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 динамической экспоненциальной балансировки загрузки МВС с использованием имитационного моделирования

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

Преподаватель _____ **Карпенко А. П.**

2022 г.

Постановка задачи исследования эффективности статистической 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 $D_x = \Pi \bigcap D$ 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 Π 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 множествуD_x.

Далее в эксперименте также будем полагать $C_\Omega = 0, C_\Phi = 0$.

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

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

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 методом равномерной декомпозиции параллелепипеда П

Положим, что из числа Z узлов расчетной сетки Ω множеству D_x принадлежит ζ узлов $\tilde{X}_1, \tilde{X}_2 \dots \tilde{X}_\zeta$.

Суть экспоненциальной декомпозиции узлов передает следующая ее схема k-ой итерации (Рис. 1).

1) Если количество оставшихся необработанными ζ_k узлов (из числа узлов $\tilde{X}_1, \tilde{X}_2 \dots \tilde{X}_\zeta$) не превышает количество процессоров в системе, то включаем во множество узлов $\widetilde{\Omega}_k$ все ζ_k необработанных узлов; $\zeta_1 = \zeta$.

2) Если количество оставшихся необработанными ζ_k узлов превышает количество процессоров в системе, то среди этих узлов выделяем множество узлов $\widetilde{\Omega}_k$, содержащее $z_k = \left\lceil \frac{\zeta_k}{\rho} \right\rceil$ узлов, где $\rho -$ коэффициент декомпозиции.

3) Множество узлов $\widetilde{\Omega}_k$ разбиваем на N подмножеств $\overline{\omega_{k,j}}$, $j \in [1:N]$, , содержащих по $s_k = \left\lceil \frac{z_k}{N} \right\rceil$ узлов.

Для простоты записи положим, что величины ζ_k кратны ρ, так что все $z_k = \frac{\zeta_k}{\rho}$ являются целыми. Положим также, что величины z_k кратны количеству процессоров N и поэтому все $s_k = \frac{z_k}{N}$ также являются целыми.

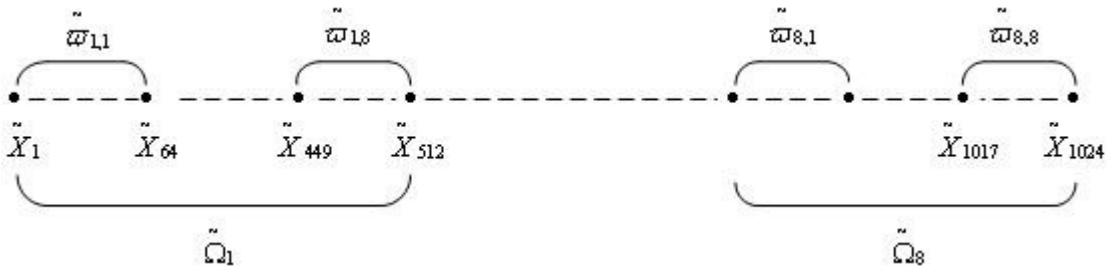

Рис. 1. Схема экспоненциального разбиения узлов

Схема параллельного решения поставленной задачи с использованием динамической экспоненциальной балансировки загрузки имеет следующий вид:

Шаг 1. Host-процессор выполняет следующие действия:

- строит сетку Ω;
- среди всех узлов $X_1, X_2 \dots X_Z$ сетки Ω выделяет узлы $\tilde{X}_1, \tilde{X}_2 \dots \tilde{X}_\zeta$;
- полагает $k = 1;\ \zeta_k = \zeta$.

Шаг 2. Host-процессор выполняет следующие действия:

- если число не обработанных узлов ζ_k не превышает количество процессоров в системе, то включает во множество $\widetilde{\Omega}_k$ все ζ_k узлов;
- если число ζ_k не обработанных узлов превышает количество процессоров N, то выделяет среди этих узлов множество узлов $\widetilde{\Omega}_k$;
- разбивает множество узлов $\widetilde{\Omega}_k$ на N подмножеств $\overline{\omega_{k,J}},\ j \in [1:N]$;
- если множество узлов $\widetilde{\Omega}_k$ не исчерпано, то передает свободному процессору P_i координаты узлов первого из необработанных подмножеств $\overline{\omega_{k,J}}$;
- если множество узлов $\widetilde{\Omega}_k$ исчерпано, то выполняет присваивания $k = k + 1,\ \zeta_k = \zeta_{k-1}(1 - \rho^{-1})$ и переходит к первому пункту данного шага.

Шаг 3. Процессор P_i выполняет следующие действия:

- принимает от host-процессора координаты узлов подмножества$\overline{\omega_{k,J}}$;
- вычисляет во всех узлах подмножества $\overline{\omega_i}$ значения вектор-функции $F(x)$;
- посылает host-процессору вычисленные значения $F(x)$;

Шаг 4. Если исчерпаны все узлы $\tilde{X}_1, \tilde{X}_2 \dots \tilde{X}_\zeta$, то host-процессор:

- посылает освободившемуся процессору P_i сообщение об окончании решения задачи;
- после получения всех вычисленных значений функции $F(x)$ от всех процессоров вычисляет приближенное значение функционала $\Phi(F(x))$;

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).

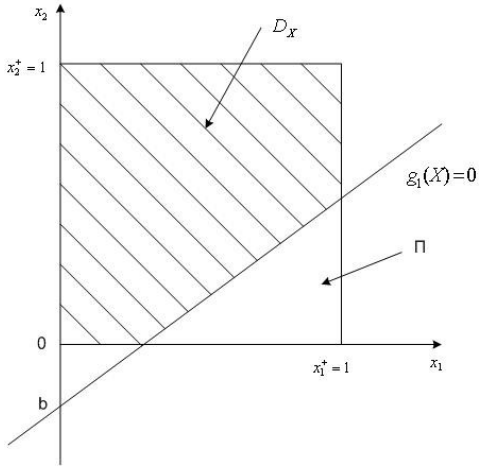

Рис. 2. Расчетная область задачи

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:

$N = 2,4,8,16,32,64,128,256$;
$C_f = 10^2, 10^3, 10^4$ с;
$m = 100$;
$l = 8$;
$t = 10 * 10^{-9}$ с;
$t_s = 50 * 10^{-6}$ с;
$t_c = \frac{1}{80} * 10^{-6}$ с;
$d(N) = 2\sqrt{N} - 1$;

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	0.992	1.14
4	1.82	2.40
8	3.43	5.15
16	6.07	11.23
32	10.29	23.76
64	17.76	51.08
128	20.08	108.74
256	13.63	187.09

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	0.991	0.021	1.134	0.023
4	1.836	0.037	2.402	0.052
8	3.365	0.066	5.119	0.116
16	5.954	0.119	11.004	0.264
32	10.406	0.213	23.676	0.604
64	17.707	0.406	50.447	1.407
128	20.055	0.368	108.624	2.483
256	13.381	0.241	188.724	4.092

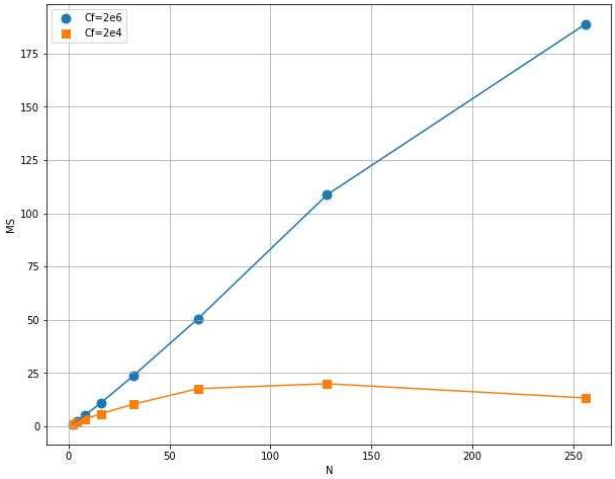

Рис. 3. График $M^*[S(N)]$

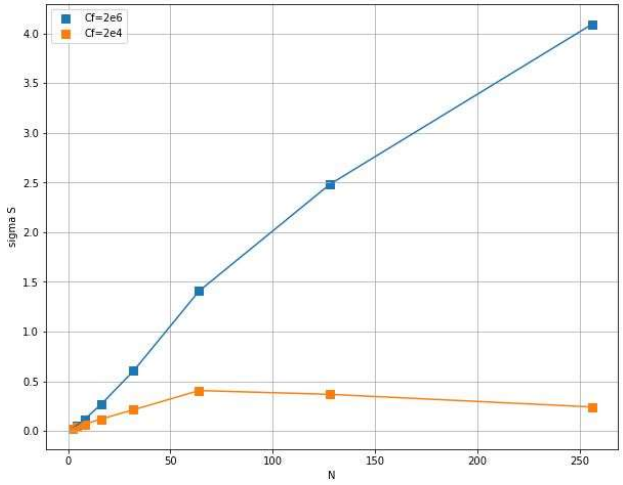

Рис. 4. График $\sigma^*[S(N)]$

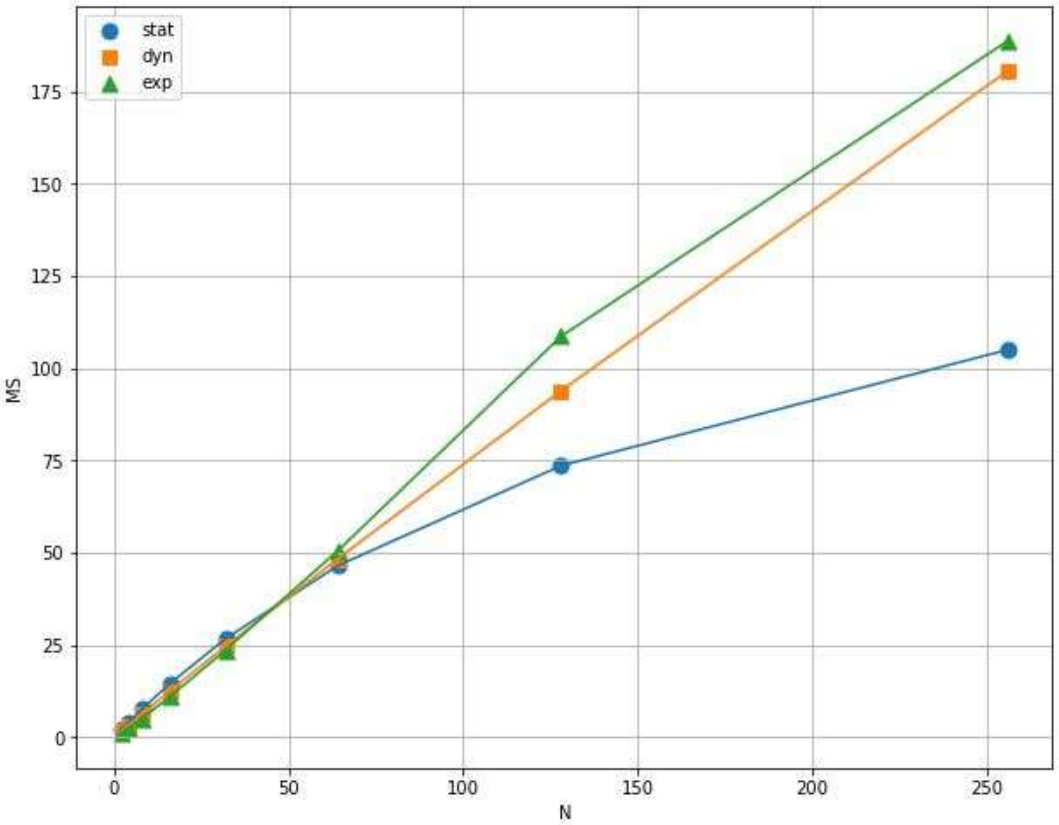

Рис. 5. График $M^*[S(N)]$ при $C_f^{max} = 2 \cdot 10^6$ для различных методов балансировки

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
k_1 EQU 4
s_1 EQU (points_N/N_proc)/k_1
T_ik EQU 2

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 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)

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
```