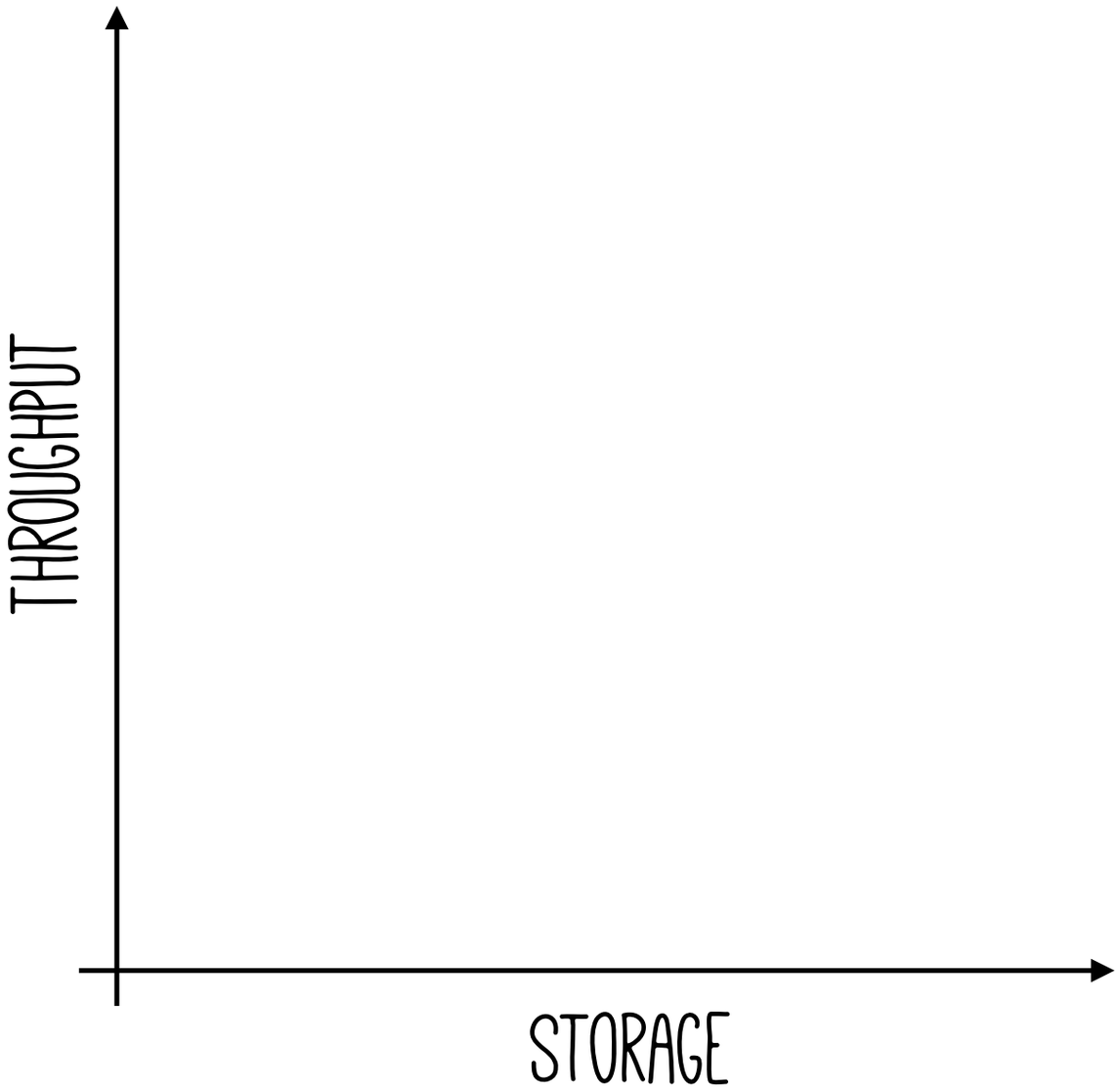
BlowFish:
Dynamic Storage-Performance Tradeoff in Data Stores

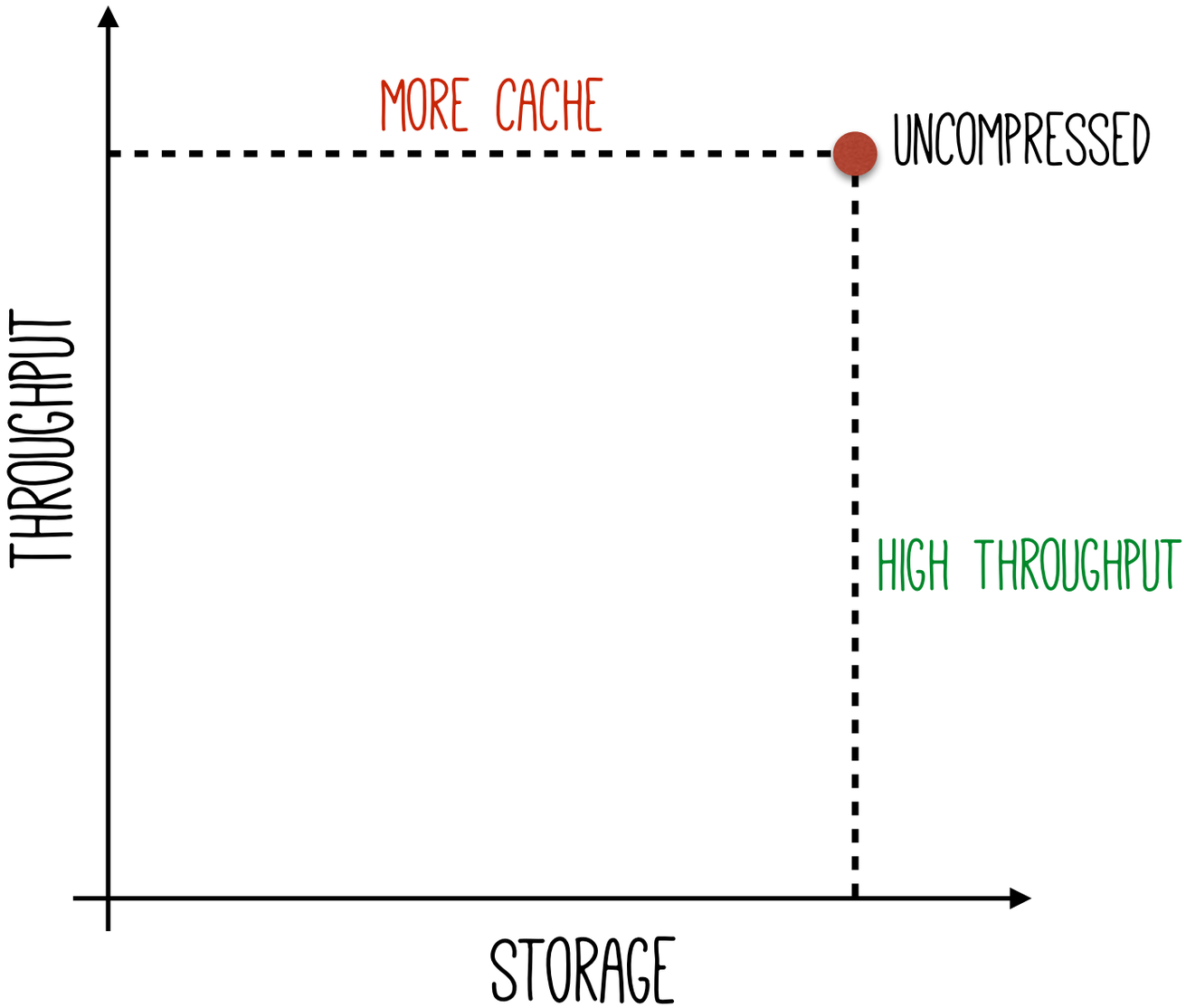
Anurag Khandelwal, Rachit Agarwal, Ion Stoica


Existing Data Stores

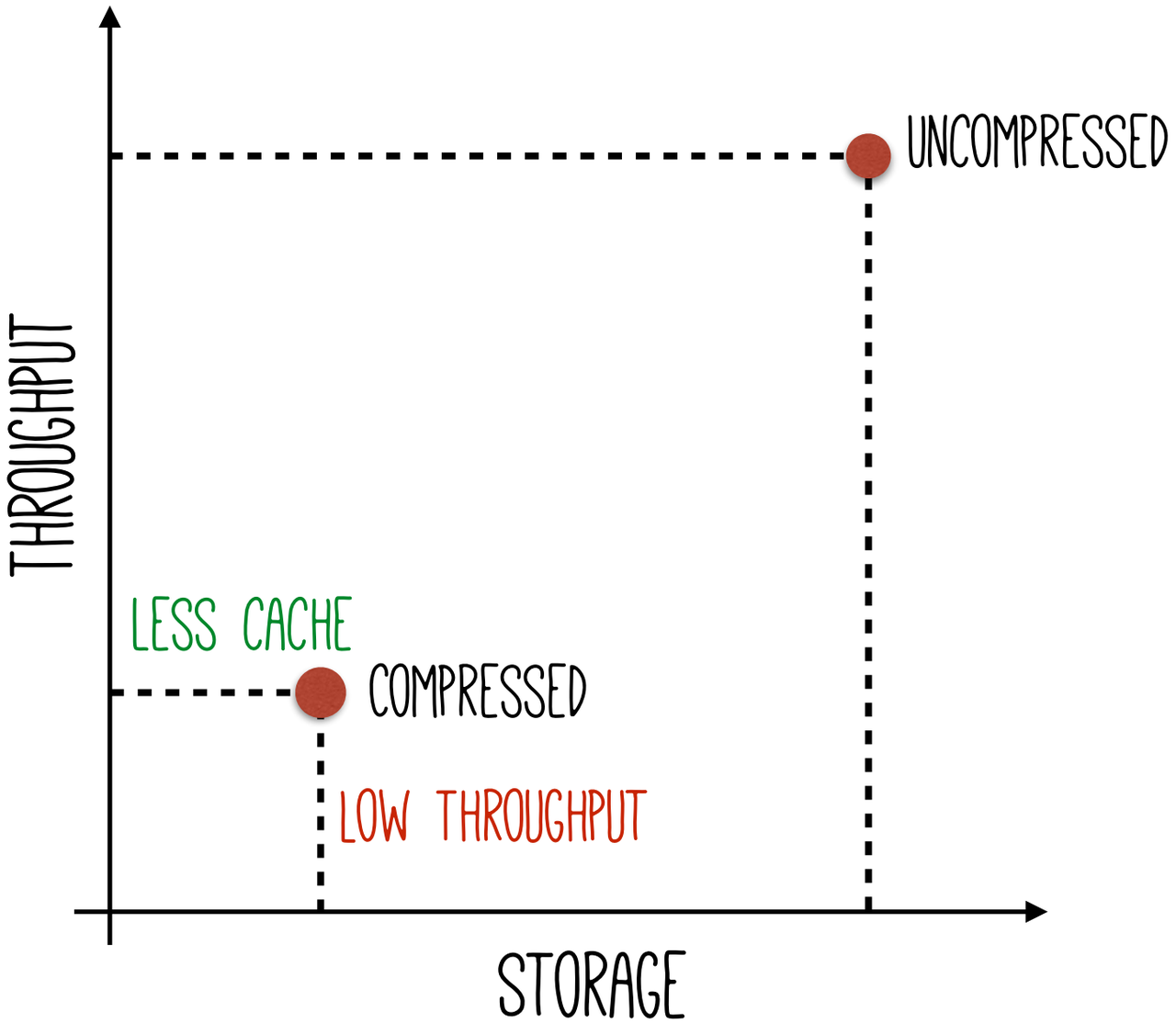
Existing Data Stores



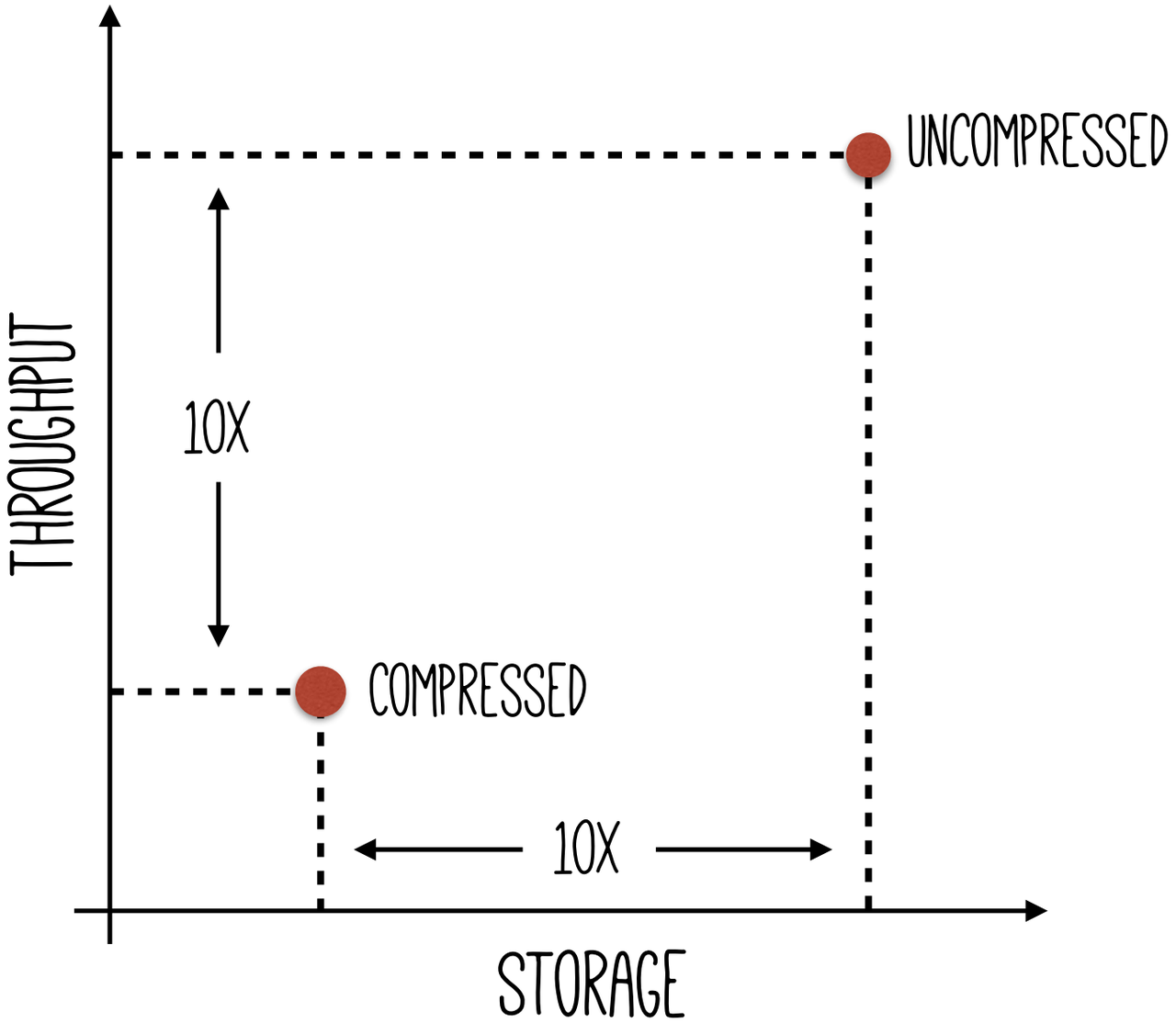
Existing Data Stores



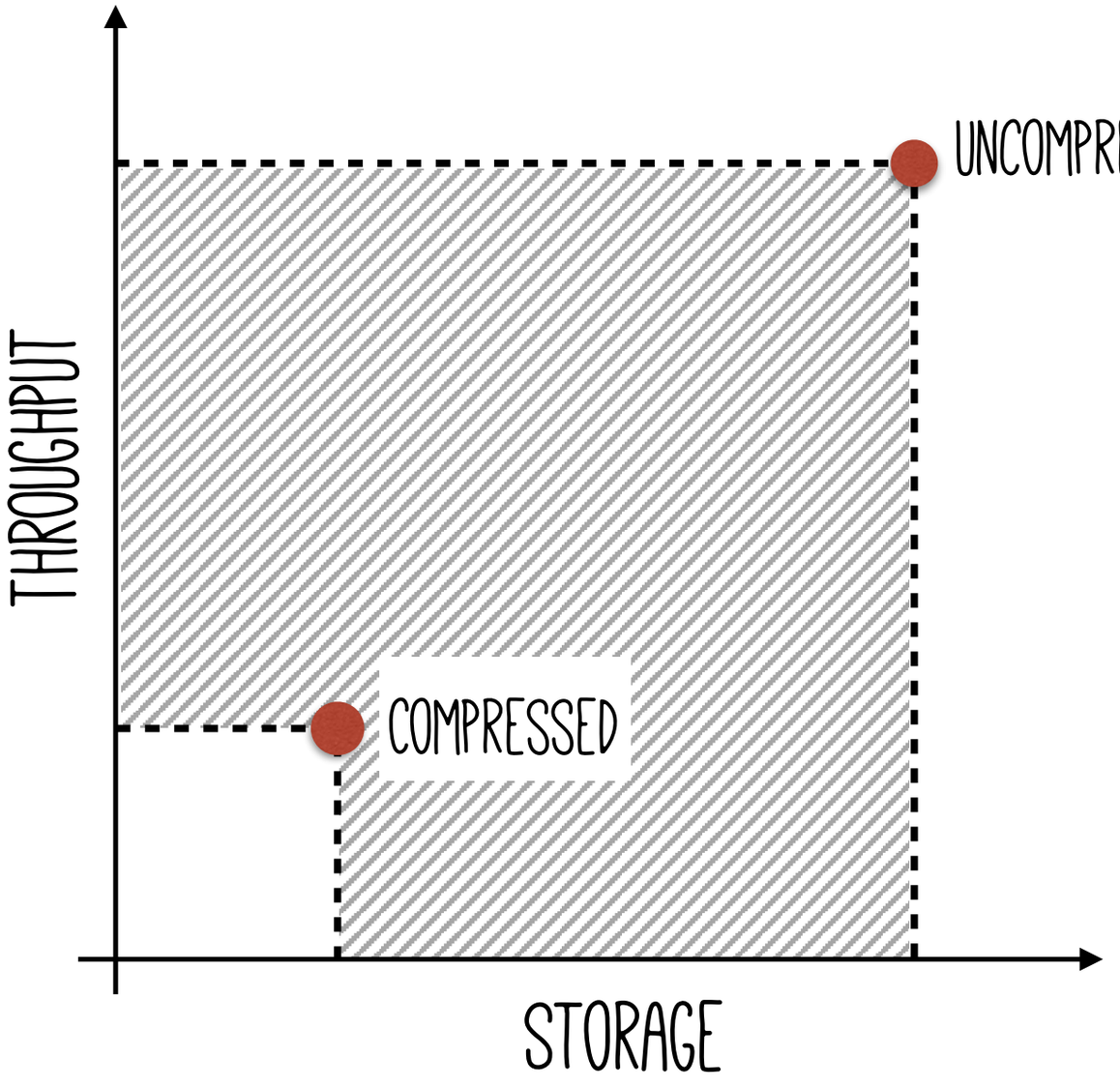
Existing Data Stores



Existing Data Stores

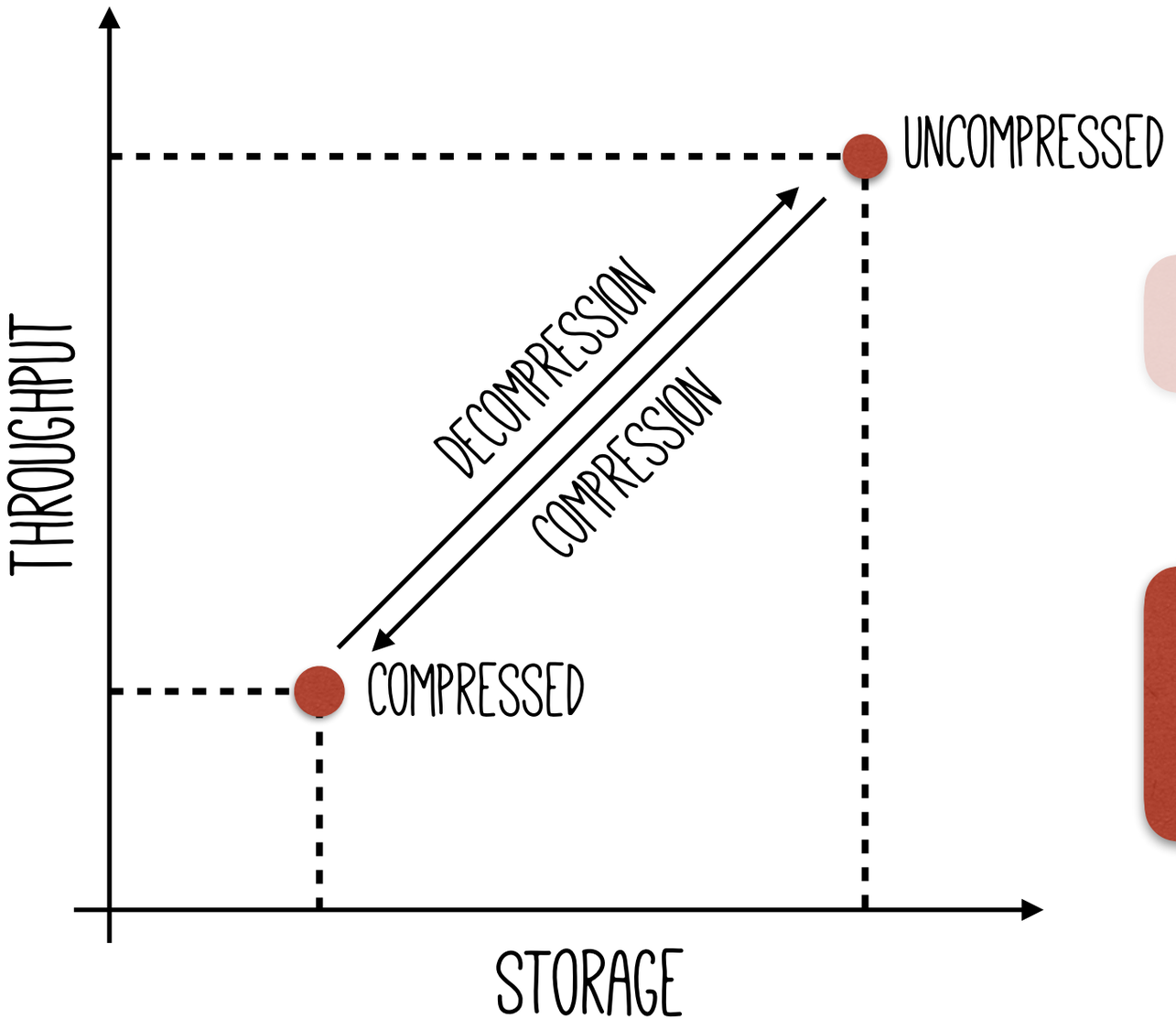


Existing Data Stores



Unachievable points

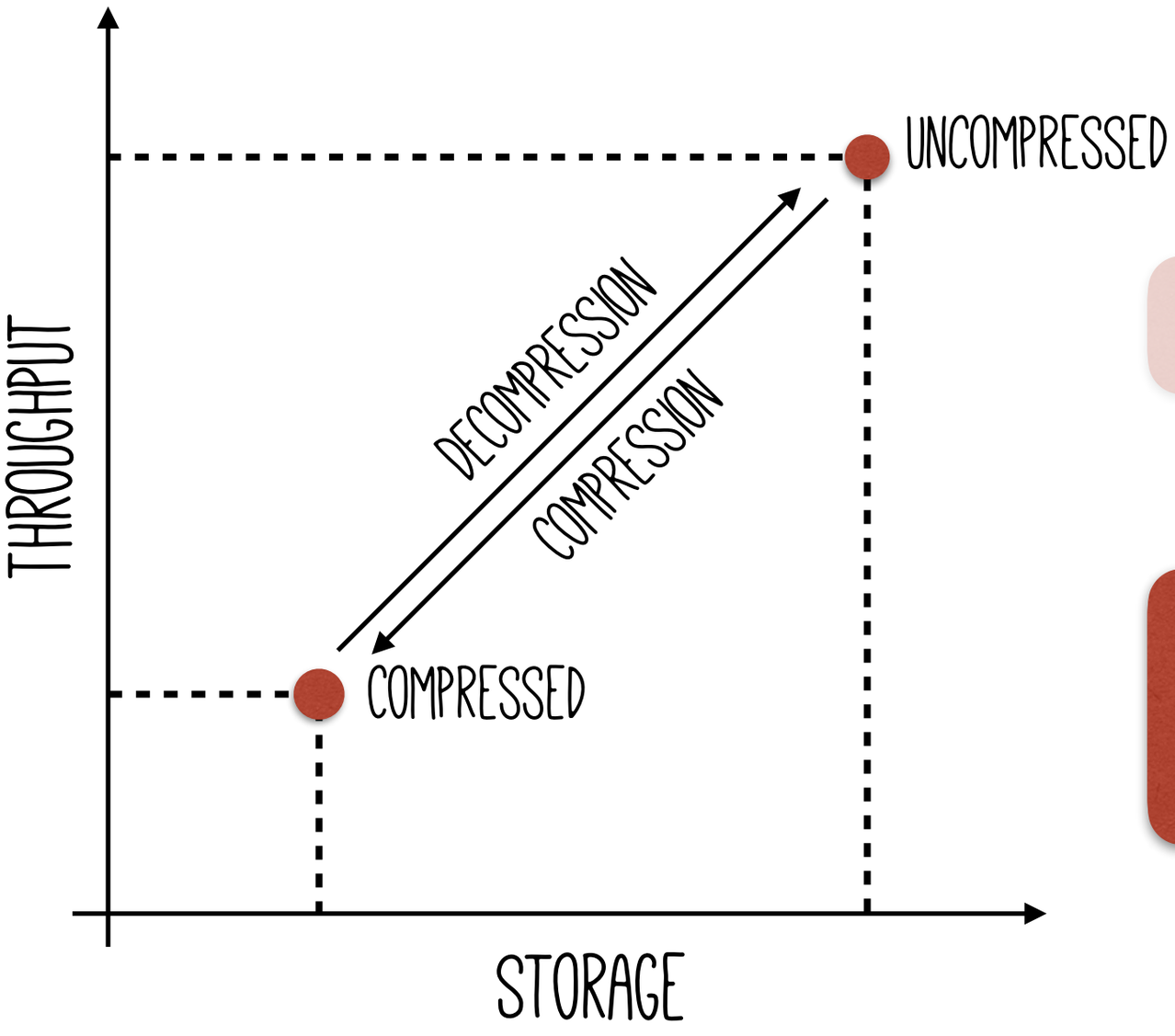
Existing Data Stores



Unachievable points

Switching between the two
incurs high latency & CPU

Existing Data Stores



Unachievable points

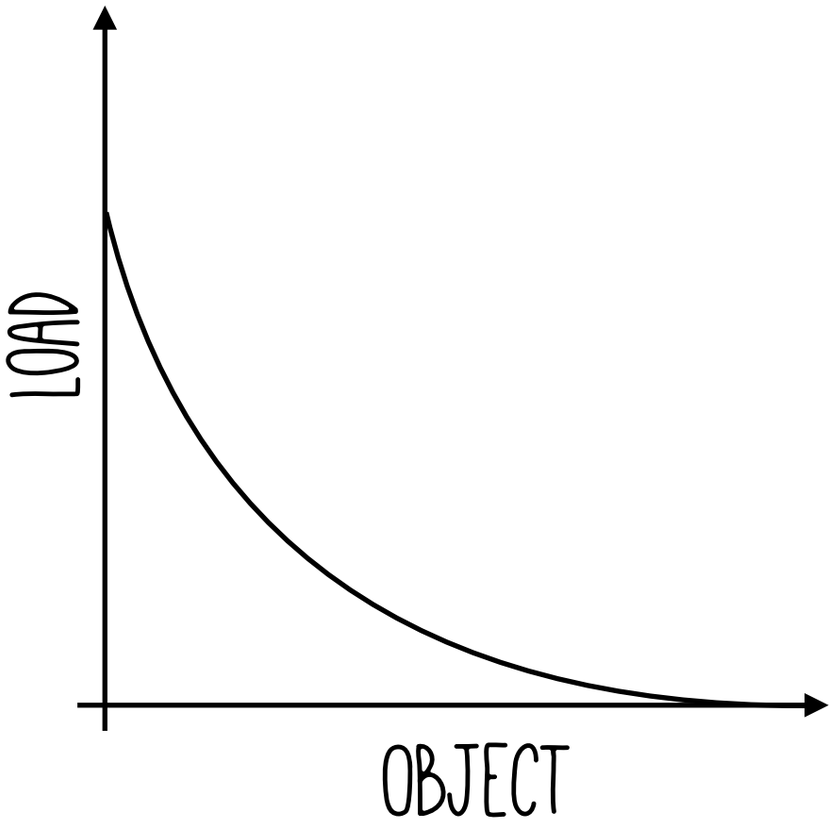
Switching between the two incurs high latency & CPU

Leads to degraded performance when underlying workload or infrastructure changes

A Motivating Example

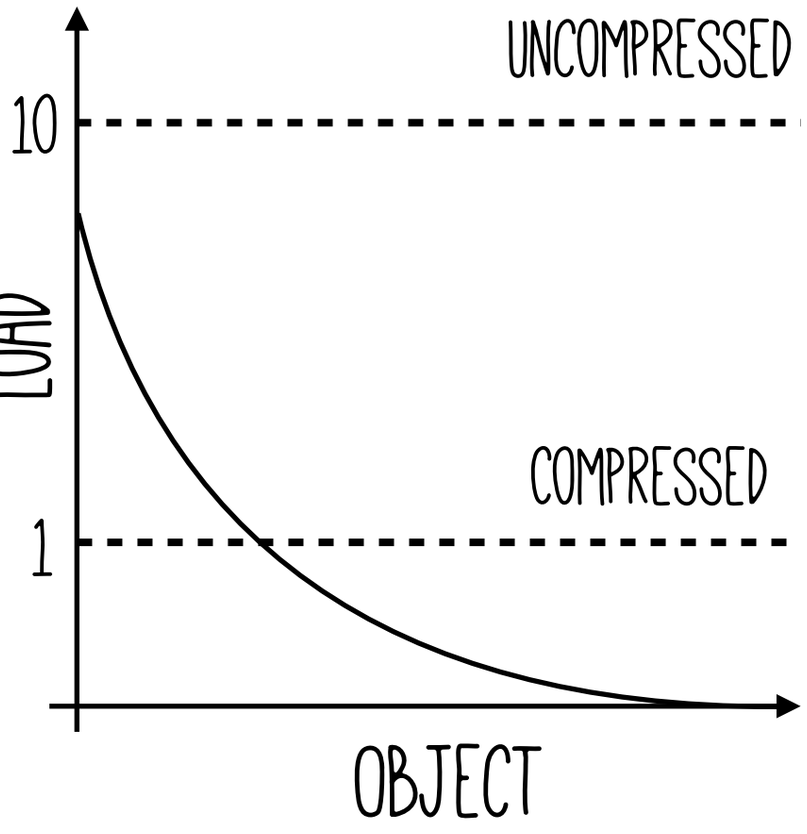
A Motivating Example

Load across items heavily skewed



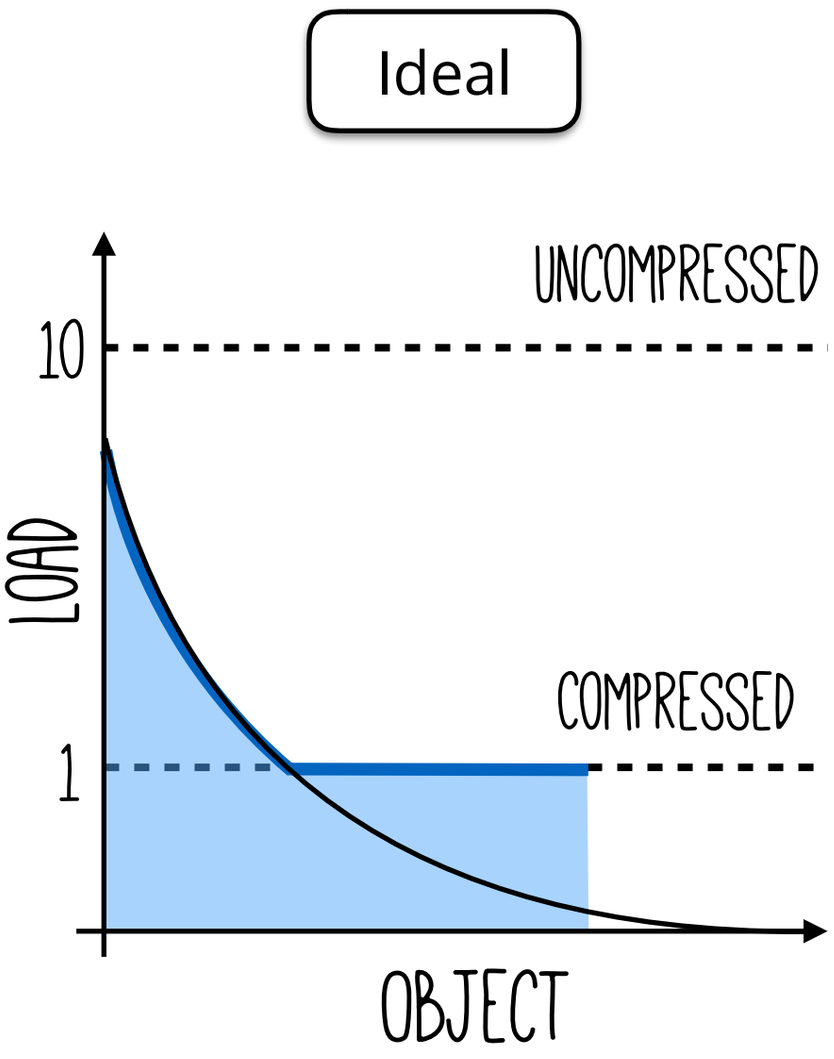
A Motivating Example

Load across items heavily skewed



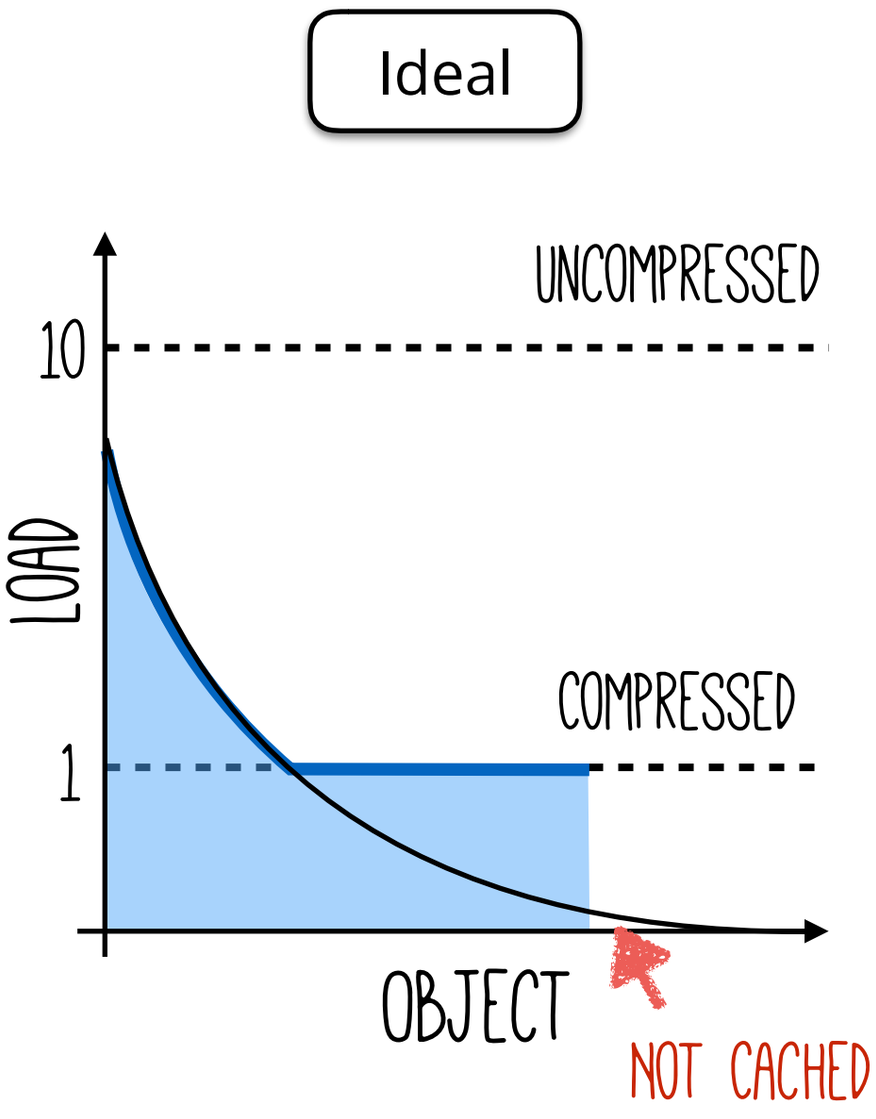
A Motivating Example

Load across items heavily skewed



A Motivating Example

Load across items heavily skewed

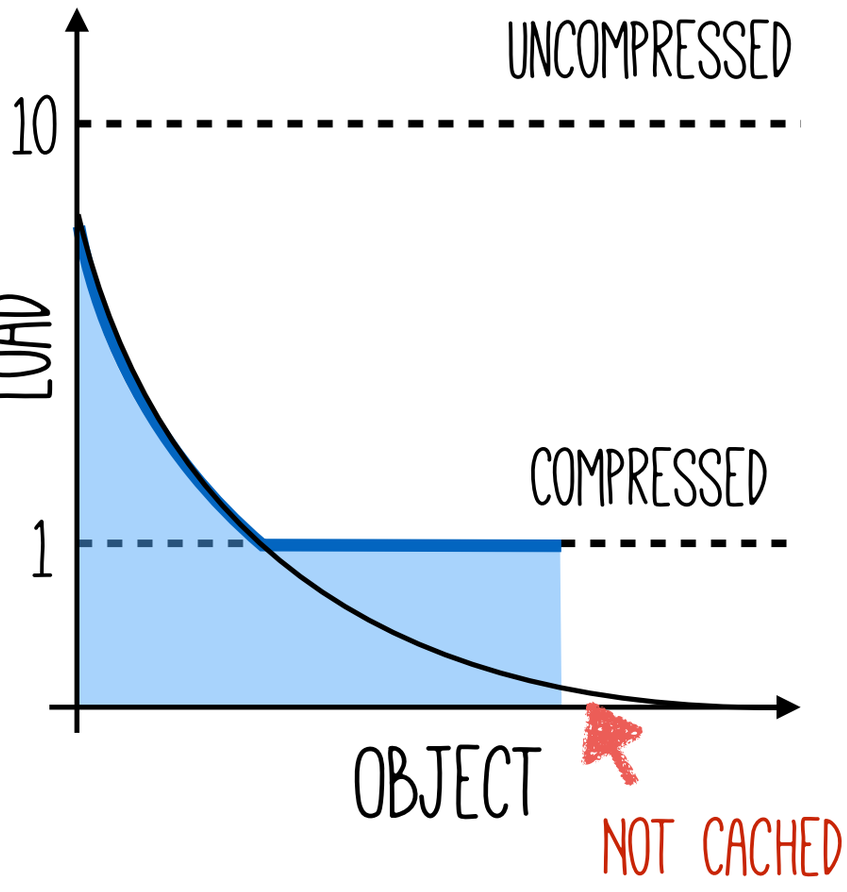


A Motivating Example

Load across items heavily skewed

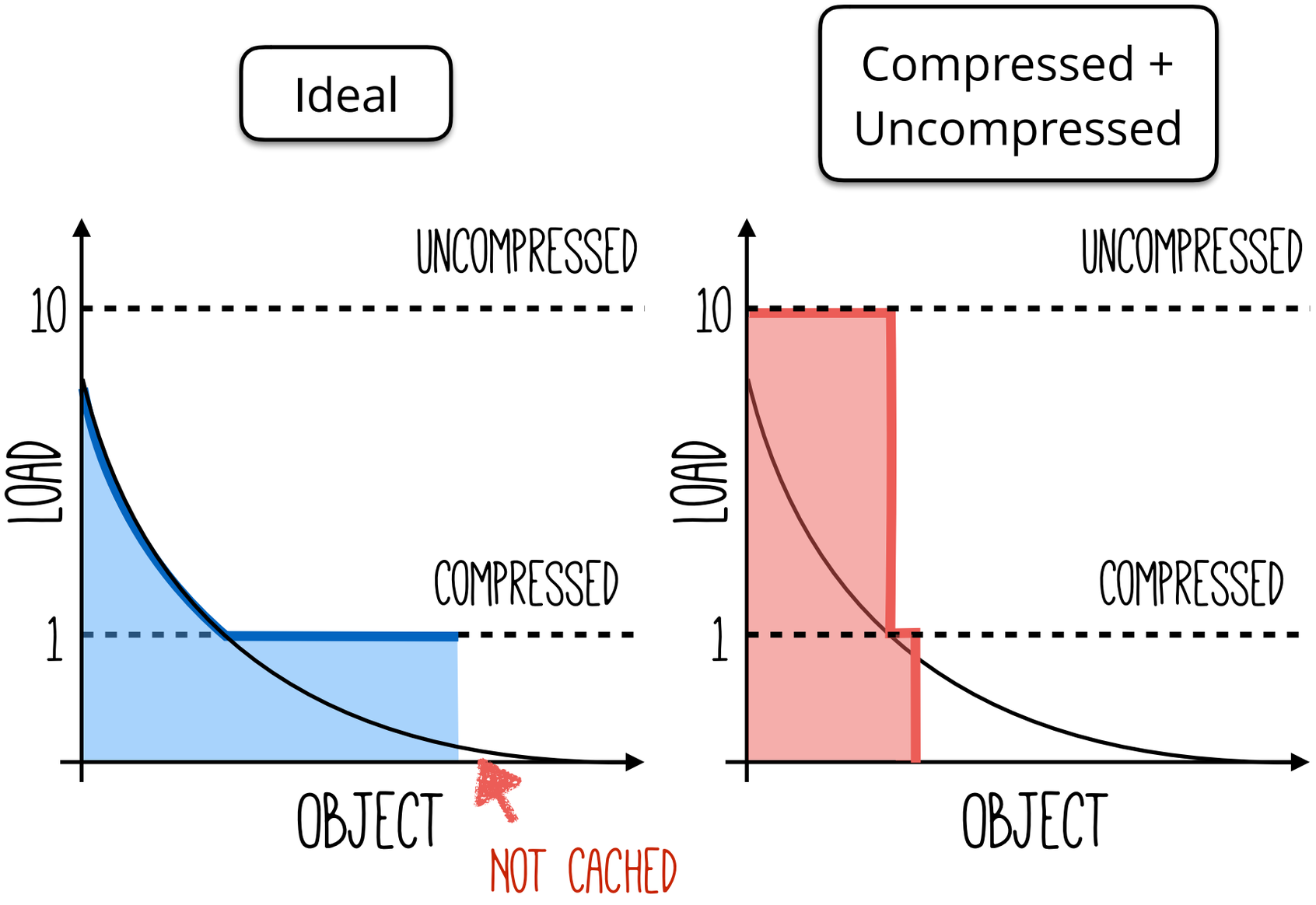
Ideal

Compressed + Uncompressed



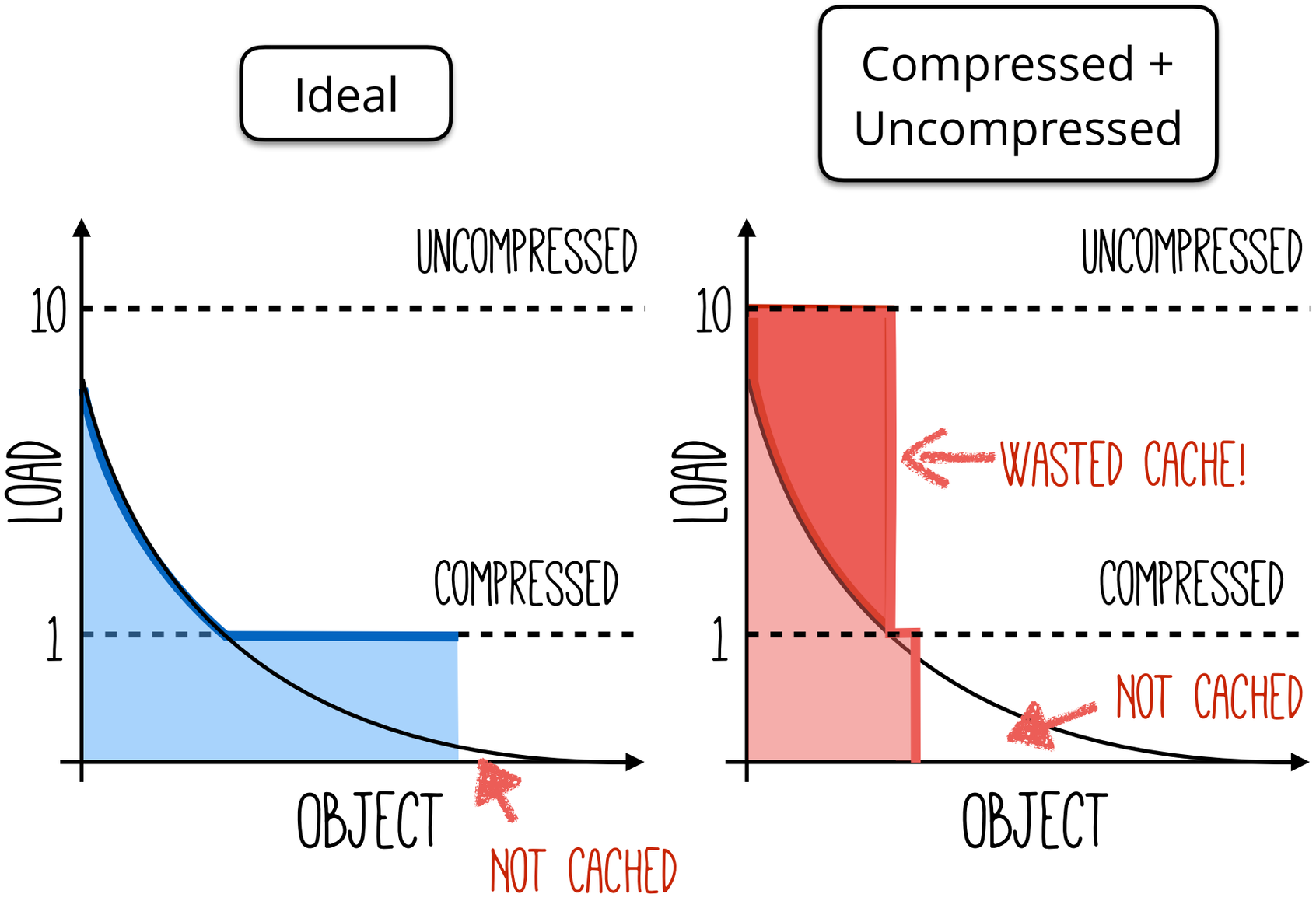
A Motivating Example

Load across items heavily skewed



A Motivating Example

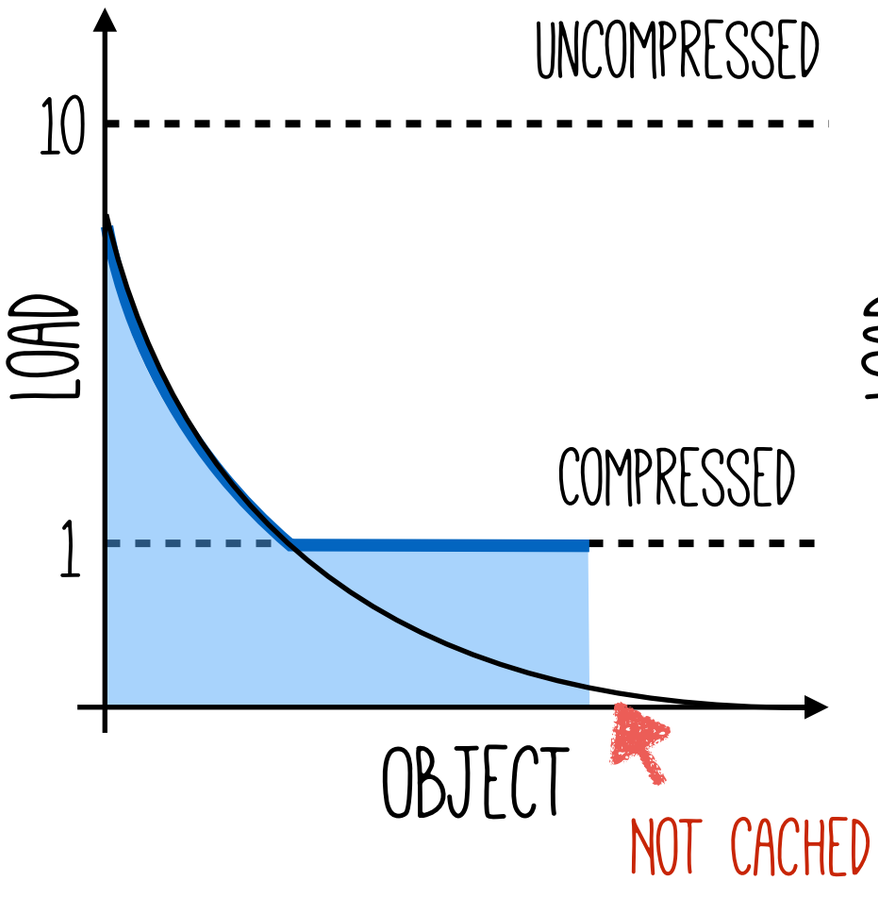
Load across items heavily skewed



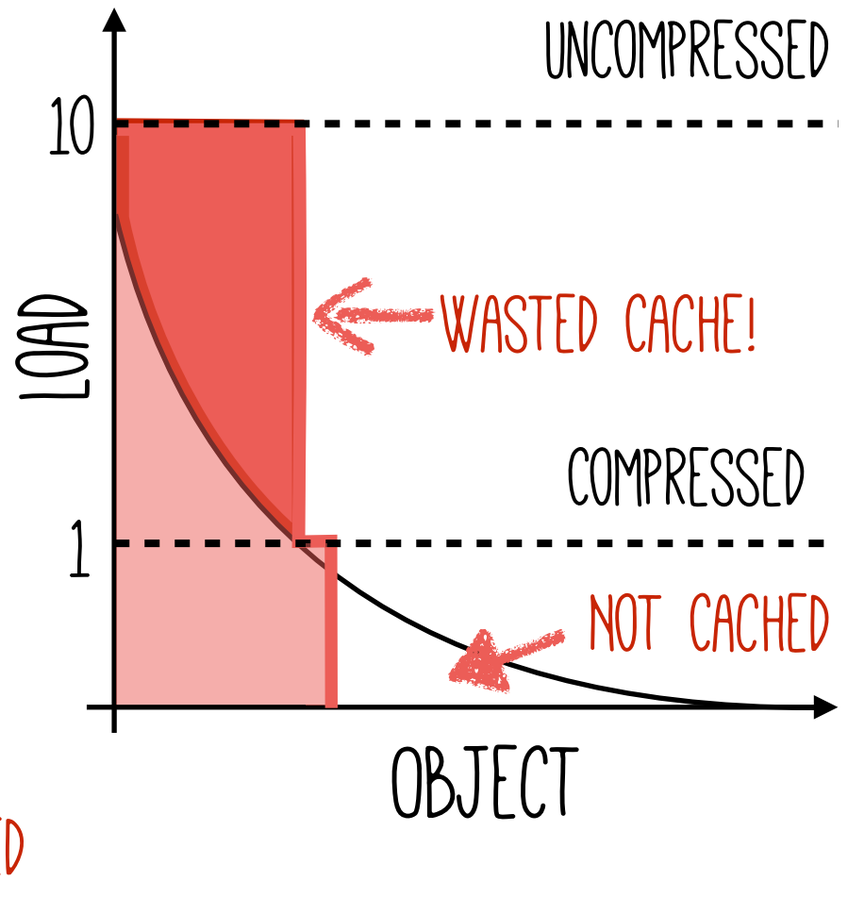
A Motivating Example

Load across items heavily skewed

Ideal



Compressed + Uncompressed

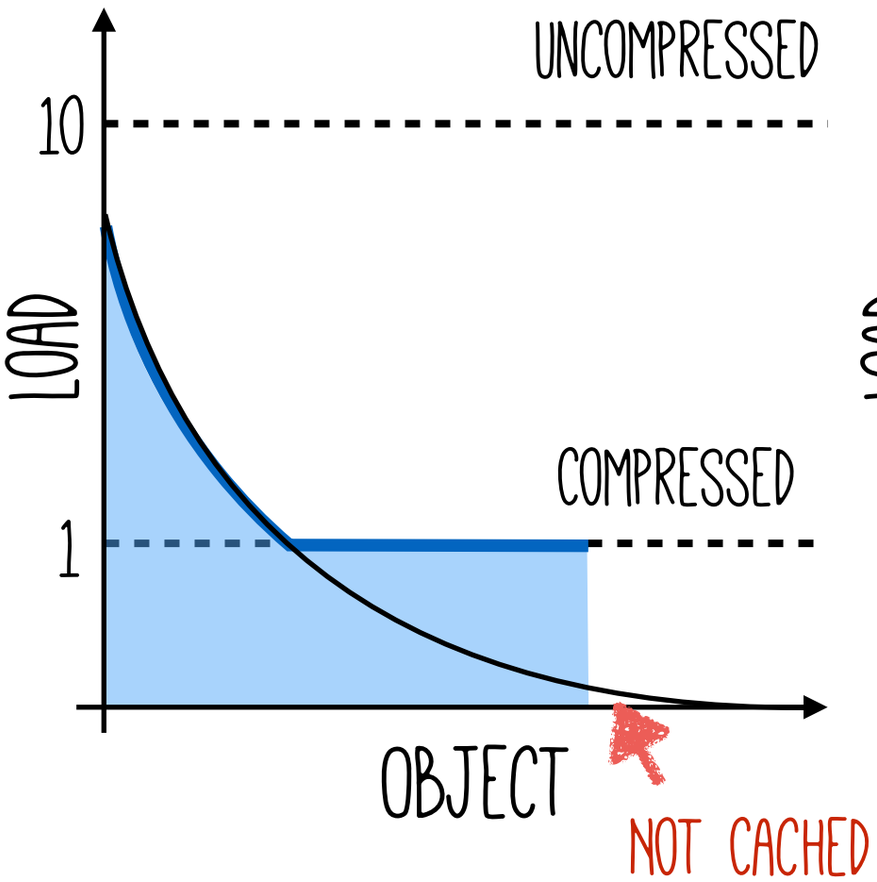


Selective Replication: #Replicas α Load

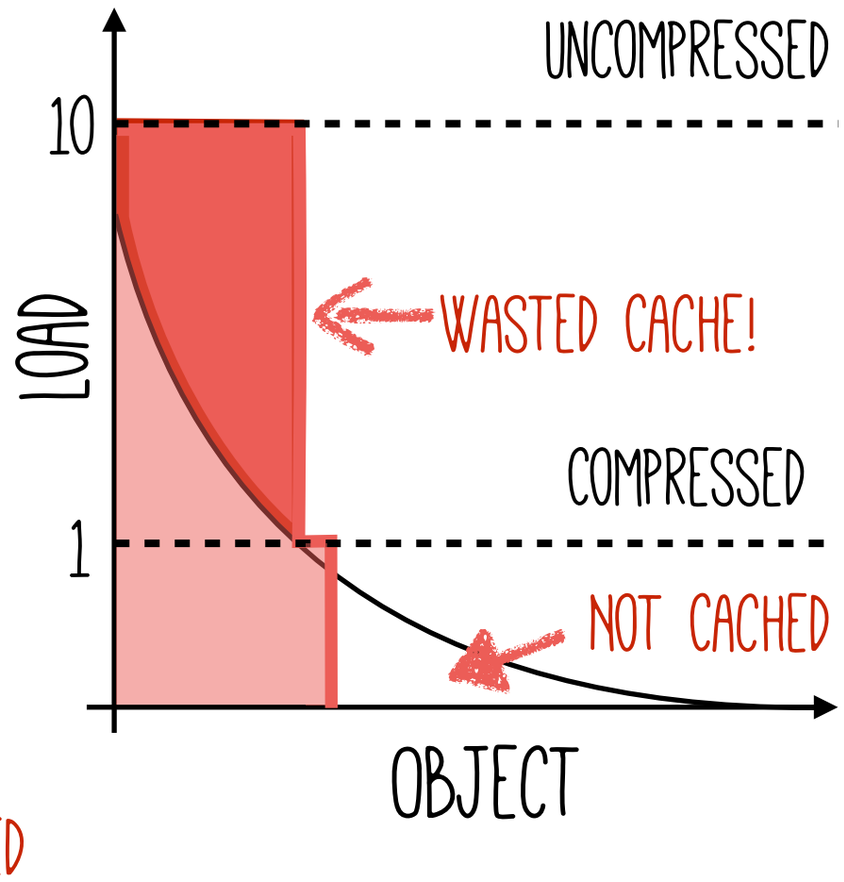
A Motivating Example

Load across items heavily skewed

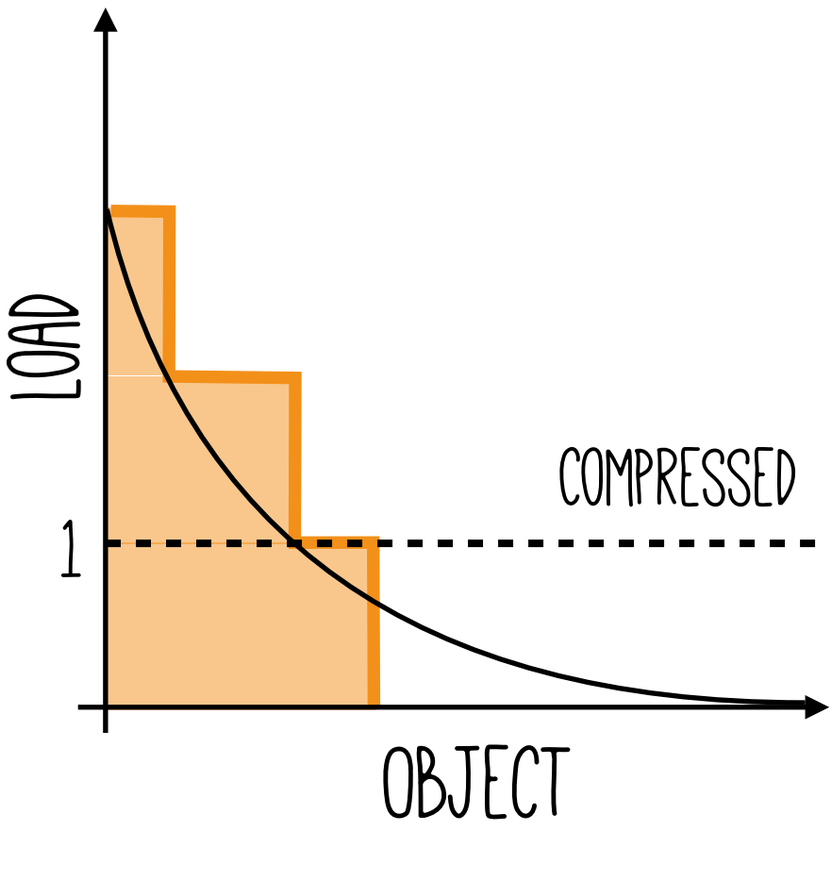
Ideal



Compressed + Uncompressed



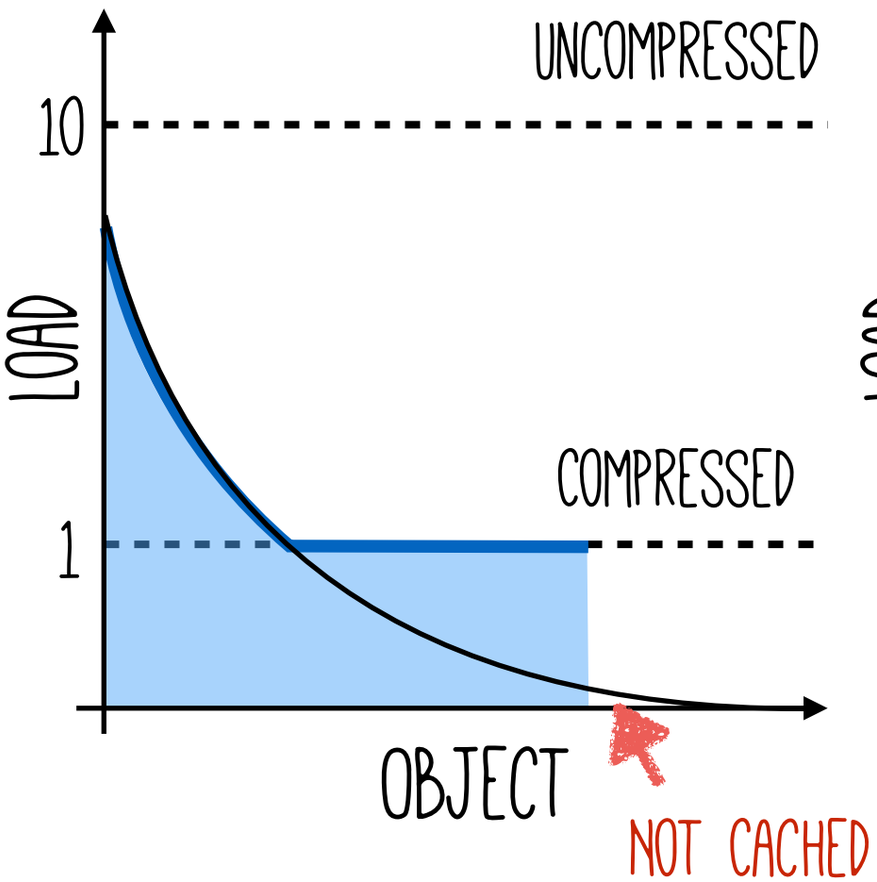
Selective Replication: #Replicas α Load



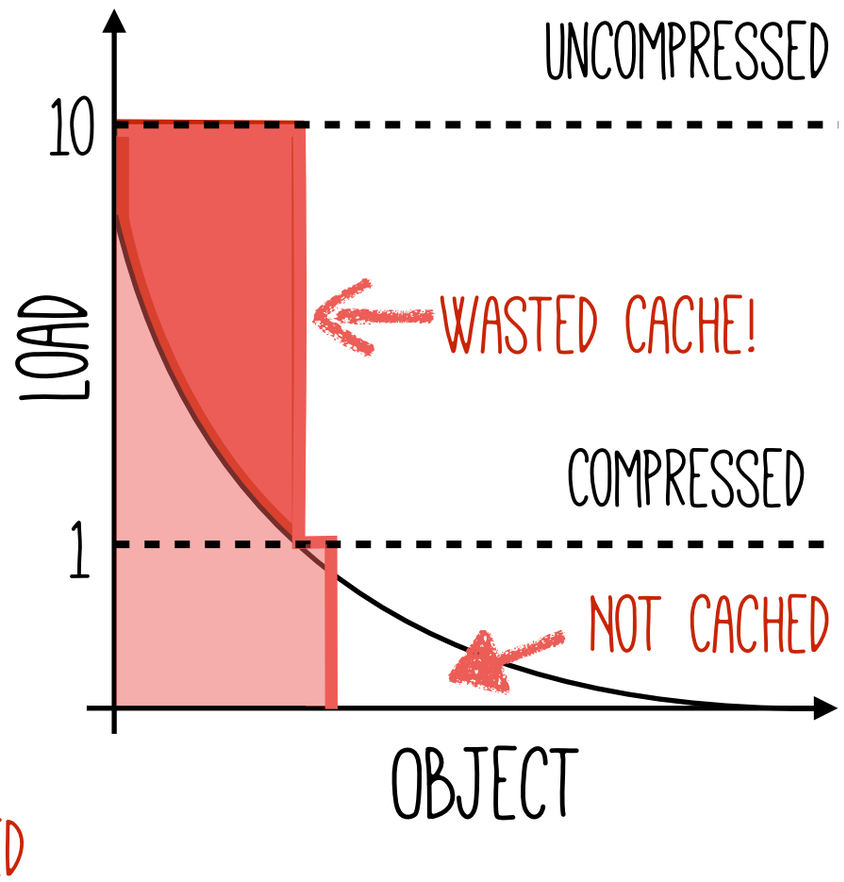
A Motivating Example

Load across items heavily skewed

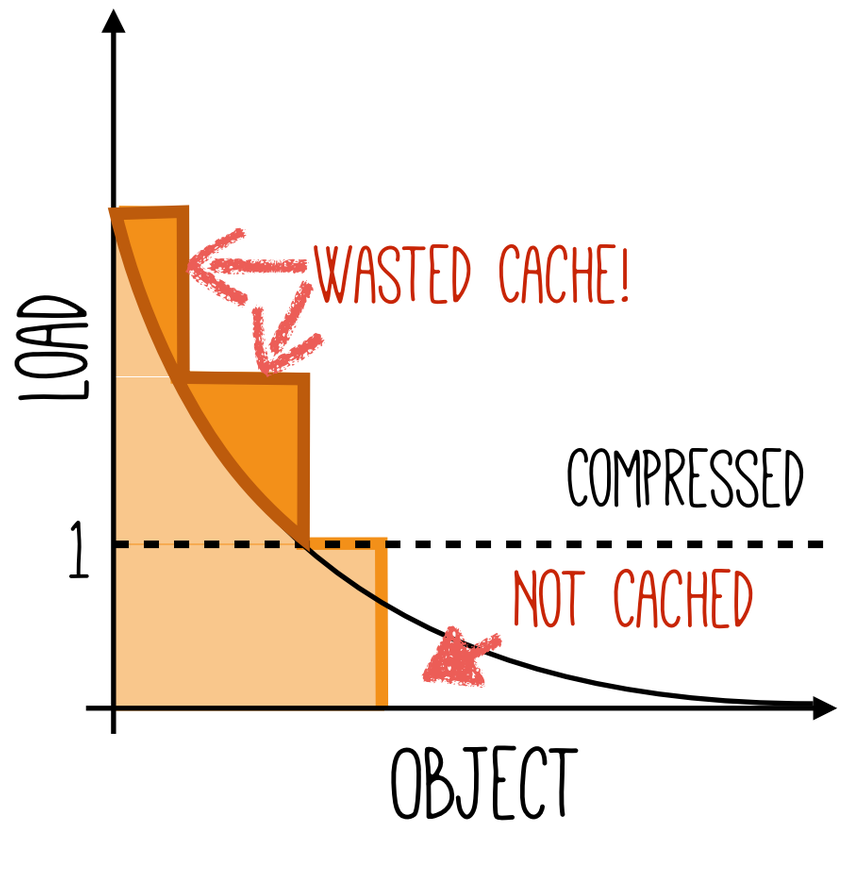
Ideal



Compressed + Uncompressed



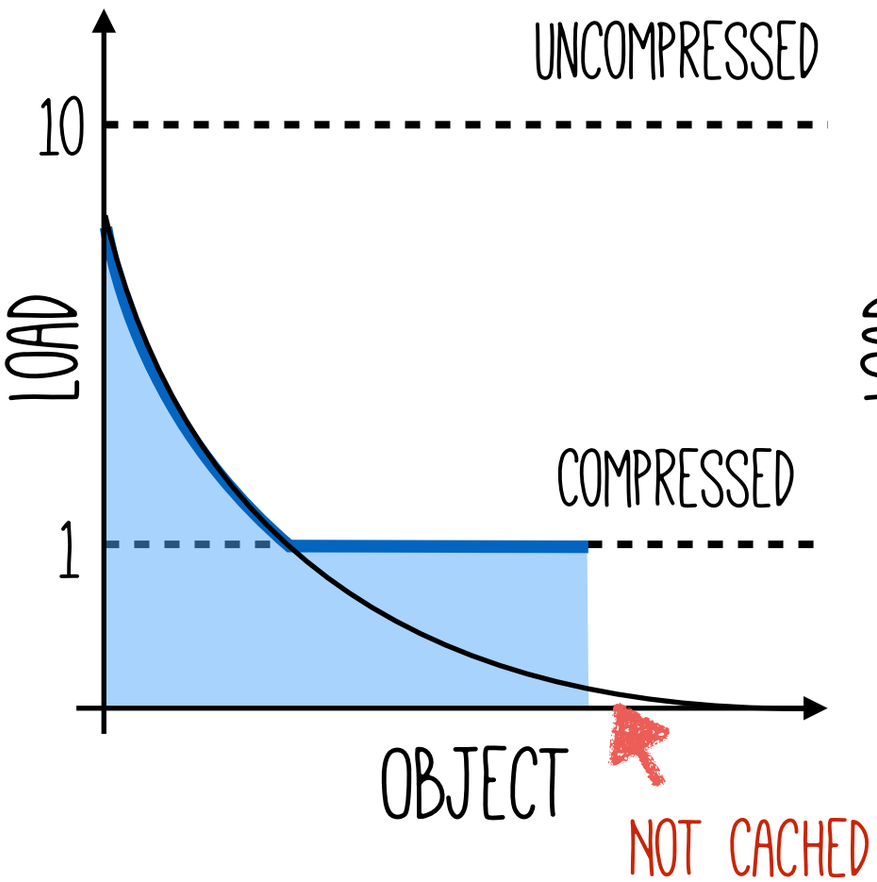
Selective Replication: #Replicas α Load



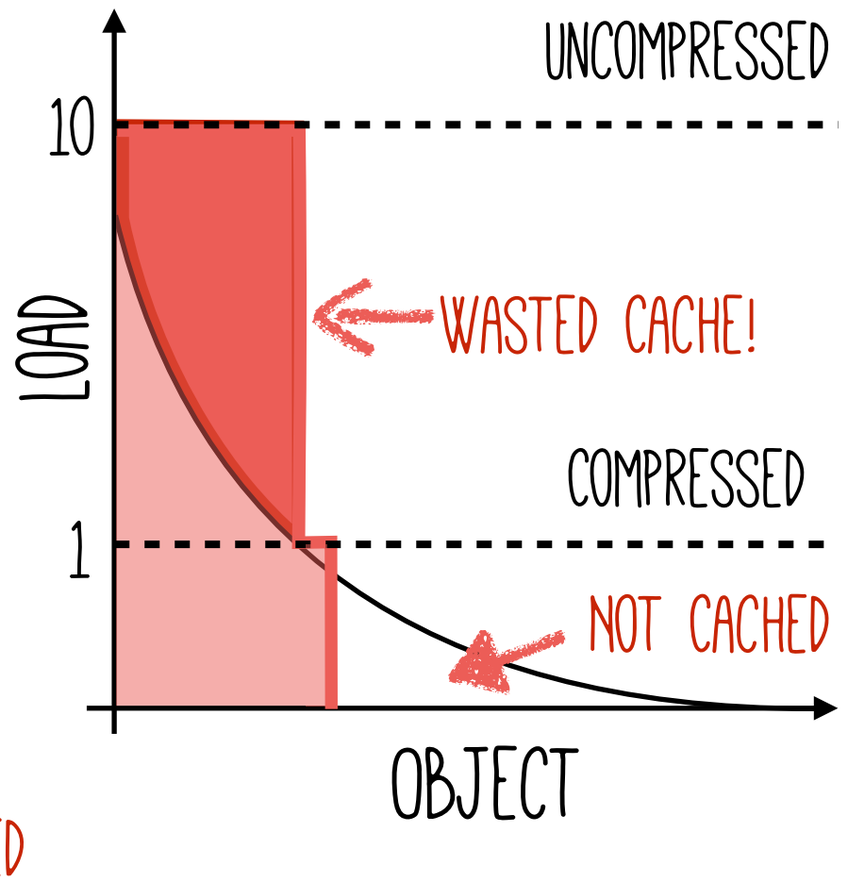
A Motivating Example

Load across items heavily skewed

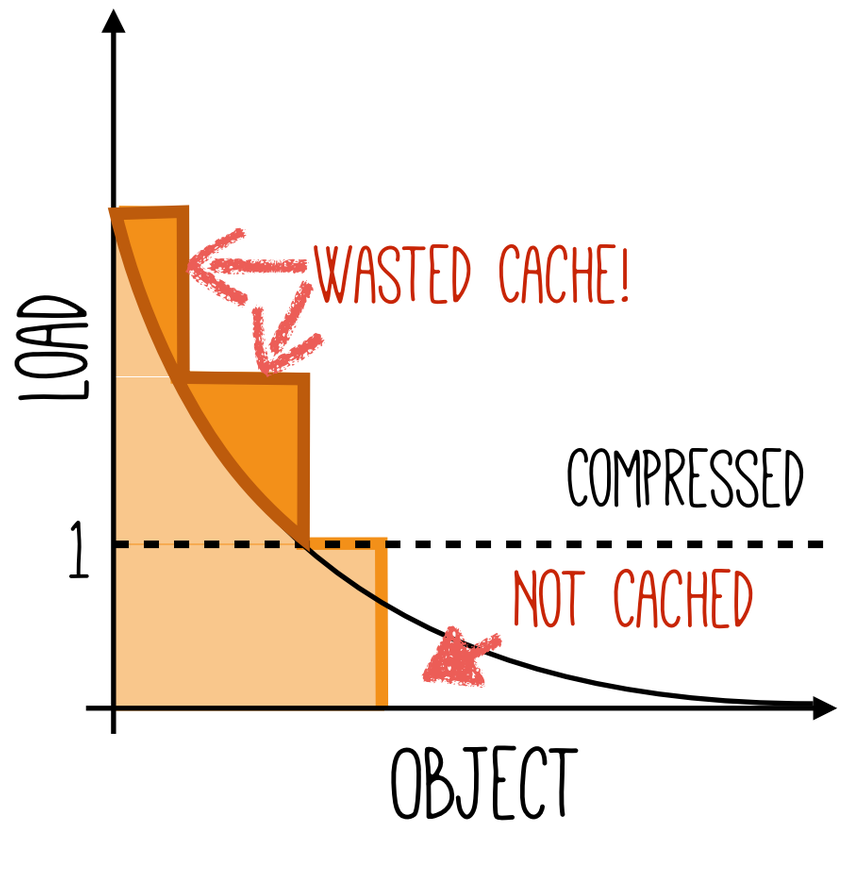
Ideal



Compressed + Uncompressed



Selective Replication: #Replicas α Load

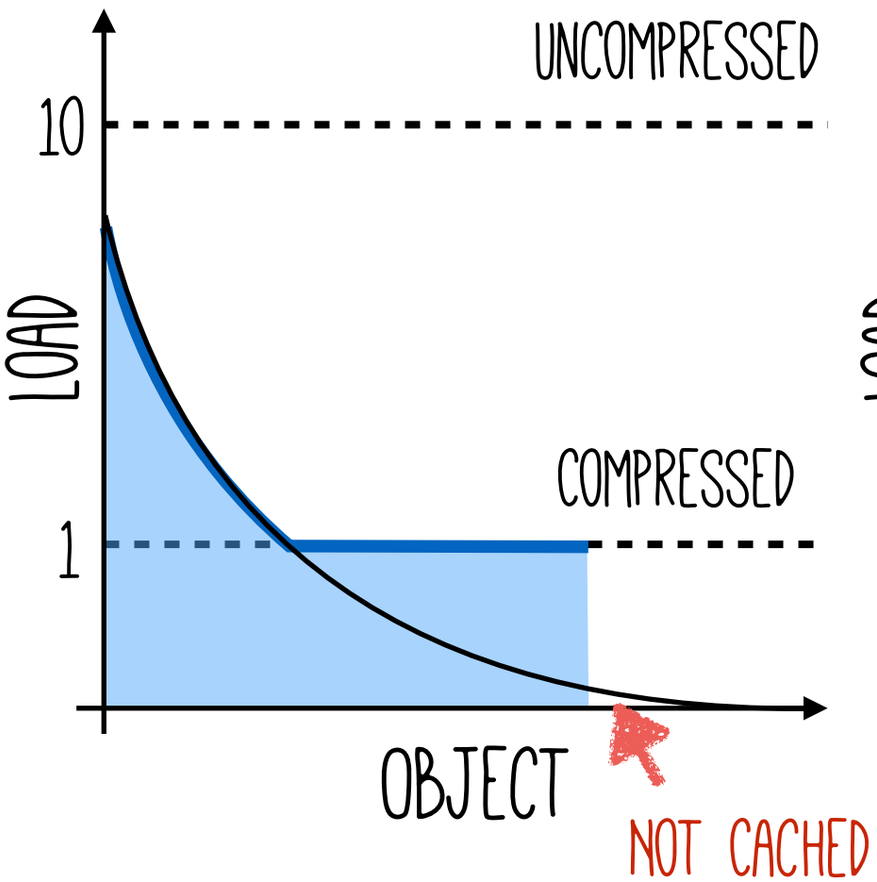


Load changes over time

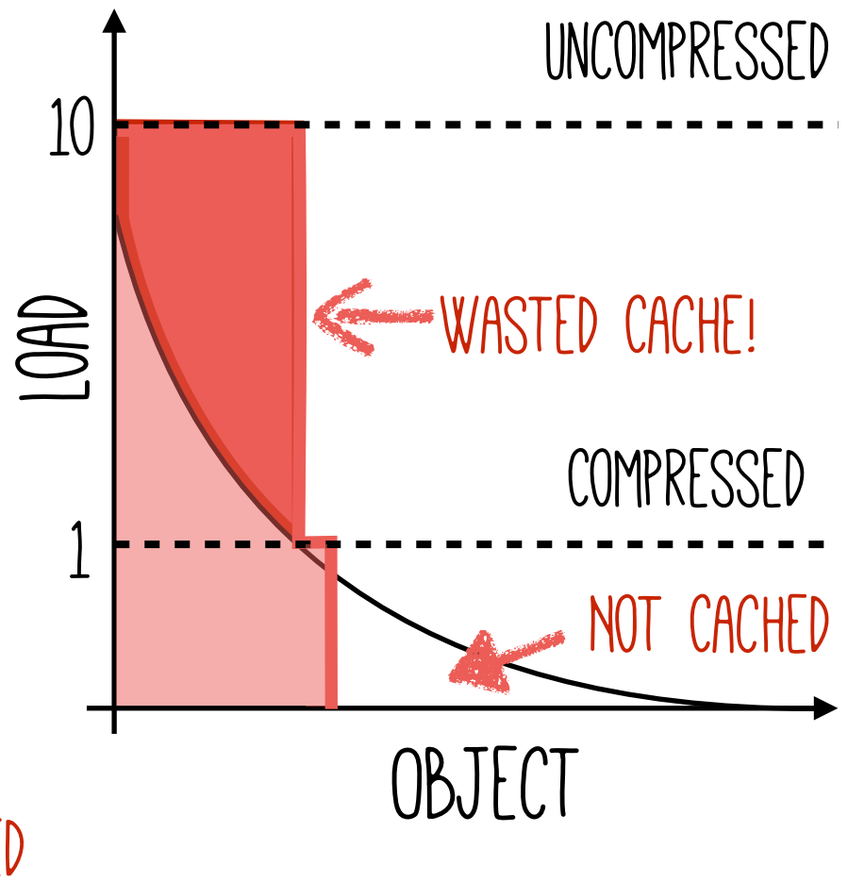
A Motivating Example

Load across items heavily skewed

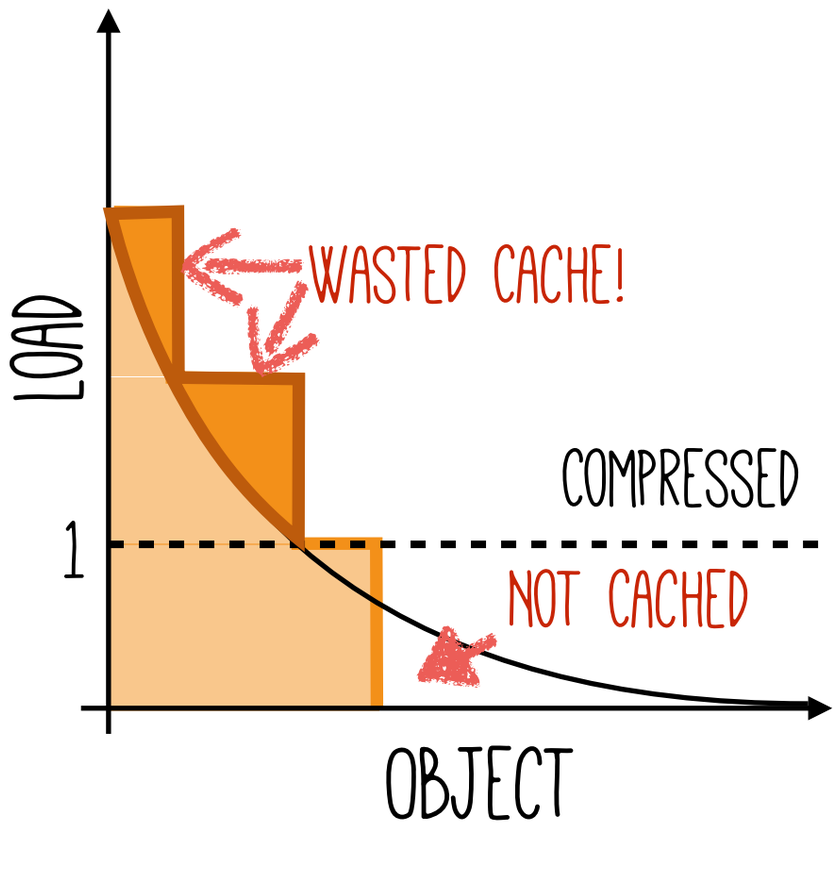
Ideal



Compressed + Uncompressed

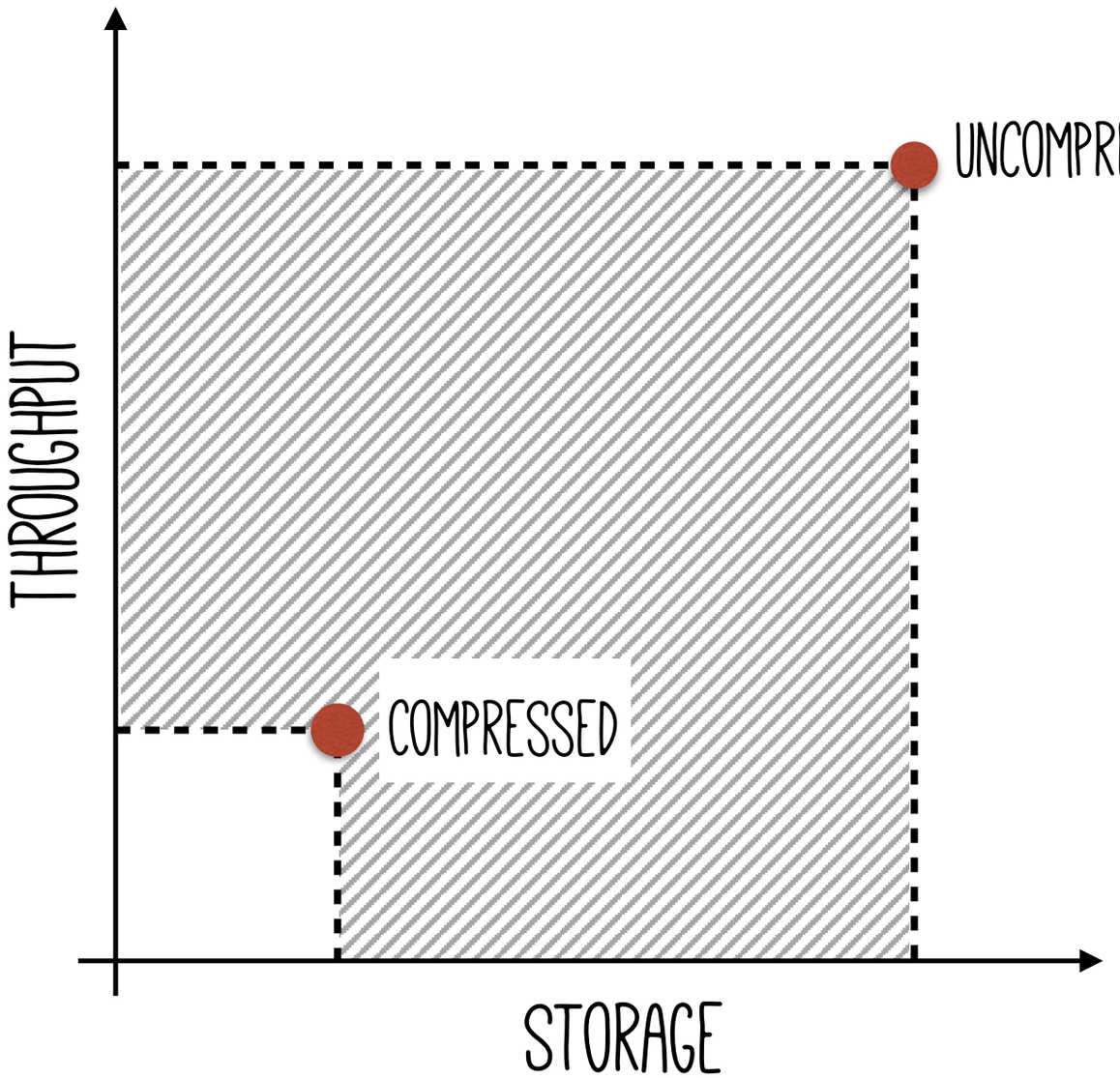


Selective Replication: #Replicas α Load

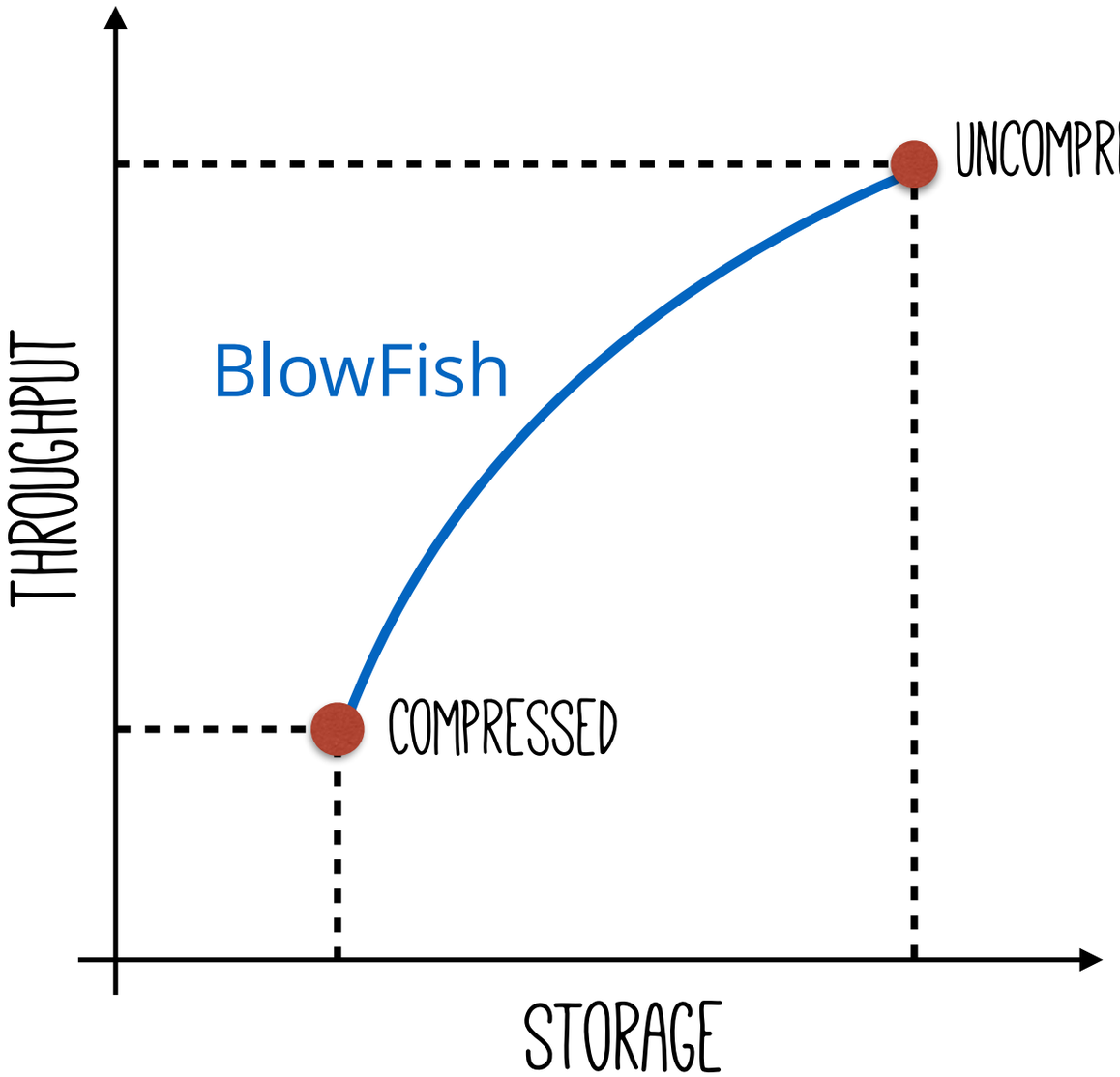


Load changes over time → Degraded performance

BlowFish

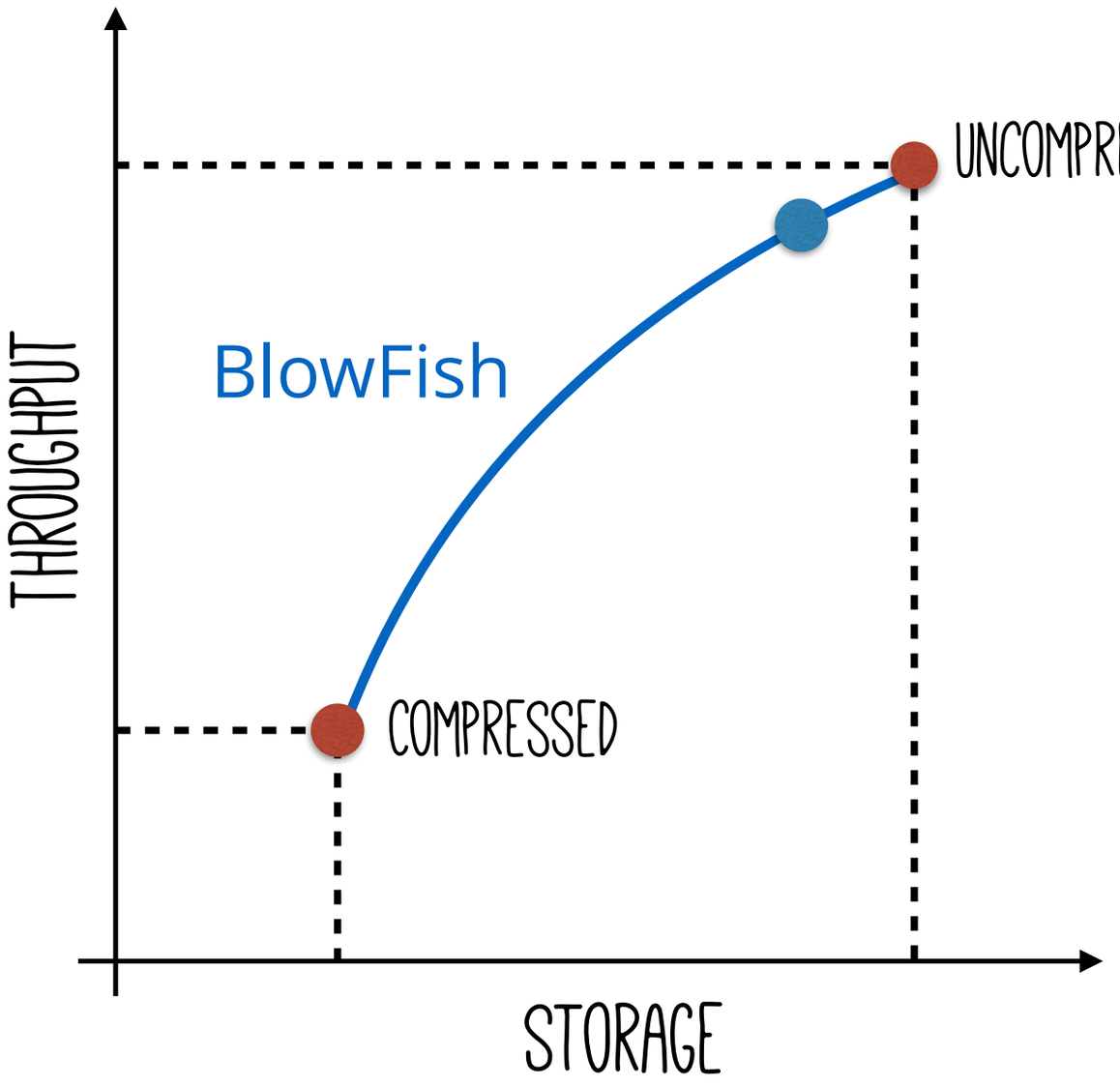


BlowFish



Smooth Tradeoff Curve

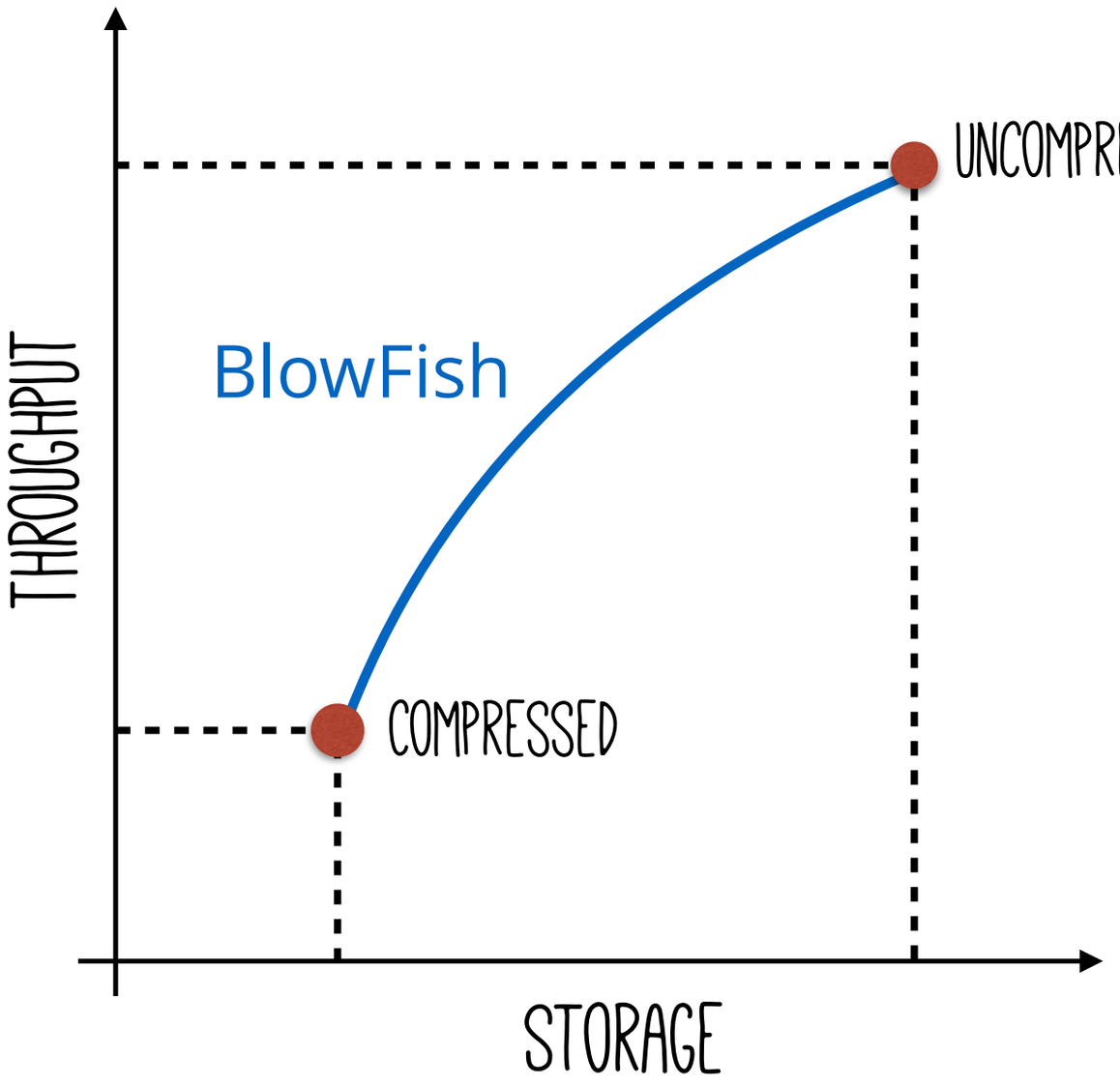
BlowFish



Smooth Tradeoff Curve

Dynamic Navigation

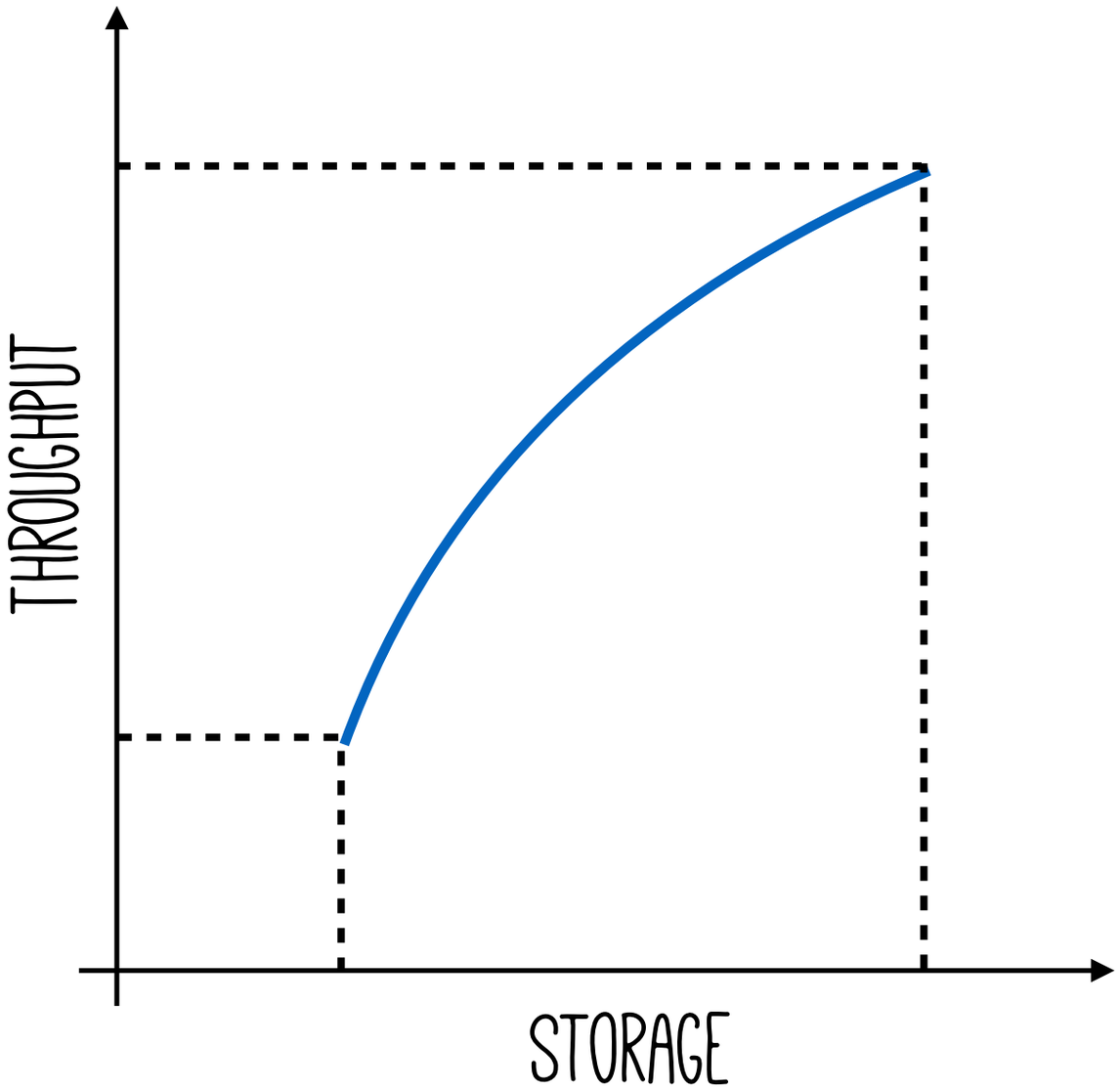
BlowFish

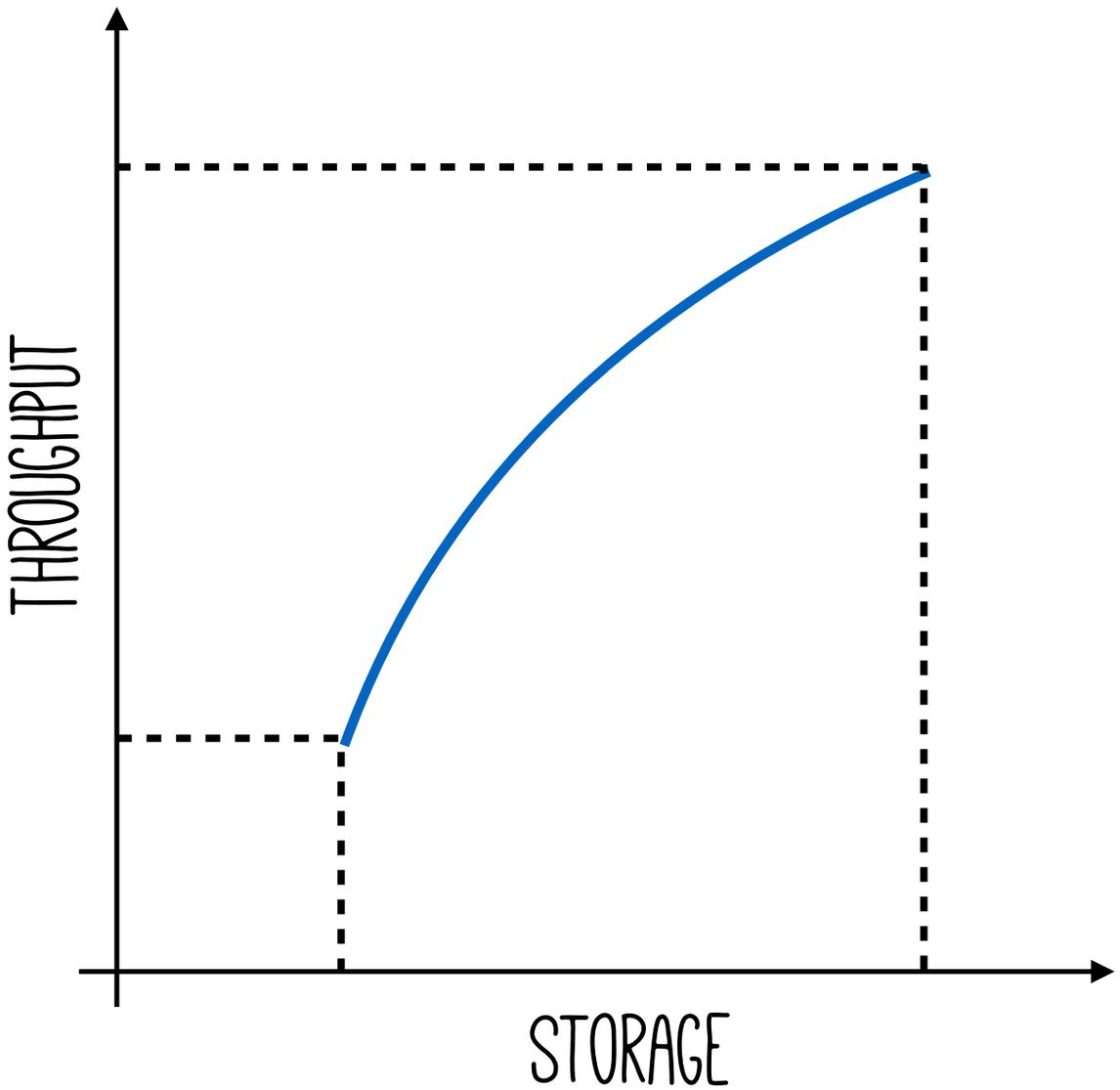


Smooth Tradeoff Curve

Dynamic Navigation

Applications in Several Classical Systems Problems


THROUGHPUT
STORAGE



Storage-Performance Tradeoff

Background

Background

Builds on Succinct [NSDI'15]

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

Step 1:
Construct set of suffixes

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

Step 1:
Construct set of suffixes

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y

Step 1:
Construct set of suffixes

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y

Step 1:
Construct set of suffixes

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y

a	p	p	y	p	u	p	p	y

Step 1:
Construct set of suffixes

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y

a	p	p	y	p	u	p	p	y

p	p	y	p	u	p	p	y

p	y	p	u	p	p	y

y	p	u	p	p	y

p	u	p	p	y

u	p	p	y

p	p	y

p	y

y

Step 1:
Construct set of suffixes

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y

a	p	p	y	p	u	p	p	y

p	p	y	p	u	p	p	y

p	y	p	u	p	p	y

y	p	u	p	p	y

p	u	p	p	y

u	p	p	y

p	p	y

p	y

y

Step 2:
Sort suffixes (lexicographically)

Array of suffixes (AoS)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y

p	p	y

Array of suffixes (AoS)

p	p	y	p	u	p	p	y

p	u	p	p	y

p	y

p	y	p	u	p	p	y

u	p	p	y

y

y	p	u	p	p	y

Suffix array (AoS2Input)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

Step 3:
Mark location of each sorted suffix (in input file)

a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y

p	p	y

p	p	y	p	u	p	p	y

p	u	p	p	y

p	y

p	y	p	u	p	p	y

u	p	p	y

y

y	p	u	p	p	y

Suffix array (AoS2Input)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

Step 3:
Mark location of each sorted suffix (in input file)

a	p	p	y	p	u	p	p	y
h	a	p	p	y	p	u	p	p	y
p	p	y						
p	p	y	p	u	p	p	y	
p	u	p	p	y				
p	y							
p	y	p	u	p	p	y		
u	p	p	y					
y								
y	p	u	p	p	y			

Suffix array (AoS2Input)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1	a p p y p u p p y
	h a p p y p u p p y
	p p y
	p p y p u p p y
	p u p p y
	p y
	p y p u p p y
	u p p y
	y
	y p u p p y

Step 3:
Mark location of each sorted suffix (in input file)

Suffix array (AoS2Input)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1	a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y

p	p	y

p	p	y	p	u	p	p	y

p	u	p	p	y

p	y

p	y	p	u	p	p	y

u	p	p	y

y

y	p	u	p	p	y

Step 3:
Mark location of each sorted suffix (in input file)

Suffix array (AoS2Input)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1	a p p y p u p p y
0	h a p p y p u p p y
	p p y
	p p y p u p p y
	p u p p y
	p y
	p y p u p p y
	u p p y
	y
	y p u p p y

Step 3:
Mark location of each sorted suffix (in input file)

Suffix array (AoS2Input)

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

AoS2Input

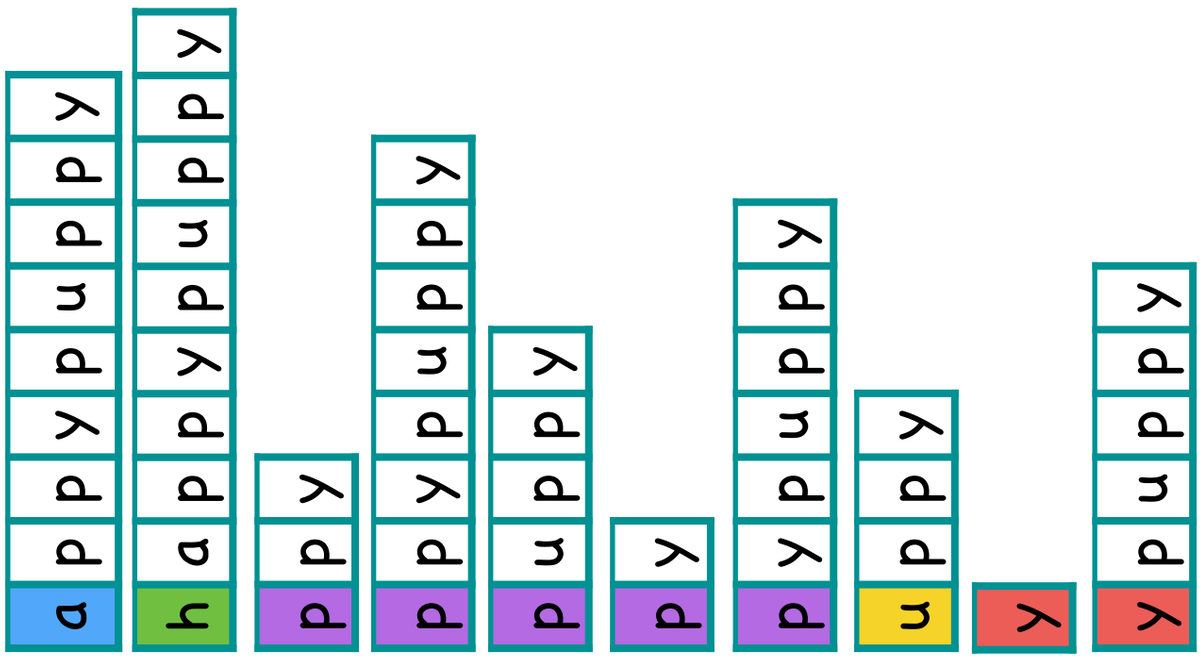
1	a p p y p u p p y
0	h a p p y p u p p y
7	p p y
2	p p y p u p p y
5	p u p p y
8	p y
3	p y p u p p y
6	u p p y
9	y
4	y p u p p y

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

Succinct Data Representation

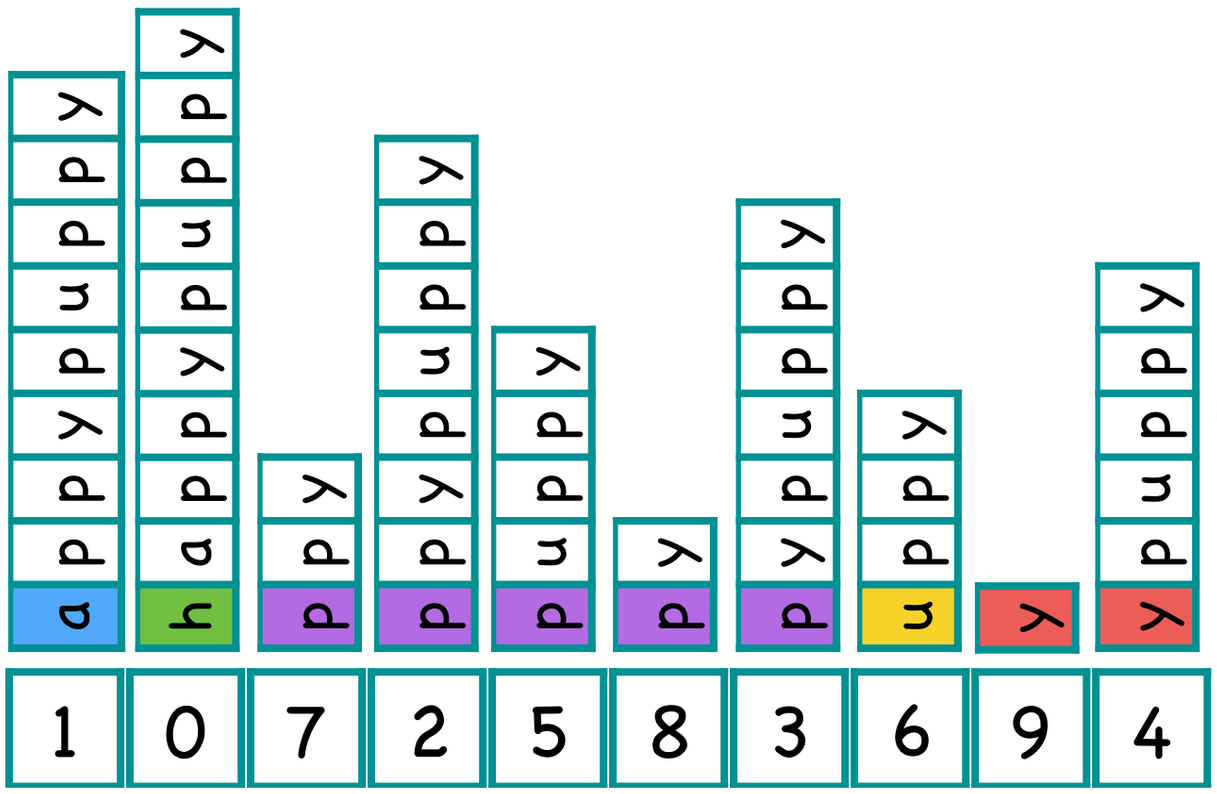
0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



Array of Suffixes (AoS)
(suffixes in sorted order)

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



Array of Suffixes (AoS)

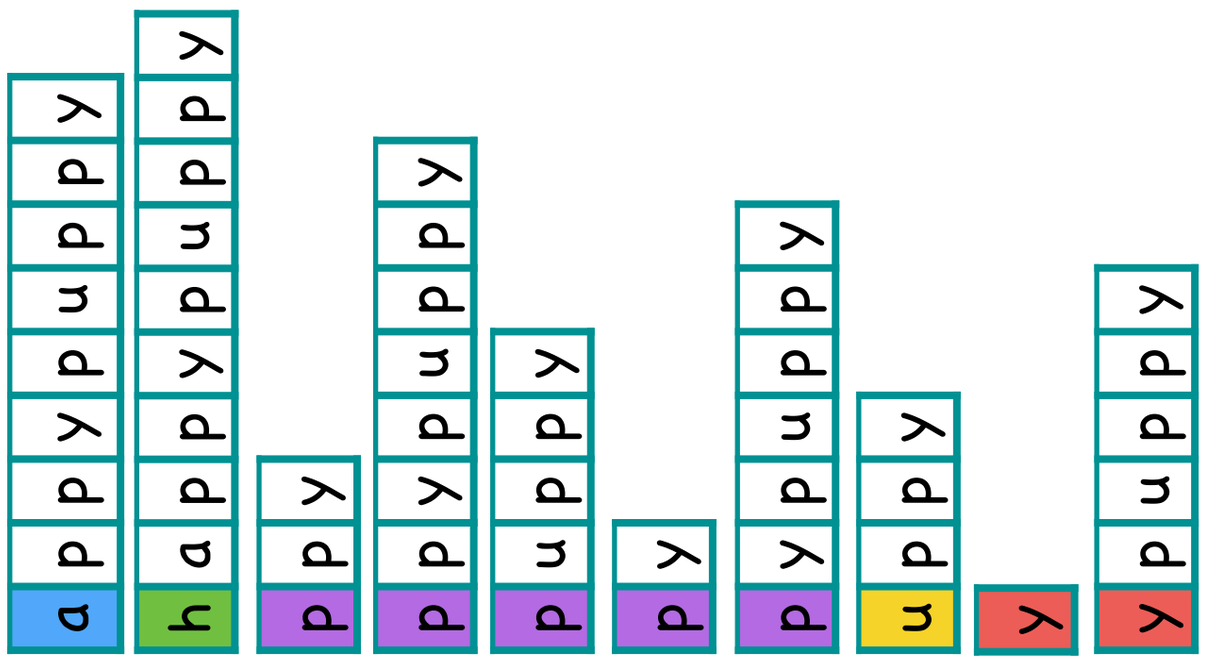
(suffixes in sorted order)

AoS2Input

(location of suffix in file)

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



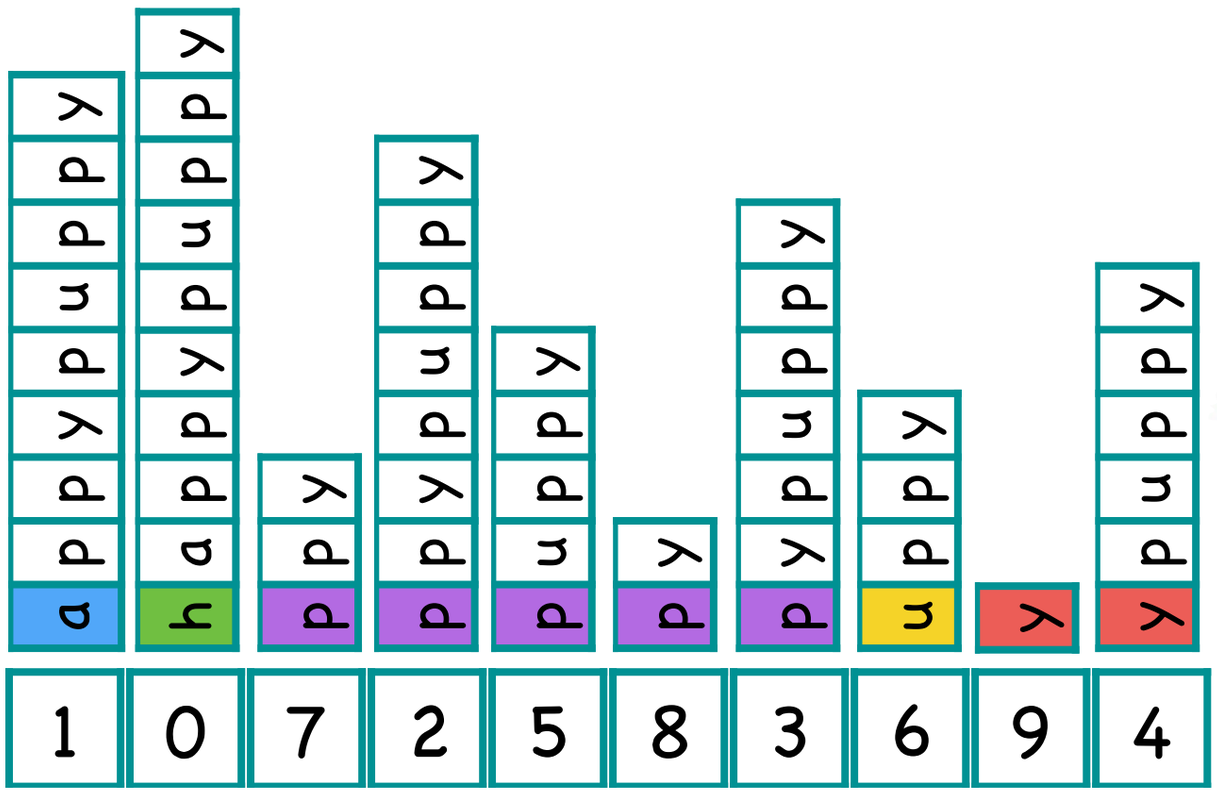
1	0	7	2	5	8	3	6	9	4

Arbitrary substring search

via binary search

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



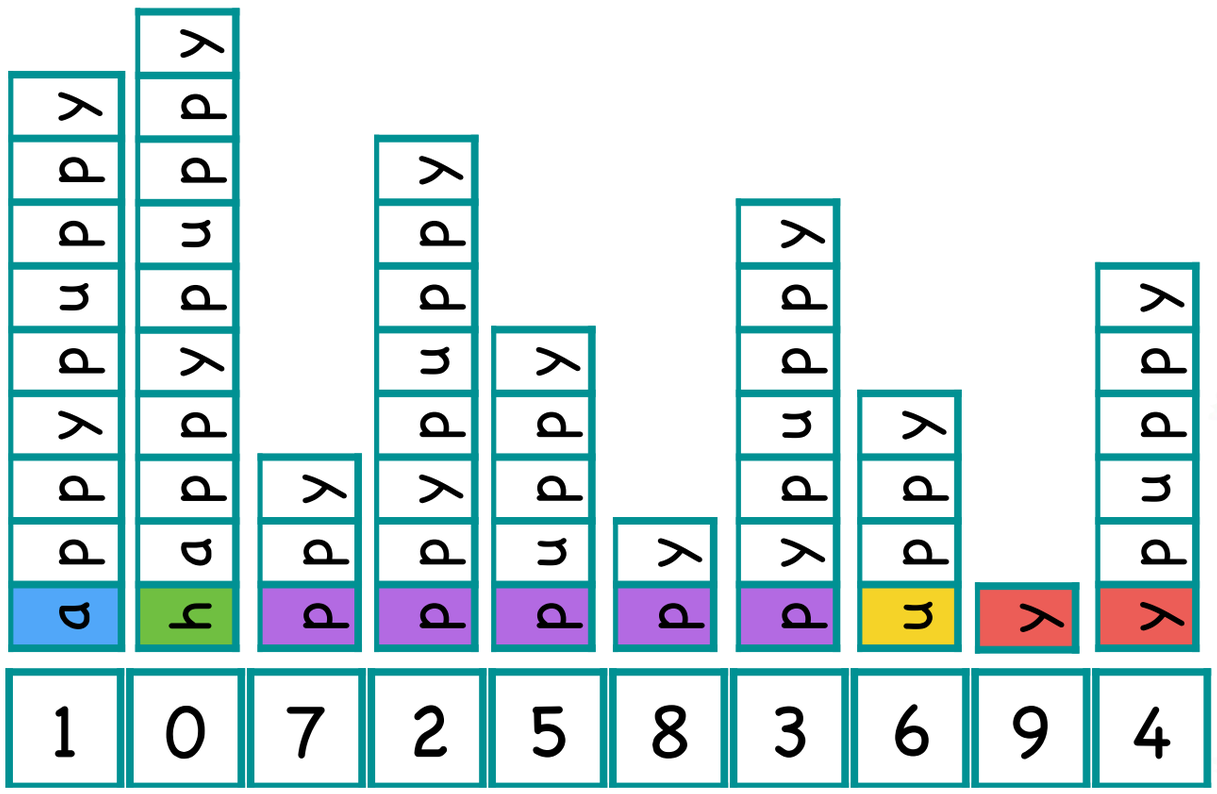
1	0	7	2	5	8	3	6	9	4

Problem1: Size of AoS

O(n*n) bits

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



Problem1: Size of AoS

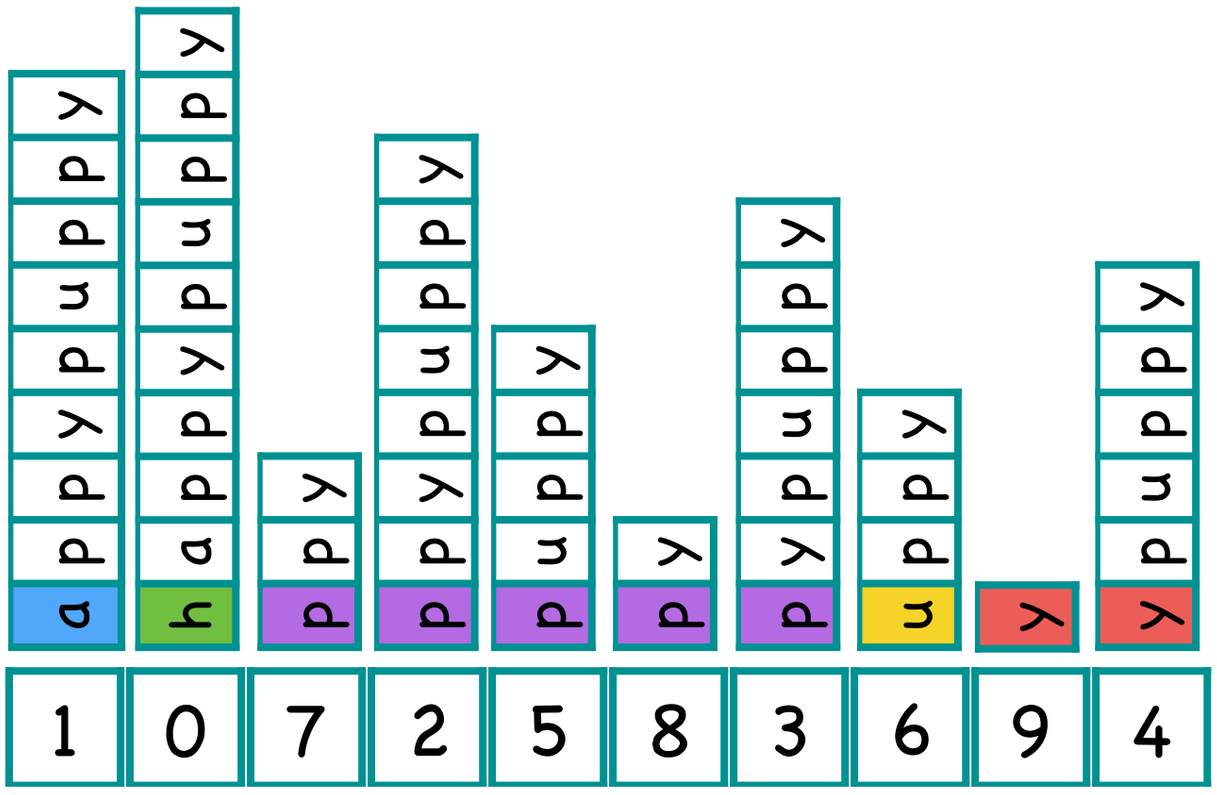
O(n*n) bits

Problem2: Size of AoS2Input

each entry takes log(n) bits

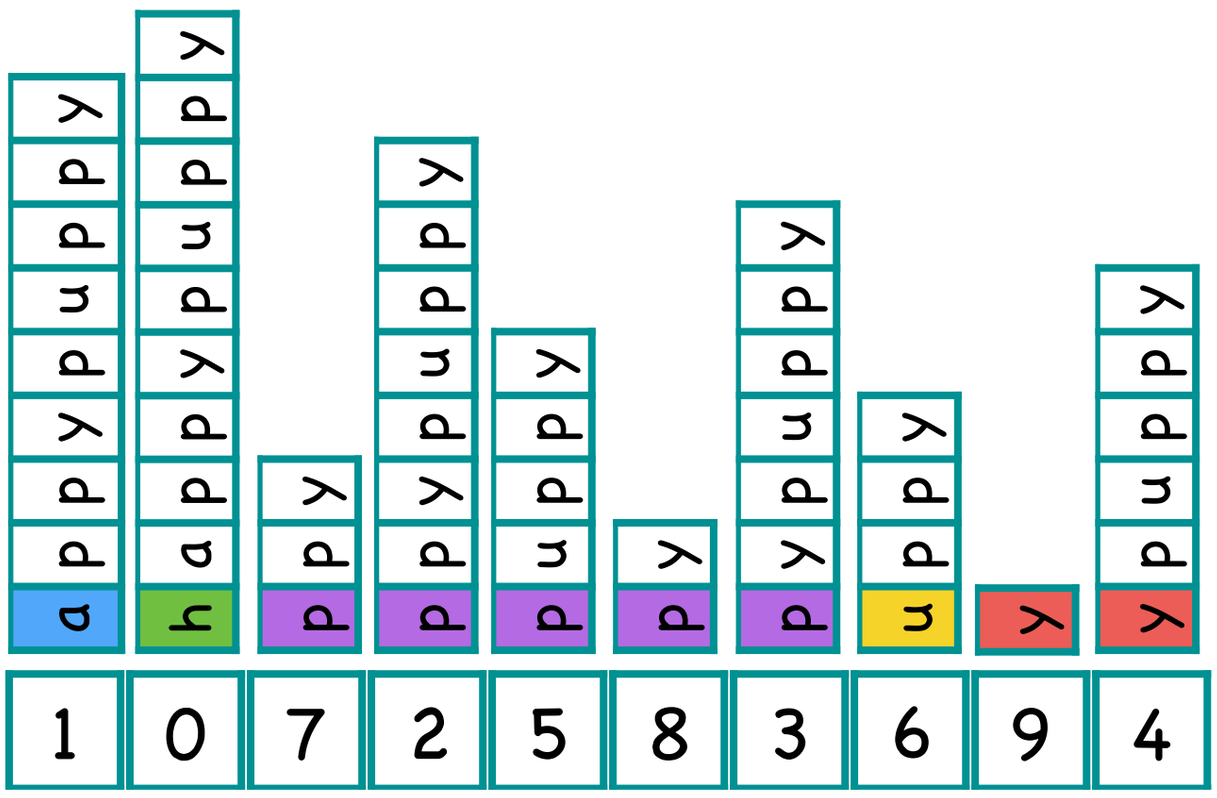
Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



Succinct Data Representation

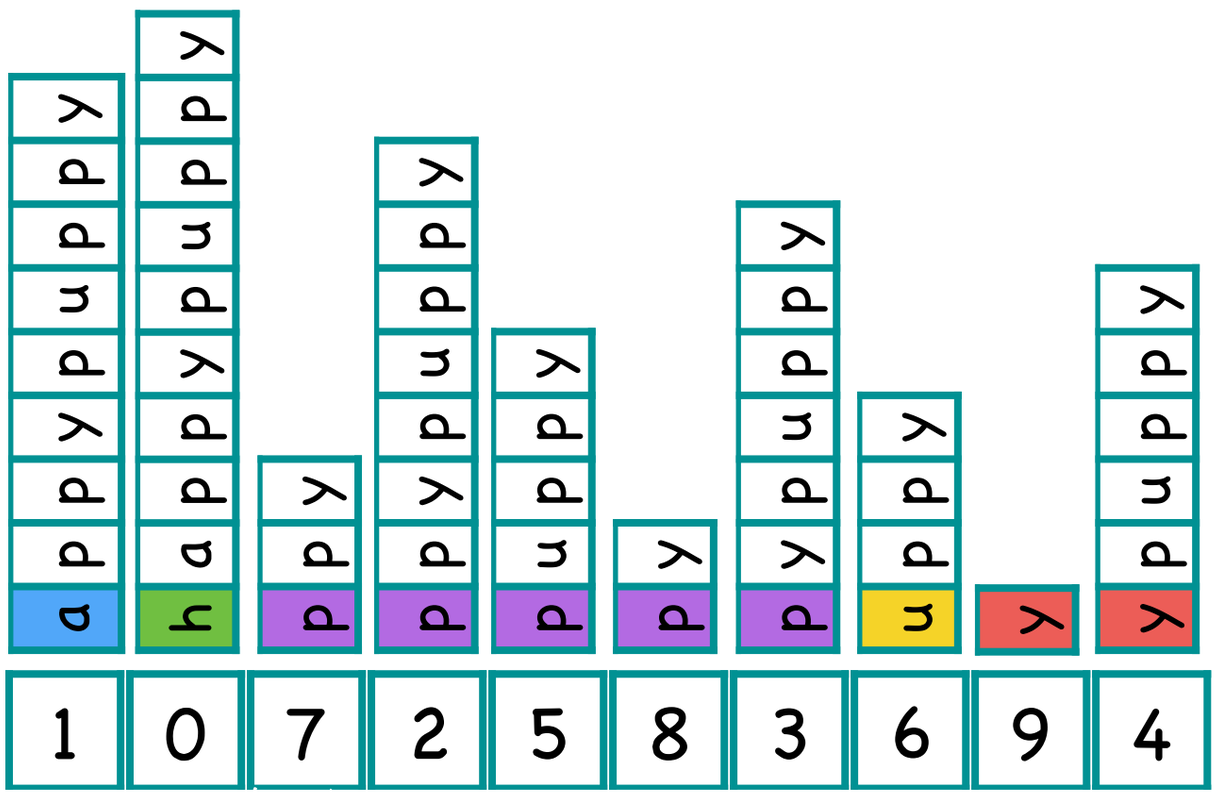
0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



0

Succinct Data Representation

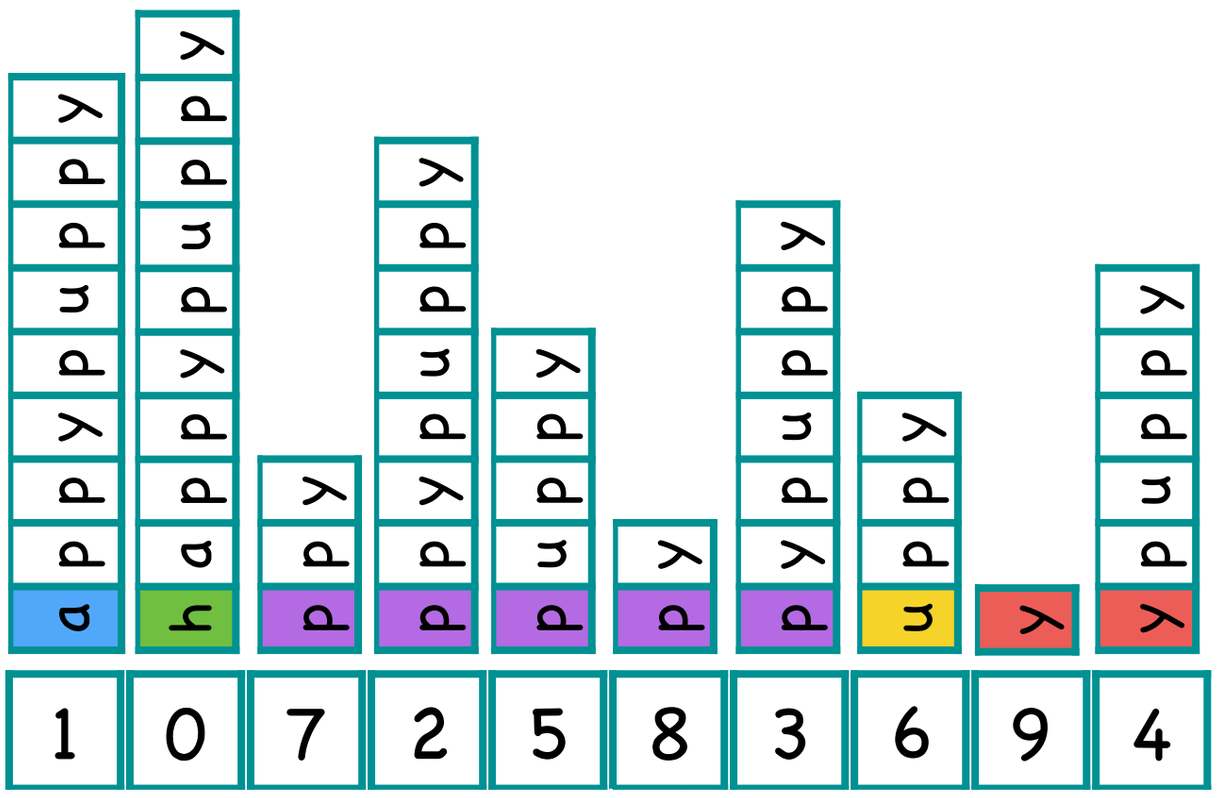
0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



3	0

Succinct Data Representation

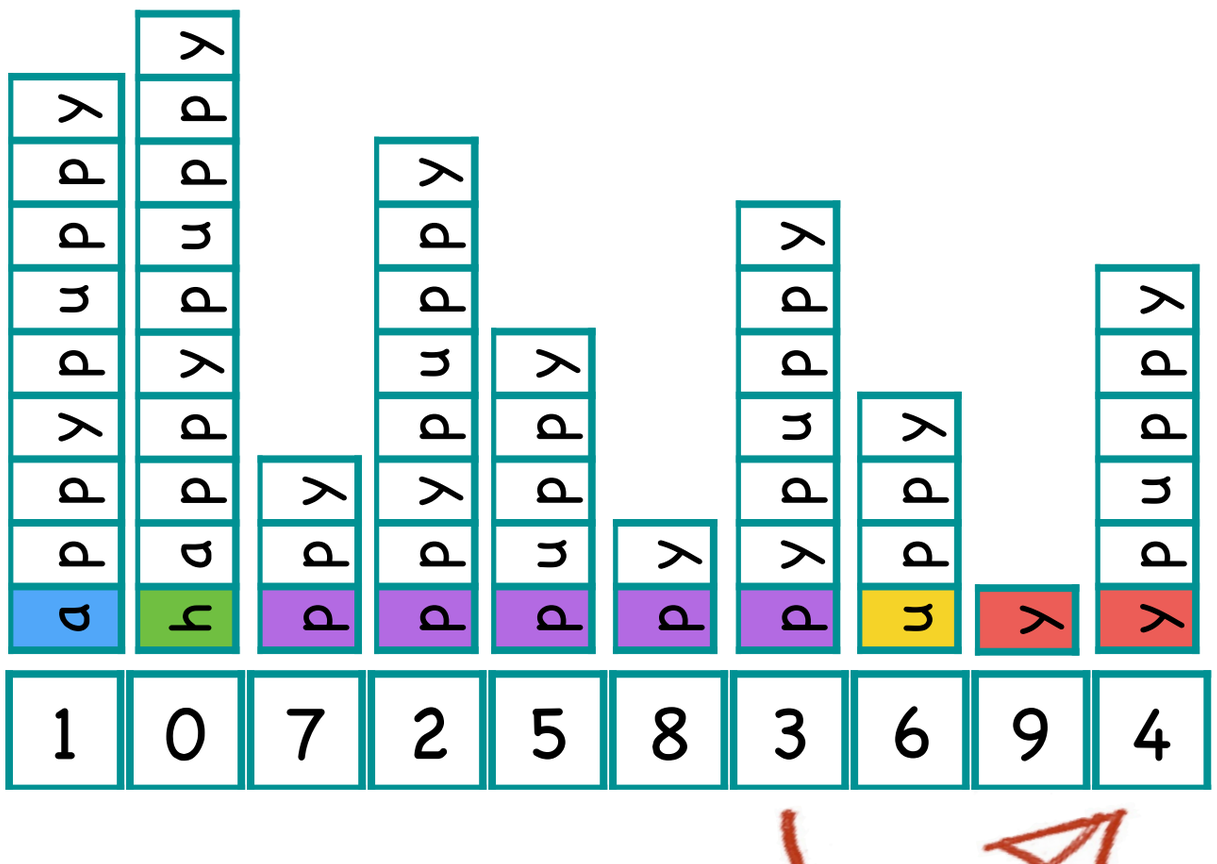
0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



3	0		6

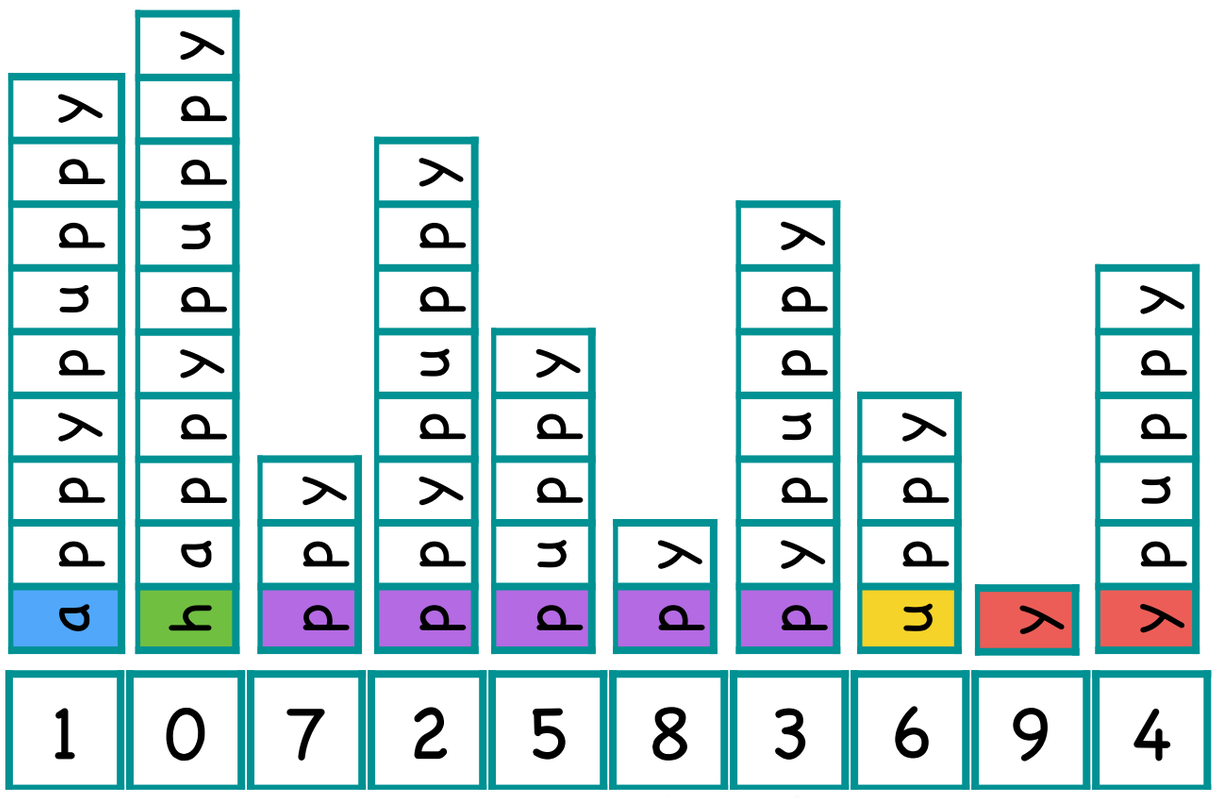
Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



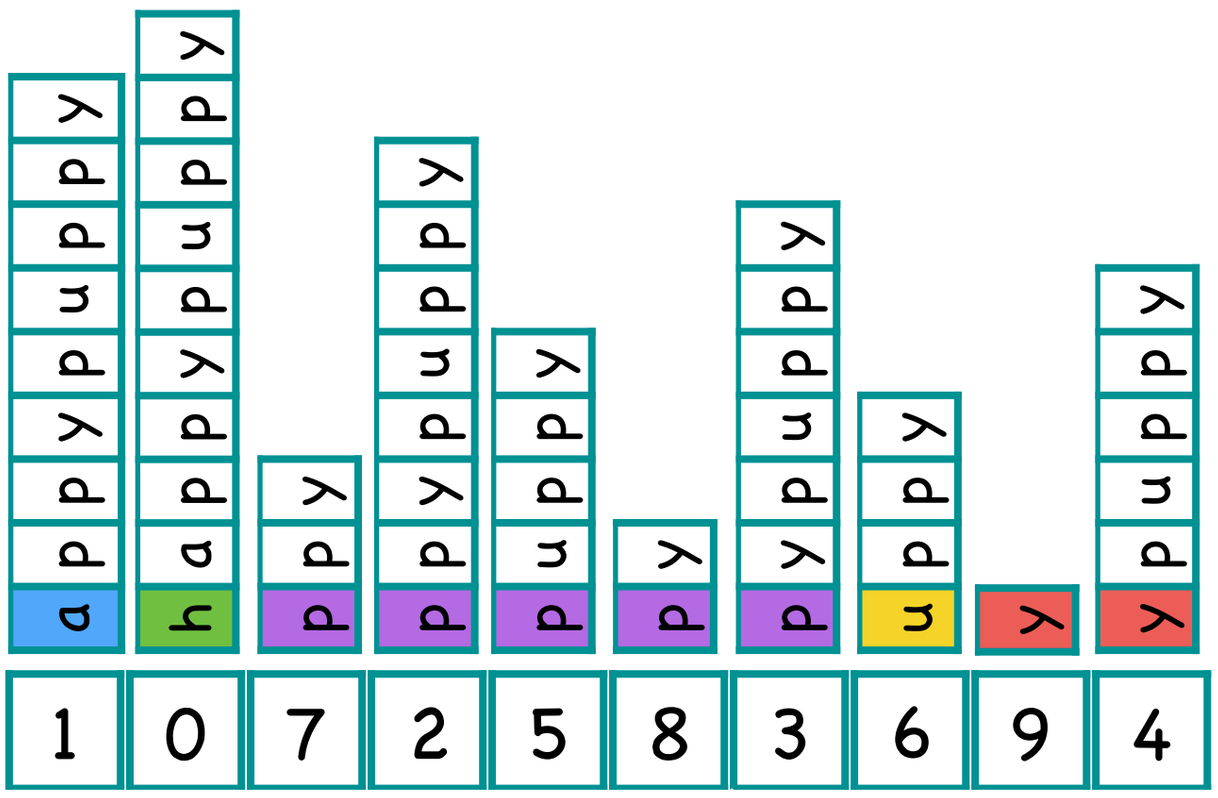
Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



Succinct Data Representation

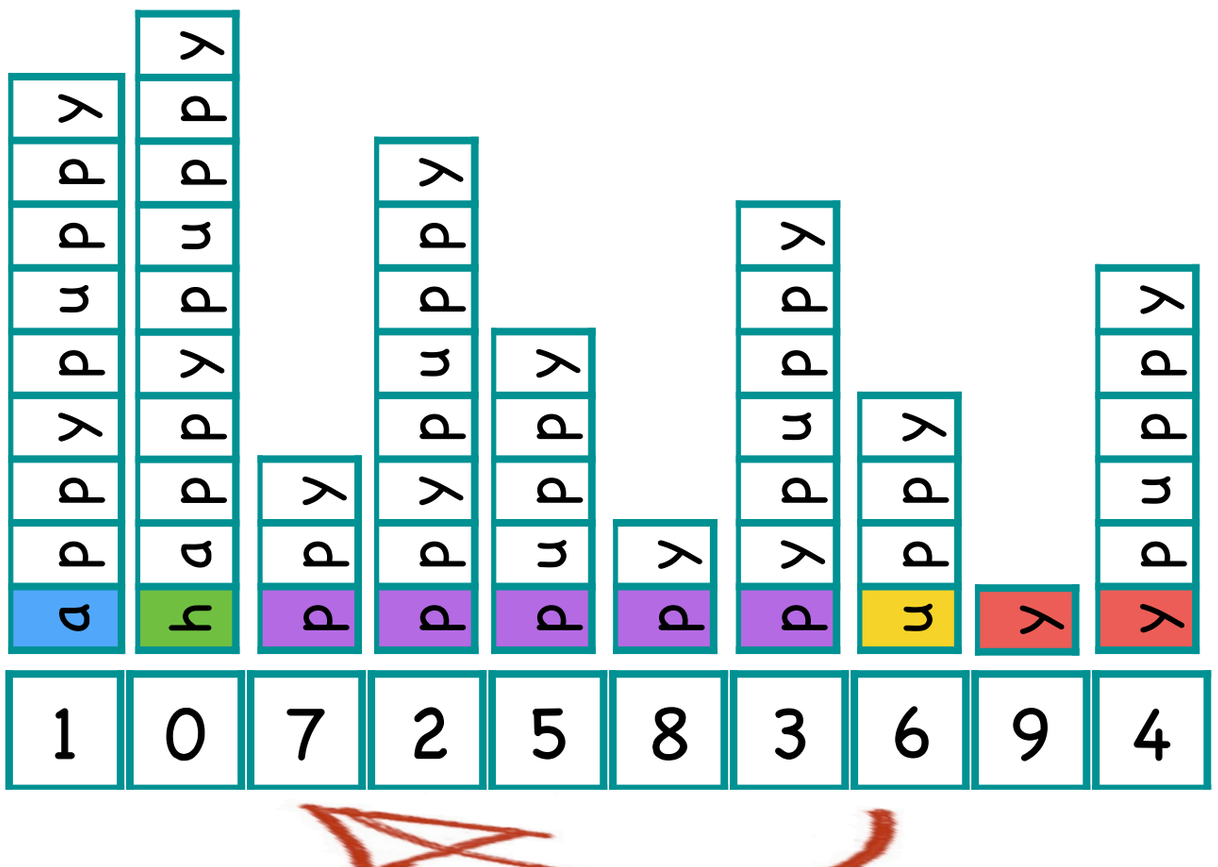
0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



3	0		6	7		9			4

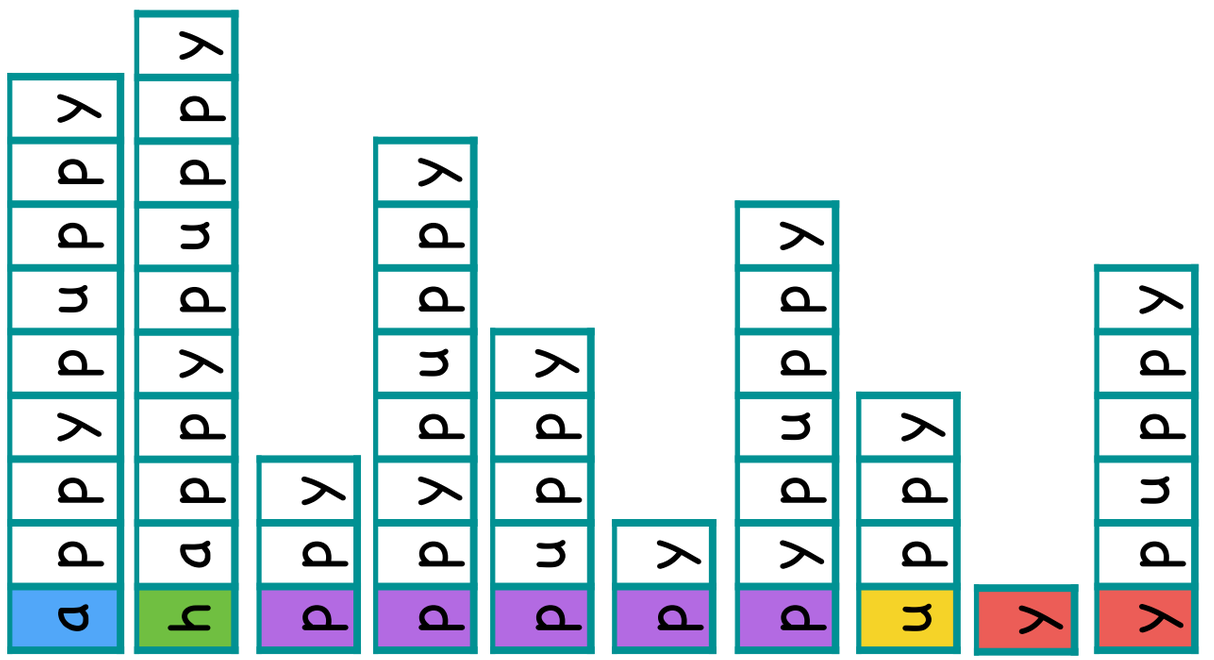
Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

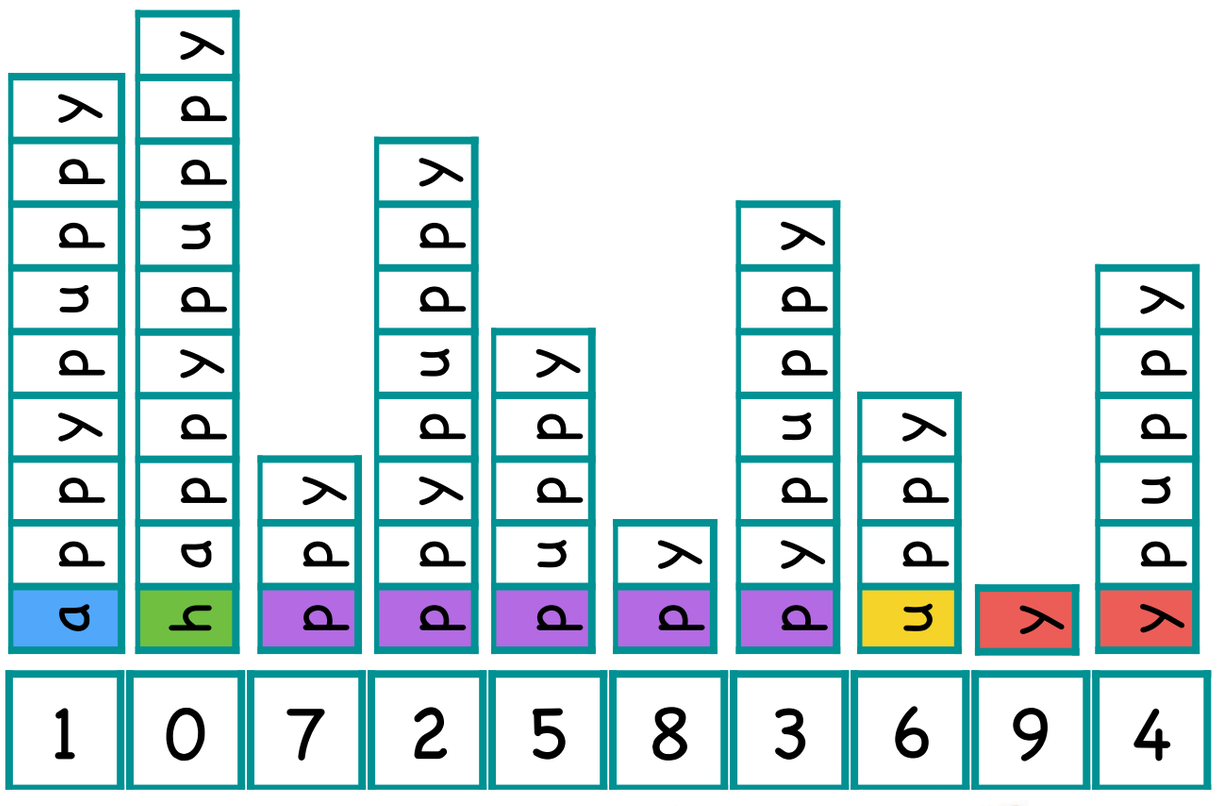


1	0	7	2	5	8	3	6	9	4

3	0	5	6	7	9	2	4

Succinct Data Representation

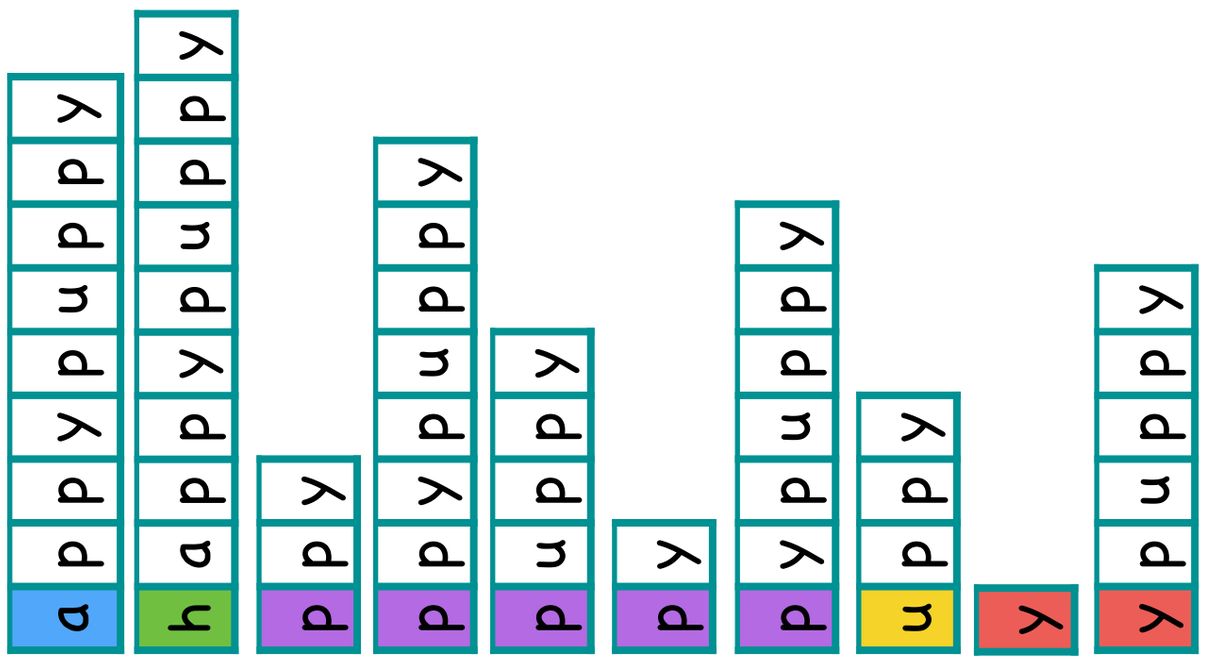
0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



3	0	5	6	7	8	9	2		4

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

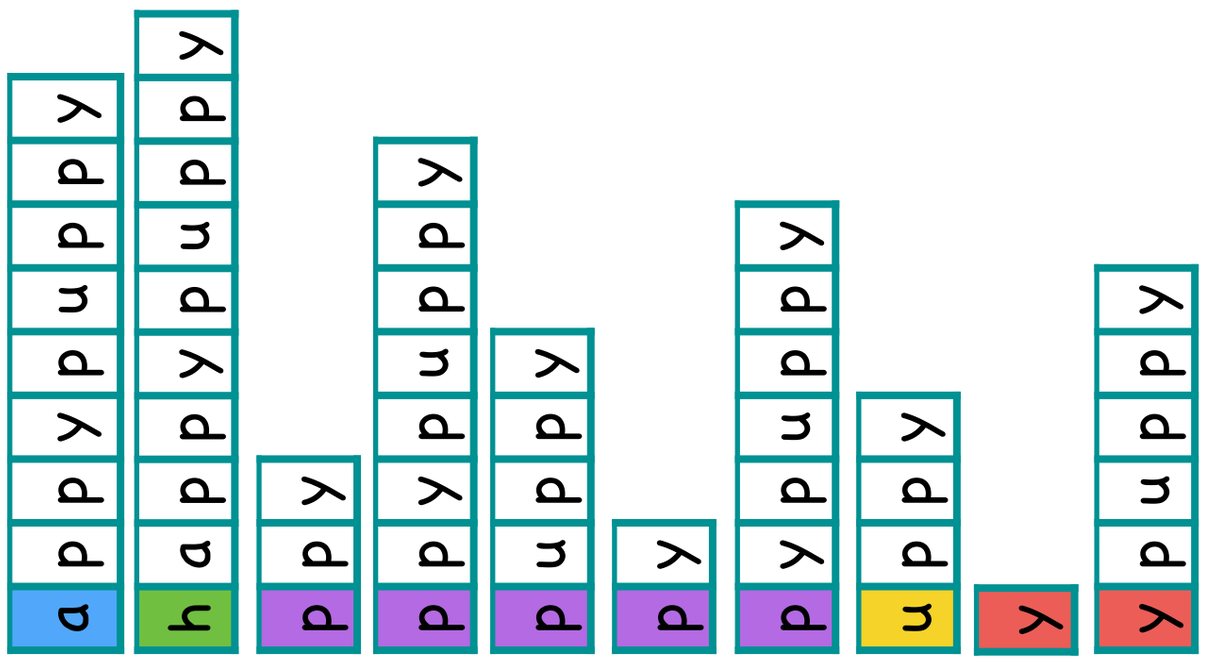


1	0	7	2	5	8	3	6	9	4

3	0	5	6	7	8	9	2	1	4

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

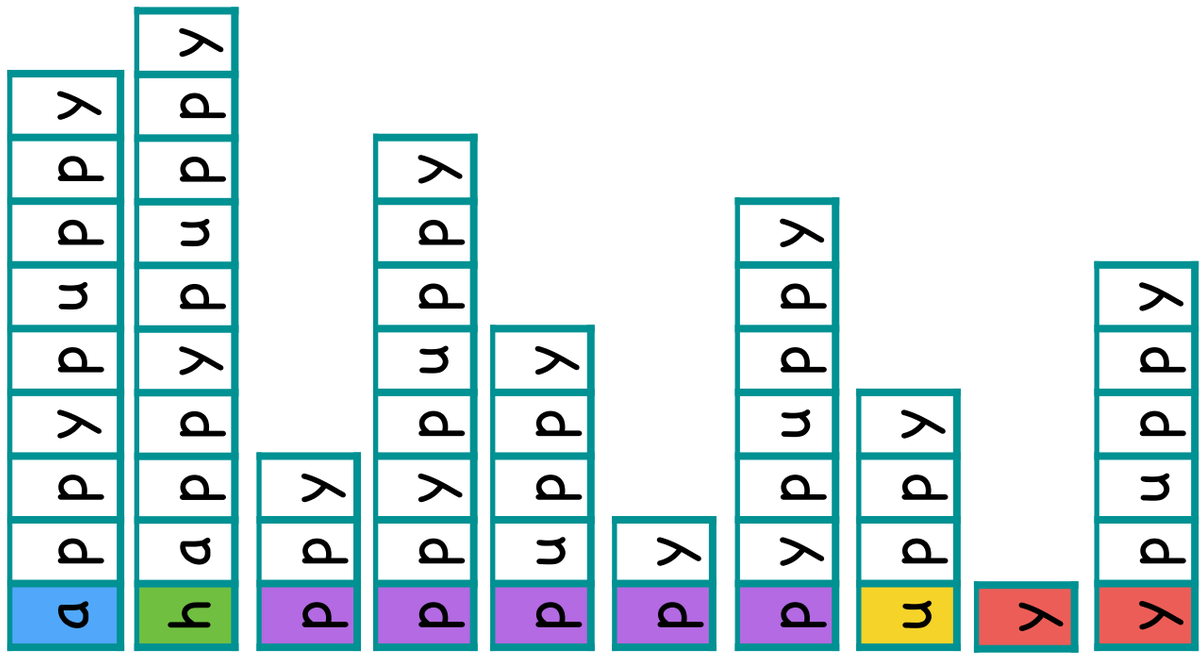


1	0	7	2	5	8	3	6	9	4

3	0	5	6	7	8	9	2	1	4

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



1	0	7	2	5	8	3	6	9	4

NextCharIdx

(index of suffix after removing the first character)

3	0	5	6	7	8	9	2	1	4

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

h	a	p	p	y	p	u	p	p	y
1	0	7	2	5	8	3	6	9	4

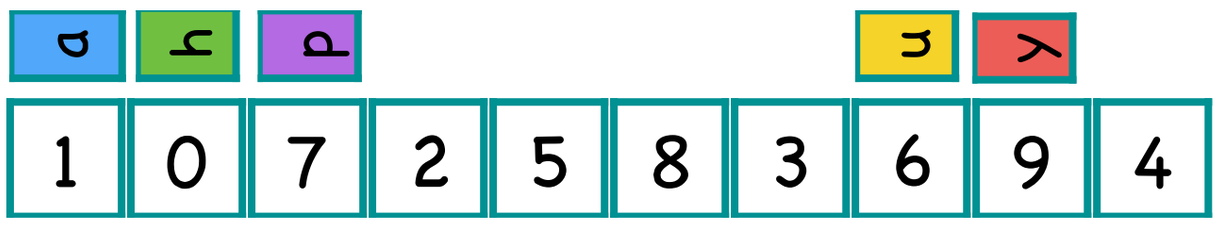
3	0	5	6	7	8	9	2	1	4

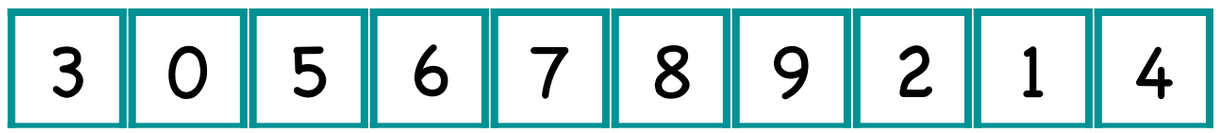
NextCharIdx

(index of suffix after removing the first character)

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y





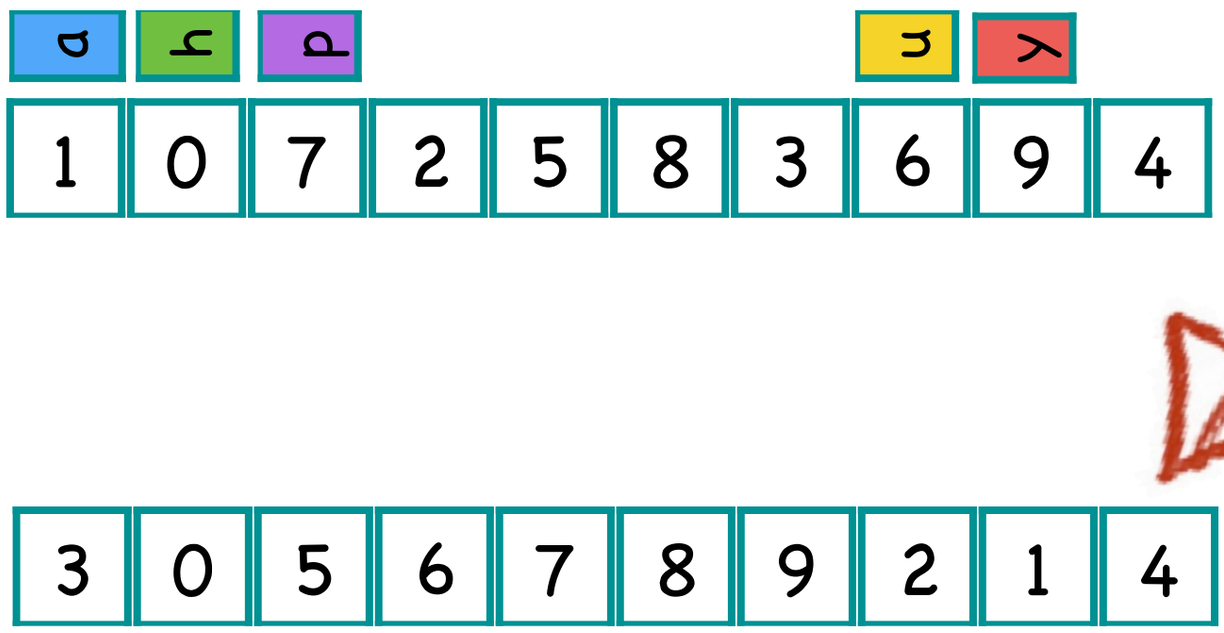
NextCharIdx

(index of suffix after removing the first character)

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

No redundancy in AoS2Input

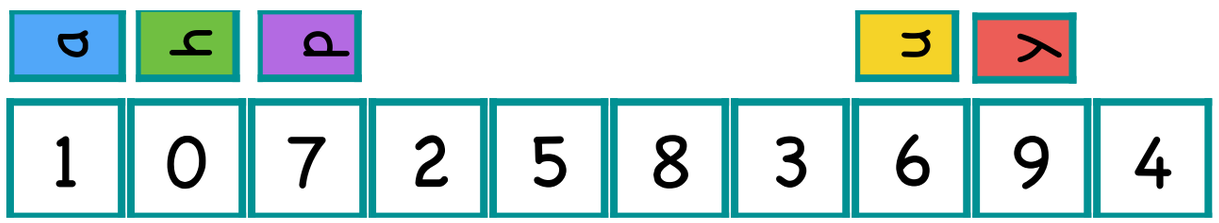


NextCharIdx

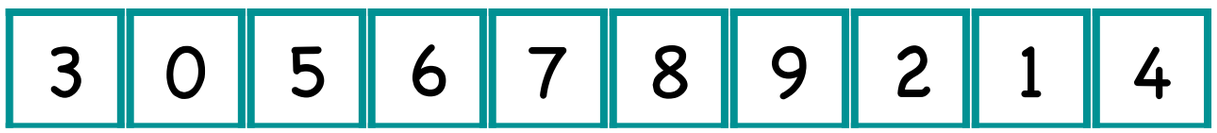
(index of suffix after removing the first character)

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



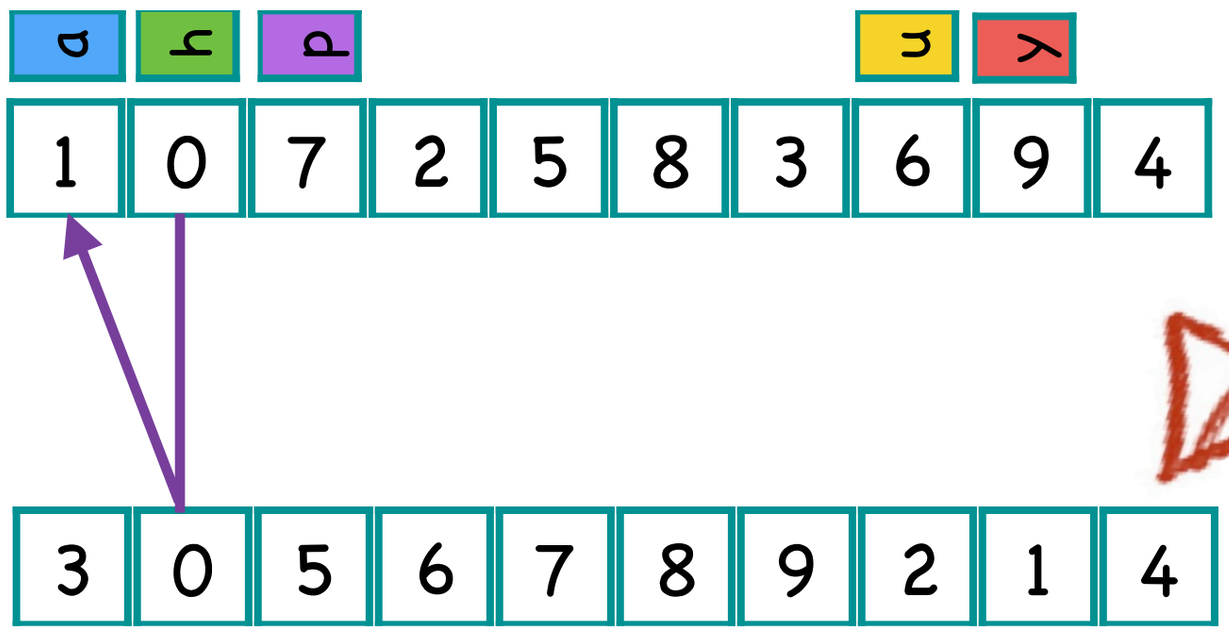
NextCharIdx
(index of next larger value in AoS2Input)



NextCharIdx
(index of suffix after removing the first character)

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

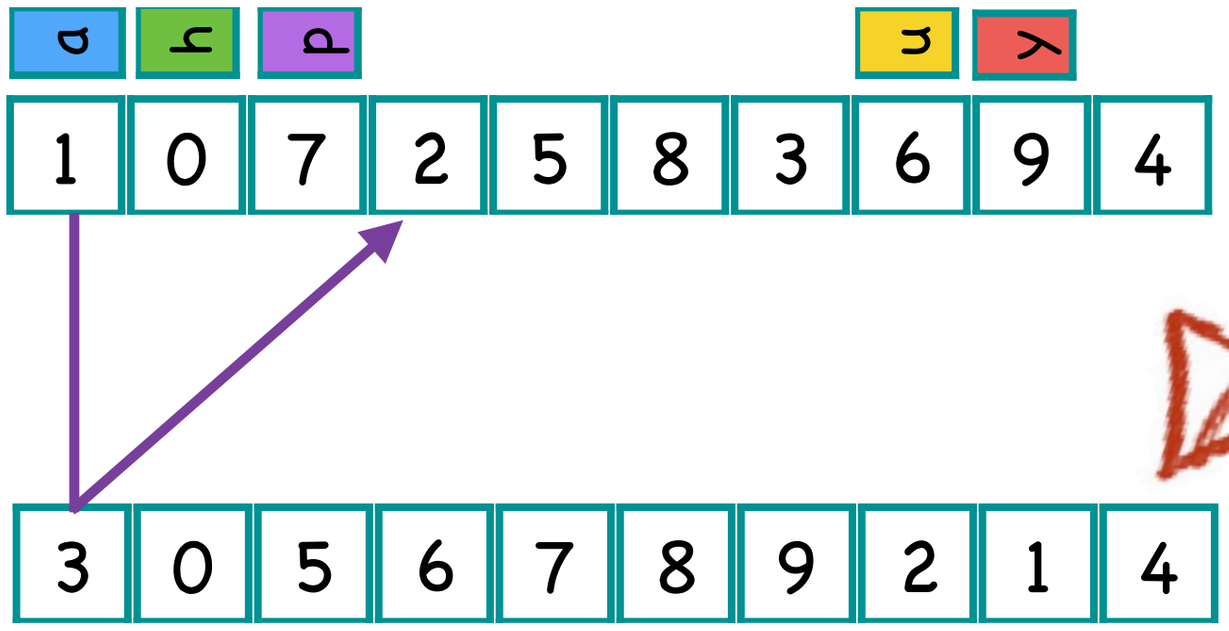


NextCharIdx
(index of next larger value in AoS2Input)

NextCharIdx
(index of suffix after removing the first character)

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y



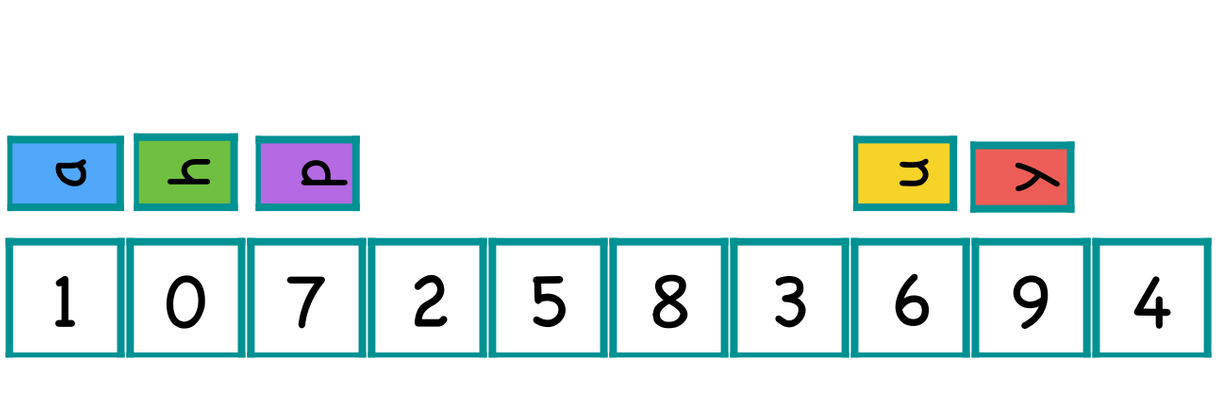
NextCharIdx
(index of next larger value in AoS2Input)

NextCharIdx
(index of suffix after removing the first character)

Succinct Data Representation

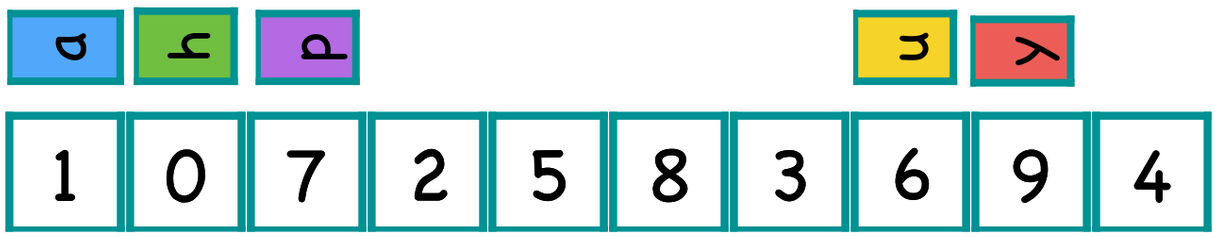
0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

No redundancy in AoS2Input



Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

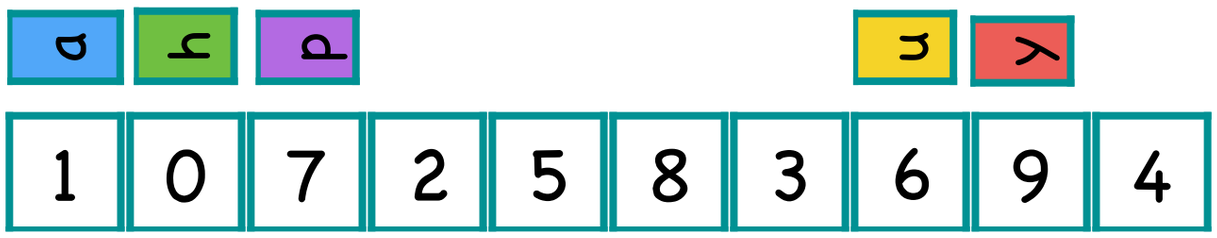


Solution: Sample!

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

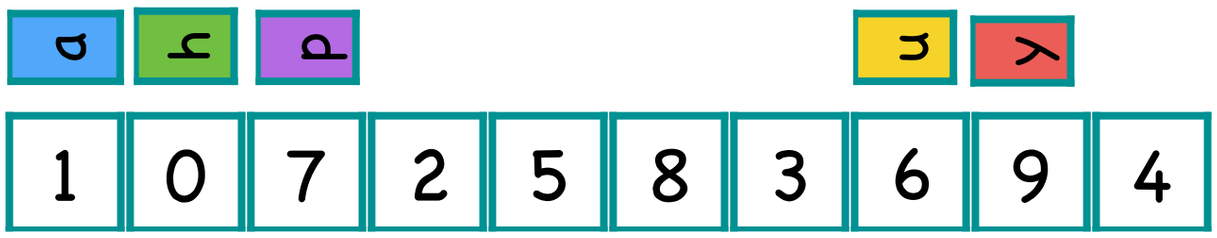
1		7		5		3		9	



Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1		7		5		3		9	



Problem: How to find unsampled values?

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1		7		5		3		9	

a	h	p	y	x

1	0	7	2	5	8	3	6	9	4

Problem: How to find unsampled values?

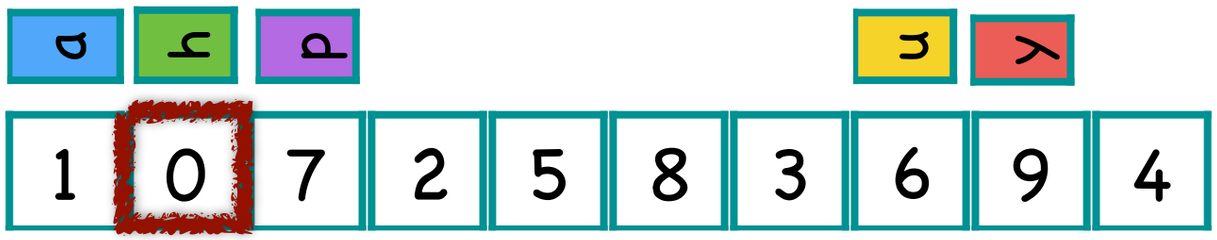
3	0	5	6	7	8	9	2	1	4

Solution:Follow NextCharIdx pointers until you hit a sampled value

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1		7		5		3		9	



Problem: How to find unsampled values?

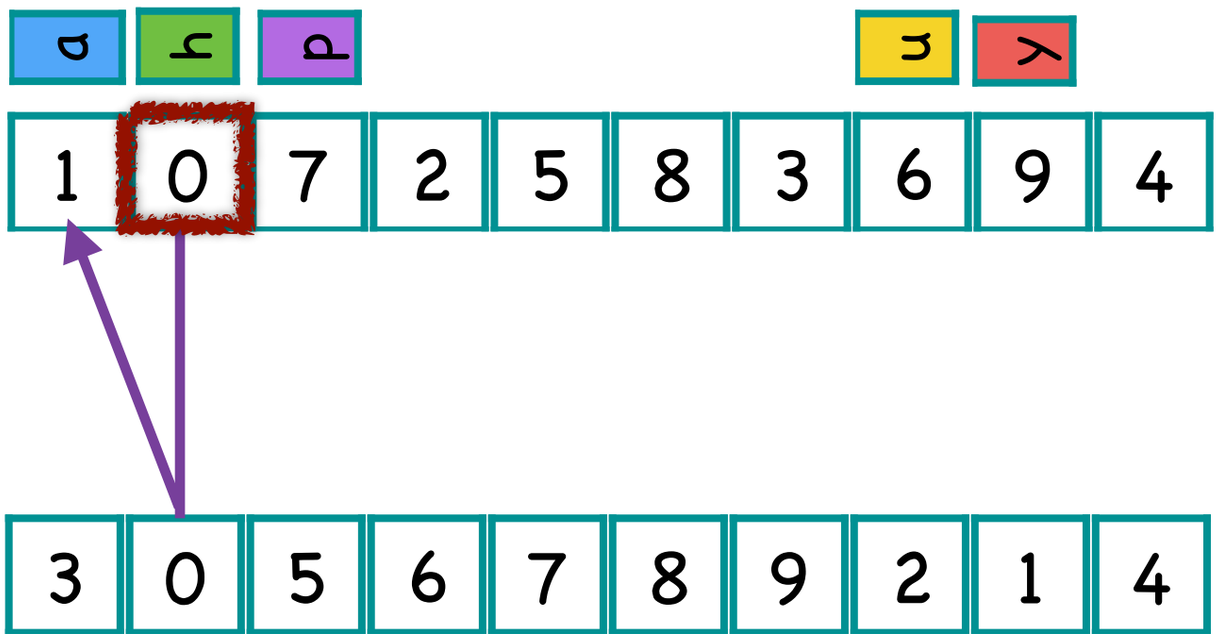
3	0	5	6	7	8	9	2	1	4

Solution:Follow NextCharIdx pointers until you hit a sampled value

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1		7		5		3		9	



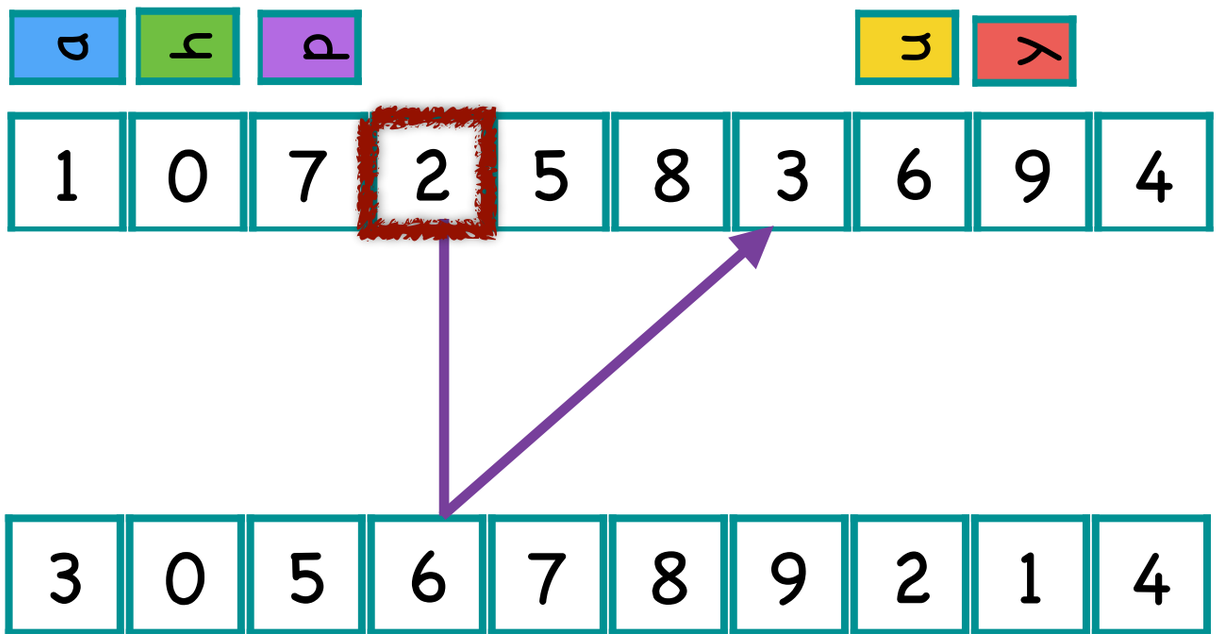
Problem: How to find unsampled values?

Solution:Follow NextCharIdx pointers until you hit a sampled value

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1		7		5		3		9	



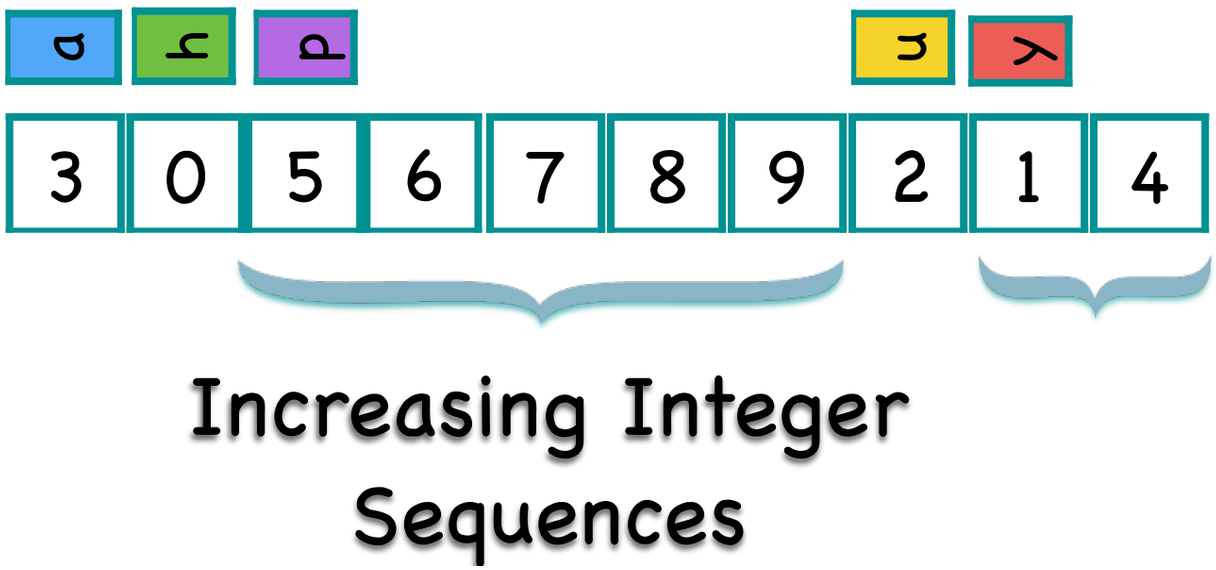
Problem: How to find unsampled values?

Solution:Follow NextCharIdx pointers until you hit a sampled value

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1		7		5		3		9	

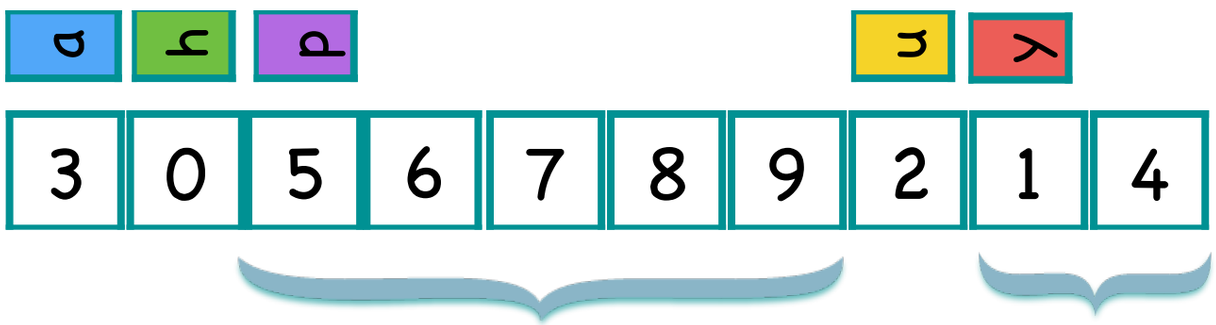


Increasing Integer Sequences

Succinct Data Representation

0	1	2	3	4	5	6	7	8	9
h	a	p	p	y	p	u	p	p	y

1		7		5		3		9	

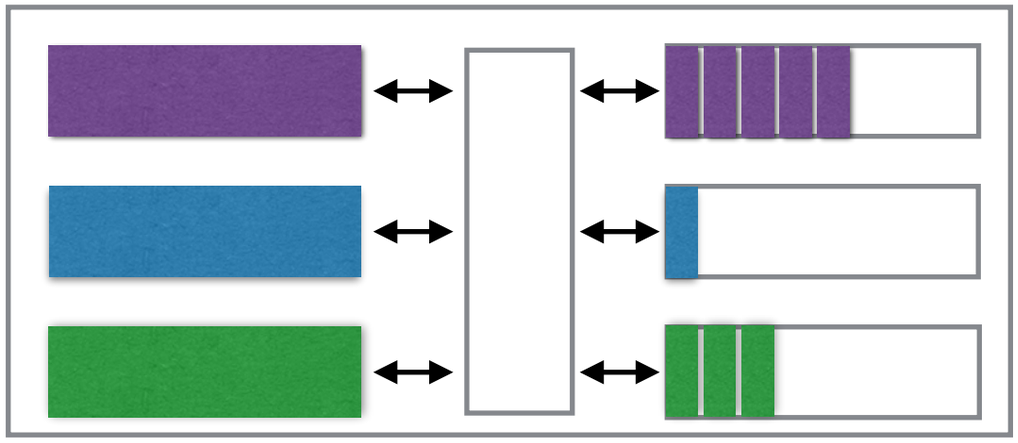


Increasing Integer Sequences

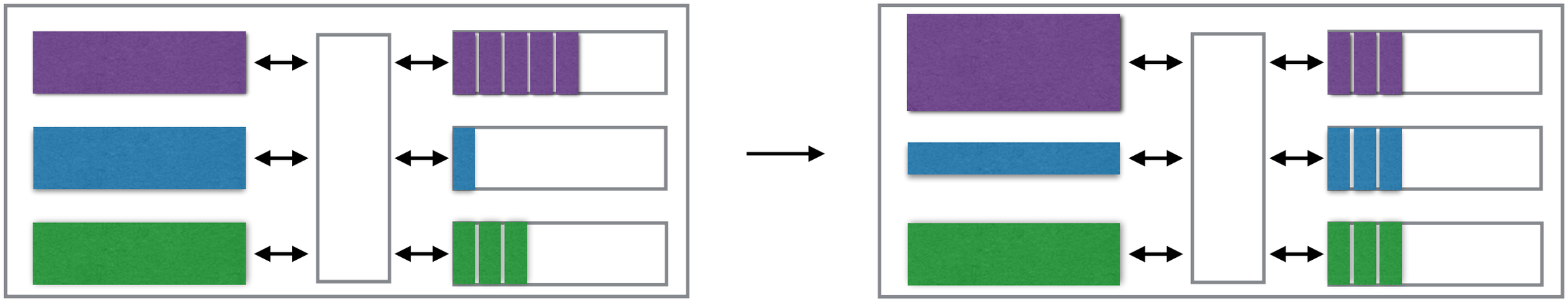
Can be compressed!

Technical **Details**

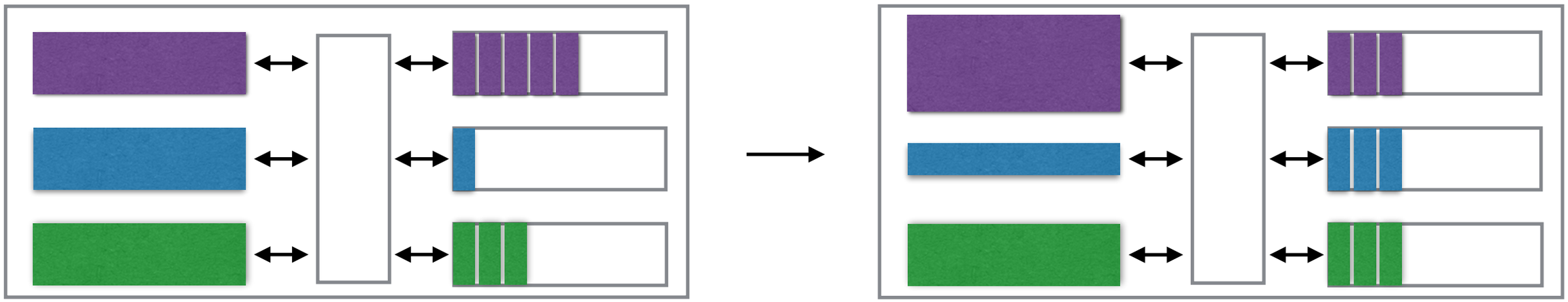
Technical **Details**



Technical **Details**

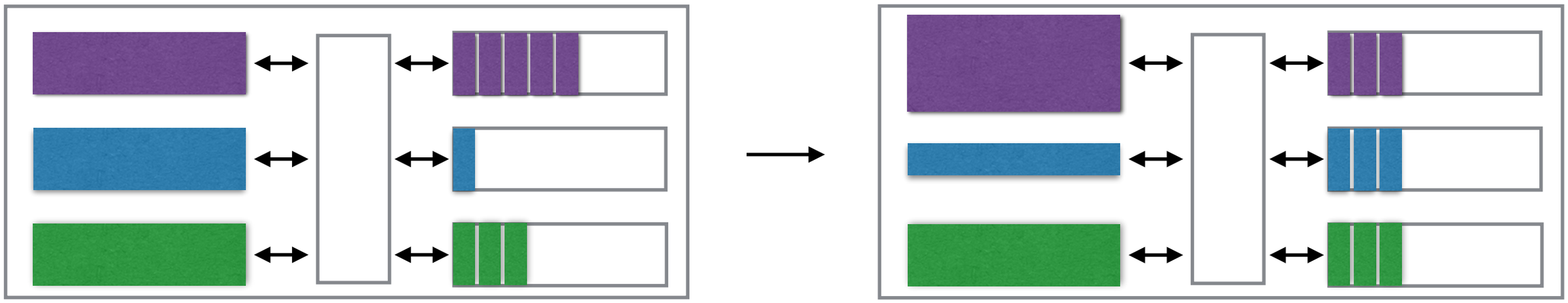


Technical **Details**



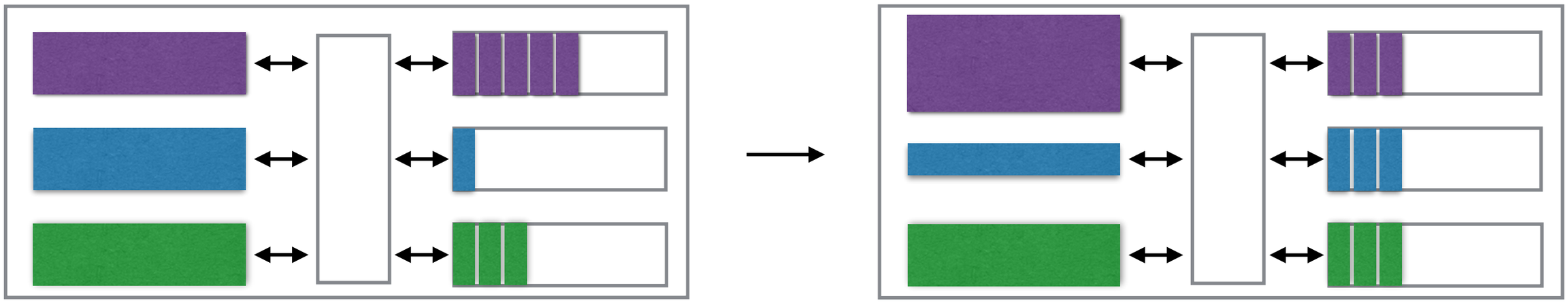
- How should partitions share cache on a server?

Technical **Details**



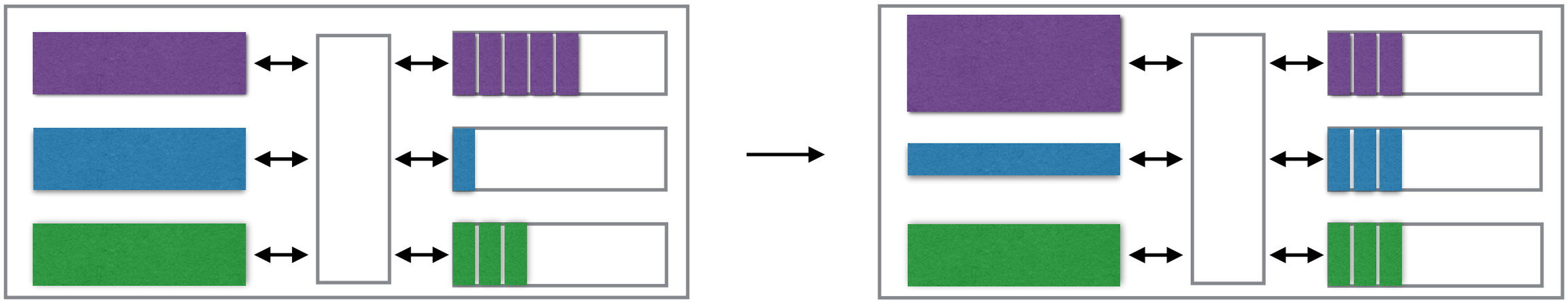
- How should partitions share cache on a server?
- How should partitions share cache across servers?

Technical **Details**



- How should partitions share cache on a server?
- How should partitions share cache across servers?
- How should requests be scheduled across replicas?

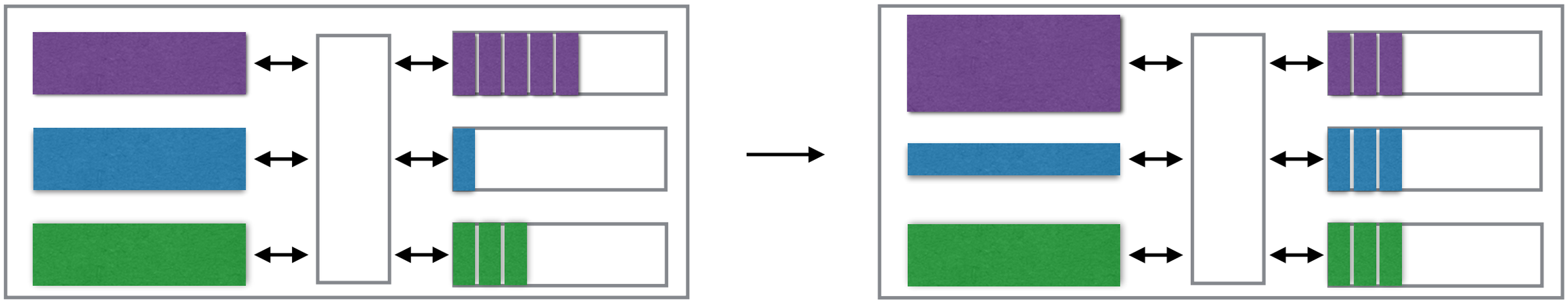
Technical **Details**



- How should partitions share cache on a server?
- How should partitions share cache across servers?
- How should requests be scheduled across replicas?

Unified Solution: Back-pressure style scheduling

Technical **Details**

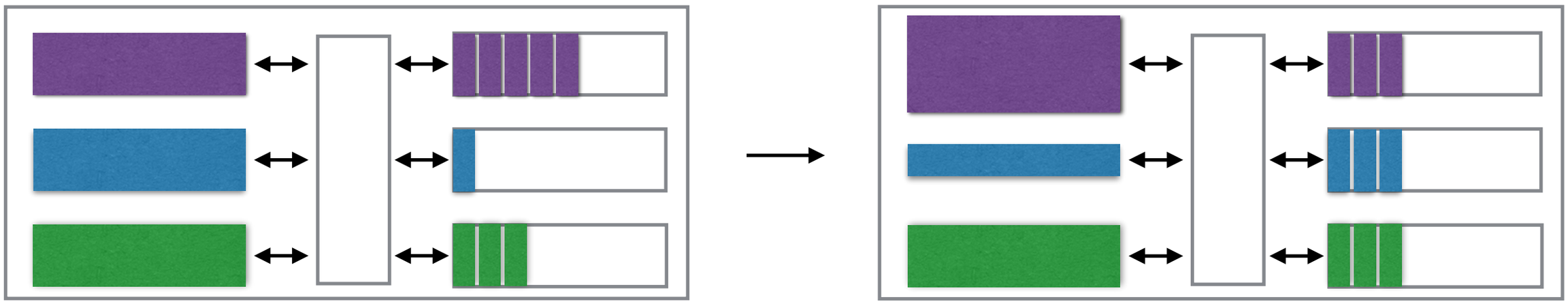


- How should partitions share cache on a server?
- How should partitions share cache across servers?
- How should requests be scheduled across replicas?

Unified Solution: Back-pressure style scheduling

Cache proportional to load,

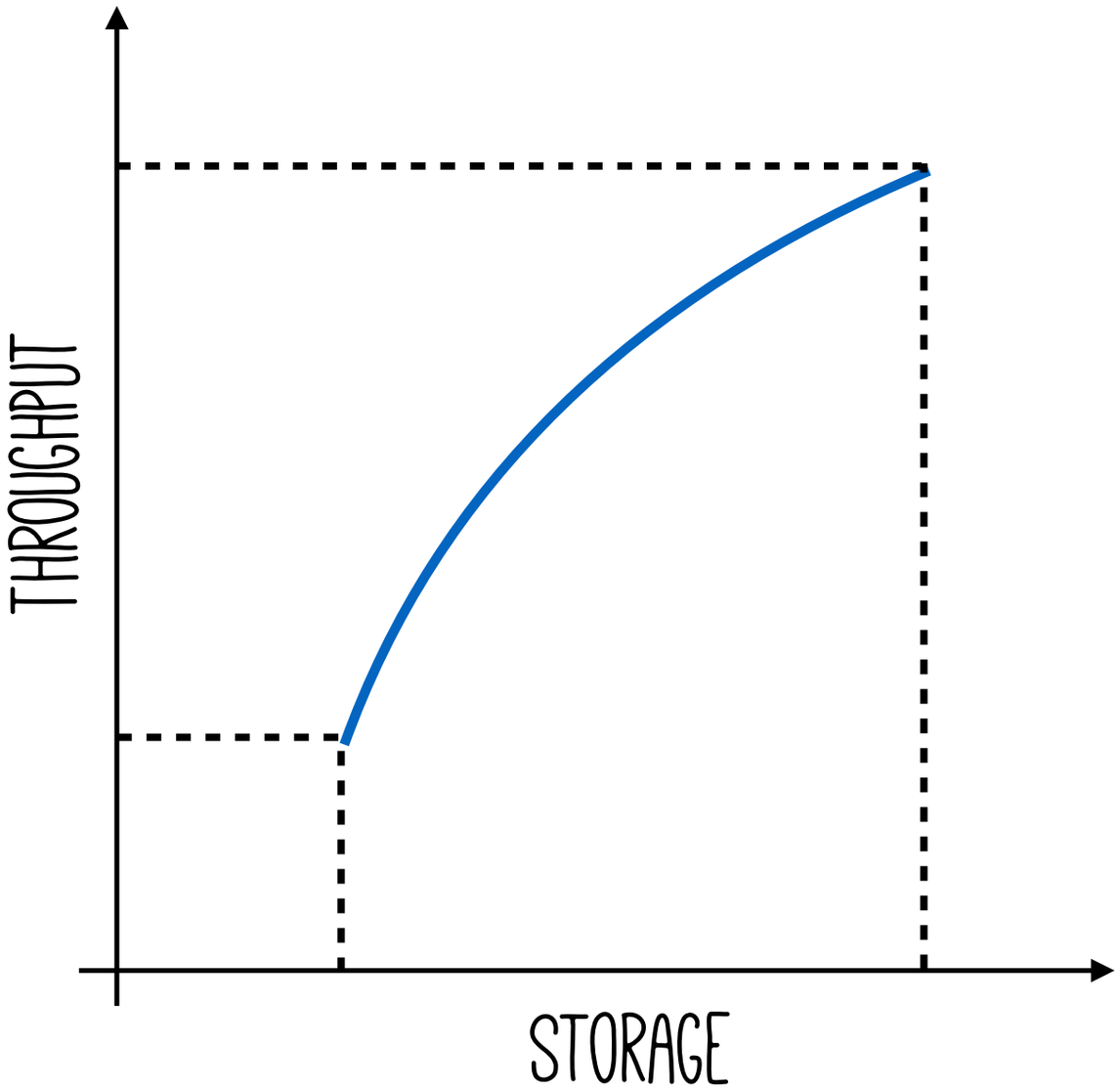
Technical **Details**

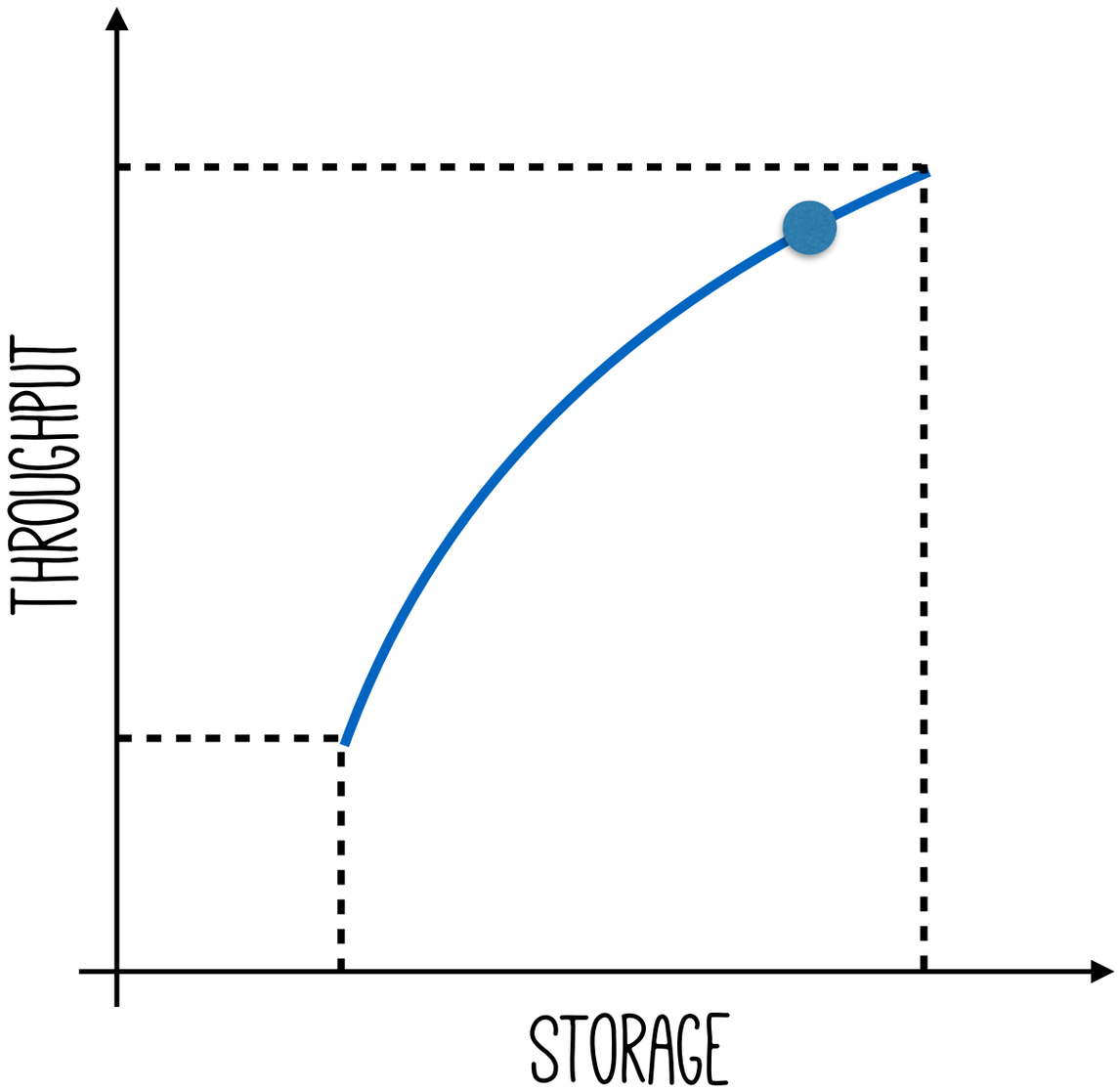


- How should partitions share cache on a server?
- How should partitions share cache across servers?
- How should requests be scheduled across replicas?

Unified Solution: Back-pressure style scheduling

Cache proportional to load, without explicit coordination


THROUGHPUT
STORAGE



Dynamic Navigation

of tradeoff curve

Assumptions & Limitations

Assumptions & Limitations

- Functionality close to state-of-the-art NoSQL stores

Assumptions & Limitations

- Functionality close to state-of-the-art NoSQL stores

`get()` `put()` `delete()` `search()` `regex()`

Assumptions & Limitations

- Functionality close to state-of-the-art NoSQL stores

`get()` `put()` `delete()` `search()` `regex()`

- Queries do not touch all servers

Assumptions & Limitations

- Functionality close to state-of-the-art NoSQL stores

`get()` `put()` `delete()` `search()` `regex()`

- Queries do not touch all servers

Many systems employ sharding schemes to avoid touching all servers, e.g., [Schism, VLDB'10]

Assumptions & Limitations

- Functionality close to state-of-the-art NoSQL stores

`get()` `put()` `delete()` `search()` `regex()`

- Queries do not touch all servers

  Many systems employ sharding schemes to avoid touching all servers, e.g., [Schism, VLDB'10]

- System is not network-bottlenecked

Assumptions & Limitations

- Functionality close to state-of-the-art NoSQL stores

`get()` `put()` `delete()` `search()` `regex()`

- Queries do not touch all servers

  Many systems employ sharding schemes to avoid touching all servers, e.g., [Schism, VLDB'10]

- System is not network-bottlenecked

  [MICA, NSDI'14] → True for most data stores today

Applications

Applications

Look at classical systems problems through a new “lens”

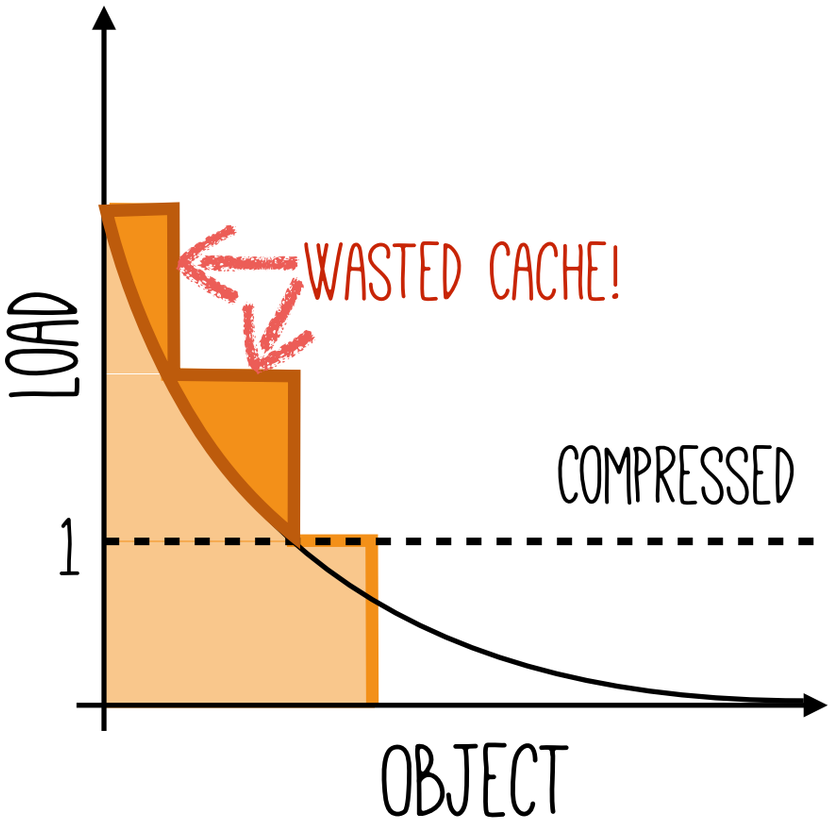
Spatial Skew

Spatial Skew

Load distribution across partitions is heavily skewed

Spatial Skew

Load distribution across partitions is heavily skewed

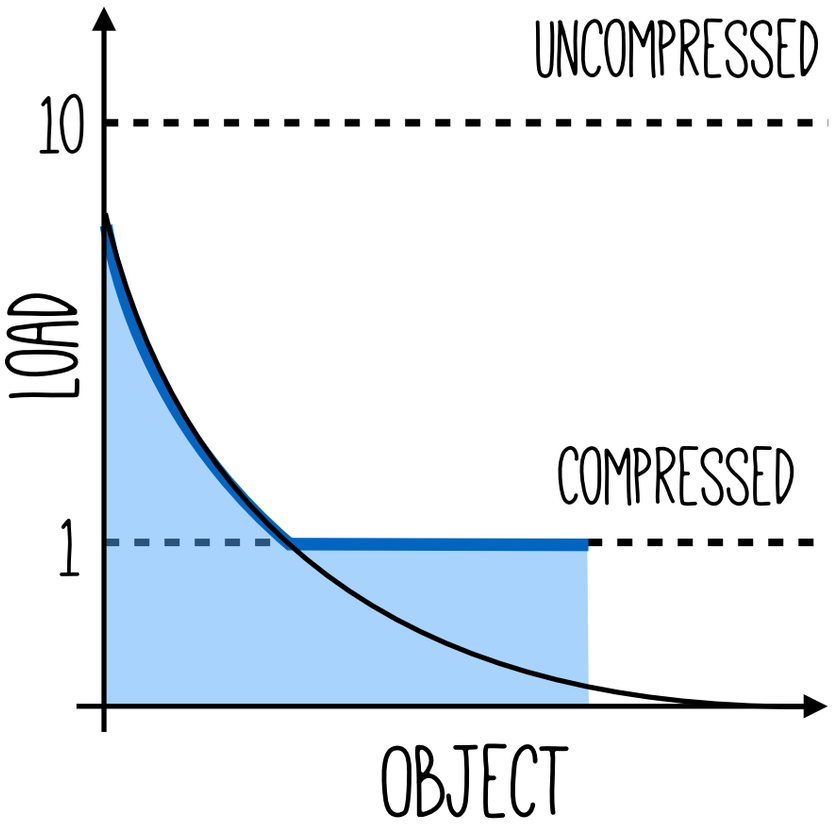


Selective Replication

#Replicas **α** Load

Spatial Skew

Load distribution across partitions is heavily skewed



Selective Replication

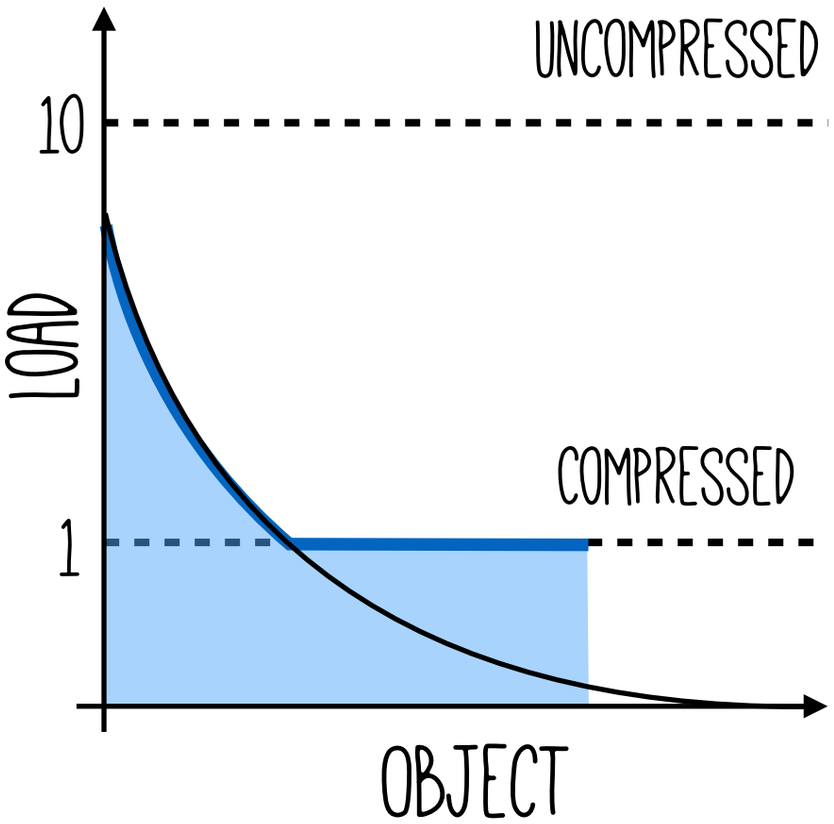
#Replicas **α** Load

BlowFish

Fractionally change storage just enough to meet load

Spatial Skew

Load distribution across partitions is heavily skewed



Selective Replication

#Replicas **α** Load

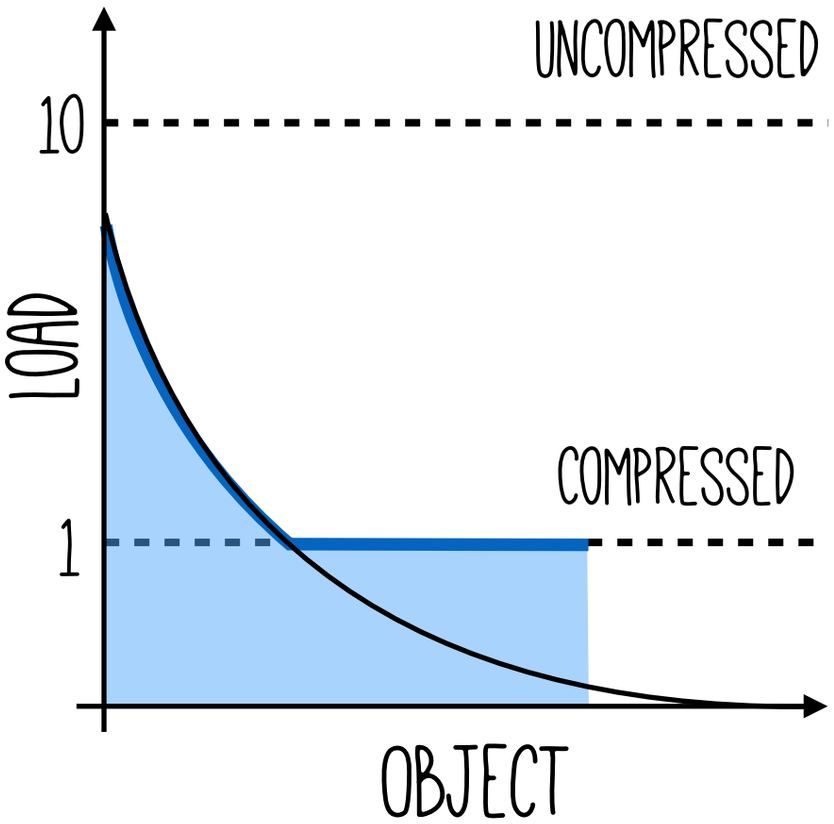
BlowFish

Fractionally change storage just enough to meet load

1.5x higher throughput than Selective Replication,

Spatial Skew

Load distribution across partitions is heavily skewed



Selective Replication

#Replicas **α** Load

BlowFish

Fractionally change storage just enough to meet load

1.5x higher throughput than Selective Replication,

within 10% of optimal

Changes in Spatial Skew

Changes in Spatial Skew

Study on Facebook Warehouse Cluster [HotStorage'13]

Changes in Spatial Skew

Study on Facebook Warehouse Cluster [HotStorage'13]

Transient failures → 90% of failures

Changes in Spatial Skew

Study on Facebook Warehouse Cluster [HotStorage'13]

Transient failures → 90% of failures

Replica creation delayed by 15 mins

Changes in Spatial Skew

Study on Facebook Warehouse Cluster [HotStorage'13]

Transient failures → 90% of failures

Replica creation delayed by 15 mins

Leads to variation in load over time

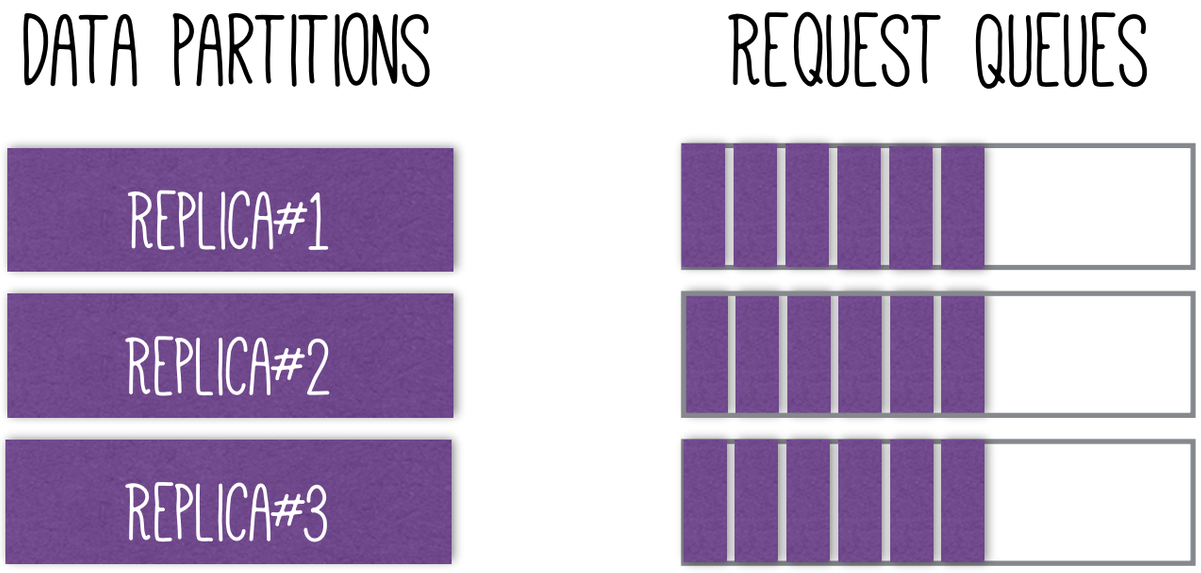
Changes in Spatial Skew

Study on Facebook Warehouse Cluster [HotStorage'13]

Transient failures → 90% of failures

Replica creation delayed by 15 mins

Leads to variation in load over time



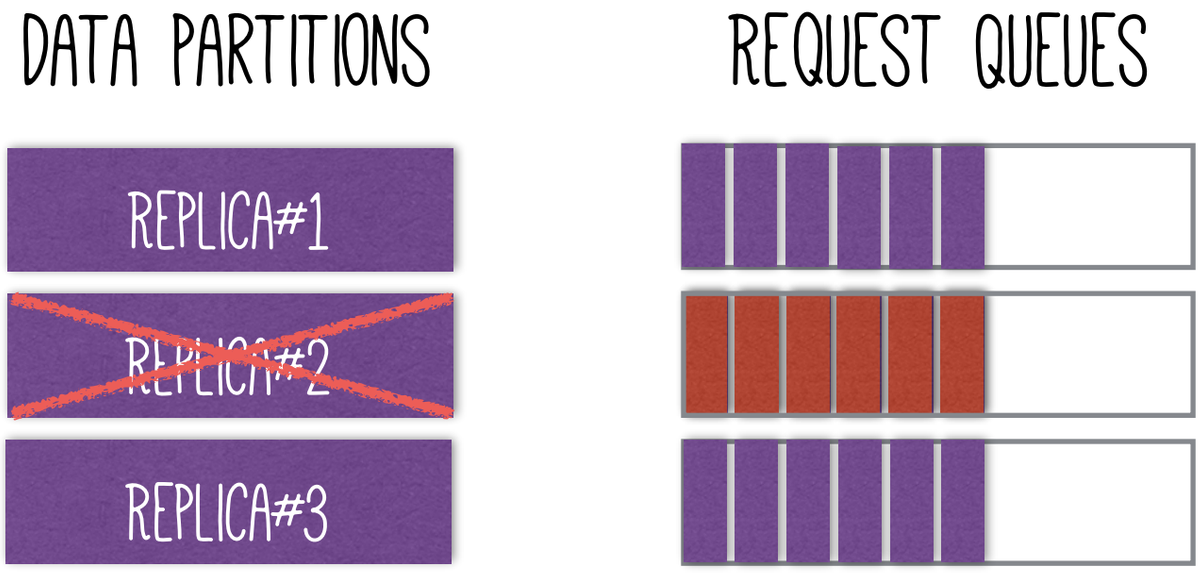
Changes in Spatial Skew

Study on Facebook Warehouse Cluster [HotStorage'13]

Transient failures → 90% of failures

Replica creation delayed by 15 mins

Leads to variation in load over time



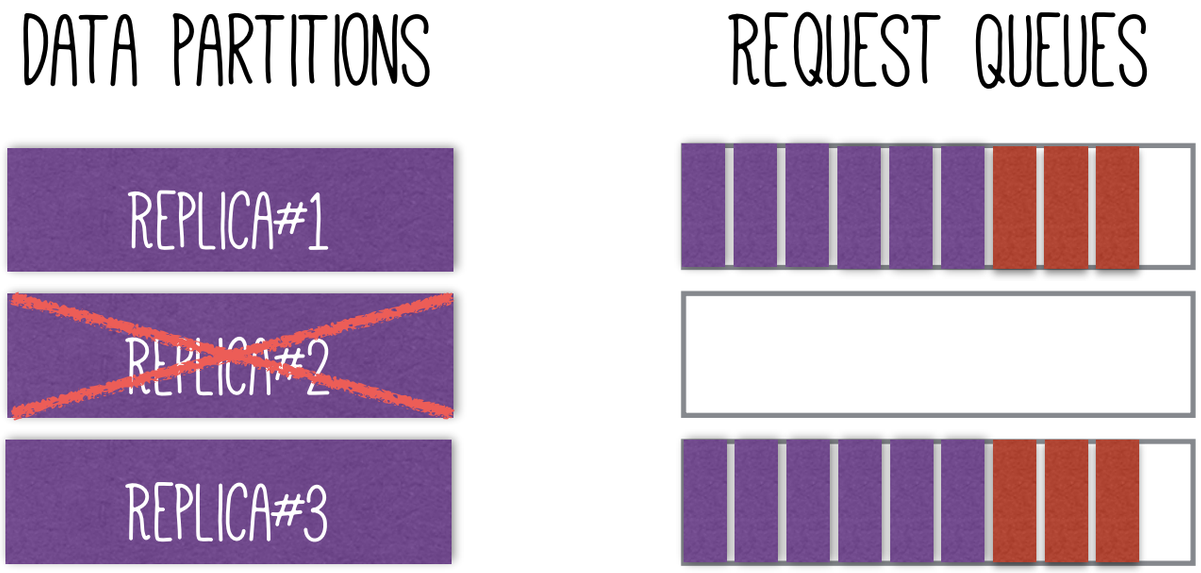
Changes in Spatial Skew

Study on Facebook Warehouse Cluster [HotStorage'13]

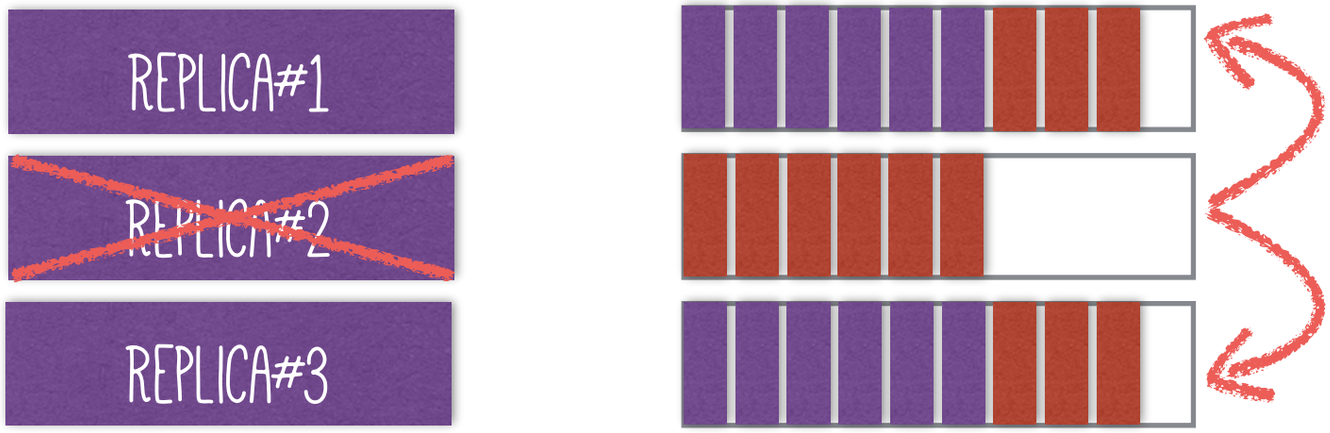
Transient failures → 90% of failures

Replica creation delayed by 15 mins

Leads to variation in load over time

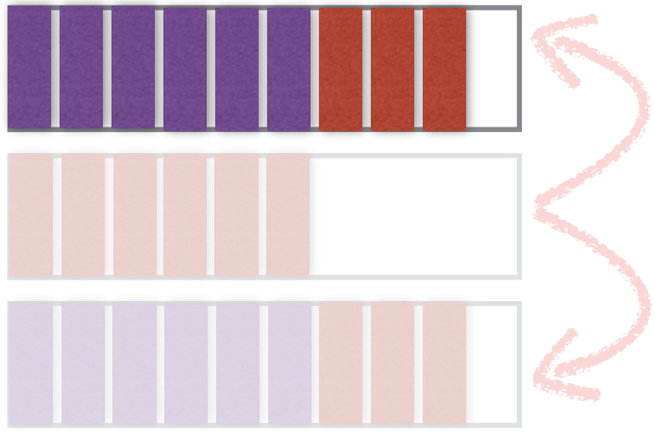


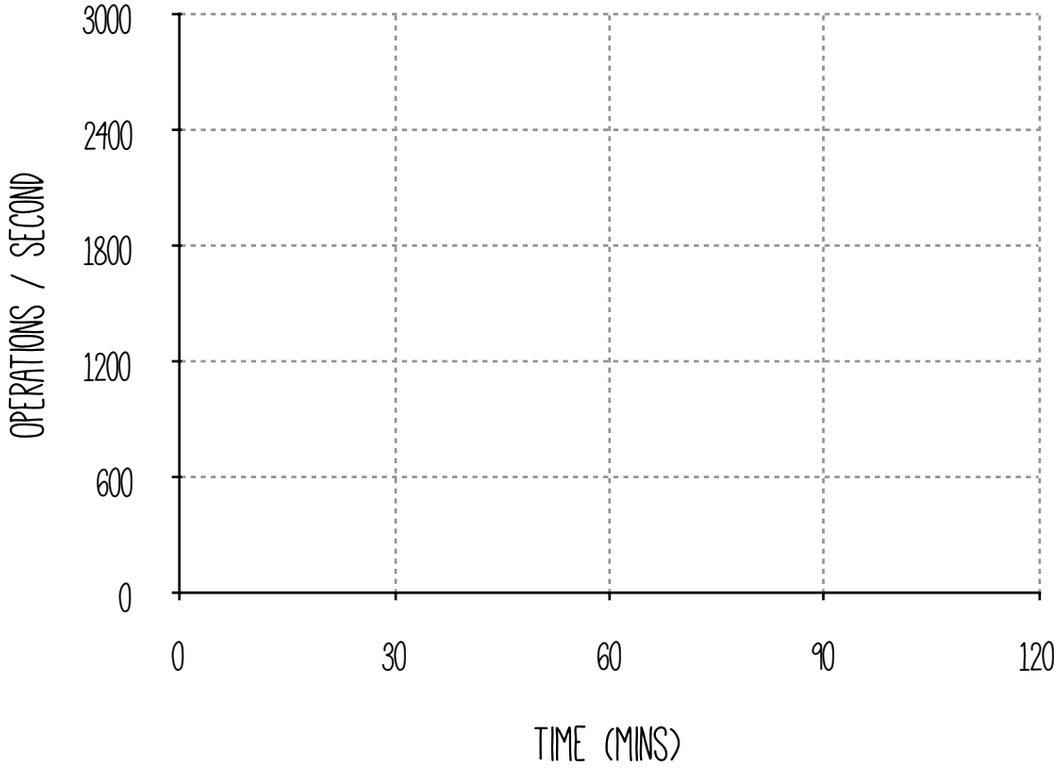
Changes in Spatial Skew



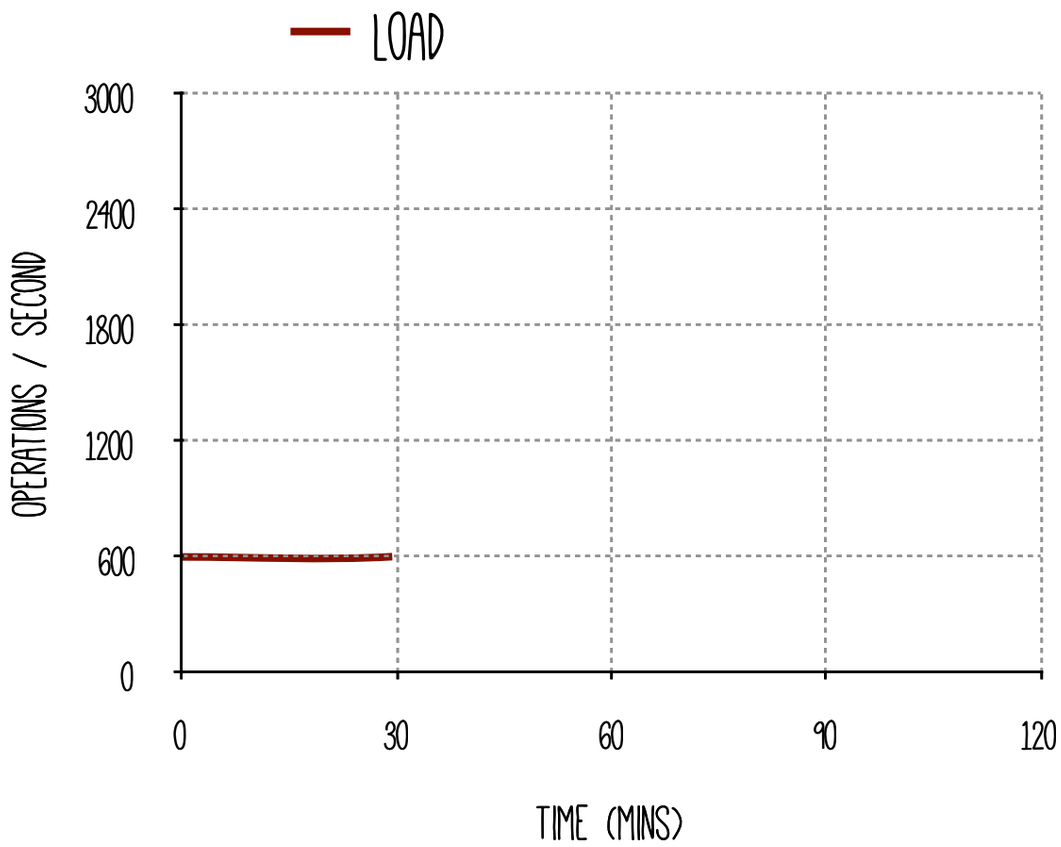
Changes in Spatial Skew

Changes in Spatial Skew

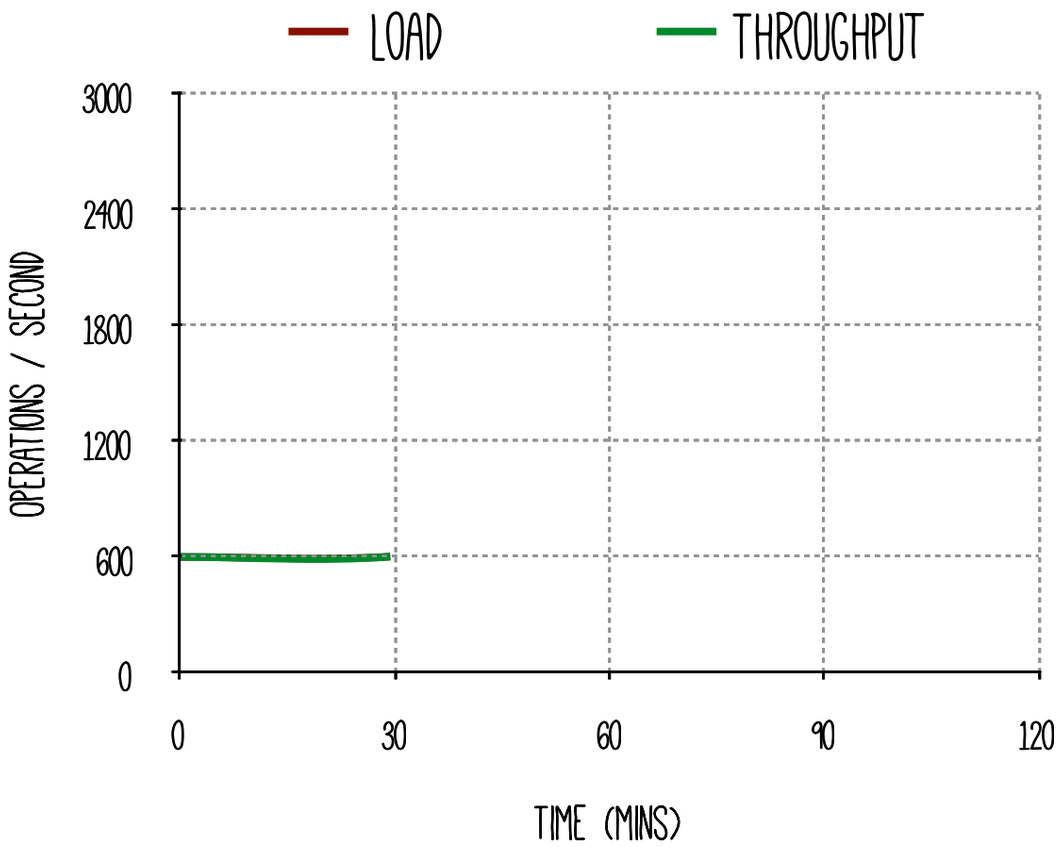




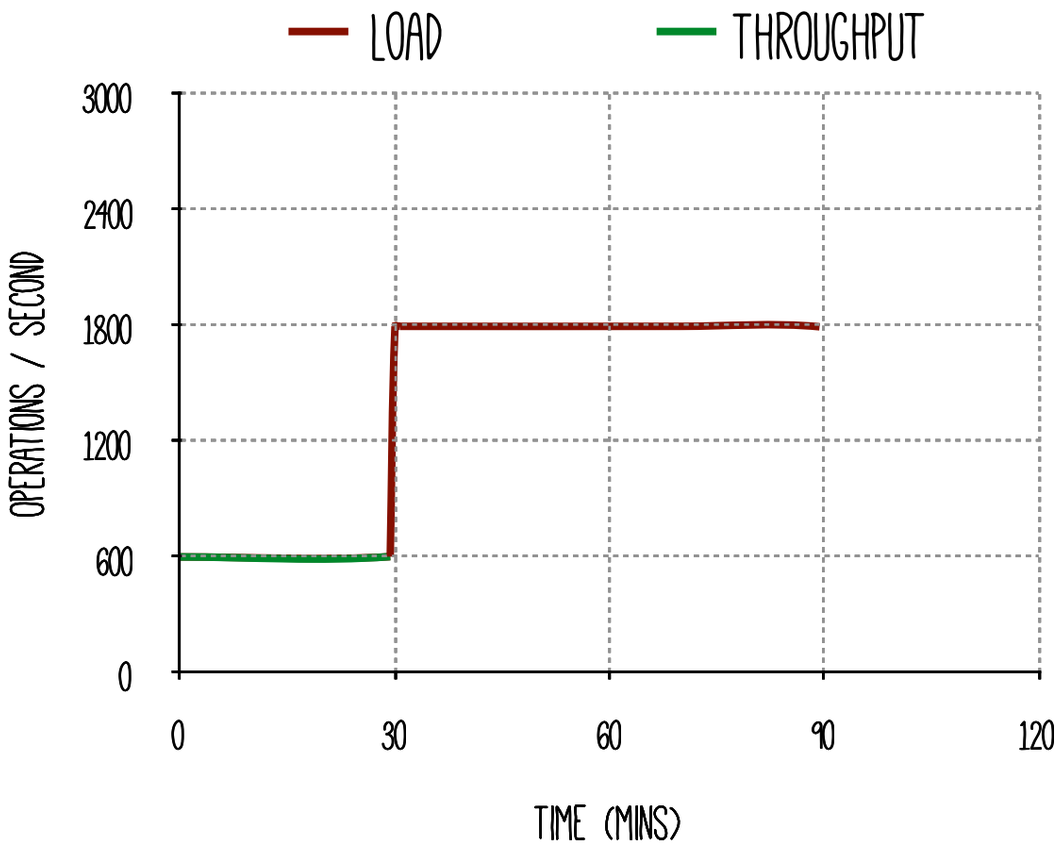
Changes in Spatial Skew



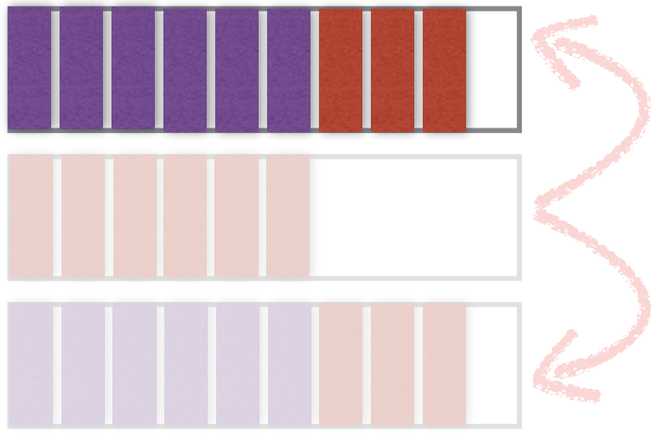
Changes in Spatial Skew

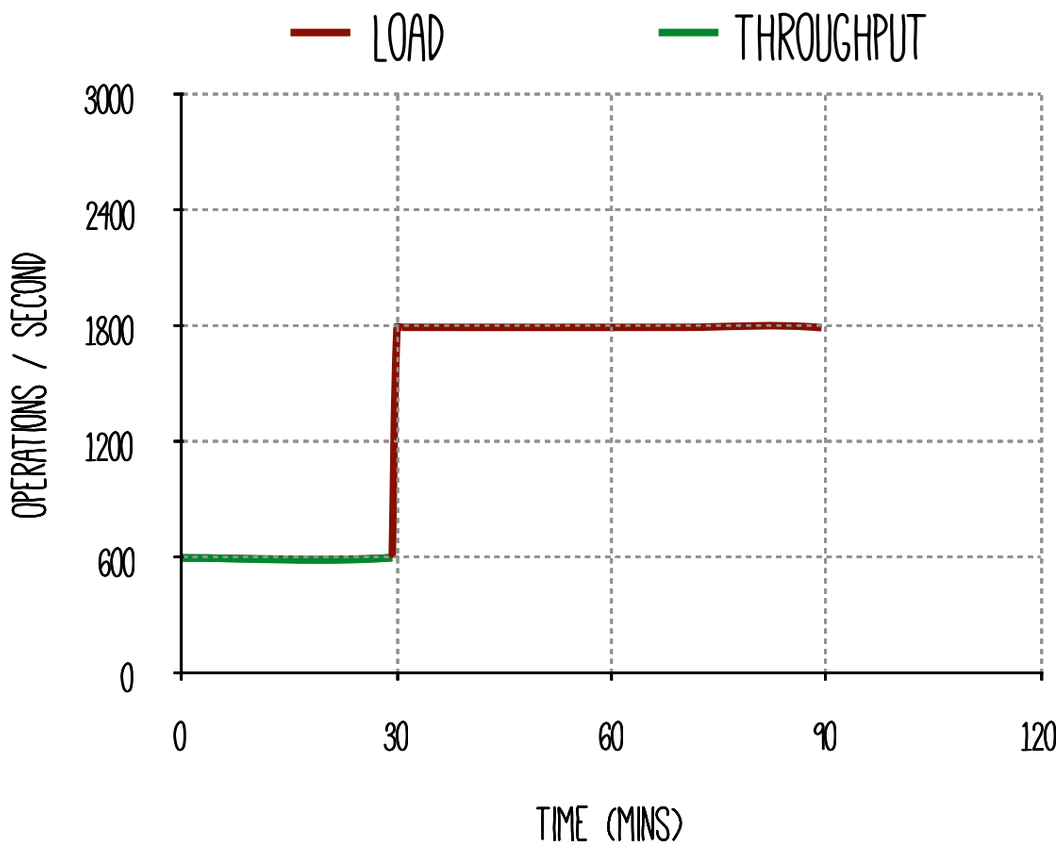


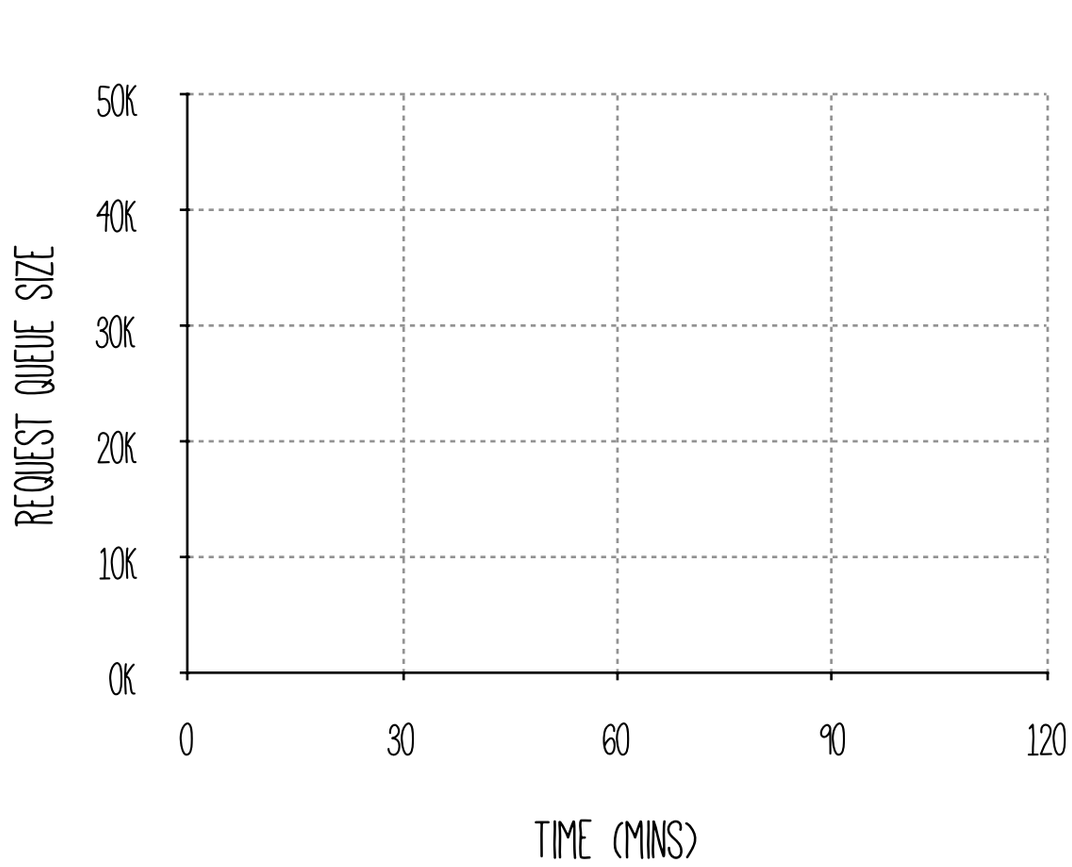
Changes in Spatial Skew



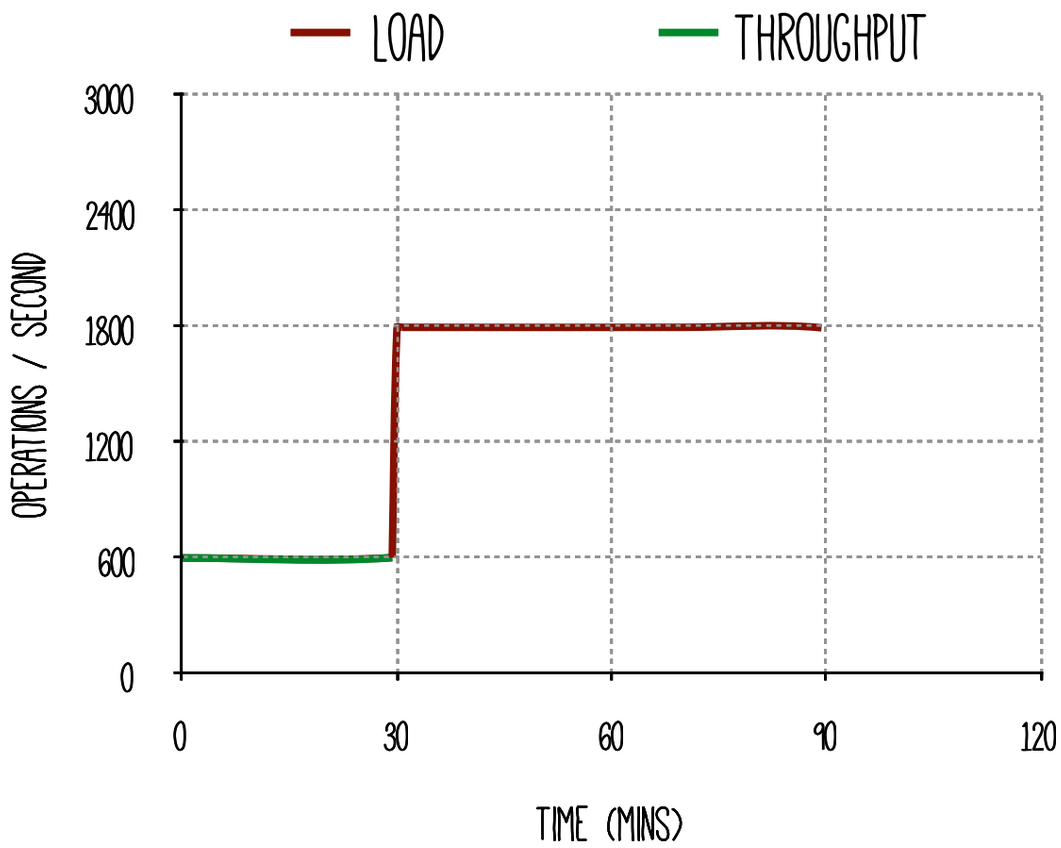
Changes in Spatial Skew

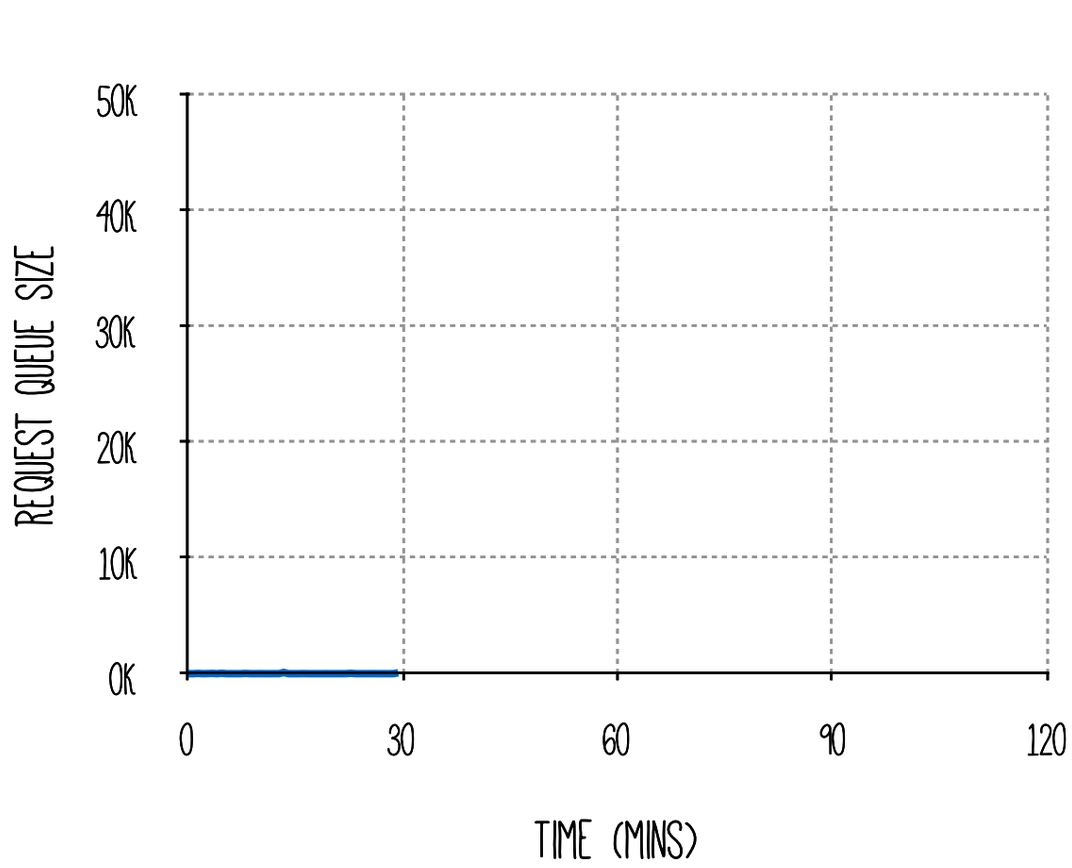




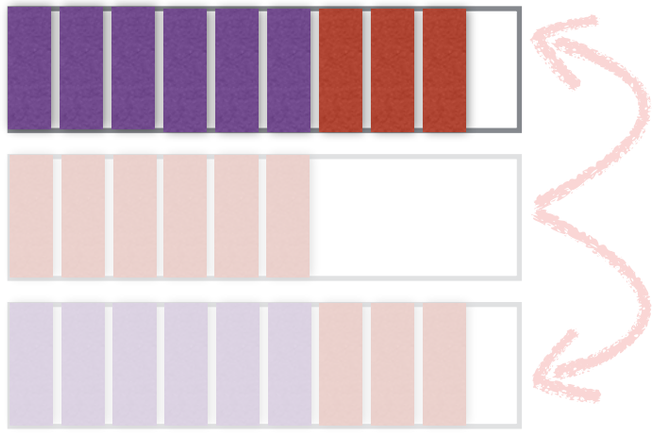


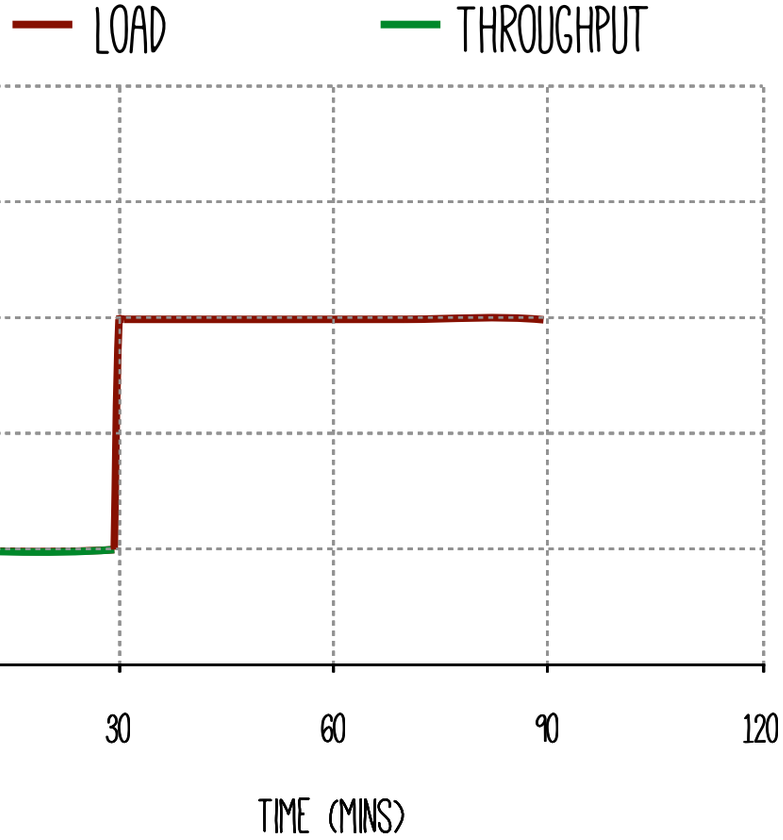
Changes in Spatial Skew

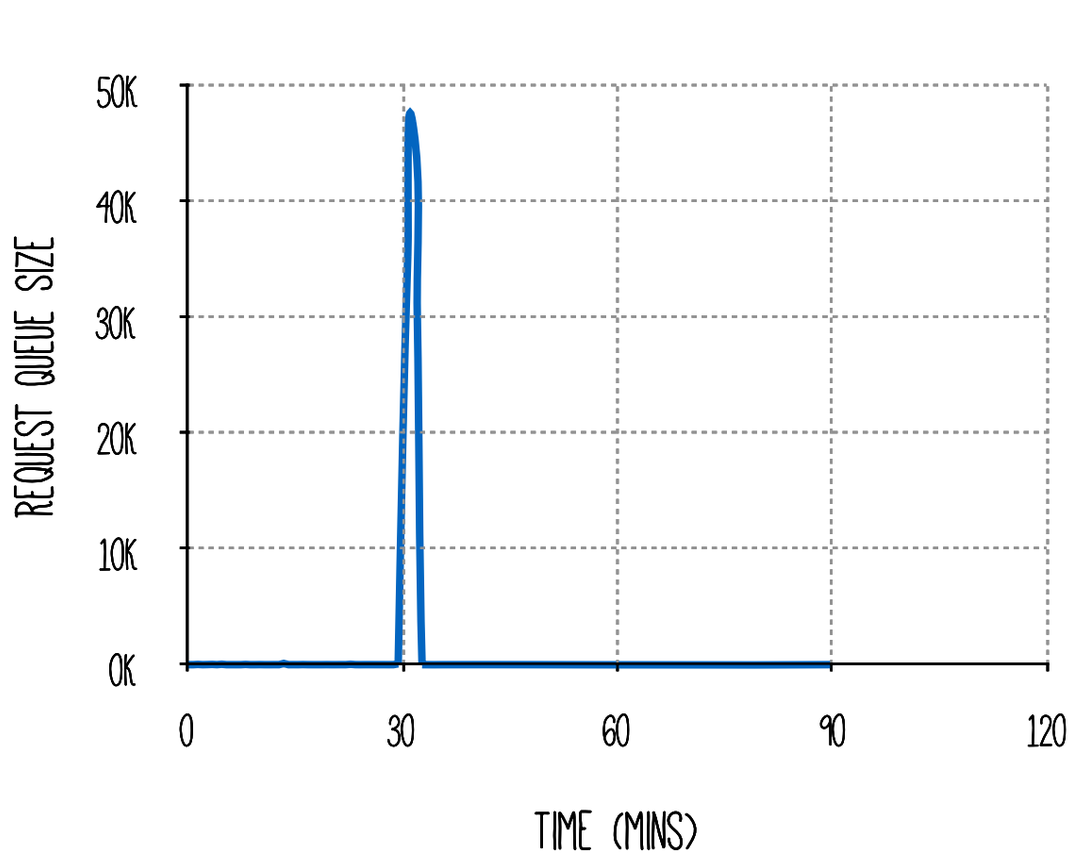




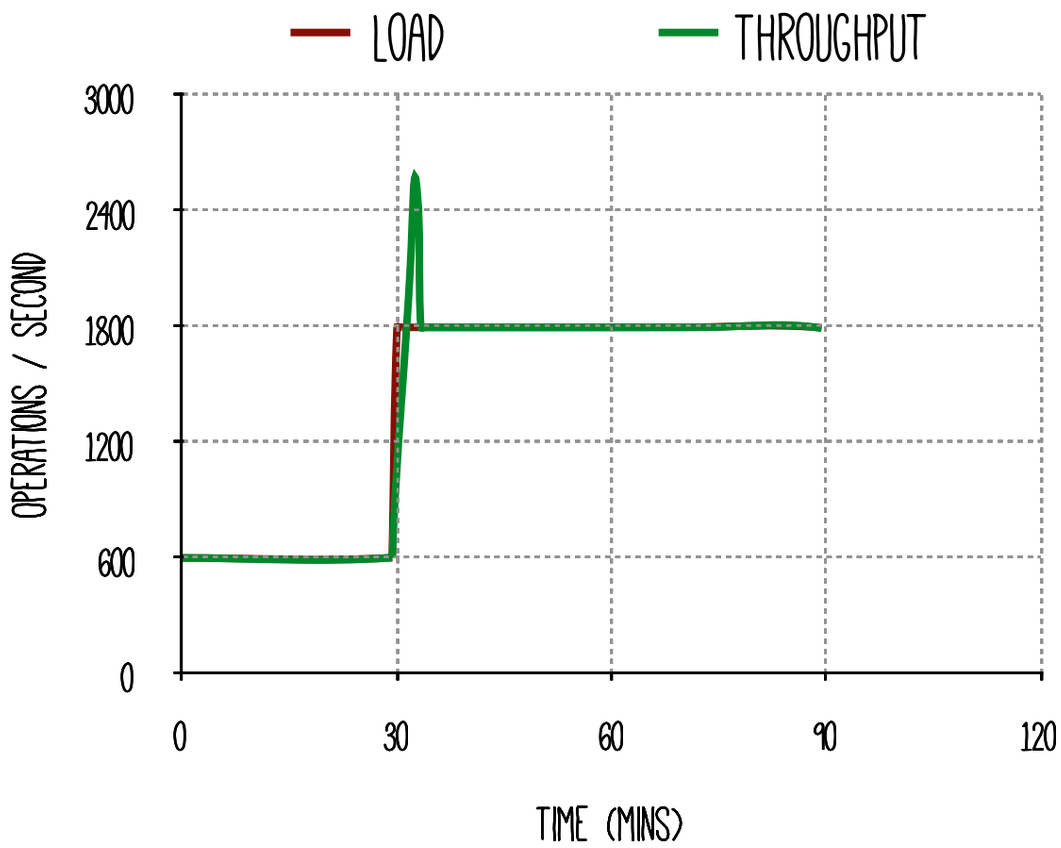
Changes in Spatial Skew

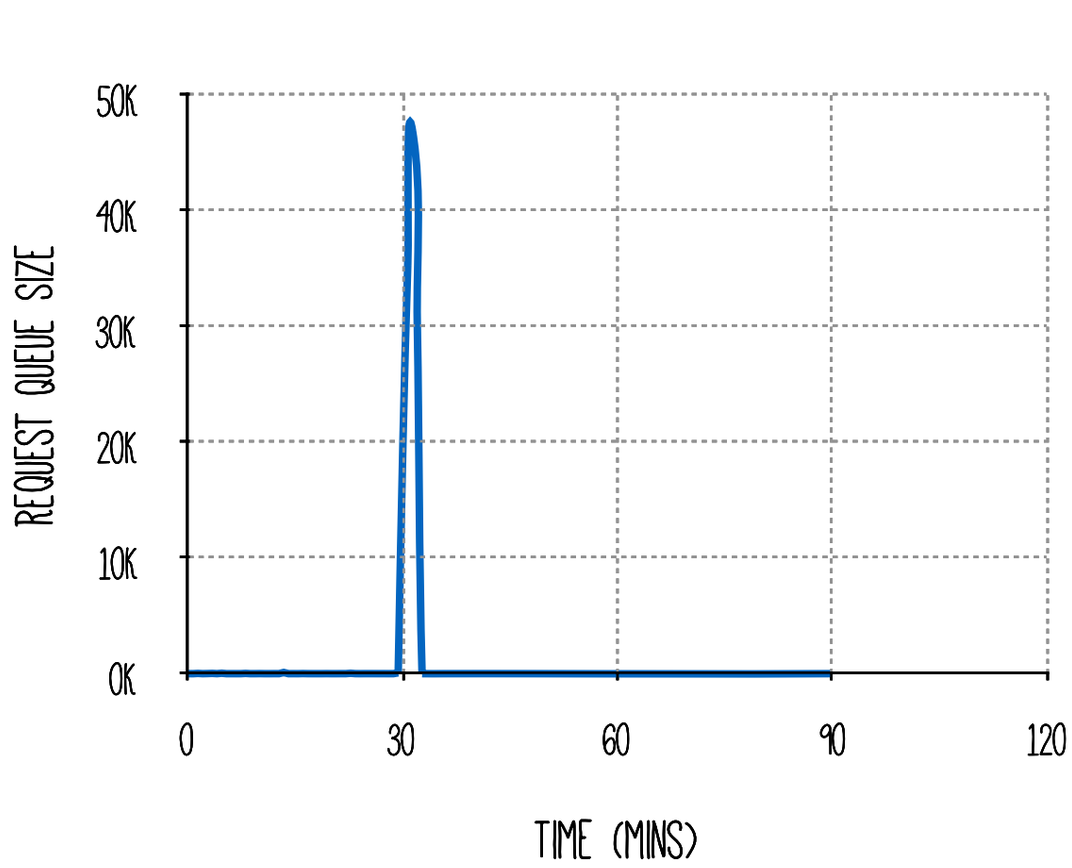




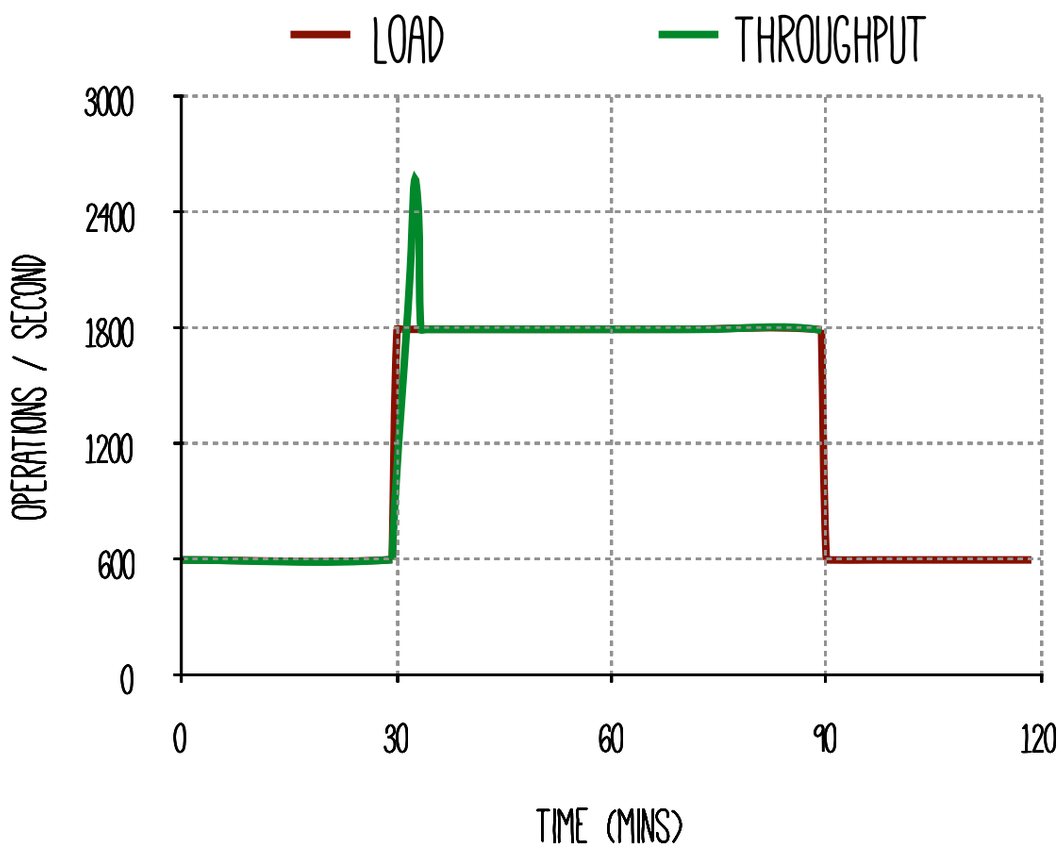


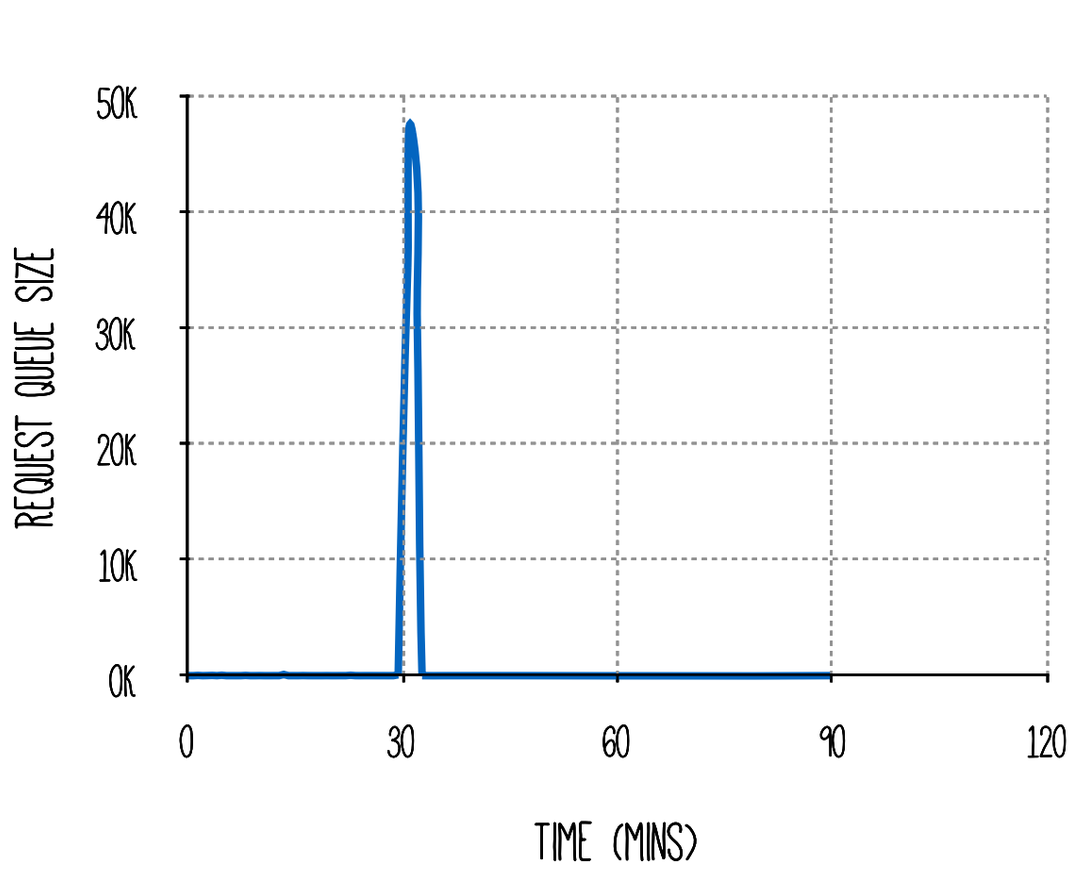
Changes in Spatial Skew



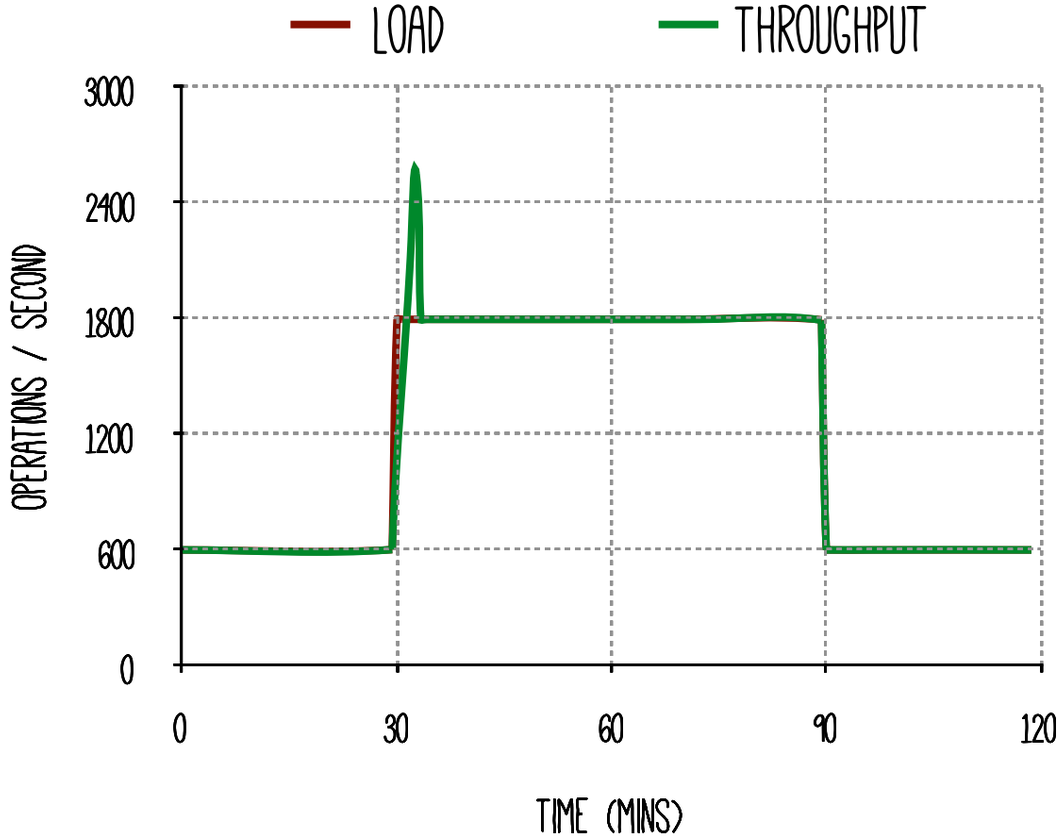


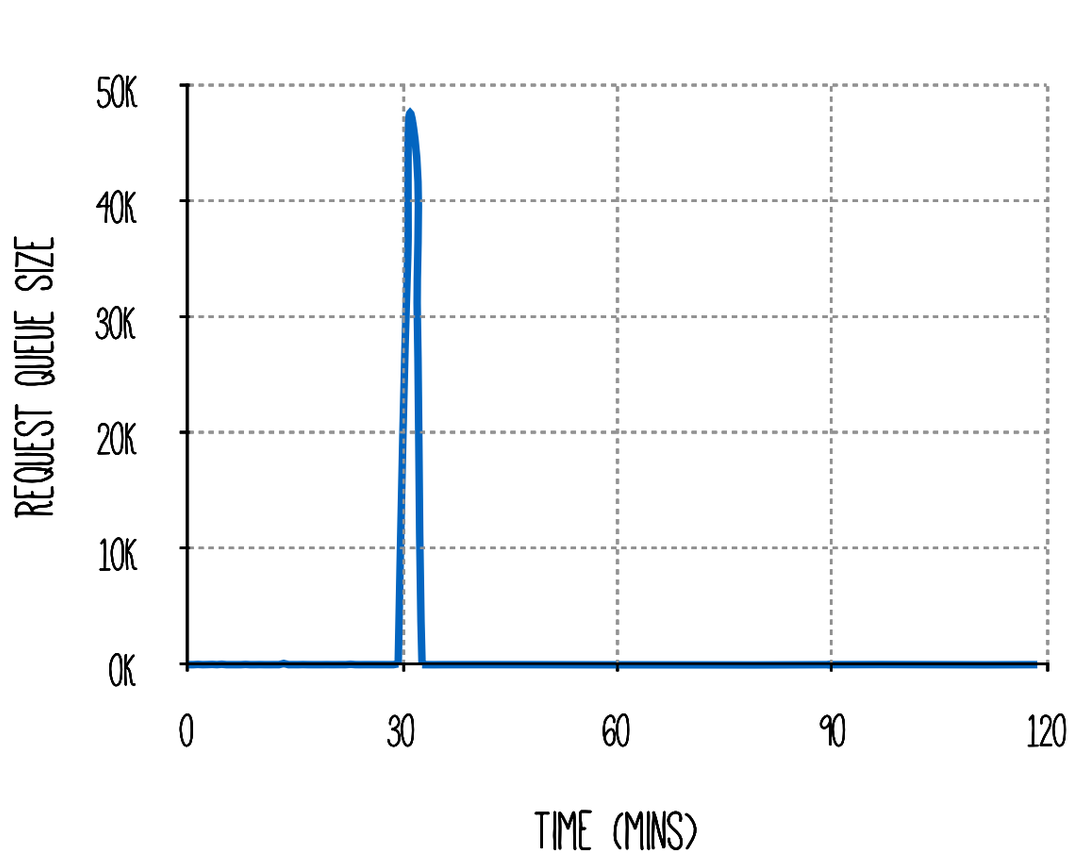
Changes in Spatial Skew



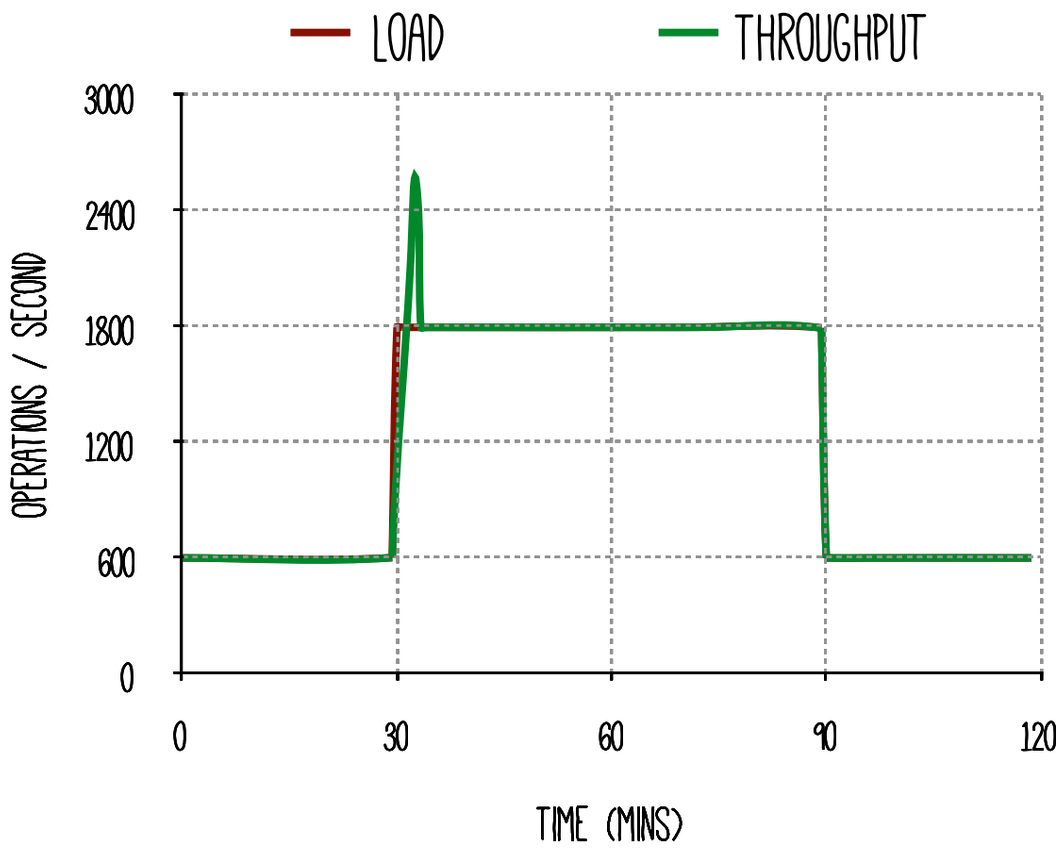


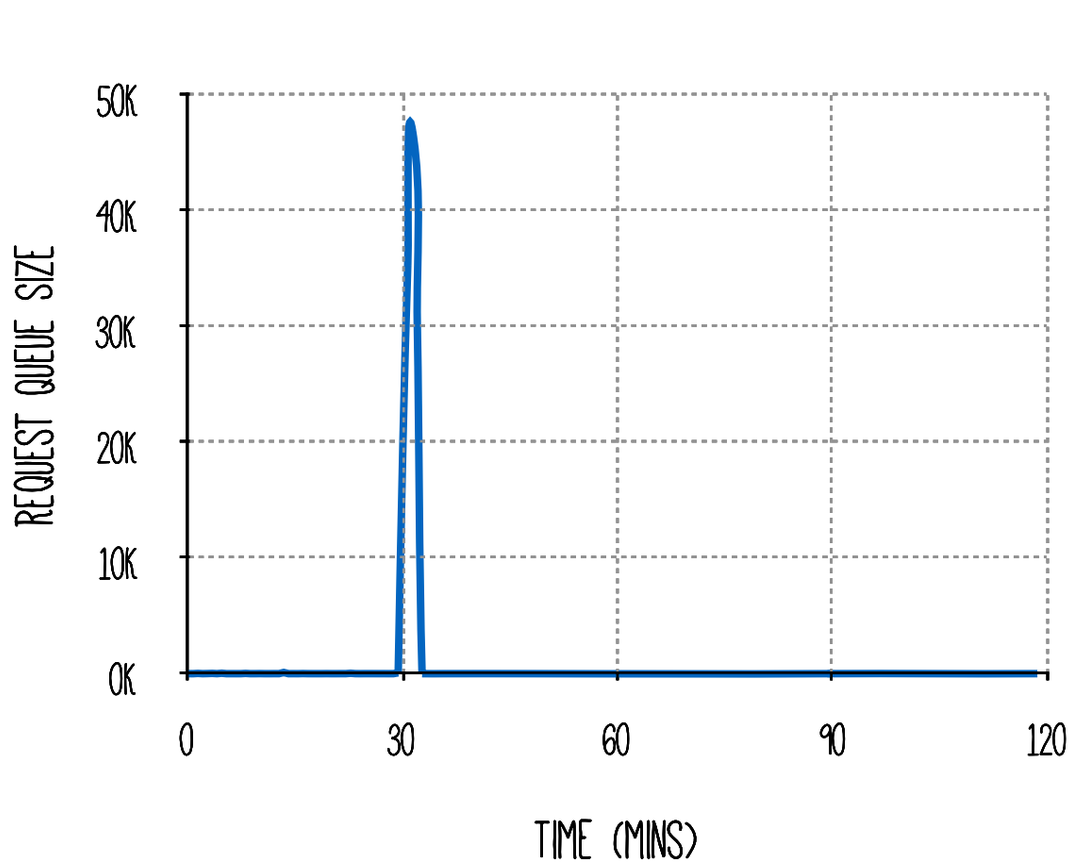
Changes in Spatial Skew





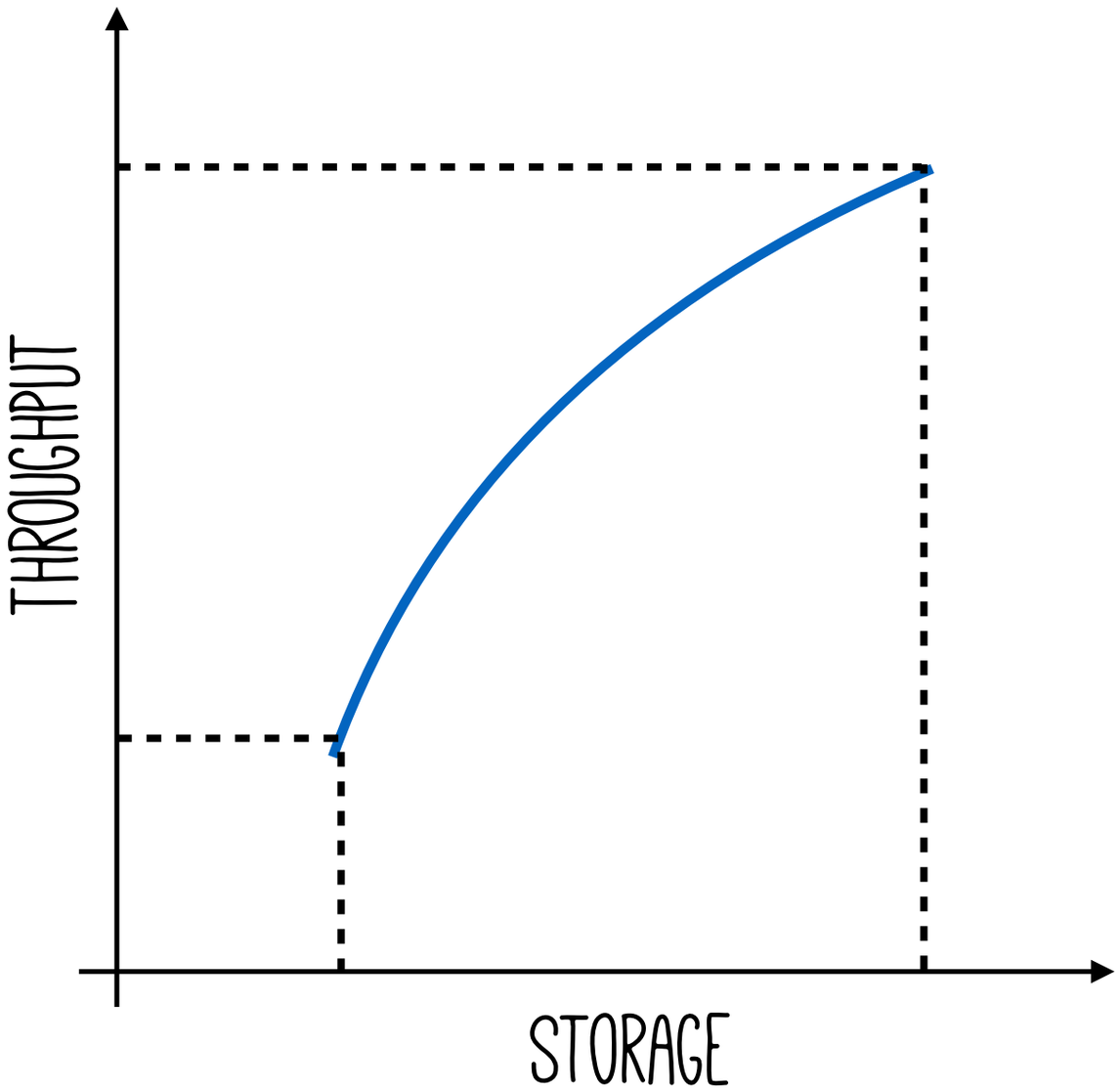
Changes in Spatial Skew



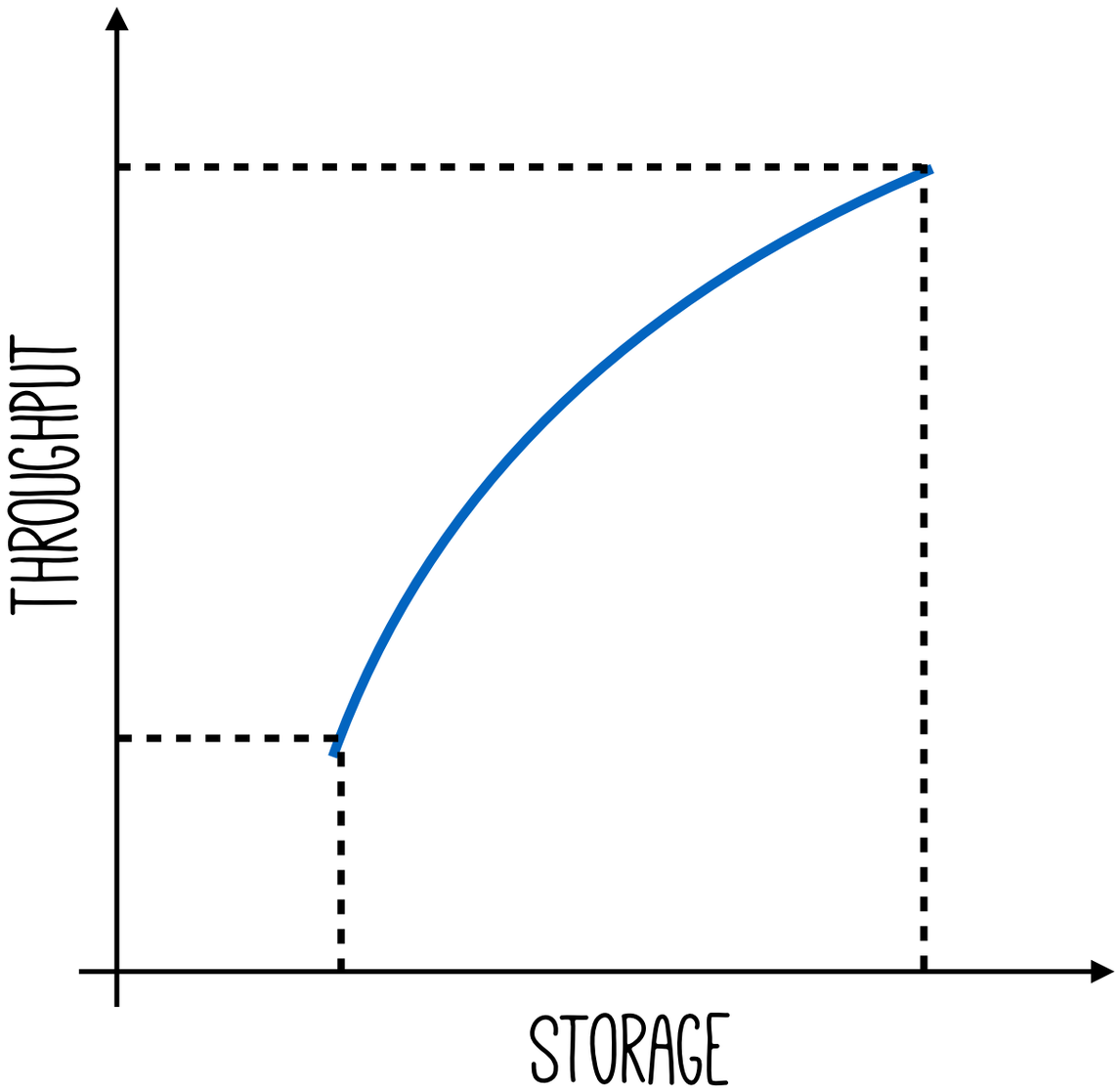


Adapts to 3x higher load in < 5 mins

Summary

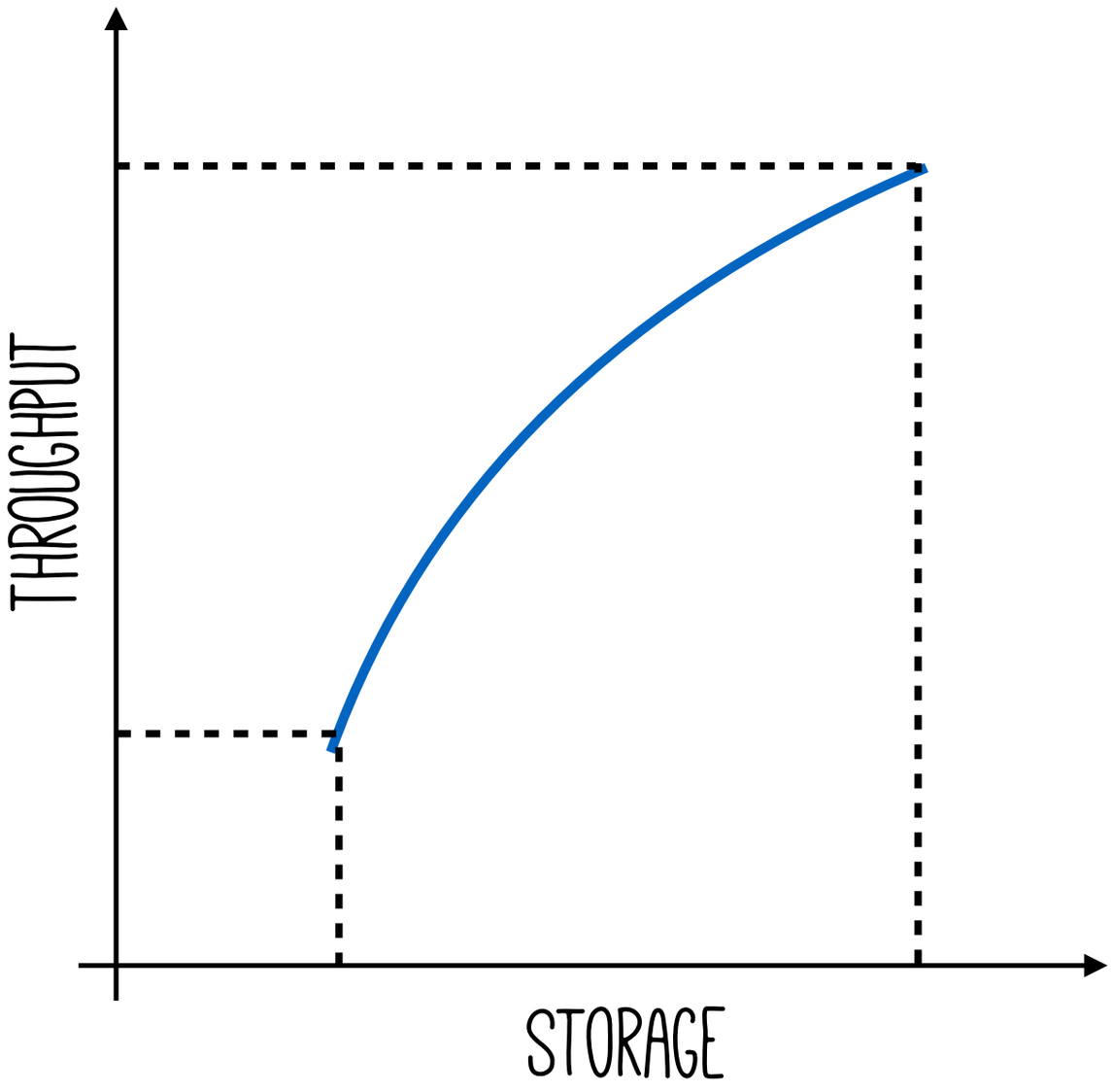


Summary



Smooth Tradeoff Curve

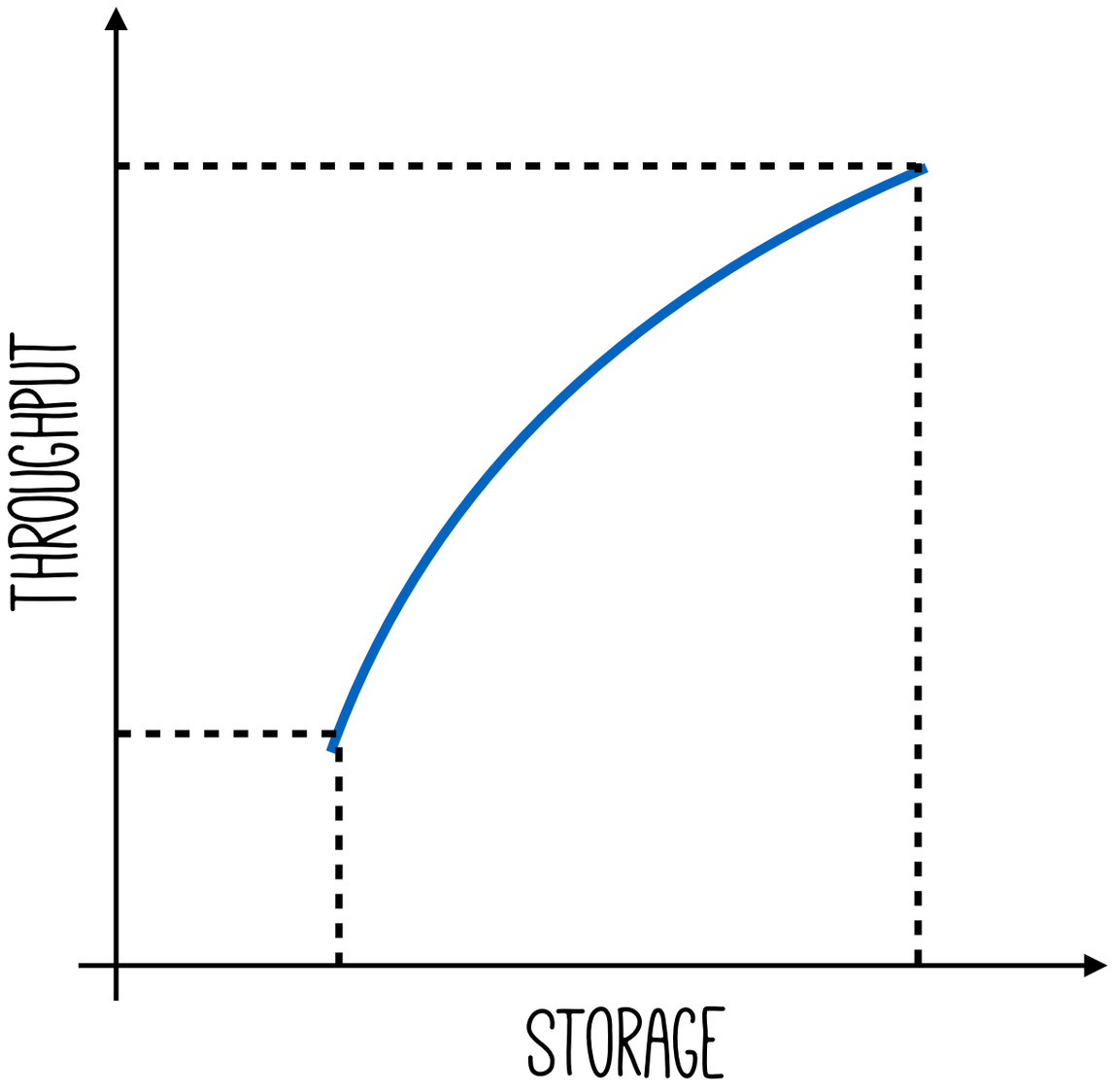
Summary



Smooth Tradeoff Curve

Dynamic Navigation

Summary

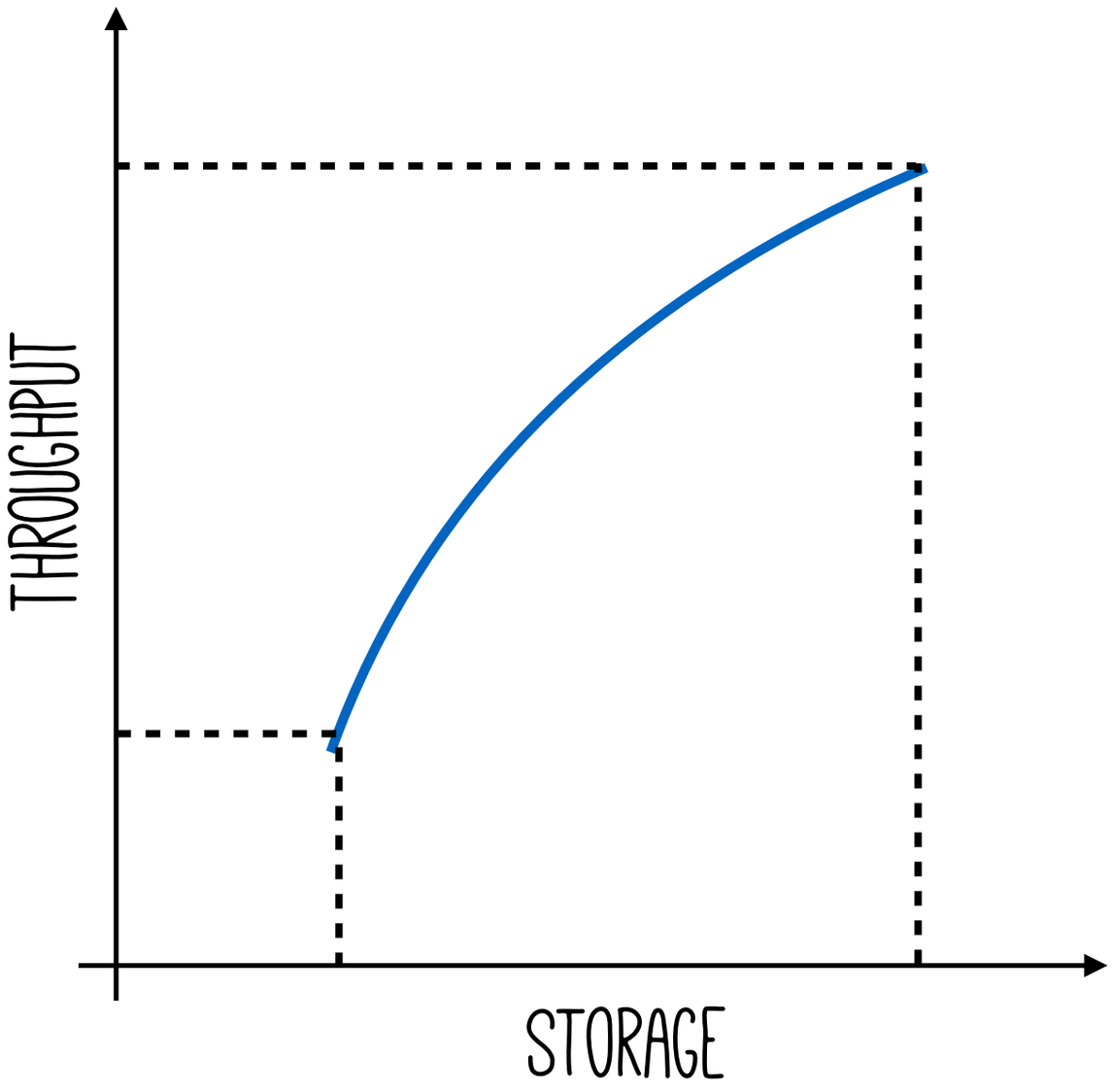


Smooth Tradeoff Curve

Dynamic Navigation

Applications in Several Classical Systems Problems

Summary



Smooth Tradeoff Curve

Dynamic Navigation

Applications in Several Classical Systems Problems

Thank You! Questions?

Backup Slides