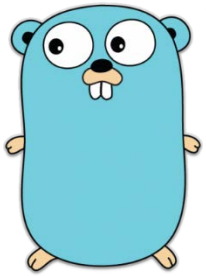
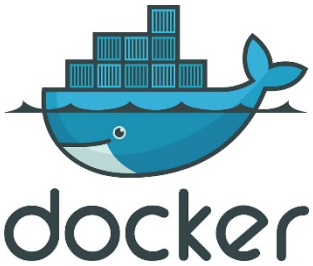




Конвейер bowtie-based анализа метагеномных данных v3



2018

- Volume (объем)
- Variety (разные типы)
- Velocity (скорость генерации)
- Veracity (достоверность)

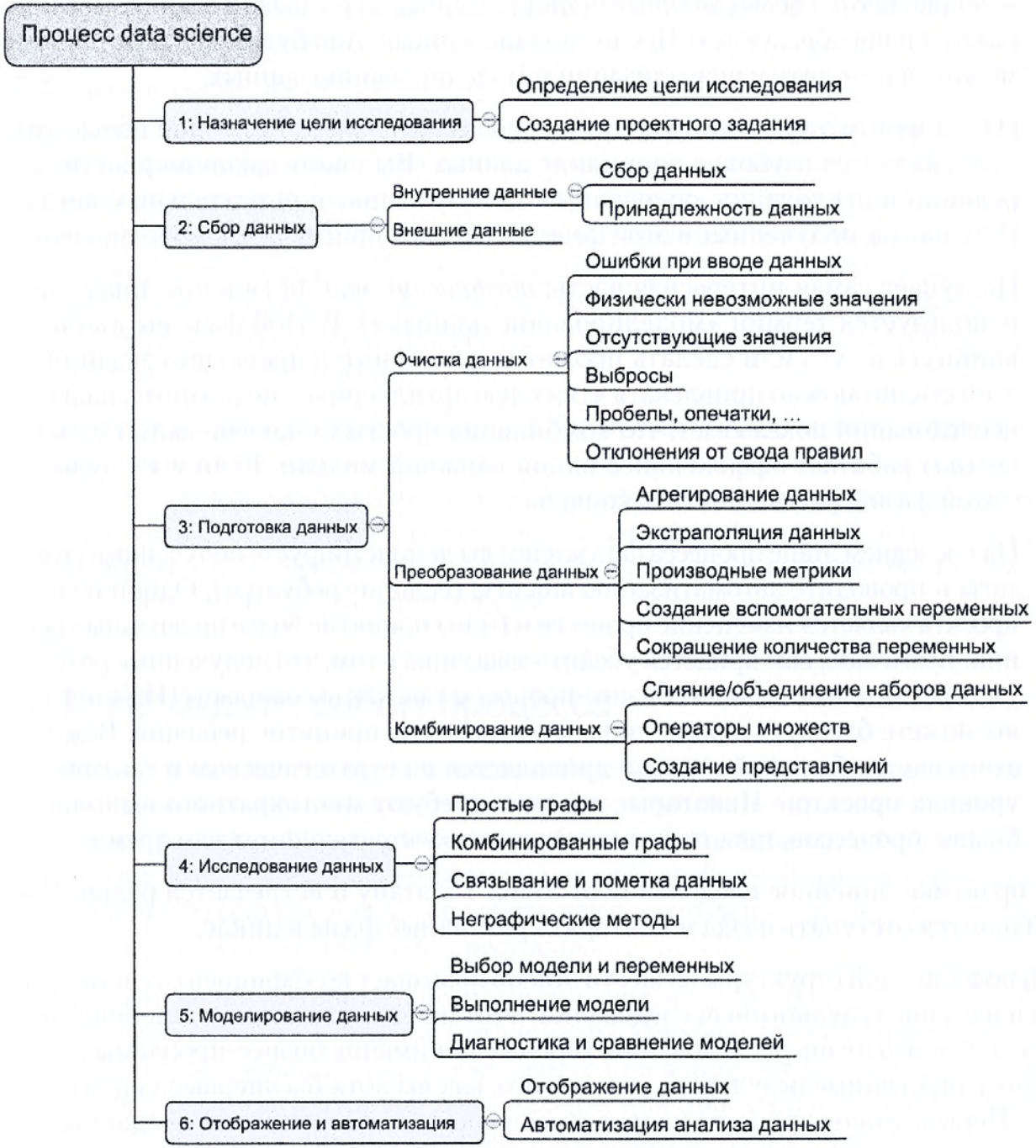


Рис. 2.1. Структура процесса data science

Дэви Силен, Арно Мейсман, Мохамед Али

Основы Data Science и Big Data

Python и наука о данных

MANNING

Конвейер ThreeBees: возникшие проблемы и их решения

- SQL-сервер виснет при доступе к покрытиям генов
  - Экспорт покрытий генов обратно в текстовый формат и их сведение для каждого конкретного сравнения
- Синтаксис SQL оказался сложен для понимания большинством сотрудников
  - Создание фронтенда под сбор метаданных
    - Отказ от использования SQL-сервера
- Высоконагруженный код на R трудно сделать эффективным
  - Распараллеливание инстансов R
  - Рефакторинг на python
- Некоторые файлы не помещаются в оперативной памяти
  - Парсинг по частям
- Метод отслеживания «бутылочных горлышек» учитывает число процессорных потоков, но не отслеживает потребление процессорного времени, ОЗУ, дискового кэша и т.д.
  - asyncio
    - Требуется доработка алгоритма создания очереди
  - Принудительный перевод нагруженных процессов в однопоточные очереди
- За состоянием кластера должен следить системный администратор
  - Культура DevOps позволяет перенести ответственность за состояние ПО на разработчика
- На каждой ноде кластера должен присутствовать актуальный код
  - Код сбрасывается в расшаренное хранилище
  - CD/CI
- Сложно отследить версию кода в общем хранилище
  - git
- Запуск обработки большого числа файлов должен производиться одновременно на нескольких нодах
  - Ansible
- Приходится вручную расписывать очереди для нод
  - Kubernetes + Redis

Структура кластера НИЛ OpenLab “Омиксные технологии” КФУ

- Железо
  - 3 NAS-сервера
    - 1x 15 Тб RAID1
    - 2x 36 Тб RAID1
  - 10 вычислительных нод
    - 4x Xeon E5-2630 v2 x2 (24 потока), 64 Гб ОЗУ
    - 6x Xeon E5-2630 v3 x2 (32 потока), 128 Гб ОЗУ
  - Сеть 1000Base-T
- Софт
  - CaaS on Bare Metal
    - Ubuntu Server 16.04 LTS + Ansible v2.4.3.0 + Docker v17.03 + Calico v1.6.1 + Kubernetes v1.9.3
  - CD/CI
    - git + DockerHub AutoBuilds / Resilio Sync / Jenkins

Уровни абстракции рабочего процесса конвейера *biopipelines-docker/bwt_filtering_pipeline*

- Программа-звено конвейера
  - C / C++ / Python etc.
- Управляющий скрипт конвейера
  - Python (CWL?) / bash / Perl etc.
- Под-воркер
  - Python-скрипт выгрузки очереди с Redis-сервера с временным интервалом
  - Инициализация конвейера при $N_{очереди} = N_{ядер}$ или при преждевременно опустошенной очереди
- Под-мастер
  - Python-скрипт построчной загрузки очереди JSON-объектов на Redis-сервер
- Конфигурация подов
  - YAML-шаблоны с поддержкой Jinja2
- Конфигурация проекта
  - ~~HELM~~ YAML-чарт с необходимой конфигурацией

Продакшн-схема конвейера *biopipelines-docker/bwt_filtering_pipeline*

1. Создаются линкерные таблицы
   - Пути к файлам референса
   - Пути к файлам образцов
   - Групповые метаданные
2. Создаются файлы конфигурации субпроекта
   - Главный чарт *config.yaml* создается вручную
   - Чарты мастера и воркеров (*master.yaml* и *worker.yaml*) генерируются скриптом из соответствующих шаблонов
   - Все три файла перемещаются в папку субпроекта и выгружаются
3. *kubectl* разворачивает мастер, который выгружает очередь из JSON-объектов в указанную очередь на Redis-сервер
4. *kubectl* разворачивает задание (*job*, не *deployment*) и контроллер репликации воркеров
5. На доступных нодах разворачивается указанное число воркеров, каждый из которых загружает очередь с Redis-сервера через определенный интервал времени
6. Воркер запускает работу конвейера

宀

roof

亻

person

一

one

白

white

III IV

3 3

11 宿題 18

シュク ダイ

inn topic

しゅくだい

【homework】

日

day

疋

bolt of cloth

頁

big shell

Проект “Kubernetes Cluster on Bare Metal”

1. Поднять на своем хосте 3 виртуальных машины
   - Ubuntu Server 16.04 LTS (core, no GUI)
   - Имена: *node0, node1, node2*
   - Установить пакеты
     - *openssh-client openssh-server python sshpass apt-transport-https ca-certificates curl software-properties-common git python-pip python3-pip*
2. Настроить сетевое окружение
   - Имя в */etc/hostname*
   - Имена и IP-адреса в */etc/hosts*
   - NAS-сервер на node0 (указать папку */data* в */etc/fstab*)
   - SSH-сервер в */etc/ssh/sshd_config*
   - Авторизация по SSH-ключам без запроса пароля (см. *ssh-copy-id*)
3. Установить на *node0* Ansible, Netaddr и сконфигурировать окружение Ansible
4. Установить Docker и Kubernetes
5. Добиться корректной работы команд *kubectl get nodes* и *kubectl get pods --show-all*