Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Факультет інформаційних технологій
(факультет)

Кафедра системного аналізу та управління
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня бакалавра

Здобувача вищої освіти Біліми Вероніки Костянтинівни

академічної групи 124-21-1

спеціальності 124 Системний аналіз

за освітньо-професійною програмою Системний аналіз

на тему: «Підвищення ефективності розподілу виробничих ресурсів при виготовлені пластикових кришок»

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**Інституційною**	
кваліфікаційної роботи	*доц. Хабарлак К.С.*			
розділів:				
Інформаційно-аналітичний	*доц. Хабарлак К.С.*			
Спеціальний розділ	*доц. Хабарлак К.С.*			
Рецензент				
Нормоконтролер	*доц. Хом'як Т.В.*			

Дніпро
2025

ЗАТВЕРДЖЕНО:
завідувач кафедри
Системного аналізу та управління
(повна назва)

_____ *к.т.н., доц. Желдак Т.А.*
(підпис) (прізвище, ініціали)

«____» _____20___року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня *бакалавра*

здобувачу вищої освіти *Білімі В.К.* **академічної групи** *124-21-1*
спеціальності: *124 Системний аналіз*
за освітньо-професійною програмою Системний аналіз
на тему «*Підвищення ефективності розподілу виробничих ресурсів при виготовленні пластикових кришок»*
затверджену наказом ректора НТУ «Дніпровська політехніка»
від *05.05.2025 р. №336-с*

Розділ	Зміст	Терміни виконання
1. Інформаційно-аналітичний розділ	*Проаналізувати виробничий процес виготовлення пластикових кришок. Визначити предмет, об'єкт дослідження та проблему, що розв'язується. Розглянути методи планування виробництва, зокрема жадібні алгоритми та job-shop scheduling, і обґрунтувати вибір оптимізаційного підходу.*	10.09.2024 – 01.03.2025
2.Спеціальний розділ	*Розв'язати поставлені задачі: розробити систему оптимізації розподілу виробничих ресурсів із урахуванням дедлайнів, переналадки та сумісності пресформ. Реалізувати модифікований job-shop підхід і веб-інтерфейс для перегляду розкладу.*	01.03.2025 – 30.05.2025

Завдання видано _____ *доц. Хабарлак К.С.*
(підпис) (прізвище, ініціали)

Дата видачі: *09.09.2024 р.*
Дата подання до екзаменаційної комісії: *09.06.2025*
Прийнято до виконання _____ Біліма В.К.
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 47 с., 14 рис., 8 додатків, 12 джерел.

Об’єктом дослідження є виробничий процес виготовлення пластикових кришок на підприємстві.

Предметом дослідження виступає алгоритмічна модель планування виробництва, яка дозволяє автоматизувати формування ефективного розкладу виготовлення продукції з урахуванням технологічних обмежень.

Метою кваліфікаційної роботи є підвищення ефективності розподілу виробничих ресурсів шляхом створення автоматизованої системи побудови виробничого розкладу із застосуванням жадібного алгоритму та модифікованого підходу до задачі job-shop scheduling.

Методи дослідження: системний аналіз виробничих процесів, побудова концептуальної моделі бази даних, розробка алгоритмів у середовищі Python, моделювання задачі розкладу як NP-складної задачі, а також реалізація веб-інтерфейсу для візуалізації результатів.

У теоретичному розділі представлено огляд існуючих підходів до виробничого планування, розкрито особливості жадібних алгоритмів і задачі типу job-shop.

У практичному розділі здійснено проєктування бази даних у середовищі MySQL, реалізовано два підходи до побудови розкладу (базовий жадібний та модифікований job-shop із локальною оптимізацією), створено веб-інтерфейс на Flask для перегляду результатів. Проведено порівняльний аналіз отриманих розкладів за критеріями витрат на переналадку, середнього часу очікування та рівня завантаження машин.

Практична цінність полягає в тому, що розроблена система дозволяє автоматизувати складний процес планування, скоротити час на підготовку розкладу, зменшити виробничі втрати, а також забезпечує гнучке масштабування під реальні умови підприємства.

В подальшому планується вдосконалити модель з урахуванням додаткових факторів (наприклад, доступності працівників, енерговитрат), що сприятиме ще глибшій інтеграції в управлінські процеси.

Ключові слова: ВИРОБНИЧЕ ПЛАНУВАННЯ, JOB-SHOP SCHEDULING, ЖАДІБНИЙ АЛГОРИТМ, РОЗПОДІЛ РЕСУРСІВ, БАЗА ДАНИХ, PYTHON, MYSQL, FLASK, ЦИКЛ ВИРОБНИЦТВА, СИСТЕМНИЙ АНАЛІЗ.

ABSTRACT

Explanatory note: 47 pages, 14 figures, 8 appendices, 12 references.

The object of the research is the production process of manufacturing plastic caps at an industrial enterprise.

The subject of the research is an algorithmic model for production scheduling, aimed at automating the generation of an efficient manufacturing timetable considering technological constraints.

The purpose of the qualification thesis is to improve the efficiency of production resource allocation by developing an automated system for constructing a production schedule using a greedy algorithm and a modified job-shop scheduling approach.

Research methods: system analysis of production processes, conceptual database design, algorithm development in Python, modeling of scheduling as an NP-hard problem, and implementation of a web interface for result visualization.

The theoretical part of the work provides an overview of existing production planning methods, explains the principles of greedy algorithms and job-shop problems, and analyzes their relevance to manufacturing tasks.

The practical part includes the design of a relational database using MySQL, implementation of two scheduling algorithms (a basic greedy approach and a modified job-shop model with local optimization), and development of a web interface in Flask to display the results. A comparative analysis of the two methods is conducted based on transition time, average order waiting time, and machine utilization levels.

The practical value of the developed system lies in automating a complex scheduling process, reducing the time required for planning, minimizing production losses, and enabling flexible adaptation to real enterprise conditions.

In future we plan to improve the model by incliding additional constraints (e.g., worker availability or energy consumption), which will support deeper integration into enterprise management systems.

Keywords: PRODUCTION SCHEDULING, JOB-SHOP SCHEDULING, GREEDY ALGORITHM, RESOURCE ALLOCATION, DATABASE, PYTHON, MYSQL, FLASK, MANUFACTURING CYCLE, SYSTEM ANALYSIS.

ЗМІСТ

ВСТУП

У сучасних умовах високої конкуренції на ринку полімерної упаковки підприємства змушені постійно вдосконалювати виробничі процеси з метою зменшення витрат, підвищення гнучкості та якості обслуговування клієнтів. Однією з актуальних проблем є формування ефективного виробничого розкладу, особливо для підприємств, що працюють з багатьма замовленнями, короткими дедлайнами та обмеженими виробничими потужностями.

На прикладі діяльності ТОВ «Ретал Україна», що спеціалізується на виготовленні пластикових кришок, виявлено, що відсутність автоматизованої системи планування призводить до значних втрат часу, великої кількості помилок під час ручного складання графіків, браку продукції та перевитрат ресурсів. Формування виробничого розкладу в Excel займає від одного до двох тижнів, особливо в пікові періоди, що свідчить про необхідність оптимізації даного процесу.

З огляду на вищезазначене, розробка програмного забезпечення для автоматизованого планування з використанням алгоритмів оптимізації, зокрема жадібного підходу та модифікованої моделі job-shop scheduling, є надзвичайно актуальною та має вагоме практичне значення для реального сектору економіки України.

Метою кваліфікаційної роботи є створення інтелектуальної системи планування виробництва пластикових кришок із використанням жадібного алгоритму та адаптованого підходу job-shop scheduling, яка дозволить зменшити витрати часу на складання графіка, мінімізувати переналадки та підвищити ефективність використання виробничого обладнання.

Для досягнення цієї мети поставлено такі завдання:

- проаналізувати існуючі підходи до планування виробництва;
- дослідити структуру виробничого процесу підприємства ТОВ «Ретал Україна»;

- спроєктувати та реалізувати структуру бази даних для зберігання інформації про замовлення, машини, пресформи та продукцію;
- розробити жадібний алгоритм планування з урахуванням дедлайнів і переналадок;
- модифікувати підхід job-shop scheduling для одноопераційних завдань;
- реалізувати локальну оптимізацію порядку виконання замовлень;
- створити веб-інтерфейс для перегляду розкладу;
- провести порівняльний аналіз ефективності моделі.

Об’єктом дослідження є процес виробничого планування на підприємстві, що виготовляє пластикові кришки.

Предметом дослідження є алгоритми розподілу виробничих ресурсів, що враховують дедлайни, переналадку пресформ і черговість замовлень.

У роботі використано методи системного аналізу для вивчення структури виробничого процесу; алгоритмічні методи, зокрема жадібний підхід та модифікований job-shop scheduling – для побудови оптимального розкладу; методи об’єктно-орієнтованого програмування (Python + Flask) – для реалізації функціоналу системи; а також методи логічного й евристичного аналізу – для локального удосконалення черговості завдань. Також при формуванні методичної бази дослідження керувалися рекомендаціями кваліфікаційної роботи бакалавра з системного аналізу [10].

Було розроблено та реалізовано модифікований підхід до job-shop scheduling для одноопераційного виробництва, з урахуванням часу переналадки між замовленнями різних кольорів і форм. Запропоновано алгоритм локальної оптимізації розкладу, який дозволяє зменшити витрати на переналадку без порушення строків доставки. Отримані результати поєднують переваги класичних евристик із практичними обмеженнями реального виробництва.

Розроблене рішення може бути використане на підприємствах із серійним та дрібносерійним виробництвом для автоматизації планування. Реалізована

система дозволяє суттєво скоротити час створення розкладу, знизити кількість помилок, уникнути перевитрат матеріалів і підвищити загальну ефективність виробничих процесів. Система легко адаптується до нових форматів продукції або змін у структурі виробництва.

ІНФОРМАЦІЙНО-АНАЛІТИЧНИЙ РОЗДІЛ

1.1 Аналіз виробничого процесу виготовлення пластикових кришок

Процес виготовлення пластикових кришок на ТОВ «Ретал Україна» є одним із ключових напрямків діяльності підприємства, що спеціалізується на виробництві полімерної упаковки для харчової та нехарчової продукції. Високий рівень автоматизації, сучасне технічне оснащення та гнучка система планування дозволяють підприємству оперативно реагувати на запити клієнтів та забезпечувати високу якість продукції.

Основна сировина для виробництва пластикових кришок – поліетилентерефталат (ПЕТ), який піддається інжекційному формуванню за допомогою прес-форм. Кожен тип кришки має чітко визначені параметри: діаметр, вагу, тип горловини та колір. Найпоширенішими форматами є 1881 (28 мм, низьке горло), 1810 (28 мм, високе горло), формати для пляшок з ручками – 38 мм та 48 мм, а також спеціалізований формат 1881 Sacmi, який виробляється на окремій машині для одного клієнта. Для кожної ваги передбачено відповідну прес-форму, яка встановлюється на відповідну машину.

Кожна прес-форма може бути сумісна лише з певними машинами. Наприклад, кришка 2,75 г виробляється лише за наявності форми ПФ-26, яка може бути встановлена лише на одну машину одночасно. Це створює обмеження у розподілі ресурсів і вимагає ретельного планування для уникнення конфліктів при паралельному виробництві. Загалом у виробництві використовуються понад 10 машин, кожна з яких має певний набір сумісних форматів, що визначає їх спеціалізацію.

Додатковою складністю у виробничому процесі є зміна кольору кришки, що потребує очищення обладнання від залишків барвника. Перехід від темних кольорів до світлих, зокрема з чорного на білий, потребує до трьох годин технічної чистки. Щоб уникнути простою, іноді у якості перехідного етапу

виробляються прозорі кришки – цей колір не потребує додаткового барвника, тому він швидко вимиває попередній залишок кольору. Таким чином, навіть технічні обмеження виробничого процесу враховуються у логіці формування розкладу.

Окрему увагу варто приділити мінімальним партіям замовлень: для 28 мм – 500 000 штук, 38 мм – 300 000, 48 мм – 100 000. Якщо клієнт замовляє менший обсяг, його замовлення або об'єднується з іншим, або планується у вигляді «доважка» до вже існуючого замовлення з аналогічними параметрами (особливо кольором і вагою), або, якщо неможливо – відхиляється. Це вимагає розробки гнучких алгоритмів об'єднання дрібних партій та їх ефективного включення у виробничий план.

Замовлення можуть надходити по-різному: деякі клієнти формують графік на весь місяць, інші надсилають тижневі графіки з рівномірним розподілом, треті – дають лише загальну потребу. Проте загальне правило вимагає надати всі замовлення на наступний місяць до 20 числа поточного. Незважаючи на це, на практиці близько 20% загального обсягу виробництва становлять додаткові або форс-мажорні замовлення. Виробництво має буфер у вигляді 3–5 днів запасу, що дозволяє гнучко адаптуватися до змін у графіку.

Усі ці фактори – обмеження щодо сумісності прес-форм, тривалість переналадки, мінімальні партії, різні підходи клієнтів до планування – потребують складної, добре структурованої системи управління розкладом виробництва. Особливе значення в цьому контексті має департамент планування, зокрема відділ планування плівки та кришок, де під час проходження практики було детально проаналізовано логіку побудови графіків виробництва.

Виробництво кришок не є безперервним через потребу в частих переналадках обладнання. Зміна прес-форми або кольору – це не лише часові витрати, а й ризик браку, тому оптимізація розкладу має враховувати порядок замовлень, щоб зменшити кількість переналадок. Наприклад, перехід «білий → прозорий → червоний» є логістично ефективнішим, ніж «чорний → білий», і це обов'язково повинно враховуватись при побудові плану.

Таким чином, виробничий процес виготовлення пластикових кришок на підприємстві ТОВ «Ретал Україна» є складним багатофакторним середовищем, що поєднує високотехнологічне обладнання з гнучкою системою управління. Автоматизація цього процесу за допомогою інтелектуальних алгоритмів, зокрема евристичних методів і модифікованого job-shop scheduling, є актуальним і необхідним етапом підвищення ефективності підприємства.

Власне ці труднощі й стали причиною вибору теми дослідження. Під час проходження виробничої практики я безпосередньо спостерігала, як формування графіків виробництва здійснюється вручну – за допомогою Excel-таблиць, візуальної оцінки навантаження машин, і часто без урахування реального часу переналадки. Цей процес займав від одного тижня до двох, залежно від сезону, особливо складним був період підготовки до пікових замовлень. Через людський фактор регулярно допускалися помилки, графіки доводилось виправляти кілька разів, а це призводило до збоїв у постачанні, виробничого браку та нераціонального використання ресурсів. Ситуація вказала на критичну потребу у впровадженні програмного рішення, яке могло б оптимізувати планування виробництва, зменшити вплив людського фактору та підвищити точність і надійність розкладу. Саме тому я обрала цю тему для своєї кваліфікаційної роботи.

1.2 Характеристика об’єкта дослідження. Постановка задачі

Об’єктом дослідження в межах цієї кваліфікаційної роботи є виробничий процес планування на підприємстві, що спеціалізується на виготовленні пластикових кришок методом лиття під тиском [11]. Це виробництво характеризується багатьма взаємопов’язаними параметрами, серед яких варто виділити: багатономенклатурність продукції, обмеження за типом пресформи та машини, значну варіативність кольорів, наявність дедлайнів, що не підлягають порушенню, а також значний вплив переналадок на загальну тривалість виробничого циклу.

Підприємство має у своєму розпорядженні декілька виробничих машин, кожна з яких сумісна з обмеженим набором пресформ. Пресформи, у свою чергу, сумісні з конкретними форматами та вагами кришок. На одну машину одночасно може бути встановлена лише одна пресформа, яка потребує часу на заміну при зміні формату продукції. Крім того, зміна кольору кришки передбачає очищення обладнання, яке також займає час – від 20 хвилин до понад 2 годин залежно від комбінації кольорів. Таким чином, впорядкування черговості замовлень із метою мінімізації кількості переналадок є критичним фактором ефективності всього процесу.

Щомісяця до відділу планування надходять замовлення від десятків клієнтів, які потребують виконання в суворо визначені терміни. Мінімальні партії залежно від типу продукції варіюються від 100 до 500 тис. одиниць. Замовлення відображаються в системі обліку (1С), однак подальше планування здійснюється вручну – у вигляді електронних таблиць або версій документів, що передаються планувальниками між собою. Унаслідок цього процес планування є трудомістким, повільним і схильним до помилок.

Наявність обмежених ресурсів – машин, пресформ, часу – у поєднанні з дедлайнами, великою кількістю замовлень та різною складністю переналадок дозволяє формалізувати задачу у вигляді модифікованої задачі розкладу типу job-shop scheduling з переналадками, що й обумовлює складність її розв’язання. На відміну від класичної задачі з фіксованим маршрутом кожної роботи, у даному випадку замовлення може бути виконане на будь-якій із сумісних машин. Проте вибір конкретної машини, послідовності замовлень та часу запуску значно впливає на загальний результат: чи буде замовлення виконане вчасно, скільки буде витрачено часу на переналадку, чи буде перевищено виробничу потужність обладнання.

Постановка задачі в рамках дослідження передбачає побудову такого розкладу, який забезпечить виконання максимального числа замовлень у межах встановлених дедлайнів із мінімальними витратами на переналадку, з урахуванням реальних обмежень щодо сумісності машин і пресформ, тривалості

переналадки при зміні кольору або формату, а також з урахуванням кількості гнізд у пресформах і продуктивності обладнання.

Таким чином, об’єкт дослідження – виробниче планування – моделюється як комбінована задача розподілу робіт на обмеженому наборі машин із додатковими умовами:

- варіативність сумісності обладнання,
- обмеження на графік доступності ресурсу (машини, форми),
- змінні витрати часу на підготовчі операції (зміна кольору, зміна форми),
- дотримання часових обмежень (дедлайни),
- мінімізація непродуктивного часу.

Підсумовуючи, постановка задачі полягає у розробці системи, яка автоматизує процес побудови виробничого розкладу із використанням евристичного підходу до розв’язання задачі типу job-shop scheduling, що адаптована до умов підприємства, з подальшим виведенням результатів у зручному для користувача вигляді.

1.3 Методи планування виробництва: огляд жадібних алгоритмів і job-shop scheduling

1.3.1 Жадібні алгоритми

Жадібні алгоритми – це клас методів, що на кожному кроці здійснюють локально оптимальний вибір з метою наблизитися до глобально оптимального розв’язку задачі [7]. Іншими словами, алгоритм завжди обирає той крок, який найкращий на даний момент, не озираючись на можливі наслідки цього вибору в майбутніх кроках [6]. Такий підхід характеризується тим, що рішення приймаються «жадібно», тобто береться найкраща поточна альтернатива і алгоритм не повертається назад для перегляду попередніх рішень [5]. Це означає, що жадібний алгоритм має лише один шанс наблизитися до оптимуму –

обираючи локально найкраще рішення, він ніколи не змінює вже прийнятих рішень у подальшому перебігу алгоритму [5].

Принципова вимога для успішності жадібного підходу – виконання певних властивостей оптимальності. Зокрема, задача повинна мати властивість жадібного вибору (Greedy Choice Property) та оптимальну підструктуру (Optimal Substructure) [6]. Властивість жадібного вибору означає, що оптимальне (глобально найкраще) рішення можна отримати шляхом послідовного вибору оптимальних рішень на локальних кроках [6]. Властивість оптимальної підструктури означає, що оптимальне розв'язання всієї задачі містить у собі оптимальні розв'язання її підзадач [6]. Якщо ці умови виконуються, жадібний алгоритм має шанс знайти глобально оптимальний розв'язок, будуючи його крок за кроком. Якщо ж ні – як, наприклад, у задачі комівояжера чи задачі про рюкзак 0/1 – тоді чисто жадібна стратегія не гарантує оптимуму [6]. Відомо, що задача комівояжера та класична задача 0/1 про рюкзак не задовольняють зазначених властивостей, тому жадібні алгоритми їх не розв'язують оптимально [6]. Натомість багато інших задач мають потрібні властивості і успішно вирішуються жадібними підходами.

Існує ряд відомих комбінаторних задач, для яких жадібні алгоритми дають оптимальний або близький до оптимального результат. Серед класичних прикладів – задача найкоротших шляхів у графі (алгоритм Дейкстри), знаходження мінімального остового дерева (алгоритми Прима та Крускала), побудова оптимального префіксного коду (алгоритм Хаффмана) та дробова задача про рюкзак [7]. В цих випадках жадібна стратегія (наприклад, вибір найменшої вершини, що не оброблена; вибір найменшого ребра, що не утворює цикл; вибір символу з найменшою частотою тощо) приводить до глобально оптимального розв'язку. Інші задачі, які часто розв'язують жадібно, включають розкладання інтервалів (задача планування відрізків) та прості випадки календарного планування, де потрібно вибрати підмножину заходів або завдань з оптимальним критерієм (наприклад, задача вибору активностей з максимальною кількістю несуперечливих інтервалів) тощо. Характерно, що у

всіх цих задачах рішення можна будувати крок за кроком, додаючи найкращий локальний елемент, і такий підхід веде до оптимуму.

Головна перевага жадібних алгоритмів – їхня концептуальна та реалізаційна простота. Часто досить легко придумати жадібну стратегію для конкретної проблеми, іноді навіть кілька варіантів стратегій [5]. Жадібні алгоритми зазвичай реалізуються за допомогою сортування або структур даних типу черги пріоритетів для вибору наступного оптимального кроку, отже вони можуть бути досить ефективними за часом (часто робота алгоритму зводиться до одного сортування $O(n \log n)$ та лінійного проходу) [7]. Аналіз часу виконання жадібного алгоритму, як правило, значно простіший, ніж для більш складних підходів (динамічного програмування, пошуку в просторі станів тощо) [5]. Однак, суттєвим недоліком є відсутність гарантії оптимальності у загальному випадку – неправильний вибір на ранньому кроці вже не можна відкоригувати згодом.

Найскладніше при розробці жадібного алгоритму – довести його коректність, тобто те, що локально оптимальні вибори дають глобально оптимальне рішення [5]. Доведення коректності часто є нетривіальним і базується на обмінних аргументах: доводиться, що будь-яке оптимальне рішення можна крок за кроком перебудувати (обмінюючи елементи) в рішення, отримане жадібним алгоритмом, без погіршення якості [2, с. 6-7]. У літературі такі доведення відомі як обмінний аргумент (exchange argument) і вимагають творчого підходу від дослідника [5, 2, с.5]. Якщо жадібний алгоритм пройшов таку перевірку, він стає дуже привабливим розв’язком завдання завдяки простоті та швидкодії.

1.3.2 Задача календарного планування виробничих робіт

Job-Shop Scheduling (JSSP, задача календарного планування виробничих робіт) – це класична комбінаторна оптимізаційна задача, в якій потрібно

запланувати виконання набору робіт на кількох ресурсах (машинах) з урахуванням порядку операцій. Кожна робота складається з послідовності операцій, які необхідно виконати у заданому порядку на визначених машинах [9]. Кожна машина може виконувати лише одну операцію одночасно, і кожна операція не може бути перервана після початку (недопустиме переривання або перемикання контексту) [9]. Таким чином, потрібно призначити всім операціям час початку на відповідних машинах, щоб дотриматися всіх технологічних пріоритетів (послідовностей операцій у рамках кожної роботи) і не допустити конфліктів за ресурси (одна машина не обслуговує дві роботи водночас) [9].

Типовою метою є мінімізація загальної тривалості виконання всіх робіт, тобто мінімізація makespan – часу від початку першої операції до завершення останньої операції в розкладі [9]. Формально кажучи, маємо n робіт і m машин; кожна i-та робота J_i визначена упорядкованим набором операцій $(O_{i1}, O_{i2}, \dots, O_{ik})$, де кожна операція O_{ij} повинна виконуватися на певній машині M_p і має відомий час обробки t_{ij} на цій машині. Потрібно знайти початкові часи виконання всіх операцій, щоб виконати всі роботи з врахуванням зазначених обмежень і оптимізувати критерій (як правило, мінімізувати час завершення всіх робіт) [9].

Задача календарного планування типу Job-Shop належить до класу надзвичайно складних комбінаторних задач. Доведено, що загальний випадок JSSP є NP-складною задачею [8]. Це означає, що задача належить до таких, розв’язок яких можна перевірити за поліноміальний час, але наразі не існує жодного алгоритму, здатного гарантовано знайти оптимальне рішення за поліноміальний час у загальному випадку. Інакше кажучи, складність розв’язання задачі експоненційно зростає з розширенням розмірності вхідних даних. Існує фундаментальне припущення, що клас задач P (які розв’язуються за поліноміальний час) не збігається з класом NP (де результат можна перевірити за поліноміальний час), тобто $P \neq NP$, і це є однією з центральних гіпотез теорії обчислювальної складності [3]. Це означає, що для задач типу JSSP не існує

ефективного алгоритму, здатного знайти точний розв’язок швидко для довільного обсягу даних, якщо тільки не буде доведено зворотне. Практично це проявляється в тому, що навіть для відносно малих розмірів (скажімо, 10 робіт на 10 машинах) повний перебір або прості методи стають непридатними – час пошуку оптимуму може бути астрономічним [8]. Задача JSSP вважається однією з найскладніших серед типових задач дискретного планування і часто слугує тестовим полігоном для нових підходів в оптимізації. Вона має велике практичне значення: в контексті виробництва оптимальний розклад на верстатах дозволяє суттєво скоротити час циклу та витрати, підвищити продуктивність і гнучкість виробництва [8]. Окрім виробничих систем, варіанти цієї задачі моделюють планування в обчислювальних кластерах, графіки транспортних завдань, логістичні операції тощо – всюди, де обмежені ресурси треба ефективно розподілити між послідовностями операцій. Формальні властивості задачі, такі як обчислювальна складність та NP-повнота, розглядаються в межах дискретної математики як одного з ключових напрямів теорії алгоритмів і комбінаторної оптимізації [12].

За 60+ років досліджень розроблено надзвичайно багато методів розв’язання задачі JSSP [8]. Умовно їх поділяють на точні методи та наближені (евристичні) методи. До точних належать алгоритми, що гарантують знайти оптимальний розв’язок оптимальний розв’язок (або встановити його відсутність) за кінцеве число кроків – наприклад, методи повного перебору з розумним обрізанням пошуку. Класичним прикладом є метод гілок і меж (branch and bound), застосований до JSSP у роботах 90-х років: такий алгоритм перевіряє часткові розклади, відсікаючи гілки, що не можуть привести до оптимального рішення, за допомогою обчислення нижніх меж [8]. Хоча вдосконалені алгоритми гілок і меж здатні розв’язати деякі нетривіальні випадки (наприклад, розклад для 10 робіт і 10 верстатів з відомого набору тесті було оптимально розв’язано після років досліджень), загалом точні методи надто повільні для великих випадків [8]. Альтернативою є формулювання JSSP як задачі цілочислового лінійного програмування або як проблеми логічного обмеженого

пошуку (Constraint Programming) і розв'язання її загальними солверами. Такі підходи (MIP-моделі, CP-моделі) можуть знайти оптимум для малих та середніх задач, але при збільшенні розмірності швидко вичерпують обчислювальні ресурси. Отже, у практично значущих випадках (наприклад, реальні виробництва з десятками верстатів і сотнями операцій) доводиться вдаватися до наближених методів – евристик та метаевристик, що знаходять гарні розв'язки за помірний час, хоча й не гарантують абсолютний оптимум [4].

Евристичні підходи є домінуючими для розв'язання великих екземплярів JSSP, і за довгі роки розроблено багато потужних методик цього класу. Прості жадібні евристики (на кшталт правил диспетчеризації) можуть швидко побудувати допустимий розклад: прикладом є правила list scheduling, коли операції впорядковуються за певним пріоритетом (найкоротша операція, найраніший дедлайн тощо) і поступово розміщуються на машинах. Однак такі прості підходи зазвичай далекі від оптимальних. Набагато кращі результати дають метаевристичні алгоритми – складніші пошукові стратегії, що реалізують багаторазове поліпшення рішення на основі локальних перестановок у розкладі. Дослідження показали, що найкращі на сьогодні результати для JSSP забезпечують саме методи на кшталт ітеративного локального пошуку та різноманітні метаевристики [8]. Зокрема, надзвичайно поширені й ефективні такі підходи, як генетичні алгоритми, імітація відпалу та пошук з табу [8]. Ці методи поступово вдосконалюють розклад, досліджуючи простір можливих перестановок операцій: наприклад, обмінюючи місцями дві операції на машині, зрушуючи блоки операцій, змінюючи призначення операції на іншу машину (в гнучких job-shop) тощо. Такі локальні перебудови розкладу називають сусідніми рішеннями (neighborhood moves), і сукупність правил для переходу між сусідніми рішеннями визначає простір пошуку для локальної оптимізації [1].

Евристика зазвичай починається з деякого початкового розкладу (можливо, побудованого жадібно) і далі намагається покращити його шляхом послідовних локальних змін. Для запобігання застряганню в нескінченних циклах або локальних мінімумах метаевристики додають різні "втечі" –

випадкові збурення (як в імітації відпалу чи в перестрибуванні між декількома рішеннями в пошуку з табу) або комбінування декількох розкладів (як у генетичних алгоритмах). Практичний ефект таких методів дуже високий: хоча вони не гарантують оптимум, на тестових задачах вони часто знаходять рішення з відхиленням від оптимального менше ніж на декілька відсотків, а для великих промислових задач – значно покращують розклад у порівнянні з ручним або жадібним плануванням. Дійсно, численні дослідження демонструють, що сучасні евристичні алгоритми можуть знайти майже оптимальні розклади там, де точні алгоритми безсилі [8] [1].

Таким чином, роль евристик і методів локальної оптимізації в розв'язанні задачі Job-Shop Scheduling є вирішальною: вони забезпечують знаходження якісних розв'язків у прийнятний час, роблячи можливим застосування календарного планування в реальних виробничих і сервісних системах, попри обчислювальну складність цієї задачі.

1.4 Обґрунтування вибору підходу до побудови виробничого розкладу

Розробка ефективного розкладу для процесу виготовлення пластикових кришок потребує врахування низки практичних обмежень: обмежена кількість пресформ і машин, суворі дедлайни поставок, технологічні витрати на переналадку, а також взаємозалежність замовлень за кольором, форматом і вагою виробу. Ці особливості формують задачу, яка належить до класу складних задач дискретної оптимізації, і наближається до задачі типу job-shop scheduling, що є NP-складною. У таких умовах використання точних методів є обчислювально недоцільним навіть для задач середнього розміру.

У зв'язку з цим було обрано евристичний підхід, який дозволяє сформувати практично прийнятне рішення за обмежений час. Як базове рішення використано жадібний алгоритм, що послідовно призначає замовлення на машини, вибираючи на кожному кроці найкращий доступний варіант відповідно

до критерію найближчого дедлайну. Цей підхід дозволяє забезпечити просту реалізацію, високу швидкодію та створення розкладу, який задовольняє ключову умову – своєчасне виконання максимальної кількості замовлень.

Жадібна стратегія також враховує технічну сумісність обладнання, кількість гнізд у пресформах, кратність виробничих партій та обмежений час на переналадку. У базовій версії ці переходи враховуються як фіксовані витрати, однак вони ще не диференційовані залежно від комбінації кольорів або специфіки пресформ. Унаслідок цього загальні витрати на переналадку залишаються суттєвими, що стимулює пошук вдосконаленого методу.

Подальший розвиток моделі реалізовано шляхом переходу до модифікованої версії job-shop scheduling, що дозволяє точніше враховувати витрати на переналадку. Зокрема, використано матрицю переходів між кольорами, в якій час зміни визначається на основі реальних виробничих даних. У моделі також реалізовано локальну оптимізацію розкладу: сусідні завдання на одній машині можуть мінятися місцями, якщо це знижує сумарні витрати часу і не порушує дедлайнів. Додаткове впорядкування завдань за кольорами дозволяє зменшити кількість очищень та механічних переналаштувань.

Об'єднання базового жадібного алгоритму з модифікованими евристиками забезпечує компроміс між обчислювальною ефективністю та якістю планування. Такий підхід є обґрунтованим як з точки зору теоретичної складності задачі, так і з огляду на практичні потреби підприємства, де розклад має формуватись автоматизовано, регулярно та без участі людини в операційних деталях. Це дозволяє знизити кількість помилок, уникнути простоїв і забезпечити більш стабільний виробничий цикл.

Висновки до розділу

У першому розділі було проведено комплексний аналіз виробничого процесу виготовлення пластикових кришок на прикладі підприємства ТОВ

«Ретал Україна». Процес характеризується багатофакторністю, наявністю обмежень щодо сумісності обладнання, чутливістю до технологічних витрат при переналадці, а також значною варіативністю клієнтських замовлень за кількістю, строками виконання та параметрами продукції.

Проаналізовані особливості виробництва – від жорстких вимог до мінімальних партій і тривалості переходів до впливу людського фактора при ручному плануванні – обґрунтовують необхідність автоматизації процесу розкладання виробничих завдань. Саме ці фактори визначили вибір теми дослідження, в межах якого задача календарного планування була формалізована як варіант класичної задачі Job-Shop Scheduling із додатковими обмеженнями на переналадку.

У теоретичному огляді наведено сучасні підходи до розв’язання задач подібного типу: від простих жадібних стратегій до модифікованих евристичних методів, орієнтованих на практичну ефективність. Визначено ключові переваги та обмеження обох класів алгоритмів, що надалі стало основою для вибору комбінованого підходу до реалізації автоматизованої системи планування на підприємстві.

СПЕЦІАЛЬНИЙ РОЗДІЛ

2.1 Проєктування структури бази даних для виробничого планування

Для забезпечення ефективного управління виробничими ресурсами у процесі виготовлення пластикових кришок було спроєктовано реляційну базу даних, яка відображає ключові сутності предметної області та логічні зв'язки між ними. В якості системи управління базами даних обрано MySQL – популярну, продуктивну та безкоштовну СУБД, яка забезпечує швидкий доступ до даних, підтримку складних запитів, транзакцій і зовнішніх ключів. Цей вибір обґрунтований доступністю, широкою підтримкою бібліотек Python та здатністю ефективно працювати з відносно великим обсягом транзакційних даних у виробничих системах. Загальна структура бази даних представлена у вигляді ER-діаграми на рис. 2.1.

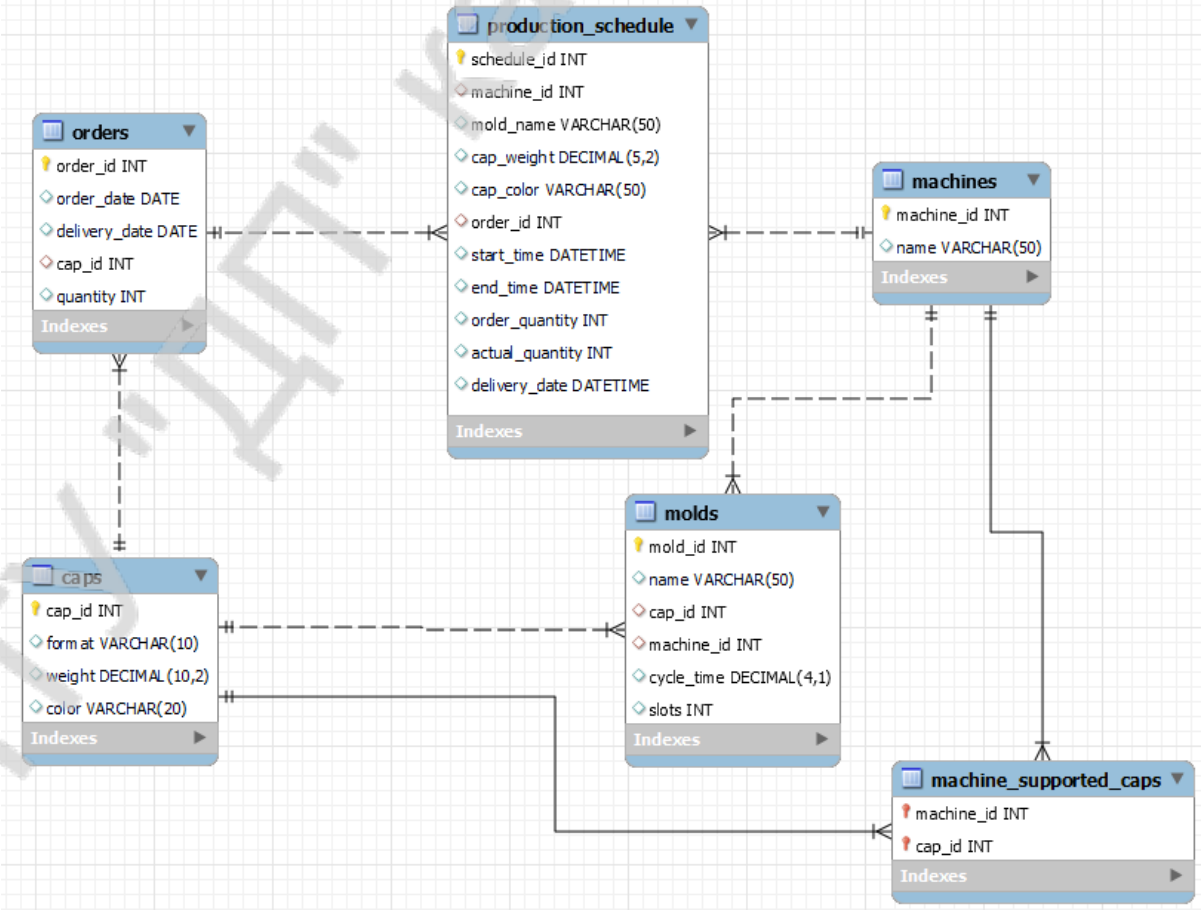

Рисунок. 2.1 – ER-діаграма структури бази даних для виробничого планування.

Структура бази даних побудована таким чином, щоб підтримувати автоматизоване формування виробничого розкладу з урахуванням характеристик замовлень, обладнання, конфігурацій машин і технологічних обмежень. Всі таблиці мають чітко визначені первинні ключі, що забезпечують унікальність записів, а також зовнішні ключі, які гарантують коректність і узгодженість між сутностями.

База даних містить довідникові, операційні та результатні таблиці. Нижче подано їх детальний опис.

Таблиця caps є довідником, що містить ключові характеристики кришок: їхній формат (наприклад, 1881 або 38 мм), масу в грамах та колір. Наприклад, формати кришок варіюються від 1,85 до 6,1 грамів, залежно від призначення (звичайна або з ручкою для великих пляшок). Таблиця caps служить базовою для зв'язку з іншими таблицями, зокрема orders, molds та machine_supported_caps. Вміст таблиці caps наведено на рисунку 2.2.

cap_id	format	weight	color
2	1881	2.05	блакитний
11	1881	2.50	блакитний
20	1881	1.85	блакитний
29	1881	1.95	блакитний
38	1881	2.35	блакитний
47	1810	2.60	блакитний
56	1810	2.95	блакитний
65	1881-Sacmi	2.05	блакитний
74	38mm	2.75	блакитний
83	38mm	3.35	блакитний
92	38mm	4.30	блакитний
101	48mm	4.50	блакитний
110	48mm	6.10	блакитний
1	1881	2.05	білий
10	1881	2.50	білий
19	1881	1.85	білий
28	1881	1.95	білий
37	1881	2.35	білий

Рисунок 2.2 – Вміст таблиці caps.

Таблиця machines містить інформацію про виробниче обладнання. Кожна машина має унікальний ідентифікатор містить інформацію про виробниче обладнання. Кожна машина має унікальний ідентифікатор та назву. У реальному виробництві кожна машина має своє маркування, наприклад, «МК-1» (в системі

– Malpha) чи «МК-12» (MIota). Список наявного обладнання з ідентифікаторами представлено на рисунку 2.3.

machine_id	name
1	MAlpha
2	MBeta
3	MGamma
4	MDelta
5	MEcho
6	MFoxtrot
7	MGolf
8	MHotel
9	MIota
NULL	NULL

Рисунок 2.3 – Вміст таблиці machines.

Таблиця machine_supported_caps реалізує зв’язок типу «багато-до-багатьох» між машинами та кришками. Наприклад, формат 2,95 г підтримується більшістю машин, тоді як формат 2,05 г Sacmi – лише однією. Завдяки цій таблиці забезпечується точне моделювання можливих комбінацій машин і кришок. Нижче наведено приклад реалізації цього зв’язку (рис. 2.4).

machine_id	cap_id
3	1
5	1
6	1
7	1
8	1
9	1
3	2
5	2
6	2
7	2
8	2
9	2
3	3
5	3
6	3
7	3

Рисунок 2.4 – Вміст таблиці machine_supported_caps.

Таблиця molds містить відомості про пресформи (ПФ), зокрема їх назви, кількість гнізд, час одного виробничого циклу, пов’язану машину та кришку, для якої призначена ця форма. Наприклад, ПФ-15 має 64 гнізда та час циклу 5,5 секунди. Як вже було зазначено у першому розділі, кожна пресформа унікальна

та не може одночасно бути встановлена на декількох машинах. Типові значення часу циклу й кількості гнізд у пресформах проілюстрована на рис. 2.5.

mold_id	name	cap_id	machine_id	cycle_time	slots
1	PF-11	37	5	6.3	96
2	PF-11	37	3	5.8	96
3	PF-11	37	6	5.8	96
4	PF-11	37	8	5.8	96
5	PF-15	55	2	6.3	64
6	PF-15	55	4	6.3	64
7	PF-15	55	7	5.5	64
8	PF-15	55	5	6.3	64
9	PF-17	10	5	5.6	96
10	PF-17	10	3	4.9	96
11	PF-17	10	6	4.9	96
12	PF-17	10	8	4.9	96
13	PF-18	10	5	5.6	96
14	PF-18	10	3	4.9	96
15	PF-18	10	6	4.9	96
16	PF-18	10	8	4.9	96
17	PF-19	109	2	7.4	24
18	PF-19	109	4	7.4	24

Рисунок 2.5 – Вміст таблиці molds.

Таблиця orders зберігає дані про замовлення: дату оформлення, дату доставки, кількість, а також cap_id як зовнішній ключ до таблиці caps. Замовлення генеруються у Python (у кількості 100 шт.) для моделювання плану на місяць. Вони охоплюють основні типи кришок, але реально виробництво має значно ширшу номенклатуру. Дані про замовлення та терміни їх виконання можна побачити на рис. 2.6.

order_id	order_date	delivery_date	cap_id	quantity
1	2025-05-23	2025-06-25	109	104660
2	2025-05-22	2025-06-14	92	346700
3	2025-05-24	2025-06-05	51	535880
4	2025-05-24	2025-06-04	75	401950
5	2025-05-25	2025-06-04	60	592360
6	2025-05-26	2025-06-30	61	605410
7	2025-05-29	2025-06-06	43	641350
8	2025-05-25	2025-06-05	34	590740
9	2025-05-27	2025-06-18	73	436700
10	2025-05-23	2025-06-09	45	661070
11	2025-05-23	2025-06-18	5	697220
12	2025-05-24	2025-06-23	99	319230
13	2025-05-22	2025-06-04	115	119990
14	2025-05-23	2025-06-24	81	334380
15	2025-05-23	2025-06-04	22	743640
16	2025-05-28	2025-06-16	6	682340
17	2025-05-29	2025-06-09	2	703350
18	2025-05-28	2025-06-25	3	586730
19	2025-05-28	2025-06-16	108	114670

Рисунок 2.6 – Вміст таблиці orders.

Таблиця production_schedule є результатною – вона зберігає сформований алгоритмом розклад виробництва. Вона агрегує інформацію з усіх основних таблиць: вказує машину, пресформу, кришку (вагу і колір), час початку й завершення, замовлення та відповідну кількість одиниць продукції. Поля order_quantity та actual_quantity показують, наскільки реальна партія могла відрізнятись від початкової через округлення на кратність гнізд. Готовий план виробництва (після виконання алгоритму) з узгодженими параметрами показано на рис. 2.7.

schedule_id	machine_id	mold_name	cap_weight	cap_color	order_id	start_time	end_time	order_quantity	actual_quantity	delivery_date
1	2	PF-19	6.10	червоний	73	2025-05-26 20:45:31	2025-05-27 08:08:25	132870	132888	2025-06-04 00:00:00
2	2	PF-19	6.10	прозорий	13	2025-05-24 15:48:39	2025-05-25 02:05:19	119990	120000	2025-06-04 00:00:00
3	2	PF-19	6.10	жовтий	67	2025-05-22 08:00:00	2025-05-22 19:39:03	136020	136032	2025-06-01 00:00:00
4	2	PF-30	4.30	жовтий	94	2025-05-22 22:09:03	2025-05-24 13:00:39	373050	373056	2025-06-07 00:00:00
5	2	PF-30	4.30	прозорий	23	2025-05-25 04:35:19	2025-05-26 17:33:31	354910	354912	2025-06-13 00:00:00
6	2	PF-30	4.30	блакитний	2	2025-06-01 07:33:10	2025-06-02 19:40:04	346700	346704	2025-06-14 00:00:00
7	2	PF-30	4.30	синій	46	2025-05-29 03:14:52	2025-05-30 14:55:58	342560	342576	2025-06-12 00:00:00
8	2	PF-30	4.30	синій	99	2025-05-30 14:55:58	2025-06-01 07:03:10	385130	385152	2025-06-21 00:00:00
9	2	PF-30	4.30	зелений	40	2025-06-02 20:16:04	2025-06-04 03:39:01	301260	301272	2025-06-14 00:00:00
10	2	PF-30	4.30	червоний	36	2025-05-27 10:38:25	2025-05-29 01:50:52	376380	376392	2025-06-22 00:00:00
11	2	PF-30	4.30	чорний	51	2025-06-04 05:27:01	2025-06-06 02:17:43	430510	430512	2025-06-21 00:00:00
12	2	PF-30	4.30	чорний	12	2025-06-06 02:17:43	2025-06-07 11:33:01	319230	319248	2025-06-23 00:00:00
13	2	PF-30	4.30	золотий	88	2025-06-09 06:12:28	2025-06-10 19:20:52	356530	356544	2025-06-24 00:00:00
14	2	PF-30	4.30	білий	41	2025-06-07 14:03:01	2025-06-09 05:24:28	377830	377832	2025-06-22 00:00:00
15	7	PF-20	4.50	білий	83	2025-05-22 08:00:00	2025-05-22 17:00:37	144150	144168	2025-06-01 00:00:00
16	7	PF-25	2.60	золотий	48	2025-05-22 20:18:37	2025-05-23 08:50:28	601450	601472	2025-06-02 00:00:00
17	7	PF-15	2.95	золотий	82	2025-05-23 11:20:28	2025-05-24 04:30:42	719280	719296	2025-06-11 00:00:00
18	7	PF-15	2.95	золотий	34	2025-05-24 04:30:42	2025-05-24 20:34:51	673110	673152	2025-06-12 00:00:00
19	7	PF-25	2.60	червоний	3	2025-05-24 23:28:51	2025-05-25 10:38:46	535880	535936	2025-06-05 00:00:00
20	7	PF-20	4.50	червоний	57	2025-05-25 13:08:46	2025-05-25 21:03:26	126560	126576	2025-06-16 00:00:00
21	7	PF-20	4.50	прозорий	80	2025-05-25 21:45:26	2025-05-26 04:00:39	100050	100056	2025-06-14 00:00:00
22	7	PF-20	4.50	прозорий	58	2025-05-26 04:00:39	2025-05-26 10:57:42	111210	111216	2025-06-23 00:00:00

Рисунок 2.7 – Вміст таблиці production_schedule (після виконання коду).

Загальна структура даних була змодельована на основі фактичної структури виробничого процесу. Наприклад, у реальному середовищі формати, кольори та обмеження на переходи між ними мають важливе значення через технічні характеристики машин і пресформ. Це враховано в моделі бази даних та забезпечує її придатність до реального використання у виробничих умовах.

Проєктування бази даних відбувалося із урахуванням вимог до цілісності даних, підтримки гнучкої модифікації конфігурації машинного парку, а також забезпечення швидкого доступу до інформації для алгоритмів планування.

Обрана структура дозволяє ефективно моделювати виробничу логіку та забезпечує надійне підґрунтя для реалізації алгоритмів формування розкладу, що розглядатимуться в наступних підрозділах.

2.2 Реалізація жадібного алгоритму планування з урахуванням обмежень

У межах дипломної роботи було реалізовано базову версію алгоритму формування виробничого розкладу на основі жадібної стратегії, що забезпечує автоматизацію процесу розподілу виробничих ресурсів у системі виготовлення пластикових кришок. Реалізація алгоритму здійснена за допомогою мови програмування Python та бібліотеки mysql-connector-python, яка забезпечує підключення і взаємодію з базою даних MySQL. Об'єктом розрахунків виступає розклад виконання замовлень, що мають обмежені строки постачання та вимагають обліку характеристик кришок, пресформ і доступності машин.

Програма починає з отримання вихідних даних із бази – таблиць orders, caps, molds та machines, після чого здійснюється попередня обробка інформації. Зокрема, для кожного замовлення визначається строк його завершення – дедлайн, який встановлюється як 08:00 за добу до дати доставки. Усі замовлення сортуються у порядку наближення дедлайну, що відповідає природній логіці пріоритетності їх виконання.

Основна ідея реалізованого алгоритму полягає в тому, щоб на кожному кроці вибирати для поточного замовлення найкращу (тобто найшвидшу) доступну пресформу з урахуванням сумісності з потрібною кришкою. Для цього здійснюється перевірка відповідності формату та ваги кришки замовлення тим параметрам, які підтримує конкретна пресформа. Якщо така відповідність наявна, далі обчислюється тривалість виготовлення партії продукції, виходячи з циклового часу однієї операції та кількості гнізд у пресформі. Кількість продукції округлюється до найближчого кратного числу слотів для забезпечення повного завантаження обладнання.

Ураховуються також витрати часу на переналадку у разі зміни кольору кришки (1 година) або зміни пресформи (2 години 30 хвилин). Початок виконання замовлення визначається як пізніша з трьох дат: завершення попереднього завдання на машині, завершення останнього використання

пресформи та дата старту виробничого періоду. Якщо замовлення вкладається у встановлений строк – воно записується у графік, а стани машини та пресформи оновлюються.

Усі результати розрахунків фіксуються у таблиці production_schedule, яка перед записом очищується за допомогою операції TRUNCATE. Дані додаються до таблиці пакетом за допомогою операції executemany, що забезпечує ефективне завантаження розкладу у базу даних.

Окрім побудови самого графіка, реалізовано блок оцінки якості розкладу. Зокрема, розраховується загальний час, витрачений на переходи (переналадки), середній час очікування між початком виробництва й стартом розрахункового періоду, а також відсоток завантаження кожної машини у межах її фактичного робочого вікна. Ці метрики дозволяють оцінити ефективність сформованого графіка та порівнювати результати для різних стратегій планування.

Жадібний алгоритм відзначається швидкістю роботи та відносною простотою реалізації. У контексті даного дослідження він виступає базовим підходом, який демонструє значне скорочення часу на планування порівняно з ручною обробкою. На підприємстві, де тестувався алгоритм, подібні розрахунки раніше виконувались вручну протягом одного-двох тижнів. Автоматизований підхід дозволив зменшити цей час до лічених секунд і суттєво знизити кількість помилок та браку, що виникали внаслідок людського фактора.

Таким чином, реалізований алгоритм виконує функцію автоматизованого планування виробництва на основі об’єктивних критеріїв, із мінімальним втручанням користувача. Надалі ця реалізація була доповнена модифікаціями, які дозволили враховувати додаткові параметри – зокрема, матрицю переходів кольорів, локальні оптимізації черговості замовлень на кожній машині та скорочення витрат на переналадку.

Докладний лістинг реалізованого алгоритму подано в Додатку Б.

2.3 Побудова модифікованої моделі задачі календарного планування виробничих робіт

У межах третього етапу розробки системи планування було реалізовано модифіковану модель job-shop scheduling, адаптовану до особливостей виробництва пластикових кришок, із додатковими механізмами локальної оптимізації. Ця модель відрізняється від класичної жадібної реалізації тим, що враховує витрати на переналадку, структуру обладнання, обмеження за дедлайнами і дозволяє суттєво покращити якість сформованого розкладу без повного перерахунку.

Основним принципом роботи алгоритму є поетапне формування графіка виробництва з урахуванням сумісності пресформ і кришок за форматом та вагою. Для кожного замовлення перебираються допустимі варіанти обладнання, зокрема пресформи, що підтримують необхідні характеристики. Вибір кращого варіанта базується на мінімізації моменту завершення виробництва при умові дотримання дедлайну. Витрати часу на переналадку залежать від зміни кольору виробу та заміни пресформи: ці затрати реалізовано через матрицю переходів кольорів і фіксовану тривалість переналадки пресформи.

Алгоритм реалізовано засобами мови програмування Python із використанням бібліотеки mysql.connector для взаємодії з базою даних MySQL. Вхідні дані (таблиці orders, caps, molds) завантажуються безпосередньо перед обробкою, а результат записується у таблицю production_schedule. Обчислення здійснюються без залучення зовнішніх оптимізаційних бібліотек, що забезпечує гнучкість, простоту адаптації та прозорість логіки.

Після формування базового графіка реалізовано механізм локального вдосконалення, що виконується окремо для кожної машини. Оптимізація здійснюється у два етапи. На першому етапі використано класичний метод локального обміну neighbor-swap: перевіряється доцільність перестановки пари суміжних замовлень з погляду зменшення загальних витрат на переходи.

Перестановка приймається, якщо не порушується жоден дедлайн і покращується обрана метрика. На другому етапі застосовано групування завдань за кольором виробу: усі замовлення на одну машину сортуються за дедлайнами всередині кожного кольору, а кольори впорядковуються за наближеністю найранішого з термінів.

Цей підхід дозволяє суттєво зменшити кількість переналадок, особливо дорогих з точки зору кольорових переходів, зменшити середній час очікування початку виробництва та збалансувати завантаження обладнання. Наприклад, порівняльний аналіз показав, що модифікована модель дозволила скоротити загальні витрати на переналадки з 138 годин до 130.4 години, а середній час очікування зменшився приблизно на 46 хвилин. Рівень використання обладнання також покращився у більшості машин, що свідчить про вищу ефективність розкладу.

Таким чином, запропонована модифікована модель job-shop scheduling з локальною оптимізацією є ефективним і практично доцільним підходом до формування виробничого розкладу. Вона забезпечує не лише виконання замовлень у межах встановлених термінів, а й мінімізує витрати часу на переналадки та простої обладнання, що позитивно впливає на загальну ефективність виробництва.

Реалізацію алгоритму представлено в Додатку В.

2.4 Веб-інтерфейс для відображення розкладу (Flask)

Для забезпечення зручної візуалізації сформованого виробничого розкладу було розроблено веб-інтерфейс на основі мікрофреймворку Flask. Такий підхід дає змогу оперативно переглядати заплановані завдання на виробничих машинах без потреби прямого звернення до бази даних. Функціонал інтерфейсу орієнтований на практичні потреби диспетчерів, планувальників та менеджерів виробництва.

Основу реалізації становить Python-додаток, який підключається до реляційної бази даних MySQL і формує структуроване представлення розкладу у форматі HTML-сторінки. Дані отримуються шляхом виконання SQL-запиту з об’єднанням таблиць production_schedule та machines, що дозволяє відобразити повну інформацію про замовлення з урахуванням відповідного обладнання. Усі записи групуються за машинами, що забезпечує чітку ієрархічну структуру відображення та підвищує читабельність.

Веб-сторінка будується на основі HTML-шаблону, який динамічно наповнюється даними. У рамках розмітки реалізовано таблиці з ключовими характеристиками замовлень, зокрема ідентифікатором, назвою пресформи, вагою кришки, кольором, кількістю продукції, часом початку й завершення виробництва, а також терміном доставки. Важливим елементом є використання колірного кодування рядків таблиці залежно від кольору кришки — це не лише візуально спрощує аналіз, а й дозволяє швидко ідентифікувати однорідні партії продукції.

Інтерфейс було стилізовано без залучення зовнішніх CSS-фреймворків. Фон сторінки витримано в нейтральному білому тоні, заголовки таблиць мають приглушене сіре тло, що не відволікає увагу користувача. Рядки таблиці забарвлюються відповідно до кольору кришки: наприклад, червоний, зелений, синій тощо, з використанням пастельних відтінків. Оформлення забезпечує візуальну легкість сприйняття даних та відповідає принципам UX-дизайну в корпоративних системах.

Розроблений інтерфейс є повністю функціональним, адаптивним до екранів різних пристроїв і не потребує встановлення додаткового програмного забезпечення — для перегляду достатньо будь-якого сучасного браузера. Сторінка автоматично оновлюється при зміні даних у базі, що дозволяє забезпечити актуальність розкладу.

Таким чином, реалізація веб-інтерфейсу суттєво підвищує зручність використання системи планування та дозволяє перейти від текстових таблиць до інтерактивного відображення інформації. Це рішення може бути

масштабованим: у майбутньому інтерфейс можливо розширити функціями редагування, фільтрації, побудови графіків або експорту даних.

Вигляд сторінки представлено в пункті 2.5. Повний код інтерфейсу наведено в Додатку Г.

2.5 Порівняльний аналіз ефективності алгоритмів

Етап порівняльного аналізу є ключовим у верифікації ефективності запропонованої моделі розподілу виробничих ресурсів. У процесі розв'язання задачі було реалізовано два алгоритмічні підходи – базовий жадібний алгоритм та вдосконалена модель, що поєднує елементи job-shop scheduling з локальною оптимізацією черговості. Основною метою цього розділу є визначення переваг і недоліків кожного з підходів, а також виявлення ситуацій, у яких один із методів виявляється переважним.

Базовий жадібний алгоритм формує розклад шляхом послідовного вибору найближчого за строком дедлайну замовлення та розміщення його на першій доступній машині, що підтримує відповідну пресформу. Цей підхід характеризується високою швидкістю обчислення, але має обмежену здатність до оптимізації загальних витрат – зокрема, не враховує порядок кольорових переходів і не виконує жодної повторної перевірки чи локального покращення.

Натомість модифікований job-shop алгоритм, реалізований у межах цієї дипломної роботи, базується на тому ж вихідному розкладі, але доповнений механізмом локального обміну задач на кожній машині. Після початкового призначення завдань виконується перестановка суміжних елементів (метод neighbor-swap), що дозволяє суттєво знизити витрати на переналадку обладнання – зокрема, у випадках частих змін кольору або пресформ.

Одним із основних критерії ефективності є загальна тривалість переналадок, спричинених переходами між завданнями. У разі використання жадібного підходу цей показник становив 138 годин, тоді як після застосування

модифікованого алгоритму значення скоротилось до 130,40 годин, що свідчить про помітне зменшення втрат часу, пов’язаних із технічними паузами.

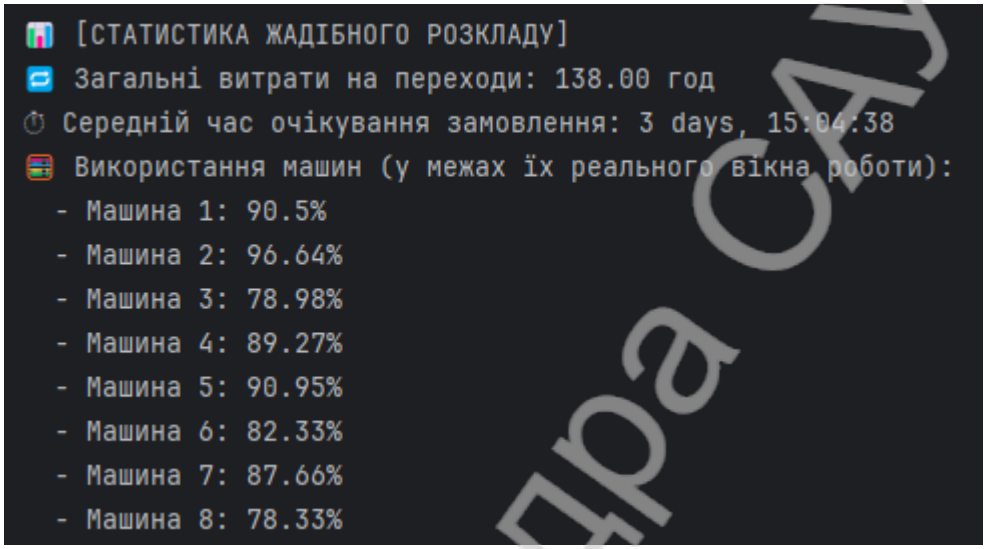

Рисунок 2.8 – Статистика виконання жадібного алгоритму.

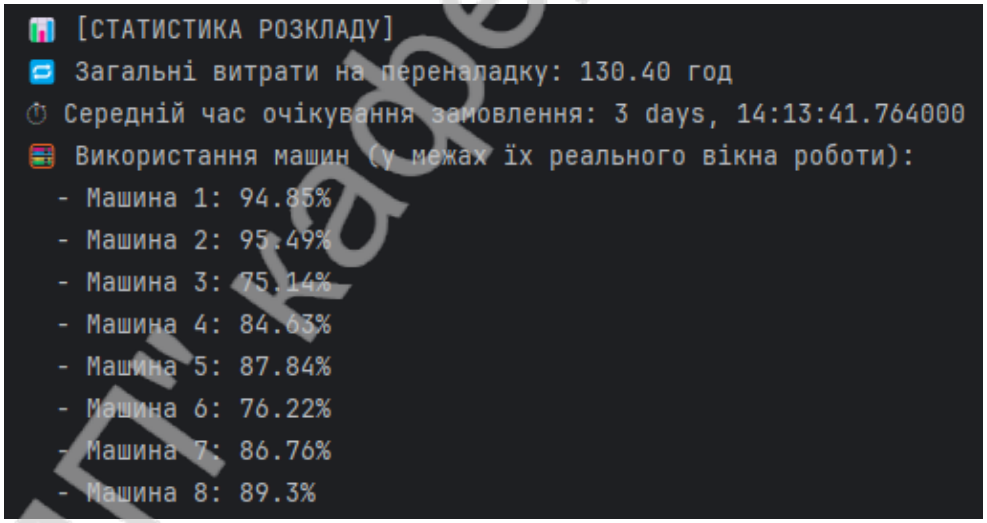

Рисунок 2.9 – Статистика модифікованого job-shop алгоритму.

Ще одним важливим показником є середній час очікування замовлення – тобто затримка між початком розрахункового періоду та моментом старту кожного виробничого циклу. Для жадібного алгоритму середнє очікування становило 3 дні 15 годин 4 хвилини, у той час як для модифікованого – 3 дні 14 години 18 хвилин, тобто майже на 46 хвилин менше. Хоча на перший погляд різниця здається несуттєвою, при масштабуванні системи на більші обсяги замовлень це дозволяє звільнити виробничі потужності раніше.

Для більш детального порівняння було проаналізовано відсоткове використання кожної з восьми машин (тобто, співвідношення часу фактичної

роботи машини до доступного робочого вікна). Результати статистик проілюстровано додатково на рис. 2.10:

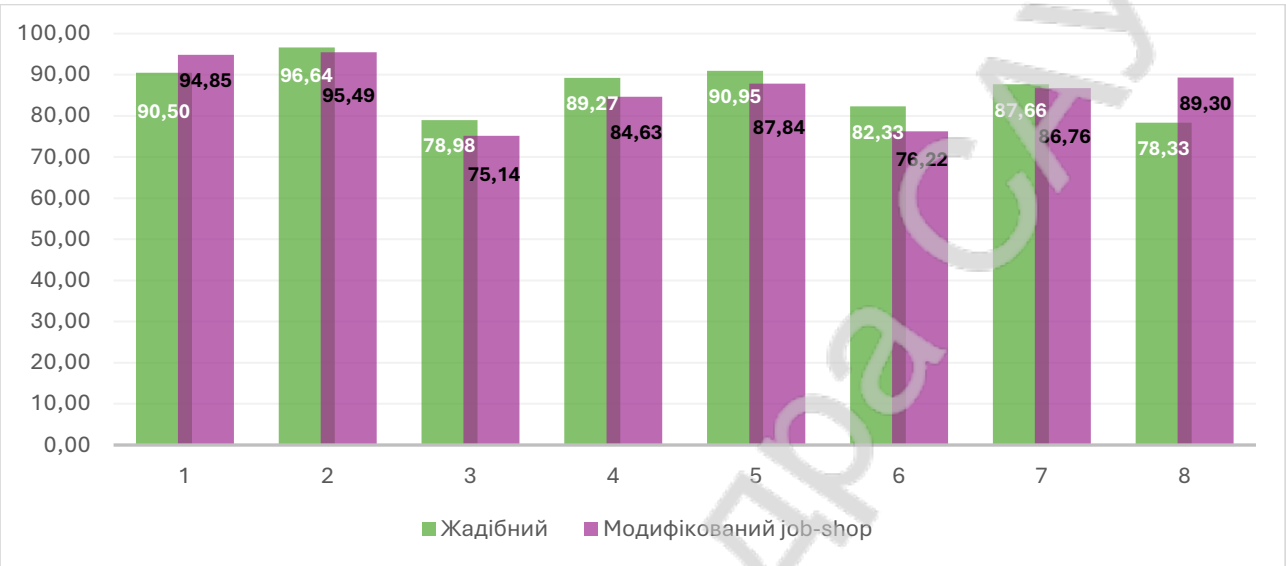

Рисунок 2.10 – Використання машин (%).

Зі статистики видно, що покращення розподілу не завжди веде до однакового результату щодо завантаження машин. Наприклад, у машин 1,2 та 8 використання зросло, тоді як у машин 3, 4, 5 та 6 – знизилося. Це пояснюється змінами в черговості, які дозволили краще збалансувати навантаження у критичних точках графіка.

Для глибшого розуміння поведінки обох алгоритмів проаналізовано конкретні приклади сформованих розкладів для машин MAlpha та MEcho. Наведені фрагменти демонструють, що у жадібному варіанті замовлення одного кольору часто розділені іншими, що зумовлює зайві витрати на очищення машин та заміну матеріалу. Наприклад, на MAlpha червоний, зелений і жовтий кольори чергуються без очевидного патерну, тоді як у модифікованому варіанті досягнуто кластеризації замовлень за кольором, що дозволяє уникнути зайвих переходів.

MAlpha

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
86	PF-21	3.35	червоний	433 090	433 152	2025-05-22 08:00:00	2025-05-22 18:09:07	2025-06-04
25	PF-21	3.35	зелений	429 130	429 216	2025-05-22 19:09:07	2025-05-23 05:12:42	2025-06-05
50	PF-21	3.35	червоний	304 890	304 896	2025-05-23 06:12:42	2025-05-23 13:21:27	2025-06-06
97	PF-21	3.35	жовтий	397 190	397 248	2025-05-23 14:21:27	2025-05-23 23:40:05	2025-06-06
47	PF-21	3.35	синій	322 960	323 040	2025-05-24 00:40:05	2025-05-24 08:14:22	2025-06-09
84	PF-21	3.35	зелений	430 940	430 944	2025-05-24 09:14:22	2025-05-24 19:20:23	2025-06-09
69	PF-21	3.35	білий	332 410	332 448	2025-05-24 20:20:23	2025-05-25 04:07:53	2025-06-12
70	PF-21	3.35	зелений	368 240	368 256	2025-05-25 05:07:53	2025-05-25 13:45:45	2025-06-12
38	PF-21	3.35	синій	358 620	358 656	2025-05-25 14:45:45	2025-05-25 23:10:06	2025-06-20
75	PF-21	3.35	золотий	403 460	403 488	2025-05-26 00:10:06	2025-05-26 09:37:30	2025-06-23
89	PF-21	3.35	синій	371 780	371 808	2025-05-26 10:37:30	2025-05-26 19:20:22	2025-06-24
54	PF-21	3.35	прозорий	339 160	339 168	2025-05-26 20:20:22	2025-05-27 04:17:19	2025-06-25
61	PF-21	3.35	синій	384 010	384 096	2025-05-27 05:17:19	2025-05-27 14:17:27	2025-06-29

Рисунок 2.11 – Розклад для машини MAlpha (жадібний підхід).

MAlpha

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
86	PF-21	3.35	червоний	433 090	433 152	2025-05-22 08:00:00	2025-05-22 18:09:07	2025-06-04
50	PF-21	3.35	червоний	304 890	304 896	2025-05-22 18:09:07	2025-05-23 01:17:52	2025-06-06
25	PF-21	3.35	зелений	429 130	429 216	2025-05-23 01:53:52	2025-05-23 11:57:27	2025-06-05
84	PF-21	3.35	зелений	430 940	430 944	2025-05-23 11:57:27	2025-05-23 22:03:28	2025-06-09
70	PF-21	3.35	зелений	368 240	368 256	2025-05-23 22:03:28	2025-05-24 06:41:20	2025-06-12
97	PF-21	3.35	жовтий	397 190	397 248	2025-05-24 07:35:20	2025-05-24 16:53:58	2025-06-06
47	PF-21	3.35	синій	322 960	323 040	2025-05-24 18:29:58	2025-05-25 02:04:14	2025-06-09
38	PF-21	3.35	синій	358 620	358 656	2025-05-25 02:04:14	2025-05-25 10:28:36	2025-06-20
89	PF-21	3.35	синій	371 780	371 808	2025-05-25 10:28:36	2025-05-25 19:11:27	2025-06-24
61	PF-21	3.35	синій	384 010	384 096	2025-05-25 19:11:27	2025-05-26 04:11:35	2025-06-29
69	PF-21	3.35	білий	332 410	332 448	2025-05-26 05:59:35	2025-05-26 13:47:06	2025-06-12
75	PF-21	3.35	золотий	403 460	403 488	2025-05-26 14:35:06	2025-05-27 00:02:30	2025-06-23
54	PF-21	3.35	прозорий	339 160	339 168	2025-05-27 00:32:30	2025-05-27 08:29:27	2025-06-25

Рисунок 2.12 – Розклад для машини MAlpha (модифікований підхід).

Аналогічну тенденцію можна спостерігати і для машини MEcho – модифікований алгоритм спочатку розміщує всі червоні, потім сині, далі прозорі замовлення, що свідчить про ефективнішу оптимізацію послідовності.

MEcho

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
5	PF-15	2.95	червоний	592 360	592 384	2025-05-22 08:00:00	2025-05-23 00:11:52	2025-06-04
78	PF-11	2.35	синій	743 340	743 424	2025-05-23 03:41:52	2025-05-23 17:15:00	2025-06-05
7	PF-11	2.35	прозорий	641 350	641 376	2025-05-23 18:15:00	2025-05-24 05:56:30	2025-06-06
31	PF-11	2.35	прозорий	710 120	710 208	2025-05-24 05:56:30	2025-05-24 18:53:17	2025-06-06
10	PF-11	2.35	чорний	661 070	661 152	2025-05-24 19:53:17	2025-05-25 07:56:25	2025-06-09
74	PF-11	2.35	білий	678 170	678 240	2025-05-25 08:56:25	2025-05-25 21:18:15	2025-06-14
27	PF-15	2.95	блакитний	556 350	556 352	2025-05-26 00:48:15	2025-05-26 16:01:01	2025-06-27
6	PF-15	2.95	прозорий	605 410	605 440	2025-05-26 17:01:01	2025-05-27 09:34:19	2025-06-30

Рисунок 2.13 – Розклад для машини MEcho (жадібний підхід).

MEcho

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
5	PF-15	2.95	червоний	592 360	592 384	2025-05-22 08:00:00	2025-05-23 00:11:52	2025-06-04
29	PF-11	2.35	червоний	654 470	654 528	2025-05-23 02:41:52	2025-05-23 14:37:46	2025-06-27
78	PF-11	2.35	синій	743 340	743 424	2025-05-23 16:01:46	2025-05-24 05:34:53	2025-06-05
76	PF-17	2.50	синій	692 430	692 448	2025-05-24 08:04:53	2025-05-24 19:18:06	2025-06-23
7	PF-11	2.35	прозорий	641 350	641 376	2025-05-24 23:18:06	2025-05-25 10:59:36	2025-06-06
31	PF-11	2.35	прозорий	710 120	710 208	2025-05-25 10:59:36	2025-05-25 23:56:23	2025-06-06
91	PF-11	2.35	прозорий	749 720	749 760	2025-05-25 23:56:23	2025-05-26 13:36:26	2025-06-29
10	PF-11	2.35	чорний	661 070	661 152	2025-05-26 15:36:26	2025-05-27 03:39:35	2025-06-09
27	PF-15	2.95	блакитний	556 350	556 352	2025-05-27 07:39:35	2025-05-27 22:52:20	2025-06-27

Рисунок 2.14 – Розклад для машини MEcho (модифікований підхід).

У сукупності порівняльний аналіз засвідчив, що застосування комбінованої моделі на основі job-shop scheduling з локальною оптимізацією дозволяє досягти суттєвих покращень як у плані зменшення витрат на переналадку, так і в аспектах розумного використання машинного часу. Незважаючи на те, що повне усунення простоїв та неефективних переходів не завжди можливе внаслідок обмежень виробничої системи, запропонований підхід довів свою перевагу над класичним жадібним методом.

У додатках Д та Е подано повні таблиці виробничих розкладів, що використовувалися для аналізу.

Висновки до розділу

У даному розділі було повністю реалізовано систему автоматизованого планування виробництва пластикових кришок, що включає етапи проєктування бази даних, розробки обчислювальних алгоритмів та створення інтерфейсу для візуалізації розкладу. На основі моделювання предметної області створено реляційну базу даних із дотриманням принципів нормалізації, підтримки цілісності та розширюваності структури. Вміст таблиць відображає реальні характеристики продукції, машин і технічних обмежень виробництва. Саме це забезпечило узгодженість програмної реалізації з умовами практичного застосування.

Використання жадібного алгоритму стало базовим підходом до автоматизованого розподілу ресурсів: він дозволив сформувати графік виробництва з урахуванням дедлайнів і технічних параметрів, скоротивши тривалість планування до лічених секунд. Проте через локальний характер вибору пресформ і ігнорування черговості замовлень, цей підхід спричиняв надлишкові витрати на переналадку.

Для подолання цих обмежень було розроблено вдосконалену модифіковану модель задачі календарного планування, побудовану на

принципах job-shop scheduling. Її особливістю стала наявність етапу локальної оптимізації: алгоритм neighbor-swap та групування замовлень за кольором дозволили суттєво зменшити час на переналадку й оптимізувати послідовність обробки замовлень. Результати підтвердили ефективність підходу: час простоїв скоротився на понад 7 годин, середній час очікування зменшився, а завантаження машин у багатьох випадках покращилося.

Доповнення системи веб-інтерфейсом на основі Flask дозволило інтерактивно переглядати сформований розклад у зручному табличному форматі. Візуалізація за допомогою колірного кодування покращила сприйняття даних і зробила інтерфейс придатним для використання в реальному виробництві. Результати моделювання переконливо демонструють потенціал впровадження такої системи в рамках підприємства для зменшення навантаження на планувальників, скорочення витрат та підвищення точності виробничого графіка.

Загалом, виконані в розділі дослідження і розробки підтверджують, що поєднання алгоритмічних підходів до планування із правильно спроєктованою структурою даних та практично-орієнтовною візуалізацією забезпечує комплексне рішення, здатне відповісти на виклики сучасного ресурсного планування.

ВИСНОВКИ

У кваліфікаційній роботі було розв’язано актуальну прикладну задачу – оптимізацію процесу виробничого планування на прикладі підприємства з виготовлення пластикових кришок. На основі аналізу реальних виробничих умов ТОВ «Ретал Україна» було виявлено ключові фактори, що ускладнюють формування розкладу; обмежена сумісність пресформ і машин, витрат часу на переналадки при зміні формату або кольору, строгі дедлайни, нерівномірність надходження замовлень та вплив людського фактору при ручному плануванні.

Для забезпечення автоматизації процесу було побудовано повноцінну систему планування, що поєднує в собі кілька важливих компонентів:

1. Реляційна база даних, що моделює реальну структуру виробництва, включаючи замовлення, типи кришок, обладнання, пресформи та обмеження на їх сумісність. Таблиці структурувалися з урахуванням цілісності, нормалізації та можливості масштабування.

2. Жадібний алгоритм, реалізований мовою Python, що формує розклад на основі дедлайнів і доступності ресурсів. Його головними перевагами стали швидкість, простота реалізації та придатність для генерації базового варіанту графіка.

3. Модифікований підхід Job-Shop Scheduling, який враховує матрицю переходів між кольорами, тривалість переналадок і виконує локальну оптимізацію послідовності замовлень на кожній машині. Завдяки цьому вдалося скоротити час простоїв, зменшити середній час очікування та підвищити ефективність завантаження обладнання.

4. Веб-інтерфейс на Flask, що забезпечує зручну візуалізацію сформованого графіка, з колірним кодуванням замовлень для спрощення сприйняття, підтримкою адаптивності та відображенням даних у табличному форматі.

Проведено глибокий порівняльний аналіз результатів обох методів. Модифікована версія алгоритму показала кращі результати за основними метриками: час переналадки скоротився на понад 7 годин, середній час очікування – майже на годину, а баланс завантаження машин покращився. Всі ці покращення досягнуто без порушення дедлайнів і без залучення надлишкових обчислювальних ресурсів, що робить модель придатною до застосування в умовах серійного виробництва.

Узагальнюючи, можна стверджувати, що розроблене рішення дозволяє ефективно автоматизувати процес планування, скоротити витрати часу, зменшити кількість помилок та підвищити точність формування виробничих графіків. Система може бути масштабована, розширена новими модулями (наприклад, редагування або валідація замовлень), а також адаптована до інших типів продукції з подібними обмеженнями. Робота підтвердила доцільність використання комбінованого підходу, в якому поєднуються простота базових алгоритмів та ефективність локальних евристичних покращень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Abdelmaguid T. F. A neighborhood search function for flexible job shop scheduling with separable sequence-dependent setup times [Електронний ресурс] // Applied Mathematics and Computation. – 2016. – Vol. 274. – P. 375–391. – Режим доступу: https://www.sciencedirect.com/science/article/abs/pii/S0096300315003732

2. Erickson J. Algorithms [Електронний ресурс] / Jeff Erickson. – Urbana, Illinois: University of Illinois, 2019. – 411 p. – Режим доступу: https://jeffe.cs.illinois.edu/teaching/algorithms/book/04-greedy.pdf

3. Goemans M. X. Lecture 16: Advanced Algorithms [Електронний ресурс] / Michel X. Goemans ; консп. Nicole Immorlica, Mana Taghdiri. – Massachusetts Institute of Technology, 2001. – Режим доступу: https://dspace.mit.edu/bitstream/handle/1721.1/49420/6-854JFall2001/NR/rdonlyres/Electrical-Engineering-and-Computer-Science/6-854JFall2001/E71F81CC-FC34-4A08-A88A-FF983ACC14D0/0/lect11_07.pdf

4. Jain A. S., Meeran S. Deterministic job-shop scheduling: Past, present and future [Електронний ресурс] // European Journal of Operational Research. – 1999. – Vol. 113, No. 2. – P. 390–434. – Режим доступу: https://www.sciencedirect.com/science/article/abs/pii/S0377221798001131

5. Basics of Greedy Algorithms [Електронний ресурс] – Режим доступу: https://www.hackerearth.com/practice/algorithms/greedy/basics-of-greedy-algorithms/tutorial/ (дата звернення: 01.04.2025)

6. DSA Greedy Algorithms [Електронний ресурс] – Режим доступу: https://www.w3schools.com/dsa/dsa_ref_greedy.php (дата звернення: 10.04.2025)

7. Greedy Algorithms [Електронний ресурс] – Режим доступу: https://www.geeksforgeeks.org/greedy-algorithms/ (дата звернення: 21.04.2025)

8. Job shop scheduling [Електронний ресурс] – Режим доступу: https://optimization.cbe.cornell.edu/index.php?title=Job_shop_scheduling (дата звернення: 30.04.2025)

9. The Job Shop Problem [Електронний ресурс] – Режим доступу: https://developers.google.com/optimization/scheduling/job_shop (дата звернення: 04.05.2025)

10. Кваліфікаційна робота бакалавра [Електронний ресурс] : методичні рекомендації для здобувачів ступеня бакалавра освітньо-професійної програми «Системний аналіз» зі спеціальності 124 Системний аналіз / уклад.: Т.А. Желдак, Т.В. Хом'як, А.В. Малієнко; М-во освіти і науки України,Нац. техн. ун-т «Дніпровська політехніка». – Дніпро: НТУ «ДП», 2025. – 32 с.

11. Хабарлак К.С. Самонавчання складних систем [Електронний ресурс]: конспект лекцій для здобувачів ступеня магістра освітньо-професійної програми «Системний аналіз» зі спеціальності 124 Системний аналіз / К.С. Хабарлак, Т.А. Желдак ; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2024. – 112с. Доступ: https://ir.nmu.org.ua/handle/123456789/167923 (дата звернення: 15.05.2024)

12. Дискретна математика: навч. посібник / В.В. Слєсарєв, І.В. Новицький, С.А. Ус. – М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2023. – 183 с. https://ir.nmu.org.ua/handle/123456789/164331

ДОДАТОК А. ВІДОМІСТЬ МАТЕРІАЛІВ КВАЛІФІКАЦІЙНОЇ РОБОТИ

№ з/п	Позначення	Найменування	Кількість аркушів	Примітки
1				
2		Документація		
3				
4	САУ.КР.25.02.ПЗ	Пояснювальна записка	91	Формат А4
5				
6		Демонстраційний матеріал	10	Презентація на CD-R
7				
8		Копія роботи	1	Диск CD-R
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				

					САУ.КР.25.02.ДА.ПЗ.			
Змін.	Аркуш	№ докум.	Підпис	Дата				
Розроб.		Біліма В.К.			**Матеріали кваліфікаційної роботи**	Літ.	Аркуш	Аркушів
К. розд.		Хабарлак К.С.						
Керівн.		Хабарлак К.С.						
Н.контр		Хом’як Т.В.						
Зав. каф.		Желдак Т.А.						

ДОДАТОК Б. ПРОГРАМНИЙ КОД РЕАЛІЗАЦІЇ БАЗОВОГО ЖАДІБНОГО АЛГОРИТМУ ПЛАНУВАННЯ

```
import mysql.connector
from datetime import datetime, timedelta
import math
from collections import defaultdict

# ------------------ 1. Підключення до MySQL ------------------
db_config = {
    'host': 'localhost',
    'user': 'bilima',
    'password': '1111',
    'database': 'caps_diploma'
}

conn = mysql.connector.connect(**db_config)
cursor = conn.cursor(dictionary=True)

# ------------------ 2. Завантаження orders ------------------
cursor.execute("SELECT order_id, order_date, delivery_date, cap_id, quantity FROM orders;")
orders = cursor.fetchall()
if not orders:
    raise Exception("Таблиця orders порожня!")

# ------------------ 3. Завантаження caps ------------------
cursor.execute("SELECT cap_id, format, weight, color FROM caps;")
caps_data = cursor.fetchall()
caps_dict = {c['cap_id']: c for c in caps_data}

# ------------------ 4. Обчислення дедлайнів (delivery_date - 1 день, 08:00) ------------------
for o in orders:
    if isinstance(o['delivery_date'], datetime):
        delivery_dt = o['delivery_date'].replace(hour=8, minute=0, second=0)
```

```
    else:
        d = o['delivery_date']
        delivery_dt = datetime(d.year, d.month, d.day, 8, 0, 0)
    o['deadline'] = delivery_dt - timedelta(days=1)

# Сортуємо замовлення за найближчим дедлайном
orders.sort(key=lambda x: x['deadline'])

# ------------------ 5. Завантаження molds + machines ------------------
query_molds = """
SELECT mld.mold_id, mld.name AS mold_name, mld.machine_id, mld.cap_id AS mold_cap_id,
       mld.cycle_time, mld.slots, mach.name AS machine_name
FROM molds mld
JOIN machines mach ON mld.machine_id = mach.machine_id;
"""
cursor.execute(query_molds)
molds_data = cursor.fetchall()
if not molds_data:
    raise Exception("Таблиця molds порожня!")

# ------------------ 6. Визначення початку виробництва ------------------
earliest_delivery = min(o['delivery_date'] for o in orders)
production_start = datetime(earliest_delivery.year, earliest_delivery.month, 1, 8, 0, 0) -
timedelta(days=10)

# ------------------ 7. Стан машин ------------------
# Для кожної машини зберігаємо, коли вона звільниться, який колір і яка пресформа (назва)
machine_state = {}
machine_ids = set(m['machine_id'] for m in molds_data)
for mid in machine_ids:
    machine_state[mid] = {
        'current_end_time': production_start,
        'current_color': None,
        'current_mold_name': None
    }
```

```
# ------------------ 8. Стан пресформ ------------------
# Для кожної унікальної назви пресформи зберігаємо, коли вона вільна
mold_state = {}
for m in molds_data:
    mold_name = m['mold_name']
    if mold_name not in mold_state:
        mold_state[mold_name] = {
            'current_end_time': production_start
        }


# ------------------ 9. Функція перевірки сумісності (за форматом і вагою) ------------------
def can_produce(mold, order_cap_id):
    mold_cap = caps_dict[mold['mold_cap_id']]  # Кришка, з якою пов'язана пресформа
    order_cap = caps_dict[order_cap_id]
    return (
            mold_cap['format'] == order_cap['format'] and
            abs(mold_cap['weight'] - order_cap['weight']) < 1e-9
    )


# ------------------ 10. Параметри переходів ------------------
color_change_time = timedelta(hours=1)  # Перехід при зміні кольору
mold_change_time = timedelta(hours=2.5)  # Перехід при зміні пресформи

# ------------------ 11. Жадібне планування ------------------
schedule_entries = []
unscheduled_count = 0

for order in orders:
    order_id = order['order_id']
    order_cap_id = order['cap_id']
    order_quantity = order['quantity']
    order_color = caps_dict[order_cap_id]['color']
```

```
deadline = order['deadline']

# Визначаємо кандидатів – пресформи, що можуть виробити потрібну кришку
candidates = [md for md in molds_data if can_produce(md, order_cap_id)]
if not candidates:
    print(f"Замовлення {order_id}: не знайдено пресформи (cap_id={order_cap_id}).")
    unscheduled_count += 1
    continue

best_option = None
best_finish_time = None

for mold in candidates:
    mid = mold['machine_id']
    mold_name = mold['mold_name']

    # Отримуємо стан машини
    machine_end = machine_state[mid]['current_end_time']
    machine_col = machine_state[mid]['current_color']
    machine_mold = machine_state[mid]['current_mold_name']

    # Отримуємо стан пресформи
    mold_end = mold_state[mold_name]['current_end_time']

    # Округлення кількості до кратності slots
    slots = mold['slots']
    actual_qty = math.ceil(order_quantity / slots) * slots

    # Обчислення часу виробництва
    cycles = actual_qty / slots
    production_sec = cycles * float(mold['cycle_time'])
    production_duration = timedelta(seconds=production_sec)

    # Обчислення часу переходу
    transition = timedelta(0)
```

```
        if machine_col and machine_col != order_color:
            transition += color_change_time
        if machine_mold and machine_mold != mold_name:
            transition += mold_change_time

        # Розрахунок старту і завершення виробництва
        start_time = max(machine_end, mold_end, production_start) + transition
        end_time = start_time + production_duration

        if end_time <= deadline:
            if best_option is None or end_time < best_finish_time:
                best_option = (mold, start_time, end_time, actual_qty)
                best_finish_time = end_time

    if best_option:
        chosen_mold, s_time, e_time, real_qty = best_option
        chosen_mold_name = chosen_mold['mold_name']
        chosen_mid = chosen_mold['machine_id']

        # Формування запису для нової таблиці:
        # - Замість mold_id – зберігаємо mold_name
        # - Замість cap_id – зберігаємо cap_weight (із таблиці caps)
        # - Додаємо cap_color (із таблиці caps)
        # - Додаємо delivery_date (із orders)
        cap_weight = caps_dict[order_cap_id]['weight']
        cap_color = caps_dict[order_cap_id]['color']
        # Форматуємо delivery_date як "YYYY-MM-DD"
        delivery_date = order['delivery_date'].strftime("%Y-%m-%d") \
            if isinstance(order['delivery_date'], datetime) else \
            order['delivery_date']

        schedule_entries.append((
            chosen_mid,
            chosen_mold_name,  # mold_name замість mold_id
            cap_weight,  # cap_weight замість cap_id
```

```
            cap_color,  # доданий стовпець з кольором
            order_id,
            s_time.strftime("%Y-%m-%d %H:%M:%S"),
            e_time.strftime("%Y-%m-%d %H:%M:%S"),
            order_quantity,  # Початкова кількість
            real_qty,  # Округлена (реальна) кількість
            delivery_date  # Дата доставки
        ))

        # Оновлення стану машини
        machine_state[chosen_mid]['current_end_time'] = e_time
        machine_state[chosen_mid]['current_color'] = order_color
        machine_state[chosen_mid]['current_mold_name'] = chosen_mold_name

        # Оновлення стану пресформи (одна фізична пресформа не може використовуватись паралельно)
        mold_state[chosen_mold_name]['current_end_time'] = e_time

        print(f"Замовлення {order_id} -> {chosen_mold['machine_name']} (mold={chosen_mold_name}) "
              f"{s_time} - {e_time}, Q={order_quantity}->{real_qty}")
    else:
        print(f"Замовлення {order_id}: не встигаємо до дедлайну {deadline}.")
        unscheduled_count += 1

print(f"\nЗаплановано {len(schedule_entries)} із {len(orders)}. Незаплановано: {unscheduled_count}.")

# ---------- 12. Видалення попередніх записів з production_schedule ----------
cursor.execute("TRUNCATE TABLE production_schedule;")
conn.commit()
print("Попередні записи з production_schedule видалено.")

# ---------- 13. Запис у production_schedule ----------
insert_query = """
```

```
INSERT INTO production_schedule
(machine_id, mold_name, cap_weight, cap_color, order_id, start_time,
end_time, order_quantity, actual_quantity, delivery_date)
VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s);
"""

if schedule_entries:
    cursor.executemany(insert_query, schedule_entries)
    conn.commit()
    print("Розклад записано у production_schedule.")
else:
    print("Немає призначених замовлень.")

# ------------------ 14. Статистика ------------------

# Переводимо графік у структуру по машинах
machine_tasks = defaultdict(list)
for entry in schedule_entries:
    (machine_id, mold_name, cap_weight, cap_color, order_id, start_str, end_str,
     order_qty, actual_qty, delivery_date) = entry
    start = datetime.strptime(start_str, "%Y-%m-%d %H:%M:%S")
    end = datetime.strptime(end_str, "%Y-%m-%d %H:%M:%S")
    machine_tasks[machine_id].append({
        'start': start,
        'end': end,
        'color': cap_color,
        'mold': mold_name
    })

# Обчислення метрик
total_transition_hours = 0.0
total_wait_time = timedelta(0)
total_jobs = 0
utilization = {}
```

```
for mid, tasks in machine_tasks.items():
    tasks.sort(key=lambda x: x['start'])
    active_time = timedelta(0)
    prev_color = None
    prev_mold = None

    for task in tasks:
        duration = task['end'] - task['start']
        active_time += duration
        total_wait_time += task['start'] - production_start
        if prev_color and prev_color != task['color']:
            total_transition_hours += color_change_time.total_seconds() / 3600
        if prev_mold and prev_mold != task['mold']:
            total_transition_hours += mold_change_time.total_seconds() / 3600
        prev_color = task['color']
        prev_mold = task['mold']

    # Вікно роботи конкретної машини
    machine_start = min(t['start'] for t in tasks)
    machine_end = max(t['end'] for t in tasks)
    machine_window = machine_end - machine_start
    utilization[mid] = round(100 * active_time.total_seconds() / machine_window.total_seconds(), 2)
    total_jobs += len(tasks)
avg_wait = total_wait_time / total_jobs if total_jobs > 0 else timedelta(0)

# Вивід статистики
print("\n📊 [СТАТИСТИКА ЖАДІБНОГО РОЗКЛАДУ]")
print(f"🔁 Загальні витрати на переходи: {total_transition_hours:.2f} год")
print(f"⏱️ Середній час очікування замовлення: {avg_wait}")
print("🧮 Використання машин (у межах їх реального вікна роботи):")
for mid in sorted(utilization.keys()):
    print(f"  - Машина {mid}: {utilization[mid]}%")
cursor.close()
conn.close()
```

ДОДАТОК В. ПРОГРАМНИЙ КОД РЕАЛІЗАЦІЇ МОДИФІКОВАНОЇ МОДЕЛІ JOB-SHOP SCHEDULING

```
import mysql.connector
from datetime import datetime, timedelta
import math
from collections import defaultdict

# ------------------ 1. Підключення до MySQL ------------------
db_config = {
    'host': 'localhost',
    'user': 'bilima',
    'password': '1111',
    'database': 'caps_diploma'
}

# ------------------ 2. Параметри переналадки ------------------
color_transition_matrix = {
    "білий": {"білий": 0.0, "прозорий": 0.2, "жовтий": 0.5, "золотий": 0.8, "червоний": 1.0, "зелений": 1.3,
              "блакитний": 1.5, "синій": 1.8, "чорний": 2.5},
    "прозорий": {"білий": 0.2, "прозорий": 0.0, "жовтий": 0.3, "золотий": 0.5, "червоний": 0.7, "зелений": 1.0,
                 "блакитний": 1.2, "синій": 1.5, "чорний": 2.0},
    "жовтий": {"білий": 0.5, "прозорий": 0.3, "жовтий": 0.0, "золотий": 0.3, "червоний": 0.6, "зелений": 0.9,
               "блакитний": 1.2, "синій": 1.6, "чорний": 2.2},
    "золотий": {"білий": 0.8, "прозорий": 0.5, "жовтий": 0.3, "золотий": 0.0, "червоний": 0.4, "зелений": 0.8,
                "блакитний": 1.2, "синій": 1.6, "чорний": 2.3},
    "червоний": {"білий": 1.0, "прозорий": 0.7, "жовтий": 0.6, "золотий": 0.4, "червоний": 0.0, "зелений": 0.6,
                 "блакитний": 1.0, "синій": 1.4, "чорний": 2.0},
```

```
    "зелений": {"білий": 1.3, "прозорий": 1.0, "жовтий": 0.9, "золотий": 0.8, "червоний": 0.6, "зелений": 0.0,
                "блакитний": 0.6, "синій": 1.0, "чорний": 1.8},
    "блакитний": {"білий": 1.5, "прозорий": 1.2, "жовтий": 1.2, "золотий": 1.2, "червоний": 1.0, "зелений": 0.6,
                  "блакитний": 0.0, "синій": 0.5, "чорний": 1.5},
    "синій": {"білий": 1.8, "прозорий": 1.5, "жовтий": 1.6, "золотий": 1.6, "червоний": 1.4, "зелений": 1.0,
              "блакитний": 0.5, "синій": 0.0, "чорний": 1.2},
    "чорний": {"білий": 2.5, "прозорий": 2.0, "жовтий": 2.2, "золотий": 2.3, "червоний": 2.0, "зелений": 1.8,
               "блакитний": 1.5, "синій": 1.2, "чорний": 0.0}
}
DEFAULT_COLOR_CHANGE = timedelta(hours=2.5)
MOLD_CHANGE_TIME = timedelta(hours=2.5)


# ------------------ 3. Завантаження даних ------------------
def load_orders(cursor):
    cursor.execute("SELECT order_id, order_date, delivery_date, cap_id, quantity FROM orders;")
    orders = cursor.fetchall()
    for o in orders:
        dt = o['delivery_date']
        if isinstance(dt, datetime):
            delivery = dt.replace(hour=8, minute=0, second=0)
        else:
            delivery = datetime(dt.year, dt.month, dt.day, 8, 0, 0)
        o['deadline'] = delivery - timedelta(days=1)
    return sorted(orders, key=lambda x: x['deadline'])


def load_caps(cursor):
    cursor.execute("SELECT cap_id, format, weight, color FROM caps;")
    caps = cursor.fetchall()
    return {c['cap_id']: c for c in caps}
```

```
def load_molds(cursor):
    cursor.execute(
        "SELECT mold_id, name AS mold_name, machine_id, cap_id AS mold_cap_id, cycle_time,
slots FROM molds;")
    return cursor.fetchall()


# ------------------ 4. Допоміжні функції ------------------
def compute_transition(prev_color, prev_mold, next_color, next_mold):
    t = timedelta(0)
    if prev_color and prev_color != next_color:
        hours = color_transition_matrix.get(prev_color, {}).get(next_color)
        t += timedelta(hours=hours) if hours is not None else DEFAULT_COLOR_CHANGE
    if prev_mold and prev_mold != next_mold:
        t += MOLD_CHANGE_TIME
    return t


def total_transition_cost(seq):
    cost = 0.0
    prev_c, prev_m = None, None
    for t in seq:
        if prev_c and prev_c != t['cap_color']:
            hrs = color_transition_matrix.get(prev_c, {}).get(t['cap_color'])
            cost += hrs if hrs is not None else DEFAULT_COLOR_CHANGE.total_seconds() / 3600
        if prev_m and prev_m != t['mold_name']:
            cost += MOLD_CHANGE_TIME.total_seconds() / 3600
        prev_c, prev_m = t['cap_color'], t['mold_name']
    return cost


def recalc_sequence(seq, prod_start):
    current = prod_start
    prev_c, prev_m = None, None
    for t in seq:
```

```
        trans = compute_transition(prev_c, prev_m, t['cap_color'], t['mold_name'])
        new_start = current + trans
        new_end = new_start + t['duration']
        if new_end > t['deadline']:
            return False
        t['start'], t['end'] = new_start, new_end
        current, prev_c, prev_m = new_end, t['cap_color'], t['mold_name']
    return True


# ------------------ 5. Основне планування ------------------
def schedule(orders, caps_dict, molds):
    earliest = min(o['delivery_date'] for o in orders)
    prod_start = datetime(earliest.year, earliest.month, 1, 8, 0, 0) - timedelta(days=10)

    # стани
    machine_state, mold_state = {}, {}
    for m in molds:
        machine_state[m['machine_id']] = {'end': prod_start, 'color': None, 'mold': None}
        mold_state[m['mold_name']] = {'end': prod_start}

    machine_schedule = defaultdict(list)

    # жадібний відбір
    for o in orders:
        best = None
        for m in molds:
            mold_cap = caps_dict[m['mold_cap_id']]
            order_cap = caps_dict[o['cap_id']]
            if mold_cap['format'] != order_cap['format'] or abs(mold_cap['weight'] - order_cap['weight'])
> 1e-9:
                continue
            slots = m['slots']
            actual_qty = math.ceil(o['quantity'] / slots) * slots
            duration = timedelta(seconds=(actual_qty / slots) * float(m['cycle_time']))
```

```
            ms = machine_state[m['machine_id']]
            md = mold_state[m['mold_name']]
            trans = compute_transition(ms['color'], ms['mold'], order_cap['color'], m['mold_name'])
            start = max(ms['end'], md['end'], prod_start) + trans
            end = start + duration
            if end <= o['deadline']:
                if best is None or end < best[0]:
                    best = (end, m, start, actual_qty, duration)
        if best:
            end, mold_obj, start, actual_qty, duration = best
            task = {
                'order_id': o['order_id'],
                'machine_id': mold_obj['machine_id'],
                'mold_name': mold_obj['mold_name'],
                'cap_weight': caps_dict[o['cap_id']]['weight'],
                'cap_color': caps_dict[o['cap_id']]['color'],
                'start': start,
                'end': end,
                'orig_qty': o['quantity'],
                'actual_qty': actual_qty,
                'deadline': o['deadline'],
                'duration': duration,
                'delivery_date': (o['delivery_date'].date()
                                  if isinstance(o['delivery_date'], datetime)
                                  else o['delivery_date'])
            }
            mid = mold_obj['machine_id']
            machine_schedule[mid].append(task)
            machine_state[mid].update(end=end, color=task['cap_color'], mold=task['mold_name'])
            mold_state[task['mold_name']]['end'] = end
        else:
            print(f"⚠ Замовлення {o['order_id']} пропущено (не встигає до {o['deadline']})")

    # локальна оптимізація (neighbor-swap)
```

```
    for mid, seq in machine_schedule.items():
        improved = True
        while improved:
            improved = False
            base_cost = total_transition_cost(seq)
            for i in range(len(seq) - 1):
                seq[i], seq[i + 1] = seq[i + 1], seq[i]
                if recalc_sequence(seq, prod_start) and total_transition_cost(seq) < base_cost:
                    improved = True
                    break
                # відкат і повернення times
                seq[i], seq[i + 1] = seq[i + 1], seq[i]
                recalc_sequence(seq, prod_start)

        # додаткове групування за кольорами, якщо не порушує дедлайнів та знижує вартість
        colors = sorted(
            {t['cap_color'] for t in seq},
            key=lambda c: min(t['deadline'] for t in seq if t['cap_color'] == c)
        )
        grouped = []
        for c in colors:
            group = [t for t in seq if t['cap_color'] == c]
            grouped.extend(sorted(group, key=lambda t: t['deadline']))
        if recalc_sequence(grouped, prod_start) and total_transition_cost(grouped) <=
total_transition_cost(seq):
            machine_schedule[mid] = grouped

    return machine_schedule


# ---------- 6. Запис результату ----------
def write_schedule(cursor, machine_schedule):
    cursor.execute("TRUNCATE TABLE production_schedule;")
    rows = []
    for _, seq in machine_schedule.items():
```

```
        for t in seq:
            rows.append((
                t['machine_id'], t['mold_name'], t['cap_weight'], t['cap_color'],
                t['order_id'], t['start'].strftime("%Y-%m-%d %H:%M:%S"),
                t['end'].strftime("%Y-%m-%d %H:%M:%S"),
                t['orig_qty'], t['actual_qty'], t['delivery_date']
            ))
    insert = (
        "INSERT INTO production_schedule"
        " (machine_id,mold_name,cap_weight,cap_color,"
        "order_id,start_time,end_time,order_quantity,actual_quantity,delivery_date)"
        " VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s);"
    )
    cursor.executemany(insert, rows)


def print_statistics(machine_schedule, production_start, _):
    total_transition_hours = 0.0
    total_wait_time = timedelta(0)
    total_jobs = 0
    utilization = {}

    for mid, seq in machine_schedule.items():
        # Витрати на переналадку
        total_transition_hours += total_transition_cost(seq)

        # Час очікування
        for t in seq:
            total_wait_time += t['start'] - production_start
        total_jobs += len(seq)

        # Активний час і вікно для кожної машини
        active_time = sum((t['end'] - t['start'] for t in seq), timedelta(0))
        machine_start = min(t['start'] for t in seq)
        machine_end = max(t['end'] for t in seq)
```

```
        machine_window = machine_end - machine_start

        utilization[mid] = round(100 * active_time.total_seconds() / machine_window.total_seconds(),
2)

    avg_wait = total_wait_time / total_jobs if total_jobs > 0 else timedelta(0)

    print("\n📊 [СТАТИСТИКА РОЗКЛАДУ]")
    print(f"🔁 Загальні витрати на переналадку: {total_transition_hours:.2f} год")
    print(f"⏱️ Середній час очікування замовлення: {avg_wait}")
    print("🧮 Використання машин (у межах їх реального вікна роботи):")
    for mid in sorted(utilization.keys()):
        print(f"  - Машина {mid}: {utilization[mid]}%")


# ------------------ 7. Головний блок ------------------
if __name__ == "__main__":
    conn = mysql.connector.connect(**db_config)
    cur = conn.cursor(dictionary=True)

    orders = load_orders(cur)
    caps_dict = load_caps(cur)
    molds = load_molds(cur)

    sched = schedule(orders, caps_dict, molds)
    write_schedule(cur, sched)

    production_start = min(t['start'] for seq in sched.values() for t in seq)
    production_end = max(t['end'] for seq in sched.values() for t in seq)
    print_statistics(sched, production_start, production_end)
    conn.commit()
    cur.close()
    conn.close()
    print("☑️ Планування завершено та записано у production_schedule.")
```

ДОДАТОК Г. ПРОГРАМНИЙ КОД ВЕБ-ІНТЕРФЕЙСУ ДЛЯ ВІЗУАЛІЗАЦІЇ ВИРОБНИЧОГО РОЗКЛАДУ (FLASK)

```
from flask import Flask, render_template_string
import mysql.connector

app = Flask(__name__)



@app.route("/")
def index():
    # Параметри підключення до MySQL
    db_config = {
        'host': 'localhost',
        'user': 'bilima',
        'password': '1111',
        'database': 'caps_diploma'
    }

    # Підключення до БД
    conn = mysql.connector.connect(**db_config)
    cursor = conn.cursor(dictionary=True)

    # Запит з приєднанням до машин
    query = """
    SELECT ps.order_id,
           ps.mold_name,
           ps.cap_weight,
           ps.cap_color,
           ps.order_quantity,
           ps.actual_quantity,
           ps.start_time,
           ps.end_time,
           ps.delivery_date,
```

```
        m.name AS machine_name
FROM production_schedule ps
LEFT JOIN machines m ON ps.machine_id = m.machine_id
ORDER BY m.name, ps.start_time;
"""
cursor.execute(query)
schedule = cursor.fetchall()
cursor.close()
conn.close()

# Групування за машинами
machine_schedules = {}
for row in schedule:
    machine = row['machine_name'] if row['machine_name'] else "Невідома машина"
    if machine not in machine_schedules:
        machine_schedules[machine] = []
    machine_schedules[machine].append(row)

# Словник кольорів (фони)
color_map = {
    "білий": "#ffffff",
    "прозорий": "#f0f8ff",
    "жовтий": "#fffde7",
    "золотий": "#fff8e1",
    "червоний": "#ffebee",
    "зелений": "#e8f5e9",
    "блакитний": "#e3f2fd",
    "синій": "#e8eaf6",
    "чорний": "#eeeeee"
}

# HTML-шаблон
template = """
<!DOCTYPE html>
<html lang="uk">
```

```
<head>
  <meta charset="UTF-8">
  <title>Розклад виробництва</title>
  <style>
    body {
      background-color: #ffffff;
      font-family: Arial, sans-serif;
    }
    .card-header {
      background-color: transparent;
      border-bottom: 1px solid #ddd;
    }

    }
    .table th {
      background-color: #eeeeee;
    }
    .card {
      border: 1px solid #ccc;
      border-radius: 5px;
      margin-bottom: 20px;
      box-shadow: 0 1px 3px rgba(0,0,0,0.1);
    }
    h1 {
      color: #2c3e50;
    }
    table {
      width: 100%;
      border-collapse: collapse;
    }
    td, th {
      padding: 8px;
      border: 1px solid #ddd;
    }
  </style>
```

```
</head>
<body>
 <div class="container my-4">
  <h1 class="mb-4">Розклад виробництва за машинами</h1>
  {% for machine, orders in machine_schedules.items() %}
   <div class="card">
    <div class="card-header">
     <h2 class="h5 mb-0">{{ machine }}</h2>
    </div>
    <div class="card-body">
     <div class="table-responsive">
      <table class="table">
       <thead>
        <tr>
         <th>id Замовлення</th>
         <th>Пресформа</th>
         <th>Вага кришки</th>
         <th>Колір кришки</th>
         <th>Кількість</th>
         <th>Фактична кількість</th>
         <th>Час початку</th>
         <th>Час закінчення</th>
         <th>Дата доставки</th>
        </tr>
       </thead>
       <tbody>
        {% for order in orders %}
         <tr style="background-color: {{ color_map.get(order.cap_color, '#ffffff') }};">
          <td>{{ order.order_id }}</td>
          <td>{{ order.mold_name }}</td>
          <td>{{ order.cap_weight }}</td>
          <td>{{ order.cap_color }}</td>
          <td>{{ "{:,}".format(order.order_quantity).replace(",", " ") }}</td>
          <td>{{ "{:,}".format(order.actual_quantity).replace(",", " ") }}</td>
          <td>{{ order.start_time }}</td>
```

```
                <td>{{ order.end_time }}</td>
                <td>
                  {% if order.delivery_date %}
                    {{ order.delivery_date.strftime("%Y-%m-%d") }}
                  {% else %}
                    -
                  {% endif %}
                </td>
              </tr>
            {% endfor %}
          </tbody>
        </table>
      </div>
    </div>
  </div>
 {% endfor %}
 </div>
</body>
</html>
"""
return render_template_string(template, machine_schedules=machine_schedules,
color_map=color_map)


if __name__ == '__main__':
    app.run(debug=True)
```

ДОДАТОК Д. РОЗКЛАД ВИРОБНИЦТВА, СФОРМОВАНИЙ БАЗОВИМ ЖАДІБНИМ АЛГОРИТМОМ

Розклад виробництва за машинами

MAlpha

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
86	PF-21	3.35	червоний	433 090	433 152	2025-05-22 08:00:00	2025-05-22 18:09:07	2025-06-04
25	PF-21	3.35	зелений	429 130	429 216	2025-05-22 19:09:07	2025-05-23 05:12:42	2025-06-05
50	PF-21	3.35	червоний	304 890	304 896	2025-05-23 06:12:42	2025-05-23 13:21:27	2025-06-06
97	PF-21	3.35	жовтий	397 190	397 248	2025-05-23 14:21:27	2025-05-23 23:40:05	2025-06-06
47	PF-21	3.35	синій	322 960	323 040	2025-05-24 00:40:05	2025-05-24 08:14:22	2025-06-09
84	PF-21	3.35	зелений	430 940	430 944	2025-05-24 09:14:22	2025-05-24 19:20:23	2025-06-09
69	PF-21	3.35	білий	332 410	332 448	2025-05-24 20:20:23	2025-05-25 04:07:53	2025-06-12
70	PF-21	3.35	зелений	368 240	368 256	2025-05-25 05:07:53	2025-05-25 13:45:45	2025-06-12

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
38	PF-21	3.35	синій	358 620	358 656	2025-05-25 14:45:45	2025-05-25 23:10:06	2025-06-20
75	PF-21	3.35	золотий	403 460	403 488	2025-05-26 00:10:06	2025-05-26 09:37:30	2025-06-23
89	PF-21	3.35	синій	371 780	371 808	2025-05-26 10:37:30	2025-05-26 19:20:22	2025-06-24
54	PF-21	3.35	прозорий	339 160	339 168	2025-05-26 20:20:22	2025-05-27 04:17:19	2025-06-25
61	PF-21	3.35	синій	384 010	384 096	2025-05-27 05:17:19	2025-05-27 14:17:27	2025-06-29

MBeta

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
67	PF-19	6.10	жовтий	136 020	136 032	2025-05-22 08:00:00	2025-05-22 19:39:03	2025-06-01
13	PF-19	6.10	прозорий	119 990	120 000	2025-05-22 20:39:03	2025-05-23 06:55:43	2025-06-04
73	PF-19	6.10	червоний	132 870	132 888	2025-05-23 07:55:43	2025-05-23 19:18:37	2025-06-04
94	PF-30	4.30	жовтий	373 050	373 056	2025-05-23 22:48:37	2025-05-25 13:40:13	2025-06-07

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
46	PF-30	4.30	синій	342 560	342 576	2025-05-25 14:40:13	2025-05-27 02:21:19	2025-06-12
23	PF-30	4.30	прозорий	354 910	354 912	2025-05-27 03:21:19	2025-05-28 16:19:31	2025-06-13
2	PF-30	4.30	блакитний	346 700	346 704	2025-05-28 17:19:31	2025-05-30 05:26:25	2025-06-14
40	PF-30	4.30	зелений	301 260	301 272	2025-05-30 06:26:25	2025-05-31 13:49:22	2025-06-14
51	PF-30	4.30	чорний	430 510	430 512	2025-05-31 14:49:22	2025-06-02 11:40:04	2025-06-21
99	PF-30	4.30	синій	385 130	385 152	2025-06-02 12:40:04	2025-06-04 04:47:16	2025-06-21
36	PF-30	4.30	червоний	376 380	376 392	2025-06-04 05:47:16	2025-06-05 20:59:43	2025-06-22
41	PF-30	4.30	білий	377 830	377 832	2025-06-05 21:59:43	2025-06-07 13:21:10	2025-06-22
12	PF-30	4.30	чорний	319 230	319 248	2025-06-07 14:21:10	2025-06-08 23:36:28	2025-06-23
88	PF-30	4.30	золотий	356 530	356 544	2025-06-09 00:36:28	2025-06-10 13:44:52	2025-06-24

MDelta

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
22	PF-20	4.50	червоний	119 880	119 880	2025-05-22 17:00:37	2025-05-23 00:38:30	2025-06-02
37	PF-20	4.50	жовтий	111 370	111 384	2025-05-23 01:38:30	2025-05-23 08:43:55	2025-06-03
43	PF-20	4.50	синій	132 980	132 984	2025-05-23 09:43:55	2025-05-23 18:11:51	2025-06-03
95	PF-20	4.50	блакитний	132 790	132 792	2025-05-23 19:11:51	2025-05-24 03:39:02	2025-06-06
63	PF-20	4.50	білий	117 190	117 192	2025-05-24 04:39:02	2025-05-24 12:06:39	2025-06-10
87	PF-20	4.50	прозорий	101 080	101 088	2025-05-24 13:06:39	2025-05-24 19:32:45	2025-06-11
45	PF-20	4.50	синій	145 740	145 752	2025-05-24 20:32:45	2025-05-25 05:49:26	2025-06-12
39	PF-19	6.10	червоний	109 340	109 344	2025-05-25 09:19:26	2025-05-25 18:41:21	2025-06-14
71	PF-19	6.10	червоний	102 150	102 168	2025-05-25 18:41:21	2025-05-26 03:26:23	2025-06-16
55	PF-25	2.60	білий	574 430	574 464	2025-05-26 06:56:23	2025-05-26 21:39:01	2025-06-20

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
60	PF-19	6.10	золотий	118 510	118 512	2025-05-27 01:09:01	2025-05-27 11:18:02	2025-06-20
52	PF-19	6.10	золотий	103 410	103 416	2025-05-27 11:18:02	2025-05-27 20:09:29	2025-06-21
68	PF-19	6.10	блакитний	123 920	123 936	2025-05-27 21:09:29	2025-05-28 07:46:22	2025-06-21
77	PF-19	6.10	чорний	143 990	144 000	2025-05-28 08:46:22	2025-05-28 21:06:22	2025-06-23
100	PF-19	6.10	прозорий	132 840	132 840	2025-05-28 22:06:22	2025-05-29 09:29:01	2025-06-24
1	PF-19	6.10	білий	104 660	104 664	2025-05-29 10:29:01	2025-05-29 19:26:53	2025-06-25
59	PF-19	6.10	білий	104 920	104 928	2025-05-29 19:26:53	2025-05-30 04:26:06	2025-06-25
35	PF-19	6.10	зелений	139 160	139 176	2025-05-30 05:26:06	2025-05-30 17:21:18	2025-06-26
32	PF-19	6.10	прозорий	142 140	142 152	2025-05-30 18:21:18	2025-05-31 06:31:48	2025-06-27
85	PF-19	6.10	синій	144 940	144 960	2025-05-31 07:31:48	2025-05-31 19:56:44	2025-06-30

MEcho

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
5	PF-15	2.95	червоний	592 360	592 384	2025-05-22 08:00:00	2025-05-23 00:11:52	2025-06-04
78	PF-11	2.35	синій	743 340	743 424	2025-05-23 03:41:52	2025-05-23 17:15:00	2025-06-05
7	PF-11	2.35	прозорий	641 350	641 376	2025-05-23 18:15:00	2025-05-24 05:56:30	2025-06-06
31	PF-11	2.35	прозорий	710 120	710 208	2025-05-24 05:56:30	2025-05-24 18:53:17	2025-06-06
10	PF-11	2.35	чорний	661 070	661 152	2025-05-24 19:53:17	2025-05-25 07:56:25	2025-06-09
74	PF-11	2.35	білий	678 170	678 240	2025-05-25 08:56:25	2025-05-25 21:18:15	2025-06-14
27	PF-15	2.95	блакитний	556 350	556 352	2025-05-26 00:48:15	2025-05-26 16:01:01	2025-06-27
6	PF-15	2.95	прозорий	605 410	605 440	2025-05-26 17:01:01	2025-05-27 09:34:19	2025-06-30

MFoxtrot

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
96	PF-26	2.75	блакитний	447 910	447 912	2025-05-22 14:33:31	2025-05-22 22:51:12	2025-06-02
72	PF-26	2.75	блакитний	420 690	420 696	2025-05-23 06:17:50	2025-05-23 14:05:16	2025-06-04
79	PF-26	2.75	червоний	396 620	396 648	2025-05-23 15:05:16	2025-05-23 22:26:00	2025-06-08
56	PF-26	2.75	білий	337 220	337 248	2025-05-23 23:26:00	2025-05-24 05:40:43	2025-06-09
28	PF-26	2.75	золотий	343 140	343 152	2025-05-24 06:40:43	2025-05-24 13:02:00	2025-06-13
21	PF-26	2.75	червоний	348 470	348 480	2025-05-24 14:02:00	2025-05-24 20:29:12	2025-06-15
93	PF-26	2.75	синій	344 570	344 592	2025-05-24 21:29:12	2025-05-25 03:52:04	2025-06-16
9	PF-26	2.75	білий	436 700	436 752	2025-05-25 04:52:04	2025-05-25 12:57:21	2025-06-18
44	PF-26	2.75	синій	412 710	412 776	2025-05-25 13:57:21	2025-05-25 21:36:00	2025-06-18
49	PF-26	2.75	синій	430 710	430 776	2025-05-25 21:36:00	2025-05-26 05:34:38	2025-06-22

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
14	PF-26	2.75	чорний	334 380	334 440	2025-05-26 06:34:38	2025-05-26 12:46:14	2025-06-24
42	PF-26	2.75	жовтий	312 870	312 912	2025-05-26 13:46:14	2025-05-26 19:33:55	2025-06-27
81	PF-39	1.85	золотий	706 850	706 944	2025-05-26 23:03:55	2025-05-27 07:27:07	2025-06-30

MGamma

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
30	PF-26	2.75	чорний	354 150	354 168	2025-05-22 08:00:00	2025-05-22 14:33:31	2025-06-02
15	PF-39	1.85	зелений	743 640	743 712	2025-05-22 18:03:31	2025-05-23 02:52:53	2025-06-04
8	PF-35	1.95	прозорий	590 740	590 784	2025-05-23 06:22:53	2025-05-23 14:14:42	2025-06-05
20	PF-28	2.05	червоний	708 530	708 576	2025-05-23 17:44:42	2025-05-24 03:10:34	2025-06-11
66	PF-28	2.05	жовтий	576 920	576 960	2025-05-24 04:10:34	2025-05-24 11:51:20	2025-06-14
64	PF-35	1.95	білий	693 110	693 120	2025-05-24 15:21:20	2025-05-25 00:34:52	2025-06-15

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
92	PF-35	1.95	прозорий	694 330	694 368	2025-05-25 01:34:52	2025-05-25 10:49:24	2025-06-17
76	PF-17	2.50	синій	692 430	692 448	2025-05-25 14:19:24	2025-05-26 00:08:28	2025-06-23
65	PF-39	1.85	червоний	622 340	622 368	2025-05-26 03:38:28	2025-05-26 11:01:28	2025-06-24
29	PF-11	2.35	червоний	654 470	654 528	2025-05-26 13:31:28	2025-05-27 00:30:33	2025-06-27
91	PF-11	2.35	прозорий	749 720	749 760	2025-05-27 01:30:33	2025-05-27 14:05:31	2025-06-29

MGolf

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
83	PF-20	4.50	білий	144 150	144 168	2025-05-22 08:00:00	2025-05-22 17:00:37	2025-06-01
48	PF-25	2.60	золотий	601 450	601 472	2025-05-22 20:30:37	2025-05-23 09:02:28	2025-06-02
3	PF-25	2.60	червоний	535 880	535 936	2025-05-23 10:02:28	2025-05-23 21:12:23	2025-06-05
82	PF-15	2.95	золотий	719 280	719 296	2025-05-24 00:42:23	2025-05-24 17:52:37	2025-06-11

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
34	PF-15	2.95	золотий	673 110	673 152	2025-05-24 17:52:37	2025-05-25 09:56:46	2025-06-12
80	PF-20	4.50	прозорий	100 050	100 056	2025-05-25 13:26:46	2025-05-25 19:41:59	2025-06-14
19	PF-20	4.50	чорний	114 670	114 672	2025-05-25 20:41:59	2025-05-26 03:52:00	2025-06-16
57	PF-20	4.50	червоний	126 560	126 576	2025-05-26 04:52:00	2025-05-26 12:46:40	2025-06-16
53	PF-20	4.50	синій	119 230	119 232	2025-05-26 13:46:40	2025-05-26 21:13:47	2025-06-19
58	PF-20	4.50	прозорий	111 210	111 216	2025-05-26 22:13:47	2025-05-27 05:10:51	2025-06-23
24	PF-20	4.50	прозорий	130 080	130 080	2025-05-27 05:10:51	2025-05-27 13:18:39	2025-06-30
98	PF-20	4.50	чорний	118 860	118 872	2025-05-27 14:18:39	2025-05-27 21:44:25	2025-06-30

MHotel

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
4	PF-26	2.75	жовтий	401 950	401 976	2025-05-22 22:51:12	2025-05-23 06:17:50	2025-06-04

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
17	PF-27	2.05	блакитний	703 350	703 392	2025-05-23 09:47:50	2025-05-23 19:09:34	2025-06-09
26	PF-39	1.85	золотий	598 180	598 272	2025-05-23 22:39:34	2025-05-24 05:45:25	2025-06-13
62	PF-17	2.50	зелений	576 870	576 960	2025-05-24 09:15:25	2025-05-24 17:26:14	2025-06-15
16	PF-27	2.05	червоний	682 340	682 368	2025-05-24 20:56:14	2025-05-25 06:01:11	2025-06-16
11	PF-27	2.05	золотий	697 220	697 248	2025-05-25 07:01:11	2025-05-25 16:18:01	2025-06-18
90	PF-27	2.05	зелений	669 190	669 216	2025-05-25 17:18:01	2025-05-26 02:12:28	2025-06-23
18	PF-27	2.05	жовтий	586 730	586 752	2025-05-26 03:12:28	2025-05-26 11:01:03	2025-06-25
33	PF-35	1.95	чорний	520 190	520 224	2025-05-26 14:31:03	2025-05-26 21:26:30	2025-06-28

ДОДАТОК Е. РОЗКЛАД ВИРОБНИЦТВА, СФОРМОВАНИЙ МОДИФІКОВАНИМ МЕТОДОМ JOB-SHOP SCHEDULING

Розклад виробництва за машинами

MAlpha

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
86	PF-21	3.35	червоний	433 090	433 152	2025-05-22 08:00:00	2025-05-22 18:09:07	2025-06-04
50	PF-21	3.35	червоний	304 890	304 896	2025-05-22 18:09:07	2025-05-23 01:17:52	2025-06-06
25	PF-21	3.35	зелений	429 130	429 216	2025-05-23 01:53:52	2025-05-23 11:57:27	2025-06-05
84	PF-21	3.35	зелений	430 940	430 944	2025-05-23 11:57:27	2025-05-23 22:03:28	2025-06-09
70	PF-21	3.35	зелений	368 240	368 256	2025-05-23 22:03:28	2025-05-24 06:41:20	2025-06-12
97	PF-21	3.35	жовтий	397 190	397 248	2025-05-24 07:35:20	2025-05-24 16:53:58	2025-06-06
47	PF-21	3.35	синій	322 960	323 040	2025-05-24 18:29:58	2025-05-25 02:04:14	2025-06-09

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
38	PF-21	3.35	синій	358 620	358 656	2025-05-25 02:04:14	2025-05-25 10:28:36	2025-06-20
89	PF-21	3.35	синій	371 780	371 808	2025-05-25 10:28:36	2025-05-25 19:11:27	2025-06-24
61	PF-21	3.35	синій	384 010	384 096	2025-05-25 19:11:27	2025-05-26 04:11:35	2025-06-29
69	PF-21	3.35	білий	332 410	332 448	2025-05-26 05:59:35	2025-05-26 13:47:06	2025-06-12
75	PF-21	3.35	золотий	403 460	403 488	2025-05-26 14:35:06	2025-05-27 00:02:30	2025-06-23
54	PF-21	3.35	прозорий	339 160	339 168	2025-05-27 00:32:30	2025-05-27 08:29:27	2025-06-25

MBeta

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
67	PF-19	6.10	жовтий	136 020	136 032	2025-05-22 08:00:00	2025-05-22 19:39:03	2025-06-01
94	PF-30	4.30	жовтий	373 050	373 056	2025-05-22 22:09:03	2025-05-24 13:00:39	2025-06-07
13	PF-19	6.10	прозорий	119 990	120 000	2025-05-24 15:48:39	2025-05-25 02:05:19	2025-06-04

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
23	PF-30	4.30	прозорий	354 910	354 912	2025-05-25 04:35:19	2025-05-26 17:33:31	2025-06-13
73	PF-19	6.10	червоний	132 870	132 888	2025-05-26 20:45:31	2025-05-27 08:08:25	2025-06-04
36	PF-30	4.30	червоний	376 380	376 392	2025-05-27 10:38:25	2025-05-29 01:50:52	2025-06-22
46	PF-30	4.30	синій	342 560	342 576	2025-05-29 03:14:52	2025-05-30 14:55:58	2025-06-12
99	PF-30	4.30	синій	385 130	385 152	2025-05-30 14:55:58	2025-06-01 07:03:10	2025-06-21
40	PF-30	4.30	зелений	301 260	301 272	2025-06-01 08:03:10	2025-06-02 15:26:07	2025-06-14
2	PF-30	4.30	блакитний	346 700	346 704	2025-06-02 16:02:07	2025-06-04 04:09:01	2025-06-14
51	PF-30	4.30	чорний	430 510	430 512	2025-06-04 05:39:01	2025-06-06 02:29:43	2025-06-21
12	PF-30	4.30	чорний	319 230	319 248	2025-06-06 02:29:43	2025-06-07 11:45:01	2025-06-23
41	PF-30	4.30	білий	377 830	377 832	2025-06-07 14:15:01	2025-06-09 05:36:28	2025-06-22

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
88	PF-30	4.30	золотий	356 530	356 544	2025-06-09 06:24:28	2025-06-10 19:32:52	2025-06-24

MDelta

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
22	PF-20	4.50	червоний	119 880	119 880	2025-05-22 08:00:00	2025-05-22 15:37:52	2025-06-02
39	PF-19	6.10	червоний	109 340	109 344	2025-05-22 18:07:52	2025-05-23 03:29:46	2025-06-14
71	PF-19	6.10	червоний	102 150	102 168	2025-05-23 03:29:46	2025-05-23 12:14:48	2025-06-16
37	PF-20	4.50	жовтий	111 370	111 384	2025-05-23 15:20:48	2025-05-23 22:26:14	2025-06-03
43	PF-20	4.50	синій	132 980	132 984	2025-05-24 00:02:14	2025-05-24 08:30:09	2025-06-03
45	PF-20	4.50	синій	145 740	145 752	2025-05-24 08:30:09	2025-05-24 17:46:51	2025-06-12
85	PF-19	6.10	синій	144 940	144 960	2025-05-24 20:16:51	2025-05-25 08:41:47	2025-06-30
95	PF-20	4.50	блакитний	132 790	132 792	2025-05-25 11:41:47	2025-05-25 20:08:58	2025-06-06

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
68	PF-19	6.10	блакитний	123 920	123 936	2025-05-25 22:38:58	2025-05-26 09:15:52	2025-06-21
63	PF-20	4.50	білий	117 190	117 192	2025-05-26 13:15:52	2025-05-26 20:43:28	2025-06-10
55	PF-25	2.60	білий	574 430	574 464	2025-05-26 23:13:28	2025-05-27 13:56:07	2025-06-20
1	PF-19	6.10	білий	104 660	104 664	2025-05-27 16:26:07	2025-05-28 01:23:58	2025-06-25
59	PF-19	6.10	білий	104 920	104 928	2025-05-28 01:23:58	2025-05-28 10:23:11	2025-06-25
87	PF-20	4.50	прозорий	101 080	101 088	2025-05-28 13:05:11	2025-05-28 19:31:17	2025-06-11
100	PF-19	6.10	прозорий	132 840	132 840	2025-05-28 22:01:17	2025-05-29 09:23:56	2025-06-24
32	PF-19	6.10	прозорий	142 140	142 152	2025-05-29 09:23:56	2025-05-29 21:34:26	2025-06-27
60	PF-19	6.10	золотий	118 510	118 512	2025-05-29 22:04:26	2025-05-30 08:13:27	2025-06-20
52	PF-19	6.10	золотий	103 410	103 416	2025-05-30 08:13:27	2025-05-30 17:04:54	2025-06-21

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
77	PF-19	6.10	чорний	143 990	144 000	2025-05-30 19:22:54	2025-05-31 07:42:54	2025-06-23
35	PF-19	6.10	зелений	139 160	139 176	2025-05-31 09:30:54	2025-05-31 21:26:07	2025-06-26

MEcho

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
5	PF-15	2.95	червоний	592 360	592 384	2025-05-22 08:00:00	2025-05-23 00:11:52	2025-06-04
29	PF-11	2.35	червоний	654 470	654 528	2025-05-23 02:41:52	2025-05-23 14:37:46	2025-06-27
78	PF-11	2.35	синій	743 340	743 424	2025-05-23 16:01:46	2025-05-24 05:34:53	2025-06-05
76	PF-17	2.50	синій	692 430	692 448	2025-05-24 08:04:53	2025-05-24 19:18:06	2025-06-23
7	PF-11	2.35	прозорий	641 350	641 376	2025-05-24 23:18:06	2025-05-25 10:59:36	2025-06-06
31	PF-11	2.35	прозорий	710 120	710 208	2025-05-25 10:59:36	2025-05-25 23:56:23	2025-06-06
91	PF-11	2.35	прозорий	749 720	749 760	2025-05-25 23:56:23	2025-05-26 13:36:26	2025-06-29

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
10	PF-11	2.35	чорний	661 070	661 152	2025-05-26 15:36:26	2025-05-27 03:39:35	2025-06-09
27	PF-15	2.95	блакитний	556 350	556 352	2025-05-27 07:39:35	2025-05-27 22:52:20	2025-06-27

MFoxtrot

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
96	PF-26	2.75	блакитний	447 910	447 912	2025-05-22 08:00:00	2025-05-22 16:17:40	2025-06-02
72	PF-26	2.75	блакитний	420 690	420 696	2025-05-22 16:17:40	2025-05-23 00:05:07	2025-06-04
17	PF-27	2.05	блакитний	703 350	703 392	2025-05-23 02:35:07	2025-05-23 11:56:51	2025-06-09
26	PF-39	1.85	золотий	598 180	598 272	2025-05-23 15:38:51	2025-05-23 22:44:42	2025-06-13
62	PF-17	2.50	зелений	576 870	576 960	2025-05-24 02:02:42	2025-05-24 10:13:31	2025-06-15
90	PF-28	2.05	зелений	669 190	669 216	2025-05-24 12:43:31	2025-05-24 21:37:58	2025-06-23
16	PF-27	2.05	червоний	682 340	682 368	2025-05-25 00:43:58	2025-05-25 09:48:55	2025-06-16

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
92	PF-35	1.95	прозорий	694 330	694 368	2025-05-25 13:00:55	2025-05-25 22:15:26	2025-06-17
18	PF-28	2.05	жовтий	586 730	586 752	2025-05-26 01:03:26	2025-05-26 08:52:02	2025-06-25
33	PF-35	1.95	чорний	520 190	520 224	2025-05-26 13:34:02	2025-05-26 20:29:29	2025-06-28

MGamma

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
30	PF-26	2.75	чорний	354 150	354 168	2025-05-22 08:00:00	2025-05-22 14:33:31	2025-06-02
15	PF-39	1.85	зелений	743 640	743 712	2025-05-22 18:51:31	2025-05-23 03:40:53	2025-06-04
8	PF-35	1.95	прозорий	590 740	590 784	2025-05-23 07:10:53	2025-05-23 15:02:42	2025-06-05
20	PF-28	2.05	червоний	708 530	708 576	2025-05-23 18:14:42	2025-05-24 03:40:34	2025-06-11
65	PF-39	1.85	червоний	622 340	622 368	2025-05-24 06:10:34	2025-05-24 13:33:35	2025-06-24
74	PF-11	2.35	білий	678 170	678 240	2025-05-24 17:03:35	2025-05-25 04:26:32	2025-06-14

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
66	PF-28	2.05	жовтий	576 920	576 960	2025-05-25 07:26:32	2025-05-25 15:07:18	2025-06-14
11	PF-27	2.05	золотий	697 220	697 248	2025-05-25 17:55:18	2025-05-26 03:12:08	2025-06-18
81	PF-39	1.85	золотий	706 850	706 944	2025-05-26 05:42:08	2025-05-26 14:05:20	2025-06-30

MGolf

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
83	PF-20	4.50	білий	144 150	144 168	2025-05-22 08:00:00	2025-05-22 17:00:37	2025-06-01
48	PF-25	2.60	золотий	601 450	601 472	2025-05-22 20:18:37	2025-05-23 08:50:28	2025-06-02
82	PF-15	2.95	золотий	719 280	719 296	2025-05-23 11:20:28	2025-05-24 04:30:42	2025-06-11
34	PF-15	2.95	золотий	673 110	673 152	2025-05-24 04:30:42	2025-05-24 20:34:51	2025-06-12
3	PF-25	2.60	червоний	535 880	535 936	2025-05-24 23:28:51	2025-05-25 10:38:46	2025-06-05
57	PF-20	4.50	червоний	126 560	126 576	2025-05-25 13:08:46	2025-05-25 21:03:26	2025-06-16

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
80	PF-20	4.50	прозорий	100 050	100 056	2025-05-25 21:45:26	2025-05-26 04:00:39	2025-06-14
58	PF-20	4.50	прозорий	111 210	111 216	2025-05-26 04:00:39	2025-05-26 10:57:42	2025-06-23
6	PF-15	2.95	прозорий	605 410	605 440	2025-05-26 13:27:42	2025-05-27 03:54:52	2025-06-30
24	PF-20	4.50	прозорий	130 080	130 080	2025-05-27 06:24:52	2025-05-27 14:32:40	2025-06-30
19	PF-20	4.50	чорний	114 670	114 672	2025-05-27 16:32:40	2025-05-27 23:42:41	2025-06-16
98	PF-20	4.50	чорний	118 860	118 872	2025-05-27 23:42:41	2025-05-28 07:08:28	2025-06-30
53	PF-20	4.50	синій	119 230	119 232	2025-05-28 08:20:28	2025-05-28 15:47:35	2025-06-19

MHotel

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
4	PF-26	2.75	жовтий	401 950	401 976	2025-05-22 08:00:00	2025-05-22 15:26:38	2025-06-04
42	PF-26	2.75	жовтий	312 870	312 912	2025-05-22 15:26:38	2025-05-22 21:14:19	2025-06-27

id Замовлення	Пресформа	Вага кришки	Колір кришки	Кількість	Фактична кількість	Час початку	Час закінчення	Дата доставки
79	PF-26	2.75	червоний	396 620	396 648	2025-05-22 21:50:19	2025-05-23 05:11:02	2025-06-08
21	PF-26	2.75	червоний	348 470	348 480	2025-05-23 05:11:02	2025-05-23 11:38:14	2025-06-15
56	PF-26	2.75	білий	337 220	337 248	2025-05-23 12:38:14	2025-05-23 18:52:57	2025-06-09
64	PF-35	1.95	білий	693 110	693 120	2025-05-23 21:22:57	2025-05-24 06:36:29	2025-06-15
9	PF-26	2.75	білий	436 700	436 752	2025-05-24 09:06:29	2025-05-24 17:11:46	2025-06-18
28	PF-26	2.75	золотий	343 140	343 152	2025-05-24 17:59:46	2025-05-25 00:21:03	2025-06-13
93	PF-26	2.75	синій	344 570	344 592	2025-05-25 01:57:03	2025-05-25 08:19:56	2025-06-16
44	PF-26	2.75	синій	412 710	412 776	2025-05-25 08:19:56	2025-05-25 15:58:34	2025-06-18
49	PF-26	2.75	синій	430 710	430 776	2025-05-25 15:58:34	2025-05-25 23:57:12	2025-06-22
14	PF-26	2.75	чорний	334 380	334 440	2025-05-26 01:09:12	2025-05-26 07:20:48	2025-06-24

ДОДАТОК Ж. ВІДГУК НА КВАЛІФІКАЦІЙНУ РОБОТУ

Відгук
на кваліфікаційну роботу бакалавра
студентки групи *124 – 21 – 1* *Біліми Вероніки Костянтинівни*
спеціальності *124 Системний аналіз*

Тема кваліфікаційної роботи: *«Підвищення ефективності розподілу виробничих ресурсів при виготовлені пластикових кришок»*

Обсяг кваліфікаційної роботи *91* стор.

Мета кваліфікаційної роботи: *підвищення ефективності розподілу виробничих ресурсів шляхом створення автоматизованої системи побудови виробничого розкладу із застосуванням жадібного алгоритму та модифікованого підходу до задачі job-shop scheduling*

Актуальність теми: *в роботі вирішено актуальну проблему підприємства ТОВ «Ретал України», а саме створення автоматизованої системи планування виробництва кришок. За результатами роботи було розроблено програмний комплекс, що 1) формує розклад використання виробничих машин на наступний місяць за кілька хвилин (раніше розклад формувався вручну людьми, що займало 1-2 тижня); 2) покращено ефективність роботи виробничих машин шляхом зменшення часу очікування та перевитрат матеріалів.*

Тема кваліфікаційної роботи безпосередньо пов'язана з об'єктом діяльності бакалавра спеціальності 124 Системний аналіз, оскільки *1) використано передові розробки в області методів календарного планування виробничих робіт та аналізу даних; 2) проведено аналіз сучасної літератури, відмічено переваги та недоліки існуючих методів; 3) самостійно розроблено програмний комплекс з календарного планування виробничих робіт, що включає в себе жадібний алгоритм, модифікований алгоритм job-scheduling.*

Виконані в кваліфікаційній роботі завдання відповідають вимогам ступеня бакалавра. Оригінальність наукових рішень полягає в *1) в розробці та впроваджені нового ефективного алгоритму календарного планування виробничих робіт job-scheduling з локальними перебудовами розкладу; 2) розробці нового програмного рішення, що складається з серверної частини з БД MySQL та веб-інтерфейсу Flask, що дозволяє зручно створювати виробничий розклад.*

Практичне значення результатів кваліфікаційної роботи полягає в *автоматизації процесу планування виробництва, зменшені часу очікування виробничих машин та мінімізації витрат матеріалів при переключені між замовленнями, що дозволяє впровадити результати роботи в виробничі підприємства.*

Висновки підтверджують можливість використання результатів роботи в *1) на підприємствах, з виробництва кольорових кришок, зокрема ТОВ «Ретал Україна»; 2) на інших підприємствах, що обробляють замовлення та мають потребу в формування розкладу виробництва по машинам.*

Оформлення пояснювальної записки та демонстраційного матеріалу до неї виконано згідно з вимогами. Роботу виконано самостійно, відповідно до завдання та у повному обсязі.

У роботі відзначено такі недоліки: _*не виявлено.*

Кваліфікаційна робота в цілому заслуговує оцінки:_*відмінно.*

З урахуванням висловлених зауважень авторка заслуговує присвоєння освітньої кваліфікації «бакалавр з системного аналізу».

Керівник кваліфікаційної роботи бакалавра,
доктор філософії, доцент каф. САУ_____/ Хабарлак К.С.