Exercise 1 - Scalability

Alessia Paoletti
Matricola SM3500374

The application we are using is a toy application: a Monte-Carlo integration of a quarter of a unit circle to compute the number PI (given as area_computed*4). We have both the basic implementation of the algorithm and the parallel MPI implementation, that computes PI by the same algorithm using MPI to compute the algorithm in parallel.

Serial program vs Parallel program

I have run the serial implementation of the program and the parallel one for the same numbers of iterations (Niter = 10^9) . For both the executions I have considered the user time, that is the time used by CPU for executing the process. As Table 1 shows the user time for the serial version of the program is 19.785 seconds, while the user time for the parallel version is 20.791 seconds. The higher time needed for the parallel execution is due to the overhead time.

Application	**User Time** $[s]$
Serial	19.785
Parallel	20.791

Table 1: User time for serial and parallel execution

Strong scalability

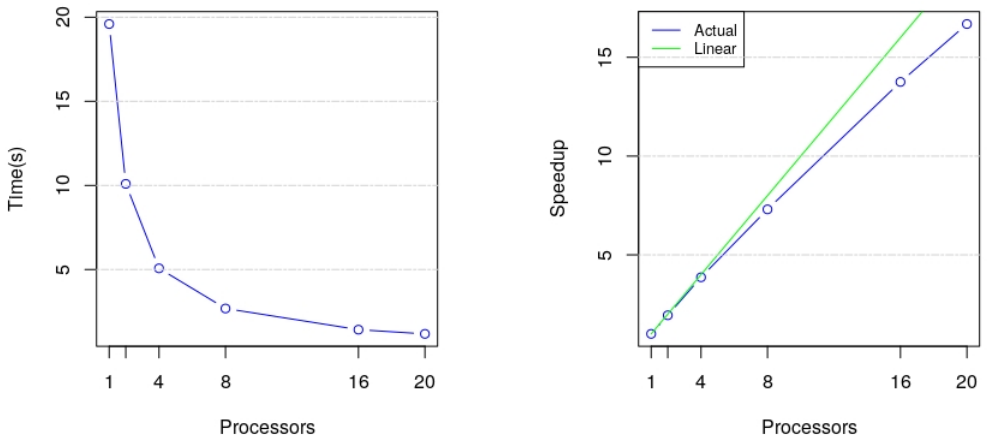
A strong scaling test means keep the problem size constant as the number of the processors increases. The goal is to compute a fixed-size problem P time faster, with P number of processors, in order to investigate how the solution time varies with the number of processors for a fixed total problem size.

I have choosen the size of the problem equal to Niter = 10^9 and I have executed the parallel version of the program with an increasing number of processors, precisly $1, 2, 4, 8, 16$ and 20. I have choosen to increase the number of the processors up to 20 and not over in order to not consider the time needed for communication among node, since 20 is the number of cores in a single Ulyesses' node.

Figure 1(a) shows, as expected, that the time of execution decreases as the number of the processors increases, due to the fact that the total job remains constant and it is divided between a growing number of prcoessors.

An important measure for a strong scalability test is the speedup, that is calculate as:

$$S(P) = \frac{Time(1)}{Time(P)} \tag{1}$$



(a) Times

(b) Speedup

Figure 1: Strong scaling test

where $Time(1)$ is the time the application takes to run only with 1 processor divided by the time it takes to run with P processors ($Time(P)$). If the $S(P) = P$ means that there is a perfect scaling, so there is a perfect speedup but, due to the Amdahl's law, it could not be achieved. The Amdahl's law states that the application will be always limited by the part of the task that cannot be paralelized. The speedup of this application is showed in Figure 1(b).

Weak scalability

A weak scaling test means increase the problem size of the problem and the number of processors in a directly proportianal way. The goal is to compute a bigger problem in the same amount of time keeping the problem size per processor fixed. Weak scaling would imply that the runtime remains constant as the problem size and the number of compute nodes increase in proportion. This is generally holds true thanks to the Gustafson's law. According to Gustafson's law the proportion of the computations that are non parallel normally decreases as the problem size increases, because, for example, program loading, serial bottlenecks and I/O that make up the serial component of the program do not grow with problem size; however, the amount of work that can be done in parallel varies linearly with the number of processors. The runtimes obtained with this application are reported in Figure 2.

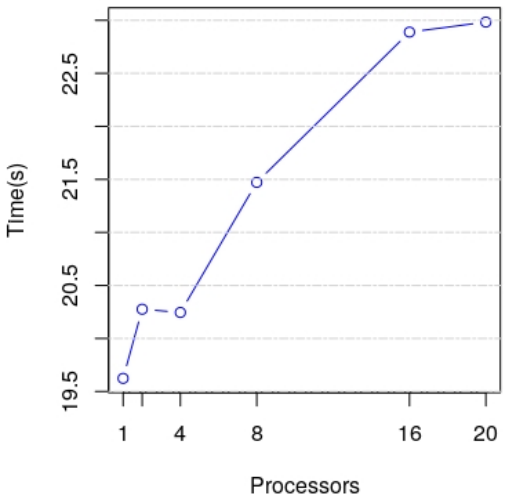


Figure 2: Time for weak scaling

Conclusions

In order to investigate if this problem is more appropriate for weak or strong scaling the efficiency can be calculated.

For the strong scaling test the efficiency is calculated like:

$$E(P) = \frac{Time(1)}{P * Time(P)} \tag{2}$$

where $Time(1)$ is the time needed to complete the work unit with 1 processors, P is the number of processors and $Time(P)$ is the time needed to complete the same work unit with P processors.

For the weak scaling test the efficiency is calculated like:

$$E(P) = \frac{Time(1)}{Time(P)} \tag{3}$$

where $Time(1)$ is the time needed to complete one work unit with one processor and $Time(P)$ is the time needed to complete P of the same work units with P processors.
Looking at the Figure 3 we can state that we achieve good results for both strong and weak scalability tests. An efficiency of around 85% is a good result, keeping in mind that a perfect efficiency cannot be achieved due to the Amdahl's law. Moreover, looking the same graph and keeping in mind that the perfect value of the efficiency is equal to

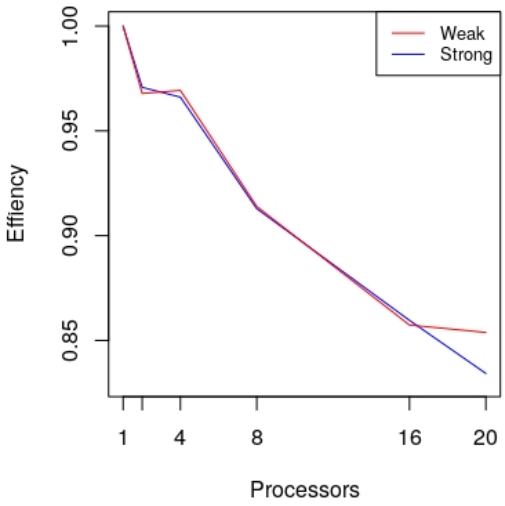


Figure 3: Comparison strong and weak efficiency

1, we can conclude that this problem is more appropiate for weak scaling due to the fact that the weak scaling efficiency decreases less than the strong scaling one.