Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Модель адаптивної мультиагентної системи для динамічної кооперації та конкуренції в ігровому середовищі, що ґрунтується на принципах MARL
(тема)

Виконав:
здобувач другого року навчання,
групи СШІм-24-1
Костянтин ПАДАЛКІН
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва освітньої програми)
Керівник проф. Олександр ШЕВЧЕНКО
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Лариса ЧАЛА
(підпис) (власне ім'я, прізвище)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Падалкіну Костянтину Дмитровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Модель адаптивної мультиагентної системи для динамічної кооперації та конкуренції в ігровому середовищі, що ґрунтується на принципах MARL

затверджена наказом університету від 6 квітня 20 26 р. № 269Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 19 травня 20 26 р.

3. Вихідні дані до роботи документації з використаних бібліотек, наукові статті та книжки, що пов'язані з мультиагентними системами

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної галузі та постановка задачі

2) Теоретичні дослідження

3) Проєктування та програмна реалізація системи

4) Експериментальні дослідження та аналіз результатів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	06.04.2026	виконано
2	Аналіз предметної галузі	08.04.2026	виконано
3	Постановка задачі	09.04.2026	виконано
4	Теоретичні дослідження	19.04.2026	виконано
5	Проєктування системи	23.04.2026	виконано
6	Програмна реалізація	01.05.2026	виконано
7	Експериментальні дослідження	08.05.2026	виконано
8	Аналіз результатів	09.05.2026	виконано
9	Написання пояснювальної записки	11.05.2026	виконано
10	Захист перед ЕК	19.05.2026	виконано

Дата видачі завдання 6 квітня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. Олександр ШЕВЧЕНКО
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 119 с., 6 рис., 2 табл., 2 дод., 14 джерел.

АКТОР, ДИНАМІЧНА КООПЕРАЦІЯ ТА КОНКУРЕНЦІЯ, КРИТИК, МУЛЬТИАГЕНТНЕ НАВЧАННЯ З ПІДКРІПЛЕННЯМ, CTDE, MAPPO, PETTINGZOO, PYTORCH.

Об’єктом дослідження є мультиагентні системи навчання з підкріпленням у частково спостережуваних динамічних середовищах.

Предметом дослідження є методи та моделі адаптивної поведінки агентів, які забезпечують динамічний перехід між кооперацією і конкуренцією в єдиному ігровому середовищі.

Мета роботи полягає в розробці та дослідженні моделі адаптивного мультиагентного навчання з підкріпленням, здатної автоматично змінювати ступінь кооперативної або конкурентної поведінки агентів залежно від поточного стану середовища, доступних ресурсів і спостережуваної поведінки інших агентів.

Методами дослідження є аналіз наукової літератури, порівняльний аналіз існуючих MARL-підходів, моделювання мультиагентного середовища, проектування та реалізація нейромережевої архітектури, навчання моделей і обробка результатів.

Результатом дослідження є розроблене оригінальне ігрове середовище Adaptive Coopetition Gridworld, програмна реалізація п’яти архітектурних варіантів агента у рамках єдиного класу через незалежні архітектурні прапорці, а також систематичне абляційне дослідження. Робота уточнює умови застосовності контекстних механізмів у MARL та має методологічну цінність як приклад мульти-сидового абляційного дослідження з виявленою пізньої фазовою збіжністю.

ABSTRACT

Master’s thesis contains: 119 pp., 6 fig., 2 tabl., 2 ann., 14 references.

ACTOR, CRITIC, CTDE, DYNAMIC COOPERATION AND COPMETITION, MAPPO, MULTI-AGENT-REINFORCEMENT LEARNING, PETTINGZOO, PYTORCH.

The object of the research is multi-agent reinforcement learning systems in partially observable dynamic environments.

The subject of the research is methods and models of adaptive agent behavior that ensure a dynamic transition between cooperation and competition within a single game environment.

The aim of the work is to develop and study a model of adaptive multi-agent reinforcement learning capable of automatically changing the degree of cooperative or competitive behavior of agents depending on the current state of the environment, available resources, and the observed behavior of other agents.

The research methods include analysis of the scientific literature, comparative analysis of existing MARL approaches, modeling of a multi-agent environment, design and implementation of a neural network architecture, model training, and result processing.

The result of the research is the development of an original game environment, Adaptive Coopetition Gridworld, a software implementation of five architectural variants of the agent within a single class through independent architectural flags, as well as a systematic ablation study. The work clarifies the conditions under which contextual mechanisms are applicable in MARL and has methodological value as an example of a multi-side ablation study with identified late-phase convergence.

ЗМІСТ

ВСТУП

Мультиагентне навчання з підкріпленням в останні роки стало одним із найбільш активно розвиваючихся напрямків штучного інтелекту, оскільки воно дозволяє моделювати і навчати системи, у яких кілька агентів приймають рішення в спільному середовищі, впливаючи як на власний результат, так і на поведінку інших учасників взаємодії [1]. Практичний інтерес до MARL зумовлений тим, що значна частина реальних задач є за своєю природою багатоагентними, наприклад координація автономних роботів, транспортна логістика, стратегічні ігри.

Класичні дослідження в галузі MARL традиційно поділяють режими взаємодії на два типи: кооперативні сценарії, де агенти спільно максимізують командну нагороду, та конкурентні ігри, де виграш одного учасника дорівнює програшу іншого [1]. Проте реальні динамічні середовища – від багатокористувацьких ігор до розподілених автономних систем – характеризуються змішаною структурою взаємодії, де кооперація та конкуренція виникають одночасно і безперервно змінюються залежно від стану середовища [2]. Для таких умов важливою стає здатність агента не просто діяти по заданій стратегії, а адаптувати поведінку до поточного контексту без явного перемикання між окремими політиками.

Більшість сучасних алгоритмів і бенчмарків MARL – PettingZoo, SMAC, LBF, Overcooked – передбачають статичний режим взаємодії, заданий на етапі проектування середовища. Динамічна зміна режимів кооперації та конкуренції всередині одного безперервного епізоду залишається недостатньо вивченою: про це прямо свідчить огляд, присвячений гібридним сценаріям у MARL, де зафіксовано «помітний дефіцит у систематичній класифікації змішаних кооперативно-конкурентних сценаріїв та уніфікованої концептуальної рамки». Відсутні також відкриті бенчмарки, у яких кооперативність та конкурентність

виникає емерджентно за рахунок внутрішніх механізмів середовища, а не за рахунок зовнішнього переключення сценаріїв [3].

Вирішення цієї проблеми пов’язане з розробкою адаптивних MARL систем, здатних у реальному часі оцінювати контекст середовища та перебудовувати стратегію поведінки всередині єдиної нейронної мережі. Такий підхід відповідає концепції кооперативної конкуренції (coopetition), описаній ще Бранденбургером і Нейлбаффом у 1996 році, і знаходить нове втілення в задачах мультиагентного навчання з підкріпленням [4].

Окремою науковою прогалиною є відсутність систематичних емпіричних досліджень того, які саме архітектурні компоненти мультиагентної політики є критичними для ефективного навчання у середовищах з динамічною coopetition. У літературі пропонуються різні контекстні механізми – рекурентні мережі для накопичення історії спостережень, окремі мета-модулі для подання поточного режиму середовища, централізовані критики з повним станом, – проте їх відносний внесок у частково спостережуваних coopetition-середовищах не вимірювався в єдиному контрольованому експерименті.

Ціллю роботи є розробка та дослідження моделі адаптивного мультиагентного навчання з підкріпленням, що здатна автоматично змінювати ступінь кооперативної або конкурентної поведінки агентів залежно від поточного стану середовища, доступних ресурсів і спостережуваної поведінки інших агентів.

Такі системи особливо потрібні в ігрових симуляціях, де динаміка coopetition робить сценарії більш реалістичними та цікавими для гравців, а також відкривають перспективи перенесення результатів на завдання автономних систем, таких як логістика та робототехніка.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис предметної галузі

Мультиагентні системи становлять клас інтелектуальних систем, у яких кілька автономних агентів діють у спільному середовищі, приймають рішення на основі власних спостережень і тією чи іншою мірою впливають один на одного. У контексті навчання з підкріпленням такі системи розглядаються як одна з найскладніших і водночас найбільш перспективних форм постановки задачі, оскільки кожен агент мусить враховувати не лише стан середовища, а й мінливу поведінку інших учасників взаємодії. Багатоагентне навчання особливо актуальне для задач, де потрібне узгоджене прийняття рішень у розподіленому середовищі [2].

Класифікація мультиагентних систем проводиться за кількома критеріями. За ступенем однорідності агентів розрізняють гомогенні (однорідні) та гетерогенні системи, де агенти відрізняються архітектурою, сенсорикою та набором дій. За характером взаємозв’язку між агентами поділяють на кооперативні, конкурентні та змішані кооперативно-конкурентні сценарії. У кооперативних задачах агенти прагнуть максимізувати загальний результат або загальну нагороду, у конкурентних вони діють в опозиційних інтересах, а в змішаних – одночасно присутні як спільні, так і конфліктні цілі [5]. Саме змішані сценарії особливо важливі для прикладних досліджень, оскільки вони ближчі до реальних умов, де учасники можуть тимчасово співпрацювати для досягнення локальної вигоди, а потім конкурувати за обмежені ресурси.

Іншим важливим фактором класифікації є ступінь централізації управління. У повністю централізованих системах один управляючий модуль приймає рішення за всіх агентів, тоді як у повністю децентралізованих системах кожен агент навчається та діє незалежно. На

практиці найбільш поширеною та зручною для MARL є centralized training with decentralized execution, при якому навчання може використовувати глобальну інформацію, але в момент виконання кожен агент спирається лише на свої локальні спостереження [6]. Такий підхід дозволяє поєднувати переваги глобального аналізу середовища із практичною реалізовністю децентралізованої політики.

Рівень часткової спостережуваності середовища визначає, наскільки повно агент може оцінити глобальний стан системи. Більшість реальних задач MARL є саме частково спостережуваними.

Для цієї роботи суттєвими є саме часткова спостережуваність, однорідність агентів і необхідність динамічної зміни типу поведінки залежно від режиму середовища. Саме ця комбінація властивостей визначає як вибір базового алгоритму, так і специфіку розробленого середовища Adaptive Coopetition Gridworld.

1.2 Навчання з підкріпленням в мультиагентному середовищі

Навчання з підкріпленням у мультиагентному середовищі має низку фундаментальних особливостей, які суттєво відрізняють його від класичного одноагентного RL. У випадку одноагентного навчання середовище зазвичай вважається стаціонарним: імовірності переходів між станами та функція винагороди не змінюються з часом. У MARL це припущення порушується, оскільки кожен агент одночасно навчається та оновлює власну політику. Внаслідок цього середовище для окремого агента постійно змінюється через зміну поведінки інших агентів. Така властивість отримала назву проблеми нестаціонарності.

Нестаціонарність є однією з головних причин нестабільності навчання у MARL. Алгоритм, який у певний момент часу вважає поведінку інших агентів частиною динаміки середовища, вже через декілька ітерацій стикається з іншою поведінкою, оскільки політики партнерів або суперників

змінилися. Це порушує базові припущення багатьох класичних алгоритмів RL, зокрема Q-learning, які спираються на стаціонарність переходів. У результаті оцінки цінності станів і дій можуть швидко застарівати, а сам процес навчання стає нестабільним або навіть розходиться.

Вже у ранніх роботах із мультиагентного навчання в кооперативно-конкурентних середовищах було показано, що стандартні методи Q-learning та policy gradient стикаються з серйозними труднощами при збільшенні кількості агентів. Зі збільшенням числа агентів ускладнюється оцінювання того, які саме дії привели до отриманої винагороди, через що сигнали навчання стають шумними та менш інформативними.

Другою суттєвою особливістю MARL є часткова спостережуваність середовища. У більшості реальних задач агент не має доступу до повного глобального стану системи та може спостерігати лише локальну інформацію. Причинами цього можуть бути фізичні обмеження сенсорів, відстань між агентами, обмеження комунікації або стратегічне приховування інформації в конкурентних сценаріях. У таких умовах кожен агент змушений приймати рішення на основі неповних або зашумлених даних, що значно ускладнює навчання оптимальної політики.

Часткова спостережуваність тісно пов’язана з проблемою нестаціонарності. Якщо агент не бачить повного стану системи, йому складніше відрізнити реальну зміну середовища від зміни поведінки інших агентів. У розподілених системах з локальним сприйняттям ці дві проблеми взаємно підсилюють одна одну та призводять до значного ускладнення процесу навчання. Саме тому сучасні MARL-архітектури часто використовують механізми агрегування глобального контексту, централізованих критиків, attention-механізми або рекурентні модулі пам’яті для компенсації неповноти спостережень.

Третьою фундаментальною проблемою MARL є проблема розподілу відповідальності (credit assignment). Її сутність полягає у визначенні того, який саме агент і якою мірою вплинув на загальний результат системи. У

кооперативних середовищах агенти часто отримують спільну командну винагороду, однак із цього сигналу неочевидно, чиї дії були корисними, а чиї – нейтральними або навіть шкідливими. Через це агентам складно зрозуміти, яку поведінку необхідно підсилювати в процесі навчання.

Проблема credit assignment особливо критична у складних кооперативних сценаріях із великою кількістю агентів, де успіх залежить від довготривалої координації. Просте використання спільної винагороди часто призводить до появи ледачих агентів, які отримують вигоду від дій інших, не роблячи власного внеску. Для подолання цієї проблеми у сучасному MARL застосовуються централізовані критики, декомпозиція функцій цінності, а також різні форми факторизації або контекстного представлення спільного стану [2]. Такі механізми дозволяють точніше оцінити індивідуальний внесок агентів і зробити процес навчання більш стабільним та інформативним.

Ще однією важливою особливістю MARL є проблема масштабованості. При збільшенні кількості агентів експоненційно зростає розмір спільного простору станів та дій. Це ускладнює як оцінку функції цінності, так і пошук оптимальної політики. У великих мультиагентних системах традиційні централізовані підходи швидко стають обчислювально непридатними через надмірну складність моделювання всіх можливих взаємодій між агентами.

Крім обчислювальної складності, проблема масштабованості також впливає на узагальнюваність політик. Політика, навчена для певної кількості агентів або конкретної конфігурації середовища, часто погано переноситься на інші сценарії. У динамічних системах, де кількість агентів або структура взаємодій може змінюватися з часом, це стає особливо критичним. Саме тому сучасні підходи MARL прагнуть використовувати параметричне спільне навчання, механізми інваріантності до кількості агентів та компактні контекстні представлення глобального стану.

Сукупність описаних проблем обґрунтовує використання парадигми CTDE (Centralized Training with Decentralized Execution) як базового підходу в даній роботі. CTDE дозволяє поєднати переваги централізованого доступу до глобальної інформації під час навчання зі здатністю агентів діяти автономно під час виконання. Завдяки цьому вдається зменшити вплив нестаціонарності, покращити координацію між агентами та забезпечити практичну придатність системи до реальних розподілених середовищ.

1.3 Аналіз підходів до моделювання кооперативної та конкурентної поведінки агентів

Підходи до моделювання поведінки агентів у мультиагентних системах історично розвивалися у двох основних напрямках: кооперативному та конкурентному. Кооперативна поведінка передбачає спільну максимізацію загальної функції корисності всіма агентами. Класичним прикладом є модель із загальною нагородою, де кожен агент отримує однаковий сигнал винагороди за досягнення колективної мети. Такі системи широко використовуються у завданнях координації роботів, логістики та колективного прийняття рішень.

Конкурентна поведінка, навпаки, будується на принципах ігор з нульовим виграшом (zero-sum), де виграш одного агента точно дорівнює програшу іншого. Тут традиційно застосовуються методи мінімізації максимальної шкоди чи алгоритми, засновані на теорії ігор. У мультиагентному навчанні з підкріпленням такі сценарії реалізуються в середовищі типу StarCraft II або у простих zero-sum іграх.

MADDPG вперше показав, що врахування дій інших агентів при навчанні суттєво підвищує ефективність у змішаних кооперативно-конкурентних середовищах. Пізніше ця ідея отримала розвиток у більш стійких on-policy варіантах на базі PPO, що зрештою призвело до MAPPO як де-факто стандарту для кооперативного MARL.

Таким чином, аналіз підходів до моделювання кооперації та конкуренції показує, що найбільший інтерес становлять не статично визначені сценарії, а середовища, в яких режим взаємодії динамічно змінюється. Для таких завдань потрібна архітектура, здатна сприймати контекст поточного режиму та адаптувати стратегію поведінки без ручного перемикання між окремими політиками.

1.4 Огляд існуючих алгоритмів мультиагентного навчання з підкріпленням

Сучасні алгоритми MARL можна умовно поділити на кілька великих груп, і кожна з них по-своєму відповідає на проблему нестаціонарності, часткової спостережуваності та складної взаємодії між агентами. Методи незалежного навчання є найпростішими в реалізації: кожен агент оновлює свою політику майже так, ніби інші учасники є частиною середовища. Саме так працює Independent Q-Learning, який історично вважають одним із базових орієнтирів для MARL. Перевага цього підходу полягає в простоті, але його суттєвий недолік – слабка стійкість до змін політик інших агентів, через що він зазвичай слугує радше нижньою межею якості, ніж практично оптимальним рішенням для складних середовищ.

Другу велику групу становлять методи декомпозиції ціннісних функцій, до яких належать VDN, QMIX і QPLEX. VDN розглядає спільну цінність як суму індивідуальних Q-функцій і орієнтується на кооперативні задачі зі спільною винагородою. QMIX розвиває цю ідею, використовуючи монотонну нелінійну змішувальну мережу, яка поєднує індивідуальні оцінки агентів у спільну цінність і тим самим дозволяє отримати децентралізовану політику, навчену в централізованому режимі. QPLEX іде далі та намагається краще реалізувати принцип IGM, поєднуючи більшу виразність представлення з узгодженістю між індивідуальним і спільним вибором дій. Усі ці підходи особливо сильні там, де є спільна командна

мета, але їхня природна область застосування все ж ближча до кооперативних сценаріїв, ніж до задач зі змішаною або змінною структурою взаємодії [1].

Найбільш універсальну лінію розвитку в сучасному MARL утворюють актор-критик методи в парадигмі CTDE. Їхня основна ідея полягає в тому, що під час навчання критик може використовувати глобальну інформацію, тоді як актор на етапі виконання працює лише з локальним спостереженням. Особливо показовим тут є MAPPO: у роботі Chao Yu та співавторів було показано, що PPO-орієнтовані мультиагентні алгоритми демонструють дуже сильні результати в particle-world, SMAC, Google Research Football і Hanabi без доменно-специфічних модифікацій архітектури. Це зумовило широке застосування MAPPO як базового алгоритму для кооперативного MARL.

Окремий клас становлять методи з комунікацією та механізмами уваги, які прагнуть зменшити нестаціонарність через явне моделювання поведінки сусідніх агентів або передачу додаткового контексту. У цій роботі досліджується інтеграція мета-енкодера як контекстного модуля в рамках єдиної політики MAPPO, при цьому окремо вивчається внесок кожного компонента у кінцеву ефективність.

1.5 Методи врахування контексту та адаптивної поведінки в мультиагентних системах

Контекстне представлення в мультиагентному навчанні з підкріпленням є одним із головних механізмів, що забезпечують адаптивність поведінки агентів у складних динамічних середовищах. На відміну від простого використання локального спостереження, контекстне представлення дозволяє агенту враховувати не лише поточний стан, який він безпосередньо спостерігає, а й ширшу інформацію про середовище, взаємодію між учасниками системи та поточний режим задачі. Це особливо

важливо в умовах часткової спостережуваності та нестаціонарності, коли окреме спостереження може бути недостатнім або навіть двозначним з погляду прийняття оптимального рішення.

До складу контексту можуть входити різні види інформації. Зокрема, це може бути агрегована статистика дій інших агентів, яка відображає загальну структуру поведінки в системі та допомагає оцінити, чи переважає кооперація, конкуренція або змішаний режим взаємодії. Також контекст може включати поточний тип задачі, наприклад інформацію про те, чи функціонує система в кооперативному сценарії, у конкурентному середовищі або в coopetition-режимі, де одночасно присутні і спільні, і конфліктні інтереси. Окрім цього, до контексту можуть входити ознаки режиму середовища, такі як рівень ризику, інтенсивність конфлікту, зміна доступності ресурсів або інші глобальні характеристики, які впливають на оптимальну стратегію поведінки.

Окреме значення має короткострокова історія взаємодії. У багатьох задачах поточне спостереження саме по собі не дає змоги однозначно відновити прихований стан системи, тоді як послідовність попередніх дій і спостережень містить важливу інформацію про динаміку середовища. Наприклад, однаковий локальний стан може виникати на різних етапах епізоду або за різних стратегій інших агентів. У такому випадку аналіз короткої історії дозволяє агенту реконструювати прихований контекст і приймати більш обґрунтовані рішення. Саме тому сучасні MARL архітектури часто використовують механізми пам’яті або часової агрегації інформації.

Серед підходів до формування контексту найбільш поширеними є рекурентні архітектури. Використання LSTM, GRU або подібних механізмів дозволяє агенту накопичувати інформацію про попередні стани та дії у власному внутрішньому прихованому представленні. Такий підхід частково компенсує неповноту спостережень і забезпечує можливість моделювання часових залежностей. Рекурентні модулі особливо корисні в середовищах,

де важлива не лише поточна конфігурація системи, а й траєкторія її попереднього розвитку. Вони дозволяють агенту не просто реагувати на миттєвий сигнал, а будувати поведінку з урахуванням попереднього досвіду.

Іншим важливим напрямом є механізми уваги над спостереженнями або представленнями інших агентів. Attention-моделі дають змогу динамічно визначати, яка інформація в конкретний момент є найбільш релевантною. У мультиагентному середовищі це особливо важливо, оскільки кількість потенційно значущих взаємодій може бути великою, а не всі агенти однаково впливають на рішення в конкретній ситуації. Механізми уваги дозволяють моделі зосереджуватися на найближчих, найактивніших або стратегічно важливих агентах, що підвищує якість контекстного представлення та робить його більш гнучким.

Окрему групу становлять мета-модулі, які формують низьковимірне представлення поточного режиму середовища. На відміну від простої агрегації спостережень, такі модулі намагаються виділити приховані фактори, що визначають загальну структуру взаємодії в системі. Це може бути, наприклад, латентний вектор, який кодує рівень кооперативності, домінуючий тип стратегії суперників або інші системні характеристики, важливі для вибору дії. Перевагою такого підходу є компактність та здатність узагальнювати складні ситуації у вигляді невеликого контекстного embedding, який легко інтегрувати в політику агента.

Важливо, щоб контекстний модуль не руйнував основну перевагу парадигми CTDE – децентралізоване виконання. Саме ця властивість робить CTDE практично придатною для реальних багатоагентних систем, де агенти повинні діяти автономно, не маючи доступу до глобального стану в момент прийняття рішення. Тому контекстне подання має бути організоване так, щоб повна інформація використовувалася лише на етапі навчання централізованим критиком, тоді як актор під час виконання спирається лише на локальне спостереження, а за потреби — на власний внутрішній

стан рекурентної мережі або на компактне контекстне представлення, доступне без порушення децентралізації.

У такій постановці централізований критик відповідає за більш точну оцінку цінності стану з урахуванням глобального контексту, тоді як актор навчається приймати локально доступні, але контекстно обґрунтовані рішення. Це дозволяє поєднати переваги глобальної обізнаності на етапі навчання з гнучкістю та автономністю на етапі виконання. У результаті модель стає здатною не лише реагувати на миттєві спостереження, а й адаптувати поведінку до зміни режиму взаємодії, структури команди або стратегії інших агентів.

Попри значний прогрес у розвитку контекстно-адаптивних методів, у літературі все ще бракує систематичних емпіричних даних щодо того, який саме клас контекстних механізмів є найбільш ефективним у динамічних coopetition-середовищах. Різні роботи використовують рекурентні модулі, attention-механізми, однак їхній відносний внесок у стабільність навчання, швидкість адаптації та кінцеву якість політики часто не порівнюється в єдиних умовах. Через це залишається відкритим питання, який саме тип контекстного представлення найбільш доцільний для задач, де одночасно присутні кооперація, конкуренція та зміна режиму середовища. Саме тому виникає потреба у систематичному абляційному дослідженні, яке дозволить оцінити ефективність різних механізмів контексту в межах однієї MARL архітектури та встановити їхній реальний внесок у адаптивність агентної поведінки.

1.6 Аналіз існуючих середовищ та бенчмарків для MARL

Якість MARL-досліджень багато в чому визначається тим, наскільки добре підібране середовище для експериментів. Для багатоагентних завдань особливе значення мають стандартні бенчмарки, які дозволяють відтворювано порівнювати алгоритми та виявляти не тільки їхню середню

ефективність, а й стійкість, здатність до координації та переносимість на нові умови. У цьому сенсі PettingZoo відіграє роль універсальної інфраструктурної основи, а SMAC, Overcooked і SISL є змістовними тестовими середовищами для різних класів завдань.

PettingZoo є стандартним API для мультиагентного навчання з підкріпленням і розроблений як простий, pythonic інтерфейс для представлення загальних MARL-задач. Його практична цінність полягає в тому, що він забезпечує єдиний спосіб взаємодії з багатьма середовищами, підтримує як AEC-модель для послідовних ходів, так і Parallel API для одночасних дій, а також містить набір reference environments і допоміжних інструментів для створення власних середовищ. Саме уніфікація інтерфейсів робить PettingZoo особливо корисним для відтворюваних експериментів і для коректного порівняння алгоритмів між різними реалізаціями [7].

StarCraft Multi-Agent Challenge (SMAC) є одним із найвідоміших кооперативних бенчмарків у MARL. Він побудований на StarCraft II і зосереджується на задачах мікрокерування, де кожен юніт контролюється окремим агентом і приймає рішення на основі лише локальних спостережень. Завдяки цьому SMAC добре підходить для оцінювання координації, розподілу ролей, спільної атаки та стабільності навчання в умовах часткової спостережуваності. У самій постановці задачі SMAC також підкреслюється, що до його появи для cooperative MARL бракувало стандартизованого бенчмарку, співставного за роллю з ALE або MuJoCo для одноагентного RL [8].

Overcooked та, зокрема, Overcooked Generalisation Challenge (OGC) використовуються для дослідження кооперації в середовищах, де важливі послідовність дій, синхронізація з партнером і переносимість поведінки на нові умови. OGC було запропоновано як новий бенчмарк для оцінювання здатності агентів співпрацювати з невідомими партнерами в незнайомих середовищах; він розширює Overcooked-AI так, щоб перевіряти не тільки

якість поведінки в навчальному середовищі, а й zero-shot generalisation. У роботі також наголошується, що OGC є повністю open-source, GPU-accelerated і містить значно багатший простір дизайну рівнів, ніж попередні варіанти для curriculum-based навчання [9].

Окреме місце серед кооперативних бенчмарків займає SISL. Це набір із трьох кооперативних мультиагентних середовищ, створених і оприлюднених у Stanford Intelligent Systems Laboratory. Наявність таких середовищ корисна тоді, коли потрібно протестувати алгоритм на класичних кооперативних сценаріях, але без надмірної складності великих ігрових симуляторів на кшталт StarCraft II. Для дослідника це зручно ще й тому, що PettingZoo підтримує SISL як окремий пакет, а в документації підкреслюється, що для цих середовищ були внесені виправлення помилок, тож результати слід інтерпретувати обережно при порівнянні зі старими роботами.

Таким чином, існуючі бенчмарки добре покривають або кооперативні сценарії, або завдання узагальнення кооперації, але не повною мірою відображають ситуацію, в якій в тому самому середовищі динамічно змінюються конкурентні та кооперативні режими. Тому для даної роботи потрібно власне середовище, яке об’єднує обидва типи взаємодії і дозволяє оцінювати як підсумкову нагороду, а й здатність агента до переключення поведінки.

1.7 Постановка задачі

Проведений аналіз засвідчує, що сучасні алгоритми MARL у принципі застосовні до середовищ зі змішаним режимом взаємодії, проте у літературі недостатньо систематичних емпіричних даних про те, які саме архітектурні компоненти політики є критичними для ефективного навчання у середовищах з динамічною coopetition. Зокрема, відносний внесок

рекурентної пам’яті, централізованого критика та мета-енкодера у єдиному контрольованому експерименті не вимірювався.

Задача цієї роботи полягає в розробці та дослідженні моделі адаптивного мультиагентного навчання з підкріпленням для ігрового середовища, у якому одночасно присутні конкурентні ресурси та кооперативні кризові події. Треба спроектувати експериментальну інфраструктуру та провести систематичне абляційне дослідження для оцінки впливу зазначених архітектурних компонентів на якість навчання MAPPO-агента у частково спостережуваному coopetition-середовищі. Для цього необхідно:

– розробити мультиагентне середовище з переключенням між конкурентним і кооперативним режимами;

– реалізувати агента в рамках парадигми CTDE з трьома незалежно вмикуваними архітектурними компонентами: рекурентний шар, мета-енкодер у критику, централізований критик;

– реалізувати та провести тренування моделей;

– провести абляційне порівняння п’яти конфігурацій з використанням мульти-сидової методології та t-критерію Велча для оцінки статистичної значущості різниць;

– проаналізувати отримані результати.

2 ТЕОРЕТИЧНІ ДОСЛІДЖЕННЯ

2.1 Формалізація мультиагентного середовища в моделях навчання з підкріпленням

Мультиагентне навчання з підкріпленням вимагає математичної формалізації середовища, в якому взаємодіють кілька автономних агентів. На відміну від одноагентного RL, де середовище описується класичною моделлю марківського процесу прийняття рішень, у MARL середовище стає значно складнішим через взаємний вплив агентів один на одного. Це призводить до необхідності використання розширених моделей, таких як частково спостережувані стохастичні ігри (POSG) та децентралізовані частково спостережувані марківські процеси прийняття рішень (Dec-POMDP). У загальному випадку середовище в POSG задається наступним кортежем [10]:

$$(N, S, A_i, O_i, T, Z_i, r_i, D, \gamma, b_0), \tag{2.1}$$

де N – множина агентів;

S – множина можливих станів;

A_i – простір дій і-го агента;

O_i – множина спостережень і-го агента;

T – функція переходів станів;

Z_i – функція спостереження і-го агента;

r_i – функція винагороди і-го агента;

D – кількість часових кроків;

γ – коефіцієнт дисконтування;

b_0 – початковий розподіл серед станів.

Кожен агент i у момент часу t отримає локальне спостереження та вибирає дію на основі своєї історії. Спільна політика максимізує очікувану дисконтовану суму винагород.

Середовище ACG формалізується як контекстний Dec-POMDP з динамічно змінюваною функцією винагороди: особисті монети вводять конкурентний компонент, а кризи – кооперативний. Поточний режим визначається станом середовища, а не фіксується заздалегідь, що зберігає марківську властивість і відповідає формалізму POSG для змішаних сценаріїв [11].

2.2 Принцип CTDE

Centralized Training with Decentralized Execution є домінуючим підходом у сучасних алгоритмах MARL, оскільки ефективно вирішує проблеми нестаціонарності та розподілу відповідальності, зберігаючи при цьому масштабованість та практичну придатність. Під час навчання використовується глобальна інформація про спільний стан і дії всіх агентів, а на етапі виконання кожен агент діє автономно, спираючись лише на локальну історію [12].

Критик максимізує спільну функцію цінності, тоді як актор кожного агента оновлюється за локальними градієнтами. На етапі виконання кожен агент діє виключно на основі своєї локальної історії, не маючи доступу до глобального стану та не здійснюючи комунікації. Політика агента є повністю незалежною.

Перевагою такого підходу є стабілізація навчання: критик бачить повну картину і тим самим зменшує ефект нестаціонарності, яку решта агентів сприймають як частину динамічного середовища.

У цій роботі парадигма CTDE використовується як базова. Мета-енкодер формує контекстний embedding виключно на основі глобальної інформації, доступної під час навчання, але цей embedding конкатенується

лише з локальним спостереженням агента. Мета-енкодер інтегрується виключно у централізованого критика і не передається в актора. Таким чином, під час виконання агент залишається повністю децентралізованим, а під час навчання контекстний модуль дозволяє політиці адаптуватися до динаміки coopetition без порушення незалежності політик. Контекстний embedding, сформований на основі глобальної інформації – кількості активних криз, рівня кооперації інших агентів, часу з моменту останньої кризової події – використовується разом з повним станом середовища лише на етапі обчислення оцінки цінності. Актор приймає рішення на основі двох джерел: локального вікна навколо агента та внутрішнього прихованого стану рекурентної мережі, що накопичує інформацію про історію локальних спостережень. Крім того, CTDE добре поєднується з сучасними алгоритмами оптимізації політик, такими як PPO, що робить його природним вибором для базового алгоритму MAPPO у запропонованій моделі.

2.3 Алгоритм MAPPO

Метод MAPPO (Multi-Agent Proximal Policy Optimization), запропонований Chao Yu у 2021 році, є однією з найефективніших і найуніверсальніших реалізацій парадигми CTDE (Centralized Training with Decentralized Execution) у сучасних мультиагентних системах. Алгоритм базується на ідеях класичного PPO і адаптує їх до умов багатоагентної взаємодії, де декілька агентів одночасно навчаються та впливають один на одного. Попри відносну простоту архітектури, MAPPO продемонстрував високу ефективність у широкому спектрі середовищ, включаючи кооперативні, конкурентні та змішані сценарії. У низці експериментів алгоритм перевершив або досяг рівня більш спеціалізованих методів, зокрема QMIX та MADDPG [13], що підтверджує його універсальність і практичну придатність.

Особливістю MAPPO є використання парадигми централізованого навчання та децентралізованого виконання. Під час етапу навчання алгоритм має доступ до глобальної інформації про середовище та стани всіх агентів, що дозволяє зменшити проблему нестаціонарності та забезпечити більш стабільне оновлення параметрів. Водночас під час виконання кожен агент приймає рішення автономно, використовуючи лише локальні спостереження, що робить систему придатною для реальних децентралізованих сценаріїв.

У MAPPO кожен агент і має свою політику. Централізований критик оцінює очікувану сумарну нагороду всіх агентів за глобального стану. Навчання здійснюється у два етапи. Навчання актора здійснюється шляхом максимізації сурогатної цільової функції:

$$L^{CLIP}(\theta)=-E_t\left[\min\left(r_t(\theta)\widehat{A_t}, \text{clip}(r_t(\theta),1-\varepsilon,1+\varepsilon)\widehat{A_t}\right)\right], \quad (2.2)$$

де $r_t(\theta)=\pi_\theta\left(a_t^i \mid s_t^i\right)/\pi_{\theta_{old}}\left(a_t^i \mid s_t^i\right)$ – відношення нових та старих ймовірностей дії;

$\widehat{A_t}$ – оцінка переваги, отримана з централізованого критика;

ε – поріг відсікання.

Критик мінімізує квадратичну похибку прогнозу цінності.

2.4 Порівняльний аналіз альтернативних методів MARL

Вибір MAPPO як базового методу обґрунтовується детальним порівняльним аналізом алгоритмів за п’ятьма критеріями, релевантними для динамічних середовищ: тип сценарію, механізм credit assignment, стійкість до нестаціонарності, обчислювальна складність та адаптивність до змінного контексту.

IQL є найпростішим серед розглянутих підходів, оскільки кожен агент навчається незалежно, без явного врахування дій інших учасників системи.

Така властивість робить метод зручним для базового порівняння, однак у багатоагентних середовищах вона суттєво обмежує якість координації. Через відсутність механізму спільного навчання IQL погано справляється з нестаціонарністю та не може ефективно моделювати взаємний вплив агентів, що особливо критично в динамічних сценаріях.

MADDPG є більш потужним підходом, оскільки використовує централізовану критику при децентралізованому виконанні дій. Це дозволяє частково враховувати взаємодію між агентами та краще працювати у змішаних або конкурентно-кооперативних середовищах. Водночас метод вимагає окремої політики для кожного агента, що ускладнює масштабування при зростанні кількості учасників. Крім того, MADDPG часто характеризується нестабільністю навчання та високою дисперсією градієнтів, що знижує його надійність у складних і змінних умовах.

QMIX орієнтований передусім на повністю кооперативні задачі, де загальна цінність команди розкладається на індивідуальні вкладки агентів за допомогою змішувальної мережі. Такий підхід добре працює, коли всі агенти мають спільну мету, однак він менш придатний для сценаріїв із динамічною структурою взаємодії або частковою конкуренцією. Крім того, жорсткі припущення щодо монотонності обмежують його гнучкість і зменшують ефективність у складніших середовищах.

COMA застосовує контрфактичний механізм credit assignment, що дозволяє оцінювати внесок кожного агента в загальний результат більш точно. Це робить метод концептуально сильним для кооперативних багатоагентних задач. Однак його практичне використання ускладнюється високою обчислювальною вартістю: для обчислення контрфактичної baseline необхідно враховувати альтернативні дії, що при великій кількості можливих рішень може ставати експоненційно дорогим. Через це COMA менш придатний для масштабних або швидко змінних середовищ.

MAPPO у цьому порівнянні є найбільш збалансованим рішенням. Він поєднує високу стабільність навчання, добру масштабованість і ефективну

роботу в умовах нестаціонарності, що особливо важливо для динамічних середовищ. Завдяки використанню централізованого критика та децентралізованого виконання MAPPO краще узгоджує навчання агентів і спрощує передачу контекстної інформації. Крім того, метод легко розширюється за рахунок додавання контекстного embedding, не потребуючи суттєвого перепроєктування функції втрат або структури алгоритму. Саме тому MAPPO є найбільш доцільним базовим методом для даної задачі.

2.5 Динамічна кооперація та конкуренція в мультиагентних системах

Динамічна coopetition у мультиагентному середовищі означає, що режим взаємодії між агентами не є фіксованим, а змінюється залежно від поточного стану середовища, доступності ресурсів і поведінки інших учасників. Бранденбургер і Нейлбафф визначили coopetition у рамках частково кооперативних ігор, де корисність гравця залежить від власних дій і від спільних стратегій суперників-партнерів.

У динамічному варіанті ця корисність є функцією часу та стану. В MARL динамічна coopetition формалізується як стохастична гра з прихованим режимом, що визначається спостережуваними ознаками середовища.

Складність динамічної coopetition – емерджентність режимів: режим не задається явним параметром, а виникає як результат взаємодії агентів із ресурсами. В ACG особисті монети генерують zero-sum компонент, а кризові події – shared reward із бонусом за спільну участь. Це відрізняє такі системи від традиційних кооперативних MARL бенчмарків, де структура взаємодії зазвичай заздалегідь задана і не змінюється протягом епізоду.

2.6 Контекстне уявлення стану

Контекстне уявлення стану використовується для того, щоб доповнити локальне спостереження агента інформацією, яка безпосередньо не входить у його сенсорне вікно, але істотно впливає на якість рішення. У MARL такі механізми особливо важливі, тому що часткова спостережуваність і нестаціонарність роблять лише локальну картину недостатньою для стійкої стратегії. В оглядах MARL та комунікаційних методів наголошується, що додатковий контекст може включати ознаки поведінки інших агентів, поточний тип завдання, агреговані характеристики середовища та історію взаємодії.

На відміну від жорстко встановлених правил перемикання режимів, такий підхід дозволяє моделі самостійно виявляти закономірності, коли вигідніше змагатися за власні ресурси, а коли співпрацювати для подолання кризи.

2.7 Модель запропонованого рішення

Запропонована модель поєднує три основні компоненти: частково спостережуване мультиагентне середовище ACG, CTDE-навчання на базі MAPPO та контекстний мета-енкодер, інтегрований у централізованого критика, що перетворює ознаки режиму взаємодії на векторне представлення, доступне для політики. Головною архітектурною характеристикою є строге розділення інформаційних потоків між актором (децентралізований, на етапі виконання) і критиком (централізований, на етапі навчання).

Актор отримує на вхід виключно локальне спостереження – тензор розміром 7×7×4. Чотири канали локального тензора кодують різні типи об’єктів окремо: канал 0 містить інших агентів у локальному вікні, канал 1 – особисті монети, канал 2 – активні пожежі, канал 3 – активних монстрів.

Таке роздільне кодування полегшує навчання згорткових фільтрів, оскільки кожен канал має чітку семантику і не вимагає від мережі додаткового розпізнавання категорії об’єкта.

Централізований критик отримує на вхід три джерела інформації: повний стан середовища – тензор 15×15×4 з тими ж семантичними каналами; глобальний дескриптор режиму – вектор з п’яти чисел: поточну кількість активних пожеж, поточну кількість активних монстрів, поточну кількість доступних монет на всій карті, середній рівень кооперації інших агентів за останні 10 кроків, кількість кроків з моменту появи останньої кризи; ідентифікатор агента у вигляді one-hot вектора, що дозволяє критику обчислювати оцінку цінності індивідуально для кожного агента.

Локальний спостерігач (LocalObserver) – згорткова нейронна мережа, що обробляє тензор 7×7×4 і виробляє 256-вимірне представлення. LSTM-модуль пам’яті – одношаровий рекурентний шар з прихованим розміром 256, що використовується для накопичення короткострокової історії локальних спостережень. Глобальний спостерігач (GlobalObserver) – згорткова мережа, що обробляє повний тензор 15×15×4 з адаптивним пулінгом, виробляючи 128-вимірне представлення, що використовується критиком. Мета-енкодер – повнозв’язна нейронна мережа з нормалізацією шарів та tanh-активацією на виході; формує контекстний embedding режиму, який також використовується виключно критиком. Така архітектура реалізує принцип CTDE строго: глобальна інформація і її контекстне уявлення повністю відсутні в потоці прийняття рішень агента.

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Вимоги до системи

Постановка задачі та аналіз існуючих підходів дозволяють сформулювати функціональні та нефункціональні вимоги до програмного середовища і моделі агента. Ці вимоги визначають як архітектуру системи, так і рамки експериментів.

До функціональних вимог системи належать:

– ігрове середовище повинно генерувати сценарії, у яких одночасно присутні елементи кооперації (загальні кризові події) та конкуренції (особисті ресурси), без явного перемикання між режимами;

– середовище повинно мати програмний інтерфейс, сумісний з PettingZoo Parallel API, для безпосередньої інтеграції зі стандартними MARL-фреймворками;

– кожен агент повинен приймати рішення на основі лише локального спостереження та внутрішнього стану рекурентної мережі;

– централізований критик повинен мати доступ до повного стану середовища і додаткового контекстного представлення режиму взаємодії, але цей доступ не повинен поширюватися на актора;

– система повинна забезпечувати запуск повного циклу навчання, збереження проміжних та фінальних етапів навчання моделі, а також автоматизоване оцінювання за заданими метриками;

– архітектура повинна допускати незалежне вмикання або вимикання окремих компонентів – мета-енкодера, рекурентної пам’яті, централізованого критика – для проведення абляційних експериментів.

До нефункціональних вимог належать:

– відтворюваність експериментів за рахунок фіксації випадкового зерна, детермінованих обчислень та чітких конфігураційних файлів;

– відсутність порушення парадигми CTDE на всіх рівнях системи;

– модульна структура коду, що забезпечує можливість заміни компонентів без переписування основного коду моделі та навчання;

– логування процесу навчання у форматі, придатному для аналізу за допомогою Wandb;

– швидкодія, достатня для запуску повного циклу навчання на одній GPU за обмежений час.

3.2 Концепція ігрового середовища Adaptive Coopetition Gridworld

В рамках цієї роботи пропонується авторське ігрове середовище Adaptive Coopetition Gridworld, що є дискретною двовимірною сіткою фіксованого розміру 15×15 клітин. На клітинах поля з’являються три типи об’єктів: особисті монети, пожежі та монстри. Особливістю середовища є те, що кооперативний і конкурентний компоненти співіснують у одному й тому ж епізоді: новий ресурс (монета чи криза) з’являється випадково у часі та просторі, тож режим взаємодії, що актуальний у даний момент, є наслідком стану середовища, а не зовнішнього параметра. Криза має обмежений час життя, по закінченні якого вона зникає без винагороди, отже агенти зацікавлені у швидкому реагуванні. Особисті монети також мають обмежений час життя.

Монети є джерелом конкурентного сигналу: за збір монети агент-першим, що увійшов на клітину, отримує винагороду +1, тоді як інші агенти не отримують нічого. У разі одночасного входу декількох агентів на клітину з монетою переможець обирається випадково, що усуває упередженість порядку обходу. Пожежі та монстри представляють кооперативні кризи: пожежа може бути ліквідована тільки за умови одночасної присутності щонайменше двох агентів на одній клітині, а монстр – трьох. У разі успішного подолання кризи усі учасники отримують спільну винагороду, що пропорційно розподіляється між ними з невеликим бонусом за залучення додаткових учасників понад мінімум.

Таким чином, середовище не є ні суто кооперативним, ні суто конкурентним. Воно реалізує зміну режимів поведінки, де учасники в одних ситуаціях оптимізують особисту вигоду, а в інших змушені об’єднувати зусилля. Саме це робить ACG зручним для дослідження динамічної адаптації політик й перевірки гіпотези у тому, що єдина модель може навчитися перемикатися між режимами без явного вибору окремої стратегії.

Іншою особливістю середовища є однорідність агентів. Всі п’ять агентів мають однаковий тип спостереження та дії, але використовують загальну архітектуру політики, що полегшує порівняння результатів і дозволяє зосередитися саме на ефекті контекстної адаптації. Локальне спостереження задається вікном 7×7 навколо агента.

Епізод триває до досягнення максимальної кількості кроків (300) або до настання умовного термінального стану. Імовірність появи нової монети або нової кризи на кожному кроці регулюється параметрами середовища.

3.3 Формалізація простору станів, дій та нагород

формально описується як децентралізований частково спостережуваний марківський процес прийняття рішень з адитивною винагородою, заданий кортежем

$$(N, S, A_i, O_i, T, Z_i, r_i, \gamma, b_0), \tag{3.1}$$

де $N = \{1, \ldots, 5\}$ – множина агентів;

S – простір глобальних станів, що включає координати агентів, координати та залишкові часи життя кожного активного ресурсу;

$A_i = \{0, 1, 2, 3, 4\}$ – простір дій агента (рух вгору, вниз, ліворуч, праворуч, бездіяльність);

O_i – простір локальних спостережень, тензор форми (7, 7, 4);

T – детермінована функція переходу за діями агентів, доповнена стохастичною появою нових ресурсів;

Z_i – функція спостереження, що дістає 7×7 локальне вікно навколо позиції і-го агента;

r_i – функція винагороди і-го агента;

$\gamma = 0.99$ – коефіцієнт дисконтування;

b_0 – початковий розподіл, у якому позиції агентів і ресурсів обираються випадково без перетинів.

Глобальний стан включає позиції агентів, координати особистих ресурсів, становище активних кризових об’єктів, поточний режим середовища. Простір дій є дискретним і включає базові переміщення агента сіткою: рух вгору, вниз, вліво, вправо, а також дія очікування або бездіяльності.

Нагорода у системі будується як композиція із двох частин. Перша частина відноситься до конкурентного режиму та пов’язана з індивідуальним збором монет. У цьому випадку агент отримує позитивну винагороду за особистий видобуток. Друга частина відноситься до кооперативної складової і виникає при вирішенні кризових ситуацій. Тут нагорода розподіляється між учасниками спільної дії, а додатковий бонус може нараховуватись за те, що у вирішенні кризи приймало участь більше мінімуму агентів.

Окрім локального спостереження, на кожному кроці середовище формується глобальний дескриптор режиму – вектор з п’яти нормалізованих компонентів: поточну кількість активних пожеж, поточну кількість активних монстрів, поточну кількість доступних монет на всій карті, середній рівень кооперації інших агентів за останні 10 кроків, кількість кроків з моменту появи останньої кризи. Цей вектор використовується централізованим критиком під час навчання.

3.4 Проєктування механізму динамічної кооперації та конкуренції

Основним елементом запропонованого середовища є механізм динамічного переходу між конкурентним та кооперативним режимами. Динаміка не задається явним перемикачем, а виникає з взаємодії таких механізмів: випадкової та незалежної появи ресурсів обох типів; обмеженого часу життя ресурсів; різного мінімального числа учасників для подолання різних типів криз; та лінійного бонусу за залучення додаткових учасників. Така комбінація призводить до того, що оптимальна стратегія агента залежить від поточного стану середовища: при достатній концентрації монет вигідним є їх збір, а при появі кризи з обмеженим часом життя більш доречним є переміщення до її локалізації навіть з втратою індивідуальної винагороди.

Для практичної реалізації такої механіки використовуються кілька внутрішніх індикаторів: кількість активних криз, відстань агентів до кризового об’єкта, агрегований показник кооперативної поведінки за останні кроки. Ці ознаки потім передаються у контекстний модуль, який формує embedding поточного режиму взаємодії.

Окремої уваги потребує метрика кооперації. Простий показник того, що агент знаходиться на клітині кризи був би грубим, оскільки не враховіє намірів агента наблизитися до кризи. У цій роботі було застосовано композитну метрику з трьох рівнів:

– 1.0 – агент знаходиться на клітині активної кризи;

– 0.5 – агент скоротив манхеттенську відстань до найближчої активної кризи на цьому кроці;

– 0 – інакше, або у разі відсутності активних криз.

Ця метрика обчислюється на кожному кроці окремо для кожного агента та зберігається в історії останніх 10 кроків. Її середнє значення для всіх агентів, окрім поточного, підставляється у глобальний дескриптор. Така трирівнева метрика розрізняє агентів, що активно прямують до кризи, а

отже виявляють кооперативну поведінку, від тих, що залишаються на іншому кінці карти. Це забезпечує більш інформативний контекст для критика, ніж бінарна метрика участі.

3.5 Архітектура агентів та контекстного модуля

Базова архітектура агента була побудована за принципом CTDE. На етапі навчання центральний критик отримує більш повну інформацію про стан середовища, ніж окремий агент, а на етапі виконання політика агента працює децентралізовано, використовуючи лише локальне спостереження. У сучасних роботах PPO та його багатоагентні варіанти показали високу стійкість та конкурентні результати у ряді популярних тестів, а CTDE залишається домінуючою схемою для cooperative MARL.

Meta-encoder було реалізовано як невелику нейромережну підмережу з декількома повнозв’язковими шарами. Її завдання – стиснути контекстні ознаки в компактну латентне представлення, придатне для подачі в критика.

Актор (рисунок 3.1) складається з трьох послідовних модулів:

– LocalObserver – згорткова мережа, що обробляє локальний тензор 7×7×4. Складається з двох згорткових шарів з GroupNorm-нормалізацією, яка стабільно працює при батч-розмірі 1, на відміну від BatchNorm. Завершальний повнозв’язний шар відображає вихід у 256-вимірне представлення;

– LSTM – одношаровий рекурентний шар з прихованим розміром 256, що накопичує історію локальних спостережень;

– голови актора – двошаровий перцептрон з 128 нейронами та виходом у 5 дій.

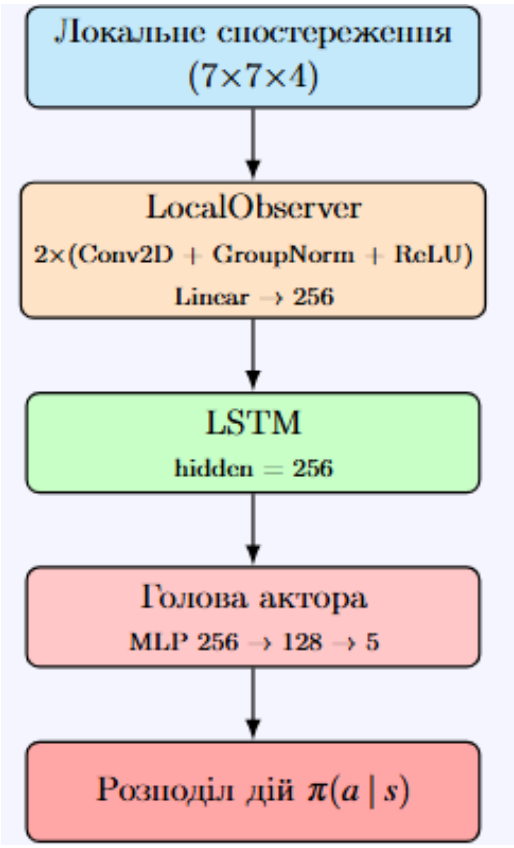

Рисунок 3.1 – Архітектура актора

Централізований критик (рисунок 3.2) задіюється тільки на етапі навчання і складається з:

– GlobalObserver – згорткова мережа над повним тензором стану 15×15×4 з адаптивним пулінгом до 5×5 та повнозв’язним шаром на 128 нейронів;

– MetaEncoder – повнозв’язна мережа з шарами 5, 128, 128, 64, з LayerNorm та tanh-активацією на виході;

– голови критика – конкатенація глобальних згорткових ознак (128), контекстного embedding (64) та one-hot вектору ідентифікатора агента (5); далі MLP 256, 128, 1.

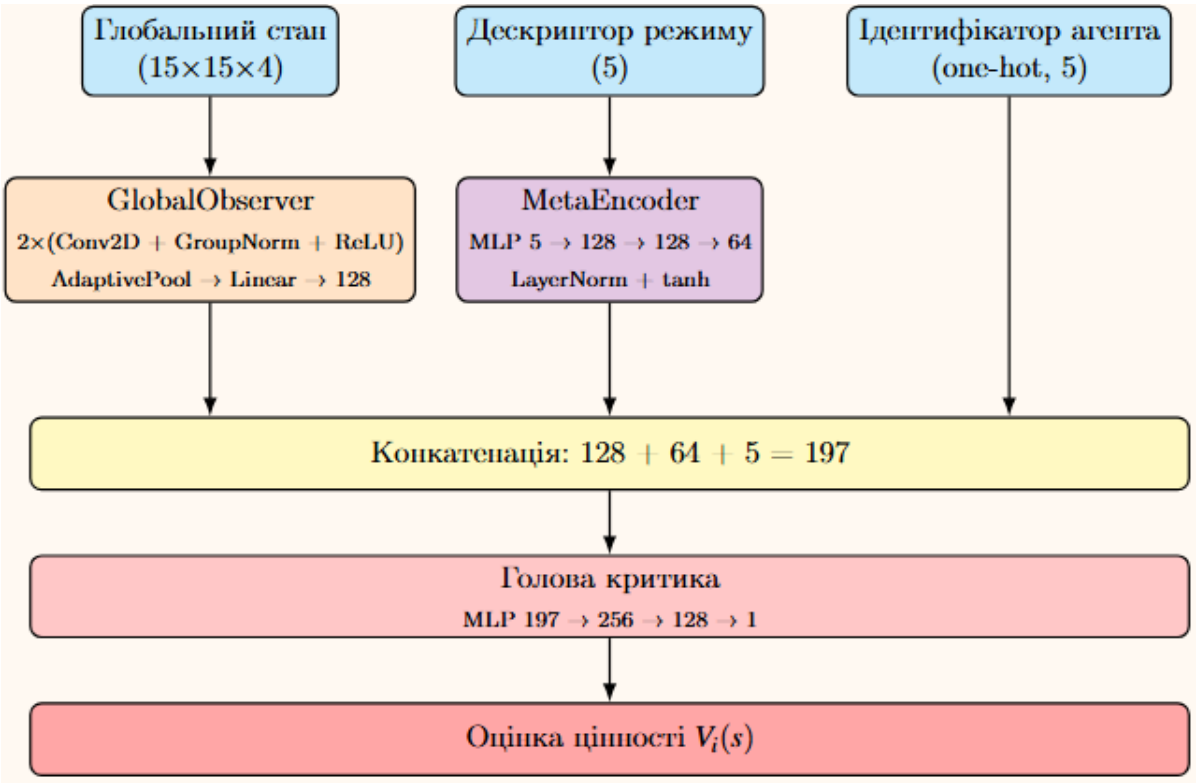

Рисунок 3.2 – Архітектура централізованого критика

Загальна кількість параметрів повної моделі складає приблизно 1.6 мільйона. Для абляційних експериментів передбачено три незалежні прапорці у конструкторі моделі: use_lstm (вмикає рекурентний шар або замінює його на повнозв’язний), use_meta_encoder (вмикає мета-енкодер у критику) та use_centralized_critic (перемикає критик між централізованим і децентралізованим режимом). Це дозволяє формувати п’ять конфігурацій (повна, базова, без LSTM, з децентралізованим критиком, з мета-енкодером без централізованого критика) у межах одного класу моделі без дублювання коду.

3.6 Реалізація алгоритму навчання

Базовим алгоритмом навчання є MAPPO, адаптований до архітектури з рекурентним актором та централізованим критиком. Цикл навчання

організований у фази збору траєкторій та оновлення параметрів. Послідовність дій на одній ітерації цикла:

– збір траєкторій (rollout). Впродовж 512 кроків середовища збираються переходи. Для кожного агента у момент дії використовується однокроковий прохід актора, який повертає логіти дій та оновлений прихований стан LSTM. Епізод, що завершився під час збору, зберігається як окрема траєкторія; епізод, обірваний на межі збору траєкторій, також зберігається з прапорцем неповноти;

– обчислення переваг (Generalized Advantage Estimation). Для кожної траєкторії обчислюється GAE з $\gamma = 0.99$ та $\lambda = 0.95$. Для повністю завершеного епізоду bootstrap-значення дорівнює нулю; для обірваного – обчислюється централізованим критиком на наступному стані;

– нормалізація переваг. Переваги нормалізуються по всьому rollout, що зменшує дисперсію градієнтів та стабілізує навчання;

– оновлення політики з послідовним BPTT. Для кожної траєкторії проводиться послідовне проходження актора через всю історію кроків з нульовим початковим прихованим станом LSTM. Це є основною відмінністю від спрощених реалізацій, де LSTM навчається лише на однокрокових градієнтах: послідовне проходження забезпечує коректне зворотнє поширення у часі, без чого рекурентний шар не може ефективно навчатися. Сурогатна цільова функція PPO використовує кліппінг $\varepsilon = 0.2$; коефіцієнт ентропійного регулярізатора – 0.01; коефіцієнт втрати критика – 0.5; обмеження норми градієнтів – 0.5;

– Оптимізатор – AdamW.

Один цикл навчання повторюється до досягнення заданої кількості кроків середовища у 4 мільйона кроків. У процесі навчання періодично зберігаються проміжні чекпоінти моделі та логуються основні метрики: середня нагорода за епізод, актор-втрата, критик-втрата, ентропія, KL-розбіжність, норма градієнту.

3.7 Програмна архітектура системи

Система складається з восьми основних модулів, кожен з яких відповідає за окрему підсистему:

– acg_env.py – реалізація середовища ACG, сумісного з PettingZoo Parallel API. Містить логіку появи ресурсів, обробки дій агентів, обчислення винагород, формування локальних спостережень та глобального стану, а також обчислення метрики кооперації. Повний вихідний код знаходиться у лістингу А.1;

– acg_models.py – модулі нейронних мереж: LocalObserver, GlobalObserver, MetaEncoder та клас ACGAgent, який поєднує актора та критика з підтримкою прапорців use_lstm, use_meta_encoder, use_centralized_critic для абляційних експериментів. Клас має два окремі публічні методи: actor_forward, який приймає тільки локальне спостереження та critic_forward, який приймає глобальний стан, контекстний вектор та ідентифікатор агента. Повний вихідний код знаходиться у лістингу А.2;

– acg_trainer.py – клас MAPPOTrainer, який реалізує цикл навчання, збір траєкторій, обчислення GAE та PPO-оновлення з послідовним BPTT. Окремий допоміжний клас Trajectory представляє одну неперервну ділянку досвіду. Повний вихідний код знаходиться у лістингу А.3;

– train.py – CLI-точка входу для запуску навчання за вказаною конфігурацією, з підтримкою параметрів --config, --seed, --total-steps, --no-wandb. Повний вихідний код знаходиться у лістингу А.4;

– training_configs.py –конфігураційний словник, що містить набір параметрів для повної моделі, основного опорної (baseline) та трьох абляційних варіантів. Повний вихідний код знаходиться у лістингу А.5;

– evaluate.py – модуль для оцінювання навченого агента з обчисленням всіх дослідницьких метрик. Використовує словник,

повернутий середовищем, для точного приписування винагород до джерел монети або кризи. Повний вихідний код знаходиться у лістингу А.6;

– run_ablation.py – скрипт для послідовного навчання та оцінювання усіх абляційних конфігурацій. Повний вихідний код знаходиться у лістингу А.7;

– evaluate_multi_seed.py – скрипт для мульти-сидового оцінювання з обчисленням t-тестів Велча на сумарній нагороді та рівнів кооперації, та збереженням результатів у JSON файл. Повний вихідний код знаходиться у лістингу А.8.

Така модульна організація відповідає принципу єдиної відповідальності: середовище, моделі, тренер, точка входу та оцінювач локалізовані у різних файлах і пов’язані через чітко визначені інтерфейси. Це забезпечує можливість заміни окремих компонентів, наприклад, тестування іншого алгоритму PPO, без переписування інших модулів.

3.8 Обґрунтування вибору інструментів та технологій

Реалізація системи виконана на мові Python 3.11. Вибір зумовлений наявністю розвиненої екосистеми бібліотек для машинного навчання та MARL, а також тим, що практично всі сучасні дослідницькі публікації у цій галузі супроводжуються кодом саме на Python.

Використання Gymnasium обґрунтовано тим, що це сучасний стандартний API для навчання з підкріпленням, що прийшов на заміну класичному Gym і зберіг при цьому просту і зрозумілу структуру середовища. У своїй основі Gymnasium надає API (інтерфейс програмування застосунків) для всіх середовищ навчання з підкріпленням для одного агента, базові операції середовища будуються навколо make(), reset(), step() і render() [14].

У центрі Gymnasium лежить Env – високорівневий клас Python, що представляє марковський процес прийняття рішень (MDP) з теорії навчання

з підкріпленням. Цей клас надає користувачам можливість запускати нові епізоди, виконувати дії та візуалізувати поточний стан агента. Поряд із Env також надаються Wrapper, які допомагають розширювати або змінювати середовище, зокрема спостереження агента, винагороди та виконувані дії.

PettingZoo обрано як універсальний API для мультиагентних середовищ. Це усуває потребу у власному ad-hoc інтерфейсі та забезпечує сумісність з усіма основними MARL-фреймворками. Сумісність з Parallel API дозволяє у разі потреби замінити власний тренер на готову реалізацію зі сторонніх бібліотек, наприклад RLlib або Stable-Baselines3.

Pygame було використано для візуалізації середовища під час налагодження та запису демонстраційних роликів. Бібліотека підтримує як інтерактивний так і програмний режим виведення, що зручно для запису відео без графічного виводу. Зовнішній вигляд розробленого середовища зображений на рисунку 3.3. Pygame – це набір модулів Python, призначених для розробки ігор. Це дозволяє створювати повнофункціональні ігри та мультимедійні програми мовою Python [15].

PyTorch обраний як основна бібліотека для побудови та навчання нейронних мереж. Перевагами PyTorch у контексті цієї роботи є динамічна побудова графа обчислень, що особливо зручно при роботі з рекурентними шарами та послідовним BPTT, зручний інтерфейс для роботи з тензорами на GPU, а також велика спільнота, що забезпечує швидке вирішення проблем при налагодженні. Якщо порівнювати з TensorFlow, PyTorch перевершує його на даний момент завдяки інтуїтивному, імперативному стилю програмування, близькому до Python, що прискорює розробку та налагодження. Він став стандартом у дослідженнях, забезпечує гнучкість та прозорість контролю, а також пропонує більш ефективну підтримку CUDA. А TensorFlow часто вибирають за надійність у великих промислових проектах.

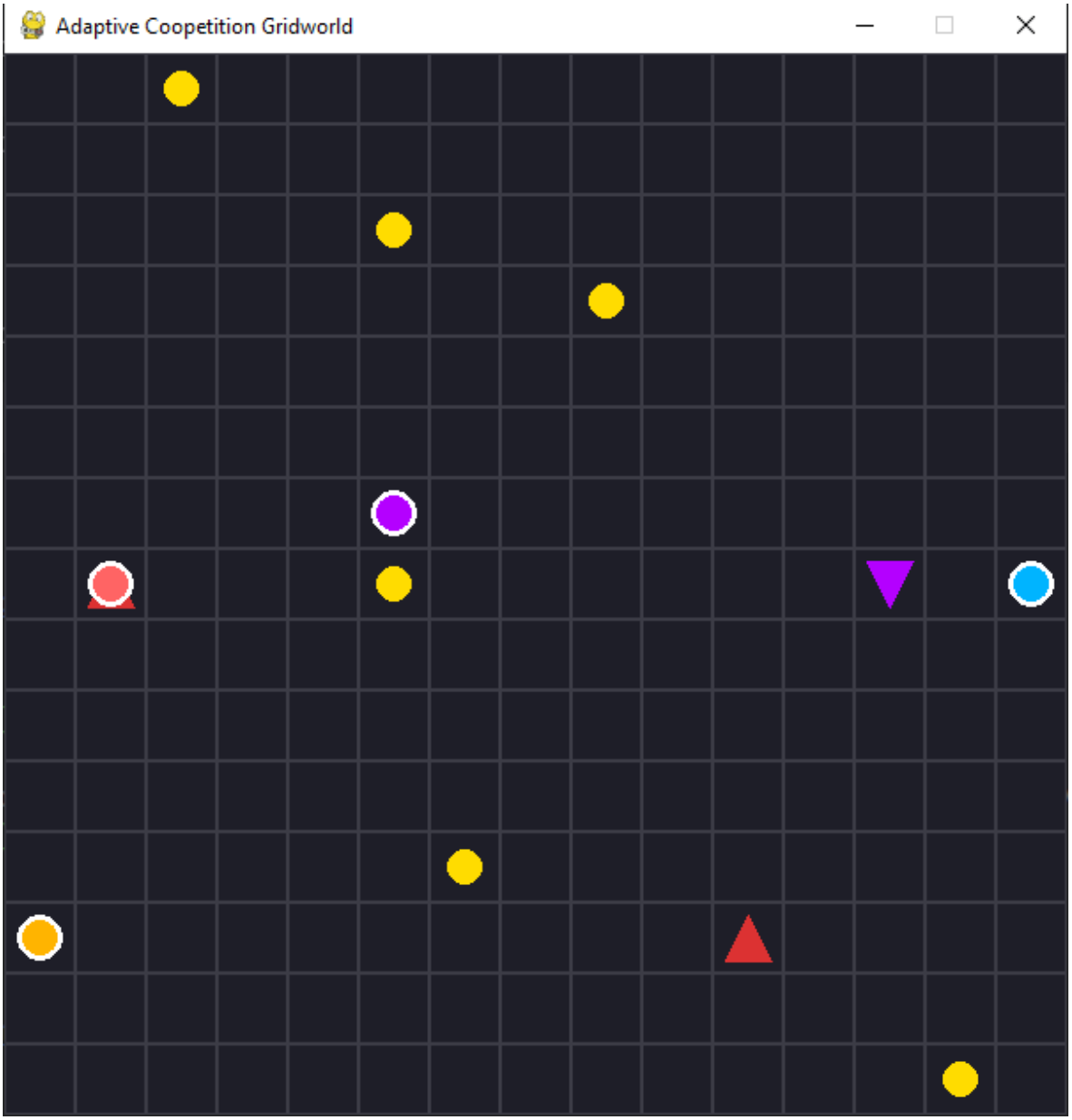

Рисунок 3.3 – Вигляд розробленого середовища

PyTorch – заснований на Python пакет для наукових обчислень, що виконує дві основні функції: заміну NumPy для можливостей графічних процесорів та інших прискорювачів, а також бібліотека автоматичного диференціювання корисна для реалізації нейронних мереж [16].

Weights & Biases, Wandb, обрано як інструмент логування експериментів через його підтримку порівняння різних запусків в одному проєкті, побудови графіків метрик у реальному часі та перегляду гіперпараметричних залежностей. Моделі W&B – це система обліку для

фахівців з машинного навчання, які хочуть організувати свої моделі, підвищити продуктивність та ефективність співпраці, а також масштабно впроваджувати машинне навчання у виробництво [17]. Альтернативно через прапорець --no-wandb логи можуть писатися лише у консоль, що було корисно для запусків у обмежених середовищах.

NumPy та SciPy використовуються для базових математичних операцій та статистичних тестів. T-тест Велча зі SciPy застосовується для порівняння конфігурацій моделей за середньою нагородою та рівнем кооперації, що дозволяє кількісно оцінити статистичну значимість спостережуваних різниць.

Навчання моделей виконувалось в блокнотах на Kaggle та у середовищі Lightning. Найпоширенішим типом є блокноти Jupyter. Блокноти Jupyter складаються з послідовності комірок, де кожна комірка форматована або у Markdown для написання тексту, або мовою програмування на для написання коду [18]. Kaggle дозволяє запускати сесію у фоновому режимі та навчати моделі на GPU.

Lightning це хмарна платформа для розробників та команд, які займаються штучним інтелектом, яка спрощує створення та розгортання високопродуктивних моделей, яка також надає безкоштовний доступ для навчання на деякий час [19].

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1 Цілі та гіпотези експериментального дослідження

Експериментальне дослідження має три основні цілі. По-перше, перевірити, що запропонована модель з мета-енкодером навчається стабільно і досягає прийнятного рівня продуктивності у середовищі ACG. По-друге, кількісно оцінити внесок мета-енкодеру до якості навченої політики порівняно з базовою моделлю без нього. По-третє, провести абляційний аналіз, щоб ізолювати внесок кожного компонента архітектури (мета-енкодер, рекурентний шар, централізований критик) і таким чином з’ясувати, які саме архітектурні рішення є визначальними для успіху моделі у динамічному coopetition-середовищі.

На основі теоретичного аналізу сформульовано наступні дослідницькі питання, що перевіряються експериментально:

– RQ1 – як впливає додавання мета-енкодеру у централізований критик MAPPO на якість навченої політики у середовищі ACG за сукупністю метрик (загальна нагорода, рівень кооперації, швидкість адаптації) порівняно з базовою моделлю без нього;

– RQ2 – як впливає рекурентний компонент (LSTM) у актора для адаптивної поведінки, оскільки агент має лише локальне спостереження і повинен накопичувати інформацію про історію взаємодії. Очікується, що видалення LSTM призводить до значного зниження як кооперативної, так і конкурентної складових нагороди;

– RQ3 – як впливає централізований критик з повним глобальним станом на навчання у порівнянні з децентралізованим IPPO-подібним критиком в умовах змішаних мотивацій ACG. Це перевіряється шляхом порівняння кривих навчання та фінальних метрик;

– RQ4 – чи мета-енкодер є взаємодоповнюючим до централізованого критика: його вилучення з повної моделі призводить до зниження результатів, але мета-енкодер у комбінації з децентралізованим критиком без повного стану також покращує результати порівняно з повністю децентралізованою моделлю.

4.2 План проведення експериментів

Експериментальний план побудований у форматі контрольованих абляцій, де кожен експеримент відрізняється від базового рівно одним компонентом, що дозволяє ізолювати внесок цього компонента. Усі експерименти виконуються у тому самому середовищі ACG з однаковими гіперпараметрами навчання, що відрізняє цю постановку від міжалгоритмічних порівнянь, де відмінності у результатах часто можуть пояснюватися відмінностями у налаштуваннях, а не у самих алгоритмах.

Кожна конфігурація навчалась з трьома незалежними випадковими зернами: 42, 56 та 10000 на 4 мільйонах кроків середовища. Це обмеження пов’язане з обчислювальним бюджетом, доступним для виконання експериментів. Підсумкові значення метрик подаються у форматі середнього ± стандартне відхилення. Оцінювання навчених моделей проводиться на 500 епізодах в детермінованому режимі.

Статистична значимість порівнянь обчислюється за t-критерієм Велча на рівні зерен. Це означає, що для кожної конфігурації обчислюється середнє значення метрики по 500 епізодах оцінювання, після чого ці три сид-середні, по три на конфігурацію, використовуються як спостереження у t-тесті. Сид-рівневий аналіз є методологічно коректним для multi-seed експериментів у RL, оскільки епізоди в межах одного навченого агента не є незалежними спостереженнями процесу навчання. Критерій обрано через його стійкість до неоднакових дисперсій у порівнюваних вибірках, що типово для нейронних мереж з різними архітектурами.

На етапі попередніх запусків було проведено калібрування коефіцієнта ентропійного регулярізатора. Початкове значення 0.01 призводило до передчасного зниження ентропії політики та стагнації середньої нагороди на рівні ~24. Збільшення коефіцієнта до 0.03 забезпечило стійке зростання нагороди вже з перших мільйонів кроків середовища. Це значення зафіксовано однаковим для всіх п’яти конфігурацій абляційного дослідження для коректності порівняння.

4.3 Базові моделі та сценарії порівняння

Було визначено п’ять конфігураційних моделей:

– full_model – повна модель з рекурентним актором, централізованим критиком та мета-енкодером. Це запропонована основна архітектура;

– baseline_no_meta – основний baseline з рекурентним актором, централізованим критиком без мета-енкодера. Дозволяє безпосередньо оцінити внесок мета-енкодеру та відповісти на RQ1;

– ablate_no_lstm – модель без LSTM, де актор використовує повнозв’язний шар замість рекурентного, інші компоненти збережені. Дозволяє оцінити внесок рекурентного компонента та відповісти на RQ2;

– ablate_decentralized_critic – IPPO-подібна модель з децентралізованим критиком, без мета-енкодера. Дозволяє оцінити внесок централізованого критика та відповісти на RQ3;

– ablate_meta_only – модель з мета-енкодером, але без децентралізованого критика. Дозволяє з’ясувати, чи є мета-енкодер самостійно цінним джерелом інформації, незалежно від доступу до повного стану та відповісти на RQ4.

Усі гіперпараметри: $lr = 5 \cdot 10^{-4}$, $\gamma = 0.99$, $\lambda = 0.95$, $\varepsilon = 0.2$, коефіцієнт ентропії 0.03, коефіцієнт критика 0.5, обмеження норми градієнта 0.5 однакові у всіх конфігураціях для коректного порівняння. Розмірності

прихованих шарів та архітектурних блоків також збережені; різниця між конфігураціями обмежена набором задіяних модулів.

4.4 Метрики оцінки ефективності

Для оцінювання якості агента використовувались метрики, що дозволяють розкрити різні аспекти поведінки – як кооперативної, так і конкурентної.

Середня сумарна нагорода за епізод (mean total reward per episode) – агрегована продуктивність агентів за всіма джерелами винагороди.

Середня нагорода від монет (mean coin reward) – компонент, що відображає конкурентну поведінку та ефективність агента у збиранні особистих ресурсів.

Середня нагорода від криз (mean crisis reward) – компонент, що відображає кооперативну поведінку та внесок агента у спільне подолання криз.

Рівень кооперації (cooperation rate) – частка кроків з активною кризою, на яких принаймні один агент знаходиться на клітині кризи. Це інтегральна оцінка готовності агентів до спільних дій.

Швидкість адаптації (adaptation speed) – середня кількість кроків від появи кризи до її успішного подолання. Менші значення вказують на швидшу реакцію агентів на зміну режиму.

Частка успішно подоланих криз (crisis resolution rate, %) – відношення кількості подоланих криз до загальної кількості тих, що з’явилися протягом епізоду.

Кількість зібраних монет за епізод (coin collection rate) – компонент, що відображає ефективність у конкурентній складовій.

Усі метрики обчислюються на основі полів словника, повернутого середовищем, зокрема coin_pickup, crisis_resolved_here, coin_reward, crisis_reward, що забезпечує точне розрізнення джерела винагороди та

виключає помилки приписування, які могли б виникнути при використанні порогових правил на сумарних нагородах.

4.5 Результати навчання та порівняння моделей

Кожна конфігурація навчалася з трьома незалежними випадковими зернами 42, 56, 10000. Для зерен виконано повні цикли навчання з кількістю кроків 4 млн.

Криві середньої нагороди за епізод при навчанні моделі з сидом 42 зображені на рисунку 4.1.

Рисунок 4.1 – Крива середньої нагороди за епізод при сиді 42

Криві середньої нагороди за епізод при навчанні моделі з сидом 56 зображені на рисунку 4.2.

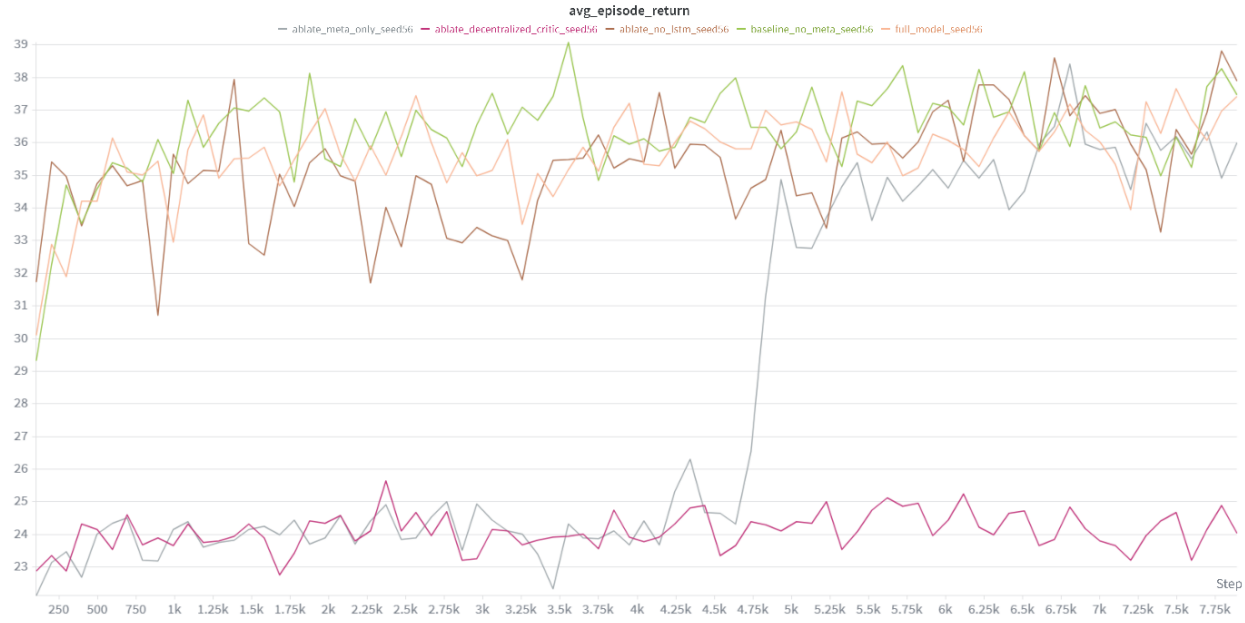

Рисунок 4.2 – Крива середньої нагороди за епізод при сиді 56

Криві середньої нагороди за епізод при навчанні моделі з сидом 10000 зображені на рисунку 4.3.

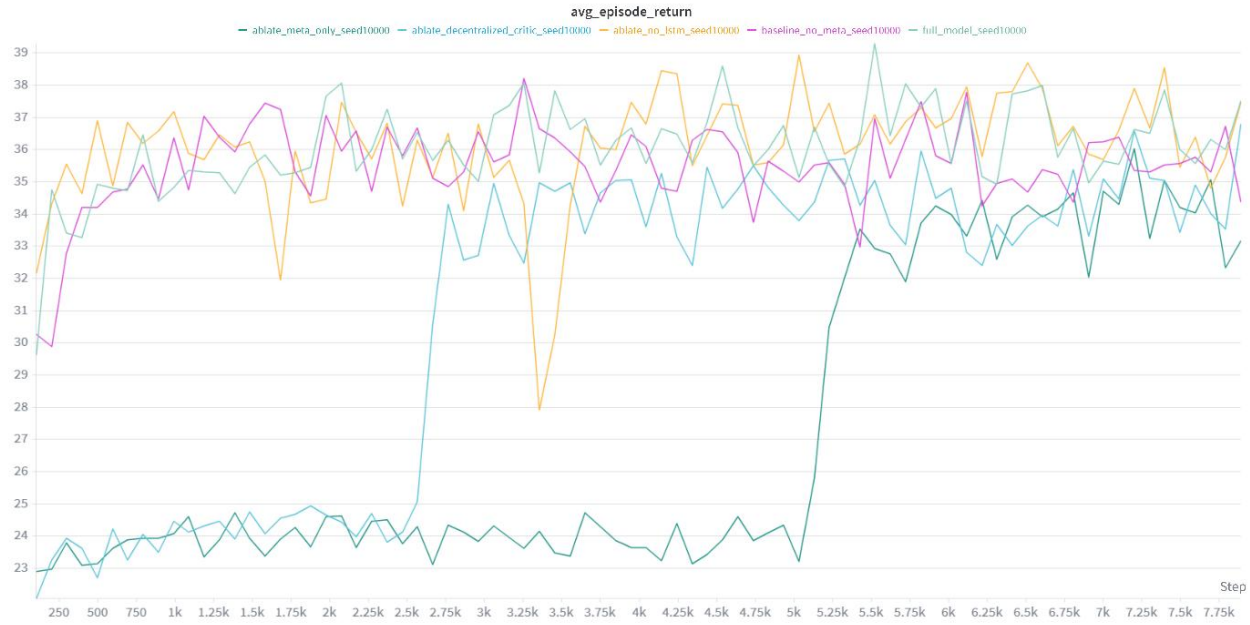

Рисунок 4.3 – Крива середньої нагороди за епізод при сиді 10000

Криві середньої нагороди за епізод, отримані з логів wandb, виявляють два чітко відмінні класи поведінки конфігурацій під час навчання. Конфігурації full_model та baseline_no_meta у всіх трьох сидах демонструють гладке монотонне зростання нагороди з ~31 на старті до ~37–38 у середині першого мільйона кроків середовища, після чого виходять на стійке плато і коливаються у вузькій смузі ~35–38 до кінця навчання. Жоден з прогонів цих двох конфігурацій не показав ознак стагнації, що дозволяє кваліфікувати їхню збіжність як стабільну та відтворювану.

Інші три конфігурації – ablate_no_lstm, ablate_decentralized_critic та ablate_meta_only – демонструють принципово іншу динаміку. У значній кількості прогонів вони тривалий час залишались на низькому рівні ~22–25, що відповідає стратегії лише збору монет, без участі у подоланні криз, а потім за короткий проміжок у декілька десятків тисяч кроків здійснюють різкий перехід до успішного режиму ~33–35. Така динаміка є характерним прикладом фазового переходу у задачах MARL з емерджентною кооперацією: оптимальна координаційна стратегія є локальним оптимумом, до якого політика виходить лише після випадкового відкриття успішної кооперативної послідовності. Час до фазового переходу істотно варіюється між сидами від ~3k оновлень до більш ніж 8k, а у одному з тестів (ablate_decentralized_critic, сид 56) фазовий перехід не відбувся протягом всього доступного бюджету навчання у 4 млн кроків. У іншому прогоні (ablate_meta_only, сид 10000) перехід відбувся на останніх 5k оновленнях навчання і не встиг закріпитися. Це означає, що ані рівень нагороди, ані сам факт досягнення кооперативної стратегії не є гарантованими для конфігурацій без повного набору контекстних механізмів у рамках практичного бюджета навчання.

У таблиці 4.1 наведено результати оцінювання усіх п’яти конфігурацій на 500 епізодах у детермінованому режимі. Для кожної метрики подано середнє значення ± стандартне відхилення по трьох сидах.

Таблиця 4.1 – Підсумкові метрики оцінювання

Конфігурація	Сумарна нагорода	Нагорода від монет	Нагорода від криз	Cooperation rate	Adaptation speed	Crisis resolution
full_model	136.15 ± 13.90	84.57 ± 10.19	51.59 ± 4.79	0.171 ± 0.036	8.14 ± 0.49	0.207 ± 0.022
baseline_no_meta	138.90 ± 18.83	87.55 ± 13.89	51.35 ± 5.23	0.171 ± 0.076	8.37 ± 0.68	0.205 ± 0.023
ablate_no_lstm	116.60 ± 13.02	61.28 ± 11.63	55.32 ± 2.73	0.085 ± 0.012	7.93 ± 0.40	0.192 ± 0.012
ablate_meta_only	118.50 ± 15.45	66.01 ± 11.50	52.49 ± 4.22	0.044 ± 0.036	7.69 ± 0.26	0.194 ± 0.023
ablate_decentralized_critic	106.51 ± 15.18	69.54 ± 7.58	36.96 ± 20.92	0.042 ± 0.007	6.92 ± 0.86	0.137 ± 0.078

З таблиці видно, що ключовою метрикою, яка найбільш чітко розрізняє конфігурації, є cooperation rate – частка кроків з активною кризою, на яких принаймні один агент знаходиться на клітині кризи. За цією метрикою конфігурації утворюють три яскраво виражені кластери: верхній ~0.17 включає обидві централізовані конфігурації з LSTM, проміжний ~0.085 містить конфігурацію без LSTM, а нижній ~0.04 – обидві конфігурації з децентралізованим критиком. Різниці в cooperation rate є істотнішими та стабільнішими, ніж різниці у сумарній нагороді, оскільки остання може компенсуватися різними стратегіями, наприклад, конфігурація без LSTM компенсує знижену кооперацію кращим збором монет. Це робить cooperation rate більш кращою діагностичною метрикою для досліджуваного класу задач.

Аналіз метрики crisis_reward у конфігурації ablate_decentralized_critic дає додаткове підтвердження виявленої проблеми стабільності збіжності. Сидові значення цієї метрики – 46.9, 12.9, 51.1 – показують аномально низький результат сиду 56, що відповідає його непереходу до

кооперативного режиму на кривій навчання. Це є наочним прикладом того, чому одно-сидові експерименти у MARL є ненадійними: різниця у поведінці окремих сидів може досягати порядку кратного. Високе стандартне відхилення цієї конфігурації є прямим наслідком цієї нестабільності.

4.6 Абляційний аналіз компонентів запропонованої моделі

Абляційний аналіз виконано шляхом парного порівняння кожної з чотирьох абляційних конфігурацій з базовою конфігурацією full_model. Для кожного парного порівняння застосовано двосторонній t-критерій Велча на сид-рівні N=3 на групу, що є методологічно коректним рівнем аналізу для multi-seed експериментів у RL. У таблиці 4.2 наведено результати t-тестів для чотирьох ключових метрик. Знак різниці (Δ) показує відносну зміну метрики порівняно з full_model; p-значення позначене зірочкою у разі статистичної значущості ($p < 0.05$).

Результати t-тестів дозволяють зробити кількісні висновки по кожному з поставлених дослідницьких питань.

Внесок мета-енкодера у full_model (full_model проти baseline_no_meta). Жодна з чотирьох ключових метрик не показує статистично значущої різниці між цими двома конфігураціями (всі $p > 0.6$, найменша різниця на cooperation rate складає лише 0.0003). Кооперативна, конкурентна та сумарна продуктивність обох конфігурацій є практично ідентичною. Висновок: введення мета-енкодера у централізованого критика, що вже має доступ до повного 15×15×4 стану середовища, не дає вимірного приросту якості навченої політики у досліджуваних умовах.

Внесок рекурентного шару (full_model проти ablate_no_lstm). Видалення LSTM призводить до зниження cooperation rate з 0.171 до 0.085, тобто на 50%, що є статистично значущим ($p = 0.043$). Зниження сумарної нагороди на 14.4% не досягає рівня значущості при N=3 ($p = 0.150$). Цікавим є те, що нагорода від криз залишається приблизно тією ж (+7.2%, $p = 0.321$),

тобто конфігурація без LSTM зрідка, але ефективно бере участь у подоланні криз, коли все ж опиняється на потрібній клітині. Це інтерпретується як те, що LSTM критичний для систематичного просторово-часового планування шляху до кризи, але разові кооперативні події відбуваються і без нього.

Таблиця 4.2 – Парні порівняння з full_model

Конфігурація	Метрика	Δ, %	t	p	Знач.
baseline_no_meta	Сумарна нагорода	+2.0	+0.203	0.850	
baseline_no_meta	Cooperation rate	-0.2	-0.005	0.966	
baseline_no_meta	Нагорода від криз	-0.5	-0.057	0.957	
baseline_no_meta	Crisis resolution rate	-1.1	-0.122	0.909	
ablate_no_lstm	Сумарна нагорода	-14.4	-1.779	0.150	*
ablate_no_lstm	Cooperation rate	-50.4	-3.918	0.043	
ablate_no_lstm	Нагорода від криз	+7.2	+1.174	0.321	
ablate_no_lstm	Crisis resolution rate	-7.2	-1.033	0.374	
ablate_decentralized_critic	Сумарна нагорода	-21.8	-2.495	0.068	
ablate_decentralized_critic	Cooperation rate	-75.2	-6.050	0.022	*
ablate_decentralized_critic	Нагорода від криз	-28.4	-1.181	0.349	
ablate_decentralized_critic	Crisis resolution rate	-33.9	-1.509	0.254	
ablate_meta_only	Сумарна нагорода	-13	-1.472	0.216	*
ablate_meta_only	Cooperation rate	-74.5	-4.293	0.013	
ablate_meta_only	Нагорода від криз	+1.8	+0.245	0.819	
ablate_meta_only	Crisis resolution rate	-6.4	-0.719	0.512	

Внесок централізованого критика (full_model проти ablate_decentralized_critic). Зниження cooperation rate з 0.171 до 0.042, тобто на 75%, є статистично значущим ($p = 0.022$). Сумарна нагорода знижується на 21.8% ($p = 0.068$, на межі значущості). Crisis resolution rate знижується на 33.9% ($p = 0.254$, не значуще при $N=3$, але з істотним розміром ефекту). Це найбільший за абсолютним розміром ефект серед усіх абляційних

порівнянь, що дозволяє кваліфікувати наявність централізованого критика як домінуючий архітектурний фактор у досліджуваному середовищі.

Чи компенсує мета-енкодер відсутність централізованого критика (ablate_meta_only проти ablate_decentralized_critic). За номінальними значеннями ablate_meta_only має на 11.3% вищу сумарну нагороду (118.5 проти 106.5; t = 0.96, p = 0.39) та на 41.7% вищу crisis resolution rate (0.194 проти 0.137; t = 1.22, p = 0.33), однак ці різниці не досягають статистичної значущості через велику варіативність окремих сидів. Зокрема, у сиді 56 конфігурація ablate_decentralized_critic не досягла кооперативної збіжності протягом 4 млн кроків (crisis_reward = 12.9 при середньому ~50 у інших сидах), що штовхає її середнє значення вниз і збільшує стандартне відхилення. У сумі дані вказують на тенденцію до підвищеної стабільності збіжності при додаванні мета-енкодера до децентралізованого критика, але кількості сидів N=3 недостатньо для статистично значущого висновку. Це є самостійним методологічним результатом: при N=3 ефекти середнього розміру у MARL можуть залишатися статистично невиявленими через високу міжсидову варіативність.

4.7 Аналіз отриманих результатів

Проведене систематичне абляційне дослідження дозволяє уточнити поточне уявлення про відносний внесок архітектурних компонентів у мультиагентному навчанні з підкріпленням для середовищ з динамічною coopetition. Основний висновок роботи полягає у тому, що в умовах змішаних мотивацій координаційна поведінка агентів визначається передусім якістю оцінки переваги централізованим критиком, а не наявністю явних контекстних модулів у архітектурі. Цей висновок випливає з трьох незалежних спостережень у експериментальних даних, що взаємно підсилюють один одного.

По-перше, перехід від централізованого до децентралізованого критика призводить до найбільшого за абсолютним розміром ефекту серед усіх досліджених абляцій – зниження cooperation rate на 75% ($p = 0.022$) та сумарної нагороди на 22% ($p = 0.068$). Жодна інша архітектурна зміна не дає такого ефекту. По-друге, додавання мета-енкодера у конфігурацію з вже наявним централізованим критиком – порівняння full_model проти baseline_no_meta – не дає вимірної користі за жодною з метрик (всі $p > 0.6$). По-третє, видалення рекурентного шару при збереженні централізованого критика хоча і знижує cooperation rate на 50%, але дає істотно менший ефект, ніж видалення централізованого критика, і не змінює середнього рівня успішно подоланих криз.

Сукупність цих результатів вказує на конкретну ієрархію впливу архітектурних компонентів у досліджуваному середовищі: централізований критик > рекурентна пам’ять > мета-енкодер. Останній у цій ієрархії виявляється функціонально надлишковим за умов наявності інформаційно багатшого критика.

Основне пояснення відсутності внеску мета-енкодера у full_model полягає в інформаційній надлишковості. Централізований критик у досліджуваній архітектурі має доступ до повного тензорного представлення стану середовища розміром 15×15×4, який через два згорткових шари може самостійно діставати ті ж агреговані ознаки: кількість активних криз кожного типу, просторовий розподіл агентів, рівень концентрації навколо кризових клітин, які компактно подає п’ятивимірний дескриптор режиму через мета-енкодер. Отже, додатковий канал інформації не приносить нової інформації – критик уже здатний обчислити необхідні агрегати з первинного стану. Тобто, явне внесення ознак, які мережа здатна самостійно вивести зі своїх входів, не дає приросту якості при достатній виразності архітектури.

Як обмеження цієї інтерпретації варто зазначити, що висновок про надлишковість мета-енкодера зроблено в умовах середовища ACG з повним 15×15×4 представленням стану для критика. У середовищах з реально

частково спостережуваним критиком, наприклад, коли критик також має лише локальне сенсорне вікно або зашумлене глобальне представлення, ситуація може бути іншою – саме на це вказує тенденція, спостережена у порівнянні ablate_meta_only проти ablate_decentralized_critic. Тому отриманий висновок слід розуміти як те, що мета-енкодер є надлишковим за умов інформаційно повного централізованого критика, а не як універсальне заперечення корисності контекстних модулів у MARL.

Аналіз кривих навчання виявив істотну якісну різницю між двома класами конфігурацій. Конфігурації з повним набором з LSTM та централізованим критиком, тобто full_model та baseline_no_meta, демонструють стабільну гладку збіжність у всіх трьох сидах. Інші три конфігурації показують характерну динаміку пізнього фазового переходу: тривале плато на рівні ~22–25 з подальшим різким стрибком до ~33–35. Час до цього стрибка істотно варіюється між сидами, а в одному з тестів перехід не відбувся протягом усіх 4 млн кроків навчання.

Ця унікальність має важливі практичні наслідки. По-перше, надійність збіжності є архітектурною властивістю, що відрізняється від асимптотичної якості: дві конфігурації з близькими середніми значеннями метрик можуть мати кардинально різну ймовірність успішного навчання за фіксованого бюджету. По-друге, у MARL-літературі мова найчастіше йде про підсумкові показники продуктивності, тоді як надійність збіжності може бути не менш важливим практичним критерієм при обмежених обчислювальних ресурсах. По-третє, наявність повного набору контекстних механізмів є тим, що гарантує стабільну збіжність, навіть якщо асимптотична нагорода у деяких сидах для абляційних конфігурацій виявляється близькою.

Проведене дослідження дозволяє зробити декілька висновків про експериментальну роботу у MARL. По-перше, мультисід підхід є обов’язковим: висновки, отримані на одному сиді, можуть бути непредставницькими через велику міжсидову варіативність. Це

безпосередньо проілюстровано прикладом ablate_decentralized_critic, де один з трьох сидів істотно відрізняється від інших і визначає форму середнього. По-друге, при N=3 ефекти середнього розміру, наприклад, +11% по сумарній нагороді у ablate_meta_only проти ablate_decentralized_critic, можуть залишатися статистично невиявленими через високу варіативність, що накладає обмеження на кількість висновків, які можна впевнено зробити з невеликого числа прогонів. По-третє, поведінкові метрики кооперації, зокрема cooperation rate, виявилися істотно більш діагностичними для досліджуваного класу задач, ніж сумарна нагорода: різниці між конфігураціями на cooperation rate є виразнішими та статистично значущими там, де відповідні різниці на сумарній нагороді залишаються невиявленими. Це підкреслює важливість використання задачно-специфічних поведінкових метрик при оцінюванні MARL-алгоритмів у середовищах з вираженою координаційною компонентою.

До основних обмежень роботи належать: обмежений обчислювальний бюджет в 4 млн кроків, що може бути недостатньо для виявлення асимптотичної поведінки конфігурацій з повільною збіжністю; невелика кількість сидів, що обмежує статистичну силу t-тестів; одне досліджуване середовище ACG, що не дозволяє узагальнити висновки на інші класи coopetition-задач. До перспективних напрямків подальших досліджень належать розширення multi-seed експериментів до N=5–10 з відповідною статистичною обробкою, тестування досліджених архітектурних компонентів у середовищах з обмеженою спостережуваністю критика, систематичне порівняння з альтернативними MARL-алгоритмами (QMIX, MADDPG, COMA) у середовищі ACG; дослідження впливу більш складних режимних структур.

ВИСНОВКИ

Отже, під час виконання кваліфікаційної роботи було розроблено та досліджено модель адаптивного мультиагентного навчання з підкріпленням, проведено дослідження впливу архітектурних компонентів агента на ефективність навчання у частково спостережуваному середовищі.

Спочатку було проаналізовано предметну область мультиагентних систем, їх класифікацію за характером взаємодії, ступенем централізації управління та спостережуваності середовища. Розглянуто особливості навчання з підкріпленням у мультиагентному середовищі, основні труднощі та існуючі підходи до моделювання кооперативної, конкурентної і змішаної поведінки агентів. Виконано огляд сучасних алгоритмів MARL, методів врахування контексту та адаптивної поведінки, а також аналіз стандартних середовищ.

Далі було викладено теоретичні основи розробки адаптивної мультиагентної системи. Обгрунтовано вибір парадигми CTDE як найбільш підходящої для задач зі змішаним режимом взаємодії. Детально розглянуто метод MAPPO як базовий алгоритм навчання. Запропоновано архітектурне рішення з трьома незалежно вмикуваними архітектурними компонентами – рекурентним шаром у акторі, мета-енкодером у централізованого критика та централізованим критиком як таким.

Розроблено та реалізовано програмну систему Adaptive Coopetition Gridworld – оригінальне ігрове середовище з 5 агентами на 15×15 сітці, у якому одночасно присутні конкурентні та кооперативні ресурси: монети та пожежі і монстр. Запропоновано та реалізовано композитну метрику кооперації, яка враховує не лише факт участі, а й намір агента наблизитися до кризи.

Проведено мульти-сидове абляційне дослідження п’яти конфігурацій з трьома незалежними випадковими зернами. Виявлено емпіричну ієрархію впливу архітектурних компонентів у досліджуваному середовищі, де

централізований критик є домінуючим фактором координаційної поведінки, рекурентний шар – вторинним за впливом, а мета-енкодер – статистично не значущим у комбінації з централізованим критиком.

Окремою емпіричною знахідкою є виявлення феномена пізньої фазової збіжності у конфігураціях без повного набору контекстних механізмів: вони можуть досягати близьких асимптотичних значень нагороди, але роблять це через раптовий стрибок після тривалого плато на рівні випадкової поведінки. Час до цього стрибка істотно варіюється між сидами, а в одному з трьох тестів конфігурації ablate_decentralized_critic фазовий перехід не відбувся протягом 4 млн кроків навчання.

Робота була апробована на конференції [20], що підтверджує її актуальність, практичну значущість та відповідність сучасному рівню розвитку наукових досліджень у галузі штучного інтелекту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Gronauer S., Diepold K. Multi-agent reinforcement learning: A Comprehensive Survey. *Multiagent Systems*. 2021. URL: https://doi.org/10.1007/s10462-021-09996-w (date of access: 27.01.2026).

2. Albrecht S. V., Christianos F. Multi-agent reinforcement learning: foundations and modern approaches. *Cambridge: MIT Press*, 2024. 366p.

3. A conceptual framework for performance evaluation and classification of multi-agent hybrid game problems based on reinforcement learning: impact factors and evaluation metrics / L. Zhang et al. *Urban and Building Science*. 2025. Т. 1, № 1. URL: https://doi.org/10.53941/ubs.2025.100004 (date of access: 02.02.2026).

4. Brandenburger A. Co-opetition. *New York : Doubleday*, 1996. 290 с.

5. Multi-agent actor-critic for mixed cooperative-competitive environments / Lowe R. et al. *Machine Learning*. 2017. URL: https://doi.org/10.48550/arXiv.1706.02275 (date of access: 02.02.2026).

6. Proximal policy optimization algorithms / Schulman J. et al. *Machine Learning*. 2017. URL: https://doi.org/10.48550/arXiv.1707.06347 (date of access: звернення 25.02.2026).

7. PettingZoo: gym for multi-agent reinforcement learning / Terry J. K. et al. Machine Learning. 2020. URL: https://doi.org/10.48550/arXiv.2009.14471 (date of access: 28.01.02026).

8. The starcraft multi-agent challenge / Samvelyan M. et al. *Machine Learning*. 2019. URL: https://doi.org/10.48550/arXiv.1902.04043 (date of access: 13.02.2026).

9. The overcooked generalisation challenge: evaluating cooperation with novel partners in unknown environments using unsupervised environment design / Ruhdorfer C. et al. *Machine Learning*. 2024. URL: https://doi.org/10.48550/arXiv.2406.17949 (date of access: 14.02.2026).

10. Becker T., Sunberg Z. Bridging the gap between partially observable stochastic games and sparse POMDP methods. *Computer Science and Game Theory*. 2024. URL: https://doi.org/10.48550/arXiv.2405.18703 (date of access 06.02.2026).

11. A survey of progress on cooperative multi-agent reinforcement learning in open environment / Yuan L. et al. *Multiagent Systems*. 2023. URL: https://doi.org/10.48550/arXiv.2312.01058 (date of access 08.02.2026).

12. Amato C. An introduction to centralized training for decentralized execution in cooperative multi-agent reinforcement learning. *Machine Learning*. 2024. URL: https://doi.org/10.48550/arXiv.2409.03052 (date of access 14.02.2026).

13. The surprising effectiveness of ppo in cooperative, multi-agent games / Yu C. et al. Machine Learning. 2021. URL: https://doi.org/10.48550/arXiv.2103.01955 (date of access 15.02.2026).

14. Basic usage. *Gymnasium Documentation*. URL: https://gymnasium.farama.org/introduction/basic_usage/ (date of access: 20.04.2026).

15. About – pygame wiki. *pygame news*. URL: https://www.pygame.org/wiki/about (date of access: 20.04.2026).

16. Deep learning with pytorch. *PyTorch*. URL: https://docs.pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html (date of access: 25.04.2026).

17. W&B models. *Wight & Biases Docs*. URL: https://docs.wandb.ai/models (date of access: 28.05.2026).

18. How to use kaggle. *Kaggle*. URL: https://www.kaggle.com/docs/notebooks (date of access: 02.05.2026).

19. Lightning documentation. *Lightning*. URL: https://lightning.ai/docs/overview/getting-started (date of access: 02.05.2026).

20. Падалкін К.Д., наук. керівник Шевченко О.Ю. Адаптивна мультиагентна система з динамічним перемиканням стратегій у змагально-кооперативному ігровому середовищі. *30-ий Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті»*. Харків: ХНУРЕ, 22–24 квітня 2026. С. 81–83.