AI-enabled wargaming in the Military Decision Making Process

Peter J. Schwartz*[a], Daniel V. O'Neill[b], Meghan E. Bentz[b], Adam Brown[c], Brian S. Doyle[a], Olivia C. Liepa[a], Robert Lawrence[c], Richard D. Hull[c]
[a]The MITRE Corporation, Aberdeen, MD; [b]U.S. Army Combat Capabilities Development Command, Aberdeen Proving Ground, MD; [c]Parsons Corporation, Aberdeen Proving Ground, MD

ABSTRACT

During the Course of Action (COA) Analysis stage of the Military Decision Making Process (MDMP), staff members wargame the options of both friendly and enemy forces in an action-reaction-counteraction cycle to expose and address potential issues. This is currently a manual, subjective process, so many assumptions often go untested and only a very small number of alternative COAs may be considered. The final COA that is produced might miss opportunities or overlook risks. This challenge will only be exacerbated during Multi-Domain Operations (MDO), in which larger numbers of entities are expected to coordinate across domains to achieve converged effects within compressed timelines. This paper describes a prototype wargaming software support tool that leverages Artificial Intelligence (AI) to recommend COA improvements to commanders and staff. The tool's design accounts for operational realities including a lack of available AI training data, limited tactical computing resources, and a need for end user interaction throughout the COA Analysis process. Given initial COAs for friendly and enemy forces, the tool searches for improvements by repeatedly proposing changes to the friendly COA and running the Data Analysis and Visualization INfrastructure for C4ISR (DAVINCI) combat simulation to evaluate them. Runtime is managed by carefully restricting the search space of the AI to only consider doctrinally relevant changes to the COA. The system architecture is designed to separate the AI, the simulation, and the user interface, simplifying continued experimentation and enhancements. The design of the AI-enabled wargaming tool is presented along with initial results.

Keywords: Artificial Intelligence, Military Decision Making Process, simulation, Multi-Domain Operations

1. INTRODUCTION

The application of artificial intelligence (AI) to the Army's Military Decision Making Process (MDMP) has tremendous potential to support the commander's ability to plan for battlespaces that are becoming both hyper-contested and exponentially more complex. With the advent of Multi-Domain Operations (MDO), the need has never been greater for our command and control (C2) systems to harmonize disparate situational awareness insights to foster situational understanding. Near-peer adversaries, with modern technology and a focus on Anti-Access Area Denial (A2AD) capabilities, present new challenges for Joint and coalition forces. To meet these new challenges in the 21st century, the rapid development of multiple friendly courses of action (COAs) in a constantly changing battlespace becomes important, and finding the friendly COAs that perform best becomes critical. The COA Analysis phase of the MDMP, along with existing planning tools in the Army's Command Post Computing Environment (CPCE), provide a fertile test and experimentation proving ground for the development and evaluation of such capabilities. This paper explores how a new AI-based tool, the Artificial Intelligence COA Recommender (AICR), can help commanders and their staff quickly optimize a friendly COA to improve likely mission outcomes given current battlefield conditions.

This introduction examines the problem spaces addressed by AICR through the application of artificial intelligence: MDO, the Army's MDMP, and the COA Analysis phase of the MDMP. Section 2 illustrates some of the foundational building blocks being used to build AI-enabled wargaming for the MDMP in related work. Section 3 defines the problem our project attempts to solve along with details about our solution's architecture, user interface, use of combat modeling and simulation, and the employment of AI principles and techniques to tune friendly COAs. Section 4 describes the early experimental results collected to date. Section 5 concludes with lessons learned and possible avenues for future development and effort expansion.

*pschwartz@mitre.org; phone 1 410 272-5051; fax 1 410 272-6080; mitre.org

Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II, edited by Tien Pham, Latasha Solomon, Katie Rainey, Proc. of SPIE Vol. 11413, 114130H

1.1 Multi-Domain Operations

The emerging MDO environment is forcing the U.S. military to compete in highly contested spaces, where battlefield conditions and situations are rapidly evolving. The domains of Space, Air, Cyber, Land, and Maritime all must be considered holistically to identify and mitigate the evolving A2AD systems that near-peer adversaries use to induce layered stand-off. To address the military problems posed by an MDO environment, the U.S. military, its allies and partners must (1) effectively compete across domains, (2) penetrate enemy A2AD systems to enable freedom of strategic and operational maneuver, (3) dis-integrate A2AD systems to enable freedom of operational and tactical maneuver, (4) exploit maneuver opportunities to defeat the enemy, and (5) return to a state of competition where consolidated gains can be sustained.[1]

To achieve these objectives the U.S. military must employ a calibrated force posture with multi-domain formations that can quickly and intelligently converge to achieve and maintain overmatch. "[T]he application of [a] calibrated force posture positions the right mix of ready forces and capabilities [that] can rapidly transition to combat operations."[1] "[Using multi-domain formations] [t]he most important non-materiel contributors to resilience are flexible planning that account for enemy actions, the ability to reorganize formations in conflict, leaders and staff capable of operating in accord with intent, and small, dispersed, cross-trained headquarters. . . . [while] convergence complicates the enemy's attempts to conceal and defend its center of gravity by providing the Joint Force with multiple options for attacking the enemy's vulnerabilities at decisive spaces."[1] These demands require that we compress the time it takes to observe, orient, decide and act against near-peer adversaries despite an increasingly complex, dynamic environment.

1.2 Military Decision Making Process

The current demands of MDO require new capabilities to better support commanders and staff as they execute the MDMP:

> The MDMP is an iterative planning methodology that integrates the activities of the commander, staff, subordinate headquarters, and other partners to understand the situation and mission, develop and compare courses of action (COAs), decide on a COA that best accomplishes the mission, and produce an operation plan or order for execution. The MDMP helps leaders apply thoroughness, clarity, sound judgment, logic, and professional knowledge to understand situations, develop options to solve problems, and reach decisions. The MDMP is a process that helps commanders, staffs, and others think critically and creatively while planning.[2]

The MDMP has seven steps (Figure 1).[3] These steps help the commander and planning staff frame the operational environment, frame the problem space, and develop a plan to get from a current state to a desired state. Upon receipt of a mission from a higher echelon (Step 1), units using MDMP perform Mission Analysis (Step 2) to frame the problem space and develop perceived enemy COAs based on available intelligence, surveillance, and reconnaissance (ISR) information. This picture of how the enemy has composed themselves on the battlefield and perceiving what they might do is refreshed as intelligence preparation of the battlefield (IPB) updates a commander's understanding of the adversary.

Figure 1. The Military Decision-Making Process (MDMP) contains seven steps.

Once an enemy COA has been developed, friendly COAs are developed (Step 3) to achieve the desired intermediate and end-states specified by mission parameters. These friendly COAs are then wargamed against an enemy COA (Step 4) to assess how performant that friendly COA might be. Based on the results of wargaming, Friendly COAs are then compared against each other (Step 5), the commander approves a COA (Step 6), and orders are produced and disseminated to communicate the approved COA to friendly forces (Step 7).

COA Development and COA Analysis are critical but time- intensive for planning staff, so historically only one large friendly COA is generated to address all enemy COAs (starting with the most likely enemy COA, and then the most dangerous).[2] The Center for Army Lessons Learned notes that the more brainstorming that is done to find the most performant friendly COA, the better.

1.3 Course of Action Analysis

AICR addresses the fourth step of the MDMP, COA Analysis, where mission planners wargame COAs to analyze and refine the details of each one:

> COA analysis enables commanders and staffs to identify difficulties or coordination problems as well as probable consequences of planned actions for each COA being considered. It helps them think through the tentative plan. COA analysis may require commanders and staffs to revisit parts of a COA as discrepancies arise. COA analysis not only appraises the quality of each COA, but it also uncovers potential execution problems, decisions, and contingencies. In addition, COA analysis influences how commanders and staffs understand a problem and may require the planning process to restart.[2]

Army doctrine further decomposes the COA Analysis step into a process of several sub-steps (Figure 2).

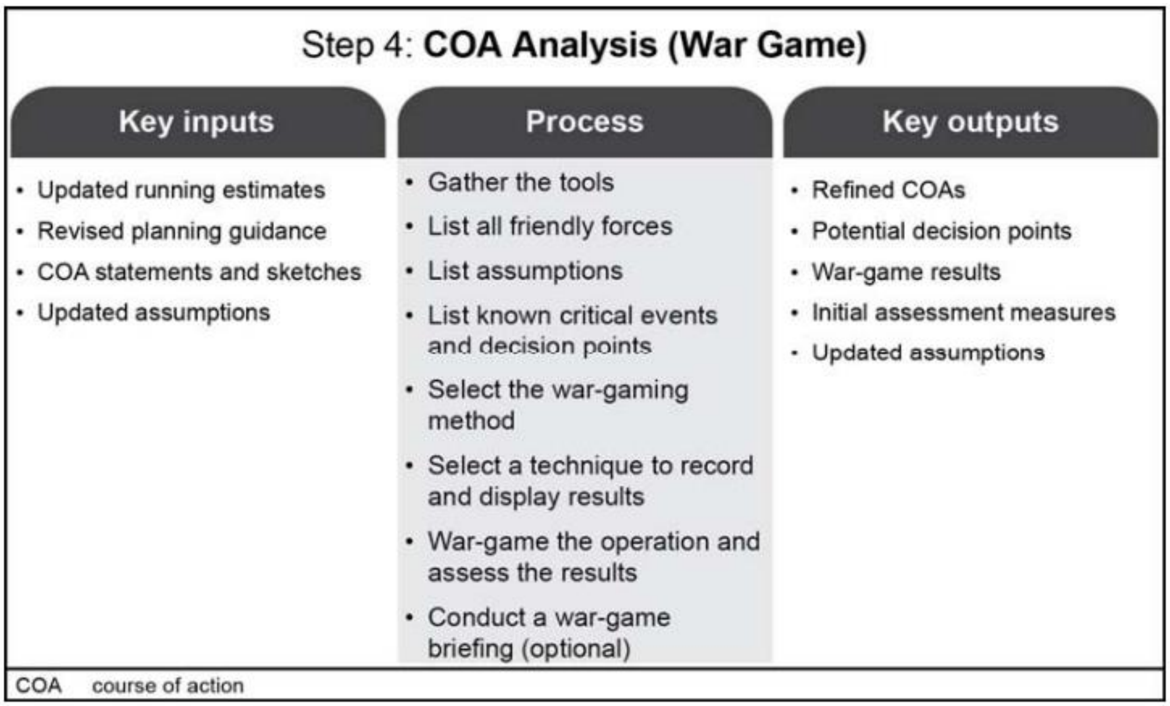

Figure 2. Inputs, process, and outputs of the COA Analysis (wargame) step of the MDMP.[3]

COA Analysis used to be a paper exercise, with relative combat power calculated manually. Today, COA Analysis can be performed digitally, within the commander's Common Operational Picture (COP), and with the benefit of data provided by other C2 systems. In the Planning context of CPCE, COA Analysis tools use modeling and simulation engines to interpret CPCE-developed COAs (units that have task and purpose assigned to them through drawn tactical graphics) and can wargame an authored friendly COA against an authored enemy COA. While the streamlining of COA development and analysis processes has benefited greatly from being integrated with the commander's COP, the refinement of the COA is still a manual process.[4]

The Center for Army Lessons Learned issues guidance on how to best to manage the Action-Reaction-Counter Action Cycles that are central to the COA Analysis activity. The guiding principles articulated are[5]:

1. The recorder projects a synchronization matrix where all can see it, and fills it out as the wargame progresses.

2. The facilitator may wait until all actions-reactions-counteractions are complete before adjudicating outcomes, particularly combat outcomes.
3. The scribe records advantages, disadvantages, risks, opportunities, assumptions, RFI decisions, and areas or contingencies needing further study as they are identified.
4. Participants update their planning products for this COA as they progress through the wargame.

2. RELATED WORK

Given 1) the complexity of COA Analysis, especially in an MDO context, 2) the time pressure that commanders and staffs face when performing COA Analysis, and 3) the existing integration of digital COA Analysis support tools into the commander's COP, it is reasonable to believe that these COA Analysis support tools could be greatly enhanced through the application of AI.

2.1 AI game playing

The MDMP is analogous to complex strategy games. Assessments and decisions must be made under time constraints, with imperfect understanding of the opponent, and with second- and third-order effects of both sides' actions impossible for a human to completely predict. For strategy games, there are numerous examples of AI systems shown to achieve and/or exceed human performance. In chess, IBM's Deep Blue system defeated the reigning world champion chess player in a six-game match in 1997.[6] Pre-dating many of the successes in machine learning, it relied heavily on symbolic reasoning. In the board game Go, AlphaGo in 2015 and its more recent successors (AlphaGo Master, AlphaGo Zero, and AlphaZero) have achieved champion-level mastery.[7] For Dota 2, a vastly complex strategy game, an AI program has recently bested world champions.[8] Libratus, a poker-playing AI program, has beaten world-class players at Texas Hold 'Em (a more complex game of poker).[9] Pluribus is another poker-playing AI program that dominates elite professional players in a six-player game, which is a more complex game space than conventional two-player poker.[10] These examples underscore AI's ability to excel at complex strategy games. They also demonstrate the potential for AI to perform complex planning tasks with human-level performance in an adversarial, imperfect information environment.

2.2 AI mission planning

The Army uses modeling and simulation to help planners study and project likely outcomes of engagements between opposing forces. Applying AI to these wargames to help planners navigate complex decision spaces and select optimal courses of action is logical and attainable.

The Army has attempted to apply AI to support mission planning in the past. The Crystal Ball and Blitzkrieg concept in Deep Green sought to actively monitor the commander's current situation and dynamically create COA options for the commander using a "futures graph" approach—potential future options would be managed by Crystal Ball through active wargaming using Blitzkrieg against current battlefield conditions.[11] The Future Combat System's Multicell & Dismount Command and Control (M&D C2) effort created the Battle Command Support Environment (BCSE), which was the first of its kind to use an automated expert system to attempt the fusion of individual Warfighter function information to feed decision support reasoning (examples include: consumable resource projections, automatic air and ground route generation).[12] As part of the Army's Command Post of the Future (CPOF), the DARPA-sponsored Personalized Assistant that Learns (PAL) attempted to alleviate the need for large command staffs by creating an "intelligent cognitive assistant" using machine learning that could learn from a CPOF user's interactions and make military decision-making more efficient.[13]

The Army is continuing to explore the application of AI to support mission planning. A recent Army publication[14] examines an AI approach to enable mission planning, command and control, and execution. The authors state that "AI-enabled systems and applications will support commands at all echelons in planning, preparing, executing, and assessing operations across the competition continuum" and that "the Army and Joint Force must leverage AI solutions to realize MDO." They delineate the key role of simulations in supplying data to drive the development of Machine Learning, and they describe using AI to generate courses of actions for assessment. The underlying technical thread in the paper is the constantly evolving windows of superiority and vulnerability and how AI can support their consideration. The authors stress that applications should integrate humans and AI in such a way that the strategic level is considered by humans, the tactical level is shaped by AI but arbitrated by humans, and sub-tactical tasks such as movement can be completely delegated to AI.

2.3 Automated Planning Framework

The Army Program Executive Office Command Control Communications Tactical (PEO C3T) oversees the development, acquisition, fielding, and support of the Army's tactical network, delivering the hardware and software required to provide an expeditionary and mobile tactical network. Project Manager Mission Command (PM MC), an organization in PEO C3T, oversees the development, deployment, and sustainment of Mission Command capabilities across the Army, supporting all of the warfighting functions.[15] PM MC is implementing the Army Common Operating Environment (COE) initiative, which is developing technologies, standards, and software to bring disparate Mission Command systems onto a common foundation intended to allow delivery of warfighting capabilities as applications. Several different Computing Environments (CEs) are part of the COE. The CPCE, the primary CE, is that central computing environment developed to support command posts and combat operations. It provides an infrastructure for convergence of current software Warfighter capabilities, as well as the integration of new capabilities.[16]

In 2018, Combat Capabilities Development Command (CCDC) transitioned into CPCE software known as the Automated Planning Framework (APF).[17] APF provides automated assistance to commanders and staff as they work through the MDMP. Support includes contextual assistance and synchronization of activities and coordination between Warfighters. COA Development and COA Analysis (wargaming) are important MDMP steps. To facilitate wargaming, CCDC included DAVINCI as part of the transitioned APF capability, giving planners quick insight into how effective their COAs are by having DAVINCI execute them and return the results. The Division Exercise Training and Review System (DXTRS), maintained by the Army's Program Executive Office Simulation, Training and Instrumentation (PEO STRI), is an instantiation of the DAVINCI Engine, which provides the core wargaming capabilities that enable evaluation of friendly COA performance against enemy COAs. APF enhanced the DAVINCI Engine as a software service within CPCE; this enabled the CPCE planning staff to execute multiple wargaming activities in parallel. DAVINCI Web application programming interfaces (APIs) were also created as a part of the APF effort. These APIs enable 3rd-party application control over management of multiple DAVINCI Engine instances, control over DAVINCI Engine wargames, and the specification and monitoring of COA analytics that are generated from each wargame run.

Knowledge of DAVINCI capabilities, how to manipulate DAVINCI from an external program, and how to build software for the CPCE are all being leveraged to benefit the effort to apply AI to DAVINCI to optimize COAs. DAVINCI will be described in greater detail in section 3.4.

3. TECHNICAL APPROACH

Given the demands of MDO, the current state of AI, and the prior integration of DAVINCI into CPCE, we are exploring the development of a wargaming tool that combines the combat simulation engine of DXTRS and an AI engine to support the MDMP COA Analysis step. Several guiding principles have shaped the design of AICR:

1. Do not attempt to replace the human decision-makers in the MDMP. Instead, aim to augment their experience and expertise.
2. Design the workflow to fit into the existing COA Analysis step of the MDMP.
3. Do not assume that high-performance computing or cloud resources will be available to users, since they might be operating in a tactical environment.

3.1 Problem definition

We frame the wargaming step of the MDMP as a combinatorial optimization problem. The user must manually define a COA for the enemy forces and an initial COA for the friendly forces. This can be done through a map-based user interface in DXTRS before using AICR. Given the enemy and friendly COAs as input, AICR must identify improvements to the friendly COA relative to the fixed enemy COA (Figure 3). Each friendly COA that AICR's AI proposes is evaluated by simulating the mission, collecting outcome metrics, and combining them according to a user-defined scoring function. Improvements to the friendly COA are presented to the user, and the user can then decide whether to incorporate them into the mission plan.

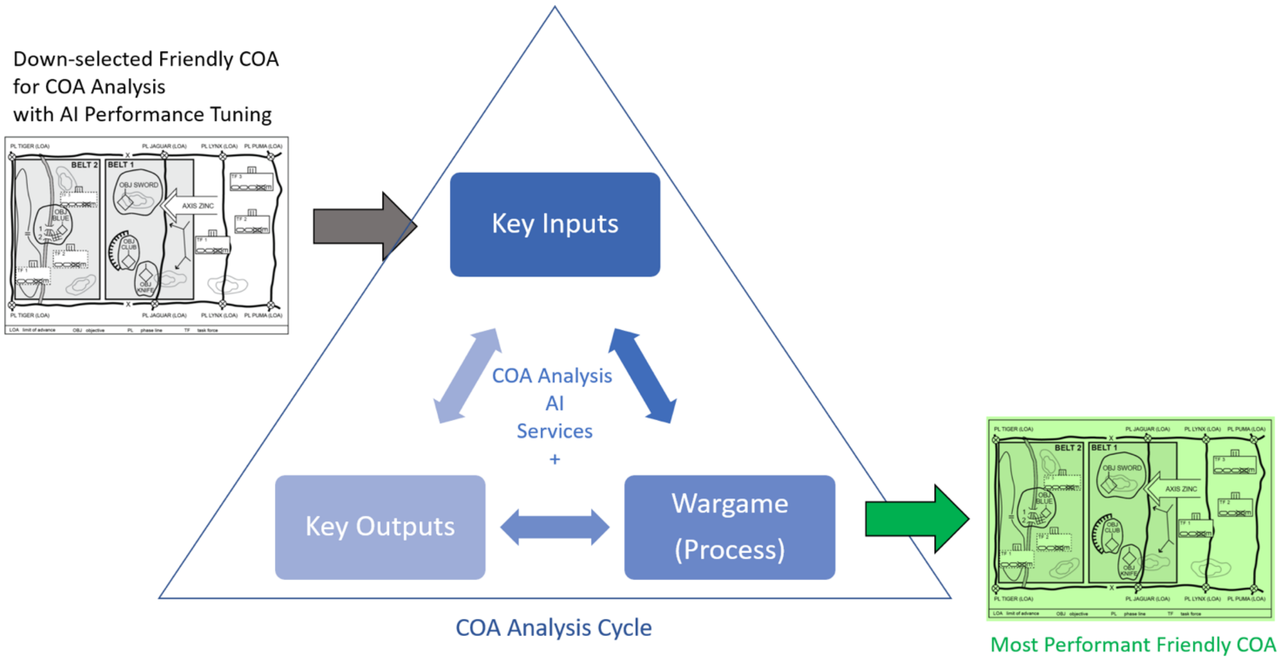

Figure 3. AI can help improve a given friendly COA by adjusting COA parameters over multiple wargame runs with the intent of "tuning" it to optimize mission outcomes.

The user must wait for AICR to return results, so runtime is a critical factor. As with any combinatorial optimization problem, the run time of a search algorithm is especially sensitive to the size of the search space. We therefore employ several methods to constrain the size of the search space:

1. We restrict AICR to only adjust the times at which tasks of friendly units begin within the execution matrix. Initial testing demonstrated that changes to the execution matrix could have a significant impact on the outcome of a mission. Future versions of the tool will expand the range of options for modifying the friendly COA, such as changing the unit tasking or the graphic control measures. Let t be the number of tasks, and let s be the number of unique start times when any task could start. Then the size of the search space is s^t.
2. We allow the user to select which units' tasks can be modified and which cannot. If a unit's tasks cannot be modified, then they are held constant during the search process, and therefore do not add to the size of the search space. This makes it possible for the user to focus AICR on specific parts of the larger mission. It reduces the size of the search space by reducing the number of tasks t that can be rescheduled.
3. We allow the user to set minimum and maximum start times of tasks. The start time of a task is only allowed to vary over its given range, not over the entire duration of the mission. This gives the user even more control over which alternative friendly COAs are considered. It reduces the size of the search space by reducing the number of unique start times s when a task can start.
4. We allow the user to specify the number of possible start times that AICR considers for any task. The possible start times are then distributed uniformly between the minimum and maximum start times. This makes it possible for the user to control the granularity of changes to alternative friendly COAs that the tool considers. It can reduce the size of the search space by reducing the number of unique start times s when a task can start.

3.2 Architecture

The system is designed using modern software development technologies and best practices (Figure 4). The modular design allows for distributed development across multiple development teams, as well as expansion or replacement of software components as the system grows and changes. Similarly, new AI components and simulators can be added or removed as needed. The three major components—the User Interface (UI) Web Application Server, the DXTRS Engine, and the AI Web Application Server—are outlined briefly below and then discussed in detail in the following sections.

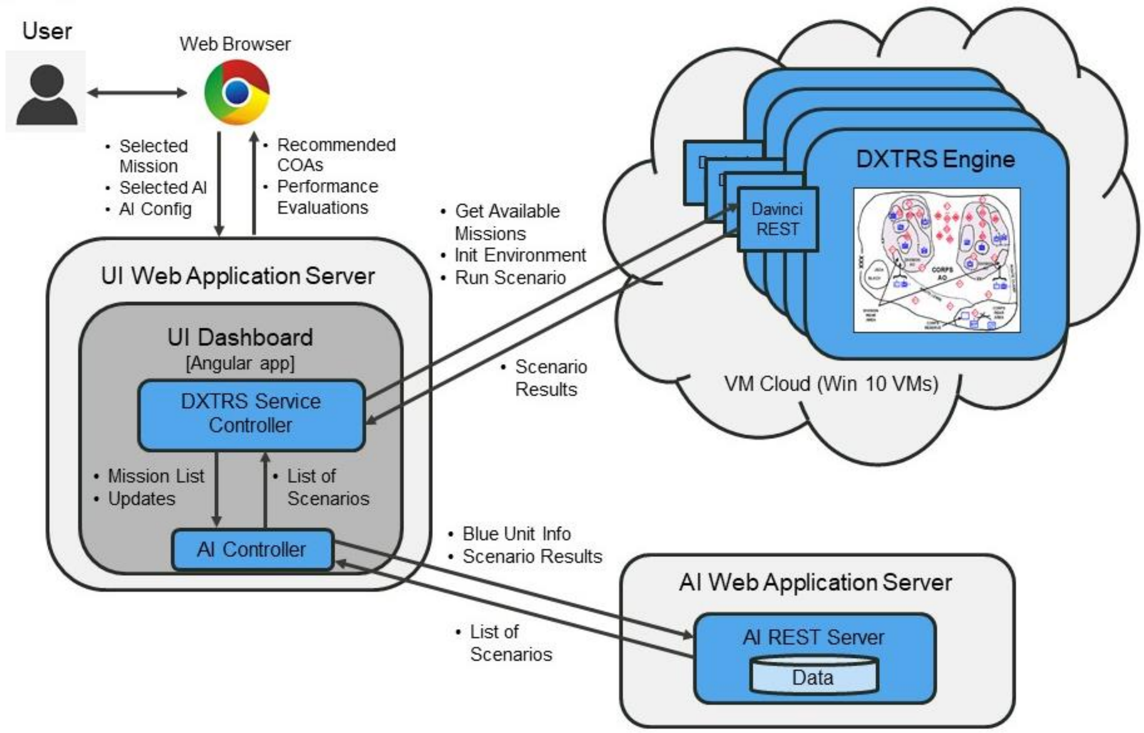

Figure 4. The system architecture is divided into three main components: the UI Web Application Server, the DXTRS Engine, and the AI Web Application Server.

The UI Web Application Server component provides a web-based user interface for users to access and control the system. It allows users to create new projects, configure those projects, run the projects through the simulation process, and view the results. It handles the communication between the other two components using provided APIs.

The DXTRS Engine component, which manages the combat simulation, is designed to allow distribution to a cloud environment and allow the scaling of the simulator for parallel execution. This distributed capability allows multiple simulations to run in parallel, which allows for faster results to be provided. The simulators are primarily installed on virtual machines (VMs) that each contain an instance of a DXTRS simulator. The simulator allows further opportunity for parallelization by running one or multiple simulations per virtual machine.

The AI Web Application Server component manages the search for improvements to the friendly COA. It uses the user configurations to decide which modifications to the friendly COA to test next and passes them to the DXTRS Engine to be simulated. The simulation results are then returned to the AI Web Application Server, so it can decide which friendly COAs to test in the next iteration of search. The AI Web Application Server is also responsible for persistent storage in a database. It provides an API for interaction with the user interface component.

3.3 User interface

AICR's user interface provides relevant customization of options, efficient workflows, and transparent results to commanders and staff. The user interface is currently designed to support developers. As the software matures and undergoes soldier evaluation, it will be modified to more accurately address the needs and workflows of mission planners. AICR's three primary pages, the Home page, the Configure page, and the Dashboard page, encourage intuitive user interaction.

The user is first brought to the Home page (Figure 5), which presents options to create a new COA configuration or to explore previously generated configurations. Each configuration includes a DXTRS scenario (including an enemy COA and an initial friendly COA, both defined manually within DXTRS) as well as a collection of settings that control the DXTRS Engine and the AI Web Application Server. The user can choose to explore the latest COA configuration or other previously generated configurations. From this page, the user can run, edit, duplicate, delete, and see the history of existing configurations. Within the history of a configuration, users can see the results of past runs of the COA through the tool. If the user chooses to create a new COA configuration, they will be navigated to the Configure page.

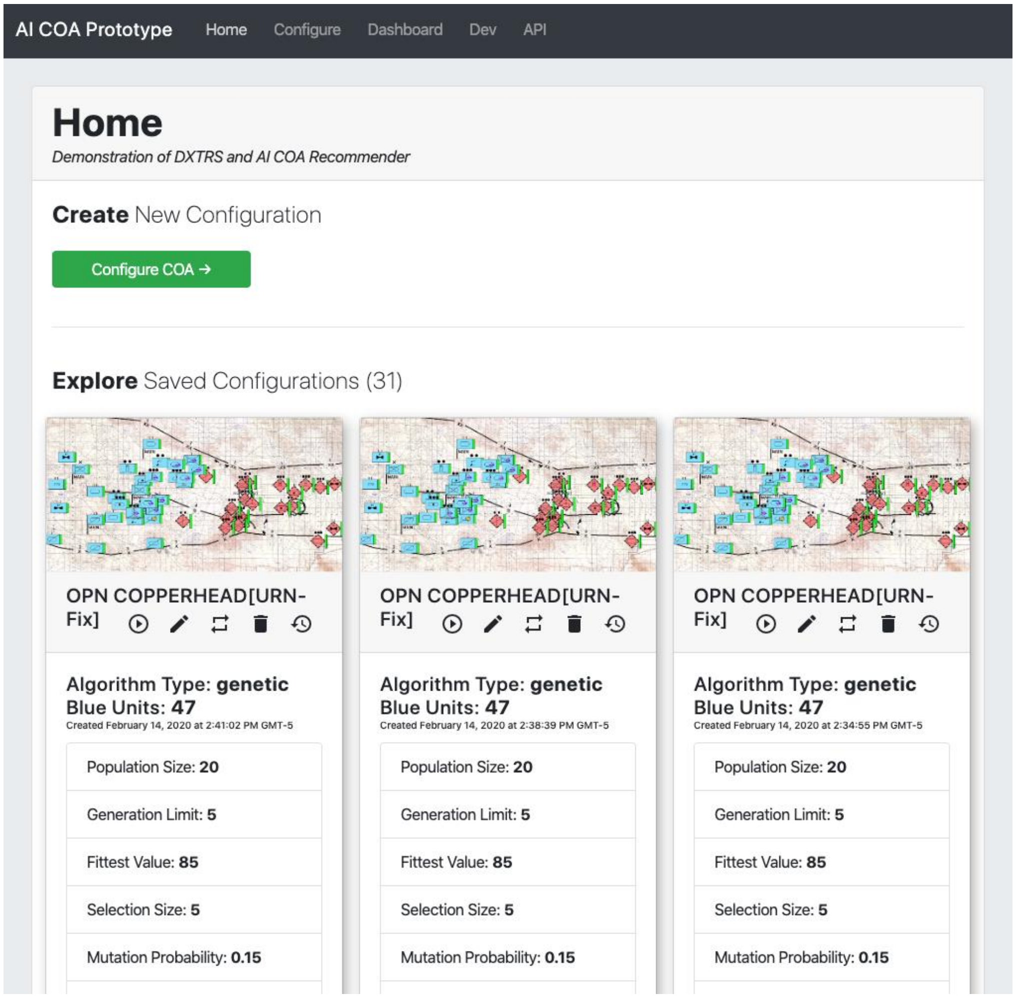

Figure 5. The Home page displays the latest configuration among other previously generated configurations. Users can run, edit, duplicate, delete, and see the history of each configuration.

Within the Configure page (Figure 6), users can generate an initial configuration file in the Mission Information section by selecting a COA to simulate. In the Algorithm Options section, users can choose to make advanced changes to the default settings of the AI. Users can also apply restrictions to changes that the AI can make. For example, users can choose to specify broad exclusions of certain units and tasks from manipulation. Alternatively, users can define more granular restrictions with the application of temporal constraints on task scheduling. In the DXTRS Options section, users can set the simulation speedup factor and specify the mission time at which the wargame ends. Permitting customization of these options can reduce the search space to improve runtime and give users more control over the balance between their own knowledge of the COA and the power of the AI. The user is presented with a short summary of the configuration in the Simulation Information panel, including an estimated runtime based on their chosen options.

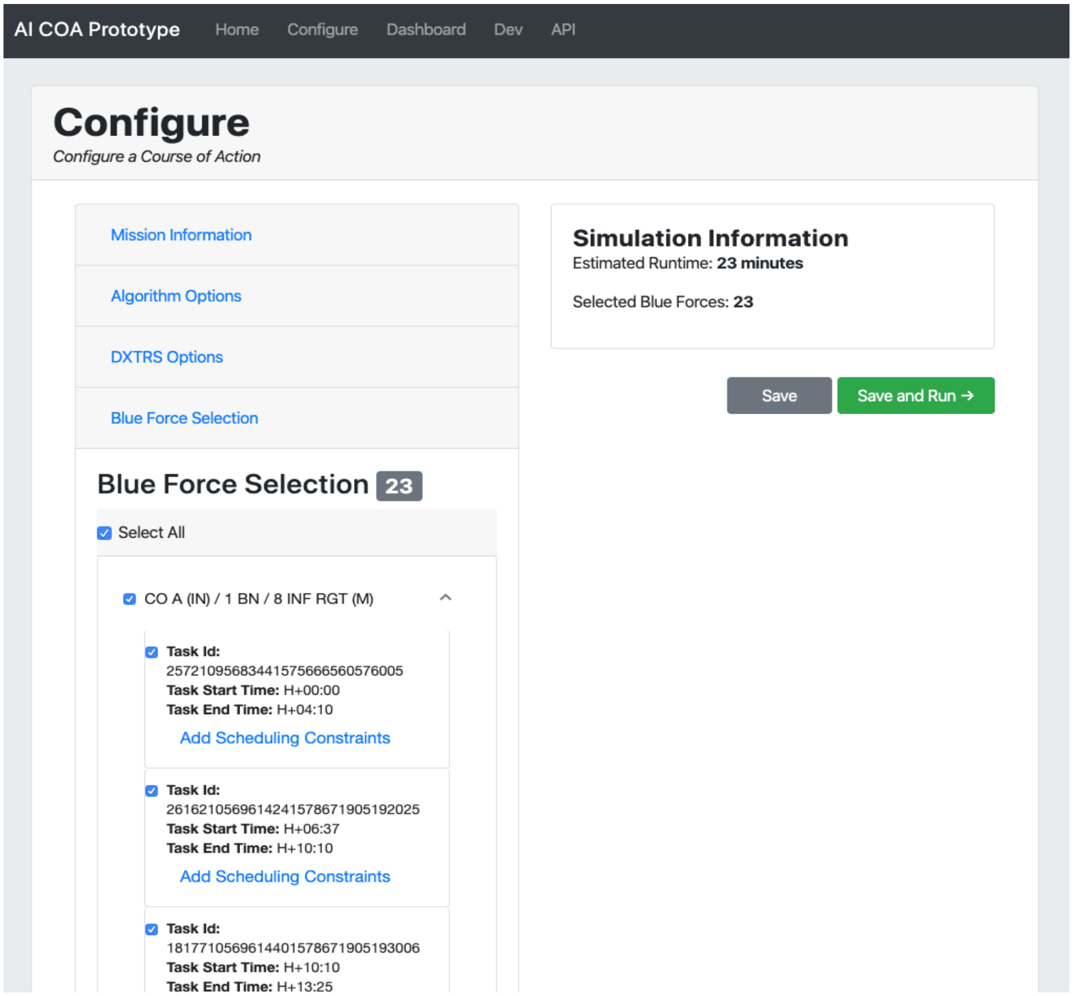

Figure 6. On the Configure page, users can make changes to the algorithm options, simulation options, and friendly unit and task selections.

Users are navigated to the Dashboard page (Figure 7) when they choose to save and run a COA configuration. The dashboard depicts the title, progress, and status of the COA simulation process in AICR. When the user starts the AI Web Application Server, the simulation process is illuminated by the display of algorithm results and data visualization in real time. Panels containing progress updates, graphs, recommendations, and summaries effectively translate AI results into a user-understandable context.

The Progress panel provides the user with information on the current stage of processing. A progress bar presents the percentage of process completion. This can help the user decide whether to terminate the process and inspect the new COAs recommended so far, or allow AICR to continue to search for more performant COAs.

The Live Metrics panel concisely displays COA improvement across simulation runs as a selection of graphs. Graph information includes the performance of the original friendly COA and the performance of each new friendly COA generated by the AI. In the case of the current genetic algorithm implementation, it also displays the best results from each population.

The Recommendations panel allows users to monitor improvements to the COA as they are simulated by the AI. As the algorithm runs, the panel updates with the best-performing simulations that resulted in the greatest improvements to the COA. Recommendations that out-perform the original COA are highlighted in green. Users can select a specific recommendation to see a short summary of the changes that were made to the COA by the AI within that run.

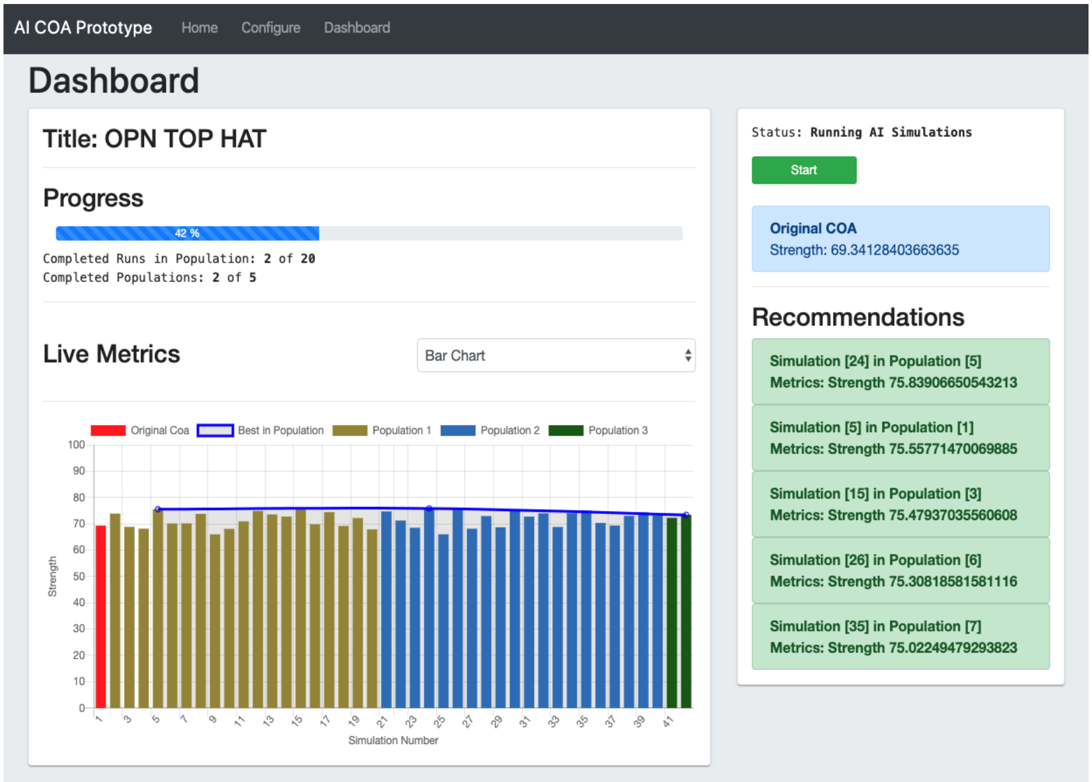

Figure 7. The Dashboard presents users with results in real time as they are returned from DXTRS. Users can view best-performing COAs in the Recommendations panel and view changes made by the AI to generate each result.

3.4 Combat simulation

AICR uses DAVINCI in its DXTRS instantiation as its combat simulation to evaluate changes to the user-defined COA that are proposed by the AI. DAVINCI is a low-overhead constructive simulation that supports the Army's CPCE for COA Development and Analysis. DXTRS also assists staff training at the crawl-walk stages and is a low-cost alternative to large-scale simulations for training at home stations and the Center of Excellence Schools.[18] DXTRS was born out of the Army Low Overhead Training Toolkit (ALOTT). The genesis of DXTRS and ALOTT were requirements for addressing a training gap between non-simulation unit exercises and large-scale constructive exercises. ALOTT sought training tools that were easy to use and reset, had low costs to prepare and support, provided rapid scenario generation, imported/used data from authoritative sources, and used a small computing footprint.

DXTRS (Figure 8) provides a robust COA Analysis capability with a small footprint. It is a deterministic, discrete-time simulation that can simulate a mission at rates much faster than real time on a standard laptop. Speedup is achieved by increasing the size of the simulation timestep, but at the possible risk of decreased fidelity. It takes a combined arms approach and supports all warfighting functions. It also provides levels of warfighting function automation—units perform their function automatically to standard, and/or can be manually directed by the operator; this lets the Training Director adapt the system to focus on the specific training objectives. DXTRS focuses on unit behavior based on each unit's composition and type (also known as an "aggregate" system) yet is also echelon agnostic so that it can model specific platform behaviors as required.

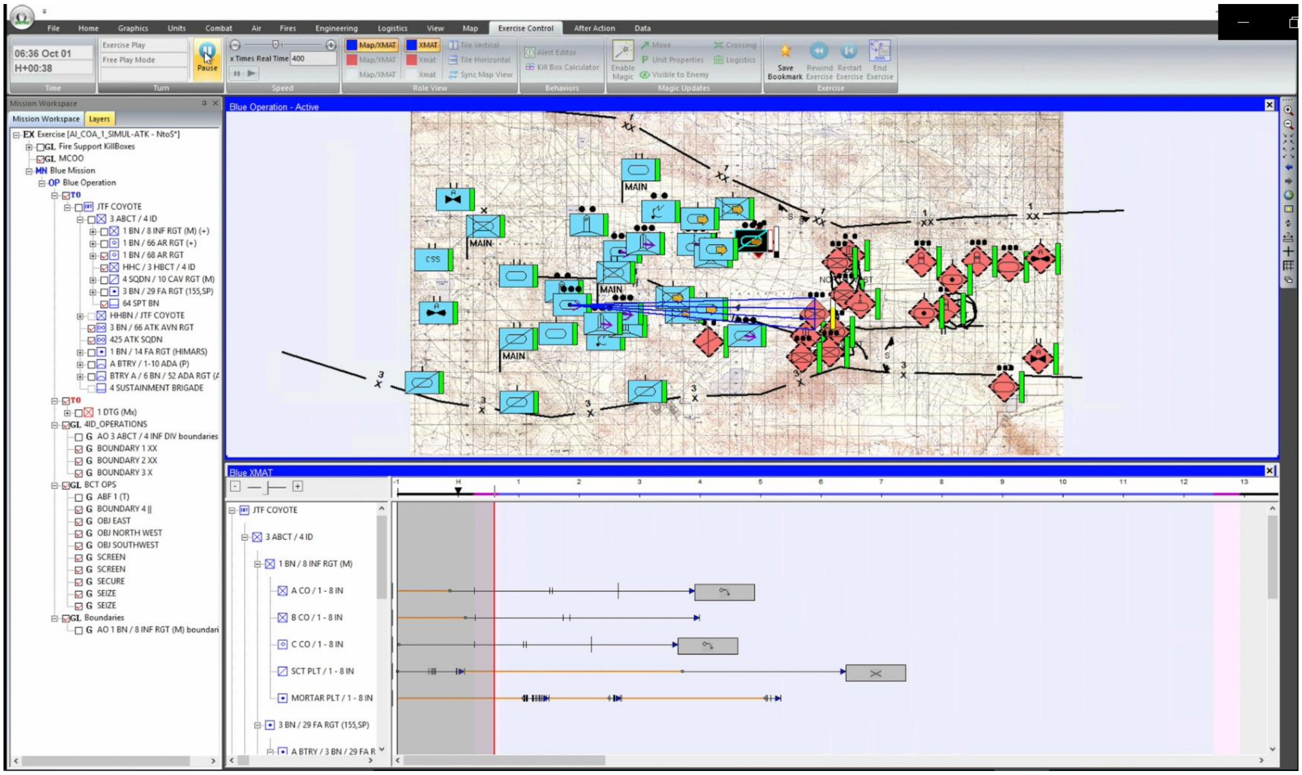

Figure 8. The DXTRS combat simulation allows a user to describe a mission scenario through a map-based graphical user interface and execution matrix timeline and then simulates the mission at rates much faster than real time.

DXTRS can[19]:

- Support single or multiple users with automated behavior processing using Division/higher or Brigade/lower models
- Support competitive Blue and Red players, supervised by a White Cell / Battle Captain, each with tailored views and system interactions
- Calculate resources, execute tasks, determine effects, and continuously re-populate the COP as it changes over time
- Have unit actions automatically modify Battlefield Synchronization and drive Logistics demands
- Simulate that situational understanding (SU) is earned (and lost) by ISR resources
- Exercise execution in real time or faster than real time

After DAVINCI simulates a friendly COA against an enemy COA, COA Analysis metrics include:

- Combat Power (aggregate strength of units, which is affected by attrition)
- Earned Situational Awareness (knowledge of location and status of enemy units)
- Unit Speed
- Ammunition Levels
- Fuel Levels

Using the COA Analysis metrics produced as a result of wargame execution, the AI can adjust and, over time, tune the friendly COA performance towards the desired end state the commander wants to achieve. As development continues, we plan to provide users with more options to specify which metrics or combinations of metrics the AI should attempt to optimize.

3.5 Artificial Intelligence

The AI Web Application Server component of AICR searches for improvements to the friendly COA using a genetic algorithm (GA).[20] We decided to begin with a GA because it is a well-known algorithm that is easy to implement. It is also easy to parallelize, which will help to decrease run time. Finally, the simple representation in a GA of a genome as a vector of parameters should make it easy to either expand the range of options AICR has for modifying the friendly COA or to replace the combat simulation with minimal impact to the AI. While newer forms of AI, such as Deep Reinforcement Learning[21], have recently grown in popularity, we decided to start with a simpler form of AI to develop AICR's architecture and performance testing environment. Once we establish a baseline with GA, we plan to explore more sophisticated types of AI.

In a GA, each solution in the search space is described as a genome that describes the parameters of that specific solution. A GA generates a population of individuals with random genomes and then searches for improvements over a series of generations. For each generation, each individual in the population is evaluated and given a fitness score. The fittest individuals are selected to survive and produce the next generation. To create an individual in the next generation, two survivors of the previous generation are selected as parents. A random crossover point is selected, and the parameters before the crossover point from one parent are appended with the parameters after the crossover point from the other parent. Then a subset of the parameters in the offspring are randomly mutated with new values to create the new genome. This process is repeated until the new generation is created, and the cycle begins again.

Within AICR, the genome of each individual is a vector of start times of the tasks that the tool can modify. The default population size is 20 and the default mutation rate is 15%, although the user has the ability to set these values manually. When the user starts the AI, AICR first simulates the user-defined friendly COA to establish a baseline score that the AI will attempt to improve. The GA then creates an initial population of 20 new COAs, each of which is a random variation on the user-defined COA within the parameters established by the user specifying which task start times can change and over what range. Each new COA is then sent to DAVINCI, which evaluates the COA and returns a score based on the simulated mission outcome.

To create the next generation of friendly COAs, a subset of elite survivors is selected from the current generation using a roulette wheel approach in which the likelihood of selection is related to its evaluation score. To encourage more rapid COA improvements in this time-sensitive application, the minimum score of the population is subtracted from the score of each individual, and the probability of an individual being selected is proportional to this difference. This gives better-performing COAs a much higher probability of being selected than lower-performing COAs. The default number of elite survivors is 5.

The elite survivors become members of the next generation and do not need to be re-evaluated in the simulation, helping to reduce runtime. They also serve as the parents of the new individuals in the next generation. Two unique elite survivors are selected with uniform probability to become the parents of the new individual. A random crossover point is selected, and the genomes of the two selected parent COAs are combined to create the genome of the new COA. 15% of the tasks in the new COA genome are then randomly selected and given new random start times within the user-defined constraints.

4. RESULTS

Testing is still very early at the time of writing, but it does indicate that AICR is capable of assisting COA Analysis. We asked a mission planning subject matter expert (SME) to design a scenario in DXTRS for testing purposes. In this scenario, dubbed "Operation Cottonmouth", the enemy force defends its position with a Motorized Infantry Battalion of BMP-2s (infantry fighting vehicles) positioned forward along a central corridor. Enemy dismounted forward reconnaissance elements are positioned on high ground. The Battalion is supported by a Field Artillery Battalion with two co-located counter-fire radars. An enemy tank company is positioned behind the Field Artillery Battalion. Air defense is provided by organic and attached BMP-2s with SA-24s (surface-to-air missiles). No air or aviation support is active for enemy forces.

For the friendly forces, an Armored Brigade Combat Team (ABCT) is tasked with destroying enemy forces. The ABCT attacks with an Armor Regiment Combined Arms Battalion (AR CAB) in the north and an Infantry Combined Arms Battalion (INF CAB) in the south, attacking frontally into the enemy. Counter-fire radar and unmanned aerial vehicles (UAVs) are committed forward to expedite destruction of enemy Field Artillery and Mortars. The ABCT ground maneuver advances and engages enemy forward defenses supported by indirect fire. No air or aviation support is active for friendly forces. Operation Cottonmouth contains 30 different tasks for friendly forces. The goal is to maximize the strength of friendly forces after they achieve their mission objectives, where a unit's strength is its combat power as a percentage of its maximum possible combat power.

Two versions of the friendly COA in Operation Cottonmouth were developed by our mission planning SME without the assistance of AI—a Baseline version and an Improved version. The Baseline COA represents what an inexperienced planner might construct. Friendly forces are not properly synchronized, putting some of them at increased risk. In DAVINCI, the Baseline COA ended with friendly forces at 75% strength. The Improved COA represents what a more seasoned planner might construct. Attack is timed to ensure that enemy observation and indirect fires have been largely neutralized prior to exposing friendly forces. The timing for 17 of the 30 tasks were changed between the two versions; the timing for the other 13 tasks did not change. In DAVINCI, the Improved COA ended with friendly forces at 85% strength. According to our mission planning SME, the difference in friendly forces strength between 75% and 85% is operationally very significant.

For testing, we wanted to see if AICR could modify the Baseline COA of Operation Cottonmouth to increase the strength of friendly forces at the end of the scenario from 75% closer to the 85% that our mission planning SME was able to achieve. The start times of the 13 tasks that did not change between the two versions of the friendly COA were held constant. For the 17 tasks that did change, we set the earliest and latest possible start times to 2 hours before and after the start times of those same tasks in the Improved COA. This helped put AICR in the general vicinity of the Improved COA, but still allowed significant opportunity for unsynchronized COAs that perform poorly. We set the number of different possible start times for each of these 17 tasks to 4, making the size of the search space 4^{17}. We configured the tool to search this space using a GA with the default values of a population size of 20, 5 elite survivors per generation, and a mutation rate of 15%.

Running on a Windows 10 laptop with 32 GB of RAM and an i7 Intel CPU at 2.60Ghz, each simulation of 40 hours of simulated mission time at 250x real time took about 10 minutes to complete. This means that the first generation, with 20 individual COAs to simulate, required about 3.5 hours to complete. Each generation after that, with 15 new individual COAs to simulate (because the other 5 were elite survivors from the previous generation), required about 2.5 hours to complete. The entire test of 5 generations took about 13.5 hours to complete.

Figure 9 shows the results of our test. The median score of each population is shown by the horizontal line in the middle of each box, and the mean of each population is shown by the X inside each box with a line connecting them. The mean population score increased steadily from 76.8% in Population 1 (consistent with the given Baseline COA) up to 81.8% in Population 3. After that, it appears to level off. This upward trend over generations demonstrates that the GA is working as intended. Furthermore, some friendly COAs generated by AICR exceed 85% as early as Population 2. This demonstrates that AICR can provide expert-level recommendations very quickly in this case.

5. CONCLUSIONS AND FUTURE WORK

Progress to date has demonstrated the potential viability of the AICR wargaming tool. The user can define a scenario and tell the AI which tasks it can adjust in the execution matrix. The genetic algorithm can search for changes in the execution matrix that improve mission outcomes. The DAVINCI combat simulation can quickly simulate different friendly COAs to inform the AI algorithm. The user interface can provide feedback to the user about which changes improve the friendly COA as AICR continues to run.

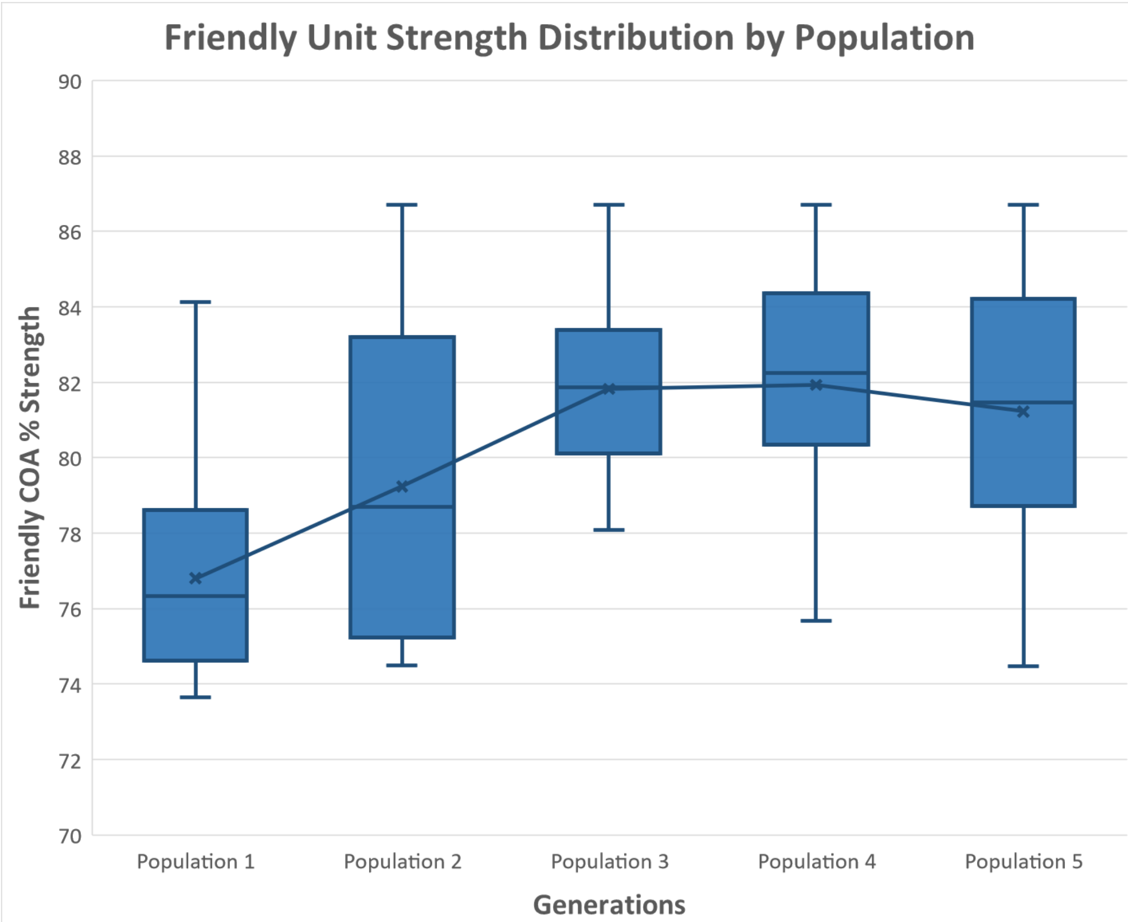

Figure 9. Early testing shows that AICR is capable of improving a given friendly COA within a few generations.

5.1 Lessons learned

Key lessons learned so far include:

- *Solicit feedback early and often*. We developed the user interface of our tool before we developed the AI. Through multiple demonstrations to different end user representatives, we were able to identify the features that are most important to our target users.
- *Design the tool around the users' needs*. Commanders and staff are highly trained and intelligent. Our user interface gives them the ability to focus the AI on regions of the search space that are relevant while holding other parts of the friendly COA constant. Understanding that our users will face significant time pressures, we designed the user interface to provide constant updates as the tool searches for improvements to the friendly COA. Users can see what changes have improved the friendly COA even as the AI continues to search.
- *Start simple*. A genetic algorithm is relatively easy to implement compared to other forms of AI. We found that it was sufficient to demonstrate the fact that AI is able to improve a given manually generated friendly COA. After that, it is simply a matter of improving the AI so it can find bigger improvements faster.
- *Design a flexible architecture*. Our modular architecture allows for rapid improvements. The AI, combat simulation, and user interface can all be updated independently. This means each major component can be updated without affecting the others. It also allows all of them to be developed simultaneously.

5.2 Future improvements

There are many opportunities to develop AICR further. To improve runtime performance, it should be relatively easy to parallelize the simulation runs with the existing genetic algorithm. A container architecture should further simplify the process of scaling up to leverage cloud resources. We can also speed up the tool with more advanced AI algorithms, such as particle swarm optimization (PSO)[22] or heuristic search based on guidance from military experts.

The biggest consumer of runtime in the current implementation is the combat simulation. Even though DAVINCI can run much faster than real time, a single simulation run of a mission can still take several minutes to complete. This means it could take hours for an AI algorithm to find improvements to the friendly COA, even with parallelization. If we could save the state of the simulation as it runs, it might be possible to start a later simulation partway through at the point where they start to diverge instead of all the way from the beginning of the mission. We could also use supervised learning to train a surrogate model offline that predicts the outcome of a scenario without simulating the details at each timestep. If prediction accuracy is an issue, we could use a multi-fidelity simulation approach that uses the surrogate model to guide the AI algorithm at first, but then runs the full simulation to refine its results.

In addition to speeding up the tool, there are also ways to improve AICR's usability. We could allow the user to specify temporal constraints between tasks; e.g., one task must finish before another can begin, or two tasks must start within 15 minutes of each other. This would give the user greater control over the search space, so the AI algorithm does not waste time considering COAs that are not consistent with the commander's intent. It would likely be easiest for the user to specify these temporal constraints through a graphical user interface that is similar to the familiar execution matrix that already exists in DAVINCI.

It would also be possible to allow the user to describe what outcomes they want to achieve by defining the objective function that the AI algorithm attempts to optimize. DAVINCI is already able to produce multiple metrics of both enemy and friendly forces at the end of a simulation run, so we could give the user the ability to assign weights to each metric and have the AI algorithm optimize the weighted sum. As the AI algorithm runs and the tool provides results of simulation runs to the user, there could be ways to allow the user to give real-time feedback, updating the objective function or temporal constraints without restarting the search process. AICR could provide additional feedback by looking for commonalities across simulation runs to inform the user about what types of changes make COAs perform better and what types of changes make them perform worse.

Finally, we can also extend the functionality of AICR. If we swap the friendly and enemy forces, AICR could be applied during the Mission Analysis step of the MDMP to help identify the most likely or most dangerous enemy COA. If we evaluate each friendly COA against multiple possible enemy COAs, the tool can find friendly COAs that are more robust. If we enable AICR to improve both the friendly and enemy COAs simultaneously, then it could mimic the action-reaction-counteraction wargaming cycle described in Army doctrine.

The current implementation of AICR only allows the AI to modify the timing of events. Future versions could allow the AI to explore other types of changes to a COA, including unit locations, tasking, and resource allocations. If AICR could be connected to real-time data sources of running estimates and mission progress, it could support not only mission planning but also mission execution. Extending or replacing the combat simulation with one that can model multi-domain operations would allow the tool to consider more complex COAs that leverage multi-domain convergence of effects.

5.3 Strategic impact

MDO is expected to significantly increase the speed and complexity of military operations. Near-peer adversaries are expected to employ A2AD capabilities to create strategic stand-off. U.S. forces and coalition partners will need to be able to effectively compete across domains, penetrate and dis-integrate enemy A2AD systems, exploit maneuver opportunities, and return to a state of competition. This will require developing and analyzing plans across Space, Air, Cyber, Land, and Maritime domains to create converged effects. The number of entities and interactions across the battlespace will create an overwhelming number of potential COAs—far more than any mission planning staff could consider without the assistance of advanced technology like AI.

AICR opens the door for computer-assisted COA Analysis in MDO. It leverages predictions from a combat simulation to quickly test many permutations of a COA to recommend potential improvements. The current prototype only employs a simple genetic algorithm to explore changes to the timing of tasks in the execution matrix using a ground combat simulation. The loosely coupled architecture, however, allows many opportunities to quickly improve the system. More advanced forms of AI can focus the search process on the most promising areas of improvement for the given COA. Extensions to the types of changes the AI can consider will allow the tool to explore COAs that span multiple domains. Enhancements to the combat simulation will allow the tool to predict the outcomes of multi-domain COAs faster and more accurately. These additional flexibilities, when implemented in a software environment where more simulation runs can be performed, should increase the tool's value, as it may be able to improve even a well-conceived COA in a complex battlespace by discovering hidden vulnerabilities. We expect that improvements to AICR such as these will allow it to become a very useful decision aid to military planners who must overcome the speed and complexity of MDO.

ACKNOWLEDGEMENTS

We would like to thank Brett Burland of the Mission Command Battle Lab, Mission Command Capability Integration Directorate, Combined Arms Center; Carl Wilk of the Command, Power, and Integration Directorate, C5ISR Center, Combat Capabilities Development Command; and Mark Curry of Parsons Corp. for their operational expertise and insights throughout the design and development of AICR. We would also like to thank Robert Amador of Parsons Corp. for his diligent work supporting the tool's software development.

This technical data deliverable was developed using contract funds under Basic Contract No. W56KGU-18-D-0004.

Approved for Public Release; Distribution Unlimited 20-0837

REFERENCES

[1] U.S. Army Training and Doctrine Command, "TRADOC Pamphlet 525-3-1: The U.S. Army in Multi-Domain Operations 2028," 16-20 (2018).
[2] Center for Army Lessons Learned, "Handbook No. 15-06: MDMP Lessons Learned and Best Practices," 7-39 (2015).
[3] Headquarters, Department of the Army, "Field Manual 6-0 Change No. 2," 9-3 – 9-26 (2016).
[4] Ballanco, E., "We Need an AI-Based Enemy Analysis Tool … Now!" War Room, 16 January 2019, <warroom.armywarcollege.edu/articles/enemy-analysis-tool-now/> (8 March 2020).
[5] Center for Army Lessons Learned, "Handbook No. 20-06: How to Master Wargaming: Commander and Staff Guide to Improving COA Analysis," 48 (2020).
[6] Greenemeier, L., "20 Years after Deep Blue: How AI Has Advanced Since Conquering Chess," Scientific American, (2017).
[7] Somers, J., "How the Artificial Intelligence Program AlphaZero Mastered Its Games," The New Yorker (2018).
[8] Albanesius, C., "Elon Musk's OpenAI Bot Beats Pro Dota 2 Player," PC Magazine (2017).
[9] Metz, C., "Inside Libratus, the Poker AI that Out-Bluffed the Best Humans," Wired Magazine (2017).
[10] Brown, N. and Sandholm, T., "Superhuman AI for Multiplayer Poker," Science Magazine (2019).
[11] "DARPA's Commander's Aid: From OODA to Deep Green," Defense Industry Daily, 3 June 2008, <https://www.defenseindustrydaily.com/darpa-from-ooda-to-deep-green-03497> (23 March 2020).
[12] Scrocca, J., Molz, M., Kott, A. "Collaborative Awareness: Experiments with Tools for Battle Command," 2006 CCRTS (2006).
[13] Sanders, R., Milligan, R., Roberts, K. "DARPA Command Post of the Future (CPOF) Enhanced with the Personalized Assistant that Learns (PAL) 2008 Demonstration: Final Report," U.S. Army TRADOC Battle Command Battle Laboratory – Leavenworth (2009).
[14] Taliaferro, A., Stump, E., Narayana, P., Kott, A., Foresta, J., Colegrove, S., and Burland B., "Discovery Enabler Concept of Operations - Artificial Intelligence: Reinforcement Learning to Enable Decision Overmatch," (2020).
[15] Program Executive Office Command Control Communications-Tactical, "Who We Are," <peoc3t.army.mil/c3t/who-we-are.php> (8 March 2020).

[16] Program Executive Office Command Control Communications-Tactical, “Project Manager Mission Command,” <peoc3t.army.mil/mc/> (8 March 2020).

[17] Jontz, S., “Rethinking Cognition,” Signal, 1 September 2017, <www.afcea.org/content/rethinking-cognition> (8 March 2020).

[18] “Enhanced Training: DXTRS Bridges the Training Gap,” Combined Arms Center – Training Newsletter, August Issue 1:2, <usacac.army.mil/sites/default/files/documents/cact/august%20newsletter.pdf> (23 March 2020).

[19] U.S. Army Program Executive Office for Simulation, Training and Instrumentation, “DXTRS: Division Exercise Training and Review System – The Low Overhead Brigade/Division Constructive Training Solution,” (2013).

[20] Mitchell, M., “An Introduction to Genetic Algorithms,” Cambridge, MA: MIT Press (1996).

[21] Francois-Lavet, V., Henderson, P., Islam, R., Bellemare, M., Pineau, J., “An Introduction to Deep Reinforcement Learning,” Foundations and Trends in Machine Learning, 11 (3-4): 219-354 (2018).

[22] Wang, D., Tan, D., and Liu, L., “Particle swarm optimization algorithm: an overview,” Soft Computing 22, 387-408 (2018).