As Time Goes By: The Law of Cloud Response Time

Working Paper, April 12, 2011

Joe Weinman[1]

Permalink: http://www.JoeWeinman.com/Resources/Joe_Weinman_As_Time_Goes_By.pdf

Abstract

We propose a *Law of Cloud Response Time* that combines network latency and parallel processing speed-up in a distributed, elastic, cloud computing environment. As the first supercomputing and parallel processing systems came into existence in the 1960s, Gene Amdahl proposed "Amdahl's Law:" the maximum possible speedup due to parallelization is $1/S$, where S is the sequential percentage of the application. Thus, at least for somewhat parallelizable applications, more processors mean less elapsed time, but there is a limit to the gains as no acceleration can occur in the serial portion of the application. However, today's geographically dispersed cloud environments comprising networked nodes of elastic resources are very different than the local, monolithic, centralized environments of a half century ago, so we propose a new law for interactive transactions over a network with parallelization:

$$T = F + \frac{N}{\sqrt{n}} + \frac{P}{p}$$

Simply put, the total response time T for an interactive networked application in which a client application requests and receives an interactive response from a cloud application over a network is a function of three components. F is a fixed interval that can't be accelerated, N is the worst case round-trip latency for an environment with a single node, n is the number of (evenly-distributed) processing nodes, P is the time for the parallelizable portion of the application to run on one processor, and p is the number of processors.

For example, consider a search query requested via an end-user client. A time F is needed for client processing and other serial tasks; sharding and parallelization can reduce processing time via the P/p component; and replicating this service in n multiple physical locations can lower the network latency, reducing the $N/\sqrt{n}$ component. In fact, if Q processors are available for deployment, the optimum latency is reached when the number of nodes is $n = \sqrt[3]{(\frac{QN}{2P})^2}$.

My 7th Law of Cloudonomics states: "Space-Time is a Continuum," i.e., more processors can mean less time, and my 8th states that "Dispersion is the Inverse Square of Latency," i.e., halving latency requires quadrupling the number of service nodes. Importantly, pay-per-use Cloud Computing services dramatically enhance these economics, as parallelization can be implemented and dispersed resources can be shared at zero marginal cost.

---

[1] Joe Weinman leads Communications, Media and Entertainment Industry Solutions for Hewlett-Packard. The views expressed herein are his own. Contact information is at http://www.joeweinman.com/contact.htm

1. Introduction

In the world of the web, time is money. Or more precisely, time costs money, since accelerating site performance has been shown to generate greater revenue, and reducing site responsiveness has been shown to reduce revenue. In situations with winner-take-all dynamics, for example the 2008 Beijing Olympics, a few milliseconds made all the difference in Michael Phelps winning a record number of gold medals in a single Olympics—beating Mark Spitz' seven golds—as he finished the 100 meter Butterfly in 50.58 seconds, narrowly beating Milorad Cavic, who finished in 50.59 seconds.[2] Such competitions exist in the world of computing as well, for example in equity trading, where NYSE Euronext CIO Steve Rubinox, in commenting on sub-millisecond performance, pointed out that there is a "world of difference" between one hundred and nine hundred microseconds.[3]

There are approaches to making web sites faster which involve such strategies as reducing file size, for example, pre-optimizing images, rather than sending a large file and then using the HTML <img> tag *height* and *width* attributes to reduce the size once it arrives at the client. Another tactic is to remove JavaScript comments, which, after all, don't get seen by the end-user. Or, one might use the more efficient and thus valuable JavaScript Object Notation (JSON) rather than the voluble eXtensible Markup Language (XML), shaving a few bytes here and there. Using shorter variable names and document object ID's might even help: why use "user" when "usr" will work just as well?

While all these techniques no doubt help, at least marginally, they miss the point in a fundamental sense: paradoxically, online experiences certainly can be enhanced by clarity and simplicity, but they can also be enhanced by richness. Improving the performance of a video sharing site by only showing low resolution videos in black and white is unlikely to lead to competitive success for the site. Consequently, as the web enters its third decade, we must look for other ways to enhance performance than slimming down the user's experience. In effect, we need to be able to do and deliver more stuff in less time.

Given the importance of time, in this paper we combine my 7th and 8th Laws of Cloudonomics[4] to introduce a new Law that updates "Amdahl's Law."[5] Amdahl's Law was appropriate to characterize centralized parallel processing supercomputing environments, where monolithic supercomputers ran batch computing jobs run by local users. While such environments still exist, we are rapidly entering into an era of "cloud computing," which also enables parallel

[2] Bill Lloyd, "Phelps Win: Faster than a Blink," at http://blogs.webmd.com/eye-on-vision/2008/08/phelps-win-faster-than-a-blink.html
[3] Patrick Thibodeau, "Stock Exchanges Start Thinking in Microseconds," Computerworld, August 4, 2008, at http://www.computerworld.com/s/article/323391/Stock_Exchanges_Start_Thinking_in_Microseconds
[4] Joe Weinman, "The 10 Laws of Cloudonomics," at http://gigaom.com/2008/09/07/the-10-laws-of-cloudonomics/
[5] "Amdahl's Law," at http://en.wikipedia.org/wiki/Amdahl's_law

processing services via scalable, elastic resources, but in addition provides access to those services in an essentially location-independent way via regional or global dispersion of the nodes providing those services.

Such an architecture has a precedent in the real world. To reduce the time required to get a cup of coffee or a hamburger, coffee shop chains and fast food restaurants have helpfully placed their locations on street corners all over the world. This reduces the "network latency," i.e., the time it takes to get from your home or office to the location. It is surely faster to run over to the coffee shop down the street than to drive over a highway network to one, say, in Seattle. At each location, these enterprises have also have parallelized their processing, for example, having multiple people assemble the burger, bun, pickle, tomato, and what-not, into a finished product, thus putting the "fast" in "fast food."

The same strategy is used in a variety of on-line applications. For example, to process a search query quickly requires extremely complex processing of the search terms that one enters against enormous multi-petabyte indexes of previously-crawled web pages and documents. Rather than conduct this on a single processor, major search engines divide the work up among up to 1000[6] individual computer servers, each conducting its own processing on a smaller segment of the data—a so-called "shard".[7] These results are then combined and returned to the user, possibly with the results of other tasks also conducted in parallel, such as spell-checking ("Did you mean _____?"). There would not be much value to such speed-up if the results then took a long time to return to the user, hence major providers such as Google[8], Microsoft[9], and Amazon Web Services[10] are continuously increasing their geographic presence through globally deployed facilities to reduce the latency associated with network transport. The driver of all of this effort is, of course, quantifiable business value.

Of course, there may be reasons other than latency and response time to disperse data or applications, e.g., reducing backbone network traffic via edge caching; enhancing business continuity; or due to regional or national cross-border data migration compliance reasons.

The terms "latency" and "bandwidth" are sometimes used as if they were equal, because sometimes they can constrain each other. If we consider a network to be a highway (or beltway) or to be a conveyor belt carrying packages, we can think of latency as how long it takes for a car or package to travel from point A to point B along the belt. We can think of bandwidth as the quantity of packages that may be carried at any given time—in effect, the width of the belt, or the number of lanes on the highway.

---

[6] Stephen Shankland, "We're all guinea pigs in Google's search experiment", May 29, 2008, CNet News, at http://news.cnet.com/8301-10784_3-9954972-7.html?tag=mncol;txt

[7] Luiz André Barroso, Jeffrey Dean, Urs Hölzle, "Web Search for a Planet: The Google Cluster Architecture," IEEE Micro, March-April 2003, pp. 22-28, at http://labs.google.com/papers/googlecluster-ieee.pdf

[8] Rich Miller, "Google Data Center FAQ," March 27, 2008, Data Center Knowledge, at http://www.datacenterknowledge.com/archives/2008/03/27/google-data-center-faq/

[9] Mary Jo Foley, "Where in the World are Microsoft's Data Centers?," ZDnet, March 26, 2010, at http://www.zdnet.com/blog/microsoft/where-in-the-world-are-microsofts-datacenters/5700

[10] Rich Miller, "Where Amazon's Data Centers are Located," November 18, 2008, Data Center Knowledge, at http://www.datacenterknowledge.com/archives/2008/11/18/where-amazons-data-centers-are-located/

The two quantities are conceptually independent: we can turn up the speed of the belt or reduce the distance between the points A and B, and that will reduce latency. Or, I can buy a wider conveyor belt or build more highway lanes and carry more packages or vehicles at any given time.

In practice however, the two can be interrelated. If the highway is congested and there are vehicles ahead that are slowed or stopped, a trip will take longer: insufficient bandwidth can increase latency. Also, even "infinite" bandwidth may not overcome issues related to data transport protocols which require acknowledgements. In effect, if one is moving a large household from say, New York to Los Angeles, one would like to move it all at once in a large moving van. But some data transport protocols don't behave that way. Instead, it's as if they move a single lamp to California, then wait for a paper receipt to be driven back before sending the next item of furniture. So, even though latency and bandwidth are nominally independent concepts, in practice, high latency can cause low "effective" bandwidth, and insufficient bandwidth can increase latency.

For the purposes of this paper, we will focus on latency and ignore bandwidth. We will also ignore the impact of bandwidth on latency, and ignore the peculiarities of various protocols that may cause unexpected behavior. While bandwidth is often a concern, there are certainly cases where it may be safely ignored as an issue: first, because there may typically be sufficient bandwidth, and second, because many applications, for example, web search, do not require substantial bandwidth.

For example, in today's search applications, each typed letter can cause a new set of suggested search terms or phrases to be provided, or can cause several actual query results to be returned based on the most likely search query based on the partial information available to that point (a partially typed string of characters). If we were to assume 5 letters per second, and 8 bits per letter, one might say that the upstream bandwidth need is 40 bits per second. Since many of today's networks can easily deliver 1 megabit per second of uplink and downlink bandwidth, there is plenty of headroom. On the return path, if each letter typed generates 10 results, and each result is 3 or 4 lines, each with 100 characters on it, then we would need bandwidth of about 120,000 to 160,000 bits/second: a lot more intensive. However, even this is but a fraction of a lower end "high-speed Internet access" service. There are other issues, such as packet headers and checksums and the like which mean that more data must be transported than the size of the payload, but these calculations give a rough sense of the network bandwidth needed at the low end. Of course, if the request string specifies a high definition video, then several megabytes to several gigabytes will need to be returned, but if streamed then there may be one to two hours of time to transport that data.

From a latency perspective, there are different issues. If one can type 5 letters per second, that equates to one letter every 200 milliseconds. Consequently, for highly interactive applications that are returning substantial information with each letter, there is an upper bound on total end-to-end latency: the response must be requested, processed, and received before the next letter is typed. Since worst-case global round-trip network latencies are right around that 200 millisecond mark, there is not much leeway for processing in a single instance architecture.

As mentioned earlier, in environments such as NYSE Euronext, rather than 200 milliseconds, the timeframe of interest is more like 200 *microseconds*. Such stringent performance requirements drive on-site infrastructure, because network propagation delays of more than a few meters can result in unacceptable financial results. We won't be concerned with those types of applications here. However, there are a broad range of response times driven by human interface requirements. These range in the tens to hundreds of milliseconds.

Meeting these requirements strains the limits of the global Internet infrastructure. For example, here are recent response time results based on a study performed by Cedexis using *Cedexis Radar*, which leverages 4.5 million locations in 233 countries across over 30,000 networks to conduct billions of measurements to determine data such as average response time for HTTP (HyperText Transport Protocol) transactions to/from the East Coast of the United States:

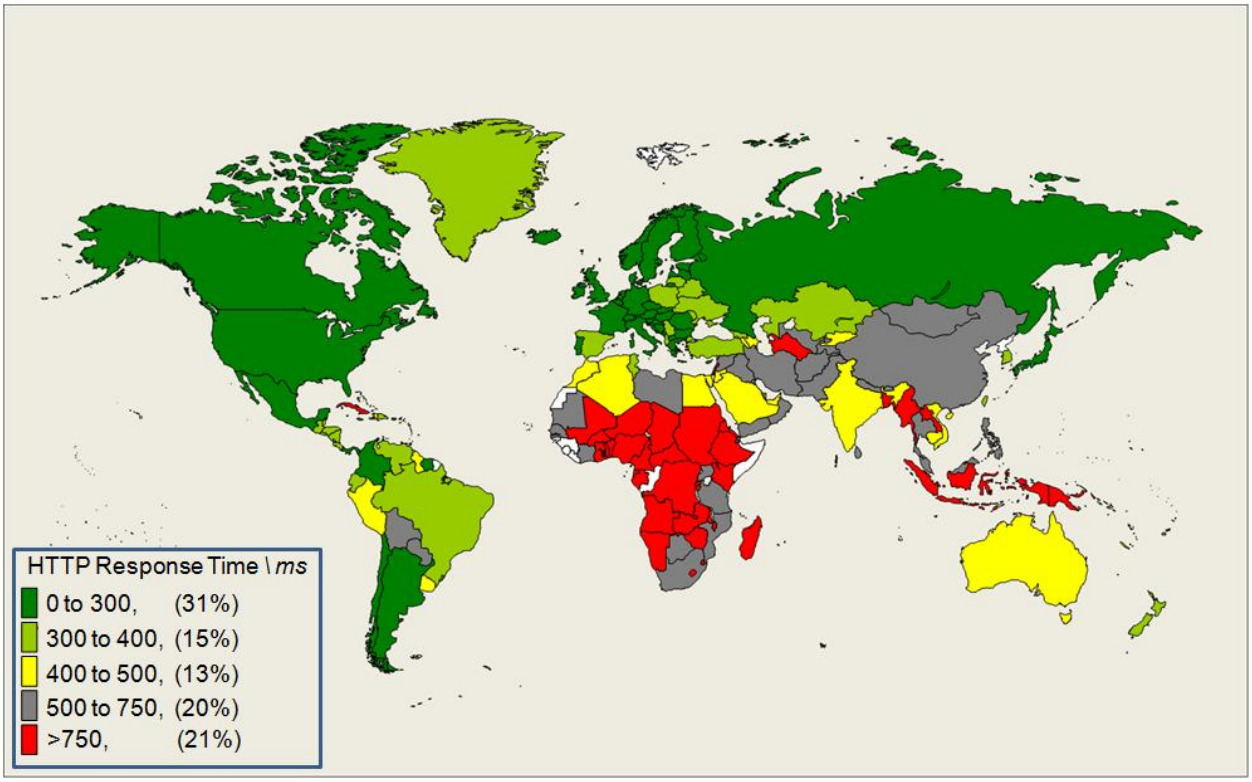

Figure 1: HTTP Response Times to the East Coast of the U.S. *Source: Cedexis*

While no doubt some transactions were in acceptable thresholds, there is clearly a challenge in globally serving transactions from a single geographic area while meeting latency requirements in the tens to one to two hundred millisecond range when the vast majority of transactions greatly exceed this. A Bitcurrent study analyzing Cedexis data determined that average time for an http request/response was 426.4 milliseconds, and therefore that "directing a regional client to the correct regionalized zone for a cloud provider does improve performance."[11] Moreover, Bitcurrent points out that "it's well understood among statisticians, performance experts, and site

[11] Alistair Croll, "Cloud Performance from the End-User Perspective," March 31, 2011, at http://www.bitcurrent.com/download/cloud-performance-from-the-end-user-perspective/

operators that averages mask problems," in other words, there can be a long tail of extremely lengthy (i.e., bad) response times much larger than this average. In any event, whether there is a long tail or not, an average time of 426.4 milliseconds means that many, if not most, transactions take too long. Consequently, a distributed infrastructure is required, but how distributed must it be, and why?

To partly answer the second question first, there is compelling data available from major web sites on the business value of latency reduction.[12,13]

A major search provider increased its number of returned results per page from 10 to 30, thus increasing the latency from 400 milliseconds to 900 milliseconds. While more results might intuitively seem likely to have a better chance of providing one of greater relevance and thus increasing click-throughs, it turned out that user traffic declined by 20%. Since search providers' revenues are directly related to click-throughs, this same search provider found that the 20% traffic decline resulted in a 20% revenue decline.

The search provider chalked this up to the ½ second increase in time to generate and deliver the results. Interestingly, offline results from experiments conducted by Iyengar and Lepper in selling jams at a stand in a mall showed that increasing consumer choice also reduces the propensity to purchase[14], so this may be partially to blame for the revenue decline associated with what has been called "choice overload" or the "tyranny of choice." Consumers were shown a display of either 6 jams or 24 jams. Regardless of how many they were shown, they tended to taste about the same number: about one and a half on average. However, about 30% of the consumers shown *fewer* jams actually then bought a jar, whereas the ones offered a choice of *four times as many* were only *one-tenth as likely* (3%) to purchase.

In this same set of experiments, subjects were also given a choice from a set of either 6 chocolates or 30 arrayed in a 5x6 array. Interestingly, the upper number of choices, 30, corresponds exactly to the search provider's upper experimental bound. As one might expect, the time to choose a chocolate was substantially greater: an average of 24.36 seconds instead of 8.91 seconds. Also, participants' perceptions generally were that 30 choices was "too many" and 6 was "about right." And, this is for selecting from among chocolates, a task for which humans innately have parallel processing skills in the human visual perception system as well as direct (short-cut) connections to emotional systems responsible for processing both the esthetic and affective dimensions of a basic human need: resolving hunger. This is in contrast to the relatively slow processes of reading and comprehension, and the fact that search results are displayed in linear fashion, requiring scrolling in most cases to review 30 results.

[12] Steve Souders, "Velocity and the Bottom Line," O'Reilly Radar, July, 2009, at http://radar.oreilly.com/2009/07/velocity-making-your-site-fast.html

[13] James Hamilton, "Perspectives: The Cost of Latency," October 31, 2009, at http://perspectives.mvdirona.com/2009/10/31/TheCostOfLatency.aspx

[14] Sheena Iyengar and Mark Lepper, "When Choice is Demotivating: Can One Desire Too Much of a Good Thing?" Journal of Personality and Social Psychology, 2000, Vol. 79, No. 6, 995-1006, at http://www.columbia.edu/~ss957/articles/Choice_is_Demotivating.pdf

However, latency is undoubtedly a component of the revenue decline, as other providers have kept the content (and thus number of choices) the same, experimenting with "delaying the page in increments of 100 milliseconds and found that even very small delays would result in substantial and costly drops in revenue."[15] Other companies in other businesses have shown a correlation between faster response times and better financial results.

Moreover, even in contexts well outside of online retailing and ad-serving, there are clear benefits to improved response time. For example, any information worker will experience higher labor productivity through faster response times, because he or she will be able to process work more quickly and thus do more work in a given time. There are natural breakpoints where studies have demonstrated that humans begin to lose focus on their work while awaiting computer task completion. Jakob Nielsen, the "guru of Web page usability,"[16] has argued that 100 milliseconds is the maximum "response time limit if you want users to feel like their actions are directly causing something to happen on the screen."[17]

Collaboration, for example, through interactive, immersive video conferencing, has a limit of about 200 to 250 milliseconds delay for a conversation to appear natural.

In computer tasks, humans notice delays at around 150 milliseconds for keystroke mirroring and at around 185-195 milliseconds for mouse operations.[18]

While human response times are arguably slow due to nerve signal transmission times even in myelinated axons (the biological equivalent of fiber optic cladding or coaxial cable sheathing), humans have learned to compensate for this by an ability to anticipate and plot trajectories, thus evolving to manage physical tasks such as coordinating visual perception with neuromotor control, for example, to swing a tennis racket to place a ball right at the opponent's backhand baseline, or to swing a bat to hit a curveball, or to move a mouse to click on a hyperlink. These skills no doubt conferred evolutionary advantage via throwing a rock or a spear at fast-moving prey and thus enhancing survival. These types of activities are no longer limited to the offline world, online games that engage humans physically, leveraging new interfaces such as those offered by the Nintendo® Wii™ or Microsoft® Kinect™ are emblematic of this new age. Consequently, global internetworks will need to provide extremely short round-trip delays to support such emerging communication, collaboration, and competition, at least for regional cohorts of participants.

However, as I point out in my 8th Law of Cloudonomics: reducing latency by a factor of n requires n^2 as many local service nodes. This means that if worst case global latency from/to a single node is 160 milliseconds, then reducing it by one-half to 80 milliseconds requires

[15] Greg Linden, "Marissa Meyer at Web 2.0," Nov 9, 2006, at http://glinden.blogspot.com/2006/11/marissa-mayer-at-web-20.html

[16] Matt Richtel, "Making Web Sites More 'Usable' Is Former Sun Engineer's Goal," http://www.nytimes.com/library/tech/98/07/cyber/articles/13usability.html

[17] Jakob Nielsen, "Powers of 10: Time Scales in User Experience," Alertbox, October 5th, 2009, at http://www.useit.com/alertbox/timeframes.html

[18] James R. Dabrowski and Ethan V. Munson, "Is 100 Milliseconds Too Fast?," Conference on Human Factors in Computing Systems, 2001, ACM.

deployment of 4 nodes[19]. A subsequent 40 millisecond reduction requires 16 total nodes, the next 20 ms needs 64 nodes, the next 10 ms of reduction requires 256 nodes, the next 5 ms requires 1024 nodes, and so forth.

For user interactions, which may require on the order of 20 to 200 milliseconds of latency, this means that a few dozen nodes would appear to be the optimal level of deployment to reach a global audience. Practically, since 2/3 of the surface of the earth is water, and some markets such as Antarctica may not be economical to reach, it suggests that a couple of dozen nodes would suffice to meet human latency requirements over a data network. However, if one further uses up time budget with the processing required for the response, even more nodes will be needed, and a challenge is to determine how to optimally allocate resources to parallel processing vs. dispersion.

2. Network Latency and Distance

If we have two points $a = (x_1, y_1)$ and $b = (x_2, y_2)$, lying on a plane, the physical distance between the two is of course $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$. If the rate of signal propagation over that distance is some speed of propagation c_p, then excluding bandwidth constraints, and assuming a direct (straight line) connection, the one-way latency between a and b is $c_p \times d$. For example, if a and b lie in a vacuum, $c_p = c = 186{,}282$ miles per second, the speed of light in a vacuum. For the single mode fiber used in today's long haul networks, the speed is reduced by a third to about 124K miles per second.

We will assume in this paper that latency is proportional to distance, and use distance and coverage areas as a proxy for network latencies. This includes first order effects such as propagation delays and fractal network topology, but ignores other effects such as nonlinearities in router hops; differences in signal propagation time between copper, fiber, and free space optics; dynamic, non-uniform congestion; and the like.

The biggest issue with this assumption is that unlike say, propagation of gravity in free space or radio or television broadcasting from a point source, network transport in routed data networks, e.g., copper and fiber, must follow the route of the cable. Such routing is not universally omnidirectional, but rather follows rights-of-way such as east-west train track routes between cities or major subsea cable routes that, e.g., follow the Suez Canal and then round the coastline of the Persian Gulf. Perhaps the most authoritative source of global subsea cable builds is Telegeography.[20] A quick perusal of the latest (2011) edition shows over 120 cable

[19] Actually these results are idealized results for a plane. On a sphere, the first reduction only requires one additional node at the antipolar point. Optimal "Circle Packing" and "Circle Covering" on a sphere are problems that have been solved mostly for individual cases, and are addressed later in this paper.

[20] 2011_0208_Cable_BaseMap.jpg, at http://www.telegeography.com/product-info/map_cable/downloads/2011_0208_CableMap_TeleGeography.jpg.zip

systems already in use and over two dozen more entering service in the next two years. While exact routes are not always shown, certain preferred paths are clear, e.g., transatlantic from North America to Europe, East-West across the Mediterranean, through the Suez Canal and around the Persian Gulf, connection via landings in Hawaii, and so forth. To these routes must be added a variety of in-country terrestrial networks, which also tend to show limited routing, with backbones typically connecting major cities. For example, the U.S. can be thought of as primarily three East-West routes at various latitudes (Boston-New York-Chicago-Seattle, Washington, D.C.-St. Louis-Denver-San Francisco, and Atlanta/Orlando-Dallas-Phoenix-Los Angeles/San Diego) and three North-South routes (Seattle-San Francisco-Los Angeles-San Diego, Detroit-Chicago-St. Louis-Dallas-Austin-Houston, and Boston-New York-Philadelphia-Washington, D.C.-Atlanta-Orlando), with a few diagonal connections and various local loops.

Due to physical network cabling constraints, as well as traffic management policies that route traffic via high-bandwidth backbone links, a formula such as $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ is not exactly true. However, we can assume that generally speaking, delay or network latency is roughly proportional to distance. To test whether this assumption is valid, we look at publicly available information on round trip network latencies between city pairs from a large network provider[21], compared against information on distances between city pairs generally available at various sites on the World Wide Web, e.g., airline ticketing sites. For example, the round trip latency from San Francisco to Hong Kong on a large global network is 160 milliseconds, and this represents an "as-the-crow-flies" (or as-the-Boeing-777-flies) distance of 6,897 miles.

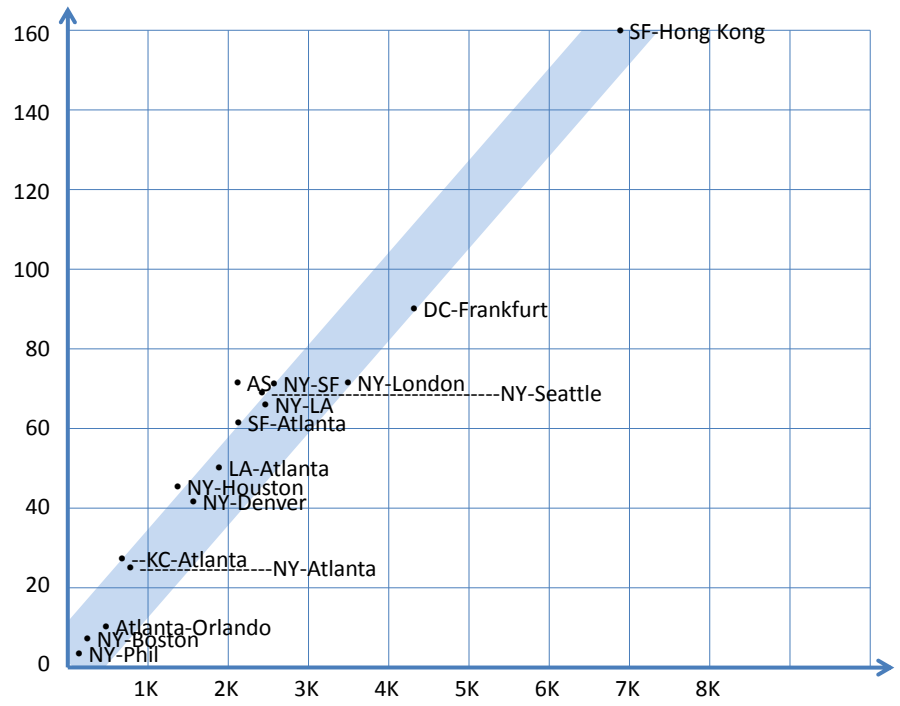

Figure 2: Network Latencies vs. Direct Travel Distances

[21] http://ipnetwork.bgtmo.ip.att.net/pws/network_delay.html

The latency and distance measures of the city pairs shown show some anomalies, e.g., Atlanta-Seattle (shown above as "AS"), NY-San Francisco, and New York to London all have measured network latencies of 72 milliseconds, even though the distances between those city pairs are 2,176 miles, 2,565 miles, and 3,466 miles respectively. However, a review of the network map from the provider with those latencies shows that there is no direct backbone path from Atlanta to Seattle, so rather than traversing a Northwesterly hypotenuse, the network routing is more along the lines of due West from Atlanta to Dallas to Phoenix to Los Angeles, and then due North from Los Angeles to San Francisco to Seattle. Total distance traversed via this route is 3,130 miles, squarely in the center of the blue band. At the other extreme, NY-London has the same latency, for a greater distance, due to its relatively direct routing.

The speed of light in a vacuum is 186,282 miles per second, but in single mode optical fiber it is only 2/3 of that, or roughly 124,000 miles per second, which of course equates to 124 miles per millisecond, one way, or, to put it another way, about 2 milliseconds for every 124 miles, round trip. Thus, for that 3,466 mile route such as New York to London, we thus would expect 56 milliseconds. The remainder of the 72 millisecond total can be attributed to other factors, e.g., indirect and sub-optimal (non-great circle) cable routes, delays within network elements due to congestion / queuing / buffering, and technically even the difference between propagation time of the leading edge of a propagating waveform vs. the transmission time of the entire packet due to "serialization delay," etc.

Moreover, network latency is not deterministic, but stochastic, due to numerous factors ranging from physical propagation delay to router hops to dynamically varying network congestion to network element outages and corresponding network reconvergence events that can arise. There is substantial complexity[22] in the definition and measurement of network latency, as well as jitter, or variation in such latency. Various means exist to measure such variation, e.g., variation from a target reference value, as well as "Inter-Packet Delay Variation," the difference in latency from one packet to the next.

However, overall, these randomly selected cities show a correlation coefficient of .98 between latency and distance. Consequently, we will assume that latency is proportional to distance and use an idealized theoretical model to show the results in this paper. How then should one cover an area within a given distance or latency constraint?

3. The Computational Complexity of Coverage

Elsewhere, I have shown[23] that the task of matching supply and demand in a networked cloud computing environment is NP-complete, that is, computationally intractable[24] and equivalent to a

[22] Leonard Ciavattone, Alfred Morton, and Gomathi Ramachandran, "Standardized Active Measurements on a Tier 1 IP Backbone," IEEE Communications Magazine, June, 2003, pp. 90-97, available at http://ipnetwork.bgtmo.ip.att.net/pws/att_ieee.pdf

[23] Joe Weinman, "Cloud Computing is NP-Complete," Working Paper, February 21, 2011, at http://www.joeweinman.com/Resources/Joe_Weinman_Cloud_Computing_Is_NP-Complete.pdf

family of other problems that are equally[25] difficult to solve. It turns out that even if there *is* sufficient capacity at all nodes to meet demand, the optimal placement of service nodes is also computationally intractable, a problem in so-called "computational geometry."

Given a set of users in specific geographic locations and *p* service nodes, the "*p*-center" problem may be stated as the task of minimizing the worst-case distance from a user to a service node, and the "*p*-median" problem may be stated as minimizing the sum of the distances between each user and its nearest service node. Therefore, since the number of users is constant, we may think of this as minimizing the average distance. Distances may be measured using our traditional notion of "as-the-crow-flies," or *Euclidean* distance, or using a "Manhattan street," or *rectilinear* distance, where one is constrained to travel in either a North-South or East-West direction only.

In any case, all four combinations—Euclidean *p*-center, Euclidean *p*-median, rectilinear *p*-center, and rectilinear *p*-median—have been shown[26] to be NP-hard, or more colloquially, among the hardest known problems, via reductions from 3-Satisfiability (3SAT). The reader is directed to Megiddo and Supowit for the detailed proof. Briefly, as with most reductions from 3SAT, there are truth-setting components to permit selection of exactly one value of TRUE or FALSE for each Boolean variable, and the placement of a limited number of circles only permits one or the other to be selected. There are also clause satisfaction-testing components, which ensure that at least one variable in each clause is TRUE. Finally, there are communications link components that permit the truth-setting and satisfaction-testing components to interact, and moreover, crossover junctions—like traffic circles—that allow North-South links to cross over East-West links without interference. The Euclidean distances are reflected via circle coverings, and the rectilinear distances are reflected via square coverings.

There are a number of variations on this problem. For example, instead of Euclidean or rectilinear distance, each of the points may be viewed as lying on a network or graph, with weights connecting the links. Nodes (vertices) may then be considered as users or service nodes. One may then solve the problem where the distance between nodes is the least sum of the weights on a path connecting nodes, and one may also require that the selected nodes be on a connected backbone (the *p*-center problem with connectivity constraint[27]). For all of these problems, the difficulty arises due to the need to use "relatively few" covering objects to span irregular terrain. We can simplify the problem by using as many objects as we need to cover most or all of the territory.

---

[24] Michael R. Garey and David S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Co., San Francisco, 1979.

[25] Subject to polynomial time reductions.

[26] Nimrod Megiddo and Kenneth J. Supowit, "On the Complexity of Some Common Geometric Location Problems," SIAM J. Computing, Vol. 13, No. 1, February 1984, at http://citeseer.ist.psu.edu/viewdoc/download;jsessionid=85C4192B7B7E08853D2213C22A8B28AB?

[27] William Chung-Kung Yen and Chien-Tsai Chen, "The *p*-Center Problem with Connectivity Constraint," Applied Mathematical Sciences, Vol. 1, 2007, no. 27, 1311-1324, at http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=7F7439CD1558337700915220D457528C?

4. Covering an Area

We will assume that a given service node, such as a content delivery server, distributed web server, edge application server, or even a coffee shop or fast food joint, can interact with users in a circular area to meet a latency threshold. For now we will assume that the latency threshold is "worst-case," later we will show that worst-case latency follows similar rules to average latency. For example, the users may be mobile users, and the service node a cellular base station, or the users may have wired endpoints that interact with a service node over a wireline network, or there may be some combination of the two.

We can cover an area with service nodes each with identical service area in a somewhat random fashion, laying down nodes by throwing darts at a map, but this is inefficient: some locations will have unnecessary excess coverage as shown in the upper right of the diagram below, and others will have no coverage, for example, the upper left of the diagram. Intelligent coverage strategies are necessary to make efficient use of resources, such as the hexagonal circle packing approach shown in the lower left.

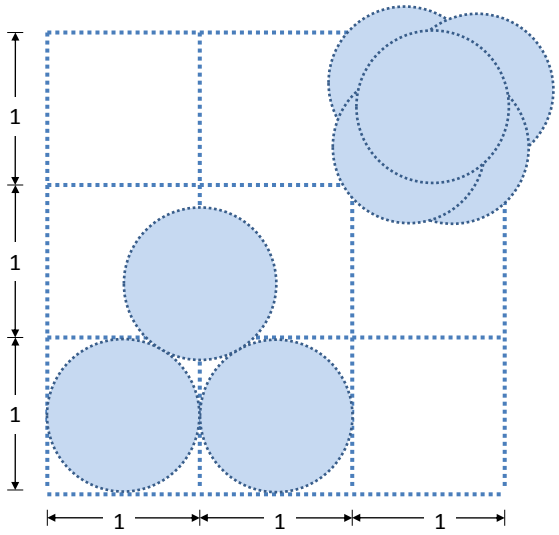

Figure 3: Attempts to Cover an Area with Circles

The term "packing density" denoted by, η (the lower case Greek letter 'eta') is used to indicate the proportion of space filled by a set of objects. For this paper, we are not concerned with volumes, e.g., close spherical packing as in cannon ball stacking, but rather in covering a planar area. We will also introduce a new measure, "overlap," which we'll denote by ψ[28] (the lower case Greek letter 'psi,' usually pronounced "sigh").

[28] We use the symbol ψ partly because wave functions can have overlaps, but primarily to maximize the multifaceted pun on the title's lyrics at the conclusion of this paper.

For n circles $c_1, c_2, \ldots c_n$ of radius r that are placed to cover an area A, $\eta = \frac{n \times \pi r^2}{A}$, and $\psi = \frac{n \times \pi r^2}{area(c_1 \cup c_2 \cup \ldots c_n)}$. If $\psi = 1$ then we have no overlap, otherwise we have $\psi > 1$. Note that these variables are independent, but there are some important combinations. When $\eta < 1$ and $\psi = 1$ we have a "circle packing." When $\eta = 1$ it isn't possible for $\psi = 1$ (at least for circles), so we have some degree of overlap. The circle packing problem can be phrased as maximize η, subject to $\psi = 1$, and the circle covering problem can be phrased as minimize ψ, subject to $\eta = 1$. Of course, there are inefficient solutions where neither is the case.

4.1 Packing

The "circle packing" problem is the challenge of attempting to cover a plane with circles of identical radius in a non-overlapping fashion, minimizing the area that is uncovered. The relevant metric is the packing density η. If we don't need to reach every square inch, but would like to be reasonably efficient, we can use a "square packing" approach. In this case, the center of each circle is at a point in a rectilinear grid, such as below:

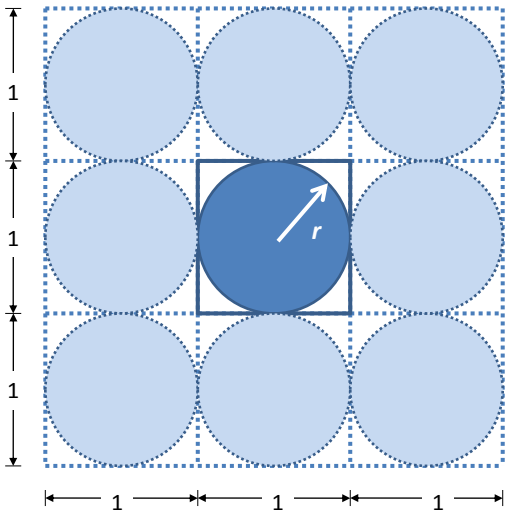

Figure 4: Square Circle Packing

Each circle has area πr^2, where $r = \frac{1}{2}$, and the total area of each square is 1, so therefore the packing density $\eta = \frac{\pi}{4}$, or about .79. However, we can do better: hexagonal packing has been proven to be the densest possible circle packing, again, this means non-overlapping coverage of a plane by circles.

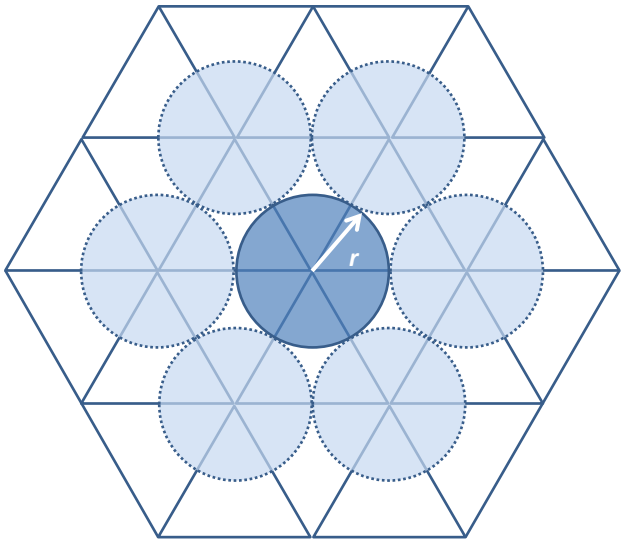

Figure 5: Hexagonal Circle Packing

Hexagonal circle packing has a packing density of $\eta = \frac{1}{6}\pi\sqrt{3}$, or about .91.[29] This may be seen by considering that the ratio of circles to equilateral triangles is 1:2, or equivalently, there is a one-to-one ratio between circles and rhombi (a rhombus is the diamond shape, or more precisely, a parallelogram with equal length sides) as shown below:

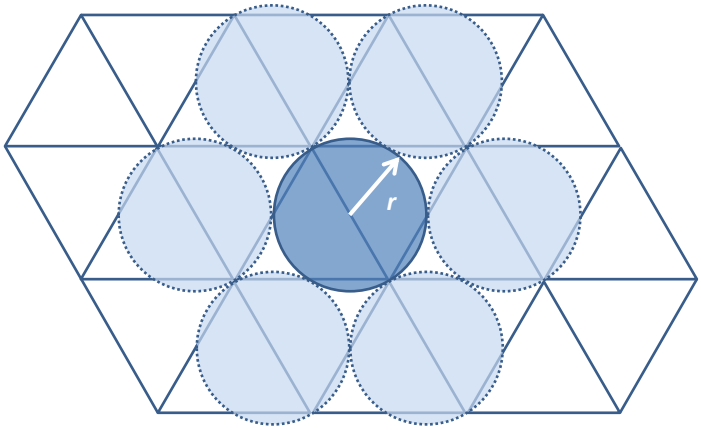

Figure 6: Lattice Translation to Determine Packing Density

[29] Eric W. Weisstein, "Circle Covering," from *MathWorld*--A Wolfram Web Resource, at http://mathworld.wolfram.com/CircleCovering.html

The area of an equilateral triangle may be determined using the classic formula for the area of a triangle: $\frac{1}{2} \times base \times height$. Slicing the equilateral triangle vertically down the middle gives us two right triangles each with base $\frac{1}{2}\ s$ and hypotenuse s. Since the square of the length of the base + the square of the length of the height equals the square of the hypotenuse, we know that the height is $\sqrt{s^2 - (\frac{1}{2}s)^2}$, which is just $\sqrt{\frac{3}{4}s^2} = \frac{\sqrt{3}}{2}s$, But two triangles with base $\frac{1}{2}\ s$ and height $\frac{\sqrt{3}}{2}s$ have area $2 \times \frac{1}{2} \times (\frac{1}{2}\ s) \times (\frac{\sqrt{3}}{2}s)$ which is equal to $\frac{s^2\sqrt{3}}{4}$.

We can therefore use the fact that the area of an equilateral triangle with side s is $\frac{s^2\sqrt{3}}{4}$, together with the fact that $s = 2r$ to determine that "each circle almost covers every two triangles" means that the packing density η is then πr^2 to $2 \times \frac{(2r)^2\sqrt{3}}{4}$, or $\frac{2\pi r^2}{4r^2\sqrt{3}}$, which may be rewritten $\frac{\pi\sqrt{3}}{6}$, which is about .91.

4.2 Covering

Often, however, we do not want to allow there to be gaps in coverage. If we are trying to cover a plane with circles this inevitably will entail overlapping. This problem is now called "circle covering" instead of "circle packing." The term "packing density" is now of less interest, so we will use overlap ψ to assess the "investment" in circles to fully cover the area. Since there is some overlap, ψ will be greater than 1.

If we permit overlapping, there are many ways to overlap. In the most pathological example, we can use an infinite number of circles all placed on top of each other, in which case no matter how many we add, we will never cover more than πr^2 of total area. If they don't overlap, we will cover $n \times \pi r^2$. Consequently, when some overlap to some extent they will cover between πr^2 and $n \times \pi r^2$ worth of area, but how much exactly?

If we use a square packing approach, but expand each circle to make it a circle covering so that there are no gaps, each circle will need to have a diameter equal to the length of the diagonal of each square. If the side of the square is of length 1, the diagonal is of length $\sqrt{2}$, and thus the radius is $\frac{\sqrt{2}}{2}$, therefore each circle spans an area of $\pi(\frac{\sqrt{2}}{2})^2$, which is of course just $\frac{\pi}{2}$.

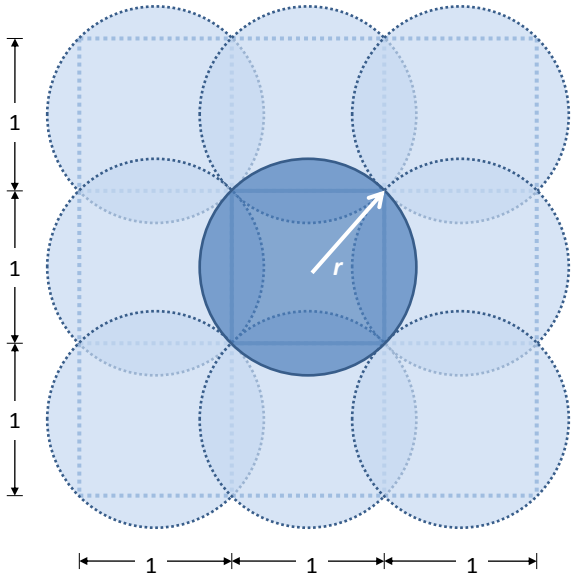

Figure 7: Square Circle Covering

If we are trying to cover 100% of a plane with circles using this approach, we will need to pay a penalty of overlapping (and thus a wasted investment), requiring at least one circle of $\frac{\sqrt{2}}{2}$ radius to cover each square unit area, for a ψ of about $\frac{3.141}{2} = 1.57$.

Suppose we take an existing hexagonal circle packing and increase each circle's radius as little as possible, but enough to fill in the small gaps between each group of 3 mutually adjacent circles? As can be seen from the diagram, the trick is to increase the radius of each circle by an amount equal to the proportion of an equilateral triangle's side to its height.

However, we already know from the analysis above that an equilateral triangle with side s has a height of $\frac{\sqrt{3}}{2}s$, so this proportion is $\frac{s}{(\frac{\sqrt{3}}{2}s)}$, which is of course just $\frac{2}{\sqrt{3}}$. Since we determined above that the packing density η was $\frac{\pi\sqrt{3}}{6}$ for hexagonal close packing, and we haven't changed the spacing of the hexagonal lattice, just increased the radius of the circles, the new area will be increased by a factor which is the square of the ratio of the radii, i.e., the square of $\frac{2}{\sqrt{3}}$ which is just $\frac{4}{3}$. So, the new coverage ratio is $\frac{\pi\sqrt{3}}{6} \times \frac{4}{3}$ which may be rewritten as $\frac{2\pi\sqrt{3}}{9}$ or also as $\frac{2\pi}{\sqrt{27}}$. This is apparently[30] the best known lowest bound for circle covering, leading to an overlap ψ of about 1.21.

[30] Eric W. Weisstein, "Circle Covering," from *MathWorld*--A Wolfram Web Resource, at http://mathworld.wolfram.com/CircleCovering.html which in turn references Williams, R. "Circle Coverings." §2-6 in The Geometrical Foundation of Natural Structure: A Source Book of Design. New York: Dover, pp. 51-52, 1979.

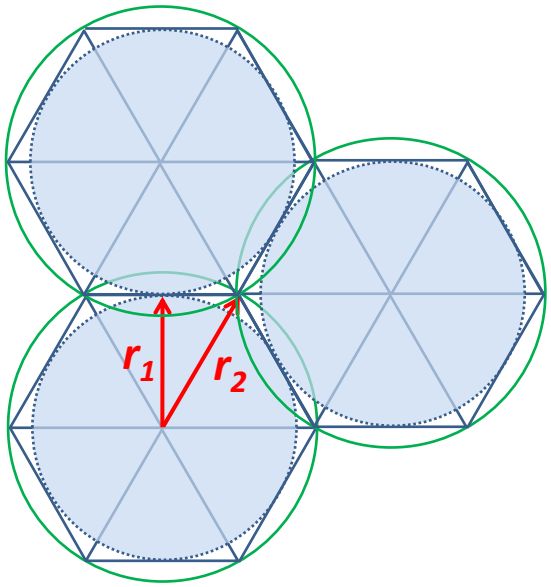

Figure 8: Extending Hexagonal Circle Packing to Achieve Hexagonal Circle Covering

Generally speaking, whichever pattern we use enables us to either "almost" cover (i.e., circle pack), or "over cover" (i.e., circle cover) each square unit of area with k circles of radius r. To put it another way, any given strategy provides us an independent $\psi \geq 1$ possibly together with a packing density $\eta \leq 1$ (when the metric may be used). For packing, the overlap $\psi = 1$, since each quantity of area of circle buys us the equivalent coverage of the plane. For coverings, the overlap $\psi > 1$, e.g., $\psi = 1.21$ in the most efficient covering known.

5. Worst-Case Latency Reduction

Suppose we are trying to deploy the minimum number of service nodes so that we achieve a given coverage ratio of a planar area where users are either homogeneously distributed in a regular lattice or uniformly stochastically distributed on a plane. In the real world of course, Manhattan, Tokyo, Seoul or San Paolo each have many more users per square mile than, say, Antarctica, but we will not concern ourselves for now.

Also, suppose that we are trying to ensure that each of the covered users is within a given distance d of a service node. Let us assume for now that latency means "worst case."

> ***Proposition 1: The 8th Law of Cloudonomics (for worst-case latency):*** *Worst-Case* Latency is inversely proportional to the square root of the number of nodes.
>
> ***Proof:*** The locus of points within distance d of a service node spans area πd^2 (it is important to note that d here stands for *distance*, not *diameter*, and that we assume that

the node does not have physical extension, i.e., it can be consider to be an idealized point).

The area A that can be covered by n non-overlapping circles of radius r, whether they are dispersed or via a circle packing approach is of course

$$A = n \times \pi d^2$$

Which we can just write as

$$\frac{A}{\pi} = n \times d^2$$

If there *is* overlap, as occurs, for example, in a circle covering approach, the actual area covered is less than A, based on the overlap ψ:

$$A = \frac{1}{\psi} \times n \times \pi d^2$$

Or

$$\frac{A \times \psi}{\pi} = n \times d^2$$

In either case, if we hold A constant and use the same dispersed, circle packing, or circle covering approach to determine node topology, we can treat $\frac{A}{\pi}$ or $\frac{A \times \psi}{\pi}$ as a constant,

$$\Rightarrow K = n \times d^2$$

$$\Rightarrow d^2 \propto \frac{1}{n}$$

$$\Rightarrow d \propto \frac{1}{\sqrt{n}}$$

As argued earlier, we will assume that latency l is proportional to distance d, i.e., $l \propto d$,

$$\Rightarrow l \propto d \propto \frac{1}{\sqrt{n}}$$

$$\Rightarrow l \propto \frac{1}{\sqrt{n}} \blacksquare$$

6. Average Latency vs. Worst-Case Latency

Suppose instead of worst-case latency we are concerned about average (i.e., typical, usual, expected, or mean) latency? Does this change any of the results so far? To answer this, we need to look at the relationship between average and worst-case.

Consider a service node serving an area. Again, we consider distance to be a proxy for latency, so we want to understand the relationship between the average distance $\bar{d}$ from the service node given a particular maximum distance $\hat{d}$. This is equivalent to determining the expected value of the distance from the center of a circle to a point selected at random within the same circle. A first guess might be $\bar{d} = \frac{\hat{d}}{2}$, but this doesn't work, since there are very few points close to the center, more "mass" is located towards the perimeter. We need to use a more formal approach.

Proposition 2: The expected distance $\bar{d}$ from the center c of a circle of radius $\hat{d}$ of a point p selected at random is $\frac{2}{3}\hat{d}$.

Proof: Let $F(x)$ be the probability that the distance d from the center c to the point p selected at random is less than or equal to x as shown in Figure 9.

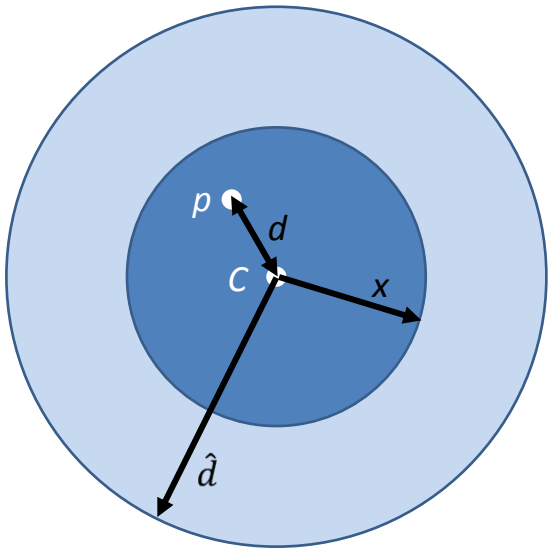

Figure 9: Determining the Cumulative Distribution Function

To determine $F(x)$, which is the cumulative distribution function, we can compare the proportions of the inner circle with radius x to the outer circle with radius $\hat{d}$, realizing then

that $F(x) = \frac{\pi x^2}{\pi \hat{d}^2}$, which, since $\hat{d}$ is a constant, may be written as $F(x) = \frac{1}{\hat{d}^2} \times x^2$. The probability density function is the derivative $f(x)$ which follows $f(x) = F'(x) = \frac{1}{\hat{d}^2} \times 2x$.

We know that the expected value $E(f(x))$ follows $E(f(x)) = \int x f(x)\, dx$.

Thus, $E\big(f(x)\big) = \int_0^{\hat{d}} x \times \frac{1}{\hat{d}^2} \times 2x = \int_0^{\hat{d}} \frac{1}{\hat{d}^2} \times 2x^2 = \frac{1}{\hat{d}^2} \times \frac{2}{3} x^3 \big|_0^{\hat{d}} = \frac{2}{3}\hat{d}$.■

What this means is that saying we'd like a worst-case latency of $\hat{d}$ is no different than saying we'd like an expected latency of $\frac{2}{3}\hat{d}$.

Proposition 3: The 8th Law of Cloudonomics (for expected latency): *Expected* Latency is inversely proportional to the square root of the number of nodes.

Proof: The proof is an echo of that for Proposition 1. The locus of points with expected distance d of a service node is identical to the set of points within (worst-case) distance $\frac{3}{2}d$, which spans area $\pi(\frac{3}{2}d)^2$ (again, d here stands for *distance*, not *diameter)*,

The area A that can be covered by n non-overlapping circles of radius r, whether they are dispersed or via a circle packing approach is of course

$$A = n \times \pi(\frac{3}{2}d)^2 = n \times \pi \times \frac{9}{4}d^2$$

Which we can just write as

$$\frac{4}{9}\frac{A}{\pi} = n \times d^2$$

If there *is* overlap, as occurs, for example, in a circle covering approach, the actual area covered is less than A, based on the overlap ψ:

$$A = \frac{1}{\psi} \times n \times \pi(\frac{3}{2}d)^2 = \frac{1}{\psi} \times n \times \pi \times \frac{9}{4}d^2$$

Or

$$\frac{4}{9}\frac{A \times \psi}{\pi} = n \times d^2$$

In either case, if we hold A constant and use the same dispersed, circle packing, or circle covering approach to determine node topology, we can treat $\frac{4}{9}\frac{A}{\pi}$ or $\frac{4}{9}\frac{A \times \psi}{\pi}$ as a constant, so as before,

$$\Rightarrow K = n \times d^2$$

$$\Rightarrow d^2 \propto \frac{1}{n}$$

$$\Rightarrow d \propto \frac{1}{\sqrt{n}}$$

As argued earlier, we will assume that latency l is proportional to distance d, i.e., $l \propto d$,

$$\Rightarrow l \propto d \propto \frac{1}{\sqrt{n}}$$

$$\Rightarrow l \propto \frac{1}{\sqrt{n}} \blacksquare$$

In short, it doesn't matter whether we mean "worst-case" or "expected," the 8th Law of Cloudonomics holds.

7. From Planes to Spheres

To this point, we have addressed packing, covering, and latencies on ideal, infinite planes. However, real latencies on our planet are due to real distances on the surface of the earth, which is, roughly[31] speaking, a finite sphere, not an infinite plane.

Unfortunately, packing and covering of circles, or "spherical caps" on spheres is as much *ad hoc* art as it is science. A spherical cap is the portion of a sphere on one side (e.g., "north of") a plane. If the plane bisects the sphere, the cap is a hemisphere (e.g., the Northern Hemisphere or Southern Hemisphere is the cap on one side of a plane containing the equator. A cap may be smaller or larger than a hemisphere. Looked at from the side, the following dimensions may be defined:

[31] Excluding such details as oblateness, mountain ranges and valleys, and other asymmetries and anomalies.

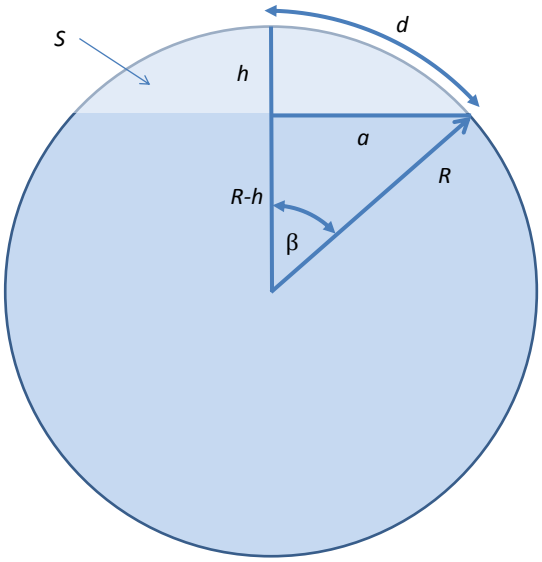

Figure 10: Interrelated Values for Spherical Caps

Let the radius of the sphere be R, the height of the cap be h, the surface area of the cap be S, and the distance along the surface of the sphere from the center of the cap to the edge of the cap be d. The radius of the base of the cap is a. The angle subtended by the distance d is β. There are now some obvious equalities as well as less obvious but known results relating these quantities. The surface area of the spherical cap is known[32] to be:

$$S = 2\pi Rh = \pi(a^2 + h^2)$$

To put it another way, the surface area of the cap is (somewhat surprisingly) proportional to the height of the cap. Also, as far as β is concerned, we can measure it in radians, realizing that

$$\beta = \frac{d}{R}$$

Since the total circumference of the sphere is $2\pi R$, when d is 360 degrees we see that $\beta = 2\pi$, when d is 180 degrees $\beta = \pi$, and so forth. Finally then, we can express $R - h$ as the cosine of β, so we have

$$(R - h) = R \times \cos(\beta)$$

Since $h = R - (R - h)$, we know that $h = R - R \times \cos(\beta) = R(1 - \cos(\beta))$. Using all of these identities together, we see that the surface area $S = 2\pi Rh = 2\pi R(R(1 - cos(\beta)))$, or in other words, $S = 2\pi R^2(1 - \cos(\beta))$. When β is π radians, i.e., 180°, $\cos(\beta) = -1$, so this formula agreeably shows that when $\beta = \pi$, the surface area for the spherical cap that is the entire sphere is just the surface area for the entire sphere, $S = 4\pi R^2$.

[32] Eric W. Weisstein, "Spherical Cap," from *MathWorld*--A Wolfram Web Resource, at, at http://mathworld.wolfram.com/SphericalCap.html

In the planar case, as d increases linearly, the surface area covered by a circle of radius d increases quadratically. On a sphere, there is no such simple relationship. For distances that are very small relative to the radius of the sphere, essentially the same relationship holds, since the sphere is "locally" practically flat. However, as the distances get larger and larger, this quadratic increase drops, slowly at first, and then more rapidly, all the way down to merely linear once we get to the size of a sphere. For example, the surface area of the *whole* earth is only *twice* that of the Northern Hemisphere, not *four* times.

Therefore, if we double the distance at the surface of the sphere from d to $2d$, we double the angular distance, i.e., the angle, from β to 2β, and therefore, while the old spherical cap surface area was proportional to $1-\cos(\beta)$, the new surface area is proportional to $1-\cos(2\beta)$ which is $1-\cos^2(\beta)+\sin^2(\beta)$, since $\cos(2\theta)=\cos^2(\theta)-\sin^2(\theta)$. These formulas aren't very intuitive, so to visualize this, let us compare the two:

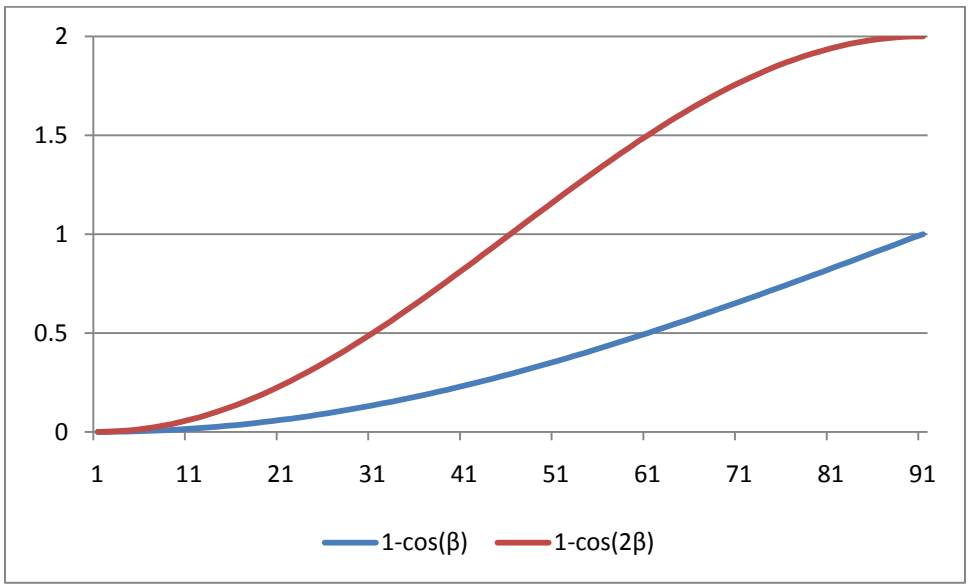

Figure 11: Comparison of $(\mathbf{1}-\boldsymbol{cos}(\mathbf{2}\boldsymbol{\beta}))$ ***to*** $(\mathbf{1}-\boldsymbol{cos}(\boldsymbol{\beta}))$ ***as*** $\boldsymbol{\beta}$ ***ranges from*** $\mathbf{0}°$ ***to*** $\mathbf{90}°$

The rolling wave of the cosine gets larger with increasing rapidity, causing the difference between the two curves to accelerate and then slow. What is of particular interest, however, is the ratio between the two curves, which is difficult to discern above at small angles, but starts out at 4 and then begins to diminish slightly, ultimately reaching a minimum at 2, which is the ratio between the surface area of a sphere and that of a hemisphere.

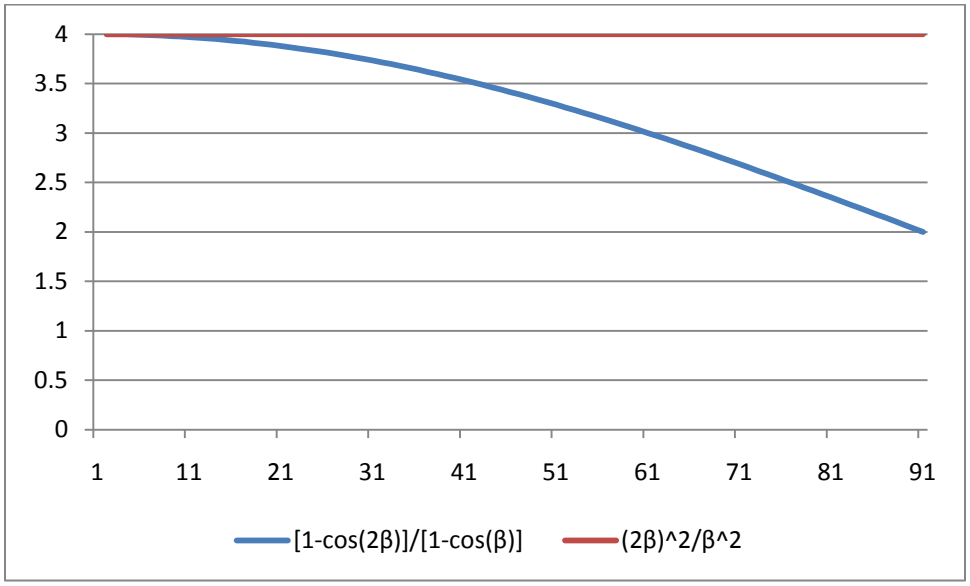

Figure 12: Comparison of $\frac{1-cos(2\beta)}{1-cos(\beta)}$ vs. $\frac{(2\beta)^2}{\beta^2}$ as β ranges from $0°$ to $90°$

For a network laid out on the surface of a sphere, which we will assume Earth is, we are interested in the total area covered by the spherical cap, i.e., that area that is within distance d. It would be nice and straightforward if there were some simple rule that aligned with the $\frac{1}{\sqrt{n}}$ rule already discovered, which is based on the fact that when we double the distance we quadruple the (surface) area that we can reach. What the chart above shows is that for "small" (relative to the size of the sphere) distances the same law holds, but that the larger the distance is the more we fall short of an actual quadrupling of area. However, even when we are talking about distances that span an entire hemisphere, we are still pretty close to the rule: the ratio in surface area between a $90°$ spherical cap (e.g., the Northern Hemisphere) and a $45°$ cap (Portland, Oregon or Milan, Italy and points North) is within 15% of the quadratic rule for planes.

8. The Tammes Problem

An even bigger issue at "long" distances (relative to the circumference of the sphere) than this slight distortion is the problem of finding ways to pack circles—or more precisely, spherical caps—onto a sphere. This does not exactly match the real world problem as we tend to only want to cover people on land, rather than sea, and in many cases, further prioritize node deployment according to economic importance. However, we will cover it briefly.

The Tammes Problem, formulated in 1930 by Dr. Pieter Merkus Lambertus Tammes, originally addressed the layout of pores on grains of pollen.[33] As can be seen from the photo below, the

[33] Pieter Merkus Lambertus Tammes, "On the origin of number and arrangement of the places of exit on the surface of pollen-grains," Dissertation, Faculty of Mathematics and Natural Sciences, University of Groningen, the

layout of these pores (at least for spherical grains) is identical to packing non-overlapping circles onto a sphere, and or distributing points "evenly" on a sphere, so as to maximize the minimum distance between centers.

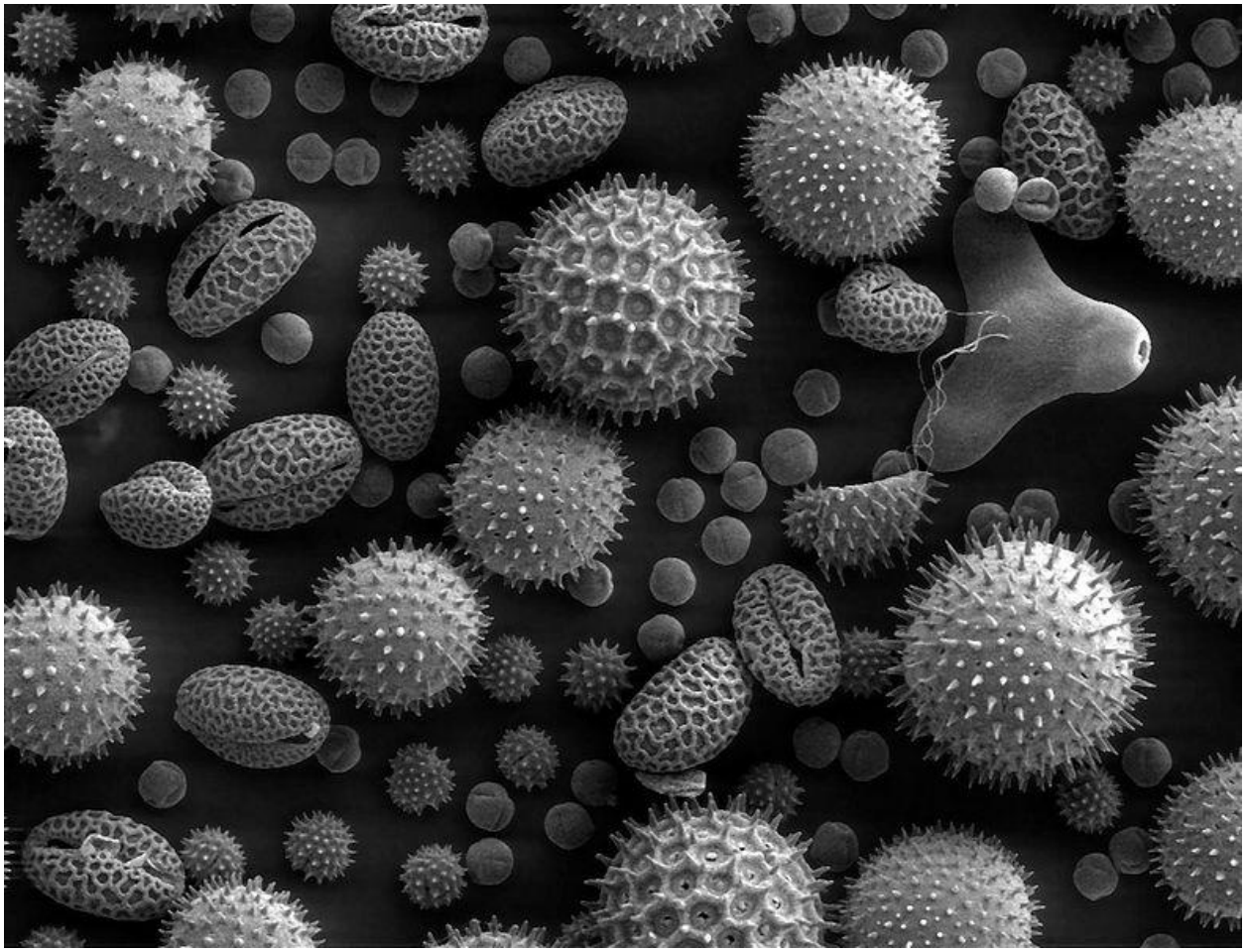

Figure 13: Grains of pollen from sunflower, morning glory, hollyhock, lily, primrose, and castor bean, magnified 500x. Photo courtesy of Dartmouth Electron Microscope Facility, Dartmouth College via Wikimedia Commons.

Tammes also concluded that there was a $\sqrt{n}$ effect at work (he termed it $\sqrt{a}$), writing that "the variation in number of places of exit was statistically traced and a typical correlation appeared to exist between the size of the pollen and the number of places of exit, in that…Diam. $= (\sqrt{a \text{ to } a+1}) \times V$, when the places of exit are distributed over the whole surface, *Diam.* representing the diameter of the grain, a the observed number of places of exit, and V a constant for pollen-grains of one species being under equal circumstances." He continues, "To

---

Netherlands, 1930, indexed at http://dissertations.ub.rug.nl/faculties/science/1930/p.m.l.tammes/?pLanguage=en&pFullItemRecord=ON with an abstract at http://dissertations.ub.rug.nl/FILES/faculties/science/1930/p.m.l.tammes/Tammes.pdf

the arrangement of the places of exit in the pollen, however variable it may be, yet a general rule obtains: that equidistance is observed in which the distance from a place of exit to the nearest places of exit is nearly equal in value…we might infer that the arrangement and the number of places of exit only depends on the closest covering of the space occupied by the places of exit…" Note that Tammes observations addressed the same rule from a different perspective. In effect, rather than keeping the size of the sphere constant and seeing how many more spherical caps could be placed by reducing their diameter, he observed that if the mechanisms responsible in a particular species drive a biologically determined V, then the diameter of the entire sphere is a function of square root of the number of "places of exit." In other words, $4\pi R^2 \propto n \times \pi d^2$ tells us that if d is a constant then $R^2 \propto n$ so $R \propto \sqrt{n}$, or in Tammes' terms, $Diam. \propto \sqrt{a}$.

Figure 14: A Golf Ball with a Dimple Pattern. Photo courtesy of office.microsoft.com.

Packing dimples onto the surface of golf balls is a similar problem, and has been teed up in the disclosures of a number of issued patents, see for example US7,179,178B2, "Golf Ball Dimple Pattern"[34], assigned to the Callaway Golf Company.

This problem also arises in other domains, e.g., the "Coulomb Potential" problem, where charged particles such as electrons constrained to a sphere try to push themselves away from

[34] See, for example, http://www.google.com/patents/download/7179178_Golf_ball_dimple_pattern.pdf?id=_fR-AAAAEBAJ&output=pdf&sig=ACfU3U3NaKCEViMk_xP4yUeVyfXDs1dsWw&source=gbs_overview_r&cad=0

each other (i.e., maximize the minimum distance between them) to reduce the total potential energy. They also arise in related areas of physics, such as "baryon density isosurfaces."[35]

While there are proposed solutions, the only solutions that have been proven to be correct[36] are for 1 to 12 circles (or spherical caps) and 24 circles. For the rest, there are only "best known" solutions.

If we take the best available solutions from Teshima and Ogawa and plot them, we see that with large circles there are some anomalies, but these subside and we end up with packing densities that are not too far from the theoretical optimum planar hexagonal packing density. A packing density of 1 occurs on the sphere only with one circle, analogous to achieving a packing density of one on an infinite plane with an infinite circle, or perhaps more exactly, achieving a packing density of 1 on a finite circular surface using exactly one circle.

As we get to a greater quantity of smaller circles, any enormous (relative to circle size) gap can be more or less filled, but ultimately the best (hexagonal) planar circle packing density of .91 can't be reached due to issues with the "kissing number."

The kissing number[37] is the maximum number of spheres that can simultaneously touch a given sphere (without overlapping), in a given number of dimensions. For 2-dimensional circles on a plane (or, say, the 8-ball relative to surrounding billiard balls before the break on a pool table), the answer is 6, in the well known hexagonal lattice configuration discussed above. But if the surface is convex, the answer is less than six. One can still achieve a very good start to some spherical packings using pentagonal instead of hexagonal close packing, but the challenges come in after the first perimeter is laid.

The gap is due to the curvature of the sphere, which prevents 6 caps from perfectly "kissing" one in the center. In fact, if the caps are large enough—namely hemispheres, of size π radians—only two can kiss, which perfectly covers the sphere. And if they are only half that size—namely $\frac{\pi}{2}$ radians—then only 4 can kiss any other cap, i.e., 4 around the circumference of a central one. At that size, the six circles defining the spherical caps can be thought of as circles maximally inscribed on the surface of a cube, or, equivalently, its dual, the vertices of a tetrahedron (two four-sided pyramids joined at their base).

Unlike in the plane, however, there can never be seven perfectly packed circles, i.e., six surrounding a central one. The way to realize this is to think of six steel bracelets surrounding one in the center, all of the same size, laying on a flat table top. If we try to make them curve at

[35] See "Figure 9: Baryon density isosurface for a B=97 Skyrmion with icosahedral symmetry." In Michael Atiyah and Paul Sutcliffe, "Polyhedra in Physics, Chemistry, and Geometry," at http://arxiv.org/PS_cache/math-ph/pdf/0303/0303071v1.pdf

[36] Yoshinori Teshima and Tohru Ogawa "Dense Packing of Equal Circles on a Sphere by the Minimum-Zenith Method: Symmetrical Arrangement," *Forma*, **15**, 347–364, 2000, at http://www.scipress.org/journals/forma/pdf/1504/15040347.pdf

[37] Florian Pfender and Gunter M. Ziegler, "Kissing numbers, sphere packings, and some unexpected proofs," Notices—American Mathematical Society," at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.90.7989&rep=rep1&type=pdf

all, at least two bracelets on the perimeter will need to overlap somewhat. To keep them from overlapping, we need to make them somewhat smaller, thus there will be larger gaps and thus a lower packing density than perfect hexagonal packing on a plane.

Plotting the best-known results compiled by Teshiwa and Ogawa leads us to a chart of packing densities as shown below. One can interpret this chart in a variety of ways. Clare and Kepert[38] claim that the packing density "is found to increase as the number of circles increases," but this is based on their observation of the then (1986) best-known packings for up to 40 circles. With the chart extended, the packing density appears to settle down at .85, somewhat short of the optimum planar packing density. It appears to be an open question whether this curve closes in on the planar optimum.

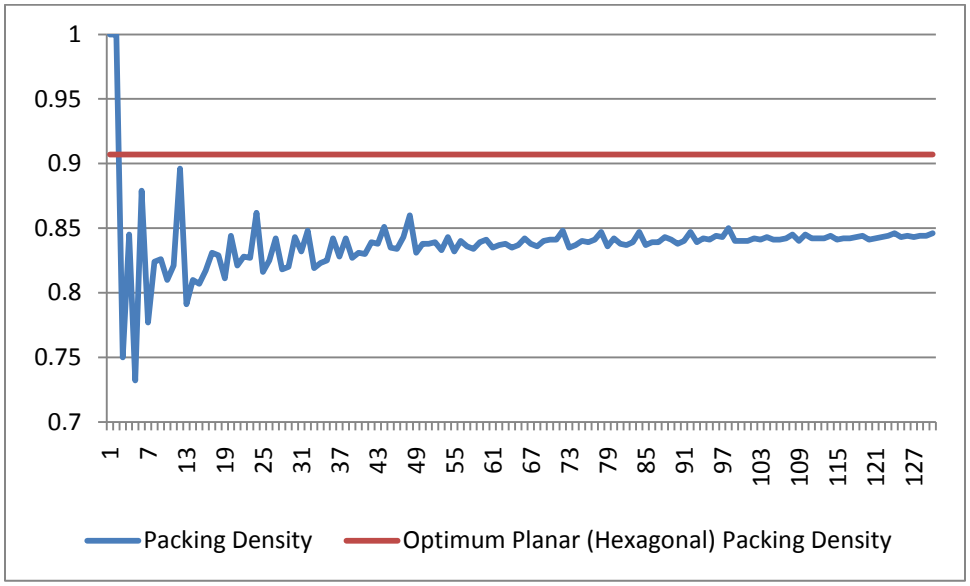

Figure 15: Best known packing densities for n circles on a sphere

If we examine the diameters of the circles in the best-known solutions, a few of which have been proven to be optimum, we can chart the data from Teshima and Ogawa as shown:

[38] B. W. Clare and D. L. Kepert, "The Closest Packing of Equal Circles on a Sphere," Proceedings of the Royal Society London A, 1986, 405, 329-344.

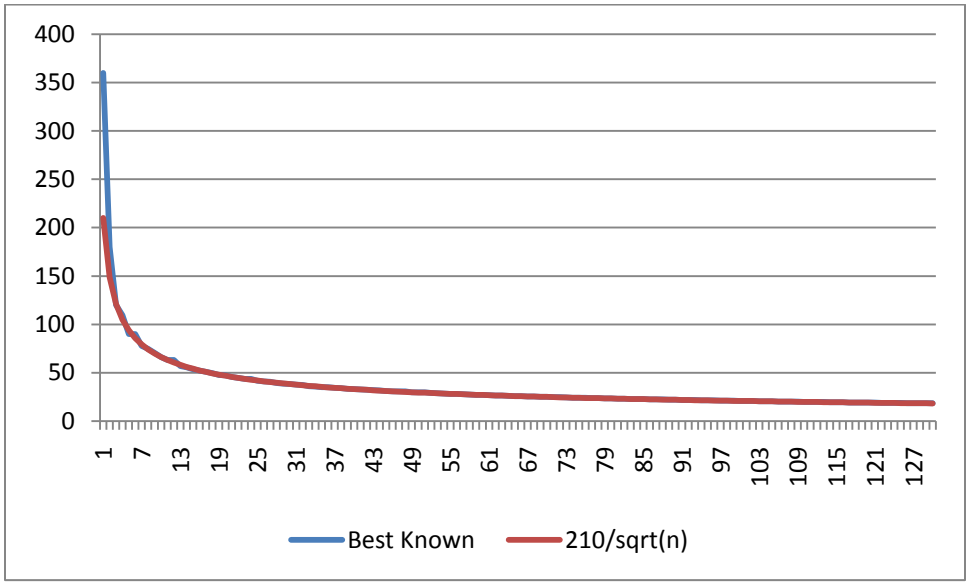

Figure 16: Diameters of closest-packed known solutions for n circles on a sphere

This chart shows, in blue, the largest possible diameters (in degrees) of the best known solutions. The red line overlays and obscures a matching blue line under it. These best-known solutions are: for one circle, $360°$; for two circles, the hemispheres at $180°$ each; for three circles, $120°$; for four circles, $109.471°$; then $90°$; $90°$; $77.87°$; $74.858°$; $70.529°$; $66.147°$; and so forth. Interestingly, once passing the somewhat anomalistic first few circles, the progression of diameters from then on closely matches $\frac{\sim 210°}{\sqrt{n}}$, as shown in the red line which exactly overlays the blue. In other words, the number of nodes that are required to cover an actual sphere in the best possible known close packing arrangement follows the $\frac{1}{\sqrt{n}}$ law. Why roughly $210°$ instead of $180°$? It appears to be due to the .85 value empirically realized in Figure 15 ($\frac{180}{210} \cong .85$).

One last note on empirical best packings: as described above, for ten circles, the largest possible diameter per circle is $66.147°$. The radius of that circle (curving along the surface of the sphere) is of course half that, or $33.074°$. At that size, the ratio $\frac{1-\cos(2\beta)}{1-\cos(\beta)}$ is 3.676, in other words, within 10% of the $\frac{1}{\sqrt{n}}$ rule. Once we get to twenty circles (i.e., nodes), the largest possible diameter per circle is $47.431°$. Half of that is $23.716°$. There, the ratio $\frac{1-\cos(2\beta)}{1-\cos(\beta)}$ is 3.831, or less than 5% off rule.

In other words, although the $\frac{1}{\sqrt{n}}$ rule was based on planar logic, even after accounting for distortions due to non-Euclidean behavior of spheres and due to arbitrary sphere packings that arise in solving the Tammes problem, the more nodes we place on the sphere the closer we get to the $\frac{1}{\sqrt{n}}$ rule, and it doesn't take very many nodes to get within a few percent of the rule.

We can also view this as saying that as we try to pack more spherical caps onto the surface of a sphere, their angular radius d must get smaller. Therefore β must get smaller, thus $\cos(\beta)$ larger and $R(1-\cos(\beta)) = h$ smaller. While $a = R\sin(\beta)$ also gets smaller, h gets smaller faster than a does, so the surface area of the cap $S = \pi(a^2 + h^2)$ becomes increasingly dominated by the a^2 term in the limit as β (and equivalently, d) approach 0. Colloquially, if d is small enough relative to R, the 3-dimensional "cap" of base radius a will appear to be a flat "disc" of radius a, and linear distance a will approach angular distance d. Consequently, since the area of each disc is, in the limit, πa^2, the number n of such discs that can be packed onto the surface is bounded by $n \times \pi d^2 \cong n \times \pi a^2 \leq \eta \times 4\pi R^2$, and thus in the limit, we again have $d \propto \frac{1}{\sqrt{n}}$.

To see how a increasingly dominates over h, we can look at the ratio $\frac{h}{a}$. We know that $\frac{h}{a} = \frac{R(1-\cos(\beta))}{R\sin(\beta)} = \frac{1-\cos(\beta)}{\sin(\beta)}$. We can now use a double-angle identity for cosine—namely $\cos(2\theta) = 1 - 2\sin^2(\theta)$ —and the double-angle identity for sine—namely that $\sin(2\theta) = 2\sin(\theta)\cos(\theta)$ — then rearrange terms and substitute $\theta = \alpha$ and $\beta = 2\alpha$, to determine that $1 - \cos(2\alpha) = 2\sin^2(\alpha)$ and therefore $\frac{h}{a} == \frac{1-\cos(2\alpha)}{\sin(2\alpha)} = \frac{2\sin^2(\alpha)}{\sin(2\alpha)} = \frac{2\sin^2(\alpha)}{2\sin(\alpha)\cos(\alpha)} = \frac{\sin(\alpha)}{\cos(\alpha)} = \tan(\alpha) = \tan\left(\frac{\beta}{2}\right)$. In the limit as $\beta \to 0$, $\frac{\beta}{2}$ also $\to 0$, and of course $\tan\left(\frac{\beta}{2}\right) \to 0$.

9. End-to-End Response Time Including Network Latency *and* Processing Latency

Assessing end-to-end response time is key to understanding the total user experience. Processing-only times can be extremely misleading. For example, consider a simple search query, documented in the Appendix. An informal experiment conducted on a Sunday afternoon in New Jersey used a Panasonic Lumix® DMC-ZS6 camera running in HD Video mode, an HP EliteBook laptop with an Intel® Core™ i5 CPU with a 2.53GHz clock speed and 4GB of RAM running Microsoft® Internet Explorer® 7.0.6002.18005 and Microsoft® Windows® Vista® Enterprise Service Pack 2, and a separate Apple® iPad® running the free BA.net "StopWatch Utility" app for use as a digital timer. The accuracy of the app was validated against a JavaScript applet using system time. A separate device was used to measure elapsed time so as not to reduce processor cycles available to the browser. The network connection used a home Wi-Fi network using a Cisco® Linksys® wireless access point, Cablevision High-Speed Internet Access, but no corporate VPN[39].

Briefly, here are the major events, including the two (in bold) which may be viewed as defining the total response time.

[39] All trademarks and registered trademarks belong to their respective owners.

Digital Stopwatch Time	Elapsed Time	Event
0.00		Timer Started
2.52	-.05	"Enter Key" Depressed
2.57	**0.00**	**"Enter" Key Down**
2.62	0.05	"Enter" Key Released
3.13	0.56	Suggested Search Terms Cleared
3.18	0.61	"Waiting for http://..."
3.98	1.41	Progress Bar begins to draw
4.89	2.32	Query Page begins to erase
4.89-4.94	2.32-2.37	Transition from query to results
4.94	**2.37**	**Search results available ("0.05 seconds")**

Figure 17: Elapsed Times for an Example Query

The executive summary: a query which was reported to take .05 seconds (50 milliseconds) actually took about fifty times as long, as measured from pressing the "Enter" key to when search results actually appeared. Such an experiment can be informally done by hitting the enter key and then counting "One Mississippi, Two Mississippi" before seeing the search results appear. One can argue that there are faster processors, faster operating systems, faster browsers, and faster Internet connections, but the inarguable point is that there is a gap between the end-to-end response time defining the customer experience and the portion dedicated to processing.

We can rephrase Amdahl's Law in terms of time required rather than speedup. Let the portion of a task that is parallelizable take time P on a single processor and the portion of a task that is non-parallelizable take time S on a single processor, and let the number of processors be p. By the 7th Law of Cloudonomics (Space-Time is a Continuum), we know that for the parallelizable portion the time required is $\frac{P}{p}$. Including the non-parallelizable portion, the total compute time T_c required for the task is $T_c = \frac{P}{p} + S$.

Amdahl's Law relates the speedup due to parallel processing to the number of processors, recognizing that only a portion of the code may be parallel, and the inherently serial code is not subject to any speedup. In a similar way, we can look at end-to-end round-trip time just due to network latency, T_N. We also must realize that moving processing or service nodes closer or farther away will not impact the total time required for local endpoint tasks, for example, browser page rendering, or service node tasks, for example, processing search queries.

We can thus state an end-to-end law for response time, which combines the 7th Law of Cloudonomics (Space-Time is a Continuum) modified to account for the serial portion of the work, which is thus a variation of Amdahl's Law, with the 8th Law of Cloudonomics on distributed nodes.

First, we may argue that when we divide up a task of size P across p processors, the total compute load or work load W per processor generally follows

$$W = k_0 + k_1 t + k_2 \frac{P}{p} + k_3(p-1)$$

The first constant k_0 is a fixed amount of work per processor, for example, initializing local variables. The second term $k_1 t$ is based on the time spent, e.g., CPU cycles devoted to running the operating system or hypervisor. If t is the elapsed time, it may be proportional to $\frac{P}{p}$, but the other coefficients k_2 and k_3 may impact it further. The third term $k_2 \frac{P}{p}$ is the application-dependent work performed by the processor. The fourth term $k_3(p-1)$ is based on communications or calculations that may need to exist with the other processors running portions of the application. This inter-process(or) communication can potentially cause damaging overhead costs: it is clear that if k_3 is non-zero, or even non-trivial, the last term will dominate. To put it differently, we want our processors to spend their time working, not coordinating. However, there are many types of computations that have no such k_3 term. One of the most intensive compute tasks today is undoubtedly responding to arbitrary queries by searching all the world's information, even when that information has been pre-indexed. However, the individual tasks associated with processing search queries are highly parallelizable: on the order of 1,000 machines may be involved to process a query. We can also frequently ignore the second term, by removing such overhead from *all* processors, and, in effect, rather than considering the total cycles available on a processor, considering only the total cycles available for computation. Finally, while the first term contributes to the total amount of work, because these operations can be conducted in parallel it does not change the total time requirement.

For an interactive request-response task served by the cloud, where sufficient bandwidth exists to not impact latency, if the amount of time required for local endpoint processing is E and there are n service nodes which can respond to the task in accordance with a parallelizable portion and a serial portion, then the total time T required is

$$T = E + \left[2 \times \frac{k_4}{\sqrt{n}}\right] + \left[\frac{P}{p} + S\right]$$

We can simplify this, by viewing the sum of the two constants $E + S$ as being a fixed quantity F, and replacing the $2 \times k_4$ by a new constant N, to arrive at:

The Law of Cloud Response Time: The response time for an interactive transaction served by a distributed, elastic cloud is

$$T = F + \frac{N}{\sqrt{n}} + \frac{P}{p}$$

When $n = 1$, the second component is just N, which of course corresponds to the worst-case round-trip latency from the most distant user to the single service node. The second and third

terms are similar, in a way. P is the processing time on a single processor, N is the (round-trip) network time for a single serving node. Of course, the two terms are also different, in as much $\frac{1}{\sqrt{n}}$ decreases more slowly than $\frac{1}{p}$.

As we've discussed earlier, there is distortion due to the curvature on the surface of a sphere, but this distortion drops to insignificance as the number of nodes passes a dozen or so. As mentioned earlier, ten nodes under the best-known spherical cap packing is within 10% of the value predicted by the formula above, and twenty nodes under the best-known spherical cap packing is within 5% of the value predicted above.

Among other things, this says that an investment in processors pays off differently than an investment in dispersion. However, it is a mistake to only focus on parallelization at the expense of dispersion, as we shall see next.

10. The Optimal Balance of Owned Resources

Suppose we have been given a quantity of Q total processors, and are trying to minimize the latency function T. We could put all Q processors in one location, or we could put one processor in each of Q evenly dispersed locations, or we could do something in between. It turns out that the optimum balance depends on Q, P, and N.

> ***Proposition 4:*** For an application with processing time P and single-node network latency N, if Q processors are evenly distributed at n well-dispersed nodes, the total latency function $T = F + \frac{N}{\sqrt{n}} + \frac{P}{p}$ is minimized when $n = \sqrt[3]{(\frac{QN}{2P})^2}$ nodes.
>
> ***Proof:*** Let the latency function T be defined as above. Since Q processors are evenly deployed at n nodes, we have $p = \frac{Q}{n}$ processors at each node. Therefore, the latency is
>
> $$T = F + \frac{N}{\sqrt{n}} + \frac{P}{p} = F + \frac{N}{\sqrt{n}} + \frac{P}{(\frac{Q}{n})} = F + \frac{N}{\sqrt{n}} + \frac{Pn}{Q}$$
>
> Thus
>
> $$T = F + N \times n^{-\frac{1}{2}} + \frac{P}{Q} \times n$$
>
> However, this is minimized when $\frac{dT}{dn} = 0$. Taking the derivative of T with respect to n gives us
>
> $$T'(n) = 0 + \frac{-N}{2} \times n^{-\frac{3}{2}} + \frac{P}{Q}$$

Then

$$0 = \frac{-N}{2} \times n^{-\frac{3}{2}} + \frac{P}{Q}$$

So

$$\frac{N}{2} \times n^{-\frac{3}{2}} = \frac{P}{Q}$$

So then

$$n^{-\frac{3}{2}} = \frac{2P}{QN}$$

Which means

$$\frac{1}{n^{\frac{3}{2}}} = \frac{2P}{QN}$$

And therefore

$$n^{\frac{3}{2}} = \frac{QN}{2P}$$

Taking the $\frac{2}{3}$ root of both sides gives us

$$n = \sqrt[3]{(\frac{QN}{2P})^{-2}} \quad \blacksquare$$

11. The Economics of Cloud Response Time Reduction

We have looked at the number of processors and the number of nodes and how they impact end-to-end response time. In traditional environments, deploying more processors requires more investment, and deploying more nodes requires more investment as well, for example, building switching centers or data centers or other nodes with the appropriate cooling, power, physical security, management, and so forth.

Using pay-per-use cloud services though, we can see that response time reduction does not necessarily impact cost.[40]

[40] Joe Weinman, "Cloud Economics and the Customer Experience, InformationWeek, March 24, 2011, at http://www.informationweek.com/news/cloud-computing/infrastructure/229400200?pgno=1

Proposition 5: The marginal cost of parallelization in the cloud is zero, i.e., for a parallelizable process of size P that may run on either p_1 processors at cost c_1 or p_2 processors at cost c_2, $c_1 = c_2$.

Proof: Let the cost in the cloud of a processor per unit time be c. The time to run a process of size P on p_1 processors is t_1. The time to run a process on p_2 processors is t_2. The cost c_1 for the first case is $c \times p_1 \times t_1$. The cost for the second case is $c_2 = c \times p_2 \times t_2$. But $t_1 = \frac{P}{p_1}$, and $t_2 = \frac{P}{p_2}$, so

$$c_1 = c \times p_1 \times t_1 = c \times p_1 \times \frac{P}{p_1} = c \times P = c \times p_2 \times \frac{P}{p_2} = c \times p_2 \times t_2 = c_2 \blacksquare$$

In distributing nodes, there are two components to consider, the data component and the processing component.

If data must be replicated at each node, then since diminishing latency reduction requires dramatic growth in nodes, the costs of data replication also increase dramatically. To put it another way, to gain incremental, rapidly diminishing returns requires exponentially greater investments.

However, from a processing standpoint, the costs remain constant, because the growth in nodes corresponds to an equal and opposite reduction in users, assuming uniform density of users across the surface. For example, if one node serves a million users, one thousand nodes only need to serve 1000 users each. Assuming that each user generates an equal amount of work—or expected value of work—the cost of processing can remain constant.

Proposition 6: The marginal processing cost of node dispersion in a pay-per-use cloud is zero, i.e., for a given number of users U each generating work u, that may run on n_1 nodes at cost c_1 or n_2 nodes at cost c_2, $c_1 = c_2$.

Proof: Let the cost in the cloud of a processor per unit time be c. Let the quantity of work required for a single user transaction on 1 processor be q, and the expected number of transactions per user be m. With n_1 nodes, the expected number of users at any node is $\frac{U}{n_1}$, the work for each of those users is $q \times m$ and thus the total work at each node is $\frac{U}{n_1} q \times m$, and thus the total cost at each node is $\frac{U}{n_1} q \times m \times c$. Therefore, the total cost c_1 across all nodes is $n_1 \times \frac{U}{n_1} q \times m \times c$. Using this logic, we see that

$$c_1 = n_1 \times \frac{U}{n_1} q \times m \times c = U \times q \times m \times c = n_2 \times \frac{U}{n_2} q \times m \times c = c_2 \blacksquare$$

Ultimately then, the cloud offers the tantalizing possibility of latency reduction at zero marginal cost, with the ultimate question being the degree of data replication.

For some applications, the cost of data management can be partitioned based on geographic localization of the data, and the same "zero marginal cost" rule applies. There is no cost

difference between using 100 gigabytes of storage in a single location to store 1 gigabyte for each of 100 users, or 10 gigabytes in ten different locations, or 1 gigabyte in 100 locations, when storage services are priced by the gigabyte per month.

For other applications, data can be managed centrally even though processing may be dispersed for latency reduction.

And for others, although the processing costs may be intransitive, the storage costs (of replicas) may grow in accordance with the n^2 growth in nodes.

12. Conclusion

The results here are couched within the domain of Cloud Computing, but apply more broadly than that. Consider another request-response model: ordering a cup of coffee from a coffee shop chain. To speed a cup of steaming Java (coffee) to your hand is not much different than speeding hot Java (the programming language) into your browser. Coffee shop chains use geographical dispersion of service nodes—more commonly thought of as a coffee shop on every corner—together with (somewhat) optimized parallelism via multiple coffee-making processors to reduce the total time. The time it takes—once you decide to get a cup of coffee—to take the first sip depends on network latency—how long it takes to walk or drive to the nearest coffee shop—as well as processing latency—how long it takes to process the order and brew the cup, so that there can be a response to your request for a cup of coffee.

The resourcing tools available to the major coffee chains include building more coffee shops in more places, to reduce network latency, and staffing the coffee shops with more baristas and espresso machines, to increase parallelism and thereby speedup the end-to-end process.

The real world has an irritating way of being more complex than such a law as proposed here can account for. Different network technologies such as Ultra Long Haul vs. local or metro area networks, delays due to OEO (optical-electronic-optical) conversions, stochastic differences in the routes that individual packets may take, are a few examples. Also, network routes are not of homogeneous density: subsea cable routes follow certain regular paths, e.g., around the Persian Gulf or through Hawaii, U.S. fiber routes have a strong East-West component but a weaker North-South component. Network latencies depend on network congestion. The same complexities exist on the processing side: virtual machines may be running on physical resources that are congested due to other applications, and some parallelizable applications may have an interprocessor communication overhead that is order $O(p)$ or even $O(p^2)$.

Generally speaking, however, the law proposed here provides a useful model for thinking about the rate at which investments in parallelization and in dispersion can pay off, and provides a useful formula for determining the optimum trade-off between parallelization and dispersion. It also supports the underlying architectural and business model of the cloud, since pay-per-use

resources enable processing speed-up at no additional cost, and multi-tenancy or other resource-sharing approaches enable developers to leverage the dispersion of the cloud in a more cost-effective manner. We haven't dealt in this paper with yet another component of latency: I/O access. Interestingly, for large scale implementations that we have been discussing, such as search, a substantial portion of data is moving into memory[41] from storage, thus reducing access times dramatically.

In *Casablanca*, Sam (Dooley Wilson) sings Herman Hupfeld's "As Time Goes By,"[42] claiming that "This day and age we're living in/Gives cause for apprehension/With speed and new invention /And things like fourth dimension…And no matter what the progress/Or what may yet be proved …You must remember this/A kiss is just a kiss/a sigh is just a sigh/The fundamental things apply/As time goes by."

In this paper we have demonstrated that, regardless of what may yet be proved, for new inventions such as the global cloud there is no need for apprehension regarding the speed of cloud response time, since given the results for kissing numbers and assuming that a ψ is just a ψ, the fundamental things that apply as time goes by are network latency and processing time.

[41] Jeffrey Dean, "Challenges in Building Large Scale Information Retrieval Systems," WSDM 2009, Second ACM International Conference on Web Search and Data Mining, February, 2009, http://research.google.com/people/jeff/WSDM09-keynote.pdf

[42] © 1931 Warner Bros. Music Corporation, ASCAP

13. APPENDIX: An Example of "End-to-End" Request-Response Time vs. "Processing" Time