Métricas para medir desempenho

- Métricas
- *Speedup* e eficiência
- Limites a escalabilidade

Aspectos do desempenho

- Dois dos principais objetivos do projeto de aplicações paralelas consistem em obter-se:
  - Desempenho: a capacidade de reduzir o tempo de resolução do problema à medida que os recursos computacionais aumentam;
  - Escalabilidade: a capacidade de aumentar o desempenho à medida que a complexidade do problema aumenta.
- Os fatores que limitam o desempenho e a escalabilidade de uma aplicação estão ligados à:
  - Limites Arquiteturais
  - Limites Algorítmicos

Limites nos Algoritmos paralelos

- Limites Arquiteturais
  - Latência e Largura de Banda da camada de interconexão
  - Capacidade de Memória

- Limites Algorítmicos
  - Falta de Paralelismo (fração sequencial/concorrente)
  - Frequência de Comunicação
  - Frequência de Sincronização
  - Escalonamento Deficiente (granularidade das tarefas/balanceamento de carga)

Métricas usuais

- Algumas das métricas mais conhecidas para medir o desempenho são expressas em:
  - MIPS: acrônimo para *Millions of Instructions Per Second.*
  - FLOPS: acrônimoimo para *FLoating point Operations Per Second.*
- Programas de teste encontrados:
  - SPECint: conjunto de programas de teste (*benchmarks*) da SPEC (*Standard Performance Evaluation Corporation*) que avaliam o desempenho do processador em aritmética de inteiros (1992).
  - SPECfp: conjunto de programas de teste da SPEC que avaliam o desempenho do processador em operações de ponto flutuante (2000).
  - *Whetstone*: programa de teste sintético que avalia o desempenho do processador em operações de ponto flutuante (1972)
  - *Dhrystone*: programa de teste sintético que avalia o desempenho do processador em aritmética de inteiros (1984)

Comparando desempenho

- Medida básica: **Tempo de Execução**

- O sistema *A* é *n* vezes mais rápido que o sistema *B* quando:
  - $Texec(A) / Texec(B) = n$

- Maior desempenho ↔ Menor tempo de execução

Speedup / Eficiência

- *Speedup*
  - Medida de ganho em tempo
  - $Speedup(P) = Texec(1\ proc) / Texec(P\ procs)$
    - Onde P = número de processadores
      - $1 \leq Speedup \leq P$

- Eficiência
  - Medida de uso dos processadores
  - $Eficiência(P) = Speedup(P) / P$
  - $0 < Eficiência \leq 1$

Lei de Amdahl

- Limitação teórica para os ganhos de desempenho

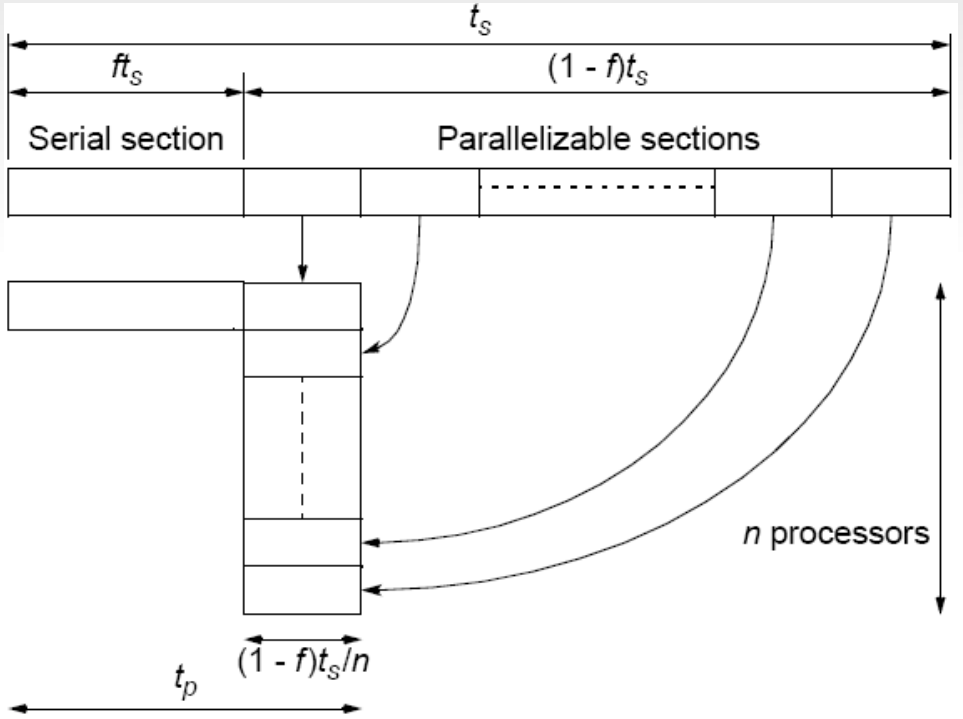
Programa serial: $Tserial = T0 = (s+q)T0$

- Onde:
  - $s+q = 1$
  - s corresponde a fração serial do código (impossível de ser paralelizada)
  - q corresponde a fração paralelizavel do código

Gene M. Amdahl, “Validity of the single processor approach to achieving large scale computing capabilities”, AFIPS spring joint computer conference, 1967

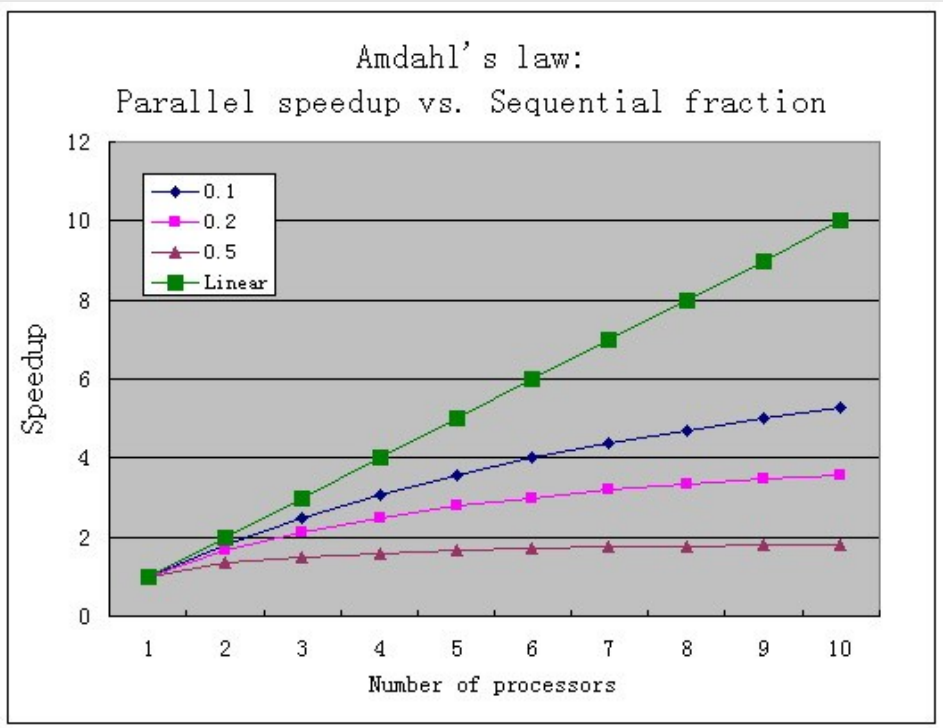
Lei de Amdahl (cont...)

- Supondo paralelização ideal:
  - *Tpar = sT0 + (q/p)T0*
- *Speedup = Tserial/Tpar = (s+q)/(s+q/P) = 1 / (s+q/P)*
- *Speedup = 1 / [s+(1-s)/P]*



Lei de Amdahl (cont...)

- Speedup =1 / [s+(1-s)/P]
- P=1 → Speedup = 1
- $P \to \infty$ → Speedup = 1/s
- $1 < P < \infty$,
  $1 \leq \text{Speedup} \leq 1/s$



Exemplo prático

- Em um procedimento de pintura gasta-se:
  - 30 minutos para carregar (*setup*) uma pistola de pintura
  - 30 minutos para descarregar e limpar a mesma pistola
  - 1 minuto para pintar cada metro quadrado

- Quanto tempo demorará a pintura de 300 metros quadrados por um único pintor?

- Se usássemos mais pintores, cada um com sua pistola de pintura, como seria?

Speed-up e Eficiência na pintura

Number of painters	Time	Speedup	Efficiency
1	360	1.0X	100%
2	210 = 30 + 150 + 30	1.7X	85%
10	90 = 30 + 30 + 30	4.0X	40%
100	63 = 30 + 3 + 30	5.7X	5.7%
Infinite	60 = 30 + 0 + 30	6.0X	very low

- Conclusão:
  - Conforme se vê na Lei de Amdhal o *Speedup* pontecial é limitado pela fração serial do procedimento

Exemplo 2: Copiar 10.240 páginas

- Suponha:
  - original entregue em ordem crescente das páginas numeradas;
  - copia deve ser entregue da mesma forma
  - uma única pessoa divide o original em partes iguais a serem copiadas, entrega para os copiadores e recolhe as cópias
  - cada copiadora copia uma única parte
  - 1 segundo para copiar uma página
  - 5 segundos para sub-dividir um bloco de páginas
  - 5 segundos para juntar dois blocos de páginas copiadas

- Estime o tempo (s) para realizar a tarefa utilizando
  - 1, 2, 4, 8, 16, 32 e 64 copiadoras

Estimativa Grosseira

Assume tempo total = tempo de entrega sequencial das partes + tempo de copiar uma parte + tempo de recolher a última parte

Máquinas	Tempo (s)				Ganho
	dividir	copiar	juntar	total	
1	0	10240	0	10240	1,00
2	5	5120	5	5130	2,00
4	15	2560	5	2580	3,97
8	35	1280	5	1320	7,76
16	75	640	5	720	14,22
32	155	320	5	480	21,33
64	315	160	5	480	21,33

falha hipótese da estimativa
(recolhe cópias enquanto o último copiador trabalha)

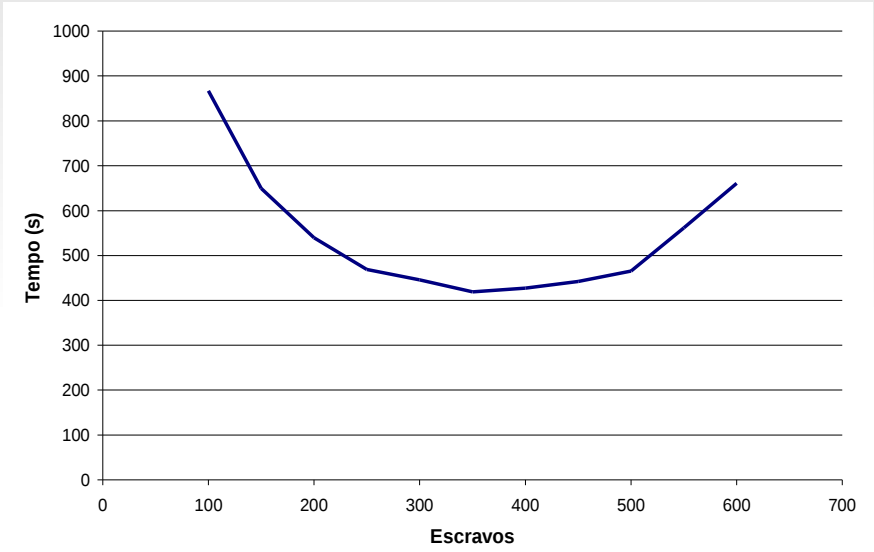
Copiar 10.240 páginas

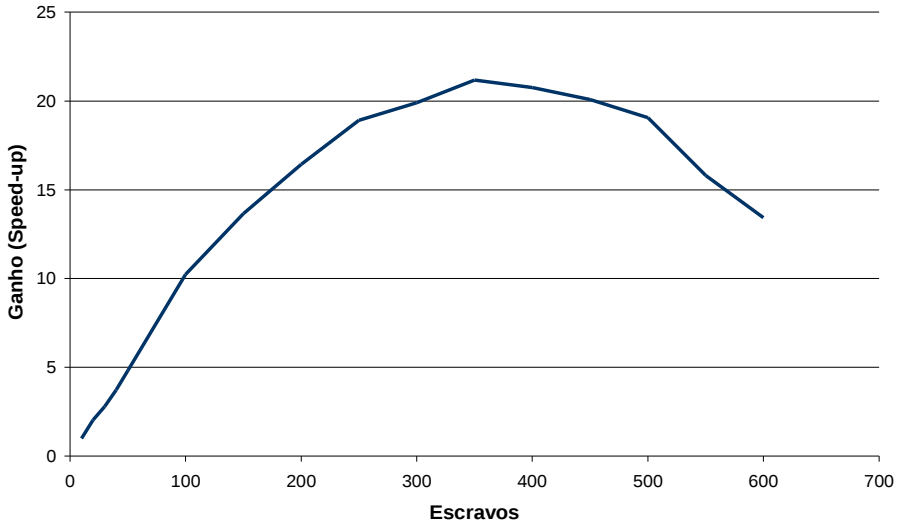
- Como utilizar eficientemente 1024 copiadoras?
  - Alterando os mecanismos de distribuição de dados e de coleta de resultados
- E se uma copiadora for 10% mais lenta que as outras?
  - típico de grande número de máquinas iguais
- E se uma copiadora quebrar?
  - Tolerância a falhas é desejável; as vezes, imprescindível

Exemplo 3: Cavar vala de 10km

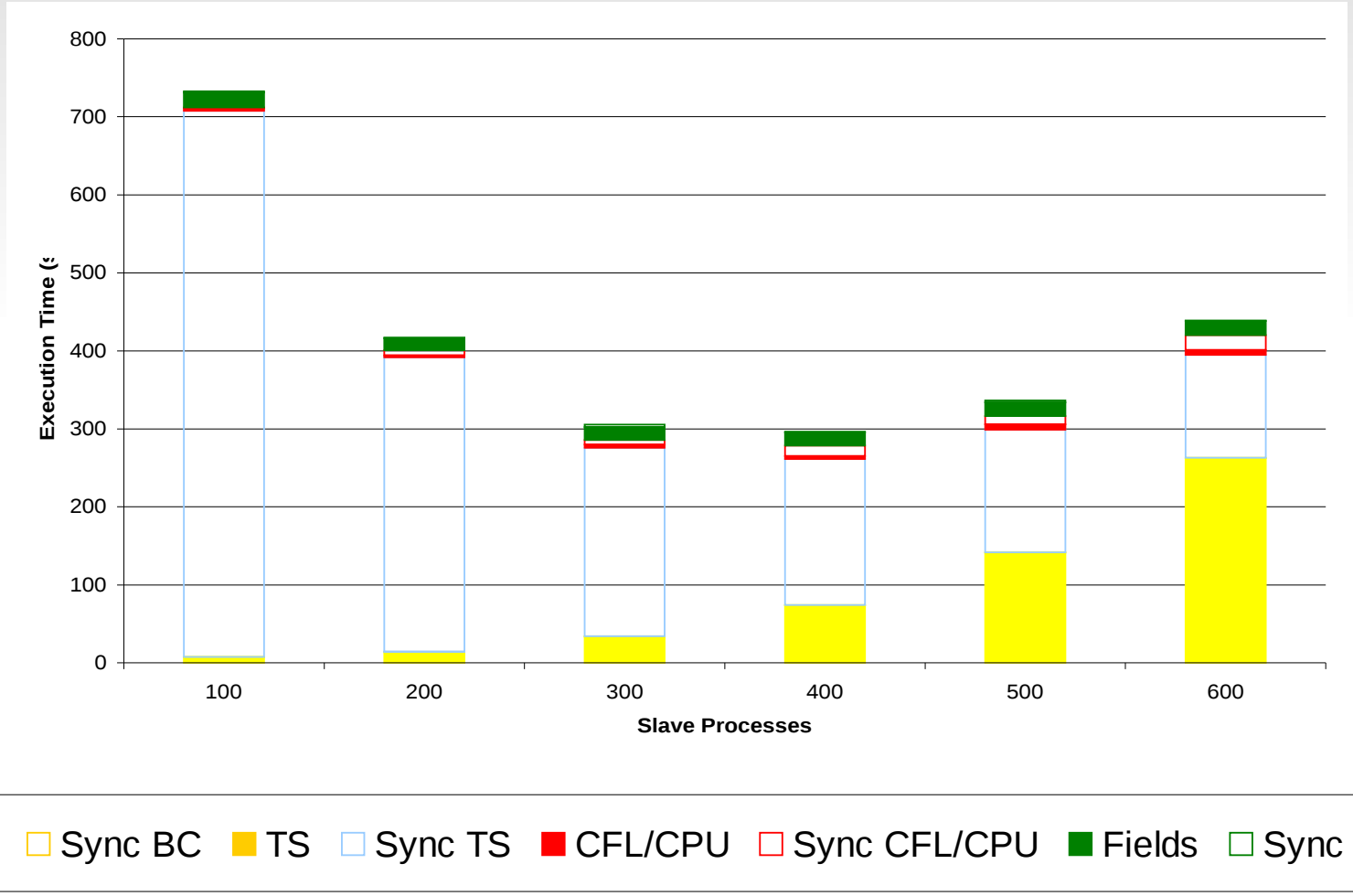
- Exemplo atribuído ao Prof. Siang W. Song (IME/USP)
- Similar ao exemplo anterior. Assuma terreno demarcado, número crescente de trabalhadores equipados e um capataz.
  - Quais são os fatores que impedem redução “ótima” do tempo de execução com o aumento do número de trabalhadores?
- E se a vala for vertical?
- Há problemas inerentemente sequenciais
  - alterar 1 bit na memória

Exemplo 4: Caso Real – Modelo de previsão de tempo BRAMS

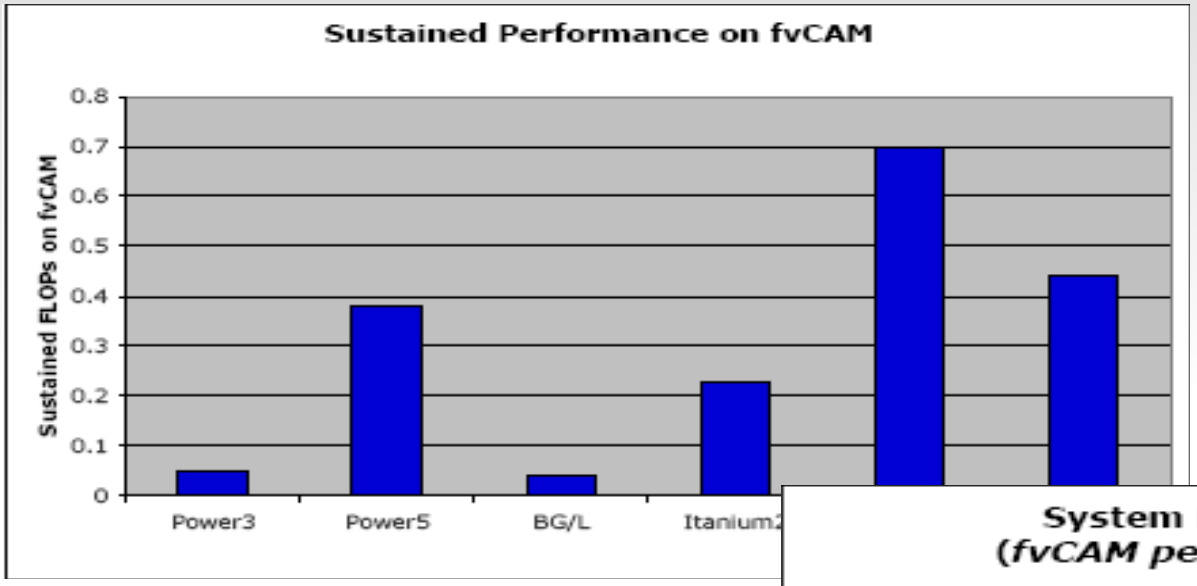




Identificar as razões do baixo desempenho requer instrumentar código por partes

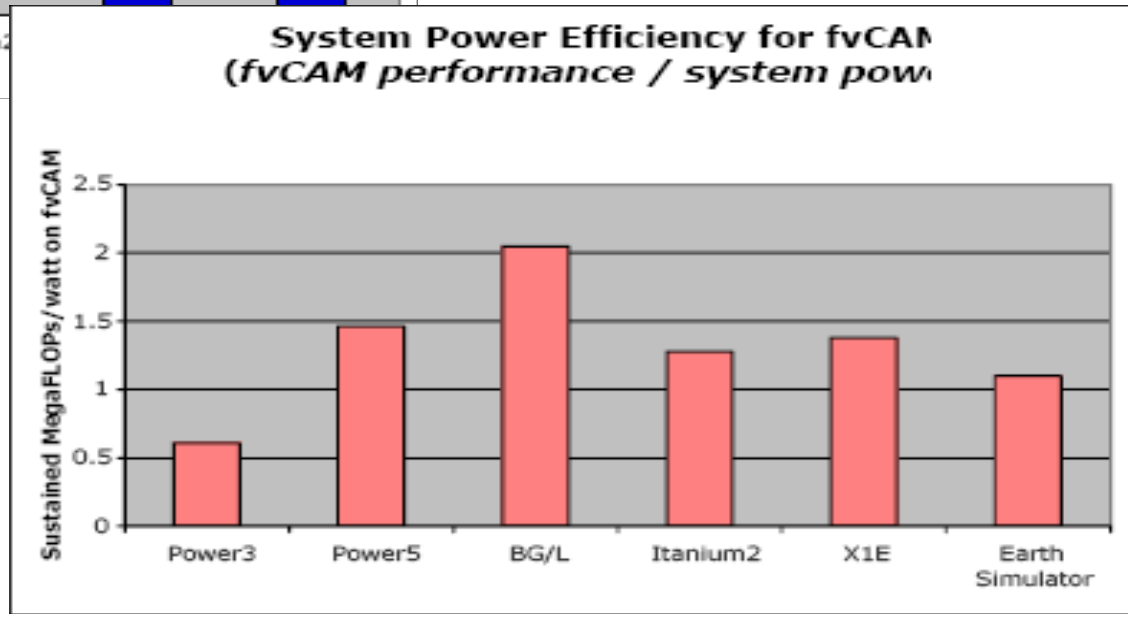


Métricas futuras (para milhares de processadores)



fvCAM:
Código de Meteorologia
Métrica de Eficiência:
MFlops efetivos

Nova métrica:
MFlops efetivos por watt consumido



Fonte: J. Shalf, D. Bailey, SC 2006

Outras Métricas aplicadas atualmente

- Lei de Gustafson-Barsis (1988):
  - fixed-time speedup model
- Limitação de memória proposta por Sun e Ni (1990)
  - memory-bounded speedup model

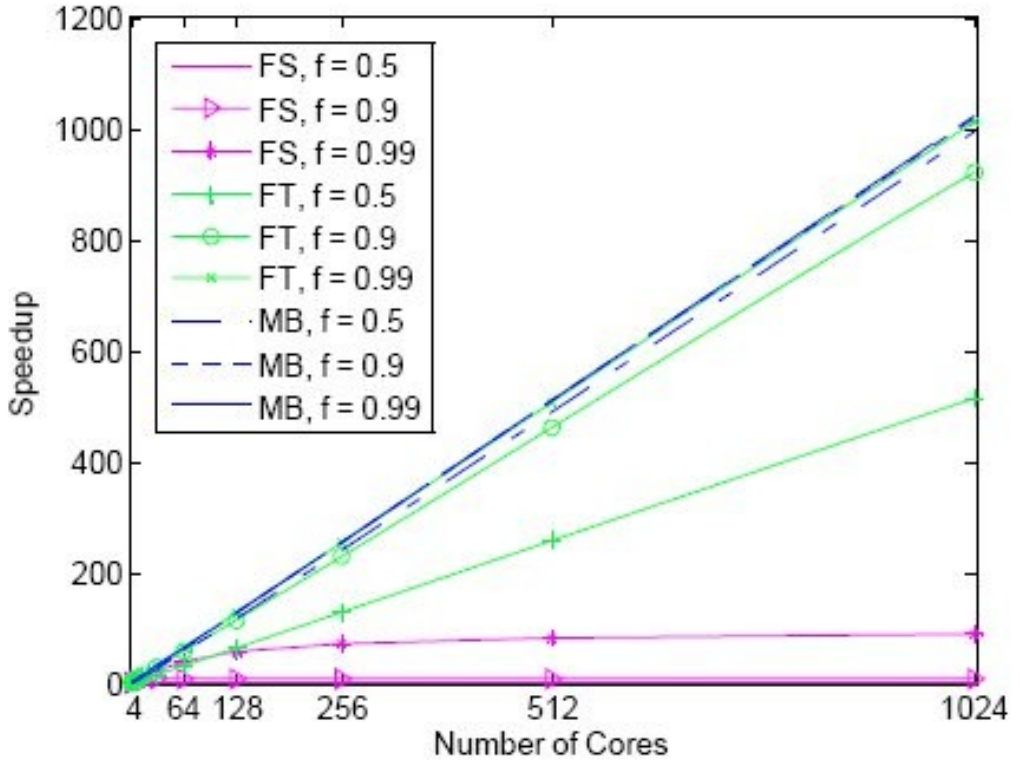


Fig. 4 Fixed-size, Fixed-time and Memory-bounded Speedup of a Multicore Architecture

Fonte: **Reevaluating Amdahl's Law in the Multicore Era**
Xian-He Sun, Yong Chen; SC 2008

Lei de Gustafson-Barsis

- $S(p) \leq p + f \times (1 - p)$
- Parte da premissa que toda aplicação tem uma fração inerentemente sequencial e que para se obter escalabilidade é necessário aumentar o tamanho da aplicação a medida que se aumentam o número de processadores

Lei de Gustafson-Barsis (cont.)

Lei de Gustafson-Barsis

- Considere que uma determinada aplicação executa em 220 segundos em 64 processadores. Qual é o *speedup* máximo da aplicação sabendo que por experimentação verificou-se que 5% do tempo de execução é passado em computações sequenciais.

$$S(p) \leq 64 + (0{,}05) \times (1 - 64) = 64 - 3{,}15 = 60{,}85$$

- Suponha que uma determinada companhia pretende comprar um supercomputador com 16.384 processadores de modo a obter um *speedup* de 15.000 num problema de fundamental importância. Qual é a fracção máxima da execução paralela que pode ser passada em computações sequenciais de modo a se atingir o *speedup* pretendido?

$$15.000 \leq 16.384 + f \times (1 - 16.384)$$

$$f \times 16.383 \leq 1.384$$

$$f \leq 0{,}084$$

Escalabilidade

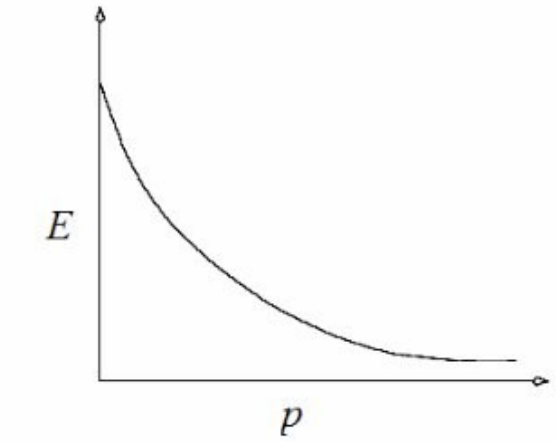
- Capacidade de “escalar” ou melhorar o desempenho de uma aplicação conforme se aumenta o número de processadores
- Duas definições se aplicam:
  - *Strong scalling*
    - Mantém o tamanho do problema e escala o número de processadores
    - Capacidade de rodar aplicações *n* vezes mais rápida, onde *n* é a quantidade de processadores utilizados (Speedup)
  - *Weak scalling*
    - Escala o tamanho do problema com o número de processadores
    - Capacidade de aumentar a carga de trabalho e a quantidade de processadores por um fator de *n* e manter o tempo de computação

Limitações para escalabilidade

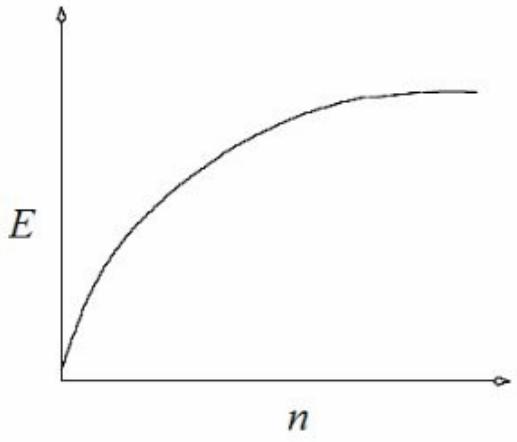
- Principais dificuldades, conforme J. Dongarra et all (Sourcebook of Parallel Computing, 2003)
  - Fração serial do código dominante (Lei de Amdahl)
  - Tempo despendido em comunicação ou coordenação de de tarefas
  - Desbalanceamento de carga entre os processadores

Eficiência e Escalabilidade

- Dos resultados anteriores podemos concluir que a eficiência de uma aplicação é:
  - Uma função decrescente do número de processadores
  - Tipicamente uma função crescente do tamanho do problema



tamanho do problema fixo (n)



número de processadores fixo (p)

Eficiência e Escalabilidade (cont.)

- Uma aplicação é dita de escalável quando demonstra a capacidade de manter a mesma eficiência à medida que o número de processadores e a complexidade do problema aumentam proporcionalmente.
- A escalabilidade de uma aplicação reflete a sua capacidade de utilizar mais recursos computacionais de forma efetiva

		1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs
Eficiência	n = 10.000	1	0,81	0,53	0,28	0,16
	n = 20.000	1	0,94	0,80	0,59	0,42
	n = 30.000	1	0,96	0,89	0,74	0,58