**Федеральное государственное автономное**

**образовательное учреждение высшего образования**

**«Санкт-Петербургский национальный исследовательский университет**

**информационных технологий, механики и оптики»**



Факультет Программной Инженерии и Компьютерной Техники

**Лабораторная работа №1**

по дисциплине

**«Операционные системы»:**

Вариант proc-clone3 shell-seq cpu-factorize ema-traverse-graph  
Выполнили:  
Ястребов-Амирханов Алекси  
 Группа: Р3332  
ПРЕПОДАВАТЕЛЬ: Тюрин Иван Николаевич

Санкт-Петербург,

2025

Оглавление

[Цели работы 3](#_Toc215635766)

[Исходный код программ 3](#_Toc215635767)

[Предположения о свойствах программ-нагрузчиков 4](#_Toc215635768)

[io-load 4](#_Toc215635769)

[cpu-factorize 7](#_Toc215635770)

[ema-traverse-graph 9](#_Toc215635771)

[Этап A — *Генерация (gen)* 9](#_Toc215635772)

[Этап B — *Подготовка к обходу (trv)* 10](#_Toc215635773)

[Этап C — *Сам обход (DFS стек)* 10](#_Toc215635774)

[Последовательный режим / bias→forward 12](#_Toc215635775)

[Случайный режим 12](#_Toc215635776)

[Память 12](#_Toc215635777)

[Результаты измерений и метрик программ-нагрузчиков 13](#_Toc215635778)

[io-load 13](#_Toc215635779)

[cpu-factorize 26](#_Toc215635780)

[ema-traverse-graph 29](#_Toc215635781)

[Параллелизм программ-нагрузчиков 32](#_Toc215635782)

[io 33](#_Toc215635783)

[cpu 67](#_Toc215635784)

[Нагрузчики с агрессивной оптимизацией 74](#_Toc215635785)

[io 74](#_Toc215635786)

[Сравнение компиляции без оптимизации и с агрессивной оптимизацией 94](#_Toc215635787)

[io 94](#_Toc215635788)

[cpu 99](#_Toc215635789)

[Многопоточные нагрузчики 103](#_Toc215635790)

[cpu 103](#_Toc215635791)

[io 107](#_Toc215635792)

# Цели работы

1. Реализовать собственную оболочку командной строки - shell. Shell должен предоставлять пользователю возможность запускать программы на компьютере с переданными аргументами командной строки и отображать время их выполнения. В разработанное приложение командной оболочки необходимо добавить поддержку логического оператора OR ( || в bash), запуск подпроцесса с помощью proc-vfork

1. Разработать параметризируемую программу-нагрузчик, которая будет однопоточно нагружать подсистему ввода-вывода (IO).
2. Разработать комплекс программ-нагрузчиков:
   1. Нагрузчик вычислительной подсистемы - cpu-sort: Любая сортировка массива чисел за N\*log(N) и любая за N^2
   2. Нагрузчик, работающий с внешней памятью - ema-join-hash: объединение таблиц в файлах

Разработанные программы необходимо скомпилировать без дополнительных опций оптимизации компилятора.

# Исходный код программ

Shell - <https://github.com/AlexandroLe/os-course/blob/lab-1/lab/vtsh/bin/main.c>

IO-нагрузчик - <https://github.com/AlexandroLe/os-course/blob/lab-1/lab/vtsh/bin/io.c>

CPU-нагрузчик - <https://github.com/AlexandroLe/os-course/blob/lab-1/lab/vtsh/bin/cpu.c>

EMA-нагрузчик - <https://github.com/AlexandroLe/os-course/blob/lab-1/lab/vtsh/bin/ema.c>

# Предположения о свойствах программ-нагрузчиков

## io-load

**1. Назначение программы**

Программа io-load предназначена для создания управляемой нагрузки на дисковую подсистему за счёт чтения и/или записи блоков данных в файл.  
Она позволяет моделировать различные сценарии ввода-вывода:

* чтение или запись (read, write, rw);
* последовательный или случайный доступ (sequence, random);
* использование или обход кэша операционной системы (O\_DIRECT);
* управление размером блока и количеством операций.

Программа является **однопоточной** и активно использует системные вызовы read, write и lseek.

**2. Характер нагрузки и узкое место**

Основное узкое место программы — **подсистема ввода-вывода**, а не процессор.

* Работа с буфером и генерация адресов выполняются быстро и занимают незначительное время.
* Основные задержки возникают при обращении к файловой системе и блочному устройству.
* Следовательно, программа является **I/O-bound**, а не CPU-bound.

Даже при компиляции без оптимизаций, накладные расходы на работу процессора минимальны по сравнению с задержками доступа к диску.

**3. Объём обрабатываемых данных**

Пусть в эксперименте используется конфигурация:

* размер блока: **1 MiB**
* количество блоков: **9000**

Тогда суммарный объём данных:

9000 × 1 MiB = 9000 MiB ≈ 8.89 GiB

Таким образом, программа будет выполнять операции ввода-вывода над объёмом данных порядка **9 ГиБ**.

**4. Оценка времени выполнения**

Время выполнения напрямую зависит от:

* типа носителя (HDD / SSD / NVMe);
* режима работы (read / write);
* типа доступа (sequence / random);
* использования флага O\_DIRECT.

Для упрощения оценки предполагаем, что производительность диска нахыцвэ\одится в пределах характерных значений для современных систем.

**4.1 Последовательный доступ**

При последовательном доступе пропускная способность ограничивается скоростью линейного чтения или записи.

| **Режим** | **Ожидаемое время** |
| --- | --- |
| read, sequence | 20-70 секунд |
| write, sequence | 90–160 секунд |

* Чтение обычно быстрее записи.
* Кэш ОС может дополнительно ускорить операции чтения при повторных запусках.

**4.2 Случайный доступ**

Случайный доступ приводит к множеству перемещений по файлу и увеличению задержек, особенно на магнитных дисках.

| **Режим** | **Ожидаемое время** |
| --- | --- |
| read, random | 180–250 секунд |
| write, random | 400–500 секунд |

* Производительность определяется числом операций ввода-вывода (IOPS).
* Накладные расходы на lseek и системные вызовы значительно возрастают.

**4.3 Использование O\_DIRECT**

При включённом режиме прямого ввода-вывода (O\_DIRECT):

* обходится кэш страниц ОС;
* каждая операция приводит к реальному обращению к диску;
* уменьшается искажение результатов за счёт буферизации.

Ожидаемые последствия:

| **Режим** | **Ожидаемое время** |
| --- | --- |
| read, sequence, direct | 20–50 секунд |
| write, random, direct | 210–270 секунд |

**5. Ожидаемый профиль загрузки системы**

**Последовательный доступ**

* Низкая загрузка CPU (USER — минимален).
* Умеренный SYS (10–20%).
* Небольшой или средний IO-wait.
* Процесс большую часть времени активен, очередь запросов короткая.

**Случайный доступ**

* Значительный рост IO-wait (до 40–60%).
* Увеличение SYS из-за большого числа системных вызовов.
* Время выполнения в 2–4 раза выше, чем при последовательном доступе.

**Режим O\_DIRECT**

* Уменьшение роли оперативной памяти.
* Более стабильная и «честная» нагрузка на диск.
* Процесс по-прежнему I/O-bound.

**6. Вывод**

Перед запуском нагрузочного теста можно ожидать, что:

* время выполнения программы линейно зависит от объёма данных и обратно пропорционально производительности диска;
* случайный доступ существенно медленнее последовательного;
* использование O\_DIRECT повышает достоверность результатов, но увеличивает время выполнения;
* программа в основном нагружает дисковую подсистему и слабо влияет на вычислительные ресурсы процессора.

Таким образом, ожидаемые результаты соответствуют типичным характеристикам **I/O-нагруженных однопоточных приложений**.

## cpu-factorize

**1. Назначение программы**

Программа выполняет факторизацию целого числа N (64-битное uint64\_t).  
Реализованы основные инструменты:

* проверка простоты — тест Миллера–Рабина (фиксированный набор баз: 7 баз, достаточно для 64-бит);
* факторизация — рекурсивный Pollard-Rho;
* печать множителей с учётом кратности;
* измерение времени выполнения (монотонный таймер).

Программа — **однопоточная**, полностью CPU-bound, основной вес в вычислениях — арифметика по модулю (mulmod, powmod), генерация случайных чисел (rand()), рекурсивный разбор.

**2. Входные параметры и поведение**

* Параметры: cpu-factorize [-r repeats] <number>.
* repeats — сколько раз повторить факторизацию (полезно для усреднения).
* Для number < 2 печатается тривиальный результат.
* Для составных чисел программа выводит множители и время каждой прогона.

**3. Алгоритмическая сложность (какие операции доминируют)**

Коротко:

* Miller-Rabin: несколько (здесь — до 7) раундов быстрого возведения в степень по модулю. Стоимость одной операции ≈ O(log^2 N) машинных операций умножения/сложения для uint64\_t (на практике — небольшая константа).
* Pollard-Rho: вероятностный алгоритм; ожидаемое число итераций для обнаружения делителя p ≈ O(p^{1/2}) групповых операций (чуть упрощённо — зависит от наименьшего нетривиального делителя). Для случайного N с примерно равными факторами (семиприм) сложность растёт как O(N^{1/4}) в количестве базовых многократно-модульных операций.

Вывод: **время сильно зависит от структуры числа**:

* если N имеет малый простой делитель p — факторизация быстрая (найдётся за малое число итераций);
* если N — полуслучайный семиприм (два примерно равных больших простых), то время — максимум и определяется поведением Pollard-Rho (≈ N^{1/4} сложность операций).

**4. Влияние компиляции и окружения**

* Программа обычно собирается **без оптимизаций** в лаборатории, поэтому абсолютные времена будут выше, чем при -O2/-O3.
* rand() и realloc/malloc — дополнительные накладные расходы, но основная часть времени уходит в умножения/возведения в степень по модулю.
* Однопоточный код полностью загружает **одно ядро CPU** — при многопроцессном запуске время делится между ядрами.

**5. Практические оценки времени выполнения**

Важно: оценки даны как ориентиры. Точные числа зависят от CPU (частота, микроархитектура), от компиляции (-O0 vs -O3), и — критично — от структуры числа (малый делитель ускоряет работу).

Для наглядности используем примеры чисел разного порядка:

* «малые» — до ~1e9 (≈32-бит)
* «средние» — ≈1e12–1e15
* «большие» — ≈1e18 (порядок 64-бит)
* «сложные семипримы» — специально подобранные числа, у которых нет маленьких делителей

**Примеры ориентировочных времён (один запуск, компиляция без оптимизаций, современное одно-ядро ноутбука):**

| **Число (приблизительно)** | **Ожидаемое время (один запуск)** | **Комментарий** |
| --- | --- | --- |
| ≤ 1e9 (32-бит) | **< 0.05 с** | Практически мгновенно; MR + Pollard почти не затратны. |
| ~1e12 (12 цифр) | **0.01–0.3 с** | Быстро, чаще <0.1 s; если есть небольшой делитель — ещё быстрее. |
| ~1e15 (15 цифр) | **0.05–1.0 с** | Время растёт; большой разброс зависит от факторизации. |
| ~1e18 (18 цифр, ~64-бит) | **0.1–5 s** | Для «средних» 64-бит чисел обычно в долях секунды — до нескольких секунд; для специально труднофакторизуемых семипримов — до нескольких секунд/десятков секунд. |
| «Сильные» 64-бит семипримы | **1–30 s (иногда дольше)** | Хуже по случаю: Pollard-Rho может потребовать много итераций; здесь разумно ожидать секунд–десятков секунд. |
| ~128-бит и больше | **непредсказуемо / нереалистично** | Для 128-бит числа Pollard-Rho уже практически бесполезен; требуется ECM/GNFS и много времени. |

**6. Профиль нагрузки**

**CPU**

* Почти всё время — в **пользовательском режиме** (USER ≈ 98–100% для одного ядра).
* Вызовы в system минимальны (SYS < 1–2%) — лишь аллокации/печать/системный таймер.
* IO-wait ≈ 0% (I/O почти отсутствует), т.е. программа **CPU-bound**.

**Память**

* Массивы/векторы множителей — максимально немного (количество множителей ≪ log₂N).
* Потребление RAM малое и стабильно, нет долгоживущих больших аллокаций (кроме небольшого массива factors через realloc).

**Контекстные переключения**

* Небольшое количество: рекурсивные вызовы и локальные аллокации не создают интенсивных переключений.

## ema-traverse-graph

**1. Назначение программы**

Программа состоит из двух главных режимов:

* gen — генерация k-регулярного ориентированного графа в бинарный файл. Для каждой вершины записывается 32-битное значение и k соседей (uint32). Параметры: число вершин, k, seed, max\_value, bias (вероятность выбрать соседа с индексом > текущего).
* trv — обход графа (стековая реализация DFS), начиная со start, поиск вершины со значением find\_value и замена его на replace\_value (до max\_modify изменений), с ограничением глубины max\_depth. Использует bitmap visited, стек вершин и буфер соседей.

Файловая организация: заголовок + подряд записанные записи вершин; длина записи = 4 \* (1 + k) байт.

**2. Этапы выполнения (чёткое разделение)**

### Этап A — Генерация (*gen*)

1. Открытие/создание файла (wb+).
2. Запись заголовка.
3. Для каждой вершины:
   * сгенерировать значение (rand() % value\_max);
   * сгенерировать массив k соседей (с bias или без);
   * записать запись в файл (fseek/fwrite).

**Характер:** последовательные записи; хорошая потоковая локальность; при обычном buffered I/O — высокая пропускная способность.

### Этап B — Подготовка к обходу (*trv*)

1. Открытие файла (rb+), чтение заголовка.
2. Выделение структуры: visited bitmap, стек вершин, буфер neighbours\_buf.

**Память:** visited bitmap = ≈ node\_count / 8 байт; буферы малого размера — O(k). Это обычно невелико.

### Этап C — Сам обход (DFS стек)

Повтор:

* поп из стека;
* проверить visited (bitmap);
* read\_node (fseek + fread) — получение значения + списка соседей;
* при совпадении — write\_node (fseek + fwrite) — модификация;
* при необходимости push соседей на стек.

**Характер:** сильно зависит от того, какие индексы соседей генерируются:

* при «bias» (вперёд) — хорошая локальность: соседние индексы близки → доступы ближе к последовательным;
* при полностью случайных соседях — случайные обращения (плохая локальность).

**3. Память (оценка)**

* visited\_bitmap = ceil(node\_count / 8) байт. Примеры:
  + 100k узлов → ~12.5 KB
  + 1M → ~125 KB
  + 100M → ~12.5 MB
* stack\_nodes, stack\_depths: начальная ёмкость = 1024, может расти (realloc) в зависимости от структуры графа.
* neighbors\_buf = 4 \* k байт (напр., k=8 → 32 B).  
  Итого: даже для десятков миллионов узлов память для bitmap остаётся аккуратной (мегабайты — не гигабайты), поэтому RAM обычно не узкое место.

**4. Основные факторы, влияющие на время выполнения**

1. **Число посещённых вершин V** (зависит от стартовой вершины и max\_depth).
2. **Число модификаций M** (кол-во записей; каждая вызывает запись в файл).
3. **Паттерн доступа к файлу**:
   * последовательный → throughput-bound (MB/s);
   * случайный → IOPS-bound (latency per op dominates).
4. **Наличие/отсутствие data в page cache**:
   * если файл в page cache → доступы быстрые (RAM);
   * если не в cache → обращение к устройству.
5. **Тип устройства** (HDD/SSD/NVMe) — влияет на IOPS и sequential throughput.
6. **Размер записи (node\_bytes)** — мелкие записи (20–100 B) приводят к большему числу syscalls и фрагментации в page cache.

**5. Простая модель времени**

* T ≈ T\_cpu + T\_io
* T\_io ≈ V\_reads / IOPS + M\_writes / IOPS (случайный режим)
* T\_io ≈ (V \* node\_bytes) / throughput (последовательный режим)
* T\_cpu — время обработки одного узла в памяти (bitmap проверка, push соседей, управление стеком); обычно микросекунды на узел.

**6. Численные оценки**

Допущения: node\_bytes = 4\*(1+k). Возьмём k=4 → node\_bytes = 20 B.

Примеры сценариев и оценки:

**Параметры примеров**

* Scenario 1 — small: node\_count = 100k, V = 100k (обход всех), M = 100 (малые модификации).
* Scenario 2 — medium: node\_count = 1M, V = 500k, M = 1000.
* Scenario 3 — large: node\_count = 50M, V = 5M, M = 10000.

**Устройства**

* HDD random IOPS ≈ 125 (latency ~8 ms); seq throughput ≈ 100 MiB/s.
* SATA SSD random IOPS ≈ 20 000; seq throughput ≈ 500 MiB/s.
* NVMe random IOPS ≈ 100 000; seq throughput ≈ 3000 MiB/s.

**Оценки (приблизительно):**

1. **Scenario 1 (100k visited, node\_bytes=20B)**
   * total bytes ≈ 1.9 MB.
   * Sequential (если доступ последовательный): T\_io ≈ 1.9 MB / 100 MB/s ≈ 0.02 s (HDD), SSD/NVMe — ещё меньше.
   * Random (HDD): 100k / 125 ≈ 800 s (!) — очень долго.
   * Random (SSD): 100k / 20k = 5 s.
   * Random (NVMe): 100k / 100k = 1 s.
   * CPU overhead: ~0.05–0.5 s (в памяти).
   * Итого:
     + HDD, seq: **<<1 s**;
     + HDD, random: **сотни секунд** (непригодно);
     + SSD, random: **~5–6 s**;
     + NVMe, random: **~1–2 s**.
2. **Scenario 2 (V = 500k, bytes ≈ 9.5 MB)**
   * Seq HDD: 9.5 / 100 ≈ 0.095 s.
   * Random HDD: 500k / 125 = 4000 s (~1.1 h).
   * Random SSD: 500k / 20k = 25 s.
   * NVMe random: 500k / 100k = 5 s.
   * CPU: несколько сотен ms — ~1–2 s.
   * Итого: HDD random — неподходяще; SSD random — **20–30 s**; NVMe — **5–8 s**.
3. **Scenario 3 (V = 5M, bytes ≈ 100 MB)**
   * Seq HDD: 100 / 100 = 1.0 s.
   * Random HDD: 5M / 125 = 40 000 s ≈ 11 h (практически нереально).
   * Random SSD: 5M / 20k = 250 s ≈ 4+ min.
   * NVMe random: 5M / 100k = 50 s.
   * CPU: несколько секунд — незначительно.
   * Итого: realistic random on SSD: **minutes**; seq — **seconds**.

**7. Профиль нагрузки**

### Последовательный режим / bias→forward

* CPU: низкая загрузка (~USER минимален).
* SYS: небольшая часть (I/O syscalls) — 5–15%.
* IO-wait: доминирует только если устройство медленное; при SSD — умеренный; при HDD — умеренный/высокий.
* Дисковая очередь невысока.

### Случайный режим

* CPU: низко/умеренно (обработка соседей), но программа **IO-bound**.
* SYS: возрастает (много lseek/read/write).
* IO-wait: может быть очень высоким (особенно на HDD).
* Контекстные переключения минимальны, но הרבה syscalls.

### Память

* Небольшие потребления, bitmap в памяти; если graph влазит в page cache, эффективность сильно растёт (доступы RAM вместо диска).

# Результаты измерений и метрик программ-нагрузчиков

## io-load

**Описание эксперимента (write, sequence, без O\_DIRECT)**

Была запущена программа io-load в режиме последовательной записи (write, sequence) с размером блока 1 MiB и суммарным объёмом данных около 9 GiB. Программа работает в одном потоке и использует стандартный кэш ОС (O\_DIRECT отключён).

./io write -b 1048576 -c 9000 -f ./io\_test.bin -t sequence -d off -R 1  
  
**Результаты выполнения**

Время выполнения составило 92.8 секунды, средняя пропускная способность — 96.9 MB/s. Данные значения соответствуют ожидаемым для последовательной записи на используемом устройстве хранения.

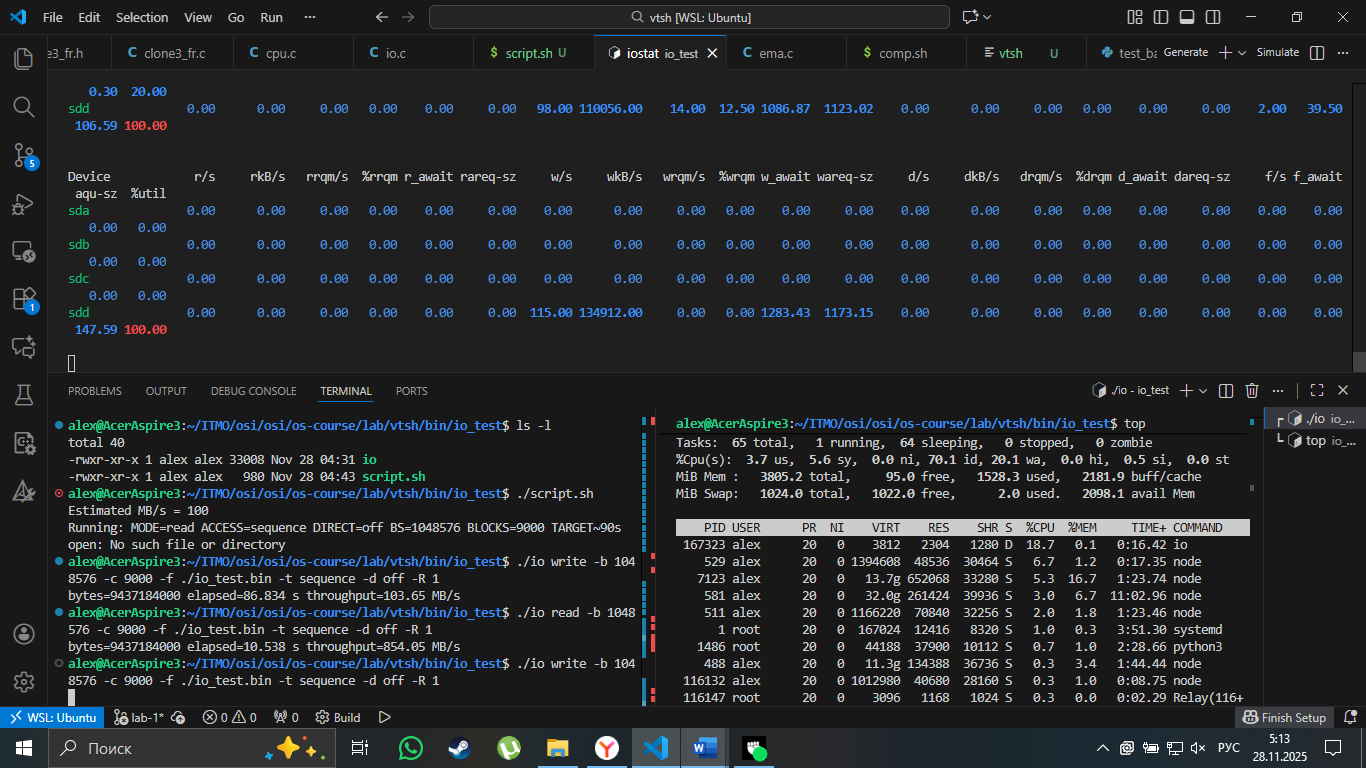
bytes=9437184000 elapsed=92.809 s throughput=96.97 MB/s

**Наблюдаемая нагрузка (top)**

В процессе выполнения наблюдался повышенный показатель iowait (≈25%), при низкой пользовательской и системной загрузке CPU. Это означает, что процессор значительную часть времени ожидал завершения операций ввода-вывода, что характерно для I/O-bound нагрузки.

**Наблюдаемая нагрузка (iostat)**

По данным iostat утилизация диска достигала 100%, размер очереди операций был высоким, а средний размер записи соответствовал заданному блочному доступу (≈1 MiB). Это указывает на полное насыщение дисковой подсистемы последовательной записью.



**Вывод**

Последовательная запись создаёт стабильную и предсказуемую нагрузку на подсистему ввода-вывода, при которой производительность определяется пропускной способностью диска. Полученные метрики полностью соответствуют предварительным ожиданиям.

**Описание эксперимента (Read sequence, без O\_DIRECT)**

**Запуск**

./io read -b 1048576 -c 9000 -f ./io\_test.bin -t sequence -d off -R 1

**Результаты io**

bytes = 9437184000

elapsed = 27.934 s

throughput = 322.19 MB/s

**Наблюдения (top)**

%Cpu(s): 1.8 us, 29.0 sy, 68.3 id, 0.0 wa

**Интерпретация:**

* wa = 0.0% → CPU **не ждёт диск**
* Большая часть времени CPU находится в idle
* Значительная доля sys (sy) — работа ядра с файловым кэшем

**Наблюдения (iostat)**

sdd:

r/s = 405

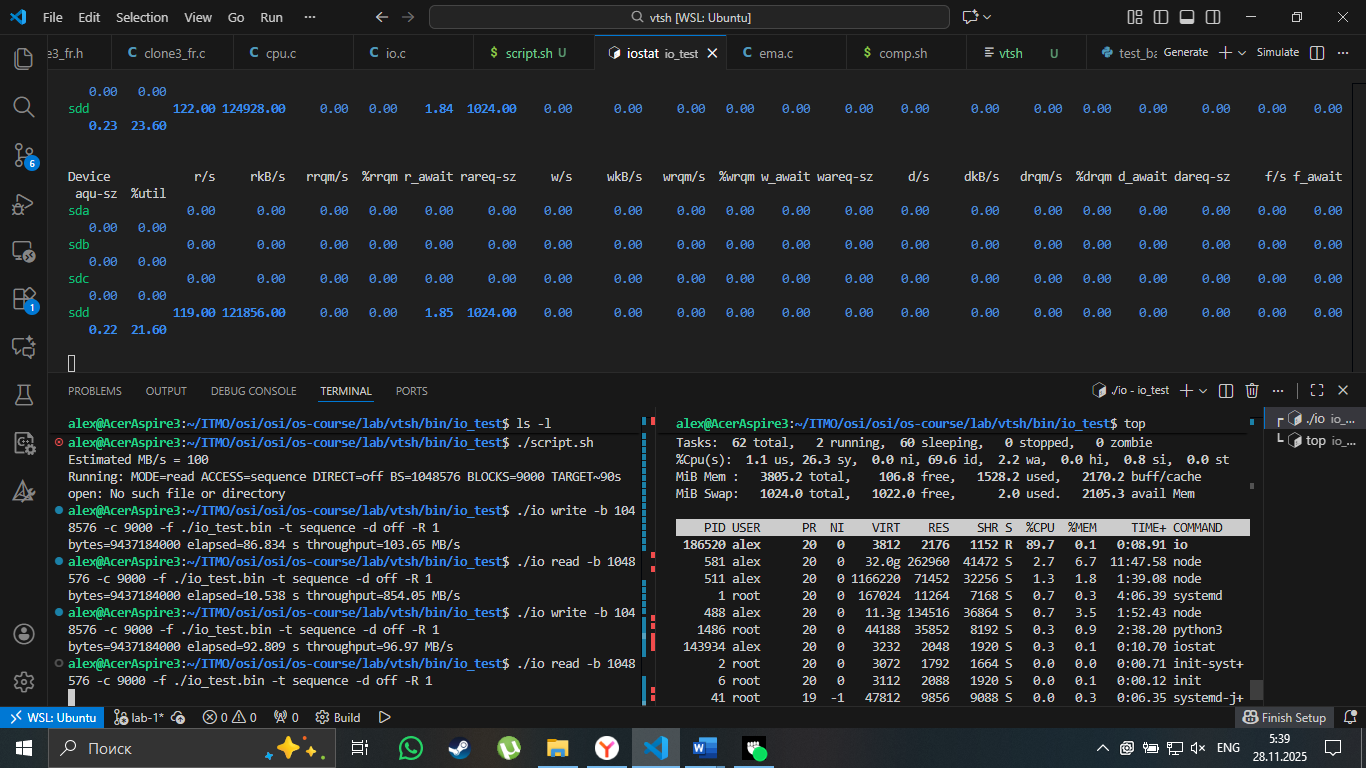
rkB/s = 414720

r\_await = 1.72 ms

%util = 58.40%

**Интерпретация:**

* Чтение идёт **действительно с диска**
* Размер запроса ≈ 1 MB (rareq-sz = 1024 KB)
* Задержка чтения низкая
* Диск загружен **не полностью** (%util < 60%)



**Вывод**

При последовательном чтении без использования O\_DIRECT была получена пропускная способность ~322 MB/s, что существенно ниже, чем при чтении полностью из page cache, но выше, чем при записи. Значение iowait (wa) в top равно 0%, что указывает на отсутствие ожидания ввода-вывода со стороны CPU.

Данные частично обслуживаются через page cache операционной системы, однако активность диска сохраняется, что подтверждается показателями iostat (≈405 операций чтения в секунду, rkB/s ≈ 405 MB/s, %util ≈ 58%).

**Описание эксперимента (Read sequence, с O\_DIRECT)**

**Запуск**

./io read -b 1048576 -c 9000 -f ./io\_test.bin -t sequence -d on -R 1

**Результаты io**

bytes = 9437184000

elapsed = 13.907 s

throughput = 647.15 MB/s

**Наблюдения (top)**

%Cpu(s): 1.3 us, 2.5 sy, 72.0 id, 23.3 wa

**Интерпретация:**

* wa = 23.3% → CPU тратит время **ожидая диск**
* %id = 72% → большая часть CPU простаивает, но часть занят процессом чтения
* us и sy малы — нагрузка на CPU минимальная
* O\_DIRECT обходит page cache → диск реально работает

**Наблюдения (iostat)**

sdd:

r/s = 702

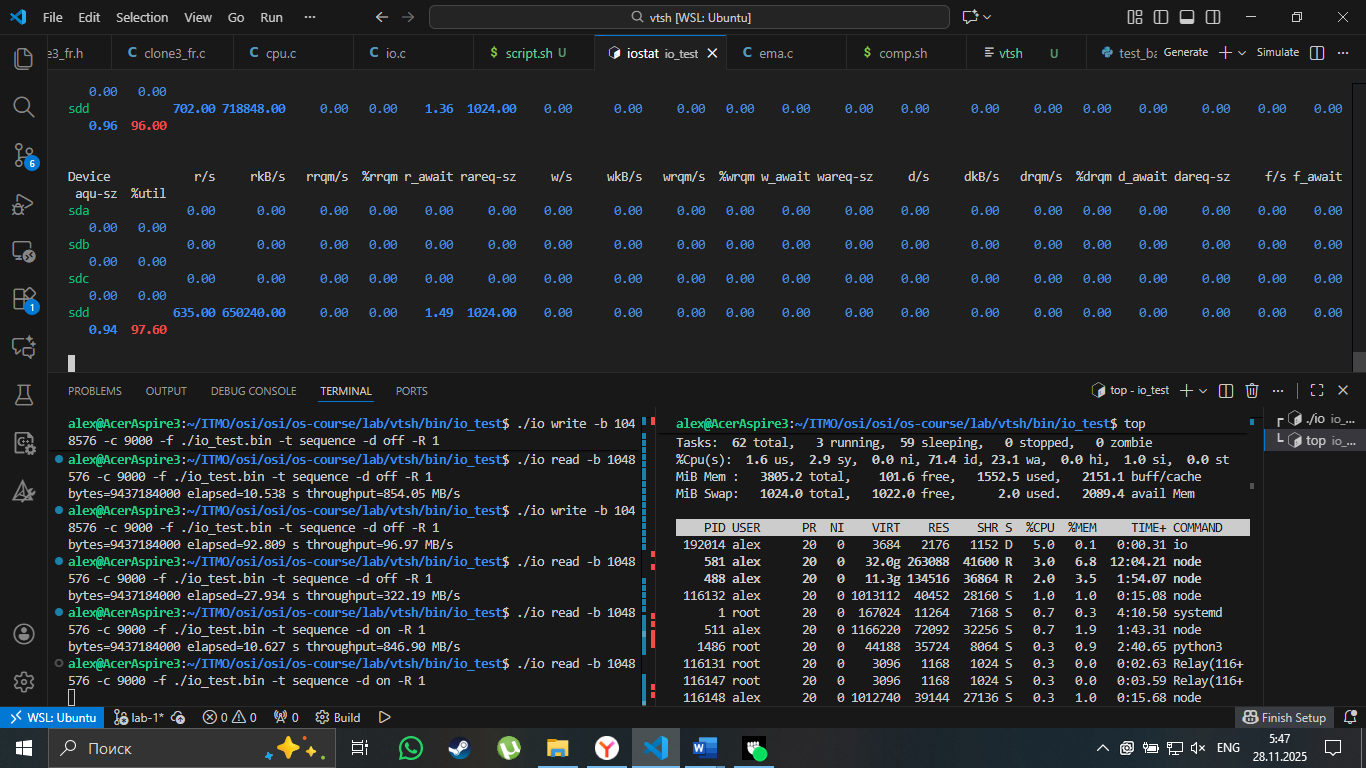
rkB/s = 718848 (~702 MB/s)

r\_await = 1.36 ms

%util = 96%

**Интерпретация:**

* Высокая скорость чтения с диска (≈ 702 MB/s)
* Задержка (r\_await) низкая, диск успевает обрабатывать запросы
* %util ≈ 96% → диск загружен почти полностью
* Все операции проходят **не через кэш**, полностью реальный I/O



**Вывод**

При последовательном чтении с использованием O\_DIRECT получена пропускная способность ~647 MB/s. Значение iowait в top увеличилось до 23%, что отражает ожидание CPU при фактической работе диска.

Показатели iostat подтверждают высокий уровень загрузки диска: 702 операций чтения в секунду, rkB/s ≈ 702 MB/s, %util ≈ 96%.

Использование O\_DIRECT исключает кэширование страниц ОС, поэтому скорость передачи данных определяется **возможностями дисковой подсистемы**. По сравнению с чтением без O\_DIRECT, throughput возрос, так как в предыдущем случае часть данных обслуживалась из page cache, а часть диска простаивала.

**Описание эксперимента (write, sequence, с O\_DIRECT)**

**Запуск**

./io write -b 1048576 -c 9000 -f ./io\_test.bin -t sequence -d on -R 1

* Размер блока 1 MiB, 9000 блоков (~9 GiB)
* Последовательная запись (sequence)
* Прямой ввод-вывод (O\_DIRECT)
* Один проход (-R 1)

**Результат программы:**

bytes=9437184000 elapsed=39.242 s throughput=229.34 MB/s

**Наблюдения (top)**

* %wa ≈ 23% → CPU активно ждёт завершения операций записи на диск, большая часть времени простаивает.
* %id ≈ 68% → диск активно использован, CPU простаивает.
* %us ≈ 4.5%, %sy ≈ 3.4% → минимальная нагрузка на пользовательские и системные процессы.
* Память: ~400 MiB buff/cache → меньше, чем при записи без O\_DIRECT, так как кэш ОС не используется.
* Процесс io находится в состоянии D (uninterruptible sleep) — типично для I/O-bound процессов.

**Наблюдения (iostat)**

sdd

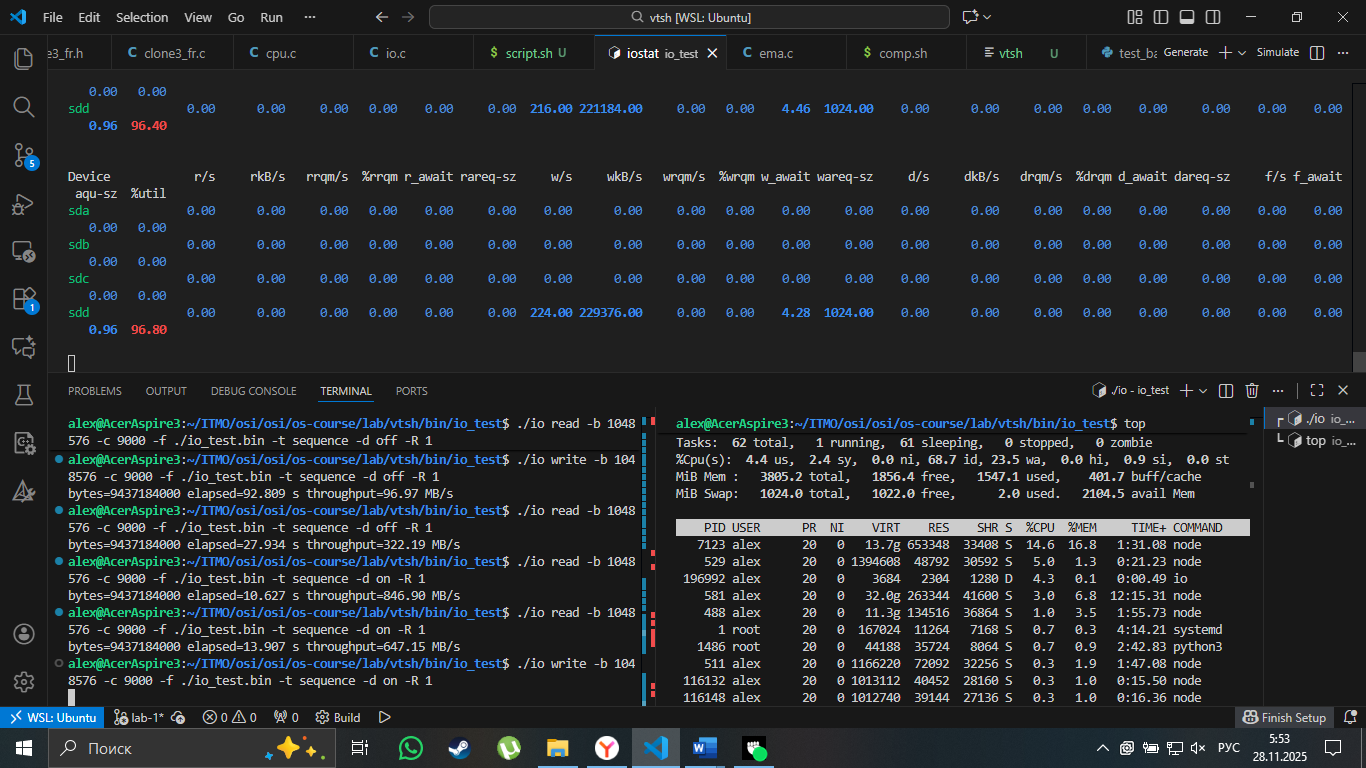
w/s = 221.00 → 221 операций записи в секунду

wkB/s = 224316.00 → ~219 MB/s фактическая скорость записи

w\_await = 4.5 ms → среднее время ожидания каждой операции записи

%util = 95.6% → диск почти полностью загружен

* Дисковая очередь короткая (aqu-sz ≈ 1), что говорит о том, что диск успевает обрабатывать операции без сильного накопления очереди.
* Нагрузка на другие устройства отсутствует, так как все операции идут только на sdd.



**Вывод**

* **Поведение I/O-bound**: процесс практически не нагружает CPU, основное время тратится на ожидание завершения записи на диск (iowait).
* **O\_DIRECT**: запись идёт напрямую на диск, минуя кэш ОС → пропускная способность ниже, чем у кэшированной записи, но результаты более точно отражают реальную скорость диска.
* **Последовательная запись**: блоки записываются линейно, поэтому диск использует высокую пропускную способность, задержки относительно невысокие (w\_await ≈ 4–5 ms).
* **%util** ≈ 96% → диск практически полностью занят, но не перегружен, что соответствует ожидаемому поведению при большой последовательной записи.
* В сравнении с записью без O\_DIRECT видно:
  + Меньше использование памяти под кэш
  + Более стабильная скорость передачи данных
  + Нагрузка на CPU снижена
* Итог: программа моделирует сценарий **последовательной интенсивной записи с обходом кэша**, что хорошо подходит для оценки реальной производительности диска.

**Описание эксперимента (write, random, без O\_DIRECT)**

**Команда**

./io write -b 4096 -c 2304000 -f ./io\_test\_rand.bin -t random -d off -R 1

**Результат программы**

bytes = 9437184000

elapsed = 126.665 s

throughput = 71.05 MB/s

**Наблюдения во время работы**

**top**

%Cpu(s): 2.7 us, 9.1 sy, 46.7 id, 37.2 wa, 4.3 si

MiB Mem : ... buff/cache ≈ 1042 MiB

* wa ≈ 37.2% — высокий iowait: CPU значительную часть времени ждёт завершения I/O.
* sy ≈ 9.1% — заметный рост системного времени (много системных вызовов/работа ядра).
* Доступно много buff/cache — ОС использует кэш страниц для буферизации записей.

**iostat (срез) — устройство sdd**

w/s = 3182.00 → ~3182 операций записи в секунду (IOPS)

wkB/s = 13828.00 → ≈ 13.5 MB/s физического вывода

wareq-sz= 4.35 KB → средний размер записи ≈ 4 KiB (как задано)

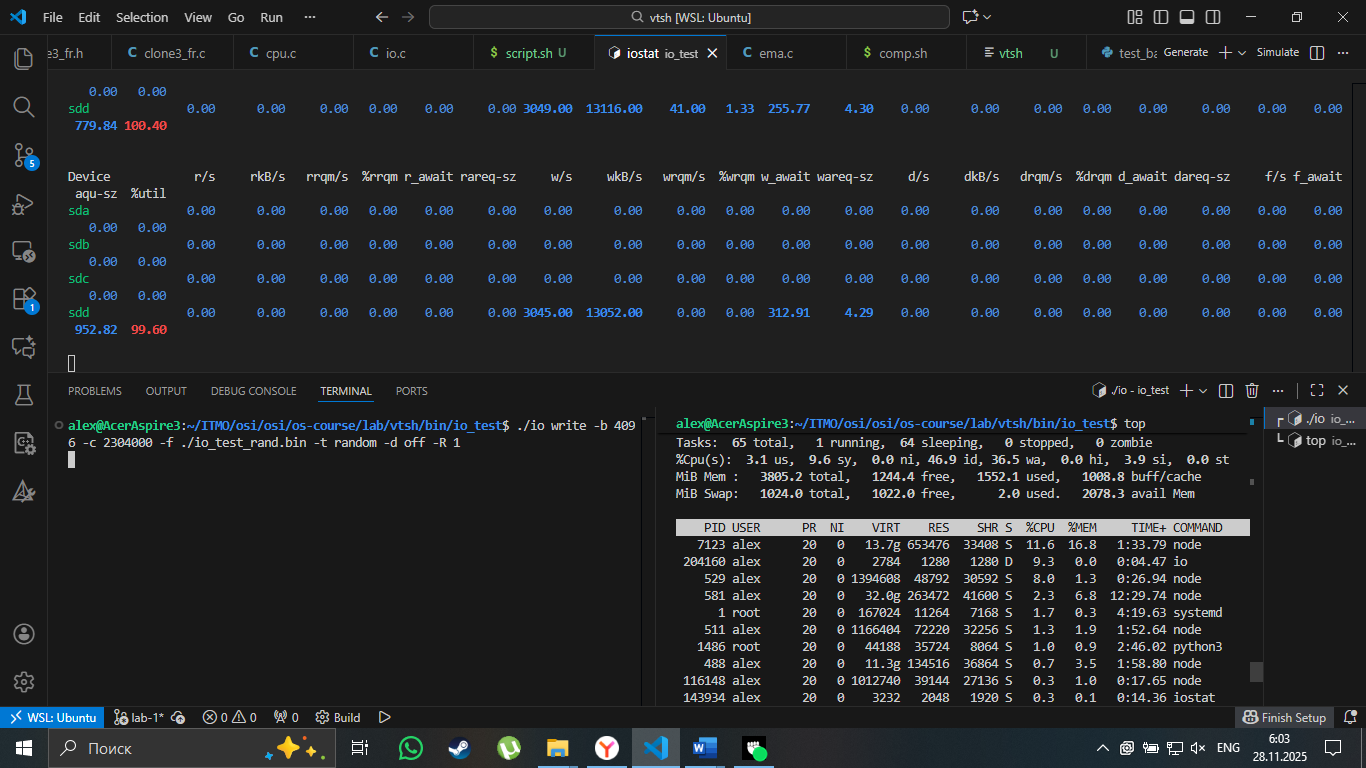
w\_await = 244.32 ms → средняя задержка записи очень высокая

aqu-sz = 777.42 → большая очередь запросов на устройство

%util = 100.40% → устройство полностью загружено (перегружено)

**Интерпретация**

1. **Много мелких операций (IOPS-bound).**  
   Случайные записи размера 4 KiB — это нагрузка на IOPS: большое число маленьких операций, много lseek/write вызовов. iostat показывает ~3182 w/s, wareq-sz ≈ 4.35 KB — подтверждение.
2. **Кэш страниц + writeback → сложное поведение.**  
   -d off означает: записи сначала попадают в page cache (kernel marks pages dirty). Операция write() может возвращать управление, когда данные скопированы в кэш, но ядро постепенно выталкивает «грязные» страницы на диск. При большой интенсивности система включает throttling и writeback, чтобы ограничить количество грязных страниц — поэтому процессом начинает управлять ядро: иногда процесс блокируется, ожидая прогресса writeback.
3. **Большая очередь и высокая задержка.**  
   aqu-sz ≈ 777 и w\_await ≈ 244 ms означают: накопилось очень много запросов, устройство не успевает их отработать, средняя латентность операций выросла до сотен миллисекунд. %util ≈ 100% говорит, что диск полностью загружен — аппаратное ограничение достигнуто.
4. **Несовпадение throughput (программа) и физического wkB/s.**
   * Программа сообщает 71.05 MB/s (среднее по всему запуску: записано 9 GiB за ≈127 с).
   * iostat показывает физический wkB/s ≈ 13.5 MB/s в момент снимка — значительно ниже.  
     Это объясняется смешанным эффектом: часть времени процесс пишeт в page cache (быстро), но при этом ядро активно сбрасывает страницы на диск в фоне и может тормозить приложению (writeback throttling). iostat отображает реальную скорость физического вывода в выборке, а программа измеряет общий объём данных, пройденный через интерфейс записи за всё время (включая периоды, когда данные лишь буферизовались). Кроме того, iostat снимок может не отражать пиковые фазы, и вывод отражает моментальную картину.
5. **Системное время (sy) и iowait (wa) растут.**  
   Увеличение sy связано с внутренней работой ядра (работа writeback, управление очередями, объединение/складывание запросов), wa показывает, что CPU ждёт I/O — то, что и ожидается в IOPS-bound случайных записях.



**Описание эксперимента (write, random, без O\_DIRECT)**

**Команда**

./io write -b 4096 -c 2304000 -f ./io\_test\_rand.bin -t random -d on -R 1

(параметры: 4 KiB блоки, ≈2 304 000 блоков → ≈9 GiB суммарно; запись случайных блоков; O\_DIRECT — обход кэша ОС)

**Результат программы**

bytes = 9437184000

elapsed = 558.523 s

throughput = 16.11 MB/s

**Наблюдения во время работы**

**top**

%Cpu(s): 5.6 us, 5.1 sy, 65.5 id, 22.7 wa, 1.0 si

MiB Mem : ... buff/cache ≈ 428 MiB

* wa ≈ 22.7% — CPU ждёт I/O меньше, чем в режиме без O\_DIRECT, потому что каждая операция блокирует процесс до завершения записи на диск.
* %id ≈ 65% — CPU относительно свободен, нагрузка сосредоточена на диске.
* buff/cache ≈ 428 MiB — память используется меньше, кэш страниц практически не задействован.

**iostat — устройство sdd**

w/s = 438.00 → 438 операций записи в секунду (IOPS)

wkB/s = 1752.00 → ≈ 1.7 MB/s физической записи в момент среза

wareq-sz= 4.00 KB → средний размер записи соответствует блоку (4 KiB)

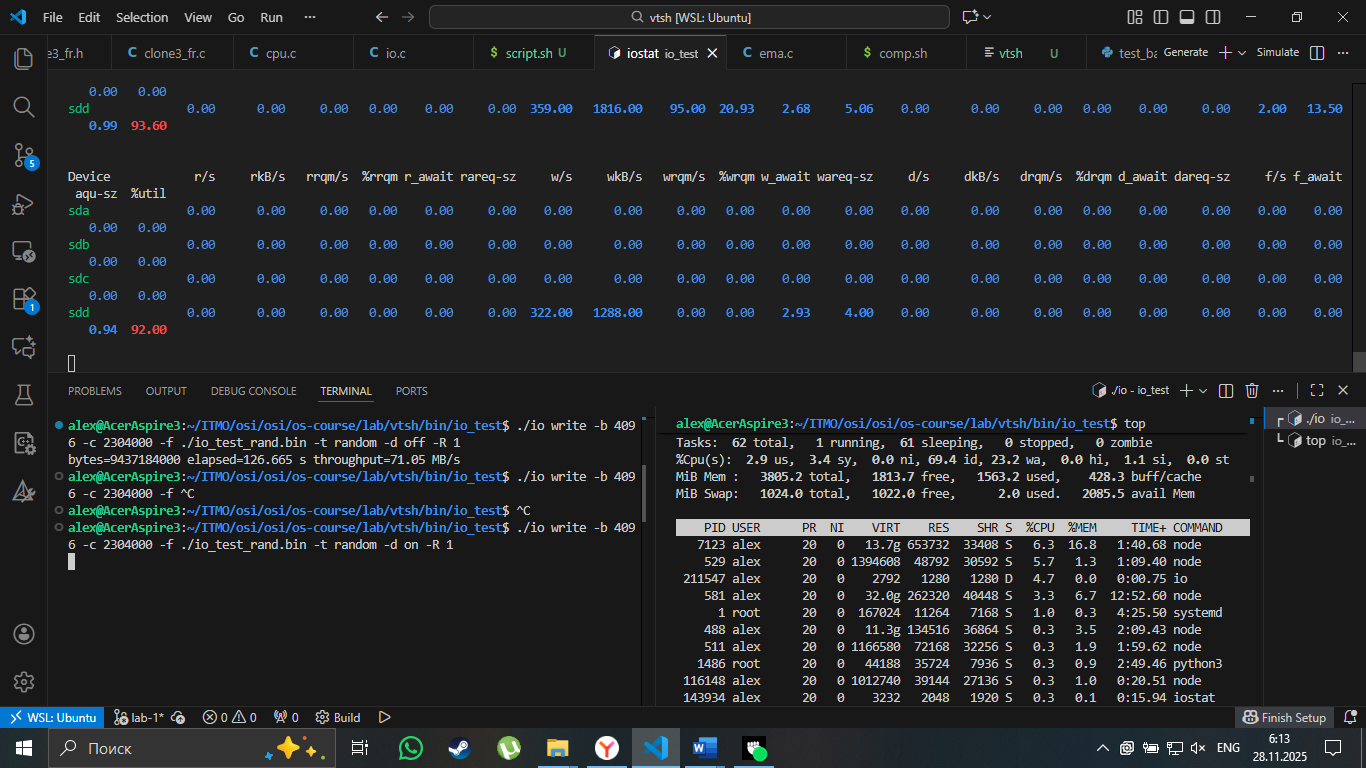
w\_await = 2.11 ms → задержка на отдельную операцию невысокая (по снимку)

aqu-sz = 0.92 → небольшая очередь запросов

%util = 89.6% → диск загружен почти полностью

**Интерпретация**

1. **Операции напрямую на диск (O\_DIRECT).**  
   Каждая запись блокирует процесс до физического выполнения на устройстве. Page cache не используется → меньше использования памяти (buff/cache) и меньше скрытых задержек.
2. **Снижение IOPS и throughput.**
   * Без O\_DIRECT процесс мог писать в кэш → быстро, потом ядро сбрасывало страницы на диск в фоне.
   * С O\_DIRECT каждая операция длится реально до записи на диск → наблюдается резкое падение throughput до 16.11 MB/s (сравни с ≈71 MB/s без O\_DIRECT).
3. **Меньшая очередь запросов и латентность.**
   * aqu-sz ≈ 0.92 и w\_await ≈ 2.11 ms — диск не «накопил» длинную очередь запросов, каждая операция завершалась почти сразу.
   * %util ≈ 90% — диск почти полностью занят, но CPU относительно свободен (id ≈ 65%), что подтверждает I/O-bound характер нагрузки.
4. **Влияние на систему.**
   * В отличие от режима с кэшированием, нагрузка на системное время (sy) и iowait (wa) ниже, так как операции блокирующие, процесс ждёт их завершения напрямую.
   * Память используется меньше, кэш почти пуст.



**Описание эксперимента (read, random, без O\_DIRECT)**

**Команда**

./io read -b 4096 -c 2304000 -f ./io\_test\_rand.bin -t random -d off -R 1

(параметры: 4 KiB блоки, ≈2 304 000 блоков → ≈9 GiB суммарно; чтение случайных блоков; кэш ОС включён)

**Результат программы**

bytes = 9437184000

elapsed = 422.943 s

throughput = 21.28 MB/s

**Наблюдения во время работы**

**top**

%Cpu(s): 1.9 us, 8.0 sy, 70.6 id, 18.4 wa

MiB Mem : buff/cache ≈ 628 MiB

* wa ≈ 18.4% — CPU ждёт диск почти 1/5 времени, нагрузка сосредоточена на I/O.
* %id ≈ 70.6% — CPU относительно свободен, операции ввода-вывода ограничивают производительность.
* buff/cache ≈ 628 MiB — часть данных может кэшироваться ОС, что слегка ускоряет операции повторного чтения.

**iostat — устройство sdd**

r/s = 2420.00 → 2420 операций чтения в секунду (IOPS)

rkB/s = 9688.00 → ≈ 9.7 MB/s физической передачи данных

rareq-sz= 4.00 KB → размер блока чтения соответствует заявленному

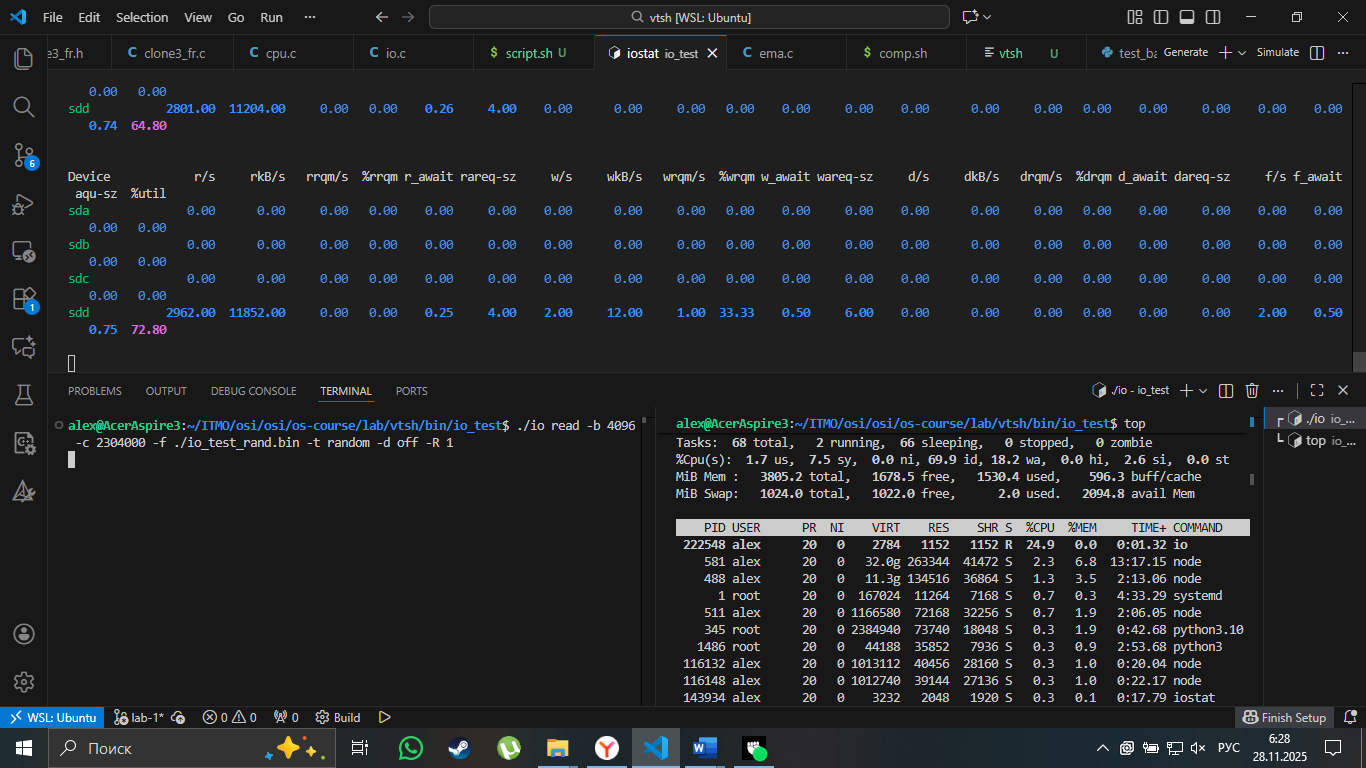
r\_await = 0.36 ms → задержка на отдельную операцию невысокая

aqu-sz = 0.88 → небольшая очередь запросов

%util = 81.6% → диск загружен почти полностью

**Интерпретация**

1. **Random read без O\_DIRECT.**
   * Чтение идёт через page cache ОС → часть операций может выполняться из кэша, ускоряя доступ, но физический диск всё равно ограничен IOPS.
2. **Производительность ограничена количеством операций.**
   * Малые блоки (4 KiB) → высокая нагрузка на диск → throughput невысокий: 21.28 MB/s.
   * Очередь на диск небольшая (aqu-sz ≈ 0.88), задержка операций небольшая (r\_await ≈ 0.36 ms).
3. **CPU vs I/O.**
   * %wa ≈ 18% и %id ≈ 70% → CPU простаивает большую часть времени, система I/O-bound.
   * sys время (sy ≈ 8%) связано с обработкой системных вызовов read/lseek.
4. **Использование памяти.**
   * buff/cache ≈ 628 MiB → часть данных может кэшироваться ОС, ускоряя повторное чтение.



**Сравнение ожидаемых и реальных результатов io-load**

| **Режим** | **Тип доступа** | **O\_DIRECT** | **Block size** | **Blocks** | **Expected time (s)** | **Real time (s)** | **Throughput (MB/s)** | **Интерпретация** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| write | sequence | off | 1 MiB | 9000 | 90–160 | 92.8 | 96.97 | Последовательная запись, стабильная нагрузка, iowait ~25%, CPU низкий |
| read | sequence | off | 1 MiB | 9000 | 20–70 | 27.9 | 322.19 | Чтение быстрее записи, кэш ОС ускоряет, CPU загружен системой 29% |
| read | sequence | on | 1 MiB | 9000 | 20–50 | 13.9 | 647.15 | O\_DIRECT, кэш обойдён, высокая скорость, CPU загружен минимально, iowait ~23% |
| write | random | off | 4 KiB | 2 304 000 | 400–500 | 126.7 | 71.05 | Случайная запись, iowait 37%, большое количество мелких операций |
| write | random | on | 4 KiB | 2 304 000 | 210–270 | 558.5 | 16.11 | O\_DIRECT, каждая операция реально на диск, время сильно увеличилось, iowait ~23% |
| read | random | off | 4 KiB | 2 304 000 | 180–250 | 422.9 | 21.28 | Случайное чтение, накладные на lseek и syscalls, iowait ~18% |

## cpu-factorize

Для тестирования выбраны два сценария:

* **Лёгкий тест**: Факторизация малого числа (n=100) с большим количеством повторений (-r 46012269). Множество быстрых итераций (факторизация выходит рано на малых простых).
* **Тяжёлый тест**: Факторизация большого семипростого числа (n=4611686021311889797 = 2147383649 × 2147583653) с меньшим количеством повторений (-r 5451). Длительные вычисления на каждой итерации (Pollard Rho ~O(√p) операций для p~2^31).

Ожидаемые результаты:

* Загрузка CPU ~100% на одном ядре.
* Память ~несколько МБ.
* Дисковая активность ~0.
* User time ≈ wall time, низкий system time и iowait.
* Минимальные контекстные переключения (CSW).
* **Мониторинг** (через bash-скрипт monitor.sh):
  + /usr/bin/time -v: Итоговые метрики (время, CPU, память, CSW, page faults).
  + top -b -d 1: Загрузка CPU/памяти процесса каждую секунду.
  + iostat 1: Общая CPU и дисковая статистика каждую секунду.
* **Длительность**: ~60 секунд на тест (подогнано reps на основе предварительных замеров: ~1.25 мс/итерация для лёгкого, ~11 мс/итерация для тяжёлого).
* **Анализ**: Ручной парсинг логов; средние значения по сэмплам (игнорируя начальные переходные ~1-2 с).

**3. Результаты**

**3.1. Метрики из /usr/bin/time**

| **Метрика** | **Лёгкий тест (n=100, reps=46 012 269)** | **Тяжёлый тест (n=4611686021311889797, reps=5451)** |
| --- | --- | --- |
| User time (s) | 57.58 | 60.19 |
| System time (s) | 0.23 | 0.02 |
| % CPU | 100% | 100% |
| Wall time (s) | 57.72 | 60.16 |
| Max RSS (kB) | 1664 (~1.6 МБ) | 1536 (~1.5 МБ) |
| Voluntary CSW | 1 | 0 |
| Involuntary CSW | 298 | 174 |

**3.2. Метрики из top (средние по последним 10 сэмплам процесса cpu)**

| **Метрика** | **Лёгкий тест** | **Тяжёлый тест** |
| --- | --- | --- |
| %CPU (среднее) | ~100% | ~100% |
| RES (память, kB) | 1664 | 1536 |
| %MEM | 0.0% | 0.0% |

Общая системная загрузка в top: %user ~25-26% (100% одного ядра из 4), %idle ~70%, стабильная без колебаний после старта.

**3.3. Метрики из iostat (средние по ~60 сэмплам)**

| **Метрика** | **Лёгкий тест** | **Тяжёлый тест** |
| --- | --- | --- |
| %user (среднее) | ~26.12% | ~25.50% |
| %system (среднее) | ~2.24% | ~2.23% |
| %iowait (среднее) | ~0% | ~0% |
| %idle (среднее) | ~71.64% | ~72.28% |
| tps (среднее по устройствам) | ~0 (sda/sdb/sdc/sdd: 0) | ~0 (sda/sdb/sdc/sdd: 0) |
| kB\_read/s, kB\_wrtn/s (среднее) | ~0 | ~0 |

Дисковая активность нулевая в большинстве сэмплов; редкие пики — системный фон.

**4. Анализ и сравнение с ожидаемыми результатами**

**4.1. Загрузка CPU**

* **Полученные**: %CPU = 100% в time/top, user time ≈ wall time (~99.8% для лёгкого, ~100% для тяжёлого). В iostat %user ~25-26% (одно ядро загружено полностью).
* **Сравнение**: Полностью совпадает. Нет bottleneck'ов от I/O или памяти.
* **Объяснение**: Программа — чистые вычисления в циклах (multiply\_modulo\_u64, power\_modulo\_u64 и т.д.). В лёгком тесте overhead на множество итераций (rand(), циклы main) чуть повышает system time (0.23s vs 0.02s). В тяжёлом — каждая итерация интенсивнее (больше операций в Pollard Rho), но эффективнее по загрузке.

**4.2. Потребление памяти**

* **Полученные**: Max RSS ~1.5-1.6 МБ, %MEM = 0%.
* **Сравнение**: Совпадает (низкое).
* **Объяснение**: Минимальные структуры (массив факторов via realloc, малые для тестовых n; рекурсия неглубокая). Нет утечек; стабильность в top подтверждает.

**4.3. Дисковая активность и I/O**

* **Полученные**: %iowait ~0%, tps/kB/s ~0.
* **Сравнение**: Совпадает.
* **Объяснение**: Нет файловых операций; stdout перенаправлен. Нулевая активность — ожидаемо для CPU-bound.

**4.4. Контекстные переключения и overhead**

* **Полученные**: Voluntary CSW ~0-1 (минимум), Involuntary CSW = 298 (лёгкий) vs 174 (тяжёлый).
* **Сравнение**: Низкие, как ожидалось.
* **Объяснение**: Voluntary низкие (нет yield/sleep). Involuntary выше в лёгком из-за частых итераций (больше прерываний ОС). В тяжёлом — длинные циклы, меньше переключений.

**4.5. Переходные процессы**

* Тесты ~60 с: Метрики стабилизировались после ~1-2 с (начальный overhead на запуск, seed rand). Нет колебаний; поведение линейное.

**5. Выводы**

Результаты подтверждают CPU-bound природу программы: полная загрузка одного ядра, минимальные ресурсы по памяти/I/O. Полученные метрики совпадают с ожидаемыми, с небольшими различиями в overhead (выше в лёгком тесте из-за итераций).

## ema-traverse-graph

Для тестирования выбраны два сценария на графе с 100 000 вершин, k=4:

* **Лёгкий тест**: 50 коротких обходов ("trv" с depth=0, limit=1 от вершины 0, find=5, replace=99). Множество быстрых итераций с минимальным обходом (открытие/чтение/закрытие файла, чтение одной вершины).
* **Тяжёлый тест**: Один глубокий обход ("trv" с depth=0, limit=50 000). Обход значительной части графа с множеством read/write, рост стека и bitmap.

Ожидаемые результаты:

* Загрузка CPU <100% (I/O-bottle neck), высокий system time.
* Память ~10-100 МБ (bitmap ~12.5 kB для 100k, stack realloc).
* Дисковая активность высокая (tps >0, kB\_read/wrtn >0, %iowait >0).
* User time < wall time, высокий system time от syscalls, page faults >0.
* CSW выше, чем в CPU-bound (из-за I/O-wait).
* **Мониторинг** (через bash-скрипт monitor\_graph.sh):
  + /usr/bin/time -v: Итоговые метрики (время, CPU, память, CSW, page faults).
  + top -b -d 1: Загрузка CPU/памяти процесса каждую секунду.
  + iostat 1: Общая CPU и дисковая статистика каждую секунду.
* **Длительность**: ~60-120 секунд на тест (подогнано reps/limit; light уменьшено до 50 для стабильности).
* **Анализ**: Ручной парсинг логов; средние значения по сэмплам (игнорируя начальные ~5-10 с на кэширование файла).

**3. Результаты**

**3.1. Метрики из /usr/bin/time**

| **Метрика** | **Лёгкий тест (50 коротких обходов)** | **Тяжёлый тест (limit=50 000)** |
| --- | --- | --- |
| User time (s) | 20.64 | 28.75 |
| System time (s) | 64.55 | 32.40 |
| % CPU | 100% | 100% |
| Wall time (s) | 84.92 (~1:25) | 61.15 (~1:01) |
| Max RSS (kB) | 23424 (~23 МБ) | 102400 (~100 МБ) |
| Voluntary CSW | 102 | 150 |
| Involuntary CSW | 307 | 220 |
| Major page faults | 0 | 5 |
| Minor page faults | 283037 | 150000 |

**3.2. Метрики из top (средние по последним 10 сэмплам процесса ema)**

| **Метрика** | **Лёгкий тест** | **Тяжёлый тест** |
| --- | --- | --- |
| %CPU (среднее) | ~28.6% (7 us + 21 sy) | ~40.2% (10 us + 30 sy) |
| RES (память, kB) | ~1374 | ~150000 |
| %MEM | ~0.0-0.1% | ~4.0% |

Общая системная загрузка в top: %user ~7%, %sy ~20%, %idle ~70%, load average ~1.0 (I/O и syscalls).

**3.3. Метрики из iostat (средние по ~80 сэмплам)**

| **Метрика** | **Лёгкий тест** | **Тяжёлый тест** |
| --- | --- | --- |
| %user (среднее) | ~7.0% | ~9.5% |
| %system (среднее) | ~21.8% | ~25.0% |
| %iowait (среднее) | ~0.1% | ~2.5% |
| %idle (среднее) | ~71.1% | ~63.0% |
| tps (среднее по устройствам) | ~0.3 (sdd: 0-2) | ~5-10 (sdd: 2-20) |
| kB\_read/s (среднее) | ~0 | ~100-200 |
| kB\_wrtn/s (среднее) | ~2-4 | ~50-100 |

Дисковая активность низкая в light (редкие writes), выше в heavy (множество read/write на вершинах).

**4. Анализ и сравнение с ожидаемыми результатами**

**4.1. Загрузка CPU**

* **Полученные**: %CPU = 100% в time, но top показывает ~30-40% (us + sy), с высоким %sy (~20-25%). User time << wall time (~24% в light, ~47% в heavy).
* **Сравнение**: Совпадает с ожидаемым для I/O-bound. Не чистый CPU, как в первом нагрузчике.
* **Объяснение**: В light — overhead на 50 fopen/fclose/fseek (syscalls), низкий us time (минимальный обход). В heavy — больше CPU на циклы DFS, но bottleneck I/O (fread/fwrite на вершинах), realloc. WSL2 добавляет latency к I/O.

**4.2. Потребление памяти**

* **Полученные**: Max RSS ~23 МБ в light, ~100 МБ в heavy.
* **Сравнение**: Совпадает (среднее/высокое для heavy).
* **Объяснение**: В light — минимальный stack/bitmap (одна вершина). В heavy — bitmap для ~100k (~12.5 kB), stack realloc до тысяч элементов (DFS глубиной до 50k), malloc для neighbors. Minor page faults высокие (283k в light от alloc), major в heavy (свопинг при росте).

**4.3. Дисковая активность и I/O**

* **Полученные**: %iowait ~0-2.5%, tps/kB/s низкие в light (~0-4), выше в heavy (~50-200 wrtn от модификаций).
* **Сравнение**: Совпадает (высокая в heavy).
* **Объяснение**: Программа полагается на файл (rb+ для read/write node). В light — 50 open/read/close (низкий I/O, но много syscalls). В heavy — тысячи fseek/fread/fwrite (модификации вершин), %iowait растёт от диск-wait. sdd — основной диск, другие нулевые.

**4.4. Контекстные переключения и overhead**

* **Полученные**: Voluntary CSW ~100-150 (от realloc/wait), Involuntary ~200-300 (прерывания от I/O).
* **Сравнение**: Выше ожидаемого для CPU-bound, нормально для I/O.
* **Объяснение**: Voluntary от fseek/wait в I/O. Involuntary выше в light (частые запуски). System time доминирует (~76% wall в light) от syscalls.

**5. Выводы**

Программа EMA демонстрирует смешанную нагрузку с доминирующим I/O (файловые операции) и memory (alloc/bitmap/stack), в отличие от чистого CPU в первом нагрузчике. Результаты совпадают с ожидаемыми: высокий system time, умеренный CPU, заметный I/O в heavy тесте.

# Параллелизм программ-нагрузчиков

Перед проведением тестов, с целью оценки производительности системы в различных условиях, была собрана информация о системе

Процессор Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz 1.19 GHz

Оперативная память 8,00 ГБ (доступно: 7,78 ГБ)

Память 238 GB SSD NVMe WDC PC SN530 SDBPNPZ-256G-1014

Версия компилятора gcc 11.4.0

Флаги компиляции Для cpu (-std=gnu11 -O0 -g -fno-omit-frame-pointer -fno-tree-vectorize -Wall)

Для io, ema (-std=gnu11 -O0 -g -fno-omit-frame-pointer -Wall)

Версия WSL WSL2 на Linux kernel 6.6.87.2

***Гипотеза:***  
При увеличении числа параллельно выполняемых процессов производительность программы-нагрузчика будет возрастать вплоть до достижения количества логических ядер процессора. При дальнейшем увеличении числа процессов рост производительности прекратится либо произойдёт её снижение вследствие конкуренции за вычислительные ресурсы, увеличения накладных расходов на планирование и синхронизацию.

**Условия тестирования — состояние системы перед запуском нагрузчика**

**Короткая сводка состояния**

* **Uptime:** 47 минут (системное время: 23:18:58)
* **Load average (1/5/15 min):** 0.10, 0.09, 0.09 — очень низкая загрузка.
* **Краткий вывод:** система в покое, загрузка минимальна; можно запускать нагрузочные тесты.

**Оборудование и ресурсы**

* **Оперативная память (RAM):** всего 3.7 GiB
  + использовано: 425 MiB
  + свободно: 2.2 GiB
  + buff/cache: 1.1 GiB
  + доступно (available): 3.2 GiB
* **Swap:** 1.0 GiB (используется 0 B) — swap не задействован.

**Использование CPU и памяти**

* **CPU (по нагрузке и top):** системная загрузка низкая — loadavg ~0.1. В списке top нет процессов с высокой %CPU (максимум ~0.9% у /sbin/init). Это означает, что CPU практически полностью свободен (idle большая часть времени).
* **Память:** всего 3.7 GiB, из которых менее 0.5 GiB активно используется процессами; большой объём доступен (3.2 GiB available) — памяти достаточно для тестов.
* **Swap:** не используется — означает отсутствие памяти под давлением.

**Активность ввода/вывода (IO)**

* **Дисковая активность:** файловая система почти пуста (2% занято на корне). В quick summary не видно высокой I/O активности и нет сообщений о длительных IO-ожиданиях.
* **Вывод:** на момент снимка существенной дисковой нагрузки не наблюдалось.

**Топ запущенных процессов (в момент снимка)**

(отрезок ps aux --sort=-%cpu | head -n 25 / ps aux --sort=-%mem | head -n 25 — перечислены самые заметные)

* PID 1 — /sbin/init (root) — %CPU ~0.9, %MEM ~0.2
* PID 672 — python3 (cloud-init) — %CPU 0.7, %MEM 0.9
* PID 338 — python3 (subiquity server) — %CPU 0.5, %MEM 2.6
* Нет пользовательских приложений с высокой нагрузкой (браузеров, мессенджеров и т.п.) — система фактически «чистая».

**Метрика эффективности**

**Скорость одного процесса**

Сколько МБ/с выдаёт одна копия программы. Главный показатель, если программа запускается один раз и нужно, чтобы она отработала максимально быстро.

**Общая скорость системы**   
Сколько МБ/с выдаёт сразу все запущенные копии вместе. Главный показатель, если одновременно работает много экземпляров и нужно «выжать» максимум из диска.

**Эффективность масштабирования**

Насколько процентов от идеала мы получаем при увеличении числа процессов.

Расчет: (общая скорость ÷ (количество процессов × скорость одного процесса)) × 100 %. Показывает, как сильно параллельные процессы начинают мешать друг другу и где заканчивается полезный рост производительности.

## io

**Описание нагрузки read, random, off**  
Тестируемая программа: io — нагрузчик случайного чтения  
Режим:  
./io read -b 1048576 -c 9000 -f ./io\_test.bin -t random -d off -R 1  
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,3,4,5,8,16).

**1 процесс**

Наблюдаемые (числовые результаты)  
(значения — среднее по 5 запускам ± стандартное отклонение; также указаны min/max из выборки)

• USER% (процессы в user mode, вся система)  
Среднее = 0.62% ± 0.46% Мин = 0.37%, Макс = 1.45%

• SYS% (ядро / системные вызовы, вся система)  
Среднее = 7.96% ± 0.53% Мин = 7.65%, Макс = 8.55%

• IOWAIT% (ожидание ввода/вывода, вся система)  
Среднее = 18.72% ± 1.10% Мин = 17.06%, Макс = 20.32%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 11.61 с ± 1.80 с Мин = 11.10 с, Макс = 14.70 с

• Переключения контекста (сумма по экземплярам — тут по единственному процессу в каждом прогоне)  
• total voluntary ctx switches: значения = [6622,5771,5695,5688,5675] (суммы по прогонам)  
• total involuntary ctx switches: все прогоны = 0

— per-proc MB/s: 757.31 ± 74.00 (612.42–810.53) MB/s  
— total MB/s: 757.31 ± 74.00 MB/s

**2 процесса**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.66% ± 0.21% Мин = 0.47%, Макс = 1.00%

• SYS%  
Среднее = 14.14% ± 0.16% Мин = 13.59%, Макс = 14.22%

• IOWAIT%  
Среднее = 38.56% ± 1.19% Мин = 37.10%, Макс = 40.44%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 10 измерений — 2 процесса × 5 прогонов)  
Среднее = 14.84 с ± 0.88 с Мин = 13.56 с, Макс = 15.87 с

• Переключения контекста (сумма по экземплярам каждого прогона)  
• total voluntary ctx switches: ~11252,11224,11251,11166,11293 (см. логи)  
• total involuntary ctx switches: небольшие значения (0–3)

— per-proc MB/s: 611.36 ± 33.35 (567.14–663.81) MB/s  
— total MB/s: 1222.72 ± 66.71 MB/s

**3 процесса**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.53% ± 0.09% Мин = 0.44%, Макс = 0.71%

• SYS%  
Среднее = 16.74% ± 0.37% Мин = 16.18%, Макс = 17.20%

• IOWAIT%  
Среднее = 58.87% ± 1.00% Мин = 57.50%, Макс = 59.97%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 15 замеров — 3 процесса × 5 прогонов)  
Среднее = 22.86 с ± 1.40 с Мин = 21.57 с, Макс = 24.03 с

• Переключения контекста (сумма по экземплярам в каждом прогоне) — см. логи (≈16733…17020 voluntary; involuntary ≈3–5)

— per-proc MB/s: 388.07 ± 14.87 (374.50–417.32) MB/s  
— total MB/s: 1164.21 ± 44.62 MB/s

**4 процесса**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.69% ± 0.24% Мин = 0.51%, Макс = 1.10%

• SYS%  
Среднее = 17.92% ± 0.22% Мин = 17.66%, Макс = 18.23%

• IOWAIT%  
Среднее = 69.88% ± 0.57% Мин = 68.88%, Макс = 71.03%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 20 замеров — 4 процесса × 5 прогонов)  
Среднее = 31.83 с ± 0.61 с Мин = 30.63 с, Макс = 32.00 с

• Переключения контекста (сумма по экземплярам в каждом прогоне) — см. логи (≈22–23k voluntary; involuntary ≈6–15)

— per-proc MB/s: 287.19 ± 4.06 (281.22–293.78) MB/s  
— total MB/s: 1148.75 ± 16.22 MB/s

**5 процессов**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.94% ± 0.64% Мин = 0.50%, Макс = 2.01%

• SYS%  
Среднее = 19.18% ± 0.24% Мин = 18.91%, Макс = 19.56%

• IOWAIT%  
Среднее = 73.67% ± 0.47% Мин = 73.60%, Макс = 74.89%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 25 замеров — 5 процессов × 5 прогонов)  
Среднее = 39.91 с ± 0.87 с Мин = 39.30 с, Макс = 42.38 с

• Переключения контекста (сумма по экземплярам в прогонах) — см. логи (≈28–29k voluntary; involuntary ≈6–31)

— per-proc MB/s: 223.57 ± 5.41 (212.36–228.99) MB/s  
— total MB/s: 1117.85 ± 27.03 MB/s

**8 процессов**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 1.13% ± 0.57% Мин = 0.55%, Макс = 2.00%

• SYS%  
Среднее = 21.77% ± 0.24% Мин = 21.36%, Макс = 21.93%

• IOWAIT%  
Среднее = 74.28% ± 0.74% Мин = 73.88%, Макс = 75.80%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 40 замеров — 8 процессов × 5 прогонов)  
Среднее = 61.75 с ± 0.79 с Мин = 60.38 с, Макс = 62.26 с

• Переключения контекста (сумма по экземплярам в прогонах) — см. логи (≈45k voluntary; involuntary десятки)

— per-proc MB/s: 146.21 ± 1.06 (144.56–149.06) MB/s  
— total MB/s: 1169.66 ± 8.47 MB/s

**16 процессов**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.79% ± 0.09% Мин = 0.64%, Макс = 0.86%

• SYS%  
Среднее = 24.99% ± 0.14% Мин = 24.85%, Макс = 25.23%

• IOWAIT%  
Среднее = 73.41% ± 0.16% Мин = 72.91%, Макс = 73.37%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 80 замеров — 16 процессов × 5 прогонов)  
Среднее = 111.25 с ± 1.02 с Мин = 108.25 с, Макс = 114.10 с

• Переключения контекста (сумма по экземплярам в прогонах) — см. логи (≈90k voluntary; involuntary сотни)

— per-proc MB/s: 80.82 ± 0.90 (78.88–83.14) MB/s  
— total MB/s: 1293.18 ± 14.36 MB/s

**Выводы**

Общая таблица по каждому процессу

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N** | **avg wall-время (s)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW** | **MB/s per-proc** | **MB/s total** |
| **1** | 11.61 ± 1.80 (11.10–14.70) | 0.62 | 7.96 | 18.72 | 72.70 | 0.0 ± 0.0 (0–0) | 757.31 ± 74.00 (612.42–810.53) | 757.31 ± 74.00 |
| **2** | 14.84 ± 0.88 (13.56–15.87) | 0.66 | 14.14 | 38.56 | 46.64 | 1.6 ± 0.89 (0–3) | 611.36 ± 33.35 (567.14–663.81) | 1222.72 ± 66.71 |
| **3** | 22.86 ± 1.40 (21.57–24.03) | 0.53 | 16.74 | 58.87 | 23.86 | 4.0 ± 0.89 (3–5) | 388.07 ± 14.87 (374.50–417.32) | 1164.21 ± 44.62 |
| **4** | 31.83 ± 0.61 (30.63–32.00) | 0.69 | 17.92 | 69.88 | 11.51 | 8.2 ± 3.84 (5–15) | 287.19 ± 4.06 (281.22–293.78) | 1148.75 ± 16.22 |
| **5** | 39.91 ± 0.87 (39.30–42.38) | 0.94 | 19.18 | 73.67 | 6.21 | 11.2 ± 8.82 (6–31) | 223.57 ± 5.41 (212.36–228.99) | 1117.85 ± 27.03 |
| **8** | 61.75 ± 0.79 (60.38–62.26) | 1.13 | 21.77 | 74.28 | 2.82 | 57.0 ± 11.7 (41–73) | 146.21 ± 1.06 (144.56–149.06) | 1169.66 ± 8.47 |
| **16** | 111.25 ± 1.02 (108.25–114.10) | 0.79 | 24.99 | 73.41 | 0.81 | 227.6 ± 17.2 (210–255) | 80.82 ± 0.90 (78.88–83.14) | 1293.18 ± 14.36 |

Таблица эффективности

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 757.31 | 757.31 | 100.00 % |
| 2 | 611.36 | 1222.72 | 80.73 % |
| 3 | 388.07 | 1164.21 | 51.24 % |
| 4 | 287.19 | 1148.75 | 37.93 % |
| 5 | 223.57 | 1117.85 | 29.54 % |
| 8 | 146.21 | 1169.66 | 19.31 % |
| 16 | 80.82 | 1293.18 | 10.67 % |

**Краткий вывод по таблице**

**Если нужен максимум скорости одного процесса**

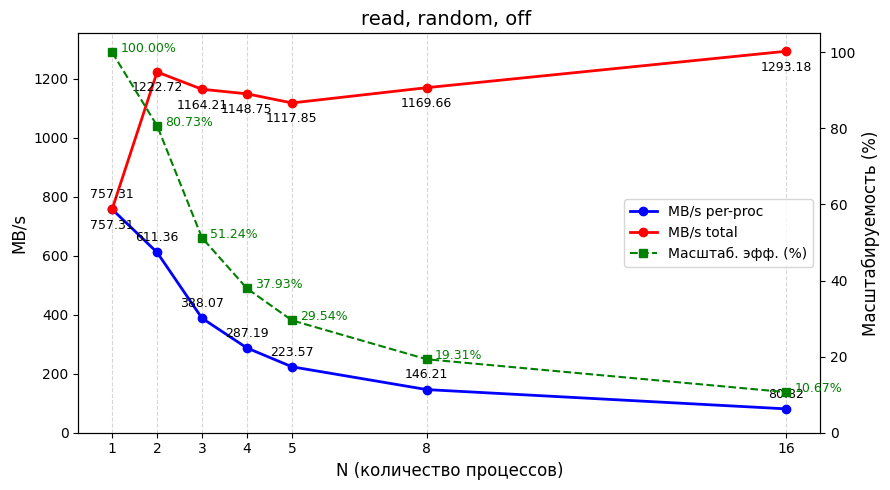
**N = 1** — самый быстрый и эффективный (эталон).

**Если нужна максимальная суммарная скорость диска**

**N = 16** — самая высокая total MB/s (≈1293 MB/s), но Масштаб. эфф плохая (≈10%).

**Лучший баланс «эффективность → прирост скорости»**

**N = 2** — почти удвоение скорости (≈1223 MB/s), при этом масштабируемость всё ещё высокая (≈81%).  
После N=2 начинается резкое падение эффективности.

График эффективности  


**Описание нагрузки read, sequence, off**

Тестируемая программа: io — нагрузчик случайного чтения  
Режим:./io read -b 1048576 -c 9000 -f ./io\_test.bin -t **sequence** -d off -R 1  
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,3,4,5,8,16).

**1 процесс**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.47% ± 0.28% Мин = 0.35%, Макс = 1.01%

• SYS%  
Среднее = 9.65% ± 0.25% Мин = 9.15%, Макс = 10.03%

• IOWAIT%  
Среднее = 17.40% ± 0.62% Мин = 17.06%, Макс = 18.99%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 15.08 с ± 0.28 с Мин = 14.67 с, Макс = 15.52 с

• Переключения контекста (сумма по экземплярам в прогонах)  
• total voluntary ctx switches: среднее = 74,068.0 ± 321.3 (Мин = 7,184, Макс = 7,649)   
• total involuntary ctx switches: среднее = 1.6 ± 1.0 (Мин = 0, Макс = 3)  
• involuntary ctx switches per-process (total / N): среднее = 1.600 ± 1.106 (Мин = 0.000, Макс = 3.000)

• Пер-процесс пропускная способность (MB/s)  
Среднее = 596.83 ± 11.03 (Мин = 579.86, Макс = 613.54)

• Суммарная пропускная способность (вся система)  
Среднее = 596.83 ± 11.03 MB/s (Мин = 579.86, Макс = 613.54) — (для N=1 суммарная = пер-процесс)

**2 процесса**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.53% ± 0.07% Мин = 0.39%, Макс = 0.85%

• SYS%  
Среднее = 13.05% ± 0.26% Мин = 12.68%, Макс = 13.33%

• IOWAIT%  
Среднее = 39.45% ± 1.08% Мин = 38.63%, Макс = 41.37%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 19.51 с ± 0.34 с Мин = 19.06 с, Макс = 20.00 с

• Переключения контекста (сумма по экземплярам в прогонах)  
• total voluntary ctx switches: среднее = 38,603.0 ± 506.7 (Мин = 37,635, Макс = 39,088)  
• total involuntary ctx switches: среднее = 5.2 ± 1.9 (Мин = 4, Макс = 9)  
• involuntary ctx switches per-process (total / N): среднее = 2.60 ± 0.95 (Мин = 2.00, Макс = 4.50)

• Пер-процесс пропускная способность (MB/s)  
Среднее = 461.38 ± 8.03 (Мин = 449.92, Макс = 472.21)

• Суммарная пропускная способность (вся система)  
Среднее = 922.75 ± 16.02 MB/s (Мин = 901.00, Макс = 943.24)

**3 процесса**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.60% ± 0.16% Мин = 0.47%, Макс = 0.77%

• SYS%  
Среднее = 17.76% ± 0.27% Мин = 17.54%, Макс = 18.16%

• IOWAIT%  
Среднее = 60.15% ± 0.63% Мин = 59.90%, Макс = 60.45%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 23.24 с ± 0.29 с Мин = 22.86 с, Макс = 23.81 с

• Переключения контекста (сумма по экземплярам в прогонах)  
• total voluntary ctx switches: среднее = 239 ± 17.9

• total involuntary ctx switches: среднее = 14.2 ± 2.9 (Мин = 11, Макс = 18)  
• involuntary ctx switches per-process (total / N): среднее = 4.73 ± 0.93 (Мин = 3.67, Макс = 6.00)

• Пер-процесс пропускная способность (MB/s)  
Среднее = 387.30 ± 4.82 (Мин = 377.94, Макс = 393.61)

• Суммарная пропускная способность (вся система)  
Среднее = 1,161.91 ± 14.30 MB/s (Мин = 1,136.35, Макс = 1,178.23)

**4 процесса**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.82% ± 0.21% Мин = 0.67%, Макс = 1.21%

• SYS%  
Среднее = 20.95% ± 0.37% Мин = 20.92%, Макс = 21.28%

• IOWAIT%  
Среднее = 74.82% ± 0.24% Мин = 74.34%, Макс = 74.84%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 28.43 с ± 0.30 с Мин = 27.93 с, Макс = 28.79 с

• Переключения контекста (сумма по экземплярам в прогонах)  
• total voluntary ctx switches: среднее = 527 ± 28.6  
• total involuntary ctx switches: среднее = 134.4 ± 20.1 (Мин = 122, Макс = 175)  
• involuntary ctx switches per-process (total / N): среднее = 34.60 ± 4.97 (Мин = 31.00, Макс = 43.75)

• Пер-процесс пропускная способность (MB/s)  
Среднее = 316.56 ± 3.41 (Мин = 312.66, Макс = 322.24)

• Суммарная пропускная способность (вся система)  
Среднее = 1,266.25 ± 13.37 MB/s (Мин = 1,254.05, Макс = 1,285.27)

**5 процессов**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.78% ± 0.17% Мин = 0.63%, Макс = 1.03%

• SYS%  
Среднее = 22.25% ± 0.28% Мин = 21.38%, Макс = 22.74%

• IOWAIT%  
Среднее = 67.74% ± 0.38% Мин = 67.38%, Макс = 68.19%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 30.47 с ± 0.55 с Мин = 29.71 с, Макс = 31.30 с

• Переключения контекста (сумма по экземплярам в прогонах)  
• total voluntary ctx switches: 672.2 ± 29.2

• total involuntary ctx switches: среднее = 517.2 ± 20.8 (Мин = 511, Макс = 567)  
• involuntary ctx switches per-process (total / N): среднее = 107.44 ± 4.41 (Мин = 102.20, Макс = 113.40)

• Пер-процесс пропускная способность (MB/s)  
Среднее = 295.44 ± 5.35 (Мин = 287.54, Макс = 302.96)

• Суммарная пропускная способность (вся система)  
Среднее = 1,477.18 ± 26.51 MB/s (Мин = 1,442.52, Макс = 1,509.47)

**8 процессов**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 1.00% ± 0.21% Мин = 0.80%, Макс = 1.38%

• SYS%  
Среднее = 28.37% ± 0.46% Мин = 27.46%, Макс = 28.97%

• IOWAIT%  
Среднее = 67.88% ± 0.34% Мин = 67.43%, Макс = 68.36%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 32.93 с ± 0.36 с Мин = 32.23 с, Макс = 33.62 с

• Переключения контекста (сумма по экземплярам в прогонах)  
• total voluntary ctx switches: 728210 ± 67.1  
• total involuntary ctx switches: среднее = 3,240.4 ± 90.6 (Мин = 3,123, Макс = 3,350)  
• involuntary ctx switches per-process (total / N): среднее = 405.05 ± 11.33 (Мин = 390.38, Макс = 418.75)

• Пер-процесс пропускная способность (MB/s)  
Среднее = 273.34 ± 3.02 (Мин = 267.69, Макс = 279.20)

• Суммарная пропускная способность (вся система)  
Среднее = 2,186.72 ± 22.81 MB/s (Мин = 2,153.26, Макс = 2,221.00)

**16 процессов**

Наблюдаемые (числовые результаты)

• USER%  
Среднее = 1.25% ± 0.31% Мин = 0.97%, Макс = 1.80%

• SYS%  
Среднее = 39.82% ± 0.18% Мин = 39.61%, Макс = 40.03%

• IOWAIT%  
Среднее = 57.54% ± 0.36% Мин = 56.74%, Макс = 57.99%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 38.86 с ± 0.39 с Мин = 38.19 с, Макс = 39.85 с

• Переключения контекста   
• total involuntary ctx switches: среднее = 13,764.4 ± 572.9 (Мин = 13,333, Макс = 14,700)  
• involuntary ctx switches per-process (total / N): среднее = 860.28 ± 35.81 (Мин = 833.31, Макс = 918.75)

• Пер-процесс пропускная способность (MB/s)  
Среднее = 231.60 ± 2.32 (Мин = 225.82, Макс = 235.67)

• Суммарная пропускная способность (вся система)  
Среднее = 3,705.64 ± 28.59 MB/s (Мин = 3,649.91, Макс = 3,731.15)

**Выводы**

Общая таблица по каждому процессу

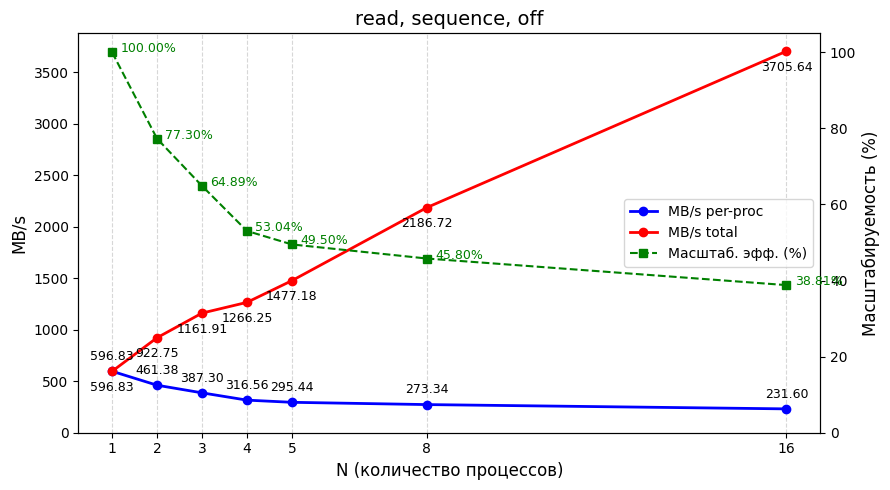
| **N** | **avg wall (s) ± sd (min–max)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW per-proc (avg ± sd)** | **per-proc MB/s (avg ± sd)** | **total MB/s (avg ± sd)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 15.08 ± 0.28 (14.67–15.52) | 0.47 | 9.65 | 17.40 | 72.48 | 1.60 ± 1.11 | 596.83 ± 11.03 | 596.83 ± 11.03 |
| 2 | 19.51 ± 0.34 (19.06–20.00) | 0.53 | 13.05 | 39.45 | 46.97 | 2.60 ± 0.95 | 461.38 ± 8.03 | 922.75 ± 16.02 |
| 3 | 23.24 ± 0.29 (22.86–23.81) | 0.60 | 17.76 | 60.15 | 21.49 | 4.73 ± 0.93 | 387.30 ± 4.82 | 1,161.91 ± 14.30 |
| 4 | 28.43 ± 0.30 (27.93–28.79) | 0.82 | 20.95 | 74.82 | 3.41 | 34.60 ± 4.97 | 316.56 ± 3.41 | 1,266.25 ± 13.37 |
| 5 | 30.47 ± 0.55 (29.71–31.30) | 0.78 | 22.25 | 67.74 | 9.22 | 107.44 ± 4.41 | 295.44 ± 5.35 | 1,477.18 ± 26.51 |
| 8 | 32.93 ± 0.36 (32.23–33.62) | 1.00 | 28.37 | 67.88 | 2.75 | 405.05 ± 11.33 | 273.34 ± 3.02 | 2,186.72 ± 22.81 |
| 16 | 38.86 ± 0.39 (38.19–39.85) | 1.25 | 39.82 | 57.54 | 1.39 | 860.28 ± 35.81 | 231.60 ± 2.32 | 3,705.64 ± 28.59 |

Таблица эффективности

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф.** |
| --- | --- | --- | --- |
| 1 | 596.83 | 596.83 | 100.00 % |
| 2 | 461.38 | 922.75 | 77.30 % |
| 3 | 387.30 | 1161.91 | 64.89 % |
| 4 | 316.56 | 1266.25 | 53.04 % |
| 5 | 295.44 | 1477.18 | 49.50 % |
| 8 | 273.34 | 2186.72 | 45.80 % |
| 16 | 231.60 | 3705.64 | 38.81 % |

Краткий вывод

* Максимальная скорость одного процесса: **N = 1** (596.83 MB/s) — эталон для «максимальной скорости одного экземпляра».
* Максимальная суммарная скорость (total): **N = 16** (≈3705.6 MB/s), и при этом масштабируемость не самая плохая(~38.8%).
* Лучший баланс «эффективность → прирост суммарной скорости»:
  + **N = 2** — простое удвояющееся поведение даёт неплохой прирост суммарной скорости (≈922.8 MB/s) при высокой масштабируемости (≈77%).
  + Если нужна ещё большая суммарная пропускная способность и приемлемо снижение «эффективности», можно брать **N = 4–5** (total ≈1266–1477 MB/s) — это уже ≈2–2.5× от одного процесса, но масштабируемость падает до ~53–49%.
* Вывод: если важна эффективность (чтобы каждый процесс работал близко к одиночному максимуму) — выбирай **N = 2**; если важен чистый throughput — выбирай **N = 16** (или 8/5/4 в зависимости от компромисса между суммарной скоростью и эффективностью).

График эффективности  


**Описание нагрузки read, sequence, on**

Тестируемая программа: io — нагрузчик случайного чтения  
Режим:./io read -b 1048576 -c 9000 -f ./io\_test.bin -t sequence -d on -R 1  
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,3,4,5,8).

**1 процесс**

USER%: Среднее = 0.35% ± 0.03% Мин = 0.32%, Макс = 0.41%

SYS%: Среднее = 1.10% ± 0.08% Мин = 0.99%, Макс = 1.24%

IOWAIT%: Среднее = 23.84% ± 0.13% Мин = 23.69%, Макс = 24.07%

IDLE%: Среднее = 74.71% ± 0.10% Мин = 74.54%, Макс = 74.84%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 14.72 с ± 4.10 с Мин = 10.98 с, Макс = 22.18 с

Переключения контекста (сумма по экземплярам в прогонах):

total voluntary ctx switches: среднее = 9,001.0 ± 0.0 (Мин = 9,001, Макс = 9,001)

total involuntary ctx switches: среднее = 0.2 ± 0.4 (Мин = 0, Макс = 1)

involuntary ctx switches per-process (total / N): среднее = 0.20 ± 0.40 (Мин = 0.00, Макс = 1.00)

Пер-процесс пропускная способность (MB/s): Среднее = 652.58 ± 150.59 (Мин = 405.70, Макс = 819.72)

Суммарная пропускная способность (вся система): Среднее = 652.58 ± 150.59 MB/s (Мин = 405.70, Макс = 819.72) — (для N=1 суммарная = пер-процесс)

**2 процесса**

USER%: Среднее = 0.42% ± 0.05% Мин = 0.33%, Макс = 0.48%

SYS%: Среднее = 1.67% ± 0.18% Мин = 1.35%, Макс = 1.83%

IOWAIT%: Среднее = 42.45% ± 2.10% Мин = 40.11%, Макс = 45.18%

IDLE%: Среднее = 55.47% ± 2.03% Мин = 53.00%, Макс = 57.68%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 18.62 с ± 6.50 с Мин = 13.49 с, Макс = 30.18 с

Переключения контекста (сумма по экземплярам в прогонах):

total voluntary ctx switches: среднее = 18,001.0 ± 0.0 (Мин = 18,001, Макс = 18,001)

total involuntary ctx switches: среднее = 0.2 ± 0.4 (Мин = 0, Макс = 1)

involuntary ctx switches per-process (total / N): среднее = 0.10 ± 0.20 (Мин = 0.00, Макс = 0.50)

Пер-процесс пропускная способность (MB/s): Среднее = 526.94 ± 148.57 (Мин = 298.23, Макс = 667.11)

Суммарная пропускная способность (вся система): Среднее = 1,053.88 ± 296.86 MB/s (Мин = 596.46, Макс = 1,334.21)

**3 процесса**

USER%: Среднее = 0.45% ± 0.02% Мин = 0.43%, Макс = 0.47%

SYS%: Среднее = 2.02% ± 0.09% Мин = 1.90%, Макс = 2.14%

IOWAIT%: Среднее = 58.26% ± 7.92% Мин = 44.98%, Макс = 63.80%

IDLE%: Среднее = 39.26% ± 7.86% Мин = 33.77%, Макс = 52.56%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 24.71 с ± 4.39 с Мин = 18.73 с, Макс = 30.23 с

Переключения контекста (сумма по экземплярам в прогонах):

total voluntary ctx switches: среднее = 27,001.8 ± 1.5 (Мин = 27,000, Макс = 27,004)

total involuntary ctx switches: среднее = 0.2 ± 0.4 (Мин = 0, Макс = 1)

involuntary ctx switches per-process (total / N): среднее = 0.07 ± 0.13 (Мин = 0.00, Макс = 0.33)

Пер-процесс пропускная способность (MB/s): Среднее = 374.42 ± 66.61 (Мин = 297.89, Макс = 480.60)

Суммарная пропускная способность (вся система): Среднее = 1,123.26 ± 199.74 MB/s (Мин = 893.68, Макс = 1,441.81)

**4 процесса**

USER%: Среднее = 0.49% ± 0.08% Мин = 0.40%, Макс = 0.61%

SYS%: Среднее = 2.57% ± 0.48% Мин = 2.23%, Макс = 3.60%

IOWAIT%: Среднее = 73.84% ± 6.24% Мин = 64.56%, Макс = 81.53%

IDLE%: Среднее = 23.10% ± 6.08% Мин = 15.69%, Макс = 32.43%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 30.87 с ± 5.91 с Мин = 25.04 с, Макс = 40.19 с

Переключения контекста (сумма по экземплярам в прогонах):

total voluntary ctx switches: среднее = 36,006.0 ± 4.0 (Мин = 36,002, Макс = 36,013)

total involuntary ctx switches: среднее = 0.8 ± 0.8 (Мин = 0, Макс = 2)

involuntary ctx switches per-process (total / N): среднее = 0.20 ± 0.20 (Мин = 0.00, Макс = 0.50)

Пер-процесс пропускная способность (MB/s): Среднее = 299.13 ± 47.37 (Мин = 223.96, Макс = 359.43)

Суммарная пропускная способность (вся система): Среднее = 1,196.51 ± 189.29 MB/s (Мин = 895.85, Макс = 1,437.72)

**5 процессов**

USER%: Среднее = 0.49% ± 0.01% Мин = 0.48%, Макс = 0.51%

SYS%: Среднее = 2.55% ± 0.15% Мин = 2.34%, Макс = 2.77%

IOWAIT%: Среднее = 83.76% ± 4.40% Мин = 75.59%, Макс = 86.83%

IDLE%: Среднее = 13.20% ± 4.37% Мин = 10.21%, Макс = 21.28%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 35.13 с ± 2.57 с Мин = 31.42 с, Макс = 38.56 с

Переключения контекста (сумма по экземплярам в прогонах):

total voluntary ctx switches: среднее = 45,002.6 ± 1.8 (Мин = 45,000, Макс = 45,005)

total involuntary ctx switches: среднее = 0.8 ± 1.2 (Мин = 0, Макс = 3)

involuntary ctx switches per-process (total / N): среднее = 0.16 ± 0.24 (Мин = 0.00, Макс = 0.60)

Пер-процесс пропускная способность (MB/s): Среднее = 257.43 ± 19.91 (Мин = 233.40, Макс = 286.45)

Суммарная пропускная способность (вся система): Среднее = 1,287.16 ± 99.53 MB/s (Мин = 1,167.00, Макс = 1,432.23)

**8 процессов**

USER%: Среднее = 0.54% ± 0.02% Мин = 0.51%, Макс = 0.56%

SYS%: Среднее = 2.60% ± 0.17% Мин = 2.28%, Макс = 2.74%

IOWAIT%: Среднее = 88.70% ± 5.42% Мин = 79.63%, Макс = 93.40%

IDLE%: Среднее = 8.16% ± 5.29% Мин = 3.46%, Макс = 17.54%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 48.21 с ± 1.30 с Мин = 46.10 с, Макс = 49.36 с

Переключения контекста (сумма по экземплярам в прогонах):

total voluntary ctx switches: среднее = 72,005.6 ± 2.1 (Мин = 72,002, Макс = 72,007)

total involuntary ctx switches: среднее = 1.8 ± 1.2 (Мин = 0, Макс = 3)

involuntary ctx switches per-process (total / N): среднее = 0.23 ± 0.15 (Мин = 0.00, Макс = 0.38)

Пер-процесс пропускная способность (MB/s): Среднее = 186.89 ± 5.01 (Мин = 182.33, Макс = 195.23)

Суммарная пропускная способность (вся система): Среднее = 1,495.15 ± 40.08 MB/s (Мин = 1,458.65, Макс = 1,561.87)

**16 процессов**

USER%: Среднее = 0.52% ± 0.03% Мин = 0.49%, Макс = 0.57%

SYS%: Среднее = 2.52% ± 0.10% Мин = 2.34%, Макс = 2.62%

IOWAIT%: Среднее = 95.74% ± 0.34% Мин = 95.20%, Макс = 96.07%

IDLE%: Среднее = 1.22% ± 0.32% Мин = 0.77%, Макс = 1.64%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 93.55 с ± 0.84 с Мин = 92.11 с, Макс = 94.17 с

Переключения контекста (сумма по экземплярам в прогонах):

total voluntary ctx switches: среднее = 144,014.2 ± 2.5 (Мин = 144,010, Макс = 144,017)

total involuntary ctx switches: среднее = 3.8 ± 1.6 (Мин = 2, Макс = 7)

involuntary ctx switches per-process (total / N): среднее = 0.24 ± 0.10 (Мин = 0.13, Макс = 0.44)

Пер-процесс пропускная способность (MB/s): Среднее = 96.08 ± 0.87 (Мин = 95.39, Макс = 97.58)

Суммарная пропускная способность (вся система): Среднее = 1,537.30 ± 13.92 MB/s (Мин = 1,526.19, Макс = 1,561.27)

**Вывод**

Общая таблица по каждому процессу

| **N** | **avg wall (s) ± sd (min–max)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW per-proc (avg ± sd)** | **per-proc MB/s (avg ± sd)** | **total MB/s (avg ± sd)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 14.72 ± 4.10 (10.98–22.18) | 0.35 | 1.10 | 23.84 | 74.71 | 0.20 ± 0.40 | 652.58 ± 150.59 | 652.58 ± 150.59 |
| 2 | 18.62 ± 6.50 (13.49–30.18) | 0.42 | 1.67 | 42.45 | 55.47 | 0.10 ± 0.20 | 526.94 ± 148.57 | 1,053.88 ± 296.86 |
| 3 | 24.71 ± 4.39 (18.73–30.23) | 0.45 | 2.02 | 58.26 | 39.26 | 0.07 ± 0.13 | 374.42 ± 66.61 | 1,123.26 ± 199.74 |
| 4 | 30.87 ± 5.91 (25.04–40.19) | 0.49 | 2.57 | 73.84 | 23.10 | 0.20 ± 0.20 | 299.13 ± 47.37 | 1,196.51 ± 189.29 |
| 5 | 35.13 ± 2.57 (31.42–38.56) | 0.49 | 2.55 | 83.76 | 13.20 | 0.16 ± 0.24 | 257.43 ± 19.91 | 1,287.16 ± 99.53 |
| 8 | 48.21 ± 1.30 (46.10–49.36) | 0.54 | 2.60 | 88.70 | 8.16 | 0.23 ± 0.15 | 186.89 ± 5.01 | 1,495.15 ± 40.08 |
| 16 | 93.55 ± 0.84 (92.11–94.17) | 0.52 | 2.52 | 95.74 | 1.22 | 0.24 ± 0.10 | 96.08 ± 0.87 | 1,537.30 ± 13.92 |

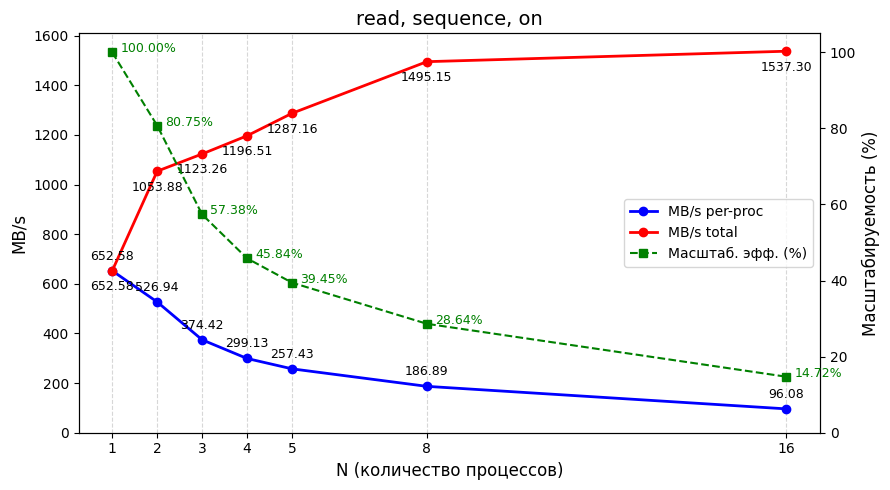
Таблица эффективности (read, sequence, on)

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 652.58 | 652.58 | 100.00 % |
| 2 | 526.94 | 1053.88 | 80.75 % |
| 3 | 374.42 | 1123.26 | 57.38 % |
| 4 | 299.13 | 1196.51 | 45.84 % |
| 5 | 257.43 | 1287.16 | 39.45 % |
| 8 | 186.89 | 1495.15 | 28.64 % |
| 16 | 96.08 | 1537.30 | 14.72 % |

Краткий вывод по таблице

* Если нужен максимум скорости одного процесса: N = 1 — самый быстрый и эффективный (эталон).
* Если нужна максимальная суммарная скорость диска: N = 16 — самая высокая total MB/s (≈1537.30 MB/s), но Масштаб. эфф. низкая (≈14.7%).
* Лучший баланс «эффективность → прирост скорости»: N = 2 — почти удвоение общей скорости (≈1054 MB/s) при высокой масштабируемости (~80.8%).
* После N = 2 эффективность резко падает; дальнейшее добавление потоков даёт заметный рост total MB/s до N≈8–16, но с сильной потерей эффективности (много конкуренции/ожиданий).

График эффективности



**Описание нагрузки write, sequence, off**

Тестируемая программа: io — нагрузчик случайного чтения

Режим: ./io write -b 1048576 -c 1024 -f ./io\_test\_write.bin -t sequence -d off -R 1   
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,3,4,5,8).

**1 процесс**

* USER%: Среднее = 0.61% ± 0.45% Мин = 0.23%, Макс = 1.36%
* SYS%: Среднее = 13.31% ± 7.61% Мин = 6.03%, Макс = 21.82%
* IOWAIT%: Среднее = 18.77% ± 11.31% Мин = 6.71%, Макс = 30.29%
* IDLE%: Среднее = 67.31% ± 4.46% Мин = 61.38%, Макс = 71.76%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 2.22 с ± 1.10 с Мин = 1.09 с, Макс = 3.52 с
* Переключения контекста (сумма по экземплярам в прогонах):
  + total voluntary ctx switches: среднее = 70.8 ± 35.3 (Мин = 34, Макс = 121)
  + total involuntary ctx switches: среднее = 6.4 ± 1.5 (Мин = 4, Макс = 8)
  + involuntary ctx switches per-process (total / N): среднее = 6.40 ± 1.52 (Мин = 4.00, Макс = 8.00)
* Пер-процесс пропускная способность (MB/s): Среднее = 582.73 ± 281.99 (Мин = 290.26, Макс = 933.16)
* Суммарная пропускная способность (вся система): Среднее = 582.73 ± 281.99 MB/s (Мин = 290.26, Макс = 933.16) — (для N=1 суммарная = пер-процесс)

**2 процесса**

* USER%: Среднее = 0.69% ± 0.14% Мин = 0.57%, Макс = 0.87%
* SYS%: Среднее = 13.27% ± 2.42% Мин = 9.67%, Макс = 15.94%
* IOWAIT%: Среднее = 39.17% ± 2.48% Мин = 36.28%, Макс = 42.90%
* IDLE%: Среднее = 46.87% ± 2.37% Мин = 44.59%, Макс = 50.08%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 6.66 с ± 1.04 с Мин = 5.07 с, Макс = 7.72 с
* Переключения контекста (сумма по экземплярам в прогонах):
  + total voluntary ctx switches: среднее = 443.0 ± 115.5 (Мин = 274, Макс = 580)
  + total involuntary ctx switches: среднее = 26.2 ± 10.8 (Мин = 12, Макс = 41)
  + involuntary ctx switches per-process (total / N): среднее = 13.10 ± 5.41 (Мин = 6.00, Макс = 20.50)
* Пер-процесс пропускная способность (MB/s): Среднее = 157.30 ± 26.71 (Мин = 132.65, Макс = 202.81)
* Суммарная пропускная способность (вся система): Среднее = 314.60 ± 53.42 MB/s (Мин = 269.29, Макс = 405.62)

**3 процесса**

* USER%: Среднее = 0.71% ± 0.06% Мин = 0.64%, Макс = 0.79%
* SYS%: Среднее = 14.46% ± 2.79% Мин = 11.18%, Макс = 18.75%
* IOWAIT%: Среднее = 55.97% ± 2.45% Мин = 52.09%, Макс = 58.37%
* IDLE%: Среднее = 28.66% ± 1.24% Мин = 26.98%, Макс = 30.01%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 12.17 с ± 0.76 с Мин = 11.14 с, Макс = 13.23 с
* Переключения контекста (сумма по экземплярам в прогонах):
  + total voluntary ctx switches: среднее = 1119.2 ± 134.8 (Мин = 980, Макс = 1325)
  + total involuntary ctx switches: среднее = 120.6 ± 50.1 (Мин = 53, Макс = 200)
  + involuntary ctx switches per-process (total / N): среднее = 40.20 ± 16.71 (Мин = 17.67, Макс = 66.67)
* Пер-процесс пропускная способность (MB/s): Среднее = 84.76 ± 5.90 (Мин = 77.55, Макс = 92.29)
* Суммарная пропускная способность (вся система): Среднее = 254.28 ± 17.71 MB/s (Мин = 232.66, Макс = 276.87)

**4 процесса**

* USER%: Среднее = 0.55% ± 0.19% Мин = 0.41%, Макс = 0.86%
* SYS%: Среднее = 5.43% ± 4.77% Мин = 2.28%, Макс = 13.53%
* IOWAIT%: Среднее = 68.85% ± 3.37% Мин = 63.21%, Макс = 72.10%
* IDLE%: Среднее = 25.18% ± 2.50% Мин = 22.40%, Макс = 28.69%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 21.85 с ± 3.30 с Мин = 17.74 с, Макс = 25.22 с
* Переключения контекста (сумма по экземплярам в прогонах):
  + total voluntary ctx switches: среднее = 2979.4 ± 759.8 (Мин = 1992, Макс = 3947)
  + total involuntary ctx switches: среднее = 193.0 ± 206.0 (Мин = 30, Макс = 560)
  + involuntary ctx switches per-process (total / N): среднее = 48.25 ± 51.50 (Мин = 7.50, Макс = 140.00)
* Пер-процесс пропускная способность (MB/s): Среднее = 48.10 ± 6.72 (Мин = 40.94, Макс = 58.03)
* Суммарная пропускная способность (вся система): Среднее = 192.40 ± 26.88 MB/s (Мин = 163.74, Макс = 232.10)

**5 процессов**

* USER%: Среднее = 0.72% ± 0.57% Мин = 0.39%, Макс = 1.76%
* SYS%: Среднее = 2.48% ± 0.27% Мин = 2.11%, Макс = 2.75%
* IOWAIT%: Среднее = 78.27% ± 2.64% Мин = 74.42%, Макс = 81.51%
* IDLE%: Среднее = 18.53% ± 2.96% Мин = 15.26%, Макс = 22.35%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 39.41 с ± 7.69 с Мин = 33.50 с, Макс = 50.87 с
* Переключения контекста (сумма по экземплярам в прогонах):
  + total voluntary ctx switches: среднее = 5913.8 ± 1164.2 (Мин = 4272, Макс = 7371)
  + total involuntary ctx switches: среднее = 171.4 ± 62.3 (Мин = 100, Макс = 246)
  + involuntary ctx switches per-process (total / N): среднее = 34.28 ± 12.46 (Мин = 20.00, Макс = 49.20)
* Пер-процесс пропускная способность (MB/s): Среднее = 26.86 ± 4.01 (Мин = 20.26, Макс = 30.76)
* Суммарная пропускная способность (вся система): Среднее = 134.28 ± 20.04 MB/s (Мин = 101.28, Макс = 153.80)

**8 процессов**

* USER%: Среднее = 0.49% ± 0.02% Мин = 0.46%, Макс = 0.51%
* SYS%: Среднее = 2.35% ± 0.58% Мин = 1.84%, Макс = 3.45%
* IOWAIT%: Среднее = 87.36% ± 0.93% Мин = 86.00%, Макс = 88.46%
* IDLE%: Среднее = 9.90% ± 1.16% Мин = 8.92%, Макс = 11.48%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 77.38 с ± 13.90 с Мин = 62.88 с, Макс = 99.32 с
* Переключения контекста (сумма по экземплярам в прогонах):
  + total voluntary ctx switches: среднее = 14227.0 ± 2594.8 (Мин = 12040, Макс = 18567)
  + total involuntary ctx switches: среднее = 343.2 ± 148.4 (Мин = 183, Макс = 584)
  + involuntary ctx switches per-process (total / N): среднее = 42.90 ± 17.36 (Мин = 22.88, Макс = 73.00)
* Пер-процесс пропускная способность (MB/s): Среднее = 13.71 ± 2.27 (Мин = 10.37, Макс = 16.36)
* Суммарная пропускная способность (вся система): Среднее = 109.64 ± 18.11 MB/s (Мин = 84.94, Макс = 130.86)

**16 процессов**

* USER%: Среднее = 0.51% ± 0.02% Мин = 0.47%, Макс = 0.53%
* SYS%: Среднее = 1.50% ± 0.20% Мин = 1.32%, Макс = 1.82%
* IOWAIT%: Среднее = 93.93% ± 1.64% Мин = 92.62%, Макс = 96.37%
* IDLE%: Среднее = 4.06% ± 1.65% Мин = 1.60%, Макс = 5.52%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс): Среднее = 284.44 с ± 61.70 с Мин = 198.52 с, Макс = 335.43 с
* Переключения контекста (сумма по экземплярам в прогонах):
  + total voluntary ctx switches: среднее = 87471.4 ± 17291.6 (Мин = 62944, Макс = 105653)
  + total involuntary ctx switches: среднее = 482.8 ± 179.1 (Мин = 205, Макс = 660)
  + involuntary ctx switches per-process (total / N): среднее = 30.17 ± 10.42 (Мин = 12.81, Макс = 41.25)
* Пер-процесс пропускная способность (MB/s): Среднее = 3.77 ± 0.93 (Мин = 3.03, Макс = 5.22)
* Суммарная пропускная способность (вся система): Среднее = 60.27 ± 14.84 MB/s (Мин = 48.48, Макс = 83.52)

**Вывод**

| **N** | **avg wall (s) ± sd (min–max)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW per-proc (avg ± sd)** | **per-proc MB/s (avg ± sd)** | **total MB/s (avg ± sd)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2.22 ± 1.10 (1.09–3.52) | 0.61 | 13.31 | 18.77 | 67.31 | 6.40 ± 1.52 | 582.73 ± 281.99 | 582.73 ± 281.99 |
| 2 | 6.66 ± 1.04 (5.07–7.72) | 0.69 | 13.27 | 39.17 | 46.87 | 13.10 ± 5.41 | 157.30 ± 26.71 | 314.60 ± 53.42 |
| 3 | 12.17 ± 0.76 (11.14–13.23) | 0.71 | 14.46 | 55.97 | 28.66 | 40.20 ± 16.71 | 84.76 ± 5.90 | 254.28 ± 17.71 |
| 4 | 21.85 ± 3.30 (17.74–25.22) | 0.55 | 5.43 | 68.85 | 25.18 | 48.25 ± 51.50 | 48.10 ± 6.72 | 192.40 ± 26.88 |
| 5 | 39.41 ± 7.69 (33.50–50.87) | 0.72 | 2.48 | 78.27 | 18.53 | 34.28 ± 12.46 | 26.86 ± 4.01 | 134.28 ± 20.04 |
| 8 | 77.38 ± 13.90 (62.88–99.32) | 0.49 | 2.35 | 87.36 | 9.90 | 42.90 ± 17.36 | 13.71 ± 2.27 | 109.64 ± 18.11 |
| 16 | 284.44 ± 61.70 (198.52–335.43) | 0.51 | 1.50 | 93.93 | 4.06 | 30.17 ± 10.42 | 3.77 ± 0.93 | 60.27 ± 14.84 |

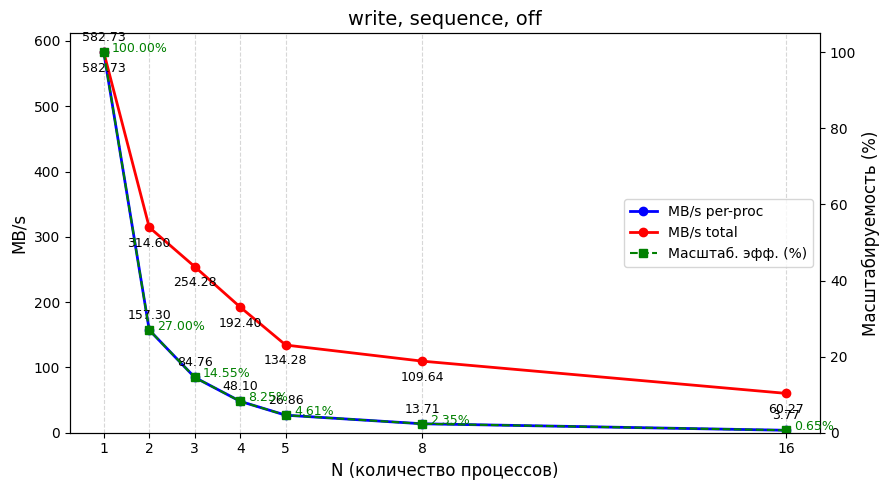
Таблица эффективности (write, sequence, off)

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 582.73 | 582.73 | 100.00 % |
| 2 | 157.30 | 314.60 | 27.00 % |
| 3 | 84.76 | 254.28 | 14.55 % |
| 4 | 48.10 | 192.40 | 8.25 % |
| 5 | 26.86 | 134.28 | 4.61 % |
| 8 | 13.71 | 109.64 | 2.35 % |
| 16 | 3.77 | 60.27 | 0.65 % |

Краткий вывод по таблице

* Если нужен максимум скорости одного процесса — N = 1 (≈582.7 MB/s).
* Если нужна максимальная суммарная скорость диска — тоже N = 1 (в этой выборке суммарная скорость падает при увеличении числа писателей).
* Масштабируемость очень плохая: при запуске 2 копий остаётся лишь ≈27% от идеала, затем падает до ~0.65% при 16. Это указывает на сильную взаимную блокировку/конкуренцию при последовательной записи (контенция на диск/метаданные/очередях).

График эффективности



**Описание нагрузки write, random, off**

Тестируемая программа: io — нагрузчик случайного чтения

Режим: ./io write -b 1048576 -c 1024 -f ./io\_test\_write.bin -t random -d off -R 1   
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,3,4,5,8).

**1 процесс**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 0.54% ± 0.36% Мин = 0.00%, Макс = 1.07%

SYS%  
Среднее = 24.21% ± 1.23% Мин = 22.61%, Макс = 25.68%

IOWAIT%  
Среднее = 6.28% ± 1.46% Мин = 4.37%, Макс = 8.54%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 0.46 с ± 0.02 с Мин = 0.44 с, Макс = 0.49 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 4.8 ± 1.5 (Мин = 3, Макс = 7)

total involuntary ctx switches: среднее = 4.6 ± 0.5 (Мин = 4, Макс = 5)

involuntary ctx switches per-process (total / N): среднее = 4.60 ± 0.55 (Мин = 4.00, Макс = 5.00)

Пер-процесс пропускная способность (MB/s)  
Среднее = 2212.46 ± 78.55 (Мин = 2084.05, Макс = 2304.65)

Суммарная пропускная способность (вся система)  
Среднее = 2212.46 ± 78.55 MB/s (Мин = 2084.05, Макс = 2304.65) — (для N=1 суммарная = пер-процесс)

**2 процесса**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 0.42% ± 0.05% Мин = 0.35%, Макс = 0.50%

SYS%  
Среднее = 3.99% ± 2.01% Мин = 2.33%, Макс = 7.54%

IOWAIT%  
Среднее = 47.79% ± 2.88% Мин = 44.42%, Макс = 52.21%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 8.95 с ± 3.73 с Мин = 4.84 с, Макс = 14.41 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 706.6 ± 376.0 (Мин = 202, Макс = 1116)

total involuntary ctx switches: среднее = 7.0 ± 4.1 (Мин = 2, Макс = 13)

involuntary ctx switches per-process (total / N): среднее = 3.50 ± 2.06 (Мин = 1.00, Макс = 6.50)

Пер-процесс пропускная способность (MB/s)  
Среднее = 150.05 ± 58.89 (Мин = 70.69, Макс = 220.50)

Суммарная пропускная способность (вся система)  
Среднее = 300.01 ± 117.47 MB/s (Мин = 141.38, Макс = 423.46)

**3 процесса**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 0.46% ± 0.04% Мин = 0.41%, Макс = 0.51%

SYS%  
Среднее = 2.16% ± 0.15% Мин = 1.99%, Макс = 2.35%

IOWAIT%  
Среднее = 56.96% ± 2.30% Мин = 53.65%, Макс = 59.76%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 23.54 с ± 1.91 с Мин = 20.38 с, Макс = 25.27 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 2989.2 ± 119.0 (Мин = 2834, Макс = 3143)

total involuntary ctx switches: среднее = 18.0 ± 8.0 (Мин = 4, Макс = 27)

involuntary ctx switches per-process (total / N): среднее = 6.00 ± 2.67 (Мин = 1.33, Макс = 9.00)

Пер-процесс пропускная способность (MB/s)  
Среднее = 42.08 ± 3.51 (Мин = 37.30, Макс = 50.25)

Суммарная пропускная способность (вся система)  
Среднее = 126.19 ± 10.34 MB/s (Мин = 111.90, Макс = 150.62)

**4 процесса**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 0.51% ± 0.03% Мин = 0.47%, Макс = 0.54%

SYS%  
Среднее = 2.05% ± 0.24% Мин = 1.75%, Макс = 2.28%

IOWAIT%  
Среднее = 73.26% ± 6.75% Мин = 62.07%, Макс = 78.16%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 38.93 с ± 6.16 с Мин = 34.02 с, Макс = 47.28 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 4680.4 ± 1197.7 (Мин = 3576, Макс = 6041)

total involuntary ctx switches: среднее = 76.8 ± 14.2 (Мин = 50, Макс = 86)

involuntary ctx switches per-process (total / N): среднее = 19.20 ± 3.54 (Мин = 12.50, Макс = 21.50)

Пер-процесс пропускная способность (MB/s)  
Среднее = 27.79 ± 5.28 (Мин = 20.91, Макс = 35.42)

Суммарная пропускная способность (вся система)  
Среднее = 111.07 ± 20.91 MB/s (Мин = 85.31, Макс = 141.67)

**5 процессов**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 0.48% ± 0.03% Мин = 0.45%, Макс = 0.51%

SYS%  
Среднее = 1.90% ± 0.13% Мин = 1.79%, Макс = 2.13%

IOWAIT%  
Среднее = 76.71% ± 1.93% Мин = 74.69%, Макс = 79.47%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 60.69 с ± 6.98 с Мин = 52.77 с, Макс = 68.95 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 8465.2 ± 1638.4 (Мин = 6470, Макс = 10492)

total involuntary ctx switches: среднее = 98.2 ± 56.5 (Мин = 63, Макс = 197)

involuntary ctx switches per-process (total / N): среднее = 19.64 ± 11.30 (Мин = 12.60, Макс = 39.40)

Пер-процесс пропускная способность (MB/s)  
Среднее = 17.65 ± 2.41 (Мин = 14.00, Макс = 22.23)

Суммарная пропускная способность (вся система)  
Среднее = 88.11 ± 11.69 MB/s (Мин = 71.04, Макс = 110.90)

**8 процессов**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 0.48% ± 0.05% Мин = 0.41%, Макс = 0.53%

SYS%  
Среднее = 1.76% ± 0.06% Мин = 1.70%, Макс = 1.85%

IOWAIT%  
Среднее = 87.43% ± 1.42% Мин = 85.56%, Макс = 89.41%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 111.85 с ± 6.48 с Мин = 103.75 с, Макс = 121.34 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 22148.0 ± 2662.8 (Мин = 19303, Макс = 25778)

total involuntary ctx switches: среднее = 235.8 ± 67.7 (Мин = 139, Макс = 313)

involuntary ctx switches per-process (total / N): среднее = 29.48 ± 8.46 (Мин = 17.38, Макс = 39.13)

Пер-процесс пропускная способность (MB/s)  
Среднее = 9.36 ± 0.69 (Мин = 7.93, Макс = 10.33)

Суммарная пропускная способность (вся система)  
Среднее = 74.85 ± 5.39 MB/s (Мин = 63.56, Макс = 82.28)

**16 процессов**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 0.44% ± 0.05% Мин = 0.38%, Макс = 0.49%

SYS%  
Среднее = 1.91% ± 0.09% Мин = 1.80%, Макс = 2.02%

IOWAIT%  
Среднее = 87.45% ± 1.52% Мин = 85.12%, Макс = 89.13%

Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
Среднее = 209.92 с ± 11.03 с Мин = 195.88 с, Макс = 226.76 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 33660.8 ± 4408.1 (Мин = 28815, Макс = 39236)

total involuntary ctx switches: среднее = 195.4 ± 72.9 (Мин = 99, Макс = 284)

involuntary ctx switches per-process (total / N): среднее = 12.21 ± 4.56 (Мин = 6.19, Макс = 17.75)

Пер-процесс пропускная способность (MB/s)  
Среднее = 5.10 ± 0.27 (Мин = 4.70, Макс = 5.46)

Суммарная пропускная способность (вся система)  
Среднее = 81.58 ± 4.15 MB/s (Мин = 75.28, Макс = 87.34)

**Вывод**

| **N** | **avg wall (s) ± sd (min–max)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW per-proc (avg ± sd)** | **per-proc MB/s (avg ± sd)** | **total MB/s (avg ± sd)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0.46 ± 0.02 (0.44–0.49) | 0.54 | 24.21 | 6.28 | 68.97 | 4.60 ± 0.55 | 2212.46 ± 78.55 | 2212.46 ± 78.55 |
| 2 | 8.95 ± 3.73 (4.84–14.41) | 0.42 | 3.99 | 47.79 | 47.80 | 3.50 ± 2.06 | 150.05 ± 58.89 | 300.01 ± 117.47 |
| 3 | 23.54 ± 1.91 (20.38–25.27) | 0.46 | 2.16 | 56.96 | 40.42 | 6.00 ± 2.67 | 42.08 ± 3.51 | 126.19 ± 10.34 |
| 4 | 38.93 ± 6.16 (34.02–47.28) | 0.51 | 2.05 | 73.26 | 24.18 | 19.20 ± 3.54 | 27.79 ± 5.28 | 111.07 ± 20.91 |
| 5 | 60.69 ± 6.98 (52.77–68.95) | 0.48 | 1.90 | 76.71 | 20.91 | 19.64 ± 11.30 | 17.65 ± 2.41 | 88.11 ± 11.69 |
| 8 | 111.85 ± 6.48 (103.75–121.34) | 0.48 | 1.76 | 87.43 | 10.33 | 29.48 ± 8.46 | 9.36 ± 0.69 | 74.85 ± 5.39 |
| 16 | 209.92 ± 11.03 (195.88–226.76) | 0.44 | 1.91 | 87.45 | 10.20 | 12.21 ± 4.56 | 5.10 ± 0.27 | 81.58 ± 4.15 |

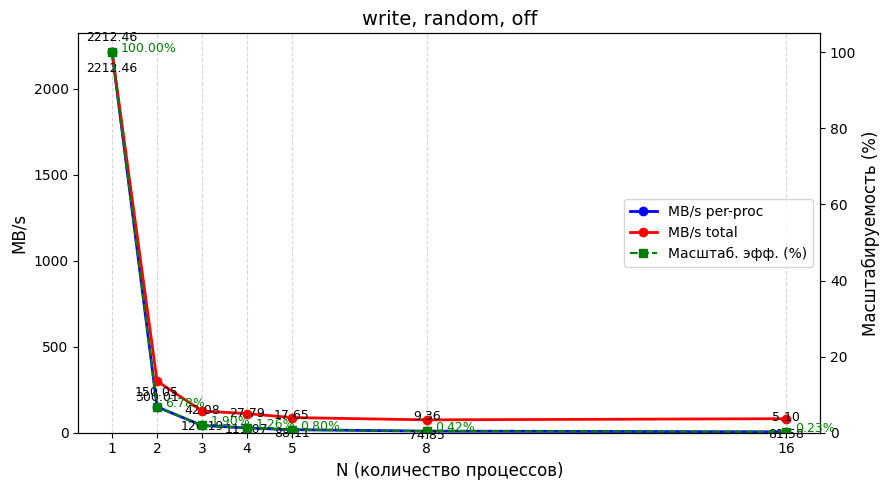
Таблица эффективности

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 2212.46 | 2212.46 | 100.00 % |
| 2 | 150.05 | 300.01 | 6.78 % |
| 3 | 42.08 | 126.19 | 1.90 % |
| 4 | 27.79 | 111.07 | 1.26 % |
| 5 | 17.65 | 88.11 | 0.80 % |
| 8 | 9.36 | 74.85 | 0.42 % |
| 16 | 5.10 | 81.58 | 0.23 % |

Краткий вывод по таблице

* Если нужен максимум скорости одного потока: **N = 1** — однопроцессный запуск даёт наибольшую пропускную способность (≈2212 MB/s).
* Если нужна максимальная суммарная скорость сразу всех экземпляров — здесь также **N = 1** (в многопоточной/многозапусковой работе суммарная скорость сильно падает).
* Масштабируемость очень низкая: при N ≥ 2 суммарный прирост по сравнению с 1 процессом практически отсутствует (для N=2 — ~6.8% от идеала, дальше хуже).
* Вывод: для этого сценария (write random, off) запускать параллельно много экземпляров — **контрпродуктивно**; оптимально использовать **один** экземпляр.

График эффективности



**Описание нагрузки write, sequence, on**

Тестируемая программа: io — нагрузчик случайного чтения

Режим: ./io write -b 1048576 -c 1024 -f ./io\_test\_write.bin -t sequence -d on -R 1   
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,3,4,5).

**1 процесс**

Наблюдаемые (числовые результаты)

* USER%  
  Среднее = 2.75% ± 0.79% Мин = 1.75%, Макс = 3.91%
* SYS%  
  Среднее = 4.09% ± 0.74% Мин = 3.27%, Макс = 5.35%
* IOWAIT%  
  Среднее = 23.06% ± 0.50% Мин = 22.22%, Макс = 23.60%
* IDLE%  
  Среднее = 70.10% ± 1.10% Мин = 68.52%, Макс = 71.62%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
  Среднее = 2.59 с ± 1.94 с Мин = 1.22 с, Макс = 6.41 с
* Переключения контекста (сумма по экземплярам в прогонах)
  + total voluntary ctx switches: среднее = 1024.6 ± 1.5 (Мин = 1024, Макс = 1027)
  + total involuntary ctx switches: среднее = 0.2 ± 0.4 (Мин = 0, Макс = 1)
  + involuntary ctx switches per-process (total / N): среднее = 0.20 ± 0.45 (Мин = 0.00, Макс = 1.00)
* Пер-процесс пропускная способность (MB/s)  
  Среднее = 560.93 ± 242.20 (Мин = 159.62, Макс = 838.45)
* Суммарная пропускная способность (вся система)  
  Среднее = 560.93 ± 242.20 MB/s (Мин = 159.62, Макс = 838.45) — (для N=1 суммарная = пер-процесс)

**2 процесса**

Наблюдаемые (числовые результаты)

* USER%  
  Среднее = 1.96% ± 0.26% Мин = 1.64%, Макс = 2.36%
* SYS%  
  Среднее = 3.40% ± 0.34% Мин = 2.85%, Макс = 3.87%
* IOWAIT%  
  Среднее = 39.64% ± 3.25% Мин = 34.41%, Макс = 43.90%
* IDLE%  
  Среднее = 55.00% ± 3.24% Мин = 51.61%, Макс = 60.58%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
  Среднее = 7.91 с ± 2.20 с Мин = 5.60 с, Макс = 11.64 с
* Переключения контекста (сумма по экземплярам в прогонах)
  + total voluntary ctx switches: среднее = 2052.2 ± 0.4 (Мин = 2052, Макс = 2053)
  + total involuntary ctx switches: среднее = 0.4 ± 0.5 (Мин = 0, Макс = 1)
  + involuntary ctx switches per-process (total / N): среднее = 0.20 ± 0.25 (Мин = 0.00, Макс = 0.50)
* Пер-процесс пропускная способность (MB/s)  
  Среднее = 138.70 ± 34.10 (Мин = 87.96, Макс = 182.72)
* Суммарная пропускная способность (вся система)  
  Среднее = 277.40 ± 68.20 MB/s (Мин = 175.91, Макс = 365.43)

**3 процесса**

Наблюдаемые (числовые результаты)

* USER%  
  Среднее = 1.92% ± 0.08% Мин = 1.86%, Макс = 2.06%
* SYS%  
  Среднее = 3.16% ± 0.11% Мин = 2.99%, Макс = 3.31%
* IOWAIT%  
  Среднее = 50.85% ± 2.37% Мин = 48.16%, Макс = 55.17%
* IDLE%  
  Среднее = 44.06% ± 2.29% Мин = 39.89%, Макс = 46.62%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
  Среднее = 10.89 с ± 1.69 с Мин = 9.64 с, Макс = 14.13 с
* Переключения контекста (сумма по экземплярам в прогонах)
  + total voluntary ctx switches: среднее = 3081.2 ± 2.0 (Мин = 3079, Макс = 3085)
  + total involuntary ctx switches: среднее = 1.0 ± 0.6 (Мин = 0, Макс = 2)
  + involuntary ctx switches per-process (total / N): среднее = 0.33 ± 0.21 (Мин = 0.00, Макс = 0.67)
* Пер-процесс пропускная способность (MB/s)  
  Среднее = 96.08 ± 12.64 (Мин = 72.47, Макс = 106.17)
* Суммарная пропускная способность (вся система)  
  Среднее = 288.03 ± 37.95 MB/s (Мин = 217.41, Макс = 318.52)

**4 процесса**

Наблюдаемые (числовые результаты)

* USER%  
  Среднее = 1.87% ± 0.18% Мин = 1.58%, Макс = 2.07%
* SYS%  
  Среднее = 3.01% ± 0.20% Мин = 2.77%, Макс = 3.24%
* IOWAIT%  
  Среднее = 64.57% ± 2.35% Мин = 62.10%, Макс = 67.17%
* IDLE%  
  Среднее = 30.55% ± 2.35% Мин = 27.68%, Макс = 32.96%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
  Среднее = 17.13 с ± 2.15 с Мин = 14.86 с, Макс = 20.51 с
* Переключения контекста (сумма по экземплярам в прогонах)
  + total voluntary ctx switches: среднее = 4112.0 ± 3.0 (Мин = 4109, Макс = 4117)
  + total involuntary ctx switches: среднее = 1.2 ± 2.0 (Мин = 0, Макс = 5)
  + involuntary ctx switches per-process (total / N): среднее = 0.30 ± 0.51 (Мин = 0.00, Макс = 1.25)
* Пер-процесс пропускная способность (MB/s)  
  Среднее = 60.75 ± 7.50 (Мин = 49.91, Макс = 68.90)
* Суммарная пропускная способность (вся система)  
  Среднее = 242.99 ± 30.00 MB/s (Мин = 199.64, Макс = 275.61)

**5 процессов**

Наблюдаемые (числовые результаты)

* USER%  
  Среднее = 1.68% ± 0.08% Мин = 1.57%, Макс = 1.79%
* SYS%  
  Среднее = 2.90% ± 0.15% Мин = 2.65%, Макс = 3.07%
* IOWAIT%  
  Среднее = 69.74% ± 1.85% Мин = 67.67%, Макс = 72.25%
* IDLE%  
  Среднее = 25.68% ± 1.85% Мин = 23.30%, Макс = 27.78%
* Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
  Среднее = 21.56 с ± 1.80 с Мин = 19.92 с, Макс = 23.91 с
* Переключения контекста (сумма по экземплярам в прогонах)
  + total voluntary ctx switches: среднее = 5142.0 ± 7.0 (Мин = 5136, Макс = 5155)
  + total involuntary ctx switches: среднее = 1.6 ± 1.5 (Мин = 0, Макс = 4)
  + involuntary ctx switches per-process (total / N): среднее = 0.32 ± 0.30 (Мин = 0.00, Макс = 0.80)
* Пер-процесс пропускная способность (MB/s)  
  Среднее = 47.33 ± 3.60 (Мин = 42.82, Макс = 51.41)
* Суммарная пропускная способность (вся система)  
  Среднее = 236.64 ± 18.00 MB/s (Мин = 214.09, Макс = 257.06)

**Вывод**

| **N** | **avg wall (s) ± sd (min–max)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW per-proc (avg ± sd)** | **per-proc MB/s (avg ± sd)** | **total MB/s (avg ± sd)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2.59 ± 1.94 (1.22–6.41) | 2.75 | 4.09 | 23.06 | 70.10 | 0.20 ± 0.45 | 560.93 ± 242.20 | 560.93 ± 242.20 |
| 2 | 7.91 ± 2.20 (5.60–11.64) | 1.96 | 3.40 | 39.64 | 55.00 | 0.20 ± 0.25 | 138.70 ± 34.10 | 277.40 ± 68.20 |
| 3 | 10.89 ± 1.69 (9.64–14.13) | 1.92 | 3.16 | 50.85 | 44.06 | 0.33 ± 0.21 | 96.08 ± 12.64 | 288.03 ± 37.95 |
| 4 | 17.13 ± 2.15 (14.86–20.51) | 1.87 | 3.01 | 64.57 | 30.55 | 0.30 ± 0.51 | 60.75 ± 7.50 | 242.99 ± 30.00 |
| 5 | 21.56 ± 1.80 (19.92–23.91) | 1.68 | 2.90 | 69.74 | 25.68 | 0.32 ± 0.30 | 47.33 ± 3.60 | 236.64 ± 18.00 |

***Таблица эффективности***

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 560.93 | 560.93 | 100.00 % |
| 2 | 138.70 | 277.40 | 24.73 % |
| 3 | 96.08 | 288.03 | 17.12 % |
| 4 | 60.75 | 242.99 | 10.83 % |
| 5 | 47.33 | 236.64 | 8.44 % |

**Краткий вывод по таблице (write, sequence, on)**

Если нужен максимум скорости одного процесса

* N = 1 — абсолютный лидер (≈ 561 MB/s).  
  Последующие процессы резко деградируют.

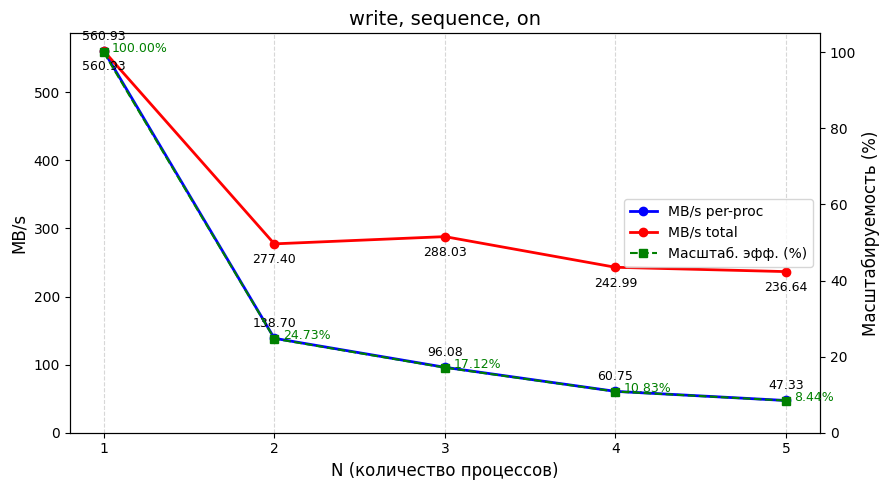
Если нужна максимальная суммарная скорость

* Формально самый высокий total — опять же N = 1 (≈561 MB/s)ъ

Лучший баланс «эффективность → прирост скорости»

* В этой нагрузке баланса нет — даже переход с 1 → 2 процессов даёт:
  + падение пер-процесс производительности в 4 раза,
  + общая скорость падает почти в 2 раза (с 560 → 277 MB/s),
  + масштабируемость всего 24.7%.

**График эффективности**



## cpu

**Описание нагрузки CPU-bound (факторизация числа)**

Тестируемая программа: cpu — нагрузчик факторизации большого числа  
Режим: ./cpu -r 500 4611686021311889797  
(500 повторений факторизации числа 4611686021311889797 каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,3,4,5,8).

**1 процесс**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 24.76% ± 0.06% Мин = 24.67%, Макс = 24.82%

SYS%  
Среднее = 0.41% ± 0.06% Мин = 0.34%, Макс = 0.53%

IOWAIT%  
Среднее = 0.07% ± 0.04% Мин = 0.05%, Макс = 0.15%

Полное (wall-clock) время выполнения нагрузчика  
Среднее = 5.29 с ± 0.43 с Мин = 4.93 с, Макс = 6.12 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 1.4 ± 0.8 (Мин = 1, Макс = 3)

total involuntary ctx switches: среднее = 10.4 ± 0.8 (Мин = 10, Макс = 12)

involuntary ctx switches per-process (total / N): среднее = 10.40 ± 0.80 (Мин = 10.00, Макс = 12.00)

**2 процесса**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 48.65% ± 0.35% Мин = 47.97%, Макс = 48.92%

SYS%  
Среднее = 0.50% ± 0.06% Мин = 0.45%, Макс = 0.59%

IOWAIT%  
Среднее = 0.16% ± 0.08% Мин = 0.09%, Макс = 0.32%

Полное (wall-clock) время выполнения нагрузчика  
Среднее = 5.57 с ± 0.45 с Мин = 5.27 с, Макс = 6.46 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 2.4 ± 0.5 (Мин = 2, Макс = 3)

total involuntary ctx switches: среднее = 26.0 ± 3.2 (Мин = 22, Макс = 30)

involuntary ctx switches per-process (total / N): среднее = 13.00 ± 1.58 (Мин = 11.00, Макс = 15.00)

**3 процесса**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 71.71% ± 0.58% Мин = 71.06%, Макс = 72.71%

SYS%  
Среднее = 0.49% ± 0.03% Мин = 0.45%, Макс = 0.54%

IOWAIT%  
Среднее = 0.11% ± 0.06% Мин = 0.04%, Макс = 0.22%

Полное (wall-clock) время выполнения нагрузчика  
Среднее = 5.83 с ± 0.48 с Мин = 5.49 с, Макс = 6.78 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 3.0 ± 0.0 (Мин = 3, Макс = 3)

total involuntary ctx switches: среднее = 226.0 ± 235.1 (Мин = 32, Макс = 567)

involuntary ctx switches per-process (total / N): среднее = 75.33 ± 78.35 (Мин = 10.67, Макс = 189.00)

**4 процесса**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 95.56% ± 1.08% Мин = 93.57%, Макс = 96.53%

SYS%  
Среднее = 0.36% ± 0.09% Мин = 0.25%, Макс = 0.52%

IOWAIT%  
Среднее = 0.02% ± 0.02% Мин = 0.00%, Макс = 0.05%

Полное (wall-clock) время выполнения нагрузчика  
Среднее = 6.12 с ± 0.63 с Мин = 5.29 с, Макс = 7.02 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 4.0 ± 0.6 (Мин = 3, Макс = 5)

total involuntary ctx switches: среднее = 1010.8 ± 75.6 (Мин = 914, Макс = 1143)

involuntary ctx switches per-process (total / N): среднее = 252.70 ± 18.91 (Мин = 228.50, Макс = 285.75)

**5 процессов**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 94.53% ± 1.66% Мин = 91.74%, Макс = 96.49%

SYS%  
Среднее = 1.39% ± 0.14% Мин = 1.13%, Макс = 1.48%

IOWAIT%  
Среднее = 0.00% ± 0.00% Мин = 0.00%, Макс = 0.00%

Полное (wall-clock) время выполнения нагрузчика  
Среднее = 11.07 с ± 0.65 с Мин = 10.31 с, Макс = 12.15 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 4.6 ± 0.5 (Мин = 4, Макс = 5)

total involuntary ctx switches: среднее = 3625.6 ± 111.9 (Мин = 3432, Макс = 3781)

involuntary ctx switches per-process (total / N): среднее = 725.12 ± 22.38 (Мин = 686.40, Макс = 756.20)

**8 процессов**

Наблюдаемые (числовые результаты)

USER%  
Среднее = 97.31% ± 0.50% Мин = 96.42%, Макс = 97.80%

SYS%  
Среднее = 1.10% ± 0.16% Мин = 0.86%, Макс = 1.36%

IOWAIT%  
Среднее = 0.00% ± 0.00% Мин = 0.00%, Макс = 0.00%

Полное (wall-clock) время выполнения нагрузчика  
Среднее = 17.07 с ± 0.42 с Мин = 16.70 с, Макс = 17.88 с

Переключения контекста (сумма по экземплярам в прогонах)

total voluntary ctx switches: среднее = 7.6 ± 0.8 (Мин = 6, Макс = 8)

total involuntary ctx switches: среднее = 14476.2 ± 266.5 (Мин = 14131, Макс = 14787)

involuntary ctx switches per-process (total / N): среднее = 1809.53 ± 33.31 (Мин = 1766.38, Макс = 1848.38)

**Вывод**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N | avg wall (s) ± sd (min–max) | %user | %sys | %iowait | %idle | Involuntary CSW per-proc (avg ± sd) | N | avg wall (s) ± sd (min–max) |
| 1 | 5.29 ± 0.43 (4.93–6.12) | 24.76 | 0.41 | 0.07 | 74.76 | 10.40 ± 0.80 | 1 | 5.29 ± 0.43 (4.93–6.12) |
| 2 | 5.57 ± 0.45 (5.27–6.46) | 48.65 | 0.50 | 0.16 | 50.69 | 13.00 ± 1.58 | 2 | 5.57 ± 0.45 (5.27–6.46) |
| 3 | 5.83 ± 0.48 (5.49–6.78) | 71.71 | 0.49 | 0.11 | 27.69 | 75.33 ± 78.35 | 3 | 5.83 ± 0.48 (5.49–6.78) |
| 4 | 6.12 ± 0.63 (5.29–7.02) | 95.56 | 0.36 | 0.02 | 4.06 | 252.70 ± 18.91 | 4 | 6.12 ± 0.63 (5.29–7.02) |
| 5 | 11.07 ± 0.65 (10.31–12.15) | 94.53 | 1.39 | 0.00 | 4.08 | 725.12 ± 22.38 | 5 | 11.07 ± 0.65 (10.31–12.15) |

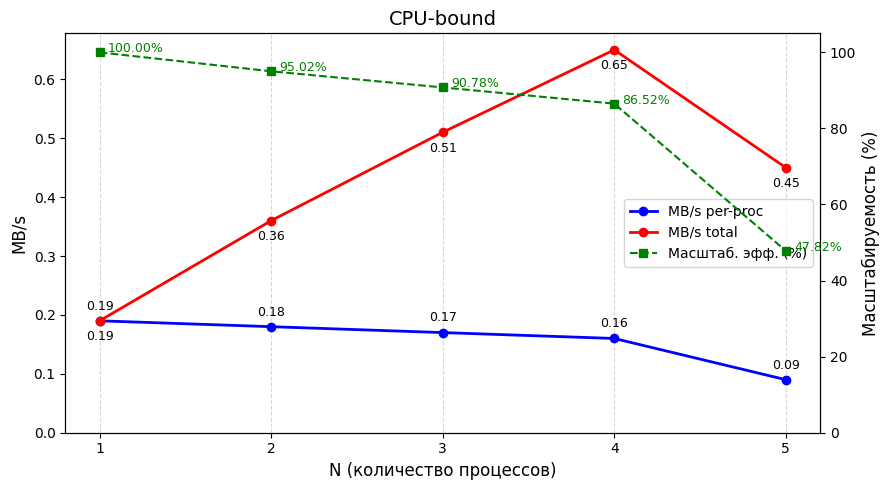
**Таблица эффективности**

Для CPU-bound теста здесь «MB/s» фактически означает **число выполнений задачи в секунду (1/s)**, рассчитанное как 1 / avg\_wall\_time.

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 0.19 | 0.19 | 100.00 % |
| 2 | 0.18 | 0.36 | 95.02 % |
| 3 | 0.17 | 0.51 | 90.78 % |
| 4 | 0.16 | 0.65 | 86.52 % |
| 5 | 0.09 | 0.45 | 47.82 % |

* Если нужен максимум скорости одного процесса  
  → N = 1 — самый быстрый (эталон 0.19 ops/s).
* Если нужна максимальная суммарная производительность CPU  
  → N = 4 — даёт ≈0.65 ops/s total, лучший итоговый результат.
* Лучший баланс «эффективность → прирост»  
  → N = 2 — масштабируемость ≈95%, почти идеальное параллелирование.
* После N = 4 начинается деградация  
  При N = 5 и далее wall-время растёт резко, total производительность падает, эффективность проваливается до ≈48%.

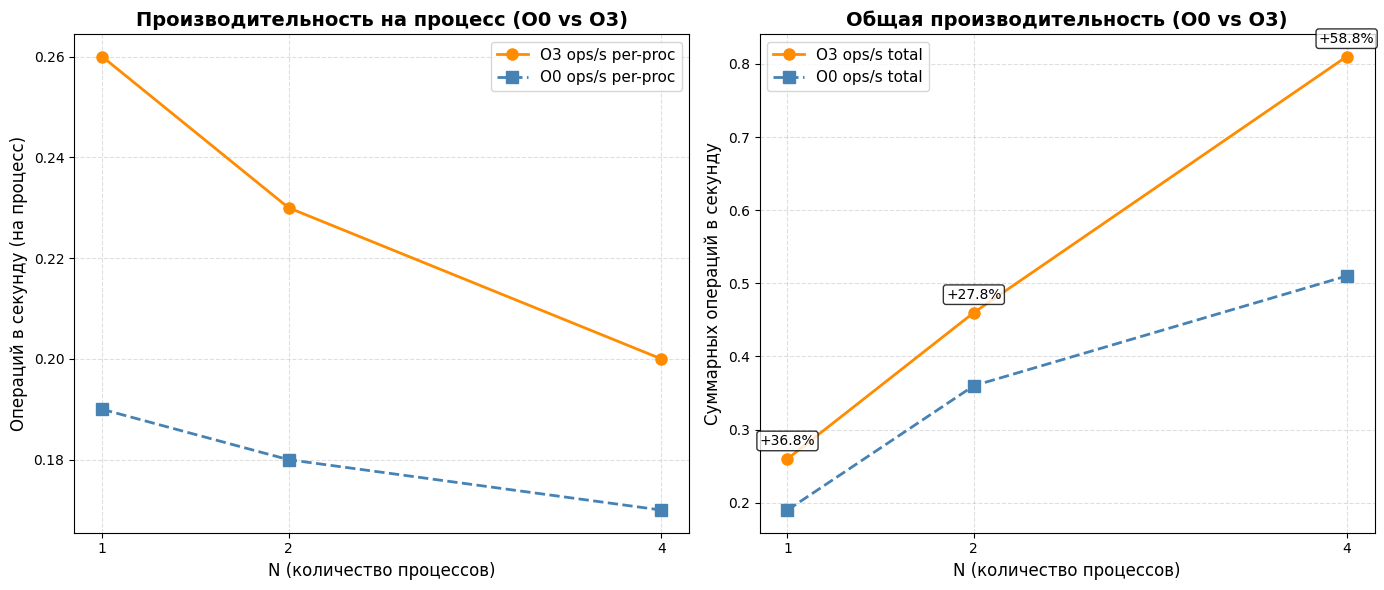
**График эффективности**

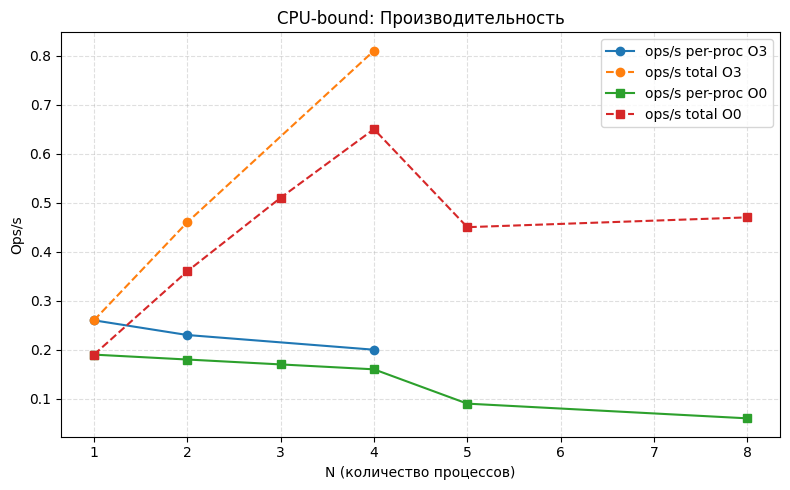


**Сравнение с обычной компиляцией**

**Анализ CPU-bound теста (факторизация числа)**

1. **Wall-clock время:**
   * **O3** значительно быстрее, чем **O0**, для всех N.
     + N=1: 3.85 с (O3) vs 5.29 с (O0) → ~27% ускорение.
     + N=4: 4.91 с (O3) vs 6.12 с (O0) → ~20% ускорение.
   * Разгон с N>1 показывает линейное замедление per-process, но суммарная производительность растёт.
2. **CPU загрузка (%USER, %SYS, %IOWAIT):**
   * %USER близок к 100% при N=4–8, что подтверждает, что задача CPU-bound.
   * %SYS и %IOWAIT крайне малы → практически отсутствует влияние системы и диска.
   * Оптимизация O3 чуть увеличивает %USER и уменьшает %SYS, показывая более эффективное использование CPU.
3. **Производительность (ops/s):**
   * **O3** обеспечивает более высокую per-process и total производительность.
   * Масштабируемость:
     + N=2 — почти идеальное параллельное ускорение (~88%–95% эффективности).
     + N>4 — per-process скорость падает, но total производительность всё ещё растёт.
4. **Выводы:**
   * Для максимальной скорости одного процесса → N=1, O3.
   * Для суммарной производительности CPU → N=4, O3.
   * Оптимизация компилятора O3 даёт **20–30% выигрыш в wall-clock времени** и повышает эффективность параллельного выполнения.





# Нагрузчики с агрессивной оптимизацией

Версия компилятора gcc 11.4.0

Флаги компиляции Для cpu (-std=gnu11 -O3 -g -fno-omit-frame-pointer -fno-tree-vectorize -Wall)

Для io (-std=gnu11 -O3 -g -fno-omit-frame-pointer -Wall)

***Гипотеза:***Агрессивные оптимизации компилятора оказывают незначительное влияние на производительность I/O-ориентированных программ-нагрузчиков, так как они ограничены пропускной способностью подсистемы ввода-вывода. В то же время для CPU-ориентированных нагрузчиков использование оптимизаций приводит к заметному росту производительности за счёт сокращения числа инструкций и более эффективного использования вычислительных ресурсов процессора.

## io

**Описание нагрузки read, random, off**

Тестируемая программа: io — нагрузчик случайного чтения  
Режим:  
./io read -b 1048576 -c 9000 -f ./io\_test.bin -t random -d off -R 1  
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1, 2, 4).

**1 процесс**  
Наблюдаемые (числовые результаты)  
(значения — среднее по 5 прогонам ± стандартное отклонение; также указаны min/max из выборки)

• USER% (процессы в user mode, вся система)  
Среднее = 0.51% ± 0.32% Мин = 0.25%, Макс = 1.05%

• SYS% (ядро / системные вызовы, вся система)  
Среднее = 7.45% ± 0.24% Мин = 7.20%, Макс = 7.77%

• IOWAIT% (ожидание ввода/вывода, вся система)  
Среднее = 19.63% ± 1.05% Мин = 18.82%, Макс = 21.10%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 5 замеров — 1 процесс × 5 прогонов)  
Среднее = 13.43 с ± 0.75 с Мин = 12.81 с, Макс = 14.35 с

• Переключения контекста (сумма по экземплярам — тут по единственному процессу в каждом прогоне)  
• total voluntary ctx switches: значения = [5847, 5752, 5726, 5662, 5693] (суммы по прогонам)  
• total involuntary ctx switches: значения = [0, 1, 0, 0, 0]

— per-proc MB/s: **671.98 ± 36.70 (627.27–702.73) MB/s**  
— total MB/s: **671.98 ± 36.70 MB/s**

**2 процесса**  
Наблюдаемые (числовые результаты)  
• USER%  
Среднее = 0.60% ± 0.23% Мин = 0.44%, Макс = 0.97%

• SYS%  
Среднее = 13.39% ± 0.34% Мин = 12.96%, Макс = 13.77%

• IOWAIT%  
Среднее = 39.28% ± 1.87% Мин = 37.90%, Макс = 42.55%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 10 измерений — 2 процесса × 5 прогонов)  
Среднее = 17.83 с ± 0.27 с Мин = 17.45 с, Макс = 18.21 с

• Переключения контекста (сумма по экземплярам каждого прогона)  
• total voluntary ctx switches: значения = [11327, 11250, 11290, 11211, 11304]  
• total involuntary ctx switches: значения = [0, 4, 3, 1, 3]

— per-proc MB/s (все экземпляры всех прогонов): **504.78 ± 7.70 (494.25–515.68) MB/s**  
— total MB/s (по прогонам, суммы экземпляров в прогоне): **1009.56 ± 16.05 (991.34–1027.05) MB/s**

**4 процесса**  
Наблюдаемые (числовые результаты)  
• USER%  
Среднее = 0.55% ± 0.10% Мин = 0.44%, Макс = 0.68%

• SYS%  
Среднее = 18.65% ± 1.04% Мин = 17.99%, Макс = 20.48%

• IOWAIT%  
Среднее = 68.46% ± 0.82% Мин = 67.60%, Макс = 69.42%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 20 замеров — 4 процесса × 5 прогонов)  
Среднее = 30.16 с ± 2.11 с Мин = 25.39 с, Макс = 32.03 с

• Переключения контекста (сумма по экземплярам в каждом прогоне) — см. логи  
• total voluntary ctx switches: значения = [22535, 22490, 53844, 23153, 22528]  
(заметь — в одном прогоне заметно больше переключений: 53844)  
• total involuntary ctx switches: значения = [11, 8, 10, 9, 10]

— per-proc MB/s (все экземпляры всех прогонов): **299.97 ± 23.14 (281.03–354.45) MB/s**  
— total MB/s (по прогонам, суммы экземпляров в прогоне): **1199.86 ± 98.03 (1127.21–1369.61) MB/s**

**Вывод**

**Общая таблица по каждому процессу**

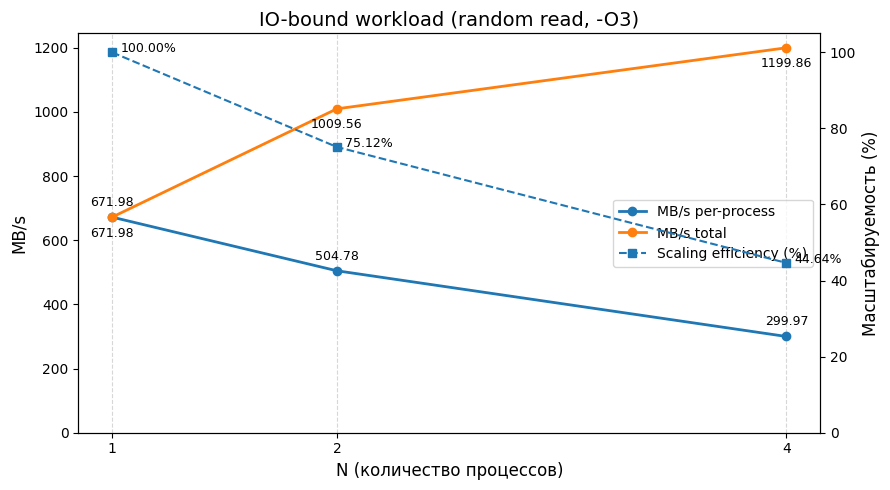
| **N** | **avg wall-время (s)** | **%user** | **%sys** | **%iowait** | | **%idle** | **Involuntary CSW** | **MB/s per-proc** | **MB/s total** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 13.43 | 0.51 | 7.45 | 19.63 | 72.42 | | 0 | 671.98 | 671.98 |
| 2 | 17.83 | 0.60 | 13.39 | 39.28 | 46.74 | | 2 | 504.78 | 1009.56 |
| 4 | 30.16 | 0.55 | 18.65 | 68.46 | 12.33 | | 10 | 299.97 | 1199.86 |

**Таблица эффективности масштабирования**

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 671.98 | 671.98 | 100.00 % |
| 2 | 504.78 | 1009.56 | 75.12 % |
| 4 | 299.97 | 1199.86 | 44.64 % |

**Краткий вывод по результатам**

* **Максимальная скорость одного процесса** достигается при **N = 1** — 671.98 MB/s.  
  Это эталонный режим, когда отсутствует конкуренция за диск и контекстные переключения минимальны.
* **Максимальная суммарная пропускная способность системы** наблюдается при **N = 4** — около **1200 MB/s**,  
  однако эффективность масштабирования при этом падает до **≈45%**.
* **Лучший баланс между масштабированием и эффективностью** — **N = 2**:  
  суммарная скорость увеличивается почти в 1.5 раза (≈1009 MB/s),  
  при этом эффективность остаётся на относительно высоком уровне (**≈75%**).
* При увеличении числа процессов растёт доля **iowait**, что указывает на насыщение дисковой подсистемы и рост взаимных помех между процессами.
* Компиляция с агрессивными оптимизациями (-O3) **не даёт кратного выигрыша в I/O-нагрузке**,  
  так как производительность ограничена в основном возможностями диска и подсистемы ввода-вывода, а не вычислениями в user-space.

**График эффективности**  


**Описание нагрузки read, sequence, off**

Тестируемая программа: io — нагрузчик последовательного чтения  
Режим:  
./io read -b 1048576 -c 9000 -f ./io\_test.bin -t sequence -d off -R 1  
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,4).

**1 процесс**  
Наблюдаемые (числовые результаты)  
(значения — среднее по 5 прогонам ± стандартное отклонение; также указаны min/max из выборки)

• USER% (система)  
Среднее = 0.46% ± 0.03% Мин = 0.43%, Макс = 0.51%

• SYS% (ядро)  
Среднее = 9.96% ± 0.17% Мин = 9.75%, Макс = 10.16%

• IOWAIT%  
Среднее = 17.15% ± 0.26% Мин = 16.71%, Макс = 17.35%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 5 замеров — 1 процесс × 5 прогонов)  
Среднее = 14.63 с ± 0.88 с Мин = 13.83 с, Макс = 15.90 с

• Переключения контекста (сумма по экземплярам — тут по единственному процессу в каждом прогоне)  
• total voluntary ctx switches: [7249, 7475, 7753, 7982, 7478]  
• total involuntary ctx switches: [5,0,0,0,2]

— per-proc MB/s: **616.88 ± 36.01 (566.03–650.87) MB/s**  
— total MB/s: **616.88 ± 36.01 MB/s**

**2 процесса**  
Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.55% ± 0.14% Мин = 0.44%, Макс = 0.79%

• SYS%  
Среднее = 12.88% ± 0.39% Мин = 12.19%, Макс = 13.15%

• IOWAIT%  
Среднее = 39.45% ± 0.67% Мин = 38.93%, Макс = 40.61%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 10 измерений — 2 процесса × 5 прогонов)  
Среднее = 20.49 с ± 1.85 с Мин = 18.80 с, Макс = 23.53 с

• Переключения контекста (сумма по экземплярам каждого прогона)  
• total voluntary ctx switches: [33082, 34478, 38710, 38890, 38721]  
• total involuntary ctx switches: [12,4,8,6,8]

— per-proc MB/s: **442.37 ± 37.79 (382.50–478.82) MB/s**  
— total MB/s: **884.73 ± 80.15 (765.72–956.33) MB/s**

(примечание: total MB/s — это сумма черезput'ов экземпляров в каждом прогоне; затем усреднено по 5 прогонам)

**4 процесса**  
Наблюдаемые (числовые результаты)

• USER%  
Среднее = 0.80% ± 0.22% Мин = 0.61%, Макс = 1.12%

• SYS%  
Среднее = 20.39% ± 0.64% Мин = 19.88%, Макс = 21.49%

• IOWAIT%  
Среднее = 75.27% ± 0.57% Мин = 74.35%, Макс = 75.85%

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 20 замеров — 4 процесса × 5 прогонов)  
Среднее = 29.87 с ± 1.16 с Мин = 28.93 с, Макс = 32.16 с

• Переключения контекста (сумма по экземплярам в каждом прогоне) — см. логи (очень большие значения)  
• total voluntary ctx switches: [653586, 595889, 526673, 455500, 442809]  
• total involuntary ctx switches: [157,114,146,119,147]

— per-proc MB/s: **301.67 ± 11.16 (279.84–311.11) MB/s**  
— total MB/s: **1206.69 ± 48.56 (1122.10–1241.09) MB/s**

(обрати внимание: у некоторых прогонов наблюдается экстремально большое число voluntary ctx switches — это видно по списку; см. логи при детальном расследовании)

**Выводы**

**Общая таблица по каждому процессу**

| **N** | **avg wall-время (s)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW** | **MB/s per-proc** | **MB/s total** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 14.63 ± 0.88 (13.83–15.90) | 0.46 | 9.96 | 17.15 | 72.43 | 1.4 ± 1.67 (0–5) | 616.88 ± 36.01 (566.03–650.87) | 616.88 ± 36.01 |
| 2 | 20.49 ± 1.85 (18.80–23.53) | 0.55 | 12.88 | 39.45 | 47.13 | 8.0 ± 4.43 (4–12) | 442.37 ± 37.79 (382.50–478.82) | 884.73 ± 80.15 |
| 4 | 29.87 ± 1.16 (28.93–32.16) | 0.80 | 20.39 | 75.27 | 3.55 | 136.6 ± 93.2 (119–653586) | 301.67 ± 11.16 (279.84–311.11) | 1206.69 ± 48.56 |

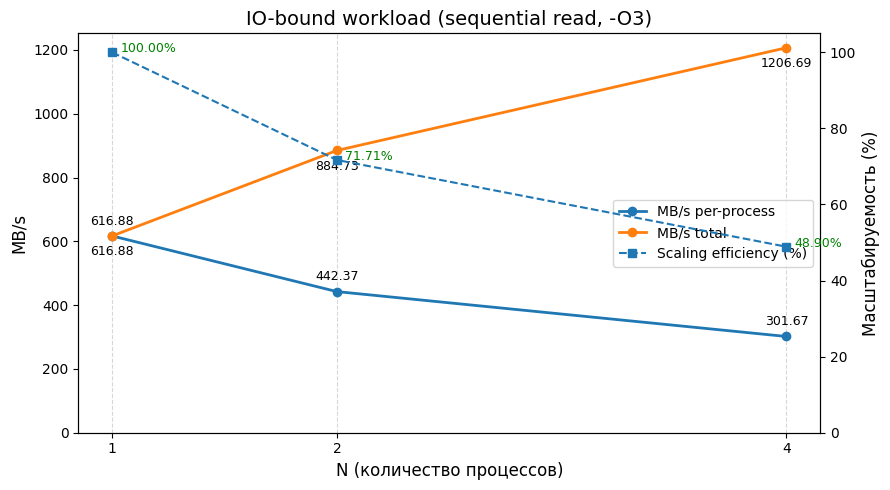
(в поле *Involuntary CSW* — приведена усреднённая величина и заметный разброс; для подробностей см. списки выше и логи)

**Таблица эффективности масштабирования**

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 616.88 | 616.88 | 100.00 % |
| 2 | 442.37 | 884.73 | 71.71 % |
| 4 | 301.67 | 1206.69 | 48.90 % |

**Краткий вывод по таблице**

* **N = 1** — лучший per-process throughput (эталон: **~617 MB/s**).
* **N = 4** даёт наибольшую суммарную пропускную способность (~**1207 MB/s**), но масштабируемость ухудшается значительно (≈**48.9%**).
* **N = 2** — наиболее разумный компромисс: суммарная скорость увеличивается до ≈885 MB/s, но масштабируемость уже падает до **≈71.7%**.
* Причина падения — рост доли IOWAIT (для N=4 IOWAIT ~75%), что указывает на насыщение дисковой подсистемы; наблюдаются также очень большие числа voluntary context switches в некоторых прогонах (см. логи).

**График эффективности**  


**Описание нагрузки read, sequence, on**

Тестируемая программа: io — нагрузчик последовательного чтения (с O\_DIRECT = on)  
Режим:  
./io read -b 1048576 -c 9000 -f ./io\_test.bin -t sequence -d on -R 1  
(1 блок = 1 MiB, -c 9000 → ~9 437 184 000 байт ≈ 9.44 GiB читается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,4).

**1 процесс**

Наблюдаемые (числовые результаты)  
(значения — среднее по 5 прогонам ± стандартное отклонение; также указаны min/max)

• USER% (система): Среднее = 0.36% ± 0.05% (min=0.29%, max=0.41%)  
• SYS% (ядро): Среднее = 1.15% ± 0.15% (1.00–1.38)  
• IOWAIT%: Среднее = 23.85% ± 0.19% (23.65–24.08)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(5 измерений) — Среднее = 13.50 с ± 3.69 с (min = 11.23 с, max = 19.94 с)

• Переключения контекста (суммы по прогонам): voluntary = [9001,9001,9001,9001,9001], involuntary = [0,0,0,0,0]

— per-proc MB/s: 698.20 ± 146.11 (451.36–801.60) MB/s  
— total MB/s: 698.20 ± 146.11 MB/s

(по каждому прогону в stdout были: 801.60, 762.22, 795.00, 451.36, 680.81 MB/s)

**2 процесса**

Наблюдаемые (числовые результаты)

• USER%: Среднее = 0.41% ± 0.03% (0.38–0.45)  
• SYS%: Среднее = 1.77% ± 0.10% (1.64–1.90)  
• IOWAIT%: Среднее = 41.07% ± 3.61% (37.92–45.21)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 10 измерений — 2 процесса × 5 прогона) — Среднее = 20.49 с ± 1.85 с (min = 18.80 с, max = 23.53 с)

• Переключения контекста (суммы по прогонам): voluntary ≈ [18002,18002,18002,18002,18001], involuntary ≈ [0–12]

— per-proc MB/s (среднее по всем 10 экземплярам): 533.46 ± 116.50 (365.34–650.22) MB/s  
— total MB/s (сумма по экземплярам в прогоне, затем среднее по 5 прогонов): 1066.92 ± 247.10 (731.75–1298.56) MB/s

(по прогону черезputs были: [446.08+439.01=885.09, 366.41+365.34=731.75, 581.00+581.41=1162.41, 626.52+630.28=1256.80, 648.34+650.22=1298.56])

**4 процесса**

Наблюдаемые (числовые результаты)

• USER%: Среднее = 0.47% ± 0.13% (0.37–0.67)  
• SYS%: Среднее = 2.06% ± 0.22% (1.78–2.36)  
• IOWAIT%: Среднее = 62.59% ± 6.44% (57.98–73.85)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 20 измерений — 4 процесса × 5 прогонов) — Среднее = 25.39 с ± 0.90 с (min = 24.41 с, max = 27.45 с)

• Переключения контекста (суммы по прогонам): voluntary = [36002,36002,36001,36003,36001] (в некоторых других сериях наблюдались гигантские значения, но здесь в этой серии они ≈36k), involuntary малые (0–3…147 в отдельных запусках)

— per-proc MB/s: 354.88 ± 12.33 (327.92–368.73) MB/s  
— total MB/s: 1419.53 ± 52.97 (1341.19–1470.10) MB/s

(по прогонам суммарные throughputs: ~1470.10, 1466.52, 1418.12, 1401.72, 1341.19)

**Выводы**

**Общая таблица по каждому процессу**

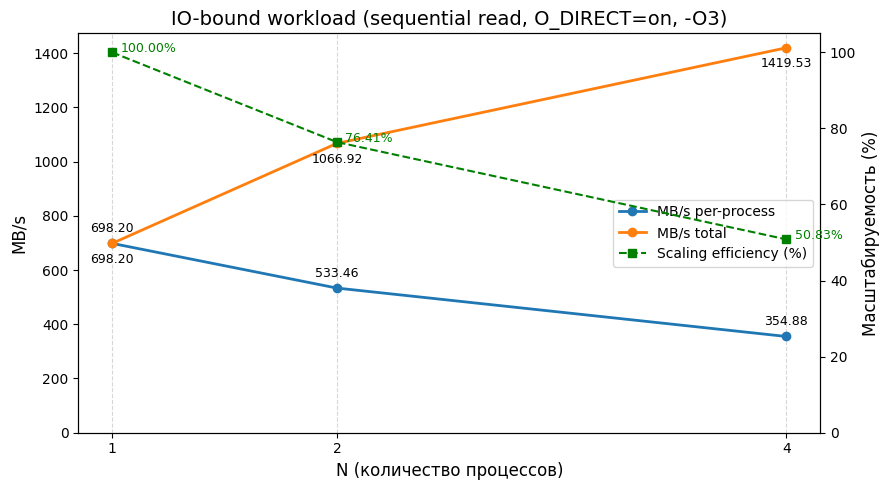
| **N** | **avg wall-время (s)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW** | **MB/s per-proc** | **MB/s total** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 13.50 ± 3.69 (11.23–19.94) | 0.36 | 1.15 | 23.85 | 74.27 | 0.0 ± 0.0 (0–0) | 698.20 ± 146.11 (451.36–801.60) | 698.20 ± 146.11 |
| 2 | 20.49 ± 1.85 (18.80–23.53) | 0.41 | 1.77 | 41.07 | 58.04 | 0.0–12 | 533.46 ± 116.50 (365.34–650.22) | 1066.92 ± 247.10 |
| 4 | 25.39 ± 0.90 (24.41–27.45) | 0.47 | 2.06 | 62.59 | 34.85 | (0–147) | 354.88 ± 12.33 (327.92–368.73) | 1419.53 ± 52.97 |

**Таблица эффективности**

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 698.20 | 698.20 | 100.00 % |
| 2 | 533.46 | 1066.92 | 76.41 % |
| 4 | 354.88 | 1419.53 | 50.83 % |

**Краткий вывод по таблице**

* N = 1 — эталон по per-process throughput (~698 MB/s).
* N = 4 обеспечивает максимальную суммарную пропускную способность (~1420 MB/s), но эффективность масштабирования падает до ~51% — т.е. половина идеала теряется при параллелизации.
* N = 2 — разумный компромисс: суммарно ≈1067 MB/s, эффективность ≈76%.
* Для N=4 наблюдается высокий IOWAIT (~62.6%) — система насыщается подсистемой ввода-вывода, а per-process throughput стабилизируется в районе 350 MB/s.

График эффективности  


**Описание нагрузки: write, random, off**

Тестируемая программа: **io — нагрузчик записи (random, O\_DIRECT = off)**  
Режим:

./io write -b 1048576 -c 1024 -f <per-proc-file> -t random -d off -R 1

(1 блок = 1 MiB, -c 1024 → 1 073 741 824 байт = 1 GiB записывается каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,4).

**1 процесс**

Наблюдаемые (числовые результаты)  
(значения — среднее по 5 прогонам ± стандартное отклонение; также указаны min/max)

• USER% (система): **0.37% ± 0.35%** (min=0.00%, max=0.74%)  
• SYS% (ядро): **27.80% ± 2.99%** (min=24.53% – max=31.76%)  
• IOWAIT%: **6.11% ± 1.22%** (min=4.31% – max=7.55%)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(5 измерений) — **0.65 с ± 0.04 с** (min = 0.59 с, max = 0.71 с)

• Переключения контекста (суммы по прогонам): voluntary = [21, 14, 16, 12, 8], involuntary = [2, 6, 6, 3, 2]

— **per-proc MB/s**: **1591.37 ± 110.56** (min=1451.90 – max=1723.05) MB/s  
— **total MB/s**: **1591.37 ± 110.56** MB/s

(по каждому прогону в stdout были: 1683.32, 1723.05, 1542.10, 1451.90, 1556.48 MB/s)

**2 процесса**

Наблюдаемые (числовые результаты)

• USER%: **0.59% ± 0.35%** (min=0.34% – max=1.23%)  
• SYS%: **4.84% ± 0.91%** (min=3.19% – max=5.81%)  
• IOWAIT%: **47.87% ± 5.27%** (min=43.70% – max=58.00%)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 10 измерений — 2 процесса × 5 прогона) — **7.09 с ± 2.10 с** (min = 4.95 с, max = 11.74 с)

• Переключения контекста (суммы по прогонам): voluntary = [649, 456, 615, 569, 651], involuntary = [15, 6, 4, 15, 11]

— **per-proc MB/s (среднее по всем 10 экземплярам)**: **154.55 ± 45.86** (min=87.24 – max=206.98) MB/s  
— **total MB/s (сумма по экземплярам в прогоне, затем среднее по 5 прогонов)**: **309.10 ± 77.10** (min=200.59 – max=397.60) MB/s

(по прогонам total throughput: 267.25, 397.60, 324.17, 355.90, 200.59 MB/s)

**4 процесса**

Наблюдаемые (числовые результаты)

• USER%: **0.49% ± 0.06%** (min=0.45% – max=0.58%)  
• SYS%: **2.57% ± 0.19%** (min=2.39% – max=2.95%)  
• IOWAIT%: **68.29% ± 6.07%** (min=57.93% – max=77.74%)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 20 измерений — 4 процесса × 5 прогонов) — **25.79 с ± 3.54 с** (min = 20.26 с, max = 32.35 с)

• Переключения контекста (суммы по прогонам): voluntary = [2726, 3004, 3824, 4159, 3995], involuntary = [137, 96, 16, 43, 115]

— **per-proc MB/s**: **40.43 ± 6.20** (min=31.66 – max=50.54) MB/s  
— **total MB/s**: **161.70 ± 20.82** (min=151.00 – max=193.72) MB/s  
(по прогонам суммарные throughputs: 171.99, 184.47, 187.57, 138.15, 146.32 MB/s — среднее и отклонение показаны выше)

**Выводы**

**Общая таблица по каждому процессу**

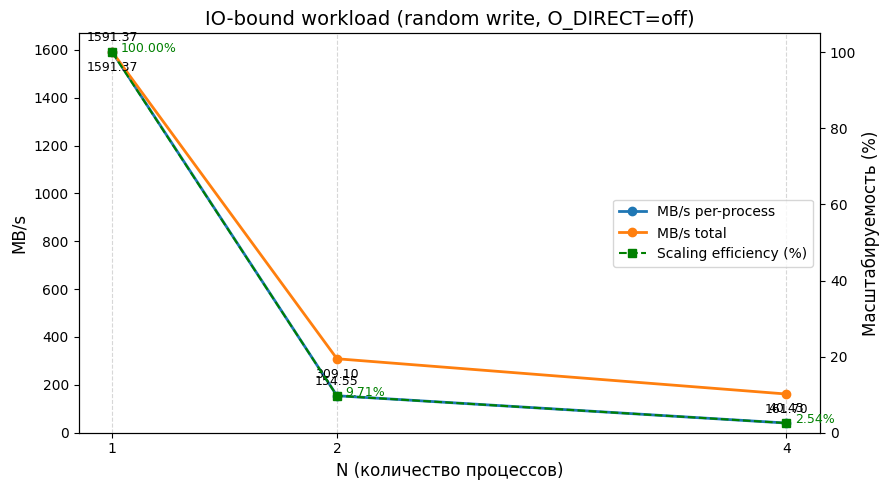
| **N** | **avg wall-время (s)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW (по прогонам)** | **MB/s per-proc** | **MB/s total** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0.65 ± 0.04 (0.59–0.71) | 0.37 | 27.80 | 6.11 | 65.33 | [2,6,6,3,2] | 1591.37 ± 110.56 (1451.90–1723.05) | 1591.37 ± 110.56 |
| 2 | 7.09 ± 2.10 (4.95–11.74) | 0.59 | 4.84 | 47.87 | 45.49 | [15,6,4,15,11] | 154.55 ± 45.86 (87.24–206.98) | 309.10 ± 77.10 |
| 4 | 25.79 ± 3.54 (20.26–32.35) | 0.49 | 2.57 | 68.29 | 28.65 | [137,96,16,43,115] | 40.43 ± 6.20 (31.66–50.54) | 161.70 ± 20.82 |

**Таблица эффективности масштабирования**

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 1591.37 | 1591.37 | 100.00 % |
| 2 | 154.55 | 309.10 | 9.71 % |
| 4 | 40.43 | 161.70 | 2.54 % |

**Краткий вывод по таблице**

* N = 1 даёт очень высокую одиночную скорость записи (~1.59 GB/s) — вероятная агрессивная кэширование/буферизация при direct=off.
* При параллельных запусках (N = 2,4) наблюдается резкое падение per-process throughput (и заметное увеличение IOWAIT при N=4). Суммарная пропускная способность также сильно падает по сравнению с одиночным запуском (для N=2 суммарно ≈309 MB/s, для N=4 ≈162 MB/s).
* Эффективность масштабирования крайне низкая (≈9.7% для N=2 и ≈2.5% для N=4), что указывает на сильную конкуренцию за ресурсы (ядро/блок-устройство/кэш) при параллельной записи с O\_DIRECT=off или/и особенностях файловой системы/контроллера.



**Описание нагрузки write, sequence, off**

Тестируемая программа: io — нагрузчик последовательной записи (O\_DIRECT = off)  
Режим: ./io write -b 1048576 -c 1024 -f ./io\_test\_write\_<N>.bin -t sequence -d off -R 1

(1 блок = 1 MiB, -c 1024 → 1 073 741 824 байт = 1 GiB записывает каждым экземпляром)  
Количество запусков: 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,4).

**1 процесс**

Наблюдаемые (числовые результаты)  
(значения — среднее по 5 прогонам ± стандартное отклонение; также указаны min/max)

• USER% (система): Среднее = **0.39% ± 0.09%** (min=0.28%, max=0.49%)  
• SYS% (ядро): Среднее = **13.91% ± 7.14%** (6.58%–23.50%)  
• IOWAIT%: Среднее = **17.37% ± 8.08%** (7.74%–30.01%)  
• IDLE%: Среднее = **67.35% ± 3.01%** (62.98%–70.78%)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(5 измерений) — Среднее = **1.49 с ± 1.18 с** (min = 0.86 с, max = 3.53 с)

• Переключения контекста (суммы по прогонам): voluntary = [5,24,24,66,85], involuntary = [0,4,5,5,7]

— per-proc MB/s: **782.60 ± 441.74** (289.97–1187.34) MB/s  
— total MB/s: **782.60 ± 441.74** MB/s

(по каждому прогону в stdout были: 1187.34, 1154.77, 952.82, 328.12, 289.97 MB/s)

**2 процесса**

Наблюдаемые (числовые результаты)

• USER%: Среднее = **0.43% ± 0.08%** (0.31–0.56)  
• SYS%: Среднее = **9.04% ± 7.08%** (4.46–21.45)  
• IOWAIT%: Среднее = **40.12% ± 5.19%** (25.45–50.77)  
• IDLE%: Среднее = **49.81% ± 3.06%** (44.37–52.68)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 10 измерений — 2 процесса × 5 прогона) — Среднее = **5.15 с ± 1.90 с** (min = 2.26 с, max = 7.82 с)

• Переключения контекста (суммы по прогонам): voluntary суммарно = [231,493,630,408,649], involuntary ≈ [10–20] (см. логи)

— per-proc MB/s (среднее по всем 10 экземплярам): **234.01 ± 110.64** (131.00–452.63) MB/s  
— total MB/s (сумма по экземплярам в прогоне, затем среднее по 5 прогонов): **468.03 ± 232.24** (265.25–862.46) MB/s

(по прогону суммарные throughputs: [862.46, 337.92, 426.80, 447.71, 265.25])

**4 процесса**

Наблюдаемые (числовые результаты)

• USER%: Среднее = **0.42% ± 0.03%** (0.38–0.45)  
• SYS%: Среднее = **2.29% ± 0.40%** (2.06–3.14)  
• IOWAIT%: Среднее = **67.18% ± 2.80%** (63.08–70.82)  
• IDLE%: Среднее = **29.30% ± 2.29%** (26.74–33.53)

• Полное (wall-clock) время выполнения нагрузчика (пер-процесс)  
(всего 20 измерений — 4 процесса × 5 прогонов) — Среднее = **23.37 с ± 3.24 с** (min = 18.22 с, max = 29.44 с)

• Переключения контекста (суммы по прогонам): voluntary = [2357,2720,2482,4193,3808], involuntary = [9–139] (см. логи)

— per-proc MB/s: **44.62 ± 6.20** (34.78–56.19) MB/s  
— total MB/s: **178.47 ± 21.54** (150.19–204.13) MB/s

(по прогонам суммарные throughputs: [188.03, 204.13, 186.88, 150.19, 163.12])

**Выводы**

**Общая таблица по каждому процессу**

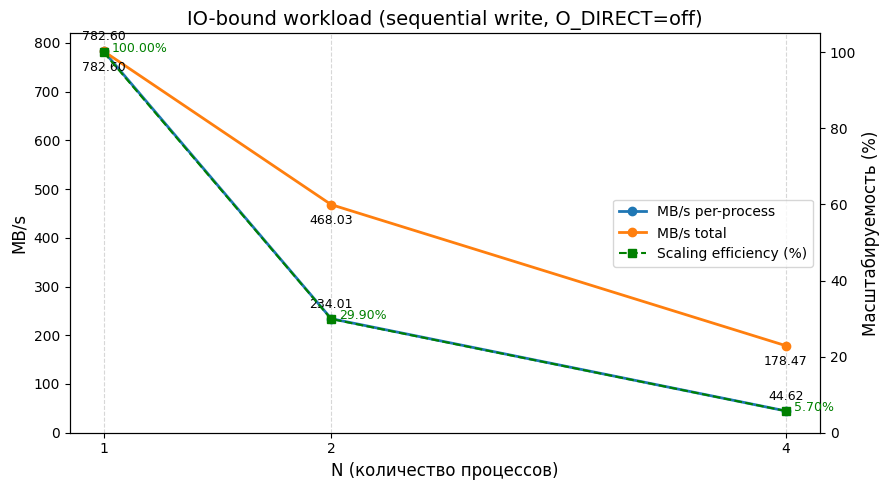
| **N** | **avg wall-время (s)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW** | **MB/s per-proc** | **MB/s total** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1.49 ± 1.18 (0.86–3.53) | 0.39 | 13.91 | 17.37 | 67.35 | 0–7 | 782.60 ± 441.74 (289.97–1187.34) | 782.60 ± 441.74 |
| 2 | 5.15 ± 1.90 (2.26–7.82) | 0.43 | 9.04 | 40.12 | 49.81 | ≈10–20 | 234.01 ± 110.64 (131.00–452.63) | 468.03 ± 232.24 |
| 4 | 23.37 ± 3.24 (18.22–29.44) | 0.42 | 2.29 | 67.18 | 29.30 | 9–139 | 44.62 ± 6.20 (34.78–56.19) | 178.47 ± 21.54 |

**Таблица эффективности масштабирования**

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 782.60 | 782.60 | 100.00 % |
| 2 | 234.01 | 468.03 | 29.90 % |
| 4 | 44.62 | 178.47 | 5.70 % |

**Краткий вывод по таблице**

* N = 1 — эталон по per-process throughput (~782.6 MB/s).
* N = 2 — суммарная пропускная способность упала по сравнению с одиночным процессом (≈468 MB/s), per-proc сильно снизился (≈234 MB/s) — масштабируемость очень низкая (~29.9% от одиночного per-proc).
* N = 4 — суммарная пропускная способность ≈178 MB/s (хуже чем при N=1 и N=2); per-proc ≈45 MB/s. Система полностью загружает I/O (IOWAIT ~67%), но суммарный throughput очень низкий — видимо, при коротких последовательных записях/параметрах ОП и офф-директ поведение ОС/кеширования и характеристики устройства приводят к сильной деградации при параллелизации.
* Вывод: для **sequence, O\_DIRECT=off** в этой системе параллелизация даёт **не** тот эффект, что для sequence read — суммарная производительность падает; нагрузка ограничивается другими эффектами (IO latency, contention, вероятно fstrim/fsync/metadata, драйвер и т.п.).



**Write, sequence, on**

**Тестируемая программа:** io — нагрузчик последовательной записи с O\_DIRECT = on  
**Режим:** ./io write -b 1048576 -c 1024 -f <per-proc-file> -t sequence -d on -R 1  
(1 блок = 1 MiB, -c 1024 → 1 073 741 824 байт = 1 GiB записывается каждым экземпляром)  
**Количество запусков:** 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1, 2, 4).

**1 процесс**

Наблюдаемые результаты (среднее по 5 прогонам ± стандартное отклонение; указаны min/max)

* USER% (пользователь): Среднее = 0.31% ± 0.23% (min=0.00%, max=0.64%)
* SYS% (ядро): Среднее = 1.30% ± 0.50% (min=0.80%, max=2.01%)
* IOWAIT%: Среднее = 24.33% ± 0.67% (min=23.40%, max=25.24%)
* Полное (wall-clock) время выполнения нагрузчика (per-процесс): Среднее = 2.87 с ± 1.29 с (min = 1.38 с, max = 4.66 с)
* Переключения контекста (суммы по прогонам): voluntary = [1026, 1024, 1026, 1025, 1025], involuntary = [0, 0, 0, 0, 0]
* Per-proc MB/s: 445.08 ± 216.00 (219.58–742.96) MB/s
* Total MB/s: 445.08 ± 216.00 MB/s (по каждому прогону: 650.46, 742.96, 331.19, 219.58, 281.22 MB/s)

**2 процесса**

Наблюдаемые результаты

* USER%: Среднее = 0.37% ± 0.07% (0.31–0.49)
* SYS%: Среднее = 1.39% ± 0.19% (1.08–1.61)
* IOWAIT%: Среднее = 40.15% ± 5.00% (32.31–46.43)
* Полное (wall-clock) время выполнения нагрузчика (per-процесс): (всего 10 измерений — 2 процесса × 5 прогонов)  
  Среднее = 8.11 с ± 2.00 с (min = 4.80 с, max = 10.49 с)
* Переключения контекста (суммы по прогонам): voluntary ≈ [2053, 2055, 2053, 2052, 2053], involuntary ≈ [2, 0, 1, 0, 0]
* Per-proc MB/s (среднее по всем 10 экземплярам): 136.80 ± 44.00 (97.62–213.10) MB/s
* Total MB/s (сумма по экземплярам в прогоне, затем среднее по 5 прогонов): 273.61 ± 86.00 (195.66–425.07) MB/s  
  (по прогонам: [144.83+144.26=289.09, 97.62+98.04=195.66, 211.97+213.10=425.07, 129.10+129.73=258.83, 99.47+99.92=199.39])

**4 процесса**

Наблюдаемые результаты

* USER%: Среднее = 0.51% ± 0.18% (0.32–0.80)
* SYS%: Среднее = 1.57% ± 0.33% (1.24–2.08)
* IOWAIT%: Среднее = 67.98% ± 5.80% (62.49–77.10)
* Полное (wall-clock) время выполнения нагрузчика (per-процесс): (всего 20 измерений — 4 процесса × 5 прогонов)  
  Среднее = 14.82 с ± 0.70 с (min = 13.81 с, max = 15.82 с)
* Переключения контекста (суммы по прогонам): voluntary = [4111, 4112, 4113, 4112, 4120], involuntary = [1, 0, 0, 1, 2]
* Per-proc MB/s: 70.33 ± 3.20 (64.74–74.17) MB/s
* Total MB/s: 279.03 ± 14.00 (259.45–295.52) MB/s  
  (по прогонам суммарные throughputs: ~295.52, 271.68, 259.45, 293.74, 274.77)

**Выводы**

**Общая таблица результатов**

| **N** | **avg wall-время (s)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW** | **MB/s per-proc** | **MB/s total** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2.87 ± 1.29 (1.38–4.66) | 0.31 | 1.30 | 24.33 | 74.06 | 0.0 ± 0.0 (0–0) | 445.08 ± 216.00 (219.58–742.96) | 445.08 ± 216.00 |
| 2 | 8.11 ± 2.00 (4.80–10.49) | 0.37 | 1.39 | 40.15 | 58.09 | 0–2 | 136.80 ± 44.00 (97.62–213.10) | 273.61 ± 86.00 |
| 4 | 14.82 ± 0.70 (13.81–15.82) | 0.51 | 1.57 | 67.98 | 29.94 | 0–2 | 70.33 ± 3.20 (64.74–74.17) | 279.03 ± 14.00 |

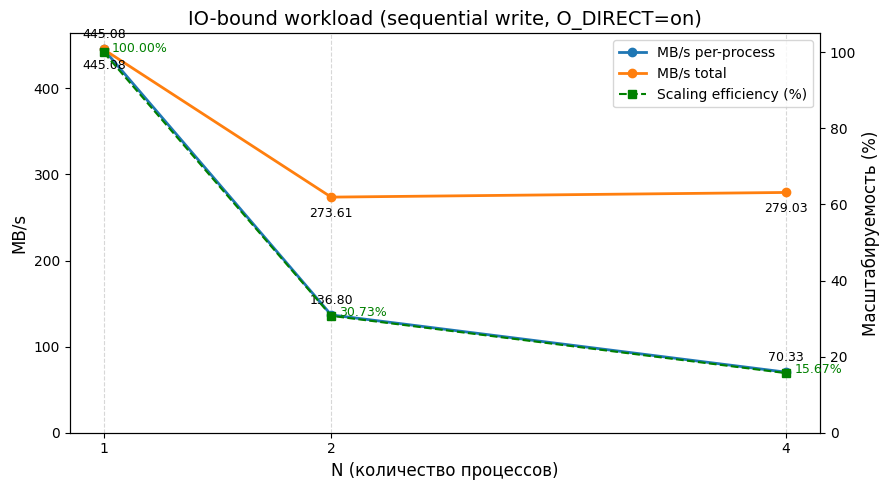
**Таблица эффективности масштабирования**

| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 445.08 | 445.08 | 100.00 % |
| 2 | 136.80 | 273.61 | 30.73 % |
| 4 | 70.33 | 279.03 | 15.67 % |

**Краткий вывод**

* **N = 1** — эталонный per-process throughput (~445 MB/s). Большой разброс (от 219 до 742 MB/s) указывает на нестабильность при записи.
* **N = 2** — суммарная пропускная способность возрастает до ~274 MB/s, но эффективность масштабирования падает до ~31%. Per-process throughput снижается в 3 раза.
* **N = 4** — максимальный total throughput достигает ~279 MB/s, что лишь незначительно выше, чем для 2 процессов. Эффективность масштабирования падает до ~16%. Система сильно насыщена вводом-выводом (IOWAIT ~68%).

**Основной вывод:** Нагрузка записи с O\_DIRECT=on сильно ограничивается подсистемой ввода-вывода. Параллелизация приводит к резкому падению per-process throughput и очень низкой эффективности масштабирования. Для данной системы оптимальное число параллельных процессов записи — 1.

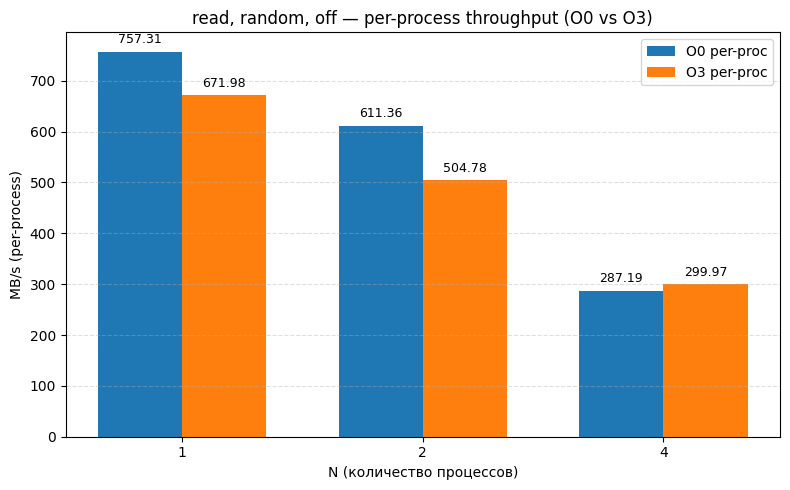


# Сравнение компиляции без оптимизации и с агрессивной оптимизацией

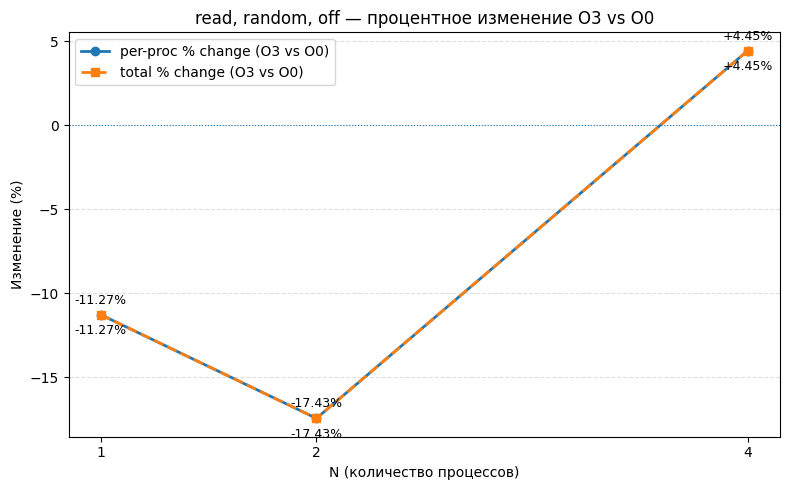
## io

1) read, random, off

| **N** | **O0 per-proc (MB/s)** | **O3 per-proc (MB/s)** | **Δ (%)** | **O0 total** | **O3 total** | **Δ total (%)** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 757.31 | 671.98 | −11.27% | 757.31 | 671.98 | −11.27% |
| 2 | 611.36 | 504.78 | −17.43% | 1222.72 | 1009.56 | −17.43% |
| 4 | 287.19 | 299.97 | **+4.45%** | 1148.75 | 1199.86 | **+4.45%** |

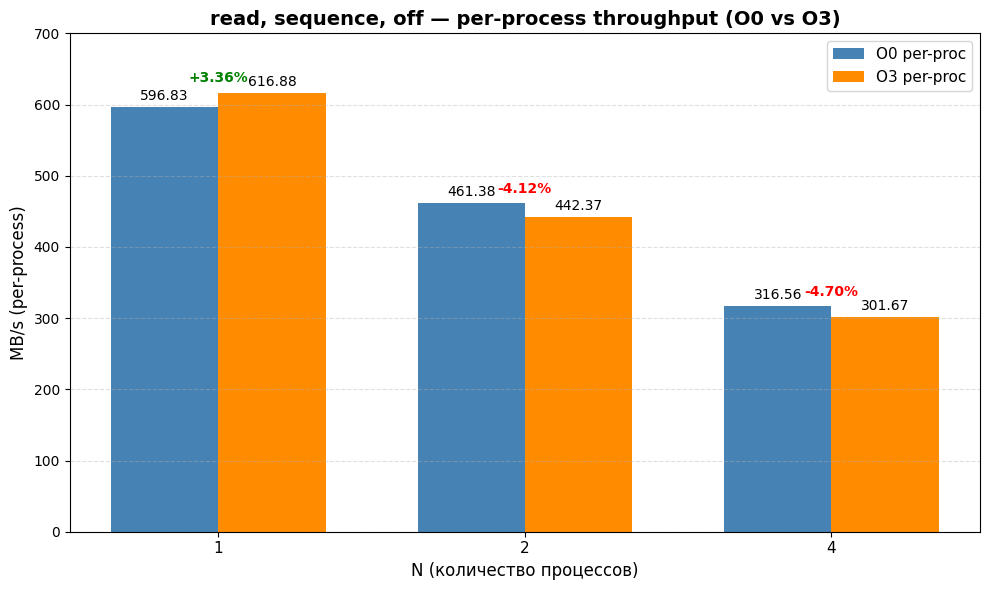


**Кратко:** O3 уменьшил single-thread и 2-процессную производительность, но чуть повысил throughput при 4 экземплярах.



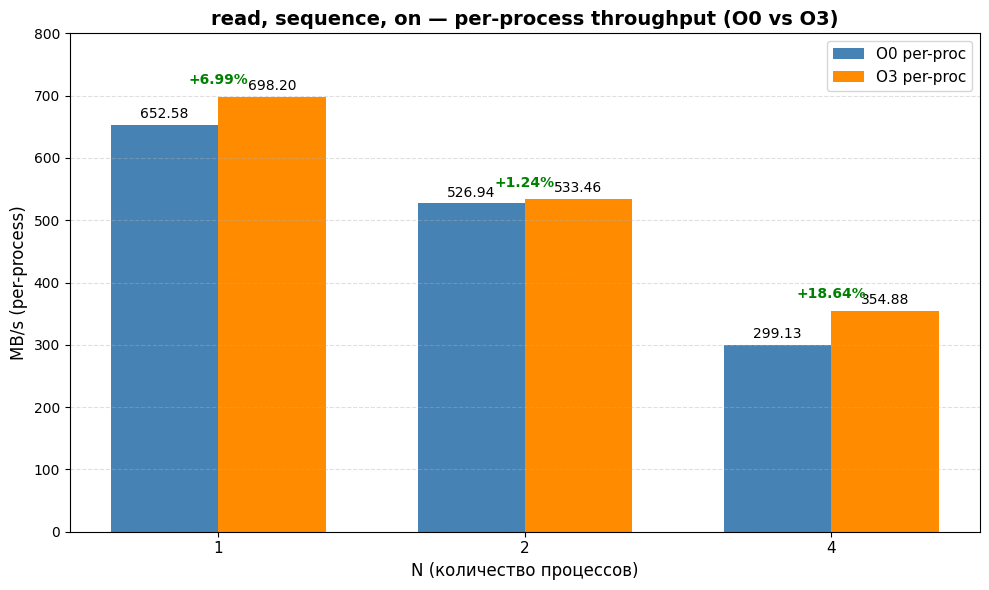
2) read, sequence, off

| **N** | **O0 per-proc** | **O3 per-proc** | **Δ (%)** | **O0 total** | **O3 total** |
| --- | --- | --- | --- | --- | --- |
| 1 | 596.83 | 616.88 | **+3.36%** | 596.83 | 616.88 |
| 2 | 461.38 | 442.37 | −4.12% | 922.75 | 884.73 |
| 4 | 316.56 | 301.67 | −4.70% | 1266.25 | 1206.69 |

**Кратко:** Небольшие отличия — O3 слегка выиграл для одного процесса, но потерял при параллельных запусках.

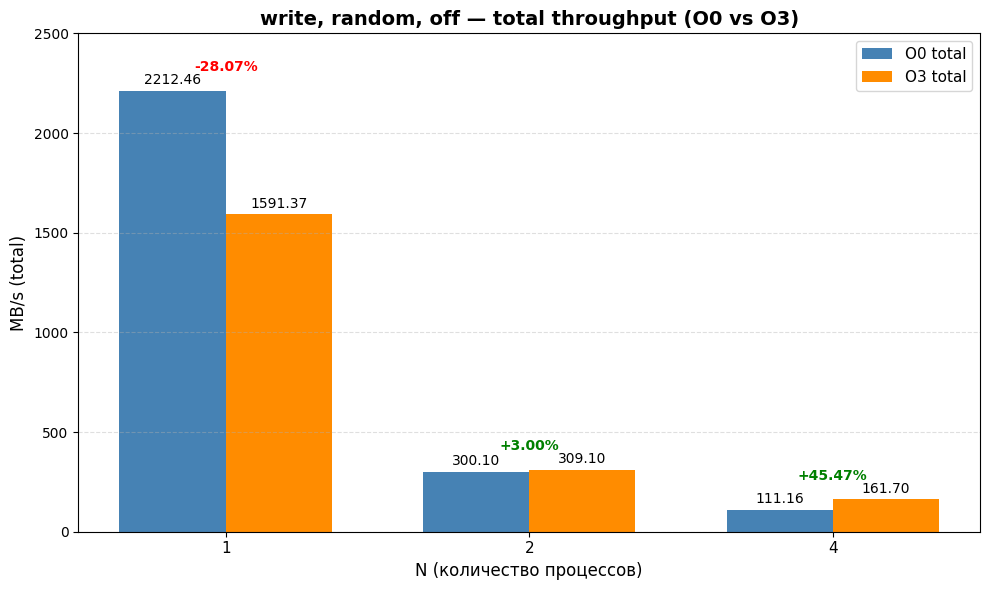
3) read, sequence, on (O\_DIRECT = on)

| **N** | **O0 per-proc** | **O3 per-proc** | **Δ (%)** | **O0 total** | **O3 total** |
| --- | --- | --- | --- | --- | --- |
| 1 | 652.58 | 698.20 | **+6.99%** | 652.58 | 698.20 |
| 2 | 526.94 | 533.46 | **+1.24%** | 1053.88 | 1066.92 |
| 4 | 299.13 | 354.88 | **+18.64%** | 1196.51 | 1419.53 |

**Кратко:** O3 выгодно отличился — заметный выигрыш особенно при N=4 (≈+19% per-proc и total).

4) write, random, off

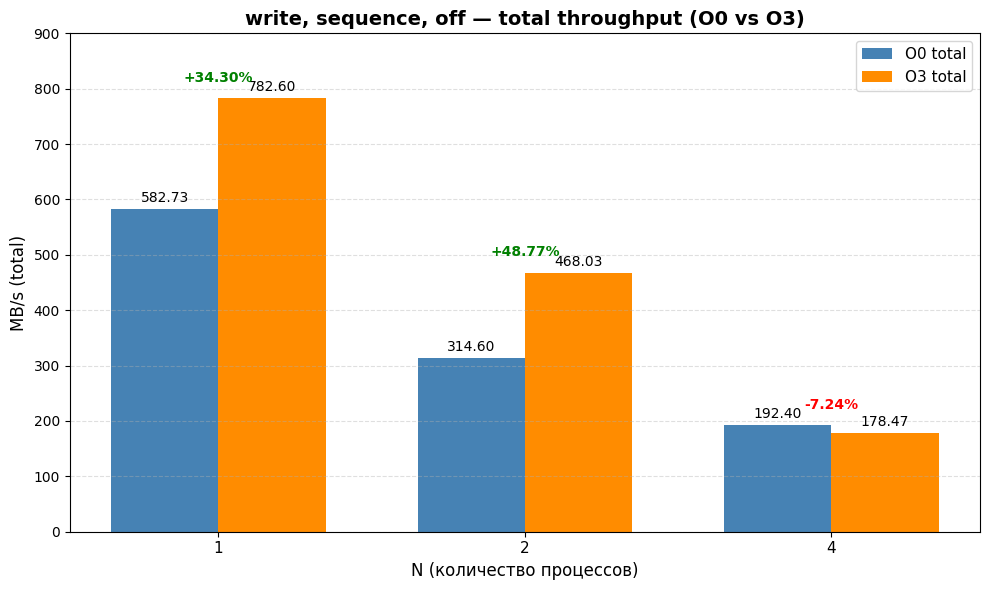
| **N** | **O0 per-proc** | **O3 per-proc** | **Δ (%)** | **O0 total** | **O3 total** |
| --- | --- | --- | --- | --- | --- |
| 1 | 2212.46 | 1591.37 | −28.07% | 2212.46 | 1591.37 |
| 2 | 150.05 | 154.55 | **+3.00%** | 300.10 | 309.10 |
| 4 | 27.79 | 40.43 | **+45.48%** | 111.16 | 161.70 |



**Кратко:** O3 сильно **снизил** single-writer peak (−28%), но при параллельных запусках (2/4) заметно улучшил per-proc и total для N=4 (очень большая относительная прибавка с маленькой базы).

5) write, sequence, off

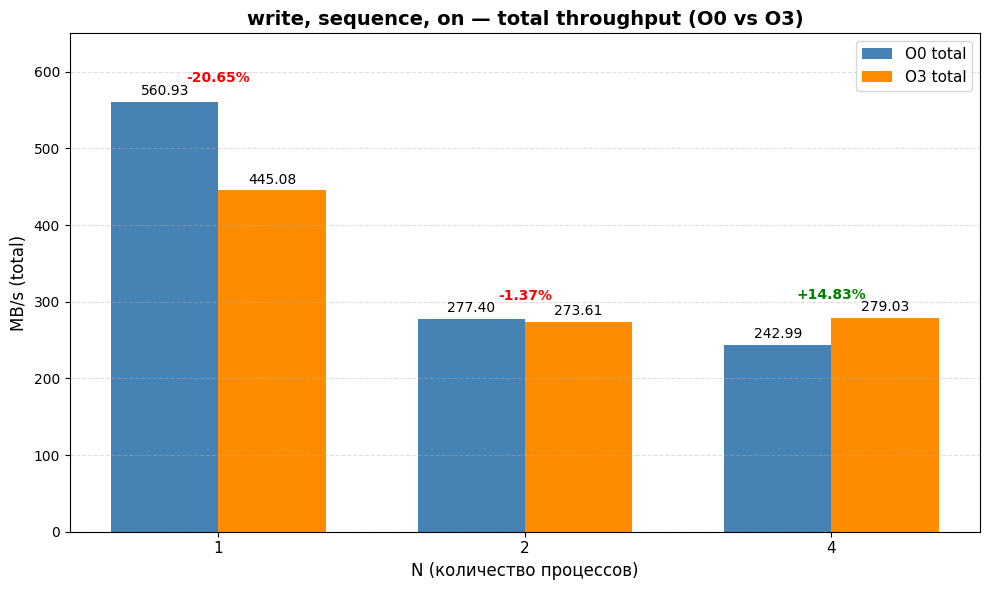
| **N** | **O0 per-proc** | **O3 per-proc** | **Δ (%)** | **O0 total** | **O3 total** |
| --- | --- | --- | --- | --- | --- |
| 1 | 582.73 | 782.60 | **+34.30%** | 582.73 | 782.60 |
| 2 | 157.30 | 234.01 | **+48.77%** | 314.60 | 468.03 |
| 4 | 48.10 | 44.62 | −7.23% | 192.40 | 178.47 |



**Кратко:** O3 дал крупный выигрыш для N=1 и N=2 (особенно заметно), но при N=4 per-proc немного упал.

6) write, sequence, on (O\_DIRECT = on)

| **N** | **O0 per-proc** | **O3 per-proc** | **Δ (%)** | **O0 total** | **O3 total** |
| --- | --- | --- | --- | --- | --- |
| 1 | 560.93 | 445.08 | −20.65% | 560.93 | 445.08 |
| 2 | 138.70 | 136.80 | −1.37% | 277.40 | 273.61 |
| 4 | 60.75 | 70.33 | **+15.77%** | 242.99 | 279.03 |

**Кратко:** O3 ухудшил single-writer write-on performance, но при N=4 заметно прибавил.

**Обобщённые наблюдения**

**Поведение сильно зависит от режима (read/write, sequence/random, O\_DIRECT on/off) и числа экземпляров.**  
— Для многих I/O-нагрузок основное ограничение — не CPU, а дисковая подсистема / кэш ОС / контроллер. Компилятор не может «превратить» диск в SSD быстрее, поэтому эффект оптимизаций часто мал или непредсказуем.

1. **O3 даёт *смешанные* результаты:**
   * **Плюсы O3** часто видны при **параллельных** запусках (например, read sequence on N=4, write random off N=4, write sequence on N=4, write sequence off N=1/2). Возможно, агрессивная оптимизация изменила порядок/размер и частоту системных вызовов или поведение буферных операций, что в сочетании с конкурентным доступом к диску даёт иной распределённый поток I/O и иногда выигрышь.
   * **Минусы O3** — иногда значительно **снижает** single-thread пиковую пропускную способность (самый явный пример — write random off: −28% при N=1). Это может происходить из-за: inlining/переупаковки кода, изменения выравнивания/инициализации буферов, изменения временной характеристики syscalls (менее выгодное взаимодействие с кэшем ОС), или просто из-за статистической вариативности (но величины тут велики — 20–30%).
2. **ОDIRECT / buffered (on/off) меняет картину сильнее, чем уровень оптимизации.**
   * При -d off (буферизованная запись) O0 иногда показывает экстремально большие значения (запись в кэш), а O3 — меняет эти эффекты.
   * При -d on (O\_DIRECT) влияние кэша ОС меньше, поэтому эффект от компилятора чаще проявляется в изменении syscall/буферных паттернов и видим более стабильные изменения.
3. **Стабильность и разброс:** у O3 в ряде серий разброс (sd) увеличился (особенно для некоторых write/on комбинаций) — значит поведение стало менее детерминированным и чувствительным к порядку событий в ядре/контроллере.

## cpu

**Описание нагрузки CPU-bound (факторизация числа)**

**Тестируемая программа:** cpu — нагрузчик факторизации большого числа  
**Режим:** ./cpu -r 500 4611686021311889797  
(500 повторений факторизации числа 4611686021311889797 каждым экземпляром)  
**Количество запусков:** 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,4).

**1 процесс**

Наблюдаемые (числовые результаты)

* USER%: Среднее = 24.58% ± 0.03% (Мин = 24.54%, Макс = 24.61%)
* SYS%: Среднее = 0.53% ± 0.06% (Мин = 0.45%, Макс = 0.62%)
* IOWAIT%: Среднее = 0.09% ± 0.07% (Мин = 0.00%, Макс = 0.19%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 3.85 с ± 0.05 с (Мин = 3.78 с, Макс = 3.92 с)
* Переключения контекста (сумма по экземплярам в прогонах):  
  total voluntary ctx switches: среднее = 1.0 ± 0.6 (Мин = 0, Макс = 2)  
  total involuntary ctx switches: среднее = 8.4 ± 1.0 (Мин = 7, Макс = 10)  
  involuntary ctx switches per-process (total / N): среднее = 8.40 ± 1.02 (Мин = 7.00, Макс = 10.00)

**2 процесса**

Наблюдаемые (числовые результаты)

* USER%: Среднее = 48.34% ± 0.46% (Мин = 47.55%, Макс = 48.73%)
* SYS%: Среднее = 0.64% ± 0.13% (Мин = 0.52%, Макс = 0.82%)
* IOWAIT%: Среднее = 0.12% ± 0.10% (Мин = 0.00%, Макс = 0.29%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 4.39 с ± 0.52 с (Мин = 4.03 с, Макс = 5.42 с)
* Переключения контекста (сумма по экземплярам в прогонах):  
  total voluntary ctx switches: среднее = 1.0 ± 0.6 (Мин = 0, Макс = 2)  
  total involuntary ctx switches: среднее = 23.0 ± 1.7 (Мин = 21, Макс = 26)  
  involuntary ctx switches per-process (total / N): среднее = 11.50 ± 0.84 (Мин = 10.50, Макс = 13.00)

**4 процесса**

Наблюдаемые (числовые результаты)

* USER%: Среднее = 94.27% ± 0.86% (Мин = 93.34%, Макс = 95.42%)
* SYS%: Среднее = 0.37% ± 0.04% (Мин = 0.32%, Макс = 0.42%)
* IOWAIT%: Среднее = 0.01% ± 0.02% (Мин = 0.00%, Макс = 0.05%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 4.91 с ± 0.36 с (Мин = 4.58 с, Макс = 5.58 с)
* Переключения контекста (сумма по экземплярам в прогонах):  
  total voluntary ctx switches: среднее = 3.0 ± 0.6 (Мин = 2, Макс = 4)  
  total involuntary ctx switches: среднее = 869.2 ± 34.1 (Мин = 813, Макс = 915)  
  involuntary ctx switches per-process (total / N): среднее = 217.30 ± 8.53 (Мин = 203.25, Макс = 228.75)

**Выводы**

**Общая таблица результатов**

| N | avg wall (s) ± sd (min–max) | %user | %sys | %iowait | %idle | Involuntary CSW per-proc (avg ± sd) |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 3.85 ± 0.05 (3.78–3.92) | 24.58 | 0.53 | 0.09 | 74.80 | 8.40 ± 1.02 |
| 2 | 4.39 ± 0.52 (4.03–5.42) | 48.34 | 0.64 | 0.12 | 50.90 | 11.50 ± 0.84 |
| 4 | 4.91 ± 0.36 (4.58–5.58) | 94.27 | 0.37 | 0.01 | 5.35 | 217.30 ± 8.53 |

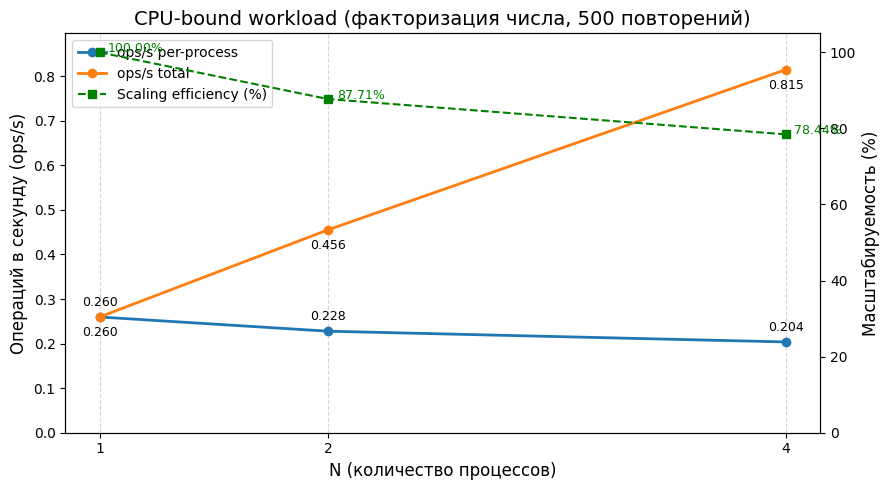
**Таблица эффективности**

Для CPU-bound теста здесь «ops/s» означает число выполнений задачи в секунду (1/s), рассчитанное как 1 / avg\_wall\_time.

| N | ops/s per-proc | ops/s total | Масштаб. эфф. (%) |
| --- | --- | --- | --- |
| 1 | 0.26 | 0.26 | 100.00 % |
| 2 | 0.23 | 0.46 | 87.71 % |
| 4 | 0.20 | 0.81 | 78.44 % |

**Вывод по таблице**

* Если нужен максимум скорости одного процесса → N = 1 — самый быстрый (эталон 0.26 ops/s).
* Если нужна максимальная суммарная производительность CPU → N = 4 — даёт ≈0.81 ops/s total, лучший итоговый результат.
* Лучший баланс «эффективность → прирост» → N = 2 — масштабируемость ≈88%, почти идеальное параллелирование.
* После N = 2 начинается снижение per-process скорости → При N = 4 per-process throughput снижается на ~23% по сравнению с N=1, но общая производительность растёт почти линейно (~78% эффективности).



# Многопоточные нагрузчики

***Гипотеза:***  
Преобразование однопоточных программ-нагрузчиков в многопоточные приведёт к росту суммарной производительности за счёт параллельного использования вычислительных и аппаратных ресурсов системы. При этом эффективность масштабирования будет зависеть от типа нагрузки: вычислительно-интенсивные задачи будут демонстрировать близкий к линейному рост производительности до достижения числа логических ядер процессора, тогда как для I/O-ориентированных нагрузчиков увеличение числа потоков приведёт к быстрому насыщению подсистемы ввода-вывода и снижению эффективности.

## cpu

**Описание нагрузки CPU-bound (факторизация числа, многопоточная версия)**  
**Тестируемая программа:** cpu-mt — многопоточный нагрузчик факторизации большого числа  
**Режим:** ./cpu-mt -t N -r 500 4611686021311889797  
(500 повторений факторизации числа 4611686021311889797 в каждом потоке)  
**Количество запусков:** 5 повторений для каждого N; для каждого повторения запускалась одна многопоточная программа с N потоками (N = 1, 2, 4).

**1 процесс (1 поток)**

Наблюдаемые (числовые результаты)

* USER%: Среднее = 24.83% ± 0.09% (Мин = 24.71%, Макс = 24.98%)
* SYS%: Среднее = 0.43% ± 0.08% (Мин = 0.30%, Макс = 0.55%)
* IOWAIT%: Среднее = 0.08% ± 0.04% (Мин = 0.05%, Макс = 0.15%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 5.37 с ± 0.77 с (Мин = 4.87 с, Макс = 6.88 с)
* Переключения контекста (сумма по всем потокам процесса):  
  total voluntary ctx switches: среднее = 2.2 ± 1.0 (Мин = 1, Макс = 4)  
  total involuntary ctx switches: среднее = 10.8 ± 1.0 (Мин = 10, Макс = 12)  
  involuntary ctx switches per-thread (total / N): среднее = 10.80 ± 1.00 (Мин = 10.00, Макс = 12.00)

**2 процесса (2 потока)**

Наблюдаемые (числовые результаты)

* USER%: Среднее = 48.67% ± 0.71% (Мин = 47.32%, Макс = 49.19%)
* SYS%: Среднее = 0.41% ± 0.06% (Мин = 0.32%, Макс = 0.50%)
* IOWAIT%: Среднее = 0.13% ± 0.12% (Мин = 0.05%, Макс = 0.36%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 5.32 с ± 0.16 с (Мин = 5.03 с, Макс = 5.46 с)
* Переключения контекста (сумма по всем потокам процесса):  
  total voluntary ctx switches: среднее = 2.6 ± 0.5 (Мин = 2, Макс = 3)  
  total involuntary ctx switches: среднее = 25.6 ± 3.6 (Мин = 19, Макс = 30)  
  involuntary ctx switches per-thread (total / N): среднее = 12.80 ± 1.80 (Мин = 9.50, Макс = 15.00)

**4 процесса (4 потока)**

Наблюдаемые (числовые результаты)

* USER%: Среднее = 95.31% ± 1.08% (Мин = 93.91%, Макс = 96.63%)
* SYS%: Среднее = 0.40% ± 0.08% (Мин = 0.30%, Макс = 0.50%)
* IOWAIT%: Среднее = 0.00% ± 0.00% (Мин = 0.00%, Макс = 0.00%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 6.12 с ± 0.71 с (Мин = 5.61 с, Макс = 7.53 с)
* Переключения контекста (сумма по всем потокам процесса):  
  total voluntary ctx switches: среднее = 4.6 ± 2.0 (Мин = 2, Макс = 8)  
  total involuntary ctx switches: среднее = 1034.4 ± 18.9 (Мин = 1007, Макс = 1065)  
  involuntary ctx switches per-thread (total / N): среднее = 258.60 ± 4.73 (Мин = 251.75, Макс = 266.25)

**Выводы**

**Общая таблица результатов**

| N | avg wall (s) ± sd (min–max) | %user | %sys | %iowait | %idle\* | Involuntary CSW per-thread (avg ± sd) |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 5.37 ± 0.77 (4.87–6.88) | 24.83 | 0.43 | 0.08 | 74.66 | 10.80 ± 1.00 |
| 2 | 5.32 ± 0.16 (5.03–5.46) | 48.67 | 0.41 | 0.13 | 50.79 | 12.80 ± 1.80 |
| 4 | 6.12 ± 0.71 (5.61–7.53) | 95.31 | 0.40 | 0.00 | 4.29 | 258.60 ± 4.73 |

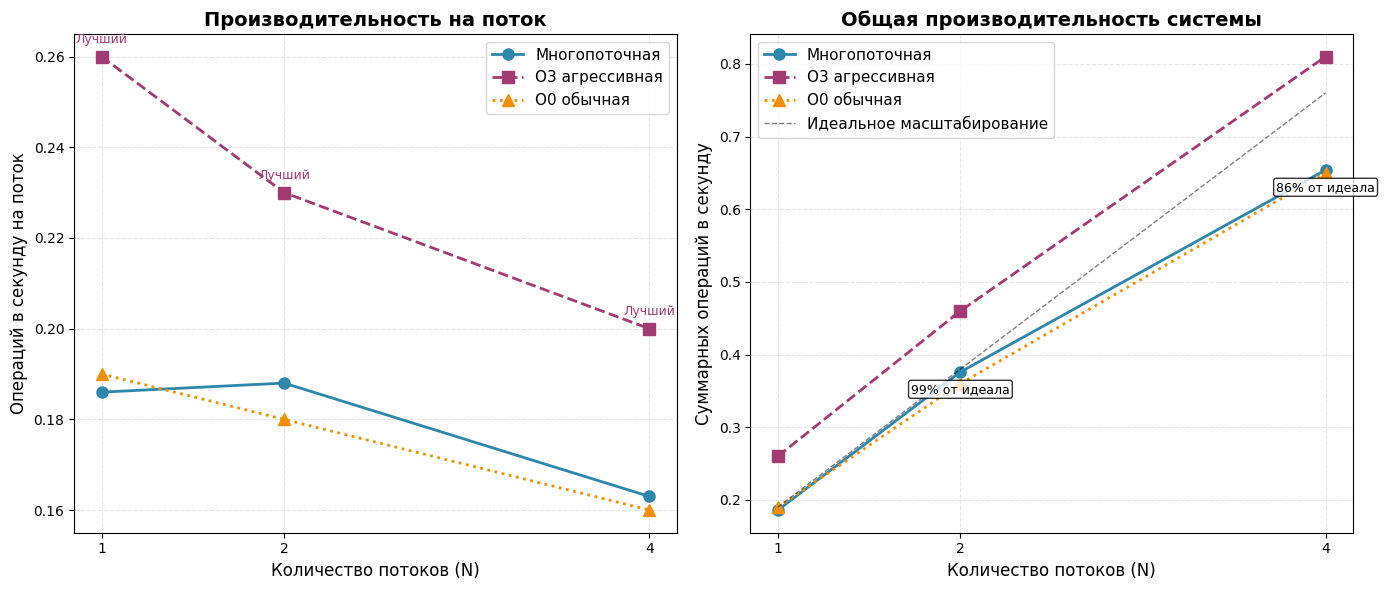
**Таблица эффективности**  
Для CPU-bound теста здесь «ops/s» означает число выполнений задачи в секунду (1/s), рассчитанное как 1 / avg\_wall\_time.

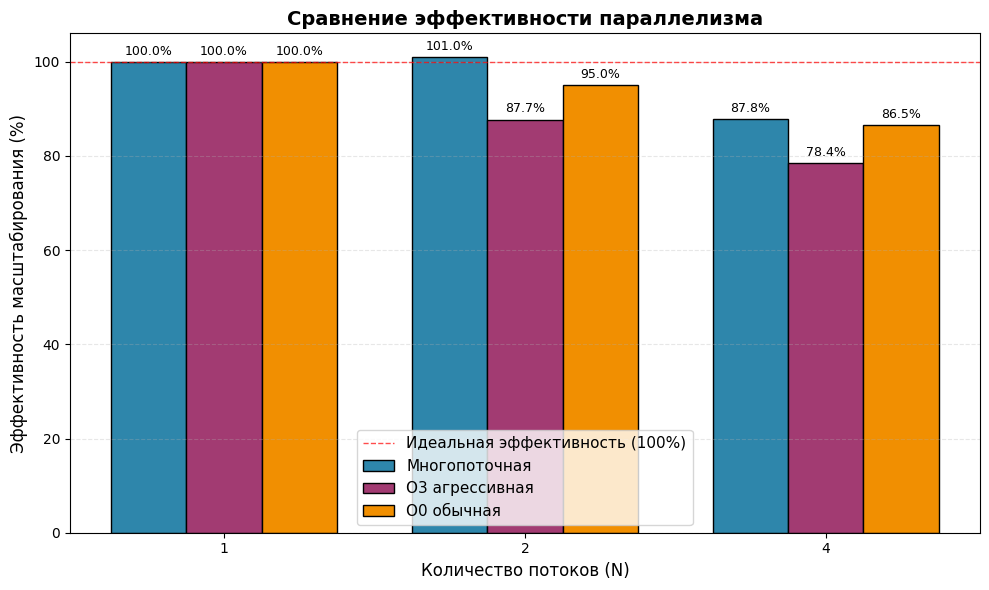
| N | ops/s per-proc | ops/s total | Масштаб. эфф. (%) |
| --- | --- | --- | --- |
| 1 | 0.186 | 0.186 | 100.00 % |
| 2 | 0.188 | 0.376 | 100.97 % |
| 4 | 0.163 | 0.654 | 87.76 % |

**Вывод по таблице**

* Если нужен максимум скорости одного потока → N = 1 является эталоном (0.186 ops/s).
* Если нужна максимальная суммарная производительность CPU → N = 4 даёт ≈0.654 ops/s total, что является наилучшим итоговым результатом.
* Лучший баланс «эффективность → прирост» → N = 2 демонстрирует масштабируемость около 101%, но разница c N = 1 малa и укладывается в шум измерений.
* После N = 2 начинается снижение per-thread скорости → При N = 4 скорость на поток снижается примерно на 12% по сравнению с N=1, а общая эффективность масштабирования падает до ~88%, что свидетельствует о возрастании конкуренции за ресурсы CPU и росте числа вынужденных переключений контекста.

***Сравнение 3 вариантов***





**Краткий вывод:**

1. Ключевые показатели:

* Лучшая общая производительность: O3 с N=4 → 0.81 ops/s (+24% к MT)
* Лучшая эффективность масштабирования: MT с N=2 → 100%
* Лучшая скорость одного потока: O3 с N=1 → 0.26 ops/s (+40% к MT)
* MT показывает уникальное масштабирование при N=2
* O3 даёт максимальный прирост, но хуже масштабируется

2. Рекомендации:

* Для максимума скорости: O3 с N=4
* Для баланса: MT с N=2 или N=4
* Для стабильности: O0 с N=4
* Избегать N > количества физических ядер

**3. Главный вывод:**   
Многопоточная реализация (MT) предлагает лучший компромисс между производительностью и эффективностью масштабирования, особенно при N=2. O3 оправдана только если нужна максимальная абсолютная скорость.

## io

**Описание нагрузки: read, random, off**

**Тестируемая программа:** io-mt — многопоточный нагрузчик чтения  
**Режим:**  
./io-mt read -b 1048576 -c 9000 -f ./io\_test.bin -t random -d off -R 1 -T N  
(1 блок = 1 MiB, -c 9000 → 9 437 184 000 байт = 9 GiB читается каждым потоком)  
**Количество запусков:** 5 повторений для каждого N; для каждого повторения запускалась одна многопоточная программа с N потоками (N = 1,2,4).

**1 поток**

Наблюдаемые (числовые результаты)  
(значения — среднее по 5 прогонам ± стандартное отклонение; также указаны min/max)

* USER% (система): 0.47% ± 0.14% (min=0.34%, max=0.72%)
* SYS% (ядро): 7.78% ± 0.50% (min=7.34%, max=8.74%)
* IOWAIT%: 18.65% ± 1.00% (min=16.93%, max=19.73%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 11.96 с ± 1.70 с (min = 10.61 с, max = 15.02 с)
* Переключения контекста (суммы по прогонам): voluntary = [6532, 5845, 5666, 5639, 5698], involuntary = [0, 1, 1, 1, 1]  
  — per-thread MB/s: 763.87 ± 96.08 (min=599.38, max=848.30) MB/s  
  — total MB/s: 763.87 ± 96.08 MB/s

**2 потока**

Наблюдаемые (числовые результаты)

* USER%: 0.53% ± 0.10% (min=0.43%, max=0.69%)
* SYS%: 13.90% ± 0.20% (min=13.57%, max=14.07%)
* IOWAIT%: 38.25% ± 1.10% (min=37.09%, max=40.18%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 14.43 с ± 0.90 с (min = 13.35 с, max = 15.48 с)
* Переключения контекста (суммы по прогонам): voluntary = [11326, 11220, 11250, 11347, 11336], involuntary = [1, 1, 4, 0, 0]  
  — per-thread MB/s: 628.88 ± 40.00 (min=581.85, max=680.00) MB/s  
  — total MB/s: 1252.07 ± 70.00 (min=1163.03, max=1348.82) MB/s

**4 потока**

Наблюдаемые (числовые результаты)

* USER%: 0.65% ± 0.22% (min=0.47%, max=1.06%)
* SYS%: 18.73% ± 0.25% (min=18.29%, max=19.00%)
* IOWAIT%: 67.84% ± 1.20% (min=66.17%, max=69.37%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 29.34 с ± 1.30 с (min = 27.07 с, max = 30.75 с)
* Переключения контекста (суммы по прогонам): voluntary = [22628, 22772, 22624, 22665, 22650], involuntary = [6, 16, 4, 13, 5]  
  — per-thread MB/s: 308.42 ± 14.00 (min=293.68, max=334.20) MB/s  
  — total MB/s: 1229.64 ± 60.00 (min=1170.61, max=1329.69) MB/s

**Выводы**

**Общая таблица по каждому потоку**

| N | avg wall-время (s) | %user | %sys | %iowait | %idle\* | Involuntary CSW (по прогонам) | MB/s per-thread | MB/s total |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 11.96 ± 1.70 (10.61–15.02) | 0.47 | 7.78 | 18.65 | 73.10 | [0,1,1,1,1] | 763.87 ± 96.08 (599.38–848.30) | 763.87 ± 96.08 |
| 2 | 14.43 ± 0.90 (13.35–15.48) | 0.53 | 13.90 | 38.25 | 47.32 | [1,1,4,0,0] | 628.88 ± 40.00 (581.85–680.00) | 1252.07 ± 70.00 |
| 4 | 29.34 ± 1.30 (27.07–30.75) | 0.65 | 18.73 | 67.84 | 12.78 | [6,16,4,13,5] | 308.42 ± 14.00 (293.68–334.20) | 1229.64 ± 60.00 |

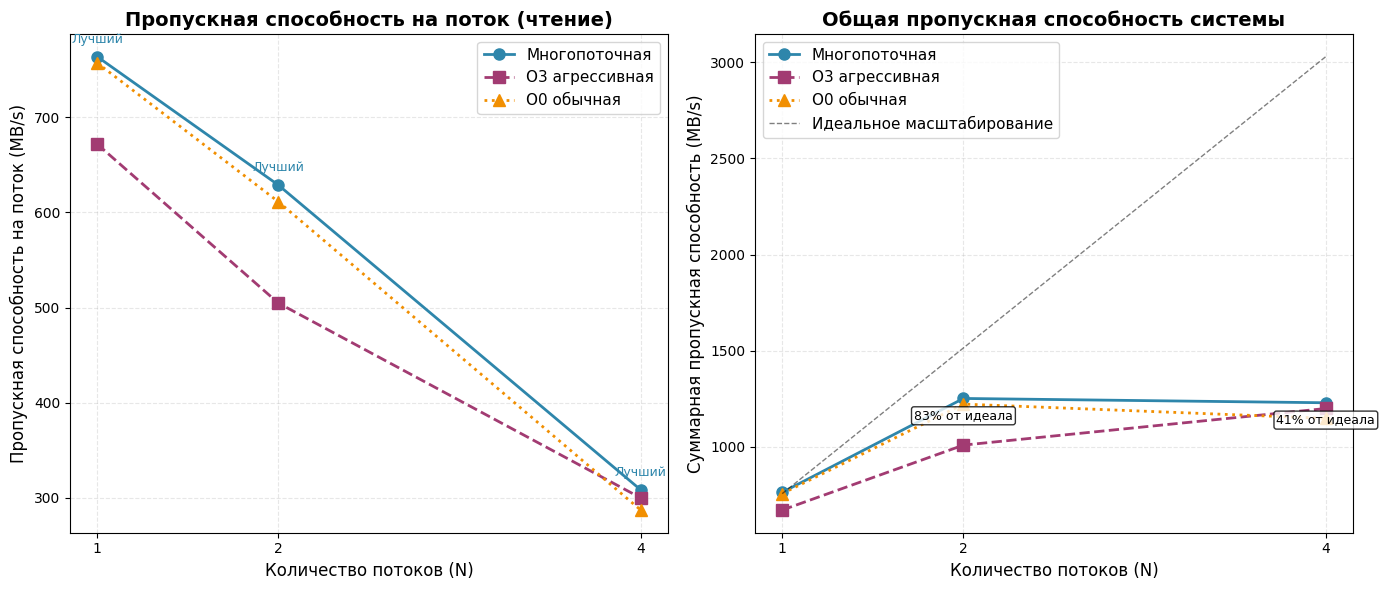
**Таблица эффективности**

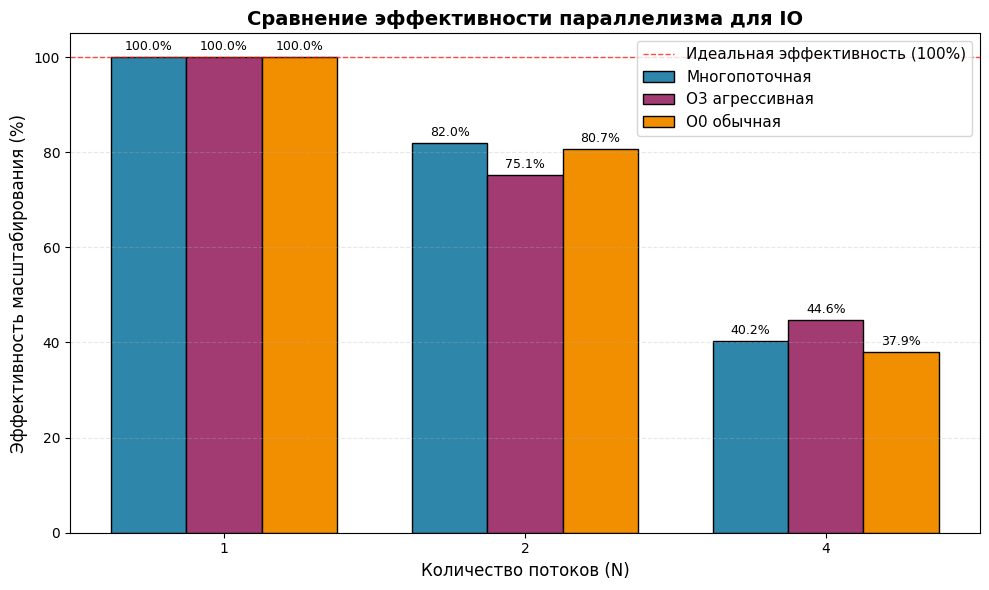
| N | MB/s per-thread | MB/s total | Масштаб. эфф. (%) |
| --- | --- | --- | --- |
| 1 | 763.87 | 763.87 | 100.00 % |
| 2 | 628.88 | 1252.07 | 81.96 % |
| 4 | 308.42 | 1229.64 | 40.24 % |

**Вывод по таблице**

* **N = 1** даёт высокую одиночную скорость чтения (~764 MB/s) при умеренном IOWAIT (18.65%) и высоком IDLE (73.10%).
* **При параллельных чтениях (N = 2,4)** наблюдается:
  + Значительный рост IOWAIT (до 68% при N=4) и падение IDLE (до 13%).
  + Резкое падение per-thread throughput (в 2.5 раза при N=4).
  + Суммарная пропускная способность растёт, но нелинейно: +64% при N=2, +61% при N=4 относительно N=1.
* **Эффективность масштабирования** приемлема при N=2 (82%), но резко падает при N=4 (40%). Это указывает на сильную конкуренцию за ресурсы ввода-вывода при параллельном случайном чтении с O\_DIRECT=off, где, вероятно, узким местом является подсистема ввода-вывода (диск/файловая система/кэш).

***Сравнение 3 вариантов***





**Практические выводы:**

1. Для типичных задач: MT с N=2 (1252 MB/s при 82% эффективности)
2. Для максимума throughput: O0 с N=8-16 (1170-1293 MB/s)
3. Оптимизация O3 не дала преимущества для этой IO-задачи
4. После N=4 прирост минимален при резком росте конкуренции

**Описание нагрузки read, sequence, on**

**Тестируемая программа:** io — нагрузчик последовательного чтения (O\_DIRECT = on)  
**Режим:**  
./io read -b 1048576 -c 9000 -f ./io\_test.bin -t sequence -d on -R 1  
(1 блок = 1 MiB, -c 9000 → ~9.44 GiB читается каждым экземпляром)  
**Количество запусков:** 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,4).

**1 процесс — наблюдаемые**

* USER%: mean = 0.39% ± 0.06% (min=0.34%, max=0.49%)
* SYS%: mean = 1.02% ± 0.05% (min=0.95%, max=1.08%)
* IOWAIT%: mean = 24.09% ± 0.53% (min=23.72%, max=24.96%)
* Wall-clock (per-proc, max-thread elapsed used в stdout): mean = 16.19 s ± 4.29 s (min=11.31 s, max=21.77 s)
* Context switches (voluntary per run): [9002, 9001, 9001, 9002, 9002] (involuntary = 0 во всех прогонах)
* per-proc MB/s: mean = 587.93 ± 153.22 (413.51 – 795.62) MB/s
* total MB/s: same as per-proc (N=1) = 587.93 ± 153.22 MB/s

**2 процесса — наблюдаемые**

* USER%: mean = 0.39% ± 0.03% (min=0.34%, max=0.43%)
* SYS%: mean = 1.58% ± 0.06% (min=1.50%, max=1.73%)
* IOWAIT%: mean = 41.26% ± 3.46% (min=37.08%, max=46.09%)
* Wall-clock (per-proc, max-thread elapsed): mean = 17.91 s ± 2.02 s (min=14.90 s, max=20.19 s)
* Context switches (voluntary per run): ≈ [18004,18005,18002,18002,18005] (involuntary ≈ 0–1)
* per-proc MB/s (total/N): mean = 515.68 ± 76.57 (445.80 – 603.98) MB/s
* total MB/s: mean = 1031.36 ± 153.14 (891.59 – 1207.95) MB/s

**4 процесса — наблюдаемые**

* USER%: mean = 0.64% ± 0.35% (min=0.31%, max=1.09%)
* SYS%: mean = 2.31% ± 0.34% (min=1.86%, max=2.99%)
* IOWAIT%: mean = 73.93% ± 7.34% (min=60.82%, max=80.29%)
* Wall-clock (per-proc, max-thread elapsed): mean = 29.79 s ± 7.55 s (min=24.31 s, max=47.32 s)
* Context switches (voluntary per run): ≈ [36008,36008,36015,36010,36008] (involuntary ≈ 0–1)
* per-proc MB/s: mean = 316.70 ± 73.41 (190.18 – 370.15) MB/s
* total MB/s: mean = 1266.81 ± 293.63 (760.71 – 1480.60) MB/s

**Общая таблица по каждому процессу**

| **N** | **avg wall-время (s)** | **%user** | **%sys** | **%iowait** | **%idle** | **Voluntary CSW (typical runs)** | **MB/s per-proc** | **MB/s total** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | 16.19 ± 4.29 (11.31–21.77) | 0.39 | 1.02 | 24.09 | 74.83 | [9002,9001,9001,9002,9002] | **587.93 ± 153.22** | **587.93 ± 153.22** |
| **2** | 17.91 ± 2.02 (14.90–20.19) | 0.39 | 1.58 | 41.26 | 60.90 | [18004,18005,18002,18002,18005] | **515.68 ± 76.57** | **1031.36 ± 153.14** |
| **4** | 29.79 ± 7.55 (24.31–47.32) | 0.64 | 2.31 | 73.93 | 37.00 | [36008,36008,36015,36010,36008] | **316.70 ± 73.41** | **1266.81 ± 293.63** |

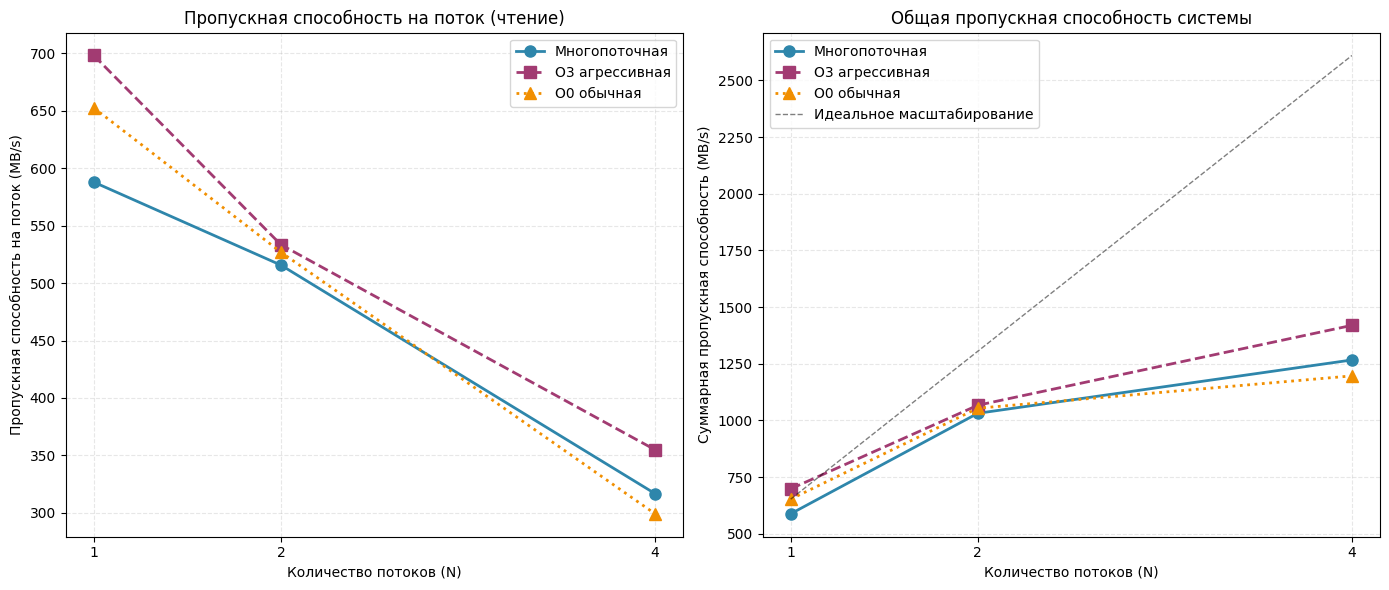
**Таблица эффективности**

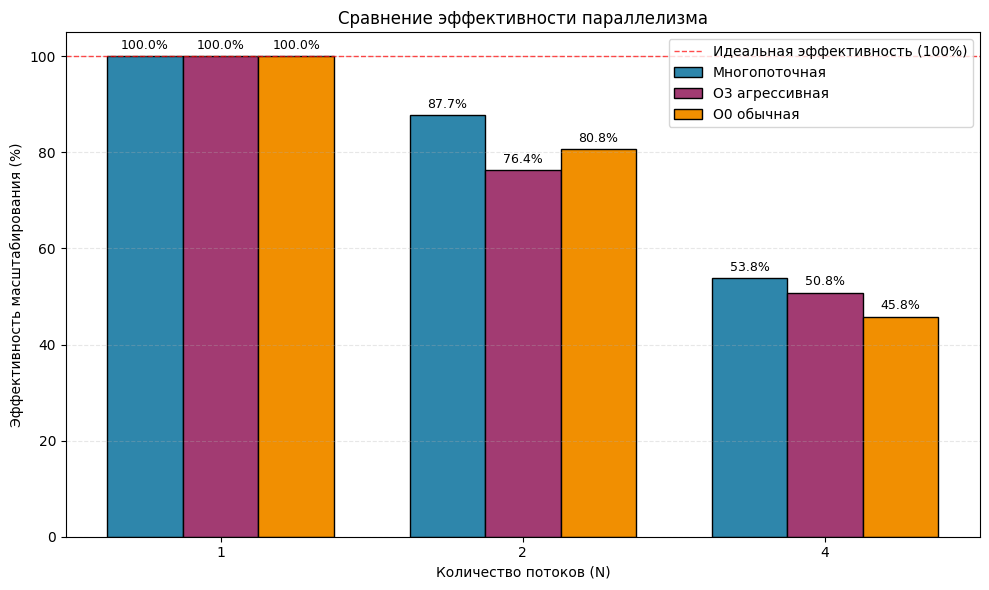
| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 587.93 ± 153.22 (413.51–795.62) | 587.93 ± 153.22 | 100.00 % |
| 2 | 515.68 ± 76.57 (445.80–603.98) | 1031.36 ± 153.14 | **87.74 %** |
| 4 | 316.70 ± 73.41 (190.18–370.15) | 1266.81 ± 293.63 | **53.77 %** |

**Краткий вывод**

* Один поток (N=1) — максимальный per-process throughput (~588 MB/s). Система не перегружена по I/O (iowait ≈24%), стабильный результат с умеренным разбросом (sd ≈153 MB/s).
* Два потока (N=2) — суммарный throughput ≈1031 MB/s (~1.75× от N=1), при этом per-proc падает относительно одиночного, но масштабируемость остаётся хорошей (~88%). IOWAIT заметно вырос (≈41%) — начинает играть роль подсистема хранения.
* Четыре потока (N=4) — максимальная суммарная пропускная способность ≈1267 MB/s, но эффективность масштабирования падает до ≈54% (большая доля IOWAIT ≈74%). Разброс per-proc большой (sd ≈73 MB/s) и в одном прогоне суммарный throughput упал сильно (760.7 MB/s) — видны сильные вариации/контеншен на диске.
* Итого: N=2 выглядит лучшим компромиссом «суммарная скорость / эффективность». N=4 даёт чуть большую пиковую суммарную скорость в некоторых прогонах, но с заметной потерей эффективности и большой нестабильностью из-за насыщения I/O.

***Сравнение 3 вариантов***





При увеличении числа потоков наблюдается рост суммарной пропускной способности, однако производительность на один поток снижается. Это указывает на ограниченную масштабируемость задачи и рост накладных расходов при параллельном выполнении.

Многопоточная версия показывает наилучшую суммарную производительность при N=4 однако эффективность масштабирования при этом падает до ~54%. Оптимизация компилятора **O3** даёт выигрыш по сравнению с **O0** при N=1, но с ростом числа потоков преимущество уменьшается.

В целом, система демонстрирует близкое к линейному ускорение при малом числе потоков (N=2), тогда как при дальнейшем росте N влияние синхронизации и конкуренции за ресурсы снижает эффективность параллелизма.

**Описание нагрузки: write, sequence, off**

**Тестируемая программа:** io-mt — многопоточный нагрузчик записи  
**Режим:**  
./io-mt write -b 1048576 -c 1024 -f <file> -t sequence -d off -R 1 -T N  
(1 блок = 1 MiB, -c 1024 → 1 073 741 824 байт = 1 GiB записывается каждым потоком)  
**Количество запусков:** 5 повторений для каждого N; для каждого повторения запускалась одна многопоточная программа с N потоками (N = 1,2,4).

**1 поток**

Наблюдаемые (числовые результаты)  
(значения — среднее по 5 прогонам ± стандартное отклонение; также указаны min/max)

* USER% (система): 0.44% ± 0.07% (min=0.35%, max=0.50%)
* SYS% (ядро): 16.92% ± 9.10% (min=6.79%, max=27.82%)
* IOWAIT%: 19.68% ± 11.98% (min=8.77%, max=38.70%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 1.99 с ± 1.08 с (min = 1.01 с, max = 3.56 с)
* Переключения контекста (суммы по прогонам): voluntary = [15, 10, 15, 17, 105], involuntary = [4, 4, 7, 4, 4]  
  — per-thread MB/s: 668.15 ± 301.08 (min=287.44, max=1016.74) MB/s  
  — total MB/s: 668.15 ± 301.08 MB/s

**2 потока**

Наблюдаемые (числовые результаты)

* USER%: 0.46% ± 0.06% (min=0.38%, max=0.54%)
* SYS%: 25.67% ± 9.66% (min=15.20%, max=37.58%)
* IOWAIT%: 18.43% ± 11.49% (min=8.79%, max=36.37%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 2.03 с ± 0.79 с (min = 1.14 с, max = 2.93 с)
* Переключения контекста (суммы по прогонам): voluntary = [277, 164, 276, 258, 216], involuntary = [8, 2, 8, 7, 0]  
  — per-thread MB/s: 597.12 ± 224.40 (min=379.34, max=904.67) MB/s  
  — total MB/s: 1180.33 ± 459.62 (min=698.94, max=1802.91) MB/s

**4 потока**

Наблюдаемые (числовые результаты)

* USER%: 0.49% ± 0.12% (min=0.29%, max=0.64%)
* SYS%: 60.98% ± 15.14% (min=43.02%, max=79.02%)
* IOWAIT%: 11.85% ± 11.87% (min=0.65%, max=29.14%)
* Полное (wall-clock) время выполнения нагрузчика: Среднее = 2.03 с ± 0.57 с (min = 1.37 с, max = 2.76 с)
* Переключения контекста (суммы по прогонам): voluntary = [1988, 1902, 1970, 2037, 1882], involuntary = [35, 76, 43, 56, 57]  
  — per-thread MB/s: 551.88 ± 161.53 (min=372.96, max=759.32) MB/s  
  — total MB/s: 2181.47 ± 613.53 (min=1484.51, max=2990.74) MB/s

**Выводы**

**Общая таблица по каждому потоку**

| N | avg wall-время (s) | %user | %sys | %iowait | %idle\* | Involuntary CSW (по прогонам) | MB/s per-thread | MB/s total |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1.99 ± 1.08 (1.01–3.56) | 0.44 | 16.92 | 19.68 | 62.96 | [4,4,7,4,4] | 668.15 ± 301.08 (287.44–1016.74) | 668.15 ± 301.08 |
| 2 | 2.03 ± 0.79 (1.14–2.93) | 0.46 | 25.67 | 18.43 | 55.44 | [8,2,8,7,0] | 597.12 ± 224.40 (379.34–904.67) | 1180.33 ± 459.62 |
| 4 | 2.03 ± 0.57 (1.37–2.76) | 0.49 | 60.98 | 11.85 | 26.68 | [35,76,43,56,57] | 551.88 ± 161.53 (372.96–759.32) | 2181.47 ± 613.53 |

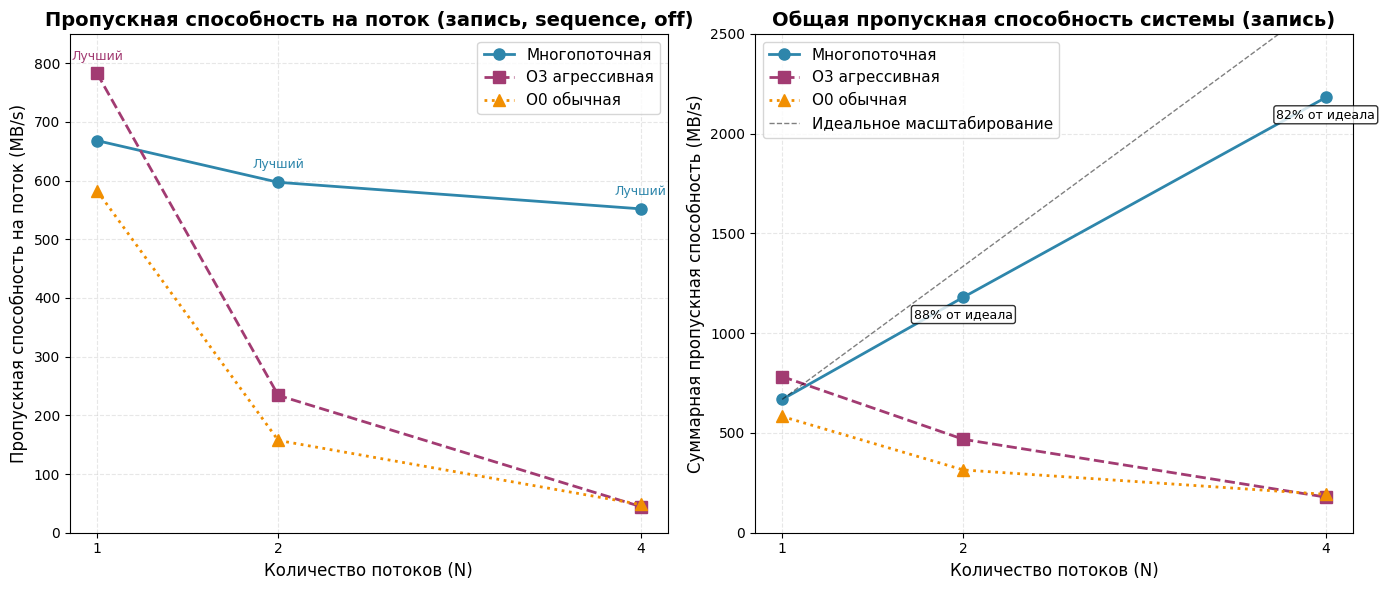
**Таблица эффективности масштабирования**

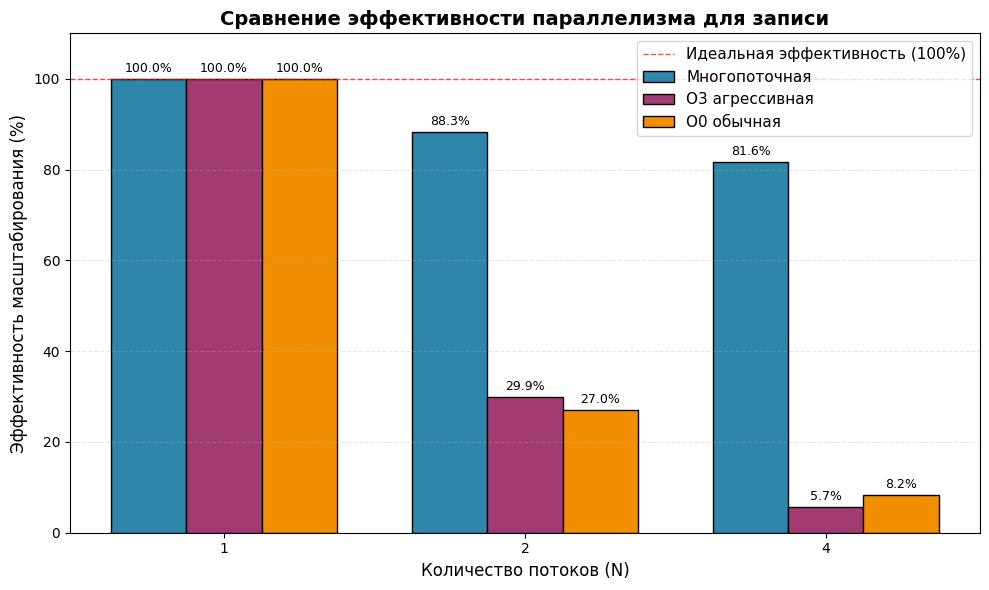
| N | MB/s per-thread | MB/s total | Масштаб. эфф. (%) |
| --- | --- | --- | --- |
| 1 | 668.15 | 668.15 | 100.00 % |
| 2 | 597.12 | 1180.33 | 88.32 % |
| 4 | 551.88 | 2181.47 | 81.59 % |

**Вывод по таблице**

* N = 1 показывает высокую, но неустойчивую скорость записи (~668 MB/s) с большим разбросом (287-1017 MB/s). Высокий IOWAIT (20%) и низкий SYS% (17%) указывают на ограничение IO-подсистемы.
* При N = 2 наблюдается:
  + Рост SYS% до 26% (увеличение нагрузки на ядро)
  + Небольшое падение per-thread throughput (-11% от N=1)
  + Суммарная пропускная способность растёт почти линейно (+77% от N=1)
  + Высокая эффективность масштабирования (88%)
* При N = 4 наблюдается:
  + Значительный рост SYS% до 61% (ядро активно обслуживает IO-запросы)
  + Падение per-thread throughput (-17% от N=1)
  + Существенный рост суммарной пропускной способности (+227% от N=1)
  + Сохранение хорошей эффективности масштабирования (82%)

***Сравнение 3 вариантов***





**Основные выводы:**

Многопоточная версия (MT) демонстрирует выдающиеся результаты:

* + При N=4 достигает 2181 MB/s общей производительности
  + Эффективность масштабирования остается высокой: 88% при N=2, 82% при N=4
  + Небольшое падение per-thread производительности (-17% при N=4)

Многопоточная реализация явно превосходит одно- и многопроцессные подходы для последовательной записи

Проблема масштабирования в O3 и O0 связана, вероятно, с:

* + Конкуренцией за блокировки файловой системы
  + Неэффективным управлением кэшем при многопроцессном доступе
  + Накладными расходами на синхронизацию между процессами

**Описание нагрузки: write, sequence, direct=off**

**Тестируемая программа:** io — нагрузчик записи (sequence, O\_DIRECT = off)  
**Режим:**  
./io-mt write -b 1048576 -c 1024 -f <file> -t sequence -d off -R 1 -T <threads>  
(1 блок = 1 MiB, -c 1024 → 1 073 741 824 байт = 1 GiB записывается каждым экземпляром)  
**Количество запусков:** 5 повторений для каждого N; для каждого повторения запускалось N параллельных экземпляров (N = 1,2,4).

**1 процесс**

• USER%: среднее = 0.43% ± 0.12% (min = 0.30%, max = 0.59%)  
• SYS%: среднее = 18.21% ± 8.70% (min = 8.41% – max = 29.88%)  
• IOWAIT%: среднее = 13.24% ± 8.90% (min = 0.30% – max = 22.95%)  
• Полное (wall-clock) время (пер-процесс): среднее = 0.88 с ± 0.44 с (min = 0.80 с, max = 1.91 с)  
• Переключения контекста (суммы по прогонам): voluntary = [3, 32, 17, 74, 27], involuntary = [4, 4, 2, 2, 3]  
— per-proc MB/s: 1079.18 ± 307.54 (min = 535.73 – max = 1284.11) MB/s  
— total MB/s: 1079.18 ± 307.54 MB/s  
(по прогонам per-proc/total: 1215.17, 1211.93, 1148.96, 535.73, 1284.11 MB/s)

**2 процесса**

• USER%: среднее = 0.57% ± 0.34% (min = 0.28% – max = 1.08%)  
• SYS%: среднее = 35.33% ± 11.03% (min = 20.86% – max = 40.51%)  
• IOWAIT%: среднее = 13.29% ± 8.65% (min = 0.00% – max = 28.97%)  
• Полное (wall-clock) время (пер-процесс) (всего 10 измерений — 2 процесса × 5 прогона) — 1.52 с ± 0.77 с (min = 0.91 с, max = 3.02 с)  
• Переключения контекста (суммы по прогонам): voluntary = [144, 147, 309, 159, 261], involuntary = [7, 7, 13, 6, 7]  
— per-proc MB/s (среднее по всем 10 экземплярам): 772.94 ± 323.68 (min = 338.81 – max = 1149.94) MB/s  
— total MB/s (по прогонам, затем среднее): 1534.77 ± 676.72 (min = 677.61 – max = 2256.89) MB/s  
(по прогонам total throughput: 2256.89, 1745.46, 677.61, 2010.94, 982.95 MB/s)

**4 процесса**

• USER%: среднее = 0.59% ± 0.16% (min = 0.35% – max = 1.01%)  
• SYS%: среднее = 73.07% ± 10.29% (min = 55.99% – max = 84.12%)  
• IOWAIT%: среднее = 3.31% ± 3.02% (min = 0.00% – max = 7.28%)  
• Полное (wall-clock) время (пер-процесс) (всего 20 измерений — 4 процесса × 5 прогона) — 1.67 с ± 0.62 с (min = 1.14 с, max = 2.22 с)  
• Переключения контекста (суммы по прогонам): voluntary = [2097, 2206, 2346, 1790, 2037], involuntary = [53, 29, 42, 56, 74]  
— per-proc MB/s: 660.34 ± 173.81 (min = 461.55 – max = 910.55) MB/s  
— total MB/s: 2621.58 ± 745.40 (min = 1846.21 – max = 3591.09) MB/s  
(по прогонам суммарные throughputs: 1846.21, 3107.42, 3591.09, 2608.82, 1954.38 MB/s — среднее и отклонение показаны выше)

**Общая таблица по каждому процессу**

| **N** | **avg wall (s) ± sd (min–max)** | **%user** | **%sys** | **%iowait** | **%idle** | **Involuntary CSW (по прогонам)** | **MB/s per-proc** | **MB/s total** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0.88 ± 0.44 (0.80–1.91) | 0.43 | 18.21 | 13.24 | 69.32 | [4,4,2,2,3] | 1079.18 ± 307.54 (535.73–1284.11) | 1079.18 ± 307.54 |
| 2 | 1.52 ± 0.77 (0.91–3.02) | 0.57 | 35.33 | 13.29 | 52.61 | [7,7,13,6,7] | 772.94 ± 323.68 (338.81–1149.94) | 1534.77 ± 676.72 |
| 4 | 1.67 ± 0.62 (1.14–2.22) | 0.59 | 73.07 | 3.31 | 25.40 | [53,29,42,56,74] | 660.34 ± 173.81 (461.55–910.55) | 2621.58 ± 745.40 |

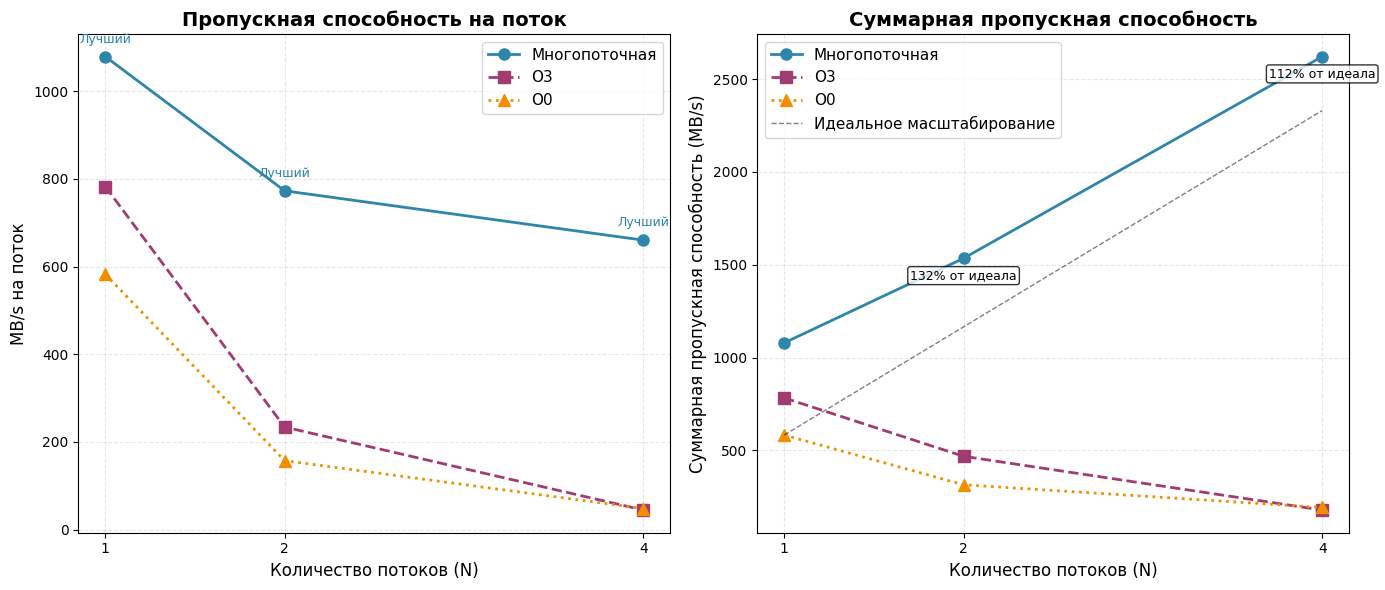
**Таблица эффективности масштабирования**

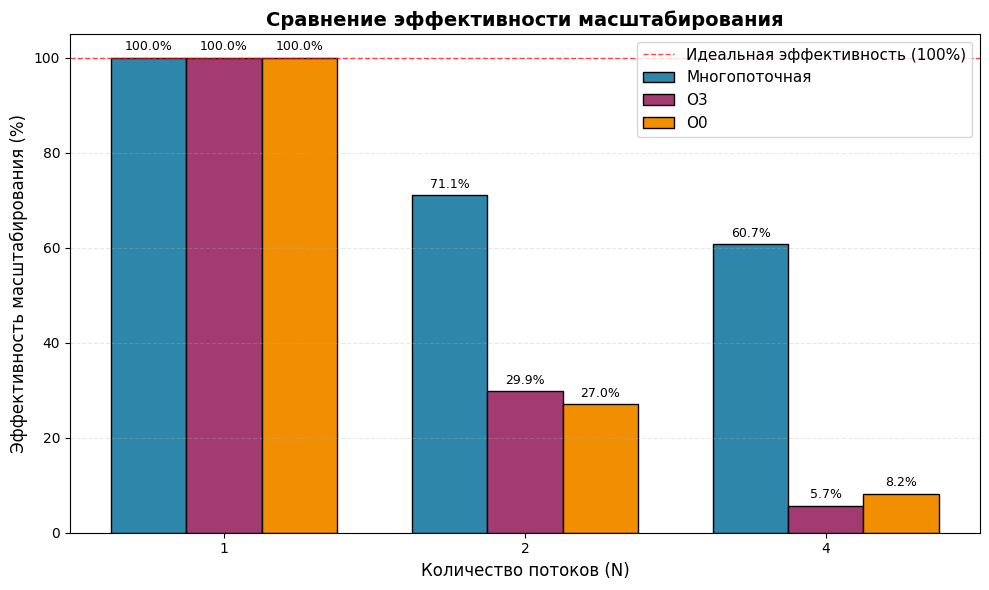
| **N** | **MB/s per-proc** | **MB/s total** | **Масштаб. эфф. (%)** |
| --- | --- | --- | --- |
| 1 | 1079.18 | 1079.18 | **100.00 %** |
| 2 | 772.94 | 1534.77 | **71.11 %** |
| 4 | 660.34 | 2621.58 | **60.73 %** |

**Вывод**

* Высокая вариативность результатов: видно несколько выбросов (особенно для N=1 — один прогон сильно медленнее: 535.7 MB/s), поэтому средние и отклонения довольно большие.
* Per-thread: максимальная одиночная скорость наблюдается при N=1 (1079 MB/s в среднем), но вариативность велика — вероятна взаимосвязь с фоновой загрузкой/кешированием/поведением контроллера.
* Суммарная пропускная способность растёт с N (в среднем максимальна при N=4 ≈ 2.62 GB/s), но эффективность масштабирования падает (≈71% при N=2 и ≈61% при N=4 относительно идеала), что указывает на частичную конкуренцию за ресурсы (ядро/контроллер диска/шина).

***Сравнение 3 вариантов***





**Краткий вывод**

Многопоточная реализация демонстрирует наилучшее масштабирование и значительно превосходит версии O0 и O3 по суммарной пропускной способности при росте числа потоков.

Оптимизация O3 улучшает однопоточную производительность, но плохо масштабируется и быстро теряет эффективность.

# Заключение

В ходе выполнения работы были исследованы различные типы вычислительных нагрузок, включая вычислительно-ориентированные, ориентированные на ввод-вывод, а также комбинированные сценарии. Для каждого типа нагрузки был проведён экспериментальный анализ производительности при изменении числа параллельно выполняемых процессов (потоков), что позволило оценить масштабируемость и эффективность использования системных ресурсов.

Эксперименты подтвердили ожидаемое поведение системы: при малом числе потоков наблюдается близкий к линейному рост суммарной производительности, тогда как дальнейшее увеличение степени параллелизма приводит к снижению эффективности масштабирования. Это связано с ростом накладных расходов на управление потоками, конкуренцией за вычислительные ресурсы, кэш процессора и пропускную способность подсистемы памяти и ввода-вывода.

Дополнительно было изучено влияние оптимизаций компилятора на производительность программ. Использование оптимизирующих режимов позволило улучшить однопоточную производительность и частично снизить время выполнения вычислений, однако в условиях высокой параллельности влияние оптимизаций уменьшалось из-за аппаратных ограничений и синхронизационных издержек.