CS 6230: High Performance Computing and Parallelization

Professor Mike Kirby
School of Computing and Scientific Computing and Imaging Institute
University of Utah

SCI INSTITUTE

THE UNIVERSITY OF UTAH

Performance Metrics

"amount of work" - percentage serial and percentage parallel

$$s + p = 1$$

Serial runtime

$$T_f^s = s + p$$

Parallel Runtime

$$T_f^p = s + \frac{p}{N}$$ Strong Scaling

Performance Metrics

Increase work as number of processors grow

$$s + pN^{\alpha}$$

Serial runtime

$$T_v^s = s + pN^{\alpha}$$

Parallel runtime

$$T_v^p = s + pN^{\alpha-1}$$

Weak Scaling

SCI INSTITUTE

THE UNIVERSITY OF UTAH

Application Speedup

Application Speedup can be defined as the quotient of parallel and serial performance for fixed problem size.

Define “performance” as “work over time”

$$P_f^s = \frac{s+p}{T_f^s} = 1 \qquad \text{serial}$$

$$P_f^p = \frac{s+p}{T_f^p(N)} = \frac{1}{s + \frac{1-s}{N}} \qquad \text{parallel}$$

Application Speedup

Application speedup ("scalability") is

$$S_f = \frac{P_t^p}{P_t^s} = \frac{1}{s + \frac{1-s}{N}}$$ "Amdahl's Law"

$$N \to \infty \text{ to } 1/s$$

This well-known function answers the question: "How much faster (in terms of runtime) does my application run when I put the same problem on N CPUs?"

SCI INSTITUTE

THE UNIVERSITY OF UTAH

Application Speedup

Define “work” as only the parallelizable part of the computation:

$$P_t^{sp} = \frac{p}{T_f^s} = p$$

Parallel performance is:

$$P_f^{pp} = \frac{p}{T_f^p(N)} = \frac{1-s}{s + \frac{1-s}{N}}$$

Application speedup is:

$$S_t^p = \frac{P_f^{pp}}{P_f^{sp}} = \frac{1}{s + \frac{1-s}{N}}$$

SCI INSTITUTE

THE UNIVERSITY OF UTAH

Parallel Performance

Weak scaling - where workload grows with CPU count. The question to ask is “How much more work can my program do in a given amount of time when I put a larger problem on N CPUs?

Serial performance

$$P_v^s = \frac{s+p}{T_f^s} = 1$$

Parallel performance (work over time) is:

$$P_v^p = \frac{s+pN^\alpha}{T_v^p(N)} = \frac{s+(1-s)N^\alpha}{s+(1-s)N^{\alpha-1}} = S_v$$

SCI INSTITUTE

THE UNIVERSITY OF UTAH

Gustafson's Law

For large CPU counts, we obtain:

$$S_v \approx \frac{s + (1-s)N^{\alpha}}{s} = 1 + \frac{p}{s}N^{\alpha}$$

Ideal case is when alpha = 1 -
"Gustafson's Law"

$$S_v(\alpha = 1) = s + (1-s)N$$

Work only including parallel fraction

Definition of “work” that only includes the parallel fraction p:

$$P_v^{sp} = p$$

Parallel Performance

$$P_v^{pp} = \frac{pN^\alpha}{T_v^p(N)} = \frac{(1-s)N^\alpha}{s + (1-s)N^{\alpha-1}}$$

Application Speedup

$$S_v^p = \frac{P_v^{pp}}{P_v^{sp}} = \frac{N^\alpha}{s + (1-s)N^{\alpha-1}}$$

SCI INSTITUTE

THE UNIVERSITY OF UTAH

Parallel Efficiency

$$\varepsilon = \frac{\text{performance on N CPUs}}{N \times \text{ performance on one CPU}} = \frac{\text{speedup}}{N}$$

Work defined as $s + pN^{\alpha}$

$$\varepsilon = \frac{S_v}{N} = \frac{sN^{-\alpha} + (1-s)}{sN^{1-\alpha} + (1-s)}$$

Work defined as pN^{α}

$$\varepsilon = \frac{S_v^p}{N} = \frac{N^{\alpha-1}}{s + (1-s)N^{\alpha-1}}$$

SCI INSTITUTE

THE UNIVERSITY OF UTAH

Karp-Flatt Metric

Given a parallel computation exhibiting speedup s on $p > 1$ processors, the experimentally determined serial fraction e is defined by:

$$e = \frac{\frac{1}{s} - \frac{1}{p}}{1 - \frac{1}{p}}$$

Iso-measures

Isoefficiency: How fast does the problem size have to grow as the number of processors grows to maintain constant efficiency.

Isotime: How does the number of processes and/or problem size have to change to deliver a solution in constant time.

Isomemory: Ho w does the memory usage change with problem size and processor number.

SCI INSTITUTE

THE UNIVERSITY OF UTAH