**МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ»**

Л.П. Фельдман, І.А. Назарова

ПАРАЛЕЛЬНІ ОДНОКРОКОВІ МЕТОДИ ЧИСЕЛЬНОГО РОЗВ'ЯЗАННЯ ЗАДАЧІ КОШІ

Монографія

**Донецьк
ДВНЗ «ДонНТУ»
2011**

УДК 517.62(075.8)
Ф-39

Рекомендовано до друку Вченою Радою ДВНЗ «Донецький національний технічний університет» (протокол № 9 від 22 грудня 2010 року)

Рецензенти: *А.О. Каргін*, доктор технічних наук, професор, завідувач кафедри комп'ютерних технологій, декан фізичного факультету Донецького національного університету;

В.А. Святний, доктор технічних наук, професор, завідувач кафедри комп'ютерної інженерії Донецького національного технічного університету.

Фельдман Л.П., Назарова І.А.

Ф-39 Паралельні однокрокові методи чисельного розв'язання задачі Коші**: монографія / Л.П. Фельдман, І.А. Назарова. – Донецьк: «ДВНЗ» ДонНТУ, 2011. – 185 с.: іл.**

ISBN 978-966-377-106-9

Монографія присвячена паралельним однокроковим методам чисельного розв'язання задачі Коші для систем звичайних диференційних рівнянь. У ній представлено паралельні методи на базі явних і неявних чисельних схем типа Рунге-Кутти та багатоточкові, блокові методи із вбудованими засобами оцінки локальної апостеріорної похибки: дублювання шагу, технологія локальної екстраполяція Річардсона та вкладені пари.

Для студентів, що отримують освіту в галузі знань 0501 «Інформатика та обчислювальна техніка», напрямів підготовки 050101 «Комп'ютерні науки», 050102 «Комп'ютерна інженерія», 050103 «Програмна інженерія». Монографію може бути рекомендовано магістрам і аспірантам усіх спеціальностей, які вивчають сучасні інформаційні технології в межах дисциплін «Чисельні методи в інформатиці», «Високопродуктивні обчислення».

УДК 517.62(075.8)
ISBN 978-966-377-106-9

ВСТУП

Застосування високопродуктивних паралельних обчислювальних систем (ОС) є одним з основних напрямків розвитку сучасної комп'ютерної науки. Ця обставина викликана не тільки принциповим обмеженням максимально можливої швидкодії звичайних послідовних машин, а й постійною появою нових обчислювальних задач, для розв’язання яких можливостей існуючих засобів обчислювальної техніки завжди виявляється недостатньо. Моделювання реальних багатовимірних динамічних процесів, що описуються системами звичайних диференціальних рівнянь (СЗДР), і являє собою клас задач, при реалізації яких використання паралельних суперкомп'ютерів не лише виправдане, але й необхідне. Про це свідчить відомий список "великий виклик", в якому такі задачі займають одне з провідних місць [1].

В останні роки істотно зросла пікова продуктивність паралельних обчислювальних систем, радикально змінилася технологічна база та архітектура. Але, як і раніше, головною перешкодою до впровадження практично всіх паралельних архітектур є відсутність ефективних прикладних паралельних методів. Як підсумок, проблема підвищення ефективності функціонування паралельних обчислювальних систем, незважаючи на наявні успіхи, далека до повного вирішення, як в теоретичному, так і в практичному відношенні.

Найбільш складною є задача визначення оптимальної алгоритмічної та структурної організації паралельних обчислень с метою досягнення достатніх характеристик ефективності [2-3]. На сьогодні не викликає сумнівів, що реалізувати потенційні можливості паралельних комп'ютерів можливо перш за все на основі проведення цілеспрямованої теоретичної роботи з побудови нових та адаптації існуючих методів розв’язання складних науково-технічних завдань.

Великий внесок у розробку теоретико-математичних основ розпаралелювання обчислень внесли Воєводін В.В., Воєводін Вл. В. [3-9],

Гергель В.П. [10-11], Gupta A., Kumar D. [12-14], Забродін А.В. [15-17]. Методи підвищення ефективності спеціалізованих паралельних обчислювальних систем досліджувалися в роботах Малиновського Б.М. [18-21], Боюна В.П. [20-23], Корнєєва В.Д. [24-25], Каляєва І.А. [26-27]. Незважаючи на результати великих досліджень в галузі паралельних обчислень, роботи в цьому напрямку не втрачають своєї значущості та потребують подальшого розвитку у зв'язку з масовим поширенням паралельних комп'ютерів.

Перераховані вище проблеми визначили спрямованість монографії, що присвячена розробці та обґрунтуванню паралельних методів розв’язання широкого класу науково-технічних і комплексних стратегічних динамічних задач із зосередженими параметрами. Актуальність роботи обумовлюється широким впровадженням високопродуктивних паралельних комп'ютерів на підприємствах і в організаціях України.

У першій главі монографії проведено огляд сучасних архітектур та класів високопродуктивних паралельних обчислювальних систем, описано базові топологічні структури, наведено основні принципи і методику розпаралелювання, виконано аналіз динамічних характеристик якості паралельних обчислень, проаналізовано сучасні методи розв’язання динамічних задач, які описуються СЗДР.

У другій главі запропоновані та дослідженні явні методи чисельного розв’язання СЗДР з оцінкою локальної апостеріорної похибки. Розроблені методи орієнтовані на використання у паралельних системах SIMD-, MIMD-, CLUSTER-архітектур з розподіленою пам’яттю та різними топологіями з’єднання процесорів. У главі використано спосіб розпаралелювання, що базується на об’єднанні ієрархічної декомпозиційної методики та математичного апарату графів впливу. Досліджено масштабування паралельних методів із використанням апарату ізоефективного аналізу.

Третя глава присвячена питанням розробки, обґрунтування та дослідження ефективності паралельних методів на основі неявних

однокрокових схем з контролем локальної похибки. У главі запропоновано і теоретично обґрунтовано нові паралельні методи розв'язання ЗДР з використанням неявних однокрокових різницевих схем: k-точковий метод з правилом дублювання кроку; вкладені блокові методи; екстраполяційні блокові методи на основі технології Річардсона. Проведено порівняльний аналіз неявних однокрокових методів розв'язання нелінійної задачі Коші на основі блокових k-точкових і повністю неявних методів Рунге-Кутти. Здійснено узагальнення методів, що реалізують різницеві k-точкові однокрокові блокові схеми для розв'язання систем ЗДР.

У четвертій главі монографії запропоновані спеціальні паралельні методи інтегрування лінійних динамічних задач з постійними коефіцієнтами. Отримані обчислювальні схеми інтегрування, що відповідають паралельним експоненціальним методам розв'язання лінійної задачі з контролем похибки на кроці. Для прискорення найбільш ресурсоємної операції експоненціального методу, матричного добутку, запропоновано паралельний алгоритм на основі комбінації швидкого рекурсивного та систоличного алгоритмів, а також модифікацію систоличного алгоритму, яка знімає обмеження на порядок перемножуваних матриць.

Дослідження, що покладені в основу монографії, проводились Фельдманом Л.П. та Назаровою І.А. на кафедрі прикладної математики та інформатики ДВНЗ «Донецький національний технічний університет» у рамках виконання науково-дослідних робіт за держбюджетними темами фундаментальних досліджень, затверджених МОН України [28-29]. Для проведення моделювання, досліджень та експериментів використовувалися пакет *Mathematica* (*Wolfram Research Inc*.) [30-31] та стандарт MPI (*Message Passing Interface*) для мови C++ [30-36]. Тестування алгоритмів проводилося з використанням тестів, розроблених у НДОЦ МДУ [37]. Результати досліджень, що викладені у монографії, було апробовано на міжнародних конференціях та семінарах, авторами опубліковано 29 печатних робіт [38-66].

ГЛАВА І

СУЧАСНІ ВИСОКОПРОДУКТИВНІ ОБЧИСЛЮВАЛЬНІ СИСТЕМИ ТА ПАРАЛЕЛЬНІ МЕТОДИ РОЗВ'ЯЗАННЯ СИСТЕМ ЗВИЧАЙНИХ ДИФЕРЕНЦІЙНИХ РІВНЯНЬ

1.1. Основні класи сучасних паралельних комп'ютерів

Сучасний етап розвитку обчислювальної техніки характеризується застосуванням величезної кількості високопродуктивних паралельних комп’ютерів. В них використовуються різні апаратні складові: процесори (Intel, IBM, AMD, HP, NEC, Cray), мережеві карти (Ethernet, Myrinet, Infiniband, SCI). Вони функціонують під управлінням різних операційних систем (Unix , Linux, Windows) та реалізують різне прикладне програмне забезпечення. Велика розмаїтість паралельних систем спричинила необхідність введення для них ефективної системи класифікації, що представляє можливість диференціації істотно різних обчислювальних систем, а також володіє простотою і практичною доцільністю [66].

Однією з перших і найбільш поширених схем класифікації архітектур обчислювальних систем є систематика М.Флінна [67-68]. В її основу покладено опис роботи комп'ютера з потоками виконуваних команд і оброблюваних даних (рис. 1.1).

У результаті такого підходу розрізняють наступні основні типи паралельних систем:

1) SISD (Single Instruction, Single Data) – поодинокий потік команд і поодинокий потік даних;

2) SIMD (Single Instruction, Multiple Data) – поодинокий потік команд і множинний потік даних;

3) MISD (Multiple Instruction, Single Data) – множинний потік команд і поодинокий потік даних;

4) MIMD (Multiple Instruction, Multiple Data) – множинний потік команд і множинний потік даних.

SISD Single Instruction, Single Data Поодинокий потік команд, поодинокий потік даних	**SIMD** Single Instruction, Multiple Data Поодинокий потік команд, множинний потік даних
MISD Multiple Instruction, Single Data Множинний потік команд, поодинокий потік даних	**MIMD** Multiple Instruction, Multiple Data Множинний потік команд, множинний потік даних

Рис. 1.1. Класифікація архітектур обчислювальних систем за Флінном

До SISD-класу відносяться, перш за все, класичні послідовні машини, або інакше, машини фон-неймановского типу, наприклад, PDP-11 або VAX 11/780 (рис. 1.2). У таких машинах є тільки один потік команд, всі команди обробляються послідовно одна за другою і кожна команда ініціює одну операцію з одним потоком даних. Не має значення той факт, що для збільшення швидкості обробки команд і швидкості виконання арифметичних операцій може застосовуватися конвеєрна обробка. У такому випадку в цей клас потраплять і такі системи, як CRAY-1, CYBER 205, машини сімейства FACOM VP і багато інших.

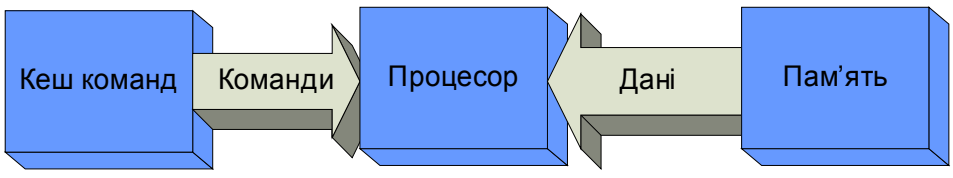


Рис. 1.2. Структура обчислювальних систем класу SISD

У SIMD-архітектурах (рис. 1.3) зберігається один потік команд, що включає, на відміну від попереднього класу, векторні команди.

Це дозволяє виконувати одну арифметичну операцію відразу над багатьма даними – елементами вектора. Спосіб виконання векторних операцій не обмовляється, тому обробка елементів вектора може проводитися або процесорною матрицею, як у ILLIAC IV, або за допомогою конвеєра, як, наприклад, у машині CRAY-1.

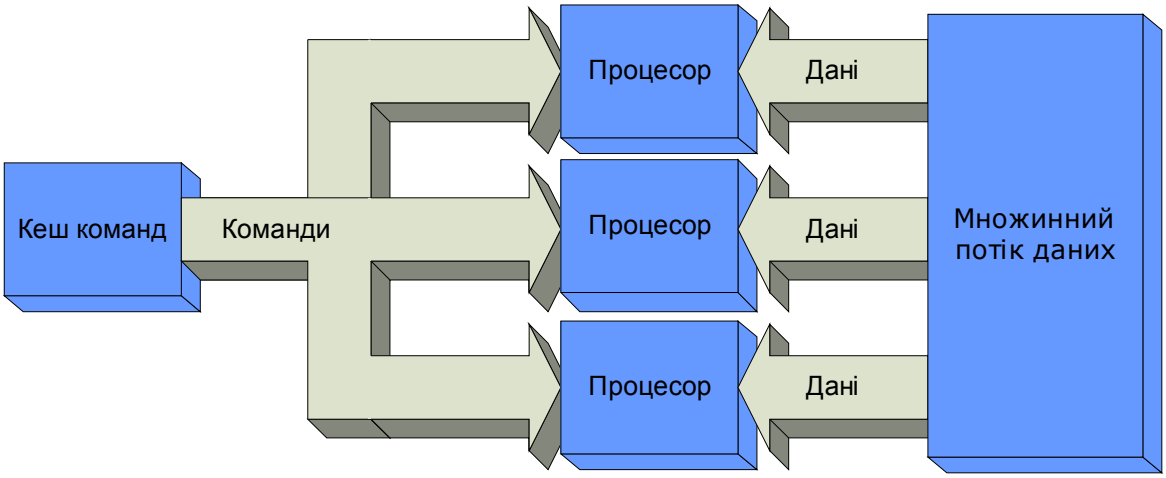


Рис. 1.3. Структура обчислювальних систем класу SIMD

Безперечними представниками класу SIMD вважаються матриці процесорів: ILLIAC IV, ICL DAP, Goodyear Aerospace MPP, Connection Machine 1 і т.п. У таких системах єдиний керуючий пристрій контролює безліч процесорних елементів. Кожен процесорний елемент одержує від пристрою керування в кожен фіксований момент часу однакову команду і виконує її над своїми локальними даними.

Визначення класу MISD має на увазі наявність в архітектурі багатьох процесорів, що обробляють один і той же потік даних. Проте ні Флінн, ні інші фахівці в області архітектури комп'ютерів до цих пір не змогли представити переконливий приклад реально існуючої обчислювальної системи, побудованої на даному принципі. Ряд дослідників [3] відносять конвеєрні машини до даного класу, однак це не знайшло остаточного визнання в науковому співтоваристві (рис. 1.4.).

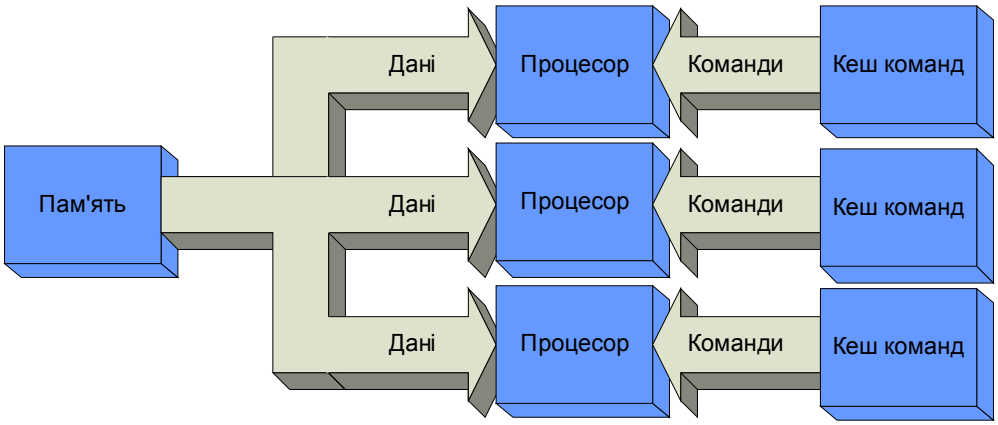


Рис. 1.4. Структура обчислювальних систем класу MISD

MIMD-архітектура (рис. 1.5) припускає, що в обчислювальній системі є кілька пристроїв обробки команд, об'єднаних в єдиний комплекс і працюючих кожне зі своїм потоком команд і даних. Клас MIMD надзвичайно широкий, оскільки включає в себе всілякі мультипроцесорні системи: Cm *, C.mmp, CRAY Y-MP, Denelcor HEP, BBN Butterfly, Intel Paragon, CRAY T3D і багато інших.

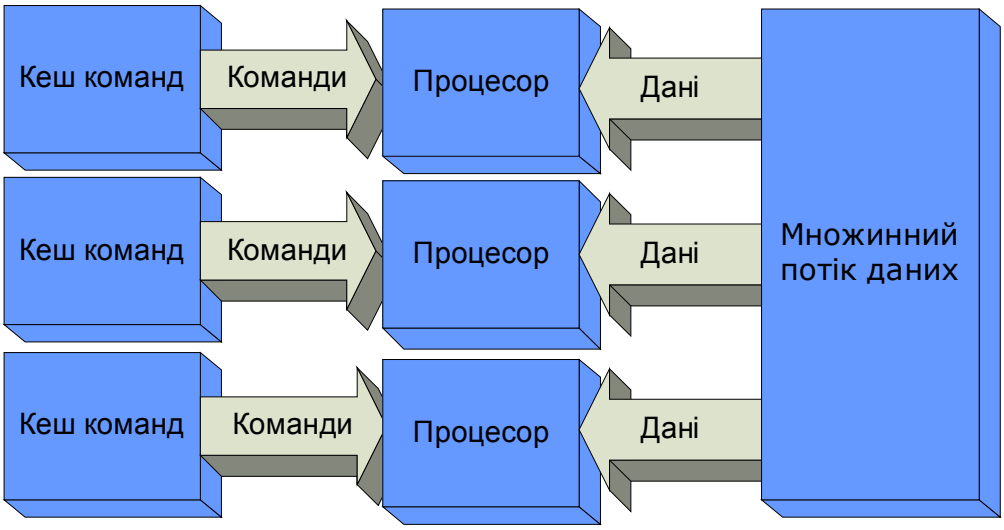


Рис. 1.5. Структура обчислювальних систем класу MIMD

Схема класифікації за Флінном до теперішнього часу є самою використовуваною при початковій характеристиці того чи іншого комп'ютера. Однак видно й явні недоліки. Зокрема, деякі архітектури, наприклад dataflow і векторно-конвеєрні машини, чітко не вписуються в цю класифікацію.

Інший недолік – це надмірна заповненість класу MIMD. Як результат, багатьма дослідниками робилися неодноразові спроби деталізації цієї систематики [69-76]. Так, MIMD-архітектури далі класифікуються у залежності від фізичного способу організації пам'яті:

1) комп'ютери з розподіленою пам'яттю;

2) комп'ютери з загальною пам'яттю;

3) комп'ютери з віртуальною загальною пам'яттю.

Деякі автори [11,69] обчислювальні системи з загальною пам'яттю називають мультипроцесорами, а системи з розподіленою пам'яттю – мультикомп'ютерами. У відповідності з типом доступу до пам'яті вводяться основні класи паралельних комп'ютерів: SMP, MPP, PVP, NUMA-архітектури та кластери.

Симетричні мультипроцесорні системи (SMP) [77-79] зазвичай складаються з декількох однакових процесорів і масиву загальної пам'яті. Всі процесори мають рівні можливості щодо доступу до єдиного адресного простору, такі ОС ще називають системами із однорідним доступом до пам'яті (*Uniform Memory Access – UMA*). Дана архітектура не є добре масштабованою і відмовостійкою. Прикладами SMP-систем можуть служити HP 9000 V-class, N-class; SMP-сервери, робочі станції на базі процесорів Intel.

Масивно-паралельні системи (MPP) [78-79] складаються із декількох однорідних обчислювальних вузлів, як правило, RISC-архітектури. Кожен такий вузол має свою локальну пам'ять, один або декілька центральних процесорів і іноді жорсткий диск, причому прямий доступ до пам'яті інших вузлів неможливий (*No Remote Memory Access – NORMA*). Вузли зазвичай пов'язані через певне комунікаційне середовище, в якості якого використовується або спеціальна високошвидкісна мережа, або комутатор. Доступ до віддаленої пам'яті реалізується в рамках моделі передачі повідомлень із процесором, якому належить адресуєма пам'ять на основі бібліотек MPI (*Message Passing Interface*) або PVM (*Parallel Virtual Machine*).

Системи масового паралелізму добре масштабуються, загальне число процесорів може досягати декількох тисяч, в них відсутня проблема когерентності кеш-пам'яті. Основний недолік таких систем полягає в складності організації обміну інформацією між процесорами. Прикладами MPP-систем служать: IBM RS/6000 SP2, Intel PARAGON, CRAY T3E, комп'ютери серії МВС.

Системи з неоднорідним доступом до пам'яті (*NUMA – Non Uniform Memory Access*) становлять щось середнє між SMP та MPP. У NUMA пам'ять фізично розподілена, але є логічно загальнодоступною. Підтримується єдиний адресний простір, апаратно підтримується доступ до віддаленої пам'яті. Масштабованість NUMA-систем обмежується обсягом адресного простору та можливостями операційної системи. Найбільш відомі NUMA-системи: SGI Origin 2000, Sun HPC 10000, IBM / Sequent NUMA-Q 2000, SNI RM600.

Загальною ознакою паралельних векторних систем (PVP) є наявність спеціальних векторно-конвеєрних процесорів, в яких передбачені команди однотипної обробки векторів незалежних даних, ефективно виконуються на конвеєрних функціональних пристроях. Найбільш відомі PVP-систем: NEC SX-4/SX-5, CRAY J90/T90.

В останні роки широкого поширення набули кластерні системи як дешева альтернатива суперкомп'ютерам [10,76]. Система необхідної продуктивності збирається з готових комп'ютерів, що серійно випускаються та об'єднаних за допомогою деякого комунікаційного устаткування, що також випускається серійно. Кластери є логічним продовженням ідей, закладених в архітектурі MPP-систем. При об'єднанні в кластер комп'ютерів різної потужності або різної архітектури, говорять про гетерогенні або неоднорідні кластери, в іншому випадку про гомогенні або однорідні кластери. Для зв'язку вузлів використовується одна із стандартних мережевих технологій таких, як Fast/Gigabit Ethernet на базі шинної архітектури або комутатора. Ряд фірм пропонують спеціалізовані рішення на

основі більш швидкісних мереж, таких як SCI фірми Scali Computer, а також Mirynet і InfiniBand [80-84].

В даний час в списку TOP500 найбільш високопродуктивних систем кластери становлять більшу частину, це Beowulf-кластери, NCSA NT Supercluster, Thunder Intel Itanium2 [84-85]. У списку Top50 для країн СНД п'яту позицію займає кластер з піковою продуктивністю 2.2 TFlop/s Інституту Кібернетики НАН України, а 7-е місце рейтингу належить комп'ютеру Київського політехнічного інституту з реальною продуктивністю 1.5 TFlop/s..

Існують декілька інших класифікацій ВС для паралельної обробки даних [5-9]. Більш детальна систематизація комп'ютерів класу MIMD запропонована Р.Хокні. Джонсон Є. проводить класифікацію MIMD архітектур на основі структури пам'яті та реалізації механізм взаємодії і синхронізації між процесорами. Особливої уваги заслуговує спосіб структурної нотації [11], що дозволяє з високою точністю описати багато характерних особливостей комп'ютерних систем.

У завершенні з усього сказаного важливо відзначити, що величезна кількість архітектурних рішень для паралельних обчислювальних систем призводить до того, що створення ефективного і масштабованого алгоритмічного та програмного забезпечення для них є трудомістким процесом. Тому на сьогодні не викликає сумнівів, що реалізувати потенційні можливості паралельних комп'ютерів можна тільки на основі проведення цілеспрямованої теоретичної роботи з побудови нових та адаптації існуючих алгоритмів розв’язання складних науково-технічних задач.

1.2. Характеристика типових топологій та комунікаційних примітивів для різних операцій передачі даних

Паралельні обчислювальні системи мають у своєму складі поле процесорних елементів, а також один або декілька модулів пам'яті. Ці

функціональні компоненти повинні бути пов'язані через відповідні комутаційні мережі. Всі високорівневі концепції комунікацій, такі, як загальна пам'ять: реальна чи віртуальна або обмін повідомленнями, реалізуються в паралельних системах деякими фізичними структурами.

У силу специфіки задач, що вирішуються на паралельних ОС [10], структура ліній зв'язку (топологія мережі передачі даних) повинна задовольняти різним вимогам. По-перше, забезпечувати зв'язок між двома будь-якими процесорами або модулями. Крім цього, підтримувати максимальну кількість одночасних зв'язків, щоб кошти комутації не обмежували паралельну обробку інформації.

До числа типових топологій за типом «точка-точка» зазвичай відносять наступні схеми: лінійка/кільце, сітка/2D-тор, гіперкуб, кліка або повний граф.

Лінійка (*linear array*) представляє собою лінійний масив процесорів. Кожен процесор з'єднаний з двома сусідніми (за винятком кінцевих процесорів, що мають по одному з'єднанню). Перевага цієї схеми – у її простоті. Недолік полягає в тому, що дані, перш ніж досягти свого кінцевого призначення, можуть пройти через кілька проміжних процесорів, $d = p - 1$.

Топологія кільце (*ring*) (рис. 1.6) виходить з лінійки шляхом з'єднання першого і останнього процесорів [10-11]. Зв'язки у топології кільце можуть бути односпрямованими або двоспрямованими. Кільцева структура зберігає переваги лінійки та скорочує максимальну відстань між процесорами – $p / 2$.

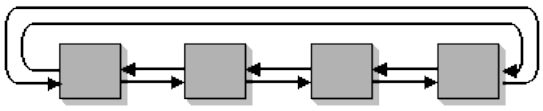


Рис. 1.6. Топологія кільце або 1D-тор

Сітка (2D-сітка, матриця – *mesh*) являє собою систему, в якій граф ліній зв’язку утворює прямокутну сітку, тобто процесорні елементи

розташовані у вигляді правильної двовимірної таблиці та кожен процесор (крім крайніх) з'єднаний із чотирма сусідами.

Перевагою схеми є простота, а недолік полягає в тому, що при обміні між віддаленими процесорами дані повинні пройти через ряд проміжних процесорів. Всі квадратні сітки мають максимальну відстань між процесорами порядку $O(\sqrt{p})$. Більшість реально побудованих граткових масивів використовують двовимірні схеми з'єднання. Крім двовимірних є схеми з тривимірним масивом процесорів [24], в яких кожен процесор з'єднаний з шістьма найближчими сусідами. У кубічної, 3D-сітці процесори розміщені в просторі куба, максимальна відстань між процесорними елементами пропорційно кореню кубічному із p: $\sqrt[3]{p}$. Якщо у сітці граничні процесори з'єднати лініями зв'язку, то вийде замкнутий варіант сітки або тор (рис. 1.7). Тороідальні 2D-схеми з'єднання мають діаметр, пропорційний $\sqrt{p}$. Квадратний тор з чотирма лініями зв'язків у кожного процесора має максимальну відстань між процесорними елементами: $\sqrt{p}$. Аналогічна схема комутації з 8 лініями зв'язків має діаметр, що дорівнює $\sqrt{p}/2$.

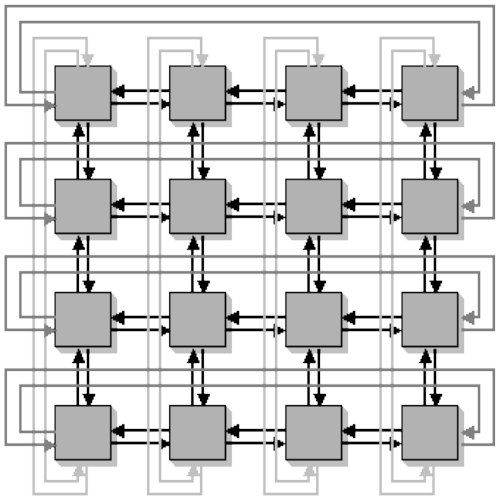


Рис. 1.7. Топологія замкнута сітка або 2D-тор

В існуючих обчислювальних системах однією з найбільш часто використовуваних та ефективних є топологія міжпроцесорних зв'язків – гіперкуб. Структура гіперкуб (*hypercube*) є окремим випадком сітки, коли за кожним розміром сітки є тільки два процесори, тобто гіперкуб містить $p = 2^N$ процесорів при розмірності N. При тривимірній схемі з'єднання процесори розміщені у вершинах тривимірного куба, а ребра куба грають роль локальних зв'язків між процесорами. Кожен процесор з'єднаний з трьома найближчими сусідами. Схема з'єднання чотиривимірного куба наведена на рис. 1.8.

При гіперкубовій архітектурі кількість зв'язків між процесорами невелика: у N-мірному гіперкубі кожний процесор має N сосідів. Невеликим порівняно з кількістю процесорів виявляється й діаметр системи, рівний $log_2 p$.

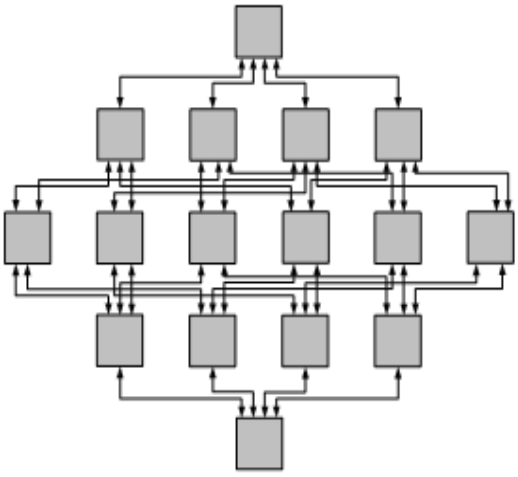


Рис. 1.8. Топологія гіперкуб $p = 2^4$

Топологія гіперкуб надає можливість логічно моделювати деякі важливі типи зв'язків: лінійка, кільце, сітка. Недоліком описаної архітектури є існування практичної межі збільшення розміру гіперкуба. Гіперкуб є універсальною і одною, що найбільш ефективних систем зв'язку, її концепції використані в обчислювальних системах Intel iPSC, Ncube Corp. та інших.

Комунікаційна структура кліка (*clique*) – передбачає існування ліній зв'язку між будь-якими двома процесорами в системі (рис. 1.9). Повна

зв'язаність системи з p процесорів вимагає, щоб з кожного процесора виходила $p-1$ лінія зв'язку, що робить вартість такої схеми досить великою при великому p. Перевагою такої схеми з'єднання є мінімальна величина діаметра, що дорівнює одиниці.

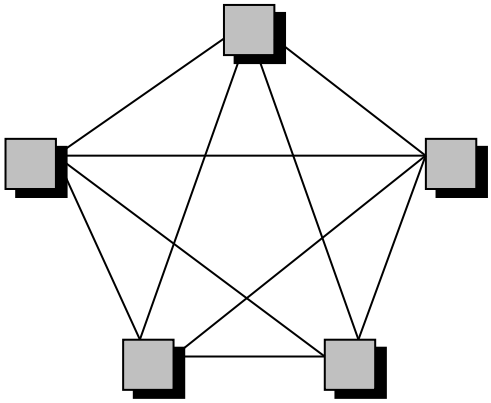


Рис. 1.9. Топологія кліка або повний граф

Для оцінки часу виконання операції передачі одного повідомлення обсягом V байтів між двома процесами, локалізованими на різних процесорах при розподіленій пам'яті, використовується наступна лінійна модель, запропонована Хокні [73]:

$$T_{p-p} = t_s + t_w \cdot V \cdot l, \; t_w = \frac{y}{B}, \tag{1.1}$$

де t_s – латентність, тривалість підготовки повідомлення для передачі;

l – довжина маршруту;

t_w – час передачі одного байта;

y – кількість байтів у слові;

B – пропускна здатність каналу передачі даних (байт/секунда).

Оскільки паралельні методи розв'язання динамічних задач із зосередженими параметрами використовують структурну інформаційну взаємодію за типом комунікацій, надається можливість розробити єдині комунікаційні примітиви для різних операцій передачі даних у різних топологіях.

Розглянемо топології, що найбільш поширені у використанні: лінійка/кільце, сітка/тор, гіперкуб та три комунікаційні операції:

1) передача даних між двома процесорами мережі – операція типу "крапка-крапка" або "point-point";

2) передача даних від одного процесора всім іншим процесорам мережі – операція "один-усім" або "one-to-all";

3) передача даних від усіх процесорів мережі усім процесорам мережі – множинна пересилка "усі-усім" або "all-to-all".

Трудомісткість одиночної операції пересилання даних між двома процесорами може бути отримана шляхом підстановки довжини максимального шляху (діаметра мережі) у вираз (1.1). Для обчислення часу виконання множинної пересилки необхідно вибрати алгоритм маршрутизації. До числа найбільш поширених оптимальних алгоритмів передачі даних відносять методи покоординатної маршрутизації [11].

Ідея цих методів полягає в тому, що пошук шляхів передачі даних здійснюється послідовно для кожної розмірності розглянутої топології.

Для кільцевої топології (рис. 1.6) кожен процесор може ініціювати розсилку повідомлення в будь-якому обраному напрямку за кільцем. Без істотних обмежень припускаємо, що будь-який процесор має можливість здійснювати одночасно прийом та передачу даних (двонапрямлені лінки). Таким чином, для топології кільце час виконання поодинокої пересилки даних складає:

$$T^R_{p\text{-}p} = t_s + V \cdot t_w \cdot \lfloor p/2 \rfloor . \qquad (1.2)$$

Передача даних від одного процесора всім іншим для топології кільце визначається співвідношенням:

$$T^R_{one\text{-}to\text{-}all} = t_s + V \cdot t_w \cdot \lceil p/2 \rceil , \qquad (1.3)$$

а час виконання множинної пересилки "усі-усім" становить:

$$T^R_{all\text{-}to\text{-}all} = (t_s + V \cdot t_w) \cdot (p - 1). \qquad (1.4)$$

Для топології сітка-тор (рис. 1.7) поодинока операція пересилки даних вимагає наступного часу для виконання з урахуванням моделі (1.1) та діаметра топології:

$$T^{M}_{p-p} = t_s + 2V \cdot t_w \cdot \left\lfloor \sqrt{p}/2 \right\rfloor. \quad (1.5)$$

Пересилка даних від одного процесора всім іншим в умовах топології тор може бути отримана з цього ж способу передачі для кільцевої топології, виконаного у два етапи:

$$T^{M}_{one-to-all} = t_s + 2V \cdot t_w . \left\lfloor \sqrt{p}/2 \right\rfloor. \quad (1.6)$$

Множинна розсилка повідомлень також може бути виконана за допомогою алгоритму, одержаного узагальненням способу передачі даних для кільцевої мережі на основі ідеї покоординатної маршрутизації:

$$T^{M}_{all-to-all} = 2t_s(\sqrt{p} - 1) + V \cdot t_w \cdot (p - 1). \quad (1.7)$$

Для топології гіперкуб (рис. 1.8) час виконання одиночної операції пересилки даних дорівнює:

$$T^{H}_{p-p} = t_s + V \cdot t_w \cdot \log_2 p. \quad (1.8)$$

Час виконання операції передачі даних від одного процесора всім іншим у топології гіперкуб становить:

$$T^{H}_{one-to-all} = (t_s + V \cdot t_w) \cdot \log_2 p. \quad (1.9)$$

Алгоритм виконання множинної розсилки повідомлень для гіперкуба може бути отриманий шляхом узагальнення способу передачі даних "усі-усім" для топології сітка на розмірність гіперкуба $lp = \log_2 p$.

Кожному процесору ставиться у відповідність двійковий еквівалент його номера. Процесори, що мають безпосереднє з'єднання у гіперкубі, будуть мати номери, що відрізняються один від одного тільки одним розрядом. На кожному етапі $i, i = \overline{1,lp}$ алгоритму функціонують усі процесори мережі, які обмінюються даними зі своїми сусідами за i-тою розмірністю та формують об'єднані повідомлення:

$$T^{H}_{all-to-all} = \sum_{i=1}^{lp} (t_s + 2^{i-1} \cdot V \cdot t_w) = t_s \cdot \log_2 p + t_w \cdot V \cdot (p - 1) \quad (1.10)$$

Особливістю кластерних систем є використання стандартних мережевих технологій. На даний момент найбільш популярними за рейтингом TOP-50 для СНД є Gigabit Ethernet, InfiniBand, Myrinet (рис. 1.10). Список наведено в порядку зменшення кількості кластерів, що використовують ту або іншу технологію. Аналогічна тенденція спостерігається й у світовому масштабі [84-85].

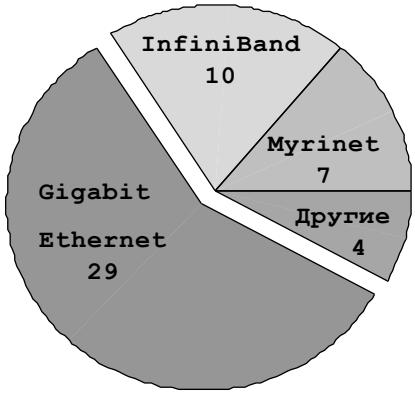


Рис. 1.10. Мережеві технології в рейтингу ОС СНД TOP-50

Ефективність використання тієї чи іншої мережевої технології визначається її числовими параметрами. У таблиці 1.1 на підставі матеріалів [79-83] приведені комунікаційні константи обміну для найбільш відомих комунікаційних мереж.

Таблиця 1.1

Комунікаційні константи для мережевих технологій

Мережева технологія	Латентність (мкс)	Швидкість передачі, Мб/с
Gigabit Ethernet	50	70
Myrinet 2000	10	200
SCI (Scalable Coherent Interface)	4	325
InfiniBand	7	1000 (пікова)

Огляд властивостей типових топологій, уведення комунікаційних примітивів операцій передачі даних, дають підстави проводити теоретичний порівняльний аналіз виконання різних паралельних алгоритмів.

1.3. Ієрархічна декомпозиційна технологія розробки паралельного алгоритму

Розвиненим і загальновизнаним математичним апаратом розробки паралельних алгоритмів є графові моделі: графи впливу, залежностей, тобто деякі інформаційні графи алгоритмів [3-5]. Однак для реальних задач застосування цієї моделі пов'язане з великими труднощами. Для прямого опису всіх вершин і дуг відповідного графа необхідні великі витрати комп'ютерної пам'яті, а для аналізу використовуються алгоритми, які практично всі мають високу обчислювальну складність, найчастіше експоненційну. Тому для розробки паралельних алгоритмів задач практичного рівня складності використовується багатоетапний процес, що починається аналізом задачі, вибором моделі обчислень, декомпозицією задачі на незалежні паралельні процеси та закінчується питаннями дослідження ефективності. Така технологія розпаралелювання отримала назву – ієрархічної декомпозиційної методики [10-11, 86-87].

Спочатку обчислювальна задача розбивається на підзадачі, аналізуються інформаційні взаємозв'язки між ними, визначається множина макрооперацій, оцінюється ефективність потенційного та реального паралелізму. У разі незадовільного результату на будь-якому з етапів схема розбиття змінюється, та весь процес повторюється з початку до тих пір, поки характеристики отриманого паралельного алгоритму не стануть задовільними або шляхом повного перебору не стане ясно, що домогтися якісного розпаралелювання неможливо. Графи впливу використовуються для побудови паралельного алгоритму як на верхньому рівні для аналізу взаємодії підзадач, так і на нижньому рівні для розпаралелювання окремих макрооперацій [3-9]. Уведення макрооперацій призводить до більш компактного подання графа алгоритму, значно спрощує проблему вибору ефективних способів розпаралелювання обчислень, дозволяє використовувати типові паралельні методи виконання макрооперацій як конструктивних елементів при розробці паралельних способів вирішення

більш складних обчислювальних задач. Процес уведення макрооперацій здійснюється поетапно з послідовно зростаючим рівнем деталізації операцій, що використовуються. Наступним етапом побудови паралельного алгоритму є визначення інформаційних потоків. Організація взаємодії процесів, що призводить до формування цілком певних, типових схем комунікації (кільце, тор, гіперкуб) – є структурною. Неструктурна взаємодія виникає, коли схема виконуваних операцій передач даних має граф "нетипового" вигляду. Останнім етапом розробки паралельного алгоритму є розподіл обчислювального навантаження за процесорами паралельної комп'ютерної системи та розв'язання задачі балансування навантаження.

Таким чином, розглядаються наступні етапи розробки паралельних алгоритмів із використанням поетапної декомпозиційної технології:

1) аналіз задачі та виявлення її потенційного паралелізму;

2) вибір моделі програмування та схеми розпаралелювання;

3) розподіл процесу обчислень на частини, які можуть бути виконані одночасно та визначення необхідних інформаційних взаємодій для паралельних процесів обробки даних;

5) відображення паралельної схеми розв'язання задачі на архітектуру комп'ютерної обчислювальної системи.

Вибір декомпозиційної методики обумовлений перевагами цього підходу, а саме: взаємозалежністю та ітеративністю етапів побудови паралельного алгоритму, можливістю забезпечити необхідний рівень масштабованості обчислень за рахунок варіювання детальності декомпозиції.

1.4. Високопродуктивні обчислення та динамічні характеристики паралельних алгоритмів

На сьогоднішній день величезна кількість робіт [4-9] присвячено аналізу існуючих і введення нових динамічних характеристик, що визначають якість паралельного алгоритму.

Наведемо загальноприйняті і найбільш використовувані в паралельних обчисленнях показники:

1) T_1 – час, необхідний для розв'язання задачі певного розміру на одному процесорі за допомогою найкращого послідовного алгоритму;

2) $T_{p,comp}$ – час для розв'язання задачі певного розміру з використанням паралельного алгоритму на комп'ютері з p процесорів без обліку обмінних операцій (час реалізації обчислень);

3) $T_{p,comm}$ – комунікаційна трудомісткість, складність алгоритму або час виконання операцій міжпроцесорного обміну;

4) T_p – загальний час реалізації паралельного алгоритму на паралельній архітектурі: $T_p = T_{p,comp} + T_{p,comm}$;

5) Z – доля часу операцій обміну в загальному часі виконання паралельного алгоритму: $Z = T_{p,comm} / T_p$;

6) Dop – ступінь паралелізму або максимальна кількість процесорів, що можуть бути використані при виконанні паралельного алгоритму;

7) S_{pot}, E_{pot} – коефіцієнти потенційного прискорення та ефективності (без урахування операцій обміну);

8) S, E – коефіцієнти реального прискорення та ефективності паралельного алгоритму (із урахуванням обмінних операцій).

Коефіцієнт прискорення (*speedup*) – одна з найбільш важливих динамічних характеристик паралельного алгоритму [10,71,75]. Коефіцієнт прискорення визначається, як відношення часу розв’язання задачі на однопроцесорній обчислювальної системі до часу виконання паралельного алгоритму. Розрізняють потенційний і реальний паралелізм, а, отже, і відповідні коефіцієнти прискорення. Коефіцієнт потенційного прискорення враховує тільки внутрішній паралелізм методу розв'язання задачі без урахування операцій обміну та інших накладних витрат:

$S_{pot} = T_1 / T_{p,comp}$,

$$S = T_1 / T_p . \qquad (1.11)$$

Наступною важливою характеристикою паралельного алгоритму є коефіцієнт ефективності [11,71], що визначає середню частку часу виконання алгоритму, протягом якого процесори реально використовуються для розв'язання задачі, тобто якість завантаження обладнання:

$$E = T_1 / p \cdot T_p = S / p . \qquad (1.12)$$

В ідеальному випадку досягається лінійне прискорення й одинична ефективність: $1 \le S \le p$, $1/p \le E \le 1$, за певних обставин може спостерігатися явище надлінійного прискорення: $S > p$ [11].

Розробка паралельних програм в якості своєї мети може мати не тільки зменшення часу виконання, але й забезпечення можливості вирішення задач більшої розмірності. Здатність паралельного алгоритму ефективно використовувати процесори при збільшенні складності розрахунків є важливою характеристикою паралельних обчислень і називається мірою масштабованості [10-14, 71]. Паралельний алгоритм є масштабовним, якщо при зростанні кількості процесорів він забезпечує збільшення прискорення при збереженні постійного рівня ефективності використання процесорів.

Існує декілька підходів для кількісної оцінки властивостей масштабованості алгоритму. Найбільш уживаною є метрика, запропонована в [11-14] та заснована на введенні функції рівної ефективності чи ізоефективності. Визначається нова динамічна характеристика – накладні витрати на паралелізм:

$$T_o(m,p) = p \cdot T_p - T_1 . \qquad (1.13)$$

Величина загальних накладних витрат включає сумарні витрати всіх процесорів паралельної системи: на реалізацію обмінів, послідовну частину розпаралеленого алгоритму, непродуктивні витрати на синхронізацію та час простою через незбалансованость завантаження процесорів. Використовуючи введені позначення, отримують нові вирази для коефіцієнтів прискорення та ефективності:

$$S = \frac{T_1}{T_p} = \frac{pT_1}{T_1 + T_o}, E = \frac{S}{p} = \frac{T_1}{T_1 + T_o} = \frac{1}{1 + T_o / T_1}. \quad (1.14)$$

Нехай $E = const$ задає необхідний рівень ефективності виконуваних обчислень, тоді:

$$T_o / T_1 = (1 - E) / E, \quad (1.15)$$

$$T_1 = E/(1 - E) \cdot T_0 = K(pT_p - T_1), \quad (1.16)$$

де $K = E/(1 - E)$ – є коефіцієнт, що залежить тільки від значення показника ефективності.

Породжену останнім співвідношенням залежність $m = f_E(p)$ між складністю розв'язуваної задачі та кількістю процесорів називають функцією ізоефективності. Вираз (1.16) є центральним співвідношенням ізоефективного аналізу. Функція f_E визначає, як треба збільшувати розмір задачі при збільшенні кількості процесорів для підтримки постійної ефективності.

Ізоефективна функція деяких паралельних систем є поліномом від розмірності процесорного поля: $O(p^x)$, де $x \geq 1$. При $x = 1$ отримуємо лінійно масштабовний паралельний алгоритм. Малий ступінь p у функції ізоефективності свідчить про високу масштабованість. Якщо ж мається експоненційна (або будь-яка інша швидко зростаюча функція) залежність, то мова йде про слабо масштабовні системи.

Близьким до лінійно масштабованих алгоритмів вважається квазілінійний алгоритм із функцією ізоефективності порядку $O(\log_x p)$ [11, 75]. Таким чином, побудова функції ізоефективності – це спроба поєднати в єдиному аналітичному виразі характеристики задачі та паралельної архітектури.

Під час аналізу ефективності паралельних алгоритмів необхідно враховувати тимчасові машинно-залежні параметри, що впливають на швидкість виконання паралельних обчислень. Таким параметром є час виконання операції з рухомою точкою. Кількість операцій з рухомою точкою

в секунду (*Floating Point Operations Per Second – FLOPS*) є характеристикою продуктивності процесора.

Час виконання операції з рухомою точкою будемо позначати:

$$t_{op} = 1 / FLOPS \ . \qquad (1.17)$$

При підрахунку динамічних характеристик алгоритмів уважається, що $t_{ad} = t_{mul} = t_{op}$ – будь-яка арифметична операція з рухомою точкою виконується за один і той же час незалежно від виду операції. Це припущення справедливе для більшості комп'ютерів RISC архітектури.

1.5. Аналіз паралельних методів розв'язання динамічних задач для систем звичайних диференційних рівнянь

Методи інтегрування динамічних задач із зосередженими параметрами, які описуються системами звичайних диференційних рівнянь,

$$\bar{y}' = \bar{f}(x, \bar{y}), \ \bar{y}(x_0) = \bar{y}_0, \qquad (1.18)$$

незважаючи на наявність величезних і всебічних знань у цій області, продовжують залишатися джерелом активних досліджень, особливо в такому новому науковому напрямку, як паралельні обчислення [93].

Розпаралелювання відомих та побудова нових паралельних методів вирішення задачі Коші має своєю метою прискорення обчислень при інтегруванні СЗДР.

Це особливо важливо для науково-технічних задач великих розмірностей, складних правих частин та для швидкого моделювання динамічних процесів у реальному часі.

Паралелізм, що використовується при розв'язанні СЗДР, можна класифікувати двома способами:

а) системний паралелізм (паралелізм у просторі);

б) паралелізм методу (паралелізм у часі).

Однокроковий s-стадійний метод типу Рунге-Кутти має вигляд:

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h_n \cdot \sum_{i=1}^{s} b_i \cdot \bar{k}_i; \\ \bar{k}_i = \bar{f}(x_n + c_i \cdot h_n; \bar{y}_n + h_n \cdot \sum_{j=1}^{s} a_{ij} \cdot \bar{k}_j); i = 1,...,s; \end{cases} \quad (1.19)$$

де s – кількість стадій методу,

а $A = (a_{ij})_{i,j=1}^{s}$, $b = (b_1,...,b_s)^T$,

$c = (c_1,...,c_s)^T$ – дійсні коефіцієнти, які визначають унікальний варіант методу.

Паралельні методи Рунге-Кутти, як і послідовні, можна розбити на два великі класи: явні (ЯМРК) і неявні (НМРК) [93-94]. Класи відрізняються один від одного способом обчислення стадійних коефіцієнтів $\bar{k}_i = (k_{i1},k_{i2},...,k_{im}), i = \overline{1,s}$, характеристиками стійкості, а, отже, й областю застосування.

Найбільш відомими є роботи [93-100], засновані на реструктуризації явних методів Рунге-Кутти. За малого потенційного паралелізму методу підходи до розпаралелювання ЯМРК експлуатують в основному системний паралелізм. У цьому сенсі інтерес представляють так звані P-оптимальні методи [93], що реалізують паралелізм у часі. Якщо деякі елементи матриці Батчера дорівнюють нулю, то з'являється можливість обчислювати відповідні стадійні коефіцієнти паралельно. Для цього будується орієнтований граф, в якому кожна дуга, проведена з вершини i у вершину j, встановлює ненульовий коефіцієнт матриці коефіцієнтів ЯМРК.

Широкого розповсюдження розпаралелювання на основі P-оптимальності не отримало, оскільки практично всі класичні явні методи Рунге-Кутти не мають таких властивостей. З іншого боку, навіть наявність таких властивостей забезпечує незначний ступінь паралелізму. Явні однокрокові методи мають обмежену область застосування через нездатність вирішувати жорсткі задачі. У зв'язку з цим значний інтерес представляють неявні схеми, які, незважаючи на велику обчислювальну складність, не мають альтернативи серед однокрокових методів при інтегруванні жорстких

рівнянь. У [101-103] запропоновані стійкі чисельні методи, що використовують неявні схеми.

Для обчислення рішення кожен з цих методів вимагає розв'язку систем нелінійних алгебраїчних рівнянь. Описаний підхід до вирішення завдання має високу ефективність, тому що дозволяє будувати чисельні методи високого порядку збіжності. Істотним недоліком цих методів є значні витрати машинного часу, що пов'язані, по-перше, зі збільшенням розмірності задачі та зі зростанням кількості ітерацій, необхідних для забезпечення збіжності. У роботах [103-106] запропоновані паралельні діагонально та одноразово-діагональні неявні методи, що істотно скорочують процес вирішення СНАР на основі модифікованого методу Ньютона. Однак ці методи мають гірші характеристики стійкості, і визначають розв'язок меншого порядку точності.

До цього часу мова йшла про реструктуризовані методи. У матеріалах [107-113] запропоновані блокові методи на основі неявних схем, орієнтовані на виконання в паралельних системах. Роботи [107-109] описують способи побудови однокрокових k-точкових методів, в яких для блоку з k точок нові k значень розв'язку можуть бути обчислені одночасно. Авторами [110-112] отримано розрахункові формули для m-крокових k-точкових блокових методів, визначено порядок їхньої точності. У [113] розглядаються загальні лінійні багатокрокові блокові методи, доведено збіжність методів до точного розв'язку, визначено умови стійкості за Далквістом.

Одним із головних питань, що виникають під час чисельного вирішення систем звичайних диференційних рівнянь, є проблема оцінки похибки наближеного розв'язку. Апостеріорна оцінка локальної похибки, яка отримується на кожному кроці обчислень, дозволяє автоматично вибирати крок інтегрування, що забезпечує задану точність чисельного розв'язку. Однокрокові методи безпосередньо не надають простої можливості отримати інформацію про локальну точність наближеного розв'язку за результатами проміжних обчислень.

Відомими та загальноприйнятими методами оцінки локальної апостеріорної похибки розв’язання СЗДР є:

- дублювання кроку за правилом Рунге,

- технологія локальної екстраполяції Річардсона,

- вкладені методи або методи вкладених форм [114-115].

Найбільш простий спосіб визначення локальної похибки це дублювання кроку за правилом Рунге. Для точного розв'язку при переході з точки x_n до точки $x_n + 2h$, $\bar{y}(x_n + 2h)$ на основі однієї й тієї ж формули порядку r отримують дві апроксимації: $\bar{y}_{n+1}^{(1)}$ (один крок $2h$) и $\bar{y}_{n+1}^{(2)}$ (два кроки h). Різниця між цими значеннями використовується в якості оцінки локальної похибки інтегрування, як розв'язок приймається апроксимація, отримана з половинним кроком, як найбільш точна. Процес уточнення розв'язку з половинним кроком $\bar{y}_{n+1}^{(2)}$ підвищує точність вирішення на одиницю та носить назву локальної екстраполяції [93]. Даний спосіб оцінки похибки досить простий, однак має великі накладні витрати: обсяг обчислень на один вузол сітки зростає майже втричі.

Метод локальної екстраполяції Річардсона є узагальненням технології подвоєння кроку за правилом Рунге [115]. Ідея методу полягає в багаторазовому подрібненні кроку інтегрування, і також у багаторазовому застосуванні процесу обчислення, названого вище локальною екстраполяцією.

Розв'язання задачі Коші розглядається при переході з точки x_n у точку $x_{n+1} = x_n + H$, де H – базова довжина кроку (рис. 1.11). Вибирається ряд натуральних чисел $P_i = \{n_1, n_2, ..., n_k, ..\}$ такий, що: $n_1 < n_2 < ... < n_k < ...$ та, відповідно, послідовність кроків: $h_1 > h_2 > ... > h_{k-1} > h_k > ...$, де $h_i = H / n_i$. Задається опорний чисельний метод порядку r_0 та, після виконання n_i

кроків інтегрування з довжиною h_i, обчислюється наближений розв'язок вхідної задачі в точці x_{n+1}:

$$T_{i,1} = \bar{y}_{h_i}(x_n + H) = \bar{y}^{(i)}_{n+1},\ i = \overline{1,k}. \tag{1.20}$$

Після виконання обчислень для ряду послідовних значень i, за рекурентним співвідношенням (1.21) визначають $T_{i,j+1}$ – екстрапольовані значення для довільних i,j:

$$T_{i,j+1} := T_{i,j} + \frac{T_{i,j} - T_{i-1,j}}{(n_i / n_{i-j})^b - 1}. \tag{1.21}$$

Обчислення за (1.21) є багаторазово повтореним процесом локальної поліноміальної екстраполяції за схемою Ейткена-Невілла [93]. Величина b дорівнює одиниці в загальному випадку, для симетричних опорних методів b дорівнює двом.

T_{ij} – наближений розв'язок задачі Коші, отриманий чисельним методом порядку $r_0 + (j-1)\cdot b$ з кроком h_i.

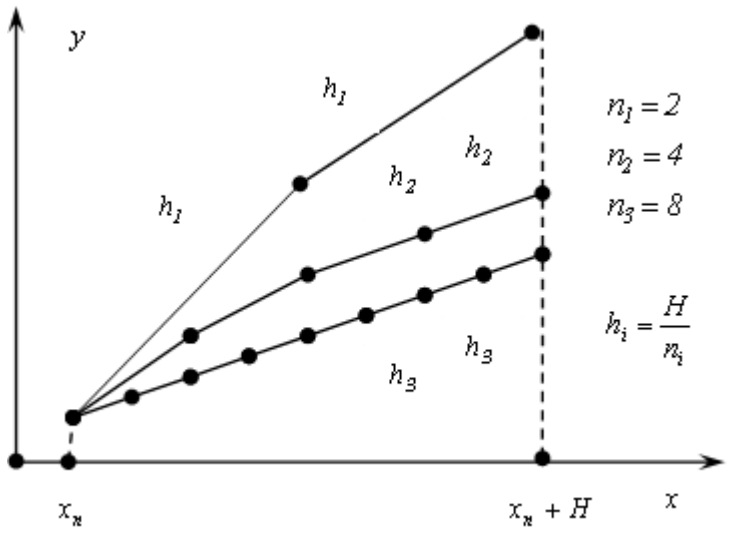


Рис. 1.11. Технологія локальної екстраполяції Річардсона

Переваги цього методу полягають у тому, що він дає таблицю результатів обчислень (табл. 1.2), які утворюють послідовність вкладених

методів, дозволяють оцінити локальну похибку та вибрати стратегію для методів змінного порядку.

Таблиця 1.2

Екстраполяційна таблиця

r_0	$r_0 + b$	$r_0 + 2b$	…	$r_0 + (k-2)b$	$r_0 + (k-1)b$
T_{11}			…		
T_{21}	T_{22}		…		
…	…	…	…	T_{k-1k-1}	
T_{k1}	T_{k2}	T_{k3}	…	T_{k-1k}	T_{kk}

Альтернативним способом визначення локальної апостеріорної похибки при розв'язанні задачі Коші є вкладені методи Рунге-Кутти (ВМРК). Цей спосіб заснований на використанні двох наближених значень розв'язку в одній точці, але на відміну від правила Рунге наближення обчислюються не за одною, а за двома формулами різних порядків точності r і $\hat{r}$ із одним й тим же кроком [115].

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h_n \cdot \sum\limits_{l=1}^{s} b_l \cdot \bar{k}_l, \\ \hat{\bar{y}}_{n+1} = \bar{y}_n + h_n \cdot \sum\limits_{l=1}^{s'} b'_l \cdot \bar{k}_l, \\ d = \left\| (\hat{\bar{y}}_{n+1} - \bar{y}_{n+1}) \right\|. \end{cases} \qquad (1.22)$$

Для визначення локальної похибки менш точного результату та управління величиною кроку інтегрування використовується величина d. Вкладений метод Рунге-Кутта $r(\hat{r})$ – це схема, в якій метод $\hat{r}$-го порядку ("оцінювача похибки") виходить як побічний продукт методу r-го порядку (рис. 1.12). Порядок апроксимацій: $\bar{y}_{n+1}$ та $\hat{\bar{y}}_{n+1}$ зазвичай відрізняються на 1, тобто $r = \hat{r} + 1$ або $r = \hat{r} - 1$. Обидва наближення використовують практично одні й ті ж значення стадійних коефіцієнтів, але комбінують їх по-різному. Вкладені методи Рунге-Кутти – це найменш трудомісткий з відомих способів

визначення локальної апостеріорної похибки розв'язку у послідовній реалізації.

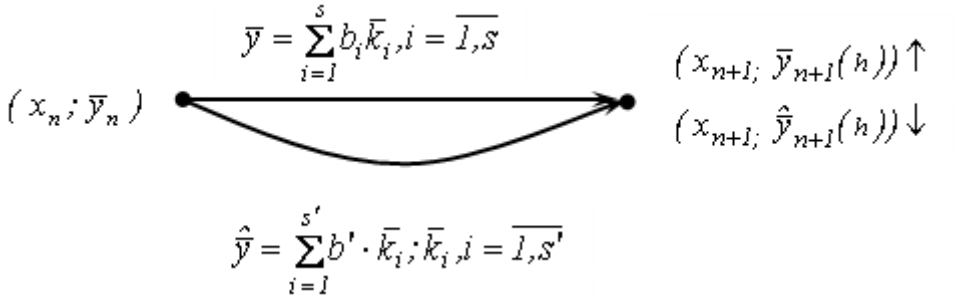


Рис. 1.12. Схема обчислень для вкладених методів

Дійсно, якщо для розв’язання задачі Коші використовується класичний метод Рунге-Кутти четвертого порядку з контролем точності за правилом Рунге, то потрібно одинадцять обчислень похідної – правої частини рівняння, а для вкладеного методу Фельбергу 4(5) того ж порядку точності всього 6.

Величина локальної похибки використовується для того, щоб забезпечити довжину кроку інтегрування, яка достатня для необхідної точності результатів, але таку, що гарантує відсутність марної обчислювальної роботи. Тому отримання надійних і в той же час простих та ефективних способів оцінки крокової похибки – одна з основних проблем, що виникають при конструюванні чисельного методу розв'язання задачі Коші в паралельній реалізації. Узагальнюючи відомі теоретичні дослідження та обчислювальні експерименти [96-113] необхідно відзначити, що, незважаючи на те, що методи вирішення СЗДР використовуються в обчислювальній практиці досить давно, розробка нових ефективних паралельних обчислювальних схем для розв'язання конкретних динамічних задач залишається актуальною проблемою.

ГЛАВА II

РОЗВ'ЯЗАННЯ НЕЛІНІЙНОЇ ЗАДАЧІ КОШІ ПАРАЛЕЛЬНИМИ ЯВНИМИ МЕТОДАМИ З ОЦІНКОЮ ЛОКАЛЬНОЇ ПОХИБКИ

Сучасні дослідження та обчислювальні експерименти в галузі високопродуктивних обчислень виявили, що практично всі нові паралельні методи, навіть ті з них, що є ефективними в теоретичному відношенні, на практиці не конкурентноспроможні. Тому на даний момент єдиним надійним джерелом створення паралельних методів є придатна реструктуризація перевірених часом послідовних алгоритмів [3].

Чисельно розв’язується задача Коші для системи звичайних диференційних рівнянь (СЗДР) першого порядку з відомими початковими умовами:

$$\begin{cases} \dfrac{dy_1(x)}{dx} = f_1(x; y_1, y_2, ..., y_m), \quad y_1(x_0) = y_{10}, \\ \dfrac{dy_2(x)}{dx} = f_2(x; y_1, y_2, ..., y_m), \quad y_2(x_0) = y_{20}, \\ \qquad ... \\ \dfrac{dy_m(x)}{dx} = f_m(x; y_1, y_2, ..., y_m), \quad y_m(x_0) = y_{m0}. \end{cases} \tag{2.1}$$

Явний s-стадійний однокроковий метод Рунге-Кутти (ЯМРК) може бути визначений таким чином:

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s} b_i \bar{k}_i, \ \bar{k}_i = F(x_n + c_i h; \bar{g}_i), \\ \bar{g}_i = \bar{y}_n + h \cdot \sum\limits_{j=1}^{i-1} a_{ij} \bar{k}_j, \ i = 1, ..., s. \end{cases} \tag{2.2}$$

Розглянемо різні обчислювальні схеми паралельних алгоритмів на базі ЯМРК для СЗДР загального типу з урахуванням вбудованих методів оцінки локальної апостеріорної похибки.

2.1. Ефективність розпаралелювання явних однокрокових чисельних схем із дублюванням кроку за правилом Рунге

При розпаралелюванні алгоритму розв'язання нелінійної задачі Коші на основі явних однокрокових чисельних схем із використанням правила Рунге, скористуймося ієрархічною декомпозиційною методикою. Спочатку обчислювальна задача розбивається на підзадачі, аналізуються інформаційні взаємозв'язки між ними, бієктивно до множини підзадач визначається множина макрооперацій, оцінюється ефективність потенційного велико- чи середньо блочного паралелізму. Потім методика застосовується для розпаралелювання кожної з макрооперацій, тобто використовується дрібнозернистий паралелізм. Для розробки паралельного алгоритму та перевірки коректності його побудови на кожному з рівнів розпаралелювання використовується математичний апарат графів впливу [3-5].

Один крок інтегрування СЗДР на основі ЯМРК із вбудованим способом оцінки локальної похибки за правилом Рунге має вигляд:

$$\begin{cases} \bar{y}_{n+1}^{(1)} = \bar{y}_n + 2h \cdot \sum\limits_{i=1}^{s} b_i \cdot \bar{k}_i^{(1)}(2h), \quad \bar{k}_i^{(1)} = F\left(x_n + c_i \cdot 2h, \bar{y}_n + 2h \sum\limits_{j=1}^{i-1} a_{ij} \bar{k}_j^{(1)} \right), \\ \bar{y}_{n+\frac{1}{2}} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s} b_i \cdot \bar{k}_i(h), \quad \bar{k}_i = F\left(x_n + c_i h, \bar{y}_n + h \sum\limits_{j=1}^{i-1} a_{ij} \bar{k}_j \right), \; i = \overline{1,s}, \\ \bar{y}_{n+1}^{(2)} = \bar{y}_{n+\frac{1}{2}} + h \cdot \sum\limits_{i=1}^{s} b_i \cdot \bar{k}_i^{(2)}(h), \quad \bar{k}_i^{(2)} = F\left(x_{n+\frac{1}{2}} + c_i h, \bar{y}_{n+\frac{1}{2}} + h \sum\limits_{j=1}^{i-1} a_{ij} \bar{k}_j^{(2)} \right). \end{cases} \tag{2.3}$$

Послідовний алгоритм, що описується обчислювальною схемою (2.3), можна розбити на три однотипні підзадачі, що виконують обчислення наступних апроксимацій розв'язку: $\bar{y}_{n+1}^{(1)}(2h)$ у точці x_{n+1} із кроком $2h$, $\bar{y}_{n+1/2}(h)$ у точці $x_{n+1/2} = x_n + h$ із кроком h, $\bar{y}_{n+1}^{(2)}(2h)$ у точці x_{n+1} через проміжну точку з кроком h. Усі ці три задачі є застосуванням явної однокрокової схеми, що дозволяє ввести в якості основної макрооперації на цьому рівні одноразове обчислення апроксимації розв’язку в деякій точці сітки з заданим кроком інтегрування. Розглянемо три різні макрооперaційні обчислювальні схеми досліджуваного алгоритму (рис. 2.1). Перший варіант

макрооперaційної обчислювальної схеми паралельного алгоритму, що конструюється, представлено на рисунку 2.1а. Тут усі три підзадачі виконуються послідовно, дані між процесорами розподілено рівномірно.

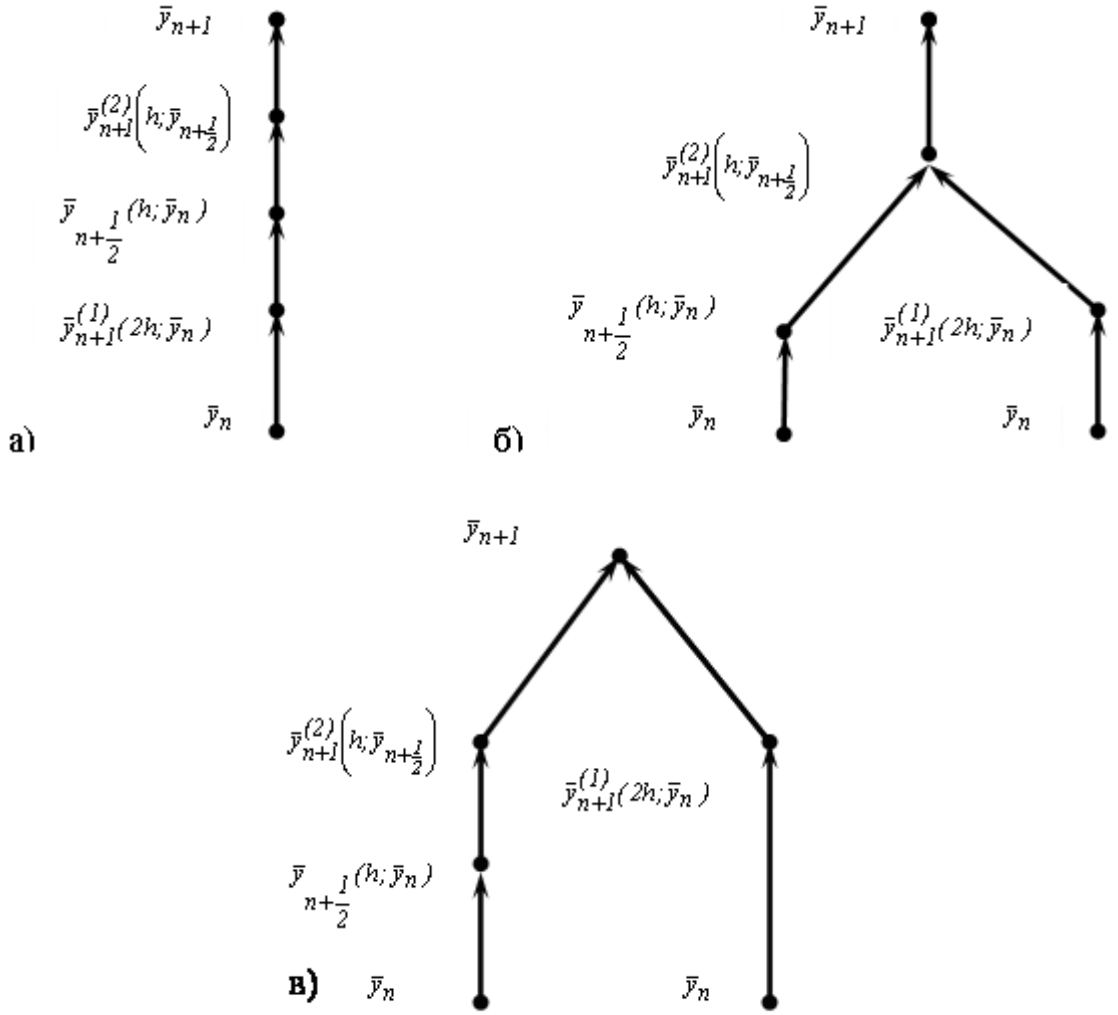


Рис. 2.1. Графи впливу макрооперaційних обчислювальних схем паралельного алгоритма ЯМРК у поєднанні з правилом Рунге,
а) схема № 1, б) схема № 2 (рівномірне розбиття),
в) схема № 3 (пропорційне розбиття)

Другий варіант представлено на рисунку 2.1б. Перша та друга підзадачі виконуються паралельно, причому вся множина процесорів розділяється на дві рівні частини: $p_1 = \lceil p/2 \rceil$. Вхідні дані рівномірно розподілено між двома рівними групами процесорів для вирішення перших

двох підзадач, а потім перерозподілено на всі процесори для вирішення третьої підзадачі. Для виконання наступного кроку інтегрування потрібно провести перерозподіл на групи процесорів. Третій варіант макрооперaційної обчислювальної схеми (рис. 2.1в) передбачає паралельне виконання з одного боку першої підзадачі на меншій кількості процесорів, а з іншого боку паралельно другої та третьої - на більшій кількості процесорів. При цьому вся множина процесорів розподіляється на дві пропорційні частини: $p_1 = 2p_2;\ p_1 + p_2 = p$, відповідно до функціонального розбиття обчислювальної схеми.

На цьому рівні деталізації вже очевидно, що друга схема гірша, ніж перша та третя. Послідовність виконання трьох макрооперацій в першій схемі компенсується рівномірним завантаженням і низькою інтенсивністю взаємодії процесів. З іншого боку, друга і третя обчислювальні схеми алгоритму мають однаковий ступінь паралелізму, у два рази більший $Dop = p_{max} = 2m$, ніж перша схема. Але збалансування завантаження в третій схемі вище, ніж у другій. Перейдемо до дослідження паралелізму на рівні окремих операцій, тобто до дрібнозернистого паралелізму.

Для розробки паралельних алгоритмів розглянутих макрооперацій використовується математичний апарат графів впливу. Задача розпаралелювання в такій постановці зводиться до пошуку максимальної незалежної множини вершин в орграфах. Причому, кожній вершині орграфу алгоритму ставиться у відповідність деяка виконувана операція. Вершини з'єднуються дугами тоді та тільки тоді, коли результат виконання однієї операції впливає на результат суміжної [3-5].

Паралельне розв'язання СЗДР на базі ЯМРК концентрується на паралельному виконанні одного кроку інтегрування. Для явних схем обчислення величин $\bar{k}_i$ є суто послідовним процесом, тому скористаємося системним паралелізмом. Послідовний алгоритм одного кроку інтегрування за явною обчислювальною схемою може бути представлений, як розв'язання

s підзадач обчислення стадійних коефіцієнтів і однієї підзадачі обчислення наближеного розв’язку. Тому в якості першої макрооперації цього рівня деталізації вводиться обчислення i-того стадійного коефіцієнта, а в якості другої – обчислення чисельної апроксимації розв'язку у деякій точці. На рисунку 2.2 приведено граф впливу першої макрооперації [38-39].

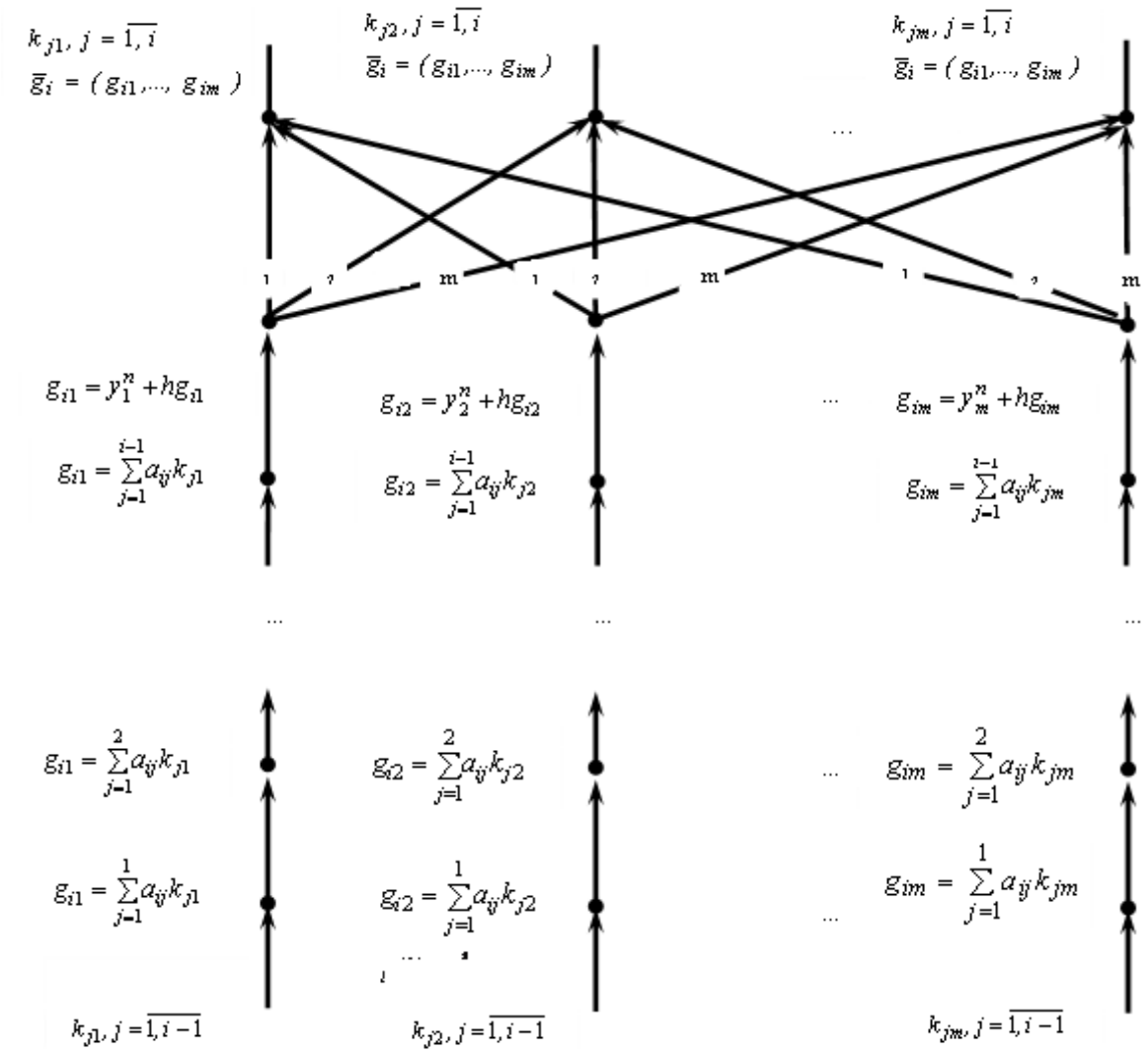


Рис. 2.2. Граф впливу обчислення i - того стадійного коефіцієнту ЯМРК для СЗДР $\bar{k}_i=(k_{i1},k_{i2},\dots,k_{im}),m=p$

Друга макрооперація визначає обчислення наближеного розв’язку на деякому кроці інтегрування, та за наявного розбиття даних представлена графом впливу, що зображений на рис. 2.3. Обидві макрооперації є базовими

для всіх явних схем і будуть використані в подальшому під час розгляду альтернативних способів визначення локальної похибки.

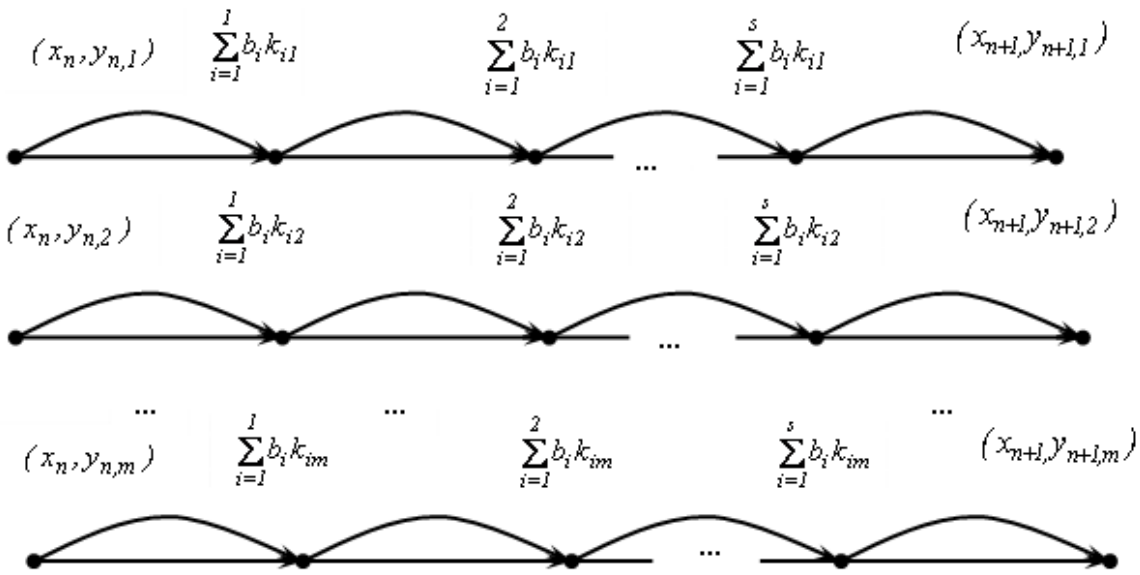


Рис. 2.3. Граф впливу другої макрооперації: обчислення апроксимації розв'язання у точці (x_{n+1}, y_{n+1})

Щоб задача побудови паралельного алгоритму була математично коректною на цьому етапі, необхідно зробити припущення щодо властивостей використаної паралельної обчислювальної системи. Застосуємо концепцію необмеженого паралелізму [4]. В якості ідеалізованої моделі паралельної ОС використовується мультикомп'ютер із розподіленою пам'яттю та комутаційною схемою довільної топології.

Рис. 2.4 представляє відображення першої макрооперaційної схеми паралельного алгоритму ЯМРК із правилом Рунге на мультикомп'ютер із p процесорів. Для системи, що складається із m звичайних диференційних рівнянь, на одному часовому кроці m_1 компонент векторів $\bar{g}_i$ та, відповідно, стадійних коефіцієнтів $\bar{k}_i$ можуть бути обчислені паралельно. Причому: 1) $m_1 = m$ при $m = p$; 2) $m_1 = \lceil m/p \rceil$ при $m > p$. Визначення апроксимацій розв’язку та перехід до наступного кроку не вимагають додаткових передач даних.

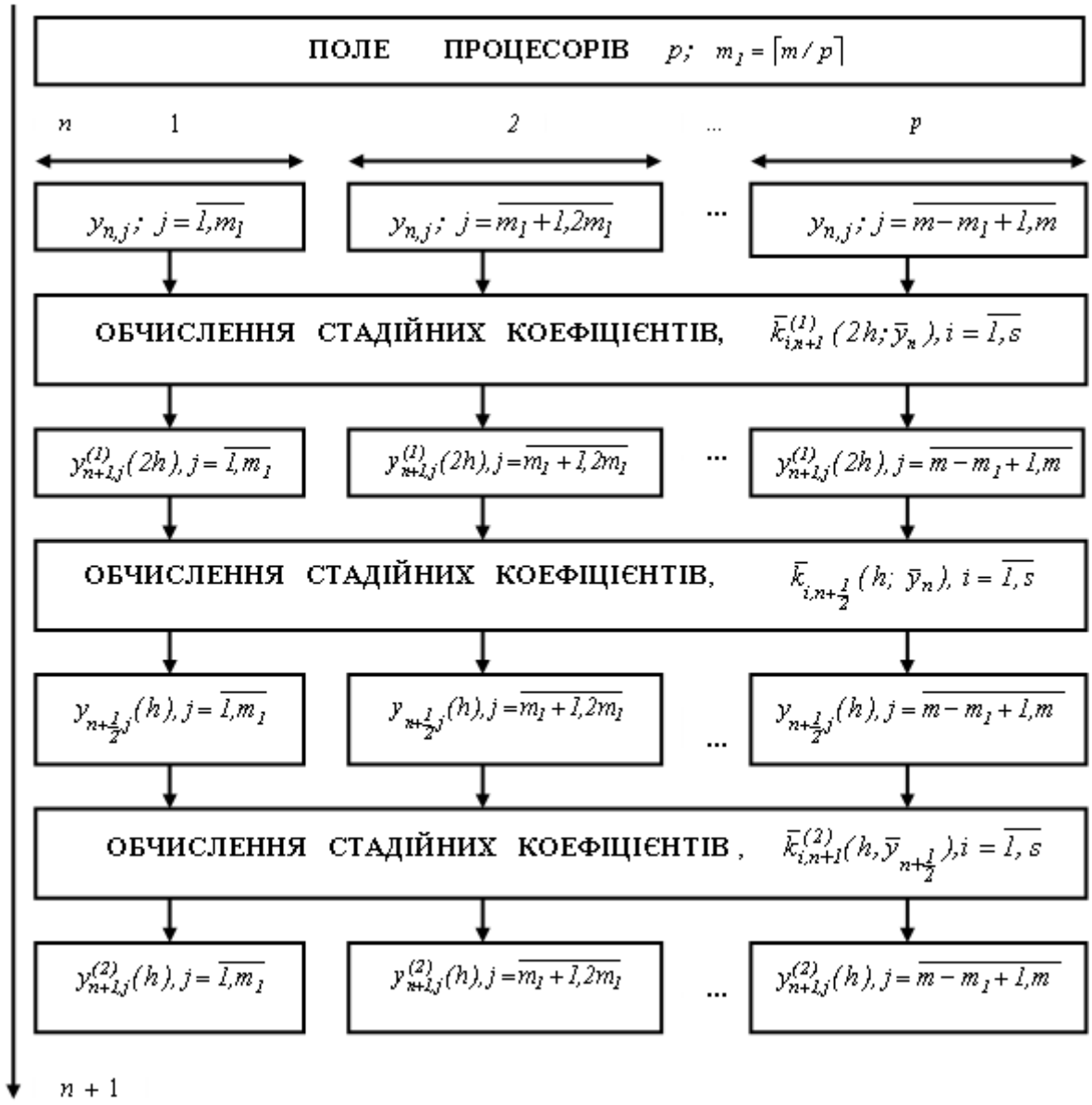


Рис. 2.4. Паралельний алгоритм ЯМРК + правило Рунге для чисельного розв'язання нелінійної задачі Коши (варіант 1)

Рисунок 2.5 описує алгоритм розв'язання тієї ж задачі Коші для третьої макрооперациоої схеми. Тут розбиття процесорів зроблено пропорційно до обсягу обчислень. Базова макрооперація описана графом впливу на рисунку 2.2. Перша група процесорів обчислює дві апроксимації розв’язку: $\bar{y}_{n+1/2}$ та $\bar{y}^{(2)}_{n+1}(h)$ і має розмір процесорного поля у два рази більший, ніж друга група. Друга група процесорів обчислює одну апроксимацію розв’язку з подвоєним кроком: $\bar{y}^{(1)}_{n+1}(2h)$.

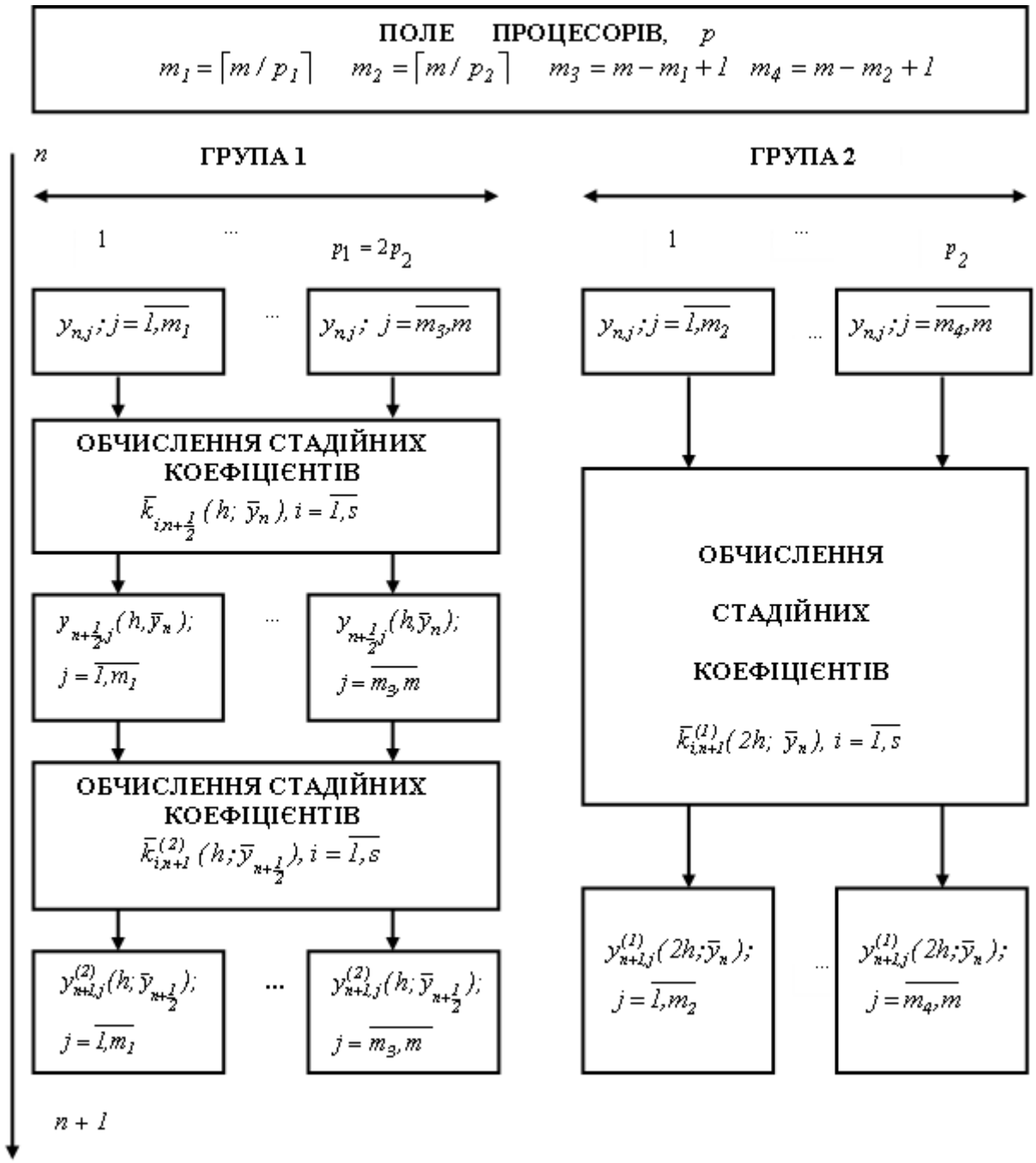


Рис. 2.5. Паралельний алгоритм ЯМРК+ правило Рунге
(варіант 3 – пропорційний розподіл даних за процесорами)

Кожен процесор першої групи обчислює $m_1 = \lceil m / p_1 \rceil$ компонент стадійних векторів і векторів рішень, і кожен процесор другої групи обчислює $m_2 = \lceil m / p_2 \rceil$ компонент таких же векторів. Таким чином, усі обчислення проводяться паралельно двома групами з внутрішньогруповими пересилками за типом "усі-усім" і лише для визначення кроку інтегрування

буде потрібна множинна пересилка даних між групами. На даному етапі досліджень очевидно, що друга та третя схеми менш збалансовані, більш складні в реалізації та, мабуть, менш ефективні. Однак для точної відповіді на це питання необхідно дослідити динамічні характеристики паралелізму для всіх трьох обчислювальних схем.

Час виконання послідовного алгоритму явного s-стадійного методу в застосуванні до нелінійної задачі Коші з правилом Рунге складається з часу обчислення трьох апроксимацій розв'язку та визначається як:

$$T_1^1 = m(3s-1)T_F + 3(ms^2 + 2ms + 2s - 2)t_{op}, T_F = \sum_{i=1}^{m} T_{f_i}. \quad (2.4)$$

Часи реалізації розрахунків для паралельних алгоритмів ЯМРК з правилом Рунге за різними обчислювальними схемами визначаються, як:

1) обчислювальна схема №1 $m \geq p$, $m_1 = \lceil m/p \rceil$:

$$T_{p,comp}^{11} = (3s-1)m_1 T_F + 3(m_1 s^2 + 2m_1 s + 2s - 2)t_{op}; \quad (2.5)$$

2) обчислювальна схема №2: $p_1 = \lceil p/2 \rceil$, $m_1 = \lceil m/p \rceil$, $m_2 = \lceil m/p_1 \rceil$:

$$T_{p,comp}^{12} = s(m_1 + m_2)T_F + [(m_1 + m_2)s^2 + 2(m_1 + m_2)s + 4s - 4)]t_{op}; \quad (2.6)$$

3) обчислювальна схема №3: $p_1 \vdots p_2 = 2 \vdots 1$; $m_1 = \lceil m/p_1 \rceil$; $m_2 = \lceil m/p_2 \rceil$:

$$T_{p,comp}^{13} = m_2 s T_F + (m_2 s^2 + 2m_2 s + 4s - 4)t_{op}. \quad (2.7)$$

Величина T_F істотно залежить від розглянутої архітектури паралельної ОС, для синхронної паралельності: $T_F = \sum_{i=1}^{m} T_{f_i}$; для асинхронної – $T_F = max_{i=1}^{m} T_{f_i}$. Для оцінки потенційного паралелізму розглянемо два варіанти: $T_F \gg t_{op}$ – час звернення до правої частини СЗДР багато більше флопа; $T_F \sim t_{op}$ – час звернення до правої частини пропорційний флопу:

1) $T_F \gg t_{op} : T_1^1 \approx 3sm T_F; \quad T_p^{11} \approx 3m_1 s T_F = 3\lceil m/p \rceil \cdot s \cdot T_F \quad \Rightarrow$

$\Rightarrow \quad S_{pot,1} = T_1^1 / T_p^{11} \approx p \Rightarrow \quad E_{pot,1} \approx 1;$

$$2)\, T_F \sim t_{op} : \; T_1^1 \approx (3s^2 m + 9sm) t_{op}; \; T_p^{11} \approx (3s^2 + 9s) m_1 t_{op} =$$

$$= (3s^2 + 9s) \lceil m / p \rceil \cdot t_{op} \;\Rightarrow\quad S_{pot,1} \approx p \;\Rightarrow\quad E_{pot,1} \approx 1.$$

Аналогічні результати для другої та третьої обчислювальних схем дозволяють зробити висновок, що характеристики потенційного паралелізму є задовільними: практично лінійне прискорення, одинична ефективність і реально не залежать від вибору макрооперацийної схеми алгоритму.

До цих пір нерідко вважається, що найкращий обчислювальний алгоритм той, який має мінімум арифметичних операцій, особливо трудомістких. Однак для паралельних обчислень важливим, а іноді і визначальним, є облік складності міжпроцесорних зв'язків. Оскільки для методів Рунге-Кутти інформаційна взаємодія за типом комунікацій є структурною, надається можливість використовувати єдині комунікаційні примітиви для різних операцій передачі даних у різних топологіях.

Визначимо час міжпроцесорного обміну для трьох обчислювальних схем на основі введених у підрозділі 1.2 комунікаційних примітивів:

$$T_{p,comm}^{11} = 3sT_{all-to-all}(m_1; p), \tag{2.8}$$

$$T_{p,comm}^{12} = sT_{all-to-all}(m_2; p_1) + (s+1)T_{all-to-all}(m_1; p) + T_{all-to-all}(m_2; p), \tag{2.9}$$

$$T_{p,comm}^{13} = max[2sT_{all-to-all}(m_1, p_1); sT_{all-to-all}(m_2, p_2)] + T_{all-to-all}(m_2, p). \tag{2.10}$$

З урахуванням моделі (1.1), обчислимо час обміну для різних топологій та двох обчислювальних схем. Оскільки потенційні характеристики обчислювальних схем наближено можна вважати однаковими, зробимо порівняння алгоритмів на основі комунікаційних складових.

Одним з основних параметрів, які впливають на час міжпроцесорного обміну, є топологія з'єднання процесорів. За даними таблиці 2.1, для кожного з розроблених паралельних алгоритмів найбільш ефективною є топологія гіперкуб. Гірший варіант - кільце, тому що час обмінів при такому з'єднанні процесорів є визначальним і, як наслідок, алгоритм має незначні оцінки

якості паралелізму. Варіант топології 2D-тор залежно від значень комунікаційних параметрів може наближатися за оцінками до гіперкуба, але не має перед ним переваг для запропонованих обчислювальних схем.

Таблиця 2.1

Час виконання міжпроцесорного обміну для трьох обчислювальних схем ЯМРК + правило Рунге та різних топологій

Обчислювальна схема	Коефіцієнт при t_s	Коефіцієнт при t_w
Топологія гіперкуб		
№1	$3s\log_2 p$	$3s\lceil m/p\rceil\cdot(p-1)$
№2	$(2s+2)\log_2 p$	$\lceil m/p\rceil\cdot(2sp-3s+3p-3)$
№3	$(2s+1)\log_2 p-2s\log_2 3+2s$	$\lceil m/p\rceil\cdot(2sp-3s+3p-3)$
Топологія 2D-тор (замкнута сітка)		
№1	$6s(\sqrt{p}-1)$	$3s\lceil m/p\rceil\cdot(p-1)$
№2	$(2+\sqrt{2})s\sqrt{p}-4s+4\sqrt{p-4}$	$\lceil m/p\rceil\cdot(2sp-3s+3p-3)$
№3	$\left(4\sqrt{2/3}s+2\right)\sqrt{p-4s-2}$	$\lceil m/p\rceil\cdot(2sp-3s+3p-3)$
Топологія 1D-тор (кільце)		
№1	$3s(p-1)$	$3s\lceil m/p\rceil\cdot(p-1)$
№2	$1{,}5sp+2p-2s-2$	$\lceil m/p\rceil\cdot(2sp-3s+3p-3)$
№3	$(4/3)sp-2s+p-1$	$\lceil m/p\rceil\cdot(2sp-3s+3p-3)$

Більш інформативною величиною є відношення комунікаційних витрат до загальних накладних витрат паралельного алгоритму. Оскільки встановлено, що топологія гіперкуб найбільш оптимальна для розглянутих задач (рис. 2.7), дамо оцінку впливу машинно-залежних комунікаційних констант на динамічні параметри паралельних алгоритмів. Тимчасові характеристики обмінних операцій істотно залежать від класу паралельних ОС, що використовується. Згідно даних за співвідношенням між латентністю та часом передачі одного слова t_s/t_w виділимо три різні групи мультикомп'ютерів: кластерні, MIMD та SIMD-системи. Кластерні паралельні системи з низькошвидкісними мережами мають високі значення

латентності та малу швидкість передачі даних, тому ефективні в алгоритмах із малою часткою обмінних операцій.

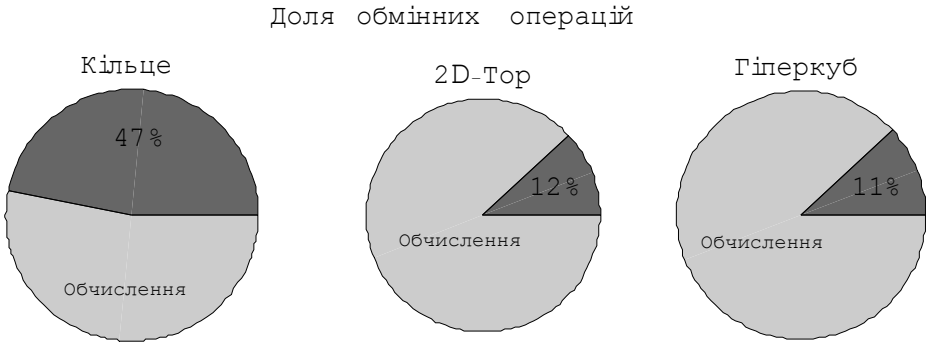
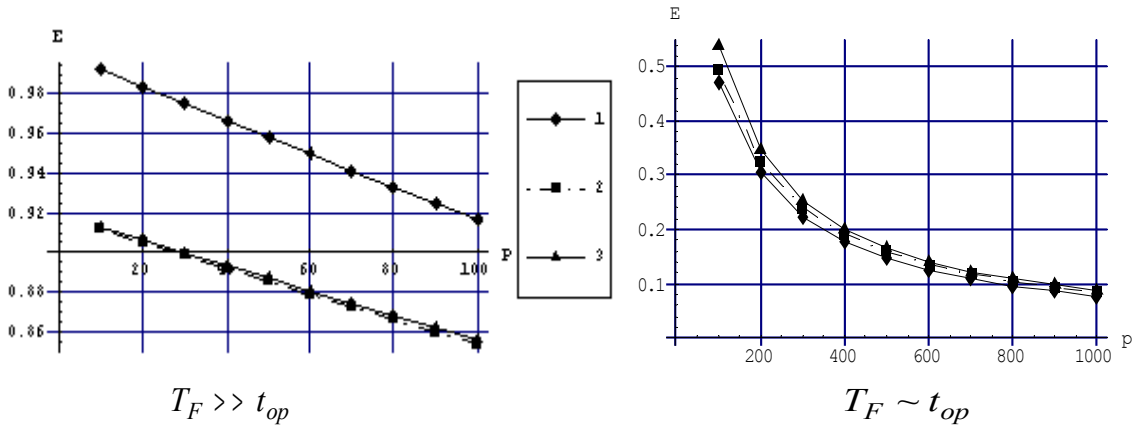


Рис. 2.7. Доля операцій обміну в загальному часі виконання паралельного алгоритму №1 у різних топологіях

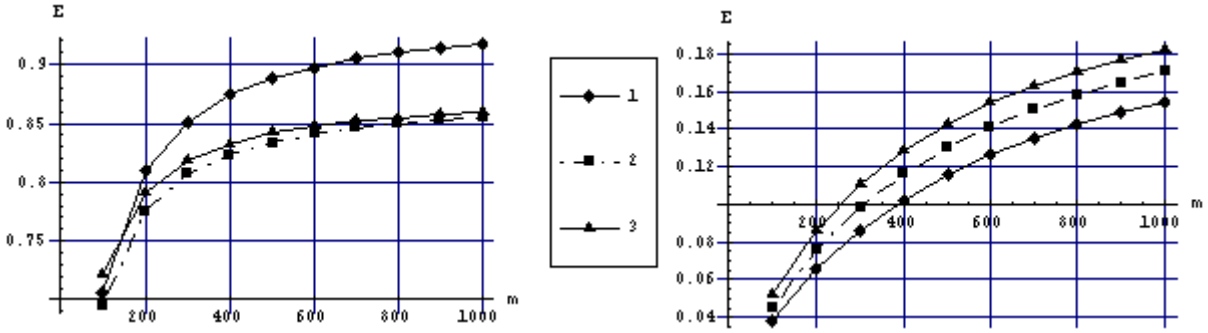
MIMD-архітектури мають меншу латентність і високу швидкість передачі даних, ефективність їх застосування визначається співвідношенням між обчислювальною та обмінною частинами алгоритму [11]. Якість паралельного алгоритму оцінюється коефіцієнтами реального прискорення й ефективності, що є функціями багатьох параметрів: $\phi(p,m,t_s,t_w,t_{op},T_F)$. Для MIMD-систем та тривіальних правих частин ефективність розпаралелювання для всіх паралельних алгоритмів досить низька (рис. 2.8).



$T_F >> t_{op}$ $T_F \sim t_{op}$

Рис. 2.8. Коефіцієнти ефективності обчислювальних схем ЯМРК + правило Рунге для MIMD-систем

Для СЗДР із домінуючими правими частинами вочевидь перевагою володіє перша схема, а друга та третя схеми дають дуже близькі результати. При збільшенні порядку методу, коефіцієнти прискорення й ефективності збільшуються, але незначно: $\uparrow r \Rightarrow \uparrow s \Rightarrow \uparrow S \Rightarrow \uparrow E$, зі зростанням розмірності задачі істотно зростають: $\uparrow m \Rightarrow \uparrow S \Rightarrow \uparrow E$. Описані закономірності аналогічні для MIMD та кластерних систем (рис. 2.9), підтверджуються серією експериментів, проведених на системі тестів для нежорстких СЗДР [37]. Цей факт свідчить про те, що поліпшення тимчасових характеристик сучасних комунікаційних мереж у кластері робить такі багатопроцесорні архітектури ефективними для розроблюваних паралельних програм.



$T_F >> t_{op}$ $T_F \sim t_{op}$

Рис. 2.9. Коефіцієнти ефективності обчислювальних схем ЯМРК + правило Рунге для кластерних систем

Паралельні обчислювальні системи типу SIMD відрізняються від інших типів ОС тим, що на кожному процесорному елементі виконується один і той же потік команд. Оскільки в другій та третій схемах обчислення в різних групах процесорів не однорідні, дослідження їх відображення на SIMD-системи неправомірне. Проаналізуємо реальний паралелізм тільки першої обчислювальної схеми. Для неї характерні невисокі показники прискорення й ефективності, причому цей факт має місце за будь-яких значеннь параметрів: розміру задачі, кількості процесорів та складності правих частин (рис. 2.10).

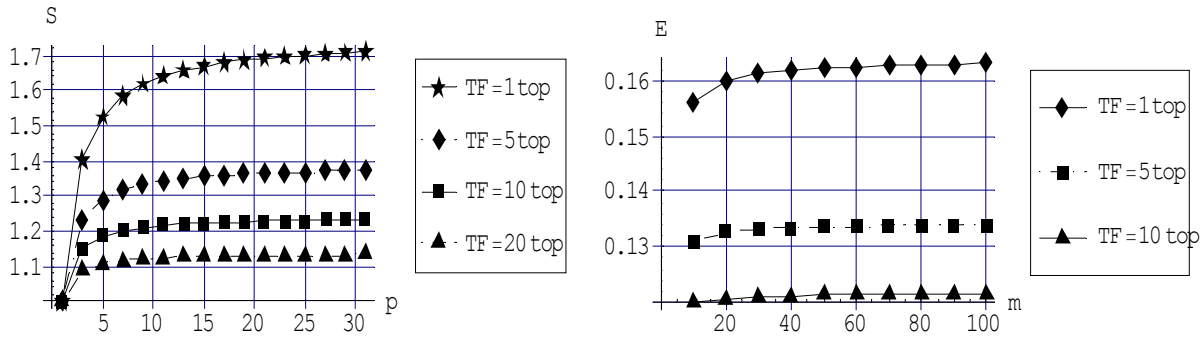


Рис. 2.10. Коефіцієнти прискорення і ефективності обчислювальної схеми № 1 ЯМРК із правилом Рунге для SIMD-систем

Невеликі витрати на реалізацію обмінів при синхронній паралельності компенсуються послідовними обчисленнями правої частини СЗДР, тому розпаралелювання нелінійної задачі Коші з правилом Рунге на ОС SIMD-архітектури є мало ефективним. Таким чином, ефективне розпаралелювання нелінійної задачі Коші на основі явних схем із правилом Рунге можливо при наявності складної правої частини СЗДР (порядку сотень і більше флопів) для асинхронних паралельних архітектур із високошвидкісними мережами обміну даними.

2.2. Підвищення ефективності паралельного розв'язання нелінійних задач Коші на основі явних вкладених методів

Найбільш ефективним способом визначення локальної похибки при вирішенні задачі Коші для однопроцесорних машин є методи вкладених форм, які додатково вимагають обчислення одного або декількох стадійних коефіцієнтів для визначення більш точної апроксимації розв'язку на кроці.

Вкладені методи визначають дві формули Рунге-Кутти суміжних порядків точності та для інтегрування СЗДР мають вигляд:

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s} b_i \cdot \bar{k}_i, \\ \hat{\bar{y}}_{n+1} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s'} \hat{b}_i \cdot \bar{k}_i. \end{cases} \qquad (2.11)$$

Якщо основу вкладених форм складають явні різницеві схеми, то стадійні коефіцієнти повинні визначатися за формулами:

$$\begin{cases} \bar{k}_i = F(x_n + c_i h; \bar{g}_i), \\ \bar{g}_i = \bar{y}_n + h\sum\limits_{j=1}^{i-1} a_{ij} \cdot \bar{k}_j, i = \overline{1,s'}, \end{cases} \quad (2.12)$$

з тією різницею, що їх кількість визначається як $max(s,s')$. Для визначеності припустимо, що $s' > s$.

При розробці паралельного алгоритму вкладеного методу Рунге-Кутти скористаємося ієрархічною декомпозиційною методикою та результатами розробки паралельного алгоритму ЯМРК із правилом Рунге.

Процес обчислень за формулами (2.11)-(2.12) можна представити як вирішення двох підзадач:

1) обчислення наближеного розв'язку $\bar{y}_{n+1}(h)$ порядку r у точці x_{n+1} із кроком h;

2) обчислення наближеного розв'язку $\hat{\bar{y}}_{n+1}(h)$ порядку $\hat{r}$ у точці x_{n+1} із тим же кроком.

Граф впливу макрооперaційної схеми алгоритму наведено на рисунку 2.11. Максимальний ступінь паралелізму алгоритму співпадає з розміром системи ЗДР та дорівнює $Dop = m$. У свою чергу макрооперації рисунку 2.11 можна представити, як вирішення s' підзадач обчислення стадійних коефіцієнтів та двох підзадач обчислення наближеного розв’язку. Очевидно, що множина макрооперацій для ВМРК збігається з множиною макрооперацій для ЯМРК із правилом Рунге. Графи впливу обох макрооперацій було наведено на рисунках 2.2-2.3.

Тоді, час на реалізацію послідовного явного вкладеного методу Рунге-Кутти $r(\hat{r})$ включає час обчислення стадійних коефіцієнтів, обох апроксимацій розв'язку та становить:

$$T_1^2 = (s+1)mT_F + [s^2m + 6sm + 2s + 4m]t_{op}. \quad (2.13)$$

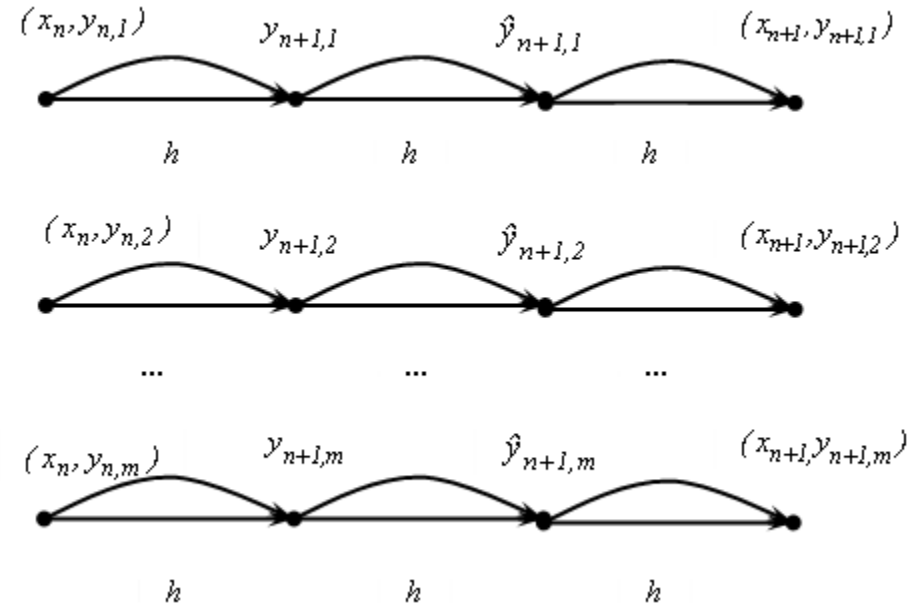


Рис. 2.11. Граф впливу макрооперaційної схеми
вкладених явних методів для розв'язання нелінійної задачі Коші

На рисунку 2.12 представлено паралельний алгоритм розв'язання задачі Коші за методом вкладених форм для мультикомп'ютера з p процесорами. Нехай СЗДР має розмірність, рівну m; тоді m_1 компонента стадійних коефіцієнтів та векторів розв'язку розподіляється на кожний з процесорів: $m_1 = m$ при $m = p$ та $m_1 = \lceil m/p \rceil$ при $m > p$.

Явні чисельні схеми визначають стадійні коефіцієнти послідовно, тому для кожного з них, m_1 компонента додаткового вектора $\bar{g}_i$ може бути обчислена паралельно. Після реалізації кожного такого обчислення компоненти вектора $\bar{g}_i, i = \overline{1,s'}$ повинні бути доступні всім процесорам для обчислення чергового стадійного коефіцієнта. Потім кожен процесор обчислює рівну частину компонент вектора стадійних коефіцієнтів, шляхом відповідного обчислення компонент правої частини вхідної СЗДР, і рівну кількість компонент векторів рішень. Зауважимо, що додаткових пересилань тут не знадобиться, тому що кожен процесор на наступному тимчасовому кроці буде працювати з однією та тією ж частиною компонент, як стадійних векторів, так і векторів рішень [47].

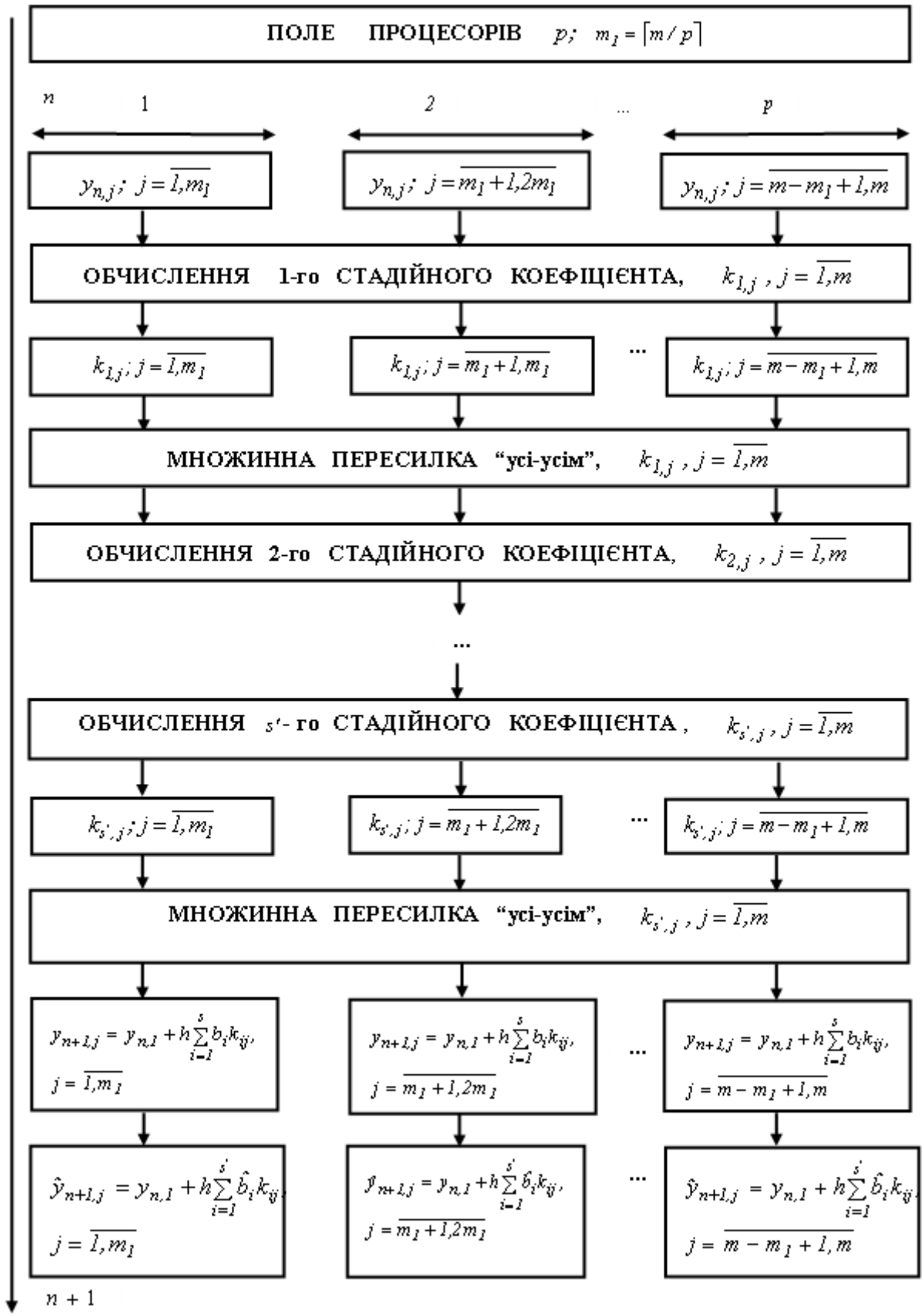


Рис. 2.12. Паралельний алгоритм розв'язання нелінійної задачі Коші для СЗДР явними ВМРК $r(\hat{r})$

Час виконання паралельного алгоритму ВМРК $r(\hat{r})$ складає:

$$T^2_{p,comp} = m_1(s+1)T_F + (s^2 m_1 + 6sm_1 + 2s + 4m_1)t_{op}. \quad (2.14)$$

$$T^2_{p,comm} = s' T_{all-to-all}(m_1, p), \text{ где } m_1 = \lceil m/p \rceil. \quad (2.15)$$

Оцінки потенційного паралелізму ВМРК близькі до оптимальних, тобто практично лінійне прискорення й одинична ефективність. Визначимо час передачі даних для алгоритму ВМРК для різних топологій, обсяг переданих даних дорівнює $m_1 = \lceil m/p \rceil$:

– кільце: $T^2_{p,comm} = T^{2,R}_{all-to-all} = s' \cdot [(t_s + \lceil m/p \rceil \cdot t_w) \cdot (p-1)];$ (2.16)

– тор: $T^2_{p,comm} = T^{2,M}_{all-to-all} = s' \cdot \left[2t_s(\sqrt{p}-1) + \lceil m/p \rceil \cdot t_w \cdot (p-1)\right];$ (2.17)

– гіперкуб: $T^2_{p,comm} = T^{2,H}_{all-to-all} = s' \cdot [t_s \cdot log_2 p + t_w \cdot \lceil m/p \rceil \cdot (p-1)].$ (2.18)

Оскільки час обчислень паралельного алгоритму для кластерних і MIMD систем ідентичний, то великого значення набуває величина комунікаційної складової. Проаналізуймо зміну часу обміну від топології з'єднання процесорів, комунікаційних констант і складності правої частини СЗДР (рис. 2.13).

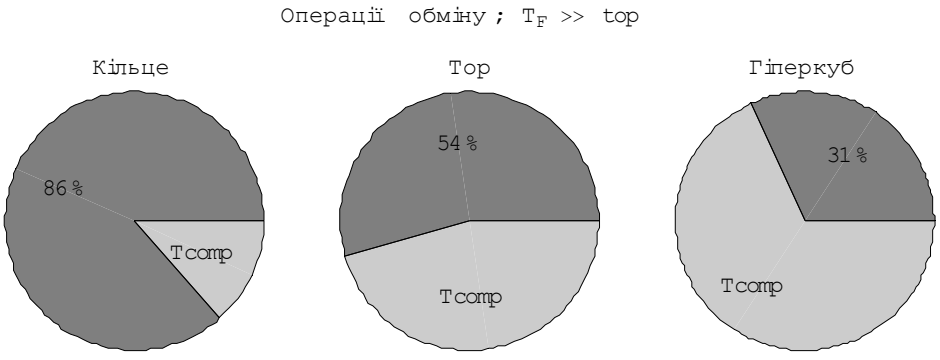
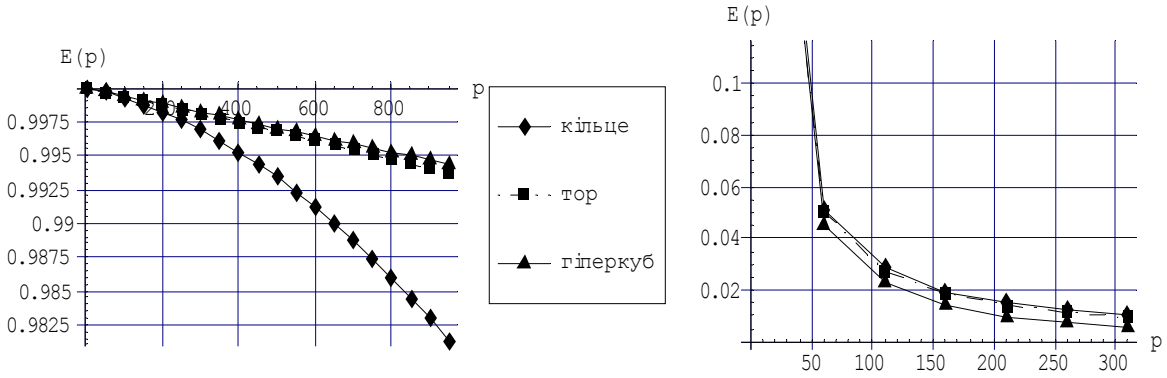


Рис. 2.13. Частка часу обмінів у загальному часі виконання паралельних ВЯМРК для MIMD-систем у різних топологіях

Як і для попередніх алгоритмів, поєднання за типом кільце є найгіршим варіантом, топологія гіперкуб є найкращою. Частка обмінних операцій зменшується з ростом складності правої частини СЗДР. Для

кластерних та MIMD-систем (рис. 2.14) поведінка динамічних характеристик паралельного алгоритму ВЯМРК ідентична. Визначимо, за яких умов розпаралелювання стає ефективним.

Коефіцієнти прискорення та ефективності збільшуються зі зростанням розміру задачі: $\uparrow m \Rightarrow \uparrow S \Rightarrow \uparrow E$ та складності правої частини СЗДР $\uparrow T_F \Rightarrow \uparrow S \Rightarrow \uparrow E$. Величина коефіцієнта прискорення збільшується, а коефіцієнта ефективності зменшується зі збільшенням кількості процесорів $\uparrow p \Rightarrow \uparrow S \Rightarrow \downarrow E$. Зауважимо, що це справедливо, поки кількість процесорів не перевищить максимальний ступінь паралелізму. Таким чином, розпаралелювання ВЯМРК ефективне при складних правих частинах СЗДР, і витрати на паралелізм не окупаються при тривіальних [48-50].



$T_f >> t_{op}$ $T_f \sim t_{op}$

Рис. 2.14. Коефіцієнти ефективності вкладених явних методів розв'язання задачі Коші для різних топологій в MIMD-архітектурі

Для SIMD-систем звернення до правої частини СЗДР виконуються послідовно, що істотно впливає на характеристики паралелізму. Навіть, якщо вектор правих частин СЗДР є тривіальним за обчисленнями, послідовна реалізація звернення до нього при великих розмірностях задачі стає визначальною. Коефіцієнти реального прискорення й ефективності для синхронних систем незначні, зменшуються зі зростанням кількості процесорів і практично збігаються для різних топологій.

2.3. Паралельна реалізація технології локальної екстраполяції Річардсона для явних однокрокових схем

2.3.1. Підвищення ефективності технології локальної екстраполяції для явних однокрокових опорних методів

Побудуємо ефективний з точки зору мінімізації обчислювальних витрат метод інтегрування СЗДР на основі технології Річардсона. Екстраполяційна технологія включає опорний чисельний метод розв'язання задачі Коші, послідовність сіток інтегрування, рекурентне правило обчислення значень наближеного розв`язку. Ефективність її застосування безпосередньо залежить від правильного вибору та поєднання всіх трьох складових. Для отримання сіток інтегрування застосовуються числові послідовності, утворені гармонійним рядом, рядом парних чисел, ступенями двійки [93]:

$$\begin{aligned} &1)\ P_1 = \{1,2,3,4,5,6,...\} - \quad n_j = j; \\ &2)\ P_2 = \{1,2,4,6,8,10,12,...\} - \quad n_j = 2(j-1), j > 1 \vee n_1 = 1; \\ &3)\ P_3 = \{1,2,4,8,16,32,64,...\} \quad - \quad n_j = 2^{j-1}. \end{aligned} \tag{2.19}$$

Обчислювальна складність генерації сіток інтегрування визначається кількістю звернень до правої частини СЗДР і дорівнює:

$$\begin{aligned} &N(k) = n_1 + n_2 + ... + n_k, \\ &P_1: \quad N(k) = (k^2 + k)/2, \\ &P_2: \quad N(k) = k^2 - k + 1, \\ &P_3: \quad N(k) = 2^k - 1, \end{aligned} \tag{2.20}$$

де k – кількість рядків екстраполяційної таблиці.

При одному й тому ж опорному методі найбільш витратною є послідовність P_3, яка фактично повторює процес подвоєння кроку, послідовності P_1 та P_2 мають приблизно однакову обчислювальну складність [46, 62].

Вибір опорного методу інтегрування базується на дослідженні впливу порядку методу та його властивостей на обсяг необхідних обчислень. Нехай опорний метод має порядок – r_0, довжина екстраполяційної таблиці дорівнює k, а кількість стадій явного методу s, визначимо необхідний розмір екстраполяційної таблиці для отримання апроксимації розв`язку порядку r. За умови симетричності опорного методу, він має асимптотичне розкладання за ступенями h^2, отже, кожна екстраполяція виключає два ступеня h [93-95]. Формування екстраполяційної таблиці для симетричних методів на основі парних рядів вимагає $k = \lceil (r - r_0)/2 + 1 \rceil$ рядків для досягнення точності порядку r, а для довільних методів – $k = r - r_0 + 1$. Таким чином, h^2-екстраполяція на основі симетричних опорних методів і другої парної послідовності для генерації сітки інтегрування є найбільш ефективною з точки зору обчислювальних витрат. Наприклад, якщо в якості опорного методу вибирається симетричний явний метод другого порядку точності, $r_0 = 2$, тоді для P_2 кількість обчислень f складає: $N(r/2) = (r^2 - 2 \cdot r + 4)/4$.

Час послідовних обчислень за схемою локальної екстраполяції складається з часу визначення апроксимації розв'язку з різними кроками інтегрування та часу складання екстраполяціонной таблиці: $T_1^3 = T_{T11} + T_{T21} + \ldots + T_{Tk1} + T_1^{ext-tab}$. Кожна з апроксимацій $T_{11},\ldots,T_{k1}$ виходить за рахунок n_i разів застосованої схеми явного однокрокового опорного методу з різними кроками інтегрування:

$$T_{T11} = T_{\bar{y}_{n+1}^{(1)}\left(\frac{h}{n_1}\right)} = n_1 \cdot T_1^{r_0}, \ldots, T_{Tk1} = T_{\bar{y}_{n+1}^{(k)}\left(\frac{h}{n_k}\right)} = n_k \cdot T_1^{r_0}, \qquad (2.21)$$

де $T_1^{r_0}$ – час рішення задачі Коші для СЗДР розмірності m опорним методом порядку r_0.

Очевидно, що $T_1^3 = T_1^{r0} \cdot \sum_{i=1}^{k} n_i + T_1^{ext-tab}$. Час обчислення екстраполяціної таблиці за кожною розмірностю СЗДР дорівнює добутку кількості екстрапольованих значень $K_{elem-ext}$ на час обчислення одного екстрапольованого значення за формулою (1.21), T_1^{ext}:

$$K_{elem-ext} = (k^2 - k)/2 \; ; \; T_1^{ext} = m(2t_{mul} + 3t_{ad}) = 5mt_{op}; \tag{2.22}$$

$$T_1^{ext-tab} = K_{elem-ext} \cdot T_1^{ext} = 5m(k^2 - k)/2 \cdot t_{op}. \tag{2.23}$$

Для симетричних опорних методів та P_2 маємо:

$$N_{P_2}(k) = k^2 - k + 1; k = \lceil (r - r_0)/2 + 1 \rceil. \tag{2.24}$$

Таким чином, час виконання послідовного алгоритму в застосуванні до нелінійної задачі Коші з оцінкою апостеріорної локальної похибки по технології локальної екстраполяції становить:

$$\begin{cases} T_1^3 = \left(\dfrac{r^2 - 2r + 4}{4} + \dfrac{a^2 - 2ar + 2a}{4} \right) T_1^{r_o} + T_1^{ext-tab}, \\ T_1^{ext-tab} = 5m \left(\dfrac{r^2 - 2r}{8} + \dfrac{a^2 - 2ar + 2a}{8} \right) \cdot t_{op}, \; a = r_0 - 2. \end{cases} \tag{2.25}$$

Оцінки часу обчислення одного кроку інтегрування для явних опорних симетричних методів різного порядку точності ($r_0 = 2 \div 10$) та другої парної послідовності зведені в таблицю 2.2. Обчислювальна складність одного кроку інтегрування за технологією ЛЕР при одних й тих же сітках інтегрування, порядку отриманого методу r істотно залежить від порядку опорного методу та його властивостей (рис. 2.15). Так, довжина екстраполяційної таблиці лінійно зменшується з ростом порядку опорного методу: $r_0 \uparrow \Rightarrow \; k \downarrow$ [63-65]. Накладні витрати на екстраполяцію зростають із зростанням порядку опорного методу: $r_0 \uparrow \Rightarrow \; T_1^{r0} \uparrow$, причому при $T_F \gg t_{op}$ у $1{,}55 \div 2$ рази для методів суміжних порядків, при $T_F \approx t_{op}$ у десятки / сотні разів.

Для зменшення накладних витрат при застосуванні локальної екстраполяції слід вибирати опорний метод малого порядку точності.

Таблиця 2.2

Обчислювальна складність одного кроку інтегрування за технологією локальної екстраполяції з симетричним опорним методом

Порядок /кількість стадій симетричного опорного ЯМРК, r_0 / s	Накладні витрати на 1 крок інтегрування за ЯМРК порядку r_0	Довжина екстраполяційної таблиці, k
2/2	$2mT_F + (8m + 2)t_{op}$	$r / 2$
4/4	$4mT_F + (24m + 6)t_{op}$	$r / 2 - 1$
6/7	$7mT_F + (63m + 12)t_{op}$	$r / 2 - 2$
8/11	$11mT_F + (143m + 20)t_{op}$	$r / 2 - 3$
10/17	$17mT_F + (323m + 32)t_{op}$	$r / 2 - 4$

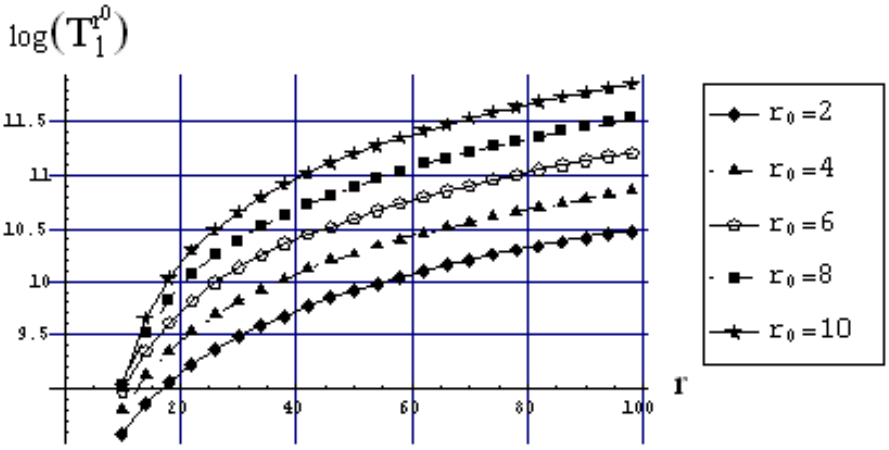


Рис. 2.15. Час виконання одного кроку інтегрування за технологією локальної екстраполяції з явним опорним методом порядку r_0

Таким чином, вибираючи симетричний явний метод Рунге-Кутти другого порядку, маємо час для виконання одного кроку інтегрування за схемою локальної екстраполяції рівним:

$$T_1^3 = \left(\frac{r^2 - 2r + 4}{2}\right) m \cdot T_F + \left[\left(\frac{21}{8}r^2 - \frac{42}{8}r + 8\right)m + \frac{r^2 - 2r + 4}{2}\right] \cdot t_{op}. \qquad (2.26)$$

2.3.2. Паралельні алгоритми розв'язання нелінійної задачі Коші на основі технології локальної екстраполяції

Потенційно обчислення за технологією локальної екстраполяції містять три джерела внутрішнього паралелізму:

- системний паралелізм (обмежений розмірністю СЗДР);
- паралелізм екстраполяції (обмежений розміром таблиці екстраполяції);
- паралелізм опорного методу (мала ступінь паралелізму).

Побудова обчислювальних схем паралельних алгоритмів розв'язання нелінійної задачі Коші для СЗДР за технологією Річардсона базується на декомпозиційній методиці. Після того, як визначені всі складові технології екстраполяції, процес розв'язання можна розбити на дві послідовно виконувані підзадачі обчислення k апроксимацій в точці x_{n+1} із різними кроками інтегрування: $\bar{y}_{h_i}(x_n + H)$ та побудова екстраполяціної таблиці. Організація паралельних обчислень може здійснюватися двома способами, звідси і дві принципово різні макрооперaційні схеми алгоритму. Перший варіант передбачає використання тільки системного паралелізму, другий - комбінацію паралелізму екстраполяції та системного.

Розглянемо першу підзадачу алгоритму. В якості основної макрооперації для обох варіантів уводиться одноразове обчислення апроксимації точного розв'язку в точці сітки з заданим кроком інтегрування на базі явного опорного методу. Граф впливу першого варіанту макрооперaційної обчислювальної схеми наведено на рисунку 2.16, усі апроксимації розв'язку обчислюються послідовно одночасно для кожної розмірності системи.

За другою макрооперaційною схемою паралельно обчислюється k незалежних апроксимацій розв'язку в точці $x_n + H$, граф впливу такої схеми наведено на рисунку 2.17. Другий варіант алгоритму має більший ступінь паралелізму, що дорівнює $Dop = m \cdot k$, на відміну від першого: $Dop = m$.

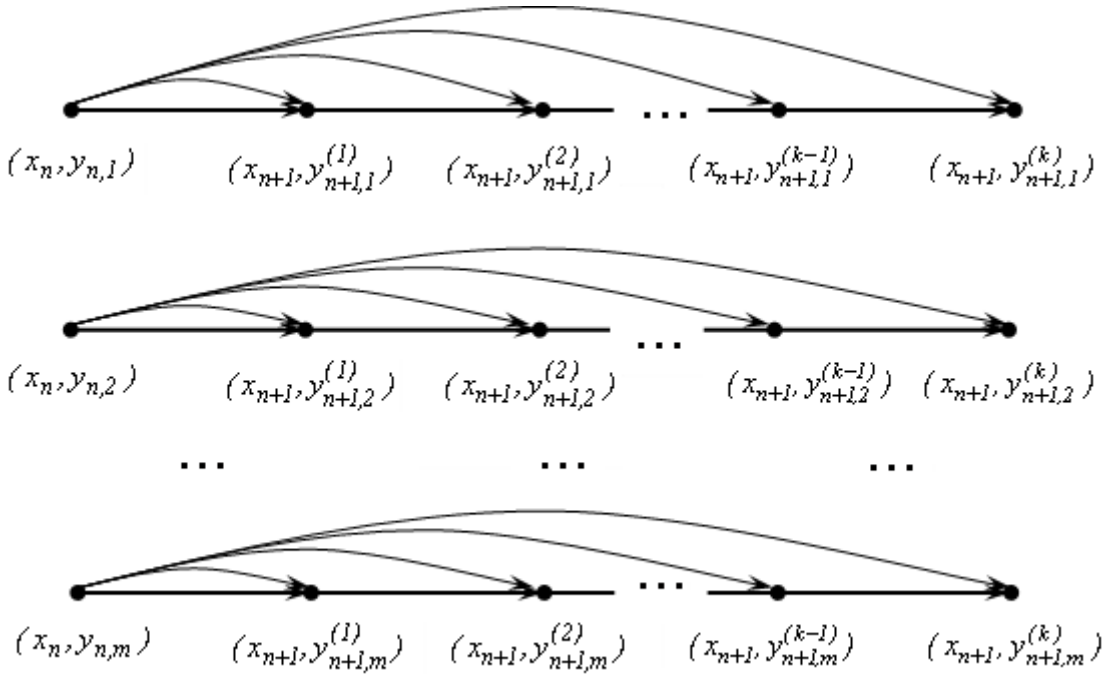


Рис. 2.16. Граф впливу макрооперaційної схеми для обчислення апроксимації розв'язку СЗДР (варіант №1)

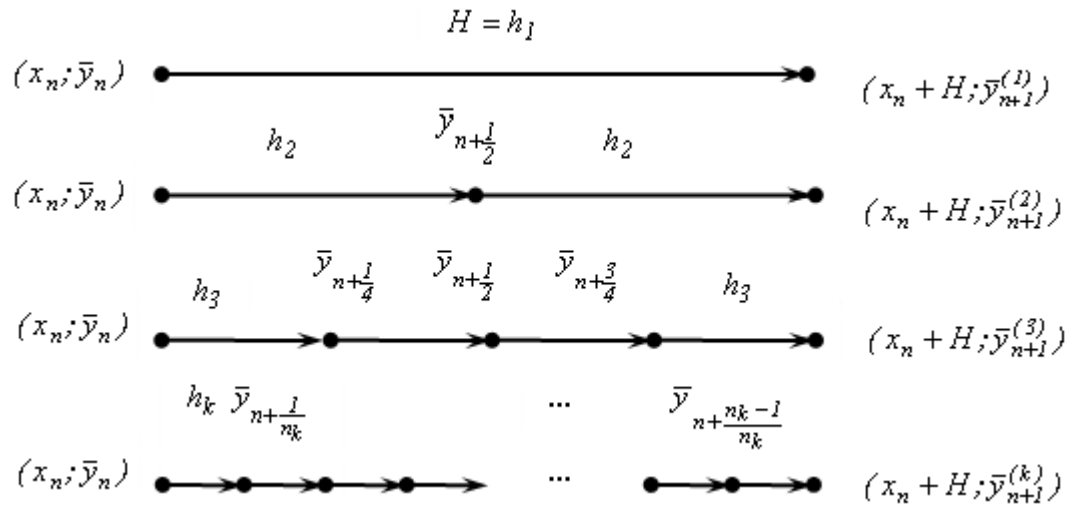


Рис. 2.17. Граф впливу макрооперaційної схеми для обчислення апроксимації розв'язку СЗДР (варіант № 2)

Проте, цей недолік схеми № 1 може бути компенсований кращим збалансуванням завантаження багатопроцесорної обчислювальної системи, а також низькою інтенсивністю обмінів. Обчислення значень екстраполяційної таблиці в залежності від способу розв'язання першої підзадачі, також може

бути распаралелено різними способами. В якості основної макрооперації для цієї підзадачі вводиться обчислення одного значення $T_{il,j}, j = \overline{1,m}, i = \overline{2,k}; l = \overline{i,k}$ за формулою поліноміальної екстраполяції Ейткена-Невілла (1.21). При побудові екстраполяційної таблиці розпаралелювання може бути здійснено за рахунок незалежного виконання обчислень за формулою поліноміальної екстраполяції по кожній розмірності системи ($Dop = m$), а також за рахунок розподіленого обчислення кожного рядка таблиці ($Dop = k - 1$). Як правило, $m > k$, тому з'являється можливість поєднати обидва види паралелізму. Граф впливу для методу побудови екстраполяційної таблиці наведено на рисунку 2.18.

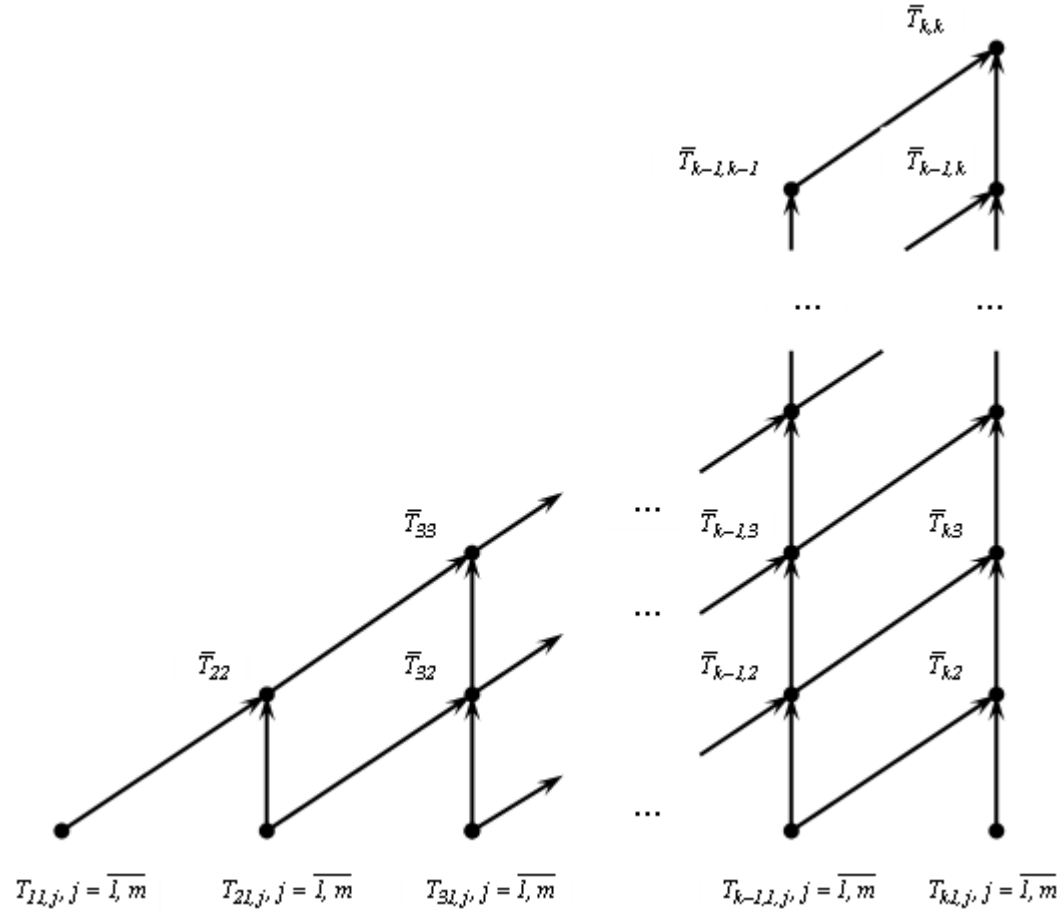


Рис. 2.18. Граф впливу методу побудови екстраполяційної таблиці

На розглянутому етапі деталізації дано попередній аналіз ефективності кожної з обчислювальних схем. Перейдемо до використання внутрішнього

паралелізму методу. Перша макрооперaційна схема алгоритму локальної екстраполяції повторює макрооперaційну схему № 1 для правила Рунге, із тією різницею, що тепер у нас не три, а k послідовно обчислюваних апроксимацій за ЯМРК. Рисунок 2.19 представляє паралельний алгоритм розв'язання системи звичайних диференційних рівнянь на мультикомп'ютері з p процесорів для першої макрооперaційної схеми. Для СЗДР, що складається з m рівнянь, на одному часовому кроці m_1, $m_1 = \lceil m/p \rceil$ компонент кожної апроксимації розв'язку може бути обчислено паралельно. Відповідно, при побудові екстраполяційної таблиці сі значення $T_{il}, i = \overline{2,k}; l = \overline{i,k}$ визначаються послідовно, але на кожному процесорі обчислюється одна при $p = m$ або $m_1 = \lceil m/p \rceil$ при $p < m$ компонент СЗДР.

Час обчислень за паралельною схемою № 1 складає: $T_p^{31} = T_p^{r_0} \cdot \sum_{i=1}^{k} n_i + T_p^{31,ext-tab}$, де $T_p^{r_0}$ – час паралельного обчислення опорного методу порядку r_0, а $T_{p,comp}^{31,ext-tab}$ – час обчислення екстраполяційної таблиці на p процесорах із використанням першої схеми локальної екстраполяції. Як результат, при використані h^2-екстраполяції маємо:

$$T_{p,comp}^{31} = \frac{(r^2 - 2 \cdot r + 4)}{2} m_1 T_F + \left[\frac{r^2 - 2r + 4}{2} + \left(\frac{21}{8} r^2 - \frac{42}{8} r + 8 \right) m_1 \right] \cdot t_{op}; \qquad (2.27)$$

$$T_{p,comm}^{31} = \frac{(r^2 - 2 \cdot r + 4)}{2} T_{all-to-all}(m_1, p). \qquad (2.28)$$

Міжпроцесорний обмін при першому варіанті розпаралелювання визначається виконанням $N(k)$ разів операції множинного пересилання даних, необхідної при обчисленні коефіцієнтів $\bar{k}_l, l = \overline{1,s}$. Перевагою розглянутої схеми є відсутність необхідності на кожному кроці перегруповувати дані для наступного кроку ітерації, тобто всі обчислення за m_1 компонентами локалізовано на одному процесорі, включаючи й обчислення для таблиці екстраполяції.

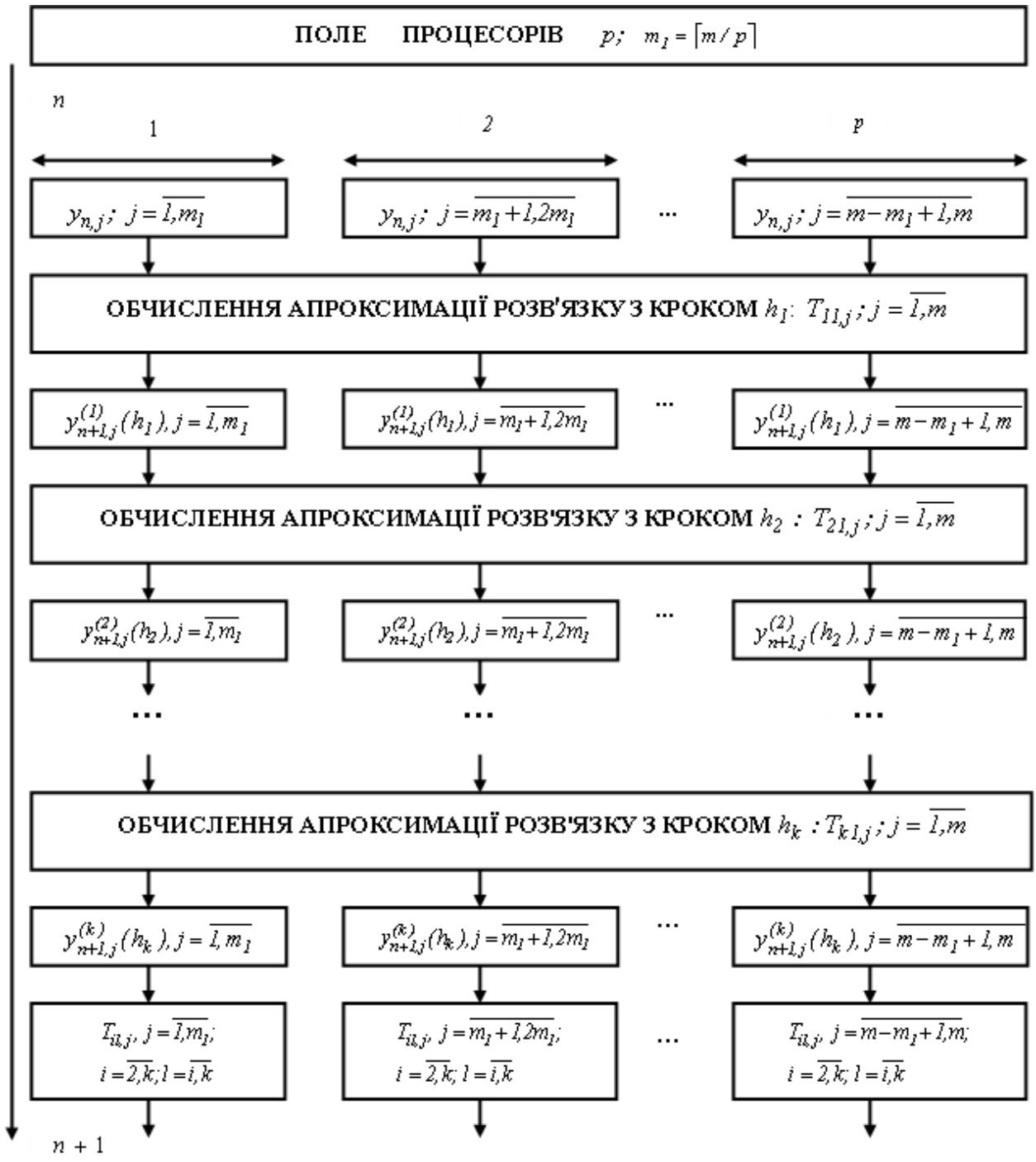


Рис. 2.19. Паралельний алгоритм розв’язання нелінійної задачі Коші на базі локальної екстраполяції з явним однокроковим опорним методом (варіант №1)

На рисунку 2.20 представлено паралельний алгоритм розв'язання СЗДР для макрооперaційної схеми № 2. Якщо кількість процесорів велика, є можливість виконувати макрооперації першої підзадачі паралельно.

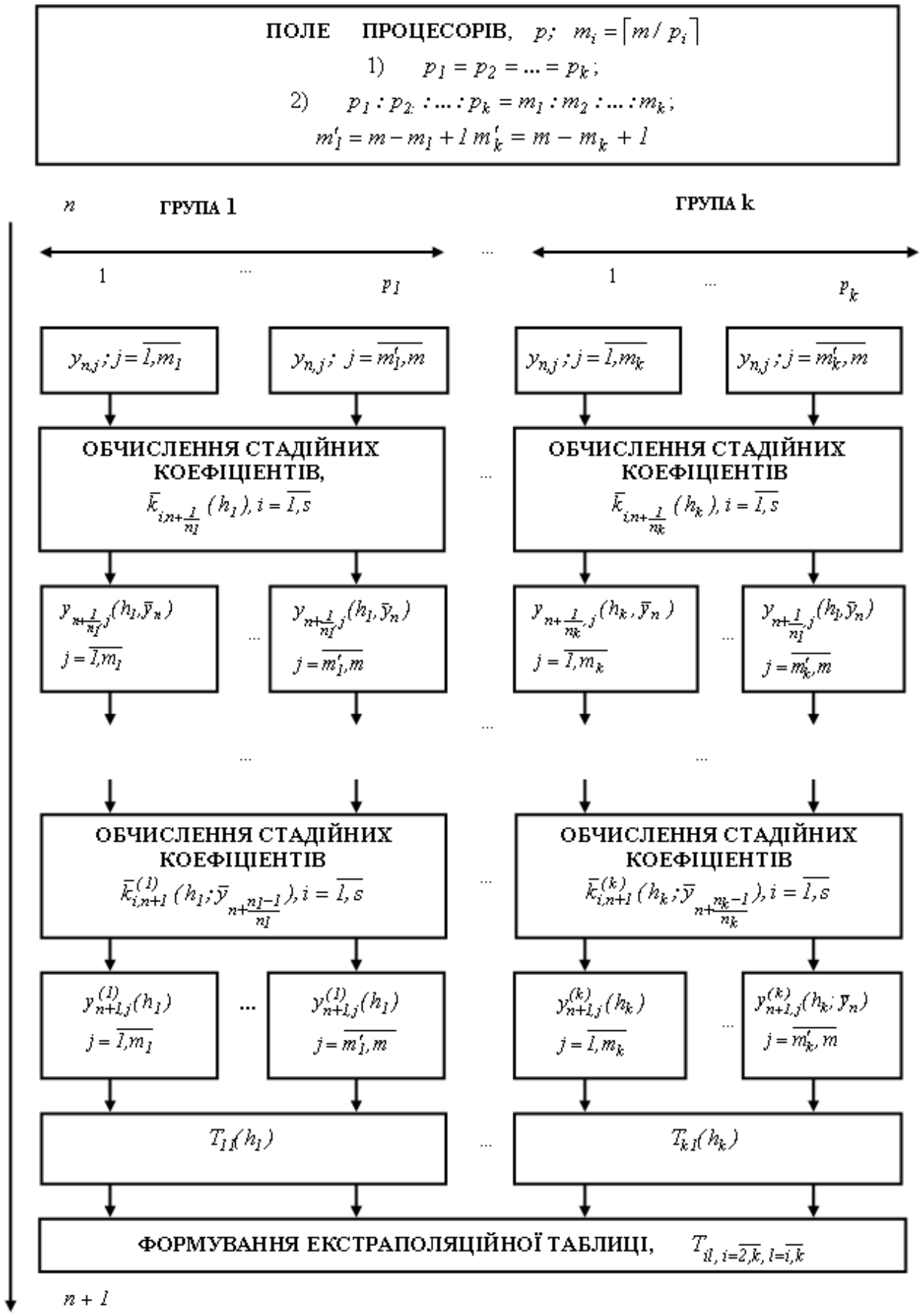


Рис. 2.20. Обчислювальні схеми паралельного алгоритму №2 на базі технології Річардсона (рівномірне та пропорційне розбиття)

Для цього необхідно розбити процесори на групи таким чином, щоб забезпечити найбільш раціональне збалансування завантаження багатопроцесорної системи. Розглянемо наступні способи розбиття процесорів, а відповідно й розподілу даних за процесорами: рівномірний та пропорційний. При найпростішому, рівномірному способі, кожен процесор або процесорна група будуть відповідати за обчислення однакової кількості апроксимацій. Якщо розмірність процесорного поля дорівнює p, кількість рядків у екстраполяційній таблиці - k, то при такому способі буде k груп із процесорів $p_i = \lceil p/k \rceil$, $i = \overline{1,k}$ у кожній. Кожен процесор в i-тій групі буде містити $m_i = \lceil m/p_i \rceil$ компонент відповідної апроксимації розв'язку та відповідного екстрапольованого значення. Кожна група буде відповідати за обчислення певної апроксимації розв'язку: $T_{j1}, j = \overline{1,k}$, час вирішення визначається як максимум із $T_{i1}, i = \overline{1,k}$, усі групи, крім останньої, будуть простоювати деякий час.

Визначимо час, що необхідний для організації паралельного процесу виконання першої підзадачі, як: $T_p^{32} = \max_{i=1}^{k}(n_i \cdot T_p^{r_0}) + T_p^{32,ext-tab}$. Оскільки при відомому опорному методі час $T_p^{r_0=2}$ не залежить від номера апроксимації, час виконання арифметичних операцій складає: $T_{p,comp}^{32} = n_k \cdot T_p^{r_0=2} + T_{p,comp}^{32,ext-tab}$. Для послідовності $P_2 : n_k = n_{r/2} = r-2$ та $m_k = m/p_k = mr/2p$, отже,

$$T_{p,comp}^{32} = (r^2 - 2r)\frac{m}{p}T_F + (4r^2 - 8r)\frac{m}{p}t_{op} + (2r-4)t_{op} + T_{p,comp}^{32,ext-tab}. \qquad (2.29)$$

Час на реалізацію міжпроцесорного обміну також визначається як максимальне значення часу обміну за всіма апроксимаціями розв'язку: $T_{p,comm}^{32} = n_k T_{all-to-all}(m_k, p_k) + T_{p,comm}^{32,ext-tab}$;

$$T_{p,comm}^{32} = (r-2)T_{all-to-all}\left(\frac{mr}{2p}, \frac{2p}{r}\right) + T_{p,comm}^{32,ext-tab}. \qquad (2.30)$$

Розглянемо різні варіанти організації паралельних обчислень екстраполяційної таблиці при рівномірному розподілі для обчислювальної схеми № 2. Після паралельного обчислення k апроксимацій розв'язку на k групах процесорів, існує наступний розподіл вхідних даних для обчислення екстрапольованих значень. Перша група процесорів містить вектор $T_{11,j}; j = \overline{1,m}$, друга – $T_{21,j}; j = \overline{1,m}$ і, відповідно, k-та – $T_{k1,j}; j = \overline{1,m}$. Кожен процесор у групах містить однакову m_i кількість компонент. Для обчислень елементів екстраполяційної таблиці може бути реалізовано два варіанти. Перший полягає у використанні системного паралелізму, кожен процесор групи передає відповідному процесору сусідньої групи всі компоненти власного вектора апроксимації розв'язку. Так, перший процесор першої групи передасть першому процесору другої групи подвектор $T_{11,j}; j = \overline{1,m_1}$: для обчислення $T_{22,j}; j = \overline{1,m_2}$; другий процесор першої групи передає $T_{11,j} j = \overline{m_1 + 1,2m_1}$ другому процесору другої групи для обчислення $T_{22,j}; j = \overline{m_2 + 1,2m_2}$ і так далі, перший процесор $(k-1)$ групи передає першому процесору k групи значення $T_{k-1,1,j} j = \overline{1,m_{k-1}}$ для обчислення $T_{k2,j}; j = \overline{1,m_k}$. Висота такого паралельного алгоритму дорівнює $k-1$, тобто на першому кроці обчислюються підвектори другого рядка таблиці екстраполяції, на останньому $k-1$ кроці обчислюються відповідні підвектори k-того рядка таблиці екстраполяції.

Час на організацію обчислень алгоритму в процесі формування екстраполяційної таблиці дорівнює:

$$T_{p,comp}^{32,1,ext-tab} = (k-1)m_i \cdot T_1^{ext} = 5m_i(k-1)t_{op} = 5 \cdot \frac{mr}{2p}\left(\frac{r-2}{2}\right)t_{op}, \qquad (2.31)$$

$$T_{p,comp}^{32,1} = (r^2-2r)\frac{m}{p}T_F + (4r^2-8r)\frac{m}{p}t_{op} + (2r-4)t_{op} + 5 \cdot \frac{mr}{2p}\left(\frac{r-2}{2}\right)t_{op}. \qquad (2.32)$$

У той же час міжпроцесорного обміну потребує $k-1$ операції передачі даних за типом "крапка-крапка" і, для підготовки наступного кроку інтегрування, одноразової передачі даних від кожного процесора останньої групи його підвектора розв'язку кожному відповідному процесору інших груп:

$$T_{p,comm}^{32,1,ext-tab} = (k-1)T_{p-p}(m_i, p) = \frac{r-2}{2} \cdot T_{p-p}\left(\frac{mr}{2p}, p\right). \quad (2.33)$$

$$T_{p,comm}^{32} = (r-2)T_{all-to-all}\left(\frac{mr}{2p}, \frac{2p}{r}\right) + \frac{r-2}{2} \cdot T_{p-p}\left(\frac{mr}{2p}, p\right) + T_{single-brodcast}(m_i, p_i) \quad (2.34)$$

Другий алгоритм обчислення екстраполяційної таблиці полягає в наступному: у кожній групі процесорів рівно один процесор буде відповідати за передачу даних між групами. Для цього після підрахунку елементів першого стовпця таблиці в кожній групі процесорів проводиться обмін значеннями за типом "усі-усім", у результаті кожен процесор у групі буде містити весь вектор апроксимації розв'язку, а не його частину: процесори першої групи $\overline{T}_{11}$, другої – $\overline{T}_{21}$ та k-тої – $\overline{T}_{k1}$. Потім i-тий процесор кожної групи, крім останньої, передає всім процесорам сусідньої групи вектор розв`язків, щоб підрахувати екстрапольовані значення. Далі кожен процесор кожної групи крім першої (ширина алгоритму зменшується з кожним кроком) обчислює свій підвектор (довжиною m_i) власного вектора екстрапольованих значень: процесори другої групи – $\overline{T}_{22}$, третьої – $\overline{T}_{32}$ та k-тої – $\overline{T}_{k2}$. На останньому $k-1$ кроці будуть працювати тільки процесори останньої групи, і кожен буде обчислювати m_k значень вектора $\overline{T}_{kk}$.

Таким чином, час на формування екстраполяційної таблиці за описаним алгоритмом складе:

$$T_{p,comp}^{32,2,ext-tab} = (k-1)m_i \cdot T_1^{ext} = 5m_i(k-1)t_{op};$$

$$T_{p,comp}^{32,2,ext-tab} = 5\frac{mr}{2p}\left(\frac{r-2}{2}\right)t_{op} = 1{,}25\frac{mr}{p}(r-2)t_{op}, \quad (2.35)$$

а час комунікацій дорівнює:

$$T_{p,comm}^{32,2,ext-tab} = T_{p,comm}^{32,2,in-gr} + T_{p,comm}^{32,2,over-gr} =$$
$$= (k-1)T_{all-to-all}(m_i, p_i) + (k-1)T_{single-brodcast}(m, p_i);$$

$$T_{p,comm}^{32,2,ext-tab} = \left(\frac{r-2}{2}\right)T_{all-to-all}\left(\frac{mr}{2p}, \frac{2p}{r}\right) + \left(\frac{r-2}{2}\right)T_{single-brodcast}\left(m, \frac{2p}{r}\right). \quad (2.36)$$

Щоб збільшити збалансування завантаження, розіб'ємо процесорне поле на таку ж кількість груп, але не рівномірно, а пропорційно до чисел: $n_i, i = \overline{1,k} : p_1 : p_2 : \ldots : p_k = n_1 : n_2 : \ldots : n_k, \sum_{i=1}^{k} p_i = p, \quad p_i = pn_i / N(k).$

Алгоритм розбиття на групи наступний: якщо кількість процесорів p не можна поділити на $N(k)$ без остачі, то береться $p_i = \lfloor pn_i / N(k) \rfloor$, а потім процесори, що залишилися по одному додаються до кожної з груп. Таким чином, кожна група буде містити тим більшу кількість процесорів, чим більше кроків інтегрування на базовому кроці їй належить виконати для отримання апроксимації розв'язку.

Оцінимо час виконання паралельного алгоритму за умови, що p поділяється без остачі на величину $N(k)$, тим самим заздалегідь збільшуючи час виконання за описаною схемою (для простоти міркувань):

$$T_{p,comp}^{33} = \max_{i=1}^{k}(n_i \cdot T_{p_i}^{EMRK,r^0}) + T_{p,comp}^{33,ext-tab},$$

$$T_{p,comp}^{33} = \frac{m}{p}\left(\frac{r^2-2r+4}{2}\right)T_F + 2\frac{m}{p}(r^2-2r+4)t_{op} + 2(r-2)t_{op} + T_{p,comp}^{33,ext-tab}. \quad (2.37)$$

З двох розглянутих варіантів отримання екстраполяційної таблиці, саме другий підходить для пропорційного розбиття процесорів, оскільки не вимагає додаткового сортування даних через нерівномірність їх розподілу.

$$T_{p,comp}^{33,ext-tab} = \sum_{j=2}^{k}\left(\max_{i=2}^{k} t_{T_{ij}}\right) = \sum_{j=2}^{k} m_i \cdot T_1^{ext} = 5\frac{mN}{p}\cdot\sum_{i=2}^{k}\frac{1}{n_i}\cdot t_{op} =$$
$$= 2{,}5\frac{m}{p}\cdot\left(\frac{r^2-2r+4}{2}\right)\cdot\sum_{i=2}^{r/2}\frac{1}{n_i}\cdot t_{op} = 2{,}5\frac{m}{p}\cdot\left(\frac{r^2-2r+4}{2}\right)\cdot\sum_{i=2}^{r/2}\frac{1}{2(i-1)}\cdot t_{op}. \quad (2.38)$$

$$T_{p,comm}^{33,ext-tab} = (k-1)\cdot\max_{i=1}^{k} T_{all-to-all}(m_i, p_i) + (k-1)\cdot T_{single-broadcast}(m, p_k). \quad (2.39)$$

Тоді загальний час на комунікаційні витрати за пропорційним способом розбиття даних складає:

$$T_{p,comm}^{33} = n_k \cdot T_{all-to-all}(m_k, p_k) + T_{p,comm}^{33,ext-tab} = \\ = (r-2) \cdot T_{all-to-all}\left(\frac{mN}{pn_k}, \frac{pn_k}{N}\right) + T_{p,comm}^{33,ext-tab} \quad (2.40)$$

Час обчислень та обмінів для третьої обчислювальної схеми з урахуванням RISC-архітектури становить:

$$T_{p,comp}^{33} = \frac{m}{p}\left(\frac{r^2-2r+4}{2}\right) \cdot T_F + 2\frac{m}{p}(r^2-2r+4) \cdot t_{op} + 2(r-2) \cdot t_{op} + \\ + 2{,}5\frac{m}{p} \cdot \left(\frac{r^2-2r+4}{2}\right) \cdot \sum_{i=2}^{r/2} \frac{1}{2(i-1)} \cdot t_{op}. \quad (2.41)$$

$$T_{p,comm}^{33,ext-tab} = (k-1) \cdot \max_{i=1}^{k} T_{all-to-all}(m_i, p_i) + (k-1) \cdot T_{single-broadcast}(m, p_k). \quad (2.42)$$

Максимальний ступінь паралелізму для другої та третьої схем збігається та дорівнює: $Dop = p_{max} = m \cdot k$. Скоротити час простою можна при використанні комбінаційного способу. Ідея комбінаційного методу полягає у використанні $\lceil k/2 \rceil$ груп процесорів, причому i-та група обчислює i-ту та ($k-i+1$)-ту апроксимації розвязку. Для отримання i-тої апроксимації необхідно n_i разів виконати звернення до опорного методу вирішення, для ($k-i+1$) -тої – n_{k-i+1} разів. Тобто перша група повинна $n_1 + n_k$ разів звернутися до обчислення рішення щодо опорного методу, друга група – $n_2 + n_{k-1}$ та, нарешті, $k1 = \lceil k/2 \rceil$ група – $n_{k1} + n_{k1+1}$. Нагадаємо, що для P_2: $\{1,2,4,6,8,10,...\}$.

У таблиці 2.3 наведено дані розбиття процесорного поля на групи в разі, якщо довжина екстраполяційної таблиці – парне число та використовується друга парна послідовність, яка формує сітки інтегрування.

Очевидно, що існує практично рівномірне забезпечення завантаження процесорних груп, окрім першої, яка відрізняється від інших незалежно від довжини таблиці на одиницю.

У випадку непарної довжини таблиці розбиття стає менш рівномірним за рахунок непарної останньої групи.

Таблиця 2.3

Розбиття процесорів на групи для P_2 та парної довжини екстраполяційної таблиці, k

№ групи k	1	2	3	4	5	6
6	11	10	10			
8	15	14	14	14		
10	19	18	18	18	18	
12	23	22	22	22	22	22

На рисунку 2.21 представлено паралельний алгоритм розв'язання системи нелінійних диференційних рівнянь на мультикомп'ютері з p процесорів для макрооперацийної схеми № 2 з комбінаційним розбиттям процесорів та даних. Кількість процесорів у кожній групі дорівнює:

$$p_i = \lceil p/k1 \rceil = \lceil 2p/k \rceil = \lceil 4p/r \rceil$$

та кількість компонент вектора розв'язку для обчислення певної апроксимації: $m_i = \lceil m/p_i \rceil = \lceil mr/4p \rceil$.

У цьому випадку час реалізації паралельних обчислень апроксимацій розв'язку задачі Коші за схемою локальної екстраполяції із комбінаційним способом розбиття складає:

$$T_{p,comp}^{34} = \max_{i=1}^{k/2}(n_i + n_{k-i+1}) \cdot T_{p_i}^{r^0=2} + T_{p,comp}^{34,ext-tab} =$$

$$= (r^2 - r) \cdot (m/2p) \cdot T_F + 2(r^2 - r)(m/p) \cdot t_{op} + 2(r-1) \cdot t_{op} + T_{p,comp}^{34,ext-tab}.$$

На обчислення екстраполяціонной таблиці потрібно:

$$T_{p,comp}^{34,ext-tab} = [5(3r-8)/16p]mr \cdot t_{op}.$$

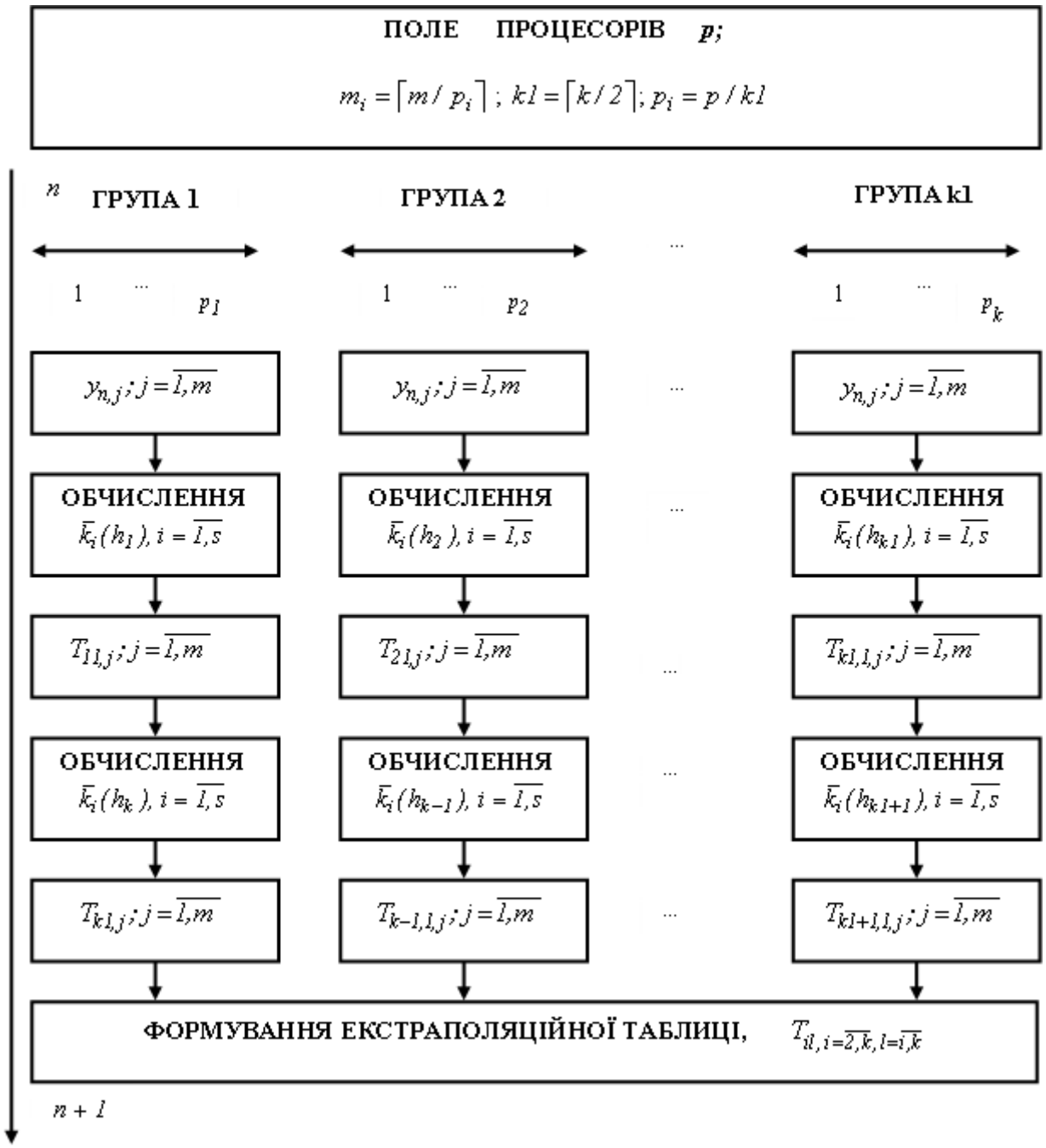


Рис. 2.21. Обчислювальна схема паралельного алгоритму на основі технології локальної екстраполяції (комбінаційне розбиття)

У цілому час арифметичних операцій для третьої схеми дорівнює:

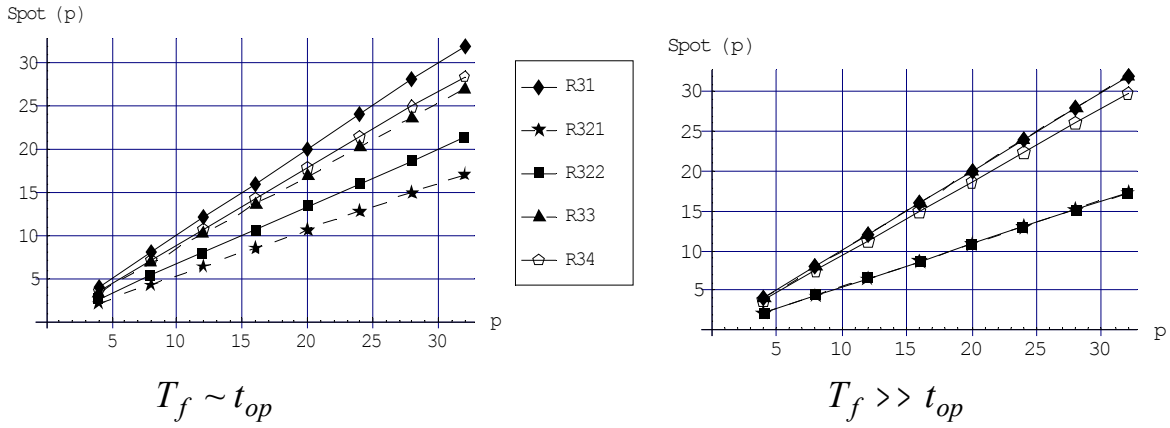
$$T_{p,comp}^{34} = (r^2 - r) \cdot \frac{m}{2p} T_F + (17r^2 - 4.5r) \cdot \frac{m}{p} t_{op} + 2(r-1) \cdot t_{op}. \quad (2.43)$$

Час обмінів становить:

$$T_{p,comm}^{34} = (n_1 + n_k) T_{all-to-all}(m_i, p_i) + T_{p,comm}^{34,ext-tab},$$

$$T_{p,comm}^{34} = (r-1) T_{all-to-all}\left(\frac{mr}{4p}, \frac{4p}{r}\right) + (1.5r - 4) T_{p-p}\left(\frac{mr}{4p}, \frac{4p}{r}\right) + T_{single-to-all}(m_1, p - p_1). \quad (2.44)$$

Дослідження потенційного паралелізму отриманих обчислювальних схем приведено на рис. 2.22. Перша обчислювальна схема незалежно від складності правої частини володіє кращими характеристиками прискорення. Найгіршим потенційним прискоренням володіють варіанти схеми 2 незалежно від складності правої частини.



$T_f \sim t_{op}$ $T_f >> t_{op}$

Рис. 2.22. Коефіцієнт потенційного прискорення методів від кількості процесорів

Оцінимо комунікаційну складову паралельних алгоритмів для різних типів паралельних ОС і топологій з'єднання процесорів. На рисунку 2.23 наведено типовий графік залежності долі часу обмінів до загальних накладних витрат на паралелізм для першої обчислювальної схеми. Незалежно від складності правої частини СЗДР і типу паралельної ОС топологія гіперкуб також є найефективнішою для всіх обчислювальних схем, а топологія кільце вимагає найбільше часу на реалізацію операцій обміну.

Проведемо порівняння обчислювальних схем на основі реальних коефіцієнтів прискорення та ефективності для топології гіперкуб. Рівномірний спосіб розбиття процесорів на групи, незважаючи на простоту реалізації, має великі накладні витрати, у тому числі й непродуктивні, має значно гірші характеристики паралелізму і тому далі обчислювальна схема № 2 не розглядається.

Рис. 2.23. Доля операцій обміну алгоритму № 1 до загальних накладних витрат при різних топологіях

Перша схема, що використовує системний паралелізм, має певні переваги, вона може бути реалізована на паралельному комп'ютері будь-якого типу, має добре збалансування завантаження процесорів. Із аналізу рисунку 2.24 випливає, що 1 та 3 схеми при складних правих частинах мають добрі показники ефективності для систем, як із високошвидкісними каналами передачі інформації, так і з низькошвидкісними.

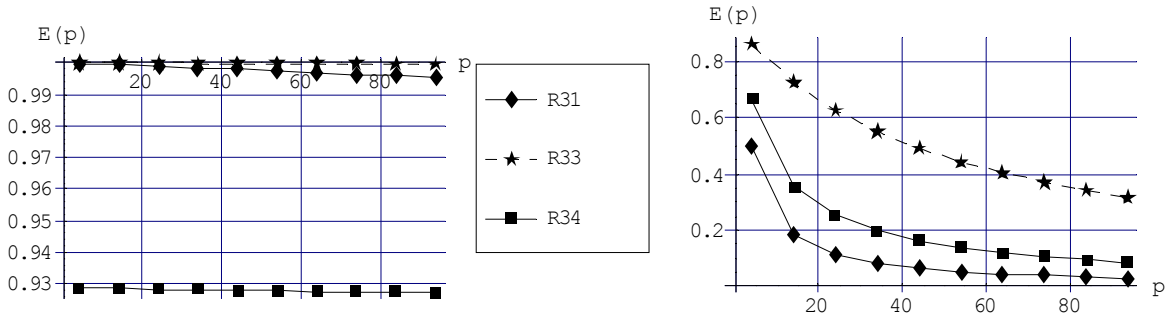


Рис. 2.24. Коефіцієнти реальної ефективності методів від кількості процесорів, $r = 8$, $m = 1000$; 1) $T_f >> t_{op}$; 2) $T_f \sim t_{op}$

Суттєвим параметром для цих схем є складність правої частини СЗДР, що забезпечує домінування обчислень над обмінами. Для нескладних правих частин третя обчислювальна схема, яка об'єднує паралелізм системи та екстраполяції, має явну перевагу, і цей ефект із зростанням числа процесорів збільшується. Для SIMD-систем, як і раніше, може бути реалізована тільки перша схема, яка має низькі показники прискорення та ефективності.

2.4. Порівняння ефективності явних паралельних методів рішення нелінійних СЗДР із альтернативними способами визначення локальної похибки

Порівняння обчислювальної складності трьох різних паралельних алгоритмів вирішення нелінійної задачі Коші на основі ЯМРК із вбудованими способами оцінки локальної апостеріорної похибки коректно, якщо ці методи дають наближений розв'язок одного й того ж порядку точності [52-54]. В явному вигляді порядок наближеного чисельного методу в оцінках накладних витрат на паралелізм не присутній, однак він може бути оцінений через параметр s, кількість стадій явних однокрокових методів Рунге-Кутти.

У таблиці 2.4 вказано мінімальна кількість стадій s, що є необхідна для однокрокового методу Рунге-Кутти на основі явної схеми порядку r. Очевидно, що для явних методів малих і середніх порядків $r \le 6$ кількість стадій дорівнює порядку або на одиницю перевищує його. Для методів більш високих порядків ця різниця більше одиниці та зростає зі зростанням порядку методу. Взаємозв'язок між цими величинами для перерахованих методів визначається на основі бар'єрів Батчера [93-95].

Таблиця 2.4

Зв'язок порядку методу та кількості стадій для відомих ЯМРК

Порядок методу, r	$r \le 4$	5	6	7	8	10
Мінімальна кількість стадій, s	r	6	7	9	11	17

У таблиці 2.5 представлено відомі вкладені методи типу Рунге-Кутти на основі явних схем різних порядків точності та відповідна кількість стадій для кожного з них. Обчислювальна складність ВЯМРК залежить від порядків методів, що складать вкладену пару $r(\hat{r})$, у свою чергу ці величини визначаються кількістю використовуваних стадій.

Порівняння накладних витрат для реалізації послідовних явних однокрокових методів із альтернативними способами оцінки локальної апостеріорної похибки показує, що вкладені методи забезпечують найбільш економний по тимчасовим характеристикам спосіб обчислення похибки на кроці.

Таблиця 2.5

Зв'язок порядку методу та кількості стадій для вкладених явних методів Рунге-Кутти (ВЯМРК)

Вкладені явні методи Рунге-Кутти $r(\hat{r})$	Порядок методу, $r(\hat{r})$	Кількість стадій, s
Фельберга 4(5)	4(5)	6
Дормана-Принса 5(4)	5(4)	7
Фельберга 7(8)	7(8)	12
Дормана-Принса 8(7)	8(7)	13

Порівняння паралельних алгоритмів визначення локальної похибки на основі явних однокрокових схем проводиться за динамічними характеристиками, які є функціями машинно-залежних констант та властивостями і параметрами систем звичайних диференційних рівнянь (табл. 2.6-2.7).

Від співвідношення цих параметрів та їх комбінацій залежить ефективність отриманих паралельних методів. Найбільш ефективний послідовний метод – це вкладений метод типа Рунге-Кутти, приведімо всі характеристики до нього. Відносні коефіцієнти ефективності обчислюються як:

$$T_1^{opt} = T_1^2; \quad E_i^{Re} = T_1^2 / pT_p^i,$$

де T_1^{opt} – час виконання найбільш оптимального послідовного алгоритму.

Таблиця 2.6

Обчислювальна складова ЯМРК із альтернативними способами оцінки локальної апостеріорної похибки

Метод оцінки локальної похибки	Коефіцієнт при T_F	Коефіцієнт при t_{op}
Правило Рунге	$(3s-1)\frac{m}{p}$	$3\frac{m}{p}s^2+6\frac{m}{p}s+6s-6$
ВМРК	$(s+1)\frac{m}{p}$	$\frac{m}{p}s^2+6\frac{m}{p}s+2s+4\frac{m}{p}$
ЛЕР	$\left(\frac{s^2-2s+4}{2}\right)\cdot\frac{m}{p}$	$\left[\left(\frac{s^2-2s+4}{2}\right)+\left(\frac{21}{8}s^2-\frac{42}{8}s+8\right)\frac{m}{p}\right]$

Таблиця 2.7

Комунікаційна складова ЯМРК із альтернативними способами оцінки локальної апостеріорної похибки

Метод оцінки локальної похибки	Коефіцієнт при t_s	Коефіцієнт при t_w
Правило Рунге	$3s\log_2 p$	$3s\frac{m}{p}(p-1)$
ВМРК	$(s+1)\log_2 p$	$(s+1)\frac{m}{p}(p-1)$
ЛЕР	$\left(\frac{s^2-2s+4}{2}\right)\log_2 p$	$\left(\frac{s^2-2s+4}{2}\right)\frac{m}{p}(p-1)$

Аналіз даних таблиць (2.6-2.7), аналогічних теоретичних і експериментальних залежностей (рис. 2.26) для динамічних характеристик трьох способів визначення локальної похибки для явних однокрокових схем дозволяють зробити висновки:

1) паралельні вкладені методи для явних схем найменш витратні для будь-яких типів паралельних ОС та топологій з'єднання процесорних елементів;

2) практично одинична ефективність для ВМРК досягається для систем типу MIMD, топології гіперкуб і складних правих частин.

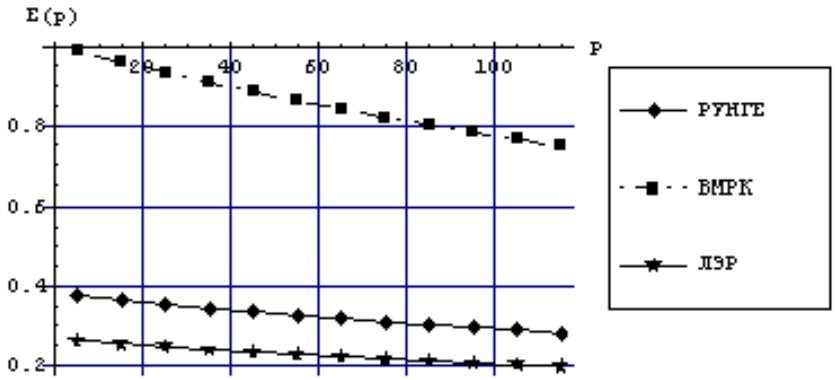


Рис. 2.26. Відносні коефіцієнти ефективності паралельних алгоритмів ЯМРК

Незважаючи на широке застосування, максимальний порядок відомих методів, що були розроблені на основі явних різницевих однокрокових схем, обмежений ($r \le 8$). Для інтегрування задач, що вимагають обчислення високоточного розв'язку (10^{-15} - 10^{-20}), не існує відповідних вкладених схем. Таким чином, метод локальної екстраполяції Річардсона має свою область застосування.

2.5. Дослідження масштабованості паралельних вкладених методів на основі явних однокрокових схем

Постановка питання про оцінку ефективності розпаралелювання при збільшенні кількості процесорів із збереженням обсягу обчислень не завжди є метою досліджень. Більше того, деякими авторами [2] вважається не обгрунтованною. Дійсно, зростання продуктивності системи в цілому обумовлено збалансованістю обчислювальної роботи та обмінів на її фоні.

Невиконання цієї умови – одна з причин погіршення якості паралелізму зі збільшенням розмірності процесорного поля. Тому висновки

слід робити на основі досягнення прийнятної ефективності при розпаралелюванні на кількість процесорів, фактично необхідну для розвязання задачі в задані часові терміни. Іншими словами, мова йде про оцінку масштабованості паралельної системи, що складається з алгоритму в комбінації з архітектурою, на якій він реалізований.

У даному підрозділі досліджується масштабованість паралельного алгоритму вкладених методів, як найбільш ефективного способу оцінки локальної похибки для явних однокрокових схем.

Математичним апаратом, що дозволяє вирішити поставлену задачу, є теорія ізоефективного аналізу [11-14]. Для побудови функції ізоефективності необхідно визначити загальні накладні витрати на паралельний алгоритм:

$$T_o^2 = (s+1)(p\log_2 p)\cdot t_s + m(s+1)(p-1)\cdot t_w + 2s(p-1)\cdot t_{op}. \qquad (2.43)$$

Тоді основне співвідношення ізоефективного аналізу для розглянутого алгоритму приймає вигляд:

$$T_1^2 = K\cdot T_o^2 \quad \Rightarrow$$

$$T_1^2 = (s+1)mT_F + (s^2m + 6sm + 2s + 4m)t_{op} = K\cdot T_o^2 \quad \Rightarrow \qquad (2.44)$$

$$T_1^2 = K[(s+1)(p\log_2 p)\cdot t_s + m(s+1)(p-1)\cdot t_w + 2s(p-1)\cdot t_{op}]. \qquad (2.45)$$

Використовуючи співвідношення (2.44-2.45), визначимо функцію ізоефективності, тобто знайдемо залежність $m = f_E(p)$, на основі якої можна обчислити, як необхідно збільшувати розмірність задачі при збільшенні кількості процесорів для підтримки постійної ефективності.

Визначимо розмірність СЗДР, визначальну складність вхідної задачі, як функцію від наступних параметрів, $m = f_E(p;T_F,t_{op},t_s,t_w;s;K)$:

- кількість процесорних елементів;
- тимчасові константи алгоритму й ОС;
- порядок і кількість стадій явного методу;
- коефіцієнт масштабованості.

Нагадаймо, що $K = E/1-E$.

$$m = \frac{K[(s+1)(p\log_2 p)\cdot t_s + 2s(p-1)\cdot t_{op}] - 2s\cdot t_{op}}{(s+1)\cdot T_F + (s^2+6s+4)\cdot t_{op} - K(s+1)(p-1)\cdot t_w}. \quad (2.46)$$

Проаналізуймо це співвідношення з урахуванням характеристик алгоритму та паралельної обчислювальної системи. Із рівності $Dop = p_{max} = m$ випливає, що кількість процесорів завжди повинна бути менша або рівна розмірності вхідної СЗДР щоб уникнути непродуктивних витрат (простою обладнання). Незалежно від співвідношення будь-яких параметрів, розмірність СЗДР повинна бути додатним числом. Виходячи з (2.46) очевидно, що чисельник дробу є число додатне, так як у самому гіршому випадку при $p = s = 2$ ця вимога виконується.

Знаменник (2.47) повинен бути строго більший нуля, тому в якості верхньої межі для кількості процесорів маємо наступний вираз:

$$p < \frac{(s+1)\cdot T_F + (s^2+6s+4)\cdot t_{op} + K(s+1)\cdot t_w}{K(s+1)\cdot t_w}. \quad (2.47)$$

Проаналізуємо співвідношення (2.46) із виведеними обмеженнями. За наведеною у розділі 1 класифікацією розглянутий алгоритм є квазілінійним щодо кількості процесорів, тобто має достатньо високу масштабованість. Рисунок 2.27 представляє коефіцієнт прискорення при 4 різних значеннях розмірності задачі для асинхронних паралельних систем із низькошвидкісними комунікаційними мережами: висока латентність і невелика швидкість передачі даних. Як і очікувалося, для малої розмірності $m \le O(100t_{op})$ прискорення досягає піку вже при невеликій кількості процесорів: $p \approx 100$. За цією точкою збільшення розміру процесорного поля істотного впливу на прискорення не надає. З іншого боку, прискорення для великих розмірностей задач близьке до лінійного.

Для MIMD-систем із швидкими каналами зв'язку масштабованість алгоритму підвищується, тобто прискорення зі зростанням кількості процесорів стає практично лінійним, особливо для задач великої розмірності.

Пік прискорення досягається при набагато більшій кількості процесорів, величина максимального можливого прискорення зростає.

Кількість процесорів, при якій відбувається насичення прискорення, прямо пропорційна складності вхідної задачі. Ця тенденція виявляється для паралельних систем будь-якої архітектури, ефективності комунікаційних мереж і топології з'єднання процесорів у них.

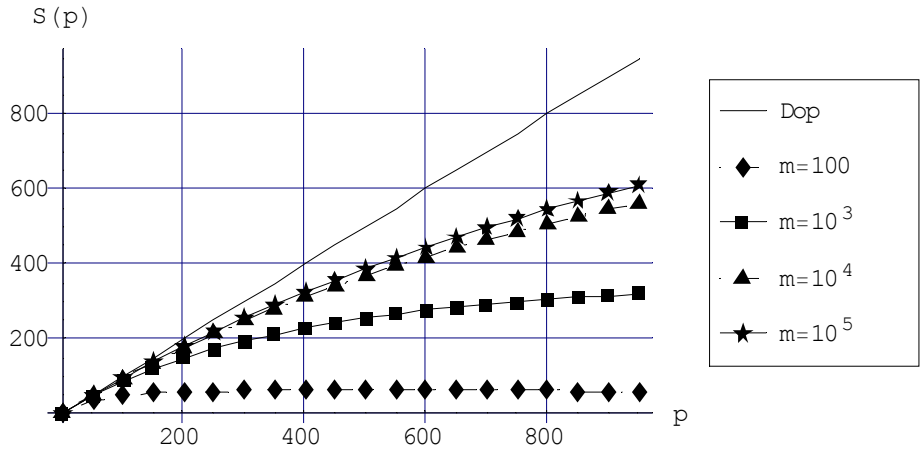


Рис. 2.27. Графіки залежності коефіцієнта прискорення ВЯМРК від кількості процесорів для паралельних MIMD-систем високої латентності

Розгляньмо вплив на величину функції ізоефективності декількох параметрів: коефіцієнта масштабованості, а, отже, достатньої ефективності, порядку досліджуваного методу, складності правої частини, при варіюванні кількості процесорів та характеристик комутаційної мережі (рис. 2.28-2.29).

Так, наприклад, щоб отримати середню ефективність використання паралельної системи ($E = 0,5$) при наявності кількості процесорів рівної $p = 400$ для низькошвидкісної мережі потрібно розв’язувати задачу розміру $m \approx 1000$, для високошвидкісної мережі при тих же даних маємо розмір значно менший $m \approx 100$.

Для отимання високої ефективності паралельної ОС ($E = 0,9$) при наявності кількості процесорів рівної $p = 400$ для низькошвидкісної мережі потрібно розв’язувати задачу розміру $m \approx 5000$.

Для високошвидкісної мережі при тих же даних маємо розмір значно менший $m \approx 600$.

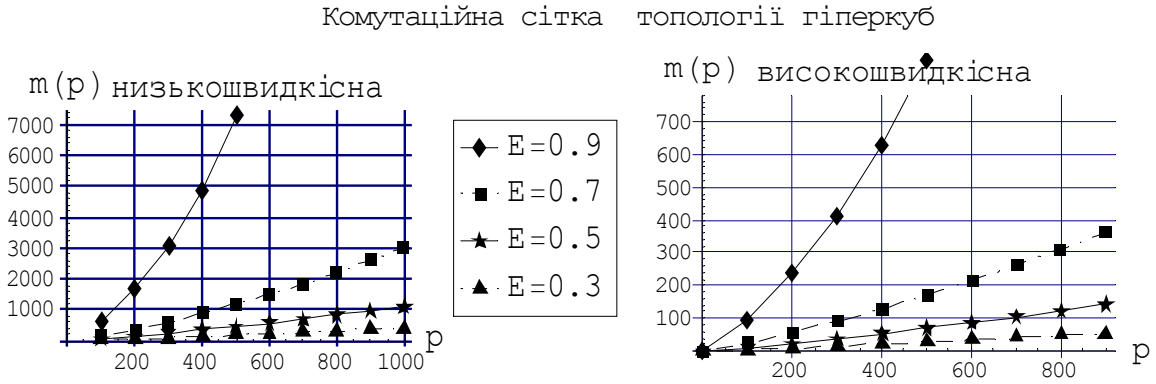


Рис. 2.28. Функція ізоефективності ВМРК для MIMD-систем при різних значеннях коефіцієнта ефективності

Тобто, знаючи функцію ізоефективності, можна визначити, якого розміру необхідно вирішувати задачі на наявній паралельної ОС, щоб ефективність використання обладнання досягала певного значення.

Так, при розмірності процесорного поля $p = 800$ та низькошвидкісних комутаційних мережах для досягнення ефективності 0.9 потрібна задача розміу не менше $m \approx 7000$. В той же час для $E = 0.3$ цей параметр меннше ($m \approx 500$). Досягнення тих же характеристик для швидких каналів передачі даних потребує вирішення задач практично в десять разів меншого розміру.

На рисунку 2.29 наведено графіки залежності функції ізоефективності від порядку для найбільш відомих вкладених методів і різних комунікаційних середовищ, причому коефіцієнт масштабованості дорівнює $K = 9$, що відповідає $E = 0.9$.

Очевидно, що чим менший порядок методу застосовано, тим більший розмір задачі необхідно реалізувати для досягнення однієї і тієї ж ефективності при однаковій кількості використовуваних процесорів.

Аналогічно, із зростанням складності правої частини СЗДР розмір розв'язуваної задачі зменшується (рис. 2.30).

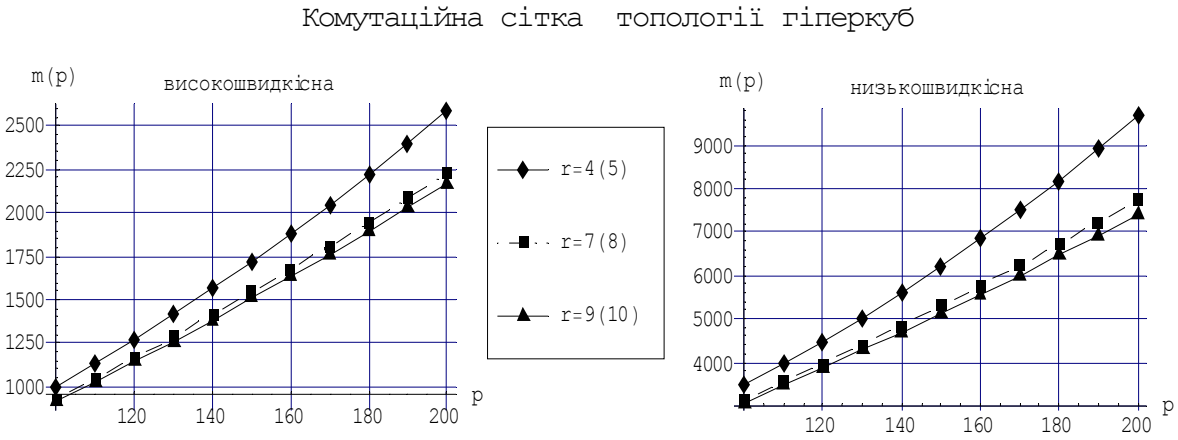


Рис. 2.29. Функція ізоефективності для MIMD-систем і вкладених явних методів різних порядків точності при $E = 0.9$,

1) Мерсона 4(5); 2) Фельберга 7(8); 3) Дормана-Принса 9(10)

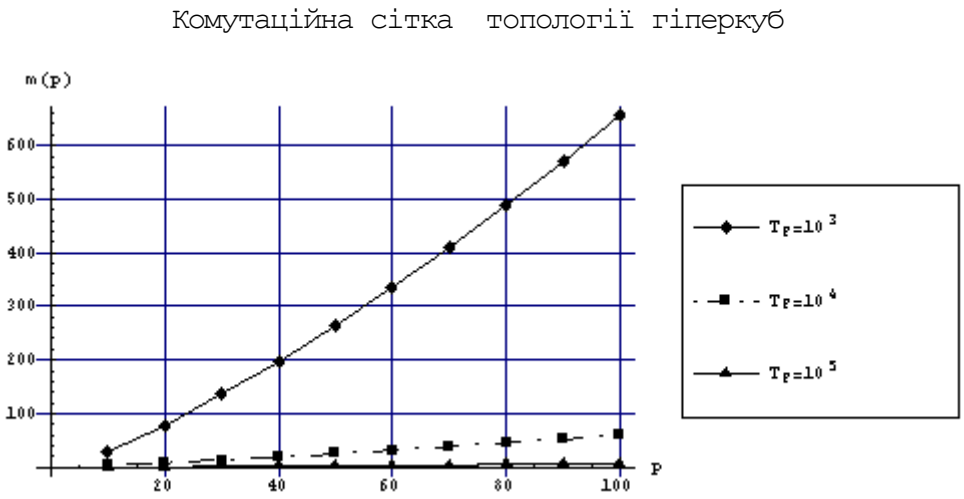


Рис. 2.30. Функція ізоефективності для MIMD-систем і вкладених явних методів різної складності првої частини СЗДР для низькошвидкісної комутаційної мережі

Застосування ізоефективного аналізу дозволяє зробити висновки щодо якості отриманого паралельного алгоритму на основі єдиного аналітичного виразу, скоротивши тим самим численні експерименти при різних комбінаціях параметрів, від яких залежать характеристики паралелізму.

ГЛАВА ІІІ

ПАРАЛЕЛЬНІ НЕЯВНІ ОДНОКРОКОВІ МЕТОДИ ЧИСЕЛЬНОГО РОЗВ'ЯЗАННЯ ЖОРСТКИХ ЗАДАЧ КОШІ

Дослідження методів розв'язання динамічних задач із зосередженими параметрами виявило, що паралельні властивості таких методів багато в чому визначаються видом чисельної схеми, покладеної в їх основу. Найменш трудомісткими є явні методи, проте властиві цим схемам недоліки, зокрема умовна стійкість, істотно обмежують сферу їх застосування. У зв'язку з цим значний інтерес представляють неявні схеми, які, не дивлячись на велику обчислювальну складність, не мають альтернативи серед однокрокових методів при вирішенні жорстких задач [94-95].

Розглядається чисельне розв'язання задачі Коші, асоційоване з розв'язанням СЗДР першого порядку з відомими початковими умовами:

$$\begin{cases} \dfrac{d\bar{y}(x)}{dx} = \bar{f}(x, \bar{y}(x)), \\ \bar{y}(x_0) = \bar{y}_0, \end{cases} \tag{3.1}$$

де права частина системи є в загальному випадку нелінійною функцією, що задає відображення $F = \bar{f} : R \times R^m \to R^m$.

3.1. Блокові паралельні методи інтегрування звичайних диференційних рівнянь з контролем локальної похибки

Блокові або багатоточкові паралельні методи розв'язання задачі Коші особливо актуальні, оскільки добре узгоджуються з архітектурою паралельних ОС і не вимагають обчислення значень у проміжних точках, що значно підвищує ефективність розрахунків. Дані методи володіють достатніми характеристиками стійкості і є по своїй суті паралельними [107-108], оскільки дозволяють отримувати розв'язок одночасно у декількох точках сітки інтегрування.

Множина точок рівномірної сітки $\Omega_h : \{x_j\}, j = \overline{1,M}$ розбивається на N блоків. Кожен блок містить k точок і при цьому $N \leq M$. Передбачається, що в межах блоку всі точки рівновіддалені одна від одної:

$$x_{n,i} = x_{n,o} + ih, i = \overline{1,k}, \tag{3.2}$$

де i – номер точки в блоці $i = \overline{1,k}$; n – номер блоку $n = \overline{1,N}$; $x_{n,i}$ – точка з номером i, що належить блоку n; $x_{n,0}$ – початкова точка n-го блоку; $x_{n,k}$ – кінцева точка n-го блоку. Множина точок n-го блоку з k точок позначається, як $T_n^{(k)}$. При цьому має місце рівність: $x_{n,k} = x_{n+1,0}$ (рис. 3.1).

Нехай $y_{n,0}$ є наближене значення розв'язку задачі Коші в точці $x_{n,0}$ – початковій точці оброблюваного блоку.

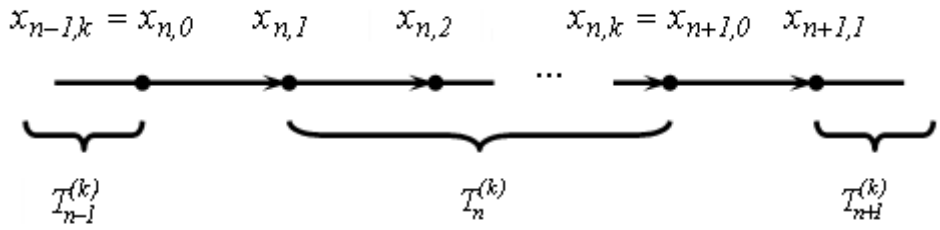


Рис. 3.1. Схема розбиття на блоки для однокрокового k-точкового методу

Рівняння однокрокових блокових різницевих методів у вживанні до ЗДР для блоку з k точок можуть бути записані таким чином:

$$y_{n,i} = y_{n,0} + ih\left[b_i F_{n,0} + \sum_{j=1}^{k} a_{i,j} F_{n,j}\right]; i = \overline{1,k}; n = \overline{1,N}. \tag{3.3}$$

Розкладом в ряд Тейлора функцій, що входять у нев'язку можна показати, що однокроковий k-точковий блоковий метод має найбільший порядок апроксимації, рівний $k+1$, отже, локальна похибка у вузлах блоку має порядок $O(h^{k+2})$ [109-113]. Блокові паралельні методи відносяться до класу неявних, тому для обчислення наближеного розв'язку задачі Коші необхідно вирішити систему, у загальному випадку нелінійних, рівнянь.

Одним із засобів одержання розв'язку є метод простої функціональної ітерації:

$$\begin{cases} y_{n,i,0} = y_{n,0} + ihF_{n,0},\ i = \overline{1,k},\ n = 1,2,...,N, \\ y_{n,i,l+1} = y_{n,0} + ih(b_i F_{n,0} + \sum\limits_{j=1}^{k} a_{i,j} F_{n,j,l}),\ \ l = \overline{0,L-1}, \end{cases} \tag{3.4}$$

де n – номер блоку, $n = 1,2,...,N$; i – номер точки блоку, $i = \overline{1,k}$;

l – номер поточної ітерації $l = \overline{0,L-1}$;

L - максимальне число ненульових ітерацій.

На відміну від явних методів розв'язання СЗДР, реалізація альтернативних засобів оцінки апостеріорної локальної похибки на основі блокових методів пов'язана з рядом особливостей:

– немає відповідних послідовних аналогів, отже, потрібно розробити і обґрунтувати метод оцінки локальної похибки;

— зміна кроку інтегрування можлива лише після виконання обчислень у всіх k вузлах поточного n-го блоку

– при умові незадовільної оцінки локальної похибки практично всі обчислення для точок блоку виявляться даремними (деякі звернення до правої частини СЗДР можуть бути використані знову).

3.1.1. Ефективність паралельної реалізації правила дублювання кроку в блокових однокрокових методах інтегрування ЗДР

Нехай розв'язання задачі Коші для ЗДР виконується на основі k-точкового однокрокового блокового методу. При реалізації правила дублювання кроку необхідно провести обчислення за однією й тією ж групою формул, що мають такий вигляд (3.3):

$$\begin{cases} y_{n,i}^{(1)} = y_{n,0}^{(1)} + ih \cdot \left[b_i f(x_{n,0}; y_{n,0}^{(1)}) + \sum\limits_{j=1}^{k} a_{i,j} f(x_{n,j}; y_{n,j}^{(1)}) \right], n = \overline{1,N},\ i = \overline{1,k} \\ y_{n,i}^{(2)} = y_{n,0}^{(2)} + i\frac{h}{2} \cdot \left[b_i f(x_{n,0}; y_{n,0}^{(2)}) + \sum\limits_{j=1}^{k} a_{i,j} f(x_{n,j}; y_{n,j}^{(2)}) \right], n = \overline{1,N1}, i = \overline{1,k} \end{cases} \tag{3.5}$$

на двох різних рівномірних сітках:

1) $\Omega_h = \{x_j\}, j = \overline{1,M}$ з кроком h у N блоках;

2) $\Omega_{h/2} = \{x_j\}, j = \overline{1,M1}$ з половинним кроком у $N1$ блоках.

На рисунку 3.2 приведена схема обчислень при використанні правила Рунге для однокрокового блокового k-точкового методу. Як і у попередньому розділі, апроксимація розв'язку з одинарним кроком позначається $y_{n,i}^{(1)}$, а з половинним, відповідно: $y_{n,i}^{(2)}$.

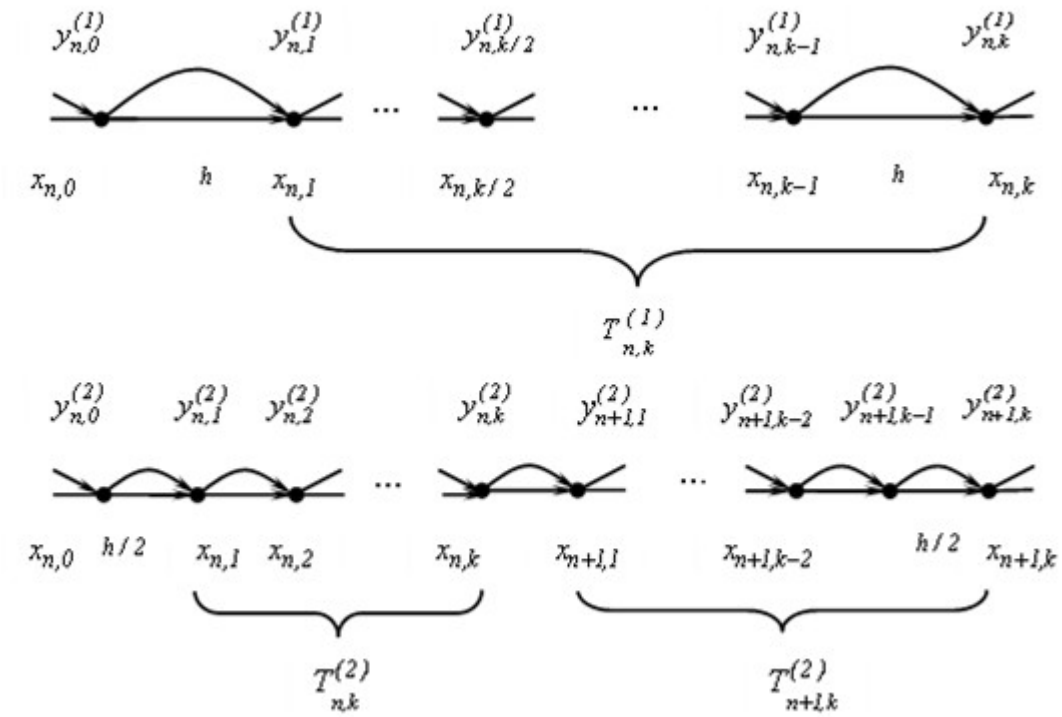


Рис. 3.2. Схема використання правила дублювання кроку для однокрокового блокового k-точкового паралельного методу

Точки n-го блоку сітки Ω_h складають множину $T_{n,k}^{(1)}$, а сітки $\Omega_{h/2}$ – $T_{n,k}^{(2)}$. Оскільки кількість точок у блоці для обох сіток дорівнює k, то для одного й того ж інтервалу інтегрування кількість блоків другої сітки точно у два рази більше, ніж для першої. Основою розрахунку при інтегруванні є сітка Ω_h, при цьому кожен вузол із парним номером у блоках сітки $\Omega_{h/2}$ використовується для обчислення оцінки локальної похибки на

цьому кроці. Більш того, як розв'язок у цих вузлах часто приймається апроксимація, отримана з половинним кроком або екстрапольована, як найбільш точна. Вузли сітки $\Omega_{h/2}$ із непарними номерами використовуються лише як допоміжні. Оскільки дані методи є неявними, вживання правила Рунге до блокових однокрокових методів вимагає розв'язання трьох різних систем нелінійних алгебраїчних рівнянь розміру k.

Загальний час послідовної реалізації блокових методів з правилом Рунге T_1^{11} складається з суми часу обчислення розв'язку з одинарним кроком у блоці n плюс час розв'язку з половинним кроком в n-тому і $(n+1)$-шому блоках. Оскільки для здобуття кожного з трьох розв'язків реалізується свій ітераційний процес, введемо наступні позначення. Нехай $L1$ – гранична кількість ітерацій для знаходження апроксимації розв'язку $y_{n,i}^{(1)}$, $L2$ – для $y_{n,i}^{(2)}$ и $L3$ – для $y_{n+1,i}^{(2)}$. Тоді, відповідно, поточну кількість ітерацій, що забезпечує достатню для кожної з даних задач точність, позначимо: $li,\ li \le Li,\ i = \overline{1,3}$.

Час обчислень для послідовного алгоритму блокових методів з правилом Рунге включає час на визначення нульових, а також подальших ітерацій вирішення:

$$T_1^{11} = \underbrace{6kt_{mul} + 3kt_{ad} + 2T_F}_{y_{n,i;0}^{(1)} + y_{n,i;0}^{(2)} + y_{n+1,i;0}^{(2)}} + (l1+l2+l3)\cdot \underbrace{[(k^2+2k)\cdot t_{mul} + (k^2+2k)\cdot t_{ad} + k\cdot T_F]}_{y_{n,i;l1}^{(1)} + y_{n,i;l2}^{(2)} + y_{n+1,i;l3}^{(2)}},$$

$$T_1^{11} = [k(l1+l2+l3)+2]\cdot T_F + [(l1+l2+l3)(2k^2+4k)+9k]\cdot t_{op} \qquad (3.6)$$

де T_F – час обчислень для правої частини ЗДР.

Обчислювальна схема паралельного блокового k-точкового методу з контролем локальної апостеріорної похибки за правилом Рунге приведена на рисунку 3.3. Тут кожен процесор обчислює розв'язок в одному вузлі сітки, тобто максимальний ступінь паралелізму обчислювальної схеми складає: $Dop = k$. Для кожної із трьох задач послідовно виконуються обчислення нульової і подальших ітерацій.

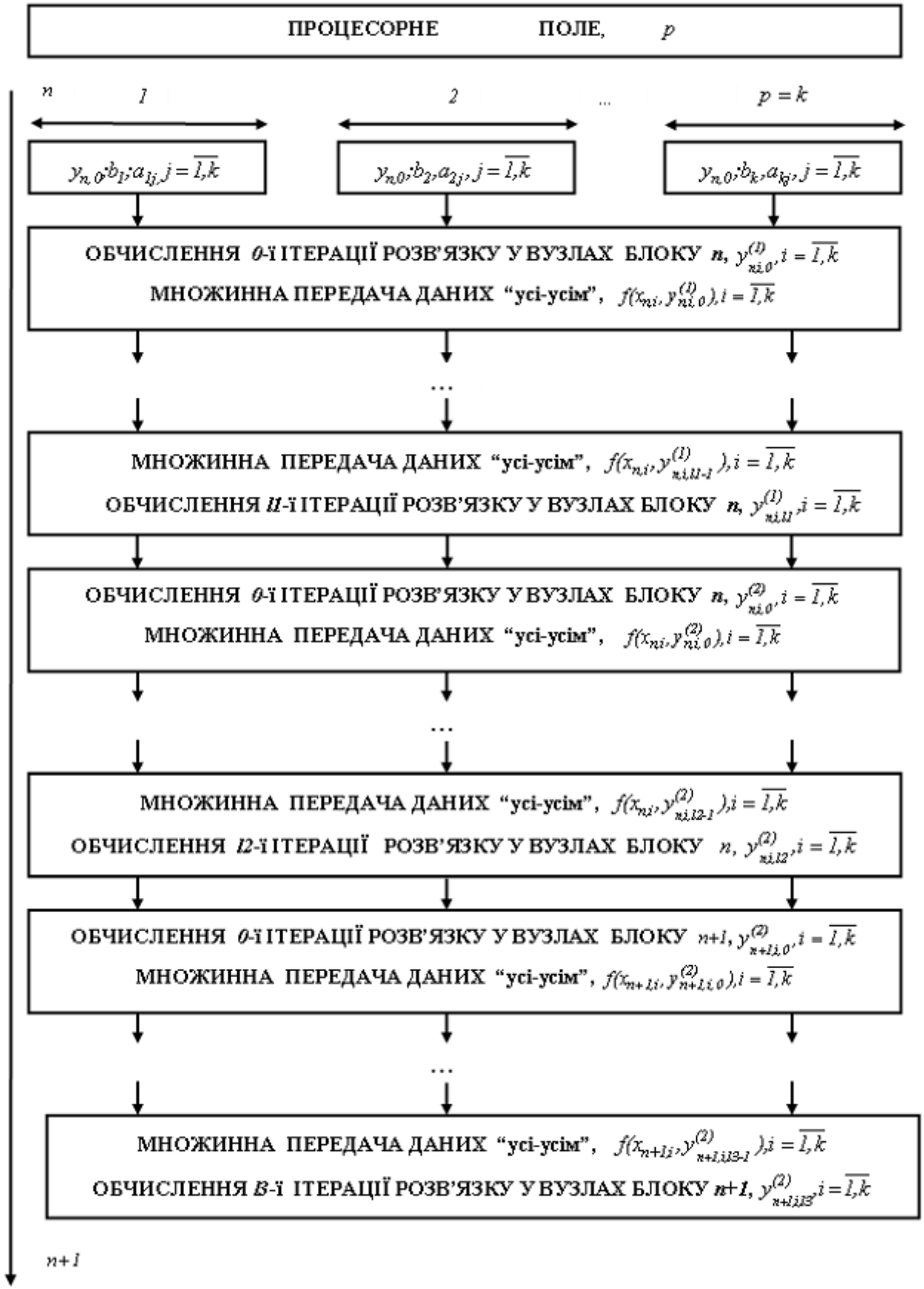


Рис. 3.3. Обчислювальна схема паралельного алгоритму блокового методу з контролем локальної похибки за правилом Рунге

При цьому нульова ітерація складається з наступних кроків: обчислення нульового наближення паралельно у кожному вузлі нового блоку за першою формулою (3.4); обчислення правої частини ЗДР від нульового наближення; множинний обмін обчисленими значеннями правої частини за типом “усі-усім”. Потім li разів виконується аналогічна група операцій для подальших ітерацій:

1) обчислення чергового наближення у кожному вузлі нового блоку за другою формулою (3.4), базовою операцією є множення матриці A на вектор значень правих частин ЗДР;

2) обчислення правої частини ЗДР від отриманого наближення і множинний обмін значеннями правої частини за типом “усі-усім”.

Таким чином, час паралельних обчислень за схемою (3.5) з локальною точністю $O(h^{k+li+2})$ у вузлах відповідних сіток складає:

$$T_{p,comp}^{11} = (\sum_{i=1}^{3} li + 2)\cdot T_F + [\sum_{i=1}^{3} li\cdot(k+3)+3]\cdot t_{mul} + [\sum_{i=1}^{3} li\cdot(k+2)+1]\cdot t_{ad}.$$

Для RISC архітектури, відповідно:

$$T_{p,comp}^{11} = (\sum_{i=1}^{3} li + 2)\cdot T_F + [\sum_{i=1}^{3} li\cdot(2k+5)+4]\cdot t_{op}. \qquad (3.7)$$

Для реалізації обмінів буде потрібно виконання групових операцій пересилок за типом “усі-усім”:

$$T_{p,comp}^{11} = \left(\sum_{i=1}^{3} li + 2\right)\cdot T_{all\text{-}to\text{-}all}(p). \qquad (3.8)$$

Потенційні характеристики паралелізму запропонованого методу можна оцінити по числу звернень до правої частини ОДУ. При $T_F \gg t_{op}$:

$$S_{pot}^{11} \approx \left[\left(k\sum_{i=1}^{3} li + 2\right)\cdot T_F\right] / \left[\left(\sum_{i=1}^{3} li + 2\right)\cdot T_F\right] \approx k,\ E_{pot}^{11} \approx \frac{S_{pot}^{11}}{p} \approx 1,$$

тобто має місце практично лінійне прискорення і одинична ефективність. Такі ж потенційні характеристики можуть бути отримані і у разі, коли права частина за часом обчислення сумірна з часом виконання однієї операції з рухомою точкою.

Реальні динамічні характеристики отриманого паралельного алгоритму істотно залежать не лише від параметрів задачі і алгоритму, але й від ефективності організації міжпроцесорних зв'язків та дорівнюють:

$$S^{11} = \frac{(k\cdot\sum_{i=1}^{3} li + 2)\cdot T_F + [\sum_{i=1}^{3} li\cdot(2k^2+4k)+5k]\cdot t_{op}}{(\sum_{i=1}^{3} li + 2)\cdot T_F + [\sum_{i=1}^{3} li\cdot(2k+5)+4]\cdot t_{op} + \left(\sum_{i=1}^{3} li + 2\right)\cdot T_{all\text{-}to\text{-}all}(p)}, \quad (3.9)$$

$$E^{11} = \frac{(k\cdot\sum_{i=1}^{3} li + 2)\cdot T_F + [\sum_{i=1}^{3} li\cdot(2k^2+4k)+5k]\cdot t_{op}}{k(\sum_{i=1}^{3} li + 2)\cdot T_F + [\sum_{i=1}^{3} li\cdot(2k+5)+4]\cdot t_{op} + \left(\sum_{i=1}^{3} li + 2\right)\cdot T_{all\text{-}to\text{-}all}(p)}. \quad (3.10)$$

Аналіз теоретичного виконання й обчислювальний експеримент показують, що для виконання групових операцій обміну в запропонованому алгоритмі ефективними є топології гіперкуб та тор (рис. 3.4), гірший варіант з'єднання процесорів – кільце.

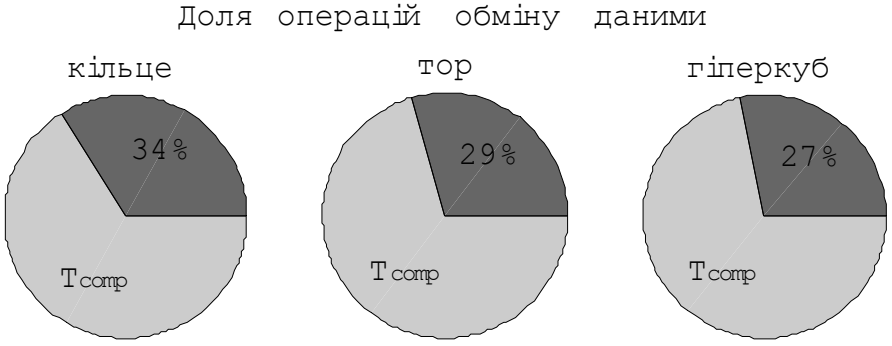
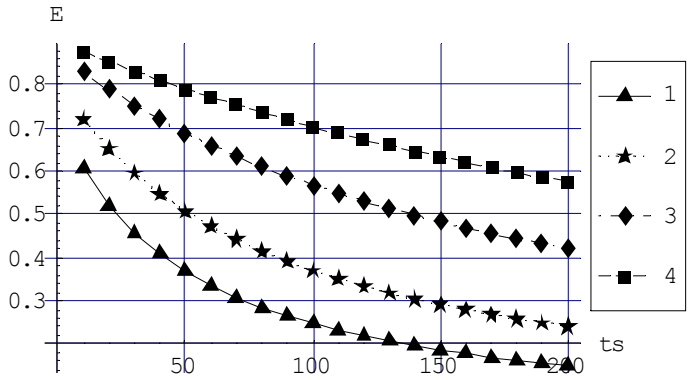


Рис. 3.4. Доля обмінів до спільного часу виконання блокового методу з правилом Рунге для різних топологій

Окрім топології з'єднання, на величину часу міжпроцесорних обмінів і динамічні характеристики паралелізму істотний вплив мають тип паралельної архітектури та визначені ним машинно-залежні константи обміну, такі, як латентність та час передачі одного слова (рис. 3.5). Відмітимо, що із зростанням величини латентності комунікаційного середовища час на реалізацію обмінів збільшується, а прискорення й ефективність алгоритму зменшуються $\uparrow t_s \Rightarrow \uparrow T_{p,comm} \Rightarrow \downarrow S \Rightarrow \downarrow E.$

Аналогічні залежності зв'язують динамічні характеристики та час передачі одного слова: $\uparrow t_w \Rightarrow \uparrow T_{p,comm} \Rightarrow \downarrow S \Rightarrow \downarrow E$. Проте величина латентності є найбільш істотним параметром, ступінь впливу швидкості передачі даних, як правило, зростає при збільшенні обсягів переданих даних.



1 - $T_F=100t_{op}$; 2 - $T_F=200t_{op}$, 3 - $T_F=500t_{op}$, 4 - $T_F=1000t_{op}$

Рис. 3.5. Залежність коефіцієнта ефективності блокового методу з правилом Рунге від значення латентності

Для даного алгоритму обчислення є однорідними, і, як результат, коефіцієнт ефективності для SIMD-архітектур за інших рівних умов вищий, ніж для MIMD-систем за рахунок меншого значення латентності мережі (рис. 3.6).

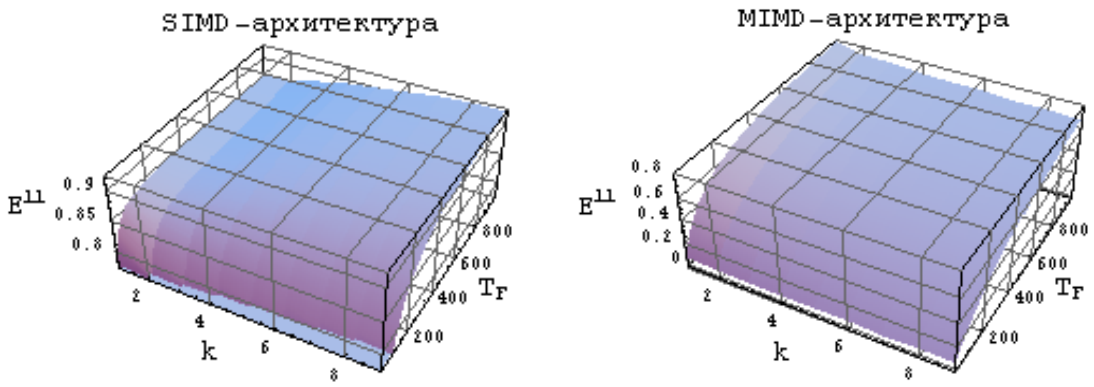
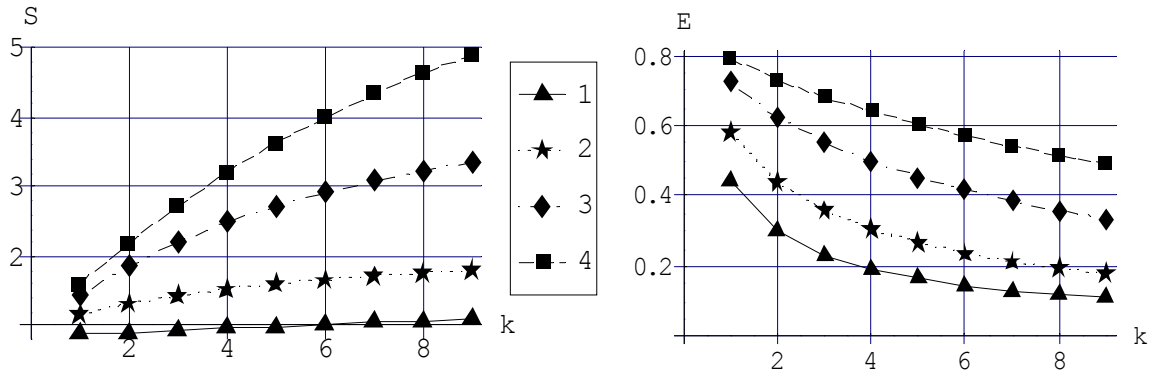


Рис. 3.6. Залежність коефіцієнтів ефективності блокових методів для ЗДР з правилом Рунге від кількості точок у блоці, k та складності правої частини, T_F

З тимчасових характеристик алгоритму та початкової задачі якість паралелізму найбільш істотно залежить від необхідного обсягу обчислень на реалізацію правої частини (3.1) й кількості точок в одному блоці.

Залежності реальних коефіцієнтів прискорення і ефективності паралельного процесу контролю локальної похибки на основі правила Рунге від кількості точок блоку при зростанні складності правих частин ЗДР представлені за допомогою рис. 3.7. Очевидно, чим складніша права частина ЗДР, тим кращі характеристики паралелізму: $\uparrow T_F \Rightarrow \uparrow S \Rightarrow \uparrow E$ і, одночасно, чим більше розмір блоку, співпадаючий з кількістю процесорів, тим більше прискорення та менша ефективність даного методу: $\uparrow k \Rightarrow \uparrow p \Rightarrow \uparrow S \Rightarrow \downarrow E$.



1 - $T_F=100t_{op}$; 2 - $T_F=200t_{op}$, 3 - $T_F=500t_{op}$, 4 - $T_F=1000t_{op}$

Рис. 3.7. Коефіцієнти прискорення та ефективності блокового методу з правилом Рунге , $p = k$

Таким чином, найкращі характеристики паралелізму при розв’язанні нелінійної задачі Коші для одного рівняння блоковими методами з контролем локальної похибки за правилом Рунге досягаються для будь-якої паралельної архітектури за умовами:

1) великого розміру задачі;

2) складної правої частини;

3) топології гіперкуб у високошвидкісних мережах передачі інформації.

3.1.2. Вкладені блокові паралельні однокрокові методи розв'язання задачі Коші

Ідея вкладених форм, що запропонована для оцінки локальної похибки чисельного розв'язання звичайних диференційних рівнянь методами Рунге-Кути, може бути використана і для однокрокових блокових багатоточкових методів. У даному підрозділі приведено обґрунтування цього способу оцінки похибки для паралельних вкладених блокових алгоритмів розв'язання нелінійної задачі Коші для ЗДР на основі двох різних підходів:

1) комбінації незалежних формул суміжних порядків точності розв'язку;

2) комбінації спеціально підібраних формул різних порядків на базі методу послідовного підвищення порядку точності.

Перший підхід полягає у використанні двох різних незалежних блокових методів суміжних порядків точності $r(\hat{r}), \hat{r} = r \pm 1$ на одній і тій же сітці інтегрування Ω_h:

$$\begin{cases} y_{n,i} = y_{n,0} + ih\left[b_i f(x_{n,0}; y_{n,0}) + \sum\limits_{j=1}^{k} a_{i,j} f(x_{n,j}; y_{n,j}) \right]; i = \overline{1,k}; n = \overline{1,N}, \\ \hat{y}_{n,i} = \hat{y}_{n,0} + ih\left[\hat{b}_i f(x_{n,0}; \hat{y}_{n,0}) + \sum\limits_{j=1}^{\hat{k}} \hat{a}_{i,j} f(x_{n,j}; \hat{y}_{n,j}) \right]; i = \overline{1,\hat{k}}; n = \overline{1,\hat{N}}. \end{cases} \quad (3.11)$$

При цьому перший метод визначає апроксимацію розв'язку на основі k-точкового однокрокового методу, а другий $\hat{k}$-точкового, також однокрокового, методу. Другий наближений розв'язок у співпадаючих вузлах блоків $T_{n,i}^{(k)}$ і $T_{n,j}^{(\hat{k})}$ сітки Ω_h використовується для оцінки апостеріорної локальної похибки (рис. 3.8). Нехай для визначеності основним є блоковий метод нижчого порядку точності, тобто $\hat{k} = k + 1$.

Локальна похибка наближеного розв'язку за однокроковим *k*-точковим методом у *i*-тому вузлі блоку визначається наступною формулою:

$$y(x_{n,i}) - y_{n,i}^{r} = O(h^{k+2}); i = \overline{1,k},$$

і для $(k+1)$-точкового методу у тому ж вузлі дорівнює:

$$y(x_{n,i}) - y^{\hat{r}}_{n,i} = O(h^{k+3}); i = \overline{1,\hat{k}}.$$

З отриманих співвідношень виходить, що оцінка локальної похибки формули меншого порядку точності k-точкового методу, приблизно може бути обчислена, як: $y^{r}_{n,i} - y^{\hat{r}}_{n,i}; i = \overline{1,k}$.

Такий підхід до оцінки локальної похибки є більш ефективним у порівнянні з правилом Рунге, оскільки при достатній простоті дозволяє зменшити обчислювальні витрати. Так, для вживання правила дублювання кроку необхідно вирішити три системи нелінійних алгебраїчних рівнянь розмірності k, а для застосування вкладеного методу дві нелінійні алгебраїчні системи: одну тієї ж розмірності, другу розмірності $(k+1)$.

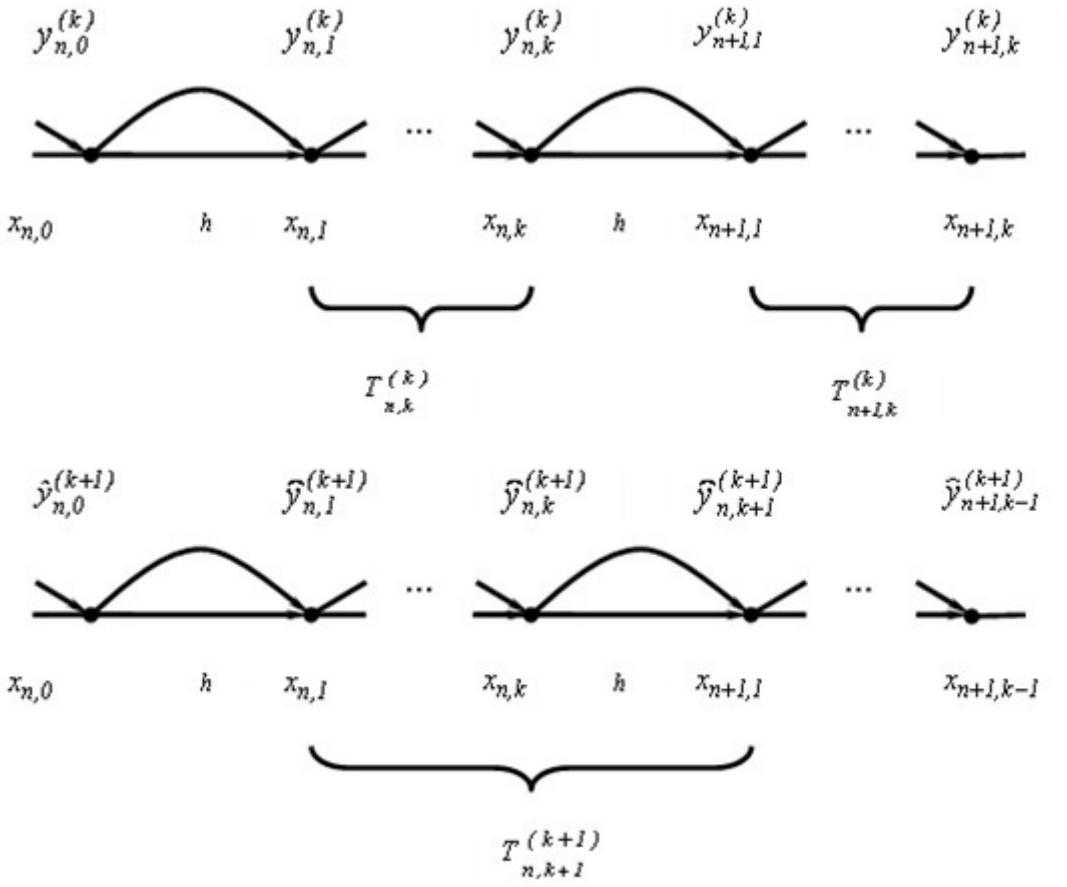


Рис. 3.8. Схема обчислення для вкладених однокрокових $k(\hat{k})$-точкових блокових методів, $\hat{k} = (k+1)$

Визначимо час послідовної реалізації даного методу в припущенні ідентичності обчислювальної та ємкісної потужності однопроцесорної та багатопроцесорної ОС. Нехай $T_1^{12}(k)$ - час послідовних обчислень за k-точковим методом, а $T_1^{12}(k+1)$ – аналогічний час, отриманий за $(k+1)$-точковим методом. Загальний час послідовного виконання схеми (3.12) дорівнює:

$$T_1^{12} = T_1^{12}(k) + T_1^{12}(k+1).$$

Кожен з блокових методів вимагає застосування ітераційного процесу для обчислення вирішення відповідних систем нелінійних рівнянь.

Нехай L – гранична кількість ітерацій для реалізації k-точкового методу, l – поточна кількість ітерацій, що забезпечує достатню для даної задачі точність $l \le L$. Аналогічно, для вкладеного методу вищого порядку маємо наступні позначення: $\hat{L}$ і $\hat{l}$, $\hat{l} \le \hat{L}$.

Тоді, $$T_1^{12}(k) = \underbrace{k[t_{mul} + t_{ad}] + T_F}_{y_{n,i,0}} + l \cdot \underbrace{[[k^2 + 2k] \cdot (t_{mul} + t_{ad}) + k \cdot T_F] + kt_{mul}}_{y_{n,i,l}},$$

$$T_1^{12}(\hat{k}) = \underbrace{\hat{k}[t_{mul} + t_{ad}] + T_F}_{\hat{y}_{n,i,0}} + \hat{l} \cdot \underbrace{\{[\hat{k}^2 + 2\hat{k}](t_{mul} + t_{ad}) + \hat{k}T_F] + \hat{k}t_{mul}}_{\hat{y}_{n,i,\hat{l}}} \quad (3.12)$$

При $t_{mul} = t_{ad} = t_{op}$ та $\hat{k} = k + 1$ час виконання послідовного алгоритму:

$$T_1^{12} = (lk + \hat{l}k + \hat{l} + 2) \cdot T_F + [2l \cdot [(k^2 + 2k) + 2\hat{l}(k^2 + 4k + 3) + 6k + 3] \cdot t_{op}, \quad (3.13)$$

а з урахуванням співвідношення $\hat{l} = l + 1$, маємо:

$$T_1^{12} = (2lk + k + l + 3) \cdot T_F + [(4l + 2) \cdot k^2 + 12lk + 6l + 14k + 9] \cdot t_{op}. \quad (3.14)$$

Обчислювальна схема паралельного алгоритму №1 на основі першого підходу, схема описується формулами (3.11), представлена на рис. 3.9. Оскільки $\hat{k} = k + 1$, то кількість процесорів складає: $p = \hat{k}$ (при реалізації k-точкового методу один процесор простоює). Кожен процесор обчислює розв'язок у одному вузлі сітки, тобто для такої обчислювальної схеми $Dop = \hat{k} = k + 1$.

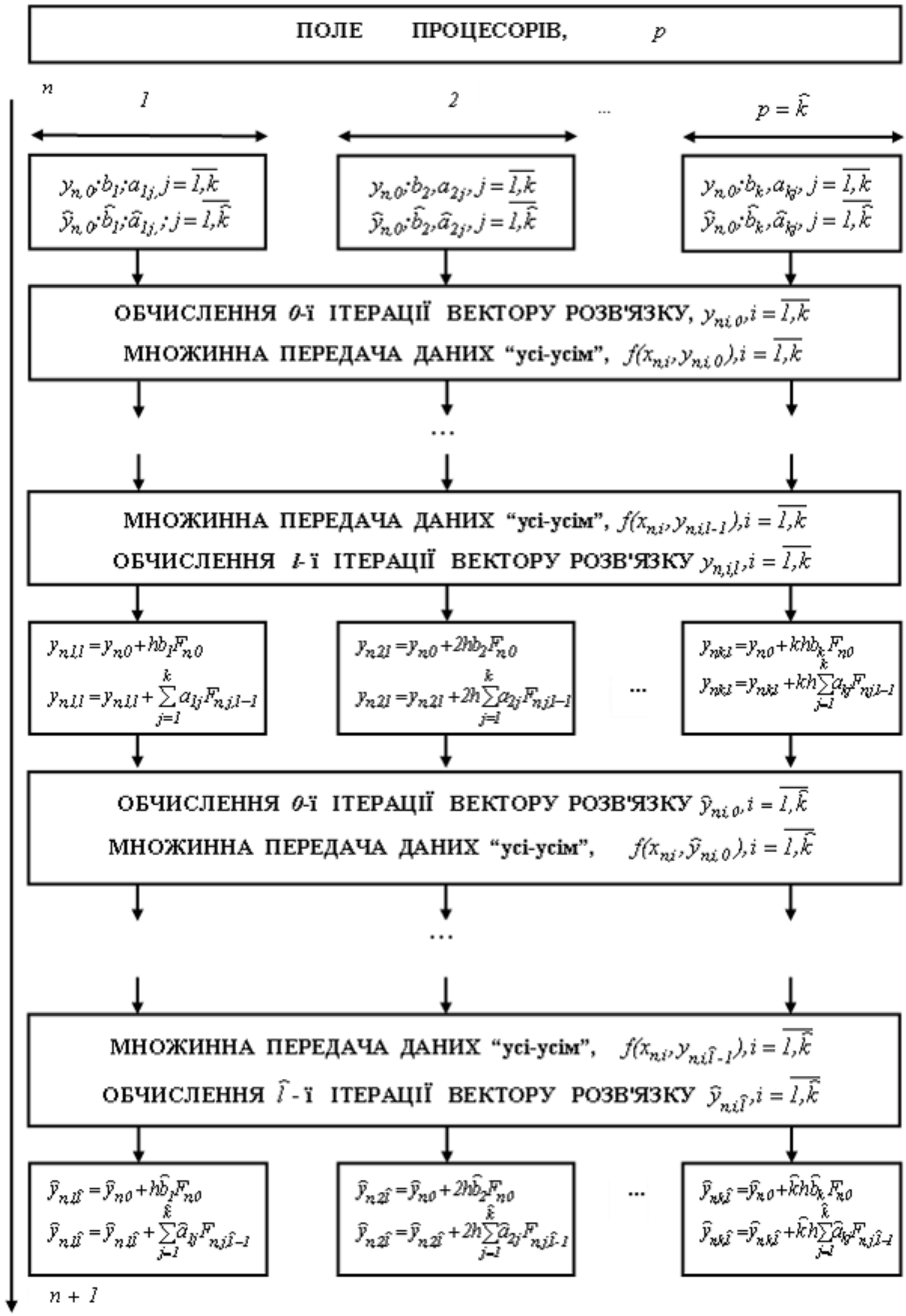


Рис. 3.9. Обчислювальна схема паралельного алгоритму №1 вкладеного однокрокового $k(\hat{k})$-точкового блокового методу для ЗДР

Спочатку визначається апроксимація розв'язку на основі k-точкового методу, потім – $(k+1)$-точкового. Для кожної з двох вирішуваних задач послідовно виконуються обчислення нульової і подальших ітерацій.

Тоді, час на обчислення й операції обміну паралельного вкладеного блокового методу №1 складає (при $\hat{l} = l+1$ та $t_{op} = t_{mul} = t_{ad}$):

$$T^{12}_{p,comp} = (2l+3)\cdot T_F + (4lk + 10l + 2k + 12)\cdot t_{op} \qquad (3.15)$$

$$T^{12}_{p,comm} = (l + \hat{l} + 2)\cdot T_{all\text{-}to\text{-}all}(p) = (2l+3)\cdot T_{all\text{-}to\text{-}all}(k+1). \qquad (3.16)$$

Потенційний паралелізм отриманого методу при складній правій частині ЗДР визначається співвідношенням:

$$S^{12}_{pot} = [k\cdot(l+\hat{l})+\hat{l}+2]\cdot T_F / [(l+\hat{l}+2)\cdot T_F] \approx k,\ E^{12}_{pot} = S^{12}_{pot}/p \approx 1.$$

Аналогічні потенційні динамічні характеристики маємо й при $T_F \approx t_{op}$, тобто при тривіальній правій частині ЗДР.

Для вкладених методів якість або ступінь впливу топології з'єднання процесорів й параметрів комунікаційного середовища на динамічні характеристики паралелізму співпадають із результатами, отриманими при аналізі блокових методів з правилом Рунге.

Другий підхід до розробки блокових вкладених методів оснований на використанні ідеї послідовного підвищення порядку точності [118-119], і має своєю метою скорочення обчислювальних витрат за рахунок комбінації спеціально підібраних формул суміжних порядків.

Нехай розв'язання задачі Коші для звичайного диференційного рівняння на деякому інтервалі інтегрування виконується на основі k-точкового однокрокового блокового методу.

Покажемо, що у якості оцінки локальної похибки у кожному вузлі поточного n-го блоку може бути прийнята наступна величина:

$$d_{n,i} = \hat{y}_{n,i} - y_{n,i} = y_{n,i,\hat{l}} - y_{n,i,\check{l}}, i = \overline{1,k}, \hat{l} = \check{l} \pm 1, \qquad (3.17)$$

де $y_{n,i,\hat{l}}$ і $y_{n,i,\check{l}}$ – $\hat{l}$ і $\check{l}$-та апроксимації, які отримані при вирішенні (3.3) ітераційним методом (3.4).

Введемо наступні позначення:

1) $y_{n,i}^{[v]}, i = \overline{1,k}$ – значення чисельного розв'язку в i-тому вузлі $x_{n,i}$ n-го блоку, що обчислене з локальною похибкою $O(h^v)$,

2) $F_{n,i}^{[v]} = f\left(x_{n,i}, y_{n,i}^{[v]}\right)$ – звернення до правої частини початкового диференційного рівняння, яке обчислене в точці $\left(x_{n,i}, y_{n,i}^{[v]}\right)$.

Нехай наближене значення вирішення $y_{n,0}$ в початковій точці n-го блоку обчислене з деякою локальною помилкою $O(h^v)$, що забезпечує достатню для поставленої задачі точність. Тоді, звернення до правої частини початкового диференційного рівняння може бути обчислене з такою ж похибкою: $F_{n,0}^{[v]}$. Виконавши обчислення для нульової ітерації за першою формулою ітераційного методу, отримаємо $y_{n,i,0}^{[2]} = y_{n,0}^{[v]} + ihF_{n,i,0}^{[v]}, i = \overline{1,k}$, оскільки локальна похибка формули Ейлера має порядок $O(h^2)$. Кожне подальше обчислення за другою формулою (3.4) дає підвищення ладу точності для методу простих ітерацій на 1:

$$l = 0: \; y_{n,i,1}^{[3]} = y_{n,0}^{[v]} + ih \cdot \left(b_i F_{n,0}^{[v]} + \sum_{j=1}^{k} a_{i,j} F_{n,j,0}^{[2]} \right), i = \overline{1,k},$$

$$l = 1: \; y_{n,i,2}^{[4]} = y_{n,0}^{[v]} + ih \cdot \left(b_i F_{n,0}^{[v]} + \sum_{j=1}^{k} a_{i,j} F_{n,j,1}^{[3]} \right), i = \overline{1,k},$$

$$\ldots,$$

оскільки $F_{n,j,0}^{[v]} = f(x_{n,j}, y_{n,j,0}^{[v]}), j = \overline{1,k}$.

Цей процес не може бути продовжений нескінченно, при $l = k - 1$, отримаємо результати, відповідні граничній локальній точності наближених формул (3.4). Оскільки різницеві схеми, відповідні блоковим однокроковим k-точковим методом, апроксимують диференційне рівняння (3.1) з порядком $O(h^{k+1})$, то подальші ітерації підвищення порядку точності результатів не дають:

$$l = k-1 : y_{n,i,k}^{[k+2]} = y_{n,0}^{[v]} + ih \cdot \left(b_i F_{n,0}^{[v]} + \sum_{j=1}^{k} a_{i,j} F_{n,j,k-1}^{[k+1]} \right), i = \overline{1,k}. \quad (3.18)$$

Таким чином, для оцінки локальної похибки можуть бути вибрані дві довільні суміжні апроксимації вирішення $y_{n,i,\hat{l}}$ і $y_{n,i,\check{l}}$ із урахуванням міркувань точності і обмежень:

$\check{l} < \hat{l} \le L-1 = k-1$.

Як правило, $\check{l} = l$, $\hat{l} = l+1$. При описаному вкладеному блоковому k-точковому методі, усі додаткові СНАР мають розмірність k.

Розрахункові формули для одного, n-го блоку вкладеного багатоточкового алгоритму №2 мають вигляд:

$$\begin{cases} y_{n,i,0} = y_{n,0} + ihF_{n,0}, i = \overline{1,k}, \\ y_{n,i,l+1} = y_{n,0} + ih(b_i F_{n,0} + \sum_{j=1}^{k} a_{i,j} F_{n,j,l}), \ l = \overline{0,\check{l}-1}, \\ \hat{y}_{n,i,\hat{l}} = y_{n,0} + ih(b_i F_{n,0} + \sum_{j=1}^{k} a_{i,j} F_{n,j,\check{l}}), \ \check{l} = l, \hat{l} = l+1. \end{cases} \quad (3.19)$$

Оцінка часу виконання послідовного алгоритму, що визначається обчислювальною схемою (3.19), дорівнює:

$$T_1^{13} = \underbrace{kt_{mul} + kt_{ad} + T_F}_{y_{n,i,0}} + \underbrace{\hat{l} \cdot [(k^2+2k) \cdot t_{mul} + (k^2+2k) \cdot t_{ad} + k \cdot T_f] + k \cdot t_{mul}}_{y_{n,i,\check{l}} + \hat{y}_{n,i,\hat{l}}}.$$

При $t_{mul} = t_{ad} = t_{op}$ маємо:

$$T_1^{13} = (\hat{l}k+1) \cdot T_F + [2\hat{l}(k^2+2k) + 3k] \cdot t_{op}. \quad (3.20)$$

Обчислювальна схема паралельного вкладеного блокового методу №2 приведена на рис. 3.10, як і раніше показано обчислення в рамках одного n-го блоку. Для простоти розглядується випадок, коли кількість процесорів збігається з розмірністю блоку і за кожним вузлом блоку закріплений один процесор.

Таким чином, для одного диференційного рівняння внутрішній паралелізм методу вичерпується незалежними обчисленнями кожної точки блоку й обмежений кількістю точок у блоці.

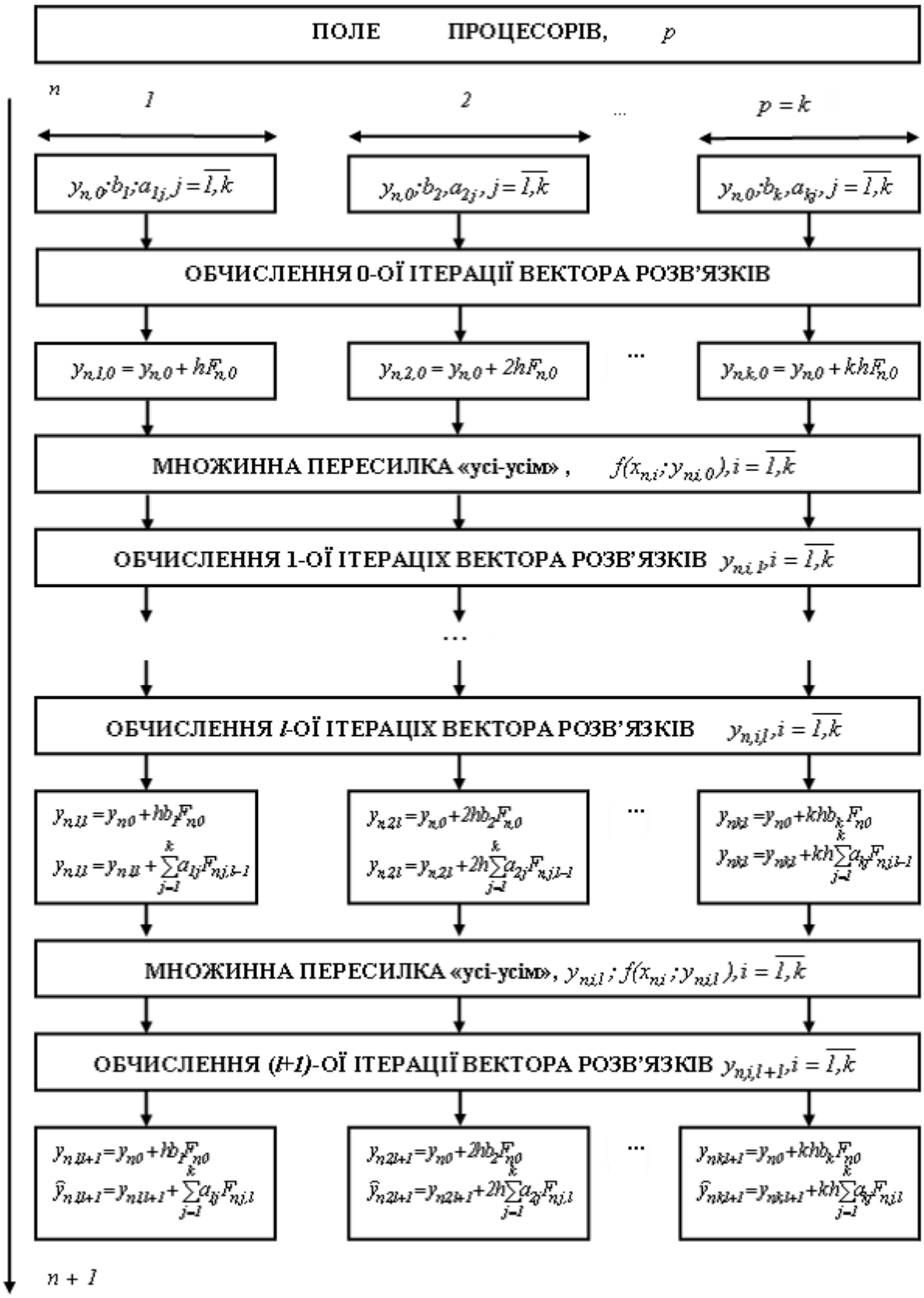


Рис. 3.10. Обчислювальна схема №3 паралельного алгоритму вкладеного однокрокового k-точкового блокового методу для ЗДР

Ітераційний процес розв'язання отриманих СНАР є суто послідовним. Після кожного з $l = \overline{0,\hat{l}}$ кроків необхідний обмін даними за типом "усі-усім". При реалізації алгоритму на k процесорах можна одночасно обчислювати значення $F_{n,i,l}$, а потім також одночасно набути по формулах значення $y_{n,i,l}$ для кожного фіксованого l. Час паралельного обчислення наближених значень розв'язку з точністю $O(h^{\hat{l}+2})$ для всіх вузлів блоку при виконанні $\hat{l}$ кроків ітераційного процесу дорівнює

$$T^{13}_{p,comp} = (\hat{l}+1)\cdot T_F + [\hat{l}(k+2)+2]\cdot t_{mul} + [\hat{l}(k+2)+1]\cdot t_{ad},$$
$$T^{13}_{p,comp} = (\hat{l}+1)\cdot T_F + [2\hat{l}k + 4\hat{l} + 3]\cdot t_{op}. \qquad (3.21)$$

Час обміну даними при цьому з використанням уведених комунікаційних примітивів буде рівний:

$$T^{13}_{p,comm} = (\hat{l}+1)\cdot T_{all\text{-}to\text{-}all}(1,p) = (k+1)\cdot T_{all\text{-}to\text{-}all}(1,p). \qquad (3.22)$$

Потенційно обидва вкладені блокові методи володіють високим ступенем внутрішнього паралелізму: практично лінійним прискоренням і одиничною ефективністю. Це витікає з приведених нижче співвідношень для коефіцієнтів потенційного прискорення і ефективності алгоритму у випадку, якщо час звернення до правої частини ЗДР домінує над іншими обчисленнями:

$$S^{13}_{pot} = T^{13}_1 / T^{13}_p \approx [(\hat{l}k+1)T_F]/[(\hat{l}+1)T_F] \approx k = p,\ E^{13}_{pot} = S^{13}_{pot}/p \approx 1.$$

Аналогічний результат маємо для тривіальної правої частини. Як і раніше найкращою топологією для розроблених методів є топологія гіперкуб. В цілому вплив параметрів комунікаційної середи на динамічні характеристики паралелізму цих двох алгоритмів збігається.

Проведемо порівняння двох різних вкладених блокових методів (рис. 3.11-3.13) з метою визначити мінімум накладних витрат при оцінці локальної похибки розв'язання на основі ідеї вкладеності.

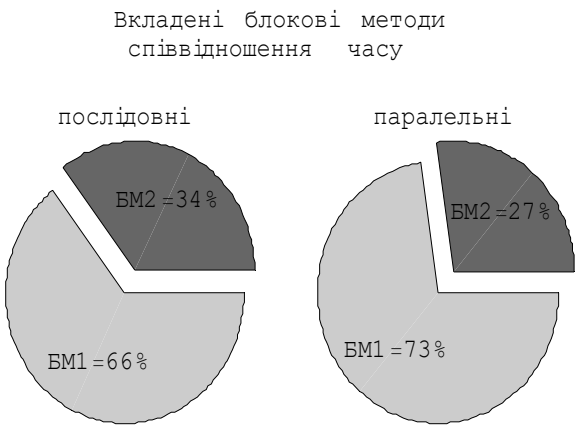


Рис. 3.11. Порівняння тимчасової складності двох вкладених блокових методів для ЗДР в послідовній та паралельній реалізаціях

Оскільки швидкість збіжності ітераційного процесу при вирішенні систем нелінійних алгебраїчних рівнянь істотно залежить від властивостей конкретної системи, а, отже, коефіцієнтів самого блокового методу, для спільності міркувань проведемо порівняння при гранично допустимій кількості ітерацій.

Аналіз теоретичного виконання й проведений експеримент дозволяють зробити наступні висновки:

1) час виконання першого вкладеного блокового алгоритму більше часу виконання другого, як у послідовній, так і в паралельній реалізаціях: $T_1^{12} > T_1^{13}$ і $T_p^{12} > T_p^{13}$, причому для паралельних алгоритмів ця різниця більша, ніж для послідовних;

2) коефіцієнти прискорення і ефективності першого вкладеного блокового методу менше відповідних коефіцієнтів другого метода при різних значеннях параметрів завдання, метода і паралельної системи: $S^{12} < S^{13}$, $E^{12} < E^{13}$ (рис. 3.12-3.13).

Відмітимо, що вплив чинника “складність правої частини ЗДР” є таким, що багато в чому визначає якість паралелізму. При однакових значеннях комунікаційних констант і параметрів, які задають унікальний блоковий метод, коефіцієнти прискорення і ефективності зменшуються практично у два рази при переході від домінуючих до тривіальних правих

частин. Ступінь впливу комунікаційних констант традиційний для даних методів і операцій обміну. Збільшення кількості точок у блоці призводить до зростання коефіцієнта прискорення $\uparrow k \Rightarrow \uparrow S$ та к зменшенню коефіцієнта ефективності $\uparrow k \Rightarrow \downarrow E$.

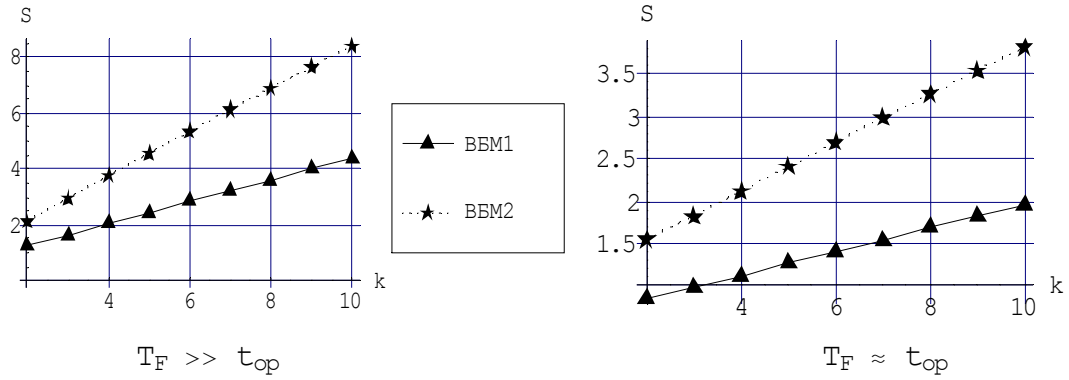


Рис. 3.12. Залежність коефіцієнта прискорення вкладених блокових методів від кількості точок блоку та складності правої частини ЗДР

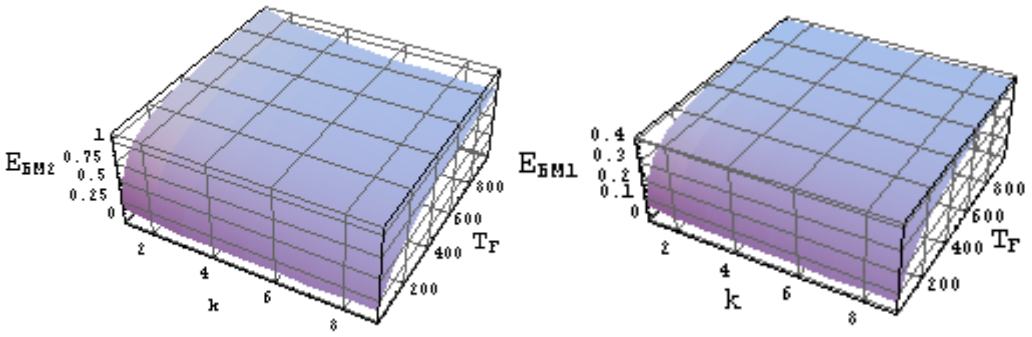


Рис. 3.13. Коефіцієнт ефективності вкладених блокових методів від числа точок блоку і складності правої частини ЗДР

Така залежність пояснюється тим, що число точок блоку пов'язане із кількістю використовуваних процесорів. Підсумовуючи всі отримані результати, можна зробити висновок, що з двох методів вкладених форм для блокових однокрокових способів розв’язання нелінійної задачі Коші для ЗДР, другий метод володіє безперечними перевагами.

3.1.3. Технологія локальної екстраполяції на основі однокрокових блокових методів для ЗДР

Реалізация технології локальної екстраполяції Річардсона для блокових методів вимагає багатократних обчислень на одному й тому ж інтервалі інтегрування з використанням опорного блокового k_0-точкового методу порядку r_0 на рівномірних сітках, що згущуються:

а) $\Omega_{h/n_1} = \{x_j\}, j = \overline{1,M_1}$ з кроком $h_1 = h/n_1$ в N_1 блоках;

б) $\Omega_{h/n_2} = \{x_j\}, j = \overline{1,M_2}$ з кроком $h_2 = h/n_2$ в N_2 блоках;

.....

в) $\Omega_{h/n_k} = \{x_j\}, j = \overline{1,M_k}$ з кроком $h_k = h/n_k$ в N_k блоках, де k – кількість рядків екстраполяційної таблиці.

Базовий крок інтегрування дорівнює $h_1 = h$, $n_1 = 1$, тобто основою рахунку є сітка: Ω_h. Для генерації допоміжних сіток вибирається гармонійний ряд $P_1 = \{n_2,...,n_k,...\} = \{2,3,4,5,...\}$, як найменш витратний у разі опорного методу довільного типу (див. 2.20). Блоковий опорний метод повинен мати малий порядок точності:

$$T_{11,i} = y_{n,i} = y_{n,0} + ih\left[b_i F_{n,0} + \sum_{j=1}^{k_0} a_{i,j} F_{n,j}\right]; i = \overline{1,k_0}, k_0 \le 4, \qquad (3.23)$$

оскільки із зростанням r_0 обчислювальні витрати на технологію у цілому істотно зростають, не дивлячись на лінійне зменшення довжини екстраполяційної таблиці. Для неявних методів, до яких відносяться і дані блокові методи, це особливо важливо, оскільки збільшення порядку, а, отже, і кількості точок блоку, спричиняє за собою збільшення порядку багатократно вирішуваних СНАР.

Схема розбиття базового інтервалу на блоки при вживанні технології локальної екстраполяції для двоточкового опорного методу і гармонійної послідовності приведена на рис. 3.14. Відмітимо, що для блокових методів базовий інтервал інтегрування – це деякий блок із номером n основної сітки

довжини $H = k_0 h$. Загальне число точок, що обчислюються з різними кроками інтегрування $h_i = h/i, i = \overline{1,k}$ і визначають вузли сіток Ω_{h_i} по P_1, дорівнює $M_i = H/h_i = (k_0 h)/h_i = k_0 i.$ Відповідно, кількість блоків i – тої сітки $N_i = M_i / k_o = i$ і $n_i = N_i, i = \overline{1,k}$.

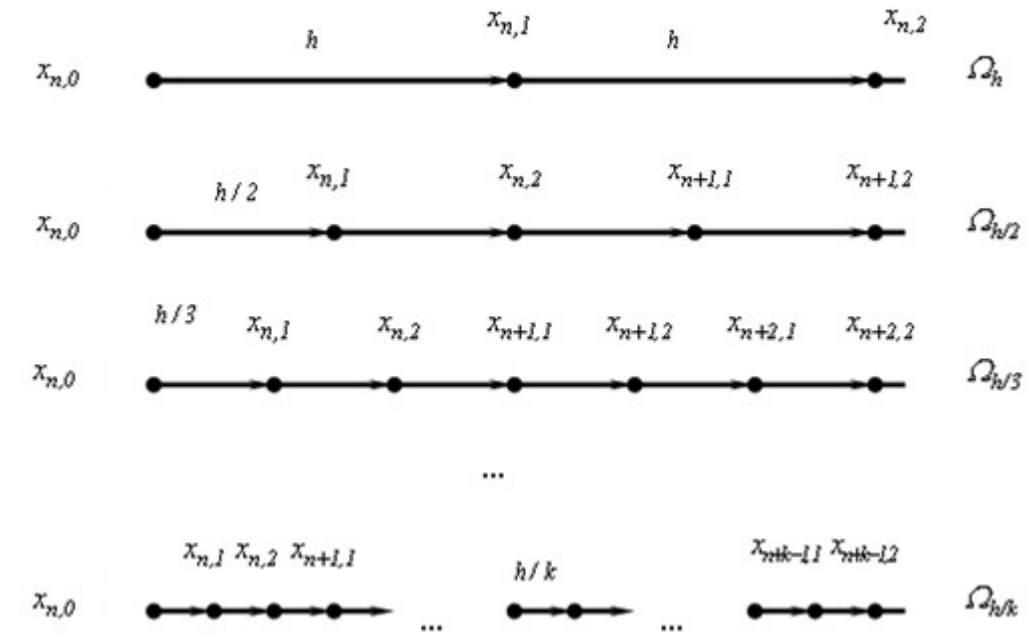


Рис. 3.14. Схема розбиття на блоки за технологією локальної екстраполяції для однокрокових блокових методів

Оскільки екстраполюються значення у вузлах базової сітки, необхідно встановити механізм відповідності вузлів сітки Ω_h основного рахунку вузлам сіток для екстраполяції $\Omega_{h_i}, i = \overline{2,k}$. Для двоточкового опорного блокового методу і гармонійного ряду маємо:

1) i - парне число: $x_{n,1}(h) \Leftrightarrow x_{n+\frac{i}{2}-1,2}(h_i);$

2) i - непарне число: $x_{n,1}(h) \Leftrightarrow x_{n+\left\lceil \frac{i}{2} \right\rceil,1}(h_i);$

3) для будь-яких i : $x_{n,2}(h) \Leftrightarrow x_{n+i-1,2}(h_i)$.

Значення першого стовпця екстраполяційної таблиці визначаються на основі наступних формул:

$$\begin{cases} T_{11,n,i} = y_{n,i}^{(1)} = y_{n,0}^{(1)} + ih \cdot \left[b_i F_{n,0} + \sum\limits_{j=1}^{k_0} a_{i,j} F_{n,j} \right]; i = \overline{1,k_0}; n = \overline{1,N_1}, \\ \ldots \\ T_{k1,n,i} = y_{n,i}^{(k)} = y_{n,0}^{(k)} + i\frac{h}{k} \cdot \left[b_i F_{n,0} + \sum\limits_{j=1}^{k_0} a_{i,j} F_{n,j} \right]; i = \overline{1,k_0}; n = \overline{1,N_k}. \end{cases} \quad (3.24)$$

Кожна з апроксимацій виходить за рахунок N_i разів застосованої схеми блокового k_0-точкового методу з різними кроками інтегрування:

$$\begin{cases} T_{T_{11}} = T_{\bar{y}_{n,i}^{(1)}(h)} = N_1 \cdot T_1^{r_0}(h), \\ \qquad\qquad \ldots \\ T_{T_{k1}} = T_{\bar{y}_{n,i}^{(k)}\left(\frac{h}{k}\right)} = N_k \cdot T_1^{r_0}(h/k), \end{cases} \quad (3.25)$$

де $T_1^{r_0}(h/i), i = \overline{1,k}$ – час, необхідний для вирішення задачі Коші для ЗДР опорним методом порядку r_0 з кроком h_i. Потім за формулою Ейткена-Невілла обчислюються наближення $T_{22}, T_{33,\ldots}$ і $T_{k,k}$. Час послідовних обчислень за схемою локальної екстраполяції складається з часу визначення k апроксимацій вирішення з різними кроками інтегрування і часу обчислення елементів екстраполяційної таблиці:

$$T_1^{14} = T_{T_{11}} + T_{T_{21}} + \ldots + T_{T_{k1}} + T_1^{ext-tab}, \quad T_1^{14} = T_1^{r0} \cdot \sum_{i=1}^{k} n_i + T_1^{ext-tab}.$$

Для блокових опорних методів порядку $r_0 = k_0 + 1$ та P_1 маємо:

$$N_{P_1}(k) = \sum_{i=1}^{k} n_i = (k^2 + k)/2; \; k = r - r_0 + 1 \Rightarrow$$

$$\Rightarrow \quad N_{P_1(k)}^{r_0} = [r^2 - r(2r_0 - 3) + (r_0 - 2)(r_0 - 1)]/2 \Rightarrow \quad (3.26)$$

$$N_{P_1(k)}^{r_0=3} = (r^2 - 3r + 2)/2.$$

Час обчислення елементів екстраполяційної таблиці дорівнює

$$T_1^{ext-tab} = \frac{(k^2 - k)}{2} \cdot (2t_{mul} + 3t_{ad}) = 5 \cdot \frac{r^2 - r(2r_0 - 1) + r_0(r_0 - 1)}{2} \cdot t_{op} \Rightarrow \quad (3.27)$$

при $r_0 = 3$: $T_1^{ext-tab} = 2.5 \cdot (r^2 - 5r + 6) \cdot t_{op}$.

Нехай L_0 – гранична, максимальна кількість ітерацій, необхідна для розв'язання СНАР (3.24) ітераційним методом. Тоді час виконання послідовного алгоритму вирішення ЗДР на базі блокових однокрокових методів з технологією локальної екстраполяції складає:

$$T_1^{14} = T_1^{r0} \cdot \left(\frac{r^2 - 3r + 2}{2} \right) + 5\left(\frac{r^2 - 5r + 6}{2} \right) \cdot t_{op}, \tag{3.28}$$

де $T_1^{r0} = (L_0 k_0 + 1) \cdot T_F + [2L_0(k_0^2 + 2k_0) + 3k_0] \cdot t_{op}$,

при $L_0 = k_0$: $T_1^{r0} = (k_0^2 + 1) \cdot T_F + [2k_0^3 + 4k_0^2 + 3k_0] \cdot t_{op}$,

$$T_1^{14} = (k_0^2 + 1)\left(\frac{r^2 - 3r + 2}{2} \right) \cdot T_F +$$

$$+ \left[(2k_0^3 + 4k_0^2 + 3k_0)\left(\frac{r^2 - 3r + 2}{2} \right) + 5\left(\frac{r^2 - 5r + 6}{2} \right) \right] \cdot t_{op}. \tag{3.29}$$

Тоді, при $k_0 = 2 \Rightarrow L_0 = 2 \Rightarrow r_0 = 3$ маємо: $T_1^{r_0=3} = 5T_F + 38t_{op}$.

Як результат: $T_1^{14} = (5T_F + 38t_{op})\left(\frac{r^2 - 3r + 2}{2} \right) + 5\left(\frac{r^2 - 5r + 6}{2} \right) \cdot t_{op}$,

$$T_1^{14} = 5 \cdot \left(\frac{r^2 - 3r + 2}{2} \right) \cdot T_F + \left(\frac{43r^2 - 89r + 106}{2} \right) \cdot t_{op}. \tag{3.30}$$

Для побудови паралельного алгоритму локальної екстраполяції на основі однокрокового блокового неявного методу (рис. 3.15) скористаємося результатами другої глави. Реалізуємо першу схему макрооперації, граф впливу якої приведений на рис. 2.20. Як основна макрооперація вибирається однократне обчислення деякої апроксимації розв'язку на основі k_0-точкового опорного методу малого порядку. Кожна макрооперація є ітераційним процесом, виконаним максимально можливим числом разів, тобто L_0. Потім формується остання частина екстраполяційної таблиці. Визначимо оцінки часу виконання паралельного алгоритму:

$$T_p^{14} = T_p^{r0} \cdot \sum_{i=1}^{k} n_i + T_p^{ext-tab}.$$

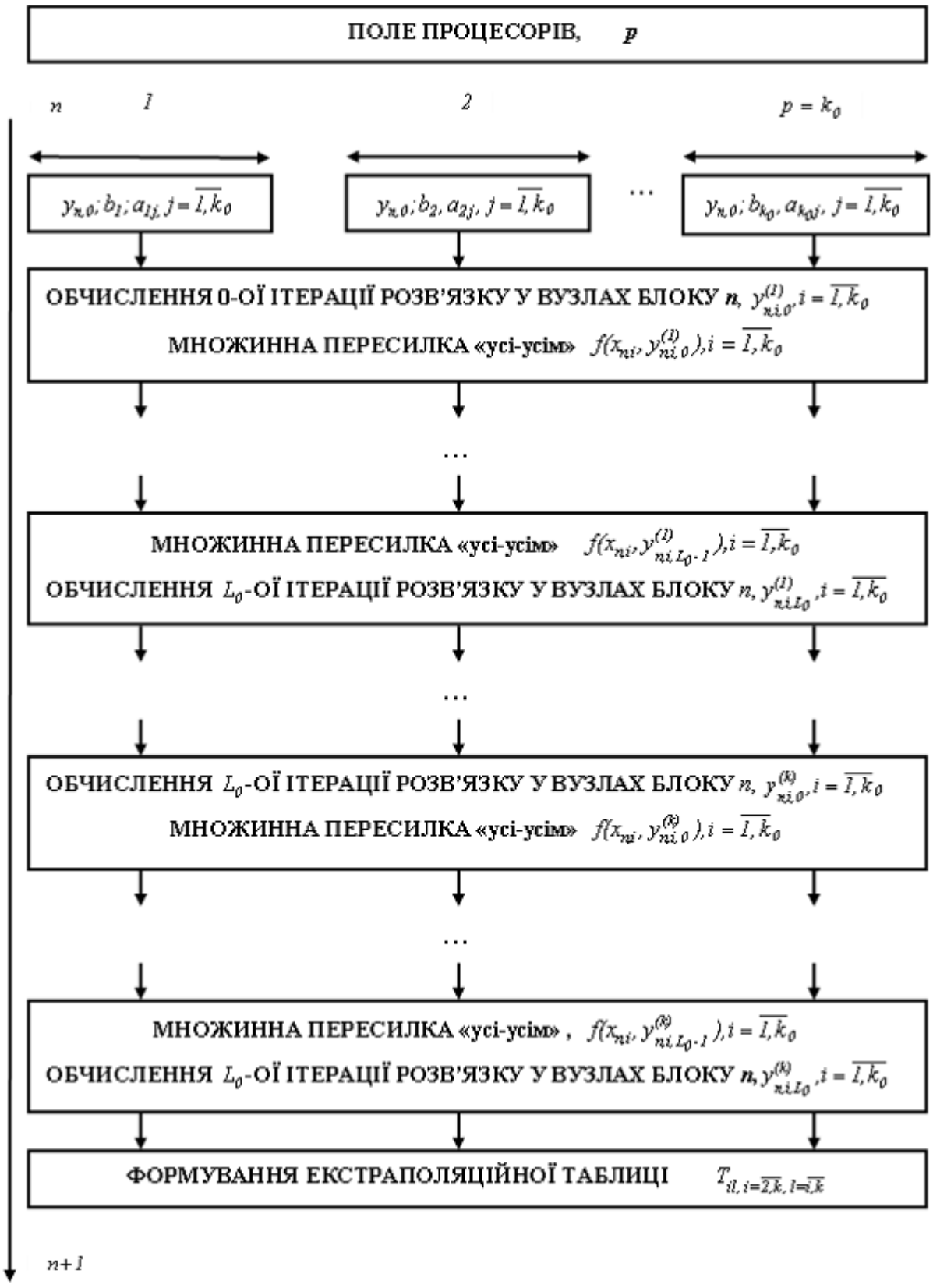


Рис. 3.15. Обчислювальна схема паралельного алгоритму технології локальної екстраполяції на основі однокрокового k_0-точкового блокового методу для ЗДР

Реалізація паралельних обчислень для опорного блокового методу потребує наступного часу обчислювальних і обмінних операцій:

$$T_{p,comp}^{r_0} = (L_0 + 1)\cdot T_F + [2L_0(k_0 + 2) + 3]\cdot t_{op}, \tag{3.31}$$

$$T_{p,comm}^{r_0} = (L_0 + 1)\cdot T_{all-to-all}(1, p). \tag{3.32}$$

Час паралельного виконання технології Річардсона складе:

$$T_p^{r_0}\cdot\sum_{i=1}^{k} n_i = \left(\frac{r^2 - r(2r_0 - 3) + (r_0 - 2)(r_0 - 1)}{2}\right)\cdot\left((L_0 + 1)\cdot T_F + [2L_0(k_0 + 2) + 3]\right)\cdot t_{op},$$

$$T_p^{ext-tab} = 5\cdot\frac{r^2 - r(2r_0 - 1) + r_0(r_0 - 1)}{2}\cdot t_{op}.$$

Для двоточкового опорного методу:

$$T_{p,comp}^{14} = T_{p,comp}^{r_0=3}\cdot\sum_{i=1}^{k} n_i + T_{p,comp}^{ext-tab} = (3T_F + 19t_{op})\cdot\left(\frac{r^2 - 3r + 2}{2}\right) + T_{p,comp}^{ext-tab}. \tag{3.33}$$

Відповідно, коефіцієнт потенційного прискорення алгоритму у випадку, якщо час звернення до правої частини ЗДР домінує над іншими обчисленнями, визначається таким чином:

$$S_{pot}^{14} = T_1^{14} / T_p^{14} \approx [(L_0k_0 + 1)\cdot T_F] / [(L_0 + 1)\cdot T_F] \approx k_0 = p,\ E_{pot}^{14} = S_{pot}^{14} / p \approx 1.$$

Аналогічний результат маємо для тривіальної правої частини, тобто потенційно метод локальної екстраполяції для багатоточкових блокових методів володіє високим ступенем внутрішнього паралелізму. Обмінні операції розробленого паралельного алгоритму, потребують:

$$T_{p,comm}^{14} = \left(\frac{r^2 - r(2r_0 - 3) + (r_0 - 2)(r_0 - 1)}{2}\right)(L_0 + 1)\cdot T_{all-to-all}(p),$$

$$T_{p,comm}^{14} = \left(\frac{r^2 - 3r + 2}{2}\right)\cdot(L_0 + 1)\cdot T_{all-to-all}(p). \tag{3.34}$$

Аналіз комунікаційної складової дає підстави стверджувати, що найкращою для запропонованого методу, як і раніше є топологія гіперкуб. Для порівняння паралельних алгоритмів різних засобів оцінки локальної апостеріорної похибки на основі багатоточкових методів оцінимо їх динамічні характеристики (рис. 3.16-3.17). Встановлено, що найбільш

простими і найменш витратними є методи вкладених форм для систем будь-якої архітектури, різних параметрів задачі і методу. Це має особливе значення, оскільки, не дивлячись на трудомісткість, теоретично немає жодних обмежень для здобуття блокових методів вищих порядків.

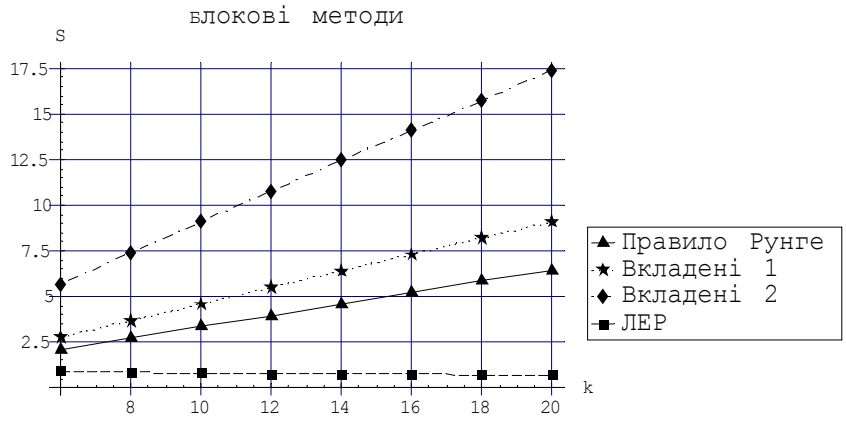


Рис. 3.16. Коефіцієнт прискорення блокових методів із різними засобами оцінки локальної похибки, $p = k$

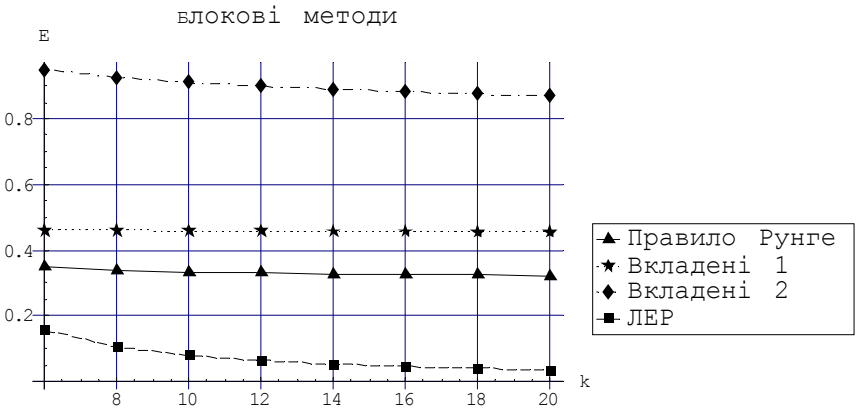


Рис. 3.17. Коефіцієнт ефективності блокових методів із різними засобами оцінки локальної похибки, $p = k$

Проте, вкладений блоковий метод на основі комбінації незалежних формул, як показали дослідження, має практично такий порядок складності, що і метод з використанням правила Рунге. Паралельний алгоритм технології локальної екстраполяції має гірші характеристики паралелізму, сферою його застосування, як і раніше залишаються високоточні розв’язки.

3.1.4. Особливості інтегрування систем звичайних диференційних рівнянь на базі вкладених блокових методів

Розроблені і обґрунтовані в попередніх підрозділах однокрокові блокові методи розв'язання задачі Коші для звичайного диференційного рівняння можна перенести на випадок системи рівнянь. При переході від одного рівняння до системи з'являється можливість використовувати системний паралелізм, який, як правило, значно більше паралелізму методу.

Блоковий однокроковий k-точковий метод для СЗДР має вигляд:

$$\bar{y}_{n,i} = \bar{y}_{n,0} + ih\left[b_i F_{n,0} + \sum_{j=1}^{k} a_{i,j} F_{n,j} \right]; i = \overline{1,k}; n = \overline{1,N}, \tag{3.35}$$

і вимагає вирішення системи алгебраїчних нелінійних рівнянь розміру $m \times k$:

$$\begin{cases} y_{n,q,i;0} = y_{n,q,0} + ihf_q(x_{n,0}; \bar{y}_{n,0}), \quad i = \overline{1,k}, \ n = \overline{1,N}, \ q = \overline{1,m}, \\ y_{n,q,i;l+1} = y_{n,q,0} + ih[b_i f_q(x_{n,0}; \bar{y}_{n,0}) + \sum_{j=1}^{k} a_{i,j} \cdot f_q(x_{n,j}; \bar{y}_{n,j,l})], \quad l = \overline{0,L-1}, \end{cases} \tag{3.36}$$

де $F_{n,q,j} = f_q[x_{n,j}; y_1(x_{n,j}), y_2(x_{n,j}), ..., y_m(x_{n,j})], \ q = \overline{1,m}$ є q-та компонента вектора правої частини СЗДР.

Особливості паралельного інтегрування систем ЗДР розгледимо на основі вкладених блокових методів:

$$\begin{cases} y_{n,q,i;0} = y_{n,q,0} + ihF_{n,q,0}, \ i = \overline{1,k}, \\ y_{n,q,i} = y_{n,q,i;l+1} = y_{n,q,0} + ih(b_i F_{n,q,0} + \sum_{j=1}^{k} a_{i,j} F_{n,q,j;l}), \quad l = \overline{0,L-2}, \\ \hat{y}_{n,q,i} = y_{n,q,i;L} = y_{n,q,0} + ih(b_i F_{n,q,0} + \sum_{j=1}^{k} a_{i,j} F_{n,q,j;L-1}). \end{cases} \tag{3.37}$$

Послідовний алгоритм інтегрування за чисельною схемою (3.37) має наступну обчислювальну складність:

$$T_1 = (Lk+1)\sum_{i=1}^{m} T_{f_i} + m[2L(k^2+2k)+3k]t_{op}. \tag{3.38}$$

Ітераційний процес, що описується обчислювальною схемою (3.37), може бути реалізований лише послідовно. Проте він дозволяє розподілити

обчислення поточної l-тої ітерації векторів розв'язку наступними способами:

1) на k незалежних процесів за кількістю точок у блоці ($Dop_1 = k$);

2) на m за кількістю компонент у системі ЗДР($Dop_2 = m$);

3) на mk процесів за максимальною можливою шириною паралельного методу і системи ($Dop_3 = mk$).

Обчислювальні схеми паралельного вкладеного блокового методу для СЗДР приведені на рисунках 3.18-3.20. Для скорочення запису введені наступні позначення. Вектор розв'язків $y_{n,q,i;l}$ має чотири індекси, n – номер блоку, q – номер компоненти початкової СЗДР, i – номер точки у блоці, l – номер ітерації при розв'язанні відповідної СНАР. Наявність прочерку для одного з індексів означатиме, що береться відповідний вектор значень. Так, $y_{n,\_,i;l}$ – позначає вектор розв'язків розміру m для СЗДР у i-й точці n-го блоку, обчислений на l-тій ітерації: $(y_{n,1,i;l}, y_{n,2,i;l}, \ldots, y_{n,m,i;l})$, а $y_{n,q,\_;l}$ – вектор розв'язку розмірності k для q-тої компоненти СЗДР, обчислений в усіх точках n-го блоку на l-тій ітерації: $(y_{n,q,1;l}, y_{n,q,2;l}, \ldots, y_{n,q,k;l})$.

Оцінимо якість отриманих паралельних алгоритмів і визначимо пріоритетні області використання кожного з них:

$$1)\ T^1_{p,comp} = (L+1)\cdot \sum_{i=1}^{m} T_{f_i} + m[2Lk + 4L + 3]\cdot t_{op}, \tag{3.40}$$

$$T^1_{p,comm} = (L+1)\cdot T_{all\text{-}to\text{-}all}(m,p) = (L+1)\cdot T_{all\text{-}to\text{-}all}(m,k), \tag{3.41}$$

$$2)\ T^2_{p,comp} = (Lk+1)T_{F_{max}} + [2L(k^2+2k) + 3k]t_{op}, \tag{3.42}$$

$$T^2_{p,comm} = (L+1)\cdot T_{all\text{-}to\text{-}all}(k,p) = (L+1)\cdot T_{all\text{-}to\text{-}all}(k,m). \tag{3.43}$$

За першою схемою (рис. 3.18) кожен процесор обчислює всі компоненти вектора розв'язку деякої точки блоку на кожному кроці ітераційного процесу.

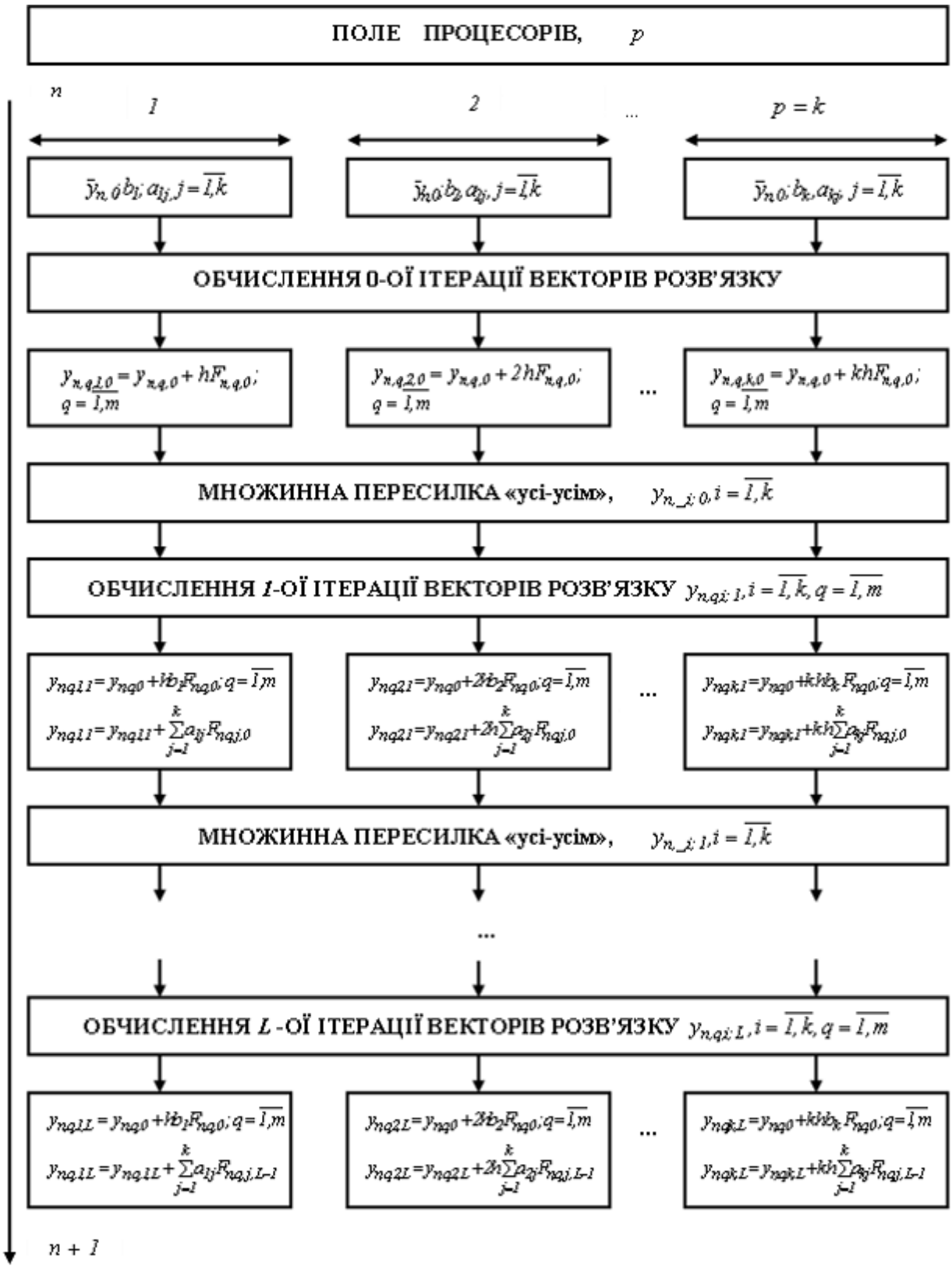


Рис. 3.18. Обчислювальна схема №1 паралельного алгоритму вкладеного однокрокового k -точкового блокового методу інтегрування систем звичайних диференційних рівнянь

За другою схемою (рис. 3.19) кожен процесор обчислює одну компоненту вектору розв’язку для всіх точок блоку на кожному кроці ітераційного процесу.

Визначимо аналітичні вирази для розрахунку часу обміну між процесорами або комунікаційну складову із урахуванням якнайкращої для даного типа операцій топології гіперкуб:

$$T^1_{p,comm} = (L+1)\cdot[t_s \cdot \log_2 k + m(k-1)\cdot t_w],$$

$$T^2_{p,comm} = (L+1)\cdot[t_s \cdot \log_2 m + k(m-1)\cdot t_w].$$

Друга обчислювальна схема має час обчислень і загальний час виконання на багатопроцесорній ОС менші, ніж перша. Числові значення коефіцієнтів прискорення варіюються при зміні машинно-залежних комунікаційних констант та складності правої частини, але при цьому коефіцієнт прискорення обчислень для першої обчислювальної схеми завжди нижчий, ніж для другої.

Досліджені обчислювальні схеми блокового вкладеного методу вирішення СЗДР мають обмеження на кількість використаних процесорів. Розробимо паралельний алгоритм, що не містить таких обмежень і що враховує як паралелізм методу, так і системний (рис. 3.20).

Нехай є паралельна ОС із p процесорами; розіб'ємо процесорне поле на k груп по кількості точок у блоці. Кількість процесорів в одній групі дорівнює $\hat{p} = \lceil p/k \rceil$, кількість компонент системи, обчислюваних одним процесором, складає $\hat{m} = \lceil m/\hat{p} \rceil$. Кожна група відповідає за обчислення розв’язку в одній точці блоку для всієї СЗДР на одній ітерації. Час послідовної реалізації цього алгоритму обчислюється за формулами (3.38), час паралельної реалізації для топології гіперкуб визначається, як:

$$T^3_{p,comp} = (L+1)\cdot \sum_{i=1}^{\lceil mk/p \rceil} T_{f_i} + \lceil mk/p \rceil \cdot [2Lk + 4L + 3]\cdot t_{op}, \qquad (3.44)$$

$$T^3_{p,comm} = (L+1)\cdot[t_s \cdot \log_2 mk + \lceil mk/p \rceil (mk-1)\cdot t_w] \qquad (3.45)$$

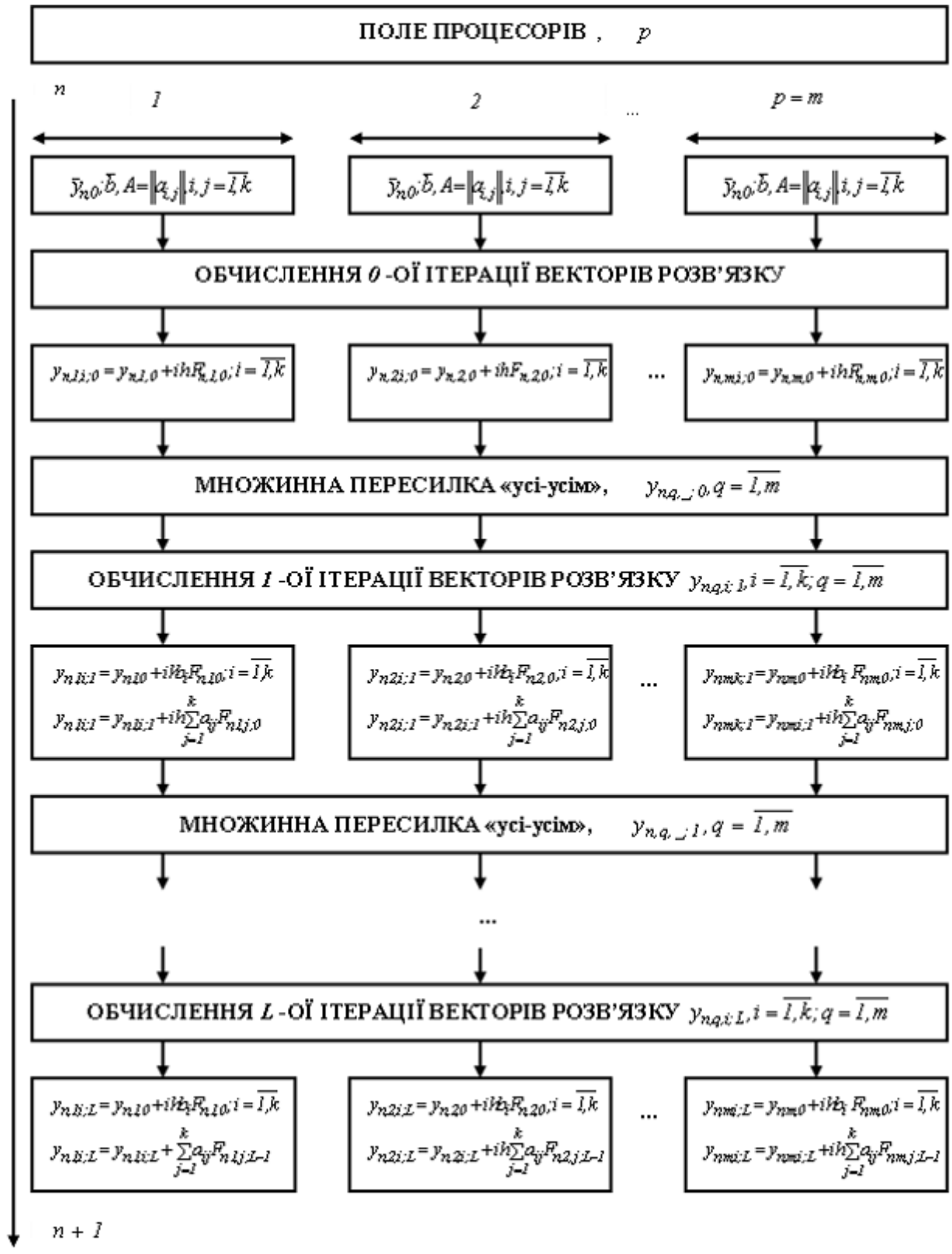


Рис. 3.19. Обчислювальна схема №2 паралельного алгоритму вкладеного однокрокового k -точкового блокового методу інтегрування СЗДР

Потенційні характеристики всіх трьох алгоритмів при довільних правих частинах близькі до ідеальних (лінійне прискорення і одинична

ефективність) за умови, що кількість процесорів дорівнює максимальному ступеню паралелізму кожного методу.

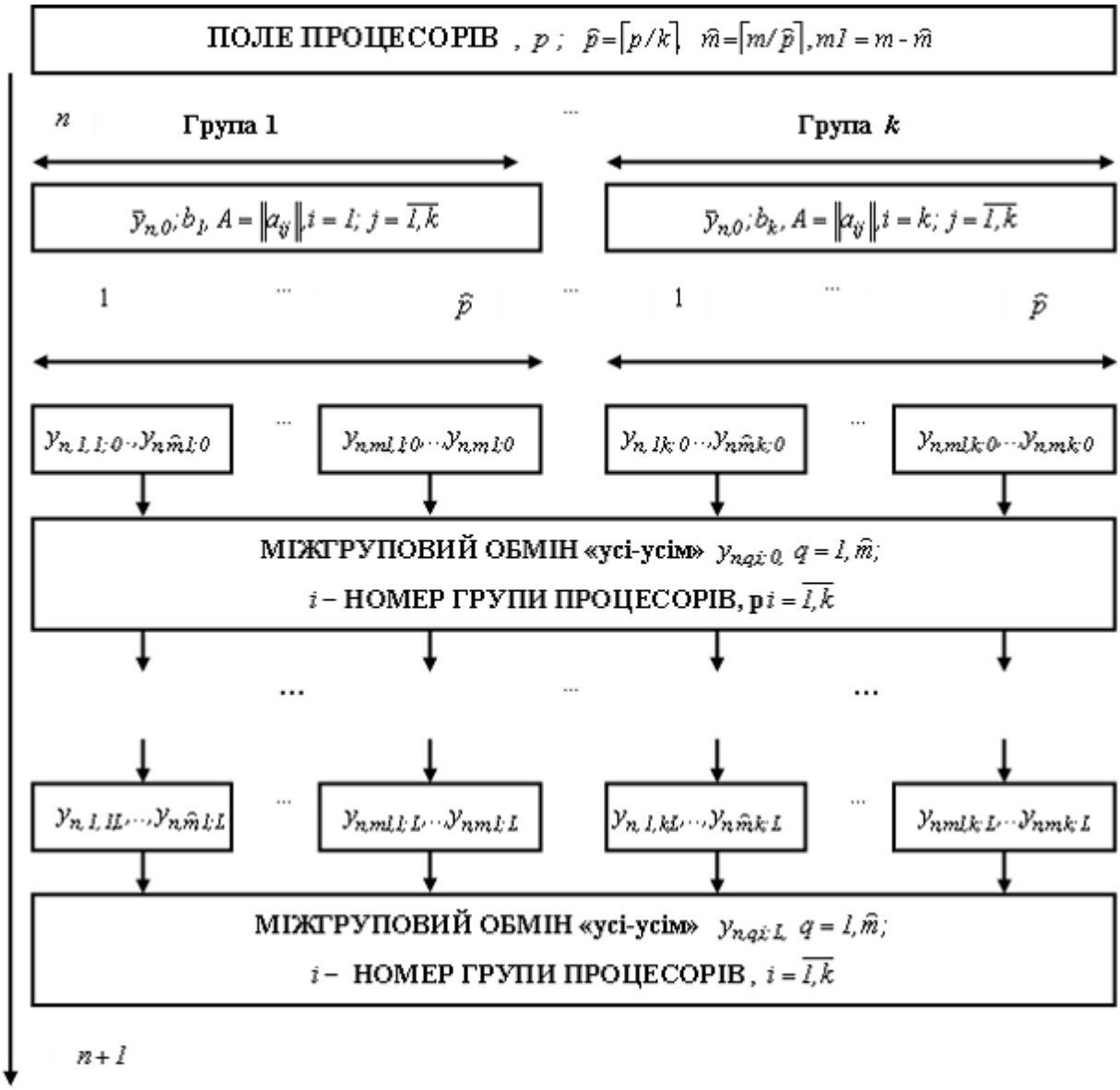


Рис. 3.20. Обчислювальна схема №3 паралельного алгоритму вкладеного однокрокового k-точкового блокового методу інтегрування СЗДР

Через різну складність обмінних операцій і максимально можливу кількість процесорів: перша обчислювальна схема, що використовує тільки паралелізм методу, має найгірші характеристики реального паралелізму. Третя обчислювальна схема володіє безперечною перевагою перед двома іншими (рис. 3.21-3.22).

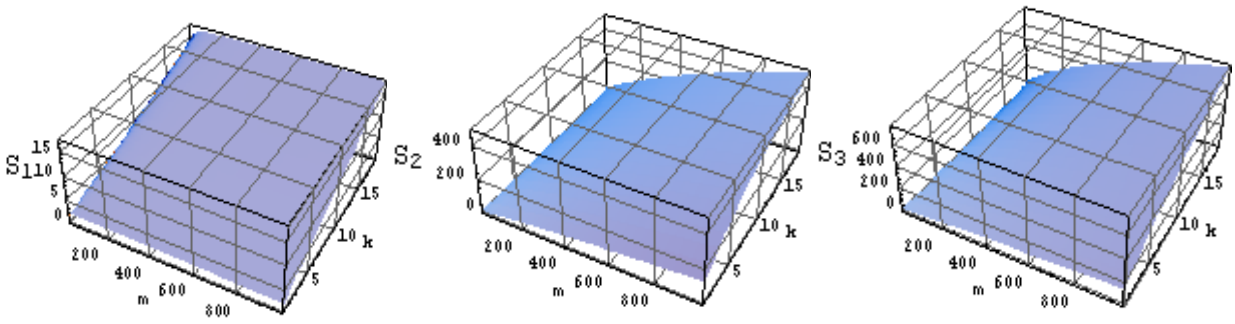


Рис. 3.21. Коефіцієнт прискорення для паралельної реалізації трьох обчислювальних схем від кількості точок у блоці і розмірності задачі

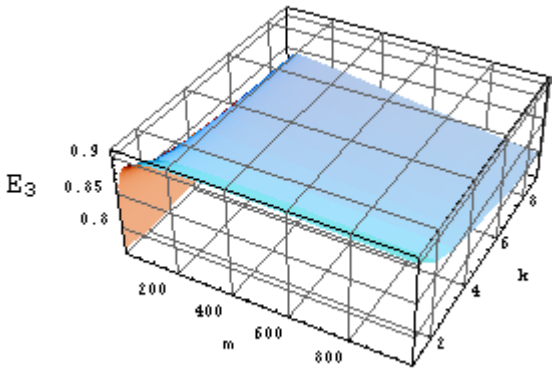


Рис. 3.22. Коефіцієнт ефективності обчислювальної схеми №3

Вочевидь, що якість розроблених алгоритмів розв’язання СЗДР істотно залежить від декількох параметрів, в першу чергу від співвідношення розміру задачі, властивостей чисельного методу та паралельної багатопроцесорної системи. Таким чином, якнайкращі характеристики паралелізму при розв’язанні систем ЗДР блоковими вкладеними методами мають місце при виконанні декількох умов:

- багатовимірність задач,

- складність правих частин,

- наявність високошвидкісних мереж передачі даних із топологією гіперкуб,

- застосування обчислювальної схеми №3, що використовує як системний паралелізм, так і внутрішній паралелізм методу.

3.2. Паралельні алгоритми розв'язання нелінійної задачі Коші повністю неявними однокроковими методами

Чисельне розв'язання систем звичайних диференційних рівнянь (3.1-3.2) повністю неявним s-стадійним методом типа Рунге-Кутти (ПНМРК) можна отримати послідовно по кроках за допомогою наступної обчислювальної схеми:

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s} b_i \bar{k}_i, \\ \bar{k}_i = \bar{f}[x_n + c_i h; y_n + h \cdot \sum\limits_{j=1}^{s} a_{ij} \bar{k}_j], i = \overline{1,s}, \end{cases} \tag{3.46}$$

де s-розмірні вектора $c = (c_1, c_2, ..., c_s)^T$, $b = (b_1, b_2, ..., b_s)^T$ та повністю заповнена матриця $A = (a_{ij}), i, j = \overline{1,s}$ описують унікальний варіант методу й вибираються з міркувань точності.

До переваг повністю неявних однокрокових методів [94] слід віднести добрі характеристики стійкості, достатні для розв'язання жорстких задач Коші. Недоліками цих методів є висока обчислювальна складність, обумовлена необхідністю розв'язання систем нелінійних алгебраїчних рівнянь на базі деякого ітераційного процесу для визначення стадійних коефіцієнтів.

При знаходженні розв'язку СЗДР із використанням повністю неявних методів ms невідомих стадійних коефіцієнтів можуть бути отримані на основі наступної ітераційної схеми:

$$\begin{cases} \bar{k}_i^{(0)} = \bar{f}[x_n + c_i h; \bar{g}_i^{(0)}], \\ \bar{k}_i^{(l+1)} = \bar{f}[x_n + c_i h; \bar{g}_i^{(l)}], \end{cases} \tag{3.47}$$

де $\bar{g}_i^{(0)} = \bar{y}_n$, $\bar{g}_i^{(l)} = \bar{y}_n + h \sum\limits_{j=1}^{s} a_{ij} \bar{k}_j^{(l)}$, $i = \overline{1,s}$, $l = \overline{1,N}$. (3.48)

Тут ітераційний процес, повторений l - разів, представляє $\bar{k}_i^{(l)}$, $i = \overline{1,s}$, як l - ту апроксимацію стадійного коефіцієнта $\bar{k}_i$.

3.2.1. Розробка і оцінка ефективності паралельного алгоритму повністю неявного методу типа Рунге-Кутти для одного звичайного диференційного рівняння

Паралельне розв'язання звичайних диференційних рівнянь на базі повністю неявного методу типу Рунге-Кутти також концентрується на виконанні одного кроку інтегрування. Як і у разі явних методів, алгоритм складається з двох складових:

1) визначення стадійних коефіцієнтів $k_i, i = \overline{1,s}$;

2) наближеного розв'язку на наступному кроці $y_{n+1}(h)$.

Зауважимо, що оскільки метод інтегрування є неявним, то для визначення стадійних коефіцієнтів потрібно розв'язати систему нелінійних у загальному випадку алгебраїчних рівнянь, метод розв'язання яких містить свій внутрішній паралелізм. Більш того, в неявних методах на відміну від явних, навіть для одного рівняння є можливість використовувати паралелізм методу.

Розподілимо обчислення кожного зі стадійних векторів на окремий процесор у межах одного кроку ітерації. Тоді, послідовний алгоритм за обчислювальною схемою 3.47-3.48 може бути представлений, як $N+1$ підзадача, що полягає у виконанні однотипних обчислень одного кроку за методом функціональної ітерації. Тому, як основна макрооперація, вводиться задача обчислення всіх коефіцієнтів $K^{(l)} = \left(k_1^{(l)}, k_2^{(l)}, ..., k_s^{(l)}\right)$ на одному l-тому кроці ітераційного процесу.

Паралельний алгоритм уведеної мікрооперації, розроблений з використанням графа впливу, зображено на рис. 3.23. Кожен із p процесорів обчислювальної системи відповідає за визначення l-той ітерації i-тої компоненти допоміжного вектору $g_i^{(l)}, i = \overline{1,s}$. Потім проводиться обмін даними між процесорами за типом "усі-усім" для обчислення чергової ітерації коефіцієнтів $k_i, i = \overline{1,s}$.

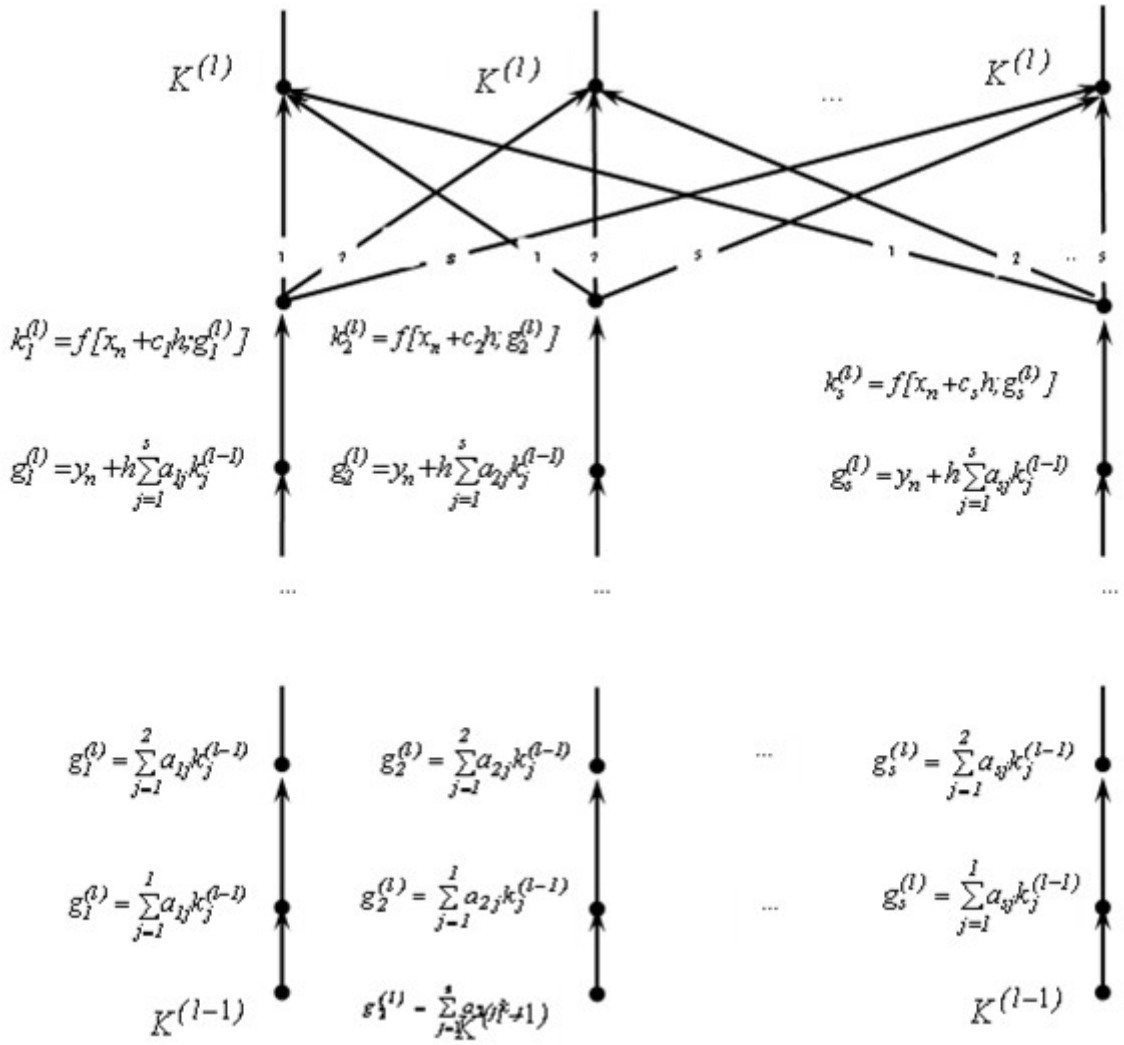


Рис. 3.23. Граф впливу обчислення l – тої ітерації стадійних векторів $K^{(l)} = (k_1^{(l)}, k_2^{(l)}, ..., k_s^{(l)})$ ПНМРК для ЗДР

Рисунок 3.24 представляє паралельний алгоритм вирішення одного диференційного рівняння повністю неявним методом на мультикомп'ютері з p процесорів. Розподіл даних збігається з розподілом даних для приведеної схеми макрооперації, на кожному кроці ітераційного процесу паралельно обчислюються стадійні коефіцієнти, потім послідовно наближене вирішення на $n+1$ кроці. Час послідовного методу для одного рівняння при реалізації ПНМРК може бути описаний наступним співвідношенням:

$$T_1^{2l} = \underbrace{s \cdot (T_f + t_{mul} + t_{ad})}_{k_i^{(0)}} + N \cdot \underbrace{[s^2 \cdot t_{mul} + s^2 \cdot t_{ad} + s \cdot T_f]}_{k_i^{(l)}} + \underbrace{(s+1) \cdot (t_{mul} + t_{ad})}_{y_{n+1}}. \quad (3.49)$$

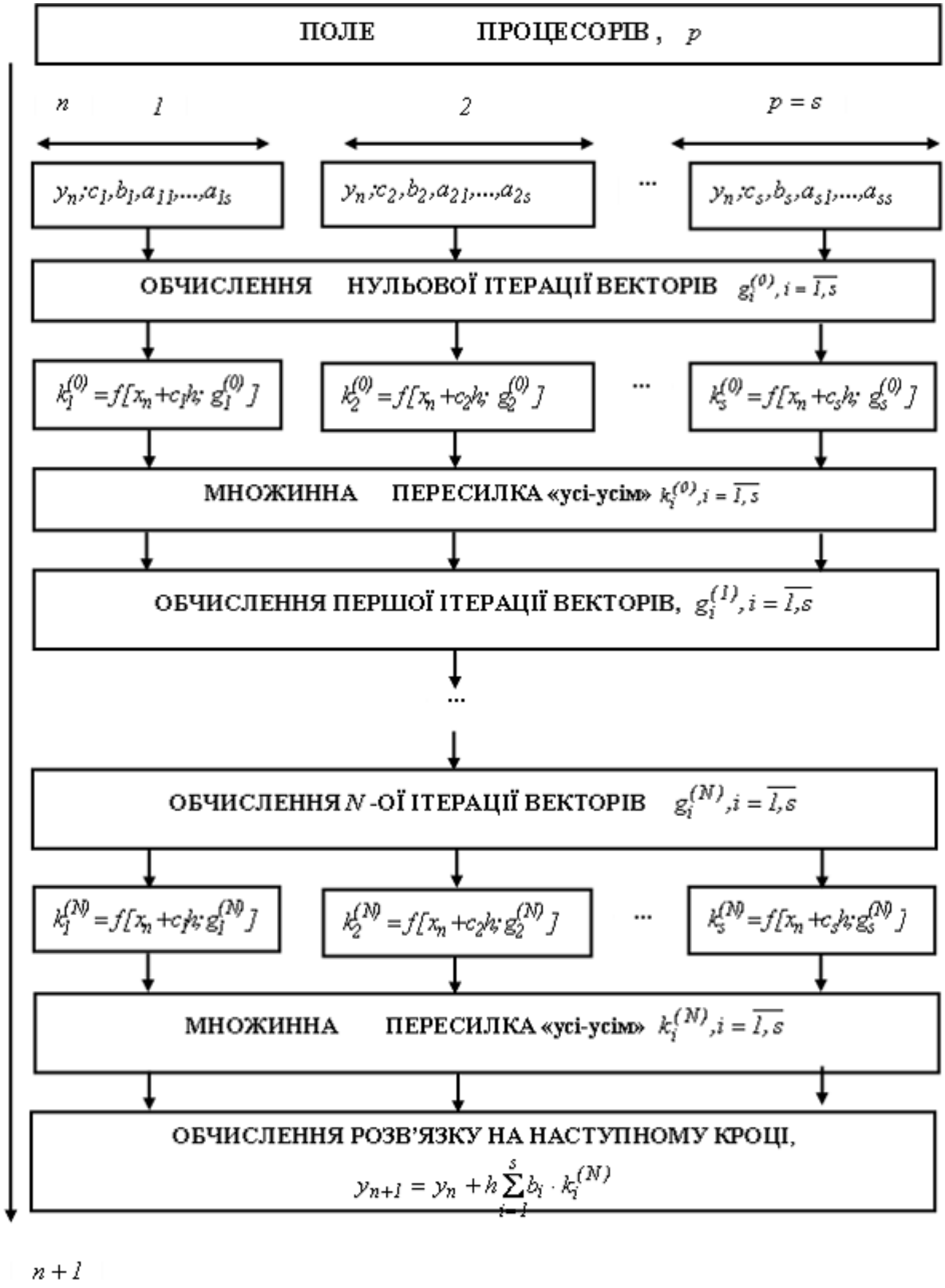


Рис. 3.24. Обчислювальна схема паралельного алгоритму повністю неявного однокрокового методу типа Рунге-Кутти для ЗДР

Для паралельного алгоритму ПНМРК (рис. 3.24) час виконання обчислень на p процесорах без урахування обмінів та інших накладних витрат дорівнює:

$$T_{p,comp}^{21} = \underbrace{T_f + t_{mul} + t_{ad}}_{k_i^{(0)}} + N \cdot \underbrace{[s \cdot t_{mul} + s \cdot t_{ad} + T_f]}_{k_i^{(1)}} + \underbrace{(s+1) \cdot (t_{mul} + t_{ad})}_{y_{n+1}}. \qquad (3.50)$$

Коефіцієнт потенційного прискорення для ЗДР зі складною правою частиною визначається таким чином: $S_{pot}^{21} = T_1^{21} / T_p^{21} \approx \dfrac{s(N+1)T_f}{(N+1)T_f} = s = p$.

Аналогічний результат маємо при $t_{ad} = t_{mul} = t_{op}$. Максимальний ступінь паралелізму алгоритму складає: $Dop = s$. Час, необхідний на реалізацію міжпроцесорних обмінів даного алгоритму визначається $N+1$ раз повтореною множинною операцією за типом “усі-усім” для p процесорів $T_{p,comm}^{21} = (N+1) \cdot T_{all\text{-}to\text{-}all}(p)$:

– кільце: $T_{p,comm}^{21,R} = (N+1) \cdot (t_s + t_w) \cdot (p-1)$;

– тор: $T_{p,comm}^{21,M} = (N+1) \cdot \left[2t_s \cdot (\sqrt{p}-1) + t_w(p-1)\right]$;

– гіперкуб: $T_{p,comm}^{21,H} = (N+1) \cdot [t_s \log_2 p + t_w \cdot (p-1)]$.

Реальна ефективність (рис. 3.25-3.26) паралельних обчислень багато в чому визначається трудомісткістю комунікаційних операцій.

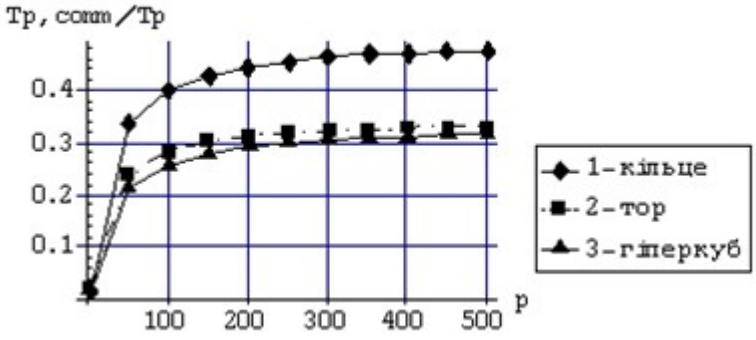


Рис. 3.25. Доля комунікаційних операцій до загальних накладних витрат паралельного алгоритму ПНМРК для ЗДР

Аналіз запропонованих топологій, як для високошвидкісних, так і для низькошвидкісних мереж показав, що, безумовно, найгіршим варіантом для даного алгоритму є з'єднання типу кільце (рис. 3.25).

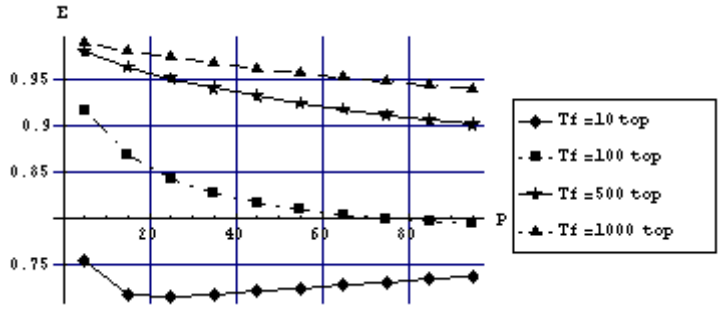


Рис. 3.26. Коефіцієнт ефективності паралельного ПНМРК для ЗДР при різній складності правої частини

Динамічні характеристики ПНМРК для ЗДР мають великий розкид по значеннях. Суттєве значення має такий фактор, як складність правої частини диференційного рівняння (рис. 3.26). Для тривіальних правих частин коефіцієнт ефективності замалий, для домінуючих достатньо великий. Це дозволяє зробити висновок щодо можливості вживання паралельних алгоритмів ітераційних методів при ретельному обліку всіх складових паралельної системи, алгоритму та вхідної задачі для того, щоб це розв’язання було ефективним та таким, що масштабується.

3.2.2. Особливості розпаралелювання повністю неявного методу Рунге-Кутти для СЗДР при рішенні нелінійної задачі

При паралельному розв’язанні систем звичайних диференціальних рівнянь із використанням повністю неявних методів Рунге-Кутти окрім паралелізму методу з'являється можливість використовувати і системний паралелізм.

Вживання ПНМРК до СЗДР вимагає вирішення системи нелінійних рівнянь для визначення $\bar{k}_i = (k_{i1}, k_{i2}, ..., k_{im}), i = \overline{1,s}$:

$$\bar{g}_i^{(0)} = 0,$$

$$\bar{k}_i^{(0)} = F(x_n + c_i h, \bar{y}_n),$$

$$\begin{cases} \bar{g}_i^{(l)} = a_{i1}\bar{k}_1^{(l)} + a_{i2}\bar{k}_2^{(l)} + \ldots + a_{is}\bar{k}_s^{(l)}, \\ \bar{k}_i^{(l)} = F[x_n + c_i h, \bar{y}_n + h \cdot g_i^{(l-1)}], \\ i = 1,\ldots,s; l = 1,\ldots,N. \end{cases} \qquad (3.51)$$

Як і для всіх методів, що вже були розглянуті, розпаралелювання повністю неявних методів типа Рунге-Кутти базується на виконанні одного кроку інтегрування. Послідовний алгоритм можна представити як виконання двох підзадач: вирішення системи нелінійних рівнянь і обчислення чергової апроксимації вирішення. Для здобуття ефективного паралельного алгоритму розв'язання поставленої задачі проаналізуємо внутрішній паралелізм кожної з підзадач. Це дозволить визначити рівень декомпозиції задачі на незалежні процеси і ввести оптимальну множину макрооперацій.

Обчислювальна схема паралельного ПНМРК для розв'язання нелінійної задачі Коші на основі СЗДР приведена на рисунку 3.27. Кількість нелінійних рівнянь, як і кількість невідомих компонент стадійних векторів $k_{ij}\ \left(i = \overline{1,s};\ j = \overline{1,m}\right)$ дорівнює sm. Ітераційний процес, що описується обчислювальною схемою (3.51), є суто послідовним. Проте він дозволяє розподілити обчислення l-тої ітерації $K(h)$ на s незалежних процесів по кількості стадійних векторів, де кожен вектор має розмірність, що дорівнює розміру системи m.

Таким чином, максимальний ступінь паралелізму цієї частини ПНМРК складає $Dop_1 = sm$, для другої підзадачі – $Dop_2 = m$. Якщо скористатися шириною всього паралельного алгоритму, рівною $Dop_1 = p = s \cdot m$, то при реалізації другої підзадачі велика частина процесорів простоюватиме ($Dop_2 = p = m$). Якщо за основу узяти $Dop_2 = m$, то не буде використаний у повному обсязі внутрішній паралелізм першої підзадачі.

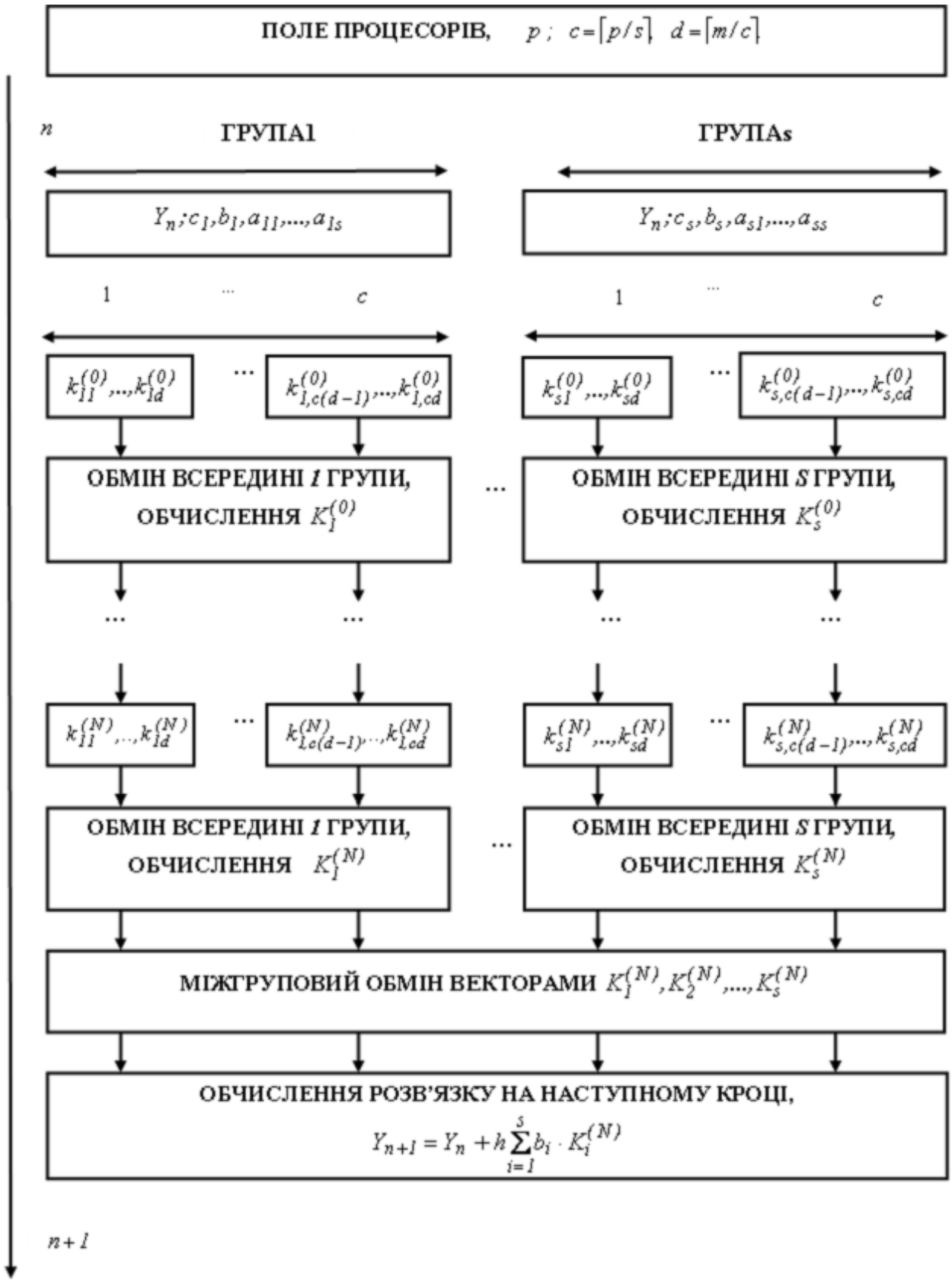


Рис. 3.27. Обчислювальна схема паралельного алгоритму повністю неявного однокрокового методу типу Рунге-Кутти для системи звичайних диференційних рівнянь

Найбільш ефективне вирішення полягає у комбінації першого і другого підходів. Розіб'ємо всю множину процесорів на s груп, кожна група відповідатиме за обчислення одного стадійного коефіцієнта $\bar{k}_i = (k_{i1}, k_{i2}, ..., k_{im}), i = \overline{1,s}$, кількість процесорів в групах однакова і рівна $p_i = \lceil p/s \rceil = c, \forall\, i = \overline{1,s}$. Далі кожен з p_i процесорів s-тої групи паралельно обчислюватиме $d = \lceil m/p_i \rceil$ компонент вектора $\bar{g}_i$ на кожній з $N+1$ ітерацій. Усередині ітераційного процесу кожний процесор у групі обмінюється своїми даними з усіма іншими (обмін «усі-усім» у групі) процесами групи компонентами вектора $\bar{g}_i$. Відмітимо, що немає необхідності у внутрішньо груповому обміні векторами $\bar{k}_i, i = \overline{1,s}$, оскільки на наступному кроці ітерації кожен процесор в групі обчислюватиме ті ж компоненти стадійних векторів.

Після закінчення ітераційного процесу необхідно перерозподілити стадійні вектора між процесорами, для цього має бути проведений обмін даними між групами – міжгруповий обмін за типом “усі-усім”. При обчисленні апроксимації розв’язку на кожному тимчасовому кроці на одному процесорі розташовується $\lceil m/p \rceil$ компонент вектора розв’язку.

Міжгрупповий обмін може бути здійснений двома способами:

1) кожен перший процесор у групі передає вектор $\bar{k}_i$ в перший елемент кожній іншій групи і проводиться груповий обмін у групі;

2) міжгрупова передача за типом “усі-усім”.

Визначимо динамічні характеристики отриманого алгоритму.

Час обчислень за послідовною реалізацією дорівнює:

$$T_1^{22} = (N+1)s \cdot \sum_{i=1}^{m} T_{f_i} + (2Nms^2 + 2ms + 2s + 2m) \cdot t_{op}. \qquad (3.52)$$

Час обчислень для алгоритму рис. 3.27 залежить від типу паралельної обчислювальної системи. Для SIMD архітектури у загальному випадку обчислення різних компонент вектора функцій F не може бути виконане паралельно, отже, на це буде потрібно:

$$T_F = \sum_{i=1}^{m} T_{f_i} .$$

У той же час для MIMD систем з урахуванням можливості паралельної реалізації звернення до правої частини СЗДР, час для обчислення ПНМРК дорівнює:

$$T_{p,comp}^{22} = \underbrace{t_{mul} + t_{ad} + \left\lceil \frac{ms}{p} \right\rceil T_F}_{\bar{k}^{(0)}} + N \cdot \underbrace{\left\lceil \frac{ms}{p} \right\rceil [T_F + s(t_{mul} + t_{ad})]}_{\bar{k}^{(1)}} + \underbrace{\left\lceil \frac{m}{p} \right\rceil (s+1)(t_{mul} + t_{ad})}_{\bar{y}_{n+1}}, \quad T = \max_{i=1}^{m} T_{f_i} \quad . \quad (3.53)$$

Легко переконатися, що коефіцієнт потенційного прискорення практично лінійно залежить від кількості процесорів як при $T_F \gg t_{op}$, так і при $T_F \approx t_{op}$. Накладні витрати на операції обміну включають час пересилок даних всередині групи та між групами:

$$T_{p,comm}^{22} = T_{p,comm;1}^{22} + T_{p,comm;2}^{22} .$$

Проаналізуємо складність комунікаційної складової паралельного алгоритму ПНМРК для СЗДР. Внутрігрупова операція обміну даними є $N+1$ разів повтореною операцією множинної пересилки за типом “усі-усім” на p_i процесорах, обсяг даних, що пересилаються, рівний d:

$$T_{p,comm;1}^{22} = (N+1) \cdot T_{all\text{-}to\text{-}all}(d, p_i) .$$

Для різних базових топологій внутрігруповий обмін потребує:

– кільце: $T_{p,comm;1}^{22,R} = (N+1) \cdot \left(t_s + \left\lceil \frac{ms}{p} \right\rceil \cdot t_w \right) \cdot \left(\left\lceil \frac{p}{s} \right\rceil - 1 \right);$

– тор: $T_{p,comm;1}^{22,M} = (N+1) \cdot \left[2\left(\sqrt{\left\lceil \frac{p}{s} \right\rceil} - 1 \right) \cdot t_s + \left\lceil \frac{ms}{p} \right\rceil \cdot \left(\left\lceil \frac{p}{s} \right\rceil - 1 \right) \cdot t_w \right];$

– гіперкуб: $T_{p,comm;1}^{22,H} = (N+1) \cdot \left[\log_2 \left(\left\lceil \frac{p}{s} \right\rceil \right) \cdot t_s + \left\lceil \frac{ms}{p} \right\rceil \cdot \left(\left\lceil \frac{p}{s} \right\rceil - 1 \right) \right] \cdot t_w .$

Міжгрупповий обмін виконується двічі для стадійних коефіцієнтів та для перегрупування даних після обчислення чергової апроксимації розв’язку.

Міжгрупповий обмін вимагає наступних тимчасових витрат:

– кільце: $T_{p,comm;21}^{22,R} = \left[2t_s + \left(\left\lceil \frac{m}{p} \right\rceil + \left\lceil \frac{ms}{p} \right\rceil \right) \right] \cdot t_w \cdot (p - 1);$

– тор: $T_{p,comm;21}^{22,M} = \left[4\left(\sqrt{p} - 1\right) \cdot t_s + \left(\left\lceil \frac{m}{p} \right\rceil + \left\lceil \frac{ms}{p} \right\rceil \right) \cdot (p - 1) \cdot t_w \right];$

– гіперкуб: $T_{p,comm;21}^{22,H} = 2 \cdot \log_2 p \cdot t_s + \left(\left\lceil \frac{m}{p} \right\rceil + \left\lceil \frac{ms}{p} \right\rceil \right) \cdot (p - 1) \cdot t_w.$

Динамічні характеристики отриманого паралельного методу розв’язання систем звичайних диференційних рівнянь ПНМРК аналогічні відповідним характеристикам інтегрування ЗДР.

3.3. Порівняльний аналіз неявних однокрокових методів розв’язання задачі Коші для ЗДР

Аналіз ефективності визначення апостеріорної локальної похибки в даному розділі базується на вживанні наступних неявних однокрокових чисельних схем:

1) розпаралелених повністю неявних методів типа Рунге-Кутти;

2) паралельних блокових багатоточкових методів.

Проведемо порівняння двох класів неявних методів при послідовній і паралельній реалізаціях на основі найбільш ефективного способу оцінки локальної похибки, а, саме, вкладених формул. Основними параметрами, які характеризують ПНМРК, є взаємозв'язані величини – порядок методу r та кількість стадій s. Крім того, при вирішенні систем нелінійних алгебраїчних рівнянь для визначення стадійних коефіцієнтів з'являється такий параметр, як необхідна кількість ітерацій L.

Обчислювальна складність блокових методів залежить від порядку методу r, кількості точок у блоці k і, а також від кількості ітерацій $\hat{L}$ для здобуття розв'язку СНАР необхідної точності. Тимчасові витрати на обчислення, а також обмінні операції при паралельній реалізації описаних методів приведені в таблиці 3.1. Для того, щоб порівняння чисельних методів на базі неявних однокрокових схем було коректним, необхідно:

а) забезпечити один і той же порядок точності r;

б) отримати рішення в ідентичній кількості нових точок k;

у) породжувані ітераційні процеси повинні реалізовувати граничну кількість ітерацій, передбачену кожним з даних методів: $L = 2s$ і $\hat{L} = k$.

Для широко вживаних ПНМРК за теоремами Батчера кількість стадій пов'язана з порядком методу одним з наступних співвідношень [94]: метод типа Гауса $r = 2s$; методи Радо $r = 2s - 1$; методи типа Лобатто $r = 2s - 2$. Візьмемо співвідношення $r = 2s$, тим самим навмисно вибираючи кращий варіант для ПНМРК.

Таблиця 3.1

Тимчасові характеристики s-стадійних ПНМРК та k-точкових блокових однокрокових методів

Методи		Коефіцієнт при T_F	Коефіцієнт при t_{op}	Коефіцієнт при $T_{all-to-all}$
ПНМРК (k - точок)	T_1	$ks(L+1)$	$2k(Ls^2+2s+1)$	
	T_p	$k(L+1)$	$2k(Ls+s+2)$	$k(L+1)$
Блокові методи	T_1	$\hat{L}k+1$	$2\hat{L}(k^2+2k)+3k$	
	T_p	$\hat{L}+1$	$2\hat{L}k+4k)+3$	$\hat{L}+1$

У той же час для блокових однокрокових методів порядок може бути визначений через кількість точок одного блоку: $r = k + 1$. Тоді кількість

стадій ПНМРК і кількість точок в блоці багатоточкового методу зв'язані наступним співвідношенням: $r = 2s = k + 1 \Rightarrow k = 2s - 1$. Використовуючи отримані співвідношення, приведемо всі тимчасові характеристики до одного параметра, нехай це буде кількість стадій s (табл. 3.2).

Таблиця 3.2

Тимчасові характеристики ПНМРК і блокових однокрокових методів, приведені до одного параметру s

Методи		Коефіцієнт при T_F	Коефіцієнт При t_{op}	Коефіцієнт при $T_{all-to-all}$
ПНМРК (k - точок)	T_1	$4s^3 - s$	$8s^4 - 4s^3 + 8s^2 - 2$	
	T_p	$4s^2 - 1$	$8s^3 + 6s - 4$	$4s^2 - 1$
Блокові методи	T_1	$4s^2 - 4s + 2$	$16s^3 - 8s^2 + 2s - 1$	
	T_p	$2s$	$8s^2 + 1$	$2s$

Відмітимо, що серед множини неявних методів типу Рунге-Кутти для порівняння вибрані саме повністю неявні методи, оскільки вони володіють ідентичними характеристиками стійкості, що й блокові однокрокові методи [113-115], а також через оптимальне співвідношення між порядком та кількістю стадій методів. Порівняємо час виконання послідовних алгоритмів даних методів. При домінуванні в обчисленнях часу звернення до правої частини ЗДР $T_F >> t_{op}$ обсяг обчислень для ПНМРК в s разів більший, ніж для блокових однокрокових методів:

$$T_1^{RK} / T_1^{BM} \approx \frac{ks(L+1)\cdot T_F}{(\hat{L}k+1)\cdot T_F} \approx \frac{s(4s^2-1)}{k^2+1} = \frac{4s^3 - s}{4s^2 - 4s + 2} \approx s .$$

Аналогічно, для паралельної реалізації ПНМРК та блокових багатоточкових однокрокових алгоритмів маємо:

$$\frac{T_p^{RK}}{T_p^{BM}} \approx \frac{k(L+1)\cdot T_F}{(\hat{L}+1)\cdot T_F} \approx \frac{4s^2-1}{k+1} = \frac{4s^2-1}{2s} \approx 2s.$$

Для тривіальної правої частини виходить аналогічний результат. Послідовна і паралельна реалізації обчислення розв'язку у нових k точках сітки інтегрування вимагають більших накладних витрат для неявних методів Рунге-Кутти у порівнянні з блоковими методами (рис. 3.28). Відмітимо, що для паралельних алгоритмів ця різниця збільшується майже у 2 рази, що підтверджується експериментом на тестових задачах.

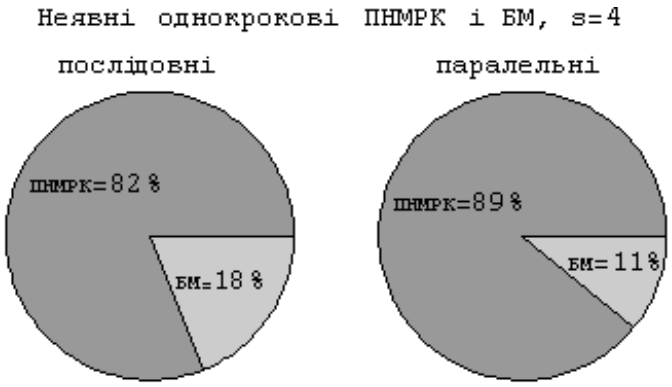


Рис. 3.28. Співвідношення часу виконання s-стадійних ПНМРК і k-точкових блокових вкладених однокрокових методів

Оцінка ефективності розпаралелювання проводиться на основі показників прискорення обчислень (рис. 3.29-3.30) у порівнянні зі своїм послідовним алгоритмом та у порівнянні з якнайкращим з послідовних алгоритмів, що були розглянуті.

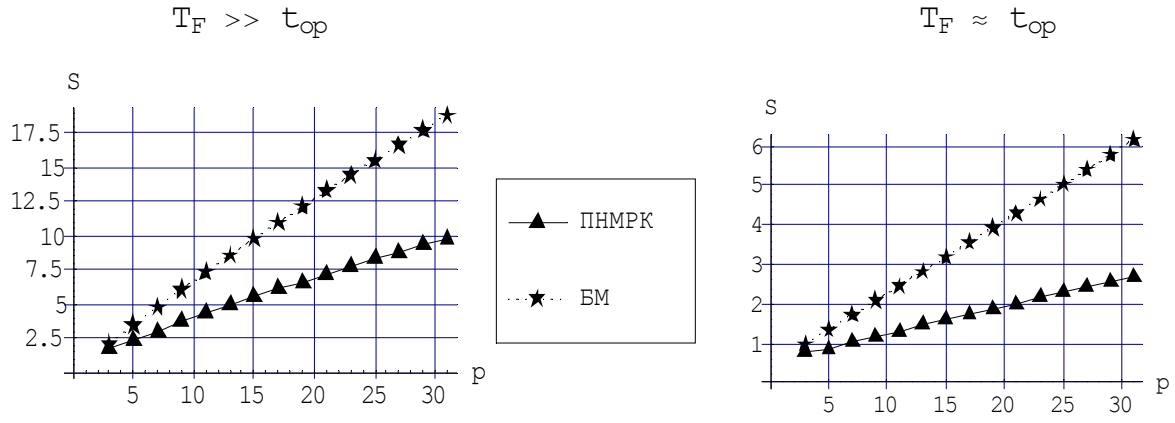


Рис. 3.29. Абсолютні коефіцієнти прискорення вкладених повністю неявних методів Рунге-Кутти та блокових однокрокових методів

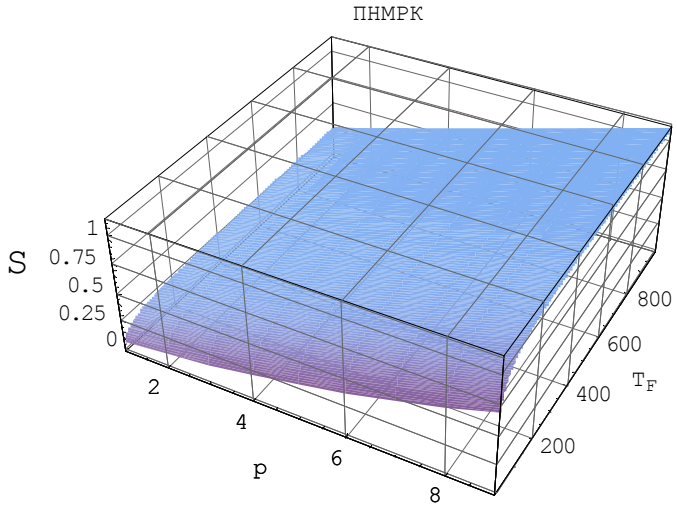


Рис. 3.30. Відносні коефіцієнти прискорення вкладених ПНМРК

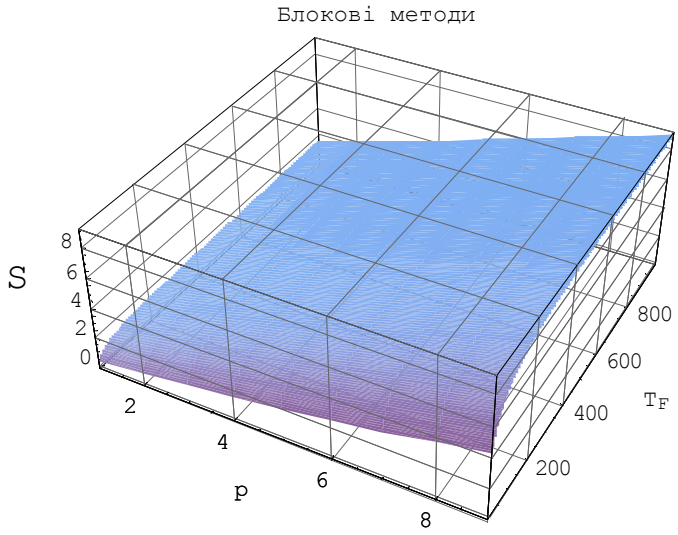


Рис. 3.31. Відносні коефіцієнти прискорення вкладених блокових однокрокових методів

На основі теоретичного аналізу виконання та аналогічних емпіричних досліджень можна зробити висновок, що для будь-яких засобів оцінки локальної похибки блокові багатоточкові однокрокові методи є найбільш ефективними з точки зору обчислювальних витрат у порівнянні з повністю неявними методами Рунге-Кутти того ж порядку точності.

ГЛАВА IV

ПАРАЛЛЕЛЬНІ МЕТОДИ РОЗВ'ЯЗАННЯ ЛІНІЙНОЇ ЗАДАЧІ КОШІ З ОЦІНКОЮ ЛОКАЛЬНОЇ ПОХИБКИ

Дана глава присвячена розробці та аналізу ефективності паралельних методів розв'язання задачі Коші для систем лінійних звичайних диференційних рівнянь (СЛЗДР) із вбудованими засобами оцінки локальної похибки. Зокрема, запропоновано спеціальні методи вирішення однорідних та неоднорідних СЛЗДР із постійними коефіцієнтами. Як показали дослідження, урахування специфіки задачі дозволяє отримати більш ефективні паралельні обчислювальні алгоритми, ніж у випадку стандартних чисельних схем. Експоненційний метод відноситься до спеціальних методів чисельного інтегрування початкової задачі Коші для СЛЗДР, заснований на точному представленні розв'язку в аналітичній формі й наближеному обчисленні матричної експоненти [115,117].

У загальному вигляді задачу Коші для однорідних СЛЗДР із постійними коефіцієнтами можна записати таким чином:

$$\begin{cases} \bar{y}'(x) = A \cdot \bar{y}(x), \\ \bar{y}(x_0) = \bar{y}_0, \ A = const, \end{cases} \tag{4.1}$$

де $\bar{y} = (y_1, y_2, ..., y_m)^T$ – вектор невідомих, $\bar{y}_0 = (y_{10}, y_{20}, ..., y_{m0})^T$ – вектор початкових умов, $A = \|a_{ij}\|, i, j = \overline{1,m}$ – матриця коефіцієнтів лінійної системи, елементи якої є константами.

Точний розв'язок задачі Коші виду (4.1) потребує обчислення матричної експоненти:

$$\begin{cases} \bar{y}(x_n + h) = e^{hA} \cdot \bar{y}(x_n), \\ e^{hA} = F(hA) = \sum\limits_{i=0}^{\infty} \dfrac{(hA)^i}{i!}. \end{cases} \tag{4.2}$$

Наближений розв'язок на кроці можна побудувати, використовуючи апроксимацію матричної експоненти відрізком ряду Тейлора при малому значенні h:

$$\begin{cases} \bar{y}_{n+1} = F_r(hA) \cdot \bar{y}_n, \\ F_r(hA) = \sum\limits_{i=0}^{r} \dfrac{(hA)^i}{i!}. \end{cases} \tag{4.3}$$

Отримана обчислювальна схема інтегрування однорідних СЛЗДР із постійними коефіцієнтами відповідає різницевому методу порядку $O(h^r)$.

На основі експоненційного методу побудовано три різні паралельні алгоритми: 1) експоненційний та правило Рунге; 2) вкладений - експоненційний; 3) експоненційний з локальною екстраполяцією.

Матрична експонента $F(hA)$ володіє наступною властивістю: $F(hA) = F\left(\frac{h}{2}A\right) \cdot F\left(\frac{h}{2}A\right)$, що дозволяє досить легко вбудовувати алгоритми визначення локальної похибки на основі дублювання кроку і локальної екстраполяції в обчислювальні схеми явних чисельних методів для лінійних СЗДР. Застосування експоненційного методу для вкладених форм робить практично непотрібними обчислення за формулою вищого порядку. Запропоновані алгоритми особливо ефективні при вирішенні задач із великою константою Ліпшиця і вимагають меншого обсягу обчислень, ніж стандартні методи розв'язання лінійної задачі.

4.1. Підвищення ефективності паралельного розв'язання лінійної задачі Коші з оцінкою локальної похибки за правилом Рунге

Інтегрування однорідних систем лінійних звичайних диференційних рівнянь із постійними коефіцієнтами на основі експоненційного методу й вбудованого засобу визначення локальної похибки на основі дублювання кроку за правилом Рунге вимагає обчислення вирішення з подвоєним і двічі з половинним кроком:

$$\begin{cases} \bar{y}^{(1)}(x_n + 2h) = F(2hA)\cdot \bar{y}(x_n), \\ \bar{y}(x_n + h) = F(hA)\cdot \bar{y}(x_n), \\ \bar{y}^{(2)}(x_n + 2h) = F(hA)\cdot \bar{y}(x_n + h). \end{cases} \quad (4.4)$$

Використовуючи апроксимацію матричної форми F, а також її властивості для половинного кроку, отримаємо наступну перетворену обчислювальну схему чисельного розв'язання при переході з точки x_n до точки x_{n+1}:

$$\begin{cases} \bar{y}_{n+1}^{(1)} = F^{(1)}\cdot \bar{y}_n = F_r(2hA)\cdot \bar{y}_n = \left(\sum_{i=0}^{r} \frac{(2hA)^i}{i!}\right)\cdot \bar{y}_n, \\ \bar{y}_{n+1}^{(2)} = F^{(2)}\cdot \bar{y}_n = F_r^{\,2}(hA)\cdot \bar{y}_n = \left(\sum_{i=0}^{r} \frac{(hA)^i}{i!}\right)^2\cdot \bar{y}_n, \end{cases} \quad (4.5)$$

де $F_r(hA) = \left(E + hA + \frac{h^2A^2}{2!} + \ldots + \frac{h^rA^r}{r!}\right)$ (4.6)

Перевагами запропонованого методу є фактична відсутність обчислень для проміжної точки з половинним кроком і можливість уведення підготовчого етапу, який виконується до початку інтегрування і обчислює найбільш ресурсоємні операції знаходження матричних форм.

Час для послідовної реалізації експоненціального методу й правила Рунге із урахуванням підготовчого етапу та за умови використання RISC-архітектури дорівнює:

$$T_{1,0}^{11} = [2(r-1)\cdot m^3 + (2r-1)\cdot m^2]\cdot t_{op}. \quad (4.7)$$

$$T_1^{11} = T_{1,0}^{11} + N\cdot(4rm^2 + 6m^2 + m)\cdot t_{op}, \quad (4.8)$$

де N – кількість кроків інтегрування.

Процес обчислення за формулами (4.5-4.6) можна представити, як розв'язання наступних підзадач: до початку інтегрування однократно виконується обчислення ступенів матриці A і множення їх на скаляри, потім на кожному з N кроків інтегрування обчислюються дві матричні форми: $F^{(1)}$ і $F^{(2)}$ і, відповідно, дві апроксимації вирішення: $\bar{y}_{n+1}^{(1)}$ і $\bar{y}_{n+1}^{(2)}$.

Для побудови паралельного експоненціального алгоритму розв’язання лінійної задачі Коші для однорідних СЗДР із постійними коефіцієнтами із правилом Рунге скористаємося результатами глави 2. Сукупність підзадач реалізується через наступну множину макрооперацій: виконання однократного матричного множення, обчислення матричної форми як функції кроку інтегрування, обчислення апроксимації розв’язку.

Усі три макрооперації виконуються послідовно, перша $(r-1)$ раз до початку інтегрування, друга і третя на кожному кроці інтегрування двічі. Паралельна реалізація двох основних найбільш ресурсоємних макрооперацій вимагає розпаралелювання операцій множення двох матриць і матриці на вектор. Для даних паралельних застосувань топологічним вирішенням, що адекватно відображує логічний зв'язок між незалежними процесами, є квадратна сітка або її замкнутий еквівалент – 2D-тор. На такій топологічній схемі ефективно виконуються матричні операції, що являються основними складовими експоненціальних форм (4.5-4.6). У цьому підрозділі обчислення матричного добутку буде виконано за блоковим варіантом систолічного алгоритму [70,119].

Алгоритм першої основної макрооперації експоненціального методу з правилом Рунге, а саме, множення двох квадратних матриць $C = A \times B$, для багатопроцесорної паралельної системи з замкнутою сіткою $p \times p$ процесорів наведено на рисунку 4.1. Використані в алгоритмі розв’язання лінійної задачі матриці розмірності $m \times m$ розбиваються на $q^2, q = p$ квадратних блоків порядку $k = \lceil m/q \rceil$. Для простоти міркувань, нехай $m \vdots q$ і $m \vdots k$.

Розподіл початкових даних і результатів по процесорах наступний: кожен процесор сітки з номером $\langle i,j \rangle$ містить два блоки початкових матриць A_{ij}, B_{ij}, і відповідає за обчислення блоку матриці результату C_{ij}. Спочатку, здійснюється косе зрушення вліво по рядках для блоків матриці A $:\leftarrow_i A$ і косе зрушення по стовпцях вгору для блоків матриці $B : B\uparrow^j$.

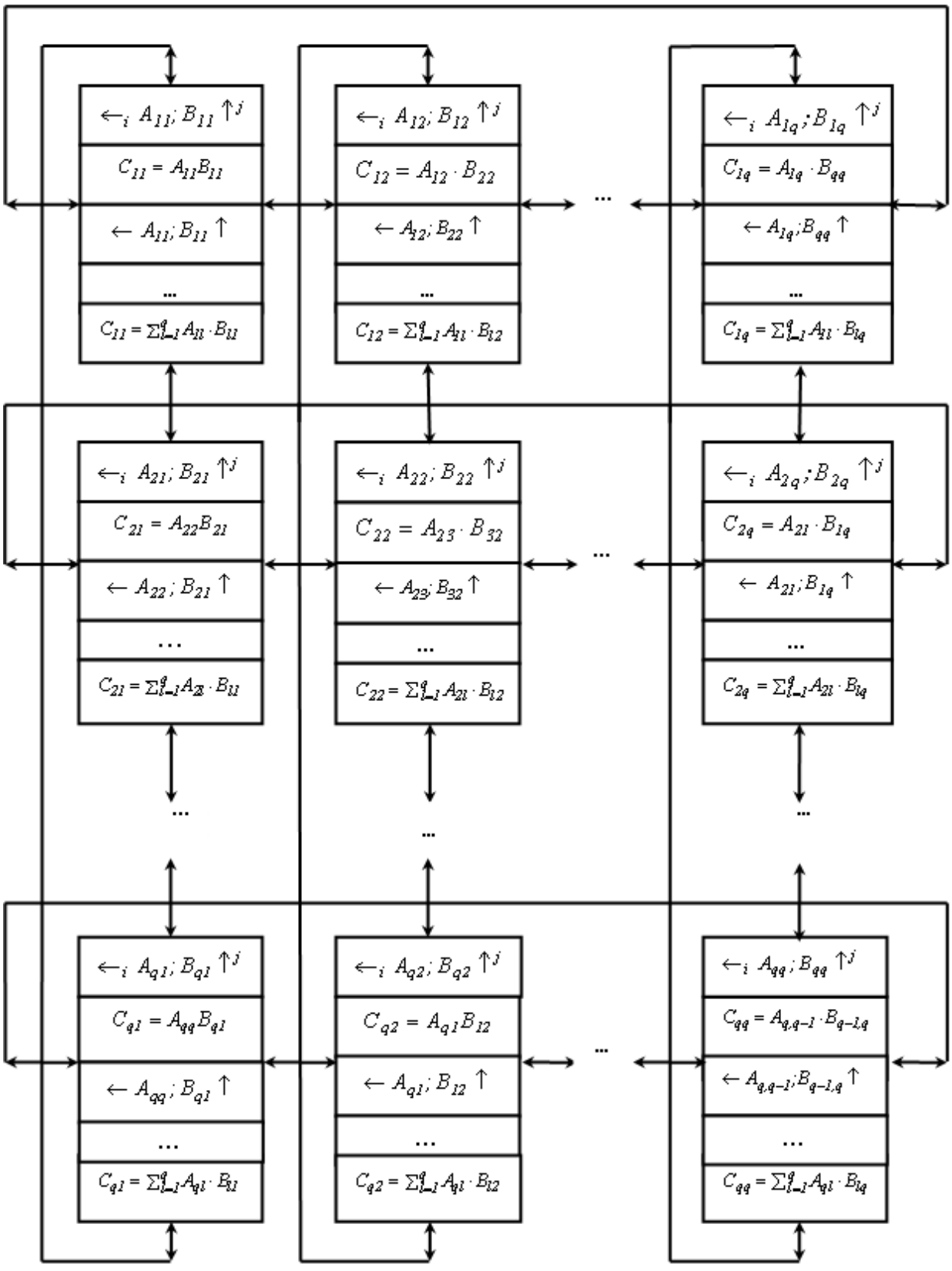


Рис. 4.1. Паралельний алгоритм першої макрооперації експоненціального методу розв'язання СЛЗДР із правилом Рунге, блокового систолічного множення матриць, $A \underset{bl-sys}{\times} B = C$

Далі над блоками виконуються ті ж дії, які виконуються систолічним алгоритмом над елементами матриць, причому складання і добуток матричних блоків виконуються на одному процесорі. Потім проводиться одиночне зрушення вліво для блоків матриці $A: \leftarrow A$ і вгору для блоків матриці $B: B\uparrow$, описані дії повторюються. У результаті кожний процесор із номером $<i,j>$ отримує блок матриці результату. Час реалізації арифметичних операцій при блоковому систолічному множенні визначається, як час q разів виконаної операції добутку блоків матриць розмірності $k \times k$ і дорівнює (табл. 4.1): $T_{p,comp}^{A \times B bl-sys} = (2m^3 / p^2) \cdot t_{op}$.

Час на попередню розсилку даних складається з часу виконання косого зрушення по рядках вліво для матриці A і косого зрушення по стовпцях вгору для матриці B: $T_{p,comm,0}^{A \times B bl-sys} = 2(p-1) \cdot [t_s + (m^2 / p^2) \cdot t_w]$.

Додатково, на кожному з $p-1$ кроків алгоритму проводиться пересилка блоків матриці A вліво і матриці B вгору сусіднім процесорам замкнутої сітки. Тоді, загальний час на операції обміну по блоковому систолічному алгоритму складає (табл. 4.1):

$$T_{p,comm}^{A \times B bl-sys} = 4(p-1) \cdot [t_s + (m^2 / p^2) \cdot t_w].$$

Алгоритм другої основної макрооперації експоненціального методу розв'язання СЛЗДР із правилом Рунге полягає у виконанні операції блокового систолічного множення матриці A розмірності $m \times m$ на вектор Y розміру m за наявності $p \times p$ процесорів (рис. 4.2).

Нехай $C_i = \sum_{l=1}^{q} A_{il} \cdot Y_l, i = \overline{1,q}$, де q^2 - кількість блоків, $q = p$, k - розмір блоку $k = \lceil m/q \rceil$, $Y_1 = (y_1,...,y_k), ..., Y_q = (y_{m-k+1},...,y_m)$ - підвектора розмірності k вектора $Y = \|y_i\|, i = \overline{1,m}$. Спочатку, блоки вектора Y розподіляються на всі процесори існуючої обчислювальної системи. Процесор із номером $<i,j>$ містить блок матриці $A: A_{ij}$ і частину вектора $Y: Y_i$.

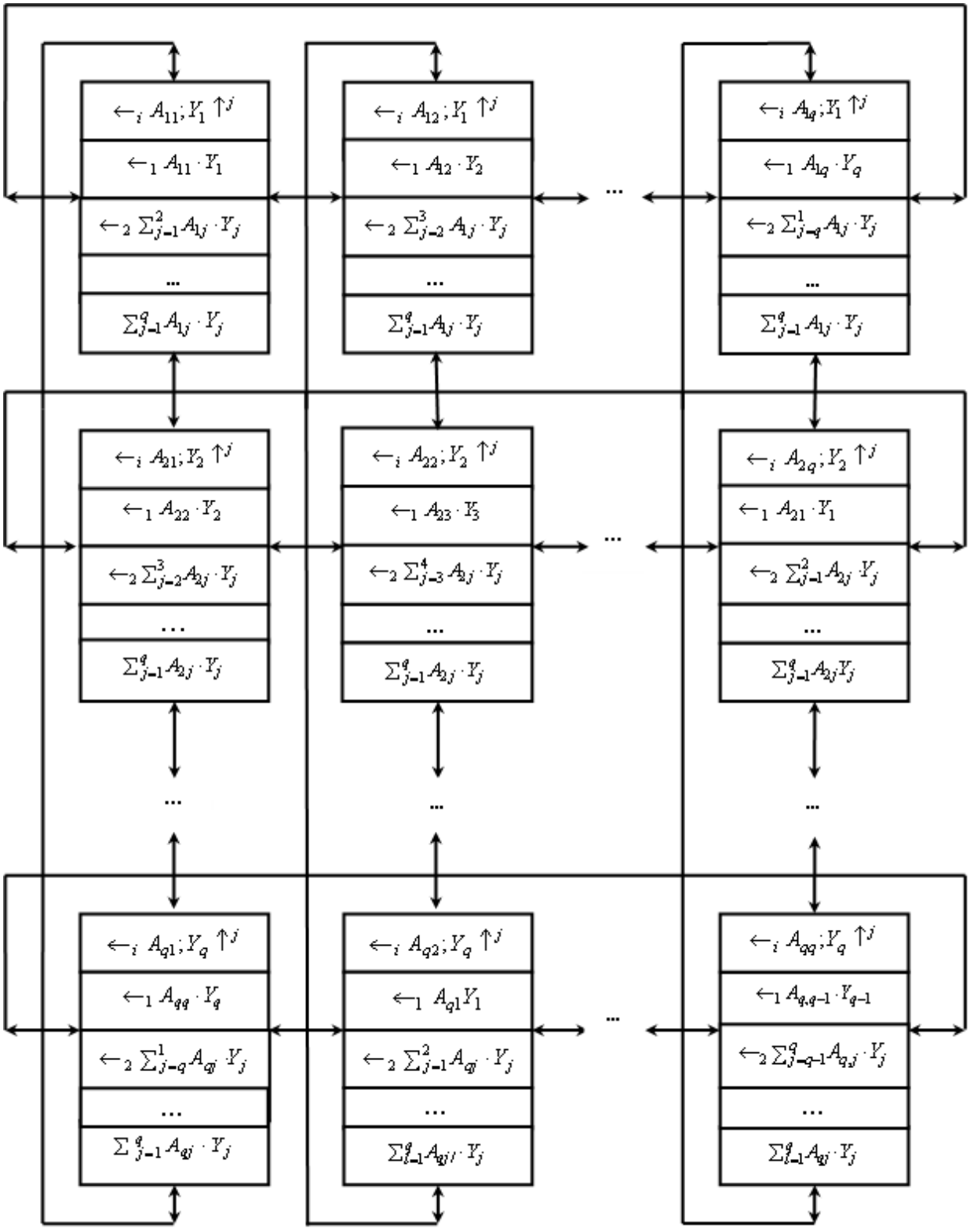


Рис. 4.2. Паралельний алгоритм другої основної макрооперації експоненціального методу розв'язання СЛЗДР із правилом Рунге, блокового систолічного множення матриці на вектор, $A \underset{bl-sys}{\times} Y$

Проводиться косе зрушення вгору по стовпцях для блоків вектора Y: $Y_i \uparrow^j$, косе зрушення вліво по рядках для блоків матриці A: $\leftarrow_i A_{ij}$. Після цього виконується добуток кожного блоку матриці на підвектор паралельно всіма процесорами. У результаті на основі алгоритму каскадного підсумовування отримуємо матрицю, кожен рядок якої містить відповідну частину вектора результату.

Час виконання арифметичних операцій для описаного алгоритму складається з часу обчислення однократного добутку $A_{ij} \cdot Y_i$ і реалізації каскадного підсумовування підвекторів і дорівнює:

$$T_{p,comp}^{A \times Y\ bl-sys} = [(2m^2 / p^2) + (m / p) \cdot \lceil \log_2 p \rceil] \cdot t_{op}.$$

Час обмінів включає первинний етап, що складається з пересилок елементів при дублюванні підвекторів Y та обмінів блоками при косому зрушенні для матриці A: $T_{p,comm,0}^{A \times Y\ bl-sys} = 2(p-1) \cdot [t_s + (m^2 / p^2 + m / p) \cdot t_w]$.

Таблиця 4.1

Час паралельного виконання матричних/векторних добутків для блокового систолічного алгоритму

Вид операції	Час обчислень у t_{op}
Добуток матриць, $A \times A$	$2m^3 / p^2$
Добуток матриці на вектор, $A \times Y$	$\frac{2m^2}{p^2} + \frac{m}{p}\lceil \log_2 p \rceil$

Для даного алгоритму на кожному кроці інтегрування здійснюється косе зрушення по стовпцях підвекторів вектора розв'язання Y та обміни при реалізації каскадного підсумовування (табл. 4.2):

$$T_{p,comm}^{A \times Y\ bl-sys} = \left(2^{\lceil \log_2 p \rceil} + p - 2\right) \cdot [t_s + (m / p) \cdot t_w].$$

Таблиця 4.2

Час комунікаційних операцій паралельного виконання матричних для блокового систолічного алгоритму при топології 2D-тор

Вид операції	Час комунікаційних операцій	
	0-й крок	i - й крок
Добуток матриць, $A \times A$	$2(p-1)\left(t_s + \frac{m^2}{p^2} t_w\right)$	$2(p-1)\left(t_s + \frac{m^2}{p^2} t_w\right)$
Добуток матриці на вектор, $A \times Y$	$2(p-1)\left(t_s + \left(\frac{m^2}{p^2} + \frac{m}{p}\right) t_w\right)$	$\left(2^{\lceil \log_2 p \rceil} + p - 2\right)\left(t_s + \frac{m}{p} t_w\right)$

Розглянемо паралельні методи вирішення лінійних СЗДР у сукупності з правилом Рунге на базі алгоритмів макрооперацій, що були введені вище. Первинний розподіл даних визначається особливостями виконання операцій блокового систолічного множення. Паралельний алгоритм розв'язання СЛЗДР із правилом Рунге на основі експоненційного методу (рис. 4.3) містить 0-й етап, який виконується один раз до початку інтегрування і обчислює блоки матриць: $A^2, A^3, ..., A^r$. Для цього на кожному процесорі з номером $< i, j >$ замкнутої сітки $p \times p$ розташовуються блоки матриці коефіцієнтів при невідомих A_{ij}, $i, j = \overline{1,q}; q = p$. За блоковим систолічним алгоритмом множення $(r-1)$ раз виконується обчислення відповідних ступенів блоків матриці A і $(2r-1)$ раз множення матриць на константи. Причому процесор p_{ij} містить блоки $< i, j >$ матриць $A^2/2!, ..., (2A)^r/r!$. Час обчислень та обмінів для підготовчого етапу складає:

$$T^{11}_{p,comp,0} = \left[(r-1)\left(\frac{2m^3}{p^2}\right) + (2r-1)\frac{m^2}{p^2}\right] \cdot t_{op}, \tag{4.9}$$

$$T^{11}_{p,comm,0} = (r-1)(3p-2) \cdot \left(t_s + \left(m^2/p^2\right) \cdot t_w\right). \tag{4.10}$$

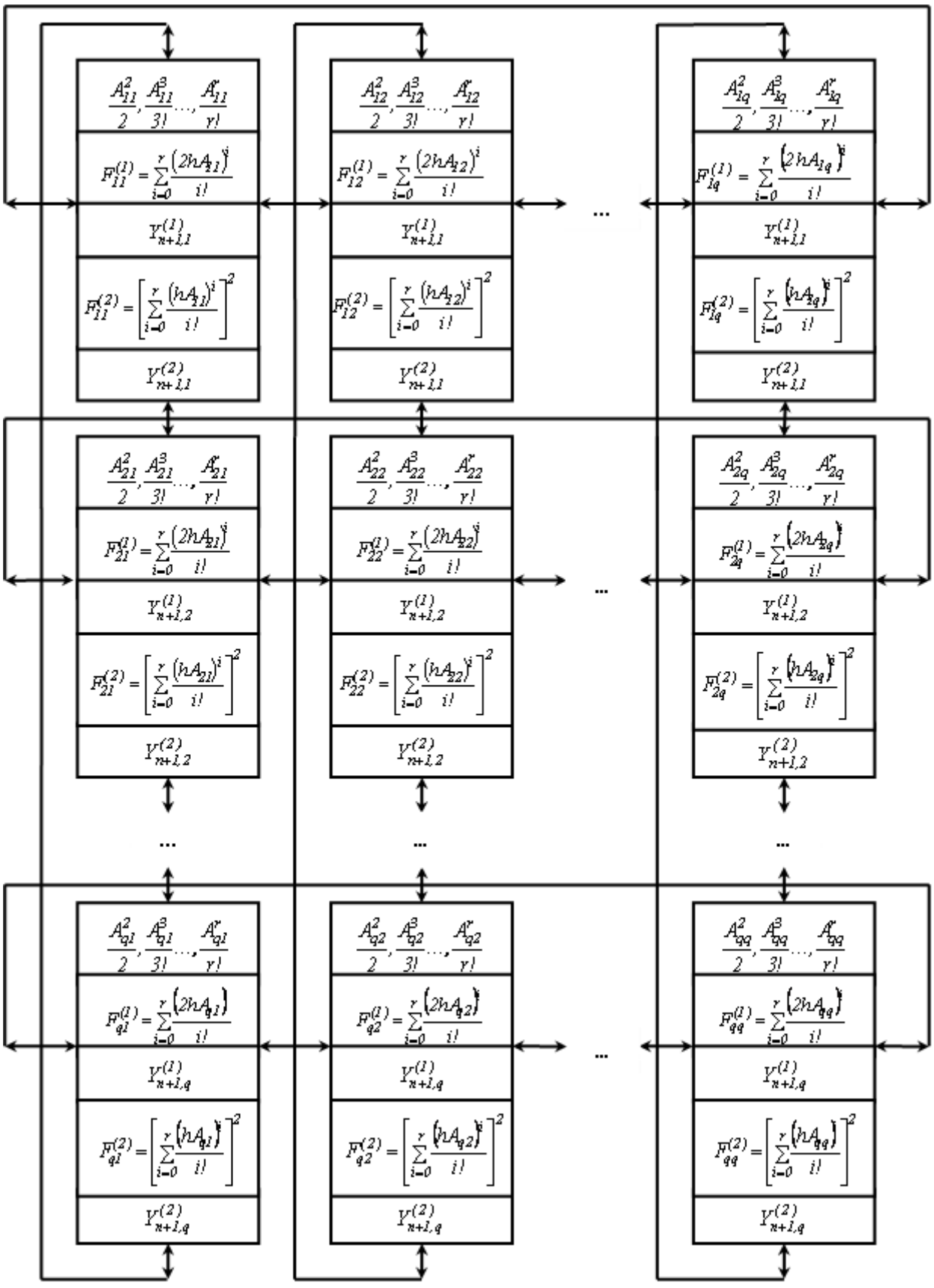


Рис. 4.3. Обчислювальна схема паралельного експоненційного алгоритму вирішення СЛЗДР із оцінкою локальної похибки за правилом Рунге

Далі виконується N кроків інтегрування, що реалізують обчислення апроксимацій розв'язку через матричні форми. На рисунку 4.3 демонструється один такий крок. Послідовне обчислення двох матричних форм і відповідних апроксимацій компенсується паралельними обчисленнями над блоками матриць/векторів початкових і проміжних даних. На кожному процесорі з номером $< i, j >$ замкнутої сітки $p \times p$ розташовуються обчислені блоки матриці коефіцієнтів при невідомих, а також блок вектора рішення попереднього кроку інтегрування. Спочатку на процесорі p_{ij} обчислюється блок матричної форми для подвоєного кроку $F^{(1)}(2hA)$ і, на основі використання методу блокового систолічного множення матриці на вектор, отримуємо апроксимацію розв'язку з подвоєним кроком $Y_{n+1}^{(1)}$. Причому процессор p_{ij} містить лише i - й блок вектора розв'язку за рахунок виконання операції каскадного підсумовування. Таке розбиття даних дозволяє уникнути додаткових пересилок даних і забезпечити рівномірне завантаження процесорів. Аналогічно, обчислюються блоки матричної форми $F^{(2)}$ і апроксимація наближеного розв'язку з половинним кроком $Y_{n+1}^{(2)}$.

Обчислювальна складність отриманого паралельного алгоритму з врахуванням 3 множень матриці на вектор, $A \times Y$, визначається як:

$$T_{p,comp}^{11} = \left[(4r + 6)\frac{m^2}{p^2} + (1 + 3\lceil \log_2 p \rceil)\frac{m}{p} \right] \cdot t_{op}, \tag{4.11}$$

у свою чергу, комунікаційна складова дорівнює:

$$T_{p,comm}^{11} = 2(p - 1) \cdot \left(t_s + \frac{m^2}{p^2} t_w \right) + \left(2^{\lceil \log_2 p \rceil + 1} + p - 3 \right) \cdot \left(t_s + \frac{m}{p} t_w \right). \tag{4.12}$$

Для оцінки ефективності розробленого експоненційного алгоритму проведемо порівняння із загальноприйнятою схемою обчислень на базі s - стадійного ЯМРК аналогічного порядку точності $O(h^r)$. Правило Рунге для

стандартного явного однокрокового методу інтегрування у вживанні до однорідної СЛЗДР із постійними коефіцієнтами реалізується через схему:

$$\begin{cases} \bar{y}_{n+1}^{(1)} = \bar{y}_n + 2h \cdot \sum\limits_{i=1}^{s} b_i \cdot \bar{k}_i^{(1)}(2h), \quad \bar{k}_i^{(1)} = A \cdot \left(\bar{y}_n + 2h \cdot \sum\limits_{l=1}^{i-1} c_{il} \cdot \bar{k}_l^{(1)} \right), \\ \bar{y}_{n+\frac{1}{2}} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s} b_i \cdot \bar{k}_i(h), \quad \bar{k}_i = A \cdot \left(\bar{y}_n + h \cdot \sum\limits_{l=1}^{i-1} c_{il} \cdot \bar{k}_l \right), i = \overline{1,s}, \\ \bar{y}_{n+1}^{(2)} = \bar{y}_{n+\frac{1}{2}} + h \cdot \sum\limits_{i=1}^{s} b_i \cdot \bar{k}_i^{(2)}(h), \quad \bar{k}_i^{(2)} = A \cdot \left(\bar{y}_{n+\frac{1}{2}} + h \cdot \sum\limits_{l=1}^{i-1} c_{il} \cdot \bar{k}_l^{(2)} \right). \end{cases} \tag{4.13}$$

Розпаралелювання обчислювальної схеми (4.13) концентрується на виконанні одного кроку інтегрування (рис. 4.4). На кожному процесорі з номером $\langle i,j \rangle$ замкнутої сітки розташовуються блоки матриці коефіцієнтів при невідомих A_{ij}, $i,j = \overline{1,q}; q = \sqrt{p}$ і блок вектора розв'язку попереднього кроку інтегрування $Y_{n,i}$. Використовуючи операцію блокового систолічного множення матриці на вектор, по крокам отримуємо s стадійних векторів. Причому процесор p_{ij} обчислюватиме лише i-й блок векторів: $K_{1,i}^{(1)}, K_{2,i}^{(1)}, ..., K_{s,i}^{(1)}$, який бере участь в обчисленні відповідного блоку вектора розв'язку $Y_{n+1,i}^{(1)}$. Аналогічно, обчислюються дві інші апроксимації вирішення $Y_{n+1/2}$, $Y_{n+1}^{(2)}$. Час реалізації паралельного алгоритму, описаного схемою рисунку 4.4, у базових операціях лінійної алгебри складає:

$$T_p^{12} = (3s-1) \cdot T_p^{A \times Y} + (1.5s^2 + 2) \cdot T_p^{Y+Y} + (1.5s^2 + 4.5s + 1) \cdot T_p^{c \cdot Y}, \tag{4.14}$$

де $T_p^{A \times Y}$ – час паралельного множення матриці на вектор, T_p^{Y+Y} – час паралельного складання двох векторів, $T_p^{c \cdot Y}$ – час паралельного множення вектора на скаляр.

Для порівняння асимптотичної складності двох паралельних алгоритмів розв'язання СЛЗДР із правилом Рунге (рис. 4.3-4.4) обмежимося обліком найбільш ресурсоємної операції, а, саме, множення матриці на вектор.

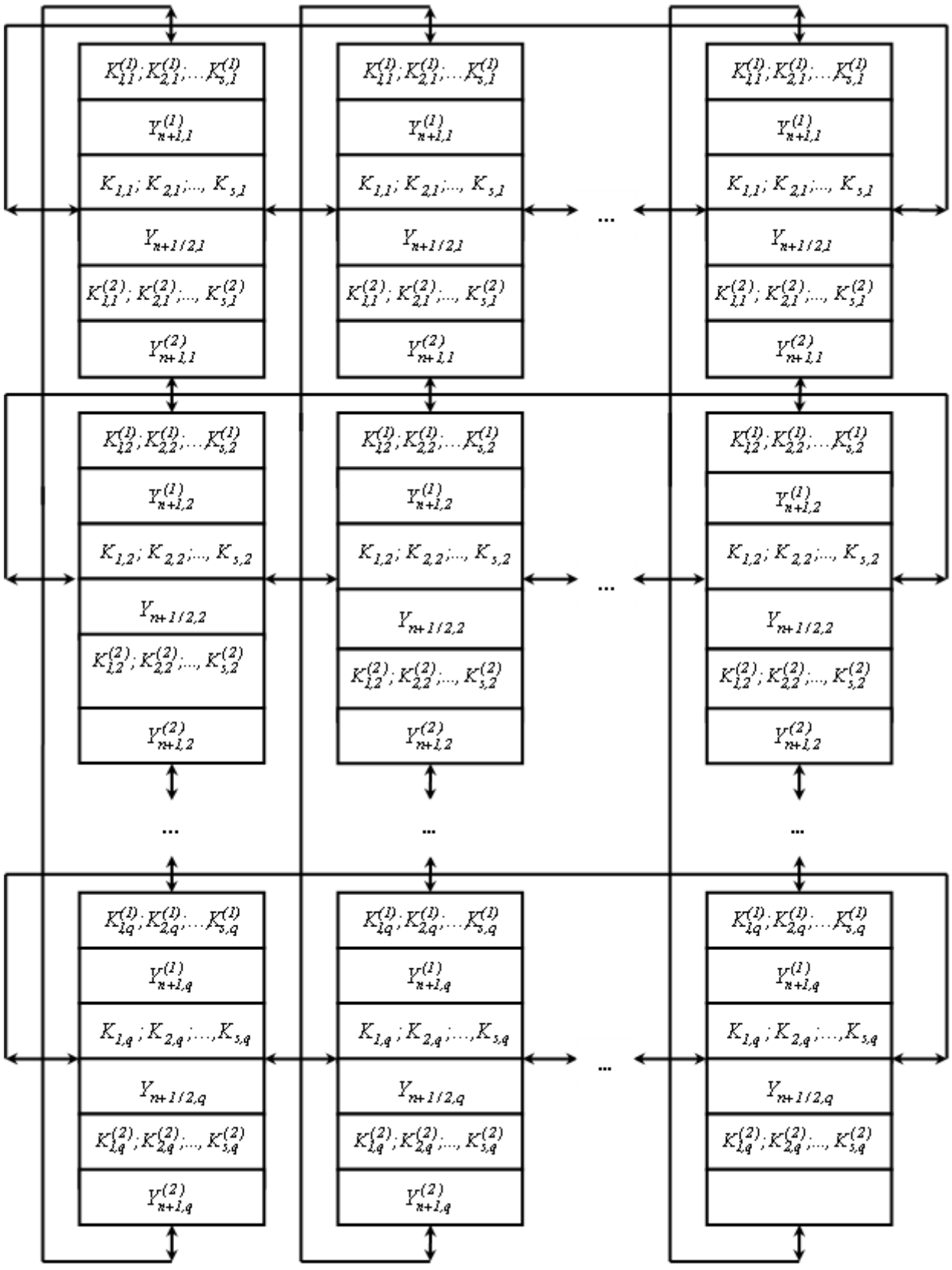


Рис. 4.4. Обчислювальна схема паралельного алгоритму вирішення СЛЗДР із оцінкою локальної похибки за правилом Рунге на базі стандартного явного однокрокового методу

Справедливо наступне співвідношення: $T_p^{12} / T_p^{11} \approx \frac{(3s-1)}{3} \approx s$, тобто стандартна схема вирішення СЛЗДР має обчислювальну складність у s разів більшу, ніж експоненційна. Аналогічний результат має місце при обліку повного часу виконання даних алгоритмів, що обчислюється експериментальним шляхом (рис. 4.5).

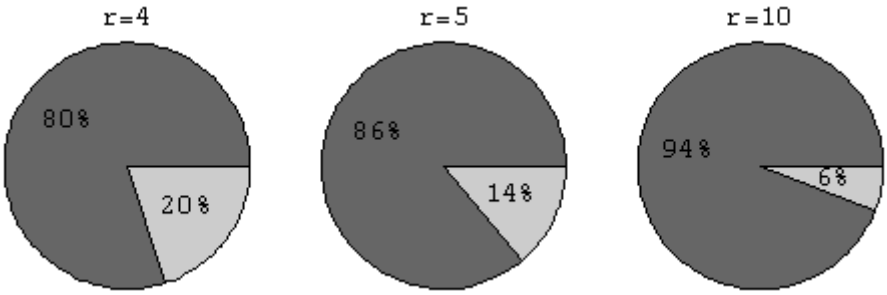


Рис. 4.5. Співвідношення часу реалізації стандартної явної однокрокової T_p^{12} і експоненційної T_p^{11} схем із правилом Рунге від порядку методу

Формула (4.14) визначає нижню межу часу виконання даного паралельного алгоритму через співвідношення між порядком, r та кількістю стадій s явної однокрокової схеми: $s = r + c,\ c = 1,2...$ Для методів високих порядків тимчасові витрати на паралельну реалізацію стандартної схеми більш ніж у s разів перевищують відповідні витрати на експоненційну схему і ця розбіжність збільшується із зростанням порядку методу: $r\uparrow \Rightarrow\ c\uparrow \Rightarrow\ s\uparrow \Rightarrow \left(T_p^{12} / T_p^{11}\right)\uparrow$.

4.2. Вкладені паралельні методи чисельного розв'язання систем лінійних звичайних диференційних рівнянь

Вживання ідеї вкладених форм у поєднанні з експоненційним методом вирішення передбачає обчислення двох апроксимацій розв'язку суміжних порядків точності.

В одній точці інтегрування маємо: $\bar{y}_{n+1}$ порядку $O(h^r)$ та $\hat{\bar{y}}_{n+1}$ порядку $O(h^{r+1})$ із вживанням матричної експоненти:

$$\begin{cases} \hat{\bar{y}}_{n+1}(x_n + h;\ \bar{y}_n) = F_{r+1}(hA)\cdot \bar{y}_n, \\ \bar{y}_{n+1}(x_n + h;\ \bar{y}_n) = F_r(hA)\cdot \bar{y}_n. \end{cases} \tag{4.15}$$

Звичайно формула вищого, $(r+1)$-го, порядку використовується лише для оцінки локальної похибки, а у якості апроксимації розв'язку на кроці інтегрування приймається формула r-го порядку. Тому обчислювальна схема вкладеного експоненціального методу після елементарних перетворень набирає вигляду:

$$\begin{cases} \bar{y}_{n+1} = \left(E + hA + \dfrac{h^2A^2}{2!} + \ldots + \dfrac{h^rA^r}{r!} \right)\cdot \bar{y}_n; \\ d_n = \left\| \frac{h^{r+1}}{(r+1)!} A^{r+1}\cdot \bar{y}_n \right\|. \end{cases} \tag{4.16}$$

Обчислювальна схема (4.16) дозволяє зменшити час кожного кроку інтегрування за рахунок створення підготовчого етапу, який виконується до початку основного розрахунку і містить операції, які не пов'язані безпосередньо з номером кроку. Основними операціями цього етапу будуть обчислення ступенів матриці коефіцієнтів однорідної системи A і множення їх на константи. Відмітимо, що ці операції були найбільш трудомісткими в схемі (4.16). Час розв'язку у точці x_{n+1} за перетвореною схемою вкладеного експоненційного методу для однопроцесорної ОС складається із 2 складових: часу виконання підготовчого етапу й часу реалізації обчислень на кроці інтегрування. Час підготовчого етапу обчислюється, як:

$$T_{1,0}^{21} = \left(2m^3 + m^2\right)\cdot r\cdot t_{op}. \tag{4.17}$$

Час послідовного виконання одного кроку інтегрування за схемою (4.16) визначається з урахуванням того факту, що немає необхідності в обчисленні обох наближень, досить знати $\bar{y}^{n+1}$ і d_n:

$$T_1^{21} = \left(2m^2r + 6m^2\right)\cdot t_{op}. \tag{4.18}$$

Реалізація побудованого паралельного алгоритму, як і в попередніх випадках, ґрунтується на декомпозиційній методиці. Незалежно від наявності або відсутності властивості лінійності для задачі Коші множини макрооперацій, а також графи впливу для макрооперацiйної схеми алгоритму явних вкладених методів (рис. 2.14) і схеми (4.16) збігаються. У свою чергу кожна з макрооперацій може бути розбита на підзадачі лінійної алгебри, паралельні алгоритми яких приведені в підрозділі 4.1 (рис. 4.1, 4.2). Паралельна реалізація вкладеного експоненційного методу також включає підготовчий етап обчислення ступенів матриці коефіцієнтів і, власне, N кроків інтегрування.

На рисунку 4.6 показано один крок інтегрування алгоритму, що складається з обчислення матричної форми, апроксимації розв'язку r - го порядку та величини d_n. До початку інтегрування, на нульовому кроці, кожен із процесорів p_{ij} обчислює відповідний блок $< i,j >$ ступенів $i = \overline{2,r+1}$ матриці коефіцієнтів СЛЗДР на основі блокового систолічного алгоритму множення матриць. На довільному кроці інтегрування кожен процесор із номером $< i,j >$ обчислює відповідний блок форми F_r, а кожен процесор рядка $< i >$ обчислює підвектора вектора розв'язку $Y_{n+1} = (Y_{n+1,1}, Y_{n+1,2}, ..., Y_{n+1,q})$ і величини d_n: $D_n = (D_{n,1}, ..., D_{n,q})$.

Час обчислень і комунікаційна складова підготовчого етапу:

$$T^{21}_{p,comp,0} = (2m^3 / p^2) r \cdot t_{op}, \tag{4.19}$$

$$T^{21}_{p,comm,0} = 2(p-1) r \cdot [t_s + (m^2 / p^2) \cdot t_w]. \tag{4.20}$$

Час паралельної реалізації одного кроку інтегрування для вкладеного експоненційного методу в операціях лінійної алгебри:

$$T^{21}_p = 2 \cdot T^{A \times Y}_p + (r+1) \cdot T^{A+A}_p + (r+1) \cdot T^{c \cdot A}_p, \tag{4.21}$$

де T^{A+A}_p - час паралельного складання матриць, $T^{c \cdot A}_p$ - час паралельного множення матриці на константу.

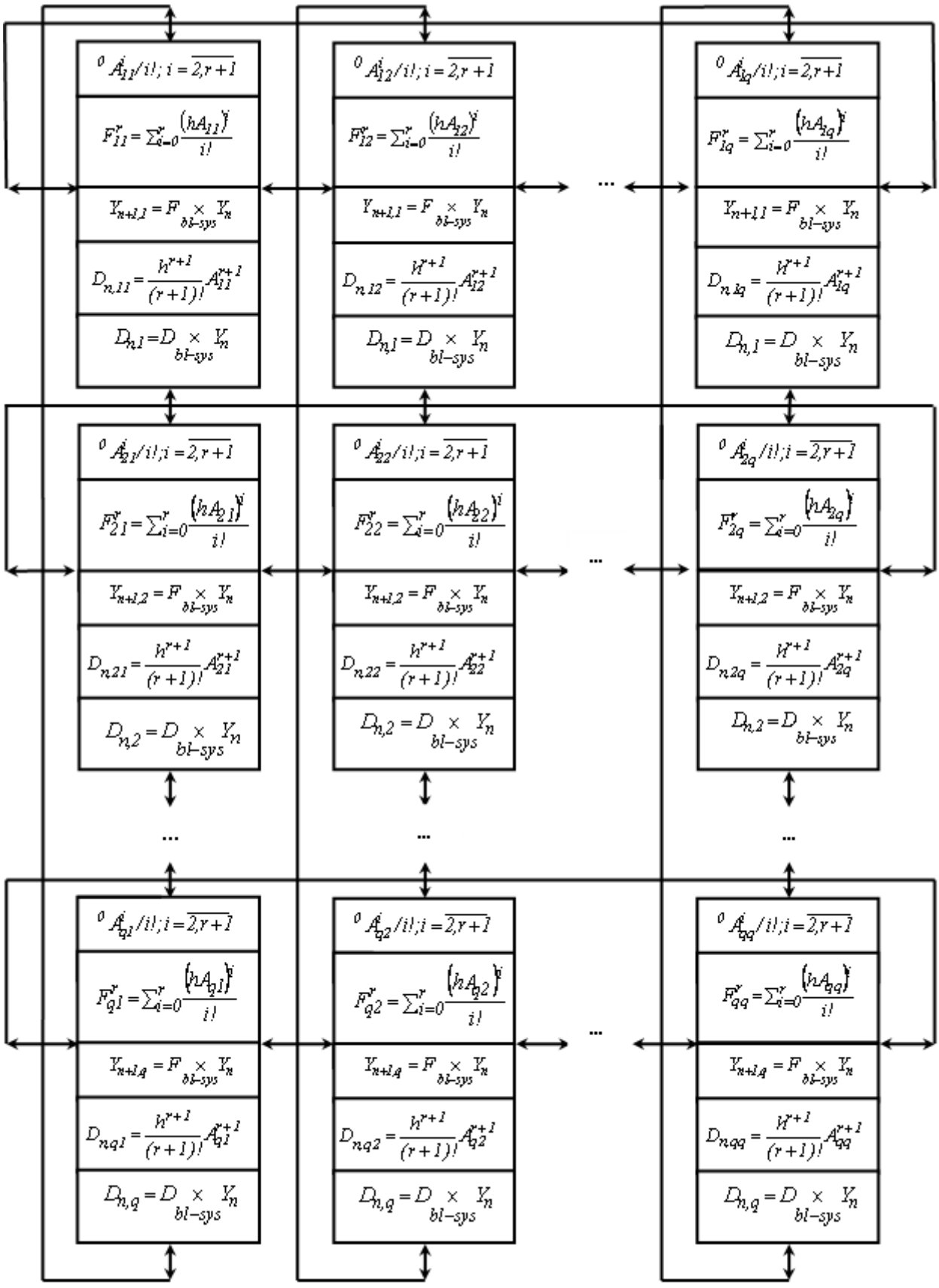


Рис. 4.6. Обчислювальна схема паралельного алгоритму розв’язання СЛЗДР на базі вкладеного експоненційного методу

Відповідно, час обчислень та обмінів для одного кроку вкладеного експоненційного методу дорівнюють:

$$T_{p,comp}^{21} = \left[(2r+6) \cdot \frac{m^2}{p^2} + 2m\lceil log_2 p \rceil \cdot / p \right] \cdot t_{op}, \tag{4.22}$$

$$T_{p,comm}^{21} = 2 \cdot \left(2^{\lfloor log_2 p \rfloor} + p - 2 \right) \cdot \left(t_s + m \cdot t_w / p \right). \tag{4.23}$$

Для порівняння розглянемо другий алгоритм вирішення СЛЗДР з постійними коефіцієнтами на основі методу вкладених форм, заснований на використанні стандартної явної однокрокової схеми:

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s} b_i \cdot \bar{k}_i, \quad \hat{\bar{y}}_{n+1} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s'} \hat{b}_i \cdot \bar{k}_i, \\ \bar{k}_l = A \cdot \bar{y}_n + h \cdot \bar{g}_l, \quad \bar{g}_l = A \cdot \sum\limits_{i=1}^{l-1} c_{li} \cdot \bar{k}_i, l = \overline{1,s'}, d_n = \left\| \hat{\bar{y}}_{n+1} - \bar{y}_{n+1} \right\|. \end{cases} \tag{4.24}$$

Послідовний алгоритм явного ВМРК $r(\hat{r})$ за схемою (4.24) включає обчислення коефіцієнтів $\bar{k}_i$, визначення двох апроксимацій розв'язку $\bar{y}_{n+1}$ і $\hat{\bar{y}}_{n+1}$, обчислення величини d_n. Відповідно, час його реалізації в базових операціях лінійної алгебри дорівнює:

$$T_1^{22} = (s+1) \cdot T_1^{AxY} + \left(\frac{s^2 + 5s + 10}{2} \right) \cdot T_1^{cxY} + \left(\frac{s^2 + 3s + 6}{2} \right) \cdot T_1^{Y+Y}.$$

Для правомірності порівняння вкладених методів вирішення СЛЗДР одного й того ж порядку точності $O(h^r)$ необхідно врахувати співвідношення між кількістю стадій і порядком явного методу. Для вкладеного методу Фельберга 4(5) кількість стадій дорівнює 6, а для методу Дормана-Прінса 8(7) $s = 13$. Далі кількість стадій методу обчислюватиметься за співвідношенням: $s := s + c$, де величина $c = 1,2,...$ і залежить від порядку використаного методу [95]. Тоді час виконання алгоритму за схемою (4.24):

$$T_1^{22} = [(2s+2)m^2 + (s^2 + 3s + 7)m] \cdot t_{op}. \tag{4.25}$$

Побудуємо паралельний алгоритм стандартної обчислювальної схеми явного вкладеного методу (рис. 4.7).

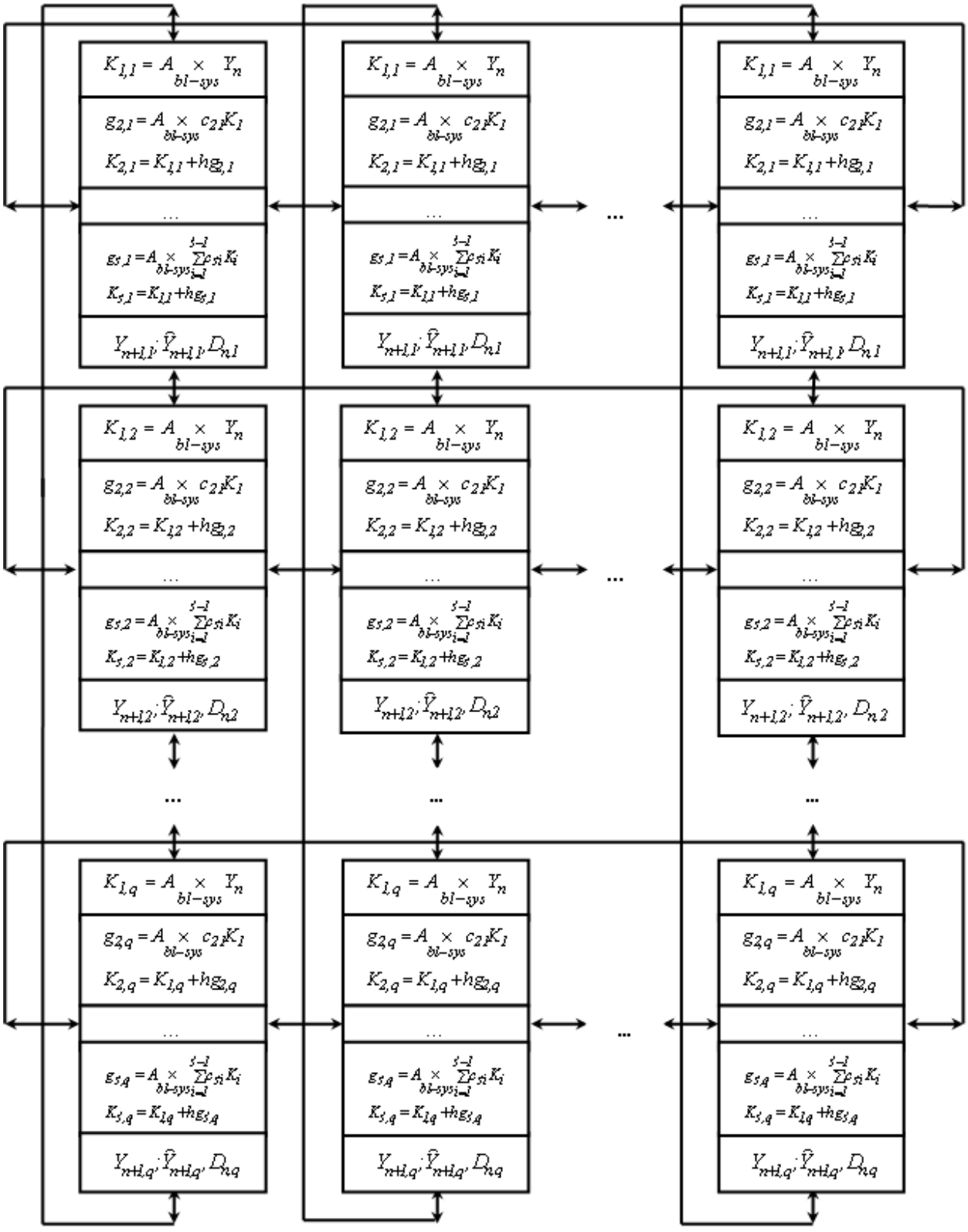


Рис. 4.7. Обчислювальна схема паралельного алгоритму розв'язання СЛЗДР на базі вкладеного ЯМРК (стандартний метод)

Спочатку обчислюються стадійні коефіцієнти: $\bar{k}_1, \bar{k}_2, ..., \bar{k}_{s'}$ із використанням алгоритму блокового систолічного множення матриці на вектор, потім обидві апроксимації розв'язку суміжних порядків точності $r(r')$ і d_n. Для виключення додаткових передач даних при обчисленні обох апроксимацій і величини d_n підтримується блокове розбиття цих величин на підвектори: $\bar{y}_{n+1} = (Y_{n+1,1},....,Y_{n+1,q})$, $\hat{\bar{y}}_{n+1} = (\hat{Y}_{n+1,1},....,\hat{Y}_{n+1,q})$, $D_n = (D_{n,1},....,D_{n,q})$.

Час арифметичних операцій і операцій обміну для алгоритму ВЯМРК визначається з використанням даних таблиць (4.1)-(4.2):

$$T^{22}_{p,comp} = \left[(2s+2)\frac{m^2}{p^2} + \left(s^2 + 4s + 7 + \lceil log_2 p \rceil\right)\frac{m}{p} \right] \cdot t_{op}, \qquad (4.26)$$

$$T^{22}_{p,comm} = (s+1)\cdot \left(2^{\lceil log_2 p \rceil} + p - 2\right)\cdot \left(t_s + \frac{m}{p} t_w \right). \qquad (4.27)$$

Отримаємо асимптотичну складність обох обчислювальних схем на основі вкладених форм (рис. 4.6-4.7). Обмежимось обліком найбільш ресурсоємної операції множення матриці на вектор, тоді, $T^{22}_p / T^{21}_p \approx (s+c)/2,\ c = 1,2...,$ тобто стандартна схема володіє більшою обчислювальною складністю, ніж експоненціальна. Порівняння реальних динамічних характеристик цих обчислювальних схем також дозволяє зробити висновки про переваги запропонованого експоненціального підходу (рис. 4.8-4.9).

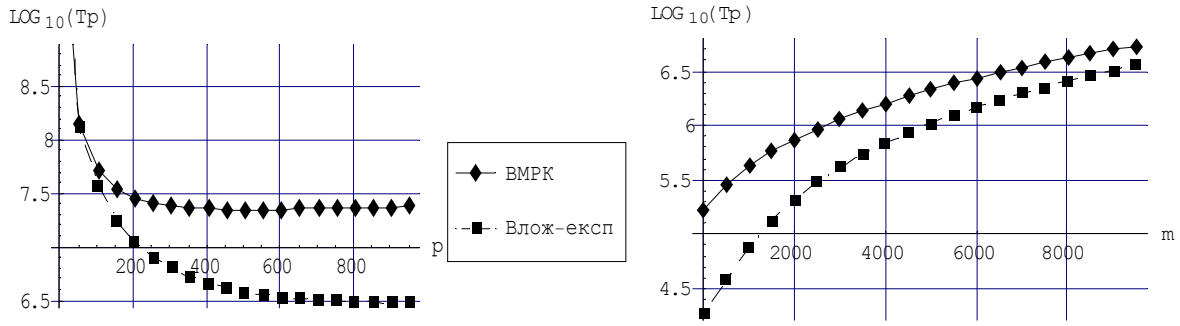


Рис. 4.8. Час реалізації паралельних вкладених методів на основі стандартної, ВМРК та експоненційної схем

Як результат, час реалізації паралельного алгоритму для експоненційної схеми менший, ніж для стандартної, традиційно зростає із зростанням розміру задачі: $m\uparrow \Rightarrow \left(T_p^{21}\uparrow\right) < \left(T_p^{22}\uparrow\right)$ і зменшується із зростанням кількості процесорів: $p\uparrow \Rightarrow \left(T_p^{21}\downarrow\right) < \left(T_p^{22}\downarrow\right)$. Доля обмінних операцій для традиційної схеми перевищує долю обмінів для схеми з експонентою (рис. 4.9).

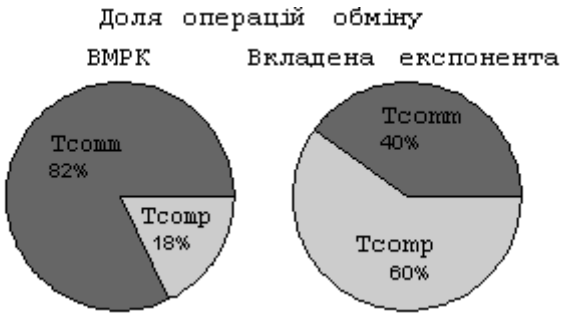


Рис. 4.9. Доля операцій обміну для паралельних вкладених методів розв’язання СЛЗДР на основі стандартної і експоненційної схем

Таким чином, вкладені методи на основі експоненти мають меншу тимчасову складність у порівнянні з традиційними схемами обчислень лінійної задачі для будь-яких розмірів задач і процесорних полів.

4.3. Реалізація технології локальної екстраполяції при паралельному чисельному розв’язанні лінійної задачі Коші

Прискорення обчислень за технологією локальної екстраполяції при вирішенні лінійної однорідної задачі Коші з постійними коефіцієнтами також може бути отримано на основі експоненційного методу. За результатами, глави 2, мінімізувати обчислювальні витрати можна реалізацією h^2-екстраполяції для симетричних опорних методів малого порядку точності. Використання у якості форми F часткової суми з перших

r_0 членів розкладання матричної експоненти в ряд Тейлора відповідає вживанню певного методу Рунге-Кутти r_0 - го порядку точності:

$$\bar{y}_{n+1} = \left(E + hA + \ldots + \frac{h^{r_0} A^{r_0}}{r_0!} \right) \cdot \bar{y}_n = F_{r_0}(hA) \cdot \bar{y}_n . \qquad (4.28)$$

Якщо h - базовий крок інтегрування, то $T_{11} = F_{r_0}(hA) \cdot \bar{y}_n$ - є апроксимацією вирішення $\bar{y}(x_n + h)$ порядку $O(h^{r_0})$. При визначенні розв'язку на кроці $n+1$ заданого порядку точності r необхідно провести обчислення для кроків $h/n_2, \ldots, h/n_k$ і отримати відповідні апроксимації розв'язку $T_{21}, \ldots, T_{k1}$: $T_{21} = F_{r_0}^{n_2}(hA/n_2) \cdot \bar{y}_n, \ldots, T_{k1} = F_{r_0}^{n_k}(hA/n_k) \cdot \bar{y}_n$. Потім, використовуючи співвідношення для поліноміальної екстраполяції (1.21), треба обчислити наближення T_{kk} порядку $O(h^r)$ і $T_{k-1,k}$ порядку $O(h^{r-1})$. Необхідна кількість рядків екстраполяційної таблиці дорівнює $k = r/2$ при симетричному опорному методі й парній послідовності P_i.

Час обчислення за послідовною схемою для симетричного опорного методу порядку точності r_0 дорівнює:

$$\begin{cases} T_{1,0}^{r_0,exp} = (r_0 - 1) \cdot T_1^{A \times A} + (r_0 - 1) \cdot T_1^{c \cdot A} = (r_0 - 1)\left(2m^3 + m^2\right) \cdot t_{op}, \\ T_1^{r_0,exp} = n_k \cdot T_1^{A \times Y} + r_0 \cdot T_1^{c \cdot A} + r_0 \cdot T_1^{A+A} = 2m^2\left(N(k) + kr_0\right) \cdot t_{op}. \end{cases} \qquad (4.29)$$

При визначенні часу послідовної реалізації технології локальної екстраполяції з опорним методом на базі матричної експоненти скористаємося (2.25), даними таблиці (4.1) і співвідношеннями:

$$T_1^{31} = T_{T_{11}} + T_{T_{21}} + \ldots + T_{T_{k1}} + T_1^{ext-tab}, \; T_1^{31} = \sum_{i=1}^{k} n_i \cdot T_1^{A \times Y} + kr_0 \cdot T_1^{c \cdot A} + kr_0 \cdot T_1^{A+A} + T_1^{ext-tab}.$$

Для числової послідовності P_2 і симетричного опорного методу другого порядку маємо: $N(k) = k^2 - k + 1$. Тоді час обчислень за схемою

опорного методу, $T_1^{r_0,exp}$ і спільний час обчислень за технологією ЛЕР з врахуванням визначення $r/2$ рядків екстраполяційної таблиці T_1^{31} дорівнює:

$$T_1^{r_0,exp} = \frac{m^2}{2}\left(r^2 + 6r + 4\right)\cdot t_{op}, \tag{4.30}$$

$$T_1^{31} = \left[\frac{m^2}{2}\left(r^2 + 6r + 4\right) + \frac{5}{8}m\left(r^2 - 2r\right)\right]\cdot t_{op}. \tag{4.31}$$

При розробці паралельного алгоритму на базі експоненційного методу з екстраполяцією за Річардсоном (рис. 4.10) необхідно виділити дві макрооперації першого рівня. Перша задача: обчислення k апроксимацій розв'язку $T_{11},T_{21},...,T_{k1}$ за опорним методом із кроками $h_1,h_2,...,h_k$ і друга: визначення рядків екстраполяційної таблиці за схемою Ейткена-Невілла та обчислення результуючих апроксимацій розв'язку T_{k-1k},T_{kk}.

Паралельне виконання першої макрооперації за аналогією з іншими має підготовчий етап. Час обчислень і час обмінів для нього дорівнюють:

$$\begin{cases} T_{p,0,comp}^{r_0,exp} = (r_0 - 1)\cdot\left(2m^3/p^2\right)\cdot t_{op}, \\ T_{p,0,comm}^{r_0,exp} = 2(r_0 - 1)(p-1)\cdot[t_s + (m^2/p^2)\cdot t_w]. \end{cases} \tag{4.32}$$

Далі обчислюються матричні форми $F_{n_i}^{r_0}, i = \overline{1,k}$ і всі апроксимації розв'язку.. Перша макрооперація реалізується через відомі операції лінійної алгебри плюс паралельне обчислення екстраполяційної таблиці:

$$T_p^{31} = \sum_{i=1}^{k} n_i \cdot T_p^{A\times Y} + kr_0 \cdot T_p^{c\cdot A} + kr_0 \cdot T_p^{A+A} + T_p^{ext-tab}. \tag{4.33}$$

Розбиття даних по процесорах відповідає топології 2D-тор і блоковому систолічному алгоритму виконання матричного добутку, отже, час обчислень в t_{op} дорівнює:

$$T_{p,comp}^{31} = \left[\frac{m^2}{p^2}\left(\frac{r^2 + 6r + 4}{2}\right) + \frac{m}{p}\left(\frac{r^2 - 2r + 4}{4}\lceil log_2\, p\rceil + \frac{5}{8}\left(r^2 - 2r\right)\right)\right]\cdot t_{op}. \tag{4.34}$$

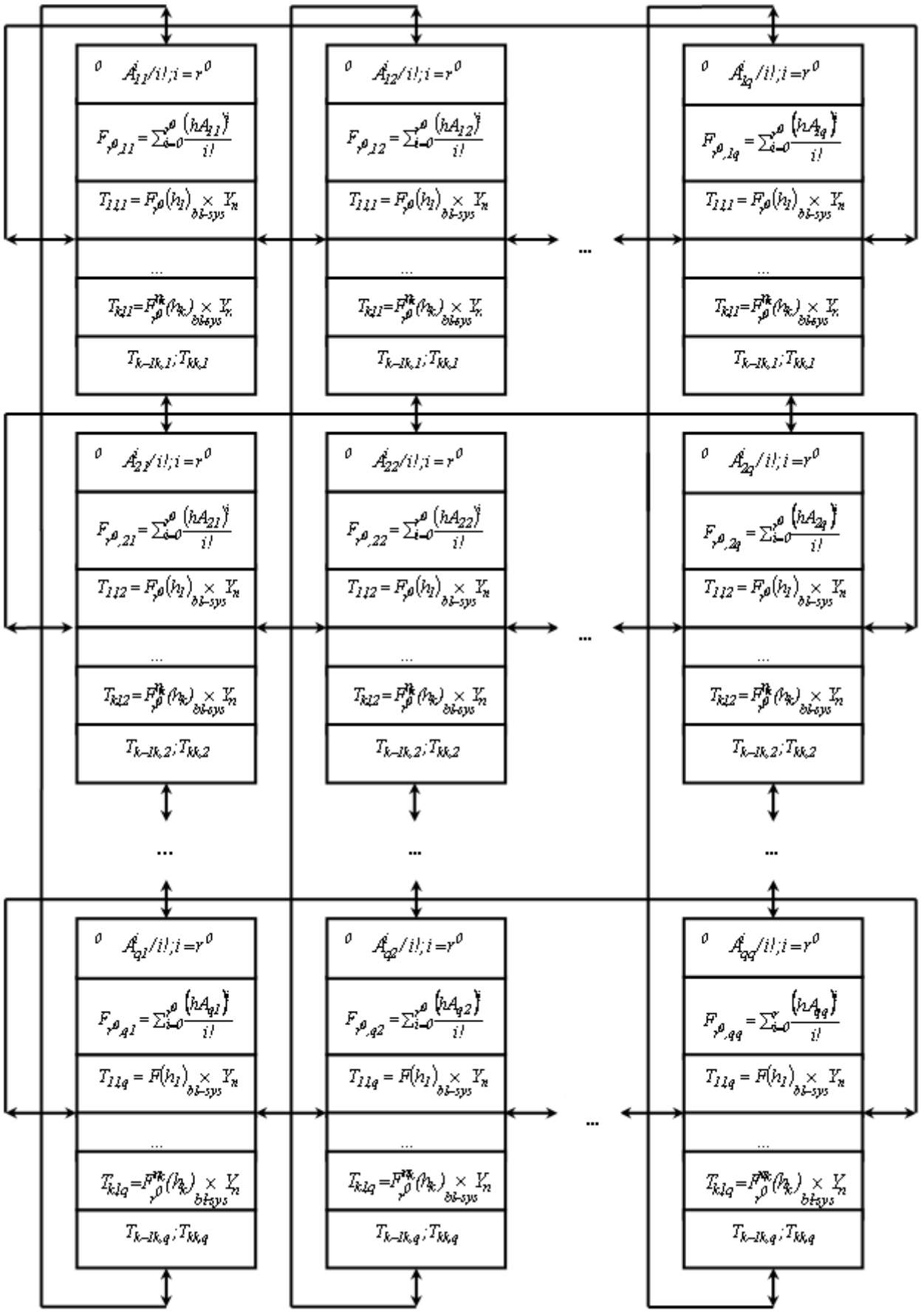


Рис. 4.10. Обчислювальна схема паралельного алгоритму розв’язання СЛЗДР з локальною екстраполяцією на базі експоненційного методу

Комунікаційна складова описаного алгоритму визначається лише обміном даними при обчисленні k апроксимацій розв'язку на основі опорного методу. Обчислення елементів екстраполяційної таблиці при вибраному засобі розпаралелювання не вимагає міжпроцесорного обміну й перегруповування даних для наступного кроку інтегрування. Загальний час на реалізацію операцій передач даних дорівнює:

$$T_{p,comm}^{31} = [(r^2 - 2r + 4)/4] \cdot \left(2^{\lceil \log_2 p \rceil} + p - 2\right) \cdot \left[t_s + (m/p)\frac{m}{p} t_w \right]. \qquad (4.35)$$

Розроблений паралельний алгоритм вирішення лінійної задачі Коші з оцінкою покрокової похибки за технологією локальної екстраполяції (рис. 4.10) порівняємо зі стандартною схемою обчислень. Для цього у якості опорного методу приймається явний однокроковий s_0-стадійний метод порядку $O\left(h^{r_0}\right)$:

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h \cdot \sum\limits_{i=1}^{s_0} b_i \cdot \bar{k}_i, \\ \bar{k}_l = A \cdot (\bar{y}_n + h \cdot \sum\limits_{i=1}^{l-1} c_{li} \cdot \bar{k}_i), l = \overline{1, s_0}. \end{cases} \qquad (4.36)$$

Час паралельної реалізації екстраполяції Річардсона при опорному методі (4.36) в базових операціях лінійної алгебри складає:

$$T_p^{32} = \sum_{i=1}^{k} n_i \cdot \left(s_0 \cdot T_p^{A \times Y} + \frac{s_0^2 + s_0}{2} \cdot T_p^{c \cdot Y} + \frac{s_0(s_0 - 1)}{2} \cdot T_p^{Y+Y} \right) + T_p^{ext-tab}. \qquad (4.37)$$

Для порівняння обчислювальної складності двох паралельних алгоритмів вирішення СЛЗДР на базі локальної екстраполяції обмежимося обліком найбільш ресурсоємної операції множення матриці на вектор. Стандартна схема вирішення володіє обчислювальною складністю в s_0 разів більшою, ніж експоненційна, через справедливість співвідношення:

$$T_p^{32} / T_p^{31} \approx \frac{s_0 N(k)}{N(k)} = s_0. \qquad (4.38)$$

Для ЯМРК високих порядків приведене співвідношення збільшується на величину, рівну різниці між порядком і кількістю стадій методу.

Аналогічна залежність (рис. 4.11) існує і при обліку повного часу виконання стандартної і експоненційної схем інтегрування СЛЗДР.

Рис. 4.11. Час реалізації паралельних алгоритмів із локальною екстраполяцією для експоненційної T_p^{31} та стандартної T_p^{32} схем

Таким чином, як і при інших засобах оцінки апостеріорної локальної похибки, вживання експоненційного методу інтегрування для технології локальної екстраполяції Річардсона, зменшує обчислювальні і комунікаційні витрати.

Експоненційний метод аналогічно може бути застосований для прискорення вирішення неоднорідних СЛЗДР із постійними коефіцієнтами:

$$\begin{cases} \vec{y}'(x) = A \cdot \vec{y}(x) + b, \\ \vec{y}(x_0) = \vec{y}_0, \ A,b = const. \end{cases} \tag{4.39}$$

Чисельне розв’язання задачі Коші для неоднорідних (4.39) має вигляд:

$$\vec{y}_{n+1} = \vec{y}_n + R_r(hA) \cdot (A\vec{y}_n + b), \tag{4.40}$$

де $R_r(hA)$ – матрична функція, яка також може бути апроксимована відрізком ряду Тейлора: $R_r(hA) = \sum_{i=1}^{r} (A^{i-1}h^i / i!)$. Розробка паралельних алгоритмів із вбудованими способами оцінки локальної похибки, теоретичний аналіз складності й проведений чисельний експеримент дозволяють зробити висновки, що вживання експоненційного методу для розв’язання неоднорідних лінійних задач Коші, як і для однорідних, дозволяє зменшити витрати на паралелізм у порівнянні зі стандартними схемами розв’язання СЛЗДР.

4.4. Розв'язання лінійної задачі Коші для СЗДР на основі методу швидкого множення матриць

Матричне множення або матричний добуток (МД) – домінуюча обчислювальна частина експоненційних методів розв'язання лінійних СЗДР. Тому прискорення паралельної реалізації цієї базової операції лінійної алгебри, означає зменшення терміну виконання методу вирішення лінійної задачі у цілому. При величезній різноманітності методів обчислення матричного добутку [120-122] для щільнозаповнених матриць є два принципово різних класи послідовних алгоритмів: традиційні та рекурсивні методи швидкого множення Штрассена [123-125]. У оригіналі алгоритм Штрассена – це алгоритм множення блокових матриць половинного розміру, де кожен блок квадратний, тобто розміри матриць є парними числами. Метод Штрассена-Винограда [123] приведено на рис. 4.12.

$$
\begin{array}{lll}
S_1 = A_{21} + A_{22}, & M_1 = S_2 S_6, & T_1 = M_1 + M_3, \\
S_2 = S_1 - A_{11}, & M_2 = A_{11} B_{11}, & T_2 = T_1 + M_4, \\
S_3 = A_{21} - A_{12}, & M_3 = A_{12} B_{21}, & T_3 = M_5 + M_6, \\
S_4 = A_{12} - S_2, & M_4 = S_3 S_7, & C_{11} = M_2 + M_3, \\
S_5 = B_{12} - B_{11}, & M_5 = S_1 S_5, & C_{12} = T_1 + T_3, \\
S_6 = B_{22} - S_5, & M_6 = S_4 B_{22}, & C_{21} = T_2 - M_7, \\
S_7 = B_{22} - B_{12}, & M_7 = A_{22} S_8, & C_{22} = T_2 + M_5, \\
S_8 = S_6 - B_{12}, & &
\end{array}
$$

$$
A = \left\langle \frac{A_{11}}{A_{21}} \middle| \frac{A_{12}}{A_{22}} \right\rangle, \quad B = \left\langle \frac{B_{11}}{B_{21}} \middle| \frac{B_{12}}{B_{22}} \right\rangle, \quad C = \left\langle \frac{C_{11}}{C_{21}} \middle| \frac{C_{12}}{C_{22}} \right\rangle.
$$

Рис. 4.12. Метод швидкого множення матриць Штрассена-Винограда

Ідея Штрассена може бути застосована рекурсивно для знаходження добутків блоків матриць $M_i, i = \overline{1,7}$. Якщо початкові матриці A і B порядку m, то алгоритм швидкого множення можна використовувати багато разів, отримуючи на самому нижньому рівні рекурсії блоки $k \times k = 1 \times 1$.

Проте немає необхідності опускатися вниз до рівня блоків одиничного порядку. При досить малих розмірах блоку ($k \le k_{min}$) може виявитися корисним обчислення блоків із використанням стандартного алгоритму МД. На першому кроці алгоритм передбачає 7 звернень до самого себе з матрицями порядку $m/2$ та 15 операцій типу складання матриць того ж порядку. Далі йде розгортка рекурсії до досягнення мінімального розміру блоку й множення блоків за традиційним алгоритмом матричного множення.

Обчислювальна складність запропонованої схеми алгоритму швидкого множення визначається функцією розміру початкових матриць і мінімального порядку перемножуваних блоків: m, k_{min}. Нехай при виконанні матричного множення алгоритм Штрассена-Винограда рекурсивно викликався d раз, тоді порядок блоків матриць, що умножаються, дорівнює $k_{min} = m/2^d$. Час реалізації послідовного алгоритму при цьому складає:

$$T_1^{Str}(m) = \left[2\left(\frac{7}{8}\right)^d m^3 + \frac{15}{4} m^2 \left(1 + \frac{7}{4} + \frac{7^2}{4^2} + \ldots + \frac{7^{d-1}}{4^{d-1}} \right) \right] \cdot t_{op}. \qquad (4.41)$$

Відомий паралельний алгоритм класичного методу Штрассена був застосований на *Intel Paragon* і показав добрі результати в порівнянні з традиційними алгоритмами МД [126-127]. Істотним недоліком цього алгоритму є відсутність масштабованості, оскільки необхідна кількість процесорів для нього кратне сімці (по кількості множень блоків матриць $M_i, i = \overline{1,7}$). Це обмеження є достатньо жорстким і не природним для більшості паралельних архітектур.

Для подолання вказаного недоліку скористаємося поліалгоритмічним підходом. Під поліалгоритмом мається на увазі комбінація двох або більше алгоритмів в одну обчислювальну схему, що реалізує поставлену задачу з метою скорочення обчислювальних і/або ємкісних витрат [14]. Алгоритм швидкого множення рекурсивний, тому є можливість побудувати поліалгоритм з деякого традиційного алгоритму множення матриць на верхньому рівні рекурсії і методу Штрассена-Винограда на нижньому рівні, і

навпаки. Алгоритм Штрассена є блоковим, тому природно комбінувати його з алгоритмом матричного добутку, що використовує відповідне розбиття даних. Побудуємо поліалгоритм із блокового систолічного алгоритму матричного множення між процесорами і серії вживань рекурсивного методу Штрассена на кожному процесорі (рис. 4.13).

Нехай є замкнута двовимірна сітка процесорів розмірності $p \times p$, початкові матриці A і B, що розподілені на блоки $k = m / p$, кількість блоків дорівнює $q^2 = p^2$. Блоки початкових матриць і результату з координатами $< i, j >$ зберігаються у відповідному процесорі з тими ж координатами. Як і раніше, передбачаємо, що $m \ mod \ p = 0$, інакше використовується доповнення нульовими елементами. Спочатку, за обчислювальною схемою блокового систолічного множення, виконується косе зрушення вліво по рядках для блоків матриці A і косе зрушення вгору по стовпцях для блоків матриці B. На кожному з p кроків алгоритму проводиться множення блоків матриць A і B, що зберігаються в процесорі з номером, і складання зі вже обчисленим значенням блоку матриці C_{ij}, розташованим на цьому ж процесорі. Для першого кроку це значення дорівнює нульовому блоку. Потім проводиться одиночне зрушення вліво по рядках паралельно для всіх блоків матриці $A: A_{ij} \leftarrow A_{i,j+1}$ і одиночне зрушення вгору по стовпцях також паралельно для всіх блоків матриці $B: B_{ij} \leftarrow B_{i+1,j}$. Множення блоків матриць виконується усередині одного процесора за рекурсивним алгоритмом, що дозволяє уникнути додаткових пересилок даних. На нижньому рівні рекурсії застосовується стандартний алгоритм множення матриць, глибина рекурсії дорівнює: d. Розроблений алгоритм є таким, що масштабується для будь-якого числа процесорів і будь-якого порядку матриць, що умножаються. Час реалізації блокового рекурсивно-систолічного алгоритму включає час виконання арифметичних і обмінних операцій:

$$T_p^{BSys-Str} = T_{p,comp}^{BSys-Str} + T_{p,comm}^{BSys-Str}.$$

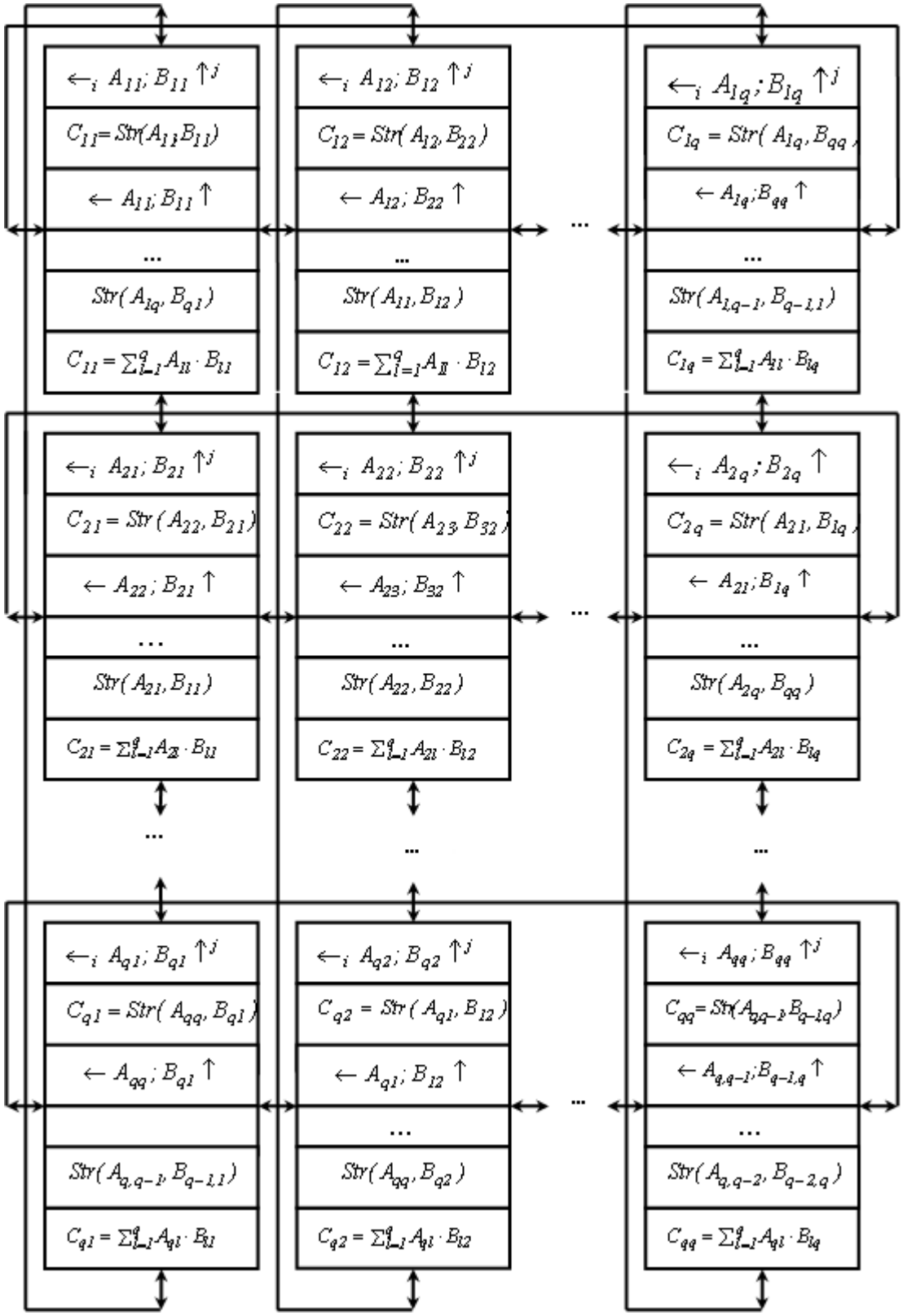


Рис. 4.13. Обчислювальна схема комбінованого алгоритму множення матриць: блокове систолічне множення + алгоритм Штрассена-Винограда для мультикомп’ютера топології 2D-тор

При цьому час виконання обчислень за схемою рис. 4.13 дорівнює:

$$T_{p,comp}^{BSys-Str} = q \cdot \left(T_{A_{ij} \underset{Str}{\times} B_{ij}} + T_{A_{ij}+B_{ij}} \right), \tag{4.42}$$

де $T_{A_{ij} \underset{Str}{\times} B_{ij}}$ – час множення блоків матриць порядку $k = m/p$ за рекурсивним алгоритмом; $T_{A_{ij}+B_{ij}} = k^2 \cdot t_{ad} = \left(m^2/p^2\right) \cdot t_{ad}$ – час складання блоків матриць того ж розміру. Час виконання множення блоків, що виконується за алгоритмом Штрассена-Винограда, задовольняє рекурентному співвідношенню:

$$T_{A_{ij} \underset{Str}{\times} B_{ij}} = T_p^{Str}(k) = 7T_p^{Str}\left(\frac{k}{2}\right) + 15\left(\frac{k}{2}\right)^2 \cdot t_{ad}. \tag{4.43}$$

Час реалізації алгоритму для матриць порядку $k/2,...,k/2^{d-1}$:

$$T_p^{Str}\left(\frac{k}{2}\right) = 7T_p^{Str}\left(\frac{k}{4}\right) + 15\left(\frac{k}{4}\right)^2 t_{ad}, \cdots, T_p^{Str}\left(\frac{k}{2^{d-1}}\right) = 7T_p^{Str}\left(\frac{k}{2^d}\right) + 15\left(\frac{k}{2^d}\right)^2 t_{ad}.$$

Виконавши підстановки та елементарні перетворення, отримаємо:

$$T_p^{Str}(k) = 7^d T_p^{Str}\left(\frac{k}{2^d}\right) + \frac{15}{4}k^2\left(1 + \frac{7}{4} + \frac{7^2}{4^2} + \ldots + \frac{7^{d-1}}{4^{d-1}}\right)t_{ad}. \tag{4.44}$$

До цих пір була визначена обчислювальна складність розгортки рекурсії. Розглянемо внутрішній рівень рекурсії, оскільки саме там виконуються операції множення блоків матриць мінімального порядку за стандартним методом матричного множення:

$$T_p^{Str}\left(\frac{k}{2^d}\right) = k_{min}^3 \cdot \left(t_{mul} + t_{ad}\right) = 2\left(\frac{k}{2^d}\right)^3 \cdot t_{op}. \tag{4.45}$$

Таким чином, час реалізації швидкого рекурсивного множення блоків матриць для мультикомп'ютера складає:

$$T_p^{Str}(k) = \left[2\left(\frac{7}{8}\right)^d k^3 + \frac{15}{4}k^2\left(\frac{1 - \frac{7^d}{4^d}}{1 - \frac{7}{4}}\right)\right] \cdot t_{op}. \tag{4.46}$$

Загальний час виконання арифметичних операцій для комбінації блокового систолічного алгоритму і рекурсивного матричного множення дорівнює:

$$T_{p,comp}^{BSys-Str} = \left[2\left(\frac{7}{8}\right)^d \frac{m^3}{p^2} + \left(5\left(\frac{7}{4}\right)^d - 4 \right) \frac{m^2}{p} \right] \cdot t_{op}. \qquad (4.47)$$

Час обмінних операцій для описаної схеми (рис. 4.13) визначається, як і для блокового систолічного алгоритму:

$$T_{p,comm}^{BSys-Str} = 4(p-1) \cdot \left(t_s + \frac{m^2}{p^2} \cdot t_w \right). \qquad (4.48)$$

Очевидно, що динамічні характеристики паралельних алгоритмів матричного добутку залежать від співвідношення між кількістю процесорів та порядком матриць. Для рекурсивного алгоритму множення матриць істотним параметром є також величина глибини рекурсії, d (рис. 4.14-4.17).

```
(* Визначення оптимального значення глибини рекурсії *)

Round[Simplify[Log[2, 2*m*(Log[8/7])] - Log[2, (5*Log[7/4])]]]

d_opt = Round[(Log[2 m Log[8/7]] - Log[5 Log[7/4]]) / Log[2]]

m_opt = N[m / 2^d_opt] = 8
```

Рис. 4.14. Визначення оптимального значення глибини рекурсії для рекурсивного алгоритму МД

Визначення оптимальної величини глибини рекурсії, а, отже, й розміру мінімального оброблюваного блоку матриць проводилося у пакеті *Mathematica*. Для цього необхідно знайти значення, що доставляє мінімум функції T_1^{Str}, причому діапазон зміни глибини рекурсії обмежений наступною нерівністю: $k_{min} = \left(m / 2^d\right) \geq 1 \Rightarrow\ m \geq 2^d \Rightarrow\ d \leq log_2 m$. Очевидно, що при великих розмірах СЛЗДР кожна з функцій $V(d) = T_1^{Str} / t_{op}$ рисунку 4.15 має явно виражений локальний мінімум, відношення ж розміру системи до глибини рекурсії при цьому залишається постійним. Наприклад, при $m = 1024$, $d_{min} = 7$ і $m / 2^{d_{min}} = 1024/128 = 8$, а при $m = 256$ маємо

$d_{min} = 5 \Rightarrow m / 2^{d_{min}} = 8$. У той же час при $m = 16$ мінімальна глибина рекурсії дорівнює 1, тобто для досягнення мінімуму обчислювальних витрат необхідний всього один крок розгортки рекурсії, а для $m = 8$ розгортки рекурсії немає, оскільки $d_{min} = 0$.

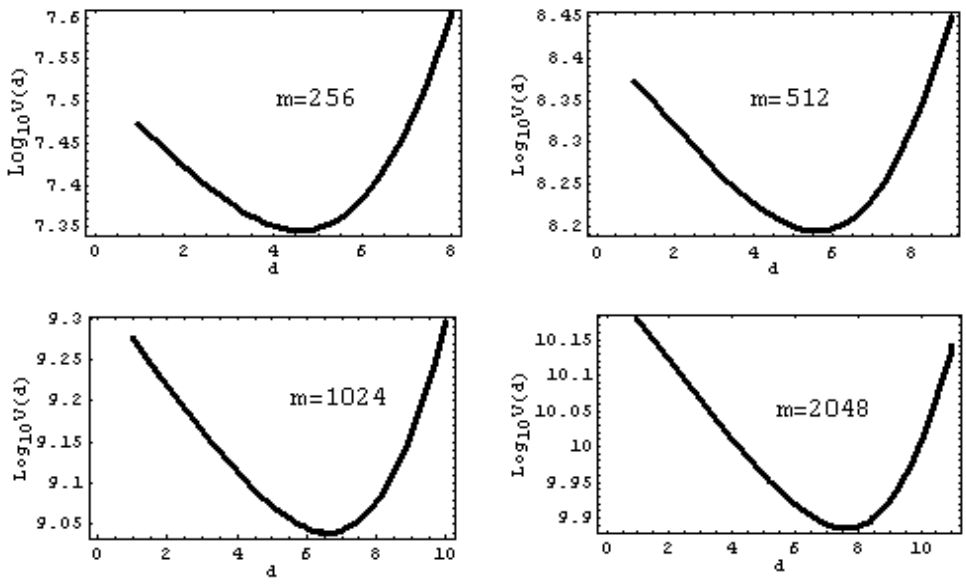


Рис. 4.15. Обчислювальна складність рекурсивно-систолічного алгоритму від глибини рекурсії у логарифмічному масштабі

Аналітичні результати підтверджуються результатами експериментів, на рис. 4.16 приведено залежність часу виконання матричного добутку від величини блоку матриць, що перемножуються.

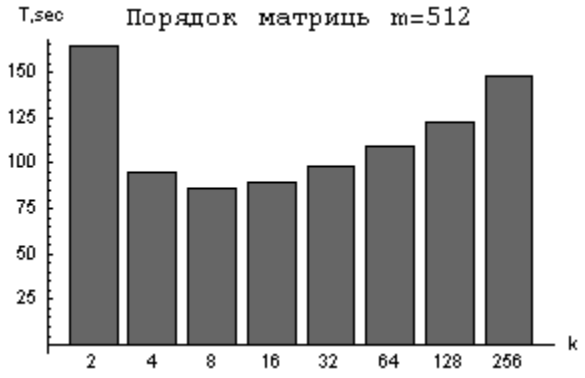


Рис. 4.16. Експериментальне визначення величини мінімального блоку для рекурсивно-систолічного алгоритму множення матриць

Для паралельного варіанту запропонованого алгоритму блокового рекурсивно-систолічного множення окрім глибини рекурсії необхідно враховувати співвідношення між кількістю процесорів p і порядком перемножуваних матриць m (рис. 4.17). Обмеження на діапазон зміни глибини рекурсії визначаються наступною нерівністю:

$$k_{min} = \frac{m}{2^d p} \geq 1 \Rightarrow \frac{m}{p} \geq 2^d \Rightarrow d \leq log_2 \frac{m}{p}.$$

При оцінюванні глибини рекурсії тимчасові характеристики приведені до часу реалізації однієї операції з рухомою точкою, для різних мультикомп'ютерів величина оптимальної глибини рекурсії має бути поправлена з урахуванням цієї машинно-залежної константи.

```
(* Визначення оптимального значення глибини рекурсії *)

pd_opt = Round[Simplify[Log[2, 2 * m / p * (Log[8 / 7])] - Log[2, (5 * Log[7 / 4])]]]

pd_opt = Round[(Log[(2 m Log[8/7])/p] - Log[5 Log[7/4]])/Log[2]]

(m / p)_opt = 8 * 2^pd_opt
```

Рис. 4.17. Визначення величини оптимальної глибини рекурсії для паралельного блокового рекурсивно-систолічного алгоритму

Аналіз аналітичних виразів, що характеризують динамічні характеристики якості паралельних алгоритмів, а також проведений чисельний експеримент, дозволяють зробити наступні висновки:

1) паралельний блоковий метод рекурсивно-систолічного множення матриць володіє меншою асимптотичною $T_p^{bl\text{-}sys} / T_p^{Bsys\text{-}Str} \approx (8/7)^d$ і реальною, підтверджуваною експериментальним шляхом, тимчасовою складністю в порівнянні з найбільш відомими традиційними паралельними алгоритмами множення матриць [12-14] (рис. 4.18);

2) динамічні характеристики запропонованого алгоритму на основі швидкого множення перевершують відповідні характеристики стандартних

аналогів, особливо для матриць великих порядків, при цьому коефіцієнти прискорення й ефективності зростають із зростанням розміру задачі: $m\uparrow \Rightarrow\ S\uparrow \Rightarrow\ E\uparrow$ і зменшуються зі зростанням кількості процесорів $p\uparrow \Rightarrow\ S\downarrow \Rightarrow\ E\downarrow$ (рис. 4.19).

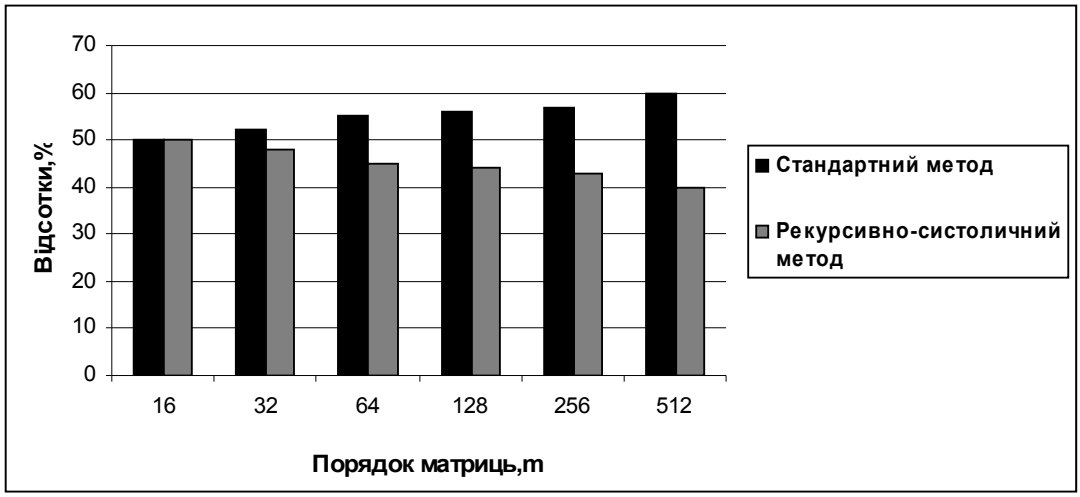


Рис. 4.18. Співвідношення часу реалізації паралельних методів стандартного і рекурсивно-систолічного матричного добутку

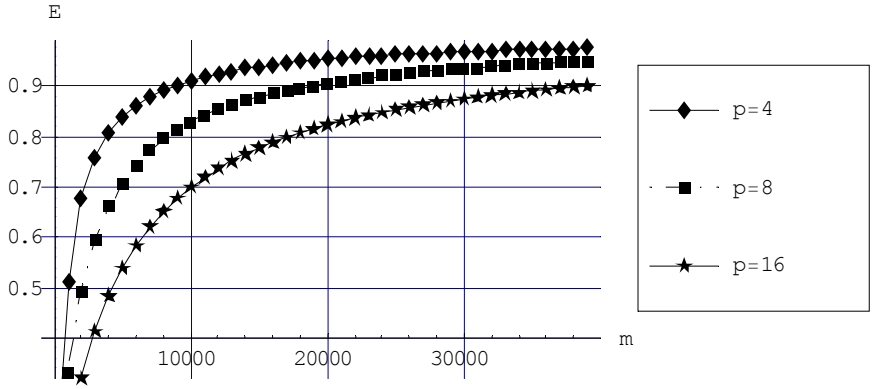


Рис. 4.19. Коефіцієнти ефективності паралельного блокового рекурсивно-систолічного методу для мультикомп’ютерів із шириной процесорної сітки, рівною p

Крім того, рекурсивний характер розробленого алгоритму дозволяє звести багатовимірну початкову задачу до вирішення підзадач меншої розмірності, і, за рахунок використання швидкої пам'яті комп'ютера, досягти

прискорення операції матричного добутку. До недоліків цього підходу слід віднести декілька гіршу чисельну стійкість, хоча й достатню для більшості практичних задач [126-127].

Друга модифікація систолічного алгоритму з блоковим розбиттям даних виконує послідовну реалізацію систолічного множення для блоків матриць. Використання блокового розбиття дозволяє на сітках розмірністю $p \times p$ перемножувати матриці порядку ip, де $i = 1,2,...$ Перемножуванні квадратні матриці розмірності $m \times m$ розбиваються на блоки, кратні ширині матриці процесорного поля: $k = ip$. Тоді, $m = q \cdot k$, де k – це розмір ширини блоку, а $q^2 = [m / ip]^2$ – кількість блоків.

Потім для кожного блоку знаходиться результат за допомогою систолічного алгоритму. При здобутті одного блоку матриці результату C_{ij} необхідно q разів застосувати звичайний алгоритм.

Час обчислень для систолічного алгоритму №2 із блоковим розбиттям визначається, як:

$$T_{p,comp}^{A\times B,bl-sys,2} = q^3 i^2 (2ip + 1) = \frac{m^3}{ip^3}(2ip + 1). \quad (4.49)$$

Комунікаційна складова цього алгоритму дорівнює:

$$T_{p,comm}^{A\times B,bl-sys,2} = \frac{m^3}{i^3 p^3} \cdot \left(4(p - 1) \cdot \left(t_s + i^2 t_w\right)\right). \quad (4.50)$$

Такий алгоритм множення матриць долає недолік традиційного систолічного алгоритму, що полягає в жорсткому обмеженні на порядок перемножуваних матриць.

4.5. Порівняння ефективності паралельних методів оцінки локальної похибки при розв'язанні лінійних задач

Дослідження ефективності альтернативних засобів оцінки апостеріорної локальної похибки при розв'язанні лінійної однорідної задачі

Коші з постійними коефіцієнтами проводилося на наступній множині методів одного й того ж порядку r:

1 – експоненціальний метод та правило Рунге;

2 – явний метод Рунге-Кутти та правило Рунге;

3 – експоненціальний метод із локальною екстраполяцією;

4 – ЯМРК та локальна екстраполяція Річардсона;

5 – вкладені методи на основі матричної експоненти;

6 – вкладені методи Рунге-Кутти.

Відзначимо, що динамічні характеристики паралелізму істотно залежать від параметрів як початкової задачі, так і паралельної системи, на якій вони реалізовані. Вплив же чисельного методу визначається наявністю внутрішнього паралелізму і порядком різницевої схеми, покладеної в його основу. Тому порівняння ефективності запропонованих паралельних методів є порівнянням не унікальних обчислювальних схем, але двох класів вирішення лінійної задачі Коші.

Проведемо оцінку асимптотичної складності отриманих паралельних алгоритмів на основі аналізу часу їх реалізації на мультикомп'ютері з p^2 процесорів із урахуванням самої трудомісткої операції (табл. 4.3):

а) для методів, заснованих на вживанні матричної експоненти, виконується наступне співвідношення: $T_p^{31} < T_p^{11} < T_p^{21}$; аналогічне співвідношення має місце і для стандартної схеми: $T_p^{32} < T_p^{12} < T_p^{22}$;

б) найбільшою тимчасовою складністю володіють методи локальної екстраполяції і, перш за все, на основі стандартної схеми:

1) $O[sr^2m^2/p^2]$ для обчислень;

2) $O\left[sr^2p\left(t_s + \frac{m}{p}\cdot t_w\right)\right]$ для операцій обміну;

в) кращими показниками володіють алгоритми, засновані на поєднанні експоненти і вкладених формул, так, обчислювальна складність вкладеного

експоненційного методу складає $O[2 \cdot m^2 / p^2]$, для експоненти з правилом Рунге практично ідентична – $O[3 \cdot m^2 / p^2]$, а для ВМРК дорівнює $O[(s+1) \cdot m^2 / p^2]$;

г) усі паралельні алгоритми, що засновані на вживанні експоненційного методу вирішення, мають меншу тимчасову складність у порівнянні зі своїми стандартними аналогами.

Таблиця 4.3

Оцінки асимптотичної тимчасової складності засобів визначення локальної апостеріорної похибки, приведені до часу виконання найбільш ресурсоємної операції

Засоби оцінки локальної апостеріорної похибки	Асимптотична тимчасова складність	
	Експоненційна схема	Стандартна схема
1. Правило Рунге	$3 \cdot T_p^{A \times Y}$	$(3s-1) \cdot T_p^{A \times Y}$
2. Локальна екстраполяція	$\left(\frac{r^2-2 \cdot r+4}{4}\right) \cdot T_p^{A \times Y}$	$\left(\frac{r^2-2 \cdot r+4}{4}\right) \cdot s \cdot T_p^{A \times Y}$
3. Вкладені форми	$2 \cdot T_p^{A \times Y}$	$(s+1) \cdot T_p^{A \times Y}$

Традиційне порівняння якості отриманих паралельних алгоритмів проводиться з використанням коефіцієнтів прискорення і ефективності (рис. 4.20-4.21). Аналіз якості паралельних алгоритмів, стосується як відносних, так і абсолютних коефіцієнтів прискорення та ефективності. В даному випадку це означає, що для кожного способу оцінки локальної похибки використовувався або найменш витратний послідовний алгоритм, або алгоритм, який було распараллелено.

Найкращими динамічними характеристиками володіють вкладений експоненційний метод і експоненційний з правилом Рунге, для них характеристики реального паралелізму практично ідентичні. Для всіх

алгоритмів, що базуються на експоненційніій схемі, абсолютні показники якості паралельних алгоритмів кращі, ніж для стандартних схем.

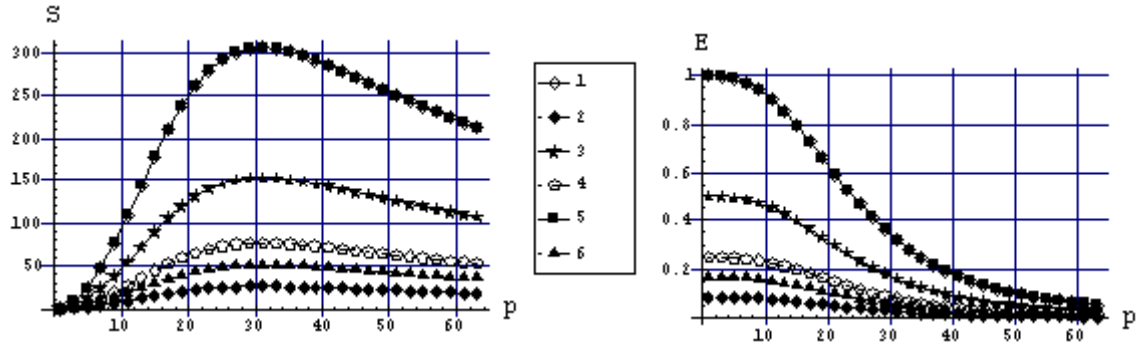


Рис. 4.20. Відносні коефіцієнти прискорення та ефективності паралельних алгоритмів рішення лінійної задачі Коші з контролем локальної похибки від порядку сітки процесорів

Наприклад, при використанні вкладених форм коефіцієнт ефективності експоненційної схеми практично в два рази вище, ніж стандартної (рис. 4.21).

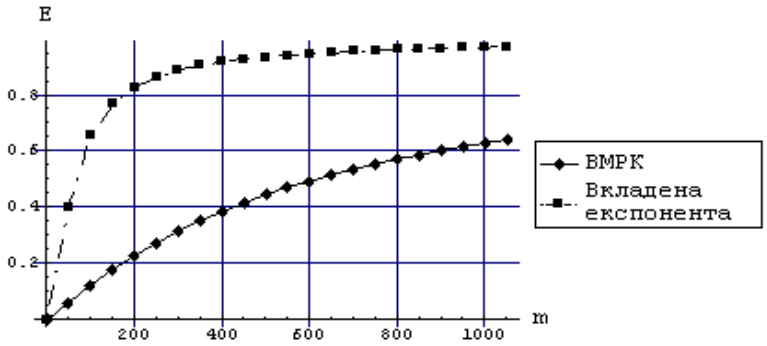


Рис. 4.21. Коефіцієнти ефективності вкладених методів розв’язання лінійної задачі Коші від розміру СЛЗДР

До наступного моменту при побудові паралельних алгоритмів глави 4, як топологічне вирішення, використовувався 2D-тор. При переході на фізичну комутаційну мережу іншої топології необхідно мати гнучкий механізм логічного відображення однієї топології на другу.

Математичним апаратом, здатним вирішити поставлену задачу, є двійкові рефлексивні коди Ґрею [10,128]. Логічне відображення двовимірних замкнутих сіток на гіперкуб виконується за правилом: кожному процесору сітки з координатами (i,j) відповідатиме процесор гіперкуба з номером $G(i,p)||G(j,p)$, де $G(i,p)$ – значення коду Ґрея, i - порядковий номер значення в коді, p - його довжина, || – операція конкатенації. Наприклад, рисунок 4.22 ілюструє відображення сітки 4×4 на гіперкуб розмірності 4.

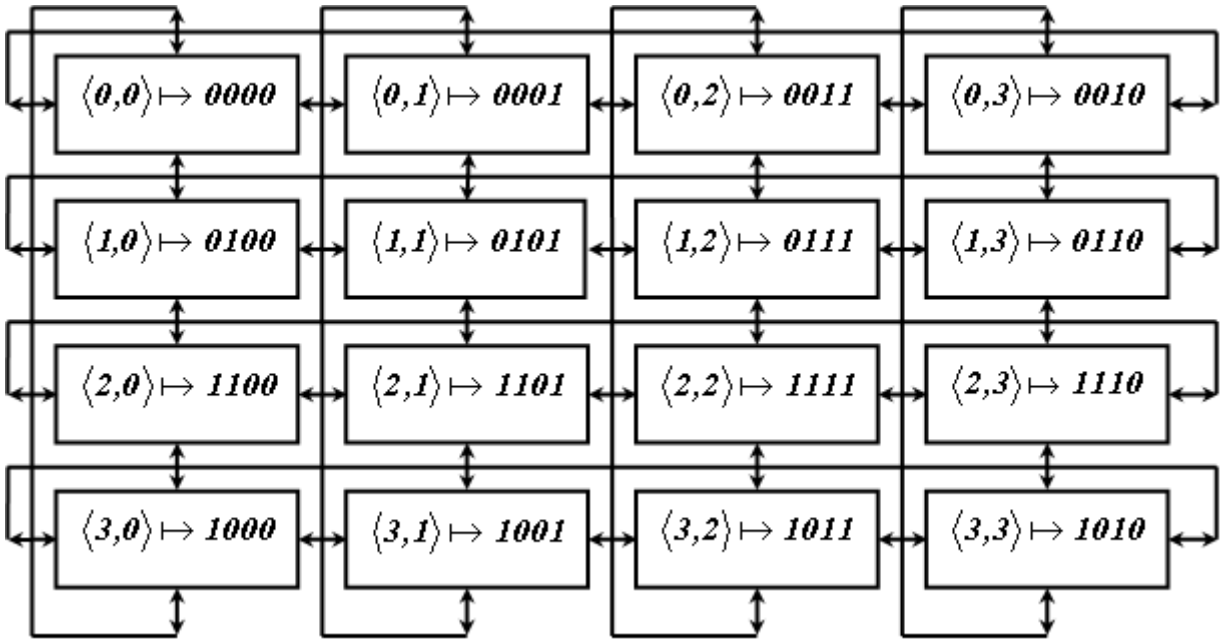


Рис. 4.22. Схема логічного відображення топології 2D-тор розмірності 4×4 на топологію гіперкуб $H(4)$

Відзначимо, що в результаті виконання такого відображення процесори, що є “безпосередніми сусідами” в топології тор, матимуть ті ж властивості у гіперкубі. Таким чином, усі паралельні алгоритми розв’язання лінійної задачі Коші на базі матричної експоненти, розроблені для топології 2D-тор, без зміни алгоритму можуть бути перенесені на гіперкуб. Побудова аналогічного відображення на топологію кільце спричиняє за собою збільшення витрат на комунікаційну складову алгоритмів і має свідомо гірші характеристики паралелізму, ніж у разі топологій тор чи гіперкуб.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Grand Challenges: High performance computing and communications // A report by the Committee on Physical, Mathematical and Engineering Science, NSF/CISE, 1800 G. Street NW, Washington, DC 20550, 2001.
2. Забродин А.В. Параллельные вычислительные технологии. Состояние и перспективы // Материалы первой молодежной школы ”Высокопроизводительные вычисления и их приложения”. – Режим доступа: ftp://parallel.ru/parallel/chg1999.
3. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608с.
4. Воеводин В.В. Математические проблемы параллельных вычислений // Тезисы докладов II Всероссийской научной конференции "Методы и средства обработки информации". 5-7 октября 2005. – Москва: МГУ, 2005. – С. 22-33.
5. Воеводин В.В. Супервычисления и структура алгоритмов // Материалы первой молодежной школы ”Высокопроизводительные вычисления и их приложения”. – Режим доступа: ftp://parallel.ru/parallel/chg1999.
6. Воеводин В.В. Информационная структура алгоритмов. – М.: Изд-во МГУ, 1997. – 139с.
7. Воеводин В.В. Математические основы параллельных вычислений. – М.: Изд-во МГУ, 1991. – 345с.
8. Воеводин В.В. Математические модели и методы в параллельных процессах. – М.: Наука, 1986. – 296с.
9. Voevodin V.V. Information structure of sequental programs // Rus. J. Num. An. and Math. Modeling. 1995. – Vol. 10, № 3. – P. 279-286.
10. Гергель В.П., Стронгин Р.Г. Основы параллельных вычислений для многопроцессорных вычислительных систем.– Н.Новгород: ННГУ, 2001.– 122с.
11. Гергель В.П. Теория и практика параллельных вычислений. – Москва: Бином. Лаборатория знаний, 2007. – 423с.

12. Grama A., Gupta A., Kumar V. Isoefficiency: Measuring the scalability of parallel algorithms and architectures // IEEE Parallel and Distributed technology, 1993. – P. 12-21.
13. Kumar V., Gupta A. Analyzing scalability of parallel algorithms and architectures // Journal of Parallel and Distributed Computing, 22(3), 1994. – P. 379-391(2nd edn., 2003).
14. Gupta A., Kumar V. Scalability of parallel algorithm for matrix multiplication // Technical report TR-91-54, Department of CSU of Minneapolis, 1994.
15. Забродин А.В. СуперЭВМ МВС-100, МВС-1000 и опыт их использования при решении задач механики и физики // Математическое моделирование, 2000, т. 12, № 5. – С. 61-66.
16. Забродин А.В., Левин В.К. Опыт разработки параллельных вычислительных технологий, создание и развитие семейства МВС // Тезисы докладов конференции "Высокопроизводительные вычисления и их приложения". – М.: МГУ, 2000. – С. 3-7.
17. Забродин А.В., Левин В.К., Каратанов В.В и др. Производительность 24-процессорного фрагмента многопроцессорной вычислительной машины МВС-1000М // Тезисы докладов конференции "Высокопроизводительные вычисления и их приложения". – М.:МГУ, 2000. – С. 9-11.
18. Самофалов К.Г., Луцкий Г.М. Структуры и организация функционирования ЭВМ и систем. – К.: Вища школа, 1978. – 392с.
19. Малиновский Б.Н., Боюн В.П., Козлов Л.Г. Вопросы построения высокопроизводительных средств обработки информации. // Управляющие системы и машины. – 1976. №6. – С. 19-22.
20. Малиновский Б.Н., Боюн В.П., Козлов Л.Г. Многопроцессорные вычислительные структуры. // Межведомственный тематический научный сборник. – Таганрог: Радиотехн. ин-т, 1987. – Вып. 9. – С. 19-21.
21. Малиновский Б.Н., Боюн В.П., Козлов Л.Г. Принципы повышения производительности проблемно-ориентированных процессоров для решения сложных научно-технических задач // Тезисы докл. IV Всесоюзн. школы-семинара "Распараллеливание обработки информации". – Львов: Физ. - мех. ин-т. АН УССР, 1985. – Ч. 3. – С. 3-6.
22. Боюн В.П., Козлов Л.Г. Высокопроизводительные средства обработки информации для систем реального времени. – Киев, 1976. – С. 31-38.

23. Боюн В.П., Козлов Л.Г., Терещенко В.И. Об одном подходе к построению мультипроцессорных систем для решения обыкновенных дифференциальных уравнений // Управляющие машины и системы. – 1982. - №2. – С. 42-45.
24. Корнеев В.В. Параллельные вычислительные системы. – М.: Нолидж, 1999. – 320с.
25. Корнеев В.В. Параллельное программирование в MPI. Москва-Ижевск: Институт компьютерных исследований, 2003. – 304с.
26. Каляев И.А., Мельник Э.В. Интеллектуальные многопроцессорные вычислительные системы // Тезисы доклада третьей международной научной конференции "Высокопроизводительные вычислительные системы". – Таганрог: Изд-во ТРТУ, 2006. – С. 14-19.
27. Каляев А. В., Левин И.И. Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений. – М.: Янус-К, 2003. – 380с.
28. Отчет о научно-исследовательской работе ГТ-11-2000 "Научные основы оптимизации структур высокопроизводительных вычислительных систем и методы реализации параллельных алгоритмов" // Л.П.Фельдман, О.А.Дмитриева, И.А.Назарова, Т.В.Михайлова. – Донецк: ДонГТУ, 2002. – 173с., № гос. регистрации 0202U002619.
29. Отчет о научно-исследовательской работе Д-1-03 "Методы алгоритмизации, топологического отображения и оптимизации структур параллельных и распределенных вычислительных систем" // Л.П.Фельдман, О.А.Дмитриева, И.А.Назарова, Т.В.Михайлова. – Донецк: ДонГТУ, 2005. – 231с., № гос. регистрации 0103U001824.
30. Дьяконов В. Mathematica 4:учебный курс. – СПб.: Питер, 2001. – 656с.
31. Дьяконов В.П. Mathematica 4 с пакетами расширений. – М.: Нолидж, 2000. – 606с.
32. Антонов А.С. Введение в параллельные вычисления. – М.: Изд-во МГУ, 2002. – 69с.
33. Group W., Lusk E., Skjellum A. Using MPI. Portable parallel programming with Message Passing Interface. – MIT Press, 1999. – 303c.

34. MPI (Message Passing Interface): теория и примеры программ. – Режим доступа: http://www.iitk.ac.in/cc/param/index.html
35. Евсеев И. MPI для начинающих. Учебное пособие и примеры. – Режим доступа: http://rsusu1.rnd.runnet.ru/ncube/koi8/mpibeg.html
36. MPI-FM пакет MPICH для высокопроизводительных кластеров. – Режим доступа: http://www-csag.ucsd.edu/projects/comm/mpi-fm.html
37. Арушунян О.Б., Залеткин С.Ф., Калиткин Н.Н. Тесты для вычислительного практикума по обыкновенным дифференциальным уравнениям // Вычислительные методы и программирование, 2002, т.3. – С. 11-19.
38. Фельдман Л.П., Назарова И.А. Эффективность параллельных алгоритмов оценки локальной апостериорной погрешности для численного решения задачи Коши // Электронное моделирование, т. 29, № 3, 2007. – С. 11-25.
39. Фельдман Л.П., Назарова И.А. Параллельные алгоритмы численного решения задачи Коши для систем обыкновенных дифференциальных уравнений // Математическое моделирование, т.18, № 9, 2006. – С. 17-31.
40. Фельдман Л.П., Назарова И.А., Хорошилов А.В. Параллельные блочные алгоритмы умножения матриц для мультикомпьютеров с распределенной памятью // Наукові праці Донецького національного технічного університету. Серія: Інформатика, кібернетика та обчислювальна техніка, випуск 8(120): – Донецьк, ДонНТУ, 2007. – С. 297-309.
41. Назарова И.А. Экспоненциальные методы решения линейной задачи Коши с альтернативными способами оценки локальной погрешности для массивно-параллельных компьютерных систем // Научно-теоретический журнал ИПИИ НАН Украины «Искусственный интеллект», №4, 2007. – Донецк: ИПИИ, 2007. – С. 474-482.
42. Назарова И.А. Эффективность применения технологии локальной экстраполяции в параллельных алгоритмах численного решения задачи Коши // Научно-теоретический журнал ИПИИ НАН Украины «Искусственный интеллект», №3, 2006. – Донецк: ИПИИ, 2006. – С. 192–202.

43. Назарова И.А. Параллельные полностью неявные методы численного решения жестких задач для СОДУ // Научно-теоретический журнал ИПИИ МОН и НАН Украины «Искусственный интеллект», №3, 2005. – Донецк: ИПИИ, 2005. – С. 185-193.
44. Назарова И.А. Повышение эффективности параллельных вычислительных систем при решении задачи Коши неявными методами Рунге-Кутты // Научные труды Донецкого национального технического университета. Выпуск 93. Серия: «Информатика, кибернетика и вычислительная техника» (ИКВТ-2005) – Донецк: ДонНТУ, 2005. – С. 58-67.
45. Назарова И.А. Эффективность численного решения нежестких СОДУ с контролем локальной погрешности для компьютеров с распределенной памятью // Научно-теоретический журнал ИПИИ НАН Украины «Искусственный интеллект», №3, 2004. – Донецк: ИПИИ, 2004. – С. 212-215.
46. Фельдман Л.П., Назарова И.А. Применение технологии локальной экстраполяции для высокоточного решения задачи Коши на SIMD-структурах // Научные труды Донецкого национального технического университета. Выпуск 70. Серия: «Информатика, кибернетика и вычислительная техника» (ИКВТ-2003) – Донецк: ДонНТУ, 2003. – С. 98-107.
47. Фельдман Л.П., Назарова И.А. Особенности использования методов Рунге-Кутты при моделировании параллельных процессов // Научные труды Луганского отделения Международной Академии информатизации №2(7). – Луганск, 2003. – С. 73-77.
48. Фельдман Л.П., Назарова И.А. Параллельная реализация численного решения нежестких обыкновенных дифференциальных уравнений вложенным методом Кутты-Мерсона // Научные труды Донецкого национального технического университета. Серия: проблемы моделирования и автоматизации проектирования динамических систем, выпуск 52. – Донецк, 2002. – С. 106-112.

49. Фельдман Л.П., Назарова И.А. Эффективность параллельных алгоритмов вложенных методов Рунге-Кутты при моделировании сложных динамических систем // Материалы II Международного научно-практического семинара «Высокопроизводительные параллельные вычисления на кластерных системах». – Нижний Новгород: Изд-во НГУ, 2002. – С. 294-301.
50. Фельдман Л.П., Назарова И.А. Моделирование сложных динамических систем на базе вложенных методов Рунге-Кутты // Сборник трудов международной научно-технической конференции. Компьютерные технологии в управлении, диагностике и образовании (КТУДО-2002). – Тверь, Тверской государственный технический университет, 2002. – С. 150-152.
51. Назарова И.А. Параллельные методы решения СОДУ большой размерности при моделировании сложных систем // Матеріали V Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених «Системний аналіз та інформаційні технології». – К.: НТУУ КПІ, 2003. – С. 87-88.
52. Фельдман Л.П., Назарова И.А. Методы контроля шаговой погрешности при параллельном решении систем обыкновенных дифференциальных уравнений // Материалы Двенадцатой Международной конференции по вычислительной механике и современным прикладным программным системам, Владимир.– М.: Изд-во МАИ, 2003. – т. 2. – С. 619-620.
53. Назарова И.А., Шаповалов В.А. Моделирование сложных динамических систем на высокопроизводительных компьютерах с распределенной памятью // Системний аналіз та інформаційні технології: Матеріали VI Міжнародної науково-практичної конференції.– К.: НТУУ «КПІ», 2004. – С. 189-191.
54. Назарова И.А., Фельдман Л.П. Эффективность параллельного численного решения нежестких систем обыкновенных дифференциальных уравнений с контролем локальной погрешности // Материалы XX Международного

семинара по струйным, отрывным и нестационарным течениям. – Санкт-Петербург. – СПб.: ИПЦ СПбГУТД, 2004. – С. 201-202.

55. Назарова И.А., Фельдман Л.П. Масштабируемый параллельный алгоритм численного решения линейных СОДУ для компьютеров с распределенной памятью // Материалы V Международной конференции по неравновесным процессам в соплах и струях (NPNJ-2004). – М.: Вузовская книга, 2004. – С. 153-155.
56. Фельдман Л.П., Назарова И.А. Эффективность способов оценки апостериорной локальной погрешности при параллельном решении систем линейных однородных ОДУ // Высокопроизводительные параллельные вычисления на кластерных системах. Материалы четвертого Международного научно-практического семинара и Всероссийской молодежной школы. – Самара, 2004. – С. 255-263.
57. Назарова И.А. Параллельные алгоритмы численного решения задачи Коши для СОДУ / Донбас-2020: наука і техніка – виробництву: Матеріали III науково-практичної конференції. м. Донецьк, 30-31 травня 2004. – Донецьк, ДонНТУ Міністерства освіти і науки, 2004. – С. 522 – 527.
58. Назарова И.А., Фельдман Л.П. Разработка и анализ эффективности параллельных алгоритмов итерационных методов решения СОДУ для мультипроцессоров с распределенной памятью // Материалы международной конференции по вычислительной механике и современным прикладным программным системам (ВМСППС-2005). – М.: Вузовская книга, 2005. – С. 343-345.
59. Назарова И.А. Масштабируемые параллельные алгоритмы численного решения жестких систем обыкновенных дифференциальных уравнений для многопроцессорных вычислительных систем с распределенной памятью // Материалы международной научно-технической конференции “Интеллектуальные и многопроцессорные системы: ИМС‘2005”.– Таганрог-Донецк-Минск: Изд-во ТРТУ, т.1, 2005. – С. 218-220.
60. Назарова И.А. Масштабируемый параллельный алгоритм численного решения задачи Коши для ВС с распределенной памятью // Тези доповідей Міжнародної науково-практичної конференції “Сучасні

проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій". – Запоріжжя, 2006. – С. 173-175.

61. Назарова И.А. Масштабируемые параллельные алгоритмы численного решения задачи Коши для СОДУ / Сборник трудов международной конференции "Моделирование – 2006". – Киев: Институт проблем моделирования в энергетике им. Г.Е. Пухова НАН Украины, 2006. – С. 335-340.
62. Фельдман Л.П., Назарова И.А. Эффективность отображения экстраполяционных параллельных алгоритмов численного решения СОДУ на вычислительные структуры различных топологий. // Материалы седьмой международной научно-технической конференции "Интеллектуальные и многопроцессорные системы: ИМС'2006".– Таганрог: Изд-во ТРТУ, т.1, 2006. – С. 269–272.
63. Фельдман Л.П., Назарова И.А. Параллельные алгоритмы численного решения задачи Коши для мультипроцессоров с распределенной памятью // Труды III международной конференции «Параллельные вычисления и задачи управления» PACO 2006 памяти И.В. Прангишвили. – М.: Институт проблем управления им. В.А. Трапезникова РАН, 2006. – С. 184-196.
64. Назарова И.А., Фельдман Л.П. Параллельные экстраполяционные схемы высокоточного решения задачи Коши для мультикомпьютеров с распределенной памятью // Материалы VI Международной конференции по неравновесным процессам в соплах и струях (NPNJ-2006), 26 июня-1июля 2006 г. Санкт-Петербург. – М.: Вузовская книга, 2006. – С. 255-257.
65. Фельдман Л.П., Назарова И.А. Параллельная реализация технологии локальной экстраполяции симметричных методов решения задачи Коши для кластерных систем // Материалы XV международной конференции по вычислительной механике и современным прикладным программным системам (ВМСППС-2007), Алушта, Крым, 25-31 мая 2007. – М.: Вузовская книга, 2007. – С. 485-487.
66. Назарова И.А. Алгоритмические методы повышения эффективности параллельных ВС при численном решении СОДУ с контролем погрешности на шаге / И. А. Назарова // Материалы II международной

конференции “Моделирование и компьютерная графика”, Донецк: ДонНТУ, 2007. – С. 202-205.

67. Flynn M. Very high-speed computing systems. Proceeding of the IEEE №54 (12), 1966. – P.1901-1909.
68. Flynn M. Some Computer Organisations and Their Effectiveness // IEEE Trans. Computers. 1972. . – –V.21. N 9. – P.948-960.
69. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем. – СПб.: Питер, 2004. – 668с.
70. Бройнль Т. Параллельне програмування: Початковий курс: Навч. посібник / Вступ. слово А. Ройтера; Пер. з нім. В.А.Святного. – К.: Вища шк., 1997. – 358с.
71. Немнюгин С., Стесик О. Параллельное программирование для многопроцессорных вычислительных систем. – СПб.: БХВ-Петербург, 2002. – 396с.
72. Барский А.Б. Параллельные информационные технологии. – Москва: ИУИТ. Бином. Лаборатория знаний, 2007. – 503с.
73. Хокни Р., Джессхоуп К. Параллельные ЭВМ: Архитектура, программирование и алгоритмы. – М.: Радио и связь, 1986. – 392с.
74. Жуков И.А. Классификация архитектур вычислительных систем // Управляющие системы и машины , 1995, №6. – С.52-56.
75. Букатов А.А., Дацюк В.Н., Жегуло А.И. Программирование многопроцессорных ВС. – Ростов-на Дону: Изд-во ООО ЦВВР, 2003. – 208с.
76. Богачев К.Ю. Основы параллельного программирования. – М.: БИНОМ. Лаборатория знаний, 2003. – 342с.
77. Информационно-аналитические материалы по параллельным вычислениям научно-исследовательского вычислительного центра (НИВЦ) МГУ. – Режим доступа: http://parallel.ru.
78. Основные классы современных параллельных компьютеров. – Режим доступа: http://parallel.ru/computers/classes.html
79. Т-Платформы. – Режим доступа: http://www.t-platforms.ru/clusters/communications.html
80. Наиболее используемые коммуникационные технологии. – Режим доступа: http://parallel.ru/computers/interconnects.html

81. Руководство по коммутаторам Myrinet. Официальное описание. – Режим доступа:http://www.myri.com/myrinet/m3switch/guide/myrinet-2000_switch_guide.pdf
82. Официальный портал разработчика SCI. – Режим доступа: http://www.dolphinics.com
83. Официальный портал Intel, раздел коммуникаций Infiniband – Режим доступа: http://www.intel.com/technology/infiniband
84. Список 500 наиболее мощных суперкомпьютеров мира. – Режим доступа: http://www.top500.org
85. Список 50 наиболее мощных суперкомпьютеров СНГ. – Режим доступа: http://www.supercomputers.ru/page=rating
86. Foster I. Designing and Building Parallel Programs. – Addison-Wesley, 1999. – 302с.
87. Берзигияров П.К., Султанов В.Г. Технология разработки масштабируемых параллельных вычислений для SMP систем на базе MPI // Материалы первой молодежной школы «Высокопроизводительные вычисления и их приложения». – Режим доступа: ftp://parallel.ru/parallel/chg1999.
88. Amdahl G. Validity of the single processor approach to achieving large scale computing capabilities // In AFIPS Conference proceeding, Washington, D.C.: Thompson Books, Vol. 30. – P.483-485.
89. Gustavson J.L. Reevaluating Amdahl’s law // Communications of the ACM. 31(5). – P.532-533.
90. Pastor L., Bosque J. L. An efficiency and scalability model for heterogeneous clusters // Proceedings of Cluster 2001, 8-11 October 2001, Newport Beach, CA, USA. IEEE Computer Society. – P.427-434.
91. Dongarra J., R. van de Geun, Walker D. Scalability Issues Affecting the Design of a Dense Linear Algebra Library // Journal of Parallel and Distributed Computing, 22, 1994. – P.523-537.
92. Kalinov A. Scalability Analysis of Matrix-Matrix Multiplication on Heterogeneous Clusters, Proceedings of 3rd ISPDC/HeteroPar'04, Cork, Ireland, July 05 - 07, 2004, IEEE CS Press. – P.303-309.

93. Хайрер Э., Нерсетт С., Ваннер Г. Решение обыкновенных дифференциальных уравнений. Нежесткие задачи. – М.: Мир, 1990. – 512с.
94. Хайрер Э., Ваннер Г. Решение обыкновенных дифференциальных уравнений. Жесткие и дифференциально-алгебраические задачи. – М.: Мир,1999. – 685с.
95. Холл Дж., Уатт Дж. Современные численные методы решения обыкновенных дифференциальных уравнений. – М.: Мир, 1999. – 311с.
96. Miranker W.L. A survey of parallelism in numerical analysis. SIAM Review, 1971, vol. 13. – P.524-547.
97. Houwen P.J., Sommeijer B.P. Parallel ODE solvers // Proceedings of the International Conference on Supercomputing, 1990, ACM Press. – P. 71-81.
98. Parallel predictor-corrector methods of Runge-Kutta type / Houwen P.J. van der, Nguyen hun Cong. // Rept/ Cent. Math. And Comput. Sci. – 1992. – Nm R9220. – P. 1-10.
99. Houwen, P.J. van der & Sommeijer, B.P. Runge-Kutta methods on parallel computers // J. Z. Angen Math. Mech. – 1992. - №68. – P. 3-10.
100. Jackson K.R., Norsett S.P. The potential for parallelism in Runge-Kutta methods. Part I: RK formulas in standart form // Rep. Depart. of Comp. Sc. 1990 – № 239/90. – P. 41-60.
101. Houwen, P.J. van der & Sommeijer, B.P. Parallel iteration of high-order Runge-Kutta methods with stepsize control // J. Comp. Appl. Math. – 1990. №29. – P. 111-127.
102. Г.Ю. Куликов. Численное решение задачи Коши для системы дифференциально-алгебраических уравнений с помощью неявных методов Рунге-Кутты с нетривиальным предиктором // Журнал вычислительной математики и математической физики, 1998, том 38, № 1. – С.68-84.
103. Shampine L.F.&Watts H.A. Block implicit one-step methods // Math. Comp. 23, 108. – P. 252-266.
104. Houwen P.J., Sommeijer B.P. Parallel ODE solver // Proceedings of the International Conference on Supercomputing. – ACM Press, 2000. – P.71-81.

105. Houwen P.J., Sommeijer B.P. CWI Contribution to the development of parallel Runge-Kutta methods / Preprint NM-R9506, CWI, Amsterdam, 2001. –23p.

106. Houwen P.J., Swart J.J. Triangularly implicit methods for the numerical solution of ordinary differential equations / Preprint NM-R9510, CWI, Amsterdam, 2003. –23p.

107. Worland P.B. Parallel methods for the numerical solution of ordinary differential equations // IEEE Trans. Comp. C. – 25, 10(1976). – P.1045-1048.

108. Молчанов И.Н. Введение в алгоритмы параллельных вычислений. – Киев: Наукова думка, 1990. – 128с.

109. Фельдман Л.П. Сходимость и оценка погрешности параллельных одношаговых блочных методов моделирования динамических систем с сосредоточенными параметрами // Наукові праці ДонДТУ. Серія: Інформатика, кібернетика та обчислювальна техніка, випуск 15, Донецьк: ДонДТУ, 2000. – С.34-39.

110. Фельдман Л.П., Дмитриева. О.А. Разработка и обоснование параллельных блочных методов решения обыкновенных дифференциальных уравнений на SIMD-структурах // Наукові праці ДонДТУ. Серія: Проблеми моделювання та автоматизації проектування динамічних систем, випуск 29:- Донецьк: ДонДТУ, 2001. – С.70-79.

111. Feldman L., Dmitriewa O.A., Gerber S. Abbildung der blockartigen Algoritmen auf die Parallelrechnerarchitekture / Simulationstechnik, 17. Symposium in Magdeburg, Sept. 2003: SCS-Europe BVBA (ISBN 3-936150-27-3), Magdeburg, Germany, 2003. – P.359-364.

112. Feldman L., Svjatnyj V., Dmitriewa O.A. Stabilitat von parallelen Simulationsverfahren fur dynamische Systeme mit konzentrierten Parametern / Simulationstechnik, 17. Symposium in Magdeburg, Sept. 2003: SCS-Europe BVBA (ISBN 3-936150-27-3), Magdeburg, Germany, 2003. – P.105-110.

113. Фельдман Л.П. Общие линейные блочные многошаговые методы решения // Наукові праці ДонНТУ. Серія: Інформатика, кібернетика і

обчислювальна техніка, випуск 8(120): – Донецьк: ДонНТУ, 2007. – С. 282-297.

114. Вержбицкий В.М. Основы численных методов. – М.: Высшая школа, 2002. – 840с.

115. Арушанян О.Б., Залеткин С.Ф. Численное решение обыкновенных дифференциальных уравнений на Фортране.– М.: МГУ,1990.–336с.

116. Крылов В.И., В.В. Бобков В.В., Монастырский П.И. Вычислительные методы, том I. – М.: Наука, 1976. – 303с.

117. Крылов В.И., В.В. Бобков В.В., Монастырский П.И. Вычислительные методы, том II. – М.: Наука, 1977. – 399с.

118. Иванов В.В. Методы вычислений на ЭВМ. – К.: Наукова думка, 1986. – 584с.

119. Голуб Д., Ван Лоун Ч. Матричные вычисления: Пер. с англ. – М.: Мир, 1999. – 548с.

120. Деммель Дж. Вычислительная линейная алгебра. Теория и приложения. Пер. с англ. – М.: Мир, 2001. – 430с.

121. Demmel J., Higham N.J. Stability of block algorithms with fast Level 3 BLAS // ACM Trans. Math. Software, 18, 1992. – P. 274-291.

122. Choi J., Dongarra J., Walker D. PUMMA: Parallel universal matrix multiplication algorithms on distributed memory concurrent computers. – Режим доступа: http://citeseer.ist.psu.edu/choi93pumma.html

123. Strassen V. Gaussian elimination is not optimal // Numer. Math. 13. – P. 354-356.

124. Pan V. How can we speed up matrix multiplication // SIAM Rev., 26, 1984. – P. 393-416.

125. Winograd S. A new algorithm for inner product // IEEE Trans. Comp. C-17. – P. 693-694.

126. Bailey D.H. Extra high speed matrix multiplication on CRAY-2 // SIAM J. Sci. and Stat. Comp. 9. – P. 603-607.

127. Bailey D.H., Lee K., Simon H.D. Using Strassen's algorithm to accelerate the solution of linear systems. J. Supercomputing, 4: 97-371, 1991.

128. Андерсон Дж. Дискретная математика и комбинаторика. – М.: Мир, 2001. – 960с.

ЗМІСТ

Наукове видання

ФЕЛЬДМАН Лев Петрович,

НАЗАРОВА Ірина Акопівна

ПАРАЛЕЛЬНІ ОДНОКРОКОВІ МЕТОДИ ЧИСЕЛЬНОГО РОЗВ'ЯЗАННЯ ЗАДАЧІ КОШІ

Монографія

(українською мовою)

Редакційно-технічне оформлення, коректура *Г.А. Федоренко*

Підп. до друку 28.01.2011. Формат 60x84/16.
Папір офсетний. Друк різографія.
Ум. друк.арк. 10,4. Обл.-вид. арк. 9,6
Тираж 300 прим. Замовлення №

Видавництво ДВНЗ «ДонНТУ»
Україна, 83001, м. Донецьк, вул. Артема, 58. Тел.: (062) 301-03-04

Свідоцтво про державну реєстрацію суб'єкта видавничої справи:
серія ДК №2982 від 21.09.2007.

Надруковано: РВВ ДонНТУ, м. Донецьк, вул. Артема, 58, 9-й уч. корп.