Масштабирование

Олег Сухорослов

Распределенные системы

Факультет компьютерных наук НИУ ВШЭ

31.10.2022

Проблема

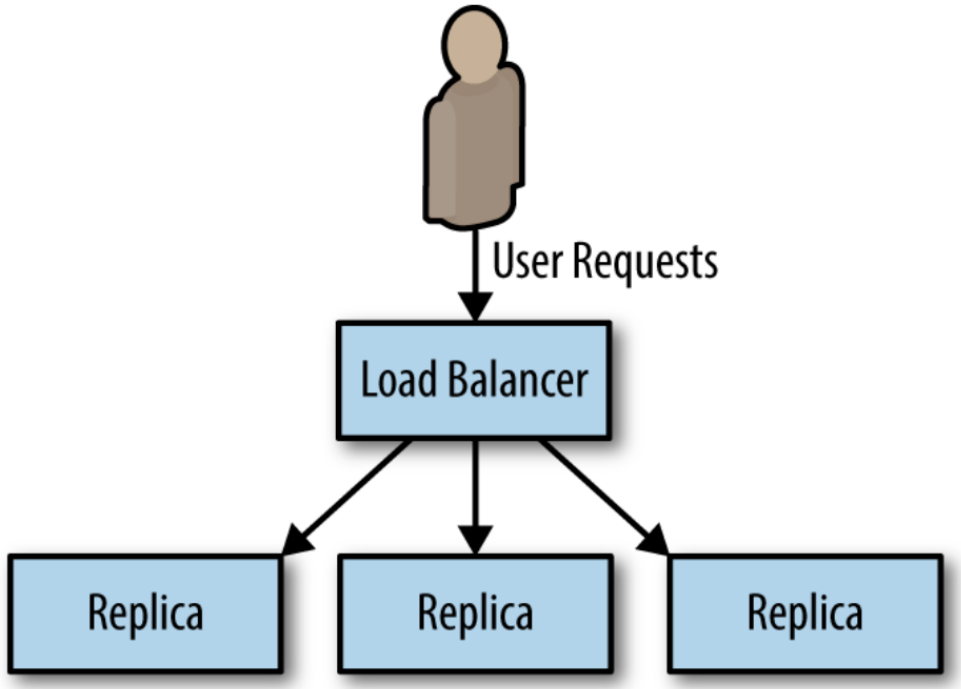
Аспекты масштабируемости

- Число обрабатываемых запросов
- Объем данных
- Объем вычислений/данных на запрос

Используемые техники

- Репликация
- Кэширование
- Шардинг
- Параллельная обработка (в другой раз)

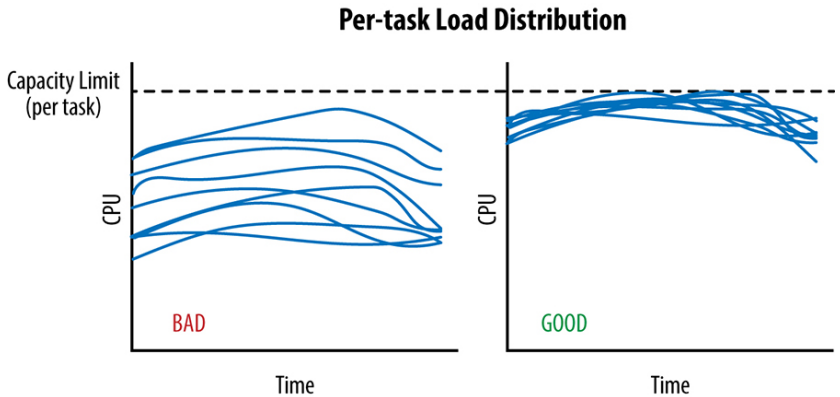
Репликация (stateless сервис)



Load Balancer: варианты реализации

- Layer 4 (connection/session)
  - Оперирует данными (пакетами) на транспортном (TCP, UDP) уровне
- Layer 7 (application)
  - Распределяет запросы на прикладном уровне (HTTP) на основе их содержимого
- Балансировка с помощью DNS
- Hardware vs Software
- Load Balancer vs Proxy

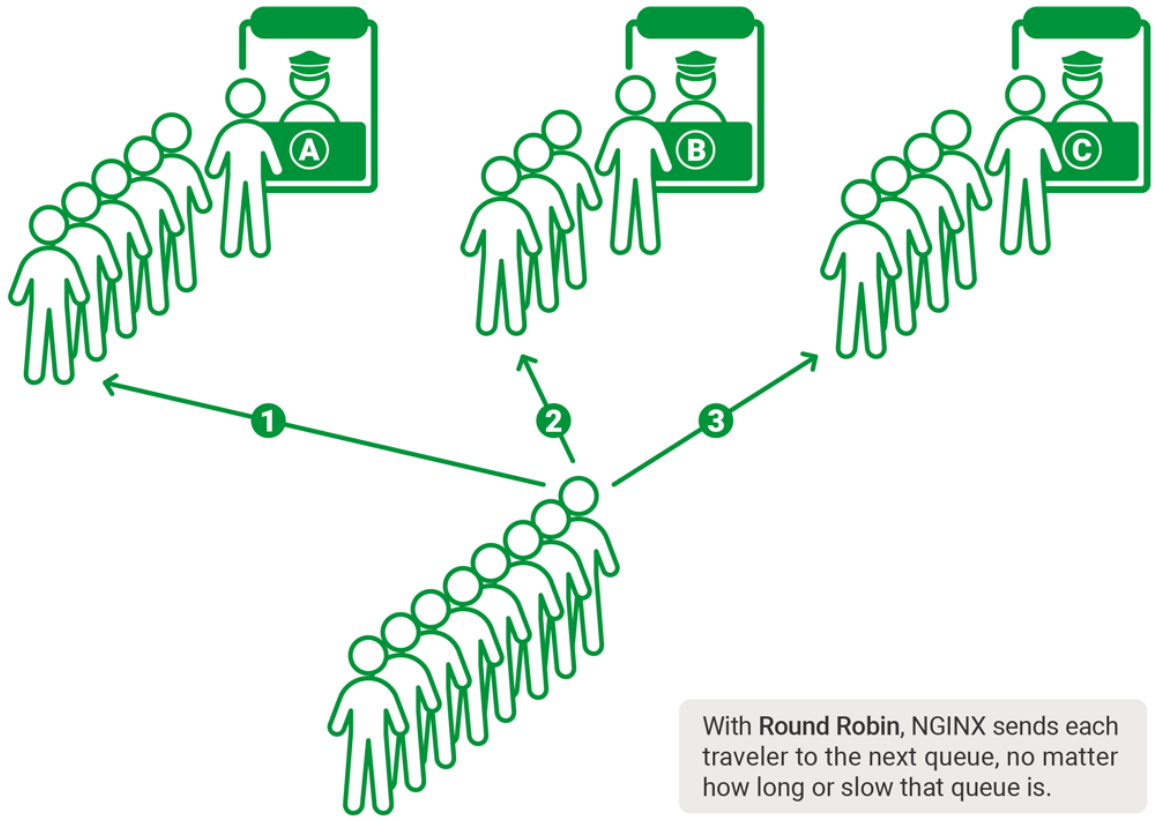
Балансировка нагрузки



Алгоритмы

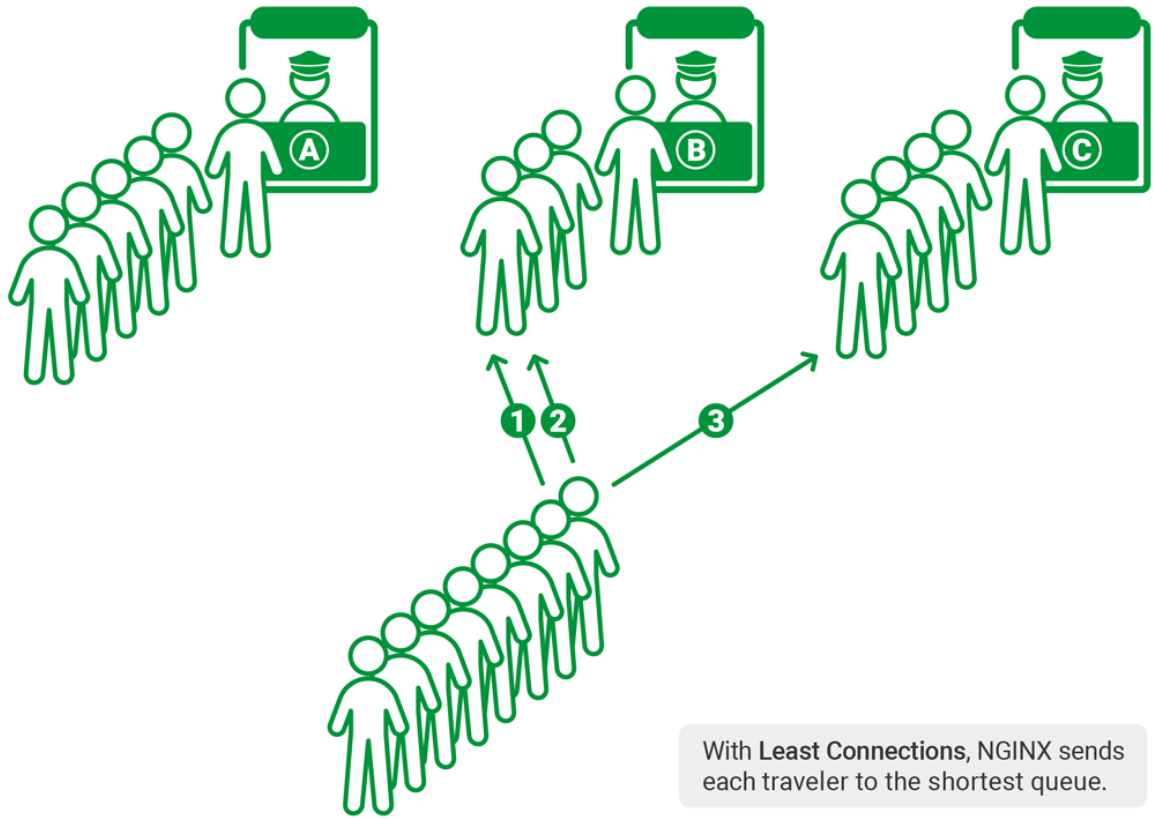
- Random
- Round Robin
- Least Connections
- Least Time
- Power of Two Choices

Round Robin



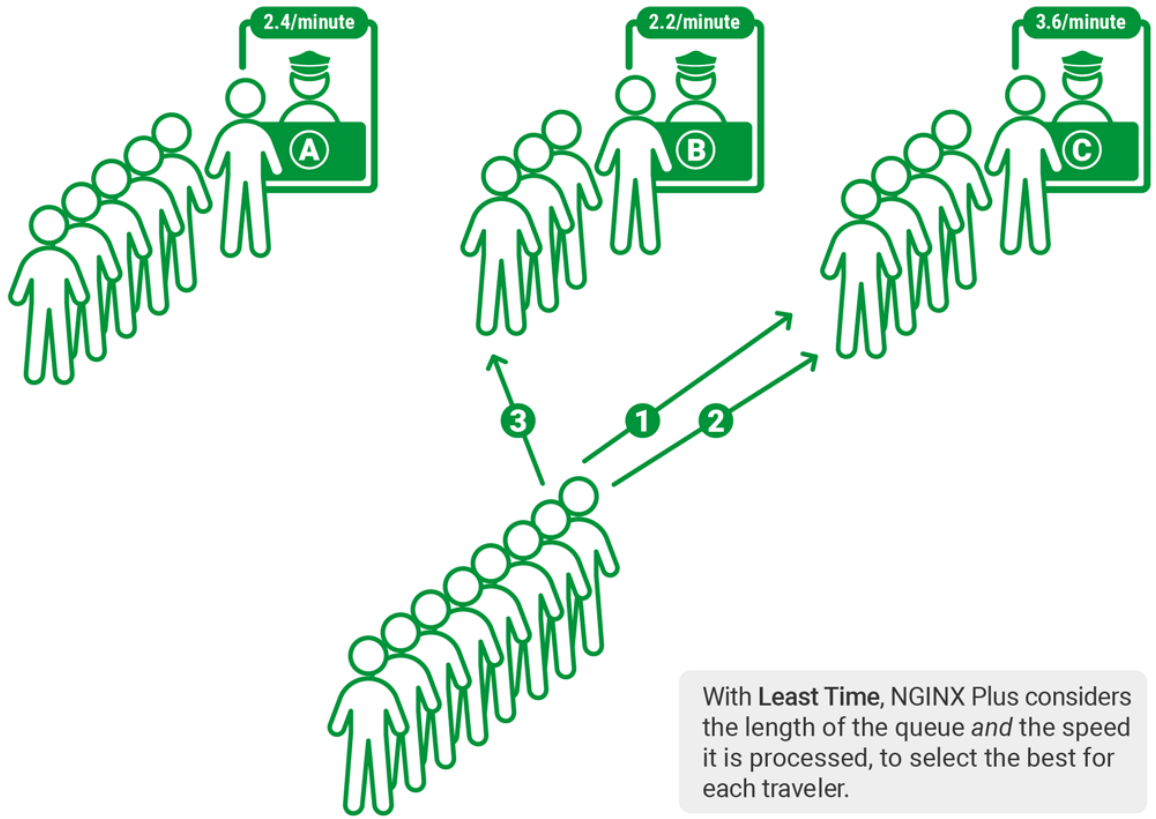
https://www.nginx.com/blog/nginx-power-of-two-choices-load-balancing-algorithm

Least Connections



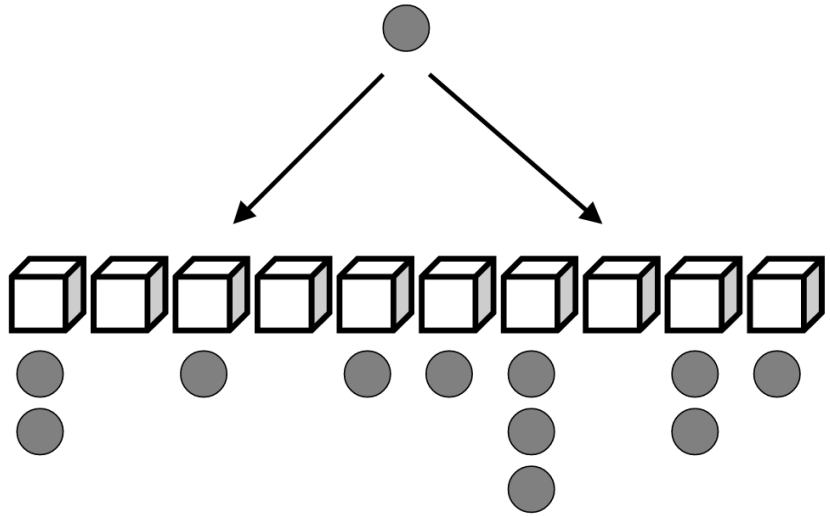
https://www.nginx.com/blog/nginx-power-of-two-choices-load-balancing-algorithm

Least Time



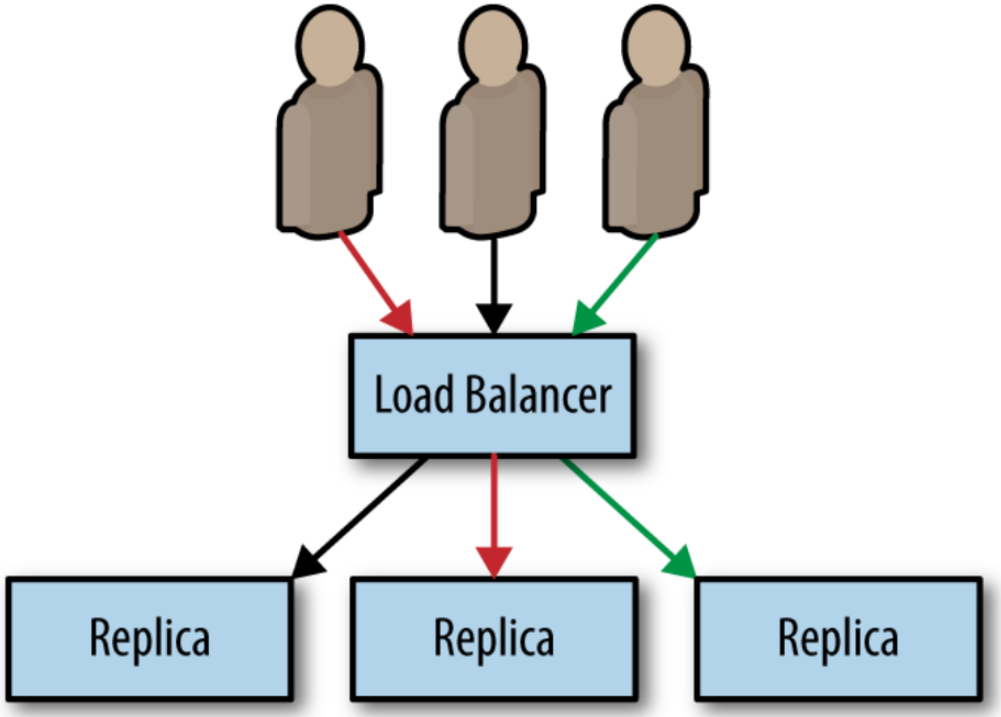
https://www.nginx.com/blog/nginx-power-of-two-choices-load-balancing-algorithm

Power of Two Choices

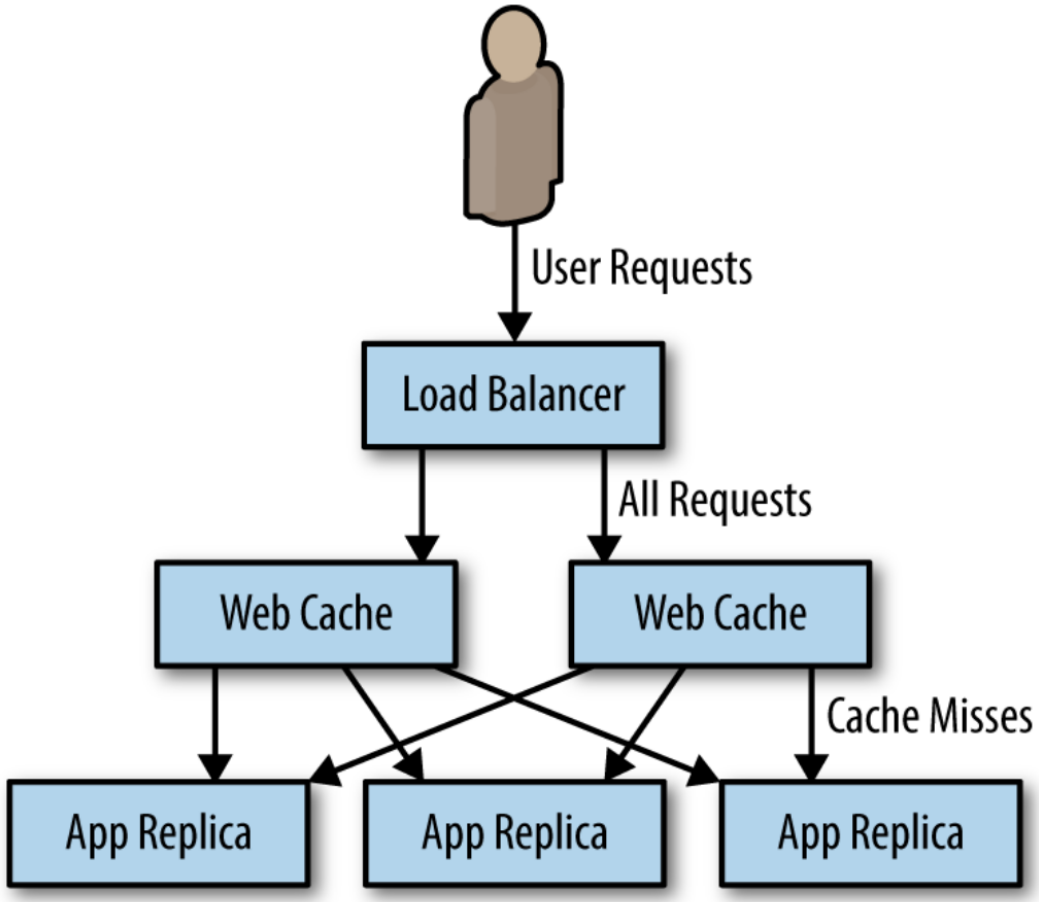


- Максимальная нагрузка:
  - Random: $O(\frac{\log n}{\log \log n})$
  - Two random choices: $O(\log \log n)$
- Mitzenmacher M. The Power of Two Choices in Randomized Load Balancing (1996)

Отслеживание клиентских сессий



Кэширование



Варианты реализации кэша

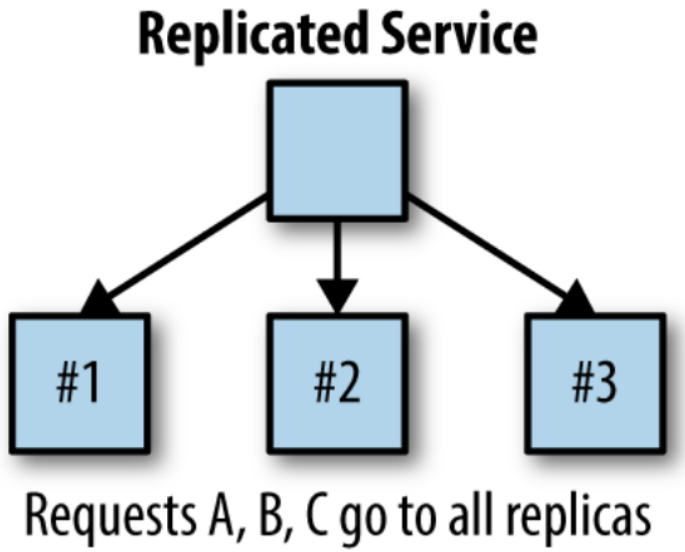
- Кэш на стороне клиента
- Промежуточные кэширующие прокси-сервера
- Content Delivery Network (CDN)
- Caching HTTP reverse proxy (Varnish, Nginx)
- Специализированные хранилища (memcached, Redis)

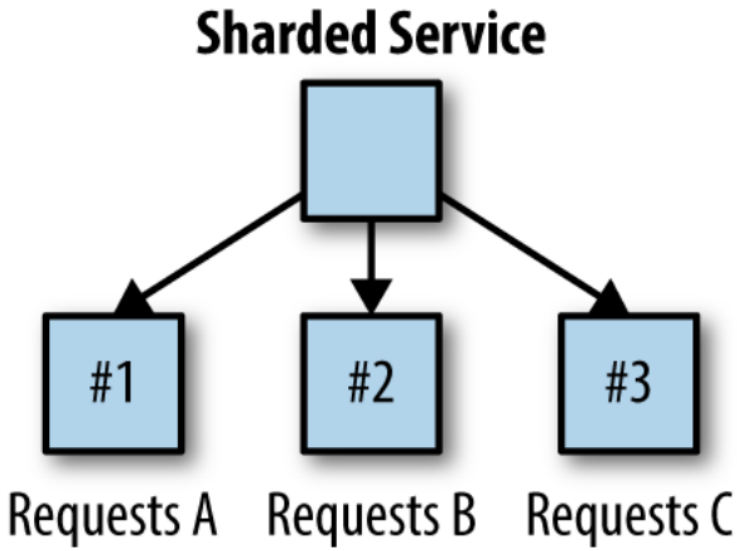
Политики вытеснения

- Least Recently Used
- Least Frequently Used
- Least Frequently Recently Used
- First In First Out

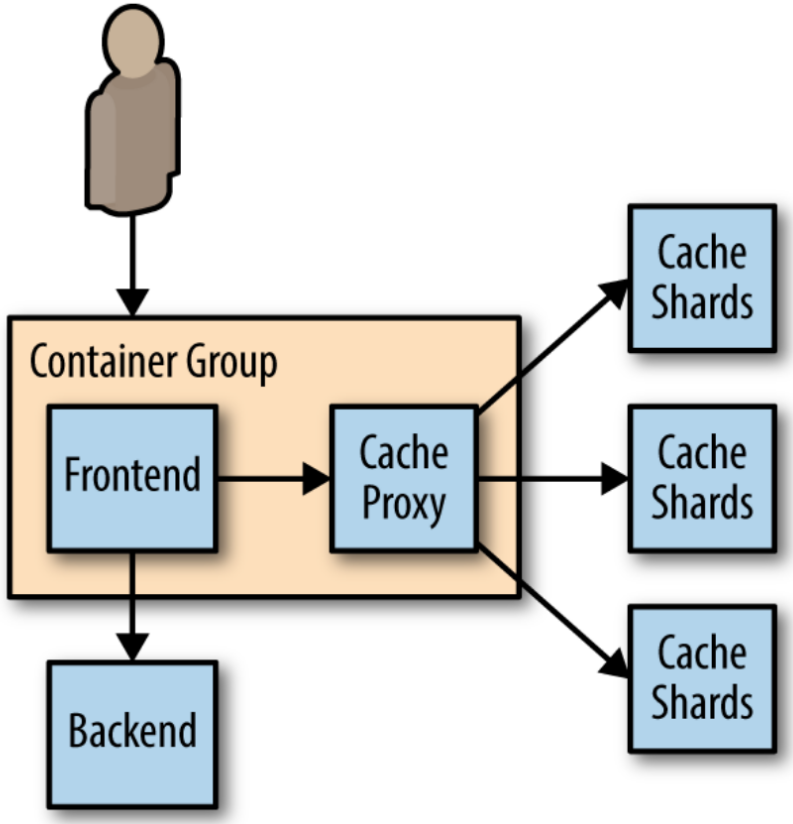
> For a significant number of workloads, FIFO has similar or lower miss ratio performance as LRU for in-memory caching workloads.
>
> A large scale analysis of hundreds of in-memory cache clusters at Twitter, 2020

Шардинг (stateful сервис)





Шардинг кэша

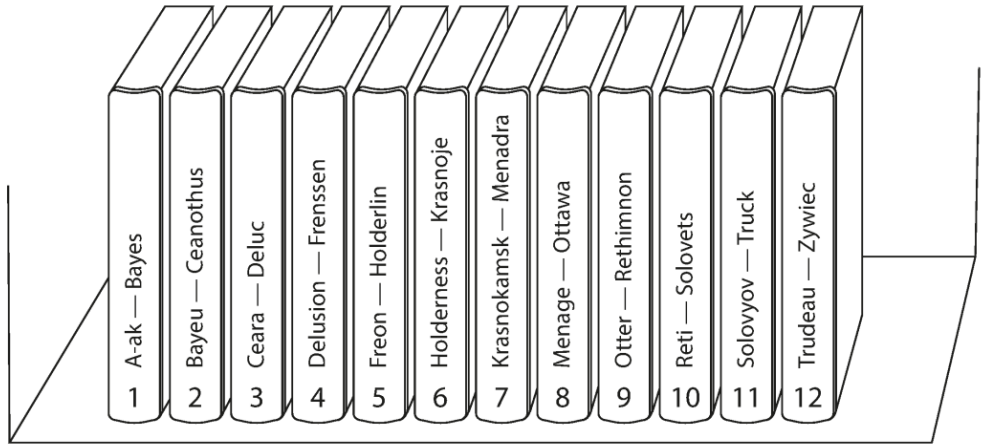


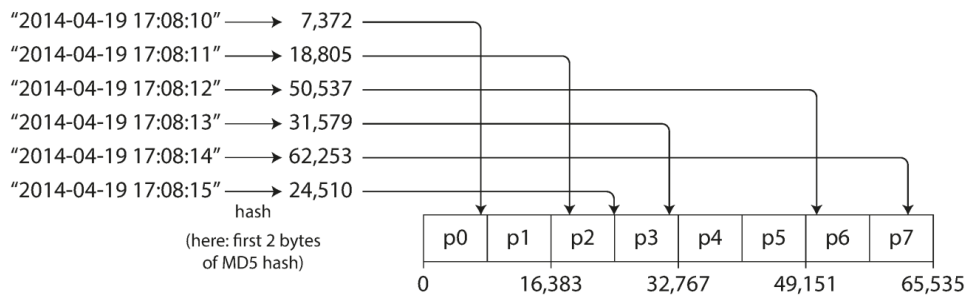
Принцип разбиения на шарды

- $Shard = ShardingFunction(Request)$
- Требования
  - Детерминированность
  - Равномерность
  - Устойчивость к изменениям состава узлов (rebalancing)

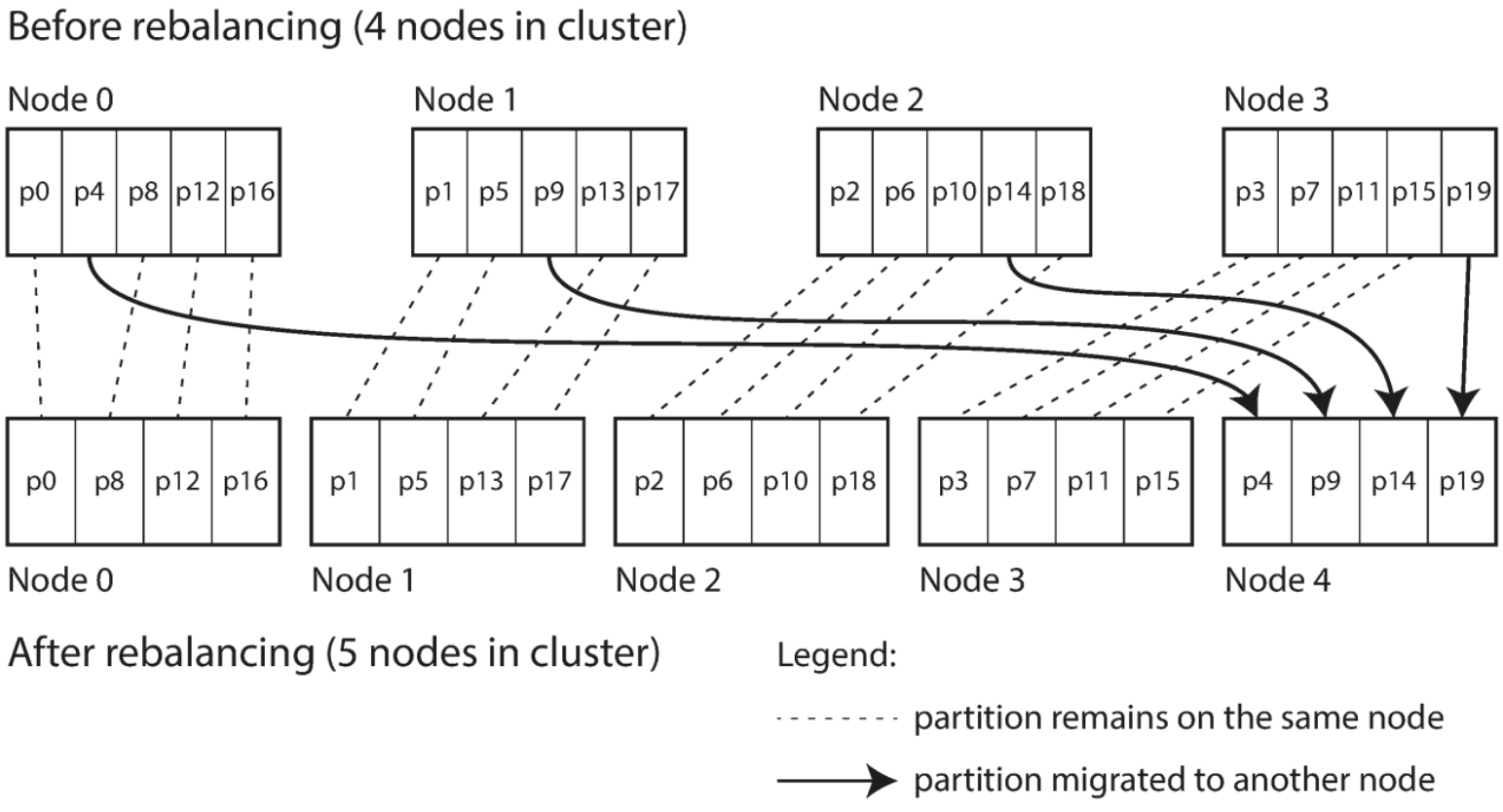
Подходы

- Вертикальный шардинг
- Горизонтальный шардинг
  - (Выбор ключа)
  - Интервалы ключей
  - Хэширование ключей
  - *hash(k) mod N*
  - Согласованное хеширование
- Число шардов
  - Фиксированное
  - Пропорционально числу узлов
  - Пропорционально размеру данных

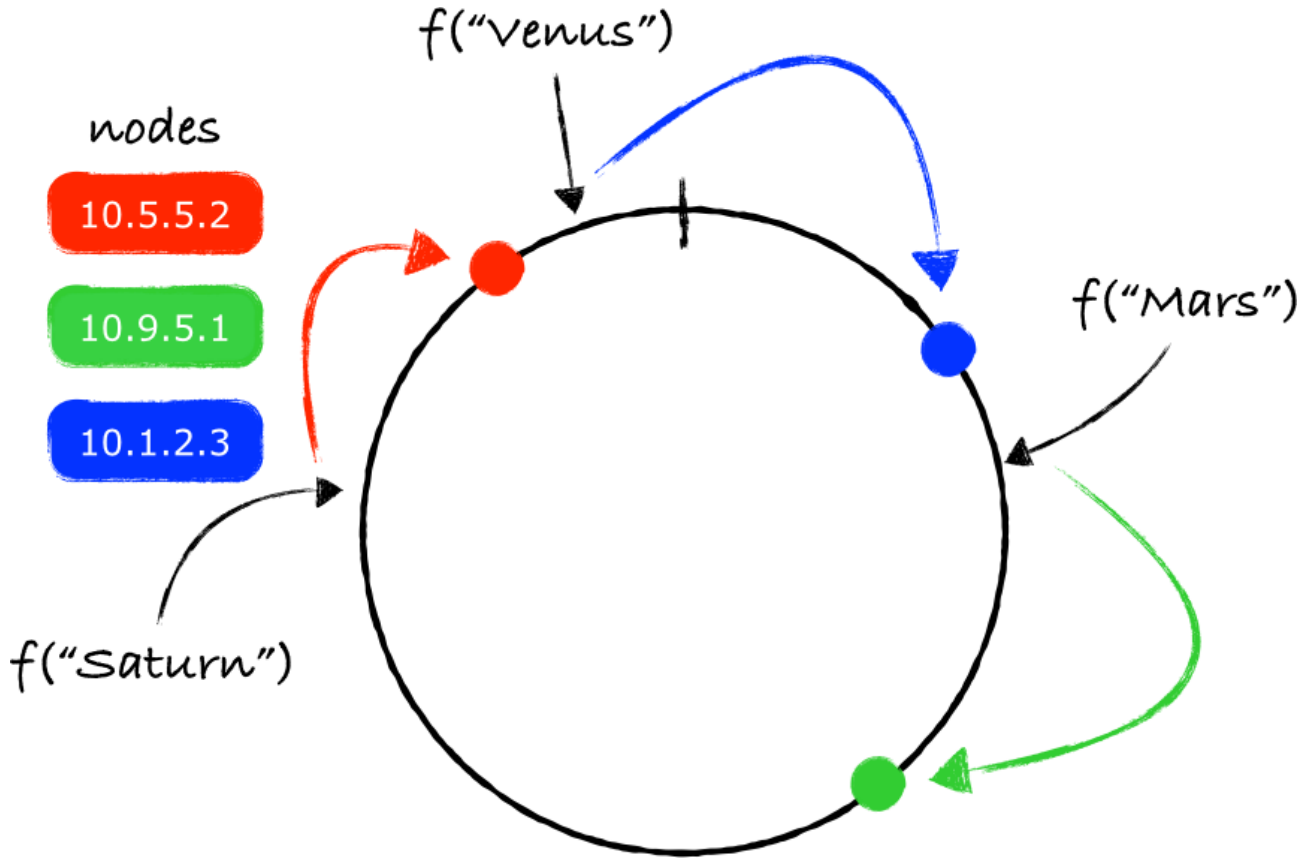




Фиксированное число шардов

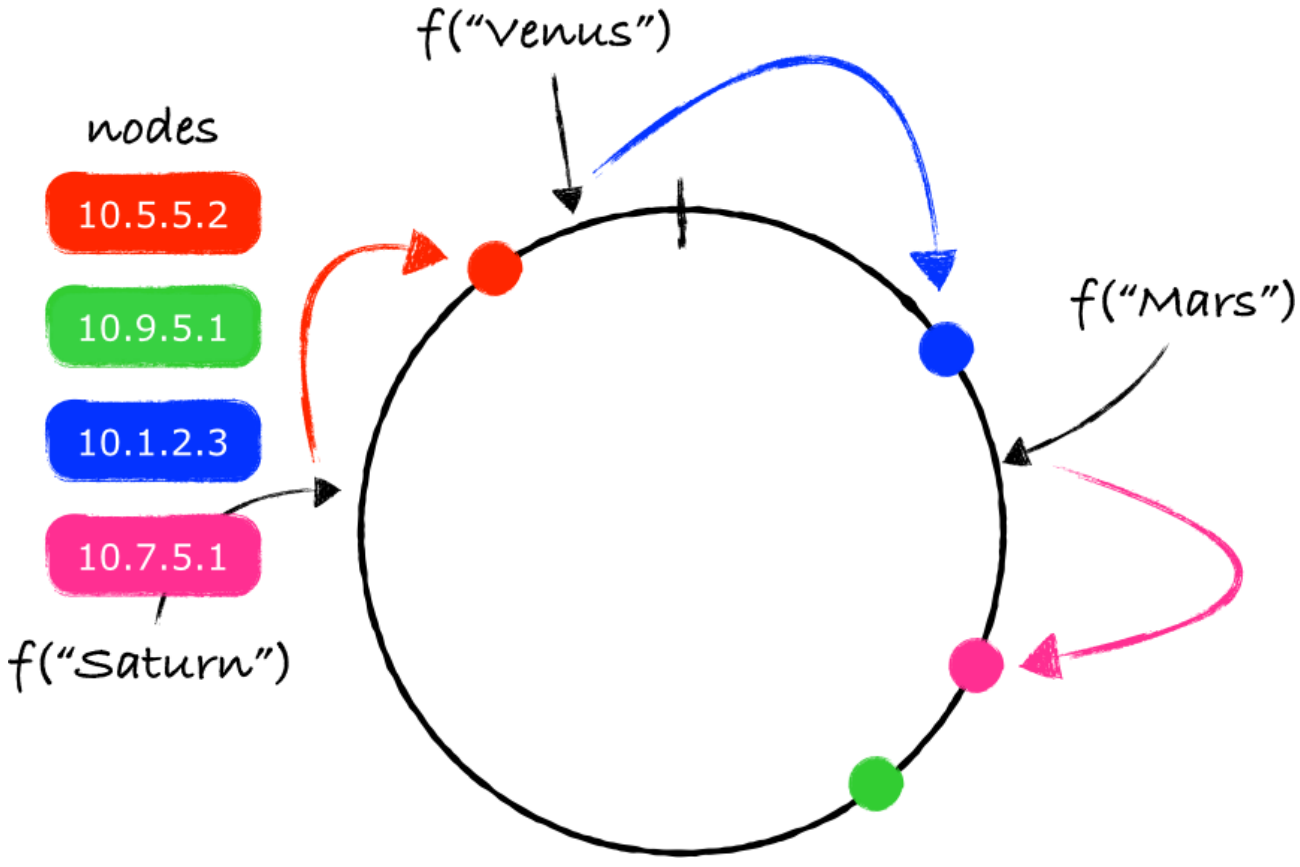


Согласованное хеширование

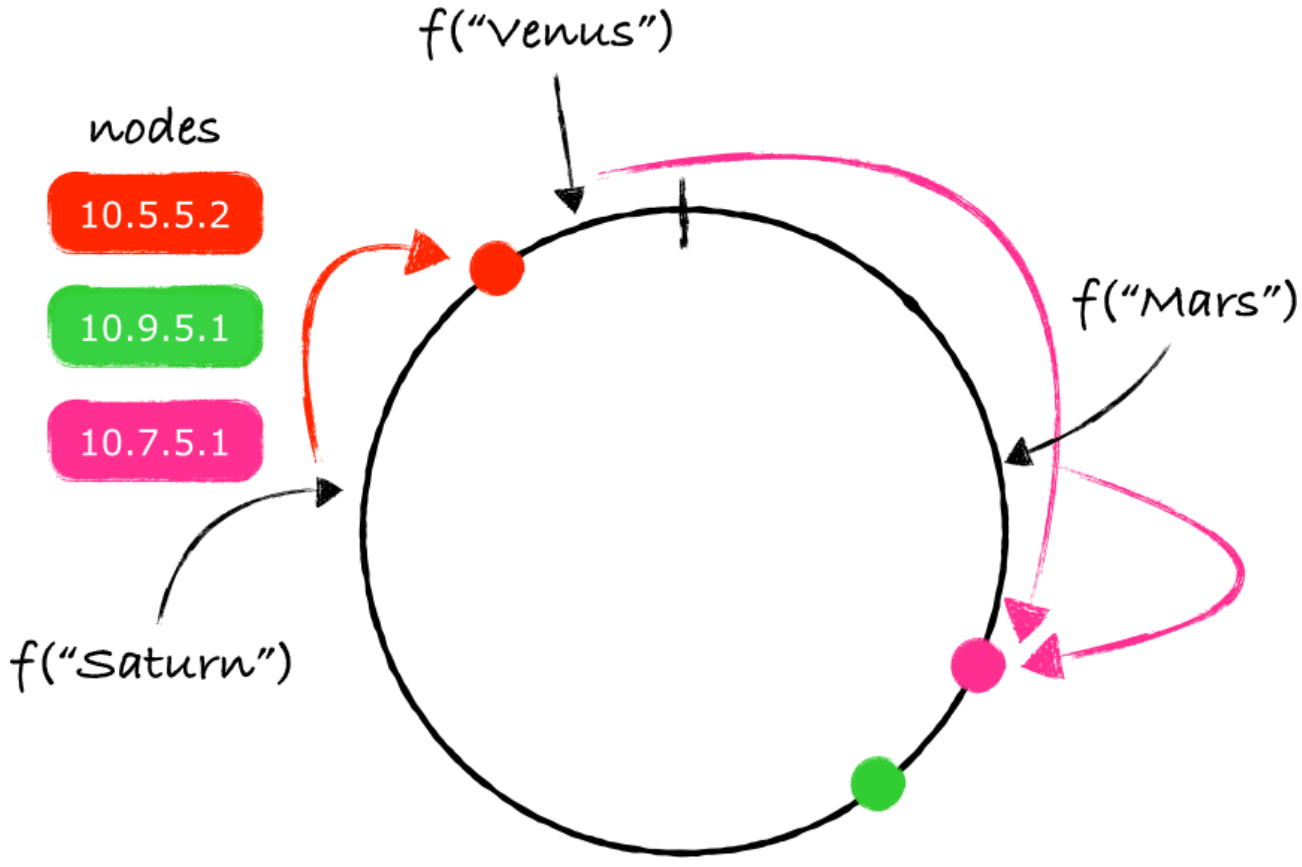


Karger D. et al. Consistent hashing and random trees: Distributed caching protocols... (1997)
https://blog.carlosgaldino.com/consistent-hashing.html

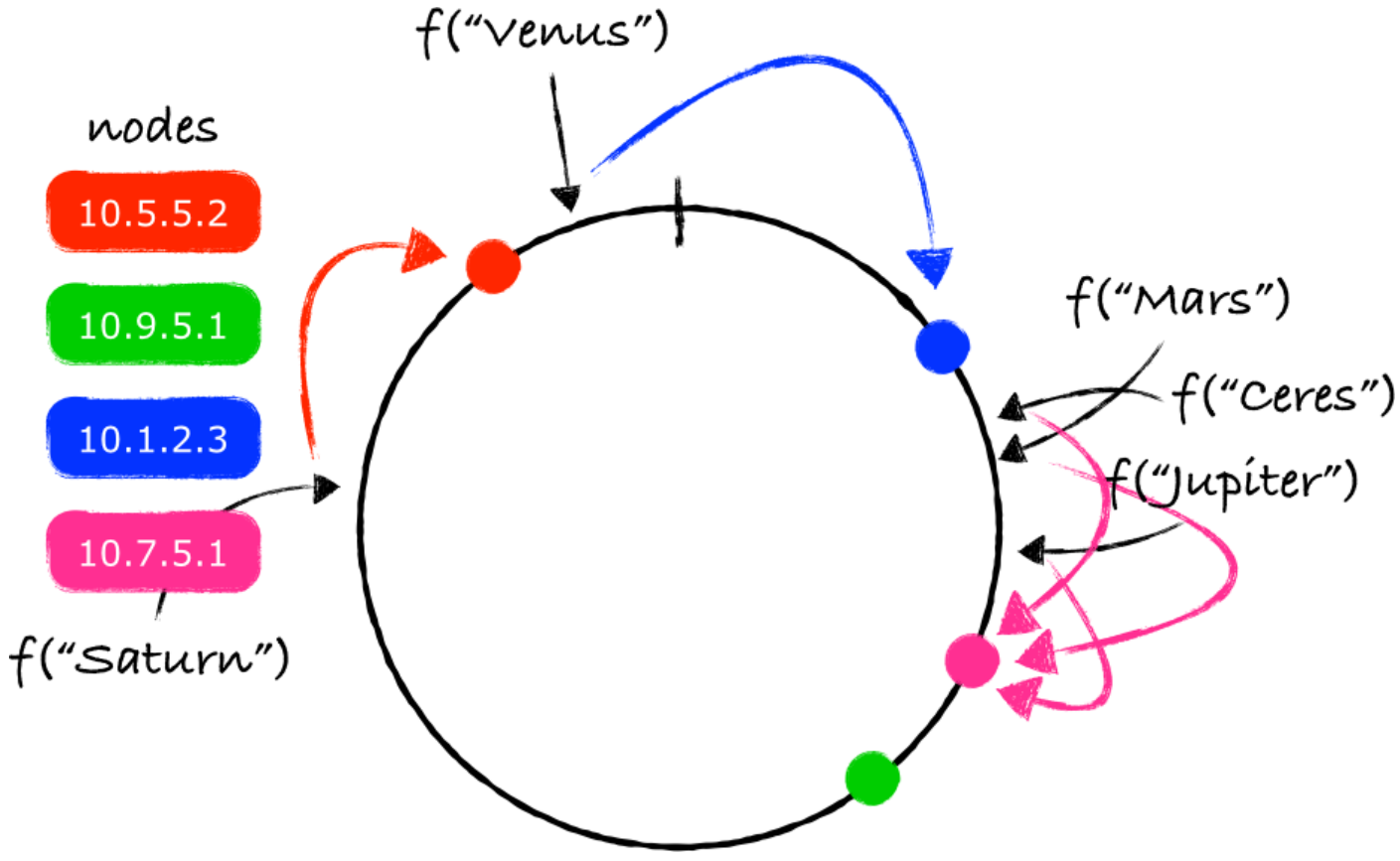
Добавление узла



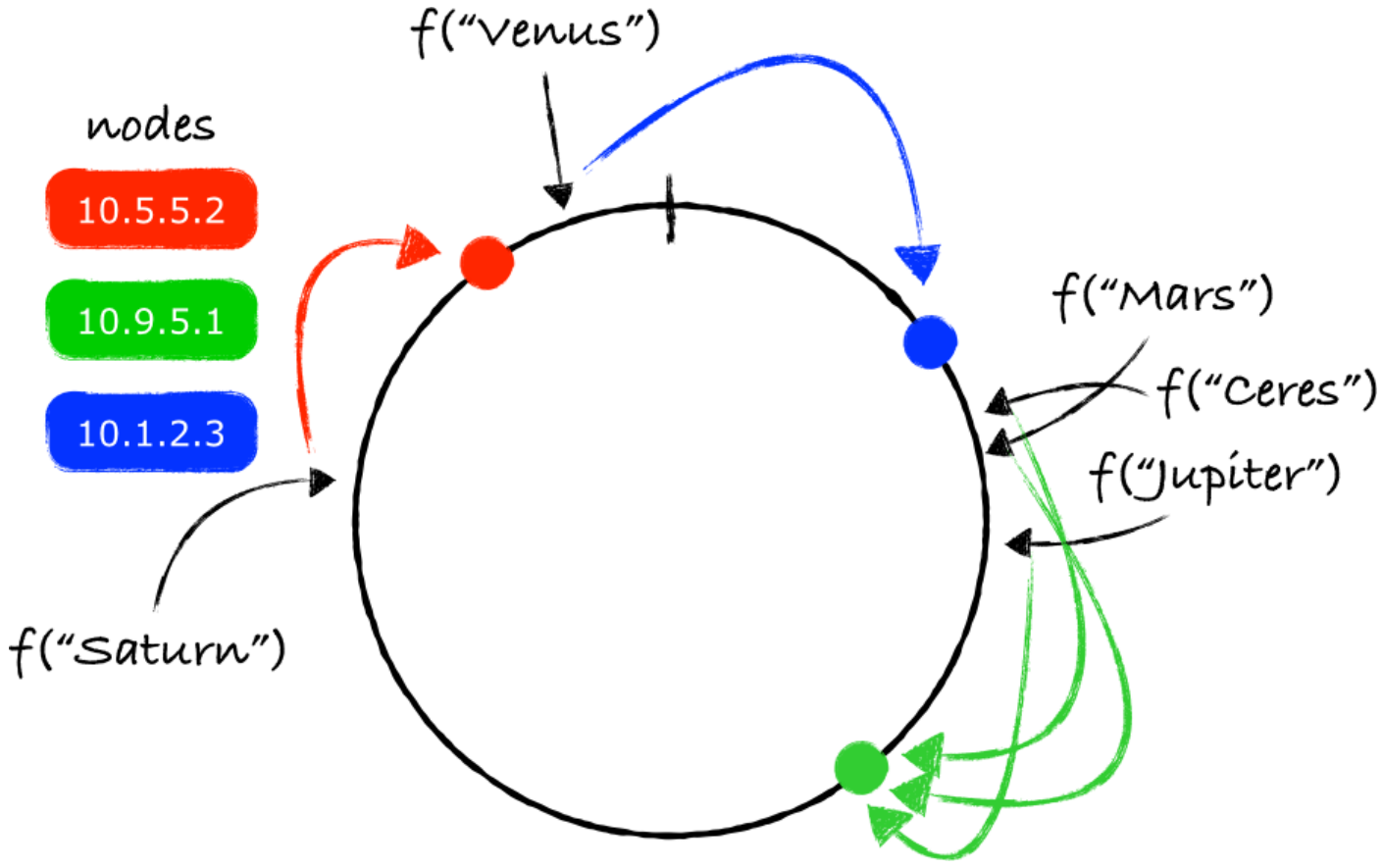
Удаление узла



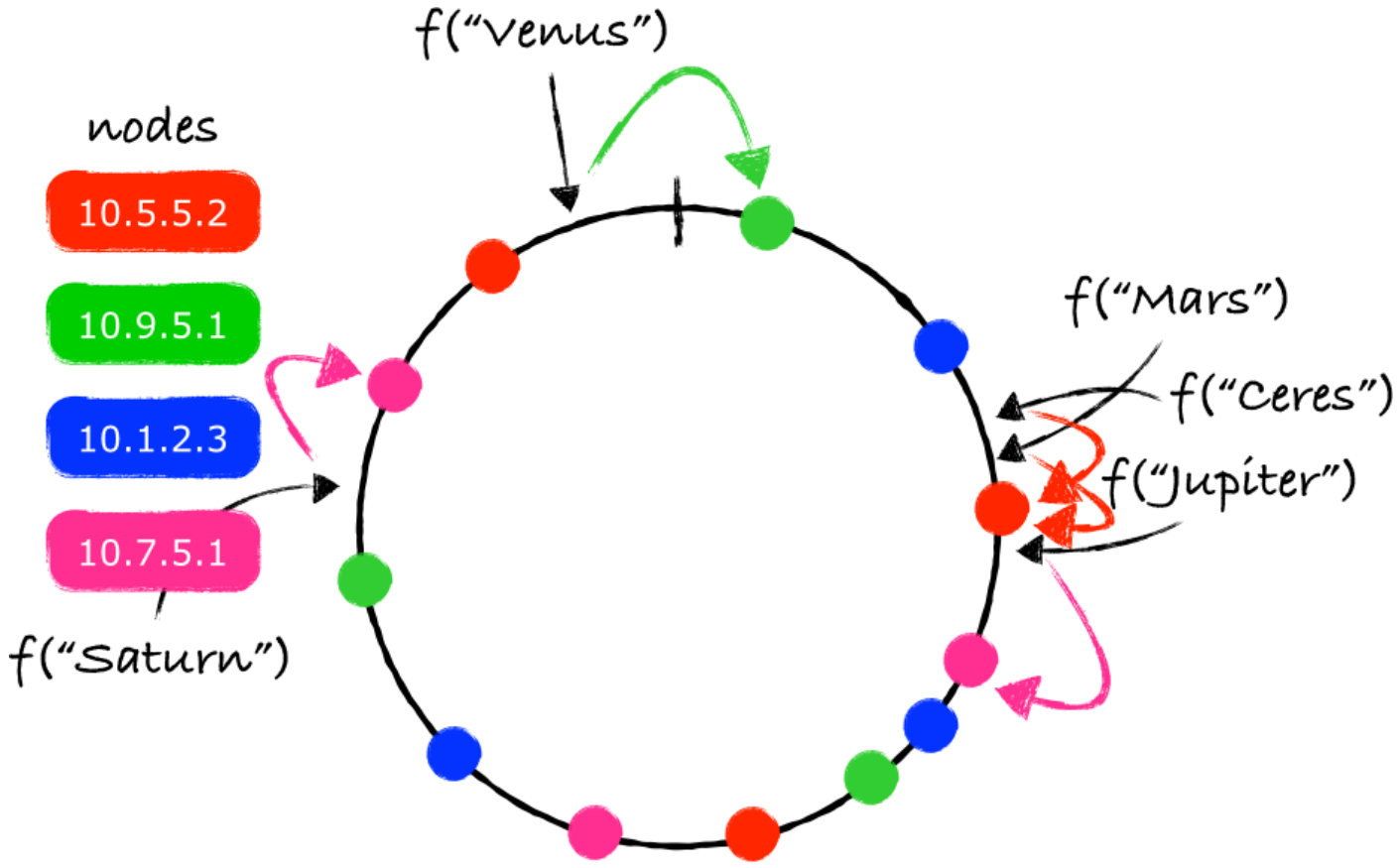
Проблема 1


f("Venus")
nodes
10.5.5.2
10.9.5.1
10.1.2.3
10.7.5.1
f("Saturn")
f("Mars")
f("Ceres")
f("Jupiter")

Проблема 2

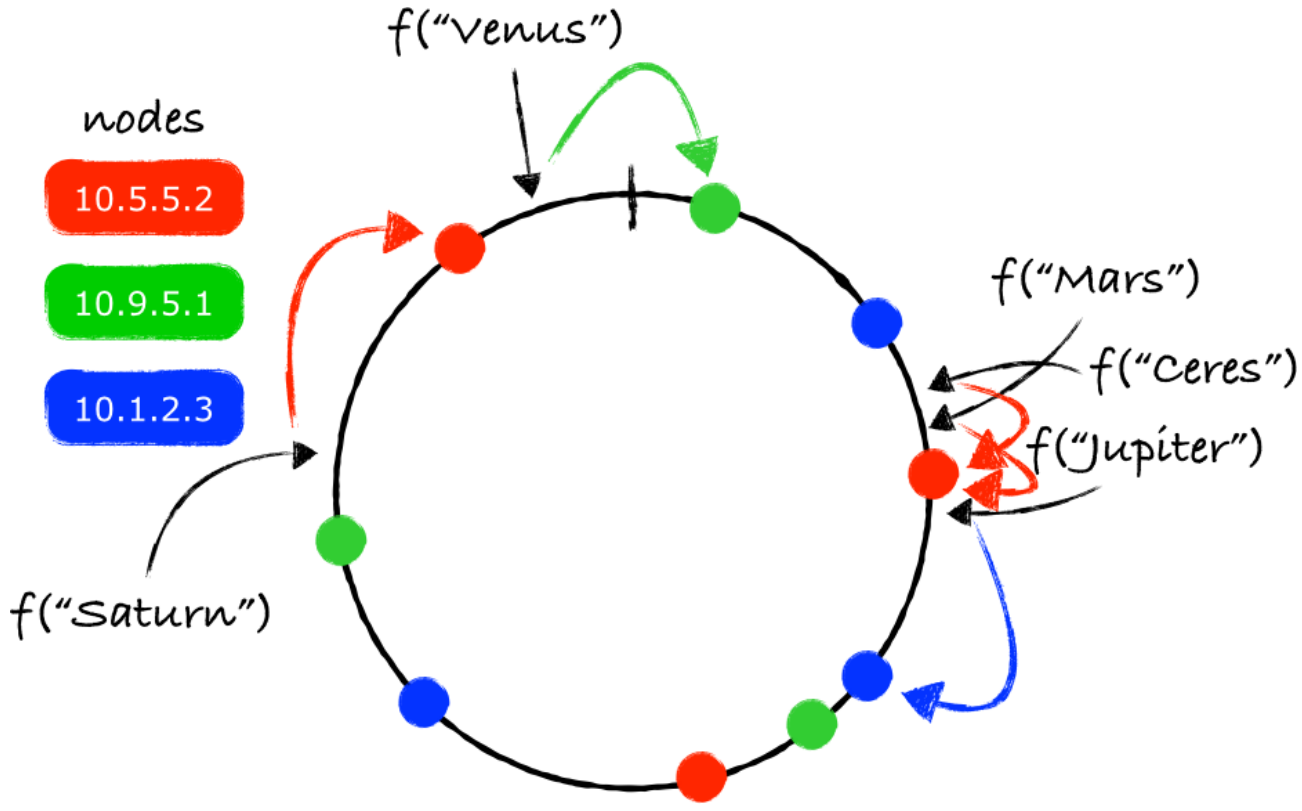

f("Venus")
nodes
10.5.5.2
10.9.5.1
10.1.2.3
f("Mars")
f("Ceres")
f("Jupiter")
f("Saturn")

Виртуальные узлы



Детерминированный алгоритм: New token allocation algorithm in Cassandra 3.0

Удаление узла



Rendezvous Hashing

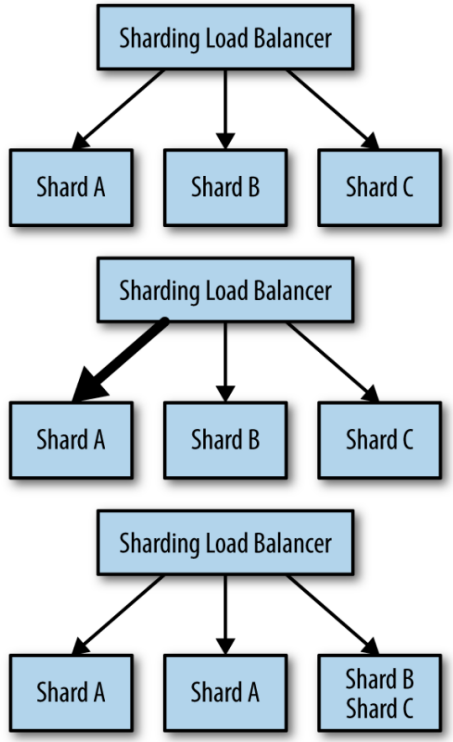
- Using name-based mappings to increase hit rates (1998)
- Другое название - Highest Random Weight (HRW)
- Вес сервера для ключа вычисляется как *hash(key,server)*
- Выбирается сервер с максимальным весом

Другие подходы

- A fast, minimal memory, consistent hash algorithm (2014)
- Multi-probe consistent hashing (2015)
- Maglev: A fast and reliable software network load balancer (2016)
- Consistent Hashing with Bounded Loads (2016)

Балансировка нагрузки при шардинге

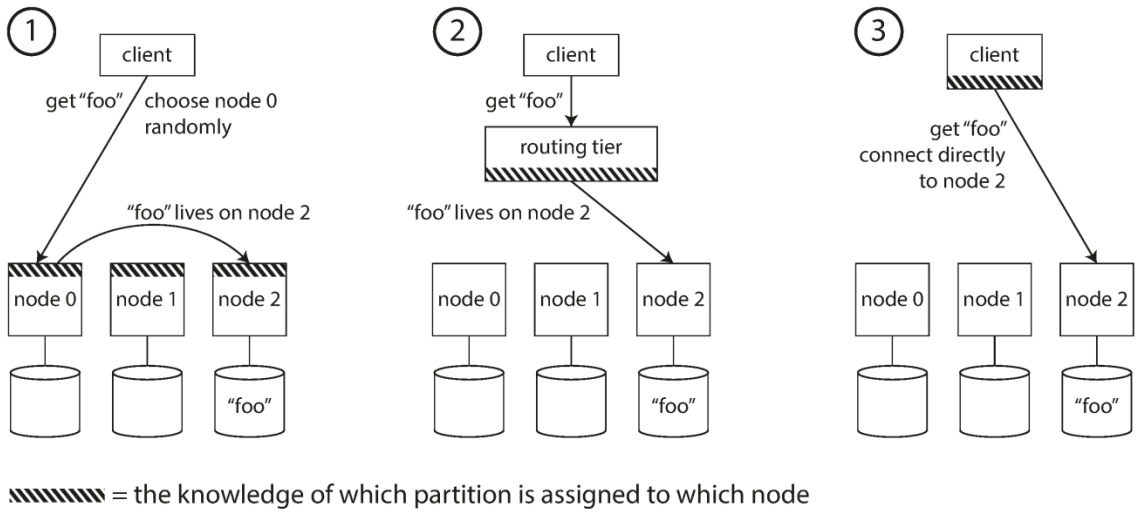
- Выбор "хорошего" принципа разбиения на шарды
  - Не гарантирует отсутствие "горячих" шардов
- Микро-шарды
  - Число шардов >> машин
- Переконфигурация шардов
  - Разбиение и слияние шардов
- Выборочная репликация шардов
  - Изменение числа реплик в зависимости от нагрузки



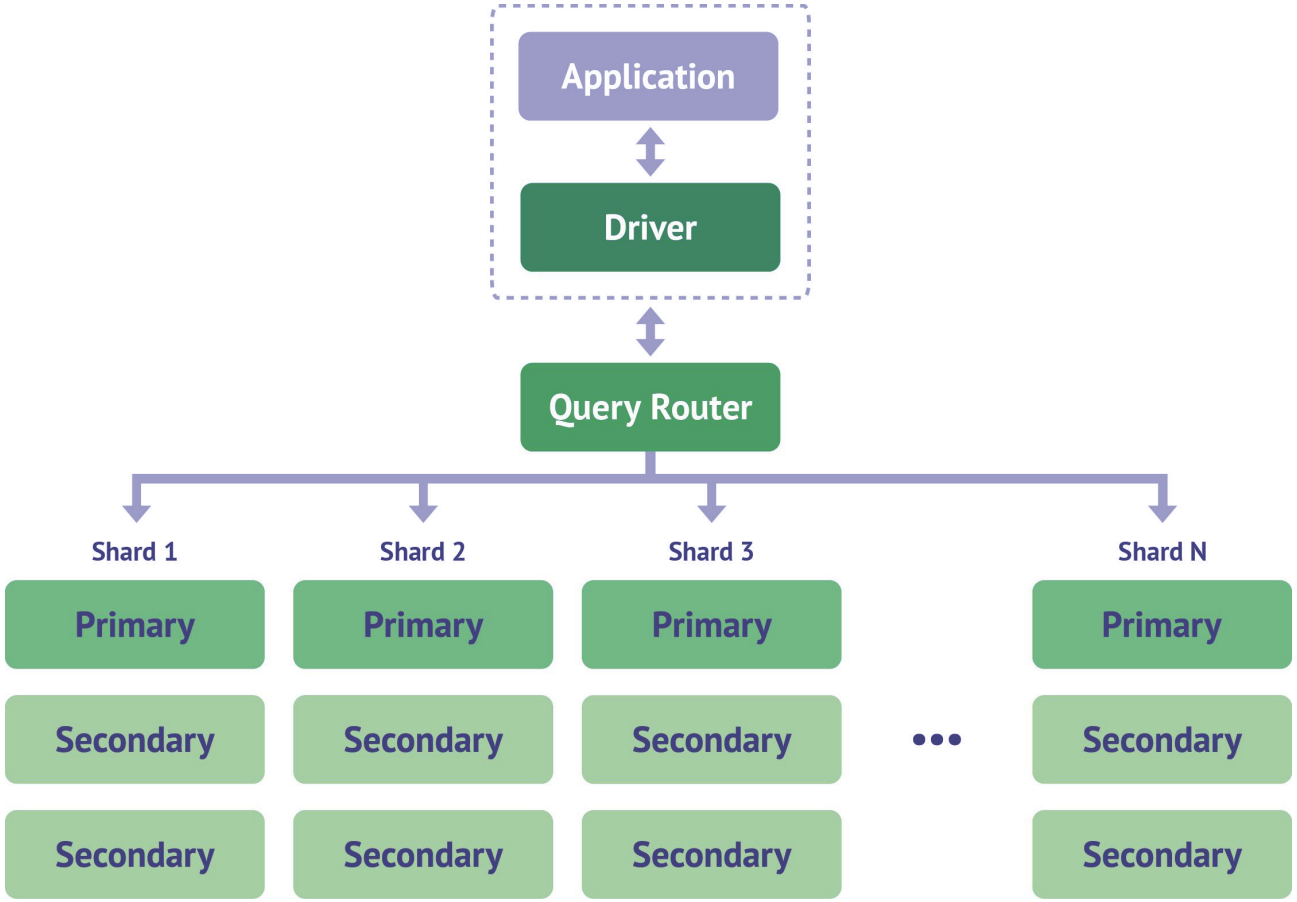
Маршрутизация запросов

Как клиент определит, на какой узел надо отправить запрос?

- Обратиться к любому узлу, а тот перенаправит запрос при необходимости
- Использовать отдельный промежуточный слой, знающий о шардах
- Хранить информацию (шард, узел) на клиенте, возможно частично (DHT)



Шардинг + репликация данных



Литература

- Burns B. Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services (главы 5-6)
- Site Reliability Engineering (глава 20)
- Klein M. Introduction to modern network load balancing and proxying
- Kleppmann M. Designing Data-Intensive Applications (глава 6)
- Roughgarden T., Valiant G. Consistent Hashing
- Gryski D. Consistent Hashing: Algorithmic Tradeoffs

Дополнительно

- Site Reliability Engineering (глава 19)
- Tarreau W. Test Driving "Power of Two Random Choices" Load Balancing
- McMullen T. Load Balancing is Impossible
- Gury S. Predictive Load Balancing: Unfair but Faster & more Robust
- Gessner K. How Etsy caches: hashing, Ketama, and cache smearing
- Holt G. Building a Consistent Hashing Ring
- Аксёнов А. Теория шардирования
- Упомянутые статьи