Universidade Federal do Rio Grande do Sul - UFRGS
Instituto de Informática - INF

INF01142 - Sistemas Operacionais I - 2014/2

Implementação de um Sistema de Arquivos

(T2FS - Task 2: File System)

Prof. Sérgio Luis Cechin

Discentes autores:
Jonas Ribeiro Flores - 171607
Lucas Herbert Jones - 124631

04 de novembro de 2014

1 | Introdução

Esse relatório é parte de uma atividade acadêmica que foi desenvolvido juntamente com um trabalho prático para a disciplina de Sistemas Operacionais I. Tem como objetivo responder questões relativas à implementação de uma biblioteca que simula um sistema de arquivos.

2 | Questionário

2.1 | Sem alterar a quantidade de ponteiros de alocação indexada, quais outros fatores influenciam no maior tamanho de arquivo T2FS possível? Como esses fatores influenciam nesse tamanho?

Mantendo o mesmo número de blocos, mas alterando o tamanho do bloco, o tamanho do arquivo poderá ser alterado, dado a sobra de bytes não utilizada em um bloco maior é maior.

2.2 | Supondo que você desejasse melhorar o T2FS, permitindo a criação de vínculos estritos (hardlinks). Que alterações seriam necessárias no T2FS? Há necessidade da criação de novas funções? Se sim, quais? Se não, porque não.

A estrutura precisaria ser ampliada para comportar mais de um tipo de link. Seria necessário criar uma estrutura semelhante a de um i-node. Além disso, seria necessário criar mais um campo na estrutura do descritor de arquivo, para que comporta-se um contador que indica quandos ponteiros referenciam esse arquivo. Quando um hardlink é feito para esse arquivo o contador e incrementado e quando se desfaz um hardlink, o contador é decrementado. Esse arquivo só pode ser deletado caso esse contador seja igual a zero, ou seja sem ponteiros fazendo referencia a ele. Não há necessidade de novas funções para atender essa nova especificação.

2.3 | As estruturas de controle do T2FS contêm informações que permitem verificar a consistência de alguns de seus elementos. Isso é possível graças a um nível de

redundância de informação (por exemplo, no registro de arquivo, nas entradas do diretório, o número total de blocos usados por um arquivo e o tamanho do arquivo – em bytes – permitem uma verificação). Identifique quais outros elementos são redundantes e discuta como seria possível usar essa redundância para aumentar a confiabilidade do T2FS.

Os diretórios tem um campo que indica o número de bytes de um arquivo, permite uma verificação, uma vez que tem informações que podem ser verificadas através dele. Dado o tamanho da estrutura do t2fs de 64 bytes, poderíamos representar a memória ocupada por um diretório pelo número de arquivos e pelo número de blocos ocupados. Dessa forma é possivel verificar a consistência do diretório com registros bem formatados.

2.4 | Como você implementou a gerência do contador de posição (current pointer) usado pela função seek2?

A função seek2 recebe o handle do arquivo aberto e um valor de offset. O handle corresponde a um identificador único de cada arquivo aberto no momento. O valor de offset corresponde ao deslocamento em bytes do início do arquivo.

A implementação é muito simples. Inicialmente é feita uma pesquisa dentro da tabela de arquivos abertos por processos TAAP utilizando o handle do arquivo. Essa pesquisa retorna o descritor do arquivo, o qual já se encontra em memória. Além disso, o endereço do currentPointer desse arquivo é apontado por um ponteiro para que seja possível modificálo.

Foi necessário verificar se o offset informado é igual a 1. Nesse caso o currentPointer do arquivo recebe o valor bytesFileSize da estrutura t2fs_record desse arquivo: corresponde a apontar para o final do arquivo. Caso currentPointer somado ao offset ultrapasse o tamanho do arquivo (bytesFileSize) é currentPointer mantém o mesmo valor e é retornado um indicativo de erro (1). Para qualquer outro caso, o currentPointer recebe o valor informado offset.

2.5 | A escrita em um arquivo (realizada pela função write2) requer uma sequência de leituras e escritas de blocos de dados e de blocos de controle. Qual é a sequência usada por essa função? Se essa sequência for interrompida (por falta de energia, por exemplo) entre duas operações de escrita de bloco, qual será o efeito na consistência

dos dados no disco? É possível projetar uma sequência de escritas no disco que minimize a eventual perda de dados?

A cada escrita no arquivo o bloco apontado pelo currentPointer é carregado em memória, o seu conteúdo é alterado e logo depois é novamente salvo no disco. Se ocorresse falta de energia o conteúdo a ser salvo nos blocos seguintes não seria salvo, afetando a consistência do sistema de arquivos. Para minimizar essa perda de dados todos os blocos alterados poderiam ser carregados em conjunto, alterados em memória e depois salvos em conjunto novamente.

2.6 | Algumas estruturas gravadas no disco são mais facilmente manipuláveis se estiverem na memória principal (como se fosse uma cache). Por outro lado, isso aumenta a possibilidade de perda de dados, pois as informações existentes nessa cache e que não foram escritas no disco, podem ser perdidas, caso ocorra alguma interrupção de operação do sistema. Quais informações do disco você está mantendo (e gerenciando) na memória principal e porque você as escolheu? Qual a política que você usou para decidir quando escrevê-las no disco?

Estão sendo mantidas em memória as informações do superbloco, do bitmap de blocos livres e os descritores dos arquivos abertos (t2fs_record). Cada vez que alguma dessas estruturas é modificada ela é salva em disco.

2.7 | Todas as funções implementadas funcionam corretamente? Relate, para cada uma das funções desenvolvidas, como elas foram testadas?

Para todas as funções os testes foram realizados utilizando arquivos do tipo teste1.c e testes unitários sobre cada uma das funções foram realizados. Não conseguimos implementar bem o readdir2 e o getcwd2. Além de alguns problemas com alguns caracteres estranhos aparecendo no nome dos arquivos quado tentávamos acessar por *relative path.*

2.8 | Relate as suas maiores dificuldades no desenvolvimento deste trabalho e como elas foram contornadas.

As maiores dificuldades foram a linguagem C e conciliar o pouco tempo para executar o trabalho com as outras tarefas da faculdade. Essas dificuldades foram contornadas através de noites mal dormidas e sacrifícios pessoais.