FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

Zápočtová úloha z předmětu KIV/ZOS

Práce s pseudo FAT

24. ledna 2016

Lukáš Černý

1 Úvod

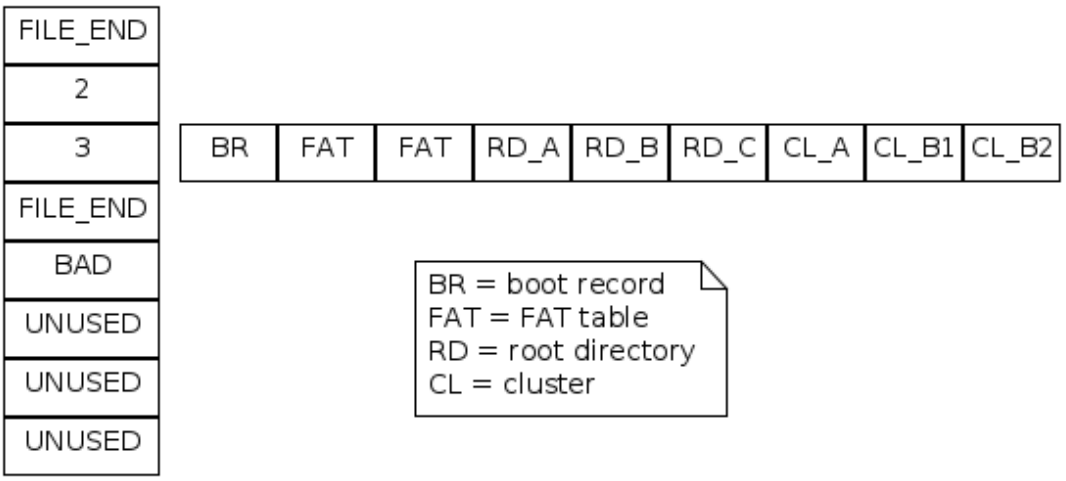
Zadání mé semestrální práce je zaměřeno na práci s pseudo souborovým systémem FAT. Zadání je rozděleno na dvě části. První zadání je kontrola, zda každý soubor má správnou délku. což znamená zkontrolovat, zda uváděná velikost opravdu odpovídá velikosti zapsané ve FAT. A druhá část je kontrola a oprava badblocků.

2 Teorie

FAT je souborový systém (FS), který se dříve využíval pro operační systém DOS. V dnešní době se již pro operační systémy nepoužívá a byl nahrazen modernějšími. Využití se však našlo u přenosných médií jako jsou flash disky nebo paměťové karty. Jelikož jde o velice rozšířený FS, je jeho podpora ve všech operačních systémech a ve většině přenosných zařízení.

Struktura fat je rozdělena do čtyř částí:

- boot record: Nachází se na začátku souborového systému. Obsahuje informace o velikostech klastrů, fat tabulky, kolik je aktuálně zabráno klastrů a další informace, které jsou důležité pro správné načtení dat.
- root directory: Každý soubor na disku obsahuje vlastní root directory. Jsou zde uvedeny informace k souboru - název, velikost, počáteční klastr, přístupová práva, typ (soubor, složka).
- fat table: FAT obsahuje dvě fat tabulky, kde jsou uloženy adresy klastrů souborů, volná místa a místa, kam neukládat data, jelikož mohou být klastry poškozené.
- cluster: V klastrech jsou fyzicky uložená data. Vyplňují zbytek kapacity disku.



Obrázek 1: Ukázka pseudo fat

Pro naše zadání semestrální práce byla verze FAT upravena (viz 1). Byly vytvořeny speciální značky, podle kterých jsem poznal, jaká data jsou v klastru uložena (viz níže).

- FAT_UNUSED: 65535
- FAT_FILE_END: 65534
- FAT_BAD_CLUSTER: 65533

3 Navržené řešení

Při kontrole délky souborů zároveň kontroluji špatné klastry (spojil jsem obě zadání dohromady). Zpracování běží paralelně ve vláknech, kde počet vláken je zadán při spuštění programu jako parametr (viz obsluha programu).

Data pro paralelní zpracování jsou rozdělena tak, že každé vlákno zpracovává jeden soubor. Tato oblast je kritická sekce pro zpracování, kdy musím zajistit, aby každé vlákno zpracovávalo jiný soubor (tj. jeden soubor nesní být zpracováván více než jedním vláknem). Tuto kritickou sekci jsem vyřešil použitím mutexu. Vydání souboru je možné pouze jednomu vláknu a ostatní čekají. Pokud už nejsou k dispozici další data ke zpracování, vlákna se po postupně ukončí.

Jakmile vlákno dostane data, začne proces kontroly. V cyklu, který běží přes bloky ve FAT tabulce, se kontrouluje délka souboru a zároveň ke každému bloku dat se najde příslušný klastr, který se zkontrouje. Pokud je klastr označen jako špatný, je zahájena jeho oprava. Při opravě klastru opět nastává kritická sekce v paralelním zpracování. Použil jsem opět mutex, protože v této sekci se provádí úprava FAT tabulky i klastrů, takže by bylo nežádoucí, aby více vláken zapisovalo na stejné místo. V kritické sekci najdu nejbližší volné místo, kam špatný klastr uložit, přeuložím získaná data ze špatného klastru a správně označím ve FAT tabulce. Po opravě se pokračuje v cyklu, který přeskakuje mezi bloky.

4 Obsluha programu

Program je možné stáhnout na mém GitHubu. Ve složce *src* se necházejí zdrojové kódy a testovací soubor. Spuštěním programu *make* se zdrojové kódy zkompilují.
Spustitelný binární soubor se jmenuje *pseudo_fat* a spuští se takto:

```
./pseudo_fat nazev_fat_souboru pocet_vlaken
```

Počet vláken je nepovinný parametr. Ve výchozím nastavení je nastaveno jedno vlákno.

5 Závěr

Test rychlosti běhu programu jsem testoval na mém notebooku Thinkpad X200 s Core 2 Duo 2,26 GHz, 8 GB RAM a SSD Crucial M550. Testovací soubor jsem použil dodávaný soubor v zadání, kde jsem si upravil klastr tak, aby obsahoval špatný klastr. Z výsledků rychlosti běhu bylo nejrychlejší zpracování na jednom vlákně, poté při spuštění na čtyřech vláknech.