Simulador de Sistemas de Arquivos

Trabalho Prático 1

Aluno: Tárcio Henrique Rodrigues Santos
Professor: Douglas Donizeti de Castilho Braz

RESUMO:

Neste responderei e utilizarei como guia o questionário da página 16 e como isso foi alterando com o tempo e minha experiência com o código.

RESPONDENDO OS TÓPICOS :

A) Detalhes das Estruturas utilizadas para armazenamento do Sistema de Arquivos:

Eu utilizo da classe HardDisk para armazenar os dados do sistema de arquivos. Ela usa uma matriz de valores booleanos para representar cada bit no disco rígido. Os valores true indicam que um bit está definido (ligado), enquanto os valores false indicam que um bit está desligado. O tamanho da matriz é calculado com base no tamanho da memória secundária especificado durante a inicialização da classe(this.numeroDeBits = tamanhoDaMemoriaSecundaria * 8 * 1024 * 1024;). Há métodos para definir e obter o valor de bits individuais no disco rígido, bem como para inicializar todo o disco rígido definindo todos os bits como desligados. Essa estrutura de dados simula um disco rígido simplificado para armazenar dados do sistema de arquivos.

EXPERIENCIA:

Boa parte do código foi discutida e apresentada tanto pelo professor e também em conversas com colegas(troca de opiniões e ideias), em sala definimos os bits e tamanhos que utilizamos para tudo, uma parte traumatizante que sem a ajuda e colaboração do professor não estaria ligando as funções já existentes no codigo.

B)Dificuldade no desenvolvimento do trabalho. Teoria e/ou prática:

Será relatado em cada trecho do trabalho as dificuldades encontradas, mas cabe discorrer sobre algumas gerais e principais, o código primário já tinha todas as funções implementadas e testadas essa versão destruiu e reconstruiu o código completamente tendo que ser totalmente remendado.

C)Complexidade das operações das chamadas de sistema implementadas (utilizando notação O):

Batch é O(n), onde n é o número de comandos encontrados no arquivo ja que as chamadas encontradas no for são somente executadas pelas suas quantidades e o “peso” encontrado é mais na função chamada em decorrência do batch.

dump a complexidade dessa função no meu código depende mais da chamada de “visitaTodosOsFilhos” que é O(n), onde n é a quantidade de diretórios existentes.

cat a complexidade é dominada pela busca no sistema de arquivos, que é O(n), onde n é o número de diretórios no caminho ou o número total de arquivos no sistema de arquivos. As outras operações são de complexidade O(1) ou têm menor impacto.

cd Mesma lógica da cat, quando usamos sem passar um caminho ele apenas lista o que esta ali apenas realizando uma operação O(1), mas quando passo um caminho absoluto navego entre os diretórios O(n).

chmod é dominada pela busca no sistema de arquivos, que é O(n), onde n é o número total de diretórios no sistema de arquivos ou o número de diretórios no caminho. As outras operações têm complexidade menor, principalmente O(k), onde k é o tamanho da string de permissões.

cp o maior custo é encontrado no HD e vai ser calculado pelo tamanho do arquivo ou dos arquivos, mas o cp em si so navega pelos diretórios e os marca, ou seja O(n).

createfile segue o mesmo padrão das outras classes O(n) para buscar e O(1) para realizar a operação.

info é O(1) já que apenas realiza a operação.

ls é dominada pela complexidade da função “listarConteudo”, que é linear em relação ao número de arquivos e diretórios a serem listados. Portanto, a complexidade de tempo da função ls também é O(n), onde 'n' é o número de arquivos e diretórios a serem listados.

mkdir possue a complexidade O(n), mas depende da função “verificaOrigem” que tambem é (O(n)) ,mas a função encontra-se optimizada.

mv exatamente a mesma coisa inclusive a dependência ou seja O(n).

rm encontra uma complexidade de O(n) porem remover também é custoso e depende dessa recursão e da entrada do usuaria O(k).

rmdir a eficiência da operação depende do número de diretórios no sistema e do número de diretórios no diretório de origem. se no local O(1), se tiver qeu nevegar pagara o preço da navegação, ou seja O(n).

D)Sugestões, considerando as estruturas utilizadas, para a atualização que especifica um sistema de arquivos multiusuário:
Não entendi realmente a pergunta, mas a autenticação de usuário e alteração do diretório raiz para cada usuário pode resultar em uma resposta satisfatória.

E)Sugestões, considerando as estruturas utilizadas, para a atualização que especifica um sistema de arquivos com acesso concorrente:
Novamente encontrei dificuldade em entender a questão, mas a realização de testes constantes e calibrando o serviço ao público alvo além de atenção para controle de conflitos e Exceções podem ser relevantes para este tópico.

F)Comentários para as perguntas:
1) O sistema de arquivos implementado é eficiente?
Levando em consideração a simulação única e de único propósito acadêmico para avaliação pode-se dizer que é bastante eficiente pelo preço de cada operação e funcionalidades apresentadas e entregues.

2)O sistema de arquivos implementado é confiável?
Negativo, todos os dados são facilmente manipulados por qualquer usuário.

3)Quais testes foram executados para ‘garantir’ a confiabilidade do software?

Apenas o teste de funcionamento de cada classe e tópicos pedidos ou seja a funcionalidade de todas as funções.

4)Você enxerga diferentes implementações de sistemas de arquivos para diferentes hardwares? Exemplifique.

Sim, pois cada usuário ou público alvo vai utilizar de forma ímpar. Além das inúmeras nuances nas tentativas de otimização ou utilização de espaços, somados às permissões que cada um provê para seu sistema de arquivos.

REFERENCIAS:

https://www.youtube.com/@canalvidadedev6874
https://www.guj.com.br/t/prionace-gerenciador-de-arquivos-em-java/255448
https://languagetool.org/pt-BR

Consulta: Sinto totalmente à vontade quanto a ser chamado para explicar meu código, possuo algumas anotações a serem consultadas no momento.