HPC-4:

параллельная эффективность алгоритмов **математической физики** для компьютеров с распределенной памятью

И.Н. Коньшин[1,2,3]

[1]Институт вычислительной математики им. Г.И. Марчука, РАН
[2]Московский физико-технический институт
[3]Сеченовский университет

План

Ситуация:

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

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
- уточнение оценок
- результатов численных экспериментов

Сначала: немного повторения и *неявные* методы матфизики

Оценка //-ной эффективности (общая память)

Закон Амдала [Amdahl's law, 1967]

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

Ускорение:

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

// (параллельная) 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 эфф-ти (распределенная память) [ИНК, 2012]

Пусть

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

являются характеристиками «параллелизма» 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:

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

Итерационные методы: модель для 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 некоторой задачи ***матфизики***, решаемой ***неявным*** методом:

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

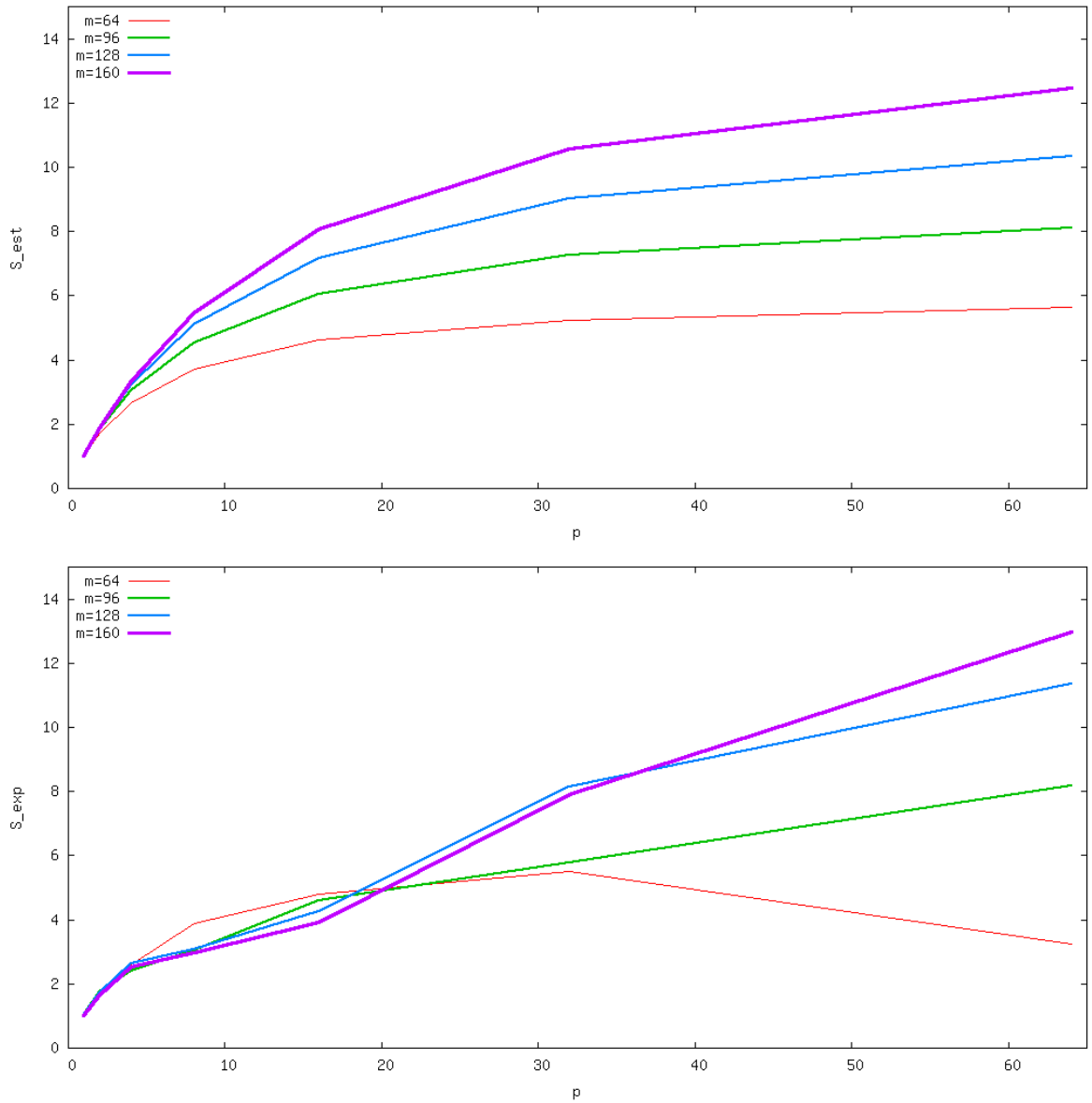


Рис.: Ускорение (теория и эксперимент) для $n = 64, 96, 128, 160$

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

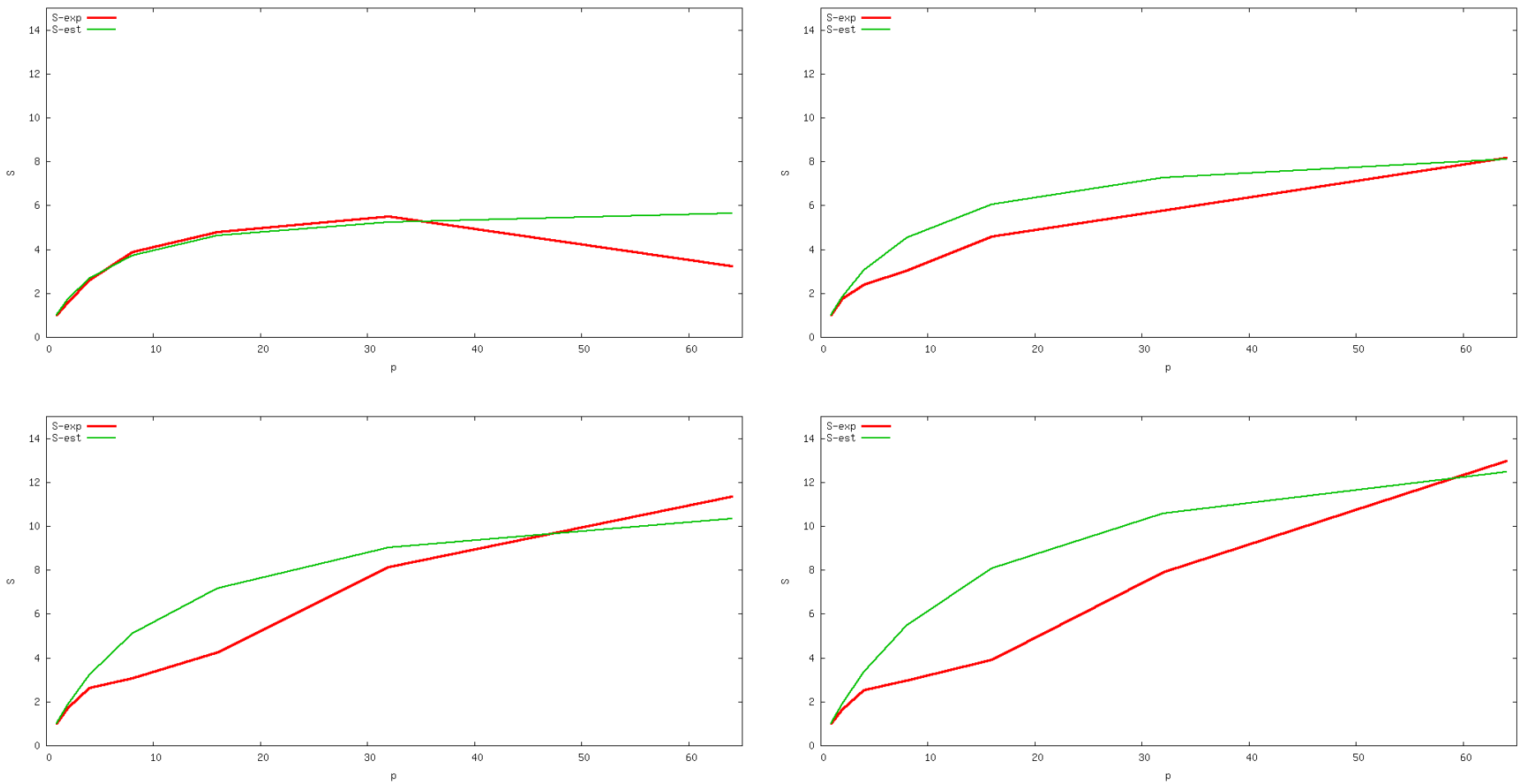
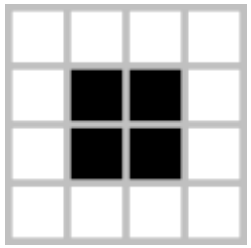
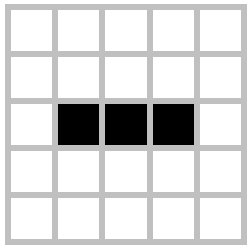
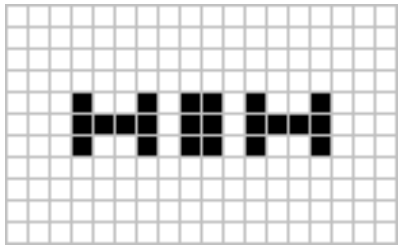
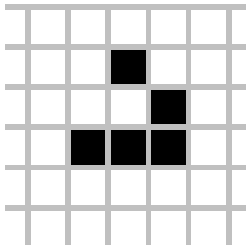
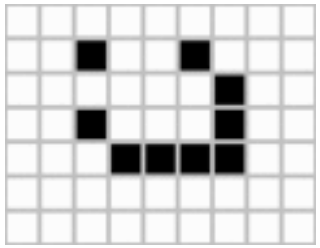


Рис.: Ускорение (теория и эксперимент) для $n = 64, 96, 128, 160$

Сначала: *модель* явных методов матфизики

2D игра “Жизнь”

- придумано английским математиком Джоном Конвеем в 1970 году (как идея создания гипотетической машины, воспроизводящей саму себя)
- Место действия – “вселенная” (например, ограниченная 2D область из “клеток”)
- Правила: если 2 или 3 соседа, то клетка живет, если нет – то умирает; а если ровно 3 соседа, то оживает
- Может ли общее количество живых клеток расти?
- Фигуры: устойчивые, периодические, долгожители, движущиеся по диагонали и вертикали. А также: ружья, паровозы, пожиратели, отражатели, размножители, “райские сады”...

2D игра “Жизнь”

https://www.youtube.com/watch?v=yw-j-4xYAN4 (5:45) ч.1 figures фигуры
https://www.youtube.com/watch?v=m4LtaZNHN9E (5:07) ч.2 blinker пульсары
https://www.youtube.com/watch?v=RSwoLHs4ZzQ (5:46) ч.3 spaceship=shuttle
https://www.youtube.com/watch?v=p4uqKtXkb3E (5:38) ч.4 Gosper glider gun
https://www.youtube.com/watch?v=PNK_a0cl9W0 (4:13) ч.5 intermission: столкновения
https://www.youtube.com/watch?v=j4aV9xm3_mk (6.22) ч.6 Puffer train, B-heptomino,
https://www.youtube.com/watch?v=xHt-kxnEwoo (5:32) ч.7 Breeder, Golly
https://www.youtube.com/watch?v=1l5ie_owyik (6:34) ч.8 OTCA metapixel, etc.
https://www.youtube.com/watch?v=hF_MZ1WO24U (10:40) ч.9 самовоспроизведение

https://www.youtube.com/watch?v=Kk2MH9O4pXY (23:32) история
https://www.youtube.com/watch?v=-FaqC4h5Ftg (2:00) битва (!)
https://www.youtube.com/watch?v=xP5-iIeKXE8 (1:29) фрактал: life in life (!)
https://www.youtube.com/watch?v=cTZkEAYeRis (2:05) Mandelbrot set
https://www.youtube.com/watch?v=3NDAZ5g4EuU (2:01) работающие часы (!)
https://www.youtube.com/watch?v=8unMqSp0bFY (1:33) programmable computer

https://www.youtube.com/watch?v=dTrNHiBbW8I (3:36) Life in Matlab
https://www.youtube.com/watch?v=2z42cnB0NTY (5:42) Life in Minecraft
https://www.youtube.com/watch?v=d73z8U0iUYE (9:54) Life in python

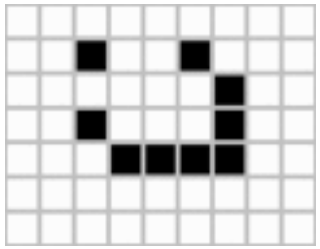
https://conwaylife.com/wiki/OTCA_metapixel

Life: цветная, треугольная, гексагональная, на ограниченном поле, на торе, 3D, ...

2D игра “Жизнь” – влияние на:

- Теория автоматов
- Теория алгоритмов
- Теория игр и математическое программирование
- Алгебра и теория чисел
- Теория вероятностей и математическая статистика
- Комбинаторика и теория графов
- Фрактальная геометрия
- Вычислительная математика
- Теория принятия решений
- Математическое моделирование

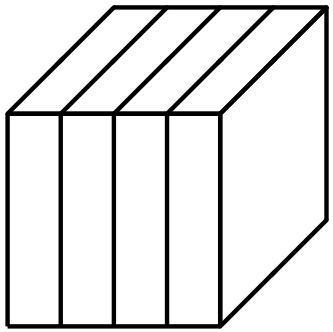
2D игра “Жизнь” – как практическое задание



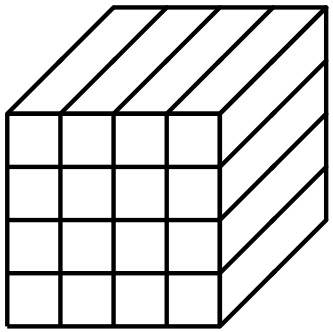
- Случайное начальное распределение живых клеток.
- Периодические условия на границе с замыканием квадратной области в тор.
- Контроль стационарности/периодичности состояния.
- Исследовать несколько правил “оживания/умирания” клетки.
- Интересуют правила, дающие наиболее длительные вариации состояния поля.
- 2D разбиение исходного поля по процессорам.
- *Самые смелые могут попробовать реализовать 3D вариант игры* (см. ссылки на https://en.wikipedia.org/wiki/3D_Life).

Теперь: настоящие ***явные*** методы матфизики

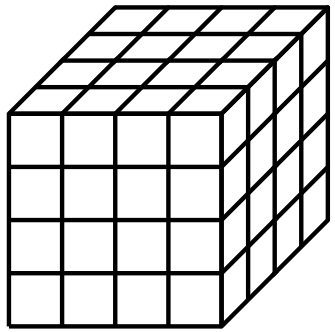
Модельная задача матфизики



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



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



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

Рис.: Распределение данных по процессорам в трехмерной задаче математической физики для r слоев в одном, двух и трех направлениях

D — кол-во измерений при расщеплении расчетной области
d-мерный “куб” ($d = 1, 2, 3$) с n ячейками в каждом измерении
$N = n^d$ — общее кол-во d-мерных кубических ячеек
V — кол-во неизвестных функций в ячейке
$(2d + 1)$-точечный шаблон d-мерной дискретизации
C — кол-во арифм. операций на ячейку на ***явном*** шаге по времени

Оценка //-ной эффективности

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

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

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

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

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

Оценки для конкретного случая

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

Таблица: Теоретические оценки //-ной эффективности для конкретного набора параметров:

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

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 C версии 4.0.1 с библиотекой Intel MPI версии 5.0.3. http://cluster2.inm.ras.ru/

Обмены на фоне счета: Тест 1 [Байдин'2008]

Асинхронные обмены данными через MPI_Isend(), MPI_Irecv(), MPI_Waitall() выполнялись 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

Таблица: Время выполнения Теста 1 для асинхронных обменов данными

Ситуации:
$p = 1$ и $N_{\text{drops}} = 1$ – синхронный обмен данными
$p = 2$ – обмены на одном вычислительном узле
$p = 13$ – обмены на двух различных узлах сегмента xbcore

Обмены на фоне счета: Тест 1 [Байдин'2008]

```
ndrop=1; sizi=siza/ndrop;
for (int j=0; j<=max_it; j++) { //global iterations
    nBeg=0;
    for (int k1=0; k1<ndrop; k1++) { //drop iteration:
        for (int k=0; k<sizi; k++) { //drop iteration:
            ij = nBeg+k;
            V[ij] = cos(ij*1e-10) * V[ij] * cos(ij*1e-9)
                  + sin(ij*1e-10) * V[ij] * sin(ij*1e-9);
        }
        MPI_Isend(V+nBeg, sizi, MPI_DOUBLE, Targ, tag,
                  MPI_COMM_WORLD, reqst+(2*k1));
        MPI_Irecv(V+nBeg, sizi, MPI_DOUBLE, Targ, tag,
                  MPI_COMM_WORLD, reqst+(2*k1+1));
        nBeg += sizi;
    }
    MPI_Waitall(ndrop*2, reqst, &status);
}
```

Время передачи сообщений: Тест 2 [Байдин'2008]

Одно большое сообщение общей 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 $L_{c,i} = 2^i$, $i = 0, ..., m$.

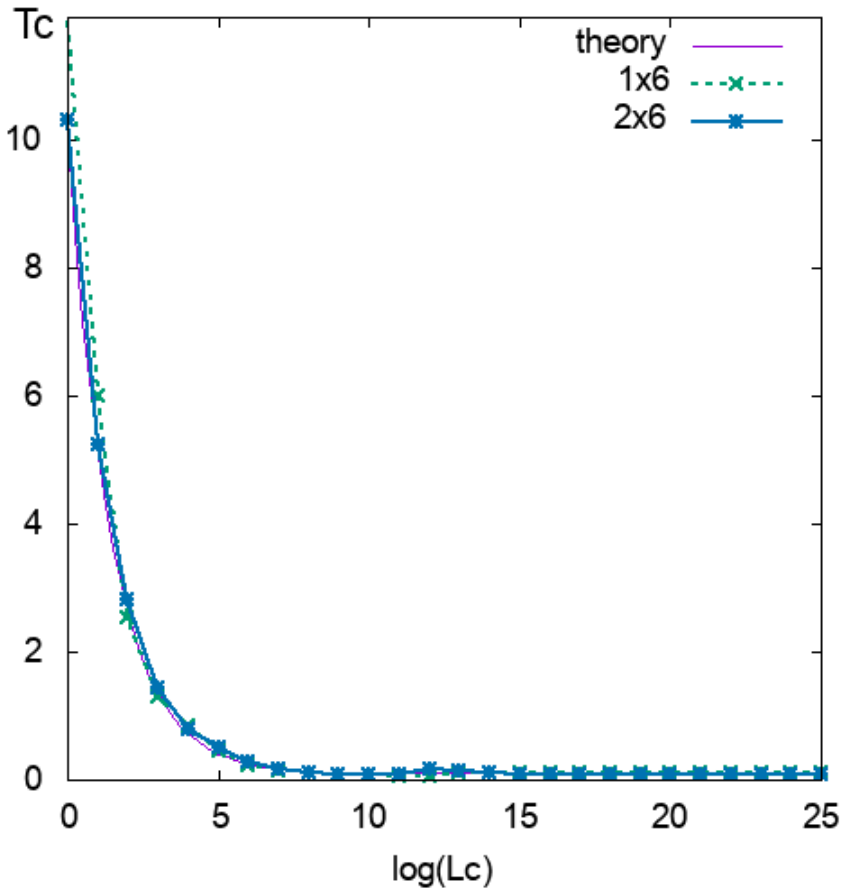
$$\tau_0 = \max_{i=0,...,m} T_c(L_{c,i})/M = T_c(1)/M = 10.0/M \approx 3.0 \cdot 10^{-7}$$

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

Значения 10.0 и 0.10 являются экспериментальными данными, т.е.

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

Время передачи сообщений...



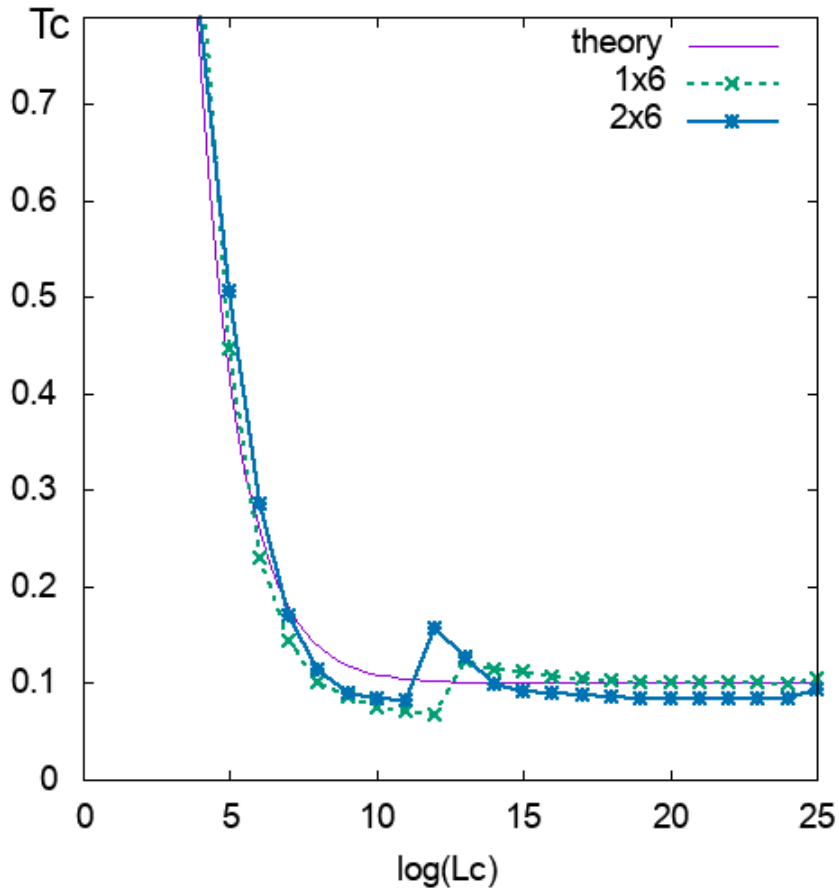


Рис.: Общее время отправки сообщения из 2^{25} слов double порциями длины L_c. Теоретическая оценка и фактические расчеты на сегменте x6core.

Уточнение оценок //-ной эффективности

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

L_c — общая ***длина*** всех коммуникаций (в пересчете на один процессор)
n_c — общее ***кол-во*** сообщений на одном процессоре
q — кол-во слоев перекрытия подобластей (при этом необходимые обмены выполняются один раз на q шагов по времени)

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

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

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

Оценка //-ной эфф-ти для модельной задачи

Доля дублирования вычислений для q слоев перекрытия:

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

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

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

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

Т.е. рекомендуется использовать $q < \sqrt{2\tau}$ 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

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

Оптимальная ширина перекрытия подобластей q

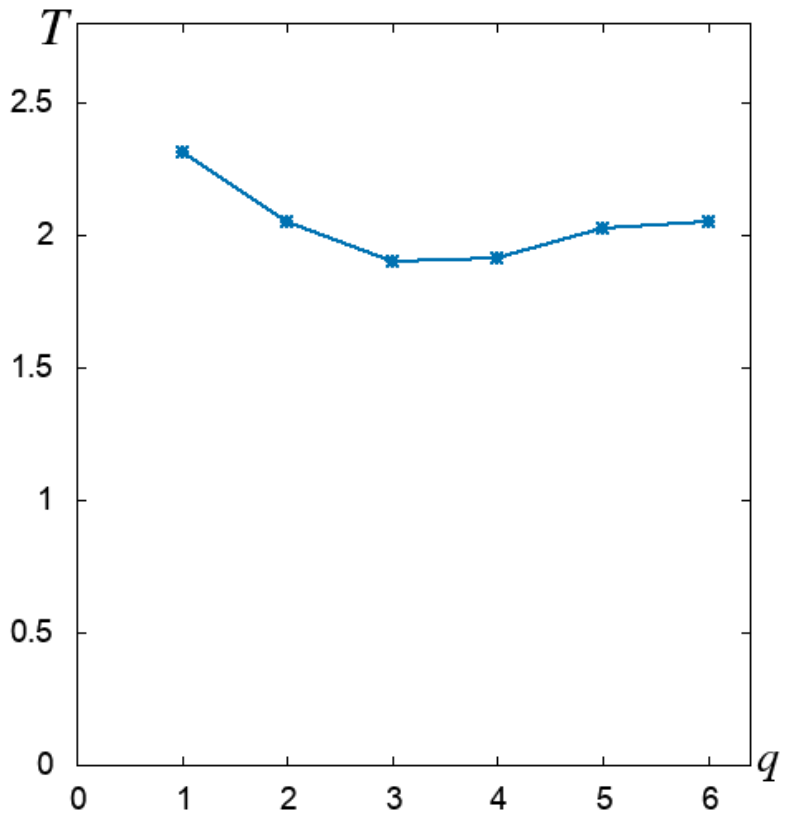


Рис.: Время решения в зависимости от ширины перекрытия подобластей $q = 1, ..., 6$ для конкретных значений параметров:

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

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

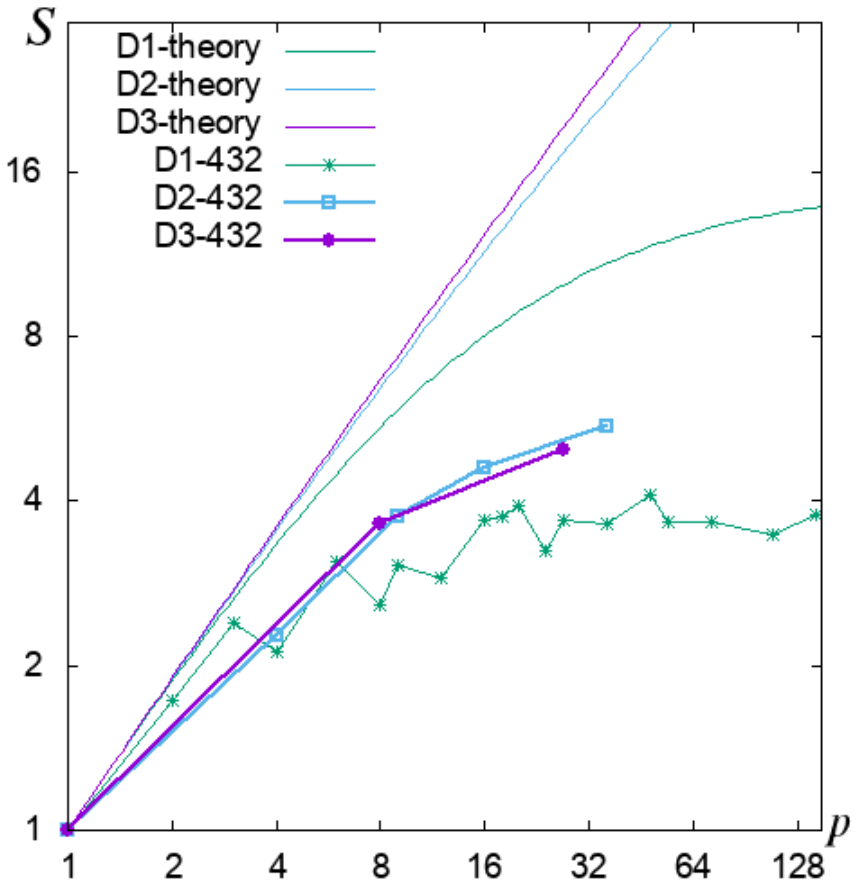


Рис.: Ускорение при $D = 1, 2, 3$ для задачи $432 \times 432 \times 432$ ($432 = 2^4 \cdot 3^3$).

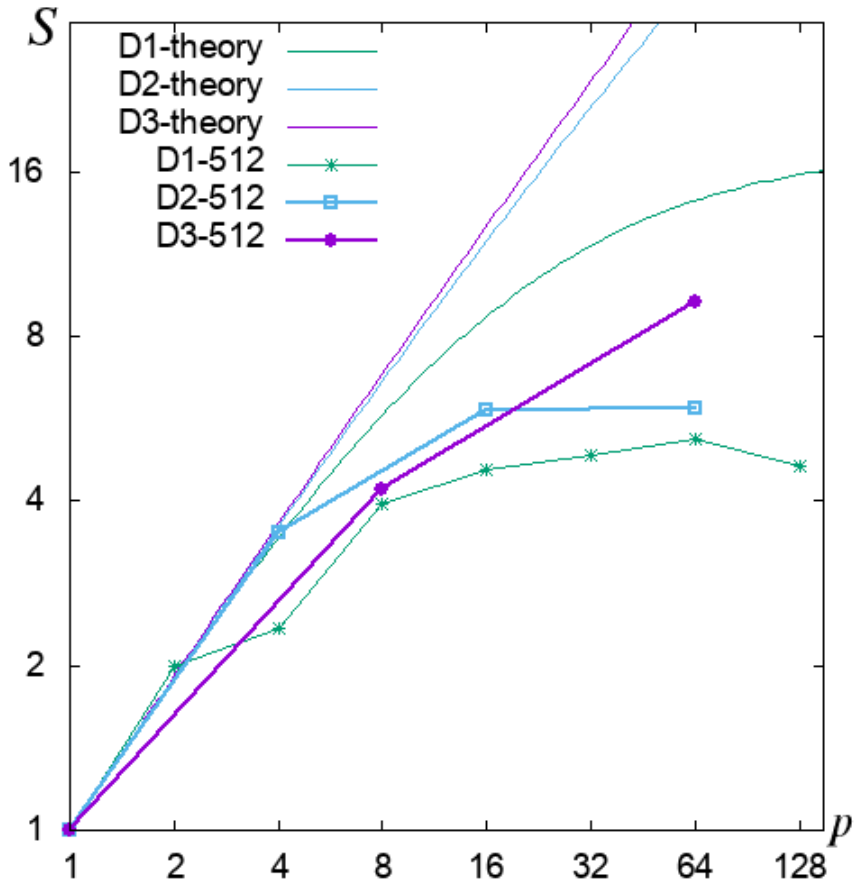


Рис.: Ускорение при $D = 1, 2, 3$ для задачи $512 \times 512 \times 512$ ($512 = 2^9$).

Литература

- Г.В. Байдин, О некоторых стереотипах параллельного программирования. Вопросы атомной науки и техники, Серия: Математическое моделирование физических 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, CCIS 965, 2019, 63–75