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 РОБОТОТЕХНИКА И КОМПЛЕКСНАЯ АВТОМАТИЗАЦИЯ (РК)

КАФЕДРА РК6 «СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ»

Отчет по лабораторной работе

Исследование эффективности статической балансировки загрузки МВС с использованием имитационного моделирования

Студент ________________ **Абидоков Р. Ш.**
РК6-21М

Преподаватель ________________ **Карпенко А. П.**

2022 г.

Постановка задачи исследования эффективности балансировки загрузки МВС

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 $\Pi = \{X \mid x_i^- \le x_i \le x_i^+, i \in [1, n]\}$, где $x_i^-, x_i^+ -$ заданные константы. На вектор X дополнительно наложено некоторое количество функциональных ограничений, формирующих множество $D = \{X \mid g_i(X) \ge 0, j = 1,2, \dots\}$, где $g_i(X) -$ непрерывные ограничивающие функции.

На множестве $D_x = \Pi \cap D$ тем или иным способом (аналитически или алгоритмически) определена вектор-функция $\mathrm{F}(X)$ со значениями в пространстве R^m. Ставится задача поиска значения некоторого функционала $\Phi(\mathrm{F}(X))$.

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 $\mathrm{F}(X)$.

Шаг 3. На основе вычисленных значений вектор функции $\mathrm{F}(X)$ находим приближенное значение функционала $\Phi(\mathrm{F}(X))$.

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 $g_i(X)$, обозначим $C_g \ge 0$. *Далее в эксперименте будем полагать* $C_g = 0$.

Неизвестную вычислительную сложность вектор-функции $\mathrm{F}(X)$ обозначим $\mathrm{C}_f(X)$. Подчеркнем зависимость величины C_f от вектора X. Величина $\mathrm{C}_f(X)$ удовлетворяет, во-первых, очевидному ограничению $\mathrm{C}_f(X) \ge 0$. Во-вторых, положим, что известно ограничение сверху на эту величину C_f^{max}, имеющее смысл ограничения на максимально допустимое время вычисления значения $\mathrm{F}(X)$. Вычислительную сложность $\mathrm{C}_f(X_i)$ 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_Ω, где ζ (дзета) - общее количество узлов сетки Ω, принадлежащих множеству D_x.

Далее в эксперименте также будем полагать $\mathrm{C}_f = C_\Omega = 0$.

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:

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

В качестве меры эффективности параллельных вычислений используем ускорение

$$S_i(N) = \frac{T(1)}{T_i(N)}, \tag{1}$$

где $T(1)$ – время последовательного решения задачи на одном процессоре системы, $T_i(N)$ – время параллельного решения той же задачи на N процессорах, $i = 1,2$ – номер метода балансировки.

Статическая балансировка загрузки

Положим, что из числа Z узлов расчетной сетки Ω множеству D_x принадлежит ζ узлов $\tilde{X}_1, \tilde{X}_2 \dots \tilde{X}_\zeta$. Обозначим $z = \left[\frac{\zeta}{N}\right]$. Тогда идею рассматриваемого метода бал ансировки загрузки можно представить в следующем виде (Рис. 1):

- среди всех узлов $X_1, X_2 \dots X_Z$ сетки Ω выделяем ζ узлов $\tilde{X}_1, \tilde{X}_2 \dots \tilde{X}_\zeta$;
- разбиваем узлы $\tilde{X}_1, \tilde{X}_2 \dots \tilde{X}_\zeta$ на N множеств $\tilde{\Omega}_i, i \in [1, N]$, где множество $\tilde{\Omega}_1$ содержит узлы $\tilde{X}_1, \tilde{X}_2 \dots \tilde{X}_z$, множество $\tilde{\Omega}_2$ - узлы $\tilde{X}_{z+1}, \tilde{X}_{z+2} \dots \tilde{X}_{2z}$ и т.д.
- назначаем для обработки процессору P_i множеств узлов $\tilde{\Omega}_i$, $i \in [1:N]$.

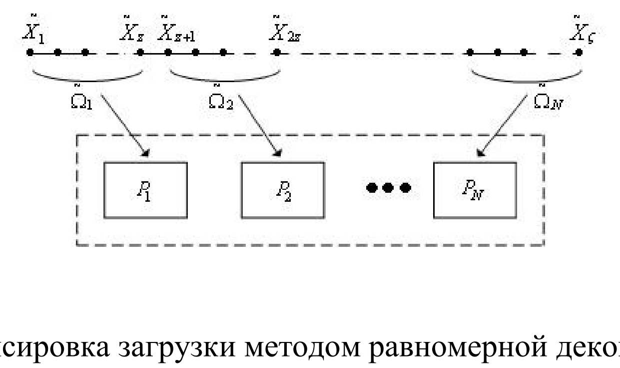

Рис. 1. Балансировка загрузки методом равномерной декомпозиции узлов

Для данного метода балансировки загрузки время решения задачи на процессоре P_i можно оценить величиной

$$\tau_i = \tau = 2t_s + znldt_c + zmldt_c + tzC_f, \qquad (2)$$

время параллельного решения всей задачи – величиной

$$T_2(N) = \tau, \qquad (3)$$

а время решение задачи на одном процессоре величиной

$$T(1) = t\zeta C_f. \qquad (4)$$

Таким образом, схема алгоритма для аналитической оценки эффективности балансировки загрузки методом равномерной декомпозиции расчетных узлов имеет следующий вид:

- в квадрате П строим равномерную по каждому из измерений сетку Ω;
- находим количества узлов ζ, z;
- по формуле (2) вычисляем значение величины τ;
- по формуле (3) находим величину $T_2(N)$;
- по формуле (4) определяем значение величины $T(1)$;
- по формуле (1) находим оценку ускорения.

Экспериментальная часть

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

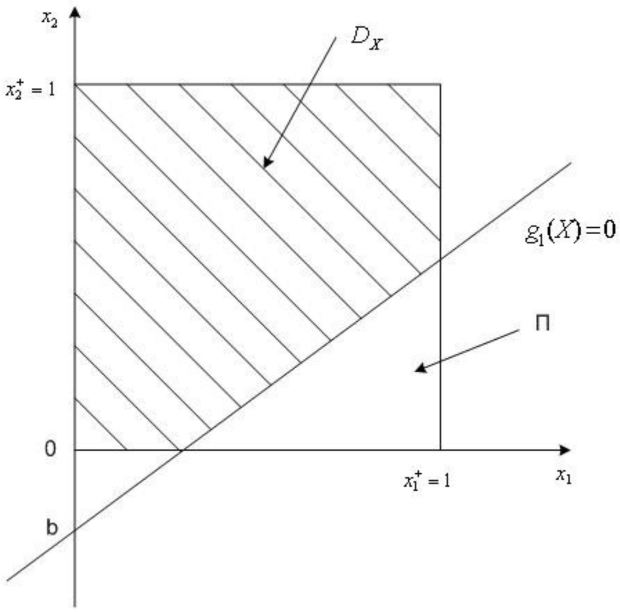

Рис. 2. Расчетная область задачи

Множество D формируется с использованием одной ограничивающей функции $g_1(X) \ge 0$, то есть $D = \{X | g_1(X) \ge 0\}$. Примем, что эта функция линейна и проходит через заданную преподавателем точку плоскости $O(x_1, x_2)$ с координатами $(0, b)$, как показано на Рис. 2.

Таким образом, уравнение этой функции имеет вид $x_2 = ax_1 + b,\ a > 0$ В соответствии с номером варианта заданы значения параметров ограничивающей функции: $a = 1.0,\ b = -0.1$. Общее количество узлов $Z = 256 * 256 = 65536$, количество попавших в область D_x узлов $\zeta = 38971$ (принимаем равным 39168, как ближайшим кратным 256).

Параметры моделируемой МВС:

$N = 2,4,8,16,32,64,128,256;$

$C_f = 10^2, 10^3, 10^4$ c;

$m = 100;$

$l = 8;$

$t = 10 * 10^{-9}$ c;

$t_s = 50 * 10^{-6}$ c;

$t_c = \frac{1}{80} * 10^{-6}$ c;

$d(N) = 2\sqrt{N} - 1;$

Полученные значения матожиданий и среднеквадратичных отклонений значений ускорений для целевой функции со сложностями, равномерно распределенными в интервалах $C_f \in [0, C_f^{max}]$, $C_f^{max} = 2 * 10^4, 2 * 10^6$, приведены в Табл. 1, 2. Исходный код программы приведен в Приложении 1.

Табл. 1. Полученные результаты для различных N при $r = 1$

N	$S,\ C_f^{max} = 2 \cdot 10^4$	$S, C_f^{max} = 2 \cdot 10^6$
2	0.99	1.98
4	1.31	3.89
8	1.57	7.61
16	1.74	14.5
32	1.84	26.7
64	1.90	46.4
128	1.92	73.6
256	1.94	105.1

Табл. 2. Полученные результаты для различных N при $r = 300$

N	$S,\ C_f^{max} = 2 \cdot 10^4$		$S, C_f^{max} = 2 \cdot 10^6$	
	$M^*[S(N)]$	$\sigma^*[S(N)]$	$M^*[S(N)]$	$\sigma^*[S(N)]$
2	0.989	0.003	1.977	0.008
4	1.313	0.004	3.902	0.016
8	1.571	0.004	7.608	0.032
16	1.743	0.005	14.565	0.091
32	1.843	0.005	26.742	0.188
64	1.899	0.005	46.467	0.409
128	1.928	0.005	73.593	0.548
256	1.942	0.005	105.029	0.852

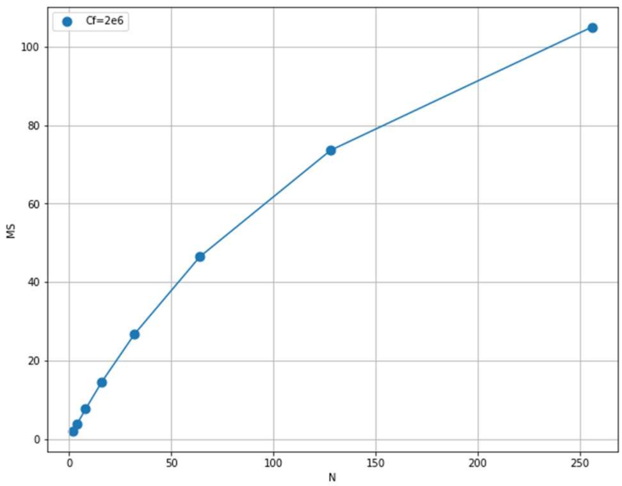

Рис. 3 $M^*[S(N)]$ для $C_f^{max} = 2 \cdot 10^6$

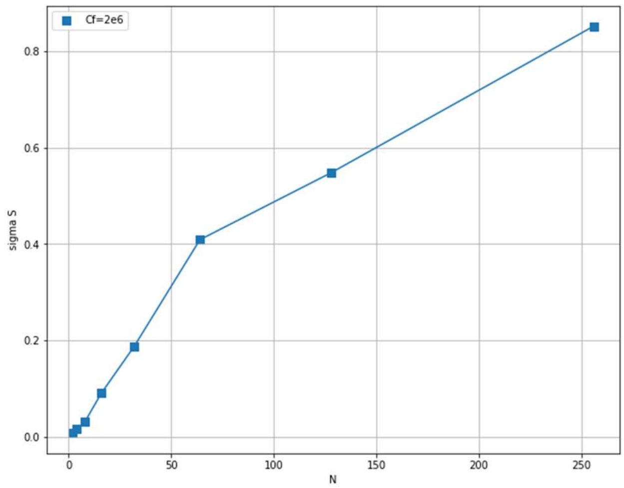

Рис. 4 $\sigma^*[S(N)]$ для $C_f^{max} = 2 \cdot 10^6$

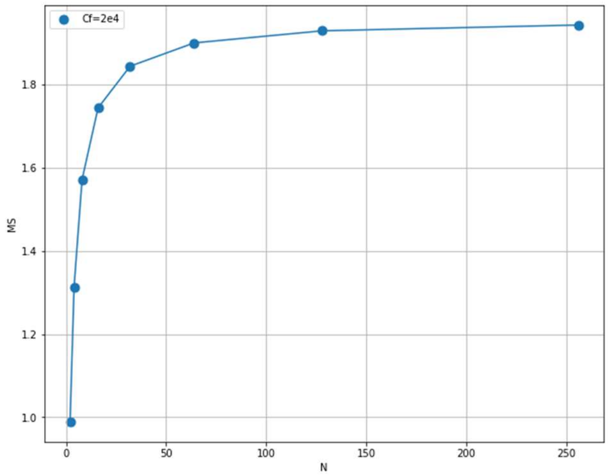

Рис. 5 $M^*[S(N)]$ для $C_f^{max} = 2 \cdot 10^6$

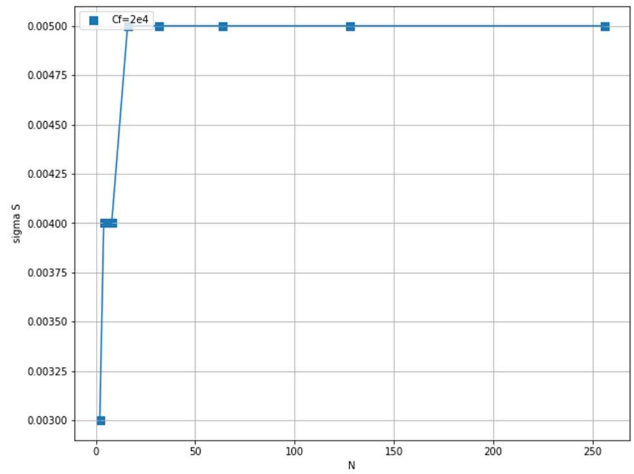

Рис. 6 $\sigma^*[S(N)]$ для $C_f^{max} = 2 \cdot 10^6$

Ответы на контрольные вопросы

1. *Чем объясняется наблюдаемый характер зависимостей ускорений $S(N)$ и оценки математического ожидания ускорения $M^*[S(N)]$?*

С ростом числа процессоров уменьшается количество полезных вычислений, совершаемых каждым процессором, при этом издержки на коммуникацию не уменьшаются – как следствие, уменьшается эффективность распараллеливания.

2. *Чем объясняется наблюдаемый характер зависимости оценки математического ожидания ускорения $M^*[S(N)]$ от величины C_f^{max}?*

С увеличением величины C_f^{max} уменьшается доля времени, затрачиваемого на коммуникацию между процессами, и увеличивается доля времени, затрачиваемого на вычисление функции в узлах, которое и делится между процессорами – как следствие, растет эффективность распараллеливания.

3. *Чем объясняется наблюдаемый характер зависимости оценки среднего квадратичного отклонения* $\sigma^*[S(N)]$ *от величины* C_f^{max}?

Поскольку форма функции распределения величины ускорения $S(N)$ не зависит от числа процессоров N, отношение между математическим ожиданием $M^*[S(N)]$ и средним квадратичным отклонением $\sigma^*[S(N)]$ также не зависит от N. По этой причине график среднего квадратичного отклонения повторяет график математического ожидания.

Исходный код программы

```
N_proc EQU 256
points_N EQU 39168

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
z EQU points_N/N_proc
tau_i EQU 2
t EQU 1e-9
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 256
proc_posl STORAGE 1
us VARIABLE p4/p3

tabl_s TABLE v$us,42,0.25,60
generate 1e8,100
split (N_proc - 1)

assign 1,z
queue qhost1_par
seize host
depart qhost1_par
advance 5e-6,3e-6
release host

queue qproc_par
enter proc_par
depart qproc_par
advance tau_i,1e-8
proc2 advance t,fn$uniform_cf_par
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

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 30
```