Instantly share code, notes, and snippets.

jboner / latency.txt

Last active May 23, 2022

Code Revisions 18 Stars 5,000+ Forks 1,825

Latency Numbers Every Programmer Should Know

latency.txt

```
Latency Comparison Numbers (~2012)
----------------------------------
L1 cache reference                           0.5 ns
Branch mispredict                            5   ns
L2 cache reference                           7   ns                      14x L1 cache
Mutex lock/unlock                           25   ns
Main memory reference                      100   ns                      20x L2 cache, 200x L1 cache
Compress 1K bytes with Zippy             3,000   ns        3 us
Send 1K bytes over 1 Gbps network       10,000   ns       10 us
Read 4K randomly from SSD*             150,000   ns      150 us          ~1GB/sec SSD
Read 1 MB sequentially from memory     250,000   ns      250 us
Round trip within same datacenter      500,000   ns      500 us
Read 1 MB sequentially from SSD*     1,000,000   ns    1,000 us    1 ms  ~1GB/sec SSD, 4X memory
Disk seek                           10,000,000   ns   10,000 us   10 ms  20x datacenter roundtrip
Read 1 MB sequentially from disk    20,000,000   ns   20,000 us   20 ms  80x memory, 20X SSD
Send packet CA->Netherlands->CA    150,000,000   ns  150,000 us  150 ms

Notes
-----
1 ns = 10^-9 seconds
1 us = 10^-6 seconds = 1,000 ns
1 ms = 10^-3 seconds = 1,000 us = 1,000,000 ns

Credit
------
By Jeff Dean:               http://research.google.com/people/jeff/
Originally by Peter Norvig: http://norvig.com/21-days.html#answers

Contributions
-------------
'Humanized' comparison:  https://gist.github.com/hellerbarde/2843375
Visual comparison chart: http://i.imgur.com/k0t1e.png
```

Load earlier comments...

AnatoliiStepaniuk commented Dec 25, 2018

Is there any resources when one can test himself with a tasks involving these numbers?
E.g. calculate how much time will it take to read 5Mb from DB in another datacenter and get it back?
That would be a great test of applying those numbers in some real use cases.

bhaavanmerchant commented Dec 26, 2018

I think given increased use of GPUs / TPUs it might be interesting numbers to add here now. Like: 1MB over PCIexpress to GPU memory, Computing 100 prime numbers per core of CPU compared to CPU, reading 1 MB from GPU memory to GPU etc.

sergekukharev commented Jan 11, 2019

Markdown version https://gist.github.com/sergekukharev/ccdd49d23a5078f108175dc71ad3c06c

binbinlau commented Jan 25, 2019

useful information & thanks

bpmf commented Feb 22, 2019 • edited

Some data of the Berkeley interactive version (https://people.eecs.berkeley.edu/~rcs/research/interactive_latency.html) is estimated, eg: 4 μs in 2019 to read 1 MB sequentially from memory; it seems too fast.

speculatrix commented Mar 25, 2019

this is a great idea.
how about the time to complete a DNS request - UDP packet request and response with a DNS server having, say, 1ms response time, with the DNS server being 5ms packet time-of-flight away?

schemacs commented Apr 12, 2019

https://computers-are-fast.github.io

joelkraehemann commented Apr 15, 2019

What effect on latency has the use multiple native threads on doing operations possible due to proper mutex locking? Assumed you have:

- an operation 1024 ns operation in 1st level cache
- 2 x lock unlock mutex (50 ns)
- move it from/to main memory (200 ns)

Now, I wonder about malloc latency, can you tell about it? It is definitely missing because I can compute data without any lock as owning the data.

haai commented Sep 4, 2019

interesting when you see in a glance. but would't it be good to use one unit in the comparison e.g. memory page 4k?

acuariano commented Sep 11, 2019

> Nanoseconds

It's an excellent explanation. I had to search the video because the account was closed. Here's the result I got: https://www.youtube.com/watch?v=9eyFDBPk4Yw

KevinZhou92 commented Jan 30, 2020

`Send 1K bytes over 1 Gbps network 10,000 ns 10 us`
This doesn't look right to me. 1 Gbps = 125, 000 KB/s, the time should be 1 / 125,000 = 8 * 10^-6 seconds which is 8000ns

andaru commented Apr 4, 2020

> `Send 1K bytes over 1 Gbps network 10,000 ns 10 us`
> This doesn't look right to me. 1 Gbps = 125, 000 KB/s, the time should be 1 / 125,000 = 8 * 10^-6 seconds which is 8000ns

For a direct host-to-host connection with 1000BaseT interfaces, a wire latency of 8μs is correct.

However, if the hosts are connected using SGMII, the Serial Gigabit Media Independent Interface, data is 8b10b encoded, meaning 10 bits are sent for every 8 bits of data, leading to a latency of 10μs.

Jeff may also have been referring to the fact that in a large cluster you'll have a few switches between the hosts, so even where 1000BaseT is in use, the added switching latency (even for switches operating in cut-through mode) for, say, 2 switches can approach 2μs.

In any event, the main thing to take away from these numbers are the orders of magnitude differences between latency for various methods of I/O.

moon-chilled commented Apr 27, 2020

Fancy unicode version:

```
Latency Comparison Numbers (~2012)
╭──────────────────────────────────────────────────────────────────────────────────────╮
│L1 cache reference                           0.5 ns
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Branch mispredict                            5   ns
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│L2 cache reference                           7   ns                      14× L1 cache
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Mutex lock/unlock                           25   ns
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Main memory reference                      100   ns                      20× L2 cache, 200× L1 cache
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Compress 1K bytes with Zippy             3,000   ns        3 µs
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Send 1K bytes over 1 Gbps network       10,000   ns       10 µs
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Read 4K randomly from SSD*             150,000   ns      150 µs          ~1GB/sec SSD
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Read 1 MB sequentially from memory     250,000   ns      250 µs
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Round trip within same datacenter      500,000   ns      500 µs
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Read 1 MB sequentially from SSD*     1,000,000   ns    1,000 µs    1 ms  ~1GB/sec SSD, 4× memory
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Disk seek                           10,000,000   ns   10,000 µs   10 ms  20× datacenter roundtrip
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Read 1 MB sequentially from disk    20,000,000   ns   20,000 µs   20 ms  80× memory, 20× SSD
│
├──────────────────────────────────────────────────────────────────────────────────────┤
│Send packet CA→Netherlands→CA      150,000,000   ns  150,000 µs  150 ms
│
╰──────────────────────────────────────────────────────────────────────────────────────╯
```

arunkumaras10 commented May 20, 2020

Are these numbers still relevant in 2020? Or this needs an update?

maning711 commented Jun 9, 2020

> Are these numbers still relevant in 2020? Or this needs an update?

I think hardwares are so expensive that can't update them~

vladimirvs commented Jul 21, 2020

One thing that is misleading is that different units are used for send over 1Gbps versus read 1 MB from RAM. RAM is at least x20 times faster, but it ranks below send over network which is misleading. They should have used the same 1MB for network and RAM.

amresht commented Aug 6, 2020 • edited

> need a solar system type visualization for this, so we can really appreciate the change of scale.

Hi
I liked your request and made an comparison. One unit is Mass of earth not radius.

Operation	Time in Nano Seconds	Astronomical Unit of Weight
L1 cache reference	0.5 ns	1/2 Earth or Five times Mars
Branch mispredict	5 ns	5 Earths
L2 cache reference	7 ns	7 Earths
Mutex lock/unlock	25 ns	Roughly [Uranus +Neptune]
Main memory reference	100 ns	Roughly Saturn + 5 Earths
Compress 1K bytes with Zippy	3,000 ns	10 Jupiters
Send 1K bytes over 1 Gbps network	10,000 ns	20 Times All the Planets of the Solar System
Read 4K randomly from SSD*	150,000 ns	1.6 times Red Dwarf Wolf 359
Read 1 MB sequentially from memory	250,000 ns	Quarter of the Sun
Round trip within same datacenter	500,000 ns	Half of the Mass of Sun
Read 1 MB sequentially from SSD*	1,000,000 ns	Sun
Disk seek	10,000,000 ns	10 Suns
Read 1 MB sequentially from disk	20,000,000 ns	Red Giant R136a2
Send packet CA->Netherlands->CA	150,000,000 ns	An Intermediate Sized Black Hole

https://docs.google.com/spreadsheets/d/13R6JWSUry3-TcCyWPbBhD2PhCeAD4ZSFqDJYS1SxDyc/edit?usp=sharing

asimilon commented Oct 4, 2020

> > need a solar system type visualization for this, so we can really appreciate the change of scale.
>
> Hi
> I liked your request and made an comparison. One unit is Mass of earth not radius.

For me the best way of making this "more human relatable" would be to treat nanoseconds as seconds and then convert the large values.

eg. 150,000,000s = ~4.75 years

sirupsen commented Jan 8, 2021

I've been doing some more work inspired by this, surfacing more numbers, and adding throughput:

https://github.com/sirupsen/napkin-math

sachin-j-joshi commented Mar 28, 2021

Is there a 2021 updated edition?

ellingtonjp commented Apr 15, 2021 • edited

@sirupsen I love your project and I'm signed up for the newsletter. Currently making Anki flashcards :)

There are some large discrepancies between your numbers and the ones found here (not sure where these numbers came from): https://colin-scott.github.io/personal_website/research/interactive_latency.html

I'm curious what's causing them. Specifically, 1MB sequential memory read: **100us** vs **3us**.

sirupsen commented Apr 15, 2021

@ellingtonjp My program is getting ~100 us, and this one says 250 us (from 2012). Lines up to me with some increases in performance since :)
Not sure how you got 3 us

ellingtonjp commented Apr 15, 2021 • edited

@sirupsen I was referring to the numbers here https://colin-scott.github.io/personal_website/research/interactive_latency.html

The 2020 version of "Read 1,000,000 bytes sequentially from memory" shows 3us. Not sure where that comes from though. Yours seems more realistic to me

sirupsen commented Apr 17, 2021 • edited

Ahh, sorry I read your message too quick. Yeah, unclear to me how someone would get 3us. The code I use for this is very simple. It took reading the x86 a few times to ensure that the compiler didn't optimize it out. I do summing, which is one of the lightest workloads you could do in a loop like that. So I think it's quite realistic. Maybe that person's script it was optimized out?

ellingtonjp commented Apr 17, 2021

To everyone interested in numbers like this:

@sirupsen 's project is *really* good. He gave an excellent talk on the "napkin math" skill and has a newsletter with monthly challenges for practicing putting these numbers to use.

Newsletter: https://sirupsen.com/napkin/
Github: https://github.com/sirupsen/napkin-math
Talk: https://www.youtube.com/watch?v=IxkSlnrRFqc

leswaters commented Jun 9, 2021

:)
`Light to reach the moon 2,510,000,000 ns 2,510,000 us 2,510 ms 2.51 s`

invisiblethings commented Nov 24, 2021

Heh, imagine this transposed into human distances.

1ns = 1 step, or 2 feet.

L1 cache reference = reaching 1 foot across your desk to pick something up
Datacentre roundtrip = 94 mile hike.
Internet roundtrip (California to Netherlands) = Walk around the entire earth. Wait! You're not done. Then walk from London, to Havana. Oh, and then to Jacksonville, Florida. Then you're done.

apimaker001 commented Dec 23, 2021

useful information & thanks

eduard93 commented Jan 3, 2022

What about register access timings?

crazydogen commented Apr 6, 2022 • edited

> Markdown version :p

Operation	ns	µs	ms	note
L1 cache reference	0.5 ns			
Branch mispredict	5 ns			
L2 cache reference	7 ns			14x L1 cache
Mutex lock/unlock	25 ns			
Main memory reference	100 ns			20x L2 cache, 200x L1 cache
Compress 1K bytes with Zippy	3,000 ns	3 µs		
Send 1K bytes over 1 Gbps network	10,000 ns	10 µs		
Read 4K randomly from SSD*	150,000 ns	150 µs		~1GB/sec SSD
Read 1 MB sequentially from memory	250,000 ns	250 µs		
Round trip within same datacenter	500,000 ns	500 µs		
Read 1 MB sequentially from SSD*	1,000,000 ns	1,000 µs	1 ms	~1GB/sec SSD, 4X memory
Disk seek	10,000,000 ns	10,000 µs	10 ms	20x datacenter roundtrip
Read 1 MB sequentially from disk	20,000,000 ns	20,000 µs	20 ms	80x memory, 20X SSD
Send packet CA -> Netherlands -> CA	150,000,000 ns	150,000 µs	150 ms	