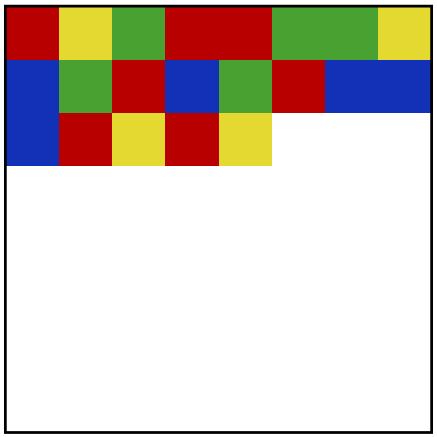
Seminar: Defragmentacija

Žan Magerl

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko

18. januar 2020



Vsebina

- Opis problema
- Fragmentacija
  - Vrste
  - Vzroki
  - Datotečni sistem
- Defragmentacija
  - Iskanje fragmentiranih datotek
  - Delovanje (teoretični algoritmi vs. implementacija v OS)
  - Implementirane rešitve
  - Defragmentacija in SSD
- Viri

Opis problema

- Fragmentacija upočasnjuje delovanje računalnika
- Razrobljenost datotek
- Lastnosti diska:
  - Mehanski deli
  - HDD vs SSD
- Defragmentacija odpravlja fragmentacijo
- Defragmentacija je težek problem (velika poraba virov, ...)

Vrste

V splošnem poznamo 3 vrste fragmentacije:

- Datotečna fragmentacija
- Fragmentacija praznega prostora
- Fragmentacija povezanih datotek (lokalnost) $\rightarrow$ bolj abstraktna

Vzroki

Dve “kritični” operaciji:

- Brisanje datoteke
  - Z brisanje za seboj puščamo luknje (2 možnosti)
  - Dodajanje datotek v luknje:
    - *first-fit algorithm*
    - *best-fit algorithm*
    - *worst-fit algorithm*
  - Fragmentacija datoteke

Vzroki

- Razširanje datoteke
  - Datoteke s pogostim spreminjanjem (logi, procesne datoteke, ipd.)
  - Datoteka se ne more širiti, če se za njo nahaja druga datoteka:
    - Premik celotne datoteke na novo lokacijo (počasno, ni vedno mogoče)
    - Fragmentiranje datoteke (hitro, poveča fragmentacijo $\rightarrow$ na dolgi rok ni dobro)

Datotečni sistem

- Organizira in nadzoruje shranjevanje na disk
- Osnovna enota: datoteka
- Datoteke poleg vsebine uporabljajo tudi metapodatke (ID, dovoljenja, tip, lokacija, velikost, ...)
- Linux (ext4) uporabljam *inode*

Terminologija

Nekaj o pogostih pojmih s področja datotečnih sistemov

- Sektor
  - Najmanjša enota na disku (del tračnice)
  - Velikost: 512 B (tradicionalno), 4 KiB (AF)
- Blok
  - Z njimi upravlja/operira datotečni sistem (bločno-orientirana naprava)
  - Fiksne dolžine (odvisno od konfiguracije): ponavadi večkratnik velikosti bloka
- Ekstent
  - Kontinuiran prostor za datoteke
  - Ekstent je označen z začetnim končnim blokom
  - Datoteka je lahko sestavljena iz več ekstentov in v primeru fragmentirane datoteke, ima vsak fragment svoj ekstent

Vrste datotečnih sistemov

- Skupaj z razvojom OS so se razvijali tudi različni datotečni sistemi
- Določeni OS so lahko naloženi na diske samo z določenim datotečnim sistemom
- Najpogostoješi datotečni sistemi
  - Linux: ext4
  - Windows: NTFS
  - MacOS: APFS (2017), HFS+

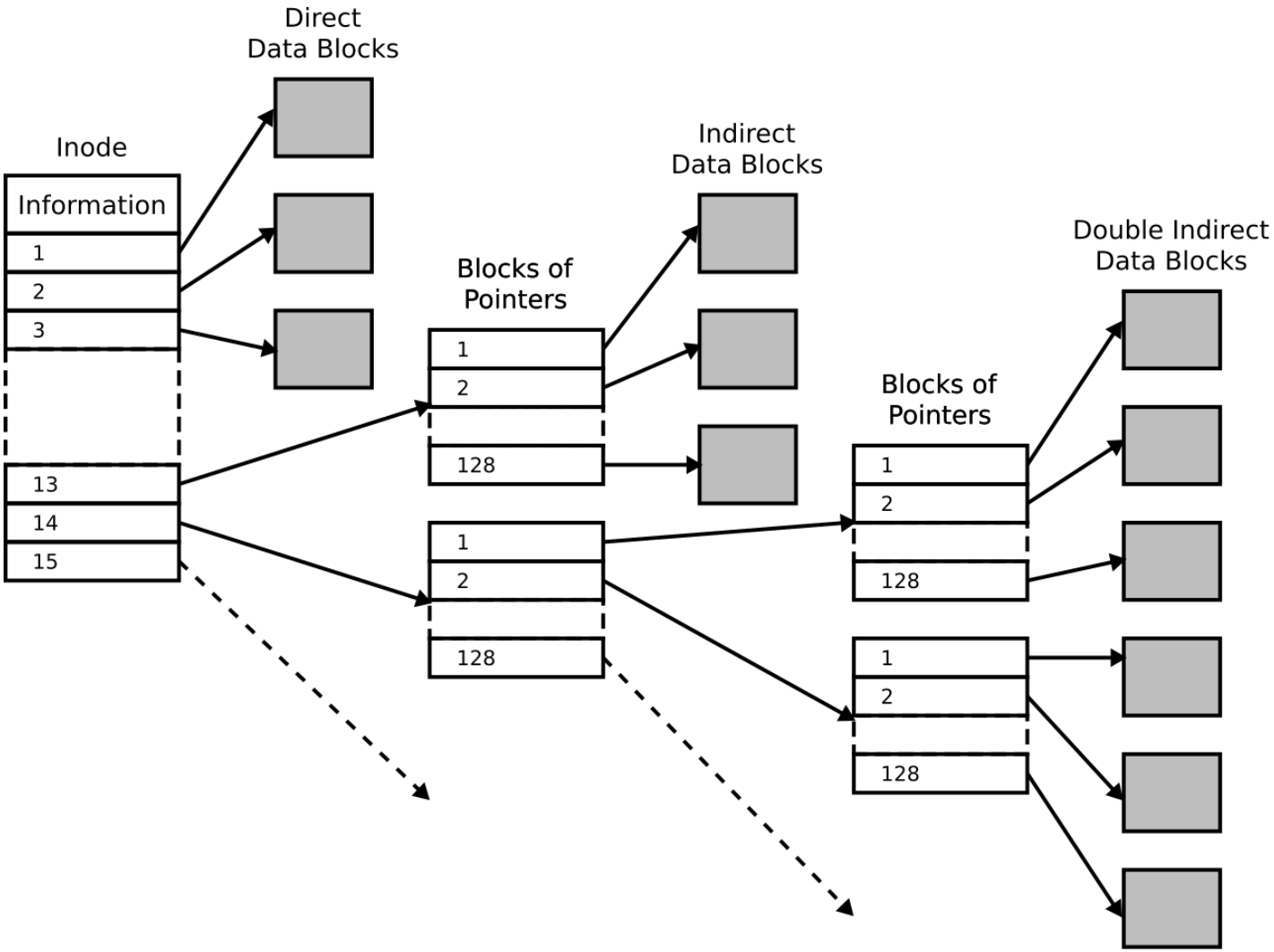
Iskanje fragmentiranih datotek

- Za postopek defragmentacije rabimo prvo najti fragmentirane datoteke
- Postopek iskanja je odvisen od OS in datočnega sistema
  - Principi so splošni, a podrobnosti se razlikujejo
- Metapodatki $\rightarrow$ ext: inode

Inode (ext2/ext3)

- Metapodatki: ID, velikost, dovoljenja, tip, ...
- Vsebujejo pa tudi lokacije blokov!
- Tabela vsebuje 15 kazalcev
  - Prvih 12 kazalcev kaže direktno na bloke
  - 13. kazalec kaže na tabelo kazalcev na bloke
  - 14. kazalec kaže na tabelo tabel kazalcev na bloke
  - 15. kazalec kaže na tabelo tabel tabel kazalcev
- Fragmentacijo datoteke ugotovimo iz skakanja naslovov

Inode (ext2/ext3)



Slika: Struktura inode kazalcev

Inode ext4

- Metapodatki: ID, velikost, dovoljenja, tip, ...
- Namesto lokacije blokov vsebuje lokacije ekstentov:
  - Boljša prostorska učinkovitost
  - Lažje iskanje delcev datotek (fragmentov)

Filefrag

```
[zanmagerl@polhec test]$ filefrag -v large_file.mkv
Filesystem type is: ef53
File size of large_file.mkv is 1062965839 (259514 blocks of 4096 bytes)
 ext:     logical_offset:        physical_offset: length:   expected: flags:
   0:        0..    2047:    9189376..   9191423:   2048:
   1:     2048..   34815:    9193472..   9226239:  32768:    9191424:
   2:    34816..   67583:    9226240..   9259007:  32768:
   3:    67584..  100351:    9259008..   9291775:  32768:
   4:   100352..  133119:    9291776..   9324543:  32768:
   5:   133120..  165887:    9324544..   9357311:  32768:
   6:   165888..  198655:    9357312..   9390079:  32768:
   7:   198656..  231423:    9390080..   9422847:  32768:
   8:   231424..  245759:    9422848..   9437183:  14336:
   9:   245760..  247807:    9648128..   9650175:   2048:    9437184:
  10:   247808..  259513:    9658368..   9670073:  11706:    9650176: last,eof
large_file.mkv: 4 extents found
```

Slika: Seznam ekstentov datoteke velike približno 1 GB

Lokacija datoteke

```
[zanmagerl@polhec test]$ stat a.txt
  File: a.txt
  Size: 47              Blocks: 8          IO Block: 4096   regular file
Device: fd02h/64770d    Inode: 3688594     Links: 1
Access: (0664/-rw-rw-r--)  Uid: ( 1000/zanmagerl)   Gid: ( 1000/zanmagerl)
Context: unconfined_u:object_r:user_home_t:s0
Access: 2020-01-12 13:41:58.341837789 +0100
Modify: 2020-01-12 13:33:04.186149731 +0100
Change: 2020-01-12 13:33:04.193149760 +0100
 Birth: 2020-01-12 13:33:04.186149731 +0100
[zanmagerl@polhec test]$ sudo debugfs -R "blocks <3688594>" /dev/mapper/fedora-home
debugfs 1.44.6 (5-Mar-2019)
9124392
[zanmagerl@polhec test]$ address=$((9124392*4096/512))
[zanmagerl@polhec test]$ sudo dd if=/dev/mapper/fedora-home bs=512 skip=$address count=1 status=none | hexdump -C
00000000  50 6f 7a 64 72 61 76 6c  6a 65 6e 20 73 70 72 65  |Pozdravljen spre|
00000010  68 61 6a 61 6c 65 63 20  70 6f 20 64 61 74 6f 74  |hajalec po datot|
00000020  65 63 6e 65 6d 20 73 69  73 74 65 6d 75 21 0a 00  |ecnem sistemu!..|
00000030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000200
[zanmagerl@polhec test]$ cat a.txt
Pozdravljen sprehajalec po datotecnem sistemu!
```

Slika: Prikaz iskanja datoteke zgolj z inode-om datoteke

Algoritmi pri defragmentaciji

- Možnosti:
  - Eno datoteko
  - Del diska
- Pri eni datoteki stvar preprosta:
  - Pregledamo disk in si shranimo prosti prostor v terice: (lokacija, velikost)
  - Poiščemo datoteko, ki jo želimo defragmentirati
  - Pregledamo št. fragmentov, se odločimo ali lahko datoteko spravimo v eno izmed prej najdenih lokacij prostega prostora
  - Če se to da, datoteko enostavno premaknemo na novo mesto.

Algoritmi pri defragmentaciji dela diska

Trije pristopi:

- Prvi naivnejši pristop bo z uporabo večjega kosa praznega diska, ki bo deloval kot začasno odlagališče.
- Drugi pristop bo bil časovno počasnejši kot prvi algoritem, a bo porabil manj diska.
- Tretji pristop se bo posluževal filozofije sortiranja s štetjem, kar bomo poiskusili implementirati s čim manj porabe zunanjega prostora in bo združeval prednosti obeh prejšnih enostavnejših pristopov.

Hipotetičen osnoven primer

Trije pristopi:

- 64 blokov velik disk in 4 močno fragmentirane datoteke
- Gledamo na bloke kot na bloke in kot na bloke, na katere kažejo kazalci
- Cilj: čim bolj učinkovita fragmentacija