УДК 519.8
DOI https://doi.org/10.32689/maup.it.2022.2.11

Дмитро ОЛЬХОВСЬКИЙ
кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій, Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, індекс 36000 (dmitriy@olhovsky.name)

ORCID: 0000-0003-0313-6977

Олена ОЛЬХОВСЬКА
кандидат фізико-математичних наук, завідувач кафедри комп'ютерних наук та інформаційних технологій, Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, індекс 36000 (lena@olhovsky.name)

ORCID: 0000-0001-5366-5995

Оксана ЧЕРНЕНКО
кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій, Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, індекс 36000 (oksanachernenko7@gmail.com)

ORCID: 0000-0002-9084-0999

Тетяна ПАРФЬОНОВА
кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій, Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, індекс 36000 (tapa.poltava@gmail.com)

ORCID: 0000-0001-9343-2061

Тетяна ЧІЛІКІНА
кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій, Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, індекс 36000 (tv.0502@gmail.com)

ORCID: 0000-0001-8234-9131

Dmytro OLHOVSKIY
Candidate of Physical and Mathematical Sciences, Associate Professor, Associate Professor at Department of Computer Science and Information Technology, Poltava University of Economics and Trade, Koval str., 3, Poltava, Ukraine, postal code 36000 (dmitriy@olhovsky.name)

Olena OLKHOVSKA
Candidate of Physical and Mathematical Sciences, Associate Professor, Associate Professor at Department of Computer Science and Information Technology, Poltava University of Economics and Trade, Koval str., 3, Poltava, Ukraine, postal code 36000 (lena@olhovsky.name)

Oksana CHERNENKO
Candidate of Physical and Mathematical Sciences, Associate Professor, Associate Professor at Department of Computer Science and Information Technology, Poltava University of Economics and Trade, Koval str., 3, Poltava, Ukraine, postal code 36000 (oksanachernenko7@gmail.com)

Tatyana PARFONOVA
Candidate of Physical and Mathematical Sciences, Associate Professor, Associate Professor at Department of Computer Science and Information Technology, Poltava University of Economics and Trade, Koval str., 3, Poltava, Ukraine, postal code 36000 (tapa.poltava@gmail.com)

Tatyana CHILIKINA
Candidate of Physical and Mathematical Sciences, Associate Professor, Associate Professor of Department of Computer Science and Information Technology, Poltava University of Economics and Trade, Koval str., 3, Poltava, Ukraine, postal code 36000 (tv.0502@gmail.com)

Бібліографічний опис статті: Ольховський, Д., Ольховська, О., Черненко, О., Парфьонова, Т., Чілікіна, Т. (2022). Програмний комплекс для розв'язування евклідових комбінаторних оптимізаційних задач точними та наближеними методами. *Інформаційні технології та суспільство*, 2 (4), 78–87. DOI: https://doi.org/10.32689/maup.it.2022.2.11

Bibliographic description of the article: Olhovskiy, D., Olkhovska, O., Chernenko, O., Parfonova, T., Chilikina, T. (2022). Prohramnyi kompleks dlia rozviazuvannia evklidovykh kombinatornykh optymizatsiinykh zadach tochnymy ta nablyzhenymy metodamy [Software package for solving euclidean combinatorial optimization problems with accurate and approximate methods]. *Informatsiini tekhnolohii ta suspilstvo – Information technology and society*, 2 (4), 78–87. DOI: https://doi.org/10.32689/maup.it.2022.2.11

ПРОГРАМНИЙ КОМПЛЕКС ДЛЯ РОЗВ'ЯЗУВАННЯ ЕВКЛІДОВИХ КОМБІНАТОРНИХ ОПТИМІЗАЦІЙНИХ ЗАДАЧ ТОЧНИМИ ТА НАБЛИЖЕНИМИ МЕТОДАМИ

Задачі комбінаторної оптимізації набувають усе більшого поширення на практиці. Це зумовлено тим, що велика кількість прикладних задач описується моделями, в яких розв'язок визначений на комбінаторних множинах. Розв'язування таких задач вимагає розробки нових або модифікації вже наявних методів, написання алгоритмів та їх програмної реалізації. ***Мета роботи*** *– створити програмний продукт для розв'язування евклідових комбінаторних оптимізаційних задач точними та наближеними методами. При цьому важливим є врахування структури комбінаторних конфігурацій, зокрема, із застосуванням теорії графів. Важливим, окрім розробки нових математичних підходів, є врахування сучасного стану обчислювальної техніки – наявність потужних багатопроцесорних систем.*

Методологія*. Для розробки програми було вибрано мову програмування високого рівня Object Pascal середовища програмування Delphi.*

Наукова новизна. *У роботі проведено опис розробленого програмного комплексу, який реалізує методи для розв'язування задач комбінаторної оптимізації різними методами.*

Представлений програмний продукт дає змогу розв'язувати задачі лінійного програмування методом комбінаторного відсікання на основі алгоритму Кармаркара для умовних лінійних задач комбінаторної оптимізації на переставленнях. На відміну від відомих методів комбінаторного відсікання для задач на вершинно розташованих множинах, тут допоміжна задача лінійного програмування розв'язується не певною різновидністю симплекс-методу, а поліноміальним алгоритмом Кармаркара.

Розроблений програмний продукт реалізує також другий метод комбінаторного відсікання в умовних лінійних задачах на вершинно розташованих множинах з виключенням виродженості в допоміжних задачах лінійного програмування. Також знайдено розв'язок задачі комбінаторної оптимізації модифікованим методом з можливістю приєднання необхідних обмежень та відкидання зайвих. Такий підхід дозволив значно збільшити вимірність задач, що можуть бути розв'язані.

Висновки. *Завдяки програмному комплексу стало можливим розв'язування комбінаторних задач оптимізації із застосуванням представлення комбінаторного многогранника у вигляді графа значних вимірностей.*

Створений програмний продукт дозволив провести чисельні експерименти всіма вище зазначеними методами для підтвердження їх практичної ефективності та коректності.

Ключові слова: *комбінаторний многогранник, поліноміальний алгоритм Кармаркара, вершинно розташовані множини, комбінаторне відсікання.*

SOFTWARE PACKAGE FOR SOLVING EUCLIDEAN COMBINATORIAL OPTIMIZATION PROBLEMS WITH ACCURATE AND APPROXIMATE METHODS

Combinatorial optimization problems are becoming more common in practice. This is due to the fact that a large number of applied problems are described by the models in which the solution is defined on combinatorial sets. Solving such problems requires the development of new methods or modification of existing methods, writing algorithms and their software implementation. ***The aim of the work*** *is to create a software product for solving Euclidean combinatorial optimization problems with accurate and approximate methods. It is important to take into account the structure of combinatorial configurations, in particular, using graph theory. In addition to developing new mathematical approaches, it is essential to consider the current state of computer technology – the presence of powerful multiprocessor systems.*

Methodology. *The high-level programming language Object Pascal of the Delphi programming environment was chosen for the development of the program.*

Scientific novelty. *The paper describes the developed software package that implements methods for solving combinatorial optimization problems by different methods.*

The presented software product allows to solve linear programming problems by the method of combinatorial clipping on the basis of Carmarcar's algorithm for conditional linear problems of combinatorial optimization on permutations. Unlike

the known methods of combinatorial clipping for problems on vertically located sets, here the auxiliary problem of linear programming is solved not by a certain kind of simplex method, but by the polynomial Carmarcar algorithm.

The developed software product also implements the second method of combinatorial clipping in conditional linear problems on vertically located sets with the exception of degeneracy in auxiliary linear programming problems. We also found a solution to the problem of combinatorial optimization by a modified method with the ability to add the necessary constraints and discard unnecessary ones. This approach has significantly increased the dimensionality of the tasks that can be solved.

Conclusions. *Due to the software package, it has become possible to solve combinatorial optimization problems using the representation of combinatorial polygon in the form of a graph of significant dimensions.*

The created software product allowed to conduct numerous experiments with all the above methods to confirm their practical effectiveness and correctness.

Key words: *combinatorial polyhedron, Carmarkar polynomial algorithm, vertically arranged sets, combinatorial clipping.*

Постановка проблеми у загальному вигляді та її зв'язок з важливими науковими чи практичними завданнями. Дослідження задач евклідової комбінаторної оптимізації є передумовою успішного моделювання важливих економічних, природних, соціальних та інших процесів. Актуальним є і подальше дослідження підходу до розв'язування комбінаторних оптимізаційних задач, що ґрунтується на ідеях методів відсікання для задач оптимізації лінійних функцій з лінійними додатковими обмеженнями, в яких допустима точка має переставні властивості. Є вже низка розроблених методів для розв'язування комбінаторних оптимізаційних задач, для яких актуальною є розробка програмних комплексів, що реалізують алгоритми методів для розв'язування такого класу задач.

Аналіз останніх досліджень і публікацій. Теорія і методи евклідової комбінаторної оптимізації включають систематичне вивчення властивостей комбінаторних множин та їх дослідження, модифікацію відомих та розробку нових методів розв'язування оптимізаційних задач комбінаторного типу. Велика кількість публікацій, що присвячена евклідовій комбінаторній оптимізації [1–10], свідчить про необхідність та важливість подібних досліджень у галузі розробки програмних пакетів, що реалізують відомі методи для розв'язання задач евклідової комбінаторної оптимізації.

У роботах [1; 2; 8] розглядається загальна задача евклідової комбінаторної оптимізації на множині переставлень. У [2; 9] запропоновано та обґрунтовано метод комбінаторного відсікання на основі алгоритму Кармаркара для умовних лінійних задач комбінаторної оптимізації на переставленнях. Одержано симплексну форму переставного многогранника, яка необхідна для застосування алгоритму Кармаркара у разі розв'язування допоміжних задач лінійного програмування в методі комбінаторного відсікання. При цьому розв'язана проблема побудови суміжних точок для розв'язку ДЗЛП та побудова нерівності-відсікання. У роботах [3–6] запропоновані методи розв'язку комбінаторних задач та елементи їх програмних реалізацій. Частково проведено порівняння цих методів та представлені результати числових експериментів.

Постановка завдання. Для доведення ефективної роботи та практичного застосування розроблених методів необхідно створити їх практичну реалізацію, що дозволить провести чисельні експерименти для підтвердження практичної ефективності та коректності отриманих результатів.

Виклад основного матеріалу дослідження. Для розробки програми було вибрано мову програмування високого рівня *Object Pascal* середовища програмування *Delphi*.

Створене програмне забезпечення призначене для функціонування в операційній системі *MS Windows*.

У разі розробки програми ставилося завдання створити універсальний програмний комплекс для розв'язування задач лінійної комбінаторної оптимізації як відомими методами, так і розробленими методами та алгоритмами в роботах [1–6; 9].

Для можливості проведення тестування розроблених та наявних методів у різних умовах було реалізовано механізм роботи програми з раціональними дробами. Така реалізація дозволяє легко проводити переключення між апаратом дійсних чисел та роботою з використанням дробів, не змінюючи алгоритми самих методів. Для цього використано відповідні директиви компілятора:

- *{$Define REAL}* – для використання дійсних чисел;
- *{$Define FRACT}* – для застосування механізму роботи з раціональними дробами.

З метою ефективного процесу розробки програми було введено уніфікований тип *TNumber*, який використовується для проведення всіх обчислень і залежно від директиви компілятора є або дійсним типом, або раціональним дробом.

Практичні чисельні експерименти показали, що використання дійсних чисел у роботі програми дозволяє досягти більшої швидкодії та економії ресурсів оперативної пам'яті, в іншому випадку – використання апарату дробів – дозволило повністю усунути можливі похибки округлення чисел при обчисленнях за рахунок більших витрат обчислювальних ресурсів комп'ютерної системи.

Програмна реалізація створена в рамках єдиного проєкту з розділенням на окремі модулі та класи.

Загальну архітектуру створеної програми можна представити у вигляді трьох основних функціональних блоків:

- блоку взаємодії з користувачем та даними;
- блоку реалізації допоміжного функціоналу та оголошень змінних та типів;
- обчислювального блоку.

Для наочного зображення загальної архітектури наведено її схематичне представлення на рисунку 1. Наведена програмна архітектура дозволила виокремити функціональні частини програми у окремі модулі та класи програми для забезпечення їхньої ефективної взаємодії та керування роботою програми.

Опишемо структуру та особливості кожного зі створених функціональних блоків програми детально.

Модуль форми *MainForm* разом із його програмною частиною *MainUnit* в єдиному однойменному класі об'єднує всі інші класи та модулі, які реалізовані в програмі. В такому модулі представлений уніфікований доступ до всіх функціональних можливостей програми, а також реалізована взаємодія з користувачем: задання вхідних даних (як користувачем з клавіатури чи файлу, так і з використанням генерації даних) для методів та алгоритмів, виведення повної інформації про процес розв'язування задачі та результатів обчислень, керування параметрами окремих алгоритмів та методів для зміни процесу розв'язування задачі тощо. Разом з модулями *HtmlOut*, *InOut* та *Generation* вони утворюють блок взаємодії з користувачем та обробки даних.

У зв'язку зі значною кількістю приватних змінних та функцій прототип класу наводити не будемо, зупинимось більш детально тільки на методі *RunMain*.

Метод *RunMain (Method: integer)* реалізовує об'єднаний інтерфейс роботи з усіма обчислювальними методами програми, забезпечує взаємодію інтерфейсу користувача з обчислювальним блоком програми. Вхідним параметром для функції є ідентифікатор методу, який необхідно використовувати на такому етапі. На початку роботи методу проводиться перевірка коректності задання всіх даних, необхідних для проведення обчислень, і у разі відсутності помилок керування передається відповідному модулю для проведення обчислень. Після завершення обчислень метод проводить збереження отриманих результатів шляхом виведення їх на екран або записування у файл з відповідним повідомленням користувача.

З метою виведення результатів обчислень (як остаточних, так і проміжних) було створено два модулі: *HtmlOut* та *InOut*.

Модуль *HtmlOut* містить у собі методи, необхідні для збереження деталізованих результатів обчислень у файл формату *html (HyperText Markup Language)*:

- метод PrepareHTMLFile(FName: string) проводить підготовку файлу для можливості виведення у нього необхідної інформації, вхідним параметром є повний шлях до збережуваного файлу;
- метод STableToHTML(var STable: STArray; var OFName: TextFile) виконує збереження однієї ітерації роботи обчислювального алгоритму у файл;
- процедура FinishHTMLFile виконує закриття відкритих тегів у файлі та закриває режим запису.

Програмний модуль *InOut* використовується для відображення отриманих результатів на екран, а також для експорту результатів у формат *MS Excel*. В ньому, зокрема, міститься реалізація таких методів:

- процедура PrintV1 – призначена для відображення результатів обчислень на екран у табличному представленні;
- метод ExportSGToExcelV1 призначений для експорту результатів у табличний процесор MS Excel для можливості проведення детального аналізу отриманих результатів обчислень.

Методи модулю *Generation* використовуються для генерації лінійних задач оптимізації на множині переставлень. Функціонал такого модулю дозволяє генерувати необхідну кількість задач з можливістю задання всіх вхідних даних задачі: множини перестановок, коефіцієнтів функції цілі, проміжків, на яких необхідно проводити генерацію коефіцієнтів задач. Особливістю модулю є те, що у разі генерації задач використовується псевдовипадковий генератор, який дозволяє для вказаних параметрів задачі відновити її за порядковим номером.

В іншому структурному блоці розміщені всі оголошення глобальних типів, констант та змінних, а також об'єднані методи реалізації допоміжних алгоритмів для проведення обчислень. Розглянемо деякі з них більш детально.

У класі *TGraph* реалізовані підходи роботи методів оптимізації з використанням графу переставного многогранника. Наведемо прототип цього класу (опустимо блок оголошення приватних змінних):

```
  TGraph = class
    private
     procedure Clear; dynamic;
     function NodeExists(aNum: integer): boolean; overload;
     function NodeExists(aP: array of integer): integer; overload;
    public
     constructor Create;
     destructor Destroy; override;
      procedure Init(aDim: integer; aP, aG: array of TNumber; aF: array of
TNumber); dynamic;
      function AddNode(aNum: integer; aDir: TDirection; aAddNext: boolean =
False): integer; dynamic;
     procedure DelNode(aNum: integer; aDir: TDirection);
     procedure GoNext(aDir: TDirection); dynamic;
     property MaxFuncVal: TNumber read cMaxFuncVal;
```

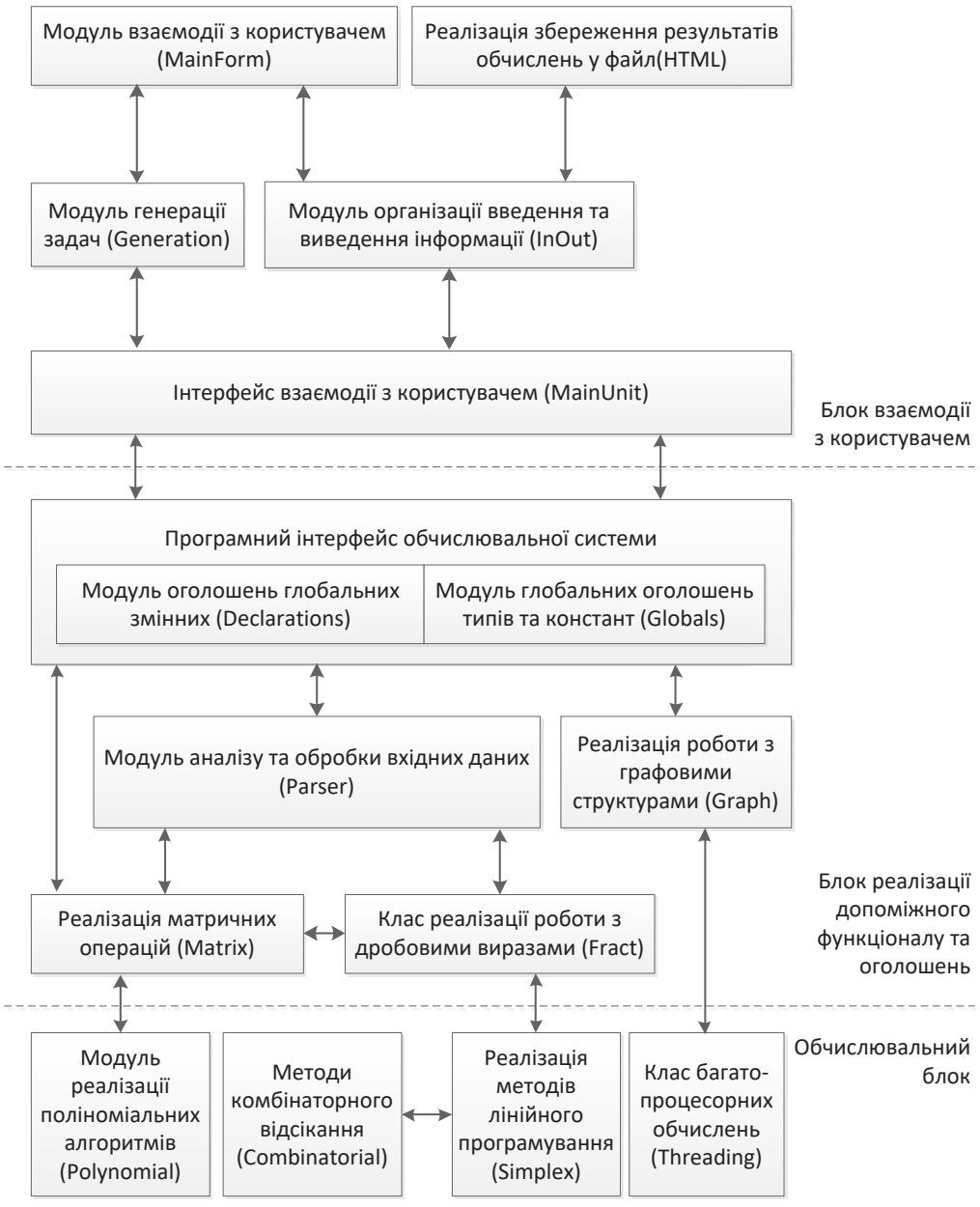

Рис. 1. Архітектура програмної реалізації

```
    property MaxFuncNode: integer read cMaxFuncNode;
    property ErrCode: byte read cErrCode;
    function OutCurrNode: string;
    function GetCurrPoint: FixedNumArray;
  end;
```

Конструктор класу *Create* не приймає вхідних параметрів, проводить задання початкових значень необхідним полям класу.

Метод *Init* здійснює ініціалізацію класу. На вхід метод приймає вимірність задачі (*aDim*), значення векторів множини переставлень, індекси початкового переставлення та функцію цілі (*aP*, *aG*, *aF*).

Процедура *AddNode* проводить додавання нового вузла в граф, при цьому за необхідності (у разі активації параметру *aAddNext*) можливе додавання всіх сусідніх вершин з поточною. На вході метод приймає параметр *aDir*, який вказує, в якому напрямку (вростання або спадання значення функції цілі) необхідно проводити додавання суміжних вершин.

Метод *DelNode* проводить видалення вказаної вершини з графу. При цьому проводиться об'єднання всіх суміжних з нею вершин ребрами.

Процедура *GoNext* проводить перехід до наступної суміжної вершини в графі з урахуванням напрямку оптимізації функції цілі (параметр *aDir*).

Властивості, описані в прототипі класу, використовуються для отримання поточного стану графу:

- властивість MaxFuncVal повертає поточне оптимальне значення функції цілі у вершині графу;
- з використанням властивості MaxFuncNode можна отримати коефіцієнти поточної вершини графу, яка забезпечує оптимальне значення функції цілі на такому етапі;
- властивість ErrCode сигналізує про помилку в процесі роботи з графом (наприклад, про відсутність вершин у графі або неможливість переходу до суміжної вершини).

Метод *GetCurrPoint* повертає значення (коефіцієнти) поточної вершини графу.

На основі класу *TGraph* був розроблений дочірній клас *TKGraph*, який реалізує механізми роботи з частковим графом переставного многогранника. Наведемо його прототип:

```
  TKGraph = class (TGraph)
    private
     cQueue: TList<TSNode>;

     procedure Clear; override;
    public
     constructor Create;
     destructor Destroy; override;
      procedure Init(aDim: integer; aP, aG: array of TNumber; aF: array of
TNumber); override;
      function AddNode(aNum: integer; aDir: TDirection; aAddNext: boolean =
False): integer; override;
     procedure GoNext(aDir: TDirection); override;
  end;
```

Більшість методів та властивостей клас наслідує у батьківського класу *TGraph*. Відрізняється тільки безпосередня реалізація деяких з них для врахування особливостей роботи з частковим графом переставного многогранника, зокрема це методи: *Init*, *AddNode*, *GoNext.*

Клас реалізації операцій з раціональними дробами *Fract* об'єднує у собі всі необхідні для роботи програми методи. Прототип класу виглядає таким чином:

```
  TFract = record
    num,
    denom: integer;

  // арифметичні оператори
  class operator Add(a, b: TFract): TFract;
    class operator Subtract(a, b: TFract): TFract;
    class operator Multiply(a, b: TFract): TFract;
    class operator Divide(a, b: TFract): TFract;

    // унарні оператори
    class operator Negative(a: TFract): TFract;
```

```
    class operator Inc(a: TFract): TFract;
  class operator Dec(a: TFract): TFract;
    class operator Round(a: TFract): integer;

    // логічні оператори
    class operator Equal(a, b: TFract): boolean;
    class operator NotEqual(a, b: TFract): boolean;
    class operator GreaterThan(a, b: TFract): boolean;
    class operator GreaterThan(a: TFract; b: real): boolean;
    class operator GreaterThanOrEqual(a, b: TFract): boolean;
    class operator LessThan(a, b: TFract): boolean;
    class operator LessThan(a: TFract; b: real): boolean;
    class operator LessThanOrEqual(a, b: TFract): boolean;

    // приведення типів
    class operator Implicit(a: TFract): integer;
    class operator Implicit(a: integer): TFract;
    class operator Implicit(a: TFract): real;
    class operator Implicit(a: string): TFract;
    class operator Implicit(a: TFract): string;

      // обчислення абсолютного значення
    class function NumAbs1(a: TFract): TFract; static;
    class function Abs(a: real): real; overload; static;

    private
     class procedure Reduct(var a: TFract); static;
  end;
```

У такому класі описані перевантажені оператори для виконання арифметичних операцій із значеннями дробового типу:

- оператор Add призначений для додавання двох дробів;
- метод Subtract виконує віднімання двох дробів;
- Multiply виконує множення двох дробів;
- у методі Divide описаний механізм ділення двох дробів.

Створений набір методів, які реалізовують деякі унарні операції з дробами:

- метод Negative – змінює знак дробу на протилежний;
- оператори Inc та Dec виконують збільшення та зменшення значення дробу на одиницю відповідно;
- у методі Round реалізований механізм заокруглення значення дробової змінної до цілого числа.

Клас містить реалізації всіх необхідних операторів для виконання порівняння двох (та більше) дробових значень:

- Equal – перевірка двох дробових значень на рівність;
- NotEqual – визначення факту нерівності двох дробів;
- GreaterThan – виконує порівняння двох дробових значень, повертає істинний результат у разі, коли перший дріб більше другого;
- метод GreaterThanOrEqual повертає істинний результат у разі порівняння двох дробів, якщо перший дріб більший або рівний другому;
- аналогічно до двох попередніх методів оператори LessThan та LessThanOrEqual виконують порівняння двох дробів тільки для випадку, коли перший дріб менший та менший або рівний відповідно.

Група операторів *Impicit* реалізовує механізм приведення змінних різних типів до дробового типу та навпаки.

Також реалізовані функції знаходження абсолютного значення дробової змінної, для можливості перевантаження операцій описана операція абсолютного значення змінних дійсного типу.

Єдиним методом у приватній зоні оголошень класу є процедура *Reduct*, яка реалізовує функціонал скорочення дробу. Цей метод постійно використовується всіма арифметичними операторами для уникнення випадку переповнення значень дробів.

Модуль *Matrix* містить у собі деякі методи, призначені для роботи з матрицями в програмі, зокрема обчислення визначника та рангу матриці, знаходження оберненої та транспонованої матриці тощо. Ці методи використовуються, зокрема, у реалізації поліноміального алгоритму Кармаркара.

Останнім модулем блоку реалізації допоміжного функціоналу ми розглянемо модуль *Parser*, в кому реалізований допоміжний функціонал аналізу вхідних даних задач та підготовки даних для подальших обчислень.

В основному обчислювальному блоці містяться процедури та функції, призначені для безпосереднього розв'язування задач лінійної оптимізації. В ньому, зокрема, реалізовані такі методи:

- для розв'язування задачі лінійного програмування (ЗЛП) реалізовано симплекс-метод, двоїстий симплекс-метод та метод штучного базису, з автовибором відповідного методу залежно від розв'язуваної задачі;
- метод Гоморі та метод Дальтона-Ллевеліна для розв'язування цілочислових та дискретних ЗЛП;
- поліноміальний алгоритм Кармаркара для розв'язування ЗЛП;
- перший та другий методи комбінаторного відсікання для розв'язування комбінаторних задач оптимізації на переставному многограннику;
- метод відсікання графа переставного многогранника та метод аналізу графа переставного многогранника.

Розглянемо більш детально основні модулі, які містяться в обчислювальному блоці.

Модуль *Polynomial* містить у собі процедури і функції, які реалізують можливість розв'язування ЗЛП (у тому числі ДЗЛП у методах комбінаторного відсікання) поліноміальним алгоритмом Кармаркара. Алгоритм Кармаркара доступний для використання на будь-якому етапі розв'язування задач для отримання розв'язку ЗЛП. Зокрема, це такі методи, як:

- метод EvalKarmarkar – основна процедура, в якій безпосередньо реалізований алгоритм Кармаркара. Вхідними даними для методу є підготовлені на попередньому етапі дані задачі лінійного програмування;
- функція FindKarmarkarLim виконує побудову правильного відсікання в методах комбінаторного відсікання для можливості проведення подальшого розв'язку задачі.

У модулі *Simplex* містяться всі процедури та функції реалізації симплекс-методів:

- метод FirstST виконує підготовку початкових даних задачі для можливості її розв'язку симплекс-методом;
- функції SSM, DSM та M_Method проводять розв'язування ЗЛП симплекс-методом, двоїстим симплекс-методом та методом штучного базису відповідно;
- метод CaseMethod реалізовує вибір необхідного варіанту симплекс-методу для поточної ЗЛП;
- процедура Marks проводить обчислення оцінок у кожному з реалізованих модифікацій симплекс-методу.

В окремому модулі *Combinatorial* зосереджені всі процедури та функції реалізації методів комбінаторного відсікання. В модулі реалізовані як відомий перший метод комбінаторного відсікання, так і другий метод комбінаторного відсікання (з його модифікаціями), розроблений у рамках дисертаційного дослідження.

Зупинимось на основних методах, реалізованих у такому модулі:

- метод AddLimit_Comb використовується для визначення коефіцієнтів правильного комбінаторного відсікання в методах комбінаторного відсікання;
- функція Check_Comb призначена для перевірки критерію зупинки методів комбінаторного відсікання (як першого, так і другого);
- метод MakeLimits за даними переставлення будує набір лінійних обмежень, які являють собою переставний многогранник.

Модуль *Threading* містить у собі класи багатопроцесорних обчислень. Основні особливості цих класів такі:

- оптимізація алгоритмів розв'язування лінійних оптимізаційних задач методом відсікання та методом аналізу вершин графа переставного многогранника для багатопроцесорних систем;
- автоматичне масштабування кількості робочих обчислювальних потоків залежно від кількості доступних процесорів у системі;
- гнучкі механізми керування процесом розв'язування задачі для розподілу різних етапів розв'язування задачі між робочими потоками.

Багатопроцесорна реалізація виконана із застосуванням механізму паралельних обчислень за принципом «ведучий – ведений», відповідно до якого один основний потік виконує розподіл робіт між декількома робочими потоками в багатопроцесорній системі.

У програмній реалізації функціонал цих потоків реалізовано в класах *TPrimeGraphThrd* та *TWorkGraphThrd*, кожен з яких містить реалізації, зокрема, таких методів:

- метод ініціалізації потоку Init проводить підготовку як основного, так і робочого потоку до процесу обчислень. Вхідними параметрами методу є початкові дані задачі та напрямок обходу графа (напрямок оптимізації цільової функції);
- основний метод потоку Execute, в якому реалізований основний функціонал потоку;
- UpdateResults – метод, який використовується для коректного відображення поточних результатів, синхронізуючи їх з основним потоком програми;
- метод HandleTerminate призначений для коректного завершення роботи потоків.

Всі розглянуті модулі, функції та процедури, які в них розміщені, використовуються в процесі розв'язування практичних задач з використанням методів лінійної оптимізації за визначеними алгоритмами роботи програмної реалізації.

Висновки з цього дослідження та перспективи подальших розвідок у такому напрямі. У цій публікації наведено детальний опис створеної програмної реалізації розроблених методів [1–10] для розв'язування задач комбінаторної оптимізації. Програмна реалізація дозволила на значній кількості прикладів дослідити ефективність розроблених методів, провести серію числових експериментів.

У подальшому доцільно застосувати такий програмний комплекс для розв'язування практичних задач економічного характеру, що зводяться до задач комбінаторної оптимізації та розв'язуються зазначеними методами.

Список використаних джерел:

1. Ємець О. О., Ємець Є. М., Ольховський Д. М. Метод отсечения вершин графа перестановочного многогранника для решения линейных условных задач оптимизации на перестановках. *Кибернетика и системный анализ*. 2014. № 4. С. 146–153.

2. Ємець О. О., Ємець Є. М., Ольховський Д. М. Оптимізація лінійної функції на переставленнях: перетворення переставного многогранника до вигляду, необхідного для використання в алгоритмі Кармаркара. *Наукові вісті НТУУ «КПІ»*. 2010. № 2. С. 43–49.

3. Ємець О. О., Ольховська О. В., Ольховський Д. М. Програмний комплекс, що реалізує методи розв'язування задач комбінаторної оптимізації ігрового типу. *Вісник Черкаського університету*. № 18 (351). 2015. С. 96–101.

4. Ємець О. О., Ольховський Д. М. Програмна реалізація точних і наближених методів відсікання для розв'язування лінійних оптимізаційних задач на перестановках. *Вісник Запорізького національного університету :* збірник наукових статей. № 2. 2015. С. 73–76.

5. Ємець О. О., Ольховська О. В., Ольховський Д. М. Сравнение методов решения игровых задач: числовые эксперименты. *Искусственный интеллект*. № 1. 2014. С. 47–56.

6. Ємець О. О., Ольховська О. В., Ольховський Д. М. Теоретична оцінка складності алгоритмів розв'язування задач комбінаторної оптимізації ігрового типу. *Вісник Черкаського університету*. № 18 (351). 2015. С. 11–18.

7. Ємець О. О., Черненко О. О., Чілікіна Т. В., Ольховська О. В. Огляд задач комбінаторної оптимізації визначення рентабельності сільськогосподарського виробництва та методи їх розв'язування. *Математичне та комп'ютерне моделювання. Серія «Фізико-математичні науки»*. Випуск 22. 2021. С. 63–74.

8. Стоян Ю. Г., Ємець О. О. Теорія і методи евклідової комбінаторної оптимізації. Київ : Інститут системних досліджень освіти, 1993. 188 с.

9. Iemets O. O., Yemets E. M., Olhovskiy D. M. The Method of Cutting the Vertices of Permutation Polyhedron Graph to Solve Linear Conditional Optimization Problems on Permutations. *Cybernetics and Systems Analysis*. 2014. V. 50, I. 4. P. 613–619.

10. Koliechkina L., Pichugina O., Chilikina T. Multicriteria combinatorial optimization model of an infocommunication system. *International Conference "Problems of Infocommunications. Science and Technology*" (PIC S&T'2021). P. 135–138.

References:

1. Iemets, O. O., Yemets, E. M., Olhovskiy, D. M. (2014). Metod otsecheniya vershin grafa perestanovochnogo mnogogrannika dlya resheniya linejnyh uslovnyh zadach optimizacii na perestanovkah [The method of cutting off the vertices of the permutation polyhedron graph for solving linear conditional optimization problems on permutations]. *Kibernetika i sistemnyj analiz – Cybernetics and systems analysis.* 4, 146–153. [in Russian]

2. Iemets, O. O., Yemets, E. M., Olhovskiy, D. M. (2010). Optymizatsiia liniinoi funktsii na perestavlenniakh: peretvorennia perestavnoho mnohohrannyka do vyhliadu, neobkhidnoho dlia vykorystannia v alhorytmi Karmarkara [Optimization of a linear function on permutations: transformation of a permutable polyhedron to the form required for use in the Carmarkar algorithm]. *Naukovi visti NTUU "KPI" – Scientific news of NTUU "KPI".* 2, 43–49. [in Ukrainian]

3. Iemets, O. O., Olkhovska, O. V., Olhovskiy, D. M. (2015). Prohramnyi kompleks, shcho realizuie metody rozviazuvannia zadach kombinatornoi optymizatsii ihrovoho typu [A software package that implements methods for solving combinatorial game-type optimization problems]. *Visnyk Cherkaskoho universytetu – Bulletin of Cherkasy University*. 18, 96–101. [in Ukrainian]

4. Iemets, O. O. & Olhovskiy, D. M. (2015). Prohramna realizatsiia tochnykh i nablyzhenykh metodiv vidsikannia dlia rozviazuvannia liniinykh optymizatsiinykh zadach na perestanovkakh [Software implementation of accurate

and approximate clipping methods for solving linear optimization problems on permutations]. *Visnyk Zaporizhskoho natsionalnoho universytetu: Zbirnyk naukovykh statei – Bulletin of Zaporizhzhya National University: Collection of scientific articles.* 2, 73–76. [in Ukrainian]

5. Iemets, O. O., Olkhovska, O. V., Olhovskiy, D. M. (2014). Sravnenie metodov resheniya igrovyh zadach: chislovye jeksperimenty [Comparison of methods for solving game problems: numerical experiments]. *Iskusstvennyj intellect – Artificial intelligence*. 1, 47–56. [in Russian]

6. Iemets, O. O., Olkhovska, O. V., Olhovskiy, D. M. (2015). Teoretychna otsinka skladnosti alhorytmiv rozviazuvannia zadach kombinatornoi optymizatsii ihrovoho typu [Theoretical assessment of the complexity of algorithms for solving combinatorial optimization problems of the game type]. *Visnyk Cherkaskoho universytetu – Bulletin of Cherkasy University*. 18, 11–18. [in Ukrainian]

7. Iemets, O. O., Chernenko, O. O., Chilikina, T. V., Olkhovska, O. V. (2021). Ohliad zadach kombinatornoi optymizatsii vyznachennia rentabelnosti silskohospodarskoho vyrobnytstva ta metody yikh rozviazuvannia [Review of combinatorial optimization problems for determining the profitability of agricultural production and methods for solving them]. *Matematychne ta kompiuterne modeliuvannia. Seriia "Fizyko-matematychni nauky" – Mathematical and computer modeling. Series "Physical and Mathematical Sciences"*. 22, 63–74. [in Ukrainian]

8. Stoyan, Yu. G. & Yemets, O. O. (1993). *Teoriia i metody evklidovoi kombinatornoi optymizatsii [Theory and methods of Euclidean combinatorial optimization].* Kyiv : Institute for Systems Research in Education. [in Ukrainian]

9. Iemets, O. O., Yemets, E. M., Olhovskiy, D. M. (2014). The Method of Cutting the Vertices of Permutation Polyhedron Graph to Solve Linear Conditional Optimization Problems on Permutations. *Cybernetics and Systems Analysis*. 4, 613–619.

10. Koliechkina, L., Pichugina, O., Chilikina, T. (2021). Multicriteria combinatorial optimization model of an infocommunication system. *International Conference "Problems of Infocommunications. Science and Technology"* (PIC S&T'2021), 135–138.