Performance Metrics

1 Performance Metrics

1.1 Speedup

You are given the following code.

```
int simulation( int n, int k )
{
   double *M = (double *) malloc( ... );
   double res;

   initialize( M, n, k );

   res = process( M, n, k );

   post_process( res );

   cleanup( M, n, k );

   return 0;
}
```

Independently of `n` and `k`, the following relations hold.

- $T_1(\texttt{initialize}) = \frac{1}{10}\, T_1(\texttt{simulation})$
- $T_1(\texttt{process}) = \frac{15}{20}\, T_1(\texttt{simulation})$
- $T_1(\texttt{post\_process}) = \frac{1}{10}\, T_1(\texttt{simulation})$
- $T_1(\texttt{cleanup}) = \frac{1}{20}\, T_1(\texttt{simulation})$

The functions `initialize` and `cleanup` are strictly sequential.

Answer the following questions.

a) When parallelizing `simulation`, what is the maximum achievable speedup?

Solution. A 15% of the code is strictly sequential, while the remaining 85% is parallelizable. Using Amdahl's law, the best possible speedup (i.e., assuming perfect scalability and infinite processors) for this program is

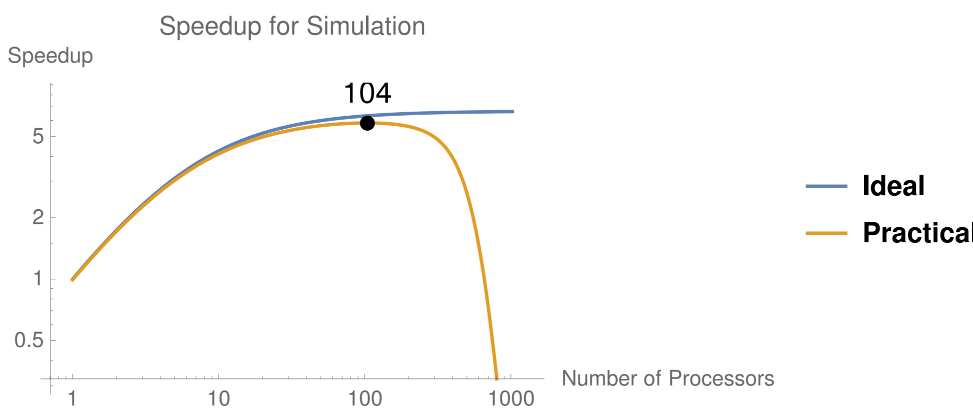
$$\lim_{p\to\infty} \frac{1}{0.15 + \frac{0.85}{p}} = \frac{20}{3}.$$

b) Use your favorite tool to plot the speedup of `simulation` vs the number of processors in the range [1, 1024].

Now assume that the speedup of the function `process` with p processors is $p \times 0.99^p$, and the speedup of `post_process` is still p.

c) Plot once again the speedup of `simulation` vs the number of processors in the same range.

Solution to b) and c).



d) Give an estimate of the optimal number of processors to maximize the speedup.

Solution. The number of processors that achieves maximum speedup is about 104.

1.2 Scalability

To solve a certain problem $\mathcal{P}$, you are given algorithms $\mathcal{A}$ and $\mathcal{B}$. Both algorithms have a quadratic complexity with respect to n, $O(n^2)$, where n is the size of the input to $\mathcal{P}$. In terms of workspace, both algorithms use $3n$ memory locations.

When solving a problem of size n = 10,000 with p processors, the following execution times (in seconds) were observed.

p	Alg. $\mathcal{A}$	Alg. $\mathcal{B}$
512	322	10.8
1024	162	10.4
2048	81.5	10.2
4096	41	10.1

Answer the questions below. In all cases, justify your answer with the necessary calculations.

Strong scalability

You have access to a computer with 32,000 processors. Assuming the parallel efficiency follows the trend observed in the table above, which algorithm do you choose to solve a problem of size 10,000? Why?

Solution. The trend indicates that for n = 10,000, Alg. $\mathcal{A}$ scales almost perfectly, while Alg. $\mathcal{B}$'s times are almost constant. Following such trends, the timing for Alg. $\mathcal{A}$ with 32k processors will be between 5 and 6 seconds, while the timing for Alg. $\mathcal{B}$ will definitely be above 9 seconds. Therefore the choice is Alg. $\mathcal{A}$.

Weak scalability

Take the configuration n = 10,000 and p = 512 as the reference load in terms of workspace per processor. You want to solve a problem of size 40,000, maintaning the ratio workspace/processor. Which algorithm do you choose? Why?

Solution. Both algorithms use a linear amount of work space, therefore in order to solve a problem 4 times as large as the reference, one has to use 4 times as many processors: 2048. The execution time for a problem of size 10,000 is known; it is also known that both algorithms have a quadratic complexity, therefore the relation $T_{2048}(40{,}000) = T_{2048}(4 \times 10{,}000) \approx 16 \times T_{2048}(10{,}000)$ holds for both of them. Hence the choice is Alg. $\mathcal{B}$.