ME 471/571

Strong and weak scaling

HOW TO MEASURE PERFORMANCE OF A PARALLEL CODE?

T_1 - execution time of a serial algorithm

p - number of processes used

T_p - execution time of a parallel algorithm on p processes

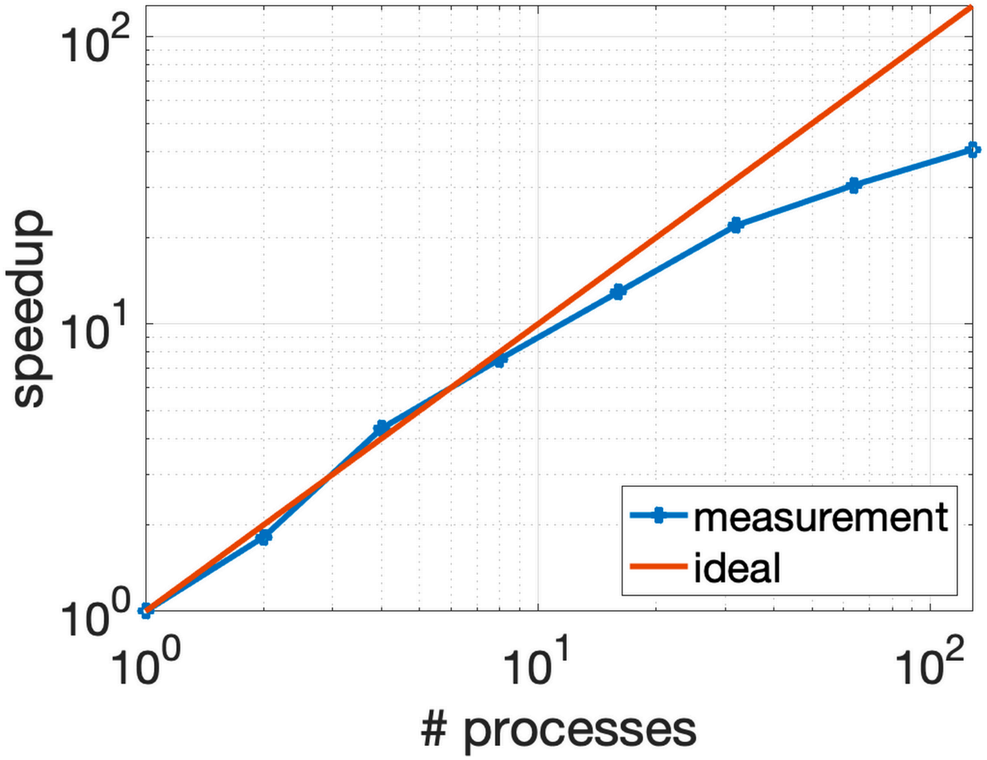
$$S_p = \frac{T_1}{T_p}$$ *- **speedup** of a parallel algorithm using p processes*

HOW TO MEASURE PERFORMANCE OF A PARALLEL CODE?

$$S_p = \frac{T_1}{T_p}$$

$$\textit{Ideally,} \quad T_p = \frac{T_1}{p}$$

which leads to a ***linear speedup****:* $S_p = p$



AMDAHL'S LAW

What if not all the code can be parallelized?

r - part of the code which remains serial

$$T_p = (1 - r)\frac{T_1}{p} + rT_1$$

In that case, speedup looks as follows:

$$S_p = \frac{T_1}{(1 - r)\frac{T_1}{p} + rT_1}$$

So even for a very large p we get:

$$\lim_{p \to \infty} S_p = \lim_{p \to \infty} \frac{T_1}{(1 - r)\frac{T_1}{p} + rT_1} = \frac{1}{r}$$

AMDAHL'S LAW

But we did not even account for communication overhead…

$$T_p = (1 - r)\frac{T_1}{p} + rT_1 + T_c$$

Speedup then becomes:

$$S_p = \frac{T_1}{(1 - r)\frac{T_1}{p} + rT_1 + T_c}$$

Assuming perfectly parallel program (r=0):

$$S_p = \frac{T_1}{\frac{T_1}{p} + T_c}$$

For scalability S_p to be close to p (linear), we require:

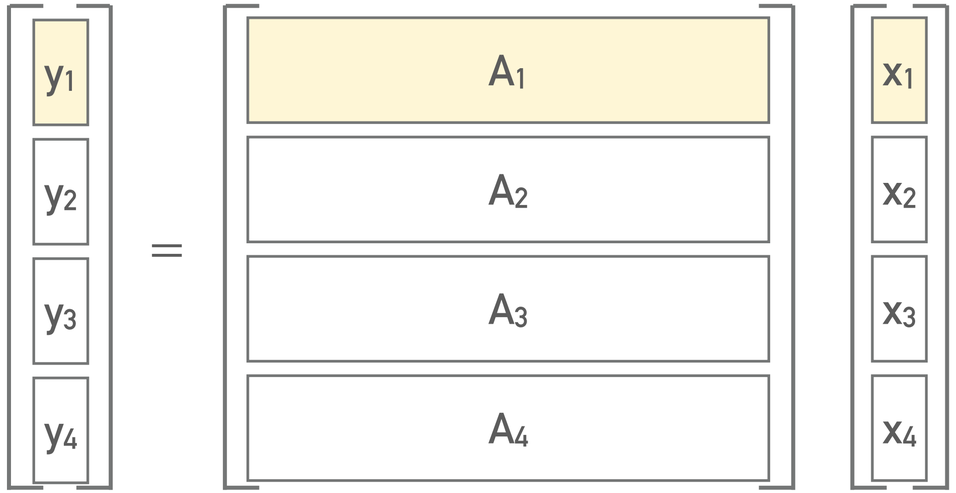
$$T_c \ll \frac{T_1}{p} \qquad or \qquad p \ll \frac{T_1}{T_c}$$

MODELING SPEEDUP

Consider a parallel matrix-vector product, where each processor holds a n/p rows of matrix A, an n/p piece of vector x, and computes an n/p piece of resulting vector y.

Parallel algorithm could look like this:

1) *communicate:* `MPI_Allgather(x)`
2) *compute:* $y_i = A_i * x$

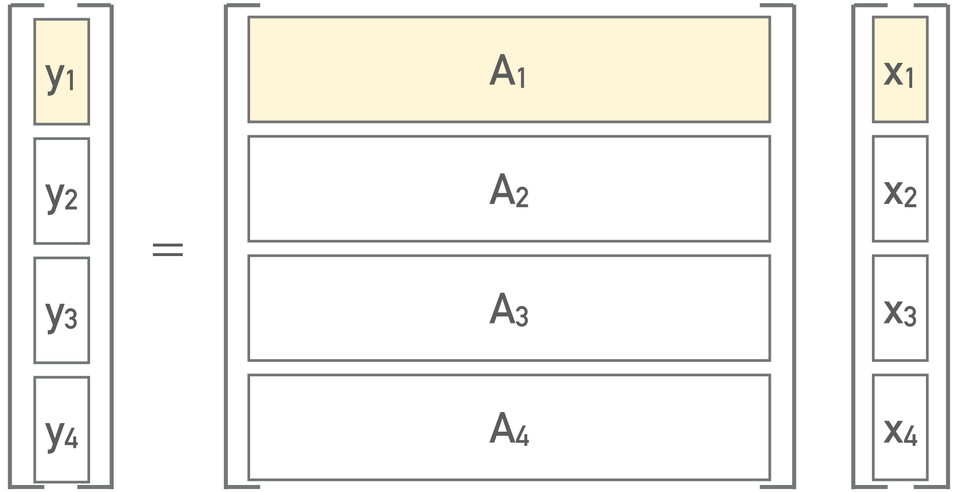


MODELING SPEEDUP

Parallel algorithm could look like this:

1) *communicate:* `MPI_Allgather(x)`

2) *compute:* $y_i = A_i * x$



Serial computation cost: $T_1 = \gamma \cdot 2n^2$

Allreduce communication cost: $T_c = \alpha \cdot \log_2(p) + \beta \cdot n$

latency *bandwidth*

Parallel computation + communication cost: $T_p = \gamma \cdot \frac{2n^2}{p} + \alpha \cdot \log_2(p) + \beta \cdot n$

Speedup: $S_p = \frac{\gamma \cdot 2n^2}{\gamma \cdot \frac{2n^2}{p} + \alpha \cdot \log_2(p) + \beta \cdot n}$

HOW TO MEASURE PERFORMANCE OF A PARALLEL CODE?

$$S_p = \frac{T_1}{T_p}$$

We can also define ***parallel efficiency***

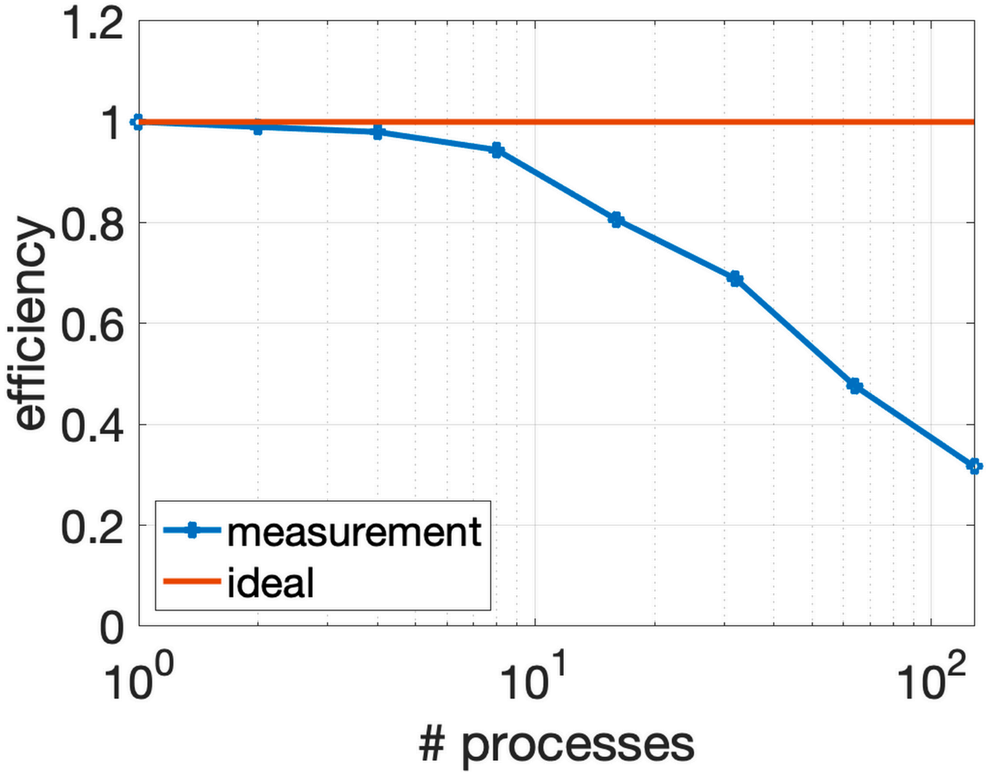
$$E_p = \frac{S_p}{p}$$

which, for a linear speedup (ideal) case leads to $E_p = 1$

regardless of the number of processes.

Speedup: $$S_p = \frac{\gamma \cdot 2n^2}{\gamma \cdot \frac{2n^2}{p} + \alpha \cdot \log_2(p) + \beta \cdot n}$$

Efficiency: $$E_P = \frac{S_p}{p} = \frac{\gamma \cdot 2n^2}{\gamma \cdot 2n^2 + p\left(\alpha \cdot \log_2(p) + \beta \cdot n\right)}$$



SCALABILITY

***Scalability** - ability to maintain efficiency with increasing process count*

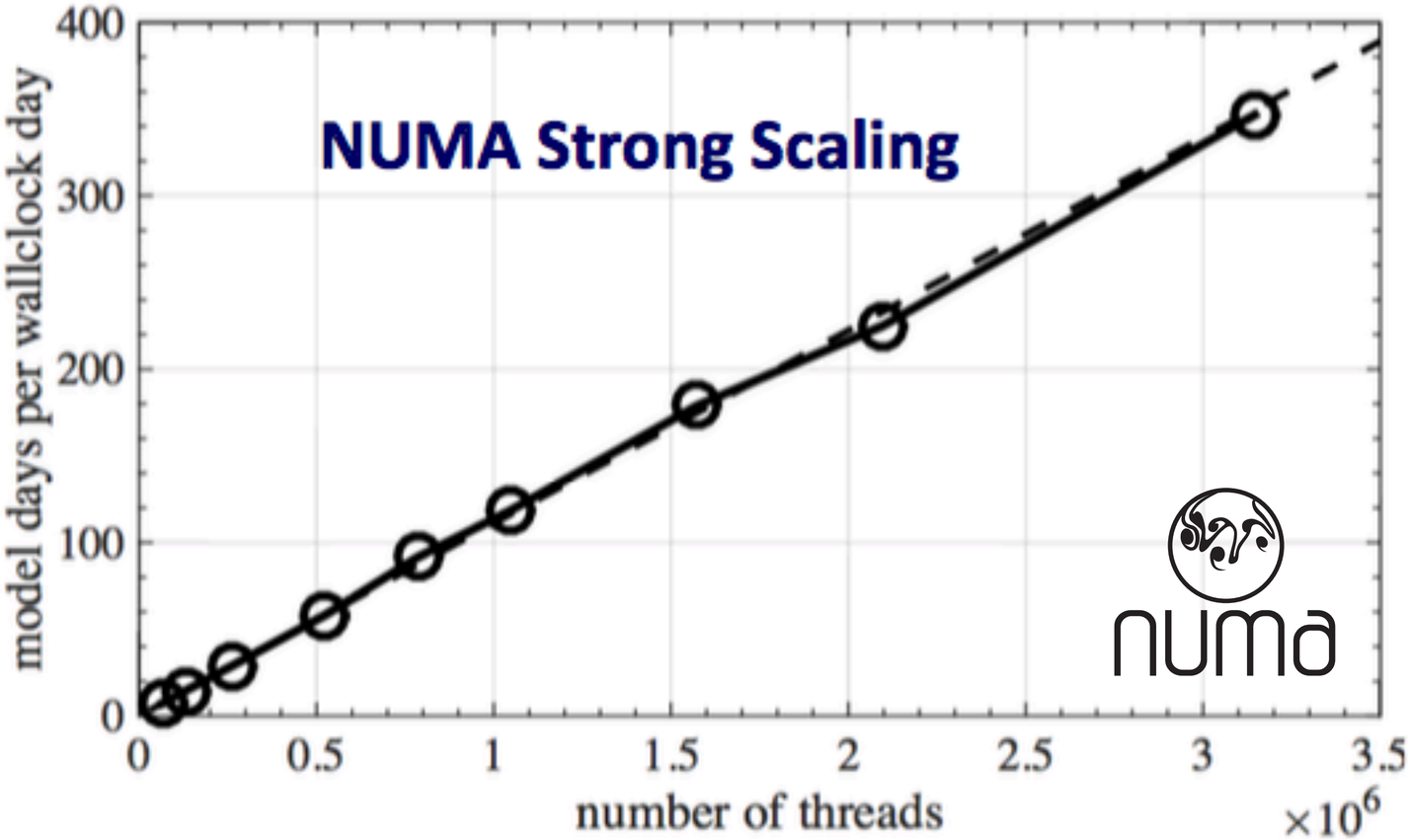
Strong scaling:

increasing number of processes does not significantly decrease efficiency for a constant problem size

Weak scaling:

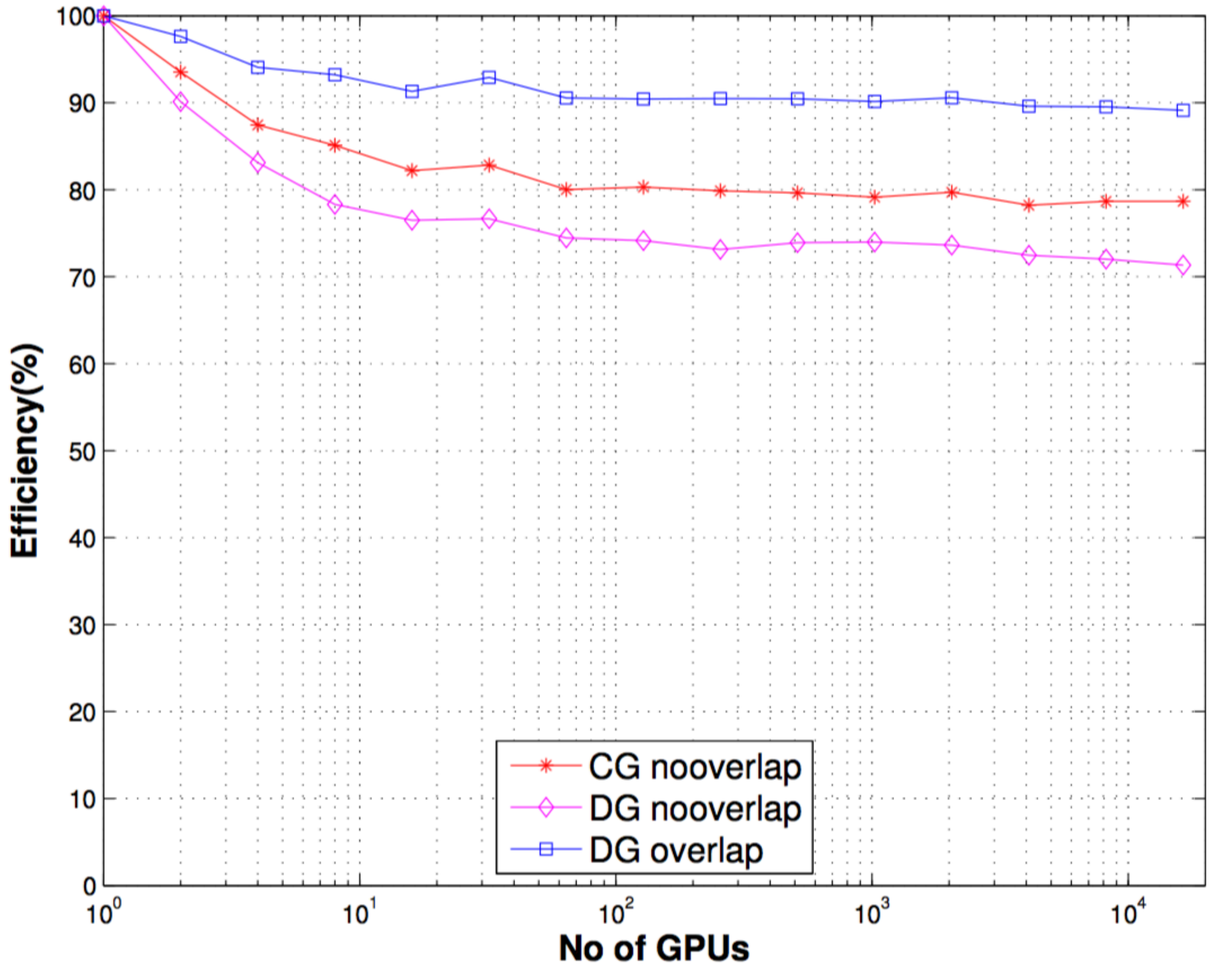
increasing number of processes does not significantly decrease efficiency when we increase problem size commensurately to the number of processes

SO, ARE WE DOOMED TO FAIL…?



Müller, A., Kopera, M. A., Marras, S., Wilcox, L. C., Isaac, T., & Giraldo, F. X. (2015). Strong scaling for numerical weather prediction at petascale with the atmospheric model NUMA. *The International Journal of High Performance Computing Applications*.

SO, ARE WE DOOMED TO FAIL…?



Abdi, D. S., Wilcox, L. C., Warburton, T. C., & Giraldo, F. X. (2019). A GPU-accelerated continuous and discontinuous Galerkin non-hydrostatic atmospheric model. *The International Journal of High Performance Computing Applications*, *33*(1), 81-109.