CONSISTÊNCIA DE DADOS

DCE540 - Computação Paralela e Distribuída

Atualizado em: 17 de janeiro de 2023

Iago Carvalho

Departamento de Ciência da Computação

CONSISTÊNCIA E REPLICAÇÃO DE DADOS

É muito comum utilizarmos replicação de dados

- Melhorar performance do sistema
- Realizar cópias de segurança de dados

Quando existem dados replicados, cria-se um novo problema

- Consistência de dados
- Caso um dado seja alterado, todas as cópias devem ser atualizadas

Este é um problema relativamente grave quando temos diversos componentes querendo acessar e alterar um mesmo dado de forma simultânea

RAZÕES PARA REPLICAÇÃO

1. Tolerância a falhas e segurança
    - Caso o servidor que contém uma cópia falhe, é possível simplesmente migrar para outro
    - Caso um servidor esteja muito congestinado, pode-se utilizar outro
    - Segurança contra dados corrompidos

2. Performance do sistema distribuído
    - Sistemas distribuídos tem que ser escaláveis
    - Em sistemas de grande porte (nível nacional ou internacional), é importante termos os dados sempre próximos (fisicamente) dos usuários
    - Balanceamento de carga

CONSISTÊNCIA DE DADOS

Quanto estamos trabalhando com consistência de dados, devemos levar em consideração duas coisas

1. Atual implementação da consistência
    - Como os dados estão distribuídos
    - Em que localização (física) eles estão
    - O acesso aos dados é rápido e efetivo?

2. Como manter os dados consistentes
    - Consistência forte
    - Cache
    - Algoritmos de atualização de dados

REPLICAÇÃO COMO TÉCNICA DE ESCALONAMENTO

A principal idéia é replicar os dados em lugares geograficamente diferentes

- Distribuídos fisicamente através dos locais onde um sistema distribuído é utilizado
- Esta distribuição leva em conta dois aspectos
  1. Número de usuários em um determinado local
  2. Distância dos usuários até o local

Entretanto, esta replicação pode levar a inconsistência

- Mesmo que não exista inconsistência, também deve-se considerar o maior tráfego de rede necessário para atualizar todos os arquivos

MAU USO DE REPLICAÇÃO

Considere uma aplicação *P* qualquer

- Ela acessa um dado replicado *N* vezes por minuto

O dado é completamente atualizado *M* vezes por minuto

- $N \ll M$

Tráfego desnecessário na rede

- Principalmente se o arquivo replicado for grande

TIPOS DE CONSISTÊNCIA DE DADOS

Consistência apertada (replicação síncrona)

- Dados só são ditos serem replicados quando realmente são iguais
- Desta forma, é necessário mante-los sempre atualizados
- Atualização é muito cara

Consistência frouxa (replicação assíncrona)

- Dado é atualizado somente onde é realizada a operação
- *Clock* é salvo e enviado para as réplicas
- Caso alguém tente acessar uma réplica, o dado é então atualizado

MODELOS DE CONSISTÊNCIA CENTRADO EM DADOS

Um modelo de consistência é basicamente um acordo entre os processos

- Todos eles concordam com um conjunto de regras
- Todo processo de atualização de dados deve seguir as regras acordadas
- Modelos de consistência são implementados pelo *middleware* de um sistema distribuído

Modelos de consistência centrado em dados

- Classifica operações como
  - Leitura
  - Escrita
- Operações de escrita tem que ser propagadas para as réplicas dos dados

CONSISTÊNCIA CONTÍNUA

Refere-se ao *quanto* de inconsistência um sistema distribuido pode tolerar

- Uma linha de código
- 5% do banco de dados alterado
- Duas atualizações do repositório
- Três horas de diferença
- . . .

Ao atingir este limite, o sistema distribuído deve então atualizar todas as réplicas dos dados

UNIDADE DE CONSISTÊNCIA

Conit (do inglês *consistency unit*)

- Representa a unidade sobre a qual consistência defe ser aferida

No nosso caso, a consistência normalmente é um único dado

- Uma entrada em um banco de dados
- Um arquivo de código-fonte
- Uma atualização de um repositório

CONSISTÊNCIA SEQUENCIAL

O resultado de qualquer execução é a mesma se as operações de todos os processos sob o conjunto de dados são executadas em uma ordem sequencial

Sequencial

P1:	W(x)a			
P2:		W(x)b		
P3:			R(x)b	R(x)a
P4:			R(x)b	R(x)a

Não sequencial

P1:	W(x)a			
P2:		W(x)b		
P3:			R(x)b	R(x)a
P4:			R(x)a	R(x)b

CONSISTÊNCIA CASUAL

Operações de escrita concorrentes podem ser vistas em diferentes ordens por cada processo

- Relaxamento da consistência sequencial

Considere $W(x)b$ e $W(x)c$ como concorrentes

P1:	W(x)a										W(x)c				
P2:				R(x)a			W(x)b								
P3:					R(x)a								R(x)c		R(x)b
P4:			R(x)a											R(x)b	R(x)c

SESSÃO CRÍTICA E CONSISTÊNCIA

Usa operações de *lock* ($L(x)$) e *unlock* ($U(x)$)

P1:	L(x)	W(x)a	L(y)	W(y)b	U(x)	U(y)																	
P2:																	L(x)	R(x)a				R(y)	NIL
P3:																			L(y)	R(y)b			