A

Amdahl's Law

John L. Gustafson
Intel Labs, Santa Clara, CA, USA

Synonyms

Amdahl's argument; Fixed-size speedup; Law of diminishing returns; Strong scaling

Definition

Amdahl's Law says that if you apply P processors to a task that has serial fraction f, the predicted net speedup is

$$\text{Speedup} = \frac{1}{f + \frac{1-f}{P}}.$$

More generally, it shows the speedup that results from applying any performance enhancement by a factor of P to only one part of a given workload.

A corollary of Amdahl's Law, often confused with the law itself, is that even when one applies a very large number of processors P (or other performance enhancement) to a problem, the net improvement in speed cannot exceed $1/f$.

Discussion

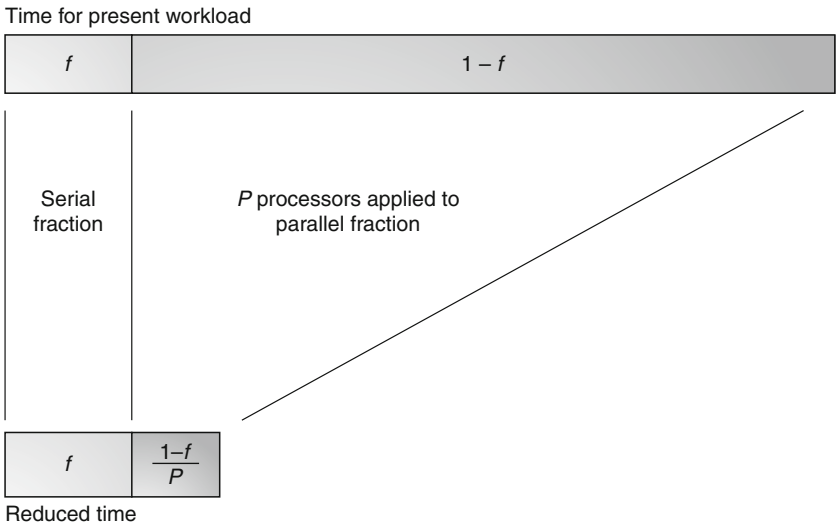
Graphical Explanation

The diagram in Fig. 1 graphically explains the formula in the definition.

The model sets the time required to solve the present workload (top bar) to unity. The part of the workload that is serial, f, is unaffected by parallelization. (See discussion below for the effect of including the time for interprocessor communication.) The model assumes that the remainder of the time, $1 - f$, parallelizes perfectly so that it takes only $1/P$ as much time as on the serial processor. The ratio of the top bar to the bottom bar is thus $1/(f + (1-f)/P)$.

History

In the late 1960s, research interest increased in the idea of achieving higher computing performance by using many computers working in parallel. At the Spring 1967 meeting of the American Federation of Information Processing Societies (AFIPS), organizers set up a session entitled "The best approach to large computing capability – A debate." Daniel Slotnick presented "Unconventional Systems," a description of a 256-processor ensemble controlled by a single instruction stream, later known as the ILLIAC IV [12]. IBM's chief architect, Gene Amdahl, presented a counterargument entitled "Validity of the single processor approach to achieving large scale computing capabilities" [1]. It



Amdahl's Law. Fig. 1 Graphical explanation of Amdahl's Law

was in this presentation that Amdahl made a specific argument about the merits of serial mainframes over parallel (and pipelined) computers.

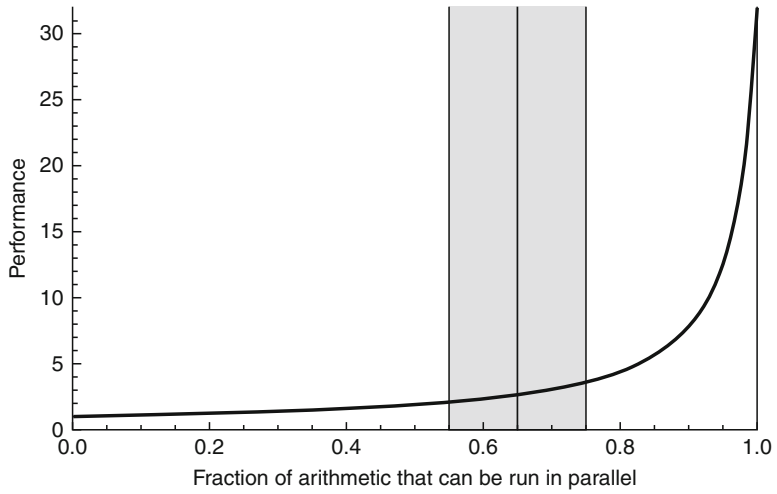
The formula known as Amdahl's Law does *not* appear anywhere in that paper. Instead, the paper shows a hand-drawn graph that includes the performance speedup of a 32-processor system over a single processor, as the fraction of parallel work increases from 0% to 100%. There are no numbers or labels on the axes in the original, but Fig. 2 reproduces his graph more precisely.

Amdahl estimated that about 10% of an algorithm was inherently serial, and data management imposed another 25% serial overhead, which he showed as the gray area in the figure centered about 65% parallel content. He asserted this was the most probable region of operation. From this, he concluded that a parallel system like the one Slotnick described would only yield from about 2X to 4X speedup. At the debate, he presented the formula used to produce the graph, but did not include it in the text of the paper itself.

This debate was so influential that in less than a year, the computing community was referring to the argument against parallel computing as "Amdahl's Law." The person who first coined the phrase may have been Willis H. Ware, who in 1972 first put into print the phrase "Amdahl's Law" and the usual form of the formula, in a RAND report titled "The Ultimate Computer" [14].

The argument rapidly became part of the commonly accepted guidelines for computer design, just as the Law of Diminishing Returns is a classic guideline in economics and business. In the early 1980s, computer users attributed the success of Cray Research vector computers over rivals such as those made by CDC to Cray's better attention to Amdahl's Law. The Cray designs did not take vector pipelining to such extreme levels relative to the rest of their system and often got a higher fraction of peak performance as a result. The widely used textbook on computer architecture by Hennessy and Patterson [7] harkens back to the traditional view of Amdahl's Law as guidance for computer designers, particularly in its earlier editions.

The formula Amdahl used was simply the use of elementary algebra to combine two different speeds, here defined as *work* per unit time, not *distance* per unit time: it simply compares two cases of the net speed as the total work divided by the total time. This common result is certainly not due to Amdahl, and he was chagrined at receiving credit for such an obvious bit of mathematics. "Amdahl's Law" really refers to the argument that the formula (along with its implicit assumptions about typical serial fractions and the way computer costs and workloads scale) predicts harsh limits on what parallel computing can achieve. For those who wished to avoid the change to their software that parallelism would require, either for economic or emotional reasons, Amdahl's Law served as a technical defense for their preference.



Amdahl's Law. Fig. 2 Amdahl's original 32-processor speedup graph (reconstructed)

Estimates of the "Serial Fraction" Prove Pessimistic

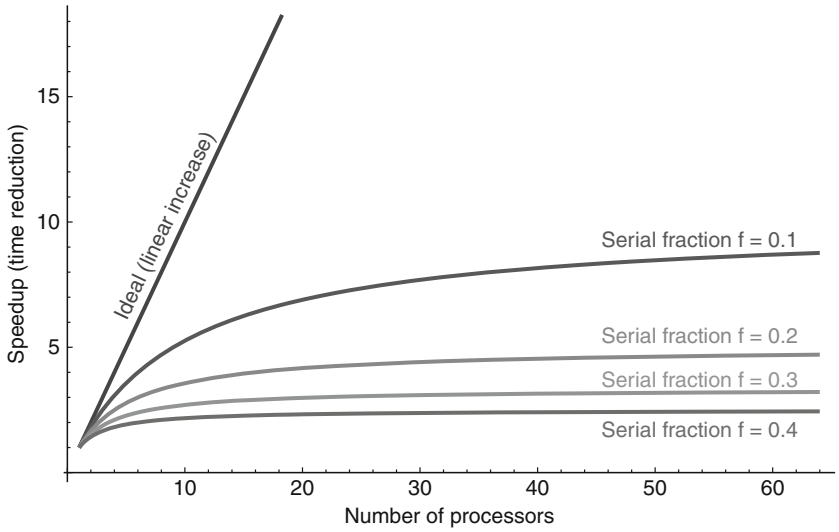
The algebra of Amdahl's Law is unassailable since it describes the fundamental way speeds add algebraically, for a fixed amount of work. However, the estimate of the serial fraction f (originally 35%, give or take 10%) was only an estimate by Amdahl and was not based on mathematics.

There exist algorithms that are inherently almost 100% serial, such as a time-stepping method for a physics simulation involving very few spatial variables. There are also algorithms that are almost 100% parallel, such as ray tracing methods for computer graphics, or computing the Mandelbrot set. It therefore seems *reasonable* that there might be a rather even distribution of serial fraction from 0 to 1 over the entire space of computer applications. The following figure shows another common way to visualize the effects of Amdahl's Law, with speedup as a function of the number of processors. Figure 3 shows performance curves for serial fractions 0.1, 0.2, 0.3, and 0.4 for a 64-processor computer system.

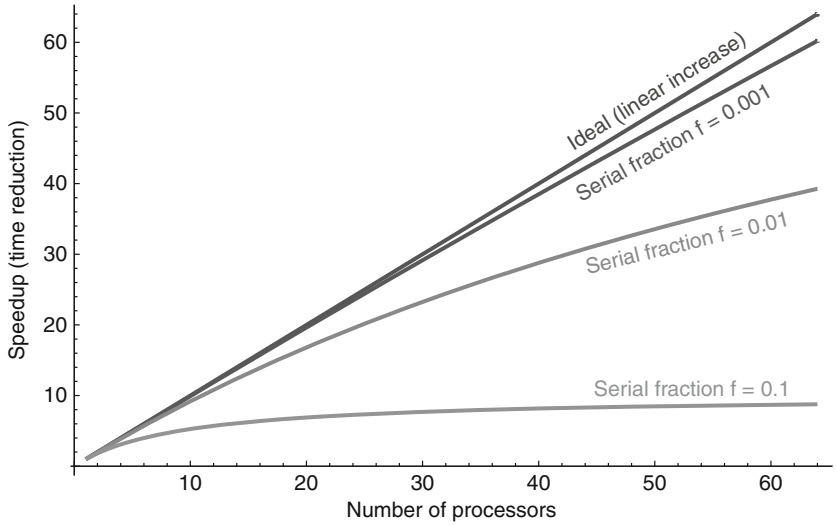
The limitations of Amdahl's Law for performance prediction were highlighted in 1985, when IBM scientist Alan Karp publicized a skeptical challenge (and a token award of $100) to anyone who could demonstrate a speedup of over 200 times on three real computer applications [9]. He had just returned from a conference at which startup companies nCUBE and Thinking Machines had announced systems with over 1,000 processors, and Karp gave the community 10 years to solve the problem putting a deadline at the end of 1995 to achieve the goal. Karp suggested fluid dynamics, structural analysis, and econometric modeling as the three application areas to draw from, to avoid the use of contrived and unrealistic applications. The published speedups of the 1985 era tended to be less than tenfold and used applications of little economic value (like how to place N queens on a chessboard so that no two can attack each other).

The Karp Challenge was widely distributed by e-mail but received no responses for years, suggesting that if 200-fold speedups were possible, they required more than a token amount of effort. C. Gordon Bell, also interested in promoting the advancement computing, proposed a similar challenge but with two alterations: He raised the award to $1,000, and said that it would be given annually to the greatest parallel speedup achieved on three real applications, but only awarded if the speedup was at least twice that of the previous award. This definition was the original Gordon Bell Prize [2], and Bell envisioned that the first award might be for something close to tenfold speedup, with increasingly difficult advances after that.

By late 1987, Sandia scientists John Gustafson, Gary Montry, and Robert Benner undertook to demonstrate high parallel speedup on applications from fluid



Amdahl's Law. Fig. 3 Speedup curves for large serial fractions



Amdahl's Law. Fig. 4 Speedup curves for smaller serial fractions

dynamics, structural mechanics, and acoustic wave propagation. They recognized that an Amdahl-type speedup (now called "strong scaling") was more challenging to achieve than some of the speedups claimed for distributed memory systems like Caltech's Cosmic Cube [5] that altered the problem according to the number of processors in use. However, using a 1,024-processor nCUBE 10, the three Sandia researchers were able to achieve performance on 1,024 processors ranging from 502X to 637X that of a single processor running the same size problem, implying the Amdahl serial fraction was only 0.0006–0.001 for those applications. This showed that the historical estimates of values for the serial fraction in Amdahl's formula might be far too high, at least for some applications. While the mathematics of Amdahl's Law is unassailable, it was a poorly substantiated opinion that the actual values of the serial fraction for computing workloads would always be too high to permit effective use of parallel computing.

Figure 4 shows Amdahl's Law, again for a 64-processor system, but with serial fractions of 0.1, 0.01, and 0.001.

The TOP500 list ranks computers by their ability to solve a dense system of linear equations. In November 2009, the top-ranked system (Jaguar, Oak Ridge National Laboratories) achieved over 75% parallel efficiency using 224,256 computing cores. For Amdahl's Law to hold, the serial fraction must be about *one part per million* for this system.

Observable Fraction and Superlinear Speedup

For many scientific codes, it is simple to instrument and measure the amount of time f spent in serial execution. One can place timers in the program around serial regions and obtain an estimate of f that might or might not strongly depend on the input data. One can then apply this fraction for Amdahl's Law estimates of time reduction, or Gustafson's Law estimates of scaled speedup. Neither law takes into account communication costs or intermediate degrees of parallelism.

A more common practice is to measure the parallel speedup as the number of processors is varied, and fit the resulting curve to derive f. This approach may yield some guidance for programmers and hardware developers, but it confuses serial fraction with communication overhead, load imbalance, changes in the relative use of the memory hierarchy, and so on. The term "strong scaling" refers to the requirement to keep the problem size the same for any number of processors. A common phenomenon that results from "strong scaling" is that when spreading a problem across more and more processors, the *memory per processor* goes down to the point where the data fits entirely in cache, resulting in *superlinear speedup* [4]. Sometimes, the superlinear speedup effects and the communication overheads partially cancel out, so what appears to be a low value of f is actually the result of the combination of the two effects. In modern parallel systems, performance

analysis with Amdahl's original law alone will usually be inaccurate, since so many other parallel processing phenomena have large effects on the speedup.

Impact on Parallel Computing

Even at the time of the 1967 AFIPS conference, there was already enough investment in serial computing software that the prospect of rewriting all of it to use parallelism was quite daunting. Amdahl's Law served as a strong defense against having to rewrite the code to exploit multiple processors. Efforts to create experimental parallel computer systems proceeded in the decades that followed, especially in academic or research laboratory settings, but the major high-performance computer companies like IBM, Digital, Cray, and HP did not create products with a high degree of parallelism, and cited Amdahl's Law as the reason. It was not until the 1988 solution to the Karp Challenge, which made clear that Amdahl's Law need not limit the utility of highly parallel computing, that vendors began developing commercial parallel products in earnest.

Implicit Assumptions, and Extensions to the Law

Fixed Problem Size

The assumption that the computing community overlooked for 20 years was that the problem size is fixed. If one applies many processors (or other performance enhancement) to a workload, it is not necessarily true that users will keep the workload fixed and accept shorter times for the execution of the task. It is common to increase the workload on the faster machine to the point where it takes the same amount of time as before.

In comparing two things, the scientific approach is to control all but one variable. The natural choice when comparing the performance of two computations is to run the same problem in both situations and look for a change in the execution time. If speed is w/t where w is work and t is time, then speedup for two situations is the ratio of the speeds: $(w_1/t_1)/(w_2/t_2)$. By keeping the work the same, that is, $w_1 = w_2$, the speedup simplifies to t_2/t_1, and this avoids the difficult problem of defining "work" for a computing task. Amdahl's Law uses this "fixed-size speedup" assumption. While the assumption is reasonable for small values of speedup, it is less reasonable when the speeds differ by many orders of magnitude. Since the execution time of an application tends to match human patience (which differs according to application), people might scale the problem such that the *time* is constant and thus is the controlled variable. That is, $t_1 = t_2$, and the speedup simplifies to w_1/w_2. See *Gustafson's Law*.

System Cost: Linear with the Number of Processors?

Another implicit assumption is that system cost is linear in the number of processors, so anything less than perfect speedup implies that cost-effectiveness goes down every time an approach uses more processors. At the time Amdahl made his argument in 1967, this was a reasonable assumption: a system with two IBM processors would probably have cost almost exactly twice that of an individual IBM processor. Amdahl's paper even states that "... by putting two processors side by side with shared memory, one would find approximately 2.2 times as much hardware," where the additional 0.2 hardware is for sharing the memory with a crossbar switch. He further estimated that memory conflicts would add so much time that net price performance of a dual-processor system would be 0.8 that of a single processor.

His cost assumptions are not valid for present-era system designs. As Moore's Law has decreased the cost of transistors to the point where a single silicon chip holds many processors in the same package that formerly held a single processor, it is apparent that system costs are far below linear in the number of processors. Processors share software and other facilities that can cost much more than individual processor cores. Thus, while Amdahl's algebraic formula is true, the implications it provided in 1967 for optimal system design have changed. For example, it might be that increasing the number of processors by a factor of 4 only provides a net speedup of 1.3X for the workload, but if the quadrupling of processors only increases system cost by 1.2X, the cost-effectiveness of the system increases with parallelism. Put another way, the point of diminishing returns for adding processors in 1967 might have been a single processor. With current economics, it might be a very large number of processors, depending on the application workload.

All-or-None Parallelism

In the Amdahl model, there are only two levels of concurrency for the use of N processors: N-fold parallel or serial. A more realistic and detailed model recognizes that the amount of exploitable parallelism might vary from one to N processors through the execution of a program [13]. The speedup is then

$$\text{Speedup}=1/\left(f_1+f_2/2+f_3/3+\ldots+f_N/\text{N}\right),$$

where f_j is the fraction of the program that can be run on j processors in parallel, and $f_1+f_2+\ldots+f_N=1$.

Sharing of Resources

Because the parallelism model was that of multiple processors controlled by a single instruction stream, Amdahl formulated his argument for the parallelism of a single job, not the parallelism of multiple users running multiple jobs. For parallel computers with multiple instruction streams, if the duration of a serial section is longer than the time it takes to swap in another job in the queue, there is no reason that $N-1$ of the N processors need to go idle as long as there are users waiting for the system. As the previous section mentions, the degree of parallelism can vary throughout a program. A sophisticated queuing system can allocate processing resources to other users accordingly, much as systems partition memory dynamically for different jobs.

Communication Cost

Because Amdahl formulated his argument in 1967, he treated the cost of *communication* of data between processors as negligible. At that time, computer arithmetic took so much longer than data motion that the data motion was overlapped or insignificant. Arithmetic speed has improved much more than the speed of interprocessor communication, so many have improved Amdahl's Law as a performance model by incorporating communication terms in the formula.

Some have suggested that communication costs are part of the serial fraction of Amdahl's Law, but this is a misconception. Interprocessor communication can be serial or parallel just as the computation can. For example, a communication algorithm may ask one processor to send data to all others in sequence (completely serial) or it may ask each processor j to send data to processor $j-1$, except that processor 1 sends to processor N, forming a communication ring (completely parallel).

Analogies

Without mentioning Amdahl's Law by name, others have referred, often humorously, to the limitations of parallel processing. Fred Brooks, in *The Mythical Man-Month* (1975), pointed out the futility of trying to complete software projects in less time by adding more people to the project [3]. "Brooks' Law" is his observation that adding engineers to a project can actually make the project take longer. Brooks quoted the well-known quip, "Nine women can't have a baby in one month," and may have been the first to apply that quip to computer technology.

From 1898 to 1906, Ambrose Bierce wrote a collection of cynical definitions called *The Devil's Dictionary*. It includes the following definition:

> Logic, *n*. The art of thinking and reasoning in strict accordance with the limitations and incapacities of human misunderstanding. The basis of logic is the syllogism, consisting of a major and a minor premise and a conclusion – thus:
> *Major Premise*: Sixty men can do a piece of work 60 times as quickly as one man.
> *Minor Premise*: One man can dig a post-hole in 60s; therefore –
> *Conclusion*: Sixty men can dig a post-hole in 1s.
> This may be called the syllogism arithmetical, in which, by combining logic and mathematics, we obtain a double certainty, and are twice blessed.

In showing the absurdity of using 60 processors (men) for an inherently serial task, he predated Amdahl by almost 70 years.

Transportation provides accessible analogies for Amdahl's Law. For example, if one takes a trip at 30 miles per hour and immediately turns around, how fast does one have to go to average 60 miles per hour? This is a trick question that many people incorrectly answer, "90 miles per hour." To average 60 miles per hour, one would have to travel back at infinite speed and *instantly*. For a fixed travel distance, just as for a fixed workload, speeds do not combine as a simple

arithmetic average. This is contrary to our intuition, which may be the reason some consider Amdahl's Law such a profound observation.

Perspective

Amdahl's 1967 argument became a justification for the avoidance of parallel computing for over 20 years. It was appropriate for many of the early parallel computer designs that shared an instruction stream or the memory fabric or other resources. By the 1980s, hardware approaches emerged that looked more like collections of autonomous computers that did not share anything yet were capable of cooperating on a single task. It was not until Gustafson published his alternative formulation for parallel speedup in 1988, along with several examples of actual 1,000-fold speedups from a 1024-processor system, that the validity of the parallel-computing approach became widely accepted outside the academic community. Amdahl's Law still is the best rule-of-thumb when the goal of the performance improvement is to reduce execution time for a fixed task, whereas Gustafson's Law is the best rule-of-thumb when the goal is to increase the problem size for a fixed amount of time. Amdahl's and Gustafson's Laws do not contradict one another, nor is either a corollary or equivalent of the other. They are for different assumptions and different situations.

Gene Amdahl, in a 2008 personal interview, stated that he never intended his argument to be applied to the case where each processor had its own operating system and data management, and would have been far more open to the idea of parallel computing as a viable approach had it been posed that way. He is now a strong advocate of parallel architectures and sits on the technical advisory board of Massively Parallel Technologies, Inc.

With the commercial introduction of single-image systems with over 100,000 processors such as Blue Gene, and clusters with similar numbers of server processor cores, it becomes increasingly unrealistic to use a fixed-size problem to compare the performance of a single processor with that of the entire system. Thus, scaled speedup (Gustafson's Law) applies to measure performance of the largest systems, with Amdahl's Law applied mainly where the number of processors changes over a narrow range.

Related Entries

►Brent's Theorem
►Gustafson's Law
►Metrics
►Pipelining

Bibliographic Notes and Further Reading

Amdahl's original 1967 paper is short and readily available online, but as stated in the Discussion section, ith has neither the formula nor any direct analysis. An objective analysis of the Law and its implications can be found in [8] or [10]. The series of editions of thehth textbook on computer architecture by Hennessey and Patterson [7] began in 1990 with a strong alignment toht Amdahl's 1967 debate position against parallel computing, and has evolved a less strident stance in more recent editions.

For a rigorous mathematical treatment of Amdahl's Law that covers many of the extensions and refinements mentioned in the Discussion section, see [13]. One of the first papers to show how fixed-sized speedup measurement is prone to superlinear speedup effects is [4].

A classic 1989 work on speedup and efficiency is "Speedup versus efficiency in parallel systems" by D. L. Eager, J. Zahorjan, and E. D. Lazowska, in *IEEE Transactions*, March 1989, 408–423. DOI=10.1109/12.21127.

Bibliography

1. Amdahl GM (1967) Validity of the single-processor approach to achieve large scale computing capabilities. AFIPS Joint Spring Conference Proceedings 30 (Atlantic City, NJ, Apr. 18–20), AFIPS Press, Reston VA, pp 483–485, At http://www-inst.eecs.berkeley.edu/~n252/paper/Amdahl.pdf
2. Bell G (interviewed) (July 1987) An interview with Gordon Bell. IEEE Software, 4(4):102–104
3. Brooks FP (1975) The mythical man-month: Essays on software engineering. Addison-Wesley, Reading. ISBN 0-201-00650-2
4. Gustafson JL (April 1990) Fixed time, tiered memory, and superlinear speedup. Proceedings of the 5th distributed memory conference, vol 2, pp 1255–1260. ISBN: 0-8186-2113-3
5. Gustafson JL, Montry GR, Benner RE (July 1988) Development of parallel methods for a 1024-processor hypercube. SIAM J Sci Statist Comput, 9(4):609–638
6. Gustafson (May 1988) Reevaluating Amdahl's law. Commun ACM, 31(5):532–533. DOI=10.1145/42411.42415

7. Hennessy JL, Patterson DA (1990, 1996, 2003, 2007) Computer architecture: A quantitative approach. Elsevier Inc.
8. Hwang K, Briggs F (1990) Computer architecture and parallel processing, McGraw-Hill, New York. ISBN: 0070315566
9. Karp A (1985) http://www.netlib.org/benchmark/karp-challenge
10. Lewis TG, El-Rewini H (1992) Introduction to parallel computing, Prentice Hall. ISBN: 0-13-498924-4, 32–33
11. Seitz CL (1986) Experiments with VLSI ensemble machines. Journal of VLSI and computer systems, vol 1. No. 3, pp 311–334
12. Slotnick D (1967) Unconventional systems. AFIPS joint spring conference proceedings 30 (Atlantic City, NJ, Apr. 18–20). AFIPS Press, Reston VA, pp 477–481
13. Sun X-H, Ni L (1993) Scalable problems and memory-bounded speedup. Journal of parallel and distributed computing, vol 19. No 1, pp 22–37
14. Ware WH (1972) The Ultimate Computer. IEEE spectrum, vol 9. No. 3, pp 84–91

AMG

►Algebraic Multigrid

Analytics, Massive-Scale

►Massive-Scale Analytics

Anomaly Detection

►Race Detection Techniques
►Intel Parallel Inspector

Anton, A Special-Purpose Molecular Simulation Machine

Ron O. Dror[1], Cliff Young[1], David E. Shaw[1,2]
[1]D. E. Shaw Research, New York, NY, USA
[2]Columbia University, New York, NY, USA

Definition

Anton is a special-purpose supercomputer architecture designed by D. E. Shaw Research to dramatically accelerate molecular dynamics (MD) simulations of biomolecular systems. Anton performs massively parallel computation on a set of identical MD-specific ASICs that interact in a tightly coupled manner using a specialized high-speed communication network. Anton enabled, for the first time, the simulation of proteins at an atomic level of detail for periods on the order of a millisecond – about two orders of magnitude beyond the previous state of the art – allowing the observation of important biochemical phenomena that were previously inaccessible to both computational and experimental study.

Discussion

Introduction

Classical molecular dynamics (MD) simulations give scientists the ability to trace the motions of biological molecules at an atomic level of detail. Although MD simulations have helped yield deep insights into the molecular mechanisms of biological processes in a way that could not have been achieved using only laboratory experiments [17, 18], such simulations have historically been limited by the speed at which they can be performed on conventional computer hardware.

A particular challenge has been the simulation of functionally important biological events that often occur on timescales ranging from tens of microseconds to a millisecond, including the "folding" of proteins into their native three-dimensional shapes, the structural changes that underlie protein function, and the interactions between two proteins or between a protein and a candidate drug molecule. Such long-timescale simulations pose a much greater challenge than simulations of larger chemical systems at more moderate timescales: the number of processors that can be used effectively in parallel scales with system size but not with simulation length, because of the sequential dependencies within a simulation.

Anton, a specialized, massively parallel supercomputer developed by D. E. Shaw Research, accelerated such calculations by several orders of magnitude compared with the previous state of the art, enabling the simulation of biological processes on timescales that might otherwise not have been accessible for many years. The first 512-node Anton machine (Fig. 1), which became operational in late 2008, completed an all-atom