Reading for This Lecture

- Paper by Kumar and Gupta
- Paper by Gustafson
- Roosta, Chapter 5
Parallel Systems

- A parallel system is a parallel algorithm plus a specified parallel architecture.

- Unlike sequential algorithms, parallel algorithms cannot be analyzed very well in isolation.

- One of our primary measures of goodness of a parallel system will be its scalability.

- Scalability is the ability of a parallel system to take advantage of increased computing resources (primarily more processors).
Empirical Analysis of Parallel Algorithms

- Modern parallel computing platforms are essentially all asynchronous.
- Threads/processes do not share a global clock.
- In practice, this means that the execution of parallel algorithms is non-deterministic.
- For analysis of all but the simplest parallel algorithms, we must depend primarily on empirical analysis.
- The realities ignored by our models of parallel computation are actually important in practice.
Scalability Example

Which is better?

Algorithm A

Algorithm B

p

8

4

2

1

log₂ scale

log₂ scale
Terms and Notations

Sequential Runtime \( T_1 \)
Sequential Fraction \( s \)
Parallel Fraction \( p = 1 - s \)
Parallel Runtime \( T_N \)
Cost \( C_N = NT_N \)
Parallel Overhead \( T_o = C_N - T_1 \)
Speedup \( S_N = T_1 / T_N \)
Efficiency \( E = S_N / N \)
Definitions and Assumptions

• The sequential running time is usually taken to be the running time of the best sequential algorithm.

• The sequential fraction is the part of the algorithm that is inherently sequential (reading in the data, splitting, etc.)

• The parallel overhead includes all additional work that is done due to parallelization.
  - communication
  - nonessential work
  - idle time
Cost, Speedup, and Efficiency

- These three concepts are closely related.
- A parallel system is cost optimal if $C_N \in O(T_i)$.
- A parallel system is said to exhibit linear speedup if $S \in O(N)$.
- Hence, linear speedup $\iff$ cost optimal $\iff$ $E = 1$
- If $E > 1$, this is called super-linear speedup.
- Superlinear speedup can arise in principle, though it is not possible in principle.
Example: Parallel Semi-group

- With $n$ data elements and $p$ processors, we first combine $n/p$ elements sequentially locally.
- Then combine local results.
- Parallel running time is $n/p + 2 \log p$. 
Factors Affecting Speedup

• Sequential Fraction
• Parallel Overhead
  – Unnecessary/duplicate work
  – Communication overhead/idle time
  – Time to split/combine
• Task Granularity
• Degree of Concurrency
• Synchronization/Data Dependency
• Work Distribution
• Ramp-up and Ramp-down Time
Amdahl's Law

- Speedup is bounded by

\[ \frac{s + p}{s + p/N} = \frac{1}{s + p/N} = \frac{N}{sN + p} \]

- This means more processors ⇒ less efficient!
- How do we combat this?
- Typically, larger problem size ⇒ more efficient.
- This can be used to "overcome" Amdahl's Law.
The Isoefficiency Function

- The isoefficiency function $f(N)$ of a parallel system represents the rate at which the problem size must be increased in order to maintain a fixed efficiency.

- This function is a measure of scalability that can be analyzed using asymptotic analysis.
Gustafson's Viewpoint

- Gustafson noted that typically the serial fraction does not increase with problem size.
- This view leads to an alternative bound on speedup called **scaled speedup**.

\[
\frac{s + pN}{s + p} = s + pN = N + (1-N)s
\]

- This may be a more realistic viewpoint.