

Принципы построения и работы баз данных

Тема 02: Технические средства и их характеристики

Основные вопросы

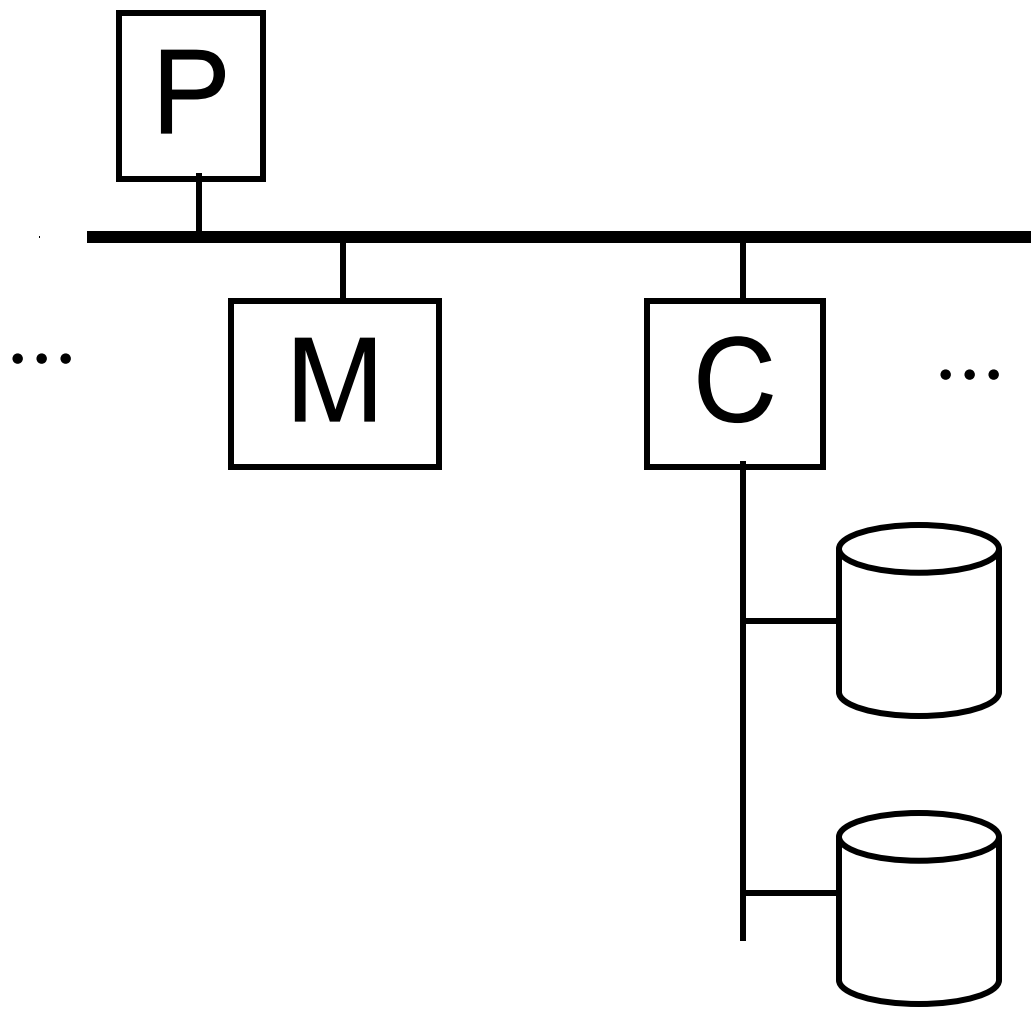
- Оборудование: Дисковая память
- Время доступа
- Пример - Megatron 747
- Оптимизация
- Другие вопросы:
 - Стоимость памяти
 - Использование вторичной памяти
 - Сбои дисков

Оборудование

СУБД

Хранилище данных





Типичный
компьютер

Вторичная
память

Процессор

Быстрый, медленный, RISC/CISC,
со встроенным кэшем, конвейерный, ...
Speed: 100 → 500 → 1000 MIPS

Оперативная память

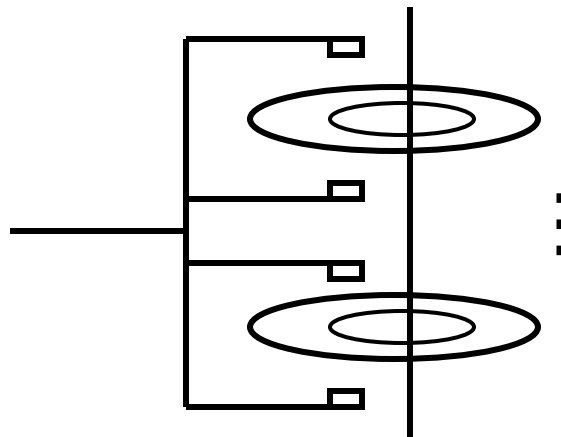
Быстрая, медленная, энергонезависимая,
только для чтения (ROM), ...
Время доступа: $10^{-6} \rightarrow 10^{-9}$ sec.
 $1 \mu\text{s} \rightarrow 1 \text{ ns}$

Вторичная память

Несколько видов:

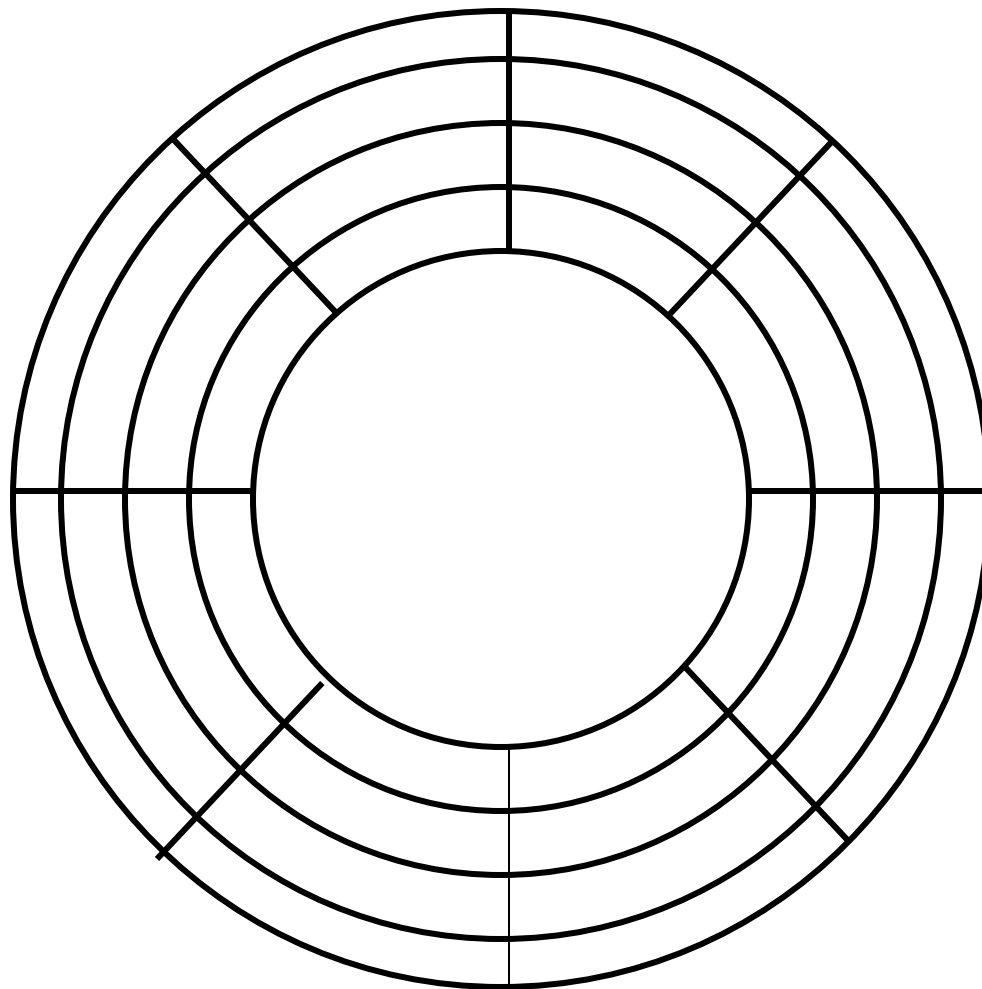
- Диск: Сменные (жесткие, мягкие)
Сменные пакеты
Обычные жесткие диски с
произвольным доступом
Оптические, CD-ROM
Дисковые массивы
- Ленты: катушечные, кассетные
- Роботизированные хранилища

Основное внимание: “Типичный диск”



Термины: поверхность, головка,
цилиндр, дорожка,
сектор (физический),
блок (логический), промежуток,

Вид сверху



“Типичные” значения

Диаметр: 1 дюйм → 15 дюймов

Цилиндров: 100 → 2000

Поверхностей: 1 (CDs) →

(дор./цил.) 2 (флоппи) → 30

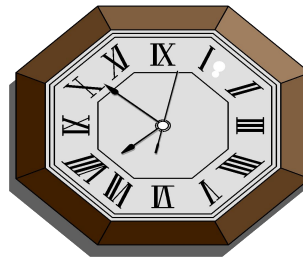
Размер сектора: 512В → 50К

Общая емкость: 360 КВ (флоппи)

→ 120 GB

Время доступа

Запрос
на блок X

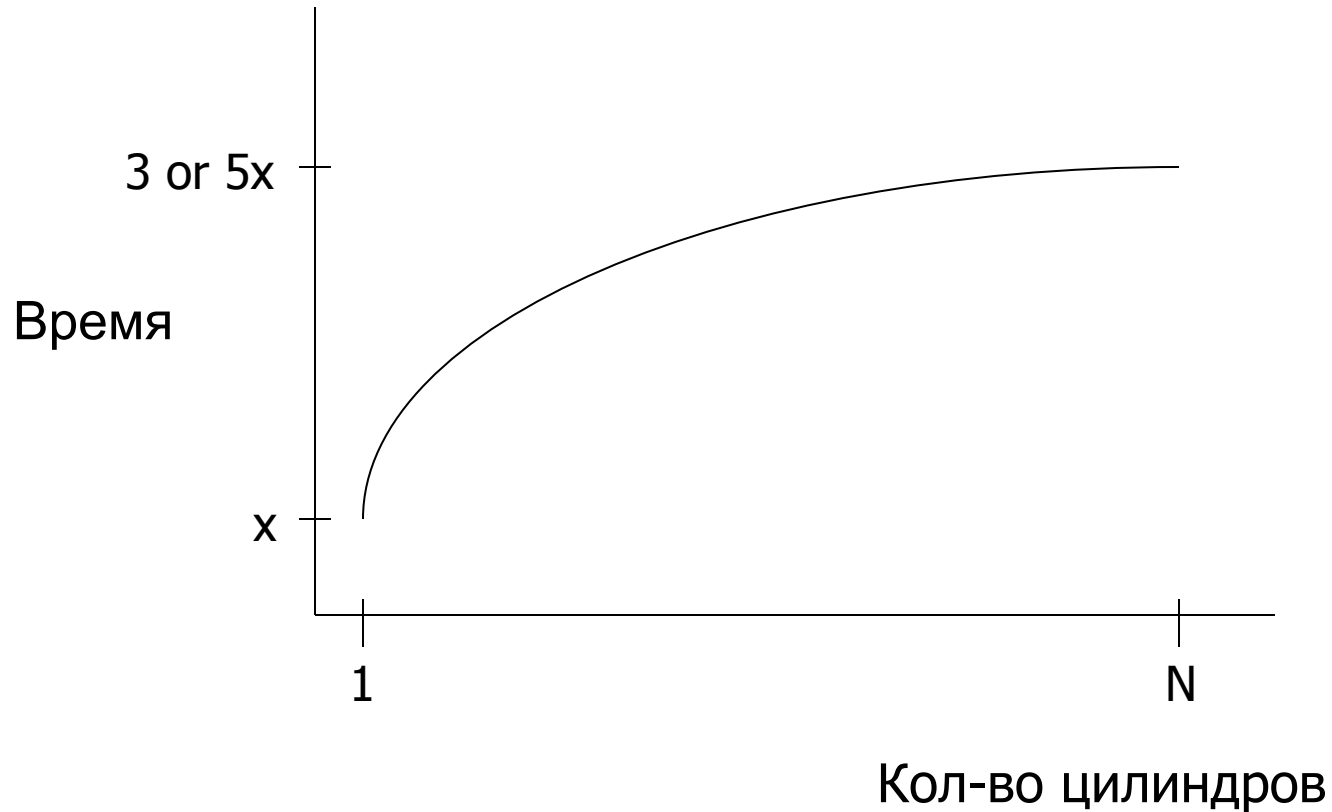


Блок X в
оп.памяти



Время = перемещение головок +
задержка вращения +
передача блока +
другое

Перемещение головок (время поиска)

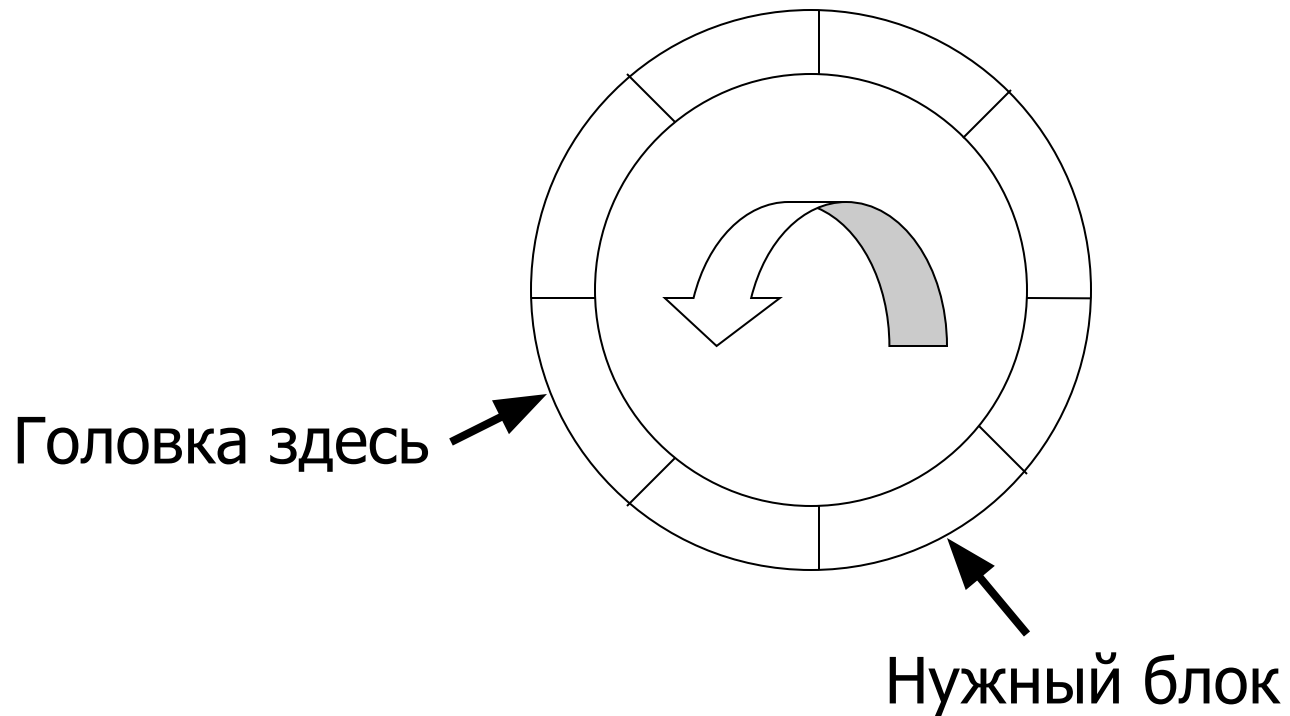


Среднее время поиска

$$S = \frac{\sum_{i=1}^N \sum_{\substack{j=1 \\ j \neq i}}^N \text{SEEKTIME} (i \rightarrow j)}{N(N-1)}$$

“Типичное” S: 5 мсек → 20 мсек

Задержка вращения



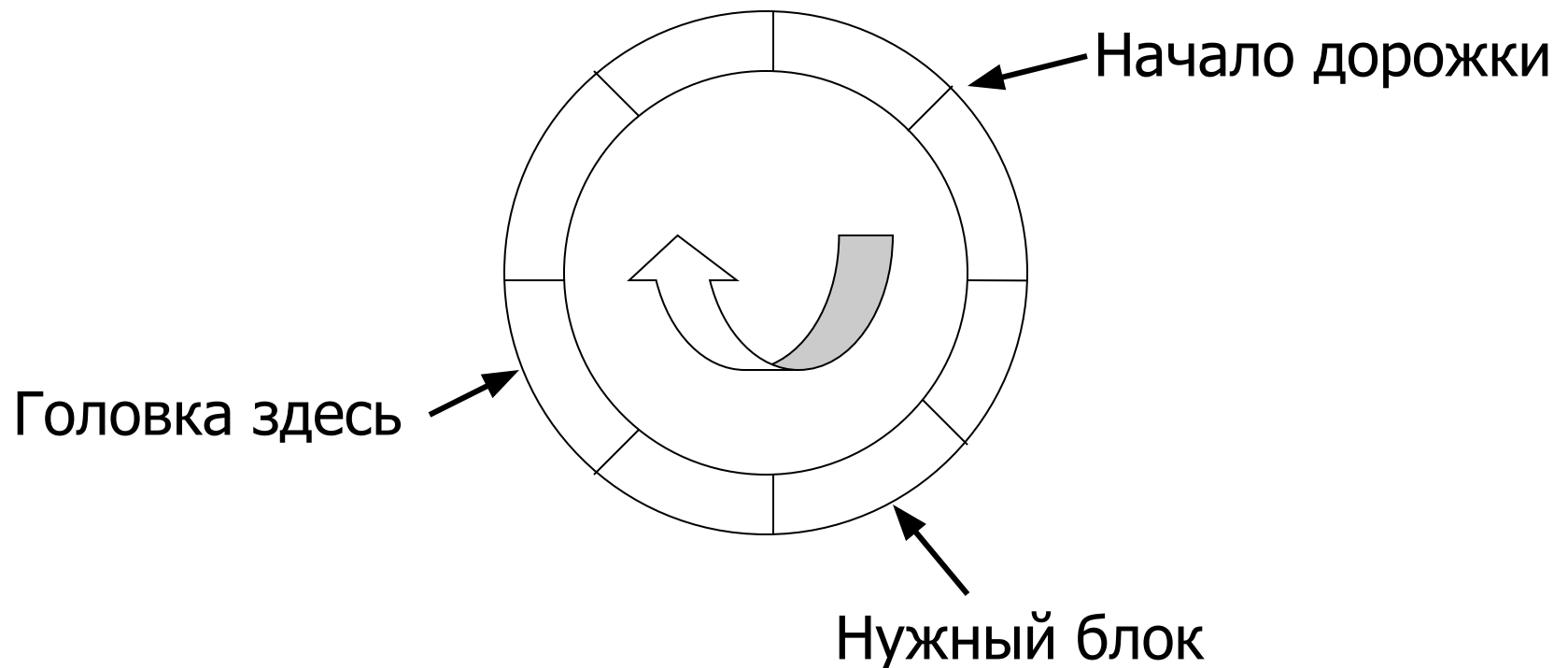
Средняя задержка вращения

$R = 1/2$ оборота диска

“типичное” $R = 4.167$ мсек (7200 об/мин)

Дополнительная сложность

- необходимо определить начало дорожки прежде, чем можно читать нужный блок



Скорость передачи: t

- “типичное” t : 60 → 100 МВ/сек
- время передачи: размер блока
 t

Другие задержки

- ожидание доступности процессора для выдачи команды I/O
- ожидание доступности контроллера
- ожидание доступности шины, памяти

“Типичные” значения : 0

- До сих пор речь шла о произвольном (случайном) доступе
- Что изменится при чтении “следующего” блока?

При эффективной организации I/O
(двойной буферизации и других приемах)

Время получения = размер блока + пренебрежимо
блока t малое время

- пропустить промежуток
- перейти на другую дорожку цилиндра
- время от времени перейти на
следующий цилиндр



Главное правило	Произвольный (случайный) доступ гораздо медленнее последовательного
------------------------	--

- Пример: 1 KB Block
 - » случайный I/O: ~ 10 мсек.
 - » последовательный I/O: ~ 0.5 мсек.

Время записи то же, что и время
чтения

За исключением записи с проверкой!
надо добавить (полный) оборот +
размер блока
t

Для модификации блока:

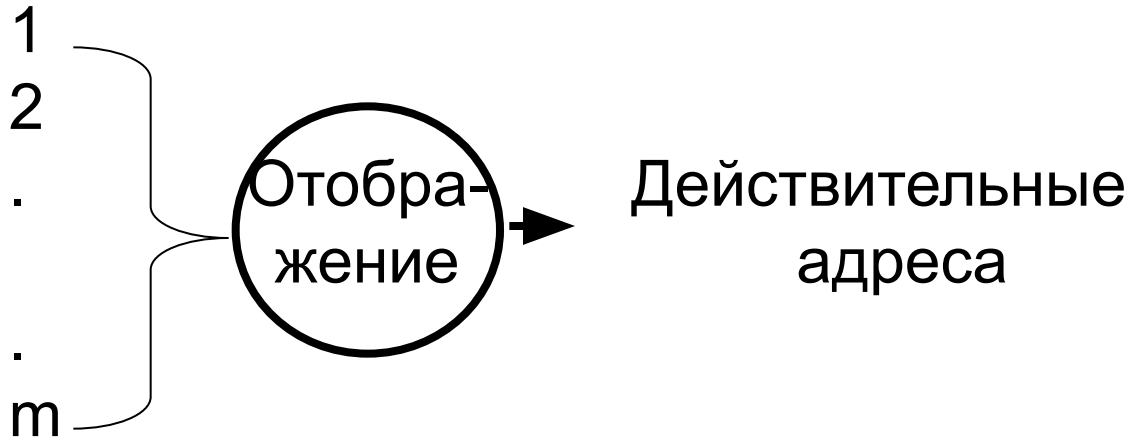
- (a) прочитать блок
- (b) модифицировать в памяти
- (c) записать блок
- [(d) проверить?]

Адрес блока:

- Номер устройства
- Номер цилиндра
- Номер поверхности
- Номер сектора

Осложнения: дефектные блоки

- Сложно обрабатывать
- Может программно отображаться на последовательность целых чисел



Пример

Megatron 747 диск (устаревший)

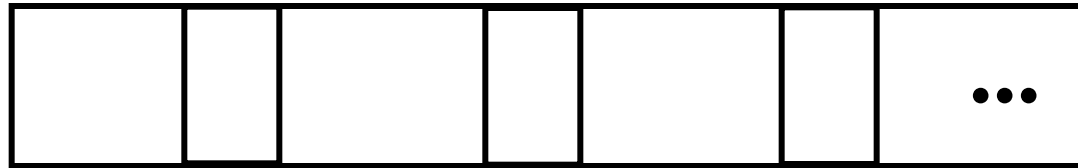
- Диаметр 3.5 дюйма
- 3600 об/мин
- 1 поверхность
- 16 МВ полезная емкость (16×2^{20})
- 128 цилиндров
- Время поиска: среднее = 25 мсек.
соседний цилиндр = 5 мсек.

- Размер сектора = 1 КВ = 1 блок
- 10% дорожки составляют межблочные промежутки
- емкость = 16 МВ = $(2^{20})16 = 2^{24}$
- число цилиндров = 128 = 2^7
- байт/цил. = $2^{24}/2^7 = 2^{17} = 128$ КВ
- блок/ цилиндр. = 128 КВ / 1 КВ = 128

3600 об/мин 60 оборотов / сек

1-об. = 16.66 ms

Дорожка:



Время над полезными данными: $(16.66)(0.9) = 15$ мсек.

Время над промежутками: $(16.66)(0.1) = 1.66$ мсек.

Время передачи 1 блока = $15/128 = 0.117$ мсек.

Время перед.1 блока+промеж. = $16.66/128 = 0.13$ мсек.

Макс.скорость передачи = $1/0.117 = 8.53$ блок/мсек =

$8.53 * 1KB * 1000 \text{ мсек/сек} * 1MB/1024KB = 8.33MB/сек$

Реальная скорость передачи (дорожки – 128KB за 16.66 мсек) = $128/16.66 = 7.68$ KB/мсек = 7.5MB/сек

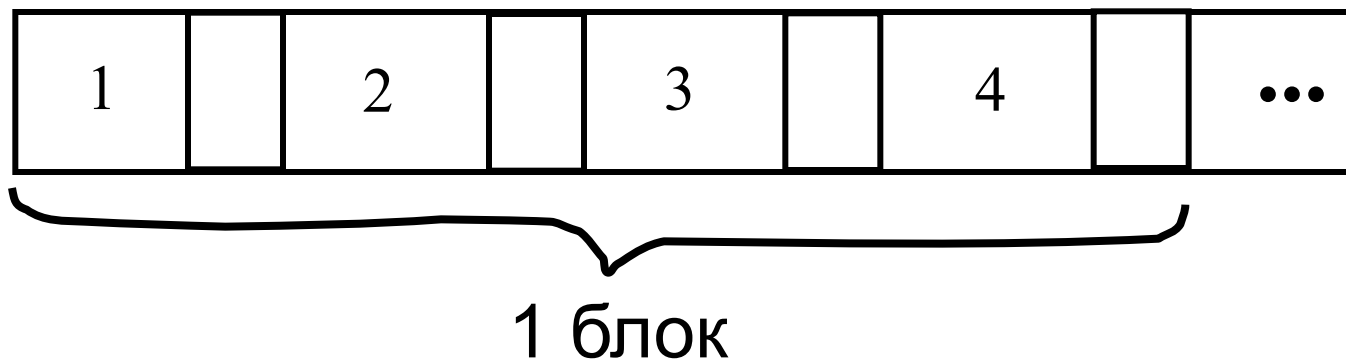
T_1 = Время чтения случайного блока

T_1 = поиск + задержка вращения + передача блока

$$= 25 + (16.66/2) + .117 = 33.45 \text{ мсек.}$$

↙
предполагая, что мы не ждем
начала дорожки

Допустим, что ОС работает с блоками размером 4 КВ




$$T_4 = 25 + (16.66/2) + (.117) \times 1 \\ + (.130) \times 3 = 33.83 \text{ мсек}$$

[Сравните с $T_1 = 33.45 \text{ ms}$]

TT = Время чтения всей дорожки
(начиная с любого блока)

$$TT = 25 + (0.130/2) + 16.66^* = 41.73 \text{ ms}$$


попасть на начало первого блока

- * На самом деле несколько меньше; не нужно читать последний промежуток

Новый Megatron 747 (Пример 11.3)

- $2^4 = 16$ поверхностей, 3.5 дюйма диаметр
 - используется только внешний 1 дюйм
- $2^{14} = 16384$ дорожек/поверхность
- $2^7 = 128$ (в среднем) секторов/дорожку
- $2^{12} = 4096$ байт/сектор
- $2^{37} = 128\text{GB}$ – общая емкость
- $2^{19} = 512\text{KB}$ – емкость 1 дорожки
- если все дорожки по 128 секторов, то
 - плотность самой внешней дорожки ~ 420000 бит/дюйм
 - плотность самой внутр. дорожки ~ 990000 бит/дюйм
- Если число секторов увел. с 96 (на внутр.) до 160 (на внеш.), то плотность вырастет с 530000 до 742000 бит/дюйм
- Скорость вращения – 7200 об/мин

Временные характеристики нового Megatron 747 (Пример 11.5)

- Допустим, что для разгона и остановки блока головок требуется 1 мсек плюс 1 мсек на движение на 1000 цилиндров(в ту или другую сторону)
- Максим. время поиска $1+16.383 = 17.383$ мсек
- Время полного оборота $= 60/7200 = 0.00833$ сек
- Угл. радиус 16384-байт блока $36*3/128+324*4/128 = 10.97$
- Время чтения 16384-байт блока:
 - Мин.: $10.97/360*8.33 = 0.25$ мсек (без задержки вращ.)
 - Мах: $17.38+8.33+0.25 = 25.96$ мсек
 - Среднее: $1+ 5.641+4.17+0.25 = 10.88$ мсек
(5641 – среднее кол-во цилиндров между 2 случайными)

Способы оптимизации (контроллер, ОС)

- Алгоритмы упорядочения
 - Например, алгоритм «лифта»
- Использование буфера размера дорожки диска (или более)
- Предварительное чтение блока (до запроса)
- Дисковые массивы
- Зеркальные диски

Двойная буферизация

Задача: Имеется файл

Последовательность блоков В1, В2,...

Программа

Обработать В1

Обработать В2

Обработать В3

Использование одного буфера

(1) Читать В1 → Буфер

(2) Обработать данные в буфере

(3) Читать В2 → Буфер

(4) Обработать данные в буфере

...

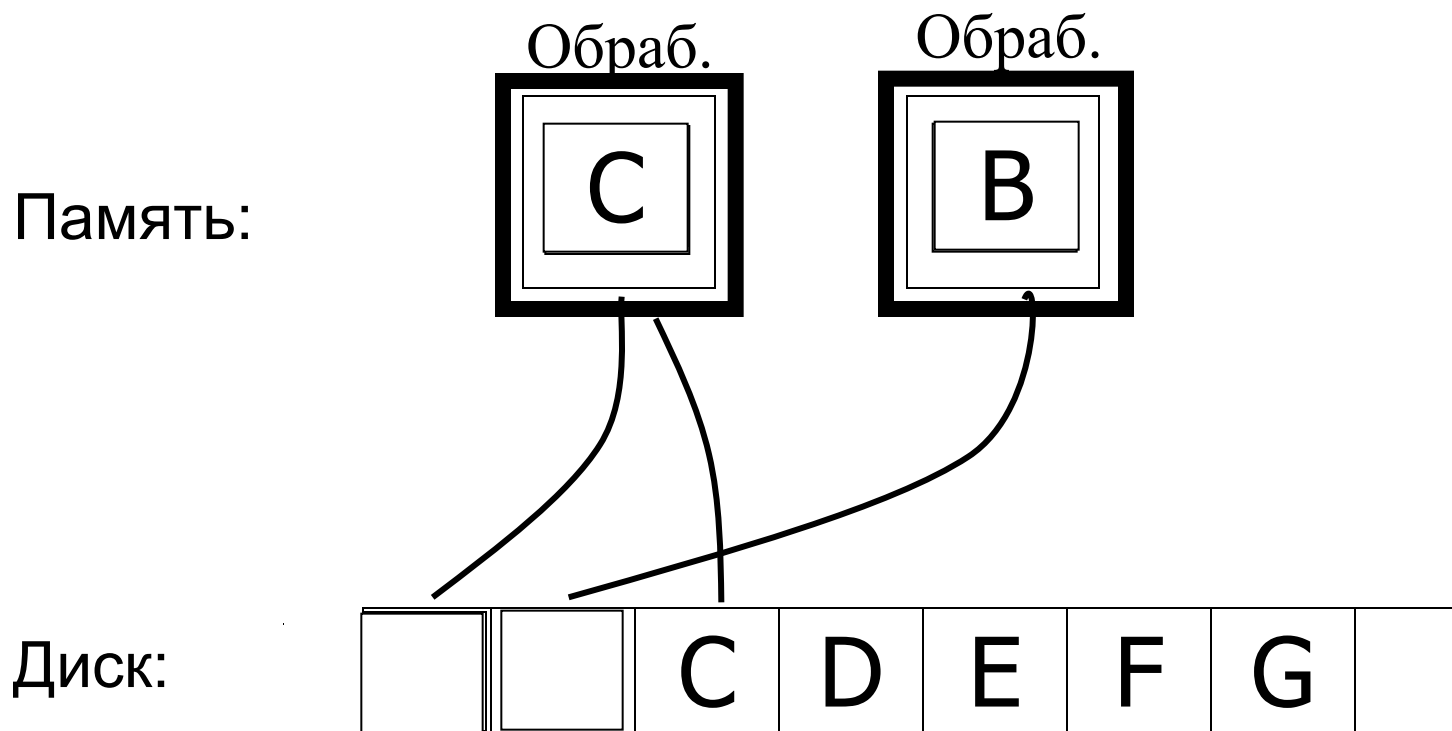
Пусть P = время обработки 1 блока

R = время чтения 1 блока

n = кол-во блоков в файле

Время обработки с одним буфером = $n(P+R)$

Двойная буферизация



Пусть $P \geq R$

P = время обработки 1 блока

R = время чтения 1 блока

n = кол-во блоков в файле

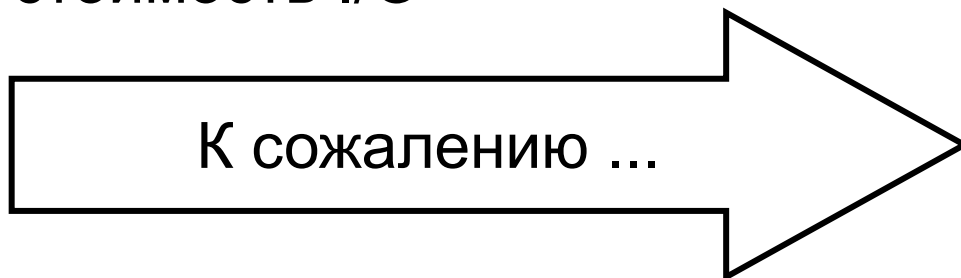
Каково время обработки?

Время обработки с 2 буферами = $n(P+R)$

Время обработки с 1 буфером = $n(P+R)$

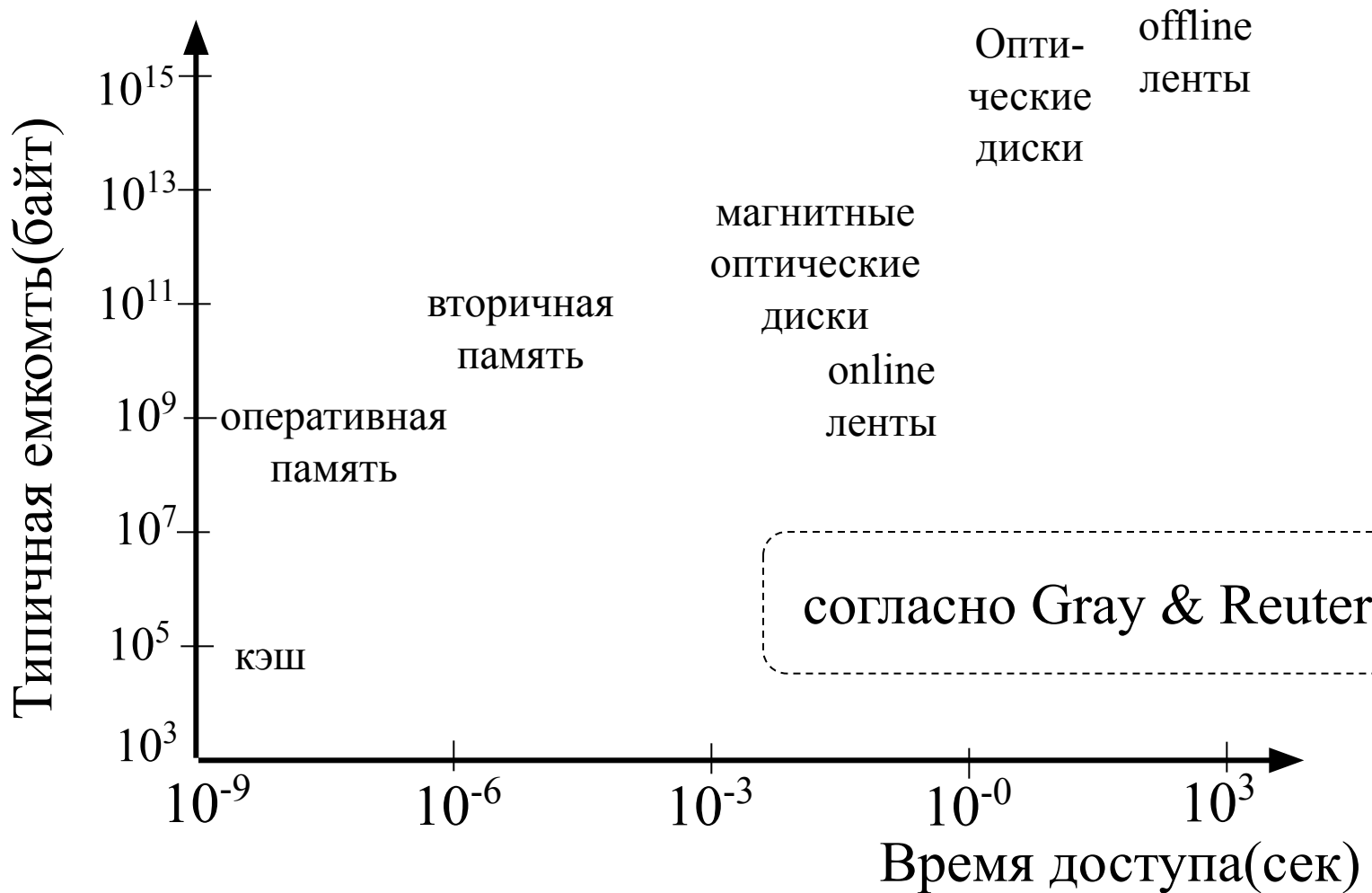
Выбор размера блока

- Большой блок → уменьшает относительную стоимость I/O

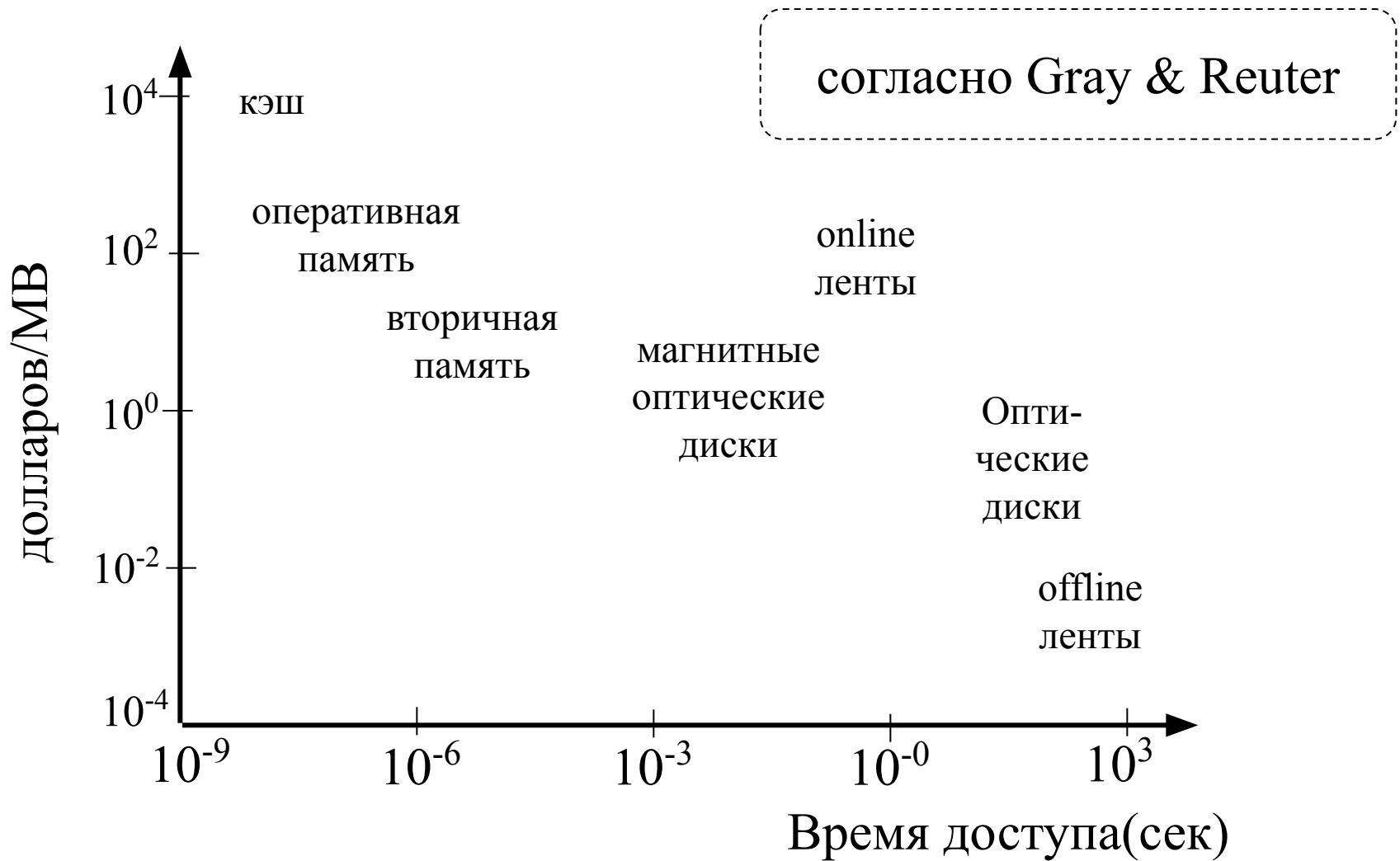


- Большой блок \Rightarrow приходится читать много бесполезной информации, больше затраты на чтение
- Тенденция – по мере удешевления памяти размер блоков увеличивается

Стоимость памяти



Стоимость памяти



Эффективное использование вторичной памяти (Раздел 11.4)

- Пример: внешняя сортировка
- Заключение:
 - Стоимость I/O доминирует
 - Необходимость разработки алгоритмов возможности уменьшающих I/O
- Каков должен быть размер блоков?

Сбои дисков (Раздел 11.6)

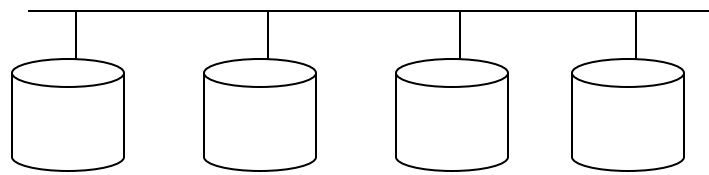
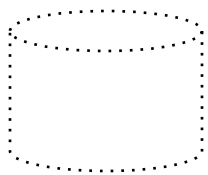
- Частичные → Полные
- Временные → Постоянные

Что делать в таких случаях?

- Детектирование
 - Например, используя контрольные суммы
- Коррекция
 - Использование избыточности

На каком уровне можно бороться со сбоями дисков?

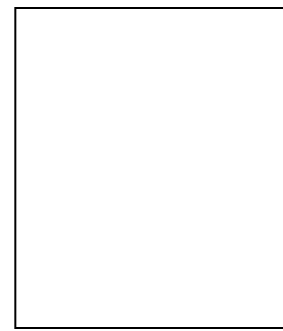
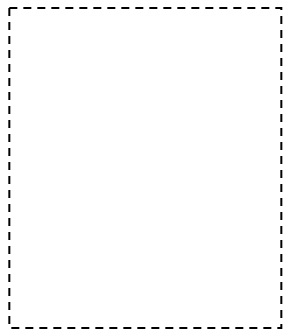
- Отдельный диск
 - Используя коды, корректирующие ошибки
- Дисковые массивы



Логический диск

Физические диски

Уровень ОС, дублирование данных



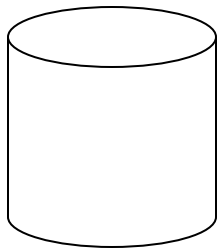
Логический блок

Копия А

Копия В

Системы баз данных

- Например,

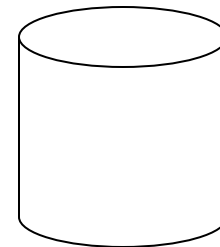


Текущая БД



Файл протокола

Недельная



копия БД

Итог

- Вторичная память, в основном, дисковая
- Характеристики (время) I/O
- Операций I/O, по-возможности, лучше избегать, особенно случайных (запросы к отдельным случайным блокам в файле или БД).