OS MMXX

MIT ;)

https://pdos.csail.mit.edu/6.828/2020

Obnova FS po zlyhani

Zlyhanie FS

- Napriklad pride k vypadku napajania

- FS moze byt v nekonzistentnom stave
- Preco? Zapis na disk je viackrokova operacia
- Riesenie – logovanie (zurnal)

Rizika FS

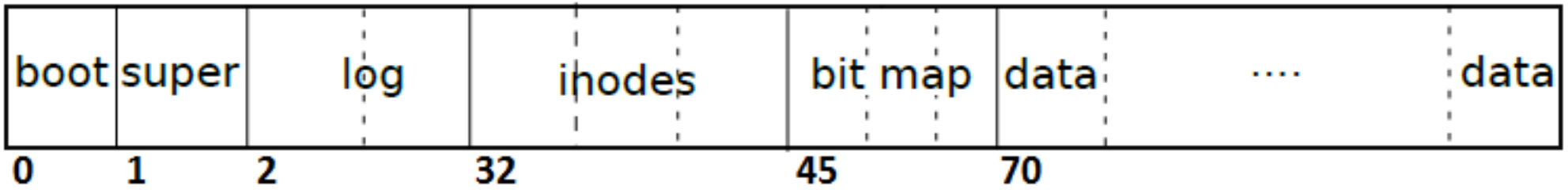
- Operacie FS pozostavaju z viacerych krokov (nie su atomicke)
  - Vytvorenie adresara alebo suboru
  - Zmazanie suboru
  - …
- Zlyhanie pri vykonavani operacie nad FS moze zanechat FS v nekonzistentnom stave (poruseny invariant FS)

Rizika FS

- Co sa moze diat po restarte (po zlyhani)

- Moze prist znovu k zlyhaniu
- Nemusi prist k zlyhaniu, ale uzivatel pomocou operacii citania/zapisu nemusi dostat tie data, ktore pozaduje

Priklad



$ 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

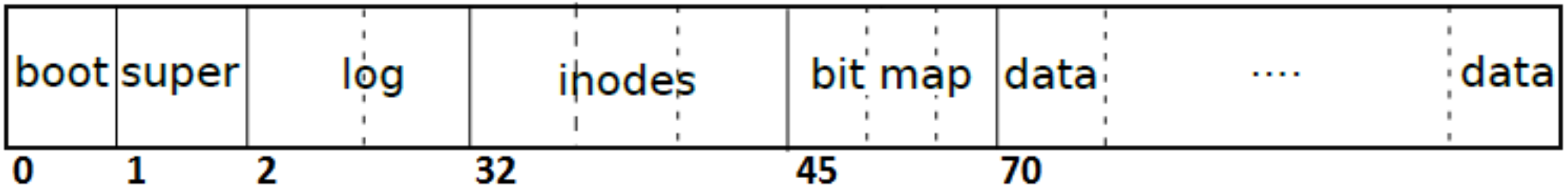
4: write: 32 update root inode

5: write: 33 update inode x

Pozorovanie

- Prehodenie poradia vykonavania zapisov neriesi problem
- Prehodenim vyriesime jeden problem, ale vytvorime dalsi (minimalne jeden)

Priklad



Zapisanie “hi\n” do suboru `x`

1: write: 45 set alloc bit in bitmap block

2: write: 644 write `hi` do bloku dat suboru

3: write: 644 write `\n` do bloku dat suboru

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

Pozorovanie

- Prehodenie poradia vykonavania zapisov neriesi problem
- Prehodenim vyriesime jeden problem, ale vytvorime dalsi (minimalne jeden)

- Riesenie: logovanie (zurnal)

Logovanie

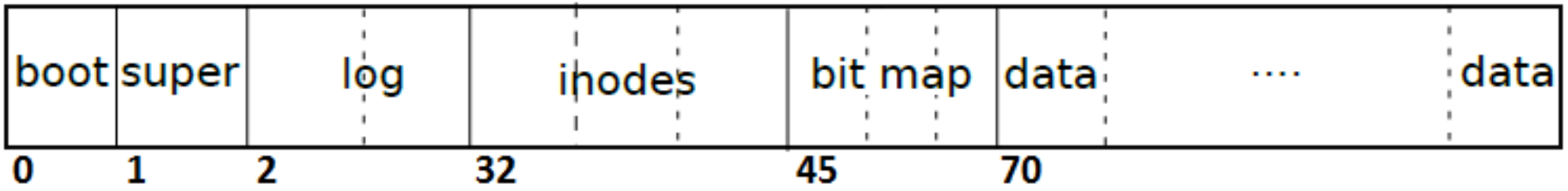
- Atomicita systemoveho volania nad FS
- Rychla obnova po zlyhani
- Zachovanie vykonu FS

Logovanie

- Atomicita systemoveho volania nad FS
- Rychla obnova po zlyhani
- Zachovanie vykonu FS
- Logovanie xv6 riesi prve dva body, treti nie

Logovanie v xv6

1. Loguju sa bloky, ktore sa zapisuju
2. Na koniec blokov sa pripaja `commit` operacie (pocet blokov, ktore sa maju zapisat)
3. V tretom kroku sa robi instalacia blokov, ktore sa nachadzaju v logu
4. Na zaver sa log vycisti (pocet blokov, ktore sa maju zapisat, sa nastavi na 0)



Logovanie xv6

- Zachovava atomicitu komplexnejsej operacie nad FS – bud sa urobia vsetky zapisy alebo ziaden

- Kde sa nachadza zaciatok blokov logu xv6?
- Aka je struktura logu? (obrazok)

Princip logovania xv6

- Pri zapise (bwrite) na disk sa prida cislo bloku do pola cisiel blokov v logu v pamati (nie na disku!)

- Samotne udaje bloku su v bcache (pin/unpin)

Princip logovania xv6

- Co sa deje pri commite
  - Zapisanie bufs z bcache do logu na disku
  - WAIT na dokoncenie zapisov (synchronnost)
  - Zapisanie hlavicky logu (1 blok) na disk
    - Nenulove N
    - Cisla blokov, ktore sa maju instalovat

Princip logovania xv6

- Co sa deje pri commite
  - Zapisanie bufs z bcache do logu na disku
  - WAIT na dokoncenie zapisov (synchronnost)
  - Zapisanie hlavicky logu (1 blok) na disk
    - Nenulove N
    - Cisla blokov, ktore sa maju instalovat
- Po commite
  - Instalacia (zapis) obsahu blokov na ich miesta na disku
  - Zavolanie unpin() na tieto bloky
  - Zapisanie 0 do hlavicky logu na disku

Princip logovania xv6

- Hodnota ‘N’ v hlavicke logu na disku znamena commit

- 0 znamena, ze k potvrdeniu transakcie neprislo, a teda nic sa nebude instalovat
- Nenulova hodnota znamena, ze potvrdenie nastalo, obsah logovacich blokov je platny, a je potrebne dokoncit transakciu

Kod xv6

- Vid vystup xv6 po modifikacii bwrite

- Vid kod sysfile.c:sys_write()
- Vid kod file.c:filewrite()
  - Vypocita maximalny pocet blokov na 1 zapis (aby sa ich pocet zmestil do logu)
  - begin_op()
  - writei()
  - end_op() vola samotny commit()

Kod xv6

- Co robi filewrite() spolu s writei()
  - begin_op()
  - bmap() – moze zapisat bitmap, indir block
  - – bzero() vyvola log_write()
  - bread()
  - modifikacia bp->data
  - log_write() – absorbovanie bzero log_write()
  - iupdate() – zapis i-uzla
  - end_op()

Kod xv6

- begin_op()
  - Kontrola prebiehajuceho commitu
  - Kontrola miesta v logu na disku
  - Oznacenie bloku operacii, ktore musia byt atomicke (zvysenim premennej `outstanding`)

Kod xv6

- begin_op()
  - Kontrola prebiehajuceho commitu
  - Kontrola miesta v logu na disku
  - Oznacenie bloku operacii, ktore musia byt atomicke (zvysenim premennej `outstanding`)
- log_write()
  - Ak je blok v logu, nepridava sa (absorbovanie)
  - pin() bloku, ktory sa ma pridat do logu
  - Pridanie cisla bloku do pola v hlavicke v pamati
  - Zvysenie poctu blokov na zapis v hlavicke v pamati

Kod xv6

- end_op()
  - Ak neprebieha ziadna dalsia operacia, ktora patri do atomickej transakcie (`outstanding` je 0), commit()
- commit()
  - Zapisanie blokov z bcache do logu na disk
  - Zapisanie hlavicky logu (cisla blokov a N) na disk
  - Instalacia blokov z logu na disku na prislusne miesto na disku
    - V ramci instalacie bunpin() blokov
  - Vymazanie hlavicky logu na disku (vynulovanie N)

Logovanie v xv6

- Je primitivne a pomale
- Predpoklada, ze zariadenie disku je bezchybne

Logovanie v xv6

- Je primitivne a pomale
- Predpoklada, ze zariadenie disku je bezchybne

- Vyzvy, s ktorymi sa treba vysporiadat pri programovani aj takto jednoducheho logovania

Vyzvy navrhu logovania

- Blok, ktory treba logovat, sa nesmie vyhodit z bcache
- Velkost dat systemoveho volania sa musi zmestit do log zaznamov
- Umoznit konkurentne vykonavanie viacerych systemovych volani
- Jeden blok moze byt zapisovany na disk pocas jednej transakcie viackrat

Vyzvy navrhu logovania

- Blok, ktory treba logovat, sa nesmie vyhodit z bcache
  - Pred pridanim do transakcie sa zisti dostatok volneho miesta v blokoch logu
  - Urobi sa pin() (zvysi sa pocet referencii v bcache)
  - Po commite sa robi unpin()
- Velkost dat systemoveho volania sa musi zmestit do log zaznamov
  - Ak sa data nezmestia do transakcie, caka sa
  - Ak sa nezmestia do jednej transakcie, data sa rozdeluju na viac atomickych zapisov → **?????**

Vyzvy navrhu logovania

- Umoznit konkurentne vykonavanie viacerych systemovych volani
  - Vstupit do transakcie je mozne, pocet aktivnych sys volani v transakcii udrziava premenna `outstanding`
  - Ak nie je miesto v logu, sys volanie nemoze vstupit do transakcie; caka na ukoncenie predoslej
- Jeden blok moze byt zapisovany na disk pocas jednej transakcie viackrat
  - Technika `absorbovania zapisu`
  - Blok v bcache odraza stav (viacerych) transakcii bez commitu; instalacia bloku vsak nastava az po commite, ked nie je ziadna transakcia aktivna

Pozitiva navrhu logovania xv6

- Korektnost vdaka pravidlu `*write-ahead*`: nezapisuj zmeny na disk, kym nepotvrdis transakciu v logu
- Priemerna priepustnost disku: vdaka logovaniu su zapisy na disk davkovane (nie su po jednom)
- Konkurentne vykonavanie systemovych volani je obmedzene velkostou dat, ktore sa pomocou nich zapisuje

Negativa navrhu logovania xv6

- Problem operacii, ktore sa nezmestia na 1x do logu (zapis velkych suborov NIE je atomicky)

- Celkova efektivita riesenia je slaba
  - Kazdy blok sa na disk zapisuje 2x (raz do logu, druhy raz pri instalacii na svoje miesto)
  - Loguju sa cele bloky, aj ked sa zmeni par bitov
  - Zapisovanie logu prebieha synchronne (caka sa na dokoncenie operacie zapisu na disk)

Domace citanie

Chapter 8

File System

xv6: a simple, Unix-like teaching operating system

Tie casti, ktore hovoria o logovani transakcii FS