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 статической балансировки загрузки МВС с помощью 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

Цель лабораторной работы

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

Постановка задачи

Пусть X – n-мерный вектор параметров задачи. Положим, что $X \in R^n$, где R^n – n-мерное арифметическое пространство. Параллелепипедом допустимых значений вектора параметров назовем не пустой параллелепипед $П = \{X \mid x_i^- \le x_i \le x_i^+, i \in [1:n]\}$, где x_i^-, x_i^+ - заданные константы. На вектор X дополнительно наложено некоторое количество функциональных ограничений, формирующих множество $D = \{X \mid g_j(X) \ge 0, j = 1,2,\dots\}$, где $g_j(X)$ - непрерывные ограничивающие функции. На множестве $D_X = П \cap D$ тем или иным способом (аналитически или алгоритмически) определена вектор-функция $F(X)$ со значениями в пространстве R^m. 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 Ω (равномерной или неравномерной, детерминированной или случайной) с узлами $X_1, X_2, \dots, X_z$.

Шаг 2. В тех узлах сетки Ω, которые принадлежат множеству D_X, вычисляем значения вектор функции $F(X)$.

Шаг 3. На основе вычисленных значений вектор функции $F(X)$ находим приближенное значение функционала $Ф(F(X))$.

В виде рассмотренной схемы можно представить, например, решение задачи вычисления многомерного определенного интеграла от функции $F(\mathrm{X})$ в области D_X.

Суммарное количество арифметических операций, необходимых для *однократного* определения принадлежности вектора X множеству D_X (т.е. суммарную вычислительную сложность ограничений $x_i^- \le x_i \le x_i^+$ и ограничивающих функций $g_j(X)$), обозначим $C_g \ge 0$. Вообще говоря, величина C_g зависит от вектора X . Мы, однако, пренебрежем этой зависимостью, и будем полагать, что имеет место равенство $C_g = const$. Заметим, что до начала вычислений величина C_g, как правило, неизвестна. Однако в процессе первого же определения принадлежности некоторого узла сетки Ω множеству D_X, эту величину можно легко определить (с учетом предположения о независимости этой величины от вектора *X*). Поэтому будем полагать величину C_g известной.

Неизвестную вычислительную сложность вектор-функции $F(\mathrm{X})$ обозначим $C_f(X)$. Подчеркнем зависимость величины C_f от вектора X . Величина $C_f(X)$ удовлетворяет, во-первых, очевидному ограничению $C_f(X) \ge 0$. Во-вторых, положим, что известно ограничение сверху на эту величину C_f^{max} , имеющее смысл ограничения на максимально допустимое время вычисления значения $F(\mathrm{X})$. Вычислительную сложность $C_f(X_i)$ назовем вычислительной сложностью узла $X_i, i \in [1:Z]$.

Вычислительную сложность генерации сетки Ω положим равной ZC_Ω, а вычислительную сложность конечномерной аппроксимации функционала $\Phi(\mathrm{F(X)})$ - равной ζC_Φ, где ζ – общее количество узлов сетки Ω, принадлежащих множеству D_X. Положим, что при данных n ,m величины C_Ω, C_Φ – известные константы.

В качестве вычислительной системы рассмотрим однородную МВС с распределенной памятью, состоящую из процессоров $P_1, P_2, \dots, P_N$ и *host*-процессора, имеющих следующие параметры:

- l – длина вещественного числа в байтах;
- t – время выполнения одной арифметической операции с плавающей запятой;
- t_s – латентность коммуникационной сети;
- t_c – время передачи байта данных между двумя соседними процессорами системы без учета времени t_s;
- $d(N)$ – диаметр коммуникационной сети.

В качестве меры эффективности параллельных вычислений используем ускорение

$$S(N) = \frac{T(1)}{T(N)},$$

где $T(1)$ – время последовательного решения задачи на одном процессоре системы, $T(N)$ – время параллельного решения той же задачи на N процессорах.

Схема метода балансировки нагрузки

Положим, что из числа Z узлов расчетной сетки Ω множеству D_X принадлежит ξ узлов $\widetilde{X_1}, \widetilde{X_2}, \dots, \widetilde{X_\zeta}$. Обозначим $z = \left\lceil \frac{\zeta}{N} \right\rceil$. Тогда идею рассматриваемого метода балансировки загрузки можно представить в следующем виде:

- среди всех узлов $\mathrm{X}_1, \mathrm{X}_2, \dots, X_N$ сетки Ω выделяем ζ узлов $\widetilde{X_1}, \widetilde{X_2}, \dots, \widetilde{X_\zeta}$;
- разбиваем узлы $\widetilde{X_1}, \widetilde{X_2}, \dots, \widetilde{X_\zeta}$ на N множеств $\widetilde{\Omega}_i\ i \in [1{:}\,N]$, где множество $\widetilde{\Omega}_1$ содержит узлы $\widetilde{X_1}, \widetilde{X_2}, \dots, \widetilde{X_z}$, множество $\widetilde{\Omega}_2$ – узлы $\tilde{X}_{z+1}, \tilde{X}_{z+2}, \dots, \tilde{X}_{2z}$ и т.д.

- назначаем для обработки процессору P_i множеств узлов $\widetilde{\Omega}_i$ $i \in [1:N]$.

Наглядно схема работы метода изображена на рисунке 2.

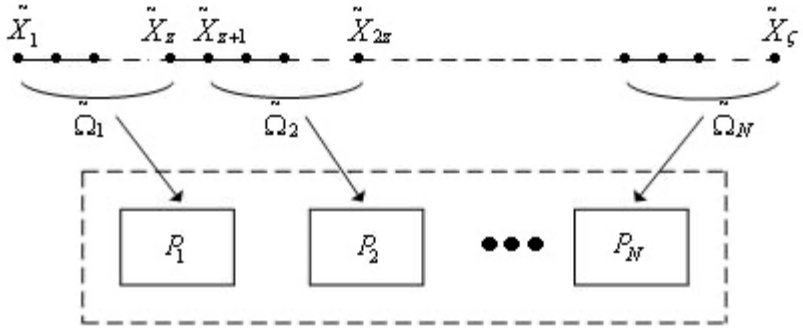

Рисунок 1 – Схема балансировки методом равномерной декомпозиции расчётных узлов.

Схему параллельных вычислений при балансировке загрузки методом равномерной декомпозиции расчетных узлов можно представить в следующем виде.

Шаг 1. Host-процессор выполняет следующие действия:

- строит сетку Ω;
- среди всех узлов $X_1, X_2, \dots, X_z$ сетки Ω выделяет ζ узлов $\widetilde{X_1}, \widetilde{X_2}, \dots, \widetilde{X_\zeta}$;
- разбивает узлы $\widetilde{X_1}, \widetilde{X_2}, \dots, \widetilde{X_\zeta}$ на N множеств узлов $\widetilde{\Omega}_i$ $i \in [1:N]$;
- передает процессору P_i координаты узлов множества $\widetilde{\Omega}_i$.

Шаг 2. Процессор P_i выполняет следующие действия:

- принимает от host-процессора координаты z узлов множества $\widetilde{\Omega}_i$;
- вычисляет в каждом из этих узлов значение вектор-функции $F(X)$;
- передает *host*-процессору z вычисленных векторов и заканчивает вычисления.

Шаг 3. *Host*-процессор выполняет следующие действия:

- принимает от процессоров $P_i, i \in [1:N]$ вычисленные ими значения вектор-функции $F(X)$;

- на основе ζ полученных значений вектор функций $F(X)$ вычисляет приближённое значение функционала Ф(F(X)).

Вычислительный эксперимент

Рассмотрим двумерную задачу (n=2). Параллелепипед $П$ в этом случае представляет собой прямоугольник $П = \{X | x_i^- \le x_i \le x_i^+, i \in [1,2]\}$. Положим, что $x_1^- = x_2^- = 0$, $x_1^+ = x_2^+ = 1$, так что область $П$ является единичным квадратом (рисунок 3).

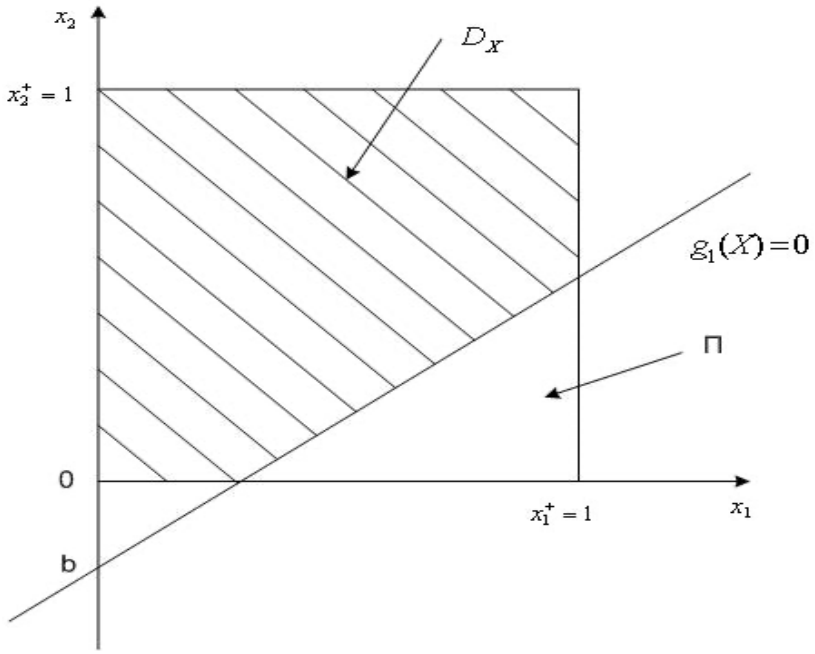

Рисунок 2 – Расчетная область задачи.

Множество D формируется с помощью одной ограничивающей функции $g_1(X) \ge 0$, т.е. $D = \{X | g_1(X) \ge 0\}$. Примем, что эта функция линейна и проходит через заданную преподавателем точку плоскости $0x_1x_2$ с координатами (0, b). Таким образом, уравнение этой функции имеет вид $x_2 = ax_1 + b, a > 0$ (при этом, очевидно, $g_1(X) = g_1(x_1, x_2) = x_2 - ax_1 - b$).

В соответствии с номером варианта заданы значения параметров ограничивающей функции: $a = 0.5,\ b = 0$. Количество узлов сетки, удовлетворяющих ограничивающей функции, равно 49152 (для сетки 256×256). График ограничивающей функции показан на рисунке 3.

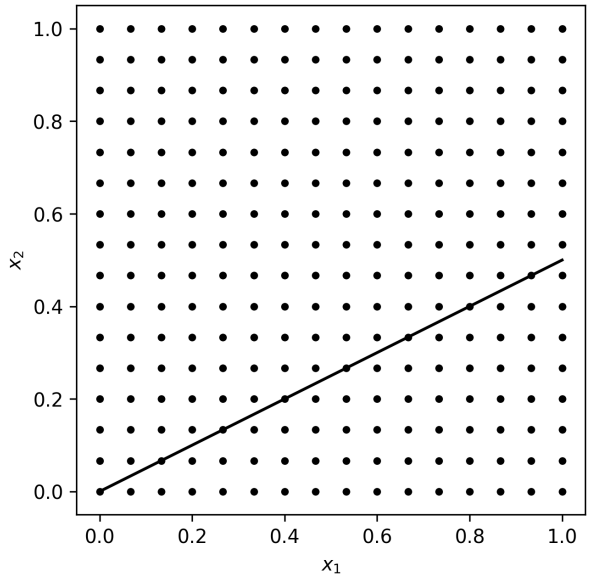

Рисунок 3 – Пересечение сетки Ω с ограничивающей функцией $g_1(x_1, x_2)$.

Будем исходить из следующих значений параметров задачи и МВС:

- $m = 100$;
- $l = 8$;
- $t = 10 * 10^{-9}[c]$;
- $t_s = 50 * 10^{-6}[c]$;
- $t_c = \frac{1}{80} * 10^{-6}[c]$;
- $d(N) = \lceil 2sqrt(N) - 1 \rceil$;
- $a = 0.5$;
- $b = 0$;
- $C_f^{max} = 0.5 * 10^7$;
- $C_f^{max} = 5 * 10^4$.

Пренебрежем вычислительными затратами на построение сетки Ω, на вычисление значений ограничивающей функции $g_1(X)$, а также на построение приближенного значения функционала $\Phi(F(X))$, т.е. положим $C_\Omega = 0$, $C_g = 0$, $C_\Phi = 0$.

Положим также, что вычислительная сложность $C_f(X)$ вектор-функции $F(X)$ есть случайная величина, равномерно распределенная на интервале

$[0, C_f^{max}]$. При этом вычислительная сложность $C_f(\tilde{X}_i)$ узла $\tilde{X}_i$, $i \in [1:\zeta]$ также представляет собой случайную величину, равномерно распределенную в указанном интервале.

Заметим, что математическое ожидание величины $C_f(\tilde{X}_i)$ в этом случае равно

$$M[C_f] = \frac{C_f^{max}}{2},$$

а её дисперсия и среднее квадратичное отклонения равны

$$D[C_f] = \frac{\left(C_f^{max}\right)^2}{12}, \quad \sigma[C_f] = \frac{C_f^{max}}{2\sqrt{3}}.$$

По результатам r «прогонов» оценка математического ожидания ускорении $\widetilde{M}[S]$ вычисляется по формуле

$$\widetilde{M}[S(N)] = \frac{1}{r}\sum_{i=1}^{r} S_i(N),$$

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

$$\widetilde{D}[S(N)] = \frac{1}{r-1}\sum_{i=1}^{r}\left(S_i(N) - \widetilde{M}[S(N)]\right)^2, \quad \tilde{\sigma}[S(N)] = \sqrt{\widetilde{D}[S(N)]}$$

соответственно. Здесь $S_i(N)$, $i \in [1:r]$ - оценка ускорения вычислений, полученная в i-ом «прогоне».

Для интерпретации результатов работы необходимы следующие сведения из теории вероятностей.

Пусть $Y_1, Y_2, \dots, Y_r$ – независимые случайные величины с математическими ожиданиями $M_1, M_2, \dots, M_r$ и дисперсиями $D_1, D_2, \dots, D_r$. Тогда математическое ожидание суммы случайных величин $Y_1, Y_2, \dots, Y_r$ равно сумме математических ожиданий слагаемых:

$$M\left(\sum_{i=1}^{r} Y_i\right) = \sum_{i=1}^{r} M_i.$$

Дисперсия суммы случайных величин $Y_1, Y_2, \dots, Y_r$ равна сумме дисперсий слагаемых:

$$D\left(\sum_{i=1}^{r} Y_i\right) = \sum_{i=1}^{r} D_i.$$

Подчеркнем, что последняя формула справедлива только для независимых (некоррелированных) случайных величин $Y_1, Y_2, \dots, Y_r$.

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 256; число процессоров
points_N EQU 49152; число узлов, удовлетворяющих ограничению

t_s EQU 50e-6; латентность коммуникационной сети
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
z EQU points_N/N_proc
tau_i EQU 2#t_s+z#N_gr#l_s#d_s#t_c+z#m_s#l_s#d_s#t_c
t EQU 1e-8

uniform_cf_par FUNCTION rn2,c2
0,0.0/1,0.5e7
uniform_cf_posl FUNCTION rn3,c2
0,0.0/1,0.5e7

proc_par STORAGE 256; число процессоров
proc_posl STORAGE 1
us VARIABLE p4/p3; p4 - последовательное, p3 – параллельное
tabl_s TABLE v$us,42,0.25,60
generate 1e8,100
split (N_proc - 1); запуск N процессов
assign 1,z; присвоение первому параметру транзакта значения z

queue qhost1_par; помещение транзакта в очередь на выдачу задания
seize host
depart qhost1_par
advance 5e-6,3e-6; задержка
release host

queue qproc_par; очередь для обработки
enter proc_par
depart qproc_par
advance tau_i,1e-8; 2 аргумент - это разброс
proc2 advance t,fn$uniform_cf_par; t * Cf(x_ij)
loop 1,proc2
leave proc_par
```

```
queue qhost2_par
seize host
depart qhost2_par
advance 5e-6,3e-6
release host
assemble (N_proc)
assign 3,m1; в 3 параметр транзакта запишем время параллельной обработки

mark 2; 2 параметр транзакта для измерения активного времени транзакта
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; запишем в 4 параметр транзакта время последовательной обработки

tabulate tabl_s
TERMINATE 1
START 300
```

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 $S(N)$

N	$S(N), C_f^{max} = 5 * 10^4$	$S(N), C_f^{max} = 0.5 * 10^7$
2	1.970	2.007
4	3.847	3.991
8	7.403	7.951
16	14.017	15.661
32	26.413	31.233
64	48.445	61.457
128	85.437	120.911
256	146.528	229.915

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.

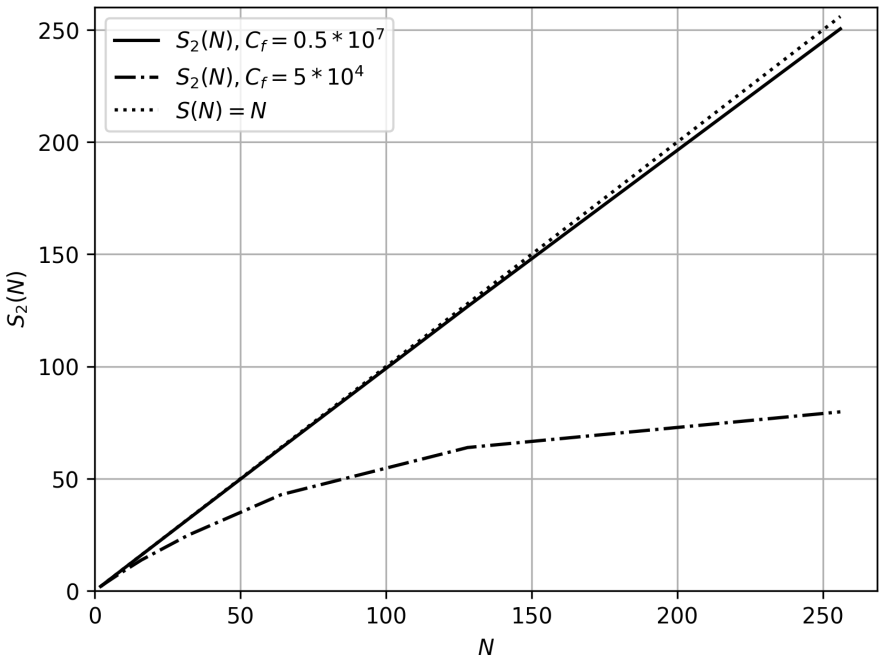

Рисунок 4 – Оценка ускорения при статической балансировке загрузки методом равномерной декомпозиции расчетных узлов, полученная с помощью программы на C.

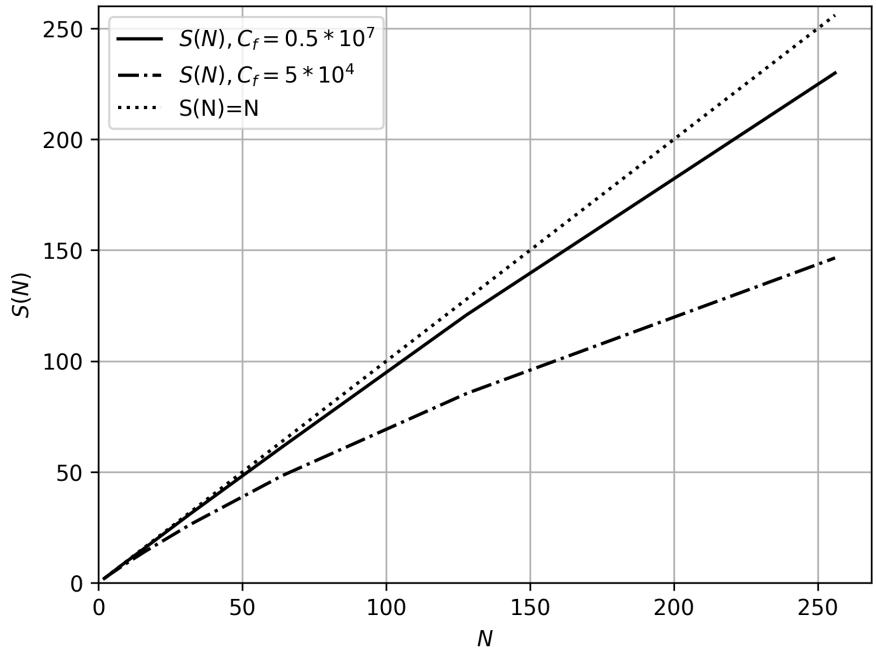

Рисунок 5 – Оценка ускорения при статической балансировке загрузки методом равномерной декомпозиции расчетных узлов, полученная с помощью программы на GPSS.

В таблице 2 представлены полученные значения оценки математического ожидания $\widetilde{M}[S(N)]$ и оценки среднего квадратичного

отклонения $\tilde{\sigma}[S(N)]$ для разного числа процессоров при заданных значениях вычислительной сложности.

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

N	$S(N), C_f^{max} = 5 * 10^4$		$S(N), C_f^{max} = 0.5 * 10^7$	
	$\widetilde{M}[S(N)]$	$\tilde{\sigma}[S(N)]$	$\widetilde{M}[S(N)]$	$\tilde{\sigma}[S(N)]$
2	1.963	0.007	1.996	0.008
4	3.824	0.017	3.977	0.016
8	7.373	0.033	7.911	0.041
16	14.030	0.073	15.689	0.094
32	26.207	0.170	31.000	0.223
64	47.885	0.357	60.875	0.542
128	85.196	0.676	118.538	1.420
256	146.475	1.358	227.697	3.293

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

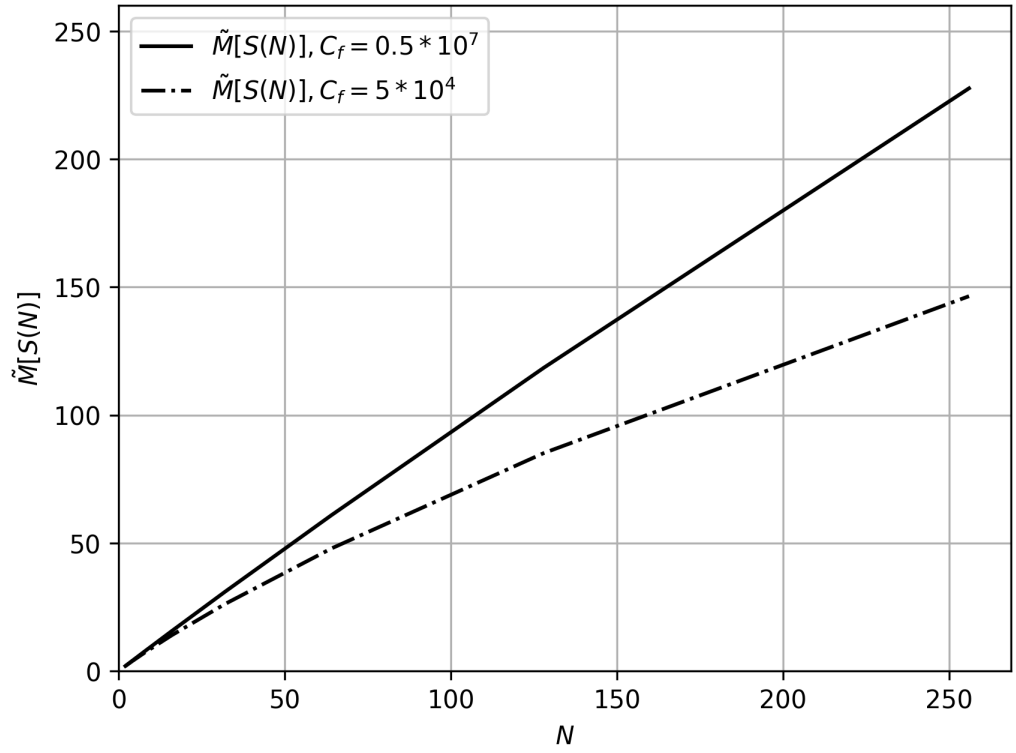

Рисунок 6 – Оценка математического ожидания при статической балансировке загрузки методом равномерной декомпозиции расчетных узлов, полученная с помощью программы на GPSS.

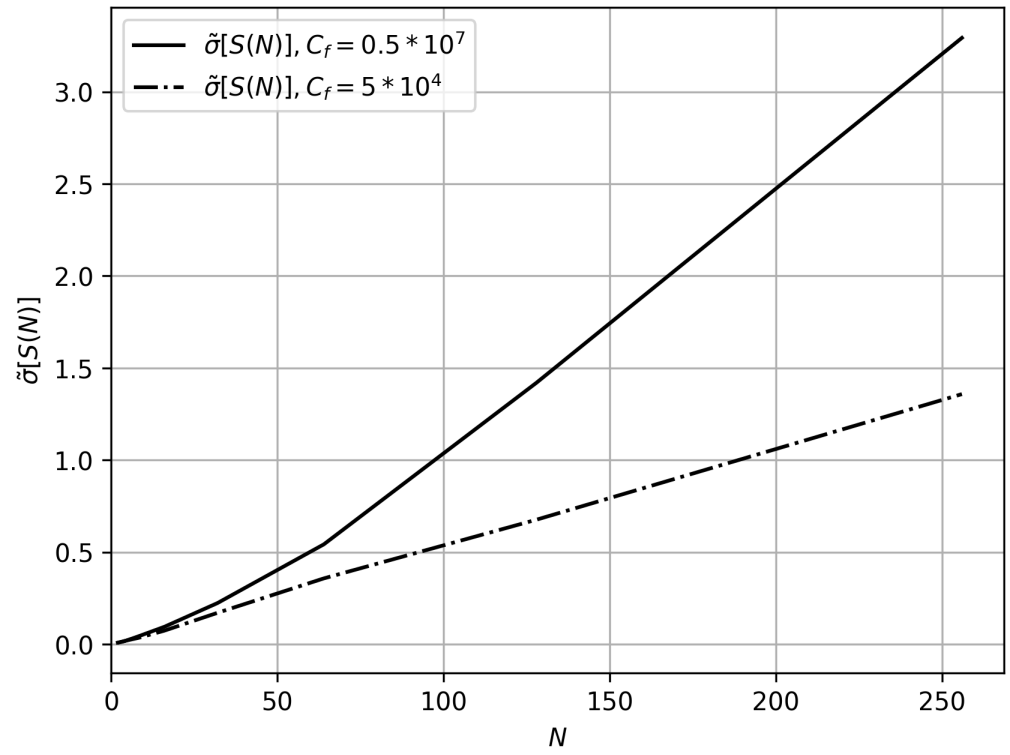

Рисунок 7 – Оценка среднего квадратичного отклонения при статической балансировке загрузки методом равномерной декомпозиции расчетных узлов, полученная с помощью программы на GPSS.

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 $[0, C_f^{max}]$. При увеличении C_f^{max}, диапазон увеличивается, вследствие этого увеличивается и разброс значений ускорения.

Заключение

Изучен метод статической балансировки загрузки МВС на основе равномерной декомпозиции узлов расчетной сетки. 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).