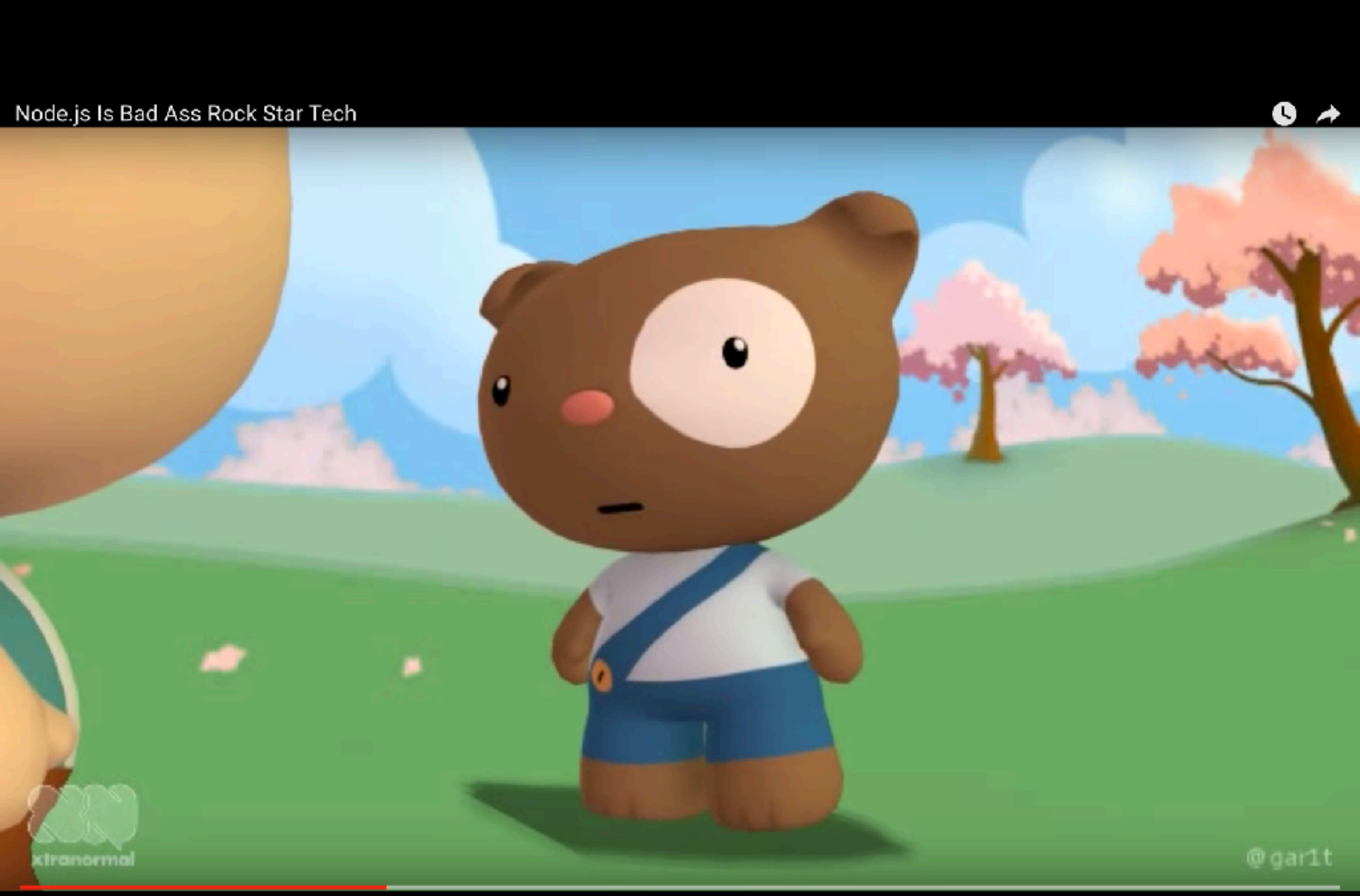
Async or Bust!?

Todd L. Montgomery
@toddlmontgomery

About me…

- ✓ ***The Myth(s)***
- ✓ ***Illusion & Cognitive Dissonance***
- ✓ ***Impact of the Myth***
- ✓ ***Subjectivity of the Myth***

Do you remember….



https://www.youtube.com/watch?v=bzkRVzciAZg

The Myth

Sequential is good enough

The Myth

Sequential is good enough

....

Async is complicated & error prone

A Right Way

&

MANY Wrong Ways

In Reality…

MANY Right Ways

&

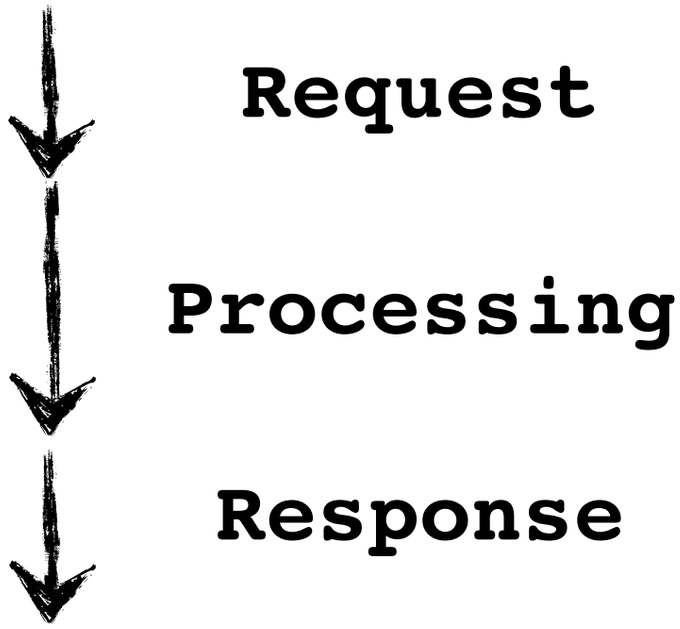
MANY Wrong Ways

Wording

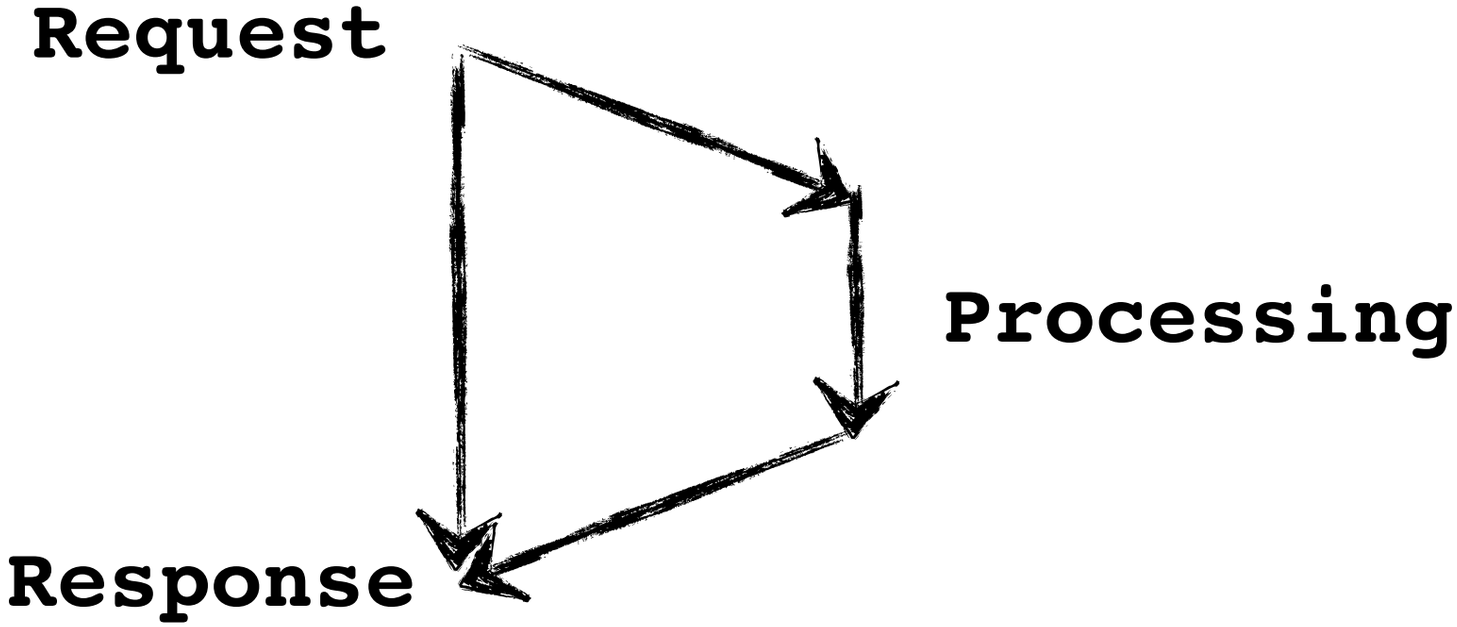
Sequential Synchronous Blocking

Asynchronous Non-Blocking

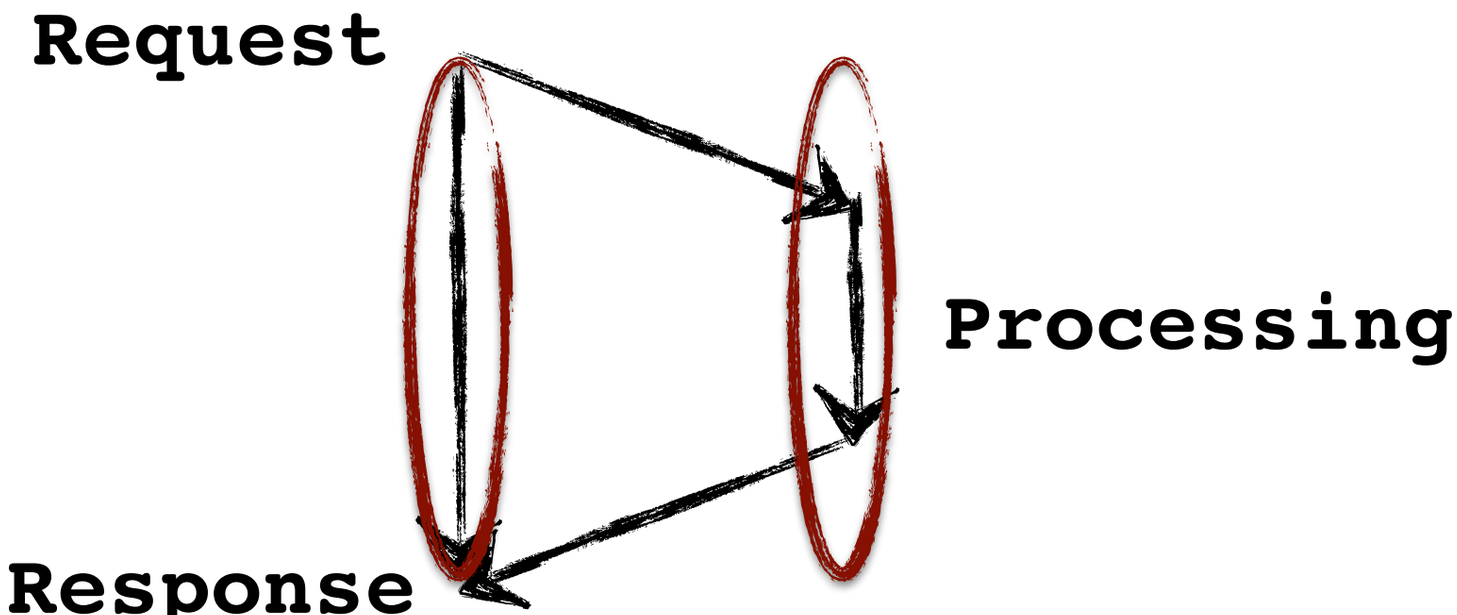
What is Sync?



What is Async?

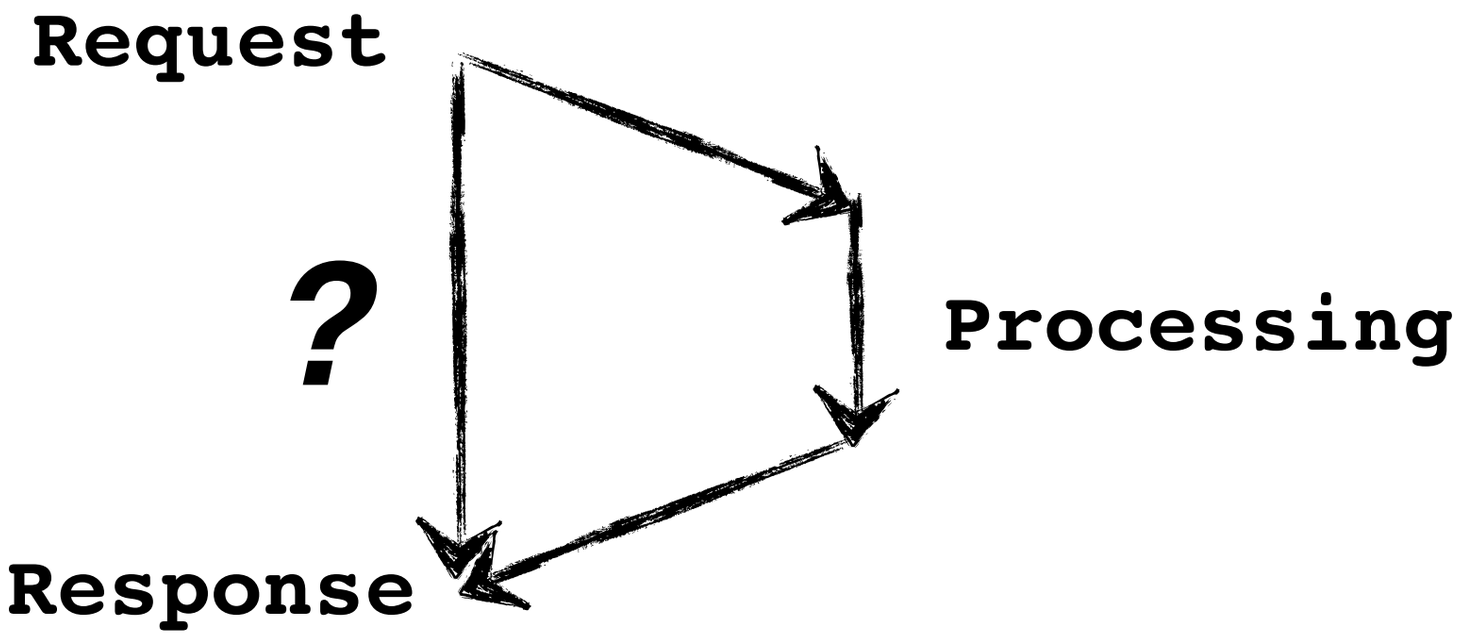


What is Async?

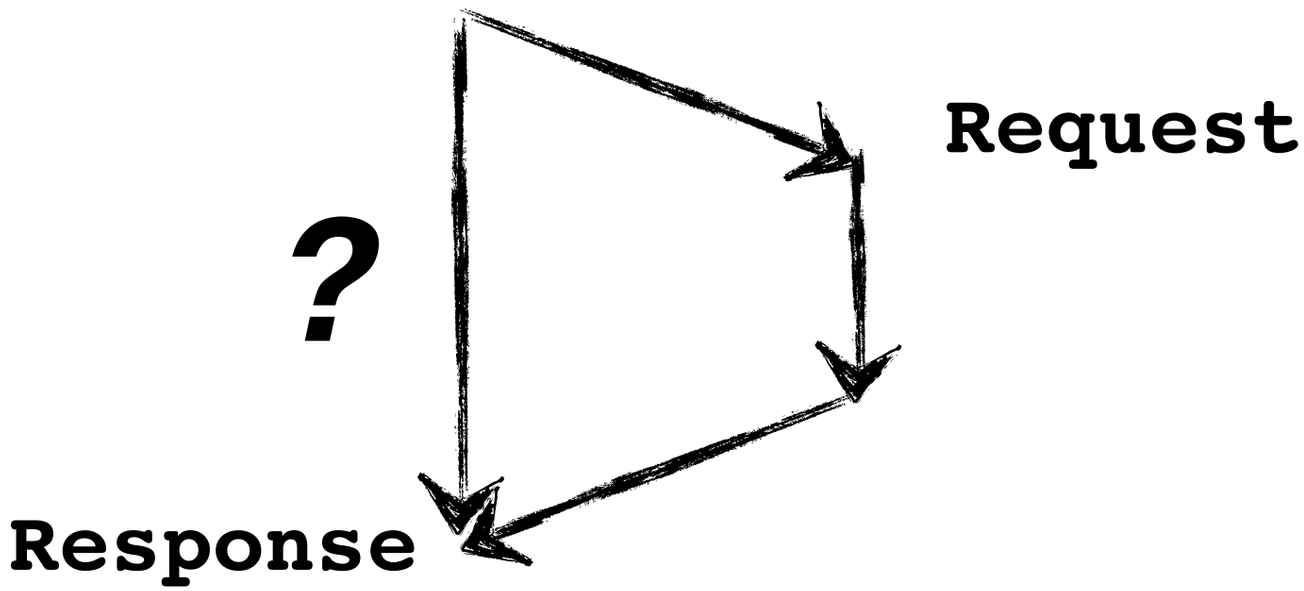


Cross Thread/Core/Node

What is Async?



What is Event-Driven?



Illusion of Sequentiality

Ordering is an Illusion

Compiler can re-order

Runtime can re-order

CPU can re-order

Ordering has to be imposed!

Illusion of Sequentiality

Illusion of Sequentiality

- ***CPUs - Load/Store Buffers***

Illusion of Sequentiality

- ***CPUs - Load/Store Buffers***
- ***Storage - Caches***

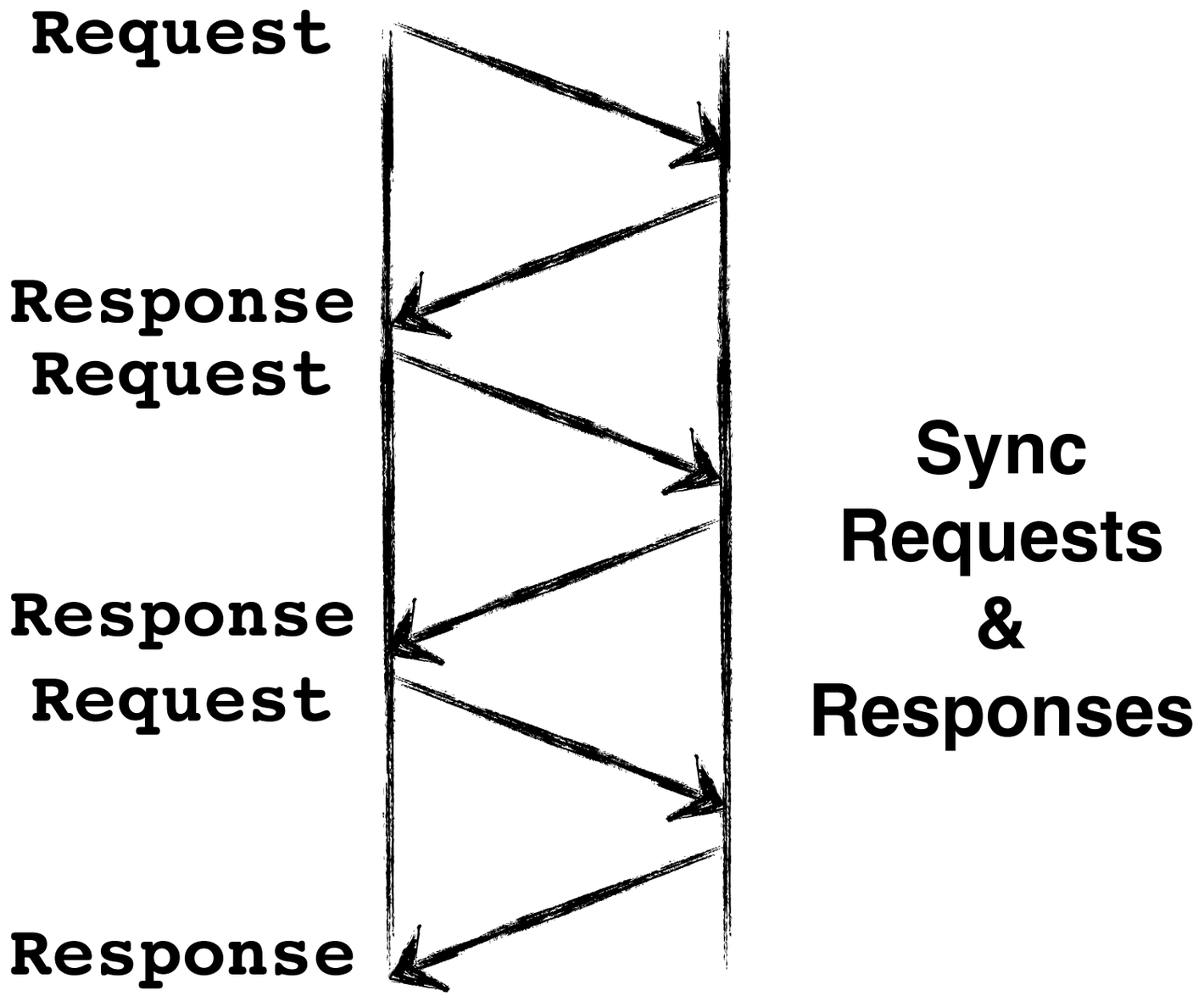
Illusion of Sequentiality

- ***CPUs - Load/Store Buffers***
- ***Storage - Caches***
- ***OS - VM & Caches***

Illusion of Sequentiality

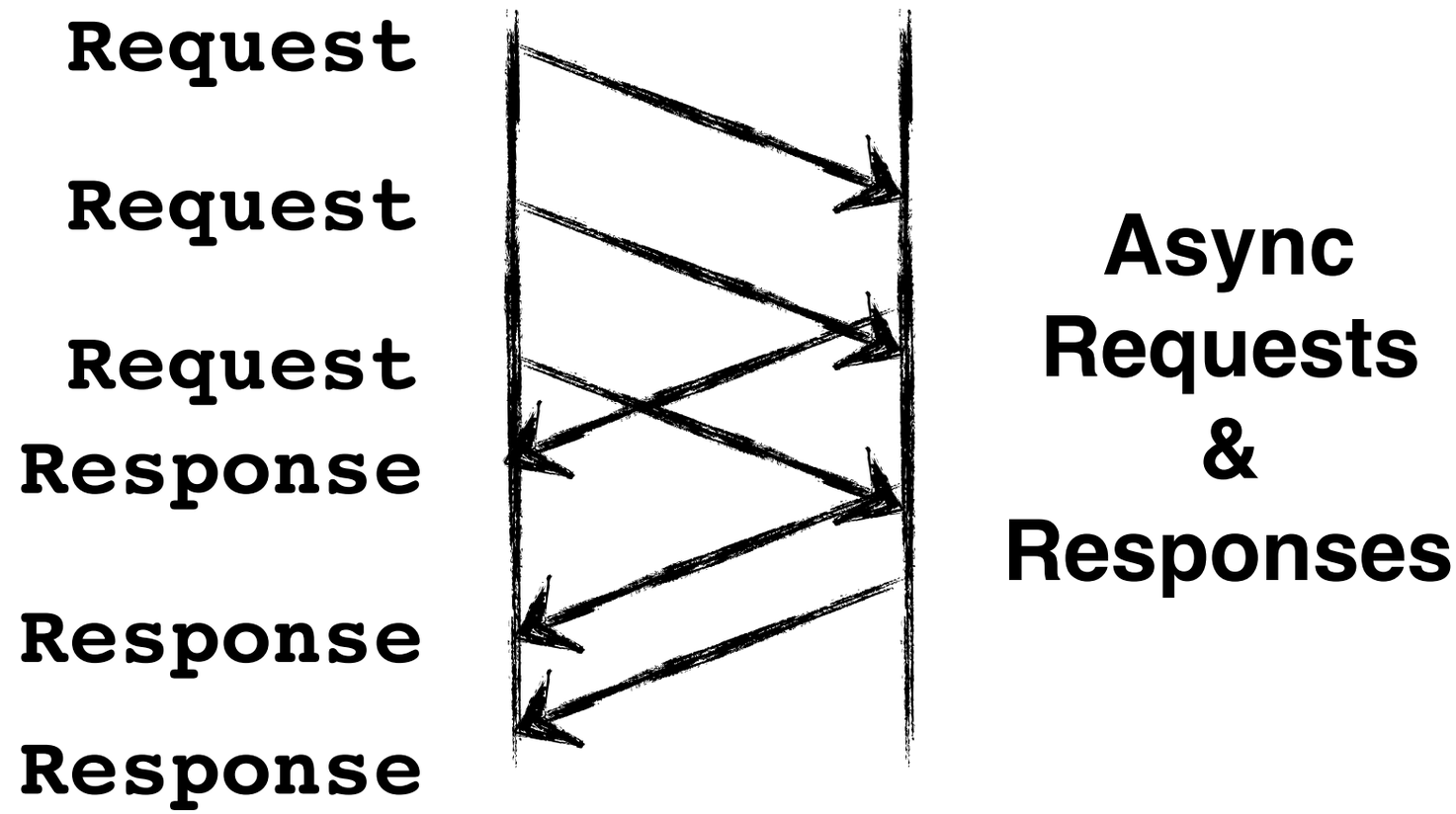
- ***CPUs - Load/Store Buffers***
- ***Storage - Caches***
- ***OS - VM & Caches***
- ***Library - Promises / Futures***

As easy as...*



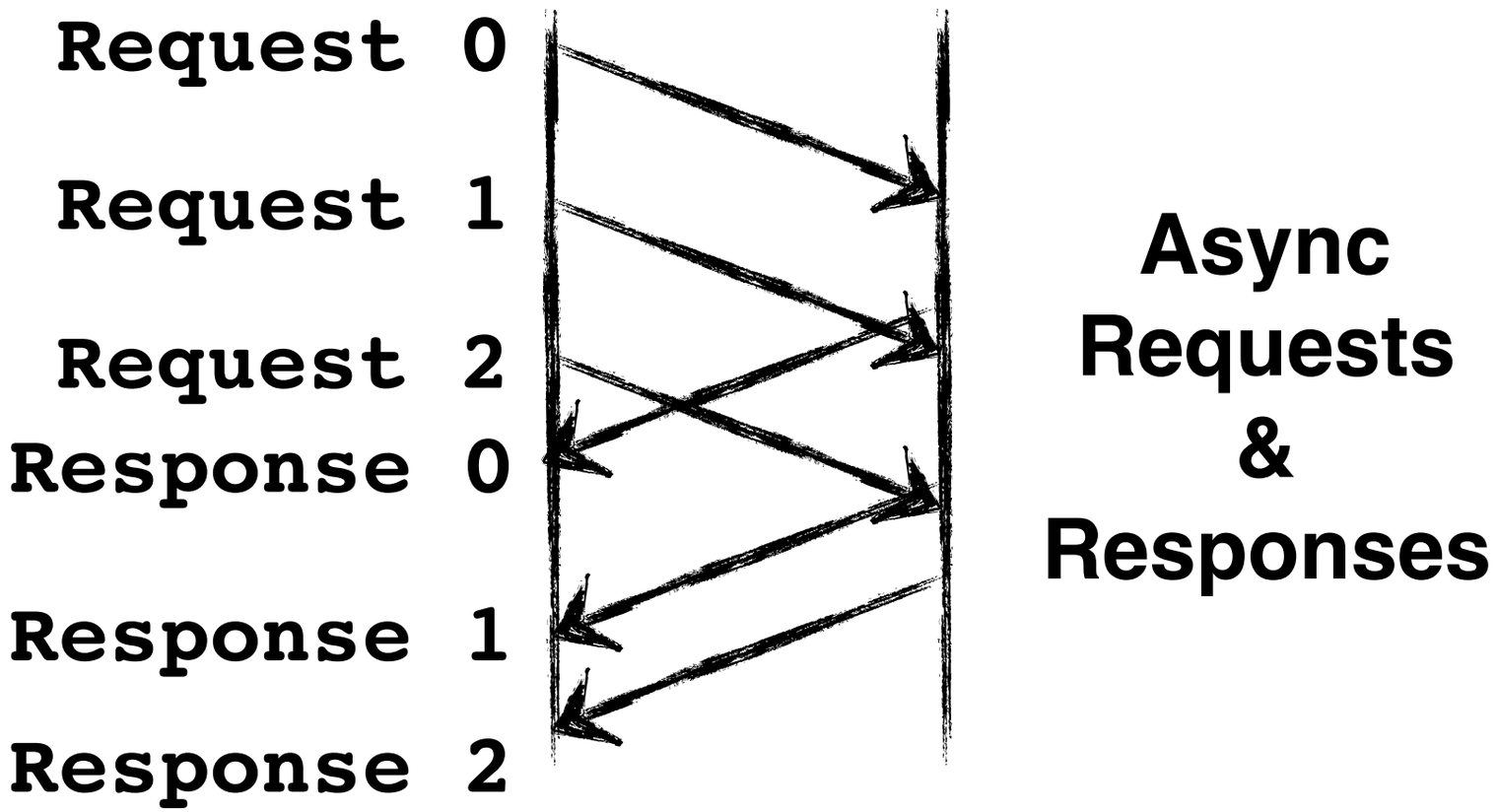
** - for some definition of*

But with efficiency of…



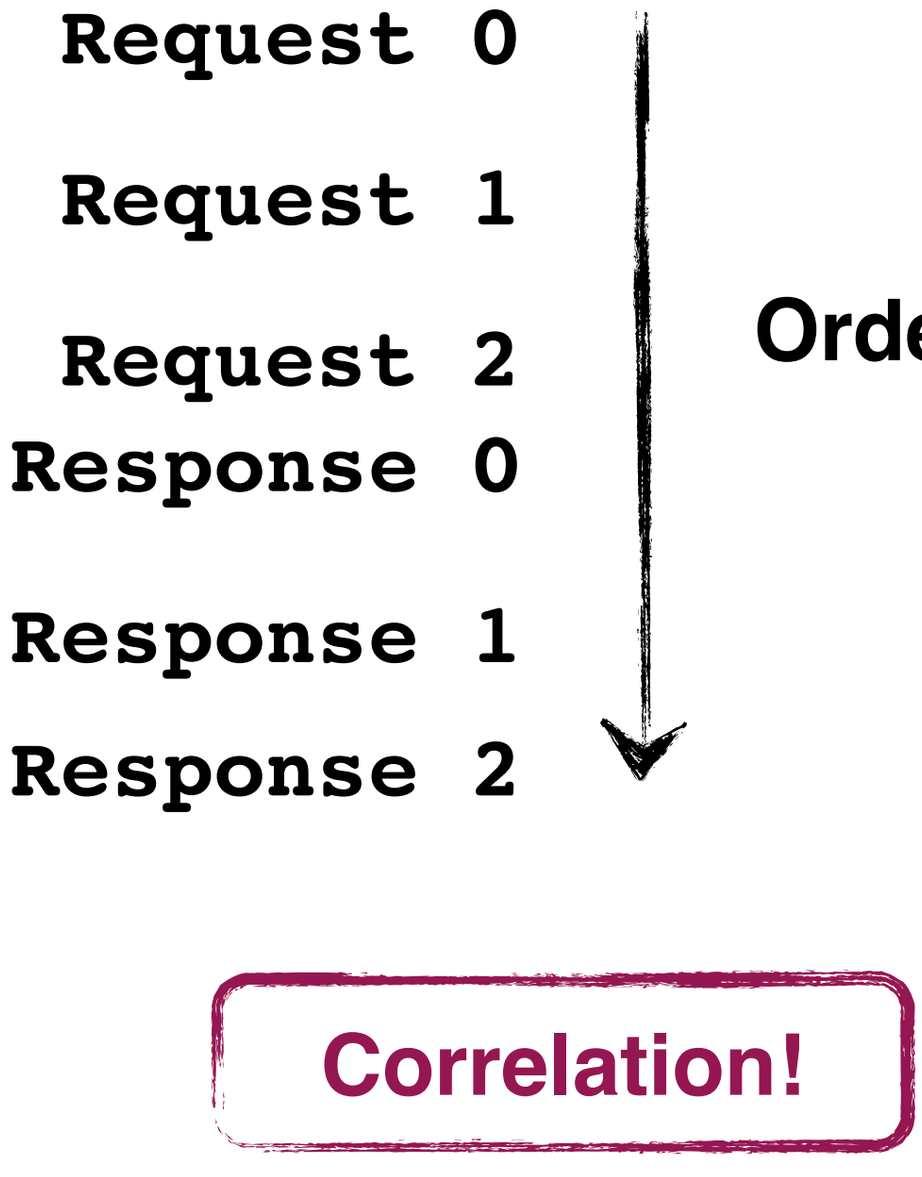
Do something

while waiting

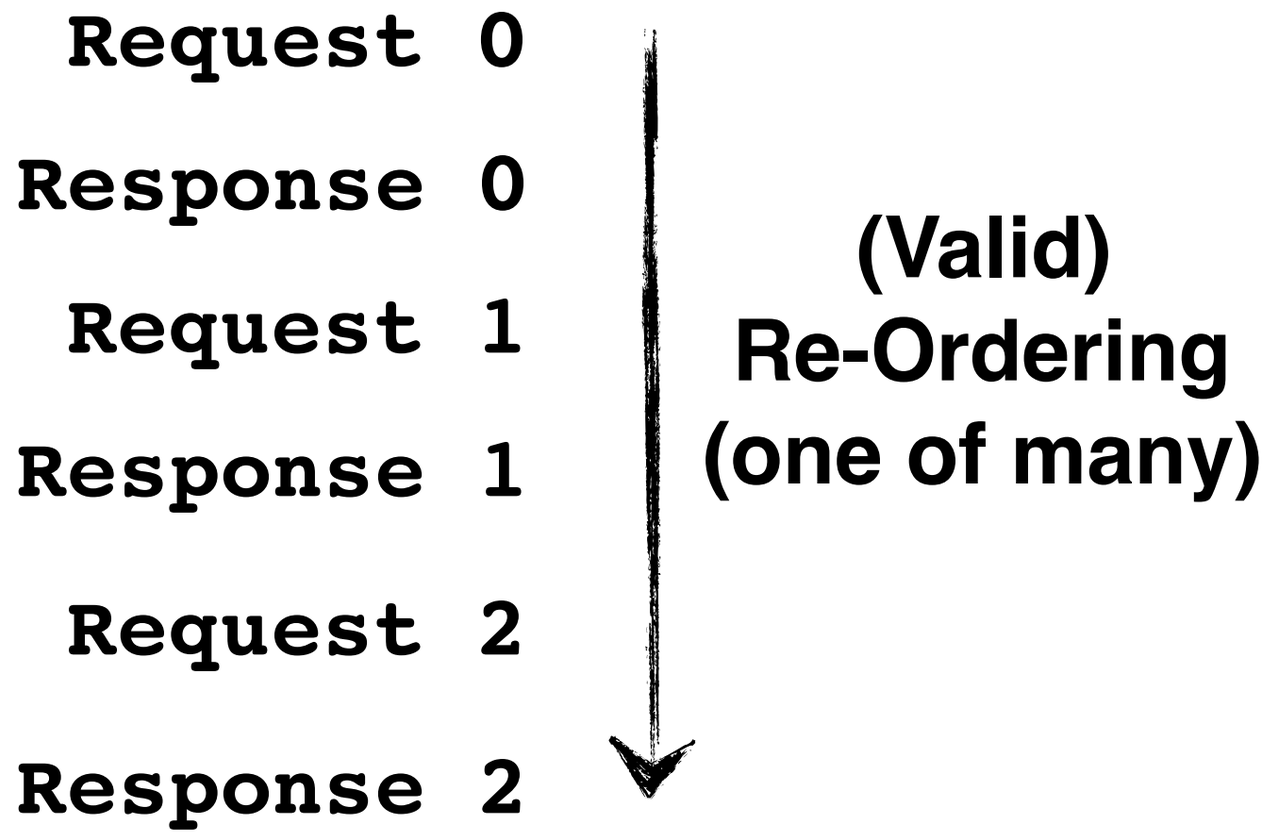


Async Requests & Responses

Correlation!



Correlation!



Correlation!

The key is to wait…

That has a price!

Price of Illusion

- ***Opportunity to De-Schedule***

Price of Illusion

- ***Opportunity to De-Schedule***
- ***Locks + Signaling***

Price of Illusion

- ***Opportunity to De-Schedule***
  - ***Locks + Signaling***
    - ***Semaphores***
    - ***Condition Variables***

Cognitive Dissonance

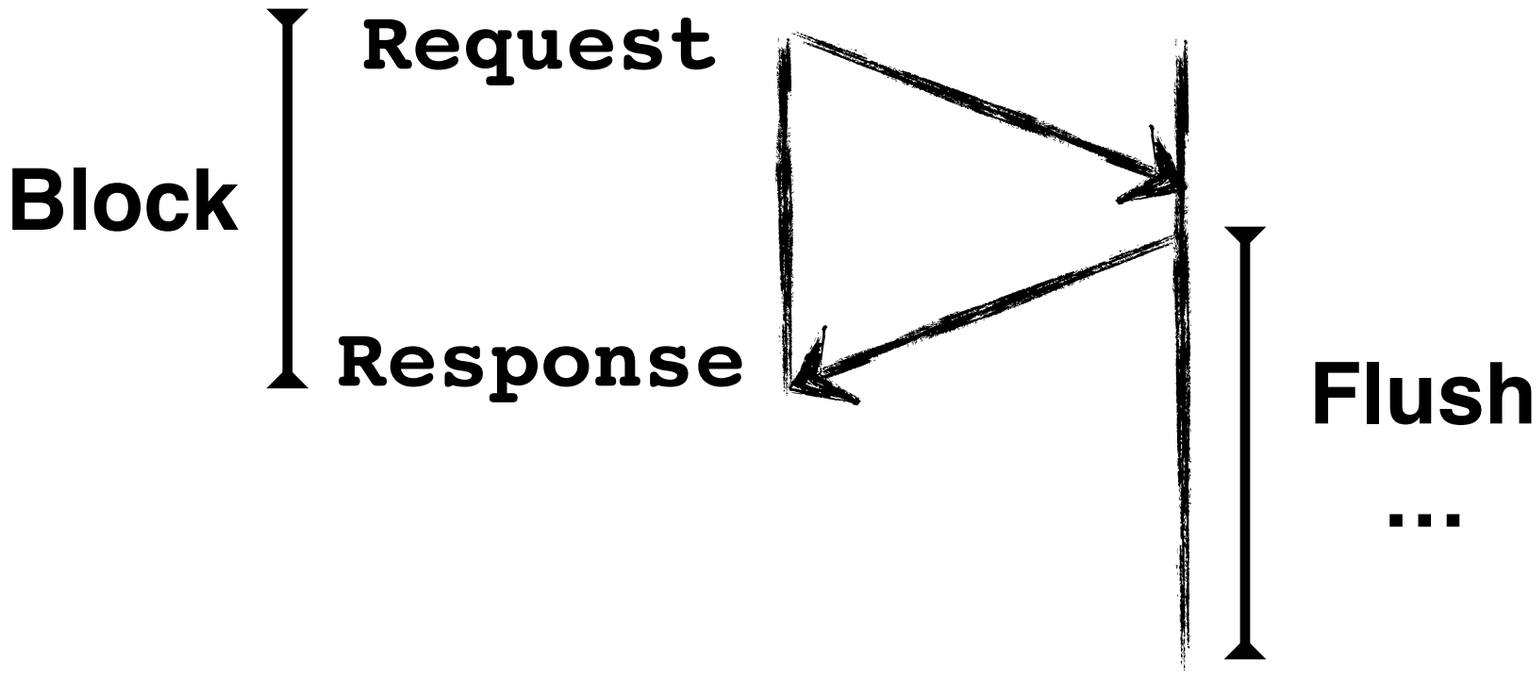
Cognitive Dissonance

- ***Completed Operation Fallacy***

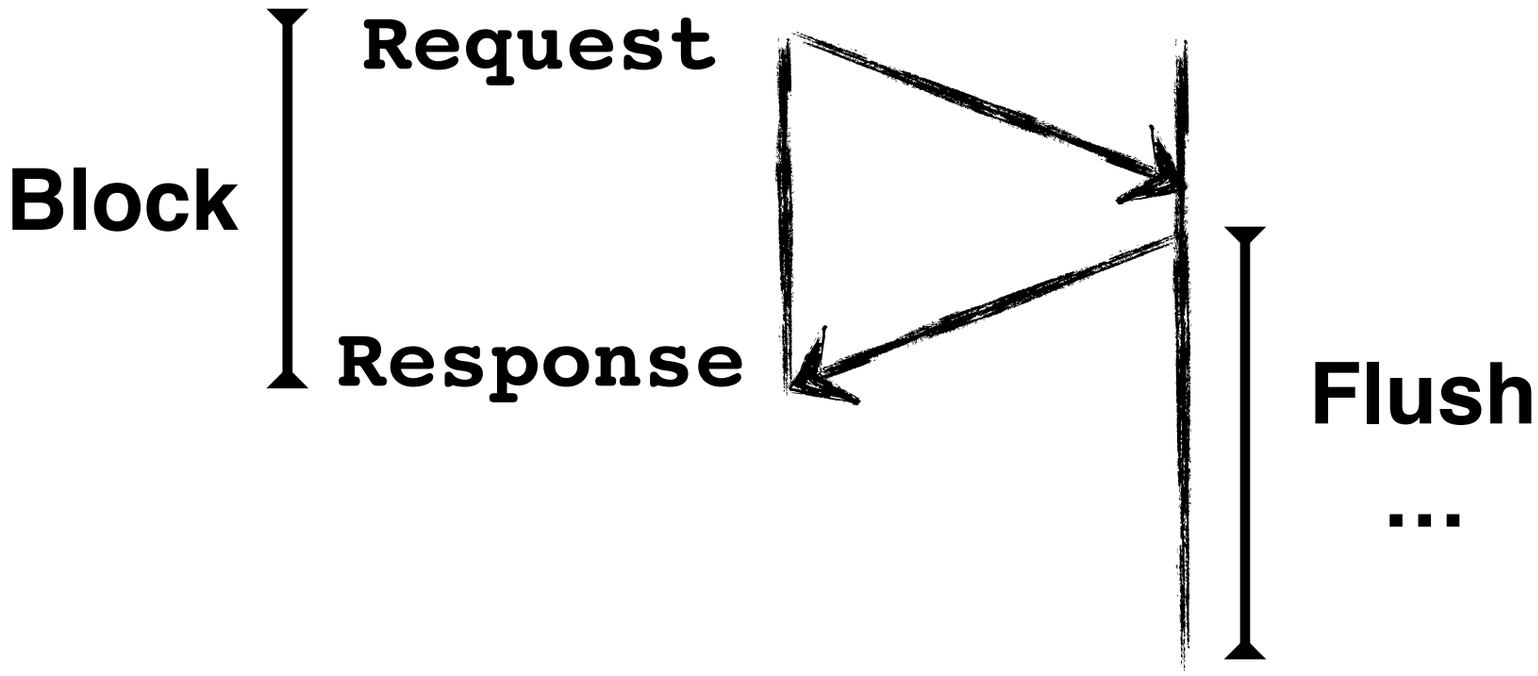
Cognitive Dissonance

- ***Completed Operation Fallacy***
- ***Caching***

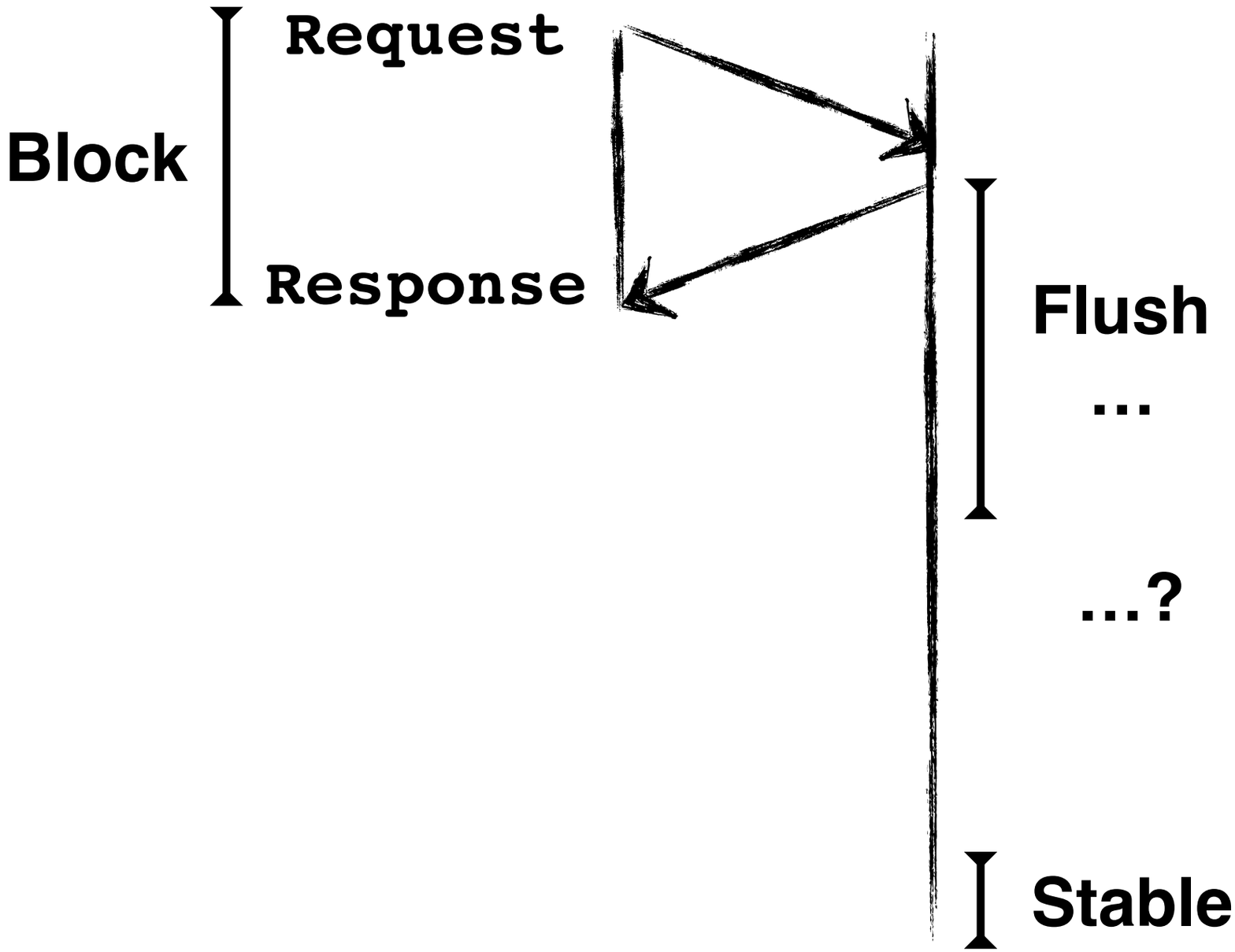
Caches



Caches



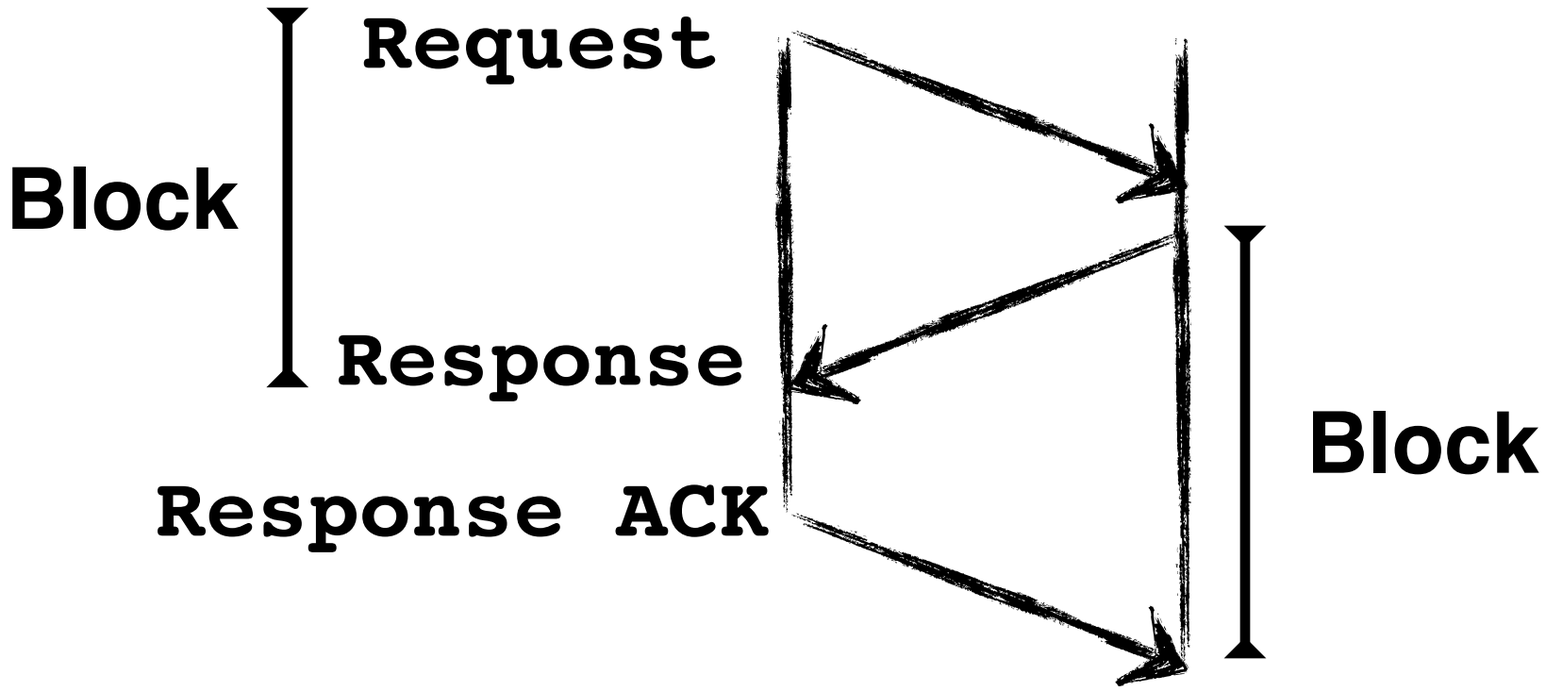
Caches



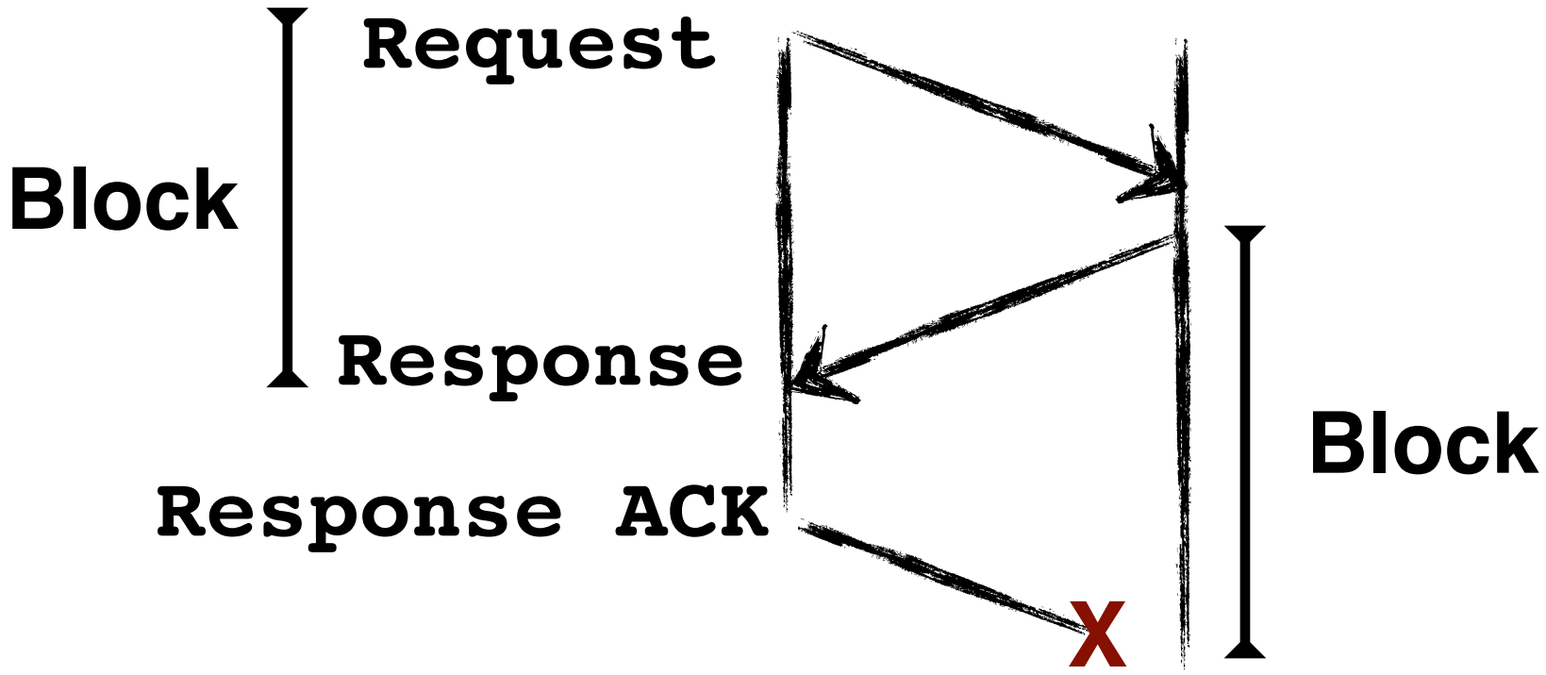
Cognitive Dissonance

- ***Completed Operation Fallacy***
  - ***Caching***
  - ***Blocking ACK Spiral***

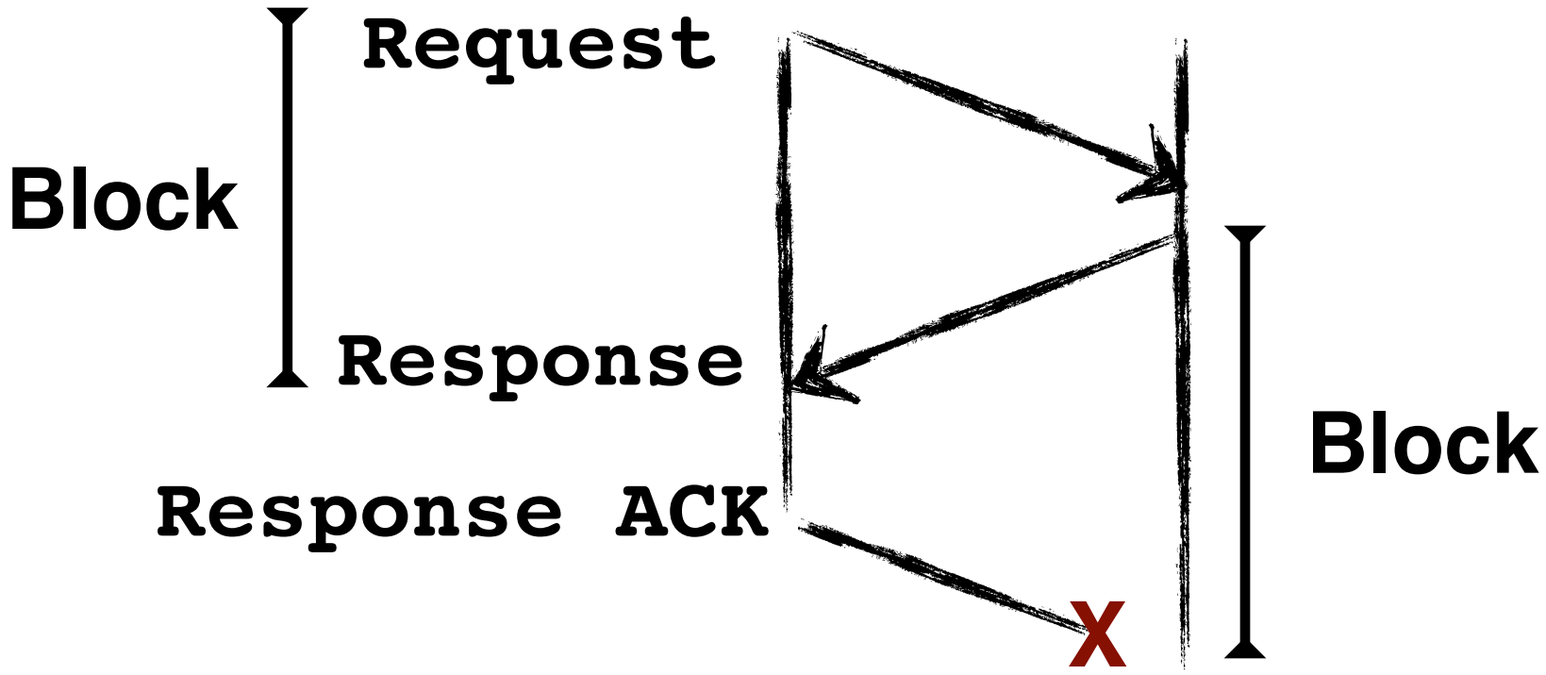
Blocking ACK



Blocking ACK

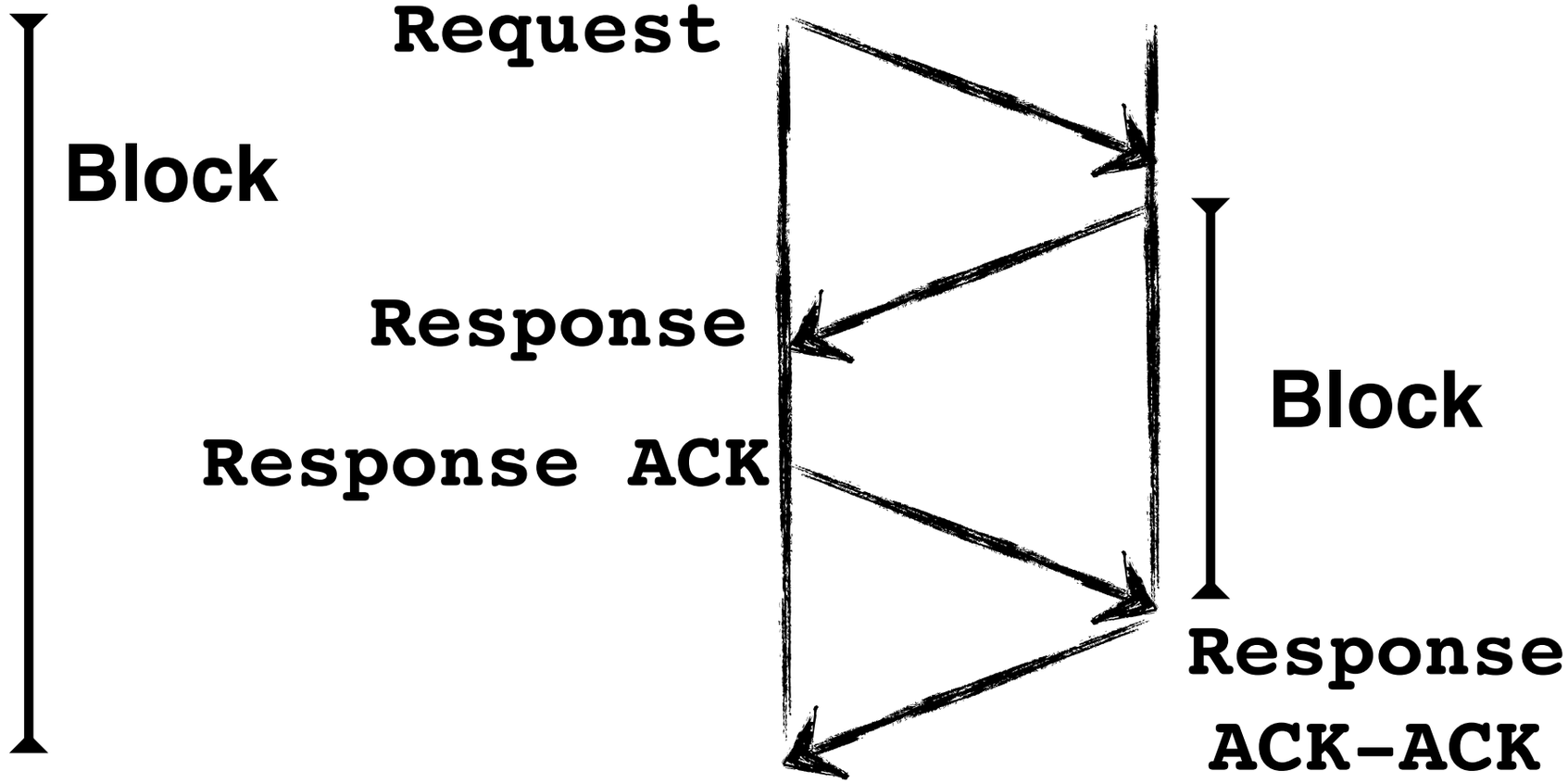


Blocking ACK



Receiver Blocks Forever

Blocking ACK



Spiral!!!

Cognitive Dissonance

- ***Completed Operation Fallacy***
  - ***Caching***
  - ***Blocking ACK Spiral***
- ***Wrong Abstraction***
  - ***Remote Procedure Call***

Abstraction

"The purpose of abstraction is not to be vague, but to create a new semantic level in which one can be absolutely precise"

— Edsger W. Dijkstra
(The Humble Programmer)

Remote Procedure Call

- ***Hiding precision***
- ***Inherent asynchronous nature***
- ***Error handling***

Remote Procedure Call

Don’t assume the network is reliable

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

Remote Procedure Call

"Yeah, yeah, but your scientists were so preoccupied with whether or not they could that they didn't stop to think if they should."

— Jurassic Park

Works sooo poorly, we took it one step further…

REST via HTTP/1.1

- ***Custom Methods***
- ***Custom Response Codes***
- ***No Pipelining***
- ***Everything Request/Response***

REST via HTTP/2

- ***Custom Methods***
- ***Custom Error Codes***
- ***Custom Frame Types***
- ~~***No Pipelining***~~
- ***Mostly Request/Response***

Cognitive Dissonance

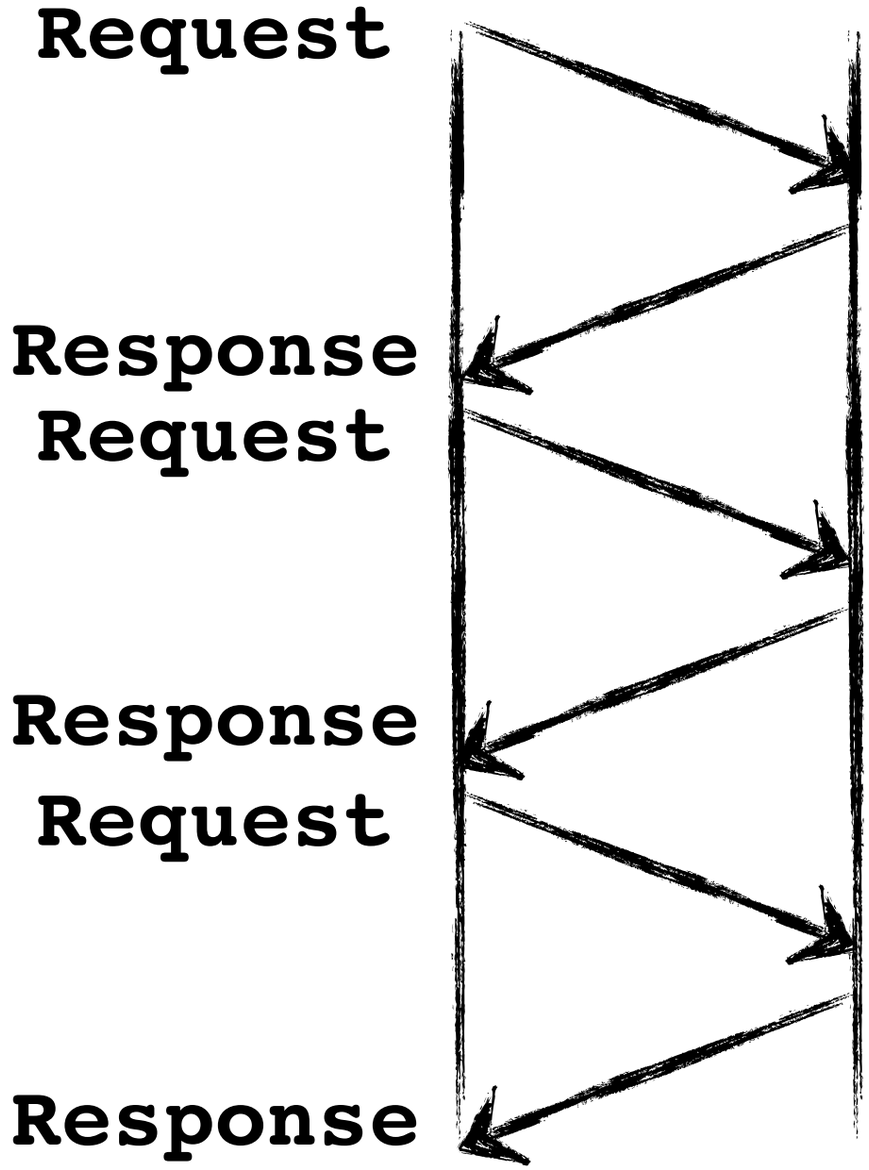
- ***Completed Operation Fallacy***
  - ***Caching***
  - ***Blocking ACK Spiral***
- ***Wrong Abstraction***
  - ***Remote Procedure Call***
- ***Coupling***

Sequential function calls

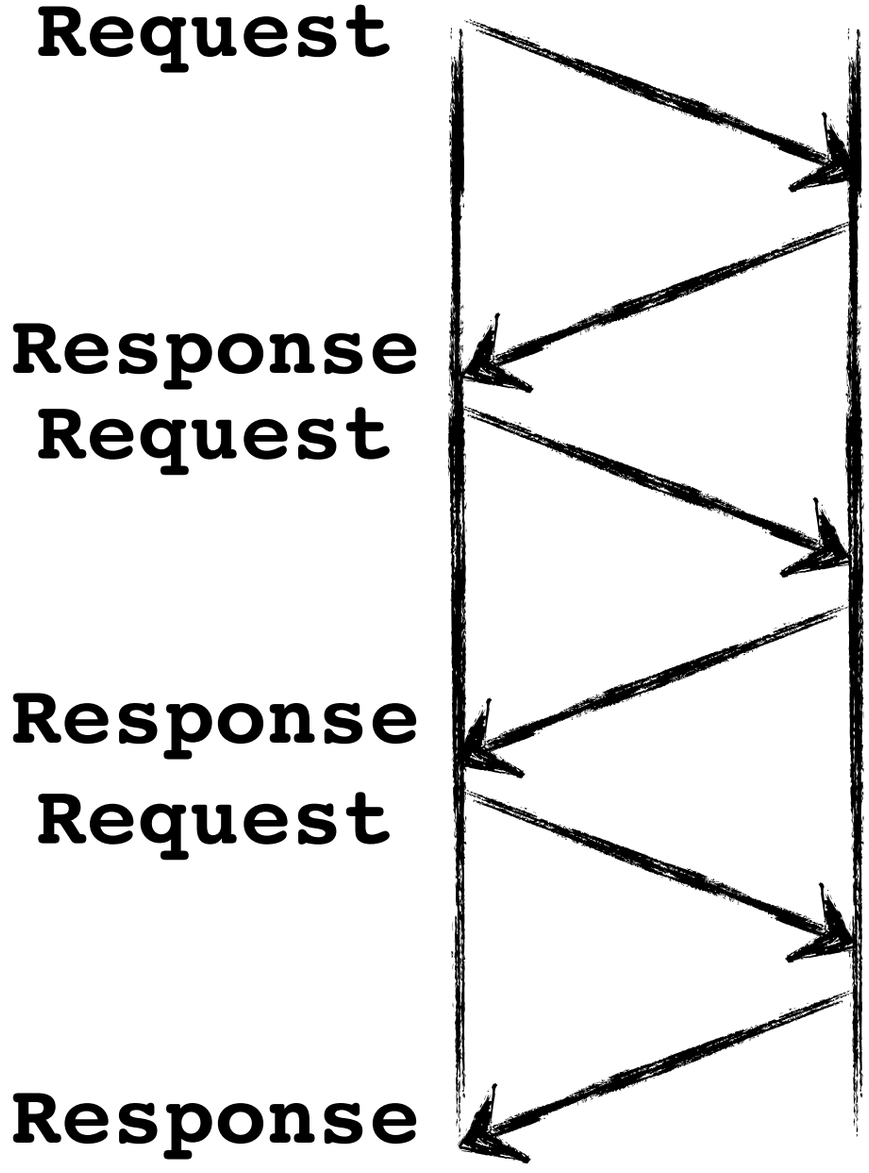
<u>can and do</u>

create Coupling

Impact



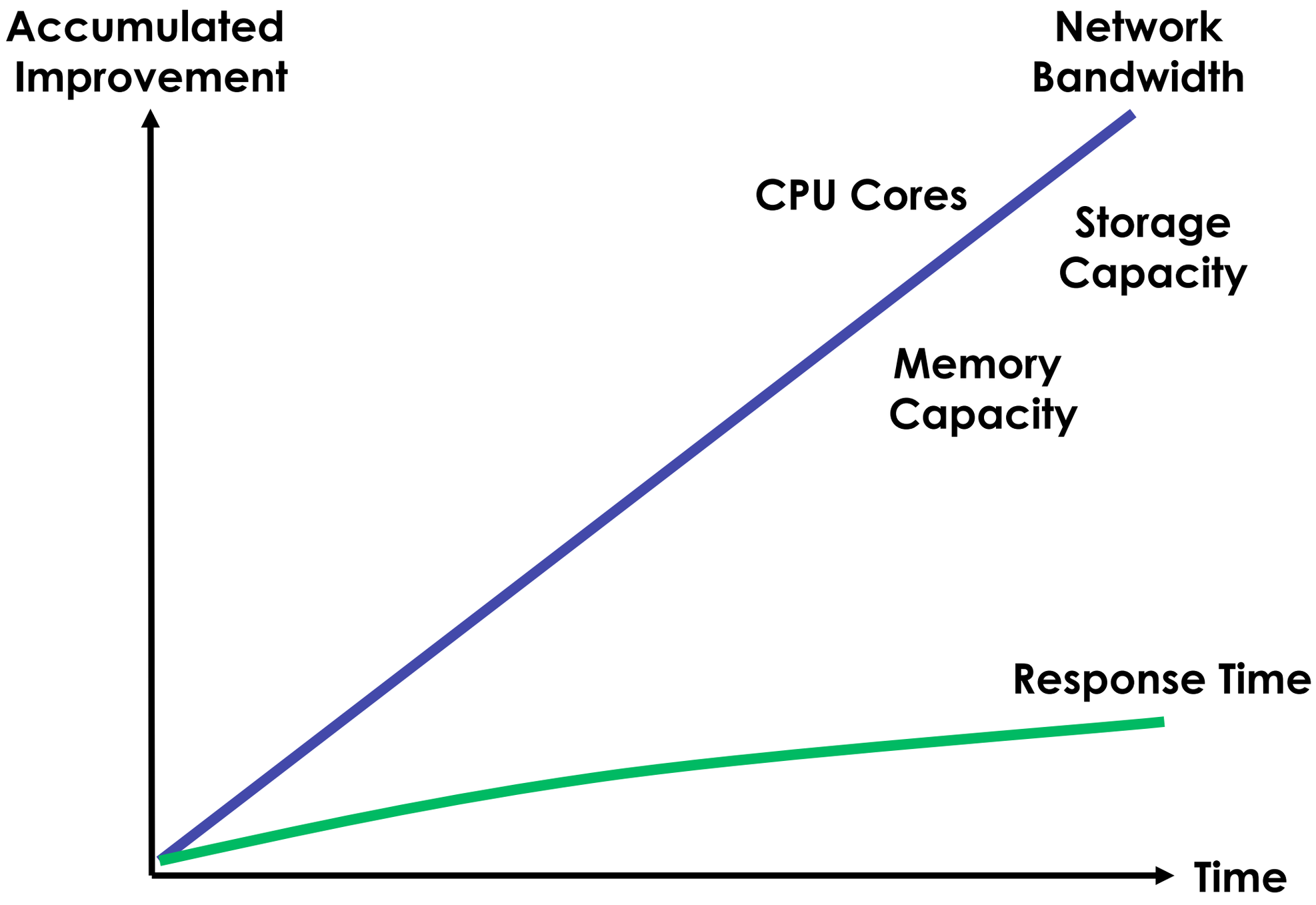
Sync Requests & Responses

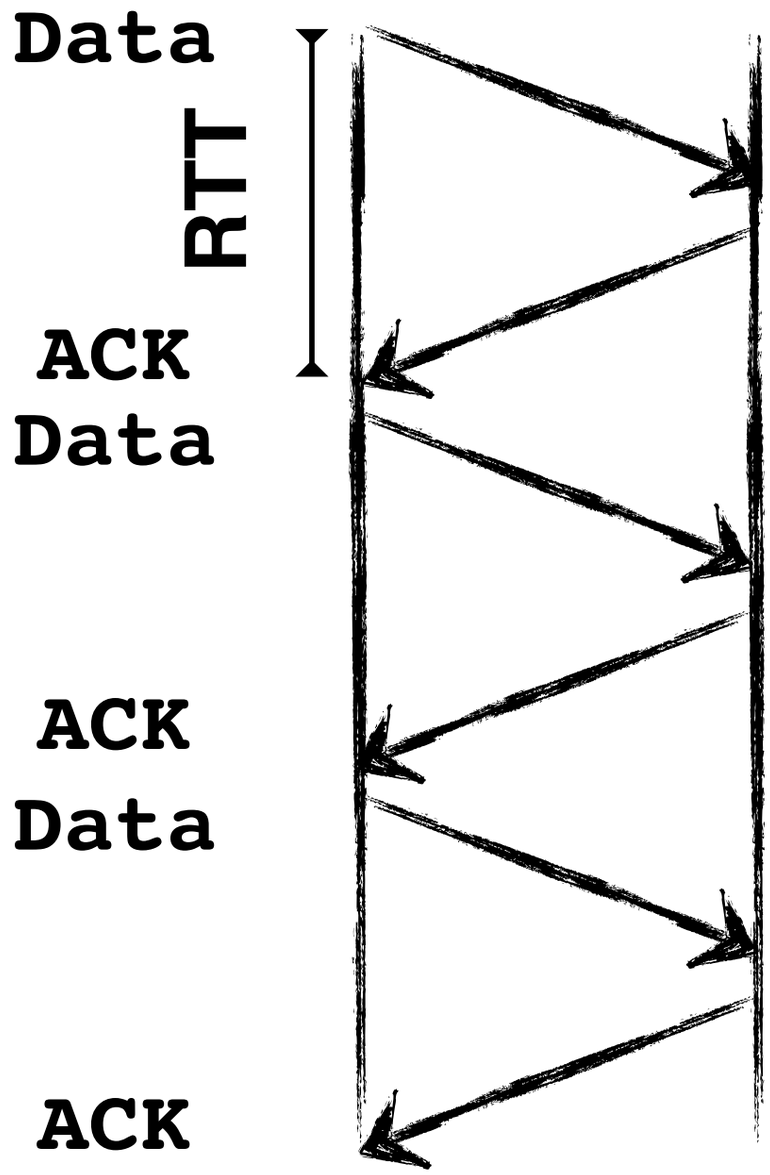


Sync Requests & Responses

Throughput limited by Round-Trip Time (RTT)

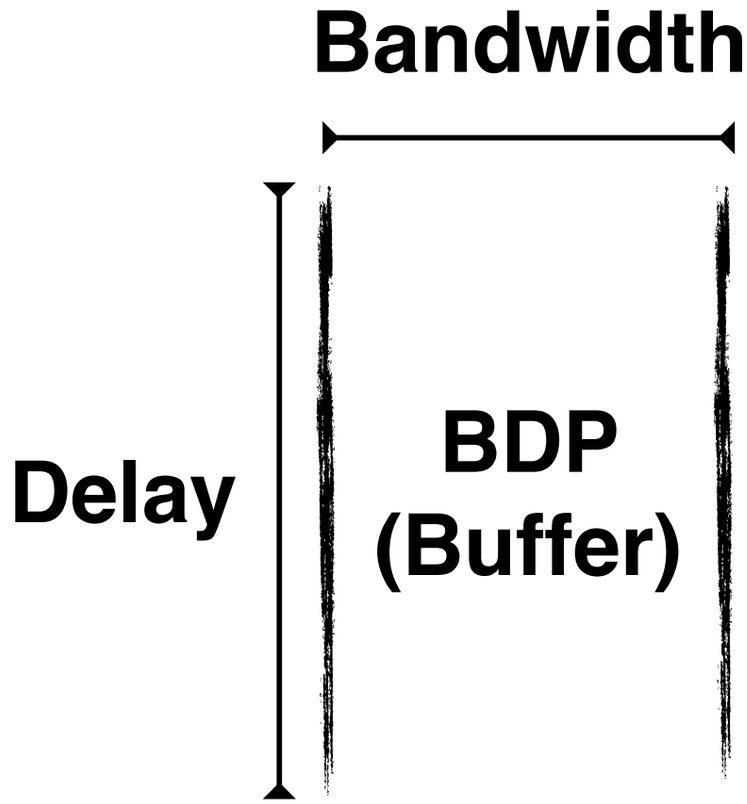
Speed of Light isn't only a good idea, it's the Law


Accumulated Improvement
Network Bandwidth
CPU Cores
Storage Capacity
Memory Capacity
Response Time
Time

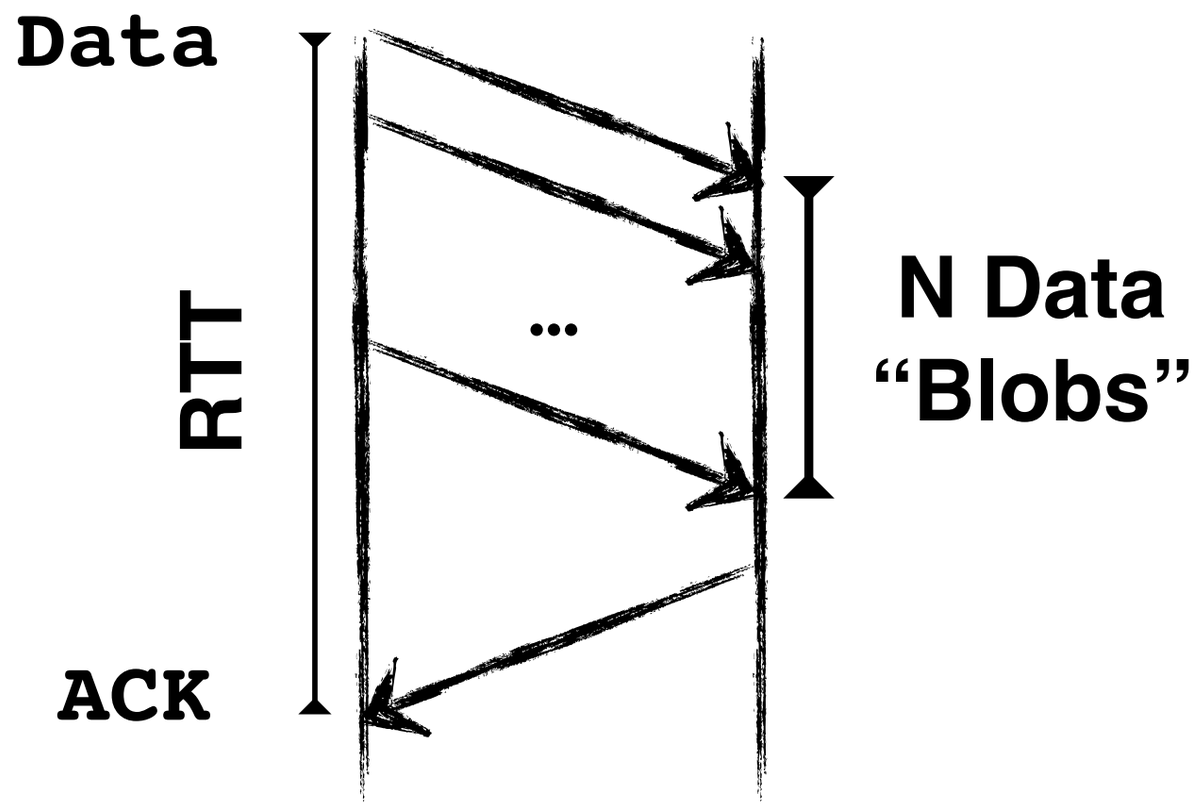


Stop-And-Wait Flow Control

Throughput = Data Length / RTT



BDP = (Byte / sec) * sec = Bytes



Throughput = N * Data Length / RTT

So…

How big is N?

Thread-Per-Request

N = Number of Cores

TCP Flow & Congestion Control

How big is N?

TCP Flow & Congestion Control

How big is N?

It depends…

Big… but

Don’t overflow receiver

Don’t overflow “network”

TCP Flow Control

Receiver advertises N

TCP Congestion Control

Sender probes for network N

TCP BBR Congestion Control

Bottleneck Bandwidth vs. Round-Trip Time

http://queue.acm.org/detail.cfm?id=3022184

TCP Sender

min(Receiver N, Network N)

Only go as fast as Network *&* Receiver

Static N?

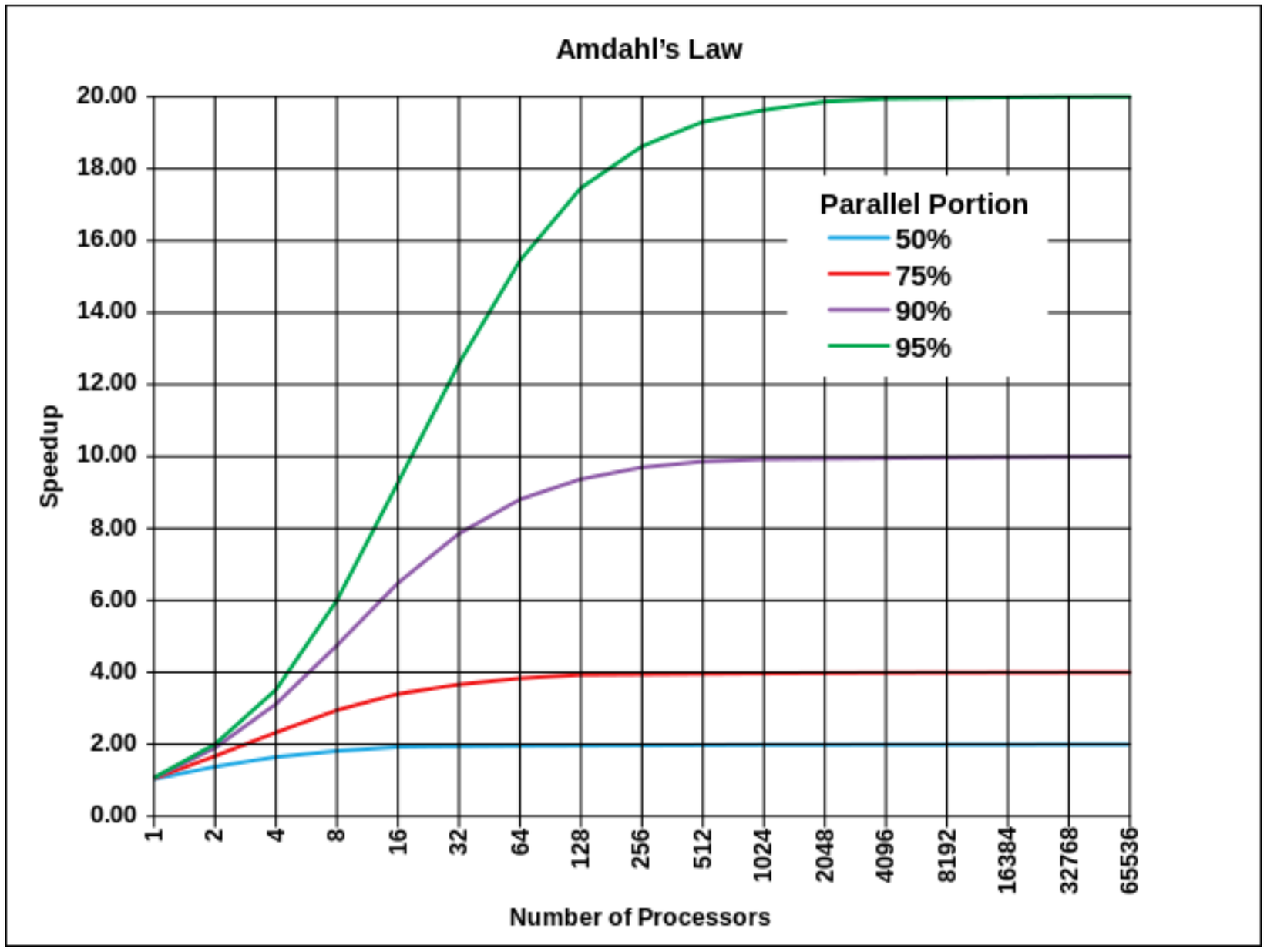
Based on number of cores(threads)?

REALLY?!

But that isn’t the worst…

Locks & Signaling

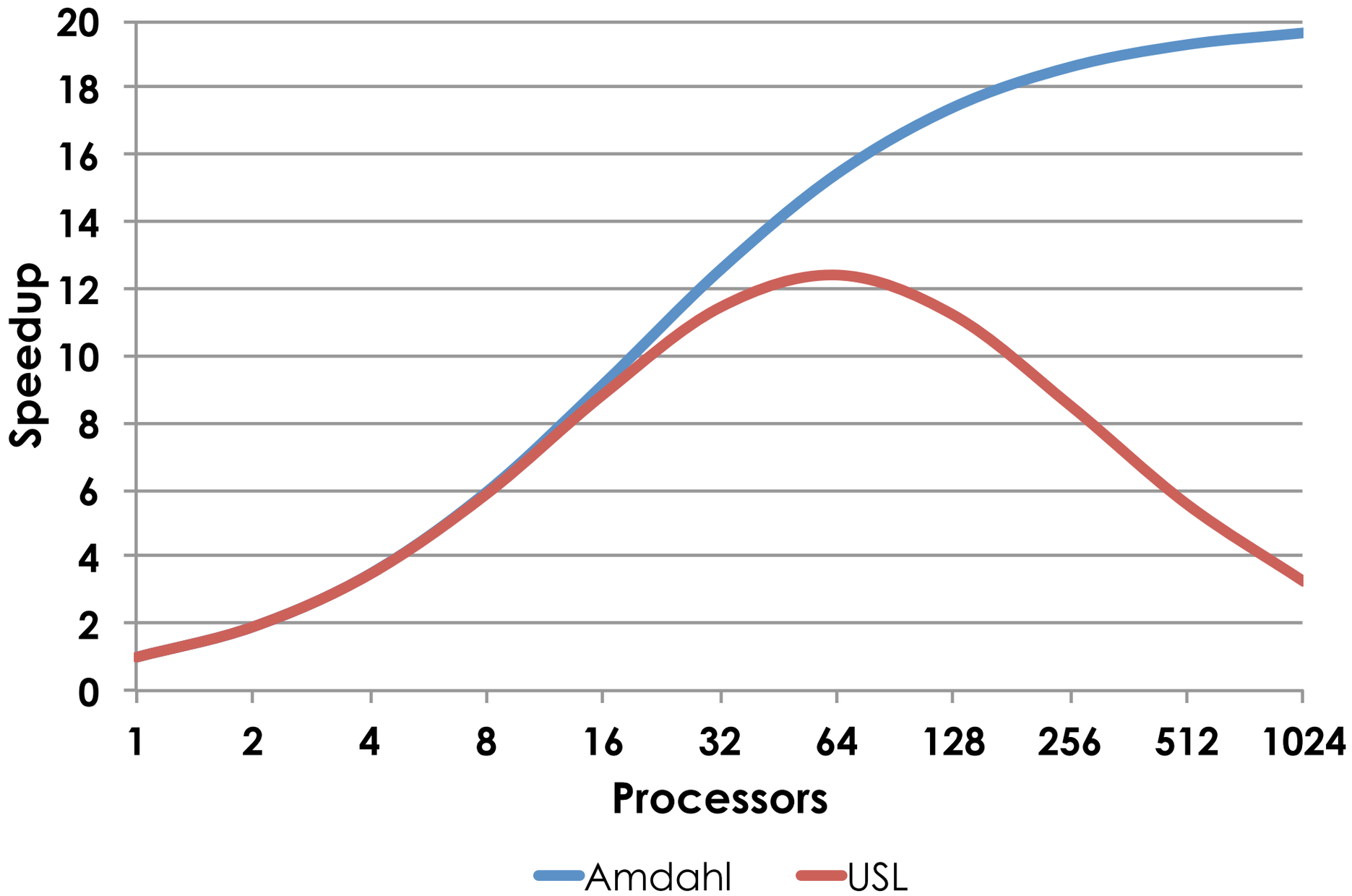
- ***Introduces Serialization***



Locks & Signaling

- ***Introduces Serialization***
- ***Introduces Coherence Penalty***

Universal Scalability Law



Locks & Signaling

- ***Introduces Serialization***
- ***Introduces Coherence Penalty***

Limits Scaling!

1 thread of awesome

>

128 cores of so-so

http://blog.acolyer.org/2015/06/05/scalability-but-at-what-cost/

http://www.frankmcsherry.org/graph/scalability/cost/2015/01/15/COST.html

Async is HARD!!!

Async is HARD!!

- ***Callback Hell***
- ***Back Pressure!***

Composition is hard

ReactiveX

http://reactivex.io/

Observables

JavaScript

- ***RxJS***
- ***ECMAScript Observables***

https://github.com/ReactiveX/RxJS

https://github.com/zenparsing/es-observable

Challenges?

Challenges

- ***Non-Blocking Back Pressure***
- ***Heterogeneous Connectivity***

Dealing with Back Pressure

- ***ReactiveStreams***
- ***RxJava 2.0***

http://www.reactive-streams.org/

Rx Heterogenous Connectivity

- ***ReactiveSocket***

http://reactivesocket.io/

Async is HARD!!

- ***Callback Hell***
- ***Back Pressure!***
- ***Breaking up work units?***

Threaded Work Units

- ***Work between System Calls***

Threaded Work Units

- ***~~Work~~ between System Calls***
- ***Time between System Calls***

High Variance

Async Duty Cycle

- ***Work within a single cycle***

First Class Concern

Async is HARD!!

- ***Callback Hell***
- ***Back Pressure!***
- ***Duty Cycle***
- ***Error Handling***

Error Handling

- ***Errors are events***

No real difference!!

Error Handling

Be Honest…

Takeaways!

Still Think…

Sequential is good enough?

….

Async is complicated & error prone?

Questions?

- **Twitter @toddlmontgomery**

Thank You!