Оценка эффективности алгоритмов математической физики для компьютеров с распределенной памятью

И. Н. Коньшин[1,2]

[1]ИВМ им. Г.И. Марчука РАН
[2]ВЦ им. А.А. Дородницына, ФИЦ ИУ РАН

25 сентября, 2018

План

Ситуация:

- // результатов много, а оценок // эффективности почти нет
- // моделей достаточно, оценки тоже встречаются,
  но либо слишком абстрактные, либо слишком подробные

Цель:

- конструктивные оценки для задач математической физики

План:

- оценки параллельной эффективности алгоритмов
- их применение к задачам математической физики
- эксперименты по проведению обменов на фоне вычислений
- скорость передачи сообщений в зависимости их длины
- результатов численных экспериментов

Оценка //-ной эффективности (общая память)

Закон Амдала (Amdahl's law)

- p – количество процессов (потоков)
- $T(p)$ – время выполнения алгоритма для p процессов
- σ – доля последовательных (не распараллеленных) операций

Ускорение:

$$S(p) = \frac{T(1)}{T(p)} = \frac{T(1)}{\sigma T(1) + \frac{(1-\sigma)T(1)}{p}} = \frac{p}{1+\sigma(p-1)}$$

Эффективность:

$$E(p) = \frac{S(p)}{p} = \frac{1}{1+\sigma(p-1)}$$

Напрямую применимо к программам на OpenMP

Распределенная память (обмены MPI)

$$T_c = \tau_0 + \tau_c L_c$$

τ_0 – время инициализации сообщения
τ_c – скорость передачи сообщений (т.е. время передачи сообщения единичной длины)
T_c – время передачи сообщения длины L_c
Пока для простоты положим $\tau_0 = 0$, тогда

$$T_c = \tau_c L_c$$

Аналогично,

$$T_a = \tau_a L_a$$

τ_a – время выполнения одной характерной арифметической операции
L_a – общее количество арифметических операций алгоритма

Оценка //-ной эффективности (распределенная память)

Пусть

$$\tau = \tau_c/\tau_a, \qquad L = L_c/L_a$$

характеристики "параллельности" используемого компьютера и исследуемого алгоритма, соответственно

Тогда

$$\begin{aligned} S &= S(p) = T(1)/T(p) = T_a/(T_a/p + T_c/p) = pT_a/(T_a + T_c) \\ &= p/(1 + T_c/T_a) = p/(1 + (\tau_c L_c)/(\tau_a L_a)) = p/(1 + \tau L) \end{aligned}$$

А для //-ной эфф-ти еще проще:

$$E = \frac{S}{p} = \frac{1}{1 + \tau L}$$

Линейная алгебра: модель для метода IC0 + AS(0) + PCG

- 3 x DAXPY
- 2 x DDOT
- 1 x MVM
- 2 x SOL с блочно-диагональной треугольной матрицей

Матрица системы получена из дискретизации задачи:
n – размерность в одном направлении
$N = n \times n \times n$ – количество неизвестных (размерность всей системы)
$r = n^2$ – полуширина ленты матрицы
$(2d + 1)$-точечный d-мерный шаблон дискретизации $(d = 3)$

$$L = L_c/L_a = (p - 1)(r + 2)/((2d + 3)N)$$

$$S = \frac{p}{1 + \tau L} = \frac{p}{1 + \frac{\tau(p-1)(r+2)}{(2d+3)N}}$$

Ускорение: теория и эксперимент: IC0 + AS(0) + PCG

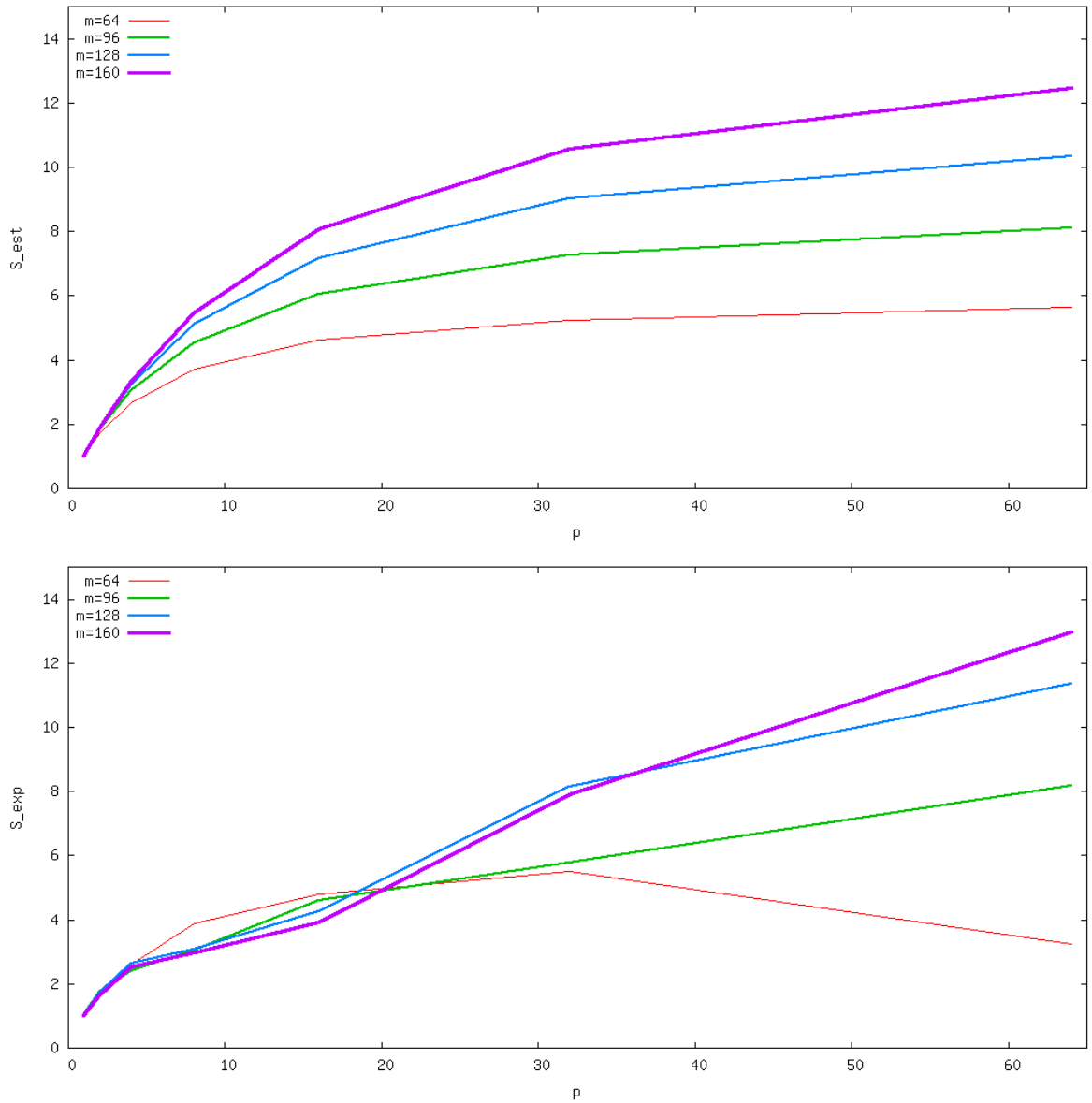


Figure : Ускорение (теория и эксперимент) для $n = 64, 96, 128, 160$

Ускорение: теория и эксперимент: IC0 + AS(0) + PCG

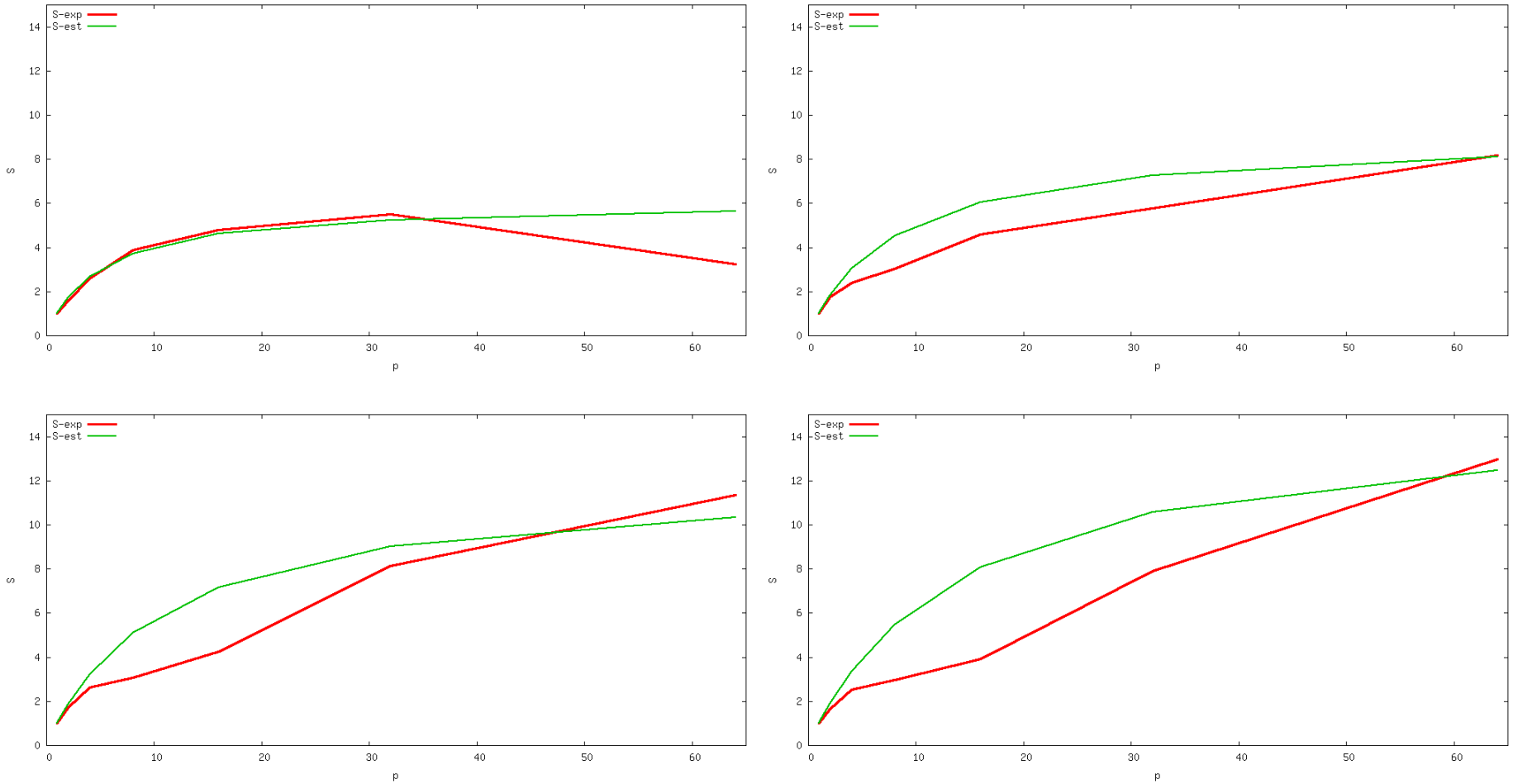
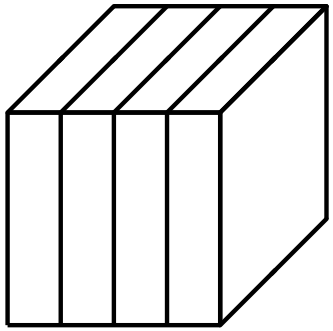
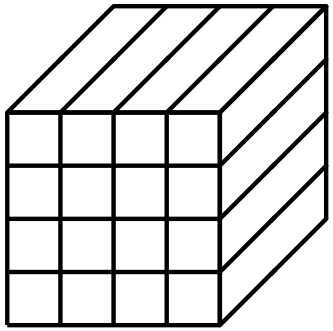


Figure : Ускорение (теория и эксперимент) для $n = 64, 96, 128, 160$

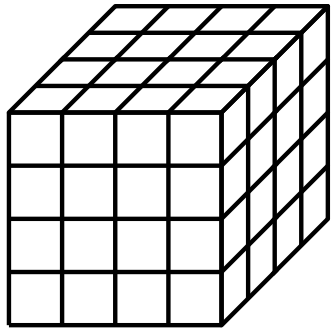
Математическая физика: модельная задача



$D = 1$,
$p = r$,
$L_a = CN/p$,
$L_c \approx 2Vn^2$,
$L \approx 2Vr/(Cn) \sim r/n$.



$D = 2$,
$p = r \times r$,
$L_a = CN/p$,
$L_c \approx 4Vn^2/r$,
$L \approx 4Vr/(Cn) \sim r/n$.



$D = 3$,
$p = r \times r \times r$,
$L_a = CN/p$,
$L_c \approx 6Vn^2/r^2$,
$L \approx 6Vr/(Cn) \sim r/n$.

Figure : Распределение данных по процессорам для трехмерной задачи математической физики по r слоев в одном, двух и трех направлениях

d-мерной "куб" ($d = 1, 2, 3$) со стороной в n ячеек
общее кол-во d-мерных кубических ячеек $N = n^d$
V – кол-во неизвестных функций на расчетную ячейку
$(2d + 1)$-точечный d-мерный шаблон дискретизации
C – кол-во арифм. операций на ячейку на ***явном*** шаге по времени

Оценка

$$N = n^d, \qquad p = r^D$$

$$L_a = CN/p = Cn^d/r^D$$
$$L_c = (2 - 2/r)DVn^{d-1}/r^{D-1}$$

$$L = L_c/L_a = (2-2/r)DVn^{d-1}r^D/(Cn^2r^{D-1}) = (2-2/r)DVC^{-1}\cdot r/n \sim r/n$$

$$E = 1/(1 + (2 - 2/r)DVC^{-1}\tau \cdot r/n), \qquad S = pE$$

$$E(d, D) = 1\Big/\left(1 + \left(2 - \frac{2}{p^{1/D}}\right)\frac{DV}{C}\tau \cdot \frac{p^{1/D}}{N^{1/d}}\right), \qquad S = pE(d, D)$$

Оценка для конкретного примера

p	$D = 1$	$D = 2$	$D = 3$
1	1.00	1.00	1.00
10	0.97	0.98	0.98
64	0.82	0.95	0.96
729	0.29	0.85	0.92

Table : Теоретическая оценка //-ной эффективности при конкретных параметрах:

$$d = 3, \quad N = n^3, \quad n = 1000, \quad V = 5, \quad C = 30, \quad \tau = 10$$

Кластер ИВМ РАН – сегмент x6core

- Compute Node Asus RS704D-E6;
- 12 ядер (два 6-ядерных процессора Intel Xeon X5650@2.67 ГГц);
- Оперативная память: 24 Гб.;
- Операционная система: SUSE Linux Enterprise Server 11 SP1 (x86_64);
- Коммутационная сеть: Mellanox Infiniband QDR 4x.

Для сборки кода использовался компилятор Intel языка C версии 4.0.1, с поддержкой MPI версии 5.0.3.

Обмены на фоне счета

Тест 1 из [Байдин-2008]

Ассинхронные обмены через MPI_Isend(), MPI_Irecv(), MPI_Waitall() выполнялись на фоне счета для массива размерности $M = 2^{25}$. Посчитанная порция длины M/N_{drops}, $N_{\text{drops}} = 1, 8, 64$ отсылалась на другой проц.

N_{drops}	$p = 1$	$p = 2$	$p = 13$
1	15.42	17.18	18.52
8	15.43	15.58	15.78
64	15.43	15.63	15.77

Table : Время работы теста 1 по проведению обменов на фоне счета

Ситуации:
$p = 1$ и $N_{\text{drops}} = 1$ – синхронность обменов
$p = 2$ – обмены на одном узле
$p = 13$ – обмены на разных узлах (x6core)

Время передачи сообщений

Тест 2 из [Байдин-2008]

Одно большое сообщение длиной $L_c = M = 2^m$ слов типа “double” разбивалось на $n_{c,i} = 2^i$ порций каждая длины $L_{c,i} = L_c/n_{c,i} = 2^{m-i}$, $i = 0, ..., m$.
Было выбрано $m = 25$ и $M = 2^m = 33554432$, и для сегмента x6core были получены значения $T_c(L_{c,i})$ для $L_{c,i} = 2^i$, $i = 0, ..., m$.

$$\tau_0 = \max_{i=0,...,m} T_c(L_{c,i})/M = T_c(1)/M = 10.0/M \approx 3.0 \cdot 10^{-7}$$

$$\tau_c = \min_{i=0,...,m} T_c(L_{c,i})/M = 0.10/M \approx 3.0 \cdot 10^{-9}$$

Величины 10.0 и 0.10 – экспериментальные данные

$$\tau_0/\tau_c \approx 100$$

Время передачи сообщений...

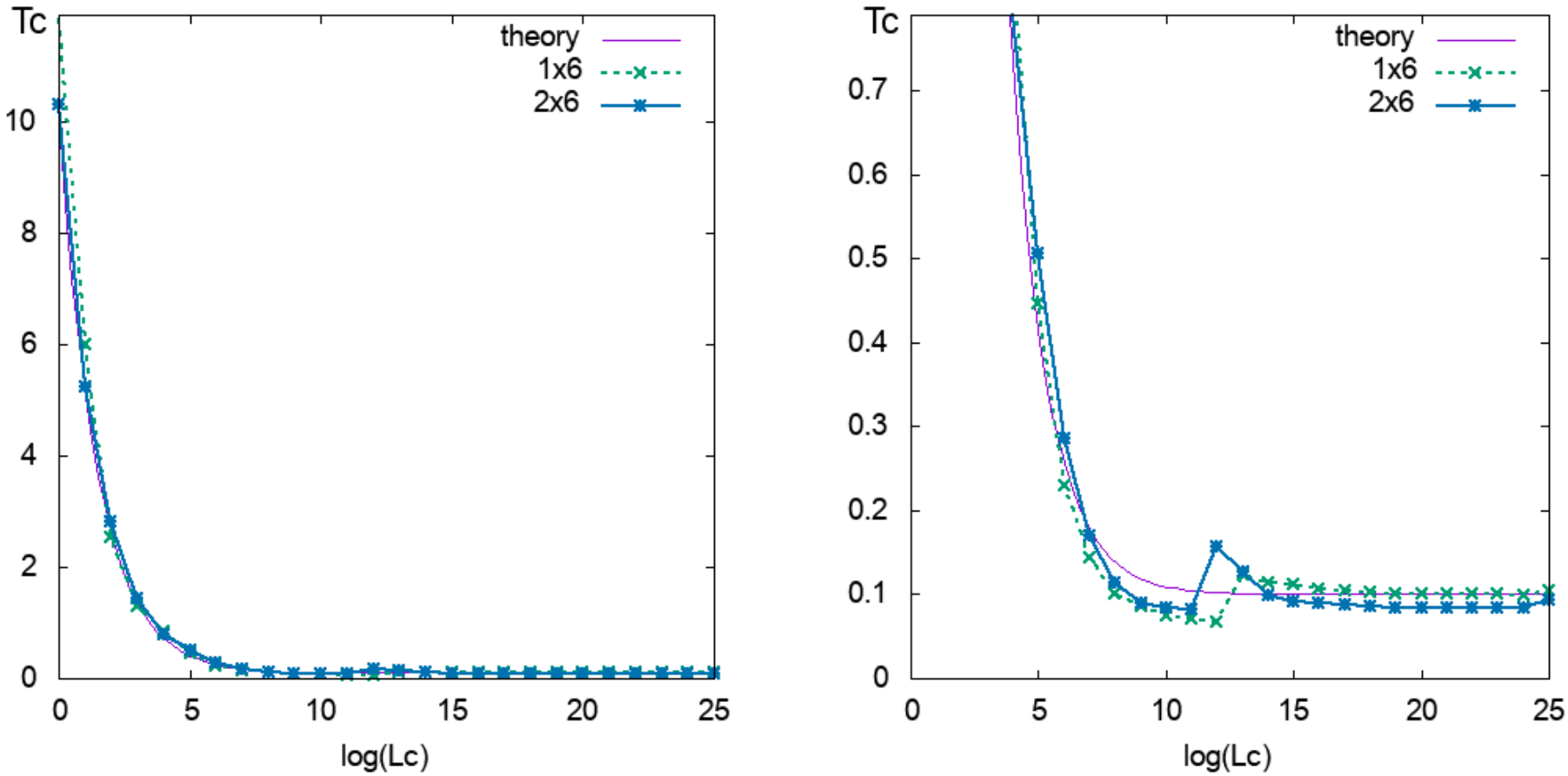


Figure : Общее время пересылки сообщения длиной 2^{25} слов типа “double” порциями длины L_c. Теоретическая оценка и расчеты на сегменте “x6core”.

Уточнение оценок

$$T_c = n_c \tau_0 / q + \tau L_c$$

L_c – общая длина всех обменов (в пересчете на один процессор)
q – кол-во слоев перекрытия подобластей (при этом необходимые обмены выполняются один раз на q шагов по времени)
n_c – общее кол-во обменов на каждом из процессоров

$$T_a = (1 + Q)\tau_a L_a$$

L_a – общее кол-во арифметических операций (в пересчете на один процессор)
Q – доля увеличения кол-ва арифметических операций, если в алгоритме, для экономии количества (или длины) обменов, было решено дублировать некоторые арифметические операции

$$\begin{aligned} S &= S(p, \tau_0, Q) = T(1)/T(p) = T_a(1)/(T_a(p) + T_c(p)) \\ &= \tau_a L_a / ((1 + Q)\tau_a L_a / p + n_c \tau_0 / q + \tau_c L_c / p) \\ &= p/(1 + Q + \tau L + p n_c \tau_0 / (q \tau_a L_a)) \end{aligned}$$

Оценки для модельной задачи

Доля дублирования вычислений для q слоев перекрытия:

$$Q = \frac{q(q-1)}{2}\frac{L_c}{L_a} = \frac{q(q-1)}{2}L$$

$Q = 0$ – дублирования вычислений нет (при $q = 1$)

$Q = 1$ – дублирование двукратное при $q \approx \sqrt{2/L}$.

Ранее, без дублирования, такое же падение //-ной эфф-ти происходило при $\tau L = 1$.

Рекомендуется брать $q < \sqrt{2\tau}$ или $q < 5$ для $\tau \approx$ 10-:-30-:-100.

$$S = p \Big/ \left(1 + \left(\tau_{ca} + \frac{q(q-1)}{2}\right)\left(2 - \frac{2}{p^{1/D}}\right)\frac{DV}{C}\frac{p^{1/D}}{N^{1/d}} + \frac{2D}{Cq}\frac{p}{N}\tau_{0a}\right)$$

где

$$\tau_{ca} = \tau = \tau_c/\tau_a, \qquad \tau_{0a} = \tau_0/\tau_a$$

Оптимальный размер перекрытия подобластей

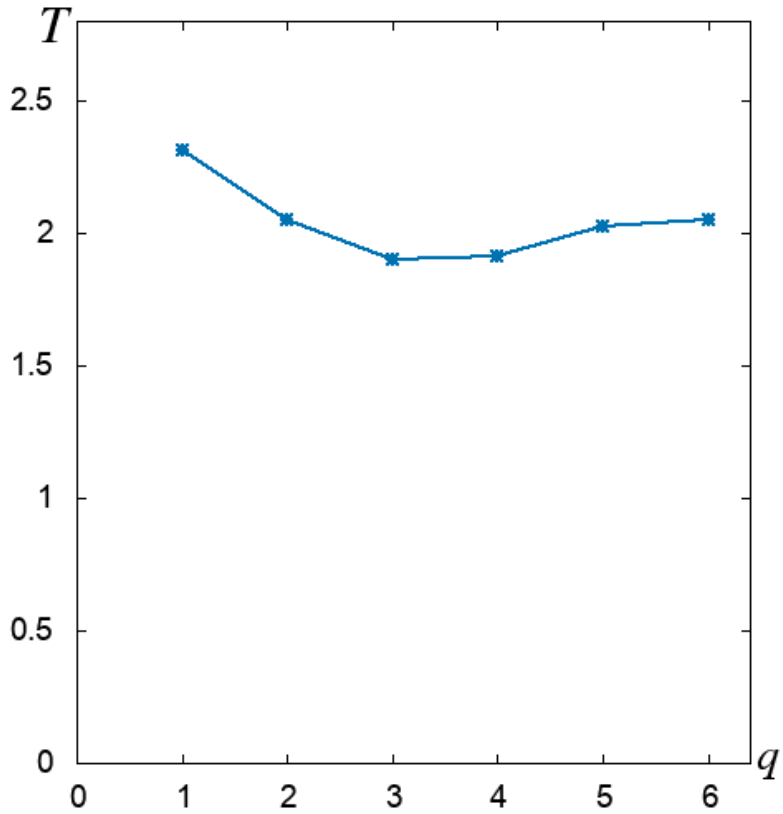


Figure : Время решения в зависимости от размера перекрытия подобластей $q = 1, ..., 6$ для параметров:

$$100p \times 100 \times 100, \quad d = 3, \quad D = 1, \quad p = 64, \quad q = 1, ..., 6$$

Ускорение: теория и эксперимент ($q = 1$)

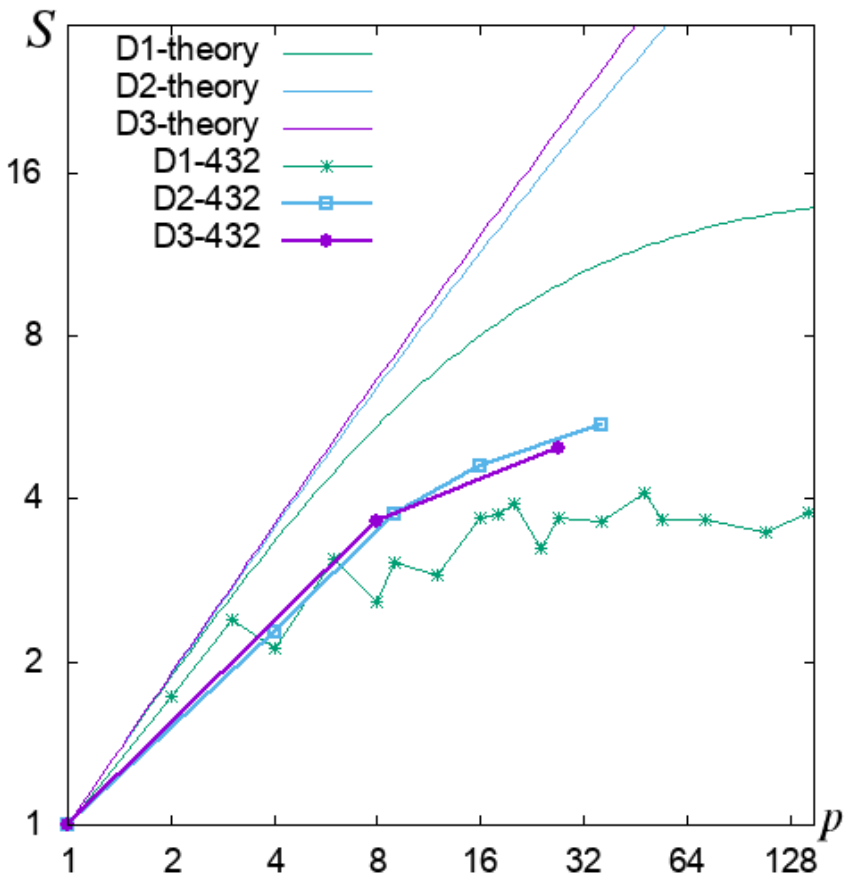


Figure : Ускорение при $D = 1, 2, 3$ для задачи $432 \times 432 \times 432$.

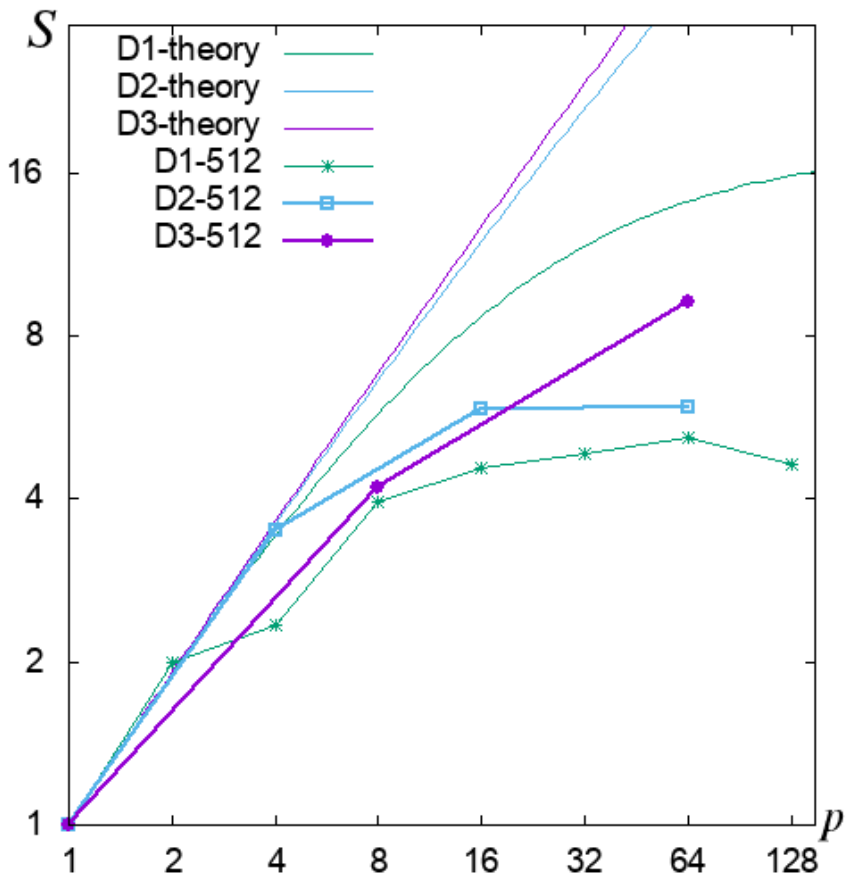


Figure : Ускорение при $D = 1, 2, 3$ для задачи $512 \times 512 \times 512$.

Литература

- Г.В.Байдин, О некоторых стереотипах параллельного программирования. Вопросы атомной науки и техники, Серия: Матем. модел. физ. процессов, 2008, No. 1, 67–75
- В.С.Гладких, Построение предсказательных моделей времени параллельной аппроксимации, XXII конф. им. К.И.Бабенко, Абрау-Дюрсо, 2018, с.38
- В.Д.Левченко, Локально-рекурсивные нелокально-ассинхронные алгоритмы и их приложения, XXII конф. им. К.И.Бабенко, Абрау-Дюрсо, 2018, с.64–65
- И.Н.Коньшин, Модели параллельных вычислений для оценки реального ускорения исследуемого алгоритма (*линейная алгебра*), Абрау-Дюрсо, 2016
  - —, RuSCDays, 2016, 269–280 (http://2016.russianscdays.org/files/pdf16/269.pdf)
  - —, Springer, CCIS 687, 2017, 304–317
- И.Н.Коньшин, Оценка эффективности алгоритмов *математической физики* для компьютеров с распределенной памятью, Абрау-Дюрсо, 2018 (http://dodo.inm.ras.ru/~konshin/papers/2018-Abrau-algo-slides-ru.pdf)
  - —, RuSCDays, 2018 (http://dodo.inm.ras.ru/~konshin/papers/2018-RuSCDays-algo-ru.pdf)
  - —, Springer, 2018