International Journal of Information Science and Management

Hard Limits on the Growth of the Internet and Computing Capacity

B. Gavish,Ph.D. Professsor
Southern Methodist University,USA
Corresponding Author: Gavishb@mail.cox.smu.edu

Abstract

The last few years have seen an explosion in the deployment and use of the Internet, networking and telecommunication technologies. This was followed by significant increases in the speed and capacity of computing, for example Petaflop supercomputers are becoming common. We will examine some of the developments; explain their importance and potential impact. Many forecasts and predictions have been made about the impact of the increases of computing capacity and the growth of the Internet and the world wide web. In this talk we will introduce some of the favorite predictions and will analyze the possibilities for their realization in the long run. The analysis shows that there exist hard limits on the growth of the Internet and the increase in computing capacity. They prove that it is unlikely that some of the predictions will hold in the long run. The restrictions are based on basic physical and economic limitations, which generate tight bounds on the realization of such predictions. The bounds will occur much faster than expected by the simple forecasters.

Data Explosion on the Internet

Vast amounts of data are generated and stored on storage devices that are accessible to users through the Internet. The data consists of data files and data bases, audio and video files (music, telephony, movies, video clips etc.), programs, messages and email to name a few. A single retailer Walmart for example, generated in 2007 one Terabytes of transactional data a day (Kirk, 2007). Lyman and Varian (Lyman & Hal ,2003) estimated that the amount of yearly data generated per person on earth is around 250 megabytes, this number clearly increased in the last five years and is significantly higher today. The same study estimated that the total amount of all types of data generated in 2002 was equal to 5 Exabytes. The vast amount of data of data linked to the internet is estimated in August 2009 to be around 10^{22} bits of data, this estimate makes sense if we consider that over one Billion PCs are connected to the internet and a PC can store between one to twenty Terabits of data on its hard disks (internal and external).

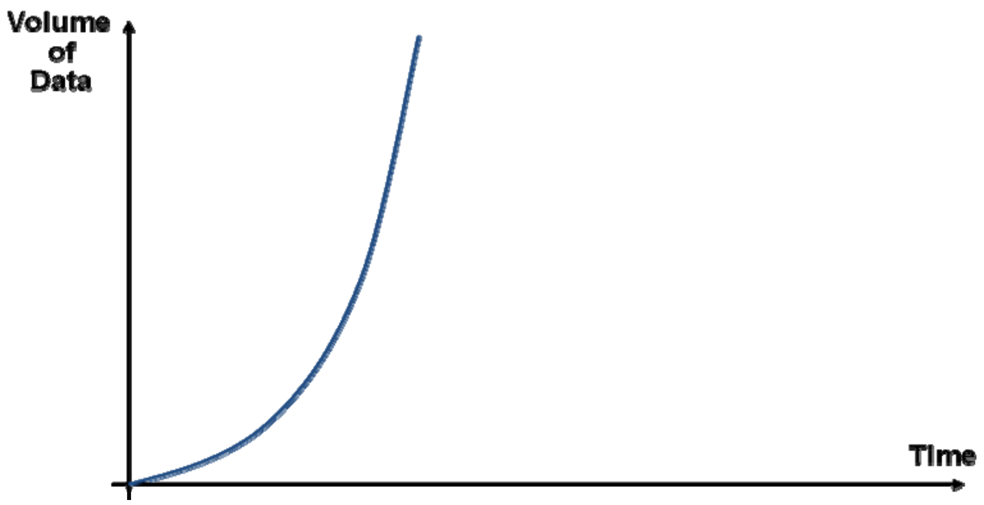

Figure 1: Projection of Volume of data on the Internet as a function of time (exponential growth)

The vast amounts of data accessible today by Internet users are infinitesimal when we consider the estimates regarding future data explosion on the Internet. An IDC study (Kirk,2007) estimated the annual data generation/duplication rate at 161 Exabytes in 2006/7, the same study estimates that the number will grow to 988 Exabytes in 2010 (or 1 Zettabyte) Considering that in 2003 the estimate was 5 Exabytes (Lyman & Varian,2003), it is safe to assume that the annual data generation/duplication rate grows exponentially as a function of time. Various estimates have been made regarding the amount of data that is stored on the Internet , they estimate that the amount of data (in bits) doubles every few months (between 2 and 4 months, depending on who makes the claim).If the estimates are correct, than we expect that the graph of the amount of data stored on the Internet will grow exponentially as a function of time exhibiting a behavior close to the graph shown in Figure1.

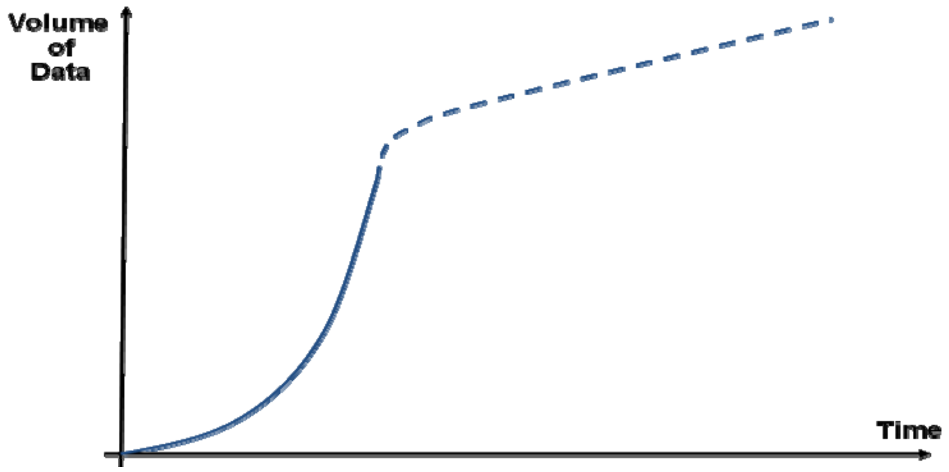

Figure 2: Volume of data on the Internet as a function of time (the leveling off effect)

In what follows we show using simple arguments that in the long run, such data explosion can't continue indefinitely and that the relationship between the volumes of data on the internet as a function of time will level off and will exhibit a behavior closer to the one shown in Figure 2.

The first set of arguments is based on the observation that data is generated by human beings; therefore the amount of data added to the internet is proportional to the number of people linked to the internet. Since the number of people on the globe is finite (and grows in a very low rate), at some point the rate in which data is added to the internet will slow down to an almost linear rate and not the exponential rate exhibited in figure 1.

The counter argument is based on several observations:

1. There are automatic methods which generate data and add it to the data stored on the internet, they include

   a. Sensors that measure and transmit/add their measurements to the data available on the Internet (Morita, Aikebaier, Enokido & Takizawa ,2008); Royo, Olivares & Orozco-Barbosa ,2009), examples include temperature, traffic, heart rate, humidity, wind velocity, barometric pressure, noise levels, water levels, to name a few.

   b. Recording of video by self guided cameras and sending it to the internet (Casaca, Silva, Grilo, Nunes, Presutto & Rebelo,2007; Chiasserini & Magli ,2004), for example security cameras, speeding traffic cameras, traffic flow cameras, patient/child monitoring cameras, etc.

The number of such sensors and recording devices can exceed by far the number of people on earth, and will generate data at a much higher data rate than the one estimated by the number of people.

2. Another factor that has a higher than linear data generation capabilities is based on the observation, that a single human being might generate the original data (a video clip, a PowerPoint presentation, a document), he sends it to a single individual, but that individual likes it and sends it to a mailing list/exploder, if many members of the list repeat the same behavior and retransmit the data to their private mailing lists, a single data generation activity might result in a much larger amount of data stored on the Internet.

3. The third argument is based on the ability of human beings to develop and have automatic agents on the internet (Gams, 2001; Yiming, Jiming & Moukas, 2001), the number of such agents per person is not bounded, and they can and will be able to perform complex tasks that will add to the amount of data collected and stored on the Internet.

All of the above may lead to linear or higher than linear growth in the amount of data collected and stored on the Internet, but it does not imply that an upper bound exists on the amount of data that can be stored. The bound developed below is based on physical arguments that put a limit on how much data (in bits) can be stored on the Internet or on all storing devices on earth.

The argument is developed as follows, data has to be stored somewhere, it is represented in bits. For obtaining an upper bound we assume that a bit is represented by an atom. Under this assumption an upper bound will be the number of atoms on earth. It turns out that such a limit can be computed and is bounded by $2^{165} \approx 10^{50}$ atoms, this includes all the mass of earth (magma, air, living things, plants and water) most of which are unlikely to be used for data storage.

The present amount of data (in bits) on the Internet is estimated at $10^{18} \approx 2^{60}$ bits, even if every atom on earth is used to represent data, an upper bound on future expansion is 10^{32}. This bound does not take into consideration that for each bit stored we need additional bits to be able to find the data, retrieve and process it. Assuming that the present rate of data doubling on the internet is doubling every 4 months, it leaves room for 165-60=105 generations of data doubling, or 420 months of data doubling. This implies that within 420/12=35 years, all atoms on earth will be consumed for data storage, an unlikely event. Thus it is clear that at some point in the near future the rate of data generation on the Internet will slow down. It is interesting to note that if we assume that a bit is represented by an electron and its state in an atom, adds just a few generations to the total number of generations in the above analysis, it will probably take 40 years instead of 35 years to exhaust all electrons and their states.

The above simple analysis also implies that at some point in time individuals, governments and businesses will have to decide what data to store, and what data to eliminate and use the freed resources to store new data. Since we will be dealing in vast amounts of data, methodologies and procedures will have to be developed to make such decisions in an efficient way. Most of the research effort today is devoted to how to generate data and how to store and manipulate it. Research effort should be devoted to help in deciding on what data to keep and what previously stored data to eliminate.

Combinatorial Optimization and Computing Capacity

Many combinatorial optimization problems require a complete enumeration to compute and identify their optimal solution. The algorithm for solving such problems may have a time complexity of $(n!)$ where n is the problem size. The Traveling salesman problem (TSP) is such a problem, a traveling salesman that is located in a home city 0 is required to visit n cities and return to his home city, each city has to be visited exactly once, the tour forms a cycle. We are given the distances between city pairs, and the objective is to find a sequence of visiting the cities so that the overall distance traveled by the salesman is minimized. A solution to the TSP can be computed by generating all the $n!$ permutations of visiting the cities, computing the distance traveled by each sequence of cities and picking the tour with the minimal associated cost. Using the TSP as an example and assuming that 10 instructions have to be executed for each evaluation of the objective function during the enumeration process. Instead of computing speed, we use the term of computing capacity, which is the number of instructions per second that can be executed by the computing facility, we prefer to use computing capacity as the measure as it is not architecture dependent and covers uniprocessors, multiprocessors, networked, decentralized and distributed processing. It turns out that the solution procedure to many hard combinatorial optimization problems can be allocated without much difficulty to multiple processors, each dealing with a specific set of instances. For example for the TSP, a processor can be allocated a partial tour consisting of a subset of cities, and concentrate on that specific set of cases.

Table 1 displays the maximal problem size (number of cities) solved by a complete enumeration for different computer capacities (instructions per second) and different elapsed times devoted to the solution process.

Most textbooks dealing with combinatorial optimization or computational complexity list problems consisting of size 13 to15 as the feasible range of problem sizes that can be solved today by complete enumeration. This is a pessimistic limit considering that at present, top super computers can execute instructions at a rate of over a Petaflop (10^{15}) Bailey (Bailey, 1997) instructions per second, putting the effective size of data independent TSP problems that can be solved today to optimality in the range of 18 to 21 cities (depending on the amount of elapsed time allocated to the solution procedure).

Table 1 shows that as computing capacity increases, so will the size of problems that can be solved by complete enumeration. Computers with a computing capacity of slightly over 10^{150} instructions per second will be able to solve problems to optimality consisting of 100 cities using complete enumeration. This is an encouraging projection considering the present state of the art that is significantly lower.

Table 1: The maximal traveling salesman problem size solvable using complete enumeration for different computing capacities and elapsed times

Number of operations per second	**Largest Problem Size Solved in a**							
	Min	**Hour**	**Day**	**Week**	**Month**	**Year**	**Decade**	**Century**
Seconds	60	3,600	86,400	604,800	2,592,000	31,104,000	311,040,000	3,110,400,000
10^3	7	8	10	11	11	12	13	14
10^6	10	11	13	13	14	15	16	16

10^{9}	12	14	15	16	16	17	18	19
10^{12}	15	17	18	18	19	20	20	21
10^{15}	17	19	20	21	21	22	23	23
10^{18}	20	21	22	23	23	24	25	25
10^{20}	21	23	24	24	25	25	26	27
10^{25}	25	26	27	28	28	29	30	30
10^{30}	28	30	31	31	31	32	33	34
10^{40}	35	36	37	38	38	39	39	40
10^{50}	41	42	43	44	44	45	45	46
10^{75}	56	57	58	58	59	59	60	60
10^{100}	70	71	72	72	72	73	73	74
10^{110}	75	76	77	77	78	78	79	79
10^{120}	81	81	82	83	83	83	84	85
10^{150}	96	97	97	98	98	99	99	100
10^{175}	108	109	110	110	111	111	112	112
10^{200}	120	121	122	122	123	123	124	124
10^{250}	144	145	145	146	146	147	147	148
10^{300}	167	168	168	169	169	169	169	170

When will such computers exist? Moore's law (Moore,1965) states that computing speed/capacity doubles roughly every 1.5 years, which is equivalent to a factor of 10^{3} every 15 years. Assuming that an identical ratio of improvement will continue into the next centuries, we should expect computers with a capacity of 10^{30} instructions per second to become available in the next 150 years. Grice (Grice, 2009) provides an empirical estimate on the advancement of supercomputers; his estimate shows that top of the line super computers advance in computing capacity by a factor of roughly 10^{3} every 10 to 11 years. Using Grice computing capacity advancement estimate, we should expect to reach computing capacities of 10^{30} instructions per second in the next 110 years. From table 1 follows that the availability of such computers will increase significantly the size of problems that can routinely be solved to optimality. It turns out that this is an optimistic view, as we will show in the next section it is unlikely that such a computer will exist (if the present laws of physics continue to hold).

Energy Based Bounds on Computing Capacity

Energy based bounds on computing capacity are based on the amount of energy needed to execute a given number of instructions per second, To develop such a bound we try to estimate a lower bound on the minimal amount of energy needed to operate on a single bit. The operations performed on a single bit can be reduced to two types only 1) probing a bit to detect if it is zero or one, and 2) changing (flipping) the value of that bit from zero to one, or from one to zero. The amount of energy required to probe a bit is probably lower than the amount of energy required to flip a bit. To simplify the analysis that follows we use the minimal value of the two as the incremental energy level per bit operation.

The first lower bound assumes that at some point in the future, computers will be constructed such that a bit will be presented by a single electron and its position in an atom. Let q be equal to the lower bound on the minimum amount of energy required to probe a bit as to its value (0 or 1) or to flip its value from 0 to 1 or from 1 to 0. Such bound is obtained by using the minimal amount of energy required to excite an electron from its present state to its next state or the minimal energy level carried by a photon.

There are several possible ways to define such a lower bound, Planck constant (Planck,1901) (= 6.626 x 10^{-34}Joules) can be assumed to be such a lower bound.

Photons are emitted when an electron in some atom moves from a higher energy state to a lower energy state. The photon has a wavelength which is exactly equal to the energy difference between the two states. The energy of a photon is equal to Planck's constant (h) times its frequency (f)

$$E_{photon} = hf$$

Since frequency times the wavelength (w) = speed of light (c) for any photon then we have (Ritz, 1908)

$$E_{photon} = hc/w$$

Thus the energy of a photon is inversely proportional to its wavelength. Short wavelength photons have more energy than long wavelength photons.

Assuming Planck's constant (h) is the lowest amount of energy required for probing or flipping a bit in a computer. A lower bound on the amount of energy required to execute instructions on a computer performing I instructions per second is given as:

$$E(I) = I \cdot \varphi \cdot h$$

Where φ is the average number of probes and/or flip-flops per instruction.

Table 2 shows the lower bound on the amount of energy required when we assume that Planck constant is used to probe/flip a bit and the computer performs I instructions per second. In what follows we assume that the number of bit level operations per instruction is $\varphi = 40$. The length of a word on a processor is at least 32 bits. Probing each bit as to its value implies 32 bit level probes; on top of it some logic operations have to be performed so all together a safe lower bound estimate per computing instruction is 40 bit level operations.

Today's top of the line supercomputers (the Roadrunner) (Shannon , Murphy , Niemier , Izaguirre & Kogge ; Grice,2009; Haviv,2006) perform a Petaflop of operations per second and consume 2 to 4 Megawatts to achieve that level of computing capacity (Grice,2009). Comparing this level of power consumption to the corresponding 10^{15} entry in Table 2 shows that the Planck based lower bounds underestimates the power consumption by a wide margin. Allowing for advancement in architecture and manufacturing technologies, Grice estimates that in the next ten years, the first Exaflop computer will be built and will consume 20 Megawatts of power which is a huge factor above the lower bound estimate computed in Table 2 for 10^{18} instructions per second computing capacity. This two data points lead us to believe that unless major breakthroughs will be made in the engineering, packaging and manufacturing technologies, power consumption and energy related bounds will become the main block for the advancement of supercomputing.

Table 2: A lower bound on the amount of energy consumed by computers of different capacities using Planck's constant

Number of operations per second	Minimal Energy needed per second in Joules=Watts/second	Minimal Energy needed in Watts/hour	Minimal Energy needed in TeraWatts/Hour
10^{3}	2.6505E-29	9.5417E-26	9.54173E-38
10^{6}	2.6505E-26	7.36244E-42	7.36244E-54
10^{9}	2.6505E-23	7.36244E-39	7.36244E-51
10^{12}	2.6505E-20	7.36244E-36	7.36244E-48
10^{15}	2.6505E-17	7.36244E-33	7.36244E-45
10^{18}	2.6505E-14	7.36244E-30	7.36244E-42
10^{20}	2.6505E-12	7.36244E-28	7.36244E-40
10^{25}	2.6505E-07	7.36244E-23	7.36244E-35
10^{30}	2.6505E-02	7.36244E-18	7.36244E-30
10^{36}	2.6505E+04	7.36244E-12	7.36244E-24
10^{37}	2.6505E+05	7.36244E-11	7.36244E-23
10^{38}	2.6505E+06	7.36244E-10	7.36244E-22
10^{39}	2.6505E+07	7.36244E-09	7.36244E-21
10^{40}	2.6505E+08	7.36244E-08	7.36244E-20
10^{50}	2.6505E+18	736.2444444	7.36244E-10
10^{55}	2.6505E+23	73624444.44	7.36244E-05
10^{60}	2.6505E+28	7.36244E+12	7.362444444
10^{61}	2.6505E+29	7.36244E+13	73.62444444
10^{62}	2.6505E+30	7.36244E+14	736.2444444
10^{63}	2.6505E+31	7.36244E+15	7362.444444
10^{64}	2.6505E+32	7.36244E+16	73624.44444
10^{65}	2.6505E+33	7.36244E+17	736244.4444
10^{75}	2.6505E+43	7.36244E+27	7.36244E+15
10^{100}	2.6505E+68	7.36244E+52	7.36244E+40
10^{110}	2.6505E+78	7.36244E+62	7.36244E+50
10^{120}	2.6505E+88	7.36244E+72	7.36244E+60
10^{130}	2.6505E+98	7.36244E+82	7.36244E+70

In order to comprehend what the numbers in table 2 imply consider that the total amount of electricity generated in the USA during the month of March 2009 was equal to 310.024TeraWatts. This illustrates that using Planck constant it is unlikely that a computer with a computing capacity greater than 10^{40} Instructions per second will be constructed[1].

When comparing power consumption of processors and supercomputers that exist in the marketplace to the estimates based on Planck constant based projections, the lower bounds on the amount of energy consumed by a computer with a given computing capacity produce lower bounds that are much lower than the expected amount of energy required when such computers will be built. In this section we develop an alternate method for computing such bounds that produce lower bounds that are much closer to the real energy requirements of such supercomputers.

The improved bound is based on the amount of energy released when an electron transitions in an atom from an $m-1$ position to a neighbor position m. Rydberg Formula (Bohr,1985 ; Ritz,1908) provides the amount of energy needed for such a change in state (in the following formula E_0 is equal to 13.6 in electron volts)

$$E_m = E_0\left(\frac{1}{(m-1)^2} - \frac{1}{m^2}\right) \qquad m \geq 2 \qquad (1)$$

From formula (1) follows that as m increases, the marginal energy required to move between two successive positions decreases. At temperatures above the absolute zero, when the level of energy is too low, the electron is no longer attached to an atom and is no longer in a stable position. This happens when $m \geq 100$, thus E_{100} can be used as a lower bound on the minimum energy level required to probe or flip a bit. Assuming that future computers will be built in which manipulations on a single electron will be equivalent to a bit level operation and using Rydberg Formula $E_{100} = 2.7614 \cdot 10^{-5}$eV (electronVolts), which translates to $4.4237 \cdot 10^{-24}$ Joules. Table 3 lists the energy consumed by such supercomputers for different computing capacities (in instructions per second) using $m = 100$, when 40 bits per instruction have to be probed/flipped.

It looks that when the electrons based bound is used, a computer performing at a rate of 10^{18} instructions per second consumes more than the total electricity output of the United States, and thus it is unlikely that such a computer will be engineered and built.

The bound is much tighter when we take into consideration; that such levels of energy consumption generate enormous amounts of heat; the heat has to be dispersed. How to disperse such quantities of heat is well beyond foreseeable cooling and heat dissipation capabilities (Bailey, 1997; Grice, 2009).

[1] This is true with the present laws of physics; new laws of physics have to be discovered in order for such limits to be removed.

It is clear from the entries in Tables 2 and 3 that heat dissipation and the amount of energy consumed by super computers is of a major concern. How to transfer the enormous amounts of heat generated by the processors, memory and storage devices and cool the system is a major concern of any present and future engineering of super high capacity computers (Grice ,2009 ; Bailey,1997).

When dealing with combinatorial optimization problems that are naturally divisible, it is possible to reduce the heat dissipation issue by relying on networked computers in which billions of computers/processors distributed all over the globe cooperate in solving the problem. Such a decentralized/ distributed approach reduces the heating/cooling problem. Energy consumption costs remains as a major issue, considering the levels of energy consumed by all those computers. If each personal computer consumes 50Watts, than one billion personal computers operating for one hour will consume 50,000 MegaWatt-Hours. Considering the energy costs and doing the cost benefit analysis will preclude the use of such distributed computing mechanisms as a practical tool for solving many classes of combinatorial optimization problems.

Table 3: A lower bound on the amount of energy consumed by computers of different capacities using Rydberg Formula

Number of operations per second	Minimal Energy needed per second in Joules	Minimal Energy needed in TeraWatts/Hour
10^{3}	1.7695E-19	4.9152E-35
10^{6}	1.7695E-16	4.9152E-32
10^{9}	1.7695E-13	4.9152E-29
10^{12}	1.7695E-10	4.9152E-26
10^{15}	1.7695E-07	4.9152E-23
10^{18}	1.7695E-04	4.9152E-20
10^{20}	1.7695E-02	4.9152E-18
10^{25}	1.7695E+03	4.9152E-13
10^{30}	1.7695E+08	4.9152E-08
10^{31}	1.7695E+09	4.9152E-07
10^{32}	1.7695E+10	4.9152E-06
10^{33}	1.7695E+11	4.9152E-05
10^{34}	1.7695E+12	0.00049152
10^{35}	1.7695E+13	0.004915205
10^{36}	1.7695E+14	0.049152046
10^{37}	1.7695E+15	0.491520457
10^{38}	1.7695E+16	4.915204571
10^{39}	1.7695E+17	49.15204571
10^{40}	1.7695E+18	491.5204571
10^{44}	1.7695E+22	4915204.571

Number of operations per second	Minimal Energy needed per second in Joules	Minimal Energy needed in TeraWatts/Hour
10^{45}	1.7695E+23	49152045.71
10^{46}	1.7695E+24	491520457.1
10^{47}	1.7695E+25	4915204571
10^{48}	1.7695E+26	49152045710
10^{49}	1.7695E+27	4.9152E+11
10^{50}	1.7695E+28	4.9152E+12
10^{60}	1.7695E+38	4.9152E+22
10^{75}	1.7695E+53	4.9152E+37
10^{100}	1.7695E+78	4.9152E+62
10^{110}	1.7695E+88	4.9152E+72
10^{120}	1.7695E+98	4.9152E+82
10^{130}	1.7695E+108	4.9152E+92

Computing Capacity versus Algorithmic Development

Another direction that could be applied to some combinatorial optimization problems is to improve the algorithms used to solve the problems to optimality using clever algorithms that do not require an explicit complete enumeration of the problems. Since the original problems are NP-complete and in many cases NP-hard the algorithms are not polynomial in the problem size, but require significantly less computational effort when compared to complete enumeration.

Using the Traveling salesman problem as an example, a dynamic programming based algorithm was developed over 50 years ago for solving the problem by Bellman (Bellman,1957; Bellman,1960; Bellman,1962) and slightly improved later by Held and Karp[Held & Karp,1962), Karp(Karp ,1982), Bertsekas(Bertsekas,2000) , Dreyfus and Law(Dreyfus & Law,1977) . Bellman algorithm has a time complexity of $O(n^2 2^n)$ which is not polynomial (as a function of n). Table 4 lists the problem sizes that can be solved using the dynamic programming based algorithm, for different computing capacities and elapsed times, the table assumes that each function evaluation during the solution process requires the execution of 10 instructions and that 40 bit level operations have to be executed for each instruction.

Problems consisting of 50 cities can be solved routinely using today's supercomputers. It seems as if problems consisting of 100 cities will be solved routinely on a computer that will be built sometime between 50 to 60 years down the road.

When comparing table 1 and 4 it is obvious that the nonpolynomial but less than complete enumeration algorithms, have a great potential of moving some of the "unsolvable" problems to the solvable range (up to the limit of problem size data) even with the limits on future development of supercomputers. This is a line of research that should be pursued. The tables also exhibit that algorithmic developments will have more impact on the solvable range of such problems when compared to future developments in computing capacity.

Table 4: The maximal Traveling Salesman problem size solvable using a simple dynamic programming algorithm for different computing capacities and elapsed times

Number of operations per second	**Largest Problem Size Solved in a**							
	Min	**Hour**	**Day**	**Week**	**Month**	**Year**	**Decade**	**Century**
Seconds	60	3,600	86,400	604,800	2,592,000	31,104,000	311,040,000	3,110,400,000
10^{3}	6	11	15	17	19	22	25	28
10^{6}	14	19	23	26	28	31	34	37
10^{9}	23	28	32	35	37	40	43	47
10^{12}	32	37	42	44	46	50	53	56
10^{15}	41	47	51	54	56	59	62	65
10^{18}	51	56	61	63	65	69	72	75
10^{20}	57	62	67	70	72	75	78	81
10^{25}	73	78	83	86	88	91	94	98
10^{30}	89	95	99	102	104	107	110	114
10^{40}	121	127	131	134	136	140	143	146
10^{50}	154	159	164	167	169	172	176	179
10^{75}	235	241	246	249	251	254	258	261
10^{100}	318	324	328	331	333	336	340	343
10^{110}	351	356	361	364	366	369	373	376
10^{120}	384	389	394	397	399	402	406	409
10^{150}	483	488	493	496	498	501	505	508
10^{175}	565	571	576	578	580	584	587	591
10^{200}	648	654	658	661	663	667	670	673
10^{250}	813	819	824	826	829	832	835	839
10^{300}	979	985	989	992	994	998	998	999

References

Bailey H. David, (1997), "Onward to Petaflops Computing", CACM, Vol. 40, pp. 90—92.

Bellman, Richard, (1957), *Dynamic Programming*, Princeton University Press. Dover, ISBN 0486428095 .

Bellman, R., (1960) ,"Combinatorial Processes and Dynamic Programming", in Bellman, R., Hall, M., Jr. (eds.), *Combinatorial Analysis, Proceedings of Symposia in Applied Mathematics 10*,, American Mathematical Society, pp. 217–249.

Bellman, R., (1962) ,"Dynamic Programming Treatment of the Travelling Salesman Problem", *JACM*, *Vol.* 9 ,pp. 61–63.

Bertsekas, D. P., (2000), *Dynamic Programming and Optimal Control* (2nd ed.), Athena Scientific, ISBN 1-886529-09-4. (Volume 1).

Bohr, N., (1985), Rydberg's discovery of the spectral laws. In J. K. ed., *N. Bohr, Collected works*. Amsterdam: North-Holland, pp. 373-379.

Casaca Augusto, Silva Tiago, Grilo António, Nunes Mário, Presutto Franck and Isabel Rebelo, (2007) , "The use of wireless networks for the surveillance and control of cooperative vehicles in an airport," *Telecommunication Systems*, Volume 36, Nos. 1-3, pp. 141-151.

Chiasserini Carla-Fabiana and Magli Enrico, (2004), "Energy-Efficient Coding and Error Control for Wireless Video-Surveillance Networks," *Telecommunication Systems*, Volume 26, Numbers 2-4, pp. 369-387.

Dreyfus, Stuart E. and Law, Averill M., (1977), *The art and theory of dynamic programming*, Academic Press, ISBN 978-0122218606.

Gams Matjaz, (2001), "A Uniform Internet-Communicative Agent," *Electronic Commerce Research*, Volume 1, No 1-2, pp. 69-84.

Graeme Stemp-Morlock, 2009, "How to Harness Petaflop Performance," CACM, ACM News, July 29,.

Grice Don, 2009 ,"The Roadrunner Project and the Importance of Energy Efficiency on the Road to Exascale Computing," Keynote address, International Conference on Supercomputing ICS09, IBM T.J. Watson Research Center, Yorktown Heights, NY, June 8-12,.

Haviv, Y., 2006, "The road to PetaFLOP clusters," In *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, Tampa, Florida, November 11 - 17, SC '06. ACM, New York, NY.

Held, M. and Karp, R. M., (1962), "A Dynamic Programming Approach to Sequencing Problems", *Journal of the Society for Industrial and Applied Mathematics, Vol.* 10 No. 1, 196–210.

Karp, R.M., (1982), "Dynamic programming meets the principle of inclusion and exclusion," *Operations Research Letters, Vol.* **1**, pp. 49–51.

Kirk Jeremy, (2007) ,"Data Explosion Shakes Up IT", IDG News Service, September 13.

Kiyohiro Morita, Ailixier Aikebaier, Tomoya Enokido and Makoto Takizawa, (2008), "A data transmission protocol for reliable and energy-efficient data transmission in a wireless sensor-actuator network," *Telecommunication Systems*, Vol. 38, Nos 3-4, (2008) pp. 71-82.

Lyman Peter and Hal R. Varian, (2003), "*How Much Information*", www2.sims.berkeley.edu/research/projects/ how-much-info.

Moore, G. E., (1965), "Cramming more components onto integrated circuits," *Electronics Magazine*.

Planck, M., (1901), “Ueber das Gesetz der Energieverteilung im Normalspectrum,” *Ann. Phys.*, pp. 553-563.

Ritz, W., (1908), Magnetische Atomfelder und Serienspektren. *Annalen der Physik, Vierte Folge* , pp. 660-696.

Royo Fernando, Teresa Olivares and Luis Orozco-Barbosa, (2009), “A synchronous engine for wireless sensor networks,” *Telecommunication Systems*, Vol. 40, Nos 3-4, pp. 151-159.

Sanchez, S. M., (2008), “Better than a petaflop: the power of efficient experimental design,” In *Proceedings of the 40th Conference on Winter Simulation* (Miami, Florida, December 07 - 10, S. Mason, R. Hill, L. Mönch, and O. Rose, Eds. Winter Simulation Conference. Winter Simulation Conference, (2008) pp. 73-84.

Shannon K. Kuntz , Richard C. Murphy , Michael T. Niemier , Jesus Izaguirre, Peter M. Kogge, “Petaflop Computing for Protein Folding,” Proceedings of the Tenth SIAM Conference on Parallel Processing for Scientific Computing, pp. 12-14.

Yiming Ye, Jiming Liu and Alexandros Moukas, (2001), “Agents in Electronic Commerce,” *Electronic Commerce Research*, Vol. 1, No 1-2, pp. 9-14.