Lecture 02—Amdahl's Law, Modern Hardware

ECE 459: Programming for Performance

Patrick Lam

University of Waterloo

January 10, 2013

Limitations

Our main focus is parallelization.

- Most programs have a sequential part and a parallel part; and,
- Amdahl's Law answers, "what are the limits to parallelization?"

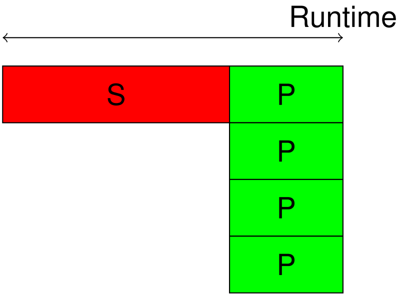
Formulation (1)

S: fraction of serial runtime in a serial execution.
P: fraction of parallel runtime in a serial execution.
Therefore, $S + P = 1$.

With 4 processors, best case, what can happen to the following runtime?

Formulation (1)

We want to split up the parallel part over 4 processors



Formulation (2)

T_s: time for the program to run in serial
N: number of processors/parallel executions
T_p: time for the program to run in parallel

- Under perfect conditions, get N speedup for P

$$T_p = T_s \cdot (S + \tfrac{P}{N})$$

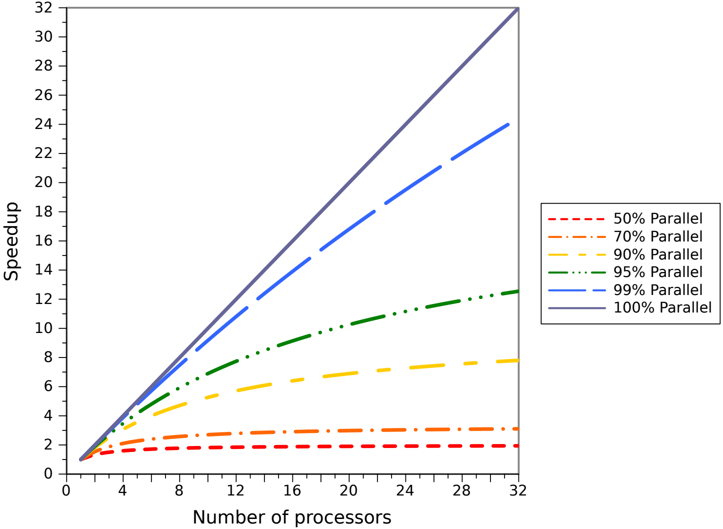
Formulation (3)

How much faster can we make the program?

$$
\begin{aligned}
\mathit{speedup} &= \frac{T_s}{T_p} \\
&= \frac{T_s}{T_S \cdot (S + \frac{P}{N})} \\
&= \frac{1}{S + \frac{P}{N}}
\end{aligned}
$$

(assuming no overhead for parallelizing; or costs near zero)

Scaling with Fraction of Parallel Code



Amdahl's Law

Replace S with $(1-P)$:

$$\textbf{\textit{speedup}} = \frac{1}{(1-P)+\frac{P}{N}}$$

$$\textit{maximum speedup} = \frac{1}{(1-P)}, \text{ since } \frac{P}{N} \to 0$$

As you might imagine, the asymptotes in the previous graph are bounded by the maximum speedup.

Amdahl's Law Generalization

The program may have many parts, each of which we can tune to a different degree.

Let's generalize Amdahl's Law.

$f_1, f_2, \ldots, f_n$: fraction of time in part n
$S_{f_1}, S_{f_n}, \ldots, S_{f_n}$: speedup for part n

$$\textbf{\textit{speedup}} = \frac{1}{\frac{f_1}{S_{f_1}} + \frac{f_2}{S_{f_2}} + \ldots + \frac{f_n}{S_{f_n}}}$$

Application (1)

Consider a program with 4 parts in the following scenario:

		Speedup	
Part	Fraction of Runtime	Option 1	Option 2
1	0.55	1	2
2	0.25	5	1
3	0.15	3	1
4	0.05	10	1

We can implement either Option 1 or Option 2.
Which option is better?

Application (2)

"Plug and chug" the numbers:

Option 1

$$speedup = \frac{1}{0.55 + \frac{0.25}{5} + \frac{0.15}{3} + \frac{0.05}{5}} = 1.53$$

Option 2

$$speedup = \frac{1}{\frac{0.55}{2} + 0.45} = 1.38$$

Empirically estimating parallel speedup P

Useful to know, don't have to commit to memory:

$$P_{\text{estimated}} = \frac{\frac{1}{speedup} - 1}{\frac{1}{N} - 1}$$

- Quick way to guess the fraction of parallel code
- Use $P_{\text{estimated}}$ to predict speedup for a different number of processors

Another Example

We run a program in serial and find it spends 12.5% of its execution on serial code and 87.5% on parallel code. How many processors do we need to get within 10% of the perfect parallel runtime?

Summary of Amdahl's Law

Important to focus on the part of the program with most impact.

Amdahl's Law:

- estimates perfect performance gains from parallelization; but,
- only applies to solving a **fixed problem size** in the shortest possible period of time

Gustafson's Law: Formulation

n: problem size
$S(n)$: fraction of serial runtime for a parallel execution
$P(n)$: fraction of parallel runtime for a parallel execution

$$\begin{aligned} T_p &= S(n) + P(n) = 1 \\ T_s &= S(n) + N \cdot P(n) \end{aligned}$$

$$\mathit{speedup} = \frac{T_s}{T_p}$$

Gustafson's Law

$$speedup = S(n) + N \cdot P(n)$$

Assuming the fraction of runtime in serial part decreases as *n* increases, the speedup approaches *N*.

- Yes! Large problems can be efficiently parallelized. (Ask Google.)

Driving Metaphor

Amdahl's Law
Suppose you're travelling between 2 cities 90 km apart. If you travel for an hour at a constant speed less than 90 km/h, your average will never equal 90 km/h, even if you energize at your destination.

Gustafson's Law
Suppose you've been travelling at a constant speed less than 90 km/h. Given enough distance, you can bring your average up to 90 km/h.