УДК 621.314

А. Павельчак[1], В. Самотий[2,3], В. Остапчук[1]
[1]Національний університет "Львівська політехніка",
кафедра комп'ютеризованих систем автоматики
[2]Politechnika Krakowska im. Tadeusza Kościuszki,
katedra Automatyki i Technik Informacyjnych
[3]Львівський державний університет безпеки життєдіяльності,
кафедра управління інформаційною безпекою

ПАРАМЕТРИЧНА ОПТИМІЗАЦІЯ СИСТЕМИ РЕГУЛЮВАННЯ НАПРУГИ ГЕНЕРАТОРА ПОСТІЙНОГО СТРУМУ ГЕНЕТИЧНИМ АЛГОРИТМОМ

Розглянуто параметричну оптимізацію для системи регулювання напруги генератора постійного струму з електромашинним підсилювачем з використанням генетичного алгоритму. Отримано якісні характеристики перехідного процесу системи регулювання.

Ключові слова: оптимізація, генетичний алгоритм, система керування.

Parametric optimization of voltage control system of oscillator DC with dynamoelectric amplifier was considered. Optimization was conducted using Genetic Algorithm. Qualitative characteristics of the transition process of control system were obtained.

Key words: optimization, genetic algorithm, control system.

Вступ

Проблема параметричної оптимізації не є новою. Алгоритми її реалізації можуть бути застосовані до будь-чого, як для оптимізації витрат на виробництві, транспортних розв'язок, організації людської життєдіяльності, розведення тварин на фермі, так і для підбору параметрів математичних моделей, законів керування, у криптографії, задачах апроксимації тощо. Фактично, можливості застосування оптимізаційних алгоритмів обмежуються лише уявою дослідника.

Задача параметричної оптимізації полягає у відшуканні таких параметрів, за яких буде отримано локальний чи глобальний оптимум для цільової функції. Останній, звісно, бажаніший. Параметрична оптимізація системи керування дає можливість спроектувати її так, щоб вона якнайкраще відповідала визначеним для неї якісним критеріям. Вибір методу оптимізації залежить від наявної у нас інформації про об'єкт дослідження, його математичної моделі, її лінійності чи нелінійності та інших факторів. Якщо деякі методи залежать безпосередньо від об'єкта дослідження, то певні методи, такі як, наприклад, еволюційні методи, можуть бути використані для найширшого класу задач. У статті ми розглядаємо алгоритмічну реалізацію класичного генетичного алгоритму із застосуванням його для мультипараметричної оптимізації системи регулювання напруги генератора постійного струму з електромашинним підсилювачем.

Аналіз публікацій

Оптимізаційні алгоритми загалом поділяють на два класи: детерміністичні та ймовірнісні [1]. На кожному кроці виконання детерміністичного алгоритму існує лише один варіант для продовження його роботи. Якщо він відсутній, то алгоритм завершує свою роботу. Для однакових вхідних даних детерміністичний алгоритм завжди запропонує ті самі результати. Однак іноді виникають ситуації, коли детерміністичні алгоритми знаходять лише локальний оптимум. Якщо

зв'язок між отриманим рішенням і його придатністю не є очевидним, динамічно змінюється, надто складний чи діапазон пошуку дуже великий, використання більшості детерміністичних підходів є неефективним. От тоді і використовують імовірнісні алгоритми оптимізації. Як правило, точні алгоритми можуть бути набагато ефективнішими, аніж ймовірнісні в багатьох сферах. До того ж імовірнісні алгоритми мають додатковий недолік, вони можуть призвести до різних результатів під час кожного запуску для тих самих вхідних даних.

Серед детерміністичних алгоритмів [2, 3] можемо виділити такі групи методів: пошук у просторі станів (State space search), метод гілок і меж (Branch and bound), метод Гоморі (Cutting-plane method) та інші. Серед імовірнісних алгоритмів [4] виділимо такі: алгоритм пошуку зі сходженням до вершини (Hill climbing), алгоритм імітації відпалу (Cutting-plane method), Табу пошук (Tabu Search), екстремальна оптимізація (Extremal optimization), метод Нелдера–Міда (Downhill simplex). Окремою групою серед імовірнісних виділимо еволюційні алгоритми [4–8]: генетичні алгоритми, еволюційні стратегії, генетичне програмування, еволюційне програмування, диференціальна еволюція, алгоритм оцінки розподілу.

Постановка задачі

Мультипараметрична оптимізація системи регулювання напруги генератора постійного струму з електромашинним підсилювачем виконується за допомогою класичного бінарного генетичного алгоритму. Тому поставлене завдання була розділене на дві частини:

1. Аналіз та алгоритмічна реалізація як складових частин генетичного алгоритму, так і самого генетичного алгоритму загалом. Алгоритмічна реалізація повинна забезпечувати певну функціональність: довільний вибір основних операторів генетичного алгоритму, вибір діапазонів значень параметрів, кількості хромосом у популяції, кількості бітів для кодування генів, ймовірності мутації, кількості поколінь, можливості призупинення та відновлення роботи генетичного алгоритму, продовження пошуку оптимуму після досягнення встановленої кількості поколінь, а також збереження у файл на диску отриманої кінцевої популяції та відновлення її з файла для продовження пошуку оптимуму. Реалізація алгоритму повинна давати змогу призупиняти процес пошуку оптимуму, змінювати значення параметрів генетичного алгоритму та продовжувати далі роботу над пошуком оптимуму.

2. Пошук оптимуму для системи регулювання напруги генератора постійного струму. Якісним критерієм для оптимізації системи виберемо реакцію системи на зміну вхідного сигналу, тобто оптимізуватимемо якість перехідного процесу в системі керування.

Реалізація генетичного алгоритму

Тип даних для кодування хромосом у генетичному алгоритмі може бути поданий у бінарному, цілому чи дійсному вигляді. Ми вибрали бінарний запис, як найпридатніший для широкого класу задач. На рис. 1 подана розроблена нами структура даних для алгоритмічної реалізації бінарного генетичного алгоритму. Основним елементом генетичного алгоритму є хромосома, яка складається з n-кількості генів. Гени, своєю чергою, представляють параметри досліджуваної для оптимізації задачі. Кодування параметрів у бінарні гени здійснюється квантуванням (1) діапазону можливих змін значень параметрів.

$$step_i = \frac{valueMax_i - valueMin_i}{2^n - 1}, \quad gene_i = \frac{param_i - valueMin_i}{step_i}. \tag{1}$$

Зворотне декодування здійснюється за формулою (2):

$$param_i = valueMin_i + gene_i \cdot step_i. \tag{2}$$

В алгоритмічній реалізації хромосома є масивом генів беззнакового цілого типу даних. Для кодування параметрів можемо вибирати довільну кількість бітів, у нашому випадку від 8 до 32 бітів. Менша кількість бітів дає змогу швидше покривати простір досліджень, а більша кількість – точність дослідження. Розроблені алгоритми для операторів генетичного алгоритму працюють лише у межах вибраних бітів, решта старших бітів у масиві генів встановлені в нуль та ігноруються.

Основним критерієм для генетичного алгоритму є фітнес-значення, згідно з яким і проводиться оптимізація.

Використання кодування генів за допомогою коду Грея не дало ніякого помітного покращення результату роботи генетичного алгоритму, тому ми від нього відмовилися.

Основними параметрами генетичного алгоритму є кількість поколінь, через які повинен бути віднайдений шуканий оптимум, кількість хромосом у кожному поколінні, кількість генів у кожній хромосомі, кількість бітів для кодування кожного гена. Також мають бути встановлені діапазони зміни значень робочих параметрів досліджуваної моделі та внутрішні коефіцієнти для операторів генетичного алгоритму.

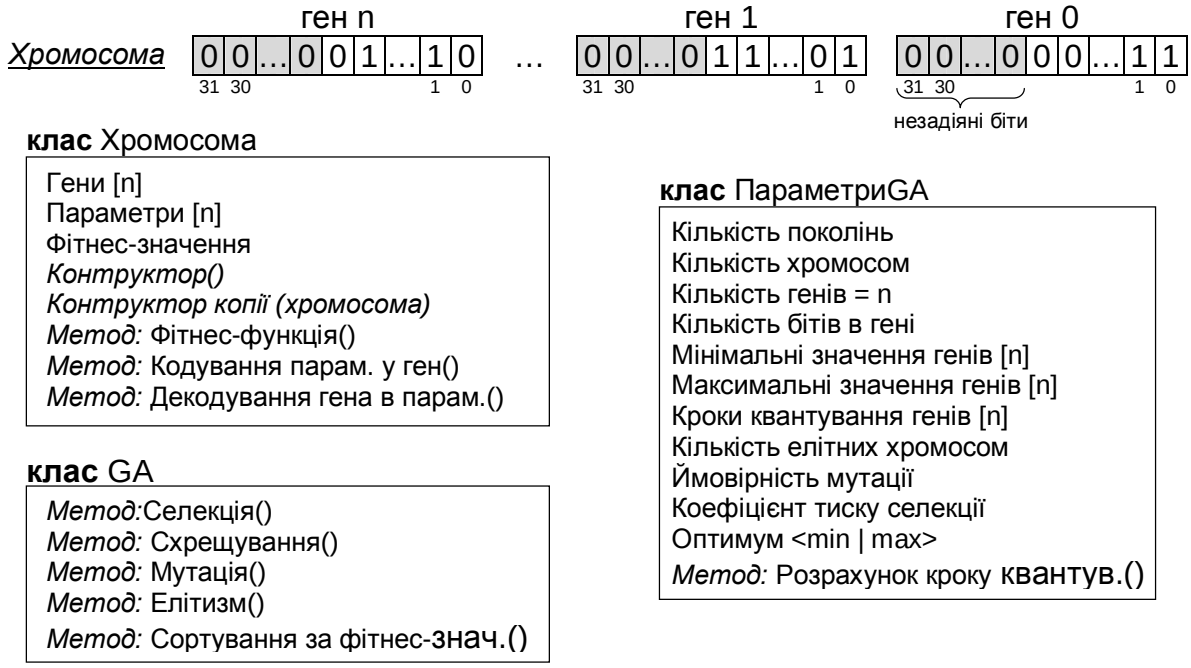

Рис. 1. Структура даних бінарного генетичного алгоритму

Розроблена нами реалізація бінарного генетичного алгоритму має таку послідовність дій:

1. *Створення початкової популяції з ініціалізацією хромосом та обчисленням фітнес-значень.* За створення та ініціалізацію відповідає конструктор класу хромосоми. Генерування бінарної стрічки виконується біт за бітом за допомогою генератора випадкових чисел. Тобто згенероване для кожного біта дійсне число від 0 до 1 заокруглюється, відповідно, в бінарний 0 чи 1. Початкова чисельність популяції залежить від типу задачі, що підлягає оптимізації. Теоретично, що більша чисельність, то краще. Далі в ході роботи генетичного алгоритму чисельність популяції можна буде зменшити.
2. *Сортування популяції за фітнес-значенням.* Виконується за найкращими фітнес-значеннями (мінімальними чи максимальними). Краще фітнес-значення при цьому міститься внизу популяції (останнє значення в масиві хромосом), гірше фітнес-значення – на початку популяції.
3. *Селекція.* Відбір хромосом для популяції здійснюємо за допомогою лінійного ранжування:

$$Fitneb(Pos) = 2 - SP + 2 \cdot (SP - 1) \cdot \frac{Pos - 1}{Nind - 1}, \quad (3)$$

де *Nind* – кількість хромосом (особин) в популяції; *Pos* – позиція хромосоми в популяції (найменш пристосовані хромосоми мають *Pos=1*, найбільш пристосовані – *Pos=Nind), SP* – коефіцієнт тиску селекції, який може набувати значення в межах [1,0; 2,0]. Згідно з отриманим значенням *Fitneb*

визначається кількість входжень хромосоми в популяцію. Додаткові копії кращих хромосом вставляємо із заміною гірших на початок популяції. Під час тестування генетичного алгоритму коефіцієнт *SP* виявився оптимальним, починаючи зі значення 1,6.

4. *Відбір елітних хромосом*. Копіюємо задану в параметрах генетичного алгоритму кількість кращих хромосом у проміжний буфер. Для копіювання використовуємо конструктор копій класу хромосоми.
5. *Схрещування*. Протестовано стандартні бінарні оператори схрещування (кросовери): одноточковий, двоточковий та Uniform. Якихось істотних відмінностей у кінцевому результаті між ними не виявлено, тому в алгоритмі використовуємо простий одноточковий кросовер. Зазначимо лише, що спаровування відбувається за ранжиром, спершу попарно кращі, а потім поступово до гірших. Варіант, коли кращі хромосоми спаровуються з гіршими хромосомами, загалом дав гірший кінцевий результат, як і варіант випадкового відбору хромосом для схрещування.
6. *Обчислення фітнес-значення для отриманих нащадків після схрещування*.
7. *Сортування популяції за фітнес-значенням*. Сортування у цьому місці необхідне для того, щоб виявити найпристосованішу хромосому, яка не повинна підлягати мутації.
8. *Мутація*. У параметрах генетичного алгоритму задається відсоток для ймовірності мутації бітів в усій популяції. На основі цього значення розраховується кількість бітів, що повинні мутувати, тобто інвертувати свої значення з 0 на 1, чи навпаки. Для цього створюємо масив масок, в яких визначаємо довільно позиції цих бітів. Спершу визначається випадково номер хромосоми в популяції, а потім номер біта в хромосомі. Після цього масив масок додається порозрядною операцією XOR до масиву хромосом, тобто виконуємо операцію інверсії. Найпристосованіша хромосома вилучається з процедури мутації. Під час тестування генетичного алгоритму значення відсотка мутації було оптимальним в межах 1–5 %.
9. *Сортування популяції за фітнес-значенням*. Цей крок можна пропустити, якщо значення елітизму дорівнює нулю.
10. *Елітизм*. Ця процедура вставляє на початок популяції відібрані елітні хромосоми у п. 4 із заміною гірших хромосом.
11. *Сортування популяції за фітнес-значенням*.
12. *Створення копії покоління*. Ця копія буде необхідною, якщо ми вирішимо продовжити пошук оптимуму за допомогою генетичного алгоритму. Можливий варіант, що під час роботи генетичного алгоритму захочеться змінити деякі із параметрів як об’єкта оптимізації, так і самого алгоритму, наприклад, зменшити відсоток ймовірності мутації чи кількості бітів у генах. Для цього потрібно буде зупинити роботу генетичного алгоритму, вибрати нові параметри та продовжити оптимізацію.
13. Якщо задана кількість поколінь пройдена, то завершуємо роботу генетичного алгоритму, а якщо ні, то переходимо до п.3.

Реалізація генетичного алгоритму виконувалася мовою C#. Відповідно, ми скористалися вбудованим генератором псевдовипадкових чисел з рівномірним законом розподілу (клас Random) з бібліотеки .NET Framework, який побудований на основі субтрактивного алгоритму генератора випадкових чисел Д. Кнута.

Реалізація генетичного алгоритму апробована на тестових задачах для оптимізації [9]: De Jong's function 1, Axis parallel hyper-ellipsoid function, Rotated hyper-ellipsoid function, Rastrigin's function 6, Schwefel's function 7. Для усіх тестових задач реалізований алгоритм знайшов визначені мінімуми та максимуми функцій.

Параметрична оптимізація системи регулювання напруги генератора

Система автоматичного регулювання напруги генератора постійного струму з електромашинним підсилювачем призначена для автоматичного регулювання напруги на виході генератора за заданою напругою. Генератори постійного струму різної потужності широко

використовуються в різних галузях промисловості. У системах автоматичного регулювання і управління, а також в системах стеження застосовуються електромашинні підсилювачі постійного струму. Розглянемо структурну схему системи регулювання напруги генератора постійного струму з електромашинним підсилювачем (рис. 2). Працює ця система так: з генератора постійного струму (Г) знімається вихідна напруга $U_{ВИХ}$, яка віднімається від задаючої напруги $U_З$ і результуюча напруга безпосередньо надходить на обмотку управління електромашинного підсилювача (ОУ ЕМП). Від якоря цього підсилювача живиться обмотка збудження генератора. На виході генератора буде напруга $U_{ВИХ}$. Якщо, наприклад, напруга генератора зменшиться, то напруга U_1 обмотки управління збільшиться, і, відповідно, буде збільшена напруга U_2 короткозамкненої обмотки генератора (КО ЕМП), а внаслідок цього зросте $U_{ВИХ}$. Напруга генератора завжди менша від задаючої напруги на значення U_1, яке достатнє для того, щоб створити напругу на генераторі, яка майже дорівнює задаючій напрузі. Різниця між задаючою напругою і вихідною напругою буде тим меншою, що вищий буде коефіцієнт підсилення електромашинного підсилювача. Отже, точність регулювання напруги в цій системі великою мірою залежить від коефіцієнта підсилення електромашинного підсилювача, а час проходження перехідного процесу залежить від сталих часу системи.

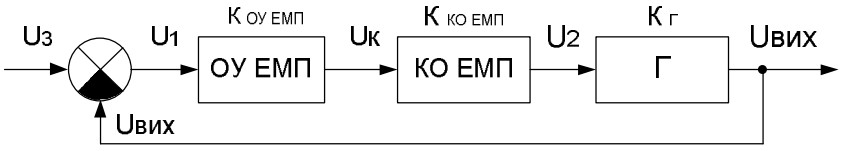

Рис. 2. Структурна схема системи регулювання напруги генератора постійного струму

Елементи цієї системи регулювання описуються такими рівняннями.

1. Схема порівняння:

$$U_1 = U_З - U_{ВИХ}\,. \tag{4}$$

2. Електромашинний підсилювач:

обмотка управління $$T_{K2}\frac{d^2U_K}{dt^2} + T_{K1}\frac{dU_K}{dt} + U_K = K_1U_1\,, \tag{5}$$

короткозамкнена обмотка $$T_K\frac{dU_2}{dt} + U_2 = K_2U_K\,. \tag{6}$$

3. Генератор постійного струму:

$$T_2\frac{dU_{ВИХ}}{dt} + U_{ВИХ} = K_З U_2\,. \tag{7}$$

Привівши рівняння (5)–(7) до нормальної форми Коші, можемо проінтегрувати систему та отримати якісну характеристику перехідного процесу. Ручний підбір коефіцієнтів для отриманої математичної моделі системи регулювання напруги генератора постійного струму потребує певного досвіду та все одно не дає змоги точно підібрати параметри для отримання необхідного оптимуму.

Для оптимізації характеристики перехідного процесу системи ми використали описаний вище генетичний алгоритм. Як критерій для значення фітнес-функції вибрано відхилення дискретних миттєвих точок перехідного процесу системи від задаючої напруги. Тобто фітнес-значення розраховують за такою формулою:

$$Fitness = \sum_i \left|U_З - U_{ВИХ}(t_i)\right|. \tag{8}$$

Моделювання системи виконуємо методом Рунге–Кутта четвертого порядку. Зауважимо, що за певних значень вибраних параметрів система може розбігатися або мати дуже значне перерегулювання. Тому ми внесли обмеження у фітнес-функцію під час інтегрування системи. Вважаємо, що якщо миттєві значення вихідної напруги $U_{ВИХ}$ більші у п’ять разів від задаючої, то

такі значення вибраних параметрів для нас неприйнятні. У цьому випадку повертаємо велике фітнес-значення для вказаної хромосоми (ми вибрали значення 10^5).

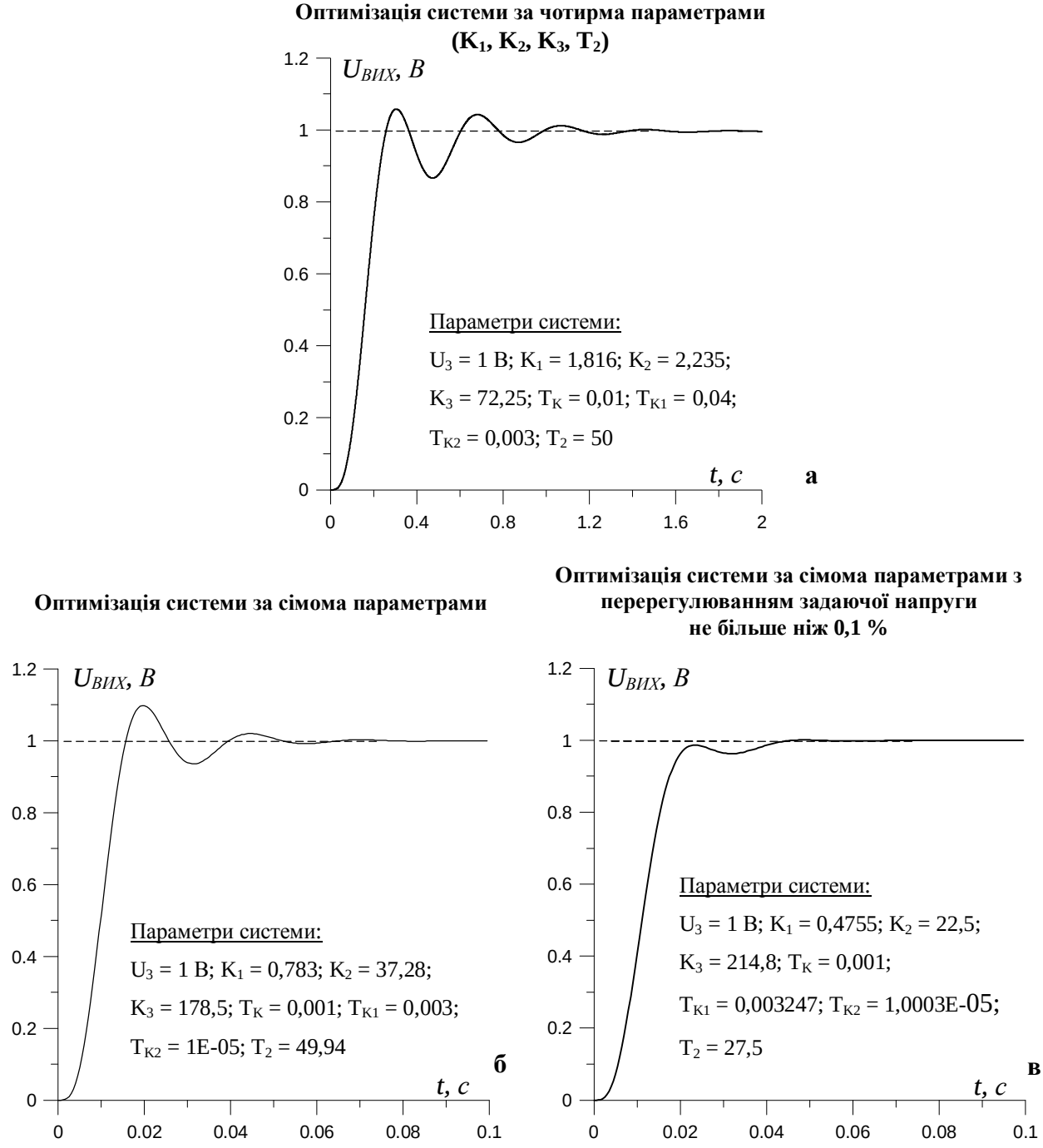

Рис. 3. Результати оптимізації системи регулювання напруги генератора постійного струму

На першому етапі дослідження оптимізація системи виконувалася за чотирма параметрами, для яких ми встановили такі межі: K_1 = [0,05; 10], K_2 = [0,2; 40], K_3 = [1; 500], T_2 = [0,5; 50]. Решта параметрів зафіксовані та мали такі значення: T_K = 0,01 с, T_{K1} = 0,04 с, T_{K2} = 0,003 с. Інтегрування системи методом Рунге-Кутта четвертого порядку виконувалося на інтервалі часу 5 с. За заданих параметрів за допомогою генетичного алгоритму віднайдено оптимум для характеристики перехідного процесу (рис. 3, а). Добитися завершення перехідного процесу менш ніж за 1,5 с нам не вдалося. Тому прийнято рішення задіяти для пошуку усі параметри системи.

На другому етапі дослідження оптимізація системи здійснювалося за сімома параметрами, для яких встановлено такі межі: $K_1 = [0,05; 10]$, $K_2 = [0,2; 40]$, $K_3 = [1; 500]$, $T_2 = [0,5; 50]$, $T_K = [0,001; 1]$ с, $T_{K1} = [0,001; 1]$ с, $T_{K2} = [0,00001; 1]$ с. Інтегрування системи виконувалося протягом 5 с. Зазначимо, що процес пошуку оптимуму відбувався значно повільніше, але і результати були набагато кращі. На рис. 3, б представлено характеристику перехідного процесу системи для знайдених параметрів генетичним алгоритмом. Фактично перехідний процес завершується за короткий інтервал часу в 0,1 с, що є хорошим результатом. Перерегулювання – близько 10 % від задаючої напруги. Тому ми вирішили виконати пошук оптимуму, внісши обмеження до 0,1 % перерегулювання у фітнес-функцію. На рис. 3, в наведено результат нашого пошуку. На відміну від попереднього результату, характеристика перехідного процесу має дещо менш різкий початковий фронт, але натомість загасання проходять плавніше.

Висновок

За допомогою бінарного генетичного алгоритму проведена мультипараметрична оптимізація системи регулювання напруги генератора постійного струму з електромашинним підсилювачем. Оптимізація дала можливість отримати якісні характеристики перехідного процесу системи регулювання. Ми змогли добитися як швидкої реакції системи на зміну задаючої напруги, так і плавного загасання перехідної характеристики.

1. Edwin K. P. Chong, Stanislaw H. Zak. An Introduction to Optimization, 4th Edition. – John Wiley & Sons. – 2013. – 640 p. 2. Reiner Horst and Tuy Hoang. Global Optimization: Deterministic Approaches, 3rd edition. – Springer-Verlag GmbH: Berlin, Germany, 1996. – 696 p. 3. Mordecai Avriel. Nonlinear Programming: Analysis and Methods. – Dover Publications: Mineola, NY, USA. – 2003. 4. Thomas Weise. Global optimization algorithms: theory and application, 3nd Edition. – Thomas Weise, 2011. – 1217 p. 5. Sivanandam S. N., Deepa S. N. Introduction to Genetic Algorithms. – Springer-Verlag Berlin Heidelberg, 2008. – 456 p. 6. Randy L. Haupt, Sue Ellen Haupt. Practical genetic algorithms. – 2nd ed. – John Wiley & Sons, Inc., Hoboken, New Jersey, 2004. – 272 p. 7. Mitchell Melanie. An Introduction to Genetic Algorithms. – A Bradford Book The MIT Press, 1999. – 162 p. 8. Субботін С. О., Олійник А. О., Олійник О. О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей. – Запоріжжя: ЗНТУ, 2009. – 375 с. 9. Hartmut Pohlheim. Examples of Objective Functions. – www.geatbx.com – 2006. – 21 p.