CAP-387(2016) – Tópicos Especiais em Computação Aplicada: Construção de Aplicações Massivamente Paralelas

Aula 43: Entrada/Saída Paralela

Celso L. Mendes, Stephan Stephany

LAC / INPE

Emails: celso.mendes@inpe.br, stephan.stephany@inpe.br

Entrada/Saída: E/S (I/O)

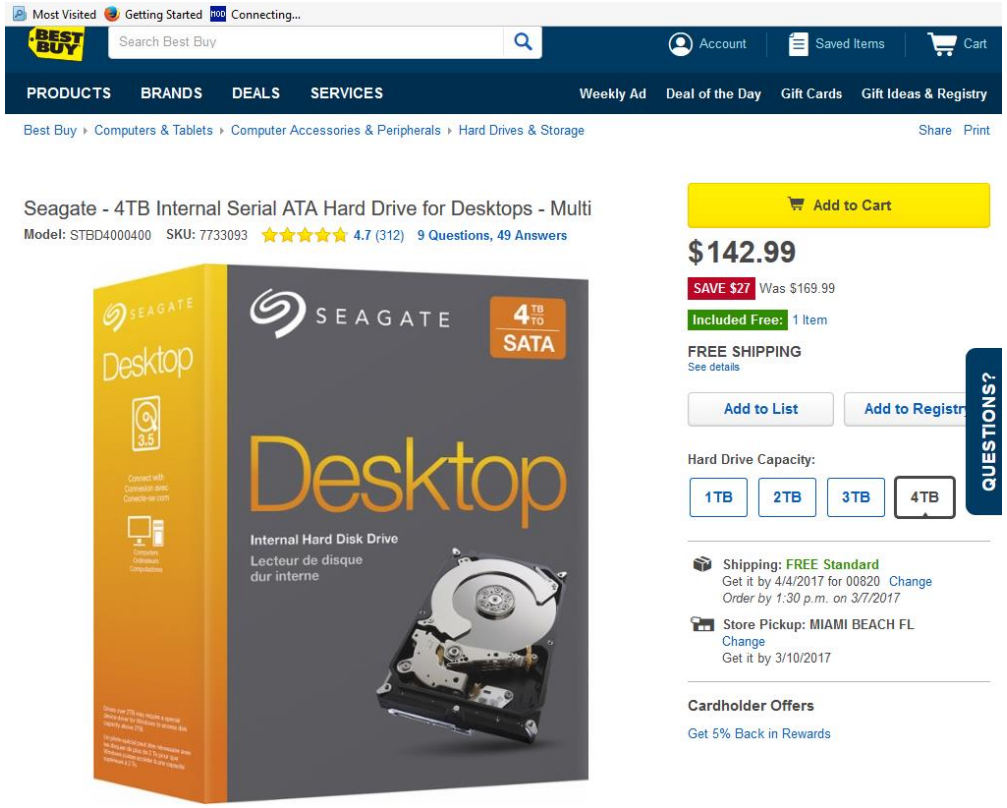
- Armazenamento de Dados
  - Muitas aplicações dependem de armazenamento persistente de dados, tanto para entrada dos dados a serem processados, como para arquivamento de resultados parciais ou finais
  - Forma mais tradicional: armazenamento em disco
  - Outras formas também utilizadas
    - Fita magnética: custo efetivo baixo para altos volumes
      - Usada em armazenamento terciário, de longo prazo
    - Memória Flash: baixo consumo de energia, bom desempenho; custo ainda relativamente alto

Parâmetros de Desempenho

- Desempenho de discos magnéticos
  - Latência típica: 2~10 ms (tempo de rotação até atingir a cabeça de leitura/gravação)
    - Ex: disco de 5.400 rpm nominal → 1 rotação = 11 ms
    - 3 ordens de grandeza mais lento que comunicação inter-nó
    - Muitas ordens de grandeza mais lento que processador
  - Largura de banda: > 100 MB/sec (para grandes volumes)
  - Desempenho é extremamente sensível ao padrão de uso
  - Benchmarks de I/O são raros e pouco informativos
    - Operações pouco representativas de aplicações reais
    - Uso eventual como indicador de limites alcançáveis
    - Benchmark típico: IOR

Exemplo Atual de Desempenho

- Disco Seagate 4TB:



Especificações:

- RPM: 5.400
  - ➢ Latência = 5,5 ms
- Larg.Banda: 6 GB/s
  - ➢ T_{transf}(~6 KB) = 1 µs
  - ➢ T_{transf}(~6 MB) = 1 ms

Tempo Total de Acesso:

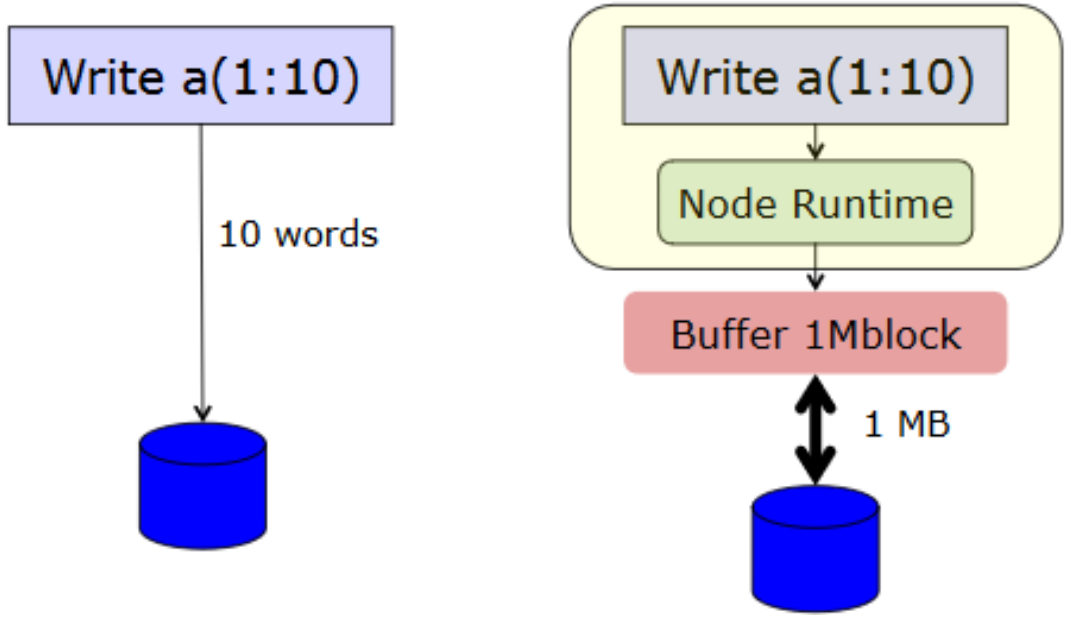
- T_{total}(6 KB) = 5,501 ms
- T_{total}(6 MB) = 6,5 ms

Arquivos e Sistemas de Arquivos

- Arquivo: Coleção ordenada de bytes
- Sistema de Arquivos: gerencia coleções de arquivos e suas propriedadoes, tais como:
  - Tamanho
  - Restrições de acesso
  - Cotas
  - Leitura e escrita de dados

Modelamento de I/O

- **Modelagem:** Forma para analisar operações de I/O
  - Visão típica do usuário:
  - Modelo mais preciso:



Dificuldades de Modelamento

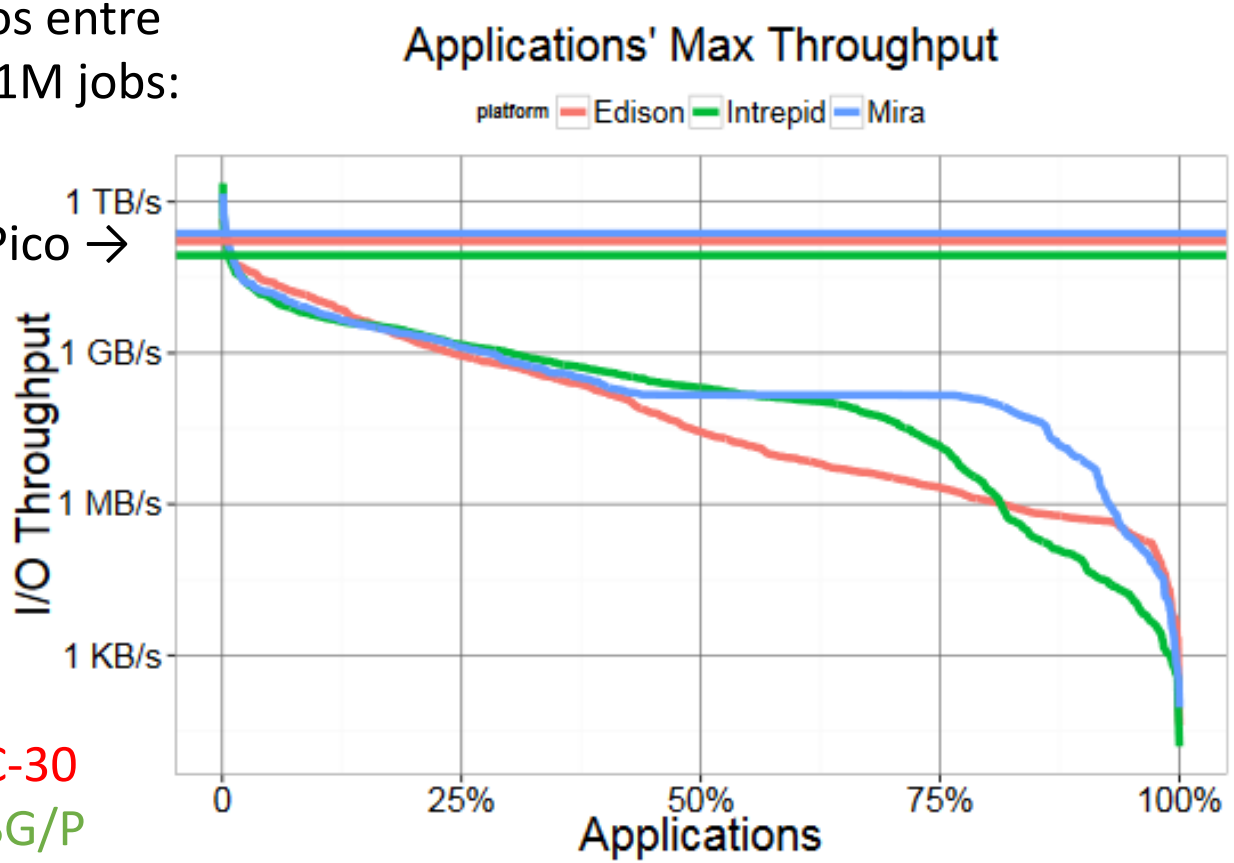
- Caches de disco raramente representadas
  - Difícil de modelar corretamente
- Operações de metadados raramente representadas
  - Acesso permitido ao arquivo?
  - Qual a data do ultimo acesso ao arquivo?
  - Qual o tamanho do arquivo?
  - Onde o arquivo está localizado?

Desempenho Observado

- Possível desencontro Modelo x Hardware
  - Modelo de programação pode não representar operação real
  - Exemplos:
    - leitura = leitura de vários blocos
    - escrita = read-modify-write
- Semântica pode não ser a representada pelo modelo
  - Ex: semântica de opeação de leitura ou escrita segue modelo sequencial
  - Difícil representar caches, operações de metadados, etc

Desempenho de I/O em Aplicações

Dados obtidos entre 2010/2014, >1M jobs:



75% das aplicações não chegam a 1 GB/s! (~1% do pico de I/O)

Edison: Cray XC-30
Intrepid: IBM BG/P
Mira: IBM BG/Q

Fonte: Luu et al:
A Multiplatform Study of I/O Behavior on Petascale Supercomputers, HPDC'15

Desempenho de I/O em Aplicações

- Coleta de Dados de Desempenho de I/O: Darshan
  - Dados são coletados automaticamente, e armazenados para análise após a execução
  - Permite análise contínua, em escala, ao nível de aplicações individuais e de sistema completo.
  - Ref.: P. Carns et al: *Understanding and improving computational science storage access through continuous characterization*, ACM Trans. on Storage, 7(3):8, 2011.

Semântica POSIX em I/O

- Ordenamento de Escrita/Leitura:
  - Uma vez terminado um *write* em algum processo, qualquer outro processo deve poder ver o seu efeito
  - Sistema de arquivos deve garantir este ordenamento
    - Para garantir isso a uma certa aplicação, pode haver impactos de desempenho em outras aplicações!
    - Sistema de arquivos atende a diversas aplicações, simultaneamente

Semântica POSIX em I/O

- Exemplo:

Processo 1	Processo 2
read a	. . .
. . .	*write b*
read b	

Se, ao ler *a*, o Processo-1 ler um bloco contendo *a* e *b*, ao ler b é necessário novo acesso ao arquivo (obter novo *b*)

- Problema similar ao de consistência de caches!

Abertura e Fechamento de Arquivos

- Abertura: *open*
  - Retorna um descriptor do arquivo
  - Problema: não há em Unix um "*exclusive access open*"
    - Sist.Arquivos deve assumir que outros processos também poderão abrir o arquivo
- Fechamento: *close*
  - Envia dados finais ao arquivo, libera descriptor
  - Problema: quando os dados de fato chegam ao disco?
    - Se houver cache, dados podem não ser enviados ao disco!
    - Como confirmar os resultados de um benchmark de I/O?

Escrita, Leitura e Avanço

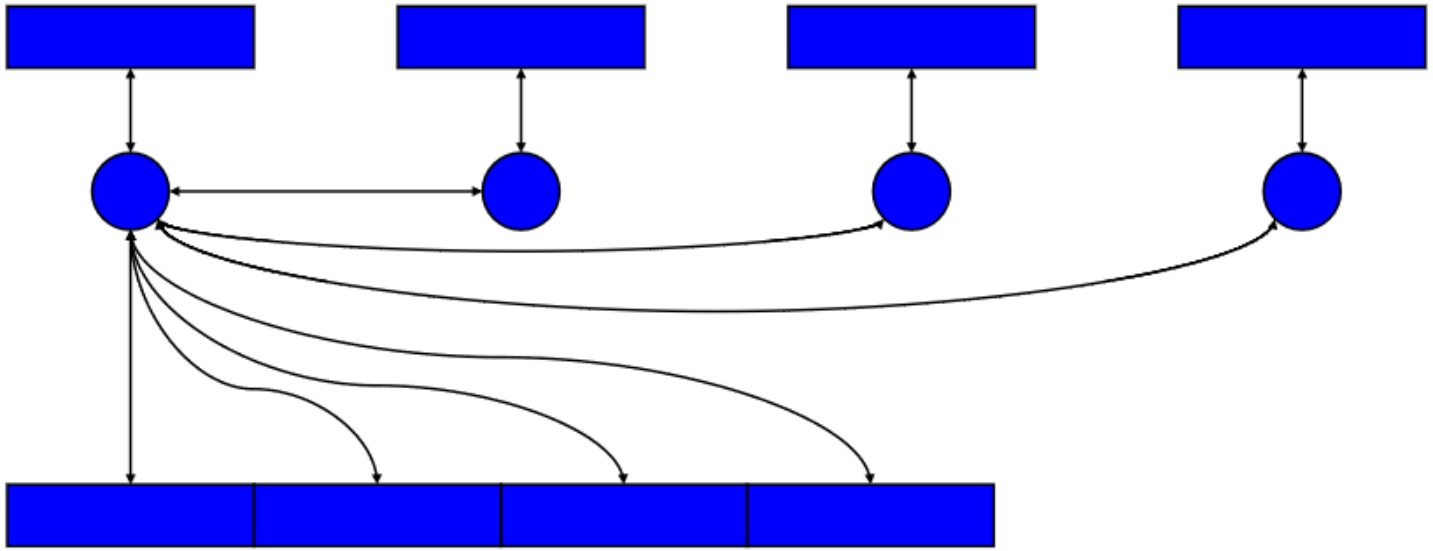
- Avanço/Procura: *seek*
  - Muda posição no arquivo para a próxima leitura/escrita
  - Expectativa do usuário: operação rápida (mas pode demorar, se a implementação exigir escrita/flush de dados, e/ou se houver vários processos fazendo *seek*)
- Leitura: *read/fread*
  - *read* (sem buffer) deveria ser mais rápido que *fread* (com buffer), mas pode não ser verdade para poucos dados
- Escrita: *write/fwrite*
  - Expectativa de cache e de melhor desempenho com grande volume de dados; contudo, alinhamento dos dados com os blocos do disco pode ser mais importante

E/S em Programas Paralelos

- Organização de E/S:
  - Deve-se considerer toda a aplicação, não apenas partes
    - Mesmo arquivo pode ter que ser usado em diversas fases
- Possibilidades de organização num programa
  - Um único arquivo por programa
    - Concorrência total; desempenho pode ser problemático
  - Um arquivo por processo
    - Evita overheads de concorrência para o sist. arquivos
    - Mais trabalho para o usuário: lidar com muitos arquivos
  - Um arquivo por nó/grupo/…
    - Solução de compromisso, intermediária; concorrência parcial, em pequena escala – bom desempenho ainda possível

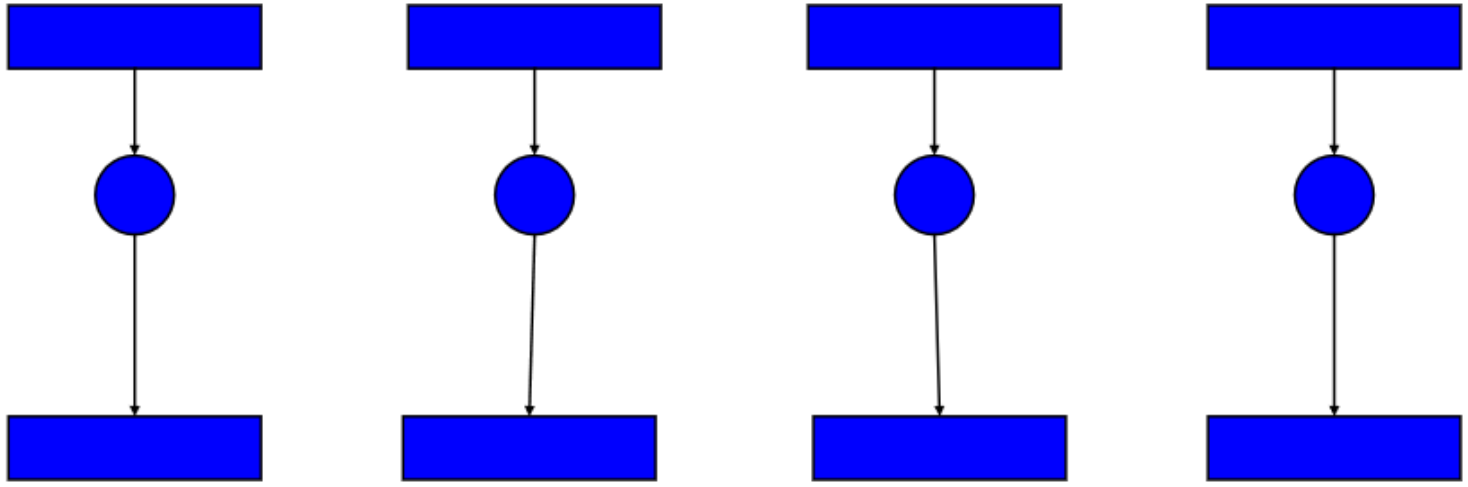
E/S sem Paralelismo

- Características principais
  - Ausência total de paralelismo
  - Desempenho pode ser pior que programa sequencial
  - Geralmente usado em “legacy codes”



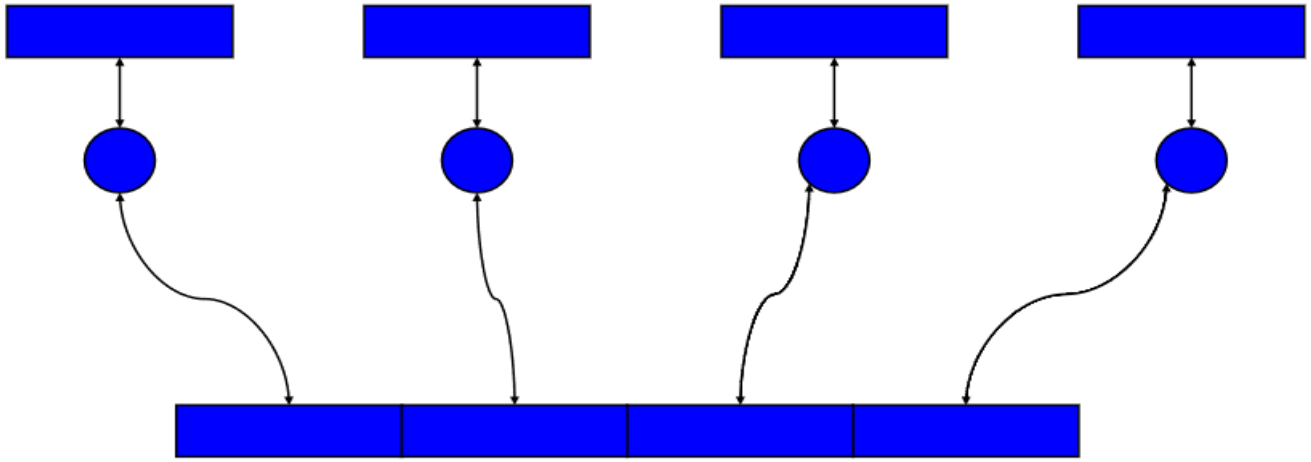
E/S Paralela Independente

- Características principais
  - Paralelismo total de E/S: um arquivo por processo
  - Overhead para o usuário: lidar com muitos arquivos (pequenos)
  - Pode ser usado em “legacy codes”, ou para evitar concorrências no sist. arquivos



E/S Paralela – Arquivo Único

- Características principais
  - Paralelismo total de E/S
  - Acessos concorrentes ao arquivo comum
  - Desempenho pode variar bastante, dependendo da implementação do sist.arquivos e do padrão de acessos



Outras Alternativas

- Sist. arquivos com outros modelos de consistência:
  - NFS: sem consistência; seguro apenas p/ acesso serial
  - PVFS: define escritas sem sobre-posição
  - HDFS: Acesso paralelo a arquivos “fixos”
- Sist. arquivos com suporte a paralelismo
  - Vários servidores de armazenamento; arquivos são “fatiados” (striped) pelos vários servidores, e podem então ser acessados em paralelo por vários processadores
    - Exemplos: GPFS, Lustre, etc.