УДК 656.222.3

Ю. В. ЧИБІСОВ (ДІІТ)

ІМІТАЦІЙНА МОДЕЛЬ РОЗПОДІЛУ ПОЇЗДОПОТОКУ ПО ОПТИМАЛЬНИХ МАРШРУТАХ

Запропоновано імітаційну модель для вирішення задачі раціонального розподілу поїздопотоків за двома показниками: час руху поїзда та затрати механічної роботи.

Ключові слова: раціональний розподіл поїздопотоків, оптимальний маршрут, імітаційна модель, час руху поїзда, затрати механічної роботи

Предложена имитационная модель для решения задачи рационального распределения поездопотоков по двум показателям: время движения поезда и затраты механической работы.

Ключевые слова: рациональное распределение поездопотоков, оптимальный маршрут, имитационная модель, время движения поезда, затраты механической работы

The simulation model for solution of the task of the rational train traffic volume distribution in consideration of two indices, namely the train traffic time and the expenditure of mechanical work, was suggested.

Keywords: rational train traffic volume distribution, optimum route, simulation model, train traffic time, expenditure of mechanical work

Метою даної роботи є розробка програмного забезпечення, в якому буде реалізована запропонована математична модель з визначення оптимальних маршрутів руху та розподілення поїздопотоків по цих маршрутах між відповідними пунктами.

Задача з визначення оптимальних маршрутів руху поїздопотоку ділиться на дві підзадачі, а саме:

– побудова всіх простих, допустимих маршрутів між станціями;

– розподілення поїздопотоку по отриманих маршрутах.

Кожну з підзадач можна представити як окрему компоненту програми. Крім того, при реалізації окремих компонент виникає необхідність у створені додаткових, допоміжних процедур.

Для побудови допустимих, простих маршрутів між усіма станціями необхідні дані про сітку мережі залізниці, а також інформація про поїздопотік між станціями, тому виникає потреба у записі цих даних у файлі з метою їх подальшого використання. Також необхідні процедури по роботі з файлом, а саме: допис інформації у файл, відображення у зручній формі інформації, що зберігається у файлі та корегування даних, що зберігаються у файлі.

Враховуючи підзадачі та зазначені вище вимоги програмний, продукт повинен забезпечувати виконання наступних функцій:

– робота з файлом;

– розрахунок всіх простих, допустимих маршрутів слідування з будь-якої станції в усі можливі;

– розподіл поїздопотоку по маршрутах з урахуванням мінімального загального часу руху, пропускної спроможності колії та кількості поїздів, що слідують з однієї станції в іншу;

– розподіл поїздопотоку по маршрутах з урахуванням мінімальної загальної механічної роботи при слідуванні поїздів, пропускної спроможності колії та кількості поїздів, що слідують з однієї станції в іншу;

– розподіл поїздопотоку по маршрутах, одночасно враховуючи загальний час руху та затрати механічної роботи, при обмеженнях з пропускної спроможності колій та заданій величині поїздопотоку між станціями;

– розрахунок загальної кількості поїздів, що пройде по кожній колії та вивід даної інформації в таблицю;

– відображення даних про сполучення між двома сусідніми станціями в таблицю.

Враховуючи те, що в один файл зберігається інформація, він повинен містити певну послідовність в збережені даних. Спочатку повинні вноситися назви станцій та номера, що присвоюються автоматично, потім дані про сполучення між двома сусідніми станціями та в кінці – дані про поїздопотік між всіма станціями.

Вибір парадигми програмування

Внутрішнє проектування слід починати з вибору парадигми програмування, тобто підходу до написання програм. Найбільш поширеними парадигмами програмування, які до того ж можуть бути використані для розробки проекту, є парадигми процедурного та об’єктно-орієнтованого програмування.

© Чибісов Ю. В., 2011

Об’єктно-орієнтований підхід полягає в наступному наборі основних принципів:

– обчислення здійснюються шляхом взаємодії (обміну даними) між об’єктами, при якому один об’єкт вимагає, щоб інший об’єкт виконав деяку дію. Об’єкти взаємодіють, посилаючи та одержуючи повідомлення. Повідомлення – це запит на виконання дії, доповнений набором аргументів, які можуть знадобитися при виконанні дії;

– кожен об’єкт має незалежну пам’ять, яка складається з інших об’єктів;

– кожен об’єкт є представником (примірником) класу, який висловлює загальні властивості об’єктів;

– у класі задається поведінка (функціональність) об’єкта. Тим самим усі об’єкти, які є екземплярами одного класу, можуть виконувати одні і ті ж дії;

– класи організовані в єдину деревоподібну структуру з загальним корінням, що названа ієрархією наслідування. Пам’ять і поведінка, пов’язані з примірниками певного класу, автоматично доступні будь-якому класу, розташованому нижче в ієрархічному дереві.

Таким чином, програма являє собою набір об’єктів, що мають стан і поведінку. Об’єкти взаємодіють за допомогою повідомлень. Природним чином вибудовується ієрархія об’єктів: програма в цілому – це об’єкт, для виконання своїх функцій вона звертається до об’єктів, які до неї входять, та які, у свою чергу, виконують запит шляхом звернення до інших об’єктів програми. Природно, щоб уникнути нескінченної рекурсії у зверненнях, на якомусь етапі об’єкт трансформує звернене до нього повідомлення в повідомлення до стандартних системних об’єктів, що надаються мовою і середовищем програмування.

Стійкість і керованість системи забезпечується за рахунок чіткого розподілу відповідальності об’єктів (за кожну дію відповідає певний об’єкт), однозначного визначення протоколів міжоб’єктної взаємодії і повної ізольованості внутрішньої структури об’єкта від зовнішнього середовища (інкапсуляції).

Вибір мови програмування

Для вибору мови програмування необхідно з’ясувати особливості розробки програми.

Програма вимагає великої кількості екранних форм і діалогових вікон для введення, перегляду і зміни інформації. Для швидкої розробки інтерфейсу потрібна система, що підтримує візуальне програмування.

Середовище програмування по можливості повинно мати вбудовані засоби налагодження та перевірки помилок на етапі виконання.

Вищевказаним вимогам повною мірою задовольняють два середовища програмування: Borland Delphi та Builder C++, що підтримують мови Object Pascal та C/C++ відповідно.

В якості середовища програмування було обрано Builder C++. Цей вибір обумовлений тим, що це середовище дозволяє створити зручний інтерфейс. Але тільки тим можливості Builder C++ не обмежуються. Ідеологія форм, об’єктно-орієнтований підхід, надзвичайно швидкий компілятор, тісна інтеграція з програмуванням у середовищі Windows і технологія компонентів – це ще один довід використання даного пакета.

Але найважливішою частиною є мова C++, на фундаменті якої будувалося все інше. Builder C++ швидко подає потрібну інформацію. А технологія Code Insight (система підказок для написання коду) допомагає швидко писати програму й уникнути помилок при написанні тих або інших назв елементів інтерфейсу або назв процедур і функцій.

Проектування інтерфейсу користувача

Програма, що розроблюється, повинна мати інтерфейс, який спроектовано під операційну систему Windows. Важливим аспектом, який враховувався під час проектування інтерфейсу, є простота освоєння і запам’ятовування операцій системи.

Основним будівельним блоком інтерфейсу є форма. Будь-яка програма має форму, яка називається головною. Ця форма з’являється на екрані під час запуску програми. Використаємо головну форму для побудови інтерфейсу. У верхній частині форми буде розташовуватись головне меню користувача. При виборі якого-небудь з пунктів меню на формі стають активними відповідні елементи інтерфейсу, що необхідні для реалізації даного пункту меню. Потрібно знати, що користувач після запуску програмного продукту отримує можливість використати програму у своїх розробках, тому затрати на вивчення роботи інтерфейсу програми повинні бути мінімальні та інтерфейс повинен бути легкозасвоюваним.

Виходячи з вищевказаних міркувань, передбачається створення короткого за змістом та легкого за ступенем сприйняття інтерфейсу, що володіє властивістю швидкості досягнення мети та ефективного отримання розв’язку задачі. На рис. 1 зображено граф станів і переходів спроектованого головного меню користувача.

Проектування архітектури програми

Виходячи з того, що задача ділиться на підзадачі, організуємо кожну підзадачу в окремий модуль.

Таким чином маємо наступні модулі:

– головний модуль програми;

– модуль для роботи з даними та файлом;

– модуль для розрахунку простих, допустимих маршрутів;

– модуль для розподілення поїздопотоку по маршрутам.

Структура програми наведена на рис. 1.

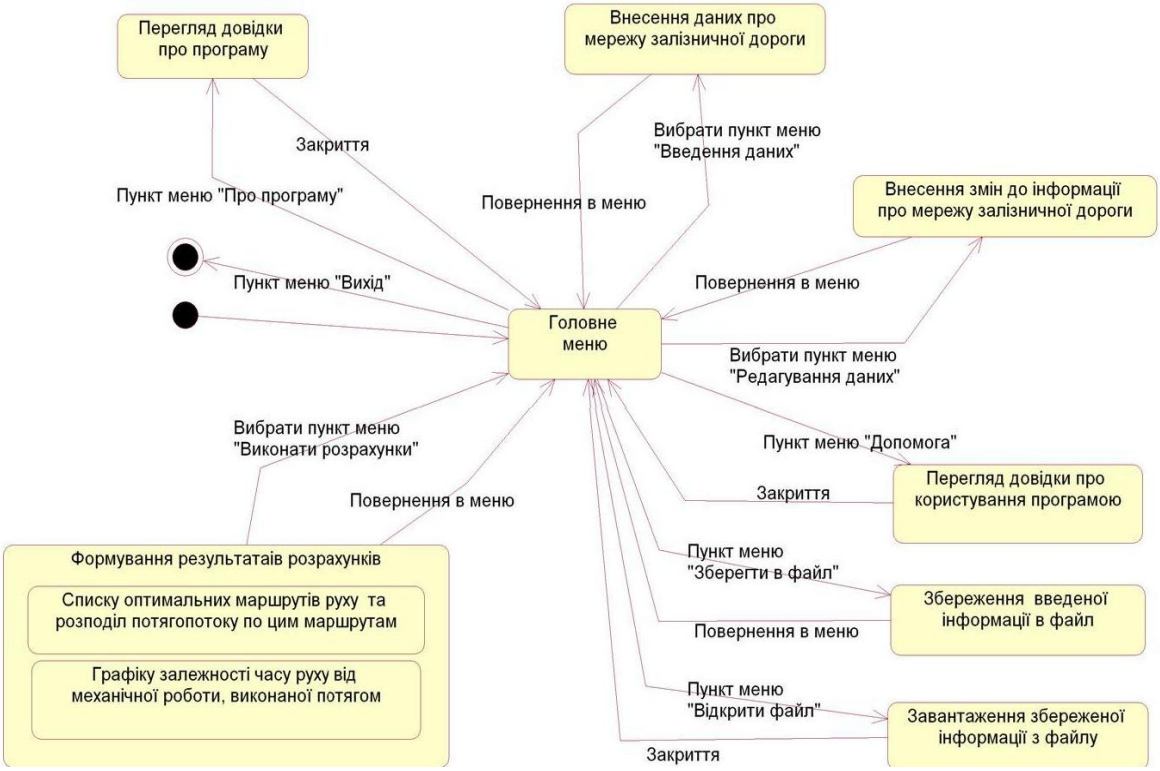

Рис. 1. Граф станів і переходів головного меню користувача

Об’єднаємо функції та процедури, що відповідають за інтерфейс програми, у головному модулі. Крім того в цьому модулі будуть знаходитись глобальні змінні, з якими будуть працювати всі модулі.

Для покращення роботи програми з даними, об’єднаємо всі процедури та функції, що виконують роботу з даними, в один модуль. Цей модуль буде містити такі основні функції:

– зчитування даних з файлу;

– додавання інформації в файл;

– вивід даних на екран, відображення на головній формі;

– редагування даних та їх збереження.

Для розрахунку простих, допустимих маршрутів треба користуватися математичними формулами та методами. Виходячи з цього, виникає необхідність всі процедури та функції, що пов’язані з цією задачею, об’єднати в модуль.

За цією ж причиною виділимо модуль для розподілення поїздопотоку по маршрутам. Реалізація цієї задачі базується на одному математичному методі. Крім того, для складання таблиці завантаження кожної колії необхідні дані, що виникають в результаті розрахунку розподілу поїздопотоку по маршрутам. Взаємозв’язок між модулями програми наведений на рис. 2.

Формалізація вихідних даних та аналіз результатів роботи програми

Результатом роботи став програмний продукт «Розрахунок оптимальних маршрутів руху поїздопотоків», який визначає раціональні маршрути слідування поїздів на заданій мережі залізниці за умов виконання обмежень з пропускної спроможності колій та розподілу заданого поїздопотоку між станціями.

Вихідні дані

Для розв’язання поставленої задачі розрахунку оптимальних маршрутів слідування поїздопотоків необхідні наступні вихідні дані, які вводяться користувачем в діалоговім режимі в таблицю:

– назва станції ділянки залізниці – текстовий формат;

– вибір зі списку введених назв станцій двох суміжних станцій;

– число колій між вибраними станціями – числовий формат;

– напрям руху по колії – текстовий формат;

– середній час руху поїзда по колії (хв.) – числовий формат;

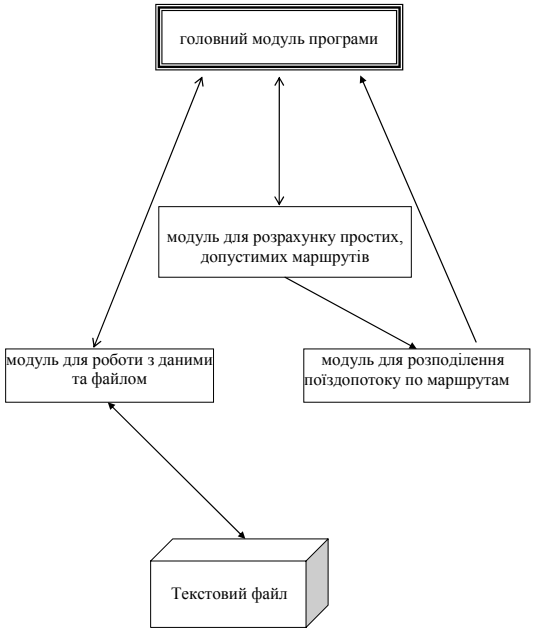

Рис. 2. Взаємозв'язок модулів програми

– пропускна спроможність колії, поїздів за добу – числовий формат;

– затрати механічної роботи при русі поїзду (КДж) – числовий формат;

– поїздопотік між станціями по всій ділянці залізниці (поїздів) – числовий формат.

Результати роботи програми

Результатом роботи програми, тобто даними, які отримає користувач на виході, є:

– список всіх оптимальних маршрутів та значення поїздопотоку, який реалізується на кожному з маршрутів; кожний рядок списку буде містити маршрут у форматі:

«Назва початкової станцій – номер колії – назва проміжної станції – номер колії –…….. – назва кінцевої станції»;

– графік залежності затрат механічної роботи від затрат часу для всієї ділянки.

Розроблений продукт дає можливість швидко розраховувати дані та видає користувачу інформацію про маршрути, які є оптимальними за двома показниками: час руху та затрати механічної роботи при русі по колії, тобто ці показники є якомога меншими.

Всі вихідні дані можна зберігати у файли та в подальшому використовувати їх у програмі з можливістю редагування. Також програма надає користувачу залежність витрат механічної роботи від часу руху поїздів для заданої мережі у вигляді графіка. В подальшому цю інформацію можна використовувати для розрахунку тарифів перевезення вантажів, враховуючи вимоги замовників.

Проаналізуємо роботу програми, рухаючись по схемі, що зображена на рис. 3.

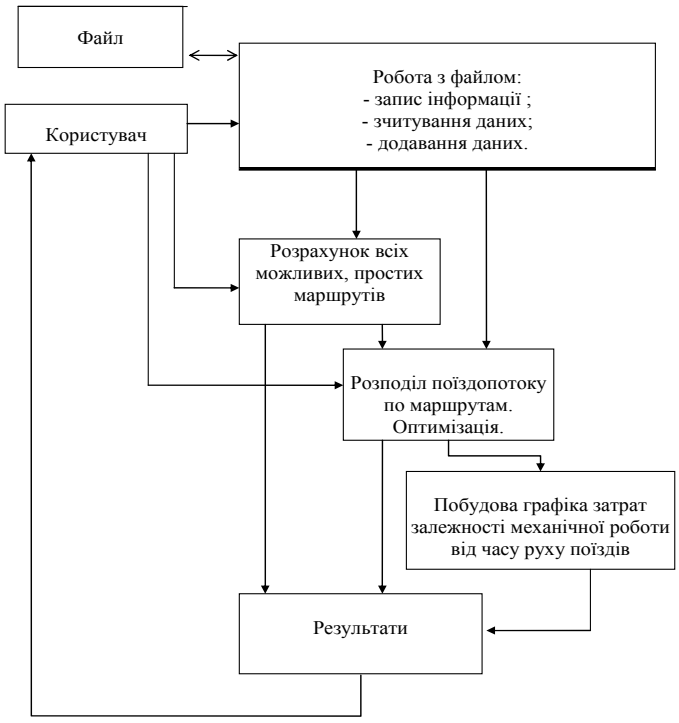

Рис. 3. Порядок роботи з програмним продуктом

За даним рисунком видно, що користувач в результаті отримає оптимальні маршрути слідування поїздів, що розраховується на основі вихідних даних, на прикладі Дніпропетровського вузла (рис. 4), інформації про колії та поїздопотік (рис. 5), та графік залежності витрат механічної роботи від часу руху поїздів. Кінцевий результат програми (рис. 6) – список маршрутів з указаною кількістю поїздів по кожному з них, графічна залежність витрат механічної роботи від часу руху поїздів, а також навантаження на кожну з колій.

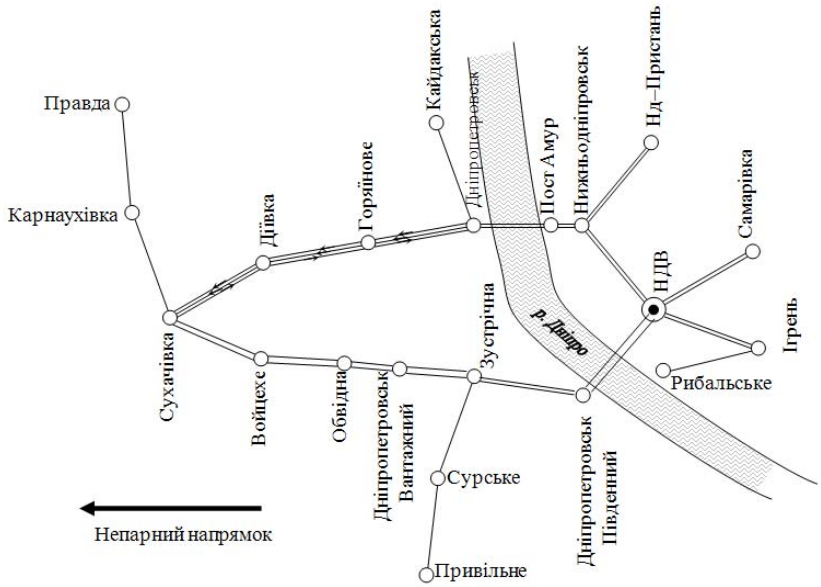

Рис. 4. Схема Дніпропетровського вузла

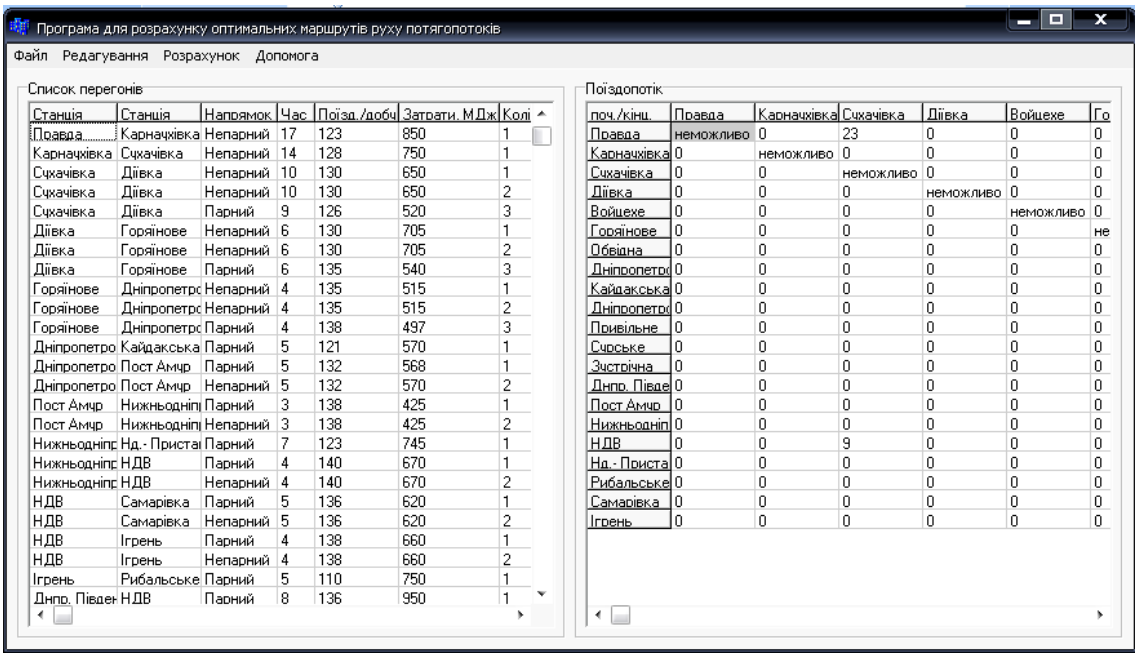

Рис. 5. Головне меню програми. Вихідні дані.

Висновки

В результаті проведеної науково-технічної роботи було розроблено програмне забезпечення, яке призначене для розрахунку оптимальних маршрутів руху поїздопотоків, враховуючи пропускну спроможність колій та заданий поїздопотік на мережі залізниці. Маршрути обираються оптимальними за двома показниками: час руху поїздів та затрати механічної роботи при русі поїзда по колії, ці показники для кожного маршруту одночасно є якомога меншими.

Також користувач має можливість графічного відображення залежності затрат часу від виконаної механічної роботи поїздів в масштабах всієї дільниці.

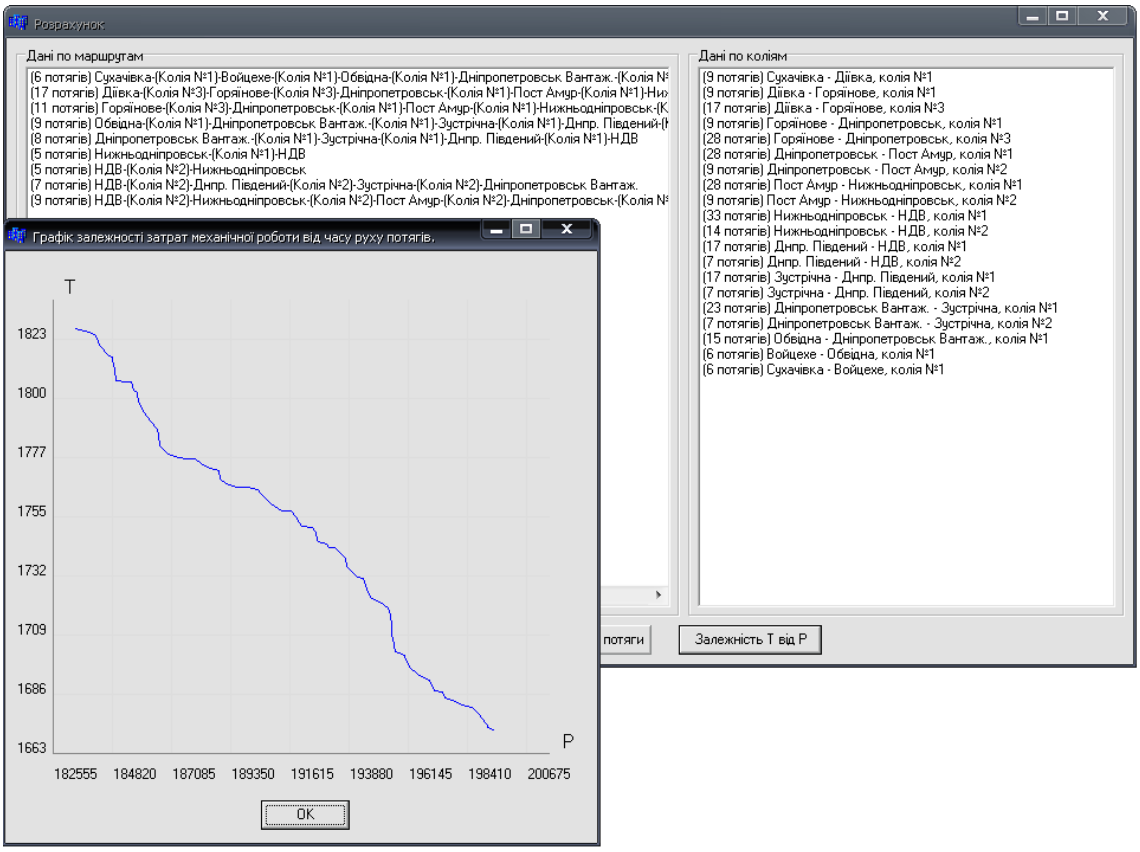

Рис. 6. Результати роботи програми

З економічної точки зору цю інформацію можна використовувати при встановлені тарифів на перевезення вантажів. Для кожного значення часу доставки вантажів, який буде влаштовувати замовника, можливо отримати значення величини виконаної механічної роботи і відповідно встановити тариф. А також розрахувати маршрути руху при заданому часі та композицію составів вантажних поїздів.

БІБЛІОГРАФІЧНИЙ СПИСОК

1. Андерсон, Дж. А. Дискретная математика и комбинаторика [Текст] / Дж. А. Андерсон : [пер. с англ.]. – М.: Изд. дом «Вильямс», 2004. – 960 с.
2. Долгополов, П. В. Побудова моделі корпоративної мережі управління експлуатаційною роботою залізничного вузла [Текст] / П. В. Долгополов // Зб. наук. пр. УкрДАЗТ. – Вип. 62. – Х., 2004.
3. Устич, П. А. Управление транспортом на основе математического моделирования [Текст] / П. А. Устич, А. А. Иванов // Ж/д трансп. – 2008. – № 7. – С. 39-42.
4. Орловский, П. Н. Моделирование на ЭЦВМ процесса пропуска поездов по главному ходу железнодорожного узла [Текст] / П. Н. Орловский // Вопросы механизации и автоматизации сортировочного процесса на железнодорожных станциях. – 1975. – 160/8. – С. 107-115.
5. Апатцев, В. И. Оптимизация работы железнодорожных узлов [Текст] / В. И. Апатцев // Ж/д трансп. – 1998. – № 11. – С. 2-6.
6. Грунтов, П. С. Управление эксплуатационной работой и качеством перевозок на железнодорожном транспорте [Текст] / П. С. Грунтов. – М.: Транспорт, 1994. – 543 с.
7. Кочнев, Ф. П. Организация движения на железнодорожном транспорте [Текст] : учеб. для ВУЗов ж.-д. транспорта / Ф. П. Кочнев – М.: Транспорт, 1963. – 520 с.
8. Соболь, И. М. Выбор оптимальных параметров в задачах со многими критериями [Текст] / И. М. Соболь, Р. Б. Статников. – М.: Наука, 1981. – 210 с.
9. Поденоский, В. В. Парето-оптимальные решения многокритериальных задач [Текст] / В. В. Поденоский, В. Д. Ногин. – М.: Наука. Главн. ред. физ.-мат. лит-ры, 1982. – 256 с.
10. Парето-оптимальное моделирование инженерных задач [Текст] / В. И. Седых [и др.] // Комп'ютерний журнал. – 2004. – 22 с.
11. Bosov, A. A. Vector Optimization by Two Objective Functions [Electron. resource] / A. A. Bosov, G. N. Kodola, L. N. Savchenko. – Access mode: http://arxiv.org/pdf/0708.4307v1

Надійшла до редколегії 22.11.2010.
Прийнята до друку 29.11.2010.