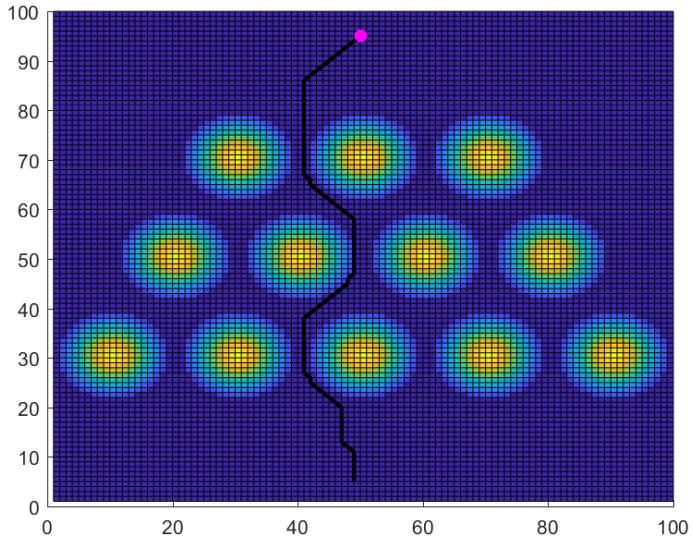



Figure 4.17: Optimal Path for an Attacker Starting at X = 500

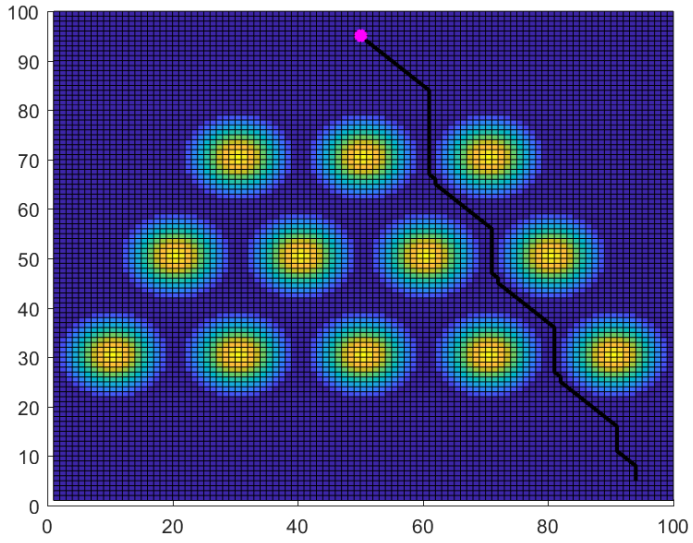



Figure 4.18: Optimal Path for an Attacker Starting at X = 950

For the $A^*$ method, there were 101 unique starting locations considered along the range [0, 1000] in steps of 10. This resolution matches the 10-unit discretization employed to reduce the problem dimensionality.

### 4.4.9 Agent-Based Variable Method

The Agent-Based Variable method is the proposed approach to not only reduce the variable space dimensionality, but also simplify the formulation of the problem. The primary objective is to compare the results from the ABV formulation to the optimization. The end-goal is to illustrate that an ABV-based method can not only retain an understanding of optimal mission plans (though with no guarantee of the global optimum), but also facilitate tradeoffs unlike the optimization methods.

The Aggression Coefficient is the sole Agent-Based Variable in this simulation example. The variable represents a probability that a given Attacker will ignore a detected enemy. For example, an Aggression Coefficient of 0.0 implies that the Attacker will always move around (evade) a Defender while an Aggression Coefficient of 1.0 implies that the Attacker will always ignore the Defender (and take damage in the process if within range). Values between the two extremes represent the probability of either evading or ignoring. Examples of paths for Attackers given different Aggression Coefficients are seen in Figure 4.19, Figure 4.20, and Figure 4.21.

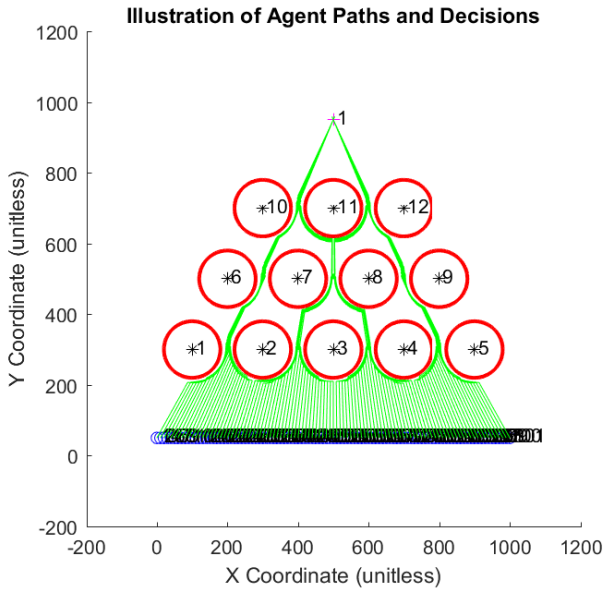



Figure 4.19: Example of Attacker Paths with a Low (0.0) Aggression Coefficient

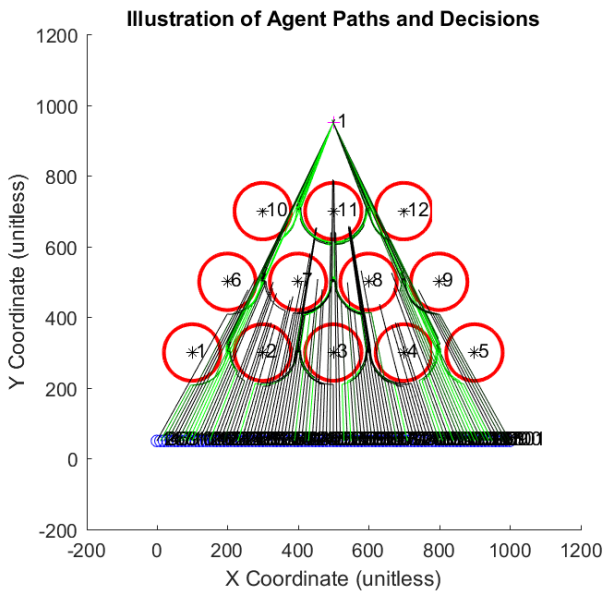



Figure 4.20: Example of Attacker Paths with a Moderate (0.6) Aggression Coefficient

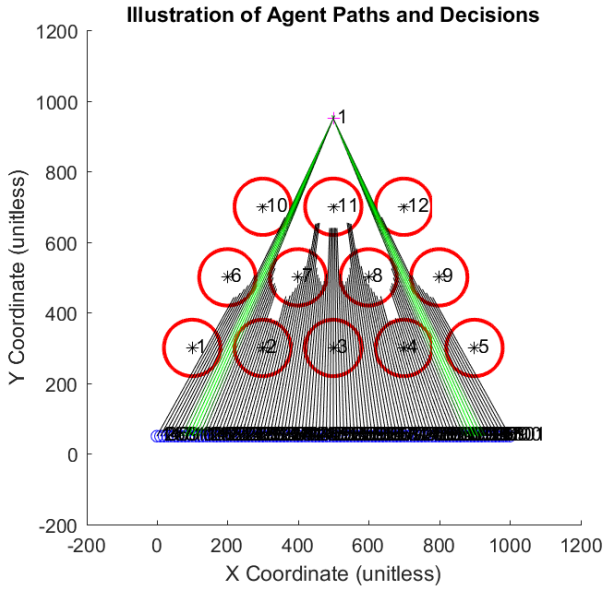



Figure 4.21: Example of Attacker Paths with a High (1.0) Aggression Coefficient

For the Agent-Based Variable formulation, the decisions made are simply whether a given Attacker will seek a path around or through the region covered by a Defender. The resulting ABV Mission Space is thus simplified into a $2 * n$ number of variables where $n$ is

the number of Attackers. The two variables for each entity are its starting location along the X-axis and its Aggression Coefficient. However, this variable space can be further simplified by making all Attackers use the same Aggression Coefficient. The result is a $n+1$ variable space. 101 Attackers were simulated in order to mimic the 101 unique starting locations considered by the $A^*$ algorithm.

Thus the ABV formulation has only these two design variables. The remaining variables are all defaulted otherwise in keeping with the variable descriptions detailed in Table 4.4. Table 4.5 provides the value and method for setting these available design variables. The 11 Aggression Coefficient settings with 101 Attacker starting points yields a total of 1111 cases to run. No repetitions or full-scale Design of Experiments were employed due to the example nature of this experiment. Future work and experiments will seek to refine these measures.

Table 4.5: The Design Variables for the Agent-Based Variable Formulation

| Entity | Property | Value | Comments |
|---|---|---|---|
| Attackers | Starting X Coordinate | Fixed | [0, 10,... 990, 1000] |
| Attackers | Aggression Coefficient | Fixed<br>Case-to-Case | [0, 0.10,... 0.90, 1.00] |

The main changes to the ABV formulation is the need to add additional logic for the agents. Their baseline behavior is to travel along the straight-line path to the Target. The single ABV of Aggression means that the only user-affected decision-making is whether to evade or ignore Defenders. For an Attacker ignoring a Defender, the simulation logic remains intact and the agent will continue along the straight-line path to the Target. However, additional simulated logic is necessary for any evasive maneuvers pursued.

When encountering a Defender that is to be evaded, the Attacker first calculates its

position relative to that Defender and the Target. If the Attacker is otherwise past the Defender, then the Defender is ignored, However, if not, then the Attacker uses a series of conditional checks to consider the different angles between itself, the Defender, and the Target. The net result is that the Attacker will choose the more "logical" route to the Target based on whether it is "above" of "below" the line between Defender and Target. Thus there are four potential positions and logic checks which determines Attacker behavior. This breakdown is seen in Table 4.6 and illustrated graphically in Figure 4.22.

Table 4.6: The Conditional Checks for Attacker Evasion Logic

| Net X-Direction to Target | Relative Position to Defender and Target | Evasion Direction |
|---|---|---|
| +X | Outside | Clockwise |
| +X | Inside | Counter-Clockwise |
| -X | Inside | Clockwise |
| -X | Outside | Counter-Clockwise |

Figure 4.22 only illustrates the two cases where the net x-direction between Attacker and Target is positive. However, a mirror image of these maneuvers would represent the actions for cases where the net x-direction is negative.

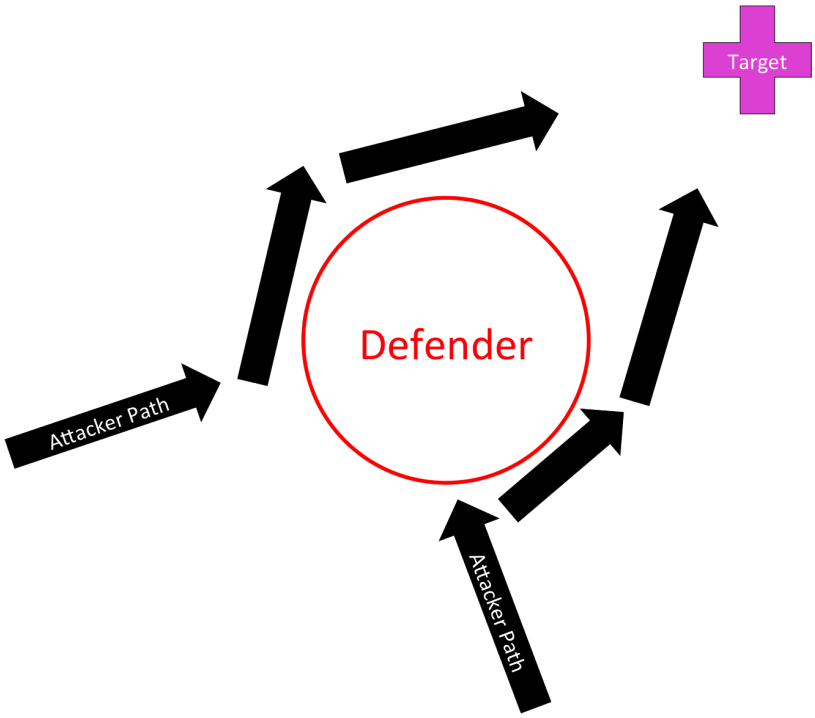



Figure 4.22: Illustration of Attacker, Single-Defender Evasion Decision-Making

Notably, there is a secondary logic check such that if an Attacker is already evading in a given direction (clockwise or counter-clockwise), that evasion maneuver is maintained if a new Defender is encountered. This scenario is illustrated in Figure 4.23.

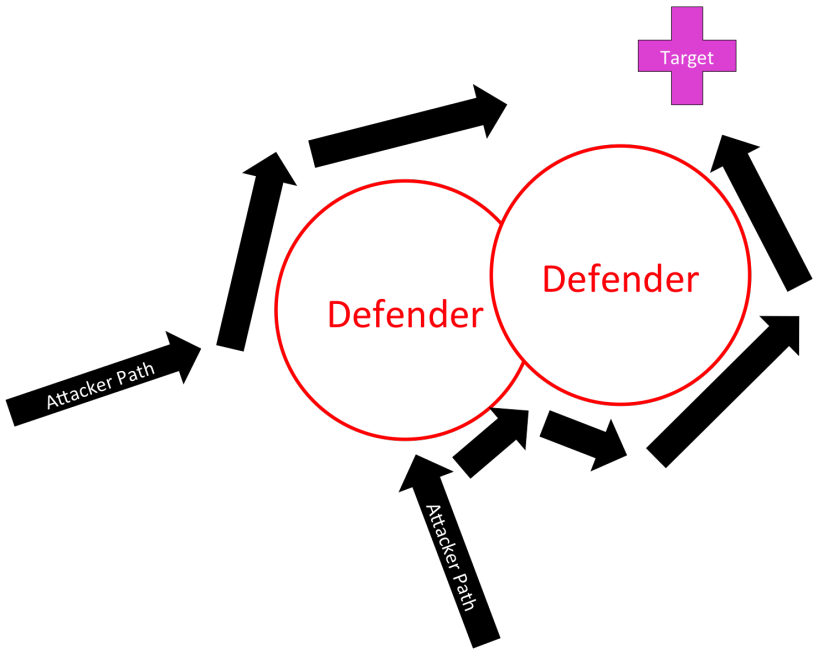



Figure 4.23: Illustration of Attacker, Double-Defender Evasion Decision-Making

This deterministic decision-making was retained as such because there is limited utility of stochastic behavior in this context. These sorts of considerations are tailor-made for implementation in an Agent-Based Model because the logic is sound and the implementation is relatively simple. As such, adding stochasticity for these decisions does not contribute to an exploration of the problem nor does it make logical sense for inclusion.

In the process of developing the ABV method, two different "resolutions" were considered. The main difference being the simulation decision-timestep used in the model. In the basic mode, decisions were made and updated every timestep. However, to better match the region resolution of the $A^*$ algorithm, the decision time was increased to be every 10 timesteps. This effectively converted the solution space to be a 100-by-100 region rather than a 1000-by-1000 region, and thus identical to that for the $A^*$ algorithm breakdown. All subsequent images of simulations using ABVs will use this lower resolution breakdown. However, the higher resolution breakdown is kept and discussed in the comparison seen in Table 4.7;

## 4.5 Experimental Results

The results for each of the five methods are compared to evaluate whether they constitute similar degrees of coverage of the space. The main goal is to find a Mission Design Space which is amenable to not only effective sampling, but also a reasonable possibility of generating close-to-optimal results without the associated computational overhead. Thus the goal should be to reduce the number of variables needed, reduce the time needed per simulation case, and to maximize the ability to identify solution regions of interest.

### 4.5.1 Random Walk

The Random Walk, as expected, was the least successful plan generation method. The approach did not have any intelligent consideration of Defenders as such, Attackers frequently "blundered" into Defenders and did not react to leave the area of effect. Figure 4.24 illustrates all successful Random Walk missions while Figure 4.25 illustrates all failed Random Walk missions. For 1000 total cases, only 9 were successful while 991 failed ( 1% success rate).

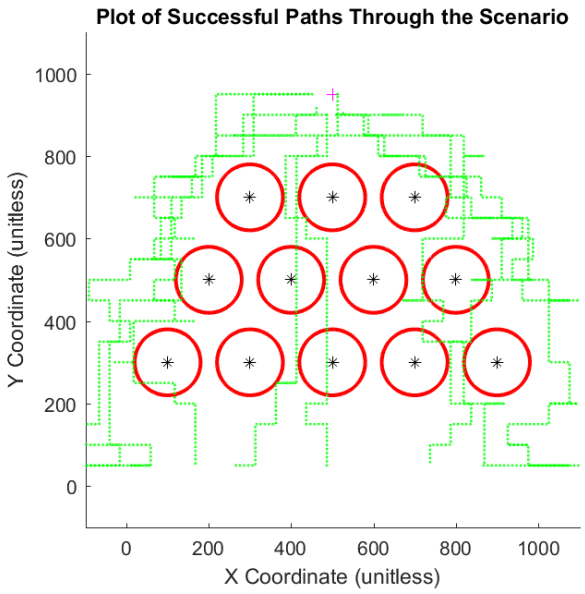



Figure 4.24: All Successful Paths Found Through the Random Walk

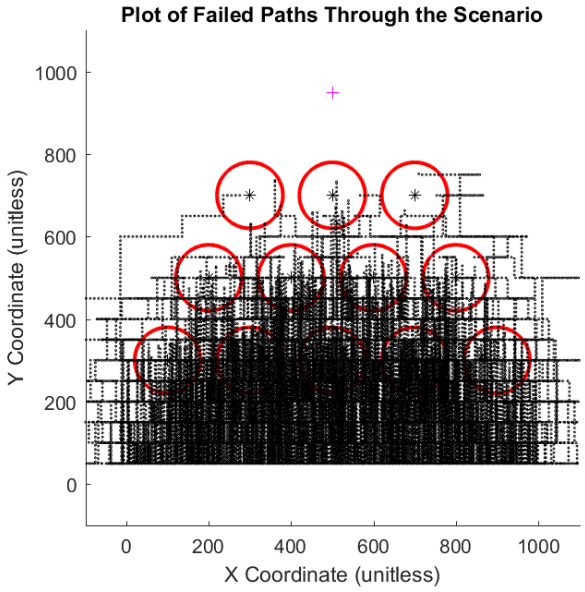



Figure 4.25: All Failed Paths Found Through the Random Walk

### 4.5.2 Waypoints

The waypoint method fared better than the Random Walk but was still not particularly capable of finding successful paths. Once more, the same issue of problem dimension-

ality remains as there were millions of potential path options. Figure 4.26 illustrates all successful waypoint missions while Figure 4.27 illustrates all failed waypoint missions. For 1000 total cases, 90 were successful while 910 failed ( 9% success rate).

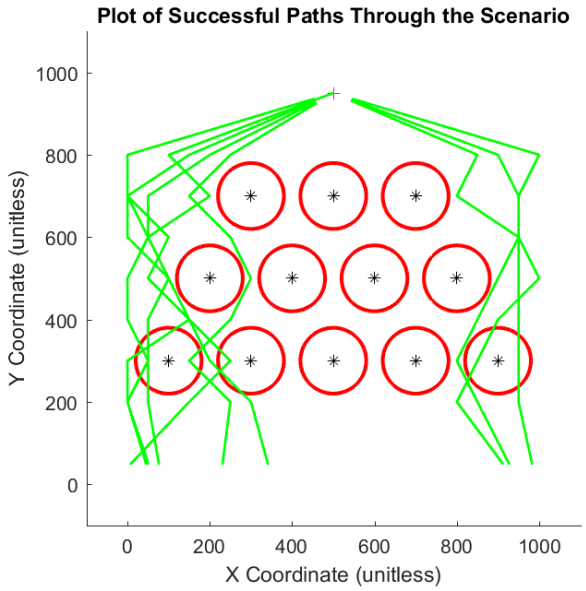



Figure 4.26: All Successful Paths Found Through the Waypoint Method

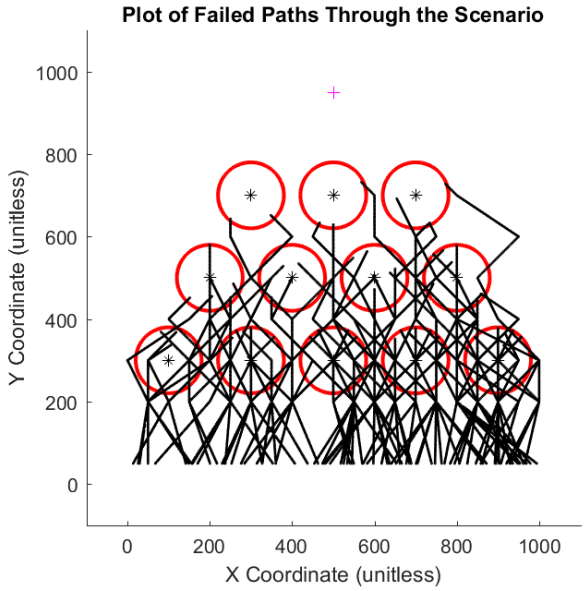



Figure 4.27: All Failed Paths Found Through the Waypoint Method

### 4.5.3 Genetic Algorithm Optimization

The GA method was better than its Monte Carlo equivalent in the baseline waypoint method. However, the same issue of problem dimensionality remains as there were millions of potential path options and the Genetic Algorithm narrowed in on a feasible and satisficing, but non-optimal solution. Of note are how the considered mission paths change generation-to-generation of the GA as seen in Figure 4.28, Figure 4.29, and Figure 4.30. For 1500 total cases, 1072 were successful while 428 failed ( 71% success rate), though that does include repeated paths across and within generations.

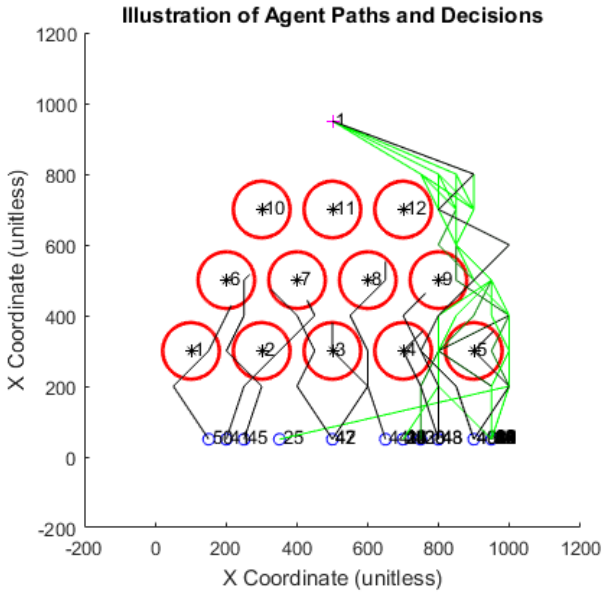



Figure 4.28: 10th Generation of the Genetic Algorithm Optimization

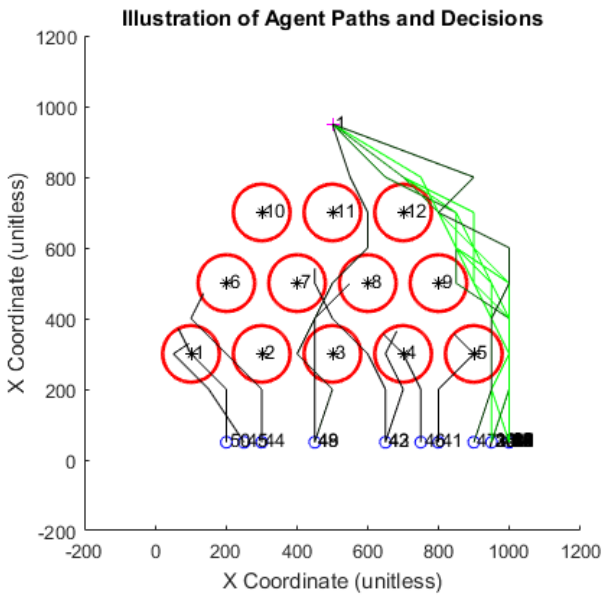



Figure 4.29: 20th Generation of the Genetic Algorithm Optimization


Figure 4.30: 30th (Final) Generation of the Genetic Algorithm Optimization

As can be seen across the simulations of the Genetic Algorithm, this optimization approach does a poor job of spanning or exploring the Mission Variable Space. While it does find successful or satisfactory paths, it does so neither efficiently nor that optimally.

Figure 4.31 illustrates all successful GA waypoint missions while Figure 4.32 illustrates all failed GA waypoint missions.

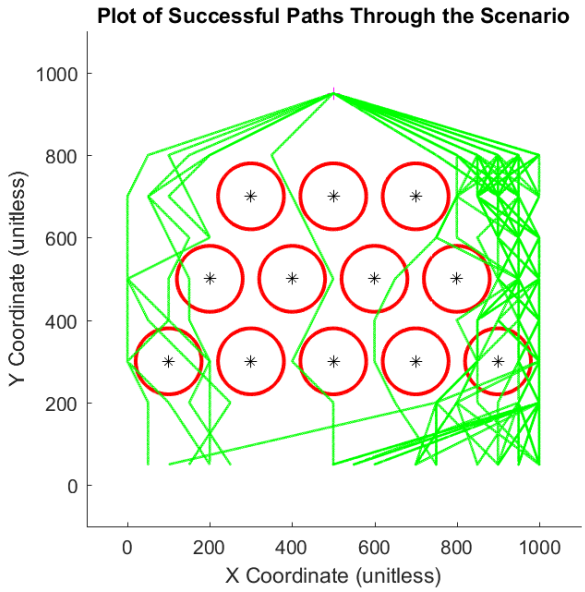



Figure 4.31: All Successful Paths Found Through the Genetic Algorithm Optimization

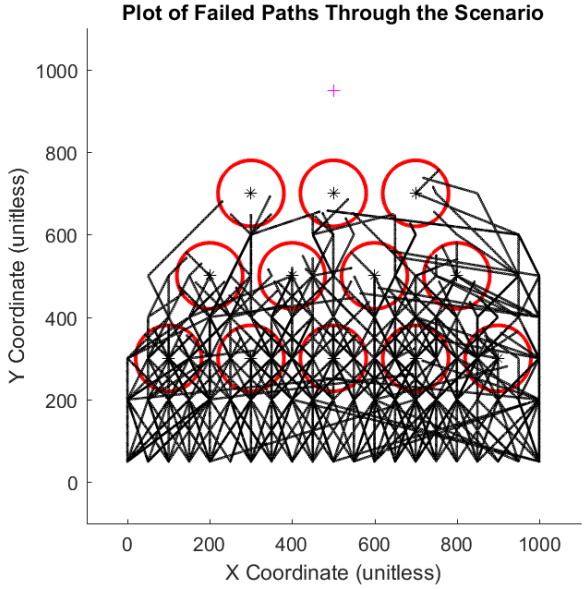



Figure 4.32: All Failed Paths Found Through the Genetic Algorithm Optimization

### 4.5.4 A* Optimization

The $A^*$ optimization routine generated all feasible, satisficing, and arguably optimal paths. While the "true optimum" is slightly nebulous, the results from this algorithm should be considered the effective optimal for each starting location. The Attackers took little-to-no damage and followed what seem to be the shortest paths. Visual inspection and human logic corroborate these findings while Figure 4.33 illustrates those optimal paths.

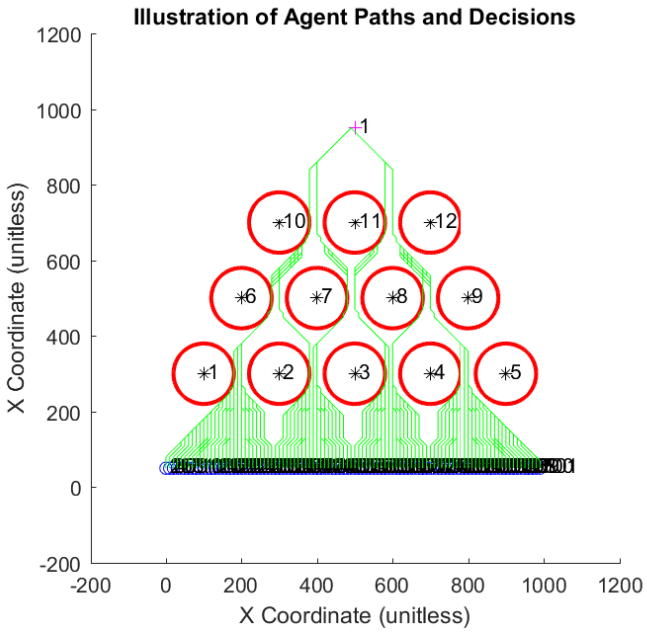



Figure 4.33: All Mission Paths Found Through the $A^*$ Algorithm

It is worth noting however, that the use of such an optimization routine effectively eliminated some degree of noise from the problem. The optimal mission plans were identified *a priori* and were done so in conjunction with a translation of the mission region from a map to a terrain system. Such a translation required full knowledge of the region ahead of time and therefore produced optimal but not robust missions. Furthermore, the resulting cases were all close to perfect, which doesn't yield sufficient data or variation for the generation of a trade space. While reformulating the terrain "conversion

function" could provide some insights, it is the disconnect between the optimization routine and the actual user-defined mission space, which introduces complications.

### 4.5.5 Agent-Based Variable Exploration

The Agent-Based Variable formulation is the proposed method for sampling across a Mission Design Space in a reasonable amount of time. Figure 4.34 illustrates all successful ABV missions while Figure 4.35 illustrates all failed ABV missions.

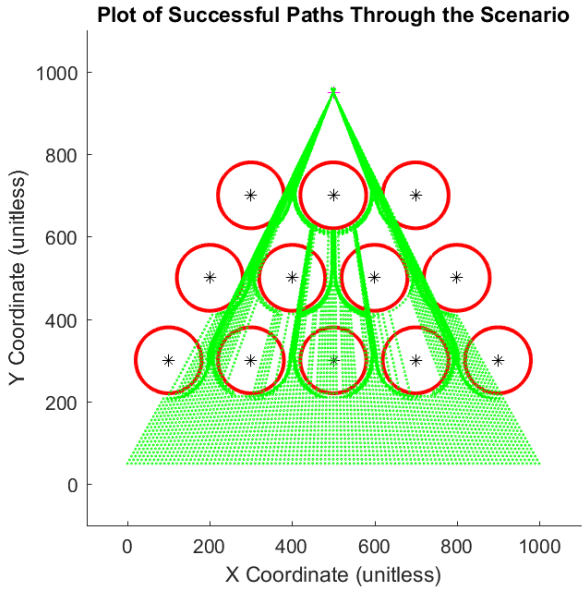



Figure 4.34: All Successful Paths Found Through the Agent-Based Variable Exploration

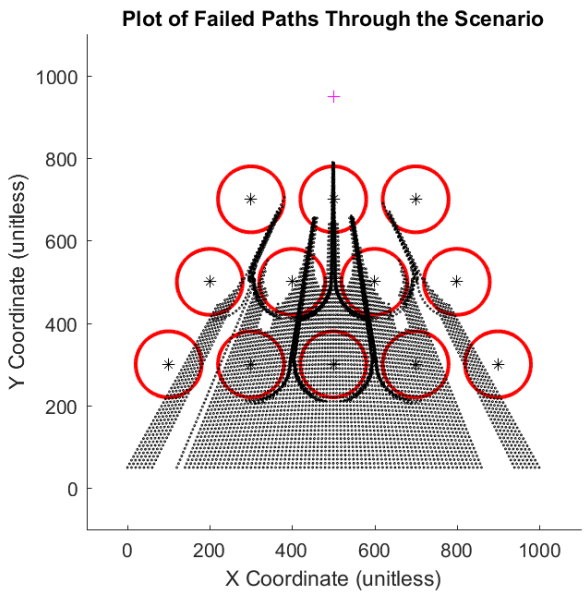



Figure 4.35: All Failed Paths Found Through the Agent-Based Variable Exploration

Of note in these images are the effects of internalizing the logic, decision-making, and thus the optimization. The Attacker agents all followed simple maneuvering logic and with the element of stochasticity from the Aggression ABV, "took some risks" which occasionally paid off by trading time to completion for health. These sorts of trades are of interest when developing a trade space for use in a campaign simulation. If agents act purely deterministically, or according to a series of binary rules, then the ability to generate a continuous representation these decision-making concerns would not be possible.

### 4.5.6 Comparison of Methods

All five methods were subjected to comparison. The two previously discussed metrics of interest are provided as is the OEC values found by the same fitness equations used for the Genetic Algorithm approach. The various computational metrics are seen in Table 4.7.

Table 4.7: Comparison of Mission Exploration Approaches

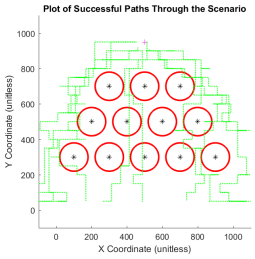
| Method | Cases | Simulation Time (s) | Max Health | Min Time | Optimal Value |
|---|---|---|---|---|---|
| Random Walk | 1000 | 692.7 | 100 | 1202 | 3.7 (100, 2392) |
| Waypoints | 1000 | 671.0 | 100 | 1312 | 3.4 (100, 2179) |
| Genetic Algorithm | 1,500 | 1,133.6 | 100 | 1015 | 2.3 (100, 1143) |
| $A^*$ | 101 | 533.0 | 100 | 986 | 2.1 (100, 986) |
| ABVs (Low Res) | 101 | 3.74 | 100 | 960 | 2.1 (100, 990) |
| ABVs (complete) | 16,665 | 592.8 | 100 | 960 | 2.1 (100, 990) |

Notable in the comparison is that the optimizations had a higher cost per case than the brute-force (Random Walk and Waypoint) or ABV formulations. This makes sense as the two brute-force approaches have a relatively low, front-loaded computational expense of randomly generating paths. The ABV method offloads some of the computational expense see in the optimization to be within the simulation itself, though that cost is still less than the "overhead" incurred by the two optimization routines.

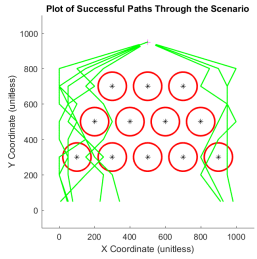
Furthermore, when it comes to find the "optimal" path according to the aforementioned fitness function, the optimization remains superior to the ABV exploration, but not by much. Furthermore, the massive added computational overhead for the $A^*$ optimization could be used by the ABV approach to run either replications or different ABV settings. With approximately two orders of magnitude of improvement in runtime per case setting, the ABV formulation could check eleven (11) different Aggression Coefficient Settings across fifteen (15) different Defender placements in a comparable amount of time (the final line of Table 4.7).

### 4.5.7 Comparing Successful Mission Plans

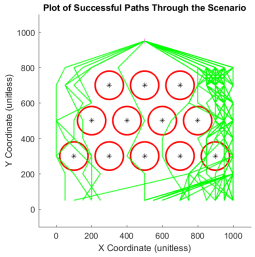
The successful Attacker paths are worth investigating because these "victories" (no matter how pyrrhic) are the most of interest to a decision-maker seeking to compare feasible solutions in a trade space. The successful paths for each simulation can be seen tessellated in Figure 4.36.



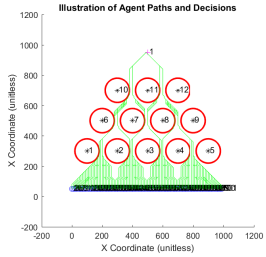

(a) All Successful Missions as Found by the Random Walk Method



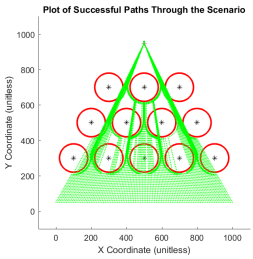

(b) All Successful Missions as Found by the Waypoint Method




(c) All Successful Missions as Found by the Genetic Algorithm Method




(d) All Successful Missions as Found by the $A^*$ Method

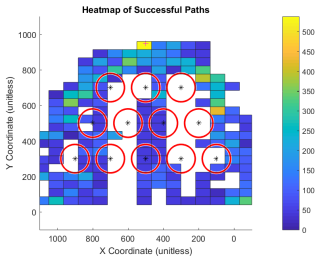



(e) All Successful Missions as Found by the Agent-Based Variables Method

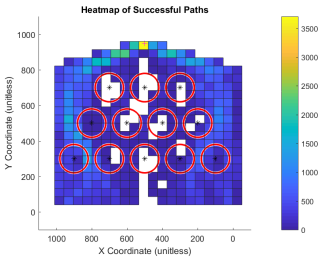
Figure 4.36: All Successful Paths Found Through the Scenario

Furthermore, there is a variation in the densities of regions investigated depending

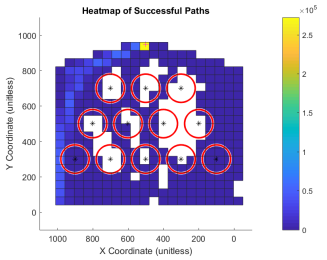
on the path-generation method utilized. The Waypoint-based and Random Walk methods both favored missions avoiding the mass of Defenders by traversing far to the extremes of the x-axis. This aligns with their methods of mission generation which was relatively random and unstructured. Missions which took the Attackers beyond the area of the Defenders were more likely to succeed because that minimized the chances of "blundering" back into a Defender. These heatmaps of successful paths can be seen in Figure 4.37.



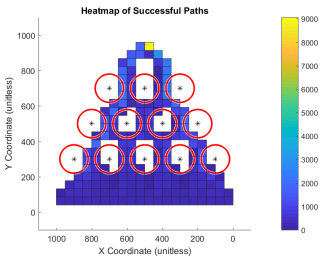

(a) Heatmap Illustrating the Regions Most Visited by Successful Random Walk Missions



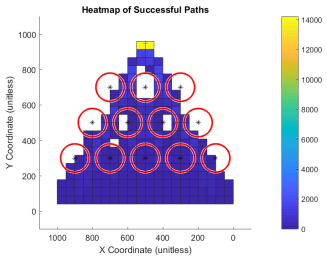

(b) Heatmap Illustrating the Regions Most Visited by Successful Waypoint Missions




(c) Heatmap Illustrating the Regions Most Visited by Successful Genetic Algorithm Missions




(d) Heatmap Illustrating the Regions Most Visited by Successful $A^*$ Missions




(e) Heatmap Illustrating the Regions Most Visited by Successful Agent-Based Variables Missions

Figure 4.37: All Successful Paths Found Through the Scenario (Heatmap)

Figures 4.36 and 4.37 also aptly illustrate the the strengths and weaknesses of the optimization approaches. The Genetic Algorithm, while returning suggested missions similar to the brute-force Waypoint method, did utilize in-mission aspects to affect which paths it investigated. The fitness metric was derived directly from the basic metrics of interest of Attacker health and time to complete the mission (or whether it was completed

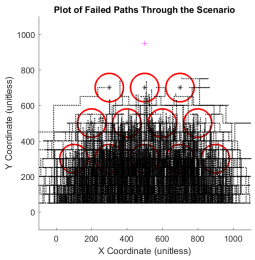
at all). However, the mission it returned was suboptimal globally and did not have the quality coverage seen in the $A^*$ of Agent-Based Variable methods.

The $A^*$ algorithm, though requiring a front-loading of calculations and a conversion of the mission space to a terrain map, did return good mission plans. However, these plans were crafted without in-simulation aspects of the mission such as the time of success or Attacker health. The method also breaks down when uncertainty is introduced because conversion of the mission space to a "terrain map" requires significant foreknowledge. Unlike the previous three methods, the $A^*$ algorithm found mission paths between Defenders because it operated on a more refined discretization of the space. $A^*$ also possessed more entity control than the semi-random Random Walk, Waypoint, and Genetic Algorithm approaches because the optimization narrowed the search automatically.

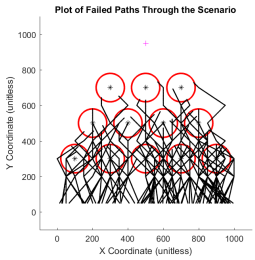
The Agent-Based Variable method, similar to the $A^*$ algorithm, found successful mission plans between and amongst the Defenders. With intelligent entities employed in this approach, they were able to acknowledge and avoid the Defenders and therefore could operate between them with reasonable confidence. In fact, the comparison between the $A^*$ optimization and the ABV approach illustrates perfectly the relative "closeness" of their solutions.

### 4.5.8 Comparing Failed Paths

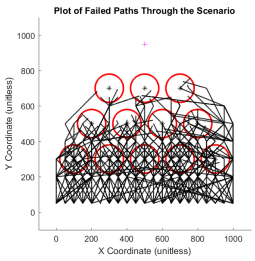
Though failed missions are necessary for full coverage, many of those failures are "dumb" in that they would never have been considered by a real-world entity or human decision-maker. These failures can be seen in Figure 4.38. Notably, there is no image for $A^*$ algorithm based "failed paths" because none existed for that optimization method.



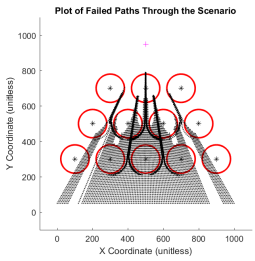

(a) All Failed Missions as Found by the Random Walk Method




(b) All Failed Missions as Found by the Waypoint Method



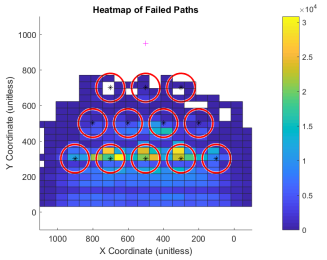

(c) All Failed Missions as Found by the Genetic Algorithm Method



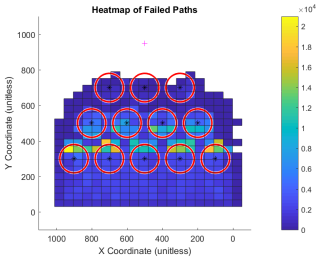

(d) All Failed Missions as Found by the Agent-Based Variables Method

Figure 4.38: All Failed Paths Found Through the Scenario

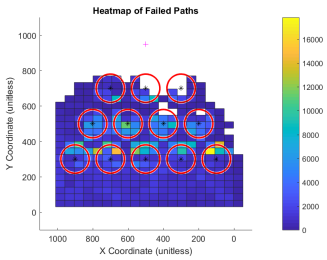
Qualitatively, the path heatmaps make sense as the regions with maximal representation (and thus where Attackers died most frequented) are those in the middle of a Defenders' radius of influence. Furthermore, as all the Attackers in this scenario failed, there is no representation of paths beyond the Defenders (e.g. a region beyond all impediments to mission success). These heatmaps are seen in Figure 4.39.



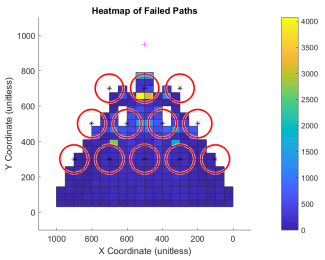

(a) Heatmap Illustrating the Regions Most Visited by Failed Random Walk Missions




(b) Heatmap Illustrating the Regions Most Visited by Failed Waypoint Missions




(c) Heatmap Illustrating the Regions Most Visited by Failed Genetic Algorithm Missions




(d) Heatmap Illustrating the Regions Most Visited by Failed Agent-Based Variables Missions

Figure 4.39: All Failed Paths Found Through the Scenario (Heatmap)

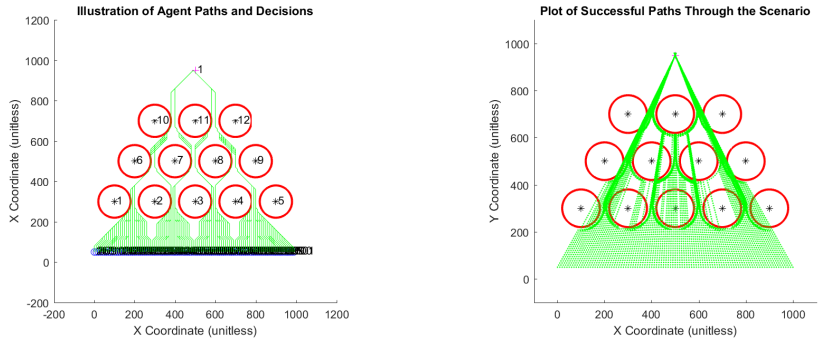
It is also worth noting and analyzing the particular regions where Attackers were more likely to die. The failed Waypoint and Random Walk methods illustrated an elevated "deadliness" for the first-line of Defenders. Those two methods did not involve intelligent entities and so if the selected paths took the Attackers through Defenders, it makes sense that Attackers would die sooner. In short, the simulated entities lacked any form of intelligence or understanding that they were in an non-ideal situation. Especially in the context of the Random Walk, most dead entities simply wandered into a Defender's aura and unintelligently remained there until death.

Failed ABV-based paths were weighted toward the "later" Defenders as Attackers ultimately lost against that final line of defense. Furthermore, the direct-line path(s) between Attacker start locations and the Target are evident. This is because the ABV-based formulation had Attackers follow this straight-line path unless the stochasticity dictated

an evasive maneuver around a Defender.

### 4.5.9 Deeper Comparison of the A* and Agent-Based Variable Approaches

The Agent-Based Variable exploration and $A^*$ optimization methods returned similar results. Even just comparing the maps graphically, there is a clear similarity as seen in Figure 4.40.

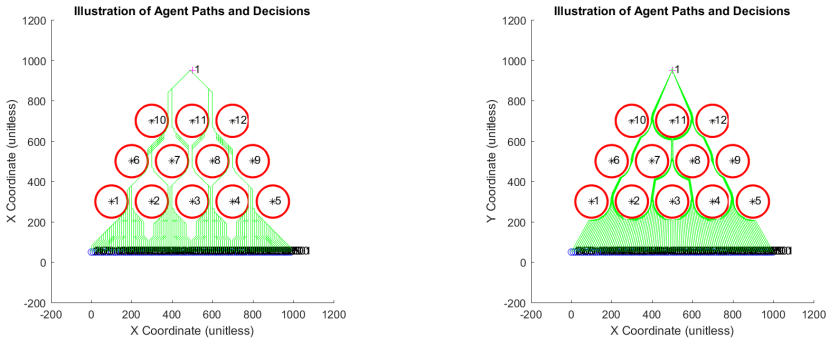



(a) All Successful Missions as Found by the $A^*$ Method

(b) All Successful Missions as Found by the Agent-Based Variables Method

Figure 4.40: Direct Comparison of the Successful $A^*$ and Agent-Based Variable Paths

The comparison of all successful paths though, is less interesting when considered against the ABV formulation with an Aggression Coefficient of 0. In this scenario, the Attacker agents are coded to always evade and avoid encountered Defenders. This behavior is effectively an in-mission implementation of the $A^*$ approach which sought similarly to avoid incurring the costs associated with traversing any defended region. This comparison is seen in Figure 4.41.




(a) All Successful Missions as Found by the $A^*$ Method

(b) All Successful Missions as Found by the Agent-Based Variables Method

Figure 4.41: Direct Comparison of Matched $A^*$ and Agent-Based Variable Paths

Breaking these two simulations down by starting point, the similarities remain interesting. All simulated missions by the two methods returned mission paths where the Attacker functionally never took any damage (health levels of 99 or 100). However, there is a valid comparison by time to mission completion as illustrated in the histogram of Figure 4.42. Notably, no ABV formulation had a higher success time than the optimal $A^*$ path.

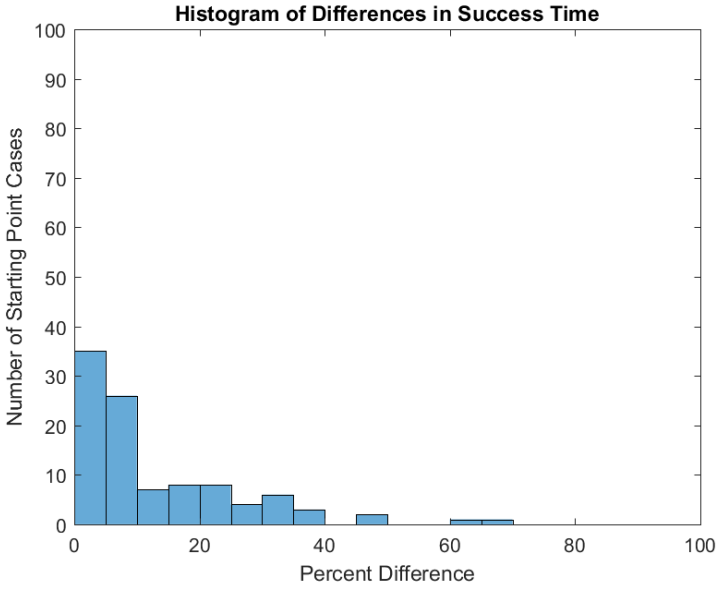



Figure 4.42: Histogram Illustrating the Differences Between the Mission Completion Times for the $A^*$ and ABV Formulations at the Same Starting Points

There is also a comparison of the fitness metrics defined by the OEC as illustrated by the histogram seen in Figure 4.43. As before, the optimal $A^*$ path was always superior than the ABV-obtained mission.

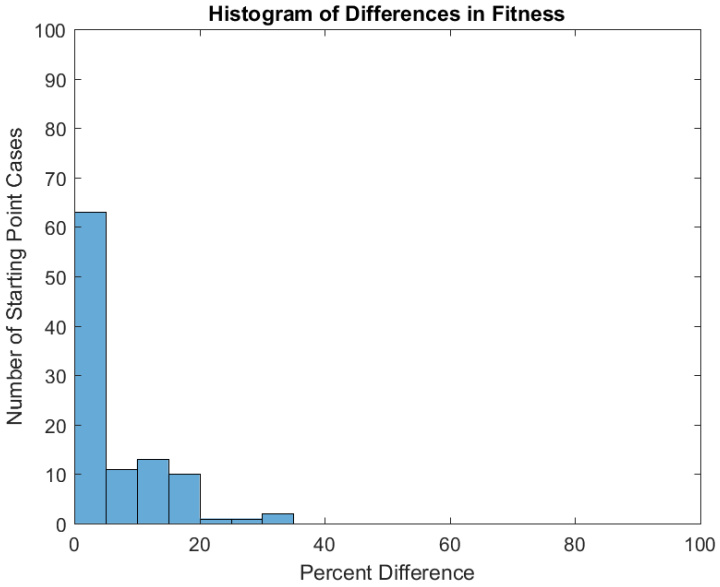


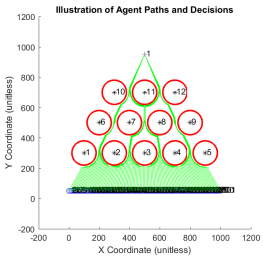

Figure 4.43: Histogram Illustrating the Differences Between the Fitness Levels for the $A^*$ and ABV Formulations at the Same Starting Points

These variations illustrated by the % difference histograms of Figure 4.42 and Figure 4.43 are generally small. Most of the variations are below 10%. Though the matches are not perfect, the general trends remain intact and the simulation time savings of more than 99% can more than justify the loss of fidelity or accuracy.

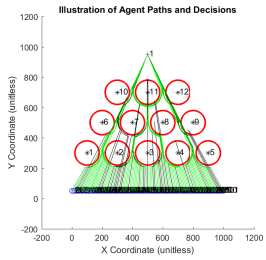
However, the strength of the Agent-Based Variable approach becomes clear when one notes that the above comparison is for a single ABV setting. In this case, the agents were instructed to avoid Defenders no matter what. This almost perfectly matches the $A^*$ optimization as that method similarly sought to perfectly avoid the high cost regions defined by the coverage from Defenders. Furthermore, the ABV method took only 3.74 seconds to run while the $A^*$ method took 533 seconds, an improvement of more than two orders of magnitude for the ABV approach.

When the values of the ABVs are changed, new mission paths open up. While the

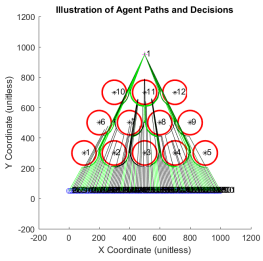
$A^*$ optimization perfectly avoided Defenders, the Agent-Based Variable method can introduce "bad decisions" as Attackers can now trade their survivability for faster time to mission success. A number of different ABV settings are seen in Figure 4.44.



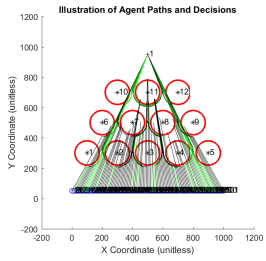

(a) The Mission Plans for Attackers with an Aggression Coefficient (ABV) of 0



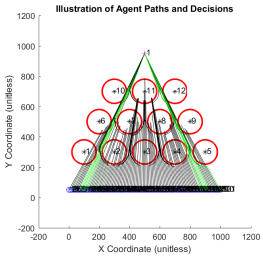

(b) The Mission Plans for Attackers with an Aggression Coefficient (ABV) of 0.2



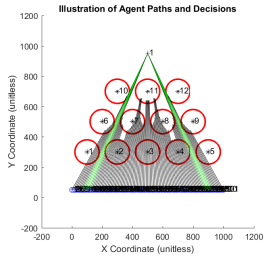

(c) The Mission Plans for Attackers with an Aggression Coefficient (ABV) of 0.4




(d) The Mission Plans for Attackers with an Aggression Coefficient (ABV) of 0.6




(e) The Mission Plans for Attackers with an Aggression Coefficient (ABV) of 0.8




(f) The Mission Plans for Attackers with an Aggression Coefficient (ABV) of 1.0

Figure 4.44: Various Mission Plans Found by Adjusting the Aggression Coefficient Agent-Based Variable Value

Thus it is clear that the "perfection" of an optimization-based approach actually yields an imperfect understanding of the engagement space. Since the simulation is coded to perform in way to guarantee success, the resulting approximation provides

less insight when decision-making parameters change. Furthermore, emergent mission behaviors illustrate how certain starting locations are robust for this particular fixed Defender placement. While the model is notional, the insights are an example of what can emerge when utilizing Agent-Based Models or stochastic simulations.

### 4.5.10 Value of the Agent-Based Approach

An advantage offered by a conversion to an ABV space is the improvement in scalability as entity types and numbers change. The dynamics of a mission undeniably shift as the design point adjusts, however some methods require a mathematical rework while an Agent-Based Model does not. Adding new agents of an existing and defined type is trivial in an ABM. Meanwhile, new agent types can share overall logic with other entities already simulated. However, for other representations like Markov Chains or the Lanchester Laws, there can be a multitude of chained effects from the introduction of new assets and asset types. Subject-matter expertise would be employed to map those effects onto the models, while an ABM does so automatically due to the decentralized entity control.

Additionally, the ABV method is not only more adaptable, it is also more extensible. For every change in the Defender locations of the mission region, a new optimization would have to be run, incurring those same higher costs. This dramatically increases the amount of initial simulation time necessary for generating a trade space. Furthermore, the optimization can only check Attacker routes and paths sequentially, though the final mission simulation can be done simultaneously. It also only generates mission plans which work for a singular mission region.

Meanwhile, the same variable combination which defines an ABV Mission Design Point could be reused for different scenario settings. For an Agent-Based Variable approach, identified design points can be attempted in alternate mission regions while still providing some degree of insight. Unlike a pre-defined path, the internalization of

the mission generation logic means that the mission region can change and the Attackers will react accordingly and in an adaptable manner. While additional simulations are necessary, like an optimization, the higher time efficiency from the **SAA** is more amenable to the additional cases.

Research Conclusion 2

Agent-Based Variables can be used to convert agent decision-making into a stochastic and continuous process, thereby enabling the use of more traditional sampling and model fitting techniques while still enabling the capture of optimal points and an understanding of the Mission Design Space.

## 4.6 Conclusion

Various methods exist to generate and consider different mission options when conducting a Mission Space Exploration. A major common weakness in the more straightforward generation methods is that it front-loads the mission planning. This *a priori* focus generates inextensible and non-robust results. While such plans work effectively when one side is passive, they break down in an active context. Furthermore, it is difficult to map discretized objectives and requirements into a method which can be effectively optimized without providing trivial solutions.

To bridge the scalability and flexibility gaps, it is appropriate to shift to an Agent-Based Model. Unlike the more generic EBCM, an ABM internalizes entity decision-making instead of relying on *a priori* planning. This decouples the Mission Design Space and the Mission Action Space. Now, the available Mission Design Variables affect agent decision-making and preferences while the Mission Action Variables still provide the conventional traceability of "which paths yielded specific results." Furthermore, the change in variable space decreases the number of unique design points by orders of magnitude. Though there is now a need for additional replications to under-

stand stochastic effects, there is no longer the intractability of millions upon millions of unique missions being articulated and iterated over directly.

The net result of having fewer design variables is a better ability to observe their effects and interactions. More simulation cases are now available to sample across each variable combination and axis, rather than the intractability of trying to explore a large-dimensional solution space. Additional samples can thus provide insights and better information.

# CHAPTER 5

# IDENTIFYING AGENT-BASED VARIABLES

> Look at situations from all angles, and you will become more open.
>
> - The $14^{th}$ Dalai Lama

## 5.1 Crafting a New Variable Type and Architecture

In a more traditional mission simulation, each action taken during the mission is tracked and articulated. Those variables compose the Mission Action Space, as discussed previously. Furthermore, these actions are often pre-specified and detailed as part of and frequently synonymous with, the Mission Design Space.

The conventional Mission Design Variables can thus be divided into two subsets: the Mission Architecture and the Mission Actions. The Mission Architecture is best articulated as the overall properties of the scenario in a more aggregate manner. For example, number and type of available assets are architectural. Other elements such as level of coordination and independence are harder to quantify but nonetheless apply to all entities on a given side.

The Mission Action Variables are the breakdown of the detailed actions of the scenario. Entity locations at certain timesteps, current targets, and firing orders are all related to the specific conduct within a simulated mission. Furthermore, it is the massive number of Mission Action Variables which necessitated the identification of a new variable space. For many forms of mission simulation (such as Discrete Event Simulation or empirical relationships), the Mission Action Space is avoided entirely.

The unique variable formulation for the proposed Mission Space Exploration focuses on the use of Agent-Based Variables which affect entity decision-making rather

than defining the actions directly. Previous approaches to affecting decision-making, like Dynamic Programming, effectively are a series of Boolean variables dictating which logic elements are incorporated into each agent in the model. However, ABVs ultimately convert these discrete or categorical decision-making spaces into continuous variables. However, not every parameter assigned to each Agent is an ABV. Some entity values are of interest for the Mission Space Exploration, but these are not unique to an ABV formulation. These properties are ones which define the scenario but rarely affect actions with the simulated mission. Example non-ABV parameters are listed in Table 5.1.

Table 5.1: Example Non-Agent-Based Variables to Define a Mission Simulation

| Parameter | Classification | Variable? |
|---|---|---|
| Number of Assets | Mission Architecture | Yes |
| Location of Facilities | Mission Architecture | No |
| Starting Location | Entity Properties | Yes |
| Sensor Range | Entity Properties | Fixed for a given entity |
| Number of Munitions | Entity Properties | Yes |

With so many variables available, it is necessary to identify what changes to the Mission Design Space are truly necessary in order to incorporate the proposed Agent-Based Variables. This need segues into Research Question 3.

Research Question 3

How can Agent-Based Variables be identified and utilized *a priori*?

### 5.1.1 Considerations for an Agent-Based Variable

Agent-Based Variables operate by converting deterministic agent behavior into a stochastic process. This means that the variables must directly affect the potential decision-

making of an entity. In short, an ABV defines a probability that a given action will be taken over another given the same stimuli. ABVs exist on a continuum from 0.00 to 1.00 and therefore can be seen as converting a categorical decision-making process into a continuous one.

Understanding where ABVs can be incorporated requires an understanding of the way decision-making is modeled. Two particular formulations are of interest for developing Agent-Based Variables: the State Machine and the Decision Tree. Both methods are popularly used in Agent-Based Models because they are comparatively easy to code and thus implement in a computer model.

It is also worth noting that both decision-making simulation approaches discussed here are of variable fidelity. In most models, assumptions are made regarding behaviors such that time is not wasted on tautological conditions. For example, an aircraft agent will not permanently fly in an unstable manner or stall. However, a higher-fidelity Agent-Based Model would consider such possibilities and decompose the decision-making to consist of specific control actions. This approach would be most applicable to letting an automated system "learn" to operate an asset [107]. Other methods are employed in the open-ended decision-making seen in Machine Learning [123].

For the purposes of developing the **Stochastic Agent Approach** and avoiding a computational explosion, the physics and decision-making of the problem will be of lower fidelity. Agents will be assumed to act in a safety-rational manner so that their stability in flight, inadvertent collisions, and other such concerns are ignored.

### 5.1.2 Decision Trees

Decision Trees are arguably the most simple form of simulated decision-making. Every layer of the Decision Tree can be represented as an if-then-else statement [127]. In other words, if a criteria is met, do action A, otherwise, do action B. This form of decision-making is the most common one seen in Agent-Based Models. If a need arises, a criteria

is met, or a target is identified, the appropriate action is taken. The previously discussed predator and prey model operates on exactly this principle of if-then-else statements as seen in the previously discussed Figure 5.1 and Figure 5.2.

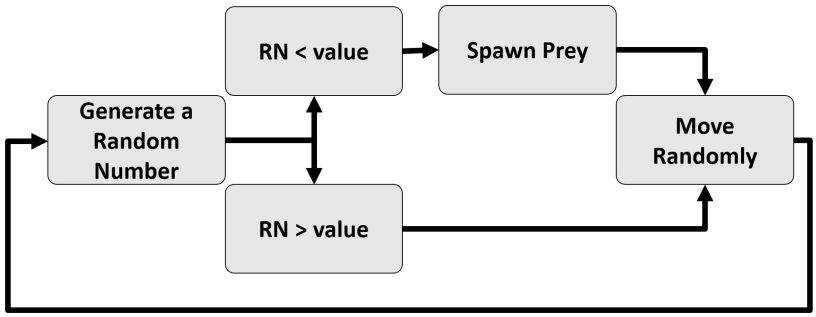



Figure 5.1: Prey Decision Tree Derived from the Notional Predator-Prey Example

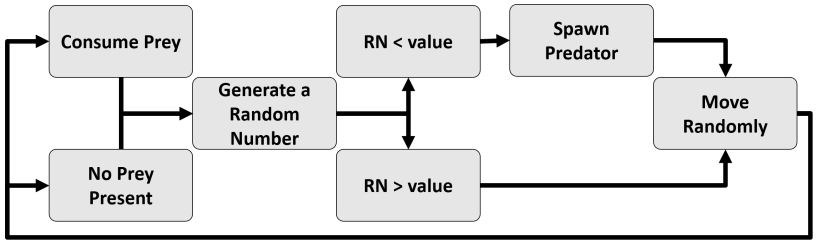



Figure 5.2: Predator Decision Tree Derived from the Notional Predator-Prey Example

Decision Trees, also referred to as “Game Trees” were a major topic of research by John von Neumann. However, increasingly complex representations of decision-making proved elusive as solutions would usually generate additional problems and complications. Furthermore, emotional and “irrational” influences on decision-making were frequently ignored or found nigh-impossible to simulate [60].

The power of the Decision Tree is when these if-then conditionals are linked into series. As criteria are evaluated, the Decision Tree branches into secondary evaluation criteria until a decision is made. The “nested” statements mean that a more appropriate degree of decision-making is simulated as conditions are sequentially evaluated and considered in turn (usually by order of importance, risk, or influence on resulting decisions).

Decision Trees, as they are already used extensively for Agent-Based Models, represent a helpful baseline for incorporating Agent-Based Variables. Furthermore, the decomposition of the decision-making process into a series of binary choices allows for easy incorporation of stochastic processes as a random number generator with an ABV cutoff-value for each relevant branch. While helpful for ABV inclusion, Decision Trees ultimately have a scaling problem tied to the limited branching and its comparative "memorylessness" wherein the Decision Tree must be traversed anew at every timestep.

### 5.1.3 State Machines

State Machines are used to detail system-wide changes in state in a usually deterministic manner [47]. In this formulation, a series of system "states" are linked by paths. Within this map is a "token" representing the current state of the system or process. The token moves between states and along paths deterministically based on inputs to the system. These inputs are usually represented as either user decisions (e.g. for the operation of a technology) or detection and reaction to stimuli (e.g. for the simulation of an autonomous system). State Machines are powerful for their ability to branch and handle a variety of possible scenarios. However, State Machines are less complete when utilized in a deterministic manner [47]. Though useful when approximating simple computer or autonomous systems, such a deterministic state-change representation does not fully encompass human decision-making and reactions. People rarely operate as consistently as a robot and so the deterministic representation of a State Machine can fail to encompass an appropriate diversity of solutions [48].

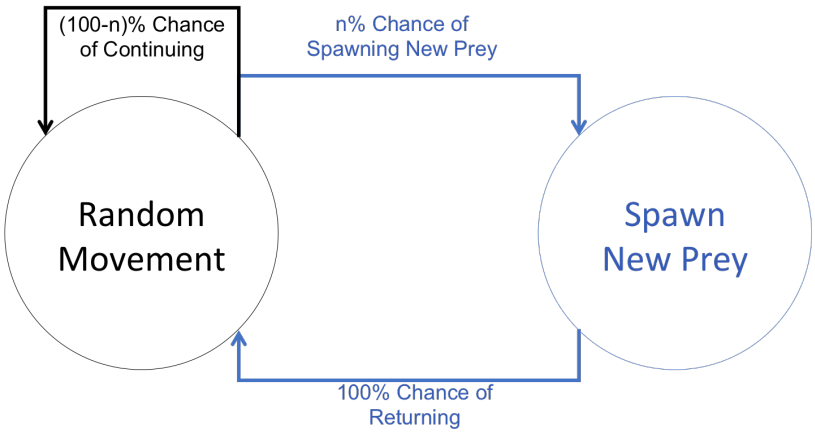



Figure 5.3: Prey State Machine Derived from the Notional Predator-Prey Example

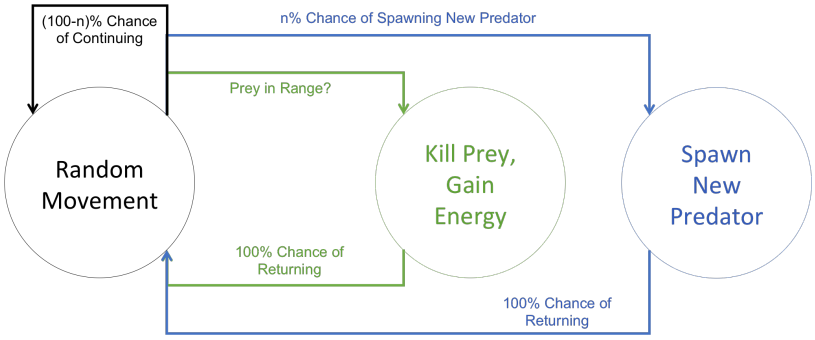



Figure 5.4: Predator State Machine Derived from the Notional Predator-Prey Example

Across these examples, Figure 5.1 and Figure 5.3, and Figure 5.2 and Figure 5.4 are effectively identical. They are inspired from the notional NetLogo predator-prey model discussed in preceding sections [92]. Regardless, these models are effective to understand. The State Machine has additional power over the Decision Tree by condensing numerous branches into states with multiple possible transitions in and out. Furthermore, the State Machine is aware of its current state and does not need to repeat the logic flow for each timestep. Instead, it only considers the transition states available from the current state.

### 5.1.4 Markov Chains and Petri Nets

Markov Chains and Petri Nets are sub-categories of State Machines [48, 124]. They operate on similar principles to the State Machine in that there are "states" and "tokens" moving along connections between those states as defined by conditional statements.

Markov Chains are used to detail system-wide changes in state in a probabilistic manner. In this formulation, a series of system "states" are linked by stochastic paths. Within this map is a "token" representing the current state of the system or process [48]. The token moves between states and along paths stochastically based on the probabilities of state change. For each time step, a random number generator determines whether a token will remain in its current state or transition to another state [124]. Notably, Markov Chains are used to represent complex dynamics and interrelationships, not necessarily individual system actions. Furthermore, they are stochastically driven rather than any form of decision-making as implemented in an ABM. The example Markov Chain illustrated in Figure 5.5 is for the weather of a particular location at a given time. Note that the state transitions are stochastic and only the current state affects the possible future steps.

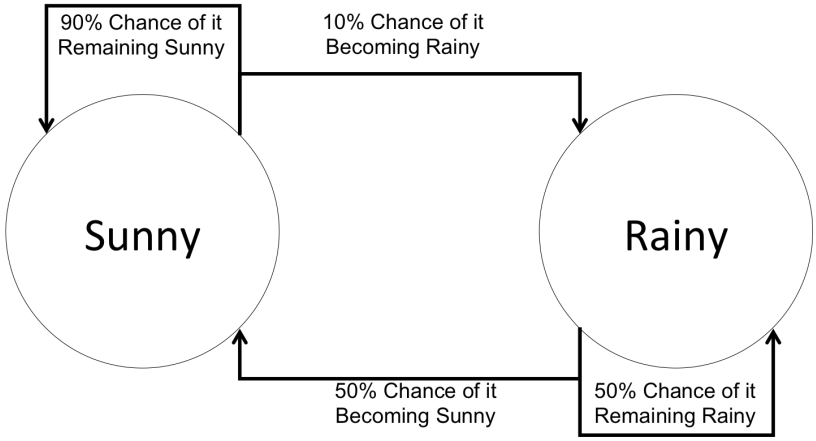



Figure 5.5: Notional Markov Chain Representation of Possible Weather Effects

Petri Nets more specifically operate in a stochastic manner with state changes [124,

128]. They are a graphical representation or implementation of the state changes, but share a common core with the "generalized semi-Markov process" [124, 128]. Petri Nets are more complex than State Machines in that they focus on the transitions and enable certain settings to "unlock" transitions as necessary [128]. However, the same complications arise in comparing a Petri Net to a more guided or reactionary decision-making system as required by an ABM, namely that there is little room for agency or effective external influences on the Petri Net behavior. However, Petri Nets are quite useful in safety simulations as illustrated in Figure 5.6. In this example, the risks to an aircraft from a failure of its Flight Control Computer (FCC) are modeled.

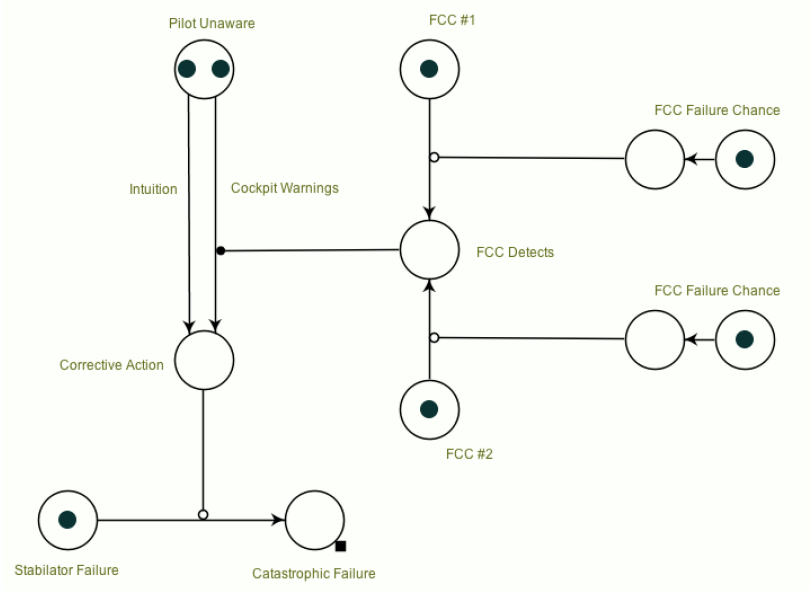



Figure 5.6: Notional Petri Net Representation Potential Risks from a Failure of an Aircraft's Flight Control Computer

The interactions inherent in the Petri Net are illustrated in Figure 5.6 in the form of the separate FCC failure probability affecting whether each of the two FCC systems transition into a failed state. That failed FCC could then affect whether the pilot successfully or unsuccessfully receives warnings about potential instabilities. The Petri Net is more interacting and complex than the Markov Chain, but that complexity might not be necessary for a basic Agent-Based Model. Furthermore, any changes to the simula-

tion dynamics, such as new assets, would necessitate a rework of the Petri Net.

### 5.1.5 Other Inspirations for Agent-Based Variables

Simulated Annealing is an optimization method which utilizes the slope and Hessian of a function to seek a minimum [38]. Notable in this heuristic is that the algorithm is allowed to "check uphill" with a certain frequency (decaying to 0% late in the method). Thus in the process of traversing and exploring the solution contours, the algorithm can check in a locally non-optimal direction with the hope that a global optimum could be identified in the future. Simulated Annealing is inspirational for the Agent-Based Variable method in that the stochasticity of ABVs enables the agents to act "locally non-optimally" but in a way which might yield better coverage and understanding of the solution space.

### 5.1.6 Conclusion

Agent-Based Variables are akin to a mix between the conventional decision-making approaches seen in Decision Trees and State Machines, and the stochasticity inherent in Markov Chains and Petri Nets. It is clear that for the latter two, the main difference is the introduction of probabilistic transitions unlike the deterministic ones of a conventional decision-making model.

Of further note is that no matter the decision-making model implemented, subject-matter expertise will be necessary. As these routines are central to the implementation of an Agent-Based Model, there will already be work completed implementing whichever architecture is selected. The hope is that the inclusion of ABVs scales proportional to the complexity of the agent behavior. Furthermore, the inclusion of ABVs should not require extensive work on the model beyond the initial arrangement.

However, bias remains an issue whenever subject-matter expertise is necessary. As cautioned by Robert Work, bias and narrowly scoping the mission space is a major risk

for wargaming [1]. However, the following sections aim to provide a straightforward method for classifying and understanding a simulation decision-making model. It is up to the end-user to identify branches in the Decision Tree or State Machine which are amenable to the infusion of Agent-Based Variables, but generally, stochastic decision-making is effective whenever a decision is not dictated by impartial model elements such as the laws of physics or direct causality.

## 5.2 Identifying Agent-Based Variables

While it is easy to discuss the general purpose and implementation of ABVs, it would be too complex to have every decision affected by an Agent-Based Variable. Furthermore, the fidelity of the model affects the fineness of the decision-making simulation required. A lower fidelity model by definition does not require full-fidelity maneuvering decision-making such as differentiating between slight variations on a "left turn."

For simplicity of visualization, a Decision Tree decision-making model will be used for illustrating the proposed method for Agent-Based Variable identification. The "layering" nature of Decision Trees lends itself to easy understanding of where ABVs can be incorporated and how the chaining between each layer of the Decision Tree could yield different results. Additionally, the Decision Tree simplifies the decision-making problem by remaining memoryless to a certain degree. While some past behavior could be considered at each decision level, the main driver in a Decision Tree is the current mission environment rather than what past action was undertaken.

Finally, there is no reason to think that a State Machine would not be amenable to inclusion, albeit with ABVs infused at the state transition points rather than Decision Tree branches.

### 5.2.1 Requirements

Agent-Based Variables are used in the context of a Decision Tree to distinguish between binary paths within an agent's decision-making logic. This means that the ABV must be a numeric value set between 0.00 and 1.00, inclusive. As for actual implementation, the ABV must be identified and the supporting previous and subsequent decision detailed. Furthermore, the Agent-Based Variable must be related to a decision of the appropriate fidelity. Lastly, the ABV should not be trivial in understanding. For example, an ABV that implies a probability of an aircraft allowing itself to crash into the ground rather than returning to safe flight, is meaningless. Additionally, there can be Decision Tree "branches" which do not actually branch and thus remain deterministic. These deterministic transitions are similarly not considered for ABV inclusion.

### 5.2.2 Proposed Approach

Given that Agent-Based Variables are built around the concept of Decision Trees, it is sensible to generate the Decision Tree for an agent first. After the logic is detailed, then different branches of the tree can be considered for reformulation into an ABV. This process yields Assertion 3.

Assertion 3

Agent-Based Variable selection can be enabled by composing a Decision Tree of all possible actions an agent may undertake. Each branch in the decision tree can then be classified as to whether it can be reasonably considered a stochastic (or preference-driven) selection. Stochastic decisions are the candidates for Agent-Based Variable infusion.

This methodology intends to address the major requirements for an ABV-selection method. By basing itself on the Decision Tree directly, the identified ABVs can be scaled based on the decision-making fidelity selected by the user. It also allows for effective un-

derstanding of the preceding and subsequent decisions in the Decision Tree depending on the model fidelity. This makes the **SAA** scalable to a variety of problems.

### 5.2.3 Identifying and Classifying Branches of Decision Trees

An effective method of defining a Decision Tree is to articulate all possible stimuli and conditions for the agents in the model. For example, what internal properties for the agent are being tracked, such as its notional health or its remaining ammunition? Does its location in space matter in a global or relative sense? Does it know the location of other entities and those positions relative to itself?

To aid with the identification of Agent-Based Variables, it is helpful to classify the types of branches and decisions within the Decision Tree. Similar to before, are the affecting stimuli internal, external, or positional? Furthermore, would the logic that informs one subset of decision-making influence other similar decisions?

#### *Navigational Category*

Navigational branches in the Decision Tree are directly linked with the Mission Actions undertaken in the simulation. For example, does an agent move left or right? Up or down?

In many cases, the navigational branches of the Decision Tree do not yield themselves to ABV incorporation. For example, in the absence of an impediment, it is reasonable to expect an Attacker seek the shortest route to its Target. Similarly, there should be nothing probabilistic about whether an agent will seek to avoid crashing into a barrier or element of impassible terrain which would result in certain destruction.

However, path-finding approaches could have a different ABV for each terrain type. How willing is an agent to traverse a waterway? How about rough terrain or a cliff? These considerations could result in a widely-branching Decision Tree which would rapidly eliminate the design variable space reduction achieved through the implementation of

Agent-Based Variables.

However, terrain traversal preference could be condensed into a single value representing the willingness to pass through difficult terrain, coupled with a function to convert each terrain type into a difficulty score. This method is not dissimilar from the simulation rework required for the inclusion of the $A^*$ path-finding algorithm seen in Chapter 4. Such traversal difficulty was also included by Kewley and Embrechts for their optimization of mission plans [57, 58]. While this reduces the dimensionality of the simulation, it also simplifies the variable space and thus the variable space scaling problem discussed previously.

*Relational Category*

Relational branches in the Decision Tree are directly linked with the basic elements of an Agent-Based Model, namely how agents interact with one another.

Similar logic or preferences for different decisions offers an opportunity to utilize ABVs more than once. Once more, poor planning and layout of the Decision Tree could cause the number of Relational Agent-Based Variables to increase uncontrollably. If there is a separate ABV for each opposing and allied agent, then the variable space is not particularly reduced. However, ABVs could be defined categorically as, for the purposes of an Agent-Based Model, an encountered agent of one type should be functionally indistinguishable from another of the same type.

Thus, rather than a separate Agent-Based Variable for each possible ally-foe relation, a single overall parameter can be selected. As seen in the original development of the ABV concept, an "aggression coefficient" could be defined which governs whether an agent evades or engages an encountered opponent.

*Trivial Category*

Trivial Decision Tree branches are ones where the outcome can be readily pre-specified or determined in-mission without a significant loss of understanding. Trivial decisions are ones where selecting one option over another is "common sense" or can be found through a quick relational consideration. For example, a naval agent should have a simple check to ensure that any selected course is through water sufficiently deep to enable safe operation. In another example, if an agent has to navigate around an obstacle, simple logic can dictate whether the maneuver is clockwise or counter-clockwise.

Logic and filtering on the Decision Tree also prevents a level of almost unnecessary fidelity. As illustrated in Figure 5.7, a full-fidelity breakdown of a maneuver as simple as "weighted-random" motion can generate a massive number of Agent-Based Variables with limited utility or understanding.

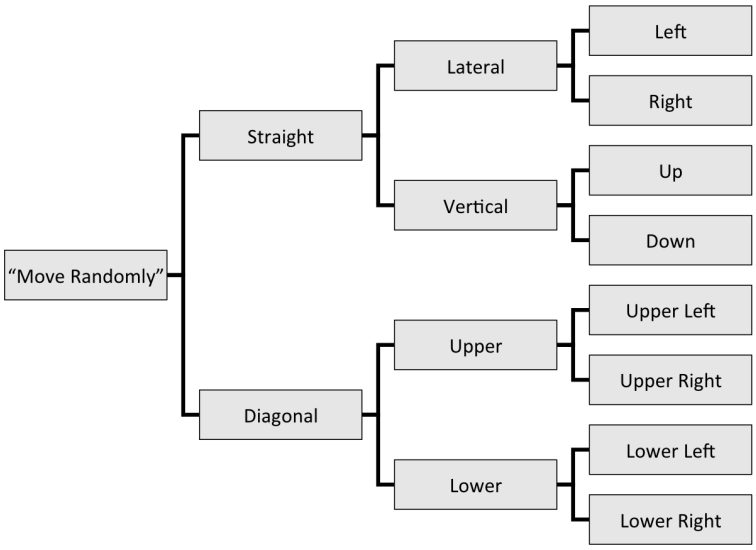



Figure 5.7: Notional High Resolution Maneuvering Decision Tree

Similarly high-fidelity Decision Trees could also exist when considering the full sequence of decision-making for a human-in-the-loop system. Usually in Agent-Based Modeling, the system is indistinguishable from the system operator(s) [15, 16, 129]. As such, any simulated decision-making can be abstracted to the level of entity actions

rather than operator-within-entity actions. A true high-fidelity Decision Tree could approach a representation more similar to a Petri Net, though that degree of action, agency, and interaction tracking could obscure understanding. As such, the **SAA** proposes keeping the agent Decision Trees at the system or entity-level, rather than modeling internal considerations.

*Completeness of Agent-Based Variable Categories*

The breakdown detailed in this section is intended to be complete regardless of the particular Agent-Based Model used or mission simulated. Decision Trees branch based on numerous criteria and the two primary ABV categories effectively encompass all agent-to-agent interactions (Relational ABVs) and all non-agent-to-agent interactions (Navigational ABVs). However, these classifications can certainly be further refined depending on the end-user and their goals. Regardless, the main purpose of the breakdown scheme within the **SAA** is to assist with understanding why simulated agents are making the decisions they are making. ABVs are infused at branches in the decision-making process where unit preference would come into play. As such, the key is identifying these branches and the specific classification of each assists with that process.

## 5.3 Illustrative Walk-through of Agent-Based Variable Generation

The most effective means of explaining the process is an illustrative breakdown of possible Agent-Based Variable inclusion in Decision Trees. Three notional agent decision-making systems are detailed: a path-finding algorithm, a strike mission, and a contested surveillance mission. These first two case studies are meant to exemplify the use of navigational and relational ABVs respectively. The third case study is effectively a fusion of the first two and thus synergizes the two types of Agent-Based Variable.

### 5.3.1 Path Finding

The path finding example illustrates how exploration can yield more understanding than optimization. In this example, the focus is on entity geospatial location rather than its interactions or relationships. This architecture means that any implemented ABVs fall into the Navigational category rather than the Relational category. In this example, the goal is find a low-cost path through a geographic region. There are competing objectives of timely arrival and low-cost transitions (e.g. minimal altitude changes). For example, the most direct route might involve traversing several hills and valleys, but saving time overall. Conversely, the route with least altitude change may stretch far away from the straight-line route, increasing time overall.

Regardless, these considerations are weighted in the $A^*$ and other path-finding heuristics which yield singular path results for a given geometry. However, such optimizations are not robust to changes in said geography and would have to be repeated for each new region or destination. Phrasing the problem in terms of agents and Agent-Based Variables could yield a more robust AI logic that would be region agnostic for finding a path through. In effect, the ABV combination would be a series of weightings on a depth-first search of pathways through the region.

For simplicity, the overriding assumption of this breakdown is that the agent knows its destination relative to its current position at all times. Thus it can properly weigh pursuing a direct (theoretically saving time) and a roundabout (theoretically saving transition cost) step as its next. Figure 5.8 is the overall decision tree for the case study.

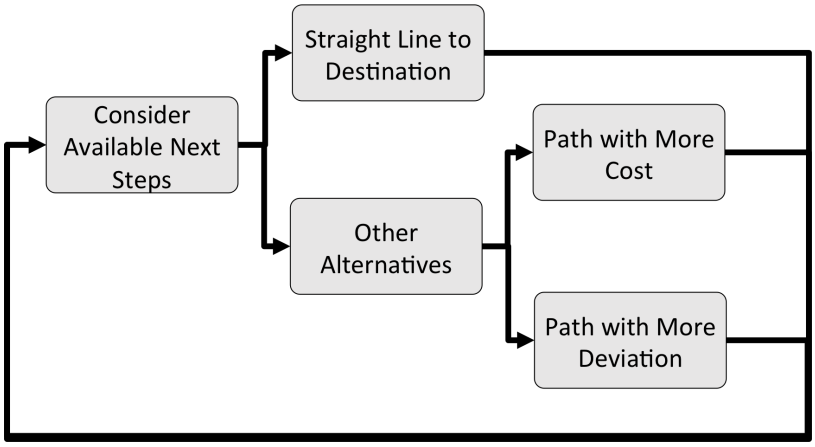



Figure 5.8: Decision Tree for the Path Finding Example

With the baseline Decision Tree seen in Figure 5.8, the next step is to break that illustration down into the trivial and ABV-compatible branches. There are no example trivial branches of the Decision Tree though the possible ABV inclusion locations are highlighted in Figure 5.9.

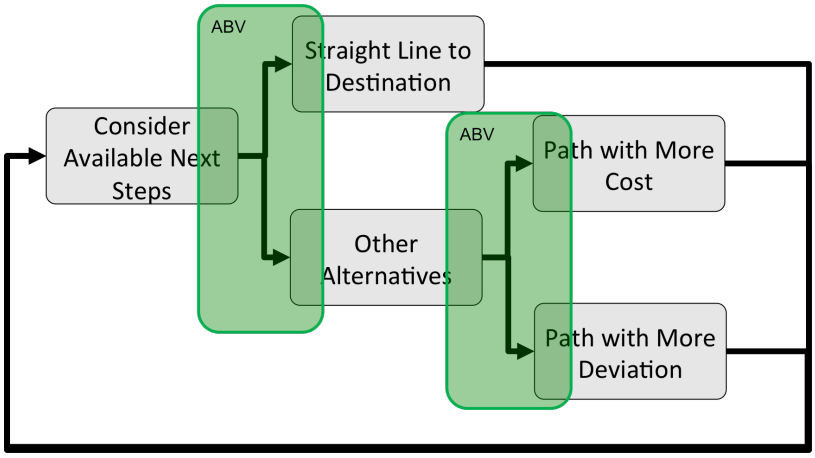



Figure 5.9: Identified Navigational Agent-Based Variables

The graphical analysis of the Decision Tree has yielded a pair of branches for logical Agent-Based Variable inclusion. However, that decision can include multiple ABV values as in this formulation, they are weightings on a fitness function of potential next steps. Again, this breakdown is similar to the Djikstra's and $A^*$ algorithms which consider both

transition cost and estimated future cost in selecting next steps [126]. Table 5.2 breaks down the identified ABV possibilities in the path finding example.

Table 5.2: Agent-Based Variables Identified for the Path Finding Example

| Agent-Based Variable | Description |
|---|---|
| Straight Preference | Preference for pursuing the straight-line route independent of cost |
| Height Preference | Relative cost of traversing changes in altitude |
| Lateral Preference | Relative cost of traversing changes in lateral direction |

In short, navigational ABVs should be evaluated by the model-creator as to whether the agent decisions are non-deterministic. Many navigational decisions are "common sense" such as avoiding truly impassible terrain (e.g. a car driving into the ocean). However, others are up to agent preference such as weighting different navigational metrics as related to an OEC's focus on time or cost. While some decision are more sensible than others, if the option is feasible, it should be a possibility depending on the value of the Navigational Agent-Based Variable.

### 5.3.2 Strike Mission

The strike mission example is derived from the illustrative example used to first describe Agent-Based Variables in Chapter 4. In this case study, the focus is on entity relationships rather than specific geospatial location. For example, it is less important where exactly an Attacker is than the fact that it is seeking a Target and evading (or not evading) Defenders.

The Decision Tree seen in Figure 5.10 illustrates the logic flow inherent in both this illustrative example and in the model detailed earlier in this report.

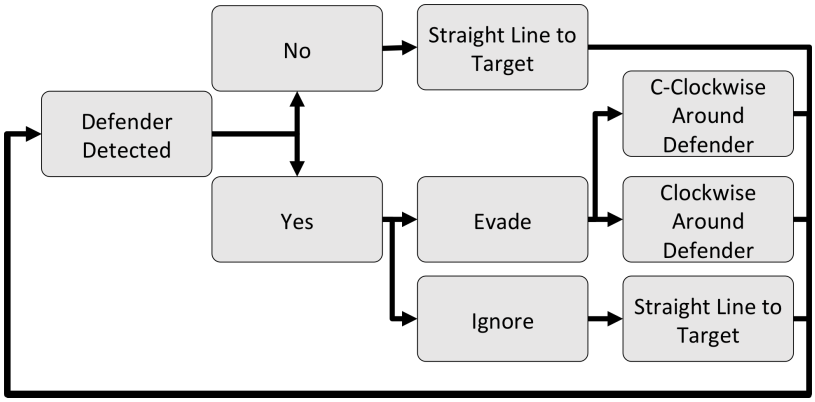



Figure 5.10: Decision Tree for the Strike Mission Example

As with the path finding example, the first step after crafting the Decision Tree is to filter out the trivial branches. In the Strike Mission Model, the most clear trivial branch relates to whether the Attacker evades clockwise or counter-clockwise around the Defender. As the Defender and Target locations are known, some trigonometric calculations can rapidly identify the most promising evasion direction to minimize the path to the target. The identified branches are highlighted in Figure 5.11 while an example of the trivial trigonometric logic used to replace the would-be ABV was seen in Figure 4.22

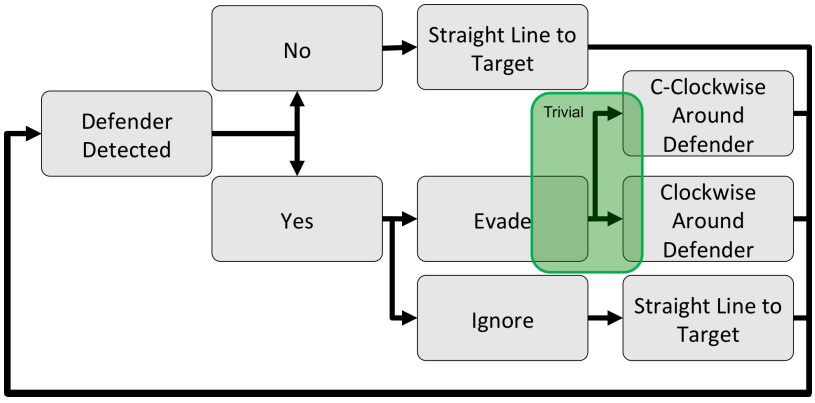



Figure 5.11: Identified Trivial Branches (No Necessary Agent-Based Variables)

With the trivial Decision Tree branches eliminated from investigation, the remaining branches are the candidates for ABV inclusion. This example is meant to illustrate a purely relational example of an Agent-Based Variable as outlined in Figure 5.12. The

Strike Mission Model does not have much in the way of simulated terrain and agents operate in a two-dimensional plane without features other than the presence or range of other agents.

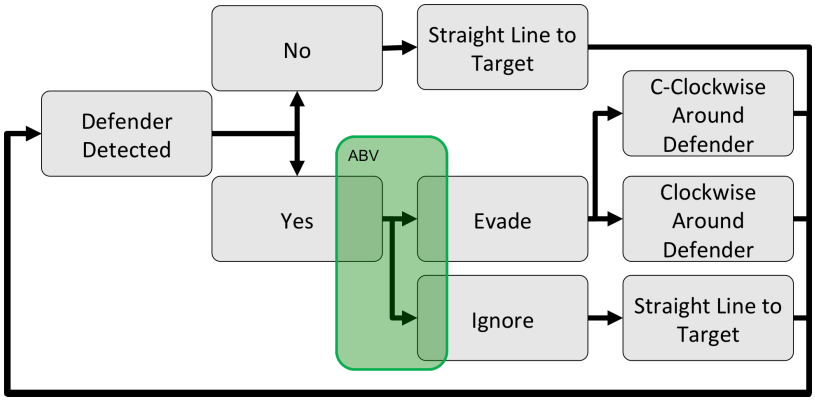



Figure 5.12: Identified Relational Agent-Based Variables

The sole relational ABV is christened the "Aggression Coefficient" and is articulated in Table 5.3.

Table 5.3: Agent-Based Variables Identified for the Strike Mission Example

| Agent-Based Variable | Description |
|---|---|
| Aggression | Preference for not evading a detected Defender |

This ABV effectively describes the probability of an Attacking agent "trying its luck" against a detected Defender. Much like the navigational Agent-Based Variable was a risk taken with regard to travel costs, this risk is taken in the hopes of trading an agent's health for a faster or more direct route to the Target.

In summary, Relational ABVs should be evaluated by the model-creator as to whether the agent decisions are a function of opposing agent positions and actions. Relational decisions tend to be more variable than navigational ones. Even in a notional search-and-destroy mission, the agent might not want to engage immediately depending on

other opponent positions or expected $P_{kill}$. As a result, it is expected that for a hybrid model, there will be more relational Agent-Based Variables since there are fewer "common sense" answers to a Decision Tree branch.

### 5.3.3 Contested Reconnaissance

The contested reconnaissance example is intended to combine the primary elements of the previous examples. The use of an intelligence-gathering mission makes navigational ABVs relevant. By conducting this collaborative mission in a contested environment, relational ABVs come into play. This model was derived from the Strike Mission Model by adding in a more general mission (no singular target), adding a third dimension to the maneuver space, and adding the ability for Attackers to engage Defenders rather than simply evading or ignoring them. The overall Decision Tree utilized is seen in Figure 5.13.

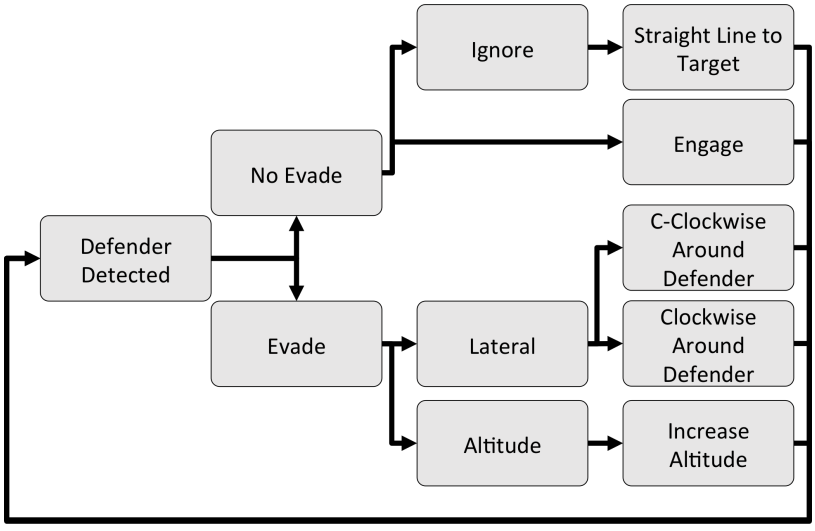



Figure 5.13: Decision Tree for the Contested Reconnaissance Example

As before, the first step of ABV identification is to eliminate the trivial decisions. These branches are highlighted in Figure 5.14.

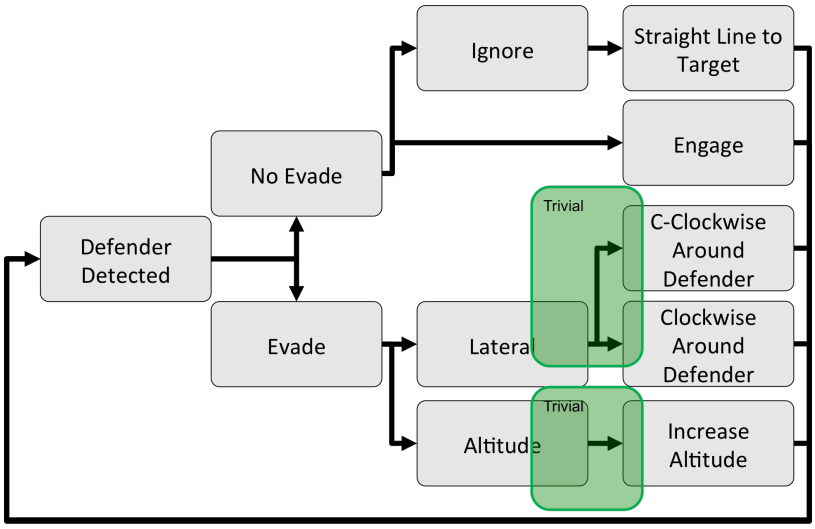



Figure 5.14: Identified Trivial Branches (No Necessary Agent-Based Variables)

As this model is ultimately derived from the Strike Mission Model, they share a common trivial Decision Tree branch in the form of the rotational direction around an evaded Defender. However, there is a new trivial branch that is not actually illustrated as such. Given that an agent has elected to evade a Defender by changing its altitude, there are theoretically two resulting options: increase altitude or decrease altitude. However, that decision is trivialized by the static ground-based nature of the Defenders. Furthermore, the Defenders have a hemispherical area of effect. Thus an evasive maneuver downward would more likely increase the risk to the Attacker by either bringing it into contact with the surface, or by closing its distance to the hemispherical area of effect of the Defender. This leaves an increase in altitude as the sole logical maneuver given a decision to evade by changing altitude.

With the trivial options pruned from consideration, the remaining ABVs can be classified as navigational or relational depending on the form of decision. The relational set is described first, as seen in Figure 5.15, because it includes the first branch of the Decision Tree.

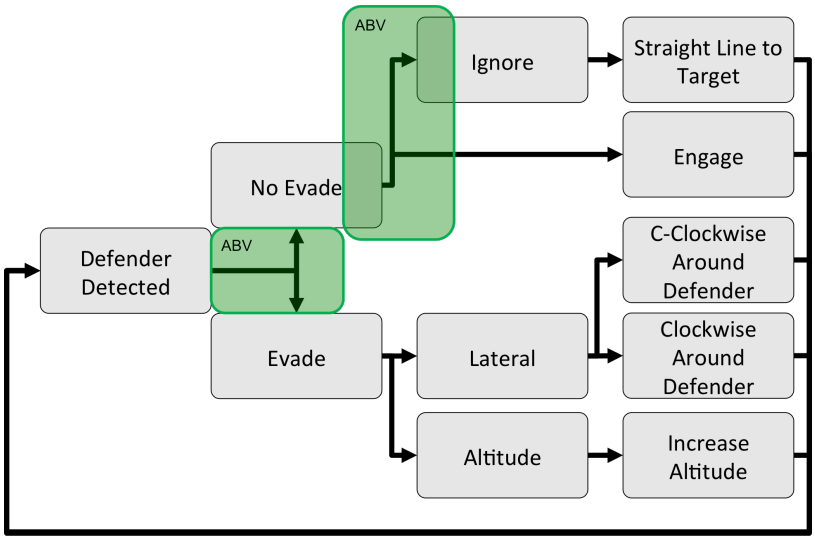



Figure 5.15: Identified Relational Agent-Based Variables

As with the Strike Mission Model, the first Relational ABV is whether an Attacker evades a Defender or not. However, there is a new Agent-Based Variable introduced in the non-evasive Decision Tree branch which presents the option of ignoring (as with the Strike Mission) or engaging and seeking to destroy the Defender. The remaining Navigational ABV is highlighted in Figure 5.16.

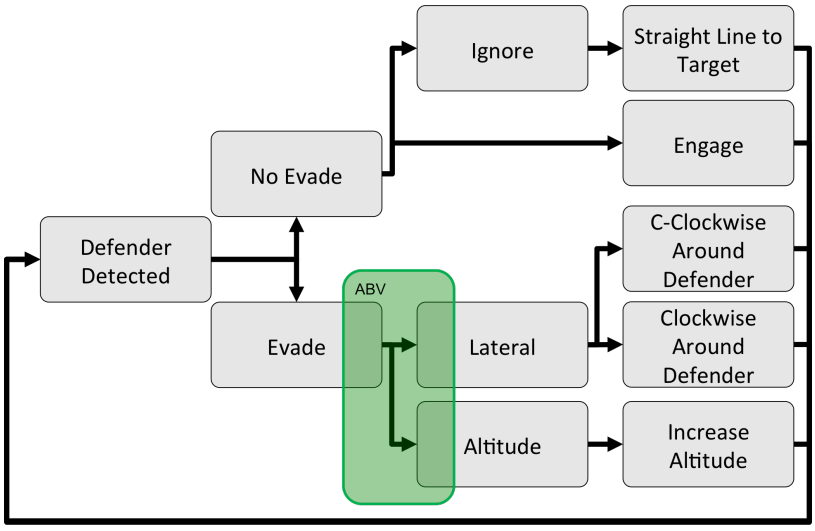



Figure 5.16: Identified Navigational Agent-Based Variables

The sole Navigational ABV is a newly introduced option to either fly around or over

the evaded Defender. Once more, this is a preference-derived maneuver decision. Going around the Defender enables superior sensor coverage of the reconnoitered space with a sacrifice of poor coverage into the region affected by a Defender. Conversely, flying higher can sacrifice some sensor coverage quality in favor of additional regions covered (those within range of a Defender). While the ABV could be considered relational, it ultimately exists after the Attacker-Defender relationship has already been decided to be evasive. Thus the Defender's position is less of interest than the particular maneuvers around it, making it a Navigational Agent-Based Variable.

The three Agent-Based Variables found using the Decision Tree branch breakdown method are described in further detail in Table 5.4.

Table 5.4: Agent-Based Variables Identified for the Contested Reconnaissance Example

| Agent-Based Variable | Type | Description |
|---|---|---|
| Aggression | Relational | Preference for not evading a detected Defender |
| Engagement | Relational | Given that a Defender will not be evaded, preference for engagement |
| Evasion | Navigational | Preference for flying over an evaded Defender |

The Contested Reconnaissance Model details all four classifications of Decision Tree Branches. There are several deterministic decision-less branches, two trivial branches, two Relational ABV branches, and one Navigational ABV branch. This model was eventually developed and expanded to generate the data necessary for Research Question 4 and Research Question 5 as discussed in Chapter 6 and Chapter 7.

### 5.3.4 Utility for Other Decision-Making Representations

The other canonical form of simulation decision-making is the State Machine. While the Decision Tree provides the most obvious breakdown of the decision options and the relationships between ABVs, it is not the only effective method. A State Machine has multiple transitions from and between different states (e.g. search state, engage state, cruise state) and those same transitions could be made probabilistic through Agent-Based Variables. In this implementation, the resulting decision-making process would be similar to the aforementioned Markov Chain process, but with the probabilities representing active decisions rather than truly random (or natural) processes.

However, State Machines are a little more difficult to cleanly articulate using ABVs, primarily because a single state might have multiple transitions whereas a Decision Tree can always be condensed into binary decisions. The binary decisions underpinning a Decision Tree make defining probabilities trivial as a single ABV is necessary to distinguish between the available branches. However, with appropriate formulation of the Agent-Based Variables or perhaps small Decision Trees within each state of the State Machine, **SAA** is feasible in a State Machine context. Future efforts involving ABVs should investigate their full utility for other decision-making systems.

## 5.4 Conclusion

Agent-Based Variables are ultimately defined and implemented by the creator of the model. However, their inclusion is comparatively simple provided the model has the ability to use random numbers. As any Agent-Based Model requires a developed model and a Decision Tree, State Machine, or other decision-making system, inclusion of ABVs is ultimately just slight adjustments on the logic of the transitions in each system. Converting the deterministic transitions to be probabilistic is all that is necessary to proceed with the **Stochastic Agent Approach**.

There are four possible classifications of Decision Tree branches when considering ABV inclusion. The first two are elements of the decision-making system where ABVs are not reasonable: deterministic changes and trivial branches. The former has no branching opportunity while the latter can be logically solved from the agent's state with little room for the "preferences" which ultimately encapsulate ABVs. The remaining two elements are Navigational and Relational Agent-Based Variables. Navigational ABVs relate to maneuvering decisions by an agent without direct consideration of opponent actions such as traveling over or around a physical obstacle. Conversely, Relational ABVs consider those interactions directly such as for combat.

The Decision Tree (or a derived method for other decision-making routines) Agent-Based Variable identification method is scalable to the fidelity of the model and the model creator's decisions. This makes it adaptable to changes in both the scenario and the behaviors of interest to the simulation evaluator. This adaptability makes the method robust to various code bases, simulation methods, and mission types. Furthermore, decision-making of assets is inherently semi-platform independent, provided adjustments are made for physics and other performance metrics. Thus ABVs are the critical development for the implementation of the **SAA**.

k

# CHAPTER 6

# DEFINING THE SCOPE OF A MISSION SPACE EXPLORATION

> Know thy self, know thy enemy. A thousand battles, a thousand victories.
>
> - Sun Tzu [105]

## 6.1 Framing the Problem

To develop an efficient approximation or model fit, there must be a satisfactory sampling of a representative data set. However, the definition of "satisfactory" in this context is less than definite. Furthermore, each variable input has a different effect on the variability of the outputs. Thus simply finding the Full Factorial combination of numerous points along every input axis is usually too "brute force" a solution and by definition, time-inefficient. These various concerns all inform and initiate Research Question 4.

> Research Question 4
>
> How many simulation cases are necessary to define a satisfactory Mission Space Exploration and how can those cases be identified?

### 6.1.1 Subdivision of the Research Question

Conducting a satisfactory exploration of a variable space is a function of two nested considerations.

The first concern is ensuring that the variability around each sampled design point is understood and addressed. In deterministic and noise-less models, this concern is not present. However, in the inherently stochastic approach using Agent-Based Variables

and an Agent-Based Model, such an understanding is necessary. To comprehend solution variability around a given design point, multiple replications of each design point are necessary. The stochasticity of the model generates distributions on the Measures of Effectiveness around that point. As the nature of mission simulation is so interconnected, a deterministic understanding of that distribution is nearly impossible. This leads to the identification of the first sub-Research Question.

Research Question 4a

How many replications of each Agent-Based Variable-involved design point are necessary to understand variability?

The second consideration is simply ensuring that the design variables are sampled in such a way as to bridge and encompass the entire variable space. The objective is to have a good sampling of both the inputs and the outputs, and is frequently obtained via an intelligently selected architecture of a DoE. If satisfactory coverage is not obtained, there is an incomplete understanding of connections between inputs and outputs. This understanding is necessary for development of an appropriate Surrogate Model. Consequently, there is a second element of Research Question 4.

Research Question 4b

What Design of Experiments most amenable for use with Agent-Based Variables?

These two sub-research questions will be further refined and subdivided later in this chapter. The final questions will be defined by a numeric-numeric designation (i.e. 4.1 rather than 4a).

### 6.1.2 Objectives

The objective of any variable and solution space exploration is the have a full understanding of both the input and output spaces [38]. However, the stochastic nature of

the Agent-Based Model implies that the output Measures of Effectiveness are not singular values. Rather, there are expected distributions whose mean, variance, and other metrics are affected by the design variable settings of that particular case.

In short, the stochastic model takes a deterministic Mission Design Point and through numerous replications, yields a distribution of the MOEs expected [20]. As the resulting metrics are themselves distributions, the objective is to create an approximation of the values needed to populate those distributions.

The final representation is intended to be as "black box" as possible with generic utilized inputs. While the ABVs and Mission Architecture variables are available to a decision maker at the campaign level, precise opponent locations are not. Thus the resulting model intends to provide an understanding of robust allied mission plan decisions at the campaign level, irrespective of opponent actions and decisions other than purely architectural parameters (like number of opponents). This precedent is set by the canonical Lanchester Equations whose primary design variables consist of the strengths (or numbers) of the two opposing sides and their relationship alone [14, 17].

### 6.1.3 The Contested Reconnaissance Model

The Contested Reconnaissance Model is the most complex mission model developed as part of this research. It is an expansion of the Strike Mission Model developed to illustrate the initial efficacy of Agent-Based Variables. The agent logic for this model was detailed previously as the capstone example of how to identify possible ABV infusion locations in a Decision Tree (see Chapter 5). To further the stochasticity and avoid improper emergent behavior, the random number seed was tied to the current time and thus would not be repeated between each run of the model.

The model represents an force of Attackers attempting to sense across a contested region. The region is 1000x1000 units and is subdivided into 10x10 unit subregions as seen in Figure 6.1.

Each Attacker has an on board sensor which covers a radius around the Attacker. Surveillance coverage is notional on a scale from 0 to 100. The sensor has a minimum range within which, the coverage is 100. From that minimum radius out to a maximum radius, the sensor coverage quality decays quadratically to 0 at maximum range. Thus at every timestep, the region's coverage quality is updated based on the proximity to the Attackers. Coverage does not decay over time. The overall objective is for the Attackers to obtain coverage such that 80% of the subregions have a coverage quality of 50 or greater. An example of the coverage from a single Attacker is seen in Figure 6.1. The green line is the Attacker's path through the space while the coverage quality transitions from blue (zero coverage) to yellow (full coverage).

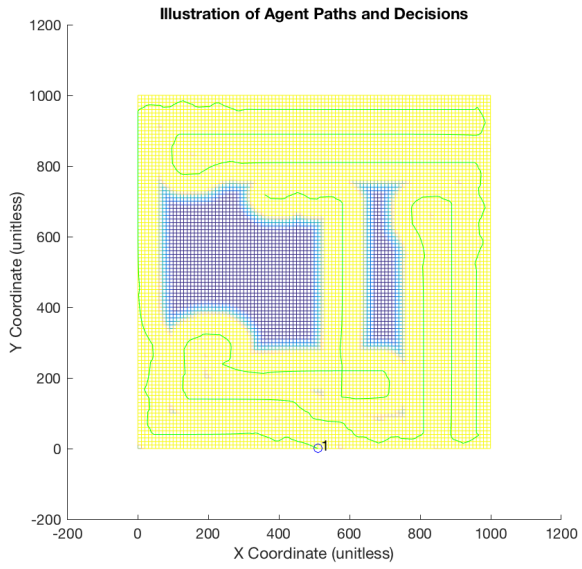



Figure 6.1: Simulated Coverage from a Single Unimpeded Attacker in the Contested Reconnaissance Model

These Attackers are modeled with perfect, nigh-omniscient communications where there is a shared knowledge of what regions have been surveyed. These agents set their baseline destination to be the closet region with a coverage quality less than 50, the cutoff for "sufficiency" for the victory condition. Additional decision-making logic af-

fects how they react to encountered Defenders. This logic is most similar to the strike mission example but with newly added caveats of either direct engagement or three-dimensional evasion options. The overall Decision Tree is seen in Figure 6.2.

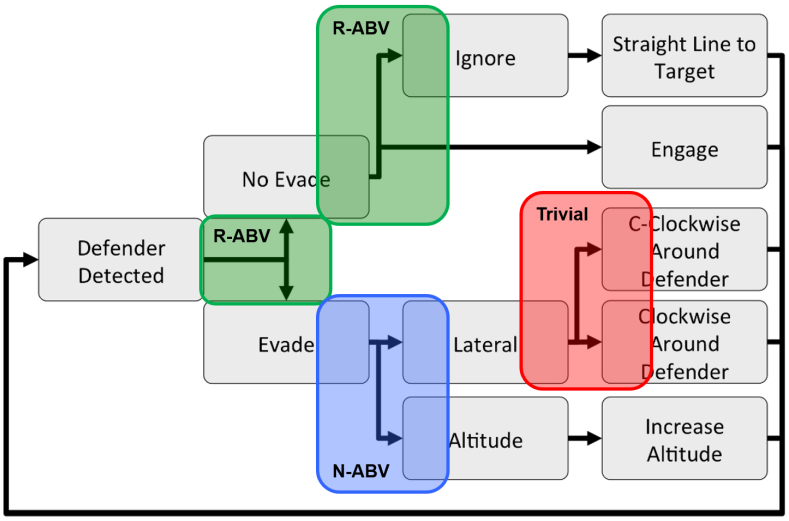



Figure 6.2: Decision Tree for the Attacker Agents in the Contested Reconnaissance Model with ABVs Highlighted by Type

The Attackers have a branching logic first depending on an ABV known as the "Aggression." This ABV determines whether an Attacker passively evades a Defender.

If the Attacker elects to evade a Defender, then it is presented with an maneuver choice of evading laterally (tangentially) or increasing its altitude and trying again. Going around the Defender enables superior sensor coverage of the reconnoitered space with a sacrifice of poor coverage into the region affected by a Defender. Conversely, flying higher can sacrifice some sensor coverage quality in favor of additional regions covered (those within range of a Defender). This decision is governed by the "Evasion" ABV.

If the attacker does not evade a Defender, then it is presented with a choice of ignoring or engaging that Defender. Ignoring a Defender is exactly as described, the Attacker continues with its reconnaissance mission irrespective of that particular Defender or any incurred damage. Meanwhile, combat for an Attacker engaging a Defender is deter-

ministic. If the Attacker has the available ammunition and gets within its attack range, the Defender will be destroyed. The Attacker will then resume its reconnaissance mission as per the decision tree seen in Figure 6.2. This decision is affected by the "Engagement" ABV.

The static Defenders operate much like in the strike mission example by projecting damage against Attackers within a radius around their location. As part of the development of a stochastic model, Defender locations were randomized for each replication, to ensure a degree of portability for a developed Mission Design Point.

From the different considerations described here, there are a total of eight (8) design variables for the Contested Reconnaissance Model. Three of these variables are the ABVs for Aggression, Evasion, and Engagement. Two Mission Architecture parameters are also introduced and consist of the numbers of entities on each side in the mission (as inspired by the Lanchester Laws). The remaining three variables are specifically related to the Attacking side. Two of these variables describe the starting location of the Attackers by the x-coordinate of the starting point and the width of the formation. The y-coordinate is defaulted while the Attackers are equally spaced laterally along the formation width centered on the assigned point. The remaining variable is the amount of "ammunition" assigned to each Attacker, or more plainly, how many Defenders could an Attacker engage within a single simulation. A breakdown of these variables is also seen in Table 6.1.

Table 6.1: Design Variable Descriptions for the Contested Reconnaissance Example

| Variable | Type | Description |
|---|---|---|
| Number of Defenders | Architecture - Defenders | |
| Number of Attackers | Architecture - Attackers | |
| Ammunition (Ammo) | Architecture - Attackers | Number of Defenders a single Attacker can engage per mission |
| Attacker Xmid | Architecture - Attackers | Center point of the Attackers' formation |
| Attacker Xrange | Architecture - Attackers | Width of the Attackers' formation |
| Aggression | ABV - Relational | Preference for not evading a Defender |
| Engagement | ABV - Relational | Given that a Defender will not be evaded, preference for engagement |
| Evasion | ABV - Navigational | Preference for flying over an evaded Defender |

A more complete region example with Defenders and Attackers is seen in Figure 6.3. The Defenders are the black stars with the red circles denoting their effect range. The Attackers are the blue circles toward the bottom of the image.


Figure 6.3: Example Starting Conditions for the Contested Reconnaissance Model

An end-state criteria was set to 80% sensor coverage of the region while an overall simulation time limit was set to be 2000 timesteps. These limits were imposed to keep the simulation runtimes reasonable and to avoid asymptotic or "stuck agent" conditions.

Figure 6.4 provides an example of the coverage amount from the Attackers' sensors for the duration of the mission. The mission simulated consisted of four Attackers and ten Defenders. The paths of the Attackers are in green while the color scale reflects the quality of sensor coverage for the various regions.

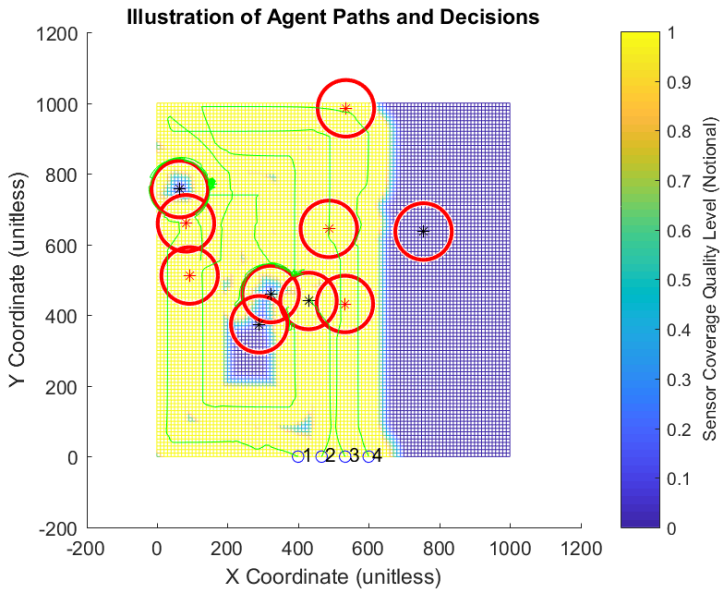



Figure 6.4: Example Surveillance Coverage for the Contested Reconnaissance Model

Figure 6.5 provides an example of how the Attackers' health levels change over the course of the mission, ranging from full health (yellow, 100/100) to dead (blue, 0/100).

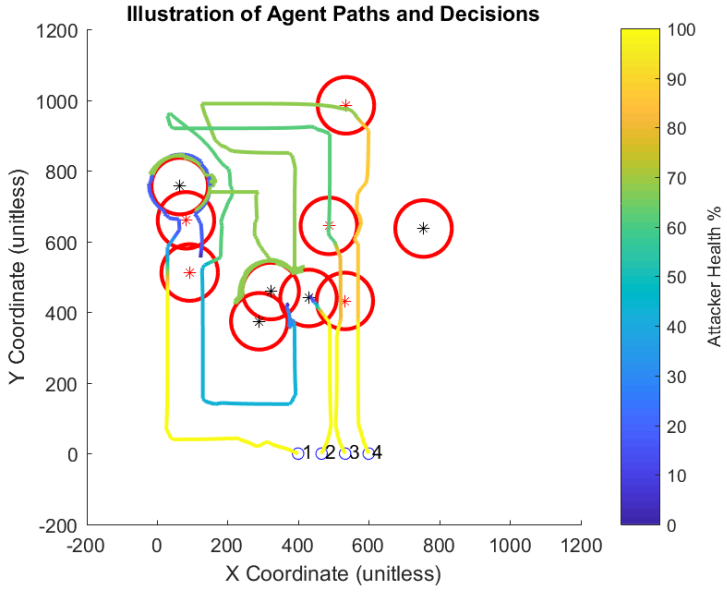



Figure 6.5: Example Attacker Health Over Time for the Contested Reconnaissance Model

Both Figure 6.4 and Figure 6.5 also show the end-states of the Defenders. A Defender

marked by a black asterisk is still alive while a Defender marked by a red asterisk has been killed.

### 6.1.4 Utilizing the Model

With the Contested Reconnaissance Model developed, the next objective was to define the various metrics of interest. The first metric of interest is the overall outcome of the mission, namely the % surveillance coverage achieved by the Attackers. Some coverage is always attained, even in a scenario where all Attackers are killed. Thus that coverage amount is the critical MOE. An additional two metrics of interest were identified, namely the survival rates of the Attackers and Defenders. The breakdown of the output metrics and their desired values is provided in Table 6.2.

Table 6.2: Measures of Effectiveness for the Contested Reconnaissance Example

| Variable | Desired Value |
|---|---|
| Attacker Survival Rate | Maximize |
| Defenders Survival Rate | Minimize or Ignore |
| Reconnaissance Coverage | Maximize |

These rates and values are all ultimately distributions within a given case through a to-be-determined number of replicates. Thus the three MOEs are not simply the value at each replicate of every design point. Instead, the values of interest are the distribution metrics for each, such as the mean ($\mu$), standard deviation ($\sigma$), skew, and kurtosis. These four values can be used to define a distribution. Traditionally, only two metrics are necessary if the shape of the distribution is known (e.g. normal, beta, Weibull) but as there is no guarantee of a particular solution probability density function, these generic metrics are the key parameters.

## 6.2 Determining the Necessary Number of Replications

Before any attempt is to be made at crafting a Design of Experiments for an ABV-involved Mission Design Space, there must first be an identification of the number of necessary replications to describe the distributions on the output MOEs. Too few samples of the stochastic simulation could yield misleading results or an improper understanding of the true underlying distribution(s) [100, 101].

### 6.2.1 Design Point of Highest Variability

This experiment is centered on finding the necessary replicate number. To err on the conservative side, the first objective is to bound the necessary number of replications. In a trivial model, all simulated missions yield the same MOEs. An example for the Contested Reconnaissance Model would be a scenario with no Defenders. As a result of no Defenders being present, there is no Attacker logic or stochasticity actually employed as the agents simply fan out and survey the regions until mission completion is achieved.

The goal of this experiment was to see which design points (cases) had the highest variability for the critical MOE, the surveillance coverage amount. This objective is described by Research Question 4.1:

> Research Question 4.1
>
> What is the Mission Design Point of highest outcome variability?

In the highest variability design point, there would be minimal trend toward one outcome or another. Thus, it would be expected to require the most replicates to capture the truth distribution.

*Considered Mission Design Points*

By definition, the upper bound on the necessary number of replications would be tied to the most variability-generating combination of design variables. Given that the **SAA** relies on Agent-Based Models, any point of highest stochasticity must account for agent decision-making.

The primary design variables to affect stochasticity are the ABVs and can be identified from consulting the Decision Tree used to set up the agent logic. The tree for the Contested Reconnaissance Model consists of three stochastic branches and four possible (ABV-informed) end states: ignore, engage, evade laterally, and evade by altitude. For maximum variability in agent decisions, each branch should be equally likely as the others, which implies that setting all three ABVs to 0.5 would accomplish that goal. That combination of ABV values would yield a uniform 25% chance of an agent acting according to each Decision Tree branch.

This informed a testable hypothesis which effectively claims that high simulation-variability is driven by high agent-level action and decision variability.

Hypothesis 4.1

Simulation variability is maximized when the Agent-Based Variables are set to result in an equal likelihood of each Decision Tree Branch being utilized.

For the non-ABV parameters, higher stochasticity should arise from enabling more random interactions between agents. As such, the remaining design variables should be set to maximize the probability of interactions. By increasing the number of Attackers and Defenders, the number of interactions and thus possible stochastic actions is increased. Maximizing the ammunition for each Attacker should keep its decision options available. Lastly, centralizing the start location and maximizing Attacker spacing enables the largest amount of possible paths as the Attackers do not start clumped to-

gether or against the region boundary. For the parameters selected, the highest variability is hypothesized to be in the subset of conditions described in Table 6.3.

Table 6.3: Maximum Variability Settings for the Contested Reconnaissance Model

| Variable | Value for Maximum Variability | Justification |
|---|---|---|
| Number of Defenders | Maximum | More Defenders should yield more decisions required of the Attackers |
| Number of Attackers | Maximum | More Attackers requires more decisions |
| Ammo | Maximum | More options for an Attacker to engage |
| Attacker Xmid | Centered | More options for an Attacker to laterally maneuver |
| Attacker Xrange | Maximum | A wider formation should enable the Attackers to more easily span the region |
| Aggression | 0.5 | A binary, equal-probability choice should result in more stochasticity |
| Engagement | 0.5 | A binary, equal-probability choice should result in more stochasticity |
| Evasion | 0.5 | A binary, equal-probability choice should result in more stochasticity |

With the defined hypothesis, the next step was to run a series of experiments to check the variability of the different Mission Design Points. Variability of each design point was defined by the highest variance among the output metrics of interest. For reference, the point of lowest simulation variability should be for an entirely deterministic Agent-Based Model. The highest possible simulation stochasticity in the Contested Reconnaissance Model would occur when the output metrics are most evenly distributed.

With the bounds on the ABV space identified and a point of highest variability hypothesized, the next step was to conduct an exploration experiment on various Agent-Based Variable settings.

A complete exploration would consist of a full-factorial combination of the three (3) ABVs each with thirteen (13) possible settings. These variable settings are described and commented on in Table 6.4. The values were selected to increase the sample density around the hypothesized maximum variability point while still maintaining some sampling and understanding of the whole range of values.

Table 6.4: Agent-Based Variable Settings for the Exploration

| Value | Justification or Comments |
|---|---|
| 0.00 | Lower Bound |
| 0.25 | |
| 0.40 | |
| 0.45 | |
| 0.48 | |
| 0.49 | |
| 0.50 | Hypothesized Maximum Stochasticity |
| 0.51 | |
| 0.52 | |
| 0.55 | |
| 0.60 | |
| 0.75 | |
| 1.00 | Upper Bound |

A full-factorial combination of these ABV settings would yield 2197 unique cases. However, that number of cases is too high for efficient simulation and so a degree of

pruning was implemented. 125 cases were run using a three (3) ABV with five (5) settings full-factorial around the middle of the variable ranges ([0.48, 0.49, 0.50, 0.51, 0.52]). A 27-case full-factorial was run over a lower resolution combination of ABV settings ([0.25, 0.50, 0.75]). Lastly, an 27-case full-factorial was run for the "corner cases" of various combinations of [0.00, 0.50, 1.00]) to complete the sampling.

Each case was replicated 1000 times to approximate the "truth" distribution and properties for each case. The Defender locations were randomized while the random number seed was tied to the current time and thus would not be repeated between replications or design points.

*Evaluating Selected Design Points*

The goal of this experiment was to see which design points (cases) yielded the highest variability in outcome. Listing all 179 cases would result in an intractable table. The result was that the comparison was forced to be more graphical in nature. Line-plot histograms of all considered design points are seen in Figure 6.6.

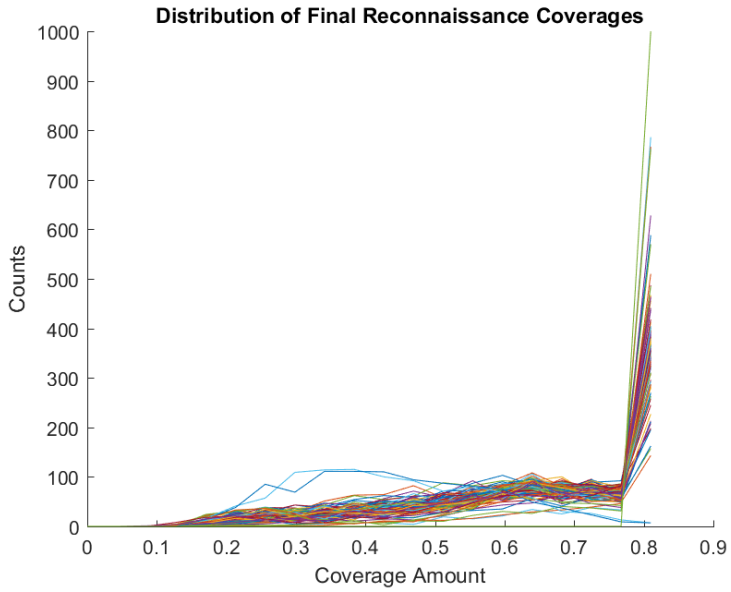



Figure 6.6: Overlaid Histogram of the Proportional Coverage for All 187 Design Points

Due to the trio of Full-Factorial DoEs used for this experiment, there were three (3) different cases with ABV values of (0.5, 0.5, 0.5). These three cases are illustrated in Figure 6.7. Notable in this figure is that the three design points are close to similar in distribution, thus confirming that the 1000 replicates is a sufficiently high number to generate "truth data."

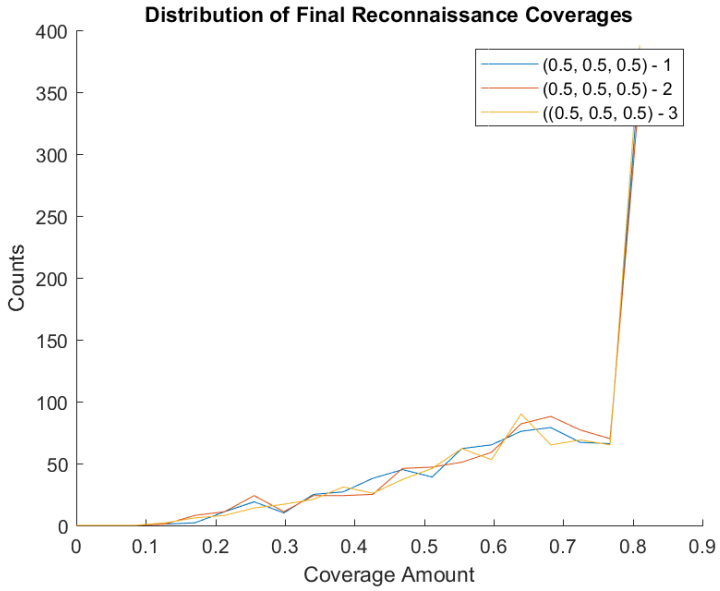



Figure 6.7: Overlaid Histogram of the Proportional Coverage for the Three (0.5, 0.5, 0.5) Design Points

Notably the general trends of all the considered Mission Design Points were similar. However, two design points stood out from the bulk. They were the ABV combinations of (1, 0, 0) and (1, 1, 0). These two, when investigating the Decision Tree as illustrated in Figure 6.2, are functionally identical. Both have a 100% Aggression value, unused Evasion value, and 0% Engagement value, respectively. This means that the Attackers always ignore the Defenders, but then also do not fight back. The result is that a more distributed coverage % is achieved, but also done so solely because every Attacker is killed at pseudo-random times. These two design points are illustrated in Figure 6.8 as compared against the average histogram of the three repetitions of the (0.5, 0.5, 0.5) design point (from Figure 6.7). Figure 6.9 provides a pruned set of histograms for select

ABV combinations.

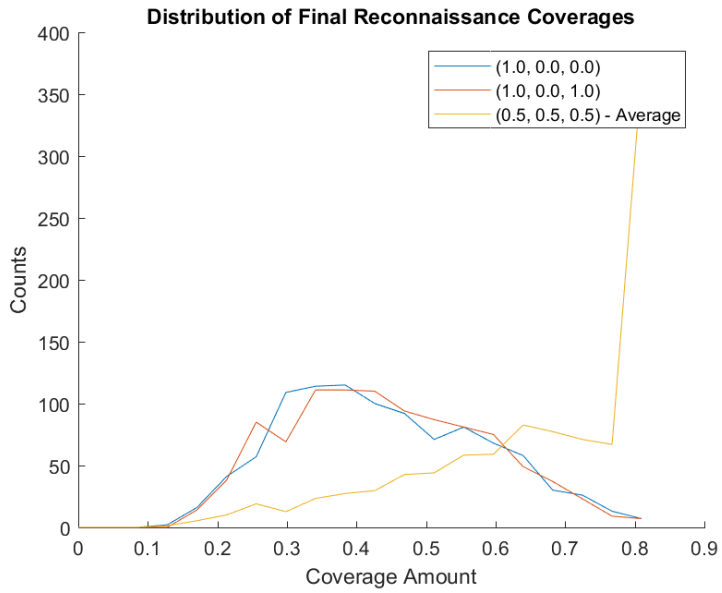



Figure 6.8: Overlaid Histogram of the Proportional Coverage for the Outlier Design Points

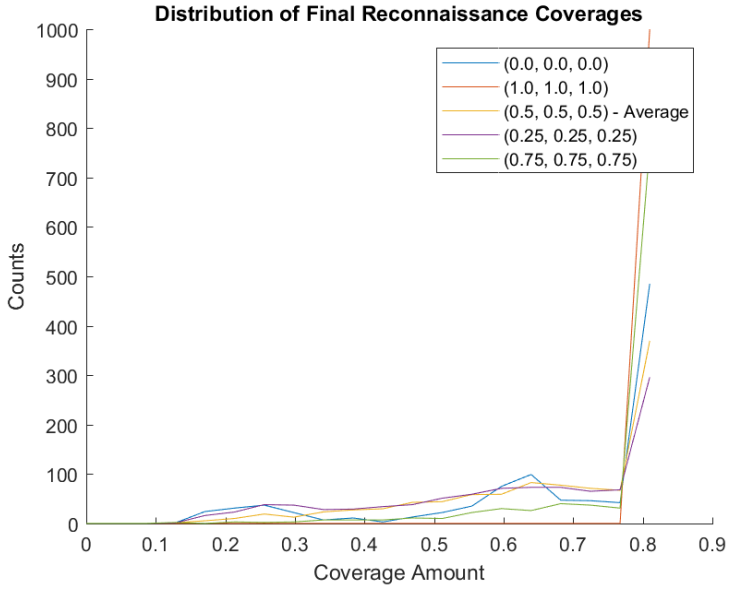



Figure 6.9: Overlaid Histogram of the Proportional Coverage for Selected Design Points

The most notable conclusion from Figure 6.6 is that all of the design points have relatively similar histograms and none approach a high variability distribution without

becoming trivial. This trend is driven overall by two competing forces at each extreme. Low coverage values are not possible because it takes a certain amount of time before the Attackers can engage (and die) against a Defender. Meanwhile, there is a limit to the coverage possible due to the simulation end-state. Even if that end-state was not imposed, surveillance coverage cannot exceed 100% of a notional region and similar high-end behavior would have been expected, albeit at a different peak value.

As the various histograms were all similar, it was unclear which design point is the most variable while remaining relevant. However, the overall objective for this experiment was to identify a design point which can be used to provide an upper bound on the number of necessary replicates per case. As many cases seem similarly variable, it ultimately appears to be a large range in which the design point approaches the solution space's "highest variability."

Research Conclusion 4.1

Simulation variability is similar at a variety of Agent-Based Variable defined design points. Thus the selection of any of these points is acceptable for generating an upper limit on the necessary replicates for each case.

One point of "high-enough" stochasticity should be enough to identify the necessary maximum replicate number for every Mission Design Point. That resulting replicate count is by definition conservative, however, it is simpler in the simulation to keep the number of replications for each design point constant. Extra replicates cost time but do not impede the results and if anything improve the understanding of the resulting MOE distributions. Future efforts can seek to better quantify design point variability, but the overall approach of identifying a select few points to quantify the necessary replicate number remains valid.

### 6.2.2 Identifying Maximum Required Replicate Number

With the variable settings defined, 1000 replications were needed to generate a "truth" dataset. As described previously, the Defender locations were randomized while the random number seed was tied to the current time and thus would not be repeated. The "rule-of-thumb" for replicate number in stochastic simulation fields ranges from 10 in traffic simulation through 100 in Cognitive Modeling [130, 131]. More specific experiments have been conducted for individual stochastic models and applications including past projects in the Aerospace Systems Design Laboratory (ASDL) which settled on a replicate number of 50 [20]. Meanwhile, the Bootstrapping statistical method (discussed in detail in a later section) is an appropriate means of identifying the necessary replicate number for a stochastic simulation. This context informs Hypothesis 4.2.

Hypothesis 4.2

Bootstrapping is an effective method to identify convergence. In most scenarios, fifty (50) replicates is sufficient to address and describe the stochasticity in the output metrics of interest for an Agent-Based Variable Mission Design Space.

*Simulating the Replications*

The Mission Design Point selected was found from the previous experiment to confirm a point with high variability in results and MOEs. To simplify the problem, the Agent-Based Variable point of (0.5, 0.5, 0.5) was selected. The resulting combination of inputs is listed in Table 6.5

Table 6.5: Design Variable Settings for Identifying the Necessary Replicate Number

| Variable | Value |
|---|---|
| Number of Defenders | 10 |
| Number of Attackers | 4 |
| Ammo | 4 |
| Attacker Xmid | 500 (centered) |
| Attacker Xrange | 100 |
| Aggression | 0.5 |
| Engagement | 0.5 |
| Evasion | 0.5 |

With that combination of inputs, the 1000 replications at that design point were re-used from the previous experiment.

*Bootstrapping the Output Metrics of Interest*

The net results from the 1000 replicates was the assumed "truth data" for the design point. With these overall Measures of Effectiveness identified, the next step was to consider how those values converge based on replicate number. The Bootstrapping method was used to identify a "sample size inflection point." Bootstrapping works by re-sampling a larger dataset in smaller amounts with replacement [132, 133]. Each sample has a resulting mean, variance, and other statistical properties. All the samples taken at a given sample size can then be compared and a resulting distribution on these distribution properties can be generated and investigated. As the sample sizes increase, their properties should slowly approximate that of the full "truth" dataset. Bootstrapping is more computationally efficient that running additional simulations because it involves the reuse of an existing sample set [132, 133, 134].

From the 1000 replicates, 100 samplings with replacement were taken of various sizes. Those sample sizes ranged from 10 to the full 1000 in steps of 10. The 100 samples at each sample size point yielded a distribution of the various MOEs. These distributions progressively narrowed towards the true distribution as the sample size increased. Notably, the parameters of interest are the mean, variance, skewness, and kurtosis of each sampling.

The distributions for the means of the Measures of Effectiveness are seen in Figure 6.10, Figure 6.11, and Figure 6.12 with the x-axis displaying the sample size and the y-axis displaying the mean value for each of the 100 samples.

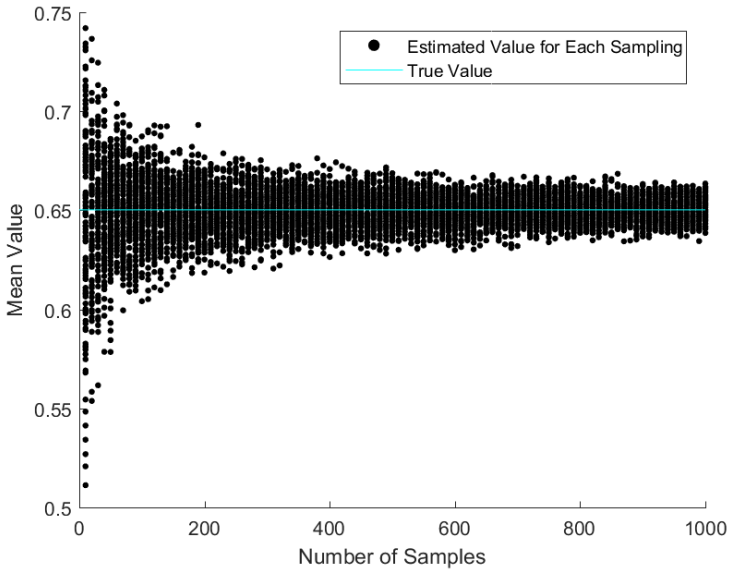



Figure 6.10: Mean of the Coverage Bootstrapping Distributions for Each of 100 Samples Taken at Each Sample Size (X-Axis)

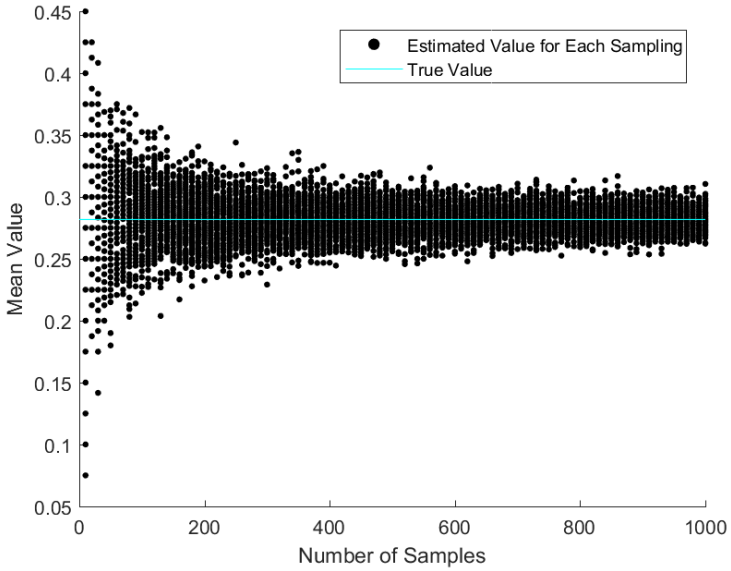



Figure 6.11: Mean of the Attacker Survival Rate Bootstrapping Distributions for Each of 100 Samples Taken at Each Sample Size (X-Axis)

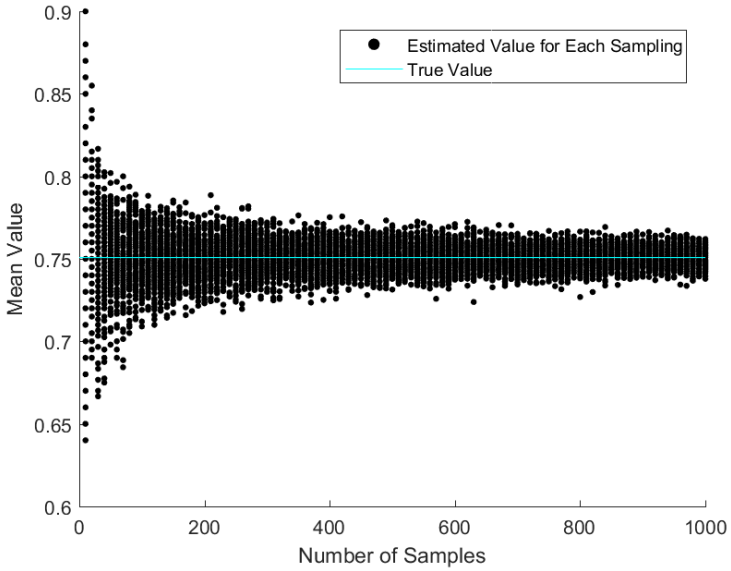



Figure 6.12: Mean of the Defender Survival Rate Bootstrapping Distributions for Each of 100 Samples Taken at Each Sample Size (X-Axis)

From these plots, it is clear that there is some form of inflection point around a replicate number of 200. However, visual inspection is not a robust method of evaluation.

The Coefficient of Variance (CoV) ($\sigma/\mu$) is used to describe the distribution by taking the ratio of the mean ($\mu$) and standard deviation ($\sigma$). The mean and variance are less useful in isolation because both are necessary to describe the output distributions while their overall relationship and ratio is more enlightening. For example, a small variance does not necessarily imply a tighter distribution if the overall sample range is similarly small. Thus the mean helps "normalize" the variance for the particular distribution being considered.

A smaller CoV implies a "tighter" distribution and, in this context, one which has converged to some distribution. The objective was to observe at what point the Coefficient of Variance dropped below a certain threshold or approximately converged to some value. The plot of the CoV for the distribution at each sample size can be seen in Figure 6.13, Figure 6.14, and Figure 6.15.

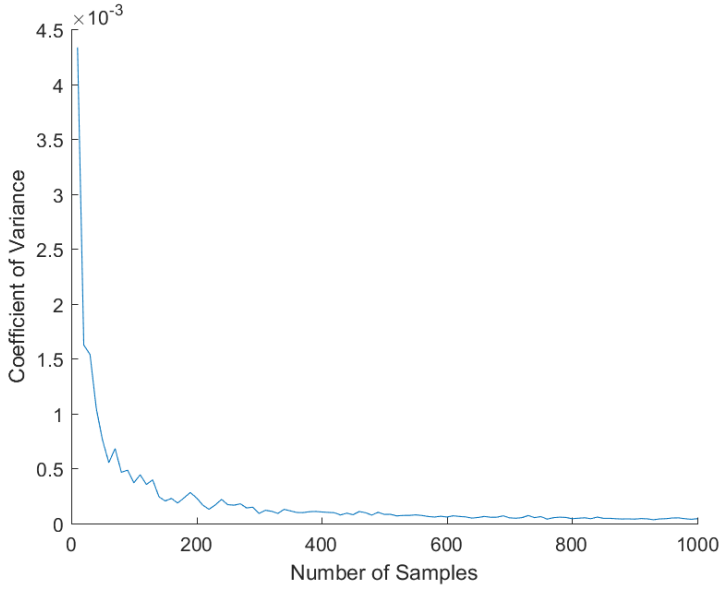



Figure 6.13: Coefficient of Variance for the Coverage for 100 Samples Taken at Each Sample Size (X-Axis)

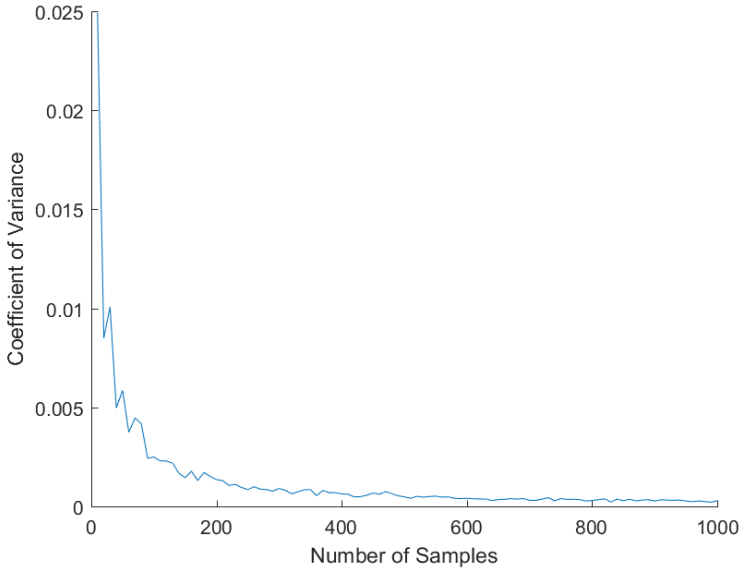



Figure 6.14: Coefficient of Variance for the Attacker Survival Rate for 100 Samples Taken at Each Sample Size (X-Axis)

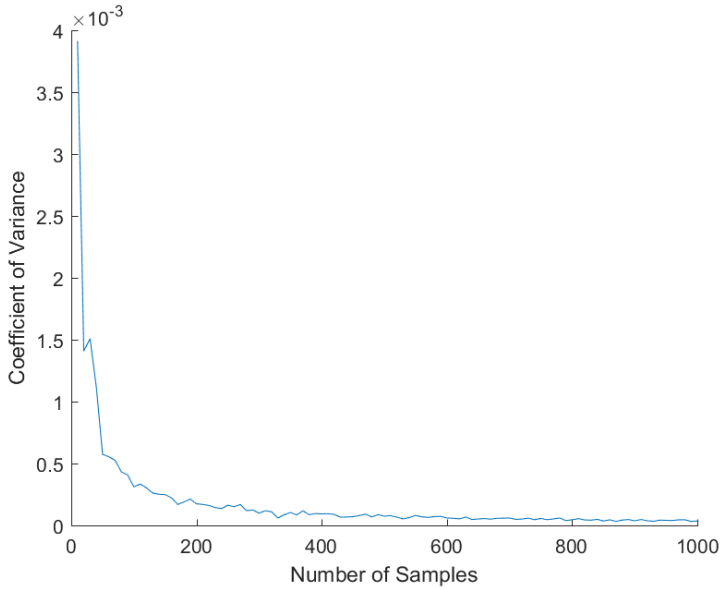



Figure 6.15: Coefficient of Variance for the Defender Survival Rate for 100 Samples Taken at Each Sample Size (X-Axis)

For all three MOEs, the Coefficient of Variance appears to stabilize absolutely in value around replicate 400. However, further investigation implies that a lower repli-

cate number of 200 is sufficient to capture the majority of the convergence behavior. Regardless, these values are higher than the "rules of thumb" for simulation replicates from both literature and industry. However, that breakdown is sensible given that the ABV-involved Mission Design Variables directly affect stochasticity in a way that previous simulation methods did not.

### 6.2.3 Conclusion

Replications are necessary to achieve an understanding of how well different stochastic ABV-based mission models perform. The results from these experiments illustrate that the required replicate number for an Agent-Based Variable Mission Design Space is actually higher than for a conventional design space. Hypothesis 4.2 was proven partially incorrect as a replicate number of 50 (the "rule-of-thumb") was too low to capture the necessary information about the simulation output distributions. This is most likely because the ABVs directly impact the simulation stochasticity rather than indirectly as with Mission Architecture or Engineering Design Variables. Consequently, Assertion 4.2 is rephrased to focus on the method of identifying an appropriate replicate number.

Research Conclusion 4.2

The existing literature rules-of-thumb do not seem valid for the necessary replicate number, though the Bootstrapping statistical method can identify it for an Agent-Based Variable-involved model

However, this assertion is not universal. Other simulations utilizing ABVs may have different replication requirements. The **Stochastic Agent Approach** illustrates that Bootstrapping and investigation of the Coefficient of Variance is an appropriate method to describe the convergence of the distributions on the simulation MOEs. Thus while 200 replicates is appropriate for the Contested Reconnaissance Model, future applications of this method may identify other values.

## 6.3 Identification of Appropriate Design of Experiment Method

With the number of necessary Mission Design Point replications found, the next step is to identify and employ a satisfactory Design of Experiments to sample across the ABV-based Mission Design Space. Design of Experiments methods vary in applicability depending on the design variable types involved or the intended use of data from the DoE.

> Research Question 4.3
>
> What Design of Experiments method is appropriate for an Agent-Based Variable-involved model and variable space?

### 6.3.1 Survey of Existing Methods

Developing an appropriate DoE is a broad topic wherein researchers seek to identify methods to sample across complex variable spaces. Numerous methods are available in literature.

*Full Factorial*

A Full Factorial Design of Experiments can be considered the "brute force" approach to the problem. This method decomposes the various input variables into discrete settings of the user's selection. Then all possible permutations and combinations of the variable settings are used to populate the cases [40, 135]. The number of unique solution cases from a Full Factorial is seen in Equation 6.1 where $n_i$ is the number of settings for variable $i$.

$$\begin{aligned} Cases &= n_1 * n_2 * n_3 * ... * n_{k-1} * n_k \\ Cases &= n^k \end{aligned} \tag{6.1}$$

Generally, the number of cases scales exponentially with the reasonable assumption

that each variable is assigned the same number of settings. For example, 3 variables of 10 settings each requires 1000 unique cases for the Full Factorial. Due to the exponential complexity scaling, this method works well with a small number of variables and a small number of settings per variable. However, the problem scales uncontrollably when higher resolutions are required, as would be the case in many continuous variables. Though useful as a representation of a "perfect" sampling approach, the scalability issues with a Full Factorial minimize its utility.

*Orthogonal Arrays*

Orthogonal Arrays are among the simplest archetypes of DoE. They decompose the various input variables into discrete settings, often just two consisting of the maximum and minimum of a pre-defined range. This converts each design point into effectively a binary string where each variable can be "high" or "low." Mathematical relationships or defined tables (such as Taguchi arrays) can then be used to craft the necessary cases for understanding of variable effects [40, 135]. Table 6.6 illustrates an example of an Orthogonal Array for three (3) variables of two (2) settings each.

Table 6.6: A Three Variable, Two Setting Orthogonal Array

| Variable 1 | Variable 2 | Variable 3 |
|---|---|---|
| High (1) | High (1) | High (1) |
| High (1) | Low (-1) | Low (-1) |
| Low (-1) | High (1) | Low (-1) |
| Low (-1) | Low (-1) | High (1) |

Orthogonal Arrays are not ideal for the generation of approximations of higher-fidelity solution spaces [40]. Their use is limited to understanding trends rather than sampling across and filling-in the full and often complicated contours of the solution space.

*Classical Designs*

Among the most popular "classical" DoE forms are the Central-Composite and Box-Behnken [40, 135]. Classical designs aim to minimize the influence of random error on the trends and behavior of sampled results. This often leads to making more samples toward the edges of the design space than in the interior [135]. An example classical DoE (Central Composite) for two (2) variables is seen in Figure 6.16.

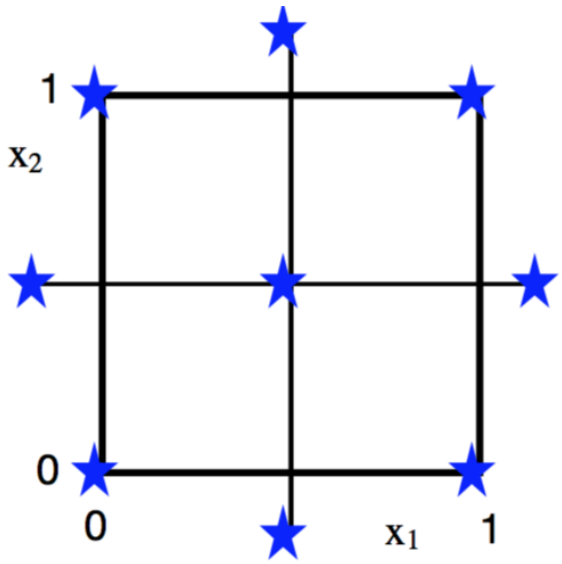



Figure 6.16: A Notional Central-Composite Design of Experiments for Two Variables [135]

However, classical DoE methods are most appropriate when applied to physical experiments, and do not perform as well in a computational context which lack the same "random error." Furthermore, the relative lack of samples toward the interior of the variable space makes the method less relevant given the expected non-linearities and interactions between Mission Design Variables.

*Space-Filling Designs*

Space-Filling Designs are Designs of Experiments intended specifically for model fit generation, but are appropriate for the fitting of any form of approximation. The goal

is to sample satisfactorily within the design variable space [40, 135]. These methods are intended to address so-called "bias error" rather than the random error of classical designs. Space-Filling Designs help develop approximations of complex behaviors and interactions, making them perfect for Surrogate Model generation or the use in a more computational context.

Various Monte Carlo random-sampling approaches aim to provide a simple method for generating a Space-Filling Design of Experiments. However, the methods do not provide a good guarantee of coverage and may leave some areas minimally sampled. "Binning" methods can be used to enforce better coverage for a Monte Carlo approach, but are still less structured than other Space-Filling methods [135].

The Latin Hypercube is a development of the "binned" Monte Carlo approach. It is now a popular DoE for computational applications. A Latin Hypercube already possesses the same general advantages of a Space-Filling Design, but now better scales with available computational budget [135]. The method effectively "bins" the design variable space and then places samples such that each bin along each axis has the same (or a similar) number of samples. This enables the scalability with computational capability, though there is a lower-bound of how many samples are needed for generally good coverage. A notional illustration of a Latin Hypercube sampling is seen in Figure 6.17.

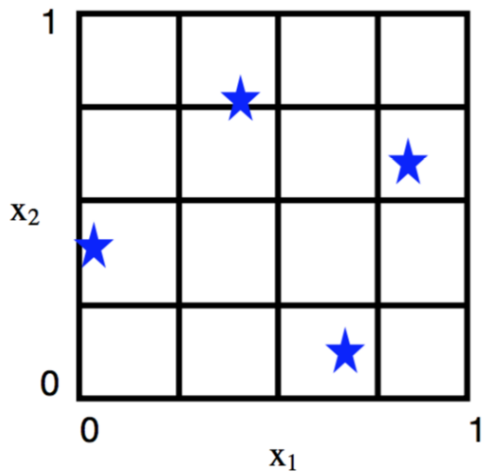



Figure 6.17: A Notional Latin Hypercube Design of Experiments for Two Variables and Four Total Samples [135]

*Complex and Adaptive Designs*

More complex Design of Experiments options exist and almost all are more active forms of design point selection. Sequential DoEs can be used to generate complementary sampling approaches and "fill-in" regions of interest [136]. Adaptive methods are popular to leverage a computer algorithm to intelligently select each design point to steadily complete the region, rather than selecting the points *a priori* as in a conventional DoE [137]. Lastly, newer methods are inspired by human decision-making and linking not only the identification of design points through DoEs but also including more adaptive identification of Measures of Effectiveness [138].

In the interest of scoping the options available for the **SAA**, these methods are not considered in detail. Future work is possible due to the complexity of any Mission Design Space and thus an improvement on the ideas presented in this document.

### 6.3.2 Selection of a Design of Experiments

Research Question 4.3 is to identify a satisfactory sampling method for a model utilizing Agent-Based Variables to explore across the Mission Design Space. The primary com-

plication arising from the use of ABVs is that there is not necessarily a direct and deterministic mapping between ABV setting and mission performance. This stochasticity requires the use of replications at a given mission design point in order to understand the solution space around that point.

Given that the end-goal of the **Stochastic Agent Approach** is to develop approximations of entity missions, a Space-Filling DoE seems most appropriate. While any Space-Filling Design should work, the Latin Hypercube is ultimately selected due to its good interior space coverage, scalability with number of cases, and popularity in a mission simulation context as exemplified by Turner [20].

Assertion 4.3

Space Filling Designs of Experiments are appropriate for an Agent-Based Variable-derived Mission Design Space and are a superior archetype for data-sampling to facilitate a model fit.

The use of a Space Filling DoE is ultimately enabled by how Agent-Based Variables convert discrete or categorical decision-making variables to be continuous. Those DoEs do not function well when operating on non-continuous parameters. The infusion of ABVs ultimately enables more conventional space exploration techniques, thereby simplifying and rendering more accessible the Mission Space Exploration and sampling ideal.

## 6.4 Conclusion

There are two steps to sampling across an ABV-involved Mission Design Space. The first is to identify the necessary number of replications to address stochasticity. The second is to select a satisfactory Design of Experiments.

The **SAA** develops an approach to find the necessary replicate number. The first step is to identify the Mission Design Point with the highest expected stochasticity and result

variability. The method to confirm that point is to run a targeted Design of Experiments with a more detailed sampling around the expected design point of highest variability (hypothesized to be when agents act most randomly). Using a large replicate number to define "truth" data, the cases were compared to identify ones with high variability. However, if more easily articulated points behave similarly variably, then they can be employed instead to simplify the process.

Once that point is found, a large number of replications are run to generate a "truth" dataset at that design point. The Bootstrapping method is used to re-sample the "truth data" at various sample sizes, multiple times. Distributions on the metrics for each sub-sampling can then be compared against the truth via visual inspection and the calculation of the Coefficient of Variance. When the CoV (or the sample means) have converged sufficiently close to the "truth" data, then the necessary replicate number has been found. For the Contested Reconnaissance Model, that replicate number is 200, not the 50 indicated by literature or the industry's "rules-of-thumb."

The **Stochastic Agent Approach** aims to develop higher efficiency approximations of mission-level effects so a Space-Filling Design of Experiments is most appropriate. While the exact DoE is up to the end-user, a Latin Hypercube provides good sample coverage and effective computational efficiency.

# CHAPTER 7

# DEVELOPING MODEL FITS AND MISSION-LEVEL APPROXIMATIONS

> Because the form of the true response function $f$ is unknown, we must approximate it.
>
> - Raymond Myers, Douglas Montgomery, and Christine Anderson-Cook [40]

## 7.1 Output Metrics of Interest

The natural solution after developing a new variable form or simulation is to identify any conclusions that can be made or identified from the method. If the model is highly efficient already, then the utility from approximations is lower. However, as evidenced by the continuing popularity of empirical relationships like the Lanchester Laws, there is a clear demand for such fits of higher-fidelity models. Furthermore, former Deputy Secretary of Defense Robert Work specifically sought for improved war games to identify underlying trends and behaviors which can lead to success or failure [1]. Thus, the end-goal is to seek the relationships between different mission-affecting elements and the outputs of interest. This can be accomplished either by direct query of an obtained dataset or by fitting approximations on that data set for infusion into other simulations or analyses.

### 7.1.1 Connecting Design Variables and Output Metrics

In conventional solution exploration, there is usually a more direct mapping between the design variables and the metrics of interest [39, 40]. For example, it is commonly understood that a higher Aspect Ratio wing will improve aerodynamic efficiency of a

notional aircraft. While this relationship is usually not perfectly captured at a lower-fidelity level, the trend remains valid. Furthermore, the relationship can be decomposed and identified through direct physics relationships.

In the case of standard Mission Space Exploration, there is some degree of that expected relationship. As seen in reports by Kewley, Revello, and others in situations of Mission Space Optimization, asset types, positions, and actions have a direct and understood effect on the metrics of mission success [56, 58]. In short, the variables in the Mission Action Space have an obvious and more direct effect on the conduct of a mission.

However, in an Agent-Based Variable formulation, the mapping between design variables and output metrics is more nebulous. This problem is further exacerbated by the stochastic nature of an ABV formulation. As a result, the conventional approach of linking design variables to outputs via physics or mission planning means is less helpful. While an ABV-based model does have all the intermediary data to make the conventional connections, it would be more illuminating to connect outputs and their trends to the Agent-Based Variables themselves. This yields the basic form of Research Question 5.

Research Question 5

What is an appropriate methodology for fitting a approximation equation to an ABV-based Mission Variable Space?

This objective is complicated by the need to identify and utilize Measures of Effectiveness. A good example of such interests would be evaluating what metrics are relevant through consulting SMEs. In this case, Likert-style questions and analysis are of interest through the use of quantified variations on the “agree” to “disagree” spectrum. Exploratory Factor Analysis (EFA) could be employed to try to ascertain the true drivers of interest in the various identified metrics of interest [139]

### 7.1.2 Design and Intermediary Variables

An additional complication is that there are two "tiers" of variables worth tracking for the purposes of model fit creation: the "design variables" and the "intermediary variables." These two archetypes have been mentioned previously in the document, though the complicating issue is the intersection between the two that arises when comparing an Agent-Based Variable problem formulation to a direct mission creation approach.

Design variables are the variables modified to obtain different experiments or cases. They are under the control the user or the automated routine which generates simulation cases [140]. Using an example scenario of one Attacker encountering one Defender:

- Design Variable = The Attacker will evade (or not evade) the Defender
- Output Metric = Attacker's position over time

However, the primary complication from the use of Agent-Based Variables is the decoupling between the design variables and the Mission Action Variables (now intermediary variables). Returning to an ABV breakdown of the same problem:

- Design Variable = Agent-Based Variable describing the probability of an Attacker evading the Defender
- Intermediary Variable = The Attacker will evade (or not evade) the Defender
- Output Metric = Attacker's position over time

As can be seen above, the former design variable is now an intermediary. While relationships can be made between the intermediary variables and the output metrics, it is less helpful and often more complex (often due to variable space intractability) than the overall relationship between the design variables and the same metrics.

Fortunately, there is a Modeling and Simulation precedent for addressing such intermediary variables. In a Design Space Exploration, available settings include not only

conventional Engineering Design Variables (wingspan, fuselage diameter, material selection) but also technology infusion (new engine type, new manufacturing method, new materials). When leveraging empirical models to predict system performance, so-called "K-factors" can be employed to reflect technology infusion and effects on intermediary properties of the vehicle [19, 41]. For example, there is an intermediary value for the weight of an aircraft's engine. However, a new technology is to be incorporated which decreases engine weight by 5% relative to historical trends. Thus a K-factor of 0.95 is added to adjust the intermediary engine weight value for all future calculations of that design.

When developing a model fit for this type of problem, the equations are frequently formulated as functions of the continuous K-factors rather than the discrete "on-off" of numerous considered technologies. Thus the resulting equation is a function of both design variables (wingspan, fuselage diameter) and intermediary values (the K-factors reflecting technology infusion). In a similar manner, approximations for mission simulation could consist of not only design variables (ABVs, number of units) but also intermediary values (what regions did the agent pass through, number of engagements rather than the ABV probabilities).

For the sake of solution simplicity, all generated model fits will rely on design variables not intermediary variables. However, their inclusion should be feasible. Furthermore, some illustrative plots may leverage intermediary conditions and values to observe trends and commonalities, though not for model fitting itself.

### 7.2 Surrogate Models

Surrogate models aim to replace a complex simulation with faster approximation equations. Due to the computational expense in setting up and running an Entity-Based Combat Model, such an approximation would be of interest for future exploration and query of the solution space. An original dataset is always required for model gener-

ation. However, running fewer cases initially through the Mission Space Exploration could then be supplemented by approximate cases generated by querying the Surrogate Model, provided the fit is sufficient [19, 41]. Model development proceeds from bottom to top in Figure 7.1 with real world data calibrating computer models which are then approximated in the form of a Surrogate Model. Note that "Metamodel" is another term for "Surrogate Model."

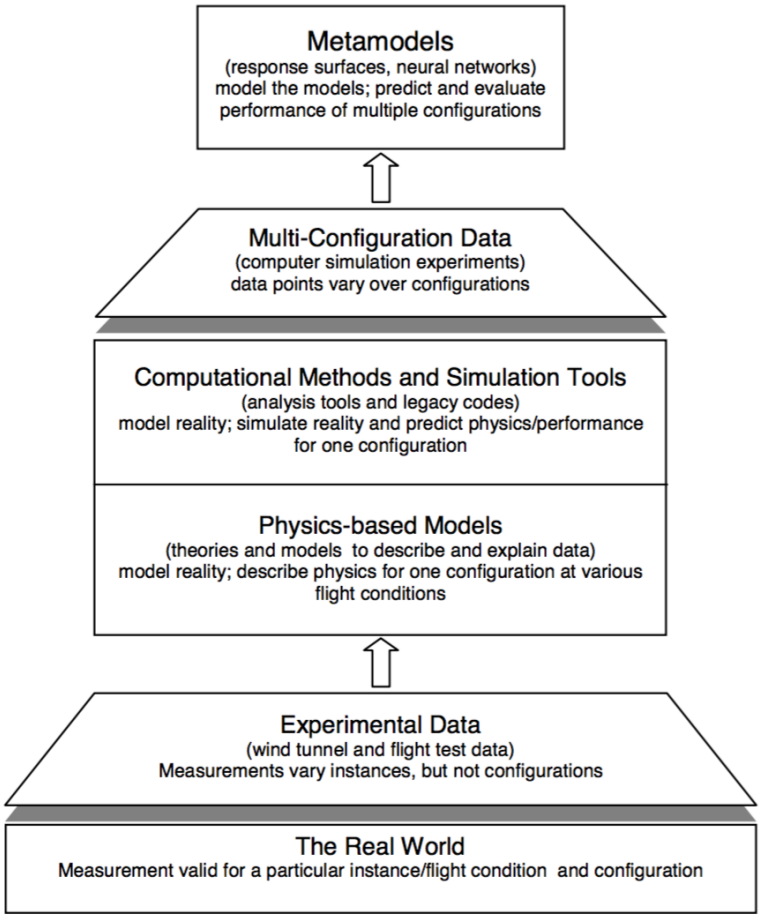



Figure 7.1: Process for Generating a Surrogate Model [41]

In short, the **Stochastic Agent Approach** addresses the intermediate layers of the Surrogate Model generation process. Previous approximation approaches for mission simulation directly link the real world to models such as the Lanchester Equations. In-

stead of that direct link, this research introduces the intermediary simulation layers which either supplement or replace direct real world experimental data. These computer-generated datasets are then used to craft the Surrogate Models.

### 7.2.1 Utility

Surrogate Models are useful for running lower-fidelity but computationally efficient cases for analysis. Such models are intended to approximate the detailed M&S of a high-fidelity model in favor of a faster data-driven method. These higher-fidelity, lower-level models are often time-intensive to run, as is the case with mission simulation [40, 41].

Surrogate Models are popular in higher-level simulations or Operations Research work because of the time savings. When modeling an entire military campaign, or multiple engagements over the life of a single asset, the focus is on the integration and life cycle effects, not specific maneuvers or activities during missions [63]. Thus a Surrogate Model would be of high utility in these larger integrated frameworks.

Many times abstracted or empirical models can be considered akin to Surrogate Models, but relying on historical data rather than computer-generated results. The vaunted Lanchester Equations abstract engagements to be differential equations based on historical data or subject matter expertise [13, 14]. However, the Lanchester Equations are multiple linked differential equations, rather than the simpler-to-use polynomials or other "fast computation" processes used in Surrogate Models. Furthermore, the datasets obtained using the **SAA** do not readily lend themselves for conversion into paired differential equations. Thus, the Lanchester Equations, while a major element in campaign-level engagement simulation, are not considered for the form of Surrogate Model.

### 7.2.2 Weaknesses

The primary complications from Surrogate Models are risks from failing to appropriately fit the original dataset. An improper approximation can be either from over-fitting or

under-fitting, as illustrated by Figure 7.2 and Figure 7.3. In both images, the data points to fit are small crosses while the approximation function is the solid line.

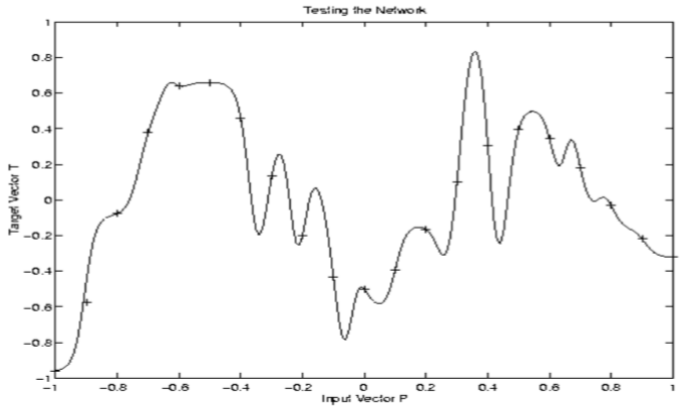



Figure 7.2: Example of an Over-Fit Surrogate Model [41]

A maximally over-fit model is one where the approximation equation simply connects all data points. Despite being a “bad” outcome, an over-fit model actually exhibits zero (0) or close to zero error (see Figure 7.2). However, over-fit models generally exhibit similar behavior of being overly contoured and “precise” to the provided data. This model form loses utility compared to a properly fit Surrogate Model because it is hard to generalize. The overall function trend and relationship is obscured by the attempt to perfectly fit every provided point [40, 41].

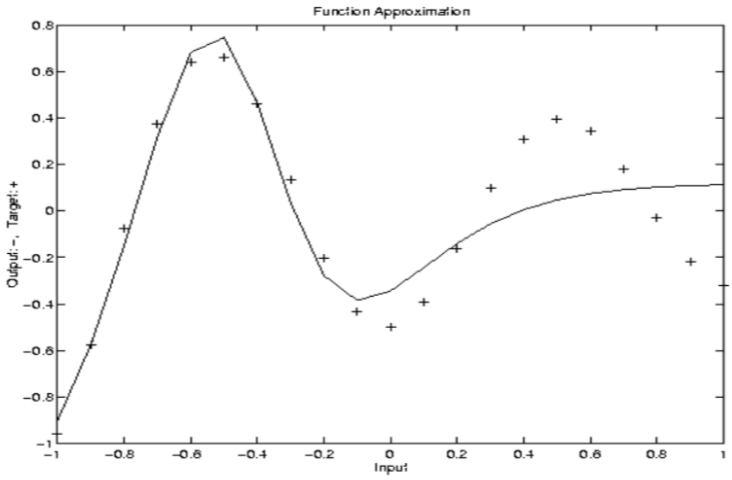



Figure 7.3: Example of an Under-Fit Surrogate Model [41]

An under-fit model fails to capture the major behavior trends which are desired from a useful Surrogate Model (see Figure 7.3). Error is increased relative to a good fit and such actions usually arise from a lack of elements to complete the fit equation. While such an approximation is indeed an estimation of output metrics, it fails to capture the full relationships and therefore loses utility in a tradeoff or sampling context [40, 41].

### 7.2.3 Archetypes

Multiple forms of Surrogate Models exist and each have their own strengths and weaknesses. Some are easier to define and utilize, while others are more complex but result in superior fits. Improved computational resources have introduced more complex methods leveraging the concept of Machine Learning [19, 41, 122]. Regardless, there are two archetypes considered for the initial development of the **SAA**: the Ordinary Least Squares Regression and the Artificial Neural Network (ANN).

*Ordinary Least Squares Regression*

The Ordinary Least Squares Regression is among the most simple Surrogate Models. It aims to develop a polynomial hyper-surface approximation of the result via minimizing the sum of the squares of the errors between the measured and estimated data [40]. For reference, the standard nomenclature is for $n$ samples being fit to $k$ coefficients in the polynomial. The objective of this model fitting method is to identify the $k$ coefficients which result in the desired (minimum) sum of the squares of the error.

The method works so long as there are more samples than coefficients ($n > k$). However, there remains the risk of over-fitting (when $n \simeq k$) and under-fitting (when $n << k$) and so appropriate considerations need to be made [19, 40]. Furthermore, issues arise due to how these regressions are functionally Taylor-series approximations and therefore break down in large variable spaces or for insufficiently differentiable functions [19].

*Artificial Neural Networks*

ANN are a method whereby a computer algorithm assembles logic to link inputs to expected outputs. This approach has already been employed in both Surrogate Modeling and in creating AI for "learning" how to accomplish an objective such as in a video game or image recognition [19, 44, 122].

The method is informed by the basic understanding of how the human nervous systems operates. Simulated artificial "neurons" are connected and generate simple relationships between themselves. These connections are weighted and as the algorithm "learns" to match estimated to true outputs, it adjusts those weights to improve its accuracy. However, the method is complicated by the existence of the "hidden" layer of artificial neurons, which can impede traceability during the model fitting process. However, this method is not as constrained as the aforementioned regression, provided sufficient numbers of neurons and data points are used [19].

The statistical software JMP possesses a built in Neural Network analysis system

which will be used to generate the fit model. This approach utilizes a supervised learning approach as illustrated by Figure 7.4).

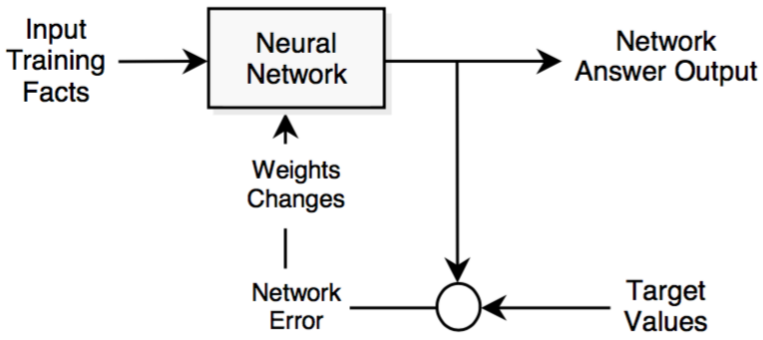



Figure 7.4: Illustration of Supervised Learning for a Neural Network [41]

Neural Network supervised learning occurs when target values are known to the algorithm. Thus the model can constantly update the neuron connections to better fit the estimated and true results. As the outputs from each simulation case are known from the high-fidelity Agent-Based Model, this approach is the most reasonable for a Surrogate Model fitting context. In comparison, unsupervised learning is used to let an algorithm "learn" and adapt rather than fit a known dataset, and is more popular for generalized AI development [41].

Two sets of data are used to evaluate the fit quality of a Surrogate Model ANN. The training data is used to generate the actual fit in the supervised learning context. This is then supplemented by the validation data which is used to check if the fit is more extensible beyond the specific training data points. In more conventional regression terminology, the training data yields the Model Fit Error and the validation data yields the Model Representation Error.

In the JMP Neural Network fit system, relatively few settings are available to the user. The major user choice is the number of artificial "neurons" of each of three (3) function

types: $TanH$, Linear, and Gaussian. Otherwise, all other properties are defaulted by the software.

### 7.2.4 Selection of a Surrogate Model Method

All Surrogate Models are ultimately derived from datasets generated by computational experiments. The exact method for creating a model can vary ranging from focusing on basic trends, to polynomial approximations, to higher-order terms and estimation methods.

Turner investigated the use of Surrogate Models in mission simulation, but with a particular focus on asset design and overall Mission Architecture, rather than affecting the decision-making logic of the agents [20]. In his work, he identified that a simple Ordinary Least Squares regression was appropriate for a stochastic mission simulation, assuming an appropriate number of replications had been run.

However, Turner's work did not consider changes to the decision-making elements of a stochastic Agent-Based Model. With the Mission Design Variables now directly impacting stochasticity (as illustrated for Research Question 4), Ordinary Least Squares Regression is unlikely to provide a satisfactory fit. Consequently, the adaptability of an Artificial Neural Network informs Hypothesis 5.

Hypothesis 5

An Artificial Neural Network will provide satisfactory approximations and surrogates of the output metrics of interest for an ABV-involved Mission Simulation and Design Space.

## 7.3 Experiment for Research Question 5

In these cases, the mission simulation was conducted and post-processed in MATLAB. JMP was used to fit both forms of Surrogate Model.

### 7.3.1 Model Overview

The model utilized is the Contested Reconnaissance Model. In this model, the Attackers are seeking to survey a region being guarded by Defenders. The Attackers' objective is to gain understanding a specific amount of the region above a given threshold (e.g. 80% of the region surveyed to at least 50% quality).

As before, an interaction-focused model is appropriate. To simplify the case study and allow easier comparison, an Agent-Based Variable scheme will only be employed on one of the two sides in the simulation. To reduce dimensionality, the case study is two-dimensional with passive opponents who operate more like "impediments" than maneuvering enemies.

The design variables identified for this model are detailed in Table 7.1. For a more in-depth explanation of the Contested Reconnaissance Model, please see Chapter 6.

Table 7.1: Design Variable Descriptions for the Contested Reconnaissance Example

| Variable | Type | Description |
|---|---|---|
| Number of Defenders | Architecture - Defenders | |
| Number of Attackers | Architecture - Attackers | |
| Ammo | Architecture - Attackers | Number of Defenders a single Attacker can engage per mission |
| Attacker Xmid | Architecture - Attackers | Center point of the Attackers' formation |
| Attacker Xrange | Architecture - Attackers | Width of the Attackers' formation |
| Aggression | ABV - Relational | Preference for not evading a Defender |
| Engagement | ABV - Relational | Given that a Defender will not be evaded, preference for engagement |
| Evasion | ABV - Navigational | Preference for flying over an evaded Defender |

### 7.3.2 Generating Cases

Each case for the data collection is generated using a Latin Hypercube Design of Experiments with 5000 unique points. Each case will be replicated 200 times. The various design variables and their ranges are described in Table 7.2.

Table 7.2: Design Variables for the Contested Reconnaissance Example

| Variable | Type | Minimum Value | Maximum Value |
|---|---|---|---|
| Number of Defenders | Discrete | 0 | 10 |
| Number of Attackers | Discrete | 1 | 4 |
| Ammo | Discrete | 0 | 4 |
| Attacker Xmid | Continuous | 100 | 900 |
| Attacker Xrange | Continuous | 10 | 100 |
| Aggression | Continuous | 0.0 | 1.0 |
| Engagement | Continuous | 0.0 | 1.0 |
| Evasion | Continuous | 0.0 | 1.0 |

### 7.3.3 Selected Metrics of Interest

The metrics of interest for this case study are identified in Table 7.3. Similarly, these MOEs were discussed in detail in Chapter 6.

Table 7.3: Design Variables for the Contested Reconnaissance Example

| Variable | Desired Value |
|---|---|
| Attacker Survival Rate | Maximize |
| Defenders Survival Rate | Minimize or Ignore |
| Reconnaissance Coverage | Maximize |

These MOEs are identical to those discussed previously and mapped and fit in Chapter 6 for Research Question 4. It is noteworthy that Table 7.3 is not an exhaustive list of MOEs. However, these metrics were selected to simplify the problem and to mirror the outcomes from the Lanchester Equations, which focus almost exclusively on the popu-

lations of each side.

For each design point in the Latin Hypercube, twelve (12) values were found. These are the mean, variance, skewness, and kurtosis for each of the three (3) MOEs, the surveillance coverage, the Attacker survival rate, and the Defender survival rate.

Visual inspection, while effective, is not an easily repeatable or standardized method for distribution type identification. However, the selected Measures of Effectiveness are all fixed in a range between 0.00 and 1.00, thereby making the Beta distribution the most reasonable for the Contested Reconnaissance Model, at least in theory. However, there were problems when trying to fit the distribution when the parameters were more discrete, such as the Attacker survival rate. That value in particular appears to cause problems because there are no more than four (4) attackers, so the possible values are only (0.00, 0.25, 0.33, 0.50, 0.67, 0.75, and 1.00). As such, the model fits were applied to the mean ($\mu$), variance ($\sigma^2$), skewness, and kurtosis of every output metric distribution with the hope of fitting a normal distribution. While there is no guarantee of a normal distribution in truth, the general method should remain relevant regardless of distribution fit utilized.

### 7.3.4 Developing Surrogate Models

With the full litany of cases developed, it is now appropriate to generate the resulting Surrogate Models. These Surrogate Models were fit using the JMP statistical software suite.

*Ordinary Least Squares Regression*

The Ordinary Least Squares Regression was fit to a third-order polynomial and up to third-order interactions. The Surrogate Models were developed and the resulting fits are seen in Table 7.4 for the Least Squares Regression. The objective is for an $R^2$ value to be greater than 0.999, thereby indicating a good overall fit.

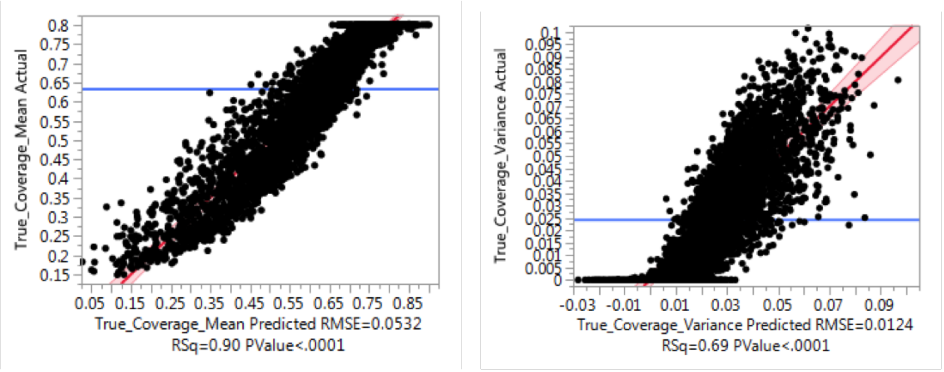
Table 7.4: The Fit Quality for the Ordinary Least Squares Regression

| Property | Coverage | Attacker Survival | Defender Survival |
|---|---|---|---|
| $R^2$ for the Mean | 0.90 | 0.93 | 0.93 |
| $R^2$ for the Variance | 0.69 | 0.77 | 0.79 |
| $R^2$ for the Skewness | 0.39 | 0.85 | 0.85 |
| $R^2$ for the Kurtosis | 0.24 | 0.49 | 49 |

As is clear from Table 7.4, the Least Squares Regression fits for most outputs are not acceptable. The $R^2$ values are high for some fits, but none above the desired 0.999 threshold for a good model fit. The lack of acceptable fit is most likely due to the clearly defined variable ranges (0.00 to 1.00) of the various MOEs, making a polynomial fit more difficult.

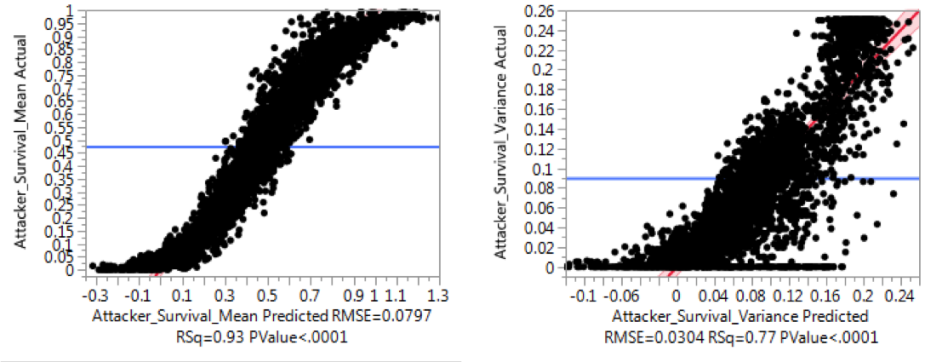
Furthermore, it is evident that fitting the higher-order distribution parameters (skew and kurtosis) is more difficult than the mean and variance. Consequently, only the fits for the mean and variance will be considered for evaluating not only the Least Squares Regression fits, but also the Artificial Neural Network.

The fit relationships for the Ordinary Least Squares Regression are illustrated in Figure 7.5, Figure 7.6, and Figure 7.7 for the estimated means of the coverage, Attacker survival rate, and Defender survival rate, respectively. The Model Fit Error is illustrated by the differences between the true and predicted values, and plotted against the predicted values. Meanwhile, the Actual by Predicted plot would display a near-perfect line should the fit be successful.



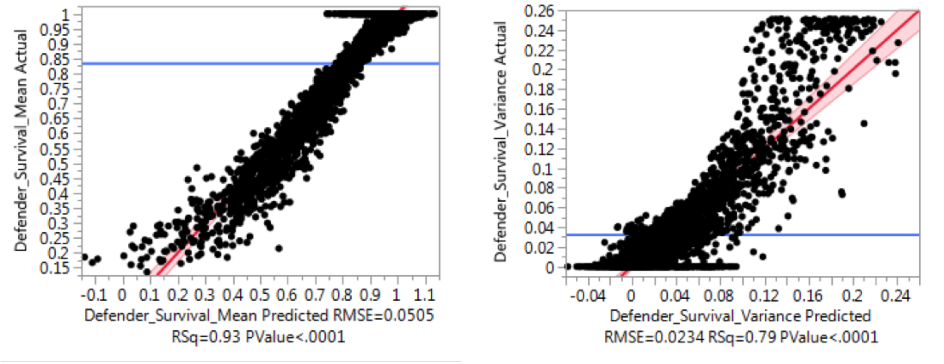

(a) Actual by Predicted for Mean (b) Actual by Predicted for Variance

Figure 7.5: The Ordinary Least Squares Regression Fit Errors for the % Coverage




(a) Actual by Predicted for Mean (b) Actual by Predicted for Variance

Figure 7.6: The Ordinary Least Squares Regression Fit Errors for the Attacker Survival Rate




(a) Actual by Predicted for Mean (b) Actual by Predicted for Variance

Figure 7.7: The Ordinary Least Squares Regression Fit Errors for the Defender Survival Rate

Clear from the Ordinary Least Squares Regression images, the bounds on the MOE ranges present fitting difficulties. Furthermore, the fits are not good overall, further confirming that Ordinary Least Squares Regression is not a feasible Surrogate Modeling technique for an ABV-based Mission Design Space formulation.

*Artificial Neural Network*

The ANN was fit using the JMP Neural Network analysis system. The provided Graphical User Interface (GUI) enabled the subdivision of the 5000 unique data points into training and test sets. The default settings were used for a 67% training and 33% testing breakdown.

As the Neural Net is a "smarter" fitting algorithm, it should be able to account for the clear variable bounds. However, setting up the Neural Network is more complex than a standard fit method. Multiple combinations of "neuron" types were considered. Thus several settings were tried to maximize the $R^2$ value of the fits. In Table 7.5, each column is labeled by the number of $TanH$, Linear, and Gaussian nodes used, respectively. Only a single-layer ANN was considered at first, with the fit metrics for the means and variances provided in Table 7.5.

Table 7.5: The $R^2$ Values for the Tests of Various Artificial Neural Network Fits as a Function of Number of First-Layer Neurons

| Property | (10, 10, 10) | (20, 20, 20) | (30, 30, 30) |
|---|---|---|---|
| Mean of Coverage | 0.91 | 0.92 | 0.93 |
| Mean of Attacker Survival Rate | 0.95 | 0.96 | 0.96 |
| Mean of Defender Survival Rate | 0.96 | 0.98 | 0.98 |
| Variance of Coverage | 0.75 | 0.77 | 0.81 |
| Variance of Attacker Survival Rate | 0.74 | 0.83 | 0.84 |
| Variance of Defender Survival Rate | 0.81 | 0.90 | 0.95 |

Notably, the fits for the skew and kurtosis were created for the first ANN fit. However, the fits were poor and unhelpful (all with $R^2$ below 0.5), and so were not listed in Table 7.5.

While the Model Representation Error $R^2$ values for the mean were trending towards acceptable ($> 0.90$) for the single-layer ANN, the fits for the variance were not as good. Furthermore, the fits for the validation data were fairly good which indicated that the fits were not over-fitted or under-fitted. The best-fit (30, 30, 30) single-layer ANN had final $R^2$ values articulated in Table 7.6.

Table 7.6: The $R^2$ Values for the Selected (30, 30, 30) Single-Layer Neural Network Fit

| Property | Training $R^2$ | Validation $R^2$ |
|---|---|---|
| Mean of Coverage | 0.93 | 0.93 |
| Mean of Attacker Survival Rate | 0.96 | 0.96 |
| Mean of Defender Survival Rate | 0.98 | 0.97 |
| Variance of Coverage | 0.81 | 0.79 |
| Variance of Attacker Survival Rate | 0.84 | 0.87 |
| Variance of Defender Survival Rate | 0.95 | 0.89 |

A second layer was added to investigate if superior fits could be achieved. The best neuron combination (30, 30, 30) was maintained for the first layer while the second layer was added and the fit attempted again as described in Table 7.7. Unlike the initial fitting attempts, which included the skew and kurtosis of the distributions, the second fits focused on only the mean and variance. As the fits were all bad for the skew and kurtosis with both the single layer Neural Network and Least Squares Regression, those fits were removed completely from the second fit attempt.

Table 7.7: The $R^2$ Values for the Two Layer Neural Network Fits as a Function of Number of Second-Layer Neurons

| Property | (5, 5, 5) | (10, 10, 10) | (15, 15, 15) |
|---|---|---|---|
| Mean of Coverage | 0.94 | 0.94 | 0.95 |
| Mean of Attacker Survival Rate | 0.98 | 0.97 | 0.98 |
| Mean of Defender Survival Rate | 0.99 | 0.99 | 0.99 |
| Variance of Coverage | 0.85 | 0.84 | 0.88 |
| Variance of Attacker Survival Rate | 0.95 | 0.92 | 0.93 |
| Variance of Defender Survival Rate | 0.99 | 0.99 | 0.99 |

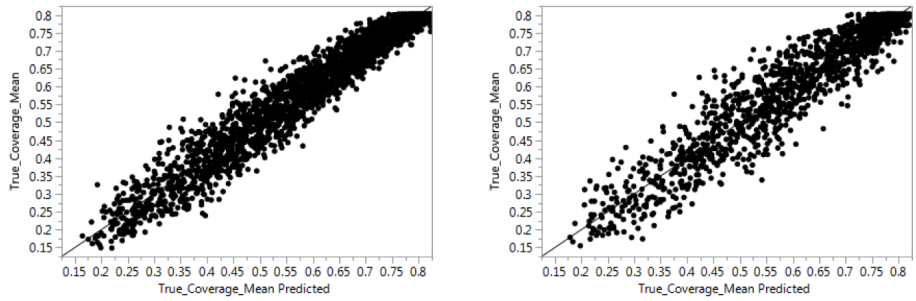
The selected ANN neuron numbers is for 30 neurons of each type in the first layer, and 10 neurons of each type in the second layer. Those fits also had superior validation $R^2$ values as articulated in Table 7.8. While the fits for the mean were comparable, the fits for the variance in the two-layer ANN improved significantly, especially for the two agent survival rates.

Table 7.8: The $R^2$ Values for the Final (30, 30, 30) and (10, 10, 10) Two-Layer Neural Network Fit

| Property | Training $R^2$ | Validation $R^2$ |
|---|---|---|
| Mean of % Coverage | 0.94 | 0.92 |
| Mean of Attacker Survival Rate | 0.97 | 0.96 |
| Mean of Defender Survival Rate | 0.99 | 0.99 |
| Variance of % Coverage | 0.84 | 0.79 |
| Variance of Attacker Survival Rate | 0.93 | 0.90 |
| Variance of Defender Survival Rate | 0.99 | 0.97 |

The fit relationships for the final ANN are illustrated in Figures 7.8 through 7.13 for

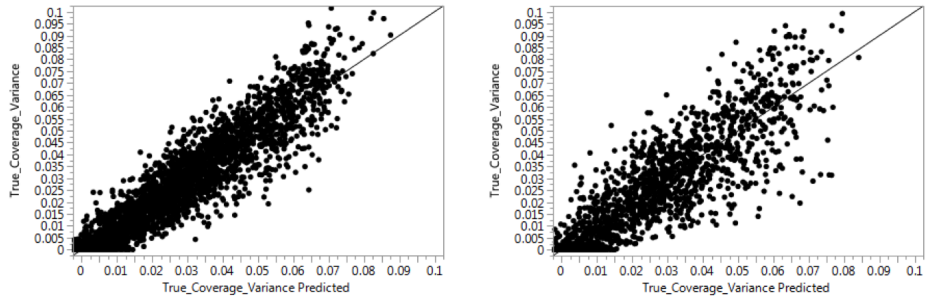
the three MOEs and comparing the training and validation data fits.




(a) Actual by Predicted for the Training Data

(b) Actual by Predicted for the Validation Data

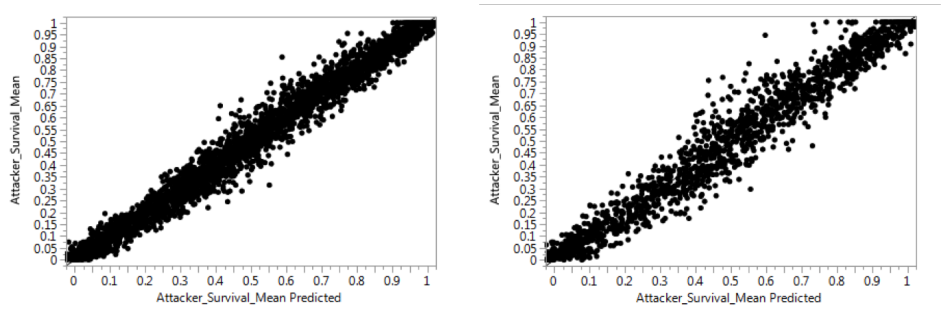
Figure 7.8: The Artificial Neural Network Fit Errors for the % Coverage Mean




(a) Actual by Predicted for the Training Data

(b) Actual by Predicted for the Validation Data

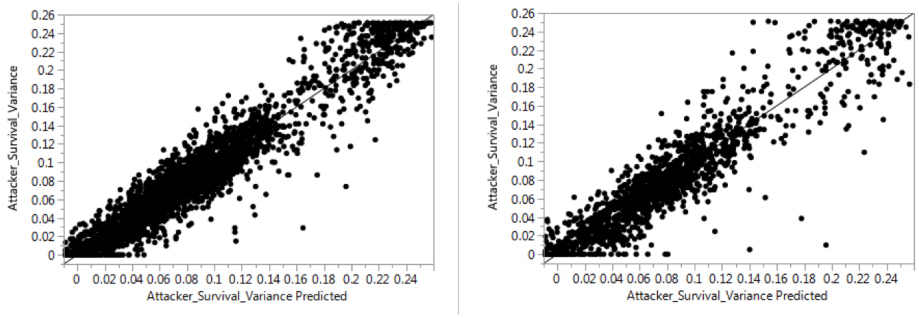
Figure 7.9: The Artificial Neural Network Fit Errors for the % Coverage Variance




(a) Actual by Predicted for the Training Data

(b) Actual by Predicted for the Validation Data

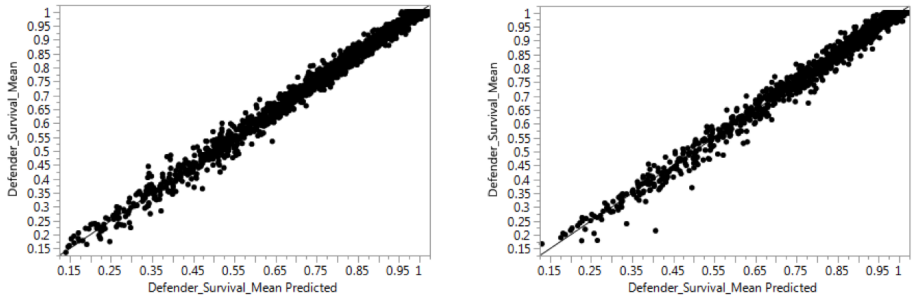
Figure 7.10: The Artificial Neural Network Fit Errors for the Attacker Survival Rate Mean




(a) Actual by Predicted for the Training Data

(b) Actual by Predicted for the Validation Data

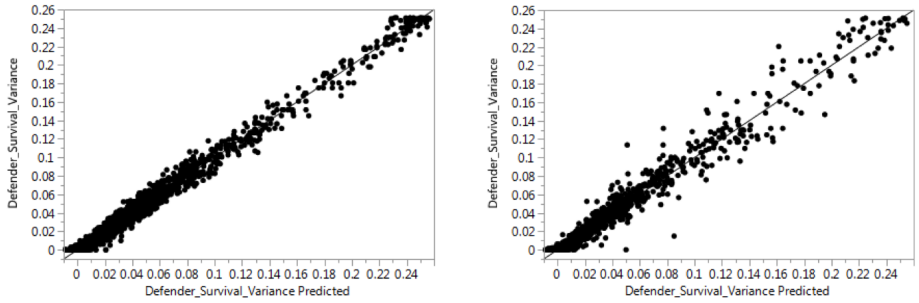
Figure 7.11: The Artificial Neural Network Fit Errors for the Attacker Survival Rate Variance




(a) Actual by Predicted for the Training Data

(b) Actual by Predicted for the Validation Data

Figure 7.12: The Artificial Neural Network Fit Errors for the Defender Survival Rate Mean




(a) Actual by Predicted for the Training Data

(b) Actual by Predicted for the Validation Data

Figure 7.13: The Artificial Neural Network Fit Errors for the Defender Survival Rate Variance

Unfortunately, the overall model fit are still not ideal ($> 0.999$). As illustrated by the

Actual by Predicted plots of Figures 7.8 through, 7.13, there remain outliers in comparing the actual values (y-axis) to the ANN predicted values (x-axis). Furthermore, while the fits were pretty good, the validation cases were not as good as desired. This perhaps implies a degree of overfitting but does not change that the ANN works for the general behavior but not for a precise and clear Surrogate Model.

When transitioning to the ANN, the fits overall appear superior to the Ordinary Least Squares Regression. However, there is an added complication of organizing the Neural Network layers, neuron types, and other considerations. This fitting process was somewhat trial-and-error, but the resulting fits are vastly superior than the Ordinary Least Squares Regression.

*Comparison and Discussion*

After considering the standard fit metrics, it is clear that the Neural Network is the superior Surrogate Modeling approach (of the two) for the Contested Reconnaissance Model. However, this experiment was only for this single case study model and did not broadly consider other Surrogate Model archetypes. While the neural network works effectively in this context, there is not reason to automatically ignore other potential methods.

Research Conclusion 5

An Artificial Neural Network is a more effective Surrogate Modeling technique for fitting mission-based Measures of Effectiveness in an Agent-Based Variable context. However, the fits are not as precise as desired and future work should investigate new methods and approaches.

A major hypothesized problem is the discretized nature of the number of Attackers and Defenders in the simulation. The selected MOEs are all related to survival rates (the inverse of attrition) from a single mission and as such are affected by both the number of entities involved (some of the Mission Design Variables) and the limits of the range from

0.00 to 1.00. That range limit resulted in the strange extremes relationships observed and a major contributor to why the Ordinary Least Squares Regression did not work satisfactorily. However, the Neural Network did not fit particularly well either. The fits for the Defender survival distribution properties were actually quite good, however the fits for the Attacker survival distribution properties were poor.

It is possible that there is a need for a better discrete prediction method when the number of involved entities is small. This is a similar problem faced by the Lanchester Laws. However, there is an ability to craft a more "clever" Surrogate Model to approximate the behavior observed in the Agent-Based Model utilized by the **Stochastic Agent Approach**.

## 7.4 Observations from the Datasets and Model Fits

Even though the model fits for both approaches were not ideal, they still can provide some insights into major Mission Design Variable effects on the various output metrics. Most noteworthy among these are Pareto Plots which help illustrate which parameters had the greatest impact on solution MOEs. Additionally, Parallel Plots can be used to observe commonalities between Mission Design Points of similar performance.

### 7.4.1 Parallel Plots

For initial data evaluation and analysis, the fit models were ignored and only the raw data considered. A useful tool for this analysis is a Parallel Plot, which can illustrate trends and commonalities between cases. A data filter can be applied to limit which cases are considered.

Figure 7.14 illustrates a single case on a Parallel Plot. The dark line is that case and the values are graphically notional with larger values to the right and smaller to the left. The grayed-out lines are the other cases.

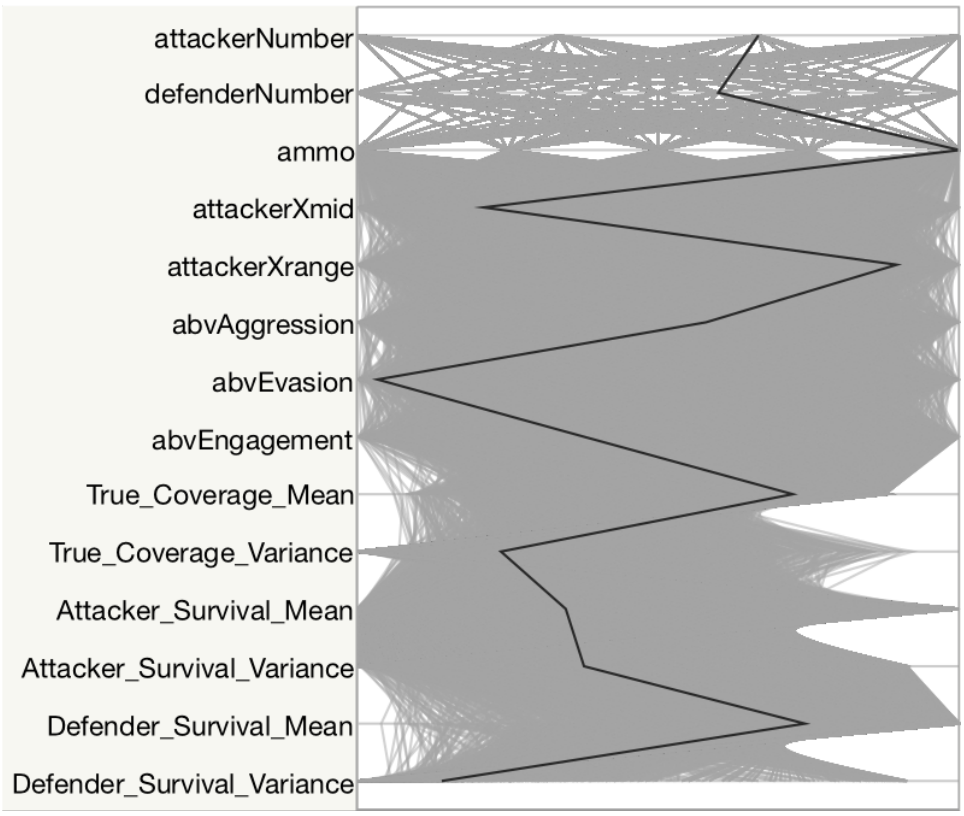



Figure 7.14: Example Parallel Plot with a Single Case Highlighted

The first set of applied filters limited the cases to ones with high surveillance coverage (> 75%) and high Attacker survival rate (> 90%). This parallel plot is seen in Figure 7.15.

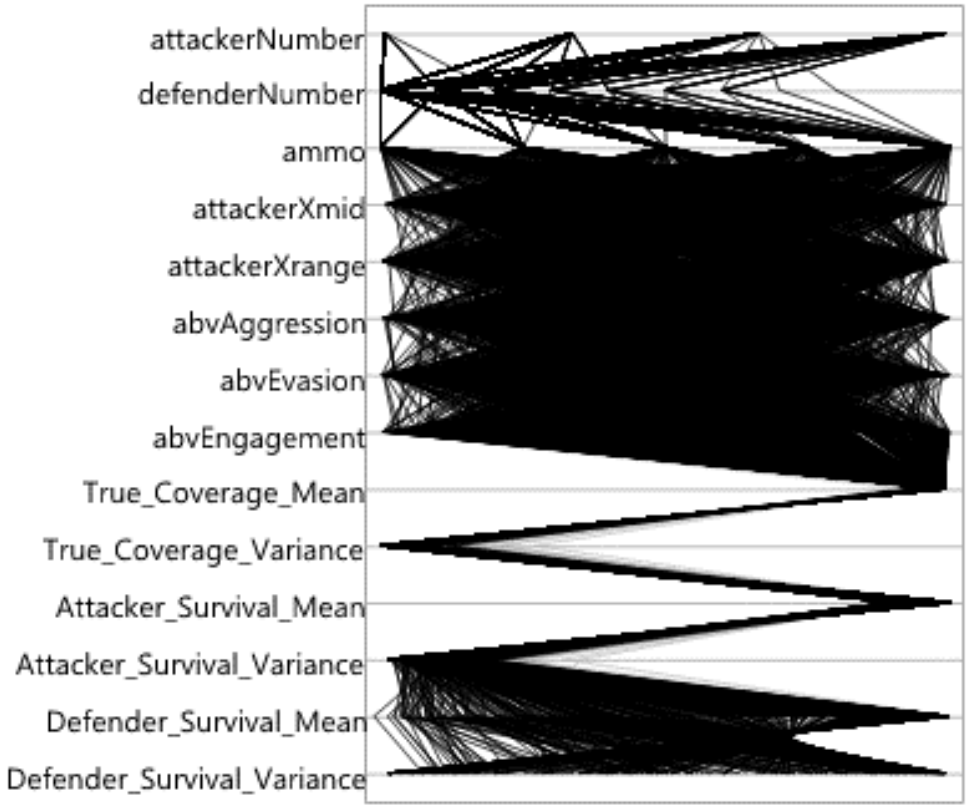



Figure 7.15: Parallel Plot for High Coverage (> 75%) and Attacker Survival (> 90%)

These cases were quite varied but all exhibited commonalities in the form of relatively few Defenders and small variances in the two filtered MOEs. It inspired a second filtering to only consider cases with only one (1) Attacker as illustrated by Figure 7.16.

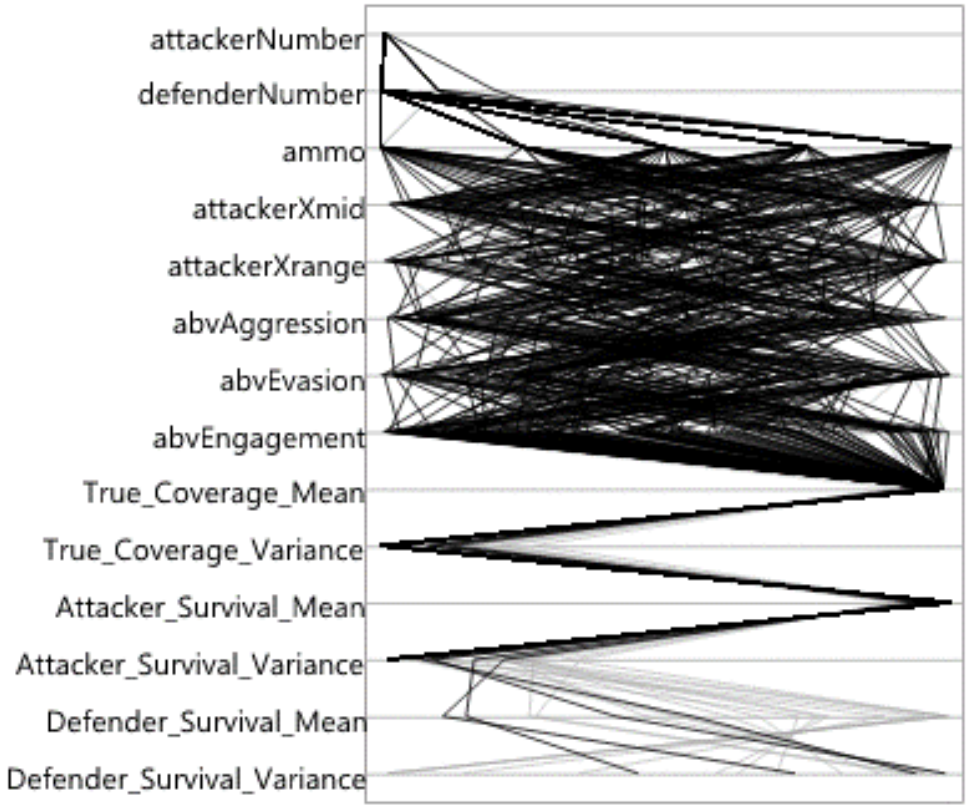



Figure 7.16: Parallel Plot for High Coverage (> 75%) and Attacker Survival (> 90%), but Only One Attacker

For this plot, the most successful missions were ones with very few Defenders (most cases have zero). This implies that a single Attacker can only accomplish the demanded surveillance if there are few impediments to its search. Furthermore, the wide spread of ABV values is expected as the agent decision-making is only triggered by the existence of Defenders. Fewer Defenders leads to fewer uses of the agent Decision Tree and thus fewer chances for the various ABVs to have any influence on the outcome.

At the other end of the spectrum, there can be an investigation of what Mission Design Variables produce generally poor MOEs. The resulting Parallel Plot for a low (< 50%) Attacker survival rate but with high surveillance coverage (> 75%) is seen in Figure 7.17.

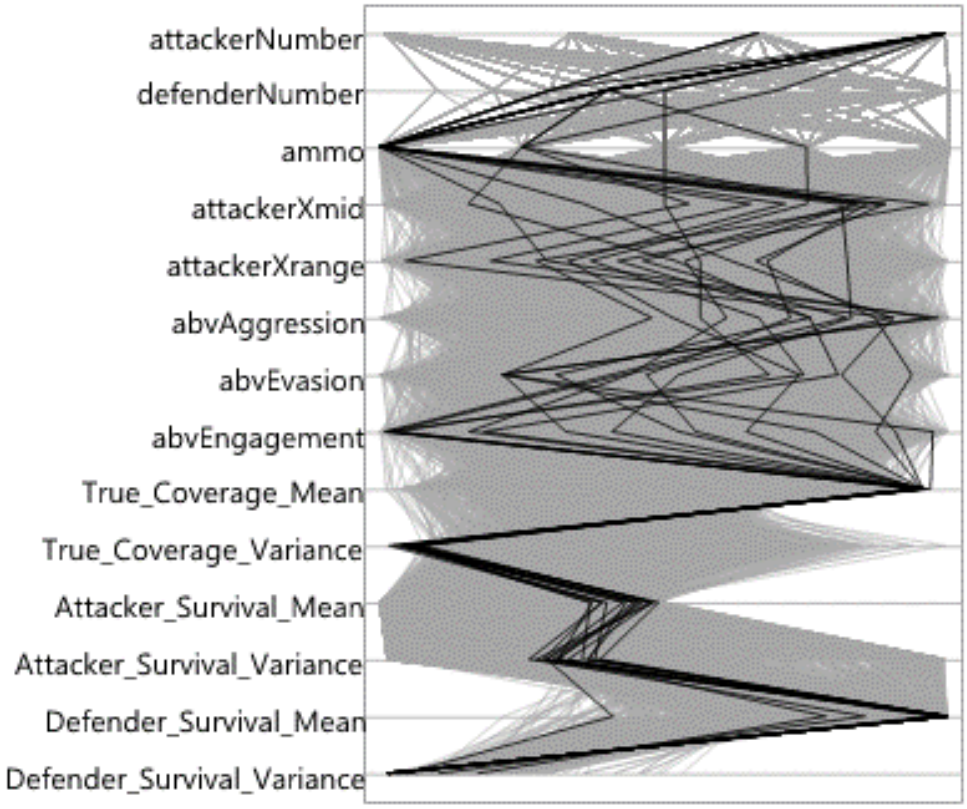



Figure 7.17: Parallel Plot for Low Attacker Survival (< 50%)

As illustrated by Figure 7.17, the majority of the low Attacker survival rate cases were unified in their lack of ammunition but with a high Aggression ABV. This matches a scenario discussed in Chapter 4 where Attacking agents would risk their survival by ignoring Defenders. The twist here is that providing no ammunition to the Attackers prevents any form of self-defense as would be otherwise available in the high-aggression branch of the Decision Tree.

Further analyses are possible due to the flexibility of the plotting method. The use of Parallel Plots is a useful tool to observe Mission Design Space MOE effects and interactions. The limitation is that the process is iterative, manual, and observation-based.

### 7.4.2 Ordinary Least Squares Regression Effects

The Ordinary Least Squares Regression had the worse fit of the two Surrogate Modeling methods considered. However, it still provides some useful insights and "sanity checks" on which Mission Design Variables influence the outcome distribution and MOEs. This is primarily achieved by investigating the effects breakdown generated from the fits through JMP. These diagrams illustrate which inputs and interactions have the largest influence on the output metrics. Those interactions are quantified via Pareto Plots. While the ANN fits were superior to the regressions, there is limited ability to create such effects plots due to the "hidden" nature of the interior of a Neural Network.

Figure 7.18 provides the variable contribution and effects breakdown for the % region coverage MOE.

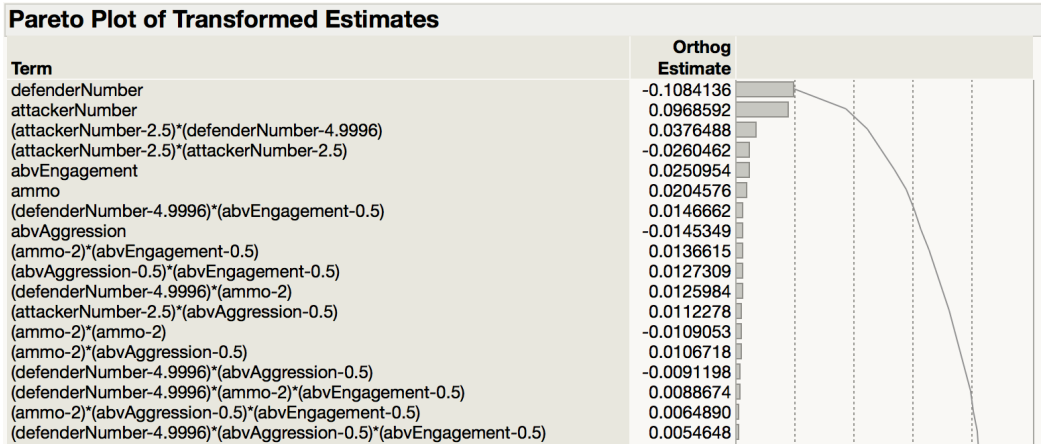



Figure 7.18: Pareto Plot of the Least Squares Regression for the Surveillance Coverage Distribution Parameters

For the surveillance coverage, the major Mission Design Variables appear to be the populations of each side, which confirms some of the underlying principles in the Lanchester Laws. Also of importance are interactions between various ABVs and Mission Architecture parameters. The ABVs were important not in isolation, but notably in the form of interactions. The interactions most of note are between "chained" ABVs such as the Aggression and Engagement, since reaching the engagement Decision Tree branch

is dependent on the aggression value. Also of interest are the relations between the Engagement ABV and the amount of ammunition assigned to each Attacker. That relationship is expected as weaponry is meaningless if not used while a "trigger-happy" Attacker cannot do much when assigned no weaponry.

Delving deeper into the Attacker survival rate parameters, similar influencing inputs appear as illustrated by Figure 7.19. The populations of each side remain most important, though key ABVs and ABV - design interactions also exist.

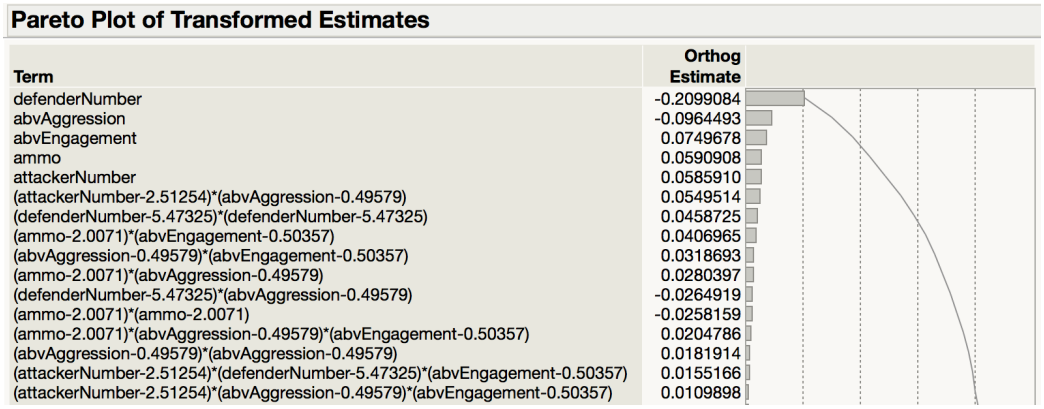



Figure 7.19: Pareto Plot of the Least Squares Regression for the Attacker Survival Rate Distribution Parameters

For the Attacker survival rate, the major design variable interactions are between the Attacker population, Defender population, and the combat-focused ABVs. Similar ABV interactions exist to the coverage effects plot (Figure 7.18), notably the Engagement and Aggression ABVs. The similarity in critical influencing parameters for the surveillance coverage and Attacker survival rate MOEs is expected as the ability to survey a region is directly linked to the Attackers surviving long enough to reach the areas of interest.

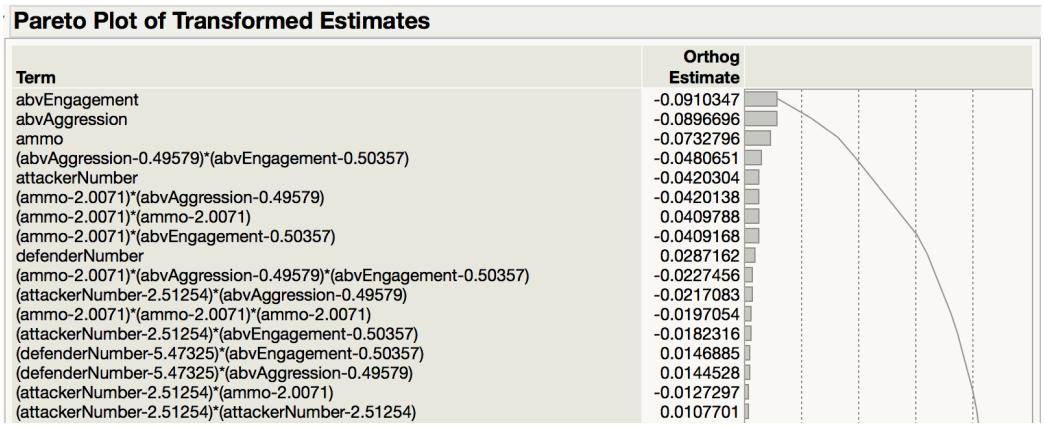



Figure 7.20: Pareto Plot of the Least Squares Regression for the Defender Survival Rate Distribution Parameters

In the Defender survival rate most affecting design variables (Figure 7.20), the order shifts dramatically. The dominant design variable effects are the decision-making ABVs, the Attacker population, and the amount of ammunition available to each Attacker. Furthermore, these effects are so dominant that they outweigh the more conventional influence of the Defender population. These critical interactions are sensible though, with the same general interactions observed previously between "chained" ABVs like Aggression and Engagement, but also the Engagement ABV and ammunition. The Defenders, as passive entities, are entirely dependent on the actions of the Attackers to define their survival. If the Attackers never encounter or elect to engage a Defender, the Defender will survive.

## 7.5 Comparison with a Lanchester-Based Engagement and Mission Model

To better evaluate and understand the proposed **SAA**, the fit models developed will be compared against a more conventional approximation of mission performance. This baseline is derived from the Lanchester Equations to represent combat and a similarly differential equation estimation of surveillance times. The Contested Reconnaissance Model will be used as the mission of comparison.

All the various agent properties discussed here are the same as for the full ABM formulation of the Contested Reconnaissance Model. Furthermore, the values remain notional and semi-unitless as the model is intended to be illustrative not authoritative.

### 7.5.1 The **SAA**-Based Mission Model

The mission model using the proposed methodology is unchanged from the Contested Reconnaissance Model developed as part of this Chapter. The Artificial Neural Network approximations of the tradeoff space will be used to evaluate different Mission Design Points. As the Lanchester-based approach is ultimately deterministic, only the fit for the mean value of each output will be considered and compared.

### 7.5.2 The Lanchester-Based Mission Model

The Lanchester Laws possess two value types which vary during a simulation: the populations of each side and the elapsed time. The coefficients on the Lanchester differential equations will be fixed for each simulation and estimated from the combat characteristics of the units involved. As the model will not encompass active decision-making, the assumption is that the Attackers will conduct a grid-search in a regular pattern. This search will be conducted regardless of the Defender placement but Attackers will engage every Defender within range.

The first step is to develop the Lanchester equations which will represent the combat in the model. As discussed previously, the basic form of these equations are seen in Equation 7.1

$$\begin{aligned} \frac{dA}{dt} &= c_1 * D \\ \frac{dD}{dt} &= c_2 * A \end{aligned} \tag{7.1}$$

The populations of each side ($A$ and $D$) are design variables from the same DoE used for Research Question 5, or the number of Attackers and Defenders in the model. The

outputs are those two populations as well as the surveillance coverage ($S$). Additionally. when computing the effects of unit performances, the population of each side is rounded up to account for how units in the ABM formulation act at full efficiency until they are destroyed. All intermediary values are identical to those found in the Contested Reconnaissance Model.

*Defender Firepower*

To simplify unit placements, it will be assumed that the Defenders do not have overlapping areas of effect. This means that the total dangerous areas for the Attackers are a series fo $n$ circles of radius equal to the range of each Defender (80). Meanwhile, the Attacker speed is 20 and the Defender damage rate is 12.62 per timestep. With an Attacker health of 100, that value is normalized to be proportional to the total unit health. This implies that an Attacker can traverse 160 units under the effect of a Defender before it is destroyed. However, not every area is defended and so the damage rate must be amortized by the probability at a given timestep that the Attacker is within range of a Defender. Thus the coefficient for Defenders damaging Attackers resolves to Equation 7.2.

$$\begin{aligned} \frac{dA}{dt} &= \frac{damage}{time} * \frac{areaDefended}{areaTotal} \\ \frac{dA}{dt} &= \frac{12.62}{100} * \frac{D * \pi * 80^2}{1000^2} \\ \frac{dA}{dt} &= 0.0025 * D \end{aligned} \tag{7.2}$$

*Attacker Firepower*

The Attackers have a weapon range of 60 but also destroy Defenders near-instantaneously. This means that their damage per time is 1.0 defenders per timestep (if within range). Meanwhile, the same expected encounter rate formulation can be adapted to be reflective of regions where the Attacker would expect to have a shot against a Defender. As the

Attackers are the active units, their firepower is a function of the number of Attackers directly.

$$\begin{aligned} \frac{dD}{dt} &= \frac{damage}{time} * \frac{areaAttacker}{areaTotal} \\ \frac{dD}{dt} &= \frac{1.0 * A}{1.0} * \frac{D * \pi * 60^2}{1000^2} \\ \frac{dD}{dt} &= 0.011 * A * D \end{aligned} \tag{7.3}$$

Unlike the Attacker attrition rate, the Defender attrition rate is a representation of the Lanchester Linear Law in that the populations of both sides affects the attrition rate.

*Elapsed Time*

The elapsed time of the simulation is a function of how much surveillance is achieved. The surveillance coverage is a monotonically increasing value that is a function of the number of Attackers remaining and the sensor coverage range. As detailed decision-making is not simulated, this model will assume that the Attackers will be semi-collaborative and thus will not double-up or repeat coverage of areas.

In the agent-based Contested Reconnaissance Model, the Attackers have a perfect (notional 100% quality) at a range of 50 with quadratically decaying coverage to be a notional 0 at a range of 100. For simplicity, only the perfect coverage region is considered. For a given timestep, the new area surveyed by an Attacker will be a function of its speed and sensor range as articulated by Equation 7.4, where $d$ is the distance between the centers of the two surveyed areas (the distance the Attacker traveled) and $r_1 = r_2 = 50$ is the surveillance range. Equation 7.5 and Equation 7.6 support Equation 7.4 by solving

for $S_{survey}$ and $S_{overlap}$.

$$\begin{aligned}
\frac{dS}{dt} &= A * (S_{survey} - S_{overlap}) \\
\frac{dS}{dt} &= A * (7853.98 - 5867.39) \\
\frac{dS}{dt} &= A * 1986.59
\end{aligned} \tag{7.4}$$

Equation 7.5 calculates $S_{survey}$ defined as a circular region within the Attacker surveillance range (50 distance units).

$$\begin{aligned}
S_{survey} &= \pi * {r_2}^2 \\
S_{survey} &= \pi * 50^2 \\
S_{survey} &= 7853.98
\end{aligned} \tag{7.5}$$

Equation 7.6 calculates the overlap between the old and new surveyed region as found through trigonometry operating on the union of two circular regions.

$$\begin{aligned}
S_{overlap} &= {r_1}^2 \arccos(\frac{d^2 + r_1^2 - r_2^2}{2dr_1}) + {r_2}^2 \arccos(\frac{d^2 + r_2^2 - r_1^2}{2dr_1}) - \frac{1}{2}\sqrt{4d^2r_1^2 - (d^2 - {r_1}^2 + {r_2}^2)^2} \\
S_{overlap} &= {r_1}^2 \arccos(\frac{d^2}{2dr_1}) + {r_2}^2 \arccos(\frac{d^2}{2dr_1}) - \frac{1}{2}\sqrt{4d^2r_1^2 - d^4} \\
S_{overlap} &= 2r_1^2 \arccos(\frac{d^2}{2dr_1}) - \frac{1}{2}\sqrt{4d^2r_1^2 - d^4} \\
S_{overlap} &= 2 * 50^2 \arccos(\frac{20^2}{2 * 20 * 50}) - \frac{1}{2}\sqrt{4 * 20^2 * 50^2 - 20^4} \\
S_{overlap} &= 6847.19 - 979.80 \\
S_{overlap} &= 5867.39
\end{aligned} \tag{7.6}$$

The total coverage value is thus increasing according to Equation 7.4 and the objective coverage value is 80% or $0.80 * 1000^2 = 800,000$ for the 1000x1000 region used in this simulation. Thus the simulation will continue to run and attrition will continue to accumulate on the Attackers and Defenders until that total coverage is reached, or the

Attacker population reaches zero.

*Model Architecture*

The model is a time-discretized representation of the differential changes affecting the three primary parameters: the Attacker population, the Defender population, and the surveillance coverage amount. Each timestep, these values increment according to the Lanchester Equations or the surveillance rate described previously. Each step is affected by the values of the step previous. The simulation ends either when all Attackers are destroyed or when the surveillance objective has been met.

*Results*

The results from the Lanchester model formulation are the time-histories of the various metrics of interest. Those time-histories of the two force populations are akin to Figure 3.8, Figure 3.9, and Figure 3.10, which covered more notional examples of differential-equation based engagement. In this scenario, there is the addition of a third metric of interest, the surveillance coverage. Plotting these three against each other yields Figure 7.21.

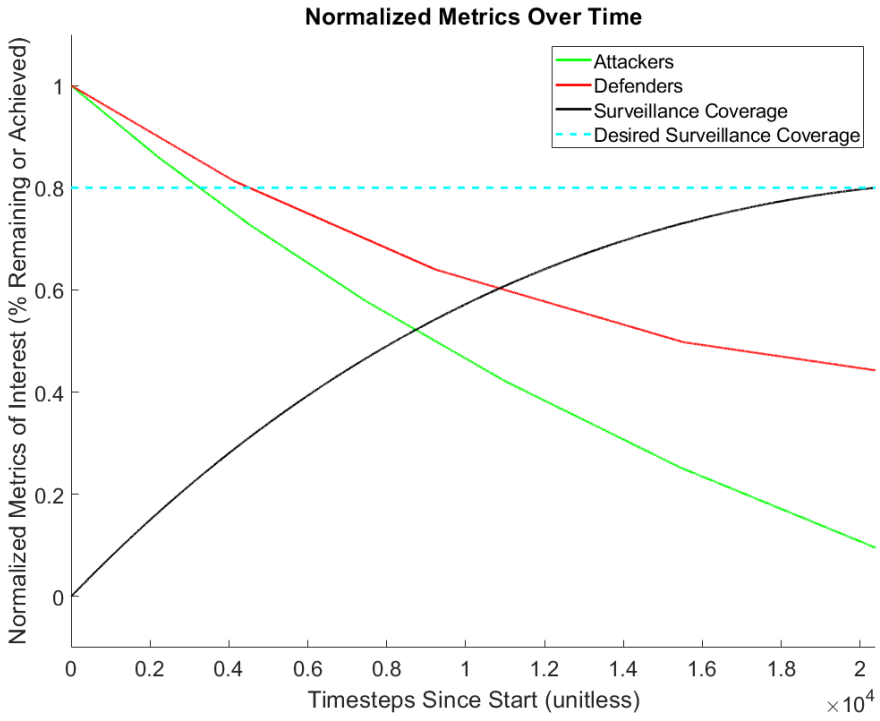



Figure 7.21: Illustration of the Proportional Metrics of Interest for a 4 Attacker, 10 Defender Scenario

Notable in Figure 7.21 are the "kinks" in the lines representing the Attacker and Defender populations. This is due to the attrition rates being a function of the rounded-up number of units remaining, based on the aforementioned assumption that a damaged unit can still engage with full capability. Furthermore, the effects of the decreasing Attacker population are apparent in the concavity of the surveillance coverage metric. As Attackers are lost, the rate of surveillance gain decreases, thereby slowing the rate at which the metric approaches the 80% coverage objective.

The trends of all three metrics are qualitatively appropriate and the interrelationship between their differential equation representations are thus verified. However, the model itself is of less interest than a comparison between it and the **SAA** which was the objective of this section.

### 7.5.3 Qualitative Comparison of Approaches

The Lanchester-Based mission model, while efficient and clear to develop, misses much of the fidelity that is present in the EBCM underpinning the **Stochastic Agent Approach**. As such, while final outputs are provided, they are ultimately less clear due to the aggregation of the populations of each side. For example, a force strength of 2.0 might correspond to four units of 0.5 "health" or two units of 1.0 "health." The Lanchester-based approach cannot distinguish the two whereas the ABM would. Similarly, it is difficult to account for specific munitions and maneuvers in the Mission Design Space in the Lanchester context. Thus only two of the original Mission Design Variables can be directly evaluated between the two methods: the number of Attackers and the number of Defenders.

The Lanchester-based model was much simpler to implement and possessed a shorter runtime when accounting for the number of simulations needed to inform the **SAA**. However, that comparison is incomplete as there should also be (or have been) a dataset to refine and calibrate the Lanchester Laws. However, in the absence of an appropriate background, subject matter expertise and basic calculations were substituted. The approximations thus employed lack the traceability of the full ABM used when employing ABVs. That lack of traceability makes future use of the approach more complex and hinders model improvement.

### 7.5.4 Numerical Comparison of Approaches

To facilitate a one-to-one numerical comparison, their mission design points must be as similar as possible. As the Lanchester-Law based model assumes aggressive Attackers, the **SAA** model will default the Agent-Based Variables to encourage those aggressive actions, namely a value of 1.0 for both the Aggression and Engagement ABVs. Similarly, the ammunition assigned to each attacker was maximized (4) as the Lanchester model does not have any accounting for munition capacity. Lastly, the Attacker start location

properties were defaulted to minimize their influence on the outcome (a centralized and maximum breadth starting formation).

A full-factorial combination of design points was generated according to Table 7.9. Note that only the numbers of Defenders and Attackers are actually varied in the Lanchester-based model.

Table 7.9: Design Variables for the Comparison of Mission Simulation Methods

| Variable | Type | Minimum Value | Maximum Value |
|---|---|---|---|
| Number of Defenders | Variable | 0 | 10 |
| Number of Attackers | Variable | 1 | 4 |
| Ammo | Fixed | 4 | - |
| Attacker Xmid | Fixed | 500 | - |
| Attacker Xrange | Fixed | 100 | - |
| Aggression | Fixed | 1.0 | - |
| Engagement | Fixed | 1.0 | - |
| Evasion | Fixed | 0.5 | - |

With these cases defined, the ABV Neural Network Approximations and the Lanchester model were used to generated expected outcomes. Those outcomes were then compared with the assumption that the "true" value originates from the higher fidelity Agent-Based Model formulation. The resulting distribution on those differences are seen in Figure 7.22, Figure 7.23, and Figure 7.24. Those histograms provide the % difference along the x-axis against the proportion of cases with that difference along the y-axis.

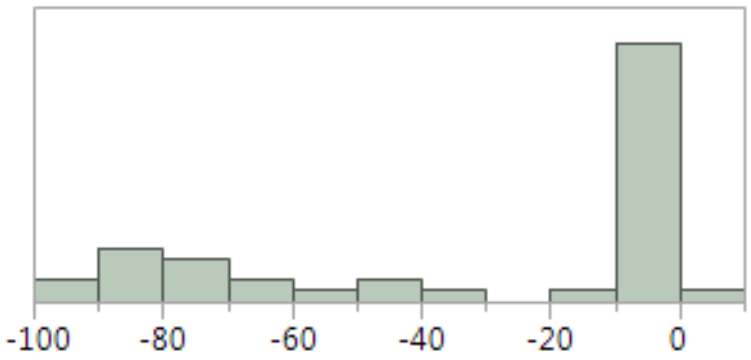



Figure 7.22: Histogram of the Percent Variation Between the Coverage of the **SAA** and the Lanchester Formulations

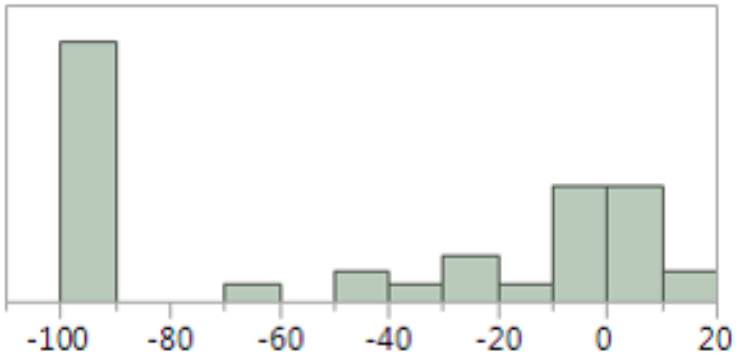



Figure 7.23: Histogram of the Percent Variation Between the Attacker Survival Rate of the **SAA** and the Lanchester Formulations

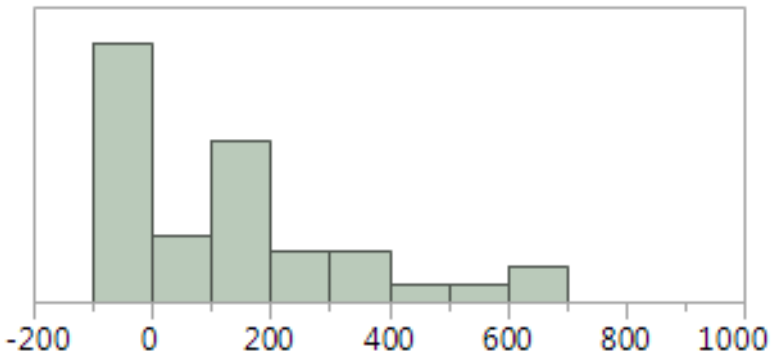



Figure 7.24: Histogram of the Percent Variation Between the Defender Survival Rate of the **SAA** and the Lanchester Formulations

Notable in these figures is that only the approximated coverage amount is similar between the two formulations. The similarity of the survival rates is poor, which is also expected given that there are fewer assets simulated than the Lanchester Equations are conventionally used for. Ironically, the use of the attrition and combat-focused Lanchester Equations only generated comparable non-combat metrics, while the MOEs of engagement were poorly matched.

### 7.5.5 Comments

Once more, the differential equation representation of missions and combat is simpler to implement than a full Agent-Based Model, but by definition loses fidelity. Consequently, there is a clear and illustrated weakness when applying these sorts of aggregate engagement representations to small-scale and higher-fidelity physics-based scenarios. The mission simulation is too small in scope to amortize the combat results over the population. As such, empirical approximations are less refined than those derived from more detailed M&S which could be improved by the use of the **SAA**. When infusing Agent-Based Variables into the model, there is an improved ability to understand trade-offs beyond performance attributes and force populations, something which is more

difficult to traceably capture in differential equations or other approximations.

## 7.6 Conclusion

The **SAA** is compatible with all Surrogate Model forms, but considered two popular forms (Ordinary Least Squares Regression and Artificial Neural Network) for fitting the notional Contested Reconnaissance Model. While this model is not exhaustive or universal, the result from these experiments is that a Neural Network Surrogate Model is the most appropriate to fit the output metrics of interest. However, superior fit methods may exist and future work should investigate additional approaches as necessary.

As a point of comparison, differential equation-based (Lanchester Law) mission approximations were generated for the same scenario. The verified approximations only generated comparable results in terms of surveillance coverage, not the direct combat-influenced parameters of the agent survival rates. The fidelity reduction required for the Lanchester Law formulation is thus contrasted with the ANN Surrogate Model. Most notably, the variable space and decision-making simplification appeared to hamper the full-scope utility of the differential equation-based models.

Despite the limitations in the approximation functions generated from the ANN and Ordinary Least Squares Regression, these models and their Surrogate Models are much more traceable than the heuristic or empirical relationships which underly the Lanchester Laws and similar formulations. The simulation cases through the ABV-based Agent-Based Models can be further dissected and discussed in a way which would not work for an aggregate model, as seen through Parallel and Pareto Plots. The actual tracing of unit actions through a simulation, and the resulting use of continuous decision-making parameters enables a degree of observation not leveraged previously. Interactions between decision-making and engineering design parameters are now more quantifiable and observable. Though these relationships are not perfect, their underlying trends deserve further study and investigation.

# CHAPTER 8

# CONCLUSIONS

## 8.1 Completed Stochastic Agent Approach

The **SAA** develops a means and process for generating appropriate estimations of mission performance and capabilities for infusion into higher-level campaign and life-cycle analyses. As these models historically relied on empirical or SME-derived information, there is a demand for more traceable and physics-based approximations. Furthermore, the method infuses variable decision-making into the simulated missions, providing further insights into potential improvements or effects from tactical changes or preferences.

### 8.1.1 Selection of a Data-Generation Model

There are numerous methods for generating the data for a higher-level mission approximation. The three main archetypes are full-scale exercises and testing, human-in-the-loop war games, and computer simulation. Due to the need for a large, physics-informed Mission Design Space, computer M&S is the only logical outcome. Furthermore, within computer M&S there are numerous forms of models. As the goal is to develop approximations of higher-fidelity simulations for infusion into larger-scale ones, the most logical model form is the Entity-Based Combat Model which simulates individual (or small groups) of units and tracks their decisions, actions, and munitions. While it is the highest-fidelity computer model archetype, the EBCM is actual quite manageable depending on the user's appropriate modulation of its physics and decision-making fidelities.

**SAA** Step 1

Develop an Agent-Based Model of the problem with appropriate levels of both physics and decision-making fidelity.

When developing the agent decision-making, ensure a reasonably balanced process such that there is no dominant branch with the majority of the end-states.

Specify both Mission Architecture variables of interest (e.g. number of units, type of units, munitions) but also output metrics of interest.

### 8.1.2 Addressing the Massive Mission Design Space

Should the Mission Design Space be decomposed akin to the Engineering Design Space, the resulting span and number of variables is intractable. Every entity in the EBCM takes numerous actions during the simulation and there is a large number of variables simply to define a single action at a single point in simulated time.

As such, the variable space is to be transformed from one focusing on the actions in the mission simulation to one focusing on the decision-making. This introduces Agent-Based Variables which convert the deterministic and binary nature of Agent-Based Model decision-making into a continuous variable space. As illustrated with a basic strike mission case study, shifting the variable space to focus on decision-making not only enabled more effective exploration of the Mission Space, but also could generate mission plans similar to those from more costly optimization approaches.

### 8.1.3 Defining Agent-Based Variables

Agent-Based Variables are the key and novel contribution of this research. Previous efforts in Agent-Based Models leverage deterministic decision-making with stochastic elements in the form of noise or probabilistic $P_{kill}$. Even the application of decision-

making routine optimizations such as Dynamic Scripting still result in deterministic agents. Converting to a variable space similar to Dynamic Scripting would not solve the Mission Design Space scoping problem as there would simply be a massive number of Boolean values (or conditional checks) defining which decision-making routines are maintained.

ABVs instead focus on the branches of an appropriate fidelity Decision Tree for an agent. As the agent's decision-making moves through the conditional statements, the transitions between branches is no longer deterministic. Instead, the ABV defines a probability of following a given branch given the same stimulus. This stochasticity and variability not only "fills in" the Mission Action Space, but also enables easier generation of DoE and Surrogate Models by converting categorical variables into continuous ones. Furthermore, reliance on the Decision Tree (or State Machine, depending on formulation) also enables an adaptive approach based on the desired fidelity. As subject matter expertise is already required for ABM development, it would be trivial to leverage that knowledge for ABV identification.

**SAA** Step 2

Investigate the decision-making algorithm in the Agent-Based Model to identify possible Agent-Based Variable branches.

Classify these branches as Trivial, Relational, or Navigational depending on their effects or influence on agent behaviors.

An effective Agent-Based Variable is a decision where "personal preference" could reasonably affect the action selected without being trivially answered by physics or another unbiased assessment.

### 8.1.4 Sampling Across the ABV-Involved Mission Design Space

With the introduction of Agent-Based Variables, any employed sampling methods must be adjusted. The first step is addressing the inherent stochasticity of the approach by identifying a necessary per-design point replicate number. This was accomplished by first identifying the Mission Design Point of highest variability, hypothesized to be the combination of Agent-Based Variables which resulted in an equal likelihood of each Decision Tree branch (i.e. completely random agent behavior). However, experimentation revealed that many design points had similar variability. Thus, the point of highest stochasticity is unclear for the Contested Reconnaissance case study, though it is clear which points are not useful. The result was the selection of the ABV variable combination in the middle of the space.

With an appropriate point identified, the next step is to utilize the Bootstrapping approach to identify the lowest replicate number which sufficiently captured the distribution properties of a massive "truth" dataset at that Mission Design Point. That replicate number (found to be approximately 200 for the Contested Reconnaissance Model) was the conservative necessary replicate number and, in the interest of simplicity, is used for all other Mission Design Points as well. That replicate number is higher than ones suggested by various "rules of thumb," which was expected due to how the stochastic agent decision-making has a "chaining" effect on the course of the mission.

**SAA** Step 3

Identify a design point(s) of non-trivial variability where the agent actions are not overly-deterministic (i.e. towards the extremes of the Agent-Based Variable values).

With the design point(s) selected, run a large number (1000) of replicates and use Bootstrapping to identify the number of necessary replicates for convergence of the metrics of interest.

Lastly, while ABVs do affect the mission simulation in a less conventional manner, they are still ultimately continuous design variables and so conventional Design of Experiments approaches remain valid. For the **SAA**, a Latin Hypercube DoE is used no only for the precedent in a mission simulation context, but also because Space Filling Designs are a logical choice when generating cases for eventual Surrogate Model development. There is no requirement in an ABV context to use the Latin Hypercube, a fortunate flexibility originating form this novel variable space formulation.

**SAA** Step 4

Run the cases to inform the Mission Space Exploration as identified by a Space-Filling Design of Experiments of choice (Latin Hypercube is recommended but not required) with the replicate number found through Bootstrapping.

Collate the data for each design point into distribution metrics for the outputs (mean, variance, and higher-level moments as desired).

### 8.1.5 Developing Approximation Equations

Surrogate Model generation is the final process for the **SAA**. The main goal is to define approximations for the Measures of Effectiveness across the various simulated cases. Due to the stochastic nature of the involved Agent-Based Model, the Surrogate Models must be fit for descriptive metrics of the MOE distributions (e.g. mean and variance) rather than singular numeric values. The resulting distributions can be inserted as approximations of mission performance much like how random chance is already used in failure or other stochastic process analysis.

While the fits for the Contested Reconnaissance Model are not perfect, they are traceable in a manner unlike existing approximation methods. The use of continuous decision-making variables in the form of ABVs facilitates this understanding. The Lanchester Equations and other current approaches are similarly imprecise, but make the approximations at the simulation level rather than at the Surrogate Model level. As such, the **SAA** is promising for infusion into higher-level models in lieu of existing estimation methods, primarily because superior or new Surrogate Modeling approaches can be utilized.

**SAA** Step 5

Fit a Neural Network (or identified superior fitting method) to the resulting distribution metrics of the outputs.

Use the equations from the fit as the "black box" equations in the higher level models or Discrete Event Simulations where there is not a detailed simulation of the involved engagements.

### 8.1.6 Completed Process for the Stochastic Agent Approach

Unifying the steps described in previous sections yields the final **SAA** as articulated in Figure 8.1.

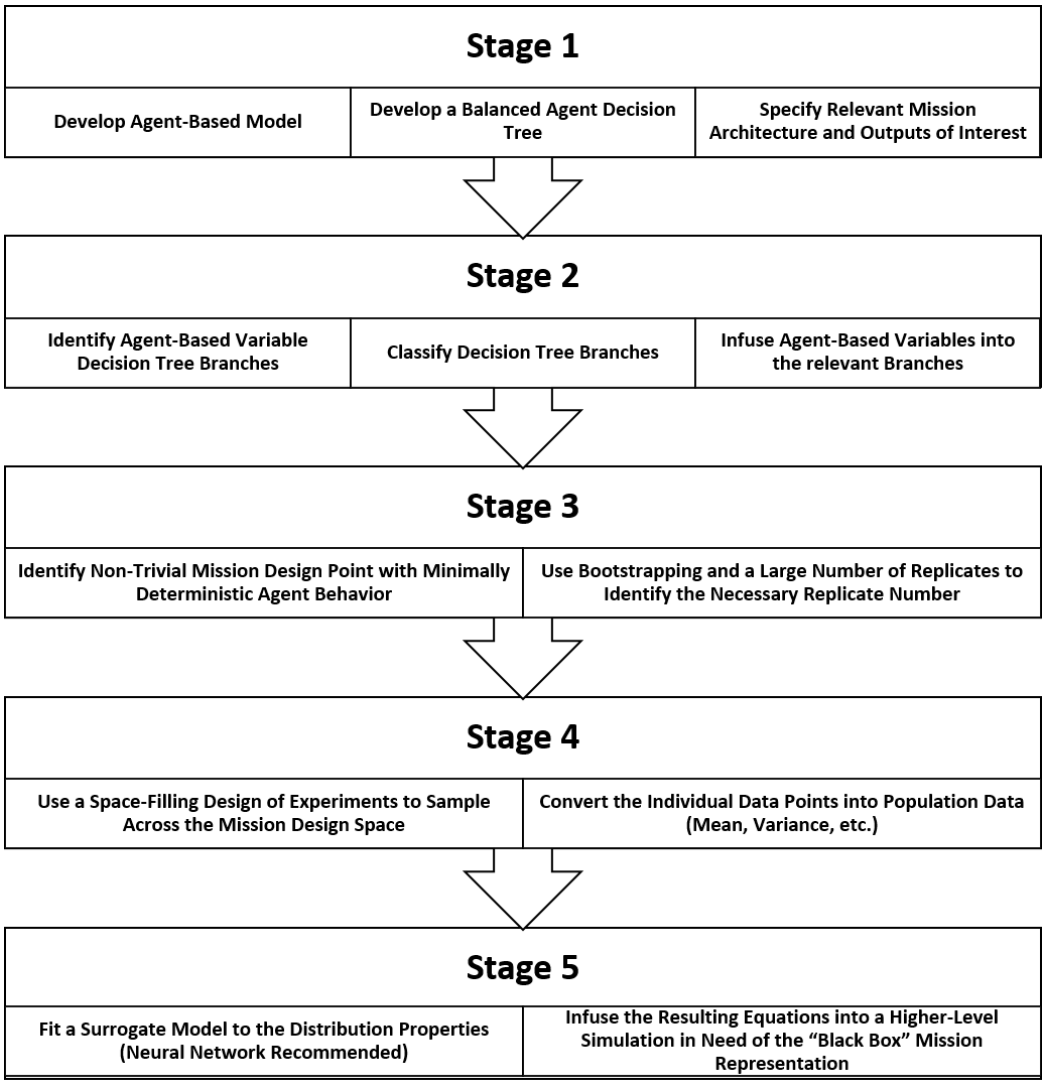



Figure 8.1: Flowchart of the Completed **SAA** Process

## 8.2 Research Contributions

Ultimately, the main goal of this research is to introduce decision-making and thus tactical-level effects into the trade space. Previous efforts either ignored these lower-

level effects or instead sought to optimize them in narrow contexts. This document describes the efforts to infuse these new variables and considerations into a mission-level simulation and solution space.

To achieve that objective, the main contribution of this research is the introduction of probabilistic decision-making for Mission Space Exploration using an Agent-Based Model. The formulation and identification method for Agent-Based Variables are novel and provided as is discussion of different supporting methods. ABVs are continuous and bounded values, which not only facilitates their use in a Mission Design Space, but also makes them more amenable to conventional variable space formulations, samplings, and approximations. Key research and investigation of mission simulation approaches, decision-making models, design space exploration means, and data analysis are all present and in support of the **Stochastic Agent Approach**.

## 8.3 Future Work

With the proposed implementation of the **SAA**, future work should focus on the utilization of this method with other combat simulation environments alongside general improvement of the method elements. Numerous commercial softwares exist for the evaluation of assets through missions, and leveraging the validated capabilities of these tools will improve the reach and applicability of the detailed approach. If these favored tools by various users can be incorporated and used alongside the **SAA**, then the method will have an even greater impact on the field through building upon the trustworthiness of these proven environments.

### 8.3.1 Infusion into a Campaign-Level Model

The detailed **Stochastic Agent Approach** develops a series of Surrogate Models for MOE distributions of missions. However, these equations have not yet been propagated or infused into a higher-level simulation. Future work could investigate and specifically

apply these ideas to a higher-level model to confirm the utility of such approximations. While such infusion should be straightforward, confirmation of their utility would reinforce the method laid out in this research.

### 8.3.2 In-Simulation Variable Agent-Based Variables

Agent-Based Variables provide insight by converting decision-making into a stochastic (or "preference-based") process rather than a deterministic one. However, such an implementation need not require fixed ABV values for the duration of a simulated mission. The parameters could be a function of the stage of the mission or even the state of allied units. While such considerations are seen in deterministic ABMs, they could equally be applied to an ABV-infused model, provided the result still maintains some decision-making stochasticity.

### 8.3.3 Utility for an Evaluation of Alternatives

An added benefit of introducing decision-making and mission actions as design variables is the potential for their use in improved comparisons of solution alternatives. With tradeoff spaces generated for both Mission Architecture (e.g. the types of forces used) and Mission Action variable spaces, there is a superior ability to compare alternatives at their best combination of mission and design parameters rather than through a fixed mission profile. As epitomized by the JCIDS process, solutions can be generated through a mix of new systems and new uses for existing systems, a unified variable space which previously had not been considered in its entirety due to static mission profiles or a reliance on less traceable mission approximations.

### 8.3.4 Improving Replicate Number

An area of improvement would be to identify a method for generating an adaptable replicate number for each Mission Design Point. The **SAA** set a fixed replicate number

based on the number necessary at the point of highest variability. However, that experiment also illustrated how certain design points are much less variable in their Measures of Effectiveness, such as certain deterministic agent behaviors yielding consistent success or failure.

Future work could modify adaptive sampling methods to the generation of replicates. As the output distribution for a given case stabilizes and converges, the replication process could be curtailed, much like an adaptive Design of Experiments [137]. This would save computational time by avoid unnecessary replication or the overly conservative replicate estimate found by investigation in this work.

### 8.3.5 Superior Surrogate Models for Small Entity Number Missions

A developed Agent-Based Model is capable of handling and simulating a wide variety of mission scopes and scales. However, as observed with the Contested Reconnaissance Model, the traditional Surrogate Modeling techniques had difficulty when fitting against small entity numbers. This is seen most readily when comparing the fit qualities of the Attacker survival rate and Defender survival rate. Surrogate Modeling techniques which focus on discrete variables may be more appropriate in this context.

Furthermore, the Ordinary Least Squares Regression and Artificial Neural Network did illuminate useful relationships and understanding, but were not the unambiguously good fits as desired. Future work could investigate either new model fitting techniques or perhaps selecting different Measure of Effectiveness distribution properties to be estimated.

### 8.3.6 Development of Universal Methods and Model Forms

The **SAA** developed an method to answer various questions about which Design of Experiments or Surrogate Model to use. However, that approach was primarily applied to the notional Contested Reconnaissance Model. As a result, there is no claim of universal

applicability of the selected methods.

Future work could conduct further experiments and observe how readily or universally these specific sampling and fitting methods are to other mission or engagement problems. The **SAA** can be used to set up these problems and investigate the specific approaches, but only by leveraging other models or scenarios will there be a more thorough investigation of how universal these selections are.

### 8.3.7 Utility of the **SAA** Method for Policy Decisions

Supplementing the strategic and military decision-making influenced by **SAA**-informed models, there is also a possible utility in affecting broader policy and acquisitions decisions. The primary improvement offered by the proposed method is additional accounting for tactical and individual decision-making as part of the Mission Design Space. With these new variables and interactions now visible, there is an improved ability to account for more aspects of the solution space within a single simulation. This superior solution identification could be employed directly during the acquisitions process to compare alternatives (both matériel and non-matériel). Most notably, the critical inclusion of variable mission operations alongside variable system parameters is necessary for more robust tradeoffs.

Furthermore, various war games conducted by both military and civilian agencies will be improved through the infusion of more stochastic decision-making. Human beings do not function deterministically, especially during periods of high stress, and so the inclusion of ABVs into the problem can be an important point of understanding through the introduction of decision-based aleatory uncertainty [96]. The Agent-Based Variable values can be permuted relative to the "ideal" deterministic Decision Tree to observe mission robustness to incorrect decisions. Ideally, no mission will fail due to a single "poor decision" at a single time point, but quantifying that vulnerability is useful and provides insight into how resources can be best committed or utilized in crisis.

The Agent-Based Model used to inform the **SAA** can help explore how collaboration or planning can compensate for human imperfections [1, 2].

Beyond the acquisitions or planning processes, there is also the possibility of policy decisions affecting the newly introduced decision-making design variables (ABVs). Policy can affect warfighter morale and consequently, there may be a possible mapping between those non-combat effects and agent decision-making. Demoralized units might be more risk-averse while nationalistic policies could encourage risk-taking.

It similarly might be worthwhile to consider these models when evaluating opponents and their units. Political or other ideologies can encourage some behaviors over others and those effects would be captured through appropriately calibrated Agent-Based Variables. Beyond policy, geographic location (e.g. home-field advantage or motivation) could similarly influence the preferences captured through ABVs.

### 8.4 Final Statements

This document outlines the series of literature reviews, independent investigations, and experiments necessary to contribute to the Operations Research, Systems Engineering, and Aerospace Engineering disciplines. Generation of more physics-informed mission-level approximations enables more effective utilization and conclusions from campaign-level wargaming and life-cycle analyses. The **Stochastic Agent Approach** outlined provides a modular framework for the definition and utilization of the proposed Agent-Based Variables that convert deterministic and discrete decision-making into a continuous variable space. This variable space adjustment enables more conventional design space exploration and approximation methods previously inhibited by the massive Mission Action Space.

Agent-Based Variables enable agent behaviors to become design variables much like more conventional engineering or Mission Architecture parameters. This concept can be expanded to new problems and provide a more quantitative method for observing

and leveraging emergent behaviors. Furthermore, the developed model fits enable a traceability not feasible through empirical approximation methods, while avoiding the use of expensive and time-consuming full-scale exercises or less impartial historical data and subject matter expertise. In short, the approximations are generated after the high-fidelity dataset is obtained, not as a means to generate the data in the first place.

In summary, the **SAA** successfully applies tried and proven variable space exploration and tradeoff methods to the Mission Design Space.

# Appendices

# APPENDIX A

# GLOSSARY OF UTILIZED TERMINOLOGY

## A.1 Capability

A capability is defined as the ability for an entity to properly complete some defined mission. Notably, a capability is defined irrespective of the means utilized to accomplish the mission [50]. As such, it is nebulous in its more specific components or definitions. Capabilities are achieved through a combination of the so-called Doctrine, Organization, Training, Materiel, Leadership, Personnel, Facilities, and Policy (DOTmLPF-P) elements which span a mix of strategic, operational, tactical, and design considerations [74].

## A.2 Capability Gap

A capability gap arises when a given mission cannot be achieved using current equipment (assets) and doctrine. As such, a new solution must be identified that encompasses a broad range of possible options including new procurement, changes to tactics or doctrine, and other such variables. A capability gap does not by definition require a new procurement cycle to address. In fact, the initial stages of the DoD JCIDS solution-generation process aim to utilize the full range of DOTmLPF-P solutions, without the need to acquire new hardware, to solve an identified capability gap [71].

## A.3 Matériel

The United States Department of Defense defines matériel as “all items necessary to equip, operate, maintain, and support military activities without distinction as to its application for administrative or combat purposes” [7]. In the context of this document and research, matériel solutions are specifically hardware and physical items necessary

to bridge a capability gap. A new matériel solution implies some degree of non-matériel adjustment. For example, one cannot fly a *F-35* identically to an *A-10*, despite the former ostensibly replacing the latter [82, 84]. Different matériel solutions are considered through the Engineering Design Space wherein the available variables relate specifically to the physical parameters (e.g. wingspan, propulsion properties, payload).

### A.4 Strategic, Operational, and Tactical

Strategic approaches to missions focus on the overall architecture of the solution as well deployments, maintenance, and infrastructure [4, 5]. Strategic concerns include things like the nation's "Naval Posture", which is notably major fleet concerns outside of individual engagements [4, 6, 7].

Operational concerns balance around the specific assets available in a singular context like fleet groups and lower-level SoS concerns [4, 6, 7].

Tactical concerns are more limited in both scope and scale. The assets and opportunities available for a tactical mission are frequently well-known. Furthermore, interactions are frequently limited to within the tactical context rather than the higher-level or longer-timescale concerns of operational or strategic [6, 7].

In the context of this research, strategic, operational, and tactical considerations represent different layers of simulation. Strategic concerns affect overall priorities and where assets are in a theater of operations. Operational concerns affect priorities and deployments within subregions of that theater of operations. Lastly, tactical concerns are within individual engagements. Assets available in a given engagement as well as the mission objective are specified by strategic and operational concerns and effectively fixed for the duration of a given engagement [6, 7].

## A.5 Mission

The United States Department of Defense defines a mission as "the task, together with the purpose, that clearly indicates the action to be taken and the reason therefore" [7]. A secondary definition details "in common usage, especially when applied to lower military units, a duty assigned to an individual or unit; a task" [7]. In the context of this document and research, the mission refers to the non-matériel parameters of a solution to a capability gap. The mission directly includes some matériel elements, but is specifically related to the non-technical aspects such as the number and flight path of the assets.

Notably, "mission" in the context of this research is constrained to be a small-unit or tactical level engagement or activity.

Different mission solutions are considered through the Mission Design Space wherein the available variables relate specifically to the conduct of systems in the mission (as constrained by their performance and other Engineering Design Space properties). In theory, some element of a mission are truly platform agnostic. For example, no matter the type of car used, the route between two points is broadly identical, though the specific driving actions undertaken will vary due to the differing capabilities of the vehicles.

## A.6 System

A system is defined by the United States Department of Defense as "functionally, physically, and/or behaviorally related group of regularly interacting or interdependent elements; that group of elements forming a unified whole" [7]. For the purposes of this research, the words system and asset will be interchangeably and both refer to a single, independently maneuvering or non-maneuvering unit whole. Examples of a system would thus be a singular aircraft, ground vehicle, stationary radar installation, etc. While these entities are all examples of several subsystems cooperating as a whole unit,

the specific actions by subsystems are not of interest to this work. Instead, the models of these systems will incorporate the attributes of the subsystems as they affect the system-level metrics.

Numerous interacting systems can in-turn be considered a System of Systems wherein it is of interest not only the actions by the component systems, but also the interactions between them [141]. As the research proposed by the document requires the analysis of military systems, it is necessary to include the coordination between allies and engagements between opponents. As such, SoS approaches and considerations need to be made during the modeling, simulation, analysis, and evaluation of scenarios.

### A.7 Design Variables

In the context of this document, design variables are the elements of a solution which are available for adjustment [140]. A design variable does not directly imply that the value affects the engineering design of the solution. These variables can be found on any adjustable element of the solution ranging from the flight plan of an aircraft, to that aircraft's particular type, to that aircraft's design parameters such as engine type or wing span.

### A.8 Design Space

The design space is the n-dimensional space generated from the variations in whatever variable form is up for consideration. This space represents all possible combinations of design variables available to the decision-maker. However, a point existing in a design space does not guarantee that it is technically feasible, economically viable, or even that logical an alternative [140].

### A.9 Engineering Design Variables

While design variables are a generic representation, Engineering Design Variables refer in this document to a specific subset. In particular, these are the parameters conventionally associated with the engineering field and directly affect the parameters and technologies considered in a particular system. The Engineering Design Variables solely affect the "means" of a solution in that they only define the raw performance, physical, and physics properties of a system. An example Engineering Design Variable would be an aircraft's wingspan. The Engineering Design Variables compose the Engineering Variable Space.

### A.10 Mission Design Variables

While design variables are a generic representation, Mission Design Variables refer in this document to a specific subset. In particular, these are the parameters conventionally associated with the planners and system operators. The Mission Design Variables solely affect the "ways" of a solution in that they only define the conduct of a system. Many Mission Design Variables are associated with requirements for a particular design. An example Mission Design Variable would be an aircraft's cruising altitude and speed in a particular mission. The Mission Design Variables compose the Mission Design Space.

There is a subset of variables referred in this document as Mission Action Variables (and the associated Mission Action Space). The Mission Action Space is separate from the Mission Design Space. Mission Action Variables are the individual actions and activities in a mission simulation. They are not necessarily design variables, but rather a reflection of the exact "means" undertaken in a solution. Some Mission Design Variables can be Mission Action Variables, but not always. Mission Action Variables can be considered "intermediary" values in the mission simulation.

### A.11 Metrics of Interest

Metrics of Interest can be broadly separated into Measures of Effectiveness (MOEs) and Key Performance Parameters (KPPs).

The Measures of Effectiveness for a given system can be considered the customer's value derived from the implementation and use of that system. They are directly related to the priorities of the customer but their exact formulation is unique depending on both who or what the customer is and what the specific objectives or mission for that system are [42, 43, 142]. The Defense Acquisition University defines a Measure of Effectiveness as a "measure designed to correspond to accomplishment of mission objectives and achievement of desired results" [143]. Despite the prevalence of this and similar terminology as far back as World War II, there is no commonly agreed upon definition of Measures of Effectiveness, nor some convenient list of relevant or applicable MOEs [142].

To simplify the formulation and cataloging of MOEs, the United States Department of Defense (DoD) process decomposes them into Measure of Performance (MOP) and Measure of Suitability (MOS) [143]. While a MOE is a capability-level metric, a MOP tends to be more specifically related to performance. In the case of capability-based approaches, MOPs can be utilized to set lower bounds of acceptable performance [143]. For example, a transport aircraft would need to carry cargo of some minimum weight corresponding to a given piece of equipment (e.g. the weight of a truck).

MOEs are quantified or described through specific Key Performance Parameters [143]. It is these KPPs that can form a "checklist" of requirements that allows for the evaluation of proposed systems and designs. KPPs are more direct measures of performance than the often aggregated MOEs.

## A.12 *A Priori* and *A Posteriori*

*A priori* information is obtained early in the solution-generation process. Webster's Dictionary defines *a priori* as "presupposed by experience" or "formed or conceived beforehand" [106]. In the context of this research, it refers to decisions and actions which occur before any form of modeling, simulation, or iteration occurs. *A priori* information is known about the problem or solution(s) before any work is attempted and can frequently be identified based on subject-matter expertise or historical record. Optimization is a notable case of requiring *a priori* information as the metrics of interest and their relative values usually need to be defined before any cases can be run.

*A posteriori* information is obtained late in the solution-generation process. Webster's Dictionary defines *a posteriori* as "relating to or derived by reasoning from observed facts" [106]. In the context of this research, it refers to decisions and actions which occur after any form of modeling, simulation, or iteration occurs. The development, evaluation, and comparison of solutions all generate the "observed facts" necessary for *a posteriori* decision and actions. Exploration is a notable case of leveraging *a posteriori* information since the options and cases have already been simulated. Given the information from the simulations, metrics of interest can be generated and then compared. These bits of information are not necessary during the simulation(s) because for many explorations, the points are selected *a priori* and the metrics of any one point do not affect the selection or simulation of the others.

## REFERENCES


[1] B. Work and P. Selva, "Revitalizing wargaming is necessary to be prepared for future wars," *War on the Rocks*, 2015.

[2] A. Butler, "Work calls for more wargaming, tech demos," *Aviation Week Intelligence Network*, p. 4, 2015.

[3] C. L. Symonds, *The Battle of Midway*. Oxford University Press, 2011.

[4] B. Clark, P. Haynes, B. McGrath, C. Hooper, J. Sloman, and T. A. Walton, *Restoring American Seapower: A New Fleet Architecture for the United States Navy*. Center for Strategic and Budgetary Assessments, 2017.

[5] B. Clark and J. Sloman, *Deploying Beyond Their Means: America's Navy and Marine Corps at a Tipping Point*. Center for Strategic and Budgetary Assessment, 2017.

[6] United States Department of Defense Modeling and Simulation Coordination Office, "Modeling and simulation glossary," Report.

[7] United States Department of Defense, "Department of defense dictionary of military and associated terms," Generic, 2015.

[8] R. C. Rubel, "Wargame rules as intellectual catalysts," *Military Operations Research Society: Phalanx*, vol. 50, no. 3, pp. 24–27, 2017.

[9] *Wikimedia image commons*, Generic.

[10] E. M. Young, *F4F Wildcat vs A6M Zero-sen: Pacific Theather 1942*. Osprey, 2013.

[11] N. Doerry and H. Fireman, *Fleet capabilities based assessment (cba)*, Computer Program, 2009.

[12] Office of Aerospace Studies, "Analysis of alternatives (aoa) handbook," Tech. Rep., 2013.

[13] R. E. Gilbert, "Strategic implications of us fighter force reductions," Thesis, 2011.

[14] B. McCue, "Instability and granularity in lanchester battles," *Military Operations Research Society: Phalanx*, vol. 49, no. 3, 2016.

[15] B. Caldwell, J. Hatman, S. Parry, A. Washburn, and M. Youngren, *Aggregated Combat Models*. Naval Postgraduate School, 2000.

[16] L. Low, *Anatomy of a Combat Model*. 1995.

[17] RAND Corporation, *Lanchester equations and scoring systems*, Web Page, 2017.

[18] T. Rogoway, "The amazing saga of how israel turned its f-15s into multi-role bombers," *Foxtrot Alpha*, 2015.

[19] D. Daberkow and D. Mavris, *An investigation of metamodeling techniques for complex systems design*, Conference Paper, 2002.

[20] A. J. Turner, "A methodology for the development of models for the simulation of non-observable systems," Thesis, 2014.

[21] A. P. Baker, D. N. Mavris, and D. P. Schrage, "Assessing the impact of mission requirements, vehicle attirubtes, technologies, and uncertainty in rotorcraft system design," *American Helicoper Society 58th Annual Forum*, 2002.

[22] D. N. Mavris and O. Bandte, "Economic uncertainty assessment using a combined design of experiments / monte carlo simulation approach with application to an hsct," *17th Annual Conference of the International Society of Parametric Analysts*, 1995.

[23] D. M. Pugh, "A validation assessment of the storm air-to-air prototype algorithm," Thesis, 2000.

[24] C. N. Seymour, "Capturing the full potential of the synthetic theater operations research model," Thesis, 2014.

[25] E. Hlywa, *Salvo model for anti-surface warfare study*, Web Page.

[26] J. Lehman, *The u.s. navy must be everywhere at once*, Newspaper Article, 2017.

[27] C. Goddard, H. Fireman, and C. Deegan, "A question of cost," *Armed Forces Journal*, 2007.

[28] United States Government Accountability Office, "Defense acquisitions: many analyses of alternatives have not provided a robust assessment of weapon system options," Government Document, 2009.

[29] J. P. Olivier, S. Balestrini-Robinson, and S. Briceno, "Approach to capability-based systtems-of-systems framework in support of naval ship design," in *IEEE International Systems Conference*.

[30] D. Cherry, "Making jcids work for the warfighter," Thesis, 2010.

[31] R. Babbage, *Countering China's Adventurism in the South China Sea*. Center for Strategic and Budgetary Assessments, 2016.

[32] B. Clark and J. Sloman, *Advancing Beyond the Beach: Amphibious Operations in an Era of Precision Weapons*. Center for Strategic and Budgetary Assessments, 2016.

[33] D. S. Alberts, J. J. Garstka, and F. P. Stein, *Network Centric Warfare: Developing and Leveraging Information Superiority*, 2nd. CCRP, 1999.

[34] D. J. Singer, N. Doerry, and M. E. Buckley, "What is set-based design," *Naval Engineers Journal*, vol. 121, no. 4, pp. 31–44, 2009.

[35] P. E. Pournelle, "Designing wargames for the analytic purpose," *Military Operations Research Society: Phalanx*, vol. 50, no. 2, pp. 48–53, 2017.

[36] J. R. Mumaw, M. B. Gailey, and D. A. Garber, "Physics-based modeling to reduce extensive full-scale testing," *Johns Hopkins APL Technical Digest*, vol. 33, no. 4, pp. 256–261, 2017.

[37] D. Locatelli, B. Riggins, J. A. Schetz, R. K. Kapania, B. Robic, C. Leenaert, and T. Poquet, *Aircraft conceptual design: tools evaluation*, Conference Paper, 2014.

[38] G. N. Vanderplaats, *Multidisciplinary Design Optimization*. 2007.

[39] W. E. I. Chen, J. K. Allen, D. N. Mavris, and F. Mistree, "A concept exploration method for determining robust top-level specifications," *Engineering Optimization*, vol. 26, no. 2, pp. 137–158, 1996.

[40] R. H. Myers, D. C. Montgomery, and C. M. Anderson-Cook, *Response Surface Methodology*. Hoboken, NJ: John Wiley and Sons, 2009.

[41] D. Daberkow and D. Mavris, *New approaches to conceptual and preliminary aircraft design: a comparative assessment of a neural network formulation and a response surface methodology*, Conference Paper, 1998.

[42] W. L. Perry, R. W. Button, J. Bracken, T. Sullivan, and J. Mitchell, "Measures of effectiveness for the information-age navy: the effects of network-centric operations on combat outcomes," RAND Corporation, Report, 2002.

[43] N. Sproles, "Formulating measures of effectiveness," *Systems Engineering*, vol. 5, no. 4, 2002.

[44] R. Lara-Cabrera, C. Cotta, and A. J. Fernandez-Leiva, "A review of computational intelligence in rts games," *IEEE*, 2013.

[45] J. M. Surman, R. G. Hilman, and E. Santos, "Adversarial inferencing for generating dynamic adversary behavior," *Proceedings of SPIE*, vol. 5091, 2003.

[46] S. Bakkes, C. Tien Tan, and Y. Pisan, *Personalized gaming: a motivation and overview of literature*, Conference Paper, 2012.

[47] I. Sakellariou, *Agent based modeling and simulation using state machines*, Conference Paper, 2012.

[48] S. Banisch, *Markov Chain Aggregation for Agent-Based Models*, ser. Understanding Complex Systems. New York: Springer, 2016.

[49] D. P. Schrage and D. N. Mavris, *Technology for affordability - how to define, measure, evaluate, and implement it?* Conference Paper, 1994.

[50] P. Charles and P. Turner, "Capabilities-based acquisition... from theory to reality," *CHIPS*, 2004.

[51] United States Department of Defense, *Manual for the Operation of the Joint Capabilities Integration and Development System (JCIDS)*. 2015.

[52] J. R. Chiles, "F-22 raptors uncaged," *Air and Space Magazine*, 2016.

[53] C. A. Baker, L. T. Watson, B. Grossman, and W. H. Mason, "Parallel global aircraft configuration design space exploration,"

[54] S. Ahmed, K. M. Wallace, and L. T. Blessing, "Understanding the differences between how novice and experienced designers approach design tasks," *Research in Engineering Design*, vol. 14, no. 1, pp. 1–11, 2003.

[55] K. Griendling, "Architect: the architecture-based technology evaluation and capability tradeoff method," Thesis, 2011.

[56] T. Revello, R. McCartney, and E. Santos, "Multiple strategy generation for wargaming," *Proceedings of SPIE*, vol. 5423, 2004.

[57] R. H. Kewley and M. J. Embrechts, "Computational military tactical planning system," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 32, no. 2, pp. 161–171, 2001.

[58] ——, "A multiagent system for tactical control of automated forces," *The International C2 Journal*, vol. 2, no. 2, 2008.

[59] B. Stillman, V. Yakhnis, and M. McCrabb, "Lg wargaming tool for effect-based operations," *Proceedings of SPIE*, vol. 4716, 2002.

[60] P. Bracken and M. Shubik, "War gaming in the information age: theory and purpose," *Naval War College Review*, vol. 54, no. 2, pp. 47–60, 2001.

[61] M. Pachter, E. Garcia, and D. W. Casbeer, *Active target defense differential game*, Conference Paper, 2014.

[62] E. Garcia, D. W. Casbeer, and M. Pachter, "Cooperative strategies for optimal aircraft defense from an attacking missile," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 8, pp. 1510–1520, 2015.

[63] T. Ender, R. F. Leurck, B. Weaver, P. Miceli, W. D. Blair, P. West, and D. Mavris, "Systems-of-systems analysis of ballistic missile defense architecture effectiveness through surrogate modeling and simulation," *IEEE Systems Journal*, vol. 4, no. 2, pp. 156–166, 2010.

[64] Force Structure, Resources, and Assessments Directorate (JSC J-8), *Capabilities-Based Assessment (CBA) User's Guide*. 2009.

[65] S. L. Caudle, "Homeland security capabilities-based planning: lessons from the defense community," *Homeland Security Affairs*, vol. 1, no. 2, 2005.

[66] J. M. Macuga, "The dhs joint requirements council: since 2014," *Military Operations Research Society: Phalanx*, vol. 50, no. 3, pp. 21–23, 2017.

[67] L. Finkelstein and A. Finkelstein, "Review of design methodology," *IEEE Proceedings*, vol. 130, no. 4, 1983.

[68] D. N. Mavris, *Ae6373: advanced design methods*, Personal Communication, 2013.

[69] United States Air Force, *Early Systems Engineering Guidebook Version 1.0*. 2008, (SAF/AQ).

[70] National Defense Industry Association, "Top 5 systems engineering issues with dod and defense industry," Report, 2006.

[71] United States Joint Chiefs of Staff, "Chairman of the joint chiefs of staff instruction cjcsi 3170.01 i," Government Document, 2015.

[72] P. K. Morrow, "Teaching note: analysis of alternatives," 2011.

[73] M. Schwartz, *Defense Acquisitions: How DoD Acquires Weapon Systems and Recent Efforts to Reform the Process*. Congressional Research Service, 2014.

[74] United States Joint Chiefs of Staff, "Chairman of the joint chiefs of staff instruction cjcsi 3010.02 d," Government Document, 2013.

[75] United States Department of Defense, *Defense Acquisition Guidebook*. 2013.

[76] United States Government Accountability Office, "Defense acquistions: fundamental changs are needed to improve weapon program outcomes," Report, 2008.

[77] F. Phillips and R. Srivastava, "Committed costs vs. uncertainty in new product development," *IC2 Institute at the University of Texas at Austin*, 1993.

[78] United States Congress, "Weapon systems acquisition reform act of 2009," Legal Rule or Regulation, 2009.

[79] M. Lush, "Implementation of weapon systems acquisition reform act (wsara) of 2009," Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics, Report, 2009.

[80] Defense Acquisition University, "Glossary of defense acquisition acronyms and terms," Report, 2016.

[81] MITRE Corporation, "Performing analyses of alternatives," Report, 2013.

[82] G. Norris and A. Butler, "F-35 flies against f-16 in basic fighter maneuvers," *Aviation Week*, 2015.

[83] C. A. Posey, "The guard at nato's northern gate," *Air and Space*, vol. 31, no. 5, pp. 48–55, 2016.

[84] K. Gray, "The last fighter pilot," *Popular Science*, vol. 288, no. 1, pp. 52–59, 2015.

[85] United States Government Accountability Office, "Defense acquisitions: dod's requirements determination process has not been effective in prioritizing joint capabilities," Report, 2008.

[86] United States Department of Defense, "Red flag fact sheet," Tech. Rep.

[87] ——, "Rimpac 2014," Government Document, 2014.

[88] M. Ponsen, P. Spronck, H. Muñoz-Avila, and D. W. Aha, "Knowledge acquisition for adaptive game ai," *Science of Computer Programming*, vol. 67, no. 1, pp. 59–75, 2007.

[89] J. E. Laird and M. van Lent, "Human-level ai's killer application: interactive computer games," *AI Magazine*, vol. 22, no. 2, pp. 15–25, 2001.

[90] A. Dahlbom and L. Niklasson, "Goal-directed hierarchical dynamic scripting for rts games," *American Association for Artificial Intelligence*, 2006.

[91] U. Wilensky, *Netlogo*, Computer Program, 1999.

[92] ——, *Netlogo wolf sheep predation model*, Aggregated Database, 1997.

[93] R. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton: Princeton University Press, 1961.

[94] P. D. Clive, J. A. Johnson, M. J. Moss, J. M. Zeh, B. M. Birkmire, and D. D. Hodson, *Advanced framework for simulation, integration, and modeling*, Conference Paper, 2015.

[95] R. C. Rubel, "The epistemology of war gaming," *Naval War College Review*, vol. 59, no. 2, pp. 108–128, 2006.

[96] M. C. Jr., "Toward a history-based doctrine for wagaming," *Aerospace Power Journal*, 2000.

[97] J. Markley, *United states army war college: strategic wargaming series*, Personal Communication.

[98] M. Albrecht, "Introduction to discrete event simulation," 2010.

[99] W. C. Baldwin, B. Sauser, and R. Cloutier, "Simulation approaches for system of systems: events-based versus agent based modeling," *Procedia Computer Science*, vol. 44, pp. 363–372, 2015.

[100] L. G. Halsey, D. Curran-Everett, S. L. Vowler, and G. B. Drummond, "The fickle p value generates irreproducible results," *Nature Methods*, vol. 12, no. 3, pp. 179–185, 2015.

[101] M. Krzywinski and N. Altman, "Power and sample size," *Nature Methods*, vol. 10, no. 12, pp. 1139–1140, 2013.

[102] United States Army War College Center for Strategic and Development Leadership, "Pacific partners wargame analysis," *Strategic Wargaming Series*, 2014.

[103] K. Engelmann and R. Erdman, "The center for army analysis "revolutionary war" wargame," *Military Operations Research Society: Phalanx*, vol. 50, no. 1, 2017.

[104] D. P. Raymer, *Aircraft Design: A Conceptual Approach*, ser. AIAA Education Series. AIAA, 2012.

[105] T. Sun and S. B. Griffith, *The Art of War*, ser. UNESCO collection of representative works : Chinese series. Oxford : Clarendon Press, 1964.

[106] *Merriam-Webster Dictionary*.

[107] R. Patel and T. Mark, "Performance evaluation of a reinforcement system for engagement scenarios - applications in ait to air combat," Thesis, 2011.

[108] E. Bonabeau, "Agent-based modeling: methods and techniques for simulation human systems," *Proceedings of the National Academy of Sciences*, vol. 99, no. 3, pp. 7280–7287, 2002.

[109] C. M. Macal and M. J. North, "Tutorial on agent-based modeling and simulation," *Proceedings of the 2005 Winter Simulation Conference*, 2005.

[110] M. Kress and I. Talmor, "A new look at the 3:1 rule of combat through markov stochastic lanchester models," *Journal of the Operations Research Society*, vol. 50, pp. 733–744, 1999.

[111] R. H. Arrowood, R. H. Donnan, J. M. Goodman, S. E. Gordon, C. G. Jenista, and A. D. Sanders, *Aiaa missile competition: georgia institute of technology high speed, advanced, long range air-to-air missile (halraam)*, Manuscript, 2014.

[112] J. Dykes, S. Gordon, J. Robinson, B. Laughlin, C. Domercant, and D. N. Mavris, "Assessing relative capabilities of missile defense architectures using advanced modeling and simulation techniques," *Naval Engineers Journal December 2014 Student Addendum*, vol. 126, no. 4, 2014.

[113] United States Army PEO STRI, "Onesaf," Tech. Rep.

[114] G. C. McIntosh, D. P. Galligan, M. A. Anderson, and M. K. Lauren, *Recent development in the mana agent-based model*, Personal Communication.

[115] Defense Systems Information Analysis Center, "Brawler: tactical air combat simulation," Personal Communication.

[116] M. Gunzinger and B. Clark, *Winning the Salvo Competition: Rebalancing American's Air and Missile Defenses*. Center for Strategic and Budgetary Assessments, 2016.

[117] S. Ray, "Beginners guide to learn dimension reduction techniques," *Analytics Vidhya*, 2015.

[118] S. Liao, "Us navy submarines are getting xbox 360 controllers to control their periscopes," *The Verge*, 2017.

[119] M. Brown, *Life as a us drone operator: 'it's like playing a video game for four years'*, Newspaper Article, 2013.

[120] L. Plunkett, "Ea created an ai that taught itself to play battlefield," *Kotaku*, 2018.

[121] P. Spronck, I. Sprinkhuizen-Kuyper, and E. Postma, *Enhancing the performance of dynamic scripting in computer games*, Conference Paper, 2004.

[122] M. Ponsen, "Improving adaptive game ai with evolutionary learning," Thesis, 2004.

[123] E. Gent, "Deepmind's new ai taught itself to be the world's greatest go player," *SingularityHub*, 2017.

[124] P. J. Haas and G. S. Shedler, "Stochastic petri net representation of discrete event simulations," *IEEE Transactions of Software Engineering*, vol. 15, no. 4, 1989.

[125] P. T. Biltgen, "A methodology for capability-based technology evaluation for systems-of-systems," Thesis, 2007.

[126] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE TRansactions of Systems Science and Cybernetics*, vol. 4, no. 2, 1968.

[127] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[128] V. Volovoi, *Abridged petri nets*, Electronic Article, 2013.

[129] W. G. Whittington, "Capability based assessment of joint urban operations using agent based simulation," Thesis, 2009.

[130] W. Burghout, "A note on the number of replication runs in stochastic traffic simulation models," 2004.

[131] M. D. Byrne, *How many times should a stochastic model be run? an approach based on confidence intervals*, Conference Paper, 2013.

[132] B. Efron, "Bootstrap methods: another look at the jackknife," *The Annals of Statistics*, vol. 7, no. 1, pp. 1–26, 1979.

[133] M. J. Steffens, "Trajectory-based launch vehicle performance analysis for design-space exploration in conceptual design," Thesis, 2016.

[134] K. Signh and M. Xie, "Bootstrap: a statistical method," 2008.

[135] A. A. Giunta, S. F. Wojtkiewicz Jr., and M. S. Eldred, *Overview of modern design of experiments methods for computational simulations*, Conference Paper, 2003.

[136] H. Chernoff, "Sequential design of experiments," *The Annals of Mathematical Statistics*, vol. 30, no. 3, pp. 755–770, 1959.

[137] C. Anderson-Cook, *Sequential design of experiments*, Government Document, 2017.

[138] J. J. Shah, S. V. Kulkarni, and N. Vargas-Hernandez, "Evaluation of idea generation methods for conceptual design: effectiveness metrics and design of experiments," *Jouranl of Mechanical Design*, vol. 122, pp. 377–384, 2000.

[139] K. A. Batterton and K. N. Hale, "The likert scale: what it is and how to use it," *Military Operations Research Society: Phalanx*, vol. 50, no. 2, pp. 32–39, 2017.

[140] Massachusetts Institute of Technology, "Design variable concepts," Online Multimedia, 2006.

[141] J. Boardman and B. Sauser, *Systems of systems - the meaning of of*, Conference Paper, 2006.

[142] N. Sproles, "Coming to grips with measures of effectiveness," 1999.

[143] G. Prothero, "Measure of effectiveness (moe)," *Defense Acquisition University*,

## VITA

Seth E. Gordon was born in Stanford, California and was mostly raised in Bethesda, Maryland. After graduating from Walt Whitman High School in 2009, he attended Princeton University and obtained his Bachelor's degree in Mechanical and Aerospace Engineering in August 2013. While at Princeton, he met his wife, Kirsten Parratt, and the two were wed in June 2015. Starting his graduate work (alongside Kirsten) at the Georgia Institute of Technology in 2013, Seth studied under Professor Dimitri Mavris in the Aerospace Systems Design Laboratory (ASDL). He completed his Master's degree in Aerospace Engineering in August 2015 before completing his Ph.D. in August 2018.