Multiprocessor Performance

Using N processors may not provide a speedup of N
- Amdahl's law still applies
- Code must contain independent parts
- Each independent part can run on a different processor
- Sequential part does not benefit from extra processors

The size of the problem or task makes a difference
- Best to use more processors on larger problems
- Processors will be idle unless they have work to do

Amdahl's Law

- Sequential part can limit speedup
- Example: 100 processors, 90× speedup?

$$T_{\text{old}} = T_{\text{sequential}} + T_{\text{parallelizable}}$$

$$T_{\text{new}} = T_{\text{sequential}} + \frac{T_{\text{parallelizable}}}{100}$$

$$\text{Speedup} = \frac{T_{\text{old}}}{T_{\text{new}}} = \frac{1}{\frac{T_{new}}{T_{\text{old}}}} = \frac{1}{\frac{T_{\text{sequential}}}{T_{\text{old}}} + \frac{T_{\text{parallelizable}}}{T_{\text{old}} * 100}}$$

Amdahl's Law

$$\text{Speedup} = \frac{1}{f_{\text{sequential}} + \dfrac{f_{\text{parallelizable}}}{100}}$$

$$\text{Speedup} = \frac{1}{(1 - f_{\text{parallelizable}}) + f_{\text{parallelizable}}/100}$$

$$\text{Speedup} = \frac{1}{(1 - f_{\text{parallelizable}}) + f_{\text{parallelizable}}/100} = 90$$

$$\text{Speedup} = \frac{1}{1 - 0.99 * f_{parallelizable}} = 90$$

$$f_{parallelizable} = \frac{1 - \dfrac{1}{90}}{0.99} = 0.999$$

So sequential part can only be 0.1% of the total.

Another example

An SMP system contains 8 processors.

A program consists of a startup sequential section that produces results used in remaining parallel part

Desired speedup = 8/3
relative to executing the program on a single processor

Parallel part must be what percent of the total code?

Example

Let f = fraction of the code corresponding to parallel part

Based on definition of speedup:

$$\frac{1}{(1-f)+\frac{f}{8}}=\frac{8}{3} \longrightarrow 1-\frac{7f}{8}=\frac{3}{8}$$

$$\frac{7f}{8}=\frac{5}{8} \longrightarrow f=\frac{5}{7}=0.7143$$

71.43% of the code must be parallelizable

Strong versus Weak Scaling

Strong Scaling:

using more processors on a given size problem

Weak Scaling:

Increasing the number of processors with problem size

Good speedup is more difficult with strong scaling

Extra processors may sit idle unless problem size grows

Strong versus Weak Scaling

Example:

A program computes the sum of two 10element vectors and the sum of two 10x10 matrices

- Each addition takes 1 cycle
- 10 processors are available
- The vector sum is computed by 1 processor
- The matrix sum is split among 10 processors

The potential speedup is a factor of 10

Total time using 1 processor = 10 + 100 = 110 cycles

Time using 10 processors = 10 + 100/10 = 20 cycles

Speedup = 110/20 = 5.5

Achieves 55% of the potential speedup

Strong versus Weak Scaling

Example:

Suppose 40 processors are used instead

The potential speedup is a factor of 40

Total time using 1 processor = 10 + 100 = 110 cycles

Time using 40 processors = 10 + 100/40 = 12.5 cycles

Speedup = 110/12.5 = 8.8

Achieves only 22% of the potential speedup

Strong versus Weak Scaling

Example:

Suppose matrix size grows to 20x20

Total cycles using 1 processor = 10 + 400 = 410

cycles using 10 processors = 10 + 400/10 = 50

cycles using 40 processors = 10 + 400/40 = 20

Speedup with 10 processors = 410/50 = 8.2

Achieves 82% of the potential speedup

Speedup with 40 processors = 410/20 = 20.5

Achieves 51.25% of the potential speedup

The size of the problem affects the speedup

Strong versus Weak Scaling

Amdahl's Law for multi-processors

Let T1 be the execution time for a program on a single processor

f = fraction of time due to the parallel part split N ways

TN is the execution time using N processors

$$TN = \left[(1-f) + \frac{f}{N}\right]\mathrm{T1}$$

Adding extra cores only improves the parallel part

$$Speedup = \frac{T1}{TN} = \frac{1}{(1-f) + \frac{f}{N}} < \frac{1}{(1-f)}$$

Indicates an upper limit on the speedup (strong scaling)

Strong versus Weak Scaling

Gustafson's Law applies when N increases with problem size

- More processors can be used with larger workloads
- TN is the execution time using N processors
- f = fraction of TN used for parallel part on N-processor system

Gustafson's law states:

$$Speedup = (1 - f) + f * N$$

Speedup varies linearly with f (weak scaling)

As workload expands, parallel part becomes a larger fraction of TN