Sistemas Distribuídos e Tolerância a Falhas

Mestrado de Engenharia Informática

2013/2014

Docente: Prof. Drª. Paula Prata

A DISTRIBUTED PROTOCOL FOR ENSURING REPLICATED DATABASE CONSISTENCY IN MOBILE COMPUTING ENVIRONMENTS

Carlos Augusto E8530

Micael Costa M6297

20 maio de 2014

UNIVERSIDADE DA BEIRA INTERIOR

Covilhã | Portugal

Agenda

- Introdução
- Trabalho Relacionado
- Replicação em base de dados
- Avaliação
- Conclusões
- Referências

Introdução

- Bases de dados móveis são constituídos por vários servidores e clientes interligados através de redes sem fios
- Para garantir a consistência dos dados em dispositivos móveis vários protocolos de controlo de replicação têm sido propostos
- A maioria revela várias limitações principalmente na troca de mensagem, inundado a rede

Introdução

- O artigo aborda e referência protocolos já existentes
- Apresenta um protocolo que garante a consistência dos dados na replicação das bases de dados para dispositivos móveis
- Aborda um exemplo de funcionamento

Introdução

- A abordagem proposta é completamente distribuída
- Usa um esquema de replicação read-any/write-any
- Aumenta a disponibilidade dos dados
- Reduz o número de mensagens trocadas entre servidores replicados, além disso, permite escolher o nível de isolamento da transação
- Resultados experimentais mostram a potencial eficiência da abordagem proposta

Introdução

- Um ambiente de computação móvel é caracterizado por possuir vários dispositivos distribuídos de forma dinâmica, sem localização fixa, ligado por meio de uma rede sem fios, em que, em geral, os dispositivos móveis podem ter conectividade intermitente à rede sem fio

- A ideia básica por trás de replicação das bases de dados é armazenar múltiplas cópias de dados (réplicas) em diferentes servidores (servidores replicados) distribuídos sobre a rede

Introdução

- **eventual consistency**

assegura que todas as copias de um determinado item dos dados X convergem para o mesmo estado final

- **one-copy serializability** (1SR)

assegura que a execução simultânea de um conjunto T de transações em J réplicas de uma base de dados móvel replicada deve ser equivalente a uma execução serial do conjunto T em relação à mesma base de dados sem a replicação (one-copy database)

Trabalho Relacionado

- Bayou
- Golding, Long
- Gifford
- Deno

Trabalho Relacionado

- No sistema Bayou, os dados são replicados em todos os servidores da rede e as aplicações clientes interagem com os servidores através da API Bayou. Esta API suporta duas operações básicas: leitura e escrita

- Uma vez que uma escrita é aceite, o cliente não tem que esperar pela sua propagação, o modelo de replicação é de consistência fraca, que permite apenas a garantia de uma consistência mínima

Trabalho Relacionado

- o sistema armazena em cada servidor Bayou um log ordenado de todas as escritas realizadas naquele servidor

- caso seja detectado um conflito, as operações são desfeitas e executadas novamente em outra ordem (é uma das grandes desvantagens)

- Bayou leva em consideração a semântica da aplicação para resolver os conflitos

Trabalho Relacionado

- O protocolo proposto em Golding Long implementa a replicação de dados como um grupo de réplicas que se comunicam através do protocolo de comunicação de grupo, que provê um serviço multicast que envia mensagem de uma réplica a todas as outras réplicas no grupo
- O fato de provocar a divergência temporária entre as réplicas e a necessidade da presença de um nó coordenador para que as funcionalidades do protocolo possam ser eficientes, o que caracteriza um ponto único de falha e pode colocar em risco toda a manutenção da consistência das réplicas

Trabalho Relacionado

- A proposta de Gifford foi projetada para sistemas em que o item de dado replicado é distribuído em múltiplos servidores e acedido por um cliente único

- É um algoritmo para manutenção da consistência de dados replicados, pessimista e centralizado, onde cada réplica i de um item dado recebe um determinado número de votos Vi.

- As réplicas possuem número de versão para permitir que clientes possam determinar quais estão atualizados

Trabalho Relacionado

- Uma das principais desvantagens do protocolo de Gifford é o fato do algoritmo necessitar de grande quantidade de troca de mensagens para resolver conflitos

- O uso deste cliente centralizado simplifica o algoritmo e sua manutenção, porém não o classifica como adequado para ambientes móveis, onde são encontrados diversos clientes que podem sair e entrar na rede a qualquer momento e possuem conectividade intermitente

Trabalho Relacionado

- O Deno é descrito como um algoritmo epidêmico baseado em votação ponderada, é um sistema de replicação de objetos para uso em ambientes móveis ou em ligações fracas

- A votação poderá ocorrer sem que seja necessária a participação de todos os membros da rede na votação, pois neste modelo as réplicas não têm conhecimento da quantidade de eleitores

Trabalho Relacionado

- O protocolo utiliza uma combinação de votação com pesos fixos e fluxo de informação epidêmico para permitir que atualizações sejam realizadas em ambientes de fraca ligação

- Mas, o protocolo poderá ser impróprio para ambientes que exigem alta conectividade

- Nos protocolos baseados em votação, a quantidade de mensagens trocadas na rede para a realização do processo de eleição provoca sobrecarga no tráfego da rede

Replicação em base de dados

- A utilização de mecanismos de bloqueio é proibitivo em bases de dados móveis

- Para garantir a consistência dos dados em bases de dados móveis replicadas esta abordagem é baseada na monitorização e gestão dinâmica de um grafo de conflito acíclico

Replicação em base de dados

- Esta abordagem chamada de temporal serialization graph testing (TSGT),
- explora informações temporais w.r.t. no momento em que uma operação de transação móvel (leitura ou escrever) é executada numa determinada base de dados.
- Além disso, o TSGT utiliza as definições de nível de isolamento generalizadas, para construir limites utilizados para representar diferentes tipos de conflitos
- Em seguida, apresentamos uma definição formal para grafo de serialização temporal (TSG).

Replicação em base de dados

- **S** – horário de transações **T={T1, T1, …, Tn}**
- **OP(Ti)**-conjunto de operações pertencentes a uma transação **Ti**
- **<s** - relação de precedência entre as operações num horário **S**
- **TSG** – Temporal Serialization Graph
- **TSG(S)=(N, E)**
- **N**- transações em **T , N = T**
- **E**- representa as seguintes formas

Replicação em base de dados

(i) $T_i \xrightarrow{RW-item} T_j$, *iff* $T_i, T_j \in N$ *and there are two operations* $p \in OP(T_i)$, $q \in OP(T_j)$, *where* p *conflicts with* q, $p <_S q$ *and* p *is a read operation and* q *is a write operation;*

(ii) $T_i \xrightarrow{RW} T_j$, *iff* $T_i, T_j \in N$ *and there are two operations* $p \in OP(T_i)$, $q \in OP(T_j)$, *where* p *conflicts with* q, $p <_S q$ *and* p *is a predicate-based read operation and* q *is a write operation;*

(iii) $T_i \xrightarrow{WR} T_j$, *iff* $T_i, T_j \in N$ *and there are two operations* $p \in OP(T_i)$, $q \in OP(T_j)$, *where* p *conflicts with* q, $p <_S q$ *and* p *is a write operation and* q *is a read operation;*

(iv) $T_i \xrightarrow{WW} T_j$, *iff* $T_i, T_j \in N$ *and there are two operations* $p \in OP(T_i)$, $q \in OP(T_j)$, *where* p *conflicts with* q, $p <_S q$ *and* p,q *is a write operations.*

Replicação em base de dados

- neste protocolo
- cada item de dados em **x** DB_{Ri} está associado com um timestamp **C (x)**
  - onde 0 < **i** <= **n** e **n** é o número dos servidores replicados
- cada operação **p ∈ {r, w}, p ∈ OP(T),** executada num objeto da base de dados **x ∈** DB_{Rj}
  - tem um timestamp **C ∈ (p (x)),** que corresponde ao timestamp atual de **x**, isto é, **C (p (x)) = C (x).**

Replicação em base de dados

- Descrição de como os timestamps são gerados e geridos pelo protocolo proposto

  - *A timestamp consists of an ordered pair (z, y). The first part (z) is called version, while the second part (y) is called subversion;*
  - *Initially $C(x) = (0, 0)$ $\forall x \in DB$, in all copy DB_{R_j}, where $0 < j \leq n$;*
  - *The version component of $C(x)$ (i.e., z) is incremented for each commit of any transaction T_i which has executed a write operation on x (new version of x), $x \in DB_{R_j}, 0 < j \leq n$;*
  - *The sub-version component of $C(x)$ (i.e., y) is set to 0, i.e., $C(x) = (v, 0)$, with $v > 0$, for each commit of a any transaction T_i which has executed a write operation on x, $x \in DB_{R_j}, 0 < j \leq n$;*
  - *The sub-version component of $C(x)$ (i.e., y) is incremented for each write operation of any active transaction T_i on x, $x \in DB_{R_j}, 0 < j \leq n$. Note that T_i may be an uncommitted transaction.* $\diamond$

Replicação em base de dados

- Para comparar dois timestamps C (qj (x)) e C (pi (x))
  - em que C (qj (x)) = (a, b) e C (pi (x)) = (c, d)
- As seguintes regras devem ser aplicadas:

(i) If $a > c$, *then* $C(q_j(x)) > C(p_i(x))$. *By the same rule, if* $a < c$ *then* $C(q_j(x)) < C(p_i(x))$;

(ii) If $a = c$, *the subversion value is verified. Thus,*

(a) If $b > d$, *then* $C(q_j(x)) > C(p_i(x))$. *By the same rule, if* $b < d$ *then* $C(q_j(x)) < C(p_i(x))$.

(b) If $b = d$, *then* $C(q_j(x)) = C(p_i(x))$. ◊

Replicação em base de dados

- Step 1

  Quando o protocolo de controlo de replica arranca, o grafo de serialização temporal (def1) é criado como um grafo vazio.

- Step 2

  Assim que o protocolo recebe a primeira operação de uma nova operação de Ti, um novo nó, que representa essa transação é inserido no TSG.

Replicação em base de dados

- Step 3

  Para cada operação de leitura que chega ri (x) ∈ OP (Ti), C (RI (x)) = C (x).

- Step 4

  Para cada operação de escrita que chega wi (x) ∈ OP (Ti), o componente de sub-versão de C (x), na DBRi, (isto é, y) é incrementado e C (Wi (x)) = C (x)

Replicação em base de dados

- Step 5

  Quando DBRi recebe uma operação de confirmação de uma transação Ti (juntamente com o número de operações de Ti), que verifica se todas as operações de Ti já foram recebidos. Neste caso, o protocolo executa o algoritmo apresentado na Figura 1

Replicação em base de dados

For each operation $p_i(x) \in T_i$ do:
  GraphUpdate($p_i(x)$), in DB_{R_i}
End For
For each active replicated server DB_{R_j}, where $1 \leq j \leq n, j \neq i$ and n represents the number of replicated servers do:
  For each data item x, read or written by T_i, do
    DB_{R_i} requests for DB_{R_j} the executed operations on x together with the operations timestamps
      For each received operation $p_j(x) \in T_j$ do:
        GraphUpdate($p_j(x)$), in DB_{R_i}
      End For
    End For
  End For
End For
Verifies if the graph has a cycle
If the graph has a cycle
  Abort T_i
  Informs the abort to client C_i
  Update the serialization graphs
  Update the timestamps of the data items updated by T_i
  Informs the abort of T_i to others replicated servers
Else
  Commits T_i
  Informs the commit to client C_i
  Update the timestamps of the data items updated by T_i
  Informs the commit of T_i to others replicated servers
  Sends the new values of the data items updated by T_i together with the new timestamps to others replicated servers
End If

Replicação em base de dados

- A fim de verificar a existência de qualquer ciclo no TSG, o descrito na Figura 1 aplica as seguintes regras:

- Se o nível de isolamento escolhido é READ UNCOMMITTED (PL-1) o protocolo verifica a existência de um ciclo envolvendo somente arestas com o nome “WW”.

- Se o nível de isolamento escolhido é READ COMMITTED (PL-2) o mecanismo verifica a existência de um ciclo envolvendo arestas com os nomes “WW” e/ou “WR”.

Replicação em base de dados

- Se o nível de isolamento escolhido é REPEATABLE READ (PL-2.99), o protocolo verifica a existência de um ciclo envolvendo arestas com os nomes “WW”, “WR” e/ou “RW-item”.

- Se o nível de isolamento é SERIALIZABLE (PL-3), o protocolo verifica a existência de um ciclo envolvendo arestas com os nomes “WW”, “WR”, “RW-item” e/ou “RW”.

Replicação em base de dados

- Quando um ciclo é detetado, Ti é cancelada
- O protocolo então envia a mensagem para o cliente que submeteu Ti informando sobre o cancelamento de Ti
- Se não há um ciclo, a operação de efetivação será executada

Level	Phenomena disallowed	Informal Description (T_i can commit only if:)
PL-1	G0	T_i's writes are completely isolated from the writes of other transactions
PL-2	G1	T_i has only read the updates of transactions that have committed by the time T_i commits (along with PL-1 guarantees)
PL-2.99	G1, G2-item	T_i is completely isolated from others transactions with respect to data items and has PL-2 guarantees for predicated-based reads
PL-3	G1, G2	T_i is completely isolated from other transactions, i.e., all operations of T_i are before or after all operations of any other transaction

Replicação em base de dados

- Depois que a transação Ti é efetivada, a versão do timestamp (ver Definição 2) de todos os itens atualizados por Ti é incrementada como a seguir:
  - para todo x, onde x foi atualizado por Ti, faça

    C(x) = (z + 1; 0).
- Assim, os novos valores dos itens de dados atualizados por Ti com os novos timestamps são propagados a todos os servidores replicados presentes na rede.

Replicação em base de dados

- Step 6

  Quando o protocolo recebe uma solicitação de cancelamento de uma transação Ti, ele desfaz os efeitos das operações de Ti, remove as arestas associadas com esta transação, envia a confirmação de cancelamento ao cliente Ci e informa o cancelamento de Ti aos outros servidores replicados, a fim destes servidores atualizem os seus grafos de serialização e os timestamps dos itens de dados atualizados por Ti, nas suas cópias locais.

Avaliação

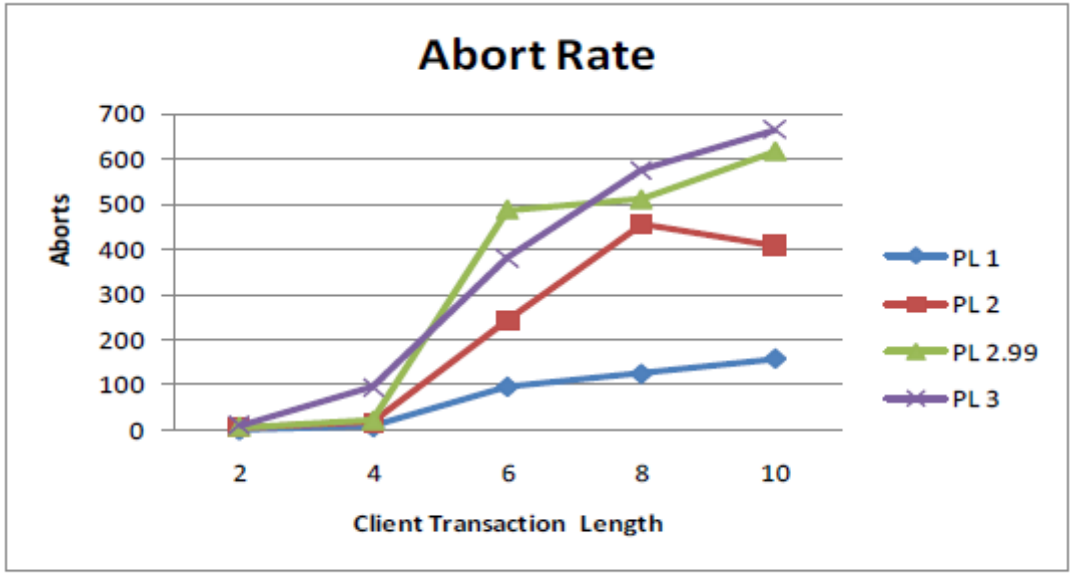

Figure 6: Abort Rate.

- para os níveis menos restritivos, para transações com grande número de operações, a tendência do comportamento do protocolo é evitar aborts

Avaliação

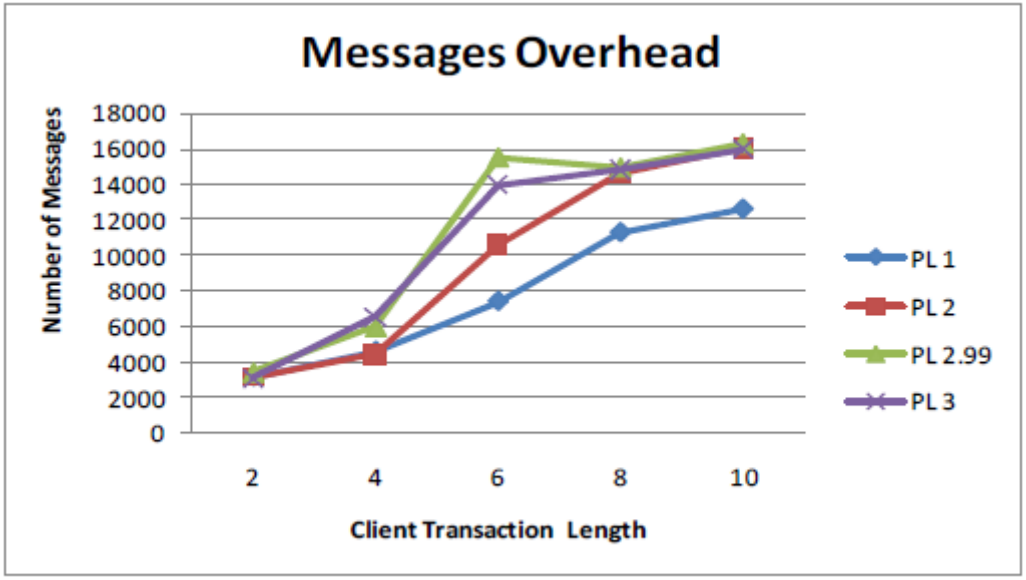

Figure 7: Messages Overhead.

- a sobrecarga de troca de mensagens apresenta um comportamento que tende a ser linear independente do tamanho da transação

Conclusões

- Neste artigo é apresentado um protocolo completamente distribuído para controlo e consistência de dados numa base de dados replicada em ambientes móveis

- Através da utilização da replicação de dados e do controlo da consistência de forma distribuída, o protocolo proposto procura aumentar a disponibilidade, a confiabilidade e a escalabilidade dos dados, bem como, o número das transações (número de transações executadas por unidade de tempo)

Conclusões

- o uso da replicação implica na utilização do esquema multi-maste (read-any/write-any).
- permite a escrita e a leitura de dados em qualquer um dos nós presentes na rede e a atualização dos dados mesmo quando alguns servidores replicados estão fora da rede
- utiliza a replicação síncrona, permite aplicar as alterações realizadas nas réplicas disponíveis de forma imediata
- reduz o número de mensagens trocadas entre os servidores (o que minimiza os custos de comunicação)

Referências

- Costa, Alex. Monteiro, Jose Maria. Brayner, Angelo. *A Distributed Protocol for Ensuring Replicated Database Consistency in Mobile Computing Environments,* Proceedings of the ACM Symposium on Applied Computing 2010, Pages 1688-1693

OBRIGADO