УДК 519.7, 519.8

Г. Г. Цегелик, Р. П. Краснюк
Львівський національний університет імені Івана Франка

ПРОБЛЕМА ОПТИМАЛЬНОЇ ОБРОБКИ ЗАДАЧ У ВУЗЛАХ РОЗПОДІЛЕНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Досліджено питання оптимальної обробки задач у вузлах розподіленої інформаційної системи на основі математичної моделі, що належить до класу задач бікластеризації, для якої сформульовано оптимізаційну задачу із дробово-лінійною цільовою функцією. Виконано процедуру лінеаризації цільової функції та наведено загальну схему ітераційного процесу побудови розв'язку оптимізаційної задачі. На кожному кроці ітерації результат можна отримати з використанням як точного методу гілок та меж, так і генетичного алгоритму. Наведено варіанти відповідних методів, у яких для стратегій галуження та обчислення верхньої межі у методі гілок і меж враховано структуру моделі. Для генетичного алгоритму запропоновано використання параметрів самонавчання алгоритму, що забезпечує корекцію популяцій у напрямку найкращої пристосованості.

Ключові слова: оптимізація, математичне моделювання, задача бікластеризації, розподілені інформаційні системи, метод гілок та меж, генетичний алгоритм.

The problem of optimal processing tasks in the nodes of a distributed information system on the basis of a mathematical model belonging to a class of two-clustering problems, for which an optimization problem with a fractional linear target function is formulated, was investigated. The procedure of linearization of the target function is carried out and the general scheme of the iterative process of constructing an optimization problem solution is presented, where at each step of the iteration the result can be obtained using both the exact method of branches and bounds and using the genetic algorithm. The variants of the corresponding methods were given, where the structure of the model was taken into account for branching strategies and calculating the upper limit in the method of branches and boundaries. For the genetic algorithm, it was proposed to use the parameters self-training of algorithm, which provides correction of populations in the direction of the best adaptability.

Key words: optimization, mathematical modeling, two-clustering problem, distributed information systems, method of branches and bounds, genetic algorithm.

Вступ

Під час проектування розподілених інформаційних систем, а також розроблення алгоритмів й програмного забезпечення, що підтримує їх функціонування, значну увагу зосереджують на формуванні математичних моделей оптимізації процесів функціонування інформаційної системи (ІС). Щодо математичних моделей та їх застосування у програмному забезпеченні ставлять низку вимог, насамперед щоб зміна структури та алгоритмів поведінки моделі не спричинила погіршення характеристик моделі, які визначають її здатність адекватно описати предметну область дослідження та задовольнити встановлені чи очікувані потреби застосування.

Розподілені інформаційні системи мають низку характерних особливостей, до основних з яких можемо зарахувати властивість масштабованості (нарощування обчислювальних потужностей) та відмовостійкості (незмінності виконання розподілених обчислень ІС у разі виходу з ладу обчислювальних вузлів). Це означає, що модель розподіленої ІС під час виконання операцій з додавання чи видалення її компонентів повинна зберігати свої характеристики й надавати користувачам вірогідні результати.

У разі зміни топології розподіленої обчислювальної системи змінюється і структура даних алгоритмів, які виконуються у цьому розподіленому обчислювальному середовищі. Як наслідок, виникає необхідність й у зміні алгоритмів та відповідних структур даних, що описують поведінку модельованих об'єктів.

Оскільки існує тенденція до збільшення кількості обчислювальних кластерів з диференційованою складністю обчислень, актуальним, з одного боку, є використання більш оптимізовано налаштованих процедур планування, що надає змогу ефективно формувати завдання на ресурси, а з іншого боку – забезпечення високого навантаження наявних ресурсів. Тому побудова та дослідження оптимізаційних моделей, пов'язаних з функціонуванням розподілених інформаційних систем, є актуальною та важливою проблемою, що і розглядатиметься у роботі.

Актуальність завдань оптимального розподілу та обробки даних у розподілених інформаційних системах зумовила появу численних публікацій, у яких розглядають широке коло науково-практичних завдань. Зокрема, формалізовану постановку задачі оптимізації розміщення даних системи автоматизованого проектування (САПР) у хмарній інфраструктурі, відповідно до запропонованого ієрархічного подання структури складної САПР, розглянуто у роботі [1]. Визначено та описано основні етапи та проблеми розв'язання цієї задачі на різних рівнях організації системи. Пропонується метод і розробляється алгоритм організації розміщення даних САПР у вузлах "хмари" на основі синергетичного підходу із застосуванням модифікованої еволюції Шмальгаузера.

Координацію розв'язків щодо розподілу ресурсів на основі генетичного алгоритму досліджено у роботі [2]. Розроблено метод виконання завдань розподілу ресурсів та синхронізації технологічних процесів у розподілених інформаційних системах. Питання оптимізації розподілу запитів між кластерами відмовостійких інформаційних систем розглянуто у роботах [3, 4]. Запропоновано оцінку надійності розподілених обчислювальних систем, що передбачає перерозподіл запитів за зміни потоків запитів, відмов та відімкнення вузлів системи, які об'єднані у сукупність кластерів. Сформульовано та розв'язано задачу оптимізації процесу перерозподілу запитів між кластерами з урахуванням його впливу на затримки обслуговування та надійність системи.

Питань дослідження розподілених математичних моделей моделювання паралельних архітектур стосується робота [5], а розроблення комплексу математичних моделей підтримки стійкості функціонування розподілених інформаційних систем висвітлено у [6].

Моделі та методи підвищення пропускної спроможності розподілених телекомунікаційних систем високодоступних хмарних сховищ розроблено у праці [7]. Створено архітектуру сховища, що дало змогу збільшити продуктивність та відмовостійкість системи загалом, введенням блока вибору оптимального шлюзу для передавання даних.

Відмінність нашої роботи від результатів інших авторів полягає у формуванні математичної моделі оптимального розміщення копій спеціалізованого програмного забезпечення, призначеного для оброблення певного класу задач та розподілу паралельно виконуваних задач сукупності завдань у вузлах інформаційної системи. Запропонована математична модель належить до типу дробово-лінійних бікластеризованих оптимізаційних задач, для побудови точного розв'язку якої запропоновано та досліджено модифікацію методу гілок та меж. Наближений розв'язок цієї оптимізаційної задачі побудовано з використанням генетичного алгоритму, в якому для підвищення обчислювальної ефективності запропоновано введення параметрів самонавчання алгоритму, що забезпечує корекцію популяцій у напрямку найкращої пристосованості.

Математична постановка проблеми

Дуже часто під час розподілених обчислень виникає проблема оптимального розміщення копій спеціалізованого програмного забезпечення, призначеного для обробки певного класу задач та розподілу паралельно виконуваних задач сукупності завдань у вузлах ІС. Очевидно, що копії спеціалізованого програмного забезпечення можуть бути завжди наявні в усіх вузлах розподіленої ІС і тут як розміщення приймається ситуація "оплати за використання". Це означає, що

розглядається розміщення m спеціалізованих програмних комплексів, на яких можуть бути оброблені l задач сукупності завдань у вузлах інформаційної системи за умов мінімізації переміщень між вузлами та максимізації оброблення задач у вузлах розподіленої інформаційної системи.

Дані для задачі подаються у вигляді матриці M розміру $m \times l$, яка містить нулі та одиниці. Елемент матриці m_{ij} дорівнює одиниці, якщо задача j повинна оброблятися програмним комплексом i. Необхідно мінімізувати кількість нулів всередині вузлів (програмний комплекс не має задач для оброблення) та кількість одиниць поза вузлами (переміщення задач між вузлами). Введемо у розгляд шукані характеристики $x_{ik} = \{0,1\}$, тоді одиницю вибирають, якщо спеціалізований програмний комплекс i розміщено у вузлі k, та $y_{jk} = \{0,1\}$, у такому разі одиницю вибирають для задачі j, яка оброблятиметься у вузлі k. Наведемо математичну постановку задачі: знайти максимум цільової функції

$$F\left(\{x_{ik}\},\{y_{jk}\}\right) = \frac{N_1}{N + N_0} \to \max, \tag{1}$$

за умов

$$\sum_{k=1}^{n} x_{ik} = 1,\ i = \overline{1,m};\ \sum_{k=1}^{n} y_{jk} = 1,\ j = \overline{1,l}, \tag{2}$$

(умова, щоб кожен спеціалізований програмний комплекс та задача були прив'язані до одного вузла мережі)

$$\sum_{i=1}^{m}\sum_{j=1}^{l} x_{ik} y_{jk} \geq \sum_{i=1}^{m} x_{ik};\ \sum_{i=1}^{m}\sum_{j=1}^{l} x_{ik} y_{jk} \geq \sum_{j=1}^{l} y_{jk},\ k = \overline{1,n}, \tag{3}$$

(умова відсутності вузлів, що містять тільки спеціалізовані програмні комплекси чи тільки задачі), де N – кількість одиниць у вихідній матриці M,

$N_1 = \sum_{k=1}^{n}\sum_{i=1}^{m}\sum_{j=1}^{l} m_{ij} x_{ik} y_{jk}$ – кількість одиниць всередині вузлів,

$N_0 = \sum_{k=1}^{n}\sum_{i=1}^{m}\sum_{j=1}^{l} (1 - m_{ij}) x_{ik} y_{jk}$ – кількість нулів всередині вузлів,

$n = \min\{m,l\}$ – максимально можлива кількість вузлів, які будуть залучені до оброблення задач. Зауважимо, що цільовою функцією у (1) є групова ефективність, вперше запропонована у роботі [8].

Проблема (1)–(3) є задачею нелінійного програмування, яку з використанням методу Дінкельбаха [9] зведемо до задачі цілочислового лінійного програмування. Для цього розглядають нову цільову функцію:

$$F_L\left(\{x_{ik}\},\{y_{jk}\}\right) = N_1 - a\left(N + N_0\right) \to \max, \tag{4}$$

де a на кожній ітерації обчислювального алгоритму дорівнює поточному кращому розв'язку для вихідної задачі (1)–(3). На першій ітерації $a = a_0$ визначається як значення групової ефективності тривіального випадку, за якого усі програмні комплекси та задачі призначені для одного вузла мережі:

$$a_0 = \sum_{i=1}^{m}\sum_{j=1}^{l} m_{ij} \Big/ \left(N + \sum_{i=1}^{m}\sum_{j=1}^{l} (1 - m_{ij}) \right). \tag{5}$$

Обґрунтуємо обчислювальний алгоритм оптимізаційної задачі (1)–(3).

Теорема. Якщо оптимальний розв'язок задачі (2)–(4) F_L^* дорівнює нулю, то знайдене значення a і є оптимальним розв'язком (1)–(3).

Доведення. Справді, якщо оптимальний розв'язок задачі (2)–(4) F_L^* дорівнює нулю, то для довільного допустимого розв'язку $F_L \le F_L^* = 0$ і, як наслідок, $N_1 - a(N + N_0) \le 0$, або $F = N_1/(N + N_0) \le a$. Це означає, що знайдене значення a і є оптимальним розв'язком (1)–(3), бо кожний допустимий розв'язок $F \le a$.

Наслідок. Якщо $F_L^* > 0$, то $N_1 - a(N + N_0) > 0$ і $F = N_1/(N + N_0) > a$. Тоді знайдений розв'язок $F_L^* > 0$ є кращим за поточне значення a. Здійснюється перепризначення $a = F_L^*$ і розв'язання лінеаризованої задачі (2)–(4) повторюється за нового значення a.

Зауваження 1. Значення F_L^* не може бути від'ємним, оскільки в цьому випадку $F_L \le F_L^* < 0$ і, як наслідок, $N_1 - a(N + N_0) < 0$ або $F = N_1/(N + N_0) < a$. Остання умова не може справдитися, оскільки вже є допустимий розв'язок $F = a$, вибраний у першій ітерації алгоритму.

Зауваження 2. Немає потреби шукати за кожної ітерації алгоритму оптимальне значення F_L^*. Достатньо знайти довільне допустиме значення $F_L > 0$, що буде використане в операції перепризначення $a = F_L$. Як наслідок, обчислювальний алгоритм реалізовуватиметься з використанням методу гілок та меж, опис якого наведемо далі.

Алгоритм гілок та меж ітераційного кроку

Як показано вище, нелінійна оптимізаційна проблема (1)–(3) зведена до задачі лінійного цілочислового програмування (2)–(4), розв'язок якої будується як ітераційний процес за вибору параметра a. Початкове значення параметра a задається виразом (5), кожне наступне – умовою $a = F_L^*$, де F_L^* – оптимальне значення лінеаризованої задачі (2)–(4), коли $0 < F_L^* < e$, де e – похибка обчислень. Як зазначено у зауваженні 2, немає потреби знаходити екстремальне значення F_L^*, достатньо знайти довільне допустиме значення $F_L > 0$, що буде використане на наступній ітерації обчислювального алгоритму. І тут застосування методу гілок та меж має істотні переваги над іншими варіантами неявного перебору.

Оскільки галуження здійснюватиметься за спеціалізованими програмними комплексами, розміщеними у вузлах розподіленої інформаційної системи, то для опису гілок введемо у розгляд вектор MV, розмірності m, котрий містить розподіл програмних комплексів за вузлами ІС. Елемент MV_i, $i = \overline{1,m}$ міститиме номер вузла, в який призначається i-й програмний комплекс, тобто $MV_i \in \{1, 2, ..., n\}$.

Стратегія галуження. Нехай k – кількість вузлів у поточному частковому розв'язку. Алгоритм починається із розподілу спеціалізованого програмного комплексу із номером 1 в один із вузлів. Потім вибирають наступний нерозподілений програмний комплекс, що призначається в наявні вузли чи у новий, створений для цього комплексу вузол із номером $k + 1$.

Листки дерева пошуку містять повні розв'язки, а інші вершини – часткові. Повне дерево пошуку без використання границь залежатиме тільки від кількості програмних комплексів. Крім того, вузол з номером $k + 2$ у частковому розв'язку ніколи не використовується, якщо попередній вузол ще не був використаний (він не був вибраний для підключення програмного комплексу) в поточному розв'язку. Дотримання цієї умови гарантує відсутність розв'язків, що відрізняються лише нумерацією вузлів.

Як наслідок, стратегією галуження є вибір тієї гілки, верхня межа якої буде найбільшою.

Стратегія обчислення верхньої межі. Для отримання оцінки зверху для кожного часткового розв'язку необхідно вирішити проблему призначення сукупності задач у вузли інформаційної системи. Для цього необхідно врахувати ситуацію, коли варіант, за якого задача призначається у вузол, в якому немає програмного комплексу для обробки її даних, не має змісту. Крім того, вузол,

в якому розміщено програмний комплекс, повинен мати хоча б одну задачу для виконання. Функція цілі моделі (2)–(4) є лінійною, а внесок призначення однієї задачі не залежить від призначення іншої задачі. За цим висновком можемо спочатку оптимально вибрати одну задачу для кожного вузла, що забезпечить максимальний загальний внесок у цільову функцію цих вибраних задач, а решту задач призначити у кращі для них вузли вже без урахування будь-яких обмежень. Якщо розглянути вираз для лінеаризованої цільової функції (4), то вартість призначення дорівнює додаванню задачі у функцію цілі зі знаком "мінус", тобто $aN_0 - N_1$. Для балансування матриці витрат цього завдання (тобто подання її у квадратній формі) додають фіктивні вузли, для яких усі вартості призначення є рівними $-\infty$.

Для побудови розв'язку задачі про призначення використано результати праць [10, 11]. Зауважимо, що за умови, коли усі спеціалізовані програмні комплекси є розподіленими за вузлами інформаційної системи, розв'язок задачі про призначення надає допустимий розв'язок для вихідної оптимізаційної задачі (2)–(4), тобто отримується оптимальний розв'язок F_L^*.

Формування стратегії обробки задач, близької до оптимальної, з використанням генетичного алгоритму

Очевидно, що застосування методу гілок та меж може бути неефективним за великої розмірності розглянутої математичної моделі. Тому альтернативним підходом до побудови розв'язку оптимізаційних задач є застосування наближених методів, які дають змогу отримати результат, близький до оптимального, з прийнятною для практичного використання точністю. Одним з таких варіантів є застосування генетичного алгоритму. Далі наведемо модифікацію генетичного алгоритму, яка містить параметри самонавчання алгоритму, що забезпечують корекцію популяцій у напрямку найкращої пристосованості.

Крок 1. Формування початкової популяції.

1.1. Задається номер популяції $np = 0$, максимальна кількість популяцій $pmax$, номер ітерації $itern = 0$.

1.2. Задається розмір популяції q та випадковим вибором формується початкова популяція розміру $2q$. Для цього за допомогою рівномірного розподілу генерується q матриць розміру $m \times n$ та q матриць розміру $l \times n$ – особин популяції, елементи яких – випадкові значення з одиничного відрізка $(0,1)$. Однак усі елементи матриці мають бути цілими числами з множини $\{0,1\}$. Тому в роботі запропоновано вводити матриці самонавчання алгоритму $l = \{l_{ik}\}$ та $m = \{m_{jk}\}$, що слугуватимуть засобом цілеспрямованого впливу на характеристики пошуку. Елементи цієї матриці спочатку також розраховують довільно із відрізка $(0,1)$ із застосуванням рівномірного розподілу. Зауважимо, що за такого підходу до кодування розв'язку особиною популяції є матриця, стовпчики якої є хромосомами особини.

1.3. Для кожної матриці – особини з популяції перед розрахунком цільової функції застосовуватиметься процедура дискретизації: якщо значення елемента матриці є меншим за відповідне значення матриці корекції чи дорівнює йому, то значення зменшується до нуля, в протилежному випадку – зростає до одиниці. Якщо за умовою обмежень оптимізаційної задачі тільки одне значення в стовпчику (хромосомі) матриці популяції повинно дорівнювати одиниці, то збільшується до одиниці тільки перше значення стовпчика, що є більшим за відповідне значення елемента матриці самонавчання.

1.4. Для кожної дискретизації матриці з популяції здійснюється розрахунок значення цільової функції (чи функції пристосування) та перевіряється задоволення додаткових обмежень, що сформовані в оптимізаційній моделі. Якщо у сформованій популяції відсутні варіанти, що задовольняють обмеження оптимізаційної моделі, то популяцію необхідно перестворити заново.

Крок 2. Селекція.

Вибір особин для схрещування запропоновано здійснювати із застосуванням стратегії панміксії – випадкового рівномірного вибору двох батьківських особин – двох матриць з популяції.

Крок 3. Схрещування – формування нових нащадків у популяції.

Для формування нащадків запропоновано застосування одноточкового та багатоточкового схрещування. За першого варіанта кожна пара стовпчиків (хромосом) матриць батьків випадково розривається в одній точці та формуються два нові стовпчики матриць нащадків із використанням двох частин хромосоми кожної батьківської особини: перший нащадок отримує, наприклад, кожну верхню частину вектора-стовпчика першого батька і нижню другого, тоді ж як другий нащадок – навпаки.

За другого варіанта схрещування випадково вибирають дві чи більше точок розриву для кожної пари хромосом з матриць батьків, і нащадки отримають нові хромосоми, що складатимуться з сегментів, розміщених між цими точками. Зауважимо, що варіант вибору кількості сегментів розбиття також визначається випадково для кожної нової популяції

Результатом цього кроку є два нащадки, отримані з використанням вибраних на другому кроці алгоритму батьківських особин.

Крок 4. Мутація – це перетворення хромосоми, що випадково змінює один чи декілька з її генів. Застосовується для підтримання різноманітності хромосом у популяції.

У роботі запропоновано для мутації використовувати оператор інверсії, за яким кожен стовпчик (хромосома) матриці особини випадково ділиться на дві частини, які потім обмінюються місцями – нижня частина стає верхньою, і навпаки.

Як наслідок, результатом цього кроку є два нащадки-мутанти, одержані застосуванням оператора мутації до двох нових особин, отриманих на третьому кроці алгоритму.

Крок 5. Формування нової популяції.

З отриманих на попередньому кроці особин вибирають одну, результат застосування якої до цільової функції є найкращим. Вона замінить у вихідній популяції особину, найгірше пристосовану. Зауважимо, що розрахунок значень пристосовуваності виконують із використанням матриці самонавчання алгоритму за схемою, наведеною на *кроці 1.3*.

Далі перевіряють умови:

якщо $itern < q$, то прийняти $itern = itern + 1$ та перейти до *кроку 2*;

якщо $itern = q$, то прийняти $np = np + 1$ та перейти до *кроку 6*.

Крок 6. Перевірка умови завершення роботи генетичного алгоритму.

Умовою завершення роботи генетичного алгоритму є формування заданої кількості популяцій $np > pmax$:

якщо умову не виконано, то приймаємо $itern = 1$ та переходимо до *кроку 2*. При цьому відбувається уточнення матриці самонавчання алгоритму. Для цього значення матриць $\{l_{ik}\}$ та $\{m_{jk}\}$ перераховуються за формулою:

$$\frac{l_{ik}}{m_{jk}} = \begin{cases} 0, & \frac{l_{ik}}{m_{jk}} + (1 - F_0 / F_1) \le 0; \\ \frac{l_{ik}}{m_{jk}} + (1 - F_0 / F_1), & 0 < \frac{l_{ik}}{m_{jk}} + (1 - F_0 / F_1) < 1; \\ 1 & \frac{l_{ik}}{m_{jk}} + (1 - F_0 / F_1) \ge 1, \end{cases}$$

де значення F_1 – найкраще значення функції пристосовуваності для поточного покоління популяції, F_0 – відповідно найкраще значення функції пристосовуваності для попереднього покоління. Очевидною є вимога побудови двох поколінь популяцій перед застосуванням цієї процедури уточнення для матриць самонавчання. Введення такої залежності робить процес самонавчання генетичного алгоритму чутливішим до змін якості розв'язку. Справді, за малих змін функції пристосовуваності значення елементів матриць змінюються незначно, і навпаки; якщо умову завершення роботи виконано, то як розв'язок (наближений) вибираємо особину із найкращою пристосованістю з поточної популяції, застосувавши процедуру дискретизації (з *кроку 1.3*).

Як показали проведені числові експерименти побудови розв'язку задачі (2)–(4), введення у генетичний алгоритм процедури самонавчання алгоритму дало змогу в середньому зменшити кількість поколінь під час формування популяцій на двадцять відсотків, а кількість задач, для яких розв'язок не знайдено за 10000 поколінь популяцій, зменшився на п'ять відсотків.

Висновки та перспективи подальших наукових розвідок

У роботі сформовано математичну модель процесу оптимальної обробки сукупності задач у вузлах розподіленої інформаційної системи. Сформульовану оптимізаційну проблему зведено до задачі нелінійного програмування, яку з використання методу Дінкельбаха зведено до задачі цілочислового лінійного програмування. Для побудови розв'язку задачі наведено загальну схему алгоритму, який може бути реалізований з використанням як евристичних, так і точних методів, зокрема із застосуванням методу гілок та меж. Стратегії галуження та обчислення верхньої межі наведено у роботі. Оскільки за великої розмірності моделі обчислювальна ефективність точного методу знижується, то запропоновано варіант генетичного алгоритму, що містить параметри самонавчання алгоритму, які забезпечують корекцію популяцій у напрямку найкращої пристосованості. Запропоновані у роботі алгоритми мають і самостійне значення, бо невеликою модифікацією їх можна адаптувати до розв'язання інших нелінійних оптимізаційних задач. Простота реалізації та обчислювальна ефективність сформованих алгоритмів підтверджується низкою числових експериментів. Предметом подальших досліджень з цієї проблематики є інтеграція оптимізаційної моделі та запропонованих обчислювальних алгоритмів у програмне забезпечення, яке здійснює керування обчислювальними ресурсами у розподілених інформаційних системах.

1. Коваленко О. С. Постановка задачи размещения данных в "облаке" // Искусственный интеллект. – 2011. – № 4. – С. 54–64. 2. Дубовой В. М., Байас М. М. Координация решений о распределении ресурсов на основе генетического алгоритма // Інформаційні технології та комп'ютерна інженерія. – 2014. – № 2. – С. 4–12. 3. Богатырев А. В., Голубев И. Ю., Богатырев С. В., Богатырев В. А. Оптимизация распеределения запросов между кластерами отказоустойчивой вычислительной системы // Научно-технический вестник информационных технологий, механики и оптики. – 2013. – № 3 (85). – С. 77–82. 4. Golubev I. Y., Bogatyrev S. V., Bogatyrev V. A. Optimization and the Process of Task Distribution between Computer System Clusters // Automatic Control and Computer Sciences. – 2012. – Vol. 46, Issue 3. – P. 103–111. 5. Kvasnica P., Kvasnica I. Distributed Mathematical Model Simulation on a Parallel Architecture // Journal of Computing and Information Technology. – 2012. – Vol. 20, Issue 2. – P. 61–68. 6. Ivutin A. N., Yesikov D. O. Complex of mathematical models to ensuring sustainability of the distributed information systems // Embedded Computing (MECO), 2015 4th Mediterranean Conference. – Budva, Montenegro, 2015. – P. 239–246. 7. Струбицький Р. П. Методи та алгоритми побудови хмаркових сховищ даних на основі розподілених телекомунікаційних систем: автореф. дис. канд. техн. наук: 05.12.02 – телекомунікаційні системи та мережі / Національний університет "Львівська політехніка". – Львів, 2017. – 20 с. 8. Kumar K. R., Chandrasekharan M. P. Grouping efficacy: A quantitative criterion for goodness of block diagonal forms of binary matrices in group technology // Intern. Jour. Production Research, 1990. – Vol. 28, No 2. – P. 233–243. 9. Dinklebach W. On nonlinear fractional programming // Management Science. – 1967. – Vol. 13. – P. 492–498. 10. Недобачій С. І., Гвоздик Д. М. Новий метод розв'язання задачі про призначення // Математичні машини і системи. – 2010. – № 1. – С. 55–59. 11. Jonker R., Volgenant A. A. Shortest augmenting path algorithm for dense and sparse linear assignment problems // Computing. – 1987. – Vol. 38. – P. 325–340.