The Tail At Scale

Dean and Barroso, CACM 2013, Pages 74-80

Presented by: Gregory Kesden, CSE-291, Fall 2016

What Is "The Tail"?

- Most requests are answered promptly
- Some requests are slower
- As response times get faster, the difference between the fast and the slow grows
  - And there are many reasons for slow, some simple, and some dynamic and resulting from interactions.
- There is a lot of mass near "fast", but a really long tail from there
- If a request isn't answered promptly, the likelihood of it taking just a little longer and much, much longer get much closer.
  - There is a lot of probability mass across a very thing tail, if it is long enough

Why Variability Exists?

- Shared (local) Resources, e.g. processor, disk, etc
- Background daemons
- Global Resource sharing, e.g. network, DFSs, authentication, etc
- Queueing
  - A dynamic phenomena, especially when multi-level
  - Consider a simple example, such as synchronization at a busy network switch and last vs random drop.
- Burst vs Steady-state processor operation (heat shedding limits)
- Energy saving state transitions
- Garbage collection

Parallelism

- Can hide latency
  - Make multiple queries, take fastest
  - Of course, expensive
- Can make it dramatically worse
  - If need to wait for slowest result
  - Make 1000 queries, depend upon slowest
  - 0.001 can be bad, and the result is bad.

Service Classes

- Interactive before batch requests, etc
- Often means higher-level queuing, rather than relying upon provided queues, e.g. within OS, etc.

Synchronizing Background Activities

- Considering highly parallel operations
  - One slow result slows all, even if "rare"
- Unsynchronized disruptions are constantly disrupting something somewhere
  - And that slows down everything
- Synchronized disruptions affect all systems at the same time
  - Short period of bad response, rather than long period
  - The rest of the time the path is clear
  - Basically stacks the disruptions in a few actions, rather than spreading across many, because the penalty is the same for an action, regardless of how many are concentrated within it.

Work-Arounds: Replication

- Especially for read-only (or read-rare or loosely synchronized), where synchronization isn't an issue
- More throughput, reduces queuing, limits impact of bad node
- Can be selective, e.g. hot items
  - Heat up to high water mark
  - Cool off to low water mark

Work-Arounds: Hedge

- Parallel requests
  - Take first
  - Expensive
- Alternately, reissue only after delay
  - If past the “sweet spot” then ask again
  - 2x sweet spot is better than much of long tail
- Can cancel prior request when issued or when succeeded

Work-Arounds: Micro-partitions

- Breaking up big units of work
  - Avoids head-of-line blocking behind them
  - Enables them to be parallelized
  - Enables finer grained load balancing

Work-Arounds: Probation

- Temporarily exclude poorly responding servers
  - After a certain amount of bad requests
  - Can exponentially back-off, etc
  - Can issue shadow requests to find out when okay again
- Many sources of latency are temporary
  - Daemons, garbage collection, backups, checkpointing and pruning, network storms
  - Just wait for them to pass

Good Enough

- Large Information Retrieval Systems may not need exact answer
  - Top 10 best advertisements? Might 10 of the top 15 due?
  - Just ignore a small fraction of slow responses and move on

Canaries

- Requests which miss caches, etc are not common
- They are more likely to find bad code paths
- Test to a few servers first, then try again to all
- Since canaries are only to a few servers, unlikely to hit server having a bad moment and add significant latency. These aren't scaled.

Going To Get Better? (Naw)

- The faster hardware gets, the greater the variability between it and zero
- Energy efficiency and heat shedding are getting more complex
  - Faster speeds in some cases yield trouble in others
- Scale is growing
- As scale shrinks, variance within same class of devices becomes more significant
- (But, there is some hope, in general, things are getting faster, wider, and more directly connected and parallel)