НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ КІБЕРНЕТИКИ ІМЕНІ В.М.ГЛУШКОВА

П.І. Стецюк, М.Г. Журбенко, О.П. Лиховид

МАТЕМАТИЧНІ МОДЕЛІ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ В ЗАДАЧАХ ЕНЕРГЕТИКИ

Рекомендовано до друку Вченою радою
Інституту кібернетики
ім. В.М. Глушкова НАН України
(Протокол № 8 від 29 травня 2012 р.)

Київ
2012

НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ КІБЕРНЕТИКИ ІМЕНІ В.М.ГЛУШКОВА

П.І. Стецюк, М.Г. Журбенко, О.П. Лиховид

МАТЕМАТИЧНІ МОДЕЛІ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ В ЗАДАЧАХ ЕНЕРГЕТИКИ

Рекомендовано до друку Вченою радою
Інституту кібернетики
ім. В.М. Глушкова НАН України
(Протокол № 8 від 29 травня 2012 р.)

Київ
2012
Ательє «Поліграфічний комплекс»

С79
УДК 519.8

ISBN 978-966-1668-07-1

Рецензенти:

А.М. Гупал, член-кор. НАН України
Є.О. Нурмінський, д-р фіз.-мат. наук
В.П. Шило, д-р фіз.-мат. наук

С79 П.І. Стецюк, М.Г. Журбенко, О.П. Лиховид. Математичні моделі та програмне забезпечення в задачах енергетики. – К.: 2012. – 64 с.

ISBN 978-966-1668-07-1

ISBN 978-966-1668-07-1

ЗМІСТ

ВСТУП

Ця робота присвячена результатам, отриманим при розробці та дослідженні нових інформаційних технологій із застосуванням теорії та методів недиференційовної оптимізації для розв'язання складних оптимізаційних задач, необхідних для прийняття оптимальних рішень при плануванні навантаження енергетичних об'єктів і при проектуванні енергетичних установок.

У розділах 1–5 досліджено оптимізаційні моделі для задач планування навантажень в енергосистемі, яка може включати теплові електростанції (ТЕС), атомні електростанції (АЕС) та гідроелектростанції (ГЕС). Ці моделі характеризуються плановим періодом, який складається із заданих інтервалів часу. Для кожного інтервалу визначена прогнозована потреба в потужності енергосистеми. Енергосистема складається з паралельно генеруючих енергоблоків, які відносяться до ТЕС, АЕС та ГЕС. Методи та програмне забезпечення для цих задач базуються на використанні алгоритмів негладкої оптимізації та методу послідовного аналізу варіантів.

Розділ 1 присвячено знаходженню оптимального (за сумарними витратами умовного палива) навантаження енергоблоків в енергосистемі на плановий період з урахуванням обмежень на екологічні фактори енергосистеми та можливості маневрування навантаженнями енергоблоків [1-3]. Екологічні параметри енергосистеми обмежують максимально допустимий рівень забруднення навколишнього середовища за весь плановий період. "Маневреність" енергосистеми характеризується обмеженнями як на сумарні зміни навантажень окремих енергоблоків за весь плановий період, так і на сумарні зміни навантажень за сусідні планові інтервали. Для даного класу задач енергоблоки не можуть відключатися і включатися в окремі інтервали планового періоду. Наведено результати чисельних експериментів з використанням стандартних програм для задач нелінійного програмування з NEOS-сервера (http://www-neos.mcs.anl.gov/) та розробленої програми на основі r-алгоритму.

У розділі 2 досліджена блочна цілочислова задача мінімального за витратами умовного палива навантаження енергосистеми на плановий період [3, 4]. Кожен блок у моделі відповідає окремому інтервалу планового періоду та представлений спеціальною задачею рюкзачного типу із сепарабельними функціями витрат умовного палива для енергоблоків. Алгоритм розв'язання задачі базується на використанні методу послідовного аналізу варіантів. Він реалізований програмою ModelA на мові РАТФОР (препроцесор Фортрану) та класом ModelAClass на мові С++. Проведено тестування розроблених програм для енергосистеми із 125 енергоблоками (така їх кількість характерна для Об'єднаної Енергетичної Системи України (ОЕС), включаючи ТЕС та ТЕЦ).

У розділі 3 наведено багатоекстремальну математичну модель для знаходження оптимального навантаження енергоблоків в енергосистемі на плановий період [5]. Для її розв'язання розроблено паралельний метод мультистарту, в якому для пошуку локальних розв’язків використовується r-алгоритм. Метод реалізовано

на мові програмування C++ у середовищі MPI та наведено результати обчислювальних експериментів на кластері СКІТ Інституту кібернетики імені В.М. Глушкова НАН України.

Розділ 4 присвячено задачі середньострокового планування завантаження енергоблоків регіональної енергосистеми. Її математична модель враховує потребу у визначеній регулярності (циклічності) режимів роботи енергоблоків, що відображає природну циклічність динаміки необхідної вихідної потужності у часі (наприклад, за робочими днями тижня). Модель подана нелінійною задачею математичного програмування. Алгоритм розв'язання задачі базується на використанні методу негладкої оптимізації (r-алгоритму). Програмне забезпечення розроблено на мові C++ у стилі об'єктно-орієнтованого програмування. Усі змістовні об'єкти задачі відображено відповідними класами C++. Алгоритм розв'язання і сервісні функції забезпечуються головним класом CPowerOperation.

У розділі 5 наведено модель короткострокового управління енергетичною системою. Знаходження розв'язку задачі базується на розв'язанні двоїстої задачі відносно зв'язуючих обмежень, для чого використано *r*-алгоритм. Розроблена програмна реалізація (на мові C++) описаного методу була випробувана для розв'язання задач оперативного управління енергетичною системою (використовувались запропоновані «Київенерго» вихідні дані задачі).

Розділ 6 присвячено задачі розрахунку теплових схем, що може використовуватися для оптимального вибору конструктивних параметрів теплових схем складних технічних об'єктів [6-8]. Метод розрахунку базується на використанні модельних теплообмінників та забезпечує обчислення похідних для вихідних температур теплообмінника за різними його конструктивними параметрами. Програмне забезпечення розроблено на мові C++ у стилі об'єктно-орієнтованого програмування. Всі змістовні об'єкти задачі відображено відповідними класами C++. Алгоритм розв'язання і сервісні функції забезпечуються головним класом CScheme_N. Ефективність методу розрахунку і програмного забезпечення випробувана при розв'язанні задачі оптимального проектування окремих підсистем промислових парових котлів.

1. ЗАДАЧІ ВИБОРУ ЕЛЕКТРИЧНИХ НАВАНТАЖЕНЬ В ЕНЕРГОСИСТЕМІ З УРАХУВАННЯМ ЕКОЛОГІЧНИХ ФАКТОРІВ ТА ЗГЛАДЖУВАННЯ ГРАФІКІВ НАВАНТАЖЕНЬ

1.1. ЛП-задача з "екологічними" обмеженнями

Нехай енергосистема складається з N паралельно працюючих енергоблоків. Для кожного енергоблока i ($i = 1,\ldots,N$) задані P_i^{low} і P_i^{up} – нижня і верхня границі його електричного навантаження[1], c_i – витрати умовного палива на вироблення одиниці електричного навантаження, a_{ik} – рівень забруднення навколишнього середовища k-м фактором на вироблення одиниці електричного навантаження, $k = 1,\ldots,K$. Нехай T – тривалість планового періоду. Плановий період розбито на однакові інтервали часу. Для кожного інтервалу t ($t = 1,\ldots,T$) задано планове електричне навантаження енергосистеми E_t . Вимоги на "екологічність" енергосистеми задано параметрами A_k , $k = 1,\ldots,K$, що характеризують максимально допустимий рівень забруднення навколишнього середовища енергосистемою за плановий період.

Нехай $x_{i,t}$ – невідоме електричне навантаження i-го енергоблока в інтервалі t планового періоду. Тоді математична модель задачі знаходження "екологічно" оптимального завантаження енергоблоків на плановий період описується наступною задачею оптимізації:

$$f^* = \min \sum_{i=1}^{N} \sum_{t=1}^{T} c_i x_{i,t} \tag{1.1}$$

при обмеженнях

$$\sum_{i=1}^{N} \sum_{t=1}^{T} a_{ik} x_{i,t} \le A_k, \quad k = 1,\ldots,K, \tag{1.2}$$

$$\sum_{i=1}^{N} x_{i,t} = E_t, \quad t = 1,\ldots,T, \tag{1.3}$$

$$P_i^{low} \le x_{i,t} \le P_i^{up}, \quad i = 1,\ldots,N, \quad t = 1,\ldots,T. \tag{1.4}$$

[1]Тут під електричним навантаженням розуміємо кількість електричної енергії, яку енергоблок може постачати в енергосистему. Реальна потужність енергоблока включає ще електричну енергію, що витрачається на власні потреби енергоблока, покриття втрат у мережі та інше.

Цільова функція (1.1) задає сумарні витрати умовного палива на вироблення електроенергії. Обмеження (1.2) означають виконання вимог на "екологічність" енергосистеми. Обмеження (1.3) гарантують виконання плану по електричній енергії у кожному інтервалі планового періоду. Обмеження (1.4) означають, що для кожного i-го енергоблока і кожного інтервалу t його електричне навантаження $x_{i,t}$ вибирається з неперервного діапазону $[P_i^{low}, P_i^{up}]$ його електричних навантажень.

Задача (1.1)–(1.4) є задачею лінійного програмування з блочною структурою матриці обмежень. Тут "екологічні" обмеження (1.2) пов'язують змінні задачі з незалежних блоків, кожен з яких пов'язаний зі своїм інтервалом t із планового періоду. Ця структурна особливість задачі враховується при розробці ефективних алгоритмів її розв'язання на основі декомпозиції і використанні методів мінімізації негладких опуклих функцій [9]. Їх складність визначається кількістю зв'язуючих обмежень (кількістю екологічних факторів). Такі алгоритми забезпечують ефективне розв'язання задачі (1.1)–(1.4) при великій кількості інтервалів планового періоду.

У задачі (1.1)–(1.4) одному й тому ж оптимальному значенню f^* може відповідати багато оптимальних розв'язків. Нехай $x_{i,t}^*$ – знайдені оптимальні значення електричних навантажень i-го енергоблока в інтервалі t планового періоду, $i = 1,\ldots,N$, $t = 1,\ldots,T$. Такий розв'язок може характеризуватися великими "перепадами" електричних навантажень енергоблоку при переході з інтервалу t в інтервал $(t+1)$. Неоднозначний оптимальний розв'язок задачі (1.1)–(1.4) призводить до питання: чи немає серед множини оптимальних розв'язків задачі (1.1)–(1.4) якогось іншого розв'язку з тим же значенням цільової функції f^*, але з більш плавним графіком завантаження всіх енергоблоків (або деякого сімейства енергоблоків)? Відповіді на нього дають розглянуті нижче моделі.

1.2. Задача з урахуванням маневреності

Нехай невід'ємний параметр Δ_t задає максимальну величину допуску на керування сумарною зміною навантажень усіх енергоблоків при переході з інтервалу t в інтервал $(t+1)$. Мінімальним значенням цієї характеристики є $|E_{t+1} - E_t|$, і воно буде реалізовуватися при $\Delta_t = 0$. Додамо до задачі (1.1)–(1.4) опуклі нерівності:

$$\sum_{i=1}^{N} |x_{i,t+1} - x_{i,t}| \le |E_{t+1} - E_t| + \Delta_t, \quad t = 1,\ldots,T-1. \tag{1.5}$$

Крім того, нехай з кожним i-м енергоблоком пов'язаний параметр Δ_i, який обмежує для нього сумарні переходи з режиму в режим за весь плановий період T. Додамо до задачі (1.1)–(1.5) такі опуклі нерівності:

$$\sum_{t=1}^{T} |x_{i,t+1} - x_{i,t}| \le \Delta_i, \quad i = 1,\ldots,N. \tag{1.6}$$

У задачі (1.1)–(1.6) за допомогою параметрів $\Delta_t, t = 1,\ldots,T-1$, і $\Delta_i, i = 1,\ldots,N$, можна забезпечити вибір електричних навантажень енергоблоків,

орієнтований або на мінімізацію змін навантажень енергоблоків між сусідніми інтервалами, або на мінімізацію переходів по навантаженнях для окремих енергоблоків за весь плановий період. Нерівності (1.5) дозволяють забезпечити невелику кількість нестандартних переходів для всіх сусідніх інтервалів планового періоду. Так, наприклад, одним з найкращих розв'язків буде такий оптимальний розв'язок, для якого f^* не змінюється по відношенню до задачі (1.1)–(1.4), але досягається при мінімальній сумі всіх значень параметра Δ_t. Обмеження (1.6) дозволяють одержати згладжені графіки електричних навантажень за плановий період для того сімейства енергоблоків, у якому параметр Δ_i є порівняно невеликим.

Задача (1.1)–(1.6) є задачею опуклого програмування з негладкими опуклими обмеженнями (1.5) і (1.6). Однак її можна звести до задачі лінійного програмування (ЛП-задачі), ввівши нові невід'ємні змінні $y_{i,t} = | x_{i,t} - x_{i,t-1} |$. Кожному $y_{i,t}$ будуть відповідати дві лінійні нерівності: $x_{i,t} - x_{i,t-1} \le y_{i,t}$ і $-y_{i,t} \le x_{i,t} - x_{i,t-1}$.

1.3. Нелінійна модель з урахуванням маневреності

Нехай з виробленням x одиниць електричного навантаження для i-ого енергоблока пов'язана нелінійна функція витрат умовного палива $f_i(x)$, а нелінійні функції $a_{ik}(x)$ описують вплив кожного з екологічних факторів $k = 1,\ldots,K$.

Забезпечимо нелінійну модель рядом можливостей для керування режимами завантаження енергоблоків, орієнтованих на активне використання обмежень (1.5) і (1.6). Для цього введемо додатково невід'ємні змінні: f_x для характеристики сумарних витрат умовного палива в енергосистемі; $y_t, t = 1,\ldots,T-1$, пов'язані з величиною допуску на керування сумарною зміною навантажень усіх енергоблоків при переході з інтервалу t в інтервал $(t+1)$; $z_i, i = 1,\ldots,N$, пов'язані із сумарною кількістю переходів по інтервалах планового періоду для кожного енергоблока. Параметрами керування для цих змінних зробимо верхні границі на їхні значення: f_{up} обмежує зверху змінну f_x, Δ_t – змінну y_t, а Δ_i – змінну z_t.

Для обмежень (1.4) введемо нижні ($x_{i,t}^{low}$) і верхні ($x_{i,t}^{up}$) границі електричних навантажень для кожного i-го енергоблока і кожного t-го інтервалу планового періоду. Керування ними дозволить локалізувати той чи інший варіант розв'язку. За допомогою цих границь можна промоделювати як фіксовані стартові навантаження для першого інтервалу $x_{i,1} = \bar{x}_{i,1}, \overline{i = 1,N}$, так і вихід енергосистеми на задані навантаження енергоблоків наприкінці планового періоду $x_{i,T} = \bar{x}_{i,T}$, $i = \overline{1,N}$.

Тоді нелінійну модель з урахуванням маневреності можна представити у формі наступної задачі математичного програмування:

$$f^*(\lambda_x, \lambda_y, \lambda_z) = \min \lambda_x f_x + \lambda_y \sum_{t \in T_\Delta} y_t + \lambda_z \sum_{i \in I_\Delta} z_i \qquad (1.7)$$

при обмеженнях

$$\sum_{i=1}^{N}\sum_{t=1}^{T} f_i(x_{i,t}) \le f_x, \quad 0 \le f_x \le f_{up}, \tag{1.8}$$

$$\sum_{i=1}^{N}\sum_{t=1}^{T} a_{ik}(x_{i,t}) \le A_k, \quad k = 1,\dots,K, \tag{1.9}$$

$$\sum_{i=1}^{N} x_{i,t} = E_t, \quad t = 1,\dots,T, \tag{1.10}$$

$$\sum_{i=1}^{N} | x_{i,t+1} - x_{i,t} | \le | E_{t+1} - E_t | + y_t, \quad t = 1,\dots,T-1, \tag{1.11}$$

$$\sum_{t=1}^{T-1} | x_{i,t+1} - x_{i,t} | \le z_i, \quad i = 1,\dots,N, \tag{1.12}$$

$$P_i^{low} \le x_{i,t}^{low} \le x_{i,t} \le x_{i,t}^{up} \le P_i^{up}, \quad i = 1,\dots,N, t = 1,\dots,T, \tag{1.13}$$

$$0 \le y_t \le \Delta_t, t = 1,\dots,T-1, \quad 0 \le z_i \le \Delta_i, i = 1,\dots,N. \tag{1.14}$$

Параметри λ_x, λ_y і λ_z дають можливість формувати різні цільові функції. Так, наприклад, при $\lambda_x = 1$, $\lambda_y = \lambda_z = 0$ цільова функція відповідає сумарним витратам умовного палива; при $\lambda_x = \lambda_z = 0$, $\lambda_y = 1$ – сумарним змінам електричних навантажень по всіх сусідніх інтервалах з множини T_Δ; при $\lambda_x = \lambda_y = 0$ і $\lambda_z = 1$ – сумарним змінам електричних навантажень за весь плановий період для енергоблоків з множини I_Δ. При $\lambda_x = \lambda_y = \lambda_z = 0$ розв'язок задачі (1.7)–(1.14) рівносильний перевірці сумісності системи обмежень (1.8)–(1.14).

Задача нелінійного програмування (1.7)–(1.14) з неперервними змінними має ту особливість, що нелінійні функції $f_i(\cdot)$, $a_{ik}(\cdot)$ сепарабельні (залежать тільки від невідомого електричного навантаження енергоблоку). Найважливішим є випадок, коли деякі з нелінійних функцій неопуклі. Тоді задача може бути багатоекстремальною і аналіз її розв'язків досить складний. Більш простим є випадок, коли всі нелінійні функції є опуклими. Тоді (1.7)–(1.14) є задачею опуклого програмування, для її розв'язання існує багато оптимізаційних програм. Проблеми тут пов'язані головним чином з розмірністю задач. Найпростішим є випадок, коли функції $f_i(\cdot)$ і $a_{ik}(\cdot)$ є лінійними. Тоді задачу (1.7)–(1.14) можна звести до задачі лінійного програмування таким же чином, як і задачу (1.1)–(1.6).

Задачу (1.7)–(1.14) і алгоритми її розв'язання доцільно використовувати для аналізу задач добового погодинного завантаження енергосистеми з паралельно працюючими енергоблоками, кількість яких не перевищує 25 і є достатньою для окремих енергокомпаній України. У випадку окремих енергокомпаній України задача (1.7)–(1.14) містить до 649 змінних, і її розв'язання можна забезпечити за допомогою r-алгоритму [9, 10]. Як альтернативний варіант для розв'язання цих задач можна використовувати відомі програми KNITRO, LOQO, MINOS, SNOPT. Вони входять у набір програм, за допомогою яких на оптимізаційному сервері NEOS (див. http://www-neos.mcs.anl.gov/) можна розв'язувати задачі нелінійного

програмування описані мовою моделювання AMPL [11]. Для кусочно-лінійних функцій $f_i(\cdot)$ і $a_{ik}(\cdot)$ задачу (1.7)–(1.14) легко описати на мові AMPL.

1.4. NEOS-програми для добового навантаження енергосистеми

Можливість використання NEOS-програм проілюструємо на прикладі задачі знаходження оптимального завантаження енергоблоків на плановий період. Їй відповідає наступна задача лінійного програмування

$$f^* = \min \sum_{i=1}^{N} \sum_{t=1}^{T} c_i x_{i,t} \tag{1.15}$$

при обмеженнях

$$\sum_{i=1}^{N} x_{i,t} = E_t, \forall t = 1,\ldots,T, \tag{1.16}$$

$$P_i^{low} \le x_{i,t} \le P_i^{up}, \forall i = 1,\ldots,N, \forall t = 1,\ldots,T. \tag{1.17}$$

Оскільки цільова функція (1.15) сепарабельна і лінійні обмеження (1.16) не зв'язують між собою змінні з різних інтервалів планового періоду, задача (1.15)–(1.17) розпадається на окремі незалежні підзадачі по інтервалах t. Лінійність цільової функції (1.15) і „рюкзачний" вигляд обмежень (1.16) для кожної незалежної підзадачі дозволяють вказати її аналітичний розв'язок. Сукупність таких розв'язків для окремих підзадач визначає аналітичний розв'язок задачі (1.15)–(1.17). Цей розв'язок завжди буде єдиним при відсутності співпадаючих значень витрат c_i. Якщо такі зустрічаються, оптимальний розв'язок може виявитися неоднозначним. При додаванні до задачі (1.15)–(1.17) опуклих обмежень

$$\sum_{i=1}^{N} \left| x_{it} - x_{it-1} \right| \le \left| E_t - E_{t-1} \right|, \ t = 2,\ldots,T, \tag{1.18}$$

оптимальне значення цільової функції не зміниться, а зміниться лише множина оптимальних розв'язків у випадку його неоднозначності. Обмеження (1.18) означає, що електричні навантаження енергоблоків при переході з інтервалу в інтервал повинні змінюватися мінімально. Задача (1.15)–(1.18) не розпадається на окремі підзадачі по інтервалах, тому що кожні два послідовних інтервали зв'язані обмеженнями (1.18). Ще більш тісний зв'язок підзадач по інтервалах зумовлений наступними обмеженнями:

$$\sum_{t=2}^{T} \left| x_{it} - x_{it-1} \right| \le \Delta_i, \forall i = 1,\ldots,N, \tag{1.19}$$

де Δ_i – параметр, який для i-го енергоблока обмежує сумарні переходи з режиму в режим за весь плановий період T. За допомогою нерівностей (1.19) можна промоделювати згладженість графіка навантажень того чи іншого сімейства енергоблоків за плановий період. Обмеження (1.18) і (1.19) дозволяють керувати мірою завантаження (маневруванням) енергоблоків у енергосистемі.

У загальному випадку (1.15)–(1.19) є задачею нелінійного опуклого програмування. Можливість розв'язання цих задач надає, зокрема, відомий оптимізаційний сервер NEOS [12]. Він дозволяє розв'язувати такі задачі в онлайн-режимі, для

чого достатньо задати програмі математичну модель задачі на мові моделювання AMPL. Програми NEOS-сервера було перевірено при знаходженні оптимальних навантажень енергоблоків.

Для тестових експериментів розглядалися п'ять варіантів задач цього сімейства. Варіант I відповідав задачі (1.15)–(1.17) , варіант II – задачі (1.15)–(1.18). Варіантові III відповідала задача (1.15)–(1.17), до якої додавалися аналоги обмежень (1.18), записані у формі

$$\sum_{i=1}^{N} \left| x_{it} - x_{it-1} \right| = y_t, t = 2,\ldots,T, \tag{1.20}$$

$$0 \le y_t \le \left| E_t - E_{t-1} \right|, t = 2,\ldots,T. \tag{1.21}$$

Варіант IV відповідає задачі (1.15)–(1.17), (1.19), де параметри $\Delta_i = \sum_{t=2}^{T} \left| x_{it}^* - x_{it-1}^* \right| \quad \forall i = 1,\ldots,N,$ обчислені на основі оптимального розв'язку задачі (1.15)–(1.17). Варіантові V відповідає задача (1.15)–(1.19), де параметри для обмежень (1.19) обчислювалися як і у варіанті IV. Усі п'ять варіантів задач мають однакове оптимальне значення цільової функції, а у випадку єдиного оптимального розв'язку задачі (1.15)–(1.17) вони мають єдиний оптимальний розв'язок.

Розглядалися два тестових приклади для $T = 24$ з різною кількістю енергоблоків. Моделі орієнтовані на добове погодинне навантаження в МВт; витрати задані у тоннах умовного палива на один МВт. Перший приклад включав 11 енергоблоків, другий – 14. Результати роботи програм NEOS-сервера для тестових задач наведено в таблиці 1.

Таблиця 1. Порівняння NEOS-програм для тестових прикладів

NEOS	Перший тестовий приклад					Другий тестовий приклад				
	I	II	III	IV	V	I	II	III	IV	V
Filter	+	-	-	+	-	+	+	+	-	-
Ipopt	+	-	-	-	-	+	-	-	-	-
KNITRO	+	+	+	+	+	+	+	+	+	+
LANCELOT	+	-	-	-	-	+	-	-	-	-
LOQO	+	+	+	+	+	+	+	+	+	+
MINOS	+	-	-	-	-	+	-	-	-	-
PENNON	-	+	-	-	+	-	+	-	-	+
SNOPT	+	+	+	-	-	+	+	+	+	+

Символ '+' означає, що метод успішно розв'язав задачу, а '-' – не зміг розв'язати задачу. З таблиці 1 видно, що з варіантом I впоралися всі програми, за винятком PENNON. Варіанти II–V включають негладкі опуклі нерівності і мають проблеми для ряду програм. Успішно впоралися з тестовими задачами тільки програми KNITRO, LOQO і SNOPT.

Ще більші проблеми для програм NEOS-сервера мають нелінійні задачі (1.7) –(1.14), де $f_i(\cdot)$, $a_{ik}(\cdot)$ – опуклі кусочно-лінійні функції. Розглядався тестовий

приклад для такої задачі з 11 енергоблоками, 24 інтервалами планового періоду та 2 екологічними факторами. У додатку А наведено код AMPL-програми та вхідні дані тестової задачі. Програми KNITRO і LOQO не розв'язали зазначену задачу, а програма SNOPT хоча і впоралась, але точність розв'язку виявилася недостатньою.

Для розв'язання задачі опуклого програмування розроблена програмна реалізація *r*-алгоритму – програма amplralg з підключенням її до інтерфейсу системи AMPL [13]. Для задач з обмеженнями використовується метод негладких штрафних функцій [9]. Опис математичної моделі задачі і вхідних даних виконувався відповідно до вимог мови моделювання AMPL. Скрипт для запуску програми amplralg та зразок текстового файлу з параметрами *r*-алгоритму наведено в додатку Б. Там же наведено результати розрахунків для тестового прикладу з 11 енергоблоками і 24 інтервалами планового періоду, які демонструють, що програма amplralg розв'язує цю задачу з заданою точністю.

Розглянуті моделі призначені для роботи з порівняно невеликою кількістю енергоблоків (декілька десятків) та розраховані на використання для окремих ТЕС і окремих енергокомпаній України. Для ОЕС України кількість енергоблоків у кілька разів більша. Навіть у випадку опуклих задач потрібно залучення більш потужних обчислювальних ресурсів. Для розв'язання цього класу задач великих розмірів природно використовувати суперкомп'ютер серії СКІТ [14]. У підрозділі 3 розглянуто використання кластера СКІТ для багатоекстремальної задачі навантаження енергоблоків.

Перспективним є розширення моделей за рахунок включення в енергосистему інших груп електростанцій, наприклад, АЕС і ГЕС. Так, у роботі [15] розглядаються такого ж типу оптимізаційні моделі для задачі оптимального керування енергосистемою, що складається з двох груп електростанцій: ТЕС і ГЕС. У задачі визначення графіків оптимальних режимів роботи цих електростанцій по інтервалах планового періоду гідроелектростанції, на відміну від ТЕС, характеризуються здатністю за малий (щодо тривалості часових інтервалів планового періоду) час істотно змінювати потужність. Включення в енергосистему окремих ГЕС дасть можливість будувати більш економні (щодо зносу устаткування) графіки завантаження енергоблоків ТЕС.

Такого типу моделі можуть використовуватися при прийнятті оптимальних рішень у задачах добової диспетчеризації. Вони дозволять уникнути нераціональних за технологією реалізацій планів завантаження енергоблоків, наприклад, з погляду "ступінчастості" їхніх графіків навантаження.

2. БЛОЧНА ЦІЛОЧИСЛОВА ЗАДАЧА ОПТИМАЛЬНОГО НАВАНТАЖЕННЯ ЕНЕРГОСИСТЕМИ НА ПЛАНОВИЙ ПЕРІОД

2.1. Постановка задачі та її аналіз

Припустимо, що енергосистема складається з N паралельно працюючих енергоблоків. Для кожного енергоблока i ($i = 1,\ldots,N$) задано набір можливих режимів роботи $K_i = \{1,\ldots,m_i\}$; для усіх $k \in K_i$ задано p_{ik} – цілі значення електричних навантажень $P_i = \{p_{i1},\ldots,p_{im_i}\}$ і відповідні витрати умовного палива $f_i = \{f(p_{i1}),\ldots,f(p_{im_i})\}$. Множина режимів K_i може включати режим 1 з потужністю $p_{i1} = 0$; це означає, що енергоблок i виключений. Ніяких припущень на вигляд функції витрат умовного палива не накладається. Нехай T – тривалість планового періоду (наприклад, у годинах). Для кожного інтервалу t ($t = 1,\ldots,T$) задана кількість електроенергії E_t (E_t – цілі числа), яку повинні надати енергоблоки.

Потрібно знайти такі навантаження енергоблоків у кожний окремий інтервал планового періоду, щоб забезпечувалася для них задана кількість електроенергії, а сумарні витрати на виробництво електроенергії за весь плановий період були мінімальними.

Нехай x_{it} – невідоме навантаження i-го енергоблока в інтервалі t планового періоду. Тоді математична модель задачі оптимального (за сумарними витратами умовного палива) завантаження енергоблоків на плановий період може бути сформульована у вигляді такої дискретної задачі оптимізації:

$$f_A^* = \min \sum_{i=1}^{N}\sum_{t=1}^{T} f_i(x_{it}) \tag{2.1}$$

при обмеженнях

$$\sum_{i=1}^{N} x_{it} = E_t, \quad \forall t = 1,\ldots,T, \tag{2.2}$$

$$x_{it} \in P_i, \quad \forall i = 1,\ldots,N, \forall t = 1,\ldots,T. \tag{2.3}$$

Оптимізаційну задачу (2.1)–(2.3) з блочною структурою матриці обмежень назвемо задачею А. Кожен із T блоків пов'язаний з окремим інтервалом t планового періоду, а між собою блоки ніяк не пов'язані. Цільова функція (2.1) сепарабельна. Розв’язання задачі (2.1)–(2.3) зводиться до розв’язання окремих T підзадач, кожна з яких пов'язана з окремим інтервалом t планового періоду.

У загальному випадку при великих $E_1,\ldots,E_T$ задача (2.1)–(2.3) досить складна і для її розв'язання неможливе використання класичних методів (наприклад, необхідних умов Лагранжа). Однак, те, що цільова функція є сепарабельною, дає можливість застосувати для її розв'язання метод послідовного аналізу варіантів (ПАВ), який реалізує ідеї динамічного програмування [16-19].

2.2. Алгоритм PAV, функції PAVTab і PAVSol

З огляду на те, що функції витрат умовного палива для енергоблоків не залежать від інтервалу t, застосування методу ПАВ для розв'язання задачі (2.1)–(2.3) пов'язано з однократною побудовою системи функціональних рівнянь для наступної задачі математичного програмування:

$$\min \sum_{i=1}^{N} f_i(x_i), \tag{2.4}$$

$$\sum_{i=1}^{N} x_i = E_{max}, \tag{2.5}$$

$$x_i \in P_i = \left\{ p_{i1},\ldots,p_{im_i} \right\}, \quad i = 1,\ldots,N. \tag{2.6}$$

де $E_{max} = \max\{E_1,\ldots,E_T\}$ – ціле додатнє число, $f_i(x_i)$ – невід'ємні дійсні значення функції витрат умовного палива для енергоблока i, задані для цілих значень електричних навантажень $P_i = \{p_{i1},\ldots,p_{im_i}\}$.

Для задачі (2.4)–(2.6) система функціональних рівнянь Белмана представляється в такий спосіб:

$$F_1(y) = \begin{cases} f_1(y), & y \in P_1, \\ \infty, & y \notin P_1, \end{cases} \tag{2.7}$$

$$F_k(y) = \min_{x \in P_k} \left\{ F_{k-1}(y), \quad f_k(x) + F_{k-1}(y-x) \right\}, \quad k = 2,3,\ldots,N, \tag{2.8}$$

де $0 \le y \le E_{max}$; $F_k(y)$ – функції Белмана, $k = 1,\ldots,N$.

Алгоритм на основі співвідношень (2.7), (2.8) умовимося називати алгоритмом PAV. Він вимагає тільки додавання дійсних чисел і не залежить від вигляду функцій $f_i(x_i)$, $i = 1,\ldots,N$. Трудомісткість алгоритму PAV визначається кількістю операцій додавання для розрахунку співвідношень (2.7), (2.8), яке обмежено зверху величиною $E_{max} \times M$. Тут $M = \sum_{i=1}^{N} m_i$ – повна кількість режимів роботи всіх N енергоблоків.

Нехай для всіх y таких, що $0 \le y \le E_{max}$, побудовано таблиці $F_k(y), k = 1,\ldots,N$, і їм відповідають таблиці $iF_k(y), k = 1,\ldots,N$, що містять ті значення змінних $x \in P_k$, при яких реалізуються k-субоптимальні розв'язки для кож-

ного $k = 1,\ldots,N$.[2] Тоді оптимальний розв'язок задачі (2.4)–(2.6) може бути знайдений як для значення правої частини E_{max}, так і для будь-якого іншого значення правої частини $E < E_{max}$. Знаходження оптимального розв'язку для довільного $0 < E \le E_{max}$ пов'язано з послідовним визначенням тих значень електричних навантажень для кожного $k = N, N-1,\ldots,1$, при яких реалізуються k-субоптимальні розв'язки $iF_k(y)$. Реалізацію алгоритму PAV визначимо двома функціями. Перша (назвемо її PAVTab) реалізує побудову для всіх y, таких, що $0 \le y \le E_{max}$, таблиць $F_k(y), iF_k(y), k = 1,\ldots,N$. Друга (назвемо її PAVSol) на основі побудованих функцією PAVTab таблиць знаходить оптимальний розв'язок задачі (2.4)–(2.6) для довільної правої частини E в обмеженні (2.5) ($0 < E \le E_{max}$). Тоді алгоритм розв'язання задачі (2.1)–(2.3) вимагатиме однократного виклику функції PAVTab і виклику не більш, ніж T раз функції PAVSol (T викликів буде тоді, коли серед значень $E_t, t = 1,\ldots,T$, немає однакових). Опис функцій–підпрограм PAVTab і PAVSol на мові РАТФОР (препроцесор Фортрану) наведено у додатку В.

2.3. Програма ModelA, клас ModelAClass і тестовий приклад

На основі підпрограм PAVTab і PAVSol для розв'язання задачі (2.1)–(2.3) реалізована Фортран-програма ModelA. У ній підпрограма PAVSol використовується при знаходженні оптимального розв'язку для кожного окремого інтервалу на основі один раз побудованої таблиці Белмана для максимального значення $E_{max} = \max\{E_1,\ldots,E_T\}$. Код програми ModelA мовою РАТФОР наведено в [3]. Програма ModelA розрахована на роботу з не більше як 125 енергоблоками; дозволяє 25000 режимів для всіх енергоблоків; дозволяє 24 інтервали в плановому періоді, тобто розрахована на добове погодинне завантаження енергоблоків; максимальна потреба в електроенергії для підперіодів дорівнює 30000 МВт, і її з запасом повинно вистачити при часовій потребі в ЕЕ для ОЕС України, яку дають разом всі енергоблоки ТЕС і ТЕЦ.

Максимальні розміри для задачі (2.1)–(2.3) встановлено на підставі таблиці 2. У ній відображено наявні енергоблоки для всіх ТЕС і ТЕЦ, що беруть участь у виробленні ЕЕ в ОЕС України. За кожним енергоблоком ТЕС або ТЕЦ закріплено свій унікальний номер. Поряд з повними назвами ТЕС і ТЕЦ у таблиці 2 дані ті скорочені назви, що використовуються в "ДП <<Енергоринок>>".

Так, для 14 ТЕС і 3 ТЕЦ, що є учасниками ринку ЕЕ України, число енергоблоків дорівнює 115, що не перевищує передбачені 125. Якщо задача розв'язується для окремих ТЕС і ТЕЦ, то число енергоблоків буде порівняно невеликим (наприклад, 2 блоки – Київська ТЕЦ-6, 3 блоки – Харківська ТЕЦ-5, 12 блоків – Бурштинська ТЕС). Для них рівень дискретизації по потужностях можна робити більш тон-

[2] Під k-субоптимальним розв'язком будемо розуміти оптимальний розв'язок задачі вигляду (2.4)–(2.6)), яка включає k перших енергоблоків. N–субоптимальний розв'язок співпадає з оптимальним розв'язком задачі (2.4)–(2.6).

ким, ніж 1 МВт, наприклад, 0.5 МВт і менше, з огляду на те, що запасу в 30000 МВт по максимальній потребі для цього повинно бути достатньо.

На основі програми ModelA реалізовано клас ModelAclass на мові C++. Клас ModelAClass містить два **конструктори**:

ModelAclass::ModelAclass (int mode, char input_files[3][80])

Аргументи**:**

mode	режим Debug вкл/викл: mode=0/1
input_files	імена файлів: *input_files*[0]-потужності станцій, *input_files*[1]-потреби в електроенергії *input_files*[2]-файл результатів розв'язання

Таблиця 2. Енергоблоки ТЕС і ТЕЦ України (http://www.er.gov.ua/)

№	Назва станції	Номери блоків	Кількість блоків
1	Бурштинська ТЕС (БуТЕС)	№1-№12	12
2	Вуглегірська ТЕС (ВуТЕС)	№1-№7	7
3	Добротвірська ТЕС (ДоТЕС)	№4-№8	5
4	Запорізька ТЕС (Затес)	№1-№7	7
5	Зміївська ТЕС (ЗмТЕС)	№1-№10	10
6	Зуївська ТЕС (ЗуТЕС)	№1-№4	4
7	Криворізька ТЕС (КРЕС)	№1-№10	10
8	Київська ТЕЦ-5 (КТЕЦ-5)	№1-№4	4
9	Київська ТЕЦ-6 (КТЕЦ-6)	№1-№2	2
10	Курахівська ТЕС (КуТЕС)	№3-№9	7
11	Ладижинська ТЕС (ЛадТЕС)	№1-№6	6
12	Луганська ТЕС (ЛуТЕС)	№4,№5,№8-№15	10
13	Придніпровська ТЕС (ПдТЕС)	№7-№14	8
14	Старобешівська ТЕС (СбТЕС)	№4-№13	10
15	Слов'янська ТЕС (СлТЕС)	№3,№5,№6,№7	4
16	Трипільська ТЕС (ТпТЕС)	№1-№6	6
17	Харківська ТЕЦ-5 (ХТЕЦ-5)	№1-№3	3
	Усього		115

ModelAclass::ModelAclass (int mode, int n_stations, int n_regims, int *index, int *power, double *cost, int n, int *demand_t)

Аргументи:

mode	Режим Debug вкл/викл: mode=0/1
n_stations	кількість станцій
n_regims	кількість допустимих режимів роботи станцій
index	номера станцій

power	значення допустимих потужностей для станцій
cost	витрати умовного палива при заданій потужності
n	кількість інтервалів часу
demand_t	потреби в електр. енергії для інтервалів часу

Методи:
void ModelAclass::Model (void)
знаходить оптимальний розв'язок для планового інтервалу
int ModelAclass::GetNerror (void)
одержання коду завершення програми
void ModelAclass::WriteResults (char **output_file_name*)
виводить інформацію про результати розв'язання у вихідний файл

Аргументи:

output_file_name	ім'я файлу результатів

Для оцінки часу роботи програми ModelA та класу ModelAclass при розв'язанні задачі (2.1)–(2.3), розраховуючи на ОЕС України, використовувався тестовий приклад з наступними параметрами. Розглядалося 125 енергоблоків з повною кількістю потужностей 19000 МВт, де кожен енергоблок включає 151 потужність з діапазону 120–270 МВт і нульову потужність, тобто будь-який з енергоблоків можна відключати. Витрати умовного палива генерувалися у діапазоні від 1 до 1000. Плановий період містив 24 інтервали, потреби в ЕЕ для них наведено на рисунку 1.

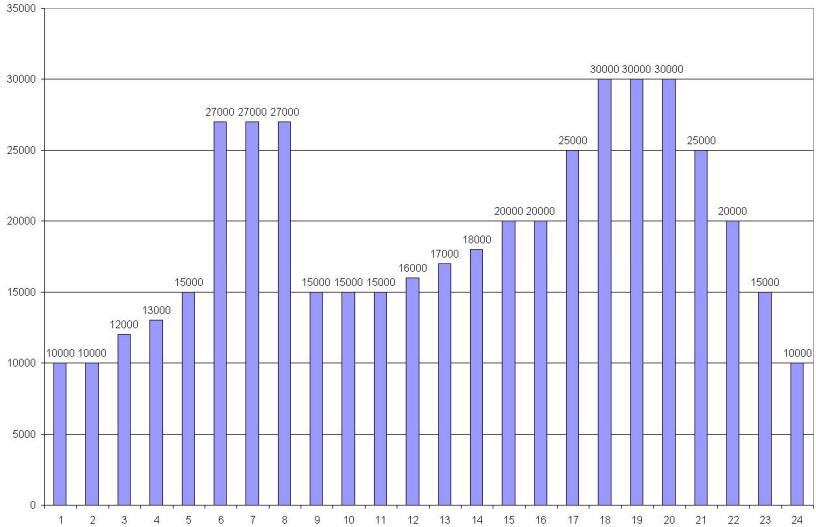


Рисунок 1. Потреби енергосистеми в електричній енергії (24 інтервали)

На розв’язання тестового прикладу на процесорі Pentium-III/733 Мгц програмою ModelA було витрачено 15 сек (з них 4 сек. – на обробку даних і 11 сек. – на роботу методу ПАВ). Час роботи класу ModelAClass становить близько 10 сек.

3. БАГАТОЕКСТРЕМАЛЬНА ЗАДАЧА НАВАНТАЖЕННЯ ЕНЕРГОСИСТЕМИ ТА ПАРАЛЕЛЬНИЙ АЛГОРИТМ ЇЇ РОЗВ'ЯЗАННЯ

3.1. Математична модель

Розглянемо багатоекстремальну оптимізаційну задачу навантаження енергосистеми:

$$f_C^* = \min \sum_{t=1}^{T} \sum_{i=1}^{N} f_i(x_{it}) \tag{3.1}$$

при обмеженнях

$$\sum_{i=1}^{N} x_{it} = E_t, \quad \forall t = 1,\ldots,T, \tag{3.2}$$

$$P_i^{low} \le x_{it} \le P_i^{up}, \quad \forall i = 1,\ldots,N, \quad \forall t = 1,\ldots,T. \tag{3.3}$$

Тут $f_i(x_{it})$ – функція витрат умовного палива для i-го енергоблока, яка для тестового прикладу має такий вигляд:

$$f_i(x_{it}) = a_i(x_{it} - \Delta_i)^3 + b_i(x_{it} - \Delta_i)^2 + c_i(x_{it} - \Delta_i) + d_i,$$

де $a_i, b_i, c_i, d_i, \Delta_i$ – задані параметри.

Використовуючи метод штрафних функцій [9], зведемо задачу (3.1)–(3.3) до задачі безумовної мінімізації наступної негладкої функції:

$$F(x) = \sum_{i=1}^{N} \sum_{t=1}^{T} f_i(x_{it}) + Q_1 \sum_{t=1}^{T} \left| \sum_{i=1}^{N} x_{it} - E_t \right| + Q_2 \sum_{t=1}^{T} \sum_{i=1}^{N} \max\{0, x_{it} - P_i^{up}, P_i^{low} - x_{it}\} \tag{3.4}$$

Тут Q_1 і Q_2 – штрафні множники, що відповідають урахуванню за допомогою негладких штрафів обмежень (3.2) у формі рівностей і обмежень (3.3) у формі нерівностей.

Функція (3.4) є багатоекстремальною. Для знаходження її оптимального розв’язку використовувався нищеописаний паралельний метод мультистарту, у якому для пошуку локальних розв’язків був обраний r-алгоритм [9].

3.2. Паралельна реалізація методу мультистарту

Метод мультистарту для знаходження оптимального розв’язку багатоекстремальної задачі можна представити у вигляді наступної процедури: за допомогою алгоритмів локального пошуку знаходяться локальні розв’язки з різних початкових точок, які генеруються випадковим або детермінованим способом, і серед отриманих розв’язків вибирається найкращий. При досить великій кількості запусків по-

шуку локальних розв’язків існує імовірність знайти оптимальний розв’язок. Очевидно, що цей метод є досить трудомістким, але його можна легко розпаралелити і реалізувати обчислення, наприклад, на кластері.

Процес розв'язання задачі проводиться на p процесорах. Згенерована випадковим чином у "головному" (Master) процесорі початкова точка передається на будь-який інший вільний (Slave) процесор за допомогою операції пересилання системи MPI. Там відбувається обчислення локального розв’язку для цієї початкової точки за допомогою алгоритму локальної оптимізації. Потім знайдений розв’язок передається в "головний" процесор, де відбувається порівняння отриманого розв’язку з найкращим із знайдених до цього моменту розв’язком (рекордним). Якщо поточний розв’язок кращий за рекордний, то він стає рекордним. Після закінчення заданої кількості запусків пошуку локального розв’язку рекордний приймається за розв’язок вихідної задачі.

Вищеописаний паралельний метод мультистарту реалізовано на мові програмування C++ у програмному середовищі MPI [20]. Для знаходження локальних розв’язків використовувався r-алгоритм [9]. Проведено ряд обчислювальних експериментів розв’язання тестових задач завантаження енергоблоків на кластерному комплексі СКІТ Інституту кібернетики. Далі наведено результати для двох тестових задач.

Перша тестова задача мала наступні значення параметрів: кількість енергоблоків – 10, що відповідає, наприклад, кількості таких блоків для Криворізької ТЕС. Число інтервалів у плановому періоді – 24. Значення потреб в енергії для них вибиралися відповідно вигляду графіка рисунка 1. Потужності енергоблоків вибиралися в інтервалі 120–280. Графік функції витрат умовного палива наведено на рисунку 2.

Друга тестова задача мала наступні значення параметрів: кількість енергоблоків – 127, що відповідає кількості таких блоків для ОЕС України; число інтервалів у плановому періоді – 24. Значення потреб в енергії для них вибиралися відповідно вигляду графіка рисунка 1. Потужності енергоблоків вибиралися в інтервалі 120–1100. Графік функції витрат умовного палива для інтервалу 120–280 наведено на рисунку 2 (для інших інтервалів його вигляд подібний).

Кількість точок, що генеруються для методу мультистарту і, відповідно, запусків пошуку локального розв’язку, для обох тестових задач дорівнювала 25. Ці точки генерувалися випадковим чином за допомогою функції рівномірного розподілу. Загальна кількість процесорів кластера змінювалася від 1 до 26.

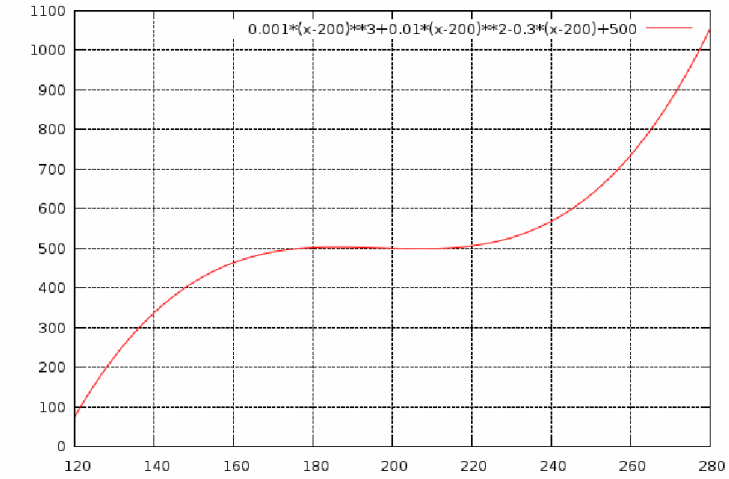


Рисунок 2. Графік функції витрат для тестових задач

Результати обчислювальних експериментів для першої і другої тестової задачі наведені відповідно в таблицях 3 і 4. Тут n – загальне число процесорів, t – час розв'язання задачі (в сек.), N_slave – число slave-процесорів, S_i – прискорення, E_i – ефективність паралельного алгоритму. На рисунках 3 і 5 наведено залежності часу розв'язання від кількості використаних процесорів, а на рисунках 4 і 6 – графіки прискорення у порівнянні з «ідеальним» лінійним прискоренням відповідно для першої і другої тестової задачі. У процесі обчислень знайдено кілька різних локальних оптимальних розв'язків, серед яких був обраний розв'язок з найменшим значенням функції витрат.

Таблиця 3. Час розв'язання, прискорення й ефективність для першої тестової задачі

n	N_slave	t	S_i	E_i
1	0	68,47	1	1,00
2	1	87,99	0,78	0,39
3	2	35,24	1,94	0,65
4	3	24,54	2,79	0,70
8	7	14,25	4,80	0,60
11	10	10,43	6,56	0,60
16	15	7,26	9,43	0,59
21	20	5,6	12,23	0,58
26	25	3,32	20,62	0,79

Таблиця 4.Часрозв'язання, прискорення й ефективність для другої тестової задачі

n	N_slave	t	S_i	E_i
1	0	79999	1	1
2	1	79792	1,01	0,5
3	2	47252	1,69	0,56
4	3	33873	2,36	0,59
8	7	23520	3,40	0,43
11	10	14841	5,39	0,49
16	15	12682	6,31	0,39
21	20	9462	8,45	0,40
26	25	7546	10,60	0,41

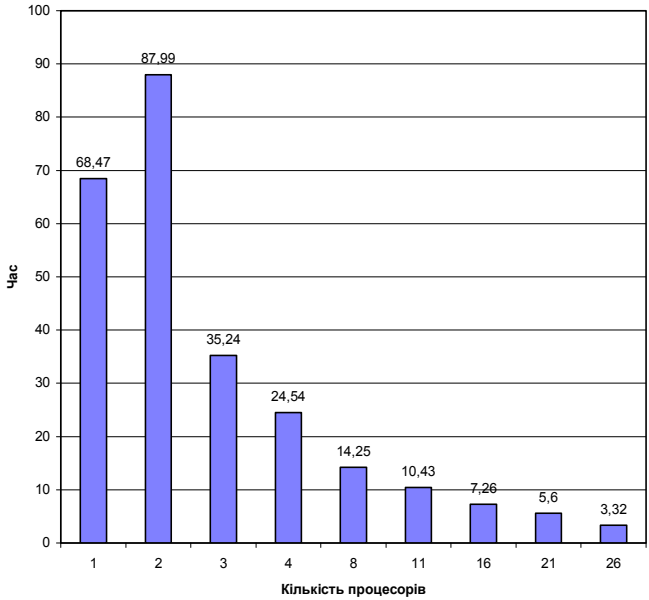


Рисунок 3. Залежність часу розв'язання від кількості процесорів для першої тестової задачі

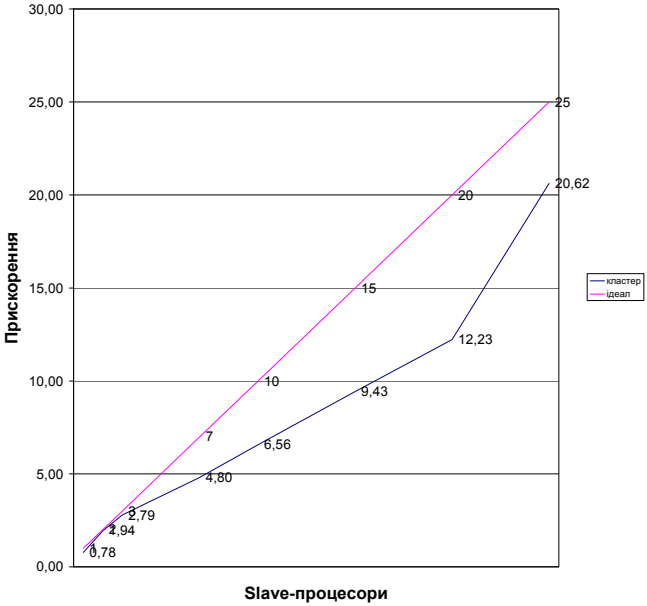


Рисунок 4. Прискорення для першої тестової задачі

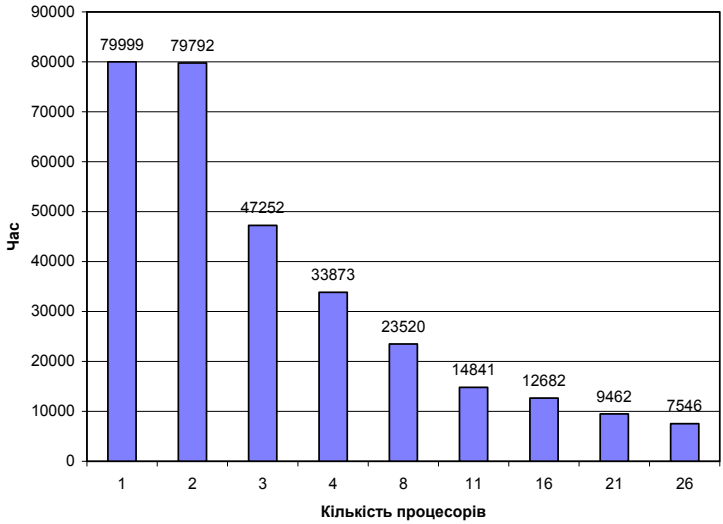


Рисунок 5. Залежність часу розв'язання від кількості процесорів для другої тестової задачі

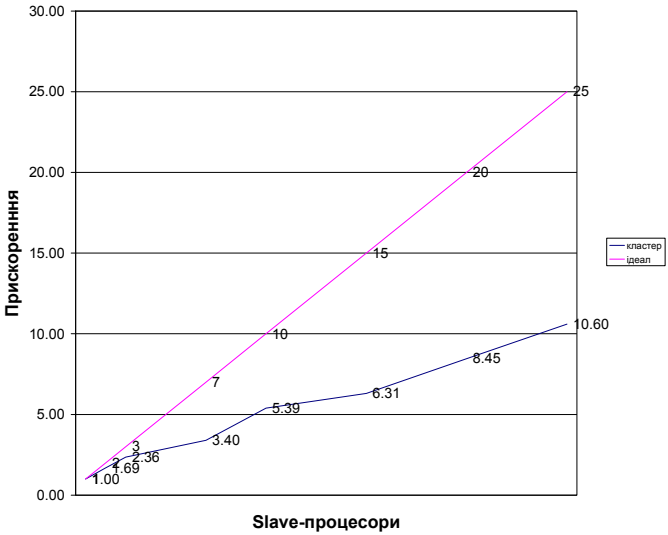


Рисунок 6. Прискорення для другої тестової задачі

З таблиці 3 і рисунка 4 видно, що для першої тестової задачі для семи slave-процесорів час розв'язання задачі зменшується в 4,8 рази, для десяти – у 6,56 рази, двадцяти – у 12,23 рази, а для двадцяти п'яти – у 20,62 рази. З таблиці 4 і рисунка 6 видно, що для другої тестової задачі для семи slave-процесорів час розв'язання задачі зменшується в 3,4 рази, для десяти – у 5,39 рази, для двадцяти – у 8,45 рази, а для двадцяти п'яти – у 10,6 рази.

4. ЗАДАЧА ВИБОРУ РЕЖИМІВ ЕНЕРГОСИСТЕМИ

4.1. Математична модель

Розглянемо задачу середньострокового планування завантаження енергоблоків регіональної енергосистеми. Нехай планований період (наприклад, доба, тиждень, місяць) складається з T малих (наприклад, година) інтервалів,. Ці інтервали будемо називати атомарними. Для кожного атомарного інтервалу t визначена прогнозована потреба у генерованій потужності $p^r(t)$.

Енергосистема складається з n генеруючих енергоблоків, кожен з яких може знаходитися в активному (включеному) або пасивному (відключеному) стані. В активному стані енергоблок j може генерувати потужність у діапазоні $[0, P_j]$. Переведення блоку з пасивного в активний стан вимагає деякого часу і пов'язано з деякими витратами. Питомі витрати на вироблення блоком j одиниці електроенергії визначаються функцією $f_j(p)$, що залежить від поточної потужності працюючого блока. Таким чином, якщо енергоблок працює при потужності p час τ, то витрати на вироблення електроенергії $p\tau$ дорівнюють $f(p)p\tau$. Вважається, що функція $f_j(p)$ опукла і досягає мінімуму при деякій номінальній для даного блока потужності p_j^* (при якій ККД блока максимальний). Відзначимо, що реальна функція питомих витрат може бути більш складною, але для розв'язання задачі середньострокового планування можна задовольнитися апроксимацією реальної функції функцією із зазначеними властивостями.

На змістовному рівні задача вибору режимів енергосистеми полягає у визначенні для кожного інтервалу планового періоду стану енергоблоків, тобто генерованої потужності для працюючих у даному інтервалі енергоблоків. При цьому для кожного інтервалу t загальна генерована потужність повинна бути не менше $p^r(t)$. Мета задачі полягає в мінімізації загальних витрат за плановий період і мінімізації переключень станів блоків. Крім того, необхідно врахувати певну регулярність (циклічність) режимів роботи енергоблоків. Це пов'язано з наступними факторами. Динаміка необхідної вихідної потужності в часі характеризується визначеною циклічністю (наприклад, по робочих днях тижня). Природно, щоб режими роботи енергоблоків також мали такі ж характеристики циклічності. Зазначені обставини (а також багато інших) показують наявність факторів даної задачі, які важко формалізувати (це характерно для будь-якої реальної задачі планування керуванням складних технічних систем). Адекватне відображення таких факторів формальною моделлю задачі математичного програмування прак-

тично неможливо. Але навіть, якщо така модель побудована (зрозуміло, що вона буде дискретного типу), то труднощі її чисельної реалізації неминуче призведуть до наближеного алгоритму евристичного характеру. Нашою метою є побудова алгоритму, що забезпечує прийнятний розв'язок задачі з урахуванням зазначених необхідних його характеристик.

Для наочності виклад алгоритму ілюструється рисунком 7, на якому необхідна по інтервалах планового періоду потужність енергосистеми представлена графіком кусочно-постійної функції p_t^r. (тонкі лінії). Помітимо, що загальний необхідний обсяг електроенергії за плановий період визначається площею, обмеженою графіком цієї функції і віссю t (підграфіком функції).

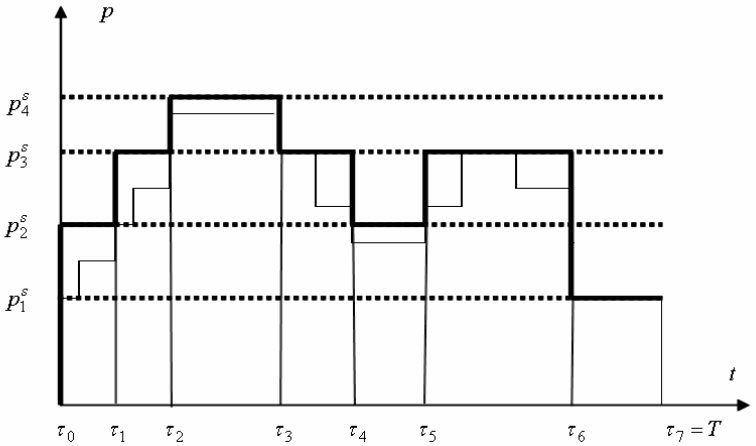


Рисунок 7. Графік необхідної потужності енергосистеми у часі

Будемо вважати, що задано набір «стандартних» рівнів потужностей, який забезпечує енергосистема: $p_k^s, k = 1,\ldots,K$, (точніше, тільки ці рівні і будуть враховуватися при розв'язанні задачі). Стандартні рівні потужностей упорядковані за зростанням: $p_{k+1}^s > p_k^s$.

Визначимо кусочно-постійну функцію $\tilde{p}^r(t)$, яка приймає значення з набору стандартних рівнів $\{p_k^s\}$, апроксимує функцію $p^r(t)$ зверху (тобто $\tilde{p}^r(t) \geq p^r(t)$) і має найменшу площу підграфіка. Алгоритм визначення цієї функції досить простий і зрозумілий з рисунка 7. Графік функції $\tilde{p}^r(t)$ позначений жирною лінією. На рисунку 7 представлені чотири стандартні рівні вихідної потужності енергосистеми (позначені пунктирними лініями). Функція $\tilde{p}^r(t)$ має сім часових інтервалів $[\tau_i, \tau_{i+1}], i = 0,\ldots 6,$ на яких вона приймає постійні значення (наприклад, на інтервалі $[\tau_0, \tau_1]$ $\tilde{p}^r(t) = p_2^s$).

Зупинимося на задачі вибору «стандартних» рівнів потужностей, яка є окремою (порівняно нескладною) задачею, що має ясну фізичну інтерпретацію. Відзна-

чимо основні вимоги до вибору «стандартних» рівнів. По-перше, алгоритм буде використовувати функцію $\tilde{p}^r(t)$ як вимогу по вихідній потужності енергосистеми (замість функції $p^r(t)$). Тому функція $\tilde{p}^r(t)$ повинна забезпечувати прийнятну апроксимацію функції попиту $p^r(t)$. По-друге, визначені графіком $\tilde{p}^r(t)$ моменти часу $\tau_i (i = 1,\ldots, I)$ будуть моментами можливого включення/вимикання енергоблоків. Як наслідок, тривалість інтервалів $[\tau_i, \tau_{i+1}]$ повинна бути достатньою для включення (кожного) енергоблоку, що знаходиться в цьому інтервалі в пасивному стані (зрозуміло, апостеріорі, ця вимога істотна лише для енергоблоків, що включаються у наступному інтервалі $[\tau_{i+1}, \tau_{i+2}]$).

Основна ідея «регуляризації» розв'язання задачі полягає в наступних вимогах (зазначений принцип обмеження множини допустимих розв'язків задачі був запропонований Трубіним В.О.):

1. Потужність станції на даному інтервалі залежить лише від стандартного рівня потужності енергосистеми цього інтервалу (незалежно від часового параметра інтервалу);

2. При переході на більш високий стандартний рівень енергосистеми потужність станції може лише зрости.

Звідси випливає, що розв'язання задачі полягає у визначенні величин x_{jk} (потужність станції j для стандартного рівня потужності k).

Нехай Δt_k – загальний час роботи енергосистеми на рівні k. Тоді потужності x_{jk} визначаються в результаті розв'язання такої задачі математичного програмування:

$$\min \sum_{k=1}^{K} \sum_{j=1}^{n} f_j(x_{jk}) \Delta t_k x_{jk} \tag{4.1}$$

$$x_{jk+1} \geq x_{jk}, j = 1,\ldots, n; k = 1,\ldots, K-1 \tag{4.2}$$

$$\sum_{j=1}^{n} x_{jk} = p_k^s, k = 1,\ldots.K; \tag{4.3}$$

$$0 \leq x_{jk} \leq P_j, j = 1,\ldots, n. \tag{4.4}$$

Цільова функція (4.1) дорівнює загальним витратам за плановий період. Обмеження (4.2) відповідають вимозі 2. Умови (4.3) означають, що сумарна потужність станцій для рівня k дорівнює потужності рівня. Обмеження (4.4) визначають діапазон зміни потужностей станцій.

Розв'язок x_{jk} задачі (4.1)–(4.4) і функція $\tilde{p}^r(t)$ однозначно визначають значення потужностей станцій для кожного інтервалу планового періоду.

Відзначимо, що в роботі [21] наведена еквівалентна модель задачі.

Зрозуміло, що при розв'язанні задачі вибору режимів енергосистеми на короткостроковий період (в якій стан енергоблоків фіксований) завантаження енергоблоків буде уточнюватися з детальнішим урахуванням динаміки вихідної потужності по часових інтервалах [22], [23].

4.2. Алгоритм розв'язання та програмне забезпечення

Математична модель (4.1)–(4.4) є нелінійною задачею математичного програмування. Для її розв'язання використовується *r*-алгоритм [10] – ефективний алгоритм негладкої оптимізації і негладкі штрафні функції.

Задача (4.1)–(4.4) зводиться до такої задачі з найпростішими обмеженнями на змінні:

$$\min\left(\sum_{k=1}^{K}\sum_{j=1}^{n} f_j(x_{jk})\Delta t_k x_{jk} + R_1 \max\{0, \max\{x_{jk+1} - x_{jk}, j = 1,\dots,n; k = 1,\dots,K-1\}\} + \right.$$

$$\left. R_2 \max\{|\sum_{j=1}^{n} x_{jk} - p_k^s|, k = 1,\dots,K\}\right), \tag{4.5}$$

$$0 \le x_{jk} \le P_j, j = 1,\dots,n, \tag{4.6}$$

де R_1, R_2 – штрафні коефіцієнти для обмежень (4.2) та (4.3) відповідно.

При розв'язанні задачі (4.5), (4.6) обмеження (4.6) враховуються спеціальним чином (шляхом заміни змінних). Розв'язання задачі здійснюється розробленим програмним забезпеченням для загальної задачі математичного програмування на основі використання *r*-алгоритму. Це програмне забезпеченням реалізується класом **CRalg_Math_Progr**. Програмне забезпечення математичної моделі розроблено на мові C++ у стилі об'єктно-орієнтованого програмування. Усі змістовні об'єкти задачі відображено відповідними класами C++. Алгоритм розв'язання і сервісні функції забезпечуються головним класом **CPowerOperation**.

Вхідні дані програмного забезпечення:

- плановий період (кількість інтервалів планового періоду, їхня тривалість);
- вимоги до вихідної потужності по інтервалах планового періоду;
- характеристики енергоблоків системи (максимальна потужність, функція питомих витрат);
- необхідна точність розв'язку.

Вихідні дані програмного забезпечення:

- потужності енергоблоків по інтервалах планового періоду;
- ступінь забезпечення вимог щодо вихідної потужності енергосистеми по інтервалах планового періоду.

Переходимо до короткого опису програмного забезпечення.

А) Класи програмного забезпечення

Повна документація програмного забезпечення задається файлами формату HTML, що генеруються системою Doxygen на основі файлів програмного забезпечення. Нижче наводиться короткий опис основних класів програмного забезпечення.

А1) CIntervalPlan

Призначення. Реалізація концепції "інтервал планового періоду".

Конструктор.

CIntervalPlan (int *id*, double *dRequiredPower*)

Аргументи:

id	код
dRequiredPower	необхідна потужність

Дані класу

double dRealizedPower	реалізована потужність
double dDeficitPower	дефіцит потужності (dDeficitPower = dRequiredPower - dRealizedPower)
int idLevel	код стандартного рівня потужності

A2) CStation

Призначення. Реалізація концепції "станція (енергоблок)"

Конструктор.

CStation (std::string *name*, double *dP_Max*)

Аргументи:

name	Ім'я
d_Max	максимальна потужність

Методи:

virtual double GetCost (double *p*) [virtual]
обчислення питомої вартості

Аргументи:

p	потужність

A3) CLevel

Призначення. Реалізація концепції "стандартний рівень потужності енергосистеми"

Конструктор.

CLevel (int *id*, double *dPower*)

Аргументи:

id	Код (ключ) рівня
dPower	Потужність рівня

Дані класу

double dTime	Загальний час роботи енергосистеми на даному рівні потужності

std::vector<CInterval_Funct<int,double>*> vpInterval	Вектор інтервалів роботи енергосистеми на даному рівні потужності

A4) CStation_Level

Призначення. Інформація для об'єктів {CStation&&CLevel}

Дані класу

CLevel* pLevel	Рівень потужності
CStation* pStation	станція
double dPower	потужність станції (розв'язок задачі)

A5) CIntervStation

Призначення. Інформація для об'єктів {CIntervalPlan&&CStation}

Дані класу

int id	код
int idIntervalPlan	код інтервалу планового періоду
int idStation	код станції
double dPower	потужність станції (розв'язок задачі)
double dCostFactor	витрати (розв'язок задачі)

A6) CPowerOperation

Призначення. Реалізація концепції "задача оптимального завантаження енергосистеми"

Конструктори.

CPowerOperation (
std::vector< CPoint< int, double > * >*vpDEMAND*,
std::vector< CStation * >*vpSTATION*,
std::vector< CLevel * >*vpCLevel*, int *iDeltaInterval*,
FILE **pFileMessage =0*
)

Аргументи:

vpDEMAND	вектор необхідної вихідної потужності по інтервалах планового періоду
vpSTATION	вектор станцій (енергоблоків)
vpCLevel	вектор стандартних рівнів
iDeltaInterval	мінімальна тривалість переходу на новий режим потужності

pFileMessage	файл повідомлень. Якщо pFileMessage == 0, то повідомлення не виводяться

CPowerOperation (
FILE **pFile_DEMAND*,
FILE **pFile_STATION*,
FILE **pFile_STATION_POWER_COST*,
FILE **pFile_PowerStandardLevel*,
FILE **pFile_DeltaInterval*,
FILE **pFileMessage =0*
)

Аргументи:

PFile_DEMAND	Необхідна вихідна потужність по інтервалах планового періоду
PFile_STATION	дані по станціях (енергоблокам)
PFile_STATION_POWER_COST	кусочно-лінійні функції питомих витрат енергоблоків
PFile_PowerStandardLevel	стандартні рівні вихідної потужності
PFile_DeltaInterval	мінімальна тривалість переходу на новий режим потужності
pFileMessage	файл повідомлень. Якщо pFileMessage == 0, то повідомлення не виводяться

Формати задання даних у файлах.

pFile_DEMAND:	**id_Time** (int)	**dVolume** (double)		
Pfile_STATION:	name_Station (string)	**id_Station** (int)	**dP_Max_Station** (double)	
pFile_STATION_POWER_COST:	**id_Staition** (int)	**id_Data** (int)	**dPower** (double)	**dCost** (double)
pFile_PowerStandardLevel:	**id_Leve** (int)	**dPower** (double)		

Методи.

int GetSolution (int *kIterMax* = 1000)

Аргументи:

kIterMax	Максимальне число ітерацій алгоритму
Return	0 – задача розв’язана;

	1 – ітерацій не досить для розв’язання задачі

std::vector< CIntervStation const*> Get_vpInterv_Station ()

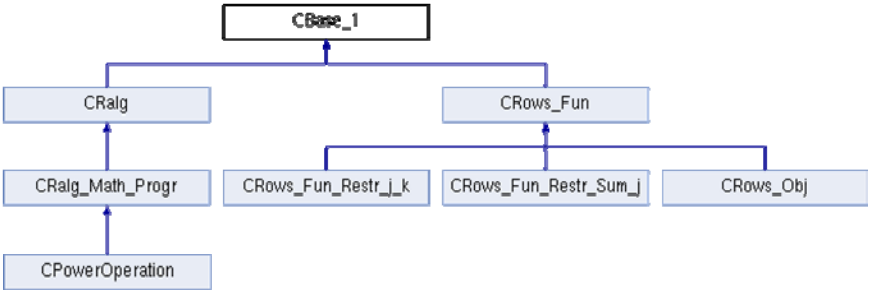
Доступ до privat члена класу, що містить інформацію про розв’язок задачі.
return: копія вектора покажчиків на **CIntervStation**

A7) Службові класи

Програмне забезпечення використовує наступні службові класи.

CBase_1	Службовий базовий клас. Містить інформацію загального характеру (ім'я, файл повідомлень, генерація виключень і інше)
Cvariable_N	Реалізація концепції "число double у математичних моделях"
CVariable_PowerOperation	Реалізація концепції "змінна математичної моделі **CPowerOperation**"
CPoint< X, Y >	Шаблон класу. Реалізація концепції "точка графіка функції"
Клас CPiece_Linear_Function	Реалізація концепції кусочно-лінійної функції
CPiece_Const_Function< X, Y >.	Шаблон класу. Реалізація концепції кусочно-постійної функції. Значення функції на інтервалі визначається лівою точкою інтервалу.
CRows_Fun	Реалізація концепцій "цільова функція" і "обмеження" задачі математичного програмування
CRows_Fun_Restr_j_k	Реалізація обмеження (4.2)
CRows_Fun_Restr_Sum_j	Реалізація обмеження (4.3)
CRows_Obj	Реалізація цільової функції математичної моделі
CRalg	Розв’язання задачі безумовної мінімізації (реалізація *r*-алгоритму)
CRalg_Math_Progr	Розв’язання загальної задачі математичного програмування *r*-алгоритмом з використанням негладких штрафних функцій

В) Граф співвідношень класів



5. МОДЕЛЬ КОРОТКОСТРОКОВОГО УПРАВЛІННЯ ЕНЕРГЕТИЧНОЮ СИСТЕМОЮ

5.1. Вхідні дані моделі та позначення

Основна особливість задачі короткострокового управління енергетичною системою полягає в наступному. Плановим періодом є доба. Тривалість інтервалів планового періоду відповідає одній годині. Для теплових електростанції задані діапазони вихідної потужності для кожного інтервалу планового періоду. Для кожної гідроелектростанції задається величина сумарної вихідної потужності для всього планового періоду.

Плановий період. T – кількість часових інтервалів, на які розбивається плановий період (t – індекс перерахунку часових інтервалів; $t = 1,\ldots,T$). Як правило, $T = 24$. P_t – прогнозована величина потрібної в интервалі t загальної вихідної потужності енергосистеми ($t = 1,\ldots,T$).

Теплові електростанції. N – кількість теплових електростанцій енергосистеми (j – індекс перерахунку теплових електростанцій; $j = 1,\ldots,N$); $p_{jt}^1 (p_{jt}^2)$ – мінімальна (максимальна) потужність електростанції j в інтервалі t; x_{jt}^T – оптимальна вихідна потужність електростанції j в інтервалі t ($j = 1,\ldots,N$; $t = 1,\ldots,T$), яка підлягає визначенню.

Змінні x_{jt}^T повинні відповідати наступним обмеженням:

$$p_{jt}^1 \le x_{jt}^T \le p_{jt}^2; j = 1,\ldots N; t = 1,\ldots T. \qquad (5.1)$$

Гідроелектростанції. M – кількість гідроелектростанцій; (i – індекс перерахунку теплових електростанцій; $i = 1,\ldots,M$;); $P_i^1 (P_i^2)$ – мінімальна (максимальна) сумарна потужність електростанції i для планового періоду; x_{it}^w – оптимальна вихідна потужність електростанції i в інтервалі t ($i = 1,\ldots,M$; $t = 1,\ldots,T$), яка підлягає визначенню.

Для кожної електростанції введемо так звані «штрафні» змінні w_i^+, w_i^- відносно умови виконання обмеження на величину сумарної потужності електростанції:

$$P_i^1 \le \sum_{t=1}^{T} x_{it}^w \le P_i^2; i = 1,...,M$$

Змінні x_{it}^w, w_i^+, w_i^- повинні відповідати таким обмеженням:

$$P_i^1 \le \sum_{t=1}^{T} x_{it}^w + w_i^+ - w_i^- \le P_i^2; i = 1,\dots,M; \tag{5.2}$$

$$x_{it}^w \ge 0;\ i = 1,\dots,M;\ t = 1,\dots,T; \tag{5.3}$$

$$w_i^+ \ge 0; w_i^- \ge 0;\ i = 1,\dots,M. \tag{5.4}$$

Вклад змінних w_{it}^+, w_{it}^- в цільову функцію буде визначатися виразом $R_i^w w_i^+ + R_i^w w_i^-$, де $R_i^w > 0 -$ величина штрафного коефіцієнта. Звідси випливає, що для оптимальних значень змінних справедливі співвідношення

$$w_i^+ = \max\{0, P_i^1 - \sum_{t=1}^{T} x_{ii}^w\};\ \ w_i^- = \max\{0, \sum_{t=1}^{T} x_{ii}^w - P_i^2\}.$$

Ці співвідношення визначають змістовний сенс штрафних змінних. Змінна $w_i^+ (w_i^-)$ «відповідальна» за порушення нижньої (верхньої) границі на величину сумарної вихідної потужності станції. Якщо границя не порушена, то значення змінної дорівнює нулю. Таким чином, якщо існує розв'язок задачі, що задовольняє обмеженням на діапазон сумарних вихідних потужностей станцій, то всі штрафні змінні будуть рівні нулю.

5.2. Математична модель

Математична модель представляється наступною задачею математичного програмування

$$\min \leftarrow \sum_{t,j=1}^{T,N} c^T(x_{jt}^T) + \sum_{t,i=1}^{T,M} c^w(x_{it}^w) + \sum_{i=1}^{M} (R_i^w w_i^+ + R_i^w w_i^-) \tag{5.5}$$

$$\sum_{j=1}^{N} x_{jt}^T + \sum_{i=1}^{M} x_{it}^w = P_t\ ;\ t = 1,\dots,T; \tag{5.6}$$

$$P_i^1 \le \sum_{t=1}^{T} x_{it}^w + w_i^+ - w_i^- \le P_i^2;\ i = 1,\dots,M; \tag{5.7}$$

$$x_{it}^w \ge 0;\ i = 1,\dots,M;\ t = 1,\dots,T; \tag{5.8}$$

$$w_i^+ \ge 0; w_i^- \ge 0;\ i = 1,\dots,M; \tag{5.9}$$

$$p_{jt}^1 \le x_{jt}^T \le p_{jt}^2; j = 1,\dots N; t = 1,\dots T, \tag{5.10}$$

де $c^T(x_{jt}^T)(c^w(x_{it}^w)) -$ опукла кусочно-лінійна функція, що визначає вартість генерації потужності p теплової станцією j (гідроелектростанцією i).

Обмеження (5.6) відповідають вимозі рівності сумарної вихідної потужності всіх електростанцій в інтервалі t прогнозованому об'єму споживання. (Обмеження (5.7) описані вище).

5.3. Метод розв'язання задачі

Метод розв'язання задачі (5.5)–(5.10) базується на розв'язку двоїстої задачі відносно обмежень (5.6), (5.7). Відповідні цим обмеженням множники Лагранжа позначимо u_t^T, u_i^w.

Двоїста задача полягає в наступному:

$$\max \psi(u), \quad u \geq 0 \tag{5.11}$$

де $u = \{u_t^T, u_i^w\}$.

Двоїста функція $\psi(u)$ визначається розв'язком задачі мінімізації функції Лагранжа:

$$\psi(u) = \min\{\sum_{t=1}^{T}(\sum_{j=1}^{N}(c^T(x_{jt}^T) + u_t^T x_{jt}^T) +$$

$$+\sum_{i=1}^{M}(c^w(x_{it}^w) + (u_t^T + u_t^w)x_{it}^w)) +$$

$$+\sum_{i-1}^{N}(w_i^+ - w_i^-)u_i^w\} -$$

$$-\sum_{i-1}^{N}\max\{P_i^2 u_i^w, -P_i^1 u_i^w\}. \tag{5.12}$$

Мінімізація в (5.12) здійснюється за змінними $x_{jt}^T, x_{it}^w, w_i^+, w_i^-$. Для розв'язання двоїстої задачі (5.11) використовується *r*-алгоритм [10]. Розроблена програмна реалізація (на мові С++) описаного методу була випробувана для розв'язання задач оперативного управління енергетичною системою (використовувались запропоновані «Київенерго» вихідні дані задачі). Розмірність задачі характеризується наступними значеннями параметрів: $T = 24, N = 15, M = 4$. Час розв'язання задачі на ПК класу Pentium 3 складає близько 1.5 хв.

6. ЗАДАЧА ТА МЕТОД РОЗРАХУНКУ ТЕПЛОВИХ СХЕМ

6.1. Математична модель

Основою для проведення чисельних розрахунків, що виконуються при проектуванні енергетичних парових котлів, є розрахунок теплової схеми парового котла [24, 25]. Існуючі методики теплового розрахунку достатньо трудомісткі. У зв'язку з цим виникає необхідність розробки швидких методів розрахунку теплових схем. Для відображення суті запропонованого методу розрахунку теплова схема теплоенергетичного об'єкта буде представлятися у такому спрощеному вигляді. Маємо два середовища. Одне з них є активним, друге – пасивним. У теплоенергетичному пристрої відбувається передача теплової енергії від активного середовища пасивному. Для змістовної інтерпретації будемо вважати, що активне середовище – це газ (продукти згорання у топці парового котла), а пасивне середовище – водяна пара парового тракту котла. Газ рухається в газоході котла (газовий тракт парового котла). Водяна пара циркулює в трубопроводі (паровий тракт котла). Для простоти викладу будемо вважати, що теплова схема містить один газовий тракт і один паровий тракт. Основний елемент теплової схеми – теплообмінник: спеціальний технічний пристрій, в якому виконується передача теплової енергії газу водяній парі. Він розміщується у газовому тракті котла. Теплообмінник будемо уявляти у вигляді деякого перетворювача («чорний ящик»), який має $m+1$ входів та $m+1$ виходів. Один із входів (один із виходів) визначається фізичними параметрами потоку газу, інші m – пару. Важливими фізичними параметрами теплотехнічних розрахунків є ентальпія, температура, тиск, масові витрати. Будемо вважати, що всі параметри, окрім ентальпії та температури, задані. Відзначимо, що тоді температура однозначно визначається ентальпією, і навпаки. Тому для визначеності подальший виклад буде виконуватися у термінах температури. $T_1^0, T_2^0\,(T_1^j, T_2^j)$ – температури газу (пару, $j=1,\ldots,m$) на вході і виході теплообмінника відповідно. Індекс 0 відноситься до характеристики газу (активний елемент), індекс j – до характеристики пари (пасивні елементи). Нехай вхідні параметри T_1^0, T_1^j теплообмінника задані. Тоді розрахунок вихідних температур T_2^0, T_2^j – достатньо складна задача математичної фізики. У теплотехніці для такого розрахунку розроблено спеціальні методики для знаходження наближеного розв'язку цієї задачі [24]. Алгоритм такого розрахунку зводиться до розв'язання системи нелінійних рівнянь з невеликим числом змінних. Однак, навіть такий наближений алгоритм розв'язання задачі теплотехнічного розрахунку теплообмінника виявляється достатньо трудомістким для розрахунку теплової схеми в цілому, яка містить не один, а десятки теплообмінників. Ситуація суттєво ускладнюється при розв'язанні задач оптимізації теплової схеми. У таких

задачах необхідно виконувати багаторазовий (десятки тисяч разів) розрахунок теплової схеми. Нижче пропонується алгоритм розрахунку теплової схеми, який базується на ідеї введення модельного теплообмінника [7].

Модельний теплообмінник вводиться таким чином. Для широкого класу (наскільки нам відомо, для всіх) теплообмінників, які використовуються в теплоенергетичних пристроях, виконуються такі умови:

$$T_2^0 \in [T_1^0, \min\{T_1^j, j = 1, m\}], \quad T_2^j \in [T_1^0, T_1^j]. \tag{6.1}$$

Тут і надалі припускається, що $T_1^0 \geq T_1^j$. Відзначимо, що у ситуації передачі теплової енергії без використання перетворення її в інші види енергії, виконання цих умов очевидне з термодинамічних міркувань. Для побудови модельного теплообмінника корисно прийняти наступну інтерпретацію (зрозуміло, що штучну) процесу теплообміну.

1. Пасивний елемент взаємодіє ("отримує тепло") з λ_j -ою частиною потоку активного елемента.
2. Тепловий обмін λ_j -ої частини активного елемента і пасивного елемента потоку j відбувається до їхньої теплової рівноваги (тобто на виході їхні температури рівні).
3. Вихідний потік активного елемента є сумішшю λ_j -их частин потоку, які беруть участь у взаємодії з пасивними елементами, та $(1 - \sum_{j=1}^{m} \lambda_j)$ -ої частини, яка не бере участь у процесі теплового обміну.

Нехай задано значення ("орієнтовні значення") вхідних температур $\tilde{T}_1^j, j = \overline{0,n}.$ Для цих температур в результаті теплового розрахунку отримано значення вихідних температур $\tilde{T}_2^j, j = \overline{0,n}.$ Для стислості модель теплообмінника, яка відповідає тепловому розрахунку, будемо називати "фізичним" теплообмінником. Наша мета полягає у побудові модельного теплообмінника на основі розрахунку фізичного теплообмінника. Будемо говорити, що такий модельний теплообмінник відповідає орієнтовним температурам $\tilde{T}_1^j$. У ньому температури на виході визначаються таким чином:

$$T_2^j = \sum_{k=0}^{m} \mu_{jk} T_1^j, \; j = \overline{0,m}. \tag{6.2}$$

$$\sum_{k=0}^{m} \mu_{jk} = 1; \; \mu_{jk} \geq 0, j = \overline{0,m}. \tag{6.3}$$

Теплоємність маси середовища потоку, яка проходить через його переріз за одиницю часу (як газу, так і пару) $c_j(T)$, взагалі кажучи, залежить від її температури T. Введемо поняття "ефективної" теплоємності $\tilde{c}_j$, що не залежить від температури середовища. Зміст ефективних теплоємностей та алгоритм їх визначення будуть уточнені нижче. Нехай модель (6.2), (6.3) задовольняє таким вимогам:

$$\tilde{T}_2^j = \sum_{k=0}^{m} \mu_{jk} \tilde{T}_1^j, \; j = \overline{0,m}; \tag{6.4}$$

$$\sum_{k=0}^{m} c_j(\tilde{T}_2^j - \tilde{T}_1^j) = 0. \quad (6.5)$$

Умова (6.4) відповідає вимозі рівності вихідних температур модельного та фізичного теплообмінників для вхідних температур $\tilde{T}_1^j$. Обмеження (6.5) відображає закон збереження теплової енергії: значення $c_j(\tilde{T}_2^j - \tilde{T}_1^j)$ є зміною величини ентальпії потоку j. Рівноважна температура T_2^{*j} λ_j -ої частини потоку активного елемента і потоку пасивного елемента j визначається, очевидно, таким чином (у відповідності із законом збереження теплової енергії):

$$T_2^j = T^{*j} = (\lambda_j c_0 T_1^0 + c_j T_1^j) / (\lambda_j c_0 + c_j),\ j = \overline{1,m}. \quad (6.6)$$

З (6.2), (6.6) випливає (для $j = \overline{1,m}$):

$$\mu_{jk} = 0, k \neq 0, j;\ \mu_{j0} = \lambda_j c_0 / (\lambda_j c_0 + c_j);\ \mu_{jj} = c_j / (\lambda_j c_0 + c_j). \quad (6.7)$$

Неважко бачити, що з (6.2), (6.4) та (6.7) маємо (для $j = \overline{1,m}$):

$$T_2^j = (1 - \tilde{\mu}_j) T_1^j + \tilde{\mu}_j T_0^j, \quad (6.8)$$

де

$$\tilde{\mu}_j = (\tilde{T}_2^j - \tilde{T}_1^j) / (\tilde{T}_1^0 - \tilde{T}_1^j) = \tilde{\mu}_j = c_j / (\lambda_j c_0 + c_j). \quad (6.9)$$

(У (6.9) припускається, що $\tilde{T}_1^0 - \tilde{T}_1^j \neq 0$. Якщо $\tilde{T}_1^0 - \tilde{T}_1^j = 0$, то $\tilde{\mu}_j$ однозначно не визначено. У цьому випадку значення $\tilde{\mu}_j$ може бути довільним числом з інтервалу $[0,1]$. Для визначеності приймемо $\tilde{\mu}_j = 1$).

З (6.3) випливає виконання умови $\tilde{\mu}_j \in [0,1]$. Це призводить до наступних вимог до фізичного теплообмінника:

$$\tilde{T}_1^0 > \tilde{T}_1^j \Rightarrow \tilde{T}_2^j > \tilde{T}_1^j; \tilde{T}_2^j < \tilde{T}_1^0; \quad (6.10)$$

$$\tilde{T}_1^0 < \tilde{T}_1^j \Rightarrow \tilde{T}_2^j < \tilde{T}_1^j; \tilde{T}_2^j > \tilde{T}_1^0; \quad (6.11)$$

$$\tilde{T}_1^0 = \tilde{T}_1^j \Rightarrow \tilde{T}_2^j = \tilde{T}_2^0 = \tilde{T}_1^0. \quad (6.12)$$

Умови (6.10)–(6.12) фізично коректні – вони дійсно повинні виконуватися для реального теплообмінника. Крім виконання умов (6.10)–(6.12), будемо вважати, що для фізичного теплообмінника виконується збереження теплової енергії:

$$\tilde{J}_1^0 - \tilde{J}_2^0 = \sum_{j=1}^{m} \tilde{J}_2^j - \tilde{J}_1^j, \quad (6.13)$$

де $\tilde{J}_1^j, \tilde{J}_2^j -$ ентальпії потоків на вході та виході теплообмінника. Введені вище "ефективні" теплоємності визначаються за вхідними та вихідними значеннями ентальпій і температур фізичного теплообмінника за формулами:

$$c_j(\tilde{T}_2^j - \tilde{T}_1^j) = \tilde{J}_2^j - \tilde{J}_1^j,\ j = \overline{0,m}. \quad (6.14)$$

Коефіцієнти μ_{0j} модельного теплообмінника (6.2), (6.3) визначаємо на основі балансу теплової енергії для модельного теплообмінника (6.5). З (6.5) та (6.8) отримуємо: $\sum_{j=1}^{m} c_j(T_2^j - T_1^j) = \sum_{j=0}^{m} c_j\tilde{\mu}_j(T_1^0 - T_2^0) = c_0(T_1^0 - T_2^0)$. Звідси

$$T_2^0 = (1-(1/c_0)\sum_{j=1}^{m} c_j\tilde{\mu}_j)T_1^0 + (\sum_{j=0}^{m}(c_j\tilde{\mu}_j / c_0)T_1^j). \qquad (6.15)$$

Порівнюючи (6.15) і (6.2), маємо: $\mu_{00} = (1-(1/c_0)\sum_{j=1}^{m} c_j\tilde{\mu}_j)$; $\mu_{0j} = c_j\tilde{\mu}_j / c_0$.

Побудова модельного теплообмінника закінчена: всі його параметри визначено. Суттєвим фактором при його побудові є задання значень вхідних температур $\tilde{T}_1^0, \tilde{T}_1^j$ ("орієнтовні температури"). Параметри модельного теплообмінника однозначно визначаються результатом теплового розрахунку фізичного теплообмінника. Модельний теплообмінник є лінійним перетворювачем. Для його вхідних температур $\tilde{T}_1^0, \tilde{T}_1^j$ вихідні температури точно відповідають їхнім значенням теплотехнічного розрахунку. Звичайно, при інших значеннях вхідних температур такої відповідності вже не буде – модельний теплообмінник буде давати деяку похибку. Однак, чисельні дослідження величини цієї похибки показали, що такий теплообмінник забезпечує відносну похибку не більше 10% для достатньо широкого діапазону значень вхідних температур.

6.2. Алгоритм розв'язання

Метод теплового розрахунку на основі використання модельних теплообмінників полягає в наступному. Нехай задано орієнтовні значення вхідних температур для усіх теплообмінників. Генеруємо орієнтовані на ці температури модельні теплообмінники. Розрахунок теплової схеми з модельними теплообмінниками зводиться до розв'язання системи лінійних рівнянь. Розв'язок цієї системи визначає розрахункові значення вхідних і вихідних температур, які відповідають обраним орієнтовним значенням. Якщо розрахункові значення з достатньою точністю дорівнюють їх орієнтовним значенням, то розрахунок схеми виконано. У протилежному випадку виконуємо корекцію орієнтовних значень. Нові значення визначаються, наприклад, як середнє арифметичне розрахункових і попередніх орієнтовних значень. Після цього ітеративно виконуємо описану процедуру до отримання заданої точності розв'язку задачі.

Звичайно, описаний метод теплового розрахунку можна розглядати як метод лінеаризації для розв'язання системи нелінійних рівнянь. Однак відзначимо, що формальне використання методу лінеаризації функціоналів системи (на основі їх диференціалів) не завжди приводить до успіху. Справа у тому, що отримані таким чином модельні теплообмінники, взагалі кажучи, не забезпечують виконання умови (6.1). Це пов'язано з тим, що система рівнянь, на основі яких виконується розрахунок вихідних температур у теплотехніці, хоча й забезпечує обчислення вихідних температур з прийнятною точністю, але лише наближено відображає фізичний процес теплообміну.

На основі використання модельних теплообмінників розроблено не тільки алгоритм розрахунку теплових схем парових котлів, але й алгоритм обчислення похідних для вихідних температур теплообмінника за різними його конструктивними параметрами (обчислення похідних зводиться до розв'язання систем лінійних рівнянь). На основі описаного алгоритму розроблено програмне забезпечення задачі розрахунку достатньо складних теплових схем парових котлів. Застосування алгоритму для розв'язання практичних задач показало його достатньо високу ефективність: для забезпечення відносної точності розв'язку $\gamma \approx 0.01$ необхідно ≈ 15 ітерацій алгоритму.

6.3. Програмне забезпечення

Програмне забезпечення математичної моделі розроблено на мові C++ у стилі об'єктно-орієнтованого програмування. Всі змістовні об'єкти задачі відображені відповідними класами C++. Алгоритм розв'язання і сервісні функції забезпечуються головним класом CScheme_N.

Програмне забезпечення містить абстрактні класи. Наприклад, клас CHeatUnit_N містить чисто віртуальну функцію розрахунку вихідних температур фізичного теплообмінника (int CHeatUnit_N::CalcOut()=0;). Вона залежить від конкретного теплообмінника і повинна визначатися користувачем шляхом механізму спадкування.

Вхідні дані програмного забезпечення визначають інформацію про такі - об'єкти:

- граф зв'язків елементів теплової схеми;
- про всі елементи схеми (тракти, труби, теплообмінники);
- точність розв'язку, яка вимагається.

Вихідні дані програмного забезпечення:

- параметри всіх потоків теплової схеми: масові витрати, температура;
- користувач може використовувати сервісні функції, що надаються програмним забезпеченням, для розрахунку різних характеристик розв’язку задач (наприклад, похідних вихідних даних за вхідними параметрами).

Переходимо до короткого опису програмного забезпечення.

А) Класи програмного забезпечення

Повна документація програмного забезпечення задається файлами формату HTML, що генерується системою Doxygen на основі файлів програмного забезпечення. Нижче наводиться короткий опис основних класів програмного забезпечення.

A1) CElement_N

Призначення. Реалізація концепції "елемент теплової схеми".

Конструктор(и).
CElement_N (string *name*, const void **pUser*, const int *nTubeInp*, const int *nTubeOut*, const CParamElement_N ***pParam* = 0, const int *nParam* = 0) throw ()

Аргументи:

name	ім'я
pUser	вказівник користувача
nTubeInp	число вхідних клем
nTubeOut	число вихідних клем
pParam	параметри
nParam	кількість параметрів

A2) CTube_N

Призначення. Реалізація концепції "труба".

Конструктор(и).
CTube_N (string *name*, const CElement_N **pElementOut*, const int *iTubeOutIndex*, const CElement_N* *pElementInp*, const int *iTubeInpIndex*, CStream_N **pStream*)

Аргументи:

pElementOut	елемент початку труби
iTubeOutIndex	індекс (номер) вихідної труби елемента, який відповідає даній трубі
pElementInp	кінцевий елемент труби
iTubeInpIndex	індекс (номер) вхідної труби елемента, який відповідає даній трубі
pStream	потік

A3) CStream_N

Призначення. Реалізація поняття "потік". Містить фізичні дані матеріального потоку у трубах (маса, температура).

Конструктор(и).
CStream_N (void **pUser*, CValue_N **pMass*, CValue_N **pTemp*)

Аргументи:

pUser	вказівник користувача
PMass	маса
PTemp	температура

А4) CTrakt_N

Призначення. Реалізація концепції "тракт".

Конструктор(и).
CTrakt_N (string *name*, ***pElement*, int *nElement*, ***pTube*, int *nTube*)

Аргументи:

Name	ім'я
pElement	елементи
nElement	кількість елементів
pTube	труби
nTube	кількість труб

А5) CHeatExchange_N

Призначення. Реалізація концепції "елемент теплової схеми" з такими властивостями: вхідні і вихідні потоки (труби) належать тільки одному тракту. Не змінюють масові потоки; містяться в об'єкті CHeatUnit_N. Містяться у теплообміннику (об'єкт CHeatUnit_N). Вихідні температури потоків визначаються функцією CHeatUnit_N::CalcOut ().

Конструктор(и).
CHeatExchange_N (string *name*, const void **pUser* = 0, const CParamElement ***pParam* = 0, const int *nParam* = 0)

Аргументи:

name	ім'я
pUser	вказівник користувача
pParam	параметри
nParam	кількість параметрів

А6) CHeatUnit_N

Призначення. Реалізація концепції теплообмінника.
Вхідні і вихідні потоки (труби) можуть належати різним трактам. Не змінює масові потоки. Змінює температуру внаслідок теплового обміну.

Конструктор(и).
CHeatUnit_N (string *name*, ***pElement*, int *nElement*, void **pUser*)

Аргументи:

name	ім'я
pElement	елементи

nElement	кількість елементів
pUser	вказівник користувача

Методи:
virtual int CalcOut () [pure virtual]
розрахунок вихідних температур реального теплообмінника

A7) CScheme_N

Призначення. Реалізація концепції "задача розрахунку теплових схем" (алгоритм модельних теплообмінників)

Конструктор(и).
CScheme_N (FILE **pFileMessage*, FILE **pFileSolution*, int *iStdOut*, CTrakt_N ***pTrakt*, int *nTrakt*, CTube_N ***pTube*, int *nTube*, CHeatUnit_N ***pHeatUnit*, int *nHeatUnit*)

Аргументи:

pFileMessage	файл повідомлень
pFileSolution	файл розв'язку
iStdOut	виведення в cout
pTrakt	тракти схеми
nTrakt	кількість трактів схеми
pTube	труби схеми між різними трактами
nTube	кількість труб між різними трактами
pHeatUnit	теплообмінники схеми
nHeatUnit	кількість теплообмінників

Методи:

const int ContrTubeScheme () [protected]	контроль коректності даних труб
int ContrConnectElement () [protected]	контроль коректності даних з'єднання елементів
int GetCountMass_LinAgr () [protected]	обчислення кількості агрегатів масових потоків
int GenMass_LinAgr () [protected]	генерація лінійного агрегату розрахунку масових потоків
int GenTempr_LinAgr () [protected]	генерація лінійного агрегату розрахунку температур
FILE* Get_pFileMessage ()	вивід даних у файл pFileMessag.
void GetMassStream ()	розрахунок масових потоків
GetDiffMassStream (CValue_N **pDiffValue*)	обчислення похідних масових потоків по змінній pDiffValue

int GetTemprStream (double *dAccuracy*, int *iIterMax*, double *dStepEstimateTempr*, double &*AccuracyOut*, int &*iRetIter*)	розрахунок температур
int GetDiffMassStream (CValue_N **pDiffValue*)	обчислення похідних температур потоків по змінній pDiffValue
void Write_Tube (FILE *pFileOut)	вивід даних труб

Аргументи функції GetTemprStream:

dAccuracy	необхідна точність
IIterMax	максимальне число ітерацій
dStepEstimateTempr	параметр алгоритму
AccuracyOut	досягнута точність
iRetIter	число виконаних ітерацій

А8) Службові класи

Програмне забезпечення використовує такі службові класи.

CBase_1	Службовий базовий клас. Містить інформацію загального характеру
CValue_N	Реалізація концепції "число double у математичних моделях" (зміст членів класа зрозумілий з їхніх імен)
CLinAgregate_N	Реалізація концепції "блочна система лінійних рівнянь". Забезпечує генерацію, контроль, розв'язок і обчислення похідних розв'язку по вхідних параметрах системи
CNik_Except	Базовий клас виключень
CNik_Except_ERR_Data	Клас виключень, пов'язаних з помилкою у даних
CNik_Except_Memory	Клас виключень, пов'язаних з виділенням пам'яті

Розроблене програмне забезпечення дозволяє не тільки виконувати розрахунок теплових схем, але й реалізує алгоритми обчислювання похідних вихідних даних по всім параметрам теплової схеми. Тому програмне забезпечення може використовуватися у задачах оптимального вибору керуючих і конструктивних параметрів теплових схем.

Програмне забезпечення апробовано на задачах оптимального проектування окремих підсистем промислових парових котлів [7, 8].

ВИСНОВКИ

Наведені у роботі математичні моделі, методи та програмне забезпечення для задач планування навантажень в енергосистемі можуть бути використані при побудові диспетчерських графіків завантаження енергоблоків ТЕС, АЕС та гідроелектростанцій. Для електрогенеруючих компаній України це дозволить зменшити кількість включень-виключень енергоблоків та запобігти технологічно нераціональним планам навантаження енергоблоків. При цьому можна врахувати циклічність роботи енергоблоків та ефективно використовувати енергоблоки при "номінальній" потужності, при якій коефіцієнт корисної дії енергоблоку максимальний.

Однак центральними у роботі є не самі математичні моделі для задач планування навантажень енергосистеми (їх потрібно уточнювати для конкретних застосувань), а ті структурні особливості задач лінійного та нелінійного програмування, які відповідають цим математичним моделям. До таких особливостей слід віднести сепарабельність цільових функцій сумарних витрат умовного палива (представлені у вигляді суми функцій витрат умовного палива по всім енергоблокам, де кожна із функцій залежить тільки від потужності окремого енергоблоку) та блочну структуру матриці обмежень (характеризує прогнозовану потребу в потужності енергосистеми в кожний із інтервалів планового періоду). Саме ці структурні особливості задач лінійного та нелінійного програмування визначили два основні методи, які використано для знаходження оптимальних навантажень енергоблоків в енергосистемі. Ними є *r*-алгоритми Шора – прискорені за рахунок використання операції розтягу простору субградієнтні методи мінімізації негладких опуклих функцій та метод ПАВ, який узагальнює "принцип оптимальності" Белмана в динамічному програмуванні.

Застосування алгоритмів недиференційовної оптимізації в комбінації зі схемами декомпозиції і негладкими штрафними функціями дозволяє ефективно розв'язувати задачі лінійного програмування великої розмірності, нелінійні мінімаксні задачі, задачі нелінійного програмування загального типу. Використання схем декомпозиції для розв'язання блочних задач лінійного та нелінійного програмування призводить до задач мінімізації негладких функцій від порівняно невеликої кількості зв'язуючих змінних або множників Лагранжа для зв'язуючих обмежень. Центральну роль при розв'язанні таких негладких задач відіграють саме прискорені варіанти субградієнтних методів – *r*-алгоритми. *r*-алгоритми дозволяють врахувати блочну структуру матриці обмежень та дають можливість ефективно розв'язувати відповідні моделям планування навантажень енергосистеми задачі лінійного та нелінійного програмування.

Сепарабельність цільових функцій сумарних витрат умовного палива дозволяє у задачах планування навантажень енергосистеми використати узагальнений принцип оптимальності для монотонно-рекурсивних функцій – центральний мо-

мент методу ПАВ, що є досить широким узагальненням "принципу оптимальності" Белмана в динамічному програмуванні. Конструювання та відсіювання варіантів в методі ПАВ полягає в такому способі побудови варіантів і виборі операторів їх аналізу, при якому відсіваються безперспективні частини варіантів без їх повної побудови. Цим забезпечується значна економія в обчислювальній процедурі, причому вона тим істотніше, чим більше специфічних властивостей задачі використано при побудові операторів аналізу та відсіву.

Наряду з *r*-алгоритмами та методами ПАВ для розв'язання задач нелінійного програмування були використані також відомі програми KNITRO, LOQO, MINOS, SNOPT. Це обумовлено тим, що завдяки Інтернету в останні роки став досить популярним онлайновий режим роботи з такими програмами. Цей сервіс забезпечує NEOS-сервер. Як правило майже всі програми підтримують роботу з мовою AMPL. Тому одним із найпростіших способів розв'язання задач лінійного та нелінійного програмування можна вважати розробку AMPL-кодів для відповідних задач та розрахунки в діалоговому режимі на NEOS-сервері з використанням того чи іншого солвера (програми). Однак, це не означає що такий шлях – найкращий для розв'язання задач загрузки енергоблоків. Структурні особливості цих задач дозволяють розробляти ефективні спеціалізовані методи, які орієнтовані на розв'язання задач великих розмірів.

Результати досліджень вказаних методів для різних формулювань задач завантаження енергосистеми наведено у розділах 1, 2 та 3 (автори – Стецюк П.І. та Лиховид О.П.) та у розділах 4 і 5 (автор – Журбенко М.Г.). Перший розділ пов'язаний із програмами NEOS-сервера, другий – з методом ПАВ, третій, четвертий та п'ятий розділи – із різними аспектами використання *r*-алгоритмів. Один із цих аспектів пов'язаний з використанням модифікації *r*-алгоритму в багатоекстремальних задачах для побудови прискорених методів за рахунок паралельних обчислень в багатопроцесорних комплексах (розділ 3). Показано, що для таких задач можливо досягти лінійного прискорення в залежності від числа процесорів.

Окремо слід виділити розділ 6 (автор – Журбенко М.Г.), де розглянута задача розрахунку теплових схем. Програмне забезпечення призначене для розрахунку теплових схем та оптимізації їх параметрів. Програмне забезпечення реалізує оригінальний алгоритм, заснований на використанні введеного поняття модельного теплообмінника. Модельний теплообмінник локально апроксимує відповідний реальний теплообмінник в деякому діапазоні температур. Модельний теплообмінник, на відміну від фізичного, характеризується малою трудомісткістю розрахунку його вихідних параметрів. Програмне забезпечення може використовуватися при вирішенні задач управління та проектування теплових енергетичних агрегатів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Стецюк П.И., Лиховид А.П., Пилиповский А.В. Задачи оптимизации для выбора электрических нагрузок в энергосистеме // Теорія оптимальних рішень. – Київ: Ін-т кібернетики ім. В.М.Глушкова НАН УкраЇни, 2009. – №8. – С. 136–141.
2. Стецюк П.И., Лиховид А.П., Пилиповский А.В. О решении одного класса оптимизационных задач нахождения нагрузок энергетических объектов с помощью NEOS-программ // Праці міжнар. симп. "Питання оптимізації обчислень (ПОО-XXXV)". – Київ: Ін-т кібернетики ім. В.М.Глушкова НАН України, 2009. – Т.2. – С. 350–354.
3. Математические и программные средства моделирования и оптимизации динамической загрузки мощностей энергосистемы / П.И. Стецюк, А.П. Лиховид, Б.М. Чумаков, А.Ю. Видил, А.В. Пилиповский // Отчет о научно-исследовательской работе № гос. регистрации 0107U004963. – Київ: Ін-т кібернетики ім. В.М.Глушкова НАН України, 2009. – 136 с.
4. Стецюк П.И., Пилиповский А.В. Математическая модель оптимальной загрузки мощностей энергосистемы с учетом их маневренности. Праці IV міжнародної школи-семінару "Теорія прийняття рішень". – Ужгород: УжНУ, 2008. – С.159.
5. Лиховид А.П. О реализации параллельного алгоритма для решения многоэкстремальных задач // Теорія оптимальних рішень. – Київ: Ін-т кібернетики ім. В.М.Глушкова НАН України, 2010. – №9. – С. 3–9.
6. Журбенко Н.Г., Чумаков Б.М. Метод модельных теплообменников в расчетах тепловых схем. // Там же. – 2009. – № 8 – С. 142–147.
7. Разработка программных средств оптимального проектирования энергетических котлоагрегатов ТЭС / Ю.П. Лаптін, М.М Левин, Н.Г. Журбенко, П.И. Волковицкая, Д.А. Коваленко // Цільова комплексна програма НАН України «Проблеми ресурсу і безпеки експлуатації конструкцій споруд та машин». Збірник наукових статей за результатами, отриманими в 2007 – 2009 рр. – Київ: Ін-т електрозварювання ім. Є.О. Патона НАН України, 2009. – С. 349 – 355.
8. О разработке программного обеспечения задач оптимального проектирования теплоэнергетических установок / Ю.П. Лаптин, Н.Г. Журбенко, М.М. Левин, П.И. Волковицкая // Кибернетика и систем. анализ. – 2011. – № 1. – С. 116–127.
9. Шор Н.З. Методы минимизации недифференцируемых функций и их приложения. – Киев: Наукова думка. – 1979. – 200 с.
10. Шор Н.З., Журбенко Н.Г. Метод минимизации, использующий операцию растяжения пространства в направлении разности двух последовательных градиентов // Кибернетика. – 1971. – № 3. – С. 51–59.

11. Fourer R., Gay D., Kernighan B. AMPL: A Modeling Language for Mathematical Programming. Duxbury Press/Brooks/Cole Publishing Company, – 2003. – 517 p.

12. NEOS Server for Optimization. http://www.neos-server.org/neos/.

13. Лиховид А.П. Об одной реализации *r*-алгоритма // Теорія оптимальних рішень. – Київ: Ін-т кібернетики ім. В.М.Глушкова НАН України, 2011. – № 10. – С. 91–95.

14. Кластерный комплекс Института кибернетики. Кластерный комплекс СКИТ. https://icybcluster.org.ua/.

15. Журбенко Н.Г., Чумаков Б.М. Модели управления энергетической системой // Теорія оптимальних рішень. – Київ: Ін-т кібернетики ім. В.М.Глушкова НАН України, 2007. – С. 100–107.

16. Михалевич В.С., Шор Н.З. Метод последовательного анализа вариантов при решении вариационных задач управления, планирования и проектирования // Доклады на IV Всесоюзном математ. съезде. – Л.: 1961. – С. 91.

17. Михалевич В.С., Шор Н.З. Численное решение многовариантных задач по методу последовательного анализа вариантов. Научно-методические материалы экономико-математического семинара. – М.: ЛЭММ АН СССР, 1962. – Вып.1. – С. 15–42.

18. Вычислительные методы выбора оптимальных проектных решений / В.С. Михалевич, Н.З. Шор, Л.А. Галустова, Н.Г. Журбенко и др. – Киев. Наук. думка, 1977. – 178 с.

19. Беллман Р. Динамическое программирование. – М.: Изд-во иностр. лит., 1960. – 400 с.

20. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608 с.

21. Журбенко Н.Г., Шарифов Ф.А. Об одном алгоритме решения задачи выбора режимов энергосистемы // Теорія оптимальних рішень. – Київ: Ін-т кібернетики ім. В.М.Глушкова НАН України, 2010. – № 9. – С. 149–154.

22. Шарифов Ф.А. Задача выбора режимов объединенной энергосистемы по активной мощности // Там же. – 2007. – №6. – С.125–130.

23. Шарифов Ф. А. Методы решения задачи выбора режимов объединенной энергосистемы по активной мощности // Там же. – 2009. – № 8. – С. 9–16.

24. Тепловой расчет котельных агрегатов (нормативный метод). – М.: Энергия,1973. – 295 с.

25. Тепловые схемы котлов / А.А. Паршин, В.В. Митор, А.Н. Безгрешнов и др. – М.: Машиностроение, 1987. – 222 с.

Додаток А
Шаблон опису нелінійної задачі з урахуванням маневреності на мові AMPL

```
param N ; #  кількість станцій
param T ; #  кількість інтервалів часу
param K ; #  кількість екологічних факторів
# кількість точок розриву для кусково-лінійних функцій
# вартості палива
param points_costs{1..N};
# кількість точок розриву для кусково-лінійних функцій
# оцінки факторів екології
param points_ecology{1..N,1..K} ;
# значення потужностей станцій в точках розриву
# кусково-лінійних функцій вартості палива
param powers_costs{i in 1..N,1..points_costs[i]} ;
# значення потужностей станцій в точках розриву
# кусково-лінійних функцій оцінки факторів екології
param powers_ecology{i in 1..N,k in
1..K,1..points_ecology[i,k]} ;
# значення вартості палива для потужностей в точках розриву
# кусково-лінійних функцій вартості палива
param costs_fuel{i in 1..N,1..points_costs[i]};
# значення вартості врахування факторів екології в точках
# розриву кусково-лінійних функцій їх оцінки
param costs_ecology{i in 1..N,k in
1..K,1..points_ecology[i,k]};
# значення коефіцієнта нахилу ліворуч від першої точки
# розриву для кусково-лінійних функцій вартості палива
param left_slope_value_costs{1..N} ;
# значення коефіцієнта нахилу ліворуч від першої точки
# розриву для кусково-лінійних функцій оцінки факторів
# екології
param left_slope_value_ecology{1..N,k in 1..K} ;
# значення коефіцієнта нахилу праворуч від останньої точки
# розриву для кусково-лінійних функцій вартості палива
param right_slope_value_costs{1..N} ;
# значення коефіцієнта нахилу праворуч від останньої точки
# розриву для кусково-лінійних функцій оцінки факторів
# екології
param right_slope_value_ecology{1..N,k in 1..K} ;
# кути нахилу лінійних відрізків для кусково-лінійних
# функцій вартості палива
param slopes_costs{j in 1..N,i in 0..points_costs[j]} := if
(i = 0) then left_slope_value_costs[j]
else  if (i = points_costs[j]) then
right_slope_value_costs[j]
else  (costs_fuel[j,i+1]-
costs_fuel[j,i])/(powers_costs[j,i+1]-powers_costs[j,i]);
# кути нахилу лінійних відрізків для кусково-лінійних
```

```
# функцій оцінки факторів екології
param slopes_ecology{j in 1..N,k in 1..K,i in
0..points_ecology[j,k]} :=
if (i = 0) then left_slope_value_ecology[j,k]
else  if (i = points_ecology[j,k]) then
right_slope_value_ecology[j,k]
else  (costs_ecology[j,k,i+1]-
costs_ecology[j,k,i])/(powers_ecology[j,k,i+1]-
powers_ecology[j,k,i]);
# зсув по осі ординат для кусково-лінійних функцій вартості
# палива
param shifti{i in 1..N} := -
left_slope_value_costs[i]*powers_costs[i,1]+costs_fuel[i,1]
;
# зсув по осі ординат для кусково-лінійних функцій оцінки
# факторів екології
param shiftik{i in 1..N,k in 1..K} := -
left_slope_value_ecology[i,k]*powers_ecology[i,k,1] +
costs_ecology[i,k,1] ;
# параметри задачі
param lambdax ;
param lambday ;
param lambdaz ;
param fup ;
param deltai{1..N}   ;
param deltat{1..T-1} ;
param demand{1..T};
param Plow{1..N} ;
param Pup{1..N}  ;
param xlow{1..N,1..T} ;
param xup{1..N,1..T}  ;
param A{k in 1..K}  ;
#змінні задачі
var x{i in 1..N,t in 1..T} >= xlow[i,t], <= xup[i,t] ;
var y{t in 1..T-1} >= 0, <= deltat[t] ;
var fx >= 0, <= fup;
var z{i in 1..N} >= 0, <= deltai[i] ;
# цільова функція задачі
minimize f:
lambdax*fx + lambday*sum {t in 1..T-1} y[t] + lambdaz*sum {i
in 1..N} z[i];
# обмеження задачі
subject to fuelcost:
sum {i in 1..N, t in 1..T}( << {j in 1..points_costs[i]}
powers_costs[i,j] ;
{j in 0..points_costs[i]} slopes_costs[i,j] >> x[i,t]  +
shifti[i]) <= fx;
subject to ecology {k in 1..K}:
sum {i in 1..N, t in 1..T}
```

```
( << {j in 1..points_ecology[i,k]} powers_ecology[i,k,j] ;
{j in 0..points_ecology[i,k]} slopes_ecology[i,k,j] >>
x[i,t] + shiftik[i,k]) <= A[k];
subject to demands {t in 1..T}:
   sum {i in 1..N} x[i,t] = demand[t];
subject to manevrt {t in 1..T-1 }:
sum {i in 1..N} ( abs(x[i,t+1]-x[i,t] )) <=
abs(demand[t+1]-demand[t]) + y[t];
subject to manevri {i in 1..N }:
sum {t in 1..T-1} ( abs(x[i,t+1]-x[i,t] )) <=  z[i];
```

Цей шаблон доповнювався даними для конкретного варіанта задачі і був випробуваний як вхідний формат для різних програм з NEOS-сервера. Для розрахунків використані наведені нижче дані.

```
data;

param N := 11; param T := 24; param K :=  2; param lambdax
:= 1;
param lambday := 1; param lambdaz := 1; param fup :=
100000000;
param points_costs :=
1  2 2  2 3  2 4  2 5  2 6  2 7  2 8  2 9  2 10 2 11 2;

param left_slope_value_costs :=
1  -1000 2  -1000 3  -1000 4  -1000 5  -1000 6  -1000
7  -1000 8  -1000 9  -1000 10 -1000 11 -1000 ;

param right_slope_value_costs :=
1  1000 2  1000 3  1000 4  1000 5  1000 6  1000 7  1000
8  1000 9  1000 10 1000 11 1000 ;

param points_ecology :
      1      2    :=
1     2      2 2     2      2 3     2      2 4     2      2
5     2      2 6     2      2 7     2      2 8     2      2
9     2      2 10    2      2 11    2      2;

param left_slope_value_ecology :
        1      2    :=
1   -1000  -1000  2   -1000  -1000  3   -1000  -1000
4   -1000  -1000  5   -1000  -1000  6   -1000  -1000
7   -1000  -1000  8   -1000  -1000  9   -1000  -1000
10  -1000  -1000  11  -1000  -1000  ;

param right_slope_value_ecology :
        1      2    :=
1     1000    1000  2     1000    1000  3     1000    1000
```

```
4     1000    1000  5     1000    1000  6     1000    1000
7     1000    1000  8     1000    1000  9     1000    1000
10    1000    1000  11    1000    1000  ;

param deltai :=
1  1000 2  1000 3  1000 4  1000 5  1000 6  1000 7  1000 8
1000
9  1000 10 1000 11 1000;

param Plow :=
1  0 2  0 3  0 4  0 5  0 6  0 7  0 8  0 9  0 10 0 11 0;

param Pup :=
1  300 2  300 3  300 4  300 5  300 6  300 7  300 8  300 9
300 10 300 11 300;

param xlow :
       1     2     3     4     5     6     7     8     9
       10    11    12    13    14    15    16    17
       18    19    20    21    22    23    24 :=
 1     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
 2     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
 3     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
 4     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
 5     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
 6     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
 7     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
 8     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
 9     0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0     0     0     0
       0     0     0     0     0     0
```

```
10      0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0
11      0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0       0       0       0
        0       0       0       0       0       0;

param xup :
        1       2       3       4       5       6       7       8       9
        10      11      12      13      14      15      16      17
        18      19      20      21      22      23      24 :=
 1      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
         2      300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300
 3      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
 4      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
 5      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
 6      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
 7      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
 8      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
 9      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
10      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
11      300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300
        300     300     300     300     300     300     300     300;

param A :=
1  1000 2  1000;
```

```
param deltat :=
1 1000 2 1000 3 1000 4 1000 5 1000 6 1000 7 1000 8 1000 9
1000 10 1000
11 1000 12 1000 13 1000 14 1000 15 1000 16 1000 17 1000 18
1000 19 1000
20 1000 21 1000 22 1000 23 1000;

param demand :=
1 1000 2 1000 3 1200 4 1200 5 1500 6 2700 7 2700 8 2700 9
1500 10 1500
11 1500 12 1600 13 1700 14 1800 15 2000 16 2000 17 2500 18
3000 19 3000
20 3000 21 2500 22 2000 23 1500 24 1000;

param powers_costs:
   1   2    :=
1  0  300 2  0  300 3  0  300 4  0  300 5  0  300 6  0  300
7  0  300
8  0  300 9  0  300 10 0  300 11 0  300;

param costs_fuel :
      1       2    :=
1       0         115890   2       0         124680   3       0
109650
4       0         126300   5       0         125130   6       0
124290
7       0         128400   8       0         114030   9       0
110250
10    0      127200  11    0      122460;

param powers_ecology :=
 [*,*,1]:       1     2 :=
        1        0      0          2        0      0          3
0     0
        4        0      0          5        0      0          6
0     0
        7        0      0          8        0      0          9
0     0
       10        0     0        11        0     0
 [*,*,2]:       1     2 :=
        1       300    300          2       300    300          3
300   300
        4       300    300          5       300    300          6
300   300
        7       300    300          8       300    300          9
300   300
       10      300   300        11      300   300;
```

```
param costs_ecology :=
 [*,*,1]:        1     2 :=
        1         0      0         2         0      0         3
0     0
        4         0      0         5         0      0         6
0     0
        7         0      0         8         0      0         9
0     0
       10         0      0        11         0      0
 [*,*,2]:        1      2 :=
        1      1.15890 1.15890         2      1.24680 1.24680
        3      1.09650 1.09650         4      1.26300 1.26300
        5      1.25130 1.25130         6      1.24290 1.24290
        7      1.28400 1.28400         8      1.14030 1.14030
        9      1.10250 1.10250        10      1.27200 1.27200
       11      1.22460 1.22460;
```

Додаток Б
Скрипт для запуску програми amplralg та тестовий приклад

```
Скрипт для запуску програми amplralg:
option solver amplralg;
model energy_pete_all_2.mod;
data energy_pete_all_2.dat;
solve;
display f;
display x;
display fx;
display sum {t in 1..T-1} y[t];
display sum {i in 1..N} z[i];
display ecology;
Зразок текстового файлу з параметрами r-алгоритму:
Точність по аргументу                1.e-6
Точність по градієнту                1.e-6
Початковий крок                      1.0
Параметр Q1                          0.95
Параметр Q2                          1.20
Коефіцієнт розтягу alpha             2.0
Штрафний коефіцієнт              90000.00
Результати розрахунків програмою amplralg для тестового
прикладу
з 11 енергоблоками і 24 інтервалами планового періоду:
sign = -1
penalty= 90000
-------------------amplRalg started--------------------
Iter     Obj             Shift X         Fun calls
-----------------------------------------------------
0        270000000           1.73205     1
1000     112827113.894       66.4769     4205
2000     109896768.531       9.12753     8285
3000     108850458.373       22916.2     12435
4000     18207128.4484       348.372     16767
5000     18093982.0891       276.323     20887
6000     17999543.3611       1.39629     24849
7000     18019435.3021       37.5787     28865
Attention,please, linear search exceeded the limit of
steps!
Iteration= 7638
Function= 1.79971e+007
8000     18017981.194        11.7606     32895
9000     17992434.5461       0.267063    36849
Attention,please, linear search exceeded the limit of
steps!
Iteration= 9091
Function= 1.79924e+007
```

```
10000    17993331.2831       1.59735     41035
Reset of the matrix B: d1= 2.02101e-016
10757    17992951.4756       6.60178     44113
11000    18000426.5443       0.0370211   45088
12000    17992849.2311       0.00280452  49428
13000    17992962.7343       0.116676    53596
14000    17992276.6073       0.0274672   57680
15000    17992277.5548       0.0076549   61758
16000    17992293.2468       0.0195926   65774
17000    17992270.8403       0.000723604 69780
18000    17992270.1119       0.00252047  73866
19000    17992270.1363       0.000172433 77916
19331    17992270.0197       9.95979e-00779217
-----------------------------------------------------
Optimal solution found
-----------------------------------
Program takes 36.548 seconds.
-----------------------------------
max_value = 2.12594e-009
delta = 0.0001
amplRALG has been finished
amplRALG has been finished
f = 17992300
x [*,*] (tr)
:       1      2      3      4      5      6 :=
1       100    3.44633e-08  300    1.00014e-07
 6.19619e-07  1.68725e-07
2       100    1.88122e-07  300    2.85058e-07
 3.19806e-07  1.17956e-08
3       300    1.35712e-07  300    3.16754e-07
 4.75973e-07  2.3536e-07
4       300    7.21429e-07  300    2.16157e-06
 1.71892e-06  9.52961e-08
5       300    1.12724e-06  300    2.84318e-06
 3.37536e-06  7.23595e-07
6       300    300    300    200    200    300
7       300    300    300    200    200    300
8       300    300    300    200    200    300
9       300    100    300    5.4622e-07   1.96236e-06
        100
10      300    99.9999      300    4.11251e-07
 1.86587e-06  100
11      300    99.9999      300    3.6026e-07
 1.9274e-06   100
12      300    99.9999      300    4.22481e-06
 5.88267e-07  100
13      300    100    300    3.77016e-07  6.25005e-06
 100
```

```
14      300    100    300    3.49115e-06  6.85383e-06
 200
15      300    200    300    4.31297e-06  1.59743e-05
 300
16      300    200    300    4.08918e-07  1.61245e-05
 300
17      300    300    300    100    300    300
18      300    300    300    300    300    300
19      300    300    300    300    300    300
20      300    300    300    300    300    300
21      300    300    300    100    300    300
22      300    200    300    1.09115e-05  3.07266e-05
 300
23      300    1.77672e-06  300    2.46058e-07
 1.04594e-06  1.51888e-05
24      100    6.11742e-07  300    3.11583e-07
 3.91674e-07  3.95784e-07
:       7      8      9      10     11     :=
1     5.7931e-07    300    300     2.09818e-07
      1.45518e-05
2     2.56683e-07   300    300     2.77027e-07
      5.42103e-07
3     1.11905e-07   300    300     2.05341e-06
      2.80413e-07
4     1.4679e-07    300    300     2.34903e-07
      3.82531e-06
5     2.42733e-07    300    300     2.92871e-06     300
6     4.2129e-05     300    300     200             300
7     2.4998e-05     300    300     200             300
8     0.000125347    300    300     200             300
9     1.48949e-06    300    300     4.71404e-06     100
10    2.24677e-07    300    300     2.67207e-07     100
11    2.17502e-06    300    300     2.13448e-07     100
12    3.09518e-07    300    300     1.36786e-07     200
13    2.48492e-07    300    300     5.00507e-08     300
14    9.65564e-07    300    300     4.93655e-06     300
15    2.40945e-06    300    300     2.10771e-06     300
16    1.03011e-06    300    300     4.86232e-08     300
17    4.95509e-07    300    300     2.58058e-06     300
18    0.000129781    300    300     300             300
19    6.35954e-05    300    300     300             300
20    7.84882e-05    300    300     300             300
21    8.95353e-07    300    300     6.26002e-06     300
22    2.72998e-07    300    300     5.40746e-06     300
23    2.15955e-06    300    300     3.38229e-08     300
24    3.01339e-07   300    300     8.08975e-08
      4.94647e-06
;
```

Додаток В
Опис функцій–підпрограм PAVTab і PAVSol (мова РАТФОР)

Підпрограми PAVTab і PAVSol використовують наступну структуру даних про енергоблоки і таблиці Белмана:

```
#  Дані, що використовуються підпрограмами PAVTab і PAVSol
#навантаження енергоблоків і витрати умовного палива для них
# integer*2 N          # кількість енергоблоків (i=1,...,N)
# integer*4 hs(N+1)    # hs(i)- індекс, з якого в масивах
# hsp(*) і fsp(*) починаються навантаження і витрати
# умовного палива для і-го енергоблока, тобто кількість
# навантажень для нього дорівнює hs(i+1)-hs(i)
# integer*2 hsp(Npelms)# навантаження всіх N енергоблоків,
# де Npelms їх повна кількість, що дорівнює hs(N+1)-hs(1)
# real*8    fsp(Npelms)# витрати умовного палива
# (відповідають навантаженням)
# Навантаження і-го енергоблока (i=1,...,N) зберігаються
# за адресами hsp(j),j=hs(i),...,hs(i+1)-1,витрати
# ум.пал. для них - fsp(j),j=hs(i),...,hs(i+1)-1
# Примітка: навантаження для кожного енергоблока повинні
# бути впорядковані за зростанням. Це потрібно тому, що
# hsp(hs(i+1)-1) використовується як максимальне
# навантаження і-го енергоблока.
# ... таблиці (функції) Белмана F(k) і i(k), де k=1,...,N
# integer*4 MaxE           # максимальна потреба в енергії
# (по інтервалах)
# використовується як максимальна довжина таблиць F
# real*8    fk(MaxE+1)  # таблиця F для енергоблока k=N,
# де fk(E+1) містить мінімальне значення цільової функції
# для всіх E<=MaxE
# integer*2 ixk(MaxE+1) # таблиця і для енергоблока k=N, де
# ixk(E+1) містить його оптим. навантаження для всіх
# можливих E<=MaxE.
# Якщо реалізувати значення E не можна, то ixk(E+1)=-1.
# integer*4  Ltab     # розмір(довжина) масиву hitab(*), що
# потрібен був для упакування таблиць i(k) для всіх
# k=1,...,(N-1)
# integer*4 htab(N)     # htab(k)-індекс початку таблиці i(k)
# у масиві hitab(*)
# integer*2 hitab(Ltab)# hitab(j),j=htab(k),...,htab(k+1)-1
# - оптимальні
# навантаження k енергоблока для всіх можливих E<=MaxE.
```

Підпрограма PAVTab будує таблиці Белмана (множину повних допустимих послідовностей з найменшим значенням цільової функції) для всіх можливих цілочислових правих частин в обмеженні (2.5). Підпрограму PAVTab реалізує такий РАТФОР-код:

```
subroutine PAVTab(N,hs,hsp,fsp, # дані по енергоблоках
                                #(вхідні)
                  MaxE,         # максимум необхідної
                                # енергії (вхідна)
                  fk,ixk,       # підсумкова (N) таблиця
                                # Белмана (вихід)
                  htab,hptb)    # таблиці 1,...,(k-1)
                                # упаковані(вихід)
implicit real*8(a-h,o-z),integer*4(i-n)
real*8 fsp(1),fk(MaxE+1),f1(MaxEnergy2)# робочий масив f1(*)
integer*4 hs(N+1),htab(N)
integer*2 hsp(1),hptab(1),ixk(MaxE+1),ix1(MaxEnergy2)
# робочий масив ix1(*)
# масиви f1(*) і ix1(*) використовуються для проміжної
# таблиці Белмана
ind =hs(1); ist=hs(2)-1;        # індекси початку і кінця
                                # першого енергоблока
hup1=hsp(ist)+1;                # верхня границя 1-ої таблиці
                                # Белмана дорівнює
if (hup1>MaxE+1) hup1=MaxE+1     # максимальному навантаженню
                                # 1-го енергоблока

for (i=1; i<=MaxE+1; i=i+1) [   # Ініціалізація 1-ої таблиці
Белмана
  f1(i)=1.d+20; ix1(i)=-1;
]
for (i=ind; i<=ist; i=i+1) [    # заповнюємо 1-у таблицю
                                # Белмана і робимо
  ij=hsp(i)+1                   # це зі зміщенням на одиницю,
                                # тобто y=1 буде
  f1(ij)=fsp(i);                # відповідати y=0 (пов'язано
                                # з Фортраном)
  ix1(ij)=hsp(i);
]
htab(1)=1; Ltab=1;              # початкові параметри
                                # упакованої таблиці
for (k=2; k<=n; k=k+1) [        # основний цикл для
                                # алгоритму PAV
   ind =hs(k); ist=hs(k+1)-1;   # установка початку і кінця
                                # k-го енергоблока
   hupk=hup1+hsp(ist);          # корекція верхньої границі
                                # для k-ої таблиці
   if (hupk>MaxE+1) hupk=MaxE+1# з урахуванням зміщення на
                                # одиницю
   for (i=1; i<=MaxE+1; i=i+1)[# ініціалізація нової таблиці
      fk(i)=1.d+20; ixk(i)=-1;
   ]
   for (i=hup1; i>=1; i=i-1) [ # перерахування для k-ої
                                # таблиці
```

```
      if (ix1(i)<0) next
      for (j=ind; j<=ist; j=j+1) [
         ij=i+hsp(j)
         if (ij>hupk) break
         ff=f1(i)+fsp(j)
         if (ff>=fk(ij)) next
         fk(ij)=ff; ixk(ij)=hsp(j)
      ]
   ]
  # зберегти (k-1)-таблицю (ix1(i),hup1) в упакованій
  # структурі
   for (i=1; i<=hup1; i=i+1) [
      hptab(Ltab)=ix1(i); Ltab=Ltab+1
   ]
   htab(k)=Ltab
  # запам'ятати k-ий елемент таблиці (fk(i),ixk(i),i=1,hupk)
  # у таблицях (f1(i),ix1(i),i=1,hup1) для наступного кроку
  # циклу
   for (i=1; i<=MaxE+1; i=i+1) [
      f1(i)=fk(i); ix1(i)=ixk(i)
   ]
   hup1=hupk
]
return
end   #PAVTab
```

На основі побудованих підпрограмою PAVTab таблиць Белмана підпрограма PAVSol знаходить оптимальний розв'язок задачі (2.4)–(2.6) для довільної правої частини E в обмеженні (2.5) $0 < E \le E_{\max}$. Підпрограму PAVSol реалізує наступний РАТФОР-код:

```
subroutine PAVSol(N,                  # число енергоблоків
                                      # (вхідний)
                  MaxE,               # максимум необхідної
                                      # енергії (вхідний)
                  NE,                 # необхідна енергія
                                      # (вхідний)
                  Fopt,ISol,          # оптимальний розв'язок
                                      # (вихід)
                  fk,ixk,             # підсумкова (N) таблиця
                                      # Белмана (вхід)
                  htab,hptab)         # таблиці 1,...,(k-1)
                                      # упаковані (вхід)
implicit real*8(a-h,o-z),integer*4(i-n)
real*8 fk(MaxE+1),Fopt
integer*4 htab(N)
integer*2 hptab(1),ixk(MaxE+1),ISol(N)
Fopt=1.d+20
for (i=1; i<=N; i=i+1) [
  ISol(i)=-1;
```

```
]
ij=NE+1  # облік зміщення правої частини на одиницю
if (NE>MaxE) return# ЕЕ-потреба E=NE недосяжна,
                   # тобто (E>Emax),
if (ixk(ij)<0) return # ЕЕ-потреба E=NE недосяжна,
                      # тобто ixk(E)=-1;
Fopt=fk(ij)       # встановити оптимальне значення
                  # цільової функції
ISol(N)=ixk(ij)   # встановити оптимальну потужність
                  # N-го енергоблока
for (k=N-1; k>=1; k=k-1) [    # розшифровка оптимальних
                              # потужностей
  ij=ij-ISol(k+1)             # інших (N-1),...,1
                              # енергоблоків
  ind =htab(k)
  ISol(k)=hptab(ind+ij-1)
]
return
end #PAVSol
```

Стецюк Петро Іванович
Журбенко Микола Георгійович
Лиховид Олексій Петрович

МАТЕМАТИЧНІ МОДЕЛІ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ В ЗАДАЧАХ ЕНЕРГЕТИКИ

В авторській редакції
Верстка авторська

Підписано до друку 20.06.2012. Зам. № 0916
Гарнітура "Таймс". Друк офсетний. Папір офсетний.
Формат 64x90 1/16. Обл.-вид. арк. 4,27.
Ум. друк. арк. 2,66.
Наклад 300 прим.

Видавець і виготовлювач: ПП «Ательє «Поліграфічний комплекс»
вул. Автозаводська, 76, Київ-114, 04114, Україна
тел. (+38 044) 502-49-65 (66, 68)
e-mail: alexprint@svitonline.com