

Article

Toward an Effective Organizational Adaptation in Multi-Agent Systems: A Model Based on Markov Decision Processes

Narimane Sahel ¹, Varun Gupta ^{2,3,*}, Toufik Marir ¹, Maroua Bouzid ⁴ and Chetna Gupta ⁵

¹ Research Laboratory on Computer Science's Complex Systems (ReLa(CS)²), University of Oum El Bouaghi, Oum El Bouaghi 04000, Algeria; narimane.sahel@univ-oeb.dz (N.S.); marir.toufik@univ-oeb.dz (T.M.)

² Multidisciplinary Research Centre for Innovations in SMEs (MrciS), Gisma University of Applied Sciences, 14469 Potsdam, Germany

³ International Services for Society Research Group (INSERAS), University of Alcalá, Plaza de la Victoria 2, 28802 Alcalá de Henares, Madrid, Spain

⁴ GREYC Laboratory, University of Caen Normandie, 14000 Caen, France; maroua.bouzid-mouaddib@unicaen.fr

⁵ School of Computer Science Engineering and Technology, Bennett University, Greater Noida 201301, India; chetna.gupta@bennett.edu.in

* Correspondence: varun.gupta@gisma.com

Abstract

Coordinating agents in dynamic and uncertain environments remains a fundamental challenge in multi-agent systems (MAS) research, particularly in contexts where the composition of agent organizations directly affects overall system performance. While significant effort has focused on task allocation and individual agent planning, predicting the systemic impact of organizational changes and selecting optimal organizational structures under uncertainty remain less explored in MAS. This paper addresses this challenge by introducing a decision-making framework that models structural reorganization as a Markov Decision Process (MDP), where actions represent organizational structures rather than individual agent behaviors, and organizational selection is guided by the anticipated impact on the overall system state. The proposed model captures the stochastic dynamics of multi-agent intervention and diverse agent capabilities through a probabilistic transition function, while a reward function guides the selection of coalition structures that maximize operational effectiveness. The framework is solved using value iteration and evaluated on the RoboCup Rescue simulation platform. Results show that the derived optimal policy identifies, at each decision step, an appropriate coalition structure that reduces system degradation while efficiently utilizing available agents.

Keywords: multi-agent systems; organizational structure; coalition formation; Markov Decision Process; sequential decision-making; value iteration; RoboCup Rescue



Academic Editor: Fernando De la Prieta Pintado

Received: 8 May 2026

Revised: 19 June 2026

Accepted: 23 June 2026

Published: 26 June 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Multi-Agent Systems (MAS) provide a powerful conceptual framework for modeling and designing complex distributed systems composed of autonomous entities capable of interacting and cooperating to achieve shared goals [1]. In such systems, coordination and collective behavior strongly depend on how interactions among agents are structured. From this perspective, the notion of organization plays a central role, as it defines relationships between agents, role distribution, and the mechanisms governing collective decision-making. Two main approaches are commonly distinguished: Agent-Centered Multi-Agent Systems (ACMAS), which emphasize emergent behaviors arising from local

interactions; and Organization-Centered Multi-Agent Systems (OCMAS), which rely on explicit organizational structures to coordinate collective actions [2,3]. Organization is therefore a key lever for managing complexity, improving performance, and enhancing adaptability in dynamic environments [4,5].

However, whether organization is implicit, as in ACMAS, or explicitly defined, as in OCMAS, it is generally characterized by a dynamic nature. MAS typically operate in dynamic environments characterized by unpredictable changes in resources, objectives, agent populations, and communication conditions [6,7]. To maintain efficiency and coherence, organizational structures must therefore adapt over time. Organizational dynamics enable systems to reconfigure roles, relationships, or structures in response to environmental changes [8]. This adaptability is a key requirement for designing robust MAS that balance individual autonomy and collective stability [9].

Accordingly, recent research has focused on designing dynamic organizational mechanisms that allow MAS to not only operate within a given structure but also to modify it when necessary. This leads to organizationally autonomous MAS capable of structural self-reconfiguration to sustain performance under changing conditions.

Several studies have investigated organizational dynamism at different levels, including structural reorganization (e.g., transitions between hierarchical and federated organizations) [10,11], role evolution through reassignment or redefinition of responsibilities, and norm adaptation through modification of rules and sanctions [12,13]. However, most existing approaches focus on generating a valid organizational change without explicitly evaluating alternative configurations or their system-wide consequences. Hence, the intelligence analytics of an organization, in order to take correct and effective decisions, is very important. In this context, this work proposes an approach based on a well-known intelligent technique (the Markov Decision Process) to select effective adaptation according to the current situation of the organization.

In dynamic environments, multiple organizational adaptations may appear locally beneficial, but their global impact on system performance can vary significantly. Therefore, MAS require mechanisms capable of predicting and comparing the outcomes of different organizational configurations in order to select the most effective one. To address this challenge, we model organizational reconfiguration as a Markov Decision Process (MDP), where states represent the different situations or configurations of the environment, and actions correspond to possible organizational configurations. A reward function is defined based on system performance objectives to evaluate each configuration, and actions correspond to structural transformations. A reward function is defined based on system performance objectives to evaluate each configuration. The MDP is solved using value iteration to compute the optimal policy, which maps each state to the organizational configuration that maximizes expected performance [14]. This approach enables the system to not only react to changes but also to anticipate their effects and proactively select optimal organizational configurations.

The paper is structured as follows. Section 2 presents the theoretical background and related work on organization in MAS. Section 3 describes the proposed methodology and model. Section 4 introduces the RoboCup Rescue case study and reports experimental results. Section 5 provides a discussion of the obtained results. Finally, Section 6 concludes the paper and discusses future work.

2. Related Works

The study of organizational change in MAS has received increasing attention in recent years. As MAS become more complex and dynamic, the ability to adapt their organization to new environments or goals has become a critical challenge. Several approaches have

been proposed to address this issue, ranging from static organization models to more flexible and adaptive frameworks. This section reviews the main contributions in this field and discusses how they relate to the objectives of our work.

Organizational change in MAS can occur at different levels and under different conditions. In some approaches, organizational changes are predefined at design time, meaning that the possible adaptations are explicitly modeled and constrained within the system architecture. Such static change mechanisms allow predictability and control but limit flexibility. Conversely, dynamic organizational change enables agents or the organization itself to modify structures, roles, and interaction patterns at runtime, in response to environmental or internal factors. These changes can be agent-centered, where individual agents initiate and negotiate adaptations, or organization-centered, where modifications are guided by global policies or organizational reasoning mechanisms [6].

In a MAS, organization refers to the set of mechanisms that structure the interactions among autonomous agents. Such organization may manifest through different dimensions: an organizational paradigm or structure (for instance, hierarchy, coalition, or team) [10], a set of roles that define the agents' responsibilities and functions [15], or a system of norms that regulates their behaviors and interactions [16].

To formalize the concept of organization in MAS, several formal models have been proposed in the literature, explicitly representing organizational entities and their relationships. Among the most widely adopted are the AGR model [17] and its normative extension NorAGR [18], which defines organizations in terms of agents, groups, and roles; the MOISE model [19], which introduces a normative and structural specification of organizations; and its extension MOISE+ [20], which further enriches it with functional and deontic dimensions.

While organizational models provide explicit representations of organizational aspects, the changes they support are often predefined or limited by design-time specifications. In dynamic environments, organizations frequently need to adapt and reconfigure in more flexible ways, responding to changing conditions, agent behaviors, or objectives. Consequently, a variety of reconfiguration and adaptation mechanisms have been proposed in the literature to handle changes at different levels. In the following, we review the main approaches addressing each type of organizational change.

2.1. Research on Dynamic Norms and Sanctioning Mechanisms

Normative systems play a central role in regulating agent behavior within MAS. However, static norm definitions often fail to cope with the evolving nature of dynamic environments, motivating a growing body of research on dynamic norm synthesis, revision, and enforcement. Several works have focused on the automatic generation and revision of norms based on runtime observations. Dell'Anna et al. [21] propose a data-driven approach—referred to as DDNR—that synthesizes and revises conditional prohibitions by analyzing execution traces of the system. This allows norms to be adjusted according to the actual behavior of agents, rather than being fixed at design time. Similarly, Alechina et al. [22] address norm synthesis through the automatic generation of dynamic norms modeled as Mealy machines, where the constraints imposed on agents vary according to the current state of the system. This context-sensitive formulation enables genuine norm change at runtime, ensuring that system-level objectives remain satisfied under varying conditions.

Beyond norm synthesis, other approaches have explored the adaptation of norm enforcement mechanisms. Yan et al. [23] introduce a normative BDI agent architecture in which norms and sanctions are treated as first-class abstractions. Rather than relying on centralized control, each agent individually evaluates norms and decides how to enforce

sanctions, resulting in a distributed and adaptive enforcement mechanism. From a different perspective, Riad et al. [24] propose the use of multi-objective evolutionary algorithms to optimize norm parameters in heterogeneous MAS. By explicitly balancing system-level and agent-level values through a multi-criteria fitness function, this approach enables structural adaptation of norms to better reflect the preferences of diverse agent populations.

2.2. Research on Role Evolution and Dynamic Role Allocation

Role allocation is a fundamental mechanism for organizing agent behavior in MAS. While early approaches assigned roles statically at design time, recent works have increasingly explored dynamic role allocation and evolution, allowing agents to adapt their roles in response to changing tasks, environments, or collective performance. Several approaches address dynamic role discovery and assignment through learning-based mechanisms. Xia et al. [25] propose a framework in which an action encoder builds vector representations of agent behaviors, enabling roles to be classified not only by actions but also by their contributions to the common objective. This allows agents to switch roles dynamically according to their capabilities and task evolution. Similarly, Hou et al. [26] introduce a role policy learning mechanism in which agents generate and select roles based on local observations, with particular emphasis on social role evolution to improve collective efficiency in cooperative tasks. Beyond assignment, other works focus on the deeper evolution of agent profiles and emergent role structures. Lu et al. [27] propose a framework for LLM-based MAS in which agents maintain self-evolving profiles, meaning their roles and capabilities are updated at runtime based on task feedback, making the system more flexible in the face of changing requirements. Taking a predictive perspective, Goel et al. [28] propose a role emergence model grounded in multi-agent reinforcement learning, where roles are shaped not only by past experience but also by predictions of future agent behavior, enabling adaptive coordination and behavioral diversity.

2.3. Research on Organizational Structure Adaptation and Reconfiguration

Organizational structure adaptation refers to the ability of a MAS to modify its global architecture—including interaction protocols, agent networks, and coordination paradigms—in response to changing conditions or requirements. This area has received growing attention, with approaches ranging from paradigm switching to fully emergent structural reorganization. Early work by Barbosa et al. [29] proposes a self-organization mechanism for holonic MAS that enables switching between hierarchical and heterarchical organizational modes depending on the system state. This reconfiguration supports both flexibility and stability, though adaptation remains constrained to a set of predefined structures with no prediction of future impact. Along similar lines, Rahmanzadeh et al. [30] propose a runtime decision-making model that selects the most suitable organizational paradigm for a self-organizing MAS by evaluating system requirements against a utility metric. While this approach enables context-sensitive paradigm switching, the initial paradigm choice may be suboptimal and changes rely on past cases rather than predictive reasoning. More recent works have shifted toward formalizing and automating structural change. Köhler-Bußmeier et al. [31] introduce a nested Petri net formalism to model and trigger runtime modifications of organizational structures, including interaction protocols and agent networks. Notably, this approach incorporates probabilistic prediction of the impact of structural adaptations on system performance, though prediction remains limited to predefined organizational configurations without enabling autonomous global optimization. From a different angle, Yang et al. [32] propose a decentralized MAS architecture in which agents interact through a dynamic directed acyclic graph that evolves during execution to adapt to tasks and environment, eliminating the need for central control.

However, adjustments rely on past experience with no guarantee of global optimality or long-term planning. Finally, two recent approaches address structural reconfiguration through task allocation and communication graph adaptation. Wang et al. [33] propose a modular architecture with a separation between a control plane and a worker plane, enabling flexible task reallocation and real-time structural reconfiguration while minimizing the impact of changes through affinity mechanisms. Tastan et al. [34] introduce a framework in which the communication graph between agents is dynamically reconstructed according to agent contributions and contextual factors, enabling self-organized and adaptive organizational structures. While promising, this approach may face scalability challenges in large agent populations and lacks explicit mechanisms for predicting the long-term effects of graph restructuring.

Table 1 summarizes the cited approaches considering their type of change, impact, and limitations.

Table 1. Overview of Existing Approaches.

Ref.	Main Objective	Type of Change	Impact Consideration	Limitations
Dell’Anna et al. [21]	Automatically revise conditional norms from runtime behavior	Norm revision	Local impact on agent compliance	No long-term prediction; context-specific
Yan et al. [23]	Agent-centric norm evaluation and sanction enforcement	Sanction adaptation	Local impact only (perception-based)	No norm revision; risk of inter-agent inconsistency
Alechina et al. [22]	Synthesise dynamic norms to meet system-level objectives	Norm generation and revision	System-level impact considered	No long-term behavioral prediction
Riad et al. [24]	Optimise norm parameters for heterogeneous agents via multi-objective evolution	Norm parameter optimisation	System- and agent-level values considered	Limited to simple scenarios; no long-term prediction
Xia et al. [25]	Dynamically discover and assign roles based on action contributions to common objectives	Dynamic Role Discovery and Assignment	Role assignment adapts to context and team performance	Role instability across tasks; no long-term impact prediction
Hou et al. [26]	Enable agents to evolve social roles dynamically during cooperative learning	Dynamic Role Evolution	Roles adapt to team performance, improving cooperation	Simulation only; slow role evolution; no long-term prediction
Lu et al. [27]	Adapt agent roles and profiles at runtime based on task feedback	Dynamic role evolution and profile adaptation	Improves adaptability and performance across changing tasks	Limited scenarios; long-term role effects unknown
Goel et al. [28]	Discover roles that shape future behavior to improve coordination and diversity	Dynamic role and profile evolution	Improves adaptability and performance across changing tasks	Limited benchmarks; no long-term cross-task prediction
Barbosa et al. [29]	Design a self-organization mechanism enabling evolutionary and reconfigurable holonic systems to handle emergent requirements	Switch between hierarchical and heterarchical modes	Structure adapts to system state for flexibility and stability	Predefined structures only; no future impact prediction
Rahmanzadeh et al. [30]	Select at runtime the best organizational paradigm for a self-organizing MAS depending on context	Paradigm/organizational structure switching at runtime	Structure selected to maximise context-dependent utility	Suboptimal initial choice; relies on past cases; no prediction
Köhler-Bußmeier et al. [31]	Model runtime modifications of organizational structures using nested Petri nets	Structural change	Probabilistic prediction of the impact of adaptation on system performance	Prediction limited to predefined organizations; no autonomous global optimization

Table 1. Cont.

Ref.	Main Objective	Type of Change	Impact Consideration	Limitations
Yang et al. [32]	Remove central control through emergent multi-agent coordination	Collaborative interaction adaptation	Adjustments based on past experiences	No global optimality guarantee; no long-term planning
Wang et al. [33]	Ensure agent adaptability through dynamic task allocation	Dynamic task reallocation and structural reconfiguration	Minimizes the impact of reallocations through affinity mechanisms	Reactive only; no predictive mechanism
Tastan et al. [34]	Dynamically reconstruct agent communication graphs based on contributions and context	Dynamic communication graph restructuring	Adaptive, self-organized structure improves coordination	Scalability concerns; no long-term restructuring prediction

Across all three research axes, common limitations can be identified. Firstly, most approaches address only one dimension of organizational change—norms, roles, or structure—without considering their interactions. Secondly, adaptation mechanisms are mostly reactive, with little attention to predicting the effects of changes. Moreover, long-term systemic consequences of organizational modifications remain largely unexplored. These gaps motivate the need for a more integrated and predictive approach to organizational change in MAS.

In this context, we propose a new approach based on the Markov Decision Process (MDP) to analyze and select effective organizational adaptation according to the current situation and the desired goal. Obviously, the proposed framework will be compared with existing approaches in terms of impact scope, predictability, and global optimality in the Discussion Section 5.

3. Materials and Methods

3.1. Problem Formulation

Markov Decision Processes (MDPs) [14] provide a principled framework for sequential decision-making under uncertainty. An MDP is defined by a tuple (S, A, T, R, γ) , where S is the state space, A is the action space, $T(s, a, s')$ is the transition function, $R(s, a, s')$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. MDPs are particularly suited to problems where decisions must be made sequentially and outcomes are stochastic, as they guarantee a globally optimal policy through exact solution methods such as Value Iteration.

The challenge of selecting the optimal configuration of agents within a MAS is formulated as a MDP, defined by the tuple (S, A, T, R, γ) . In this framework, each state represents a specific situation of the environment, and each action corresponds to a particular organization of agents designated to manage it. The key novelties of this formulation are twofold: organizational structures serve as the MDP action space, replacing the conventional atomic action model; and the reward function acts as an efficiency metric that guides policy selection toward organizational structures that maximize collective performance while minimizing resource cost. The objective is to find an optimal policy $\pi^* : S \rightarrow A$ that maximizes the expected discounted cumulative reward (Equation (1)).

$$\pi^* = \arg \max_{\pi} \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \right] \quad (1)$$

In Equation (1), $\arg \max_{\pi}$ selects the policy that maximizes the objective; $\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})$ accumulates all future rewards discounted by γ^t , where a larger γ gives more weight to distant future rewards and a smaller γ produces myopic policies; and $\mathbb{E}[\cdot]$ averages over all possible stochastic trajectories induced by the transition function.

Figure 1 illustrates the overall MDP framework, where organizational structures serve as actions, operating over a state space, determining transition probabilities, and generating associated rewards.

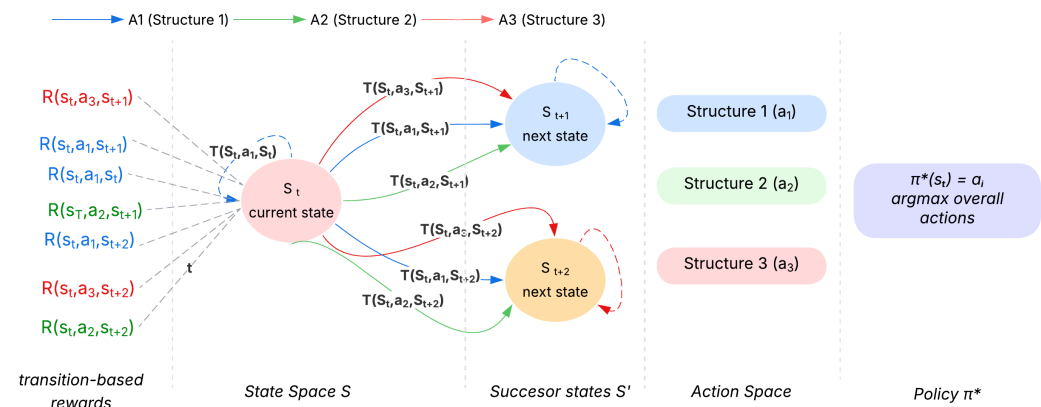


Figure 1. MAS Organizational Structures as an Action Space for MDP-Based Decision-Making. Dashed arrows indicate self-transitions (the system remains in the same state).

3.2. Organization-Based Action Space

In the proposed framework, an action $a \in A$ represents an organizational structure composed of heterogeneous agents rather than an atomic task assigned to a single agent. Heterogeneous agents refer to agents possessing different capabilities, leading to varying contributions across organizational structures. It is worth noting that this work is conducted at the macro level. Therefore, the internal architecture of individual agents is abstracted away, and the analysis focuses exclusively on the organizational structure. Formally, an action is defined as a vector (Equation (2)).

$$a = (n_1, n_2, \dots, n_k) \quad (2)$$

where $n_i \geq 0$ denotes the number of agents of type i included in the structure, and k is the number of distinct agent types. The total size of the structure is $|a|$, which represents the sum of agents belonging to all types. A coalition is used here as one concrete instantiation of this organizational structure, though the framework naturally generalizes to other organizational forms such as hierarchical, teams, or role-based assignments.

This organization-centric action space offers three favorable properties: (1) it abstracts away individual agent behavior, focusing on how many agents of each type are deployed rather than how they internally coordinate; (2) it remains compact even for large agent populations, since actions are described by agent type counts rather than individual assignments; (3) it enables the planner to reason at the macro level, making organizational decisions about resource allocation rather than managing at the micro-level individual agents.

3.3. Transition Function

Before defining the transition function, it is important to specify the structure of the transition matrices associated with each action. For each organizational structure (action) $a \in A$, a transition matrix $P^{(a)}$ is defined to represent the probabilities of transitioning from one state to another within the same state space.

In this matrix, each row corresponds to a non-goal state s_i , and each column corresponds to a possible next state s_j , including the goal state. Each entry $p_{ij}^{(a)}$ denotes the probability of transitioning from the state associated with the corresponding row to the state associated with the corresponding column under action a .

The goal state is excluded from the set of row states because it is considered an absorbing or terminal state. Once the system reaches the goal state, no further transitions to non-goal states are allowed. Consequently, transitions are only defined from non-goal states, while the goal state remains reachable as a destination state in the matrix columns. This matrix representation naturally leads to the definition of the transition function.

The transition function $T(s, a, s') = P(s' | s, a)$ defines the probability of transitioning from state s to state s' under organizational structure a . It is derived analytically through five steps based on a pre-specified capacity model. In this process, each agent is assigned competence scores that are explicitly defined and can be adjusted depending on the context. The main steps of this process are detailed in this subsection, and an overview is presented in Figure 2.

(1) Each agent type i is assigned a efficiency score $e_i(d) \in [0, 1]$ for each state dimension d , representing its individual probability of improving that dimension when deployed.

(2) For each dimension d , the probability that at least one agent in the structure a successfully improves it is computed via a complementary product formulation (Equation (3)).

$$P_{\text{succ}}(d) = 1 - \prod_{i \in a} (1 - e_i(d))^{n_i} \quad (3)$$

where the product runs over all agent types i present in a , each raised to the power n_i .

Assumption (locality): The model assumes that a single deployment can affect at most one unit per dimension per time step. This constraint reflects the sequential nature of the MDP, where each action produces only a local and incremental effect on the state. Without it, the model would systematically favor actions that lead directly to the goal regardless of their cost, bypassing the sequential decision-making process. Any transition implying a change greater than one unit on any dimension is therefore assigned zero probability (Equation (4)).

$$|\Delta d| > 1 \implies T(s, a, s') = 0 \quad (4)$$

(3) For each dimension d with current value s_d , three mutually exclusive outcomes are defined (Equations (5)–(7)).

$$P_d(s_d \rightarrow s_d - 1) = P_{\text{succ}}(d) \quad \text{if } s_d > 0 \quad (5)$$

$$P_d(s_d \rightarrow s_d) = \begin{cases} (1 - P_{\text{succ}}(d)) \times 0.8 & \text{if } s_d < d_{\max} \\ (1 - P_{\text{succ}}(d)) & \text{if } s_d = d_{\max} \end{cases} \quad (6)$$

$$P_d(s_d \rightarrow s_d + 1) = \begin{cases} (1 - P_{\text{succ}}(d)) \times 0.2 & \text{if } s_d < d_{\max} \\ 0 & \text{if } s_d = d_{\max} \end{cases} \quad (7)$$

When $s_d = d_{\max}$, the dimension has reached its maximum severity level, meaning that no further degradation is possible. The full failure probability is therefore redistributed to the stability case, ensuring that the three outcomes sum up to one for each dimension (Equation (8)).

$$P_d(s_d - 1) + P_d(s_d) + P_d(s_d + 1) = 1 \quad \forall d \quad (8)$$

More generally, the coefficients 0.8 and 0.2 encode a modeling assumption that, within a single sequential time step, state persistence dominates short-term dynamics. In particular, the system is assumed to remain unchanged with higher probability than it transitions to a degraded state, reflecting the relatively low likelihood of abrupt deterioration.

(4) Since the state dimensions are assumed to evolve conditionally independently given the organizational action, the full transition probability is obtained as the product of marginal probabilities across all dimensions (Equation (9)).

$$T(s, a, s') = \prod_d P_d(s_d \rightarrow s'_d) \quad (9)$$

(5) The resulting probabilities are normalized to ensure a valid distribution over successor states (Equation (10)).

$$T(s, a, s') \leftarrow \frac{T(s, a, s')}{\sum_{s' \in S} T(s, a, s')} \quad (10)$$

This normalization guarantees that $\sum_{s'} T(s, a, s') = 1$ for all (s, a) , ensuring a valid probability distribution over the next states.

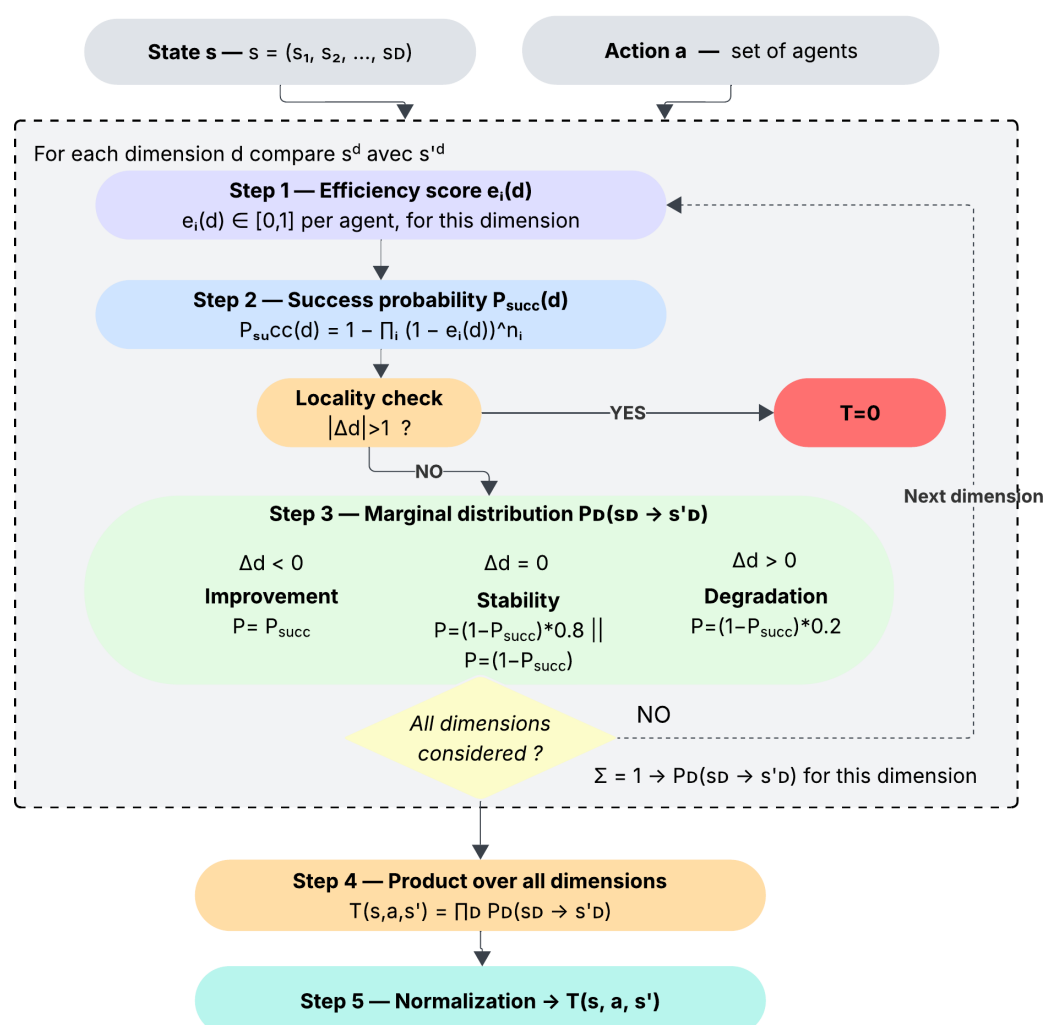


Figure 2. Computation method of the transition function $T(s, a, s')$.

3.4. Reward Function

The reward function quantifies the outcome of deploying organizational structure a in state s (Equation (11)).

$$R(s, a, s') = \frac{-\sum_d \Delta d}{|a|} \quad (11)$$

where $\Delta d = s'_d - s_d$ is the signed change in dimension d , and $|a| = \sum_i n_i$ is the total number of deployed agents. The negation ensures that improvements in the state yield

positive rewards, while normalization by $|a|$ penalizes oversized structures and encourages resource-efficient organizations.

Similarly to the transition function, the reward is precomputed for each action and encoded in a reward matrix, where each entry associates a state transition (s, s') under action a with its corresponding reward value.

Depending on the outcome of the deployed structure, three cases arise, as summarized in Table 2.

Table 2. Reward function behavior across the three possible outcome cases.

Case	R	Interpretation
Improvement	$R > 0$	$\Delta s_d < 0$: positive reward, higher when improvement is large and $ a $ is small.
No change	$R = 0$	$\Delta s_d = 0$: zero reward regardless of coalition size.
Degradation	$R < 0$	$\Delta s_d > 0$: Negative reward, which becomes more negative as $ a $ increases.

3.5. Resolution Method

The optimal policy $\pi^*(s)$ is obtained via Value Iteration [14], which computes the fixed point of the Bellman optimality equation (Equation (12)).

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} T(s, a, s') [R(s, a, s') + \gamma V^*(s')] \quad (12)$$

Starting from $V^0(s) = 0$ for all $s \in S, s \neq s_{\text{goal}}$, the algorithm iteratively updates the value function until convergence, defined as the Bellman error (Equation (13)).

$$\max_{s \in S} \underbrace{|V^{k+1}(s) - V^k(s)|}_{\text{Bellman error}} < \varepsilon \quad (13)$$

The $\max_{s \in S}$ operator is used because convergence must be guaranteed for every state simultaneously. Once this maximum difference falls below the threshold $\varepsilon > 0$, the value function is considered stable across the entire state space, and the optimal policy is extracted as (Equation (14)).

$$\pi^*(s) = \arg \max_{a \in A} \sum_{s' \in S} T(s, a, s') [R(s, a, s') + \gamma V^*(s')] \quad (14)$$

The $\arg \max$ operator selects, for each state s , the action a that maximizes the expected sum of immediate reward $R(s, a, s')$ and discounted future value $\gamma V^*(s')$, directly yielding the globally optimal policy once V^* has converged, without any additional optimization step. Value Iteration guarantees global optimality because it exhaustively evaluates all possible actions in all states at every iteration, without making any local assumption. The Bellman update considers the full action space A at each state, ensuring that no organizational structure is overlooked. Convergence to V^* is guaranteed by the iterative convergence property of the Bellman operator, and since V^* is unique, the extracted policy π^* is the globally optimal solution over the entire state-action space.

4. Case Study: RoboCup Rescue

To validate the proposed approach, we instantiate it on the RoboCup Rescue simulation environment [35], a well-established benchmark for evaluating multi-agent coordination in emergency response scenarios. RoboCup Rescue simulates large-scale urban disaster

situations in which agents must collaborate to mitigate the effects of fires, road blockages, and civilian casualties. This environment is particularly well-suited for evaluating our framework for three reasons. First, it involves multiple interacting crisis dimensions that evolve stochastically over time, which aligns naturally with the MDP state representation. Second, it provides a clear role-based agent structure, where each agent type has a distinct primary function and measurable influence on the environment. Third, it is a recognized benchmark in the MAS community, allowing for meaningful comparison with existing approaches.

The following subsections detail the environment description, the model instantiation, the implementation settings, and the obtained results.

4.1. Environment and Agent Structure

RoboCup Rescue is a large-scale multi-agent simulation environment for urban disaster response. It was introduced as a grand challenge for the artificial intelligence and multi-agent systems communities to develop autonomous agents that can coordinate under uncertainty and time pressure. In this environment, a city is affected by a major disaster, typically an earthquake, which leads to multiple simultaneous crises that agents must handle together. The simulation involves three main types of catastrophes that evolve dynamically over time:

- **Building fires** spread across buildings if not suppressed in time, threatening both infrastructure and civilian lives.
- **Road blockages** caused by collapsed buildings prevent rescue teams from reaching affected areas, slowing down the overall response.
- **Injured civilians** accumulate as injured civilians remain untreated, worsening their condition over time.

The simultaneous and interdependent nature of these crises makes RoboCup Rescue a challenging testbed for evaluating organizational decision-making under uncertainty. Three types of agents operate in the RoboCup Rescue environment, each with a specific role and a distinct influence profile on the crisis dimensions:

- **Firefighters** are specialized in fire suppression. Their primary action is to extinguish fires, giving them a high efficiency score on the fire dimension ($e_F(B) = 0.8$). They can also clear debris to gain access to fire locations, contributing secondarily to road clearance ($e_F(R) = 0.1$), and may provide basic first aid while awaiting paramedics ($e_F(I) = 0.1$).
- **Police officers** are specialized in road clearance. Their primary role is to remove blockages and manage traffic flow ($e_O(R) = 0.8$). They can also secure perimeters around fire zones, contributing marginally to fire management ($e_O(B) = 0.1$), and escort casualties to safety or clear access routes for paramedics ($e_O(I) = 0.1$).
- **Paramedics** are specialized in casualty rescue and treatment ($e_P(I) = 0.8$). Through their continuous navigation across the rescue area, they may signal road obstacles ($e_P(R) = 0.1$) and evacuate civilians near fire zones, indirectly reducing fire-related danger ($e_P(B) = 0.1$).

Agent efficiency scores are defined based on the role specifications of the RoboCup Rescue environment [36], where each agent type is associated with a primary function and possible secondary contributions. High scores are assigned to dimensions corresponding to the agent's main role, while lower scores reflect indirect but operationally plausible contributions across other dimensions.

The resulting efficiency matrix $e_i(d)$ is summarized in Table 3. These values are treated as configurable parameters and can be refined using simulation data or learned from interaction logs in future extensions.

Table 3. Agent efficiency scores $e_i(d)$ in the RoboCup Rescue environment.

Agent Type	Building Fire (B)	Roads (R)	Injured (I)
Firefighter	0.8	0.1	0.1
Police	0.1	0.8	0.1
Paramedic	0.1	0.1	0.8

4.2. Model Instantiation

To enumerate all possible states, each dimension d is encoded using n_d bits, where n_d corresponds to the number of severity levels of that dimension. The full state is represented as a binary string of total length $n_B + n_R + n_I$ bits, enabling efficient enumeration of all states by iterating over all integers from 0 to $2^{n_B+n_R+n_I} - 1$. The full state space is generated by enumerating all possible combinations of crisis severity levels across the three dimensions, as described in Algorithm 1. The goal state is defined as $s^* = (0, 0, 0)$, corresponding to the complete resolution of all crisis dimensions.

Algorithm 1 State Space Generation

```

1: Input:  $n_B, n_R, n_I$  : number of severity levels per dimension
2: Output: State space  $S$ 
3:  $S \leftarrow \emptyset$ 
4:  $N \leftarrow n_B + n_R + n_I$ 
5: for  $i = 0$  to  $2^N - 1$  do
6:   Decode  $i$  into  $(b, r, w)$ 
7:    $S \leftarrow S \cup \{(b, r, w)\}$ 
8: end for
9: return  $S$ 

```

The binary string is structured as $[b_1 \dots b_{n_B} \mid r_1 \dots r_{n_R} \mid i_1 \dots i_{n_I}]$, where the first n_B bits are reserved for fire severity B , the next n_R bits for road blockages R , and the last n_I bits for injured civilians I , with each bit indicating presence (1) or absence (0). For example, with $n_B = n_R = n_I = 2$, every integer from 0 to $2^{2+2+2} - 1 = 63$ is enumerated, where each integer decodes into a unique state triplet (b, r, w) : integer 0 = 00|00|00 = (0, 0, 0) = goal state; integer 1 = 00|00|01 = (0, 0, 1) = 1 injured civilian only; integer 63 = 11|11|11 = (3, 3, 3) = maximum severity on all dimensions.

In the proposed approach, an action represents a coalition of heterogeneous agents, as defined in organizational MAS. Three agent types are considered: firefighters (F), police officers (O), and paramedics (P). A coalition is formed by selecting a subset of agents from these types, under the constraint that it must contain at least two agents.

Similarly to the state representation, each coalition is encoded as a binary string of length $n_F + n_O + n_P$, where n_F , n_O , and n_P denote the number of agents of each type. All possible combinations are generated by iterating over the integers from 0 to $2^{n_F+n_O+n_P} - 1$, then excluding two categories:

- **The empty coalition** ($i = 0$): no agents deployed.
- **Singletons**: combinations containing only one agent, corresponding to powers of 2, efficiently detected by the binary condition $(i \& (i - 1)) = 0$.

The resulting action space size is (Equation (15)).

$$|A| = 2^{n_F+n_O+n_P} - (n_F + n_O + n_P) - 1 \quad (15)$$

The coalition generation process is summarized in Algorithm 2.

Algorithm 2 Coalition Generation

```

1: Input:  $n_F, n_O, n_P$  : number of agents per type
2: Output: Set of valid coalitions  $A$ 
3:  $A \leftarrow \emptyset$ 
4:  $N \leftarrow n_F + n_O + n_P$ 
5: for  $i = 1$  to  $2^N - 1$  do
6:   if  $(i \& (i - 1)) \neq 0$  then
7:     Decode  $i$  into  $(k_F, k_O, k_P)$ 
8:      $A \leftarrow A \cup \{(k_F, k_O, k_P)\}$ 
9:   end if
10: end for
11: return  $A$ 

```

The binary string is structured as $[f_1 \dots f_{n_F} \mid o_1 \dots o_{n_O} \mid p_1 \dots p_{n_P}]$, where the first n_F bits are reserved for firefighters, the next n_O for police officers, and the last n_P for paramedics, each bit indicating presence (1) or absence (0). For example, with $n_F = n_O = n_P = 2$, every integer from 1 to $2^{2+2+2} - 1 = 63$ is enumerated. Singletons correspond to powers of 2 and are detected by $(i \& (i - 1)) = 0$: subtracting 1 from a power of 2 flips all its bits, so the AND always yields 0 (e.g., $100 \& 011 = 000$), while any valid coalition gives a non-zero result (e.g., $011 \& 010 \neq 0$).

The binary encoding is chosen for three reasons: it is semantically natural, since each agent type is either present or absent, mapping directly onto a binary representation; it is memory-efficient, as each coalition is stored as a single compact value rather than a list or dictionary object; and it offers computational performance, since binary operations are executed at the CPU level, making them significantly faster than equivalent high-level data structure operations.

To assess the scalability and generality of the framework, four experimental configurations are defined in Table 4, varying the number of severity levels per dimension and the number of agents per type.

Table 4. Experimental configurations.

Config	B	R	I	O	F	P	States	Coalitions	Justification
S1	3	2	1	2	2	1	64	26	Minimal asymmetric baseline configuration
S2	4	2	2	3	2	2	256	120	Asymmetric state space with a dominant dimension
S3	3	3	3	3	3	3	512	502	Balanced reference configuration
S4	3	3	2	4	3	3	256	1013	Large action space, coalition scalability

Due to the exponential growth of the state and action spaces with respect to the problem parameters, a full display of the transition matrices is impractical for the configurations used in the experiments (e.g., S1 with $|S| = 64$ states and $|A| = 26$ coalitions yields matrices of size 64×64 per coalition). To illustrate the instantiated transition model concretely, we consider a minimal configuration with $N_B = N_R = N_I = 1$ and $N_F = N_O = N_P = 1$, yielding $|S| = 8$ states and $|A| = 2^3 - 1 - 3 = 4$ valid coalitions. The complete set of transition matrices for this configuration is presented in Table 5, where each matrix T^{a_k} is of size 7×8 (the goal state $s^* = (0, 0, 0)$ is excluded from rows as it is absorbing). The bold entries indicate the most likely transition from each state, consistently directing toward the goal state or its most probable nearest neighbors.

Table 5. Transition matrices for the minimal configuration ($N_B = N_R = N_I = 1$, $N_F = N_O = N_P = 1$): 4 coalitions, 7 states (goal state $s^* = s'_0 = (0, 0, 0)$).

Coalition $a_0 = (0_O, 1_F, 1_P)$									Coalition $a_1 = (1_O, 0_F, 1_P)$								
$s \backslash s'$	s'_0	s'_1	s'_2	s'_3	s'_4	s'_5	s'_6	s'_7	$s \backslash s'$	s'_0	s'_1	s'_2	s'_3	s'_4	s'_5	s'_6	s'_7
s_1	0.5444	0.0956	0.1361	0.0239	0.1361	0.0239	0.0340	0.0060	s_1	0.5344	0.1056	0.1336	0.0264	0.1336	0.0264	0.0334	0.0066
s_2	0.5344	0.1336	0.1056	0.0264	0.1336	0.0334	0.0264	0.0066	s_2	0.0849	0.0212	0.5551	0.1388	0.0212	0.0053	0.1388	0.0347
s_3	0.5683	0.0998	0.1122	0.0197	0.1421	0.0249	0.0281	0.0049	s_3	0.0886	0.0175	0.5794	0.1144	0.0222	0.0044	0.1449	0.0286
s_4	0.0787	0.0197	0.0197	0.0049	0.5613	0.1403	0.1403	0.0351	s_4	0.5334	0.1334	0.1334	0.0333	0.1066	0.0266	0.0266	0.0067
s_5	0.0837	0.0147	0.0209	0.0037	0.5968	0.1048	0.1492	0.0262	s_5	0.5568	0.1100	0.1392	0.0275	0.1112	0.0220	0.0278	0.0055
s_6	0.0822	0.0205	0.0162	0.0041	0.5859	0.1465	0.1157	0.0289	s_6	0.0884	0.0221	0.5784	0.1446	0.0177	0.0044	0.1155	0.0289
s_7	0.0874	0.0153	0.0173	0.0030	0.6229	0.1094	0.1230	0.0216	s_7	0.0923	0.0182	0.6037	0.1192	0.0184	0.0036	0.1206	0.0238
Coalition $a_2 = (1_O, 1_F, 0_P)$									Coalition $a_3 = (1_O, 1_F, 1_P)$								
$s \backslash s'$	s'_0	s'_1	s'_2	s'_3	s'_4	s'_5	s'_6	s'_7	$s \backslash s'$	s'_0	s'_1	s'_2	s'_3	s'_4	s'_5	s'_6	s'_7
s_1	0.0849	0.5551	0.0212	0.1388	0.0212	0.1388	0.0053	0.0347	s_1	0.5454	0.0946	0.1363	0.0237	0.1363	0.0237	0.0341	0.0059
s_2	0.5444	0.1361	0.0956	0.0239	0.1361	0.0340	0.0239	0.0060	s_2	0.5454	0.1363	0.0946	0.0237	0.1363	0.0341	0.0237	0.0059
s_3	0.0903	0.5902	0.0159	0.1037	0.0226	0.1476	0.0040	0.0259	s_3	0.5810	0.1008	0.1008	0.0175	0.1452	0.0252	0.0252	0.0044
s_4	0.5444	0.1361	0.1361	0.0340	0.0956	0.0239	0.0239	0.0060	s_4	0.5445	0.1361	0.1361	0.0340	0.0955	0.0239	0.0239	0.0060
s_5	0.0903	0.5902	0.0226	0.1476	0.0159	0.1037	0.0040	0.0259	s_5	0.5800	0.1006	0.1450	0.0252	0.1017	0.0176	0.0254	0.0044
s_6	0.5788	0.1447	0.1017	0.0254	0.1017	0.0254	0.0179	0.0045	s_6	0.5800	0.1450	0.1006	0.0252	0.1017	0.0254	0.0176	0.0044
s_7	0.0960	0.6276	0.0169	0.1102	0.0169	0.1102	0.0030	0.0194	s_7	0.6178	0.1072	0.1072	0.0186	0.1084	0.0188	0.0188	0.0033

Similarly to the transition model, displaying all reward matrices becomes impractical for the configurations used in the experiments. We therefore adopt the same minimal configuration ($N_B = N_R = N_I = 1$, $N_F = N_O = N_P = 1$) to illustrate the instantiated reward model.

The corresponding reward matrices are presented in Table 6. Each matrix R^{a_k} has size 7×8 , where the goal state $s^* = (0, 0, 0)$ is excluded from the rows as it is absorbing, while it is retained as a column to reflect the maximum reward associated with reaching it.

It is also worth noting that coalitions a_0 , a_1 , and a_2 produce identical reward matrices due to having the same coalition size $|a| = 2$, whereas coalition $a_3 = (1_O, 1_F, 1_P)$ with $|a| = 3$ yields proportionally lower reward values as a result of the normalization by coalition size, which penalizes larger coalitions in favor of resource-efficient configurations.

Table 6. Reward matrices for the minimal configuration ($N_B = N_R = N_I = 1$, $N_F = N_O = N_P = 1$): 4 coalitions, 7 states (goal state $s^* = s'_0 = (0, 0, 0)$).

Coalition $a_0 = (0_O, 1_F, 1_P)$									Coalition $a_1 = (1_O, 0_F, 1_P)$								
$s \backslash s'$	s'_0	s'_1	s'_2	s'_3	s'_4	s'_5	s'_6	s'_7	$s \backslash s'$	s'_0	s'_1	s'_2	s'_3	s'_4	s'_5	s'_6	s'_7
s_1	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000	s_1	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000
s_2	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000	s_2	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000
s_3	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500	s_3	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500
s_4	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000	s_4	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000
s_5	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500	s_5	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500
s_6	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500	s_6	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500
s_7	1.500	1.000	1.000	0.500	1.000	0.500	0.500	0.000	s_7	1.500	1.000	1.000	0.500	1.000	0.500	0.500	0.000
Coalition $a_2 = (1_O, 1_F, 0_P)$									Coalition $a_3 = (1_O, 1_F, 1_P)$								
$s \backslash s'$	s'_0	s'_1	s'_2	s'_3	s'_4	s'_5	s'_6	s'_7	$s \backslash s'$	s'_0	s'_1	s'_2	s'_3	s'_4	s'_5	s'_6	s'_7
s_1	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000	s_1	0.333	0.000	0.000	−0.333	0.000	−0.333	−0.333	−0.667
s_2	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000	s_2	0.333	0.000	0.000	−0.333	0.000	−0.333	−0.333	−0.667
s_3	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500	s_3	0.667	0.333	0.333	0.000	0.333	0.000	0.000	−0.333
s_4	0.500	0.000	0.000	−0.500	0.000	−0.500	−0.500	−1.000	s_4	0.333	0.000	0.000	−0.333	0.000	−0.333	−0.333	−0.667
s_5	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500	s_5	0.667	0.333	0.333	0.000	0.333	0.000	0.000	−0.333
s_6	1.000	0.500	0.500	0.000	0.500	0.000	0.000	−0.500	s_6	0.667	0.333	0.333	0.000	0.333	0.000	0.000	−0.333
s_7	1.500	1.000	1.000	0.500	1.000	0.500	0.500	0.000	s_7	1.000	0.667	0.667	0.333	0.667	0.333	0.333	0.000

4.3. Implementation and Parameter Settings

As described in Section 3, value iteration is applied to solve the MDP by iteratively updating the utility of each state until convergence, using the Bellman optimality equation. The algorithm was implemented in Java and applied across the four experimental configurations S1, S2, S3, and S4 defined in Table 4. The following subsections detail the concrete parameter settings adopted in these experiments.

4.3.1. Algorithm Parameters

The value iteration algorithm was configured with the parameters listed in Table 7, selected empirically based on preliminary experiments.

Table 7. Value iteration parameter settings.

Parameter	Value	Justification
Discount factor γ	0.75	Best trade-off between convergence speed and policy quality
Convergence threshold ε	0.001	Sufficient precision without excessive iterations
Initial value $U(s_0)$	10	Positive reward assigned to goal state
Initial value $U(s \neq s_0)$	0	Neutral initialization for all non-goal states

4.3.2. Discount Factor Selection

The choice of $\gamma = 0.75$ was evaluated experimentally by running value iteration on configuration S3 for $\gamma \in \{0.25, 0.50, 0.75, 0.90\}$. Configuration S3 ($n_B = n_R = n_I = 3$, 512 states, 502 coalitions) is selected for this analysis because it is the balanced reference configuration. In the Bellman update (Equation (12)), an asymmetric configuration would cause uneven reward propagation across dimensions, making it impossible to isolate the pure effect of γ . As shown in Figure 3, smaller values such as $\gamma = 0.25$ lead to faster convergence in terms of Bellman error and require fewer iterations, but result in short-sighted policies that favor immediate rewards over potentially more beneficial long-term outcomes. In contrast, larger values increase the number of iterations but better capture future consequences of actions within the decision process.

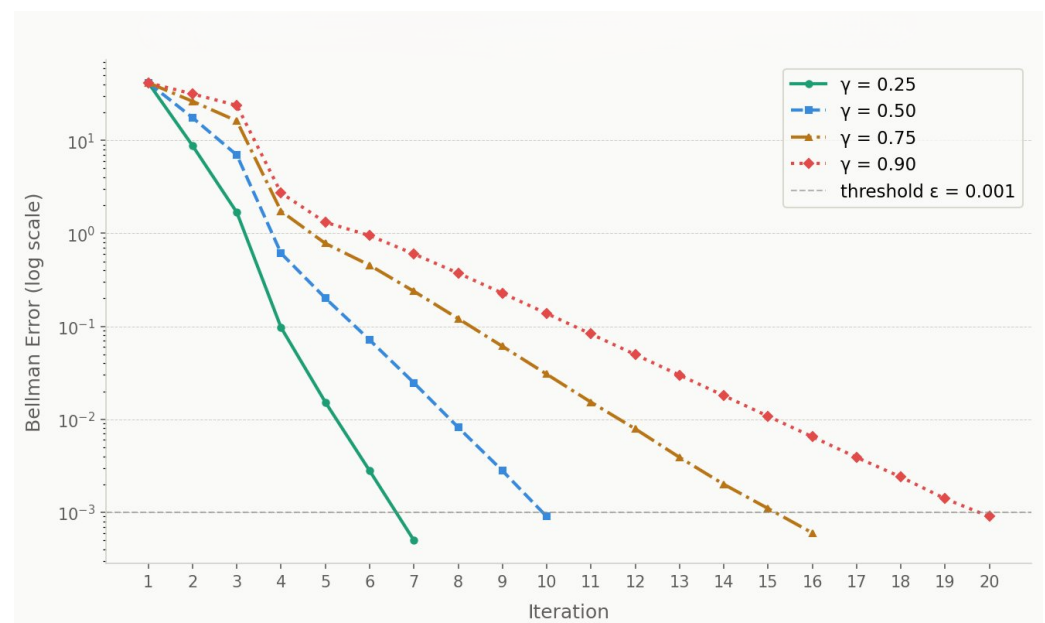


Figure 3. Bellman error under different discount factors on configuration S3.

Although $\gamma = 0.25$ achieves faster convergence, $\gamma = 0.75$ is selected as it achieves the best trade-off between convergence speed and solution quality. This is theoretically grounded: γ directly controls the weight assigned to future rewards in Equation (1), meaning that a higher γ forces the algorithm to propagate long-term consequences across more iterations before stabilizing, leading to policies that maximize cumulative returns rather than immediate rewards. It is therefore used in all subsequent experiments.

4.4. Results and Analysis

This section presents the experimental results obtained by applying the value iteration algorithm to the four configurations defined in Table 4. Each configuration is evaluated in terms of convergence behavior, policy stabilization, coalition efficiency, and final policy distribution.

4.4.1. Convergence Analysis

Figure 4 illustrates the evolution of the Bellman error across iterations for all four configurations. All configurations converge within 16 iterations, confirming the effectiveness of the value iteration algorithm under the proposed MDP formulation.

Configuration **S1**, with the smallest state space (64 states), converges the fastest, reaching the threshold $\varepsilon = 0.001$ around iteration 13. Configurations **S2** and **S3** exhibit the slowest convergence, reaching the threshold at approximately iterations 15–16.

Interestingly, **S4**, despite having the same state space size as **S2** (256 states), converges faster, suggesting that convergence speed is not solely determined by the size of the state space but also depend on the structure of the action space or the distribution of coalitions.

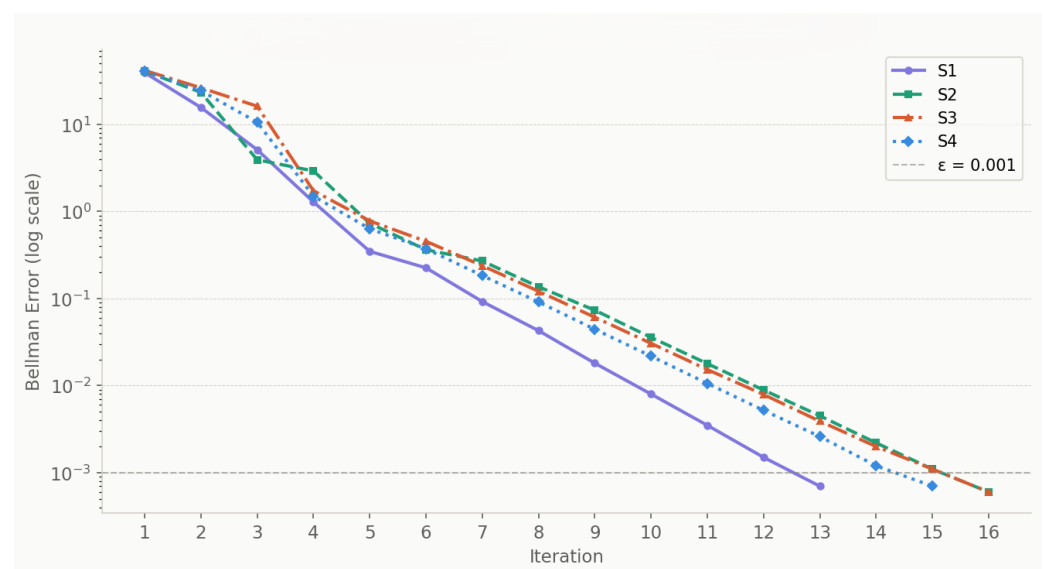


Figure 4. Bellman error convergence across all configurations.

Figure 5 shows the evolution of the L1 norm of the value function $\sum_s U(s)$ across iterations. All configurations exhibit a similar pattern, with a rapid increase during the first two to three iterations as value information propagates from the goal state, followed by gradual stabilization.

Configuration **S3** reaches a significantly higher L1 norm ($\sim 15,000$ – $16,000$) compared to **S2** and **S4** (~ 7000), reflecting its larger state space (512 states) and higher number of coalitions (502). In contrast, **S1**, with a smaller state space (64 states), stabilizes at a lower value (~ 1400) and shows only a limited initial increase, consistent with reduced value propagation.

Overall, convergence speed is comparable across all configurations, with all curves reaching a stable plateau before iteration 10.

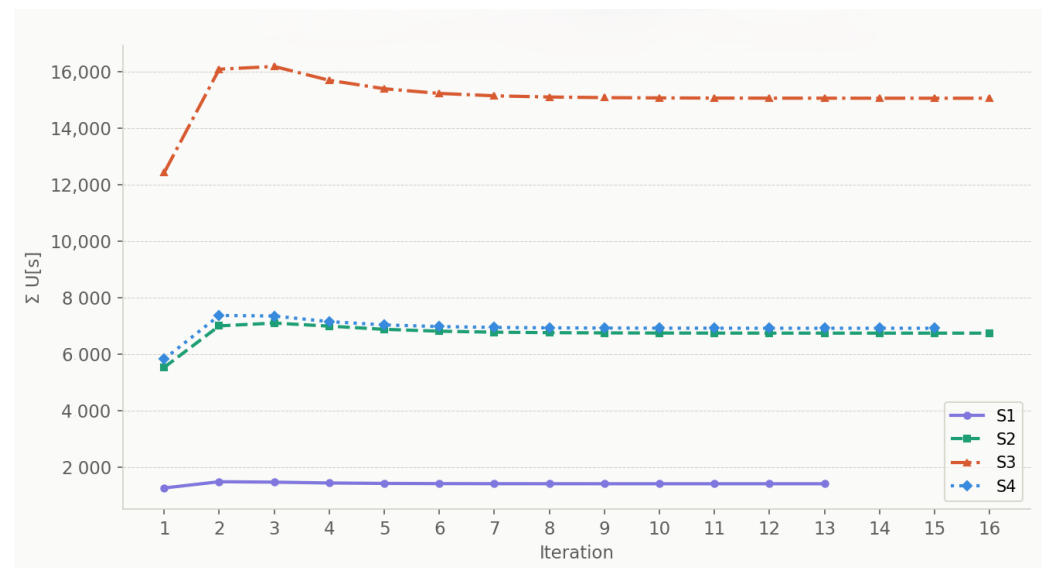


Figure 5. L1 norm of the value function across all configurations.

4.4.2. Policy Stabilization

Figure 6 illustrates the number of states whose optimal coalition changes between two consecutive iterations (i.e., $\Delta k = \pi_k(s) \neq \pi_{k-1}(s)$), serving as a measure of policy stabilization. In all configurations, the policy stabilizes within the first three iterations, with minor transient adjustments observed in later iterations for S2 (iteration 6, 16 states) and S4 (iteration 4, 18 states). Configuration S3 exhibits the largest initial policy change (~ 511 states at iteration 1), consistent with its full 512-state space. Configurations S2 and S4 show approximately 255 changes at the first iteration, consistent with their 256-state spaces, while S1 shows a considerably lower number of initial changes (~ 63 states), reflecting its smaller 64-state space.

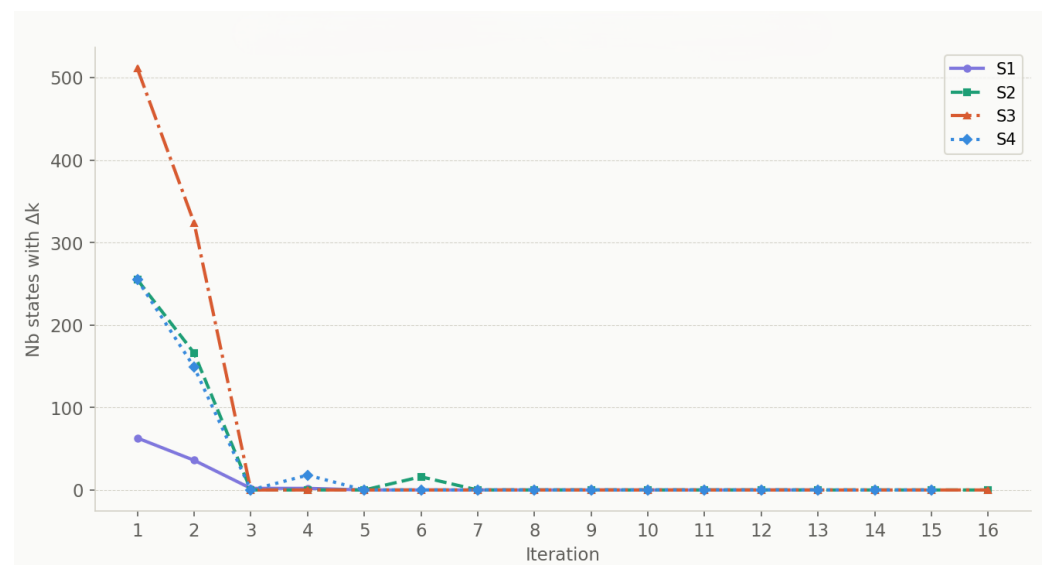


Figure 6. Policy stabilization across all configurations.

4.4.3. Coalition Efficiency

Figure 7 illustrates the evolution of the average coalition size selected by the optimal policy across iterations. All configurations converge rapidly: S1, S3, and S4 stabilize within two iterations, while S2 requires six iterations due to a two-step convergence pattern.

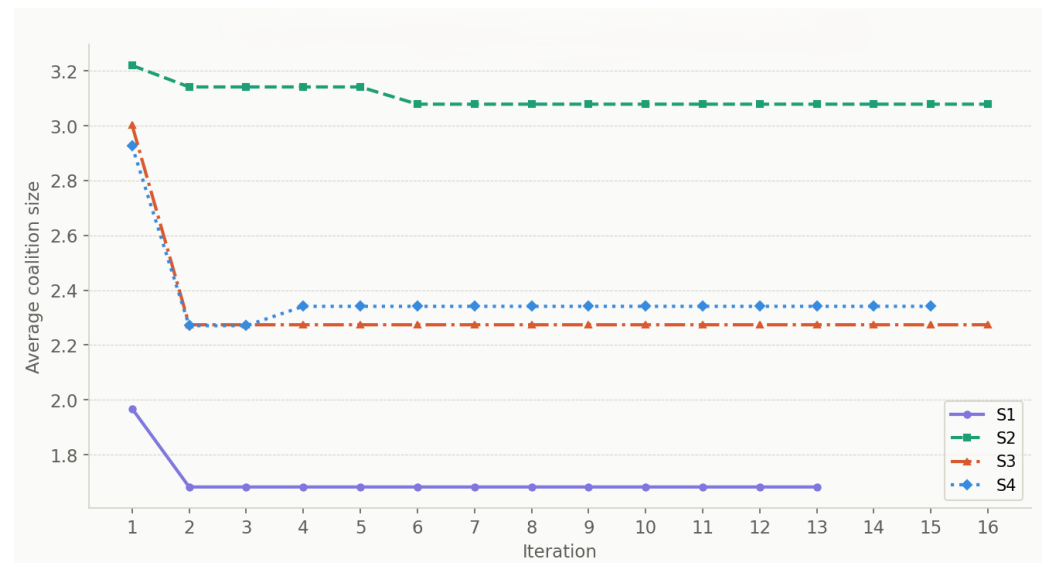


Figure 7. Average coalition size across all configurations.

The final average coalition sizes are approximately 1.71, 3.08, 2.28, and 2.34 agents for S1, S2, S3, and S4, respectively. S1, with the smallest state and action spaces (64 states, 26 coalitions), yields the lowest average coalition size, consistent with its highly constrained structure.

A comparison between S2 and S4, which share the same state space (256 states), shows that S4 achieves significantly smaller coalitions (2.34 vs. 3.08 agents) due to its richer action space (1013 vs. 120 coalitions), enabling more flexible organizational choices. Similarly, S3 and S4 produce comparable average coalition sizes (2.28 and 2.34 agents) despite different state spaces (512 vs. 256 states), further suggesting that the action space has a stronger influence on coalition compactness than the state space.

4.4.4. Final Policy Distribution

Figure 8 presents the distribution of coalitions selected in the final optimal policy for each configuration. The y-axis shows the coalition identifier k along with its size, and the percentage on each bar indicates the proportion of states assigned to that coalition. In configurations S3 and S4, a single coalition of size 3 ($k = 65$) dominates the policy with 61.8% and 50.6% of states, respectively. In S2, coalition $k = 15$ (size 3) is assigned to 46.7% of states. In S1, coalition $k = 5$ (size 2) is the most frequently selected (27.0%), followed by $k = 6$ (size 3) at 20.6%, reflecting a more distributed policy over the smaller state space. Across all configurations, the largest coalitions (size 6) are systematically assigned to the fewest states, validating the reward function design that penalizes unnecessary resource deployment and favors minimal yet effective coalitions.

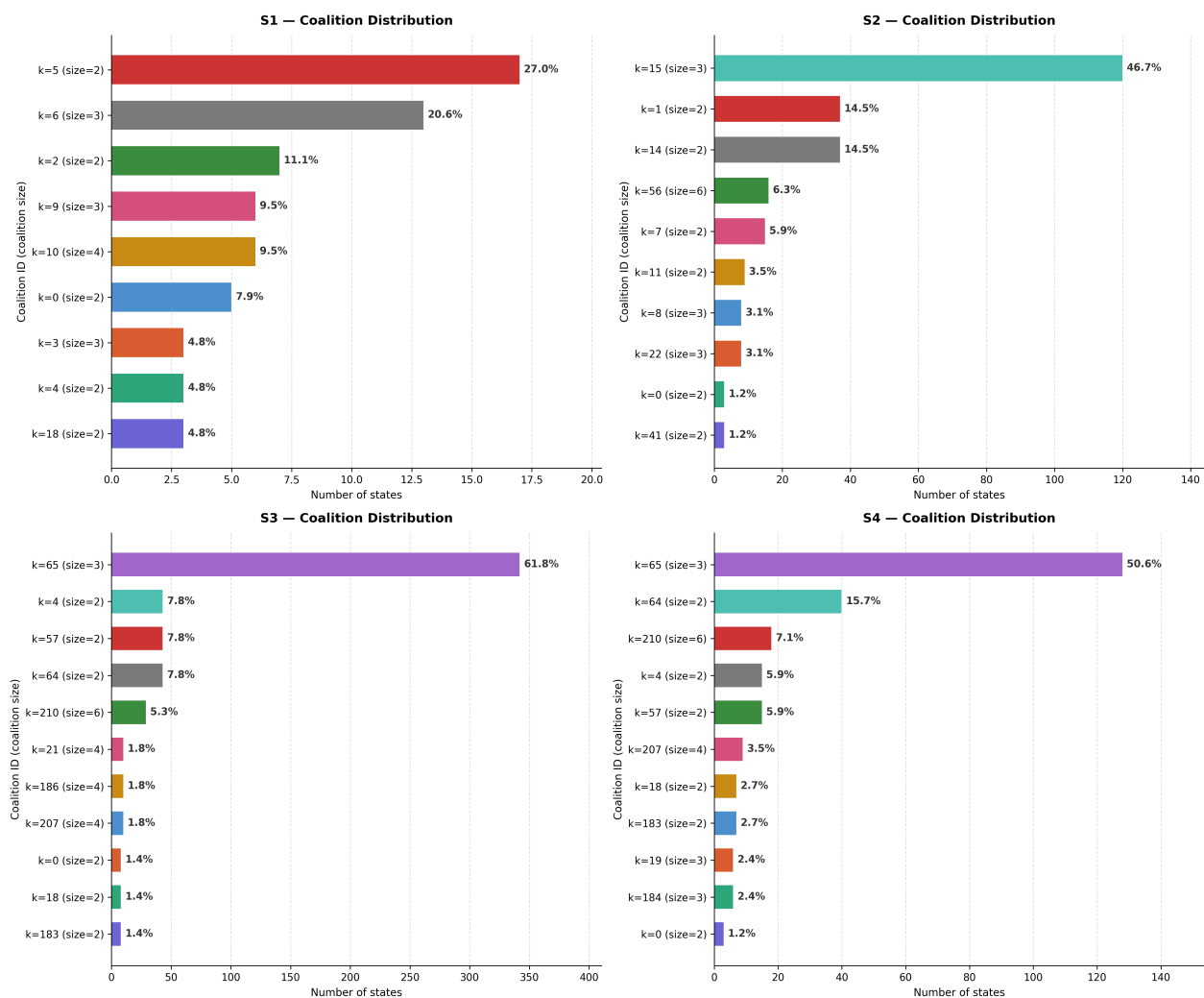


Figure 8. Final policy coalition distribution across all configurations.

4.4.5. Policy Illustration

Table 8 presents representative policy samples for configurations S1 through S4. Only eight state examples are included, as the full policy is too long to display.

The policy exhibits a structured mapping between system states and coalition activations. The goal state consistently requires no intervention, reflecting a stable absorbing configuration.

For non-goal states, coalition selection is conditional and depends on specific combinations of active dimensions rather than following a simple additive rule. Overall, the learned policy achieves a compressed allocation of resources, balancing efficiency and coverage by activating only the necessary subsets of agents instead of deploying all available resources whenever multiple dimensions are present.

Furthermore, the application of a coalition to a given state induces a transition to another state, which is itself represented within the policy alongside its associated optimal action.

This process continues recursively, forming a structured sequence of state-action pairs that progressively guides the system toward the goal state while ensuring optimal action efficiency.

Table 8. Representative policy samples (8 states each) for configurations S1–S4.

R	B	I	Optimal Coalition
S1 ($N_B = 3, N_R = 2, N_I = 1, N_O = 2, N_F = 2, N_P = 1$)			
0R,0R	0B,0B,0B	0I	Goal state—no coalition required
0R,0R	0B,0B,0B	1I	(0O,0O, 0F,1F, 1P)
0R,0R	0B,0B,1B	0I	(0O,0O, 1F,1F, 0P)
0R,0R	0B,0B,1B	1I	(0O,0O, 1F,1F, 1P)
0R,1R	0B,0B,0B	0I	(1O,1O, 0F,0F, 0P)
0R,1R	0B,0B,0B	1I	(0O,1O, 0F,0F, 1P)
1R,1R	0B,1B,1B	0I	(0O,1O, 0F,1F, 0P)
1R,1R	1B,1B,1B	1I	(0O,1O, 0F,1F, 1P)
S2 ($N_B = 4, N_R = 2, N_I = 2, N_O = 3, N_F = 2, N_P = 2$)			
0R,0R	0B,0B,0B,0B	0I,0I	Goal state—no coalition required
0R,0R	0B,0B,0B,0B	0I,1I	(0O,0O,0O, 0F,0F, 1P,1P)
0R,0R	0B,0B,0B,1B	0I,0I	(0O,0O,0O, 1F,1F, 0P,0P)
0R,0R	0B,0B,1B,1B	1I,1I	(0O,0O,0O, 0F,1F, 0P,1P)
0R,1R	0B,0B,0B,0B	0I,1I	(0O,0O,1O, 0F,0F, 0P,1P)
0R,1R	0B,1B,0B,0B	0I,0I	(0O,0O,1O, 1F,1F, 0P,0P)
1R,1R	0B,0B,1B,1B	0I,1I	(0O,0O,1O, 0F,1F, 0P,1P)
1R,1R	1B,1B,1B,1B	1I,1I	(0O,0O,1O, 0F,1F, 0P,1P)
S3 ($N_B = 3, N_R = 3, N_I = 3, N_O = 3, N_F = 3, N_P = 3$)			
0R,0R,0R	0B,0B,0B	0I,0I,0I	Goal state—no coalition required
0R,0R,0R	0B,0B,0B	0I,0I,1I	(0O,0O,0O, 0F,0F,0F, 0P,1P,1P)
0R,0R,0R	0B,0B,1B	0I,0I,0I	(0O,0O,0O, 0F,1F,1F, 0P,0P,0P)
0R,0R,0R	0B,0B,1B	0I,0I,1I	(0O,0O,0O, 0F,1F,1F, 0P,1P,1P)
0R,1R,1R	0B,0B,0B	0I,0I,0I	(0O,1O,1O, 0F,0F,0F, 0P,0P,0P)
0R,1R,1R	0B,1B,1B	0I,1I,1I	(0O,0O,1O, 0F,0F,1F, 0P,0P,1P)
1R,1R,1R	0B,0B,1B	0I,1I,1I	(0O,0O,1O, 0F,0F,1F, 0P,0P,1P)
1R,1R,1R	1B,1B,1B	1I,1I,1I	(0O,0O,1O, 0F,0F,1F, 0P,0P,1P)
S4 ($N_B = 3, N_R = 3, N_I = 2, N_O = 4, N_F = 3, N_P = 3$)			
0R,0R,0R	0B,0B,0B	0I,0I	Goal state—no coalition required
0R,0R,0R	0B,0B,0B	0I,1I	(0O,0O,0O,0O, 0F,0F,0F, 0P,1P,1P)
0R,0R,0R	0B,0B,1B	0I,0I	(0O,0O,0O,0O, 0F,1F,1F, 0P,0P,0P)
0R,0R,0R	0B,0B,1B	0I,1I	(0O,0O,0O,0O, 0F,1F,1F, 0P,0P,1P)
0R,0R,0R	0B,1B,1B	0I,0I	(0O,0O,0O,0O, 0F,1F,1F, 0P,0P,0P)
0R,1R,1R	0B,0B,0B	0I,1I	(0O,0O,0O,1O, 0F,0F,0F, 0P,0P,1P)
1R,1R,1R	0B,1B,1B	0I,1I	(0O,0O,0O,1O, 0F,0F,1F, 0P,0P,1P)
1R,1R,1R	1B,1B,1B	1I,1I	(0O,0O,0O,1O, 0F,0F,1F, 0P,1P,1P)

5. Discussion

Across the three categories of organizational adaptation surveyed in Section 2, two common limitations can be identified: adaptation mechanisms are mostly reactive or based on past cases, with little attention to predicting the effects of changes on future system states, and the long-term systemic consequences of organizational modifications remain largely unexplored. Norm adaptation approaches adjust agent behavior through norm synthesis or

revision, but their evaluation of impact remains local or reactive. Role-based approaches enable fine-grained behavioral adaptation but focus on individual role assignment rather than global organizational structure, systematically lacking long-term prediction mechanisms. Among structural reconfiguration works, some incorporate probabilistic prediction, yet this remains limited to predefined configurations without autonomous global optimization.

The proposed MDP-based framework addresses these difficulties simultaneously by modeling reorganization as a sequential decision problem that explicitly estimates the probable impact of each organizational structure on future system states via the transition function, selects the optimal structure via Value Iteration, and uses the reward function as an efficiency metric that guides policy selection toward organizational structures that maximize collective performance while minimizing resource cost. As summarized in Table 9, it is the only approach that combines system-level impact evaluation, predictive reasoning, and globally optimal decision-making within a unified model.

Table 9. Comparative summary of organizational adaptation approaches.

Approach	Impact Scope	Predictive	Globally Optimal
Norm adaptation works	Local	✗	✗
Role evolution works	Team-level	Partial	✗
Structural reconfiguration	System-level	Partial	✗
Proposed framework	System-level	✓	✓

✓: supported; ✗: not supported.

Nevertheless, the framework carries notable limitations. The efficiency scores are pre-specified and may not capture real-world variability. The locality assumption restricts state changes to one unit per dimension per time step, which may oversimplify abrupt crisis dynamics. The transition model is fixed and must be redefined when the problem setting changes, limiting flexibility across different scenarios. Finally, scalability remains a concern as the state and action spaces expand exponentially with the number of agent types or severity levels.

Future work will focus on learning efficiency scores from simulation data, adopting approximate reinforcement learning methods to handle larger state spaces, and validating the framework on real incident data.

6. Conclusions

Organizational adaptation in MAS remains a critical challenge, particularly when it comes to selecting the most effective organizational configuration under uncertainty and dynamic conditions. Although significant effort has been devoted to norm adaptation, role evolution, and structural reconfiguration, most existing approaches react to changes without anticipating their systemic consequences or guaranteeing global optimality. In this work, we presented a decision-making framework that models organizational reorganization as a Markov Decision Process, where coalition structures constitute the action space and the optimal policy is derived via value iteration. The framework was validated on the RoboCup Rescue simulation platform across four experimental configurations, demonstrating rapid convergence, stable policy derivation, and consistent selection of resource-efficient coalitions. According to the obtained results, the proposed approach effectively bridges the gap between reactive organizational adaptation and proactive, globally aware structural planning. In prospect, we plan to extend the framework toward non-stationary environments, incorporate learned transition parameters from simulation data, and explore approximate reinforcement learning methods to scale the approach to larger and more complex multi-agent deployments.

Author Contributions: Conceptualization, N.S., T.M. and M.B.; Methodology, N.S., V.G., T.M., M.B. and C.G.; Software, N.S. and T.M.; Validation, N.S. and T.M.; Formal analysis, N.S. and T.M.; Investigation, N.S., V.G. and T.M.; Resources, N.S. and T.M.; Data curation, N.S. and T.M.; Writing—original draft, N.S. and T.M.; Writing—review and editing, N.S., V.G., M.B. and C.G.; Visualization, N.S., T.M., V.G., M.B. and C.G.; Supervision, T.M., V.G. and M.B.; Project administration, N.S., T.M., V.G., M.B. and C.G.; Funding acquisition, V.G. and C.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All data generated or analyzed during this study are included in this published article.

Acknowledgments: The authors gratefully acknowledge the support of the Research Laboratory on Complex Systems and Computer Science (ReLa(CS)²), and extend their sincere thanks to the Directorate General for Scientific Research and Technological Development (DGRSDT, MESRS).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wooldridge, M. *An Introduction to Multiagent Systems*; John Wiley & Sons: Hoboken, NJ, USA, 2009.
2. Ferber, J.; Gutknecht, O.; Michel, F. From agents to organizations: An organizational view of multi-agent systems. In *Agent-Oriented Software Engineering IV*; International Workshop on Agent-Oriented Software Engineering; Springer: Berlin/Heidelberg, Germany, 2003; pp. 214–230.
3. Jensen, A.S.; Villadsen, J. A comparison of organization-centered and agent-centered multi-agent systems. *Artif. Intell. Res.* **2013**, *2*, 59–69.
4. Zambonelli, F.; Jennings, N.R.; Wooldridge, M. Developing multiagent systems: The Gaia methodology. *ACM Trans. Softw. Eng. Methodol. (TOSEM)* **2003**, *12*, 317–370.
5. Tamrabet, Z.; Nessah, D.; Marir, T.; Gupta, V.; Mokhati, F. The Impact of the Organization on the Autonomy of Agents. *Information* **2025**, *16*, 838. [\[CrossRef\]](#)
6. Picard, G.; Hübner, J.F.; Boissier, O.; Gleizes, M.P. Reorganisation and self-organisation in multi-agent systems. In *Proceedings of the 1st International Workshop on Organizational Modeling, ORGMOD, 2009*; Universität Hamburg: Hamburg, Germany, 2009; pp. 66–80.
7. Abbas, H.A.; Shaheen, S.I.; Amin, M.H. Organization of multi-agent systems: An overview. *arXiv* **2015**, arXiv:1506.09032.
8. Sims, M.; Corkill, D.; Lesser, V. Automated organization design for multi-agent systems. *Auton. Agents Multi-Agent Syst.* **2008**, *16*, 151–185.
9. Weyns, D.; Georgeff, M. Self-adaptation using multiagent systems. *IEEE Softw.* **2009**, *27*, 86–91. [\[CrossRef\]](#)
10. Dorri, A.; Kanhere, S.S.; Jurdak, R. Multi-agent systems: A survey. *IEEE Access* **2018**, *6*, 28573–28593. [\[CrossRef\]](#)
11. Horling, B.; Lesser, V. A survey of multi-agent organizational paradigms. *Knowl. Eng. Rev.* **2004**, *19*, 281–316. [\[CrossRef\]](#)
12. Sahel, N.; Marir, T.; Gupta, V.; Bouzid, M. Dynamic Aspects of Multiagent Systems Organizations: A Survey. In *Proceedings of the International Conference on Computer Science's Complex Systems and their Applications*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 207–221.
13. Ye, D.; Zhang, M.; Vasilakos, A.V. A survey of self-organization mechanisms in multiagent systems. *IEEE Trans. Syst. Man Cybern. Syst.* **2016**, *47*, 441–461. [\[CrossRef\]](#)
14. Puterman, M.L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*; John Wiley & Sons: Hoboken, NJ, USA, 2014.
15. Han, S.; Zhang, Q.; Yao, Y.; Jin, W.; Xu, Z. LLM multi-agent systems: Challenges and open problems. *arXiv* **2024**, arXiv:2402.03578.
16. Cordova, C.; Taverner, J.; Del Val, E.; Argente, E. A systematic review of norm emergence in multi-agent systems. *arXiv* **2024**, arXiv:2412.10609.
17. Ferber, J.; Gutknecht, O. A meta-model for the analysis and design of organizations in multi-agent systems. In *Proceedings International Conference on Multi Agent Systems (Cat. No. 98EX160)*; IEEE: New York, NY, USA, 1998; pp. 128–135.
18. Chebout, M.S.; Marir, T.; Gupta, V. Noragr: A normative agent groupe role model for organizational centered open multi-agent systems. In *2023 Third International Conference on Theoretical and Applicative Aspects of Computer Science (ICTAACS)*; IEEE: New York, NY, USA, 2023; pp. 1–5.

19. Hannoun, M.; Boissier, O.; Sichman, J.S.; Sayettat, C. MOISE: An organizational model for multi-agent systems. In *Proceedings of the Ibero-American Conference on Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 156–165.
20. Hübner, J.F.; Sichman, J.S.; Boissier, O. A model for the structural, functional, and deontic specification of organizations in multiagent systems. In *Proceedings of the Brazilian Symposium on Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 118–128.
21. Dell’Anna, D.; Alechina, N.; Dalpiaz, F.; Dastani, M.; Logan, B. Data-driven revision of conditional norms in multi-agent systems. *J. Artif. Intell. Res.* **2022**, *75*, 1549–1593.
22. Alechina, N.; De Giacomo, G.; Logan, B.; Perelli, G. Automatic synthesis of dynamic norms for multi-agent systems. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*; IJCAI Organization: California City, CA, USA, 2022; Volume 19, pp. 12–21.
23. Yan, E.; Nardin, L.G.; Hübner, J.F.; Boissier, O. An agent-centric perspective on norm enforcement and sanctions. In *Proceedings of the International Workshop on Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 79–99.
24. Riad, M.; de Carvalho, V.R.; Golpayegani, F. Multi-Value Alignment in Normative Multi-Agent System: Evolutionary Optimisation Approach. *arXiv* **2023**, arXiv:2305.07366.
25. Xia, Y.; Zhu, J.; Zhu, L. Dynamic role discovery and assignment in multi-agent task decomposition. *Complex Intell. Syst.* **2023**, *9*, 6211–6222. [[CrossRef](#)]
26. Hou, Y.; Sun, M.; Zeng, Y.; Ong, Y.S.; Jin, Y.; Ge, H.; Zhang, Q. A multiagent cooperative learning system with evolution of social roles. *IEEE Trans. Evol. Comput.* **2023**, *28*, 531–543. [[CrossRef](#)]
27. Lu, S.; Shao, J.; Luo, B.; Lin, T. Morphagent: Empowering agents through self-evolving profiles and decentralized collaboration. *arXiv* **2024**, arXiv:2410.15048.
28. Goel, H.; Omama, M.; Chalaki, B.; Tadiparthi, V.; Pari, E.M.; Chinchali, S. R3DM: Enabling Role Discovery and Diversity Through Dynamics Models in Multi-Agent Reinforcement Learning. *arXiv* **2025**, arXiv:2505.24265.
29. Barbosa, J.; Leitão, P.; Adam, E.; Trentesaux, D. Dynamic self-organization in holonic multi-agent manufacturing systems: The ADACOR evolution. *Comput. Ind.* **2015**, *66*, 99–111.
30. Rahmanzadeh, A.; Rahmani, M.; Nazemi, E. A Model for Selecting the Best Organizational Paradigm at Run Time for Self-Organizing Multi-Agent Systems; an IoT Case Study. *Int. J. Inf. Commun. Technol. Res.* **2019**, *11*, 29–39.
31. Köhler-Bußmeier, M.; Capra, L. Modelling and Simulation of Adaptive Multi-Agent Systems with Stochastic Nets-Within-Nets. In *Proceedings of the 16th International Joint Conference on Computational Intelligence*; SciTePress: Setúbal, Portugal, 2024; pp. 313–320.
32. Yang, Y.; Chai, H.; Shao, S.; Song, Y.; Qi, S.; Rui, R.; Zhang, W. Agentnet: Decentralized evolutionary coordination for llm-based multi-agent systems. *arXiv* **2025**, arXiv:2504.00587.
33. Wang, N.; Zhang, Y.; Liu, S.; Zhao, X.; Cheng, G.; Xu, Y. DRAMA: A Dynamic and Robust Allocation-Based Multi-Agent System for Changing Environments. *arXiv* **2025**, arXiv:2508.04332.
34. Tastan, N.; Horvath, S.; Nandakumar, K. Stochastic Self-Organization in Multi-Agent Systems. *arXiv* **2025**, arXiv:2510.00685.
35. Kitano, H.; Tadokoro, S. Robocup rescue: A grand challenge for multiagent and intelligent systems. *AI Mag.* **2001**, *22*, 39–39.
36. Tadokoro, S.; Kitano, H.; Takahashi, T.; Noda, I.; Matsubara, H.; Shinjoh, A.; Koto, T.; Takeuchi, I.; Takahashi, H.; Matsuno, F.; et al. The robocup-rescue: An it challenge to emergency response problem in disaster. In *2000 26th Annual Conference of the IEEE Industrial Electronics Society. IECON 2000. 2000 IEEE International Conference on Industrial Electronics, Control and Instrumentation. 21st Century Technologies*; IEEE: New York, NY, USA, 2000; Volume 1, pp. 126–131.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.