КОМП'ЮТЕРНІ НАУКИ

УДК 004.832.2, 004.023 DOI: https://doi.org/10.32515/2664-262X.2022.6(37).2.3-16

Є. Є. Федоров, проф., д-р техн. наук, **О. В. Нечипоренко**, доц., канд. техн. наук
Черкаський державний технологічний університет, м. Черкаси, Україна
e-mail: fedorovee75@ukr.net, olne@ukr.net

Мультиагентні метаевристичні методи рішення задачі управління запасами

Пропонується задача керування запасами як складова частина задачі ефективного керування ланцюгами постачання. Для вирішення цієї задачі пропонуються мультиагентні метаевристичні методи на основі оптимізації рою частинок та алгоритму рою штучних риб, які використовують динамічні параметри та враховують кількість запасів товару в кінці кожного етапу. Для цих методів запропоновано паралельні алгоритми на основі технології CUDA. Ці методи досліджувалися на основі даних логістичної компанії «Ekol Ukraine» та призначені для інтелектуальних комп'ютерних систем управління ланцюгами постачання.
задача управління запасами, управління ланцюгами постачання, мультиагентні метаевристичні методи, оптимізація рою частинок, алгоритм рою штучних риб

Постановка проблеми. В даний час актуальною є проблема недостатньої ефективності управління ланцюгами постачання [1]. Однією із задач, розв'язуваних у межах зазначеної проблеми, є оптимізаційна задача управління запасами [2]. Методи оптимізації, що знаходять наближене рішення за допомогою спрямованого пошуку, мають високу ймовірність потрапляння до локального екстремуму. Методи оптимізації, що знаходять точне рішення, мають високу обчислювальну складність. Методи випадкового пошуку не гарантують збіжності. У зв'язку з цим виникає проблема недостатньої ефективності методів оптимізації, яка потребує вирішення.

Для прискореного знаходження квазіоптимального рішення та зниження ймовірності попадання в локальний екстремум використовуються метаевристики (або сучасні евристики) [3]. Метаевристика розширює можливості евристики, комбінуючи евристичні методи на основі високорівневої стратегії [4].

Проблема підвищення ефективності пошуку рішення задачі управління запасами на основі мультиагентних метаевристичних методів представляється як проблема знаходження такого впорядкованого набору операторів $\{A_i\}$, ітеративне застосування якого забезпечує знаходження такого рішення x^*, при якому функція цілі $F(x^*) \to \min$ і $T \to \min$.

Як функцію цілі в роботі пропонується використовувати комбінацію двох функцій:

$$F(x,z) = F1(x,z) + F2(x,z) \to \min_x,$$

$$F1(x,z) = \sum_{m=1}^{M} w1 \cdot \max\{0, z^{\min} - (x_m + z_{m-1} - D_m)\},$$

$$F2(x,z) = \sum_{m=1}^{M} w2 \cdot \max\{0, x_m + z_{m-1} - D_m - z^{\max}\},$$

$$z_m = x_m + z_{m-1} - D_m,$$

де $F(\cdot)$ – функція цілі;

$F1(\cdot)$ – витрати від дефіциту товару;

$F2(\cdot)$ – витрати від зберігання товару;

$w1$ – прибуток від продажу однієї одиниці товару, задається;

$w2$ – витрати на зберігання однієї одиниці товару, задається;

x_m – кількість товару, що закуповується у постачальника протягом m-го етапу;

z_m – кількість запасів товару в кінці m-го етапу;

z_0 – вхідна кількість запасів товару, задається;

$z^{\min}, z^{\max}$ – мінімальна та максимальна кількість запасів товару в кінці кожного етапу, задається;

D_j – кількість товару, що продається протягом m-го етапу, задається;

M – кількість етапів.

Показник $z^{\min}$ може розглядатися як межа між чорною та червоною зонами буфера запасів. Показник $z^{\max}$ може розглядатися як межа між зеленою та синьою зонами буфера запасів. Потрапляння до чорної зони буфера запасів описується умовою $F1(x,z) > 0$. Попадання в червону/жовту/зелену зону буфера запасів описується умовою $F1(x,z) + F2(x,z) = 0$. Попадання в синю зону буфера запасів описується умовою $F2(x,z) > 0$.

Аналіз останніх досліджень і публікацій. Існуючі метаевристики мають один або більше з таких недоліків:

- не враховується номер ітерації для процесу пошуку рішення [5];
- не автоматизовано процедуру обчислення значень параметрів [6, 7];
- є лише проблемно-орієнтований або лише абстрактний опис методу [8, 9];
- не гарантується збіжність методу [10];
- недостатня точність методу [11];
- відсутня можливість вирішувати задачі умовної оптимізації [12, 13];
- вимога використовувати лише не бінарні чи бінарні потенційні рішення [14];
- обмежуються лише рівномірним розподілом [15, 16].

У зв'язку з цим постає задача побудови ефективних метаевристичних методів оптимізації. Однією з найпопулярніших метаевристик є алгоритм оптимізації рою частинок [17] та алгоритм рою штучних риб [18].

Постановка мети і задач дослідження. Метою роботи є розробка мультиагентних метаевристичних методів розв'язання задачі управління запасами для підвищення ефективності управління ланцюгами поставок. Для досягнення мети було поставлено та вирішено такі задачі:

- провести аналіз існуючих мультиагентних метаевристичних методів;
- запропонувати мультиагентний метаевристичний метод на основі оптимізації рою частинок;
- запропонувати паралельний алгоритм мультиагентного метаевристичного методу на основі оптимізації рою частинок;
- запропонувати мультиагентний метаевристичний метод на основі алгоритму рою штучних риб;
- запропонувати паралельний алгоритм мультиагентного метаевристичного методу на основі алгоритму рою штучних риб;
- виконати чисельні дослідження.

Виклад основного матеріалу.

1. Мультіагентний метаевристичний метод розв'язання задачі управління запасами на основі оптимізації рою частинок. Запропонований мультиагентний метаевристичний метод вирішення задачі управління запасами DPSO заснований на оптимізації рою частинок (PSO). Особливість запропонованого методу полягає в тому, що для керування параметрами методу, а також для забезпечення того, щоб на початкових ітераціях досліджувався весь простір пошуку, а на заключних ітераціях пошук ставав спрямованим і сходився, при генерації потенційних рішень враховується номер ітерації. Крім того, для модифікації потенційних рішень замість рівномірного розподілу використовуються розподіл Гауса та розподіл Коші, причому їх вибір залежить від номера ітерації. Також проводиться обчислення кількості запасів товару наприкінці кожного етапу.

Запропонований метод складається з наступних етапів:

1. Ініціалізація.

1.1. Завдання максимальної кількості ітерацій N, розміру рою K, кількості етапів M, мінімального та максимального значення параметра $\theta^{\min}, \theta^{\max}$, керуючого вкладом інерційної, когнітивної та соціальної компонент у швидкість частки, кількості товару, що продається, протягом усіх етапів $D = (D_{k1},...,D_{kM})$, вхідної кількості запасів товару z_0.

1.2. Створення початкового рою частинок $Q = \{(x_k, z_k, x_k^{best}, z_k^{best}, v_k)\}$.

1.2.1. Ініціалізація кожної позиції $x_k = (x_{k1},...,x_{kM})$, $x_{ki} = U(0,1)$, $k \in \overline{1,K}$, $i \in \overline{1,M}$, де $U(0,1)$ – функція, що повертає рівномірно розподілене випадкове число в діапазоні $[0,1]$.

1.2.2. Обчислення кількості запасів товару наприкінці кожного етапу $z_{kj} = x_{kj} + z_{k,j-1} - D_j$, $k \in \overline{1,K}$, $j \in \overline{1,M}$.

1.2.3. Ініціалізація персональної (локальної) найкращої позиції $x_k^{best} = x_k$, $k \in \overline{1,K}$.

1.2.4. Обчислення кількості запасів товарів за всіма періодами $z_k^{best} = z_k$, $k \in \overline{1,K}$.

1.2.5. Ініціалізація кожної швидкості $v_k = (v_{k1},...,v_{kM})$, $v_{ki} = 0$, $k \in \overline{1,K}$, $i \in \overline{1,M}$.

1.3. Обчислення глобальної кращої позиції з усіх ітерацій $k^* = \arg\max_{k \in \overline{1,K}} F(x_k, z_k)$, $x^* = x_{k^*}$, $z^* = z_{k^*}$.

1.4. Встановлення номера поточної ітерації n в одиницю.

2. Обчислення параметра, що управляє вкладом інерційної компоненти у швидкість частки на поточній ітерації $w(n) = \theta^{\min} + \left(\theta^{\max} - \theta^{\min}\right)\left(\frac{N-n}{N}\right)$.

3. Обчислення параметра, що управляє вкладом когнітивної компоненти у швидкість частки на поточній ітерації $\alpha_1(n) = \theta^{\min} + \left(\theta^{\max} - \theta^{\min}\right)\left(\frac{N-n}{N}\right)$.

4. Обчислення параметра, що управляє вкладом соціальної компоненти у швидкість частки на поточній ітерації $\alpha_2(n) = \theta^{\min} + \left(\theta^{\max} - \theta^{\min}\right)\left(\frac{N-n}{N}\right)$.

5. Оператор обчислення швидкості кожної частки

$$r1_{ki} = \left(n < \frac{N}{2} \right) C(0,1) + \left(n \geq \frac{N}{2} \right) N(0,1),\ k \in \overline{1,K},\ i \in \overline{1,M},$$

$$r2_{ki} = \left(n < \frac{N}{2} \right) C(0,1) + \left(n \geq \frac{N}{2} \right) N(0,1),\ k \in \overline{1,K},\ i \in \overline{1,M},$$

$v_k = (v_{k1},...,v_{kM})$, $v_{ki} = w(n)v_{ki} + \alpha_1(n)(x_{ki}^{best} - x_{ki})r1_i + \alpha_2(n)(x_i^* - x_{ki})r2_i$, $k \in \overline{1,K}$, $i \in \overline{1,M}$,

де $N(0,1)$ – функція, що повертає стандартно нормально розподілене випадкове число, $C(0,1)$ – функція, що повертає стандартно Коші розподілене випадкове число.

6. Оператор обчислення позиції кожної частки $x_k = x_k + v_k$, $k \in \overline{1,K}$.

7. Обчислення кількості запасів товару наприкінці кожного етапу $z_{kj} = x_{kj} + z_{k,j-1} - D_j$, $k \in \overline{1,K}$, $j \in \overline{1,M}$.

8. Обчислення персональної (локальної) кращої позиції кожної частки за всіма ітераціями. Якщо $F(x_k, z_k) > F(x_k^{best}, z_k^{best})$, то $x_k^{best} = x_k$, $z_k^{best} = z_k$, $k \in \overline{1,K}$.

9. Визначення штучної риби з найкращою позицією по всій поточній популяції $k^* = \arg\max_{k \in 1,K} F(x_k, z_k)$.

10. Обчислення глобальної кращої позиції з усіх ітерацій. Якщо $F(x_{k^*}, z_{k^*}) > F(x^*, z^*)$, то $x^* = x_{k^*}$, $z^* = z_{k^*}$

11. Умова зупину. Якщо $n < N$, то збільшити номер ітерації n на одиницю та перейти до блоку 2. Результатом є x^*.

2. Алгоритм мультиагентного метаевристичного методу розв'язання задачі управління запасами на основі оптимізації рою частинок. Для запропонованого методу DPSO розроблено алгоритм, призначений для реалізації на GPU за допомогою технології паралельної обробки інформації CUDA та представлений на рис. 1. Ця блок-схема функціонує наступним чином.

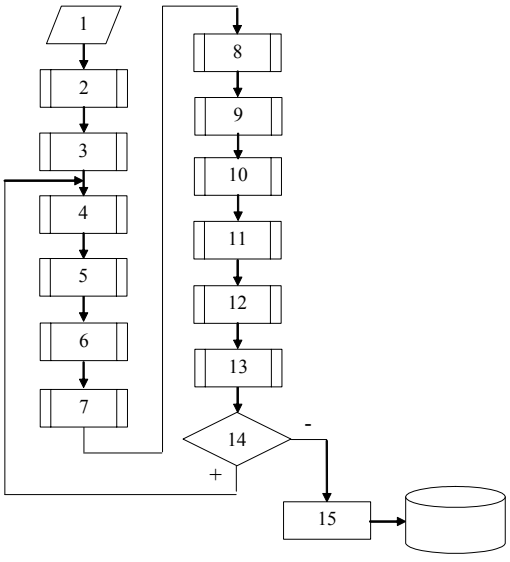

Рисунок 1 – Блок-схема алгоритму DPSO

Джерело: розроблено авторами

Крок 1 – Введення оператором максимальної кількості ітерацій N, розміру популяції K, кількості етапів M, мінімального та максимального значення параметра

$\theta^{\min},\theta^{\max}$, керуючого вкладом інерційної, когнітивної та соціальної компонент у швидкість частки, кількості товару, що продається, протягом усіх етапів $D=(D_{k1},...,D_{kM})$, вхідної кількості запасів товару z_0.

Крок 2 – Створення вхідної популяції $Q=\{(x_k,z_k,x_k^{best},z_k^{best},v_k)\}$.

Крок 3 – Обчислення глобальної кращої позиції з усіх ітерацій x^*.

Крок 4 – Встановлення номера ітерації $n=1$.

Крок 5 – Обчислення параметра, що управляє вкладом інерційної компоненти у швидкість частки на поточній ітерації $w(n)=\theta^{\min}+\left(\theta^{\max}-\theta^{\min}\right)\left(\frac{N-n}{N}\right)$.

Крок 6 – Обчислення параметра, що управляє вкладом когнітивної компоненти у швидкість частки на поточній ітерації $\alpha_1(n)=\theta^{\min}+\left(\theta^{\max}-\theta^{\min}\right)\left(\frac{N-n}{N}\right)$.

Крок 7 – Обчислення параметра, що управляє вкладом соціальної компоненти у швидкість частки на поточній ітерації $\alpha_2(n)=\theta^{\min}+\left(\theta^{\max}-\theta^{\min}\right)\left(\frac{N-n}{N}\right)$.

Крок 8 – Обчислення швидкості кожної k-ї частинки, використовуючи $K\cdot M$ ниток GPU, які згруповані в K блоків. Кожна нитка обчислює

$$r1_{ki}=\left(n<\frac{N}{2}\right)C(0,1)+\left(n\geq\frac{N}{2}\right)N(0,1),\ r2_{ki}=\left(n<\frac{N}{2}\right)C(0,1)+\left(n\geq\frac{N}{2}\right)N(0,1),$$

$$v_{ki}=w(n)v_{ki}+\alpha_1(n)(x_{ki}^{best}-x_{ki})r1_i+\alpha_2(n)(x_i^*-x_{ki})r2_i.$$

Крок 9 – Обчислення позиції кожної k-ї частинки, використовуючи $K\cdot M$ ниток GPU, які згруповані в K блоків. Кожна нитка обчислює $x_{ki}=x_{ki}+v_{ki}$.

Крок 10 – Обчислення кількості запасів товару в кінці кожного етапу на основі паралельної префіксної суми вектора $(x_{k1}+z_{k0}-D_1,x_{k2}-D_2,...,x_{kM}-D_M)$, використовуючи $K\cdot M$ ниток GPU, згрупованих в K блоків. Кожен блок обчислює z_k.

Крок 11 – Обчислення персональної (локальної) кращої позиції кожної k-ї частинки по всіх ітераціях, використовуючи K ниток GPU, які згруповані в 1 блок. Кожна нитка обчислює: Якщо $F(x_k,z_k)>F(x_k^{best},z_k^{best})$, то $x_k^{best}=x_k$, $z_k^{best}=z_k$.

Крок 12 – Визначення на основі паралельної редукції штучної риби з кращою позицією по всій поточній популяції, використовуючи K ниток GPU, які згруповані в 1 блок. Кожна нитка обчислює функцію цілі $F(x_k,z_k)$, $k^*=\arg\max_{k\in1,K}F(x_k,z_k)$.

Крок 13 – Обчислення кращої позиції з усіх ітерацій. Якщо $F(x_{k^*},z_{k^*})>F(x^*,z^*)$, то $x^*=x_{k^*}$, $z^*=z_{k^*}$.

Крок 14 – Умова зупину. Якщо $n<N$, то $n=n+1$ і перейти до кроку 5.

Крок 15 – Запис отриманої глобальної кращої позиції до бази даних.

3. Мультіагентний метаевристичний метод розв'язання задачі управління запасами на основі алгоритму рою штучних риб. Запропонований мультиагентний метаевристичний метод вирішення задачі про призначення DAFSA заснований на алгоритмі рою штучних риб (AFSA).

Особливістю запропонованого методу є те, що для керування параметрами методу, а також для забезпечення того, щоб на початкових ітераціях досліджувався весь простір пошуку, а на заключних ітераціях пошук ставав спрямованим і сходився при генерації потенційних рішень, враховується номер ітерації. Крім того, для

модифікації потенційних рішень замість рівномірного розподілу використовуються розподіл Гауса та розподіл Коші, причому їх вибір залежить від номера ітерації. Також проводиться обчислення кількості запасів товару наприкінці кожного етапу.

Запропонований метод складається з наступних етапів:

1. Ініціалізація.

1.1. Завдання максимальної кількості ітерацій N, розміру популяції K, кількості етапів M, мінімальної та максимальної кількості повторень $E^{\min}, E^{\max}$, мінімального та максимального кроку зміни вектора позиції $\beta^{\min}, \beta^{\max}$, $0 < \beta^{\min} < \beta^{\max} < 1$, мінімального та максимального параметра скупчення $\delta^{\min}, \delta^{\max}$, $0 < \delta^{\min} < \delta^{\max} < 1$, мінімального та максимального радіусу околу $R^{\min}, R^{\max}$, кількості товару, що продається, протягом усіх етапів $D = (D_{k1}, ..., D_{kM})$, вхідної кількості запасів товару z_0.

1.2. Створення початкової популяції риб $Q = \{(x_k, z_k)\}$.

1.2.1. Ініціалізація кожної позиції $x_k = (x_{k1}, ..., x_{kM})$, $x_{ki} = U(0,1)$, $k \in \overline{1,K}$, $i \in \overline{1,M}$, де $U(0,1)$ – функція, що повертає рівномірно розподілене випадкове число в діапазоні [0,1].

1.2.2. Обчислення кількості запасів товару наприкінці кожного етапу $z_{kj} = x_{kj} + z_{k,j-1} - D_j$, $k \in \overline{1,K}$, $j \in \overline{1,M}$.

1.3. Обчислення кращої позиції з усіх ітерацій $k^* = \arg\max_{k \in \overline{1,K}} F(x_k, z_k)$, $x^* = x_{k^*}$, $z^* = z_{k^*}$.

1.4. Встановлення номера поточної ітерації n в одиницю.

2. Обчислення радіусу околу на поточній ітерації $R(n) = R^{\min} + \left(R^{\max} - R^{\min}\right)\left(\frac{N-n}{N}\right)$.

3. Обчислення кроку зміни вектора позиції на поточній ітерації $\beta(n) = \beta^{\min} + \left(\beta^{\max} - \beta^{\min}\right)\left(\frac{N-n}{N}\right)$.

4. Обчислення параметра накопичення на поточній ітерації $\delta(n) = \delta^{\min} + \left(\delta^{\max} - \delta^{\min}\right)\left(\frac{N-n}{N}\right)$.

5. Обчислення кількості повторень на поточній ітерації $E(n) = E^{\min} + \left(E^{\max} - E^{\min}\right)\left(\frac{n}{N}\right)$.

6. Встановлення номера штучної риби k в одиницю.

7. Створення околу для k-ї штучної риби (відібрати найближчих за метрикою штучних риб з популяції Q) $U_{x_k} = \{\breve{x}_{kl} \mid \breve{x}_{kl} = x_h, 0 < \|x_h - x_k\| < R(n)\}$.

8. Обчислення кількості запасів товару в кінці кожного етапу для k-ї штучної риби $\breve{z}_{kli} = \breve{x}_{kli} + \breve{z}_{kl,i-1} - D_i$, $i \in \overline{1,M}$.

9. Оператор «пошук».

9.1. Встановлення лічильника повторень e в 1.

9.2. Обчислення позиції кожної k-ї штучної риби $r_{ki} = \left(n < \frac{N}{2}\right)C(0,1) + \left(n \geq \frac{N}{2}\right)N(0,1)$, $i \in \overline{1,M}$, $\hat{x}_{ki} = x_{ki} + r_{ki}R(n)$, $i \in \overline{1,M}$, де $N(0,1)$ – стандартний нормальний розподіл, $C(0,1)$ – стандартний Коші розподіл.

9.3. Обчислення кількості запасів товару в кінці кожного етапу

$\hat{z}_{ki} = \hat{x}_{ki} + \hat{z}_{k,i-1} - D_i,\ i \in \overline{1,M}$.

9.4. Обчислення позиції кожної k -ї штучної риби

$$r_{ki} = \left(n < \frac{N}{2}\right)C(0,1) + \left(n \geq \frac{N}{2}\right)N(0,1),\ i \in \overline{1,M},$$

$$x_k^1 = [x_{ki}^1],\ x_{ki}^1 = \begin{cases} x_{ki} + r_{ki}\beta(n)\dfrac{\hat{x}_{ki} - x_{ki}}{\| \hat{x}_k - x_k \|}, & F(\hat{x}_k, \hat{z}_k) > F(x_k, z_k) \\ x_{ki} + r_{ki}R(n), & \textit{інакше} \end{cases},\ i \in \overline{1,M}.$$

9.5. Обчислення кількості запасів товару в кінці кожного етапу $z_{ki}^1 = x_{ki}^1 + z_{k,i-1}^1 - D_i,\ i \in \overline{1,M}$.

9.6. Якщо $F(\hat{x}_k, \hat{z}_k) < F(x_k, z_k) \wedge e < E(n)$, то збільшити лічильник повторень e на одиницю і перейти до кроку 9.2.

10. Оператор «роїння».

10.1. Обчислення усередненої позиції по всьому околу кожної k -ї штучної риби $\breve{x}_k^{mean} = \dfrac{1}{|U_{x_k}|}\sum_{l=1}^{|U_{x_k}|}\breve{x}_{kl}$.

10.2. Обчислення кількості запасів товару в кінці кожного етапу $\breve{z}_{ki}^{mean} = \breve{x}_{ki}^{mean} + \breve{z}_{k,i-1}^{mean} - D_i,\ i \in \overline{1,M}$.

10.3. Обчислення позиції кожної k -ї штучної риби

$$r_{ki} = \left(n < \frac{N}{2}\right)C(0,1) + \left(n \geq \frac{N}{2}\right)N(0,1),\ i \in \overline{1,M},$$

$$x_{ki}^2 = \begin{cases} x_{ki} + r_{ki}\beta(n)\dfrac{\breve{x}_{ki}^{mean} - x_{ki}}{\| \breve{x}_k^{mean} - x_k \|}, & F(\breve{x}_k^{mean}, \breve{z}_k^{mean}) > F(x_k, z_k) \wedge \dfrac{|U_{x_k}|}{K} < \delta(n) \\ x_{ki}^1, & \textit{інакше} \end{cases},\ i \in \overline{1,M}.$$

10.4. Обчислення кількості запасів товару в кінці кожного етапу $z_{ki}^2 = x_{ki}^2 + z_{k,i-1}^2 - D_i,\ i \in \overline{1,M}$.

11. Оператор «слідування».

11.1. Обчислення штучної риби з кращою позицією по околу кожної k -ї штучної риби $l_k^* = \arg\min_l \{F(\breve{x}_{kl}, \breve{z}_{kl})\},\ l \in \overline{1,|U_{x_k}|},\ \breve{x}_k^{\min} = \breve{x}_{kl_k^*}\ \breve{z}_k^{\min} = \breve{z}_{kl_k^*}$.

11.2. Обчислення позиції кожної k -ї штучної риби

$$r_{ki} = \left(n < \frac{N}{2}\right)C(0,1) + \left(n \geq \frac{N}{2}\right)N(0,1),\ i \in \overline{1,M},$$

$$x_{ki}^3 = \begin{cases} x_{ki} + r_{ki}\beta(n)\dfrac{\breve{x}_{ki}^{\min} - x_{ki}}{\| \breve{x}_k^{\min} - x_k \|}, & F(\breve{x}_k^{\min}, \breve{z}_k^{\min}) > F(x_k, z_k) \wedge \dfrac{|U_{x_k}|}{K} < \delta(n) \\ x_{ki}^1, & \textit{інакше} \end{cases},\ i \in \overline{1,M}.$$

11.3. Обчислення кількості запасів товару в кінці кожного етапу $z_{ki}^3 = x_{ki}^3 + z_{k,i-1}^3 - D_i,\ i \in \overline{1,M}$.

12. Оператор «рух».

12.1. Обчислення позиції кожної k -ї штучної риби $r_{ki} = \left(n < \dfrac{N}{2}\right)C(0,1) + \left(n \geq \dfrac{N}{2}\right)N(0,1),\ i \in \overline{1,M},\ x_{ki}^4 = x_{ki} + r_{ki}R(n),\ i \in \overline{1,M}$.

12.2. Обчислення кількості запасів товару в кінці кожного етапу $z_{ki}^4 = x_{ki}^4 + z_{k,i-1}^4 - D_i$, $i \in \overline{1,M}$.

13. Оператор модифікації позиції k-ї штучної риби $l_k^* = \arg\max_l \{F(x_k^l, z_k^l)\}$, $l \in \overline{1,4}$, $x_k^{new} = x_k^{l_k^*}$, $z_k^{new} = z_k^{l_k^*}$.

14. Якщо $k < K$, то збільшити номер штучної риби k на 1 і перейти до кроку 6.

15. Оновлення популяції $x_k = x_k^{new}$, $z_k = z_k^{new}$, $k \in \overline{1,K}$.

16. Визначення штучної риби з найкращою позицією по всій поточній популяції $k^* = \arg\max_k F(x_k, z_k)$, $k \in \overline{1,K}$.

17. Обчислення кращої позиції з усіх ітерацій. Якщо $F(x_{k^*}, z_{k^*}) > F(x^*, z^*)$, то $x^* = x_{k^*}$, $z^* = z_{k^*}$.

18. Умова зупину. Якщо $n < N$, то збільшити номер ітерації n на одиницю та перейти до кроку 2. Результатом є x^*.

4. Алгоритм мультиагентного метаевристичного методу вирішення задачі управління запасами на основі алгоритму рою штучних риб. Для запропонованого методу DAFSA розроблено алгоритм, призначений для реалізації на GPU за допомогою технології паралельної обробки інформації CUDA та представлений на рис. 2. Ця блок-схема функціонує наступним чином.

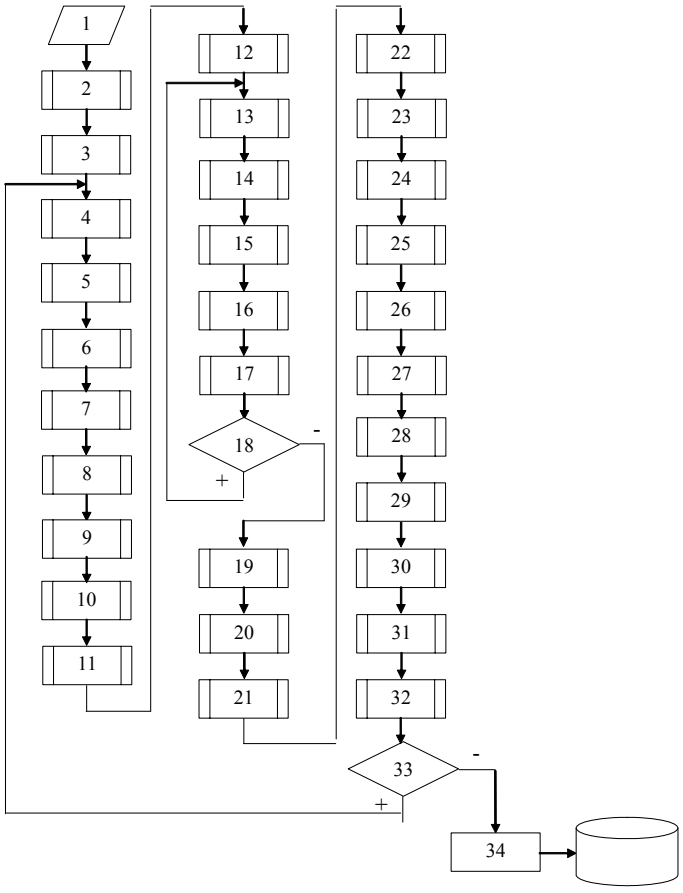

Рисунок 2 – Блок-схема алгоритму DAFSA

Джерело: розроблено авторами

Крок 1 – Введення оператором максимальної кількості ітерацій N, розміру популяції K, кількості етапів M, мінімальної та максимальної кількості повторень $E^{\min}, E^{\max}$, мінімального та максимального кроку зміни вектора позиції $\beta^{\min}, \beta^{\max}$, $0 < \beta^{\min} < \beta^{\max} < 1$, мінімального та максимального параметра скупчення $\delta^{\min}, \delta^{\max}$, $0 < \delta^{\min} < \delta^{\max} < 1$, мінімального та максимального радіусу околу $R^{\min}, R^{\max}$, кількості товару, що продається, протягом усіх етапів $D = (D_{k1}, ..., D_{kM})$, вхідної кількості запасів товару z_0.

Крок 2 – Створення вихідної популяції $Q = \{(x_k, z_k)\}$.

Крок 3 – Обчислення кращої позиції з усіх ітерацій x^*.

Крок 4 – Встановлення номера ітерації $n = 1$.

Крок 5 – Обчислення радіусу околу на поточній ітерації $R(n) = R^{\min} + \left(R^{\max} - R^{\min}\right)\left(\frac{N-n}{N}\right)$.

Крок 6 – Обчислення кроку зміни вектора позиції на поточній ітерації $\beta(n) = \beta^{\min} + \left(\beta^{\max} - \beta^{\min}\right)\left(\frac{N-n}{N}\right)$.

Крок 7 – Обчислення параметра накопичення на поточній ітерації $\delta(n) = \delta^{\min} + \left(\delta^{\max} - \delta^{\min}\right)\left(\frac{N-n}{N}\right)$.

Крок 8 – Обчислення кількості повторень на поточній ітерації $E(n) = E^{\min} + \left(E^{\max} - E^{\min}\right)\left(\frac{n}{N}\right)$.

Крок 9 – Обчислення на основі паралельної редукції відстані між кожною k-ою штучною рибою та всіма рибами популяції, використовуючи $K \cdot K \cdot M$ ниток GPU, які згруповані у $K \cdot K$ блоків. Кожен блок обчислює одну відстань $d_{hk} = \|x_h - x_k\|$.

Крок 10 – Створення околу для кожної k-ї штучної риби, використовуючи K ниток GPU, які згруповані в 1 блок. Кожна нитка обчислює $U_{x_k} = \{\breve{x}_{kl} \mid \breve{x}_{kl} = x_h, 0 < \|x_h - x_k\| < R(n)\}$.

Крок 11 – Обчислення кількості запасів товару в кінці кожного етапу на основі паралельної префіксної суми вектора $(\breve{x}_{k1} + z_{k0} - D_1, \breve{x}_{k2} - D_2, ..., \breve{x}_{kM} - D_M)$, використовуючи $K \cdot M$ ниток GPU, згрупованих у K блоків. Кожен блок обчислює $\breve{z}_k$.

Крок 12 – Встановлення лічильника повторень $e = 1$.

Крок 13 – Обчислення позиції кожної k-ї штучної риби відповідно до поведінки «рух», використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожна нитка обчислює $r_{ki} = \left(n < \frac{N}{2}\right)C(0,1) + \left(n \geq \frac{N}{2}\right)N(0,1)$, $\hat{x}_{ki} = x_{ki} + r_{ki}R(n)$.

Крок 14 – Обчислення кількості запасів товару в кінці кожного етапу на основі паралельної префіксної суми вектора $(\hat{x}_{k1} + z_{k0} - D_1, \hat{x}_{k2} - D_2, ..., \hat{x}_{kM} - D_M)$, використовуючи $K \cdot M$ ниток GPU, згрупованих у K блоків. Кожен блок обчислює $\hat{z}_k$.

Крок 15 – Обчислення на основі паралельної редукції відстані між кожними двома k-ми штучними рибами, використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожен блок обчислює одну відстань $d_k = \|\hat{x}_k - x_k\|$.

Крок 16 – Обчислення позиції кожної k-ї штучної риби відповідно до поведінки «пошук», використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожна нитка обчислює

$$r_{ki} = \left(n < \frac{N}{2}\right)C(0,1) + \left(n \geq \frac{N}{2}\right)N(0,1),\ x_{ki}^1 = \begin{cases} x_{ki} + r_{ki}\beta(n)\dfrac{\hat{x}_{ki} - x_{ki}}{d_k}, & F(\hat{x}_k, \hat{z}_k) > F(x_k, z_k) \\ x_{ki} + r_{ki}R(n), & \textit{інакше} \end{cases}.$$

Крок 17 – Обчислення кількості запасів товару в кінці кожного етапу на основі паралельної префіксної суми вектора $(x_{k1}^1 + z_{k0} - D_1, x_{k2}^1 - D_2, ..., x_{kM}^1 - D_M)$, використовуючи $K \cdot M$ ниток GPU, згрупованих у K блоків. Кожен блок обчислює z_k^1.

Крок 18 – Якщо $F(\hat{x}_k, \hat{z}_k) < F(x_k, z_k) \wedge e < E(n)$, то $e = e + 1$, перехід на крок 13.

Крок 19 – Обчислення усередненої позиції по всій околиці кожної k-ї штучної риби використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожна нитка обчислює $\breve{x}_{ki}^{mean} = \dfrac{1}{\left|U_{x_k}\right|}\sum_{l=1}^{\left|U_{x_k}\right|}\breve{x}_{kli}$.

Крок 20 – Обчислення кількості запасів товару в кінці кожного етапу на основі паралельної префіксної суми вектора $(\breve{x}_{k1}^{mean} + z_{k0} - D_1, \breve{x}_{k2}^{mean} - D_2, ..., \breve{x}_{kM}^{mean} - D_M)$, використовуючи $K \cdot M$ ниток GPU, згрупованих у K блоків. Блок обчислює $\breve{z}_k^{mean}$.

Крок 21 – Обчислення на основі паралельної редукції відстані між кожними двома k-ми штучними рибами, використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожен блок обчислює одну відстань $d_k = \| \breve{x}_k^{mean} - x_k \|$.

Крок 22 – Обчислення позиції кожної k-ї штучної риби відповідно до поведінки «роїння», використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожна нитка обчислює

$$r_{ki} = \left(n < \frac{N}{2}\right)C(0,1) + \left(n \geq \frac{N}{2}\right)N(0,1),$$

$$x_{ki}^2 = \begin{cases} x_{ki} + r_{ki}\beta(n)\dfrac{\breve{x}_{ki}^{mean} - x_{ki}}{d_k}, & F(\breve{x}_k^{mean}, \breve{z}_k^{mean}) > F(x_k, z_k) \wedge \dfrac{|U_{x_k}|}{K} < \delta(n) \\ x_{ki}^1, & \textit{інакше} \end{cases}.$$

Крок 23 – Обчислення кількості запасів товару в кінці кожного етапу на основі паралельної префіксної суми вектора $(x_{k1}^2 + z_{k0} - D_1, x_{k2}^2 - D_2, ..., x_{kM}^2 - D_M)$, використовуючи $K \cdot M$ ниток GPU, згрупованих у K блоків. Кожен блок обчислює z_k^2.

Крок 24 – Обчислення штучної риби з кращою позицією по околу кожної k-ї штучної риби, використовуючи K ниток GPU, які згруповані в 1 блок. Кожна нитка обчислює $l_k^* = \arg\min_l \{F(\breve{x}_{kl}, \breve{z}_{kl})\}$, $l \in \overline{1, \left|U_{x_k}\right|}$, $\breve{x}_k^{\min} = \breve{x}_{kl_k^*}$, $\breve{z}_k^{\min} = \breve{z}_{kl_k^*}$.

Крок 25 – Обчислення на основі паралельної редукції відстані між кожними двома k-ми штучними рибами, використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожен блок обчислює одну відстань $d_k = \| \hat{x}_k^{\min} - x_k \|$.

Крок 26 – Обчислення позиції кожної k-ї штучної риби відповідно до поведінки «слідування», використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожна нитка обчислює

$$r_{ki} = \left(n < \frac{N}{2} \right) C(0,1) + \left(n \geq \frac{N}{2} \right) N(0,1),$$

$$x_{ki}^3 = \begin{cases} x_{ki} + r_{ki}\beta(n)\dfrac{\breve{x}_{ki}^{\min} - x_{ki}}{d_k}, & F(\breve{x}_k^{\min}, \breve{z}_k^{\min}) > F(x_k, z_k) \wedge \dfrac{|U_{x_k}|}{K} < \delta(n) \\ x_{ki}^1, & \textit{інакше} \end{cases}.$$

Крок 27 – Обчислення кількості запасів товару в кінці кожного етапу на основі паралельної префіксної суми вектора $(x_{k1}^3 + z_{k0} - D_1, x_{k2}^3 - D_2, ..., x_{kM}^3 - D_M)$, використовуючи $K \cdot M$ ниток GPU, згрупованих у K блоків. Кожен блок обчислює z_k^3.

Крок 28 – Обчислення позиції кожної k-ї штучної риби відповідно до поведінки «рух», використовуючи $K \cdot M$ ниток GPU, які згруповані в K блоків. Кожна нитка обчислює $r_{ki} = \left(n < \frac{N}{2} \right) C(0,1) + \left(n \geq \frac{N}{2} \right) N(0,1)$, $x_{ki}^4 = x_{ki} + r_{ki} R(n)$.

Крок 29 – Обчислення кількості запасів товару в кінці кожного етапу на основі паралельної префіксної суми вектора $(x_{k1}^4 + z_{k0} - D_1, x_{k2}^4 - D_2, ..., x_{kM}^4 - D_M)$, використовуючи $K \cdot M$ ниток GPU, згрупованих у K блоків. Кожен блок обчислює z_k^4.

Крок 30 – Модифікація позиції кожної k-ї штучної риби на основі чотирьох поведінок, використовуючи K ниток GPU, які згруповані в 1 блок. Кожна нитка обчислює $l_k^* = \arg\max_l \{F(x_k^l, z_k^l)\}$, $l \in \overline{1,4}$, $x_k = x_k^{l_k^*}$, $z_k = z_k^{l_k^*}$.

Крок 31 – Визначення на основі паралельної редукції штучної риби з кращою позицією по всій поточній популяції, використовуючи K ниток GPU, ниток, які згруповані в 1 блок. Кожна нитка обчислює функцію цілі $F(x_k, z_k)$, $k^* = \arg\max_k F(x_k, z_k)$.

Крок 32 – Обчислення найкращою позицією щодо всіх ітерацій. Якщо $F(x_{k^*}, z_{k^*}) > F(x^*, z^*)$, то $x^* = x_{k^*}$, $z^* = z_{k^*}$.

Крок 33 – Умова зупину. Якщо $n < N$, то $n = n + 1$ і перейти до кроку 5.

Крок 34 – Запис отриманої кращої позиції з усіх ітерацій у базу даних.

Експерименти та результати. Чисельне дослідження запропонованих мультиагентних метаевристичних методів розв'язання задачі управління запасами проводилося на основі даних логістичної компанії «Ekol Ukraine» з використанням технології паралельної обробки інформації CUDA в пакеті Matlab.

У роботі для запропонованого методу DPSO розмір популяції $K = 3 \cdot M$, максимальна кількість ітерацій $N = 100$, мінімального та максимального значення параметра $\theta^{\min} = 0.1, \theta^{\max} = 0.98$, управляючого вкладом інерційної, когнітивної та соціальної компонент у швидкість частинки.

В роботі для запропонованого методу DAFSA розмір популяції $K = 3 \cdot M$, максимальна кількість ітерацій $N = 100$, мінімальна та максимальна кількість повторень $E^{\min} = 1, E^{\max} = N$, мінімальний та максимальний крок зміни вектора позиції $\beta^{\min} = 0.1, \beta^{\max} = 0.98$, мінімальний та максимальний параметр скупчення $\delta^{\min} = 0.5, \delta^{\max} = 0.98$, мінімальний та максимальний радіус околу $R^{\min} = \beta^{\min} \cdot M, R^{\max} = \beta^{\max} \cdot M$.

Результати порівняння запропонованих методів з традиційними з використанням критеріїв середньоквадратичної помилки (MSE) та обчислювальної складності (T) та

без використання паралелізму представлені в табл.1. При цьому враховувалася обчислювальна складність функції цілі $F(\cdot)$.

Таблиця 1 – Порівняння запропонованих мультиагентних метаевристичних методів із традиційними

Критерії	Мультиагентні метаевристичні методи			
	DPSO	PSO	DAFSA	AFSA
MSE	0.07	0.12	0.05	0.1
T без CUDA	$6 \cdot K \cdot M \cdot N$	$5 \cdot K \cdot M \cdot N$	$(16 + K + 5 \cdot E) \cdot K \cdot M \cdot N$	$(20 + K + 7 \cdot E) \cdot K \cdot M \cdot N$
T з CUDA	$N \cdot \log_2 M$	$N \cdot \log_2 M$	$(6 + 2 \cdot E) \cdot N \cdot \log_2 M$	$(6 + 2 \cdot E) \cdot N \cdot \log_2 M$

Джерело: розроблено авторами

Методи PSO і DPSO не враховують номер ітерації при генерації потенційних рішень популяції та модифікації параметрів методу, що знижує точність пошуку рішення (табл.1); не використовують комбінацію розподілів Гауса та Коші при генерації потенційних рішень, що знижує точність пошуку рішення (табл.1); не гарантують збіжність; не використовуються для задач динамічного програмування.

Запропоновані методи дозволяють усунути зазначені недоліки. Швидшим є метод DPSO, а точнішим є метод DAFSA.

Динамічна зміна параметрів запропонованих методів оптимізації забезпечує сильні зміни потенційного рішення на початкових ітераціях та слабкі зміни потенційного рішення на заключних ітераціях.

Висновки. У статті розглядається задача управління запасами як складова задачі ефективного управління ланцюгами постачання. Для вирішення цієї задачі було досліджено існуючі мультиагентні метаевристичні методи. Для підвищення якості розв'язання цієї задачі було обрано оптимізацію рою частинок та алгоритм рою штучних риб, які модифіковані впровадженням динамічних параметрів та розподілів Коші та Гауса. Для цих методів запропоновано паралельні алгоритми на основі технології CUDA. Це дозволило забезпечити високу швидкість та точність рішення.

Запропоновані методи призначені для програмної реалізації у пакеті Matlab з використанням Parallel Computing Toolbox, що прискорює процес пошуку рішення. Програмне забезпечення, що реалізує запропоновані методи, було розроблено та досліджено на основі даних логістичної компанії «Ekol Ukraine».

Проведені експерименти підтвердили працездатність розробленого програмного забезпечення та дозволяють рекомендувати його для використання на практиці при вирішенні задач управління ланцюгами постачання. Перспективи подальших досліджень полягають у тому, щоб перевірити запропоновані методи на більш широкому наборі тестових баз даних.

Список літератури

1. Cox J. F., Schleher J. G. Theory of Constraints Handbook . New York: NY, McGraw-Hill, 2010. 1175 p.
2. Cluster Policy of Innovative Development of the National Economy: Integration and Infrastructure Aspects : monograph / under the editorship of professor Svitlana Smerichevska. Poznań: Wydawnictwo naukowe WSPIA, 2020. 380 p.
3. Diagnostic Rule Mining Based on Artificial Immune System for a Case of Uneven Distribution of Classes in Sample / S. Subbotin, A. Oliinyk, V. Levashenko, E. Zaitseva . *Communications*. 2016. Vol.3. P.3-11.
4. Nakib A., Talbi El-G. Metaheuristics for Medicine and Biology . Berlin: Springer-Verlag, 2017. 211 p.

5. Engelbrecht A. P. Computational Intelligence: an introduction . Chichester, West Sussex: Wiley & Sons, 2007. 630 p. DOI: 10.1002/9780470512517.
6. Yang X.-S. Nature-inspired Algorithms and Applied Optimization . Charm: Springer, 2018. 330 p. DOI: 10.1007/978-3-642-29694-9
7. Martí R. Pardalos P. M.,. Resende M. G. C. Handbook of Heuristics . Charm: Springer, 2018. 1289 p. DOI: 10.1007/978-3-319-07124-4
8. Blum C., Raidl G. R. Hybrid Metaheuristics. Powerful Tools for Optimization . Charm: Springer, 2016. 157 p. DOI: 10.1007/978-3-319-30883-8
9. Yang X.-S. Optimization Techniques and Applications with Examples . Hoboken, New Jersey: Wiley & Sons, 2018. 364 p. DOI: 10.1002/9781119490616
10. Chopard B., Tomassini M. An Introduction to Metaheuristics for Optimization . New York: Springer, 2018. 230 p. DOI: 10.1007/978-3-319-93073-2
11. Radosavljević J. Metaheuristic Optimization in Power Engineering . New York: The Institution of Engineering and Technology, 2018. 536 p. DOI: 10.1049/PBPO131E
12. Du K.-L., Swamy M. N. S. Search and Optimization by Metaheuristics. Techniques and Algorithms Inspired by Nature . Charm: Springer, 2016. 434 p. DOI: 10.1007/978-3-319-41192-7
13. Bozorg Haddad O., Solgi M., Loaiciga H. Meta-heuristic and Evolutionary Algorithms for Engineering Optimization . Hoboken, New Jersey: Wiley & Sons, 2017. 293 p. DOI: 10.1002/9781119387053
14. Alba E., Nakib A., Siarry P.. Metaheuristics for Dynamic Optimization . Berlin: Springer-Verlag, 2013. 398 p. DOI: 10.1007/978-3-642-30665-5
15. Method For Parametric Identification Of Gaussian Mixture Model Based On Clonal Selection Algorithm / E. Fedorov, V. Lukashenko, T. Utkina, A. Lukashenko, K. Rudakov . *CEUR Workshop Proceedings*. 2019. Vol. 2353. P. 41-55.
16. Grygor O. O. Optimization method based on the synthesis of clonal selection and annealing simulation algorithms / O. O. Grygor, E. E. Fedorov, T. Yu. Utkina, A. G. Lukashenko, K. S. Rudakov, D. A. Harder et al . *Radio Electronics, Computer Science, Control*. 2019. № 2. P. 90-99. DOI: 10.15588/1607-3274-2019-2-10.
17. Spears W. M., Green D. T., Spears D. F. Biases in particle swarm optimization . *International Journal of Swarm Intelligence Research*. 2010. Vol. 1, No. 2. P. 34–57. DOI: 10.4018/jsir.2010040103
18. Review of Artificial Fish Swarm Optimization Methods and Applications / M. Neshat, A. Adeli, G. Sepidnam, M. Sargolzaei, A. N. Toosi. *International Journal on Smart Sensing and Intelligent Systems*. 2012. Vol. 5, No. 1. P. 107–148. DOI: 10.21307/ijssis-2017-474

References

1. Cox, J. F. & Schleher, J. G. (2010). *Theory of Constraints Handbook*. New York: NY, McGraw-Hill [in English].
2. Smerichevska, S. (Eds.). (2020). *Cluster Policy of Innovative Development of the National Economy: Integration and Infrastructure Aspects: monograph*. Poznań: Wydawnictwo naukowe WSPIA [in English].
3. Subbotin, S., Oliinyk, A., Levashenko, V. & Zaitseva, E. (2016). Diagnostic Rule Mining Based on Artificial Immune System for a Case of Uneven Distribution of Classes in Sample. *Communications, Vol.3, 3-11* [in English].
4. Nakib, A. & Talbi, El-G. (2017). *Metaheuristics for Medicine and Biology*. Berlin: Springer-Verlag [in English].
5. Engelbrecht, A. P. (2007). *Computational Intelligence: an introduction*. Chichester, West Sussex: Wiley & Sons. DOI: 10.1002/9780470512517 [in English].
6. Yang, X.-S. (2018). *Nature-inspired Algorithms and Applied Optimization*. Charm: Springer. DOI: 10.1007/978-3-642-29694-9 [in English].
7. Martí, R., Pardalos, P. M. & Resende, M. G. C. (2018). *Handbook of Heuristics*. – Charm: Springer. DOI: 10.1007/978-3-319-07124-4 [in English].
8. Blum, C. & Raidl, G. R. (2016). *Hybrid Metaheuristics. Powerful Tools for Optimization*. Charm: Springer. DOI: 10.1007/978-3-319-30883-8 [in English].
9. Yang, X.-S. (2018). *Optimization Techniques and Applications with Examples*. Hoboken, New Jersey: Wiley & Sons. DOI: 10.1002/9781119490616 [in English].
10. Chopard, B. & Tomassini, M. (2018). *An Introduction to Metaheuristics for Optimization*. New York: Springer. DOI: 10.1007/978-3-319-93073-2 [in English].
11. Radosavljević, J. (2018). *Metaheuristic Optimization in Power Engineering*. New York: The Institution of Engineering and Technology. DOI: 10.1049/PBPO131E [in English].

12. Du, K.-L. & Swamy, M. N. S. (2016). *Search and Optimization by Metaheuristics. Techniques and Algorithms Inspired by Nature*. Charm: Springer. DOI: 10.1007/978-3-319-41192-7 [in English].
13. Bozorg Haddad, O., Solgi, M. & Loaiciga, H. (2017). *Meta-heuristic and Evolutionary Algorithms for Engineering Optimization*. Hoboken, New Jersey: Wiley & Sons. DOI: 10.1002/9781119387053 [in English].
14. Alba, E., Nakib, A. & Siarry, P. (2013). *Metaheuristics for Dynamic Optimization*. Berlin: Springer-Verlag. DOI: 10.1007/978-3-642-30665-5 [in English].
15. Fedorov, E., Lukashenko, V., Utkina, T., Lukashenko, A. & Rudakov K. (2019). Method For Parametric Identification Of Gaussian Mixture Model Based On Clonal Selection Algorithm. *CEUR Workshop Proceedings, Vol. 2353,* 41-55 [in English].
16. Grygor, O. O., Fedorov, E. E., Utkina, T. Yu., Lukashenko, A. G., Rudakov, K. S. & Harder, D. A. et al. (2019). Optimization method based on the synthesis of clonal selection and annealing simulation algorithms. *Radio Electronics, Computer Science, Control,* 2*,* 90-99. DOI: 10.15588/1607-3274-2019-2-10 [in English].
17. Spears, W. M., Green, D. T. & Spears, D. F. (2010). Biases in particle swarm optimization. *International Journal of Swarm Intelligence Research, Vol. 1, 2,* 34–57. DOI: 10.4018/jsir.2010040103 [in English].
18. Neshat, M., Adeli, A., Sepidnam, G., Sargolzaei, M. & Toosi A. N. (2012). A Review of Artificial Fish Swarm Optimization Methods and Applications. *International Journal on Smart Sensing and Intelligent Systems, Vol. 5, 1,*107–148. DOI: 10.21307/ijssis-2017-474 [in English].

Eugene Fedorov, Prof., DSc., **Olga Nechyporenko**, Assoc. Prof., PhD tech. sci.
Cherkasy State Technological University, Cherkasy, Ukraine

Multi-Agent Metaheuristic Methods for Solving the Inventory Management Problem

Currently, the problem of insufficient efficiency of supply chain management is relevant. One of the problems solved within the limits of the specified problem is the optimization problem of inventory management. Optimization methods that find an approximate solution using a directed search have a high probability of reaching a local extremum. Optimization methods that find an exact solution have a high computational complexity. Random search methods do not guarantee convergence. In this connection, there is a problem of insufficient efficiency of optimization methods, which needs to be solved.

The article considers the task of inventory management as a component of the task of effective supply chain management. To solve this problem, the existing multi-agent metaheuristic methods were investigated. To improve the quality of solving this problem, particle swarm optimization and artificial fish swarm algorithm were chosen, which are modified by introducing dynamic parameters and Cauchy and Gaussian distributions. Parallel algorithms based on CUDA technology are proposed for these methods. This made it possible to ensure high speed and accuracy of the decision.

The proposed methods are designed for software implementation in the Matlab package using the Parallel Computing Toolbox, which speeds up the process of finding a solution. The software that implements the proposed methods was developed and researched based on the data of the logistics company "Ekol Ukraine". The conducted experiments confirmed the functionality of the developed software and allow us to recommend it for practical use in solving supply chain management problems. Prospects for further research are to test the proposed methods on a wider set of test databases.

inventory management problem, supply chain management, multi-agent metaheuristic methods, particle swarm optimization, artificial fish swarm algorithm

Одержано (Received) 24.11.2022 *Прорецензовано (Reviewed) 06.12.2022*
Прийнято до друку (Approved) 29.12.2022