ARL-TR-10252 ● JAN 2026

DEVCOM ARMY RESEARCH LABORATORY

Task Assignment Optimization in Operational Mission Planning for Scalable Cross-Echelon Command and Control of the Future

by Angelique Scharine, Mark Dranias, Gregory M. Gremillion, Sarah Al-Hussaini, Ahmed Khalil, Yoonjae Lee, and Efstathios Bakolas

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-10252 ● JAN 2026

Task Assignment Optimization in Operational Mission Planning for Scalable Cross-Echelon Command and Control of the Future

Angelique Scharine and Gregory M. Gremillion
DEVCOM Army Research Laboratory

Mark Dranias
DCS Corporation

Sarah Al-Hussaini
Army Educational Outreach Program

Ahmed Khalil, Yoonjae Lee, and Efstathios Bakolas
University of Texas at Austin

REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		START DATE	END DATE
January 2026	Technical Report	10/1/2024	9/30/2025

4. TITLE AND SUBTITLE
Task Assignment Optimization in Operational Mission Planning for Scalable Cross-Echelon Command and Control of the Future

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
W911NF2220091		
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)
Angelique Scharine, Mark Dranias, Gregory M. Gremillion, Sarah Al-Hussaini, Ahmed Khalil, Yoonjae Lee, and Efstathios Bakolas

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
DEVCOM Army Research Laboratory ATTN: FCDD-RLA-FD Aberdeen Proving Ground, MD 21005	ARL-TR-10252

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES
ORCID(s): Angelique Scharine, 0000-0001-8230-1824; Mark Dranias, 0000-0003-4615-1069; Sarah Al-Hussaini, 0000-0003-4175-4506; Gregory M. Gremillion, 0000-0002-0205-688X; Ahmed Khalil, 0009-0009-4859-8201; Yoonjae Lee, 0000-0003-2953-8907; Efstathios Bakolas, 0000-0003-4104-0108

14. ABSTRACT
The military decision-making process has evolved to ensure that all factors of planning are accounted for. Its current form is complex, labor intensive, and slow. To streamline this process, ARL researchers have developed an integrated technology system consisting of a large language model artificial intelligence tool to generate courses of action (COAs), and a constructive simulation tool to allow commanders to forecast outcomes of potential COAs. To augment the capability of that system, this effort examines the development of a greedy task assignment model to optimize the allocation of military units to mission tasks in a simplified but applied domain of brigade-level mission planning. This application yielded mixed results, as mission planning depends on more than combat power; commanders consider tactical roles, senior commander guidance, and terrain features, which are challenging to account for in a simplified utility model. Here, two implementations are described—a simultaneous assignment and a phased greedy assignment—and the strengths and drawbacks of each method are discussed.

15. SUBJECT TERMS
greedy assignment optimization, military decision-making process, MDMP, courses of action, COA, utility-based optimization, decision-making, task allocation, distributed task assignment, Humans in Complex Systems

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	C. THIS PAGE		
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU	53

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
Angelique Scharine	(520) 691-5562

STANDARD FORM 298 (REV. 5/2020)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

List of Tables

1. Introduction

The objective of this research is to apply optimization methods to the development and refinement of military operational courses of action (COA) in terms of the assignment of military units to mission tasks based on a calculated utility using a greedy algorithm to augment machine learning (ML)-based methods for COA development. Such ML approaches include the generative artificial intelligence tool COA-GPT, which was developed and integrated with the Integrated Technology System (ITS) under the Scalable Cross-Echelon Command and Control (SC3) program within the U.S. Combat Capabilities Development Command (DEVCOM) Army Research Laboratory (ARL). The optimization approaches described herein were also developed under SC3, with the goal of serving as an integrated complement to COA-GPT, which utilizes large language models (LLMs) that do not have the ability to pursue or guarantee optimality, by providing the capability for refinement and optimization of the COAs it generates. Thus, the research detailed in this report is part of a larger research effort within SC3 to develop tools to accelerate and compress the military decision-making process (MDMP).

The MDMP (Headquarters, Department of the Army 2012) is a sequential process used by commanders and planning staff to plan combat missions. The process originates at the highest echelon and is communicated down to lower echelons, wherein each successive, subordinate echelon conducting its own, more granular, planning. This is currently a highly manual, labor-intensive, and hierarchical process to maintain higher-echelon commander intent, while allowing lower-echelon commanders to execute that intent with some flexibility. This work aims to complement other technologies our program is developing to accelerate this laborious process, such as generative artificial intelligence, by providing a more traditional computational approach for COA design that is potentially more transparent, rigorous, and optimal.

2. Background

2.1 Greedy Optimization for Task Assignment in Multi-Agent Systems

Effective allocation of tasks to agents is a fundamental problem across a wide range of applications, including robotics, logistics, and distributed computing. In many real-world scenarios, this translates to assigning resources (agents) to objectives (tasks) in a manner that optimizes a defined performance metric, often minimizing

cost, maximizing efficiency, or balancing workload. The complexity of this problem increases exponentially with the number of agents and tasks, quickly rendering exhaustive search methods computationally intractable. Consequently, heuristic approaches, such as greedy algorithms, have become prevalent due to their relative simplicity and speed.

Task assignment methods are either centralized, relying on a single entity with full system knowledge, or distributed, where each agent knows only its own resources. Centralized techniques like integer programming (e.g., Schumacher et al. [2004]) and the Hungarian algorithm (e.g., Kuhn and Munkres [1958]) ensure optimal solutions but require static, global information, making them less practical in dynamic or decentralized settings. Distributed strategies, such as market-based mechanisms, use local interactions to efficiently allocate tasks without central authority.

Within the realm of distributed task allocation, auction-based mechanisms have gained significant attention. These mechanisms draw inspiration from economic principles, where agents bid on tasks based on their perceived cost or utility. A notable example is the greedy decentralized auction-based task allocation algorithm proposed by Braquet and Bakolas (2021). This approach iteratively assigns tasks to the highest bidder, prioritizing immediate gains at each step. The algorithm's decentralized nature allows for scalability and robustness in dynamic environments.

Greedy algorithms are known for their computational efficiency, which makes them appropriate for real-time applications. One limitation of these methods is that they can result in solutions that are not optimal, as they prioritize immediate rewards over potential long-term gains. The effectiveness of a greedy algorithm depends on the bidding strategy used by agents and the attributes of the task and agent environment.

This report extends the work of Braquet and Bakolas (2021) by examining whether their algorithm can be modified to optimize tasks that occur across multiple phases. In many military mission contexts, tasks often proceed in sequential phases, with decisions in earlier stages influencing options and costs in subsequent ones. The MDMP organizes a mission in phases. Therefore, it must plan for unit movement from one area of interest (AOI) to another and ensure actions are synchronized across units. For example, a reconnaissance phase typically comes before a direct-action phase, and logistical support must be provided for later mission phases. Combat strength is dynamic and will change from one phase to the next, and these changes must be predicted and planned for. Synchronization in military missions is important because later phases frequently depend on the successful completion of previous ones and avoid gaps in the control of key areas. Lack of synchronization

can result in critical terrain being left unprotected, which may allow opposing forces to interfere with operations or gain an advantage.

A greedy algorithm requires a utility function that considers both task value and the tactics or standard operating procedures used in mission planning. The MDMP combines staff expertise with redundant processes to ensure all contingencies are addressed. Below is a description of the implementation of a basic greedy algorithm, followed by the implementation of one adapted for sequential military missions.

2.2 The Military Decision-Making Process

Here we briefly summarize the steps of the MDMP. These steps occur at each echelon above Company, allowing planning to occur at the appropriate level of detail for that echelon. The process requires a great deal of communication across all levels to ensure that the commander's intent is understood and that all contingencies are accounted for. Consequently, this is a lengthy, labor-intensive process.

2.2.1 Receipt of Mission

Starts with receiving a directive or order from higher headquarters. This outlines the initial task, purpose, and available resources.

2.2.2 Mission Analysis

- Understand the problem.
- Analyze the area of operations (terrain, weather, civil considerations, threats).
- Identify constraints on time, resources, and rules of engagement.
- Develop a concise, clear statement of the commander's intent.
- Determine which key tasks are required for success.

2.2.3 COA Development

- Generate multiple ways to accomplish the mission.
- Briefly outline each COA, how it will work, resources needed and potential outcomes.
- Assess each COA against friendly and enemy capabilities, terrain, and potential risks.

2.2.4 COA Comparison

- War-game each COA, walking through each step. This identifies potential problems and vulnerabilities.
- Compare the results of each COA based on criteria like feasibility, acceptability, distinguishability, completeness, and suitability.
- Evaluate the risks associated with each COA and develop mitigation strategies.

2.2.5 COA Approval

- Brief the commander on the COA comparison, recommendations, and associated risks.
- The commander selects a COA or directs modifications/further development.

2.2.6 Order Issuance

Develop and disseminate the operation order (OPORD), a detailed written plan outlining *how* the selected COA will be executed. Includes tasks, timelines, responsibilities, and coordinating instructions.

2.2.7 Supervision, Assessment, and Liaison

- Track execution progress and identify any deviations from the plan.
- Continuously evaluate the operational environment and the effectiveness of the plan. Be prepared to modify the plan based on changing circumstances.
- Maintain communication with other units and stakeholders.

2.3 Approaches to an Integrated Technology System

The MDMP forces leaders to consider all aspects of the planning process, from consideration of resources to intelligence about the opposition force (OPFOR). While time consuming, the danger of automating this process is that part of the planning process is a training process that ensures that all elements understand the objectives and have an opportunity to give feedback on potential issues that require additional planning.

Because of this, it is important that commanders be able to maintain control of at least some aspects of the planning process, while others can be sped up and automated. The ITS consists of three efforts: the first, COA-GPT, generates potential COAs using a large-language model that ingests all related documentation; a constructive simulator that estimates probability of success for

each COA generated, and a user interface that allows the commander to adjust and make changes to the COA based on its output. As a future component of the ITS, we developed a non-ML–based task-assignment algorithm that optimizes task assignments using greedy assignment over a combination of performance measures and mission constraints. This report describes this greedy assignment algorithm, which was intended as both a check to the outputs of COA-GPT and a tool that leverages and drives the constructive simulation.

3. The Optimization Problem

3.1 Greedy Assignment Optimization

Braquet and Bakolas (2021) introduce a decentralized auction-based algorithm for dynamic task allocation in multi-agent systems. The problem involves agents (e.g., robots, drones, or autonomous vehicles) completing spatially distributed tasks while negotiating task assignments without relying on a central authority. The proposed method, called the Greedy Coalition Auction Algorithm (GCAA), uses auction-based principles where agents bid on tasks based on their individual utilities, which are calculated as the difference between the rewards for completing tasks and the costs incurred.

The GCAA algorithm operates by maintaining three vectors for each agent: a task assignment vector, a utility vector, and a finalized allocation vector. Agents communicate locally within their range, updating their vectors iteratively to determine the best task allocation. The algorithm guarantees convergence within the number of steps that is equal to the number of agents, ensuring computational efficiency.

Khalil et al. (forthcoming 2026) modify this approach to operate in a hierarchical environment where lower-level agents are constrained by upper-level assignments. They, instead, model the problem as a Mixed Integer Linear Programming (MILP) problem and use the Hungarian algorithm to ensure computational efficiency. The hierarchical approach resembles a two-echelon version of the command problem described here, except that the upper-level echelon is essentially the aggregate of the lower-level units.

A modified version of these approaches is described in Section 4.1 that combines the use of MILP and the Hungarian algorithm with a utility function consisting of the value of a particular assignment minus the cost. Two implementations are described, simultaneous and phased versions.

3.2 Operation Tropic Tortoise

Operation Tropic Tortoise (OTT) is a fictional mission used to train Soldiers in the MDMP process. Although the model is intended to be generalizable, this will be the mission used in all examples. It consists of a division-level OPORD that would be given to brigade-level commanders. Each brigade commander would develop their own plans for the battalions within their brigade. In this case, the plans are being developed for the 2nd Brigade Armored Division (2/1AD, Figure 1). Appendix A provides a sample JSON file describing the combat elements of the 2/1AD. In demonstrating this model, engineering and support units are ignored as the model currently targets the combat elements (circled in red) of the mission planning process.

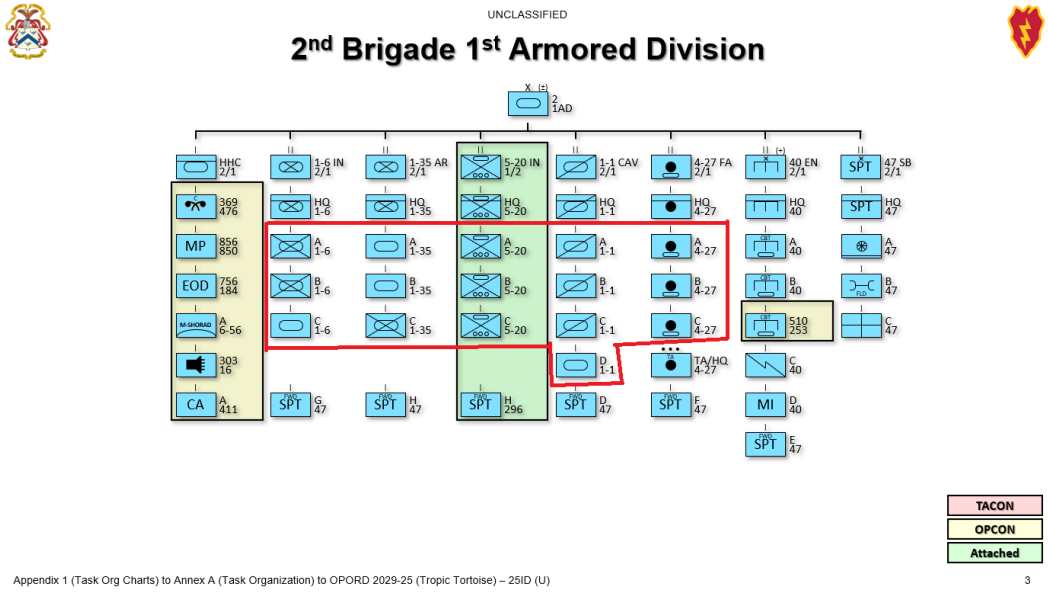

Figure 1. Table of Organization and Equipment chart for the 2/1AD. The units circled in red are the mechanized infantry, armor, Stryker, cavalry, and artillery units and constitute the main combat elements.

Similarly, Appendix B shows the JSON file information about the OPFOR hostile combat units. Note that only the elements believed to be within 2/1AD's area of responsibility (AOR) are included here. Once again, for this initial model, we omit the support and engineering units.

3.3 Can Greedy Optimization Imitate Mission Commanders?

Typically, each echelon is assigned to a specific geographical AOR for which they plan operations. The division OPORD specifies these boundaries and divides the region into several mission phases. This allows the division to synchronize their activities across brigades (Figure 2). Mission tasks or objectives are specified here

as Objective (OBJ) MADISON and OBJ MONROE. The artillery elements are assigned to the Position Area for Artillery (PAA) locations on the map. The dotted vertical line running from phase line (PL) BRONZE to PL BLUE is where the 20 ABCT (another brigade) takes on the main effort in the eastern side and 2/1AD is assigned to the western side.

The overall mission of OTT is to neutralize OPFOR and move the blue force (BLUFOR) to PL Blue at the southern end of the map. The map is divided into four phases, and the tasks for each phase can be described as 1) Move into and clear first phase area and emplace artillery units at new Position Areas for Artillery (PAAs) in the phase 1 AOR to soften their next target, OBJ MADISON; 2) Attack, clear, and occupy OBJ MADISON in the phase 2 AOR; 3) Move into position along the southeastern border of the phase 3 AOR to assist the 20 ABCT brigade (if needed) in fixing OBJ MONROE; and finally, 4) Clear phase 4 and move to PL Blue.

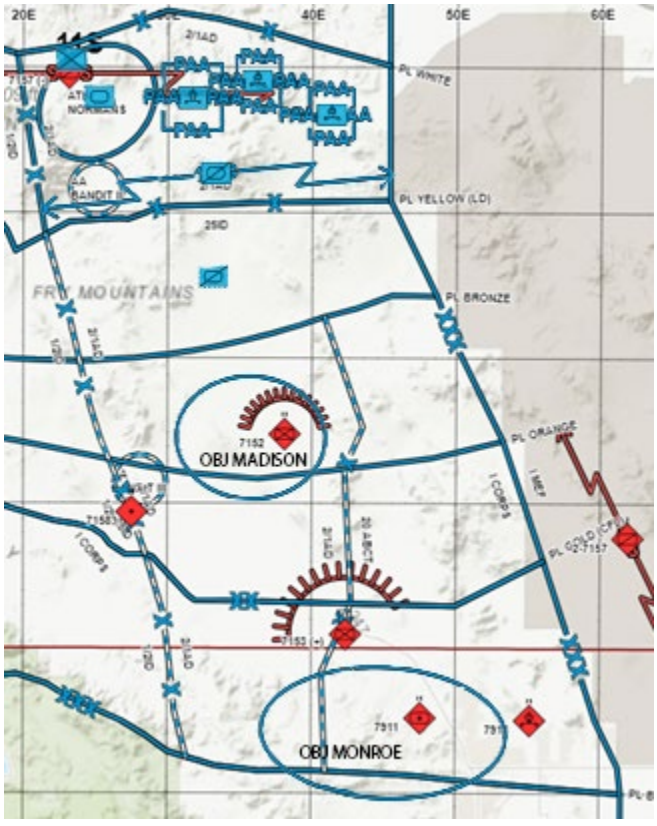

Figure 2. AOR for the 2/1AD. The horizontal lines represent the different PLs. The line of departure is PL YELLOW. The blue rectangles represent the initial position of the BLUFOR. The red diamond is the approximate location of the OPFOR.

Mission commanders plan in time and space, understanding that units will need time to reach the more distant objectives, this time will depend on the terrain, and their resources will change dynamically as a function of time. The original greedy assignment model does not account for time and assigns all elements simultaneously.

Mission commanders understand that different types of units have different tactical strengths and have experience using these units tactically. Using the example above, artillery units are typically assigned to protect assets and to soften OPFOR targets because they can operate at a distance from their targets. While their combat power may not be as high as the armored units, they are important tactically. Commanders usually assign reconnaissance to obtain intelligence about their targets, and reserve units assist other units in case initial estimates are incorrect and more forces are needed.

The combination of combat power, space/time features, and tactical/commander preferences is difficult to reduce into a single utility metric for use in greedy assignment. The aim of this project was to modify the greedy assignment algorithm to function more like a mission commander.

3.4 Correlation of Forces Matrix

Commanders have traditionally used the Correlation of Forces Matrix (COFM) to estimate outcomes of BLUFOR and OPFOR engagements. The COFM considers unit type (e.g., "Mechanized Infantry," "ABCT," "IBCT," or "SBCT"), posture (offense or defense, planned or hasty), and health to predict lethality and attrition for both sides. While not exact, these tables help assess whether a BLUFOR attack on OPFOR is likely to succeed and its resultant impact.

Typically, this calculation provides a rough estimate of the units needed for success and loss percentages for BLUFOR and OPFOR. The definition of "win" is unclear; but, for general purposes, we defined a "win" when the OPFOR was reduced by 40%. Therefore, the utility of any assignment of a unit to a task within a phase starts with the unit's COFM utility value.

4. Methods

A task-assignment model is proposed that couples a greedy algorithm with a utility function that depends on unit level properties to optimize task assignment.

4.1 Task Assignment Algorithm

The greedy assignment algorithm developed by our University of Texas at Austin colleagues consists of three loops and is shown as pseudocode below (Figure 3).

```
FUNCTION UTAlgorithm(units, tasks):

   1. assignments ← empty, phase utility ← 0
   2. WHILE |unassigned(units)| > 0:
         a. FOR t IN tasks:                                  # loop over tasks
               i.  FOR u IN unassigned(units):    # loop over unassigned units
                     1. unit-task-utility[u,t] ← ComputeUtility(unit, task,
                        assignments)
         b. (u*,t*) ← argmax {u,t} unit-task-utility [u,t]    # the best unit-
            task pair
         c. assignments ← assignments ∪ {(u*,t*)}      #Update assignments
         d. phase_utility ← phase_utility + unit-task-utility_[u*,t*]  #phase
            utility
         e. Update units and tasks

RETURN assignments, phase utility
```

Figure 3. Pseudocode for the greedy assignment algorithm.

The outer loop (2.) ensures the algorithm assigns the best tasks for every friendly unit. The middle loop (2.a.) cycles through tasks and the inner loop (2.a.i. cycles through unassigned friendly units. The inner two loops ensure the best unit assignment is found for each task and the outer loop locks in the best overall (unit, task) assignment (2.b., 2.c.) from those pairings and removes that unit from future tasks (2.e.).

4.2 Utility Algorithm

A utility function was developed to guide task assignments by computing the total utility of a task assignment using a weighted combination of values from four criteria or modules: combat, guidance, spatial, and doctrine (Figure 4).

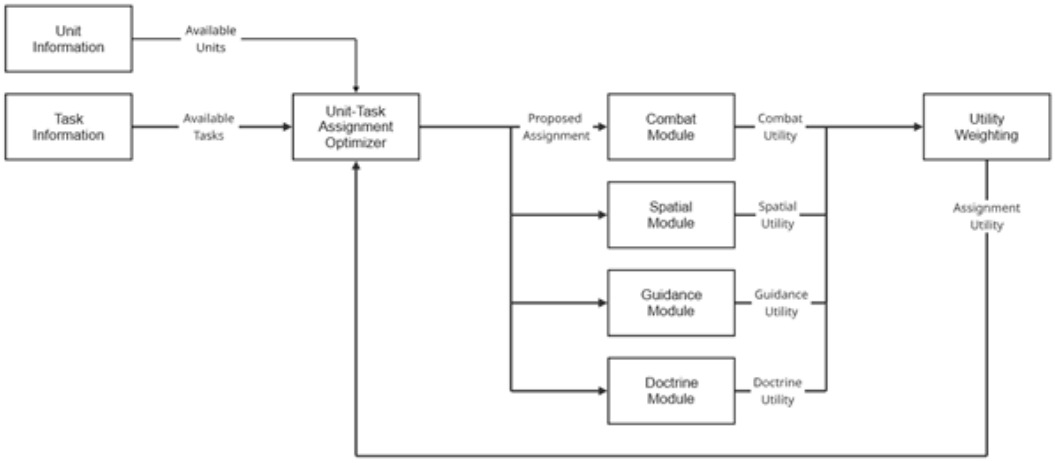

Figure 4. Optimization can be calculated based on several factors, including combat strength, relative locations of forces, commander guidance, and general doctrinal factors. The combination of these factors weighs the utility of any task assignment; therefore, the goal is to discover the set of task assignments that provides the greatest utility. The Unit-Task Assignment Optimizer corresponds to the UT greedy algorithm.

Different approaches were used to compute the utility function, and simulations were performed to assess the relative strengths of these approaches. The present set of simulations weighed all modules equally except the spatial module where it was set to zero (deactivated). The general form for computing utility is detailed below as pseudocode (Figure 5).

```
FUNCTION ComputeUtility(unit, task, assignments):

   1. combat ← combat utility
   2. spatial ← spatial utility
   3. guidance ← guidance utility
   4. doctrine ← doctrine utility

RETURN    weights.combat * combat   + weights.spatial * spatial

         + weights.guidance * guidance+ weights.doctrine * doctrine
```

Figure 5. Pseudocode representation of assignment utility, shown as a function of each component.

4.2.1 Combat Utility

For the pseudocode used to compute combat utility, see Appendix C. Four variants in the formula for computing combat utility are explored. Combat utility is driven by COFM predictions of expected losses. Units are considered disabled when their health dropped below 60%; this was imperfectly captured by capping losses per round at 40%. Define the cap,

$$c(x) = \min(40, x) \tag{1}$$

COFM takes as arguments the health adjusted force value of units and a posture where the default COFM postures for BLUFOR and OPFOR are deliberate attack and deliberate defense, respectively.

Let $COFMdef$ represent the defender losses returned by the COFM function and $COFMatt$ be the attacker losses, where $CUij$ indicates combat utility for the assignment of BLUFOR unit Ai to attack OPFOR unit Tj, and Sj is the set of BLUFOR units currently assigned to attack OPFOR unit Tj. The four variants are defined below:

OPFOR losses (raw). Combat utility is directly expressed as the percentage of enemy losses predicted by the COFM calculator:

$$CU_{ij} = \text{c}(\ COFM_{def}(S_j \cup A_i, T_j, default) \tag{2}$$

OPFOR losses (marginal). Combat utility is expressed as the marginal gain of assigning a unit to a given task:

$$CU_{ij}^{\Delta} = c(COFM_{def}(S_j \cup A_i, T_j, default\ ) - COFM_{def}(S_j, T_j, default\ )\ ) \tag{3}$$

Net losses (raw). Both OPFOR and BLUFOR losses are considered, and combat utility is the weighted net loss:

$$CU_{ij}^{net} = c\left(\ COFM_{def}(S_j \cup A_i, T_j, default\ )\right) - 0.5 * COFM_{att}(S_j \cup A_i, T_j, default\ ) \tag{4}$$

A weight of 0.5 is placed on BLUFOR to encourage attacks.

Net losses (marginal). This last case computes combat utility as an additional increment in net losses:

$$CU_{ij}^{\Delta net} = CU_{ij}^{net} - \left(c\left(COFM_{def}(S_j, T_j)\right) - 0.5 * COFM_{att}(S_j, T_j)\right) \tag{5}$$

4.2.2 Guidance Utility

Commander guidance was implemented by creating different types of points to add to map locations. Orders are encoded as salience maps over nodes: RESERVE, CLEAR, OBJECTIVE, AOI/PAA, END-RALLY, THREAT. Each node stores a 6-vector of utilities. Each BLUFOR unit has a 6-vector of decision weights w. Guidance utility at a node is the layerwise dot product:

$$GU = w^T \cdot node.utilities \tag{6}$$

Figure 6 shows these nodes as a function of task type. Most point values clear after one round of occupation. Figure 6A shows the AOI source locations and relative influence used by the occupy utility Figure 6B shows the Clear source locations, which are areas designated for clearance that contribute to the occupy utility. Figure 6C highlights the Objective nodes that accrue points when occupied. Figure 6D shows the End Stage source location that indicates the node that serves as the staging area at the end of the operation. Figure 6E gives the Reserve position where the reserve element is assigned; a higher weight encourages holding behavior. Figure 6F shows the Threat locations. This is a binary indicator based on presence of active OPFOR (threat) units.

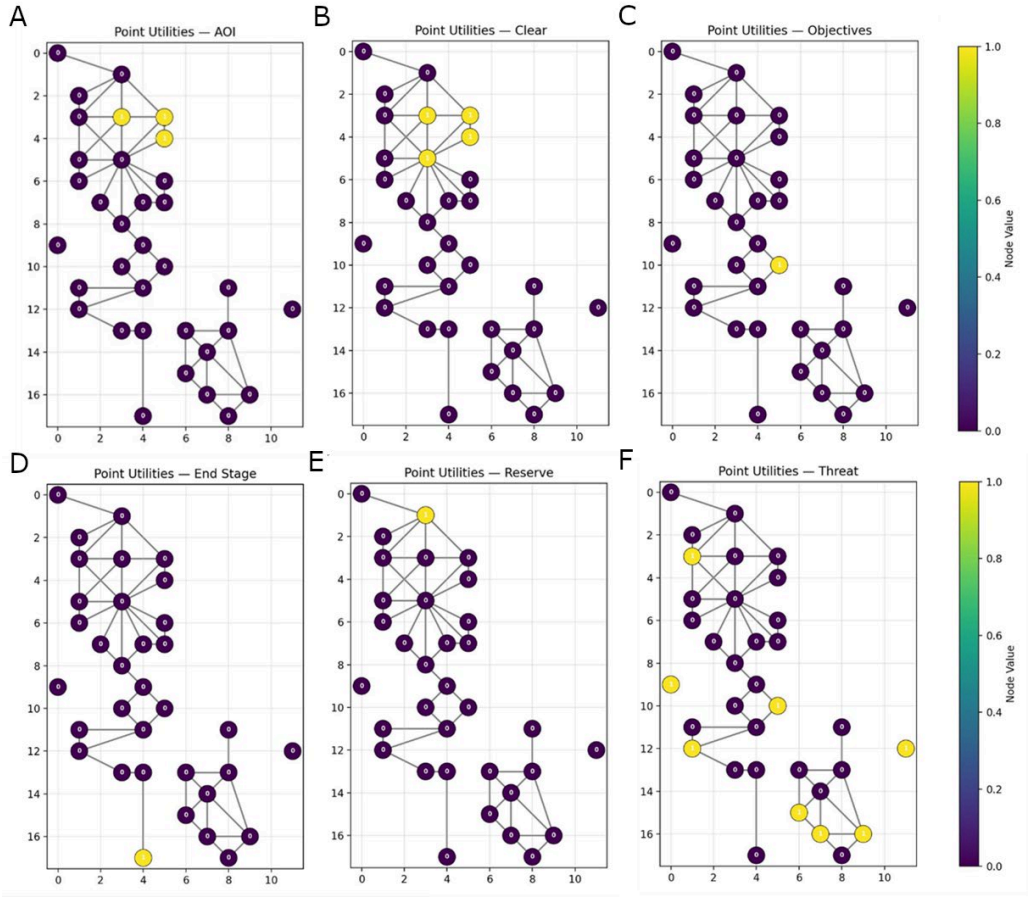

Figure 6. Commander guidance utilities assigned to OTT map.

4.2.3 Doctrine Utility

A herding penalty was incorporated as a doctrine utility. This was a simple rule to discourage units with similar orders from piling up on one location. Appendix D lists the full set of doctrinal rules that are defined for inclusion in future work.

The herding penalty was actualized as a diminishing-returns term when many friendlies are already assigned to the same location:

$$DU \approx (w[layer] * node.utilities[layer]) * (1 - expit(assigned\_friendly\_at\_loc / total\_friendly\_force)) \quad (7)$$

This approximates saturation as more friendly force piles onto the same location during the assignment pass. Distance penalties are supported (and currently set to zero by default in the provided scripts).

4.2.4 Spatial Utility

The spatial utility incorporates a Euclidean distance penalty for the distance from unit to task. However, the weight is set to zero in this study.

4.2.5 Simulation Platform

The game was coded in Python (3.x) and used pandas and NumPy for state/logging, metrics, and analysis; matplotlib for plotting; and scipy.special.expit to compute when the assignment of BLUFOR units to a task has saturated and additional assignments would provide diminishing returns. Information about units and unit strength of both OPFOR and BLUFOR elements are provided as JSON files. The code is stored in a GitLab repository (https://gitlab.sitcore.net/sc3/sc3-greedy/-/tree/main/UT_greedy_game). More implementation details, including file structure and commands, are included in Appendix E.

The game board was implemented as a graph with vertices or nodes and edges ($G(V, E)$). The graph was derived from the OTT map (Figure 7). Nodes represent map grid locations; edges represent traversable adjacencies.

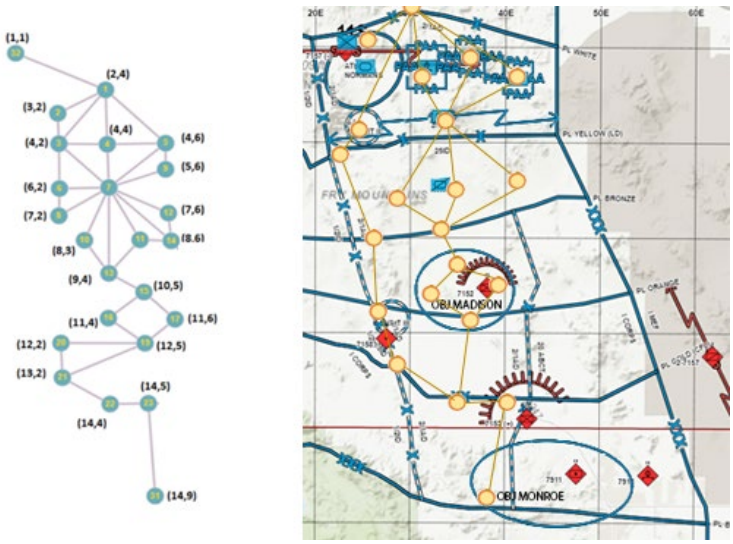

Figure 7. Graph nodes and how they overlay on the AOR.

Two forces (BLUFOR and OPFOR) are placed on this graph as units. Included in the simulations were 16 BLUFOR companies from 5 divisions in the 2/1AD. These 16 companies are represented as units and exclude headquarters and support. In the AOR, BLUFOR's opponent consisted of nine companies from four divisions of the OPFOR's 715 Combined Arms Brigade. Each military unit (Table 1) has static attributes, like type, echelon, and team, as well as dynamic ones, such as location, health, task assignment, and posture. Understanding each unit's combat strength and role is key to optimizing task assignments within command.

Table 1. Unit attributes.

Attribute	Description
Name	Unit type (light armor, heavy armor, infantry, artillery)
Echelon	Corps, Division, Brigade, Battalion, Company, Platoon, Section
Team	Friendly/Hostile
Health	Percentage of unit/weapons available for combat
Force	Strength of combat power
Location	Current position on map
Speed	Speed at which unit can travel
Task	(assigned value consisting of location and hostile entity)
Posture	Planned/Hasty – Defense/Attack

The two modes of play are simultaneous and phased. In simultaneous mode, all nodes and enemy units in the same connected graph are available for assignment and friendly units have no movement penalty. In phased mode, the OTT map is partitioned into subgraphs and only nodes or enemy units within the currently active subgraph are considered during task allocation.

Game play during phase mode can be summarized using the pseudocode in Figure 8.

```
FUNCTION Main()

   1. Initialize
   2. operation_utility ← 0
   3. FOR phase IN phases   #loop over phases
         a. Update units and tasks
         b. (assignments, phase utility) ← UTAlgorithm(units, tasks)
         c. Update states
         d. Compute operation utility += phase utility
```

Figure 8. Pseudocode representation of game play during the phased mode.

Figure 9 shows the OTT map populated with units at the initial stage.

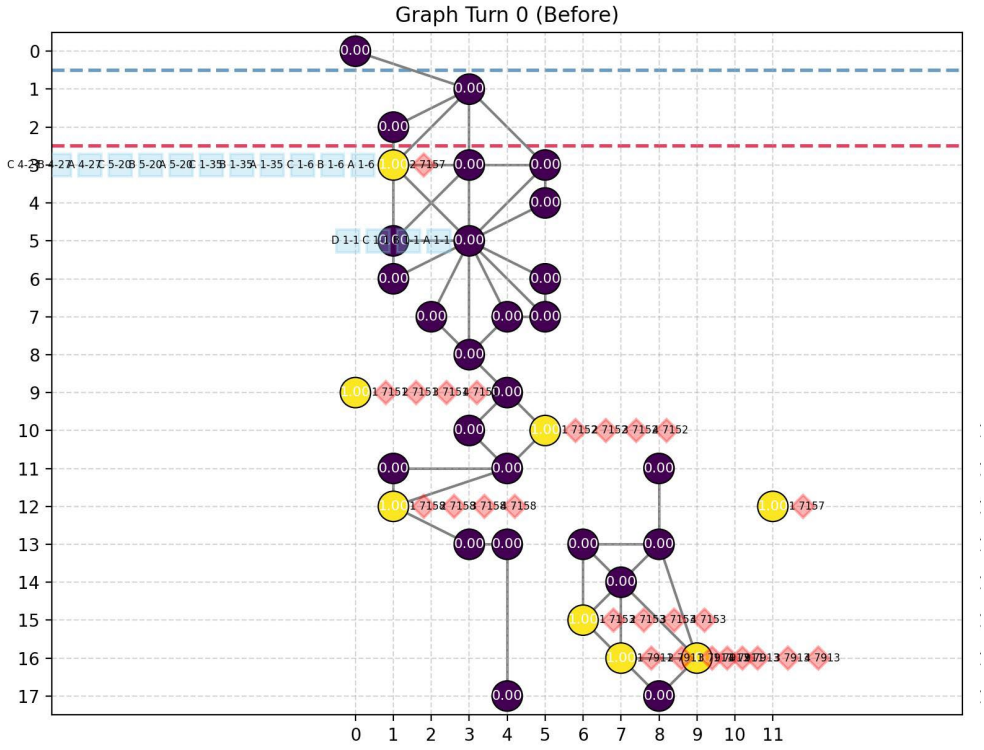

Figure 9. Initial game map and overlays (turn 0). Baseline force disposition with overlays for sources. Note: phase boundary lines are only relevant to the phase-mode experiments; they do not constrain movement in simultaneous mode. In the phased implementation of the greedy model, phase "bands" are identified by color (e.g., white, yellow, red). Each phase is defined as a subgraph of nodes available for assignment within that phase.

Turn order, saving/loading game state, and removal of units are handled by the game engine. Each unit's specific decision weights (i.e., how much to value each salience map) are defined in JSON configuration files, which act as their orders.

4.2.6 Experiments

Simulations were performed to evaluate six decision policies across two connectivity regimes (simultaneous and phased) on a graph-based board derived from the OTT map. Policies included two baselines (random, uniform–random) and four greedy variants that differ in the objective function underlying combat utility calculations: absolute enemy loss, marginal enemy loss, absolute net loss (enemy − α·friendly), and marginal net loss. In all runs, movement and engagements were constrained by connected components (simultaneous) or by phase-banded subgraphs (phase).

The first random baseline randomly selected nodes from among available targets with utility value. The second random baseline (uniform–random) uniform randomly selected nodes from the connected graph.

In addition to doing simulations with different utility functions we observed the effect of parameter changes on the greedy utility functions. In these functions the total utility was the sum of combat, guidance, doctrine, and spatial utilities, each with their own weight. In an "attack only" simulation, combat utility was set to 1 while guidance, doctrine, and spatial utility were set to zero. In a "mixed" simulation, combat, guidance, and doctrine utilities were set to 1 and spatial was set to zero. For these comparisons the average number of movement assignments and attack assignments across the four greedy policies was computed.

Simulation and utility function performance were evaluated by tracking a number of state features across allocations: objectives secured, movement of the BLUFOR centroid across game (front depth), force value of disabled OPFOR units, force value of disabled BLUFOR units, BLUFOR units destroyed, OPFOR units destroyed, and an "allocation objective" proportional to the geometric mean of the force ratios of individual attack assignments:

$$Allocation\ Objective = \sum_j \log\left(S_j/T_j\right) \tag{8}$$

where j indexes OPFOR units T that have been matched to a group of BLUFOR units to attack (the force ratio applied to each unit attacked).

5. Results and Discussion

5.1 Utility Function Behavior

Figure 10 shows that utility function parameters alter unit behavior. In these simulations the weights on the utility function ComputeUtility() were either mixed (combat, guidance, doctrine, spatial = 1,1,1,0) or set to attack only (combat, guidance, doctrine, spatial = 1,0,0,0). Movements were allocated only when the parameters of the utility function were set to mixed.

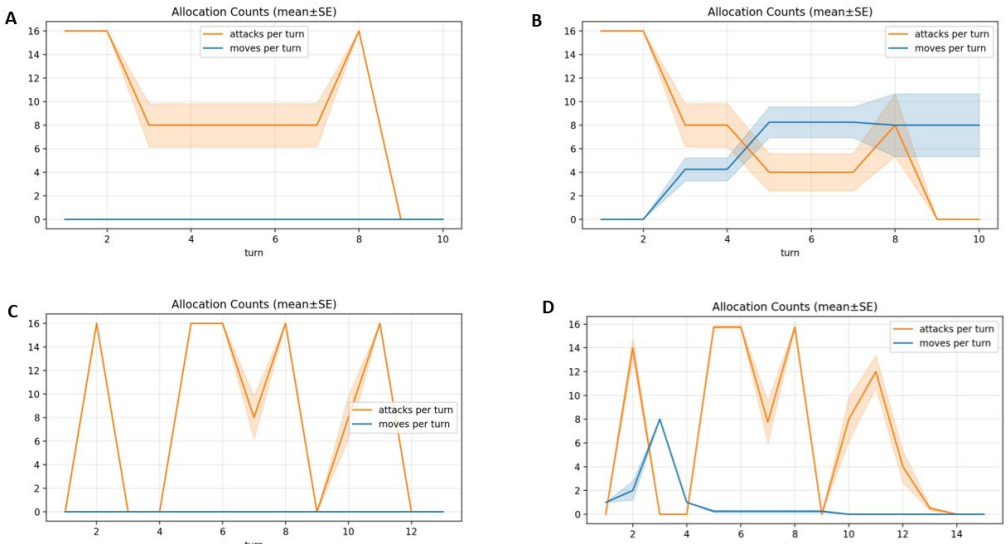

Figure 10. Utility function parameters drive behavior. Allocated attacks (orange) and movements (blue) averaged across all four greedy utility conditions for attack-only utility weightings (left, A and C) and mixed utility weightings (right, B and D). A and B tally allocations during the simultaneous game mode while C and D tally allocations during phased assignment. Shadowing is standard deviation for five simulations of each greedy policy (n = 20 overall).

5.2 Game Outcomes

Observations were also made to assess different utility policies on game outcomes in both simultaneous and phased game modes. To assess the performance of these different policies, the changes in several key game metrics were observed across trials as well as a candidate objective function for optimization. The results from the simultaneous game mode are shown in Figure 11. The worst performance here is seen when the utility function uses raw OPFOR losses (green); no objectives are secured, the least advancement in the map is seen, and the highest number of BLUFOR casualties is acquired.

The utility function that relies on the net losses of OPFOR and BLUFOR (purple) is the only utility function where BLUFOR ends the game without any losses. It also ties the best-performing algorithm in securing objectives. This performance comes at the expense of leaving most of OPFOR intact.

The best-performing utility functions use the marginal losses. When marginal OPFOR losses are used (red), the most objectives are secured (7/9), the second most advancement on the map is observed, OPFOR is quickly destroyed, and BLUFOR suffers the second least losses. The marginal net loss (brown) performs similarly in destroying OPFOR and securing objectives, but there is an additional BLUFOR loss.

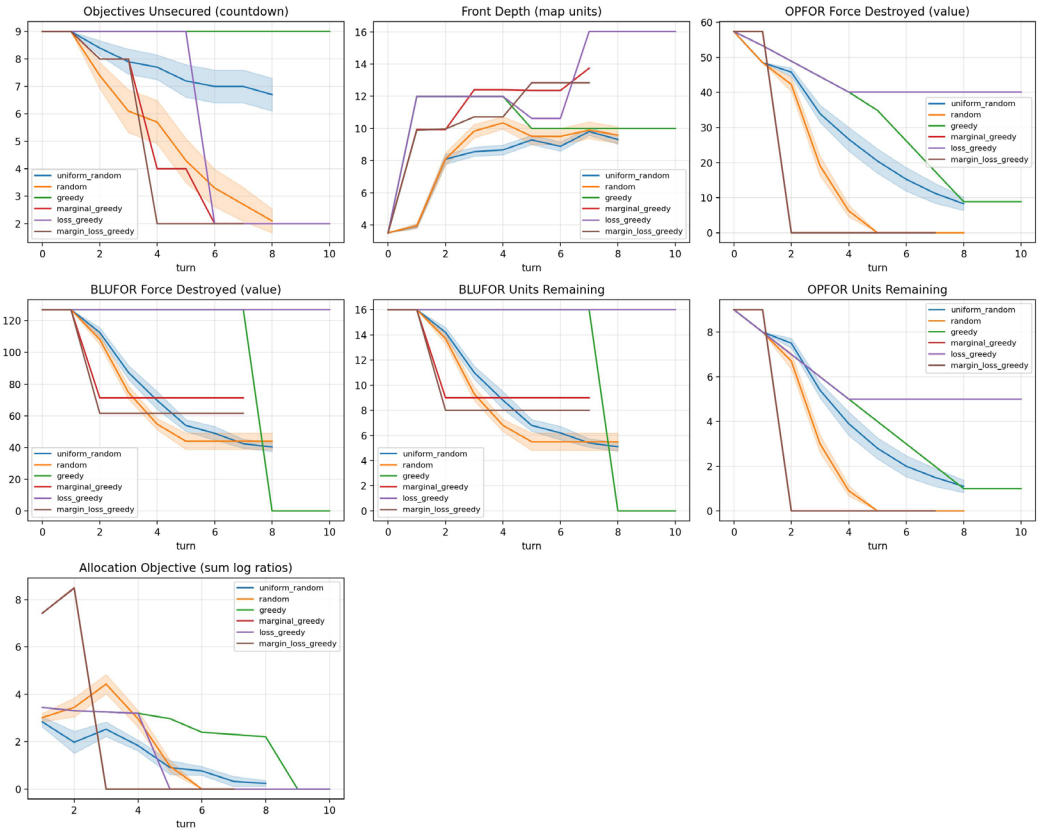

Figure 11. Changes in Game Metrics during Simultaneous Mode Game Play. Observations are drawn from six utility function policies: uniform random across all locations, random from among all targets, based on OPFOR losses, based on the net losses between OPFOR and BLUFOR, based on marginal OPFOR, and net losses.

Uniform–random (blue) performed the second worst overall with fewer losses and more objectives obtained than OPFOR losses. Randomly choosing among available targets (orange) performed intermediately—not as good as marginal utilities but better than others.

5.3 Simultaneous vs. Phased

Results from the phased mode game play are shown in Figure 12. The best and worst algorithms do not segregate themselves out as clearly as in simultaneous mode. In terms of BLUFOR losses they all perform as poorly as the worst utility function in simultaneous mode; only when targets are picked using a uniform–random policy are there any BLUFOR survivors. However, uniform–random performs the worst on the other available metrics. Setting aside the ultimate doom of BLUFOR, the utility function relying on the marginal OPFOR losses performs best, securing the most objectives (8/9) and destroying all OPFOR. Choosing randomly from among available targets performed second best, finishing with

similar results as the marginal OPFOR losses, just taking longer to accomplish it. The utility function relying on the marginal net losses performed next best, differing only in destroying less OPFOR. Utility functions driven by OPFOR losses or net losses performed better than a uniform–random policy only in the number of objectives secured.

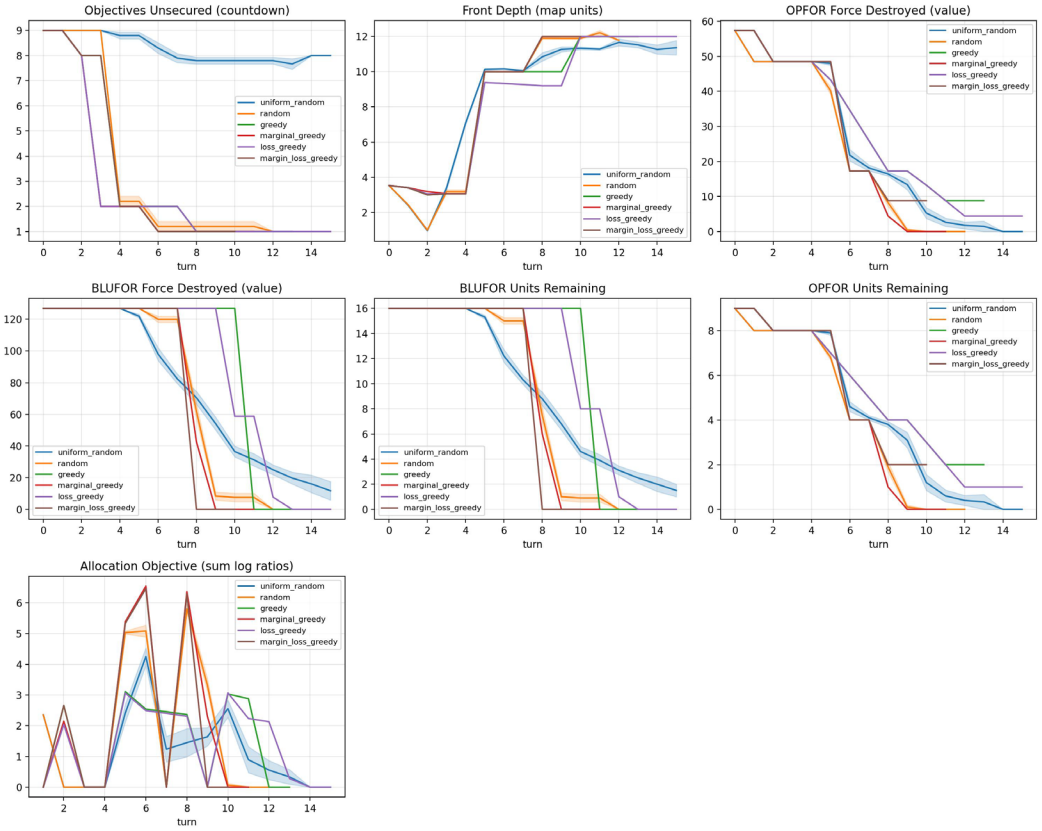

Figure 12. Changes in game metrics during phase mode game play. Observations are drawn from the same six utility function policies as Figure 11.

5.4 Discussion

Two consistent findings emerged: marginal utilities outperformed utilities based on raw losses and simultaneous mode tended to outperform phased mode.

Marginal utility functions outperformed utility functions relying on raw losses. The greedy optimization function is not designed to work with convex cost functions—it results in piling on behavior. Hence, when utility functions relied on OPFOR losses or net losses, the game would have BLUFOR units pile on individual OPFOR companies until they were all destroyed. When concave marginal utility functions were used, BLUFOR forces distributed across OPFOR targets during game play, destroying OPFOR with better results.

Phase mode underperformed simultaneous mode because the game penalizes piling-on strategies. One of the mechanisms driving this is the rule that units with less than 60% health are disabled. When a pure piling-on strategy is followed, all BLUFOR units suffer the same damage in each encounter (damage is distributed uniformly across all attackers). After a couple battles, the entire BLUFOR force suffers the same fate when they drop to 59% health: they are disabled. If forces distribute, some battles will result in units having more damage and others less. In simultaneous mode and with marginal utility functions, these distributed attacks happen naturally. In phase mode forces tend to mass in the current subgraph where there is often only one division of enemies to attack, resulting in piling on. Phase mode has a greater need for doctrinal constraints that encourage some forces to be held in reserve for future targets.

Two consistent findings emerged. First, in simultaneous mode the marginal net-loss policy outperformed other approaches, reflecting the benefit of incremental, utility-aware concentration of force when movement is unconstrained within components. Second, in phase mode the same policy underperformed because the phased subgraph staging fragmented the force and delayed massing effects; the policy's incremental advantage was blunted by the exogenous phase boundaries.

A practical implication is that parametric choices matter as much as policy form. The relative weighting of decision dimensions (e.g., threat vs. objectives vs. end-stage) and the saturation strength that governs herding can materially change outcomes. This suggests parametric studies should accompany policy selection to identify robust operating regions under a given scenario and phase design.

Future parametric studies (decision_weights) could explore 1) early-phase emphasis on objectives (increasing OBJ weight) versus late-phase sprint to exits (increasing END weight), 2) shaping with stronger AOI weighting to pre-position mass prior to contact, 3) differential reserve (RES) weighting to enforce hold behavior for a reserve element, 4) phase-aware schedules that anneal weights as the game advances, and 5) controlled heterogeneity across units (e.g., high threat weight for armor vs. high OBJ weight for infantry) to test combined-arms effects under both connectivity regimes.

6. Conclusions

6.1 Key Findings

This study demonstrates the significant impact of utility function parameters on unit behavior and overall game outcomes in a war-gaming environment. Specifically, utility functions leveraging *marginal* losses—both OPFOR and net losses—

consistently outperformed those based on raw loss calculations. Furthermore, the *simultaneous* game mode generally yielded superior results compared to the *phased* mode.

6.2 Marginal Utility Superiority

The observed outperformance of marginal utility functions stems from their ability to mitigate "piling on" behavior. Raw loss-based utilities incentivize concentrating all available firepower on a single target, leading to inefficient resource allocation and, ultimately, diminished returns. Marginal utilities, by focusing on incremental gains, encourage a more distributed approach to target engagement, maximizing overall effectiveness.

6.3 Impact of Game Mode

The phased game mode's underperformance is attributed to its inherent constraints, which inadvertently promote the very piling-on strategies it penalizes. The rule disabling units below 60% health exacerbates this issue, leading to rapid attrition of BLUFOR forces when concentrated attacks occur. In reality, commanders combine attritted units with reserve or other units to compensate for attrition, a feature that was not possible in this model. Simultaneous mode, with its unconstrained movement, allows for the natural distribution of forces facilitated by marginal utility functions.

6.4 Parametric Sensitivity

This research underscores the critical importance of parametric choices within utility functions. The weighting of different decision dimensions (threat, objectives, end-state) and the strength of saturation effects can dramatically alter outcomes. This highlights the need for comprehensive parametric studies to identify robust operating regions tailored to specific scenarios and phase designs.

6.5 Recommendations

It is surprising that the simultaneous method worked better than the phased method. However, it is likely that military commanders plan across phases simultaneously, prioritizing different units during each phase and then plan the phases around this structure. Future work should try implementing a simultaneous allocation initially and then optimize assignments within each phase. It should also focus on exploring phase-aware schedules that dynamically adjust utility function weights as the game progresses. There needs to be a recombining feature for the BLUFOR troops that enables regrouping of units when units fall below 60%. Additionally, investigating

controlled heterogeneity in unit weighting (e.g., prioritizing threat for armor and objectives for infantry) could reveal valuable insights into combined-arms effects under varying connectivity regimes. Finally, more sophisticated optimization approaches that can search over the solutions space, beyond the limitations of a myopic greedy approach, will be necessary to produce more optimal plans, particularly if optimality guarantees are desired or required. These findings have practical implications for the development and implementation of AI-driven decision support tools for military planning and execution.

7. References

Braquet M, Bakolas E. Greedy decentralized auction-based task allocation for multi-agent systems. ArXiV. 2021. arXiv:2107.00144 [eess.SY]. https://arxiv.org/abs/2107.00144

Headquarters, Department of the Army. The operations process. Headquarters, Department of the Army (US); 2012 May 17. Army Doctrine Publication No.: (ADP) 5-0.

Khalil A, Lee Y, Bakolas E, Gremillion G. Coupled task assignment for hierarchical multi-agent systems in disaster response missions: a Hungarian method approach. AIAA SCITECH 2026 Forum; forthcoming 2026.

Kuhn HW, Munkres JE. A Hungarian method for the assignment problem. Nav Res Logist Q. 1958;5(1–2):83–97.

Schumacher C, Chandler P, Pachter M, Pachter L. UAV task assignment with timing constraints via mixed-integer linear programming. Proceedings of AIAA 3rd "Unmanned Unlimited" Technical Conference; 2004; Chicago, IL. https://doi.org/10.2514/6.2004-6410

Appendix A. BLUFOR Units

```
{
    "Entity": "25ID",
    "echelon": "Division",
    "cofm value": 807.3,
    "Description": "Army Division",
    "children": [
        {
            "Entity": "2/1 AD",
            "echelon": "Brigade",
            "cofm value": 248.4,
            "Description": "Armored Bde (ABCT) (M1A1)",
            "children": [
                {
                    "Entity": "1-6 IN",
                    "echelon": "Battalion",
                    "cofm value": 37.04,
                    "Description": "IN Combined Arms Bn (M1A1)",
                    "children": [
                        {
                            "Entity": "A 1-6",
                            "echelon": "Company",
                            "cofm value": 8.95,
                            "Description": "Infantry Co (M2A2)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0.66, "reserve":0, "threat":0.0, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                        },
                        {
                            "Entity": "B 1-6",
                            "echelon": "Company",
                            "cofm value": 8.95,
                            "Description": "Infantry Co (M2A2)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0.66, "reserve":0, "threat":0.0, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                        },
                        {
                            "Entity": "C 1-6",
                            "echelon": "Company",
                            "cofm value": 7.7,
                            "Description": "Tank Co (M1A1)",
```

```
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0, "reserve":1.0, "threat":0, "end": 0},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                          }
                    ]
                },
                {
                    "Entity": "1-35 AR",
                    "echelon": "Battalion",
                    "cofm value": 37.24,
                    "Description": "IN Combined Arms BN (M1A2)",
                    "children": [
                        {
                            "Entity": "A 1-35",
                            "echelon": "Company",
                            "cofm value": 7.8,
                            "Description": "Tank Co (M1A2)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0.66, "reserve":0, "threat":0.0, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                          },
                        {
                            "Entity": "B 1-35",
                            "echelon": "Company",
                            "cofm value": 7.8,
                            "Description": "Tank Co (M1A2)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0.66, "reserve":0, "threat":0.0, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                          },
                        {
                            "Entity": "C 1-35",
                            "echelon": "Company",
                            "cofm value": 8.95,
                            "Description": "Infantry Co (M2A2)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0.66, "reserve":0, "threat":0.0, "end": 0.33},
                            "mission_complete":0,
```

```
                            "disabled":0,
                            "children": []                          }
                    ]
                },
                {
                    "Entity": "5-20 IN 1/2 IN",
                    "echelon": "Battalion",
                    "cofm value": 33.45,
                    "Description": "Infantry Bn (SBCT)",
                    "children": [
                        {
                            "Entity": "A 5-20",
                            "echelon": "Company",
                            "cofm value": 8.57,
                            "Description": "Infantry Co (SBCT)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0.66, "reserve":0, "threat":0.0, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                          },
                        {
                            "Entity": "B 5-20",
                            "echelon": "Company",
                            "cofm value": 8.57,
                            "Description": "Infantry Co (SBCT)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0.66, "reserve":0, "threat":0.0, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                          },
                        {
                            "Entity": "C 5-20",
                            "echelon": "Company",
                            "cofm value": 8.57,
                            "Description": "Infantry Co (SBCT)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0, "aoi":0,
"obj":0.33, "reserve":0, "threat":0.33, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                          }
                    ]
                },
```

```
                {
                    "Entity": "1-1 CAV",
                    "echelon": "Battalion",
                    "cofm value": 39.88,
                    "Description": "Cav Sqdn (M1A2)",
                    "children": [
                        {
                            "Entity": "A 1-1",
                            "echelon": "Troop",
                            "cofm value": 8.55,
                            "Description": "Cav Trp (ABCT)",
                            "initial location": "(5,3)",
                            "decision_weights":{"clear":0.33,
"aoi":0, "obj":0, "reserve":0, "threat":0.33, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                        },
                        {
                            "Entity": "B 1-1",
                            "echelon": "Troop",
                            "cofm value": 8.55,
                            "Description": "Cav Trp (ABCT)",
                            "initial location": "(5,3)",
                            "decision_weights":{"clear":0.33,
"aoi":0, "obj":0, "reserve":0, "threat":0.33, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                        },
                        {
                            "Entity": "C 1-1",
                            "echelon": "Troop",
                            "cofm value": 8.55,
                            "Description": "Cav Trp (ABCT)",
                            "initial location": "(5,3)",
                            "decision_weights":{"clear":0.33,
"aoi":0, "obj":0, "reserve":0, "threat":0.33, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                        },
                        {
                            "Entity": "D 1-1",
                            "echelon": "Company",
                            "cofm value": 7.8,
                            "Description": "Tank Co (M1A2)",
                            "initial location": "(5,3)",
```

```
                            "decision_weights":{"clear":0.33,
"aoi":0, "obj":0, "reserve":0, "threat":0.33, "end": 0.33},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                        }
                    ]
                },
                {
                    "Entity": "4-27 FA",
                    "echelon": "Battalion",
                    "cofm value": 28.13,
                    "Description": "FA Bn (M109A6)",
                    "children": [
                        {
                            "Entity": "A 4-27",
                            "echelon": "Battery",
                            "cofm value": 5.87,
                            "Description": "FA Btry (M109A6)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0,
"aoi":1.0, "obj":0, "reserve":0, "threat":0, "end": 0.5},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                        },
                        {
                            "Entity": "B 4-27",
                            "echelon": "Battery",
                            "cofm value": 5.87,
                            "Description": "FA Btry (M109A6)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0,
"aoi":1.0, "obj":0, "reserve":0, "threat":0, "end": 0.5},
                            "mission_complete":0,
                            "disabled":0,
                            "children": []                        },
                        {
                            "Entity": "C 4-27",
                            "echelon": "Battery",
                            "cofm value": 5.87,
                            "Description": "FA Btry (M109A6)",
                            "initial location": "(3,1)",
                            "decision_weights":{"clear":0,
"aoi":1.0, "obj":0, "reserve":0, "threat":0, "end": 0.5},
                            "mission_complete":0,
                            "disabled":0,
```

```
                                "children": []                          }
                            ]
                        }

                    ]
                }
            ]
        }
```

Appendix B. OPFOR Units

```
{
    "Entity": "71 SHC",
    "echelon": "Division",
    "cofm value": 1681.83,
    "Description": "Army Division",
    "children": [

                {
                    "Entity": "7152 715",
                    "echelon": "Battalion",
                    "cofm value": 42.9,
                    "Description": "Infantry Bn (BTR-80)",
                    "children": [
                        {
                            "Entity": "1 7152",
                            "echelon": "Company",
                            "cofm value": 8.67,
                            "Description": "Infantry Co (BTR-80)",
                            "initial location": "(10,5)",
                            "children": []
                        },
                        {
                            "Entity": "2 7152",
                            "echelon": "Company",
                            "cofm value": 8.67,
                            "Description": "Infantry Co (BTR-80)",
                            "initial location": "(10,5)",
                            "children": []
                        },
                        {
                            "Entity": "3 7152",
                            "echelon": "Company",
                            "cofm value": 8.67,
                            "Description": "Infantry Co (BTR-80)",
                            "initial location": "(10,5)",
                            "children": []
                        },
                        {
                            "Entity": "4 7152",
                            "echelon": "Company",
                            "cofm value": 5.22,
                            "Description": "Tank Co (T-72A)",
                            "initial location": "(10,5)",
                            "children": []
                        }
```

```
    ]
},
{
    "Entity": "7157 715",
    "echelon": "Battalion",
    "cofm value": 17.74,
    "Description": "2* Recon Co (BMP)",
    "children": [
        {
            "Entity": "2 7157",
            "echelon": "Company",
            "cofm value": 8.87,
            "Description": "Recon Co (BMP)",
            "initial location": "(3,1)",
            "children": []
        }
    ]
},
{
    "Entity": "7158 715",
    "echelon": "Battalion",
    "cofm value": 21.88,
    "Description": "FA Bn (2S1)",
    "children": [
        {
            "Entity": "1 7158",
            "echelon": "Company",
            "cofm value": 4.41,
            "Description": "FA Btry (2S1)",
            "initial location": "(12,1)",
            "children": []
        },
        {
            "Entity": "2 7158",
            "echelon": "Company",
            "cofm value": 4.41,
            "Description": "FA Btry (2S1)",
            "initial location": "(12,1)",
            "children": []
        },
        {
            "Entity": "3 7158",
            "echelon": "Company",
            "cofm value": 4.41,
            "Description": "FA Btry (2S1)",
```

```
                            "initial location":"(12,1)",
                            "children": []
                        },
                        {
                            "Entity": "4 7158",
                            "echelon": "Company",
                            "cofm value": 4.03,
                            "Description": "FA Btry (9A51/PRIMA/BM-
21)",
                            "initial location": "(12,1)",
                            "children": []
                        }
                    ]
                }

            ]
        }
    ]
}
```

Appendix C. Combat Module

In [1]:
```
from cofm_demo import cofm_calculator, calculate_total_forces, compute_pct_losses
```

COFM CALCULATOR FUNCTION

In order to run in an optimization routine the **cofm_calculator()** is executed as a single call. The function assumes that the default inventory files are named "FriendlyForces.csv","EnemyForces.csv" and stored in a local "../data/" directory so these arguments can be dropped from the call.

Inputs: the user specifies the friendly and enemy detachments, the friendly and enemy postures.
Outputs: a dictionary of the total force equivalences for each detachment and a dictionary of percent losses.

In [3]:
```
#dictionary of friendly forces {type: [count, strength]}
friendly_force = {"FA Bn (M777)": [3, 0.8],
                  "Armored Bde (ABCT) (M1A1)": [5, 0.9]
                  }

#dictionary of enemy forces {type: [count, strength]}
enemy_force = {"Infantry Co (BMP-3)": [2, 1.0],
               "Tank Bn (T-80U)": [3, 0.7]
               }

#must be assigned one of "deliberate attack", "hasty attack", "deliberate defense", "hasty defense", or "meeting"
posture_dict = {"friendly":"hasty attack",
                "enemy":"deliberate defense"
                }

#cofm calculator returns total force equivalent for parties and percent losses
total_forces, pct_losses = cofm_calculator(friendly_force,
                                           enemy_force,
                                           posture_dict["friendly"],
                                           posture_dict["enemy"],
                                           friendly_force_file = "FriendlyForces.csv",
                                           enemy_force_file = "EnemyForces.csv")

#print out results
print(total_forces)
```

Appendix D. Doctrine Module

```
# Rule that protects assets
IF there is a Main Command Post, Tactical Command Post or a Rear
       Command Post AND
       there are no AT or ADA forces assigned to it
       THEN decrease overall utility by 10% (multiply by .9).

# Rule that encourages setting a reserve aside
IF all units with force equivalent =/= NaN are assigned AND
       a unit from 2 echelons down is not held in reserve
       THEN decrease utility of main effort by 10%

# Rule that puts the main effort first
IF no units are assigned to main effort
       THEN decrease overall utility by 50%

# Rule that increases chance that the optimal unit is chosen
IF hostile forces exist at the OBJ OR ABF, AND
       the enemy is not in the air AND
       the unit assigned to it is not a combat brigade unit (SBCT,
       IBCT, ABCT, mounted cavalry)
       THEN decrease utility of unit by 20%

# Rule that assigns AntiTank artillery to PAA
IF all PAAs in this phase have >=1 AT FA unit assigned,
       THEN utility is high/increased, ELSE low/decreased.

# Rule that assigns recon prior to movement to an OBJ
IF unit is assigned to an objective OBJ OR ABF AND
       no form of recon is assigned #this could be a UAV, forward
       team, cavalry, even a BCT - may want to ignore this
       THEN the utility of that unit is decreased by 10%

# Rule that encourages Key_Terrain to be defended
IF a location is Key_Terrain AND
       a friendly unit is occupying that location
       THEN the utility of that unit is increased by 5%
```

Appendix E. Repository Structure and Experiments

E.1 Repository Structure

Table E-1. Repository structure.

Filename	Type	Functions
/core	Subdirectory	. . .
/policies	Subdirectory	. . .
forces/*.json	OOB scenario order of battle BLUFOR (e.g., 21AD.json) & OPFOR/Red (e.g., SSB715.json)	Encodes initial positions and decision_weights per company/troop/battery
visualization/*.py	Maps	Turn maps (with phase band lines), per unit top k heatmaps, plotting helpers
notebooks/*_OTT.py	Exported run scripts	Exported run scripts for adjacent/phase/sim modes with reporting

Table E-2. Repository structure /core.

Filename	Type	Functions
game.py	Simulation engine (WarGame)	initialization, turn loop, losses, movement, salience recomputation, adjacency construction, dynamic source updates
graph.py	Graph model	nodes/edges, neighbors, reachability, spreading activation, city-block Gaussian
node.py	Node data model	unit containers, 6-layer utilities, static source flags, dynamic flags (e.g., secured_objective)
edge.py	Data structure	Edge data structure with weights

Table E-3. Repository structure policies/.

Filename	Type	Function
greedy_agent_rules.py	Task assignment policies	original greedy, generalized assignment, turn glue
compute_losses.py	Force-ratio loss models	produces percentage losses (friendly/enemy) by posture
utility_function.py	Node utilities	Linear combination of node utilities with per-unit decision weights (threat, area of interest, objective, clear, reserve, end).

E.2 Experiments

E.2.1 Scenarios and Modes

Turnbook scripts "*_OTT.py" can be found under "notebooks/". Each script runs one policy in either simultaneous or phase mode, logs state (state_metrics.csv), and emits visualizations (e.g., turn overlays, feature trajectories).

E.2.2 Connectivity

To model simultaneous greedy assignment, the following modules are required: Greedy_game_simultaneous_OTT2.py, Margin_Greedy_game_simultaneous_OTT2.py, Loss_Greedy_game_simultaneous_OTT2.py, Margin_Loss_Greedy_game_simultaneous_OTT2.py, Random_game_simultaneous_OTT2.py, Uniform_Random_game_simultaneous_OTT2.py

To model phased greedy assignment, the following modules are required: Greedy_game_phase_OTT2.py, Margin_Greedy_game_phase_OTT2.py ,Loss_Greedy_game_phase_OTT2.py, Margin_Loss_Greedy_game_phase_OTT2.py, Random_game_phase_OTT2.py, Uniform_Random_game_phase_OTT2.py

E.2.3 Batch Runners

Full epoch runner: scripts/run_epoch.py — executes all six policy variants in both modes, collects session folders, and generates five-way comparison plots.

Threshold comparison: scripts/run_epoch_greedy_thresholds.py — runs the four greedy variants with and without the 40% cap in both modes, then plots grouped results.

E.2.4 Metrics and Plots

The engine logs the following, per-turn metrics in core/game.py:: log_state_features() including:

Force ratio (BLUFOR/filtered OPFOR): Red force (OPFOR) is limited to nodes in connected components that are reachable from any BLUFOR unit (OPFOR is passive in this simulation).

Objectives secured: Points captured by layer and in total, a measure of objectives "secured" is logged as a decreasing total (starting at the maximum possible points and counting down as points are secured).

BLUFOR centroid position in map Y units, with y-axis limits pinned to the map's min/max row for consistent interpretation.

Remaining force value (decreasing): Starts at initial force values and subtracts initial force of units that are now disabled (tracked separately for BLUFOR and BLUFOR-reachable OPFOR).

Remaining disabled counts (decreasing): Starts at initial unit counts and subtracts currently disabled units (OPFOR counts restricted to BLUFOR-reachable subgraphs).

There are plotting helpers in visualization/feature_plots.py render multi-run overlays; visualization/plot_rand_greedy.py that provide five-way summaries for phase and simultaneous runs.

E.2.5 Run Instructions (Examples)

Simultaneous greedy-Q:
PYTHONPATH=
src python UT_greedy_game/notebooks/Greedy_game_simultaneous_OTT2.py

Phase marginal net-loss greedy-Q:
PYTHONPATH=
src python UT_greedy_game/notebooks/Margin_Loss_Greedy_game_phase_OTT2.py

Full epoch:
PYTHONPATH=src python UT_greedy_game/scripts/run_epoch.py

Greedy cap vs no-cap suite:
PYTHONPATH=
src python UT_greedy_game/scripts/run_epoch_greedy_thresholds.py

E.2.6 Limitations and Assumptions

The 40% disabled cap is implemented at decision time (utility capping) and does not alter the underlying engagement loss model. This provides a practical approximation for "no incremental benefit beyond disable" without altering unit-health plumbing inside the inner loops.

OPFOR (red) is passive and cannot move or initiate attacks; OPFOR in disconnected components does not contribute to the effective OPFOR denominator for force-ratio calculations.

List of Symbols, Abbreviations, and Acronyms

2/1AD	2nd Brigade Armored Division
ABCT	Armored Brigade Combat Team
AOI	Area of Interest
AOR	Area of Responsibility
ARL	Army Research Laboratory
BLUFOR	Blue Forces (Friendly Forces)
COFM	Correlation of Forces Matrix
COA	Course of Action
COA-GPT	LLM Used to Generate COAs
DEVCOM	U.S. Army Combat Capabilities Development Command
END	End-Point of Mission
GCAA	Greedy Coalition Auction Algorithm
GPT	Generative Pre-Trained Transformer
IBCT	Infantry Brigade Combat Team
ITS	Integrated Technology System
JSON	Javascript Object Notation
LLM	Large Language Model
MDMP	Military Decision-Making Process
MILP	Mixed Integer Linear Programming
ML	Machine Learning
OBJ	Objective
OPFOR	Opposition Forces (Red Forces)
OPORD	Operation Order
OTT	Operation Tropic Tortoise
PAA	Position Area for Artillery
PL	Phase Line

RES	Reserve
SBCT	Stryker Brigade Combat Team
SC3	Scalable Cross-Echelon Command and Control

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
1 (PDF)	DEVCOM ARL FCDD RLB CI TECH LIB