Unidade VI

Bancos de Dados Paralelos

- Paralelismo de I/O
- Paralelismo em Consultas
- Paralelismo em Operações

Bancos de Dados Paralelos

- BDs exigem cada vez mais das máquinas
  - Cada vez mais consultas são feitas nos BDs
  - Cada vez mais aplicações dependem dos BDs
  - Novos tipos de dados, como som e imagem, exigem maior poder de processamento e armazenamento
- Máquinas paralelas vem sendo usadas para supotar uma carga maior de trabalho dos SBDs
  - Vários processadores executam as operações em paralelo → uso de controle de concorrência
  - Memória e disco podem ser compartilhados ou não
  - Custo de máquinas multi-processadas está caindo

Bancos de Dados Paralelos

- Sistemas paralelos são usados para aumentar:
  - Escala: processar mais transações
  - Desempenho: tornar o processamento mais rápido
- Problema: mais hardware → mais falhas
  - Solução: replicar os dados → controlar consistência
- Paralelismo pode ser usado:
  - Na entrada e saída de dados (I/O)
  - No processamento de consultas
    - Inter- e Intra-consultas
  - No processamento de operações individuais
    - Inter- e Intra-operação

Paralelismo de I/O

- Instruções de entrada e saída podem ser processadas em paralelo
  - Basta particionar os dados em vários dispositivos de armazenamento de dados
  - Operações podem ser executadas em paralelo se cada uma acessar um dispositivo
  - Cada processador pode trabalhar com os dados de uma partição
  - Um plano de execução em paralelo podem ser criado ao otimizar a operação

Paralelismo de I/O

- Técnicas de particionamento:
  - Round-Robin: cada nova tupla é colocada em um dispositivo diferente
  - Hash : um ou mais atributos da relação são usados por uma função hash para determinar o dispositivo
  - Por faixa: cada dispositivo armazena as tuplas com um certo atributo dentro de uma faixa de valores
- Comparação
  - Hash e faixa são ótimos para fazer seleções de igualdade do atributo de particionamento
  - Faixa é o melhor para procurar por faixas de valores do atributo de particionamento
  - São equivalentes para acesso seqüencial à relação

Paralelismo em Consultas

- Paralelismo Inter-consultas
  - Consultas são executadas em paralelo pelos vários processadores da máquina paralela
  - Aumenta o número de consultas processadas, mas não diminui o tempo de resposta das consultas
  - Desnecessário trocar mensagens entre processadores se tivermos paralelismo de disco e memória
  - É preciso garantir a consistência quando dois ou mais processadores acessam o mesmo dado
    - Usar controle de concorrência
    - Fundamental usar operações de bloqueio se os processadores usarem cache

Paralelismo em Consultas

- Paralelismo Intra-consultas
  - Partes de uma consulta são executadas em paralelo nos diversos processadores
  - Diminui o tempo de resposta das consultas
  - Planos de execução são feitos na forma de árvores, e cada ramo pode ser processado em paralelo
  - Pipelining pode ser feito entre operações → a saída de uma operação é a entrada da outra
  - A memória e os discos compartilhados servem para troca de dados entre processadores caso não seja possível fazer o pipelining

Paralelismo em Operações

- Formas de paralelismo entre operações de uma mesma consulta
  - Paralelismo inter-operação: as operações de uma consulta são executadas em paralelo
  - Paralelismo intra-operação: uma operação é dividida em várias partes, sendo cada uma delas executada por um processador
- As duas formas de paralelismo podem ser usadas simultaneamente por um SBD paralelo

Paralelismo em Operações

- Paralelismo Inter-Operação
  - Escalas de execução podem prever a avaliação de operações em paralelo
  - Cabe ao otimizador de consultas determinar as operações que serão executadas em paralelo
  - Paralelismo é usado se o ganho de processamento for maior que o custo de comunicação
  - As escalas possíveis são ainda mais numerosas que nas execuções seqüenciais → usar heurística (ex.: escolher a escala seqüencial mais eficiente e paralelizar)
  - É necessário alocar os recursos – processador, memória e disco – que serão usados por cada operação executada em paralelo

Paralelismo em Operações

- Formas de paralelismo inter-operação
  - Paralelismo Independente
    - Operações de uma consulta não dependem necessariamente uma da outra
    - Operações independentes são executadas em paralelo por processadores diferentes
  - Paralelismo Pipeline
    - Algumas operações não são independentes
    - As tuplas produzidas por uma operação vão sendo passadas para as operações que precisam delas
    - Cada operação é executada por um processador

Paralelismo em Operações

- Paralelismo Intra-Operação
  - Usada quando a operação trabalha com um número muito grande de tuplas
  - Cada processador pode executar a operação trabalhando com uma parte das tuplas
  - Exige algoritmos paralelos, que permitam o particionamento da operação
- Seleção paralela
  - Cada processador procura na sua partição dos dados as tuplas nas quais a condição de seleção é válida
  - Simplificado se a tabela for particionada por faixa ou hash → não é preciso usar todos os processadores

Paralelismo em Operações

- Classificação em paralelo
  - Algoritmo de classificação por faixas
    - Particionar a tabela por faixas com base no atributo de classificação
    - Ordenar cada partição independentemente
  - Algoritmo de sort-merge externo paralelo
    - Supondo que a tabela já foi particionada em vários discos usando qualquer método
    - Sort: cada processador ordena uma partição
    - Merge: partições já em ordem são particionadas novamente por faixas, e são enviadas aos processadores, que as unem às outras partições recebidas e as classificam

Paralelismo em Operações

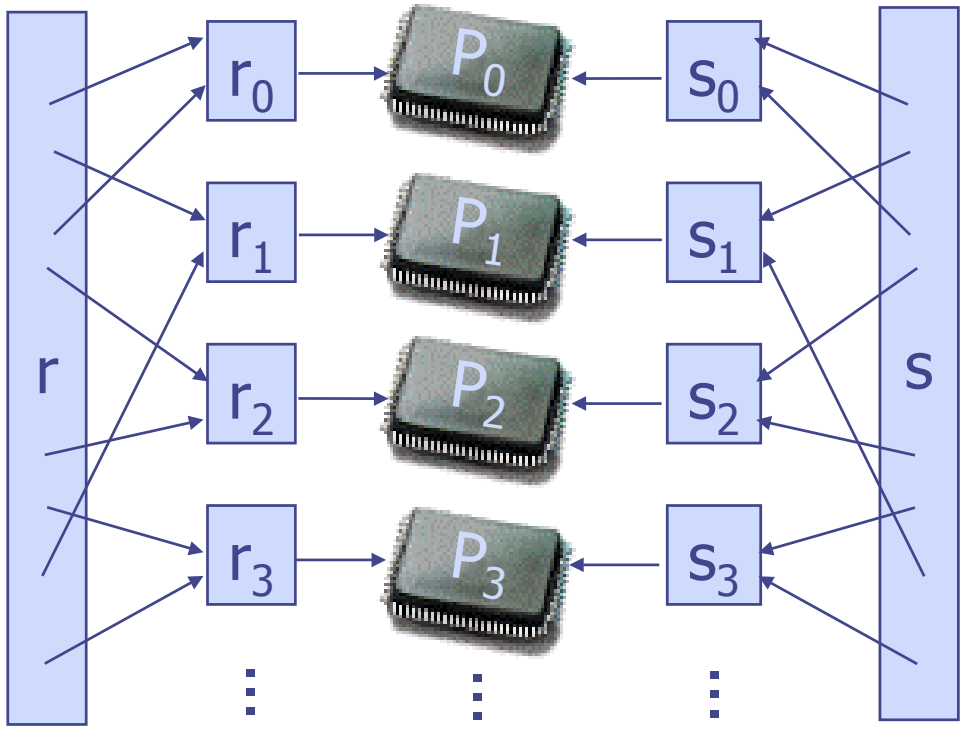
- Eliminação de duplicatas em paralelo
  - Pode ser feita na classificação das tuplas ou nas faixas ou hashs por cada processador
- Projeção paralela
  - Efetuada pelos processadores à medida que as tuplas são lidas em paralelo dos discos
- Agregação
  - Particionar em faixa ou hash a relação usando os atributos de agrupamento
  - Computar os valores agregados em cada processador

Paralelismo em Operações

- Junção paralela
  - Algoritmos dividem entre os processadores os pares de tuplas a serem testados na junção
  - Pares de tuplas oriundos de todos os processadores para os quais a condição de junção é válida são reunidos para se ter o resultado final da junção
  - Algoritmo de junção particionada
    - Usado em junções naturais e junções de igualdade
    - A mesma função de faixa ou hash deve ser usada para particionar as tabelas
    - Cada processador faz a junção das partições usando um algoritmo de junção seqüencial

Paralelismo em Operações

- Algoritmo de junção particionada

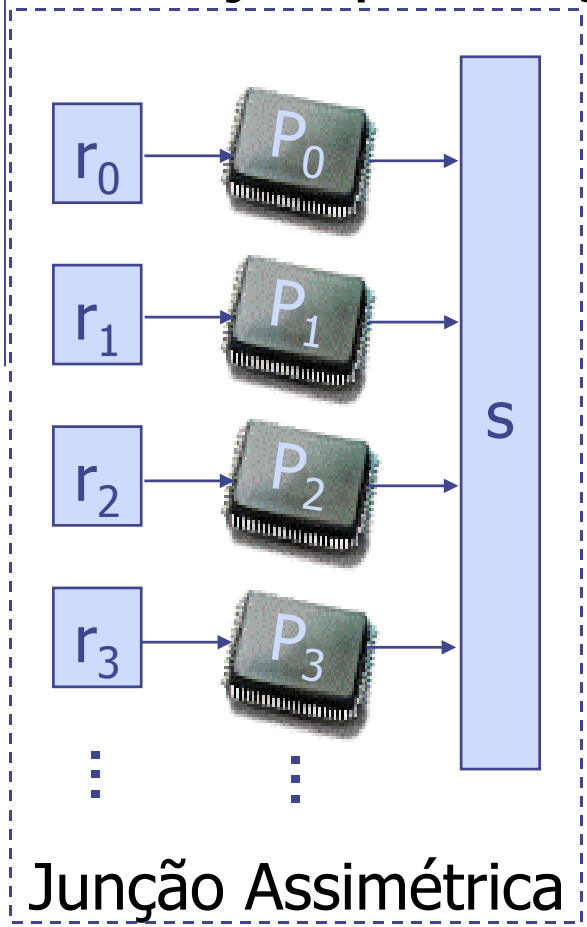


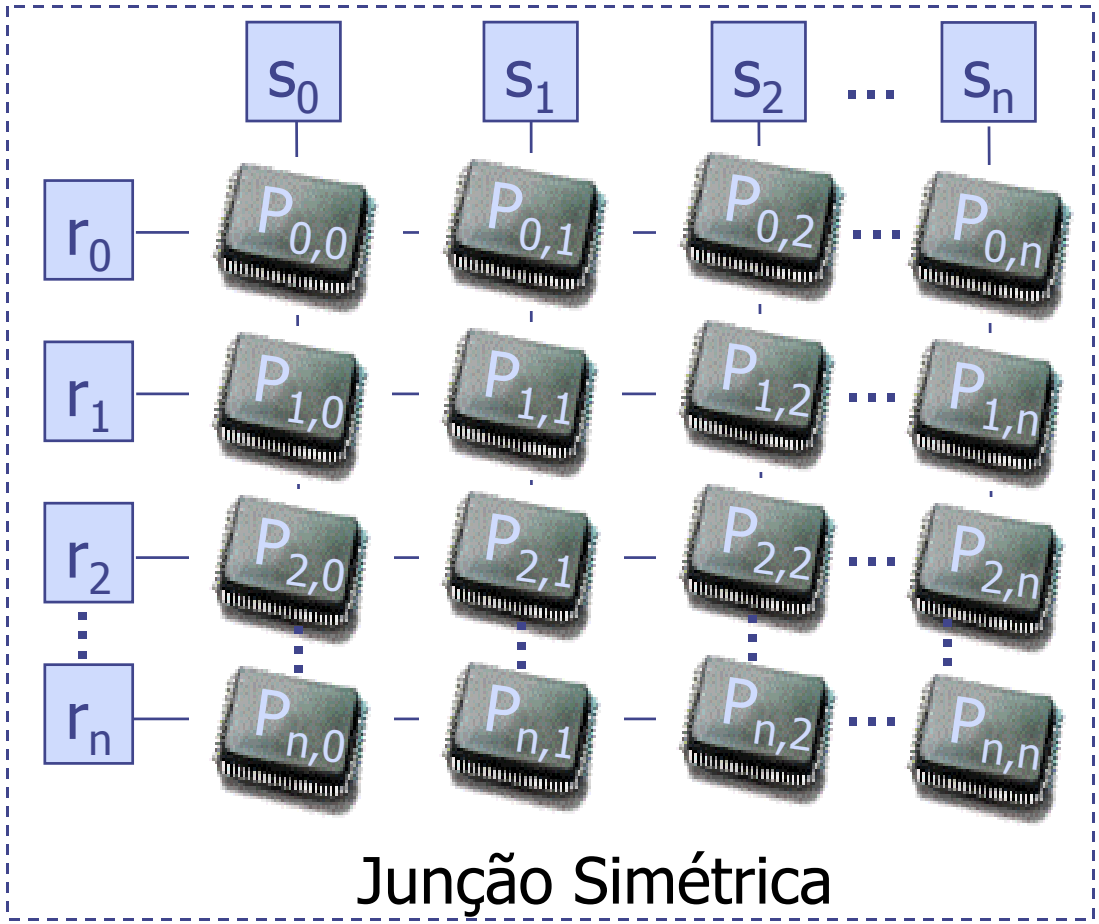
Paralelismo em Operações

- Junção paralela (cont.)
  - Algoritmo de junção por fragmentação e replicação
    - Usado para junções que não sejam de igualdade ou naturais
    - As duas tabelas são fragmentadas em m e n partições respectivamente
    - Cada processador executa a junção entre uma partição da primeira tabela e outra da segunda tabela, usando qualquer algoritmo de junção local
    - Ao final, as tuplas resultantes são reunidas para obter o resultado
    - Caso especial: junção assimétrica, onde n = 1

Paralelismo em operações

- Junção por fragmentação e replicação





Paralelismo em Operações

- Junção paralela (cont.)
  - Algoritmo de junção hash paralela
    - Usa funções de hash para dividir as relações
    - Uma variante do algoritmo de junção hash calcula o resultado da junção
  - Algoritmo de junção paralela de laço aninhado
    - Cada processador faz a junção indexada de laço aninhado da relação menor, replicada em todos os processadores, com uma partição da relação maior
    - Usa índice da relação maior no atributo de junção
    - É usado quando uma das relações é muito menor que a outra