Симулация на течения в порести среди чрез масивно паралелен Многонивов Монте Карло алгоритъм

Николай Шегунов

Автореферат

Факултет по Математика и Информатика, СУ "св. Климент Охридски"

Настоящата работа се представя за придобиване на образователна и научна степен "Доктор"по направление 4.6

Научен ръководител:

доц. Петър Армянов

2021

0. Съдържание

1. Въведение

1.1 Мотивация

Окачествяването на неопределеността в моделите е важен компонент в различни научни области. С все по-голямото търсене на по-точни прогнозни модели, стохастичното моделиране е бързо нарастваща област в приложната математика и научните изчисления. Този вид моделиране намира широко приложение в различни научни сфери като невроните мрежи и машинното самообучение, финансовата математика и управлението на риска. Друга важна област е окачествяването на неопределеността в индустриалните математически модели. Нарастващият интерес породи нови списания като SIAM и свързани годишни конференции [1]. Важна област, в която окачествяването на неопределеността е неизменна част, е при модели свързани с течения в порести среди. Такива изследвания са важни за много индустриални проблеми, например при разработването на композитни материали, при моделиране на безопасно съхранение на радиоактивни материали, при симулиране на процес на филтрация и други. Съществен проблем при такъв тип модели е нуждата от високопроизводителни системи, чрез които могат да се моделират реалистични проблеми. Надеждите са, че с развитието в сферата на изчислителните системи и появата на мощни компютърни клъстери може да бъде постигнат напредък в тази област. Такъв напредък поражда естествената нужда от разработване на нови алгоритми, които могат да използват ефективно наличния изчислителен ресурс.

Съществуват няколко различни разработени подхода, които могат да отговорят на такъв тип модели. Важен алгоритъм, широко разпространен в практически задачи, използващи високо-производителни системи, е добре познатия Монте Карло алгоритъм. Той се основава на повтарящо се провеждане на експерименти и последващата им статистическа обработка. Тази проста идея намира приложение в много проблеми от индустрията, където е трудно да се формулира детерминистичен проблем. Чрез този вид алгоритъм се решават сложни задачи, като например при разработването на нови материали. Макар, че може да се използва за решаване на проблеми, свързани с изчислителна флуида динамика, той не е особено приложим, поради необходимостта от изключителни изчислителни ресурси. За такива проблеми се търсят по-бързи методи. За да се пре-

одолее тази изчислителна бариера през последните години голям интерес в научните среди предизвиква обобщена версия на алгоритъма Монте Карло, наречена Многонивов Монте Карло. Този обобщен подход дава възможност за много по-бързо изчисление на модели от флуидната динамика. За пръв път той е използван за параметрично интегриране на $E[f(x, \lambda)]$, където x е крайномерна случайна величина, а λ е параметър, в работата на Хайнрих от началото на 21-ви век **[2, 3, 4]**. В наши дни може да бъден намерен приложен в множество различни области **[1]**.

Основната идея на алгоритъма е проблемът, който се решава, да бъде разделен на сходни под-проблеми с различни характеристики, наречени нива. Всяко ниво е приближение на оригиналния проблем и се определя от изчислителна цена. Нивата се характеризират с по-малко време за изчисление спрямо целия проблем. Комбинирани по правилен начин, те могат да доведат до значително намаляване на изчислителните ресурси необходими за симулацията. Многонивовият Монте Карло алгоритъм идва с набор от различни предизвикателства, свързани с прилагането му. Конструирането на алгоритъма изисква правилно дефиниране и комбиниране на различните нива. Съществен проблем също е паралелното му изпълнение, тъй като работата по нивата изисква сложна синхронизация **[5]**.

В тази работа е разгледан Многонивов Монте Карло алгоритъм, приложен за проблеми свързани с течения в порести среди. За такъв тип проблеми са разгледани двата аспекта на алгоритъма: как нивата биват дефинирани и как той може да бъде паралелизиран. Симулацията на течения в порести среди изисква решаването на стохастично диференциално уравнение. Този тип уравнения са разширение на идеята за частно диференциално уравнение, чрез въвеждането на неопределеност в модела. Тази неопределеност се моделира чрез стохастични процеси върху вероятностно пространство, от което се генерират експерименти. Окачествяването на неопределеността води до изключително сложни изчислителни задачи.

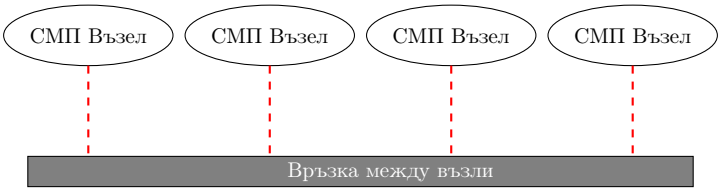
1.2 Стохастични изчисления

Както бе споменато, в много изчислителни модели съществува някакъв вид неопределеност. Тя може да се поражда от физичен феномен, скъпо струващ експеримент или комбинация от двете. Неопределеността в моделите може да бъде част от различни контексти: като входни данни, като случайни коефициенти на модела и други. Съществен въпрос при моделирането на такива процеси е описването на неопределеността като множество от крайномерни величини. В литературата се намират

няколко основни подхода за това. Популярен подход за параметеризация е развитието в ред на Кархунен-Лиове [6]. Друг подход е използването на права и обратна Фурие трансформация върху циркулаторна матрица [7] или декомпозиция на Холески [8].

1.3 Съвременни високо-производителни системи

Съвременните високопроизводителни системи са изградени чрез хибридна хардуерна архитектура (фигура **1.1**). Всеки възел се състои от симетрична много-процесорана единица (СМП). Разпределената паралелизация в такава система се постига чрез комуникация, протичаща по мрежов канал. За постигането на ниска латентност и висококоростен обмен на данни се използват специални инфини-банд мрежи.



Фигура 1.1: Хибридна архитектура

За този модел основните ресурси, които програмата трябва да управлява, са оперативната памет и процесорните ядра. Това означава че една програма може да разпределя работа между различни процесори, както във възел, така и между възли, а също да разпределя паметта, която използва между различните възли. За този тип архитектура, общоприетата програмна рамка се нарича интерфейс за обмяна на съобщения[1]. Интерфейсът за обмяна на съобщения е стандартен модел от тип актьор, който използва изпращане на съобщения между различните възли за осъществяване на комуникация и обмяна на данни [9]. Разработен е с цел да бъде приложим в широк набор от паралелни архитектури. В стандарта са дефинирани синтаксис и семантика на множество библиотечни процедури. Значими имплементации на стандарта са: “Open MPI” и имплементацията на Intel. Комуникационният протокол предлага както комуникация от точка до точка, така и колективна комуникация. Процесите са групирани в единици, наречени комуникатори. Всеки процес може да комуникира с

[1] на английски: Message Passing Interface (MPI)

друг процес в рамките на своя комуникатор. Процеси от различни комуникатори не могат да комуникират директно.

Освен библиотека с имплементация на интерфейс за обмяна на съобщения, при симулация на различни индустриални процеси често се използват допълнителни програмни библиотеки. Най-важната група такива библиотеки са тази, отговорни за изчисления и алгоритми от линейната алгебра. Това обуславя и големия брой съществуващи решения.*"LAPAK", "Eigen", "Armadio", "SuperLU"* и други. За съжаление съществуват малко програмни библиотеки за линейна алгебра, способни да използват масивно паралелни системи. Съществуват само няколко основни библиотеки, които се разработват активно: *"Deal.ii"* и *"Dune"*[2] **[10, 11]**; *"OpenFoam"*, която е специализирана за изчисления за динамика на флуиди, както и "Portable, Extensible Toolkit for Scientific Computation (PETSc)"[3].

Библиотеката "Dune" е изградена от модули, всеки от които е отговорен за различен компонент. Основната ѝ идея е да обедини стари и ново разработвани библиотеки в единен интерфейс, който да даде възможност за лесно разработване на програми, свързани със симулации на диференциални уравнения. Този модулен подход дава лесен начин за програмно решаване на диференциални уравнения. Библиотеката е написана на езика "C++" с широко използване на шаблони и различни новости в езика, с цел постигане на максимално бързодействие **[12]**.

1.4 Цели и задачи на тази работа

В тази работа се разглежда Многонивовия Монте Карло алгоритъм за симулации на проблеми, свързани с течения в порести среди. Основната заложена цел е да се адаптира и реализира ефикасна паралелна версия на този алгоритъм, по-бърза в сравнение с класическата Монте Карло алгоритмична схема. Тази реализация трябва да дава възможност за реалистични симулации във високо производителна изчислителна среда. За постигането на таз цел са дефинирани следните задачи:

- Изследване на съществуващи решения на проблема;
- Избор на подходящ начин за генериране на случайни полета на пропускливост за порести среди;
- Разработване на ефективен начин за комбинация на резултати от симулации при различни нива на гранулярност;

[2] http://dealii.org/ и https://www.dune-project.org/
[3] https://www.mcs.anl.gov/petsc/

- Разработване на подходящ, ефективен подход за разпределение на ресурсите при паралелна реализация на Многонивовия Монте Карло алгоритъм;
- Избор на подходящ софтуер за ефективна реализация на алгоритъма във високо производителна среда.

2. Математически модели

2.1 Алгоритми за генериране на случайни експерименти

Основен проблем в програмните приложения за окачествяване на неопределеността е как да бъде генериран експеримент по предварително зададена ковариационна матрица, задаваща многомерно разпределение.

Ковариацията е мярка за съвместната вариация на две случайни величини. В зависимост от начина на разпределение на случайните величини, тя може да има положителна или отрицателна стойност:

$$cov(X,Y) = E[(X - E[X])(Y - E[Y])] = E[XY] - E[X]E[Y]$$

Пряко свързаната статистическа мярка *корелация* се определя като:

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_x \sigma_y} = E[(X - E[X])(Y - E[Y])]/(\sigma_x \sigma_y)$$

Корелацията е мярка, която определя до колко две случайни величини са зависими една от друга или как промяна в стойността на едната влияе на стойността на другата.

Нека $X = (X_1, X_2, \dots X_n)^T$ е вектор от случайни величини, всяка от които е с крайна вариация и очаквана стойност. Тогава ковариационната им матрица е дефинирана като матрица с елементи: $\Sigma_{i,j} = cov(X_i, X_j)$ а корелационната, матрица, съответно, има елементи: $cov(X_i, X_j)/\sigma_x \sigma_y$. За да може да бъдат генерирани случайни величини се използва ковариационната матрица. Основната идея е да се генерира вектор от независими случайни величини, който чрез линейна трансформация да бъде трансформиран, така че да се получи определена корелация между елементите му. Това изисква да се намери такава трансформация, чрез която ковариационната матрица да бъде диагонализирана. Лесен, но не оптимален, начин за това е да се използва декомпозицията на Холески [13, 14, 15, 16]. От изчислителна гледна точка този метод може да бъде приложен върху всякакъв тип ковариационни матрици. Съществен недостатък на този подход е голямото количество памет, необходимо за съхранението на

матрицата. При определени допълнителни условия за равномерно разположение на точките, в които се генерира извадка от разпределението, може да бъде използван по-ефективен алгоритъм, върху опростена ковариационна матрица. Той е едновременно по-бърз и по-икономичен към памет спрямо декомпозицията на Холески. Този алгоритъм се нарича циркулаторно влагане [13, 14].

2.2 Метод на крайните обеми

Методът на крайните обеми е един от най-универсалните методи, широко използван при изчисления от областта на флуидната динамика. Разработен е за дискретизация на уравнения, описващи консервативни закони във физиката. Типичен пример за такъв закон е едномерното частно диференциално уравнение [17, 18]:

$$q_t(x,t) + f(q(x,t))_x = 0 \tag{2.1}$$

Обикновено консервативните закони възникват и се записват в интегрална форма. Уравнението **2.1**, при фиксирани x_1, x_2 придобива вида:

$$\frac{d}{dt}\int_{x_1}^{x_2} q(x,t)dx = f(q(x_2,t)) - f(q(x_1,t)) \tag{2.2}$$

Всеки компонент на q измерва плътността на някаква величина. Уравнение **2.2** постулира, че общото количество на величината зависи само от двете крайни точки, поради масовия поток между x_1 и x_2. Ако се допусне, че масата не е консервативна, тогава уравнение **2.2** трябва да съдържа член с източник на маса. Съществуват много такива консервативни системи в практиката като повечето от тях не са линейни. Това води до трудности при изчислението им, което може дори да стане физически некоректно [17, 18]. Вместо да се приближават производните, както е в много други методи, методът на крайните обеми разглежда системата в контролни обеми, наречени клетки. Клетките са обикновено многоъгълници в съответното пространство. Неизвестното количество се приближава като средна стойност в клетката и потокът се балансира в двата и края.

2.3 Многонивов Монте Карло алгоритъм

Идеята на Монте Карло алгоритмите е много интуитивна. Ако се допусне, че трябва да бъде пресметната очакваната стойност на дадено количество Q в модела, то за да се постигне това могат да бъдат генерирани множество експерименти Q^i, за който да се пресметне очакваната стойност $E[Q]$:

$$E[Q_{M,N}] = \frac{1}{N}\sum_{i=1}^{N} Q_M^i, \text{ където грешката се контролира от } \frac{1}{\sqrt{N}}$$

Тук M е характеристика на приближението на диференциалната задача, а N е броят проведени експерименти. Коренът от средно квадратичната грешка е $\mathcal{O}\left(\frac{V[Q]}{\sqrt{N}}\right)$, където $V[Q]$ е вариацията. Това означава, че за постигане на точност на метода от ϵ трябва да бъдат проведени поне $\mathcal{O}(\epsilon^{-2})$ експеримента. Това е и основният недостатък на чистия Монте Карло алгоритъм - неговата изчислителна сложност. Може да се наложи провеждането на голям брой скъпо струващи експерименти за достигане на желаната точност. Един начин за преодоляване на този проблем е изборът на специално подбрани експерименти - това са т.нар. Квази Монте Карло методи **[1]**, **[19]**. По-добра стратегия за преодоляване на бавната сходимост на метода е разделянето на проблема на под-проблеми, като част от тях са евтини за изчисление, а друга са по-скъпи и след това правилно комбиниране на резултатите. Това води до по-добра сходимост от класическия Монте Карло метод. Основната идея е търсената очаквана стойност да се разгледа като телескопична сума:

$$Q(\omega) = \underbrace{Q_{M_0}(\omega)}_{Y_0(\omega)} + \underbrace{Q_{M_1}(\omega) - Q_{M_0}(\omega)}_{Y_1(\omega)} + \cdots + \underbrace{Q_{M_L}(\omega) - Q_{M_{L-1}}(\omega)}_{Y_L(\omega)}$$

В това уравнение за изчислението на всеки член $Y(\omega)$, се използва стандартен Монте Карло подход. Един член на тази сума се нарича ниво. Този подход се нарича Многонивов Монте Карло алгоритъм.

Нека $E[Q_M]$ бъде приближение, достатъчно близко до $E[Q]$, чрез избор на достатъчно голямо M . Целта е $E[Q]$ да бъде приближено чрез $E[Q_M]$. Това може да бъде постигнато чрез пресмятане на $\widehat{Q}_M$ и окачествяване на неопределеността чрез корен от средно квадратичната грешка:

$$e(\widehat{Q}_M) = (E[(\widehat{Q}_M - E[Q])^2])^{1/2} \tag{2.3}$$

При така въведената грешка, стандартният Монте Карло метод се дефинира чрез:

$$\widehat{Q}^{MC}_{M,N} = \frac{1}{N}\sum_{i=1}^{N} Q^i_M \tag{2.4}$$

Където Q^i_M, $i = 1, \dots N$ са независими експерименти от търсената величина Q_M. При избор за изчислителна цена на един експеримент $C(Q^i_M) = \mathcal{O}(M^\gamma)$, за положителна константа γ, средно квадратичната грешка придобива вида:

$$\begin{aligned} e(\widehat{Q}^{MC}_{M,N})^2 &= E[(\widehat{Q}^{MC}_{M,N} - E[\widehat{Q}^{MC}_{M,N}] + E[\widehat{Q}^{MC}_{M,N}] - E[Q])^2] \\ &= E[(\widehat{Q}^{MC}_{M,N} - E[\widehat{Q}^{MC}_{M,N}])^2] + (E[\widehat{Q}^{MC}_{M,N}] - E[Q])^2 \\ &= V[\widehat{Q}^{MC}_{M,N}] + (E[\widehat{Q}^{MC}_{M,N}] - E[Q])^2 \end{aligned} \tag{2.5}$$

Тъй като:

$$E[\widehat{Q}^{MC}_{M,N}] = E[Q_M], \text{ и } V[\widehat{Q}^{MC}_{M,N}] = N^{-1}V[Q_M]$$

грешката приема вида:

$$e(\widehat{Q}^{MC}_{M,N})^2 = N^{-1}V[Q_M] + (E[Q_M] - E[Q])^2 \tag{2.6}$$

В разгледаните модели на диференциални уравнения M характеризира броя на клетките в дискретното приближение на диференциалното уравнение, а Q_M приближава Q. Вторият член в уравнение **2.6** представлява грешката от дискретизация на използвания числен метод. Преминаването от стандартен Монте Карло към Многонивов е много естествено. Нека $\{M_l : l = 0 \dots L\} \in N$ е растяща редица от числа, съответни за нивата l и на всяко ниво е съпоставено определено количество $\{Q_{M_l}\}_{l=0}^{L}$. Нека също така $s \geq 2$ е константа (фактор на загрубяване), такава че $M_l = sM_{l-1}$, за $l = 1 \dots L$. Тогава дефинирайки, $Y_l = Q_{M_l} - Q_{M_{l-1}}$, $Y_0 = Q_{M_0}$, може да бъде формулирано следното развитие за $E[Q_M]$:

$$E[Q_M] = E[Q_{M_0}] + \sum_{l=1}^{L} E[Q_{M_l} - Q_{M_{l-1}}] = \sum_{l=0}^{L} E[Y_l] \tag{2.7}$$

Очакваната стойност на най-финото ниво L се получава като сума от изчислената стойност на това ниво плюс корекции, получени чрез стойностите от по-грубите нива. Всеки член на сумата **2.7** се пресмята чрез стандартен Монте Карло алгоритъм с N_l брой експерименти.

$$\widehat{Y}_l = \frac{1}{N_l} \sum_{i=1}^{N_l} (Q_{M_l}^{(i)} - Q_{M_{l-1}}^{(i)}) \tag{2.8}$$

За минимизиране на общото време на изпълнение се дефинират критерии за прекратяване на алгоритъма и оценка на грешката като брой експерименти за отначало зададен средно квадратичен толеранс ϵ. Нека C_0, V_0 дефинират съответно цената и вариацията на един експеримент на Y_0 и C_l, V_l е цената и вариацията на един експеримент на Y_l. Тогава общата изчислителна сложност и вариация на метода се пресмятат чрез формулите:

$$C^{total} = \sum_{l=0}^{L} C_l N_l \tag{2.9}$$

$$V^{total} = \sum_{l=0}^{L} N_l^{-1} * V_l \tag{2.10}$$

При фиксирана вариация, цената за изчисление се минимизира, избирайки N_l така, че да минимизира следната сума **[1]**:

$$C^{total} + \lambda^2 V^{total} \tag{2.11}$$

При това положение общата изчислителна цена приема вида:

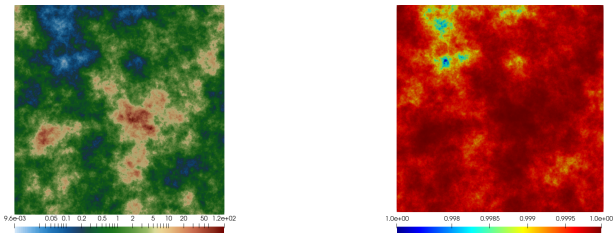
$$C^{total} = \frac{1}{\epsilon^2} \left(\sum_{l=0}^{L} \sqrt{V_l / C_l} \right)^2 \tag{2.12}$$

Непосредствено следствие, което определя ефективността на алгоритъма, е дали произведението $V_l C_l$ се увеличава или намалява с промяна на нивото l - дали цената расте по-бързо от намаляването на вариацията с увеличаването на нивото. Например, ако това произведение се увеличава с нивото, то доминиращ член ще е цената на най-финото ниво (членът

$V_L C_L$). Тогава за общата цена е в сила: $C^{total} \approx \epsilon^{-2} V_L C_L$. Ако доминиращ е най-грубият член на сумата (членът $V_0 C_0$), то за общата цена е в сила: $C^{total} \approx \epsilon^{-2} V_0 C_0$. Това контрастира с класическия Монте Карло подход, при който общата изчислителна цена е $\epsilon^{-2} V_0 C_L$, при предположение, че цената за изчисление на един експеримент Q_{M_L} е подобна на цената за изчисление на $Q_{M_L} - Q_{M_{L-1}}$, и $V[Q_{M_L}] \approx V[Q_{M_0}]$. Това показва, че в първия случай за Многониводия Монте Карло, изчислителната цена е намалена с фактор от V_L/V_0, съответстваща на отношението на вариацията $V[Q_{M_L} - Q_{M_{L-1}}]$ с $V[Q_{M_L}]$, докато във втория, цената е намалена с фактор от C_0/C_L - отношението на цената за изчисление на Q_{M_0} и $Q_{M_L} - Q_{M_{L-1}}$ [1].

2.4 Порестост и пропускливост

Две важни физични характеристики при моделите на течения в пресрти среди са *порестост* и *пропускливост*. Те присъстват в много модели от областта и са тясно свързани. *Порестост* е мярка, характеризираща отношението на запълненото спрямо свободното пространство в разглеждания материал. *Пропускливост* е свойството на разглеждания материал (например скална маса) да пропуска флуид.



(а) Пропускливост (б) Порестост

Фигура 2.1: Една реализация на поле на пропускливост и съответното поле на порестост, пресметнато чрез уравнението на Козени-Карман с параметри $\sigma = 2, \lambda = 0.2$

Двете свойства са свързани с броя, големината и свързаността на порите в материала. Например дадена скала може да бъде изключително пореста, но ако порите не са свързани, тя ще има ниско ниво на пропускливост и обратно - ако има малко на брой пори, но те са свързани, то скалата ще има висока пропускливост при ниска порестост. Когато потокът в средата е ламинарен, пропускливостта и порестостта могат да бъдат свързани

чрез уравнението на Козени–Карман [20]. В практиката пропускливостта се моделира като стохастично поле, следващо логаритъм от нормално разпределение и ковариационна функция с два параметъра: σ - стандартно отклонение и λ - константа, наречена корелационна дължина.

3. Многонивов Монте Карло алгоритъм за течения в порести среди

Стохастичните диференциални уравнения, както беше споменато, представляват сериозен интерес за много индустриални модели. Благодарение на развитието на високо-производителните системи, числените симулиране на такива процеси с реални параметри става възможно. Например модели, свързани със симулация на разпределение на подпочвени води в област от десетки километри. В настоящата теза са разгледани са два основни модела, но изследвания подход не се ограничава само до тях. Първият е стохастичното уравнение на Лаплас - прост, но добре установен метод, широко използван за изследване на алгоритми за симулация на течения в порести среди. Той показва добре трудностите пред симулацията на такива модели. Вторият е уравнението на процесът на конвекция-реакция-дифузия. Това уравнение е широко използвано за моделиране на различни процеси като филтрация, а също е и основен компонент в по-сложни съставни модели от биологията и химията.

3.1 Стохастично уравнение на Лаплас

Нека разгледаме стационарен поток в куб от порест материал с област $D = (0,1)^2$, дефинирано случайно поле Ω и пад в налягането между границите на областта

$$-\nabla \cdot [k(x,\omega)\nabla p(x,\omega)] = 0 \text{ за } x \in D = (0,1)^2, \omega \in \Omega. \tag{3.1}$$

при гранични условия:

$$\begin{aligned} p_{x_1=0} &= 1 \\ p_{x_1=1} &= 0 \\ \partial_n p &= 0 \text{ върху другите граници.} \end{aligned} \tag{3.2}$$

Коефициентът $k(x,\omega)$ описва пропускливостта на полето за дадената област, а решението $p(x,\omega)$ описва разпределението на налягането в областта. Интерес за модела представлява общия поток на границата на областта:

$$Q(x,\omega) := \int_{x_1=1} k(x,\omega)\partial_n p(x,\omega)dx. \tag{3.3}$$

3.1.1 Конструкция на Многонивовия Монте Карло

Общата дефиниция на ниво във формулировката на уравнение **2.7** дава голяма свобода за начина на конструиране на алгоритъма. Естествен подход за дефиниция на ниво в Многонивовия Монте Карло метод за разглеждания проблем е размерността на диференчната мрежа. За даден Монте Карло алгоритъм в уравнение **2.8** нивата са дефинирани като броя на разглежданите клетки по дадено направление на оста. Така, за дадена мрежа от $2^M \times 2^M$ клетки на фината мрежа, съответният брой клетки на по грубата мрежа е $2^{M-1} \times 2^{M-1}$. Тази дефиниция непосредствено означава, че генерираното поле на пропускливост трябва да бъде приближено на грубото ниво. Интуитивен начин това да бъде направено е чрез аритметично усредняване на всеки 4 съседни клетки в двумерния случай и на всеки 8 в тримерния. Този подход, за съжаление, не води до добро запазване на вариацията, а това е от решаващо значение за ефективността на алгоритъма. Добро запазване на вариацията означава малка вариация на разлика на оценката на две нива при голяма вариация на грубото ниво. Поради тази причина за приближение на полето е разгледана техника, наречена опростена ренормализация, широко използвана в миналото и все още използвана на някои места. Детайлно описание може да бъде намерено в **[21], [22], [23]**. Накратко, опростената ренормализация се основава на рекурсивно прилагане на хармонично, аритметично и геометрично усредняване. Ако две клетки са успоредни на посоката на потока, то се използва средна хармонична стойност на тези две клетки. Ако не са, се взима средно аритметично. В зависимост от начина на започване на процедурата се получават две различни стойности. Крайната апроксимация се получава чрез средно геометрично на тези две стойности.

3.1.2 Експерименти

Таблица **3.1** съдържа резултати от симулация на Многонивов Монте Карло с ренормализация. В таблицата са показани конструкции на алгоритъма на 2 и 3 нива. Като техника за приближение, опростената ренор-

мализация има заглаждащ ефект. Така полето, получено чрез опростена ренормализация, е подобно на оригиналното поле - двете полета имат сходна вариация. Това може да бъде потвърдено количествено чрез представените в таблицата данни. Получените експериментални резултати в показват:

- След ренормализация, вариацията на най-грубото ниво е близка то тази на оригиналното поле.
- Вариацията на корекциите спрямо оригиналното поле намалява много бързо с нарастване на нивата.
- Изчислената стойност на средния поток изпълнява отначало зададения толеранс $\epsilon = 3e - 3$ за двете конструкции на алгоритъма.
- От последната колона проличава, че докато класическият Монте Карло се нуждае от стотици хиляди реализации на ниво с дадена гранулярност, то Многонивовият се нуждае само от няколкостотин за съответното разбиване. При него основната част от изчисленията се извършват на най-грубото ниво, което в случая е 16 пъти помалко.

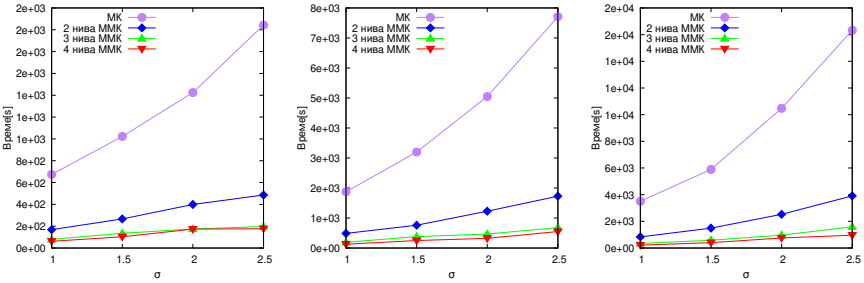
	$\lvert E[Q_{MLMC}] - E[Q_{MC}]\rvert$	$V[Y_l]$		Мрежа	N_l
МК	1.28937	$V[Y_0]$:	1.10483	$2^{10} \times 2^{10}$	122761
2н. ММК	1.30096	$V[Y_0]$:	1.17	$2^9 \times 2^9$	132229
		$V[Y_1]$:	3.36e−5	$2^{10} \times 2^{10}$	324
3н. ММК	1.29527	$V[Y_0]$:	1.26	$2^8 \times 2^8$	128315
		$V[Y_1]$:	8.89e−6	$2^9 \times 2^9$	205
		$V[Y_2]$:	9.82e−6	$2^{10} \times 2^{10}$	107

Таблица 3.1: Симулация с генериращи параметри на полето $\sigma = 2$, $\lambda = 0.3$ и толеранс $\epsilon = 3e - 3$

Предимството на Многонивовия Монте Карло спрямо класическия, е ясно изразено във времето на изчисление на двата подхода при един и същ толеранс $\epsilon = 0.003$, представени на фигура **3.1**.

3.2 Стохастично уравнение на конвекция-реакция-дифузия

Нека е дефинирана следната област $\bar{A} = (0, \bar{L})^2$ и в тази област е разгледано стационарното уравнение на конвекция-реакция-дифузия, описващо



(а) $\lambda = 0.1$ (б) $\lambda = 0.2$ (в) $\lambda = 0.3$

Фигура 3.1: Ускорение на Многонивов спрямо стандартен Монте Карло на 196 ядра

реактивен транспорт. Нека също така са дадени следните безразмерни величини:

$$x = \frac{\bar{x}}{\bar{L}}, \quad v = \frac{\bar{v}}{\bar{v}_{in}}, \quad t = \frac{\bar{\kappa}\bar{L}^2}{\bar{D}}, \quad C = \frac{\bar{C}}{\bar{C}_{in}}$$

Където, $\bar{v}_{in}$ е характерната скорост, $\bar{C}_{in}$ е характерната концентрация (началната концентрация в системата), $D = \bar{D}$ да бъде характерната стойност на дифузията, Pe е числото на Пекле, а Da - числото на Дамкохлер и означавайки с $\bar{\kappa}$ характерната скорост на реакция, то безразмерната форма на уравнението за конвекция-реакция-дифузия има следния вид при предварително зададени гранични условия:

$$-\nabla \cdot (\nabla C) + Pe(\omega)\nabla \cdot (v(x,\omega)C) + Da(\omega)C = 0, \ x \in (0,1)^2, \omega \in \Omega. \quad (3.4)$$

Тук безразмерните числа на *Пекле* и *Дамкохлер* в уравнение **3.4** са дефинирани по следния начин:

$$Pe(\omega) = \frac{\tilde{Pe}}{\phi(\omega)}, \quad Da(\omega) = \frac{\tilde{Da}}{\phi(\omega)} \quad (3.5)$$

Където $\tilde{Pe}, \tilde{Da}$ са предварително дефинирани, базирани на дифузията на нивото на порите, константи, а ϕ е порестоста. За този модел, разглежданите количества от интерес, пресметнати чрез Многонивов Монте Карло са потока и концентрацията на изходната граница на областта.

3.2.1 Конструкция на Многонивовия Монте Карло

В разгледания модел от уравнение **3.4** трите основни източника на неопределеност, които трябва да бъдат моделирани, са неопределеността в числата на *Пекле* и *Дамкохлер* и полето на скоростта. Подобно на случая с уравнението на Лаплас, нивата са дефинирани като размерността на разглежданата мрежа на дискретната задача. За приближение на полето на скоростта са разгледани два подхода:

- **"Решение преди ренормализация"** (SR): Уравнението **3.1**, описващо скоростта, първо се решава на най-финото ниво, след което се приближава на по-грубите нива със средно аритметично.
- **"Ренормализация преди решение"** (RS): Първо полето на пропускливост се генерира на финото ниво и се ренормализира до съответното по-грубо ниво, след което се решава уравнението **3.1**, описващо скоростта чрез налягането и закона на Дарси.

3.2.2 Експерименти

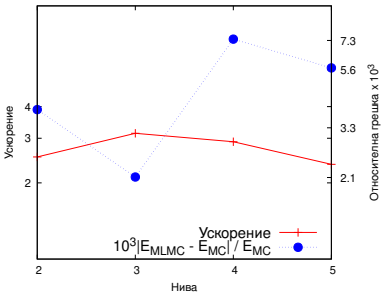
В таблица **3.2** е представен експеримент за ефективността на Многонивовия Монте Карло с подхода *SR* спрямо класическия Монте Карло метод.

$\sigma : 2$			$\lambda : 0.2$		
Пекле : 1.5			Дамкохлер : 0.5		
Ниво	Y_0	Y_1	Y_2	Y_3	Y_4
2	41502	607	-	-	-
3	43456	1536	536	-	-
4	47063	2809	1630	553	-
5	53093	5039	3084	1728	691

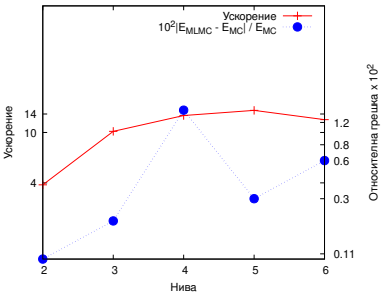
Таблица 3.2: Експерименти по нива; разглеждано количество - концентрация

С този подход за всяка реализация, независимо от нивото, уравнението за налягането се решава на най-финото ниво. Това ще доведе до по-малка разлика във времето за изчисление, между бързите пресмятания (пресмятанията на грубите нива), и бавните пресмятания (пресмятанията на финитe нива), което от своя страна ще намали скоростта на алгоритъма спрямо класическия подход. При този вид конструкция на Многонивовия Монте Карло се очаква максимална скорост на алгоритъма да бъде постигната при малък брой нива. Резултатите, изобразени на фигура **3.2** а), показват точно това - най-голямо ускорение се постига при три нива.

На фигура **3.2** b) е разгледан другият подход - RS. Тестовите параметри са: $\sigma = 2, \lambda = 0.2, \epsilon = 3e{-}2, Pe = 2.5$ и $Da = 0.5$. Използването на подхода с ренормализация преди решение дава значително по-добро бързодействие спрямо класическия Монте Карло.



(а) SR, Концентрация



(б) RS, Поток

Фигура 3.2: Ускорение на Многонивов и стандартен Монте Карло

4. Стратегии за паралелно пресмятане на Многонивов Монте Карло алгоритъм

4.1 Алгоритмична процедура за Многонивов Монте Карло

Както класическият Монте Карло, така и Многонивовият Монте Карло се основават на повтаряне на независими експерименти, генерирани от статистическо поле. След всяка генерация на конкретен експеримент, диференциалното уравнение става детерминистично и може да бъде решено с някой от класическите методи. Решението или тази част от него, която представлява интерес, се прибавя към статистиката на вече генерираните решения. След определен брой експерименти очакваната стойност се пресмята заедно със стоп критерия на алгоритъма - корен от средно квадратичната грешка. Ако грешката е в рамките на изискваното - процедурата се прекратява. Ако броя на експериментите не е достатъчен, то процедурата се повтаря отново.

Фигура 4.1: Блок схема на изчисление на Многонивов Монте Карло

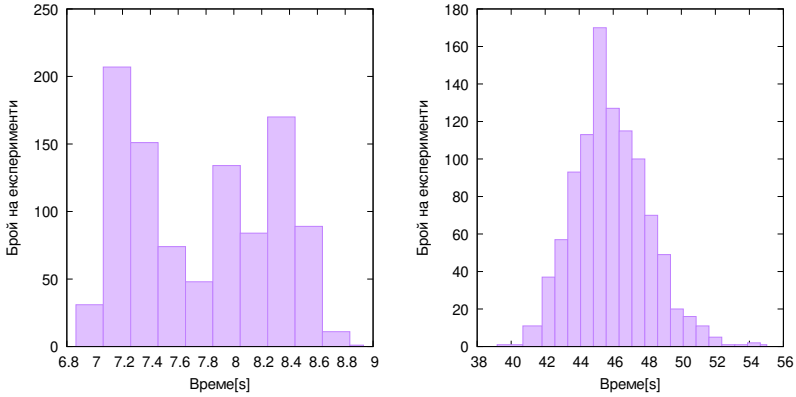
На всяка стъпка на оценка се изчислява броя на експериментите, които трябва да се проведат. Това става чрез уравнение 4.1, където v_l е експерименталната вариация на ниво l, t_l е очакваното време за изчисление на един експеримент на ниво l, а ϵ е търсеният толеранс.

$$N_l = \lceil \lambda \sqrt{(v_l/t_l)} \rceil \ where\ \lambda = \frac{1}{\epsilon^2} \sum_{l=0}^{L} \sqrt{(v_l/t_l)} \tag{4.1}$$

След всяка фаза на оценка на броя нужни експерименти започва изчислителна фаза. За всеки експеримент от всяко ниво се генерира съответното

стохастично поле, след което то се адаптира към стохастично диференциално уравнение за да се конструира детерминистично такова. Детерминистичното уравнение се решава и количеството представляващо интерес за модела се пресмята и прибавя към статистиката. След приключване на изчислителната фаза се прилага отново уравнение 4.1. Този процес се повтаря докато има още експерименти за пресмятане. Фигура **4.1** демонстрира тази идея. Всяка фаза на *оценка* е следвана от *изчислителна* фаза, като блокът от двете фази се повтаря докато изискването за точност на алгоритъма бъде спазено.

Многонивовият Монте Карло може да подобри значително скоростта на сходимост като има по-добра изчислителна сложност, сравнен с класическия Монте Карло. Въпреки това, симулациите изискват значителни изчислителни ресурси. Една симулация може да изисква милиони повторения на експерименти. Поради това, изпълнението на Многонивов Монте Карло за реални задачи на един процесор е нереалистично. Ето защо е необходимо разработването на паралелна стратегия за изчисление. Една такава ефективна стратегия трябва да взима предвид не само различните нива на паралелизъм в алгоритъма, но и отклоненията във времето за изчисление на експерименти от едно и също ниво. Тези отклоненията могат да бъдат значими, особено при полета имащи голяма вариация. Това личи от данните на фигура **4.2**



(а) Уравнение **3.1** с $\sigma = 2.0$ $\lambda = 0.2$ (б) Уравнение **3.4** с $\sigma = 2.0$ $\lambda = 0.2$ $Pe = 2.5$ $Da = 0.5$

Фигура 4.2: Хистограма на времето за провеждане на експеримент на мрежа $2^{10} \times 2^{10}$ за 1000 повторения

И за двете задачи, дадени на фигура **4.2**, разликата във времената за пресмятане на различните експерименти е значителна, което може сериозно да намали ефективността на паралелния алгоритъм. В случая на експеримент, проведен за уравнението на конвекция-реакция-дифузия, тази разлика може да достигне над 10 секунди. За големи задачи, решавани върху големи области, където един експеримент трябва да бъде решен на няколко процесора, поради ограничения на оперативната памет, паралелните стратегии ще зависят силно и от скалируемостта при един експеримент. Като основна цел, паралелната стратегия трябва да може да планира максимално количество експерименти за минимално време на дадено количество процесори. Така проблемът може да бъде формулиран като оптимизационна задача с поставени ограничения. За така дефинираната в най-обща форма задача, този проблем е NP пълен **[5]**. За постигане на ефективен паралелен алгоритъм се изисква опростявания и предположения за задачата. По построение Многонивовият Монте Карло алгоритъм дефинира три основни нива, на които може да бъдат извършвани паралелни изчисления:

- (i) Паралелизация на ниво експеримент (решаване на диференциалното уравнение).
- (ii) Паралелизация на експериментите от дадено ниво.
- (iii) Паралелизация при едновременно изчислението на всички нива на Многонивовия Монте Карло.

4.2 Параметри на ефективността и измерването ѝ

Дизайнът на паралелна стратегия изисква отчитането на набор от параметри. Най-значимите такива са борят на различните нива на алгоритъма, броят на нужните експерименти на дадено ниво, както и времето за изчисление на един експеримент на дадено ниво. В случая на експеримент, решаван на повече от един процесор, комуникацията при това изчисление ще допринесе към общото време на синхронизация. Ефективността на алгоритъма, използван за решаване на един експеримент, има важна роля за поведението на паралелната стратегия. Съвременните алгебрични мулти грид методи предлагат много добро скалиране **[5]**. При предположение, че няма загубва на изчислителен ресурс при паралелната работа, теоретично минималното време за изчисление може да бъде пресметнато по формулата:

$$T^{min} = \frac{1}{P^{all}} \sum_{l=0}^{L} N_l E[t_l] \tag{4.2}$$

където P^{all} е общият брой на налични изчислители (процесори или ядра), N_l е броят на експериментите на ниво l и $E[t_l]$ е очакваната стойност на времето за изчисление на един експеримент на ниво l. Когато един експеримент на едно ниво се решава от един изчислител, минималното време може да бъде пресметнато чрез проследяване на времената на изчисления по време на симулация. В случая на повече от един изчислител за даден експеримент, минималното време може само да бъде оценено. Нека θ е метрика, която измерва колко ефективен е паралелният изчислител за един експеримент. Тогава времето за изчисление на експеримента може да бъде изразено чрез:

$$C_l = \theta_l C_l^{min}, \tag{4.3}$$

където в уравнение **4.3** C_l^{min} е времето за изчисление на един експеримент на ниво l на P_{min}^l процесора. Преформулирайки уравнение **4.2** чрез замяна на t_l с C_l, уравнението приема вида:

$$T^{min} = \frac{1}{P^{all}} \sum_{l=0}^{L} N_l E[C_l] = \frac{1}{P^{all}} \sum_{l=0}^{L} N_l E[\theta_l C_l^{min}] = \frac{1}{P^{all}} \sum_{l=0}^{L} N_l \theta_l E[C_l^{min}] \tag{4.4}$$

В случая, когато $\theta_l = 1$ за всички нива, двете формулировки са еквивалентни. Когато един изчислител се използва за решението на един експеримент на ниво l трябва да бъде пресметнато θ_l. Това може да стане в подготвителна фаза на алгоритъма като се изчисли θ_l за всички стойности в рамките на даден прозорец $\{P_{min}^l, \ldots, P_{max}^l\}$, където P_{max}^l е максималният брой изчислители, които изпълняват отначало зададената ефективност за изчисление на експеримент на дадено ниво, а P_{min}^l е минималният брой изчислители, на които може да бъде решен проблема, предвид използването на наличната оперативна памет. Нека C_l^p е времето нужно за изчисление на един експеримент на едно ниво l на p изчислителя, тогава θ_l приема вида:

$$\theta_l = C_l^p / C_l^{min} \tag{4.5}$$

За да се определи ефективността на дадено ниво на Многонивовия Монте Карло алгоритъм, относителната загуба от паралелни изчисления се пресмята като времето за изчисление C_l^{comp} се извади T_l^{min} и резултата се съпостави на минималното време за изчисление на съответното ниво T_l^{min}. По този начин може да се изчисли какъв е процентът на загубеното време за синхронизация спрямо минималното изчислително време. Ако изчисленото време е близо до минималното време, стойността на тази фракция ще бъде близо до 1. Загубеното време за разпределение на процесорите и синхронизацията им, ще доведе до стойности, по-големи от 1. За да се постигне намаляваща функция, от резултата се изважда 1:

$$Eff_l = 1 - (C_l^{comp} - T_l^{min})/T_l^{min} \tag{4.6}$$

Изразявайки **4.6** чрез θ и C_l^p имаме:

$$Eff_l(\theta, p) = 1 - \frac{(C_l^{comp} - N_l\theta_l E[C_l^{min}])}{(N_l\theta_l E[C_l^{min}])} \tag{4.7}$$

Общата ефективност на Многонивовия Монте Карло се изчислява като сума от ефективностите на различните нива:

$$Eff(\theta) = 1 - \frac{(\sum_l C_l^{comp} - \sum_l N_l\theta_l E[C_l^{min}])}{\sum_l (N_l\theta_l E[C_l^{min}])} \tag{4.8}$$

където $l \in \{0, \dots L\}$.

4.3 Паралелни стратегии

Лесен, но не оптимален, начин за паралелизация е третирането на Многонимивия Монте Карло алгоритъм като колекция от независими Монте Карло алгоритми. Тогава всички налични процесори изчисляват ниво по ниво. Всички процесори първо пресмятат най-грубото ниво, след това следващото по сложност и така, докато не се премине през всички нива. На всяко ниво работата между процесорите се разпределя равномерно. Макар че такава стратегия е лесна за реализация, тя не води до оптимално ефективни изчисления. При комбиниране с други паралелни подходи се постигат много ефективни алгоритми. Фокус на тази работа са динамичните подходи. Те се опитват да подобрят разпределението на ресурсите чрез лакома схема.

Независимо от паралелната стратегия, след всяка изчислителна фаза оценъчната фаза фигура (4.1) се нуждае от информация от всички нива, тъй като уравнение **4.1** изисква информация за вариацията от всяко ниво за да пресметне коректно необходимия брой експерименти.

4.3.1 Динамична стратегия

При тази стратегия се използва лаком подход. Тя се приспособява по време на симулация и може да бъде комбинирана с други различни стратегии за пресмятане на класически Монте Карло алгоритъм при разпределение на ресурсите. Схемата включва всички паралелни нива на Многониводия алгоритъм.

Изискване към нея е наличността на времена на експерименти и вариации от всички нива. Тази информация може да бъде набавена по два начина: като пре-изчислителна стъпка или като първа стъпка от алгоритъма, пресмятайки един цикъл *Оценка-Изчисление* с неоптимално разпределение на процесорите и запаметяване на времената за изчисление. За простота тук, без ограничение на общността, е разгледан Многонивов Монте Карло на 3 нива, като пресмятанията за повече нива са същите. Нека $N_i, i = \{0, 1, 2\}$, е борят на изискваните експерименти по нива - $\widehat{Y_l}$, където N_0 е броят на експерименти на най-грубото ниво $\widehat{Y_0}$. Нека освен това p_i е борят на изчислителите, разпределени съответно за $\widehat{Y_i}$, $p^g_{l_i}$ е съответната група изчислители, който решават един експеримент на ниво l и t_i е времето за решаване на един експеримент на един изчислител. Нека също P^{total} е общият брой изчислители (ядра или процесори), които участват в решаване на задачата. При тези означения общото време за изчисление за един цикъл *Оценка-Изчисление* може да бъде пресметнат чрез:

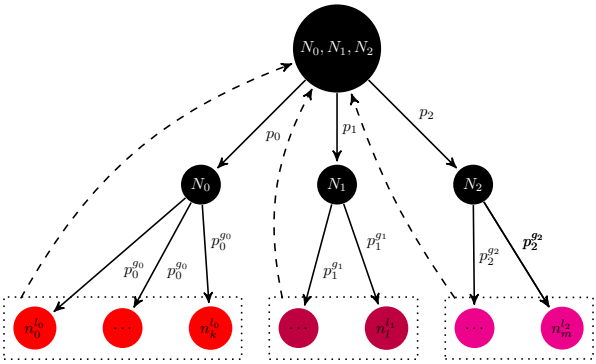
$$T^{total}_{CPU} = N_0 t_0 + N_1 t_1 + N_2 t_2 \tag{4.9}$$

Оптималното време за изчисление за всеки един изчислител е:

$$T^{p}_{CPU} = \frac{T^{total}_{CPU}}{P^{total}} \tag{4.10}$$

Чрез разделяне на общото време за изчисление на $\widehat{Y_i}$ на T^p_{CPU} може да бъде получена непрекъсната стойност за оптималния брой процесори на дадено ниво:

$$p^{ideal}_i := \frac{N_i t_i}{T^p_{CPU}} \text{ за } i = \{0, 1, 2\} \tag{4.11}$$



Фигура 4.3: Схематично разпределение на експериментите по нива и групи процеси

Нека предположим, че всички налични изчислители трябва да бъдат разпределени на всички нива $\widehat{Y}_i$. Чрез закръгляване надолу до цяло число на p_i^{ideal} може да бъде намерено разпределение за брой процесори по нива:

$$p_i := \left\lfloor p_i^{ideal} \right\rfloor, \text{ за } i = \{0, 1, 2\} \tag{4.12}$$

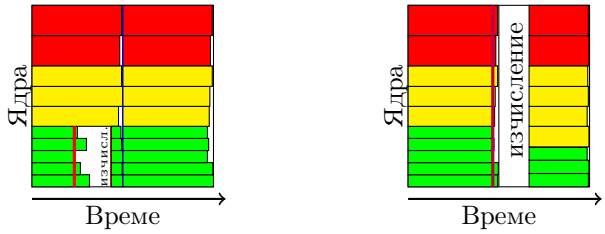
В зависимост от стратегията за паралелно изчисление на даден оценител $\widehat{Y}_i$ могат да бъдат наложени допълнителни ограничения за p_i. За да се подобри оценката и за да бъдат включени в изчислението всички ресурси, комбинацията от всички долни и горни граници на закръгление се претърсва за оптимално разпределение. Търси се целочислено решение в границите $\sum_{i=0}^{2} p_i$ и p^{total}.

До този момент конструкцията разглежда разпределение на всички процесори на всички нива едновременно. Това може да не е оптимално. Експериментите показват, че почти никога не е такова, поради силния дисбаланс между времето за изчисление на различните оценители Y_i. По тази причина различни комбинации от нива, изчислени паралелно, също трябва да бъдат проверени. Това подсигурява случаи, когато е по-добре първо всички изчислители да решават експерименти от оценител $\{\widehat{Y_0}\}$, след което оценители $\{\widehat{Y_1}, \widehat{Y_2}\}$ да бъдат изчислени паралелно.

4.3.2 Динамична стратегия с прекъсвания

Тази идея наподобява много идеята на процесорните прекъсвания. Стратегията започва като стандартна динамична стратегия. Прави се пред-

положението, че няма отклонения във времето на изчисление на два експеримента, или ако има те ще се балансират. По време на паралелните изчисления, поради неравномерно натоварване на изчислителите или поради отклонение между два експеримента на дадено ниво, дадена група изчислители или отделен изчислител приключва работа преди останалите. В такъв случай групата, която първа е приключила изчисления, изпраща сигнал до част или всички останали групи от изчислители, да спрат изчисленията като ги информира, че е приключила работа. При получаване на такъв сигнал, работещите груби преустановяват работа. След приключването на тази обмяна на съобщения, всички групи преминават в състояние на изчакване и се извършва преразпределение на изчислителите. За този тип схема са разгледани два типа прекъсвания. Първият е *локално прекъсване* - прекъсване в рамките на ниво. Вторият тип е *глобално* - изчислението се прекратява за всички групи на всички нива, за всички изчислители, след което се извършва преразпределение на ресурсите. Фигура **4.4** демонстрира идеята.



(а) Локално (на ниво)

(б) Глобално (на всички нива)

Фигура 4.4: Схематично представяне на двата вида прекъсвания. Разгледаните цветове показват различно ниво.

На практика сигналите се моделират като обмяна на съобщения между различните групи процеси в MPI като “master-slave” подход. Дадена група изпраща съобщение до предварително определената “master” група. Тази “master” група поема отговорност да информира останалите групи да прекратят работа и да организира последващата синхронизация между тях. Всяко едно от съобщенията носи малко система информация и така доминиращ фактор за скоростта на прекъсването е латентността на системата.

4.3.3 Динамичен опашков паралелизъм

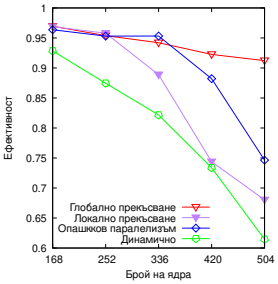
Този подход симулира идеята за опашков паралелизъм или задачов паралелизъм в многонишкова среда за програмиране, пригодена към особеностите на интерфейса за обмяна на съобщения "MPI". Първо се намира оптимално разпределение на процесорите чрез уравнение **4.11**. За всеки оценител $\widehat{Y}_l$, чрез "master-slave" програмна парадигма се определя "master" изчислител. Той играе ролята на диспечер, който раздава работа на останалите. Всеки изпълнява определената му задача и докладва обратно до "master" процеса за да получи нова порция работа. По този начин всеки процес работи независимо от времето за изчисление на един експеримент. Това е за сметка на обмяната на голямо количество съобщения. За сравнение, при стратегията на прекъсванията има голямо количество на съобщения в даден момент от време, докато при този подход съобщенията са значително повече, но са разпределени във времето.

4.3.4 Експерименти

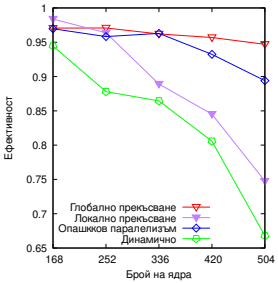
Експерименти за уравнението на Лаплас

На фигура **4.5** са поместени резултатите от експеримент, тестващ ефективността на различните паралелни подходи. Тя е изчислена чрез уравнение **4.8**. В тези експерименти, размерът на фината мрежа е: $2^{10} \times 2^{10}$ брой клетки. Това е приблизително 10^6 неизвестни. Броят на нивата на Многонивовия Монте Карло алгоритъм е определен на 4. На **4.5 a)** е представена паралелната ефективност при параметри $\sigma = 2,\ \lambda = 0.3,\ \epsilon = 1e-3$. На **4.5 b)** е представен по-тежък изчислителен проблем с параметри на генериране $\sigma = 2.25,\ \lambda = 0.4,\ \epsilon = 1e-3$. И за двата случая всеки експеримент се изчислява от точно едно процесорно ядро. На фигура **4.5 c)** е представен същия проблем като **4.5 b)**, но с различен брой процесори за всеки експеримент. По нива, един експеримент, се решава съответно от: $p_0 = 1,\ p_1 = 5,\ p_2 = 9,\ p_3 = 11$ процесорни ядра. За това разпределение на процесорите, функцията на ефективност има вида $Eff(1,\ 3.6,\ 7.35,\ 9.00)$.

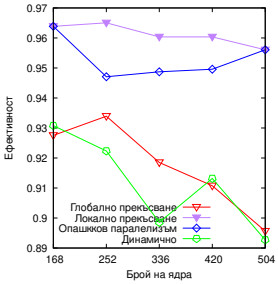
Фигура **4.5 a) b)** показва, че за сравнително малко на брой експерименти локалното прекъсване постига най-добри резултати. Тези резултати не се запазват при увеличаване на броя процеси. В случая на 504 ядра, ефективността е само 8% по-висока от чисто динамичния подход. Има две основни причини за това. Първата е по-големият брой процеси, който води до по-лош баланс на работата в различните нива. Дори алгоритъмът да намери добро разпределение на изчислителите, при което времето за изчакване е малко, това време е много по-значимо при голям брой изчислители в сравнено с малък брой, тъй като сумарното време на изчакване



(а) $\sigma = 2.0$ $\lambda = 0.3$
1 процес за всеки експеримент



(б) $\sigma = 2.25$ $\lambda = 0.4$
1 процес за всеки експеримент



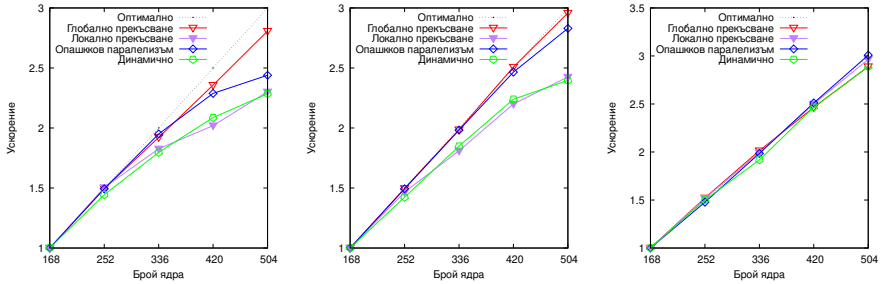
(в) $\sigma = 2.25$ $\lambda = 0.4$
p_0=1, p_1=5, p_2=9, p_3=11

Фигура 4.5: Ефективност за различните стратегии за уравнението на Лаплас

нараства заедно с броя изчислители. Втората причина е големият брой съобщения, които трябва да бъдат обменени в малък интервал от време. Това води до наводняване на комуникационния канал на мрежата. Този проблем е проличава ясно, при сравнението на опашковия метод с локалния. Макар че при първия има много повече съобщения, те се обменят в продължителен период от време и всички са с малък обем системна информация. Това води до по-равномерно натоварване на комуникационния канал и съответно по-добра ефективност. Най-добрият метод, сред разгледаните при разпределяне на едно ядро за всеки експеримент, е техниката на глобално прекъсване. Тестът показва слабо намаляване на ефективността при нарастване на броят на процесорите. Фигура **4.5 с)** показва използване на всички нива на паралелизъм. Това води до много ефективни алгоритми. Всички разглеждани подходи постигат повече от 0.89% от теоретично-оптималната ефективност. В този случай локалното прекъсване превъзхожда останалите. Подобрената ефективност идва от факта, че общото време за провеждане на един експеримент намалява и съответно - дисбаланса на работа между нивата. Фигура **4.6** показва относителната скалируемност на алгоритмите.

Експерименти за уравнението на конвекция-реакция-дифузия

На фигура **4.7** ефективността е измерена чрез уравнение **4.8** за трите различни динамични стратегии: *локално прекъсване, глобално прекъсване, опашков паралелизъм.* Всеки експеримент е решен на едно ядро. Тестовите параметри са: $\sigma = 2,\ \lambda = 0.2,\ \epsilon = 1e-2,\ Pe = 2.5,\ Da = 0.5$. Фи-



(а) Експеримент 4.5 a) (б) Експеримент 4.5 b) (в) Експеримент 4.5 c)

Фигура 4.6: Силна скалируемост за експериментите от фигура **4.5**

ната мрежа е с размер: $2^{10} \times 2^{10}$, броят нива е 4. Използва се *RS* подхода при пресмятане на потока. Резултатите показват много добра ефективност за подхода с *локално прекъсване*. Този подход превъзхожда другите два метода. Ефективността на тази стратегия значително намалява при по-висок брой ядра. Същевременно, при глобалното прекъсване и опашковият паралелизъм загубата на ефективност е по-малка. Средният брой експерименти по нива за този тест е показан в таблица **4.1**

# Ниво	0	1	2	3
# Локално прекъсване	444975	9466	3382	991
# Глобално прекъсване	447649	9571	3359	962
# Опашков паралелизъм	449734	9398	3289	999

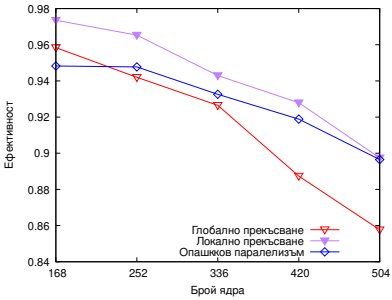
Таблица 4.1: Направени експерименти за конвекция-реакция-дифузия

Експерименти на клъстера SuperMUC за уравнението на Лаплас

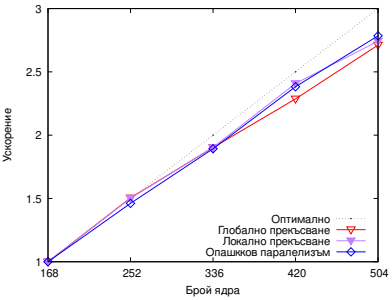
Експериментите в тази секция са направени на изчислителния клъстер *SuperMuc* в техническия университет в Мюнхен (TUM). Всеки възел се състои от 48 ядра. Общата памет на възел е $96GB$.[1]

Тестът, с резултати показани на на фигура **4.8**, е предназначен да до-

[1] https://doku.lrz.de/display/PUBLIC/Hardware+of+SuperMUC-NG



(а) $Eff(1,1,1,1)$



(б) Скалируемост

Фигура 4.7: Ефикасност и скалируемост

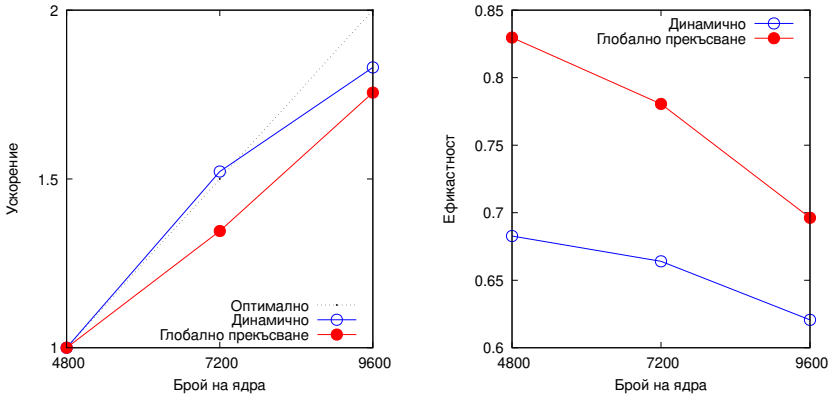
каже ефекта от глобалната оптимизация на прекъсванията за динамичния подход при голям брой ядра върху относително малък спрямо броя ядра проблем. В такъв случай времето, загубено за синхронизация, ще има по-значително влияние върху общото изчислително време и съответно ефективността. Параметрите на теста са обобщени в таблица **4.2**. За приближение на стохастичното поле е използвана опростена ренор-

Максимална мрежа: $2^{10} \times 2^{10}$	$\sigma = 3$	λ=0.3		ϵ= 1e-3	
Брой ядра по нива		Lvl. 0	Lvl. 1	Lvl. 2	Lvl. 3
		1	1	1	1

Таблица 4.2: Симулационни параметри за фигура **4.8**

мализация. Тестът показва добро разпределение на процесорите между различните нива в случая с 4800 ядра и доказва ефективността на динамичния подход за такъв сценарий. Постигнатата ефективност е близо до 70%. Използването на оптимизация с глобално прекъсване води до значително подобрение на ефективността - тя достига близо до 85%. В случай на 7200 ядра използването на тази оптимизация води до около 12% по-добра производителност в сравнение със случая без оптимизация. При 9600 ядра предимството е около 7%. Този постоянен спад се дължи на увеличеното време за комуникация и като цяло по-малкото изчислително време. В случай на 9600 ядра, общото време на симулация е само 621 секунди при използване на глобалната стратегия за прекъсване. При

по-тежките изчислителни симулации се очаква по-висока ефективност.



(а) Силна скалируемост (б) Ефикасност

Фигура 4.8: Голям брой ядра, 1 ядро на експеримент

5. Библиография

[1] M.B. Giles. “Multilevel Monte Carlo Methods”. В: *Acta Numerica* (2018). DOI: 10.1017/S09624929.

[2] S. Heinrich. “Monte Carlo complexity of global solution of integral equations.” В: *Journal of Complexity* 14 (1998), с. 151—175. DOI: 10.1006/jcom.1998.0471.

[3] S. Heinrich и E. Sindambiwe. “Monte Carlo complexity of parametric integration.” В: *Journal of Complexity* 15 (1999), с. 317—341. DOI: 10.1006/jcom.1999.0508.

[4] S. Heinrich. “The multilevel method of dependent tests.” В: *Advances in Stochastic Simulation Methods* (2000), с. 47—61. DOI: 10.1007/978-1-4612-1318-5_4.

[5] D. Drziga и др. “SCHEDULING MASSIVELY PARALLEL MULTIGRID FOR MULTILEVEL MONTE CARLO METHODS”. В: *SIAM J. SCI. COMPUT* 39.5 (2017), S873—S897. DOI: 10.1137/16M1083591.

[6] Wolfgang Betz, Iason Papaioannou и Daniel Straub. “Numerical methods for the discretization of random fields by means of the Karhunen-Loève expansion”. В: *Computer Methods in Applied Mechanics and Engineering* 271 (2014), с. 109—129. DOI: doi:10.1016/j.cma.2013.12.010.

[7] I. G. Graham и др. “Analysis of circulant embedding methods for sampling stationary random fields”. В: *SIAM Journal on Numerical Analysis* 56 (2018). DOI: https://doi.org/10.1137/17M1149730.

[8] Francisco Cuevas, Emilio Porcu и Denis Allard. *Fast and exact simulation of isotropic Gaussian random fields on $\mathbb{S}^2$ and $\mathbb{S}^2 \times \mathbb{R}$*. 2018. eprint: arXiv:1807.04145.

[9] Walker DW. *Standards for message-passing in a distributed memory environment*. 1992. URL: https://www.osti.gov/biblio/7104668 (дата на посещ. 01.05.2021).

[10] Bungartz Hans-Joachim, Neumann Philipp и Nagel Wolfgang. “Software for Exascale Computing - SPPEXA 2013-2015”. В: (2016). DOI: 10.1007/978-3-319-40528-5.

[11] Bastian P и др. “A Generic Grid Interface for Parallel and Adaptive Scientific Computing. Part I: Abstract Framework.” В: *Computing* 103–119 (2008). DOI: 10.1007/s00607-008-0003-x.

[12] *Dune Numerics*. URL: https://dune-project.org (дата на посещ. 01.05.2021).

[13] E.Powell. *Numerical Methods for Generating Realisations of Gaussian Random Fields*. URL: www.maths.manchester.ac.uk/~cp (дата на посещ. 01.05.2021).

[14] Gabriel Lord, E.Powell и Tony Shardow. *An introduction to Computational Stochastic PDEs*. Cambridge University Press, 2014. ISBN: 978-0521728522. DOI: 10.1017/CBO9781139017329.

[15] Robert Gould и Colleen Ryan. *Introductory Statistics*. Pearson, 2016. ISBN: 978-0321978271.

[16] Joseph F. Hair. *Multivariate Data Analysis – A Global Perspective, 7th Edition*. Pearson Education, 2010. ISBN: 9780135153093.

[17] Randall LeVeque. *Numerical Methods for Conservation Laws*. Birkhauser-Verlag, 1990. ISBN: 978-3-0348-8629-1.

[18] Randall LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002. DOI: 10.1017/CBO9780511791253.

[19] J. Dick, F. Kuo и I. Sloan. "High-dimensional integration: The quasi-Monte Carlo way." В: *Acta Numerica* (2013). DOI: 10.1017/s0962492913000044.

[20] Jack Dvorkin. *Kozeny-Carman equation revisited*. 2009. URL: https://pangea.stanford.edu/~jack/KC_2009_JD.pdf (дата на посещ. 01.05.2021).

[21] Renard P. и De Marsily G. "Calculating equivalent permeability: a review." В: *Adv. Water Resour.* 20 (1997), с. 253—278. DOI: 10.1016/S0309-1708(96)00050-4.

[22] Wen X.H. и Gomez-Hern ández J.J. "Upscaling hydraulic conductivities in heterogeneous media: an overview." В: *Journal of Hydrology* 183 (1996), с. ix—xxxii. DOI: 10.1016/S0022-1694(96)80030-8.

[23] Ivan Lunati и др. "A numerical comparison between two upscaling techniques: non-local inverse based scaling and simplified renormalization." В: *Advances in Watter Resources* 24 (2001), с. 913—929. DOI: 10.1016/S0309-1708(01)00008-2.

Научни приноси на автора

Научни приноси на автора

- Научни приноси
  - Направен е преглед, обзор и анализ на съществуващите решение на поставените проблеми. Оценени са предимствата и недостатъците на съществуващите решения за генериране на стохастични полета и семплиращи алгоритми;
  - Сравнени и анализирани са различни подходи за приближение на стохастичното ниво за задачата на Лаплас;
  - Разработен е ефективен метод за ренормализация на стохастичното поле за целите на Многонивовия Монте Карло;
  - За пръв път успешно е приложен Многонивовият Монте Карло метод за уравнението на конвекция-реакция-дифузия;
  - Разработен е адаптивен алгоритъм за разпределение на ресурсите върху различните нива на Многонивовия Монте Карло алгоритъм.
- Научно-приложни приноси
  - Дефиниран е подход за определяне на нивата за Многонивовия Монте Карло за двете разгледани задачи;
  - Направен е анализ и сравнение на двата подхода за конструкция (coarse grain) на Многонивов Монте Карло спрямо класическия Монте Карло за задачата на конвекция-реакция-дифузия;
  - Направен е анализ и сравнение между скоростта на сходимост и времето за изчисление на Многонивов Монте Карло метод с опростена ренормализация и класически Монте Карло;
  - Направен е преглед, анализ и сравнение на 6 паралелни стратегии за паралелизация.
- Приложни приноси
  - Направена е програмна реализация на описаната стратегия за генериране на случайни полета с използване на графични ускорители;
  - Реализирани и сравнени са четири от предложените алгоритми;

– Приложимостта на разгледаните подходи за реалистични симулации на много ядра е потвърдена с реални тестове.

Списък с публикации

- **[1]** Iliev, O., Mohring, J., **Shegunov, N.**, *Renormalization Based MLMC Method for Scalar Elliptic SPDE*, International Conference on Large-Scale Scientific Computing, pp.295-303, 2017, Springer, ISSN: 0302-9743, SJR (2017) - 0.295
- **[2] Shegunov, N.**, Armianov, P., Semerdjiev, A., Iliev, O., *GPU accelerated Monte Carlo sampling for SPDEs*, 2019, Conf. Proc. of the 12th ISGT 2018, ISSN:1613-0073, SJR (2019) - 0.177
- **[3]** Zakharov, P., Iliev, O., Mohring, J., **Shegunov, N.**, *Parallel Multilevel Monte Carlo Algorithms for Elliptic PDEs with Random Coefficients*, International Conference on Large-Scale Scientific Computing, pp.463-472, 2019, Springer, ISSN: 0302-9743, SJR (2019) - 0.427
- **[4]** Bastian, P., Altenbernd, M., Dreier, N., Engwer, Ch., Fahlke, J., Fritze, R., Geveler, M., Göddeke, D., Iliev, O., Ippisch, O., Mohring, J., Müthing, S., Ohlberger, M., Ribbrock, D., **Shegunov, N.**, Turek, S., *Exa-Dune: Flexible PDE Solvers, Numerical Methods and Applications, Software for Exascale Computing-SPPEXA, 2016-2019*, pp. 225-269, 2020, https://doi.org/10.1007/978-3-030-47956-5_9, Springer
- **[5] Shegunov, N.**, Iliev, O., *On Dynamic Parallelization of Multilevel Monte Carlo Algorithm, Cybernetics and Information Technologies*, Volume 20, No 6, pp. 116-125, 2020, Journal Sciendo Print ISSN: 1311-9702, Online ISSN: 1314-4081, SJR (2020) - 0.310

Благодарности

Бих искал да благодаря на моя ръководител доц. Петър Армянов и на моя научен консултант професор Олег Илиев за тяхната подкрепа през цялото време на моята докторска работа. Освен това бих искал да благодаря на Fraunhofer ITWM и на техническия университет в Мюнхен за достъпа до техните високо-производителни изчислителни системи - Beehive и SuperMuc.

Декларация за оригиналност

Декларирам, че настоящият дисертационен труд съдържа оригинални резултати, получени при проведените от мен научни изследвания (с подкрепата и съдействието на научния ми ръководител и всичките ми съавтори). Резултатите, които са получени, описани и публикувани от други учени, са надлежно и подробно цитирани в библиографията. Настоящата работа не е прилагана за придобиване на научна степен в друго висше училище, университет или научен институт.

Декларатор: Nikolay Shegunov Digitally signed by Nikolay Shegunov Date: 2021.06.07 10:52:15 +03'00'