

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Факультет комп'ютерних наук і технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проекту (роботи)
магістра

на тему: КОМПЛЕКС РЕКОМЕНДАЦІЙ ТА ІНСТРУМЕНТІВ ДЛЯ
ОПТИМІЗАЦІЇ КОНФІГУРАЦІЙ КОМП'ЮТЕРНИХ СИСТЕМ

Виконав: студент 2 курсу, групи КНТ-512м
спеціальності _____

123 Комп'ютерна інженерія

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні системи та мережі

КУДЕРМЕТОВ А. Р.

(ПРИЗВИЩЕ та ініціали)

Керівник ЗЕЛЕНЬОВА І. Я.

(ПРИЗВИЩЕ та ініціали)

Рецензент ЄФИМЕНКО М. В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук і технологій
Кафедра «Комп'ютерні системи та мережі»
Ступінь вищої освіти магістерський
Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
Освітня програма (спеціалізація) «Комп'ютерні системи та мережі»
назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ
Завідувач кафедри Кудерметов Р. К.
« » 2023 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

КУДЕРМЕТОВ Андрій Равільсвич
(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Комплекс рекомендацій та інструментів для оптимізації конфігурацій комп'ютерних систем
керівник проєкту (роботи) к. т. н., доцент, ЗЕЛЕНЬОВА Ірина Яківна
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від «24» жовтня 2023 року №400

2. трок подання студентом проєкту (роботи) 10 грудня 2023 року

3. Вихідні дані до проєкту (роботи) існуючі методи багатокритеріального прийняття рішень, сучасні моделі та характеристики компонентів комп'ютерних систем (КС).

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) огляд та аналіз сучасних тенденцій конфігурування КС;

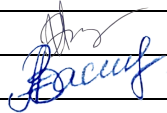
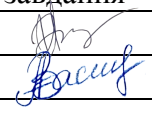
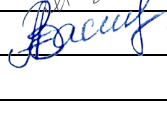
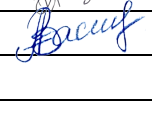
2) огляд методів оптимізації, які б відповідатиме розв'язанню задач конфігурування КС з урахуванням вподобань користувача;

3) Розробка програмних інструментів (системи) підтримки процесу вибору оптимальних конфігурацій КС;

4) Реалізація та апробування розробленої системи на реальному прикладі.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	ЗЕЛЕНЬОВА І. Я., к. т. н, доц.		
нормоконтроль	ПОЛЬСЬКА О.В., ст. викл.		

7. Дата видачі завдання «01» жовтня 2023року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Огляд та аналіз сучасних тенденцій конфігурування КС	10.10.23	
2	Огляд методів векторної оптимізації	1.11.23	
3	Збір та систематизація інформації та фактичних даних про комп'ютерні компоненти	1.11.23	
4	Дослідження способів побудови векторів пріоритетів комп'ютерних компонентів у середовищах MS Excel, Syper Decisions, Python	14.11.23	
5	Розробка алгоритму роботи модулів і системи	14.11.23	
6	Розробка програмних модулів обчислення векторів пріоритетів комп'ютерних компонентів	20.11.23	
7	Розробка головного програмного модуля системи	23.11.23	
8	Складання реального прикладу конфігурування КС та розв'язання його за допомогою розробленої системи	29.11.23	
9	Складання пояснювальної записки	09.12.23	
10	Оформлення допоміжного матеріалу	10.12.23	

Студент  **Андрій КУДЕРМЕТОВ**
(підпис) (Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)  **Ірина ЗЕЛЕНЬОВА**
(підпис) (Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 70 с., 22 рис., 37 табл., 15 джерел

**КОМП'ЮТЕРНА СИСТЕМА, ОПТИМАЛЬНА КОНФІГУРАЦІЯ,
МЕТОД АНАЛІТИЧНОЇ ІЄРАРХІЇ, МНОЖИНА ПАРЕТО, SUPER DECISIONS**

Об'єкт дослідження – конфігурування комп'ютерних систем.

Предмет дослідження – методи та засоби оптимізації вибору комп'ютерних компонентів та конфігурування комп'ютерних систем з урахуванням уподобань користувача.

Мета роботи – розробка рекомендацій щодо конфігурування, пошук та розробка інструментів для підтримки рішень вибору оптимальних конфігурацій комп'ютерних систем.

У першому розділі виконано аналіз вимог до конфігурації комп'ютерних систем, огляд багатокритеріальних методів прийняття рішень та сформульована задача оптимізації конфігурації комп'ютерної системи.

У другому розділі розглянуто застосування Методу Аналітичної Ієрархії та поняття парето-оптимальної множини для вибору оптимальної альтернативи компоненту комп'ютерної системи.

У третьому розділі апробовані інструменти реалізації системи оптимального конфігурування комп'ютерної системи.

У четвертому розділі розроблено систему оптимального конфігурування комп'ютерної системи та продемонстровано її працездатність на конкретному прикладі.

ABSTRACT

Explanatory note to the master's work: 70 p., 22 figures, 37 tables, 15 sources.

COMPUTER SYSTEM, OPTIMAL CONFIGURATION, ANALYTICAL
HIERARCHY PROCESS, PARETO SET, SUPER DECISIONS

The object of study is the configuration of computer systems.

The subject of study is methods and means of optimizing the selection of computer components and configuring computer systems following user preferences.

The purpose of the work is the development of configuration recommendations, search and development of tools to support decisions on selecting the optimal computer system configurations.

In the first chapter, the requirements for the configuration of computer systems were analysed, the multi-criteria decision-making methods were reviewed, and the problem of the computer system configuration optimisation was formulated.

In the second chapter, the application of the Analytical Hierarchy Process method and the Pareto frontier concept for selecting the optimal computer component alternative are discussed.

In the third chapter, tools for implementing a system for computer system optimal configuration are tested.

In the fourth section, a system for computer system optimal configuration selection was developed and demonstrated using a specific example.

ЗМІСТ

Вступ.....	8
1 Визначення проблеми конфігурування КС	11
1.1 Попередні вимоги до конфігурації КС.....	11
1.2 Задача багатокритеріального прийняття рішень.....	13
1.3 Формулювання задач оптимізації.....	18
1.4 Уточнена постановка завдання розробки	22
2 Розробка системи підтримки рішень щодо конфігурації КС	23
2.1 Метод Аналітичної Ієрархії.....	23
2.2 Перевірка узгодженості матриць попарного порівняння.....	29
3 Вибір інструментів для реалізації методу МАІ.....	32
3.1 Використання табличних процесорів для реалізації методу МАІ	32
3.2 Використання пакету Super Decisions	34
3.3 Програма реалізації вибору оптимальної компоненти КС	38
4 Реалізація програми вибору оптимальної конфігурації	40
4.1 Реалізація модулів обчислення цінностей комп'ютерних компонентів	40
4.2 Реалізація головного модуля вибору оптимальної конфігурації КС	51
Висновки	58
Перелік джерел посилання	60
Додаток А. Лістинги коду програми	62

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

KC	– Комп’ютерна система
ОПР	– Особа, яка приймає рішення
MAI	– Метод аналізу ієрархій
AMD	– Advanced Micro Devices
CI	– Consistency Index
CPU	– Central Processing Unit
CR	– Consistency Ratio
ELECTRE	– ELimination Et Choix Traduisant la Realite
HDD	– Hard disk drive
LGA	– Land Grid Array
MCDM	– Multiple-criteria decision-making
PROMETHEE	– Preference Ranking Organisation Method for Enrichment
Evaluations	
RI	– Index of a Random-like Matrix
SAW	– Simple Additive Weighting
TOPSIS	– Technique for Order Preference by Similarity to an Ideal
Solution	
VIKOR	– Visekriterijumsko Kompromisino Rangiranje

ВСТУП

Комп'ютерні системи (КС) зазвичай мають велику кількість компонентів з окремою ціною, що створює велику кількість можливих конфігурацій. Навіть для комп'ютерів офісного або домашнього користування з невисокою вартістю можна створити тисячі різних конфігурацій. Наприклад, якщо є вибір серед двох типів процесорів, п'яти типів материнських плат, чотирьох типів корпусів, чотирьох приводів жорсткого диска, п'яти типів моніторів, трьох типів відеоадаптерів, двох типів мережевих адаптерів та трьох типів блоків живлення, можна створити 14400 конфігурацій. Якщо при цьому враховувати вподобання користувача щодо виробників (брендів), при тому, що кожен із перерахованих компонентів сьогодні виробляється від двох до семи відомих виробників, кількість варіантів можливих конфігурацій може зрости до мільйонів.

Кожна конфігурація КС з одного боку, має певну цінність чи рівень задоволення потреб користувача, з іншого – певну вартість. Необхідно також враховувати, що звичайному користувачеві не завжди легко розібратися у своїх уподобаннях, які можуть бути суперечливими, помилковими, неусвідомленими та змінними залежно від досвіду, порад оточуючих, дослідження відгуків в Інтернеті тощо.

Подібні питання виникають як в окремих користувачів, так і в організаціях, які замовляють великі партії КС. Безумовно, у таких випадках для вирішення питань із відповідними конфігураціями КС залучаються експерти, проте й тут не виключено проблем. Це може бути пов'язано з кваліфікацією експертів, кількістю експертів, методів, що використовуються для колективної експертизи.

Навіть у разі вибору КС, яка зконфігурована спеціалізованими постачальниками, споживачеві хотілося б розуміти, наскільки доброякісно зконфігурована КС, і на її конфігурацію не вплинули кон'юнктурні цілі

постачальника, динамічні коливання ринку, наявність або відсутність найбільш підходящих компонентів на момент конфігурування тощо. Таким чином, актуальним є питання розробки та вивчення методів оптимізації конфігурації КС. Ця магістерська робота присвячена дослідженню сучасних тенденцій конфігурування КС та методів оптимізації їх конфігурацій.

Об'єктом дослідження є конфігурування комп'ютерних систем.

Предметом дослідження є методи та засоби оптимізації вибору комп'ютерних компонентів та конфігурування комп'ютерних систем з урахуванням уподобань користувача.

Метою роботи є розробка рекомендацій щодо конфігурування, пошук та розробка інструментів для підтримки рішень вибору оптимальних конфігурацій комп'ютерних систем.

Для досягнення цієї мети сформульовано такі задачі:

- провести огляд сучасних тенденцій конфігурування КС;
- провести огляд методів оптимізації, які б відповідалиме розв'язанню задач конфігурування КС з урахуванням вподобань користувача;
- розробити чи підібрати програмні інструменти підтримки процесу вибору оптимальних конфігурацій;
- продемонструвати на прикладах результати досліджень та розробки.

Новизна роботи полягає у спільному застосуванні методу аналізу ієрархій для ранжування компонентів КС та альтернатив конфігурацій КС з погляду уподобань користувача та побудови їх парето-оптимальних множин, які можуть використовуватися для кінцевого прийняття рішення вибору конфігурації КС.

Основні результати:

- сформульовано задачі оптимізації, які полягають у визначенні поняття оптимальної конфігурації КС з погляду: максимального співвідношення цінність-вартість; максимальної загальної цінності КС, за якої вартість КС не

перевищує прийнятне для користувача значення; мінімальної загальної вартості КС, за якої цінність КС має заданий користувачем рівень вподобання;

- виконано огляд ринку основних комп'ютерних компонентів;
- виконано огляд та проведено аналіз широко використовуваних методів багатокритеріального прийняття рішень, з яких обрано найбільш підходящий для вирішення поставленого завдання;
- розроблено програмні модулі для визначення вектора пріоритетів порівнюваних альтернатив комп'ютерних компонентів та КС в цілому, та перевірки несуперечності суджень ОПР;
- розроблено програмний модуль, що дозволяє формувати множину можливих альтернатив конфігурацій КС, визначення цінності цих альтернатив у відповідність до вподобань користувача та обчислення повної вартості кожної альтернативи, які використовуються для визначення оптимального варіанту синтезованої КС.

Результати досліджень та розробки можуть бути використані для розробки повнофункціонального програмного продукту системи підтримки прийняття рішення як для споживачів КС та компонентів, так і для організацій, які займаються їх продажем, ремонтом та модернізацією.

1 ВИЗНАЧЕННЯ ПРОБЛЕМИ КОНФІГУРУВАННЯ КС

1.1 Попередні вимоги до конфігурації КС

Оцінка та вибір КС – це проблема вибору найбільш підходящої конфігурації апаратного та програмного забезпечення виходячи з вимог конкретного користувача комп'ютера. Кожна з цих конфігурацій має певну вартість і певний ступінь задоволення вимог користувача. Це непроста проблема, оскільки більшість КС можуть мати велику кількість різних конфігурацій. Однією із складових цієї проблеми є велика кількість компонентів КС.

Враховуючи, що величезна кількість комп'ютерних компонентів пропонуються постачальниками та представлені в електронних каталогах комп'ютерної техніки (наприклад, <https://ek.ua/ua/>), ми вирішили згрупувати комп'ютерні компоненти за їх найбільш популярними виробниками (брендами). На рис. 1.1 представлені ці виробники, типи компонентів та доцільна послідовність комплектування КС (зверху вниз). Ця схема буде відправною точкою для формування альтернативних конфігурацій КС. Формування альтернативних рішень є обов'язковим етапом процесу прийняття рішення.

Відповідно до безлічі рекомендацій з комплектування КС, в першу чергу споживач повинен вибрати тип процесора, потім узгоджену з процесором основну плату, яку прийнято називати материнською платою. Далі можна вибирати інші компоненти, які найчастіше сумісні з популярними материнськими платами. На рис. 1.1 показані основні або найбільш популярні виробники головних компонентів КС, якими вважаються процесор, материнська плата, жорсткий диск, корпус і монітор.

Для конкретизації задачі конфігурування зробимо кілька припущень.

Припущення 1. Припускаємо, що споживач бажає отримати бюджетний комп'ютер для домашнього або офісного застосування і може вибрати один із

процесорів: i3, i5 або i7 компанії Intel; Ryzen 3, Ryzen 5 або Ryzen 7 компанії Advanced Micro Devices (AMD). Процесори повинні включати вбудоване графічне ядро.

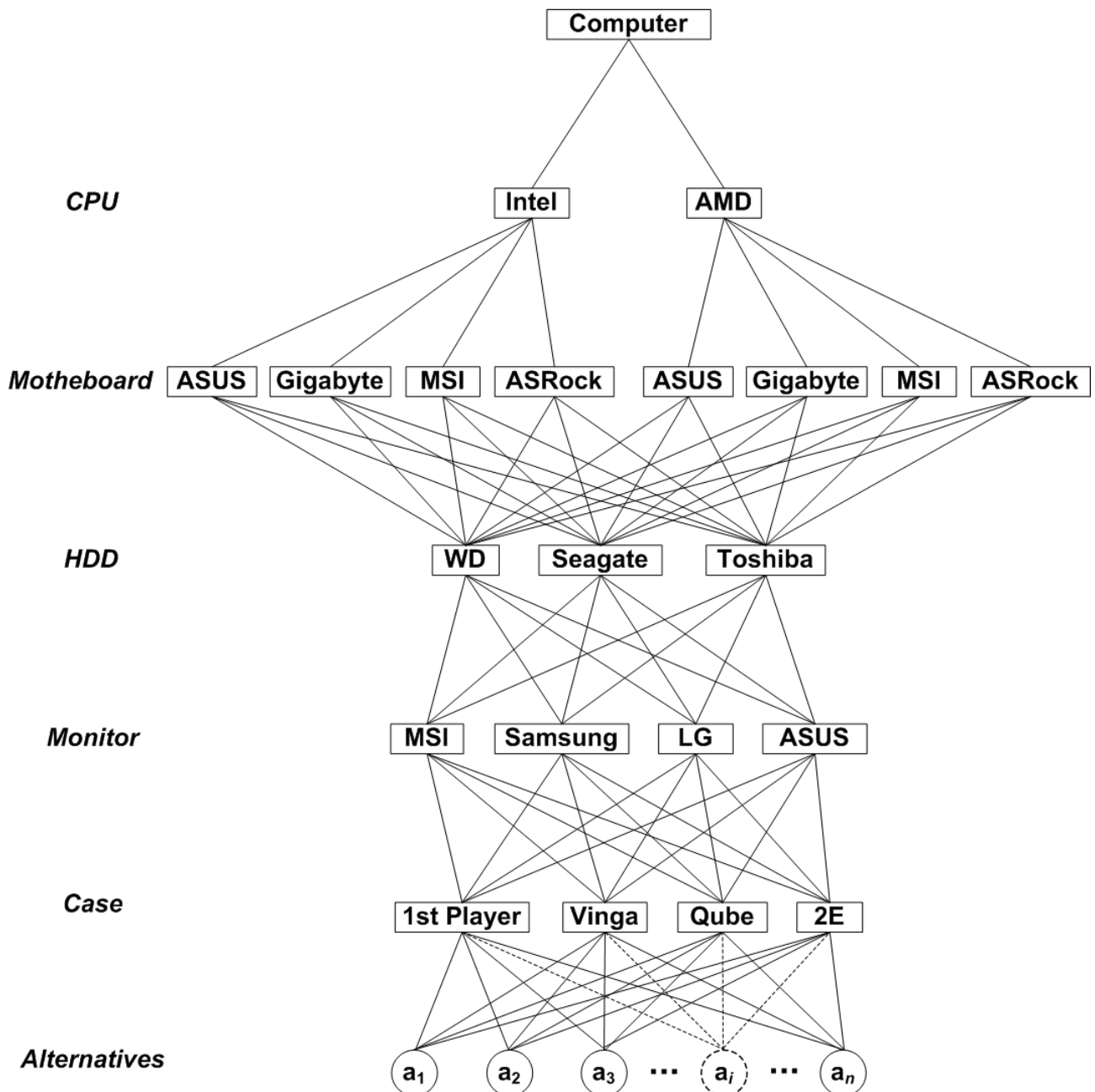


Рисунок 1.1 – Схема вибору компонентів конфігурації

Припущення 2. Постачальники пропонують дуже багато компонентів того самого типу, наприклад, материнські плати одного типу, але кожна з них має деякі особливості. Насамперед передбачається, що тип процесора, сокет та

чіпсет узгоджені [1]. Ми розглядаємо материнські плати для процесорів Intel з сокетом LGA 1200 і чіпсетом B560. Для процесорів AMD материнські плати зазвичай мають сокет AM4 та чіпсети B550, X570. Зазвичай, кожен споживач вміє чи бажає оцінити ці конкретні характеристики материнської плати, наприклад число фаз живлення. З одного боку, можна сказати, що це не важливо, проте з часом виявиться, що система працює ненадійно. Від числа фаз залежить надійність, вартість, можливість прискорення процесора та пам'яті комп'ютера. Для формування альтернатив ми беремо середні характеристики за кількістю фаз (≤ 10).

Припущення 3. Оперативна пам'ять: ми припускаємо, що користувач потім нарощуватиме обсяг оперативної пам'яті (ОЗУ), тому ми включаємо до списку альтернатив материнські плати з чотирма каналами (роз'ємами) для підключення ОЗУ типу DDR4.

Припущення 4. Для підключення жорстких дисків материнська плата повинна мати до чотирьох інтерфейсів SATA 3, і можливість встановлення твердотілого диска (SSD), тобто ми розглянемо варіанти материнської плати з інтерфейсом M2.

1.2 Задача багатокритеріального прийняття рішень

Задача вибору чи синтезу конфігурації КС належить до задач прийняття рішення за багатьох критеріях. Дійсно, при виборі конфігурації у користувача може бути кілька вимог до системи, наприклад, надійність, вартість, дизайн, зручність використання вдома або в офісі, швидкодія, якість графіки, можливість модернізації (апгрейду) і т.д. Зазвичай деякі критерії є суперечливими, наприклад, продуктивність-вартість. Неможливо запропонувати спосіб конфігурування, при якому були б повністю задоволені

всі вимоги до системи. Тому багатокритеріальні задачі оптимізації ще називаються задачами пошуку компромісів. В даній роботі досліджується можливість компромісу між цінністю КС для користувача та вартістю, за якою користувач може придбати цю систему.

На відміну від звичайних задач оптимізації, результат прийняття рішення оцінюється кількома критеріями $c_1, c_2, \dots, c_n$, кожен з яких відповідає певній корисності з погляду якісної або кількісної шкалі. Існує велика кількість різноманітних постановок багатокритеріальних задач. Розглянемо загальну постановку задачі. У нашому випадку ми вважаємо, що кількість можливих альтернатив є кінцевою $D = \{a_1, a_2, \dots, a_m\}$ і корисність кожної їх оцінюється за n критеріями. Тоді можна скласти матрицю корисності альтернатив $U = [u_{ij}]_{m \times n}$, представлену як табл. 1.1.

Таблиця 1.1 – Матриця корисності альтернатив

	c_1	c_2	...	c_n
a_1	u_{11}	u_{12}	...	u_{1n}
a_2	u_{21}	u_{22}	...	u_{2n}
...	...	...	...	...
a_m	u_{m1}	u_{m2}	...	u_{mn}

Рідко буває так, що деяка альтернатива є найкращою відразу за всіма критеріями. Як правило, результат виявляється кращим за одним із критеріїв і поступається за іншими, тому доводиться шукати компроміс. Задача може бути поставлена як пошук однієї «найкращої» в певному сенсі альтернативи або вибір підмножини «прийнятних» альтернатив. Також може бути сформульована задача впорядкування альтернатив за перевагою. Задачу також можна сформулювати як оптимізацію вектора цілей при обмеженнях.

Кількість розроблених методів для прийняття рішень за багатьох критеріях запропоновано дуже багато. Наприклад, у роботі [2] виконано огляд понад сотню методів. Вибір і порівняння методів залежить від конкретної постановки завдання та кваліфікації ОПР. Тут коротко перелічимо

характеристики лише кількох, найпопулярніших методів багатокритеріального прийняття рішень.

Найпростішим і найчастіше використовуваним методом MCDM є метод SAW (Simple Additive Weighting) – метод простого адитивного зважування. ОПР безпосередньо надає відносну важливість кожному критерію оцінки і потім обчислюється загальний рейтинг кожної альтернативи шляхом обчислення суми добутків ваги важливості кожного критерію на відповідне масштабоване значення цього критерію. Далі вибирається альтернатива із найбільшим значенням рейтингу. Недоліком цього є те, що під час використання адитивної згортки можлива компенсація одних важливих критеріїв іншими [2, 3].

Основна ідея методу TOPSIS (Technique for Order Preference by Similarity to Ideal Solution) полягає у визначенні ідеальних позитивних та негативних рішень на основі найкоротшої відстані до ідеального рішення. Ідеальні позитивні рішення – це гіпотетичні рішення, в яких всі значення критеріїв максимальні. Ідеальні негативні рішення – це гіпотетичні рішення, в яких всі значення критеріїв мінімальні [4].

Метод VIKOR (Visekriterijumsko Kompromisino Rangiranje) схожий метод TOPSIS, у ньому також альтернативи порівнюються з ідеальним рішенням. Метод VIKOR орієнтований на ранжування та вибір за різними варіантами та визначає компромісні вирішення проблеми з несумісними критеріями. За допомогою цього методу ОПР можуть дійти остаточного рішення. Компромісне рішення може бути ближчим до ідеального рішення, а компромісом є угода про двосторонній обмін. Особливість методу VIKOR полягає в тому, що компромісне рішення відображає позицію більшості ОПР [3,4].

Метод аналізу ієрархій (MAI), англomовна аббревіатура АНР (Analytic Hierarchy Process) використовується для визначення компромісного рішення, яке задовольняє вимогам ОПР [5, 6]. Цей метод складається з наступних етапів:

- спочатку визначається мета, яку планується досягти за допомогою якнайкращого вибору;
- потім визначаються критерії, що дозволяють оцінити, наскільки можливі альтернативи задовольняють цілі. Критерії порівнюються попарно, виходячи з оцінок формується матриця пріоритетів критеріїв. Кожному критерію в результаті оцінки важливості надається певна вага;
- далі розглядаються альтернативи, які попарно порівнюються за кожним критерієм окремо. На підставі порівнянь формуються матриці пріоритетів альтернатив за кожним критерієм;
- остаточні оцінки альтернатив складаються з оцінок критеріїв та оцінок альтернатив за критеріями.

Сімейство методів ELECTRE (ELimination Et Choix Traduisant la Realite – виключення та вибір, що відображають реальність) включає кілька модифікацій (ELECTRE, ELECTRE I – ELECTRE IV, ELECTRE A, ELECTRE TRI) [2,4]. Метод ELECTRE базується на індексах згоди та незгоди (concordance и discordance). Прийняття рішення здійснюється в наступній послідовності:

- визначаються значення попарного порівняння альтернатив для матриць індексів згоди та незгоди;
- вибирається максимальне значення для матриці індексів згоди та мінімальне значення для матриці індексів незгоди.
- якщо для пари альтернатив індекс згоди більший або дорівнює рівню згоди, а індекс незгоди менший або дорівнює рівню незгоди, то альтернатива в рядку домінує над альтернативою в стовпці. Домінована альтернатива виключається з розгляду, і матриці згоди та незгоди коригуються на виключену альтернативу;
- якщо таких пар не виявилось, то рівень згоди знижують, а рівень незгоди підвищують. І так доти, доки не залишилася одна альтернатива, яка домінує над усіма, або незрівнянні альтернативи, які становлять ядро найкращих альтернатив.

Труднощі при застосуванні методу ELECTRE пов'язані з призначенням ОПР ваг.

Автор методу PROMETHEE (Preference Ranking Organisation METHod for Enrichment Evaluations) Дж. Бранс (J. Brans) запропонував будувати відносини ранжирування, що включають лише реалістичні збагачення відносини домінування. В цьому випадку не всі розбіжності виключені, але інформація є достовірною. Методи PROMETHEE належать до класу методів випередження. При розробці методу враховувалися такі вимоги [5]:

- слід враховувати амплітуду відхилень між оцінками альтернатив у межах кожного критерію. Цю інформацію легко вирахувати, але вона не враховується в теорії ефективності. Коли ці відхилення незначні, відносини домінування можуть бути збагачені;

- оскільки оцінки кожного критерію виражаються в окремих одиницях, ефекти масштабування мають бути повністю усунені. Неприпустимо отримувати висновки залежно від шкал, у яких виражені оцінки;

- у разі парних порівнянь відповідний багатокритеріальний метод повинен надавати таку інформацію: a_1 краще a_2 ; a_1 та a_2 не відрізняються; a_1 і a_2 непорівнянні, де a_1 і a_2 – альтернативи, що порівнюються. Мета полягає в тому, щоб максимально скоротити кількість незрівнянностей, але не тоді, коли це нереально;

- метод має бути зрозумілим для ОПР;

- алгоритм методу має надавати інформацію про суперечливий характер критеріїв;

- більшість багатокритеріальних методів розподіляють ваги щодо відносної важливості критеріїв. Ці ваги, як правило, дуже залежать від переваг ОПР та виправити їх непросто. Зазвичай особи, які приймають рішення, сильно вагаються. Метод повинен пропонувати інструменти чутливості, що дозволяють тестувати різні набори ваги.

Методи PROMETHEE, та його модифікації (PROMETHEE I – PROMETHEE VI, PROMETHEE TRI та PROMETHEE CLUSTER) та пов'язаний з ними візуальний інтерактивний модуль GAIA враховують усі ці вимоги. З іншого боку, можна також враховувати деякі математичні властивості, якими можуть мати багатокритеріальні задачі. Такі властивості, пов'язані з методами PROMETHEE, проаналізовано в [7].

1.3 Формулювання задач оптимізації

Отже, вважатимемо, що оптимальна конфігурація комп'ютера залежить від двох основних критеріїв: загальної вартості та загальної цінності кожної конфігурації, що розглядається. Під показником загальної цінності розумітимемо глобальне вподобання користувача. У роботі [8] пропонується під показником цінності розуміти відсоток виконаних вимог користувача до системи, що оцінюється. У цій роботі запропоновано три варіанти мети оптимізації:

- максимізація загальної цінності за обмеженої вартості системи;
- мінімізація загальної вартості, доки загальна цінність перевищує задану потрібну цінність;
- максимізація співвідношення цінність/вартість.

Аби вирішити ці задачі, передусім, необхідно створити кількісну модель розрахунку цінності конфігурацій КС. Для цього сформуємо впорядковане дерево, яке включатиме варіанти компонентів, що утворюють кожну з можливих альтернатив конфігурації КС. Очевидно, багато варіантів компонентів не дозволяє наочно графічно зобразити таке дерево. Фрагмент такого дерева показано на рис. 1.2. Для обчислення вартості кожної альтернативи необхідно обчислити суми вартості всіх компонентів, що входять

до конфігурації. Це зробити досить легко. Для визначення цінності кожного компонента та глобальної цінності кожної альтернативної конфігурації необхідно запропонувати метод, що враховує вподобання користувача.

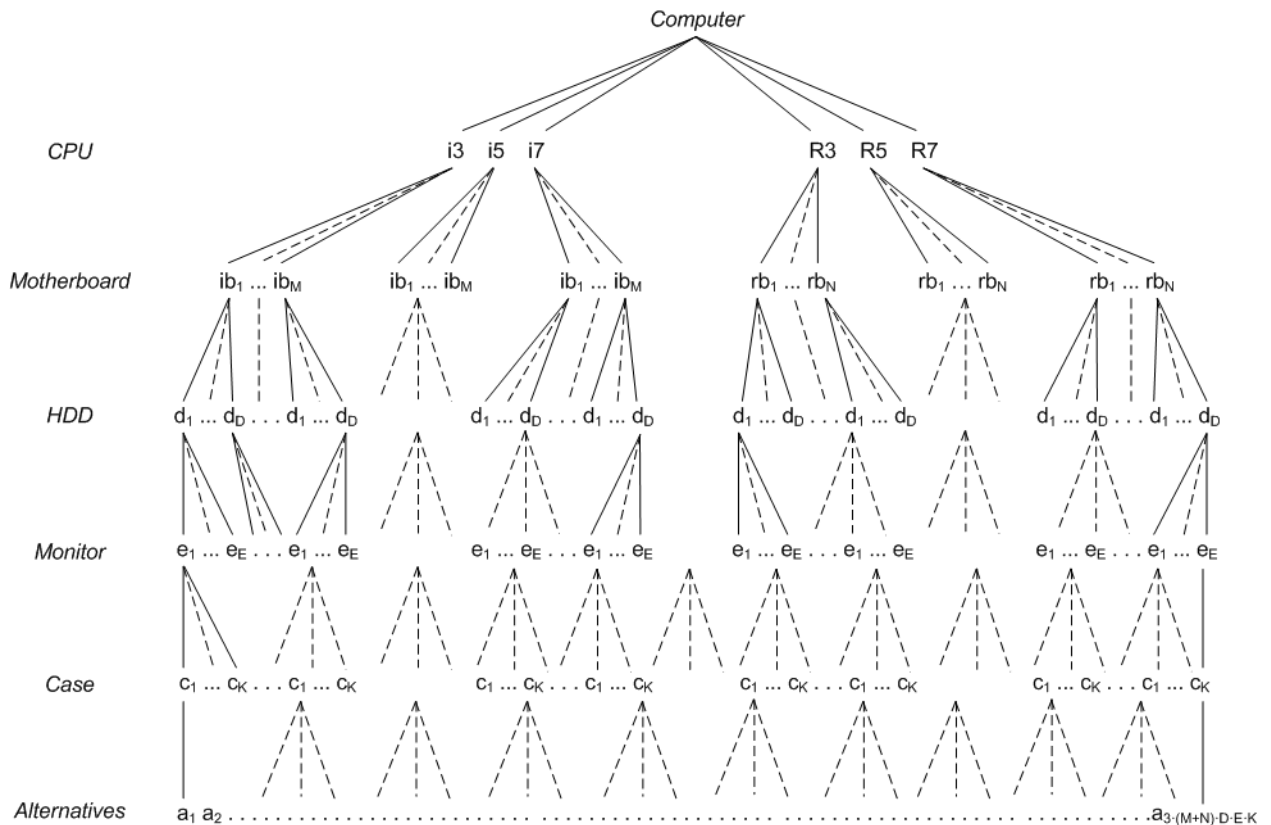


Рисунок 1.2 – Дерево альтернативних конфігурацій

Для цього в даній роботі пропонується використовувати Метод Аналітичної Ієрархії (MAI) [2]. Цей метод буде детально розглянутий у наступному розділі.

Припустимо, що ми визначили цінність і вартість кожної конфігурації. Усі можливі конфігурації можна представити як K точок на площині цінність-вартість показаної рис. 1.3. Очевидно, що одні конфігурації кращі, а інші менш кращі. Розглядаючи сформульовані вище три задачі оптимізації, можна на площині цінність-вартість позначити три напрямки. Напрямок, позначений вертикальною стрілкою, це напрям максимізації цінності. Напрямок, позначений горизонтальною стрілкою, – це напрям мінімізації вартості.

Оптимізація, спрямована на задоволення одночасно цінності та вартості, показано діагональною стрілкою.

Кожна конфігурація характеризується масивами координат $\{C_1, C_2, \dots, C_K\}$, $\{V_1, V_2, \dots, V_K\}$. У прийнятих позначеннях три задачі оптимізації можуть бути записані таким чином:

Максимальна цінність конфігурації – це найбільш підходяща конфігурація, вартість якої менша за задану максимально доступну межу:

$$V_{\max} = \max_{C \leq C_{\max}} (V_1, V_2, \dots, V_K).$$

Мінімальна вартість конфігурації – це найдешевша конфігурація, яка досягає мінімальної допустимої цінності:

$$C_{\min} = \min_{V \geq V_{\min}} (C_1, C_2, \dots, C_K).$$

Найкраща конфігурація – це конфігурація, яка досягає найвищого співвідношення V/C :

$$S_{\text{opt}} = \max \left(\frac{V_1}{C_1}, \frac{V_2}{C_2}, \dots, \frac{V_K}{C_K} \right).$$

На рис. 1.3 чорними точками зазначено альтернативи, які називаються парето-оптимальними (оптимальними за Парето, ефективними). Парето-оптимальні альтернативи – це такі альтернативи для яких у нашому випадку не існує інших альтернатив, які за всіма параметрами кращі. Парето-оптимальні альтернативи – це такі альтернативи, для яких серед інших альтернатив не знайдеться жодної, яка була б кращою за вартістю, а за іншим критерієм –

цінності була б не гіршою, а також не знайдеться жодної альтернативи, яка була кращою за цінністю та не гірше за вартістю.

Таким чином, оптимальна конфігурація КС буде такою конфігурацією, для якої ми не можемо зменшити вартість без зменшення цінності, і не можемо збільшити цінність, не збільшивши вартість. Користувач змушений йти на компроміс, погоджуючись на деяке зменшення цінності, якщо хоче зменшити вартість конфігурації. Згідно з принципом Еджворта-Парето, якщо користувач поводить себе “розумно”, то він повинен вибирати конфігурацію системи тільки серед парето-оптимальних альтернатив.

На рис. 1.3 показано оптимальну альтернативу, вона має максимальний кут на площині цінність-вартість (третя задача оптимізації). Для вибору альтернативи з максимальною цінністю (перша задача оптимізації) слід провести вертикальну пряму на рівні максимальної граничної вартості та вибрати парето-оптимальну альтернативу ліворуч від цієї прямої, яка має максимальну цінність. Для вибору альтернативи з мінімальною вартістю (друга задача оптимізації) слід на площині провести горизонтальну пряму на рівні допустимої мінімальної цінності та вибрати найближчу над цією прямою парето-оптимальну альтернативу.

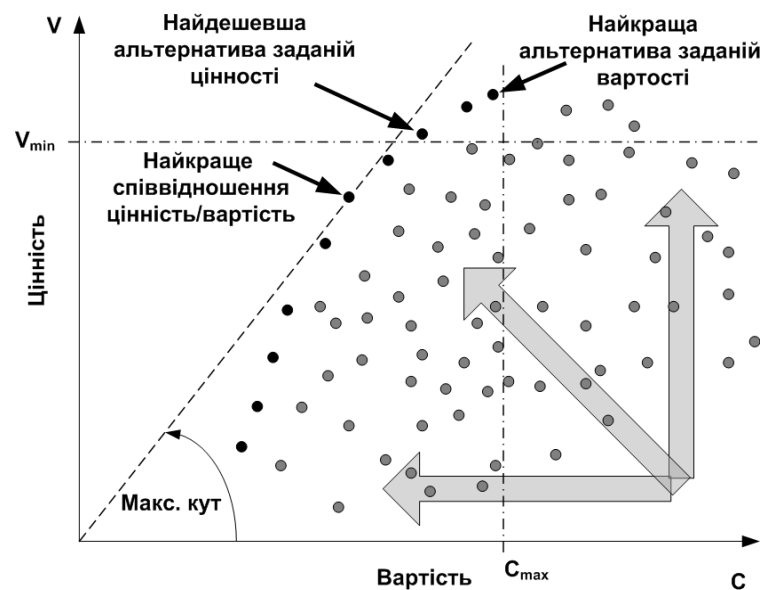


Рисунок 1.3 – Альтернативи на площині цінність-вартість

Слід розуміти, що остаточне рішення щодо вибору конфігурації КС, як і в інших випадках вибору альтернативи, приймає користувач або особа яка приймає рішення (ОПР). Незважаючи на отримані результати оптимізації, користувач може мати власні міркування щодо відповідної йому конфігурації КС. Він може отримати додаткову інформацію про ті чи інші компоненти системи, а також скористатися рекомендаціями експертів. Вибір єдиного остаточного рішення є дуже складною задачею і навіть одна з неефективних альтернатив може виявитися найкращою у конкретній ситуації прийняття рішення. Методи, що дозволяють оцінити ефективність альтернативи, можуть бути корисними для отримання числових значень ефективності в тому чи іншому сенсі, остаточний вибір альтернативи стоїть за ОПР.

1.4 Уточнена постановка завдання розробки

У результаті розгляду питань конфігурування КС ухвалено такі рішення:

- для визначення рейтингу цінностей комп'ютерних компонентів і КС у цілому застосувати метод MAI, оскільки він найпростіше дозволяє виконати декомпозицію задачі визначення цінності конфігурації на більш прості задачі визначення цінностей компонентів КС; метод не вимагає визначення від користувача ваг альтернатив, навпаки цей метод часто використовується для визначення ваг альтернатив для інших методів, наприклад, SAW, TOPSIS, ELECTRE тощо [3];

- оскільки при конфігуруванні КС користувач найчастіше оцінює систему з погляду цінності-вартості (якості-ціни), ми будемо обчислювати значення відносини цінність-вартість для всіх можливих конфігурацій і будувати множину парето-оптимальних альтернатив конфігурацій, надаючи цю множину користувачеві для прийняття остаточного рішення;

- корисною є також побудова парето-оптимальних множин для всіх основних компонентів КС, що може допомогти користувачеві при апгрейді системи шляхом заміни окремих компонентів;
- виконати експерименти з прикладу вибору окремого компонента системи (жорсткого диска) з використанням кількох програмних середовищ для остаточного вибору інструмента розробки.

2 РОЗРОБКА СИСТЕМИ ПІДТРИМКИ РІШЕНЬ ЩОДО КОНФІГУРАЦІЇ КС

2.1 Метод Аналітичної Ієрархії

Суть методу розглянемо з прикладу вибору однієї з компонентів КС – жорсткого диска (HDD). Нехай метою є купівля нового жорсткого диска. Користувачеві запропонували кілька моделей жорсткого диска. Найбільш значущі характеристики користувача запропонованих моделей наведені в табл. 2.1. Розглянуті HDD мають однаковий форм-фактор (3,5 дюйми) та інтерфейс (SATA III).

Щоб проаналізувати рішення щодо купівлі HDD з використанням Методу аналітичної ієрархії (MAI), необхідно виконати наступні кроки [9, 10].

Крок 1. Розробити модель рішення: розбити рішення на ієрархію цілей, критеріїв та альтернатив.

Крок 2. Визначити пріоритети (ваги) критеріїв:

- важливість критеріїв попарно порівнюючи з бажаною метою отримання їх ваг;
- перевірити несуперечність суджень, тобто переглянути рішення з погляду розумного рівня узгодженості;
- визначити локальні пріоритети (переваги) для альтернатив, тобто отримати пріоритети альтернатив за кожним критерієм окремо.

Крок 3. Визначити глобальні пріоритети (синтез моделі): усі отримані альтернативні пріоритети об'єднуються у вигляді виваженої суми для врахування ваги кожного критерію. Альтернатива з найвищим загальним пріоритетом є найкращим вибором.

Крок 4. Прийняти остаточне рішення: виходячи з результатів синтезу.

Таблиця 2.1 – Моделі запропонованих жорстких дисків

Виробник	Найменування моделі	Ємність, Тб	Швидкість обертання, об/хв	Ємність буферу, Мб	Вартість, грн.
Western Digital	WD5000AVDS	0,5	5400	32	920
	WD10EZEX	1	7200	64	1700
	WD20EZBX	2	7200	256	2400
	WD40EZZ	4	5400	256	3850
Seagate	ST5000M002	0,5	7200	16	900
	ST1000DM010	1	7200	64	1650
	ST2000DM008	2	7200	256	1900
	ST4000DM004	4	5400	256	3600
Toshiba	HDWD105UZSVA	0,5	7200	64	860
	HDWD110UZSVA	1	7200	64	1300
	HDWD320UZSVA	2	7200	256	2200
	HDWD240UZSVA	4	7200	128	3350

Процес аналітичної ієрархії структурує проблему як ієрархію. На рис. 2.1 показано ієрархію, запропоновану для нашого прикладу. Перший рівень ієрархії у нашому прикладі – це покупка жорсткого диска. Другий рівень в ієрархії складається з критеріїв, які ми будемо використовувати для ухвалення рішення про покупку. У нашому прикладі ми вирішили використати 4 критерії: ємність жорсткого диска, бренд (виробника), статистику продажів жорстких дисків відповідної ємності та статистику продажів відповідних брендів. Третій рівень складається з альтернатив запропонованих жорстких дисків (табл. 2.1).

Переваги такої ієрархічної декомпозиції є очевидними. Структуруючи проблему таким чином, краще зрозуміти рішення, яке необхідно прийняти, критерії, які слід використовувати, і альтернативи, які необхідно оцінити. Цей крок має вирішальне значення, і саме тут у складніших проблемах можна

запросити участь експертів, щоб переконатися, що всі критерії та можливі альтернативи були враховані. Також у складних задачах може бути необхідним включити до ієрархії додаткові рівні, такі як підкритерії.

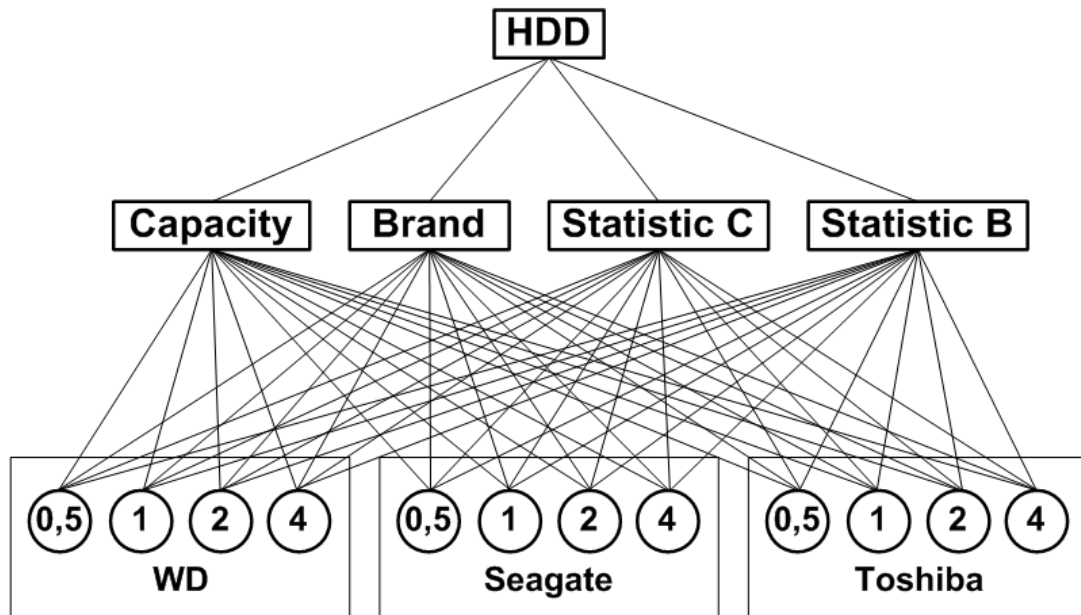


Рисунок 2.1 – Ієрархія рішення при виборі жорсткого диска

Не всі критерії матимуть однакове значення. Отже, другим кроком у процесі МАІ є визначення відносних пріоритетів (ваг) критеріїв. Вони називаються відносними, тому що пріоритети отриманих критеріїв вимірюються відносно один одного.

Важливість (вага) кожного критерію буде різною, тому спочатку необхідно шляхом попарних порівнянь вивести відносний пріоритет кожного критерію стосовно кожного з інших, використовуючи числову шкалу для порівняння, розроблену Т. Сааті [6] (табл. 2.2). Слід відзначити, що існують інші шкали парних порівнянь, наприклад, сбалансована шкала, яка запропонована Pöhjönen, шкала запропонована Ji i Jian [10].

Незалежно від того, скільки факторів бере участь у прийнятті рішення, метод МАІ вимагає порівняння лише пари елементів. У цьому полягає перевага методу, тому що два фактори легше порівняти, ніж одночасно кілька факторів.

Таблиця 2.2 – Шкала Т. Сааті

Вербальне (словесне) судження	Числове значення
Надзвичайна важливість (дуже велика перевага)	9
–	8
Дуже сильна важливість (значна перевага)	7
–	6
Сильна важливість (суттєва перевага)	5
–	4
Помірна важливість (перевага)	3
–	2
Однакова важливість	1
8, 6, 4, 2 – проміжні значення, коли необхідний компроміс	

Нехай для конкретного користувача відносна важливість кожного з критеріїв набуває значень, наведених у табл. 2.3. Таку таблицю в методі МАІ називають матрицею попарного порівняння.

Таблиця 2.3 – Матриця попарного порівняння критеріїв

Criteria	Capacity	Brand	Capacity Stat.	Brand Stat.
Capacity	1	4	2	3
Brand	0,25	1	0,5	0,75
Capacity Stat.	0,5	2	1	1,5
Brand Stat	0,33	1,33	0,67	1

Використовуючи матрицю попарних порівнянь, обчислюється вектор пріоритетів або ваги кожного їх критеріїв. Для цього є два методи: точний і наближений. Для пояснення методу використовуватимемо наближений метод. Наближений метод у тому, що спочатку виконується нормалізація матриці, та усереднюються значення матриці по рядкам. Ці значення утворюють вектор пріоритетів.. Зазначимо, що є кілька методів нормалізації матриці. Найбільш простий метод такий: розділимо кожен елемент на суму елементів стовпця, в якому елемент знаходиться. Нормалізована матриця для табл. 2.3 показана у табл. 2.4. Очевидно, що середні значення кожного рядка утворюють вектор (0,48; 0,12; 0,24; 0,16), з якого випливає, що користувач надає більше значення ємності, за яким слідує статистика продажів жорстких дисків відповідних

ємностей і статистика продажів жорстких дисків від відповідних виробників Критерій «Бренд» отримав для цього споживача найменший пріоритет.

Таблиця 2.4 – Нормалізована матриця попарного порівняння критеріїв

Критерії	Capacity	Brand	Capacity S	Brand S
Capacity	0,48	0,48	0,48	0,48
Brand	0,12	0,12	0,12	0,12
Capacity S	0,24	0,24	0,24	0,24
Brand S	0,16	0,16	0,16	0,16

Наступним кроком методу МАІ є визначення відносних пріоритетів альтернатив щодо кожного критерію. Ці пріоритети називають локальними. Для цього проведемо попарні порівняння всіх альтернатив щодо кожного критерію. Враховуючи, що запропоновані чотири моделі (0,5 Тб; 1 Тб; 2 Тб і 4 Тб) від кожного з трьох виробників (Western Digital; Seagate і Toshiba), матриці попарних порівнянь можуть бути представлені табл. 2.5, 2.6, 2.7 і 2.8.

Таблиця 2.5 – Попарне порівняння за критерієм «Ємність»

Ємність	0,5 Тб	1 Тб	2 Тб	4 Тб
0,5 Тб	1	0,2	0,25	0,33
1 Тб	5	1	1,25	1,67
2 Тб	4	0,8	1	1,33
4 Тб	3	0,6	0,75	1

Таблиця 2.6 – Попарне порівняння за критерієм «Бренд»

Бренд ст.	WD	Seagate	Toshiba
WD	1	0,75	1,5
Seagate	1,33	1	2
Toshiba	0,67	0,5	1

Таблиця 2.7 – Попарне порівняння обсягів продажів жорстких дисків за ємностями

Ємність ст.	0,5 Тб	1 Тб	2 Тб	4 Тб
0,5 Тб	1	0,4	1	2
1 Тб	2,5	1	2,5	5
2 Тб	1	0,4	1	2
4 Тб	0,5	0,2	0,5	1

Таблиця 2.8 – Попарне порівняння обсягів продажів жорстких дисків за брендів

Бренд ст.	WD	Seagate	Toshiba
WD	1	0,75	1,5
Seagate	1,33	1	2
Toshiba	0,67	0,5	1

Нижче наведено обчислені у відповідність до вище викладеної методикою вектори пріоритетів (табл. 2.9. та табл. 2.10).

Таблиця 2.9 – Вектори пріоритетів щодо ємностей жорстких дисків

Ємність	Пріоритет користувача	Статистика продажів
0,5 Тб	0,077	0,2
1 Тб	0,38	0,5
2 Тб	0,31	0,2
4 Тб	0,23	0,1

Таблиця 2.10 – Вектори пріоритетів щодо виробників жорстких дисків

Бренд	Пріоритет користувача	Статистика продажів
Western Digital	0,63	0,333
Seagate	0,21	0,444
Toshiba	0,16	0,222

На наступному етапі необхідно визначити глобальні пріоритети кожної альтернативи. Ці пріоритети враховують локальні пріоритети та пріоритети користувача щодо критеріїв (вектор). Вище ми отримали цей вектор:

$$P = (0,477; 0,128; 0,256; 0,148).$$

Користуючись даними табл. 2.9 та 2.10, запишемо зважену суму локальних пріоритетів у вигляді добутку матриці на вектор у наступному вигляді (див. рис.2.2). Тут виконується множення матриці розмірністю 8×4 , складеної зі стовпців чисел лівої таблиці, на вектор розмірністю 4, складений із

чисел середньої таблиці. В результаті ми отримали глобальні пріоритети кожної альтернативи (права таблиця).

Alternative	Capacity	Brand	Capacity Stat.	Brand Stat.	×	Criteria	=	Priority
WD5000AVDS	0,077	0,63	0,2	0,33		0,477		0,21
WD10EZEX	0,38	0,63	0,5	0,33		0,128		0,43
WD20EZBX	0,31	0,63	0,2	0,33		0,256		0,32
WD40EZA	0,23	0,63	0,1	0,33		0,148		0,26
ST5000M002	0,077	0,21	0,2	0,44	×		=	0,18
ST1000DM010	0,38	0,21	0,5	0,44				0,40
ST2000DM008	0,31	0,21	0,2	0,44				0,29
ST4000DM004	0,23	0,21	0,1	0,44				0,23
HDWD105UZSVA	0,077	0,16	0,2	0,22				0,14
HDWD110UZSVA	0,38	0,16	0,5	0,22				0,36
HDWD320UZSVA	0,31	0,16	0,2	0,22				0,25
HDWD240UZSVA	0,23	0,16	0,1	0,22				0,19

Рисунок 2.2 – Обчислення вектора глобальних пріоритетів

2.2 Перевірка узгодженості матриць попарного порівняння

Якщо матриця попарних порівнянь складається з урахуванням суджень, її необхідно перевірити на консистентність (узгодженість). Справа в тому, що судження виводяться з суб'єктивних уподобань споживача (експерта), що в свою чергу може призвести до деяких невідповідностей (неконсистентності) матриці попарних порівнянь. Консистентність можна також інтерпретувати як ступінь несуперечності суджень. У методі MAI для перевірки консистентності обчислюється коефіцієнт узгодженості (CR) порівняння індексу узгодженості (CI) з індексом узгодженості випадкової матриці (RI). Випадкова матриця – це матриця, в якій судження вводяться випадковим чином, тому очікується, що вона буде непослідовною [2, 4]. Індекс RI був обчислений на випадкових матрицях різного розміру Т. Сааті [4]. У табл. 2.11 наведено кілька таких індексів, де n – це розмір матриці. Т. Сааті показав, що для методу MAI,

прийнятним значенням коефіцієнта RI є значення 0,1. Якщо коефіцієнт узгодженості більший за 0,1, то вважається, що експерт вибирав важливість порівнюваних об'єктів випадковим чином. У цьому випадку експерту слід переглянути свої судження.

Таблиця 2.11 – Індеси RI

n	3	4	5	6	7	8	9	10
RI	0,58	0,9	1,12	1,24	1,32	1,41	1,45	1,49

Хоча задача визначення консистентності може бути легко вирішена за допомогою математичних методів та комп'ютерів, її інтерпретація залишається складною для практиків. Тому наведемо один із спрощених методів обчислення коефіцієнта узгодженості матриці парних порівнянь. Для прикладу візьмемо нашу матрицю парних порівнянь критеріїв (табл. 2.3). Матриця в табл. 2.3 є консистентною, тому для демонстрації ступеня неузгодженості змінимо значення деяких комірок таблиці, саме у комірках [2,4] і [3,4]. В результаті отримаємо матрицю, показану в таблиці 2.12.

Таблиця 2.12 – Матриця попарного порівняння критеріїв

Крітерії	Capacity	Brand	Capacity S	Brand S
Capacity	1	4	2	3
Brand	0,25	1	0,5	1
Capacity S	0,5	2	1	2
Brand S	0,33	1,33	0,67	1

Алгоритм перевірки консистентності складається з наступних кроків.

Крок 1. Виконаємо нормалізацію матриці (табл. 2.12).

Таблиця 2.13 – Нормалізація матриці парних порівнянь

Крітерії	Capacity	Brand	Capacity S	Brand S
Capacity	0,48	0,5	0,5	0,428571
Brand	0,12	0,125	0,125	0,142857
Capacity S	0,24	0,25	0,25	0,285714
Brand S	0,16	0,125	0,125	0,142857

Крок 2. Обчислимо вектор пріоритетів (P), значеннями елементів вектора будуть середні значення рядків нормалізованої матриці:

$$P = (0,477; 0,128; 0,256; 0,148).$$

Крок 3. Помножимо матрицю парних порівнянь (табл. 2.12) на вектор пріоритетів, результатом буде вектор, який позначимо як B :

$$B = (1,917; 0,514; 1,028; 0,554).$$

Крок 4. Кожен елемент вектора B , розділимо на відповідне значення вектора пріоритетів, отримаємо:

$$L = (4,019; 4,008; 4,008; 4,006).$$

Крок 5. Обчислимо середнє значення елементів цього вектора, в результаті отримаємо значення консистентності, яке прийнято позначати $\lambda_{\max}$. Таким чином $\lambda_{\max} = 4,01$.

Крок 6. Індекс консистентності (CI) обчислюється за формулою, яку запропонував Т. Саати:

$$CI = \frac{\lambda_{\max} - n}{n - 1} = \frac{4,01 - 4}{3} \approx 0,003455,$$

де n – розмір матриці.

Крок 7. Використовуючи значення індексу випадкової матриці із табл 2.11 ($RI = 0,9$), обчислимо коефіцієнт консистентності:

$$CR = \frac{CI}{RI} = \frac{0,003455}{0,9} \approx 0,004.$$

Таким чином, ступінь узгодженості матриці, наведеної в табл. 2.12, є допустимим ($0,004 < 0,1$) у методі МАІ. Зазначимо, що вихідна матриця, наведена в табл. 2.3, є консистентною, її коефіцієнт консистентності дорівнює нулю.

3 ВИБІР ІНСТРУМЕНТІВ ДЛЯ РЕАЛІЗАЦІЇ МЕТОДУ МАІ

3.1 Використання табличних процесорів для реалізації методу МАІ

Як ми вже побачили, у методі МАІ багато операцій виконується із таблицями (матрицями). Тому логічно розглянути використання табличного процесора для вирішення задач методом МАІ. табличний процесор (або електронна таблиця) – це прикладна програма, призначена для організації табличних обчислень на комп'ютері. До сучасних табличних процесорів можна віднести MS Excel, SuperCalc, LibreOffice Calc, GNumersc із пакета GNOME Office тощо. На рис. 3.1 наведено хід розв'язання задачі вибору жорсткого диска, згодні вище наведеним вихідним даним за допомогою пакета MS Excel, чотири матриці парних порівнянь HDD і матриця парних порівнянь критеріїв.

Практично всі операції, необхідні реалізації методу МАІ, можна виконати з допомогою MS Excel. Ми використовували MS Excel для попередньої перевірки поставленої задачі оптимізації конфігурації КС, перевірки консистентності матриць попарного порівняння, для аналізу ринку компонентів КС та їх виробників, побудови діаграм рейтингів, а також формування таблиць вихідних даних, що містять характеристики комп'ютерних компонентів та їх вартості.

Результати, отримані з допомогою MS Excel, використовувалися для верифікації результатів, отриманих з допомогою інших обчислювальних програм (див. нижче).

Capacity	0,5	1	2	3		Priority	Normalized				
0,5	1	0,2	0,25	0,333333		0,076923	0,076923	0,076923	0,076923	0,076923	
1	5	1	1,25	1,666667		0,384615	0,384615	0,384615	0,384615	0,384615	
2	4	0,8	1	1,333333		0,307692	0,307692	0,307692	0,307692	0,307692	
3	3	0,6	0,75	1		0,230769	0,230769	0,230769	0,230769	0,230769	
sum	13	2,6	3,25	4,333333							
Brand	WD	Seagate	Toshiba			Priority	Normalized				
WD	1	3	4			0,631579	0,631579	0,631579	0,631579		
Seagate	0,333333	1	1,333333			0,210526	0,210526	0,210526	0,210526		
Toshiba	0,25	0,75	1			0,157895	0,157895	0,157895	0,157895		
sum	1,583333	4,75	6,333333								
Capacity Stat.	0,5	1	2	3		Priority	Normalized				
0,5	1	0,4	1	2		0,2	0,2	0,2	0,2	0,2	
1	2,5	1	2,5	5		0,5	0,5	0,5	0,5	0,5	
2	1	0,4	1	2		0,2	0,2	0,2	0,2	0,2	
3	0,5	0,2	0,5	1		0,1	0,1	0,1	0,1	0,1	
sum	5	2	5	10							
Brand Stat	WD	Seagate	Toshiba			Priority	Normalized				
WD	1	0,75	1,5			0,333333	0,333333	0,333333	0,333333		
Seagate	1,333333	1	2			0,444444	0,444444	0,444444	0,444444		
Toshiba	0,666667	0,5	1			0,222222	0,222222	0,222222	0,222222		
sum	3	2,25	4,5								
Criteria	Capacity	Brand	Capacity Stat.	Brand Stat		Priority	Normalized				
Capacity	1	4	2	3		0,48	0,48	0,48	0,48	0,48	
Brand	0,25	1	0,5	0,75		0,12	0,12	0,12	0,12	0,12	
Capacity Stat.	0,5	2	1	1,5		0,24	0,24	0,24	0,24	0,24	
Brand Stat	0,333333	1,333333	0,666667	1		0,16	0,16	0,16	0,16	0,16	
sum	2,083333	8,333333	4,166667	6,25							
Alternative	Capacity	Brand	Capacity Stat.	Brand Stat		Criteria		Priority	Priority, %	Rank	
WD5000AVDS	0,076923	0,631579	0,2	0,333333		0,48		0,214046	49,3491	9	
WD10EZEX	0,384615	0,631579	0,5	0,333333		0,12		0,433738	100	1	
WD20EZBX	0,307692	0,631579	0,2	0,333333		0,24		0,324815	74,88737	4	
WD40EZA	0,230769	0,631579	0,1	0,333333		0,16		0,263892	60,84132	6	
ST5000M002	0,076923	0,210526	0,2	0,444444				0,181297	41,7988	11	
ST1000DM010	0,384615	0,210526	0,5	0,444444	x		=	0,40099	92,4497	2	
ST2000DM008	0,307692	0,210526	0,2	0,444444				0,292067	67,33707	5	
ST4000DM004	0,230769	0,210526	0,1	0,444444				0,231143	53,29102	8	
HDWD105UZSVA	0,076923	0,157895	0,2	0,222222				0,139426	32,1452	12	
HDWD110UZSVA	0,384615	0,157895	0,5	0,222222				0,359118	82,7961	3	
HDWD320UZSVA	0,307692	0,157895	0,2	0,222222				0,250195	57,68347	7	
HDWD240UZSVA	0,230769	0,157895	0,1	0,222222				0,189272	43,63742	20	
								0,433738	=max		

Рисунок 3.1 – Виконання задачі МАІ за допомогою MS Excel

3.2 Використання пакету Super Decisions

Super Decisions – програмний пакет, розроблений Creative Decisions Foundation для аналізу, синтезу та обґрунтування складних рішень на основі методології MAI. Пакет Super Decisions є вільно розповсюджуваним, його можна завантажити із сайту <https://www.superdecisions.com/>. Зазначимо, що Creative Decisions Foundation було засновано 1996 року Томасом Л. Сааті, професором Університету Піттсбурга, Пенсільванія, США. Ми будемо аналізувати останню версію 3.2 [10].

Виконаємо задачу вибору жорсткого диска у програмному пакеті Super Decisions. Кожен рівень ієрархії у цьому пакеті називається кластером (cluster). У нашому випадку це кластери Goal, Criteria, Alternatives. У кожному кластері створимо відповідні елементи, що називаються вузлами (nodes). Створення ієрархії виконується у вікні Network (рис. 3.2). Кожному вузлу надається необхідне ім'я. Далі за допомогою опції «make/show connections» створюються необхідні зв'язки між вузлами кластерів. В результаті, на екрані з'являться стрілки між кластерами. За допомогою опцій пакета необхідно перевірити правильність та наявність усіх потрібних зв'язків.

На вкладці Judgments (Судження) виконується налаштування попарного порівняння вузлів ієрархії. У пакеті є кілька інструментів для попарного порівняння критеріїв та альтернатив: за допомогою матриці (Matrix), вербальної шкали (Verbal), анкети (Questionnaire), графічно (Graphical) та шляхом безпосереднього введення значень переваг. На рис. 3.3 показаний приклад попарного порівняння критеріїв за допомогою матриці та відповідно до табл. 2.3. Стрілки в матриці вказують напрямки переваг. Наприклад, бачимо з рис. 3.3, що критерій Capacity вчетверо важливіший за критерій Brand і вдвічі важливіший за критерій Capacity Stat. У правій частині екрана виводиться вектор пріоритетів та значення неконсистентності матриці. Аналогічно виконується попарне порівняння альтернатив за відповідними критеріями.

На рис. 3.4. показано попарне порівняння альтернатив за критерієм Capacity. Наприклад, із рис. 3.4 ми бачимо, що за даним критерієм альтернативи HDWD110UZSVA і ST1000DM010 у п'ять разів краще альтернативи HDWD105UZSVA.

На рис. 3.5 показано попарне порівняння критеріїв методом анкетування. Ліва панель вікна показує вікна, що вузол Buying a HDD є опорним для парного порівняння в кластері Criteria, в середній панелі можна вводити порівняльні судження, права панель відображає обчислені значення компонентів вектора пріоритетів критеріїв.

Network		Judgments		Ratings			
1. Choose		2. Node comparisons with respect to Capacity				3. Results	
Node Cluster		Graphical Verbal Matrix Questionnaire Direct				Normal Hybrid	
Choose Node		Comparisons wrt "Capacity" node in "Alternatives" cluster				Inconsistency: 0.00000	
Capacity		HDWD110UZSVA is 5 times more important than HDWD105UZSVA					
Cluster: Criteria		Inconsistency					
Choose Cluster		HDWD105UZS~				HDWD105UZ~0.02564	
Alternatives		HDWD110UZS~				HDWD110UZ~0.12820	
		HDWD240UZS~				HDWD240UZ~0.10256	
		HDWD320UZS~				HDWD320UZ~0.07693	
		ST1000DM01~				ST1000DM0~0.12820	
		ST2000DM00~				ST2000DM0~0.10256	
						ST4000DM0~0.07693	
						ST5000M0020.02564	
						WD10EZEX0.12820	
						WD20EZBX0.10256	
						WD40EZA0.07693	
						WD5000AVDS0.02564	

Рисунок 3.4 – Матриця попарного порівняння альтернатив

Network		Judgments	Ratings
1. Choose		2. Node comparisons with respect to Buying a HDD	
Node Cluster		Graphical Verbal Matrix Questionnaire Direct	
Choose Node ▶▶		Comparisons wrt "Buying a HDD" node in "Criteria" cluster	
Buying a HDD Cluster: Goal		Brand Stat is equally as important as Brand	
Choose Cluster ◀▶			
Criteria			

Рисунок 3.5 – Порівняння критеріїв методом анкетування

Network | **Judgments** | **Ratings**

1. Choose | **2. Node comparisons with respect to Buying a HDD** | **3. Results**

Node: Cluster: **Buying a HDD** | Cluster Goal: **Buying a HDD**

Choose Node: **Brand** | Choose Cluster: **Criteria**

Graphical | Verbal | Matrix | Questionnaire | Direct

Brand: 0.12
Brand Stat: 0.16
Capacity: 0.48
Capacity Stat: 0.24

This is the direct data input area. Type in new direct data here, and/or Click the invert box invert priorities for this direct data.

NOTE: Any changes made in direct data take effect immediately and overwrite pre-existing data inputted in the other modes.

Normal | Hybrid

Inconsistency: 0.00000

Brand	0.12000
Brand Stat	0.16000
Capacity	0.48000
Capacity ~	0.24000

Рисунок 3.6 – Порівняння критеріїв прямим методом

Наступним кроком є процес загального синтезу; в результаті якого ми отримаємо спільні пріоритети альтернатив з урахуванням рівнів важливості, які ми надали критеріям. Синтез виконується з головного вікна моделі за допомогою команди меню Computations та опції Synthesize. Внаслідок цього з'явиться вікно результату синтезу (рис. 3.7). У стовпці "Normals" якого виводяться глобальні пріоритети альтернатив. Стовпець "Ideals" виходить шляхом поділу кожного значення в стовпці "Normals" на найвище значення цього стовпця (0,130564 у цьому прикладі). Отже, найвищий пріоритет має значення 1 чи 100%.

New synthesis for: Main Network: hdd.sdmod: formulaic: ratings

Here are the overall synthesized priorities for the alternatives. You synthesized from the network Main Network: hdd.sdmod: formulaic: ratings

Name	Graphic	Ideals	Normals	Raw
HDWD105UZSVA		0.313614	0.040947	0.020473
HDWD110UZSVA		0.874483	0.114176	0.057088
HDWD240UZSVA		0.596400	0.077868	0.038934
HDWD320UZSVA		0.440896	0.057565	0.028783
ST1000DM010		1.000000	0.130564	0.065282
ST2000DM008		0.721924	0.094257	0.047129
ST4000DM004		0.566428	0.073955	0.036977
ST5000M002		0.439135	0.057335	0.028668
WD10EZEX		0.994680	0.129869	0.064935
WD20EZBX		0.716607	0.093563	0.046781
WD40EZA		0.561109	0.073261	0.036630
WD5000AVDS		0.433814	0.056640	0.028320

Okay Copy Values

Рисунок 3.7 – Вікно з глобальними пріоритетами альтернатив

Однією з переваг пакету Super Decisions є автоматичне обчислення коефіцієнта неузгодженості, що дозволяє у будь-якому вікні (рис. 3.4 –3.6) контролювати значення цього коефіцієнта. Зазначимо, що повна консистентність – це ідеальна узгодженість матриці з погляду математики, але може суперечити ОНР щодо пріоритетів критеріїв. Тому ОНР має можливість змінювати інтенсивність пріоритетів, одночасно спостерігаючи за тим, щоб неконсистентність залишалася в межах допустимого ($CR \leq 1$).

Пакет Super Decisions має багато різноманітних опцій, призначених для всебічного аналізу прийнятого рішення. Зокрема, за допомогою Super Decisions можна, задаючи різні сценарії вибору альтернатив, виконати аналіз чутливості, що дозволяє з'ясувати, наскільки чутливими є результати стосовно важливості критеріїв.

3.3 Програма реалізації вибору оптимальної компоненти КС

Розглянуті вище підходи до реалізації вибору найкращих альтернатив комп'ютерних компонентів не вирішують задачі оптимізації КС, але є складовою цієї задачі, оскільки для визначення оптимальної конфігурації необхідно визначити значення цінності кожного компонента. Метод МАІ, як показано на прикладі вибору жорсткого диска, дозволяє визначити значення цінності об'єктів, що оцінюються. Тому цей метод було взято за основу програмного застосування, що розробляється.

Відповідно до Розділу 1, де сформульовано задача оптимізації конфігурації КС, програмне застосування складатиметься з реалізації методу МАІ для визначення цінності кожного компонента, вибору найбільш підходящих екземплярів кожного компонента з погляду відношення цінність/вартість та синтезу оптимальної конфігурації з цих компонентів. У ході науково-дослідницької (переддипломної) практики було розроблено

програмні модулі (мова Python) для вибору оптимальних з погляду відношення цінність/вартість основних компонентів КС. У Додатку А наведено код модуля для вибору оптимальних екземплярів жорсткого диска (лист А1). На рис. 3.8 показаний графік, отриманий внаслідок роботи модуля. На цьому графіку точками відзначені всі альтернативи запропонованих моделей жорсткого диска, точки парето-оптимальних альтернатив з'єднані лінією. Похила пряма відзначає оптимальну точку з погляду співвідношення цінність-вартість. На рис. 3.9 наведено екран із результатами обчислень, значення цінності та вартості кожної моделі, а також назва оптимальної моделі жорсткого диска.

Для обчислення векторів пріоритетів та перевірки консистентності матриць парних порівнянь розроблено функції `pref_vector()` та `consi()`, які включені в модуль `ahplib`. Парето-оптимальна множина обчислюється за допомогою функцій бібліотеки `paretoset` [11].

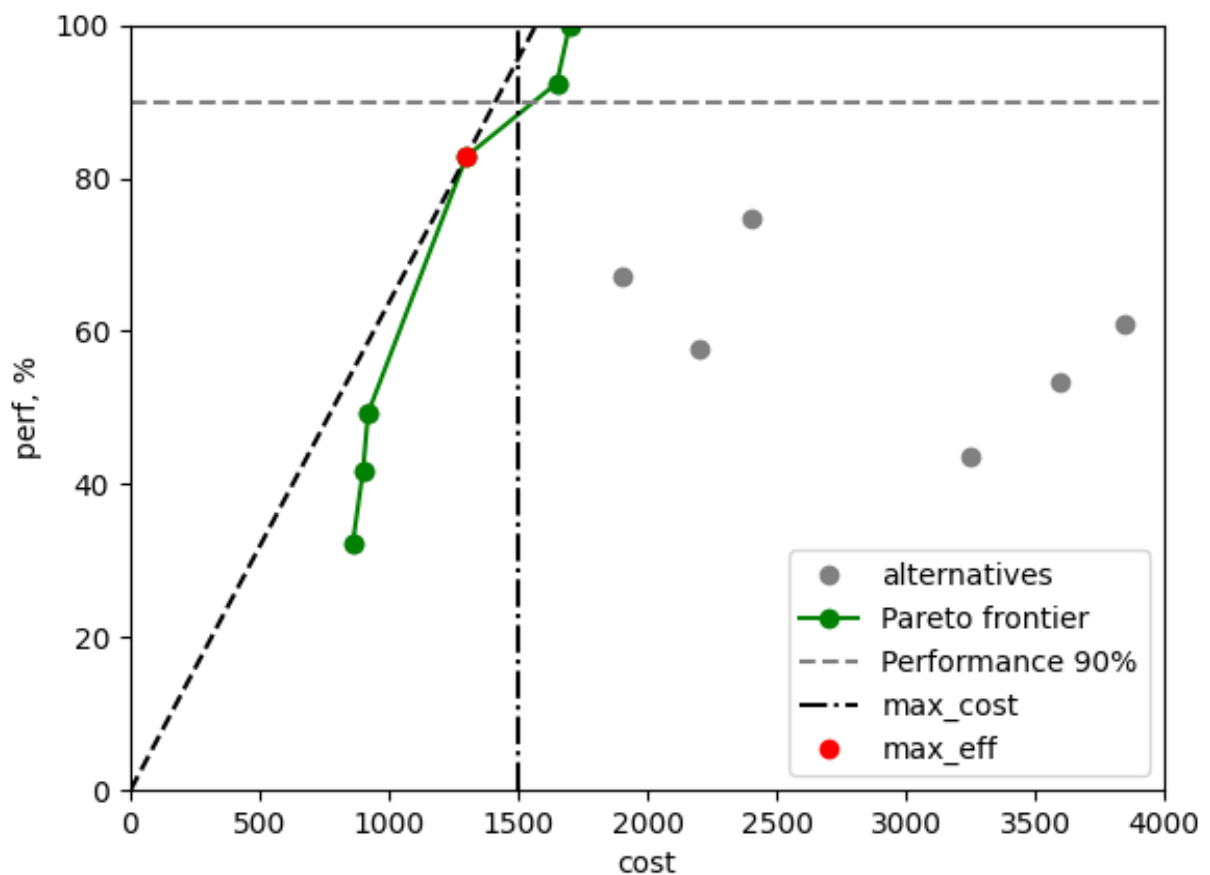
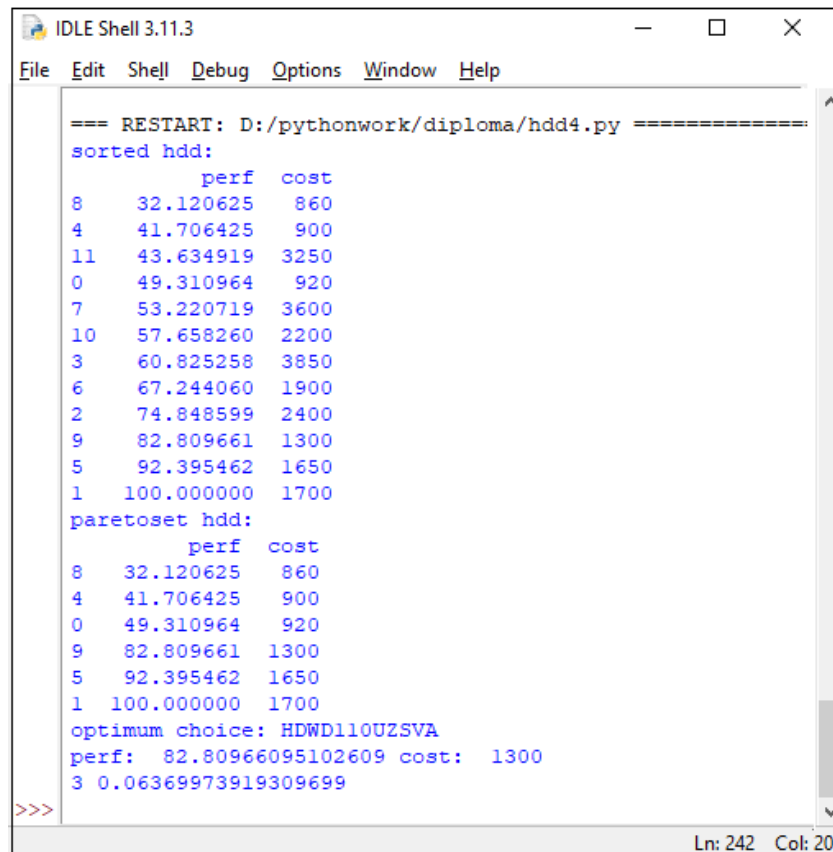


Рисунок 3.8 – Визначення оптимальних екземплярів жорсткого диска



```

IDLE Shell 3.11.3
File Edit Shell Debug Options Window Help

=== RESTART: D:/pythonwork/diploma/hdd4.py ===
sorted hdd:
      perf  cost
8  32.120625  860
4  41.706425  900
11 43.634919 3250
0  49.310964  920
7  53.220719 3600
10 57.658260 2200
3  60.825258 3850
6  67.244060 1900
2  74.848599 2400
9  82.809661 1300
5  92.395462 1650
1 100.000000 1700
paretoset hdd:
      perf  cost
8  32.120625  860
4  41.706425  900
0  49.310964  920
9  82.809661 1300
5  92.395462 1650
1 100.000000 1700
optimum choice: HDWD110UZSVA
perf:  82.80966095102609 cost:  1300
3 0.06369973919309699
>>>
Ln: 242 Col: 20

```

Рисунок 3.9 – Екран IDLE Shell 3.11.3 з результатом роботи програми

4 РЕАЛІЗАЦІЯ ПРОГРАМИ ВИБОРУ ОПТИМАЛЬНОЇ КОНФІГУРАЦІЇ

4.1 Реалізація модулів обчислення цінностей комп'ютерних компонентів

У даному підрозділі будемо припускати, що ОПР порівнює компоненти КС та складає матриці парних порівнянь за кількома критеріями, причому ОПР може покладатися на власний досвід, інтуїцію, думки експертів, а також статистиці продажів та відгуків покупців. У разі власних суджень ОПР використовує шкалу Т. Сааті, у разі використання статистики, наприклад продажів комп'ютерних компонентів або відкликання покупців, матриці парних порівнянь ОПР становить використовуючи рейтинги виробників або компонентів цих статистик. Приклад складання матриць парних порівнянь,

визначення вектора пріоритетів (цінності) та перевірки консистентності матриць парних порівнянь для одного з компонентів – жорсткого диска – наведено у підрозділах 2.1 та 2.2

4.1.1 Процесор (CPU)

Сьогодні переважну більшість CPU створюють три компанії. Корпорація Intel виробляє CPU з архітектурою x86-64 і здебільшого розробляє чіпи, а також займається їх виробництвом. Компанія Advanced Micro Devices, Inc. (AMD) також виробляє процесори з архітектурою x86-64 та розробляє мікросхеми. AMD є напівпровідниковою компанією без власних виробничих потужностей, тобто вона покладається на іншу компанію для виробництва фізичних процесорів. Компанія Arm Ltd. виготовляє CPU з архітектурою Advanced RISC Machine (ARM). Arm ліцензує свої процесори багатьом іншим компаніям, таким як Apple, Samsung та Qualcomm, які розробляють ці чіпи [1, 12].

Для кожного комп'ютера потрібна апаратна платформа, яка розроблена для конкретного CPU. Система заснована на процесорах Intel потребує материнської плати, розробленої конкретного набору Intel процесорів. Для процесорів AMD також потрібна материнська плата саме під цей процесор [13].

Виробники процесорів розрізняють лінійки продуктів, використовуючи різні назви продуктів, і ці назви змінювалися з роками. Довгий час Intel використовувала Pentium для своєї флагманської моделі, лише додаючи номери моделей, щоб показати наступні покоління – Pentium, Pentium II, Pentium III тощо. AMD використовувала бренд Athlon подібним чином. Сьогодні Intel здебільшого використовує торгову марку Core; AMD використовує торгову марку Ryzen. Більшість обговорень процесорів для ПК зосереджено на чотирьох напрямках конфігурацій КС: настільні ПК, бюджетні ПК, портативні/мобільні ПК та серверні комп'ютери [14, 15]. Як ми вже говорили, розглядатимемо конфігурації тільки настільних КС.

Оскільки процесор працює швидше, ніж оперативна пам'ять може забезпечити його кодом, CPU оснащено вбудованою високошвидкісною статичною оперативною пам'яттю (SRAM). Ця SRAM попередньо завантажує

якомога більше інструкцій і зберігає копії вже запущених інструкцій і даних на випадок, якщо CPU знову буде працювати над ними. SRAM, що використовується таким чином, називається кешем. Кеш, вбудований у ЦП, назвали кеш-пам'яттю першого рівня (L1). Багато сучасних CPU включають три кеші: кеш L1, L2 і L3 (Smart Cache). Інтеграція графічного процесора (GPU) у центральний процесор підвищує загальну продуктивність комп'ютера і водночас знижує енергоспоживання, розмір та вартість КС.

У будь-якому випадку, при виборі процесора, а потім материнської плати або навпаки, спочатку материнської плати, а потім процесора, необхідно враховувати два ключові моменти. По-перше, чи підтримує материнська плата процесори Intel чи AMD. По-друге, який роз'єм (сокет) процесора встановлений на материнській платі. У таблиці 4.1 представлені процесори Intel та AMD та їх основні характеристики. Процесори Intel мають сокет LGA 1200; чіпсет B560; тип пам'яті DDR4, 2 канали, а процесори AMD – сокет AM4; чіпсети B550, X570; тип пам'яті DDR4, 2 канали.

Таблиця 4.1 – Вихідні дані для вибору процесора

Виробник, найменування моделі	Кількість ядер (потоків)	Тактова частота, ГГц	Кеш: 1,2,3 рівень	Інтегрована графіка	Вартість, грн.
Intel i3 10100	4(8)	3,6-4,3	256 Kb, 1024 Kb, 6 Mb	UHD Graphics 630	4600
Intel i5 10400	6(12)	2,6	256 Kb, 1024 Kb, 12 Mb	UHD Graphics 730	6200
Intel i7 11700	8(16)	2,5-4,9	256 Kb, 1024 Kb, 16 Mb	UHD Graphics 730	8800
AMD Ryzen 3 4300G	4(8)	3,8-4,0	256 Kb, 2 Mb, 4 Mb	AMD Radeon Graphics	3800
AMD Ryzen 5 4600G	6(12)	3,7-4,2	384 Kb, 3 Mb, 8 Mb	AMD Radeon Graphics	4600
AMD Ryzen 7 5700G	8(16)	3,8-4,6	512 Kb, 4 Mb, 16 Mb	Radeon Vega 8	6800

Таблиця 4.2 – Матриця парних порівнянь процесорів

	i3	i5	i7	Ryzen 3	Ryzen 5	Ryzen 7	Priority
i3	1	0,714	0,286	0,714	0,429	0,143	0,06
i5	1,4	1	0,4	1	0,6	0,2	0,08
i7	3,5	2,5	1	2,5	1,5	0,5	0,21
Ryzen 3	1,4	1	0,4	1	0,6	0,2	0,08
Ryzen 5	2,333	1,667	0,667	1,667	1	0,333	0,14
Ryzen 7	7	5	2	5	3	1	0,42

4.1.2 Материнська плата

Три взаємопов'язані характеристики визначають сучасні материнські плати: форм-фактор, набір мікросхем і компоненти. Форм-фактор визначає фізичний розмір материнської плати, а також загальне розташування компонентів та портів. Важливим фактором материнських плат, який іноді не враховують недосвідчені користувачі, є організація електроживлення.

Набір мікросхем (чипсет) визначає тип процесора та оперативної пам'яті, які потрібні материнській платі, а також певною мірою визначає вбудовані пристрої, які підтримують материнську плату, включаючи слоти розширення. Вбудовані компоненти визначають основну функціональність системи.

Майже всі чіпсети, що використовуються в настільних комп'ютерах та ноутбуках, виробляються Intel або AMD.

У табл 4.3 та 4.4 наведено екземпляри категорій материнських плат, які узгоджені за сокетом та чіпсетом з наведеними в табл. 4.2 екземплярами CPU Intel та AMD, відповідно.

Таблиця 4.3 – Вихідні дані для вибору материнської плати для процесорів Intel

Виробник	Найменування моделі	Вартість, грн.
ASUS	Pro B560M-C/CSM	5200
	TUF GAMING B560M-PLUS	7200
	ROG STRIX B560-E GAMING WIFI	8800
Gigabyte	B560M H	3000
	B560M DS3H V2	4700
MSI	B560M PRO-E	3000
	B560M PRO WIFI	4500
	MAG B560 TORPEDO	8000
ASRock	B560M-HDV/M.2	2700

Таблиця 4.4 – Вихідні дані для вибору материнської плати для процесорів AMD

Виробник	Найменування моделі	Вартість, грн.
ASUS	PRIME B550M-K	3800
	PRIME B550M-A/CSM	3900
	PRIME B550-PLUS	4300
Gigabyte	B550M K	3500
	B550M DS3H	3900
	B550 AORUS ELITE V2	5000
MSI	PRO B550M-P GEN3	3800
	B550-A PRO	5000
	MAG B550 TOMAHAWK	6500
ASRock	B550M PG Riptide	4500
	B550M Pro4	5000
	B550M Steel Legend	5300

Нехай користувач вирішив оцінювати материнські плати за п'ятьма критеріями: Бренд, Рейтинг продажів (рис 4.1), Рейтинг позитивних відгуків (рис. 4.2), власним вподобанням (Preference) та можливості модифікації КС. Тоді ієрархію задачі вибору материнської плати можна представити у вигляді, показаному на рис. 4.3.

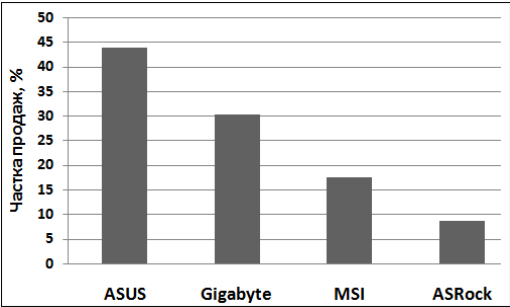


Рисунок 4.1 – Рейтинг продажів материнських плат

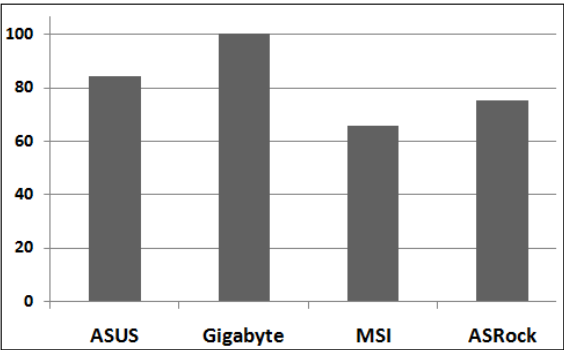


Рисунок 4.2 – Материнські плати: рейтинг позитивних відгуків

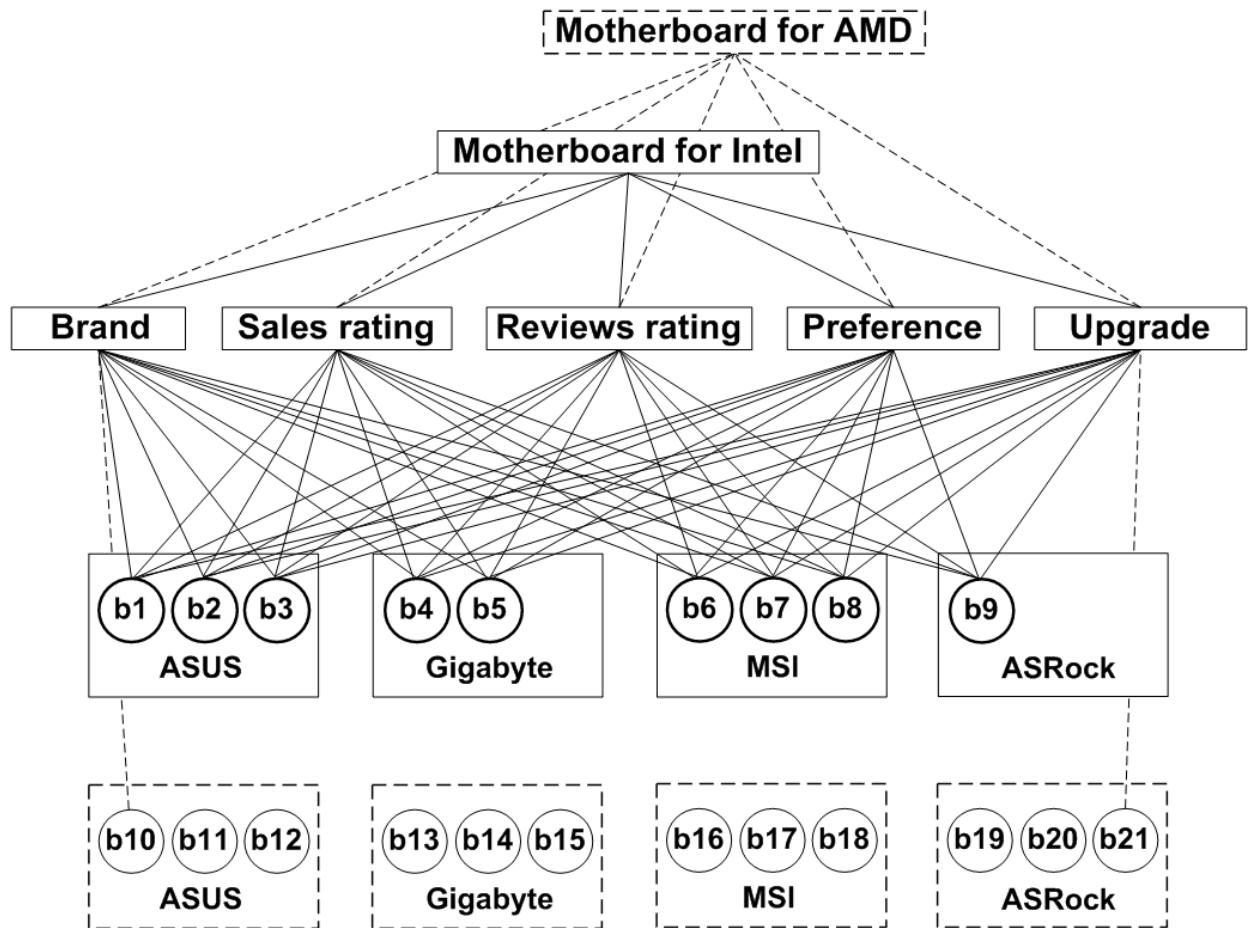


Рисунок 4.3 – Ієрархія прийняття рішення щодо вибору материнської плати

Таблиця 4.5 – Матриця парних порівнянь материнських плат за критерієм рейтингу продажів

Sales rating	ASUS	Giga	MSI	ASRock	Priority
ASUS	1	1,43	2,457	5,375	0,437
Giga	0,698	1	1,714	3,75	0,305
MSI	0,407	0,583	1	2,188	0,178
ASRock	0,186	0,267	0,457	1	0,081

Таблиця 4.6 – Матриця парних порівнянь материнських плат за критерієм позитивних відгуків

Positive rating	ASUS	Giga	MSI	ASRock	Priority
ASUS	1	0,82	1,323	1,093	0,257
Giga	1,22	1	1,613	1,333	0,313
MSI	0,756	0,62	1	0,827	0,194
ASRock	0,915	0,75	1,21	1	0,235

Таблиця 4.7 – Матриця парних порівнянь за критерієм вподобання користувача

Preference	ASUS	Giga	MSI	ASRock	Priority
ASUS	1	2	0,333	3	0,207
Giga	0,5	1	1,667	1,5	0,103
MSI	3	6	1	9	0,621
ASRock	0,333	0,667	0,111	1	0,069

Таблиця 4.8 – Матриця парних порівнянь за критерієм апгрейд

Upgrade	Mboard-1	Mboard-2	Mboard-3	Priority
Mboard-1	1	2	0,333	0,1
Mboard-2	0,5	1	1,667	0,4
Mboard-3	3	6	1	0,5

Таблиця 4.9 – Матриця парних порівнянь критеріїв оцінки

Criteria	Sales rating	Positive rating	Preference	Upgrade	Priority
Sales rating	1	0,2	0,333	0,143	0,0625
Positive rating	5	1	1,667	0,714	0,3125
Preference	3	0,6	1	0,429	0,1875
Upgrade	7	1,4	2,333	1	0,4375

4.1.3 Монитор

Невід’ємним компонентом домашнього та офісного комп’ютера є монітор (дисплей). Нехай користувачеві запропоновано вибрати монітор зі списку, наведеного в таблиці 4.10. До основних характеристик сучасних моніторів, якими користувач може віддати перевагу певної моделі монітора, є розмір діагоналі, співвідношення сторін екрану, тип матриці екрана, частота оновлення, яскравість і контрастність. Багато користувачів звертають увагу на зручність та дизайн, тому що монітор – це видима частина комп’ютера, і вона повинна гармонійно вписуватися в інтер’єр робочого місця вдома або в офісі.

У табл. 4.10 наведено деякі основні характеристики моніторів. Крім того, ми припускаємо, що багато характеристик наведеного списку моніторів однакові або схожі. До цих характеристик відносяться: співвідношення сторін

16:9, яскравість дисплея 250 кд/м², контрастність 1000:1, підсвітка матриці LED, кут огляду 178°.

Таблиця 4.10 – Вихідні дані для вибору монітора

Виробник	Найменування моделі	Розмір діагоналі, дюйми	Тип матриці	Частота оновлення, Гц	Вартість, грн.
MSI	MSI PRO MP223	21,45	VA	100	3600
	MSI PRO MP243	23,8	IPS	75	4200
	MSI PRO MP271	27	IPS	75	5400
Samsung	S22C310	21,5	IPS	75	4200
	S24R358FZ	23,8	IPS	75	4600
	S27C360	27	VA	75	5600
LG	LG 22MP410-B	21,5	IPS	75	4200
	LG 24MK430H-B	23,8	IPS	75	4700
	LG 27MP400-B	27	IPS	75	5800
ASUS	VP229HE	21,5	IPS	75	4000
	VA24EHF	23,8	IPS	100	4800
	VZ27ENE	27	IPS	75	6000

Для нашого завдання оптимізації ми оцінюватимемо монітори лише за трьома критеріями, як показано на рис. 4.4, хоча, безумовно, досвідчений користувач може при виборі врахувати більшу кількість характеристик і, відповідно, критеріїв оцінки.

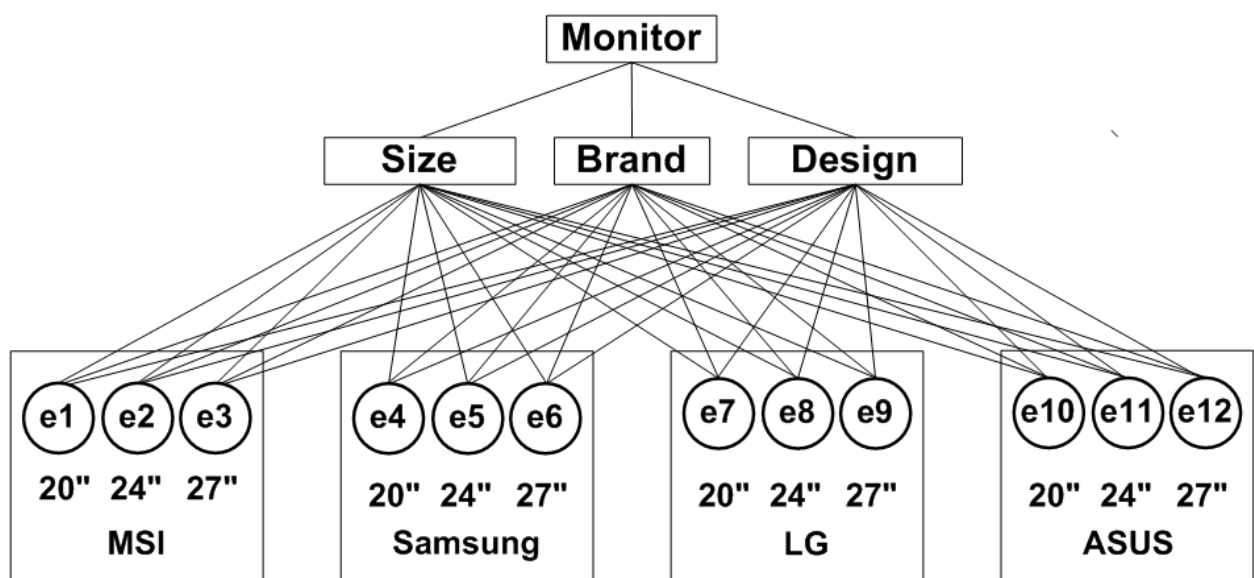


Рисунок 4.4 – Ієрархія прийняття рішення щодо вибору монітору

В табл. 4.11 – 4.13 представлені матриці парних порівнянь моніторів за вибраними критеріями У табл. 4.14 – матриця парних порівнянь критеріїв вибору монітора.

Таблиця 4.11 – Матриця парних порівнянь за критерієм Бренд

Brand	MSI	Samsung	LG	ASUS	Priority
MSI	1	1,5	3	5	0,455
Samsung	0,667	1	2	3,333	0,303
LG	0,333	0,5	1	1,667	0,152
ASUS	0,2	0,3	0,6	1	0,091

Таблиця 4.12 – Матриця парних порівнянь за критерієм Розмір діагоналі

Size	20"	24"	27"	Priority
20"	1	0,5	0,91	0,244
24"	2	1	1,82	0,488
27"	1,1	0,55	1	0,268

Таблиця 4.13 – Матриці парних порівнянь за критерієм Дизайн

	MSI					Samsun	Samsung				
MSI		e1	e2	e3	Priority			e4	e5	e6	Priority
	e1	1	0,5	0,33	0,167		e4	1	0,33	0,5	0,167
	e2	2	1	0,67	0,333		e5	3	1	1,5	0,5
	e3	3	1,5	1	0,5		e6	2	0,67	1	0,333
	LG					ASUS	ASUS				
LG		e7	e8	e9	Priority			e10	e11	e12	Priority
	e7	1	0,2	0,143	0,077		e10	1	0,33	0,25	0,125
	e8	5	1	0,714	0,385		e11	3	1	0,75	0,375
	e9	7	1,4	1	0,538		e12	4	1,33	1	0,5

Таблиця 4.14 – Матриця парних порівнянь критеріїв оцінки

Criteria	Size	Brand	Design	Priority
Size	1	3	5	0,652
Brand	0,33	1	1,667	0,218
Design	0,2	0,6	1	0,13

4.1.4 Корпус

Корпус також може бути видимою частиною КС, проте важливими характеристиками корпусу є його розмір, функціональність, якість системи охолодження, розташування блоку живлення. У табл. 4.15 наведені найбільш

популярні (рис. 4.5) моделі корпусів типу midiTower, в які можна розмістити материнські плати з форм-фактором ATX або microATX. Всі наведені корпуси не містять блоку живлення, але в них передбачено розміщення живлення в нижній частині, що на думку багатьох користувачів забезпечує кращий баланс температури всередині корпусу.

Таблиця 4.15 – Вихідні дані для вибору корпусу

Виробник	Найменування моделі	Вартість, грн.
Vinga	Legend	1350
	Pillar	2150
2E	Virtus G3301	1900
	Virtus Neo G3301N	2200
Qube	Neptune	2100
	Shadow	2400
1stPlayer	F4-3B1	1200
	F4-3R2	1650

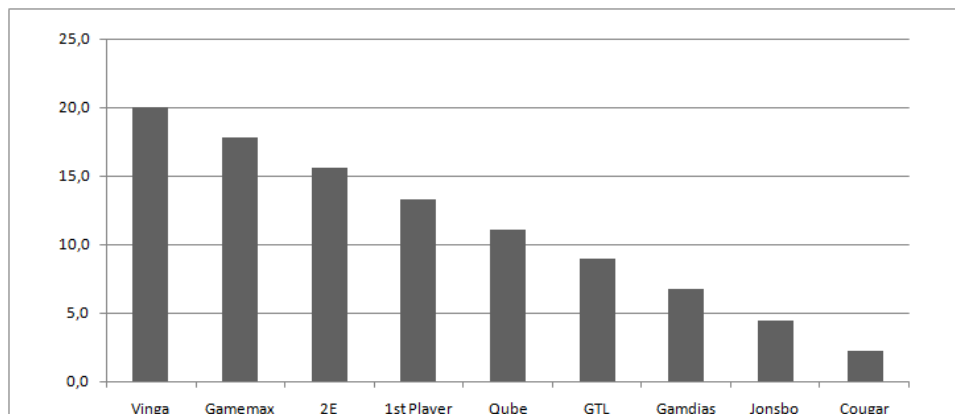


Рисунок 4.5 – Рейтинг продажів корпусів

На рис. 4.6 показано ієрархію та критерії оцінки корпусів. У табл. 4.16 – 4.18 представлені матриці парних порівнянь корпусів за критеріями рейтингу продажів, дизайну, вподобань користувача.

У табл. 4.19 – матриці парних порівнянь корпусів за критерієм функціональність, який може включати можливість встановлення додаткових жорстких дисків, набір необхідних роз'ємів, засоби поглинання шуму та

пилових фільтрів та зручність кабель-менеджменту. У табл. 4.20 представлено матрицю парних порівнянь критеріїв оцінки корпусів.

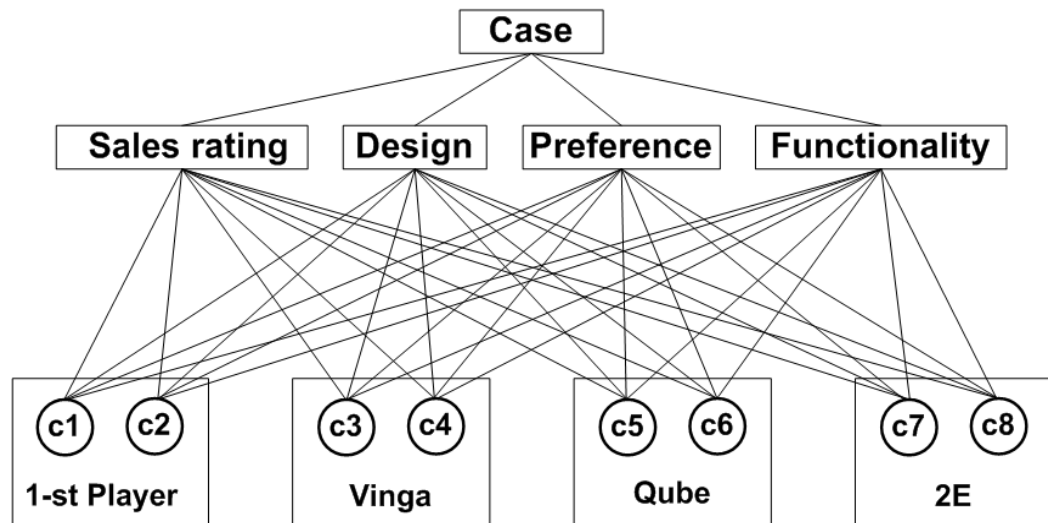


Рисунок 4.6 – Ієрархія прийняття рішення щодо вибору корпусу

Таблиця 4.16 – Матриця парних порівнянь корпусів за критерієм рейтингу продажів

Sales rating	Vinga	2E	Qube	1st Player	Priority
Vinga	1	1,25	1,82	1,54	0,33
2E	0,8	1	1,45	1,23	0,27
Qube	0,55	0,69	1	0,85	0,18
1st Player	0,65	0,81	1,18	1	0,22

Таблиця 4.17 – Матриця парних порівнянь корпусів за критерієм дизайн

Design	Vinga	2E	Qube	1st Player	Priority
Vinga	1	5	3	2	0,49
2E	0,2	1	0,6	0,4	0,1
Qube	0,33	1,67	1	0,67	0,16
1st Player	0,5	2,5	1,5	1	0,25

Таблиця 4.18 – Матриця парних порівнянь корпусів за критерієм вподобання користувача

Preference	Vinga	2E	Qube	1st Player	Priority
Vinga	1	0,33	0,5	0,2	0,09
2E	3	1	1,5	0,6	0,27
Qube	2	0,67	1	0,4	0,18
1st Player	5	1,67	2,5	1	0,45

Таблиця 4.19 – Матриці парних порівнянь за критерієм Функціональність

	1st Player					Vinga			
1st Player		c1	c2	Priority	Vinga		c3	c4	Priority
	c1	1	0,4	0,29		c3	1	0,33	0,25
	c2	2,5	1	0,71		c4	3	1	0,75
	Qube					2E			
Qube		c5	c6	Priority	2E		c7	c8	Priority
	c5	1	0,67	0,4		c7	1	0,5	0,33
	c6	1,5	1	0,6		c8	2	1	0,67

Таблиця 4.20 – Матриця парних порівнянь критеріїв оцінки корпусів

Criteria	Sales rating	Design	Preference	Functionality	Priority
Sales rating	1	0,33	0,2	0,14	0,062
Design	3	1	0,6	0,43	0,188
Preference	5	1,67	1	0,71	0,312
Functionality	7	2,33	1,4	1	0,438

Програмні реалізації визначення цінності компонентів КС, наведені у цьому розділі аналогічні реалізації модуля оцінки жорсткого диска (підрозділ 2.1). Кожен програмний модуль можна використовувати окремо, що дозволяє користувачеві оцінити та вибрати оптимальну альтернативу будь-якої компоненти КС.

4.2 Реалізація головного модуля вибору оптимальної конфігурації КС

У головному програмному модулі обчислюються глобальна оцінка цінності та повної вартості кожної з альтернатив КС. Для цього використовуються оцінки цінностей, отримані в програмних модулях, для кожної компоненти КС, та відомі вартості кожної з компонентів. Оцінка глобальної цінності КС обчислюється виходячи з матриці парних порівнянь, складеної користувачем з урахуванням важливості для нього кожної з компоненти. Приклад такої матриці наведено у табл. 4.21.

Таблиця 4.21 – Приклад матриці парних порівнянь компонентів КС

Component	CPU	Motherboard	HDD	Monitor	Case
CPU	1	2	5	7	9
Motherboard	0,5	1	2,5	3,5	4,5
HDD	0,2	0,4	1	1,4	1,8
Monitor	0,143	0,286	0,714	1	1,286
Case	0,111	0,222	0,556	0,778	1

На підставі таблиць з вихідними даними по всіх компонентах складемо зведену таблицю для визначення кількості альтернатив компонентів (табл. 4.22).

Таблиця 4.22 – Кількість альтернатив компонентів КС

Компоненти КС	Кількість альтернатив компонентів	
CPU	Intel	AMD
	3	3
Motherboard	9	12
HDD	12	
Monitor	12	
Case	8	

З табл. 4.22 можна підрахувати кількість альтернативних конфігурацій КС: $N = 3 \cdot (9+12) \cdot 12 \cdot 12 \cdot 8 = 72576$.

На рис. 4.7 представлено блок-схему реалізації головного модуля програми. Лістинг програми наведено у додатку А (Ліст. А2). Як видно з блок-схеми та Ліст. А2 цінності альтернатив компонентів обчислюються у модулях `casemodule`, `monitormodule`, `hddmodule`, `processormodule`, `motherboardIntelmodule` і `motherboardAMDmodule`.

У разі неконсистентності матриці парних порівнянь на екран виводиться відповідне повідомлення. Це стосується і матриці парних порівнянь компонентів з погляду їх важливості для користувача. Обчислення векторів цінностей комп'ютерних компонентів та КС і перевірка консистентності всіх матриць парних порівнянь виконується у розробленому модулі `ahplib`.

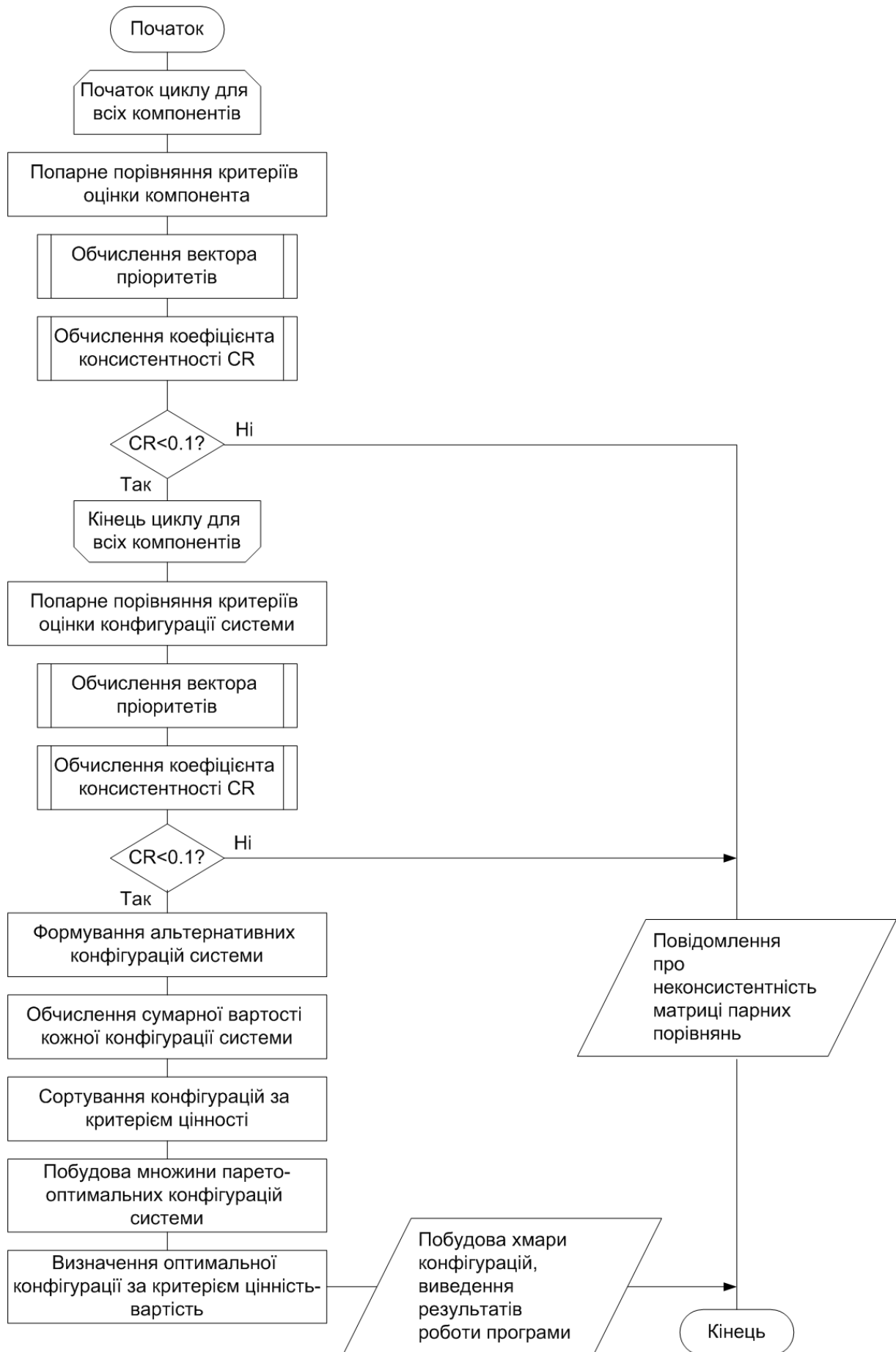


Рисунок 4.7 – Алгоритм роботи програми

В результаті роботи програми отримані оцінки всіх 72576 альтернативних конфігурацій, які утворюють хмару конфігурацій, показаних на рис. 4.8, побудована парето-оптимальна множина, що складається з 55 альтернативних конфігурацій, які наведені в табл. 4.23, визначено оптимальні конфігурації з погляду співвідношення цінність, вартість (model 61927), що складається з компонентів: процесор AMD Ryzen 7, материнська плата ASUS PRIME B550-PLUS, жорсткий диск Toshiba HDWD110UZSVA, монітор MSI PRO MP223, корпус 1st Player F4-3R2. Загальна вартість такої конфігурації дорівнює 17 650 грн.

З табл. 4.23 користувач також може вибрати оптимальну конфігурацію КС, яка задовольняє його вимоги на задану кількість відсотків (наприклад, 90%), а також оптимальну альтернативу, яка є прийнятною для нього за вартістю (наприклад, 18650 грн.).

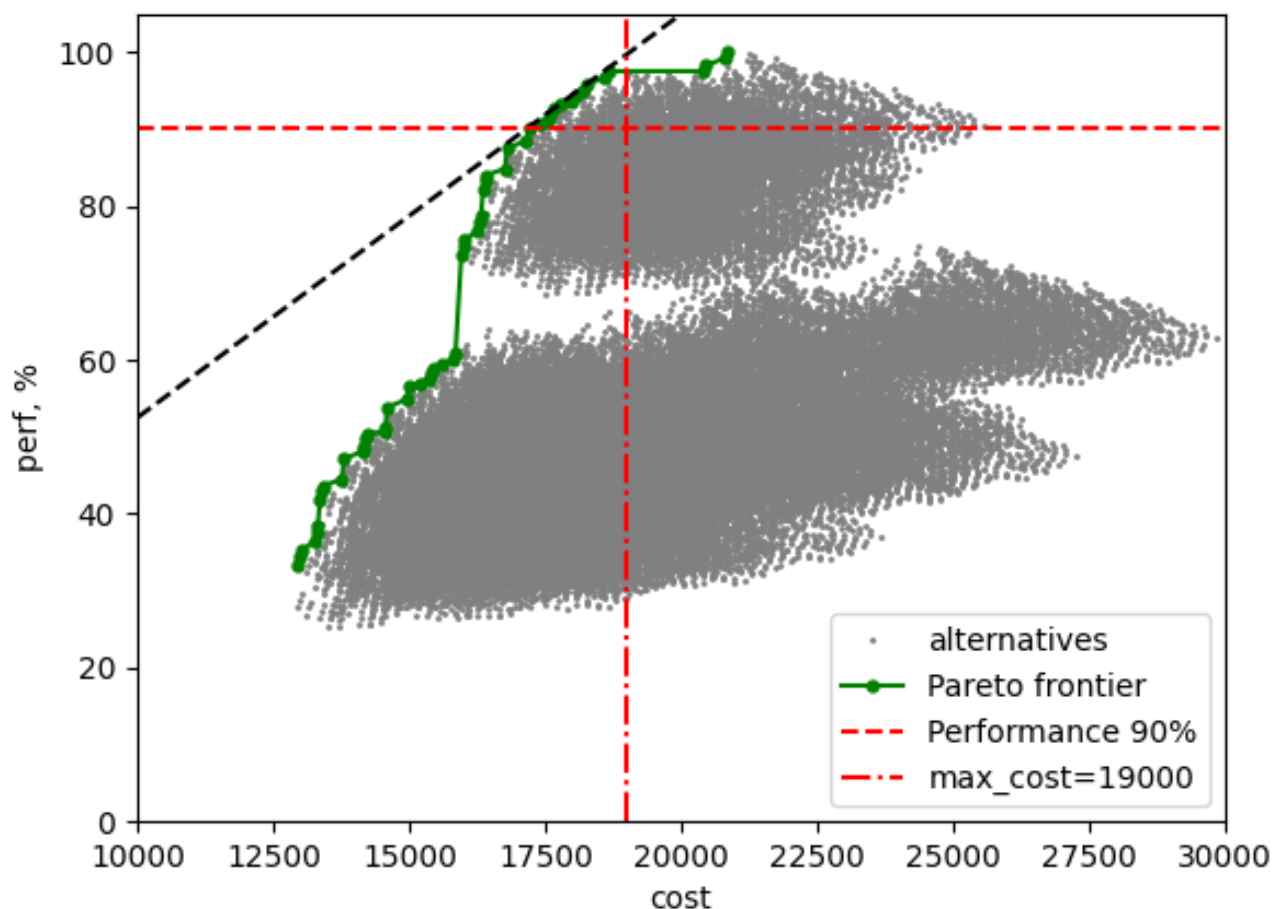


Рисунок 4.8 – Множина альтернатив комп'ютерних конфігурацій

Таблиця 4.23 – Список парето-оптимальних конфігурацій

Ідентифікатор моделі конфігурації	Назва материнської плати	Назва жорсткого диска	Назва монітора	Назва корпусу	Рейтинг та вартість моделі конфігурації
model-14598 (Intel) i5	B560M H	HDWD105UZSVA	MSI PRO MP223	F4-3B1	33.4 %, 12960
model-14214 (Intel) i5	B560M H	ST5000M002	MSI PRO MP223	F4-3B1	34.4 %, 13000
model-13830 (Intel) i5	B560M H	WD5000AVDS	MSI PRO MP223	F4-3B1	35.2 %, 13020
model-16902 (Intel) i5	B560M PRO-E	HDWD105UZSVA	MSI PRO MP223	F4-3B1	36.5 %, 13260
model-16518 (Intel) i5	B560M PRO-E	ST5000M002	MSI PRO MP223	F4-3B1	37.5 %, 13300
model-16134 (Intel) i5	B560M PRO-E	WD5000AVDS	MSI PRO MP223	F4-3B1	38.3 %, 13320
model-12294 (Intel) i5	TUF GAMING B560M-PLUS	HDWD105UZSVA	MSI PRO MP223	F4-3B1	41.8 %, 13360
model-11910 (Intel) i5	TUF GAMING B560M-PLUS	ST5000M002	MSI PRO MP223	F4-3B1	42.8 %, 13400
model-11526 (Intel) i5	TUF GAMING B560M-PLUS	WD5000AVDS	MSI PRO MP223	F4-3B1	43.6 %, 13420
model-13446 (Intel) i5	ROG STRIX B560-E GAMING	HDWD105UZSVA	MSI PRO MP223	F4-3B1	44.5 %, 13760
model-12390 (Intel) i5	TUF GAMING B560M-PLUS	HDWD110UZSVA	MSI PRO MP223	F4-3B1	47.1 %, 13800
model-12006 (Intel) i5	TUF GAMING B560M-PLUS	ST1000DM010	MSI PRO MP223	F4-3B1	48.1 %, 14150
model-46854 (AMD) R5	PRIME B550M-A/CSM	HDWD105UZSVA	MSI PRO MP223	F4-3B1	48.6 %, 14160
model-13542 (Intel) i5	ROG STRIX B560-E GAMING	HDWD110UZSVA	MSI PRO MP223	F4-3B1	49.7 %, 14200
model-46086 (AMD) R5	PRIME B550M-A/CSM	WD5000AVDS	MSI PRO MP223	F4-3B1	50.4 %, 14220
model-13158 (Intel) i5	ROG STRIX B560-E GAMING	ST1000DM010	MSI PRO MP223	F4-3B1	50.7 %, 14550
model-48006 (AMD) R5	PRIME B550-PLUS	HDWD105UZSVA	MSI PRO MP223	F4-3B1	51.2 %, 14560
model-46950 (AMD) R5	PRIME B550M-A/CSM	HDWD110UZSVA	MSI PRO MP223	F4-3B1	53.8 %, 14600
model-46566 (AMD) R5	PRIME B550M-A/CSM	ST1000DM010	MSI PRO MP223	F4-3B1	54.8 %, 14950

Ідентифікатор моделі конфігурації	Назва материнської плати	Назва жорсткого диска	Назва монітора	Назва корпусу	Рейтинг та вартість моделі конфігурації
model-48102 (AMD) R5	PRIME B550-PLUS	HDWD110UZSVA	MSI PRO MP223	F4-3B1	56.5 %, 15000
model-46958 (AMD) R5	PRIME B550M-A/CSM	HDWD110UZSVA	MSI PRO MP243	F4-3B1	56.9 %, 15200
model-47718 (AMD) R5	PRIME B550-PLUS	ST1000DM010	MSI PRO MP223	F4-3B1	57.5 %, 15350
model-47334 (AMD) R5	PRIME B550-PLUS	WD10EZEX	MSI PRO MP223	F4-3B1	58.3 %, 15400
model-48103 (AMD) R5	PRIME B550-PLUS	HDWD110UZSVA	MSI PRO MP223	F4-3R2	59.0 %, 15450
model-48110 (AMD) R5	PRIME B550-PLUS	HDWD110UZSVA	MSI PRO MP243	F4-3B1	59.6 %, 15600
model-47719 (AMD) R5	PRIME B550-PLUS	ST1000DM010	MSI PRO MP223	F4-3R2	60.0 %, 15800
model-47335 (AMD) R5	PRIME B550-PLUS	WD10EZEX	MSI PRO MP223	F4-3R2	60.8 %, 15850
model-62982 (AMD) R7	B550M K	HDWD105UZSVA	MSI PRO MP223	F4-3B1	73.7 %, 15960
model-62598 (AMD) R7	B550M K	ST5000M002	MSI PRO MP223	F4-3B1	74.7 %, 16000
model-62214 (AMD) R7	B550M K	WD5000AVDS	MSI PRO MP223	F4-3B1	75.5 %, 16020
model-66438 (AMD) R7	PRO B550M-P GEN3	HDWD105UZSVA	MSI PRO MP223	F4-3B1	76.8 %, 16260
model-66054 (AMD) R7	PRO B550M-P GEN3	ST5000M002	MSI PRO MP223	F4-3B1	77.8 %, 16300
model-65670 (AMD) R7	PRO B550M-P GEN3	WD5000AVDS	MSI PRO MP223	F4-3B1	78.6 %, 16320
model-60678 (AMD) R7	PRIME B550M-A/CSM	HDWD105UZSVA	MSI PRO MP223	F4-3B1	82.2 %, 16360
model-60294 (AMD) R7	PRIME B550M-A/CSM	ST5000M002	MSI PRO MP223	F4-3B1	83.2 %, 16400
model-59910 (AMD) R7	PRIME B550M-A/CSM	WD5000AVDS	MSI PRO MP223	F4-3B1	84.0 %, 16420
model-61830 (AMD) R7	PRIME B550-PLUS	HDWD105UZSVA	MSI PRO MP223	F4-3B1	84.8 %, 16760
model-60774 (AMD) R7	PRIME B550M-A/CSM	HDWD110UZSVA	MSI PRO MP223	F4-3B1	87.5 %, 16800
model-60390 (AMD) R7	PRIME B550M-A/CSM	ST1000DM010	MSI PRO MP223	F4-3B1	88.5 %, 17150
model-61926 (AMD) R7	PRIME B550-PLUS	HDWD110UZSVA	MSI PRO MP223	F4-3B1	90.1 %, 17200
model-60782 (AMD) R7	PRIME B550M-A/CSM	HDWD110UZSVA	MSI PRO MP243	F4-3B1	90.6 %, 17400

Ідентифікатор моделі конфігурації	Назва материнської плати	Назва жорсткого диска	Назва монітора	Назва корпусу	Рейтинг та вартість моделі конфігурації
model-61542 (AMD) R7	PRIME B550-PLUS	ST1000DM010	MSI PRO MP223	F4-3B1	91.1 %, 17550
model-61158 (AMD) R7	PRIME B550-PLUS	WD10EZEX	MSI PRO MP223	F4-3B1	91.9 %, 17600
model-61927 (AMD) R7	PRIME B550-PLUS	HDWD110UZSVA	MSI PRO MP223	F4-3R2	92.6 %, 17650
model-61934 (AMD) R7	PRIME B550-PLUS	HDWD110UZSVA	MSI PRO MP243	F4-3B1	93.2 %, 17800
model-61543 (AMD) R7	PRIME B550-PLUS	ST1000DM010	MSI PRO MP223	F4-3R2	93.6 %, 18000
model-61159 (AMD) R7	PRIME B550-PLUS	WD10EZEX	MSI PRO MP223	F4-3R2	94.4 %, 18050
model-61166 (AMD) R7	PRIME B550-PLUS	WD10EZEX	MSI PRO MP243	F4-3B1	95.0 %, 18200
model-61935 (AMD) R7	PRIME B550-PLUS	HDWD110UZSVA	MSI PRO MP243	F4-3R2	95.7 %, 18250
model-61551 (AMD) R7	PRIME B550-PLUS	ST1000DM010	MSI PRO MP243	F4-3R2	96.7 %, 18600
model-61167 (AMD) R7	PRIME B550-PLUS	WD10EZEX	MSI PRO MP243	F4-3R2	97.5 %, 18650
model-68078 (AMD) R7	MAG B550 TOMAHAWK	WD10EZEX	MSI PRO MP243	F4-3B1	97.5 %, 20400
model-68847 (AMD) R7	MAG B550 TOMAHAWK	HDWD110UZSVA	MSI PRO MP243	F4-3R2	98.2 %, 20450
model-68463 (AMD) R7	MAG B550 TOMAHAWK	ST1000DM010	MSI PRO MP243	F4-3R2	99.2 %, 20800
model-68079 (AMD) R7	MAG B550 TOMAHAWK	WD10EZEX	MSI PRO MP243	F4-3R2	100.0 %, 20850

ВИСНОВКИ

У ході виконання дипломної магістерської роботи була сформульована задача оптимізації конфігурації КС. Ця задача стосується багатокритеріального прийняття рішень. Особливістю задачі конфігурування КС є велика кількість можливих альтернатив, які можуть вимірюватися десятками і навіть сотнями тисяч. Розв'язання подібної задачі звичайному користувачеві дуже складне і це визначило актуальність роботи.

Оскільки КС містить в собі багато компонентів, серед них було обрано лише основні – процесор, материнська плата, жорсткий диск, монітор і корпус та досліджено ринок цих компонентів для конфігурування бюджетних КС домашнього та офісного користування. Але й при цьому задача не спрощується, тому що на сучасному ринку пропонується багато альтернатив комп'ютерних компонентів від різних виробників. Враховуючи те, що кожен компонент може оцінюватися за декількома критеріями, задача оцінки залишається складною за рахунок багатомірності, і для ухвалення рішення про вибір конфігурації є дуже багато альтернативних рішень. З цієї точки зору були проаналізовані найбільш популярні методи багатокритеріального прийняття рішень, зокрема SAW, TOPSIS, VIKOR, АНР (MAI), ELECTRE, POMETHEE та ін. В результаті цього аналізу вибір було зроблено на користь MAI, що дозволяє розбити складну задачу оцінки альтернатив на дрібніші за рахунок побудови ієрархії – ціль-критерії-альтернативи.

Метод MAI було апробовано на прикладі вибору одного з компонентів КС – жорсткого диска. Розглянуто три способи реалізації методу MAI для визначення цінності комп'ютерних компонентів для користувача, а саме з використанням пакетів MS Excel, Super Decisions, Scilab та за допомогою власної програмної реалізації мовою Python. В результаті експериментів з вказаними інструментами було прийнято рішення розробити систему підтримки прийняття рішення мовою Python. Цей вибір зумовлено тим, що

вищеназвані інструменти не зовсім зручні для роботи з великою кількістю альтернатив, а метод МАІ розв'язує лише частину сформульованої задачі, а саме ранжування альтернатив за пріоритетом, але не розв'язує задачу оптимізації. Для розв'язання задачі оптимізації в тому вигляді, як вона сформульована, необхідно побудувати парето-оптимальну множину альтернатив. В результаті було розроблено додаток на мові Python, який включає модулі обчислення вектора пріоритетів альтернатив та перевірки консистентності матриць парних порівнянь, використана бібліотека Python для побудови парето-оптимальної множини альтернатив, і головний модуль, який визначає оптимальне рішення з точки зору КС. Розроблений додаток дозволяє знаходити не тільки оптимальну альтернативну конфігурацію КС, але й оптимальний варіант окремої компоненти КС.

Таким чином, всі поставлені задачі дипломної роботи виконані, що свідчить про досягнення поставленої мети.

Надалі планується продовжити дане дослідження та розробити графічний додаток, що дозволяє візуально проводити синтез альтернатив КС, налаштовувати матриці парних порівнянь, формувати оптимальні конфігурації з будь-якими іншими критеріями для підтримки прийняття рішень. Такий додаток має практичне значення для використання в організаціях, що займаються синтезом, ремонтом, модернізацією КС, в тому числі продажем комп'ютерних компонентів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Meyers M., Everett T. A., Hutz A. Mike Meyers' CompTIA A+ Guide to Managing and Troubleshooting PCs, Seventh Edition. – McGraw Hill, 2023.
2. Multiple Criteria Decision Analysis: State of the Art Surveys. Second Edition / Edited by J. Figueira, S. Greco, M. Ehrgott. – Springer, 2018.
3. Vakiliipour S., Sadeghi-Niaraki A., Ghodousi M., Choi S. Comparison between Multi-Criteria Decision-Making Methods and Evaluating the Quality of Life at Different Spatial Levels. Sustainability 2021, 13, 4067
4. Audin F., Gümüş B. Comparative analysis of multi-criteria decision making methods for the assessment of optimal SVC location, Bull. Pol. Acad. Sci, Tech. Sci., vol. 70(2), 2022, pp. 1-11.
5. Катренко А. В., Пасічник В. В. Прийняття рішень: теорія та практика: підручник. – Львів: «Новий Світ – 2000», 2023. – 447 с.
6. Saaty T. L., Vargas L. G. Models, Methods, Concepts & Applications of the Analytic Hierarchy Process. Second ed. – Springer, 2012.
7. Bouyssou, D., Perny, P. Ranking methods for valued preference relations – a characterization of a method based on leaving and entering flows. Eur. J. Oper. Res. 61(1–2), 1992, pp. 186–194.
8. Dujmović J. Soft Computing Evaluation Logic: The LSP Decision Method and Its Applications. – John Wiley & Sons, 2018.
9. Brunelli M. Introduction to the Analytic Hierarchy Process. – Springer Cham, 2015.
10. Mu E., Pereyra-Rojas M. Practical Decision Making using Super Decisions v3. – Springer Cham, 2017.
11. Project paretoset 1.2.3. URL: <https://pypi.org/project/paretoset> (дата звернення: 10.11.2023).
12. Wilson K. Essential Computer Hardware. – Elluminet Press, 2017.

13. Plunkett J. W. Plunkett's computers, hardware & software industry almanac 2022. – Plunkett Research, Ltd, 2022.
14. Plantz R. G. Introduction to computer organization. – No Starch Press, Inc., 2022.
15. Justice M. How Computers Really Work. – No Starch Press, Inc., 2021.

ДОДАТОК А

Лістинги коду програми

Лістинг А.1 – Модуль для визначення оптимальної альтернативи
жорсткого диска

```
import numpy as np
import pandas as pd
from paretoiset import paretoiset
import matplotlib.pyplot as plt
import math

import ahplib

items = ['WD5000AVDS', 'WD10EZEX', 'WD20EZBX', 'WD40EZZ',
        'ST5000MM002', 'ST1000DM010', 'ST2000DM008',
        'ST4000DM004', 'HDWD105UZSVA', 'HDWD110UZSVA',
        'HDWD320UZSVA', 'HDWD240UZSVA']

# capacity pair comparison: 0.5Tb 1Tb 2Tb 4Tb
capacity = [[1, 0.2, 0.25, 0.33],
            [5, 1, 1.25, 1.67],
            [4, 0.8, 1, 1.33],
            [3, 0.6, 0.75, 1]]

# brand pair comparison: WD, Seagate, Toshiba
brand = [[1, 3, 4],
         [0.33, 1, 1.33],
         [0.25, 0.75, 1]]

# statistics sales of HDD by capacity: 0.5Gb 1Tb 2Tb 4Tb
scap = [[1, 0.4, 1, 2],
        [2.5, 1, 2.5, 5],
        [1, 0.4, 1, 2],
        [0.5, 0.2, 0.5, 1]]

# statistics sales of HDD by vendors: WD, Seagate, Toshiba
sbrand = [[1, 0.75, 1.5],
          [1.33, 1, 2],
          [0.67, 0.5, 1]]
```

```

# criteria pair comarision: cap:brand:cap_stat:brand_stat
criteria = [[1,    4,    2,    3   ],
            [0.25, 1,    0.5,  0.75],
            [0.5,  2,    1,    1.5 ],
            [0.33, 1.33, 0.67, 1   ]]

# compute preference vectors
cap_pref_vect = ahplib.pref_vector(capacity)
brand_pref_vect = ahplib.pref_vector(brand)
scap_pref_vect = ahplib.pref_vector(scap)
sbrand_pref_vect = ahplib.pref_vector(sbrand)
criteria_pref_vect = ahplib.pref_vector(criteria)

# check consistency pair comparision matrix
cap_pref_CR = ahplib.consi(capacity)
brand_pref_CR = ahplib.consi(brand)
scap_pref_CR = ahplib.consi(scap)
sbrand_pref_CR = ahplib.consi(sbrand)
criteria_pref_CR = ahplib.consi(criteria)

#create vectors for convert to pandas DataFrame
V_cap = []
V_brand = []
V_scap = []
V_sbrand = []
for i in range(0, len(brand_pref_vect)):
    for j in range(0, len(cap_pref_vect)):
        V_cap.append(cap_pref_vect[j])
        V_scap.append(scap_pref_vect[j])

for i in range(0, len(cap_pref_vect)):
    V_brand.append(brand_pref_vect[0])
    V_sbrand.append(sbrand_pref_vect[0])
for i in range(0, len(cap_pref_vect)):
    V_brand.append(brand_pref_vect[1])
    V_sbrand.append(sbrand_pref_vect[1])
for i in range(0, len(cap_pref_vect)):
    V_brand.append(brand_pref_vect[2])

```

```

V_sbrand.append(sbrand_pref_vect[2])

matrix1 = pd.DataFrame({"cap":V_cap, "brand":V_brand,
                        "scap":V_scap,"sbrand":V_sbrand})
ranks = matrix1.dot(criteria_pref_vect)
ranks_list = ranks.values.tolist()
ranks_max = max(ranks_list)
perf = []
for i in range(0,len(ranks_list)):
    perf.append(100*ranks_list[i]/ranks_max)
cost = [920, 1700, 2400, 3850, 900, 1650, 1900, 3600,
        860, 1300, 2200, 3250]

hdd1 = pd.DataFrame({"perf":perf,"cost":cost})
hdd = hdd1.sort_values(by=['perf'])
print('sorted hdd:')
print(hdd)

mask = paretoiset(hdd,sense=["max","min"])
paretoiset_hdd = hdd[mask]
print(paretoiset_hdd)

pareto_perf = np.array(paretoiset_hdd.perf)
pareto_cost = np.array(paretoiset_hdd.cost)

tg = [] #tangents
for i in range(0,len(pareto_perf)):
    tg.append(pareto_perf[i]/pareto_cost[i])

tg_max = max(tg)

idx = tg.index(tg_max)
hdd_optimum_perf = pareto_perf[idx]
hdd_optimum_cost = pareto_cost[idx]

idx_optimum_hdd = perf.index(hdd_optimum_perf)

print('optimum choice:', items[idx_optimum_hdd])

```

```

print('perf:', hdd_optimum_perf, 'cost:', hdd_optimum_cost)
print(idx, tg_max)

plt.xlabel('cost')
plt.ylabel('perf, %')
plt.xlim(0, 4000)
plt.ylim(0, 100)
plt.plot(cost, perf, 'o', color='gray', label='alternatives')
plt.plot(pareto_cost, pareto_perf, '-og',
         label='Pareto frontier')
plt.axhline(y=90, xmin=0, xmax=4000, color='gray',
            linestyle='dashed', label='Performance 90%')
plt.axline((0, 0), slope=tg_max, color='black', linestyle='--')
plt.axvline(x=1500, color='black', linestyle='dashdot',
            label='max_cost')
plt.plot(pareto_cost[idx], pareto_perf[idx], 'or',
         label='max_eff')
plt.legend(loc='lower right')
plt.show()

```

Лістинг А.2 – Головний модуль програми

```

import numpy as np
import pandas as pd
from paretoset import paretoset
import matplotlib.pyplot as plt
import math
from timeit import default_timer as timer
import hddmodule
import monitornextmodule
import processormodule
import motherboardIntelmodule
import motherboardAMDmodule
import casemodule
import ahplib

start = timer()
proc_info = processormodule.proc()
mother_Intel_info = motherboardIntelmodule.motherIntel()

```

```

mother_AMD_info = motherboardAMDmodule.motherAMD()
hdd_info = hddmodule.hdd()
monitor_info = monitornextmodule.monitore()
case_info = casemodule.case()

proc_list_perf = proc_info['ranks'].values.tolist()
proc_list_cost = proc_info['cost'].values.tolist()
proc_list_items = proc_info['items'].values.tolist()
procIntel_list_perf = proc_list_perf[0:3]
procIntel_list_cost = proc_list_cost[0:3]
procIntel_list_items = proc_list_items[0:3]
procAMD_list_perf = proc_list_perf[3:6]
procAMD_list_cost = proc_list_cost[3:6]
procAMD_list_items = proc_list_items[3:6]

motherIntel_list_perf = mother_Intel_info['ranks'].values.tolist()
motherIntel_list_cost = mother_Intel_info['cost'].values.tolist()
motherIntel_list_items = mother_Intel_info['items'].values.tolist()

motherAMD_list_perf = mother_AMD_info['ranks'].values.tolist()
motherAMD_list_cost = mother_AMD_info['cost'].values.tolist()
motherAMD_list_items = mother_AMD_info['items'].values.tolist()

hdd_list_perf = hdd_info['ranks'].values.tolist()
hdd_list_cost = hdd_info['cost'].values.tolist()
hdd_list_items = hdd_info['items'].values.tolist()
monitor_list_perf = monitor_info['ranks'].values.tolist()
monitor_list_cost = monitor_info['cost'].values.tolist()
monitor_list_items = monitor_info['items'].values.tolist()
case_list_perf = case_info['ranks'].values.tolist()
case_list_cost = case_info['cost'].values.tolist()
case_list_items = case_info['items'].values.tolist()
#criteria pair comparision proc:motherboard:hdd:monitor:case
criteria = [[1,      2,      5,      7,      9      ],
            [0.5,    1,      2.5,    3.5,    4.5    ],
            [0.2,    0.4,    1,      1.4,    1.8    ],
            [0.143, 0.286, 0.714, 1,      1.286],
            [0.111, 0.222, 0.556, 0.778, 1      ]]

```

```

criteria_pref_vect = ahplib.pref_vector(criteria)
criteria_pref_CR   = ahplib.consi(criteria)
counter = 0
models_list = []
for i in range(0, len(procIntel_list_items)):           #Intel
    for m in range(0, len(motherIntel_list_items)):
        for j in range(0, len(hdd_list_items)):
            for k in range(0, len(monitor_list_items)):
                for q in range(0, len(case_list_items)):
                    model_info=[]
                    model = 'model-'+str(counter)+' (Intel)'
                    model_info.append(model)
                    model_info.append(procIntel_list_perf[i])
                    cost = procIntel_list_cost[i]
                    config = procIntel_list_items[i]+'; '
                    model_info.append(motherIntel_list_perf[m])
                    cost = cost+motherIntel_list_cost[m]
                    config = config+motherIntel_list_items[m]+'; '
                    model_info.append(hdd_list_perf[j])
                    cost = cost+hdd_list_cost[j]
                    config = config+hdd_list_items[j]+'; '
                    model_info.append(monitor_list_perf[k])
                    cost = cost+monitor_list_cost[k]
                    config = config+monitor_list_items[k]+'; '
                    model_info.append(case_list_perf[q])
                    cost = cost+case_list_cost[q]
                    config = config+case_list_items[q]
                    model_info.append(cost)
                    model_info.append(config)
                    models_list.append(model_info)
                    counter = counter+1
for i in range(0, len(procAMD_list_items)):             #AMD
    for m in range(0, len(motherAMD_list_items)):
        for j in range(0, len(hdd_list_items)):
            for k in range(0, len(monitor_list_items)):
                for q in range(0, len(case_list_items)):
                    model_info=[]
                    model = 'model-'+str(counter)+' (AMD)'

```

```

        model_info.append(model)
        model_info.append(procAMD_list_perf[i])
        cost = procAMD_list_cost[i]
        config = procAMD_list_items[i]+'; '
        model_info.append(motherAMD_list_perf[m])
        cost = cost+motherAMD_list_cost[m]
        config = config+motherAMD_list_items[m]+'; '
        model_info.append(hdd_list_perf[j])
        cost = cost+hdd_list_cost[j]
        config = config+hdd_list_items[j]+'; '
        model_info.append(monитор_list_perf[k])
        cost = cost+monитор_list_cost[k]
        config = config+monитор_list_items[k]+'; '
        model_info.append(case_list_perf[q])
        cost = cost+case_list_cost[q]
        config = config+case_list_items[q]
        model_info.append(cost)
        model_info.append(config)
        models_list.append(model_info)
        counter = counter+1

V_proc_pref=[]
V_mother_pref = []
V_hdd_pref=[]
V_monitor_pref=[]
V_case_pref = []
V_config_cost=[]
V_config_items=[]

for i in range(0,len(models_list)):
    V_proc_pref.append(models_list[i][1])
    V_mother_pref.append(models_list[i][2])
    V_hdd_pref.append(models_list[i][3])
    V_monitor_pref.append(models_list[i][4])
    V_case_pref.append(models_list[i][5])
    V_config_cost.append(models_list[i][6])
    V_config_items.append(models_list[i][7])
matrix=pd.DataFrame({"proc":V_proc_pref,"mother":V_mother_pref,
    "hdd":V_hdd_pref,"monitor":V_monitor_pref,"case":V_case_pref})

```

```

config_ranks = matrix.dot(criteria_pref_vect)
config_ranks_list = config_ranks.values.tolist()
config_ranks_max = max(config_ranks_list)
perf=[]
cost=[]
for i in range(0,len(config_ranks_list)):
    perf.append(100*config_ranks_list[i]/config_ranks_max)
    cost.append(V_config_cost[i])

config_ranks_df=pd.DataFrame({'perf':perf, 'cost':cost})
config_ranks_df_sorted=config_ranks_df.sort_values(by=['perf'])
mask = paretoiset(config_ranks_df_sorted,sense=["max", "min"])
config_paretoiset = config_ranks_df_sorted[mask]
config_pareto_perf=np.array(config_paretoiset.perf)
config_pareto_cost=np.array(config_paretoiset.cost)
tg = [] #tangents

for i in range(0,len(config_pareto_perf)):
    tg.append(config_pareto_perf[i]/config_pareto_cost[i])

tg_max = max(tg)
idx = tg.index(tg_max)
config_optimum_perf=config_pareto_perf[idx]
config_optimum_cost=config_pareto_cost[idx]
idx_config_optimum = perf.index(config_optimum_perf)
#config_optimum = V_config_items[idx_config_optimum]
config_optimum = models_list[idx_config_optimum][7]
model_optimum = models_list[idx_config_optimum][0]
print(model_optimum,config_optimum)
idx_of_pareto_front = []
for i in range(0,len(config_paretoiset)):
    idx_of_pareto_front=perf.index(config_pareto_perf[i])
    model_pareto = models_list[idx_of_pareto_front][0]
    config_pareto = models_list[idx_of_pareto_front][7]
    print(model_pareto,config_pareto,',';', '%.1f'%
    config_pareto_perf[i], '%, ', config_pareto_cost[i])

print(config_optimum_perf,config_optimum_cost)

```

```

end = timer()
print('elapsed time: ',end-start,'sec') # time in second
max_cost = 19000
plt.xlabel('cost')
plt.ylabel('perf, %')
plt.xlim(10000,30000)
plt.ylim(0,105)
plt.plot(cost,perf,'o',color='gray',markersize=1,
         label='alternatives')
plt.plot(config_pareto_cost,config_pareto_perf,
         '-og',markersize=3.5,label='Pareto frontier')
plt.axline((0,0),slope =tg_max, color='black', linestyle="--")
plt.axhline(y = 90, xmin=0, xmax=30000,color='red',
            linestyle='dashed', label='Performance 90%')
plt.axvline(x = max_cost, color = 'red',linestyle="dashdot",
            label="max_cost="+str(max_cost))
plt.legend(loc = 'lower right')
plt.show()

```