Amdahl's and Gustafson-Barsis laws revisited

Andrzej Karbowski
Institute of Control and Computation Engineering,
Warsaw University of Technology,
ul. Nowowiejska 15/19, 00-665 Warsaw, Poland,
NASK (Research and Academic Computer Network),
ul. Wąwozowa 18, 02-796 Warsaw, Poland
E-mail: A.Karbowski@ia.pw.edu.pl

September 6, 2008

Abstract

The paper presents a simple derivation of the Gustafson-Barsis law from the Amdahl's law. In the computer literature these two laws describing the speedup limits of parallel applications are derived separately. It is shown, that treating the time of the execution of the sequential part of the application as a constant, in few lines the Gustafson-Barsis law can be obtained from the Amdahl's law and that the popular claim, that Gustafson-Barsis law overthrows Amdahl's law is a mistake.

Keywords: parallel computing, distributed computing, speedup

1 Introduction

The Amdahl's law formulated about four decades ago [1] is considered to be one of the most influential concepts in parallel and distributed processing [7]. It describes the limits on the speedup obtained owing to the execution of the application on the parallel machine with relation to the single-processor, sequential machine. More precisely, Amdahl's law says, that the speedup of

an application obtained owing to the execution on the parallel machine cannot be greater that the reciprocal of the sequential fraction of the program. Speedup restrictions resulting from Amdahl's law prevented designers from exploiting parallelism for many years, being a nuisance to vendors of parallel computers [4]. The rescue came from Sandia Labs. On the basis of some experiments, Gustafson [2] claimed that "the assumptions underlying Amdahl's 1967 argument are inappropriate for the current approach to massive ensemble parallelism". Furthermore, Gustafson formulated "an alternative to Amdahl's law suggested by E. Barsis at Sandia". The so-called Gustafson-Barsis law is said to vindicate the use of massively parallel processing [5], [6]. However, in the author's opinion, when we analyze deeper both laws, we will see, that Gustafson's results do not refute the Amdahl's law, and the Gustafson-Barsis law can be directly derived from the Amdahl's law.

2 Amdahl's and Gustafson-Barsis laws in the original form

Although in the original Amdahl's paper [1] there were no equations, basing on the verbal description one may present his concept formally. The way of our presentation is similar to that of [3], [4], with only one difference, which will be explained later on. It is assumed in the model, that the program consists of two parts: sequential and parallel. While the time of the execution of the sequential part for a given size n is the same on all machines, independently of the number of processors p, the parallel part is perfectly scalable, that is, the time of its execution on a machine with p processors is one p-th of the time of the execution on the machine with one processor. Let us denote by $\beta(n,p)$ the sequential fraction of the total real-time $T(n,p)$ of the execution of the program on a machine with p processors (the mentioned difference introduced here is treating both the fraction β and time T as functions of n and p; it will prove to be very useful afterwards).

With this notation we may calculate the sequential part time T_s for the given problem size n from the expression

$$T_s(n) = \beta(n,1) \cdot T(n,1) \tag{1}$$

and the parallel part time T_p, which is dependent on the problem size n and

the number of processors p, from the expression

$$T_p(n,p) = \frac{(1-\beta(n,1)) \cdot T(n,1)}{p} \tag{2}$$

If we ignore communication costs and overhead costs associated with operating system functions, such as process creation, memory management, etc. [4], the total time $T(n,p)$ will be the sum of sequential and parallel part time, that is

$$T(n,p) = T_s(n) + T_p(n,p) = \beta(n,1) \cdot T(n,1) + \frac{(1-\beta(n,1)) \cdot T(n,1)}{p} =$$

$$= \left[\beta(n,1) + \frac{1-\beta(n,1)}{p}\right] \cdot T(n,1) \tag{3}$$

From (3) we get directly the formula for the speedup $S(n,p)$ obtained due to the parallelization of the application:

$$S(n,p) = \frac{T(n,1)}{T(n,p)} = \frac{1}{\beta(n,1) + \frac{1-\beta(n,1)}{p}} \tag{4}$$

The formula (4) is called Amdahl's law. It is seen, that in the limit

$$S(n,p) \underset{p \to \infty}{\to} \frac{1}{\beta(n,1)} \tag{5}$$

It means, that even when we use infinitely many parallel processors, we cannot accelerate the calculations more than the reciprocal of the sequential fraction of the execution time of the program on a sequential machine. That is, for example, when this factor equals $\frac{1}{2}$, the program can be accelerated at most twice, when $\frac{1}{10}$ – ten times! Speedup restrictions resulting from Amdahl's law prevented designers from exploiting parallelism for many years, being a problem to vendors of parallel computers [4].

The help came from Sandia Labs. In some experiments described by Gustafson [2] it was taken, that the run time was constant, while the problem size scaled with the number of processors. More precisely, the time of the sequential part was independent, while the work to be done in parallel varied linearly with the number of processors. Since the time of the execution in Gustafson's paper [2] was normalized to 1, that is

$$T_s(n) + T_p(n,p) = 1 \tag{6}$$

we had actually the equivalence

$$\beta(n,p) \equiv T_s(n) \tag{7}$$

and

$$T_p(n,p) = 1 - \beta(n,p) \tag{8}$$

Following Gustafson, a serial processor would require time $T_s(n)+T_p(n,p)\cdot p$ to perform the task, so the scaled speedup on the parallel system was equal:

$$S(n,p) = \frac{T_s(n) + T_p(n,p)\cdot p}{T_s(n) + T_p(n,p)} = T_s(n) + T_p(n,p)\cdot p = p + (1-p)\cdot T_s(n) \tag{9}$$

Using the equivalence (7) we may write (9) in the following form:

$$S(n,p) = p + (1-p)\cdot T_s(n) = p + (1-p)\cdot \beta(n,p) = p - (p-1)\cdot \beta(n,p) \tag{10}$$

The last equation is called Gustafson-Barsis law.

3 The main results

In the Gustafson's paper [2], three things raise some doubts:

1. Mixing the problem size and the number of processors, treating both as tightly connected ("the problem size scales with the number of processors")

2. Normalizing the time of calculations on the sequential machine to 1 (eq. (6)) for all problem sizes and numbers of processors

3. Treating assessment (10) as a better alternative to Amdahl's law, derived independently, basing on different assumptions

The truth is, that Gustafson-Barsis law is nothing but a different form of Amdahl's law, and that better values of the speedup in the Gustafson's experiments with the growing size of the problem could be obtained directly from the Amdahl's law.

To show this it is sufficient to notice, that for a given problem size n there is a constant in all executions of the program, on machines with different number of processors. This constant is the time of the execution of

the sequential part $T_s(n)$ for the given problem size n. It is independent of the number of processors p, that is:

$$T_s(n) = \beta(n,p) \cdot T(n,p) = const., \quad p = 1, 2, 3, \ldots \tag{11}$$

So, it will be for any $p = 1, 2, 3, \ldots$

$$\beta(n,1) \cdot T(n,1) = \beta(n,p) \cdot T(n,p) \tag{12}$$

From the equation (12) we get:

$$\beta(n,1) = \beta(n,p) \cdot \frac{T(n,p)}{T(n,1)} \tag{13}$$

Replacing $\beta(n,1)$ in equation (3) by (13) we will get:

$$T(n,p) = \beta(n,p) \cdot T(n,p) + \frac{T(n,1)}{p} - \frac{\beta(n,p) \cdot T(n,p)}{p} \tag{14}$$

Now, multiplying both sides by p and moving all components with $T(n,p)$ to the left hand side we receive:

$$p \cdot T(n,p) - p \cdot \beta(n,p) \cdot T(n,p) + \beta(n,p) \cdot T(n,p) = T(n,1) \tag{15}$$

We will get the value of speedup $S(n,p)$ obtained owing to the parallelization dividing both sides of the equation (15) by $T(n,p)$. So, it will be equal:

$$S(n,p) = \frac{T(n,1)}{T(n,p)} = p - (p-1) \cdot \beta(n,p) \tag{16}$$

In this way we received nothing but Gustafson-Barsis law (10).

What concerns the better speedup in Gustafson's experiments with the growing size of the problem (and the number of processors which was linked there) we may explain it in the following way. Gustafson assumed, that the time spent in the serial part ("for vector startup, program loading, serial bottlenecks, I/O operations") do not depend on the problem size, that is

$$T_s(n) = const. = T_s = \beta(n,1) \cdot T(n,1) = \beta_s \cdot T(1,1), \quad \forall n \tag{17}$$

while the total time of the execution of the parallel part on the sequantial machine was proportional to the problem size n. In this way the serial factor on the sequential machine $\beta(n,1)$ was equal

$$\beta(n,1) = \frac{\beta_s \cdot T(1,1)}{\beta_s \cdot T(1,1) + n \cdot (1-\beta_s) \cdot T(1,1)} = \frac{\beta_s}{\beta_s + n \cdot (1-\beta_s)} = \frac{1}{1 + n \cdot (\frac{1}{\beta_s} - 1)} \tag{18}$$

A similar situation would be when the time $T_s(n)$ is proportional to the problem size n (e.g. $n \cdot \beta_s$), but the time spent in the parallel part is proportional to n^2 (e.g. $n^2 \cdot (1 - \beta_s)$). In such cases

$$\beta(n, 1) \underset{n \to \infty}{\to} 0 \tag{19}$$

what means, taking into account (4), that

$$S(n, p) \underset{n \to \infty}{\to} p \tag{20}$$

In other words, also from Amdahl's law we may conclude, that the bigger the size of the problem, the closer the speedup to the number of processors.

4 Conclusions

In the paper it is shown, that the Gustafson-Barsis law can be directly derived from the Amdahl's law, without strange assumptions as normalizng to one the time of execution of the program on the sequential machine. Moreover, the speedups approaching the number of processors observed in the experiments described in the Gustafson's paper can be concluded from the Amdahl's law, when we take into account as the arguments of the serial factor the size of the problem and the number of processors.

Acknowledgments

This research was supported by the Polish Ministry of Science and Higher Education under grant N N514 416934 (years 2008-2010).

References

[1] G. M. Amdahl, Validity of the Single-Processor Approach to Achieving Large Scale Computing Capabilities, In AFIPS Conference Proceedings vol. 30 (Atlantic City, N.J., Apr. 18-20), AFIPS Press, Reston, Va., pp. 483–485, April 1967.

[2] J. Gustafson, Reevaluating AmdahlŠs Law, Communications of the ACM 31(5), pp. 532–533, 1988.

[3] T. Lewis, The next $10,000_2$ years: Part I, IEEE Computer, 29(4), pp. 64 – 70, 1996.

[4] T.G. Lewis, H. El-Rewini, Introduction to Parallel Computing, Prentice-Hall, Englewood Cliffs, N.J., 1992.

[5] E.D. Reilly, Milestones in Computer Science and Information Technology, Greenwood Press, 2003.

[6] A. Ralston, E.D. Reilly, D. Hemmendinger (eds.), Encyclopedia of Computer Science, Fourth Edition, Wiley, 2003.

[7] M.D. Theys, S. Ali, H.J. Siegel, M. Chandy, K. Hwang , K. Kennedy, L. Sha, K.G. Shin, M. Snir, L. Snyder, T. Sterling, What are the top ten most influential parallel and distributed processing concepts of the past millenium?, Journal of Parallel and Distributed Computing, 61 (12), pp. 1827-1841, 2001.