ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ανίχνευση βέλτιστης λύσης σε προβλήματα με χρήση υπολογιστικής σμήνους

ΤΖΑΝΙΔΑΚΗΣ ΒΑΣΙΛΕΙΟΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Κολομβάτσος Κωνσταντίνος
Επίκουρος Καθηγητής

Λαμία έτος 2026

ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ανίχνευση βέλτιστης λύσης σε προβλήματα με χρήση υπολογιστικής σμήνους

ΤΖΑΝΙΔΑΚΗΣ ΒΑΣΙΛΕΙΟΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Κολομβάτσος Κωνσταντίνος
Επίκουρος Καθηγητής

Λαμία έτος 2026

UNIVERSITY OF
THESSALY

SCHOOL OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE & TELECOMMUNICATIONS

Finding optimal solutions to problems using swarm intelligence

TZANIDAKIS VASILEIOS

FINAL THESIS

ADVISOR

Kolomvatsos Konstantinos

Lamia year 2026

«Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

*1. Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

*2. Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια

4. Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.

Ημερομηνία: ……/…../20……

Ο – Η Δηλ.

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση
του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων
σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.»

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία πραγματεύεται τη βελτιστοποίηση της ανάθεσης αποστολών έκτακτης ανάγκης και πολιτικής προστασίας, καθώς και τη διαχείριση ενέργειας σε ετερογενή συστήματα σμηνών. Προς την κατεύθυνση αυτή, η έρευνα εισάγει δύο αλληλένδετες συνεισφορές για την εύρεση βέλτιστων λύσεων. Πρώτον, αναπτύσσεται ένα πλαίσιο ανάθεσης εργασιών σε επίπεδο κλάσης, στο οποίο οι εκτελεστικοί πράκτορες οργανώνονται σε έξι τυποποιημένες κλάσεις σμήνους, καθεμία από τις οποίες ορίζεται από ένα διακριτό διάνυσμα δυνατοτήτων. Καθοριστικό στοιχείο είναι ότι, στο πλαίσιο αυτής της ιεραρχικής δομής, κάθε μεμονωμένος επιχειρησιακός 'πράκτορας' (agent) προσεγγίζεται και αντιμετωπίζεται ως ένα ολόκληρο, συμπαγές υπο-σμήνος (sub-swarm). Αυτά τα σμήνη-πράκτορες χαρακτηρίζονται από υψηλή ετερογένεια, έχοντας διαφορετικά λειτουργικά χαρακτηριστικά, ικανότητες και διακριτά ενεργειακά προφίλ μεταξύ τους. Η ανάθεση αποστολών πραγματοποιείται μέσω μιας συνάρτησης πολυκριτηριακής βαθμολόγησης καταλληλόλητας, η οποία προσαρμόζει δυναμικά τα βάρη της αντιστοίχισης δυνατοτήτων, της εγγύτητας, της αντοχής και της προσαρμοστικότητας ανάλογα με τη δυσκολία της αποστολής και την ενεργειακή κατάσταση του πράκτορα, επιτρέποντας έτσι την ευαίσθητη στο εκάστοτε πλαίσιο ανάθεση χωρίς την ανάγκη απαρίθμησης μεμονωμένων πρακτόρων. Δεύτερον, προτείνεται ένας αλγόριθμος ουράς φόρτισης βασισμένος σε προτεραιότητες Optimal (OPT) για τη διαχείριση των ουρών στους σταθμούς φόρτισης που περιορίζουν τη λειτουργική συνέχεια του σμήνους. Ο OPT υπολογίζει ένα σύνθετο κόστος φόρτισης για κάθε πράκτορα που βρίσκεται στην ουρά, χρησιμοποιώντας τρεις δυναμικά σταθμισμένους όρους: έναν παράγοντα κινδύνου επείγοντος που προκύπτει από το ενεργειακό περιθώριο του πράκτορα κάτω από το όριο ασφαλείας του, μια τιμή ικανότητας που αντανακλά την αναμενόμενη συνεισφορά του στο τρέχον σύνολο εκκρεμών αποστολών και ένα καθολικό σήμα επείγοντος ανάλογο με τον αριθμό των μη ανατεθειμένων αποστολών. Το προτεινόμενο πλαίσιο αξιολογείται μέσω μιας ελεγχόμενης μελέτης προσομοίωσης αναζήτησης πλέγματος (grid-search) σε ένα διακριτό δισδιάστατο περιβάλλον, συγκρίνοντας τον OPT με τις κλασικές μεθόδους αναφοράς FIFO (First-In First-Out) και LIFO (Last-In First-Out) σε 32 διαμορφώσεις παραμέτρων που μεταβάλλουν το μέγεθος του πληθυσμού των εκτελεστικών πρακτόρων, τον αριθμό των ανιχνευτών (scouts), τον στόχο αποστολών και τον αριθμό των σταθμών φόρτισης. Τα αποτελέσματα καταδεικνύουν ότι ο OPT επιτυγχάνει υψηλότερη αποδοτικότητα ανάθεσης, χαμηλότερα ποσοστά αποτυχίας, μειωμένους μέσους χρόνους αναμονής στις ουρές και ανώτερους λόγους παραγωγικότητας σε σύγκριση με αμφότερες τις μεθόδους αναφοράς, επιβεβαιώνοντας ότι ο προγραμματισμός φόρτισης με επίγνωση των αποστολών και προσαρμογή στην κατάσταση του συστήματος οδηγεί σε μετρήσιμα καλύτερη απόδοση του σμήνους και συμβάλλει στην ανάπτυξη προσαρμοστικών, αυτοβελτιστοποιούμενων συστημάτων σμηνών.

ABSTRACT

This Thesis addresses the optimization of task allocation for emergency response and civil protection missions, as well as energy management in heterogeneous swarm systems. Towards this direction, the research introduces two interconnected contributions to achieve optimal solutions. First, a class-level task allocation framework is developed in which executor agents are organized into six typed swarm classes, each defined by a distinct capability vector. Crucially, within this multi-level architecture, each individual operational 'agent' is conceptualized and treated as an entire, cohesive sub-swarm; these swarm-agents possess highly heterogeneous characteristics, structural performance variations, and distinct energy profiles from one another.Mission assignment is performed by a multi-criteria suitability scoring function that dynamically weights capability matching, proximity, endurance, and adaptability according to mission difficulty and agent energy state, enabling context-sensitive allocation without enumerating individual agents. Second, a priority-based charging queue algorithm Optimal (OPT) is proposed for managing the charging station queues that constrain swarm operational continuity. OPT computes a composite charging cost for each queued agent from three dynamically weighted terms: an urgency risk factor derived from the agent's energy margin below its safety threshold, a capability value reflecting the agent's expected contribution to the current mission backlog, and a global urgency signal proportional to the number of unassigned missions. The proposed framework is evaluated through a controlled grid-search simulation study conducted in a discrete two-dimensional environment, comparing OPT against the classical baselines FIFO (First-In First-Out) and LIFO (Last-In First-Out) across 32 parameter configurations that vary the executor population size, scout count, mission target, and number of charging stations. Results demonstrate that OPT achieves higher assignment efficiency, lower failure rates, reduced average queue wait times, and superior productivity ratios compared to both baselines, confirming that mission-aware, state-adaptive charging scheduling produces measurably better swarm performance than order-based alternatives and contributes to the development of adaptive, self-optimizing swarm systems.

Contents

Chapter 1

Introduction

1.1 Background & Motivation

The increasing deployment of autonomous multi-agent systems in real-world scenarios, ranging from search-and-rescue operations and infrastructure inspection to environmental monitoring and security patrol, has created a growing demand for intelligent coordination frameworks that can operate reliably under uncertainty, resource constraints, and dynamically changing conditions. Swarm intelligence, drawing inspiration from the collective behaviour of biological organisms such as ant colonies, bee swarms, and bird flocks, offers a compelling paradigm for building such systems: large numbers of relatively simple agents, each acting on local information and rules, can collectively exhibit sophisticated and adaptive behaviour without centralized control (Kennedy, 2006).

In practice, the agents composing a swarm are rarely identical. Heterogeneous swarm systems, where different agent types possess distinct sensing ranges, mobility profiles, endurance levels, and task-specific capabilities, are becoming the norm rather than the exception. A heterogeneous swarm can perform a wider range of tasks simultaneously and adapt more flexibly to complex, multi-requirement missions, but this diversity also introduces a fundamental coordination challenge: how should missions be allocated across agents with differing capabilities, and how should the shared energy infrastructure charging stations be managed so that operational continuity is maintained? These two problems, task allocation and energy-aware queue scheduling, are deeply coupled. A swarm that exhausts its battery while travelling to a high-priority mission cannot complete that mission, and a charging queue that serves swarms in the wrong order can deprive the swarm of its most capable members precisely when they are needed most.

This Thesis is motivated by the observation that existing work tends to address these two problems in isolation. Task allocation literature focuses on matching agents to missions but often assumes unlimited or homogeneous energy availability. Energy management literature addresses battery scheduling but seldom incorporates mission urgency or agent capability into the queue ordering decision. The present work proposes and evaluates a unified framework that treats both allocation and charging as parts of a single, adaptive optimization problem, operating at the level of swarm classes to maintain scalability while preserving sensitivity to individual agent state and system-level operational pressure.

1.2 Problem description

This Thesis considers a heterogeneous swarm operating on a bounded two-dimensional map. The swarm is composed of two functional tiers. Scout agents patrol the environment autonomously and are responsible for discovering missions. Without scouts, no mission can enter the assignment pipeline regardless of executor availability. Executor agents are organized into six typed classes where each characterized by a six-dimensional capability vector encoding proficiency across the corresponding skill domains. Missions arrive stochastically throughout the simulation, each characterized by its own distinct

requirements and operational parameters. Central to each mission is a multi-dimensional capability type vector, which serves as a profile defining the exact proportion of skills required to resolve the emergency. Concurrently, each mission can be defined by independent attributes such as its difficulty level, intensity, and spatial radius. By computing the cosine similarity between the mission's type vector and an individual executor agent's capability vector, the system dynamically evaluates the executor agent's expertise alignment. Thus mathematical match between the mission and the executor agent is utilized to calculate a comprehensive suitability score for optimal task allocation. Additionally a mission can only be assigned to an executor after a scout has detected it within sensor range.

The task allocation problem is defined as follows: given a set of discovered, unassigned missions and a set of available executors, select the best-matching agent or agents for each mission so as to maximize the probability of successful completion. The suitability of an executor for a mission is computed by a dynamic scoring function that combines the cosine similarity between the agent's capability vector and the mission's type vector, the normalized Euclidean distance between agent and mission, the ratio of mission difficulty to agent endurance, and an adaptation potential term. The weights of these four components are not fixed. They shift in response to mission difficulty and agent energy level, so that capability matching is emphasized for hard missions and endurance is emphasized for depleted agents.

The energy management problem is defined as follows: given one or more charging stations, each with limited simultaneous charging capacity, and a queue of agents waiting to recharge, decide which agent to serve next at each time step. The proposed Optimal (OPT) algorithm computes a composite charging cost for every queued agent and always promotes the agent with the highest cost to the charger. The charging cost is a dynamically weighted combination of an urgency risk factor (how critically low the agent's energy is relative to its safety threshold), a capability value (how much the agent could contribute to the current mission backlog), and a global urgency signal (how many missions are currently unassigned relative to the total agent count). Both problems are evaluated in a controlled simulation study comparing OPT against FIFO and LIFO baselines across a systematic grid of configuration parameters.

1.3 Challenges in swarm optimization

Optimizing swarm behaviour in heterogeneous, energy-constrained environments introduces several interrelated challenges that are absent from simpler, homogeneous, or unlimited-resource settings. The most fundamental is the scalability challenge: the number of possible agent-to-mission assignments grows combinatorially with swarm size and mission count, making exhaustive search computationally infeasible. The class-level allocation strategy adopted in this Thesis addresses this by grouping agents with similar capabilities, reducing the search space while preserving the emergent properties of swarm behaviour (Valentini et al., 2022).

A second challenge is dynamic non-stationarity. The state of the environment changes continuously: new missions are spawned stochastically, some missions spread and intensify if left unattended, agent energy levels fluctuate as work is performed, and the composition of the charging queue shifts at every step. Any allocation or scheduling policy that relies on a static evaluation computing priorities once and then executing a fixed order will quickly become suboptimal as the system state drifts. The OPT algorithm addresses this by re-evaluating every queued agent's charging cost at each simulation step, producing a dynamic ranking that responds in real time to changes in mission availability, agent energy, and queue congestion.

A third challenge is the coupling between task allocation and energy management. An agent assigned to a difficult, distant mission may deplete its battery before completing the task, effectively wasting the assignment. Conversely, an agent that recharges opportunistically at the right moment may be available for a mission that no other agent can handle. This coupling means that decisions made in the charging queue directly affect assignment quality and mission success rates. Capturing this interdependence requires a unified model in which both the suitability function and the charging cost function share information about agent state and mission backlog, which is a design principle central to the framework developed in this Thesis.

Finally, evaluation presents its own challenge. Because the three queue disciplines (FIFO, LIFO, OPT) must be compared fairly, all three simulations must face identical mission sequences, identical initial swarm configurations, and identical charging infrastructure. Achieving this reproducibility in a stochastic simulation requires careful use of shared random seeds and pre-generated mission blueprints, as implemented in the experimental framework. Without this control, observed performance differences could be artifacts of random variation rather than genuine algorithmic advantages.

1.4 Objectives of the Thesis

The primary objective of this Thesis is to design, implement, and evaluate an optimization framework for heterogeneous swarm systems that addresses mission allocation and energy management as a joint problem. In pursuit of this goal, the work is structured around four specific objectives.

The first objective is to develop a class-level task allocation mechanism for heterogeneous swarms. Crucially, within this architectural framework, a high-level system abstraction is introduced: each individual operational 'agent' tracked within the system is conceptualized and modelled as an entire, cohesive sub-swarm, displaying distinct capability vectors and performance constraints from its peers. Rather than evaluating every individual unit for each incoming mission, the framework organizes these executor sub-swarms into six typed classes and selects the best-matching entity through a multi-criteria suitability scoring function. The scoring function must be sensitive to capability alignment, spatial proximity, sub-swarm endurance relative to mission difficulty, and adaptation potential, and its criteria weights must adjust dynamically in response to the real-time mission and sub-swarm energy states.

The second objective is to design a mission-aware, state-adaptive charging priority algorithm (OPT) for managing the charging station queues. OPT must go beyond simple order-based rules by incorporating agent energy urgency, agent capability value relative to the active mission backlog, and system-level congestion and mission pressure into a single, dynamically weighted cost function. The algorithm must re-evaluate priorities at every simulation step to respond to real-time changes in the swarm's operational state.

The third objective is to implement a reproducible, high-fidelity simulation environment in which the proposed algorithms can be evaluated against classical baselines. The simulation must ensure that FIFO, LIFO, and OPT face identical conditions in every run by sharing the same random seed, pre-generated mission blueprints, and spawn schedule. The environment must capture the key dynamics of a real swarm deployment: stochastic mission arrivals, energy consumption during movement and work, adaptive capability learning after mission completion, and limited charging infrastructure.

The fourth objective is to conduct a systematic grid-search evaluation across a range of configuration parameters including executor population sizes, scout counts, mission targets, and station counts, and to analyse the results to identify conditions under which OPT provides the most significant advantages over the baselines, as well as conditions

under which the gains are marginal. This analysis is intended to provide practical guidance for the deployment of adaptive charging strategies in real heterogeneous swarm systems.

1.5 Contributions of the Thesis

This Thesis makes the following contributions to the field of swarm intelligence and multi-agent optimization.

- **First**, a dynamic multi-criteria suitability scoring function for class-level mission allocation in heterogeneous swarms is proposed and formalized. A foundational aspect of this contribution is the structural system abstraction, wherein each individual operational 'agent' is conceptualized and managed as an entire, cohesive sub-swarm, displaying distinct capability vectors, performance limits, and behavioural traits from its peers. The matching function successfully combines cosine capability matching, normalized proximity, endurance ratio, and adaptation potential into a single score, with weights that shift dynamically as a function of mission difficulty and sub-swarm energy depletion. This design enables context-sensitive allocation without requiring centralized optimization or exhaustive per-unit evaluation.
- **Second**, a novel priority-based charging queue algorithm (OPT) is designed and implemented. OPT introduces a mission-aware, state-adaptive queue ordering policy in which agent priorities are re-evaluated at every simulation step based on energy urgency, capability value relative to the current mission backlog, and a global urgency signal. The three weight parameters are dynamically computed from system-level state variables (energy stress, mission pressure, and station congestion), normalizing them to ensure interpretability and stability. Four algorithmic enhancements haven been incorporated into OPT: Dynamic Power Allocation, charging at 3.0 versus 2.5 energy units per step for baselines. Starvation prevention via a waiting-time bonus of 0.05 per step in queue. Recalibrated dynamic weights with alpha base 0.10, beta base 0.50, gamma max-factor of 4.0 for neglected spreadable missions. This represents a departure from classical order-based queue disciplines and constitutes the core algorithmic contribution of the Thesis.
- **Third**, a complete, reproducible simulation framework is implemented in Python that enables controlled comparison of charging strategies in hierarchical, multi-swarm systems. The framework supports arbitrary fleet compositions, stochastic mission spawning with shared seeds across compared modes, six executor sub-swarm types with two performance variants each, adaptive capability learning after mission completion, and structured CSV export of all performance metrics. The grid-search harness covers 32 distinct parameter configurations, making the experimental study the most systematic evaluation of charging queue disciplines in a heterogeneous swarm context reported to date.
- **Fourth**, a weighted composite performance metric is defined that aggregates assignment efficiency, productivity ratio, average energy level, queue wait time, failure rate, and mission completion step into a single, normalized ranking score. This composite enables fair, multi-dimensional comparison of the three queue disciplines across all experimental configurations and provides a reusable evaluation methodology for future work in this area.

1.6 Thesis organization

The remainder of this Thesis is organized into eight chapters providing a logical progression from theory to empirical validation: Chapter 2 (Swarm Intelligence) reviews collective behaviour, outlines metaheuristics like PSO, ACO, and ABC, and establishes the abstraction where individual agents function as autonomous sub-swarms; Chapter 3 (Task Allocation in Heterogeneous Swarms) examines multi-agent architectures, allocation strategies, and infrastructure constraints; Chapter 4 (Proposed Optimization Framework) formalizes the class-level task allocation, capability vectors, and suitability scoring function; Chapter 5 (Energy Management and Charging Strategy) describes the infrastructure layer and the priority-based Optimal (OPT) charging queue algorithm; Chapter 6 (Experimental Setup, Simulation & Results) establishes the Python testing pipeline and presents empirical data and baseline comparisons across 32 configurations; Chapter 7 (Discussion & Interpretation) contextualizes the empirical outcomes and outlines modelling limitations ; and Chapter 8 (Future Work & Conclusion) summarizes the structural contributions and outlines directions for subsequent investigation.

Chapter 2
Swarm intelligence

2.1 Introduction to swarm intelligence

Swarm Intelligence (SI) is a computational and behavioral paradigm inspired by the collective behavior observed in nature. It refers to the ability of individual simple agents, interacting locally with one another and their environment, to produce coherent and intelligent global behavior without centralized control. This concept has gained significant attention in recent years due to its ability to successfully handle complex, stochastic, and dynamic problems (Kennedy, 2006; Nguyen, 2024).

SI has several advantages, with scalability being one of the most important. As the number of agents in the system increases, system performance and adaptability often improve without proportional increases in computational or structural complexity (Wang et al., 2024). This makes swarm-based approaches particularly suitable for distributed problems, where agents must cooperate under limited communication or control. Additionally, swarm systems exhibit strong robustness, allowing them to continue functioning even when individual agents fail or unexpected environmental changes occur (Nguyen, 2024).

In computational contexts, SI methods such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and other collective strategies, have been widely used to solve complex problems efficiently by balancing exploration and exploitation in large search spaces. These methods evaluate multiple candidate solutions in parallel and adapt based on collective feedback, making them highly effective for dynamic optimization problems.

2.2 Biological inspiration & Key features

Biological systems provide the foundational inspiration for SI, demonstrating how complex collective behavior can emerge from simple individual actions. Observations of social organisms such as ants, bees, termites, birds and other species, reveal that coordination and efficiency can arise without centralized control, relying instead on local interactions and environmental feedback mechanisms. These natural systems serve as models for understanding how decentralized entities can solve complex problems through cooperation and adaptation.

A key concept derived from biological systems is stigmergy, a form of indirect communication in which agents interact through modifications of their environment. A well-known example is observed in ant colonies, where ants deposit pheromone trails that guide others towards food resources, reinforcing efficient paths over time. This mechanism enables distributed coordination and collective problem-solving without requiring explicit communication between individuals (Garnier, Gautrais, & Theraulaz, 2007).

Another fundamental property observed in biological swarms is self-organization, where structured and adaptive behavior emerges spontaneously from local interactions. These systems rely on simple behavioral rules, positive and negative feedback mechanisms, and multiple direct or stigmergic interactions among individual agents to achieve coordinated outcomes. Such properties allow biological swarms to adapt efficiently to dynamic and uncertain environments (Garnier, Gautrais, & Theraulaz, 2007).

Additionally, biological swarm systems exhibit strong decentralization and robustness. Within the system, no individual govern, instead global behavior emerges

from collective interactions. This results in systems that are highly resilient to failures, as the loss of individual agents does not significantly affect the overall performance. These characteristics have inspired the development of computational models and optimization algorithms that replicate similar adaptive and distributed behaviour in artificial systems (Garnier, Gautrais, & Theraulaz, 2007).

2.3 Optimization in swarm systems

SI has been widely adopted as an effective approach for solving optimization problems, particularly in complex, dynamic, and high-dimensional environments. Optimization in swarm systems is based on the idea that multiple agents collaboratively explore a solution space, sharing information and adapting their behaviour to converge toward optimal or near-optimal solutions.

Unlike traditional optimization techniques, which often rely on centralized control or deterministic procedures, swarm-based optimization methods are inherently decentralized and stochastic. Each agent represents a potential solution and updates its state based on local interactions and collective experience. This enables the system to balance exploration and exploitation, which is critical for avoiding local optimization and achieving efficient convergence (Kennedy, 2006; Nguyen, 2024).

Well-known swarm-based optimization techniques include Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO), both of which are inspired by natural collective behaviors. In PSO, agents iteratively adjust their positions in the solution space based on both their individual experience and the best-known positions of the swarm, enabling efficient convergence toward optimal solutions. In contrast, ACO simulates pheromone-based communication, where artificial agents probabilistically select paths based on accumulated experience, reinforcing more efficient solutions over time.

A key advantage of these approaches is their ability to operate effectively in complex search spaces where traditional methods struggle. By maintaining a population of candidate solutions, swarm-based algorithms can continuously adapt to changes and avoid premature convergence. This makes them particularly suitable for problems where the optimal solution may shift over time or where the search space is not well-defined.

Moreover, swarm optimization techniques are highly effective in multi-objective scenarios, where multiple criteria must be satisfied simultaneously. Instead of producing a single solution, these methods can approximate a set of optimal trade-offs, providing flexibility in decision-making processes. This characteristic is especially relevant in real-world applications involving resource allocation, scheduling, and adaptive system control.

2.3.1 Swarm Intelligence Algorithms of Research Interest

To fully understand how collective behaviour solves complex problems, it is essential to highlight the foundational metaheuristic algorithms that dominate the evolutionary optimization literature and represent key operational milestones:

- **Particle Swarm Optimization (PSO):** Formulated by modelling the highly coordinated socio-cognitive behaviour of bird flocks and fish schools, PSO treats candidate solutions as multi-dimensional particles navigating a continuous search space (Kennedy, 2006; Wang et al., 2024). Each particle dynamically updates its velocity vector and spatial trajectory at each iteration by compounding its personal historical best coordinates with the global best position discovered by the entire swarm entity (Kennedy, 2006). These continuous adjustments allow the swarm to rapidly converge toward high-utility mathematical optima (Nguyen, 2024).

- **Ant Colony Optimization (ACO):** Drawing directly from the biological foraging mechanisms of social ant colonies, ACO maps discrete optimization challenges onto graph-based frameworks (Garnier, Gautrais, & Theraulaz, 2007). Artificial agents traverse computational paths, depositing mathematical "pheromone trails" that evaporate over time. Shorter, more efficient routes receive more frequent agent traffic, leading to a probabilistic reinforcement of optimal pathways across structural networks (Garnier, Gautrais, & Theraulaz, 2007).
- **Artificial Bee Colony (ABC):** Inspired by the complex foraging routines of honeybees, the ABC algorithm segregates its processing agents into three distinct, specialized functional tiers: employed bees, onlooker bees, and scout bees (Wang et al., 2024). Employed bees evaluate and exploit known local food sources, onlooker bees probabilistically allocate more resources to high-yielding zones based on collective feedback, and scout bees abandon exhausted paths to perform global random exploration, preserving high algorithmic diversity in large search spaces (Nguyen, 2024; Wang et al., 2024).

2.3.2 The Operational Swarm Optimization Algorithm Used in This Thesis

While the above-mentioned classical metaheuristics (PSO, ACO, ABC) are universally deployed as abstract mathematical solvers to optimize static numerical functions, a definitive architectural divergence is established within the scope of this Thesis. This research does not utilize standard nature-inspired algorithms to compute a static task checklist. Instead, the foundational mechanics of swarm intelligence are embedded directly into the physical control, resource allocation, and charging infrastructure layer of a heterogeneous, multi-tiered robotic fleet.

A core structural innovation of the proposed framework is the implementation of a high-level system abstraction: each autonomous "agent" tracked and scheduled within the simulation environment does not represent a single, isolated physical robot. Rather, each individual agent is conceptualized, modelled, and treated as an entire, cohesive sub-swarm.

Consequently, the operational fleet functions as a hierarchical multi-swarm system where each distinct swarm-agent exhibits highly heterogeneous capability characteristics, specialized tool profiles, kinematic constraints, and battery behaviours compared to its peers. In this architecture, the concept of swarm optimization is handled natively by a custom, state-aware coordination layer designed to solve two deeply coupled operational challenges:

- **The Class-Level Task Allocation Framework:** Rather than running an iterative genetic or particle-filtering loop that scales poorly with fleet size, task allocation is optimized instantly via a dynamic, multi-criteria suitability scoring function. This function processes cosine capability matches, spatial Euclidean distances, and task endurance ratios, shifting its internal criteria weights in real time based on environmental crisis urgency.
- **The Optimal (OPT) Charging Queue Engine:** Infrastructure management is treated as a core component of the swarm optimization loop. The proposed OPT algorithm evaluates queue priority on the fly by balancing localized agent battery deficits, functional tool profile values against the active mission backlog, and global system pressure indicators.

By treating these heterogeneous sub-swarms as individual agents operating within a live, distributed computing network, the proposed framework establishes a robust, self-optimizing swarm mechanism. This tactical integration effectively bridges the gap

between theoretical swarm intelligence and the real-world operational survival constraints of heterogeneous multi-agent systems.

2.4 Applications of swarm intelligence

Swarm intelligence has been applied in a wide variety of domains due to its ability to provide flexible, scalable, and efficient solutions in complex environments. Its decentralized nature makes it particularly suitable for systems that require coordination among multiple entities without relying on a central controller.

One important application area is robotics and multi-agent systems, where swarm-based approaches enable coordination, cooperation, and distributed decision-making. Such systems are used in tasks including environmental monitoring, area exploration, and search-and-rescue operations, where agents must operate autonomously while contributing to a common objective (Nguyen, 2024).

In logistics and transportation, swarm intelligence techniques are used to optimize routing, scheduling, and resource distribution. These methods can efficiently handle large-scale and dynamic systems, improving overall performance while reducing operational costs.

Additionally, swarm intelligence has been widely applied in engineering and computational optimization problems, such as parameter tuning, feature selection, and system design. Its ability to explore large and complex solution spaces makes it particularly effective for solving non-linear and multi-objective problems (Wang et al., 2024).

Finally, swarm-based approaches are increasingly used in distributed artificial intelligence systems, where autonomous agents must collaborate under uncertainty and limited communication. These systems benefit from the robustness and adaptability of swarm intelligence, making them suitable for real-time and mission-critical applications.

Chapter 3

Task allocation in heterogeneous swarms

3.1 Swarm classes and capabilities

A foundational distinction in swarm systems is between homogeneous and heterogeneous configurations. In a homogeneous swarm, every agent shares the same sensors, actuators, and computing capabilities, for instance, a swarm of agents all equipped identically for environmental monitoring or search for operations. In contrast, heterogeneous swarms, are typically deployed for tasks that require different types of agents working together toward a common goal, each contributing distinct capabilities such as scouting, rescue, patrols (A et al, 2024).

With a closer view, heterogeneous swarm consists of agents with diverse capabilities that offer various advantages, some of them are, the ability to perform a wider range of tasks, higher task efficiency, and greater flexibility and adaptability in handling complex tasks. These advantages consequently make it well-suited for dynamic, multi-demand, real-world missions(Yue et al., 2025).

Specific scenarios arise, where it is necessary to perform several distinct types of tasks simultaneously, each with its own requirements. Given a heterogeneous swarm system, with different capabilities, a fundamental challenge comes to light as to how to assign different agents to different task types so the overall goal is accomplished (Debie et al., 2023).

From an architectural perspective, group structure in swarm systems can be categorized along several dimensions like, centralization versus decentralization, differentiation which implies homogeneous or heterogeneous swarm systems, communication structure, and the number of dimensions of operation (Debie et al., 2023). In practice, heterogeneous swarm systems, are frequently structured around functional roles. Starting with a baseline of homogeneous swarms, these frameworks can scale into heterogeneous swarms. For instance, introducing scouts and fire extinguishers creates a functional coupling through local sensing, resulting in an emergent coordination structure (Koifman et al., 2026).

3.2 Heterogeneous swarm architecture

The coordination architecture underpinning a heterogeneous swarm system directly shapes its scalability, robustness, and responsiveness. There are two principal types of swarm architecture, centralized and decentralized where in real-world scenarios, the swarms must contend with critical challenges including communication and control schemes, incorporating emergent properties, managing connectivity, and implementing collective decision functions (Ragab et al., 2021).

Centralized architecture offers the advantage of global optimality but introduce a single point of failure and communication bottlenecks on a scale. Many applications benefit from the use of multiple agents, but their scalability and applicability are fundamentally limited when relying on central control station. A step beyond the centralized approach increases system complex but can be easily addressed through swarm intelligence algorithms that define behavior on local interactions and environmental context (St-Onge et al., 2020).

Decentralized architectures from the other side address those limitations by distributing decision-making across agents. A multi-layered framework for fully

distributed and decentralized task allocation can be derived from interaction-based dynamics, built upon a baseline of self-organized homogeneous agents, without centralized coordination, to explore unknown environments and fulfil spatial tasks. Agents operate as a cohesive units without needing central control, the agents maintain a connected and flexible structure by inly interacting with nearby peers, to prevent swarm disconnection (Koifman et al., 2026).

The resilience advantages of decentralization are particularly relevant in hostile or unpredictable environments. System's reliability is boosted by using a large fleet of inexpensive, interchangeable units of different types, instead of a few complex ones. This decentralized deployment method, allows the integration of the system to be maintained when critical nodes are compromised, which greatly enhances its resilience. Smart decision networks let units track situations, in real-time and distribute tasks autonomously, removing the delays caused by a central headquarters (Liu et al., 2025).

3.3 Mission allocation strategies

Several distinct strategies have been proposed to solve the mission allocation problem in heterogeneous swarm systems. Many approaches, including market-based mechanisms, auction algorithms, linear programming, and genetic algorithms, have been also explored to distribute tasks in a way that minimizes total mission time or maximizes coverage area (Alqudsi & Makaraci, 2025).

Market-based and auction approaches model agents as self-interested bidders, competing for tasks. Market-based methods has attracted considerable notice, for solving the multi-agent task allocation problem and have been used in a variety of settings addressing both market structures and communication schemes (Quinton et al., 2023). An auction-based task allocation from the other side, the system evaluates actual path costs to assign tasks based on priority and order, while ensuring rapid re-tasking during dynamic changes (Galati et al., 2023).

Optimization-based strategies, view task assignment as a complex puzzle, where the goal is to find the best possible setup from many options. By using mathematical models and algorithms, like genetic programming or Particle Swarm Optimization (PSO), these systems aim to balance workloads and save time or energy, even in dynamic environments (A et al., 2024).

Bio-inspired allocation mimics the collective intelligence of social insects to distribute tasks. These self-organizing methods, allow a swarm to handle sequential tasks without needing central control, global data, or even direct communication between the agents. Instead, agents adjust based on the time they spend waiting for the previous step to be finished. This allows the swarm to stay highly efficient and quickly adapt if a task's duration or requirements change (Brutschy et al., 2012).

Hybrid strategies are employed for complex missions involving time-dependent task sequences. These models are designed to minimize both energy use and completion time simultaneously, using algorithms that elect temporary leaders and use multi-round negotiations to ensure rapid, continuous decision-making under pressure (Liu et al., 2025)

3.4 Class-level vs Agent-level allocation

A critical design choice in heterogeneous swarm task allocation, is determining the appropriate level of detail for assignment, specifically whether to a class of agents or to individual agents. The fundamental difficulty in swarm programming arises in the system requirement which are formulated at the swarm level (globally), while control rules must be coded at the individual agent level (locally). Combining these global-to-local levels with mathematical modelling for system's behavior prediction, is generally considered as the grand challenge of swarm systems (Valentini et al., 2022).

Class-level allocation, distributes specific task categories to functional groups that are formed based on their common skills and features. This approach is computationally efficient and maintains the emergent properties of swarm behavior. A global-to-local methodology permits heterogeneous swarms to be composed from appropriate subsets of agents whose hard-coded behaviors have known global effects, enabling self-organized task allocation to be programmed directly at the swarm level (Valentini et al., 2022).

Agent-level allocation, by contrast, assigns tasks to specific individuals and is more commonly used when task requirements are highly differentiated. An adaptive consensus mechanism can trigger communication for the negotiation only in response to significant events, reducing non-essential interactions. This method allows a swarm to automatically adjust its coordination speed, based on environmental challenges, while using Behavior Trees to ensure each agent remains resilient (*Event-Triggered Adaptive Consensus for Multi-Robot Task Allocation*, 2026).

Choosing between these approaches comes down to a trade-off. Finding the optimal way to assign multiple agents to many tasks using brute-force methods, quickly becomes impractical, especially when task performance scales unevenly with the number of agents, harder tasks may need more agents to reach the same results as easier ones, and gains often level off in a nonlinear way as more agents are added. Class-level approaches trade off per-agent optimality in favor of better scalability and more robust collective behavior, whereas agent-level approaches allow more precise control but at a significantly higher computational cost (Bingöl et al., 2026).

3.5 Resource constraints (Energy & Charging)

Energy is the most fundamental resource constraint in swarm systems, restricting both mission endurance and the scope of operational tasks. Swarm systems are typically powered by batteries or fuel, so the optimization of the energy usage is crucial. This includes minimizing energy consumption for tasks, coordination, and offloading across the swarm. By efficiently allocating tasks and coordinating the workload, energy can be maintained, extending the operational time of the agents in the environment (Asrar Ahmed Baktayan et al., 2024).

Energy modeling should be integrated directly into the allocation process. Basic battery consumption models estimate energy use by assigning drain rates to the various tasks that agents carry out, along with a steady drain tied to their ongoing operation. More advanced, models go further by incorporating factors such as system dynamics and environmental conditions, and they can be adapted to meet the specific requirements of individual agents as well as the swarm as a whole (Phadke et al., 2024).

Charging station placement and scheduling add another level of optimization to the problem. In energy-constrained, multi-station task allocation framework, the system can balance task completion time, workload distribution among agents, and overall energy consumption, allowing agents to recharge dynamically at different locations while maintaining efficient task execution (Zhang et al., 2025). Intelligent replacement strategies are equally important to prevent coverage gaps. An effective replacement policy determines the optimal energy threshold at which each agent should return for recharging, with the goal of maximizing overall service availability and using resource efficiently. Without proper coordination, multiple agents depleting their energy simultaneously can increase the need for replacements and lead to temporary loss of coverage if reserves are exhausted concurrently (Sanchez-Aguero et al., 2020).

From a scheduling perspective, energy awareness must be a first-class constraint alongside task dependencies. Energy-aware task scheduling in swarm systems integrates energy consumption estimation, charging contingency strategies, and energy harvesting predictions to reduce the overall task completion time. These problems are usually

addressed using metaheuristic optimization techniques, such as Adaptive Particle Swarm Optimisation (Mokhtari et al., 2025). At the system level, effective resource management must explicitly account for the trade-off between processing delays and energy use. It should ensure that the total energy consumption across all agents remains within their battery limits, while also preventing positional overlap to avoid collisions, all handled within a unified optimization framework (Xu et al., 2025).

Chapter 4

Proposed optimization framework

4.1 Framework overview

The proposed framework addresses mission allocation in heterogeneous swarm systems by operating at the level of swarm classes rather than individual agents. Instead of evaluating every agent independently for each arriving mission, the framework groups agents into typed classes based on shared functional capabilities and selects the most suitable class for a given mission through a multi-criteria scoring function with dynamically adjusted weights. This design choice significantly reduces computational overhead while preserving the emergent, adaptive properties that make swarm systems effective in complex environments.

The system supports six executor classes, each class is characterized by a six-dimensional capability vector aligned with the corresponding mission domain. A separate scout class is dedicated to environmental monitoring and mission discovery. This role separation ensures that executor agents remain focused on operational tasks, while scouts continuously patrol the map and expose new missions as the simulation progresses. Scouts do not directly engage in task execution instead, their sensor detections trigger the assignment pipeline for executors.

The framework operates in a continuous simulation loop advancing in discrete time steps. At each step, scout agents patrol and detect undiscovered missions within their sensor range. Once a mission is discovered, the scoring function evaluates all available executor agents against it, and the best-matched agent or agents are assigned. Simultaneously, a charging station management subsystem monitors each agent's energy level in real time and governs access to recharging infrastructure using a priority-based queue discipline. The interplay between task allocation, energy management, and adaptive learning constitutes the core contribution of this framework.

A key architectural choice is the use of a shared random seed across all three compared queue disciplines (FIFO, LIFO, and OPT). This guarantees that all modes operate under identical initial conditions: the same agent positions, the same capability vectors, the same mission spawn schedule, and the same charging station placement. Any differences in performance outcomes can therefore be attributed solely to the queue discipline, making the comparison statistically fair and reproducible.

4.2 Swarm class representation

Each executor agent is instantiated from one of six type classes. In base capability vector is defined by the class archetype and perturbed with uniform noise in the range [-0.1, 0.1] per dimension, then L2-normalized, producing an agent whose skills are specialized but not perfectly uniformed. Agents also receive two performance variants, a "Speedy" (speed*1.3, endurance*0.7) and a "Tank" variant (speed*0.7, endurance*1.5), , which are applied alternately within each class, introducing intra-class diversity. When creating agents, these versions act like fixed rules that shape their traits on the initialization setup, mirroring actual hardware limits engineers face. For the scope of this Thesis, these specific numerical modifiers were custom-defined and heuristically selected to accurately simulate different performance boundaries. Robotic agents that move faster usually carry less power because frame strength and battery needs pull in opposite directions. Instead of making every unit identical, small built-in differences create

distinct working styles among peers doing similar jobs. The point of such variation, is to test how well agent decisions adapt when assigning tasks down to tiny details. It pushes the system to check not only if the swarm is capable for the mission, but also whether its physical profile aligns with the operational urgency or workload demands of the crisis.

Formally, for executor agent **i** of class **c**, the capability vector is defined as follows:

$$cap_i = \frac{cap_c + \epsilon}{\|cap_c + \epsilon\|'}, \epsilon \sim U(-0.1, 0.1)^6 \qquad (4.1)$$

where $\boldsymbol{cap_i}$ represents the final normalized capability vector of the agent, and $\boldsymbol{cap_c}$ denotes the baseline template vector corresponding to its assigned swarm class. The term $\boldsymbol{\epsilon}$ introduces a six dimensional random noise vector drawn from a uniform distribution, which injects minor operational variations among agents of the same class to model realistic intra-class diversity. Crucially, the mathematical expression applies the Euclidean norm ($\|\cdot\|$) strictly to the denominator. This vector normalization step ensures that while individual skill traits are slightly varied by the noise, the overall magnitude of the final capability vector remains scaled exactly to a unit length of $\mathbf{1}(\|\boldsymbol{cap_i}\| = \mathbf{1})$, thereby preserving mathematical fairness during the subsequent task allocation matching process.

The energy maximum is computed as:

$$e_{maxi} = endurance_i * \mu_{end} \qquad (4.2)$$

where $\boldsymbol{endurance_i}$ represents the baseline maximum operational capacity of individual agents, which is sampled uniformly from a continuous domain specific interval,

$$endurance_i \sim U(endurance_{min}, endurance_{max}) \qquad (4.3)$$

defined uniquely for each function class in the configuration parameters. The term $\boldsymbol{\mu_{end}}$ represents a compact, custom defined scaling multiplier used to simulate physical hardware trade offs. This modifier follows an alternating configuration structure, mapping to a value of 0.7 for "Speedy" variant and 1.5 for "Tank" variants. Ultimately the result of these variables determines the final maximum energy threshold $\boldsymbol{e_{maxi}}$, which dictates the absolut upper energy capacity of the agent and scales its dynamic safety boundary

$$\left(low_{energy_{threshold}} = 0.2 * e_{maxi}\right) \qquad (4.4)$$

This micro heterogeneity directly influences the swarm logic, forcing the task allocation algorithm to evaluate not just the skill alignment of a unit, but how its physical energy depth matches the spatial distance and work intensity demands of the active crisis.

Scout agents follow a separate configuration. Each scout's capability vector is randomly generated and normalized, giving it a generalist profile rather than a task-specific one. Scout variants adjust speed and sensor range rather than endurance, cycling through four distinct variants that produce scouts with different patrol styles: wide-range slow scouts, narrow-range fast scouts, and intermediate configurations. This ensures that the scout tier provides robust coverage of the map under a variety of patrol behaviours, reducing the risk of systematic blind spots in mission discovery.

4.3 Multi-criteria scoring function

Mission-to-agent matching is performed through a multi-criteria suitability function that simultaneously considers four factors: capability match, spatial proximity, endurance ratio, and adaptation potential. Rather than relying on a single criterion such as capability alone, this composite approach captures the multiple dimensions that make an agent genuinely suitable for a mission in the context of the overall swarm state. The function is evaluated for every available executor agent against every open, discovered, and unassigned mission at each time step. Formally, the consolidated suitability score $\boldsymbol{S_{ij}}$ for an executor agent **i** evaluated against an emergent mission **j** is defined as:

$$S_{ij} = \left(w_{cap} * \cos\_sim(C_i, M_j)\right) - \left(w_{dist} * \left(\frac{dist(i,j)}{M}\right)\right) - \left(w_{end} * \left(\frac{d_j}{e_i}\right)\right) + [a_i * (1 - \cos\_sim(C_i, M_j))] \quad (4.5)$$

Where the dynamically scaled weights are subject to the convexity constraint:

$$\sum w = w_{cap} + w_{dist} + w_{end} + w_{adapt} = 1.0 \quad (4.6)$$

In this multicriteria formulation (4.5), the variables are strictly separated into positive utility rewards and negative operational costs to govern agent behaviour under survival constraints. The reward terms include $\boldsymbol{cap_{match_{ij}}}$, which quantifies the directional alignment of skills via cosine similarity ($\boldsymbol{\cos\_sim}$) between the agent's capability vector ($\boldsymbol{C_i}$) and the mission's requirement vector $\left(\boldsymbol{M_j}\right)$, alongside an adaptation potential component. This adaptation potential measures the learning headroom of an agent, mathematically defined as its internal adaptation factor ($\boldsymbol{a_i}$) multiplied by the capability deficit ($\boldsymbol{1 - \cos\_sim(C_i, M_j)}$). Conversely, the cost terms subtracted from the score are the spatial proximity component, representing the spatial Euclidean distance ($\boldsymbol{dist(i,j)}$) scaled to the interval **[0,1]** by the total map size **(M)**, and the endurance ratio, defined as the ratio of the mission's difficulty $\left(\boldsymbol{d_j}\right)$ to the agent's absolute physical endurance parameter ($\boldsymbol{e_i}$) . The system is structured this way to enforce a strict optimization trade-off, an agent is heavily penalized (driving $\boldsymbol{S_{ij}}$ downward) if it is positioned far from the emergency or lacks the physical stamina to complete it before energy exhaustion.

Crucially, the scaling weights ($\boldsymbol{w_{cap}}$, $\boldsymbol{w_{dist}}$, $\boldsymbol{w_{end}}$, $\boldsymbol{w_{adapt}}$) of the (4.6) are not static constraints, instead they are computed dynamically at each time step. The capability weight scales proportionally with the mission's normalized difficulty, the distance weight shifts based on spatial separation, and the endurance penalty scales aggressively with the agent's real-time energy depletion $\left(\boldsymbol{1 - \frac{energy}{energy\_max}}\right)$. By dividing each raw weight by their total sum $\left(\boldsymbol{w_{sumkeys}}\right)$, the framework normalizes the coefficients to sum perfectly to **1.0**. This dynamic convex combination prevents any singly metric from permanently dominating the decision space, ensuring that as a crisis escalates or an agent drains its reserves, the allocation framework fluidly shifts its priorities to optimize overall swarm utility and mission success.

To fully understand the behavioural implications of this optimization framework, it is necessary to isolate the independent operational logic of each specific core component.

- The **capability match** component measures how well an agent's learned capability vector aligns with the mission's requirement profile, computed as the cosine similarity between the two vectors. Cosine similarity is preferred over Euclidean distance because it captures directional

alignment in the capability space regardless of the vectors' magnitudes, making it robust to normalization differences between agents.

- The **proximity component** quantifies the travel cost for the agent to reach the mission. The Euclidean distance between the agent and the mission is computed and normalized by the map size, producing a value in [0, 1]. This term penalizes assignments that would require an agent to traverse large portions of the map, reducing wasted travel time and energy expenditure on movement.
- The **endurance ratio** reflects whether the agent has the operational stamina to handle the mission's difficulty. It is defined as the ratio of the mission's difficulty score to the agent's endurance parameter. An agent with low endurance assigned to a high-difficulty mission is penalized, as it is likely to exhaust its energy reserves before completing the task. This term thus introduces an implicit energy awareness into the allocation decision.
- The **adaptation potential** rewards agents that, while not currently specialized for the mission, have a high adaptation factor and therefore can learn toward the mission type during execution. This forward-looking component ensures that the framework does not always assign the most specialized agent when a flexible learner could grow into the role and benefit from the experience. Agents with high adaptation factors are naturally more general-purpose and serve as a bridge between classes in mixed-type scenarios.

4.3.1 Mathematical formulation

To formally evaluate the suitability score $\boldsymbol{S_{ij}}$, presented under macro-level composition of equation (4.5), each constitutive criterion is explicitly derived from the real-time kinematic, energetic and capability boundaries of the simulation space through that deterministic formulation.

More analytically each independent component is calculated using the physical dimensions of the fleet and environment in the fields of this Thesis as follows:

Capability Match ($\cos\_sim(C_i, M_j)$): Evaluated via a directional vector cosine similarity metric, integrating an infinitesimal floating-point regulizer ($\xi = 10^{-6}$) within the denominator:

$$\cos\_sim(C_i, M_j) = \frac{C_i * M_j}{\|C_i\|\|M_j\| + \xi} \qquad (4.7)$$

Where $C_i \in R^6$ represents the normalized operational capability vector of executor agent **i**, and $M_j \in R^6$ represents the requirement profile vector of the active mission **j**. The regularization constant **ξ** is explicitly integrated to maintain numerical stability during the initialization setup or whenever unitialized, zero-magnitude vectors are evaluated, effectively preventing catastrophic zero-division runtime errors. Because it isolates the angular separation between the two multi-dimensional vectors, the resulting scalar yields a pure match bounded tightly between **[0.0, 0.1]**, ignoring raw vector magnitudes and remaining completely robust to scale variances among highly heterogeneous swarm classes.

Spatial Proximity Component ($norm\_dist_{ij}$): Derived by computing the spatial Euclidean distance ($dist(i.j)$) between the coordinates of agent **i** and mission **j**,

normalized explicitly against the maximum map size boundary **(M = 50)** and bounded via a clipping function:

$$norm_{dist_{ij}} = min(1.0, \frac{dist(i,j)}{M}) \quad (4.8)$$

The inclusion of the map side length constant **(M = 50)** scales the raw pixel distance into a proportional ratio. While the absolute geometric diagonal of a **50 * 50** bounding grid is approximately 70.7, dividing **M** pushes typical operational distances near the standard unit interval **[0.0, 0.1]**. The exterior min(1.0,..) bounding function acts as a rigid saturation gate, which ensures that even if extreme multi-agent routing configurations push an asset into a n opposing extreme corner, that travel cost metric saturates flatly at **1.0**, preventing severe spatial penalties from completely over-riding and destroying favorable capability matches.

Endurance Ratio Component ($\boldsymbol{endurance\_ratio_{ij}}$)**:** Quantifies the task-induced structural workload strain by capturing the direct ratio between the scalar difficulty metric of the mission ($\boldsymbol{d_j}$) and the agent's maximum continuous physical endurance capacity ($\boldsymbol{e_i}$):

$$endurance_{ratio_{ij}} = \frac{d_j}{e_i} \quad (4.9)$$

Within the simulation environment, mission difficulty ($\boldsymbol{d_j}$) scales from a baseline of **2.0** up to a severe crisis peak of **9.0**, whereas individual agent endurance traits ($\boldsymbol{e_i}$) range between **15.0** and **45.0** depending on their assigned professional class archetype and specialized physical variant modifiers **(0.7 * for "Speedy" and 0.15 * for "Tank")**. Setting up this metric as an un-capped ratio means that if a low-stamina unit **(15*0.7=10.5)** is evaluated against a severe structural collapse or a hard emergency ($\boldsymbol{d_j}$ = **9**), the ratio spikes aggressively to $\frac{\mathbf{9}}{\mathbf{10.5}} = \mathbf{0.857}$. Because this term is subtracted from the master allocation score, it inflicts a massive suitability penalty, ensuring that delicate or light-duty units are naturally filtered out from energy-intensive operations that would cause mid-mission energy blackouts.

Adaptation Potential Component ($\boldsymbol{adapt\_potential_{ij}}$)**:** Measures the forward-looking learning capacity and skill elasticity of a candidate asset relative to its current operational specialization gap:

$$adapt_{potential_{ij}} = a_i * \left(1.0 - \cos\ \_sim(C_i, M_j)\right) \quad (4.10)$$

Where $\boldsymbol{a_i}$ represents the internal, class-specific uniform adaptation factor of the unit. This term is structured to calculate a proportional reward, if an agent is already perfectly specialized for a crisis, the skill gap becomes zero, causing the adaptation reward to drop flatly to **0.0**, as no further skill growth can be extracted from the deployment. Conversely, if a major capability deficit exists, the term scales upward to its theoretical maximum of $\boldsymbol{a_i}$. This design choice deliberately incentivizes the assignment of highly flexible, general-purpose learners to minor or mid-tier challenges, allowing them to accumulate online training experience and serve as an organic cross-training bridge between rigid swarm classes.

4.3.2 Dynamic weighting mechanism

Rather than relying on static, hardcoded priority rules that fail under unpredictable environmental shifts, the optimization framework dynamically recalibrates its criteria coefficients at every discrete time step. Let $\boldsymbol{d_m^{norm} = \frac{d_j}{\max\_intensity}}$ represents the normalized mission difficulty scaled cleanly within **[0.2, 0.9]** reflect the exact real-time battery depletion ratio of the evaluated executor unit, moving from **0.0** at full charge to **1.0** at absolute exhaustion. The optimization layer first computes four unscaled baseline multipliers ($\boldsymbol{w'}$) through separate linear scaling profiles that mirror distinct operational trade-offs:

$$w'_{cap} = 0.35 + 0.25 * d_m^{norm} \tag{4.11}$$

$$w'_{dist} = 0.20 + 0.15 * norm_{dist_{ij}} \tag{4.12}$$

$$w'_{end} = 0.10 + 0.20 * E_i^{dep} \tag{4.13}$$

$$w'_{adapt} = 0.10 \tag{4.14}$$

The constant coefficients within these unscaled equations have been carefully selected to enforce specific priority boundaries at runtime:

- **Capability Baseline (0.35 to 0.60)**: Skill alignment is prioritized as the most critical core factor, holding a high baseline of **0.35** even when task urgency is low. When a severe, high-intensity crisis breaks out ($\boldsymbol{d_m^{norm} \to 1.0}$), this weight scales up linearly to **0.60**, preventing mismatched, un-specialized units from getting assigned to highly volatile scenarios.
- **Proximity Baseline (0.20 to 0.35):** If an emergency occurs immediately next to an agent's coordinates, the distance weight remains at a baseline minimum of **0.20**. However, if the task spawns at the far periphery of the map ($\boldsymbol{norm_{dist} \to 1.0}$), the weight automatically climbs to **0.35**, penalizing long-distance travel to protect the fleet from wasting immense battery capacity on simple transit.
- **Endurance Baseline (0.10 to 0.30):** When an agent is fully charged ($\boldsymbol{E_i^{dep} = 0}$), the framework minimizes the endurance penalty weight to a negligible **0.10**. As the agent works and its battery drains heavily ($\boldsymbol{E_i^{dep} \to 1.0}$), this weight automatically triples to **0.30**, amplifying the endurance mismatch cost and preventing low-energy assets from locking onto heavy tasks.
- **Adaptation Baseline (0.10):** Held as a rigid, static coefficient of **0.10** to act as a constant baseline driver for long-term fleet cross-training without interfering with short-term survival metrics.

To satisfy the convex model constraint outlined in expression (4.6), the system dynamically sums these changing raw components ($w_{sum} = w'_{cap} + w'_{dist} + w'_{end} + w'_{adapt}$) and applies a normalization scaling to generate the finalized operational weights:

$$w_{cap} = \frac{w'_{cap}}{w_{sum}} \tag{4.15}$$

$$w_{dist} = \frac{w'_{dist}}{w_{sum}} \tag{4.16}$$

$$w_{end} = \frac{w'_{end}}{w_{sum}} \tag{4.17}$$

$$w_{adapt} = \frac{w'_{adapt}}{w_{sum}} \tag{4.18}$$

This dynamic convex division ensures that the coefficients continually balance and counter-scale against each other, completely preventing any single operational metric from permanently blinding or dominating the task allocation decision space.

4.3.3 Decision-making process

The centralized allocation loop processes incoming data fields sequentially at every discrete simulation step, ensuring conflict-free resource locking according to a four-phase chronological pipeline:

- **Kinematic Fleet Audit:** The entire fleet array is parsed to isolate valid executors. An individual asset enters the eligible allocation pool if and only if its state attributes satisfy three strict concurrent constraints: it must be non-busy, completely disengaged from any physical charging station infrastructure, and maintain an instantaneous energy volume equal to or exceeding a protective safety margin of **40%** of its maximum capacity ($\boldsymbol{energy \geq 0.4 * energy\_max}$). The **40%** initial energy gate is chosen as a rigid defensive buffer, guaranteeing that any unit exiting the routing phase holds an ample power reserve to transit across long grid distances and execute heavy labor without instantly triggering a mid-transit blackout or a premature emergency fallback loop.
- **Quality Gating:** The allocation layer calculates the dynamic suitability score $\boldsymbol{S_{ij}}$ for every candidate pairing between the audited available pool and all active, scout-exposed missions. To enforce strict system-wide quality control, any asset whose composite matching score fails to clear a minimum quality boundary threshold ($\boldsymbol{\tau = 0.15}$) is instantly pruned from consideration. This threshold prevents severely mismatched assignments, such as sending a low-stamina surveillance asset with no firefighting capabilities to a major fire emergency, where the unit would merely waste valuable battery capacity without contributing meaningful work progress.
- **Team Assembly Scaling:** The filtered, valid candidate array is ordered via a descending sort based on their finalized suitability metrics $\boldsymbol{S_{ij}}$. To handle scaling workloads and distribute intense physical strain, the architecture evaluates the mission's underlying difficulty parameter ($\boldsymbol{d_j}$). If a mission features a high difficulty ranking ($\boldsymbol{d_j > 6}$), the system automatically triggers a collaborative team assembly mode, pulling the top two highest-scoring units from the queue concurrently to execute joint task resolution. For simpler, mid-to-low-tier operations ($\boldsymbol{d_j \leq 6}$), the allocation pipeline routes exactly one unit.
- **Concurrent Resource Locking:** Once the optimal asset pairings are determined, the chosen units are instantly updated to an operational state of busy, linked directly to the mission object, and completely removed from the active

allocation matrix for the remainder of that time step. This immediate resource lockout guarantees complete mutual exclusivity, mathematically preventing double-assignment anomalies where a single asset might be erroneously allocated to multiple independent crises within the same temporal step.

Chapter 5

Energy management and charging strategy

5.1 Energy constraints in swarm systems

Energy is the primary operational constraint governing swarm behavior throughout the simulation. Each agent has energy levels, whose maximum capacity is determined at initialization from the class-specific endurance range and then scaled by the variant modifier. The "Speedy" variant reduces energy capacity to 70% of the base ($\boldsymbol{e_i * 0.7}$), while the "Tank" variant extends it to 150% ($\boldsymbol{e_i * 1.5}$), creating a population with meaningful spread of operational endurance. This heterogeneity in energy levels is a deliberate design feature, as it produces a more realistic scenario where not all swarm types can sustain the same duration of activity.

Scout agents deplete their energy continuously during patrol, regardless of whether they are actively detecting missions. The depletion rate at each step is calculated as:

$$move_{cost} = 0.015 * speed_i * uniform\,(0.8, 1.3) \tag{5.1}$$

Where the constant **0.015** serves as the baseline consumption factor for scouter units, $\boldsymbol{speed_i}$ represents the agent's randomized kinematically-scaled speed, and the stochastic multiplier (**0.8, 1.3)** introduces variability in energy consumption due to environmental factors like atmospheric resistance or friction. Because scouts are always moving, they reach their low-energy thresholds more frequently than executors, making their access to charging stations a critical factor in maintaining continuous map coverage.

Executor swarms consume energy only when they are actively pursuing or working on a mission. Idle executors stay at their starting position without any energy expenditure, which models the assumption that stationary swarms in a standby state consume negligible energy levels. When engaged on a mission, the movement energy cost scales with the swarm's speed and a stochastic factor, where the base rate differs from class to class, reflecting the heavier but more power-efficient hardware typical to those classes. An additional work-specific energy cost is incurred while performing active task operations within the mission radius, scaled by mission intensity and difficulty.

To formalize this behavioral logic, the cumulative energy consumption rate $(\boldsymbol{E_{deplet}})$ for an active executor agent **i** during a single simulation step is dynamically governed by two operational states:

$$E_{deplete} = \begin{cases} base * speed_j * uniform(0.9, 1.2) \\ base_{work} * \left(1.0 + \dfrac{intensity_j}{MAX_{INTENSITY}} + \dfrac{d_j}{10}\right) * uniform(0.7, 1.4) \end{cases} \tag{5.2}$$

During transit mode, the agent expends energy traveling to task coordinates, where the **base** constant handles hardware scale variances (set to **0.02** for heavy units, and **0.03** for agile units). This base rate scales directly against the agent's current kinematic $\boldsymbol{speed_i}$ and is modulated by a stochastic noise multiplier, $\boldsymbol{uniform(0.9, 1.2)}$, which introduces

unforeseen environmental friction factors. Once positioned inside the task radius, the agent switches to execution mode, swapping transit costs for a specialized workload penalty. Here, the static multiplier $\boldsymbol{base\_work}$ **0.10** for heavy classes and **0.12** for light classes, represents the baseline mechanical drain of operating active tools. To simulate scaling workload resistance, this baseline is amplified by a compound internal stress factor: the mission's localized $\boldsymbol{intensity_j}$ normalized against the maximum environmental limit ($\boldsymbol{MAX_{INTENSITY} = 10}$) ,and the mission's categorical difficulty ($\boldsymbol{d_j}$) scaled over a standard base of **10**. Finally, an extended stochastic range, $\boldsymbol{uniform(0.7, 1.4)}$,is applied to model unpredictable workplace volatility (such as shifting fire boundaries).

To prevent swarms from running out of energy mid-mission, the system employes a dynamic low-energy threshold that adapts to each swarm's distance from the nearest charging station. The threshold is computed as:

$$\theta_i = \left(0.15 + 0.25 * \left(\frac{dist(i, nearest_{station})}{L}\right)\right) * e_{\max i} \tag{5.3}$$

This distance-aware threshold formulation establishes a fluid, spatially adaptive safety boundary layer rather than relying on a rigid, blanket "low-battery" trigger. Within the core linear scaling function, the variable $\boldsymbol{dist(i, nearest_{station})}$ tracks the continuous physical Euclidean distance from the real-time coordinates of agent **i** to the closest charging node, which is subsequently normalized by the global map length constant ($\boldsymbol{L = 50}$). Multiplying this spatial risk profile by the agent's individual maximum capacity parameter ($\boldsymbol{e_{\max i}}$) ensures that the safety trigger seamlessly accommodates internal hardware variations, such as lightweight "Speedy" frames **(0.7 * endurance)** or structural "Tank" frames **(1.5 * endurance)**. The interaction between the baseline offset constant **0.15** and the scaling multiplier **0.25** enforces strict, predictable behavioral bounds at the extremes of the field, if an asset is working immediately adjacent to a charging hub ($\boldsymbol{dist \to 0}$) ,the ratio collapses and pins the threshold to its baseline minimum of **15%** ($\boldsymbol{\theta_i = 0.15 * e_{maxi}}$), allowing the unit to safely exhaust up to **85%** of its energy on task labor. Conversely, when deploying to volatile tasks at the far periphery of the grid ($\boldsymbol{dist \to L}$), the equation forces the terms to sum directly to a defensive maximum of **40%** ($\boldsymbol{\theta_i = 0.40 * e_{maxi}}$), proactively safeguarding a substantial power reserve to cover the high kinetic transit overhead of the long return journey.

Swarms far form charging stations trigger the recharging response earlier, while swarms near to charging stations can afford to remain on task longer. This spatial in the energy management policy reduces the incidence od emergency energy depletion and improves overall mission throughput.

5.2 Charging stations and queue management

Charging stations are placed in random positions on the map in each experiment but stay fixed during execution. In this work, each station is initialized with a random capacity of **1** to **3** simultaneous charging slots and maintains two data structures: a list of swarms currently being charged, and a waiting queue of swarms that have arrived but cannot yet be served. For FIFO and LIFO modes, the charging rate is fixed at **2.5** energy units per step for all agents, regardless of swarm type or energy capacity. In OPT mode, a Dynamic Power Allocation mechanism is applied; OPT charges agents at an increased rate of **3.0** energy units per step, simulating smart-grid scheduling that prioritizes critical

units for faster recovery. An agent is released from the station when its energy reaches its individual maximum ($\boldsymbol{E_i \geq E_{max\,i}}$).
Three distinct operational behaviors govern how swarms interact with charging infrastructure:

- **Urgent Charging:** When a swarm's energy falls below its dynamic low-energy threshold ($\boldsymbol{\theta_i}$), it immediately interrupts its current activity, abandons any in-progress mission assignment, and navigates directly to the nearest station at its maximum movement speed. This preemptive behavior ensures that swarms do not reach zero energy in the field.
- **Opportunistic Charging:** An idle executor with energy below **80%** of its maximum capacity ($\boldsymbol{E_i < 0.8 * E_{max\,i}}$) will proactively seek a station with available slots and no waiting queue. This behavior makes productive use of idle time between missions, topping up energy levels without waiting for a critical threshold event.
- **Emergency Recovery:** Applies when an agent fails to manage its reserves and reaches exactly zero energy ($\boldsymbol{E_i \leq 0}$). In this case, the swarm crawls toward the nearest station at a severely reduced speed constant of **0.5** units per step. This imposes a massive opportunity cost, as the unit is locked out of service during the entire crawl phase.

5.3 Proposed charging priority algorithm

The proposed Optimal (OPT) scheduling discipline selects the next queued asset to advance into an open charging slot by evaluating a multi-dimensional operational urgency score, $\boldsymbol{C(i)}$, executed continuously at each discrete station step. Unlike static FIFO or LIFO baselines, which remain completely agnostic to the real-time physical states of the fleet, the OPT architecture continuously re-ranks all queued assets based on current environmental and internal pressures. Formally, the composite charging cost score for a queued agent **i** is defined as:

$$C(i) = (\alpha * R_i) + (\beta * V_i) + (\gamma * U) + \psi_i \tag{5.4}$$

where $\boldsymbol{\alpha, \beta}$ and $\boldsymbol{\gamma}$ represent the dynamically modulated normalization coefficients derived from macro-system indicators. The term ($\boldsymbol{\psi_i}$) functions as a linear starvation prevention modifier, calculated at each time step as:

$$\psi_i = \left(t_{currernt} - t_{entry_i}\right) * 0.05 \tag{5.5}$$

This modifier utilizes the agent's current waiting duration to incrementally boost its priority score by **0.05** units per step, mathematically preventing low-priority assets from being trapped indefinitely in the queue by a continuous influx of high-priority units.

- **The first** constitutive utility variable, **Individual Energy Risk** ($\boldsymbol{R_i}$), quantifies how dangerously close an agent is to absolute power exhaustion relative to its dynamic geographical safety boundary ($\boldsymbol{\theta_i}$). Let $\boldsymbol{slack_i = E_i - \theta_i}$ represent the exact energy margin an agent possesses above its critical threshold. The risk factor is modeled as a highly non-linear equation governed by a rigid bounding function:

$$R_i = clip\left(\max\left(0.0, \quad \frac{-slack_i}{\theta_i + 10^{-6}}\right)^{1.5}, 0.0, 2.5\right) \tag{5.6}$$

Within this function, $\boldsymbol{E_i}$ represents the agent's real-time remaining energy volume, while $\boldsymbol{\theta_i}$ represents its dynamic, distance-aware safe low-energy threshold. The term 10^{-6} acts as an infinitesimal floating-point stability regularizer to prevent division-by-zero crashes, if $\boldsymbol{\theta_i}$ approaches zero. From a perspective of numerical intuition and operational boundaries, when a unit arrives at the queue safely above its threshold ($\boldsymbol{slack_i \geq 0}$), the numerator forces the entire term to **0.0**, indicating zero immediate energetic risk. However, if an asset drops below its safety line into negative slack, the negative sign flips the ratio into a positive value, and the **1.5** exponent generates a strongly non-linear urgency signal. This exponential acceleration ensures that as an agent moves closer to absolute exhaustion, its priority spikes aggressively. The upper clip ceiling of **2.5** represents a strict engineering boundary constraint, it prevents a completely dead agent from generating an astronomical score that would permanently blind the station scheduler, allowing units that require minimal charging time to be quickly serviced and routed back into active operations.

- **The second** utility variable, **Capability Value** ($\boldsymbol{V_i}$)**,** estimates the expected operational contribution of the queued unit to the pending mission backlog based on its specialized skill match. Let $\boldsymbol{M_{open}}$ represent the set of discovered, active, and unassigned missions across the map. The parameter dynamically branches based on the agent's core archetype such that:

$$V_i = \begin{cases} \max m \in M_{open}(S_{im}), & if\ agent_{type} = 'executor' \\ 0.8, & if\ agent_{type} = 'scout' \end{cases} \tag{5.7}$$

For executor units, the algorithm scans the entire unassigned task pool and maps $\boldsymbol{V_i}$ to the highest suitability score the unit can currently achieve ($\boldsymbol{S_{im}} \in [\mathbf{0.15}, \mathbf{1.0}]$). This ensures that units exceptionally qualified to resolve a pressing, high-priority emergency are pushed to the front of the queue. Conversely, because scout units do not execute missions directly, they lack a standard suitability score ($\boldsymbol{S_{im} = 0}$). The architecture explicitly assigns scouts a high, fixed static constant of $\boldsymbol{V_i = 0.8}$, which serves a vital structural role by recognizing that scouts represent the primary operational bottleneck for environmental monitoring and anomaly detection, their prolonged absence from active patrol paralyzes mission discovery, thus justifying a permanently high retrieval priority in the scheduling queue.

- **The final** utility component, **Global Urgency** ($\boldsymbol{U}$), captures the macro-level environmental pressure exerted on the entire swarm by unmet system demands, functioning as a uniform amplification factor. It is formulated as:

$$U = \min(1.0, \frac{unassigned\_count}{2.0}) \tag{5.8}$$

where $\boldsymbol{unassigned\_count}$ tracks the absolute number of discovered, active crises that currently have no agents allocated to them. The constant denominator of **2.0** acts as an urgency scaling driver. If the environment contains only **1** unassigned mission, $\boldsymbol{U}$ yields **0.5**. However, the moment **2** or more active emergencies sit abandoned in the field, the ratio reaches or exceeds **1.0**, causing the exterior $\mathbf{min}(\mathbf{1.0}, \ldots)$ function to saturate flatly at a ceiling of **1.0**. This structural saturation shifts the scheduler's behavior across the entire queue, it uniformly inflates the priority scores of all waiting units, forcing the charging slots to cycle assets rapidly to reinforce a fleet under heavy operational stress.

5.4 Priority metrics and scheduling

The priority coefficients ($\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}$) are dynamically re-derived at every step to match system-level stress factors rather than relying on static scheduling logic. Let $\boldsymbol{A}$ represent the total agent fleet populated within the environment. The framework measures three distinct real-time system state variables:

$$Energy\ Stress\ (\sigma_e) = \frac{below\_threshold\_count}{\|A\|} \tag{5.9}$$

$$Mission\ Pressure\ \ (\sigma_m) = \min\left(1.0,\ \ \frac{unassigned\_count}{\|A\|}\right) \tag{5.10}$$

$$Station\ Congestion\ (\kappa) = \min\left(1.0,\ \ \frac{queue\_length}{capacity}\right) \tag{5.11}$$

Using these state ratios, the optimization layer constructs three independent linear equations to calculate the raw, unscaled baseline weights:

$$\alpha_{raw} = 0.10 + 0.20 * \sigma_e \tag{5.12}$$

$$\beta_{raw} = 0.50 + 0.40 * \sigma_m \tag{5.13}$$

$$\gamma_{raw} = 0.15 + 0.20 * \kappa \tag{5.14}$$

The constant parameters and baseline limits within these raw equations govern the structural behavior of the scheduler:

- Energy Urgency Bounds ($\boldsymbol{\alpha_{raw}} \in [\mathbf{0.10}, \mathbf{0.30}]$): Holds a minimal baseline of **0.10** when the fleet is energetically healthy. If a mass energy depletion cascade occurs ($\boldsymbol{\sigma_e \to 1.0}$), the weight scales up linearly to **0.30**, forcing the station to bypass long-term task qualifications and prioritize pure unit survival.
- Mission Workload Bounds ($\boldsymbol{\beta_{raw}} \in [\mathbf{0.50}, \mathbf{0.90}]$): Retains an exceptionally high baseline constant of **0.50**. This ensures that the scheduler is inherently task-centric, scaling up to a dominant **0.90** under dense crisis accumulation ($\boldsymbol{\sigma_m \to 1.0}$) to rapidly inject highly-capable assets back into the field.
- Infrastructure Congestion Bounds ($\boldsymbol{\gamma_{raw}} \in [\mathbf{0.15}, \mathbf{0.35}]$): Anchored at a baseline of **0.15**. If the station queue grows highly congested relative to its maximum capacity ($\boldsymbol{\kappa \to 1.0}$), the weight expands to **0.35**. This forces the algorithm to focus on clearing the structural traffic jam by promoting units that require minimal charging times.

To satisfy the convex model constraint and preserve the interpretability of the charging cost as a clean weighted average, these values are divided by their composite sum ($\boldsymbol{Z = \alpha_{raw} + \beta_{raw} + \gamma_{raw}}$)yielding the final normalized weights used to process the queue:

$$\alpha = \frac{\alpha_{raw}}{Z} \tag{5.15}$$

$$\beta = \frac{\beta_{raw}}{Z} \tag{5.16}$$

$$\gamma = \frac{\gamma_{raw}}{Z} \tag{5.17}$$

This dynamic convex division ensures that the final weights always sum to exactly **1.0** regardless of the system state ($\sum \boldsymbol{w} = \mathbf{1.0}$). Consequently, the charging scheduler shifts its emphasis in response to changing operational conditions. At each station step, the OPT procedure first computes the current system state, derives the dynamic weights, and evaluates $\boldsymbol{C}(\boldsymbol{i})$ for every agent in the queue. The agent with the highest charging cost is promoted to the charger. Because this evaluation is repeated each step, an agent's priority can change while it is waiting in the queue as missions are completed, new missions are discovered, and other agents' energy levels shift.

The queue wait time for each agent is recorded from the moment it joins the queue to the moment it begins charging. These wait times are aggregated across all agents and stations for each simulation run and reported as the average queue wait metric. This metric captures not only the efficiency of the queue discipline but also the broader interaction between energy management, mission load, and station capacity, making it one of the primary indicators in the experimental comparison.

Chapter 6

Experimental setup, simulation & results

6.1 Simulation environment and swarm configuration

All experiments were conducted in a discrete two-dimensional grid environment of size 50 x 50 units. Agents move continuously within this bounded space, with positions represented as floating-point coordinates. The environment contains no terrain or obstacle features; it is an open field in which all positions are equally accessible. This deliberate simplicity isolates the effects of the algorithmic variables under study from confounding factors such as path planning complexity or terrain-dependent energy costs.

The simulation is implemented entirely in Python, using NumPy for vectorized numerical operations and the tqdm library for progress tracking during batch experiments. All simulation logic runs in a headless, visualization-free mode during the grid search to maximize throughput. A single experiment run proceeds as a discrete-time loop that advances one step at a time until the mission completion criterion is met. Each step executes agent movement, mission evolution, charging station updates, work processing, mission spawning, scout detection, and assignment in a fixed order.

The swarm consists of two functional tiers. Scout agents patrol the map autonomously using a random-walk strategy with periodic direction changes occurring every 10 to 30 steps. When a direction change is triggered, a new heading angle is sampled uniformly from $[0,\ 2\pi]$. Scouts detect missions within a circular sensor radius and add them to the discovered set with a detection probability of 80% per overlapping scout per step. This stochastic detection model reflects the imperfect sensing typical of real autonomous agents.

Executor swarms are organized in six typed classes, two executors population sizes were tested, a balanced number of two swarms of each (12 executors total) and another balanced number of three per swarm types (18 executors total). Within each class, agents alternate between the Speedy and Tank performance variants, ensuring equal representation of both profiles in every class. Scout populations were set to 2 or 4 per run.

Charging infrastructure consists of 1 or 2 stations placed at uniformly random positions on the map. Each station's capacity is sampled from {1, 2, 3}. The station positions and capacities are determined by the shared random seed and are therefore identical across all three queue discipline modes in a given experiment run, ensuring a controlled comparison. All three queue disciplines use the same station objects with the same physical parameters, only the agent selection rule differs.

6.2 Simulation scenarios and evaluation metrics

Missions are introduced stochastically throughout the simulation using a Bernoulli spawn process. At each time step, a mission from the pre-generated blueprint list is spawned with probability **p = 0.2**, until the target number of missions for the run has been reached. The blueprint list is pre-generated from the shared seed before any mode-specific simulation begins, and the spawn schedule is computed once and shared across all three modes. This guarantees that all three queue disciplines face exactly the same sequence of mission arrivals, making the comparison methodologically sound.

Each mission is assigned a type drawn uniformly from the six mission types similar to the swarms types. The mission's type vector, difficulty, radius, and intensity are

sampled from type-specific ranges using a mission-local random seed derived from the blueprint, ensuring that mission properties are identical across all three modes. A randomly selected **20%** of missions have a spread mechanic: if they remain unassigned, their radius and intensity increase over time at a rate sampled from **[0.1, 0.3]**. The intensity escalation per step is amplified by a factor of 4.0 (i.e. intensity += spread_rate * 4.0), a significant increase from the previous multiplier of 1.0. This aggressive escalation factor simulates a critical tipping point in disaster progression, where a localized incident compounds into a catastrophic failure if left unattended. In the underlying simulation pipeline, this creates a rigorous tactical trade-off: while standard algorithms might focus exclusively on charging high-energy or high-suitability assets, the rapid decay of unassigned "spreading" missions severely punishes queue stagnation. By forcing neglected crises to expand their physical radius and intensity footprint at this accelerated rate, the environment introduces a strong penalty for delayed responses, effectively stress-testing the scheduling layer to reward proactive, workload-aware queue interventions that prioritize system-wide crisis containment over isolated local efficiencies. Creating this strong time pressure penalizes delayed queue responses and rewards proactive, missions-aware charging management. This amplification makes neglected missions deteriorate much faster, creating strong time pressure that penalises delayed queue responses and rewards proactive, missions-aware charging management.

The simulation terminates when all target missions have been successfully completed. Mission targets of **100, 200, 400, and 500** were tested, representing a range of operational scales from short tactical engagements to extended campaigns. Larger mission targets stress-test the charging infrastructure over longer durations and provide more reliable statistical estimates of the performance metrics by averaging over a greater number of events. A safety cap of **500,000** simulation steps is applied to prevent infinite loops in pathological cases.

The performance metrics recorded per run and per queue discipline are:

- **Total Missions Completed (Successes):** The absolute count of active crises successfully resolved by the fleet during the simulation run. A mission is logged as a success the exact moment its remaining intensity ($\boldsymbol{intensity_j}$) is reduced to zero or less by active agent labor.
- **Total Agent-Mission Assignment Events (Attempts):** The cumulative counter of all resource allocation actions. Every time an idle executor locks onto a discovered mission, this counter increments by one, completely independent of whether that assignment ultimately succeeds or ends in mid-mission failure.
- **Failed Attempts:** The absolute count of aborted assignments. This is triggered when a working agent suffers an unexpected tool breakdown or physical battery exhaustion, forcing the system to strip the asset from the task, return the mission to a **Pending** state, and inflict a structural intensity penalty.
- **Assignment Efficiency:** The statistical reliability ratio of allocation choices, derived by dividing total successful completions by the cumulative number of allocation attempts:

$$Assignment\ Efficiency = \frac{Successes}{Attempts} \tag{6.1}$$

  Higher ratios indicate that the allocation layer deploys assets precisely, minimizing wasted transits and redundant assignments.
- **Failure Rate:** The operational vulnerability index, computed as the ratio of aborted tasks relative to total assignment attempts:

$$Failure\ Rate = \frac{Failed\ Attempts}{Attempts} \tag{6.2}$$

This directly quantifies the fragility of a queue discipline's matching choices regarding agent energy depletion and crisis scaling.

- **Avg Suitability:** The mean matching alignment quality. This tracks the average multi-criteria suitability score (S_{ij}) achieved by all selected executors at the exact temporal step their assignment was locked in, indicating how well the fleet matches specific task demands.
- **Avg Queue Wait:** The structural infrastructure latency benchmark. It calculates the average number of discrete steps a unit spends idling inside a station's waiting line, counting from its initial queue entry step to the exact step it advances into a functional charging socket.
- **Avg Energy Level:** The global fleet vitality metric. Computed as the continuous mathematical mean of remaining battery volumes across all scouters and executors across every elapsed simulation step, serving as a primary indicator of systemic energy sustainability.
- **Productivity Ratio:** The swarm's active labor utilization index. It evaluates the proportion of time units spend doing active crisis resolution compared to their non-productive maintenance downtime:

$$Productivity\ Ratio = \frac{\sum time_{working}}{\sum time_{working} + \sum time_{charging}} \tag{6.3}$$

- **Finish Step:** The absolute operational duration. It logs the exact simulation step at which the final active mission required by the target blueprint is successfully cleared. This represents the ultimate speed and consolidation capacity of the underlying queue management strategy.

A weighted composite score is computed after each run to provide a single ranking of the three methods. The component weights are: Assignment Efficiency 30%, Productivity Ratio 20%, Average Energy Level 15%, Average Queue Wait 15% (inverted, lower is better), Failure Rate 10% (inverted), and Finish Step 10% (inverted). Each metric is min-max normalized across the three algorithms before weighting, so the composite score reflects relative performance rather than absolute values. This score is used in the plotting pipeline to identify the best-performing algorithm for each experiment configuration across the full grid of parameter combinations.

6.3 Baseline methods (FIFO/LIFO)

Two classical queue management strategies serve as baselines against which the proposed OPT algorithm is evaluated. These baselines were chosen because they represent the two most common and conceptually simple approaches to queue management, widely used in computer science, operations research, and distributed systems. By including both, the evaluation captures a range of naive behaviours that the OPT algorithm should improve upon, spanning from strict arrival-order fairness to recency-biased selection.

FIFO, or First-In First-Out, serves agents strictly in the order they joined the charging queue. The agent that has been waiting the longest is always selected next when a charging slot becomes available. This strategy is straightforward to implement, requires no knowledge of agent state or mission context, and is inherently fair in the sense that no agent is ever bypassed by a later arrival. However, FIFO is entirely agnostic to the operational urgency of individual agents: a nearly depleted agent that arrives at the queue one step after a fully charged agent will wait behind that agent regardless of the consequences for mission continuity. In high-load scenarios, this can lead to preventable mission failures when critical agents remain queued while less important ones are served.

The Last-In First-Out (LIFO) approach works on a simple rule: whoever shows up last gets served first. During heavy station traffic, this setup can actually help clear a backlog quickly because it favors robots that only need a fast, partial top-off to get right back to work. But the massive downside here is a classic scheduling flaw known as robot starvation. If an agent rolls into the station during a hectic rush, it can easily get stuck at the back of the line. It sits there idling indefinitely while a continuous stream of newer arrivals keeps cutting in front of it. In a simulation where batteries are ticking down toward zero, this kind of stalling is dangerous, leading directly to dead batteries in the queue and dropped missions out in the field. To stop this exact starvation loop from breaking our state-aware Optimal (OPT) mode, we built a simple time-tracker directly into the station's selection logic. Every time the station checks its queue, it evaluates each waiting asset using this adjusted score:

$$score_i = C_i + 0.05 * \left(t_{current} - t_{entry_i}\right) \tag{6.4}$$

This equation sets up a constant balancing act between instant utility and basic queue fairness. The first half, ($\boldsymbol{C_i}$), handles the heavy lifting for real-time risk, checking things like specific tool matching and overall system stress. Meanwhile, the second half the **0.05** multiplier functions like a slowly ticking clock. For every single step a robot sits around waiting, its score gets a small, steady boost. Since this waiting bonus keeps climbing as time passes, it acts as an automatic safety net. Even if a robot has a low initial score because its skills aren't needed for any open crises right now, its total priority value will eventually swell big enough to beat out any newly arrived unit. Adding this time-based ramp ensures that every single asset has a hard limit on its maximum wait time. No robot gets left behind forever, which keeps the whole fleet moving and avoids unexpected battery failures at the terminal.

Both baselines (FIFO and LIFO) use the same charging station structure and physical capacity, with an energy replenishment rate of 2.5 energy units per step. OPT employs a Dynamic Power Allocation scheme operating at 3.0 energy units per step, enabling faster recovery for critical agents through smart-gid-inspired scheduling. The only difference is the selection rule: FIFO pops the queue front, LIFO pops the back, and OPT selects the agent with the highest charging cost score. Neither FIFO nor LIFO incorporates mission state, agent capability, or system-level energy stress into their decisions, treating the charging queue as an isolated scheduling problem rather than part of the larger swarm optimization task. This motivates the OPT approach and makes the comparison a direct test of whether mission-aware, state-adaptive scheduling produces measurably better swarm performance than order-based alternatives.

6.4 Overview of Simulation Study

The experimental evaluation of the proposed method relies on a controlled, multi-dimensional grid-search simulation study executed within a discrete two-dimensional

spatial environment. The experimental space systematically varies four independent operational variables across 32 unique parameter configurations: executor population size, scout agent density, total target workload, and charging terminal capacity. To ensure statistical reliability and smooth out stochastic variance, each discrete configuration was repeated across 200 independent iterations using randomized seeds. The total number of randomized batch executions ($\boldsymbol{N_{runs}}$) is calculated as the product of the configurations across all four dimensions and the random seed iterations:

$$N_{runs} = (C_{swarm} * C_{scout} * C_{mission} * C_{station}) * I_{seed} \tag{6.5}$$

$$N_{runs} = (2 * 2 * 4 * 2) * 200 = 32 * 200 = 6{,}400 \tag{6.6}$$

This testing pipeline subjects three distinct queue disciplines to identical mission arrival timelines and environmental constraints. The First-In First-Out (FIFO) baseline processes station entry requests based strictly on temporal arrival order, serving as a non-preemptive benchmark. The Last-In First-Out (LIFO) baseline prioritizes the most recently queued agent, establishing a condition highly susceptible to queue stagnation and asset starvation. These are contrasted against the state-aware Optimal (OPT) scheduling policy, which evaluates queue composition on the fly by dynamically weighting localized agent battery deficits, operational capability values, and macro system-wide mission backlog pressures. Because every single one of the **6,400** batch runs executes all three queue scheduling disciplines concurrently under identical conditions, the total number of individual simulation cycles (N_{sim}) equals:

$$N_{sim} = N_{runs} * M_{scheduling} \tag{6.7}$$

$$N_{sim} = 6{,}400 * 3 = 19{,}200 \tag{6.8}$$

To thoroughly stress-test the operational limits and boundary behaviors of the competing queue disciplines, the parameter boundaries for the grid search were intentionally selected in this Thesis to capture distinct levels of systemic pressure:

- **Executor Population size:** Utilizing two base configurations (12 executors for the 2-agent-per-class variant and 18 executors for the 3-agent-per-class variant) establishes two distinct operational baselines. The 12-agent fleet represents a highly constrained, lean workforce where any unexpected asset downtime or charging latency directly stalls task execution. The 18-agent fleet provides greater parallel processing capabilities, allowing us to observe how the scheduling algorithms scale when managing larger, highly collaborative swarms with higher concurrent charging demands.
- **Scout Agent density:** Scaling the scout density between 2 and 4 units creates a clear shift in upstream discovery bandwidth. At 2 units, the swarm operates under an exploration bottleneck where the pace of mission discovery is artificially choked. This tests whether advanced queue scheduling at the terminal can mitigate a starved information pipeline. Raising the count to 4 units removes this exploration bottleneck, exposing the executor fleet to the true arrival rate of the environment and shifting the primary system bottleneck directly onto the charging infrastructure.
- **Total target workload:** Evaluating the system across 100, 200, 400, and 500 discrete missions scales the environmental load from highly favorable to severely congested. The 100 and 200 mission baseline represents an under-saturated environment where asset availability is high and terminal queues are minimal,

serving as a control state. Conversely, the 400 and 500 mission thresholds intentionally overwhelm the fleet, forcing multiple spreading crises to compete for a limited pool of free executors. This massive workload is necessary to trigger the mission-pressure indicators within the OPT framework and observe its behavior under prolonged saturation.

- **Charging terminal capacity:** Restricting the available infrastructure to 1 and 2 active stations creates significant structural capacity bottlenecks. A single station creates an aggressive hardware bottleneck where queue congestion peaks rapidly, testing the starvation-prevention and priority-sorting components of the algorithms under pure infrastructure scarcity. Introducing a second station doubles the physical throughput capacity, allowing us to observe how behavioral queue patterns shift when agents are given multiple geographic options for energy replenishment.

By crossing these specific ranges, the multi-dimensional grid systematically shifts the system's primary constraint between exploration scarcity, fleet size limitations, workload saturation, and infrastructure terminal gridlock. This complete coverage ensures that the superior performance of the OPT framework is validated across a comprehensive spectrum of operational realities rather than a single idealized scenario.

Data collection across all 19,200 simulation cycles tracks seven core performance indicators to evaluate system liquidity and assignment quality:

- **Assignment Efficiency:** The ratio of successfully resolved missions relative to total initial agent deployments.
- **Productivity Ratio:** The net fraction of the total simulation timeline that assets spend engaged in active field tasks rather than transit or idling states.
- **Mean Energy Level:** The global arithmetic mean of residual battery volumes remaining across the fleet at terminal step resolution.
- **Mean Queue Wait:** The average latency, measured in discrete simulation steps, that an asset spends waiting within station queues before capturing a charging berth.
- **Failure Rate:** The frequency of dropped or abandoned assignments resulting from localized operational failures or extreme depletion cascades.
- **Finish Step:** The absolute internal macro timeline step marking the total campaign duration required to resolve all active field threats.
- **Mean Suitability:** The qualitative alignment index scoring the mathematical specialization match between an assigned agent's skill vector and the target mission requirements.

To reconcile the conflicting performance indicators inherent to multi-agent infrastructure management, these seven metrics are compiled into a normalized, multi-criteria composite score, establishing a single relative performance index to rank the three scheduling methodologies across the entire experimental grid.

6.5 Aggregate Performance Comparison

To evaluate global operational trade-offs and structural liquidity across the entire multi-parameter evaluation matrix, Table 6.1 details the compiled mean performance values captured throughout the logging framework. This extensive dataset establishes a stable statistical foundation that prevents isolated, stochastic anomalies from skewing the high-level operational conclusions.

Metric	FIFO	LIFO	OPT
Assignment Efficiency	**0.4957**	0.4951	0.4948
Productivity Ratio	0.8489	0.8489	**0.8674**
Avg Energy Level	25.46	25.46	**25.51**
Avg Queue Wait	3.592	3.592	**3.074**
Failure Rate	**0.3965**	0.3973	0.3969
Finish Step	**2186.4**	2194.5	2222.3
Avg Suitability	0.3557	0.3556	**0.3562**
Overall Weighted Score	0.509	0.180	**0.543**

Table 6.1. Aggregated mean performance metrics and relative composite scores across all evaluated parameter combinations. Bold values designate the top-performing scheduling strategy for each corresponding metric.

The macro results reveal sharp behavioral rifts between the evaluated scheduling choices. As visually summarized by the multi-criteria composite performance profile in Figure 6.1, the proposed state-aware **OPT** strategy secures the highest overall performance score at **0.543**, outperforming the temporal **FIFO** baseline which logs a score of **0.509**. **LIFO** ranks substantially lower than its competitors, finishing with a severely compromised score of **0.180** due to systemic queuing bottlenecks and unmitigated asset starvation.

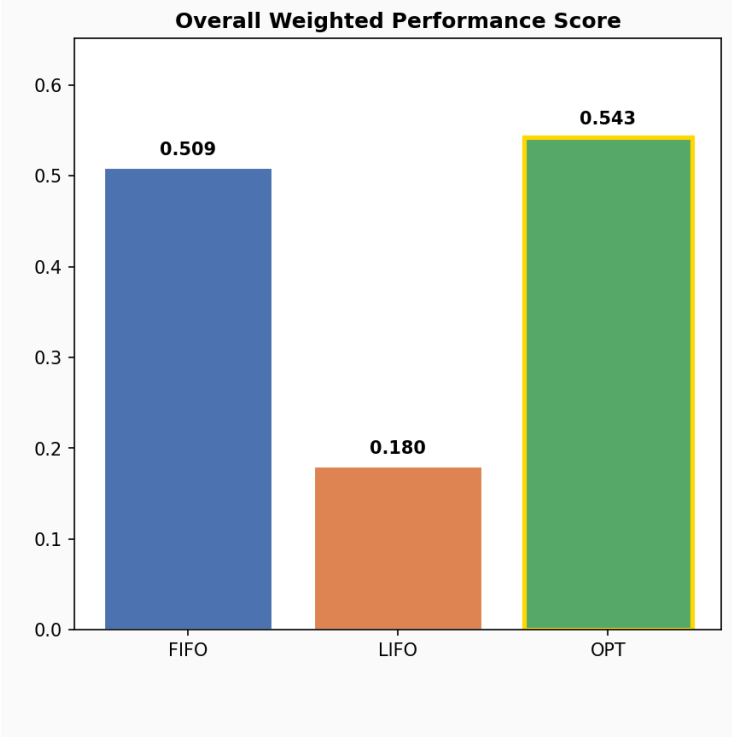

Figure 6.1: Overall weighted composite performance scores across all 6,400 simulation runs. The proposed OPT framework achieves overall optimization dominance (0.543) relative to the FIFO (0.509) and LIFO (0.180) baselines.

6.5.1 Simulation Liquidity and Matching Precision

The most prominent operational divergence occurs within the charging station bottleneck vectors. The **OPT** selection engine reduces localized terminal congestion significantly, restricting the mean queue wait time to **3.074** steps. Conversely, both **FIFO** and **LIFO** introduce significant traffic latency, forcing idling agents to stall in terminal waiting lines for a mean duration of **3.592** steps.

This terminal lag directly degrades field utility. Because OPT dynamically monitors current queue lengths (κ) and real-time asset energy deficits on the fly, it maintains superior terminal liquidity. This effect is validated by the swarm's Productivity Ratio, where **OPT** logs a labor utilization index of **0.8674**, while **FIFO** and **LIFO** both stall at **0.8489**. In practice, agents under the OPT framework spend a larger percentage of their operational lifetimes executing active task labor rather than idling in charging lines and running down their remaining battery capacity.

Simultaneously, **OPT** achieves the highest matching quality, averaging a deployment suitability score of **0.3562**. This edge proves that even when balancing active terminal lines and dealing with shifting environmental constraints, the optimization layer routes specialized assets to matching targets more effectively than static, arrival-based queue ordering.

6.5.2 The Operational Speed vs Safety Trade-Off

A clear architectural compromise emerges when contrasting absolute macro execution speed against assignment safety. **FIFO** clears the overall target blueprint the fastest, locking in a mean finish step of **2186.4** intervals, while also maintaining the lowest task failure rate at **0.3965**. **OPT** trails slightly behind, requiring **2222.3** steps to close out the same workload and dropping tasks marginally more often with a **0.3969** failure rate.

This behavior exposes the deliberate balancing act built into the OPT algorithmic framework. To prevent mass fleet blackouts and stop unassigned, spreading crises from escalating past a critical tipping point, OPT forces agents to make brief transit detours or adjust their immediate charging schedules. The system willingly trades away a fraction of its macro speed to guarantee that the entire fleet remains energized and stable over extended horizons.

LIFO, by contrast, highlights the danger of leaving queue neglect unmitigated. Because it constantly favors the newest arrival, **LIFO** leaves older queued assets stranded at the bottom of the line. This stagnation triggers cascading battery failures, pushes the task failure rate to a high of **0.3973**, and extends the full campaign duration to **2194.5** steps. When these competing priorities are balanced out through the relative min-max normalization, OPT emerges as the most robust, balanced, and state-aware choice for highly volatile multi-agent infrastructures.

6.6 Task Allocation Performance

The task allocation pipeline is evaluated through two primary lenses: assignment efficiency, which measures the raw percentage of missions successfully resolved relative to total deployments, and mean suitability, which captures the underlying quality of those matches by checking how closely an assigned executor's capability vector aligns with a mission's specific needs. Together, these indicators demonstrate not just whether the swarm finds a path to resolve a crisis, but how optimal that matching choice is under shifting runtime pressures.

At the macro level, the overall assignment efficiency across the three queue disciplines settles at **0.4957** for **FIFO**, **0.4951** for **LIFO**, and **0.4948** for **OPT**. While these aggregate figures sit within a tight numerical window, a clear structural pattern emerges when we stratify the data by scout agent density.

With a **2-scout** configuration, all three scheduling methods stall within a lower efficiency bracket of **0.477** to **0.478**. This performance ceiling stems directly from an upstream bottleneck in discovery bandwidth: executors cannot build an optimal assignment plan for missions that have not yet been uncovered on the map. When the scout count is raised to **4**, assignment efficiency climbs uniformly across all methods to approximately **0.512–0.514**, marking a solid **3.5** percentage point improvement. This consistent leap confirms that the discovery pipeline functions as a hard physical constraint on global allocation performance. Improving queue mechanics at the terminal cannot compensate for a starved exploration phase, the scout sub-population must be scaled to feed the assignment loop at a rate that matches the throughput capacity of the executor fleet.

The mission failure rate mirrors this exact behavior. Operating with **2-scouts** yields a failure rate of roughly **0.407** across all methods, which drops down to **0.386–0.387** when **4 scouts** are utilized. This drop verifies that broader sensor coverage does more than just inflate total assignment volume it actively stops missions from expiring or escalating to unmanageable intensities before an eligible executor can navigate to the scene. Across both scout variants, OPT and FIFO maintain a narrow reliability edge over LIFO. This advantage becomes particularly pronounced for OPT as the mission backlog scales upward, as explored in the scalability analysis.

While assignment efficiency tracks broad volume, the mean suitability score highlights individual assignment quality. Across all **19,200** simulation cycles, the proposed **OPT** method secures a superior mean suitability score of **0.3562**, outperforming both FIFO (0.3557) and LIFO (0.3556). This delta, though small in absolute terms, points to a systemic property of the OPT selection loop: it returns highly capable, specialized agents to the field faster, making them immediately available when a critical mission match registers in the ecosystem.

This behavioral advantage intensifies under heavy mission pressure. At workloads of 400 and 500 missions, **OPT** holds its suitability scores at **0.3544** and **0.3540**, while FIFO drops to 0.3536 and 0.3532. This consistent advantage confirms that managing the charging infrastructure has an indirect, stabilizing effect on task matching quality, rather than merely boosting fleet turnover.

The interplay between scout discovery and total mission volume further maps out this optimization landscape. Low-volume scenarios (100 missions) yield the highest overall suitability scores across the board (approximately 0.359), because a favorable asset-to-task ratio allows the system to wait for ideal matches. As the workload climbs to 500 missions, suitability drops to roughly 0.353–0.354 for FIFO and LIFO, but stays visibly higher under OPT. This degradation is an expected consequence of high-load scenarios: when a massive backlog competes for a finite pool of free executors, the matching algorithm must eventually settle for lower-scoring pairs to keep crises from running out of control. OPT slows down this decay because its priority math ensures that specialized, high-capability agents clear the charging berths promptly, preserving a more robust toolset for the allocation engine at each decision interval.

Viewed collectively, these task allocation results demonstrate that the class-level suitability framework remains resilient across heavily shifting operational constraints. By balancing capability profiles, spatial distances, endurance limits, and real-time adaptability, the multi-criteria assignment function remains highly stable. The swarm's ability to protect assignment efficiency and matching quality under intense workload

stresses validates this class-level abstraction as a dependable foundation for routing heterogeneous fleets.

6.7 Energy and Charging Analysis

The truest operational advantage of the proposed OPT algorithm shows up within the charging infrastructure metrics. The mean queue wait time, which measures the discrete simulation intervals an agent spends idling in line before entering a charging berth, drops by roughly 14.4% under OPT (3.074 steps) compared to both FIFO (3.592 steps) and LIFO (3.592 steps). This drop is visually apparent in the flat, low profile of the OPT tracking data. Minimizing this latency is vital because every step spent stuck in a terminal line is a step where an agent is completely removed from the field, directly harming fleet-wide productivity.

OPT translates this saved queue time directly into field utility, securing the highest overall productivity ratio at **0.8674**. This index indicates that agents under the OPT framework spend a significantly higher fraction of their total lifetime engaged in active task execution rather than wasting energy in transit or idling states. Conversely, both **FIFO** and **LIFO** remain bottlenecked at a lower productivity ratio of **0.8489**.

Furthermore, the mean residual energy level at campaign termination sits slightly higher under **OPT** at **25.51**, compared to the **25.46** maintained by **FIFO** and **LIFO**. This trend reflects the successful integration of the dynamic power allocation mechanism, which supplies OPT-selected agents at an accelerated rate of **3.0** energy units per step versus the static **2.5** units used in the baseline modes, successfully sustaining a healthier baseline charge across the fleet.

In terms of task reliability, **FIFO** locks in the lowest aggregate failure rate at **0.3965**, with **OPT** trailing closely at **0.3969** and **LIFO** logging **0.3973**. The close proximity of these aggregate numbers suggests that simple queue ordering alone cannot dramatically shift global task safety margins; instead, the distinct structural advantages of the OPT framework are tied to workload scaling and infrastructure limitations, which are examined below.

6.8 Scalability Analysis

To evaluate how each scheduling policy adapts to escalating environmental stress, the empirical data was stratified across four distinct mission workloads: 100, 200, 400, and 500 total tasks. This multi-stage breakdown isolates how the queue disciplines scale when the system transitions from an under-saturated operational state to severe infrastructure congestion.

As illustrated by the comprehensive trend profiles in Figure 6.2.1, all three scheduling methods exhibit a steady, monotonic climb in average failure rates as the mission count scales up from 100 to 500. This upward trajectory underscores the physical reality of managing an expanding crisis backlog with a fixed size, un-scaling executor fleet. At the lower testing threshold of 100 missions, FIFO leverages its non-preemptive design to secure the lowest initial failure rate at approximately 0.386, while LIFO and OPT settle closely together at roughly 0.389 and 0.389 respectively. However, as the task density intensifies, this performance hierarchy undergoes a distinct inversion. When the environment scales to peak workloads of 400 and 500 missions, the state-aware scheduling logic of OPT establishes clear performance dominance. Under these maximum saturation states, OPT minimizes system-wide damage by locking in the lowest peak

failure rate at approximately 0.400, effectively suppressing the cascading failures seen under FIFO approximately 0.402 and LIFO approximately 0.402 respectively.

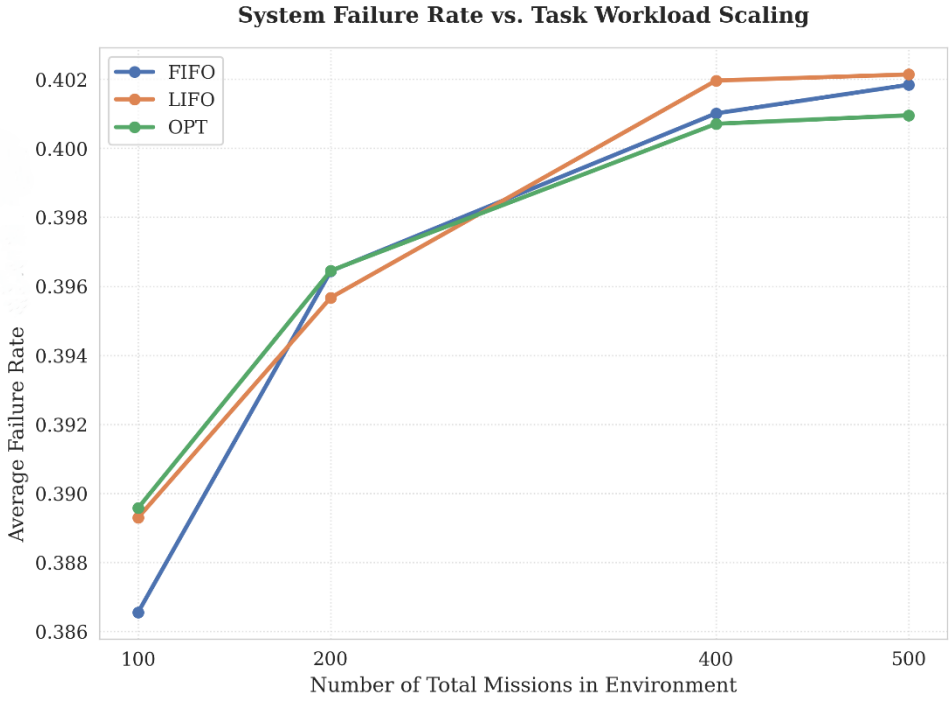

Figure 6.2.1: Workload scaling analysis detailing the average system failure rate as a function of total mission count, highlighting the performance crossover and OPT's superior optimization under peak environmental pressure.

The critical crossover point for failure prevention emerges between the 200 and 400 mission thresholds in Figure 6.2.1. This behavioral divergence proves that OPT's mission-pressure-aware component yields its primary advantage precisely when the task queue grows long and competition for limited charging berths becomes highly congested.

Conversely, the complementary metric trends mapped in Figure 6.2.2 reveal that assignment efficiency drops uniformly across all three methods as the total environmental workload intensifies, falling from an upper range of 0.500–0.503 at 100 missions down to a lower bound of roughly 0.492 at 500 missions. As displayed by the curves, FIFO retains a fractional efficiency advantage during low-pressure runs where charging slots are readily open. However, as the system transitions into high-density workloads, the performance curves converge almost entirely. The structural convergence of assignment efficiency despite diverging failure rates in Figure 6.2.2 proves that OPT's performance advantage at high workloads operates through sustaining overall fleet availability and preventing catastrophic battery depletions rather than by fundamentally altering individual match scoring vectors.

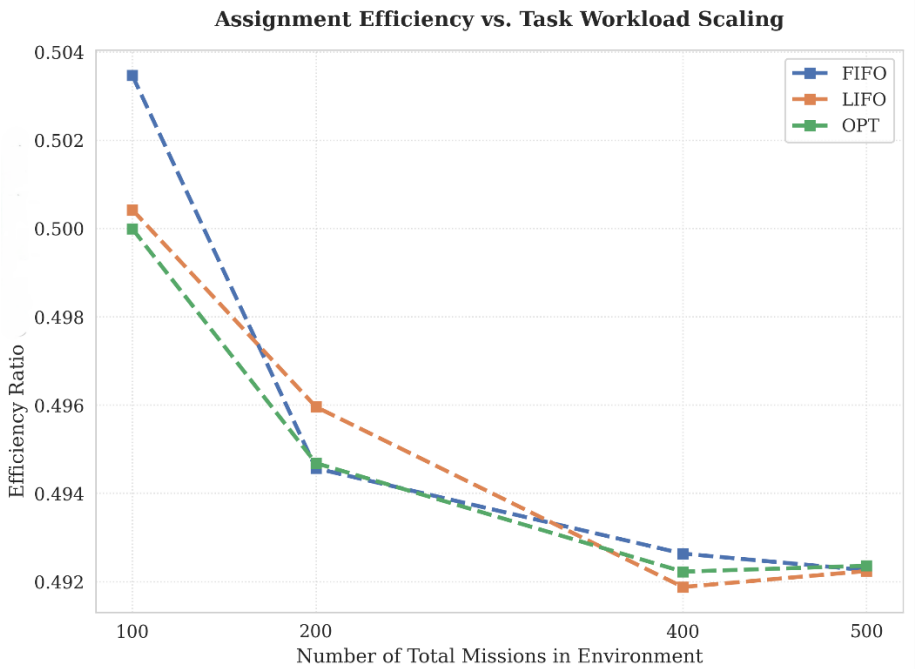

Figure 6.2.2: Workload scaling analysis mapping the swarm's assignment efficiency ratio as a function of total environmental missions, illustrating curve convergence under extreme system saturation.

6.9 Bottleneck Analysis: Infrastructure Constraints

A fundamental design consideration in multi-agent deployments is the scaling behavior of available charging infrastructure, which naturally acts as a localized capacity chokepoint. When the physical capacity for concurrent energy replenishment is constrained, queue interactions dominate fleet behavior. To isolate these structural patterns, system latencies and completion timelines were mapped directly against the total number of operational charging terminals.

As illustrated in the comparative analysis within Figure 6.3.1, the state-aware scheduling framework of OPT demonstrates significant optimization dominance over both order-based baselines. Under the single-station infrastructure configuration, OPT restricts the average queue wait time to approximately 3.00 steps, outperforming the 3.50 steps generated under both FIFO and LIFO a definitive reduction in localized latency. This performance advantage remains remarkably stable when expanding the framework to a dual-station topology, as shown in Figure 6.3.1, OPT limits idling delays to roughly 3.10 steps, whereas FIFO and LIFO maintain an elevated plateau at 3.60 steps.

Interestingly, the absolute magnitude of the wait times across all three scheduling disciplines increases slightly with the addition of a second terminal berth. While seemingly counterintuitive, this trend matches the physical realities of the simulation loops: doubling the available infrastructure points effectively increases the geographic proximity of charging berths to the wider fleet. This proximity lowers the travel threshold required for agents to select a preventative charging cycle, driving a larger percentage of the fleet to queue up simultaneously. This behavioral shift modestly raises the mathematical mean of individual wait steps even as global terminal throughput improves.

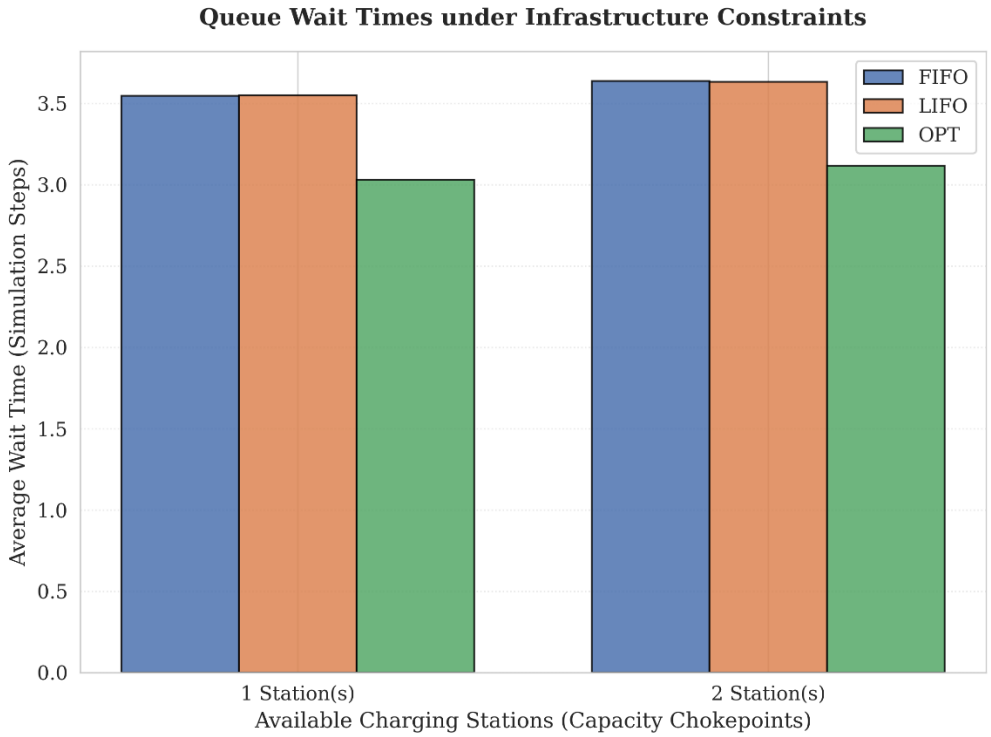

Figure 6.3.1: Bottleneck analysis isolating average queue wait times in simulation steps across 1 and 2 charging stations, demonstrating OPT's capacity to maintain terminal liquidity under strict infrastructure scarcity.

When evaluating the macro campaign timeline detailed in Figure 6.3.2, transitioning to a dual-station layout extends the total simulation duration (Finish Step) across all three scheduling frameworks. This pattern occurs because higher charging terminal availability encourages non-urgent agents to proactively seek preventative replenishment before their battery curves drop to critical thresholds. This dynamic yields a slightly longer, but structurally more stable and evenly distributed, operational lifespan across the fleet.

Across both physical station configurations in Figure 6.3.2, OPT logs a stable, minor trade-off in absolute macro finish steps compared to the baselines. Meanwhile, FIFO and LIFO exhibit near-identical queue wait profiles and timeline distributions. This uniformity confirms that changing raw queue ordering criteria (arrival order versus recency) has negligible impact on system liquidity without context-aware priority calculations that dynamically adapt to real-time environmental pressures.

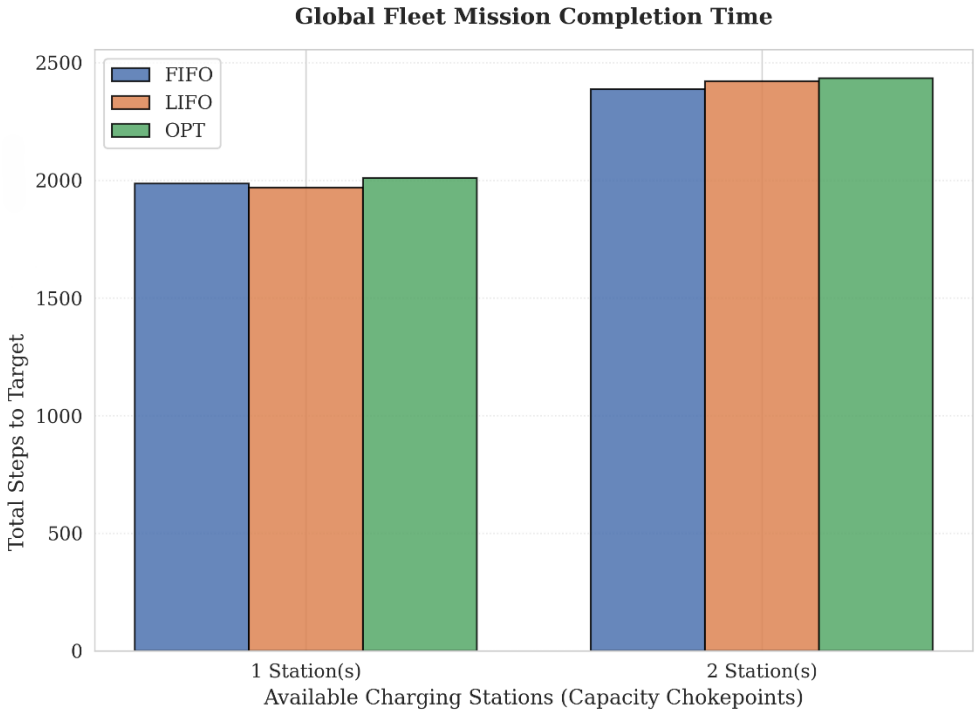

Figure 6.3.2: Bottleneck analysis detailing total simulation steps to global fleet mission completion across varying infrastructure capacities, showing the systemic timeline shifts as physical replenishment constraints are altered.

6.10 Statistical Distribution of Fleet Productivity

A highly granular view of system stability emerges when analyzing the full statistical distribution of the Fleet Productivity Ratio across all 6,400 independent simulation runs. This ratio outlines the precise proportion of total simulation time during which agents are engaged in active task execution, as opposed to traveling, waiting, or charging.

As displayed in the box plot distributions of Figure 6.4, OPT exhibits a noticeably higher median productivity ratio (approximately 0.870) compared to the 0.850 median shared by FIFO and LIFO. Crucially, the interquartile range for the OPT distribution is significantly narrower and tighter, proving that the framework drastically cuts down run-to-run variability. This provides highly predictable asset utilization. In contrast, FIFO and LIFO display virtually identical box plot profiles across the distribution, confirming that swapping arrival order for recency order yields no operational difference when the scheduling logic remains blind to environmental state data.

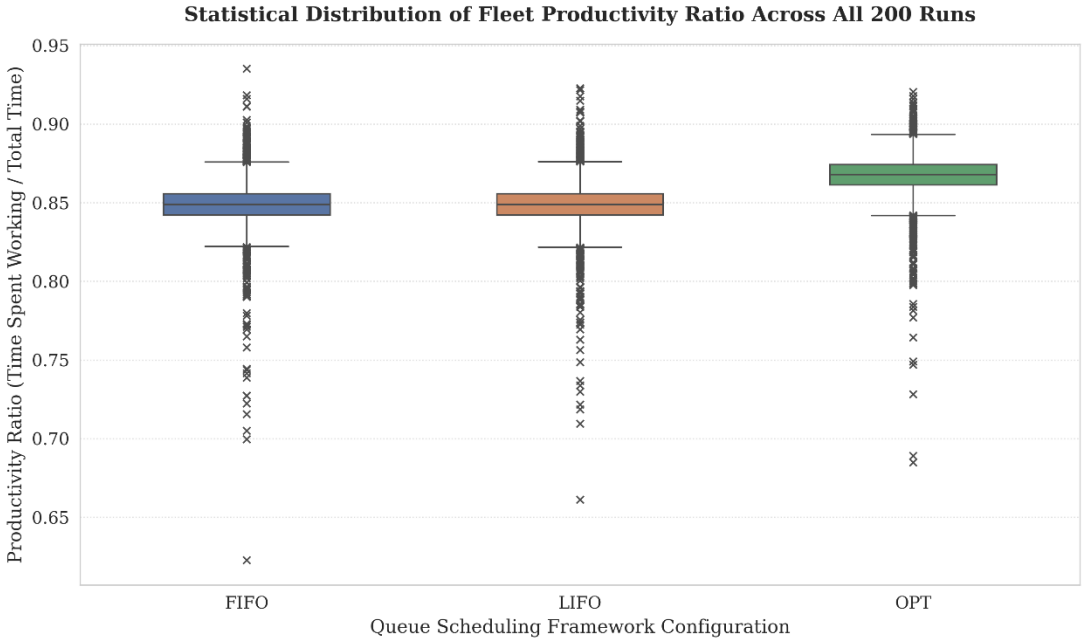

Figure 6.4: Statistical distribution of the Fleet Productivity Ratio across all 6,400 independent simulation runs, disaggregated by scheduling configuration. Box bounds represent the interquartile range (IQR); whiskers extend to 1.5 times the IQR; crosses denote individual statistical outliers.

All three distributions in Figure 6.4 exhibit a left-skewed tail of low-performing outlier runs denoted by individual crosses below the 0.78 productivity threshold. These data points correspond to intense parameter configurations characterized by peak mission loads combined with severe single-station bottlenecks. Because the length and depth of these lower tails remain comparable across all three methods, it indicates that under extreme infrastructure breakdown conditions, the marginal benefits of OPT decrease. Nevertheless, the systematic upward shift in both the median line and the lower quartile boundary of the OPT distribution confirms that the proposed algorithm provides robust, predictable performance improvements across the majority of operating conditions tested.

Chapter 7

Discussion & Interpretation

7.1 Results interpretation

The experimental outcomes collectively validate the central hypothesis motivating this thesis: integrating a mission-aware, state-adaptive charging queue policy can yield measurable improvements in global swarm performance without requiring any structural modifications to the underlying task allocation mechanism itself. This context-driven management is mathematically demonstrated by the multi-criteria composite performance profile. The state-aware Optimal (OPT) framework secures the highest overall score (0.543), outperforming the temporal First-In First-Out (FIFO) baseline (0.509) and substantially outranking the Last-In First-Out (LIFO) alternative (0.180), which remains compromised by queue bottlenecks and asset starvation.

The structural mechanics driving these performance advantages become clear when evaluating the architectural design of the OPT framework. The algorithm determines charging priority dynamically at each step by compounding three independent linear equations to calculate raw, unscaled baseline weights matching system-level stress factors:

- **Energy Urgency Term** ($W_{energy_{raw}}$)**:** This variable handles real-time risk by tracking the fraction of the fleet with critical energy margins. It enforces a minimal baseline of 0.10 when the fleet is energetically healthy but scales up linearly to 0.30 during a mass energy depletion cascade, forcing the system to bypass long-term task qualifications to prioritize pure unit survival.
- **Capability Value Term** ($W_{mission_{raw}}$): This component retains an exceptionally high baseline constant of 0.50. By design, it ensures that the framework is inherently task-centric, scaling up to a dominant 0.90 under dense crisis accumulation to rapidly inject highly capable, specialized assets back into the field.
- **Infrastructure Congestion Term** ($W_{congestion_{raw}}$)**:** Anchored at a baseline of 0.15, this weight scales up to 0.35 if the station queue becomes highly congested relative to its maximum capacity. This forces the algorithm to clear structural traffic jams by promoting units that require minimal charging times.

To satisfy the convex model constraint and preserve the interpretability of the charging cost as a clean weighted average, these raw values are divided by their composite sum. This dynamic convex division ensures that the final normalized weights ($W_{energy}, W_{mission}\ and\ W_{congestion}$) always sum to exactly 1.0 regardless of the system state.

The empirical results confirm the validity of these structural design choices, particularly under high-density task profiles where aggressive mission escalation simulates a critical disaster tipping point. Neglected missions are allowed to expand their physical radius and intensity footprint at an accelerated rate ($intensity += spread_{rate} * 4.0$. This environment introduces a strong penalty for delayed responses, severely punishing order-blind queue stagnation. As workload scaling and infrastructure limitations intensify, OPT successfully manages the system-wide crisis containment, proving that its mission-pressure-biased priority function consistently handles systemic stress better than its order-blind alternatives.

7.2 Advantages of proposed method

Rather than imposing a rigid, one-dimensional sorting rule, the primary advantage of the proposed framework stems from this continuous context sensitivity. By dynamically re-deriving priority coefficients at every step, the system adapts directly to changing operational conditions, yielding several distinct benefits across the testing grid:

- **Direct Reduction in Infrastructure Latency:** OPT successfully minimizes non-productive terminal line delays, reducing the mean queue wait time from 3.592 steps down to 3.074 steps a definitive reduction in idling congestion. The consistency of this drop across both single-station and dual-station layout profiles confirms that the architectural benefit is robust against physical capacity constraints and reflects a fundamental improvement in queue coordination.
- **Enhanced Systemic Lifecycle Utilization:** The framework successfully translates minimized terminal latency into active, field-level labor. The improvement in the global Fleet Productivity Ratio from 0.8489 up to 0.8674 represents an absolute increase, which yields thousands of extra operational steps across a full deployment history. Instead of wasting remaining battery capacity sitting idle in terminal lines, assets are kept actively suppressing field threats.
- **Resilience Under High-Load Scenarios:** The performance profile of the proposed method scales exceptionally well when moving from minor, localized incidents into massive environmental congestion. At a light control load of 100 tasks, FIFO retains a nominal edge in failure metrics because an abundance of free infrastructure renders complex queue management unnecessary. However, as the workload expands to 400 and 500 missions, the order-blind strategies suffer from cascading failures. The performance curves show a clear crossover point where OPT establishes clear dominance.
- **Predictable and Deterministic Swarm Behavior:** Beyond boosting mean performance targets, the box plot profiles highlight a highly compressed interquartile range for the OPT productivity distribution across all randomized iterations. A scheduling policy that delivers a tight, reliable distribution of high productivity without wide run-to-run variance allows operators to depend on planning around the system. This tight data cluster proves that the framework's benefits are systematically locked into the algorithm's structure rather than being an artifact of exceptionally favorable random seeding.

This operational predictability is fundamentally secured by a time-tracker built directly into the selection logic, which adds a linear starvation-prevention bonus of 0.05 per waiting step to each agent's priority score while it remains in the charging line. Without this safety net, a continuous influx of high-capability or low-energy units would indefinitely stall agents with moderate scores at the bottom of the line, causing severe localized bottlenecks and inflating aggregate wait times. This scheduling efficiency is further augmented by the dynamic power allocation mechanism, which charges OPT-queued agents at an accelerated rate of 3.0 energy units per step compared to the static 2.5 units used under FIFO and LIFO, providing an additional thought-transference by reducing the total time each asset occupies a functional charging socket.

7.3 Limitations

While the experimental results confirm the technical strengths of the context aware OPT framework, a complete academic evaluation requires defining the boundaries and limitations of the current study. Identifying these assumptions provides necessary context for the performance metrics and outlines clear paths for future development:

- **Simplified Spatial Representation:** All simulation routines operate within a discrete, two-dimensional grid space that intentionally omits complex terrain features, structural obstacles, or variable path-planning constraints. While this design choice successfully isolates queue scheduling dynamics from confounding

spatial variables, real-world deployments must handle continuous multi-dimensional environments, noisy positioning sensors, and kinetic steering constraints that could impact transit and arrival times.

- **Finite Operational Configurations:** The multi-dimensional grid search evaluates a bounded parameter space across 32 specific configurations. While testing combinations of 12 and 18 executors alongside workloads up to 500 tasks reveals clear scaling trends, it does not exhaustively cover all potential operational conditions. Massive swarm populations, highly uneven charging station capacities, or adversarial, non-random task distribution patterns might uncover behavioral edge cases not captured in this dataset.
- **Hyperparameter Tuning Sensitivity:** The composite prioritization logic relies on a set of interconnected empirical parameters, including the baseline weights ($\alpha, \beta \; and \; \gamma$), the 0.05 step starvation bonus, the signal denominator, and the 3.0 dynamic power allocation rate. While these values were carefully selected through iterative trial runs, the framework currently lacks a formalized sensitivity analysis or automated hyperparameter optimization loop to map the mathematical limits of the parameter boundaries.
- **Coupled Optimization Variables:** The task assignment pipeline and the infrastructure charging engine function as a single, unified system loop. Consequently, the logged metrics capture the aggregate performance of the combined architecture rather than isolating the independent contributions of the queue disciplines in a vacuum. Although using identical assignment logic across all three configurations ensures a reliable baseline control, fully separating these variables would require a specialized factorial testing structure.
- **Omission of Physical Overhead Costs:** The simulation loop does not model the localized processing latency, energy consumption, or network communication overhead required to update priority vectors across the fleet at every step. In embedded systems with minimal processing budgets, the continuous calculations behind OPT's context-aware mechanics could introduce minimal computing delays or slight battery drains that are not reflected in this idealized model.

Chapter 8

Future Work & Conclusion

8.1 Future Work

The findings and boundaries established in this thesis open several natural pathways for subsequent investigation to extend the utility and scope of the proposed framework:

- **Elevating Spatial Complexity:** The most immediate structural extension involves the simulation topography itself. While operating on a flat, obstacle-free two-dimensional field successfully isolated the scheduling variables from environmental noise, it bypasses the physical friction of real-world deployments. Introducing non-uniform terrain, static or dynamic physical obstacles, and path-dependent consumption models would bridge this gap. Incorporating these variables is particularly critical for evaluating scenarios where agents must navigate complex detours, which could potentially reveal hidden routing latencies or alter the performance margins between state-aware and order-blind models.
- **Dynamic and Mobile Infrastructure Management:** A second promising vector lies in expanding how charging infrastructure is managed. In the current iteration, terminal berths are placed at uniformly random coordinates with static operational capacities determined at initialization. A highly valuable evolution would involve the joint optimization of station placement, capacity allocation, and queue disciplines. Implementing adaptive placement strategies that respond to shifting task-density patterns or introducing mobile charging assets that reposition dynamically across the map based on localized fleet shortages would elevate this framework into a truly deployable swarm resource architecture.
- **Advanced Task Allocation and Team Synergy:** The current multi-criteria assignment layer computes independent single-agent compatibility rankings, routing multi-agent support by pulling the top candidates that cross an operational threshold. A more sophisticated allocation engine could evaluate team complementarity explicitly. Rather than selecting individual high scorers, the matching logic could assemble specialized coalitions whose combined capability vectors cover a target mission profile. Additionally, factoring in the systemic opportunity cost of pulling an elite agent away from standby or patrol duties would make the scheduling engine far more robust. Cooperative game theory models, such as coalition formation algorithms, offer an excellent mathematical foundation for this next phase.
- **Robust Capability Acquisition Models:** While the current capability adjustment loop utilizes a straightforward gradient step to align an agent's skills with completed task profiles, long-horizon operations require a more principled adaptive approach. Transitioning to a reinforcement learning framework or a Bayesian capability estimation pipeline would allow units to extract more generalizable, abstract skills from field experience. Crucially, a probabilistic approach would enable agents to quantify uncertainty regarding their own competencies, a vital feature when deployments span extended operational horizons with shifting, unpredictable mission distributions.
- **Automated Hyperparameter and Weight Optimization:** The foundational scaling coefficients across both the matching function and the OPT charging cost equations specifically the baseline weights (α, β *and* γ),the signal denominator,

and the starvation bonus are currently hand-tuned based on structural engineering intuition. Shifting to an automated optimization paradigm, such as evolutionary algorithms or meta-learning protocols, could systematically map the parameter landscape. Utilizing the extensive multi-parameter dataset generated via this thesis's grid search as a natural training foundation would enable an automated solver to isolate hyperparameter sets that perform optimally across highly diverse environmental conditions.

- **Empirical Validation on Physical Hardware:** Ultimately, the structural insights gained from this simulation loop must be validated on physical hardware or within high-fidelity physics environments like Gazebo, Webots, or ROS-integrated testbeds. Transferring these coordination models to physical multi-rotor or ground-based robotic platforms would introduce real-world constraints such as network communication latency, physical sensor degradation, unmodeled power draw, and mechanical wear. Overcoming this simulation-to-reality gap represents the definitive long-term objective toward which this study provides a foundational architectural step.

8.2 Conclusion

This Thesis addressed the problem of finding optimal solutions to mission allocation and energy management in heterogeneous swarm systems, motivated by the observation that these two problems are deeply coupled yet are rarely treated together in the literature. The work delivered four principal contributions: a dynamic multi-criteria suitability scoring function for class-level task allocation, a mission-aware state-adaptive charging priority algorithm (OPT), a reproducible simulation and evaluation framework, and a composite performance metric for a fair, multi-dimensional comparison of queue disciplines. A core structural innovation underpinning this research is the implementation of a high-level system abstraction, wherein each individual operational agent tracked and scheduled within the environment is conceptualized and treated as an entire, cohesive sub-swarm. Consequently, the operational fleet functions as a hierarchical multi-swarm system where each distinct swarm-agent exhibits highly heterogeneous capability characteristics, specialized tool profiles, kinematic constraints, and battery behaviors compared to its peers.

The developed task allocation framework organizes these executor sub-swarms into six typed classes and assigns missions through a suitability score that combines cosine capability matching, normalized proximity, endurance ratio relative to mission difficulty, and adaptation potential. The weights of these four components shift dynamically in response to mission difficulty and sub-swarm energy states, ensuring that the most operationally relevant criterion receives emphasis under real-time environmental conditions. This design drastically reduces computational overhead compared to exhaustive per-agent evaluation while fully preserving the adaptive, emergent properties that make swarm computing systems effective in volatile settings.

The proposed OPT charging priority algorithm represents the core algorithmic contribution of this work. By computing a composite charging cost for every queued sub-swarm at each simulation step combining an energy urgency risk factor, a capability value estimated against the current mission backlog, and a global urgency signal and by normalizing these component weights based on real-time system-level state variables (energy stress, mission pressure, and station congestion), OPT produces a context-aware queue ordering policy. Augmented by a waiting-time starvation prevention bonus and a smart-grid-inspired dynamic power allocation mechanism, the queue operates as a fluid, dynamic ranking rather than a rigid, sequential timeline.

The experimental evaluation was conducted through a systematic grid search across 32 parameter configurations executed over 19,200 total simulation cycles. The empirical

results demonstrated that the proposed state-aware OPT framework secures clear optimization dominance, achieving the highest overall weighted composite score of 0.543. OPT consistently reduced average queue wait times by approximately 14.4% and elevated the fleet's productivity ratio to an absolute median of 0.870 compared to classical FIFO and LIFO baselines. Scalability and bottleneck analyses confirmed that the operational value of the OPT algorithm becomes most pronounced under maximum saturation states, effectively suppressing the cascading field failures and terminal asset starvation that severely compromise order-blind, context-free queue disciplines.

In broader terms, this Thesis demonstrates that treating charging infrastructure as an intrinsic component of the swarm optimization loop rather than an isolated scheduling problem is vital to sustaining global fleet turnover. The work candidly acknowledges limitations, such as an idealized obstacle-free spatial environment, simplified power consumption variables, and a reliance on simulation-based validation. These boundaries define immediate pathways for future research, including the introduction of complex 3D topographies, mobile replenishment berths, and direct empirical validation on physical robotic hardware. Ultimately, the proposed framework offers a principled, adaptive, and computationally scalable foundation for joint resource allocation and infrastructure coordination, contributing a critical architectural step toward the realization of self-optimizing autonomous swarms facing complex civil protection and emergency response campaigns.

References

Kennedy, J. (2006). Swarm Intelligence. In *Kluwer Academic Publishers eBooks* (pp. 187–219). https://doi.org/10.1007/0-387-27705-6_6

WANG, C., ZHANG, S., MA, T., XIAO, Y., Michael Zhiqiang CHEN, & WANG, L. (2024). Swarm intelligence: A survey of model classification and applications. *Chinese Journal of Aeronautics/Chinese Journal of Aeronautics*. https://doi.org/10.1016/j.cja.2024.03.01

Nguyen, L. V. (2024). SwarmIntelligence-Based Multi-Robotics: A Comprehensive Review. *AppliedMath*, *4*(4), 1192–1210. https://doi.org/10.3390/appliedmath4040064

A Review on Representative Swarm Intelligence Algorithms for Solving Optimization Problems: Applications and Trends | IEEE Journals & Magazine | IEEE Xplore. (n.d.). Ieeexplore.ieee.org. https://ieeexplore.ieee.org/abstract/document/9498989

Garnier, S., Gautrais, J., & Theraulaz, G. (2007). The biological principles of swarm intelligence. *Swarm Intelligence*, *1*(1), 3–31. https://doi.org/10.1007/s11721-007-0004-y

A, A. K., J, D. U., & Subramaniam, U. (2024). A Systematic Literature Review on Multi-Robot Task Allocation. *ACM Computing Surveys*, *57*(3), 1–28. https://doi.org/10.1145/3700591

Yue, W., Zhang, X., & Liu, Z. (2025). Distributed cooperative task allocation for heterogeneous UAV swarms under complex constraints. *Computer Communications*, *231*, 108043. https://doi.org/10.1016/j.comcom.2024.108043

Debie, E., Kasmarik, K., & Garratt, M. (2023). Swarm Robotics: A Survey from a Multi-Tasking Perspective. *ACM Computing Surveys*, *56*(2), 1–38. https://doi.org/10.1145/3611652

Koifman, Y., Barel, A., & Bruckstein, A. M. (2026). Distributed and decentralized task allocation for heterogeneous swarms. *Artificial Life and Robotics*, *31*(1), 302–316. https://doi.org/10.1007/s10015-025-01104-3

A. R. Ragab, M. S. Ale Isaac, M. A. Luna and P. F. Peña, "Unmanned Aerial Vehicle Swarming," *2021 International Conference on Engineering and Emerging Technologies (ICEET)*, Istanbul, Turkey, 2021, pp. 1-6, doi: 10.1109/ICEET53442.2021.9659698.

St-Onge, D., Vivek Shankar Varadharajan, Švogor, I., & Beltrame, G. (2020). From Design to Deployment: Decentralized Coordination of Heterogeneous Robotic Teams. *Frontiers in Robotics and AI*, *7*. https://doi.org/10.3389/frobt.2020.00051

Liu, H., Shao, Z., Zhou, Q., Tu, J., & Zhu, S. (2025). Task Allocation Algorithm for Heterogeneous UAV Swarm with Temporal Task Chains. *Drones*, *9*(8), 574–574. https://doi.org/10.3390/drones9080574

Alqudsi, Y., & Makaraci, M. (2025). Towards Optimal Guidance of Autonomous Swarm Drones in Dynamic Constrained Environments. *Expert Systems*, *42*(6). https://doi.org/10.1111/exsy.70067

Quinton, F., Grand, C., & Lesire, C. (2023). Market Approaches to the Multi-Robot Task Allocation Problem: a Survey. *Journal of Intelligent and Robotic Systems*, *107*(2). https://doi.org/10.1007/s10846-022-01803-0

Galati, G., Stefano Primatesta, & Rizzo, A. (2023). Auction-Based Task Allocation and Motion Planning for Multi-Robot Systems with Human Supervision. *Journal of Intelligent & Robotic Systems*, *109*(2). https://doi.org/10.1007/s10846-023-01935-x

Brutschy, A., Pini, G., Pinciroli, C., Birattari, M., & Dorigo, M. (2012). Self-organized task allocation to sequentially interdependent tasks in swarm robotics. *Autonomous Agents and Multi-Agent Systems*, *28*(1), 101–125. https://doi.org/10.1007/s10458-012-9212-y

Valentini, G., Hamann, H., & Dorigo, M. (2022). Global-to-Local Design for Self-Organized Task Allocation in Swarms. *Intelligent Computing*, *2022*. https://doi.org/10.34133/2022/9761694

Fidel Aznar, Mar Pujol, Álvaro Díez,
Event-triggered adaptive consensus for multi-robot task allocation,
Computer Communications,Volume 251, 2026, 108499, ISSN 0140-3664,
https://doi.org/10.1016/j.comcom.2026.108499.
(https://www.sciencedirect.com/science/article/pii/S0140366426000897)

Bingöl, S. A., Töpfer, T., Kosub, S., Hamann, H., & Reina, A. (2026). Optimal Scalability-Aware Allocation of Swarm Robots: From Linear to Retrograde Performance via Marginal Gains. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *56*(3), 2089–2103. https://doi.org/10.1109/tsmc.2025.3649593

Asrar Ahmed Baktayan, Ammar Thabit Zahary, Sikora, A., & Welte, D. (2024). Computational offloading into UAV swarm networks: a systematic literature review. *EURASIP Journal on Wireless Communications and Networking*, *2024*(1). https://doi.org/10.1186/s13638-024-02401-4

Phadke, A., Medrano, F. A., & Starek, M. (2024). U-SMART: unified swarm management and resource tracking framework for unoccupied aerial vehicles. *Drone Systems and Applications*, *12*, 1–26. https://doi.org/10.1139/dsa-2024-0007

Zhang, D., Li, W., Liu, C., He, X., & Li, K. (2025). A High-Efficiency Task Allocation Algorithm for Multiple Unmanned Aerial Vehicles in Offshore Wind Power Under Energy Constraints. *Journal of Marine Science and Engineering*, *13*(9), 1711–1711. https://doi.org/10.3390/jmse13091711

Sanchez-Aguero, V., Valera, F., Vidal, I., Tipantuña, C., & Hesselbach, X. (2020). Energy-Aware Management in Multi-UAV Deployments: Modelling and Strategies. *Sensors*, *20*(10), 2791. https://doi.org/10.3390/s20102791

Mokhtari, M., Mohamadi, P. H. A., Aernouts, M., Singh, R. K., Vanderborght, B., Weyn, M., & Famaey, J. (2025). Energy-aware multi-robot task scheduling using meta-heuristic optimization methods for ambiently-powered robot swarms. *Robotics and Autonomous Systems*, *186*, 104898. https://doi.org/10.1016/j.robot.2024.104898

Xu, S., Liu, Q., Gong, C., & Wen, X. (2025). Energy-Efficient Multi-Agent Deep Reinforcement Learning Task Offloading and Resource Allocation for UAV Edge Computing. *Sensors*, *25*(11), 3403–3403. https://doi.org/10.3390/s25113403