Міністерство освіти і науки України
Національний Університет «Києво-Могилянська Академія»
Кафедра математики факультету інформатики

Застосування марківських випадкових полів для моделювання економічних процесів з конкурентними технологіями

Курсова робота за спеціальністю «Прикладна математика»

Керівник курсової роботи
к. фіз.-мат. наук, доцент
Чорней Р.К.

Виконав студент
Кулинич Б.В.

Київ – 2014

Зміст

Вступ

В економіці часто виникає [2] задача моделювання системи підприємств (економічних агентів), що взаємодіють, де кожне з підприємств у кожен момент часу обирає певну технологію виробництва, яка впливає на його власний стан та стани й поведінку інших підприємств у майбутньому.

Об'єктом дослідження є стохастична марківська модель такої системи. Предметом дослідження є задача знаходження оптимальної стратегії вибору технологій, яка мінімізуватиме витрати (максимізуватиме доходи) підприємств. Метою роботи є формулювання та програмна реалізація методу знаходження оптимальної стратегії. У роботі побудовано модель системи, сформульовано алгоритм і реалізовано програму для знаходження оптимальної стратегії. Використано методи теорії ймовірностей, теорії керованих марківських процесів та лінійного програмування.

Теоретична частина

2.1 Постановка задачі

У загальному випадку система описується [1] для дискретного часу $t = 0, 1, \ldots$ скінченною множиною економічних агентів V і неорієнтованим графом їхніх взаємодій $G = (V, E)$, скінченною множиною $X^t = \times_{v \in V} X_v^t,\ X_v^t = \{x_v^{t,1}, x_v^{t,2}, \ldots, x_v^{t,n_v}\}$ можливих станів агента $v \in V$ у момент часу t, множиною рішень (про вибір відповідних технологій) $\Delta_v^t : \times_{t=0,1,\ldots} X^t \to U_v^t$, де U_v^t – скінченна множина можливих дій, та відповідних їм функцій витрат (доходів) $r_v^t : X_v^t \times U_v^t \to \{u \mid u \in \mathbb{R}, |u| < C\}$ для агента $v \in V$ у момент часу t.

2.1.1 Припущення

Для побудови моделі, щодо системи буде додатково введено ряд припущень.

Стохастичність. Для всіх $v \in V$

$$\xi_v = \{\xi_v^t \mid t = 0, 1, \ldots\}$$

$$(\Omega, \mathcal{F}, P). \quad \xi_v^t : \Omega \to X_v \ ,$$

де ξ_v – стохастичний процес, що визначає стан v у кожен момент часу t.

Локальність. Для всіх $v \in V$

$$\Delta_v^t = \Delta_v^t(x^0, x^1, \ldots, x^t) = \Delta_v^t(x^0_{\tilde{N}(v)}, x^1_{\tilde{N}(v)}, \ldots, x^t_{\tilde{N}(v)})$$

$$\begin{gathered} P(\xi_v^{t+1} = x_v \mid \xi^0 = x^0, \Delta^0 = u^0,\ \ldots, \xi^t = x^t, \Delta^t = u^t) = \\ = P(\xi_v^{t+1} = x_v \mid \xi^0_{\tilde{N}(v)} = x^0_{\tilde{N}(v)}, \Delta^0_{\tilde{N}(v)} = u^0_{\tilde{N}(v)},\ \ldots, \\ \xi^t_{\tilde{N}(v)} = x^t_{\tilde{N}(v)}, \Delta^t_{\tilde{N}(v)} = u^t_{\tilde{N}(v)}) \end{gathered}$$

Локальність є природньою в контексті задачі, оскільки усі агенти, що не взаємодіють з даним відповідно до графу взаємодії $G = (V, E)$, не чинять на нього безпосереднього впливу. Звідси, агентові для прийняття рішення достатньо знати стани тільки тих агентів, з якими він взаємодіє за графом G.

Синхронність. Для всіх $W \subset V$

$$\begin{gathered} P(\xi_W^{t+1} = x_W \mid \xi^t = x^t, \Delta^t = u^t) = \\ = \prod_{w \in W} P(\xi_w^{t+1} = x_w \mid \xi^t = x^t, \Delta^t = u^t) \end{gathered}$$

Усі агенти системи переходять у свій наступний стан одночасно. Це припущення узгоджується з інтерпретацією t як конкретного моменту в часі, однакового для всіх агентів.

Повнота стану. (припущення Маркова) Для всіх $v \in V$

$$\Delta_v^t(x_{\tilde{N}(v)}^0, x_{\tilde{N}(v)}^1, \dots, x_{\tilde{N}(v)}^t) = \Delta_v^t(x_{\tilde{N}(v)}^t)$$

$$\begin{aligned} P(\xi_v^{t+1} = x_v \mid \xi_{\tilde{N}(v)}^0 = x_{\tilde{N}(v)}^0, \Delta_{\tilde{N}(v)}^0 = u_{\tilde{N}(v)}^0,\ \dots, \\ \xi_{\tilde{N}(v)}^t = x_{\tilde{N}(v)}^t, \Delta_{\tilde{N}(v)}^t = u_{\tilde{N}(v)}^t) = \\ = P(\xi_v^{t+1} = x_v \mid \xi_{\tilde{N}(v)}^t = x_{\tilde{N}(v)}^t, \Delta_{\tilde{N}(v)}^t = u_{\tilde{N}(v)}^t) \end{aligned}$$

Припущення є природнім, оскільки для прийняття рішення про вибір технології найважливішим є стан сусідніх агентів у попередній момент часу, тоді як повною історією їх рішень можна знехтувати.

Інваріантність рішень, та просторів станів і можливих дій

Для всіх $v \in V$, у кожен момент часу $t', t'' = 0, 1, ...$

$$X_v^{t'} = X_v^{t''} = X_v$$
$$U_v^{t'} = U_v^{t''} = U_v$$
$$\Delta_v^{t'} = \Delta_v^{t''} = \Delta_v$$

Оскільки в реальних системах набір доступних технологій змінюється рідко (внаслідок інновацій, або застаріння технологій), змінністю простору можливих дій, так само, як і змінністю простору станів, можна знехтувати в багатьох випадках. Відповідно, за незмінних просторів станів та рішень, має сенс припущення стаціонарність стратегії.

Вважитемо також, що для кожного агента $v \in V$ всі можливі дії U_v допустимі.

2.1.2 Марківська модель

Для неорієнтованого графа $G = (V, E)$ позначимо як $N(v)$ множину вершин, що з'єднані з вершиною $v \in V$, і як $\tilde{N}(v)$ — множину $N(v)$

разом із самою v:

$$N(v) = \{w \mid \{v, w\} \in E\},\ \tilde{N}(v) = N(v) \cup \{v\}$$

Означення 1. Неорієнтований граф $G = (V, E)$, множина випадкових величин, визначених на імовірнісному просторі $(\Omega, \mathcal{F}, P)$, $\xi_v : X_v \to [0, 1]$, $v \in V$ утворюють *марківське випадкове поле*, якщо для всіх $v \in V$:

$$P(\xi_v = x_v \mid \xi_{V \setminus \{v\}} = x_{V \setminus \{v\}}) = P(\xi_k = x_v \mid \xi_{N(v)} = x_{N(v)})$$

Поняття можна розширити до стохастичних процесів.

Означення 2. Нехай $\xi = \{\xi^t \mid t = 0, 1, \ldots\}$, $v \in V$ — стохастичний процес з дискретним часом. Якщо виконуються:

1 (Локальність) Для всіх $v \in V$

$$\begin{aligned} &P(\xi_v^{t+1} = x_v \mid \xi^0 = x^0, \xi^1 = x^1, \ldots, \xi^t = x^t) = \\ =&P(\xi_v^{t+1} = x_v \mid \xi^t_{\tilde{N}(v)} = x^t_{\tilde{N}(v)}) \end{aligned}$$

2 (Синхронність) Для всіх $W \subset V$

$$P(\xi_W^{t+1} = x_W \mid \xi^t = x^t) = \prod_{w \in W} P(\xi_w^{t+1} = x_w \mid \xi^t = x^t)$$

Тоді процес ξ разом із графом $G = (V, E)$ утворює *марківське випадкове поле із синхронними компонентами, що локально взаємодіють*, або *марківське випадкове поле з дискретним часом*.

Відразу з означення маємо, що для будь-якого $W \subset V$

$$P(\xi_W^{t+1} = x_W^{t+1} \mid \xi^t = x^t) = \prod_{w \in W} P(\xi_w^{t+1} = x_w^{t+1} \mid \xi^t_{\tilde{N}(w)} = x^t_{\tilde{N}(w)})$$

Нехай V – скінченна множина агентів, що приймають рішення. U_v — скінченний простір можливих дій для агента $v \in V$, причому U_v незалежний від часу, $\Delta_v^t : \times_{t=0,1,\ldots} X^t \to U_v$ – рішення агента v, залежні від історії станів.

Означення 3. Стратегія $\delta = \{\delta_v \mid v \in V\}$, де $\delta_v = \{\Delta_v^t \mid t = 0, 1, \ldots\}$, називається локальною, якщо для всіх $v \in V,\ t = 0, 1, \ldots$

$$\Delta_v^t = \Delta_v^t(x^0_{\tilde{N}(v)}, \ldots, x^t_{\tilde{N}(v)})$$

Означення 4. Локальна стратегія називається марківською, якщо для всіх $v \in V,\ t = 0, 1, \ldots$

$$\Delta_v^t = \Delta_v^t(x^t_{\tilde{N}(v)})$$

Означення 5. Стратегія називається стаціонарною, якщо для всіх $v \in V,\ t', t'' = 0, 1, \ldots,$

$$\Delta_v^{t'} = \Delta_v^{t''}$$

Якщо застосувати локальну марківську стратегію δ до марківського випадкового поля ξ відносно графа G, то пара (ξ, δ) утворює керований марківський ланцюг. Якщо при цьому стратегія δ – стаціонарна, то ланцюг однорідний.

Аналогічно до означення 2, визначимо контрольоване марківське випадкове поле у часі.

Означення 6. Нехай (ξ, δ) — керований стохастичний процес з дискретним часом та з графом взаємодії $G = (V, E)$, де δ – локальна марківська стратегія. Якщо виконується:

1 (Локальність) Для всіх $v \in V$

$$P(\xi_v^{t+1} = x_v \mid \xi^0 = x^0, \Delta^0 = u^0,\ \ldots, \xi^t = x^t, \Delta^t = u^t) =$$
$$= P(\xi_v^{t+1} = x_v \mid \xi^t_{\tilde{N}(v)} = x^t_{\tilde{N}(v)}, \Delta^t_{\tilde{N}(v)} = u^t_{\tilde{N}(v)})$$

2 (Синхронність) Для всіх $W \subset V$

$$P(\xi_W^{t+1} = x_W \mid \xi^t = x^t, \Delta^t = u) = \prod_{w \in W} P(\xi_w^{t+1} = x_w \mid \xi^t = x^t, \Delta^t = u)$$

Тоді (ξ, δ) разом із графом $G = (V, E)$ утворює *кероване марківське поле з дискретним часом*.

Як наслідок з умов означення, маємо для будь-якого $W \subset V$

$$
\begin{aligned}
&P(\xi_W^{t+1} = x_W \mid \xi^t = x^t, \Delta^t = u^t) = \\
= &\prod_{w \in W} P(\xi^{t+1} = x_w \mid \xi^t_{\tilde{N}(w)} = y_{\tilde{N}(w)}, \Delta^t_w(\xi^t_{\tilde{N}(w)}) = u_w)
\end{aligned}
$$

Означення 7. Позначимо

$$
Q_w(x_w / y_{\tilde{N}(w)}, u_w) = P(\xi^{t+1} = x_w \mid \xi^t_{\tilde{N}(w)} = y_{\tilde{N}(w)}, \Delta^t_w(\xi^t_{\tilde{N}(w)}) = u_w)
$$

$$
Q_W(x_W / y, u) = \prod_{w \in W} Q_w(x_w / y_{\tilde{N}(w)}, u_w)
$$

Покладаючи $W = V$ для $Q_W(x_W/y, u)$, назвемо ядром переходу

$$
Q(x/y,\ u) = P(\xi^{t+1} = x \mid \xi^t = y, u^t = u)
$$

$$
\sum_{x \in X} Q(x/y, u) = 1,\ y \in X
$$

Визначений у 2.1.1 керований стохастичний процес (ξ, δ) відносно графа взаємодії $G = (V, E)$ утворює кероване марківське поле з дискретним часом.

Сформулюємо задачу знаходження стратегії, що мінімізує витрати підприємств. Нехай E_y^δ – математичне сподівання, що відповідає процесу (ξ, δ) за початкового стану $\xi^0 = y$. Тоді C_T^δ — середні очікувані витрати за час T:

$$
\begin{aligned}
C_T^\delta = &E\left[\frac{1}{T+1} \sum_{t=0}^{T} r(\xi^t, \Delta^t(\xi^0, \ldots, \xi^t))\right] \\
= &E_y^\delta\ \frac{1}{T+1}\ \sum_{t=0}^{T} r(\xi^t, \Delta^t(\xi^0, \ldots, \xi^t))
\end{aligned}
$$

Задача полягає в знаходженні оптимальної стратегії δ^*, яка мінімізує $C_T^\delta(y)$ при $T \to \infty$ для всіх $y \in X$:

$$R_y^\delta = \lim_{T\to\infty} \sup C_T^\delta(y)$$
$$\delta^* = R_y^{\delta^*},\ y \in X$$

2.2 Метод знаходження оптимальної стратегії

Нехай $D_y^\delta(u)$ – імовірність вибору дії u за попереднього стану системи y для стратегії δ:

$$D_y^\delta(u) = P(\Delta^{t+1} = u \mid \xi^t = y)$$
$$\sum_{u\in U} D_y^\delta(u) = 1,\ y \in X$$

Функція D_y^δ однозначно визначає стратегію δ.

Згідно з результатом у [1], для керованого марківського поля зі скінченними просторами станів та скінченними й інваріантними у часі просторами можливих дій, існує оптимальна стратегія, яка є нерандомізованою, стаціонарною, та марківською. Таким чином, при пошуку оптимальної δ^* можемо обмежитися лише класом *нерандомізованих* стратегій.

Розподіл $D_y^{\delta^*}$ у такому випадку є виродженим, тобто існує таке u_y^*, що:

$$D_y^{\delta^*}(u) = \begin{cases} 1, & u = u_y^* \\ 0, & u \neq u_y^* \end{cases}$$

Оскільки кероване марківське поле за умови стаціонарності стратегії утворює однорідний марківський ланцюг, δ^* можна знайти, використовуючи методи лінійного програмування [3].

Покладемо z_{xu}:

$$z_{xu} = \left[\sum_{u\in U} z_{xu}\right] D_x^{\delta^*}(u) = \pi_x\ D_x^{\delta^*}(u)$$

Тоді задача лінійного програмування для знаходження z_{xu} формулюється так:

$$\min \sum_{x \in X} \sum_{u \in U} r(x,u)\ z_{xu}$$

З обмеженнями:

$$\sum_{u \in U} z_{xu} = \sum_{y \in X} \sum_{u \in U} Q(x/y,u)\, z_{yu},\ x \in X$$

$$\sum_{x \in X} \sum_{u \in U} z_{xu} = 1$$

$$z_{xu} \geq 0,\ x \in X, u \in U$$

Враховуючи нерандомізованість,

$$D_x^{\delta^*}(u) = \begin{cases} 1, & z_{xu} \neq 0 \\ 0, & z_{xu} = 0 \end{cases}$$

Аналогічно формулюватиметься задача максимізації доходів.

Програмна реалізація

Реалізуємо задачу знаходження оптимальної стратегії вибору технологій за допомогою бібліотеки для моделювання задач лінійного програмування `PuLP` [7] для мови Python.

Програма отримує на вхід кількість станів системи $|\times_{v\in V} X_v| = |X|$, кількість можливих дій системи $|\times_{v\in V} U_v| = |U|$, тривимірний масив ядер переходу Q розмірності $|X| \times |X| \times |U|$, де $Q_{ij}^k = Q(x_j/x_i, u_k)$, та матрицю витрат R розмірності $|X| \times |U|$, де $R_{ij} = r(x_i, u_j)$. Як результат повертає множину пар $\{(x, u) \mid D_x^{\delta^*}(u) = 1\}$, яка задає оптимальну стратегію δ^*.

Приклад 1. Нехай $G = (V, E)$, кількість вершин $|V| = 2$, множина можливих станів для кожного агента $X_v = \{a, b\}$, множина можливих впливів $U_v = \{1, 2\}$. Тоді $X = \{(a, a), (a, b), (b, a), (b, b)\}$, $|X| = 4$, $U = \{(1, 1), (1, 2), (2, 1), (2, 2)\}$, $|U| = 4$. Відповідно, маємо 4 матриці ядер переходу Q^u розмірності 4×4. Позначимо

$$x_0 = (a, a),\ x_1 = (a, b),\ x_2 = (b, a),\ x_3 = (b, b),$$
$$u_0 = (1, 1),\ u_1 = (1, 2),\ u_2 = (2, 1),\ u_3 = (2, 2)$$

Тоді матриці ядер переходу Q^u:

$$Q^u = \begin{pmatrix} Q(x_0/x_0, u) & Q(x_1/x_0, u) & Q(x_2/x_0, u) & Q(x_3/x_0, u) \\ Q(x_0/x_1, u) & Q(x_1/x_1, u) & Q(x_2/x_1, u) & Q(x_3/x_1, u) \\ Q(x_0/x_2, u) & Q(x_1/x_2, u) & Q(x_2/x_2, u) & Q(x_3/x_2, u) \\ Q(x_0/x_3, u) & Q(x_1/x_3, u) & Q(x_2/x_3, u) & Q(x_3/x_3, u) \end{pmatrix}$$

Матриця витрат R розмірності 4×4:

$$R = \begin{pmatrix} r(x_0, u_0) & r(x_0, u_1) & r(x_0, u_2) & r(x_0, u_3) \\ r(x_1, u_0) & r(x_1, u_1) & r(x_1, u_2) & r(x_1, u_3) \\ r(x_2, u_0) & r(x_2, u_1) & r(x_2, u_2) & r(x_2, u_3) \\ r(x_3, u_0) & r(x_3, u_1) & r(x_3, u_2) & r(x_3, u_3) \end{pmatrix}$$

На виході програми отримаємо 4 пари значень (x, u), для яких $\Delta_x^{\delta^*}(u) = 1$.

Частково наведемо основну процедуру знаходження стратегії `find_optimal_strategy` на мові Python.

Оголошення функції та ініціалізація об'єктів:

```
def find_optimal_strategy(states, controls, costs, kernels):
    X = range(states)
    U = range(controls)
    R = costs
    Q = kernels

    # LP object
    optm = LpProblem("Optimal strategy", sense=LpMinimize)

    # Variables (continuous in range [0, 1])
    Z = [[LpVariable("({},{})".format(x, u), 0, 1) \
                for u in U] for x in X]
```

Задання оптимізованої функції та обмежень:

```
    # Objective
    optm.objective = sum(np.dot(Z[x], R[x]) for x in X)

    # Constraints
    for x in X:
        cn = (sum(Z[x]) == sum(Q[y][x][u]*Z[y][u] \
                                for u in U for y in X))
        optm.add(cn)

    cn = sum(Z[x][u] for u in U for x in X) == 1
    optm.add(cn)

    optm.solve()
```

```
    return [(x, u) for u in U for x in X \
                    if value(Z[x][u]) != 0]
```

Хоча алгоритм задано не у векторизованій формі, програмне забезпечення для розв'язку задач ЛП, яке використовується бібліотекою `PuLP` — `COIN-OR`, `GLPK` тощо — автоматично векторизує оптимізовану функцію та обмеження, тому ефективність при такому заданні не втрачається. За замовчуванням `PuLP` використовує `COIN-OR` [6], інші варіанти можна обрати при виклику `optm.solve(solver=CUSTOM_SOLVER)`.

Повний код програми можна знайти у Git-репозиторії за посиланням `http://github.com/bogdan-kulynych/mrf-in-economics`.

Приклад 2. В умові з попереднього прикладу, нехай ядра переходу і матриця витрат R задаються таким чином :

$$Q^{u_0} = \begin{pmatrix} 0.1 & 0.1 & 0.4 & 0.4 \\ 0.7 & 0.1 & 0.1 & 0.1 \\ 0.3 & 0.1 & 0.2 & 0.4 \\ 0.1 & 0.5 & 0.2 & 0.2 \end{pmatrix}, \; Q^{u_1} = \begin{pmatrix} 0.2 & 0.1 & 0.3 & 0.4 \\ 0.1 & 0. & 0. & 0.9 \\ 0.3 & 0.5 & 0. & 0.2 \\ 0.1 & 0.1 & 0.7 & 0.1 \end{pmatrix}$$

$$Q^{u_2} = \begin{pmatrix} 0.7 & 0.1 & 0.1 & 0.1 \\ 0.2 & 0.2 & 0.5 & 0.1 \\ 0.9 & 0. & 0.1 & 0. \\ 0. & 0.6 & 0.2 & 0.2 \end{pmatrix}, \; Q^{u_3} = \begin{pmatrix} 0.7 & 0.1 & 0.2 & 0. \\ 0.5 & 0.2 & 0.1 & 0.2 \\ 0.1 & 0.7 & 0.1 & 0.1 \\ 0.8 & 0.1 & 0. & 0.1 \end{pmatrix}$$

$$R = \begin{pmatrix} 1 & 11 & 1 & 3 \\ 1 & 20 & 2 & 3 \\ 1 & 99 & 2 & 4 \\ 1 & 9 & 1 & 9 \end{pmatrix}$$

Цим умовам відповідає програма:

```
states = 4
controls = 4

costs = np.array([
    [1, 11, 1, 3],
    [1, 20, 2, 3],
    [1, 99, 2, 4],
    [1, 9 , 1, 9]
])

kernels = np.array([
    [[0.1, 0.2, 0.7, 0.7], [0.1, 0.1, 0.1, 0.1], \
     [0.4, 0.3, 0.1, 0.2], [0.4, 0.4, 0.1, 0  ]],
    [[0.7, 0.1, 0.2, 0.5], [0.1, 0  , 0.2, 0.2], \
     [0.1, 0  , 0.5, 0.1], [0.1, 0.9, 0.1, 0.2]],
    [[0.3, 0.3, 0.9, 0.1], [0.1, 0.5, 0  , 0.7], \
     [0.2, 0  , 0.1, 0.1], [0.4, 0.2, 0  , 0.1]],
    [[0.1, 0.1, 0  , 0.8], [0.5, 0.1, 0.6, 0.1], \
     [0.2, 0.7, 0.2, 0  ], [0.2, 0.1, 0.2, 0.1]]
])

strategy = find_optimal_strategy(states, controls, costs, kernels)
print(sorted(strategy))
```

Варто зазначити, що вхідна змінна `kernels` — тривимірний масив, де третій вимір відповідає можливим діям u.

В результаті виконання програми буде виведено на екран:

```
[(0, 0), (1, 0), (2, 0), (3, 2)]
```

Тобто $D_{x_0}^{\delta^*}(u_0) = D_{x_1}^{\delta^*}(u_0) = D_{x_2}^{\delta^*}(u_0) = D_{x_3}^{\delta^*}(u_2) = 1$.

Отже, згідно зі знайденою оптимальною стратегією, в станах системи $x_0 = (a, a)$, $x_1 = (a, b)$, $x_2 = (b, a)$ слід обирати дії $u_0 = (1, 1)$, а в стані $x_3 = (b, b)$ дії $u_2 = (2, 1)$. Дії $u_1 = (1, 2), u_3 = (2, 2)$ не слід обирати взагалі.

Висновки

У даній роботі було побудовано стохастичну марківську модель системи економічних агентів, що взаємодіють. За певних досить реалістичних припущень щодо системи, використовуючи результати з теорії керованих марківських процесів, було сформульовано алгоритм знаходження оптимальної стратегії вибору технологій за допомогою лінійного програмування. Програму знаходження такої оптимальної стратегії в загальному випадку було реалізовано за допомогою бібліотеки `PuLP` для мови Python.

Література

1. Chornei R.K., Daduna Hans, Knopov P.S. Controlled Markov fields with finite state space on graphs // Stochastic models. — 2005. — Vol. 21, no. 4. — P. 847–874.

2. David Paul, Foray Dominique, Dalle Jean-Michelle. Marshallian externalities and the emergence and spatial stability of technological enclaves // Economics of Innovation and New Technologies. — 1998. — Vol. 6, no. 2&3. — P. 147–182.

3. Knopov P.S., Chornei R.K. Control problems for markov processes with memory // Cybernetics and Systems Analysis. — 1998. — Vol. 34, no. 3. — P. 368–376. — URL: http://dx.doi.org/10.1007/BF02666978.

4. Knopov P.S., Samosonok A.S. On Markov Stochastic Processes with Local Interaction for Solving Some Applied Problems // Cybernetics and Sys. Anal. — 2011. — Vol. 47, no. 3. — P. 346–359. — URL: http://dx.doi.org/10.1007/s10559-011-9317-3.

5. Koller Daphne, Friedman Nir. Probabilistic graphical models: principles and techniques. — MIT press, 2009.

6. Документація бібліотеки COIN-OR. — URL: http://www.coin-or.org/documentation.html.

7. Документація бібліотеки PuLP. — URL: http://www.coin-or.org/PuLP.