Кобильник Т.П., Когут У.П.
Дрогобицький державний педагогічний університет імені Івана Франка

Використання системи Maxima для розв'язування оптимізаційних задач на графах

Деякі оптимізаційні задачі зручно розв'язувати за допомогою графів, зокрема задачі на знаходження мінімального шляху, про максимальний потік, знаходження потоку найменшої вартості, побудова каркасу графа мінімальної вартості. Граф є математичною моделлю різноманітних об'єктів, явищ і процесів, що досліджуються і використовуються в науці, техніці та на практиці. Використання засобів теорії графів у процесі розв'язування оптимізаційних задач дозволяє алгоритмізувати процес пошуку оптимальних рішень [1].

Питанням, пов'язаним з використанням графів для розв'язування оптимізаційних задач, присвячені роботи [2-4]. М.Н. Кірсанов [5] розглядає можливості використання системи комп'ютерної математики (СКМ) Maple для розв'язування задач з теорії графів. У дисертаційній роботі Н.Р. Балик [6] елементи теорії графів розглядаються як засіб:

- формування навичок інформаційного моделювання;
- розвитку алгоритмічного стилю мислення;
- формування пізнавального інтересу до вивчення як інформатичних, так математичних дисциплін.

Для розв'язування задач з теорії графів зручно використовувати СКМ, що містять функції для їх розв'язування. Це значно спрощує дослідження математичних моделей таких задач, оскільки не потрібно програмувати певний алгоритм (наприклад, алгоритм Дейкстри для знаходження найкоротшого шляху), а тільки використати функцію його реалізації, і досліджувати власне модель задачі. Незважаючи на те, що використання СКМ значно спрощує процес розв'язування прикладних задач, ефективне їх використання неможливе без знання математичної термінології та методів розв'язування, здатності передбачати результат, вміння аналізувати і досліджувати отриманий результат.

Розглянемо як приклад можливості використання СКМ Maxima [7] до розв'язування задач з теорії графів.

Спочатку наведемо основи синтаксису системи Maxima. Мінімум, що потрібно для того, щоб почати роботу із системою Maxima в будь-якому розповсюджуваному Linux-дистрибутиві, це пакет maxima. Цей пакет містить насправді мінімум: консольну версію програми з необхідними бібліотеками та кілька демо-файлів. В консольній версії забезпечуються доволі бідні візуальні можливості: всі математичні формули будуються звичайними текстовими символами в кілька рядків дисплею, а зображення графіків відображаються в окремому вікні (причому продовження роботи можливе тільки після його закривання). Проте за рахунок цього різко зменшуються вимоги до технічних характеристик комп'ютера – систему Maxima в консольному варіанті можна використовувати навіть на комп'ютерах з досить скромними характеристиками. Для системи Maxima розроблено кілька графічних інтерфейсів: xmaxima, emaxima, imaxima та інші. Робота в будь-якому з цих інтерфейсів системи Maxima відбувається в напівавтоматичному режимі.

Систему Maxima можна використовувати з усіма сучасними варіантами операційних систем Linux та UNIX, Windows 9x/2000/XP. Розглянемо роботу з системою Maxima з графічним інтерфейсом wxMaxima, який базується на wxWidgets, під управлінням операційної системи Windows.

На початку кожного рядка є деяке позначення комірки цього рядка. Кожен рядок введення в системі Maxima позначається символом (%i) з номером, рядок виведення – (%o) з відповідним номером. У деяких версіях системи Maxima рядок введення позначається (C), рядок виведення – (D) з відповідними номерами. Для повторення раніше введеної команди, наприклад (%i2), досить ввести два апострофи і потім мітку потрібної команди, наприклад (''%i2). Звернутися до останнього обчислення можна за допомогою символу %, до будь-якого попереднього – %on, де n – порядковий номер обчислення. Введення виразу закінчується крапкою з комою (;) або символом $.

У системі Maxima розрізняються регістри введених символів в іменах вбудованих констант та функцій. Запис sin(x) нееквівалентний запису SIN(X). Імена вбудованих функцій задаються малими літерами. Регістр букв також важливий при використанні змінних, наприклад `X` та `x` – різні змінні.

Основні команди та функції системи Maxima містяться у ядрі. В систему Maxima, як і в більшості СКМ, входить також пакети розширень, за рахунок чого збільшуються можливості її використання при розв'язуванні спеціальних задач, зокрема такими є задачі теорії графів.

Для використання команд для роботи з графами попередньо треба звернутися до пакету розширень *graphs* за вказівкою *load*(*graphs*). Наведемо деякі функції з цього пакету.

- *create_graph*(*V*, *E*, *directed*) – створюється граф, що складається з множини вершин *V* та множини ребер *E*. За опцією *directed=true* вказується, що граф є орієнтованим (за замовчуванням *directed=false*, тобто задається неорієнтований граф).
- *print_graph*(*G*) – виводяться відомості про граф *G*: кількість вершин і ребер у графі та вказуються вершини, які зв'язані ребрами з даною.
- *draw_graph*(*G*, *opt*) – подається графічне зображення графу *G* з відповідними опціями побудови (за необхідності): колір та товщина ребер, величина вершин графу, виведення ваг ребер тощо.
- *shortest_weight_path*(A, B, G) – відшуковується найкоротший шлях з вершини A до вершини B у графі *G*. Зауважимо, що граф *G* може бути як орієнтованим, так і неорієнтованим.

Розглянемо процес розв'язування оптимізаційних задач на графах з використанням СКМ Maxima з графічним інтерфейсом wxMaxima під управлінням операційної системи Windows.

Приклад 1. Нехай задано орієнтований граф (див. рис. 1). Знайти найкоротший шлях з вершини A до вершини B [5].

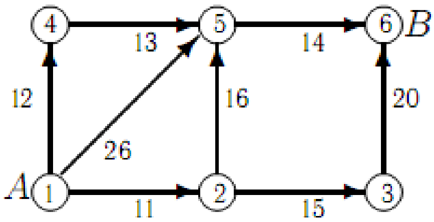


Рис. 1

Задання графу та виведення відомостей про нього зображено на рис. 2.

За функцією знаходимо мінімальну відстань від вершини 1 до вершини 6 (що дорівнює 39) та шлях, що їй відповідає (що проходить через вершини 1,4,5,6 в заданому порядку).

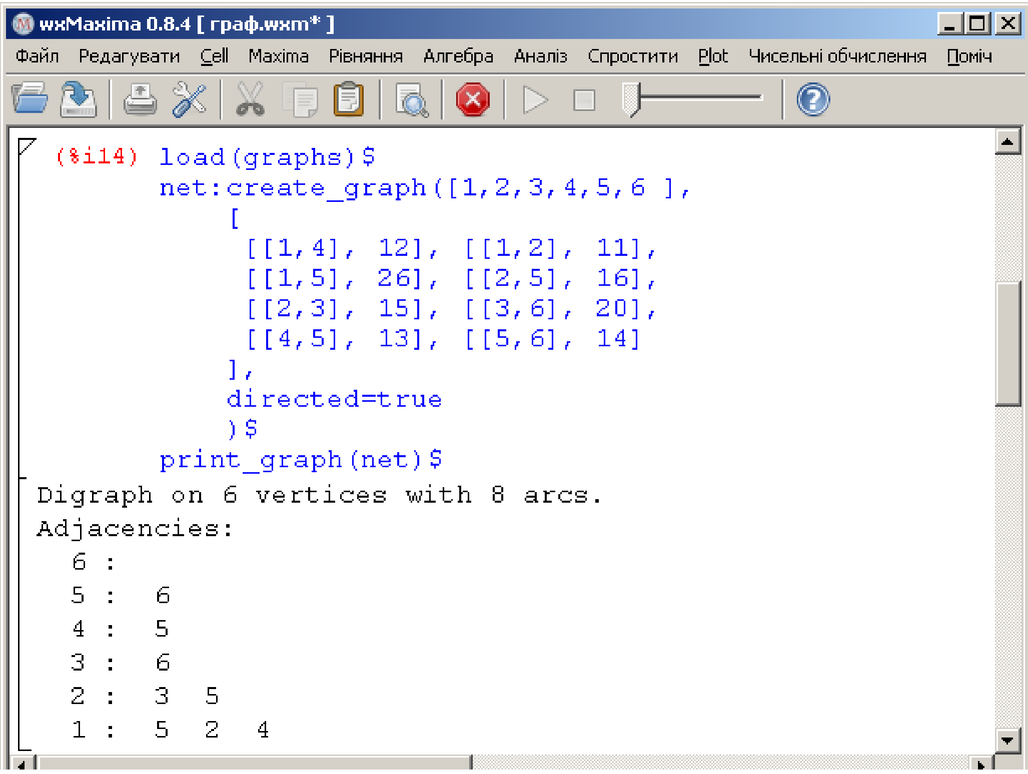


Рис. 2

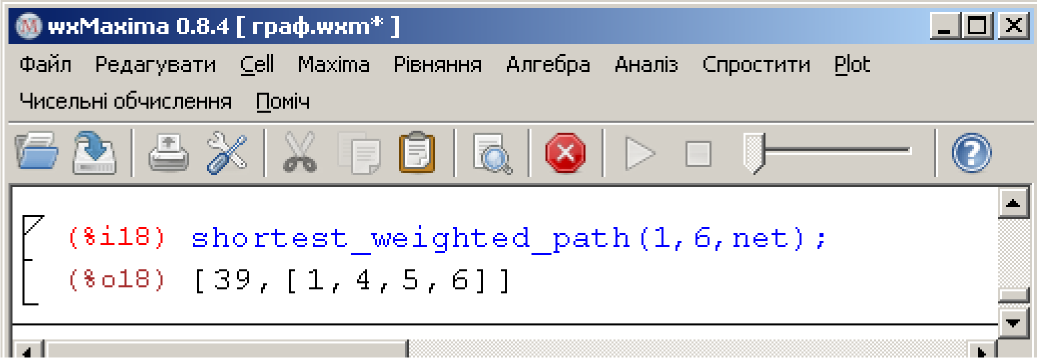


Рис. 3

Після ознайомлення з основними функціями системи Maxima для розв'язування задач з теорії графів студентам пропонуються завдання, які зводяться до побудови та дослідження графів. Це можуть бути такі задачі.

Приклад 2. Розглянемо дослідження задачі про розміщення за допомогою моделювання [8].

Постановка задачі. Припустимо, що є система з n населених пунктів і доріг, що їх з'єднують. Розмірами населених пунктів можна знехтувати, зображуючи їх точками; між населеними пунктами задані відстані (дорогою). Потрібно оптимально в цій системі розмістити школу.

Вивчення властивостей. Спершу треба визначити, що означає оптимальне розміщення. Очевидно, що початкових відомостей для розв'язування задачі недостатньо – потрібно ще знати, скільки учнів живе в кожному пункті. Нехай відомо такі дані: p_i, $i = 1, 2, \ldots, n$ – кількість учнів в i-ому пункті. Припустимо, що існує всього два населених пункти: селище, де живе 100 учнів, та віддалений хутір, де живе 2 учні. Очевидно, зсувати школу у бік хутора було б неправильно, потрібно мінімізувати суму учне-кілометрів.

Школу потрібно розміщувати в населеному пункті. Це твердження за необхідності можна запропонувати студентам довести.

Для формального розв'язування цієї задачі використовується теорія графів: населені пункти вважатимуться вершинами графу, ребра – дорогами, що з'єднують населені пункти. Школу потрібно розміщувати у вершині графу. Це означає, що потрібно вибрати місце для школи не з нескінченної кількості точок на площині, а з n точок, що робить повний перебір легкоздійсненним з використанням певної мови програмування або СКМ. Для простоти розглянемо випадок, коли $n = 10$.

Граф системи продемонстровано на рис.4.

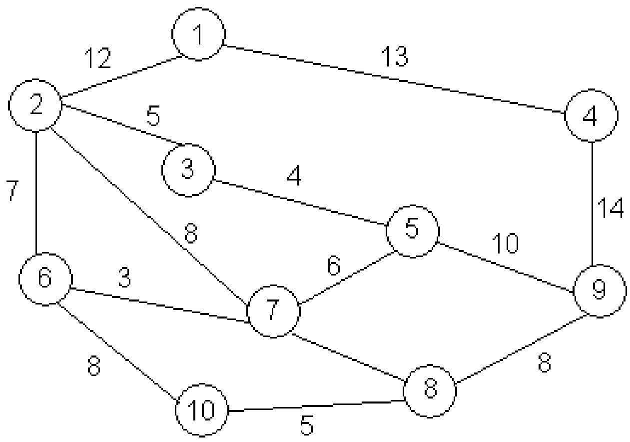


Рис. 4

У середині кола, що зображає вершину, стоїть номер населеного пункту. Кількість учнів у вершинах $i = 1, \ldots 10$ задано відповідно числами $p = \{80, 40, 65, 100, 74, 90, 56, 34, 120, 23\}$. Ребра позначені числами, що характеризують відстані між відповідними населеними пунктами.

Насамперед знайдемо найкоротші ланцюжки з кожної вершини в кожну іншу вершину за допомогою якогось з відомих алгоритмів (наприклад, алгоритм Дейкстри або засобами динамічного програмування). Отримаємо результат, записаний у матриці. Нехай d_{ij} позначає мінімальний шлях між вершинами i та j.

$$\begin{pmatrix} 0 & 12 & 17 & 13 & 21 & 19 & 20 & 29 & 27 & 27 \\ 12 & 0 & 5 & 25 & 9 & 7 & 8 & 17 & 19 & 15 \\ 17 & 5 & 0 & 28 & 4 & 12 & 10 & 19 & 14 & 20 \\ 13 & 25 & 28 & 0 & 24 & 32 & 30 & 22 & 14 & 27 \\ 21 & 9 & 4 & 24 & 0 & 9 & 6 & 15 & 10 & 17 \\ 19 & 7 & 12 & 32 & 9 & 0 & 3 & 12 & 19 & 8 \\ 20 & 8 & 10 & 30 & 6 & 3 & 0 & 9 & 16 & 11 \\ 29 & 17 & 19 & 22 & 15 & 12 & 9 & 0 & 8 & 5 \\ 27 & 19 & 14 & 14 & 10 & 19 & 16 & 8 & 0 & 13 \\ 27 & 15 & 20 & 27 & 17 & 8 & 11 & 5 & 13 & 0 \end{pmatrix}$$

Якщо поставити школу у вершині i, то загальна кількість учне-кілометрів дорівнюватиме сумі добутків i-го рядка матриці на відповідні числа масиву p (скалярному добутку i-го рядка матриці на вектор p). Перебравши всі вершини, знайдемо мінімум цього добутку. У вершині, де досягається мінімум, і потрібно розмістити школу.

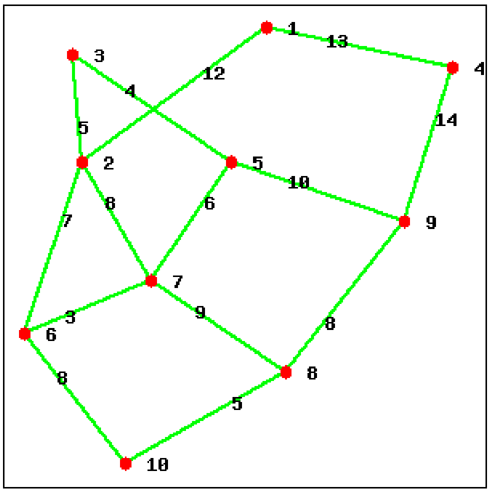
При цьому студенти зіштовхуються з проблемою недостатніх математичних знань, вмінь та навичок, наслідком чого є неефективність використання математичного апарату під час розв'язування таких задач, зокрема при побудові та дослідженні різноманітних моделей. Наприклад, у попередній задачі студент може не вміти використати алгоритм Дейкстри або засоби динамічного програмування для знаходження найкоротших ланцюжків з кожної вершини графа в кожну іншу вершину. У такому випадку він може використати СКМ (наприклад, Maxima) для знаходження матриці найкоротших шляхів.

Задамо відповідний граф та виведемо відомості про нього:

```
(%i1) load(graphs)$
      g:create_graph([1,2,3,4,5,6,7,8,9,10], [[[1,2],12],[[1,4],13],
      [[2,6],7],[[2,7],8],[[2,3],5],[[3,5],4],[[4,9],14],[[5,7],6],
      [[5,9],10],[[6,7],3],[[6,10],8],[[7,8],9],[[8,9],8],[[8,10],5]])$
      print_graph(g)$
Graph on 10 vertices with 14 edges.
Adjacencies:
 10 :  8  6
  9 :  8  5  4
  8 : 10  9  7
  7 :  8  6  5  2
  6 : 10  7  2
  5 :  9  7  3
  4 :  9  1
  3 :  5  2
  2 :  3  7  6  1
  1 :  4  2
```

Подамо графічну ілюстрацію заданого графу з вказанням вершин та ваг ребер

```
(%i4) draw_graph(g, show_vertex_size=2,
      edge_color=green,
      show_weight=true,
      edge_width=2,
      show_id=true,
      text_color=brown)$
```

(%t4) 

Складемо програму, за допомогою якої реалізується повний перебір всіх можливих варіантів і вибирається той, який задовольняє умові задачі:

```
(%i5) p:[80,40,65,100,74,90,56,34,120,23]$
      n:10$
      mn:10^4$
      for i thru n do
      (s:0,
      for j thru n do
      (g1:shortest_weighted_path(i,j,g),
      s:s+g1[1]*p[j]),
      if s<mn then (mn:s,k:i))$
      print("школа розміщується у населеному пункті №",k)$
  школа розміщується у населеному пункті № 5
```

Звідси видно, що школу потрібно розташовувати у населеному пункті з номером 5.

При вивченні питання про знаходження найкоротшого шляху в мережі студентам можна запропонувати таке завдання. Студент щоденно (крім вихідних) ходить до університету. Він визначив найкоротший шлях з дому до університету. Проте на цьому шляху він зустрічає друзів і з ними кілька хвилин спілкується. Таким чином, найкоротший шлях виявився не найшвидшим. Тому студент хоче визначити новий маршрут, на якому він би мав найбільшу ймовірність не зустріти своїх друзів. Схема мережі доріг, якими студент може потрапити з дому до університету, показана на рис. 1. На цій же схемі наведені ймовірності *не зустріти друзів* для кожного сегмента мережі доріг. Ймовірність не зустріти друзів дорівнює добутку ймовірностей на кожному сегменті вибраного шляху. Студенту необхідно розв’язати задачу вибору маршруту, який би максимізував ймовірність не зустріти друзів.

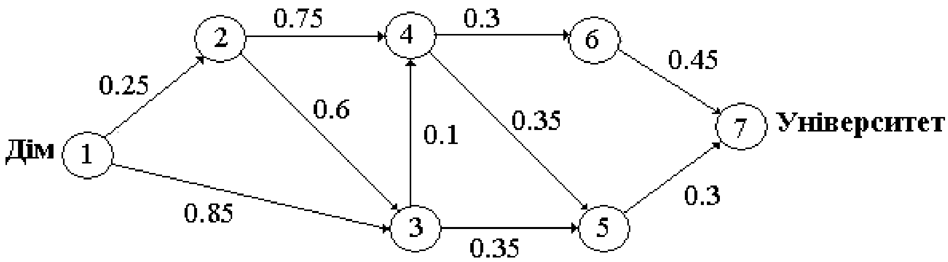


Рис. 5. Схема мережі доріг з дому до університету

Головне завдання студента полягає у формулюванні даної задачі як задачі на знаходження найкоротшого шляху. Далі використати СКМ Maxima для знаходження цього шляху.

При розв’язуванні оптимізаційних задач на графах реалізовуються міжпредметні зв’язки інформатичних, математичних, економічних та інших дисциплін, що сприяє інтелектуальному розвитку студентів на основі формування уявлень про цілісність знань, забезпечує формування навичок володіння не тільки декларативними, але й процедурними знаннями. Використання теорії графів до розв’язування задач формує у студентів вміння подавати умови задачі мовою теорії графів, а потім інтерпретувати отриманий розв’язок в термінах початкової задачі.

Можливості використання системи Maxima для розв’язування задач з теорії графів значні. Студент, використовуючи СКМ Maxima, розв’язує поставлену перед ним задачу, і таким чином у нього не виникає психологічного бар’єру у застосуванні математичного апарату, а крім того він також усвідомлює, який матеріал треба повторити (або вивчити). Розв’язування задач прикладного характеру (такими є оптимізаційні задачі на графах) з використанням СКМ надає знанням і вмінням студентів практично значущого характеру. Цікавими також є дослідження задач з методів оптимізації, зокрема реалізації чисельних методів як умовної, так і безумовної оптимізації засобами СКМ Maxima.

Література

1. Оре О. Теория графов / О. Оре. – М.: Наука, 1980. – 408 с.
2. Воденин Д. Р. Оптимизационные задачи на графах: Учебно-методическое пособ. для студ.экон.эфак./ Д.Р. Воеводин. – Ульяновск: УлГУ.Мех.-мат.фак,1999. – 72 с.
3. Новиков Ф.А. Дискретная математика для программистов / Ф.А. Новиков. – СПб.: Питер, 2005. – 364 с.
4. Таха Х. А. Введение в исследование операций / Хемди А. Таха; пер. с англ. – [7-е издание]. – М. : Издательский дом „Вильямс”, 2005. – 912 с.
5. Кирсанов М.Н. Графы в Maple. Задачи, алгоритмы, программы / М.Н. Кирсанов. – М.: Издательство ФИЗМАТЛИТ, 2007. – 168 с.
6. Балик Н. Р. Методика вивчення експертних систем у курсі інформатики та обчислювальної техніки : дис... канд. пед. наук: 13.00.02 / Балик Надія Романівна; УДПУ імені М.П. Драгоманова. – К, 1995. – 191 с.
7. Семеріков С. О. Maxima 5.13: довідник користувача / С.О. Семеріков; за ред. академіка М. І. Жалдака. – Київ, 2007. – 48 с.
8. Глибовець М. М. Штучний інтелект: підруч. [для студ. вищ. навч. закладів, що навчаються за спец. „Комп’ютерні науки” та „Приклад. математика”] / М. М. Глибовець, О. В. Олецький – К. : Вид. дім „КМ Академія”, 2002. – 366 с.