COMPUTER SYSTEMS AND NETWORKS

Vol. 5, No. 1, 2023

DOI: https://doi.org/10.23939/csn2023.01.079

УДК 004.4

КОМПЛЕКСНИЙ АЛГОРИТМ ДЛЯ СИСТЕМИ ПРОГРАМНОГО ОБСЛУГОВУВАННЯ ЛОГІСТИКИ ГУМАНІТАРНИХ ПОСЛУГ

А. Ф. Обшта, В. В. Бугаєць

Національний університет "Львівська політехніка",
кафедра електронних обчислювальних машин
E-mail: anatolii.f.obshta@lpnu.ua , vladyslav.buhaiets.mkisp.2022 @lpnu.ua

© Обшта А. Ф., Бугаєць В. В., 2023

Розглянуто проблему проєктування кросплатформної системи програмного обслуговування логістики гуманітарних послуг. Розглянуто програмні аналоги міжнародних поставок. Наведено порівняння популярних програмних продуктів у сфері міжнародної торгівлі.

Оглянуто технології для розроблення програмного обслуговування для забезпечення кросплатформності системи для логістики гуманітарних послуг.

Наведено алгоритми, використовувані у системі для розв'язування транспортної задачі, а саме задачі маршрутизації. за допомогою яких вибирають найоптимальніший шлях між пунктами постачання.

Запропоновано загальний алгоритм роботи системи та наведено структурну схему аплікації.

Ключові слова: кросплатформність; програмне обслуговування; логістика; алгоритм; аплікація.

Вступ

Сфера автоматизації інформаційних технологій логістичних послуг сьогодні доволі добре розвинена, однак не досконала. Зацікавленість людей у використанні логістичних послуг постійно зростає, адже клієнтів та робітників з кожним роком стає все більше і більше. Особливо помітно зростає кількість вакансій та потреба у професіоналах, що спонукає університети, інститути та різні освітні платформи створювати курси, на яких навчаються та готуються до майбутньої роботи нові робочі кадри. Результат – підвищення попиту на сферу інформаційних технологій, що спонукає клієнтів вкладати більші інвестиції, що повинно сприяти нарощуванню кількості та поліпшенню якості функціонування робочих ресурсів. Все це дає можливість галузі рости і розвиватися в геометричній прогресії.

Огляд наявних засобів і програмного забезпечення показує, що немає сервісів доставки гуманітарних послуг за умов ведення війни, які можуть працювати практично з будь-якими конфігураціями інформаційних пристроїв, різними версіями програмного забезпечення, різним обладнанням та платформами. Можливість використання аплікації на будь-якій платформі – головний пріоритет. Поки що дуже мало сучасних аналогів, застосовних саме для доставки гуманітарних вантажів. Одним із важливих завдань постачання гуманітарних вантажів за умов війни є автоматизація знаходження оптимального маршруту для доставки вантажів у сегменті міжнародних та міжконтинентальних постачань. Більшість сервісів автоматизації постачання у цьому сегменті не вирішує поставленого завданняі. Основна мета праці – створити кросплатформну аплікацію, яка зможе функціонувати незалежно від операційної системи або апаратного забезпечення та розв'язуватиме задачі автоматизації вибору оптимальних за різними критеріями маршрутів постачання гуманітарних послуг.

1. Проблема та аналіз програмного забезпечення логістики міжнародної торгівлі

Розроблення кросплатформної системи програмного обслуговування логістики гуманітарних послуг є основною метою цієї роботи. Програмні продукти, що працюють незалежно від конфігурацій засобів, тобто операційних систем, апаратних засобів, підтримки широкого спектра пристроїв, дуже цінують на ринку завдяки їхнім позитивним рисам. Розглянемо основні, на нашу думку, популярні версії програмного забезпечення логістики міжнародних послуг доставок.

DHL: DHL – один з найбільших і найвідоміших міжнародних сервісів доставки. Надає широкий спектр послуг, зокрема доставку документів, товарів і вантажів. Сервіс використовує власні технології, такі як PHP, для розроблення вебсайту та мобільних додатків. Додаток не підтримує кросплатформність.

FEDEX: FedEx – міжнародний сервіс доставки, який спеціалізується на надшвидкісній доставці. Сервіс використовує власні технології, такі як .NET, для розроблення вебсайту та мобільних додатків. Додаток не підтримує кросплатформність.

Nova Poshta: Нова пошта – український міжнародний сервіс доставки, який пропонує широкий спектр послуг, серед яких доставка документів, товарів і вантажів. Сервіс використовує власні технології, такі як PHP, для розроблення вебсайту та мобільних додатків. Додаток не підтримує кросплатформність.

Meest Express: Meest Express – український міжнародний сервіс доставки, який пропонує широкий спектр послуг, зокрема доставку документів, товарів і вантажів. Сервіс використовує власні технології, такі як PHP, для розроблення вебсайту та мобільних додатків. Додаток не підтримує кросплатформність.

Сервіси, характеристики яких подано в таблиці, неможливо використовувати як кросплатформні додатки через використання власних технологій. Це пов’язано з тим, що такі технології, як PHP, .NET і MySQL, не є загальнодоступними, їх потрібно спеціально розробляти для кожної платформи. Виконуючи поставлене перед нами завдання, потрібно вибирати ті технології розроблення, які забезпечують кросплатформність.

Порівняльна характеристика онлайн-платформ відеохостингу

Сервіс	Платформа	Технології	Неможливе використання як кросплатформного додатка
DHL	Вебсайт, мобільний додаток	PHP, MySQL, JavaScript	Неможливе через PHP
FEDEX	Вебсайт, мобільний додаток	.NET, SQL Server, JavaScript	Неможливе через .NET
Nova Poshta	Вебсайт, мобільний додаток	PHP, MySQL, JavaScript	Неможливе через PHP
Meest Express	Вебсайт, мобільний додаток	PHP, MySQL, JavaScript	Неможливе через PHP

2. Огляд технологій для розроблення кросплатформного програмного продукту

Як ми уже зазначали, основна наша мета – забезпечення кросплатформності створюваного програмного комплексу.

Кросплатформність ми розуміємо як здатність програми працювати на різних операційних системах і апаратних платформах без необхідності модифікації коду програми. Цього досягають використанням абстракції, яка дає змогу програмі працювати незалежно від конкретної платформи.

Кросплатформність є універсальною, оскільки дає змогу розробникам створювати програми, доступні для більшої кількості користувачів. Це особливо важливо в епоху, коли у користувачів є широкий спектр пристроїв і платформ.

Перерахуємо переваги кросплатформності:

- Широке охоплення користувачів. Кросплатформні програми можна запустити на будь-якій платформі, що підтримує мову програмування, на якій вони написані. Це дає змогу розробникам залучити більшу кількість користувачів, незалежно від того, яку операційну систему або апаратну платформу вони використовують.
- Зменшення витрат. Кросплатформна розробка може допомогти заощадити гроші, оскільки користувачам не потрібно розробляти окремі програми для кожної платформи. Це може бути особливо корисно для малих і середніх підприємств, які не мають великих бюджетів на розроблення.
- Збільшення продуктивності. Кросплатформні програми можуть бути продуктивнішими, оскільки не потребують додаткових витрат на розроблення та підтримку. Це особливо корисно для розробників, які хочуть швидко вивести свої програми на ринок.

Є декілька мов програмування та фреймворків, які підтримують кросплатформність: Java [1], C#, Python, JavaScript. Для розроблення системи програмного обслуговування логістики гуманітарних послуг використовують такі інструментальні продукти, як Java та Spring Framework[2]. Вибір цих технологій є майже очевидним з декількох причин:

- Популярність. Java – дуже популярна мова програмування, підтримується багатьма платформами і фреймворками. Це означає, що розробники можуть знайти чимало ресурсів, таких як навчальні матеріали, бібліотеки й інструменти, для розроблення кросплатформних додатків на Java.
- Поширеність. Java підтримується багатьма платформами, серед яких веббраузери, мобільні пристрої, сервери і десктопні комп'ютери. Це означає, що додатки, написані на Java, можна запускати на широкому спектрі пристроїв і платформ.
- Ефективність. Java – ефективна мова програмування, яка може допомогти створити додатки, які працюють швидко і плавно.
- Безпека. Java має вбудовані функції безпеки, які допомагають захистити ваші додатки від шкідливого програмного забезпечення і атак.

Java має JVM [3]. Саме цей інструмент надає можливість забезпечити кросплатформність. Аналоги JVM, після Java, розроблені також для інших мов (C#, JavaScript, Swift, Rust).

JVM (Java Virtual Machine) – віртуальна машина, яка забезпечує виконання програм Java на різних платформах. JVM компілює код Java в байт-код, який потім виконується на JVM. Вона складається з таких компонентів: класлоадер, читач байт-коду, інтерпретатор байт-коду, менеджер пам'яті, система безпеки. Робота цього інструменту передбачає декілька кроків. Перший крок – компіляція. JVM компілює код Java в байт-код, який потім виконується на JVM. Компіляцію виконує компілятор Java, який є частиною JDK (Java Development Kit). Потім виконується байт-код Java, що забезпечує доступ до апаратних ресурсів платформи. JVM робить це за допомогою інтерпретатора байт-коду, який перекладає байт-код у машинний код безпосередньо перед його виконанням. Менеджмент пам'яті необхідний для роботи віртуальної машини, для того щоб управляти пам'яттю під час виділення і звільнення пам'яті для об'єктів Java. JVM робить це за допомогою менеджера пам'яті, який використовує техніку під назвою "збирання сміття" (garbage collector). Невід'ємною частиною є система безпеки, яка запобігає несанкціонованому доступу до пам'яті та ресурсів платформи за допомогою функцій перевірки типовості та прав доступу. Цей інструмент складний для розуміння, однак ефективний для підтримання кросплатформності програмного забезпечення.

3. Аналіз методологій вирішення проблеми маршрутизації

Транспортна задача – це задача лінійного програмування, яка формулюється як задача складання оптимального плану перевезення вантажів із M пунктів відправлення у N пунктів призначення за умови, що вартості перевезення із пункту відправлення i в пункт призначення j відомі [20].

Наведемо математичне формулювання транспортної задачі: якщо x_{ij} – об'єм вантажу, який перевозиться від постачальника i до споживача j, то вартість такого перевезення дорівнює $c_{ij}x_{ij}$. Сумарні витрати на перевезення усіх вантажів становитимуть $\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$. Треба мінімізувати вартість перевезень, тому цільова функція транспортної задачі матиме вигляд

$$Z(X) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \to Min.$$

Система обмежень містить дві групи рівнянь. Перша група із m рівнянь вказує на те, що запаси всіх m постачальників вивозяться повністю, тобто

$$\sum_{j=1}^{n} x_{ij} = a_i, \quad i = 1,2,\dots m.$$

Друга група обмежень із n рівнянь свідчить, що потреби всіх споживачів повністю задовольняються, тобто $\sum_{i=1}^{m} x_{ij} = b_j, \ \ j = 1,2,\dots n.$ Оскільки $x_{ij} \geq 0$, математичну модель транспортної задачі можна записати так

$$Z(X) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \to Min \quad (1)$$

$$\sum_{j=1}^{n} x_{ij} = a_i, i = 1,2,\dots m \quad (2)$$

$$\sum_{i=1}^{m} x_{ij} = b_j, \ j = 1,2,\dots n \quad (3)$$

$$x_{ij} \geq 0, i = 1,2,\dots m, j = 1,2,\dots n. \quad (4)$$

В розглянутій моделі транспортної задачі вважається, що сумарні запаси постачальників дорівнюють сумарним потребам споживачів, тобто виконується умова

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j \quad (5)$$

Така задача називається *задачею з правильним балансом*, а її модель –*закритою*. Якщо ж рівність (5) не виконується, то задачу називають *задачею із неправильним балансом*, а її модель – *відкритою*.

Різновидом її є проблема маршрутизації (Vehicle Routing Problem). Для цієї роботи означимо, що проблемою є автоматизована логістична проблема (Automated logistics problem – ALP).

Транспортні задачі – часто використовуваний у логістиці клас алгоритмів, велика група з яких є варіаціями Проблеми маршрутизації транспортних засобів (VRP):

- Задача маршрутизації транспортних засобів з обмеженою місткістю (CVRP) [4]: задача планування маршрутів для транспортних засобів з обмеженою місткістю від центрального складу до цільових складів.
- Задача маршрутизації транспортних засобів з часовими вікнами (VRPTW)[5]: варіація CVRP, в якій склади повинні обслуговуватися в заданому інтервалі часу.
- Стохастична задача маршрутизації транспортних засобів (SVRP)[6]: варіація VRP, в якій деякі елементи задачі є випадковими, наприклад, запити клієнтів або часи поїздки.

Транспортна задача є одним із найпростіших прикладів багатокритеріальних задач. У багатокритеріальній постановці [21] елемент транспортної таблиці β_{ij} є складним типом даних, названим вектором критеріїв.

$$\beta_{ij} = \{c_{ij}^{k}, k = \overline{1,2}\}, \quad (6)$$

де c_{ij}^{k} – значення k-го критерію елемента рішення β_{ij}. Функціонал двокритеріальної транспортної задачі має вигляд

$$\begin{cases} w\left(E_{a_i,b_j}^{1}\right) = \sum_{i}^{M}\sum_{j}^{N}\left(x_{ij}c_{ij}^{1}\right) \to min \\ w\left(E_{a_i,b_j}^{2}\right) = \sum_{i}^{M}\sum_{j}^{N}\left(x_{ij}c_{ij}^{2}\right) \to min \\ \sum_{i=1}^{M} x_{ij} = b_j, j = \overline{1,N} \\ \sum_{j=1}^{N} x_{ij} = a_i, i = \overline{1,M} \end{cases} \quad (7)$$

де c_{ij}^1 – вартість доставки вантажу з *i*-го пункту відправлення до *j*-го пункту призначення; c_{ij}^2– штраф за затримку виконання замовлення, який оцінюється в балах або умовних одиницях і набуває значення від 0 до 1; x_{ij} – обсяг вантажу, що переміщується з *i*-го пункту відправлення до *j*-го пункту призначення; a_i – запас вантажу в постачальника; b_j – потреба замовника; $w\left(E_{a_i,b_j}^1\right)$ – оптимальний план перевезень із мінімальною сумарною вартістю виконання робіт; $w\left(E_{a_i,b_j}^2\right)$ – оптимальний план перевезень із мінімальним сумарним штрафом за затримку виконання замовлення.

Наведемо схему розв'язання двокритеріальної транспортної задачі.

1. Нормалізація критеріїв. Нормалізація кожного критерію задачі за допомогою алгоритму блокової нормалізації.

2. Виділення критерію, який домінує.

3. Розв'язання класичної ТЗ і визначення ефективного плану перевезень.

Розглянемо декілька алгоритмів, які використовують у комплексному алгоритмі роботи системи.

3.1. Мережа потоків

Проблема багатокомпонентних потоків передбачає одночасне перевезення декількох товарів через одну мережу так, щоб загальна кількість потоків на кожному ребрі графу багатокомпонентного потоку не перевищувала його пропускної здатності.

Подамо формальний опис багатокомпонентних потоків.

Дано неорієнтований граф (рис. 1) $G = (V, E)$ з позитивною пропускною здатністю $c(uv)$ для кожного ребра $uv \in E$ та набором товарів, пронумерованих від 1 до k, де кожен товар i визначається парою джерело-споживач $(s_i, t_i) \in V$ і позитивним попитом d_i. Для кожного товару i передається кількість, пропорційна до його попиту d_i, від його джерела s_i до його споживача t_i. Це дає одиничний потік f_i, який визначається набором потоків по ребрах f_i(vw) на ребрах $vw \in E$, де кожне ребро має довільний напрямок для відстеження напрямку, в якому потоки рухаються по ньому. Позитивний потік по ребру $f_i(vw) > 0$ позначає рух товару вперед і щодо напрямку ребра vw, тоді як негативний потік $f_i(vw) < 0$ позначає рух назад. Багатокомпонентний потік f складається із k одиничних потоків, по одному для кожного товару. У багатокомпонентному потоці f загальний потік $f(vw)$ на кожному ребрі $vw \in E$ дорівнює сумі $\sum_{i=1}^{k} |f_i(vw)|$ одиничних потоків на цьому ребрі. Прикладом застосування цієї схеми є алгоритм Форда – Фалкерсона [7].

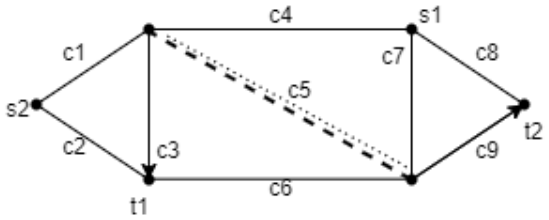

Рис. 1. Приклад багатокомпонентного потоку

3.2. Програмування обмежень

Як зазначено в [8], проблеми постачання товарів в мережах можна моделювати за допомогою обмежень (Constraint programming problems in networks – CSP). Є три основніх типи моделей: лінко-орієнтована модель, шляхо-орієнтована модель та вузло-орієнтована модель. Повний опис методу програмування обмежень подано в [9]. Одним з різновидів програмування обмежень є метод гілок і меж [19]

3.3. Мішана цілочислова оптимізація

Мішана цілочислова оптимізація (Mixed integer programming, MIP) – це тип оптимізації, який часто використовується для моделювання реальних проблем, таких як планування виробництва, розподіл ресурсів та оптимізація логістики. Ми використовуємо MIP, яка ефективно працює з

моделями на основі сум та нерівностей, подібних до математичної моделі планування передач у мережах [10, 11]. Мішана цілочислова оптимізація – метод, похідний від методу лінійного програмування. Задачу лінійного програмування сформульовано, наприклад, в [12]. В [13] розглянуто алгоритм розв'язання MIP, який використовує метод розгалужень та меж.

Алгоритми розв'язування задачі маршрутизації, імплементовані у систему: наївний алгоритм [19], алгоритм на основі Форда – Фалкерсона, CSP модель, MIP модель.

Для розв'язання задачі на основі MIP ми використовуємо програму SCIP [16], яка написана мовою Java в OR-tools, розробленою Google [17]. Система SCIP є одним з найшвидших некомерційних програм для мішаного цілочислового програмування [18]. Система SCIP у вільному доступі лише для некомерційного використання.

3.4. Комплексний алгоритм системи

Розглянуті алгоритми є частиною цілісного комплексного алгоритму, розробленого у цій роботі. Кожен із них має певні недоліки.

Наївний алгоритм не враховує порядок пріоритету замовлень на відправлення та обмеження місткості на складах, хоча він простий і швидкий, тобто за деяких умов він все ще працюватиме ефективно і буде оптимальним.

Для алгоритму Форда – Фалкерсона це неможливість опрацювання обмежень, щоб запобігти перевантаженням, що може призвести до ускладнень з погляду витрат або часу. В такому разі він не буде оптимальним і це може вплинути на якість результату.

Для CSP моделі в простих ситуаціях кількість обмежених змінних величезна навіть для конкретної невеликої задачі, тому, щоб зменшити її, необхідно використовувати додаткові методики. Щоб вирішити проблему маршрутизації за допомогою цієї моделі, потрібно використовувати CSP solver, що використовує стратегію пошуку, але вона може виявитися не оптимальною для такої задачі.

MIP модель зазвичай дає найкращі результати, але також має певні недоліки, такі як складність для розроблення та реалізації, іноді комп'ютерно складна для застосування, особливо для великих задач, або не знаходить оптимальне рішення.

Кожен з алгоритмів може бути оптимальним, залежно від різних ситуацій. Це означає, що не можна вибрати один конкретний алгоритм, що буде оптимальним у всіх випадках. Саме тому в цій роботі розроблений алгоритм, який буде універсальним, оскільки порівнюватиме результати алгоритмів, імплементованих у програмі відповідно до різних конфігурацій та параметрів. Також є можливість розширювати алгоритм завдяки тому, що легко розширювати програму, не модифікуючи, а лише додаючи в неї нову логіку, тобто додаткові алгоритми. Результатом цього алгоритму є порівняння вихідної інформації від імплементованих частин у програмі та рекомендація використовувати конкретний маршрут.

Тобто реалізована мета цієї роботи, яка полягає у тому, що розроблено алгоритм розрахунку маршруту, оптимального в найбільшій кількості ситуацій.

Принцип роботи алгоритму системи:

- Отримання вхідних даних. Користувач повинен надати вхідні дані або обмеження.
- Верифікація даних. Вхідні дані верифікуються, після чого починається пошук оптимального шляху.
- Сервіси алгоритмів. Ця частина алгоритму повинна знайти оптимальний маршрут, використовуючи імплементовані алгоритми розв'язку логістичної задачі.
- Збирач результатів. На цьому етапі збирають результати від кожного з імплементованих алгоритмів.
- Порівнювач. Це останній блок алгоритму, що відповідає за порівняння результатів та рекомендацію користувачу щодо шляху, оптимального в його випадку, та виведення інформації.

Кожен з блоків подано на рис. 2, тобто на структурній схемі системи.

4. Структура аплікації для знаходження оптимального маршруту

Для кросплатформної системи програмного обслуговування логістики гуманітарних послуг запропоновано структуру, наведену на рис. 2.

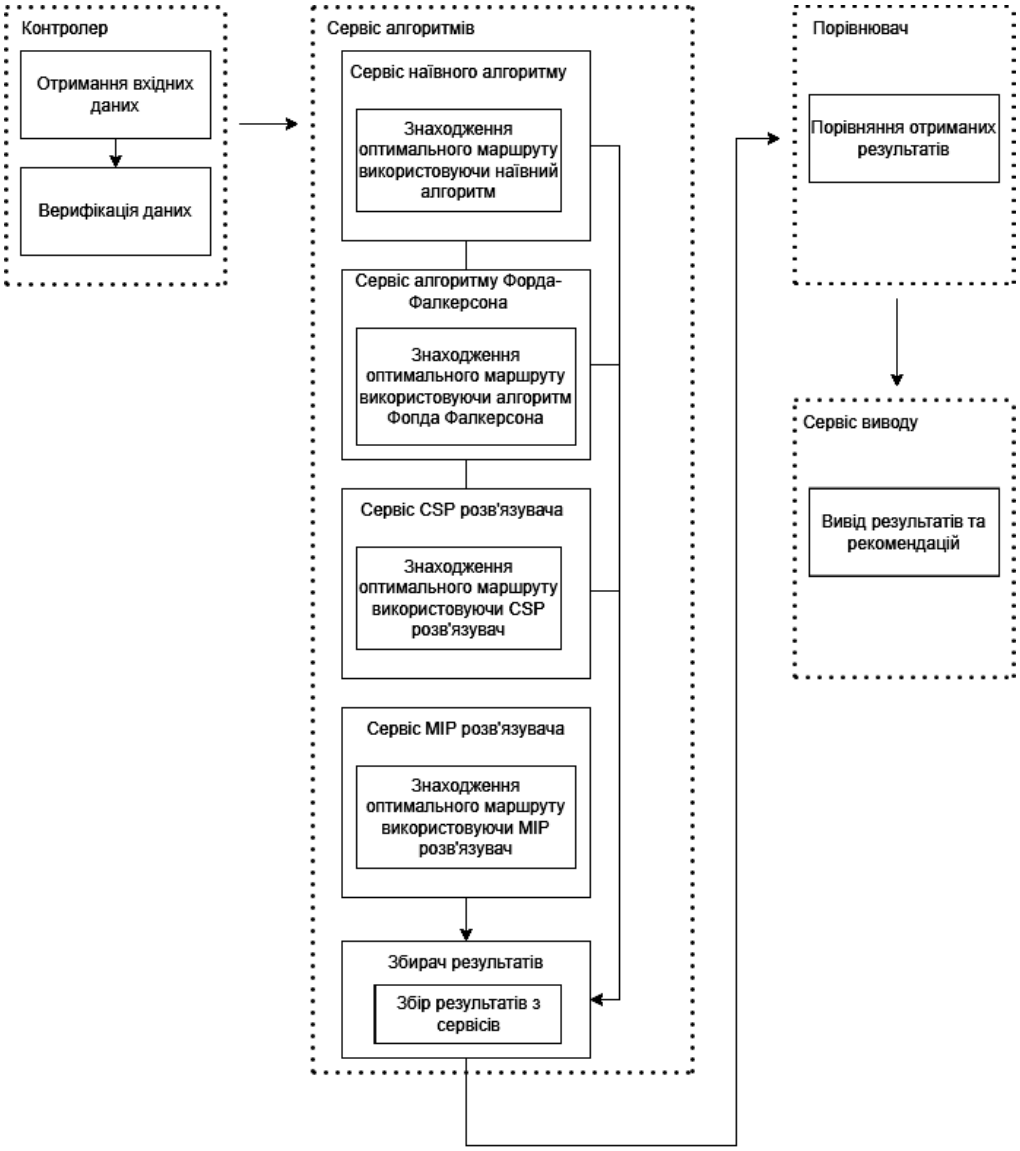

Рис. 2. Структурна схема серверної системи

Під час розроблення структури використано принципи SOLID [14], окрім цього було розглянуто Design Patterns [15], для того щоб класифікувати компоненти системи.

5. Алгоритм роботи кросплатформної системи програмного обслуговування логістики

Алгоритм роботи системи програмного обслуговування логістики наведено на рис. 3.

Розглянемо послідовність дій, які відпрацьовує система. Користувач надає вхідні дані. відтак здійснюється їх валідація. За позитивних результатів система продовжує працювати, в іншому разі припиняє роботу з поверненням до старту. Після верифікації паралельно відпрацьовують імплементовані алгоритми для знаходження маршруту.

Рис. 3. Загальний алгоритм роботи

Наступний крок – порівняння результатів. Після порівняння система виводить інформації щодо оптимального маршруту та рекомендацій.

6. Діаграма класів розробленої аплікації

Частиною цієї праці є розроблення програмного забезпечення, яке виконуватиме цілісний комплексний алгоритм, описаний вище. На рис. 4 зображена діаграма класів, створена у цій аплікації у програмному середовищі розробки Intellij IDEA.

На цій діаграмі зображено класи та інтерфейси. Їх розроблено за допомогою принципів SOLID та Design Patterns мовою Java. Кожен із класів реалізує певну мету, має свої поля, методи, застосування. На цій діаграмі також наведена реалізація (приклад – FFAlgService реалізовує AlgorithmService), агрегацію (приклад – DataController агрегує ResultCollector), залежність (між двома класами FFAlgService). Також є допоміжні класи, такі як DataRequest, Parameter, Result.

Висновки

Розглянуто проблему проєктування кросплатформної системи програмного обслуговування логістики гуманітарних послуг та програмні аналоги міжнародних постачань. Наведено порівняння популярних програмних продуктів у сфері міжнародної торгівлі.

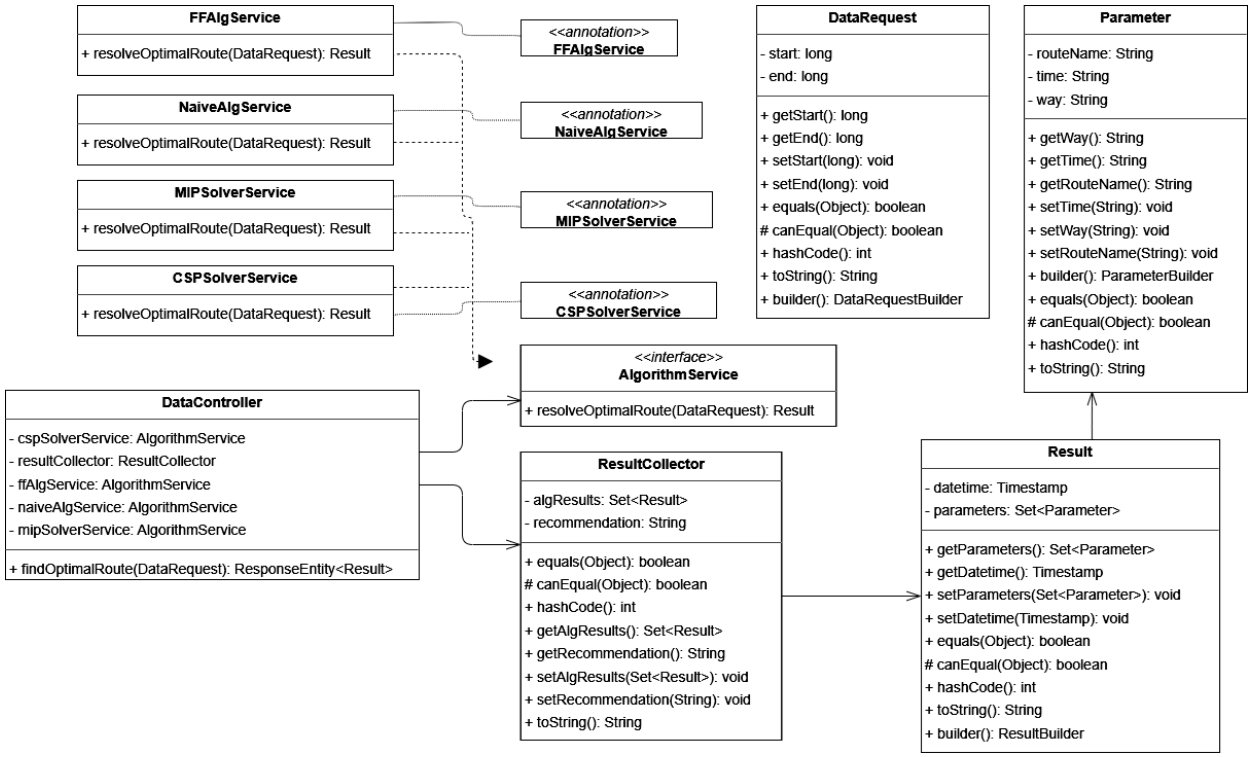

Рис. 4. Діаграма класів кросплатформної системи програмного обладнання логістики гуманітарних об'єктів

Розглянуто технології, використані для розроблення кросплатформної системи програмного обслуговування забезпечення логістики гуманітарних послуг.

Запропоновано загальний алгоритм роботи системи та подано структурну схему аплікації.

Наведено алгоритми та методи, що використовуються у системі для розрахунку оптимальних шляхів постачання гуманітарних вантажів.

Тестування програми підтвердили її ефективність та дають підстави стверджувати, що її можна упроваджувати у системи міжнародних постачань гуманітарних вантажів.

Список літератури

1. Bloch, J. (2018). Effective Java: A Tutorial. 3rd ed. Upper Saddle River, NJ: Addison-Wesley. 901 p. DOI: 10.1145/2185359.2185361.

2. Walls, C. (2018). Spring in Action: A Tutorial. Shelter Island, NY: Manning Publications Co. 500 p. DOI: 10.1145/3174280.3174282.

3. Chen, C., Xu, Y., Chen, X., Luo, X., & Wang, W. (2021). Java Virtual Machine: A Survey. ACM Computing Surveys, 54(3), 1–38. DOI: 10.1145/3456553.

4. Al-Khayyal, M. A., Birch, D. M. T., & Hassan, M. B. M. E. (2014). A review of truck dispatching problems and solution approaches. Transportation Research Part B: Methodological, 66, 1–34. DOI: 10.1016/j.trb.2014.07.002.

5. Olli Bräysy and Michel Gendreau. (2005). Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms. Transportation Science, 39(1):104–118. DOI: 10.1287/trsc.1030.0056

6. Gendreau, M., Laporte, G., & Séguin, R. (1996). Stochastic vehicle routing. European Journal of Operational Research, 88(1), 3–12. DOI: 10.1016/0377-2217(95)00090-3.

7. Shmoys, D. B., & Tardos, É. (2000). Network flows and graph algorithms. In Handbook of Discrete and Computational Geometry, 609–655. CRC Press. DOI: 10.1007/3-540-36484-0_19.

8. Simonis, H. (2006). Constraint applications in networks. In F. Rossi, P. van Beek, & T. Walsh (Eds.), Handbook of Constraint Programming, Vol. 2, 875–903. Elsevier. DOI: 10.1016/S1361-8608(05)00062-

9. Dechter, R. (2003). Constraint processing. Morgan Kaufmann. DOI: 10.1016/B978-1-55860-890-X5000-2.

10. Carsten. J. Drexl, A. (1995). A Comparison of Constraint and Mixed-Integer Programming Solvers for Batch Sequencing with Sequence Dependent Setups. ORSA Journal on computing, 7(2):160–165. DOI: 10.1287/ijoc.7.2.160

11. Holub, P., Liška, M., Rudová, H., & Troubil, P. (2010). "Comparison of CP and IP Techniques for Data Transfer Planning". In 28th Workshop of the UK Special Interest Group on Planning and Scheduling, 69–70. ISBN 978-88-904924-1-9

12. Dantzig, G. B. (1951). Maximization of a linear function of variables subject to linear inequalities. In T. C. Koopmans (Ed.), Activity Analysis of Production and Allocation, 339-347. Wiley & Chapman-Hall. ISBN: 978-0471387594.

13. Achterberg, T. (2007). "Constraint Integer Programming". Doctoral thesis, Technische Universität Berlin. DOI: 10.1007/978-3-540-68155-7.

14. Fowler, M. (2014). SOLID Principles: An Introduction. MartinFowler.com. DOI: 10.1145/2531701.2531703.

15. Freeman, E., Robson, E., Bates, B., & Sierra, K. (2014). Head First Design Patterns (1st ed.). O'Reilly. 867 p. ISBN: 978-0-596-00712-4.

16. Achterberg, T., Berthold, T., Koch, T., & Wolter, K. (2008). "Constraint Integer Programming: A New Approach to Integrate CP and MIP". In: "Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems", Vol. 01, 6–20. Springer, Berlin, Heidelberg. DOI: 10.1007/978-3-540-68155-7_4.

17. Perron, L. (2011). "Operations Research and Constraint Programming at Google." In: "Principles and Practice of Constraint Programming – CP 2011", p. 2. Publisher: Springer, Berlin Heidelberg. DOI: 10.1007/978-3-642-23786-7_1

18. Mittelmann, H. (2014). Mixed Integer Linear Programming Benchmark (MIPLIB2010). [online] Available at: <http://plato.asu.edu/ftp/milpc.html>.

19. Tuláček, M. (2014). Algorithms for automated logistics. Bc. Michal Tuláček Algorithms for automated logistics. http://www.tulacek.eu/mgr_thesis/thesis.pdf. DOI: 10.1145/2531701.2531703.

20. Molodid, O. K. (2018). The transportation problem. (Electronic resource). Kyiv, Ukraine: Igor Sikorsky Kyiv Polytechnic Institute. DOI: 10.13140/RG.2.2.27260.79842

21. Podotyaka, O. O., & Podotyaka, O. M. (2021). Solving a bicriteria transportation problem based on block normalization of criteria. Bulletin of Kharkiv National Automobile and Highway University, 92(1), 60. DOI:10.30977/BUL.2219-5548.2021.92.1.60

CROSS-PLATFORM SOFTWARE SYSTEM FOR THE LOGISTICS OF HUMANITARIAN SERVICES

A. Obshta, V. Buhaiets

Lviv Polytechnic National University,
Computer Engineering Department

© Obshta A., Buhaiets V., 2023

The problem of designing a cross-platform software system for the logistics of humanitarian services is considered. Existing software analogs of international deliveries are considered. A comparison of popular software products in the field of international trade is given.

The technologies for developing software services to ensure the cross-platform system for the logistics of humanitarian services are reviewed.

The algorithms used in the system for solving the transportation problem, namely the routing problem, are presented, with the help of which the most optimal path between the supply points is selected.

The general algorithm of the system operation is proposed and the structural diagram of the application is presented.

Key words: cross-platform; software service; logistics; algorithm' application.