Scaling and scalable

• Scaling: running a parallel program on 1 to n processes
  • 1, 2, 3, … , n
  • 1, 2, 4, 8, …, n

• Scalable: A program is scalable if its performance improves when using more resources
Scaling and scalable

- **Scaling**: running a parallel program on 1 to n processes
  - 1, 2, 3, … , n
  - 1, 2, 4, 8, …, n
- **Scalable**: A program is scalable if its performance improves when using more resources

![Graph showing execution time (minutes) versus number of cores (Actual vs. Extrapolation)]
Weak versus strong scaling

- **Strong scaling**: *Fixed total* problem size as we run on more processes
  - Sorting n numbers on 1 process, 2 processes, 4 processes, …

- **Weak scaling**: Fixed problem size per process but *increasing total* problem size as we run on more processes
  - Sorting n numbers on 1 process
  - 2n numbers on 2 processes
  - 4n numbers on 4 processes
Amdahl’s law

- Speedup is limited by the serial portion of the code
  - Often referred to as the serial “bottleneck”
- Lets say only a fraction $f$ of the code can be parallelized on $p$ processes

\[
\text{Speedup} = \frac{1}{(1 - f) + f/p}
\]
Amdahl’s law

- Speedup is limited by the serial portion of the code
  - Often referred to as the serial “bottleneck”
- Lets say only a fraction $f$ of the code can be parallelized on $p$ processes

$$\text{Speedup} = \frac{1}{(1-f) + \frac{f}{p}}$$
Amdahl’s law

- Speedup is limited by the serial portion of the code
  - Often referred to as the serial “bottleneck”
- Lets say only a fraction \( f \) of the code can be parallelized on \( p \) processes

\[
\text{Speedup} = \frac{1}{(1 - f) + \frac{f}{p}}
\]
Performance analysis

- The process of studying the performance of parallel code
- Identify why performance might be slow
  - Serial performance
  - Serial bottlenecks when running in parallel
  - Communication overheads
Performance analysis methods

- Analytical techniques: use algebraic formulae
  - In terms of data size (n), number of processes (p)
- Time complexity analysis
- Scalability analysis (Isoefficiency)
- Model performance of various operations
  - Analytical models: LogP, alpha-beta model
Parallel prefix sum
Parallel prefix sum
Parallel prefix sum for $n >> p$

- Assign a $n/p$ block to each process
- Do calculation for the blocks on each process locally
  - Number of calculations:
- Then do parallel algorithm with partial prefix sums
  - Number of phases:
  - Total number of calculations:
Parallel prefix sum for $n \gg p$

- Assign a $n/p$ block to each process
- Do calculation for the blocks on each process locally
  - Number of calculations: $\frac{n}{p}$
- Then do parallel algorithm with partial prefix sums
  - Number of phases:
- Total number of calculations:
Parallel prefix sum for $n \gg p$

- Assign a $n/p$ block to each process
- Do calculation for the blocks on each process locally
  - Number of calculations: $\frac{n}{p}$
- Then do parallel algorithm with partial prefix sums
  - Number of phases: $\log(p)$
- Total number of calculations:
Parallel prefix sum for $ n \gg p $

- Assign a $ n/p $ block to each process
- Do calculation for the blocks on each process locally
  - Number of calculations: $ \frac{n}{p} $
- Then do parallel algorithm with partial prefix sums
  - Number of phases: $ \log(p) $
- Total number of calculations: $ \log(p) \times \frac{n}{p} $
Modeling communication: LogP model

- Model for communication on an interconnection network

L: latency or delay

O: overhead (processor busy in communication)

G: gap

P: number of processors / processes

\[ \frac{1}{g} = \text{bandwidth} \]
alpha + n * beta model

- Another model for communication

\[ T_{\text{comm}} = \alpha + n \times \beta \]

\( \alpha \): latency

\( n \): size of message

\( \frac{1}{\beta} \): bandwidth
Isoefficiency

• Relationship between problem size and number of processors to maintain a certain level of efficiency

• At what rate should we increase problem size with respect to number of processors to keep efficiency constant
Speedup and efficiency

- **Speedup**: Ratio of execution time on one process to that on \( p \) processes

  \[
  \text{Speedup} = \frac{t_1}{t_p}
  \]

- **Efficiency**: Speedup per process

  \[
  \text{Efficiency} = \frac{t_1}{t_p \times p}
  \]
Efficiency in terms of overhead

- Total time spent in all processes = (useful) computation + overhead (extra computation + communication + idle time)

\[ p \times t_p = t_1 + t_o \]

Efficiency = \[ \frac{t_1}{t_p \times p} = \frac{t_1}{t_1 + t_o} = \frac{1}{1 + \frac{t_o}{t_1}} \]
Isoefficiency function

\[
\text{Efficiency} = \frac{1}{1 + \frac{t_o}{t_1}}
\]

- Efficiency is constant if \( \frac{t_o}{t_1} \) is constant (\( K \))

\[t_o = K \times t_1\]
Isoefficiency analysis

• 1D decomposition:
  • Computation:
  • Communication:

• 2D decomposition:
  • Computation:
  • Communication
Isoefficiency analysis

\[ \sqrt{n} \]

- **1D decomposition:**
  - Computation: \( \sqrt{n} \times \frac{\sqrt{n}}{p} = \frac{n}{p} \)
  - Communication:

- **2D decomposition:**
  - Computation:
  - Communication
Isoefficiency analysis

- 1D decomposition:
  - Computation: \( \sqrt{n} \times \frac{\sqrt{n}}{p} = \frac{n}{p} \)
  - Communication: \( 2 \times \sqrt{n} \)

- 2D decomposition:
  - Computation:
  - Communication
Isoefficiency analysis

• 1D decomposition:
  • Computation: \( \sqrt{n} \times \frac{\sqrt{n}}{p} = \frac{n}{p} \)
  • Communication: \( 2 \times \sqrt{n} \)

\[
\frac{t_0}{t_1} = \frac{2 \times \sqrt{n}}{n/p} = 2 \times \frac{p}{\sqrt{n}}
\]

• 2D decomposition:
  • Computation:
  • Communication

\[
\frac{\sqrt{n}}{\sqrt{p}} = \frac{\sqrt{n}}{\sqrt{p}}
\]
Isoefficiency analysis

- **1D decomposition:**
  - Computation: \( \sqrt{n} \times \frac{\sqrt{n}}{p} = \frac{n}{p} \)
  - Communication: \( 2 \times \sqrt{n} \)
  
  \[
  \frac{t_0}{t_1} = \frac{2 \times \sqrt{n}}{\frac{n}{p}} = \frac{2 \times p}{\sqrt{n}}
  \]

- **2D decomposition:**
  - Computation: \( \frac{\sqrt{n}}{\sqrt{p}} \times \frac{\sqrt{n}}{\sqrt{p}} = \frac{n}{p} \)
  - Communication
Isoefficiency analysis

• 1D decomposition:
  • Computation: \( \sqrt{n} \times \frac{\sqrt{n}}{p} = \frac{n}{p} \)
  • Communication: \( 2 \times \sqrt{n} \)

\[
\frac{t_0}{t_1} = \frac{2 \times \sqrt{n}}{\frac{n}{p}} = \frac{2 \times p}{\sqrt{n}}
\]

• 2D decomposition:
  • Computation: \( \frac{\sqrt{n}}{\sqrt{p}} \times \frac{\sqrt{n}}{\sqrt{p}} = \frac{n}{p} \)
  • Communication:

\[
4 \times \frac{\sqrt{n}}{\sqrt{p}}
\]
Isoefficiency analysis

• 1D decomposition:
  • Computation: $\sqrt{n} \times \frac{\sqrt{n}}{p} = \frac{n}{p}$
  • Communication: $2 \times \sqrt{n}$

\[
 \frac{t_0}{t_1} = \frac{2 \times \sqrt{n}}{\sqrt{n}} = 2 \times \frac{p}{\sqrt{n}}
\]

• 2D decomposition:
  • Computation: $\frac{\sqrt{n}}{\sqrt{p}} \times \frac{\sqrt{n}}{\sqrt{p}} = \frac{n}{p}$
  • Communication: $4 \times \frac{\sqrt{n}}{\sqrt{p}}$

\[
 \frac{t_0}{t_1} = \frac{4 \times \frac{\sqrt{n}}{\sqrt{p}}}{\sqrt{n}} = 4 \times \frac{\sqrt{p}}{\sqrt{n}}
\]