

Semestrání práce

z předmětu ZOS

Virtuální souborový systém

Jan Rádl

Obsah

1	Zadání	2
1.1	Téma	2
1.2	Příkazy	2
2	Analýza úlohy	4
2.1	I-uzlový souborový systém	4
2.1.1	I-uzel(i-node)	4
2.1.2	Adresářový záznam (Dir item)	5
2.1.3	SuperBlok	5
3	Popis implementace	6
3.1	Prostředí	6
3.2	Řešení	6
3.2.1	Struktura souboru	6
3.2.2	Výpočet rozložení	7
3.2.3	Bitové pole	7
3.2.4	Alokace a přidávání	7
3.2.5	Dealokace a odstranění	8
4	Uživatelská příručka	9
4.1	Překlad	9
4.2	Nastavení parametrů	9

Kapitola 1

Zadání

1.1 Téma

Tématem semestrální práce, bude práce se zjednodušeným souborovým systémem založeným na i-uzlech. Vaším cílem bude splnit několik vybraných úloh. Základní funkčnost, kterou musí program splňovat. Formát výpisů je závazný. Program bude mít jeden parametr a tím bude název vašeho souborového systému. Po spuštění bude program čekat na zadání jednotlivých příkazů s minimální funkčností viz níže (všechny soubory mohou být zadány jak absolutní, tak relativní cestou)

- Maximální délka názvu souboru bude $8+3=11$ znaků (jméno.přípona) + `\0` (ukončovací znak v (C/C++), tedy 12 bytů.
- Každý název bude zabírat právě 12 bytů (do délky 12 bytů doplníte `\0` - při kratších názvech)

1.2 Příkazy

1. cp - kopíruje soubory
2. mv - přesouvá soubory
3. rm - maže soubory
4. ln - vytvoří hardlink na soubor
5. cat - vypíše obsah souboru jako sekvenci charů
6. mkdir - vytváří adresář
7. rmdir - ruší prázdný adresář
8. ls - vypíše obsah adresáře

9. cd - změnění aktuální adresář
10. pwd - vypíše cestu od root adresáře k aktuálnímu adresáři
11. info - vypíše informace do daném i-uzlu
12. incp - nahraje soubor do vfs
13. outcp - vytvoří kopii souboru z vfs do domovkého souborového systému
14. load - začne vykonávat příkazy ze zadaného souboru
15. format - provede zformátování vfs na požadovanou velikost v MB

Kapitola 2

Analýza úlohy

2.1 I-uzlový souborový systém

2.1.1 I-uzel(i-node)

Základní jednotka souborového systému obsahující všechny podstatné informace o datech souboru nikoliv však jeho jméno. Jmenovitě:

- unikátní identifikátor i-uzlu
- typ souboru (soubor/adresář)
- velikost souboru
- počet odkazů ukazující na tento soubor
- kolekce ukazatelů na data

Přímé adresování

I-uzel obsahuje přímo adresu data bloku.

Inline adresování

Některé moderní souborové systémy dovolují malé množství dat uložit přímo v i-uzlové struktuře místo data bloku. Tento styl je pro tuto implementaci nevhodný kvůli poměrně malé velikosti i-uzlu.

Nepřímé adresování n řádu

Obdobně jako u nepřímého adresování pro proměnné tak i zde je uložena pouze adresa na datový blok obsahující odkazy o jeden řád nižší a pokud řád dosáhne 0, tak daný odkaz opět jako u přímého adresování obsahuje data souboru.

2.1.2 Adresářový záznam (Dir item)

Je další velice důležitou součástí toho systému ukládání souboru, protože dovoluje soubory ve souborovém systému pojmemovávat. Jedná se o jednoduchou strukturu obsahující jméno souboru a unikátní identifikátor i-uzlu s tímto názvem.

2.1.3 SuperBlok

Je první struktura v souboru, která obsahuje informace pro zavedení a obsluhu daného souborového systému. Určuje rozdělení paměti na 4 části. Obsahuje:

- celkovou velikost disku
- velikost data bloku
- počet inodu
- počet data bloků
- adresu pole bitů reprezentují použité i-uzly
- adresu pole bitů reprezentují použité data bloky
- adresu prvního i-uzlu (adresa i-uzlové části systému)
- adresu prvního data bloku (adresa datatové části)

Kapitola 3

Popis implementace

3.1 Prostředí

Pro implementaci jsem si vybral programovací jazyk C++(CPP) v jeho podobě definované v jeho standartu 17(c++17). Pro jednoduché sestavení využívám nástroj CMake. Program je členěn do 2 částí:

- inodef - knihovna, která simuluje samotný souborový systém
- fsterminal - slouží pro obsluhu systému pomocí příkazů

3.2 Řešení

3.2.1 Struktura souboru

Samotný soubor obsahující filesystem, je rozdělen do 5 částí:

1. SuperBlok
2. I-uzlové bitové pole
3. Data blokové bitové pole
4. Pole I-uzlů
5. Pole data bloků

Takto může vypadat rozložení pro předpokladané využití souboru:



Obrázek 3.1: Ilustrace rozložení souborového systému

3.2.2 Výpočet rozložení

Velikost superbloku(SB_s) je stálá, ale ostatní bloky jsou vázané na jiné bloky nebo na využitelnou velikost souboru. Ze zadané celkové velikosti souboru(D_s) se odečte velikost superbloku a 2 byty pro případ, že počty bloků a inodů nebudou dělitelné 8. Dále potřebuje poměr i-uzlů k data blokům v procentech(P_{ib}) a velikost bloku(B_s) nacházející se v souboru `config.hpp` a velikost i-uzlu(I_s). Poté se spočítá počet bloků(B_{cnt}) pro tuto velikost podle:

$$B_{cnt} = \frac{D_s - SB_s - 2}{\frac{P_{ib}}{8} + \frac{1}{8} + P_{ib} \cdot I_s + B_s}$$

Jednotlivé části jmenovatele odpovídají velikostem jednotlivých bloků po přenásobení B_{cnt} v bytech.

3.2.3 Bitové pole

Občas bývá také pojmenováno bitmapa. Moje implementace využívá implementaci bitSetu v CPP pro obsluhu bitových úprav a vlastního iterátoru. Logická představa o lineárním poli bitů se neshoduje s implementací. Reálně je toto pole je tvořeno jako posloupnost 0..n bytů, kde jsou bity v bytu jsou LSB.

	0														1																	
R:	0	7	0	6	0	5	0	4	0	3	0	2	0	1	0	0	1	5	1	4	1	3	1	2	1	1	1	0	0	9	0	8
L:	0	0	0	1	0	2	0	3	0	4	0	5	0	6	0	7	0	8	0	9	1	0	1	1	1	2	1	3	1	4	1	5

Obrázek 3.2: Logická vs Reálná

3.2.4 Alokace a přidávání

V celém programu je hodnota 0 považována jako prázdná/neplatná hodnota, až na root prvek, který této hodnoty nabývá číslo i-uzlu a je to jediné legální místo, které této hodnoty může nabývat. A jeho validitu potvrzuje bitové pole pro příznak obsazenosti tohoto i-uzlu.

AddPointer

Funkce, která pro zadaný i-uzel přidá ukazatel na zadaný datablok. Pokud program běží v debug konfiguraci, tak při přetečení adresovatelného postoru i-uzlem se program ukončí assetem a pokud je zapnuté logování, tak se zapíše overflow error. Tato situace je logická chyba, která nelze nijak vyřešit, protože její vyřešení by vyžadovalo rozšíření adresovatelného prostoru. V

release režimu je v i-uzlu uloženo maximum adresovatelných dat a zbytek je zapsán, ale je nedostupný.

AlocateX

Je dvojice funkcí, kde X je nahrazeno (inode/datablock), která projde příslušné bitové pole a pokud narazí na volný prvek, tak ho zabere a vrátí jeho indentifikátor.

3.2.5 Dealokace a odstranění

freeX

Obdobně jako AlocateX, ale jedná se o inverzní operaci, která navíc dané místo naplní opakujícím se znakem \0.

Remove dir item

Je funkce pro odstranění záznamu adresáře z rodičovského adresáře. Pokud daný záznam je nalezen, tak je odstraněn a nahrazen posledním záznamem v tomto bloku a pokud je to zároveň poslední záznam v tomto bloku, tak tento blok zůstane alokovaný, ale bude prázdný. Regenerace na úrovni bloků by byla příliš nákladná a navíc by zanesla do systému problém s nespojitostí datových ukazatelů v i-uzlu.

Kapitola 4

Uživatelská příručka

4.1 Překlad

Pro sestavení je potřeba využít nástroje CMake a díky tomu máte tyto možnosti:

- Logování - vytvoření souboru `main.log` obsahující informace o běhu programu podle nastavení `LOGLEVEL` v `log.hpp`
- Debug build - sestavení programu s kontrolou stavu programu za běhu
- Release build - výchozí verze programu, která nechrání proti erroru, ale přesto error může nastat

Příklad sestavení pro normální běh programu:

```
cmake -S <adresář projektu> -B <adresář sestavení>
```

Sestavení v debug modu:

```
cmake -DCMAKE_BUILD_TYPE=Debug -S <adresář projektu> -B <adresář sestavení>
```

Sestavení s logováním:

```
cmake -DLOGFLAG=ON -S <adresář projektu> -B <adresář sestavení>
```

Poté můžeme zavolat náš systémový nástroj pro setavení projektu nad adresářem setavení. Následně v adresáři `setavení/app` můžeme najít sestavený program pro obsluhu souborového systému pomocí terminálu.

4.2 Nastavení parametrů

Věškeré technické parametry souborového systému lze změnit před sestavením samotné aplikace. Při změnách velikosti bloku je potřeba zachovat dvě podmínky: velikost bloku je celočíselně dělitelná velikostí typu ukazatele a zároveň datablok musí být větší než velikost 2 diritemů. Dříve sestavený souborový systém se při změnách typů stává nepřenositelný.