Análise do Artigo EtherH - A Hybrid Index to Support Blockchain Data Query

1. Introdução

O artigo "EtherH: A Hybrid Index to Support Blockchain Data Query" propõe uma solução para o problema de consulta de dados em blockchain, utilizando o Ethereum como plataforma experimental. A solução, chamada EtherH, é um índice híbrido baseado em B-tree e Skip-list, projetado para suportar consultas de valor único (Single-V query) e consultas de intervalo (Range query) em dados do blockchain Ethereum.

1.1. Problema Abordado

O artigo destaca que, embora o blockchain seja uma tecnologia disruptiva com características como descentralização e imutabilidade, o armazenamento subjacente do blockchain oferece suporte limitado para consultas de dados. Isso restringe a utilidade da tecnologia blockchain, especialmente em cenários que exigem acesso eficiente a grandes volumes de dados. O Ethereum, por exemplo, armazena dados em LevelDB, que suporta apenas inserção e consulta baseada em chave-valor, não sendo eficiente para consultas complexas.

1.2. Solução Proposta: EtherH

O EtherH é um índice híbrido que extrai dados de blocos do Ethereum e os organiza em uma estrutura que combina B-tree e Skip-list. A B-tree é utilizada por sua eficiência em operações de busca, inserção e exclusão, sendo uma estrutura de árvore de busca autobalanceada. A Skip-list é empregada para classificar e mesclar dados de blocos em lotes no índice da B-tree, oferecendo desempenho de busca comparável a árvores AVL e árvores rubro-negras.

1.2.1. Estrutura do EtherH

O EtherH é composto por:

- **B-tree**: Uma árvore de busca multi-caminho auto-balanceada. O artigo ressalta que, em comparação com a B+ tree, todos os nós da B-tree contêm o domínio de dados, o que significa que a consulta não precisa percorrer até o nó folha para retornar o dado, tornando a consulta única mais rápida. Além disso, não há sobrecarga de ponteiro extra de nó folha.
- **Skip-list**: Utilizada para organizar e inserir dados em lotes na B-tree. Ela adiciona um índice multi-nível a uma lista encadeada ordenada, melhorando o desempenho de busca.

1.2.2. Inserção de Dados no Blockchain

O processo de inserção de dados no EtherH (especificamente na B-tree) envolve:

1. Carregar o nó raiz da B-tree. Se não existir, um novo nó folha é criado e o dado é inserido.
2. Encontrar o nó folha apropriado e inserir a chave e o dado. Se chaves iguais forem encontradas, os dados são mesclados.
3. Verificar se o nó atual precisa ser dividido (se o número de chaves for maior que M-1, onde M é a ordem da B-tree).
4. Se sim, o nó é dividido e a chave do meio é inserida no nó pai, apontando para as duas partes do nó original. O processo de divisão e inserção no nó pai é repetido recursivamente até que nenhuma divisão seja necessária.

1.2.3. Consulta de Dados no Blockchain

O EtherH suporta dois tipos de consulta:

- **Consulta de Valor Único (Single-V query)**: A busca é realizada na B-tree, comparando a chave a ser buscada com as chaves nos nós. O processo se repete até que o registro correspondente seja encontrado ou até que o nó folha seja alcançado sem sucesso.
- **Consulta de Intervalo (Range query)**: Considerada uma generalização da consulta de valor único, retorna todos os dados correspondentes às chaves dentro de um intervalo fechado.

1.3. Implementação e Avaliação

Os autores implementaram o EtherH no cliente Ethereum Geth v1.9 e realizaram experimentos para avaliar o consumo de armazenamento, o desempenho de inserção e o desempenho de consulta. Os resultados experimentais indicam que o EtherH apresenta bom desempenho tanto na inserção quanto na consulta de dados, mesmo com grandes volumes de blocos.

2. Arquitetura Blockchain e B-trees

2.1. Fundamentos da Arquitetura Blockchain

Um blockchain é uma cadeia de blocos interligados por ponteiros de hash criptografados. Cada bloco contém um cabeçalho e um corpo.

- O **cabeçalho do bloco** contém informações como o valor hash do bloco anterior, o endereço do minerador, a raiz da árvore de estado da conta (StateRoot), a raiz da árvore de recibos (ReceiptsRoot) e a raiz da árvore de transações (TransRoot).
- O **corpo do bloco** contém a lista de transações e a lista de cabeçalhos de blocos órfãos (UncleHeaderList).

O Ethereum utiliza três árvores Merkle Patricia modificadas (Merkle Patricia Tree - MPT) para manter:

- A árvore de transações
- A árvore de recibos
- A árvore de status
  Essas estruturas garantem integridade e verificabilidade dos dados no blockchain.

2.2. B-trees em Bancos de Dados Blockchain

Embora o blockchain seja inerentemente sequencial, a necessidade de consultas eficientes levou à exploração de estruturas como B-trees. Ao construir índices baseados em B-trees, é possível:

- Melhorar o desempenho de consulta (busca por valor único e intervalo)
- Suportar consultas complexas (filtros e agregações)
- Otimizar o uso de armazenamento (redução de varreduras completas)

2.3. B-Merkle Tree

A B-Merkle Tree (BMT) combina propriedades de autobalanceamento das B-trees com verificação de integridade das árvores Merkle. Funcionalidades incluem:

- Operações básicas: Inserção, Obtenção, Remoção e Intervalo
- Provas de tamanho constante para integridade dos dados
- Adaptação a bancos de dados descentralizados autenticados