Masterpraktikum Scientific Computing – High Performance Computing

Gaurav Kukreja, Evangelos Drossos

Exercise Sheet 3

Exercise 8

According to Amdahl's Law, the speedup of a parallel application with p processes in comparison to serial execution can be calculated as follows:

$$S_p = \frac{1}{s + (1-s)/p}$$

$s \in [0, 1]$ denotes the serial portion of the problem. Furthermore, the parallel efficiency of an application can be computed as follows: $Eff = {}^{S_p}/_{p}$

a) For an application with a serial portion s = 10%, the maximum number of processes p_{max} needed to achieve a parallel efficiency of at least 70% can be calculated as follows:

$$Eff \geq 0.7$$
$$\Leftrightarrow {}^{S_p}/_{p} \geq 0.7$$
$$\Leftrightarrow S_p \geq 0.7p \text{ (since p> 0)}$$
$$\Leftrightarrow \frac{1}{s + (1-s)/p} \geq 0.7p$$
$$\Leftrightarrow \frac{p}{0.1p + 0.9} \geq 0.7p$$
$$\Leftrightarrow 1 \geq 0.7(0.1p + 0.9)$$
$$\Leftrightarrow 1 - 0.7 * 0.9 \geq 0.7 * 0.1p$$
$$\Leftrightarrow p \leq 5.2857$$
$$\Rightarrow p_{max} = \lfloor 5.2857 \rfloor = 5$$

b) As was seen in exercise part a), a parallel efficiency of at least 70% can be achieved with a maximum of 5 processes. If the number of processes increases past that point, parallel efficiency starts decreasing, meaning that further parallelization yields diminishing returns. This is due to the fact that the serial part of the application plays a relatively larger role. Following observation based on Amdahl's law can be made:

$$\lim_{p\to\infty} S_p = \lim_{p\to\infty} \frac{1}{s + (1-s)/p} = \frac{1}{s}, \qquad s \in (0, 1]$$

It is therefore clear that for an arbitrarily high number of processes the speedup value converges to $1/_s$ for a problem of fixed size.

Further, if we examine parallel efficiency, then following conclusion can be drawn:

$$\lim_{p\to\infty} Eff = \lim_{p\to\infty} \frac{S_p}{p} = \lim_{p\to\infty} \frac{\frac{1}{s + (1-s)/p}}{p} = \lim_{p\to\infty} \frac{1}{sp + (1-s)} = 0, \qquad s \in (0, 1]$$

Therefore, a very high number of processes actually decrease parallel efficiency, at least as long as some portion of the problem cannot be parallelized.

As we can see, the predictions of Amdahl's Law are somewhat pessimistic.

A different approach is followed by Gustafson's Law, where the main idea is that programmers tend to adjust problem size to the available equipment. As more processors become available a larger problem can still be solved in approximately a constant amount of time (disregarding overhead). The important consideration in this case is that the parallel portion of a program scales with problem size. The non-parallelizable part, usually involving vector or matrix initializations, I/O operations etc. tends to remains fairly constant. Scaled speedup can subsequently be calculated as follows:

$$S(P) = \frac{s + bP}{s + b}$$

In this case s denotes the serial time of the program and b the parallel time per processors. Assuming that $s + b = 1$, then scaled S(P) can be written as follows:

$$S(P) = s + (1-s)P = P + (1-P)s$$

Gustafson's Law offers a different perspective, examining the scaled speedup that can be achieved when the problem size is not kept constant but rather scales with the number of available processors. In this scenario, scaled speedup grows linearly to the number of processors.