

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «магістр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

«Модифікована задача розподілу ресурсів виробничого підприємства»

Виконав студент групи ПЗм-23-1
ЧИЖМОТРЯ Максим Олексійович

Керівник роботи:
ЧИЖМОТРЯ Олексій Володимирович

Рецензент:
ЛЕВКІВСЬКИЙ Віталій Леонідович

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«01» жовтня 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ЧИЖМОТРИЯ Максим Олексійович**

Керівник роботи: **ЧИЖМОТРИЯ Олексій Володимирович**

Тема роботи: **«Модифікована задача розподілу ресурсів виробничого підприємства»**, затверджена Наказом закладу вищої освіти від **«1» жовтня 2024 р., № 520/с**

Вихідні дані для роботи: **процеси розподілу ресурсів у виробничих підприємствах з урахуванням виробничих циклів, методи та алгоритми розв'язання модифікованої задачі розподілу ресурсів із урахуванням тривалості виробничих циклів.**

Керівники / консультанти з магістерської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Керівник/ консультант	Завдання видав	Завдання прийняв
1	Чижмотря О. В.	01.10.2024р.	01.10.2024р.
2	Чижмотря О. В.	01.10.2024р.	01.10.2024р.
3	Чижмотря О. В.	01.10.2024р.	01.10.2024р.

РЕФЕРАТ

Кваліфікаційна робота освітнього ступеня «магістр» присвячена модифікованій задачі розподілу ресурсів виробничого підприємства. Пояснювальна записка складається з вступу, трьох розділів та списку використаних джерел. Пояснювальна записка до кваліфікаційної роботи містить 68 сторінок, 18 ілюстрацій, 12 таблиць, 3 додатки та 40 посилань на зовнішні джерела.

Метою роботи є розробка методу та алгоритмів для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства з урахуванням тривалості виробничих циклів. Реалізація цих алгоритмів у вигляді веб-орієнтованої системи спрямована на підвищення ефективності управління виробничими процесами.

У роботі описано математичну модель модифікованої задачі розподілу ресурсів виробничого підприємства, розроблено метод розв'язання модифікованої задачі та метод для аналізу запасів виробничих ресурсів підприємства. Проведено аналіз ефективності розроблених методів. Розроблено прототип вебдодатку для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства на основі відповідного методу.

КЛЮЧОВІ СЛОВА: ПЛАН ВИРОБНИЦТВА, ВИРОБНИЧИЙ ЦИКЛ, ВИРОБНИЧІ ЗАПАСИ, СИМПЛЕКС-МЕТОД, АЛГОРИТМ ГОМОРИ, ВЕБДОДАТОК.

ABSTRACT

The qualification work of the educational degree “master” is devoted to the modified problem of resource allocation of a production enterprise. The explanatory note consists of an introduction, three chapters and a list of references. The explanatory note to the qualification work contains 68 pages, 18 illustrations, 12 tables, 3 appendices and 40 references to external sources.

The purpose of the work is to develop a method and algorithms for solving a modified problem of resource allocation of a manufacturing enterprise, taking into account the duration of production cycles. The implementation of these algorithms in the form of a web-based system is aimed at improving the efficiency of production process management.

The paper describes the mathematical model of the modified problem of resource allocation of a manufacturing enterprise, develops a method for solving the modified problem and a method for analyzing the enterprise's production resource stocks. The efficiency of the developed methods is analyzed. A prototype of a web application for solving the modified problem of resource allocation of a manufacturing enterprise based on the corresponding method has been developed.

KEYWORDS: PRODUCTION PLAN, PRODUCTION CYCLE, PRODUCTION INVENTORY, SIMPLEX METHOD, GOMORI ALGORITHM, WEB APPLICATION.

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	6
ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	9
1.1. Задачі економіко-математичного моделювання: основи теорії та практичні аспекти	9
1.2. Аналіз вхідних даних задачі розподілу ресурсів виробничого підприємства	13
1.3. Постановка задачі дослідження	15
Висновки до першого розділу	17
РОЗДІЛ 2. МЕТОДИ РОЗВ'ЯЗАННЯ МОДИФІКОВАНОЇ ЗАДАЧІ РОЗПОДІЛУ РЕСУРСІВ	18
2.1. Математична модель та алгоритм розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства	18
2.2. Оцінка ефективності методу розв'язання модифікованої задачі	25
2.3. Розробка методу для аналізу наявних ресурсів підприємства	34
Висновки до другого розділу	37
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ	38
3.1. Визначення архітектури та засобів реалізації прототипу вебдодатку	38
3.2. Реалізація окремих модулів прототипу вебдодатку	42
3.3. Опис інтерфейсу та порядок роботи із вебдодатком	46
Висновки до третього розділу	51
ВИСНОВКИ	53
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	54
ДОДАТКИ	58

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ЛП – лінійне програмування

МП – математичне програмування

ВЦ – виробничий цикл

ЗЛП – задача лінійного програмування

СМ – симплексний метод

ПЗ – програмне забезпечення

БД – база даних

СКБД – система керування базами даних

ВСТУП

Актуальність теми. Сучасні виробничі підприємства стикаються з необхідністю ефективного управління ресурсами для забезпечення конкурентоспроможності та підвищення продуктивності. Особливої уваги потребують задачі розподілу ресурсів, які враховують час на виконання плану виробництва, що суттєво впливає на досягнення оптимальних результатів.

Актуальність теми «Модифікована задача розподілу ресурсів виробничого підприємства» зумовлена потребою розробки моделей і методів, які дозволяють знаходити ефективні рішення з урахуванням тривалості виробничих циклів, оптимального використання ресурсів і мінімізації витрат. Розв'язування таких задач є критично важливим для підприємств, які працюють у динамічних умовах ринку з обмеженими ресурсами.

Метою роботи є розробка методу та алгоритмів для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства з урахуванням тривалості виробничих циклів. Реалізація цих алгоритмів у вигляді веб-орієнтованої системи спрямована на підвищення ефективності управління виробничими процесами.

Поставлена мета обумовлює необхідність виконання наступних завдань:

- дослідження існуючих задач економіко-математичного моделювання;
- розробка математичної моделі модифікованої задачі розподілу ресурсів;
- визначення підходу для отримання цілочисельних результатів;
- проведення аналізу напрямків використання інформаційних технологій для розробки прототипу вебдодатку для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства;
- розробка методу аналізу запасів ресурсів;
- проектування алгоритму розв'язання модифікованої задачі розподілу ресурсів;
- реалізація прототипу вебдодатку.

Об'єктом дослідження є процеси розподілу ресурсів у виробничих підприємствах з урахуванням виробничих циклів.

Предметом дослідження є методи та алгоритми розв'язання модифікованої задачі розподілу ресурсів із урахуванням тривалості виробничих циклів.

Наукова новизна. Наукова новизна роботи полягає у розробці власної математичної моделі та методу розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства, розробці власного методу для аналізу наявних ресурсів підприємства, вдосконаленні підходів до розподілу ресурсів виробничого підприємства шляхом введення до математичної моделі параметру тривалості виробничого циклу для кожного виду продукції. Це дозволяє підвищити точність оптимізаційних розрахунків за рахунок врахування часових характеристик виробничих процесів, адаптувати класичні методи оптимізації, такі як симплекс-метод і алгоритм Гоморі, для задач із додатковими обмеженнями, пов'язаними з виробничими циклами, а також розширити можливості математичного моделювання для задач управління виробничими процесами в сучасних умовах.

Практичне значення. Практичне значення роботи полягає у створенні вебдодатку, який може бути використаний на виробничих підприємствах для вирішення реальних задач оптимізації розподілу ресурсів з урахуванням тривалості виробничих циклів. Це сприятиме підвищенню ефективності управлінських рішень, мінімізації витрат на виробництво та покращенню використання ресурсів, включаючи трудові, матеріальні та часові. Крім того, результати дослідження можуть стати основою для подальших наукових розробок у сфері оптимізації виробничих процесів, вдосконалення існуючих методів і алгоритмів розподілу ресурсів, а також адаптації запропонованих підходів для інших галузей. Розроблена модель також може знайти застосування в навчальному процесі та дослідницькій діяльності, сприяючи поглибленню знань у галузі математичного моделювання та оптимізації виробничих систем.

Апробація. За темою кваліфікаційної магістерської роботи було опубліковано 2 тези конференцій [1], [2].

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Задачі економіко-математичного моделювання: основи теорії та практичні аспекти

Економіко-математичне моделювання передбачає застосування математичних методів і моделей для дослідження економічних процесів і явищ. Основною метою цього підходу є формалізація економічних зв'язків і їх кількісна оцінка, що дозволяє прогнозувати розвиток подій і приймати зважені рішення. Розробка таких методів почалася ще в першій половині XX століття, і вони набули широкого поширення в макроекономічному управлінні, бізнес-плануванні та фінансовому аналізі.

Одним із провідних інструментів економіко-математичного моделювання є лінійне програмування, яке забезпечує оптимізацію використання ресурсів за певних обмежень. Лінійне програмування (ЛП) – це розділ математики, в якому розглядаються методи розв'язування екстремальних задач з лінійним функціоналом і лінійними обмеженнями, яким повинні задовольняти шукані змінні [3]. У свою чергу, нелінійне програмування дозволяє враховувати більш складні економічні залежності, які не піддаються лінійній інтерпретації. Крім того, статистичні методи відіграють ключову роль у аналізі даних і створенні моделей, що враховують випадкові зміни та невизначеність [4].

Розглянемо приклади основних оптимізаційних задач.

Задача використання сировини є важливим аспектом управління виробництвом, що полягає в оптимальному розподілі обмежених ресурсів для максимізації прибутку або зменшення витрат. У цій задачі підприємство має певну кількість сировини, яку потрібно використовувати для виготовлення різних продуктів. Кожен продукт має свою потребу в сировині, а також приносить певний прибуток або має певну цінність. Основна мета задачі полягає в тому, щоб визначити, скільки одиниць кожного продукту слід виготовити, щоб максимально використати доступну сировину, при цьому дотримуючись обмежень, таких як максимальна кількість сировини, що може бути використана,

або обмеження на виробничі потужності. Задача може включати різні фактори, такі як витрати на виробництво, ринкові ціни на готову продукцію, а також можливі обмеження на кількість виготовлених одиниць. Задача використання сировини може бути сформульована як задача лінійного програмування, де цільова функція максимізує прибуток, а обмеження враховують доступну кількість сировини та інші виробничі обмеження. Вирішення цієї задачі дозволяє підприємствам ефективно планувати виробництво, знижувати витрати та підвищувати конкурентоспроможність на ринку. Застосування таких методів, як симплекс-метод або динамічне програмування, допомагає знайти оптимальні рішення для задачі використання сировини, що в свою чергу сприяє більш ефективному управлінню ресурсами [5].

Яскравим представником оптимізаційних задач є також *задача про рюкзак*. Суть задачі полягає в оптимізації вибору предметів для заповнення рюкзака з обмеженою місткістю, щоб максимізувати загальну цінність вибраних предметів. Рюкзак має певну максимальну місткість (вагу), яку не можна перевищити, а кожен предмет має свою вагу і цінність. Мета задачі полягає в тому, щоб вибрати певну комбінацію предметів, яка не перевищує місткість рюкзака, але при цьому має максимальну загальну цінність. Існує кілька варіантів задачі про рюкзак, зокрема 0/1 рюкзак, де кожен предмет можна взяти або не взяти, зважений рюкзак, в якому предмети мають різні ваги, і задача полягає в максимізації цінності при обмеженій загальній вазі, а також дробовий рюкзак, де дозволяється брати частини предметів, що робить задачу більш гнучкою. Задача про рюкзак має численні практичні застосування, включаючи оптимізацію ресурсів у виробництві, планування фінансів (вибір інвестицій), логістику (упаковка вантажів) та розподіл обмежених ресурсів у різних сферах. Загалом, задача про рюкзак є класичним прикладом комбінаторної оптимізації, і її рішення може бути знайдено за допомогою різних алгоритмів, таких як динамічне програмування, жадібні алгоритми або методи перебору [6].

Задача складання раціону полягає в розробці збалансованого меню для задоволення харчових потреб людини або групи людей при обмежених ресурсах.

Основна мета цієї задачі полягає в тому, щоб визначити оптимальне поєднання продуктів харчування, яке забезпечить необхідну кількість поживних речовин, вітамінів і мінералів, при цьому мінімізуючи витрати або максимізуючи задоволення від їжі. У задачі складання раціону враховуються різні фактори, такі як харчова цінність продуктів, їх вартість, а також індивідуальні потреби в калоріях і поживних речовинах, які можуть варіюватися в залежності від віку, статі, рівня фізичної активності та стану здоров'я. Задача може бути сформульована як задача лінійного програмування, де цільова функція полягає в мінімізації витрат на продукти, а обмеження враховують необхідні норми споживання поживних речовин. Рішення цієї задачі може бути досягнуто за допомогою різних методів, таких як симплекс-метод, динамічне програмування або жадібні алгоритми. Застосування таких методів дозволяє розробити ефективні раціони, які відповідають вимогам здорового харчування та економічним можливостям. Задача складання раціону має численні практичні застосування в дієтології, харчовій промисловості, закладах громадського харчування та в особистому харчуванні. Ефективне вирішення цієї задачі сприяє покращенню здоров'я, підвищенню якості життя та оптимізації витрат на харчування [7].

Основна мета *задачі оптимального розкрою матеріалів* полягає в тому, щоб визначити, як найкраще розрізати великий шматок матеріалу на менші частини, щоб мінімізувати відходи та максимізувати використання сировини, при цьому задовольняючи вимоги до розмірів і форм готових виробів. У задачі оптимального розкрою матеріалів враховуються різні фактори, такі як розміри та кількість необхідних частин, характеристики матеріалу, а також можливі обмеження на розкроювання, наприклад, технологічні особливості або обмеження на використання обладнання. Задача може бути сформульована як задача комбінаторної оптимізації, де цільова функція полягає в мінімізації відходів або максимізації використання матеріалу. Рішення цієї задачі може бути досягнуто за допомогою різних методів, таких як динамічне програмування, жадібні алгоритми або евристичні підходи. Застосування таких методів дозволяє

підприємствам знижувати витрати на сировину, підвищувати ефективність виробництва та зменшувати негативний вплив на навколишнє середовище за рахунок зменшення відходів. Задача оптимального розкрою матеріалів є актуальною в багатьох галузях, включаючи текстильну, меблеву, металургійну та будівельну промисловість, де ефективне використання матеріалів має значний вплив на загальну рентабельність виробництва [8].

Задача комівояжера є класичним прикладом задачі комбінаторної оптимізації, що полягає в знаходженні найкоротшого маршруту для комівояжера, який повинен відвідати певну кількість міст і повернутися до початкової точки. Основна мета цієї задачі полягає в тому, щоб визначити оптимальний порядок відвідування міст, щоб мінімізувати загальну відстань або витрати на подорож. У задачі комівояжера враховуються різні фактори, такі як відстані між містами, можливі маршрути, а також обмеження на час або ресурси, які можуть впливати на вибір маршруту. Задача може бути сформульована як задача графа, де міста представляють собою вершини, а відстані між ними — ребра. Рішення задачі комівояжера є складним, оскільки кількість можливих маршрутів зростає експоненційно з кількістю міст, що робить її NP-складною. Для розв'язання цієї задачі використовуються різні методи, такі як жадібні алгоритми, динамічне програмування, генетичні алгоритми та евристичні підходи. Задача комівояжера має численні практичні застосування в логістиці, плануванні маршрутів, управлінні постачанням та інших сферах, де важливо оптимізувати переміщення між різними точками. Ефективне вирішення цієї задачі дозволяє знижувати витрати на транспортування, покращувати обслуговування клієнтів і підвищувати загальну ефективність бізнес-процесів [9].

Транспортна задача є класичним прикладом задачі лінійного програмування, що полягає в оптимізації процесу транспортування товарів з декількох джерел постачання до декількох пунктів споживання з метою мінімізації загальних витрат на транспортування. Основна мета цієї задачі полягає в тому, щоб визначити оптимальний розподіл товарів між

постачальниками і споживачами, враховуючи обмеження на кількість товарів, які можуть бути доставлені з кожного джерела, а також потреби кожного споживача. У транспортній задачі враховуються різні фактори, такі як витрати на транспортування між кожною парою постачальника і споживача, обсяги товарів, які потрібно перевезти, а також обмеження на постачання і споживання. Задача може бути представлена у вигляді матриці, де рядки відповідають постачальникам, а стовпці – споживачам, а елементи матриці відображають витрати на транспортування. Рішення транспортної задачі може бути знайдено за допомогою різних методів, таких як метод південно-західного кута, метод потенціалів або метод модифікації. Ці методи дозволяють ефективно знаходити оптимальні рішення, які забезпечують мінімальні витрати на транспортування. Транспортна задача має численні практичні застосування в логістиці, управлінні ланцюгами постачання, плануванні виробництва та інших сферах, де важливо оптимізувати процеси перевезення товарів. Ефективне вирішення цієї задачі дозволяє знижувати витрати, покращувати обслуговування клієнтів і підвищувати загальну ефективність бізнес-процесів [10].

Економіко-математичне моделювання залишається ключовим інструментом аналізу й прогнозування в сучасній економіці. Попри виклики, такі як складність моделей і потреба в якісних даних, розвиток технологій і методологічних підходів сприяє подоланню цих труднощів та відкриває нові перспективи.

1.2. Аналіз вхідних даних задачі розподілу ресурсів виробничого підприємства

Задача про оптимальне використання ресурсів виробничого підприємства є однією з основних планово-виробничих і економічних задач та відноситься до задач математичного програмування (МП).

Більшість таких задач пов'язані з розподілом певних, як правило, обмежених ресурсів. Виходячи з того, що ресурси можна використати у різній кількості або у різних пропорціях, можна отримати певний набір можливих

результатів. Результат завжди оцінюється у вигляді певного критерію (прибуток, обсяг продукції, затрачений час і т.д.) або декількох з них. Таким чином, суть задач МП можна звести до пошуку такого варіанту розподілу наявних ресурсів, при якому результуючий критерій буде гарантувати найбільший економічний ефект [11].

Для задачі про оптимальне використання ресурсів метою є максимізація прибутку. У класичному формулюванні задачу про оптимальне використання ресурсів можна подати у наступному вигляді: підприємство виготовляє n видів продукції ($A_1, A_2, \dots, A_n$). Для цього використовуються m видів сировини/матеріалів ($B_1, B_2, \dots, B_m$). Відомі запаси сировини/матеріалів ($b_1, b_2, \dots, b_m$) та прибуток від реалізації одиниці продукції кожного виду ($p_1, p_2, \dots, p_n$).

Вхідні дані зручно подавати у вигляді таблиці (табл. 1.1):

Таблиця 1.1

Таблична форма запису задачі

Види ресурсів	Види продукції			Запаси ресурсів
	A_1	...	A_n	
B_1	a_{11}	...	a_{1n}	b_1
...	...	...	...	...
B_m	a_{m1}	...	a_{mn}	b_m
Прибуток	p_1	...	p_n	

У таблиці через $a_{11}, \dots, a_{1n}, \dots, a_{m1}, \dots, a_{mn}$ позначені потреби у відповідному виді ресурсу для виготовлення одиниці відповідного виду продукції.

Потрібно скласти такий план виробництва продукції, який принесе максимальний прибуток при заданих запасах.

Математична модель задачі буде виглядати наступним чином:

$$P = p_1q_1 + p_2q_2 + \dots + p_nq_n \rightarrow \max, \quad (1.1)$$

де $q_1, q_2, \dots, q_n$ – кількість одиниць продукції кожного виду.

Система обмежень:

$$\begin{cases} a_{11}q_1 + a_{12}q_2 + \dots + a_{1n}q_n \leq b_1, \\ a_{21}q_1 + a_{22}q_2 + \dots + a_{2n}q_n \leq b_2, \\ \dots \\ a_{m1}q_1 + a_{m2}q_2 + \dots + a_{mn}q_n \leq b_m, \end{cases} \quad (1.2)$$

при цьому $q_1 \geq 0, q_2 \geq 0, \dots, q_n \geq 0$ [12].

Початковий аналіз вхідних даних дозволяє зробити певні висновки щодо практичного застосування даної задачі. У разі наявності певних фіксованих запасів ресурсів та/або коли мова йде про одиничне виробництво, то дана задача є актуальною та незамінною, а існуючий алгоритм її розв'язання дійсно підбере оптимальний план виробництва з максимальним прибутком на виході. Проте, в умовах серійного або масового виробництва задача у поточному її формулюванні та з існуючим алгоритмом розв'язання підходить для практичного застосування вже не так очевидно або далеко не завжди.

Згадаємо таке поняття як «виробничий цикл» (ВЦ) та наведемо один з варіантів його визначення. Виробничий цикл – це інтервал від початку до закінчення процесу виготовлення продукції, тобто час, протягом якого запуснені у виробництво предмети праці перетворюються на готову продукцію [13].

Для різних видів продукції тривалість виробничого циклу у реальних умовах, зазвичай, різна. На підставі цього факту можна зробити припущення, що за рахунок меншої тривалості виробничого циклу можна отримати більший прибуток за одиницю часу, дотримуючись іншого плану виробництва, ніж отриманого за алгоритмом розв'язання класичної задачі розподілу ресурсів. Цього неможливо досягти у межах одного виробничого циклу, але стає реальним у умовах виробництва безперервного. Виходячи з вищенаведеного, робимо висновок про актуальність подальших досліджень з даної теми.

1.3. Постановка задачі дослідження

Впровадження модифікації до задачі розподілу ресурсів виробничого підприємства у вигляді тривалостей виробничих циклів має змогу покращити часові, фінансові та інші затрати на виробництві. Використання програми

дозволить автоматизувати процес розв'язання задачі з розподілу ресурсів, що значно збільшить швидкість та точність отримання оптимального розподілу ресурсів. Порівняно з ручним розв'язанням, програма може обробити значно більші обсяги даних та забезпечити більш точні результати.

Розроблений прототип може бути налаштований та адаптований під конкретні потреби та обмеження організації. Тривалості виробничих циклів можуть бути задані окремо під кожен вид виробів, що дозволить знайти оптимальне рішення, яке відповідає конкретним потребам та вимогам. Розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства за допомогою програми може допомогти знизити витрати на ресурси. Оптимальне розподілення ресурсів дозволить мінімізувати зайві витрати та збільшити ефективність виробництва.

Розробка програми дозволить проводити сценарне планування та виконувати аналіз різних варіантів розподілу ресурсів. Використовуючи різні комбінації запасів ресурсів і тривалостей виробничих циклів, можна здійснити дослідження та знайти оптимальний варіант, що найкраще відповідає потребам. Програма може надати зручний інтерфейс для введення даних та відображення результатів. Візуалізація допоможе краще розуміти та аналізувати отримані результати, а також спростить взаємодію з програмою.

Створення прототипу для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства починається з аналізу вимог. Цей крок включає в себе ретельний аналіз щодо вимог та функціональності системи. На цьому кроці необхідно дати відповіді на наступні запитання: які можливості має надавати додаток, які дані потрібно обробляти, які операції потрібно виконувати та які мають бути основні функціональні модулі. При проектуванні інтерфейсу користувача потрібно визначитись, які елементи потрібно включити, які функції та опції повинні бути доступні користувачам. Також потрібно розглянути зручність взаємодії та візуалізацію результатів.

Створення серверної частини додатку, яка буде відповідати за обробку та розв'язання модифікованої задачі розподілу ресурсів включає в себе

використання відповідних алгоритмів та методів, розробку логіки обробки даних та алгоритми оптимізації. Наступним кроком є розробка клієнтської частини вебдодатку, яка буде взаємодіяти з сервером та надавати можливості користувачам вводити дані, налаштовувати параметри, відображати результати та здійснювати інші необхідні операції. Необхідно забезпечити зручний та інтуїтивно зрозумілий інтерфейс.

Для перевірки системи на придатність до використання необхідно провести ретельне тестування системи, перевірити правильність розв'язання модифікованої задачі розподілу ресурсів, а також функціональність інтерфейсу. За потреби, – виправити помилки та виявлені проблеми.

Отже, розробка прототипу для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства має потенціал покращити ефективність, знизити витрати та забезпечити оптимальний розподіл ресурсів, що має значення для різних галузей та організацій.

Висновки до першого розділу

В даному розділі було проведено аналіз предметної області дослідження. Були розглянуті приклади основних оптимізаційних задач, а саме задача використання сировини, задача про рюкзак, задача складання раціону, задача оптимального розкрою матеріалів, задача комівояжера та транспортна задача.

Також було проведено аналіз вхідних даних задачі розподілу ресурсів виробничого підприємства. На основі проведеного аналізу було зроблено висновки про доцільність врахування показника тривалості виробничого циклу та актуальність подальших досліджень з даної теми.

Виконана постановка задачі дослідження. Був визначений перелік завдань, які необхідно виконати для реалізації прототипу вебдодатку для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства.

РОЗДІЛ 2. МЕТОДИ РОЗВ'ЯЗАННЯ МОДИФІКОВАНОЇ ЗАДАЧІ РОЗПОДІЛУ РЕСУРСІВ

2.1. Математична модель та алгоритм розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства

Варто розпочати цей розділ з того, що «класична» задача використання ресурсів завдяки використанню симплексного методу на виході надає оптимальний план виробництва, при якому досягається максимальний прибуток. Тобто, ми отримуємо те, що вже неможливо покращити навіть теоретично.

У реальних умовах виробництва кожний вид продукції має певне значення тривалості виробничого циклу. І це є тим, що жодним чином не враховується у «класичній» задачі. А це, у свою чергу, означає, що «класична» задача дуже обмежена у своїх можливостях щодо її використання в реальних умовах.

«Класична» задача може бути використана або за умови однакових показників тривалості виробничого циклу для всіх видів продукції, або за умови одинарного (разового) виробництва. Обидва допустимі випадки є фактично нереальними для виробничих підприємств із серійним або масовим виробництвом.

Серійне виробництво – це тип виробничого процесу, при якому виготовлення продукції здійснюється у визначених обсягах або серіях. Цей метод передбачає виробництво певної кількості однорідних виробів за один раз, що дозволяє оптимізувати використання ресурсів, зменшити витрати та підвищити ефективність виробництва. Цей тип виробництва часто використовується в промисловості, де є потреба у виготовленні великої кількості однакових або схожих виробів, таких як автомобілі, електроніка, побутова техніка та інші товари [14].

Масове виробництво – це тип виробничого процесу, при якому виготовляється велика кількість однорідних товарів або продукції за стандартними технологічними процесами. Основними характеристиками масового виробництва є високий обсяг виготовлення, що дозволяє задовольнити

великий попит на товари, а також стандартизація виробничих процесів, матеріалів і готової продукції, що забезпечує однорідність і високу якість товарів. Масове виробництво широко використовується в таких галузях, як автомобілебудування, електроніка, харчова та текстильна промисловість. Прикладами можуть бути виробництво автомобілів, побутової техніки, упаковки та інших товарів, які користуються великим попитом [14].

Сама ідея дослідження виникла із наступного припущення: якщо ми не можемо вплинути на загальний прибуток від реалізації того чи іншого плану виробництва (а саме, збільшити його), то, можливо, за рахунок різної тривалості виробничих циклів для різних видів продукції ми зможемо збільшити показник прибутку у розрахунку на одиницю часу. Адже будь-який план виробництва має такий показник як період часу на його реалізацію чи виконання. Якщо виходити з того, що нам відома тривалість виробничого циклу для кожного виду продукції, то наступним припущенням, яке було зроблено, стало припущення про те, що теоретично можна розробити такий метод, який гарантовано знайде оптимальний план виробництва, але вже відносно прибутку за одиницю часу.

Для створення математичної моделі модифікованої задачі розподілу ресурсів виробничого підприємства та алгоритму її розв'язання було розглянуто та проаналізовано різні можливі підходи задля досягнення кінцевої мети. Серед невдалих спроб отримати адекватну математичну модель та алгоритм, який дійсно зможе розв'язати поставлену задачу, можна виокремити спробу із використанням вагових коефіцієнтів. Припускалося, що вплив показника тривалості виробничого циклу кожного з видів продукції на ті чи інші початкові дані «класичної» задачі у вигляді вагових коефіцієнтів зможуть призвести до успіху. Проте й на жаль, максимум, чого ми змогли досягти за цими моделями, це певне збільшення показника прибутку за одиницю часу без гарантії його оптимальності, тобто максимуму. При цьому було важливо залишатись у межах задачі лінійного програмування, адже «технічний» етап пошуку оптимального рішення мав би залишатися за симплексним методом.

Далі будемо розглядати вже саме ту модель та той метод, які змогли розв'язати поставлену задачу та отримати дійсно оптимальне значення прибутку за одиницю часу.

Отже, модифікуємо «класичну» задачу розподілу ресурсів шляхом введення до умови задачі поняття виробничого циклу відповідної тривалості для кожного виду продукції ($t_1, t_2, \dots, t_n$).

На першому етапі роботи розробленого методу маємо розв'язати «класичну» задачу, описану у пункті 1.2. Результатом розв'язання цієї задачі буде безпосередньо план виробництва продукції ($q_1, q_2, \dots, q_n$) та максимально можливий при цьому прибуток P .

Оскільки нам вже відома тривалість виробничого циклу по кожному виду продукції, то для будь-якого плану виробництва таким чином з'являється можливість визначити, скільки часу (TQ) необхідно витратити для виготовлення відповідної кількості кожного з видів продукції:

$$TQ = t \times q. \quad (2.1)$$

Найбільше з отриманих значень позначимо TQ_{max} :

$$TQ_{max} = \max\{TQ_1, TQ_2, \dots, TQ_n\}. \quad (2.2)$$

Прибуток за одиницю часу (PT) будемо шукати за формулою:

$$PT = P / TQ_{max}. \quad (2.3)$$

Слід зазначити, що саме значення прибутку за одиницю часу PT ми будемо максимізувати протягом подальшої роботи розробленого методу. Це й буде цільовою функцією нашої задачі надалі. Отже,

$$PT \rightarrow \max. \quad (2.4)$$

Перед початком другого етапу роботи методу маємо внести зміни до системи обмежень шляхом додавання нових. Нові обмеження будуть стосуватися лімітів по часу виконання плану виробництва. Кількість нових обмежень дорівнюватимуть кількості видів товарів для виробництва. Запишемо оновлену систему обмежень у наступному вигляді:

$$\left\{ \begin{array}{l} a_{11}q_1 + a_{12}q_2 + \dots + a_{1n}q_n \leq b_1, \\ a_{21}q_1 + a_{22}q_2 + \dots + a_{2n}q_n \leq b_2, \\ \dots \\ a_{m1}q_1 + a_{m2}q_2 + \dots + a_{mn}q_n \leq b_m, \\ \\ TQ_1 \leq TQ^*_{max}, \\ TQ_2 \leq TQ^*_{max}, \\ \dots \\ TQ_n \leq TQ^*_{max}, \end{array} \right. \quad (2.5)$$

при цьому $q_1 \geq 0, q_2 \geq 0, \dots, q_n \geq 0$.

TQ^*_{max} – це максимально допустимий час на виконання плану виробництва, який буде зменшуватись на 1 для кожної наступної ітерації роботи методу, починаючи з другої.

Математична модель набуває наступного вигляду:

$$PT = \max\{P_i / TQ_{max\ i}\}, \quad (2.6)$$

де значення i буде зменшуватись від значення TQ_{max} до t_{min} .

Результатом кожної наступної ітерації буде оптимальний план виробництва продукції ($q_1, q_2, \dots, q_n$) при заданих обмеженнях та максимально можливий при цьому прибуток P .

Після розрахунків за формулами (2.1)-(2.3) для кожної ітерації буде отримано значення прибутку за одиницю часу PT . Якщо отримане значення буде більшим за поточний максимум, то значення максимум буде оновлено.

Залежно від початкових умов «класичної» задачі розподілу ресурсів та тривалості виробничих циклів отримані на кожній з ітерацій результати розрахунків можуть утворювати принципово різні набори значень з різним трендом.

Найчастіше утворюються такі набори вихідних значень, у яких присутні різні значення цільової функції: як менші за початкове значення, так й більші за нього (рис. 2.1).

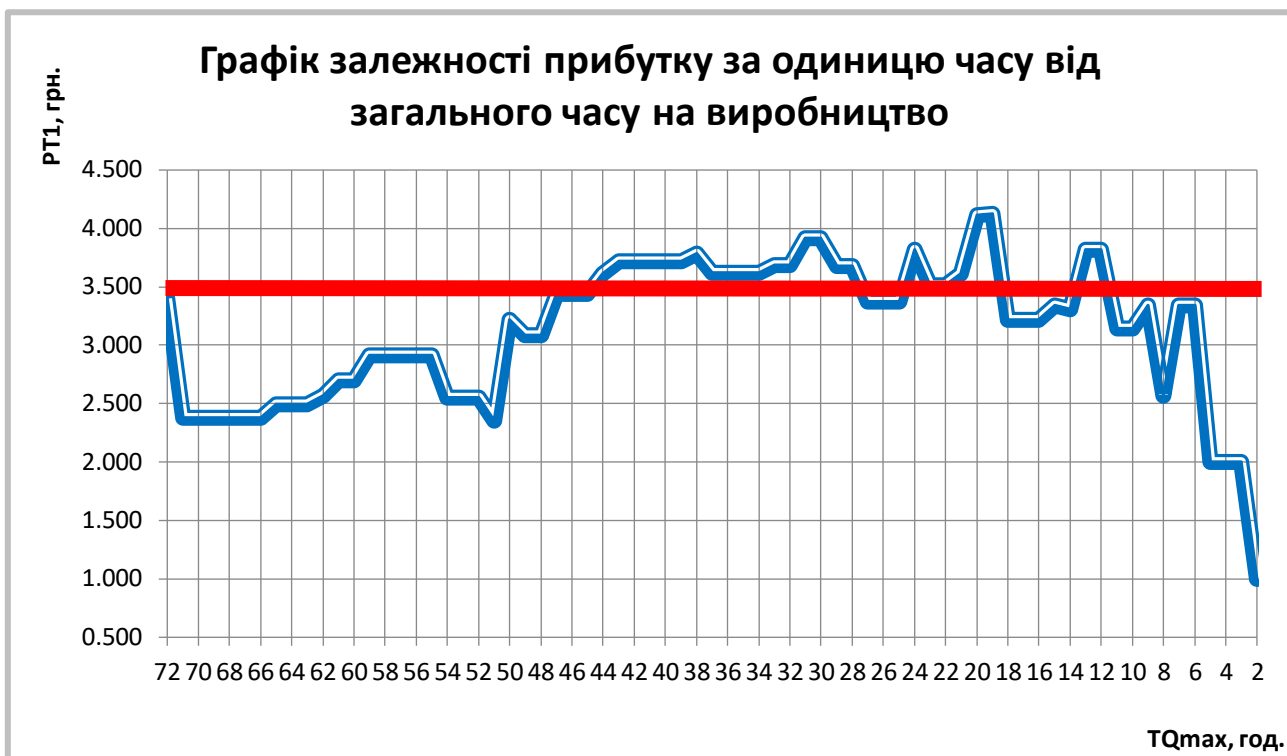


Рис. 2.1. Графік залежності прибутку від затраченого часу на виробництво першого типу

Другим типом можуть бути набори вихідних значень, у яких більшість значень цільової функції більші за початкове значення (рис. 2.2).

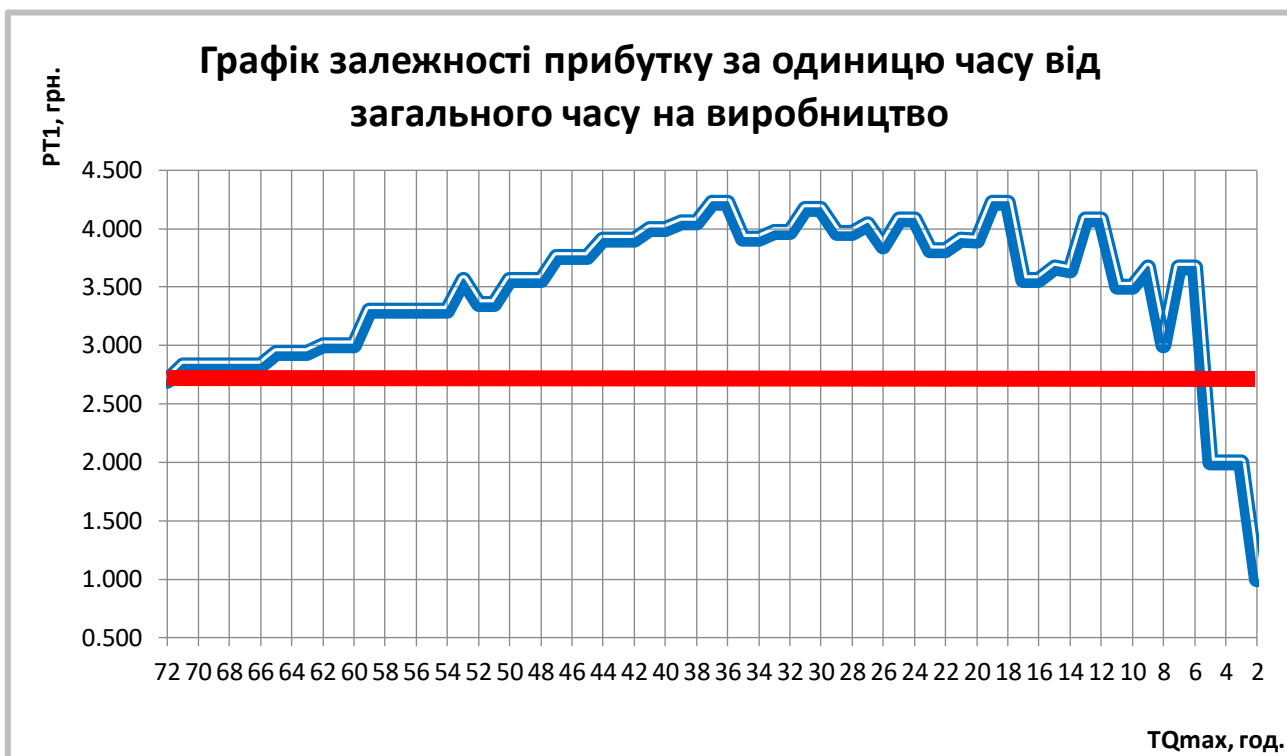


Рис. 2.2. Графік залежності прибутку від затраченого часу на виробництво другого типу

Останнім третім типом можуть бути набори вихідних значень, у яких більшість або всі значення цільової функції менші за початкове значення (рис. 2.3).

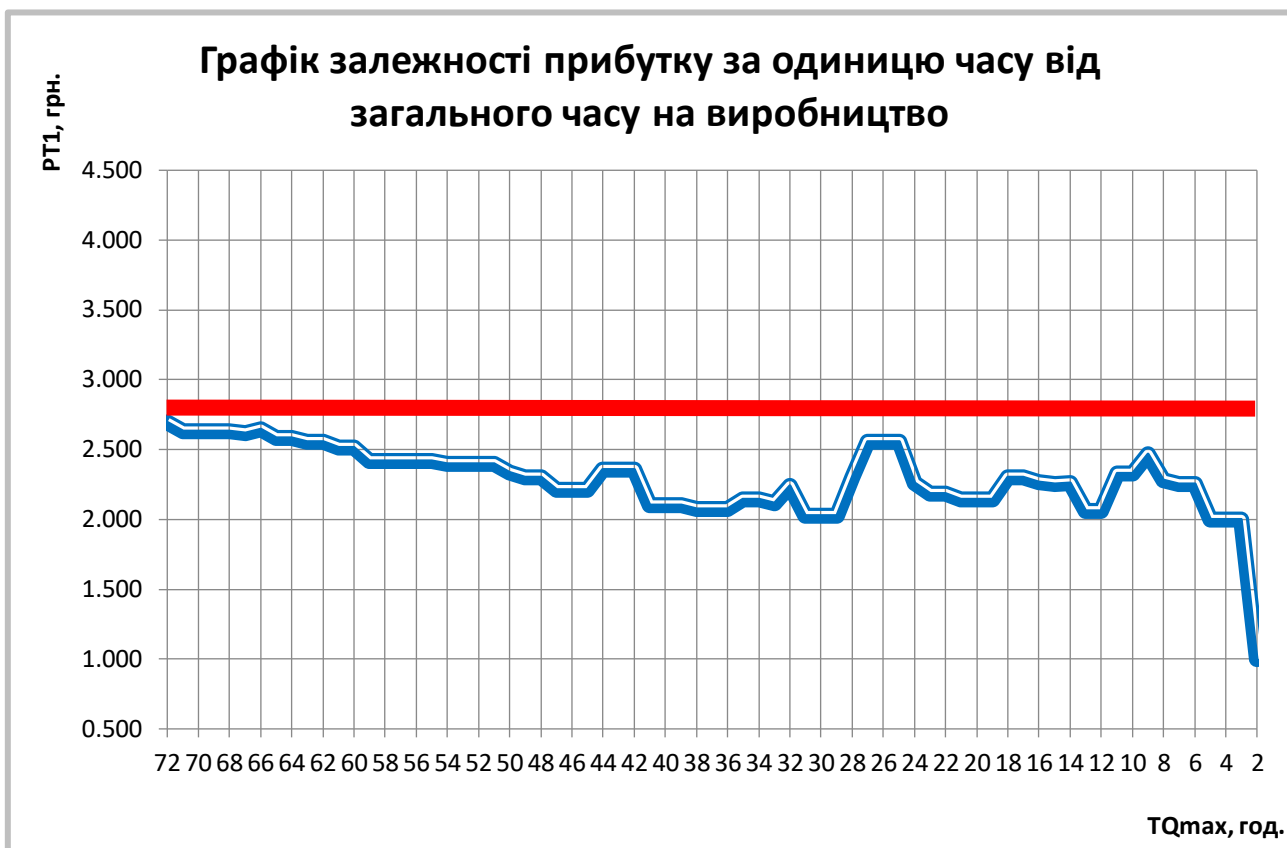


Рис. 2.3. Графік залежності прибутку від затраченого часу на виробництво третього типу

На кожній ітерації для розв'язання поточної задачі використовується симплексний метод та алгоритм Гоморі.

Симплексний метод є універсальним способом розв'язання задач лінійного програмування. У процесі його реалізації здійснюється орієнтований перебір вершин многогранника, що представляє собою множину можливих рішень задачі. Оскільки кількість вершин цього многогранника є скінченною, алгоритм симплекс-методу має властивість скінченності: за скінченну кількість кроків реалізації алгоритму ми отримаємо оптимальне рішення або ж доведемо, що цільова функція на множині планів є необмеженою. Винятком є явище зациклювання, яке полягає у можливому повторенні циклу обстеження тих самих вершин, хоча на практиці це трапляється дуже рідко.

Обстеження вершин можна розпочати лише після знаходження початкового опорного плану задачі. Тому весь алгоритм симплекс-методу ділиться на два етапи: на першому етапі знаходять початковий опорний план, а на другому – оптимальний [15].

Для отримання цілочисельних розв'язків Р. Гоморі розробив алгоритм, що базується на симплексній процедурі. Спочатку задача цілочисельного лінійного програмування (ЗЦЛП) розв'язується без врахування умови цілочисельності. Якщо отриманий розв'язок є цілим, задача вважається розв'язаною. Якщо ж розв'язок не є цілим, до обмежень задачі додається нове, яке називається правильним відтинанням [16].

Після цього нова задача лінійного програмування (ЗЛП) розв'язується з урахуванням цього додаткового обмеження. Якщо це необхідно, може бути додано ще одне відтинаюче обмеження, і так далі, поки не буде отримано повністю цілочисельний розв'язок задачі.

Загальна схема роботи методу на цьому етапі виглядає наступним чином (рис. 2.4):

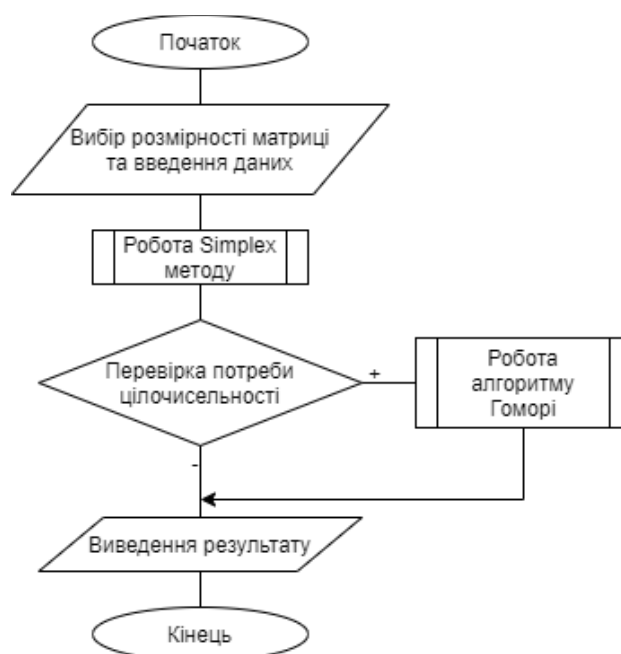


Рис. 2.4. Загальна блок-схема роботи симплексного методу та алгоритму Гоморі

Після виконання останньої з передбачених ітерацій запропонований метод зафіксує результат розв'язання задачі. Окрім максимального значення прибутку

за одиницю часу PT , результатом роботи методу будуть також план виробництва, отриманий прибуток, значення часу, затраченого на реалізацію відповідного плану виробництва та затратені для цього ресурси.

2.2. Оцінка ефективності методу розв'язання модифікованої задачі

Ефективність роботи розробленого методу проаналізуємо на довільному прикладі задачі розподілу ресурсів виробничого підприємства. Нехай підприємство виготовляє 4 види продукції (A_1, A_2, A_3, A_4). Для цього використовуються 4 види матеріалів (B_1, B_2, B_3, B_4). Відомі запаси матеріалів (b_1, b_2, b_3, b_4) та прибуток від реалізації одиниці продукції кожного виду (c_1, c_2, c_3, c_4).

Вхідні дані зобразимо у вигляді таблиці (табл. 2.1):

Таблиця 2.1

Приклад задачі розподілу ресурсів

Види матеріалів	Види продукції				Запаси матеріалів
	A_1	A_2	A_3	A_4	
B_1	1	1	0	2	45
B_2	0	1	3	2	55
B_3	2	1	1	3	87
B_4	1	2	3	0	65
Прибуток P , грн.	2	4	8	5	

У таблиці у стовпцях для кожного виду продукції зазначено витрати матеріалів для виготовлення одиниці відповідного виду продукції.

Потрібно скласти такий план виробництва продукції, який принесе максимальний прибуток P при заданих запасах.

На першому етапі роботи методу маємо знайти оптимальний план виробництва «класичної» задачі. Після послідовного відпрацювання симплекс-методу та алгоритму Гоморі ми отримаємо план виробництва, який принесе підприємству максимальний прибуток без урахування тривалості виробничого циклу (табл. 2.2):

Таблиця 2.2

План виробництва продукції «класичної» задачі

	Види продукції				Прибуток P , грн.
	A_1	A_2	A_3	A_4	
Кількість виробів Q , шт.	23	3	12	8	194

Як видно з таблиці, максимальний прибуток (P) за оптимальним планом виробництва складає 194 грн.

Модифікуємо нашу задачу, додавши до вхідних даних тривалість виробничого циклу для кожного виду продукції (табл. 2.3):

Таблиця 2.3

Вхідні дані тривалості виробничого циклу

	Види продукції			
	A_1	A_2	A_3	A_4
Тривалість виробничого циклу T_1 , год.	2	3	6	9

Наступним кроком маємо визначити тривалість періоду, необхідного для виробництва відповідної кількості кожного виду продукції (табл. 2.4)

Таблиця 2.4

Витрати часу на виробництво кожного виду продукції

	Види продукції			
	A_1	A_2	A_3	A_4
Тривалість виробничого циклу T_1 , год.	2	3	6	9
Кількість виробів Q , шт.	23	3	12	8
Час на виготовлення TQ , год.	46	9	72	72

Серед знайдених значень фіксуємо найбільше значення ($TQ_{max} = 72$ год.).

Останнім кроком аналізу результатів першої результуючої таблиці буде знаходження прибутку від виробництва за одиницю часу:

$$PT_1 = P / TQ_{max} = 194 / 72 = 2,694 \text{ грн./год.}$$

Запам'ятаємо це значення ($PT_1 = 2,694$ грн./год.), адже саме його ми будемо максимізувати на наступних етапах роботи запропонованого методу.

На початку наступного етапу роботи методу маємо модифікувати початкову таблицю даних, додавши до неї часові обмеження на виробництво кожного виду продукції (табл. 2.5):

Таблиця 2.5

Модифікована задача розподілу ресурсів

Види ресурсів	Види продукції				Запаси ресурсів
	A ₁	A ₂	A ₃	A ₄	
B ₁	1	1	0	2	45
B ₂	0	1	3	2	55
B ₃	2	1	1	3	87
B ₄	1	2	3	0	65
TQ _{A1} , год.	2	0	0	0	71
TQ _{A2} , год.	0	3	0	0	71
TQ _{A3} , год.	0	0	6	0	71
TQ _{A4} , год.	0	0	0	9	71
Прибуток P, грн.	2	4	8	5	

На цьому етапі для кожної наступної ітерації будемо брати значення TQ_{max} , зменшене на одиницю порівняно з попередньою ітерацією. Для поточної ітерації маємо:

$$TQ^*_{max} = TQ_{max} - 1 = 72 - 1 = 71 \text{ год.}$$

Для останньої ітерації TQ^*_{max} буде дорівнювати мінімальній тривалості виробничого циклу T_1 серед усіх видів продукції. З таблиці 2.3 отримуємо значення TQ^*_{max} для останньої ітерації – 2 год.

Після відпрацювання симплекс-методу та алгоритму Гоморі на цій ітерації ми отримаємо наступний план виробництва (табл. 2.6):

Таблиця 2.6

План виробництва продукції модифікованої задачі для поточної ітерації

	Види продукції				Прибуток P, грн.
	A ₁	A ₂	A ₃	A ₄	
Кількість виробів Q, шт.	20	6	11	7	187

Як і на попередній ітерації, визначаємо тривалість періоду, необхідного для виробництва відповідної кількості кожного виду продукції (табл. 2.7)

Таблиця 2.7

Витрати часу на виробництво кожного виду продукції для поточної ітерації

	Види продукції			
	A ₁	A ₂	A ₃	A ₄
Тривалість виробничого циклу T ₁ , год.	2	3	6	9
Кількість виробів Q, шт.	20	6	11	7
Час на виготовлення TQ, год.	40	18	66	63

Серед знайдених значень фіксуємо найбільше значення ($TQ_{max} = 66$ год.).

Останнім кроком аналізу результатів першої результуючої таблиці буде знаходження прибутку від виробництва за одиницю часу:

$$PT_1 = P / TQ_{max} = 187 / 66 = 2,833 \text{ грн./год.}$$

Вже на цій ітерації ми домоглися покращення показника прибутку від виробництва за одиницю часу PT_1 (2,833 грн./год. проти 2,694 грн./год. на попередній ітерації). Варто також зауважити, що не дивлячись на допустимий ліміт у часі 71 год., поточний план виробництва буде реалізований протягом 66 год. (TQ_{max}). До цього факту ми ще повернемося при подальшому аналізі.

Зменшуючи ліміт часу на виробництво (TQ^*_{max}) для кожної наступної ітерації на 1 год. та після відпрацювання на кожній ітерації симплекс-методу та алгоритму Гоморі ми будемо отримувати чергові плани виробництва (Q), значення прибутку (P), час, необхідний для виробництва відповідної кількості продукції (TQ_{max}) та прибуток за одиницю часу (PT_1).

Отримані дані зведемо до наступної таблиці (табл. 2.8):

Таблиця 2.8

Зведена таблиця результатів за всіма ітераціями

TQ^*_{max}	P	TQ_{max}	PT_1	Q_{A1}	Q_{A2}	Q_{A3}	Q_{A4}	B_1^*	B_2^*	B_3^*	B_4^*
72	194	72	2,694	23	3	12	8	42	55	85	65
71-66 *	187	66	2,833	20	6	11	7	40	53	78	65
65-63	185	63	2,937	21	7	10	7	42	51	80	65
62	180	60	3,000	13	11	10	6	36	53	65	65
61-60	180	60	3,000	25	5	10	6	42	47	83	65
59-54	178	54	3,296	26	6	9	6	44	45	85	65
53	171	48	3,563	23	9	8	5	42	43	78	65
52-51	171	51	3,353	7	17	8	5	34	51	54	65
50-48	171	48	3,563	9	16	8	5	35	50	57	65
47-45	169	45	3,756	14	15	7	5	39	46	65	65
44-42	164	42	3,905	16	14	7	4	38	43	65	65
41-40	160	40	4,000	20	13	6	4	41	39	71	64
39	158	39	4,051	19	13	6	4	40	39	69	63
38	154	38	4,053	19	12	6	4	39	38	68	61
37-36	152	36	4,222	18	12	6	4	38	38	66	60
35-34	133	34	3,912	17	11	5	3	34	32	59	54
33	131	33	3,970	16	11	5	3	33	32	57	53
32	127	32	3,969	16	10	5	3	32	31	56	51
31-30	125	30	4,167	15	10	5	3	31	31	54	50
29-28	111	28	3,964	14	9	4	3	29	27	50	44
27	109	27	4,037	13	9	4	3	28	27	48	43
26	100	26	3,846	13	8	4	2	25	24	44	41
25-24	98	24	4,083	12	8	4	2	24	24	42	40
23-22	84	22	3,818	11	7	3	2	22	20	38	34
21	82	21	3,905	10	7	3	2	21	20	36	33
20	78	20	3,900	10	6	3	2	20	19	35	31
19-18	76	18	4,222	9	6	3	2	19	19	33	30
17-16	57	16	3,563	8	5	2	1	15	13	26	24
15	55	15	3,667	7	5	2	1	14	13	24	23
14	51	14	3,643	7	4	2	1	13	12	23	21
13-12	49	12	4,083	6	4	2	1	12	12	21	20
11-10	35	10	3,500	5	3	1	1	10	8	17	14
9	33	9	3,667	4	3	1	1	9	8	15	13
8	24	8	3,000	4	2	1	0	6	5	11	11
7-6	22	6	3,667	3	2	1	0	5	5	9	10
5-4	8	4	2,000	2	1	0	0	3	1	5	4
3	6	3	2,000	1	1	0	0	2	1	3	3
2	2	2	1,000	1	0	0	0	1	0	2	1

* – ітерації, на яких був отриманий однаковий план виробництва, у таблиці об'єднані.

У таблиці зеленим кольором позначені ітерації, на яких відбувалось покращення значення результуючого показника прибутку від виробництва за одиницю часу PT_1 .

Проаналізуємо отримані результати. Для початку побудуємо графік залежності прибутку P від загального часу на реалізацію відповідного плану виробництва TQ^*_{max} (рис. 2.5).

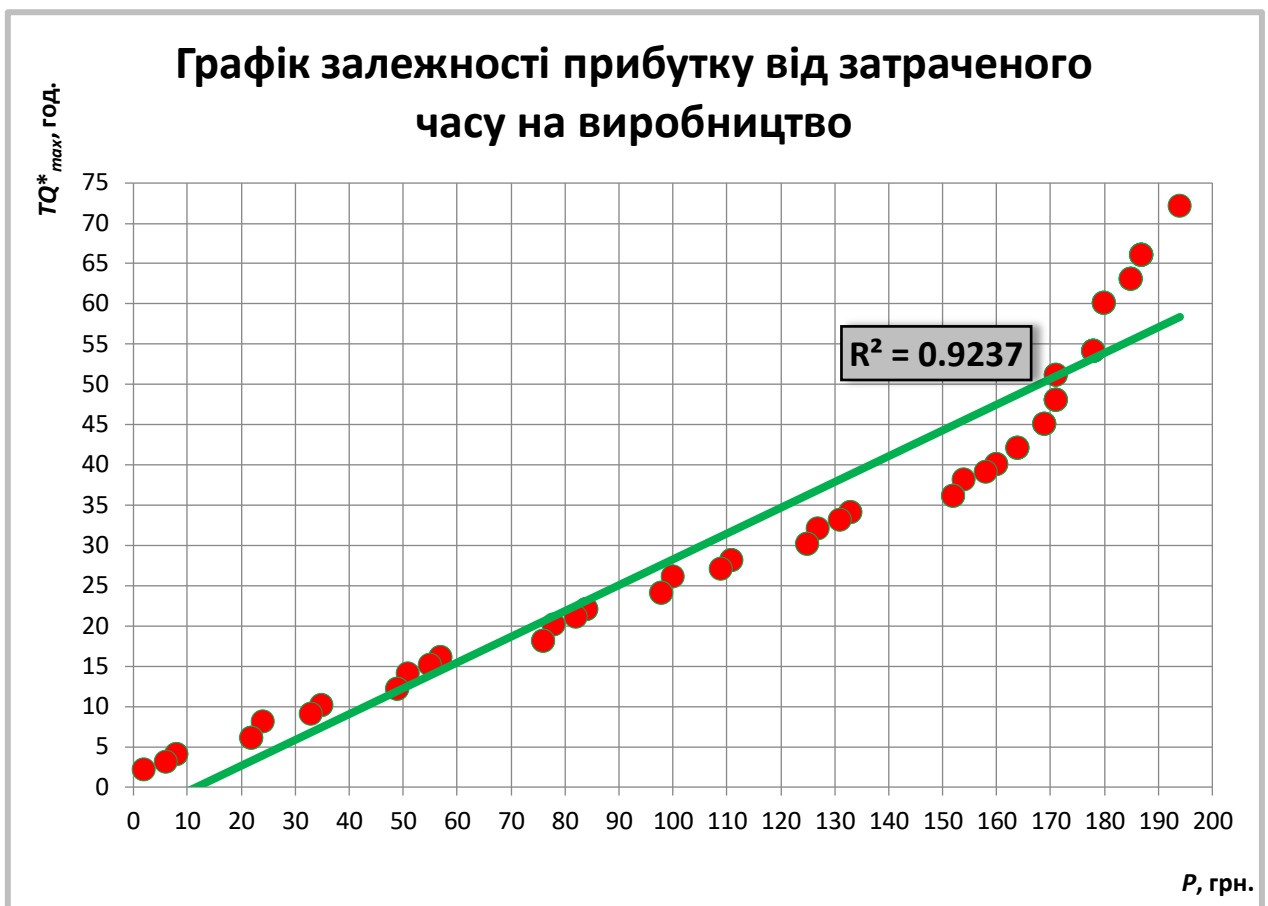


Рис. 2.5. Графік залежності прибутку від затраченого часу на виробництво

Наведений графік функції дозволяє зробити важливий висновок про те, чи модифікована нами задача розподілу ресурсів виробничого підприємства залишилась задачею лінійного програмування та чи мали ми право користуватись відповідними відомими алгоритмами розв'язання задач такого класу. Для перевірки було побудовано лінію тренду лінійного типу та розраховано значення коефіцієнта детермінації R^2 (сума квадратів відхилень фактичних значень від середнього значення).

Коефіцієнт детермінації, позначуваний як R^2 , є статистичним показником, який використовується для оцінки якості моделі регресії. Він відображає частку варіації залежної змінної, яка може бути пояснена незалежними змінними в моделі. Значення коефіцієнта детермінації коливається від 0 до 1. Якщо $R^2 = 0$, це означає, що модель не пояснює жодної варіації в даних, тобто незалежні змінні не мають жодного впливу на залежну змінну. Якщо $R^2 = 1$, це свідчить про те, що модель повністю пояснює варіацію залежної змінної, і всі дані точно

лежать на регресійній лінії. Значення, близькі до 1, вказують на те, що модель є хорошою, тоді як значення, близькі до 0, свідчать про слабку відповідність моделі даним. Коефіцієнт детермінації часто використовується в контексті лінійної регресії, але його можна застосовувати і в інших типах регресійного аналізу [17].

Отримане значення 0,9237 фактично є величиною достовірності апроксимації. Модель вважається неадекватною, якщо $R^2 < 0,9$ [18]. Отже, маємо підтвердження факту, що модифікована задача, як і «класична», залишилась задачею лінійного програмування (ЗЛП).

Для наведеного прикладу на етапі аналізу результатів нами були перевірені результати для всіх можливих значень TQ^*_{max} , а не лише у діапазоні від 72 год. до 2 год. Для вхідних даних прикладу найбільшим значенням TQ^*_{max} є значення 192 год. (виріб 4 має найбільшу тривалість виробничого циклу – 9 год., наявні запаси дозволяють виготовити 22 одиниці цього виробу). Наведемо та проаналізуємо розширений варіант графіку залежності прибутку за одиницю часу від загального часу на виробництво TQ^*_{max} (рис. 2.6).

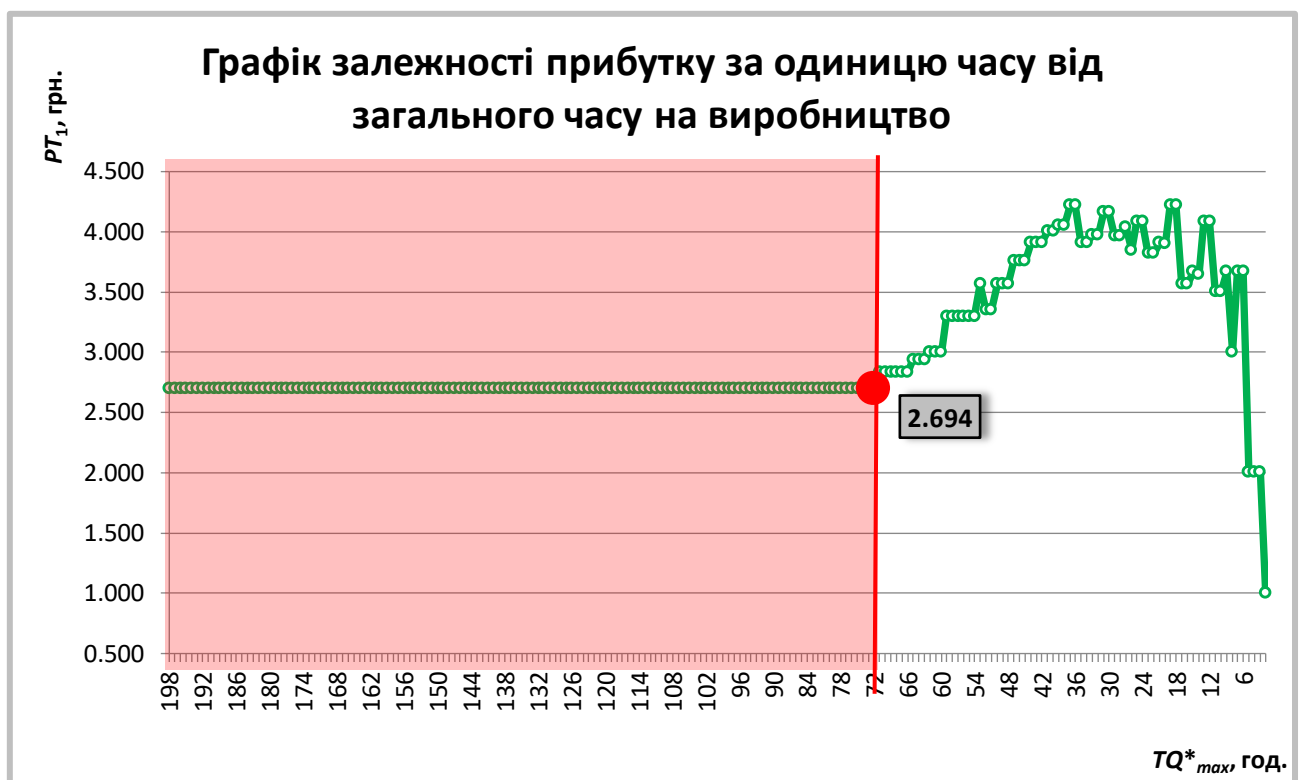


Рис. 2.6. Графік залежності прибутку за одиницю часу від загального часу на виробництво

Червона область на графіку яскраво демонструє той факт, що для всіх ітерацій із значеннями загального часу на виробництво TQ^*_{max} , які більше за визначене на першій ітерації, на виході буде запропонований однаковий план виробництва з тим самим значенням $TQ_{max} = 72$ год., що й для «класичної» задачі. Збільшення ліміту у часі не призводить до збільшення прибутку, адже симплекс-метод та алгоритм Гоморі ще на першій ітерації вже «знайшли» оптимальний план виробництва, який забезпечує максимальний прибуток. А збільшення ліміту у часі могло лише зменшити величину прибутку за одиницю часу.

Після отримання цього висновку надалі розглядалися значення загального часу на виробництво лише у сторону зменшення.

З урахуванням вищезазначеного попередній графік для поточного прикладу набуває наступного вигляду (рис. 2.7):

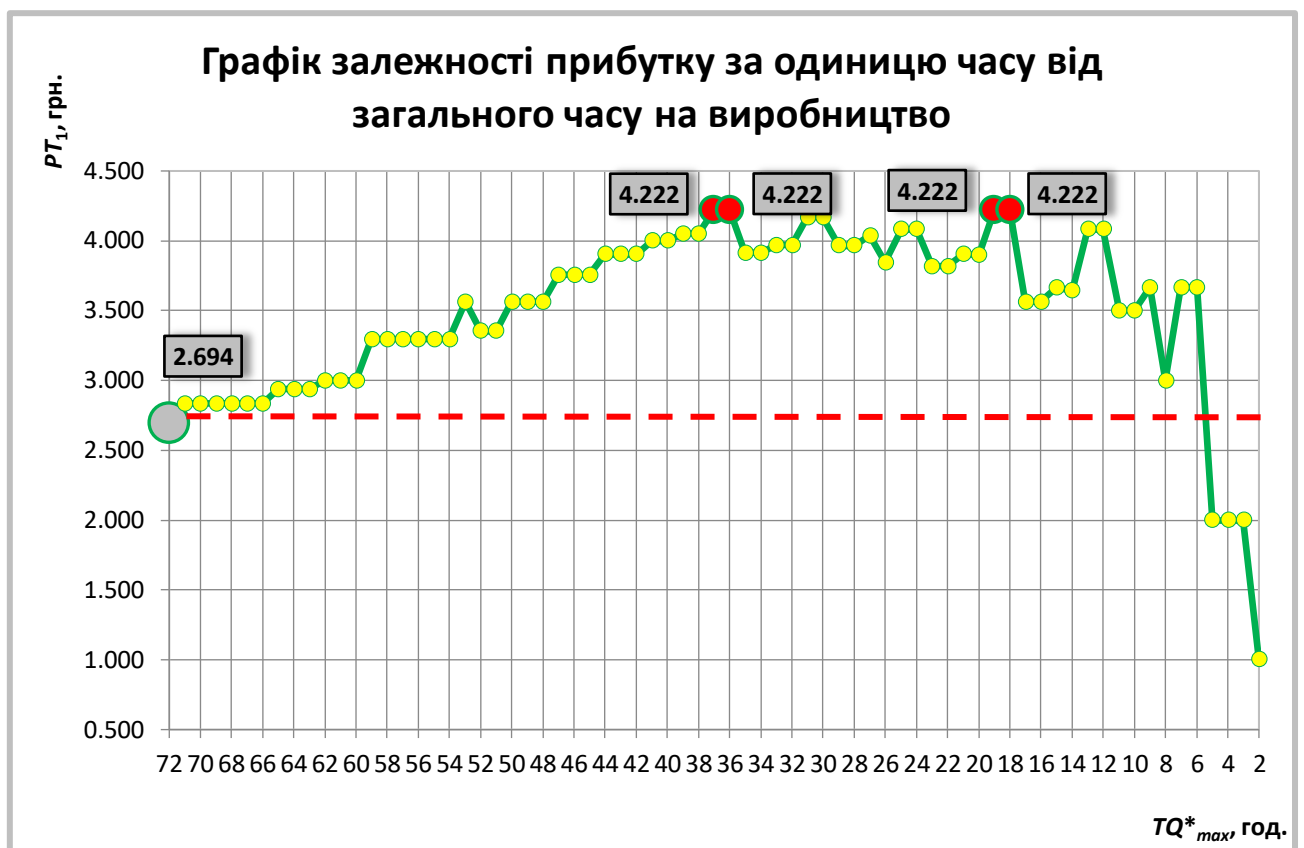


Рис. 2.7. Графік залежності прибутку за одиницю часу від загального часу на виробництво для зменшеного діапазону по часу

Як видно з вищенаведеного графіку та з даних таблиці 2.8, отримане максимальне значення прибутку за одиницю часу

$$PT_1 = 4,222 \text{ грн./год.}$$

Слід зауважити, що дане значення було отримано на декількох ітераціях (при лімітах часу $TQ^*_{max} = 37$ год., $TQ^*_{max} = 36$ год., $TQ^*_{max} = 19$ год. та $TQ^*_{max} = 18$ год.). Відповідно до розробленого методу, для такої ситуації у якості відповіді буде взято план виробництва з найменшим значенням TQ^*_{max} .

Таким чином, результатом роботи розробленого методу та відповіддю поточної задачі буде наступний набір даних:

– план виробництва:

– виріб A_1 : $Q_{A1} = 9$ шт.;

– виріб A_2 : $Q_{A2} = 6$ шт.;

– виріб A_3 : $Q_{A3} = 3$ шт.;

– виріб A_4 : $Q_{A4} = 2$ шт.;

– затрачені ресурси:

– ресурс B_1 : $B_1^* = 19$;

– ресурс B_2 : $B_2^* = 19$;

– ресурс B_3 : $B_3^* = 33$;

– ресурс B_4 : $B_4^* = 30$;

– загальний прибуток $P = 76$ грн.

– час, необхідний для виконання плану виробництва $TQ_{max} = 18$ год.

– прибуток за одинцю часу $PT_1 = 4,222$ грн.

Врахування у моделі такого показника як тривалість виробничого циклу та завдяки розробленому методу нам вдалось збільшити рівень прибутку за одиницю часу у середньо- та довгостроковій перспективах на 56,7% (з 2,694 грн./год. до 4,222 грн./год.).

Слід додати, що обрані для даного прикладу одиниці виміру є формальними та використані лише для спрощення сприйняття та розуміння роботи методу на всіх його етапах й отриманих результатів.

Наведений приклад підтверджує доцільність додавання до моделі показника тривалості виробничого циклу та ефективність розробленого методу розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства.

Отримані результати надають точну інформацію для прийняття відповідних управлінських рішень, зокрема, щодо змін у виробництві та управління запасами задля збільшення прибутку підприємства у середньо- та довгостроковій перспективах.

2.3. Розробка методу аналізу наявних ресурсів підприємства

У рамках проведеного дослідження, окрім основного методу розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства, було розроблено ще метод для аналізу наявних ресурсів підприємства. Цей метод працює незалежно від попереднього та аналізує дані «класичної» задачі.

Мета даного методу полягає у виявленні того або тих ресурсів, запаси яких «стримують» виробництво або ж знаходяться у незбалансованому стані щодо кількості порівняно з іншими.

Відповідно до розробленого методу пропонується по чергово «штучно» збільшувати запаси кожного з наявних ресурсів у 10 разів. При цьому запаси всіх інших ресурсів залишаються незмінними. Після кожного такого збільшення потрібно розв'язати отриману «нову» задачу симплексним методом та алгоритмом Гоморі. Окрім отриманого плану виробництва, нас будуть цікавити витрати ресурсів для виконання плану. Ці значення й будуть основною для подальшого аналізу та прийняття відповідних управлінських рішень відповідними службами підприємства.

Розглянемо роботу даного методу на прикладі з попереднього параграфу.

Вхідні дані наведемо у табличному вигляді (табл. 2.9):

На першому етапі роботи методу маємо знайти оптимальний план виробництва даної задачі. Після послідовного відпрацювання симплекс-методу та алгоритму Гоморі ми отримаємо план виробництва, який принесе підприємству максимальний прибуток без урахування тривалості виробничого циклу (табл. 2.10):

Таблиця 2.9

Приклад задачі розподілу ресурсів

Види матеріалів	Види продукції				Запаси матеріалів
	A_1	A_2	A_3	A_4	
B_1	1	1	0	2	45
B_2	0	1	3	2	55
B_3	2	1	1	3	87
B_4	1	2	3	0	65
Прибуток P , грн.	2	4	8	5	

Таблиця 2.10

План виробництва продукції «класичної» задачі

	Види продукції				Прибуток P , грн.
	A_1	A_2	A_3	A_4	
Кількість виробів Q , шт.	23	3	12	8	194

Відповідно до розробленого методу для аналізу запасів ресурсів, маємо почергово збільшувати кожен з видів ресурсів у 10 разів, залишаючи при цьому значення решти видів ресурсів без змін. Після кожної зміни симплекс-метод та алгоритм Гоморі будуть надавати змінені плани виробництва та загальне значення прибутку.

Для наведеного прикладу результуюча таблиця для аналізу набуде наступного вигляду (табл. 2.11):

Таблиця 2.11

Результуюча таблиця для аналізу запасів ресурсів

	B_1	B_2	B_3	B_4	P	Q_{A1}	Q_{A2}	Q_{A3}	Q_{A4}	B_1^*	B_2^*	B_3^*	B_4^*
	45	55	87	65	194	23	3	12	8	42	55	85	65
B_1	450	55	87	65	197	10	26	1	13	62	55	86	65
B_2	45	550	87	65	278	0	0	21	22	44	107	87	63
B_3	45	55	870	65	197	26	1	12	9	45	55	92	64
B_4	45	55	87	650	226	29	16	13	0	45	55	87	100

Отримані результати дозволяють зробити наступні висновки:

– для матеріалу B_1 навіть десятикратне збільшення запасів не призведе до суттєвого збільшення прибутку P . Із заявлених 450 од. цього матеріалу буде використано 62 од. (збільшено на 37,7% порівняно з початковим значенням у 45 од.). При цьому прибуток буде збільшено лише на 1,5% (зі 194 грн. до 197 грн.), що є несуттєвим;

– для матеріалу B_2 десятикратне збільшення запасів призводить до суттєвого збільшення прибутку P . Із заявлених 550 од. цього матеріалу буде використано 107 од. (збільшено майже у 2 рази порівняно з початковим значенням у 55 од.). При цьому прибуток буде збільшено на 43,3% (зі 194 грн. до 278 грн.), що є суттєвим;

– для матеріалу B_3 навіть десятикратне збільшення запасів не призведе до суттєвого збільшення прибутку P . Із заявлених 870 од. цього матеріалу буде використано 92 од. (збільшено на 5,7% порівняно з початковим значенням у 87 од.). Прибуток при цьому буде збільшено лише на 1,5% (зі 194 грн. до 197 грн.), що є несуттєвим;

– для матеріалу B_4 десятикратне збільшення запасів призводить до суттєвого збільшення прибутку P . Із заявлених 650 од. цього матеріалу буде використано 100 од. (збільшено у 1,5 рази порівняно з початковим значенням у 65 од.). При цьому прибуток буде збільшено на 16,5% (зі 194 грн. до 226 грн.).

Загальний висновок після проведеного аналізу можна сформулювати наступним чином:

1. Однозначно потребує коригування у бік збільшення кількість ресурсу B_2 . Саме нестача цього ресурсу не дозволяє доволі суттєво збільшити прибуток. Потрібні відповідні управлінські рішення на середньо- та довгострокову перспективи щодо збалансування кількості ресурсу B_2 порівняно з іншими ресурсами.

2. Ресурси B_1 та B_3 мають доволі збалансовану кількість. Коригування не потрібні.

3. Щодо ресурсу B_4 , то коригування кількості допустиме, але при цьому не є пріоритетним. Для цього ресурсу потрібні окремі управлінські рішення з боку відповідних служб підприємства.

Висновки до другого розділу

У даному розділі розроблені та описані математична модель та алгоритм розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства.

Ефективність роботи розробленого методу проаналізовано на реальному прикладі задачі розподілу ресурсів виробничого підприємства. Завдяки розробленому методу вдалося збільшити показник прибутку за одиницю часу на 56,7% (з 2,694 грн./год. до 4,222 грн./год.). Опрацьований приклад підтвердив доцільність додавання до моделі показника тривалості виробничого циклу та ефективність розробленого методу розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства.

У даному розділі, окрім основного методу розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства, було розроблено та описано й метод для аналізу наявних ресурсів підприємства. Роботу даного методу було розглянуто на реальному прикладі. Результати роботи методу надали можливість провести детальний аналіз наявних запасів виробничих ресурсів підприємства.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ

3.1. Визначення архітектури та засобів реалізації прототипу вебдодатку

Додаток буде побудований згідно архітектури RESTful API [28]. Оскільки RESTful API базується на стандартному протоколі HTTP та використовує формати передачі даних, такі як JSON [29] або XML [30], розроблений сервер буде сумісний з будь-якою платформою або мовою програмування.

У цій архітектурі додатку було обрано Symfony [31] як технологію серверної сторони, що реалізує RESTful API, бібліотеку Doctrine [32], що використовується для зручності роботи з СКБД MySQL [33] та клієнтську сторону, реалізовану за допомогою React [34]. Для зручності створення та керування RESTful API буде використано ApiPlatform [35]. Для реалізації алгоритмів симплекс-метода та метода Гоморі буде використано Python [36] та модуль Numpy [37]. Все це буде розгорнуто на середовищі Docker [38] з використанням Nginx [39] у якості вебсервера.

Основний принцип роботи цієї архітектури полягає у розділенні функціональності між серверною та клієнтською сторонами додатку. Серверна сторона, реалізована за допомогою Symfony, надає RESTful API для взаємодії з клієнтом.

Symfony дозволяє визначати маршрути, які відповідають різним запитам від клієнта, і виконує відповідні дії. Запити включають HTTP методи, такі як GET, POST, PUT і DELETE, які відповідають операціям з ресурсами. Symfony обробляє ці запити, взаємодіє з базою даних за допомогою Doctrine та повертає відповідь у вигляді JSON.

Клієнтська сторона, реалізована за допомогою React, відображає інтерфейс користувача і взаємодіє з сервером через API. Клієнтська програма React відправляє HTTP запити до сервера, за допомогою бібліотеки Axios [40] для асинхронних запитів, отримує дані та відмальовує актуальний інтерфейс додатку. На рисунку 3.1 зображено схему роботи додатку:

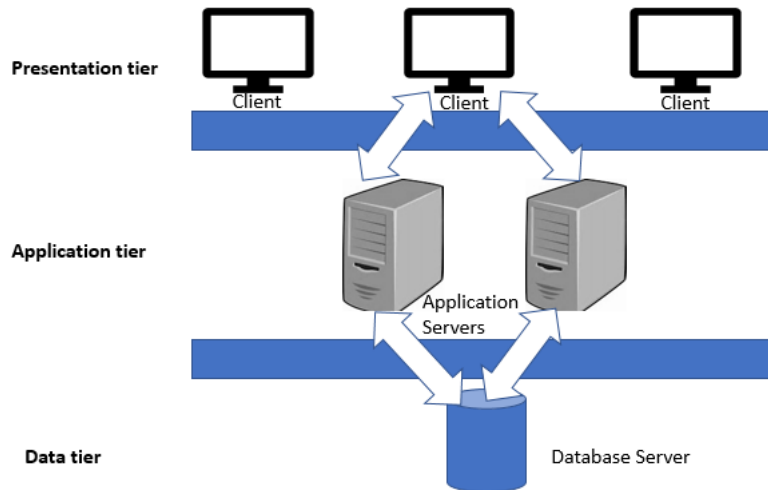


Рис. 3.1. Схема роботи клієнт-сервер додатку

Усе це розгорнуто за допомогою Docker, що дозволяє створити контейнеризовані середовища для серверної і клієнтської частин додатку. Контейнеризація дозволяє легко розгортати, масштабувати і керувати додатком.

Nginx використовується як веб-сервер, який служить проксі-сервером для перенаправлення запитів клієнта до відповідного контейнера з Symfony або React. Також він забезпечує обробку статичних файлів та управління HTTP запитами.

Під час розробки програмного продукту будуть задіяні наступні патерни:

- Dependency Injection, який дозволяє зручно керувати залежностями між класами, а за допомогою DI контейнера Symfony можна легко визначати, впроваджувати та управляти цими залежностями в додатку, що забезпечує більшу модульність та можливість перевикористання коду;

- Repository, який дозволяє відокремити логіку доступу до даних від моделей даних, що сприяє зручному виконанню запитів до бази даних та отриманню об'єктів моделей, що призводить до покращення модульності коду та полегшує процес його тестування;

- Active Record, який дозволяє забезпечити зв'язок між об'єктами моделей та записами в базі даних. Цей підхід спрощує взаємодію з базою даних, надаючи зручні методи для збереження, оновлення та видалення даних;

– Event-driven Architecture, який дозволяє під час розробки створювати та обробляти події, що виникають у додатку. Цей підхід дозволяє виконувати певну логіку при виникненні конкретних подій і реагувати на них відповідним способом;

– Object-Relational Mapping (ORM), який спрощує взаємодію з базою даних, дозволяючи працювати з даними у зручному об'єктно-орієнтованому стилі. Замість прямої роботи з SQL-запитами можна використовувати звичні методи та запити, що значно полегшує роботу з базою даних.

Далі розглянемо обрані інструментальні засоби.

Symfony – це потужний фреймворк для розробки вебдодатків на PHP. Він пропонує широкий спектр функцій та інструментів, які спрощують процес розробки, забезпечують структуру та організацію коду. У порівнянні з Laravel, Symfony має більш гнучку архітектуру, що дає багато можливостей для налаштування та розширення. Крім того, він славиться своєю високою продуктивністю та надійністю. До переваг використання Symfony слід віднести можливість організації додатку відповідно до найкращих практик та стандартів розробки, забезпечуючи стандартизовану структуру коду, гнучкість розширення шляхом додавання бібліотек та компонентів, що спрощує розширення функціональності додатку, великий набір готових компонентів, які надають різноманітну функціональність, включаючи обробку HTTP-запитів, роботу з базами даних, кешування, форми, безпеку та багато іншого та активну спільноту розробників, яка надає підтримку, документацію та навчальні ресурси, сприяючи вашому процесу розробки.

React – це популярна JavaScript бібліотека для створення інтерфейсів користувача, яка надає можливість розробляти ефективні, масштабовані та зручні використовувати користувацькі інтерфейси. Порівняно з іншими бібліотеками, такими як Vue або Angular, React має перевагу у швидкості відображення змін на сторінці, завдяки використанню віртуального DOM і механізму React Hooks. Основними перевагами використання React є принцип компонентної архітектури, де інтерфейс розбивається на невеликі, повторно

використовувані компоненти, що сприяє легкості організації коду, забезпечує модульність та зручне керування станом додатку, використання технології віртуального DOM, що дозволяє ефективно оновлювати лише необхідні частини інтерфейсу, що забезпечує високу продуктивність та швидкодію додатку, можливість відстежувати зміни стану та автоматично оновлювати відповідні компоненти, що спрощує керування станом та реагування на події.

Python – це універсальна мова програмування, яка відома своєю простотою, читабельністю та великою кількістю бібліотек для різних завдань. Одним із таких потужних інструментів є модуль NumPy – бібліотека для роботи з багатовимірними масивами та виконання числових обчислень. NumPy широко використовується в науці про дані, машинному навчанні та розв'язанні математичних задач, включаючи задачі лінійного програмування. Основні переваги використання Python і NumPy – це простий синтаксис та легка підтримка коду, навіть для складних математичних моделей, наявність в NumPy швидких операцій над масивами, підтримка лінійної алгебри, числової оптимізації та інших математичних обчислень, через що вона працює набагато швидше, ніж стандартні Python-цикли, завдяки оптимізації на низькому рівні, ефективне вирішення задач оптимізації, зокрема знаходження розв'язків систем лінійних нерівностей, максимальне використання ресурсів або мінімізація витрат.

Docker – це платформа для контейнеризації додатків, яка дозволяє упаковувати додатки та їх залежності в контейнери, що забезпечує переносимість та прискорює процес розгортання та налаштування. У порівнянні з іншими аналогами, такими як Podman та LXD, Docker має вагому перевагу у використанні ресурсів. Docker використовує оптимізовану віртуалізацію на рівні контейнерів, що дозволяє ефективно використовувати системні ресурси, такі як CPU, пам'ять та дисковий простір. Це дозволяє зменшити витрати на інфраструктуру та підвищити масштабованість додатків, забезпечуючи ефективне використання обчислювальних ресурсів. Основними перевагами використання Docker є переносимість і незалежність від конкретної системи

контейнери, що дозволяють легко розгортати додатки для різних середовищ, таких як розробка, тестування і продакшн, ізолювання додатків та їх залежності, що забезпечує стабільність і безпеку, власне середовище для контейнерів, що запобігає конфліктам та впливу одного додатку на інший, швидке розгортання та масштабування додатків за допомогою контейнерів, інструменти для автоматизації процесу розгортання та оркестрації контейнерів.

Ця архітектура дозволяє розділити серверну і клієнтську частини додатку, над якими розробники можуть працювати незалежно. Крім того, використання Docker допомагає забезпечити консистентність середовища розробки і виробничого середовища, а також спрощує процес розгортання та масштабування додатку.

Загалом, використання Symfony, React і Docker дозволяє створювати ефективні, масштабовані та зручні у розробці та розгортанні додатки. Ці технології забезпечують швидку відповідь користувачеві та зручний інтерфейс, що спрощує взаємодію з додатком.

3.2. Реалізація окремих модулів прототипу вебдодатку

Робота серверної частини, реалізованої за допомогою Symfony, полягає в правильній обробці і маніпуляції даних, необхідних для роботи симплекс-метода та алгоритму Гоморі. Для цього було створено контролер, який приймає і конвертує в необхідний формат дані з матриці витрат «А», запасів ресурсів «В», цільової функції (прибутку) «с» і значення тривалостей виробничого циклу:

```
$content = json_decode($request->getContent(), true);

$A = [];
$b = $content['materialQuantities'];
$c = $content['profits'];

$productionCycleDurations = $content['productionCycleDurations'];

foreach ($content['materialRequirements'] as $productIndex => $productRequirements) {
    foreach ($productRequirements as $materialIndex => $quantity) {
        $A[$materialIndex][$productIndex] = $quantity;
    }
}
```

```

        unset($productIndex, $productRequirements, $materialIndex, $quantity);
    }

```

Після приведення даних до необхідного вигляду, потрібно передати їх у вигляді JSON в Python-скрипт, що реалізує симплекс-метод і алгоритм Гоморі.:

```

public function executeScript($A, $b, $c, $maxProductionTime = null,
$minProductionCycleDuration = null): mixed
{
    $A_json = escapeshellarg(json_encode($A));
    $b_json = escapeshellarg(json_encode($b));
    $c_json = escapeshellarg(json_encode($c));

    $command = "python3 " . $this->scriptsDir . "SimplexGomory.py $A_json $b_json
$c_json";

    if ($maxProductionTime && $minProductionCycleDuration) {
        $maxProductionTime_json = escapeshellarg(json_encode($maxProductionTime));
        $minProductionCycleDuration_json =
escapeshellarg(json_encode($minProductionCycleDuration));

        $command .= " $maxProductionTime_json $minProductionCycleDuration_json";
    }

    $output = shell_exec($command);

    return json_decode($output, true);
}

```

Для знаходження початкового (дробового) рішення використовувався двофазний симплекс-метод, лістинг коду якого наведений у додатку А. Для знаходження цілочисельного рішення використовувався алгоритм Гоморі і двоїстий симплекс-метод, лістинги коду яких наведені у додатках Б і В.

Після отримання початкового результату, відповідно до розробленого методу для аналізу запасів ресурсів, по чергово збільшимо кожен з видів ресурсів у 10 разів, залишаючи при цьому значення решти видів ресурсів без змін і виконаємо симплекс-метод і алгоритм Гоморі:

```

public function getMultipliedResourcesResult(
    array $initialResult,
    array $A,
    array $b,
    array $c
): array
{
    $multipliedResults = [$initialResult];

    foreach ($b as &$value) {
        $value *= 10;
    }
}

```

```

        $multipliedResults[] = $this->executeScript($A, $b, $c);

        $value /= 10;
    }

    return $multipliedResults;
}

```

Наступний код відповідає за знаходження найбільшої і найменшої кількості часу для виготовлення для подальшої роботи алгоритму:

```

        $productionTimeValues = [];

        foreach ($productionCycleDurations as $key => $cycleDuration) {
            $productionTimeValues[$key] = intval($cycleDuration *
$initialResult['integerSolution'][$key]);
        }

        $maxProductionTime = max($productionTimeValues);
        $minProductionCycleDuration = min($productionCycleDurations);

```

Після цього модифікується матриця «A», заносючи до неї значення тривалостей виробничого циклу:

```

public function modifyMatrix($A, $productionCycleDurations): array
{
    $A_modified = $A;

    foreach ($A as $rowIndex => $row) {
        $newRow = [];
        foreach ($productionCycleDurations as $productionCycleDurationIndex =>
$productionCycleDuration) {
            if ($rowIndex === $productionCycleDurationIndex) {
                $newRow[] = $productionCycleDuration;

                continue;
            }

            $newRow[] = 0;
        }

        $A_modified[] = $newRow;
    }

    return $A_modified;
}

```

Після відпрацювання симплекс-методу та алгоритму Гоморі, до результату заносяться додаткові дані, такі як прибуток за одиницю часу та кількість витрачених ресурсів:

```

public function getSolutionValues(
    array $results,
    array $productionCycleDurations,
    array $A
): array
{
    foreach ($results as &$result) {
        $resultProductionTimeValues = [];

        foreach ($productionCycleDurations as $key => $cycleDuration) {
            $resultProductionTimeValues[$key] = intval($cycleDuration *
$result['integerSolution'][$key]);
        }

        $result["maxProductionTime"] = max($resultProductionTimeValues);
        $result["profitPerTimeUnit"] = $result["optimalValue"] /
$result["maxProductionTime"];

        foreach ($A as $ARow) {
            $resourcesSpent = 0;

            foreach ($ARow as $key => $value) {
                $resourcesSpent += $value * $result["integerSolution"][$key];
            }

            $result["resourcesSpent"][] = $resourcesSpent;
        }
    }

    return $results;
}

```

Далі визначаємо покращення серед результатів роботи алгоритму і найкращий з отриманих розрахунків:

```

$highestProfitPerTimeUnit = PHP_FLOAT_MIN;
$lowestMaxProductionTime = PHP_INT_MAX;

$bestResult = reset($results);

foreach ($results as &$result) {
    if ($result["profitPerTimeUnit"] < $highestProfitPerTimeUnit) {
        continue;
    }

    if ($result["profitPerTimeUnit"] > $highestProfitPerTimeUnit) {
        $highestProfitPerTimeUnit = $result["profitPerTimeUnit"];

        $result["isBetterProfitPerTimeUnit"] = true;
        $bestResult = $result;

        continue;
    }

    if ($result["profitPerTimeUnit"] === $highestProfitPerTimeUnit &&
        $result["maxProductionTime"] < $lowestMaxProductionTime) {

```

```

        $lowestMaxProductionTime = $result["maxProductionTime"];

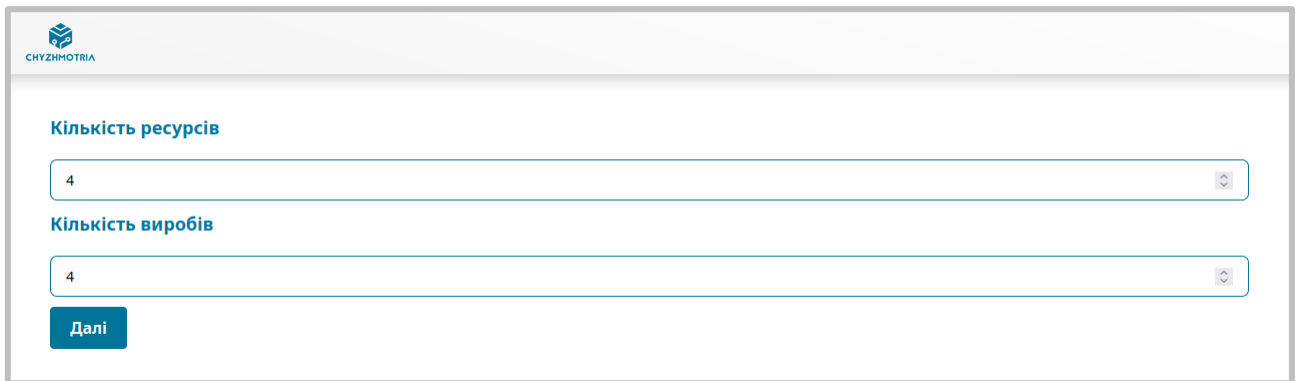
        $result["isBetterMaxProductionTime"] = true;
        $bestResult = $result;
    }
}

```

3.3. Опис інтерфейсу та порядок роботи із вебдодатком

Розроблений прототип вебдодатку наочно демонструє роботу розроблених у роботі методів. Додаток має інтуїтивно зрозумілий інтерфейс. Він дозволяє поетапно вводити початкові дані задачі. Після їх обробки та застосування відповідних розроблених алгоритмів як результат роботи додаток надає детальну інформацію по кожному з описаних у роботі методів, включно із оптимальним планом виробництва. Вихідна інформація подається у текстовому, табличному та графічному вигляді.

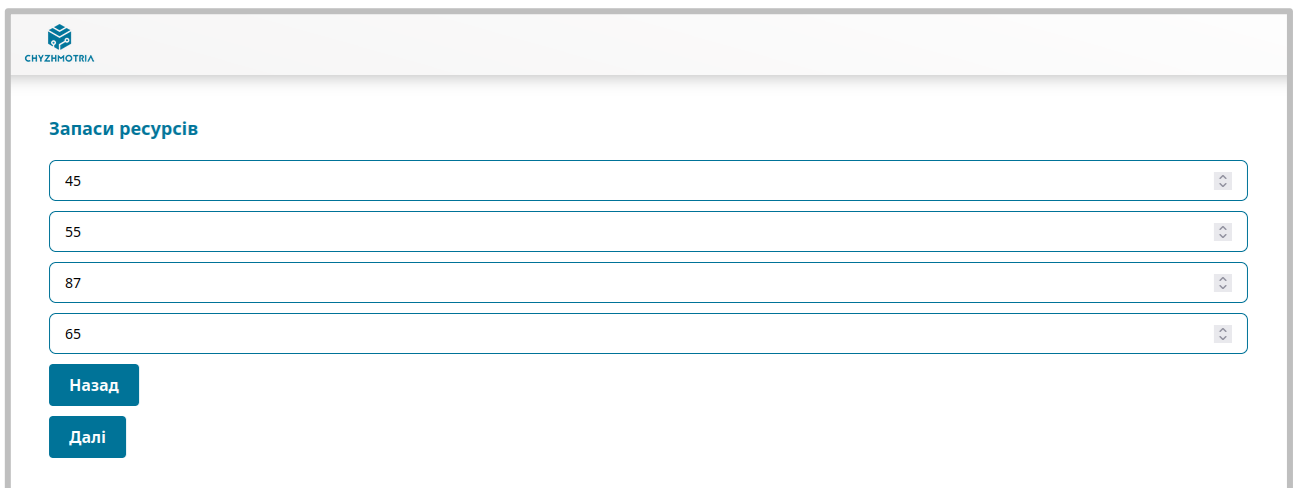
На початку роботу додаток пропонує користувачеві ввести розмірність задачі, а саме кількість видів виробів, що виробляє підприємство, та кількість видів ресурсів, які використовуються у виробничому процесі. Для прототипу вебдодатку обрана максимально допустима розмірність задачі 10×10 (рис. 3.2).



The screenshot shows a web application interface with a header containing a logo and the text 'CHYZHMOTRIA'. Below the header, there are two input fields. The first field is labeled 'Кількість ресурсів' (Number of resources) and contains the value '4'. The second field is labeled 'Кількість виробів' (Number of products) and also contains the value '4'. At the bottom of the form, there is a blue button labeled 'Далі' (Next).

Рис. 3.2. Форма введення розмірності задачі

На другому кроці користувачу пропонується ввести значення запасів по кожному виду ресурсів (рис. 3.3).



Запаси ресурсів

45

55

87

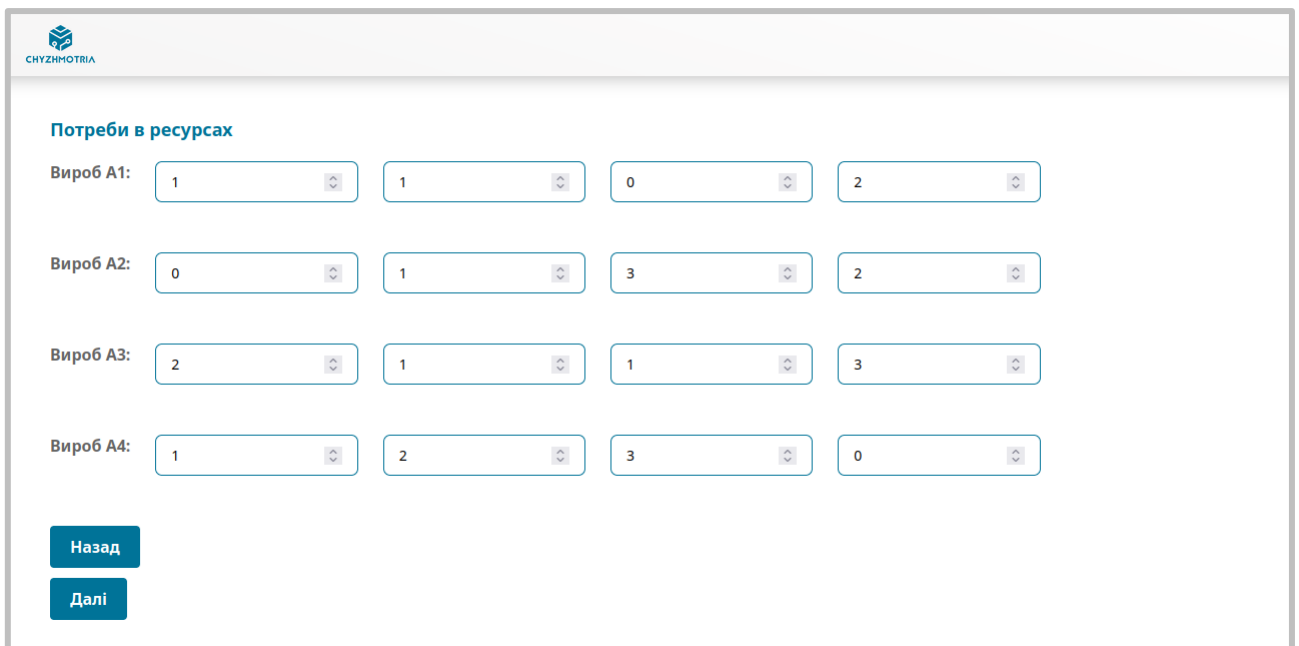
65

Назад

Далі

Рис. 3.3. Форма введення запасів ресурсів

На наступному кроці зазначаються витрати кожного виду ресурсу для виготовлення одиниці кожного з видів продукції (рис. 3.4).



Потреби в ресурсах

Вироб А1: 1 1 0 2

Вироб А2: 0 1 3 2

Вироб А3: 2 1 1 3

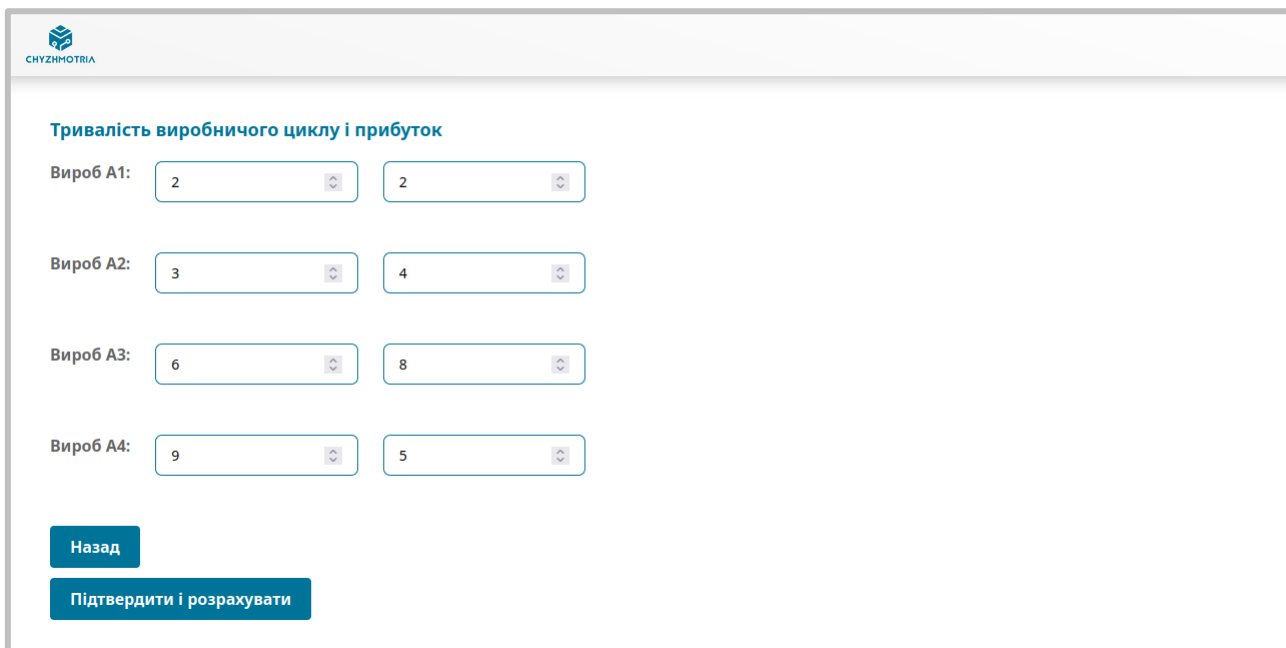
Вироб А4: 1 2 3 0

Назад

Далі

Рис. 3.4. Форма введення витрат ресурсів на одиницю продукції

На останньому кроці введення початкових даних для кожного виду виробу необхідно зазначити тривалість виробничого циклу та прибуток від реалізації одиниці цього виробу (рис. 3.5).



Тривалість виробничого циклу і прибутку

Вироб А1: 2 2

Вироб А2: 3 4

Вироб А3: 6 8

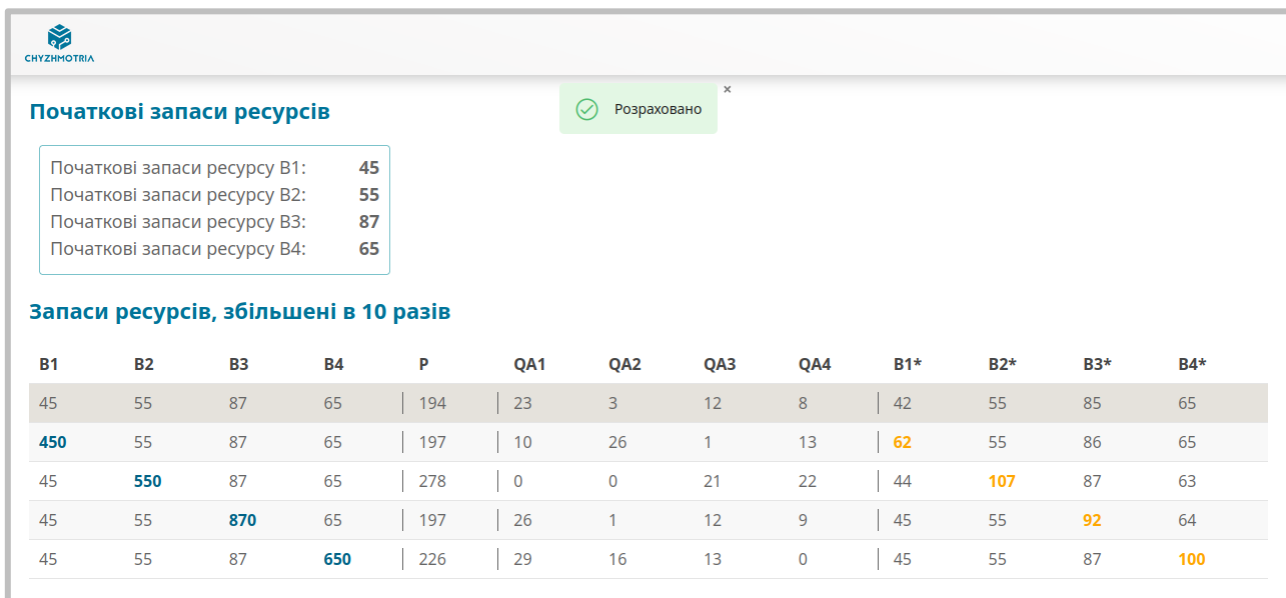
Вироб А4: 9 5

Назад

Підтвердити і розрахувати

Рис. 3.5. Форма введення значень тривалості виробничого циклу та прибутку від реалізації одиниці продукції

Після введення початкових даних та відпрацювання першого з двох розроблених методів, а саме методу для аналізу запасів ресурсів, користувач отримає на екрані результати роботи методу. Дані будуть відображені у табличному (рис. 3.6) та графічному вигляді (рис. 3.7).



Початкові запаси ресурсів

Розраховано

Початкові запаси ресурсу В1:	45
Початкові запаси ресурсу В2:	55
Початкові запаси ресурсу В3:	87
Початкові запаси ресурсу В4:	65

Запаси ресурсів, збільшені в 10 разів

В1	В2	В3	В4	Р	QA1	QA2	QA3	QA4	В1*	В2*	В3*	В4*
45	55	87	65	194	23	3	12	8	42	55	85	65
450	55	87	65	197	10	26	1	13	62	55	86	65
45	550	87	65	278	0	0	21	22	44	107	87	63
45	55	870	65	197	26	1	12	9	45	55	92	64
45	55	87	650	226	29	16	13	0	45	55	87	100

Рис. 3.6. Таблична форма виведення результатів роботи методу для аналізу виробничих запасів

У таблиці наведена інформація про початкові значення запасів ресурсів, відповідний план виробництва та значення запасів ресурсів, фактично необхідних для реалізації цього плану. Перший запис таблиці містить вхідні дані задачі. У наступних записах таблиці відповідно до розробленого методу по чергово вдесятеро збільшуються запаси по кожному з видів ресурсів. Також зазначаються відповідні плани виробництва, прибуток та фактичні витрати ресурсів.

Інформація щодо прибутку та фактичних витрат ресурсів виводиться на екран також й у графічному вигляді (рис. 3.7).

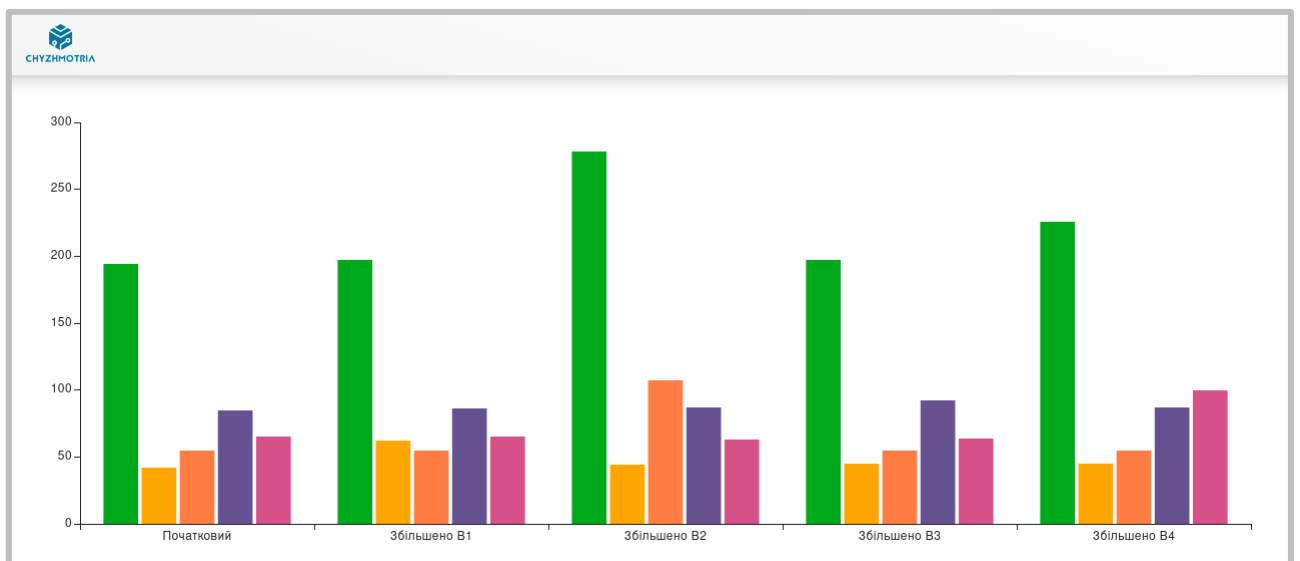


Рис. 3.7. Графічна форма виведення результатів роботи методу для аналізу виробничих запасів

Проміжні результати роботи основного методу розв'язання модифікованої задачі розподілу ресурсів по кожній ітерації виводяться на екран у вигляді зведеної таблиці (рис. 3.8) та у графічному вигляді (рис. 3.10). Значення результуючої змінної PT_1 , яке на відповідній ітерації стає проміжним максимумом, виділені у таблиці зеленим кольором.

 CHYZHMOTRIA											
Результат											
#	P	TQmax	PT1	QA1	QA2	QA3	QA4	B1	B2	B3	B4
72	194	72	2.694	23	3	12	8	42	55	85	65
71	187	66	2.833	26	3	11	7	43	50	87	65
70	187	66	2.833	26	3	11	7	43	50	87	65
69	187	66	2.833	26	3	11	7	43	50	87	65
68	187	66	2.833	26	3	11	7	43	50	87	65
67	187	66	2.833	26	3	11	7	43	50	87	65
66	187	66	2.833	26	3	11	7	43	50	87	65
65	185	63	2.937	25	5	10	7	44	49	86	65
64	185	63	2.937	25	5	10	7	44	49	86	65
63	185	63	2.937	25	5	10	7	44	49	86	65
62	180	60	3.000	27	4	10	6	43	46	86	65
61	180	60	3.000	27	4	10	6	43	46	86	65
60	180	60	3.000	27	4	10	6	43	46	86	65
59	178	54	3.296	26	6	9	6	44	45	85	65
58	178	54	3.296	26	6	9	6	44	45	85	65
57	178	54	3.296	26	6	9	6	44	45	85	65
56	178	54	3.296	26	6	9	6	44	45	85	65
55	178	54	3.296	26	6	9	6	44	45	85	65
54	178	54	3.296	26	6	9	6	44	45	85	65
53	171	51	3.353	7	17	8	5	34	51	54	65

Рис. 3.8. Зведена таблиця результатів роботи методу розв’язання модифікованої задачі розподілу ресурсів

Кращий результат по всіх ітераціях виділений у таблиці фоном зеленого кольору (рис. 3.9).

 CHYZHMOTRIA											
23	84	22	3.818	11	7	3	2	22	20	38	34
22	84	22	3.818	11	7	3	2	22	20	38	34
21	82	21	3.905	10	7	3	2	21	20	36	33
20	78	20	3.900	10	6	3	2	20	19	35	31
19	76	18	4.222	9	6	3	2	19	19	33	30
18	76	18	4.222	9	6	3	2	19	19	33	30
17	57	16	3.563	8	5	2	1	15	13	26	24
16	57	16	3.563	8	5	2	1	15	13	26	24
15	55	15	3.667	7	5	2	1	14	13	24	23
14	51	14	3.643	7	4	2	1	13	12	23	21

Рис. 3.9. Фрагмент зведеної таблиці із позначенням кращим результатом

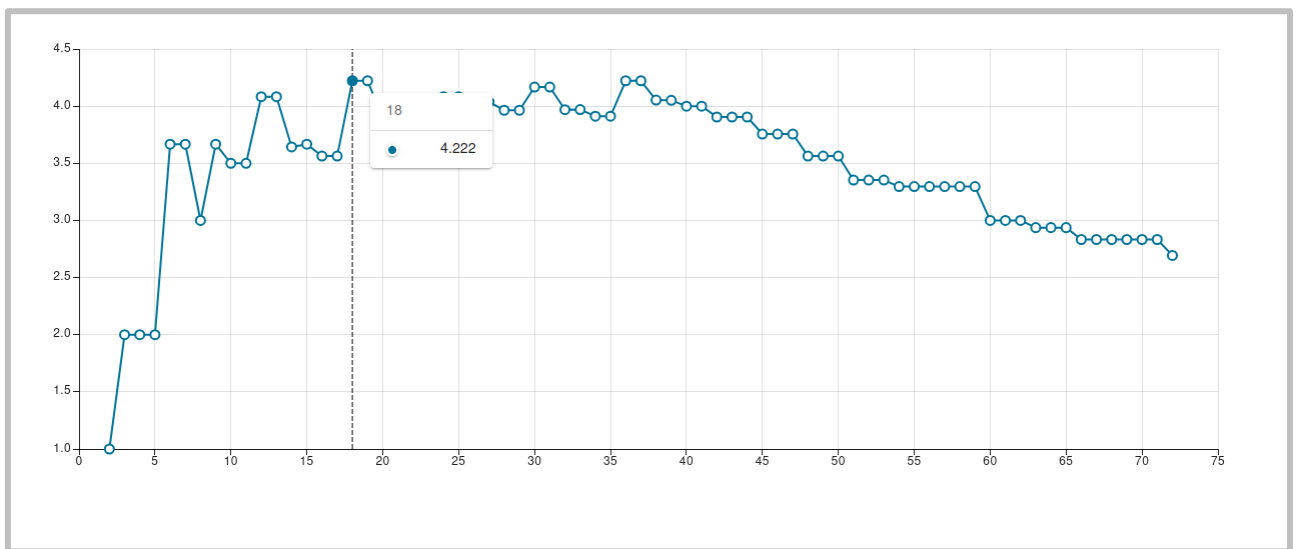


Рис. 3.10. Графік результатів роботи методу розв'язання модифікованої задачі розподілу ресурсів за всіма ітераціями

Результат роботи програми та відповідь задачі подана у текстовому вигляді (рис. 3.11). Інформація містить безпосередньо оптимальний план виробництва, затрачені для цього ресурси, загальний прибуток, час на виконання плану виробництва та знайдений максимальний прибуток за одиницю часу.

Найкращий результат					
Загальний прибуток (P):	76	К-сть товару №1 (QA1):	9	Витрачено ресурсу №1 (B1):	19
Кількість часу для виготовлення (TQmax):	18	К-сть товару №2 (QA2):	6	Витрачено ресурсу №2 (B2):	19
Прибуток за одиницю часу (PT1):	4.222	К-сть товару №3 (QA3):	3	Витрачено ресурсу №3 (B3):	33
		К-сть товару №4 (QA4):	2	Витрачено ресурсу №4 (B4):	30

Рис. 3.11. Результат роботи програми та відповідь задачі

Висновки до третього розділу

У третьому розділі здійснено реалізацію прототипу програмного продукту. Описано визначення архітектури та засобів реалізації прототипу, які мають забезпечувати правильну обробку даних.

Реалізовано окремі модулі для знаходження початкового рішення задачі і рішення задачі з додатковими даними, модифікованої шляхом введення змінних, що визначають тривалості виробничих циклів.

Описано інтерфейс та порядок роботи з вебдодатком від поетапного введення початкових даних задачі до демонстрації результатів відпрацювання двох розроблених методів з графічними формами виведення і зведеними таблицями. Структура інтерфейсу і наочне відображення результуючих даних забезпечує зручний та інтуїтивно зрозумілий порядок роботи.

ВИСНОВКИ

Під час виконання випускної роботи було проведено аналіз предметної області дослідження, розглянуті приклади основних оптимізаційних задач, проведено аналіз вхідних даних задачі розподілу ресурсів виробничого підприємства, виконана постановка задачі дослідження, визначений перелік завдань, які необхідно виконати для реалізації прототипу вебдодатку для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства.

Розроблені та описані власні математична модель та алгоритм розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства, проаналізовано ефективність роботи розробленого методу. Аналіз підтвердив доцільність додавання до моделі показника тривалості виробничого циклу та ефективність розробленого методу розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства. Отримані результати надають точну інформацію для прийняття відповідних управлінських рішень, зокрема, щодо змін у виробництві та управління запасами задля збільшення прибутку підприємства у середньо- та довгостроковій перспективах.

Окрім основного методу розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства, також було розроблено та описано власний метод для аналізу наявних ресурсів підприємства. Результати роботи методу надають можливість провести детальний аналіз наявних запасів виробничих ресурсів підприємства.

Таким чином, в результаті виконання роботи було розроблено прототип вебдодатку для розв'язання модифікованої задачі розподілу ресурсів виробничого підприємства, який може бути корисним при прийнятті управлінських рішень відповідними підрозділами підприємств задля збільшення прибутку та більш ефективного використання виробничих ресурсів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Чижмотря М. О., Чижмотря О. В. Аналіз вхідних даних задачі економіко-математичного моделювання розподілу ресурсів виробничого підприємства. Комп'ютерні технології: інновації, проблеми, рішення: тези VII Всеукраїнської науково-технічної конференції, м. Житомир, 02-03 грудня 2024 р. – Житомир: Державний університет «Житомирська політехніка», 2024. (подано до друку)
2. Чижмотря М. О., Чижмотря О. В. Огляд задач економіко-математичного моделювання. Комп'ютерні технології: інновації, проблеми, рішення: тези VII Всеукраїнської науково-технічної конференції, м. Житомир, 02-03 грудня 2024 р. – Житомир: Державний університет «Житомирська політехніка», 2024. (подано до друку)
3. Стадник Ю. А. Економіко-математичне моделювання. Конспект лекцій. – Львів, 2017. – 44 с.
4. Математичне програмування: Навч. посіб. / Наконечний С. І., Савіна С. С. – К.: КНЕУ, 2003. – 452 с.
5. Левченко І. В. Задачі економіко-математичного моделювання в управлінні підприємствами / І. В. Левченко. // Вісник економіки транспорту і промисловості. – 2019. – С. 34–39.
6. Гончаров В. В., Гончарова С. Я., Кривоблоцька Л. М. Математичне програмування: Навч. посібник. – Кіровоград, 2012. – 151 с.
7. Характеристика економіко-математичних методів и моделей прийняття рішень [Електронний ресурс] / Н. В. Омецинська. – ТОВ «ДКС Центр» – 2015. – Режим доступу до ресурсу: <http://www.economy.nauka.com.ua/?op=1&z=4030>
8. Самсонов В. В. Алгоритми розв'язання задач оптимізації: Навчальний посібник. / Самсонов В. В. – К.: НУХТ, 2014. – 300 с.
9. Економічна кібернетика / Шарапов О. Д., Дербенцев В. Д., Семьонов Д. Є. Навч. посібник. – К.: КНЕУ, 2004. – 231 с.
10. Попов Ю. Д., Тюптя В. І., Шевченко В. І. Методи оптимізації. [Електронний ресурс] / Попов Ю. Д., Тюптя В. І., Шевченко В. І. – Ел. бібліотека

факультету кібернетики Київського національного університету імені Тараса Шевченка, 2003. – 215 с. Режим доступу до ресурсу: <https://www.csc.knu.ua/en/library/books/popov-30.pdf>

11. Глушик М. М., Копич І. М., Пенцак О. С., Сороківський В. М. Математичне програмування: Навч. посібник. – Львів, 2005. – 216 с.

12. Боровська Т. М. Оптимальна система управління запасами при невизначеності попиту / Т. М. Боровська, П. В. Северілов, А. С. Васюра. – Інформаційні технології та комп'ютерна інженерія: – 2006. – № 1(5) – С. 113–118.

13. Прохорова В. В. Організація виробництва: Навч. посібник. – Х., 2018. – 275 с.

14. Економіка підприємства: підручник. / І. М. Бойчик. – К.: Кондор - Видавництво, 2016. – 378 с.

15. Застосування симплекс-методу для розв'язання прикладних задач / Халанчук, А. В. – ТДАТУ – 2021. – С. 49–51

16. Задачі цілочисельного програмування. Метод Гоморі / Малкіна В. М., Зінов'єва О. Г. – Таврійський державний агротехнологічний університет, 2017. – 16 с.

17. Econometrics: Definition, Models, and Methods [Електронний ресурс] / Adam Hayes. – Investopedia, 2024. – Режим доступу до ресурсу: <https://www.investopedia.com/terms/e/econometrics.asp>

18. Економетрія: Підручник. вид. 3-тє, доп. та перероб. / Наконечний С. І., Терещенко Т. О., Романюк Т. П. – К.: КНЕУ, 2006. 520 с.

19. Maximization By The Simplex Method [Електронний ресурс] / Rupinder Sekhon and Roberta Bloom. – De Anza College, 2016. – Режим доступу до ресурсу: [https://math.libretexts.org/Bookshelves/Applied_Mathematics/Applied_Finite_Mathematics_\(Sekhon_and_Bloom\)/04%3A_Linear_Programming_The_Simplex_Method/4.02%3A_Maximization_By_The_Simplex_Method](https://math.libretexts.org/Bookshelves/Applied_Mathematics/Applied_Finite_Mathematics_(Sekhon_and_Bloom)/04%3A_Linear_Programming_The_Simplex_Method/4.02%3A_Maximization_By_The_Simplex_Method)

20. Winston W. L. Operations Research: Applications and Algorithms / Wayne L. Winston. – 4th ed. – Belmont, CA: Cengage Learning, 2004. – 672 p.

21. Taha H. A. Operations Research: An Introduction / Hamdy A. Taha. – 10th ed. – Upper Saddle River, NJ: Pearson, 2017. – 960 p.
22. Bazaraa M. S., Jarvis J. J., Sherali H. D. Linear Programming and Network Flows / Mokhtar S. Bazaraa, John J. Jarvis, Hanif D. Sherali. – 4th ed. – Hoboken, NJ: Wiley, 2010. – 720 p.
23. Dantzig G. B., Thapa M. N. Linear Programming 1: Introduction / George B. Dantzig, Mukund N. Thapa. – New York: Springer, 2003. – 400 с.
24. Метод Жордана-Гауса для Розв'язання Систем Лінійних Рівнянь: Від Теорії до Практики [Електронний ресурс] / Ростислав Верещак // www.mathros.net.ua – 2023. – Режим доступу до ресурсу: <https://www.mathros.net.ua/metod-zhordana-gaussa-rozvjazok-system-linijnyh-rivnjan-metodom-zhordana-gaussa.html>
25. Dual Simplex [Електронний ресурс] / Mihai Banciu. – ResearchGate GmbH – 2011. – Режим доступу до ресурсу: https://www.researchgate.net/publication/233748737_Dual_Simplex
26. Simplex method | Definition, Example, Procedure, & Facts [Електронний ресурс] / Stephen J. Wright // www.britannica.com – 2024. – Режим доступу до ресурсу: <https://www.britannica.com/topic/simplex-method>
27. Optimization of Production Benefits Through the Linear Program Graph Method: A Case Study Zentha Meubel Hersiyati [Електронний ресурс] / Н. Palayukan. Mathematics Education Journal – 2021. – Режим доступу до ресурсу: <https://ejournal.umm.ac.id/index.php/MEJ/article/view/15055>
28. Документація RESTful API [Електронний ресурс] – Режим доступу до ресурсу: <https://aws.amazon.com/what-is/restful-api/>.
29. Документація JSON [Електронний ресурс] – Режим доступу до ресурсу: <https://www.json.org/json-en.html>.
30. Документація XML [Електронний ресурс] – Режим доступу до ресурсу: <https://aws.amazon.com/what-is/xml/>.
31. Документація Symfony [Електронний ресурс] – Режим доступу до ресурсу: <https://symfony.com/doc/current/index.html>.

32. Документація Doctrine [Електронний ресурс] – Режим доступу до ресурсу: <https://www.doctrine-project.org/projects/doctrineorm/en/2.15/index.html>
33. Документація MySQL [Електронний ресурс] – Режим доступу до ресурсу: <https://dev.mysql.com/doc/>.
34. Документація React [Електронний ресурс] – Режим доступу до ресурсу: <https://legacy.reactjs.org/docs/getting-started.html>.
35. Документація ApiPlatform [Електронний ресурс] – Режим доступу до ресурсу: <https://api-platform.com/docs/core/>.
36. Документація Python [Електронний ресурс] – Режим доступу до ресурсу: <https://www.python.org/>.
37. Документація NumPy [Електронний ресурс] – Режим доступу до ресурсу: <https://numpy.org/>.
38. Документація Docker [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.docker.com/>.
39. Документація NGINX [Електронний ресурс] – Режим доступу до ресурсу: <https://www.nginx.com/resources/glossary/nginx/>.
40. Документація Axios [Електронний ресурс] – Режим доступу до ресурсу: <https://axios-http.com/docs/intro>.

ДОДАТКИ

Лістинг коду реалізації двофазного симплекс-методу

```

def two_phase(dct,bss,n_initial):

    ans = {'initial_tableau': None, 'final_tableau': None, 'solution_status': '',
'optimal_solution': None, 'optimal_value': None}

    n_original = len(dct['[A]'][0])
    unbdd = False

    for i in range(len(dct['[b]'])):
        if dct['[b]'][i][0]<fr(0):
            dct['[A]'][i] = fr(-1)*dct['[A]'][i]
            dct['[b]'][i] = fr(-1)*dct['[b]'][i]

    nb = 0
    ids = []

    for i in range(len(dct['[A]'][0])):
        if (nb==len(dct['[A]'])):
            break
        column_vector = dct['[A]'][:, i]
        cond1 = (np.sum(column_vector==0) == (len(column_vector) - 1))
        is_condition_met = False
        if (cond1):
            for j in range(len(column_vector)):
                if (column_vector[j]>fr(0)):
                    ids.append(j)
                    is_condition_met = True
                    break
        if (is_condition_met):
            nb+=1
            bss['idx'].append(i)
            bss['vars'].append(dct['vars'][i])

    flag = True
    if (nb==len(dct['[A]'])):
        flag = False

    if (flag):
        m = len(dct['[A]'])
        n = len(dct['[A]'][0])

        c_new = np.full(dct['[c]'].shape, fr("0"), dtype = fr)

        var = 'y'
        art = 0
        for i in range(m):
            if i not in ids:
                bss['vars'].insert(i,var+str(art))
                bss['idx'].insert(i,len(dct['[A]'][0]))
                dct['vars'].append(var+str(art))
                col = np.full((m, 1), fr("0"), dtype = fr)
                col[i][0]=1
                dct['[A]'] = np.hstack((dct['[A]'],col))
                c_new = np.append(c_new, [[fr("1")]], axis=1)
                art+=1

        m = len(dct['[A]'])
        n = len(dct['[A]'][0])

```

```

dct['tableau']=np.full((m+1, n+1), fr("0"), dtype = fr)
bss['mtx'] = dct['[A]'][:,bss['idx']]
B = bss['mtx']
B_1 = np.copy(bss['mtx'])
np.fill_diagonal(B_1, fr(1) / np.diag(B))

xB = B_1@dct['[b]']
dct['tableau'][1:,0]= (B_1@dct['[b]']).reshape(m,)

h = B_1@dct['[A]']
dct['tableau'][1:,1:] = h

cB = c_new[:,bss['idx']]

dct['tableau'][0,1:] = (c_new +fr(-1)*cB@h).reshape(n,)
dct['tableau'][0,0] = (fr(-1)*cB@xB)[0][0]

dct['[c_new]'] = c_new
ans['initial_tableau'] = np.copy(dct['tableau'][1:])

while True:
    found = False
    j = 0
    for idx in range(1,n+1):
        if (dct['tableau'][0,idx]<fr("0")):
            found = True
            j = idx
            break

    if not found:
        break

    u = dct['tableau'][1:,j]

    if (np.all(u<=fr("0"))):
        unbdd = True
        break

    ratios = []

    for i in range(len(u)):
        if (u[i]>fr("0")):
            ratios.append(dct['tableau'][i+1,0]/u[i])

    mini = min(ratios)

    l = 0
    for i in range(len(u)):
        if (u[i]>fr("0")):
            if dct['tableau'][i+1,0]/u[i] == mini:
                l = i+1
                break

    dct['tableau'][1] = dct['tableau'][1]/dct['tableau'][1][j]

    for k in range(m+1):
        if k!=l:
            dct['tableau'][k] = dct['tableau'][k] +fr(-
1)*dct['tableau'][k][j]*dct['tableau'][1]

    bss['vars'][l-1] = dct['vars'][j-1]

```

```

        bss['idx'][l-1] = j-1

    if (unbdd):
        ans['final_tableau'] = dct['tableau'][1:]
        ans['solution_status'] = 'unbounded'
        return ans

    if (abs(dct['tableau'][0][0])>fr("0")):
        ans['final_tableau'] = dct['tableau'][1:]
        ans['solution_status'] = 'infeasible'
        return ans

    drive_out = []
    for k in range(len(bss['vars'])):
        if bss['vars'][k][0]=='y':
            drive_out.append(k)

    popy = []
    while (len(drive_out)!=0):
        l = drive_out.pop()
        all_zero = True
        for j in range(n_original):
            if dct['tableau'][l+1][j+1]!=0:
                all_zero=False

                dct['tableau'][l+1] = dct['tableau'][l+1]/dct['tableau'][l+1][j+1]
                for k in range(m+1):
                    if k!=l+1:
                        dct['tableau'][k] = dct['tableau'][k] +fr(-1)*
dct['tableau'][k][j+1]*dct['tableau'][l+1]

                bss['vars'][l] = dct['vars'][j]
                bss['idx'][l] = j

        if (all_zero):
            popy.append(l+1)

    for id in popy:
        dct['[A]'] = np.delete(dct['[A]'],id-1, axis = 0)
        dct['[b]'] = np.delete(dct['[b]'],id-1, axis =0)
        dct['tableau'] = np.delete(dct['tableau'],id, axis=0)
        bss['vars'].pop(id-1)
        bss['idx'].pop(id-1)

    bss['mtx'] = dct['[A]'][:,bss['idx']]
    dct.pop('[c_new]')

    start_idx = 0

    for k in range(len(dct['vars'])):
        if (dct['vars'][k][0]=='y'):
            start_idx=k
            break

    if (k!=n+1):
        tableau = np.delete(dct['tableau'], np.s_[start_idx+1:], axis=1)
        dct['tableau'] = tableau
        A = np.delete(dct['[A]'], np.s_[start_idx:], axis=1)
        dct['[A]'] = A
        del dct['vars'][start_idx:]

    n = len(dct['[A]'][0])

```

```

h = dct['tableau'][1:,1:]
xB = dct['tableau'][1:,0]

cB = dct['[c]'][:,bss['idx']]

dct['tableau'][0,1:] = (dct['[c]'] + fr(-1)*cB@h).reshape(n,)
dct['tableau'][0,0] = (fr(-1)*cB@xB)[0]

else:
    m = len(dct['[A]'])
    n = len(dct['[A]'][0])

    dct['tableau']=np.full((m+1, n+1), fr("0"), dtype = fr)
    bss['mtx'] = dct['[A]'][:,bss['idx']]
    B = bss['mtx']
    B_1 = np.copy(bss['mtx'])
    np.fill_diagonal(B_1, 1 / np.diag(B))

    xB = B_1@dct['[b]']
    dct['tableau'][1:,0]= (B_1@dct['[b]']).reshape(m,)

    h = B_1@dct['[A]']
    dct['tableau'][1:,1:] = h

    cB = dct['[c]'][:,bss['idx']]

    dct['tableau'][0,1:] = (dct['[c]'] + fr(-1)*cB@h).reshape(n,)

    dct['tableau'][0,0] = (fr(-1)*cB@xB)[0][0]

    ans['initial_tableau'] = np.copy(dct['tableau'][1:])

m = len(dct['[A]'])
n = len(dct['[A]'][0])
unbdd = False

while True:
    found = False
    j = 0
    for idx in range(1,n+1):
        if (dct['tableau'][0,idx]<fr("0")):
            found = True
            j = idx
            break

    if not found:
        break

    u = dct['tableau'][1:,j]

    if (np.all(u<=fr("0"))):
        unbdd = True
        break

    ratios = []

    for i in range(len(u)):
        if (u[i]>fr("0")):
            ratios.append(dct['tableau'][i+1,0]/u[i])

    mini = min(ratios)
    l = 0

```

```

for i in range(len(u)):
    if u[i]>fr("0") and dct['tableau'][i+1,0]/u[i] == mini:
        l = i+1
        break

dct['tableau'][l] = dct['tableau'][l]/dct['tableau'][l][j]

for k in range(m+1):
    if k!=l:
        dct['tableau'][k] = dct['tableau'][k] + fr(-
1)*dct['tableau'][k][j]*dct['tableau'][l]

    bss['vars'][l-1] = dct['vars'][j-1]
    bss['idx'][l-1] = j-1

if (unbdd):
    ans['final_tableau'] = dct['tableau'][1:]
    ans['solution_status'] = 'unbounded'
    return ans

ans['final_tableau'] = dct['tableau'][1:]
ans['solution_status'] = 'optimal'
if (dct['[objective]'][0]=='minimize'):
    ans['optimal_value'] = fr("-1")*dct['tableau'][0][0]
else:
    ans['optimal_value'] = dct['tableau'][0][0]

vec = {}
var = 'x'
for i in range(n_initial):
    vec[var + str(i)] = fr("0")

for i in range(len(bss['vars'])):
    if (bss['vars'][i]) in vec:
        vec[bss['vars'][i]] = dct['tableau'][i+1][0]

ans['optimal_solution'] = []

for i in range(n_initial):
    ans['optimal_solution'].append(fr(vec[var + str(i)]))

return ans

```

Лістинг коду реалізації алгоритму Гоморі

```

def gomory_cut_algo_fun(A, b, c, b_filler = None):

    dct = {}
    {'[objective]': ['maximize'], '[A]': [], '[b]': [], '[constraint_types]': [], '[c]': []}
    bss = {'vars': [], 'idx': []}
    ans = {'initial_solution': None, 'final_solution': None, 'solution_status': '',
    'number_of_cuts': 0, 'optimal_value': None}

    n_initial = input_handling(dct, A, b, c, b_filler)

    ans_initial = two_phase(dct, bss, n_initial)
    ans['initial_solution'] = np.copy(ans_initial['optimal_solution'])
    ans['final_solution'] = np.copy(ans_initial['optimal_solution'])

    if (ans_initial['solution_status'] != 'optimal'):
        ans['solution_status'] = ans_initial['solution_status']
        return ans

    var = 's'
    cnt = 0

    while True:
        check = True
        for i in range(len(ans['final_solution'])):
            if (not is_integer(ans['final_solution'][i])):
                check = False
                break

        if (check):
            ans['solution_status'] = 'optimal'
            ans['optimal_value'] = dct['[c]'][0][:n_initial]@ans['final_solution']
            if (dct['[objective]'][0] == 'maximize'):
                ans['optimal_value'] = fr(-1)*ans['optimal_value']
            return ans
        else:
            ans['number_of_cuts'] += 1

            ind = 0

            for i in range(len(dct['tableau'])):
                if (not is_integer(dct['tableau'][i, 0])):
                    ind = i
                    break

            new_row = np.full(dct['tableau'][ind].shape, fr("0"), dtype = fr)
            for i in range(len(dct['tableau'][ind])):
                new_row[i] = fr((-1)*fractional_part(dct['tableau'][ind][i]))

            dct['tableau'] = np.vstack((dct['tableau'], new_row))

            new_col = np.full((len(dct['tableau']), 1), fr("0"), dtype = fr)
            new_col[-1][0] = fr("1")

            dct['tableau'] = np.hstack((dct['tableau'], new_col))

            dct['vars'].append(var + str(cnt))
            cnt += 1
            bss['idx'].append(len(dct['vars']) - 1)

```

```

bss['vars'].append(dct['vars'][-1])

ans_dual = dual_simplex(dct,bss,n_initial)
ans['final_solution'] = np.copy(ans_dual['optimal_solution'])

keep_row = [0]
popy_bss = []
popy_var = []
bool_col = [1 for i in range(len(dct['tableau'][0]))]
for i in range(len(bss['vars'])):
    if (bss['vars'][i][0]!='s'):
        keep_row.append(i+1)
    else:
        bool_col[bss['idx'][i] + 1] = 0
        popy_bss.append(i)
        popy_var.append(bss['idx'][i])

keep_col = []
for i in range(len(bool_col)):
    if (bool_col[i]):
        keep_col.append(i)

if (len(popy_bss)>0):
    popy_set = set(popy_var)
    in popy_set]
    dct['vars'] = [elem for idx, elem in enumerate(dct['vars']) if idx not
    in popy_set]

    popy_set_2 = set(popy_bss)
    in popy_set_2]
    bss['vars'] = [elem for idx, elem in enumerate(bss['vars']) if idx not
    in popy_set_2]

    vars_indices_map = {var: idx for idx, var in enumerate(dct['vars'])}

    bss['idx'] = [vars_indices_map[var] for var in bss['vars']]

    dct['tableau'] = dct['tableau'][keep_row,:]
    dct['tableau'] = dct['tableau'][:,keep_col]

if (ans_dual['solution_status']=='unbounded'):
    ans['solution_status'] = 'infeasible'
return ans

```

Лістинг коду реалізації двоїстого симплекс-методу

```

def dual_simplex(dct, bss, n_initial):
    ans = {'initial_tableau': None, 'final_tableau': None, 'solution_status': '',
          'optimal_solution': None, 'optimal_value': None}
    ans['initial_tableau'] = np.copy(dct['tableau'])

    m = len(dct['tableau']) - 1
    unbdd = False

    while True:
        idx = 0
        found = False
        for i in range(1, m+1):
            if (dct['tableau'][i, 0] < fr("0")):
                idx = i-1
                found = True
                break

        if not found:
            break

        v = dct['tableau'][idx+1, 1:]

        if (np.all(v>=0)):
            unbdd = True
            break

        ratios = []
        for i in range(len(v)):
            if (v[i]<fr(0)):
                ratios.append(dct['tableau'][0, i+1]/abs(v[i]))
        mini = min(ratios)

        l = 0
        minis = []
        for i in range(len(v)):
            if (v[i]<fr(0)) and ((dct['tableau'][0, i+1]/abs(v[i])) == mini):
                minis.append(i+1)

        if len(minis)>1:
            all_cols = np.copy(dct['tableau'][:, minis])
            for j in range(len(minis)):
                all_cols[:, j] = all_cols[:, j]/abs( v[ minis[j]-1 ] )
            comp = [list(lst) for lst in all_cols.T]
            lst = min(comp)
            l = minis[comp.index(lst)]
        else:
            l = minis[0]

        dct['tableau'][idx+1] = dct['tableau'][idx+1]/dct['tableau'][idx+1, l]

        for k in range(m+1):
            if k!=idx+1:
                dct['tableau'][k] = dct['tableau'][k] + fr("-1")*dct['tableau'][k, l]*dct['tableau'][idx+1, l]

        bss['vars'][idx] = dct['vars'][l-1]
        bss['idx'][idx] = l-1

```

```

if (unbdd):
    ans['final_tableau'] = np.copy(dct['tableau'])
    ans['solution_status'] = 'unbounded'
    return ans

ans['final_tableau'] = np.copy(dct['tableau'])
ans['solution_status'] = 'optimal'
if (dct['[objective]'][0]=='minimize'):
    ans['optimal_value'] = fr(-1)*dct['tableau'][0][0]
else:
    ans['optimal_value'] = dct['tableau'][0][0]

vec = {}
var = 'x'
for i in range(n_initial):
    vec[var + str(i)] = fr("0")

    vec.update({var: dct['tableau'][i+1][0] for i, var in enumerate(bss['vars']) if var
in vec})
ans['optimal_solution'] = []

for i in range(n_initial):
    ans['optimal_solution'].append(fr(vec[var + str(i)]))

return ans

```