OS MMXXIV

MIT ;)

https://pdos.csail.mit.edu/6.828

Obnova FS po zlyhaní

Zlyhanie FS

- Napríklad príde k výpadku napájania

- FS môže byť v nekonzistentnom stave
- Prečo? Zápis na disk je viackroková operácia
- Riešenie – logovanie (žurnál)

Riziká FS

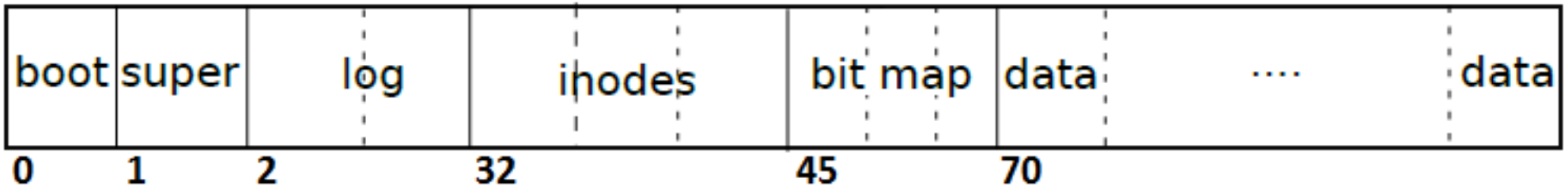
- Operácie FS pozostávajú z viacerých krokov (nie sú atomické)
  - Vytvorenie adresára alebo súboru
  - Zmazanie súboru
  - …
- Zlyhanie pri vykonávaní operácie nad FS môže zanechať FS v nekonzistentnom stave (porušený invariant FS)

Riziká FS

- Čo sa môže diať po reštarte (po zlyhaní)

- Môže prísť znovu k zlyhaniu
- Nemusí prísť k zlyhaniu, ale používateľ pomocou operácií čítania/zápisu nemusí dostať tie údaje, ktoré požaduje

Príklad



$ echo “hi” > x

1: write: 33 allocate inode for x

2: write: 33 init inode x

3: write: 70 record x in / directory’s data block

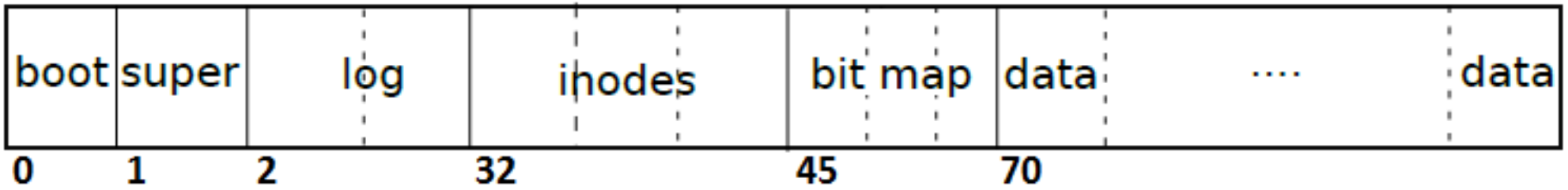
4: write: 32 update root inode

5: write: 33 update inode x

Pozorovanie

- Prehodenie poradia vykonávania zápisov nerieši problém
- Prehodením vyriešime jeden problém, ale vytvoríme ďalší (minimálne jeden)

Príklad



Zapísanie "hi\n" do súboru x

1: write: 45 set alloc bit in bitmap block

2: write: 845 write "hi" into file’s data block

3: write: 845 write "\n" into file’s data block

4: write: 33 update inode x (size, addr[0])

Pozorovanie

- Prehodenie poradia vykonávania zápisov nerieši problém
- Prehodením vyriešime jeden problém, ale vytvoríme ďalší (minimálne jeden)

- Riešenie: logovanie (žurnál)

Logovanie

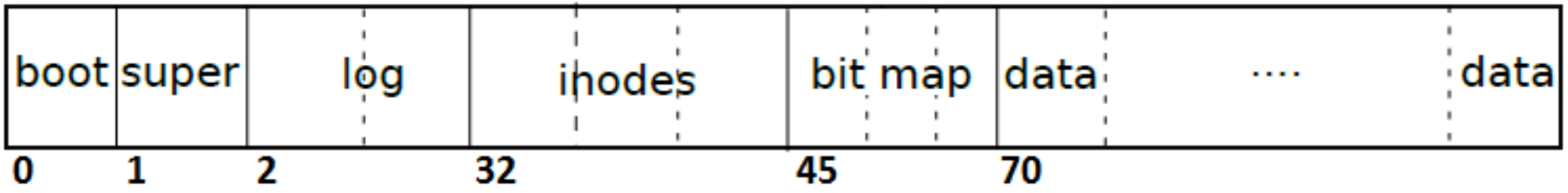
- Atomicita systémového volania nad FS
- Rýchla obnova po zlyhaní
- Zachovanie výkonu FS

Logovanie

- Atomicita systémového volania nad FS
- Rýchla obnova po zlyhaní
- Zachovanie výkonu FS
- Logovanie xv6 rieši prvé dva body, tretí nie

Logovanie v xv6

1. Logujú sa bloky, ktoré sa zapisujú
2. Do prvého log bloku sa vkladá potvrdenie (*commit*) operácie (počet blokov, ktoré sa majú zapísať) a čísla týchto blokov
3. V treťom kroku sa robí inštalácia blokov, ktoré sa nachádzajú v logu
4. Na záver sa log vyčistí (počet blokov, ktoré sa majú zapísať, sa nastaví na 0)



Logovanie xv6

- Zachováva atomicitu komplexnejšej operácie nad FS – buď sa urobia všetky zápisy alebo žiaden

- Kde sa nachádza začiatok blokov logu xv6?
- Aká je štruktúra logu? (obrázok na disku a v pamäti)

Princíp logovania xv6

- Pri zápise (`log_write`) na disk sa pridá číslo bloku do poľa čísiel blokov v logu v pamäti (nie na disku!)

- Samotné údaje bloku sú v `bcache` a nesmú byť z nej vyhodené (`pin`/`unpin`)

Princíp logovania xv6

- Čo sa deje pri vyvolaní operácie *commit*
  - Zapísanie bufs z `bcache` do logu na disku
  - WAIT na dokončenie zápisov (synchrónnosť)
  - Zapísanie hlavičky logu (jeden blok) na disk
    - Nenulové N
    - Čísla blokov, ktoré sa majú inštalovať

Princíp logovania xv6

- Čo sa deje pri vyvolaní operácie *commit*
  - Zapísanie bufs z `bcache` do logu na disku
  - WAIT na dokončenie zápisov (synchrónnosť)
  - Zapísanie hlavičky logu (jeden blok) na disk
    - Nenulové N
    - Čísla blokov, ktoré sa majú inštalovať
- Po operácii *commit*
  - Inštalácia (zápis) obsahu blokov na ich miesta na disku
  - Zavolanie `unpin()` na tieto bloky
  - Zapísanie N = 0 do hlavičky logu na disku

Princíp logovania xv6

- Hodnota ‘N’ v hlavičke logu na disku určuje platný *commit*

- 0 znamená, že k potvrdeniu transakcie neprišlo, a teda nič sa nebude inštalovať
- Nenulová hodnota znamená, že potvrdenie nastalo, obsah logovacích blokov je platný, a je potrebné dokončiť transakciu

Kód xv6

- Viď výstup xv6 po modifikácii `bwrite`

- Viď kód `sysfile.c:sys_write()`
- Viď kód `file.c:filewrite()`
  - Vypočíta maximálny počet blokov na 1 zápis (aby sa ich počet zmestil do jednej transakcie logu)
  - `begin_op()`
  - `writei()`
  - `end_op()` volá samotný `commit()`

Kód xv6

- Čo robí `filewrite()` spolu s `writei()`

`begin_op()`

`bmap()` – môže zapísať bitmap, indir block

– `bzero()` vyvolá `log_write()`

`bread()`

modifikácia `bp->data`

`log_write()` absorbovanie `bzero`

`iupdate()` – zápis i-uzla vyvolá `log_write()`

`end_op()`

Kód xv6

- `begin_op()`
  - Kontrola prebiehajúceho commitu
  - Kontrola miesta v logu na disku
  - Zvýšenie počtu prebiehajúcich sys. volaní, ktoré chcú zapisovať na disk (zvýšením hodnoty premennej `outstanding`)

Kód xv6

- `begin_op()`
  - Kontrola prebiehajúceho commitu
  - Kontrola miesta v logu na disku
  - Zvýšenie počtu prebiehajúcich sys. volaní, ktoré chcú zapisovať na disk (zvýšením hodnoty premennej `outstanding`)
- `log_write()`
  - Ak je blok v logu, nepridáva sa (**absorbovanie**)
  - `pin()` bloku, ktorý sa má pridať do logu
  - Pridanie čísla bloku do poľa v hlavičke v pamäti
  - Zvýšenie počtu blokov na zápis v hlavičke v pamäti

Kód xv6

- `end_op()`
  - Ak neprebieha žiadna ďalšia operácia, ktorá patrí do atomickej transakcie s diskom (`outstanding` je 0), `commit()`
- `commit()`
  - Zapísanie blokov z `bcache` do logu na disk
  - Zapísanie hlavičky logu (čísla blokov a N) na disk
  - Inštalácia blokov z logu na disku na príslušné miesto na disku
    - V rámci inštalácie `bunpin()` blokov
  - Vymazanie hlavičky logu na disku (vynulovanie N)

Logovanie v xv6

- Je primitívne a pomalé
- Predpokladá, že zariadenie disku je bezchybné

- Výzvy, s ktorými sa treba vysporiadať pri programovaní aj takto jednoduchého logovania

Výzvy návrhu logovania

- Blok, ktorý treba logovať, sa nesmie vyhodiť z bcache
- Veľkosť dát systémového volania sa musí zmestiť do log záznamov na disku
- Umožniť konkurentné vykonávanie viacerých systémových volaní
- Jeden blok môže byť zapisovaný na disk počas jednej transakcie viackrát

Výzvy návrhu logovania

- Blok, ktorý treba logovať, sa nesmie vyhodiť z `bcache`
  - Pred pridaním do transakcie sa zistí dostatok voľného miesta v blokoch logu
  - Urobí sa `pin()` (zvýši sa počet referencií v `bcache`)
  - Poi inštalácii bloku sa robí `unpin()`
- Veľkosť dát systémového volania sa musí zmestiť do log záznamov na disku
  - Ak sa dáta nezmestia do transakcie, čaká sa
  - Ak sa nezmestia do jednej transakcie, dáta sa rozdeľujú na viac atomických zápisov → **?????**

Výzvy návrhu logovania

- Umožniť konkurentné vykonávanie viacerých systémových volaní
  - Vstúpiť do transakcie je možné, počet aktívnych sys. volaní v transakcii udržiava premenná `outstanding`
  - Ak nie je miesto v logu, sys. volanie nemôže vstúpiť do transakcie; čaká na ukončenie predošlej
- Jeden blok môže byť zapisovaný na disk počas jednej transakcie viackrát
  - Technika absorbovania zápisu
  - Blok v `bcache` odráža stav (viacerých) transakcií bez operácie `commit`; inštalácia bloku však nastáva až po op. `commit`, keď nie je žiadna transakcia aktívna

Pozitíva návrhu logovania xv6

- Korektnosť vďaka pravidlu *write-ahead*: nezapisuj zmeny na disk, kým nepotvrdíš transakciu v logu
- Priemerná priepustnosť disku: vďaka logovaniu sú zápisy na disk dávkované (nie sú po jednom)
- Konkurentné vykonávanie systémových volaní je obmedzené veľkosťou dát, ktoré sa pomocou nich zapisuje

Negatíva návrhu logovania xv6

- Problém operácií, ktoré sa nezmestia na 1x do logu (zápis veľkých súborov NIE je atomický)

- Celková efektivita riešenia je slabá
  - Každý blok sa na disk zapisuje 2x (raz do logu, druhý raz pri inštalácii na svoje miesto)
  - Logujú sa celé bloky, aj keď sa zmení pár bitov
  - Zapisovanie logu prebieha synchrónne (čaká sa na dokončenie operácie zápisu na disk)

Domáce čítanie 1

Chapter 8

File System

xv6: a simple, Unix-like teaching operating system

Tie časti, ktoré hovoria o logovaní transakcií FS

Domáce čítanie 2

Na cvičenie si naštudujte manuálovú stránku `mmap` a `munmap`.

- V časti „The flags argument“ si všímajte iba úvod a popis nasledovných príznakov: `MAP_SHARED`, `MAP_PRIVATE`, `MAP_ANONYMOUS`, `MAP_UNINITIALIZED`
- Vynechajte sekcie Errors, Attributes, Versions, History, Bugs, See also, Colophon
- Z časti Notes čítajte iba po „Using MAP_FIXED safely“

https://man7.org/linux/man-pages/man2/mmap.2.html