УДК 519.688
П.Г.Стахів, І.П.Струбицька
Тернопільський національний економічний університет

МЕТОД РОЗПАРАЛЕЛЕННЯ ЗАДАЧІ ІДЕНТИФІКАЦІЇ ПАРАМЕТРІВ ДИСКРЕТНИХ ДИНАМІЧНИХ МАКРОМОДЕЛЕЙ НА МАСИВНО-ПАРАЛЕЛЬНИХ ПРОЦЕСОРАХ

В роботі запропоновано метод розпаралелення задачі ідентифікації параметрів дискретної динамічної макромоделі на масивно-паралельних процесорах. Запропоновано структуру алгоритму розпаралелення та визначено клас задач, для яких даний підхід буде оптимальний.

Ключові слова: *динамічні макромоделі, масивно-паралельні процесори, розпаралелення задачі ідентифікації параметрів.*

Постановка проблеми. Побудова математичної моделі з використанням оптимізації в загальному випадку зводиться до знаходження мінімуму певної функції, яка характеризує відхилення поведінки моделі від поведінки модельованого об'єкта. З одного боку, це дозволяє значно універсалізувати даний підхід, а з іншого – приводить до появи складних оптимізаційних задач, які важко розв’язати навіть з використанням сучасних засобів обчислювальної техніки.

Аналіз останніх досліджень та публікацій. Задачі параметричної ідентифікації дискретних динамічних моделей в достатній мірі описані в працях В.М. Кунцевича [9], М.М. Личака [10], Ф.Л. Черноусько [14-15], Л. Заде та Ч. Дезоера [8], В. Стрейца [3].

З точки зору комп'ютерної імітації найбільш перспективним є метод дискретних рівнянь стану [11-12]. Цей підхід, з математичної точки зору, є найбільш формалізованим.

Мета дослідження. Дослідити методи розпаралелення задачі ідентифікації параметрів дискретної динамічної макромоделі на масивно-паралельних процесорах.

Основні результати дослідження. Детальне дослідження складних систем є важливим напрямком сучасної науки. Традиційно вивченням поведінки складних систем займається загальна теорія систем, яка зосереджена на розробці концептуального і математичного апарату системних досліджень. Проте за останні роки широкої популярності набуло дослідження складних систем як самостійний міждисциплінарний напрямок науки. Дана теорія направлена на розробку загальних підходів до дослідження систем, що складаються з великої кількості елементів зі складними зв'язками, з метою підвищення рівня розуміння універсальних принципів, що лежать в основі їх функціювання та розвитку. Даний підхід не відкидає досягнень системного аналізу і теорії систем, а доповнює їх специфічним апаратом досліджень. Це принципово важливо через те, що експериментально досліджувати складні реальні системи дуже важко, а інколи й неможливо, і основним методом отримання знань про систему є її математичне моделювання.

Важливим етапом у дослідженні складних динамічних систем є ідентифікація параметрів побудованої моделі. Цей етап відіграє ключову роль при оцінці адекватності моделі у відношенні до реальної системи, тому що дозволяє застосувати формальні методи перевірки і тестування. Даний етап практично неминучий при побудові адекватних моделей, так як модель складної динамічної системи дуже важко побудувати. Дуже багато параметрів складних систем погано піддаються формалізації з точки зору реальних процесів, і для їх оцінки потрібні спеціальні підходи. Як правило, такі підходи базуються на пошукових методах оптимізації, коли функція мети є мірою відхилення між спостереженнями за станом реальної системи і результатом її моделювання, а параметри цієї функції є невідомими параметрами моделі досліджуваної системи. Дана задача ускладнюється неможливістю одночасного спостереження за всіма компонентами складної системи в усіх діапазонах їх зміни, роз'єднаністю експериментальних даних і високими вимогами до обчислювальних ресурсів, за допомогою яких проводиться комп'ютерне моделювання. У результаті вирішення задачі узагальнення розрізнених експериментальних даних і проведення комп'ютерного моделювання за розумний час можливо лише при використанні технологій високопродуктивних обчислень. При цьому формальні способи розпаралелювання (на рівні коду) часто не є ефективними через ієрархічність створюваних моделей, які описують систему в різних часових і просторових діапазонах, так і через допущення про слабкі зв'язки між компонентами.

Крім того, сучасні складні системи у більшості випадків потребують опису динамічними моделями, що породжує залежність між станами моделі в часі, що вкрай небажано з точки зору їх паралельної реалізації формальними методами.

Ці всі проблеми вимагають спеціальних підходів до розробки проблемно-орієнтованих паралельних алгоритмів ідентифікації моделей складних систем, які б враховували специфіку математичних методів, особливості модельованого явища і характеристики паралельної обчислювальної архітектури.

Будь-яка складна система містить певні особливості, які стають важливими чинниками при її моделюванні засобами комп'ютерної техніки. До таких особливостей варто віднести: велика кількість компонентів, які утворюють систему, існування сильних і слабких взаємовпливів між компонентами та багатомасштабність системи (часова, просторова, міжкомпонентна). Ці чинники при моделюванні системи породжують в тій чи іншій мірі певні проблеми. Основними з них є: потреба в великих ресурсах для обчислювальних процедур, великі об'єми оперативної пам'яті для зберігання даних, складність проведення паралельних обчислень формальними методами та можливість зрозумілого пояснення результатів моделювання емпіричними даними та закономірностями.

Для ідентифікації моделей використовуються реальні дані досліджуваних систем, які також мають в деяких випадках негативний вплив на якість створених моделей. У першу чергу це стосується того факту, що для більшостей систем неможливо зібрати всі дані про систему, в результаті чого дані стають фрагментарними, можуть містити пропуски, викиди, а в деяких випадках є досить нерівномірними в часі. Неоднорідність даних може бути проявом різних постановок задач або методів і засобів їх виміру.

Враховуючи запропонований підхід до ідентифікації динамічної моделі складної системи, можна запропонувати наступну структурну схему такої ідентифікації (рис. 1).

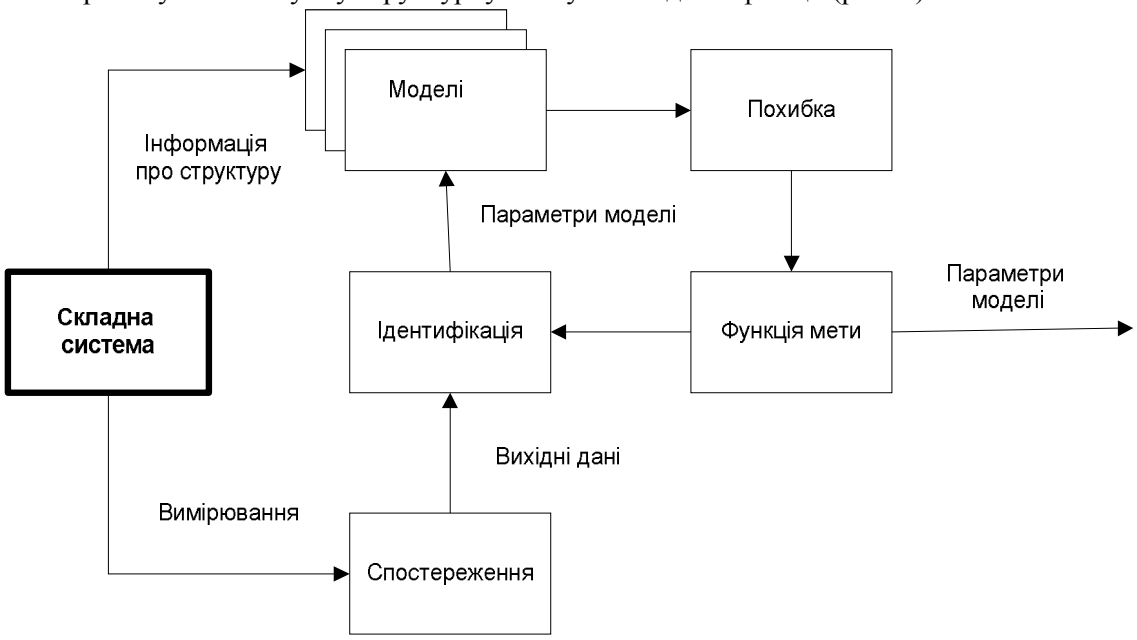

Рис. 1. Узагальнена схема ідентифікації моделі

Пояснення даної узагальненої схеми ідентифікації моделей доволі просте.

На основі деякої інформації про систему, її інфраструктуру будується загальний вигляд (форма) моделі системи. Досить добре себе зарекомендували моделі на основі рівнянь стану, тому у дослідженні основна увага буде приділена саме їм. Тобто модель системи буде записана як система дискретних рівнянь стану.

Ідентифікація параметрів такої моделі буде проводитись за деяким алгоритмом, використовуючи вхідні і вихідні дані реальної системи. Але, на відміну від послідовного підходу, алгоритм ідентифікації буде породжувати не один набір параметрів моделі стану, а деяку множину параметрів. Це дасть змогу у блоці «Моделі» провести порівняння моделей з однаковою структурою, але з різним набором параметрів. Оцінка похибки ідентифікації та обчислення функцій мети з різними параметрами дозволять визначити адекватність моделі до реального

об'єкта. При недостатній адекватності буде потрібно провести повторну ідентифікацію зі зміною порядку самої моделі або зміненою формою врахування нелінійності в динамічній моделі.

Розглянемо деяку загальну систему, поведінку якої моделюють. Для цього за її узагальнену структуру приймають модель «вхід-вихід». Позначимо внутрішні змінні, що описують стан системи, вектором $\vec{x}$, зовнішні збурення, що впливають на стан системи, – вектором $\vec{v}$, а вихідні величини — вектором $\vec{y}$ (рис. 2).

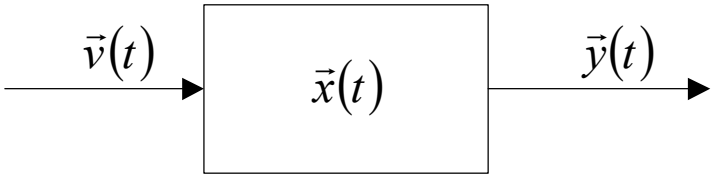

Рис. 2. Загальна схема моделі

Оскільки розглядаються динамічні системи, то очевидно, що усі три згадані вектори будуть залежними від часу. В одних випадках це можуть бути неперервні функції від часу ($\vec{x}(t), \vec{v}(t), \vec{y}(t)$), тоді модель системи буде представлена у вигляді системи диференціальних рівнянь з деякими початковими значеннями. В іншому ж випадку — це набір дискретних значень. Зауважимо, що будь-який чисельний метод розв'язку системи диференціальних рівнянь проводить дискретизацію змінних, що зводить систему диференціальних рівнянь до системи різницевих рівнянь з дискретними змінними. Тому зупинимося на аналізі дискретного випадку, оскільки він більш зручний для реалізації засобами комп'ютерної техніки.

Розглянемо макромодель системи у формі дискретних рівнянь стану [8]:

$$\begin{cases} x^{(k+1)} = Fx^{(k)} + Gv^{(k)} + \Phi\left(x^{(k)}, v^{(k)}\right) \\ y^{(k+1)} = Cx^{(k+1)} + Dv^{(k+1)} \end{cases}, \quad (1)$$

де $x^{(k)}$ - вектор змінних стану; $v^{(k)}$ – вектор вхідних значень; $y^{(k+1)}$ – вектор вихідних значень; F, G, C, D - матриці, з невідомими коефіцієнтами, які необхідно знайти при побудові макромоделі; $\Phi(\vec{x}^{(k)}, \vec{v}^{(k)})$ - деяка вектор-функція, форму і коефіцієнти якої також потрібно відшукати.

Дана форма макромоделі, тобто рівняння стану системи (1), характеризується деяким набором невідомих параметрів $\vec{\lambda}$. Для конкретної моделі цей вектор складається з елементів матриць F, G, C, D і коефіцієнтів розкладу вектор-функції $\Phi(\vec{x}^{(k)}, \vec{v}^{(k)})$.

Для оцінювання адекватності побудованої моделі необхідно оцінити її точність. Критерій точності моделі $Q(\vec{\lambda}) > 0$ відображає похибку відтворення моделлю поведінки системи. Функція мети Q може відображати не лише точність відтворення поведінки системи, а й враховувати деякі бажані критерії самої моделі, наприклад її складність. Як правило, в якості функції мети вибирають середньоквадратичне відхилення поведінки моделі від поведінки модельованого об'єкта:

$$Q(\vec{\lambda}) = \sum_k |\vec{y} - \vec{y}'|^2, \quad (2)$$

де $\vec{y}$ - перехідні характеристики розраховані за допомогою макромоделі, $\vec{y}'$ - наперед відомі характеристики модельованого об'єкта.

При такому підході побудова моделі зводиться до знаходження такого значення $\tilde{\tilde{\lambda}}$, при якому значення функції Q буде мінімальним. Отже, задача побудови макромоделі зводиться до пошуку мінімуму функції, а знаходження вектора невідомих параметрів $\tilde{\tilde{\lambda}}$ проводиться за допомогою оптимізації.

В загальному випадку задача знаходження мінімуму нелінійної функції Q є доволі складним завданням, якому присвячено багато праць. Для оптимізації пошуку застосовуються різні методи, які враховують характерні особливості конкретної задачі. У нашому випадку побудови моделі

характерною особливістю функції, яку потрібно оптимізувати, є складність її розрахунку в сенсі обчислювальних затрат (процесорний час). Оскільки функція мети є сумою квадратів відхилень поведінки моделі від поведінки модельованого об'єкта на певній множині часових відліків, то, відповідно, й час, що необхідний для її обчислення, буде лінійно залежати від кількості дискретних значень. Тобто обчислювальна складність задачі буде пропорційна кількості вхідних даних, а не порядку створюваної моделі. Для побудови якісної макромоделі кількість дискретних значень, які використовуються для обчислення функції мети, буде доволі значною, а отже, значними будуть і затрати машинного часу на обчислення.

Практичне використання наведеної методики побудови макромоделей вимагає великих обчислювальних затрат, тому доцільним є розгляд питання про розпаралелення обчислювального процесу на декількох комп'ютерах чи процесорах одного комп'ютера.

Оптимізаційна задача, що виникає в процесі побудови дискретних динамічних маромоделей, характеризується складністю обчислення функції мети. З іншого боку, практично будь-який оптимізаційний алгоритм передбачає декілька (інколи десятки) розрахунків функції мети на одній ітерації, причому координати точок (значення вектора параметрів $\vec{\lambda}$), для яких відбувається цей розрахунок, можна визначити одразу на початку ітерації (як для детермінованих, так і для стохастичних оптимізаційних алгоритмів). Це дає можливість виконувати обчислення значення функції мети одночасно для різних точок.

Практична реалізація даного підходу може мати наступний вигляд (рис. 3).

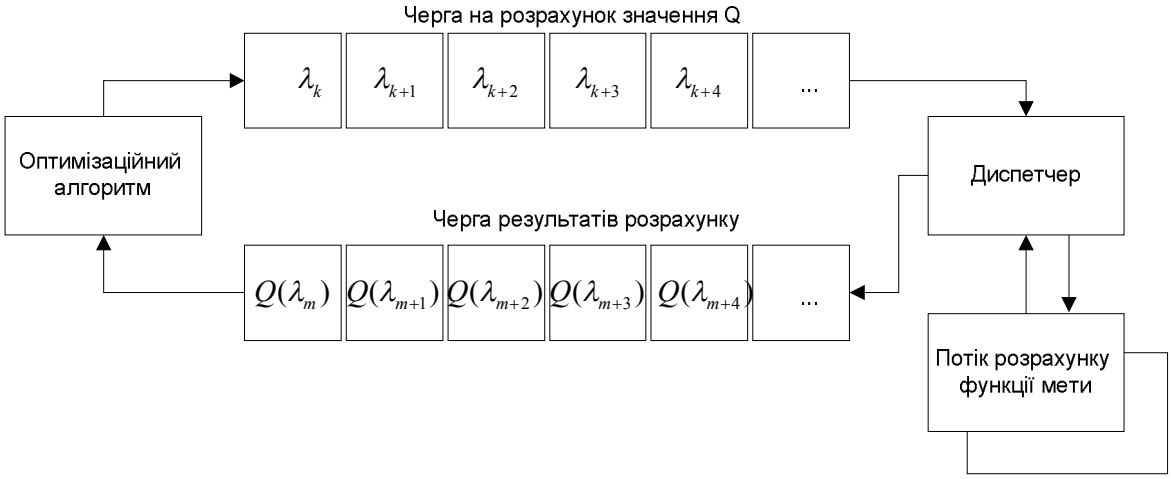

Рис. 3. Структура програмної реалізації оптимізаційного алгоритму з паралельним обчисленням функції мети

Оптимізаційний алгоритм в даній реалізації виконується в окремому потоці. Він формує список точок, для яких потрібно обчислити значення функції мети, і додає їх до черги. Після того, як усі необхідні точки для даної ітерації сформовані, потік переходить в стан очікування на результати. Коли значення функції мети для усіх доданих точок пораховані, протік оптимізаційного алгоритму прокидається, аналізує отримані значення і переходить до наступного кроку. Хоча при деяких практичних реалізаціях даного підходу не потрібно очікувати завершення повного обчислення усіх результатів, а проводити їх обробку в міру надходження від обчилювачів.

За саме обчислення функції мети для заданих точок відповідає множина потоків, які координуються диспетчером і можуть виконуватися як на тому ж комп'ютері, що й оптимізаційний алгоритм різними потоками, на різних ядрах процесора та на сопроцесорах, так і на інших комп'ютерах. Обмеженням тут виступає співвідношення часу, необхідного для обчислення функції мети в одній точці, і часу комунікації між диспетчером і потоком обчислення. Час комунікації зростає у випадку використання декількох комп'ютерів, проте для складних функцій мети, які обчислюються на основі десятка тисяч дискретних значень, комунікаційні затримки можуть виявитися незначними порівняно з часом, витраченим на розрахунок значення функції мети, навіть при комунікації через інтернет.

Перевагою даного підходу є можливість його повної автоматизації, що є важливим з огляду на доступність обчислювальної техніки і тенденції до зростання кількості процесорних ядер на одному комп'ютері, яка спостерігається останнім часом. Недоліком можна вважати незмінність або

навіть збільшення сумарних обчислювальних затрат, які необхідні для побудови макромоделі, хоча зі зростанням продуктивності сучасних комп'ютерів цей недолік не можна вважати суттєвим.

Як бачимо з постановки задачі, для її оптимального (в сенсі часових затрат) вирішення необхідно проводити паралельні обчислення функції мети (за єдиним алгоритмом) для великої кількості даних. Тобто найбільш правильним буде використати для даного підходу архітектуру обчислювальної системи типу SIMD (одиночний потік команд з багатьма потоками даних). Тому для практичної реалізації потрібно вибрати обчислювач саме даної категорії.

Ріст частот універсальних процесорів зупиняють фізичні обмеження і високе енергоспоживання, і збільшення їх продуктивності все частіше відбувається за рахунок розміщення декількох ядер в одному процесорі. Присутні зараз на ринку процесори містять лише до чотирьох ядер (подальше зростання не буде швидким) і призначені для звичайних програм, які використовують MIMD — множинний потік команд і даних. Кожне ядро працює окремо від інших, виконуючи різні інструкції для різних процесів.

Спеціалізовані векторні можливості появилися в універсальних процесорах, в першу чергу, через високі вимоги графічних програм. Саме тому для певних задач (виконання однотипних дій з різними даними) застосування GPU (Graphics Processing Unit) вигідніше ніж CPU (Central Processing Unit), тому що вони від початку призначені для таких задач.

Сучасні GPU - це багатопроцесорні багатоядерні системи SIMD-архітектури з достатньо високою (до 1 Тфлопс) піковою продуктивністю. Порівняно з традиційними архітектурами (наприклад, кластерами), вони мають порівняно низьку характеристику «ціна/продуктивність», що викликає зацікавлення використовувати GPU не тільки для обробки графічної інформації, але й для вирішення будь-яких обчислювальних задач [2]. Зокрема, до таких задач відносяться сортування великих масивів даних [3], розв'язок систем лінійних рівнянь [4], розв'язок рівнянь математичної фізики [5-7] та інші.

Наприклад, в процесорах NVIDIA, які побудовані на основі архітектури Tesla G80, основний блок — це мультипроцесор с 8-10 ядрами і сотнями ALU (Arithmetic logic unit) в цілому, декількома тисячами регістрів і невеликим об'ємом загальної пам'яті. Крім того, відеокарта має швидку глобальну пам'ять з доступом до неї всіх мультипроцесорів, локальну пам'ять в кожному мультипроцесорі, а також спеціальну пам'ять для констант.

Крім того, центральні процесори використовують SIMD (одна інструкція виконується над численними даними) блоки для векторних обчислень, а відеопроцесори застосовують SIMT (одна інструкція і кілька потоків) для скалярної обробки потоків. SIMT не вимагає, щоб розробник перетворював дані в вектори, і допускає довільні розгалуження в потоках.

Узагальнюючи обґрунтування вибору GPU для проведення паралельних обчислень до даної задачі, можна сказати, що, на відміну від сучасних універсальних CPU, відеопроцесори призначені для паралельних обчислень з великою кількістю арифметичних операцій.

Основою для ефективного використання потужності GPU в наукових та інших неграфічних обчисленнях є розпаралелювання алгоритмів на сотні виконавчих блоків, що є в відеопроцесорі. Задача обчислення функції мети в багатьох точках, як жодна інша пристосована для обчислення на відеопроцесорі, вимагає великих обчислювальних потужностей і тому зручна для паралельних обчислень. А використання декількох GPU, що в сучасних умовах є досить реальним і дешевим, дає ще більше обчислювальних потужностей для вирішення подібних задач.

Отже, на сьогоднішній день існує декілька типів загальнодоступних апаратних засобів для виконання паралельних обчислень – це кластерні системи, багатопроцесорні або багатоядерні системи і графічні карти GPU, які недавно стали використовуватися для обчислень загального призначення (GP GPU). Кластерні і багатопроцесорні системи достатньо дорогі, а графічні карти, навпаки, широко розповсюджені в настільних комп'ютерах і мають великий потенціал для паралельних векторних обчислень.

Програмування на графічних картах здійснюється за допомогою мов шейдерів, використовувати які для загальних обчислень не завжди зручно. Реалізовувати обчислення значень функції мети на графічній карті планується за допомогою технології NVidia CUDA, яка дозволяє писати програми для GPU на C++, або Python, який у світі почав широко використовуватися для модельних експериментів. Для тестових експериментів було вибрано, як базовий, персональний комп'ютер з центральним процесором Intel Core 2 Duo та об'ємом оперативної пам'яті 2 Гігібайти та графічний процесор NVIDIA GTS250 з 16 мультипроцесорами, кожен з яких має вісім ядер, та загальною пам'яттю 1 Гігабайт. Програмне забезпечення — CUDA Toolkit.

Висновки

Особливостями запропонованого методу розпаралелення у порівнянні з реалізаціями без розпаралелення є:

1) потік оптимізаційного алгоритму не обчислює значення функції мети самостійно, а обмежується визначенням координат точок, для яких слід її обчислити;

2) оптимізаційний алгоритм визначає координати одразу усіх точок, необхідних на даній ітерації, і лише після цього очікує на результати обчислення функції мети;

3) для даної задачі, з великою кількість арифметичних операцій, запропоновано використовувати GPU, який виконує однотипні операції над даними;

4) використання GPU, що в сучасних умовах є досить реальним і дешевим, дає великі обчислювальні потужності для вирішення задач ідентифікації параметрів дискретних динамічних макромоделей.

У подальшому планується дослідити залежність продуктивності від конфігурації ядра графічного процесора та різні алгоритми оптимізації.

1. General-Purpose Computation Using Graphics Hardware: [http://gpgpu.org/]
2. Purcell T.J., Donner C., Commarano M., Jensen H.W., Hanrahan P. Photon mapping on programmable graphic hardware // Proceeding of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware — Eurographics Association, 2003. — pp. 41-50.
3. Göddeke M., Strzodka R., Turek S. Accelerating Double Precision FEM Simulations with GPUs // Proceeding of ASIM 2005 — 18th Symposium on Simulation Technique — 2005. — pp. 139-144.
4. Hagen T.R., Henriksen M.O., Hjelmervik J.M., Lie K.-A. Using the graphic processor as a highperformance computational engine for solving system of hyperbolic conservation low // Geometric Modelling, Numerical Simulation, and Optimization: Applied Mathematics at SINTEF — Springer, 2007, pp. 211-264.
5. Hagen T.R., Hjelmervik J.M., Lie K.-A., Natvig J.R., Henriksen M.O. Visual simulation of shallow-water waves // Simulation Practice and Theory. Special Issue on Programmable Graphics Hardware, 13(9) — 2005. — pp. 716-726.
6. Hagen T.R., Lie K.-A., Natvig J.R. Solving the Euler equation on graphical processing units // Computational Science — ICCS 2006: 6th International Conference, Reading, UK, May 28-31, 2006, Proceedings, Part IV, volume 3994 of Lecture Notes in Computational Science (LNCS) — Springer Verlag, 2006. — pp. 220-227.
7. Hagen T.R., Lie K.-A., Natvig J.R. Solving the Euler equation on graphical processing units // Computational Science — ICCS 2006: 6th International Conference, Reading, UK, May 28-31, 2006, Proceedings, Part IV, volume 3994 of Lecture Notes in Computational Science (LNCS) — Springer Verlag, 2006. — pp. 220-227.
8. Заде Л., Дезоер Ч. Теория линейных систем. Метод пространства состояний. – М.: Наука, 1970.
9. Кунцевич В.М., Кунцевич А.В. Активная идентификация и управление при ограниченных шумах // Пр. міжн. конф. з управління “АВТОМАТИКА-2000”, Львів 11-15 вересня 2000: В 7-ми томах. – Львів: Держ. НДІ інформ. інфраструкт., 2000.- Т. 1. - С .7-13.
10. Кунцевич В.М., Лычак М.М. Об оптимальном и адаптивном управлении динамическими объектами в условиях неопределенности // Автоматика и телемеханика. – 1979. - №1.- С. 79 – 88.
11. Стахив П.Г. Анализ динамических режимов в электронных схемах с многополюсниками. – Львов: Высш. школа, 1988. – 154 с.
12. Стахів П.Г., Козак Ю.П. Побудова макромоделей електромеханічних компонент з використанням оптимізації // Технічна електродинаміка. – 2001. - №4. – С.33-36.
13. Стрейц В. Метод пространства состояний в теории дискретних линейных систем управления/ Пер. с англ. Под ред. Я.З. Цыпкина. – М.: Наука. Главная редакция физико-математической литературы, 1985. – 296.
14. Черноусько Ф. Л. Оценивание фазового состояния динамических систем. Метод эллипсоидов. – М.: Наука. Гл. ред. физ.-мат. лит., 1988. – 320 с.
15. Черноусько Ф.Л. Про оптимальне еліпсоїдальне оцінювання для динамічних систем, на які діють невизначені збурення // Кибернетика и системный анализ. – 2002. - № 2. – С. 85-95.