Sistemas Computacionais Distribuídos

Prof. Marcos José Santana

SSC-ICMC-USP

São Carlos, 2008

Grupo de Sistemas Distribuídos e Programação Concorrente

Departamento de Sistemas de Computação - SSC

Sistemas Computacionais Distribuídos

Técnicas para Permitir Recuperação de Dados

Conteúdo

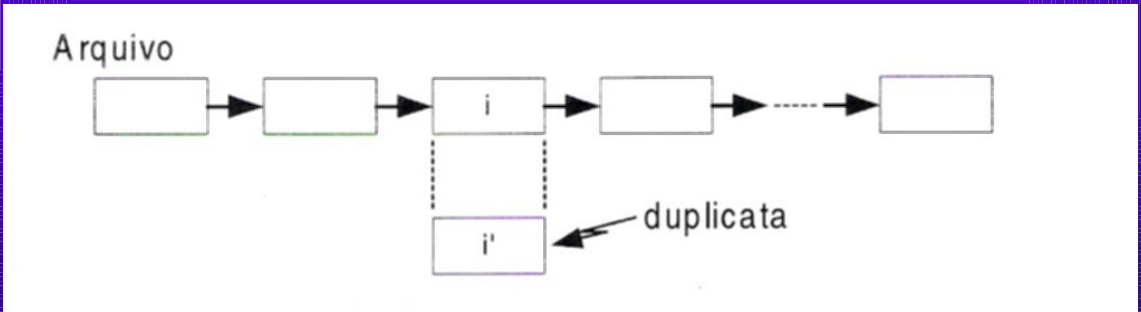
- Conceito Básico
- Shadow-Page
- Undo-Redo Log
- Intentions Log
- Tentative Versions
- O Problema do “Commit Point”

Conceito Básico

- Dados que estão atualizados não podem JAMAIS ser re-escritos (ou escritos por cima – OVER WRITTEN), a não ser que eles possam ser recuperados a partir de outras informações

Shadow-Page

- Técnica mais utilizada em servidores de arquivos
- Toda página é um arquivo que é considerado por uma transação para escrita ⇒ deve ser duplicada!!!



Shadow-Page

- As novas páginas ⇒ “shadow-pages” ocupam blocos livres do disco
- O mapa de páginas do arquivo deve ser atualizado (no final) ⇒ também usam-se “shadow-pages”
- As páginas atualizadas são transformadas em páginas permanentes no final da transação (*commit point*)
- Se uma transação é abortada antes do seu final ⇒ as páginas antigas continuam válidas

Shadow-Page

- É um método eficiente em termos de confiabilidade, mas pode trazer problemas de desempenho
- Ordenação das páginas ⇒ reduz desempenho
- Páginas velhas ⇒ liberação (ok)
- Páginas novas ⇒ liberação (se falhou)
  - Coleta de lixo
- Depois de um crash ⇒ limpeza

Undo-Redo Log

- Manter anotações sobre TODAS as modificações efetuadas no arquivo
- Cada atualização ⇒ ⟨valor antigo – valor novo⟩
- Anotações permitem
  - Desfazer toda a transação
  - Refazer toda a transação
- Atualização é feita no local
- Técnica considerada BOA!

Intentions Log

- Baseado no uso de listas de intenções
- Atualização não destrói dados
- Estados para um bloco de dados
  - Livre (a)
  - Alocado (b)
  - Intenção de liberação (c)
  - Intenção de alocação (d)
- Quando um bloco B vai ser atualizado ⇒ um bloco B' livre (a) é alocado em (d). B é então colocado em (c). Novo bloco (B') pode então ser escrito

Intentions Log

- No final ⇒ lista de pares
  - Blocos com intenção de alocação
  - Blocos com intenção de liberação
- Se necessário ⇒ estado inicial é restaurado
- Se a transação é encerrada com sucesso ⇒
  - Intenção de alocação ⇒ alocados
  - Intenção de liberação ⇒ livres

Tentative Versions

- Baseada no uso de versões
- Não reescreve um dado, isto é, não é destrutiva!
- Atualização ⇒ Nova versão é criada (arquivo todo)
- Se a transação termina com sucesso ⇒ nova versão é colocada ao público
- Se a transação falhou ⇒ nova versão é ignorada
- Variação
  - Versão antiga é descartada!!! Uma espécie de “shadow-file”

O Problema do “Commit Point”

- Parte crítica de uma transação
- Todas atualizações são, inicialmente, temporárias
- Depois do “commit point” ⇒ permanentes
- Servidor TEM de mudar todas as atualizações “provisórias” para “permanentes” mesmo se houver um crash
  - Completar todas as transações bem sucedidas
  - Desfazer todas as abortadas

O Problema do “Commit Point”

- Para uma transação envolvendo apenas 1 servidor
  - Armazenar confiavelmente TODAS as informações necessárias para completar a transação
  - Mudar atomicamente o estado da transação para “terminada” (este é o commit point)
  - Tornar as alterações permanentes
  - Registrar o fim da transação

Fim!