Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities

Reprinted from the AFIPS Conference Proceedings, Vol. 30 (Atlantic City, N.J., Apr. 18–20), AFIPS Press, Reston, Va., 1967, pp. 483–485, when Dr. Amdahl was at International Business Machines Corporation, Sunnyvale, California

Dr. Gene M. Amdahl

This article was the first publication by Gene Amdahl on what became known as Amdahl's Law. Interestingly, it has no equations and only a single figure. For this issue of the SSCS News, Dr. Amdahl agreed to redraw the figure. In the available hard copy it was illegible. We print this historic paper to enable members to read the original source from some 40 years ago.

The Editors

For over a decade prophets have voiced the contention that the organization of a single computer has reached its limits and that truly significant advances can be made only by interconnection of a multiplicity of computers in such a manner as to permit cooperative solution. Variously the proper direction has been pointed out as general purpose computers with a generalized interconnection of memories, or as specialized computers with geometrically related memory interconnections and controlled by one or more instruction streams.

Demonstration is made of the continued validity of the single processor approach and of the weaknesses of the multiple processor approach in terms of application to real problems and their attendant irregularities.

The arguments presented are based on statistical characteristics of computation on computers over the last decade and upon the operational requirements within problems of physical interest. An additional reference will be one of the most thorough analyses of relative computer capabilities currently published- "Changes in Computer Performance," Datamation, September 1966, Professor Kenneth E. Knight, Stanford School of Business Administration.

The first characteristic of interest is the fraction of the computational load which is associated with data management housekeeping. This fraction has been very nearly constant for about ten years, and accounts for 40% of the executed instructions in production runs. In an entirely dedicated special purpose environment this might be reduced by a factor of two, but it is highly improbably that it could be reduced by a factor of three. The nature of this overhead appears to be sequential so that it is unlikely to be amenable to parallel processing techniques. Overhead alone would then place an upper limit on throughput of five to seven times the sequential processing rate, even if the housekeeping were done in a separate processor. The non-housekeeping part of the problem could exploit at most a processor of performance three to four times the performance of the housekeeping processor. A fairly obvious conclusion which can be drawn at this point is that the effort expended on achieving high parallel processing rates is wasted unless it is accompanied by achievements in sequential processing rates of very nearly the same magnitude.

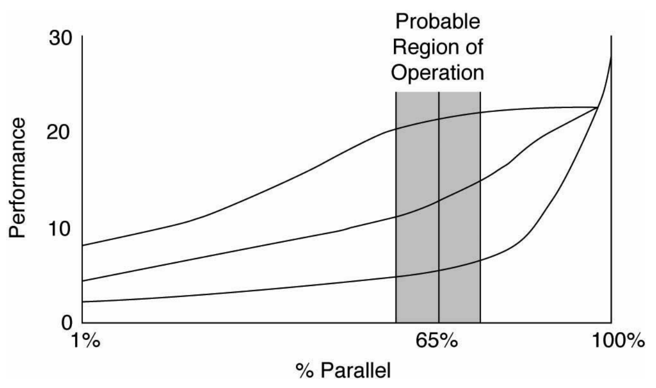
Data management housekeeping is not the only problem to plague oversimplified approaches to high speed computation. The physical problems which are of practical interest tend to have rather significant complications. Examples of these complications are as follows: Boundaries are likely to be irregular; interiors are likely to be inhomogeneous; computations required may be dependent on the states of the variables at each point; propagation rates of different physical effects may be quite different; the rate of convergence, or convergence at all, may be strongly dependent on sweeping through the array along different axes on succeeding passes, etc. The effect of each of these complications is very severe on any computer organization based on geometrically related processors in a paralleled processing system. Even the existence of regular rectangular boundaries has the interesting property that for spatial dimension of N there are 3N different point geometries to be dealt with in a nearest neighbor computation. If the second nearest neighbor were also involved, there would be 5N different point geometries to contend with. An irregular boundary compounds this problem as does an inhomogeneous interior. Computations which are dependent on the states of variables would require the processing at each point to consume approximately the same computational time as the sum of computations of all physical effects within a large region. Differences or changes in propagation rates may affect the mesh point relationships.

Ideally the computation of the action of the neighboring points upon the point under consideration involves their values at a previous time proportional to the mesh spacing and inversely proportional to the propagation rate. Since the time step is normally kept constant, a faster propagation rate for some effects would imply interactions with more distant points. Finally, the fairly common practice of sweeping through the mesh along different axes on succeeding passes poses problems of data management which affects all processors; however, it affects geometrically related processors more severely by requiring transposing all points in storage in addition to the revised input-output scheduling. A realistic assessment of the effect of these irregularities on the actual performance of a parallel processing device, compared to its performance on a simplified and regularized abstraction of the problem, yields a degradation in the vicinity of one-half to one order of magnitude.

To sum up the effects of data management housekeeping and of problem irregularities, the author has compared three different machine organizations involving approximately equal amounts of hardware. Machine A has thirty two arithmetic execution units controlled by a single instruction stream. Machine B has pipelined arithmetic execution units with up to three overlapped operations on vectors of eight elements. Machine C has the same pipelined execution units, but initiation of individual operations at the same rate as Machine B permitted vector element operations. The performance of these three machines is plotted in Figure I as a function of the fraction of the number of instructions which permit parallelism. The probable region of operation is centered around a point corresponding to 25% data management overhead and l0% of the problem operations forced to be sequential.

The historic performance versus cost of computers has been explored very thoroughly by Professor Knight. The carefully analyzed data he presents reflects not just execution times for arithmetic operations and cost of minimum of recommended configurations. He includes memory capacity effects, input-output overlap experienced, and special functional capabilities. The best statistical fit obtained corresponds to a performance proportional to the square of the cost at any technological level. This result very effectively supports the often invoked “Grosch’s Law.” Utilizing this analysis, one can argue that if twice the amount of hardware were exploited in a single system, one could expect to obtain four times the performance. The only difficulty is involved in knowing how to exploit this additional hardware. At any point in time it is difficult to foresee how the previous bottlenecks in a sequential computer will be effectively overcome. If it were easy they would not have been left as bottlenecks. It is true by historical example that the successive obstacles have been hurdled, so it is appropriate to quote the Rev. Adam Clayton Powell- "Keep the faith, baby!" If alternatively one decided to improve the performance by putting two processors side by side with shared memory, one would find approximately 2.2 times as much hardware. The additional two tenths in hardware accomplish the crossbar switching for the sharing. The resulting performance achieved would be about 1.8. The latter figure is derived from the assumption of each processor utilizing half of the memories about half of the time. The resulting memory conflicts in the shared system would extend the execution of one of two operations by one quarter of the execution time. The net result is a price performance degradation to 0.8 rather than an improvement to 2.0 for the single larger processor.

Comparative analysis with associative processors is far less easy and obvious. Under certain conditions of regular formats there is a fairly direct approach. Consider an associative processor designed for pattern recognition, in which decisions within individual elements are forwarded to some set of other elements. In the associative processor design the receiving elements would have a set of source addresses which recognize by associative techniques whether or not it was to receive the decision of the currently declaring element. To make a corresponding special purpose non-associative processor one would consider a receiving element and its source addresses as an instruction, with binary decisions maintained in registers. Considering the use of thin film memory, an associative cycle would be longer than a non-destructive read cycle. In such a technology the special purpose non-associative processor can be expected to take about one-fourth as many memory cycles as the associative version and only about one-sixth of the time. These figures were computed on the full recognition task, with somewhat differing ratios in each phase. No blanket claim is intended here, but rather that each requirement should be investigated from both approaches.



The diagram above illustrating “Amdahl’s Law” shows that a highly parallel machine has a harder time delivering a fair fraction of its peak performance due to the sequential component of the given computation and the overhead of coordination (e.g. synchronization) between the processors. Assuming a fixed sized problem, Amdahl speculated that most programs would require at least 25% of the computation to be sequential (only one instruction executing at a time), with overhead due to interprocessor coordination averaging 10%. The curves show that the more you depend on parallelism for performance, the slower the system is likely to be in the probable case, 65%. The lowest curve (A) represents the 32-wide SIMD processor, and the top curve (C) is for the modified vector processor. Scaled problems reduce the sequential component and the coordination overhead to a negligible level, making large numbers of processors very efficient in those cases. *Justin Rattner, Intel Senior Fellow, justin.rattner@intel.com, July 2007.*