Further results using the overhead model for parallel systems

by H. P. Flatt

A performance model that takes into consideration the overhead incurred in the use of a parallel system is used to show that the maximum value of the speedup achieved by the parallel system for a fixed problem may be much smaller than the number of processors required to achieve that value. It is also shown that under certain conditions, the problem size may be varied so as to achieve a speedup closely approximating the number of processors used.

Introduction

The availability of relatively inexpensive microprocessors with significant computing power has stimulated many people to search for the best way to utilize multiple processors to obtain the solution of a given problem. The hope, of course, is that the use of n processors to solve a problem will require only $1/n$ times the amount of time required on a single processor. It was recognized early that this expectation could not be realized, for there were parts of most computing problems that could be processed by only a single processor, thereby forcing many processors to remain idle during processing of these parts. This led to the development of a simple performance model (Amdahl's model or "law" [1]) that indicated that the amount of time required for solution of a problem by a computer system utilizing multiple processors was limited not only by the number of processors utilized, but also by the total amount of time required by the sequential portion of the problem—that is, that part of the problem for which only one processor could be used.

Furthermore, computations (e.g., [2–4]) showed that while the use of an increasingly larger number of processors could initially decrease the time required for solution of a problem, the use of "too many" processors would result in a larger execution time than that required by a smaller number of processors (see **Figure 1**). This led to investigations concerning the number of processors that would lead to the minimum execution time. The problem is very complex, involving, as it does, the characteristics of particular computer algorithms used to solve the problem, the programming techniques used, and the method of physically interconnecting the processors. As noted in [5], a kind of "folklore" developed concerning the proper solution of this problem. In two important special cases, an analytical solution was developed [6], while a more

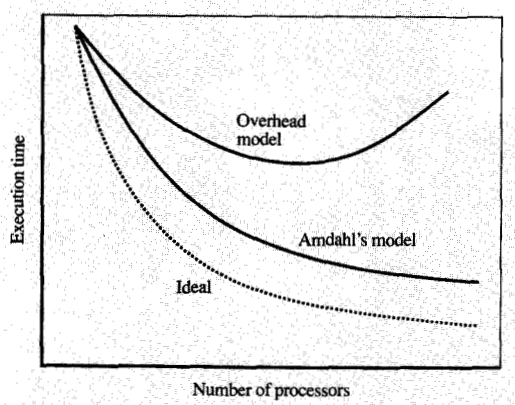

Figure 1

Three performance models of parallel processing systems.

general solution was given in [5]. It was shown in the latter reference that there was also an "optimal" number of processors that could be utilized on a problem—optimal in the sense that a point is reached beyond which a further decrease in execution time requires a disproportionately large number of processors, a concept made precise in that reference.

However, in an important paper [7], numerical results were reported that appeared to contradict Amdahl's law. It appeared that the problem solution time would be almost inversely proportional to the number of processors if the size of the parallel component of the problem were increased as the number of processors increased. A new concept of "scaling" was introduced to explain this apparent contradiction.

In the present paper, two primary results are established. The first result relates to an estimate of the minimum execution time. Ideally, one would hope that the minimum execution time would be nearly proportional to the reciprocal of the number of processors required to achieve that minimum. However, it is shown that for many cases of practical interest, this minimum execution time is closer to being proportional to the reciprocal of the "optimal" number of processors (discussed in [5])—possibly a much larger value than that suggested by the ideal case. The second result is that the extension of Amdahl's model considered in [5], the "overhead" model, is sufficient to explain the numerical results obtained in [7]. A generalization of the analytical results in that paper is also established.

Background

In this section, we summarize the notation and definitions used in [5]. That reference may be consulted for more details and additional references to the literature.

• *Notation and definitions*

If we solve a problem requiring T_{seq} units of time on a single processor of a parallel system consisting of n identical processors, then

$$T_{seq} = T_s + T_p,$$

where T_s is that part of T_{seq} which *must* be executed on a single processor and T_p that part which *could* be executed in parallel on two or more processors. *Let* $\tau_s \equiv T_s/T_{seq}$ and $\tau_p \equiv T_p/T_{seq}$ ($\tau_s + \tau_p = 1$). Term τ_s is the fraction of execution time spent in a serial mode, while τ_p is the fraction that can be spent in a parallel mode. Denote by $T(n)$ the time required to solve the problem on the parallel system and by $\tau(n)$ the normalized time [$\tau(n) \equiv T(n)/T_{seq}$].

We define the speedup by the relation

$$S(n) \equiv \frac{T_{seq}}{T(n)} = \frac{1}{\tau(n)}.$$

In [5], additional definitions of the parallel cost function $Q(n)$ and the performance-to-cost ratio $F(n)$ are given:

$$Q(n) \equiv n\tau(n) = \frac{n}{S(n)}$$

and

$$F(n) \equiv \frac{S(n)}{Q(n)},$$

The value of n for which $F(n)$ has a maximum is defined as n_F. We further denote by $E(n)$ the "efficiency" of the parallel system and define the efficiency as

$$E(n) \equiv \frac{1}{Q(n)}.$$

• *Models*

What we use to approximate $T(n)$ distinguishes various models for parallel processing. The simplest approximation to $T(n)$, ignoring the time required to execute the parallel part, is

$$T(n) = T_s.$$

Thus,

$$S(n) = \frac{T_{seq}}{T_s} = \frac{1}{\tau_s}.$$

This relationship shows that for any problem, $1/\tau_s$ is the maximum speedup for a parallel system. However, we

note that S is actually a function of the two independent variables T_s and T_{seq}. The value of T_{seq} usually depends upon one or more parameters such as the mesh size, and will, in general, vary as these parameters are changed. It is clear that the *ratio* of T_{seq} to T_s *could* increase as T_{seq} increases (or decreases), thereby increasing the value of $S(n)$ for that particular problem. Nonetheless, the value of $S(n)$ (again, for that particular problem) is bounded by $1/\tau_s$ in the general case.

The "overhead model" is a more general model. It assumes that

$$T(n) = T_s + T_o(n) + \frac{T_p}{n},$$

where $T_o(n)$ is the overhead due to synchronization cost, communication costs, etc. [5]. [A special case is Amdahl's law, for which $T_o(n) \equiv 0$.] We may also write

$$\tau(n) = \tau_s + \tau_o(n) + \frac{\tau_p}{n},$$

where $\tau_o(n) \equiv T_o(n)/T_{seq}$.

It is shown in [5] that for suitable restrictions on $\tau_o(n)$ [especially the restriction that $\tau_o'(n) > 0$ for $n > 1$], $S(n)$ takes on its maximum value for $n = n_o$, where

$$n_o^2 = \frac{\tau_p}{\tau_o'(n_o)},$$

and $\tau_o'(n)$ is the derivative of $\tau_o(n)$ with respect to n.

Bounds on the maximum speedup

In [8], a function similar to the following is considered:

$$Z(n) = E(n) + \frac{S(n)}{S(n_o)}. \tag{1}$$

$Z(n)$ is the sum of two positive, differentiable functions: the efficiency $E(n)$, which is a monotonically decreasing function of n, and the fraction of the maximum speedup that is obtained. That fraction is monotonically increasing for $1 \le n \le n_o$. We note that $Z(1)$ and $Z(n_o)$ are both greater than unity, and that $Z'(n_o) < 0$. Thus, $Z(n) > 1$ for $1 \le n \le n_o$, for otherwise $Z(n)$ would have at least one relative minimum and at least one relative maximum in this range—an impossibility. Hence the efficiency and the fraction of maximum speedup cannot be small simultaneously for $1 \le n \le n_o$. However, $Z(n)$ approaches zero for $n > n_o$ for the overhead model, in contrast to the behavior of the Amdahl model, as noted in [8].

Through the use of the function $Z(n)$, we may motivate a closer examination of the actual magnitude of $S(n_o)$. If we substitute the value $S(n_o)$ for n in Equation (1), we have

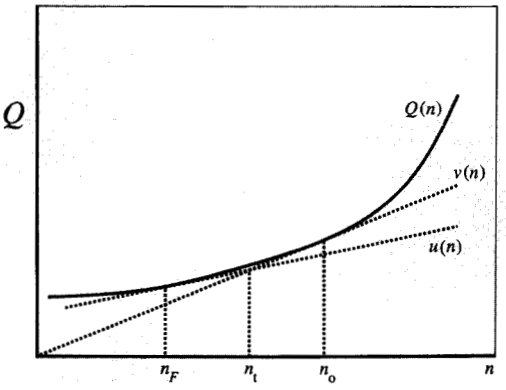

Figure 2

Tangents to the parallel cost function $Q(n)$.

$$Z[S(n_o)] = E[S(n_o)] + \frac{S[S(n_o)]}{S(n_o)}.$$

Because $E(n) = S(n)/n$, this reduces to

$$Z[S(n_o)] = 2E[S(n_o)].$$

Since $S(n_o) < n_o$, $Z[S(n_o)] > 1$. Therefore, $E[S(n_o)] > 1/2$. This result is of interest because of the observations in [5] on the value of $E(n_o)$. In particular, for a logarithmic overhead, $E(n_o)$ may be much less than 1/2. Since $E(n)$ is monotonically decreasing, there is a suggestion that at least in these cases, $S(n_o)$ may possibly be much smaller than the ideal, n_o.

As is known, $S(n_o) < n_o$; however, much tighter bounds on $S(n_o)$ may be obtained. For any overhead function satisfying the assumptions given in [5], we first show that

$$S(n_o) < n_t,$$

where n_t is the intersection of the tangent $u(n)$ to $Q(n)$ at $n = n_F$ [$F(n_F)$ is the maximum value of $F(n)$] and the tangent $v(n)$ to $Q(n)$ at $n = n_o$ (see **Figure 2**). As shown in [5], $n_F < n_o$ and $v(n) = n/S(n_o)$. Then, $n_F < n_t < n_o$, and

$$\frac{n_t}{S(n_o)} = v(n_t) = u(n_t) > Q(n_F) > 1,$$

thereby establishing the desired result. In [5], it is also shown that

$$\frac{S(n_F)}{S(n_o)} > \frac{n_o + n_F}{2n_o}.$$

From this, it follows immediately that

Table 1 Two overhead functions [linear (top line) and logarithmic (bottom line)] and associated parameters.

$\tau_o(n)$	a	b
$\alpha(n-1)$	$\alpha(3n_F - 1)$	$\alpha(2n_o - 1)$
$\alpha \log_2 n$	$\alpha(\log_2 n_F + 2\log_e 2)$	$\alpha(\log_2 n_o + \log_e 2)$

$$S(n_o) < 2S(n_F).$$

Thus, the maximum speedup is less than twice the speedup achieved for $n = n_F$.

However, geometrical considerations suggest the possibility that an even tighter bound might exist—at least in special circumstances. We show below that, in fact,

$$S(n_o) \le n_F \tag{2}$$

in many cases of interest for parallel computing. We have noted that $n_F < n_o$, but n_F is of importance because, as argued in [5], it represents the optimum number of processors to utilize in the solution of the problem—optimum in the sense that if $n < n_F$, an increase in n produces a larger fractional change in the speedup than fractional change in cost [as measured by $Q(n)$], while if $n > n_F$, an increase in n produces a smaller fractional change in the speedup than fractional change in the cost.

To establish Equation (2), we consider the defining equation for n_F:

$$n_F[\tau_s + \tau_o(n_F) + 2n_F\tau_o'(n_F)] = \tau_p, \tag{3}$$

which may be found by solving the equation $F'(n) = 0$. Let

$$a = \tau_o(n_F) + 2n_F\tau_o'(n_F)$$

and

$$b = \tau_o(n_o) + n_o\tau_o'(n_o).$$

Then, noting that $\tau_p = n_o^2\tau_o'(n_o)$, we see that

$$n_F(\tau_s + a) = n_F[\tau_s + b + (a - b)]$$

$$= n_F\left[\frac{1}{S(n_o)} + a - b\right].$$

Therefore,

$$n_F = \frac{\tau_p S(n_o)}{1 + (a - b)S(n_o)},$$

and $S(n_o) \le n_F$ if and only if

$$\tau_s b \ge a + \tau_s^2, \tag{4}$$

since $S(n_o) = 1/(\tau_s + b)$.

Consider the overhead functions shown in **Table 1**. Then, for the linear overhead function we have, from (4),

$$\tau_p(2n_o - 1) \ge 3n_F - 1 + \tau_s\left(\frac{\tau_s}{\alpha}\right), \tag{5}$$

and, from Equation (3),

$$n_F\left(\frac{\tau_s}{\alpha} + 3n_F - 1\right) = \frac{\tau_p}{\alpha}. \tag{6}$$

For the logarithmic overhead, we have that

$$\tau_p \log_2 n_o \ge \log_2 n_F + (1 + \tau_s)\log_2 e + \tau_s\left(\frac{\tau_s}{\alpha}\right) \tag{7}$$

and

$$n_F\left(\frac{\tau_s}{\alpha} + \log_2 n_F + 2\log_e 2\right) = \frac{\tau_p}{\alpha}. \tag{8}$$

We may quickly verify that both inequalities (5) and (7) are satisfied if

$$\frac{\tau_s}{\alpha} \le 10;\ \tau_s \le 0.02;\ \text{and}\ n_F \ge 4. \tag{9}$$

These inequalities may appear to be overly restrictive; indeed, they are known to be unnecessarily tight. Nonetheless, if we are to productively use parallel systems containing a large number of processors, we must have a high percentage of parallelism (note that if $\tau_s = 0.02$, we can productively use no more than 50 processors). For such large systems, only the restriction $\tau_s/\alpha \le 10$ is of any possible consequence.

It also follows immediately from Equation (3) and the definition of Q (noting that $\tau_o'(n) > 0$ by one of the basic assumptions in Reference [5]), that $Q(n_F) < 2$. Since $S(n_F)Q(n_F) = n_F$,

$$S(n_F) > \frac{n_F}{2}.$$

Therefore, if the restrictions of (9) are met,

$$\frac{n_F}{2} < S(n_o) \le n_F.$$

We note that n_F may be much less than n_o, for if the overhead is linear (as defined above), we see from Equation (3) that

$$n_o > \sqrt{2}n_F.$$

If the overhead is logarithmic (as defined above),

$$n_o > 2n_F.$$

Numerical computations show that for a logarithmic overhead, n_o may in practice be much, much greater than twice n_F.

It is clear that it is not productive to use more than n_o processors for the solution of a problem. However, the

above bounds on $S(n_0)$ and n_F suggest that unless processor cost is of no consequence, it is probably not worth the investment required to attain the maximum speedup possible. Rather, it may be more cost-effective to use a smaller number of processors, for which the efficiency remains much higher.

Largest possible problem

In the preceding section, we have considered the maximum speedup attainable for a fixed problem. We have shown that this maximum may be much smaller than might be expected, and we suggest that it may be more useful to utilize a smaller number of processors for which the efficiency remains acceptably high. But if parallel systems containing a large number of processors are to be cost-effective, the challenge is to characterize problems for which the efficiency remains large for a large number of processors.

Problems of varying size

One of the more striking results of Gustafson et al. [7] is the demonstration that very large speedups may be obtained in certain cases by increasing the problem size—i.e., changing the parameters of the problem so that T_{seq} increases. This result is in accord with the earlier results of Rosenfeld [3] and the observation of Cytron [6], and may be readily explained using either Amdahl's model or the overhead model, despite the arguments in [7].

In [7], it is assumed that the problem size is increased in such a manner that the serial part remains fixed, while the parallel part is increased in proportion to the number of processors used. In terms of our earlier definitions and notation, it is assumed that

$$T_{seq}(n) \equiv T_s(n) + T_p(n) = \hat{T}_s + n\hat{T}_p, \tag{10}$$

where $\hat{T}_s + \hat{T}_p = \hat{T}_{seq}$, the time required for the reference computation on a serial processor. For Amdahl's model,

$$S(n) = \frac{T_{seq}(n)}{T(n)} = \frac{T_{seq}(n)}{\hat{T}_s + \frac{n\hat{T}_p}{n}} = n + (1 - n)\frac{\hat{T}_s}{\hat{T}_{seq}}.$$

Thus, if the problem size is increased in accord with Equation (10), the speedup (as a function of the fractional serial part of the base computation), i.e., with n fixed, is a straight line of slope $1 - n$, as noted in [7]. For $S(n)$ to be approximately equal to n, $\hat{T}_s/\hat{T}_{seq}$ must be small.

However, it is not necessary to assume that the serial part of the problem remains fixed or that the overhead is ignored. Let

$$\varepsilon(n) \equiv \frac{n - S(n)}{n} = 1 - E(n).$$

This quantity $\varepsilon(n)$ is a measure of the relative deviation of $S(n)$ from its ideal value, n. For the overhead model,

$$Q(n) = \tau_p + n\tau_s + n\tau_o(n) = 1 + (n - 1)\tau_s + n\tau_o(n).$$

Thus,

$$(n - 1)\tau_s + n\tau_o(n) = Q(n) - 1 = \frac{n - S(n)}{S(n)}$$

or

$$\tau_s + \frac{n}{n - 1}\tau_o(n) = \frac{\varepsilon}{(1 - \varepsilon)(n - 1)}. \tag{11}$$

From (11), we see that for any value of ε $(0 < \varepsilon < 1)$, the inequalities

$$\tau_s < \frac{\varepsilon}{(1 - \varepsilon)(n - 1)} \tag{12}$$

and

$$\tau_o(n) < \frac{\varepsilon}{(1 - \varepsilon)n} \tag{13}$$

must be satisfied. If we wish $S(n)$ to remain close in value to n, i.e., for the efficiency to remain high as n increases, then as long as the inequalities (12) and (13) are satisfied for a fixed ε, this will be the case. Since $\tau_s = T_s/T_{seq}$, we see that for a fixed value of ε, T_s may increase as the problem size increases (i.e., as T_{seq} increases), but only as long as (12) is satisfied. It is not necessary that the serial part of the problem remain fixed in size, although clearly τ_s must decrease as n increases. A more restrictive condition may be imposed by the second inequality, for, as noted previously, $\tau_o'(n) > 0$; however, as τ_s, $\tau_o(n)$ is normalized, and as long as T_{seq} grows sufficiently rapidly, the second inequality may be satisfied.

A numerical illustration

In [7], for a particular problem in wave mechanics, it is shown that a speedup of 1019 was obtained on a 1024-processor system. For this problem, $\varepsilon = 0.0049$. Relations (12) and (13) require that $\tau_o(1024)$ and τ_s must each be less than (approximately) 5×10^{-6}. This compares with the estimate of 3×10^{-6} for τ_s given in [7]. This, in turn, requires that the running time of the reference problem itself must be increased in size by about 1000 times.

This particular example illustrates one possible difficulty with simply increasing the problem size. The problem size was increased by decreasing the spatial mesh size; however, the algorithms used for the solution of the problem require that for computational stability, the time step size must decrease proportionately. Thus, while the mesh size was decreased by a factor of 1024, so was the time step. The time required for the original sequential

calculation was estimated to be about five months. Although a very high speedup (over 1000) was obtained, the amount of work to be done was increased by at least an equal amount. Thus, if the original problem was impractical because of the running time on the sequential processor, it remained impractical on the parallel processor. This highlights the necessity of using numerical methods for which T_s does not grow too rapidly as T_{seq} increases, and for which $T(n)$ remains at an acceptable level.

In view of the above restriction on the possible growth of time on the parallel system, it is instructive to consider the following model comparison. Suppose we have a problem for which $\tau_s = 10^{-5}$ and $\tau_o(n) = \alpha \log_2 n$, where $\alpha = 10^{-6}$. We can show that $n_o \sim 693\,000$ and $n_F \sim 33\,000$, while $S(n_o) \sim 32\,000$ and $S(n_F) \sim 18\,000$. In today's technology, it would be impossible to employ so many processors; nevertheless, suppose we do have a system with 1024 processors. For this problem, $S(1024) = 965$ and $E(1024) = 0.94$. If we force the efficiency higher, by increasing the problem size, we could obtain an equivalent speedup using only 970 processors. This would require τ_s to decrease by a factor of 1.9, but would free a number of processors for other work. Alternatively, for the same value of ε, all 1024 processors could be employed to yield a speedup of about 1019. This would require τ_s to decrease by about a factor of 10. For suitable problems and numerical methods, it is clear that varying the problem size may lead to more efficient system utilization.

Conclusions

The overhead model is shown to be useful in helping us understand the performance restrictions that exist for the usage of parallel systems. It is shown that for some problems of interest for parallel computations, the maximum speedup that can be obtained may be much smaller than the number of processors required to obtain that maximum value. It is also shown that varying the problem size in some cases may lead to very large speedups, but that one must still be concerned with total execution time.

Both results point to the importance of the mathematical algorithm development required for the most effective use of parallel systems.

Acknowledgment

The author is indebted to A. H. Karp for preparation of the figures.

References

1. G. M. Amdahl, "Validity of the Single Processor Approach to Achieving Large Scale Computer Capabilities," *Proc. AFIPS Comp. Conf.* **30**, 483 (1967).
2. M. Lehman, "A Survey of Problems and Preliminary Results Concerning Parallel Processing and Parallel Processors," *Proc. IEEE* **54**, 1889–1901 (1966).
3. J. L. Rosenfeld, "A Case Study in Programming for Parallel-Processors," *Commun. ACM* **12**, No. 12, 645–655 (1969).
4. L. M. Adams and T. W. Crockett, "Modelling Algorithm Execution Time on Processor Arrays," *Computer* **17**, No. 7, 38–43 (1984).
5. H. P. Flatt and K. Kennedy, "Performance of Parallel Processors," *Parallel Computing* **12**, No. 1, 1–20 (1989).
6. R. Cytron, "Useful Parallelism in a Multiprocessing Environment," *Proceedings of the International Conference on Parallel Processing*, IEEE Computer Society Press, Washington, DC, 1985, pp. 450–457.
7. J. L. Gustafson, G. R. Montry, and R. E. Benner, "Development of Parallel Methods for a 1024-Processor Hypercube," *SIAM J. Sci. Stat. Comput.* **9**, No. 4, 609–638 (1988).
8. D. L. Eager, J. Zahorjan, and E. D. Lazowska, "Speedup Versus Efficiency in Parallel Systems," *IEEE Trans. Computers* **38**, No. 3, 408–423 (1989).

Received August 1, 1990; accepted for publication May 22, 1991

Horace P. Flatt *209 Brook Hollow Drive, Terrell, Texas 75160.* Dr. Flatt (now retired) was manager of the U.S. Scientific Centers. He joined IBM in Los Angeles in 1961 as a special representative working on large-scale scientific applications. In 1964 he became manager of an applications group in the Palo Alto Scientific Center, and after an assignment in Washington, DC, he returned to Palo Alto as a manager of the Center. In 1991, Dr. Flatt was appointed manager of the U.S. Scientific Centers. Prior to joining IBM, he was manager of the applied mathematics group at the Atomics International Division of North American Aviation. He received a B.A. in physics from Rice University, Houston, Texas, in 1951, and an M.A. in 1953 and a Ph.D. in 1958, both in mathematics.