УДК 681.3.012:621.1

Г.Г. Швачич[1], С.О. Чернецький[2], О.Г. Холод[3]
[1]Національна металургійна академія України,
[2]Дніпропетровський національний університет імені Олеся Гончара,
[3]Дніпропетровський університет економіки та права

ДЕЯКІ АСПЕКТИ КЛАСТЕРНИХ ТЕХНОЛОГІЙ МОДЕЛЮВАННЯ ЗАДАЧ МОНТЕ–КАРЛО

Розглядаються кластерні технології моделювання задач Монте-Карло. Показано, що лише застосування сучасних суперпродуктивних систем дозволило по-новому реалізувати механізм відповідних розподілених обчислень. Приводяться схеми обчислень, які забезпечують збільшення продуктивності і швидкодії. Ефективність запропонованого підходу ілюструється порівняльним аналізом розв'язку деякого класу задач.

Рассматриваются кластерные технологии моделирования задач Монте-Карло. Показано, что только применение современных суперпродуктивных систем позволило по-новому реализовать механизм соответствующих распределенных вычислений. Приводятся схемы вычислений, которые обеспечивают увеличение продуктивности и скорости. Эффективность предложенного подхода иллюстрируется сравнительным анализом решения некоторого класса задач.

Clustering technologies of Monte-Carlo problems' modeling are considered. It is shown that using of modern superproduction systems only allowed to newly implementing the mechanism of relevant distribution calculations. We present computational scheme which provides productivity and speed increasing. The effectiveness of the proposed approach is illustrated by a comparative analysis of a certain class problems solution.

Ключові слова: метод Монте-Карло, кластер, паралельні обчислення, крайова задача.

Вступ. Серед різноманіття числових методів у математичних розрахунках останнім часом можна говорити про відродження таких методів, як методи Монте-Карло [1 – 4]. Ця назва об'єднує групу числових методів, основаних на одержанні великого числа реалізацій стохастичного процесу, який формується таким чином, щоб його ймовірнісні характеристики збігалися з аналогічними величинами розв'язуваної задачі. Методи Монте-Карло використовуються для

© *Г.Г. Швачич, С.О. Чернецький, О.Г. Холод, 2011*

розв'язування задач у галузях фізики, математики, економіки, оптимізації, теорії керування та ін. Вітчизняні праці про методи Монте-Карло з'явились у 1955 – 1956 роках. З того часу написано багато наукових праць, що описують застосування цього методу [5 – 10]. Навіть поверховий перегляд назв робіт дозволяє зробити висновок про застосовуваність методу Монте-Карло для розв'язку прикладних задач у багатьох галузях науки і техніки.

Відмітною рисою методів Монте-Карло вважається їх експериментальний характер. Для певності будемо називати методом Монте-Карло процедуру, що включає використання прийомів статистичної вибірки для наближеного розв'язку тієї чи іншої математичної або фізичної задачі.

Методи Монте-Карло суттєво впливали і впливають на розвиток методів обчислювальної математики, а при розв'язку певного класу задач успішно поєднуються з іншими обчислювальними методами і доповнюють їх. Їх застосування виправдане, в першу чергу, до тих задач, які допускають теоретико-ймовірнісний опис. Це пояснюється як природністю одержання відповіді з певною заданою ймовірністю в задачах, що мають імовірнісний зміст, так і суттєвим спрощенням процедури розв'язку. На даний час існують статистичні методи розв'язування для деяких класів диференціальних рівнянь у частинних похідних, інтегральних рівнянь, задач про власні значення і системи лінійних алгебраїчних рівнянь.

Серед інших обчислювальних методів метод Монте-Карло виділяється своєю простотою і універсальністю. Повільна збіжність є істотним недоліком методу, проте в даній статті будуть описані його обчислювальні модифікації, які забезпечують високий порядок збіжності при застосуванні певних припущень. Правда, обчислювальна процедура при цьому ускладнюється і цим наближається до інших процедур обчислювальної математики. Збіжність методу Монте-Карло – це збіжність за ймовірністю. Цю обставину навряд чи слід відносити до його недоліків, бо ймовірнісні методи великою мірою виправдовують себе в практичних додатках. Що ж до задач, які мають ймовірнісний опис, то при їх розв'язуванні збіжність за ймовірністю є навіть до певної міри природною.

Нарешті зауважимо, що точність обчислень методу дуже залежить від якості застосовуваного генератора псевдовипадкових чисел, а швидкість

обчислень визначається функцією, що описує аналізований процес і, звичайно ж, продуктивністю самого «обчислювача». На сьогодні тактова частота сучасних процесорів вже перейшла межу ГГц, а обсяг оперативної пам'яті персональних комп'ютерів вимірюється в гігабайтах. Зважаючи на те, що відповідний клас задач буде оброблятися на персональному обчислювальному кластері, продуктивність «обчислювача» не є більше стримуючим чинником (а скоріше, визначальним) для застосування чутливих до потужності «обчислювача» числових алгоритмів при розв'язуванні багатовимірних задач. Практичну ілюстрацію механізму роботи методу і деякі принципові особливості його реалізації будуть розглядатися на фоні розв'язку типових задач теплофізики.

Постановка задачі. Цілі і задачі досліджень. Завдання, на вирішення якого спрямовано представлене дослідження, полягає у розробці та дослідженні деяких аспектів кластерних технологій моделювання задач Монте-Карло. Повільна збіжність методу є його істотним недоліком, проте в даній статті необхідно розробити його розподілені обчислювальні модифікації, які забезпечують високий порядок збіжності. При цьому для прискорення процесу обчислень алгоритм паралельних обчислень необхідно представити у вигляді наступної схеми. Генератор випадкових чисел генерує кожному «обчислювачу» випадкове число. Таким чином формується пошукова послідовність. Проміжні розрахунки здійснювати незалежно на різних, окремо взятих лезах кластера – «обчислювачах», а результати вже повинні оброблятися на окремо узятому MASTER – лезі – «аналізаторі». Такий підхід дозволяє позбавитися від неодмінної присутності між генератором випадкових чисел і «обчислювачем» маршрутизатора-комунікатора. Виграш продуктивності при цьому оцінити експериментальним шляхом. Таким чином, ставиться задача розробки відповідних ефективних алгоритмів розподіленого моделювання.

Практичну ілюстрацію механізму роботи розробленого підходу і деякі принципові особливості його реалізації необхідно розглянути на прикладі розв'язку крайових задач і задач з початковими умовами для лінійних диференціальних рівнянь. Це пояснюється тим, що зазначений клас задач є однією із цікавих і перспективних областей додатків методу Монте-Карло.

Вивести і проаналізувати аналітичні співвідношення, які дають статистичну оцінку досліджуваних величин, які застосовуються для наближеного вирішення досліджуваного класу задач.

Показати, що розроблені алгоритми стійкі для будь-яких вхідних даних, мають максимальну паралельну форму, і, отже, мінімальний можливий час реалізації на паралельних обчислювальних системах.

Виконати порівняльний аналіз розв'язків поставленої задачі числово-аналітичним методом і методом статистичних випробувань. Це дозволить якнайкраще оцінити похибку обчислень методом Монте-Карло. Для реалізації такого підходу ставиться задача розробки числово-аналітичного алгоритму для розв'язку досліджуваного класу задач.

Викладення основного матеріалу досліджень

Персональний обчислювальний кластер як ефективний засіб технології проведення складних розрахунків. Застосування паралельних обчислювальних систем (*ПОС*) є стратегічним напрямом розвитку обчислювальної техніки. Ця обставина викликана не тільки принциповим обмеженням максимально можливої швидкодії звичайних послідовних ЕОМ, але й практично постійним існуванням обчислювальних задач, для вирішення яких можливостей існуючих засобів обчислювальної техніки завжди виявляється недостатньо. До таких задач відносяться, наприклад, чисельне моделювання процесів гідродинаміки і металургійної теплофізики [11,12], задачі розпізнавання образів, оптимізаційні задачі з великим числом параметрів, моделювання клімату, генна інженерія, проектування інтегральних схем, аналіз забруднення навколишнього середовища [13], рішення широкого кола багатовимірних нестаціонарних задач [14] і т. д.

У наші часи істотний інтерес до побудови багатопроцесорних паралельних обчислювальних систем (кластерів) визначається застосуванням стандартних загальнодоступних технологій і компонентів. Це зумовлено рядом чинників. Відзначимо основні з них. По-перше, зростання, відповідно до потреб ринку, продуктивності таких стандартних мережевих технологій як *Ethernet* (характерна послідовним підвищенням швидкості передачі інформації до 10, 100, 1000 Мбіт/с, застосуванням комутаторів замість модулів з розділюваним середовищем) дозволяє розглядати їх як комунікаційне середовище для багатопроцесорних

обчислювальних систем. По-друге, дуже важливим чинником стало збільшення популярності вільно поширюваної операційної системи *Linux*. Дана операційна система спочатку позиціонувалась як варіант *UNIX* для платформ на базі архітектури *Intel*, але досить швидко з'явилися версії для інших популярних мікропроцесорів, у тому числі й для лідерів за продуктивністю протягом останніх років – мікропроцесорів *Alpha*.

З урахуванням економічних реалій нашої країни використання систем, побудованих на базі стандартних технологій, стає більш ніж актуальним. Причому залежно від особливостей задач і величини бюджету проект системи може мати всілякі варіанти конфігурації. Найбільш доступним варіантом можна вважати стандартні материнські плати для процесорів *Intel Pentium III* і мережеві адаптери *Fast Ethernet*. Вузли кластера об'єднуються між собою за допомогою комутатора *Fast Ethernet*, розрахованого на відповідне число портів. Кількість вузлів у системі та їх конфігурація залежить від вимог, що висуваються до обчислювальних ресурсів конкретними завданнями, і залежать від фінансових можливостей користувачів.

Крім того, гостра конкуренція між виробниками комп'ютерної техніки широкого вжитку приводить до того, що ситуація з цінами на ринку комплектуючих змінюється досить динамічно, особливо у зв'язку з випуском виробів нових моделей. Враховуючи також широкий асортимент сучасної електронної продукції, можна з упевненістю констатувати, що при використанні стандартних комплектуючих можлива побудова потужних обчислювальних систем загального призначення у вельми стислі терміни з максимально повним урахуванням потреб і можливостей різних користувачів.

Останнім часом розвиток суперпродуктивних обчислювальних систем визначається застосуванням персональних обчислювальних кластерів (ПОК). Персональний обчислювальний кластер – це обчислювальна система, всі вузли якої розміщені в одному корпусі. На думку багатьох авторів [15 – 17], реального перелому в опануванні технологій паралельних обчислень можна досягти розвитком багатопроцесорних обчислювальних систем *MPP*-архітектури саме шляхом запровадження ПОК. Таким чином, створюється фундамент піраміди апаратних засобів для їхнього застосування в технології паралельних обчислень у вигляді ПОК, аналогічний вже наявному фундаменту піраміди апаратури для традиційних технологій послідовних

обчислень у вигляді ПЕОМ. Відзначимо, що технології паралельного програмування і паралельних обчислень можна вважати результатом широкого застосування ПОК. Початок масового використання ПЕОМ припав на другу половину 1980-х років, у той же час, за нашими прогнозами, середину першого десятиліття XXI-го ст. можна буде вважати початком поширення ПОК у формі мультипроцесорних обчислювальних систем з розподіленою пам'яттю. Сфера застосування таких систем – опанування технологій паралельного програмування, створення і налагодження паралельних програм, модельна прогонка розробленого ПЗ, набуття навичок керування багатопроцесорними системами та ін.

У сучасних умовах кластерні системи конструюються з обчислювальних вузлів на базі стандартних процесорів, з'єднаних високошвидкісною системною мережею (інтерконектом), а також, як правило, допоміжною і сервісною мережами. Проте, останнім часом лідери-виробники апаратних засобів комп'ютерної техніки пропонують свій формфактор, зокрема, компанії *IBM, Verari, LinuxNetworx* та ін. мають у своєму розпорядженні кластерні рішення, побудовані на базі лезових технологій (так званих блейд-технологій). Дійсно, формуючи кластер на базі серверів-лез, користувач фактично отримує готове шасі, оснащене необхідним інтерконектом і засобами керування. Досить розглянути розробки від компаній *HP BladeSystem c7000, IBM BladeCenter, SUN Blade 6000, Dell PoweEdge1955*, щоб виявити цілий ряд можливостей для побудови кластерів на їх основі. Блейд-системи компактні й зручні в обслуговуванні, їхнє конструктивне рішення дозволяє зручно формувати необхідну конфігурацію, проте воно трохи дорожче в реалізації порівняно з традиційним підходом.

Розглянемо деякі особливості такої реалізації кластерних систем. Кожен вузол працює під керівництвом своєї копії стандартної операційної системи, в більшості випадків це *Linux*. Склад і потужність вузлів можуть бути різними в рамках одного кластера, проте частіше будуються однорідні кластери. Вибір конкретного комунікаційного середовища (інтерконекту) зумовлюється багатьма чинниками: особливостями розв'язуваних задач, доступним фінансуванням та ін.

На рис. 1 представлена блок-схема персонального обчислювального кластера. Він містить один майстер-вузол (*PM*000) і п'ять обчислювальних *slave*-вузлів (*PN*001, *PN*002, *PN*003, *PN*004, *PN*005), три керовані

комутатори (*SW*1, *SW*2, *SW*3), проміжні буфери пам'яті комутаторів, реконфігуровану мережу для обміну даних між обчислювальними вузлами, віртуальні локальні мережі (*VS*123, *VS*23, *VS*012, *VS*022, *VS*032, *VS*042, *VS*052, *VS*013, *VS*023, *VS*033, *VS*043, *VS*053), механізм резервування ключових компонентів, а також передбачає мережене завантаження вузлів. У майстер-вузлі та *slave*-вузлах застосовуються одні й ті самі комплектуючі (материнські плати, процесори, інтегровані мережеві плати *Fast Ethernet*, зовнішні мережеві плати *Gigabit Ethernet*). Зокрема, майстер-вузли обладнано додатково жорсткими дисками (*HDD*), *CD/DVD*, дисководами (*FDD*).

Особливості функціонування персонального обчислювального кластера полягають у наступному. Після подачі енергоживлення на блок *ATX*1 і надходження зовнішнього сигналу *PUSK* з панелі П00 починається запуск та ініціалізація роботи майстер-вузла модуля багатопроцесорної системи. Безпосередньо завантаження ОС може здійснюватися або з жорсткого диска, або з *CD/DVD*-пристрою. Після завантаження операційної системи запускається конфігураційний скрипт, який налаштовує роботу *DHCP*-сервера, крім того, тут визначається кількість обчислювальних вузлів системи, в разі потреби налаштовується доступ у середовище Інтернет або зовнішню мережу, при цьому зазначаються основні налаштування й параметри. Оскільки активізація режиму *Power on After Power Fail / Former-Sts* і подача живлення на відповідний блок (*ATX*2) запускає всі обчислювальні *slave*-вузли, то й завантаження операційної системи обчислювальних вузлів відбувається групами, без перевантаження першої мережі. Після завантаження всіх обчислювальних вузлів кластера завершується робота скрипту і кластер готовий до виконання паралельних обчислень.

Майстер-вузол (*PM*00) через комутатор *SW*1 забезпечує спрямування потоку даних керування, діагностики та прийому / передачі умов вирішуваних завдань до *slave*- вузлів. *Slave*-вузли відповідно до алгоритму вирішуваного завдання і виконуваних процесів реалізують режим необхідних обчислень. Обмін даними між обчислювальними вузлами винесено до окремої мережі, організованої за допомогою керованих комутаторів *SW*2 ─ *SW*3. Для досягнення максимальної ефективності кластерної системи здійснюється процес реконфігурації структури другої мережі залежно від класу вирішуваних завдань. Прийом / передача даних у *slave*-вузлах відбувається за допомогою проміжних буферів пам'яті.

Результати обчислень через першу комунікаційну мережу, за допомогою керованого комутатора *SW*1, передаються майстер-вузлу (*PM*000), де здійснюється необхідний режим видачі й обробки даних.

В якості конструктива обрано єдиний корпус, що являє собою осередок обчислювальної шафи. Це пов'язане з тим, що, з одного боку, при необхідності можна декілька ПОК розміщати в єдиному корпусі, а з іншого –при такому підході забезпечується компактність, успішне охолодження й легкий доступ до гнізд і елементів плат, які налагоджуються. ПОК включає вертикальне, паралельне одне стосовно другого, розташування системних плат, що відповідає ідеї «Blade» - серверів.

Персональний обчислювальний кластер має такі розміри: ширина 19', висота 10,9', глибина 9'. Вага пристрою приблизно 7 кг.

Після завантаження операційної системи доступ до ПОК можна отримати за стандартними мережними протоколми (*telnet, ssh, rsh*), як до звичайного ПК. Дякуючи чому для організації суперкомп'ютера на основі робочого ПК і ПОК необхідний лише мережний зв'язок між ними, який може бути організований як через локальну LAN так і через глобальну мережу internet.

Відзначимо деякі особливості зазначеного персонального обчислювального кластера (рис. 1), які дозволяють виділити його серед аналогічних багатопроцесорних систем:

по-перше, висока швидкодія обчислень під час вирішення сильнозв’язаних завдань, забезпечення високошвидкісного доступу до пам'яті вузлів кластера й обмін даними між ними, знизивши завантаження каналу, що проходить між вузлами кластера, за рахунок формування окремої обчислювальної мережі й реалізації механізмів *channel bonding* і *VLAN*;

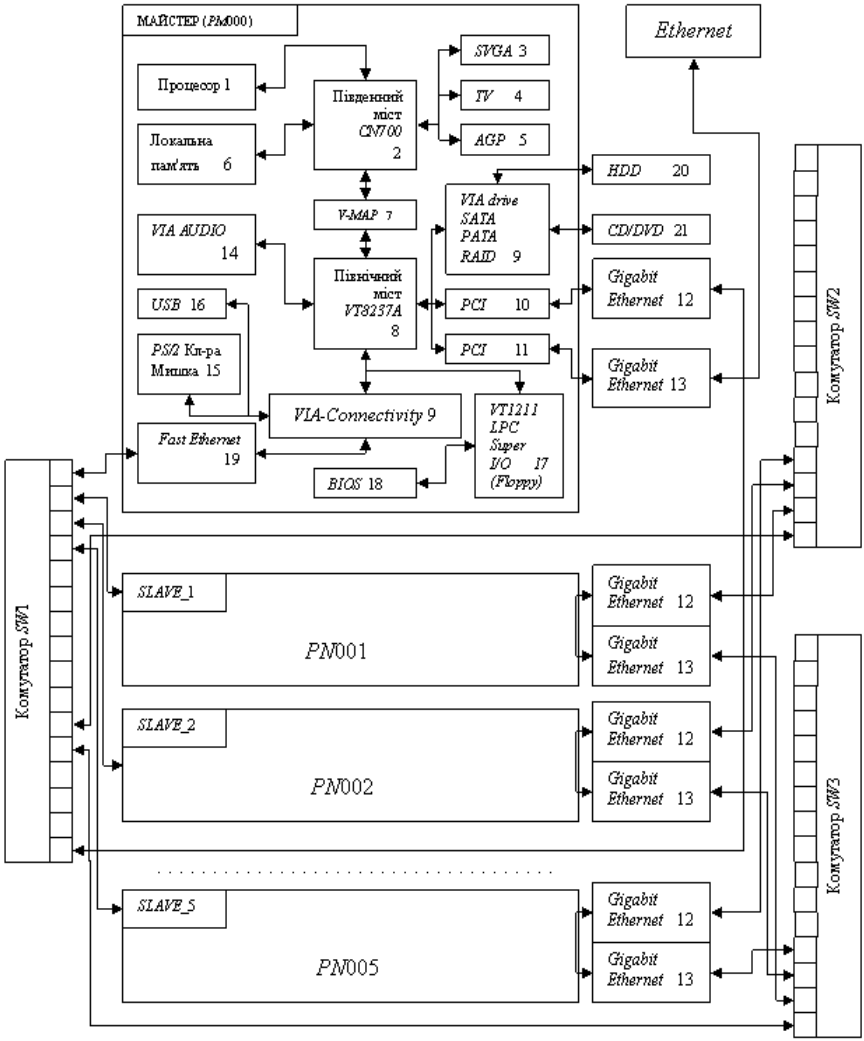


Рис. 1. Блок-схема персонального обчислювального кластера

по-друге, за рахунок проміжного буфера пам'яті з'явилася можливість «розвантажити» центральний процесор *CPU* в моменти передачі та прийому пакетів між вузлами кластера, що зумовлює підвищення продуктивності обчислювальної системи в цілому;

по-третє, висока ефективність кластерної системи, яка досягається за рахунок адаптації структуру її мережі до вирішення кожного конкретного типу завдань;

по-четверте, за рахунок модульного принципу побудови спростити проектування, нарощування або заміну кластерних вузлів, що вийшли з ладу, а також роботу й експлуатацію системи в цілому;

по-п'яте, за рахунок мережевого завантаження модуля багатопроцесорної системи підвищеної готовності, а також введення механізму резервування ключових компонентів модуля істотно зменшити кількість її компонентів і тим самим значно підвищити надійність функціонування;

по-шосте, забезпечення здатності до перенесення програмного забезпечення на інші кластерні системи з метою виконання подібних розрахунків.

Особливості реалізації паралельних обчислень методом Монте-Карло. Значний обсяг програмного забезпечення, розробленого раніше для послідовної (однопроцесорної) обчислювальної техніки, не має необхідного запасу паралелізму, і спроби його розпаралелювання на рівні окремих циклів, як правило, не мають добрих результатів. Продуктивність коду залишається низькою. Серед різноманіття числових методів у математичних розрахунках останнім часом окреме місце посідає якраз метод Монте-Карло. Проте як було раніше зазначено застосування цього методу вимагає великих обчислювальних потужностей, що до певного часу стримувало процес його використання. Зауважимо, що за допомогою методу Монте-Карло можна безпосередньо знайти наближений розв'язок задачі в єдиній фіксованій точці, не знаючи розв'язку задачі для основних точок сіткової області. Цією обставиною метод Монте-Карло, наприклад, у застосуванні до задачі Діріхле, різко відрізняється від звичайних стандартних способів розв'язку.

Спрощену схему обчислень методом Монте-Карло зображено на рис. 2.

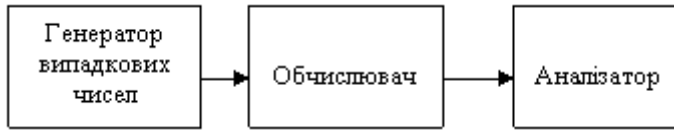


Рис. 2. Схема обчислень методом Монте-Карло

Застосування цього методу дає можливість по-новому розглянути ідею розпаралелювання обчислень і застосування кластерних технологій обчислення. Справді, проміжні розрахунки у методі Монте-Карло можуть здійснюватися незалежно на різних, окремо взятих, лезах кластера – «обчислювачах», а результати оброблятися на якомусь окремо взятому *master*-лезі – «аналізаторі» [18]. У зв'язку з цим для даної кластерної системи алгоритм паралельних обчислень можна уявити у вигляді схеми, поданої на рис. 3.

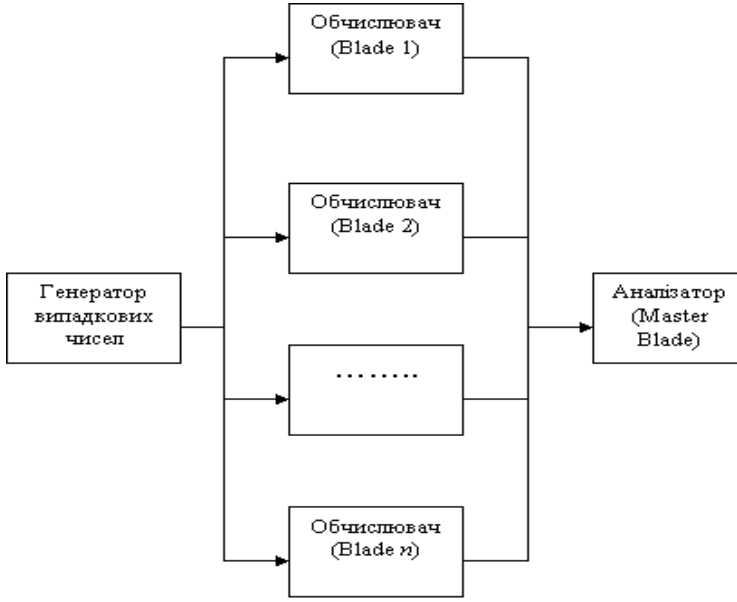


Рис. 3. Алгоритм паралельних обчислень методом Монте-Карло

Згідно із такою схемою, один генератор випадкових чисел видає кожному «обчислювачу» випадкове число, яке вже було згенеровано. Така схема має істотний недолік, що знижує продуктивність паралельної кластерної системи. Інформація постійно передається комутаційними каналами, які мають відповідну латентність. Зазначені обставини і будуть знижувати продуктивність та швидкодію системи в цілому. Звичайно, ефективність такої системи залежатиме і від якості генератора випадкових чисел, оскільки існує досить висока ймовірність того, що серед послідовності випадкових чисел виникатимуть повторення або досить близькі за величиною значення. Хоча можна зауважити, що зазначені обставини істотно не вплинуть на допустиму похибку обчислень. Досвід експлуатації персонального обчислювального кластера дозволив удосконалити схему обчислень, зображену на рис. 3.

На рис. 4 подано модифікований алгоритм обчислень методом Монте-Карло. Тут кожен обчислювач має власний генератор випадкових чисел. Це дозволяє уникнути неодмінної присутності між генератором випадкових чисел і «обчислювачем» маршрутизатора-комунікатора. Очевидно, що таке рішення дозволяє прискорити процес обчислень. Виграш у продуктивності можна оцінити експериментальним шляхом.

Таким чином, алгоритми Монте-Карло є стійкими по відношенню до будь-яких вхідних даних, мають максимальну паралельну форму і, отже, мінімальний можливий час реалізації на паралельних обчислювальних пристроях. Якщо можна призначити один процесор на один вузол розрахунку, то стає можливим проведення розрахунків у всіх вузлах сіткової області паралельно й одночасно.

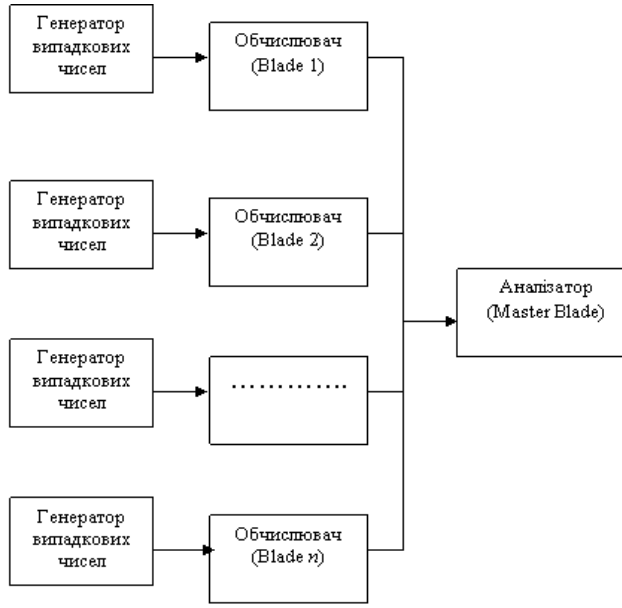


Рис. 4. Модифікований алгоритм паралельних обчислень методом Монте-Карло

Аналіз проблеми блукань і розв'язування крайових задач. Крайові задачі й задачі з початковими умовами для лінійних диференціальних рівнянь є однією з найцікавіших сфер застосування методу Монте-Карло. Зв'язок між цими задачами був відомий давно [5 – 9]. Проте можливість застосування цього зв'язку для фактичного відшукання результатів з'явилася тільки завдяки розвитку обчислювальних машин. Застосування цього методу дає можливість по-новому сприйняти ідею розпаралелювання обчислень і застосування кластерних технологій обчислення.

Щоб пояснити основну ідею методу, розглянемо задачу Діріхле для рівняння Лапласа. Нехай маємо певну однозв'язну область *G*, на межі якої задано функцію *f(Q)*. Потрібно знайти таку функцію *U(P),* яка всередині даної області *G* задовольняє рівняння Лапласа, тобто

$$\Delta U = 0, \tag{1}$$

а на межі області P набуває таких заданих значень:

$$U_{|\Gamma} = f(Q). \tag{2}$$

Зазвичай цю задачу зводять до певної кінцево-різницевої схеми. Область G вкривається вузлами квадратної сітки. У внутрішніх вузлах шукають значення функції $U(P)$, виходячи з такої системи рівнянь

$$U(P) = \frac{1}{4}[U(P_1) + U(P_2) + U(P_3) + U(P_4)]. \tag{3}$$

Тут символами $\{P_1, P_2, P_3, P_4\}$ позначають чотири вузли, сусідні з внутрішнім вузлом P, вони лежать в області G або на її межі.

Розглянемо пов'язану з цією системою теоретико-ймовірнісну схему. Уявімо собі частинку, яка змушена пересуватися по вузлах сітки з такими цілочисловими координатами (i, j) на площині:

$$\left.\begin{array}{l} x_i = x_0 + i\eta,\ y_j = y_0 + j\eta \\ (i, j = 0, \pm 1, \pm 2, \ldots) \end{array}\right\},$$

при цьому

$$\Delta x_i = x_{i+1} - x_i,\ \Delta y_j = y_{j+1} - y_j.$$

Припустимо, що сітка S_η складається з внутрішніх і граничних вузлів, у яких задано граничні умови першого роду. Граничні вузли являють собою сукупність лінійних точок $M_{pq}(x_p, y_q)$, яка апроксимує криволінійну межу Γ області G з точністю до η. Частинка M здійснює рівномірне випадкове блукання серед вузлів сітки (3). Зокрема, перебуваючи у внутрішньому вузлі $M_{i0,j0}$ сітки S_η, ця частинка за один перехід, з однією і тією самою ймовірністю, яка дорівнює ¼, може переміститися в один з сусідніх вузлів, зокрема, або в $M_{i-1,j}(x_{i-\eta}, y_j)$, це буде крок уліво, або в $M_{i+1,j}(x_{i+\eta}, y_j)$ – крок управо, або в $M_{i,j-1}(x_i, y_{j-\eta})$ – крок униз, або $M_{i,j+1}(x_i, y_{j+\eta})$ – крок угору. Причому, кожен такий одиничний перехід є абсолютно випадковим і не залежить від положення частинки та її попередніх переміщень (історії блукань). Вважатимемо, що блукання частинки M закінчується, як тільки вона потрапляє на межу Γ_η; в цьому випадку межа Γ_η є «поглинальним екраном». Можна довести [5], що

блукання точки M через скінченне число кроків закінчується саме на цій межі.

Якщо частинка M почала своє блукання з фіксованої внутрішньої точки $M_{i0,j0}$ на сітці S_η , то скінченну сукупність її послідовних положень можна записати таким чином:

$$M_{i0j0,} M_{i1,j1}, \ldots, M_{iS,jS},$$

причому

$$M_{i_k j_k} \in \Gamma\eta\,(k = 0,1,\ldots,S-1).$$

Тут вираз $M_{i_k j_k} \in \Gamma\eta$ відображає траєкторію частинки (з кількістю кроків S), яку прийнято називати історією блукання.

Рівномірне випадкове блукання частинки можна організувати за допомогою рівномірно розподіленої послідовності однорозрядних випадкових чисел [5; 7 – 10], що набувають таких значень:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Для цього, наприклад, досить проводити розіграш, тобто випадкову вибірку з чисел {0 – 9}, дотримуючись інструкції, поданої в табл. 1.

Таблиця 1

Визначення кроку частинки залежно від випадкового числа

Випадкове число	Характер переміщення
0 або 4	$\Delta x_i = \eta$ (крок управо)
1 або 5	$\Delta Y_Y = \eta$ (крок угору)
2 або 6	$\Delta\Delta x_i = -\eta$ (крок уліво)
3 або 7	$\Delta\Delta Y_Y = -\eta$ (крок униз)

Випадкові числа беруться з готових таблиць або визначаються генератором псевдовипадкових чисел [7]. Останній спосіб набув широкого застосування під час роботи на ПОК, оскільки він не дозволяє занадто завантажувати пам'ять системи. Частинка, що почала своє випадкове блукання з внутрішнього вузла $M_{i0,j0}$, після першого кроку з імовірністю, яка дорівнює ¼, потрапляє в один з чотирьох сусідніх вузлів, а саме:

I. $M_{i,j}, M_{i-1,j}, \ldots, ;$

II. $M_{i,j}, M_{i+1,j}, \ldots, ;$
III. $M_{i,j}, M_{i,j-1}, \ldots, ;$
IV. $M_{i,j}, M_{i,j+1}, \ldots .$

За формулою повної ймовірності встановлюємо, що:

$$P(i,j,p,q)=\frac{1}{4}P(i-1,j,p,q)+\frac{1}{4}P(i+1,j,p,q)+ \\ +\frac{1}{4}P(i,j-1,p,q)+\frac{1}{4}P(i,j+1,p,q). \tag{4}$$

Звідси, перемножуючи обидві частини рівності (4) на граничні значення γ_{pq} і підсумовуючи за всіма можливими значеннями p і q, одержимо, що

$$\vartheta_{ij}=\frac{1}{4}(\vartheta_{i-1,j}+\vartheta_{i+1,j}+\vartheta_{i,j-1}+\vartheta_{i,j+1}). \tag{5}$$

Величини ϑ_{ij} допускають експериментальне визначення, для цього треба замінити математичне очікування ϑ_{ij} емпіричним математичним очікуванням, тоді відповідний вираз набуде такого вигляду:

$$U_{ij}=\frac{1}{N}\sum_{k=1}^{w}\varphi(x_p^{(k)}y_q^{(k)}). \tag{6}$$

Формула (6) дає статистичну оцінку величини $U_{i,j}$ і може бути застосована до наближеного розв'язку задачі Діріхле. Проілюструємо розроблений алгоритм, розв'язуючи конкретні приклади.

Обчислювальні експерименти. Обчислювальні експерименти виконувалися на основі застосування персонального обчислювального кластера, блок-схема якого наведена на рис. 1, при цьому методика відповідних розрахунків проводилася з урахуванням розробленого підходу. Розглянемо характерні приклади обчислень.

Приклад 1. Методом Монте-Карло знайти величину $U(2,2)$, якщо

$$\Delta U(x,y)=0, \text{ в області } G\ \{0\le x\le 4;\ 0\le y\le 4\}, \tag{7}$$

а

$$\left.\begin{aligned} &U(x,0)=0, \quad 0\le x\le 4;\\ &U(4,y)=y, \quad 0\le y\le 4;\\ &U(x,4)=x, \quad 0\le x\le 4;\\ &U(0,y)=0, \quad 0\le y\le 4. \end{aligned}\right\} \qquad (8)$$

Розв'язок. Для квадрата G з межею Γ побудуємо квадратну сітку S, в якій крок η =1. Далі розглядаємо серію рівномірних блукань частинки серед вузлів сітки S_η, виходячи з початкового положення (2,2). Блукання закінчуються на межі Γ, у граничних вузлах сітки $G_к$, заданих умовами (8). Переміщення частинки відбувалося відповідно до описаної вище інструкції (табл. 2), причому появу випадкових чисел 8 і 9 вважаємо стоянням частинки на місці.

У табл. 2 наведено траєкторії для 20 історій двовимірного випадкового блукання, коли число $N = 20$.

На підставі формули (6) можемо встановити, що:

$$U(2,2)=\frac{1}{20}\sum_k \varphi(x_p^{(k)} y_q^{(k)} = \frac{1}{20}\cdot 20 = 1.$$

Зауважимо, що в даному випадку відомий точний розв'язок задачі Діріхле (7), (8), а саме:

$$U(x,y)=\frac{xy}{4}.$$

Тому

$$U(2,2)=\frac{2\cdot 2}{4}=1.$$

Таким чином, за допомогою методу статистичних випробувань одержали точне значення величини $U(2,2)$.

Приклад 2. Розглянемо задачу про температурне поле вугільної адіабатичної пластини [19]. Температурне поле $T(x,y)$ має задовольняти таке рівняння:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0, \tag{9}$$

а також систему довільно вибраних граничних значень температури:

$$T = \begin{cases} F_1(x) \text{ уздовж границі } y = \ell, \\ F_2(x) \text{ уздовж границі } y = 0, \\ G_1(y) \text{ уздовж границі } x = L, \\ G_2(y) \text{ уздовж границі } x = 0, \end{cases} \tag{10}$$

Потрібно визначити температурне поле пластини, якщо $F_1(x) = F_2(x) = 1$ і $G_1(y) = G_2(y) = 0$, а геометричні розміри пластини $l = 1$ і $L = 2l$.

Таблиця 2

Траєкторії блукання робочої точки

№ блукання, k	Траєкторія блукання	Значення функції $u(x,y)$ у точці виходу на межу області G
1	(2,2) > (2,3) > (2,2) > (2,1) > > (3,1) > (3,2) > (3,1) > (3,2) > > (2,2) > (2,3) > (2,3) > (2,2) > > (2,1) > (2,0);	0
2	(2,2) > (2,3) > (3,3) > (3,2) > (4,2)	2
3	(2,2) > (2,3) > (2,2) > (2,3) > (2,4)	2
4	(2,2) > (1,2) > (1,2) > (0,2);	0
5	(2,2) > (2,3) > (2,4);	2
6	(2,2) > (2,1) > (2,0);	0
7	(2,2) > (1,2) > (2,2) > (3,2) > (3,1)	0

	> (3,2) > (2,2) > (1,2) > (0,2);	
8	(2,2) > (1,2) > (0,2);	0
9	(2,2) > (2,1) > (2,2) > (3,2) > (3,3) > (3,3) > (2,3) > (1,3) > (0,3);	0
10	(2,2) > (1,2) > (0,2);	0
11	(2,2) > (2,2) > (2,2) > (2,1) > > (2,2) > (3,2) > (3,1) > (3,1) > (4,1)	1
12	(2,2) > (2,2) > (2,1) > (2,1) > (4,1);	0
13	(2,2) > (2,1) > (3,1) > (3,0) ;	0
14	(2,2) > (3,2) > (4,2);	2
15	(2,2) > (2,3) > (2,4);	2
16	(2,2) > (2,3) > (2,3) > (1,3) > (0,3);	0
17	(2,2) > (3,2) > (4,2);	2
18	(2,2) > (3,2) > (3,1) > (2,1) > > (2,2) > (3,2) > (4,2);	3
19	(2,2) > (3,2) > (4,2);	2
20	(2,2) > (2,3) > (2,3) > (2,3) > (2,4);	2
	$\sum$	20

Виконаємо порівняльний аналіз розв'язків поставленої задачі числово-аналітичним методом і методом статистичних випробувань.

Отже, прямокутна область заданого розміру вкривається сітковими вузлами таким чином:

$$x_i = x_0 + i\eta,\ y_j = y_0 + j\eta\ ,$$

(11)

при цьому

$$\left.\begin{aligned} Dx1 = x_{i+1} - x_i = \frac{2}{2m_x},\ m_x = 10 \\ Dy1 = y_{j+1} - y_j = \frac{2}{2m_y},\ m_y = 5 \end{aligned}\right\},\ \eta = 0{,}1.$$

Доцільність такого нормування вузлів приводить до їх рахунковості через цілочислові значення параметрів m_x і m_y. Маємо таку потужність рахунковості:

$$i = \overline{1,2m_x - 1}, \quad j = \overline{1,2m_y - 1}.$$

Граничні поверхні виражаються параметрично:

$$x_0 = 0, \;\; y_0 = 0, \;\; x_{2m_x} = 2, \;\; y_{2m_y} = 1,$$

при цьому, x_{m_x} y_{m_y} і (x_{m_x}, y_{m_y}) відповідають серединним поверхням і центральному вузлу. Замість незалежних змінних x та y введемо нормовані одиницею, тобто

$$\left.\begin{aligned} \varepsilon_x &= \frac{x - x_i}{x_{i+1} - x_i} \in [-1, +1], \\ \varepsilon_y &= \frac{y - y_j}{y_{j+1} - y_j} \in [-1, +1]. \end{aligned}\right\} \tag{12}$$

Тоді рівняння (9) для рівномірно розподілених вузлів ($x_{i+1} - x_i = x_i - x_{i-1}$, $y_{j+1} - y_j = y_j - y_{j-1}$) виявиться інваріантним відносно сіткових вузлів, а саме:

$$\frac{\partial^2 T(\varepsilon_x, \varepsilon_y)}{\partial \varepsilon_x^2} + \frac{\partial^2 T(\varepsilon_x, \varepsilon_y)}{\partial \varepsilon_y^2} = 0. \tag{13}$$

Подібний підхід допускає можливість будувати алгоритми з урахуванням апріорної інформації про шуканий розв'язок. Припустимо, що шуканий розв'язок рівняння (9) належить до класу аналітичних функцій. Отже, стає можливим існування шуканої функції у вигляді рядів Тейлора:

$$T_{j+\varepsilon_y,1}(x,y) = \sum_{n=0}^{\infty} \varepsilon_y^n T_{j,n+1}(\varepsilon_x) \tag{14}$$

або

$$T_{i+\varepsilon_x,1}(x,y)=\sum_{n=0}^{\infty}\varepsilon_x^n T_{i,n+1}(\varepsilon_y). \tag{15}$$

Після підстановки (14), (15) у рівняння (13) одержимо сукупність диференціальних рівнянь такого вигляду:

$$(n+1)(n+2)T_{j,n+3}(\varepsilon_x)=-T''_{j,n+1}(\varepsilon_x) \tag{16}$$

або

$$(n+1)(n+2)T_{i,n+3}(\varepsilon_x)=-T''_{i,n+1}(\varepsilon_y). \tag{17}$$

Легко помітити, що тейлорівські компоненти (16), (17) можуть бути виражені через дані Коші $\{T_{j,1}(\varepsilon_x),T_{j,2}(\varepsilon_x)\}$, $\{T_{i,1}(\varepsilon_y),T_{i,2}(\varepsilon_y)\}$ та їх похідні за незалежними змінними ξ_x, ξ_y. Тоді для різних значень *n* (наприклад: 0, 1, 2, 3) відповідні рівняння можна записати таким чином:

$$\left.\begin{aligned}
T_{J,3}(\varepsilon_x)&=-\frac{1}{2!}T''_{j,1}(\varepsilon_x),\\
T_{J,4}(\varepsilon_x)&=-\frac{1}{3!}T''_{j,2}(\varepsilon_x),\\
T_{J,5}(\varepsilon_x)&=+\frac{1}{4!}T^{(4)}_{j,1}(\varepsilon_x),\\
T_{J,6}(\varepsilon_x)&=+\frac{1}{5!}T^{(4)}_{j,2}(\varepsilon_x),
\end{aligned}\right\} \tag{18}$$

і т. д.

Отже, замість ряду Тейлора (14) ми маємо локальний розв'язок задачі Коші відповідно до вузлів $(j=\overline{1,2m_y-1})$, а саме:

$$T_{y+\varepsilon_{y,1}}(x,y)=\sum_{n=0}^{\infty}(-1)^n\left[\frac{\varepsilon_y^{2n}}{(2n)!}T^{(2n)}_{y,1}(\varepsilon_x)+\frac{\varepsilon_y^{2n+1}}{(2n+1)!}T^{(2n)}_{y,2}(\varepsilon_x)\right], \tag{19}$$

при цьому дані Коші $\{T_{J,1}(\varepsilon_x), T_{J,2}(\varepsilon_x)\}$ являють собою невідомі функції змінної ε_x.

Довизначаючи розв'язок (19) граничними умовами Діріхле, і вважаючи, що $\varepsilon_x = \pm 1$, отримаємо розділений розв'язок задачі Коші в такому вигляді:

$$\sum_{n=0}^{\infty}(-1)^n \frac{1}{(2n)!} T_{J,1}^{(2n)}(\varepsilon_x) = \frac{1}{2}\left[T_{j+1,1}(\varepsilon_x) + T_{j-1,1}(\varepsilon_x)\right], \quad (20)$$

або

$$\sum_{n=0}^{\infty}(-1)^n \frac{1}{(2n+1)!} T_{j,2}^{(2n)}(\varepsilon_x) = \frac{1}{2}\left[T_{j+1,1}(\varepsilon_x) - T_{j-1,1}(\varepsilon_x)\right], \quad (21)$$

при цьому $j = \overline{1, 2m_y - 1}$, а $T_{0,1}(\varepsilon_x), T_{2m_{y,1}}(\varepsilon_x)$ – відомі граничні функції першого роду.

Диференціюючи (19) за ε_y і розділяючи дані Коші, одержимо, що:

$$\sum_{n=0}^{\infty}(-1)^n \frac{1}{(2n)!} T_{j,1}^{(2n)}(\varepsilon_x) = \frac{1}{2}\left[T_{j+1,2}(\varepsilon_x) + T_{j-1,2}(\varepsilon_x)\right] \quad (22)$$

або

$$\sum_{n=0}^{\infty}(-1)^n \frac{1}{(2n-1)!} T_{j,1}^{(2n)}(\varepsilon_x) = \frac{1}{2}\left[T_{j+1,2}(\varepsilon_x) - T_{j-1,2}(\varepsilon_x)\right], \quad (23)$$

де $\{T_{0,2}(\varepsilon_x), T_{2m_{y,1}}(\varepsilon_x)\}$ – відомі граничні функції другого роду.

Таким чином, редукуючи нескінченні ряди (20) − (23) з урахуванням, що n прямує до ($n = M \in Z$), ми одержуємо частинні математичні моделі з довільним порядком точності. При цьому, якщо $M = 1$, то маємо такі звичайні кінцево-різницеві схеми:

– для задачі Дірихле:

$$T_1(i,j) = \frac{1}{4}[T_1(i-1,j) + T_1(i+1,j) + T_1(i,j-1) + T_1(i,j+1)]; \quad (24)$$

– для задачі Неймана:

$$T_2(i,j)=\frac{1}{4}[T_2(i-1,j)+T_2(i+1,j)+T_2(i,j-1)+T_2(i,j+1)]\,. \quad (25)$$

Отримані співвідношення (24) та (25) можна застосовувати і для методу статистичних випробувань. Так, просте випадкове блукання частинки прямокутною сіткою легко поширюється і на задачі Неймана, причому $T_2(i,j)$ у (25) являє собою градієнту функцію. Можливі також інші модифікації процесу вибірки. Припустимо, наприклад, що вирішено змінити ймовірності переходу частинки в певній точці. Ця зміна може бути уточнена, якщо прийняти в ролі вагових коефіцієнтів схему p_α / r_α, за якою p_α являє собою гіпотетичну ймовірність одержання заданого результату на будь-якій стадії, а використаний процес вибірки реалізує цей результат з імовірністю r_α. Подібну процедуру «блукання» частинки можна застосовувати на кожному етапі випадкового процесу за умови, що обчислюються добутки відповідних вагових коефіцієнтів до кінця кожної історії.

Вельми неприємною рисою немодифікованого випадкового блукання, на якому базувався попередній розгляд, є повільна збіжність. Із самої природи випадкового процесу випливає, що потрібно зробити велику кількість кроків, щоб кудись дійти. За таких обставин зазвичай доцільно розглянути використання спеціальних методів вибірки стосовно окремих груп [7].

Для розв'язання задачі (9), (10) граничні умови (10) конкретизуємо такими вихідними даними:

$$T[\,j,0\,]=T[\,j,2m_x\,]=0,$$
$$T[\,0,i\,]=T[\,2m_x,i\,]=1, \quad (26)$$

при цьому $j=\overline{1,2m_y-1}, i=\overline{1,2m_x-1}$. Переміщення частинки визначалося відповідно до поданої вище ілюстрації (табл. 2). Для організації випадкових блукань використовуємо рівномірно розподілену послідовність випадкових чисел, підібрану за допомогою ПОК. Результати розрахунків у колонках $[\,j,1\,]$, $[\,j,m_x\,]$ при різних значеннях

величини N оброблялися у вигляді відносної похибки, що має такий вигляд:

$$\beta[j,i] = \frac{\left|T_t[j,i] - T_p[j,i]\right|}{T_t[j,i]} \cdot 100\%, \qquad (27)$$

при цьому $T_t[j,i]$ відповідає точному розв'язку, $T_p[j,i]$ визначають за схемою методу Монте-Карло. Результати розрахунків зведено в табл. 3.

Із порівняльного аналізу результатів моделювання випливає, що зі збільшенням числа N відносна похибка зменшується. У прикутових вузлах (1, 1), (m_j,1) похибка не зменшується через близькість граничних вузлів (0, 0) і (m_j, 0), де функція має розрив першого роду відповідно до умови (26).

Таблиця 3

Відносна похибка результатів розв'язку задач методом Монте-Карло при різних значеннях блукань робочої частинки N

Номер блукання	N = 1 000		N = 10 000		Аналітичний розв'язок
	β, %	$\overline{m}_x$	β, %	$\overline{m}_x$	$T(x,y)$
(1,1)	6,332	0,5273	2,359	0,5076	0,4959
(2,1)	1,314	0,5938	0,614	0,5897	0,5861
(3,1)	1,121	0,6858	0,236	0,6798	0,6782
(4,1)	0,967	0,7725	0,209	0,7667	0,7651
(5,1)	0,913	0,8948	0,101	0,8876	0,8867
(6,1)	0,967	0,7725	0,209	0,7667	0,7651
(7,1)	1,121	0,6858	0,236	0,6798	0,6782
(8,1)	1,314	0,5938	0,614	0,5897	0,5861
(9,1)	6,332	0,5273	2,359	0,5076	0,4959

Висновки. У наведеній статті набув подальшого розвитку процес математичного моделювання прикладних задач на основі

використання персонального обчислювального кластера. Досвід експлуатації перших паралельних систем показав, що для їх ефективної роботи потрібно радикально змінювати структуру числових методів. У зв'язку з цим розроблено відповідні розподілені алгоритми, виявлено і показано особливості моделювання прикладних задач на основі застосування багатопроцесорних систем.

Останнім часом можна говорити про відродження методу Монте-Карло. Це пояснюється тим, що цей метод ідеально підходить до кластерної системи. Причому, чим більше процесорів буде в кластері, тим ефективніше буде розв'язуватися задача. Метод Монте-Карло справив і продовжує справляти істотний вплив на розвиток методів обчислювальної математики. Його застосування виправдане, в першу чергу, до розв'язку таких задач, які допускають теоретико-ймовірнісний опис. Це пояснюється як природністю одержання відповіді з певною заданою ймовірністю у задачах з імовірнісним змістом, так і суттєвим спрощенням процедури розв'язку. За таких обставин у даній статті висвітлено кластерні проблеми застосування методу Монте-Карло до розв'язування багатовимірних задач.

Повільна збіжність методу є його недоліком. Проте в проведених дослідженнях показано, що формування вибіркових випадкових чисел стосовно окремих груп дозволяє істотно підвищити точність методу.

Крім того, показано, що метод Монте-Карло досить вдало пристосований до розв'язування багатовимірних задач. Наприклад, застосовуючи звичайний метод розв'язку систем лінійних алгебраїчних рівнянь для обчислення одного невідомого, треба визначити також і решту. У даному способі цього робити не потрібно, кожного разу визначається тільки одна необхідна координата розв'язуваної системи.

Крайові задачі та задачі з початковими умовами для лінійних диференціальних рівнянь є однією з найцікавіших сфер застосування методу Монте-Карло, а саме це стало можливим тільки завдяки розвитку кластерних обчислювальних систем. З цих причин у статті наведено приклади розв'язку задач Неймана та Діріхле за допомогою методу Монте-Карло.

Застосування цього методу дає можливість по-новому подивитись на ідею розпаралелювання обчислень та використання кластерних технологій обчислення. У статті запропоновано модифікований алгоритм паралельних обчислень за методом Монте-Карло. Тут кожен обчислювач

має власний генератор випадкових чисел. При цьому проміжні розрахунки здійснюються незалежно на різних, окремо взятих лезах кластера – «обчислювачах», а результати вже обробляються на якомусь окремо взятому *master*-лезі – «аналізаторі». Ця обставина дозволяє позбутися неодмінної присутності маршрутизатора-комунікатора між генератором випадкових чисел та «обчислювачем». Очевидно, що таке рішення дозволяє прискорити процес обчислень.

Показано, що розроблені паралельні алгоритми методу Монте-Карло стійкі для будь-яких вхідних даних, мають максимальну паралельну форму і, отже, мінімальний можливий час реалізації на паралельних обчислювальних пристроях. Якщо можна призначити один процесор на один вузол розрахунку, то стає можливим проведення обчислень у всіх вузлах сіткової області паралельно й одночасно.

Бібліографічні посилання

1. **Михайлов Г.А.** Численное статистическое моделирование. Методы Монте-Карло / Г.А. Михайлов, А.В. Войтишек. – М., 2006. – 368 с.

2. **Михайлов Г. А.** Оптимизация весовых методов Монте-Карло по вспомогательным переменным / Г. А. Михайлов, И. Н. Медведев // Сиб. матем. журн. – 2004. – № 45. – С. 399 – 409.

3. Інтернет-ресурс http://ru.wikipedia.org/wiki/метод_Монте-Карло.

4. Інтернет-ресурс http://www.riskglossary.com/link/monte_carlo_method.htm.

5. **Ермаков С.М**. Метод Монте-Карло и смежные вопросы / С.М. Ермаков. – М., 1971. – 471 с.

6. **Соболь И.М**. Метод Монте-Карло / И.М. Соболь. – М., 1968. – 64 с.

7. **Браун Дж**. Методы Монте-Карло / Дж. Браун // Современная математика для инженеров; под ред. Э.Ф. Беккенбаха. – М., 1958. – 500 с.

8. **Китов Н.А**. Электронные цифровые машины и программирование / Н.А. Китов, Н.А. Криницкий. – М., 1959. – 572 с.

9. **Демедович Б.П**. Основы вычислительной математики / Б.П. Демедович, И.А. Марон. – М., 1963. – 660 с.

10. **Березин И.С.** Методы вычислений / И.С. Березин, Н.П. Шибков. – Т. 1. – М., – 1966. – 632 с.

11**. Коздоба Л. А.** Вычислительная теплофизика / Л.А. Коздоба. – К., 1992. – 224 с.

12. **Иващенко В.П**. Параллельные вычисления и прикладные задачи металлургической теплофизики / В.П. Иващенко, Г.Г. Швачич, А.А. Шмукин // Системні технології. Регіональний збірник наукових праць. – Вип. 3(56). – Т. 1. – 2008. – С. 123 – 138.

13. **Швачич Г.Г**. К вопросу конструирования параллельных вычислений при моделировании задач идентификации параметров окружающей среды / Г.Г. Швачич // Математичне моделювання. – 2006. – № 2 (14). – С. 23 – 34.

14. **Швачич Г.Г.** О параллельных компьютерных технологиях кластерного типа решения многомерных нестационарных задач / Г.Г. Швачич // Materiály IV mezinárodní vědecko- praktická konference [«Vědecký potenciál světa - 2007»]. – D. 7. – Technické vědy. Matematika. Fyzika. Moderní informační technologie. Výstavba a architektura. – Praha: Publishing House «Education and Science» s.r.o. – P. 35 – 43.

15. **Воеводин Вл. В**. Вычислительное дело и кластерные системы / Вл.В. Воеводин, С.А. Жуматий. – М., 2007. – 150 с.

16. **Баканов В.М**. Персональный вычислительный кластер как недостающее звено в технологии проведения сложных технологических расчетов / В.М. Баканов // Метизы. – 2006. – 2 (12). – С. 33 – 36.

17. **Shvachych G.G**. Prospects of construction highly–productive computers systems on the base of standard technologies / G.G. Shvachych // IV Intrenational Conference «Strategy of Quality in Indastry and Education»; May 30 – June 6, 2008, Varna; Bulgaria. – Proceedings. – V. 2. – P. 815 – 819.

18. **Швачич Г. Г.** Некоторые особенности применения метода Монте - Карло для распределенного моделирования многомерных нестационарных задач / Г.Г. Швачич // Сборник научных трудов по материалам международной конференции «Научные основы внедрения новых технологий в эпоху Нового Возрождения». – 2009. – С. 340 – 341.

19. **Шнейдер П**. Инженерные проблемы теплопроводности / П. Шнейдер; пер. с англ. – М., 1960. – 478 с.

Надійшла до редколегії 25.01.11