Performance Metrics

1 Basic concepts

1. **Performance**. Suppose we have two computers A and B. Computer A has a clock cycle of 1 ns and performs on average 2 instructions per cycle. Computer B, instead, has a clock cycle of 600 ps and performs on average 1.25 instructions per cycle. Consider the program P, which requires the execution of the same number of instructions in both computers:
   - Which computer is faster for this program?
   - What if P, when executed in Computer B, performs 10% more instructions than when executed in Computer A?
2. **Speedup**. Assume the runtime of a program is 100 seconds for a problem of size 1. The program consists of an initialization phase which lasts for 10 seconds and cannot be parallelized, and a problem solving phase which can be perfectly parallelized and grows quadratically with increasing problem size.
   - What is the speedup for the program as a function of the number of processors p and the problem size n.
   - What is the execution time and speedup of the program with problem size 1, if it is parallelized and run on 4 processors?
   - What is the execution time of the program if the problem size is increased by a factor of 4 and it is run on 4 processors? And on 16 processors? What is the speedup of both measurements?
3. **Amdahl's law**. Assume an application where the execution of floating-point instructions on a certain processor P consumes 60% of the total runtime. Moreover, let's assume that 25% of the floating-point time is spent in square root calculations.
   - Based on some initial research, the design team of the next-generation processor $P2$ believes that they could either improve the performance of all floating point instructions by a factor of 1.5 or alternatively speed up the square root operation by a factor of 8. From which design alternative would the aforementioned application benefit the most?
   - Instead of waiting for the next processor generation the developers of the application decide to parallelize the code. What speedup can be achieved on a 16-CPU system, if 90% of the entire program can be perfectly parallelized? What fraction of the code has to be parallelized to get a speedup of 10?

4. **Efficiency**. Consider a computer that has a peak performance of 8 GFlops/s. An application running on this computer executes 15 TFlops, and takes 1 hour to complete.

   - How many GFlops/s did the application attain?
   - Which efficiency did it achieve?

5. **Parallel efficiency**. Given the data in Tab. 1, use your favorite plotting tool to plot

   a) The scalability of the program (speedup vs number of processors)

   b) The parallel efficiency attained (parallel efficiency vs number of processors)

   In both cases plot also the ideal case, that is, scalability equal to the number of processors and parallel efficiency equal to 1, respectively.

# Processors	Best seq.(1)	2	4	8	16
# GFlops/s	4.0	7.6	14.9	23.1	35.6

Table 1: Performance attained vs number of processors.

6. **Weak scalability**. Algorithm $\mathcal{A}$ takes an integer k as input, and uses $O(k^2)$ workspace. In terms of complexity, $\mathcal{A}$ is cubic in k. Let $T_p(k)$ be the execution time of $\mathcal{A}$ to solve a problem of size k with p processes. It is known that $T_1(8000) = 24$ minutes. Keeping the ratio workspace/processor constant, and assuming perfect scalability, what is the expected execution time for $k = 128000$?

2 Real world scenarios

1. A given program was run on a node consisting of 4 multi-core processors, each of which comprises 10 cores. The peak performance of the node is 320 GFlops/s. To study the scalability of the program, it was run using 1, 2, 4, 8, 16, 24, 32 and 40 of the cores, for fixed problem size. For this problem size, the program executes $22{\times}10^{12}$ flops. The execution time for each number of cores is collected in Tab. 2. Complete the table with the performance, speedup, efficiency, and parallel efficiency for each case.

# Cores	**Time (s)**	**Perf. (GF/s)**	**Eff.**	**Speedup**	**Par. Eff.**
1	34566.227				
2	17133.639				
4	8535.915				
8	4326.953				
16	2206.018				
24	1531.882				
32	1133.534				
40	943.301				

Table 2: Study of performance, speedup and efficiency.

- What do you think of the attained efficiency?
- And what about the scalability (speedup)?

2. A program P consists of parts A, B and C. Part A is not parallelized while parts B and C are parallelized. The program is run on a computer with 10 cores. For a given problem size, the execution of the program in 1 of the cores takes 10, 120, and 53 seconds, for parts A, B, and C, respectively. Answer the following questions:

   - Which is the minimum execution time we can attain for each part if we now execute the same program with same problem sizes on 5 of the cores? What is the best speedup we can expect for the entire program?
   - What if we run it using the 10 cores?

   In practice, the execution of the program using 5 and 10 cores to the time shown in Tab. 3. What is the speedup attained in each case for each part?

# Cores	**Part A(s)**	**Part B (s)**	**Part C(s)**
1	17	120	53
5	17	33	18
10	17	17	9

Table 3: Timings for program P using 1, 5 and 10 cores.