Week 5

Speedup and Efficiency

Learning Objectives

- Predict performance of parallel programs
- Understand barriers to higher performance

Outline

- General speedup formula
- Amdahl's Law
- Gustafson-Barsis' Law
- Karp-Flatt metric

Speedup Formula

$$\text{Speedup} = \frac{\text{Sequential execution time}}{\text{Parallel execution time}}$$

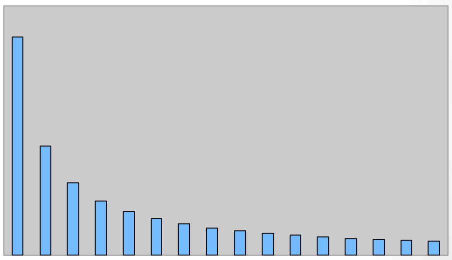
Execution Time Components

- Inherently sequential computations: $\sigma(n)$
- Potentially parallel computations: $\varphi(n)$
- Communication operations: $\kappa(n,p)$

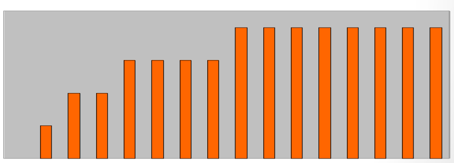
Speedup Expression

$$\psi(n,p) \le \frac{\sigma(n)+\varphi(n)}{\sigma(n)+\varphi(n)/p+\kappa(n,p)}$$

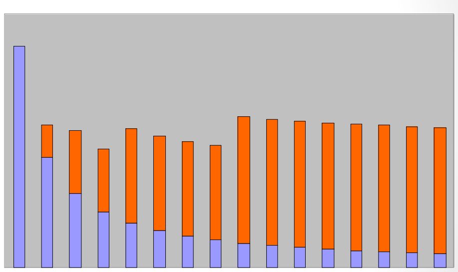
φ(n)/p



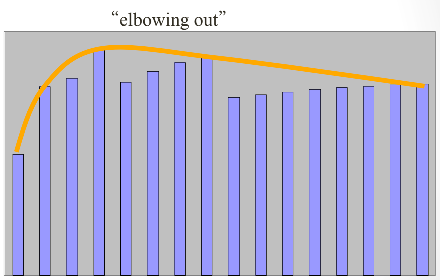
κ(n,p)



φ(n)/p + κ(n,p)



Speedup Plot



Efficiency

$$\text{Efficiency} = \frac{\text{Sequential execution time}}{\text{Processors} \times \text{Parallel execution time}}$$

$$\text{Efficiency} = \frac{\text{Speedup}}{\text{Processors}}$$

$0 \leq \varepsilon(n,p) \leq 1$

$$\varepsilon(n,p) \leq \frac{\sigma(n) + \varphi(n)}{p\sigma(n) + \varphi(n) + p\kappa(n,p)}$$

All terms $> 0 \Rightarrow \varepsilon(n,p) > 0$

Denominator $>$ numerator $\Rightarrow \varepsilon(n,p) < 1$

Amdahl's Law

$$\psi(n,p) \le \frac{\sigma(n)+\varphi(n)}{\sigma(n)+\varphi(n)/p+\kappa(n,p)}$$
$$\le \frac{\sigma(n)+\varphi(n)}{\sigma(n)+\varphi(n)/p}$$

Let $f = \sigma(n)/(\sigma(n) + \varphi(n))$, fraction that must be performed sequentially

$$\psi \le \frac{1}{f+(1-f)/p}$$

Research Computing @ CU Boulder week 5 - speedup 13 2/13/12

Example 1

- 95% of a program's execution time occurs inside a loop that can be executed in parallel. What is the maximum speedup we should expect from a parallel version of the program executing on 8 CPUs?

$$\psi \le \frac{1}{0.05+(1-0.05)/8} \cong 5.9$$

Research Computing @ CU Boulder week 5 - speedup 14 2/13/12

Example 2

- 20% of a program's execution time is spent within inherently sequential code. What is the limit to the speedup achievable by a parallel version of the program?

$$\lim_{p\to\infty} \frac{1}{0.2+(1-0.2)/p} = \frac{1}{0.2} = 5$$

Research Computing @ CU Boulder week 5 - speedup 15 2/13/12

Pop Quiz

- An oceanographer gives you a serial program and asks you how much faster it might run on 8 processors. You can only find one function amenable to a parallel solution. Benchmarking on a single processor reveals 80% of the execution time is spent inside this function. What is the best speedup a parallel version is likely to achieve on 8 processors?

Pop Quiz

- A computer animation program generates a feature movie frame-by-frame. Each frame can be generated independently and is output to its own file. If it takes 99 seconds to render a frame and 1 second to output it, how much speedup can be achieved by rendering the movie on 100 processors?

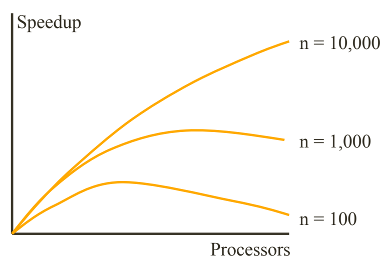
Limitations of Amdahl's Law

- Ignores $\kappa(n,p)$
- Overestimates speedup achievable

Amdahl Effect

- Typically $\kappa(n,p)$ has lower complexity than $\varphi(n)/p$
- As n increases, $\varphi(n)/p$ dominates $\kappa(n,p)$
- As n increases, speedup increases

Illustration of Amdahl Effect



Review of Amdahl's Law

- Treats problem size as a constant
- Shows how execution time decreases as number of processors increases
- Strong scaling

Another Perspective

- We often use faster computers to solve larger problem instances
- Let's treat time as a constant and allow problem size to increase with number of processors

Gustafson-Barsis's Law

$$\psi(n,p) \le \frac{\sigma(n)+\varphi(n)}{\sigma(n)+\varphi(n)/p}$$

Let $s = \sigma(n)/(\sigma(n)+\varphi(n)/p)$
S fraction of time spent in the parallel computation performing inherently sequential operations

$$\psi \le p + (1-p)s$$

Gustafson-Barsis's Law

- Begin with parallel execution time
- Estimate sequential execution time to solve same problem
- Problem size is an increasing function of *p*
- Predicts **scaled speedup**
- Weak Scaling

Example 1

- An application running on 10 processors spends 3% of its time in serial code. What is the scaled speedup of the application?

$$\psi = 10 + (1 - 10)(0.03) = 10 - 0.27 = 9.73$$

…except 9 do not have to execute serial code

Execution on 1 CPU takes 10 times as long…

Research Computing @ CU Boulder week 5 - speedup 25 2/13/12

Example 2

- What is the maximum fraction of a program's parallel execution time that can be spent in serial code if it is to achieve a scaled speedup of 7 on 8 processors?

$$7 = 8 + (1 - 8)s \Rightarrow s \approx 0.14$$

Research Computing @ CU Boulder week 5 - speedup 26 2/13/12

Pop Quiz

- A parallel program executing on 32 processors spends 5% of its time in sequential code. What is the scaled speedup of this program?

Research Computing @ CU Boulder week 5 - speedup 27 2/13/12

The Karp-Flatt Metric

- Amdahl's Law and Gustafson-Barsis' Law ignore $\kappa(n,p)$
- They can overestimate speedup or scaled speedup
- Karp and Flatt proposed another metric

Experimentally Determined Serial Fraction

$$e = \frac{\sigma(n) + \kappa(n,p)}{\sigma(n) + \varphi(n)}$$

Inherently serial component of parallel computation + processor communication and synchronization overhead

Single processor execution time

$$e = \frac{1/\psi - 1/p}{1 - 1/p}$$

Experimentally Determined Serial Fraction

- Takes into account parallel overhead
- Detects other sources of overhead or inefficiency ignored in speedup model
  - Process startup time
  - Process synchronization time
  - Imbalanced workload
  - Architectural overhead

Example 1

p	2	3	4	5	6	7	8
ψ	1.8	2.5	3.1	3.6	4.0	4.4	4.7

What is the primary reason for speedup of only 4.7 on 8 CPUs?

e	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Since *e* is constant, large serial fraction is the primary reason.

Example 2

p	2	3	4	5	6	7	8
ψ	1.9	2.6	3.2	3.7	4.1	4.5	4.7

What is the primary reason for speedup of only 4.7 on 8 CPUs?

e	0.070	0.075	0.080	0.085	0.090	0.095	0.100

Since *e* is steadily increasing, overhead is the primary reason.

Pop Quiz

p	4	8	12
ψ	3.9	6.5	?

- Is this program likely to achieve a speedup of 10 on 12 processors?

Summary (1/3)

- Performance terms
  - Speedup
  - Efficiency
- Model of speedup
  - Serial component
  - Parallel component
  - Communication component

Summary (2/3)

- What prevents linear speedup?
  - Serial operations
  - Communication operations
  - Process start-up
  - Imbalanced workloads
  - Architectural limitations

Summary (3/3)

- Analyzing parallel performance
  - Amdahl's Law
  - Gustafson-Barsis' Law
  - Karp-Flatt metric