МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
(повне найменування факультету)

Кафедра «Системний аналіз та обчислювальна математика»
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістра
(ступінь вищої освіти)

на тему РОЗРОБКА СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ ОПТИМІЗАЦІЇ РОБОЧИХ ГРУП
(назва теми)

Виконав(ла): студент(ка) 2м курсу, групи КНТз-813м

Спеціальності 124 – Системний аналіз
(код і найменування спеціальності)

Освітня програма (спеціалізація)
«Інтелектуальні технології та прийняття рішень в складних системах»

ВОВК О.О.
(ПРІЗВИЩЕ та ініціали)

Керівник РЯБЕНКО А.Є.
(ПРІЗВИЩЕ та ініціали)

Рецензент ЛОЗОВІСЬКА Л.І.
(ПРІЗВИЩЕ та ініціали)

2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
Кафедра «Системний аналіз та обчислювальна математика»
Ступінь вищої освіти магістр
Спеціальність 124-Системний аналіз
(код і найменування)
Освітня програма (спеціалізація) «Інтелектуальні технології та прийняття рішень в складних системах»
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ
В. о. завідувача кафедри
Еліна ТЕРЕЩЕНКО
«20» січня 2025 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТА
ВОВК Олени Олександрівни
(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) РОЗРОБКА СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ ОПТИМІЗАЦІЇ РОБОЧИХ ГРУП
керівник проєкту (роботи) к.ф.-м.н., доц. РЯБЕНКО Антон Євгенович
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)
затверджені наказом закладу вищої освіти від «20» листопада 2024 року № 480
2. Строк подання студентом проєкту (роботи) «20» січня 2025 року
3. Вихідні дані до проєкту (роботи)
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) пошук та аналіз інформації, завантаження бази даних, аналіз фільтрів та побудова графіків емпіричних функцій, заповнення таблиці статистичними даними.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
6. Консультанти розділів проєкту

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1	РЯБЕНКО А.Є., к.ф.-м.н., доц.	03 жовтня 2024	20 січня 2025
2	РЯБЕНКО А.Є., к.ф.-м.н., доц.	03 жовтня 2024	20 січня 2025
3	РЯБЕНКО А.Є., к.ф.-м.н., доц.	03 жовтня 2024	20 січня 2025
Нормоконтроль	ШИРОКОРАД Д.В., к.ф.-м.н., доц.	20 січня 2025	20 січня 2025

7. Дата видачі завдання « 03 » жовтня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	1-2 тиждень	Розділ 1
3	Проєктування СППР	3 тиждень	Розділ 2
4	Вибір програмних засобів для розробки	3 тиждень	Розділ 2
5	Програмна реалізація СППР	3-6 тижні	Розділ 2
6	Проведення експериментів	7 тиждень	Розділ 3
7	Оформлення пояснювальної записки та додатків	8 тиждень	Додатки
8	Нормоконтроль та рецензування	9 тиждень	
9	Захист дипломної роботи	11 тиждень	

Студент(ка) _____ Олена ВОВК
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи) _____ Антон РЯБЕНКО
(підпис) (Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Дипломна робота: 50 с., 8 рис., 2 табл., 2 дод., 18 джерел.

Об’єкт дослідження – ефективність командної роботи в робочій групі.

Предмет дослідження – підвищення рівня ефективності робочої групи за рахунок оптимального підбору учасників, що демонструють найбільший рівень продуктивності у співпраці.

Мета роботи – розробка системи підтримки прийняття рішень для оптимізації робочих груп.

Методи роботи – методи теорії графів, багатокритеріальної оптимізації, комбінаторики та розробки програмних продуктів.

В дипломній роботі розроблено систему підтримки прийняття рішень для оптимізації робочих груп за рахунок ефективного підбору учасників, що демонструють найбільший рівень продуктивності у співпраці.

РОБОЧА ГРУПА, КОМАНДНА РОБОТА, БАГАТОКРИТЕРІАЛЬНА ОПТИМІЗАЦІЯ, PYTHON, DJANGO

ЗМІСТ

Завдання……...2
Реферат……4
Вступ.....7
1 ТЕОРЕТИЧНІ ПІДҐРУНТЯ ПІДВИЩЕННЯ ОПТИМІЗАЦІЇ РОБОЧОЇ ГРУПИ……...……..10
1.1 Актуальність розробки методів підвищення ефективності робочої групи……...10
1.2 Постановка задачі оптимального підбору учасників робочої групи, що демонструють найбільший рівень продуктивності у співпраці……12
1.3 Математична модель оптимального підбору учасників робочої групи, що демонструють найбільший рівень продуктивності у співпраці …..…….……13
1.4 Найімовірніше число настання події за схемою Бернуллі…..……...……17
1.5 Висновки розділу 1…..…….....……18
2 2 СИСТЕМА ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ ОПТИМІЗАЦІЇ РОБОЧИХ ГРУП…….....……19
2.1 Функціональна та структурна схеми системи підтримки прийняття рішень для оптимізації робочих груп ...……...19
2.2 Алгоритми роботи блоків СППР …….21
2.3 Програмна реалізація СППР ……..21
2.4 Висновки розділу 2……...……...31
3 МОДЕЛЮВАННЯ СЕРІЇ ЗМАГАНЬ ТА РОБОТИ СППР ……32
3.1 Визначення кількість змагань א ... ...……..……32
3.2 Числовий експеримент по генерації команд ……...33
3.3 Розв’язання двокритеріальної задачі (1.1)-(1.3) ……….35
3.4 Висновки розділу 3……...……...36

Висновок.....38
Перелік посилань……...……39
Додаток А. Код Python (файл models.py)……...…… 42
Додаток Б. Код Python (файл views.py)…….45

Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian

ВСТУП

Робоча група — це об'єднання людей, які мають спільну мету та працюють разом для досягнення визначеного результату. Члени робочої групи можуть бути фахівцями в різних областях, і їхня діяльність спрямована на виконання конкретних завдань або вирішення певних проблем у межах організації чи проєкту. Члени групи працюють над досягненням певної спільної мети або вирішенням конкретної проблеми. Робочі групи зазвичай вимагають активної співпраці та взаємодії між учасниками для досягнення результату. У межах робочої групи члени можуть виконувати різні ролі, залежно від їхніх кваліфікацій і завдань. Робочі групи можуть мати чітко визначені терміни для виконання завдань або бути створеними для вирішення постійних питань. Ефективність командної роботи в робочих групах оцінюються за результатами їхньої діяльності, ефективністю взаємодії та здатністю досягати поставлених цілей.

Ефективність командної роботи в робочих групах є одним із пріоритетних напрямків розвитку в багатьох сферах — від бізнесу до наукових досліджень. Це особливо важливо в умовах швидко змінюваного світу, де складні задачі часто вимагають міждисциплінарних підходів і співпраці між людьми з різними вміннями та досвідом. У командній роботі часто виникають нові ідеї, оскільки різноманітність думок та підходів стимулює креативність. Робота в команді дає можливість знаходити інноваційні рішення завдяки спільним зусиллям, де кожен учасник вносить свій унікальний вклад.

Командна робота стимулює розвиток комунікативних навичок, таких як уміння слухати, чітко висловлювати свої думки, надавати і приймати зворотний зв'язок. Це важливий аспект для успішної роботи, оскільки погана комунікація може призвести до непорозумінь і зниження результативності.

У командній роботі кожен член може брати на себе роль лідера в певних ситуаціях, що сприяє розвитку лідерських навичок. Це також

допомагає створити більш рівноправну атмосферу в команді, де кожен може проявити ініціативу.

Робочі групи, які ефективно працюють разом, здатні швидше адаптуватися до змін. Коли члени робочої групи діляться знаннями і підтримують один одного, це дозволяє команді більш ефективно реагувати на зовнішні виклики, зберігаючи стійкість і стабільність у роботі.

Коли люди працюють разом, вони можуть підтримувати один одного, що підвищує мотивацію. Спільне досягнення цілей приносить більше задоволення і стимулює до подальших досягнень.

У робочих групах можна ефективно розподіляти відповідальність за різні аспекти проєкту, що дозволяє зменшити навантаження на кожного окремого члена та збільшити загальну ефективність роботи.

Таким чином, сприяння розвитку ефективності командної роботи в робочих групах є важливим кроком для досягнення успіху в будь-якій сфері.

Об’єктом дослідження є ефективність командної роботи в робочій групі. Предметом дослідження є підвищення рівня продуктивності робочої групи за рахунок ефективного підбору учасників, що демонструють найбільший рівень продуктивності у співпраці. Метою роботи є розробка системи підтримки прийняття рішень для оптимізації робочих груп.

Задачі дослідження:

а) аналіз факторів підвищення рівня продуктивності робочої групи за рахунок ефективного підбору учасників, що демонструють найбільший рівень продуктивності у співпраці;

б) проєктування системи підтримки прийняття рішень для оптимізації робочих груп;

в) програмна реалізація СППР;

г) моделювання процесу оптимізації робочих груп.

Актуальність роботи зумовлена необхідністю підвищення ефективності створення робочих груп в умовах, де складні задачі вимагають

міждисциплінарних підходів і співпраці між людьми з різними вміннями та досвідом.

Результати дослідження можуть бути корисними при формуванні ефективних робочих груп за можливості попереднього проведення серії змагань.

1 ТЕОРЕТИЧНІ ПІДҐРУНТЯ ПІДВИЩЕННЯ ОПТИМІЗАЦІЇ РОБОЧОЇ ГРУПИ

1.1 Актуальність розробки методів підвищення ефективності робочої групи

Підвищення ефективності робочих груп та команд є ключовим завданням для сучасних організацій в умовах швидких змін та глобальних економічних викликів. Сьогодні існує підхід, згідно з яким команди розглядаються як складні динамічні системи, що функціонують у певному контексті, розвиваються через взаємодію учасників з часом, еволюціонують та адаптуються до вимог ситуації [1]. Члени команди об’єднані спільною метою та діяльністю, мають спільні цілі, цінності та взаємодоповнювальні навички, беруть відповідальність за результати та здатні виконувати різні ролі в команді.

Оцінка ефективності командної роботи робочих груп здійснюється через показники досягнення поставлених цілей у визначений час, якості виконаної роботи, ефективності використання ресурсів, а також здатності утримувати або розширювати свою «нішу» в діяльності. Моніторинг конкурентоспроможності робочої групи здійснюється через різноманітні групи показників, які включають виробничо-економічні показники, показники ринкової стійкості та показники психологічної стабільності. Моніторинг перших двох груп показників є важливим для забезпечення успішного розвитку команди. Графік їхнього руху за часом є основною характеристикою системного моніторингу, що дозволяє відслідковувати та контролювати зміни. Періодичність моніторингу визначається для кожного показника окремо, від одного місяця до півроку, в залежності від специфіки діяльності команди. Показники перших двох груп залежать від сфери професійної діяльності команди, а моніторинг третьої групи пов’язаний із психологічним супроводом, що суттєво підвищує ефективність роботи команди загалом. Методи та

інструменти моніторингу мають забезпечувати об’єктивність, статистичну значимість, порівнювальність оцінок, багатокритеріальність інтегральних оцінок, мінімізацію критеріїв, використання додаткової інформації та організацію вибіркових досліджень.

З 1980-х років кількість досліджень, що стосуються методів підвищення ефективності командної роботи в різних сферах діяльності, є стабільно високою [1–15]. Дослідження охоплюють командну роботу в корпораціях [2], медицині [3], промисловості [4], спорті [5], творчих колективах, інноваційній діяльності, освіті [6, 7], сільському господарстві [8].

Огляд 50 років досліджень, проведений в роботі [1], показує, що команда є динамічною системою, яка змінюється та адаптується під впливом зовнішнього середовища та взаємодії її членів. Це підкреслює важливість формалізації задач, вибору критеріїв якості та розробки методів оцінки.

Серед ключових показників, що вивчаються, виділяють командну довіру [9], соціальну згуртованість [10], мотивацію, рівень проактивної участі в досягненні результатів розвитку, потребу в знаннях та технічній підтримці [2], а також collective intelligence (CI) [11].

В роботі [9] здійснено ретроспективний аналіз експериментів з оцінки командної довіри, визначено переваги і недоліки застосованих підходів. Автори роблять висновок, що оцінка довіри має враховувати її багаторівневий характер, динаміку розвитку протягом життя команди та нові способи вимірювання довіри без втручання в особисті сфери учасників.

У роботі [12] доведено, що командні змагання сприяють підвищенню ефективності роботи команди.

В роботі [3] зроблено аналіз наукових досліджень щодо використання гейміфікація для мотивації співпраці та про впливи гейміфікації в контексті співпраці. Виділено 11 пунктів порядку денного щодо тематики, теоретичні та методологічні шляхів майбутніх досліджень гейміфікації кооперативної діяльності. Проводяться багато досліджень щодо потенційних взаємозв’язків між атрибутами гри та командною поведінкою. Ці пропоновані зв’язки в

кінцевому підсумку призначені для того, щоб розкрити спосіб, у який можна використовувати ігрове навчання для сприяння ефективній командній роботі [4].

Важливим напрямком досліджень є побудова математичних моделей та підбір методів побудови ефективних команд та оцінки продуктивності команди. Наприклад, у роботі [13] запропоновано використання Disclosure games для покращення комунікації через зворотний зв'язок. Нелінійні зв’язки між характеристиками завдань, продуктивністю команди та навичками її членів були досліджені в роботі [14], де використовується оцінка як зважений добуток найкращих і найгірших результатів. Робота [15] досліджує сучасні форми командної співпраці в індустріальних кібер-фізичних соціальних системах та пропонує підхід на основі прихованих марковських моделей (HMM) для покращення співпраці між фахівцями та агентами, застосовуючи показник collective intelligence (CI).

Аналіз літератури вказує на актуальність розробки методів ефективного підбору членів команди.

1.2 Постановка задачі оптимального підбору учасників робочої групи, що демонструють найбільший рівень продуктивності у співпраці

Задача сформульована зціллю підвищення рівня продуктивності командної роботи за рахунок ефективного підбору учасників, що демонструють найбільший рівень продуктивності у співпраці. Продуктивність співпраці визначається на основі на основі фактору змагання. Під фактором змагання розуміємо гру двох команд, в якій визначаються команда переможець та команда, що програла. Вважаємо, що виграш команді забезпечує ефективна взаємодія між членами цієї команди.

Будемо вважати продуктивність взаємодії в парі двох учасників високоефективною, якщо пара грала в команді, яка перемогла. Серія змагань складатися з окремих змагань, у кожному з яких беруть участь команди з однаковою кількістю учасників. Після проведення змагальної серії на основі отриманої інформації про склади команд та їх результати проводиться оцінка успішності взаємодії кожної пари учасників. Отримана оцінка дає підстави для формування команди.

1.3 Математична модель оптимального підбору учасників робочої групи, що демонструють найбільший рівень продуктивності у співпраці

В роботі [~~17~~16] авторами представлено математичну модель модель оптимального підбору учасників робочої групи, що демонструють найбільший рівень продуктивності у співпраці на повному неорієнтованому графі $G = (V, E)$, $|V| = n, |E| = \frac{n(n-1)}{2}$. Множина вершин $V = \{v_i | i = \overline{0, n-1}\}$ відповідає множині незалежних учасників, які необхідно об'єднати в m команд з заданою кількістю k гравців в кожній команді. Кожне ребро $e_{ij} \in E$ має чисельну вагу w_{ij}, яка відображає рівень продуктивності співпраці двох учасників v_i та v_j.

Допустимим розв'язком задачі є незв'язний підграф $x = (\tilde{V}, \tilde{E}), \tilde{V} \subset V, \tilde{E} \subset E$, компонентами зв'язності якого є m клік розмірності k. Кожна кліка об'єднує k гравців однієї команди. Кількість клік $m = \left[\frac{n}{k}\right]$, де $\left[\frac{n}{k}\right]$ — ціла частина від відношення $\frac{n}{k}$. Кількість вершин $r = |VV| = n - \left[\frac{n}{k}\right] k$ відповідає кількості учасників, що не ввійдуть в жодну з команд.

Множину всіх допустимих розв'язків (МДР) на графі $G = (V, E)$ позначимо через $X = X(G) = \{x\}$. На МДР визначимо векторну цільову функцію (ВЦФ):

Formatted: Russian

Formatted: Ukrainian

Formatted: Ukrainian

Formatted: Ukrainian

Formatted: Ukrainian

$$F = (F_1, F_2), \quad (1.1)$$

що складається з критеріїв:

$$MAXSUM: F_1 = \sum_{e_{ij} \in \tilde{E}} w_{ij} \rightarrow max \quad , \quad (1.2)$$

$$MAXMIN: F_2 = \min_{e_{ij} \in \tilde{E}} w_{ij} \rightarrow max. \quad (1.3)$$

ВЦФ визначає в МДР паретівську множину (ПМ) $\tilde{X}$, що складається з усіх паретівських оптимумів. Множина Парето — це множина всіх таких точок x*, які є парето-оптимальними. Тобто, множина Парето містить всі такі точки, де не існує іншої точки в множині допустимих рішень, яка б дала кращі результати за всіма критеріями одночасно [1817].

Будемо розглядати алгоритмічну проблему знаходження повної множини альтернатив (ПМА) X^0, яка є підмножиною ПМ мінімальної потужності і при цьому виконується умова $F(X^0) = F(\tilde{X})$ [1817].

Для визначення потужності МДР застосовується формула для невпорядкованого розбиття множини потужністю n на підмножини, серед яких для кожного $i = \overline{1,n}$ існує $m_i \geq 0$ з i елементами, де $\sum_{i=0}^{n} \Im_i = n$, тоді потужність визначимо за формулою

$$N(m_1, m_2, \dots, m_n) = \frac{n!}{m_1!, m_2!, \dots, m_n! (1!)^{m_1} (2!)^{m_2} \dots (n!)^{m_n}} \text{ [198]}.$$

Для описаного випадку ця формула буде мати два варіанти. Перший варіант: кількість n учасників є кратною до кількісті k гравців в кожній команді. Тоді всі учасники точно увійдуть до однієї з команд, тобто r=0. Кількість команд $m = \frac{n}{k}$. Розбиття множини учасників на команди є

Formatted: Russian
Formatted: Ukrainian
Formatted: Russian
Formatted: Russian
Formatted: Russian
Formatted: Indent: First line: 3 cm

невпорядкованим розбиттям множини з потужністю n на m підмножин однакової потужності k, де $km = n$. Тоді потужність МДР визначаємо формулою

$$|X| = \frac{n!}{m!(k!)^m}. \quad (1.4)$$

Formatted: Russian

Другий варіант: кількість n учасників не є кратною до кількісті k гравців в кожній команді. Кількість команд $m = \left[\frac{n}{k}\right]$. Тоді $r=n - \left[\frac{n}{k}\right]k$ учасників не стануть гравцями певної команди, тобто можна говорити, що створюється ще одна додаткова команда з r<k гравців. В цьому випадку розбиття множини учасників на команди є невпорядкованим розбиттям множини з потужністюn на m підмножин однакової потужності k та одну підмножину потужності r,— тобто $km + r = n$. Тоді потужність МДР визначаємо формулою

$$|X| = \frac{n!}{1!m!(r!)^1(k!)^m} = \frac{n!}{m!r!(k!)^m}. \quad (1.5)$$

Formatted: Russian

Є очевидним, що формула (1.5) співпадає з формулою (1.4) з огляду на $r!$=0!=1.——

Formatted: Russian

Formatted: Russian

Визначимо кількість команд, що дорівнює кількості числу клік $m = \left[\frac{n}{k}\right]$. Фактор змагання обумовлює, що варіанти, які є спрятливими для проведення змагань визначаються за природною умовою $m \geq 2$. Пріоритетним є варіант, в якому m є парним числом, $m = 2h, h \in N$.

Вважаємо, що перемогу команди забезпечує продуктивна взаємодія членів цієї команди кожного з кожним. Визначимо вагу ребер графу $G = (V, E)$ як визначенні частку перемог пари учасників v_i та v_j у загальній кількості змагань, де брала участь пара учасників v_i та v_j. В такій постановці кількість перемог пари визначається — кількістю N_{ij} перемог команд, до складу яких

входили учасники v_i та v_j, а загальна кількість змагань, де брала участь пара учасників v_i та v_j, загальною кількістю N_{ij}команд, до складу яких входили учасники v_i та v_j. Отже, для розрахунку реберної ваги w_{ij} будемо використовувати формулу:

$$w_{ij} = \frac{N_{ij}^*}{N_{ij}}, \tag{1.6}$$

де N_{ij}^* – кількість перемог команд, до складу яких входили учасники v_i та v_j,

N_{ij} – загальна кількість команд, до складу яких входили учасники v_i та v_j.

Загальна кількість команд N_{ij}, до складу яких входили учасники v_i та v_j, визначається кількістю можливих розбиттів множини $n-2$,так як одну пару зафіксовано, а всі інші учасники розбиваються на команди з k гравцями. Тоді– формула (1.5) приймає вид

$$N_{ij} = |X_{n-2}| = \frac{(n-2)!}{\left[\frac{n-2}{k}\right]!\left((n-2)-\left[\frac{n-2}{k}\right]k\right)!(k!)^{\left[\frac{n-2}{k}\right]}}. \tag{1.7}$$

Якщо кількість гравців в команді $k = 2$, то

$$N_{ij}{}' = |X_{n-2}| = \frac{(n-2)!}{\left[\frac{n-2}{2}\right]!\left((n-2)-\left[\frac{n-2}{2}\right]*2\right)!(2!)^m}. \tag{1.8}$$

Ціллю проведення змагань є визначення найбільш точного значення ваги ребра, тобто взаємодії в парі. В деяких випадках повний перебір всіх варіантів складів команд є практично неможливим. Розглянемо варіанти зменшення складності повного перебору МДР. Першим розглянемо варіант

Formatted: Russian

Formatted: Russian

Formatted: Russian

збільшення кількості пар за одне змагання, через збільшення кількості гравців в команді. В скільки разів зміниться кількість зустрічей можна підрахувати за формулами

$$|X_1| = \frac{n!}{m_1! r_1! (k_1!)^{m_1}}, \quad (1.10)$$

$$|X_2| = \frac{n!}{m_2! r_2! (k_2!)^{m_2}}, \quad (1.11)$$

$$\frac{|X_1|}{|X_2|} = m_2!\, r_2!\, (k_2!)^{m_2} / m_1!\, r_1!\, (k_1!)^{m_1}. \quad (1.12)$$

Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian

1.4 Найімовірніше число настання події за схемою Бернуллі

Проведення змагань є практичною реалізацією схеми Бернуллі [19]. Схема Бернуллі (або біноміальний розподіл) є класичним прикладом дискретного ймовірнісного розподілу, який описує кількість успіхів у серії незалежних випробувань, де кожне випробування має два можливих результати: успіх або невдача. Схема Бернуллі використовується для моделювання ситуацій, де подія може або відбутися, або не відбутися.

Формально, схема Бернуллі має такий опис: в кожному випробуванні є два можливі результати *успіх* з ймовірністю p та *невдача* з ймовірністю $1-p$. Параметрами є кількість випробувань n та кількість успіхів у серії випробувань k. Ймовірність того, що в серії з n випробувань буде рівно k успіхів, обчислюється за формулою біноміального розподілу

$$P = C_n^k p^k q^{n-k}.$$

Найімовірніше число k_0 настання події за схемою Бернуллі

$$\mu p - q \le k_0 \le \mu p + p,$$

де k_0 – найімовірніше число настання події А для μ;

μ – кількість експериментів;

$p(A)$ – ймовірність настання події А;

$q(A) = 1 - p(A)$.

Якщо організатори визначать пріоритетний варіант по бажаному числу зустрічей команд k_0, то оцінити необхідну кількість змагань $\aleph$ за формулою

$$(k_0 - p)/p \le \aleph \le (k_0 + q)/p, \qquad (1.13)$$

де $\aleph$ – необхідна кількість змагань $\aleph$ для n;

n – кількість учасників змагань;

k – кількість членів команди, $m = \left[\frac{n}{k}\right]$, $\mathrm{p} = \frac{|X_{n-2}|}{|X\ |}$, $|X| = \frac{n!}{r!m!(k!)^m}$,

$|X_{n-2}| = \frac{(n-2)!}{\left[\frac{n-2}{k}\right]!\left((n-2)-\left[\frac{n-2}{k}\right]k\right)!(k!)^{\left[\frac{n-2}{k}\right]}}$, $q = 1 - p$.

1.5 Висновки розділу 1

У цьому розділі було проведено аналіз актуальності підвищення ефективності створення робочих груп в умовах, де складні задачі вимагають міждисциплінарних підходів і співпраці між людьми з різними вміннями та досвідом. Сформульовано постановку задачі оптимального підбору учасників робочої групи, що демонструють найбільший рівень продуктивності у співпраці. Наведено математичну модель оптимального підбору учасників робочої групи, що демонструють найбільший рівень продуктивності у

Formatted: Font: (Default) Times New Roman, Not Italic

Formatted: Indent: First line: 1,5 cm

Formatted: Russian

Formatted: Font: Times New Roman, Not Italic

співпраці. Визначено всі математичні закономірності, які будуть застосовано для алгоритмів підтримки прийняття рішень.

Formatted: Justified, Indent: First line: 1,5 cm

2 СИСТЕМА ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ ОПТИМІЗАЦІЇ РОБОЧИХ ГРУП

2.1 Функціональна та структурна схеми системи підтримки прийняття рішень для оптимізації робочих груп

Функціональна схема системи підтримки прийняття рішень (СППР) для підбору учасників в команду на основі результатів змагань включає такі блоки (рис. 2.1).

Formatted: Russian

Блок введення початкової інформації дозволяє користувачу або адміністраторам системи вводити загальну інформацію, необхідну для роботи системи: прізвища учасників, кількість команд.

В *блоці формування пропозиції щодо довжини серії змагань* проводиться оцінка необхідної кількості змагань в серії для забезпечення бажаного числа зустрічей команд. В залежності від конкретної мети, цю підбір цього параметру можна проводити необхідну кількість разів.

В *блоці формування команд* здійснюється створення команд на основі заданих критеріїв. В залежності від конкретної мети, блок може автоматично формувати команди з учасників на основі їхніх характеристик або ж враховувати попередні результати розподілення учасників за командами.

В *блоці введення даних результатів серії змагань* відбувається фіксація результатів змагань між командами. Ці дані будуть використовуватися для подальшого аналізу взаємодії учасників та їхньої продуктивності.

На основі введених результатів і попередніх змагань *блок виведення рейтингу пар учасників* генерує рейтинг пар учасників, що базується на їхній взаємодії та результатах у серії змагань.

Зв'язки між блоками визначено так.

Введення початкової інформації служить для налаштування всієї системи, визначення вихідних умов та параметрів. Формування команд ґрунтується на цих вихідних даних для створення ефективних команд.

Введення даних серії змагань забезпечує процес фіксації та аналізу результатів кожного змагання. Виведення рейтингу пар учасників базується на результатах проведеної серії змагань і дозволяє оцінити ефективність співпраці між учасниками.

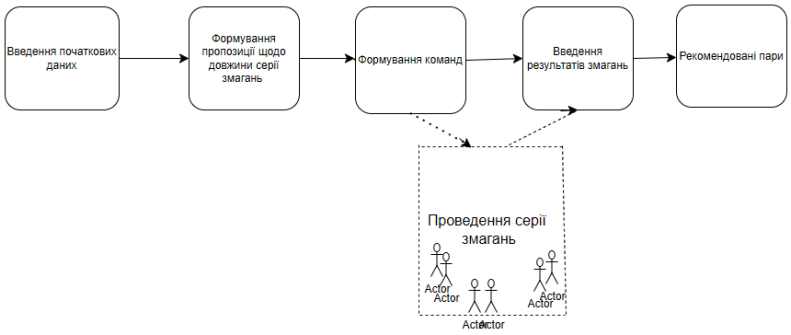


Рисунок 2.1 — Функціональна схема СППР для оптимізації робочих груп

На основі запропонованої функціональної схеми розроблено структурну схему (рис. 2.2).

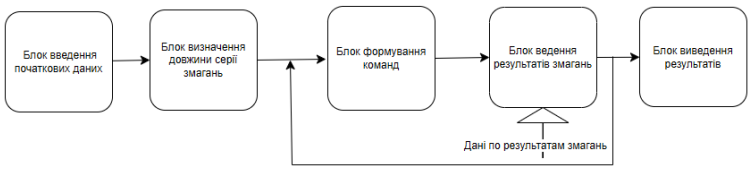


Рисунок 2.2 — Структурна схема СППР для оптимізації робочих груп

На цій схемі відображено, що застосування *блоків формування команд* та *введення результатів змагань* має відбуватися стільки разів, скільки є

попередньо визначена довжина серії змагань в *блоці визначення довжини серії змагань*.

2.2 Алгоритми роботи блоків СППР

В *блоці визначення довжини серії змагань* необхідна кількість змагань визначається за формулою (1.13) по бажаному числу зустрічей команд.

В *блоці формування команд* здійснюється створення команд на основі заданої кількості учасників та кількості команд. Застосовується алгоритм повного перебору всіх можливих комбінацій учасників.

На основі введених результатів і попередніх змагань *блок виведення рейтингу пар учасників* генерує рейтинг пар учасників, що базується на їхній взаємодії та результатах у серії змагань.

Рейтинг розраховується відповідно до моделі (1.1)—(1.3). За формулою (1.5) визначається вага ребер. Для цього йде перебір результатів серії змагань з підрахунком даних по кількості перемог команд, до складу яких входили учасники v_i та v_j, та загальної кількості команд, до складу яких входили учасники v_i та v_j.

Formatted: Russian

Formatted: Russian

Formatted: Russian

2.3 Програмна реалізація СППР

У сучасному світі програмування для розробки веб-додатків та систем все більш популярними стають різноманітні фреймворки, які спрощують процес створення складних і ефективних програмних рішень. Один із таких фреймворків, що отримав визнання серед розробників, — це Django, який широко використовується для створення веб-додатків з використанням мови

програмування Python. Оскільки в даному дослідженні розглядається модель формування команд гравців, вибір фреймворку Django для її програмної реалізації є цілком обґрунтованим і ефективним рішенням. У цьому контексті важливо розглянути переваги та особливості Python і Django, що роблять їх оптимальними для вирішення поставлених завдань.

Мова програмування Python є однією з найбільш популярних серед розробників завдяки своїй простоті, гнучкості та великим можливостям. Однією з головних переваг Python є його зрозумілий синтаксис, який дозволяє програмістам швидко писати та підтримувати код. Python підтримує різноманітні парадигми програмування, включаючи об'єктно-орієнтоване, функціональне та процедурне програмування, що дозволяє адаптувати мову для вирішення різних типів задач.

Python має велику кількість бібліотек і фреймворків, що робить його універсальним інструментом для розробки різних типів програмного забезпечення — від веб-додатків до наукових і статистичних обчислень. Для створення моделей і систем, таких як модель формування команд, Python забезпечує зручні інструменти для роботи з даними, математичними розрахунками, статистичними моделями і алгоритмами машинного навчання, що дозволяє розробникам зосередитися на логіці вирішення завдання, а не на низькорівневій реалізації.

Крім того, Python підтримує інтеграцію з іншими мовами програмування та технологіями, що робить його ідеальним вибором для розробки складних гібридних систем. Він також має активну спільноту розробників, що сприяє постійному розвитку мови та появі нових інструментів для вирішення сучасних завдань.

Django — це високорівневий фреймворк для розробки веб-додатків на Python, що забезпечує зручне середовище для створення масштабованих, надійних і безпечних веб-ресурсів. Однією з головних переваг Django є його "рішення всього": він включає в себе більшість необхідних інструментів для

створення веб-додатків, що значно прискорює розробку та дозволяє програмістам зосередитися на специфічних аспектах проєкту.

Однією з головних особливостей Django є використання принципу "не повторюйся" (DRY — Don’t Repeat Yourself), що дозволяє уникати дублювання коду та сприяє його чіткій організації. Завдяки цьому фреймворк забезпечує високу продуктивність і легкість в обслуговуванні розроблених систем. Django автоматизує багато процесів, таких як робота з базами даних, обробка запитів користувачів, формування шаблонів і маршрутизація, що робить його надзвичайно зручним для розробки складних веб-додатків.

Крім того, Django має вбудовані інструменти для роботи з базами даних, що дозволяє легко інтегрувати їх у проєкт і виконувати запити до даних без необхідності писати складні SQL-запити. Для моделі формування команд гравців це є важливою перевагою, оскільки робота з даними про гравців і їхні характеристики потребує ефективної взаємодії з базою даних.

Ще однією важливою особливістю Django є його система аутентифікації та авторизації користувачів. Це забезпечує високий рівень безпеки веб-додатків, що важливо при обробці персональних даних гравців. Django також підтримує інтернаціоналізацію та локалізацію, що дозволяє створювати багатомовні додатки для міжнародних команд.

Фреймворк має велику кількість готових до використання модулів і бібліотек для реалізації різних функцій, таких як обробка форм, взаємодія з API, інтеграція з платежами, реалізація обробки файлів тощо. Це дозволяє розробникам швидко і ефективно додавати потрібні функціональності без необхідності писати код з нуля.

Django дозволяє безперешкодно інтегруватися з іншими популярними технологіями для розробки сучасних веб-додатків. Для реалізації моделі формування команд можна використовувати різноманітні бібліотеки машинного навчання, наприклад, Scikit-learn, для обробки великих обсягів даних та побудови математичних моделей. Інтеграція Django з бібліотеками

для роботи з даними та аналітики дозволяє створювати потужні веб-додатки, які можуть обробляти складні алгоритми та моделі.

Додатково, Django підтримує різноманітні інтерфейси для взаємодії з фронтендом, що дозволяє створювати інтерактивні та адаптивні інтерфейси користувачів, що є важливим для надання користувачам можливості ефективно взаємодіяти з системою формування команд.

У сучасній веб-розробці ефективне управління даними є одним із ключових аспектів, що визначає продуктивність та масштабованість програмного забезпечення. Одним із найпотужніших інструментів для роботи з базами даних у фреймворку Django є система об'єктно-реляційного відображення (ORM — Object-Relational Mapping). Django ORM забезпечує розробникам зручний спосіб взаємодії з базами даних, дозволяючи працювати з ними через об'єкти Python без необхідності написання SQL-запитів вручну.

Django ORM дозволяє розробникам працювати з базами даних, використовуючи зрозумілу та зручну об'єктно-орієнтовану модель. Наприклад, для створення таблиці в базі даних достатньо оголосити клас моделі:

```
from django.db import models
class Player(models.Model):
    name = models.CharField(max_length=100)
    age = models.IntegerField()
    team = models.CharField(max_length=50)
```

Завдяки такій простій декларації моделі Django автоматично генерує необхідні SQL-інструкції для створення відповідної таблиці в базі даних.

Django ORM пропонує вбудований механізм міграцій, що дозволяє легко змінювати структуру бази даних без необхідності ручного написання SQL-коду. Це забезпечує узгодженість даних та спрощує процес оновлення додатків.

Django ORM підтримує роботу з різними системами управління базами даних (PostgreSQL, MySQL, SQLite, Oracle), що дозволяє легко змінювати СУБД без необхідності внесення значних змін у код додатка.

Django ORM надає потужні механізми оптимізації запитів, такі як використання методів select_related() і prefetch_related(), які допомагають мінімізувати кількість запитів до бази даних та запобігають проблемі "N+1 запитів". Наприклад:

```
players = Player.objects.select_related('team').all()
```

Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian
Formatted: Ukrainian

Цей запит дозволяє отримати всі об'єкти гравців разом із пов'язаними командами одним SQL-запитом, що значно підвищує продуктивність.

Використання Django ORM забезпечує автоматичне екранування введених користувачем даних, що значно знижує ризик SQL-ін'єкцій та підвищує рівень безпеки веб-додатків.

Розглянемо приклад вибірки даних з використанням фільтрації для того щоб отримати всіх гравців віком понад 18 років:

```
adult_players = Player.objects.filter(age__gt=18)
```

Або додавання нового запису до бази даних:

```
new_player = Player.objects.create(name="Іван", age=25, team="Динамо")
```

Ці операції виконуються просто та ефективно, дозволяючи зосередитися на бізнес-логіці, а не на написанні складних SQL-запитів.

Класи моделей Django потрібні для роботи з базою даних у веб-додатках. Вони дозволяють зручно визначати структуру даних та

взаємозв’язки між ними, використовуючи об'єктно-реляційне відображення (ORM). В розробленому додатку використовуються такі класи:

UserProfile (Профіль користувача). Цей клас пов'язаний із вбудованою в Django моделлю користувача User за допомогою зв’язку OneToOneField, тобто кожен користувач має один профіль.

Зберігає такі дані:

- img — зображення профілю.;
- num_games — кількість зіграних ігор.;
- won_games_ratio — коефіцієнт перемог (співвідношення виграних ігор до загальної кількості).);
- Метод метод won_games_list() повертає список виграних ігор користувача.;
- Метод метод lost_games_list() повертає список програних ігор користувача.;
- Зв’язокзв’язок: один профіль належить одному користувачу.

Command (Команда):

- Кожна кожна команда прив'язана до конкретної гри через ForeignKey (game), тобто кожна гра може мати кілька команд.
- Користувачі можуть входити в кілька команд завдяки зв’язку ManyToManyField.
- Поле is_winner вказує, чи виграла команда.
- Автоматично зберігаються час створення (created) та оновлення (updated).
- Зв’язок: одна гра може мати багато команд, а один користувач може належати до кількох команд.

Game (Гра):

- Містить містить поля для запису часу створення та оновлення гри.
- Поле related_name='commands' у моделі Command дозволяє отримувати всі команди, пов’язані з певною грою.
- Зв’язок: одна гра — багато команд.

Formatted: Indent: Left: 0 cm, First line: 1,5 cm, Outline numbered + Level: 1 + Numbering Style: Bullet + Aligned at: 0,75 cm + Tab after: 1,25 cm + Indent at: 1,25 cm, Tab stops: Not at 0 cm + 1,25 cm

Formatted: Ukrainian

Formatted: Indent: Left: 0 cm, First line: 1,5 cm, Outline numbered + Level: 1 + Numbering Style: Bullet + Aligned at: 0,75 cm + Tab after: 1,25 cm + Indent at: 1,25 cm, Tab stops: Not at 0 cm + 1,25 cm

Formatted: Indent: Left: 0 cm, First line: 1,5 cm, Outline numbered + Level: 1 + Numbering Style: Bullet + Aligned at: 0,75 cm + Tab after: 1,25 cm + Indent at: 1,25 cm, Tab stops: Not at 0 cm + 1,25 cm

Couple (Пара гравців):

- Визначає пару користувачів, які можуть грати разом.
- Поля player1 і player2 використовують зв’язок ForeignKey з моделлю користувача, що дозволяє встановити унікальні пари гравців.
- Містить статистику:
- num_games – кількість ігор, у яких вони брали участь разом;
- weight – певний показник важливості або ефективності пари.
- Зв’язок: кожен користувач може бути у кількох парах з іншими гравцями.

Solution (Рішення):

- Містить текстові поля team1 та team2, які, ймовірно, зберігають інформацію про команди у певному рішенні.
- Поля maxsum і maxmin можуть використовуватися для зберігання розрахункових значень, наприклад, максимального загального рахунку та мінімального значення серед команд.
- Зв’язок: модель є незалежною і може використовуватись для аналізу та збереження рішень.

Діаграма класів представлена на рис. 2.3.

Детальний лістинг класів моделей наведено в додатку А.

Formatted: Indent: Left: 0 cm, First line: 1,5 cm, Outline numbered + Level: 1 + Numbering Style: Bullet + Aligned at: 0,75 cm + Tab after: 1,25 cm + Indent at: 1,25 cm, Tab stops: Not at 0 cm + 1,25 cm

Formatted: Indent: Left: 1,5 cm, No bullets or numbering

Formatted: Indent: Left: 0 cm, First line: 1,5 cm, Bulleted + Level: 2 + Aligned at: 2,14 cm + Indent at: 2,77 cm, Tab stops: Not at 0 cm

Formatted: Indent: Left: 0 cm, First line: 1,5 cm, Outline numbered + Level: 1 + Numbering Style: Bullet + Aligned at: 0,75 cm + Tab after: 1,25 cm + Indent at: 1,25 cm, Tab stops: Not at 0 cm + 1,25 cm

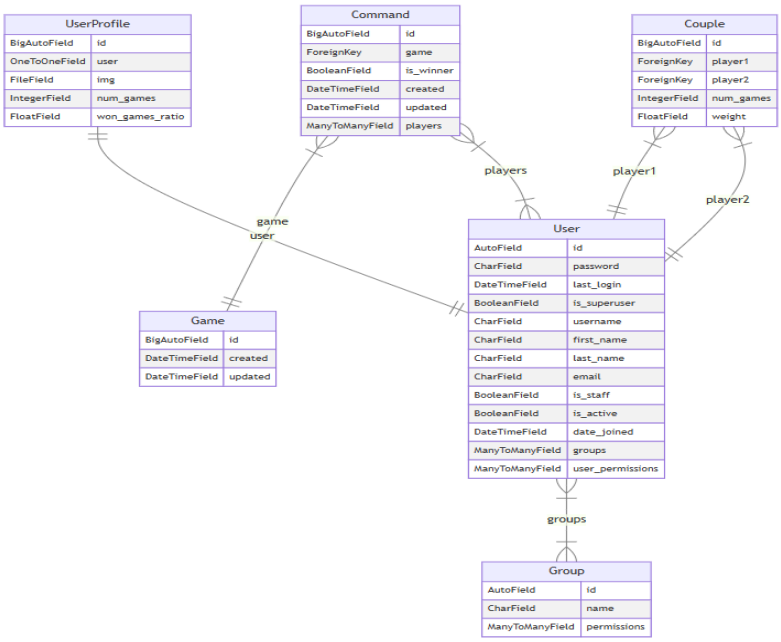


Рисунок 2.3 — Діаграма класів додатку

Взаємодія з базою даних відбувається через браузер у вигляді пов'язаних веб-сторінок. Для створення нової гри використовується екранна форма (рис. 2.4)

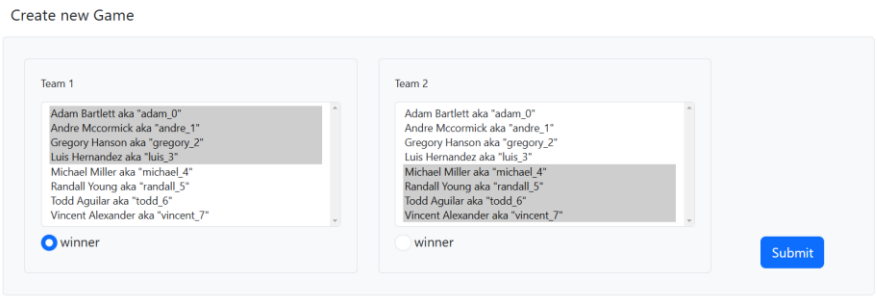
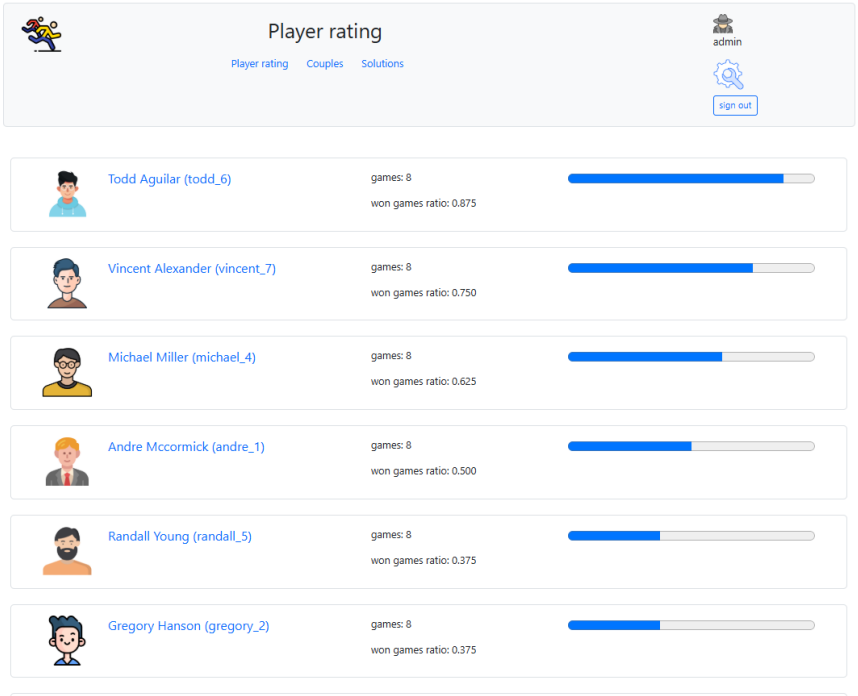


Рисунок 2.4 — Форма створення нової гри

Після обчислення даних гравців ми можемо побачити їх рейтинг на окремій сторінці (рис. 2.5)



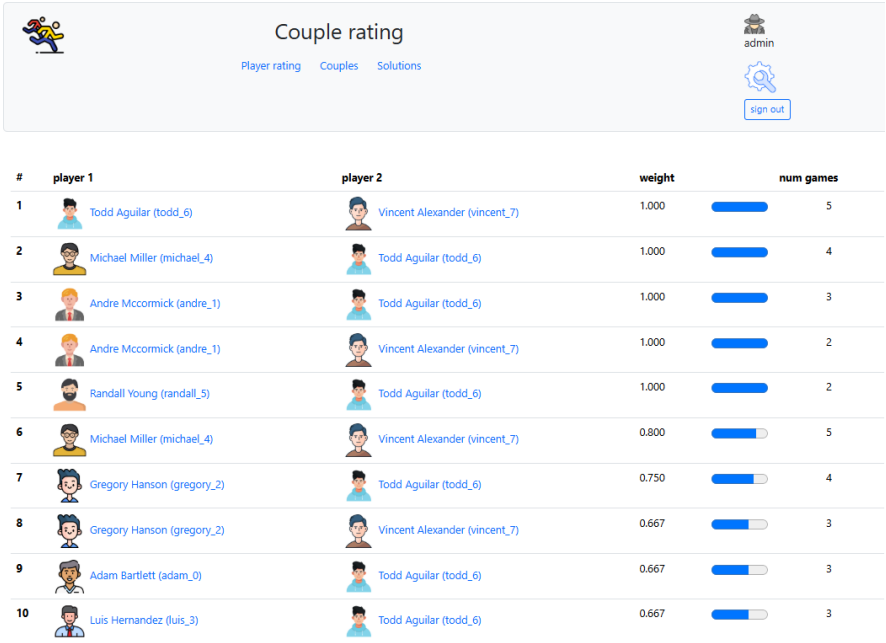
-Рисунок 2.5 — Сторінка рейтингу гравців

На окремій сторінці додатку наведені усі можливі пари гравців, відсортовані за їх вагою та кількістю ігор (рис. 2.6).

Для візуалізації паретівської множини використана діаграма з маркерами на окремій сторінці (рис. 2.7).

Formatted: Justified

Formatted: Justified



-Рисунок 2.6 — Сторінка з інформацією про пари гравців

-Рисунок 2.7 — Сторінка з діаграмою паретівської множини

Представлення в Django — це шар додатка, який обробляє HTTP-запити та формує HTTP-відповіді, використовуючи дані з моделей та

шаблонів. Вони слугують своєрідним «посередником» між користувачем і системою, реалізуючи бізнес-логіку, обробку форм, фільтрацію даних та перенаправлення. У Django є два основні типи представлень: функціональні (function-based views) та класові (class-based views). Функціональні представлення є простішими та використовуються для невеликих завдань, тоді як класові надають більше можливостей для повторного використання коду та організації складних запитів. Представлення отримують дані від моделі, передають їх у шаблони та відображають користувачеві готовий контент, такий як веб-сторінки або JSON-відповіді в API. Лістінг наведено в додатку Б.

2.4 Висновки розділу 2

У цьому розділі було побудовано функціональну та структурну схеми системи підтримки прийняття рішень для оптимізації робочих груп, описано алгоритми, які застосовуються для реалізації функціоналу, розглянуто особливості програмної реалізації СППР.

3 МОДЕЛЮВАННЯ СЕРІЇ ЗМАГАНЬ ТА РОБОТИ СППР

3.1 Визначення кількість змагань ℵ

Розглянемо експеримент, в якому 8 гравців розподіляються за двома командами по 4 гравці.

Проведемо розрахунок потужності МДР за формулою (1.4).

$$|X_8| = \frac{n!}{m!(k!)^m} = \frac{8!}{2!(4!)^2} = 35.$$

Загальну кількість команд N_{ij}, до складу яких входили учасники v_i та v_j, визначаємо за формулою (1.7):

$$|X_6| = \frac{(n-2)!}{\left[\frac{n-2}{k}\right]!\left((n-2)-\left[\frac{n-2}{k}\right]k\right)!(k!)^{\left[\frac{n-2}{k}\right]}} = \frac{6!}{(2!)(4!)} = 15.$$

Проведемо розрахунок ймовірності p входження пари гравців до однієї команди в кожному з незалежних випробувань за формулою

$$p = \frac{|X_6|}{|X_8|} = \frac{15}{35} = \frac{3}{7},$$

де $|X_8|$ – загальна кількість можливих команд з 8 учасників по чотири гравці згідно формули.

Formatted: English (United States)

Formatted: Indent: First line: 1,5 cm

$|X_6|$ – загальна кількість можливих команд з 6 учасників, де одна команда об'єднує чотири гравці та одна команда складається з двох гравців формула (1.1), так як одну пару зафіксовано.

Будемо планувати серію з $\aleph$ змагань.

Зафіксуємо число входжень пари гравців до однієї команди в серії з $\aleph$ змагань $k = 3$.

Тоді за формулою (1.13)

$$(k_0 - p)/p \le \aleph \le (k_0 + q)/p,$$
$$(3 - \tfrac{3}{7})/(\tfrac{3}{7}) \le \aleph \le (3 + (\tfrac{1}{7}))/(\tfrac{3}{7}),$$
$$6 \le \aleph \le 8.$$

Приймемо, $\aleph = 8$.

3.2 Числовий експеримент по генерації команд

Побудуємо числовий експеримент та випадковим чином згенеруємо склад двох команд по 4 гравці з 8 гравців для серії з $\aleph = 8$ змагань, таким чином, щоб кожна пара гравців грала за одну команду не менше двох разів (табл. 3.1). Гравці позначені індексами від 0 до 7 включно. Розподіл кількості входжень наведено на рис. 3.1. Максимальна кількість входжень в одну команду дорівнює 5 (наприклад для пари (v_0, v_2)), а середнє значення кількості дорівнює 3.43, що відповідає проведеним теоретичним оцінкам.

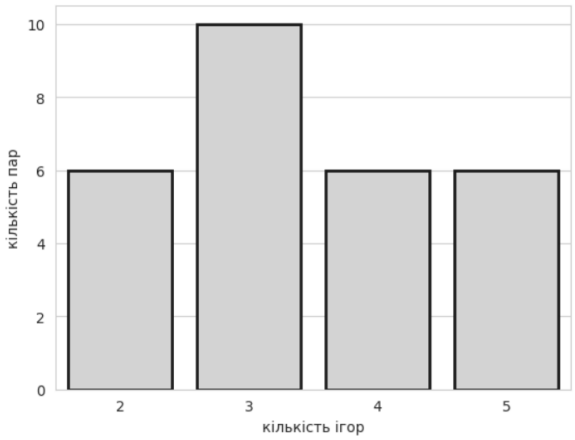


Рисунок 3.1 – Розподіл кількості входжень пар гравців залежно від кількості змагань

Запропонований план змагань не є єдиним можливим. Однак, зазначимо, що повна кількість варіантів формування команд в цьому прикладі $|X_8| = 35$, тобто було реалізовано 22.8% можливих варіантів.

В табл. 3.1 наведено склад команд в серії з 8 змагань та зафіксовано переможців.

Таблиця 3.1 –

Розподіл гравців за 2 командами

№	Склад команди 1	Склад команди 2	Переможець
1	0, 2, 3, 4	1, 5, 6, 7	команда 2
2	0, 2, 3, 6	1, 4, 5, 7	команда 2
3	0, 1, 2, 3	4, 5, 6, 7	команда 2
4	0, 1, 3, 5	2, 4, 6, 7	команда 2
5	0, 2, 6, 7	1, 3, 4, 5	команда 1
6	0, 1, 4, 6	2, 3, 5, 7	команда 1
7	0, 1, 2, 5	3, 4, 6, 7	команда 2
8	0, 4, 5, 7	1, 2, 3, 6	команда 2

Знайдемо ваги ребер за формулою (1.6). Повний перелік значень реберних ваг за зменшенням та відповідних до них пар учасників наведено в таблиці 2. Наприклад, $w_{26} = 0.750$, тому що пара (v_2, v_6) брала участь в чотирьох змаганнях (за номерами 2, 4, 5 та 8), з яких 3 були для неї переможні (4, 5 та 8). Також $w_{03} = 0.0$, тому що пара (v_0, v_3) жодного разу не брала участь в переможній команді.

Таблиця 3.2

– Значення реберних ваг та індекси пар гравців

w_{ij}	Індекси учасників (i, j)
1.000	(1, 6), (1, 7), (4, 6), (5, 6), (6, 7)
0.800	(4, 7)
0.750	(2, 6)
0.667	(0, 6), (1, 4), (2, 7), (3, 6)
0.600	(5, 7)
0.500	(0, 7), (2, 4), (3, 7), (4, 5)
0.400	(1, 5)
0.333	(0, 4), (1, 2), (3, 4)
0.250	(0, 1), (1, 3)
0.200	(0, 2), (2, 3)
0.000	(0, 3), (0, 5), (2, 5), (3, 5)

3.3 Розв’язання двокритеріальної задачі (1.1)–(1.3)

Для демонстрації розв’язання задачі (1.1)–(1.3) побудуємо аналогічний експеримент для $n = 8, k = 3$. Метою було створення двох команд по три учасника з восьми кандидатів. МДР є множиною підграфів x, кожна з яких містить дві кліки (m =2) розмірністю три ($k = 3$). Таких варіантів

за формулою (1.4) $|X| = \frac{8!}{2!(3!)^2} = 280$, що визначає потужність МДР. Після цього треба виконати розрахунки значень ВЦФ для усіх елементів МДР. Візуалізація цих розрахунків надана на рис. 3.2.

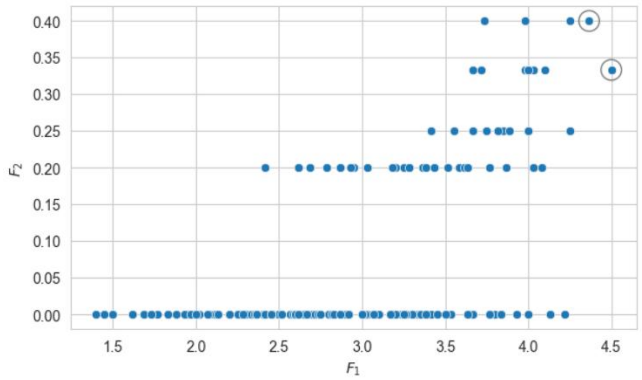


Рисунок 3.2 – Значення ВЦФ (1.1) – (1.3) для МДР

Маємо два розв'язки, які утворюють ПМ : $x_{229} = \big((v_1, v_2, v_7), (v_3, v_5, v_6)\big)$, $x_{260} = \big((v_1, v_5, v_6), (v_2, v_4, v_7)\big)$. На рис. 3.2 маркерами виділено два розв'язки, які утворюють ПМА.

В даному випадку ПМ та ПМА збігаються. Можна стверджувати, що побудовано множину кращих варіантів формування команд із заданими параметрами з урахуванням позитивного практичного досвіду взаємодії між учасниками.

3.4 Висновки розділу 3

У цьому розділі наведено результати проведених експериментів по роботі СППР на змодельованих даних серії змагань. У результаті проведеного експерименту було проаналізовано розподіл 8 гравців на дві команди по 4

учасники та ймовірність входження певної пари гравців до однієї команди. Під час числового експерименту була отримана середня кількість входжень пари до однієї команди, що відповідає теоретичним оцінкам. Визначення ваг ребер і аналіз результатів МДР продемонстрували ефективність запропонованих методів для оптимізації розподілу учасників та врахування взаємодії між гравцями.

ВИСНОВОК

Дослідження присвячене розробці системи підтримки прийняття рішень для оптимізації робочих груп.

Проведено аналіз актуальності підвищення ефективності створення робочих груп в умовах, де складні задачі вимагають міждисциплінарних підходів і співпраці між людьми з різними вміннями та досвідом. Визначено як фактор підвищення рівня продуктивності робочої групи ефективний підбір учасників, що демонструють найбільший рівень продуктивності у співпраці. Доведено, що розробка СППР для оптимізації робочих груп є актуальною задачею.

Проведено етапи проєктування та програмної реалізації СППР для оптимізації робочих груп з використанням мови програмування Python фреймворк Django. Аналіз результатів роботи СППР для даних змодельованної серії змагань показав ефективність обраної стратегії вибору пар учасників, які є ефективними у співпраці.

Рекомендації щодо використання результатів роботи — застосування для формування робочих груп при можливості попереднього проведення серії змагань.

ПЕРЕЛІК ПОСИЛАНЬ

1. Kozlowski, S.W.J. Enhancing the Effectiveness of Work Groups and Teams / S.W.J. Kozlowski, D.R. Ilgen // Psychological Science in the Public Interest. — 2006. — Vol. 7, I. 3. — P. 77—124. — doi: https://doi.org/10.1111/j.1529-1006.2006.00030.x.

2. Unveiling the IDB's Project Executing Units: Performance Indicators, Results-Based Management, and Demand for Knowledge / Acosta Mejía, C. [et al.]. — USA, 2024. — doi :http://dx.doi.org/10.18235/0013040.

3. Sa'diyah, K. Occupational Health and Safety, Training, and Teamwork for Hospital Medical Performance / K. Sa'diyah // Safety and Health for Medical Workers. — 2024. — Vol. 1, I. 1. — P. 29—38. — doi :https://doi.org/10.69725/ehxp3d73.

4. Gamification of cooperation: A framework, literature review and future research agenda / M. Riar [et al.] // International Journal of Information Management. — 2022. — Vol. 67. — 102549. — 10.1016/j.ijinfomgt.2022.102549.

5. Poth, A. Orchestrating Agile IT Quality Management for Complex Solution Development Through Topic-Specific Partnerships in Large Enterprises – An Example on the EFIS Framework / A. Poth, M. Kottke, A. Riel // Systems, Software and Services Process Improvement (EuroSPI 2021). Communications in Computer and Information Science. — 2021. — Vol. 1442. — doi :10.1007/978-3-030-85521-5_7.

6. Yang, W. Z. Teamwork triumphs: fostering student development through cooperative learning in university physical education / W.Z. Yang // International Journal of Current Educational Practice. — 2024. — Vol. 12, I. 2. — P. 114—120. —https://doi.org/10.5281/zenodo.10954880.

7. Hebles, M. Teamwork competence and collaborative learning in entrepreneurship training / M. Hebles, C. Yaniz-Alvarez-de-Eulate, M. Jara //

Formatted: Justified

European J. of International Management. — 2023. — Vol. 20, I. 2. — P. 238—255. — doi :10.1504/EJIM.2020.10022225.

8. Sangadji, H. (2023). The Influence of Emotional Intelligence, Teamwork, And Organizational Citizenship Behavior on Performance Agricultural Extension Workers in Banten Province. / H. Sangadji // International Journal of Human Capital Management). — 2023. — Vol. 7, I. 1. — P. 141—157. — doi :https://doi.org/10.21009/IJHCM.07.01.11I.

9. Measuring team trust: A critical and meta-analytical review / J. Feitosa [et al.] // Journal of Organizational Behavior. — 2020. — Vol. 41, I. 5. — P. 1–23. — doi: https://doi.org/10.1002/ job.2436.

10. Team Incentives, Social Cohesion, and Performance: A Natural Field Experiment. / J. Delfgaauw // Management Science. — 2022. — Vol. 68, I. 1. — P. 230—256. — doi:https://doi.org/10.1287/mnsc.2020.3901.

11. Emergence of collective intelligence in industrial cyber-physical-social systems for collaborative task allocation and defect detection / I.L.D. Makanda // Computers in Industry. — 2023. — Vol. 152. — 104006. — doi:https://doi.org/10.1016/j.compind.2023.104006.

12. Putting teams into the gig economy: A feld experiment at a ride-sharing platform / A. Wei // Management Science. — 2023. — Vol. 69, I. 9 — P. 5336—5353. — doi:https://doi.org/10.1287/mnsc.2022. 4624.

13. Ertac, S. Strategic feedback in teams: Theory and experimental evidence / S. Ertac, M. Gümren, L. Koçkesen // Journal of Economic Behavior & Organization. — 2019 — Vol. 162. — P. 1—23. — doi: https://doi.org/10.1016/j.jebo.2019.04.005.

14. Fischer, M. When, and why, do teams benefit from self-selection? / M. Fischer, R.M. Rilke, B.B. Yurtoglu // Experimental Economics. — 2023. — Vol. 26. — P. 749—774. — doi: https://doi.org/10.1007/s10683-023-09800-2.

15. Teaching agents to understand teamwork: Evaluating and predicting collective intelligence as a latent variable via Hidden Markov Models / M. Zhao [et

Formatted: English (United States)

Formatted: English (United States)

Formatted: English (United States)

al.] // Computers in Human Behavior. — 2023. — Vol. 139. — doi: https://doi.org/10.1016/j.chb.2022.107524.

16. Competition design to create high-performance teams / A. Ryabenko [et al.] // *ScienceRise*. — 2024. — Vol. 2. — P. 26-35. — https://doi.org/10.21303/2313-8416.2024.003623.

17. Perepelytsya V.A. Bahatokryterial′ni modeli ta metody zadach optymizatsiyi hrafiv / V.A. Perepelytsya. — Saarbrücken: LAP LAMBERT Academic Publishing, 2013. — 336 p.

18. Mansour, T. Combinatorics of Set Partitions (1st ed.) / T. Mansour. — New York : Chapman and Hall/CRC, 2012. — 516 p. — doi: https://doi.org/10.1201/b12691.

19. Огірко О.І. Теорія ймовірностей та математична статистика: навчальний посібник / О.І. Огірко, Н.В. Галайко. — Львів: ЛьвДУВС, 2017. — 292 с.

1. Kozlowski, S. W. J., & Ilgen, D. R. (2006). Enhancing the Effectiveness of Work Groups and Teams. Psychological Science in the Public Interest, 7(3), 77-124. doi : https://doi.org/10.1111/j.1529-1006.2006.00030.x

2. Acosta Mejía, C., Cambra Carrizo, G., Martinez-Carrasco, J., Cerda, M.,Kang, M. (2024). Unveiling the IDB's Project Executing Units: Performance Indicators, Results-Based Management, and Demand for Knowledge. doi :http://dx.doi.org/10.18235/0013040.

3. Sa'diyah, K. (2024). Occupational Health and Safety, Training, and Teamwork for Hospital Medical Performance. Safety and Health for Medical Workers, 1(1), 29-38. doi :https://doi.org/10.69725/ehxp3d73.

4. Riar, Marc & Morschheuser, Benedikt & Zarnekow, Ruediger & Hamari, Juho. (2022). Gamification of cooperation: A framework, literature review and future research agenda. International Journal of Information Management. 67. 102549. 10.1016/j.ijinfomgt.2022.102549.

5. Poth, A., Kottke, M., Riel, A. (2021). Orchestrating Agile IT Quality Management for Complex Solution Development Through Topic Specific

Formatted: Russian

Partnerships in Large Enterprises – An Example on the EFIS Framework. In: Yilmaz, M., Clarke, P., Messnarz, R., Reiner, M. (eds) Systems, Software and Services Process Improvement. EuroSPI 2021. Communications in Computer and Information Science, vol 1442. Springer, Cham. doi :10.1007/978-3-030-85521-5_7.

6. Yang, W. Z. (2024). Teamwork triumphs: fostering student development through cooperative learning in university physical education. International Journal of Current Educational Practice, 12(2), 114-120. https://doi.org/10.5281/zenodo.10954880

7. Hebles, M., Yaniz-Alvarez-de-Eulate, C., Jara, M. (2023). Teamwork competence and collaborative learning in entrepreneurship training, European J. of International Management, 20 (2), 238-255. doi :10.1504/EJIM.2020.10022225.

8. Sangadji, H. (2023). The Influence of Emotional Intelligence, Teamwork, And Organizational Citizenship Behavior on Performance Agricultural Extension Workers in Banten Province. IJHCM (International Journal of Human Capital Management), 7(1), 141-157. doi :https://doi.org/10.21009/IJHCM.07.01.11I

9. Feitosa, J., Grossman, R., Kramer, Ws., Salas, E.(2020). Measuring team trust: A critical and meta-analytical review. Organ Behav, 1-23. doi: https://doi.org/10.1002/ job.2436.

10. Delfgaauw, J., Dur, R., Onemu, O., Sol, J. (2022). Team Incentives, Social Cohesion, and Performance: A Natural Field Experiment. Management Science, 68(1), 230-256. doi:https://doi.org/10.1287/mnsc.2020.3901

11. Makanda, I.L.D., Jiang, P., Yang, M.,Shi, H. (2023). Emergence of collective intelligence in industrial cyber physical social systems for collaborative task allocation and defect detection, Computers in Industry, Volume 152. doi:https://doi.org/10.1016/j.compind.2023.104006.

12. Wei, A., Chen, Y., Mei, Q., Ye, J., & Zhang, L. (2023). Putting teams into the gig economy: A feld experiment at a ride sharing platform. Management Science. doi:https://doi.org/10.1287/mnsc.2022. 4624

13. Seda Ertac, Mert Gümren, Levent Koçkesen,Strategic feedback in teams: Theory and experimental evidence,Journal of Economic Behavior & Organization,Volume 162, 2019, Pages 1-23, doi: https://doi.org/10.1016/j.jebo.2019.04.005.

14. Fischer, M., Rilke, R.M. & Yurtoglu, B.B. (2023). When, and why, do teams benefit from self-selection? Exp Econ 26, 749–774. doi: https://doi.org/10.1007/s10683-023-09800-2.

15. Zhao, M., Eadeh, F. R., Nguyen, Th.-N., Gupta, P., Admoni, H., Gonzalez, C., Williams Woolley, A. (2023). Teaching agents to understand teamwork: Evaluating and predicting collective intelligence as a latent variable via Hidden Markov Models, Computers in Human Behavior, Volume 139. doi: https://doi.org/10.1016/j.chb.2022.107524.

16. Mansour, T. (2012). Combinatorics of Set Partitions (1st ed.). Chapman and Hall/CRC. doi: https://doi.org/10.1201/b12691.

17. Ryabenko, A., Bakurova, A., Tereschenko, E., MatiukhinA., & Pyrozhok, A. (2024). Competition design to create high-performance teams. *ScienceRise*, (2), 26–35. https://doi.org/10.21303/2313-8416.2024.003623

18. Perepelytsya V. A. Bahatokryterial′ni modeli ta metody zadach optymizatsiyi hrafiv. Saarbrücken: LAP LAMBERT Academic Publishing, 2013. 336 p.

19. Огірко О. І., Галайко Н. В. Теорія ймовірностей та математична статистика: навчальний посібник / О. І. Огірко, Н. В. Галайко. — Львів: ЛьвДУВС, 2017. — 292 с.

ДОДАТОК А

КОД PYTHON (файл models.py)

```
from django.contrib.auth.models import User
from django.db import models



class UserProfile(models.Model):
    user = models.OneToOneField(User,
related_name='profile', on_delete=models.CASCADE)
    img = models.ImageField(blank=True, upload_to='users/')
    num_games = models.IntegerField(default=0)
    won_games_ratio = models.FloatField(default=0.0)


    def __str__(self):
        return f'{self.user.get_full_name()}
({self.user.username})'


    def won_games_list(self):
        return
self.user.commands.filter(is_winner=True).select_related('ga
me').all()


    def lost_games_list(self):
        return
self.user.commands.filter(is_winner=False).select_related('g
ame').all()
```

```
class Command(models.Model):
    game = models.ForeignKey('Game',
related_name='commands', on_delete=models.CASCADE,
blank=True)
    players = models.ManyToManyField(User,
related_name='commands', blank=True)
    is_winner = models.BooleanField(default=False)
    created = models.DateTimeField(auto_now_add=True)
    updated = models.DateTimeField(auto_now=True)
    objects = models.Manager()



    def __str__(self):
        return f'command-{self.id}'



class Game(models.Model):
    # commands: Command
    created = models.DateTimeField(auto_now_add=True)
    updated = models.DateTimeField(auto_now=True)
    objects = models.Manager()



    def __str__(self):
        return f'game-{self.id}'



class Couple(models.Model):
```

```
    player1 = models.ForeignKey(User,
on_delete=models.CASCADE, related_name='couples1')
    player2 = models.ForeignKey(User,
on_delete=models.CASCADE, related_name='couples2')
    num_games = models.IntegerField(default=0)
    weight = models.FloatField(default=0.0)
    objects = models.Manager()



class Solution(models.Model):
    team1 = models.TextField(blank=True)
    team2 = models.TextField(blank=True)
    maxsum = models.FloatField(default=0.0)
    maxmin = models.FloatField(default=0.0)
```

ДОДАТОК Б

КОД PYTHON (файл views.py)

```
from django.contrib import messages
from django.contrib.auth import authenticate, login, logout
from django.contrib.auth.mixins import LoginRequiredMixin
from django.http import HttpResponseRedirect
from django.shortcuts import render, redirect
from django.urls import reverse
from django.views.generic import TemplateView, DetailView,
ListView
from .models import *

NUM_PLAYERS_IN_COMMAND = 4


class IndexView(TemplateView):
    template_name = 'commands/index.html'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['game_list'] =
Game.objects.order_by('created')
        return context


class SettingsView(LoginRequiredMixin, TemplateView):
    template_name = 'commands/settings.html'

    def get_context_data(self, **kwargs):
```

```
        context = super().get_context_data(**kwargs)
        context['gamer_list'] =
User.objects.exclude(username='admin').order_by('first_name'
)
        # print('context', context)
        return context


class PlayerDetailView(DetailView):
    model = User
    template_name = 'commands/player_detail.html'
    context_object_name = 'player'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['num_won'] =
self.object.profile.won_games_list().count()
        context['num_lost'] =
self.object.profile.lost_games_list().count()
        context['num_total'] = context['num_won'] +
context['num_lost']
        # print('context', context)
        return context


class PlayerRatingView(ListView):
    queryset =
User.objects.exclude(username='admin').order_by('-
profile__won_games_ratio')
```

```
    template_name = 'commands/player_rating.html'
    context_object_name = 'players'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        # print('context', context)
        return context


class CoupleRatingView(ListView):
    queryset = Couple.objects.order_by('-weight', '-
num_games')
    template_name = 'commands/couple_rating.html'
    context_object_name = 'couples'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        # print('context', context)
        return context


class SolutionsView(TemplateView):
    template_name = 'commands/solutions.html'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        data, data1, data2 = [], [], []
        pareto_solutions = []
        solution_list = list(Solution.objects.all())
```

```
        for s in solution_list:
            data.append({'x': s.maxsum, 'y': s.maxmin})
        for item1 in data:
            # is_pareto = True
            for item2 in data:
                if item1 == item2:
                    continue
                if item2['x'] >= item1['x'] and item2['y']
>= item1['y']:
                    is_pareto = False
                    data1.append(item1)
                    break
            else:
                data2.append(item1)
                idx = data.index(item1)
                pareto_solutions.append(solution_list[idx])
        context['data1'] = str(data1).replace("'", "")
        context['data2'] = str(data2).replace("'", "")
        context['pareto_solutions'] = pareto_solutions
        # print('context', context)
        return context


def user_sign_in(request):
    username = request.POST.get('username')
    password = request.POST.get('password')
    user = authenticate(request, username=username,
password=password)
    if user:
```

```
        login(request, user)
    return HttpResponseRedirect(reverse("commands:index"))


def user_sign_out(request):
    logout(request)
    return HttpResponseRedirect(reverse("commands:index"))

def create_new_game(request):
    # messages.add_message(request, messages.INFO, 'Hello
world.')

    players1_idx_list = request.POST.getlist('players1')
    players2_idx_list = request.POST.getlist('players2')
    is_winner = request.POST.get('winner')
    print(players1_idx_list)
    print(players2_idx_list)
    print('winner', is_winner)

    if
set(players1_idx_list).intersection(set(players2_idx_list)):
        messages.add_message(request, messages.INFO, 'one
player can not be in two commands')
        return redirect(to='commands:settings')
    elif not len(players1_idx_list) ==
len(players2_idx_list) == NUM_PLAYERS_IN_COMMAND:
        messages.add_message(request, messages.INFO,
f'number of players in commands must be
{NUM_PLAYERS_IN_COMMAND}')
```

```
        return redirect(to='commands:settings')
    elif is_winner is None:
        messages.add_message(request, messages.INFO, 'you
must choose the winner command')
        return redirect(to='commands:settings')
    else:
        new_game = Game()
        new_game.save()
        new_command1 = Command(game=new_game)
        new_command2 = Command(game=new_game)
        if is_winner == '1':
            new_command1.is_winner = True
        elif is_winner == '2':
            new_command2.is_winner = True
        new_command1.save()
        new_command2.save()
        for idx in players1_idx_list:

new_command1.players.add(User.objects.get(id=idx))
        for idx in players2_idx_list:

new_command2.players.add(User.objects.get(id=idx))
        messages.add_message(request, messages.INFO, 'new
game added successfully')

    return redirect(to='commands:index')
```