爬虫系统与“打个头”系统

Web Crawler & Typeahead

课程版本 v5.0　　欧阳锋 老师

扫描二维码关注微信/微博
获取最新面试题及权威解答

微信：ninechapter
知乎专栏：http://zhuanlan.zhihu.com/jiuzhang
微博：http://www.weibo.com/ninechapter
官网：www.jiuzhang.com

本节重点

九章算法

Design a web crawler

Dropbox, Google, Turn, Alibaba

Design thread-safe producer and consumer

Google, Amazon, TripAdvisor, Microsoft, Snapchat

Design a Typeahead

LinkedIn, Uber, Hulu

你会掌握

1. Producer consumer pattern
2. How to design distributed web crawler
3. How search engine works
4. How to design Google Suggestion

几个概念

1. HTTP
2. BFS（宽度优先搜索）
3. Thread（线程）/ multi-thread（多线程）
4. Serialization（序列化）
5. Hash（哈希）

Design a web crawler

九章算法

Interviewer: How to design a web crawler?

Design a web crawler

How to design a web crawler?

- multi-threading
- system design

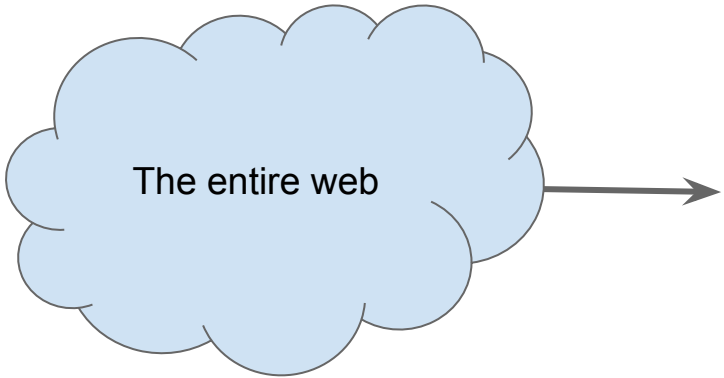
What is a web crawler?

- For collecting data/information from the web

Design a web crawler

九章算法

What does Google's crawler do?



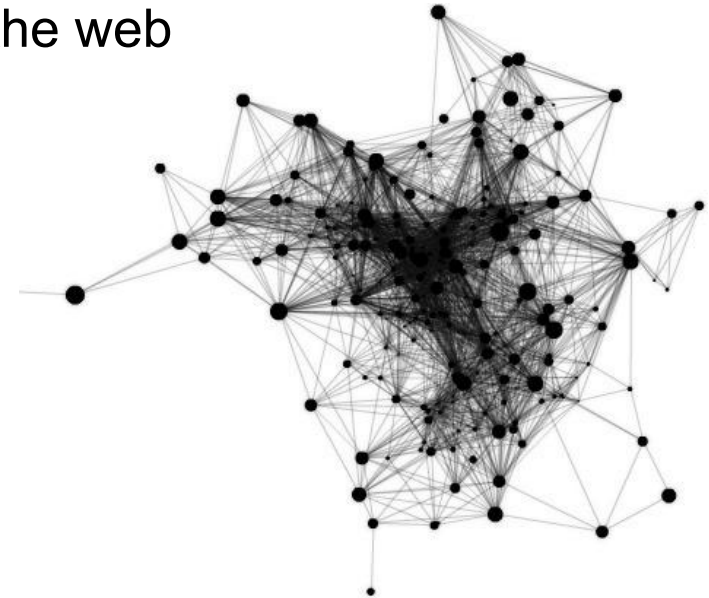
url	page_content
“https://www.wikipedia.org/”	“<!DOCTYPE html><html>...</html>”
“http://www.sina.com/”	“<!DOCTYPE html><html>...</html>”
...	...
“http://www.bbc.co.uk/”	“<!DOCTYPE html><html>...</html>”
...	...

Design a web crawler

九章算法

Scenario

Given seeds, crawl the web



Design a web crawler

Scenario: How many web pages? how long? how large?

1. crawl **1.6m web pages per second**
    - 1 trillion web pages
    - crawl all of them every week
2. 10p (petabyte) web page storage
    - average size of a web page: 10k

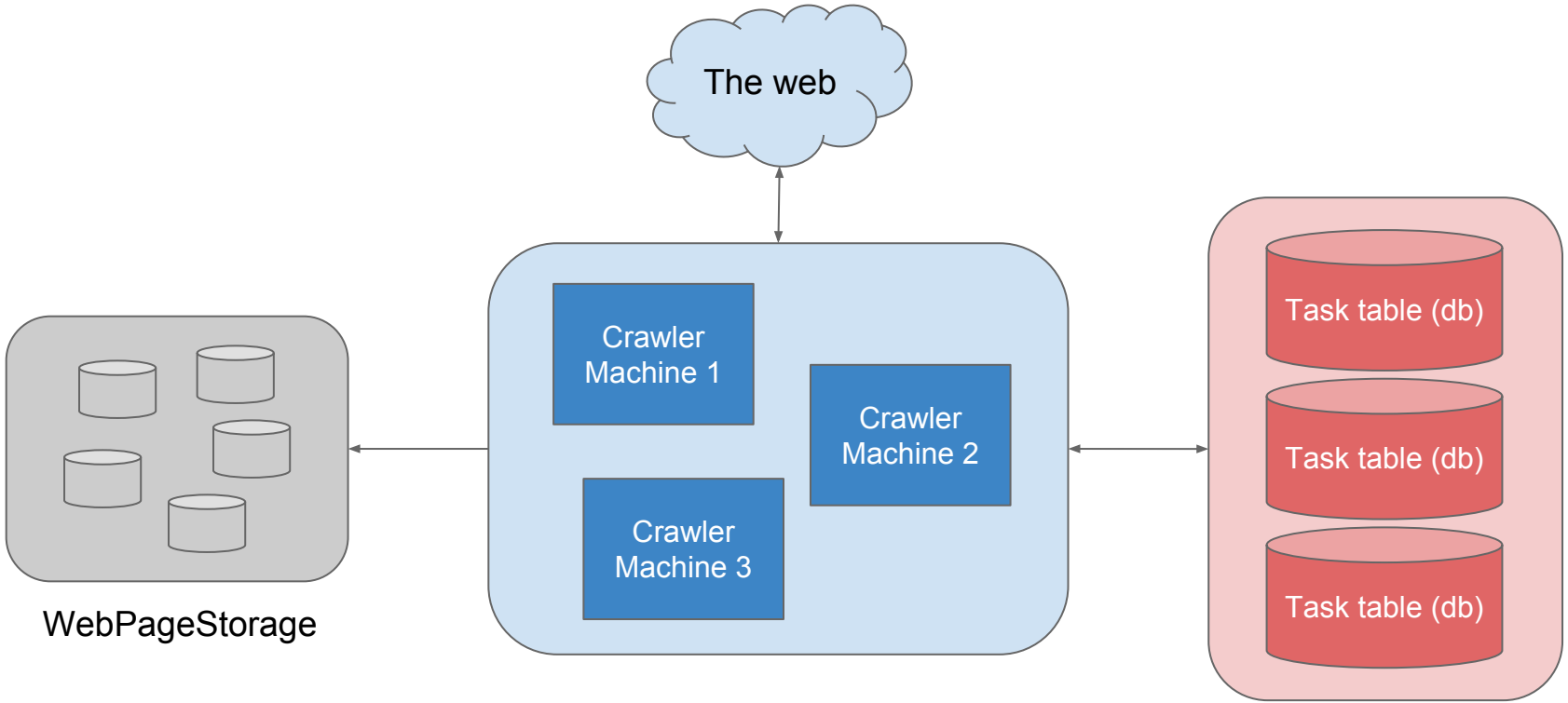
Design a web crawler

Scenario: How many web pages? how long? how large?

Service: Crawler, TaskService, StorageService

Storage: Use db to store tasks, BigTable to store web pages

Design a web crawler



Design a web crawler

A simplistic news crawler

A simplistic web crawler

A single-threaded web crawler

A multi-threaded web crawler

Dropbox interview question:

Program a web crawler, then make it multi-threaded

A Simplistic News Crawler

How a simplistic news crawler works

- given the URL of news list page

1. Send an HTTP request and grab the content of the news list page
2. Extract all the news titles from the news list page

A Simplistic News Crawler

九章算法

Input: URL of the news list page

http://tech.163.com/it

A Simplistic News Crawler

Grab the content of the page

```
import urllib2
url = 'http://tech.163.com/it'
request = urllib2.Request(url)
response = urllib2.urlopen(request)
page = response.read()
```

```
>>> request = urllib2.Request('http://www.baidu.com')
>>> response = urllib2.urlopen(request)
>>> response.read()
'<!DOCTYPE html><!--STATUS OK--><html><head><meta http-equiv="content-type" cont
ent="text/html;charset=utf-8"><meta http-equiv="X-UA-Compatible" content="IE=Edg
e"><meta content="always" name="referrer"><meta name="theme-color" content="#293
2e1"><link rel="shortcut icon" href="/favicon.ico" type="image/x-icon" /><link r
el="search" type="application/opensearchdescription+xml" href="/content-search.x
ml" title="\xe7\x99\xbe\xe5\xba\xa6\xe6\x90\x9c\xe7\xb4\xa2" /><link rel="icon"
sizes="any" mask href="//www.baidu.com/img/baidu.svg"><link rel="dns-prefetch" h
ref="//s1.bdstatic.com"/><link rel="dns-prefetch" href="//t1.baidu.com"/><link r
el="dns-prefetch" href="//t2.baidu.com"/><link rel="dns-prefetch" href="//t3.bai
du.com"/><link rel="dns-prefetch" href="//t10.baidu.com"/><link rel="dns-prefetc
h" href="//t11.baidu.com"/><link rel="dns-prefetch" href="//t12.baidu.com"/><lin
k rel="dns-prefetch" href="//b1.bdstatic.com"/><title>\xe7\x99\xbe\xe5\xba\xa6\x
e4\xb8\x80\xe4\xb8\x8b\xef\xbc\x8c\xe4\xbd\xa0\xe5\xb0\xb1\xe7\x9f\xa5\xe9\x81\x
93</title>\n<style index="index"  id="css_index">html,body{height:100%}html{over
flow-y:auto}body{font:12px arial;text-align:;background:#fff}body,p,form,ul,li{m
argin:0;padding:0;list-style:none}body,form,#fm{position:relative}td{text-align:
left}img{border:0}a{color:#00c}a:active{color:#f60}input{border:0;padding:0}#wra
```

A Simplistic News Crawler

九章算法

Extract all the news URLs from the news list page

Regular Expression

```
<h3[^>]*><a[^>]*>(.*?)<\/a><\/h3>
```

A Simplistic News Crawler

Output: a list of news titles

[

“富士康或将收购夏普交易推迟到下周”,

“美的董事长回应董明珠:怎么能说我们是骗子”,

“终于来了 亚马逊招开发经理打造VR平台”,

…

“Skylake Mac mini?小众用户也想要更强性能”

]

Design a web crawler

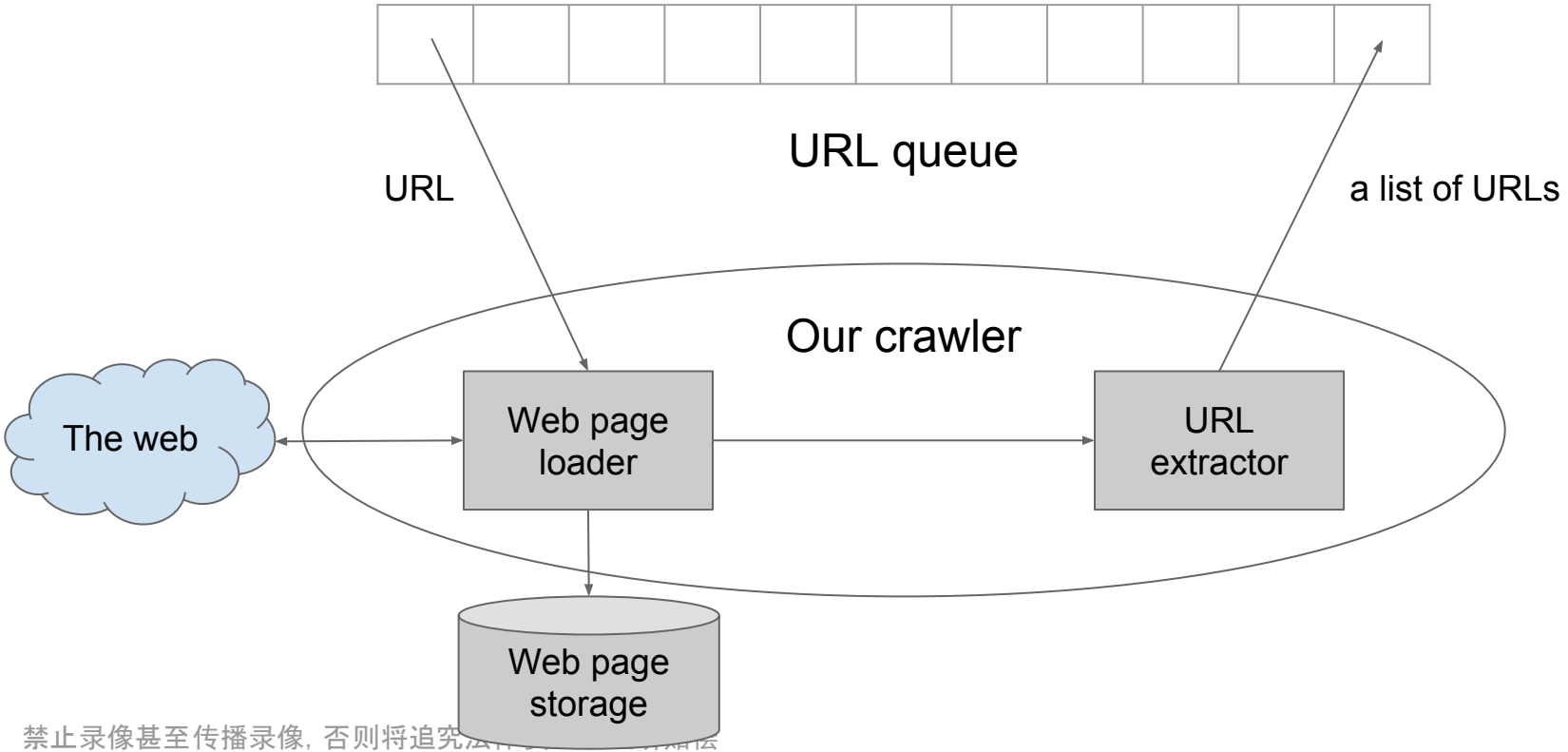
Dropbox interview question:

Program a web crawler, then make it multi-threaded

Input: url seeds

Output: list of urls

A Single-threaded Web Crawler

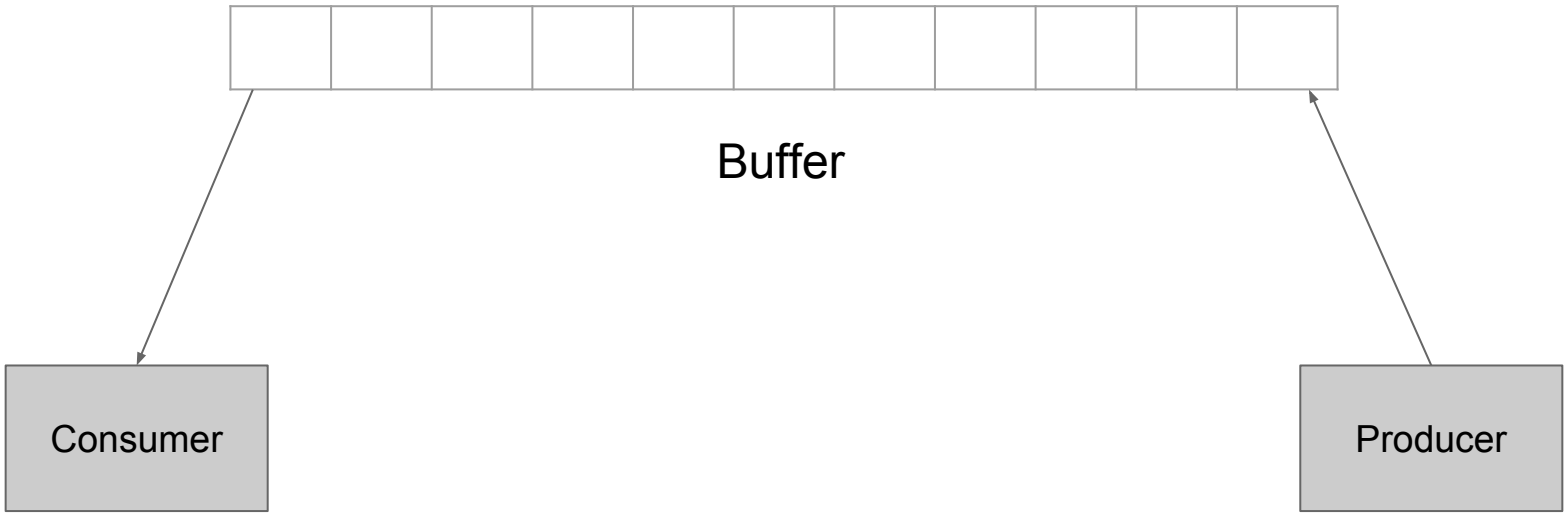


A Single-threaded Web Crawler


```
thread crawler
    function run
        while (url_queue not empty)
            url = url_queue.dequeue()
            html = web_page_loader.load(url) // consume
            url_list = url_extractor.extract(html) // produce
            url_queue.enqueue_all(url_list)
        end
```

A Single-threaded Web Crawler

Producer Consumer Pattern



Snapchat: write producer consumer

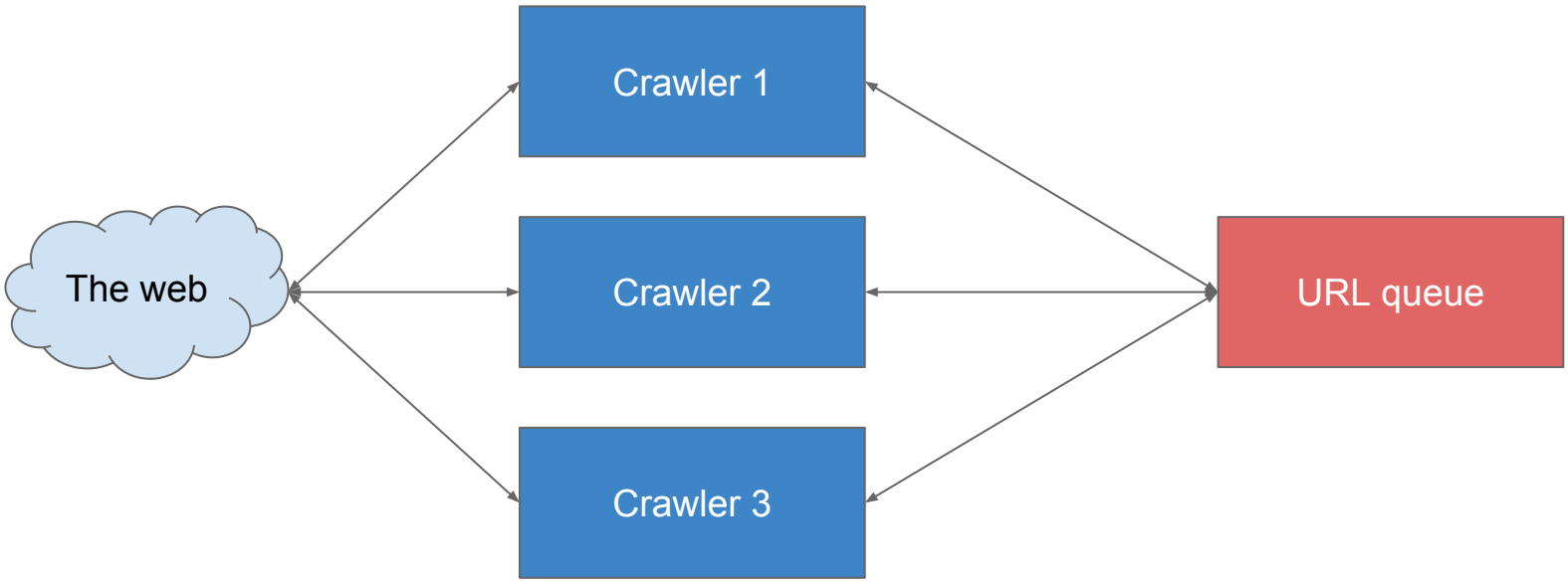
答案见附录

A Single-threaded Web Crawler

What's the problem of single thread?

Too slow?

A Multi-threaded Web Crawler



A Multi-threaded Web Crawler

How different threads work together?

Three approaches:

1. sleep
2. condition variable
3. semaphore

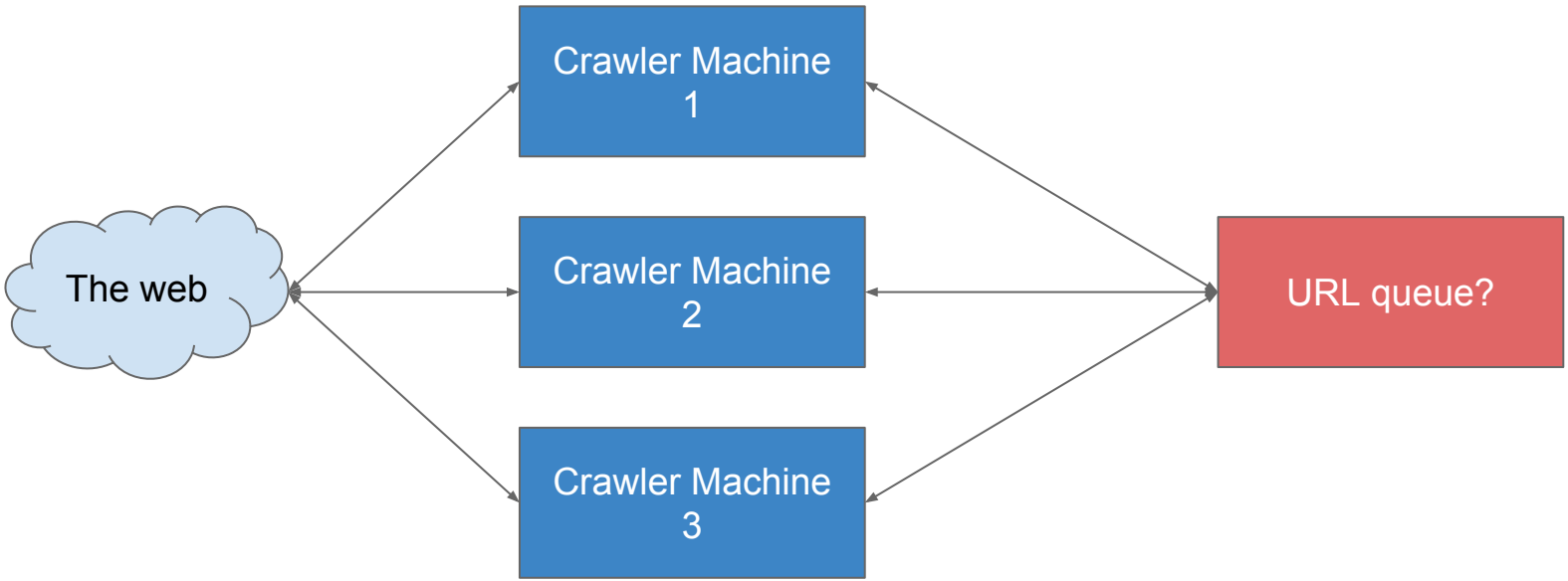
However, more threads doesn’t necessarily mean more performace

A Distributed Web Crawler

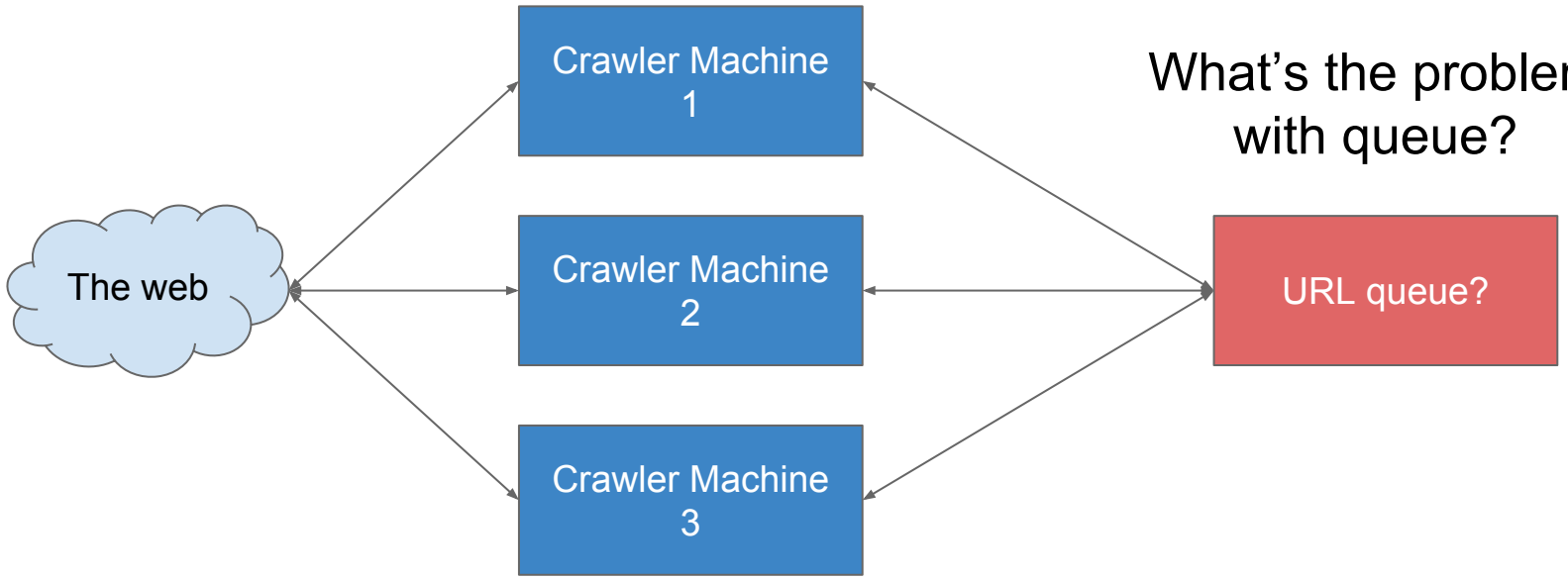
Why?

- context switch cost (CPU number limitation)
- thread (port) number limitation (16 bits = 65536)
- network bottleneck for single machine (10Gbps)

A Distributed Web Crawler



A Distributed Web Crawler



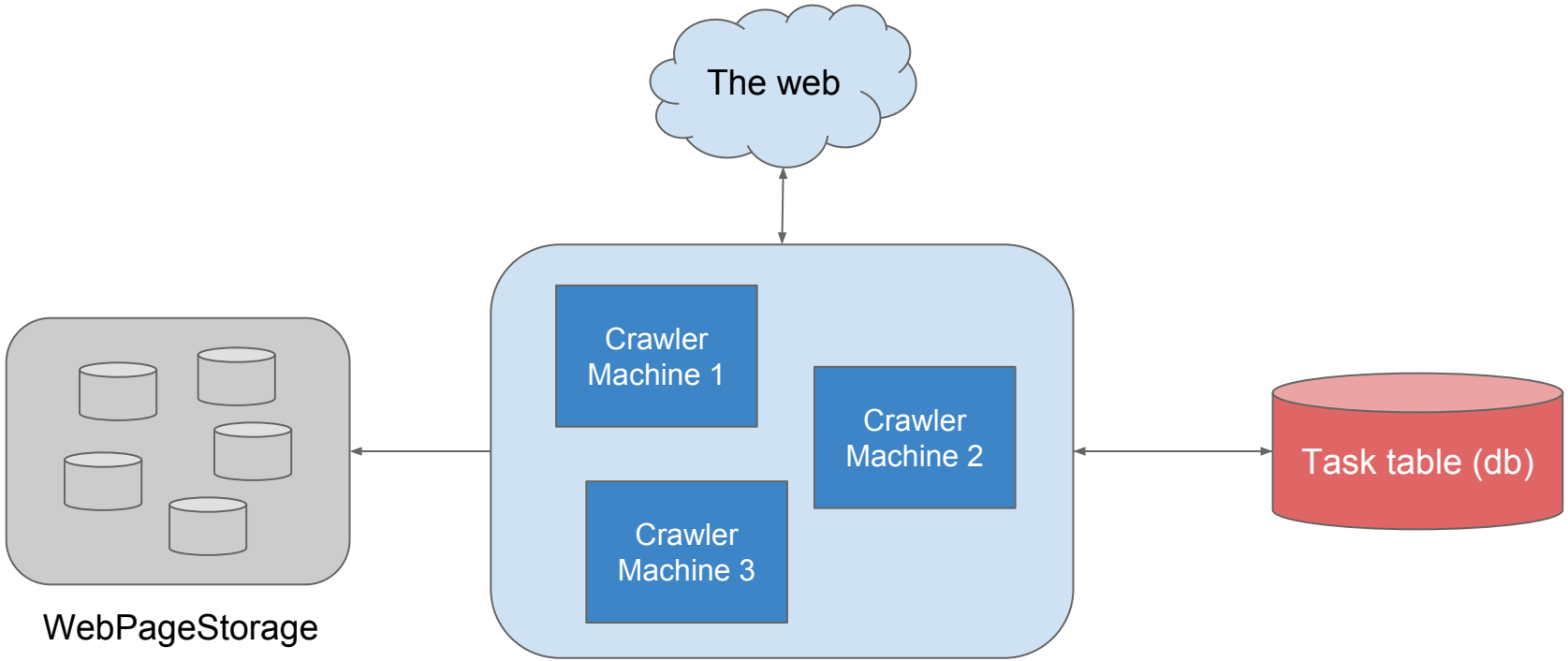
A Distributed Web Crawler

九章算法

How to design the task table

id	url	state	priority	available_time
1	“http://www.sina.com/”	“idle”	1	“2016-03-04 11:00 am”
2	“http://www.sina1.com/”	“working”	1	“2016-03-04 12:00 am”
3	“http://www.sina2.com/”	“idle”	0	“2016-03-14 02:00 pm”
4	“http://www.sina3.com/”	“idle”	2	“2016-03-12 04:25 am”
...	...	...	...	...

A Distributed Web Crawler



A Distributed Web Crawler

Now we have a work solution!

Scenario: How many web pages? how long? how large?

Service: Crawler, TaskService, StorageService

Storage: Use db to store tasks, BigTable to store web pages

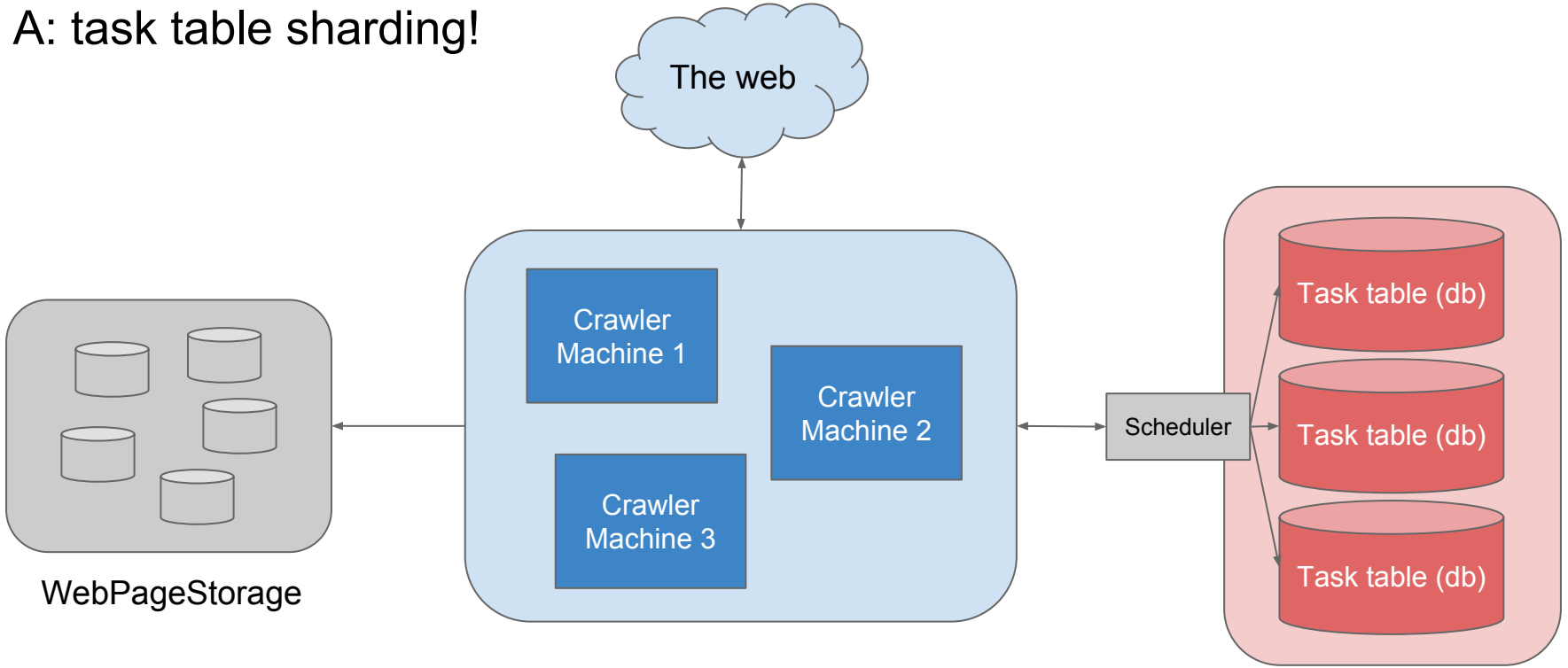
Let’s see how to scale!

Interviewer: How to handle slow select?

A Distributed Web Crawler

九章算法

A: task table sharding!



Interviewer: How to handle update for failure?

(i.e. content update, crawl failure)

A Distributed Web Crawler

Answer: Exponential back-off!

success: crawl after 1 week

no.1 failure: crawl after 2 week

no.2 failure: crawl after 4 weeks

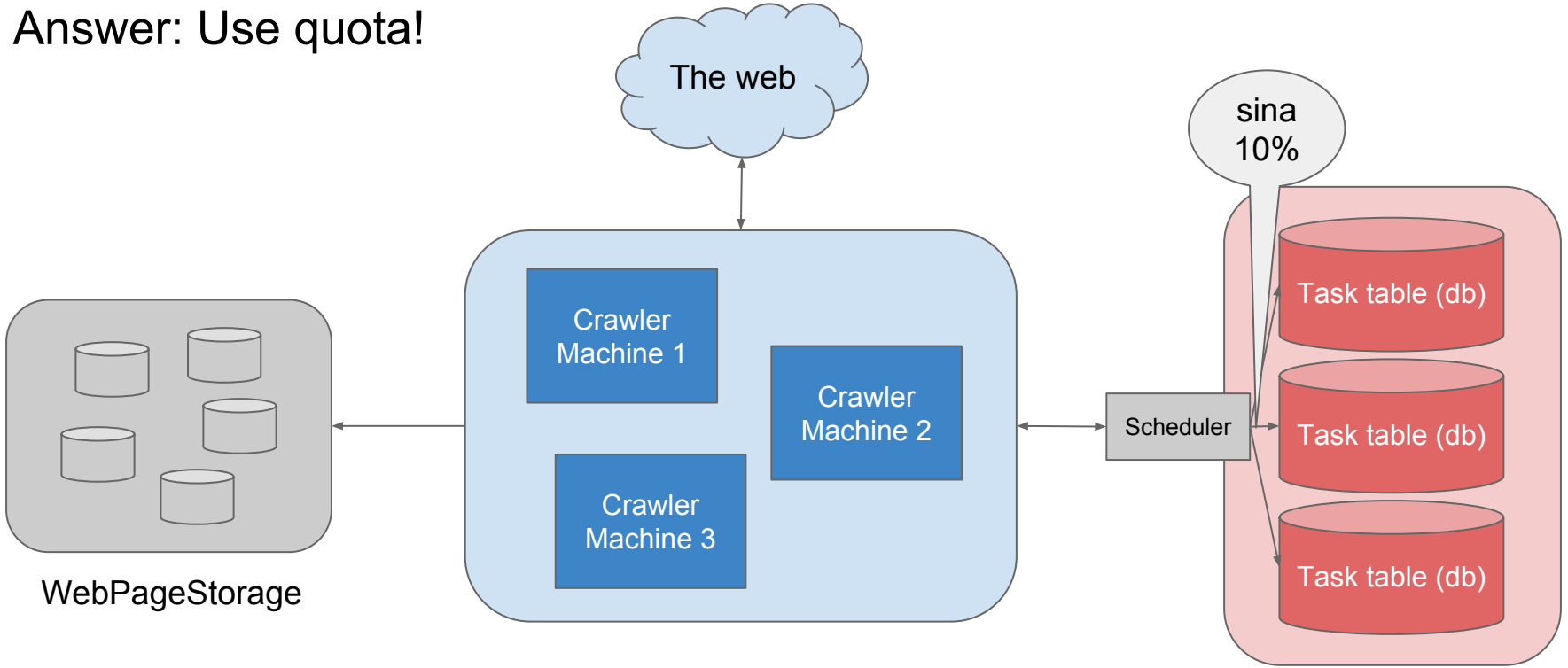
no.3 failure: crawl after 8 weeks ...

Interviewer: How to handle dead cycle?

(Too many web pages in sina.com, the crawler keeps crawling sina.com and don’t crawl other websites)

A Distributed Web Crawler

Answer: Use quota!



A Distributed Web Crawler

Scenario: How many web pages? how long? how large?

Service: Crawler, TaskService, StorageService

Storage: Use db to store tasks, BigTable to store web pages

Scale: single -> multi, multi -> distributed, queue -> table, slow select (db sharding), crawl failure/update handle, dead cycle (sina.com -> quota), multi-region

Design a Typeahead

九章算法

Interviewer: How to design a Typeahead?

Design a Typeahead

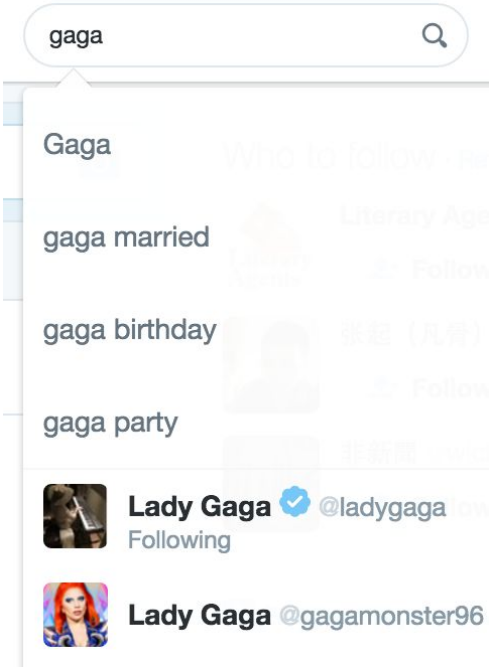
九章算法

What is Typeahead?

Design a Typeahead

九章算法

Twitter Typeahead



Design a Typeahead

Google suggestion

- prefix -> top n hot key words

Twitter typeahead

- suggestion + user + hashtag

Design a Typeahead

Google Suggestion

Scenario: prefix -> top n search keywords

DAU: 500m

Search: 6 * 6 * 500m = 18b

QPS = 18b / 86400 ≈ 200k

Peak QPS = QPS * 2 ≈ 400k

Design a Typeahead

Google Suggestion

Service

What service(s) do we need?

Design a Typeahead

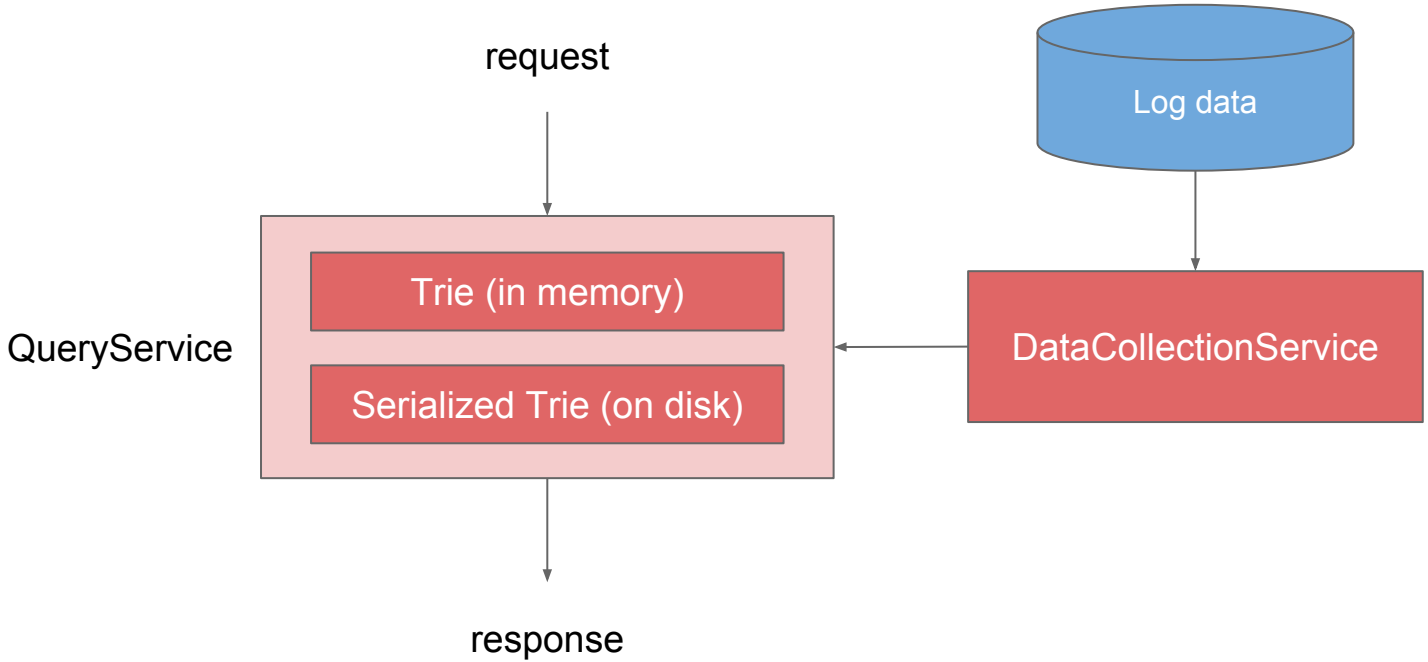
Google Suggestion

Service:

1. QueryService
2. DataCollectionService

Design a Typeahead

Google Suggestion



Design a Typeahead

九章算法

Google Suggestion

Storage

What storage do we need for QueryService?

Design a Typeahead

九章算法

Google Suggestion

Storage

1. QueryService

   what kind of data

   do we need to store?

   The naive way

keyword	hit_count
“amazon”	20b
“apple”	15b
“adidas”	7b
“airbnb”	3b
...	...

hit_stats

Design a Typeahead

How to query on the db?

Query payload: { key }

Query SQL:

```
SELECT * FROM hit_stats
WHERE keyword LIKE `${key}%`
ORDER BY hit_count DESC
LIMIT 10
```

keyword	hit_count
“amazon”	20b
“apple”	15b
“adidas”	7b
“airbnb”	3b
...	...

hit_stats

Design a Typeahead

Interviewer: What's the problem with this approach?

Design a Typeahead


```
SELECT * FROM hit_stats
WHERE keyword LIKE `${key}%`
ORDER BY hit_count DESC
LIMIT 10
```

LIKE operation is expensive!

```
WHERE keyword LIKE `abc%`
```

is equivalent to

```
WHERE keyword >= `abc` AND keyword < `abd`
```

Design a Typeahead

To reduce query time

Remember, this is database!

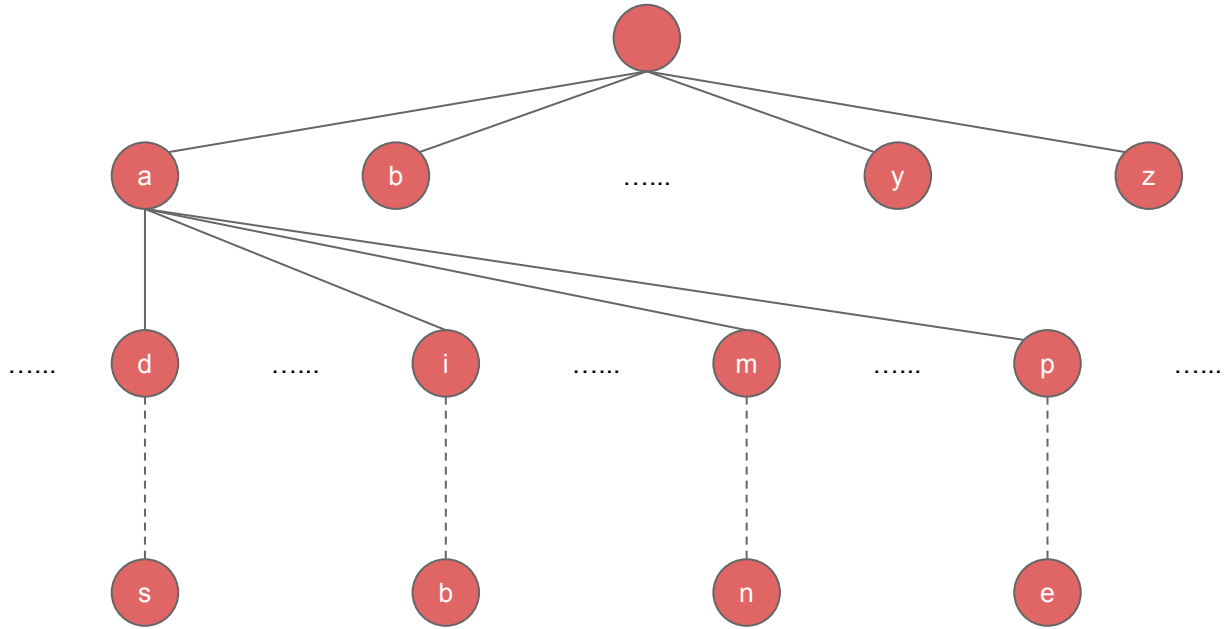
Which means we need cache!

Q: how do we generate this table given the table in the previous slide?

prefix	keywords
“a”	[“amazon”, “apple”, ...]
“am”	[“amazon”, “amc”, ...]
“ad”	[“adidas”, “adobe”, ...]
“don”	[“don’t have”, “donald trump”, ...]
...	...

Design a Typeahead

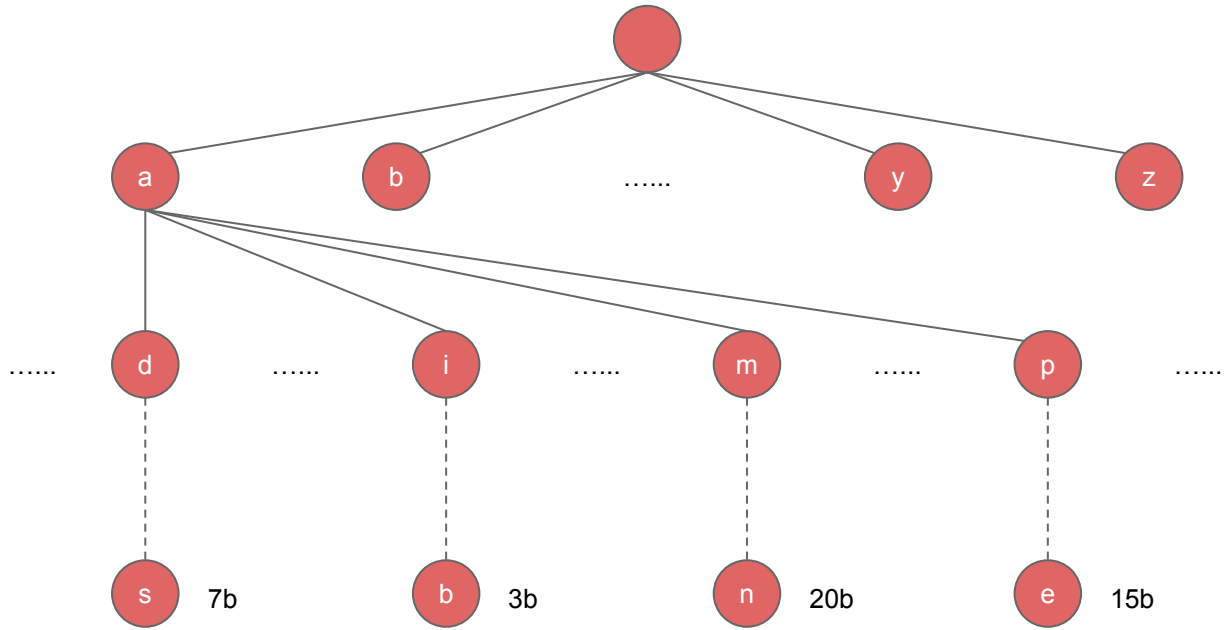
A better way: trie!



Design a Typeahead

We can store search count at node, but it's slow

keyword	hit_count
"amazon"	20b
"apple"	15b
"adidas"	7b
"airbnb"	3b
...	...

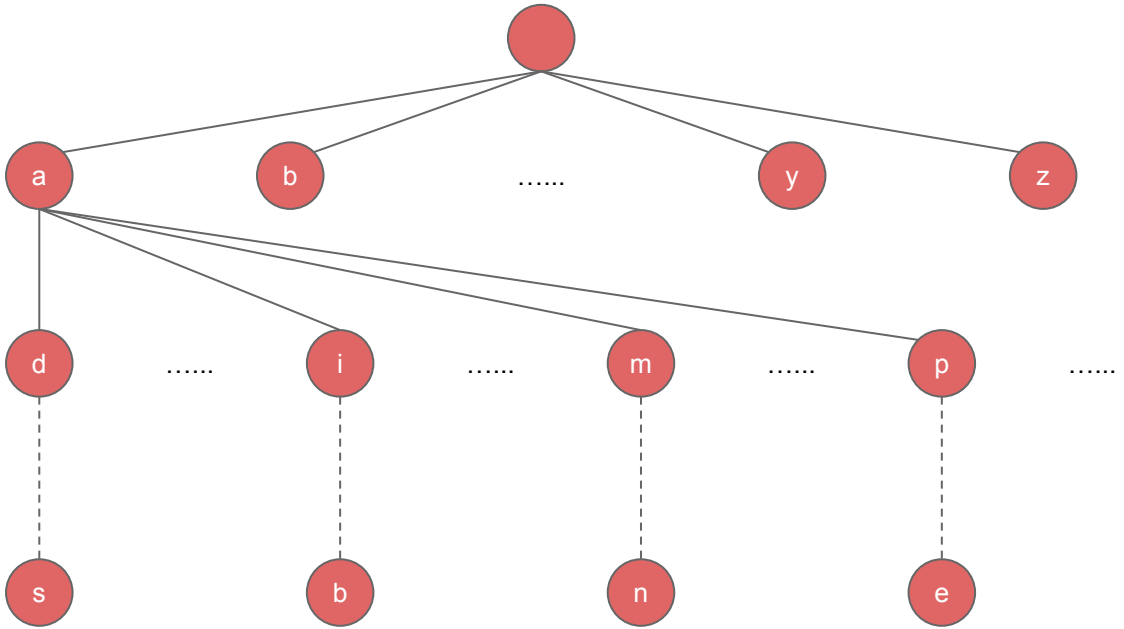


Design a Typeahead

Instead, we can store the top n hot key words, search becomes O(len)

```
[
  { adidas: 7b },
  { airbnb: 3b },
  { amazon: 20b },
  { apple: 15b },
  …
]
```

```
[
  { adidas: 7b },
  { adobe: 1b },
  { adele: 2b },
  { adblock: 1b },
  …
]
```

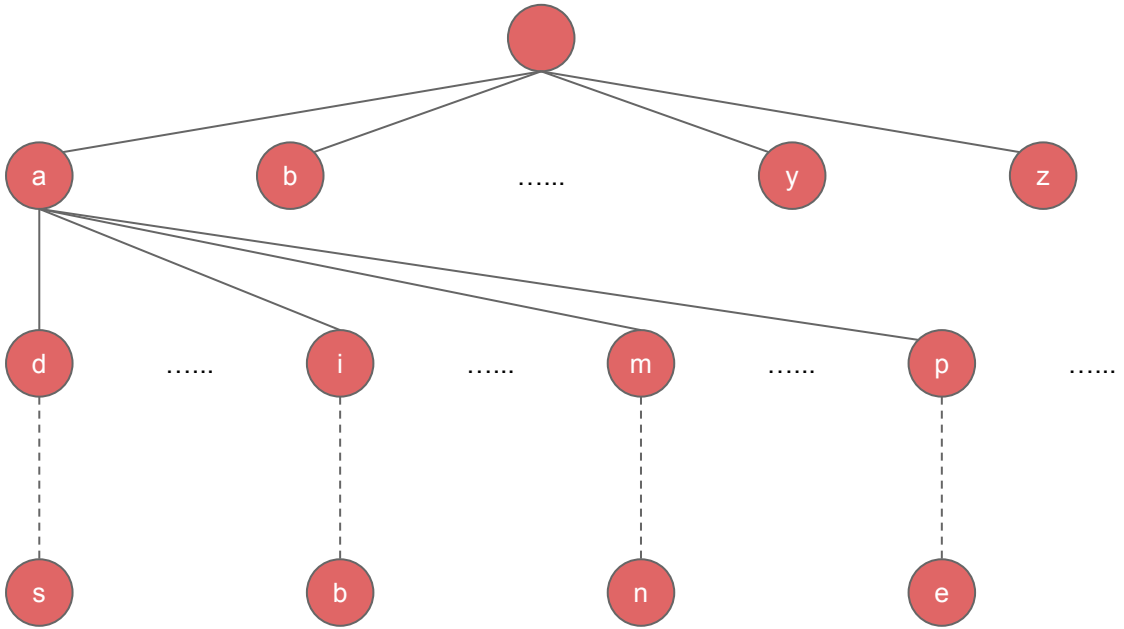


Design a Typeahead

How to we add a new record ({adb: 3b}) to the trie? See demo

```
[
  { adidas: 7b },
  { airbnb: 3b },
  { amazon: 20b },
  { apple: 15b },
  …
]
```

```
[
  { adidas: 7b },
  { adobe: 1b },
  { adele: 2b },
  { adblock: 1b },
  …
]
```



Design a Typeahead

Google Suggestion

Storage

What storage do we need for DataCollectionService?

Design a Typeahead

Interviewer:

Where does raw data come from?

keyword	hit_count
“amazon”	20b
“apple”	15b
“adidas”	7b
“airbnb”	3b
...	...

Design a Typeahead

Where does raw data come from?

(i.e. What does DataCollectionService do?)

user	keyword	time
xxx	“amazon”	1463379797
yyy	“apple”	1463379797
zzz	“adidas”	1463379798
...	...	...

log data

group by

keyword	hit_count
“amazon”	20b
“apple”	15b
“adidas”	7b
“airbnb”	3b
...	...

Design a Typeahead

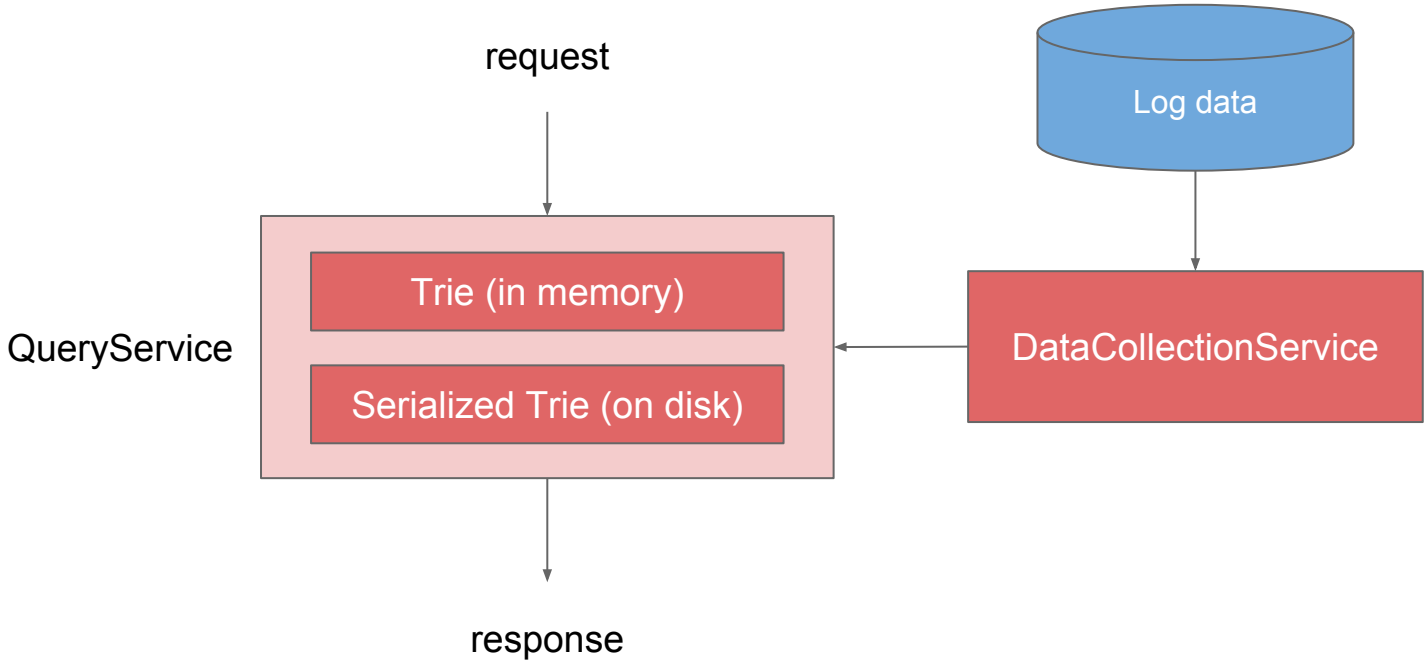
Google Suggestion

Storage

1. QueryService: in-memory trie along with disk serialization
2. DataCollectionService: BigTable

Design a Typeahead

Google Suggestion



Design a Typeahead

Now we have a work solution!

Design a Typeahead

九章算法

Interviewer: How to qualify this system?

Design a Typeahead

How to qualify this system?

Key metric: response time

Bottom line: result quality

Interviewer: How to reduce reponse time

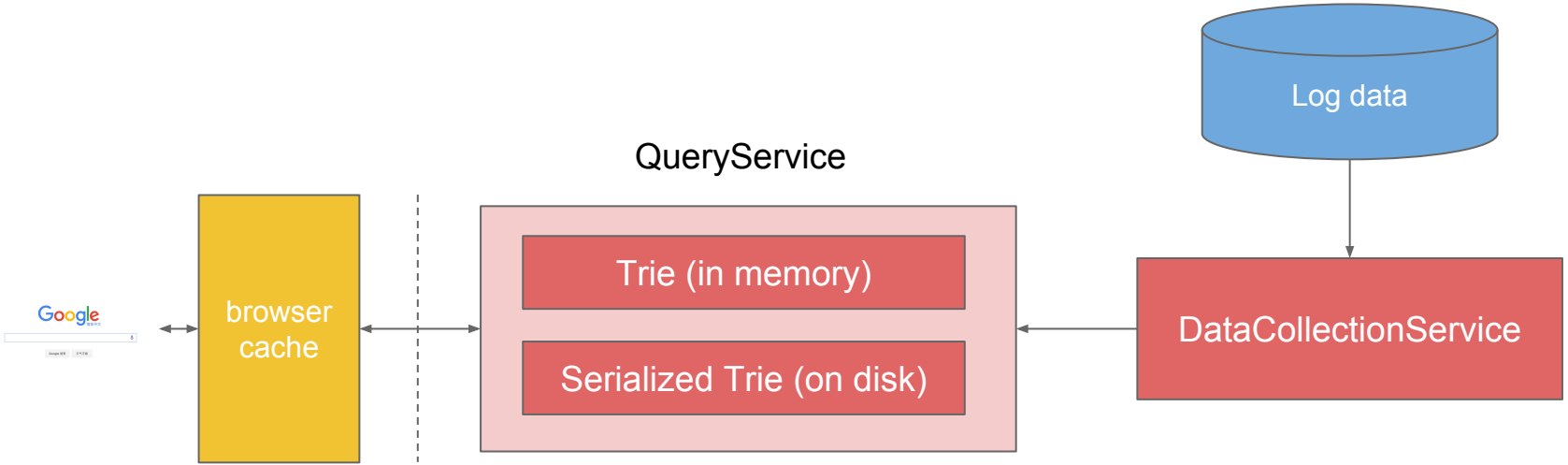
Design a Typeahead

How to reduce reponse time **in front-end (browser)**

1. cache result
2. pre-fetch

Design a Typeahead

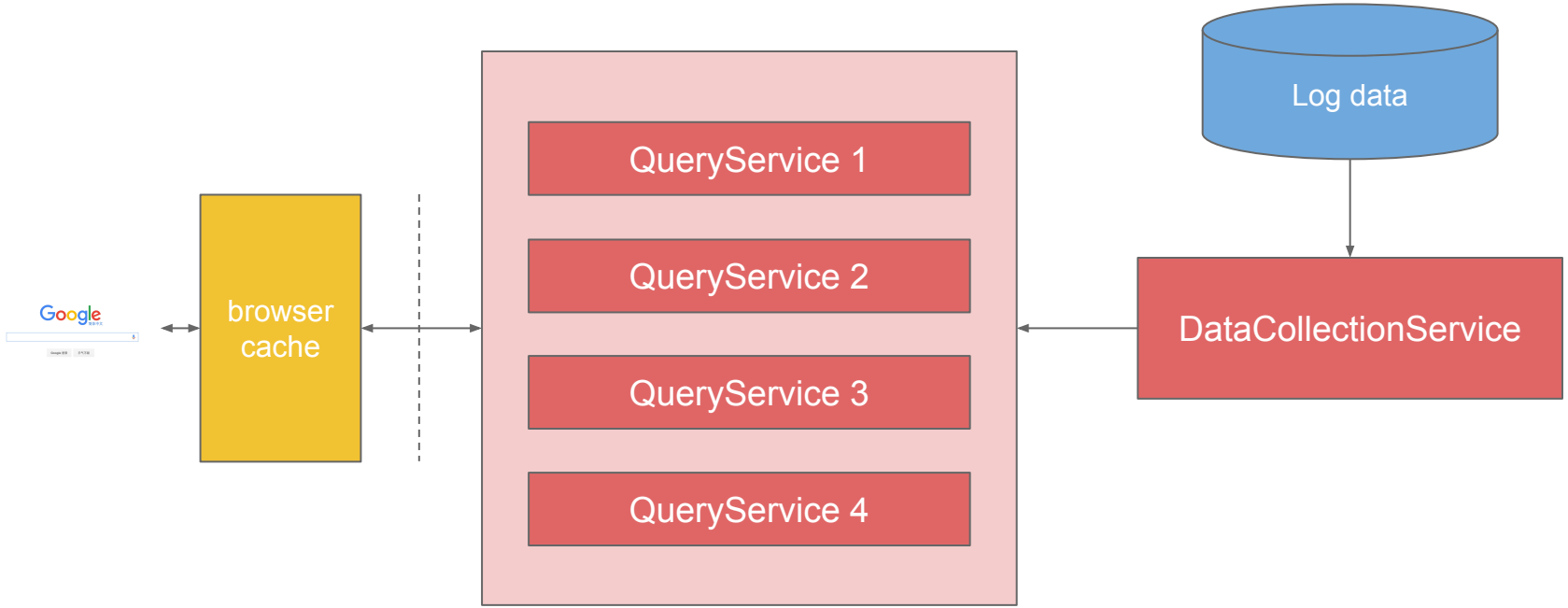
How to reduce reponse time **in front-end**



Interviewer: What if the trie gets too large for one machine?

Design a Typeahead

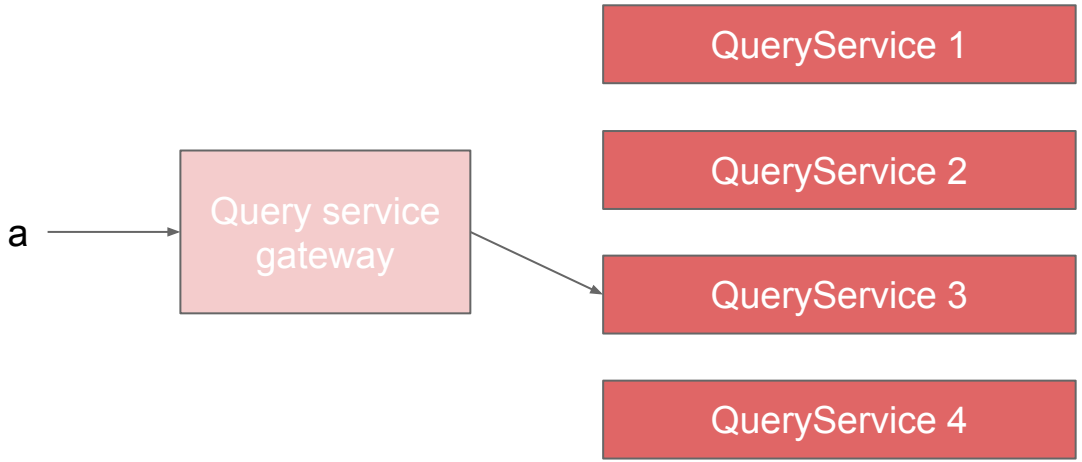
What if the trie gets too large for one machine



Design a Typeahead

How is trie stored across multiple machines?

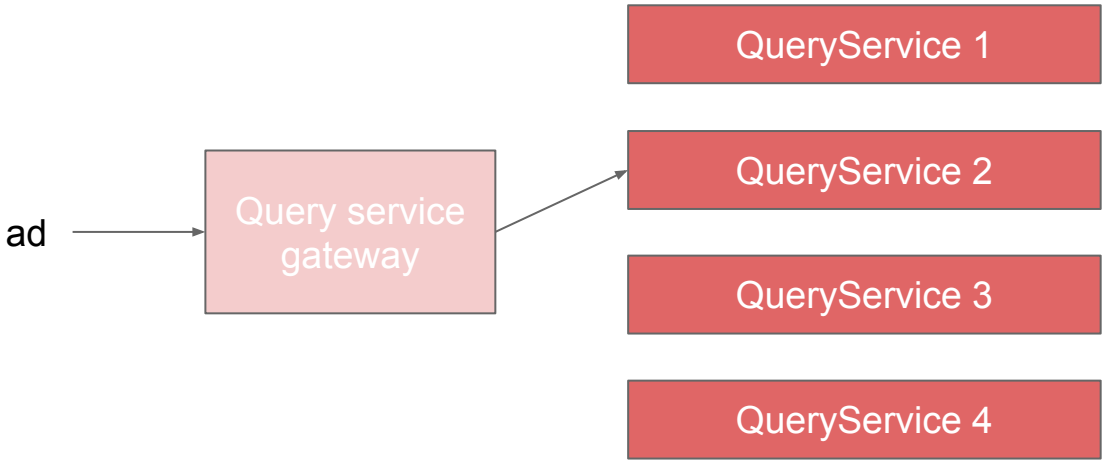
We use consistent hashing to decide which machine a particular string belongs to.



Design a Typeahead

How is trie stored across multiple machines?

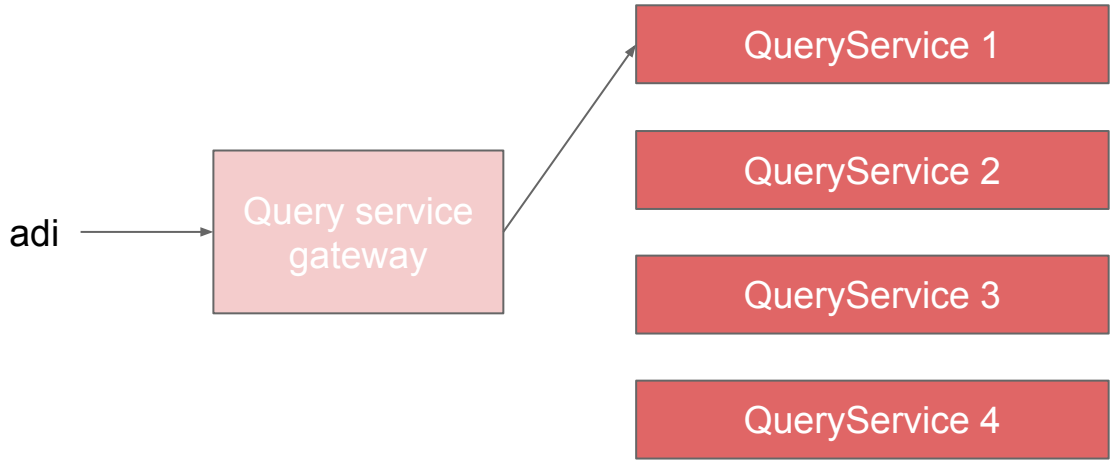
We use consistent hashing to decide which machine a particular string belongs to.



Design a Typeahead

How is trie stored across multiple machines?

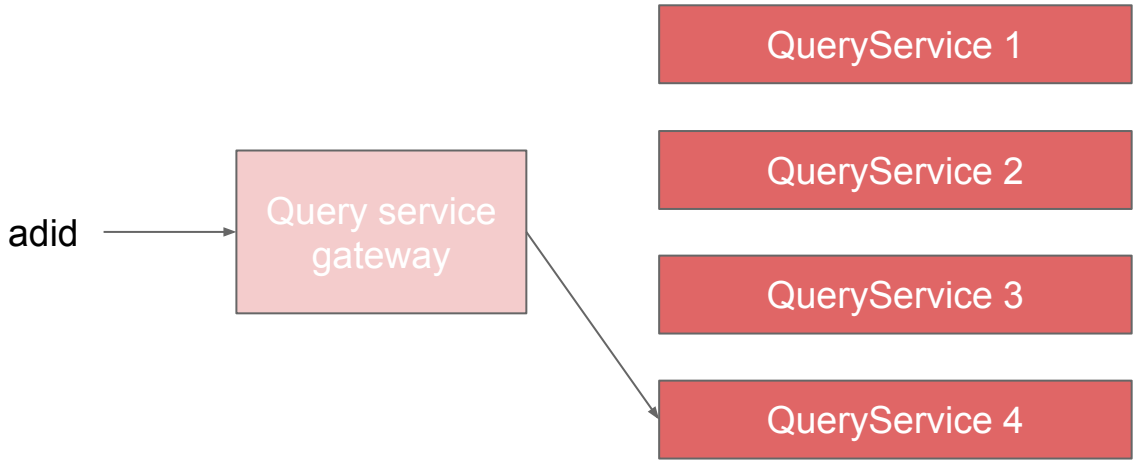
We use consistent hashing to decide which machine a particular string belongs to.



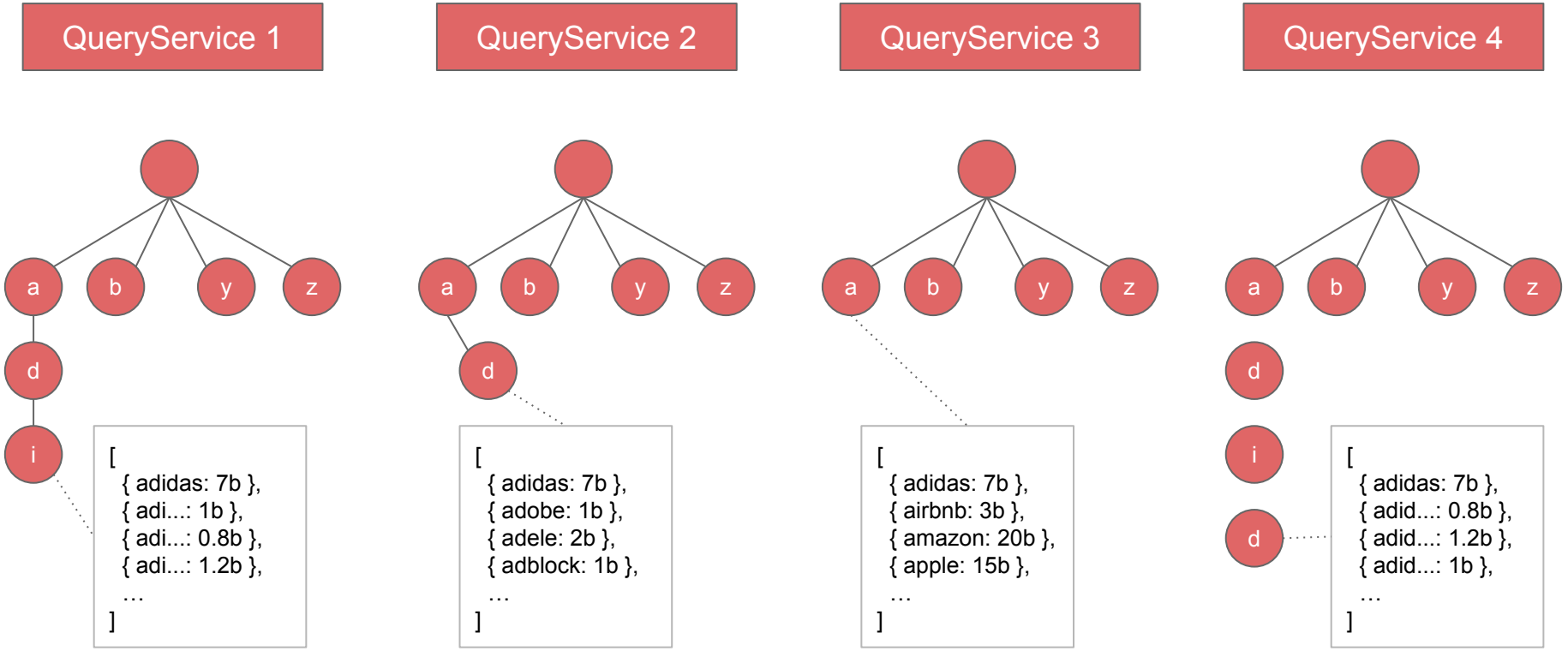
Design a Typeahead

How is trie stored across multiple machines?

We use consistent hashing to decide which machine a particular string belongs to.



Design a Typeahead



Design a Typeahead

How is trie stored across multiple machines?

We use consistent hashing to decide which machine a particular string belongs to.

Interviewer: How to reduce the size of log file?

Design a Typeahead

How to reduce the size of log file

Probabilistic logging

Log with 1 / 10,000 probability

Appendix

课后练习

- http://www.lintcode.com/en/problem/url-parser/
- http://www.lintcode.com/en/problem/implement-trie/
- http://www.lintcode.com/en/problem/trie-serialization/
- http://www.lintcode.com/en/problem/typeahead/
- http://www.lintcode.com/en/problem/webpage-crawler/

Appendix

自学材料

- http://www.codeguru.com/cpp/sample_chapter/article.php/c13533/Why-Too-Many-Threads-Hurts-Performance-and-What-to-do-About-It.htm
- http://agiliq.com/blog/2013/10/producer-consumer-problem-in-python/

谢谢大家