Multithread conveyor

Sundukov Andrey

О себе

- Андрей
- SRE Engineer at Natera (USA)
  Trainer / Consultant at Digital Habits (Спб)
- Core skills: Java, Multithreading, clean code, databases, networks, general computer science
- Первый IT опыт: инженера тех.поддержки в 2004-2005 годах
  С тех пор: инженер датацентра, все стадии PHP и Java разработчика, тимлид, SRE инженер, преподаватель

Помощники и спонсоры

dhabits.ru

@Coding_Java

цифровые привычки

Типовые задачи на многопоточность

- Формирование отчетов
- Парсинг и анализ данных
- Инфраструктурные процессы (деплои, сканирования, тестирования)

Почему “конвейер”?

Конвейерное производство — система поточной организации производства, при которой оно разделено на простейшие короткие операции.

… весь процесс производства разделяется на последовательность стадий с целью повышения производительности путем одновременного независимого выполнения операций над несколькими объектами, проходящими различные стадии.

Почему “конвейер”?

Конвейерное производство — система поточной организации производства, при которой оно разделено на простейшие короткие операции.

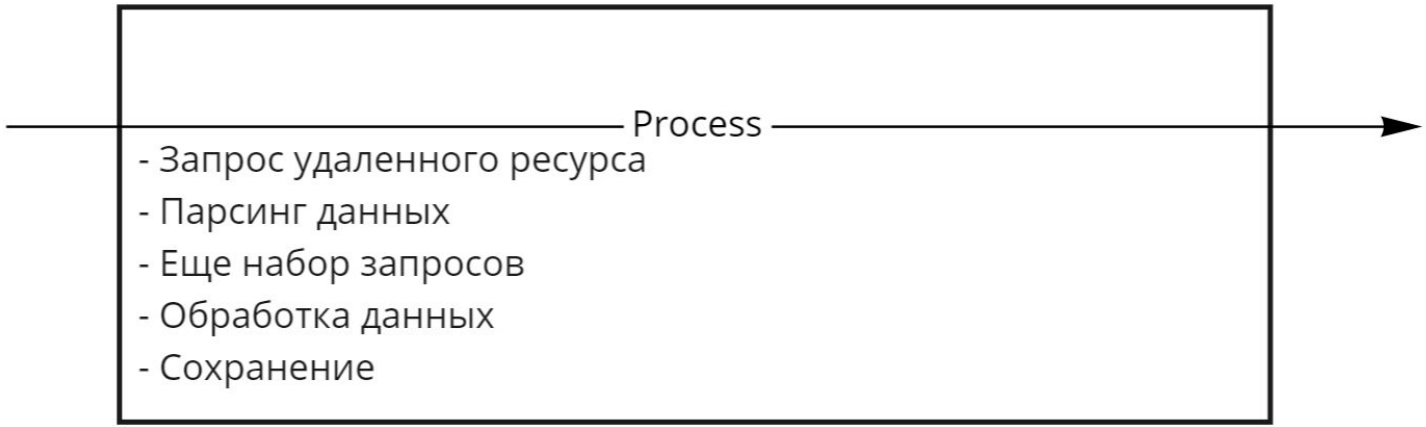
… весь процесс производства разделяется на последовательность стадий с целью повышения производительности путем одновременного независимого выполнения операций над несколькими объектами, проходящими различные стадии.

В чем плюсы конвейера

- Гибкое масштабирование отдельных элементов конвейера
- Хорошая утилизация ресурсов
- Как итог высокая эффективность производства

Как работает “ремесленный подход”

Как работает однопоточный код



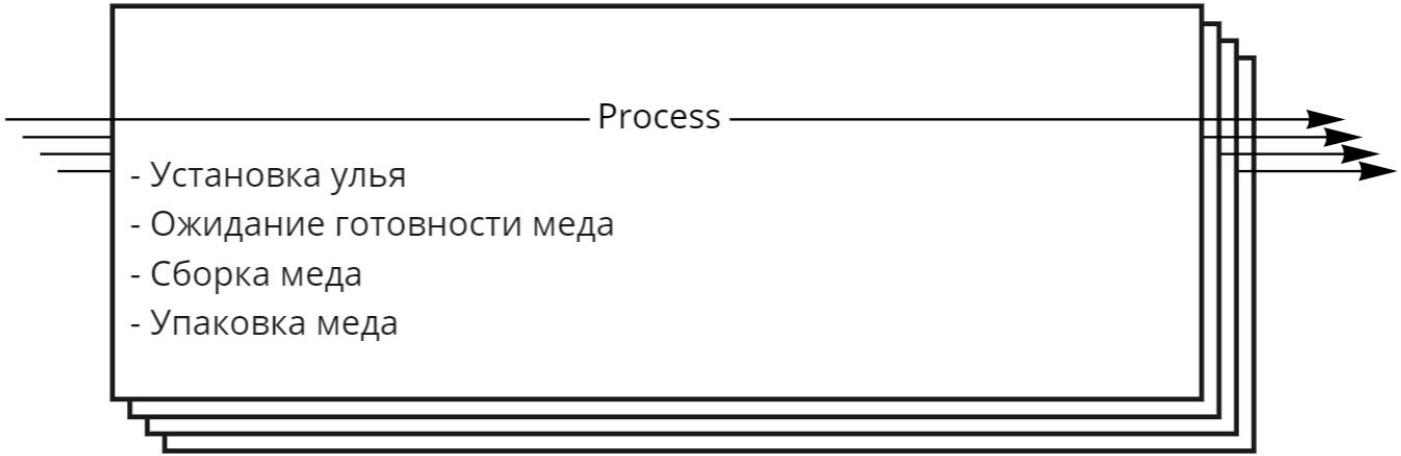
Как работает однопоточный код

```
while (true) {
    DownloadData data = downloadData(nextTask());
    ProcessResult result = process(data);
    saveData(result);
}
```

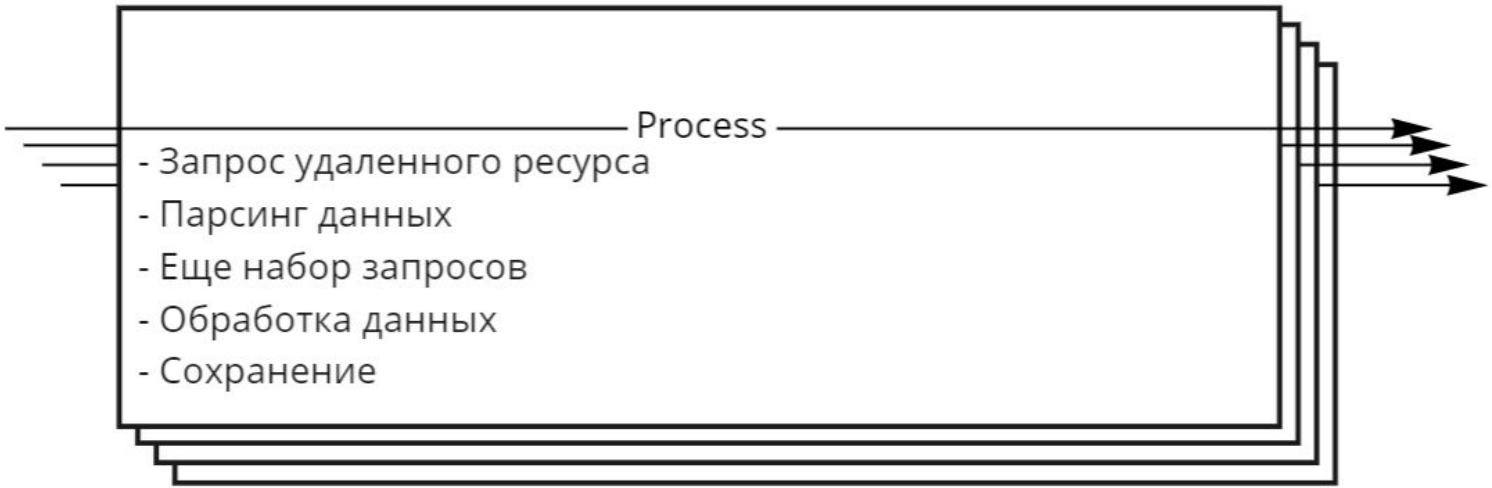
Однопоточный процесс

- Очень просто в реализации
- Вполне достаточен когда нет требований к скорости, нет highload, нет bigdata
- Дешево разрабатывать

Добавим потоков



Добавим потоков



Добавим потоков

```
while (true) {
    new Thread(() -> {
        DownloadData data = downloadData(nextTask());
        ProcessResult result = process(data);
        saveData(result);
    }).start();
}
```

Добавим потоков

```
ExecutorService executorService = Executors.newCachedThreadPool();
while (true) {
    Task task = nextTask();
    executorService.submit(() -> {
    DownloadData data = downloadData(task);
     ProcessResult result = process(data);
        saveData(result);
    });
}
```

Добавим потоков

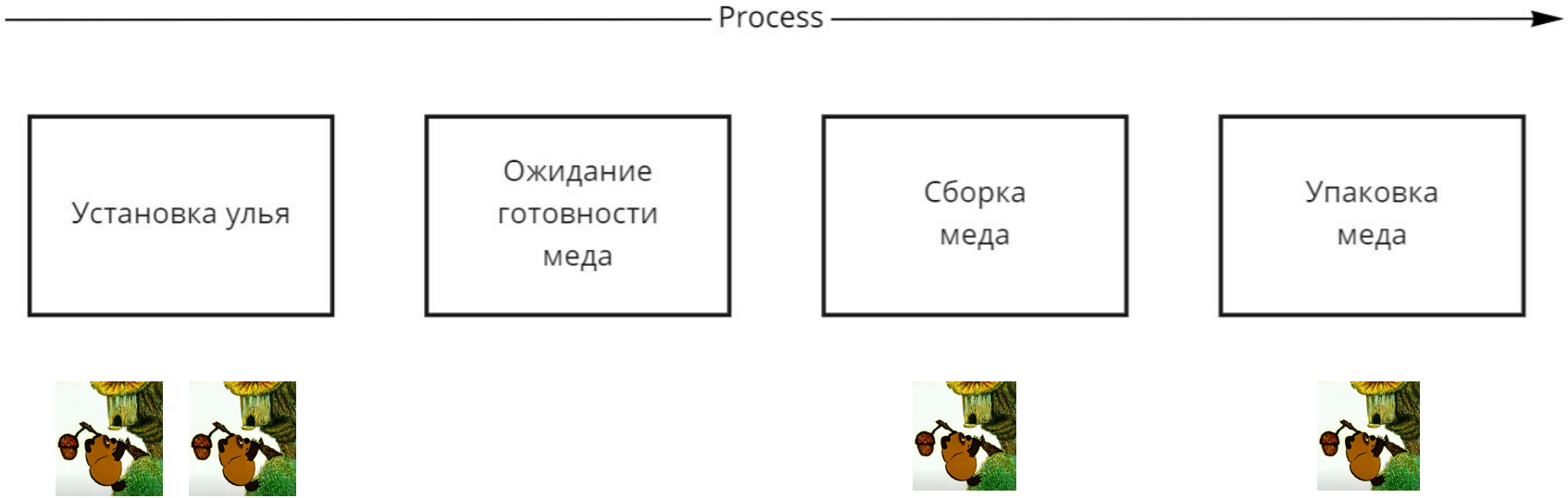
```
ExecutorService executorService
    = Executors.newFixedThreadPool(5); // carefully tuned with magic
while (true) {
    Task task = nextTask();
    executorService.submit(() -> {
    DownloadData data = downloadData(task);
     ProcessResult result = process(data);
      saveData(result);
    });
}
```

Многопоточный процесс

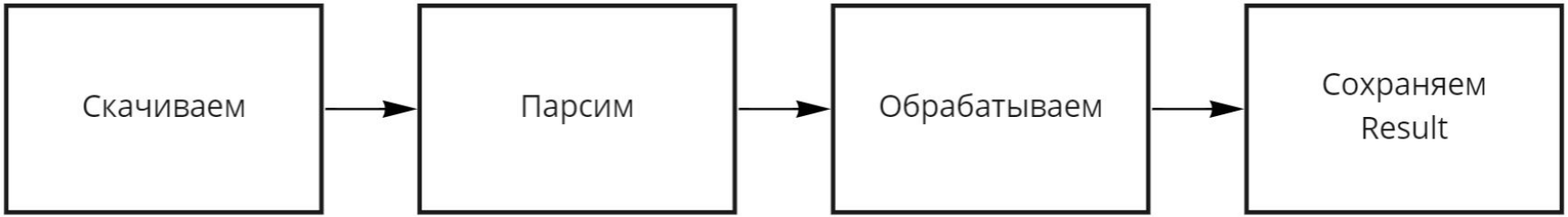
- Просто в реализации
- “Рваная” утилизация ресурсов
- Дешево разрабатывать, дорого выполнять

А что там с конвейером

А что там с конвейером



А что там с конвейером



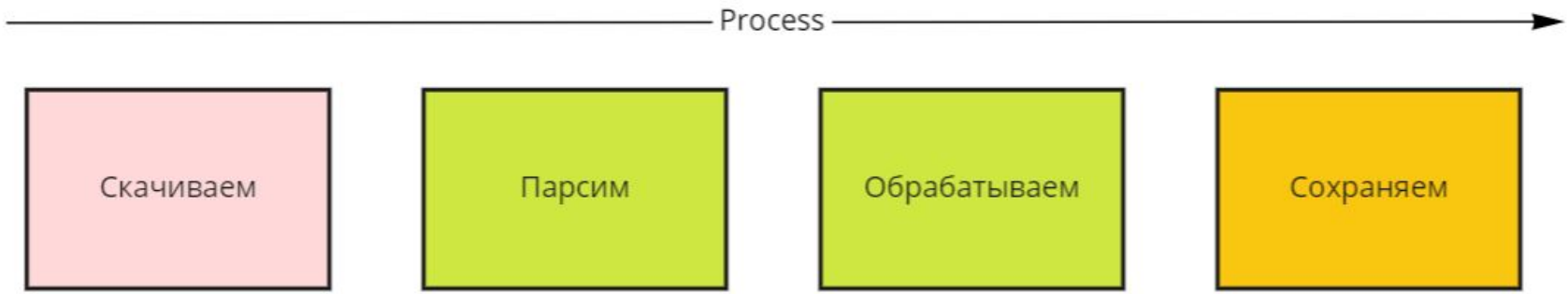
А что там с конвейером

```
ExecutorService executorService = Executors.newFixedThreadPool(CPU_COUNT);

while                                    (true)                                    {
                              Task          nextTask          =          nextTask();

          CompletableFuture<DownloadData>    downloadResultFuture    =    CompletableFuture
                        .supplyAsync(()   ->  this.downloadData(nextTask),  executorService);

          CompletableFuture<ProcessResult>    calcResultFuture    =    downloadResultFuture
                                        .thenApplyAsync(this::process,    executorService);

                                                                          calcResultFuture
                                      .thenAcceptAsync(this::saveData,    executorService);
}
```

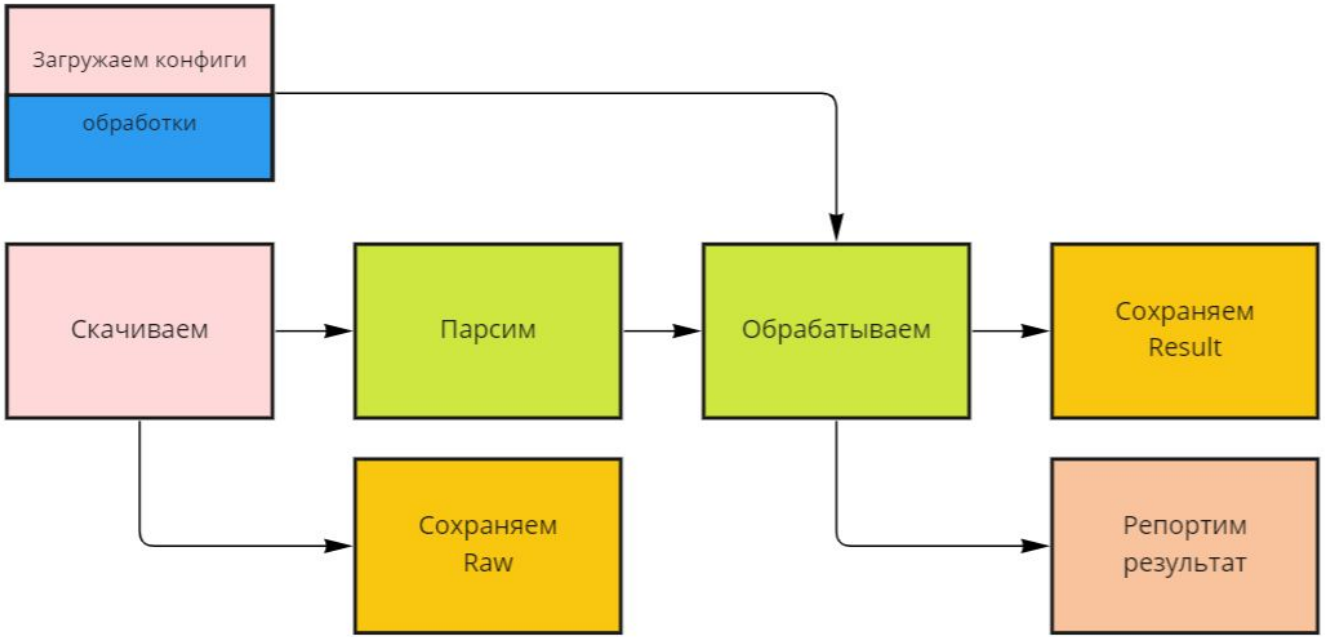
А что там с конвейером

```
ExecutorService downloadExecutorService = Executors.newFixedThreadPool(DOWNLOAD_POOL_SIZE);
ExecutorService  processExecutorService  =  Executors.newFixedThreadPool(PROCESS_POOL_SIZE);
ExecutorService saveDataExecutorService = Executors.newFixedThreadPool(SAVEDATA_POOL_SIZE);
while                                        (true)                                        {
                                Task            nextTask             =            nextTask();

          CompletableFuture<DownloadData>    downloadResultFuture    =    CompletableFuture
                .supplyAsync(()  ->  this.downloadData(nextTask),  downloadExecutorService);

          CompletableFuture<ProcessResult>    calcResultFuture    =    downloadResultFuture
                                    .thenApplyAsync(this::process,    processExecutorService);

                                                                                calcResultFuture
                                  .thenAcceptAsync(this::saveData,   saveDataExecutorService);
}
```

А что там с конвейером



Более сложный вариант



Конвейерный процесс

- Сложнее в реализации
- Равномерная утилизация ресурсов
- Дороже разрабатывать, дешевле выполнять
- Код больше отвечает принципам SOLID (в основном SRP) и становится гибче

Какие вообще бывают ресурсы?

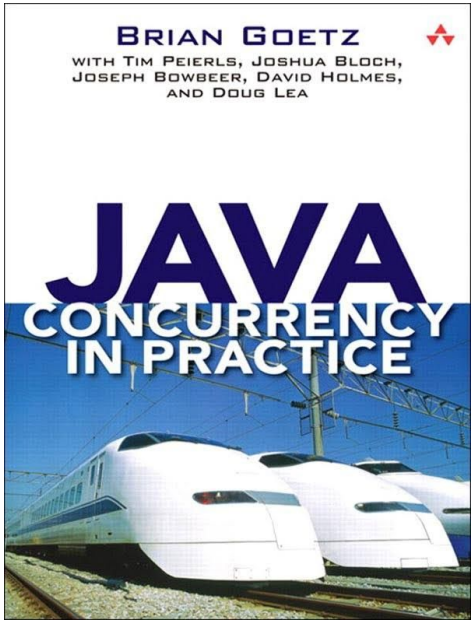
Виды ресурсов

- CPU
- Disk - IOPS & write/read speed
- Network - download/upload speed

CPU

N = number of CPUs
U = target CPU utilization, 0 <= U <= 1
W / C = ratio of wait time to compute time

PoolSize = N * U * (1 + (W/C))



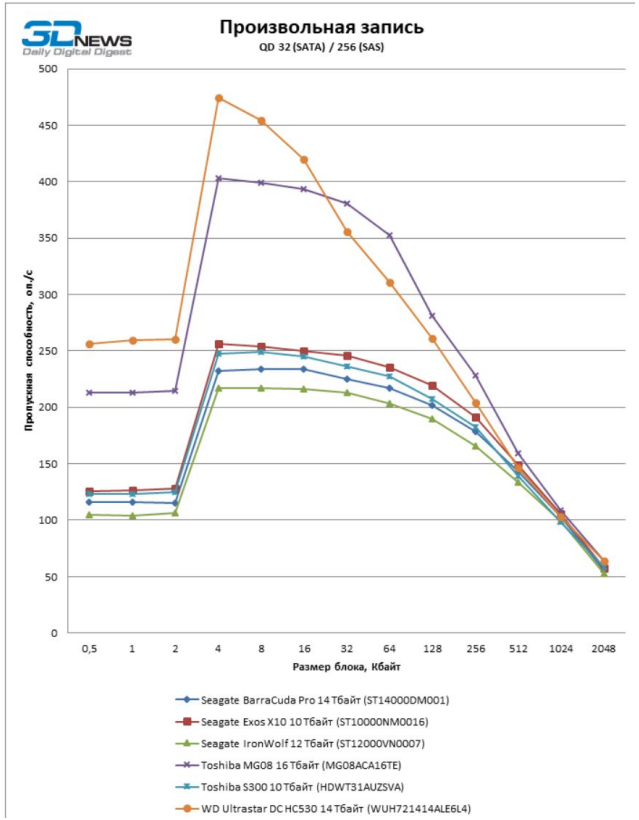
Disk

- IOPS
- Sequence read
- Sequence write

Для маленьких порций данных
PoolSize = disk iops / task iops

Для больших порций данных
В зависимости от типа диска

Disk


3DNEWS
Daily Digital Digest
Произвольная запись
QD 32 (SATA) / 256 (SAS)
Пропускная способность, оп./с
500
450
400
350
300
250
200
150
100
50
0
0,5
1
2
4
8
16
32
64
128
256
512
1024
2048
Размер блока, Кбайт
Seagate BarraCuda Pro 14 Тбайт (ST14000DM001)
Seagate Exos X10 10 Тбайт (ST10000NM0016)
Seagate IronWolf 12 Тбайт (ST12000VN0007)
Toshiba MG08 16 Тбайт (MG08ACA16TE)
Toshiba S300 10 Тбайт (HDWT31AUZSVA)
WD Ultrastar DC HC530 14 Тбайт (WUH721414ALE6L4)

Network

Net speed of local server
Net speed of remote servers

PoolSize = network speed / avg task download speed

Учитываем FULL DUPLEX / HALF DUPLEX

Виды ресурсов

- CPU (+ Memory)
- Disk IO - IOPS
- Disk IO - seq write/read
- Net - UP/DOWN
- Protocols’ limits
- Human

Memory

Для memory-heavy потоков учитываем объем памяти чтоб не вылазить в swap.

PoolSize = memory amount / memory per task

Protocols

TCP/IP и UDP/IP

- Максимум одновременных соединений на один IP адрес
- Размер заголовков

Простой пример

Состав одной задачи

- Загрузка XML-документа - avg 1 mbit / sec
- Парсинг + обработка CPU utilization 100%, wait factor 0%
- Сохранение на локальный диск - 100 Kb seq write

Доступное железо:

- 100 mbit сеть
- 32 CPU
- HDD

Простой пример

Ответ

- 100 Mbit / 1 Mbit = 100 потоков на сеть
- PoolSize = 32 ядра * 100% utilization * (1 + 0% load factor) = 32 потока
- 100 задач в секунду = 100 * 100 Kb = 10 Mb (достаточно одного потока seq write)

Как считать количество потоков

- Понимать ограничения доступных ресурсов
- Понимать издержки слишком большого количества потоков
- Понимать характер нагрузки ваших задач

Специализация воркеров

Установка ульев 100%
Ожидание 100%
Сборка 30%
Упаковка 90%

Установка ульев 30%
Ожидание 100%
Сборка 230%
Упаковка 10%

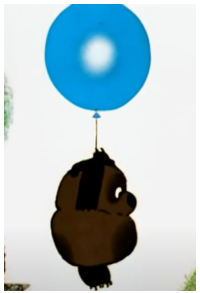
Установка ульев 80%
Ожидание 100%
Сборка 50%
Упаковка 100%

Установка ульев 0%
Ожидание 100%
Сборка 0%
Упаковка 0%

А что там с конвейером

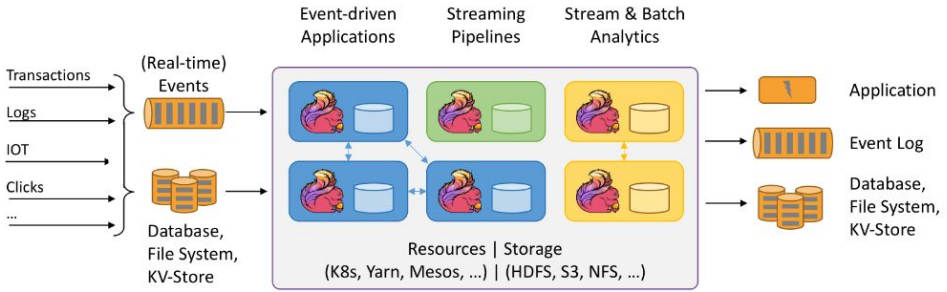
Process

Установка улья	Ожидание готовности меда	Сборка меда	Упаковка меда



Дальнейшее масштабирование

- Узкоспециализированные ноды
- Специальные системы стриминговой обработки данных



А есть смысл что-то оптимизировать?

Оптимизация деплоя Bamboo specs 45 min -> 15 min

~6 деплоев в рабочий день ≈ 1200 в год

Экономия на ожидании разработчиком 30 мин * 1200 = 600 разрабочасов

600 разрабочасов ≈ $18000 в год (с допущениями вверх и вниз)

Подведем итоги

- Любой медленный код можно ускорить конвейером?
- Можно ли просто добавить потоков если этого достаточно?
- Применимо ли это к real-time задачам? Например, запрос в вебе от браузера
- Звучит сложно, это точно выгодно?
- Правда надо учитывать детали типа характера дисков?

Вопросы и розыгрыш футболок

Спасибо за внимание

Sundukov Andrey

TG channel: @developers_mind