

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДВНЗ «УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ»**

**Факультет інформаційних технологій
Кафедра інформаційних управляючих систем і технологій**

**МЕРЕЖНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ
Методичні рекомендації до вивчення курсу**

УЖГОРОД – 2014

Мережні інформаційні технології: методичні рекомендації до вивчення курсу для студентів спеціальності 8.05010101 “Інформаційні управляючі системи та технології” та 8.05010301 “Програмне забезпечення систем”

У методичних рекомендаціях до курсу «Мережні інформаційні технології» розглянуто спосіб розв'язання задач математичного програмування за допомогою мережевого NEOS-сервера та мови моделювання AMPL. Наведено приклади задач лінійного, цілочислового та нелінійного програмування. Значна увага приділена транспортній задачі. Розглянуті приклади можуть бути основою для розробки студентами власних програмних проєктів, пов'язаних з оптимізацією.

Розробники: Стецюк П. І., Міца О. В, Пецко В. І.

Рецензенти:

- Нурмінський Є.О., д.ф.-м.н., професор, провідний науковий співробітник Далекосхідного федерального університету (м. Владивосток, Росія)
- Гече Ф.Е., д.т.н., професор, зав. кафедри кібернетики і прикладної математики ДВНЗ “Ужгородський національний університет”

Затверджено рішенням Вченої ради факультету інформаційних технологій від 15.05.2014 р. протокол №5

ЗМІСТ

Вступ	4
I. NEOS-сервер та NEOS-солвер як інтерфейс для розв'язання оптимізаційних задач	7
II. Мова моделювання AMPL як засіб опису задач математичного програмування.	11
2.1. Основні особливості програмування на AMPL.....	11
2.2. Декларації компонент моделі	19
III. Задачі математичного програмування та їх AMPL-код....	27
3.1. Задача лінійного програмування	27
3.2. Задача цілочислового лінійного програмування	32
3.3. Задача нелінійного програмування.....	36
3.4. Транспортна задача	40
IV. Дві ЛП-задачі з булевими змінними для відмовостійкої мережі.....	61
Література	65

ВСТУП

Вибіркова навчальна дисципліна «Мережеві інформаційні технології» є фундаментальною дисципліною освітньо- професійної програми підготовки фахівців за освітньо-кваліфікаційним рівнем «магістр» за спеціальностями 8.05010101 «Інформаційні управляючі системи та технології» та 8.05010301 «Програмне забезпечення систем».

Метою дисципліни «Мережні інформаційні технології» є навчити студентів факультету інформаційних технологій Ужгородського Національного Університету способам опису задач математичного програмування на мові моделювання AMPL та дистанційному способу розв'язання транспортних задач, задач лінійного та нелінійного програмування за допомогою програм NEOS-сервера.

Завдання навчальної дисципліни «Мережні інформаційні технології» є розвиток аналітичного мислення у студентів, вміння застосовувати одержані знання в процесі побудови та аналізу алгоритмів розв'язання екстремальних задач.

За результатами проведення лекцій з курсу «Мережні інформаційні технології» студенти повинні:

Знати – загальні формулювання задач лінійного та нелінійного програмування, способи їх опису за допомогою мови моделювання AMPL, найбільш відомі програми для розв'язання задач лінійного програмування, задач цілочислового лінійного програмування та задач нелінійного програмування.

Вміти – надсилати AMPL-коди для задач математичного програмування на відповідні програми із NEOS-солвера, слідкувати за процесом розв'язання задач, отримувати та аналізувати розв'язки, знайдені програмами gurobi та MINOS.

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Тема 1. NEOS-сервер та NEOS-солвер як інтерфейс для розв'язання оптимізаційних задач. (2 год.)

1. NEOS-сервер, його призначення та основні компоненти.
2. Ознайомлення із NEOS-солвером та його оптимізаційними програмами.

[1], [2].

Тема 2. Мова моделювання AMPL як засіб опису задач математичного програмування. (2 год.)

1. Ознайомлення з мовою моделювання AMPL.
2. Основні групи операторів для опису задач математичного програмування в AMPL (param, var, maximize, minimize, subject to, data, solve, display).

[3], [4].

Тема 3. Задачі лінійного програмування та їх розв'язання за допомогою NEOS-солвера. (2 год.)

1. Задача лінійного програмування. Її формулювання. Симплекс метод та методи внутрішніх точок.
2. Програми NEOS-солвера для розв'язання задач лінійного програмування.

[5], [6], [2].

Тема 4. Програма gurobi як сучасний засіб розв'язання ЛП-задач великих розмірів. (2 год.)

1. Програма gurobi як сучасний засіб розв'язання ЛП-задач великих розмірів.
2. Характеристика роботи програми gurobi для структурованих ЛП-задач з сотнями тисяч змінних та обмежень.

[7],[2].

Тема 5. Задачі цілочислового лінійного програмування та програми NEOS-солвера. (2 год.)

1. Задача цілочислового лінійного програмування. Типи змінних integer та boolean в AMPL.

2. Програми NEOS-солвера для задач цілочислового лінійного програмування.

[3], [11].

Тема 6. Задачі нелінійного програмування та їх розв'язання за допомогою NEOS-солвера. (2 год.)

1. Задача нелінійного програмування. Її формулювання.

2. Програми NEOS-солвера для розв'язання задач нелінійного програмування.

[9], [2].

Тема 7. Програма MINOS – засіб розв'язання задач лінійного та нелінійного програмування. (2 год.)

1. Програма MINOS для розв'язання задач нелінійного програмування.

2. Особливості використання програми MINOS для задач великих розмірів.

[8], [2]

Тема 8. Транспортна матрична задача та її AMPL-код. (2 год.)

1. Транспортна задача. Метод потенціалів.

2. Можливості AMPL для опису транспортних задач.

[3], [8], [10], [11].

I. NEOS-сервер та NEOS-солвер як інтерфейс для розв'язання оптимізаційних задач.

NEOS(Network-Enabled Optimization System)-сервер дозволяє вирішувати в онлайн-режимі задачі математичного програмування. Математичну модель задачі можна описати на мові моделювання GAMS (<http://www.gams.com/>) або AMPL (<http://www.ampl.com>) і передати її через Web інтерфейс NEOS-програми.

Набираємо <http://www.neos-server.org/neos/> і отримаємо наступну сторінку NEOS-сервера (мал. 1.), на якій переходимо до NEOS-солверів по ссилці [NEOS Solvers](#) (мал. 2).

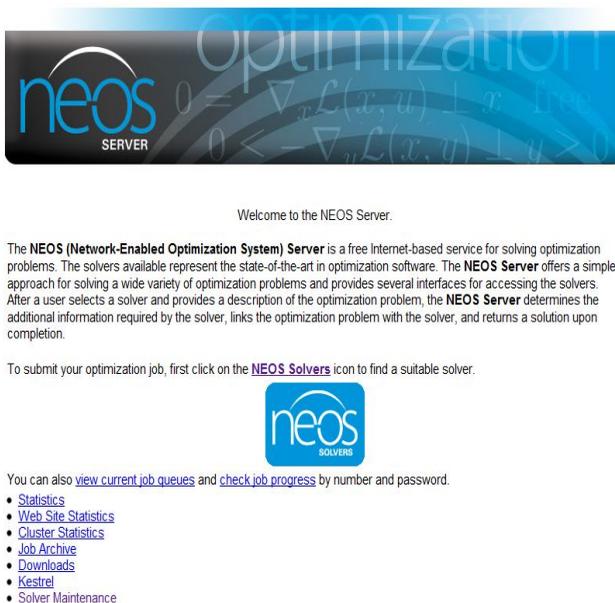


Рис. 1.1. Сторінка NEOS-сервера



Listed below are the available solvers organized by problem type. If you need help in selecting a solver, consult the [Optimization Tree](#) of the NEOS Guide. The choice of solver then determines the available input options for defining the optimization problem.

Each solver has sample problems and background information on the solver. Be sure to submit a sample problem to get a feel for how to submit optimization problems to NEOS. If you encounter problems, consult the [NEOS Server FAQ](#), or contact us by clicking on the [Comments and Questions](#) link at the bottom of the page.

- Administration

- [view job queue](#)
- [view job status/results](#)
- [kill/queue job](#)

- Bound Constrained Optimization

BLMVM ([C input](#))([Fortran input](#))
 L-BFGS-B ([AMPL input](#))
 TRON ([FORTRAN input](#))

- Combinatorial Optimization and Integer Programming

BiqMac ([SPARSE input](#))
 concorde ([TSP input](#))

- Complementarity Problems

filterMPEC ([AMPL input](#))
 KNITRO ([AMPL input](#))
 MILES ([GAMS input](#))
 NLPEC ([GAMS input](#))
 PATH ([AMPL input](#))([GAMS input](#))

- Global Optimization

ASA ([AMPL input](#))
 BARON ([GAMS input](#))
 loos ([AMPL input](#))
 LINDOGlobal ([GAMS input](#))
 PGAPack ([AMPL input](#))
 Pswarm ([AMPL input](#))
 scip ([AMPL input](#))([CPLEX input](#))([GAMS input](#))([MPS input](#))([OSIL input](#))([ZIMPL input](#))

- Linear Network Programming

- Linear Programming

BDMLP ([GAMS input](#))
 bopnd ([AMPL input](#))([LP input](#))([MPS input](#))([QPS input](#))
 Cjp ([MPS input](#))
 Gurobi ([AMPL input](#))([GAMS input](#))([MPS input](#))
 MOSEK ([AMPL input](#))([GAMS input](#))([MPS input](#))
 COOP ([AMPL input](#))
 XpressMP ([AMPL input](#))([GAMS input](#))([MOSEL input](#))([MPS input](#))

Рис. 1.2. Стартова сторінка NEOS-солвера

За допомогою NEOS-солверів можна розв'язувати різноманітні оптимізаційні задачі: задачі лінійного програмування, задачі цілочислового лінійного програмування, задачі нелінійної оптимізації з обмеженнями та багато інших. Для задач лінійного програмування використовуються такі програми:

- BDMLP [[GAMS Input](#)]
- bmpnd [[AMPL Input](#)][[LP Input](#)][[MPS Input](#)][[QPS Input](#)]
- Clp [[MPS Input](#)]
- Gurobi [[AMPL Input](#)][[GAMS Input](#)][[MPS Input](#)]
- MOSEK [[AMPL Input](#)][[GAMS Input](#)][[MPS Input](#)]
- OOQP [[AMPL Input](#)]
- XpressMP [[AMPL Input](#)][[GAMS Input](#)][[MOSEL Input](#)][[MPS Input](#)]

Для розв'язання за допомогою Gurobi на мові моделювання AMPL вибираємо відповідну посилку Gurobi [[AMPL Input](#)] і на відкритій сторінці загрузаємо файл моделі **Model File**, файл даних **Data File**, файл команд **Commands File** і вводимо адресу електронної пошти, на яку має прийти результат(результат також з'явиться на даній сторінці, після закінчення обчислень).

Для задач змішаного цілочислового лінійного програмування використовуються такі програми:

- Cbc [[AMPL Input](#)][[GAMS Input](#)][[MPS Input](#)]
- feaspump [[AMPL Input](#)][[CPLEX Input](#)][[MPS Input](#)]
- Gurobi [[AMPL Input](#)][[GAMS Input](#)][[MPS Input](#)]
- MINTO [[AMPL Input](#)]
- MOSEK [[GAMS Input](#)]
- proxy [[CPLEX Input](#)][[MPS Input](#)]
- qsopt_ex [[AMPL Input](#)][[LP Input](#)][[MPS Input](#)]
- scip [[AMPL Input](#)][[CPLEX Input](#)][[GAMS Input](#)][[MPS Input](#)][[OSIL Input](#)][[ZIMPL Input](#)]

- SYMPHONY [[MPS Input](#)]
- XpressMP [[AMPL Input](#)][[GAMS Input](#)][[MOSEL Input](#)][[MPS Input](#)]

Для задач нелінійного програмування з обмеженнями використовуються такі програми:

- CONOPT [[AMPL Input](#)][[GAMS Input](#)]
- filter [[AMPL Input](#)]
- Ipopt [[AMPL Input](#)][[GAMS Input](#)]
- KNITRO [[AMPL Input](#)][[GAMS Input](#)]
- LANCELOT [[AMPL Input](#)]
- LOQO [[AMPL Input](#)]
- LRAMBO [[C Input](#)]
- MINOS [[AMPL Input](#)][[GAMS Input](#)]
- MOSEK [[AMPL Input](#)][[GAMS Input](#)]
- PATHNLP [[GAMS Input](#)]
- SNOPT [[AMPL Input](#)][[GAMS Input](#)]

Розв'язання задач даних типів здійснюється так же, як і в попередньому випадку.

II. Алгебраїчна мова моделювання AMPL

У розділі наведено короткий опис мови моделювання AMPL на основі препринта [3].

2.1. Основні особливості програмування на AMPL

AMPL (A Mathematical Programming Language) [2] – це мова високого рівня для опису задач математичного програмування, що використовує декларативно-алгебраїчний стиль представлення моделей математичного програмування, близьке до традиційної математичної нотації. Разом з тим AMPL дає можливість описати і складні моделі оптимізації з різними логічними умовами, з використанням складних систем індексації змінних і обмежень. AMPL дозволяє задати модель математичного програмування незалежно від даних, що використовуються для конкретного прикладу моделі. У даному розділі наведені лише базові відомості про AMPL, які можуть знадобитися для опису моделей і виконання лабораторних робіт цього курсу.

AMPL вимагає також завдання вхідних даних (оснащення моделі). Модель і один (або більше) файлів даних направляються в систему AMPL. AMPL працює подібно компілятору: модель і дані з'єднуються в проміжний файл, який передається вирішувачу. Вирішувач фактично знаходить оптимальне рішення задачі, використовуючи проміжний файл, побудований AMPL, і застосовуючи відповідний алгоритм. Вирішувач видає рішення у вигляді текстового файлу.

Робота з AMPL з командного рядка

Моделі записуються у вигляді текстового файлу *<назва файлу>.mod*. При написанні моделі мовою AMPL можна використовувати будь-який текстовий редактор.

При написанні моделей використовуються основні команди AMPL. AMPL-модель містить описання об'єктів

моделі, тобто множин, змінних, параметрів, цільової функції і обмежень. Для опису об'єктів використовуються службові слова **set**, **var**, **param**, **minimize/maximize**, **subject to** (або коротко **s. t.**). При цьому AMPL-модель містить кілька типів елементів, які докладніше описані нижче: декларації з ключовими словами: **set** (множина індексів), **param** (параметр), **var** (змінна), **arc** (дуга – для опису мережевих моделей); цільові функції виду **maximize/minimize**, обмежень, **subject to** (при обмеженнях), **node** (вершина – для опису мережевих моделей).

Багато в чому синтаксис команд AMPL дуже подібний до C. AMPL підтримує такі функції, як **abs()**, **cos()**, **sin()**, **log()**, **sqrt()**, **exp()** з використанням основних операцій **+**, **-**, *****, **/**, **^** чи ******. Всі команди закінчуються крапкою з комою «;». До команд виводу відносяться **display**, а також команди **write** і **print**. До інших корисних елементів відносяться **option** – для зміни опцій AMPL або вирішувача, **include** – читання з окремого файлу, а також **quit** – для виходу з AMPL.

Імена (ідентифікатори) складаються з латинських букв (прописних і малих), цифр і знаків підкреслення. Символ **#** означає початок коментаря. Все, що знаходиться за цим символом ігнорується AMPL. Коментарі можуть бути також обмежені символами **/ * і * /**, причому вони можуть бути відокремлені один від одного кількома рядками.

Команди AMPL використовують простий синтаксис: Змінні описуються з використанням службового слова **var**.

Параметри описуються з використанням службового слова **param**.

Сума $\sum_{i=1}^n$ записується так: **sum {i in 1..n}** .

Службові слова AMPL (такі як **var**, **param**, **solve**, **maximize** та ін), а також імена функцій (наприклад, **sum**, **log**, **sin**) зарезервовані і не можуть використовуватися для імен

об'єктів. До службових зарезервованих слів належать також **for, if, elseif, else, while, file, system**.

Індекси змінних і обмежень вставляються у квадратні дужки (наприклад, a[i]).

Числа можуть записуватися у різних форматах. Так, 0.0123, 1.23D-2, 1.23e-2, 1.23E-2 – це еквівалентні записи одного і того ж числа 0,0123.

Літерали – це рядки, взяті в лапки (одинарні або подвійні). Наприклад, 'abc', 'x', 'y', "ABC".

Команда вигляду

include <назва файлу>

вставляє вказаний файл.

Команди

model; include <назва файлу>

data; include <назва файлу>

можуть бути скорочені до наступних:

model <назва файлу>;

data <назва файлу>;

Команда **commands** аналогічна команді **include**, але це оператор і повинен закінчуватися крапкою з комою.

У AMPL модель і дані розділені. Множини, описані в моделі не мають розмірів або заданих елементів. Використання в моделі множини індексів означає лише те, що використовуються елементи цієї множини, причому неважливо скільки їх є. Задання величин здійснюється тільки у файлі даних, який має вигляд <*назва файлу*>. dat. Тут елементи множин і параметри визначаються явно, повторно записуючи службове слово, назву об'єкта і перераховуючи значення після ": =".

Функціонування моделі та даних у системі моделювання AMPL наведено на рис. 1.

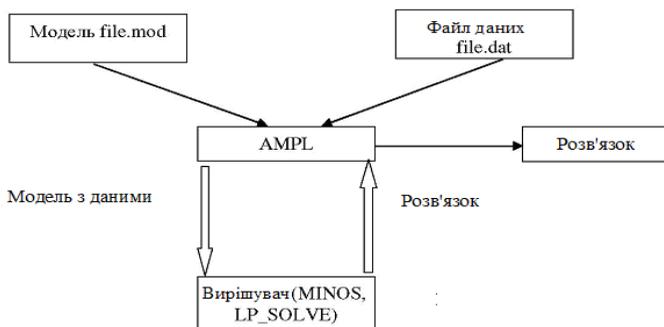


Рис.2.1. Схема роботи системи моделювання AMPL.

Приклад створення моделі AMPL

Розв'язати наступну задачу, використовуючи MINOS.

$$z = x_1 + x_2 \rightarrow \min$$

при обмеженнях

$$x_1 \cdot x_2 \geq 4,$$

$$0 \leq x_1, x_2 \leq 3.$$

Створимо наступний текстовий файл з назвою ex1.mod

```
# ex1.mod – Model
var x1 >= 0, <= 3; # Задання нижньої та верхньої
меж змінної x1
var x2 >= 0, <= 3; # Задання нижньої та верхньої
меж змінної x2
minimize z: x1 + x2; # Цільова функція z
s.t. con1: x1 * x2 >= 4; # Обмеження 1 – му
присвоєно ім'я con1
data; # Задання початкової точки
для вирішувача
var x1 := 1;
var x2 := 1;
```

Потім наберіть **>ampl** (перебуваючи в папці, де знаходиться виконуючий файл `ampl`). З'явиться командне вікно `AMPL` з підказкою **ampl**:

У вікні `AMPL` наберіть наступні команди для розв'язання цього завдання (не забудьте про крапку з комою ";" в кінці кожної команди) за допомогою вирішувача `MINOS` (за замовчуванням).

```
ampl: include ex1.mod;  
ampl: solve;  
ampl: display x1, x2, z;
```

`AMPL` видає наступну інформацію про розв'язок $x_1 = 2$, $x_2 = 2$ з оптимальним значенням цільової функції $z = 4$.

```
MINOS 5.5: optimal solution found.  
15 iterations, objective 4  
Nonlin evals: constrs = 38, Jac = 37.  
x1 = 2  
x2 = 2  
z = 4
```

Для редагування текстового файлу (у зв'язку з виявленими помилками або необхідністю зміни моделі), внесіть необхідні зміни в текстовому файлі моделі і збережіть його. Потім у вікні `AMPL` наберіть наступні команди для введення зміненої моделі:

```
ampl: reset;  
ampl: model ex1.mod;
```

Запишемо на `AMPL` модель цілочисельного лінійного програмування (ЦЛП) вигляду:

$$\sum_{j=1}^n c_j x_j \rightarrow \max$$

при обмеженнях

$$\sum_{j=1}^n a_{ij} x_j \leq b_i, \quad i = 1, \dots, m,$$

$$x_j = 0, 1, \quad j = 1, \dots, n.$$

Опис параметра m в AMPL виглядає так:

```
param m;
```

Аналогічно опишемо параметр n :

```
param n;
```

Опишемо множину індексів $i = 1, \dots, m$ та $j = 1, \dots, n$:

```
set I=1..m;
set J=1..n;
```

Для описання параметрів c_j , b_i , a_{ij} використовуємо запис:

```
param c{J};
param b{I};
param a{I, J};
```

Для опису змінних використовуємо запис:

```
var x{J} binary;
```

який означає, що x_j – бінарна змінна, що приймає тільки значення 0 або 1.

Опис цільової функції $\sum_{j=1}^n c_j x_j \rightarrow \max$ має наступний

вигляд:

```
maximize z: sum {j in 1..n} c[j] * x[j];
```

Обмеження $\sum_{j=1}^n a_{ij}x_j \leq b_i, \quad i = 1, \dots, m$, запишуться так:

```
con {i in 1..m} : sum {j in 1..n} a[i,j] * x[j] <= b[i];
```

Таким чином, модель ЦЛП записується на AMPL у вигляді ip.mod:

```
param m;  
param n;  
set I:=1..m;  
set J:=1..n;  
param c {J};  
param b {I};  
param a {I, J};  
var x {J} binary;  
maximize z: sum {j in J} c[j]*x[j];  
subject to res {i in I} :  
sum {j in J} a[i,j]*x[j]<= b[i];
```

Для розв'язку конкретної задачі ЦЛП вигляду

$$z = 2x_1 + 3x_2 + x_3 \rightarrow \max$$

при обмеженнях

$$x_1 + 2x_2 + x_3 \leq 2,$$

$$2x_1 + 3x_2 + 2x_3 \leq 4$$

$$x_1, x_2, x_3 = 0, 1$$

можна використовувати описаний вище файл моделі ip.mod разом з файлом даних ip.dat наступного вигляду:

```
param m:=2;  
param n:=3;  
param c [1] 2 [2] 3 [3] 1;  
param b [1] 2 [2] 4;  
param a: 1 2 3:=
```

```
1      1  2  1
2      2  3  2;
```

У вікні AMPL набираємо оператори:

```
model ip.mod;
data ip.mod;
option solver lpsolve;
solve;
display x, obj;
```

В результаті отримаємо:

```
LP_SOLVE 4.0.1.0: optimal, objective 3
5 simplex iterations
3 branch & bound nodes: depth 2
x [*] :=
1 1
2 0
3 1
;

obj = 3
```

Зауваження. Запис `option solver lpsolve;` означає явне задання вирішувача завдань задач змішаного цілочисельного лінійного програмування `lp_solve` (<http://lpsolve.sourceforge.net/>).

Загальні синоніми для змінних, обмежень і цільових функцій

У багатьох випадках потрібно роздрукувати або перевірити всі наявні в моделі змінні, обмеження, цільові функції. З цією метою AMPL використовує спільні синоніми всіх компонентів:

_nvars – число змінних в розглянутій моделі;
_ncons – число обмежень у розглянутій моделі;
_nobjs – число цільових функцій в розглянутій моделі;

Відповідно параметри з індексами містять імена всіх компонент моделі AMPL:

_varname {1 .. _nvars} – імена змінних у розглянутій моделі;
_conname {1 .. _ncons} – імена обмежень у розглянутій моделі;
_objname {1 .. _nobjs} – імена цільових функцій в розглянутій моделі;

І, нарешті, є наступні синоніми для компонент:

_var {1 .. _nvars} – синоніми змінних у розглянутій моделі;
_con {1 .. _ncons} – синоніми обмежень у розглянутій моделі;
_obj {1 .. _nobjs} – синоніми цільових функцій в розглянутій моделі.

Графічний інтерфейс AMPL

Існує графічний інтерфейс AMPL, який називається AMPL Studio і доступний на Web-сторінці <http://www.optirisk-systems.com>.

2.2. Декларації компонент моделі

Надалі для формального опису операторів використовуємо **форму Бекуса-Наура** (скор. БНФ, Бекуса-Наура форма) – формальну систему опису синтаксису, в якій одні синтаксичні категорії послідовно визначаються через інші категорії. БНФ-конструкція складається з рядочків (речень) вигляду

<визначуваний символ> ::= <посл.1> | <посл.2> | ... | <посл.n> ,

які описують правила. Таке правило означає, що символ <визначуваний символ> може замінюватися на одну з послідовностей посл.1, посл.2, ..., посл.n. Знак визначення зазвичай виглядає як :: =. Крім того, використовуються квадратні дужки: [A] – елемент A входить або не входить.

Декларації об'єктів моделі AMPL мають наступний загальний вигляд <об'єкт> <назва об'єкта> [<аліас>] [<індексний вираз>] [<тіло>] <об'єкт> :: = set | param | var | arc | minimize | maximize | subject to | node

Декларації параметрів

Параметрами можуть бути скалярні величини, вектори, матриці й масиви відомих даних. Декларація параметра має вигляд:

param <назва_змінної> <аліас> <індексний вираз>
<атрибути>;

<аліас> ::= <літерал>

<індексний вираз> ::= <set_expression_list

<арифметичний_оператор> ::= + | - | less | * | / | mod |

div | ^ | ** |

<унарний оператор> ::= + | -

<оператор редукції> ::= sum | prod | max | min

Необов'язкові атрибути декларацій параметрів можуть розділятися комами, нижче наведено зазначені атрибути:

Атрибут:	Примітка
Binary	Обмежує можливі значення змінної 0 або 1.
Integer	Обмежує можливі значення змінної цілочисельними значеннями

Symbolic	Якщо задано symbolic , то мають також бути задані in sexpr , а також виключаються атрибути, що вимагають задання числових значень, такі як $> = \langle \text{вираз} \rangle$
$> = \langle \text{Вираз} \rangle$	$> =$ Задає нижню межу
$\leq \langle \text{Вираз} \rangle$	\leq Задає верхню межу
$:= \langle \text{Вираз} \rangle$	$:=$ Задає початкове значення
default $\langle \text{вираз} \rangle$	Задає значення за замовчуванням для початкових значень, які можуть бути задані в розділі даних data
$= \langle \text{Вираз} \rangle$	Задає певну змінну

AMPL дозволяє визначати параметри через інші, які вже раніше були визначені. Параметри можуть також обчислюватися і визначатися рекурсивно. Наприклад, для числа комбінацій C_n^m по m із n предметів може бути визначено, використовуючи наступну декларацію

```
param comb {i in 0..n, k in 0..m} :=
  if k = 0 or k = i then 1 else comb[i-1,k-1] + comb[i-1,k];
```

Декларації змінних

Декларації змінних починаються зі службового слова **var**:

```
var <назва змінної> <аліас> <індексний вираз>
<атрибути>,
```

де всі елементи декларації такі ж, як у декларації параметра (див. вище).

Декларації обмежень

Обмеження записуються в наступній загальній формі :

```
<Декларація обмеження> ::=
  [subject to] <назва обмеження> [<аліас>]
  [<індексний вираз>] <атрибути>
  [: <Вираз обмеження>];
```

```
< вираз обмеження > ::= <вираз> <= <вираз> |
< вираз > =<вираз> | <вираз> > =<вираз> |
```

Декларація цільової функції

Декларація (опис) цільової функції має вигляд:

```
< декларація цільової функції > ::= maximize
<назва> [<аліас>] [<індексний вираз>] [: вираз ]; |
minimize <назва > [<аліас >] [<індексний вираз >]
[: вираз ];
```

Огляд команд AMPL

Команда	Коментар
call	Виклик імпортованої функції
cd	Перехід в інший каталог
check	Виконує всі команди check
close	Закриває файл
commands	Читання та інтерпретація команд з файлу
data	Перехід до даних
delete	Видалення компонент моделі
display	Друк компонент моделі і виразів
drop	Виняток обмеження або цільової функції

end	Закінчення введення з поточного файлу введення
environ	Встановити середовище для моделі
exit	Вийти з AMPL зі значенням статусу
expand	Показати детально компоненти моделі
fix	Зафіксувати змінну на її поточному значенні
include	Включити вміст файлу
let	Змінити значення даних
load	Завантажити динамічну бібліотеку
model	Завантажує модель
objective	Вибрати цільову функцію для оптимізації
option	Встановити або видати значення опцій
print	Неформатований друк компонент моделі і виразів
printf	Форматований друк компонент моделі і виразів
problem	Визначити задачу або перейти до задачі
purge	Видалення компонент моделі
quit	Закінчити роботу AMPL
read	Читання з файлу
read table	Читання з таблиці даних
redeclare	Зміна декларації об'єкта
reload	Перезавантаження динамічної бібліотеки функцій
remove	Видалення файлу
reset	Скидання об'єктів, відновлення їх початкового стану
restore	Скасувати дію команди drop

shell	Тимчасовий вихід в операційну систему
show	Показати імена компонент моделі
solexpand	Показати детальне розширення, видиме вирішувачем
solution	Імпортує значення змінної з вирішувача
solve	Конкретний приклад моделі надсилається вирішувачу і повертається знайдене рішення
update	Дозволити зміну даних
unfix	Скасувати дію команди fix
unload	Вивантажити динамічну бібліотеку
write	Видача конкретного прикладу задачі
write table	Записати таблицю в таблицю даних
xref	Показати залежності між компонентами моделі

Файли, що згадуються в командах **include**, **model**, **data**, **commands**, мають прості імена (наприклад, не містять слеш /), шукаються в папках, що задаються опцією **ampl_include**: кожний непорожній рядок **\$ampl_include** задає таку папку; якщо ж **\$ampl_include** порожній або є чистим, файли шукаються в поточній папці.

Перенаправлення виводу

Для перенаправлення виводу у файл, досить додати > і назву файла. Для відкриття вже існуючого файлу і додавання до нього виведення потрібно використовувати >> замість >.

Наприклад:

```
ampl: model ip.mod;
ampl: data ip.dat;
ampl: solve;
```

LP_SOLVE 4.0.1.0: optimal, objective 3
--

```
5 simplex iterations
3 branch & bound nodes: depth 2
```

```
ampl: display x, obj > ip.out;
```

Скрипти та оператори потоку управління

У AMPL передбачені оператори, подібні операторам потоку управління у звичайних мовах програмування, які дають можливість писати програму за допомогою операторів, яка буде виконуватися автоматично.

При роботі з моделлю користувач часто застосовує послідовності одних і тих же команд, які потрібно набирати на екрані. Для прискорення процесу можна записати ці команди в спеціальний скриптовий файл `<file>.run`, викликаючи його потім за допомогою

```
include <file>.run
```

Так, для виклику файлів з прикладу можна записати у файлі `ip.run`

```
model ip.mod;
data ip.dat;
option solver lpsolve;
solve;
display x, z;
```

Для запуску скрипта на екрані AMPL набирається `include ip.run;`

Стандартний формат даних

AMPL підтримує стандартний формат, щоб описати значення множин і параметрів, які об'єднуються з моделлю, щоб отримати конкретну задачу оптимізації.

Розділ даних складається з послідовності токенів (лексем) (літералів і рядків друкованих символів), відділених пробілами, символами табуляції і переведення рядка. Лексеми включають ключові слова, літерали, числа, і розділювачі `() []:: = *`. Затвердження (оператор) –

послідовність лексем, що закінчується крапкою з комою. Коментарі можуть використовуватися так само, як в деклараціях. У всіх випадках розташування даних в чітких рядках і стовпцях – для зручності читання людиною; AMPL ігнорує форматування.

Розділ даних починається з команди `data` і закінчується кінцем-вводу або командою, яка повертає в режим моделювання.

У розділі даних, об'єктам моделі можуть бути призначені значення в будь-якому зручному порядку, незалежному від порядку їх декларування.

Дані для множин

Оператори, що визначають множину, складаються з ключового слова `set` (множина), назви множини, опціональних: `=`, і членів. Одновимірну множину найбільш просто визначити, задаючи список її членів, опціонально відокремлених комами. Одиничні або подвійні лапки символічного рядка можуть бути опущені, якщо рядок алфавітно-цифровий, але не визначає число.

Приклади задання даних для параметрів наведені вище в прикладі для моделі ЦЛП.

Більш докладні відомості про мову алгебраїчного моделювання AMPL можна знайти в книзі [4] і на сайті <http://www.ampl.com>.

III. Задачі математичного програмування та їх AMPL-коди

У розділі наведено різноманітні формулювання задач лінійного [5,6,8], цілочислового та нелінійного [9,10] програмування, способи їх опису за допомогою мови моделювання AMPL, результати розв'язання програмами gurobi [7] та MINOS [8]. Значна увага приділена розв'язанню транспортної задачі у матричній формі [10].

3.1. Задача лінійного програмування

3.1.1. Найпростіша математична модель та її AMPL-код.

Розглянемо таку модель задачі лінійного програмування:

$$z = x_1 + 2x_2 - 3x_3 \rightarrow \min$$

$$\begin{cases} -2x_1 + x_2 + 3x_3 \leq 4, \\ 2x_1 + 3x_2 \leq 6, \\ x_1 + x_3 \leq 2, \end{cases}$$

$$x_j \geq 0, \quad j = \overline{1,3}.$$

AMPL-код для вказаної моделі має наступний вигляд:

```
var x1 >=0;
var x2 >=0;
var x3 >=0;

minimize z: x1+2*x2-3*x3;
s.t. con1: -2*x1+x2+3*x3 <= 4;
s.t. con2: 2*x1+3*x2 <= 6;
s.t. con3: x1+x3 <= 2;

solve;

display z;
display x1, x2, x3;
```

Результат виконання на NEOS-сервері:

Solved in 2 iterations and 0.00 seconds
Optimal objective -4.400000000e+00
Gurobi 5.5.0: optimal solution; objective -4.4
2 simplex iterations

$z = -4.4$
 $x_1 = 0.4$
 $x_2 = 0$
 $x_3 = 1.6$

Результат виконання на AMPL:

MINOS 5.5: optimal solution found.
2 iterations, objective -4.4

$z = -4.4$
 $x_1 = 0.4$
 $x_2 = 0$
 $x_3 = 1.6$

3.1.2. Математична модель загального виду та її AMPL-код. Вона включає 3 змінних та 3 обмеження та має наступний вигляд:

$$Z = c^T x \rightarrow \min \quad (3.1.1)$$

при обмеженнях:

$$Ax \leq b, \quad (3.1.2)$$

$$x \geq 0, \quad (3.1.3)$$

де вектори c , b та матриця A є наступними:

$$c = (1, 2, -3)^T; \quad b = (4, 6, 2)^T; \quad A = \begin{pmatrix} -2 & 1 & 3 \\ 2 & 3 & 0 \\ 1 & 0 & 1 \end{pmatrix}.$$

AMPL-код для моделі (3.1.1)–(3.1.3) має наступний вигляд:

```

param m;
param n;

set I=1..m;
set J= 1..n;

param c{J};
param b{I};
param a{I,J};

var x{J} >=0;

minimize z: sum {j in J} c[j] * x[j];
subject to con{i in I}: sum {j in J} a[i,j] * x[j] <= b[i];

data;
param m:=3;
param n:=3;
param c [1] 1 [2] 2 [3] -3;
param b [1] 4 [2] 6 [3] 2;
param a:   1 2 3:=
           1 -2 1 3
           2  2 3 0
           3  1 0 1;

solve;

display z, x;

```

Результат виконання на NEOS-сервері:

```
Solved in 2 iterations and 0.00 seconds
Optimal objective -4.400000000e+00
Gurobi 5.5.0: optimal solution; objective -4.4
2 simplex iterations
z = -4.4
x [*] :=
1 0.4
2 0
3 1.6
```

Результат виконання на AMPL:

```
CPLEX 12.6.0.0: optimal solution; objective -4.4
2 dual simplex iterations (2 in phase I)
z = -4.4
x [*] :=
1 0.4
2 0
3 1.6
```

3.1.3. Задачі для розв'язання.

$$\begin{array}{ll} z = 2x_1 - x_2 - 3x_3 \rightarrow \max & z = 3x_1 + 2x_2 - 2x_3 + x_4 \rightarrow \min \\ 1. \quad \begin{cases} 2x_1 + x_3 \leq 6, \\ x_1 - x_2 - x_3 = -2, \\ -x_1 + 6x_3 \geq -4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} & 2. \quad \begin{cases} -x_1 + x_2 - x_4 = -5, \\ 2x_2 + x_3 \leq 4, \\ -x_1 - x_2 + 2x_3 \geq -8, \\ x_j \geq 0, \quad j = \overline{1,4}. \end{cases} \end{array}$$

$$z = 2x_1 + x_2 - x_3 \rightarrow \max$$

$$3. \quad \begin{cases} x_1 + x_2 \leq 1, \\ -x_2 - x_3 \geq -2, \\ x_1 + x_3 \leq 2, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = -2x_1 - 3x_2 + x_3 \rightarrow \min$$

$$4. \quad \begin{cases} 2x_1 + x_2 - x_3 = -2, \\ x_1 + 2x_2 \leq 5, \\ 3x_1 - 2x_2 \geq -4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 4x_1 + 3x_2 + 5x_3 \rightarrow \max$$

$$5. \quad \begin{cases} x_1 - x_3 \leq 6, \\ x_1 + x_2 + 2x_3 = 5, \\ -2x_1 + 2x_3 \geq -4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = x_1 + 2x_2 - 3x_3 \rightarrow \min$$

$$6. \quad \begin{cases} -2x_1 + x_2 + 3x_3 \leq 4, \\ 2x_1 + 3x_2 \leq 6, \\ x_1 + x_3 \leq 2, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = x_1 - 5x_2 - x_3 \rightarrow \max$$

$$7. \quad \begin{cases} x_1 + 3x_2 + x_3 \geq 3, \\ 2x_1 + x_3 = 4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 5x_1 + 4x_2 + 2x_3 \rightarrow \min$$

$$8. \quad \begin{cases} x_1 + 2x_2 + x_3 = 5, \\ 2x_1 + 3x_2 + x_3 \geq 8, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 5x_1 + 4x_2 + 2x_3 \rightarrow \min$$

$$9. \quad \begin{cases} x_1 + 2x_2 + x_3 = 5, \\ 2x_1 + 3x_2 + x_3 \geq 8, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 2x_1 + 3x_2 - 5x_3 \rightarrow \min$$

$$10. \quad \begin{cases} x_1 + x_2 + x_3 = 7, \\ 2x_1 - 5x_2 + x_3 \geq 10, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 2x_1 + 4x_2 + 4x_3 \rightarrow \max$$

$$11. \quad \begin{cases} x_1 + x_2 = 4, \\ x_1 + 4x_2 + x_3 < 8, \\ x_j \geq 0, \quad j = \overline{1,4}. \end{cases}$$

$$z = 3x_1 + 2x_2 + 3x_3 \rightarrow \min$$

$$12. \quad \begin{cases} x_1 + 4x_2 + x_3 \geq 8, \\ 2x_1 + x_2 + x_3 = 10, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$\begin{array}{ll}
 z = 3x_1 + 2x_2 + 3x_3 \rightarrow \min & z = x_1 + x_2 + x_3 + x_4 \rightarrow \min \\
 13. \quad \begin{cases} x_1 + 4x_2 + x_3 \geq 8, \\ 2x_1 + x_2 + x_3 = 10, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} & 14. \quad \begin{cases} 2x_1 + 2x_2 + x_3 \geq 7, \\ 4x_1 + 2x_2 - x_3 + x_4 = 8, \\ x_j \geq 0, \quad j = \overline{1,4}. \end{cases}
 \end{array}$$

$$\begin{array}{ll}
 z = x_1 + x_2 \rightarrow \max & z = -3x_1 + 2x_2 + 5x_3 \rightarrow \min \\
 15. \quad \begin{cases} 2x_1 + 3x_2 = 5, \\ 3x_1 + 2x_2 \leq 6, \\ x_j \geq 0, \quad j = \overline{1,2}. \end{cases} & 16. \quad \begin{cases} x_1 - x_2 + 3x_3 \leq 10, \\ x_1 + 3x_2 + 2x_3 = 8, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}
 \end{array}$$

$$\begin{array}{ll}
 z = x_1 - x_2 + 3x_3 \rightarrow \max & z = 3x_1 - 2x_2 - x_3 \rightarrow \min \\
 17. \quad \begin{cases} x_1 + 3x_3 \leq 10, \\ x_1 - x_2 - x_3 = -3, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} & 18. \quad \begin{cases} -x_1 - 3x_2 + 3x_3 \geq -6, \\ -2x_1 + x_2 + x_3 \leq 4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}
 \end{array}$$

$$\begin{array}{ll}
 z = 30x_1 + 30x_2 + 45x_3 \rightarrow \min & z = x_1 + x_2 + x_3 - 4x_4 \rightarrow \max \\
 19. \quad \begin{cases} 3x_1 + x_2 - x_3 \leq 0, \\ -5x_1 + 3x_2 - x_3 \leq 0, \\ x_j \geq 0, \quad j = \overline{1,3} \end{cases} & 20. \quad \begin{cases} x_1 + 4x_2 + x_3 - x_4 < 10, \\ 2x_1 - x_2 + x_3 + x_4 < 10, \\ x_j \geq 0, \quad j = \overline{1,4} \end{cases}
 \end{array}$$

3.2. Задача цілочислового лінійного програмування

3.2.1. Математична модель з цілочисловими та лінійними змінними та її AMPL-код. Розглянемо задачу змішаного лінійно-цілочислового програмування у наступному вигляді:

$$z = x_1 + 2x_2 - 3x_3 \rightarrow \min$$

$$\begin{cases} -2x_1 + x_2 + 3x_3 \leq 4, \\ 2x_1 + 3x_2 \leq 6, \\ x_1 + x_3 \leq 2, \end{cases}$$

$$x_1 - \text{цiле}, x_j \geq 0, \quad j = \overline{1,3}.$$

Вона включає одну цілочислову та дві лінійні змінні.

AMPL-код для вказаної моделі має наступний вигляд:

```
var x1 integer >=0;
var x2 >=0;
var x3 >=0;

minimize z: x1+2*x2-3*x3;
s.t. con1: -2*x1+x2+3*x3 <= 4;
s.t. con2: 2*x1+3*x2 <= 6;
s.t. con3: x1+x3 <= 2;

solve;

display z;
display x1, x2, x3;
```

Результат виконання на NEOS-сервері:

```
Solved in 2 iterations and 0.00 seconds
Optimal objective -4.000000000e+00
Gurobi 5.5.0: optimal solution; objective -4
plus 2 simplex iterations for intbasis
z = -4
x1 = 0
x2 = 0
x3 = 1.33333
```

Результат виконання на AMPL:

```
CPLEX 12.6.0.0: optimal integer solution; objective -4
0 MIP simplex iterations
0 branch-and-bound nodes
z = -4
x1 = 0
x2 = 0
x3 = 1.33333
```

3.2.2. Математична модель з цілочисловими змінними та її AMPL-код. Розглянемо задачу цілочислового лінійного програмування у наступному вигляді:

$$z = x_1 + 2x_2 - 3x_3 \rightarrow \min$$

$$\begin{cases} -2x_1 + x_2 + 3x_3 \leq 4, \\ 2x_1 + 3x_2 \leq 6, \\ x_1 + x_3 \leq 2, \end{cases}$$

$$x_1, x_2, x_3 - \text{цілі}, x_j \geq 0, \quad j = \overline{1,3}.$$

AMPL-код для вказаної моделі має наступний вигляд:

```
var x1 integer >=0;
var x2 integer >=0;
var x3 integer >=0;

minimize z: x1+2*x2-3*x3;
s.t. con1: -2*x1+x2+3*x3 <= 4;
s.t. con2: 2*x1+3*x2 <= 6;
s.t. con3: x1+x3 <= 2;

solve;

display z;
display x1, x2, x3;
```

Результат виконання на NEOS-сервері:

```
Solved in 0 iterations and 0.00 seconds
Optimal objective -3.000000000e+00
Gurobi 5.5.0: optimal solution; objective -3
z = -3
x1 = 0
x2 = 0
x3 = 1
```

Результат виконання на AMPL:

```
CPLEX 12.6.0.0: optimal integer solution; objective -3
0 MIP simplex iterations
0 branch-and-bound nodes
z = -3
x1 = 0
x2 = 0
x3 = 1
```

3.2.3. Задачі для розв'язання.

Ними є задачі підрозділу 3.1.3, в яких перша або усі змінні є цілочисловими. Для них обмеження $x_j \geq 0, j = \overline{1,3}$ потрібно замінити на обмеження $x_1 - \text{цiле}, x_j \geq 0, j = \overline{1,3}$, або на обмеження $x_1, x_2, x_3 - \text{цiлi}, x_j \geq 0, j = \overline{1,3}$. Якщо задача включає чотири змінних, то обмеження $x_j \geq 0, j = \overline{1,4}$ замінюється обмеженням $x_1 - \text{цiле}, x_j \geq 0, j = \overline{1,4}$, або обмеженням $x_1, x_2, x_3, x_4 - \text{цiлi}, x_j \geq 0, j = \overline{1,4}$.

3.3. Задача нелінійного програмування

3.3.1. Математична модель та її AMPL-код. Розглянемо задачу нелінійного програмування у наступному вигляді

$$z = x_1 + 5x_2 + 3x_3 \rightarrow \min$$

$$\begin{cases} 2x_1 + x_2 = 4, \\ x_1 + 2x_2 + x_3 = 6, \\ x_1x_2 \geq 2, \end{cases}$$

$$x_j \geq 0, \quad j = \overline{1,3}.$$

Вона включає лінійну цільову функцію, два лінійних та одне нелінійне обмеження.

AMPL-код для вказаної моделі має наступний вигляд:

```
var x1 >=0;
var x2 >=0;
var x3 >=0;

maximize z: x1+5*x2+3*x3;
s.t. con1: 2*x1+x2 = 4;
s.t. con2: x1+2*x2+x3 = 6;
s.t. con3: x1*x2 >= 2;

solve;

display z;
display x1, x2, x3;
```

Результат виконання на NEOS-сервері:

```
MINOS 5.51: optimal solution found.
0 iterations, objective 14
Nonlin evals: constrs = 16, Jac = 15.
```

$z = 14$
 $x_1 = 1.00006$
 $x_2 = 1.99988$
 $x_3 = 1.00018$

Результат виконання на AMPL:

MINOS 5.5: optimal solution found.
 0 iterations, objective 14
 Nonlin evals: constra = 16, Jac = 15.
 $z = 14$
 $x_1 = 1.00006$
 $x_2 = 1.99988$
 $x_3 = 1.00018$

3.3.2. Задачі для розв'язання

$$z = 2x_1 - 2x_2 - 3x_3 \rightarrow \max \quad z = 3x_1 + 2x_2 - 2x_3 + 2x_4 \rightarrow \min$$

$$1. \quad \begin{cases} 2x_1 + x_3 \leq 6, \\ x_1 - x_2 - x_3 = -2, \\ x_1 x_3 \leq 4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} \quad 2. \quad \begin{cases} -x_1 + x_2 - x_4 = -5, \\ x_2 x_3 \leq 14, \\ -x_1 - x_2 + 2x_3 \geq -8, \\ x_j \geq 0, \quad j = \overline{1,4}. \end{cases}$$

$$z = 2x_1 + x_2 - 2x_3 \rightarrow \max \quad z = -2x_1 - 3x_2 + 2x_3 \rightarrow \min$$

$$3. \quad \begin{cases} x_1 x_2 \leq 1, \\ -x_2 - x_3 \geq -2, \\ x_1 + x_3 \leq 2, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} \quad 4. \quad \begin{cases} 2x_1 + x_2 - x_3 = -2, \\ x_1 x_2 \leq 5, \\ 3x_1 - 2x_2 \geq -4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 4x_1 + 6x_2 + 5x_3 \rightarrow \max \quad z = 3x_1 + 2x_2 - 3x_3 \rightarrow \min$$

$$5. \quad \begin{cases} x_1x_3 \leq 6, \\ x_1 + x_2 + 2x_3 = 5, \\ -2x_1 + 2x_3 \geq -4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} \quad 6. \quad \begin{cases} -2x_1 + x_2 + 3x_3 \leq 4, \\ 2x_1 + 3x_2 \leq 6, \\ x_1x_3 \leq 2, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = x_1 - 5x_2 - 2x_3 \rightarrow \max \quad z = 5x_1 + 4x_2 + 3x_3 \rightarrow \min$$

$$7. \quad \begin{cases} x_1 + 3x_2 + x_3 \leq 13, \\ x_1x_3 \leq 14, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} \quad 8. \quad \begin{cases} x_1x_2x_3 \geq 5, \\ 2x_1 + 3x_2 + x_3 \geq 8, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 5x_1 + 4x_2 + 2x_3 \rightarrow \min \quad z = 2x_1 + 3x_2 - 5x_3 \rightarrow \min$$

$$9. \quad \begin{cases} x_1x_2 \leq 5, \\ 2x_1 + 3x_2 + x_3 \geq 8, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} \quad 10. \quad \begin{cases} x_1x_2x_3 \geq 7, \\ 2x_1 - 5x_2 + x_3 \geq 10, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 2x_1 + 7x_2 + 4x_3 \rightarrow \max \quad z = 3x_1 + 2x_2 + 7x_3 \rightarrow \min$$

$$11. \quad \begin{cases} x_1x_2 = 4, \\ x_1 + 4x_2 + x_3 < 8, \\ x_j \geq 0, \quad j = \overline{1,4}. \end{cases} \quad 12. \quad \begin{cases} x_1x_2 \geq 8, \\ 2x_1 + x_2 + x_3 \geq 10, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}$$

$$z = 3x_1 + 6x_2 + 3x_3 \rightarrow \min \quad z = x_1 + x_2 + 4x_3 + x_4 \rightarrow \min$$

$$13. \quad \begin{cases} x_1 + 4x_2 + x_3 \geq 8, \\ x_1x_3 = 10, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} \quad 14. \quad \begin{cases} x_1x_2x_3 \geq 7, \\ 4x_1 + 2x_2 - x_3 + x_4 = 8, \\ x_j \geq 0, \quad j = \overline{1,4}. \end{cases}$$

$$\begin{array}{ll}
 z = x_1 + 2x_2 \rightarrow \max & z = -3x_1 + 2x_2 + 8x_3 \rightarrow \min \\
 15. \quad \begin{cases} x_1x_2 \leq 4, \\ 3x_1 + 2x_2 \leq 6, \\ x_j \geq 0, \quad j = \overline{1,2}. \end{cases} & 16. \quad \begin{cases} x_1 - x_2 + 3x_3 \leq 10, \\ x_1x_2 \leq 8, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}
 \end{array}$$

$$\begin{array}{ll}
 z = x_1 - 2x_2 + 3x_3 \rightarrow \max & z = 3x_1 - 3x_2 - x_3 \rightarrow \min \\
 17. \quad \begin{cases} x_1x_3 \leq 10, \\ x_1 - x_2 - x_3 = -3, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases} & 18. \quad \begin{cases} -x_1 - 3x_2 + 3x_3 \geq -6, \\ x_2x_3 \leq 4, \\ x_j \geq 0, \quad j = \overline{1,3}. \end{cases}
 \end{array}$$

+ -

$$\begin{array}{ll}
 z = 30x_1 + 10x_2 + 45x_3 \rightarrow \min & z = x_1 + 2x_2 + x_3 - 4x_4 \rightarrow \max \\
 19. \quad \begin{cases} x_1x_2 \leq 10, \\ -5x_1 + 3x_2 - x_3 \geq 0, \\ x_j \geq 0, \quad j = \overline{1,3} \end{cases} & 20. \quad \begin{cases} x_1x_3 < 10, \\ 2x_1 - x_2 + x_3 + x_4 < 10, \\ x_j \geq 0, \quad j = \overline{1,4} \end{cases}
 \end{array}$$

3.4 . Транспортна задача

3.4.1. Приклад транспортної задачі. Компанія контролює три фабрики (постачальники) A_1, A_2, A_3 , здатні виготовляти 150, 60 та 80 тис. од. продукції щотижня. Компанія уклала договір із чотирма фірмами-замовниками (споживачі) B_1, B_2, B_3, B_4 , яким потрібно щотижня відповідно 110, 40, 60 та 80 тис. од. продукції. Вартість транспортування 1000 од. продукції фірмам-замовникам з кожної фабрики наведено в таблиці.

Фабрика	Вартість транспортування 1000 од. продукції за фірмами-замовниками			
	B_1	B_2	B_3	B_4
A_1	4	4	2	5
A_2	5	3	1	2
A_3	2	1	4	2

Визначити для кожної фабрики оптимальний план перевезення продукції до замовників, що мінімізує загальну вартість транспортних послуг.

3.4.2. Математична модель та АМРЛ-код. Нехай x_{ij} — кількість продукції, що перевозиться від i -ї фабрики до j -го замовника ($i = 1, 3; j = 1, 4$). Оскільки транспортна задача

за умовою є збалансованою, закритою $\left(\sum_{i=1}^3 a_i = \sum_{j=1}^4 b_j = 290 \right)$,

то обмеження задачі можна записати за допомогою двох груп лінійних рівностей: перша група рівностей стосується фабрик, а друга – фірм-замовників.

Лінійні рівності для фабрик мають наступний вигляд

$$\begin{cases} x_{11} + x_{12} + x_{13} + x_{14} = 150, \\ x_{21} + x_{22} + x_{23} + x_{24} = 60, \\ x_{31} + x_{32} + x_{33} + x_{34} = 80. \end{cases}$$

Економічний зміст записаних лінійних обмежень полягає у тому, що уся вироблена на фабриках продукція має повністю вивозитися до фірм-замовників.

Аналогічні обмеження можна записати відносно фірм-замовників: продукція, що надходить до споживача, має повністю задовольняти його попит. Математично це записується так:

$$\begin{cases} x_{11} + x_{21} + x_{31} = 110, \\ x_{12} + x_{22} + x_{32} = 40, \\ x_{13} + x_{23} + x_{33} = 60, \\ x_{14} + x_{24} + x_{34} = 80. \end{cases}$$

Загальні витрати, пов'язані з транспортуванням продукції, складаються як добуток обсягу перевезеної продукції та питомої вартості перевезень за відповідним маршрутом і за умовою задачі мають бути мінімальними. Тому математичну модель поставленої задачі можна записати так:

$$Z = 4x_{11} + 4x_{12} + 2x_{13} + 5x_{14} + 5x_{21} + 3x_{22} + x_{23} + 2x_{24} + 2x_{31} + x_{32} + 4x_{33} + 2x_{34} \rightarrow \min$$

при обмеженнях

$$\begin{cases} x_{11} + x_{12} + x_{13} + x_{14} = 150, \\ x_{21} + x_{22} + x_{23} + x_{24} = 60, \\ x_{31} + x_{32} + x_{33} + x_{34} = 80, \\ x_{11} + x_{21} + x_{31} = 110, \\ x_{12} + x_{22} + x_{32} = 40, \\ x_{13} + x_{23} + x_{33} = 60, \\ x_{14} + x_{24} + x_{34} = 80. \end{cases}$$

$$x_{ij} \geq 0, \quad i = \overline{1, 3}; \quad j = \overline{1, 4}.$$

AMPL-код для вказаної моделі має наступний вигляд:

```
var x11 >= 0; var x12 >= 0; var x13 >= 0; var x14 >= 0;
var x21 >= 0; var x22 >= 0; var x23 >= 0; var x24 >= 0;
var x31 >= 0; var x32 >= 0; var x33 >= 0; var x34 >= 0;

minimize z: 4*x11+4*x12+2*x13+5*x14+5*x21+3*x22+x23
            +2*x24+2*x31+x32+4*x33+2*x34;
subject to A1: x11+x12+x13+x14=150;
subject to A2: x21+x22+x23+x24=60;
subject to A3: x31+x32+x33+x34=80;
subject to B1: x11+x21+x31=110;
subject to B2: x12+x22+x32=40;
subject to B3: x13+x23+x33=60;
subject to B4: x14+x24+x34=80;
solve;
display z;
display x11, x12, x13, x14;
display x21, x22, x23, x24;
display x31, x32, x33, x34;
```

Результат виконання на NEOS-сервері:

```
Gurobi 5.5.0: outlev=1
threads=4
Optimize a model with 7 rows, 12 columns and 24 nonzeros
Presolve time: 0.00s
Presolved: 7 rows, 12 columns, 24 nonzeros

Iteration   Objective      Primal Inf.   Dual Inf.     Time
    0   4.8000000e+02  3.000000e+02  0.000000e+00   0s
    3   7.2000000e+02  0.000000e+00  0.000000e+00   0s

Solved in 3 iterations and 0.00 seconds
Optimal objective 7.200000000e+02
Gurobi 5.5.0: optimal solution; objective 720
```

3 simplex iterations

$$z = 720$$

$$x_{11} = 90$$

$$x_{12} = 0$$

$$x_{13} = 60$$

$$x_{14} = 0$$

$$x_{21} = 0$$

$$x_{22} = 0$$

$$x_{23} = 0$$

$$x_{24} = 60$$

$$x_{31} = 20$$

$$x_{32} = 40$$

$$x_{33} = 0$$

$$x_{34} = 20$$

Результат виконання на AMPL:

MINOS 5.5: optimal solution found.

5 iterations, objective 720

$$z = 720$$

$$x_{11} = 90$$

$$x_{12} = 0$$

$$x_{13} = 60$$

$$x_{14} = 0$$

$$x_{21} = 0$$

$$x_{22} = 0$$

$$x_{23} = 0$$

$$x_{24} = 60$$

$$x_{31} = 20$$

$$\begin{aligned}x_{32} &= 40 \\x_{33} &= 0 \\x_{34} &= 20\end{aligned}$$

3.4.3. Математична модель у вигляді задачі лінійного програмування загального виду та її AMPL-код. Ця задача включає 12=3*4 змінних та 7=3+4 обмежень. Вона має наступний вигляд :

$$Z = c^T x \rightarrow \min \quad (3.4.1)$$

при обмеженнях:

$$Ax = b, \quad (3.4.2)$$

$$x \geq 0, \quad (3.4.3)$$

де вектори c , b та матриця A є наступними:

$$c = (4, 4, 2, 5, 5, 3, 1, 2, 2, 1, 4, 2)^T; \quad b = (150, 60, 80, 110, 40, 60, 80)^T;$$

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

AMPL-код для моделі (3.4.1)–(3.4.3) має наступний вигляд:

```
param m;
param n;

set I=1..m;
set J= 1..n;
```

```

param c{J};
param b{I};
param a{I,J};

var x{J}>=0;
minimize z: sum {j in 1..n} c[j] * x[j];
subject to con{i in 1..m}: sum {j in 1..n} a[i,j] * x[j] = b[i];
data;
param m:=7;
param n:=12;
param c [1] 4 [2] 4 [3] 2 [4] 5 [5] 5 [6] 3 [7] 1 [8] 2 [9] 2 [10] 1
[11] 4 [12] 2;
param b [1] 150 [2] 60 [3] 80 [4] 110 [5] 40 [6] 60 [7] 80;
param a:   1 2 3 4 5 6 7 8 9 10 11 12 :=
    1  1 1 1 1 0 0 0 0 0 0 0
    2  0 0 0 0 1 1 1 1 0 0 0 0
    3  0 0 0 0 0 0 0 0 1 1 1 1
    4  1 0 0 0 1 0 0 0 1 0 0 0
    5  0 1 0 0 0 1 0 0 0 1 0 0
    6  0 0 1 0 0 0 1 0 0 0 1 0
    7  0 0 0 1 0 0 0 1 0 0 0 1;

solve;
display z, x;

```

Результат виконання на NEOS-сервері:

```

Gurobi 5.5.0: outlev=1
threads=4
Optimize a model with 7 rows, 12 columns and 24 nonzeros
Presolve time: 0.00s
Presolved: 7 rows, 12 columns, 24 nonzeros

Iteration   Objective      Primal Inf.   Dual Inf.     Time
    0   4.8000000e+02  3.000000e+02  0.000000e+00    0s
    3   7.2000000e+02  0.000000e+00  0.000000e+00    0s

```

```
Solved in 3 iterations and 0.00 seconds
Optimal objective 7.200000000e+02
Gurobi 5.5.0: optimal solution; objective 720
3 simplex iterations
z = 720
x [*] :=
1 90
2 0
3 60
4 0
5 0
6 0
7 0
8 60
9 20
10 40
11 0
12 20
;
```

Результат виконання на AMPL:

```
MINOS 5.5: optimal solution found.
5 iterations, objective 720
z = 720
x [*] :=
1 90
2 0
3 60
4 0
5 0
6 0
7 0
8 60
9 20
```

10	40
11	0
12	20
;	

3.4.4. Загальна математична модель та її AMPL-код.

Для транспортної задачі з m постачальниками та n споживачами загальна математична модель має такий вигляд:

$$Z = \sum_{i=1}^m \sum_{j=1}^n c_{ij} \cdot x_{ij} \rightarrow \min \quad (3.4.4)$$

при обмеженнях:

$$\sum_{j=1}^n x_{ij} = a_i \quad (i = \overline{1, m}), \quad (3.4.5)$$

$$\sum_{i=1}^m x_{ij} = b_j \quad (j = \overline{1, n}), \quad (3.4.6)$$

$$x_{ij} \geq 0 \quad (i = \overline{1, m}; j = \overline{1, n}), \quad (3.4.7)$$

де x_{ij} — кількість продукції, що перевозиться від i -го постачальника до j -го споживача; c_{ij} — вартість перевезення одиниці продукції від i -го постачальника до j -го споживача; a_i — запаси продукції i -го постачальника; b_j — попит на продукцію j -го споживача.

AMPL-код для задачі (3.4.4)–(3.4.7) має наступний вигляд

```

set I; # постачальники
set J; # споживачі

param A {I} >= 0; # продукти постачальників
param B {J} >= 0; # продукти споживачів

check: sum {i in I} A[i] = sum {j in J} B[j];

param c {I,J} >= 0; # транспортні затрати

```

```

var x {i in I, j in J} >= 0;
minimize z: sum {i in I, j in J} c[i,j] * x[i,j];
subject to Supply {i in I}: sum {j in J} x[i,j] = A[i];
subject to Demand {j in J}: sum {i in I} x[i,j] = B[j];
data;
param: I: A := # постачальники та їх продукти
    A1 150
    A2 60
    A3 80;
param: J: B := # споживачі та їх продукти
    B1 110
    B2 40
    B3 60
    B4 80;
param c:
    B1 B2 B3 B4 :=
    A1 4 4 2 5
    A2 5 3 1 2
    A3 2 1 4 2 ;
display I; display J;
display A; display B;
display c;

solve;
display z;
display x;

```

Результат виконання на NEOS-сервері:

```

set I := A1 A2 A3;
set J := B1 B2 B3 B4;

A [*] :=
A1 150
A2 60

```

A3 80

;

B [*] :=

B1 110

B2 40

B3 60

B4 80

;

c :=

A1 B1 4

A1 B2 4

A1 B3 2

A1 B4 5

A2 B1 5

A2 B2 3

A2 B3 1

A2 B4 2

A3 B1 2

A3 B2 1

A3 B3 4

A3 B4 2

;

12 variables, all linear

7 constraints, all linear; 24 nonzeros

7 equality constraints

1 linear objective; 12 nonzeros.

Gurobi 5.5.0: outlev=1

threads=4

Optimize a model with 7 rows, 12 columns and 24 nonzeros

Presolve time: 0.00s

Presolved: 7 rows, 12 columns, 24 nonzeros

Iteration	Objective	Primal Inf.	Dual Inf.	Time
0	4.8000000e+02	3.0000000e+02	0.0000000e+00	0s

```
3 7.2000000e+02 0.000000e+00 0.000000e+00 0s
```

Solved in 3 iterations and 0.00 seconds

Optimal objective 7.200000000e+02

Gurobi 5.5.0: optimal solution; objective 720

3 simplex iterations

z = 720

x :=

A1 B1 90

A1 B2 0

A1 B3 60

A1 B4 0

A2 B1 0

A2 B2 0

A2 B3 0

A2 B4 60

A3 B1 20

A3 B2 40

A3 B3 0

A3 B4 20

;

Результат виконання на AMPL:

```
set I := A1 A2 A3;
```

```
set J := B1 B2 B3 B4;
```

```
A [*] :=
```

```
A1 150
```

```
A2 60
```

```
A3 80
```

```
;
```

```
B [*] :=
```

```
B1 110
```

```
B2 40
```

```
B3 60
B4 80
;
c :=
A1 B1 4
A1 B2 4
A1 B3 2
A1 B4 5
A2 B1 5
A2 B2 3
A2 B3 1
A2 B4 2
A3 B1 2
A3 B2 1
A3 B3 4
A3 B4 2
;
MINOS 5.5: optimal solution found.
5 iterations, objective 720
z = 720

x :=
A1 B1 90
A1 B2 0
A1 B3 60
A1 B4 0
A2 B1 0
A2 B2 0
A2 B3 0
A2 B4 60
A3 B1 20
A3 B2 40
A3 B3 0
A3 B4 20
;
```

3.4.5 Математична модель у вигляді задачі нелінійного програмування та її AMPL-код. Для m постачальників та n споживачів математична модель має такий вигляд:

$$Z_1 = \sum_{i=1}^m \sum_{j=1}^n \varepsilon \left(x_{ij} + \frac{c_{ij}}{2\varepsilon} \right)^2 \rightarrow \min \quad (3.4.8)$$

при обмеженнях:

$$\sum_{j=1}^n x_{ij} = a_i \quad (i = \overline{1, m}), \quad (3.4.9)$$

$$\sum_{i=1}^m x_{ij} = b_j \quad (j = \overline{1, n}), \quad (3.4.10)$$

$$x_{ij} \geq 0 \quad (i = \overline{1, m}; j = \overline{1, n}). \quad (3.4.11)$$

При малих значеннях ε оптимальний план задачі співпадає із оптимальним планом задачі (3.4.4)–(3.4.7), а оптимальні значення цільових функцій Z_1^* та Z^* відрізняються на величину

$$Z_1^* - Z^* = \sum_{i=1}^m \sum_{j=1}^n \varepsilon (x_{ij}^*)^2 + \frac{1}{4\varepsilon} \sum_{i=1}^m \sum_{j=1}^n c_{ij}^2$$

AMPL-код для задачі (3.4.8)–(3.4.11) має наступний вигляд:

```

set I;
set J;
param A {I} >= 0;
param B {J} >= 0;
check: sum {i in I} A[i] = sum {j in J} B[j];
param c {I,J} >= 0;
param eps := 0.001;
var x {i in I, j in J} >= 0;
minimize z1: sum {i in I, j in J}
eps*(x[i,j]+c[i,j]/(2*eps))*(x[i,j]+c[i,j]/(2*eps));
subject to Supply {i in I}: sum {j in J} x[i,j] = A[i];

```

```

subject to Demand {j in J}: sum {i in I} x[i,j] = B[j];
data;
param: I: A :=
    1 150
    2 60
    3 80;
param: J: B :=
    1 110
    2 40
    3 60
    4 80;
param c:
    1 2 3 4 :=
    1 4 4 2 5
    2 5 3 1 2
    3 2 1 4 2;
solve;
display z1;
display z1 - sum {i in I, j in J} eps*x[i,j]*x[i,j]
- sum {i in I, j in J} eps*(c[i,j]/(2*eps))*(c[i,j]/(2*eps));
display x;

```

Результат виконання на NEOS-сервері:

```

Gurobi 5.5.0: optimal solution; objective 31987.7
8 barrier iterations
No basis.
z1 = 31987.7
z1 - sum{i in I, j in J} eps*(c[i,j]/(2*eps))*(c[i,j]/(2*eps)) -
sum{i in I, j in J} eps*x[i,j]*x[i,j] = 720
x :=
1 1 90
1 2 2.70289e-08
1 3 60
1 4 1.4378e-08
2 1 2.75146e-08

```

```
2 2 2.0718e-08
2 3 4.85608e-08
2 4 60
3 1 20
3 2 40
3 3 2.25829e-08
3 4 20
;
```

Результат виконання на AMPL:

```
MINOS 5.5: optimal solution found.
7 iterations, objective 31987.7
Nonlin evals: obj = 18, grad = 17.
z1 = 31987.7
z1 - sum{i in I, j in J} eps*(c[i,j]/(2*eps))*(c[i,j]/(2*eps)) -
sum{i in I, j in J} eps*x[i,j]*x[i,j] = 720

x :=
1 1 90
1 2 0
1 3 60
1 4 0
2 1 0
2 2 0
2 3 1.94588e-15
2 4 60
3 1 20
3 2 40
3 3 0
3 4 20
;
```

3.4.6. Транспортні задачі для розв'язання.

У пунктах постачання A_1, A_2, A_3 є однорідний вантаж в обсязі a_1, a_2, a_3 одиниць відповідно; цей вантаж треба транспортувати у пункти B_1, B_2, B_3, B_4, B_5 з в обсязі відповідно b_1, b_2, b_3, b_4, b_5 одиниць. Потреби замовника (в умовних одиницях), запаси вантажу на кожному пункті постачання (у тих же одиницях) і тарифи (вартість перевезення одиниці вантажу з даного пункту постачання даному замовнику) вказані в таблиці. Потрібно спланувати перевезення так, щоб загальна сума вартості перевезень була найменшою.

Задача 1.

Пункти постачання	Пункти призначення					Запаси
	B_1	B_2	B_3	B_4	B_5	
A_1	2	3	4	2	4	140
A_2	8	4	1	4	1	180
A_3	3	7	3	1	2	160
Потреби	60	70	120	130	100	480

Задача 2.

Пункти постачання	Пункти призначення					Запаси
	B_1	B_2	B_3	B_4	B_5	
A_1	4	5	2	8	6	115
A_2	3	1	9	7	3	175
A_3	9	6	7	2	1	130
Потреби	70	220	40	30	60	420

Задача 3.

Пункти постачання	Пункти призначення					Запаси
	B_1	B_2	B_3	B_4	B_5	
A_1	3	8	7	11	15	260
A_2	14	3	1	8	6	400
A_3	9	5	16	7	12	240
Потреби	180	200	190	230	100	900

Задача 4.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	2	10	15	14	4	150
A ₂	3	7	12	5	8	170
A ₃	21	18	6	13	16	260
Потреби	100	90	160	150	80	580

Задача 5.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	14	8	17	5	3	120
A ₂	21	10	7	11	6	180
A ₃	3	5	8	4	9	230
Потреби	70	120	105	125	110	530

Задача 6.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	12	9	7	11	6	175
A ₂	4	3	12	2	8	165
A ₃	5	17	9	4	11	180
Потреби	90	120	110	130	70	520

Задача 7.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	3	8	7	11	15	260
A ₂	14	3	1	8	6	400
A ₃	9	5	16	7	12	240
Потреби	180	200	190	230	100	900

Задача 8.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	2	4	11	5	3	250
A ₂	8	17	13	7	6	300
A ₃	14	10	5	8	9	270
Потреби	120	200	190	230	80	820

Задача 9.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	21	18	14	3	6	370
A ₂	7	11	10	5	12	450
A ₃	4	8	12	8	13	430
Потреби	300	230	330	290	100	1250

Задача 10.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	3	10	15	17	9	560
A ₂	2	16	3	15	4	570
A ₃	8	5	12	14	7	620
Потреби	350	300	350	500	250	1750

Задача 11.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	11	4	15	1	2	350
A ₂	20	9	7	14	5	350
A ₃	18	10	3	8	6	300
Потреби	180	220	230	270	100	1000

Задача 12.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	2	4	5	11	3	400
A ₂	12	8	6	14	11	370
A ₃	10	15	7	9	18	380
Потреби	250	200	290	260	150	1150

Задача 13.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	20	3	9	15	35	300
A ₂	14	10	12	20	46	150
A ₃	25	11	16	19	48	250
Потреби	150	100	150	100	200	700

Задача 14.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	7	3	9	15	35	180
A ₂	14	10	12	20	46	100
A ₃	15	11	14	17	39	120
Потреби	100	60	90	70	80	400

Задача 15.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	12	42	15	17	9	250
A ₂	16	16	13	15	4	125
A ₃	16	34	22	14	10	225
Потреби	120	110	80	190	100	600

Задача 16.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	7	20	3	15	35	200
A ₂	3	16	10	20	41	100
A ₃	15	25	11	19	40	200
Потреби	80	100	70	130	120	500

Задача 17.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	1	20	3	9	35	220
A ₂	3	14	10	12	42	100
A ₃	15	31	21	16	47	180
Потреби	70	110	80	100	140	500

Задача 18.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	7	20	3	9	15	210
A ₂	3	14	10	12	20	140
A ₃	15	25	11	16	19	150
Потреби	80	120	90	110	100	500

Задача 19.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	11	7	3	9	15	170
A ₂	12	3	10	12	20	120
A ₃	19	15	11	16	19	110
Потреби	90	60	100	80	70	400

Задача 20.

Пункти постачання	Пункти призначення					Запаси
	B ₁	B ₂	B ₃	B ₄	B ₅	
A ₁	11	1	20	9	15	250
A ₂	12	3	14	12	20	200
A ₃	13	15	25	16	19	150
Потреби	75	125	150	50	200	600

IV. Дві ЛП-задачі з булевими змінними для відмовостійкої мережі.

Нехай $N = (V, A)$ – орієнтована мережа з множиною вершин V та множиною дуг A . Пропускна здатність дуги $a = (i, j) \in A$, спрямованої з вершини $i \in V$ у вершину $j \in V$, позначимо y_a . Поломкою дуги $a \in A$ назвемо зменшення її пропускної здатності за правилом: $y'_a = \mu y_a$, де $\mu \in [0, 1)$. Якщо $\mu = 0$, то $y'_a = 0$ і це рівносильно відмові дуги $a \in A$.

Нехай для орієнтованої мережі $N = (V, A)$ задані:

- 1. Мережевий трафік:** множина трійок $\{d_k, s_k, r_k : k \in K\}$, де d_k – об'єм потоку від вершини-відправника $s_k \in V$ до вершини-одержувача $r_k \in V$, $K = \{1, 2, \dots, k_{\max}\}$.
- 2. Сценарій відмов:** множина поломок T , де кожна поломка $t \in T$ визначається пропускними здатностями дуг $y'_a = \mu_{at} y_a$, $\mu_{at} \in [0, 1]$, $\forall a \in A$.

Мережу $N = (V, A)$ назвемо відмовостійкою, якщо значення пропускних здатностей дуг $y = \{y_a : a \in A\}$ такі, що **мережевий трафік** можна виконати завжди, якщо відбудеться тільки одна, але довільна, поломка з сценарію відмов.

Нехай для кожної дуги $a \in A$ задані: y_a^0 – значення вже існуючої пропускної здатності, y_a^{low} , y_a^{up} – границі знизу і зверху на додану до існуючої пропускну здатність, c_a – вартість пересилання одиничного потоку, F_a – витрати на створення доданої пропускної здатності..

Потрібно знайти не більше, ніж N_u дуг, таких, що, якщо їх пропускні здатності додати до вже існуючих, то мережа буде відмовостійкою, а сумарні витрати на створення цих дуг та пересилку потоків по них будуть мінімальними.

Розглянемо дві задачі лінійного програмування (ЛП-задачі) з булевими змінними [12]. В **задачі А** для пересилання потоків може використовуватися будь-який можливий шлях мережі. **Задача Р** визначається тим, що для пересилання потоків використовуються тільки шляхи із заданої множини шляхів $P = \cup_{k \in K} P_k$, де P_k – множина шляхів для потоку k .

Нехай змінні $y = \{y_a : a \in A\}$ – невідомі значення пропускних здатностей доданих дуг, їм відповідають булеві змінні $u = \{u_a : a \in A\}$, де u_a дорівнює одиниці, якщо дуга $a \in A$ додається до мережі, і нулю, в іншому випадку; змінні $x = \{x_{akt}, \forall a, k, t\}$ – невідомі обсяги потоків в мережі, де x_{akt} – обсяг потоку k по дузі a для поломки t .

Нехай A_i^+ та A_i^- – множини вхідних та вихідних дуг із вершини $i \in N$. Тоді формулювання **задачі А** має вигляд: знайти

$$f_A^* = \min_{x, y, u} \sum_{a \in A} c_a y_a + \sum_{a \in A} F_a u_a \quad (1A)$$

при обмеженнях:

$$\sum_{k \in K} x_{akt} \leq \mu_{at} (y_a^0 + y_a), \quad \forall a \in A, t \in T, \quad (2A)$$

$$\sum_{a \in A_i^+} x_{akt} - \sum_{a \in A_i^-} x_{akt} = \begin{cases} d_k, & \text{якщо } i = s_k, \\ -d_k, & \text{якщо } i = r_k, \\ 0, & \text{інакше,} \end{cases} \quad \forall i \in V, k \in K, t \in T, \quad (3A)$$

$$y_a^{low} u_a \leq y_a \leq y_a^{up} u_a, \quad \forall a \in A, \quad \sum_{a \in A} u_a \leq N_u, \quad (4A)$$

$$x_{akt} \geq 0, \forall a \in A, k \in K, t \in T, \quad u_a = 0 \vee 1, \forall a \in A. \quad (5A)$$

Цільова функція (1A) дорівнює сумі затрат на пересилку потоків по доданих дугах і витрат на їх створення. Обмеження (2A) означають, що сумарні потоки по кожній дузі не перевищують її пропускної здатності при довільній поломці з сценарію відмов; обмеження (3A) відповідають за виконання мережевого трафіку; обмеження (4A) відповідають за вибір не більше N_u дуг та гарантують, що їх

пропускні здатності вибираються з діапазону $[y_a^{low}, y_a^{up}]$; обмеження (5A) відповідають за невід'ємність потокових змінних x та булевих змінних u .

Формулювання **задачі Р** має такий вигляд:
знайти

$$f_P^* = \min_{z, y, u} \sum_{a \in A} c_a y_a + \sum_{a \in A} F_a u_a \quad (1P)$$

при обмеженнях:

$$\sum_{k \in K} \sum_{p \in P_k} \delta_{kpa} z_{kpt} \leq \mu_{at} (y_a^0 + y_a), \quad \forall a \in A, t \in T, \quad (2P)$$

$$\sum_{p \in P_k} z_{kpt} = d_k \quad \forall k \in K, t \in T, \quad (3P)$$

$$y_a^{low} u_a \leq y_a \leq y_a^{up} u_a, \quad \forall a \in A, \quad \sum_{a \in A} u_a \leq N_u, \quad (4P)$$

$$z_{kpt} \geq 0, \quad \forall k \in K, p \in P, t \in T, \quad u_a = 0 \vee 1, \quad \forall a \in A, \quad (5P)$$

де

$$\delta_{kpa} = \begin{cases} 1, & \text{якщо шлях } p \in P_k \text{ використовує дугу } a, \\ 0, & \text{інакше.} \end{cases}$$

Тут інші змінні $z = \{z_{kpt}, \forall k, p, t\}$ (z_{kpt} – обсяг потоку $k \in K$ по шляху $p \in P_k$ для поломки $t \in T$), а зміст обмежень (2P)–(5P) такий же, як і в **задачі А**.

Задачі А та **Р** належать до задач великих розмірів, в яких кількість змінних та обмежень вимірюється мільйонами навіть для мереж середніх розмірів. Так, у **задачі А** кількість змінних дорівнює $N_A = |A| * |K| * |T| + 2|A|$, а кількість обмежень – $M_A = |V| * |K| * |T| + |A| * |T| + |A| + 1$. Для середніх розмірів мережі задача (1A)–(5A) є задачею великих розмірів. Наприклад, якщо $|V| \approx 40$, $|A| \approx 50$, $|T| \approx 50$ та $|K| \approx 1000$, то $N_A \approx 2.500.000$, $M_A \approx 2.000.000$. У **задачі Р** кількість змінних дорівнює $N_P = |A| + |P| * |T| + 2|A|$, а кількість обмежень – $M_P = |K| * |T| + |A| * |T| + 3|A| + 1$. Якщо $|A| \approx 50$,

$|T| \approx 50$, $|P| \approx 2000$, $|K| \approx 1000$, то задача (1P)-(5P) має $N_p \approx 100.000$ змінних та $M_p \approx 50.000$. обмежень.

Наведені ЛП-задачі з булевими змінними узагальнюють ЛП-задачі для знаходження пропускних здатностей дуг відмовостійкої орієнтованої мережі [11, стор. 99–110]. Для середнього розміру мереж їх можна ефективно розв'язувати за допомогою NEOS-сервера та програми *gurobi*. Це підтверджують обчислювальні експерименти для задачі А, коли неорієнтована мережа net-46-56 включала 46 вершин та 52 ребра. У таблиці 1 наведено розміри ЛП-задач без поломок та для однієї, двох і трьох поломок.

Таблиця 1. Розміри задач для мережі net-46-56.

Задача	Поломки	Змінні	Обмеження	Ненул. коэф.
net-48-56	0	252.784	108.400	758.128
net-48-56-1	1	500.944	216.798	1.502.718
net-48-56-2	2	749.104	325.196	2.247.308
net-48-56-3	3	997.264	433.594	2.991.898

Ініціатором цих тестових експериментів був професор Нурмінський Є.О. (Владивосток, Росія). У таблиці 2 наведено затрати програми *gurobi* на розв'язання задач із таблиці 1.

Таблиця 2. Програма *gurobi* у задачах Нурмінського

Задача	Ітерації	Час (сек)
net-48-56	47.099	1.74
net-48-56-1	175.646	264.31
net-48-56-2	267.152	935.75
net-48-56-3	382.015	498.68

Із таблиці 2 бачимо, що розв'язок ЛП-задачі з мільйоном змінних та п'ятистами обмежень отримано менше, ніж за десять хвилин. Тому застосування NEOS-інтерфейсу є перспективним, якщо потрібно досить швидко розв'язати ту чи іншу оптимізаційну задачу для мережі середніх розмірів.

Література

1. NEOS Server [Электронный ресурс]: <http://www.neos-server.org>. – Режим доступа: свободный.
2. NEOS Solver [Электронный ресурс]: <http://www.neos-server.org/neos/solvers/>. – Режим доступа: свободный.
3. Щербина О.А. Краткое введение в AMPL – современный алгебраический язык моделирования. – Препринт. – 2012. – 329 с.
4. Fourer R. AMPL, A Modeling Language for Mathematical Programming, Second Edition / R. Fourer, D. Gay, B. Kernighan. – Belmont: Duxbury Press, 2003. – 517 p.
5. Данциг Д. Линейное программирование, его применения и обобщения. – М., Прогресс, 1966. – 600 с.
6. Канторович Л.В. Математические методы организации и планирования производства, 1939.
7. Gurobi Optimization, Inc., Gurobi Optimizer Reference Manual, 2014, <http://www.gurobi.com>.
8. Муртаф Б. Современное линейное программирование. М.: Мир, 1984. – 224 с.
9. Поляк Б.Т. Введение в оптимизацию. – М.: Наука, 1983. – 384 с.
10. Михалевич В.С., Трубин В.А., Шор Н.З. Оптимизационные задачи производственно-транспортного планирования. – М.: Наука, 1986. – 264 с.
11. Шор Н.З., Сергиенко І.В., Шило В.П., Стецюк П.І. та ін. Задачі оптимального проектування надійних мереж. – К.: Наукова думка, 2005. – 230 с.
12. Сергиенко І.В., Стецюк П.І. Две ЛП-задачи с булевыми переменными для отказоустойчивой сети // Информатика та системні науки (ІСН-2014): матеріали V Всеукр. наук.-практ. конф. (м. Полтава, 13-15 березня 2014 року). – Полтава: ПУЕТ, 2014. – С. 284–287.