Calculating the Computational Capacity of a Spherical System

Alfie Ranstead - https://alfieranstead.com - 21/07/2022

Abstract

This paper attempts a very approximate method of constructing a series of rules and an equation that can be used to estimate the maximum computational capacity of a spherical system. One example of such a system would be the observable universe, which is commonly referred to as spherical in shape and hence if organised correctly could be used as a spherical computational system. The model created during this paper will then be used to estimate the computational capacity of various scenarios which will be used to draw a conclusion on the model.

Introduction

There are three key limits to the computational capacity a system that follows the laws of physics can possess, and since the spherical system being theorised is a physical system that follows these laws these are the limits at which this system can process information. However in order to address these limits a few key definitions need to be made:

A tick will be defined as the singular smallest moment within the computational system, this would be one clock cycle in a traditional computer, however due to the nature of this system, it should achieve far far greater computational capacity per cycle than a traditional computer (to the extent at which when talking about simulating a universe one tick would compute one plank second within the simulated universe).

Information is used as any kind of operation code or operation data required for a process or computational method, this could be stored as the quantum spin of a particle or other methods. The term “information” will also be used as in exchange for the term of a “bit” at various points as computational bits aren’t really what is being talked about, rather the smallest quantity which can store a value is.

The First Limit: Operations

The first limit is the speed at which the system can operate. This is the actual speed at which the system “computes” its fundamental operations, in a traditional binary computer this would be operations such as AND, NOT, OR, XOR, etc. Since this computational system shall be quantum in order to reach it’s true capacity there are two key equations for calculating the time taken per operations: $\pi h/2E$ and $\pi h/E$ **[2]**, the former being used for simple

computational operations which contain few quantum flips, and the latter being used for complex computations that contain lots of quantum flips.

As it is not actually known how a computational system this massive would operate and which of the two primary categories it would fall into, I will be assuming that a computational system this large tasked with such a task as simulating a universe or other similarly scaled processes would use the more complex but slower estimate of $\pi h / E_p$.

π being the numerical constant pi ~3.1415;

h being Planck's reduced constant, 1.0545×10^{-34};

E_p being the energy supplied to this operation;

Taking the number of operations that need to be operated per tick to be Iy :

I being the quantity of information that needs to be processed;

y being the number of operations required per piece of information;

The time per operation can then be combined with the number of estimations to create the equation for the amount of time required for all operations to take place per tick $T_o = Iy\pi h / E_o$ where E_o is the total energy supplied to all the operations and T_o is the time taken for all operations per tick.

Typically, it would also be assumed that parallel operations would then lower this quantity of time, but interestingly it actually doesn't. This is because $E_o = \sum E_p$ and by segmenting the operations into n parallel operations, the total Energy is then also segmented into n portions $E_p = E_o / n$, hence the value of energy supplied to the system is divided by n, whilst the value of time per subsection is also divided by n to account for the parallel processing, causing the equation involving n to be $T_o = Iy\pi hn / E_p = Iy\pi hn / E_o n$ which circles back to be $T_o = Iy\pi h / E_o$ .The reason then that parallel processing is typically used is to decrease the quantity of energy at which any part of traditional computer is exposed to so as to prevent connections within chips from melting due to spreading the energy input across a larger volume, but as these calculations already assume that the system is organised perfectly and operated on at an atomical level it can be assumed that such an organisation would allow particles to be independent and retain their approximate positions due to the cohesion of their neighbours or some other method invented by a society great enough to create such a computational system.

The Second Limit: Memory

The second limit is the amount of memory available to the universe. In this case memory means the total number of bits available for a computer system to process, this is the equivalent of RAM in a traditional computer. Modern Quantum computers can store a single bit of memory within the spin of an electron or the spin of a nucleus **[4]**, these are separate methods but as far as I can tell an electron is required to move the data around anyway, so it's safe to say that in this scenario the smallest quantity of particles required to store a bit is a hydrogen atom, containing a singular electron and a singular proton. This then means that continuing this assumption, each pair of electrons and protons dedicated to memory can store one bit of data to be used by the system. Therefore the number of bits of which a system can store can be approximated to be equal to $min(e, p)$, or the minimum value from the subset of quantities of electrons supplied and protons supplied, as each atom will require both an electron and a proton.

The Third Limit: Transferring Information

The third, final and most complex limit is the speed at which information can be moved and therefore transferred. As the computational system being calculated is spherical and it is assumed that the entire system contains useful and constantly used locations at which information may need to be transferred to and from, it is now required to estimate the average distance at which a piece of information will need to travel between ticks. This can be completed via calculating the average distance between two points within a sphere, which requires some calculus.

Interestingly, this problem often arises in nature, one such example being calculating the mean distance between amino acids within proteins, which is where this very brief yet surprisingly well summarised explanation can be found: **[3]**

> *Mean distances between amino acids in proteins*
>
> If proteins are assumed to be spherical and the amino acids are randomly distributed in the sphere the mean distance between two amino acids can be calculated as
>
> $$2\int_0^R\int_0^q \frac{3r^2}{R^3}\frac{3q^2}{R^3}$$
>
> $$\left[\int_0^\pi \frac{\sin\phi}{2}\sqrt{(q-r\cos\phi)^2+(r\sin\phi)^2}\,d\phi\right]drdq = \tfrac{36}{35}R,$$
>
> where R is the radius of the sphere.

Now that an approximate distance for the average journey a piece of information within the system will need to take has been reached, being $36R/35$, it can be used to define the approximate average time a piece of information will be required to travel, this being the distance travelled divided by the speed travelled at.

To actually calculate this value, another assumption needs to be made, that the spherical system is not expanding and is instead constant. This is a rather large assumption, but thinking about the result of this assumption we can realise that assuming the expansion of such a system could result in the ripping apart of computational operators and would vary the radius, hence slowing the system down as it expands.

Using this assumption, the velocity in relation to this path can be calculated in relation to the energy supplied, which assuming this system is constructed within a vacuum is directly calculated using the relativistic kinetic energy equation:

$$E_k = \frac{mc^2}{\sqrt{1-(v/c)^2}} - mc^2 \qquad \text{which becomes} \qquad V = c\sqrt{1-\left(\frac{mc^2}{E_k + mc^2}\right)^2}$$

With: m being the mass; v being velocity; c being speed of light;

The relativistic kinetic energy equation is used instead of the simplified classical $E_k = mv^2/2$ due to the system abiding by the laws of physics, which states the maximum rate at which an individual piece of information can be transferred through a vacuum is limited by the speed of light, defined as 2.9979×10^8 meters/sec **[5]**, also referred to as c, and the relativistic equation prevents any quantity of energy from calculating a velocity greater than this.

It's also important to note how the mass will be calculated for use in this equation, this could be done a variety of ways, but in this paper will be done using Avogadro's constant to 6 d.p. 6.022140×10^{23} **[6]**, the assumption that each bit of information will be transferred and stored using a hydrogen atom **[limit 2]** and that a hydrogen atom has an approximate atomic mass of 1.00797 **[7]**. Using this, the mass in grams of a hydrogen atom can be calculated to be approximately $1.00797/6.022140 \times 10^{23}$, although as the kinetic energy equation requires a value of m in terms of kilograms this must be divided by 1000, which is the equivalent of increasing Avogadro's constant to 6.022140×10^{26}, giving the mass of a hydrogen atom in kg to have an estimate of $\frac{1.00797}{6.022140 \times 10^{26}} \approx 1.673774 \times 10^{-27}$. Hence the value m can be

taken to represent 1.673774×10^{-27} multiplied by the number of bits to be transferred, since this a constant it will be referred to as k for simplicities sake.

Introducing the kinetic energy equation to the distance equation calculated previously creates an estimate for the amount of time each information transfer should take:

$$t = \frac{36R}{35c\sqrt{1 - (\frac{mc^2}{E_k + mc^2})^2}}$$

(Keeping m as m in this version for simplicity.)

Due to the nature of how square numbers work, splitting these transfers into multiple parallel operations does actually linearly increase the time efficiency of this part of the system.
This can be calculated to be the sum of all information transfers (which is equivalent to multiplying the equation by the quantity of transfers) then dividing this value by the number of transfers able to be sent at one time to reduce by the scale concurrent transfers. Using I as the quantity of information to be transferred and b as the “batch size” to be the number of transfers that can take place concurrently.

This gives the final equation for this limit to be:

$$T_a = \frac{36RI}{35bc\sqrt{1 - (\frac{kIc^2}{E + kIc^2})^2}}$$

Putting These Limits Together

Based upon these limits a summarised estimate for the speed at which a spherical system computes can be created, through connecting the time at which all the operations per cycle take to occur [limit 1], to the time at which is required for all the information required to travel to it's required destination [limit 3] and how much information can actually be stored by the system [limit 2].

To combine limit 1 and 3 is actually pretty trivial, all that is required is to add the equations together to get the time taken per cycle as: $T_c = T_a + T_o$, however it is also important to note that both of these equations main inner limits is by the energy supplied, hence the energy requirements must be given separate identities such that the ratio of energy supplements can be explored separately and not locked into a 50/50 split. Therefore to represent the separate energy splits it should be noted that $E = E_o + E_a$: where E is the total energy supplied to the system; E_o is the energy supplied to the operations; and E_a is the energy supplied to the transfer of information.

Bringing limit 2 into this, a ratio also needs to be attached for the quantity of bit stores used for storing data between cycles and the quantity used for transferring information during cycles, although it can be assumed that between cycles the information transfers also retain their values so as to be used in the next cycle, some information is likely also required to be stored during a cycle and not processed - however this quantity will vary across the whatever the system is being used for - so it can be defined that $I = I_c + I_s$ where $I = min(e, p)$; I_c is the number of information stores used in a cycle for data transfer; and I_s is the number of information stores used for data storage during and between cycles.

I_s will not be mentioned again since it does not affect the operational speed, and is only important if the number of total bits is set, since I_s would be "sat aside" during an operation.

This gives:

$$T_c = \frac{36RI_c}{35bc\sqrt{1 - (\frac{kI_c c^2}{E_a + kI_c c^2})^2}} + I_c y\pi h / E_o$$

(See individual limits for explanations on individual components and their representations)

Bearing in mind which values are set constants using the set $[\pi, h, k, c]$ and which are parameters to be set by the calculator $[R, I_C, b, E_{[o,a]}, y]$, this equation can be used to

generate some interesting estimations for the time required to complete a set of operations within a spherical system, and hence calculate the computational capacity of such a system.

Examining the Model's Estimations Using Data

Setting the Values of each Variable

The set of data that will be fed into the model is that of a computational device of similar size to a standard desktop computer, although it will obviously be spherical, this is so that its results can be compared to real-world computational devices and hence see if the estimations given seem reasonable. In order to use the model the parameter set needs a complete set of values, this set being $[R, I_C, b, E_{[o,a]}, y]$, although all of these values won't have specific values but can instead be fed continuous data in order to receive a continuous output and hence study their affects on the system as a whole. In the scenario of modelling a spherical desktop computer, however, there is a value that will be set as a constant (discrete) value, that being the radius of the system, like all the other values it could be continuous, however it must remain within a reasonable range to continue being classified as a "Desktop sized" computer.

Hence the radius (R) of the system will be defined as a value approximately within the range of 0.03 to 0.10m, simply because this device would need to be contained within a shell of some sort and these values are roughly a bit below what most people would classify as a desktop computer (it's important to remember this is the radius of a sphere, and therefore the diameter is twice these values). For simplicities sake the data fed to the model will have a constant radius of 0.05m, giving the device a 10cm diameter, which paired with a shell of width 3-5cm would give the device a fairly reasonable total diameter of 16-20cm.

Then for this model the number of bits to be used as memory can be set to an amount higher than an average workstation as these bits are also used as the bus bits alongside purely storage based bits, for example a reasonable estimate for a suitable range could be 128 to 1024 Gigabytes, or approx $I_c : 1 \times 10^{12} < I_c < 8.2 \times 10^{12}$, hence for a base value 4×10^{12} will be used as I_c .

Assuming that the buses in this model are large, the batch size b can be estimated to be between a tenth and a twentieth of the value of I_c such that an approximate base value for this can be 4×10^{11}, this isn't quite as an exact a value as the previous two parameters as it's quite complex to estimate and to do so would be outside the scale of this paper.

The ratios between $E_{[o,a]}$ are likely to have a profound yet non-linear affect on the time output of the estimation model therefore prior to any kind of calculations I will be just setting this ratio to 1:1 and can then vary this whilst graphing the results to see it's effects visibly. The total value of E can be set fairly easily as it can basically be any real, positive value, so for this estimation it will be set to a 10000 Joule input, purely because it's large enough to hopefully display some variation in the ideal ratios between $E_{[o,a]}$ whilst still being a value that could probably be supplied to the system by the time humans get anywhere near to this quantity of computing power.

The number of operations required per bit, y, only affects the output by a scale and average gradient of the output, meaning varying it would create some slightly different outputs, but the general shape of the graph would remain fairly similar, hence it will be set to 1 for this simulation.

Summarising these values

The variable set of inputs is then:

- R - Radius of system - $\{R : 0.03 < 0.05 < 0.10\}$
- I_c - Bits to be used as memory - $\{I_c : 1 \times 10^{12} < 4 \times 10^{12} < 8.2 \times 10^{12}\}$
- b - Number of bits that can be simultaneously transferred - $\{b : I_c/20 < 4 \times 10^{11} < I_c/10\}$
- E - Total Energy of system - $\{E : 1 < 10000 < 1 \times 10^{8}\}$
- $(E_o : E_a)$ - Ratio of movement energy to operational energy- $\{(E_o : E_a) : (100 : 1) < (1 : 1) < (1 : 100)\}$
- y - Operations required by each bit - {1}

(Midpoints of inequalities are the default values to be used in this simulation)

With the constants being:

- c - Speed of light $\approx 2.9979 \times 10^{8}$
- π - pi ≈ 3.1415
- h - Planck's reduced constant $\approx 1.0545 \times 10^{-34}$
- k - the approximate mass of a hydrogen atom in kilograms $\approx 1.673774 \times 10^{-27}$

Graphing this Data

Ideally, each value would be varied in its own graph, varying between the extremes of its range to see how it truly affects the final model estimation. However if that was done, more of this paper would be graphs than the actual model itself, hence the two variables which shall be varied and graphed shall be the total energy supplied to the simulation and the ratios between the two energy inputs.

First, the total energy, to graph this the value X will represent the total energy, using a ratio of 1:1 between energy inputs, making the line to be modelled:

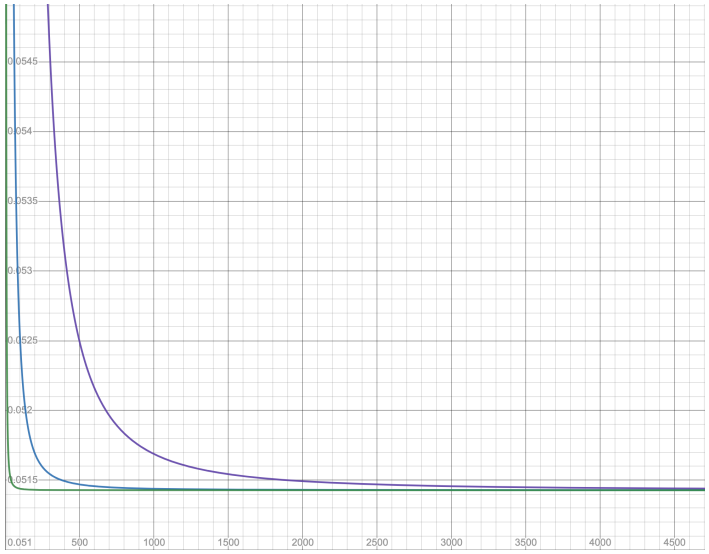
$$y = \frac{36 \times (0.05) \times (4 \times 10^{12})}{35 \times (4 \times 10^{12}) \times \sqrt{1 - (\frac{(1.673774 \times 10^{-27}) \times (4 \times 10^{12}) \times (2.9979 \times 10^{8})^2}{\frac{x}{2} + ((1.673774 \times 10^{-27}) \times (4 \times 10^{12}) \times (2.9979 \times 10^{8})^2)})^2}} + ((4 \times 10^{12}) \times 1 \times (3.1415) \times (1.0545 \times 10^{-34}) \times \frac{2}{x})$$

$$y = \frac{7.2 \times 10^{-2}}{1.4\sqrt{1 - (\frac{601.7153531577336}{\frac{x}{2} \times 601.7153531577336})^2}} + \frac{(2.65017 \times 10^{-21})}{x}$$

Plotting this line shows the above graph, bearing in mind that for this case y is the time required to complete a complex computation approximated in the value summarisations and x is the energy input to such a system. Knowing this gives a somewhat expected but still very interesting summary, that the model has effectively theorised that initially as more energy is added to a system it gets exponentially faster (shown by time per operation, y, decreasing dramatically) before tapering off and flattening out. The exact values at which this occurs is not hugely significant as this estimation is based upon semi-random inputs but the trend still shows that in terms of energy, feeding more and more energy into a system gives diminishing benefits, which can be shown in the current chip industry with processors getting smaller and more densely packed with transistors rather than just putting more and more power into similar density chips.

It is also important to remember that although this simulation uses a “desktop” sized computer, it still assumes that the majority of the computer would be the processing unit which in traditional computers is not the case. Additionally it is assumed that the device is completely efficient (which in real world situations is effectively impossible), alongside various other assumptions which *shouldn’t* effect the data trends too much but would affect the actual value output.

Introducing two more lines with varying ratios of energy given to transporting the information and actually operating on it shows how although this can greatly affect the time required per operation on smaller energy inputs, yet once these energy inputs get larger the results all converge on the same maximum time efficiency.



In this case the green line is the same 1:1 ratio shown in the previous graph, with the blue line then being a 1:19 ratio ($E_a : E_o$ - Transfer Energy : Operational Energy) and the purple line being a 1:99 ratio. This suggests that the part of the computation that requires the most energy is not the operations, but moving the atoms needed for those operations to wherever they need to go. Which taking a logical approach to actually makes a lot of sense, as quantum operations occur on such a small scale that the energy required per operation makes sense to be lower than moving an atom by over a million times the width of itself.

This could then suggest that by compressing the computational system into a smaller volume yet keeping the same amount of particles and bits so as to complete the same computation the energy required to complete that same operation would decrease in a proportionate fashion, which can once again be linked to the computer processing industry, which is indeed compressing processing units into more and more dense packages and is seeing a proportional increase in power efficiency (and increased computational power due to keeping chip sizes somewhat constant whilst increasing density). Although to ensure the model does accurately simulate this it would require more analysis with more graphs, as this suggestion is just based upon the data collected in the graphs above and expectations based upon the model's equational form and is not certain.

Conclusion

Taking a spherical computational system which is assumed to have all its parts equally distributed and valued across itself, the following model gives an estimation on the time required to complete a complex quantum computation:

$$T_c = \frac{36RI_c}{35bc\sqrt{1-(\frac{kI_cc^2}{E_a+kI_cc^2})^2}} + I_cy\pi h/E_o$$

This model can then be used to study the relationship between different aspects of a computational system and the time taken per complex computation such as the relationship between energy input and time taken, or any other variable parameter from the set $[R, I_C, b, E_{[o,a]}, y]$.

Although the model has not been examined in detail and is unlikely to give precise, definitive, values for a set of parameters; based upon limited analysis it appears to provides somewhat valuable and approximately accurate trends for continuous parameters. In some ways this is actually more useful for this kind of model, as it can easily and clearly show how each parameter affects the expected output and with a lot more refinement - to the point at which the model could be trusted to give consistent accurate estimations - could then be used to learn which aspect of a computational system should be improved to gain the most computational capacity for the smallest price (and a myriad of other things).

To conclude, the model is useful for basic trend analysis and with additional work and calibration could be useful for a myriad of reasons, including: progressing the computational capacity of processors; increasing energy efficiency within a processor and studying the effect of the density of a system on it's processing power.

References

1) S. Lloyd, “Computational Capacity of the Universe,” *Physical Review Letters*, vol. 88, no. 23, May 2002, doi: 10.1103/physrevlett.88.237901.

2) S. Lloyd, “Physical Limits to Computation,” in *Quantum Communication, Computing, and Measurement 3*, MIT, Cambridge: MIT Department of Mechanical Engineering, 2001, pp. 189–198. doi: 10.1007/0-306-47114-0_30.

3) O. Lund *et al.*, “Protein distance constraints predicted by neural networks and probability density functions,” *Protein Engineering Design and Selection*, vol. 10, no. 11, pp. 1241–1248, Nov. 1997, doi: 10.1093/protein/10.11.1241.

4) J. J. L. Morton *et al.*, “Solid-state quantum memory using the 31P nuclear spin,” *Nature*, vol. 455, no. 7216, pp. 1085–1088, Oct. 2018, doi: 10.1038/nature07295.

5) T. Blaney *et al.*, “Measurement of the speed of light Accurate wavelength measurement on up-converted CO 2 laser radiation,” *Phys. Rev. Lett*, vol. 251, p. 215, 1972, Accessed: Jul. 10, 2022. [Online]. Available: https://www.nature.com/articles/251046a0.pdf

6) Y. Azuma *et al.*, “Improved measurement results for the Avogadro constant using a 28 Si-enriched crystal,” Mar. 2015. Accessed: Jul. 10, 2022. [Online]. Available: https://arxiv.org/pdf/1512.05642.pdf

7) Angelo State University, “The Parts of the Periodic Table,” *Angelo.edu*, 2019. https://www.angelo.edu/faculty/kboudrea/periodic/structure_mass.htm