SYSTEMS ENGINEERING

Prof. Christof Fetzer, TU Dresden

SE Agenda

- Motivation / Scalability
- Consensus
- Paxos
- Raft
- CAP Theorem: DynamoDB vs Spanner
- Chubby and K8s
- Memcached / REDIS
- Object Store: S3 and Ceph
- Virtual Time / Vector Time / Real Time
- DynamoDB (eventual consistency)
- Spanner

Administrative Stuff

- Slide decks, calendar, etc on Opal
- **Exams**:
  - written or oral depending on program
- **Exercises**:
  - on Fridays

A SCALABLE ARCHITECTURE

SEARCH QUERIES

- Search queries in 2011:
  - **Google**: about **34,000** queries per second (1-3 billions per day)
  - **Yahoo**: about 3,200 queries per second
  - **Bing**: about 930 queries per second
- Search queries in Sept 2019:
  - Google: 74% (desktop), 92% (mobile)
  - Bing: 11.4% (desktop), 1% (mobile)
  - Yahoo: 2.2% (desktop), 1.3% (mobile)
- 2023 (all platforms):
  - Google (92%), Bing (3%), Yahoo (1,2%)

SERVING A SEARCH QUERY

(Google)

PROBLEMS

- Need to use energy-efficient, low-cost CPUs
  - low peak performance
- Need to guarantee fast response times
  - to keep users happy
- **Need to parallelize queries**
  - since there are tens of billions instructions per query to execute

HOW PARALLEL?

- **Rough estimate:**
  - **34,000** queries per second and say, > 8 seconds per query (sequential execution)
- **Hence**,
  - one would need to involve > **34,000** computers (with 8 cores each)
- **Notes**:
  - a very rough estimate since, e.g., caching of results might reduce the number of computers, etc
  - might be more limited by memory than CPU speed
  - is this the average or the peak queries per second?

QUERY (2007)

1. www.google.com/search?q=systems+engineering

2. Browser resolves www.google.com

```
www.google.com is an alias for www.l.google.com.
www.l.google.com has address 64.233.183.104
www.l.google.com has address 64.233.183.147
www.l.google.com has address 64.233.183.99
www.l.google.com has address 64.233.183.103
```

(located in the US)

QUERY (2007)

URI - Uniform Resource Identifier

1. www.google.com/search?q=systems+engineering

2. Browser resolves www.google.com

```
www.google.com is an alias for www.l.google.com.
www.l.google.com has address 64.233.183.104
www.l.google.com has address 64.233.183.147
www.l.google.com has address 64.233.183.99
www.l.google.com has address 64.233.183.103
```

(located in the US)

QUERY (2007)

URI - Uniform Resource Identifier

1. www.google.com/search?q=systems+engineering

Query parameters are key-value pairs (e.g., k=v) added to the end of a URI, typically after a question mark

2. Browser resolves www.google.com

```
www.google.com is an alias for www.l.google.com.
www.l.google.com has address 64.233.183.104
www.l.google.com has address 64.233.183.147
www.l.google.com has address 64.233.183.99
www.l.google.com has address 64.233.183.103
```

(located in the US)

QUERY (2007)

1. www.google.com/search?q=systems+engineering

Need to translate www.google.com to an IP address - using DNS

2. Browser resolves www.google.com

```
www.google.com is an alias for www.l.google.com.
www.l.google.com has address 64.233.183.104
www.l.google.com has address 64.233.183.147
www.l.google.com has address 64.233.183.99
www.l.google.com has address 64.233.183.103
```

(located in the US)

QUERY (2007)

1. www.google.com/search?q=systems+engineering

www.google.com is a CNAME that maps to www.l.google.com

2. Browser resolves www.google.com

```
www.google.com is an alias for www.l.google.com.
www.l.google.com has address 64.233.183.104
www.l.google.com has address 64.233.183.147
www.l.google.com has address 64.233.183.99
www.l.google.com has address 64.233.183.103
```

(located in the US)

QUERY (2007)

1. www.google.com/search?q=systems+engineering

www.l.google.com maps to multiple IP addresses

2. Browser resolves www.google.com

```
www.google.com is an alias for www.l.google.com.
www.l.google.com has address 64.233.183.104
www.l.google.com has address 64.233.183.147
www.l.google.com has address 64.233.183.99
www.l.google.com has address 64.233.183.103
```

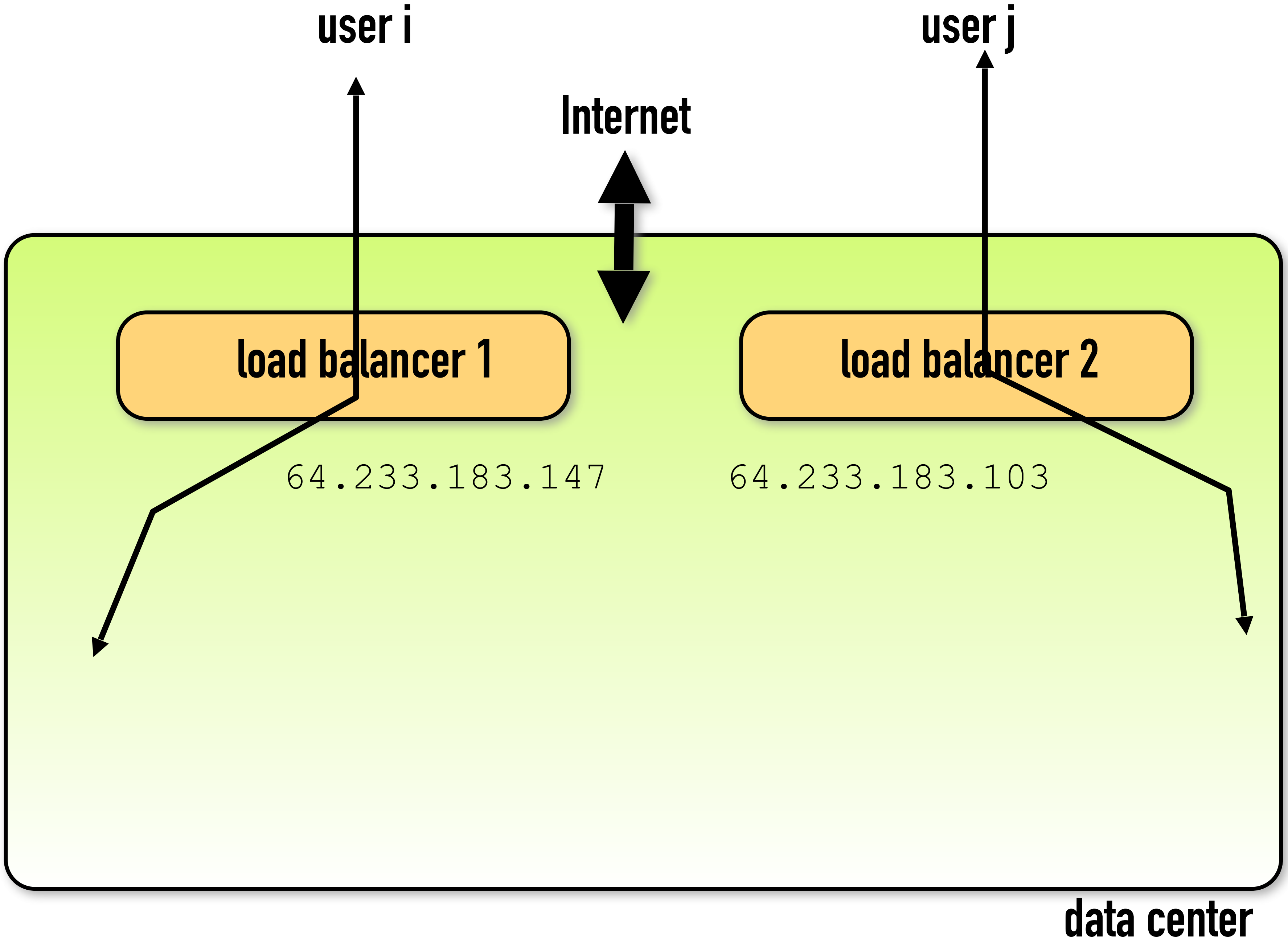
(located in the US)

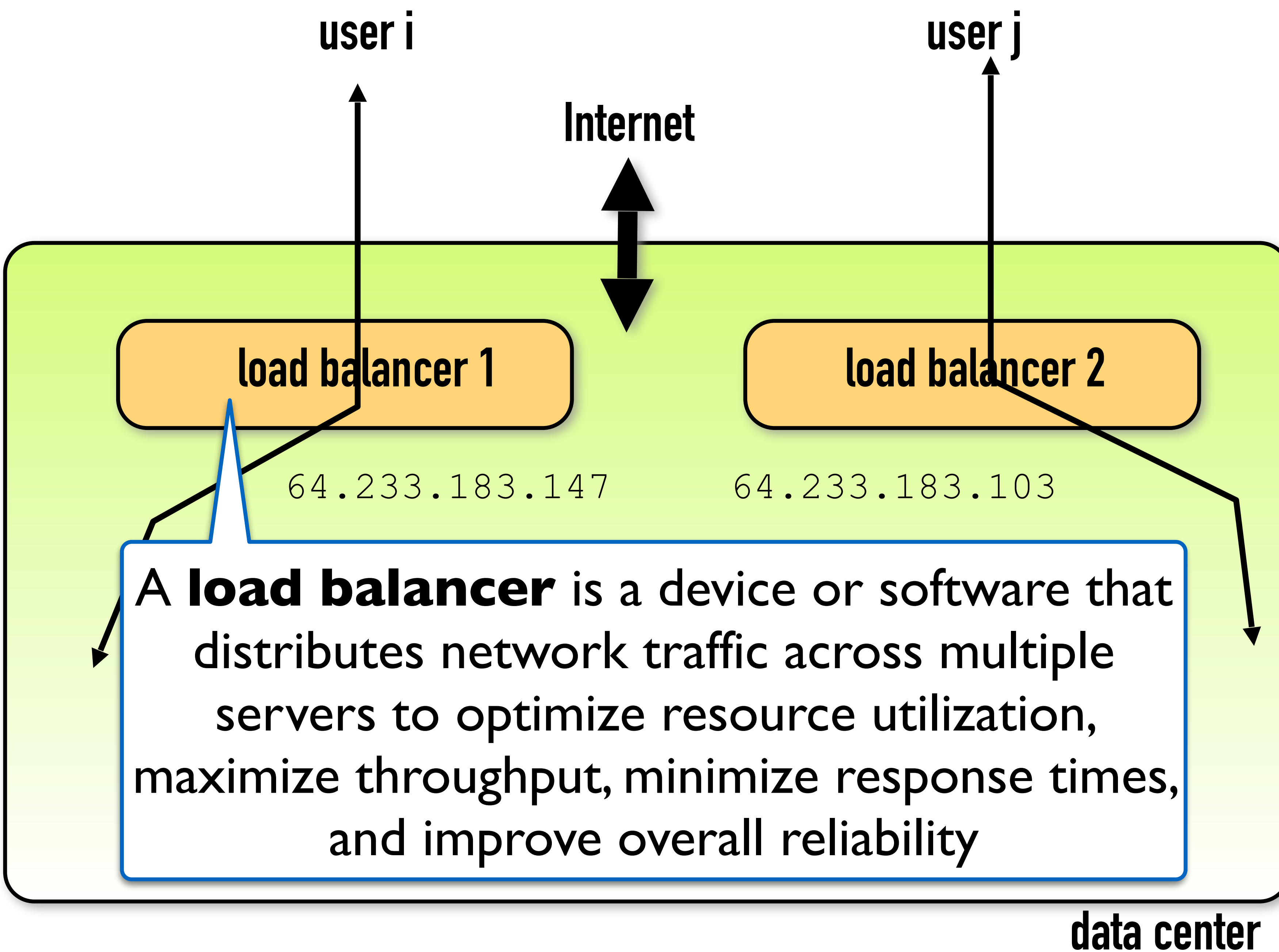
ROUND-ROBIN DNS

- DNS rotate results:

```
www.l.google.com has address 64.233.183.147
www.l.google.com has address 64.233.183.99
www.l.google.com has address 64.233.183.103
www.l.google.com has address 64.233.183.104
```

IP addresses rotate in round-robin fashion


user i
user j
Internet
load balancer 1
load balancer 2
64.233.183.147
64.233.183.103
data center



A **load balancer** is a device or software that distributes network traffic across multiple servers to optimize resource utilization, maximize throughput, minimize response times, and improve overall reliability

IN 2009, 2010:

Browser resolves www.google.com

www.google.com is an alias for www.l.google.com

www.l.google.com has address 74.125.39.103

www.l.google.com has address 74.125.39.105

www.l.google.com has address 74.125.39.106

www.l.google.com has address 74.125.39.147

www.l.google.com has address 74.125.39.104

www.l.google.com has address 74.125.39.99

(located in Berlin)

5 MINUTES LATER:

Browser resolves www.google.com

```
www.google.com is an alias for www.l.google.com
www.l.google.com has address 209.85.129.103
www.l.google.com has address 209.85.129.147
www.l.google.com has address 209.85.129.147
www.l.google.com has address 209.85.129.99
```

(located in Mountain View, CA)

DNS

- Google's DNS servers used to map www.google.com on IP addresses based on
  - geographical location of user
    -> minimize round-trip times
  - load of the individual Google clusters
    -> coarse grain load balancing

2013/4

```
www.google.com has address 173.194.69.106
www.google.com has address 173.194.69.99
www.google.com has address 173.194.69.147
www.google.com has address 173.194.69.104
www.google.com has address 173.194.69.105
www.google.com has address 173.194.69.103
www.google.com has IPv6 address
2a00:1450:4008:c01::63
```

(located in Mountain View, CA?)

2013/4

```
www.google.com has address 173.194.69.106
www.google.com has address 173.194.69.99
www.google.com has address 173.194.69.147
www.google.com has address 173.194.69.104
www.google.com has address 173.194.69.105
www.google.com has address 173.194.69.103
www.google.com has IPv6 address
2a00:1450:4008:c01::63
```

DNS also returns an **IPv6** address

(located in Mountain View, CA?)

2019–2024: SINGLE ADDRESS

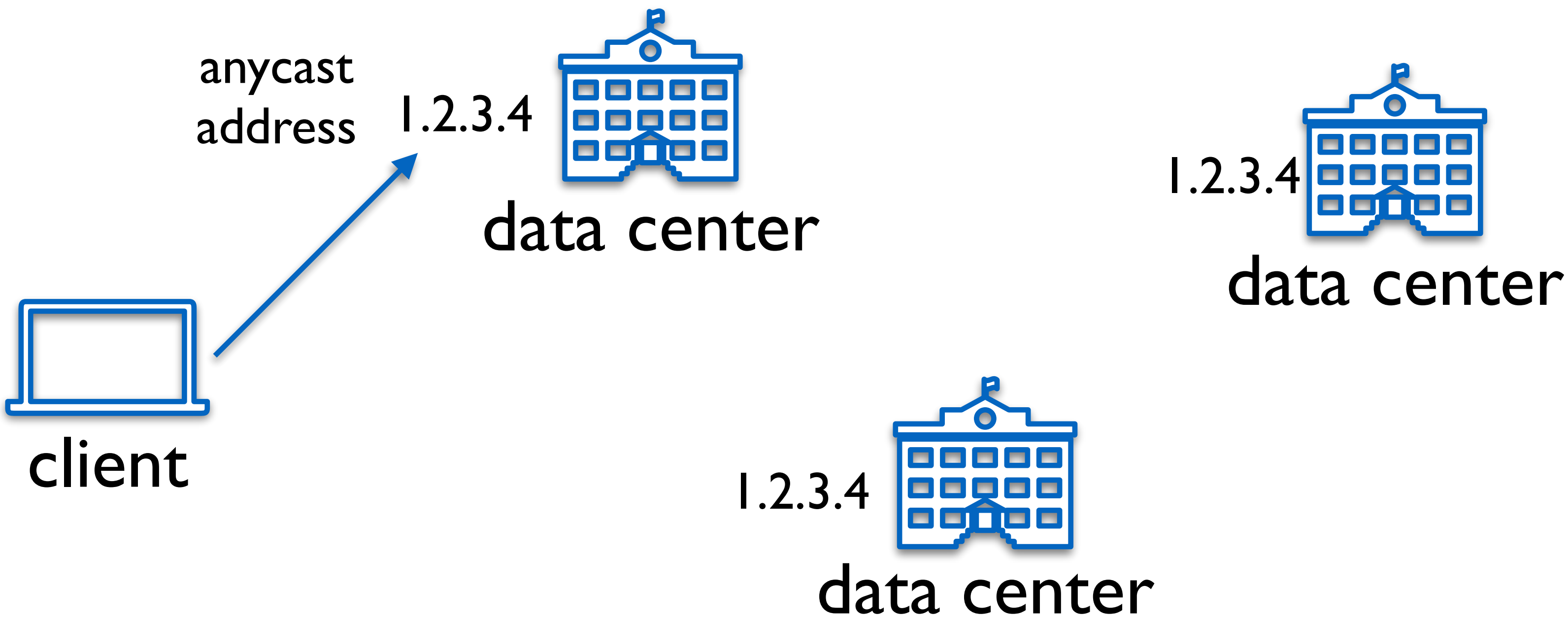
```
www.google.com has address 216.58.207.68


www.google.com has IPv6 address
                 2a00:1450:400e:800::2004
```

IPv6 address the same but IPv4 has changed in 2024

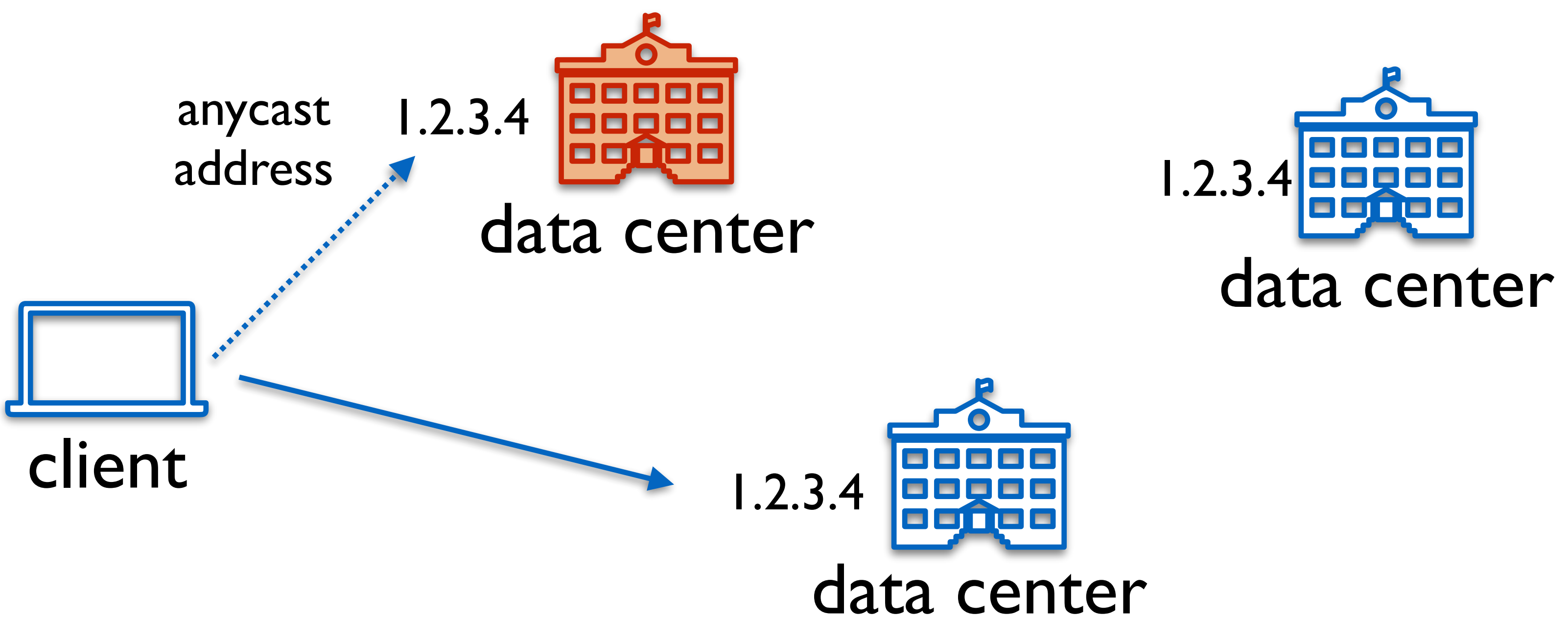
Anycast

- Ensure that load is routed to closest data center (a „1 to 1 routing scheme“)
    - minimizing Round Trip Time (RTT)



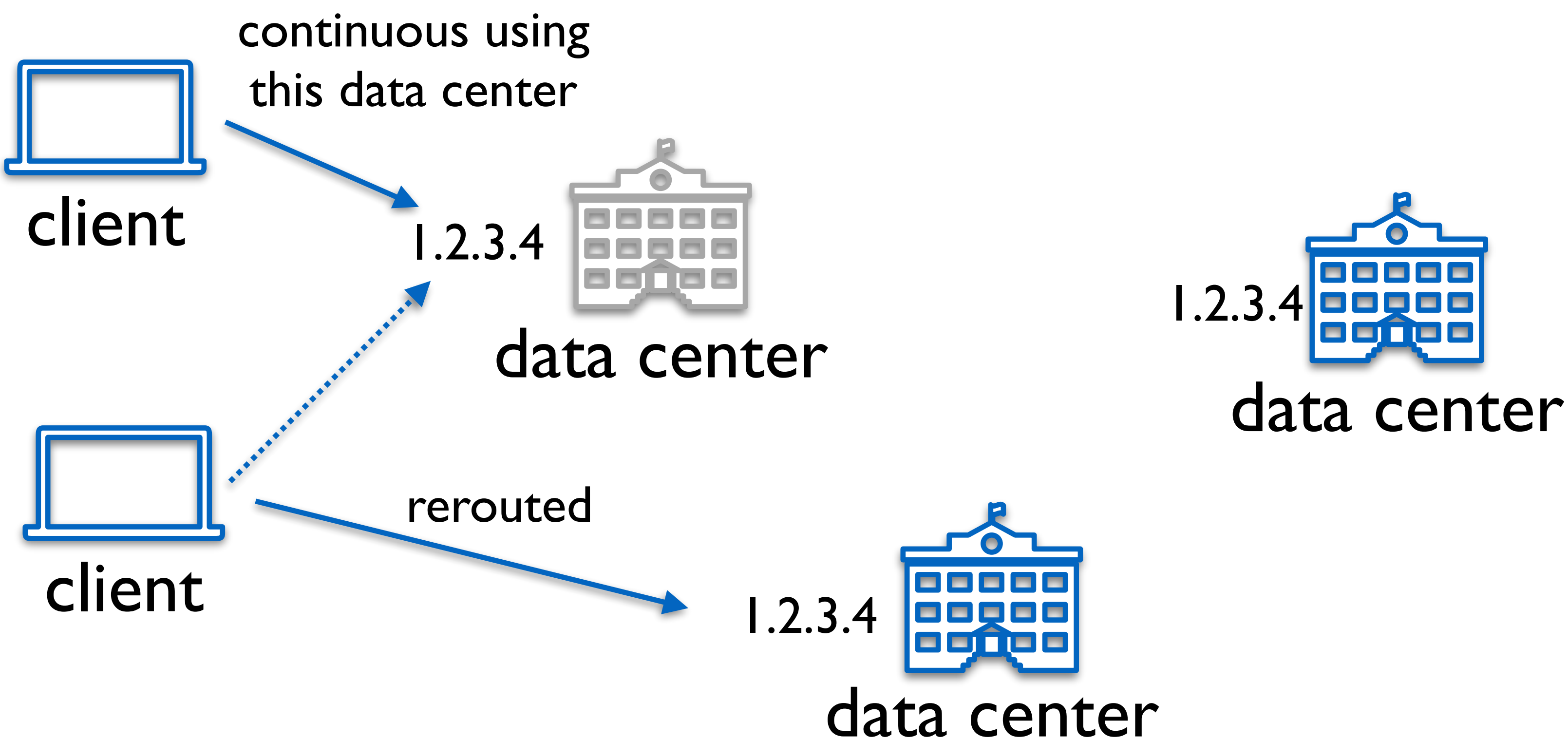
Rerouting

- In case closest data center is overloaded, traffic is rerouted to another datacenter



Load Balancing

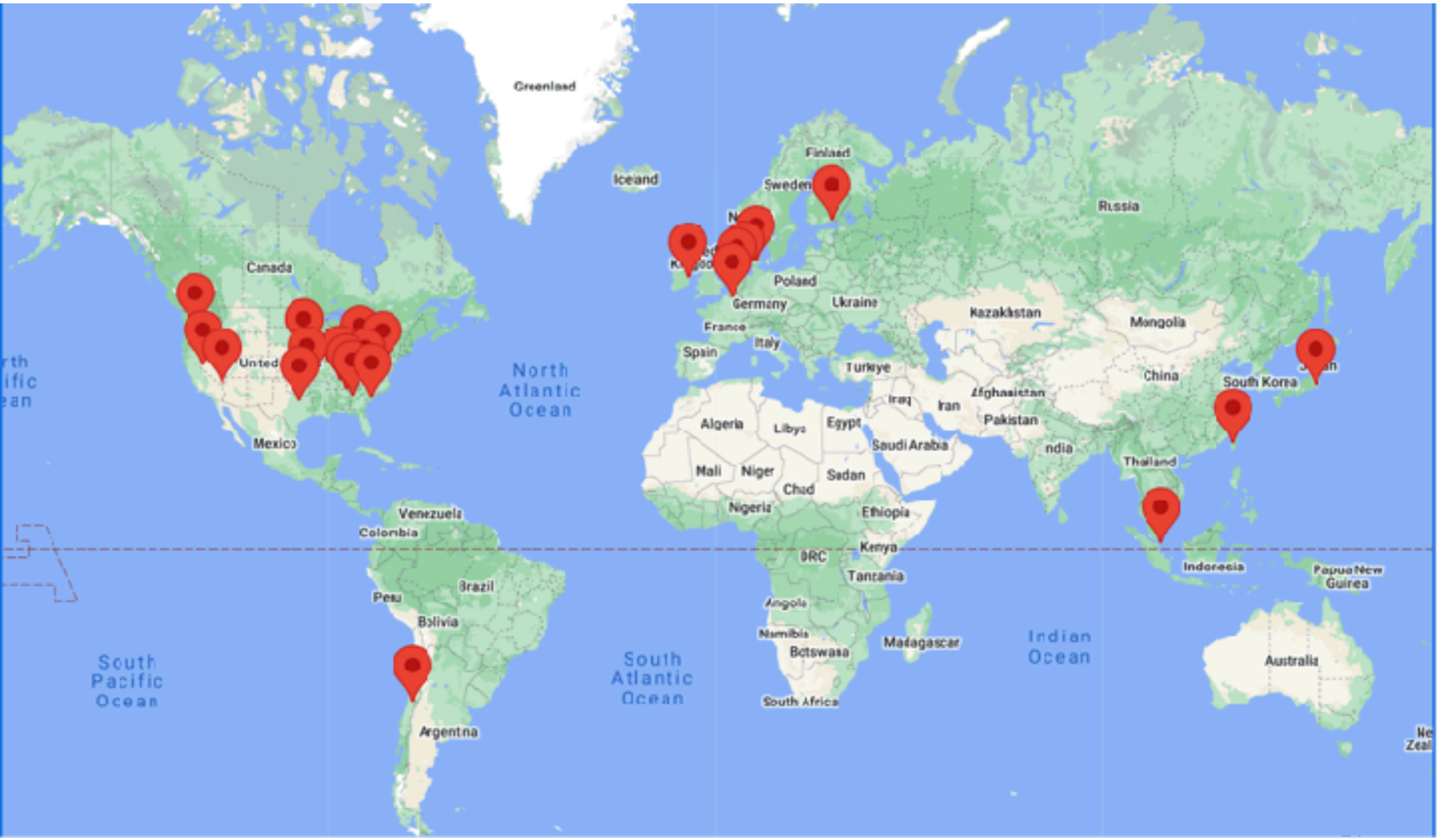
- One might only reroute parts of the traffic to keep data centers from being overloaded



Technical Challenges

- Data to compute answer must be accessible in all data centers
- **Google search:**
  - need to replicate the index for searching
  - personalized search - one needs to keep personalized data accessible in all data centers?
- Mail / Documents / …:
  - need to keep data available in all data centers?

GOOGLE DATA CENTERS – GEOGRAPHICALLY DISTRIBUTED



GOOGLE'S DATACENTERS

- Google has geographically distributed
  - data centers consisting of thousands of servers each
  - 2009 guess: about 1million servers total - now ??

(Google data center, Oregon US, (C) New York Times)

GOOGLE'S DATACENTERS

Look at electricity consumption?

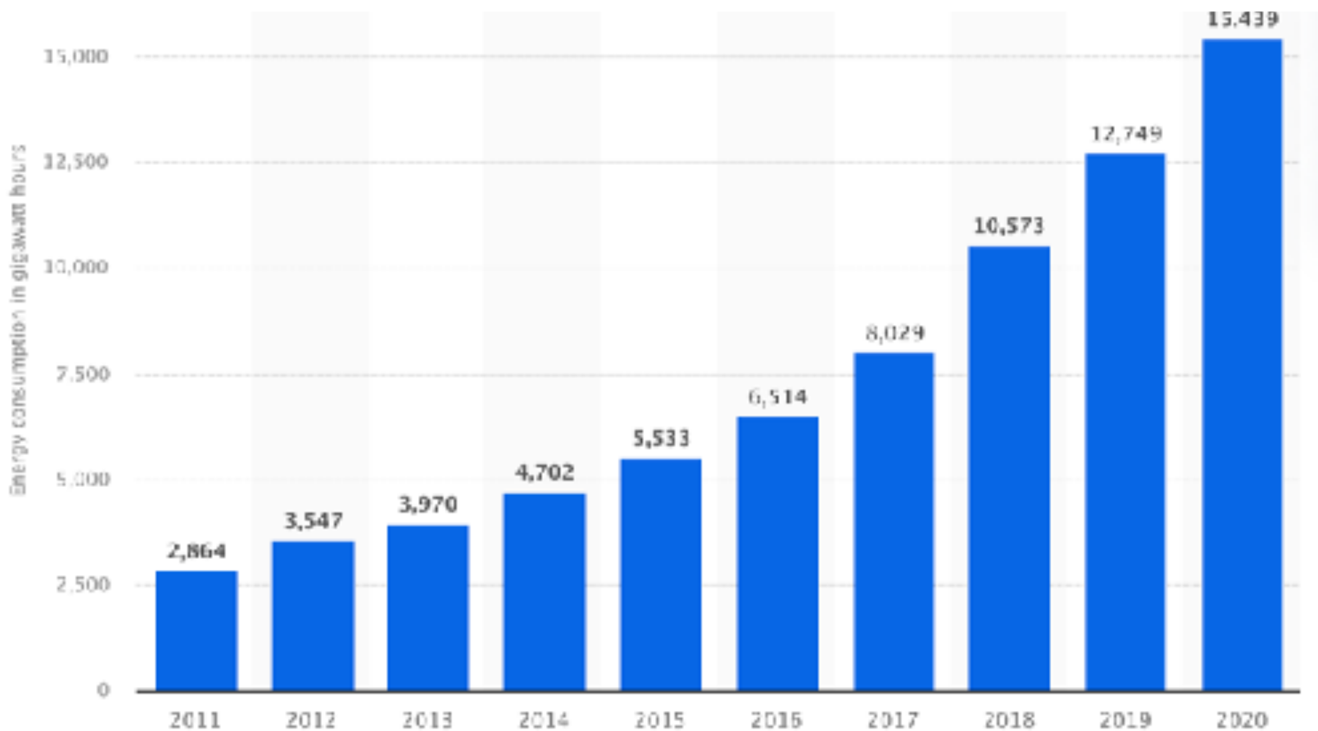
- Google has geographically distributed
  - data centers consisting of thousands of servers each
  - 2009 guess: about 1million servers total - now ??

(Google data center, Oregon US, (C) New York Times)

https://en.wikipedia.org/wiki/List_of_countries_by_electricity_consumption

POWER CONSUMPTION!

- Server computer:
  - about 200-400Watts
- Electricity consumption of Google in 2020



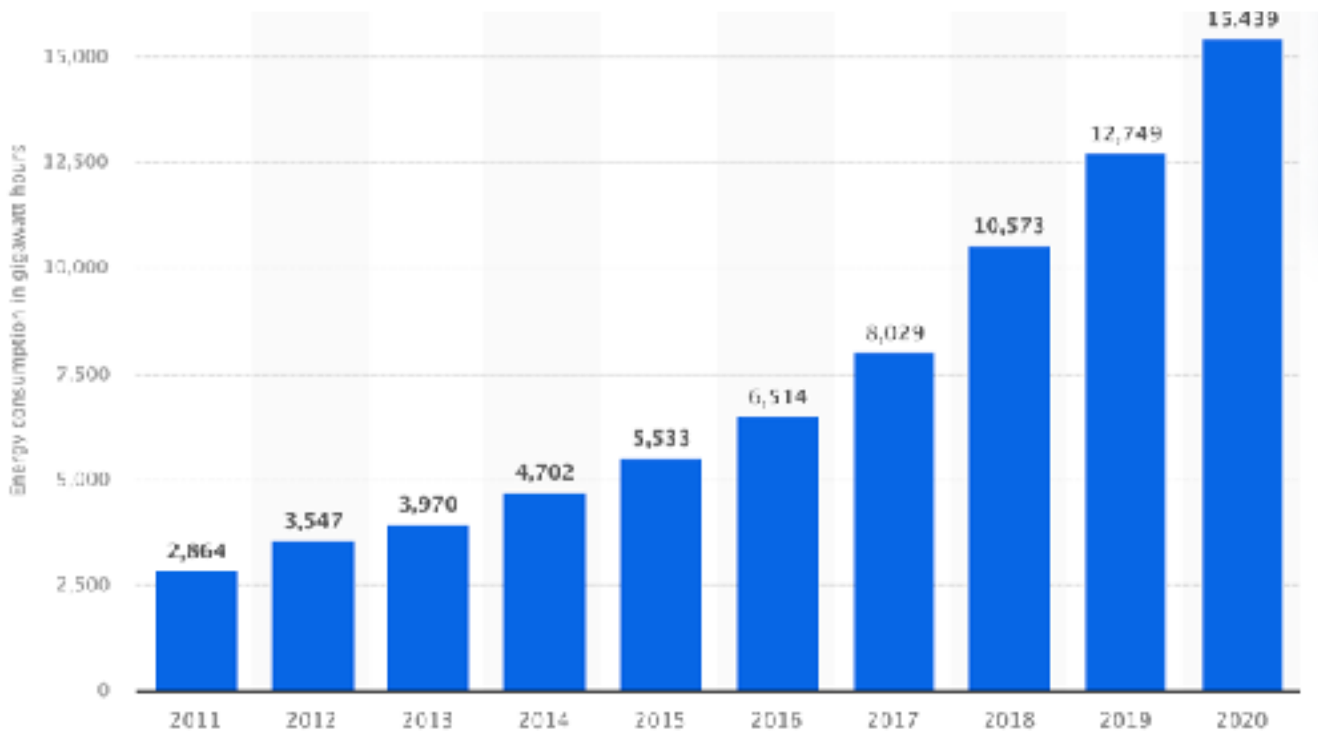
http://www.nytimes.com/2011/09/09/technology/google-details-and-defends-its-use-of-electricity.html
https://www.statista.com/statistics/788540/energy-consumption-of-google/

https://en.wikipedia.org/wiki/List_of_countries_by_electricity_consumption

POWER CONSUMPTION!

2023: Google and Microsoft each consumed more than 24 TWh

- Server computer:
  - about 200-400Watts
- Electricity consumption of Google in 2020



http://www.nytimes.com/2011/09/09/technology/google-details-and-defends-its-use-of-electricity.html

https://www.statista.com/statistics/788540/energy-consumption-of-google/

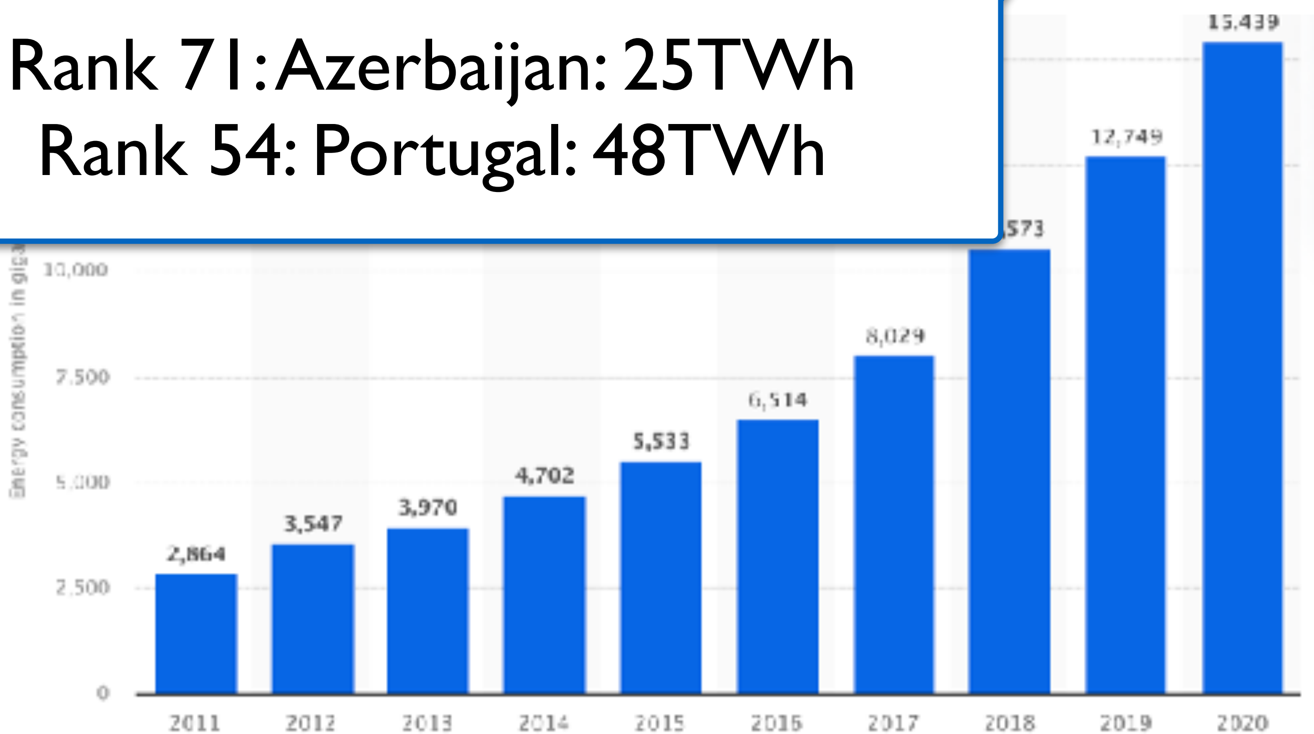
https://en.wikipedia.org/wiki/List_of_countries_by_electricity_consumption

POWER CONSUMPTION!

- Server computer:
  - about 200-400Watts
- Electricity consumption of Google in 2020

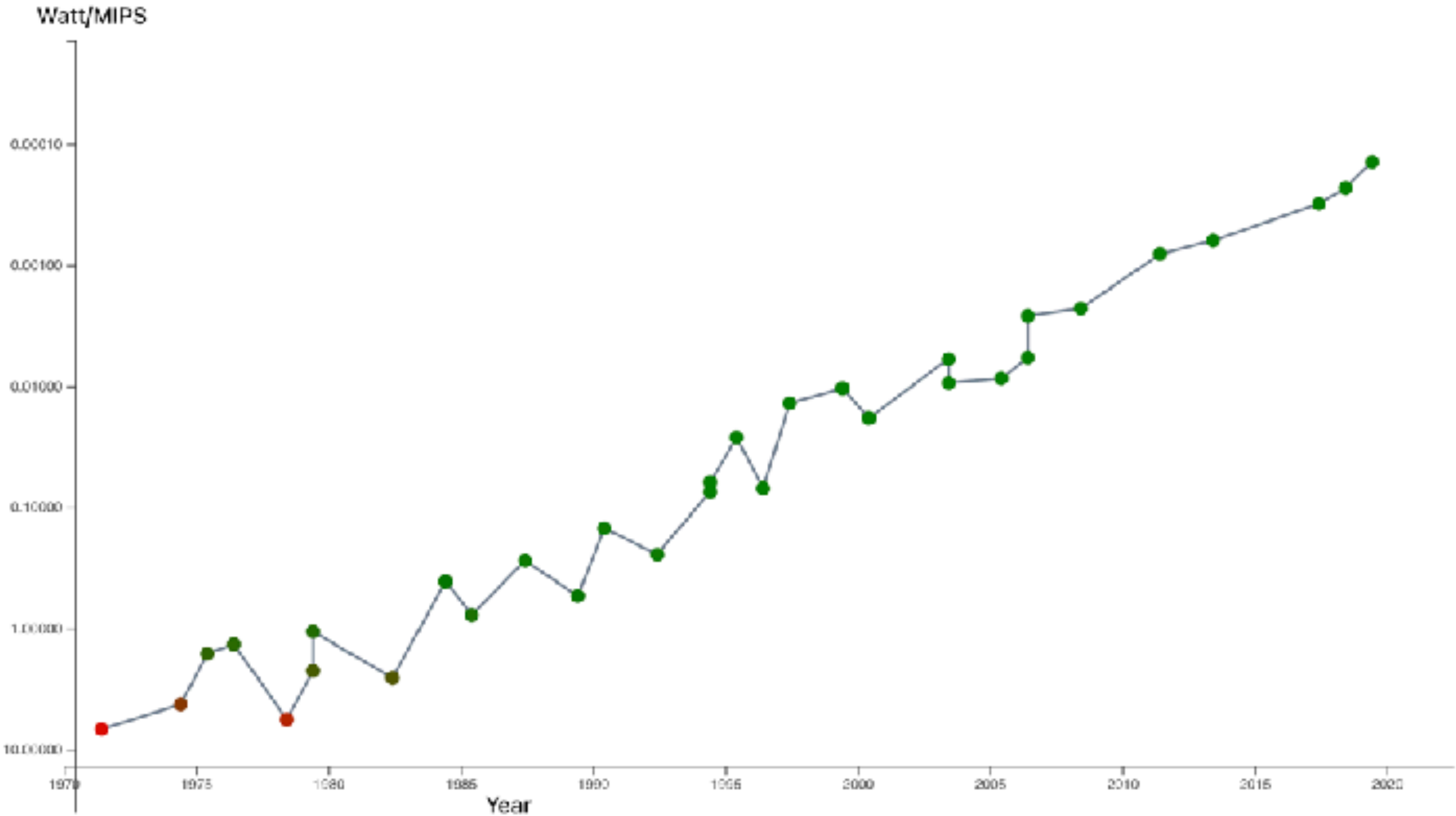
2023: Google and Microsoft each consumed more than 24 TWh

Rank 71: Azerbaijan: 25TWh
Rank 54: Portugal: 48TWh

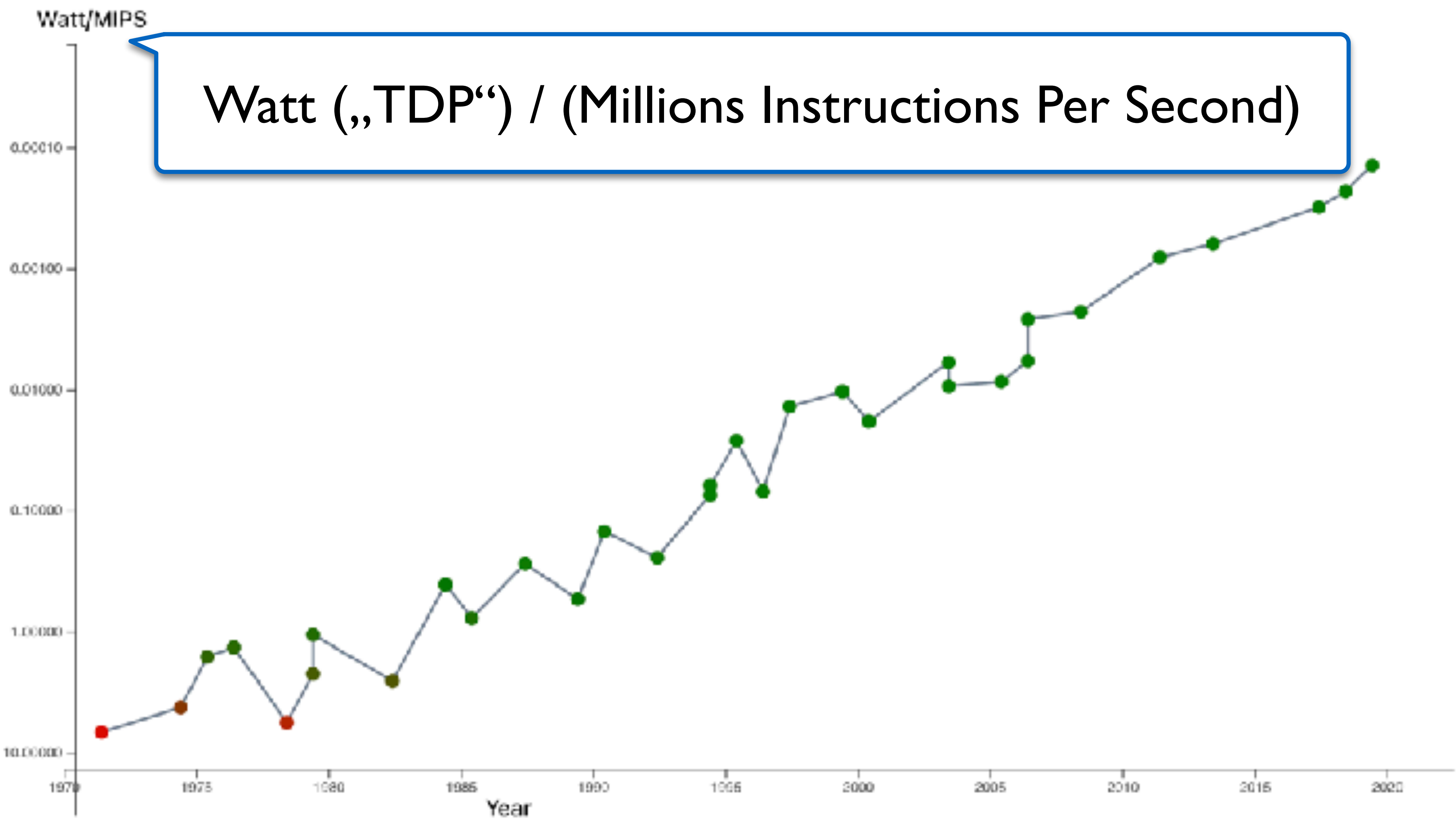


http://www.nytimes.com/2011/09/09/technology/google-details-and-defends-its-use-of-electricity.html
https://www.statista.com/statistics/788540/energy-consumption-of-google/

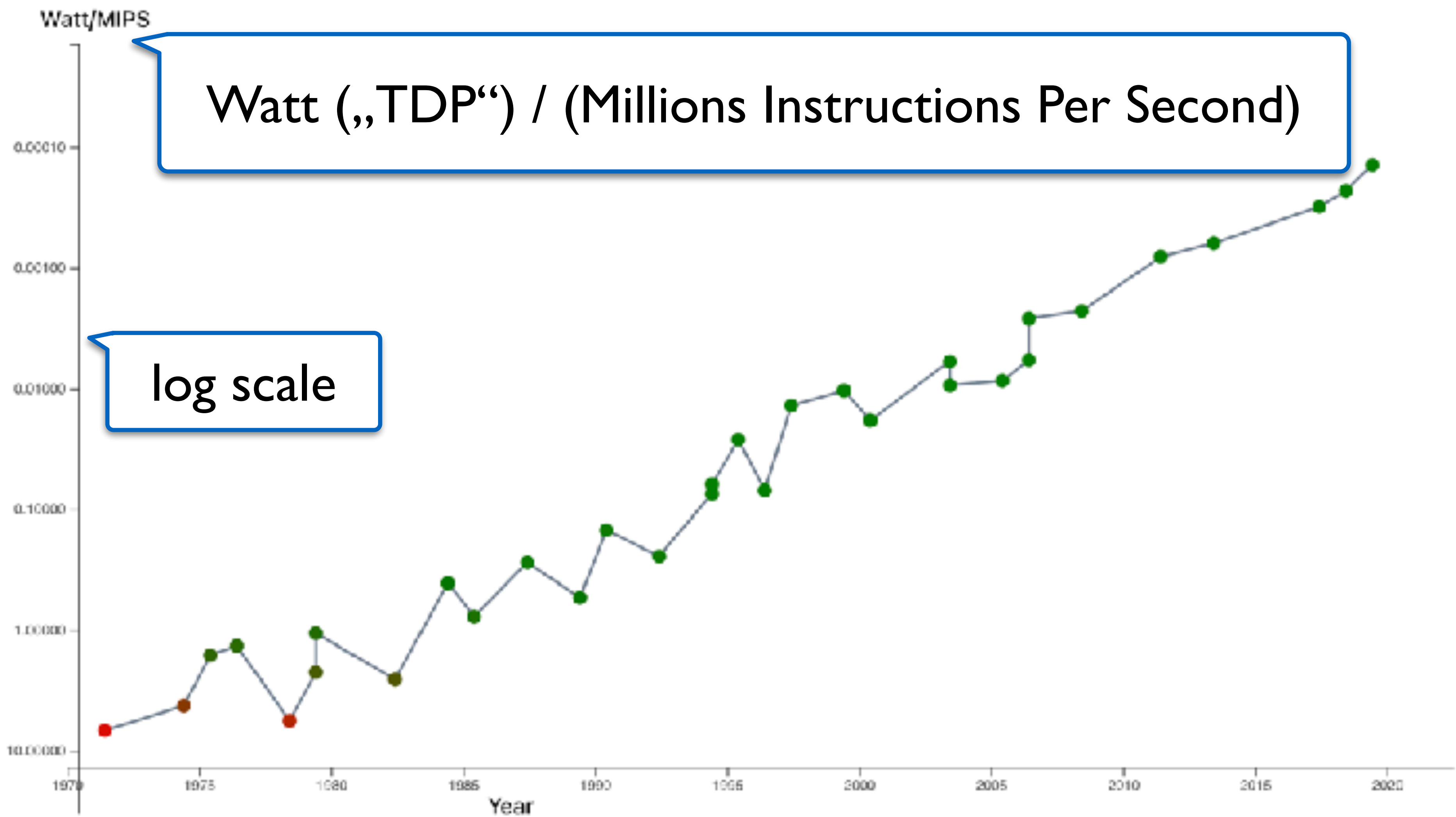
Energy Efficiency Increases



Energy Efficiency Increases

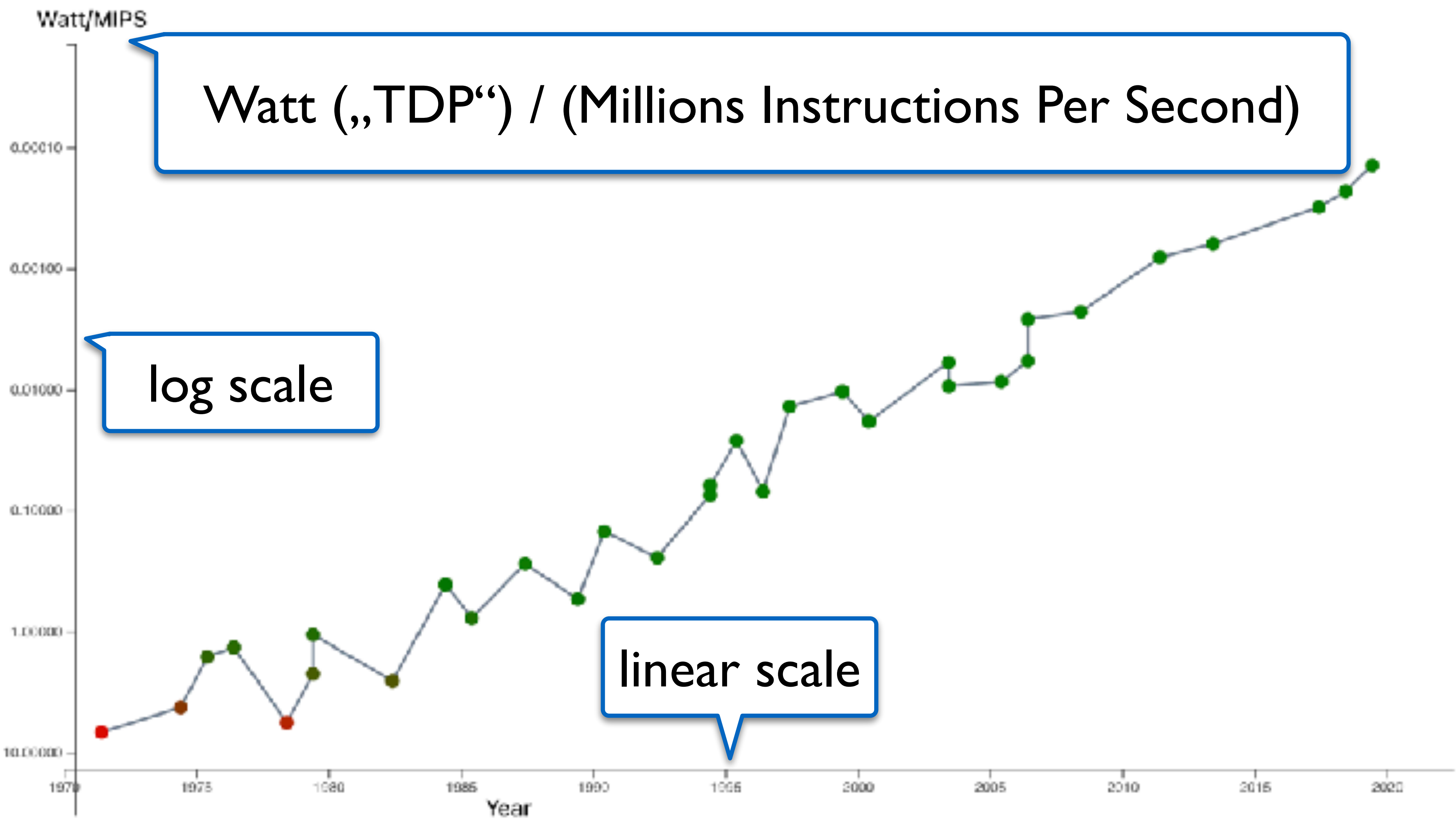


Energy Efficiency Increases

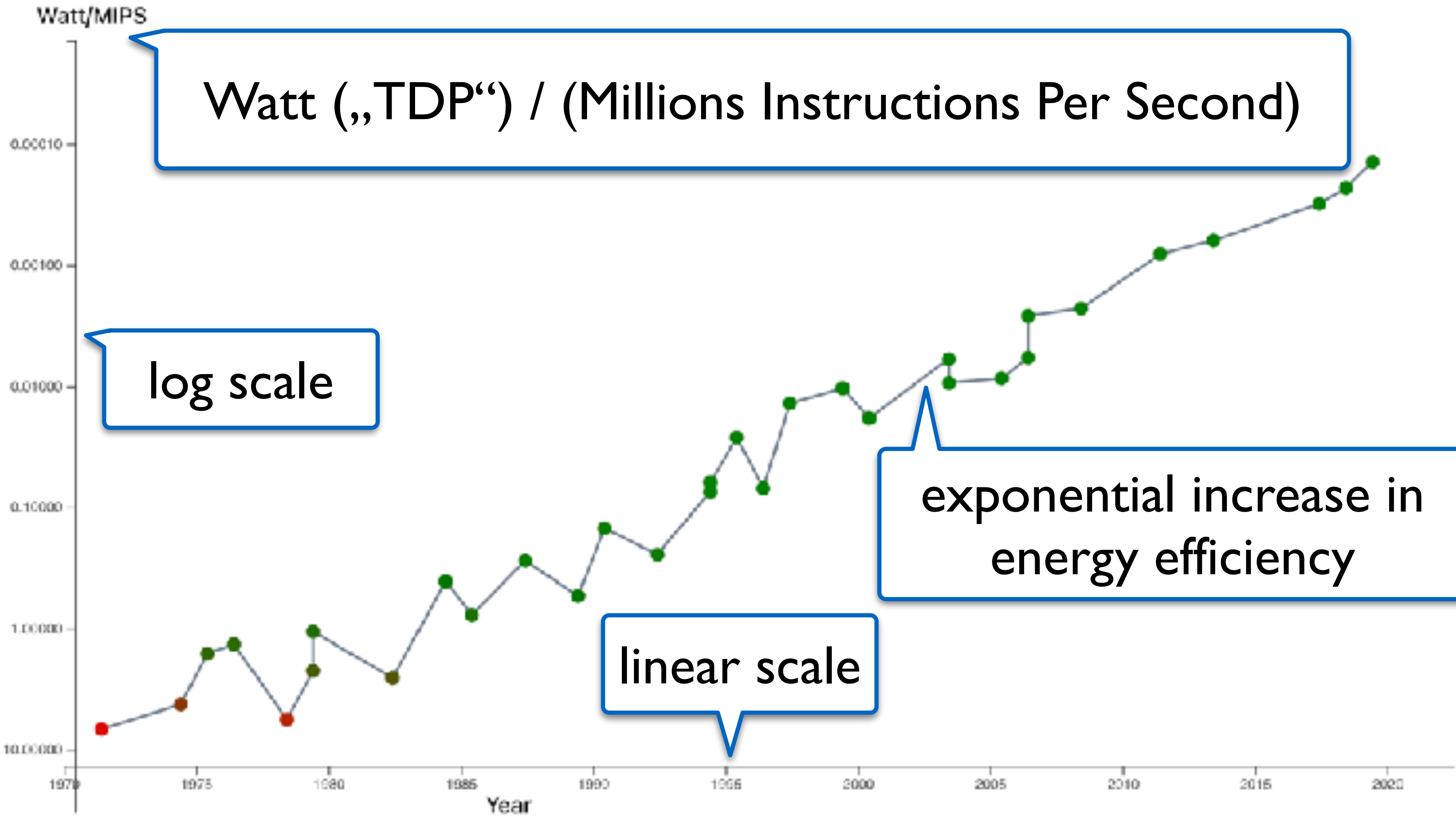


https://urbanspr1nter.github.io/dataviz/

Energy Efficiency Increases



Energy Efficiency Increases



Note: Exponential Increase in Computing Power

- over last decades -

Note: Exponential Increase in Computing Power

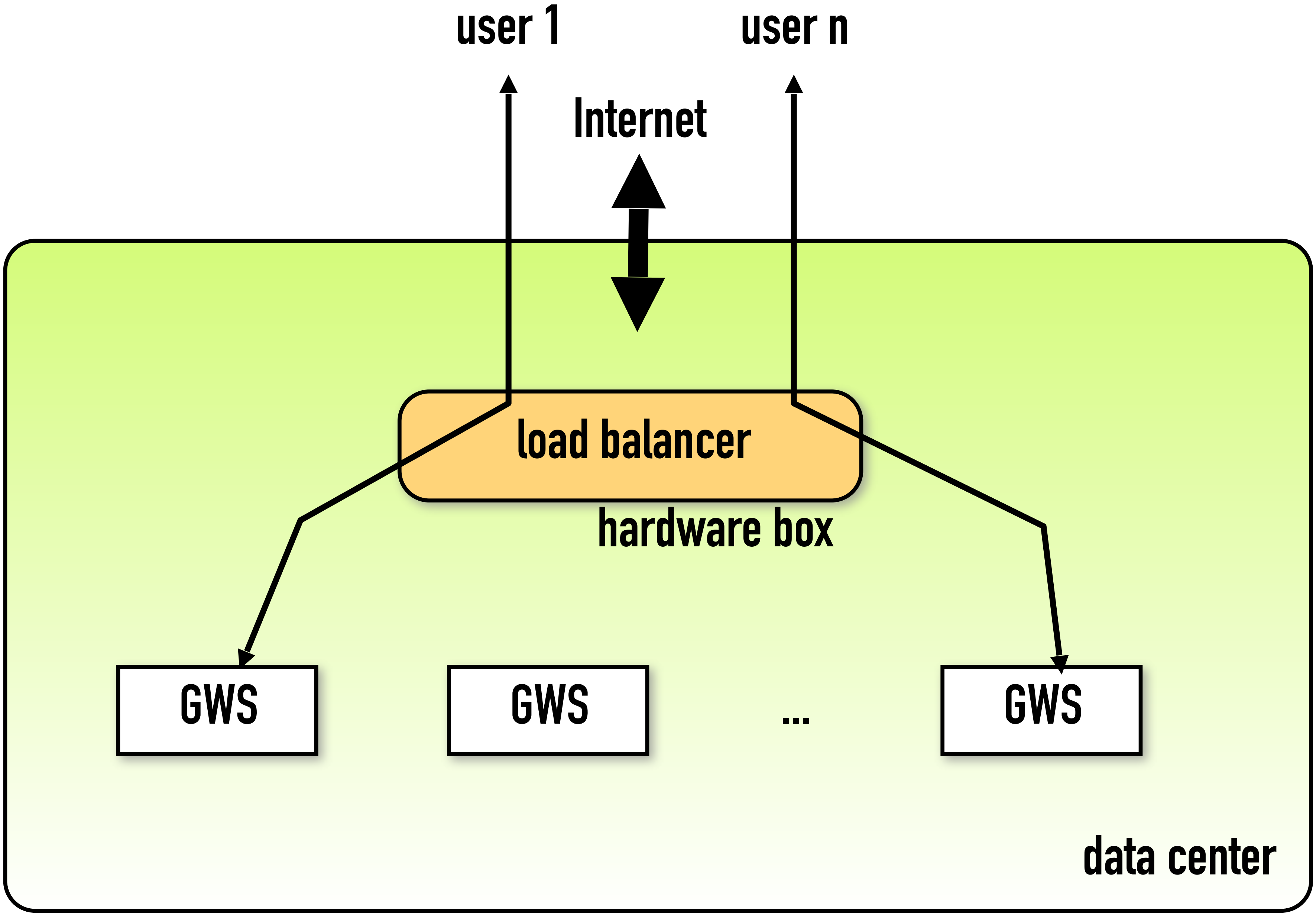
- over last decades -

AI workloads use lots of computing power…

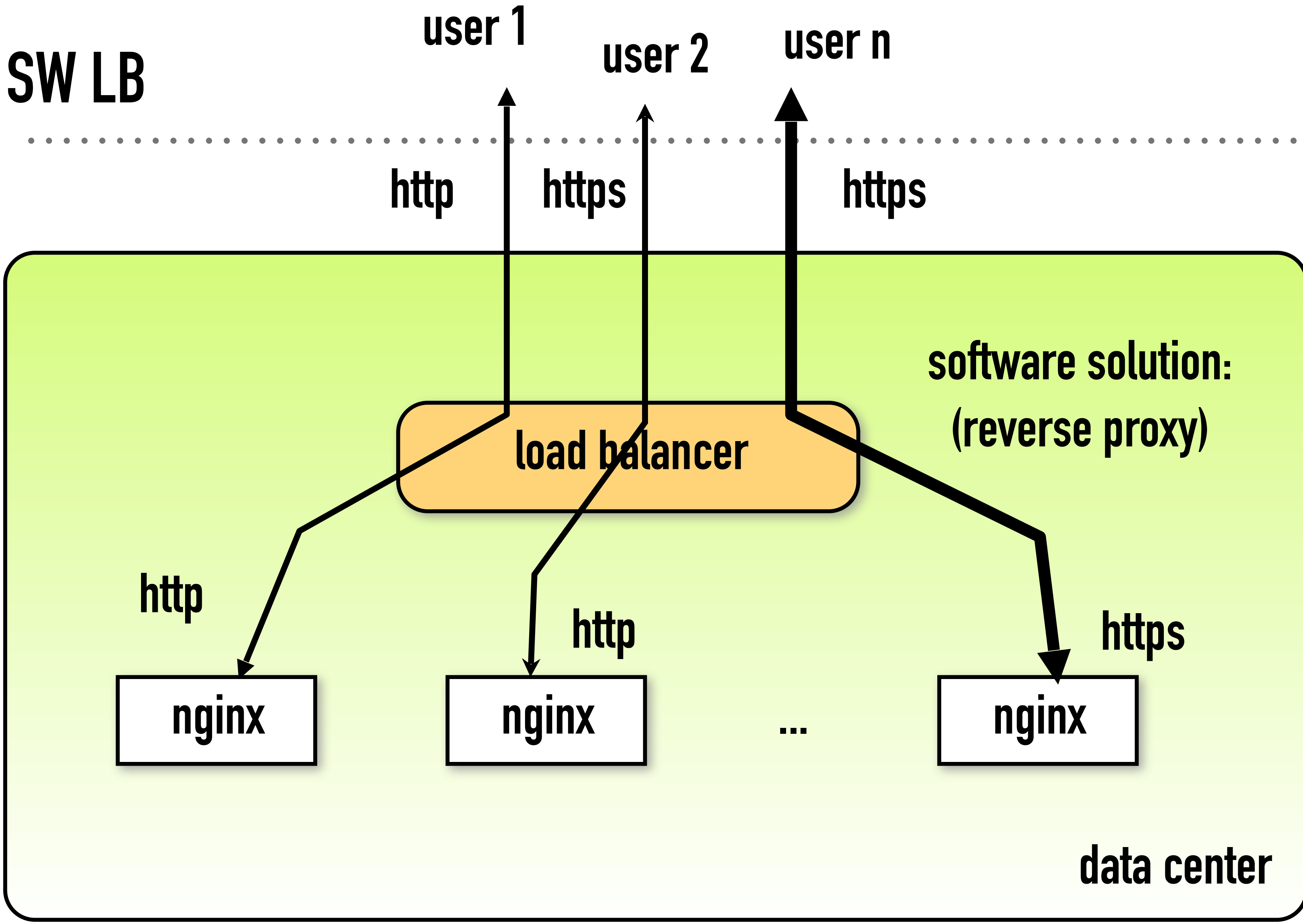
QUERY (3)

Browser (Chrome, Firefox, …) sends http request to IP address

- **Hardware load balancer** distributes requests amongst **Google Web Servers** (GWS)
- GWS coordinates execution of a request


user 1
user n
Internet
load balancer
hardware box
GWS
GWS
...
GWS
data center

SW LB



user 1
user 2
user n
http
https
https
software solution:
(reverse proxy)
load balancer
http
http
https
nginx
nginx
...
nginx
data center

SEQUENTIAL QUERY PROCESSING

- GWS queries index server
- Index server contains:
  - inverted index: (word , list(URL,score))
- Index server computes hit list for
  - each query word, and
  - computes intersection of individual hit lists
- Computes score of the documents
  - in the intersection

INDEX SERVER: HIT LISTS

systems

URLx, 1000

...

URLy, 800

...

URLz, 600

engineering

URLy, 900

...

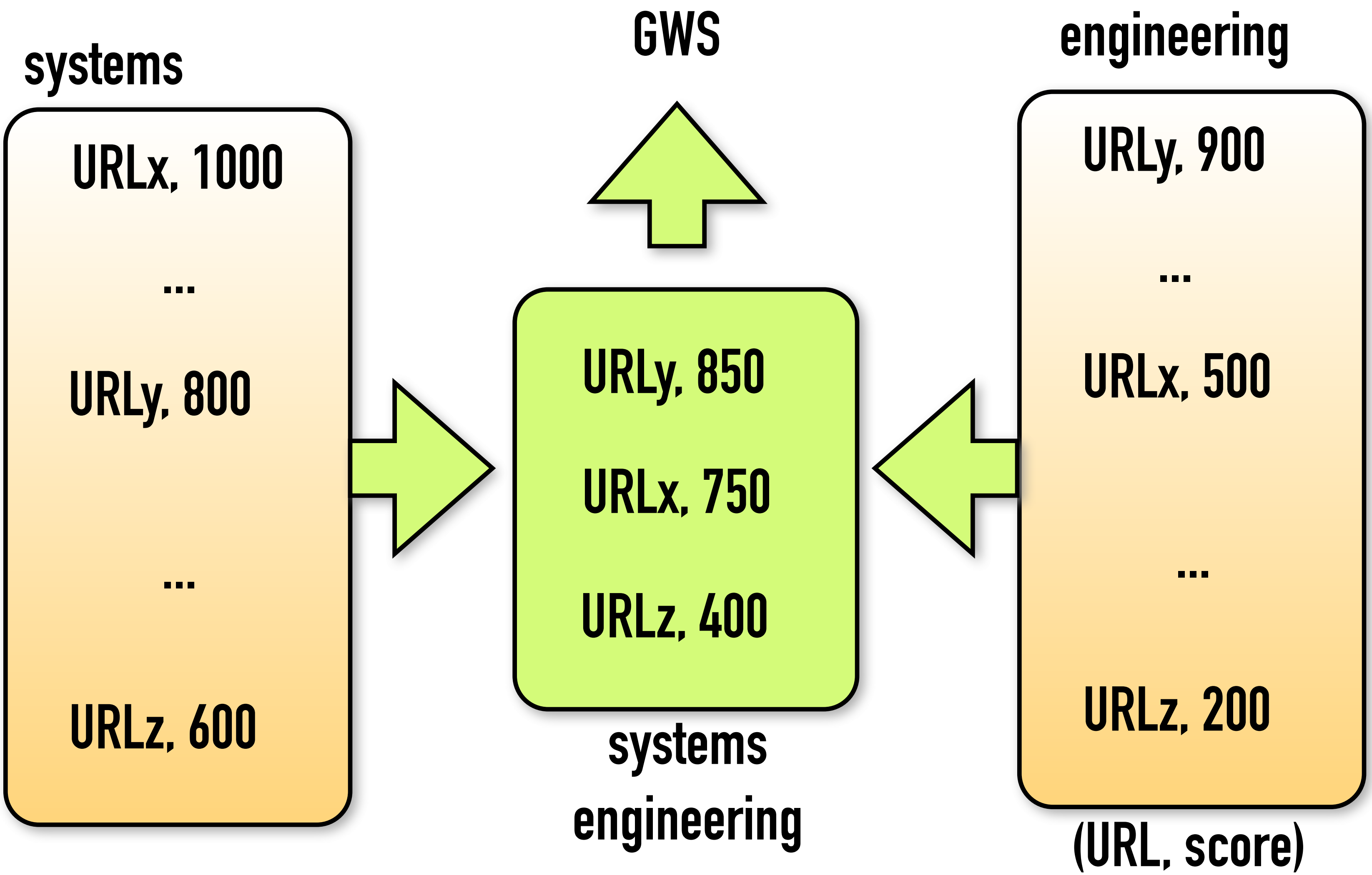
URLx, 500

...

URLz, 200

(URL, score)

COMBINE HIT LISTS



PROBLEMS?

- **Billions of web pages!**
  - index does not fit into single machine

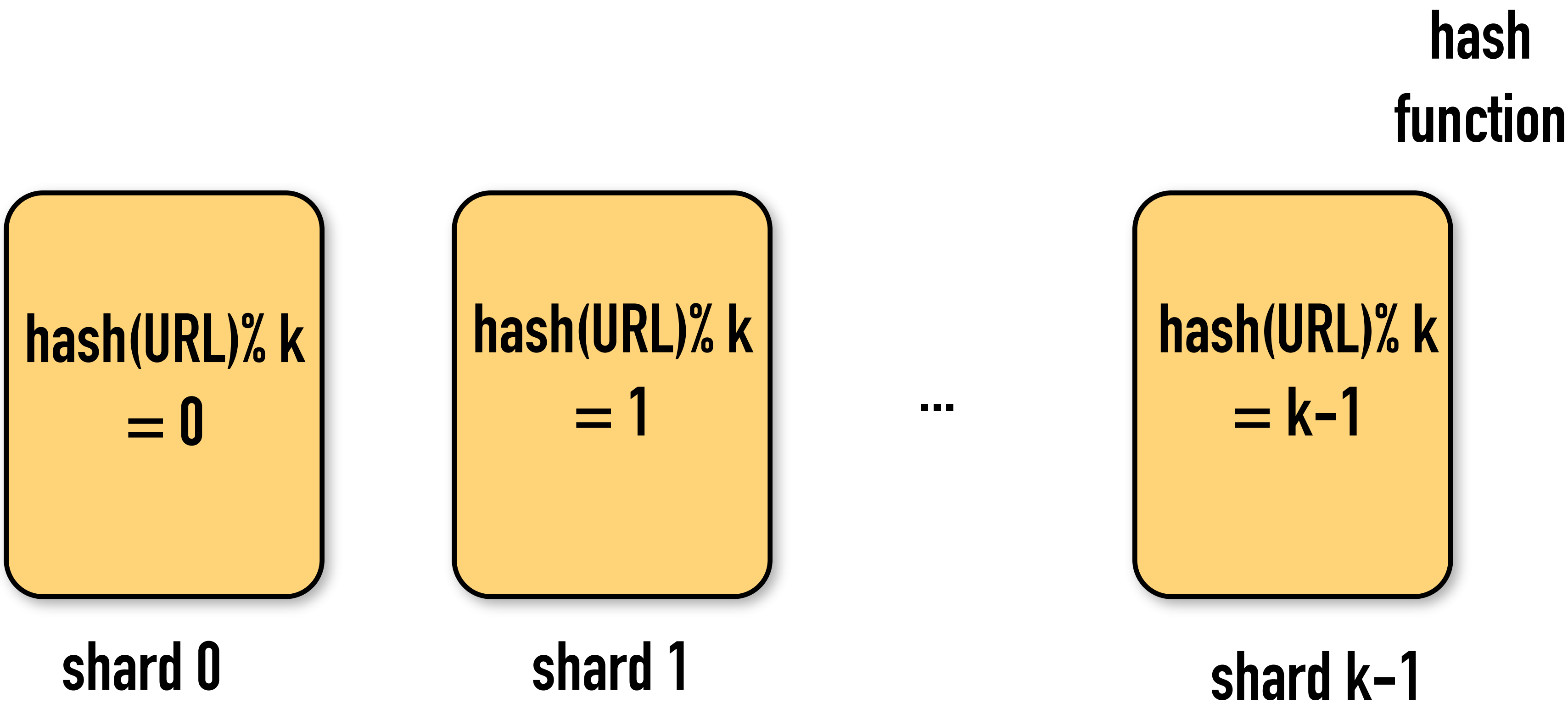
- **Very high # of requests per second**
  - single index server cannot serve all requests
    - even if index would fit on single machine

PARTITION!

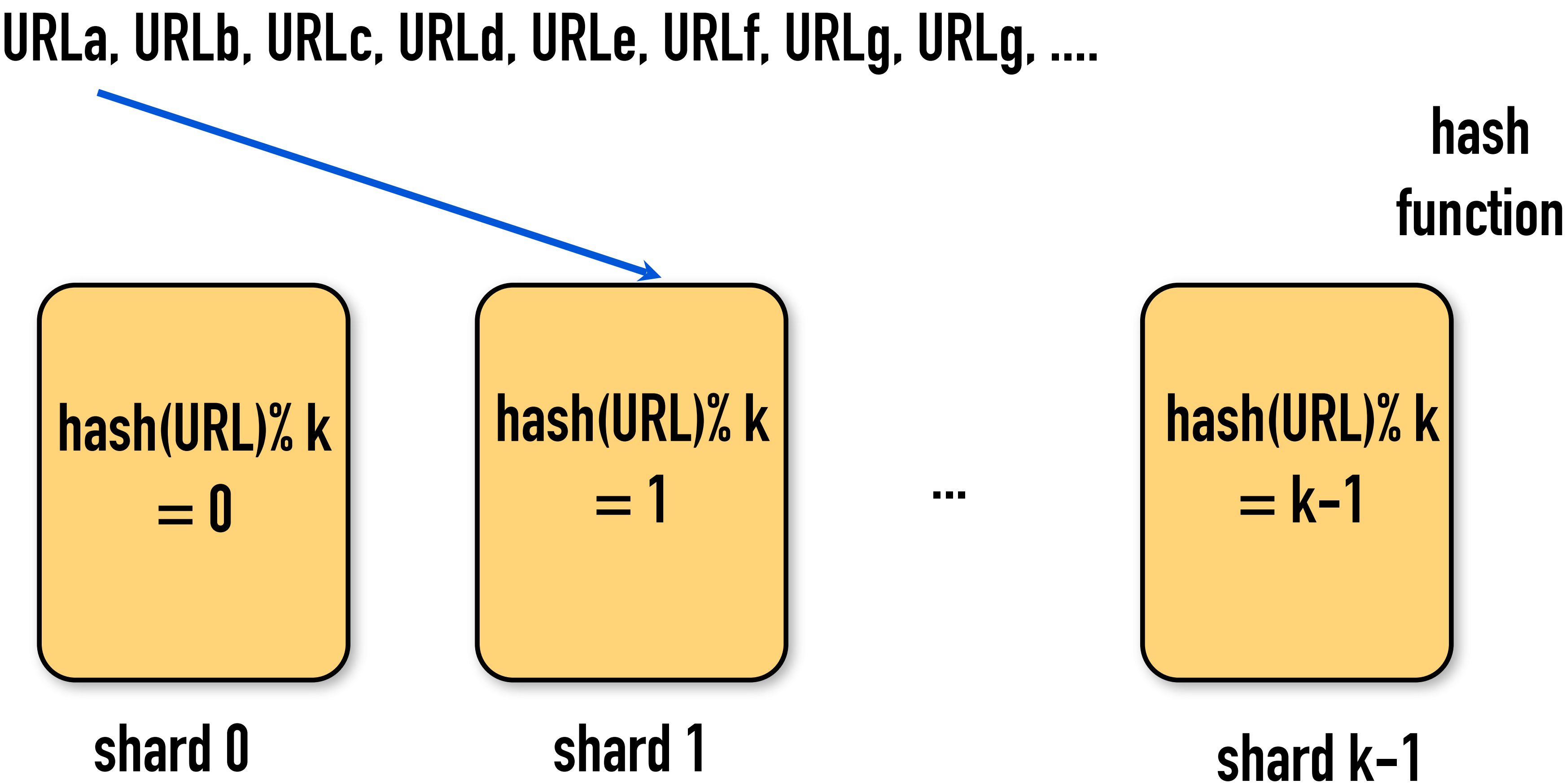
- **Index is randomly partitioned into**
  - **Index shards**
    - each shard contains index for a disjoint **subset of URLs**
  - e.g., use hash function to partition URLs
- **Pool of servers serves each shard**
  - requests are broadcast to all shards
  - load balancer assigns request to one or more servers in a shard

IDEA: PARTITION URLS

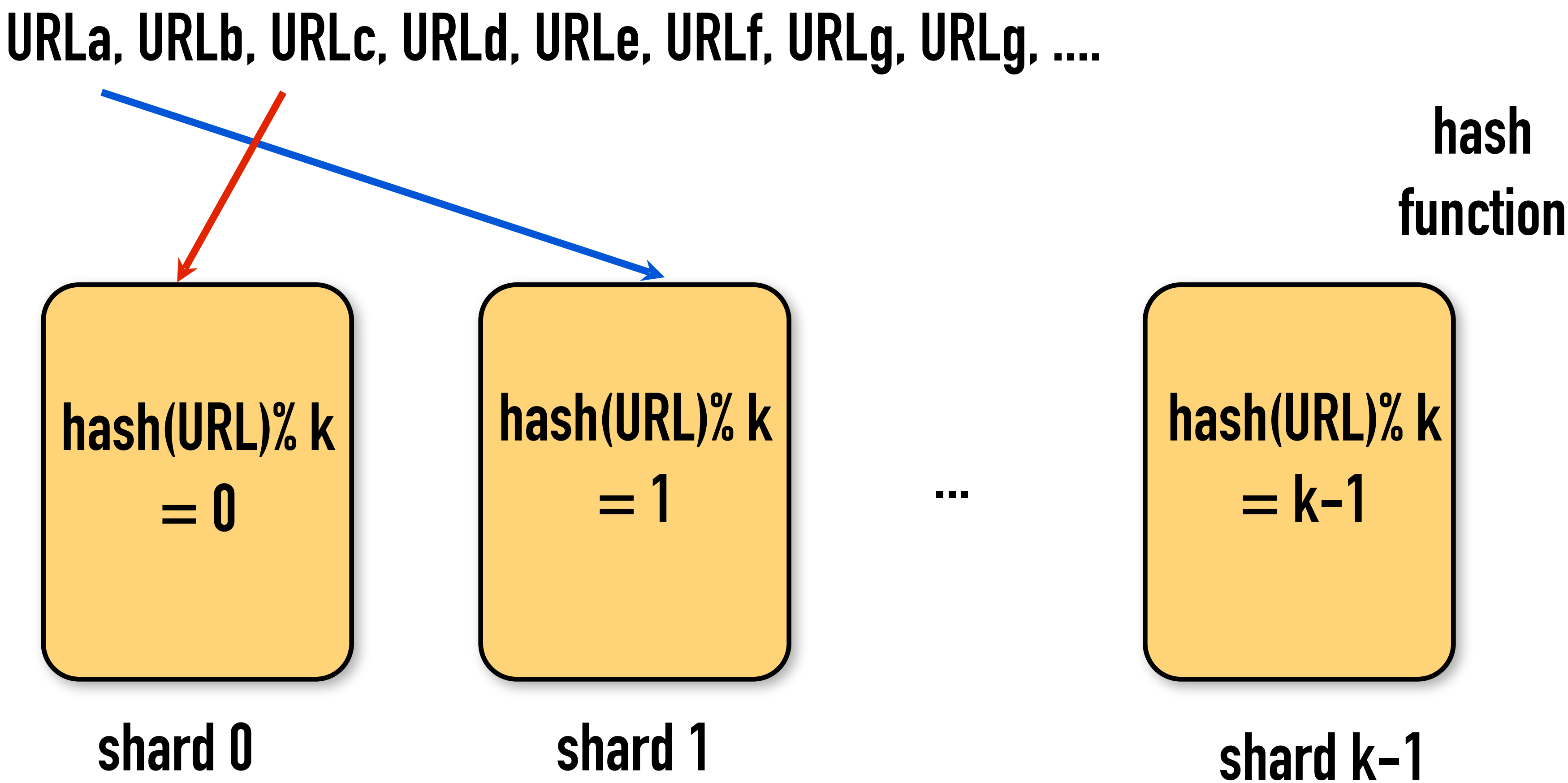
URLa, URLb, URLc, URLd, URLe, URLf, URLg, URLg, ...



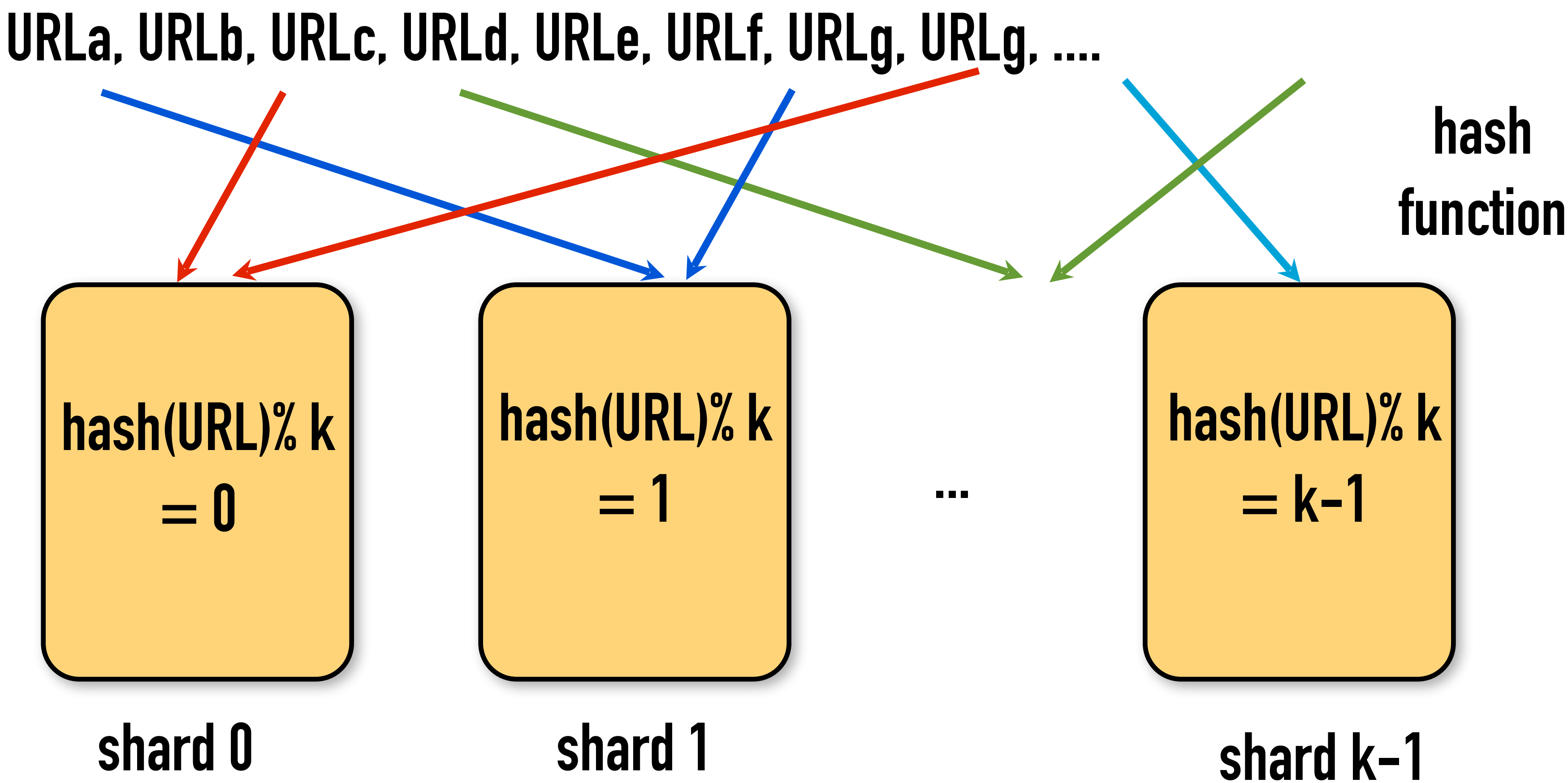
IDEA: PARTITION URLS

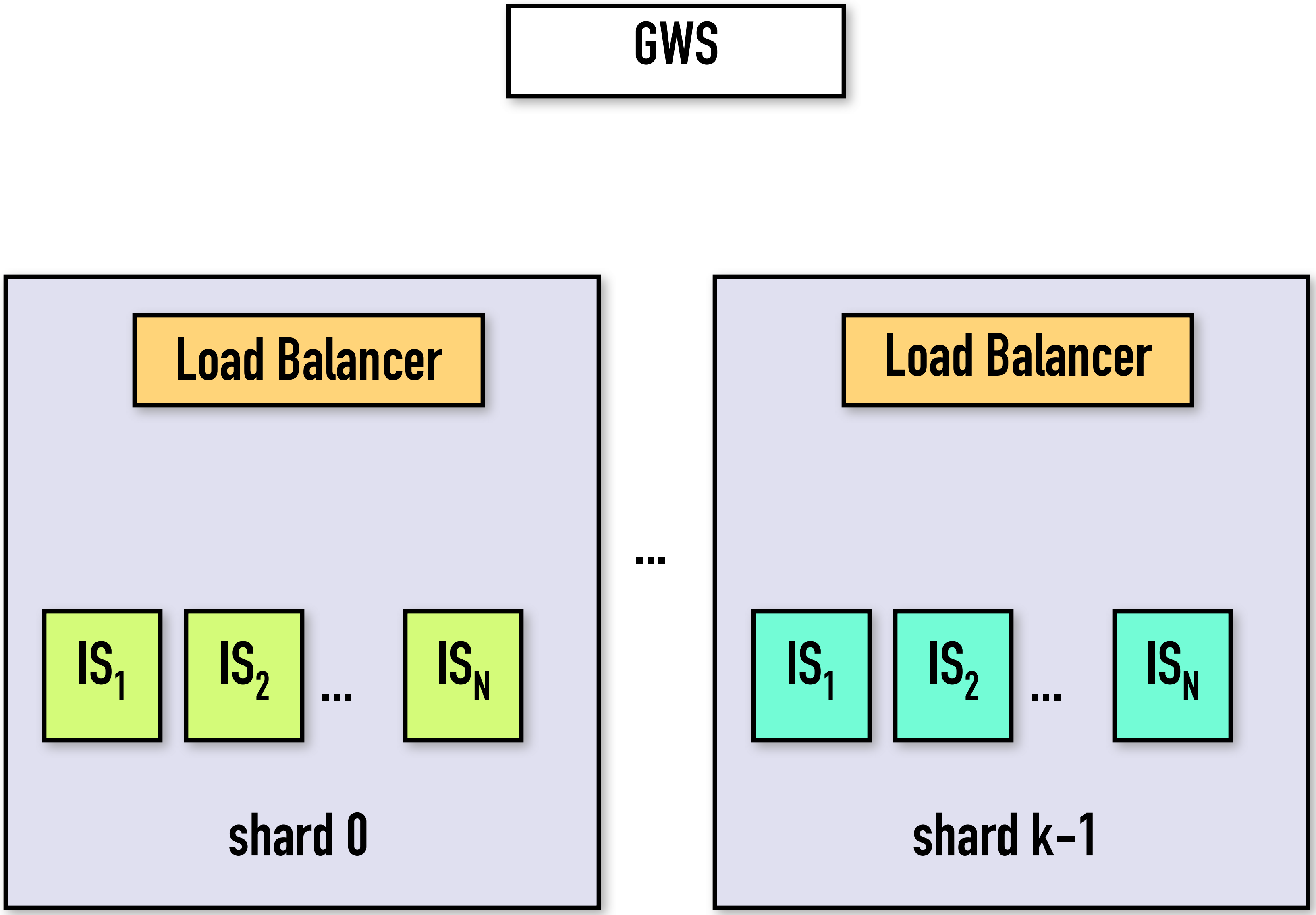


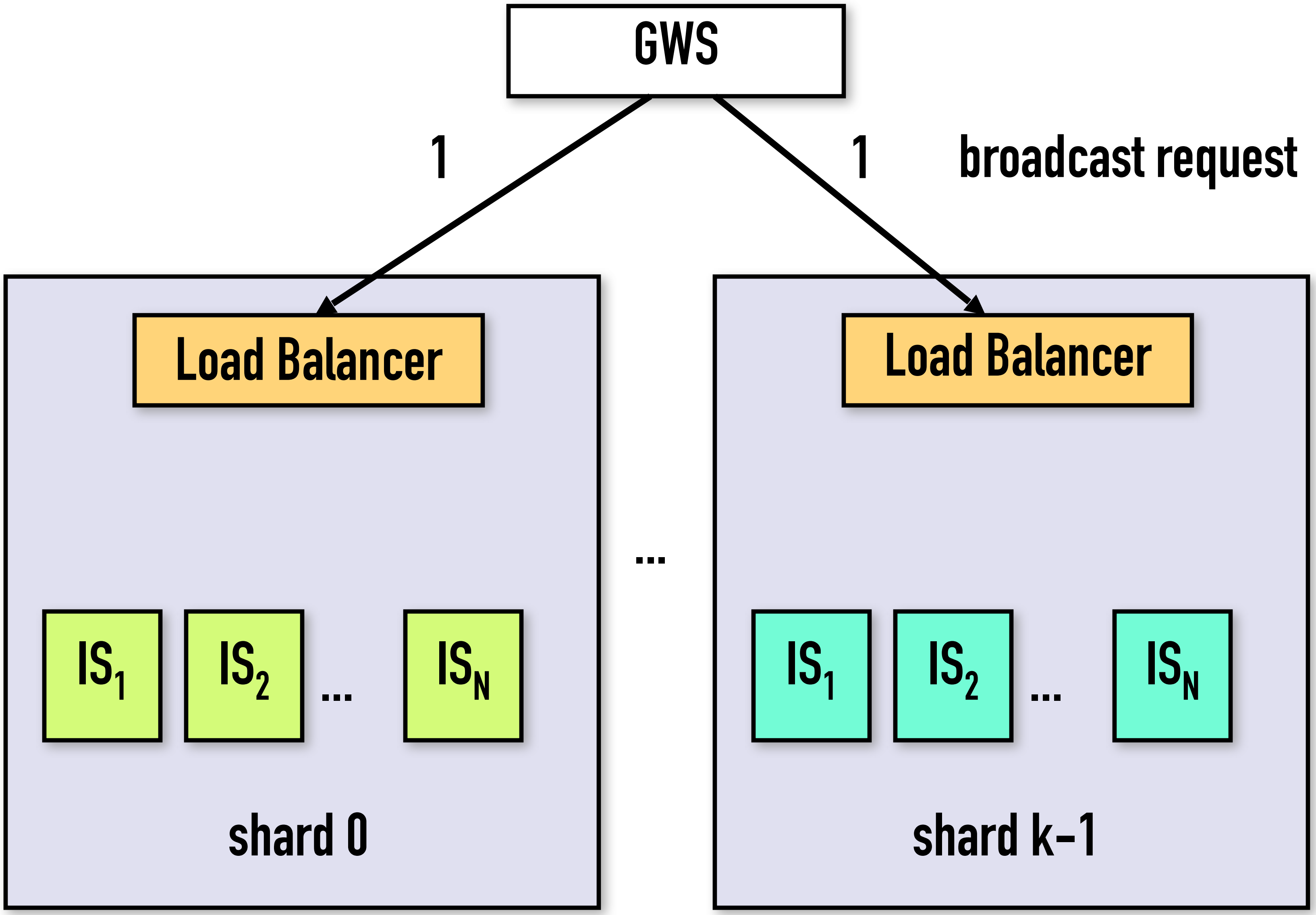
IDEA: PARTITION URLS



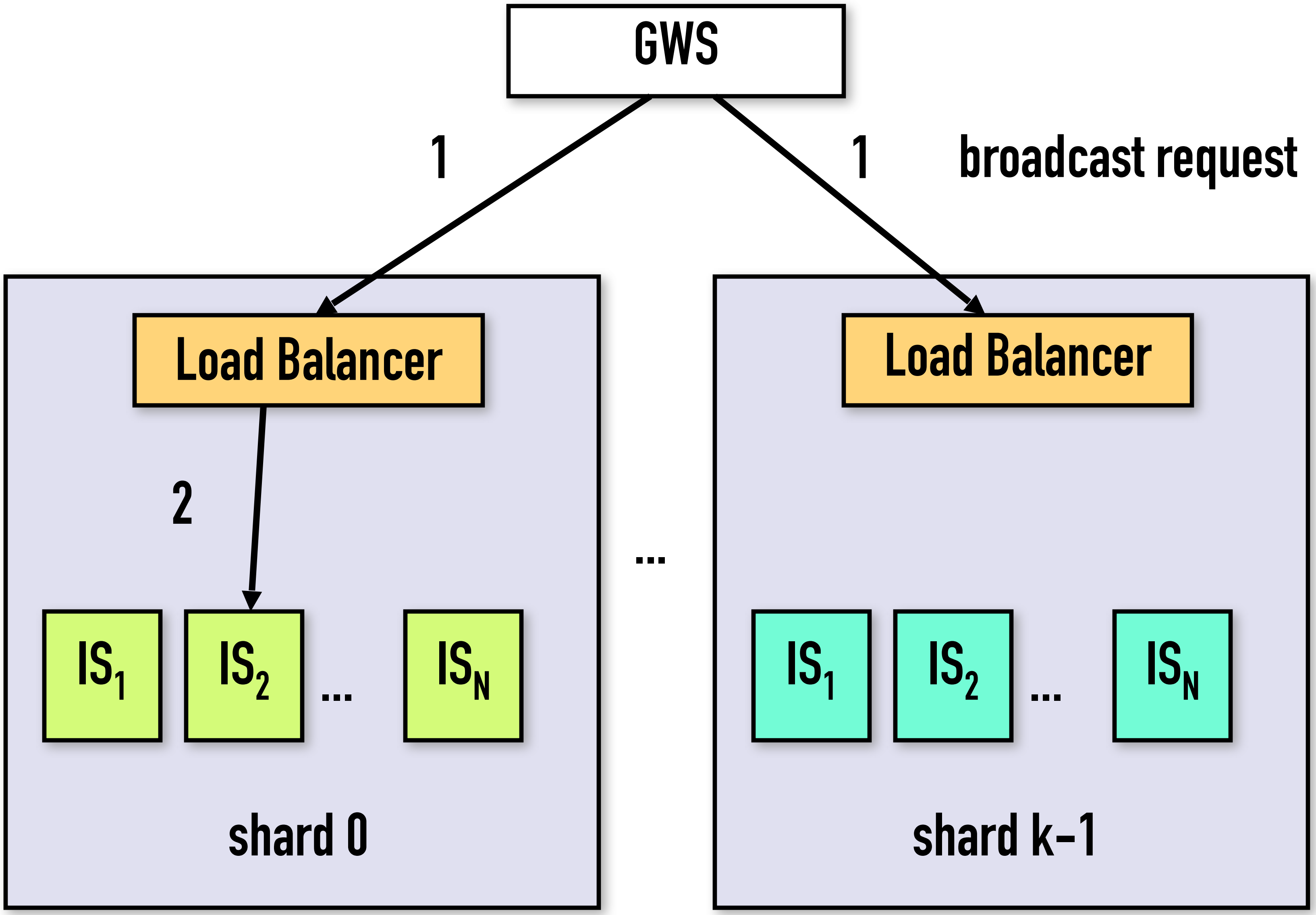
IDEA: PARTITION URLS

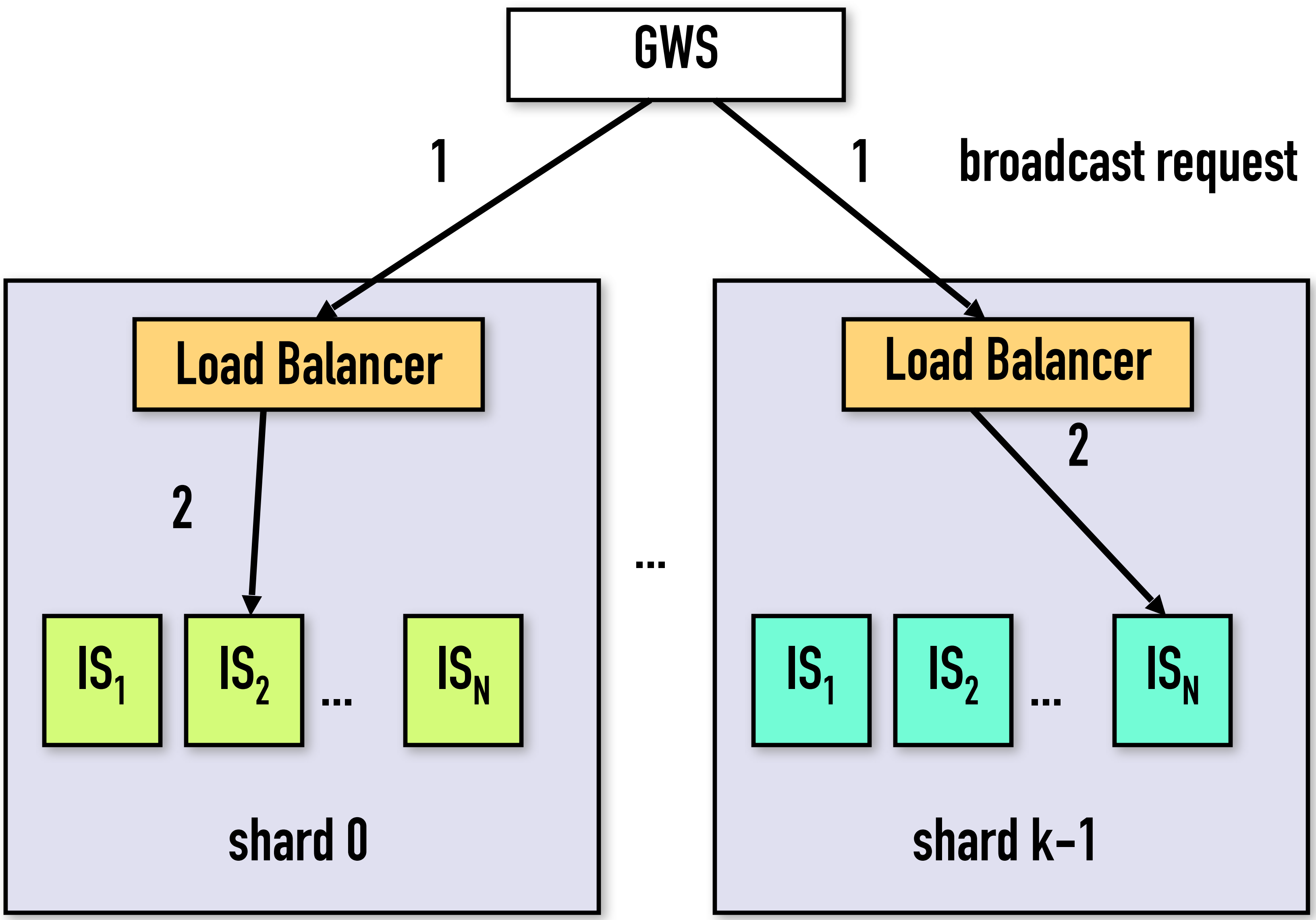


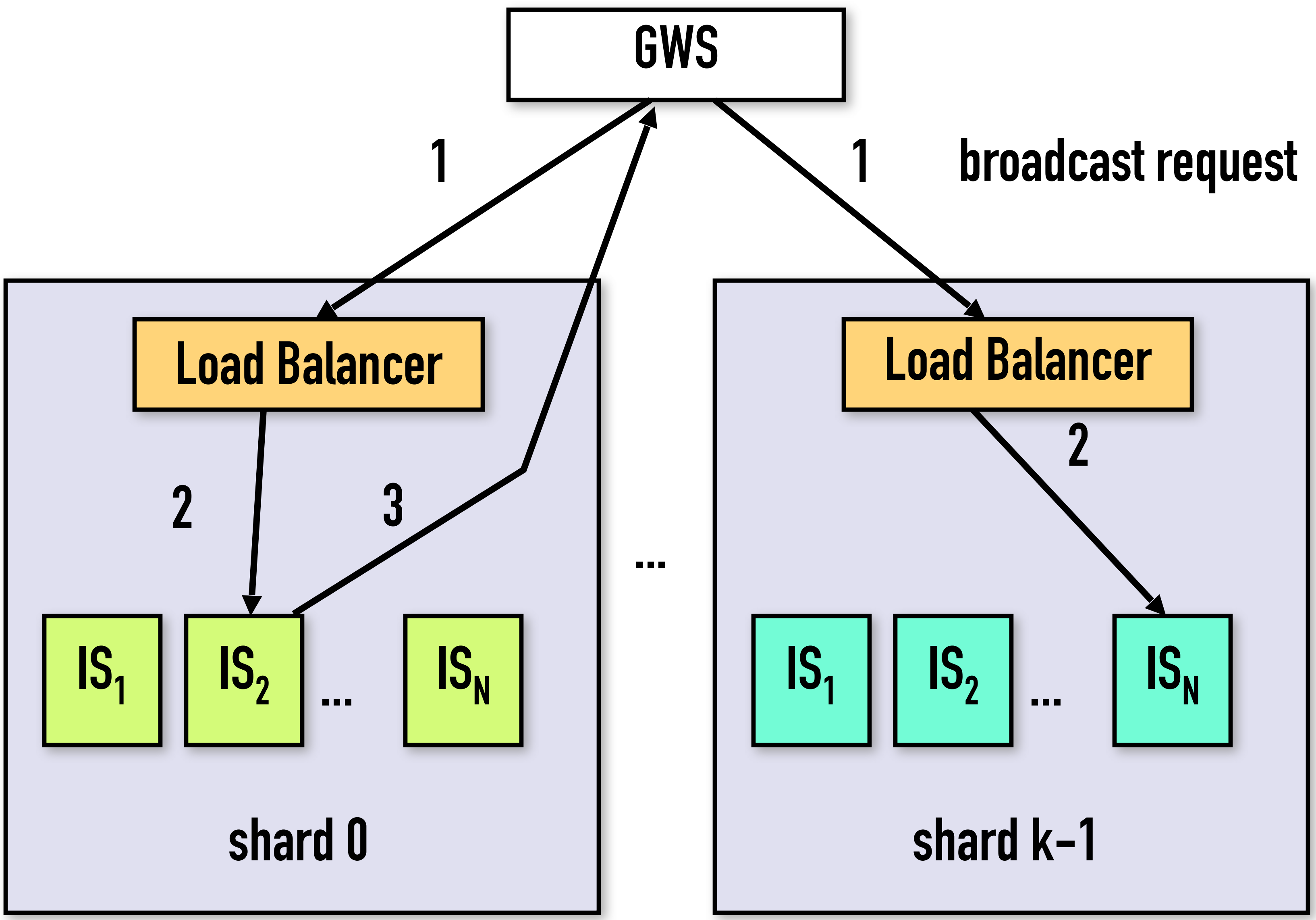

GWS
Load Balancer
IS_1
IS_2
...
IS_N
shard 0
...
Load Balancer
IS_1
IS_2
...
IS_N
shard k-1



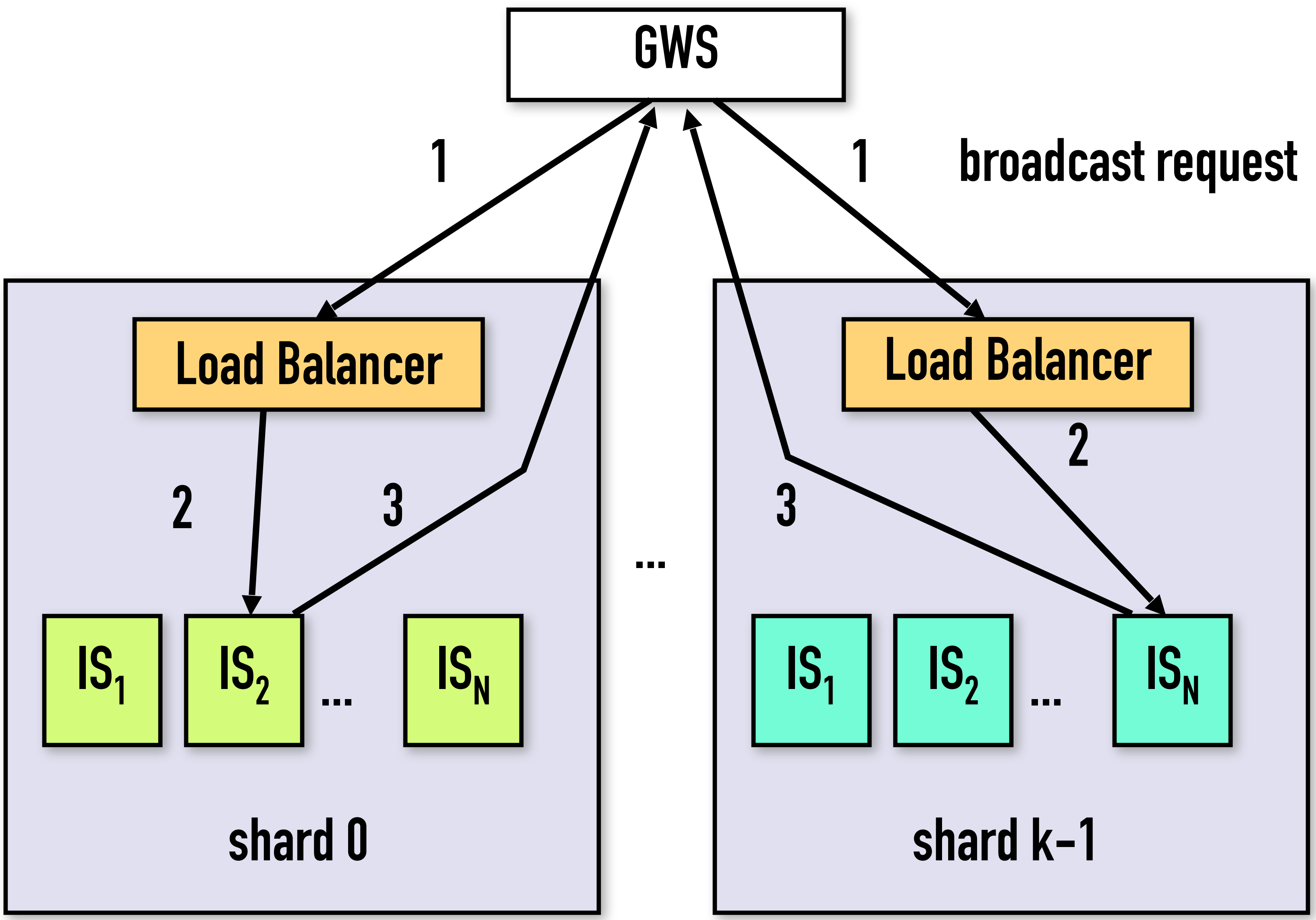
GWS
1
1
broadcast request
Load Balancer
Load Balancer
...
IS1
IS2
...
ISN
IS1
IS2
...
ISN
shard 0
shard k-1


GWS
1
1
broadcast request
Load Balancer
Load Balancer
2
IS1
IS2
...
ISN
...
IS1
IS2
...
ISN
shard 0
shard k-1

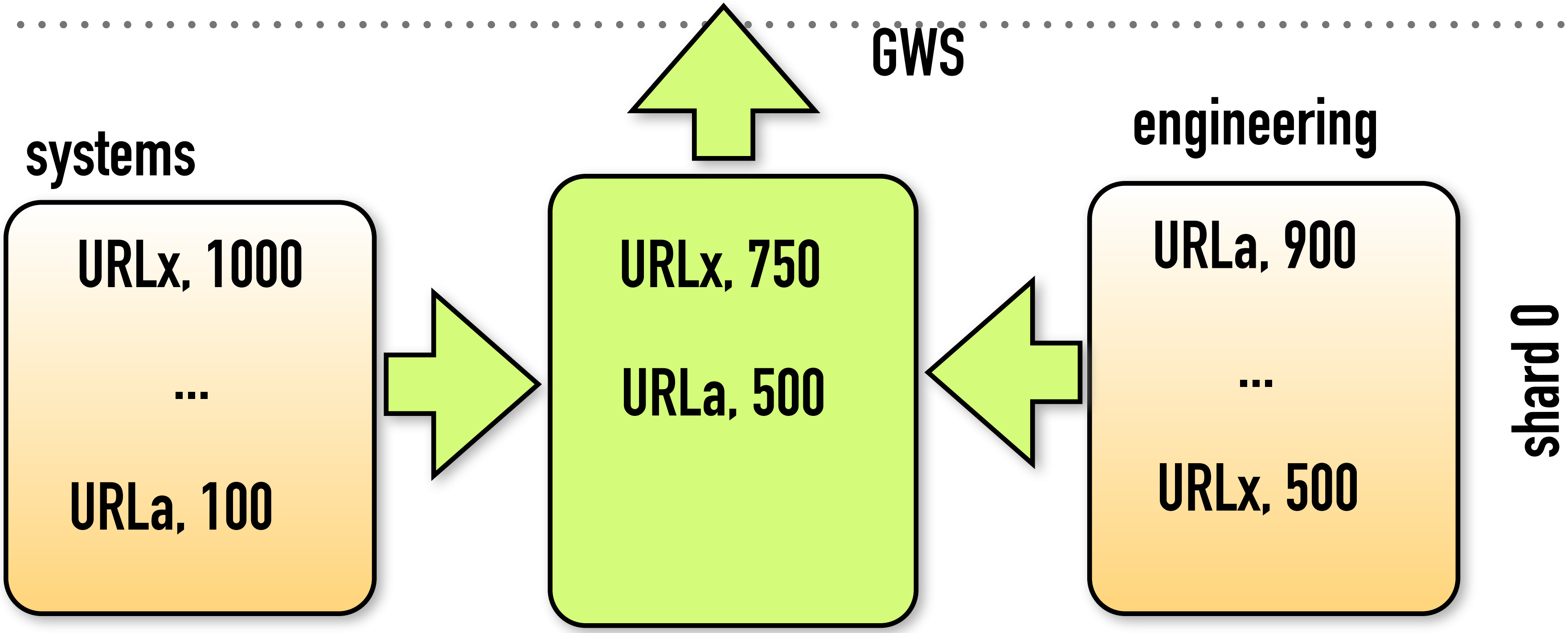

GWS
1
1
broadcast request
Load Balancer
2
IS1
IS2
...
ISN
shard 0
...
Load Balancer
2
IS1
IS2
...
ISN
shard k-1



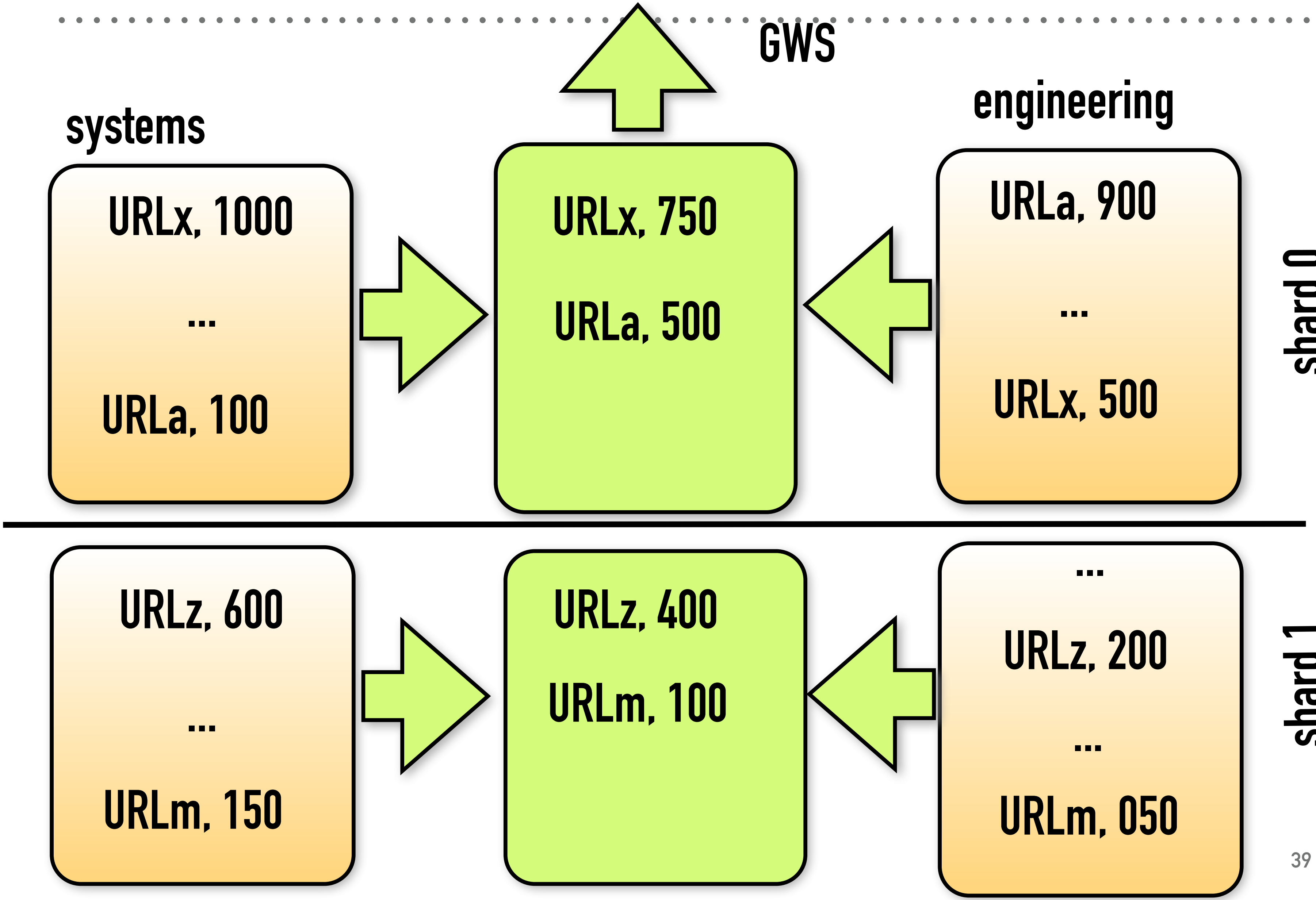
GWS
1
1
broadcast request
Load Balancer
Load Balancer
2
3
2
...
IS1
IS2
...
ISN
IS1
IS2
...
ISN
shard 0
shard k-1


GWS
1
1
broadcast request
Load Balancer
Load Balancer
2
3
3
2
...
IS$_1$
IS$_2$
...
IS$_N$
IS$_1$
IS$_2$
...
IS$_N$
shard 0
shard k-1

SHARDED COMPUTATION



SHARDED COMPUTATION



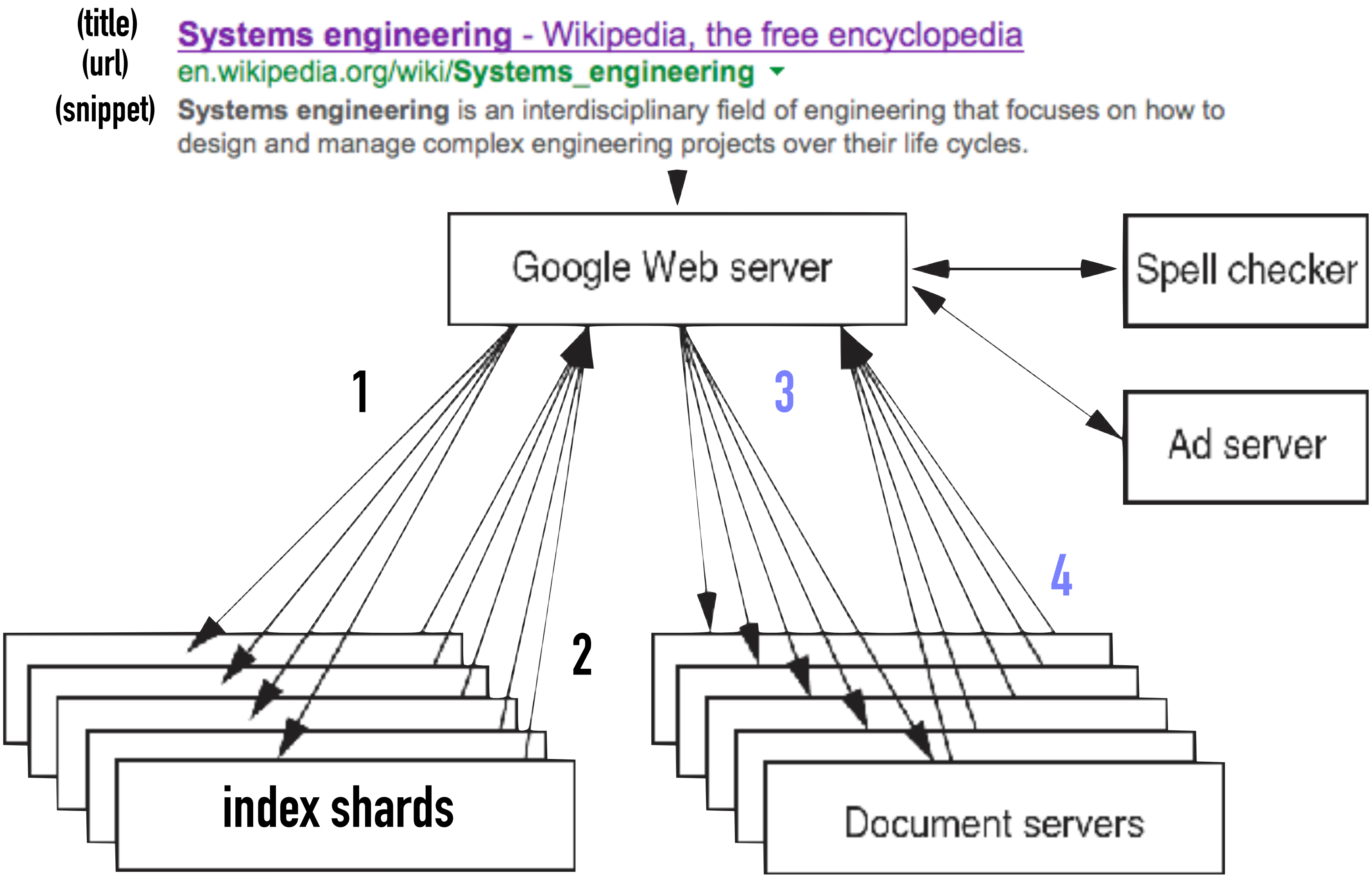
SCALABILITY

- Increasing number of requests / second
  - add more GWS & more replicas per shard
- „Imperfect“ hash function:
  - use different number of replicas per shard
- Increasing size of index
  - add more shards
- Reaching limit of a data center
  - add new data center using DNS / network load balancer

2ND PHASE: RESULT PROCESSING

- Phase 1:
  - ordered list of document ids from index shards
  - merge sort of the lists

- **Phase 2: Result processing**
  - retrieve all documents in list
  - compute title, url, text snippet



(title)
(url)
(snippet)
Systems engineering - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Systems_engineering
Systems engineering is an interdisciplinary field of engineering that focuses on how to design and manage complex engineering projects over their life cycles.
Google Web server
Spell checker
Ad server
1
2
3
4
index shards
Document servers

PARALLELIZE DOCUMENT RETRIEVAL

- Parallelize by:
  - randomly assign documents to shards
  - each shard is served by a pool of servers
  - request sent to a document server in a shard via a load balancer

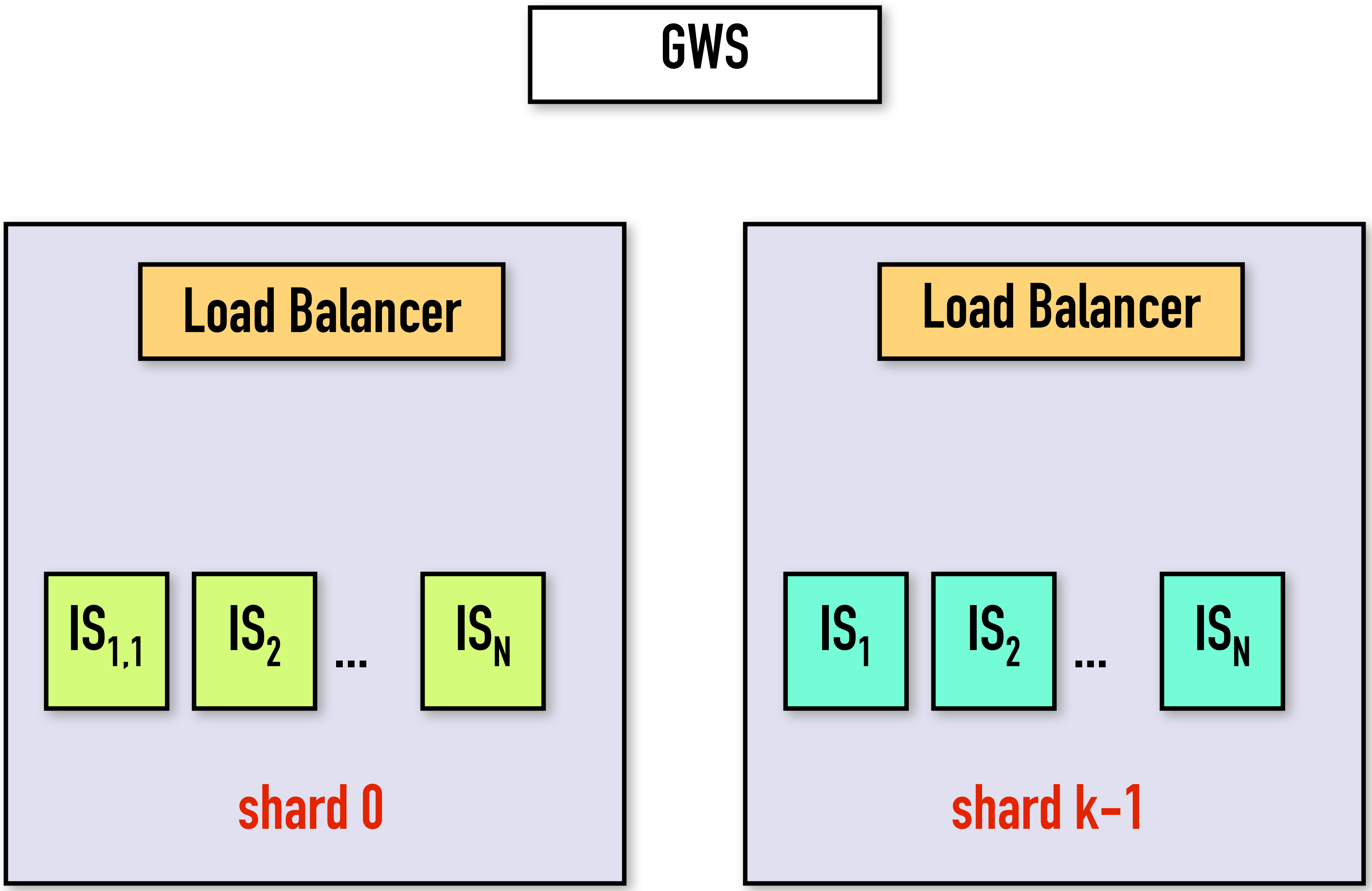
REPLICATION FOR FAULT-TOLERANCE & CAPACITY

- Synchronization index shards:
  - can be avoided because data is read only
  - but one needs to be able to update the index!
- Practical requirements:
  - no downtime by update of index
  - rebalancing of mapping of URLs to shards
    - needs to be supported

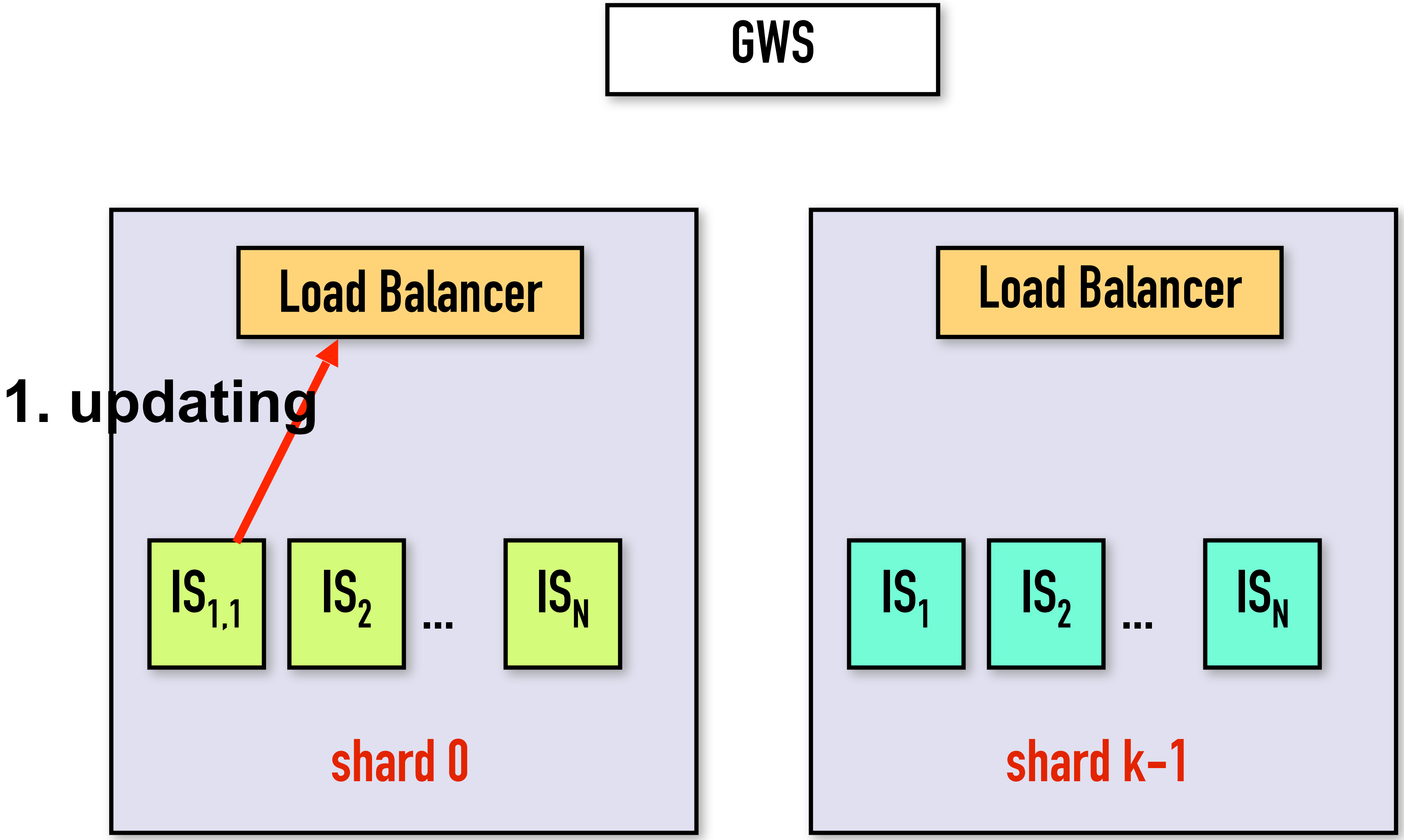
UPDATE OF INDEX

- To update index server
  - divert all requests to other servers in the pool

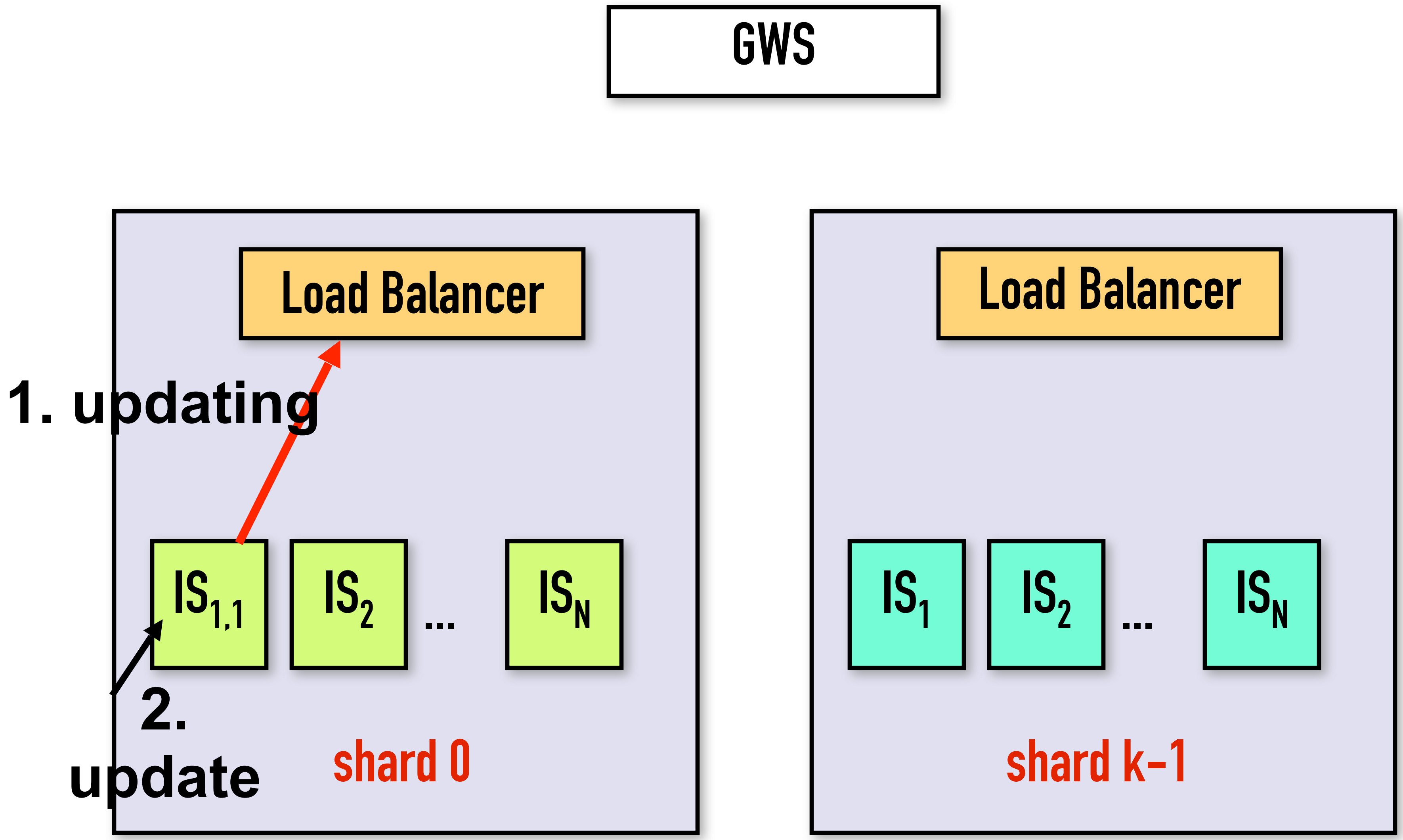
UPDATE OF INDEX



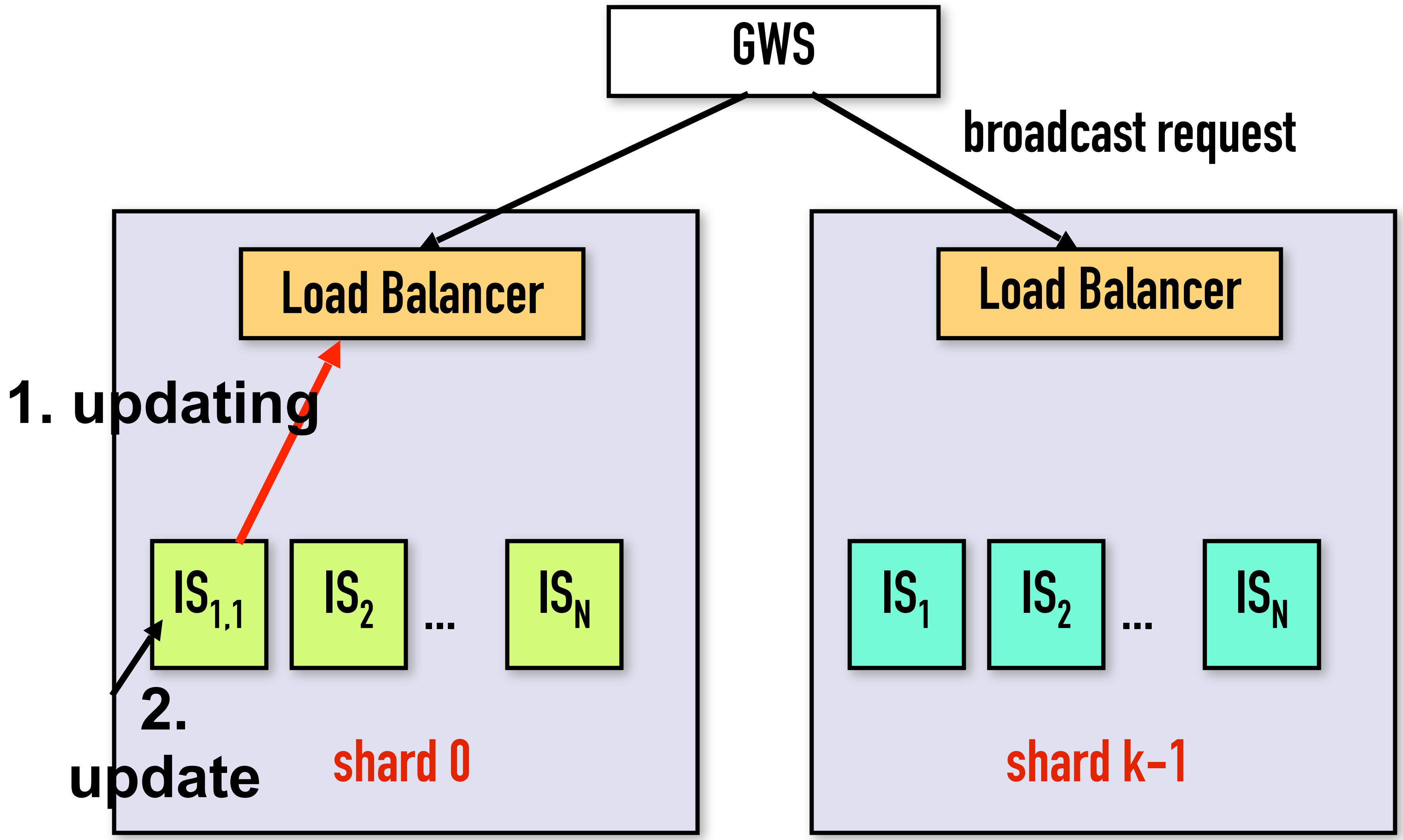
UPDATE OF INDEX



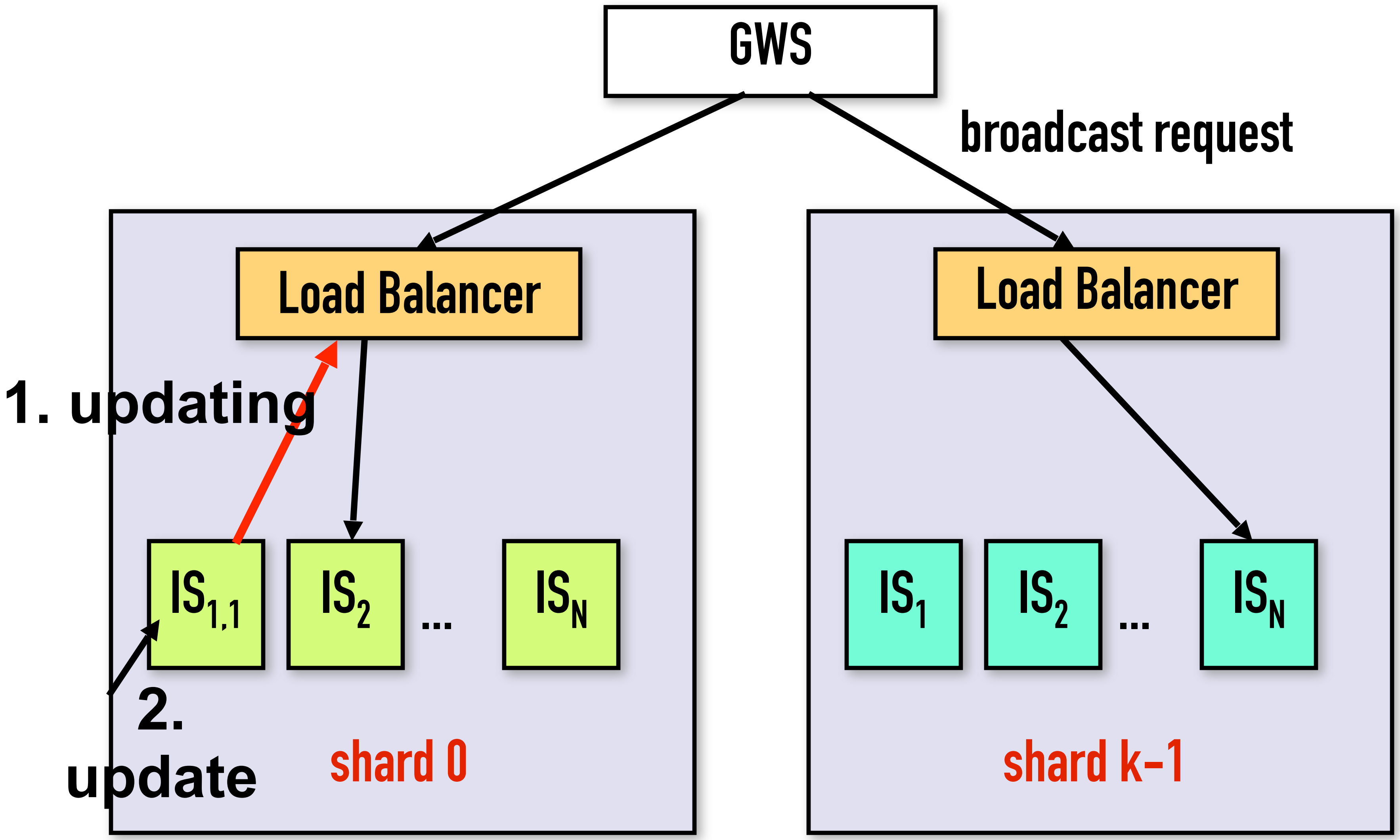
UPDATE OF INDEX



UPDATE OF INDEX



UPDATE OF INDEX



HOMEWORK:

- What if mapping of URLs to index servers changes?
  - e.g., could have double entries, or
  - omitted entries

- How to deal with this?
  - without additional overhead in index servers?

REAL-TIME PROBLEM

- What if index is continuously updated?
  - e.g., news sites, tweets must be indexed in near real-time
  - how can we update index efficiently?

SCALABILITY LESSONS

- **Scalability**:
  - use the inherent parallelism in the application
    - e.g., retrieve doc list in parallel & inexpensive merge
  - use multiple clusters to divide the load
- **Increase the number of shards**
  - to scale with an increase of the index
  - to accommodate slower CPUs
- **Increase the number of servers per pool**
  - to increase throughput of system

REFERENCES

[1] L. A. Barroso, J. Dean, and U. Hölzle. Web search for a planet: The google cluster architecture. IEEE Micro, 23(2):22–28, 2003.

[2] Luiz André Barroso and Urs Hölzle, "The Datacenter as a Computer", 2013 by Morgan & Claypool. (available online)