Sistemas Operacionais - Trabalho Prático II

Sistema de arquivos FAT e um simples *shell*

O segundo trabalho prático da disciplina de Sistemas Operacionais consiste na implementação de um simulador de um sistema de arquivos simples baseado em tabela de alocação de 16 bits (FAT) e um *shell* usado para realizar operações sobre este sistema de arquivos.

O sistema de arquivos virtual

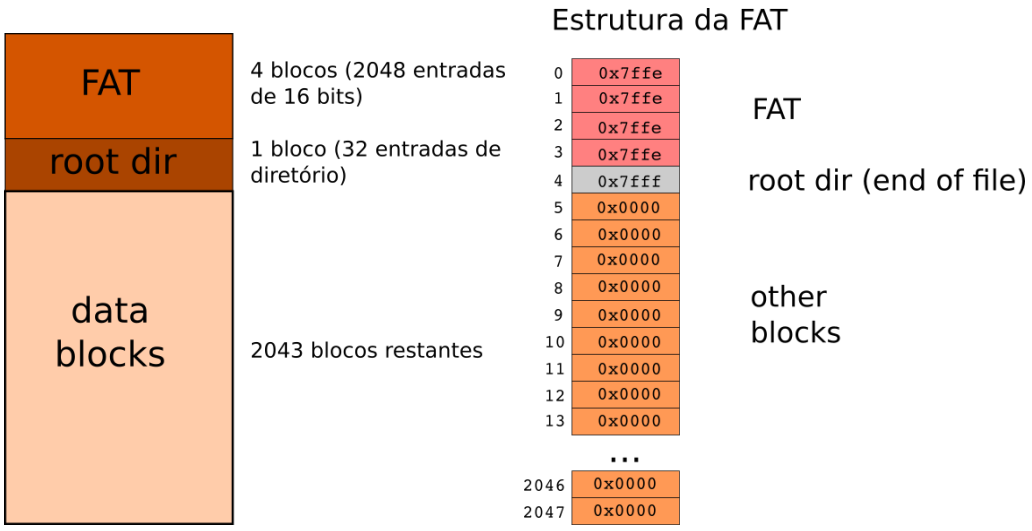
O sistema de arquivos deverá ser armazenado em uma partição virtual e suas estruturas de dados mantidas em um único arquivo nomeado *filesystem.dat*, que representará fisicamente a partição. O tamanho da partição virtual será determinado por:

- blocos de 1024 bytes;
- 2048 blocos;

Os primeiros quatro blocos do disco armazenarão uma tabela de alocação (FAT), que representa um mapa de todos os blocos da partição, incluindo a própria FAT, diretório raiz e os outros arquivos e diretórios, bem como a forma como esses blocos são encadeados para que arquivos que possuem mais de um bloco de tamanho (1024 bytes) possam ser reconstruídos. A FAT possui um tamanho que é determinado pelo número de blocos do disco (2048 blocos no total). Cada entrada na FAT possui um tamanho de 16 bits e representa um bloco.

O diretório raiz (*root*) estará localizado logo após a FAT e terá um tamanho de 1 bloco (assim como todos os outros diretórios ou subdiretórios). O diretório raiz possui um conjunto de 32 entradas de diretório, e cada entrada pode apontar para o primeiro bloco de um arquivo ou para um outro diretório, sendo o encadeameto de blocos adicionais realizado na FAT (ou marcado na FAT como último bloco). Após a FAT e o diretório *root*, encontra-se

a seção de dados contendo o restante dos blocos. Inicialmente, o disco deve ser preenchido com a FAT inicializada e o restante dos blocos zerados.



(a) (b)

Detalhes sobre o sistema de arquivos

O sistema de arquivos possui uma série de limitações, que foram determinadas com o intuito de simplificar a implementação do trabalho. A primeira limitação refere-se ao tamanho da FAT, onde é possível armazenar apenas 2048 entradas para blocos, o que limita o tamanho da partição virtual em 2MB. Lembre que a FAT é um "mapa" que representa a estrutura dos blocos da partição. A segunda limitação refere-se ao número de entradas de diretório em cada nível da árvore. Cada entrada ocupa 32 bytes, o que limita o número de entradas de diretório em 32, tanto no diretório raiz quanto em sub-diretórios (os diretórios possuem tamanho de um bloco e não podem ser aumentados).

Para manipular o sistema de arquivos, deve-se ler e escrever sempre utilizando a unidade *bloco*, independente de ser um diretório ou bloco de dados que representa uma porção de um arquivo. A FAT pode ser lida/gravada completamente no disco (para fins de simplificação). Sugere-se manter duas estruturas de dados em memória - FAT (4 blocos) e um bloco para entradas de diretório ou dados. Não esqueça de após manipular a FAT ou dados de atualizar o sistema de arquivos virtual. Lembre-se que o sistema precisa manter-se consistente, ao ponto de poder ser recuperado a qualquer instante.

- Informações sobre o valor das entradas na FAT de 16 bits:

```
0x0000                -> cluster livre
0x0001 - 0x7ffd       -> arquivo (ponteiro p/ proximo bloco)
0x7ffe                -> FAT
0x7fff                -> fim do arquivo
```

- Informações sobre a estrutura das entradas de diretório:

```
25 bytes          -> nome do arquivo
1 byte            -> atributo do arquivo
2 bytes           -> numero do primeiro bloco ocupado
4 bytes           -> tamanho do arquivo
```

Byte de atributo do arquivo - valor: 0x00 - entrada de diretório vazia, 0x01 - arquivo regular, 0x02 - diretório.

- Tipos e estruturas pré-definidas (usadas como referência)

```
/* entrada de diretorio, 32 bytes cada */
struct dir_entry_s {
    int8_t filename[25];
    int8_t attributes;
    int16_t first_block;
    int32_t size;
};

/* 4 blocos da tabela FAT, 2048 entradas de 16 bits = 4096 bytes*/
int16_t fat[2048];

/* diretorios (incluindo ROOT), com 32 entradas de diretorio */
struct dir_entry dir_block[32];

/* bloco de dados */
int8_t data_block[1024];
```

Depuração

Sugere-se utilizar a ferramenta de sistema *hexdump* (*hexdump -C filesystem.dat*) para realizar a depuração e visualização das estruturas de dados do sistema de arquivos virtual. Esta ferramenta faz parte do sistema Linux.

Detalhes sobre o shell

Um pequeno *shell* deve ser implementado para a manipulação do sistema de arquivos. Este shell deve oferecer recursos para a carga do sistema de arquivos e manipulação de diretórios e arquivos. Os seguintes comandos devem ser implementados no shell:

- init - inicializar o sistema de arquivos com as estruturas de dados, semelhante a formatar o sistema de arquivos virtual
- load - carregar o sistema de arquivos do disco
- ls [/caminho/diretorio] - listar diretório
- mkdir [/caminho/diretorio] - criar diretório
- create [/caminho/arquivo] - criar arquivo
- unlink [/caminho/arquivo] - excluir arquivo ou diretório (o diretório precisa estar vazio)
- write "string" [/caminho/arquivo] - escrever dados em um arquivo (sobrescrever dados)
- append "string" [/caminho/arquivo] - anexar dados em um arquivo
- read [/caminho/arquivo] - ler o conteúdo de um arquivo

Entrega

O trabalho deverá ser realizado em duplas ou trios. Qualquer linguagem de programação pode ser utilizada (preferencialmente C ou Java) para o desenvolvimento do trabalho, desde que as estruturas de dados que implementam o sistema de arquivos sejam manipuladas e armazenadas em disco de acordo com a especificação. A entrega do trabalho deverá ser realizada pelo moodle em um arquivo *.tar.gz* contendo a implementação, instruções de uso e um relatório, contendo o nome dos integrantes e apresentação da implementação exemplificando o uso de cada comando do shell no sistema de arquivos, juntamente com capturas de tela e saída da ferramenta hexdump.