УДК 519.6

ПРО РОЗВ'ЯЗУВАННЯ ЗАДАЧІ СКЛАДАННЯ РОЗКЛАДУ ЗАНЯТЬ, ВИКОРИСТОВУЮЧИ ГЕНЕТИЧНИЙ АЛГОРИТМ

О. Заневич, В. Кухарський

Львівський національний університет імені Івана Франка,
вул. Університетська, 1, Львів, 79000,
e-mail: oleh.zanevych@gmail.com, vitaliy.kukharskyy@lnu.edu.ua

Задача складання розкладу занять – актуальна проблема у закладах вищої освіти. Для її розв'язування існує низка класичних методів: методи динамічного програмування, цілочисельного програмування, нелінійного програмування, метод гілок і меж, методи імітаційного моделювання, метод розмальовки графа, задача про призначення та ін. Особливістю цих методів є математична строгість як формулювання задачі складання розкладу занять та алгоритмів її розв'язання. Вони мають заздалегідь визначені час збіжності і точність розв'язку, дають змогу проводити оцінки впливу різних чинників на час знаходження розв'язку. Недоліком усіх класичних методів є те, що вони використовують ітераційну процедуру пошуку або поліпшення деякого початкового наближення, де пошук результату відбувається в околі цього наближення. Це означає, що отриманий результат безпосередньо залежить від деякого початкового наближення і природно виникає проблема його вибору, яка призводить до необхідності множинного експерименту з різними значеннями початкового наближення, що суттєво збільшує час пошуку остаточного розв'язку. Також для класичних методів характерна складність одержуваної математичної моделі та різке (експоненціальне) зростання витрат часу на пошук прийнятного розв'язку з ростом обсягів вихідної інформації. Для уникнення вищенаведених недоліків класичних методів задачу складання розкладу занять можна розв'язати, застосовуючи генетичний алгоритм. Запропоновано один варіант заданння цільової функції для оптимізації розкладу та визначено на її підставі функцію придатності. Описано реалізацію класичних генетичних операторів: кросинговера, мутації та вибірки для популяції розкладів, а також запропоновано оператор поправки, який допомагає поліпшити варіанти розкладу, отримані внаслідок викликів класичних генетичних операторів. Для реалізації операторів кросинговеру та поправки введено поняття локальної цільової функції. Наведено загальну схему генетичного алгоритму для розв'язування задачі складання розкладу та результати числового експеременту.

Ключові слова: задача складання розкладу, генетичний алгоритм, функція придатності, локальна цільова функція, еволюційні алгоритми, вибірка, кросинговер, мутація.

1. Вступ

Однією з задач, які виникають під час планування й організації навчального процесу у будь-якому навчальному закладі, є складання розкладу занять. Від якості складання розкладу значною мірою залежить ефективність навчання студентів, робота викладачів, оптимальність використання аудиторного фонду, затребуваних для навчання ресурсів, витрати на підтримку навчального процесу.

Перші роботи в галузі автоматизації складання розкладу з'явилися в 50-60-ті роки XX ст. у зв'язку з впровадженням автоматизованих систем управління виробництвом. Розв'язування задач, які пов'язані зі складанням розкладу виконання робіт у виробничих системах, привело до створення теорії розв'язання задач складання розкладів (ЗСР) [8, 10–12]. Ця теорія дає універсальні розв'язки різноманітних виробничих завдань, які пов'язані з календарним плануванням, упорядкуванням робіт у часі і просторі тощо, з урахуванням наявних обмежень на доступні ресурси. Розв'язування ЗСР в рамках цієї теорії у підсумку зводиться до використання математичного апарату цілочисельного програмування [2, 22, 23].

На рубежі XX і XXI ст. актуальним стало створення систем автоматизованого управління навчальним процесом в освітніх системах навчання. Це пов'язано з посиленням вимог до якості навчання, появою різних форм навчання, розвитком форм дистанційного навчання, необхідністю підвищення економічної ефективності навчання і т. п.

2. Класичні методи розв'язування ЗСР

Перші роботи в галузі автоматизації розв'язування ЗСР для освітніх систем ґрунтувалися на адаптації або модифікації створеної в 50-60 роках XX ст. класичної теорії розв'язання ЗСР. Такий підхід передбачає формулювання ЗСР занять у термінах теорії розкладів (виділення приладів і вимог) з подальшим розв'язуванням її одним з відомих методів цілочисельного програмування. Найзагальніше формулювання ЗСР у термінах теорії розкладу таке. За допомогою деякої кількості ресурсів або приладів має виконуватися деяка фіксована система вимог (завдань). Мета полягає в тому, щоб при заданих властивостях вимог і ресурсів та накладених на них обмеженнях знайти ефективний алгоритм упорядкування завдань, що повністю або частково оптимізує необхідну міру ефективності. У відомих працях, що реалізують такий підхід [5, 29], при формулюванні ЗСР навчальних занять як прилади виступають аудиторії, а вимог – навчальні групи.

Варто зазначити, що подібне формулювання та розв'язування ЗСР занять за допомогою апарата класичної теорії розкладів пов'язане з низкою складнощів і потребує модифікації (в тім числі і термінологічної) традиційної постановки ЗСР. Це виражається в необхідності врахування специфічних особливостей організації процесу навчання в освітніх системах. У [19, 21] пропонується для розв'язування ЗСР навчальних занять використовувати теоретичний апарат складання "виробничих" розкладів. Це призводить до необхідності введення нових, часто штучних понять, таких як "багатофункціональні прилади" і "комплексні вимоги", у цьому випадку прилади розглядаються аудиторії, а вимоги – заняття (на відміну від традиційного – навчальні групи).

Необхідність такої модифікації обумовлена специфікою освітніх систем: наприклад, існують такі види занять, які проводяться одночасно для декількох навчальних груп (потокові лекції), а в рамках теорії розкладів один прилад у деякий момент часу може обслуговувати не більше однієї потреби.

Область ефективного застосування відомих методів складання "виробничих" розкладів для складання розкладу навчальних занять обмежена невеликими освітніми системами з малою кількістю обмежень, що накладаються на розклад. Все це призвело до появи нового напряму розв'язування ЗСР навчальних занять, заснованого на безпосередньому використанні методів цілочисельного програмування без залучення методів традиційної теорії складання "виробничих" розкладів.

Усі відомі праці в цьому напрямі присвячені автоматизації процедури складання розкладу занять [4, 8, 10, 12, 16, 25–28] описують передусім класичні методи її розв'язування: методи динамічного програмування, цілочисельного програмування, нелінійного програмування, метод гілок і меж [3], методи імітаційного моделювання, метод розмальовки графа [2, 7], задача про призначення [3] та ін.

Відмінною рисою класичних методів є математична строгість як формулювання задачі складання розкладу занять та алгоритмів її розв'язання (використовують "жорсткі" алгоритми). Ці методи дають підстави:

а) розробляти ітеративні методи розв'язання ЗСР навчальних занять, які мають прийнятний час збіжності і точність розв'язку;

б) оцінювати вплив на час і точність виконання ЗСР навчальних занять цікавих нам чинників.

Серед методів, які використовують для розв'язування ЗСР занять, можна також виділити методи, що засновані на використанні математичного апарата теорії графів. Використовуючи такі методи, ЗСР занять зводиться до задачі розмальовки графа.

У цьому випадку будується граф, в якому кожна вершина представляє заплановане навчальним планом заняття. Якщо між якимись двома вершинами можливі конфлікти (з'являються накладки), наприклад, два заняття проводяться в одній аудиторії, то вони з'єднуються ребром. Це еквівалентно забороні одночасного проведення цих занять. Тоді ЗСР можна сформулювати як задачу мінімізації кількості кольорів, потрібних для розмальовки графа, де кожен колір відповідає одному часовому періоду в розкладі (день і номер пари) [7].

Застосування такого методу для рішення реальних ЗСР для освітніх систем масового навчання малоефективне, але поєднання цього підходу з іншими методами може виявитися корисним [22, 23].

Подібною до ЗСР є задача про призначення, яка є окремим випадком транспортної задачі. Задача про призначення трапляється у призначенні людей на посади або роботи, автомашин на маршрути, водіїв на машини, у разі розподілу груп по аудиторіях і т. п.

Задача про призначення – це задача, в якій для виконання кожної роботи потрібно один і тільки один ресурс (одна людина, одна автомашина і т. п.), а кожен ресурс може використовуватися на одній і тільки одній роботі. Тобто ресурси не подільні між роботами, а роботи не діляться між ресурсами [28].

Специфічна структура задачі про призначення дала змогу розробити для її розв'язування так званий "Угорський метод". Цей метод заснований на симплекс-методі [4].

3. Розв'язування ЗСР, використовуючи генетичний алгоритм

Еволюційні алгоритми (ЕА) вважаються зручними для оптимізації загального призначення через високу гнучкість, що супроводжується концептуальною простотою. Вони виявились досить ефективним для розв'язування задачі складання розкладу ([6], [7], [8]), тому їх можна брати як основу для побудови універсальних розкладів занять.

Загальним недоліком всіх класичних методів є те, що вони в своїй основі використовують ітераційну процедуру пошуку або поліпшення деякого початкового наближення (опорного плану розкладу), де пошук результату відбувається в околі цього наближення. Це означає, що отриманий результат безпосередньо залежить від деякого початкового наближення і природно виникає проблема його вибору.

Крім того, організація пошуку екстремуму цільової функції в класичних методах відбувається:

по-перше, на підставі вивчення її властивостей у малому околі початкового наближення,

по-друге – тільки в одному напрямі, який визначається напрямом бажаної зміни цільової функції в малому околі її спостереження.

Все це призводить до необхідності множинного експерименту з різними значеннями початкового наближення, що суттєво збільшує час пошуку остаточного розв'язку.

Також недоліком класичних методів є громіздкість і складність одержуваної математичної моделі та різке (експоненціальне) зростання витрат часу на пошук прийнятного розв'язку з ростом обсягів вихідної інформації.

Для уникнення вищенаведених недоліків, ЗСР занять можна розв'язати шляхом застосування генетичного алгоритму (ГА). У [31] наведено реалізацію ГА для складання розкладу відвідуваності пацієнтів у лікарнях. Ефективну реалізацію ГА для побудови шкільного розкладу описано у [14]. Класичний ГА для розв'язування ЗСР у закладах вищої освіти (ЗВО) наведено в [24, 32]. У [33] пропонується алгоритм табу-пошуку для подальшої оптимізації попередньо отриманого розкладу за допомогою ГА. Мета нашої праці – реалізація варіанту ГА для складання розкладу у ЗВО з оператором поправки, який викликатиметься на кожній стадії роботи алгоритму після застосування класичних генетичних операторів (кросинговеру, мутації та вибірки), подібно як у [14].

ГА є специфічним представником обчислювальних парадигм сім'ї штучного інтелекту. У ГА кожен можливий розв'язок поданий "цифровою особою", і після створення вихідного набору реальних рішень (популяція), окремі особи випадково поєднуються, що дає змогу рекомбінувати генетичний матеріал. Отримані особи (в нашому випадку це можливі варіанти розкладу) можуть бути мутовані з певною ймовірністю мутацій. Отже, отримана нова популяція цифрових осіб "переживає" процес природного відбору, який сприяє виживанню найпридатніших представників (найліпші варіанти розкладу), і є основою для нового еволюційного циклу. Придатність окремих варіантів розкладу виявляється за допомогою функції, яка називається функцією придатності ($f.f.$, від *fitness function*) і тісно пов'язана з цільовою функцією ($o.f.$, від *objective function*) для оптимізації. $f.f.$ кількісно визначає, наскільки хорошим є розклад. У ГА індивіди іноді називають хромосомами, а положення в хромосомі називаються генами. Значення, яке приймає ген, називається аллелем (або аллельним значенням). Для задачі складання розкладу у ЗВО ми пропонуємо використовувати хромосоми із 5-х генів, які відповідно визначають навчальне заняття з посиланням на конкретне викладацьке навантаження, день, години проведення (або номер пари), місце проведення та періодичність проведення (що тижня, по чисельнику чи по знаменнику).

ГА породжує нову популяцію з попередньої, використовуючи деякі генетичні оператори. Нова популяція створюється за допомогою оператора вибірки, що дає ліпшу вірогідність відтворення для осіб з більшим значенням $f.f.$ Далі вона поповнюється новими особинами, які породжуються викликами операторів кросинговеру та мутацій. Загальний ефект генерації розкладу ГА полягає в тому, що ми у кожній новій популяції отримуємо нові згенеровані випадково варіанти розкладу з кращими значенням $f.f.$

Вибір ГА для розв'язування задачі складання розкладу ми обґрунтовуємо тим, що він допомагає проводити ефективний пошук оптимальних розв'язків у просторах з дуже великою розмірністю. Це підтверджено тим, що ГА успішно був застосований до різних видів оптимізаційних задач, враховуючи NP-повні проблеми [20], такі як задача комівояжера [30] та задача здійсненності бульових формул (SAT) [17].

Вхідні дані для задачі складання розкладу у ЗВО формуються на підставі робочих навчальних планів, викладацьких навантажень, аудиторного фонду (лекційні та звичайні аудиторії, лабораторії, спортзали, спортивні майданчики, басейн тощо), побажань викладачів і студентів стосовно часу проведень занять. На підставі цих даних ми формуємо вимоги для проведення навчальних занять. Кожна вимога охоплює предмет і тип навчального заняття (лекція, практична, лабораторна), викладача, який проводить заняття, групи студентів та бажані години проведення.

Опишемо детальніше процес генерації розкладу занять за допомогою ГА. Для цього визначимо функцію придатності $f.f.$, класичні генетичні оператори (вибірка, кросинговер, мутація) та оператор поправки для популяції розкладів.

Функцію придатності $f.f.$ можна визначити багатьма способами. Один із простих варіантів такий. Пронумеруємо викладачів від 1 до L, групи студентів від 1 до G, місця проведення занять від 1 до A. Для довільного варіанта розкладу s виділяємо декілька критеріїв його несумісності. Наприклад, нехай $\Pi_1(s,i)$ – кількість часових накладок у викладача з індексом i, $\Pi_2(s,j)$ – кількість часових накладок у групи студентів з індексом j, $\Pi_3(s,t)$ – кількість накладок у місці проведення з індексом t, $\Pi_4(s,k)$ – кількість вікон у викладача з індексом k, $\Pi_5(s,p)$ – кількість вікон у групи студентів з індексом p, $\Pi_6(s,h)$ – кількість пар на тиждень, які викладач з індексом h проводить у небажаний час, $\Pi_7(s,z)$ – кількість пар на тиждень, які має група з індексом z у небажаний час.

Визначимо цільову функцію

$$o.f.(s) = \beta_1 \sum_{i=1}^{L} \Pi_1(s,i)^{\alpha_1} + \beta_2 \sum_{j=1}^{G} \Pi_2(s,j)^{\alpha_2} + \beta_3 \sum_{t=1}^{A} \Pi_3(s,t)^{\alpha_3} +$$
$$+ \beta_4 \sum_{k=1}^{L} \Pi_4(s,k)^{\alpha_4} + \beta_5 \sum_{p=1}^{G} \Pi_5(s,p)^{\alpha_5} + \beta_6 \sum_{h=1}^{L} \Pi_6(s,h)^{\alpha_6} + \beta_7 \sum_{z=1}^{G} \Pi_7(s,z)^{\alpha_7},$$

де $\beta_1, \beta_2, \beta_3 \gg \beta_4, \beta_5, \beta_6, \beta_7 > 0$ і $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7 > 1$ – деякі задані додатні константи, які дають змогу визначати переваги одних критеріїв над іншими. Тепер задачу розкладу можна трактувати як задачу знаходження мінімуму $o.f.$ Однак використання методу Монте-Карло в операторі вибірки потребує визначення функції придатності $f.f.$, яка набуває більші значення для ліпших варіантів розкладу. Її можна визначити через $o.f.$, наприклад, так:

$$f.f.(s) = \frac{M}{m + o.f.(s)},$$

де $M \gg m > 0$ – деякі задані додатні константи.

Для довільного варіанта розкладу s та індексу заняття i визначимо локальну цільову функцію

$$l.o.f.(s,i) = \beta_1 \Pi_1(s,l)^{\alpha_1} + \beta_2 \sum_{j \in g} \Pi_2(s,j)^{\alpha_2} + \beta_3 \Pi_3(s,d)^{\alpha_3} + \beta_4 \Pi_4(s,l)^{\alpha_4} +$$

$$+ \beta_5 \sum_{p \in g} \Pi_5(s,p)^{\alpha_5} + \beta_6 \Pi_6(s,l)^{\alpha_6} + \beta_7 \sum_{z \in g} \Pi_7(s,z)^{\alpha_7},$$

де l – індекс викладача, який проводить це заняття; d – індекс аудиторії, в якій проводиться заняття і g – список груп студентів, для яких проводиться заняття.

Далі опишемо генетичні оператори, які використаємо для генерації нових варіантів розкладу.

Кросинговер – оператор, який застосовують для будь-якої пари з двох різних варіантів розкладу. Для них він генерує двох синів, кожен з яких має частину хромосом від одного батька, а частину від іншого. Ми реалізовуємо цей оператор так. Випадково вибираємо s_1 і $s_2 \in P$, де P – популяція розкладів. Нехай V – загальна кількість занять (генів) у розкладі. Генеруємо випадкове число r таке, що $1 < r < V$. Створюємо два нові варіанти розкладу s'_1 і s'_2 і додаємо їх у поточну популяцію шляхом обміну місцями r занять з найбільшим значенням $l.o.f$ розкладу s_1 з відповідними значення з s_2.

Мутація – оператор, який викликає зміну аллельного значення одного з генів, для яких дозволена мутація. Тобто ми випадково вибираємо $s \in P$ і на його підставі створюємо новий варіант розкладу шляхом випадкової зміни дня, номера пари, заміною чисельника на знаменник чи навпаки, а також місця проведення.

Поправка – оператор, який оптимізовує варіанти розкладу так. $\forall s \in P$ і $\forall i : 1 \leq i \leq V$ намагаємось виконати таку зміну алейних значень i-го заняття розкладу s, щоб породити новий розклад s' так, щоб виконувалось

$$l.o.f(s,i) > l.o.f(s',i),$$

якщо таке можливо, то ми оновлюємо ці характеристики заняття. Також намагаємось знайти такий індекс $j : 1 \leq j \leq V$, $j \neq i$, що обмін місцями одного чи декількох алейних значень i-го та j-го заняття породжував новий розклад s', так щоб виконувалось

$$l.o.f(s,i) + l.o.f(s,j) > l.o.f(s',i) + l.o.f(s',j),$$

якщо такі індекси існують, то ми міняємо місцями відповідні характеристики занять.

Вибірка – оператор відтворення, який вибирає з попередньої популяції варіанти розкладів з більшими значеннями $f.f.$ Ми завжди копіюємо у нову популяцію задану сталу кількість із B варіантів розкладу з найменшим значеннями $o.f.$ Усім іншим варіантам із попередньої популяції ми присвоюємо вірогідність відтворення, яка дорівнює значенню $f.f.$ конкретного розкладу, поділеному суму значень $f.f.$ усіх варіантів розкладу, і, застосовуючи метод Монте-Карло, визначаємо, які з них увійдуть у нову популяцію.

Загалом складання розкладу з V занять з використанням ГА виглядає так.

Крок 0. Генеруємо випадково чином деяку кількість довільних варіантів розкладу (початкову популяцію) і переходимо на крок 3.

Крок 1. Генеруємо нові варіанти розкладу, застосовуючи до існуючих оператори кросинговеру $C * V$ разів, де $C \in (0,1)$ – деяка задана константа.

Крок 2. Генеруємо нові варіанти розкладу, застосовуючи до існуючих оператори мутації $U * V$ разів, де $U \in (0,1)$ – деяка задана константа.

Крок 3. Застосовуємо до усіх нових варіантів розкладу оператор поправки.

Крок 4. Обчислюємо значення $o.f.$ та $f.f.$ для нових варіантів розкладу.

Крок 5. Отримуємо нову популяцію розкладів, застосовуючи до попередньої оператора вибірки.

Крок 6. Якщо протягом O останніх ітерації не вдається згенерувати розклад з меншим значенням $o.f.$ за вже існуюче, то ми зупиняємо роботу алгоритму і за шуканий розклад приймаємо один із варіантів, для якого $o.f.$ має найменше значення. Інакше переходимо на крок 1.

Нижче зображено загальну схему роботи алгоритму (рис. 1).

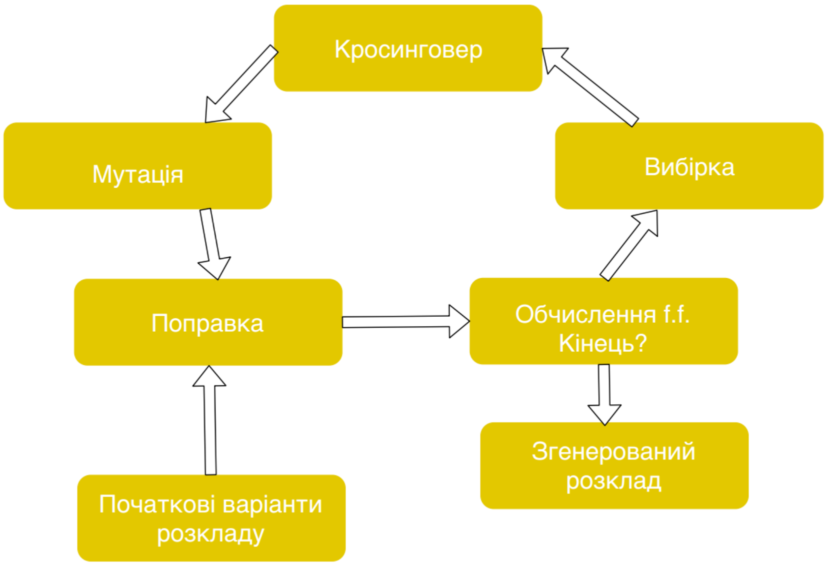

Рис. 1. Схема роботи алгоритму

Зауважимо, що ми генеруємо початкові варіанти розкладу на кроці 0 із правильною кількістю годин проведення кожного заняття і реалізовуємо оператори кросинговеру, мутації та поправки так, щоб вони не змінювали її. Тобто у нас не можуть виникнути ситуації, коли у розкладі немає якогось заняття, або є меншу чи більшу кількість разів ніж треба.

4. Чисельний експеримент

Для апробації алгоритму було використано випадково згенеровані набори вхідних даних для складання розкладу з V занять, які проводять L викладачів у G групах та у A приміщеннях. Було задано такі значення штрафних параметрів функції мети $o.f.$: $\beta_1 = \beta_2 = \beta_3 = 100$, $\beta_4 = \beta_5 = 5$, $\beta_6 = \beta_7 = 1$, $\alpha_1 = \alpha_2 = \alpha_3 = 4$, $\alpha_4 = \alpha_5 = 3$, $\alpha_6 = \alpha_7 = 2$, констант, що визначають функцію придатності $o.f.$: $M = 10000$, $m = 1$, та параметрів алгоритму $B = 5$, $C = 0.2$, $U = 0.1$, $O = 100$.

Нижче наведено кількість зроблених ітерацій ГА для наборів вхідних даних різної розмірності, а також мінімальне значення цільової функції (min $o.f.$) та відповідне йому максимальне значення функції придатності (max $f.f$), які були досягнуті на варіанті розкладу, який був повернутий алгоритмом (див. табл. 1).

Для простеження тенденції збіжності описаного ГА для розв'язування ЗСР наве-

Таблиця 1

Кількість ітерацій ГА для наборів вхідних даних різної розмірності

Номер набору даних	V	L	G	A	Кількість ітерацій	min $o.f.$	max $f.f$
1	200	15	20	25	7124	25	384.615
2	400	30	40	50	9973	73	135.135
3	800	60	80	100	13488	167	59.524
4	1600	120	160	200	21542	258	38.610

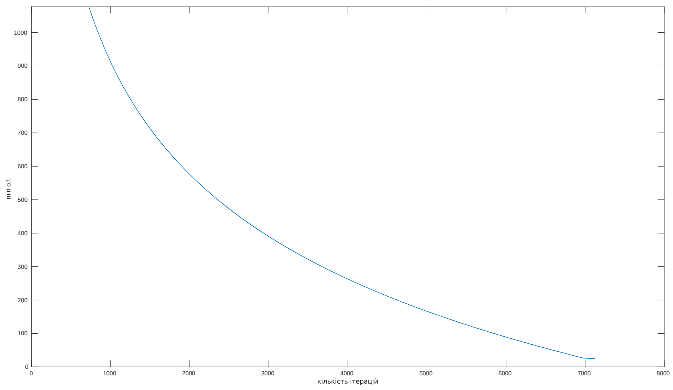

Рис. 2. Графік зміни min $o.f.$ для набору даних № 1

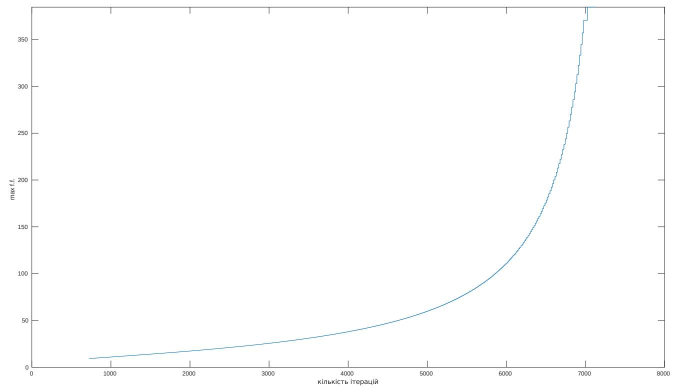

Рис. 3. Графік зміни min $f.f.$ для набору даних № 1

дено, відповідно, графіки зміни найменшого значення цільової функції (min $o.f.$) та відповідного йому максимального значення функції придатності (max $f.f$) в процесі виконання алгоритму для кожного набору даних (див. рис. 2–9).

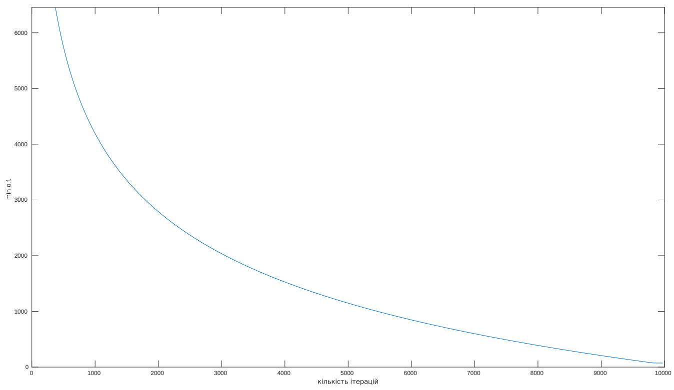

Рис. 4. Графік зміни min $o.f.$ для набору даних № 2

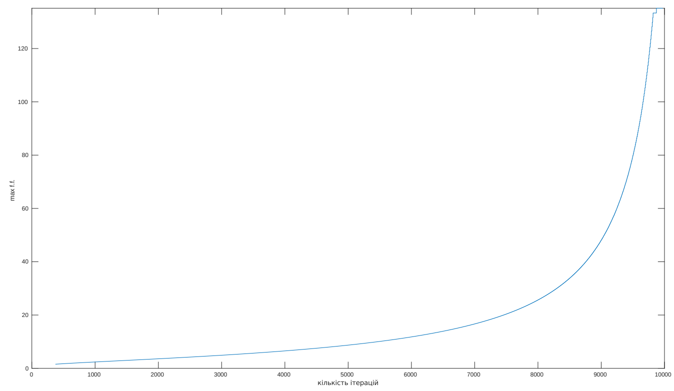

Рис. 5. Графік зміни min $f.f.$ для набору даних № 2

5. Висновки

Як видно з результатів проведеного чисельного експерименту, зі збільшенням розмірності вхідних даних удвічі, кількість зроблених ітерацій зростала, відповідно, приблизно в 1,4, 1,35 та 1,6 раза, що є доволі добрим результатом. Хоча його можна піддавати сумніву, оскільки вхідні дані не були реальними, а лише згенеровані випадково.

З отриманих графіків зміни min $o.f.$ можна зробити висновок про те, що

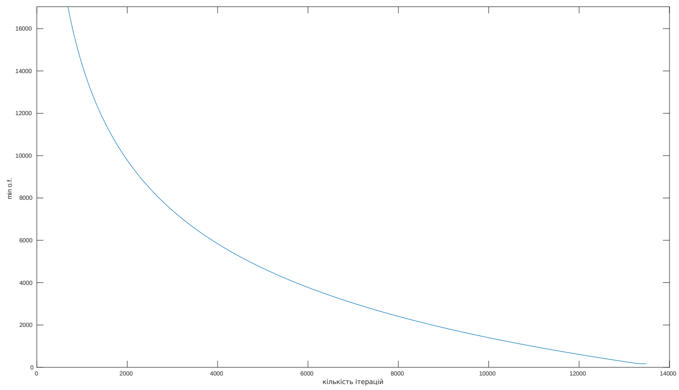

Рис. 6. Графік зміни min $o.f.$ для набору даних № 3

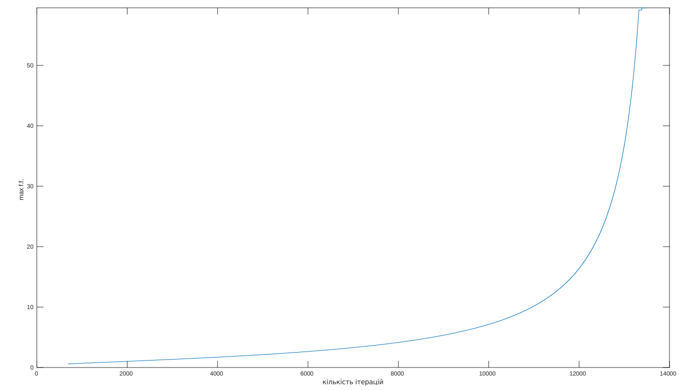

Рис. 7. Графік зміни min $f.f.$ для набору даних № 3

поліпшення розкладу відбувається краще на початкових ітераціях і є можливою втрата ефективності алгоритму на останніх стадіях виконання. Як бачимо, в усіх чотирьох випадках не вдалось досягнути "ідеального" розкладу з нульовим значенням $o.f.$ (хоча тестові вхідні дані згенеровані так, що це є можливим). Отож, виникає думка про те, що ГА доцільно застосовувати для того, щоб отримати якийсь можливо не зовсім оптимальний упорний варіант розкладу, який потім ще можна буде оптимізовувати класичними методами. Перевірці цієї гіпотези буде присвячена наша подальша робота над розв'язуванням ЗСР у ЗВО.

Також перевагою описаного ГА для розв'язування ЗСР є можливість генерувати бажаний розклад за будь-якими критеріями. Для цього нам потрібно лише врахувати їх при визначенні $o.f.$. Надзвичайно корисним є те, що структура алгоритму допомагає нам легко розпаралелити його виконання у декількох потоках, що дає

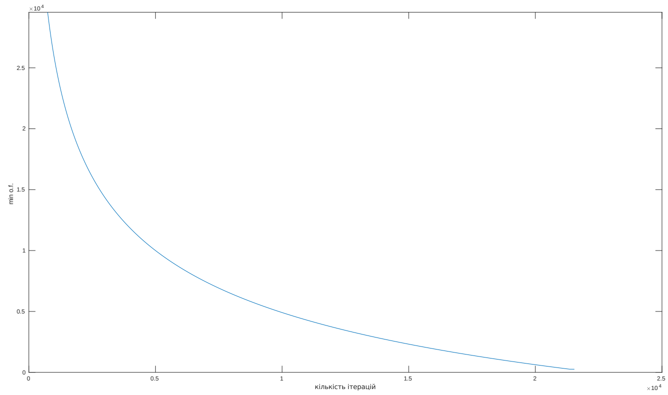

Рис. 8. Графік зміни min $o.f.$ для набору даних № 4

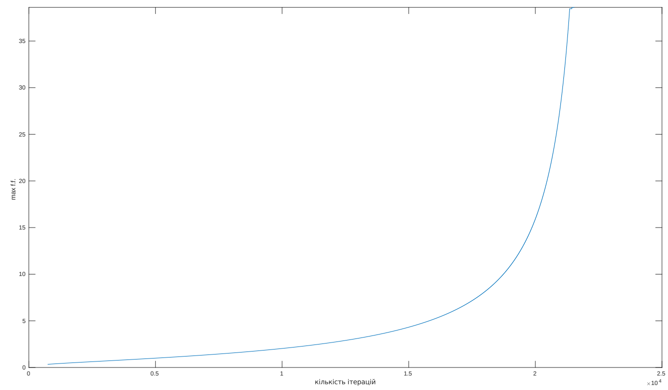

Рис. 9. Графік зміни min $f.f.$ для набору даних № 4

змогу ліпше використовувати обчислювальні ресурси.

Недоліками ГА є значні затрати пам'яті, оскільки ми одночасно обробляємо доволі велику кількість різних варіантів розкладу.

Список використаної літератури

1. *Бабкин Э. А.* Проектирование и реализация алгоритмов составления учебного расписания на основе многоагентных технологий / Э. А. Бабкин, И. М. Ретинский // Материалы научно-технической конференции. Технические, программные и математические аспекты управления сложными распределенными системами. – Нижний Новгород. – 2003. – С. 10–12.
2. *Безгинов А. Н.* Обзор существующих методов составления расписания / А. Н. Безгинов, С. Ю. Трегубов // Информационные технологии и программирование: межвузовский

сборник статей. – Выпуск 2 (14). – Москва: МГИУ, 2005.

3. *Вагнер Г.* Основы исследования операций: в 3 т. Т. 2 / Г. Вагнер, пер. с англ. В. Я. Алтаева. – Москва: Мир, 1973. – 488 с.
4. *Вагнер Г.* Основы исследования операций: в 3 т. Т. 3 / Г. Вагнер, пер. с англ. Б. Т. Вавилова. – Москва: Мир, 1972. – 501 с.
5. *Каширина И. Л.* Генетический алгоритм решения квадратичной задачи о назначениях специального вида / И. Л. Каширина // Вестник ВГУ. – 2003. – № 1.
6. *Клеванский Н. Н.* Моделирование стратегии формирования расписания занятий ВУЗа средствами реляционной алгебры / Н. Н. Клеванский, Е. А. Макарцова, С. А. Костин // Прикладные проблемы образовательной деятельности: Межвуз. сб. научн. тр. Воронеж. – ВГПУ. – 2003. – С. 71–74.
7. *Клеванский Н. Н.* Формирование расписания с использованием динамических критериев загруженности / Н. Н. Клеванский, Е. А. Макарцова // XI Международная конференция-выставка "Информационные технологии в образовании". Часть IV. – Москва: МИФИ, 2001. – С. 139–140.
8. *Конвей Р. В.* Теория расписаний / Р. В. Конвей. – Москва: Наука, 1975. – 395 с.
9. *Маслов М. Г.* Эвристический алгоритм решения задачи составления расписания учебных занятий в вузе / М. Г. Маслов // Математические методы в технике и технологиях: Сб. трудов XV Международной научной конференции. В 10-и т. – 2–4 июня 2002 г. – Тамбов. – 2002. – Т. 9. – С. 86–88.
10. *Строкина Ю. Г.* Алгоритмические процедуры формирования гетерогенных расписаний для производственных систем: дис. на соискание ученой степени канд. техн. наук / Ю. Г. Строкина. – Уфа: УГАТУ, 1997. – 150 с.
11. *Танаев В. С.* Теория расписаний / В. С. Танаев. – Москва: Наука, 1989. – 256 с.
12. *Танаев В. С.* Введение в теорию расписаний / В. С. Танаев, В. В. Шкуба. – Москва: Наука, 1975. – 256 с.
13. *Coello C. A.* Evolutionary Algorithms for Solving Multi-Objective Problems. Second Edition / C. A. Coello, D. A. Van Veldhuizen, G. B. Lamont. – Springer, 2007.
14. *Colorni A.* Genetic Algorithm To Solve The Timetable Problem / A. Colorni, M. Dorigo, V. Maniezzo // Research Gate. – 1994.
15. *Cowling P.* An Investigation of a Hyperheuristic Genetic Algorithm Applied to a Trainer Scheduling Problem / P. Cowling, G. Kendall, L. Han // Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002). – 2002. – P. 1185–1190.
16. *Cowling P.* Hyperheuristics: A Robust Optimisation Method Applied to Nurse Scheduling / P. Cowling, G. Kendall, E. Soubeiga // Proceedings of the VII Parallel Problem Solving From Nature (PPSN VII), Lecture Notes in Computer Science. – Springer. – 2002. – Vol. 2439. – P. 7–11.
17. *De Jong K. A.* Using genetic algorithms to solve NP-complete problems / K. A. De Jong , W. M. Spears // Proceedings of the Third International Conference on Genetic Algorithms. – Morgan Kaufmann. – 1989.
18. *Deb K.* Towards a Quick Computation of Well-Spread Pareto Optimal Solutions / K. Deb, M. Manikanth, S. Mishra // Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Faro Portugal, Lecture Notes in Computer Science. – Springer. – 2003. – Vol. 2632.– P. 222–236.
19. *Farina M.* Fuzzy Optimality and Evolutionary Multi-Objective Optimization / M. Farina, P. Amato // Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Faro Portugal, Lecture Notes in Computer Science Springer. – 2003. – Vol. 2632. – P. 58–72.
20. *Garey M. R.* Computers and intractability / M. R. Garey, D. S. Johnson W. H. Freeman & Company, 1979.

21. *Gaw A.* Distributed Choice Function Hyper-Heuristics for Timetabling and Scheduling / A. Gaw, P. Rattadilok, R. S. Kwan // Proceedings of the 2004 International Conference on the Practice and Theory of Automated Timetabling (PATAT 2004). – Pittsburgh. – 2004. – P. 495–497.
22. *Gunawan S.* Multilevel Multiobjective Genetic Algorithm Using Entropy to Preserve Diversity / S. Gunawan, A. Farhang, S. Azarm // Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Faro Portugal, Lecture Notes in Computer Science – Springer. – 2003. – Vol. 2632. – P. 148–161.
23. *Han L.* Investigation of a Tabu Assisted Hyper-Heuristic Genetic Algorithm / L. Han, G. Kendall // Proceedings of the 2003 Congress on Evolutionary Computation (CEC2003). – Canberra. – 2003. – P. 2230–2237.
24. *Jain T.* Genetic Algorithm as a General Approach to Time Tabling Problem / T. Jain, N. Jamil // Eur. J. Bus. Manag. – 2015. – Vol. 7, No. 4. – P. 7–11.
25. *Horn J.* Niche Distributions on the Pareto Optimal Front / J. Horn // Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Faro Portugal, Lecture Notes in Computer Science. – Springer. – 2003. – Vol. 2632. – P. 365–375.
26. *Jin H.* Adaptive Diversity Maintenance and Convergence Guarantee in Multiobjective Evolutionary Algorithms / H. Jin, M. L. Wong // Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003). – Camberra. – 2003. – P. 2498–2505.
27. *Kumar R.* Improved Sampling of the Pareto-front in Multiobjective Genetic Optimization by Steady-state Evolution: A Pareto Converging Genetic Algorithm / R. Kumar, P. Rockett // Evolutionary Computation. – 2002. – Vol. 10, No. 3. – P. 283–314.
28. *Laumams M.* Combining Convergence and Diversity in Evolutionary Multiobjective Optimization / M. Laumams, L. Thiele, K. Deb, E. Zitzler // Evolutionary Computation. – 2002. – Vol. 10, No. 3. P. 263–282.
29. *Mostaghim S.* The Role of e-dominance in Multi-objective Particle Swarm Optimization Methods / S.Mostaghim, J. Teich // Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003). – Camberra. – 2003. – P. 1764–1771.
30. *Muehlenbein H.* Parallel genetic algorithms, population genetics and combinatorial optimization, / H. Muehlenbein // Proceedings of the Third International Conference on Genetic Algorithms. – Morgan Kaufmann, 1989.
31. *Peng Y.* A hybrid simulation and genetic algorithm approach to determine the optimal scheduling templates for open access clinics admitting walk-in patients / Y. Peng, X. Qu, J. Sh // Computers & Industrial Engineering. – 2014 – Vol. 72. – P. 282–296.
32. *Premasiril D. M.* University Timetable Scheduling Using Genetic Algorithm Approach Case Study: Rajarata University of Sri Lanka / D. M. Premasiril // Journal of Engineering Research and Application ISSN: 2248-9622. – 2018. – Vol. 8, Issue 12. – Part-II. – P. 30–35.
33. *Premlata L. R.* Solving the class timetable problem by using genetic algorithm and tabu search algorithm / L. R. Premlata, A. Sonawane // IRD India. – 2014. – Vol. 3, No. 5. – P. 45–50.

Стаття: надійшла до редколегії 11.11.2019
доопрацьована 20.11.2019
прийнята до друку 20.11.2019

ABOUT SOLVING THE TIMETABLE PROBLEM USING A GENETIC ALGORITHM

O. Zanevych, V. Kukharskyy

Ivan Franko National University of Lviv,
Universytetska Str., 1, Lviv, 79000,
e-mail: oleh.zanevych@gmail.com, vitaliy.kukharskyy@lnu.edu.ua

The timetable problem of classes is an urgent problem in higher education institutions. There are a number of classical methods for solving it: dynamic programming methods, integer programming, nonlinear programming, the branch and bound method, simulation methods, the graph coloring method, assignment problem, and others. The peculiarity of these methods is the mathematical rigor of both the formulation of the timetable problem of classes and algorithms for its solution. They have predetermined convergence time and accuracy of the solution and allow to estimate the influx of different factors for time of finding the solution. The disadvantage of all classic methods is that they basically use an iterative search procedure or refinement of some initial approximation, whereby the result is searched around that approximation. This means that the result directly depends on some initial approximation and naturally there is a problem of its choice, which leads to the need for a multiple experiment with different values of the initial approximation, which significantly increases the time to find the final solution. Also, the classical methods are characterized by the complexity of the mathematical model obtained and the sharp (exponential) increase in time spent finding an acceptable solution as the volume of source information increases. To avoid the above disadvantages of the classical methods, the timetable problem of classes can be solved by applying a genetic algorithm. The paper proposes one variant of setting a objective function for timetable optimization and defining a fitness function based on it. The article describes the implementation of classical genetic operators: crossover, mutations, and selection for a population of timetables, and also proposes a correction operator that improves the variants of timetables obtained by calls of classical genetic operators. For the implementation of crossover and correction operators, the concept of a local target function is introduced. The general scheme of the genetic algorithm for solving the timetable problem is presented in the paper, and the results of the numerical experiment are given.

Key words: Timetable problem, genetic algorithm, fitness function, local objective function, evolutionary algorithms, selection, crossover, mutation.