

Käyttöjärjestelmät II

Muistinhallinnan esimerkit

UNIX, Solaris, Linux
W2000

Ch 8.3-6 [Stal 05]

Käyttöjärjestelmät II

UNIX / Solaris MUISTINHALLINTA

UNIX / Solaris (SVR4)

n Vanhoissa UNIXeissa ei virtuaalimuistia

- u heittovaihtoivat aina kokonaisia prosesseja

n Sivutus

- u tarvenouto
- u osittain ytimessä
- u osittain pagedaemon-prosessissa (pid=2)
 - F käynnistyy aika-ajoin (esim. 250 ms:n välein) tarkistamaan, onko riittävästi vapaita sivutiloja (25%?)

n Heittovaihto

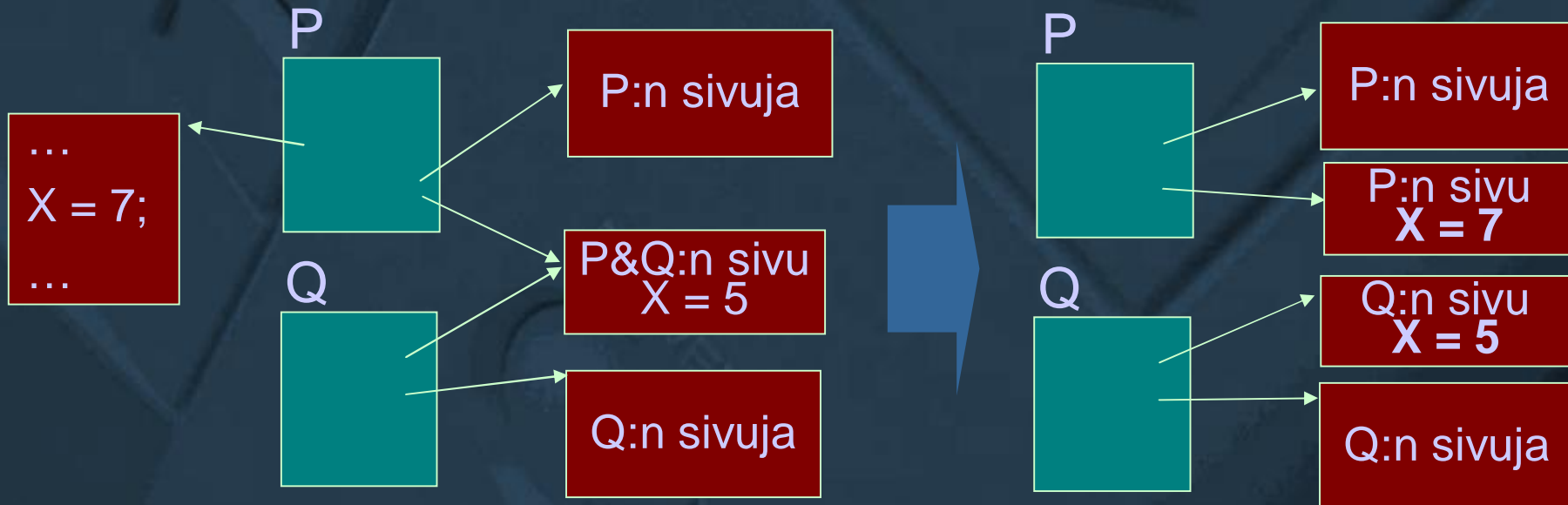
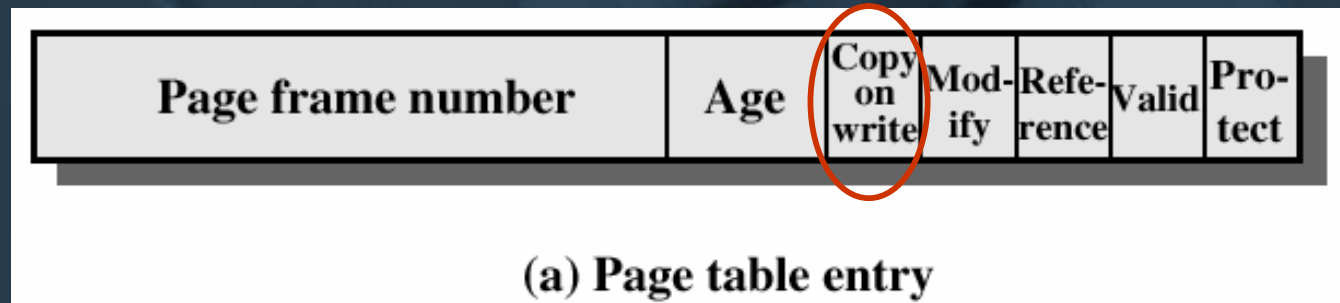
- u swapper-prosessi (pid=0)
- u pagedaemon käynnistää tarvittaessa
- u vain PCB (user structure) ja sivutaulu jää muistiin

UNIX/Solaris: Tietorakenteita

Tbl 8.5 [Stal05]

- Page table** - jokaisella prosessilla oma
- alkiot per virt. muistin looginen sivu

Missä sivu
keskusmuistissa?



UNIX/Solaris: Tietorakenteita

Disk block descriptor - alkio per virt. muistin looginen sivu

Missä sivu
tukimuistissa
(eli levyllä)?

| | | |
|--------------------|---------------------|-----------------|
| Swap device number | Device block number | Type of storage |
|--------------------|---------------------|-----------------|

(b) Disk block descriptor

Tbl 8.5 [Stal05]

UNIX/Solaris: Tietorakenteita

Tbl 8.5 [Stal05]

Page frame data table - alkio per fyysinen sivukehys,
vapaat sivutilat lisäksi free-listassa

Muistin
sivukehysten
tila?

| | | | | |
|-------------------|------------------------|-----------------------|---------------------|-----------------------|
| Page state | Reference count | Logical device | Block number | Pfdata pointer |
|-------------------|------------------------|-----------------------|---------------------|-----------------------|

(c) Page frame data table entry

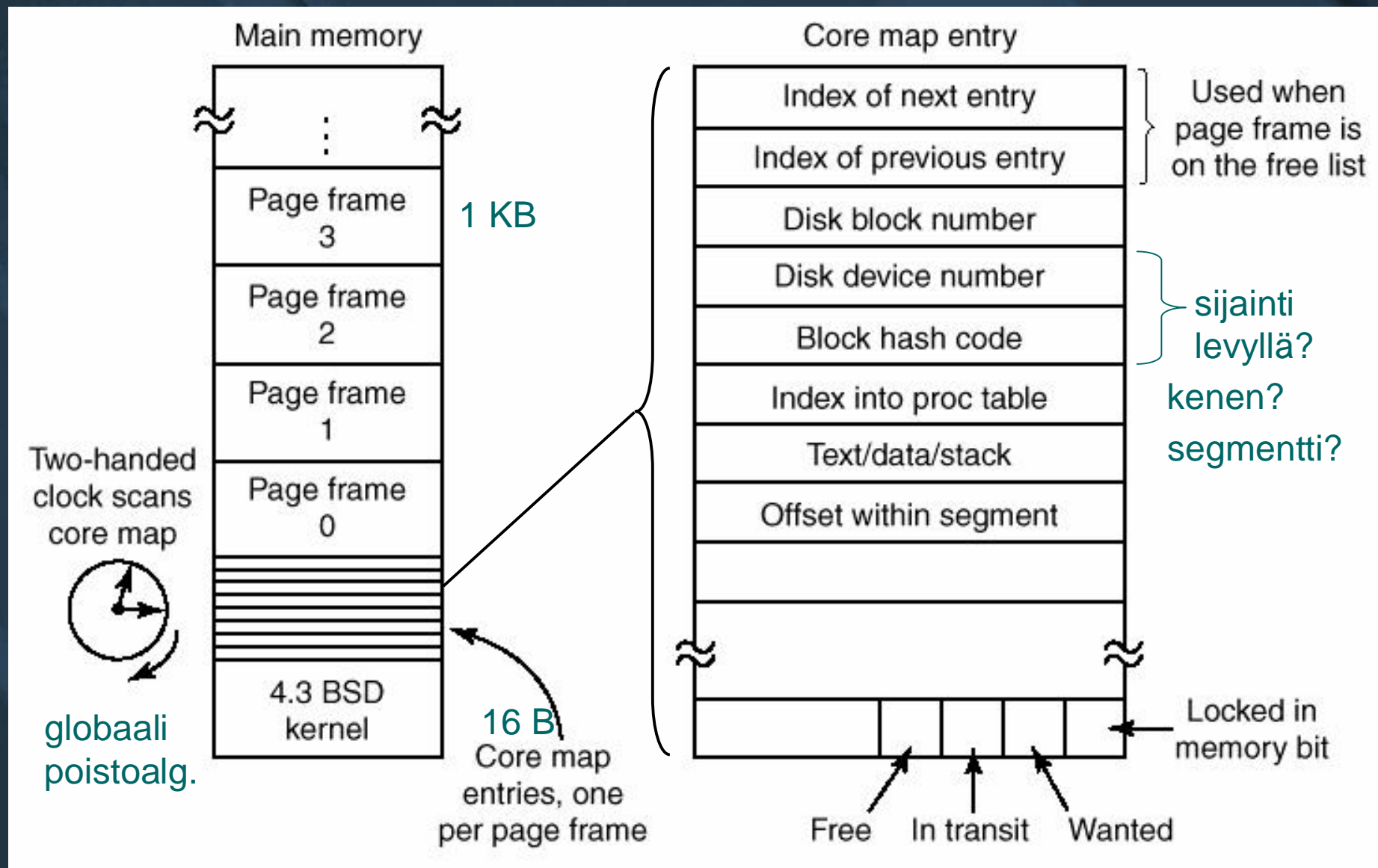
Swap-use table - alkio per levyllä oleva sivu

Tukimuistissa
olevan sivun
tiedot?

| | |
|------------------------|---------------------------------|
| Reference count | Page/storage unit number |
|------------------------|---------------------------------|

(d) Swap-use table entry

UNIX/4BSD: Sivukehystaulu (core map)



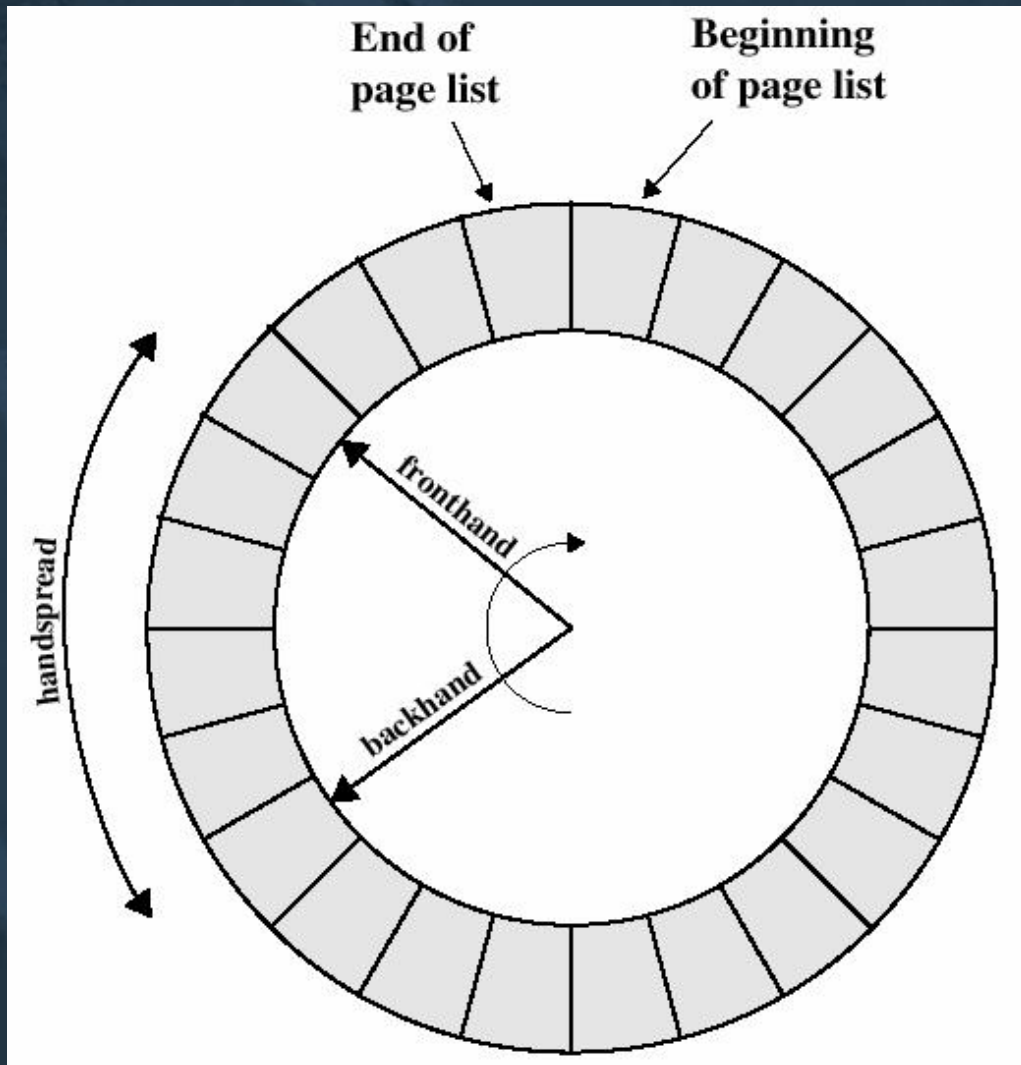
(Fig 10-16 [Tane01])

UNIX SVR4: Korvausalgoritmi

n Kaksiviisarinen globaali Clock poistoalgoritmi

- u tutki sivukehystaulua
- u prosessille annettujen sivukehysten (sivutilojen) lkm vaihtelee
- u etummainen viisari asettaa *Reference count* = 0
- u perässä tuleva viisari siirtää vapautettavaksi ne, joiden *Reference count* == 0 (edelleen?)

Kaksiviisarinen Clock



(Fig 8.23 [Stal05])

Fronthead:

set Reference = 0

Backhand:

if (Reference == 0)

page out

Idea: jos ei viittausta viime aikoina (eli viisarien pyyhkimisaikavälillä), niin kehyksessä oleva sivu ei kuulu käyttäjoukkoon

Miten nopeasti (frame/sec) viisarit liikkuvat?
Montako kehystä on viisareiden väli?

Unix SVR4: korvausalgoritmi

n Kaksiviisarinen clock

Fig 8.23 [Stal05]

- u viisarien nopeus (scanrate) välillä [slowscan, fastscan]
 - F molemmat viisarit etenevät samaa tahtia kehys kerrallaan, kunnes tarpeeksi vapaata tilaa
 - reset (fronthead); jos $U(\text{backhand})=0$, vapauta se ; etene
- u gap (handspread) viisarien välissä
 - F jos spread on pieni, vain hyvin usein viitattuihin sivuihin ehtii tulla viitebitti päälle
 - F Jos $\text{gap} = 359^\circ$, niin sama kuin 1-viisarinen clock (Fig. 8.16)
- u sopii paremmin suurille muisteille, nopeampi kuin tavallinen clock

n Ylläpidä: $\text{minfree} < \text{vapaiden sivutilojen lkm} < \text{maxfree}$

n Jos $\text{lkm} < \text{minfree}$, käynnistä swapper heittovaihtamaan prosesseja, jolloin sivutiloja vapautuu äkkiä paljon

- u pisimmän aikaa vähintään 20 sek. idle'nä ollut prosessi?
- u 4 suurimmasta prosessista kauttaen muistissa ollut prosessi?

(UNIX 4BSD)


n Jos lkm lähellä minfree-arvoa

- u kasvata scanrate-arvoa (vapauta herkemmin, käynnistä algoritmi useammin)

n Jos lkm lähellä maxfree-arvoa,

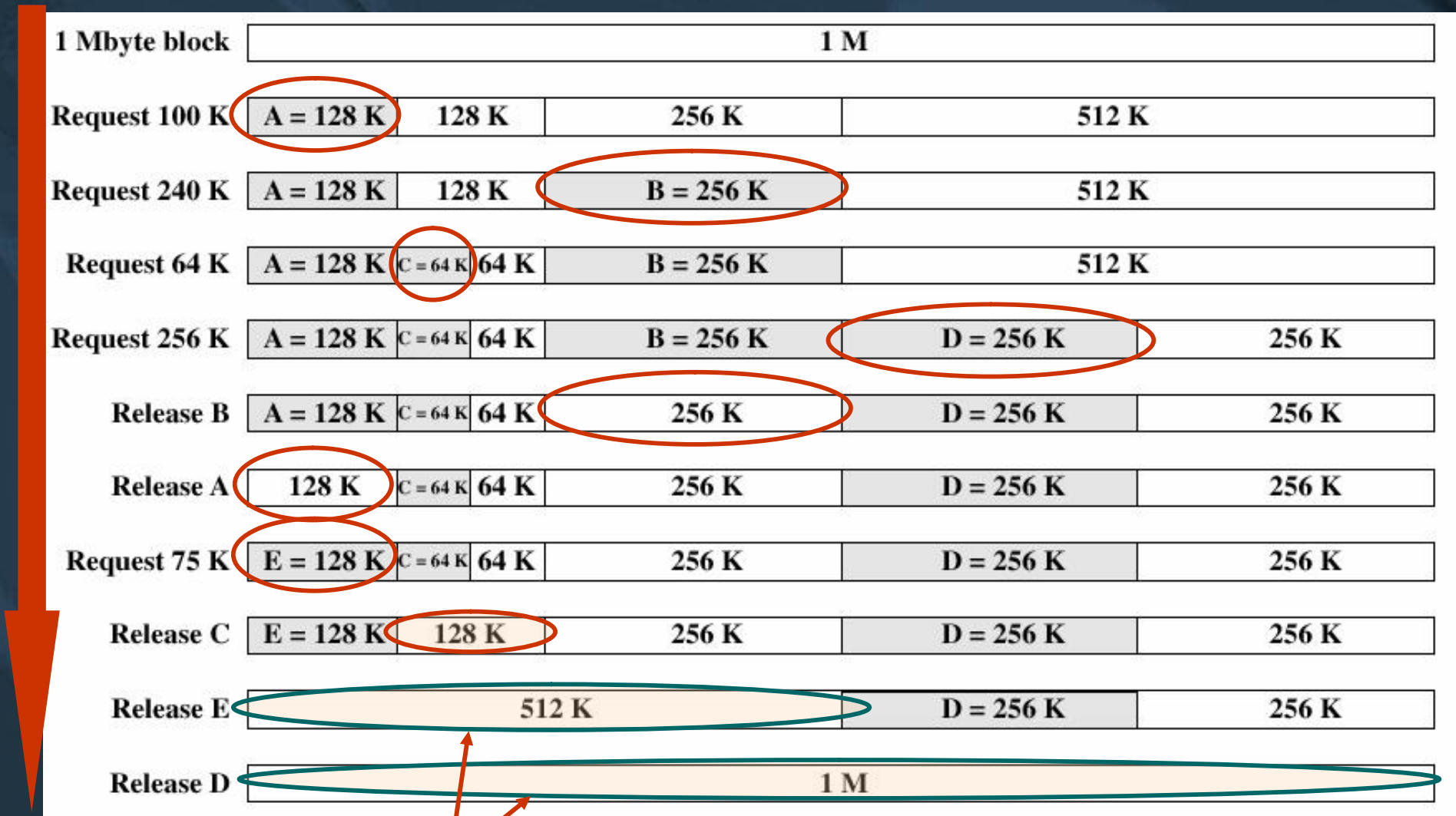
- u pienennä scanrate-arvoa (viitebitin asetukselle aikaa lisää)

UNIX/Solaris: ytimen (kernel) muisti

- n Perusvaraus sivutusta käyttäen
 - n Sivuja voi lukita muistiin
 - n Ydin varaa ja vapauttaa paljon pieniä alueita ja puskureita
 - u tiedoston polkunimen selvitys
 - u zombie-prosessista jälkeenjäävät tiedot
 - u dynaaminen varaus: proc, vnode, tiedostokuvaaja
 - n Ydin varaa ja vapauttaa myös sivun osia
 - u sivun sisällä tehtäviä varauksia hallitaan dynaamista tilanvarausta käyttäen (Ch 7 [Stal05])
 - u varaus ja vapautus oltava tehokkaita
- => Laiska Buddy System (Lazy Buddy)**
- u lohkojen yhdistelyä viivästetään, kunnes vapaita lohkoja on "liikaa"
- 

Buddy System (& lazy buddy system)

time



lazy buddy ei tee heti!

(Fig 7.6 [Stal05])

Muistiinkuvatut tiedostot

- n **Memory-mapped files**
- n **Tiedostovälimuisti on muistissa**
- n **Mapataan virtuaalimuistialue tiedostovälimuistin päälle**
- n **Tiedoston luku/kirjoitus muistin luku/kirjoitusoperaatioilla**
 - u nopea!
- n **Monta prosessia voi mapata helposti saman alueen**
 - u yhteiskäyttöiset tiedostot (koodi ja/tai data)
 - u ks. kuva seuraavalla sivulla

Muistiinkuvattu tiedosto

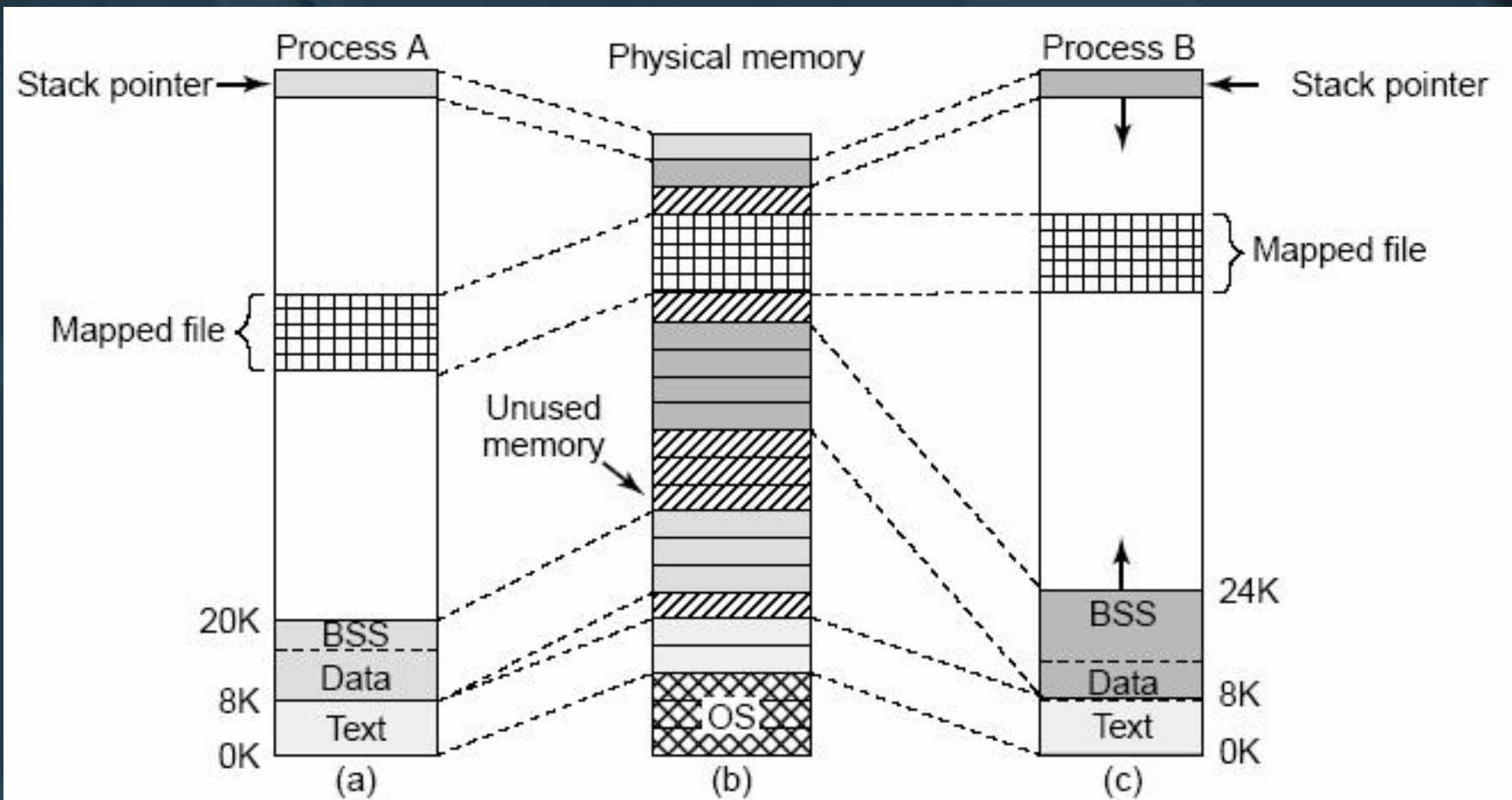
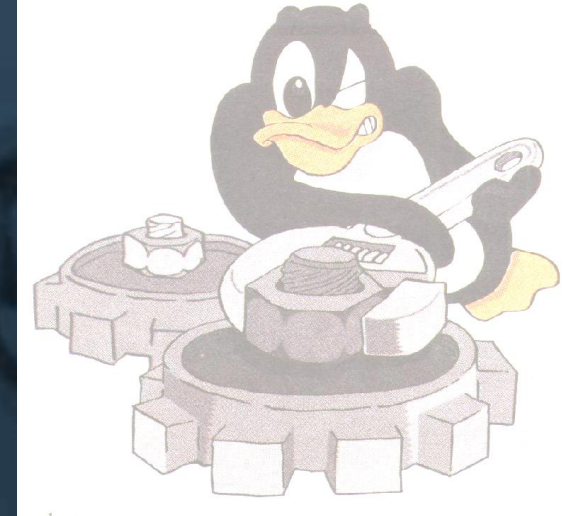


Fig. 10-14. Two processes can share a mapped file. [Tane01]

Käyttöjärjestelmät II



Linux

MUISTINHALLINTA

Ch 8.4 [Stal 05]

Linux: muistinhallinta



n 32-bittisissä arkkitehtuureissa

- u 3 GB:n virtuaaliavaruus prosesseille
- u 1 GB ytimelle (mm. sivutaulut)
 - F fyysisesti todellisen muistin alussa
- u etuoikeutetussa tilassa viitattavissa 4 GB
 - F prosessin oma 3 GB
 - F ytimen 1 GB

n Virtuaalinen osoiteavaruus jaettu yhtenäisiin alueisiin, joilla joka sivulla samat suojaukset ja ominaisuudet

- u vrt. segmentti

n Ytimen pienin allokoinnin yksikkö: slab (<< sivu)

- u ei ole tehokasta varata aina kokonaisia sivuja
- u vain ytimen käyttöön, joka tarvitsee paljon pieniä muistialueita

Linux: sivutus

n Laitteistoriippumaton toteutus

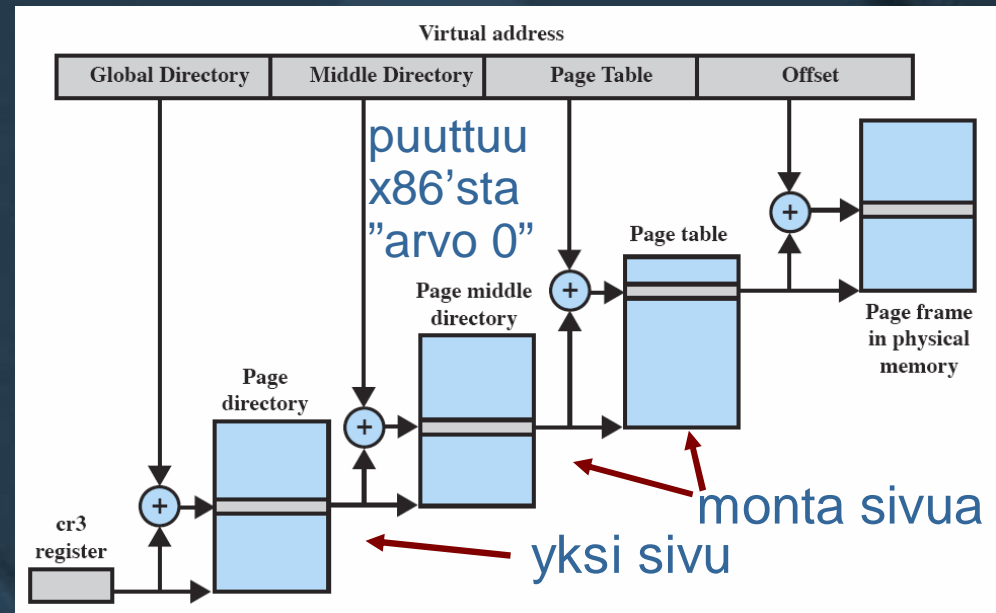
- u Alpha: 64b osoitteet, tuki 3:lle tasolle, sivu 8KB
F offset 13 bittiä
- u x86: 32b osoitteet, käyttää 2-tasoa, sivu 4KB
F offset 12 bittiä

n 3-tasoinen sivutaulu

- u page directory (PD), 1 sivu
- u page middle directory (PMD), monta sivua
- u page table (PT), monta sivua

n Ylin taso aina muistissa

- u säikeen vaihdossa fyysinen osoite MMU:hun (PD)
- u muut voivat olla tukimuistissa!



(Fig 8.25 [Stal05])



Linux: osoitemuunnos



n Jaa osoite 4:ään osaan

- u indeksi ylimmän tason hakemistoon (PD-offset)
- u indeksi välitason hakemistoon (PMD-offset)
- u indeksi sivutauluun (PT-offset)
- u siirtymä sivun sisällä (offset)

```
PMD_base = PD_base + PD[PD_offset]
PT_base  = PMD_base + PMD[PMD_offset]
Page_base = PT_base  + PT[PT_offset]
fyys.os  = Page-base + offset
```

n X86-jippo

- u välitason hakemiston (PMD) koko vain yksi alkio!
- u laitteisto tulkitsee ylimmän tason (PD) alkion osoittavan suoraan alimman tason sivutauluun (PT)
- u sopii muillekin laitteistoille, joissa vain 2 tasoa

Linux: varausalgoritmi

- n **Varaa yhtenäinen lohko (area, region) peräkkäisille sivuille**
 - u tehostaa levyltä noutoa ja takaisin kirjoitusta
 - u optimoi siis levyn käyttöaika (tilan kustannuksella)
- n **Buddy System: 1, 2, 4, 8, 16 tai 32 sivutilaa**
 - u toteutus: taulukko, jossa osoittimet koon mukaisiin vapaiden listoihin
 - u paljon sisäistä pirstoutumista
 - F tarvitset 18 sivua, mutta varaat 32 sivua!



Linux: noutoalgoritmi

n Tarvenouto

- u ei ennaltanoutoa, ei käyttäjien algoritmeja

n Toisaalta, lohkon usea sivu voidaan ehkä hakea kerralla (buddy system)

- u ennakova nouto?

- F ei viitepaikallisuuden vaan levypaikallisuuden mukaan (alueellinen paikallisuus)

- F toivon mukaan lähellä toisiaan

Linux: korvausalgoritmi

- n **Globaali Clock M-bitillä + eräänlainen LRU**
 - u prosessilla dynaaminen varattujen sivutilojen lkm
- n **8 bitin ikälaskuri (age)** vrt. U-bitti
 - u MMU kasvattaa, kun sivuun viitataan
 - u taustaprosessi (kswapd) tutkii sekunnin välein
 - F vähentää 1 ikälaskurista jokaisella tutkintakerralla
 - F jos vapaana liian vähän, vapauttaa sivutiloja
- n **Mitä suurempi ikälaskuri, sitä tiuhempaan sivuun viitattu**
- n **Jos ikälaskuri = 0, sivuun ei ole viitattu**
- n **Korvaa se, jonka laskuri pienin**
- n **Käyttää sivujen puskurointia (pagebuffer)**



Linux memory manager

260 * For choosing which pages to swap out, inode pages carry a
261 * `PG_referenced` bit, which is set any time the system accesses
262 * that page through the (mapping,index) hash table. This referenced
263 * bit, together with the referenced bit in the page tables, is used
264 * to manipulate page->age and move the page across the active,
265 * `inactive_dirty` and `inactive_clean` lists.

266 *

267 * Note that the referenced bit, the page->lru list_head and the
268 * active, inactive_dirty and inactive_clean lists are protected by
269 * the pagemap_lru lock, and *NOT* by the usual PG_locked bit!

(include/linux/mm.h)

click!

Linux daemon *kswapd*

[Stal 05]

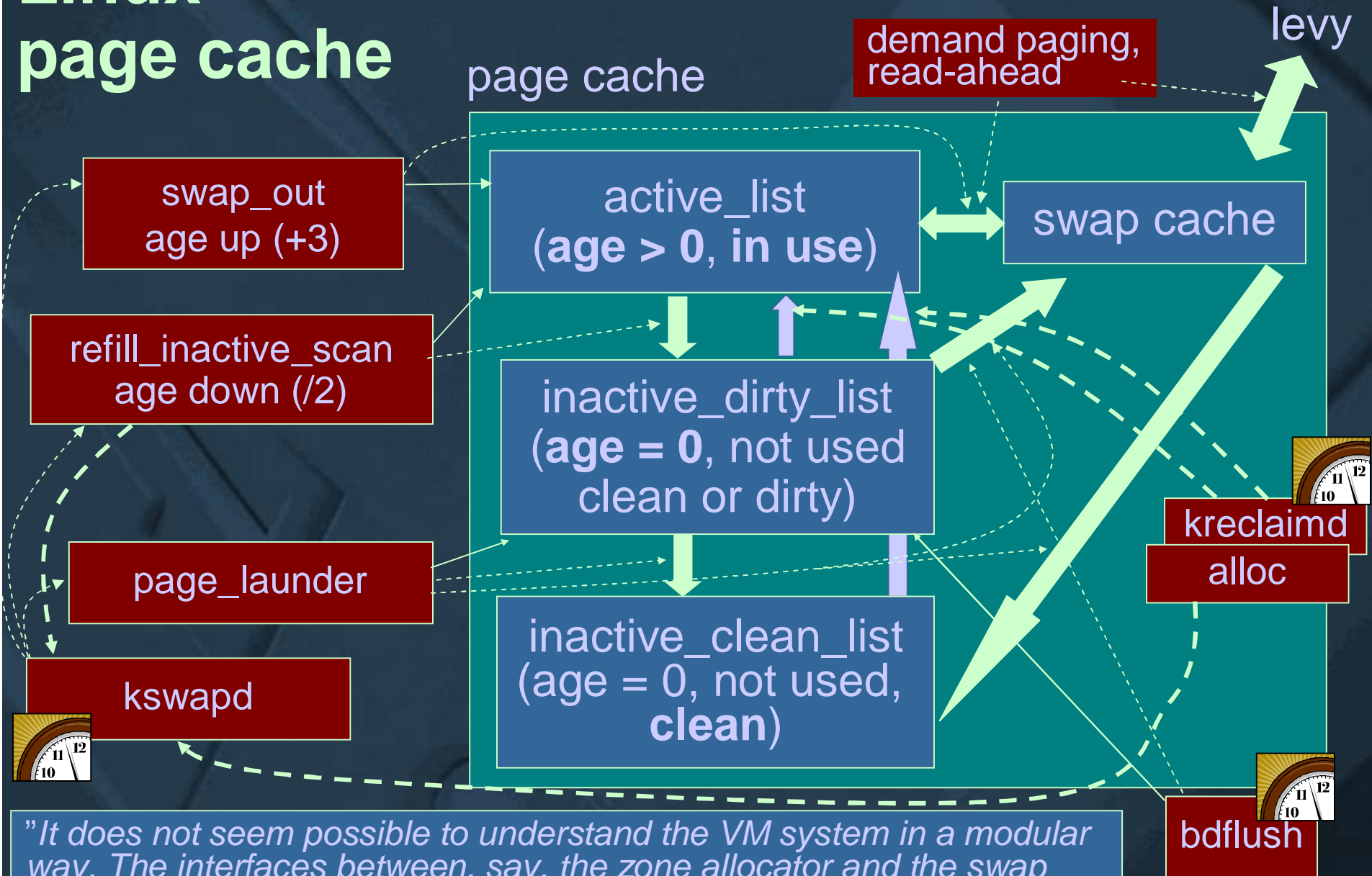
- n Suorittaa kerran sekunnissa. Jos tarvitaan lisää vapaita sivukehyksiä, niin käy kaikki käytössä olevat sivut läpi, max 6 kierrosta, kunnes tarpeeksi vapaita
 - u etsi käyttämättömiä sivuja modifioidulla clock'illa sivupuskurista (page buffer) ja tiedostovälimuistista
 - u etsi yhteiskäytössä olevia käyttämättömiä sivuja
 - u etsi tavallisista käytössä olevista sivuista
 - F käy sivut läpi 1 prosessi kerrallaan
 - F poista käyttämättömät puhtaat heti
 - F pistä käyttämättömät likaiset levyjonoon
 - F pistä käytössä olevat likaiset sivupuskuriin

Linux daemon *bdflush*

- n Herää henkiin aika ajoin, tai explisiittisesti
- n Jos ”liikaa” likaisia sivuja, rupea kirjoittamaan niitä levyille

Linux page cache

<http://home.earthlink.net/~jknappa/linux-mm/vmoutline.html>



"It does not seem possible to understand the VM system in a modular way. The interfaces between, say, the zone allocator and the swap policy are many and varied... What a nightmare" (Joe Knapka)

Linux: tilanvaraus ytimessä

- n Ydin lukittu pysyvästi muistiin
- n Dynaaminen varaus sivuittain
- n Dynaamisesti ladattavat moduulit kokonaisina ytimen muistialueelle
 - u buddy system sivutilatasolla
- n Ytimessä usein tarv `kmalloc()` yös pienille lyhytaikaisille varauksille, slab'eille
 - u yksittäinen sivu leikattavissa pienempiin osiin (slab)
 - u buddy system sivun sisäisesti

☞ x86: 32, 64, 128, 252, 508, 2040, 4080 tavu
(sivu 4KB)



Käyttöjärjestelmät II



Windows 2000

MUISTINHALLINTA

Ch 8.5 [Stal 05]

Ch 11.5 [Tane 01]

W2K

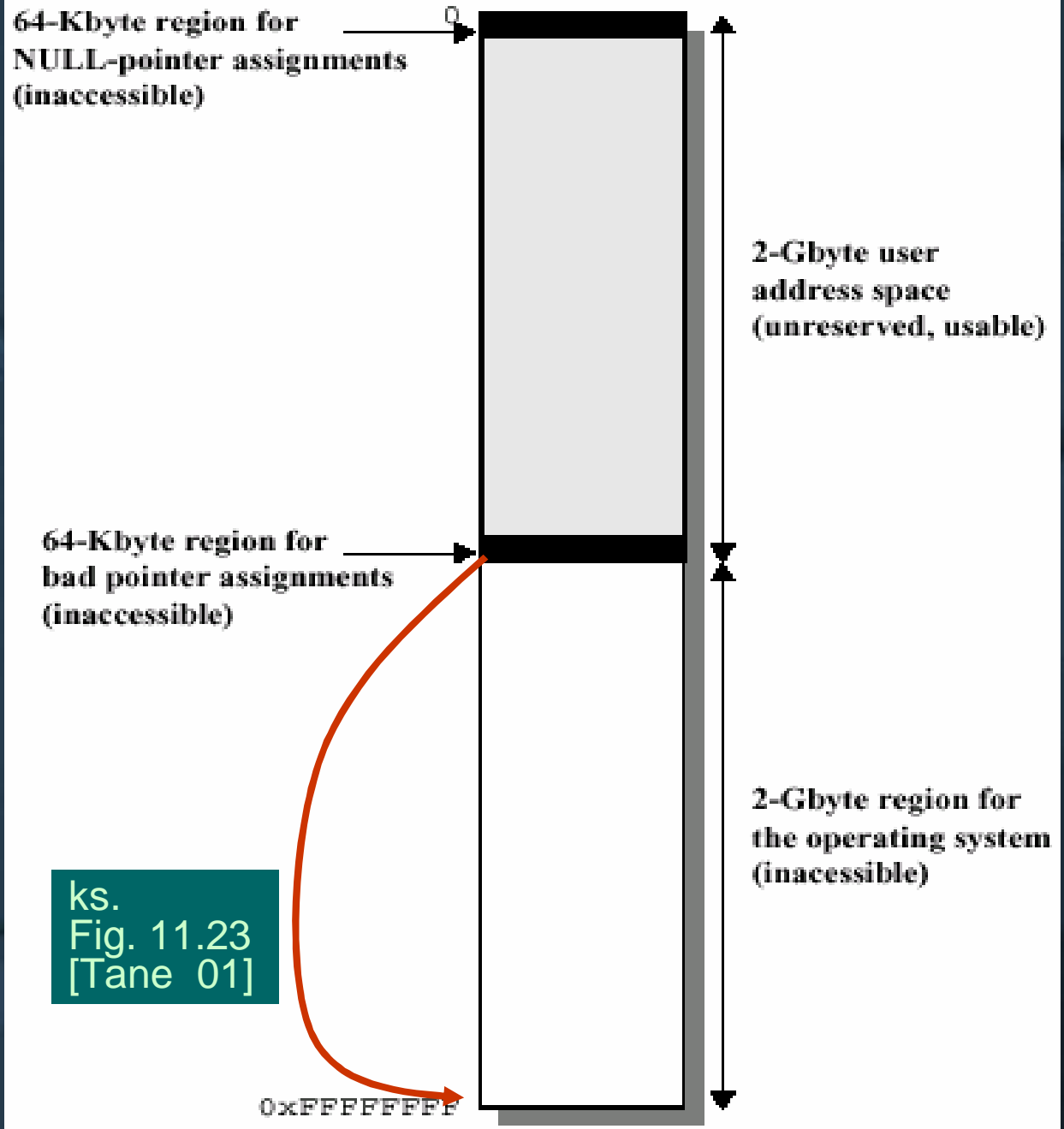
"0x0000????:"

- 32-bittiset osoitteet
=> 4 GB virt. os.
- Sivukoko 4K - 64K
Pentium: 4KB
Alpha: 8K

"-1"

(Fig 8.26 [Stal05])

"0xFFFF????:"



W2K: Sivutus

n Ei segmentointia

n Vaihtuva määrä sivukehyksiä / prosessi

- u aluksi kiinteä määrä (muistin määrän mukaan)
 - F minimi 20-50, maksimi 45-345 (muistin koosta riippuen)
- u jos vapaata paljon, prosessi voi saada lisää
 - F ahneen pitää jättää viimeiset 512 kehystä muille prosesseille
- u jos vapaan muistin määrä vähenee, prosessikohtaista sivutilamäärää vähennetään

n Tarvenouto

- u ei ennaltanoutoa

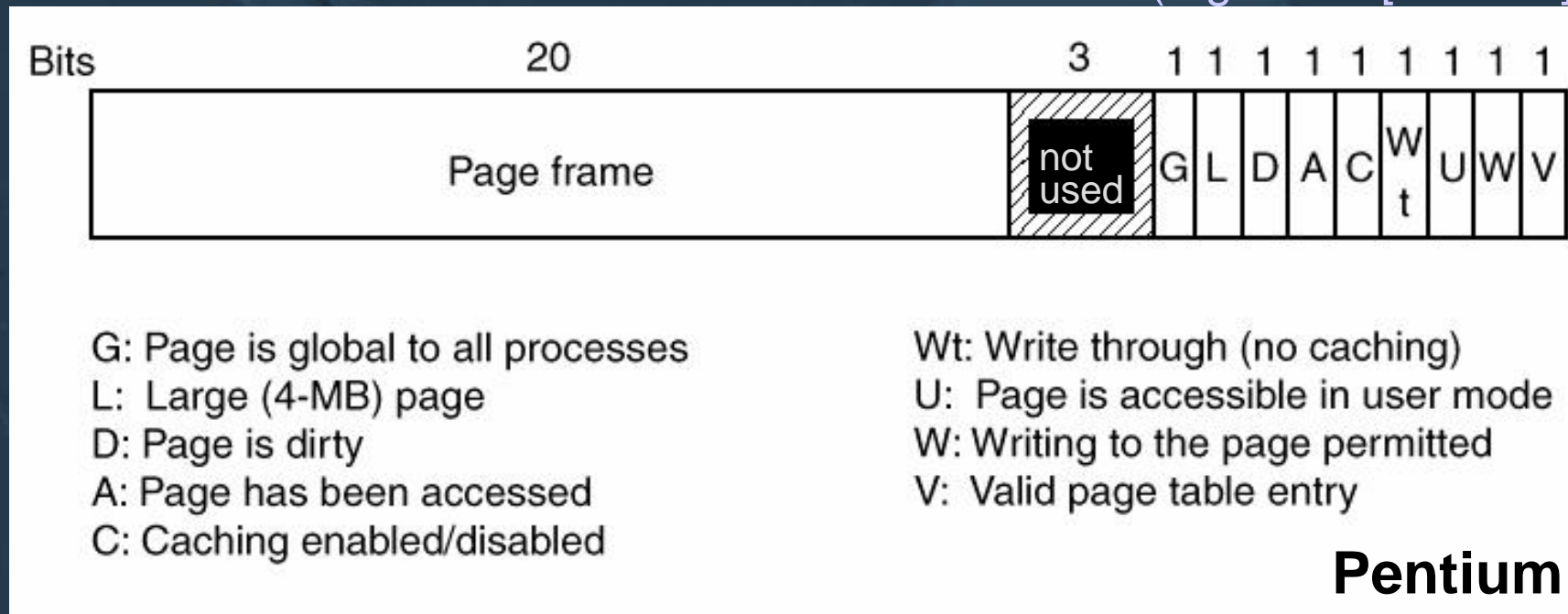
n Levy I/O isompina paloina

- u 1-8 sivua kerrallaan
- u eli siis myös jonkin asteen ennaltanouto!

W2K: Sivutus

n Sivutaulun alkioiden eroja eri järjestelmissä

(Fig 11.26 [Tane01])



n Sivun tilat

u Available

u Reserved:

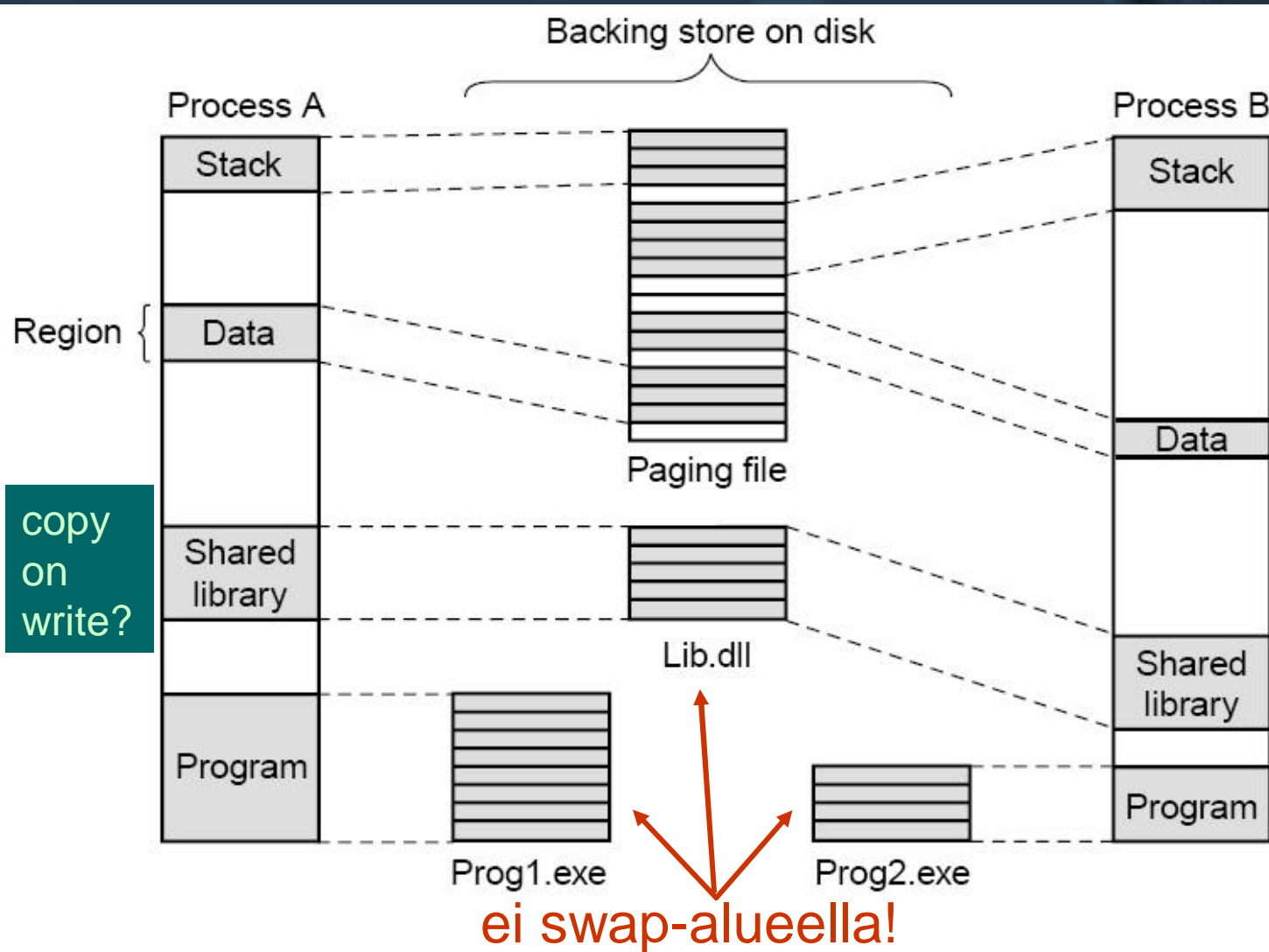
F varaus, jota ei vielä laskettu prosessin osuuteen

F nopea allokointi, kun tarvitaan (esim. pino)

u Committed: käytössä, tila myös tukimuistissa

Fig 11.24 [Tane01]
(seur. kalvo)

W2K osoite- avaruus



[Tane01]

Fig. 11-24. Mapped regions with their shadow pages on disk. The *lib.dll* file is mapped into two address spaces at the same time.

W2K: Korvausalgoritmi

n Käyttäjoukkoalgoritmi, lokaali korvaus

- u käyttäjoukko per prosessi, ei per säie

n Balance set manager -daemon

- u tutkii 1 sek välein, onko riittävästi vapaita sivutiloja

n Working set manager

- u käynnistyy tarvittaessa
- u käyttäjoukon koko määräytyy dynaamisesti
- u vapauttaa prosessikohtaisia sivutiloja
- u aloittaa isoista idle-prosesseista

n Swapper-daemon

- u ajetaan 4 sek välein: etsi prosesseja, joiden kaikki säikeet olleet passiivisia kauan aikaa

n Osa KJ:sta ja puskuriallas lukittu muistiin

n Vapautettujen sivujen puskuointi

W2K Poistettavan sivun valinta

n Joka sivulla laskuri – kauanko sivu käyttämättä

n Yksi suoritin

vrt. Linux Age

- u jos ei viitettä, lisää laskuria
- u jos viite, nollaa laskuri
- u modifioitu Clock?

n Monta suoritinta

- u viitebitit eivät toimi, koska lokaaleja suorittimelle!
- u FIFO-järjestys (onko hyvä? nyt yleinen tapaus!)

n Neljä ”vapaiden sivujen” listaa

- u dirty, clean, free, zeroed free (ja bad)

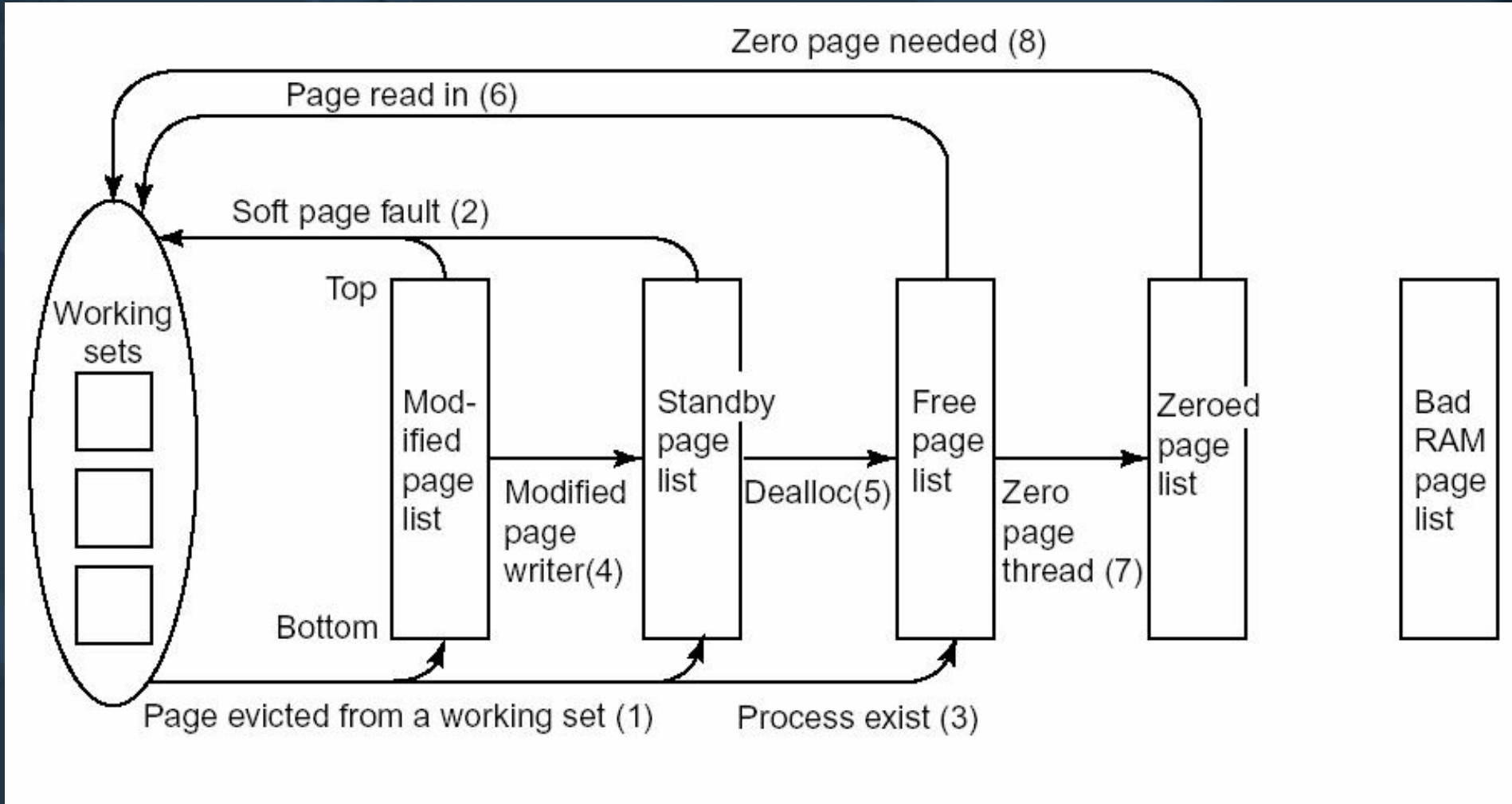
n Kukin fyysinen sivu

- u jossakin käyttöjoukossa
- u jossakin vapaiden sivujen joukossa

F silti prosessin käytettävissä?

Fig 11.27 [Tane01]
(seur. kalvo)

W2K: Vapaat sivutilat



(Fig 11.27 [Tane01])

W2K

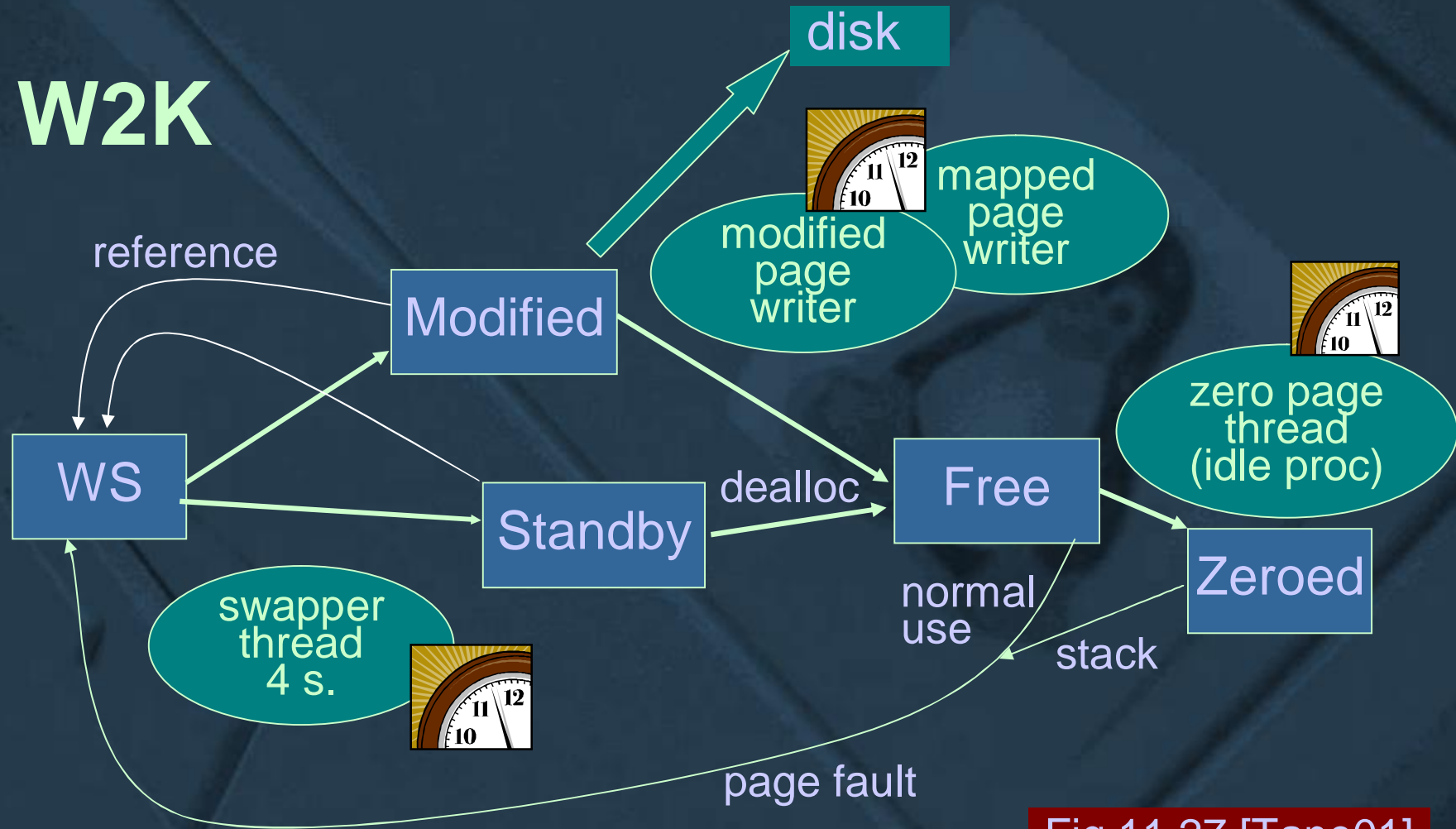


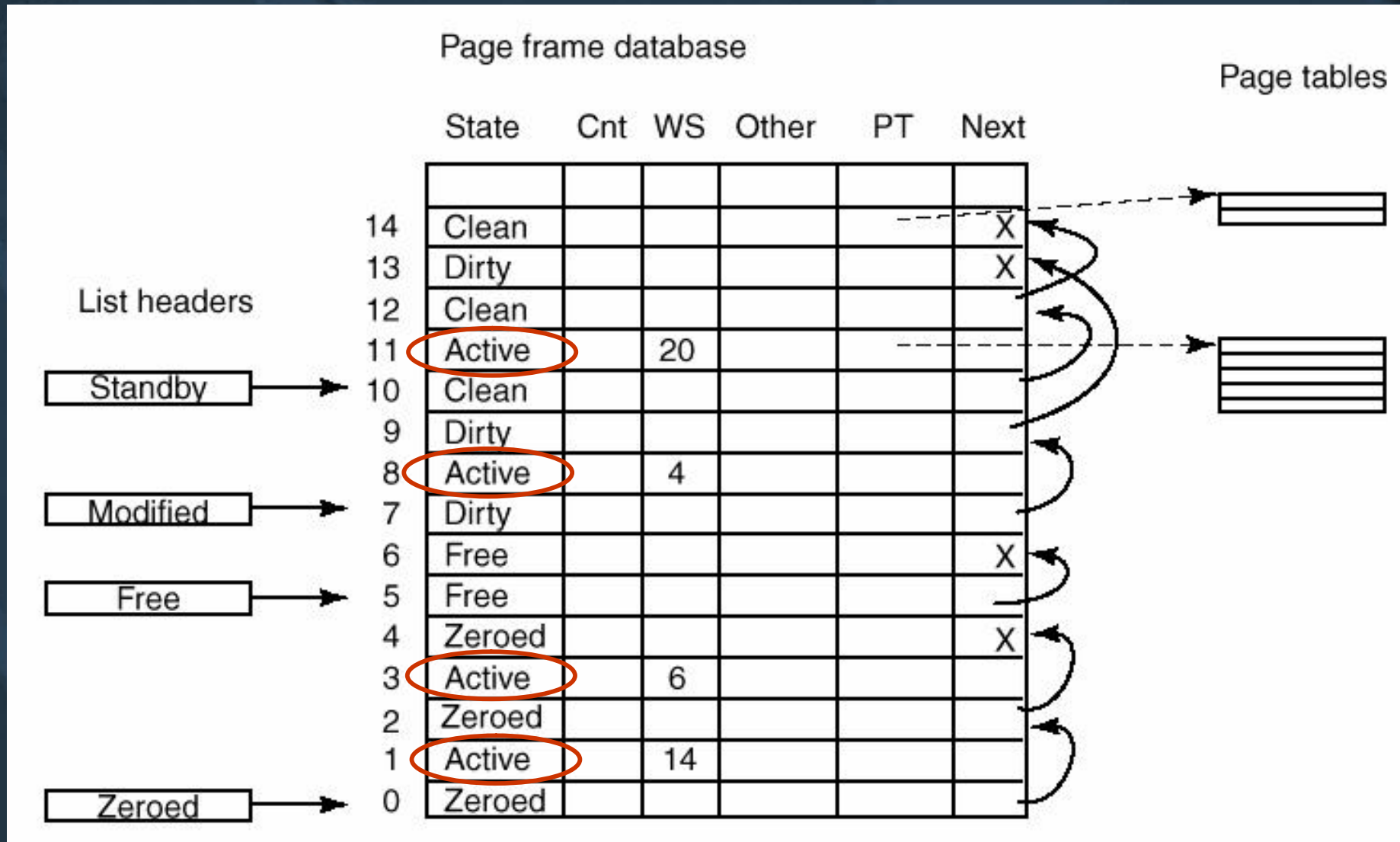
Fig 11.27 [Tane01]
(edell. kalvo)

Fig 11.28 [Tane01]
(seur. kalvo)

"The code is so complex that the designers loathe to touch parts of it for fear of breaking something somewhere"

Andrew Tanenbaum [Tane01, p. 823]

W2K: Sivutilataulu



(Fig 11.28 [Tane01])

Kertauskysymyksiä

- n Mitä eroa on sivutaululla ja sivutilataululla?
- n Miksi paikallisuuden käsite on keskeinen muistinhallinnassa?
- n Mitä tarkoitetaan käyttöjoukolla?
- n Mitä tarkoitetaan käyttöjoukon ikkunakoolla?
- n Mitkä kaksi noutopolitiikkaa?
- n Mitä hyötyä sivupuskuroinnista?
- n Miten globaali ja lokaali korvauspolitiikka eroavat toisistaan?
- n Miten globaali Clock-algoritmi toimii?
- n Suurimmat erot Linuxin ja W2K:n muistinhallinnalla?