|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Робототехники и комплексной автоматизации

КАФЕДРА Системы автоматизированного проектирования (РК-6)

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

по дисциплине: «Параллельные методы и алгоритмы»

|  |  |  |
| --- | --- | --- |
| Студент |  | Макаров Тимофей Геннадьевич |
| Группа |  | РК6-22М |
| Тип задания |  | Лабораторная работа |
| Тема |  | Исследование эффективности динамической равномерной балансировки загрузки МВС с использованием имитационного моделирования |

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Макаров Т.Г.**

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Карпенко А.П.**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Москва, 2023 г.*

Оглавление

[Оглавление 2](#_Toc133406909)

[Цель лабораторной работы 3](#_Toc133406910)

[Постановка задачи 3](#_Toc133406911)

[Схема метода балансировки нагрузки 5](#_Toc133406912)

[Вычислительный эксперимент 7](#_Toc133406913)

[Программная реализация 10](#_Toc133406914)

[Результат работы программы 11](#_Toc133406915)

[Заключение 16](#_Toc133406916)

[Список использованных источников 16](#_Toc133406917)

Цель лабораторной работы

**Цель выполнения лабораторной работы** – изучение метода динамической балансировки загрузки многопроцессорной вычислительной системы (МВС) на основе равномерной декомпозиции узлов расчетной сетки и исследование его эффективности с помощью имитационного моделирования.

Постановка задачи

Пусть – n-мерный вектор параметров задачи. Положим, что , где – n-мерное арифметическое пространство. Параллелепипедом допустимых значенийвектора параметров назовем не пустой параллелепипед *П*, где - заданные константы. На вектор *X* дополнительно наложено некоторое количество функциональных ограничений, формирующих множество , где - непрерывные ограничивающие функции. На множестве тем или иным способом (аналитически или алгоритмически) определена вектор-функция  со значениями в пространстве . Ставится задача поиска значения некоторого функционала Ф(F(X)).

Положим, что приближенное решение поставленной задачи может быть найдено по следующей схеме.

*Шаг 1.* Покрываем параллелепипед *П* некоторой сеткой (равномерной или неравномерной, детерминированной или случайной) с узлами .

*Шаг 2.* В тех узлах сетки , которые принадлежат множеству , вычисляем значения вектор функции .

*Шаг 3*. На основе вычисленных значений вектор функции  находим приближенное значение функционала .

Введём следующие обозначения: – суммарная вычислительная сложность ограничений, формирующих множество , ; – вычислительная сложность вектор-функции , – вычислительная сложность генерации сетки ; – вычислительная сложность конечномерной аппроксимации функционала . Здесь - заданное ограничение сверху на величину , - общее количество узлов сетки , принадлежащих множеству .

В качестве вычислительной системы рассмотрим однородную МВС с распределенной памятью, состоящую из процессоров и *host*-процессора, имеющих следующие параметры:

* – длина вещественного числа в байтах;
* – время выполнения одной арифметической операции с плавающей запятой;
* – латентность коммуникационной сети;
* – время передачи байта данных между двумя соседними процессорами системы без учета времени ;
* – диаметр коммуникационной сети.

В качестве меры эффективности параллельных вычислений используем ускорение

где – время последовательного решения задачи на одном процессоре системы, – время параллельного решения той же задачи на *N* процессорах.

Схема метода балансировки нагрузки

Положим, что из числа узлов расчетной сетки множеству принадлежит узлов. Разобьем эти узлы на непересекающихся подмножеств и для простоты записи примем, что величины, кратны, так что каждое из подмножеств содержит по узлов. Совокупность подмножеств обозначим . Правило построение совокупности подмножеств иллюстрирует рисунок 1.



Рисунок 1. Схема построения совокупности .

Заметим, что при каждое из подмножеств содержит по одному узлу сетки ; при и , когда , динамическая равномерная балансировка вырождается в статическую.

Схема параллельного решения поставленной задачи с использованием динамической равномерной балансировки загрузки имеет следующий вид.

*Шаг 1.* Host-процессор выполняет следующие действия:

* строит сетку ;
* среди всех узлов сетки выделяет узлы ;
* разбивает эти узлы на  подмножеств ;
* посылает процессору координаты узлов первого из нераспределенных подмножеств .

*Шаг 2.* Процессор выполняет следующие действия:

* принимает от host-процессора координаты узлов подмножества ;
* вычисляет во всех узлах подмножества значения вектор-функции ;
* посылает host-процессору вычисленные значения .

*Шаг 3.* Если исчерпана не вся совокупность подмножеств , то host-процессор посылает, а процессор принимает координаты следующего подмножества из указанной совокупности узлов, которое обрабатывается процессором аналогично шагу 3 и т.д.

*Шаг 5.* Если исчерпаны все подмножества совокупности , то host-процессор:

* посылает освободившемуся процессору сообщение об окончании решения задачи;
* после получения всех вычисленных значений вектор-функции от всех процессоров вычисляет приближенное значение функционала Ф(F(X)).

Вычислительный эксперимент

Рассмотрим двумерную задачу (n=2). Параллелепипед *П* в этом случае представляет собой прямоугольник *П*. Положим, что , , так что область *П* является единичным квадратом (рисунок 2).

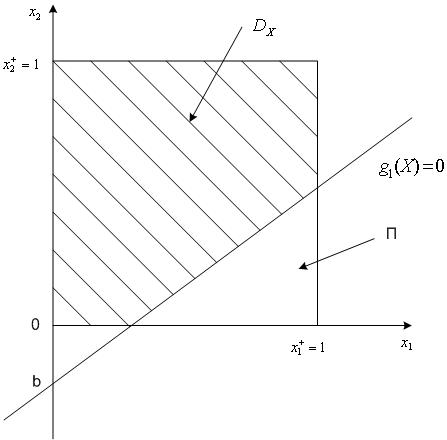


Рисунок 2 – Расчетная область задачи.

Множество *D* формируется с помощью одной ограничивающей функции , т.е. . Примем, что эта функция линейна и проходит через заданную преподавателем точку плоскости с координатами (0, b). Таким образом, уравнение этой функции имеет вид (при этом, очевидно, ).

В соответствии с номером варианта заданы значения параметров ограничивающей функции: . Количество узлов сетки, удовлетворяющих ограничивающей функции, равно 49152 (для сетки ). Полученное множество показано на рисунке 2.

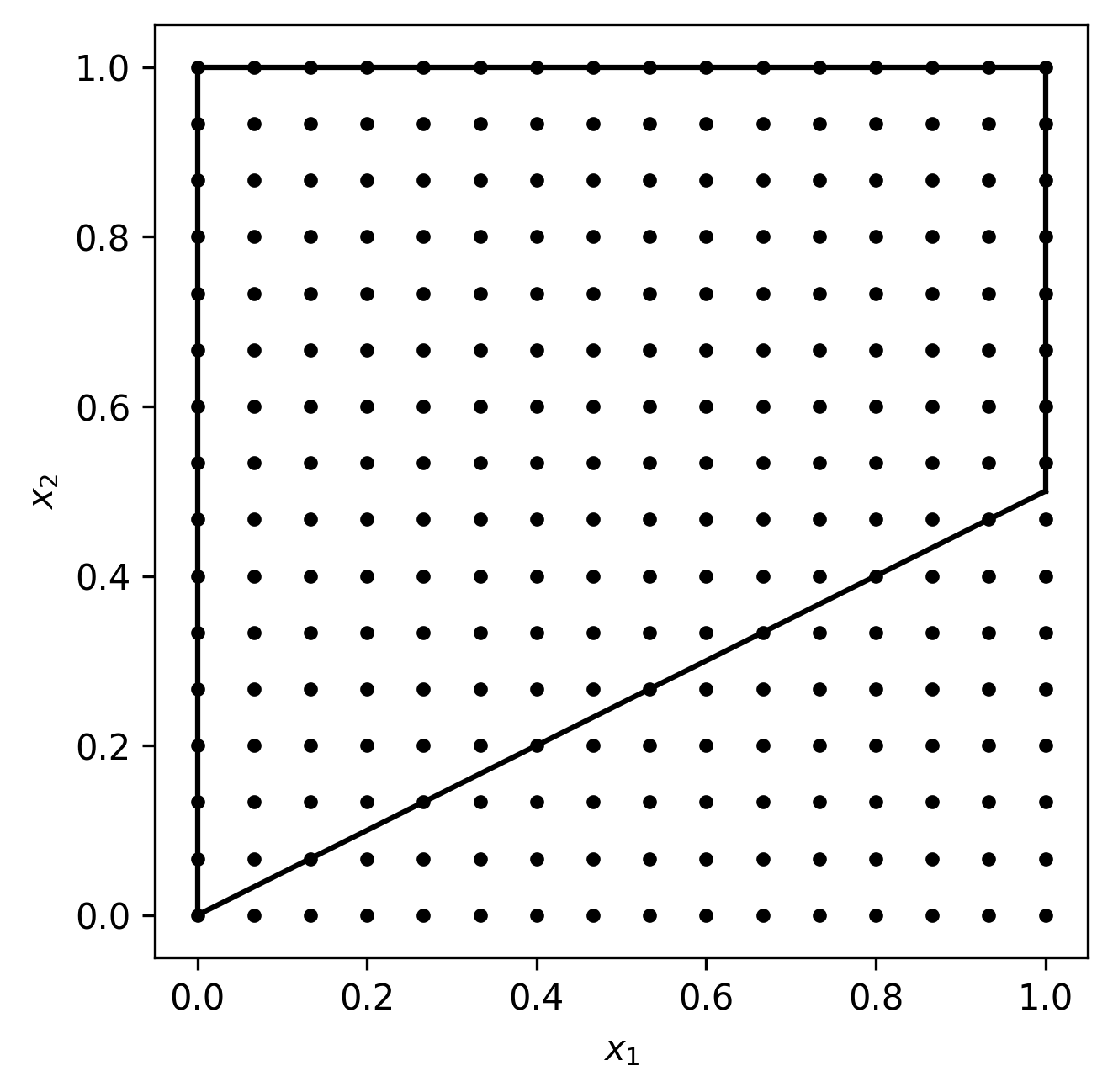


Рисунок 2 – Сетка и множество . Границы множества обозначены чёрной линией.

Будем исходить из следующих значений параметров задачи и МВС:

* ;
* ;
* ;
* ;
* ;
* ;
* .5;
* ;
* ;
* .

Пренебрежем вычислительными затратами на построение сетки , на вычисление значений ограничивающей функции 𝑔1(𝑋), а также на построение приближенного значения функционала Ф(𝐹(𝑋)), т.е. положим 𝐶𝛺 = 0, 𝐶𝑔 = 0, 𝐶Ф = 0. Положим также, что вычислительная сложность вектор-функции есть случайная величина, равномерно распределенная на интервале .

По результатам *r* «прогонов» оценка математического ожидания ускорении вычисляется по формуле

а оценка дисперсии этого ускорения и оценка его среднего квадратичного ускорения – по формулам

соответственно. Здесь, - оценка ускорения вычислений, полученная в *i*-ом «прогоне».

Программная реализация

Для написания программной реализации был использован язык GPSS и среда разработки GPSS World. Код программы приведён в листинге 1.

Листинг 1. Программная реализация.

N\_proc EQU 2

points\_N EQU 45056; кратно K = k \* N

t\_s EQU 50e-6

m\_s EQU 100

l\_s EQU 8

t\_c EQU 0.125e-7

N\_gr EQU 2

d\_s EQU SQR(N\_proc)-1

k\_1 EQU 4; (K = k \* N)

s\_1 EQU (points\_N/N\_proc)/k\_1;

T\_ik EQU 2#t\_s+s\_1#N\_gr#l\_s#d\_s#t\_c+s\_1#m\_s#l\_s#d\_s#t\_c; 0.007 N = 256

t EQU 1e-8

uniform\_cf\_par FUNCTION rn2,c2

0,0.0/1,2e4

uniform\_cf\_posl FUNCTION rn3,c2

0,0.0/1,2e4

proc\_par STORAGE 2

proc\_posl STORAGE 1

us VARIABLE p4/p3

tabl\_s TABLE v$us,42,0.25,60

generate 1e8,100

split (N\_proc - 1)

queue qhost1\_par

seize host

depart qhost1\_par

advance 5e-6,3e-6

release host

assign 1,s\_1

assign 5,k\_1

queue qproc\_par

enter proc\_par

depart qproc\_par;

proc3 advance T\_ik,1e-8

proc2 advance t,fn$uniform\_cf\_par

loop 1,proc2;

assign 1,s\_1

loop 5,proc3

leave proc\_par

queue qhost2\_par

seize host

depart qhost2\_par

advance 5e-6,3e-6

release host

assemble (N\_proc)

assign 3,m1

\* последовательная обработка

mark 2

split (points\_N - 1)

queue qhost1\_posl

seize host

depart qhost1\_posl

advance 5e-6,3e-6

release host

queue qproc\_posl

enter proc\_posl

depart qproc\_posl

advance t,fn$uniform\_cf\_posl

leave proc\_posl

queue qhost2

seize host

depart qhost2

advance 5e-6,3e-6

release host

assemble points\_N

assign 4,mp2

tabulate tabl\_s

TERMINATE 1

START 50

Результат работы программы

В таблице 1 представлены полученные с помощью написанной программы результаты вычисления ускорений для различных значений вычислительной сложности и числа процессоров.

Таблица 1. Оценка ускорения

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 2 |  |  |
| 4 |  |  |
| 8 |  |  |
| 16 |  |  |
| 32 |  |  |
| 64 |  |  |
| 128 |  |  |
| 256 |  |  |

На рисунке 4 представлен график оценки ускорения при статической балансировке загрузки методом равномерной декомпозиции узлов. На рисунке 5 представлен аналогичный график, полученный средствами имитационного моделирования. На обоих графиках при больших значениях вычислительной сложности наблюдаем большее ускорение. Это объясняется тем, что при увеличении вычислительной сложности относительный вклад коммуникационной загрузки уменьшается и ускорение от распараллеливания возрастает.

При оценке ускорения с использованием имитационного моделирования наблюдается снижение ускорения по сравнению с результатом лабораторной работы №2. Это объясняется тем, что в лабораторной работе №2 вычислительная сложность задавалась ее максимальным значением, а в программе на языке GPSS равномерно распределенной случайной величиной.

Рисунок 3 – Оценка ускорения при статической балансировке загрузки методом равномерной декомпозиции расчетных узлов, полученная с помощью программы на GPSS.

В таблице 2 представлены полученные значения оценки математического ожидания и оценки среднего квадратичного отклонения для разного числа процессоров при заданных значениях вычислительной сложности.

Таблица 2. Оценка математического ожидания и среднего квадратичного отклонения ускорения при динамической балансировке загрузки

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | |  | |
|  |  |  |  |
| 2 |  |  |  |  |
| 4 |  |  |  |  |
| 8 |  |  |  |  |
| 16 |  |  |  |  |
| 32 |  |  |  |  |
| 64 |  |  |  |  |
| 128 |  |  |  |  |
| 256 |  |  |  |  |

На рисунках 6 и 7 представлены графики оценки математического ожидания и среднего квадратичного отклонения для разного числа процессоров при заданных значениях вычислительной сложности.

Рисунок 4 – Оценка математического ожидания при динамической балансировке загрузки методом равномерной декомпозиции расчетных узлов, полученная с помощью программы на GPSS.

Рисунок 5 – Оценка среднего квадратичного отклонения при динамической балансировке загрузки методом равномерной декомпозиции расчетных узлов, полученная с помощью программы на GPSS.

По графику оценки среднего квадратичного отклонения видно, что разброс значений ускорения при увеличении вычислительной сложности возрастает. Это объясняется тем, что вычислительная сложность задается, как равномерно распределенная величина в диапазоне . При увеличении , диапазон увеличивается, вследствие этого увеличивается и разброс значений ускорения.

Заключение

Изучен метод динамической равномерной балансировки загрузки МВС. Исследована эффективность метода с использованием имитационного моделирования. Эффективность метода исследована с использованием имитационного моделирования.

Список использованных источников

1. Карпенко А.П., Федорук Е.В. Параллельные вычисления. Учебнометодическое пособие к лабораторным работам по курсу «Параллельные вычисления». М.: НУК ИУ МГТУ им. Н.Э. Баумана, 2010. 72 с.
2. Карпенко, А.П. Параллельные вычисления: учебное пособие [Электронный ресурс] / А.П.Карпенко // (<http://bigor.bmstu.ru/?cnt/?doc=Parallel/base.cou>).