

① UFS, i-Node 12 řetězích, 1x jed, dvoj, troj Blok = 8KiB odkazy 64b

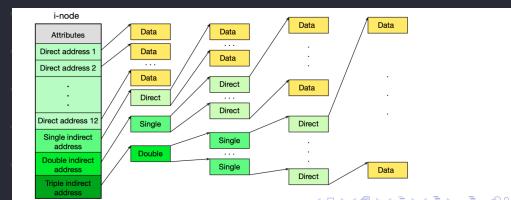
a) Lze připojit k 32-bit systému? Ano (vidy lze)

b) Max. velikost souboru?

$$\text{Odkazy v jednom bloku} = \text{Blok} / \text{velikost odkazu}$$

$$= 8192 / 8 = 1024 \text{B}$$

↑ ↑
3KiB 64b



$$\text{Počet odkazujících bloků} = 12 + 1024 + 1024^1 + 1024^2 = 1074791436$$

↑ ↑ ↑ ↑
Plné 1. úrovň 2. úrovň 3. úrovň

$$\text{Max. velikost} = \text{poč. odkaz. bloků} * \text{velikost jednoho} = 1074791436 * 8192 = 8.304 \text{TB}$$

$$c) \text{Max. velikost FS} = 2^{\text{velikost odkazu}} * \text{velikost Bloku} = 2^{64} * 8192 = 128 \text{TiB}$$

d) Nejvýšší max. vel. souboru?

odkazy 4, 5, 6 ... úrovni

e) Co vše musíme přečíst, pokud chceme v /aa/bb/cc.bin najít offsetu 1234567890

obsah Adressic /

I-Node adresic /aa

obsah Adressic /aa

I-Node adresic /aa/bb

obsah Adressic /aa/bb

I-Node souboru /aa/bb/cc.bin

Být 1234567890 je v dat. bloku 2. úrovni položka

čtení 2 nejdůležitějších odkazů

čtení dostupného bloku

$$\text{Blok} = 1234567890 / 8192 = 15704$$

$$12 + 1024 \leq 15704 \leq 12 + 1024 + 1024^2$$

f) Co vše musíme přečíst při vytvoření sym linku v /aa/bb/cc.link odkazující na /aa/dd/ec/ff/GG/hh.txt
obsah /

i-nodec /aa

obsah /aa

:

Zápis i-node linku /aa/bb/cc.link

Zápis diskového bloku linku (" /aa/dd/ec/ff/GG/hh.txt ")

Zápis obsahu adresácie /aa/bb

cylinder povrch sektor

(2) CHS geometrie 16384/255/63, velikost bloku 512B (př. na HDD)

a) Kapacita = počet cylinderů * počet stran na cylinderu * počet sektorů na stranu * velikost bloku
 $= 16384 * 255 * 63 * 512 = 134 \text{ GB}$

b) LBA adresy pro [186, 24, 15]

- počítá se to od vnitřku

Přeskočím 186 cylinderů $\Rightarrow 186 + 255 * 63$

24 povrchů $\Rightarrow 24 * 63$ +

15 sektory $\Rightarrow (15 - 1)$ +

↗

indexuje se od 1

\Rightarrow sečtene $\Rightarrow 2989614$

c) CHS sektor 1234567 $\downarrow 1234567 \% (255 * 63)$

Celký cylinder = 1234567 / 255 * 63 = 76 128 bytec 15627

Celký povrchy = 15627 / 63 = 216 128 bytec 19

Celk. sektory = 19 + 1 (indexováno od 1)

$\Rightarrow [76, 216, 19]$

d) Zápis [75, 0, 1], konec [300, 254, 63] jdeš je užitost

=> 225 cylindrů, 254 pozice, 62 sektory

$$=> 225 * 255 * 63 + 254 * 63 + 62 = 1,35 \text{ GB}$$

$$\text{hcho } 1\text{c } LBA([300, 254, 63]) - LBA([75, 0, 1]) = 4835564 - 1204875 + 1 = 1,35 \text{ GB}$$

(3) HDD, 1TB, 5000 stop, 15000 rpm, Hlavníky jsou hod.stupeň 1000

Fronts pořadatel: [1700, 2600, 1500, 800, 4200] Rychlosť hlavník: 1000 stop z, 3 ms

$$=> jedna stopa = 3 / 1000 = 0.003 \text{ ms}$$

a) odbočení při FIFO

$$1000 \rightarrow 1700 \rightarrow 2600 \rightarrow 1500 \rightarrow 800 \rightarrow 4200$$

$$700 + 1100 + 1300 + 700 + 3400 = 7200 \quad (\text{jedná se o pozici})$$

$$\tilde{C}_{OS} = \text{počet stop} * \text{trvaní jedné stopy} = 7200 * 0.003 = 21,6 \text{ ms}$$

b) SCAN (Elevator) - základní směrem hore (beru pořadatel v jednom směru, daješ akum. ho dolů)

$$1000 \rightarrow 1500 \rightarrow 1700 \rightarrow 2600 \rightarrow 4200 \rightarrow 800$$

$$500 + 200 + 1100 + 1400 + 3400 = 6600$$

$$\tilde{C}_{OS} = 6600 * 0.003 = 19,8 \text{ ms}$$

c) SSTF (Shortest seek Time First)

Fronts pořadatel: [1700, 2600, 1500, 800, 4200]

$$1000 \rightarrow 800 \rightarrow 1500 \rightarrow 1700 \rightarrow 2600 \rightarrow 4200$$

$$200 + 700 + 200 + 1100 + 1400 = 3600$$

$$\tilde{C}_{OS} = 3600 * 0.003 = 10,8 \text{ ms}$$

- Pro každou řešení musíme počítat průměrné počet otáček když lze dát do společného sektoru

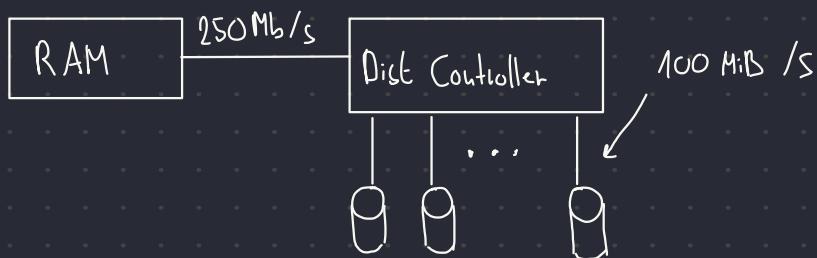
$$15000 \text{ rpm} = 250 \text{ rps} \quad 1 \text{ Točto} = \frac{1}{250} = 0.004 = 4 \text{ ms} \Rightarrow \text{Počet Točto} 2 \text{ ms}, 5 \text{ cíle} \Rightarrow +10 \text{ ms}$$

Plo okamžity (sokoj) RPM → RPS

$$15000 \text{ rpm} = 15000 \text{ r/m} = \frac{15000 \text{ r}}{\text{m}} = \frac{15000 \text{ r}}{60 \text{ s}} = 250 \text{ r/s} = 250 \text{ RPS}$$
$$\rightarrow \text{m} = 60 \text{ s}$$

(4) RAID $10 \times 1\text{TiB}$ disků, SW RAID, čtení/zápis 100mb/s , propustnost shémuje 250mb/s

- Plo kódová konfiguraci:
 - a) kapacita
 - b) adomost
 - c) vychlast čtení/zápis
 - d) obnovu při 1 řešení disku



RAID 0 (JBOD)

- | | |
|-----------------------|---|
| a) 10TiB | RAID 0 (Striping) |
| b) 0 | a) 10TiB |
| c) 100MiB/s | b) 0 |
| d) ∞ | c) můžu zápisovat do 10 nožek, ohrozí je jen shéma $\Rightarrow 250 \text{MiB/s}$ |
| | d) ∞ |

RAID 1 (5x mirror)

- a) 5TiB
- b) 1 až 5
- c) čtení paralelně od 200MiB/s , zápis 100MiB/s
- d) kopírování, zápis shémuje $\Rightarrow 100 \text{MiB/s}$ cílem $1\text{TiB} \rightarrow 1\text{TiB} / 100 \text{MiB/s} = 2^{20} / 100 = 10485,76 \text{s}$

RAID 1e (3 mirror, 1x spare)

- a) 3TiB
- b) 2 až 6-7 (spare)
- c) čtení 250MB/s , zápis Potřebují psát 300 shémuje 250 , $3 \times$ ty samé daty $250/3 = 83.33 \text{ MiB}$
- d) čtu ze 2 a zapisuju do 1 \Rightarrow pouze $100\text{MiB/s} \Rightarrow 1\text{TiB} / 100\text{MiB/s} \Rightarrow 10485.76 \text{s}$

RAID 5

a) 9 TiB (jeden na parity)

b) 1

c) čtení z 9 disků, omezeno sháními $\Rightarrow 250 \text{ MiB/s}$, zápis - 9/10 disků 1/10 parity

$$\Rightarrow \text{kyklast} = 9/10 \times 250 = 225 \text{ MiB/s}$$

d) shání 250 MiB/s \Rightarrow aktuální 1 zápis $\Rightarrow 25 \text{ MiB/s}$

$$\text{Celkově } 1 \text{ TiB} / 25 \text{ MiB/s} = 41943 \text{ s}$$

RAID 6

a) 8 TiB (2 parity disků)

b) 2

c) čtení 250 MB/s, zápis 8/10 využitých, 2/10 parity $\Rightarrow \frac{8}{10} \cdot 250 = 200 \text{ MiB}$

d) obnovovat daty: je pro každou parity pauza 8 disků \Rightarrow číst 1 zápisovat

$$\Rightarrow 250 / 9 = 27,7 \text{ MiB/s}$$

$$\text{Celkově } 1 \text{ TiB} / 27,7 \text{ MiB/s} = 37855 \text{ s}$$

Raid 01 (Striping + mirroring)

a) 5 TiB

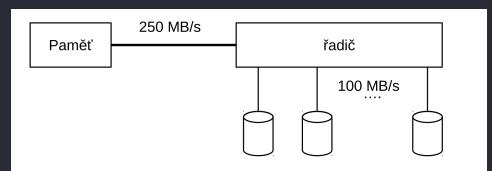
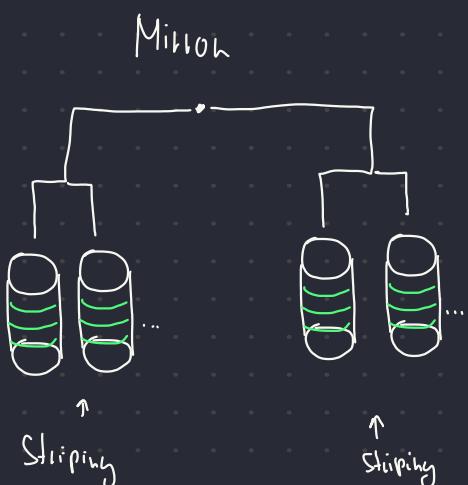
b) 1 až 5

c) čtení 250 MiB/s, zápis 125 MiB/s

$$d) 250 \text{ MiB/s} \Rightarrow 5 \text{ TiB} / 250 \text{ MiB/s} = 41943 \text{ s}$$



- musíme obnovit všechna data v kopiích, číslo by mohlo být mnohem větším!



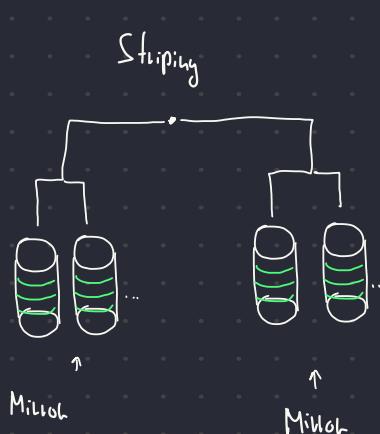
Raid 10 (mirroring + striping)

a) 5 TiB

b) 1 až 5

c) čtení 250 MiB, zápis 125 MiB

$$d) \text{čtení zápisu } 100 \text{ MiB} \Rightarrow 1 \text{ TiB} / 100 \text{ MiB} = 41943 \text{ s}$$



⑤ Spolehlivost RAID $n = 10$ disků, $t = 98\%$, spolehlivost

$$a) \text{pst všechny fungují } (0,98)^{10} = 81,71\%$$

$$b) \text{pst výpadku 1 disku } \binom{10}{1} (0,98)^9 \cdot (0,02) = 16,6\%$$

$$c) \text{pst výpadku 2 disků } \binom{10}{2} (0,98)^8 \cdot (0,02)^2 = \frac{10!}{2! \cdot 8!} \cdot 0,00340305 = \frac{10 \cdot 9}{2} \cdot (...) = 1,53\%$$

$$d) \text{pst výpadku 3 disků } \binom{10}{3} \cdot (0,98)^7 \cdot (0,02)^3 = 0,08\%$$

$$\text{Obecná pst výpadku } X \text{ z } n \text{ disků} = \binom{n}{X} \cdot (t)^{n-X} \cdot (1-t)^X$$

$t = \text{spolehlivost}$

Spolehlivost:

$$R_{\text{ RAID 0}} = \text{pst všechny fungují} = (0,98)^{10} = 81,71\%$$

$$R_{\text{ RAID 5}} = \text{všechny 0k, 1 zlkc} (0,98)^{10} + \binom{10}{1} \cdot (0,98)^9 \cdot (0,02) = 0,817 + 0,1674 = 98,38\%$$

$$R_{\text{ RAID 6}} = \text{všechny 0k, 1 kdo 2 zlkc} (0,98)^{10} + \binom{10}{1} \cdot (0,98)^9 \cdot (0,02) + \binom{10}{2} (0,98)^8 \cdot (0,02)^2 = 99,91\%$$

MTBF (Mean time between failure) - střední doba mezi pochodem

- hpi 750 000 h
- 1000 lidí ve věku 25 let, za jeden rok zemře 5 lidí

$$\text{MTBF} = \frac{\text{time} \cdot \text{count}}{\text{dead}} = \frac{365 \cdot 24 \cdot 1000}{5} = 1752000 \text{ h} = 200 \text{ let}$$

$$\text{pst, že funguje v dali } t = R(t) = e^{-\frac{t}{\text{MTBF}}}$$

AFR (Annualized Failure Rate) - pst že disk selže během jednoho roku

$$\text{AFR} = 1 - R(24 * 365) = 1 - e^{-\frac{24 * 365}{\text{MTBF}}}$$

⑥ OS - struktura 8 KiB, 32b virtuální adresy, 30 bit Fyzické

Jst kódem využitelné tabule pro přechod

19b 13b

Virtuální

	Offset
--	--------

17b 13b

Fyzický

	Offset
--	--------

$$\text{Struktura } 8 \text{ KiB} = 2^3 \cdot 2^{10} = 2^{13} \Rightarrow 13 \text{ bit Offset}$$

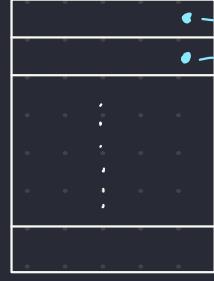
32/64 bit dle typu OS

=> chce se uložit 2 úrovně

$$T_1 = 2^8 = 256 \text{ položek}$$



$$T_2 = 2^{11} \text{ položek}$$



8b 11b 13b

Virtuální

Index ₁	Index ₂	Offset
--------------------	--------------------	--------

$$\text{Struktura } 8 \text{ KiB}$$

17b

Control bits

Cílovo Rámec

↑
Hlavní adresy

↑
Adresy struktury

* u tab. první úrovni bude čisté hrančítko. Je v ní
256 položek každou, velikost každého je jedna struktura 8KiB

$$\text{Počet položek} = 8 \text{ KiB} / 2^{13} = 2048, \text{ použito } 256, \text{ nevyužito } 1792$$

Montáž:

$$\text{Offset} = \text{Hodnota druhovního exponentu vel. struktury } 4 \text{ KiB} = 2^{13} \text{ B} \Rightarrow 13$$

$$2. \text{ úrovni} = \text{Hodnota druhovního exponentu Vel. struktury} / \text{velikost položky} = 4 \text{ KiB} / 32 \Rightarrow 2^{11} \Rightarrow 11$$

32/64 B podle OS kterou má adresu

$$1. \text{ úrovni} = 32 / 64 - \text{Offset} - 2. \text{ úrovni} = 32 - 13 - 11 = 8$$

↑

Typ OS / virt. adresy

8b

11b

13b

1. úr	2. úr	Offset
-------	-------	--------

(7) Standard 8KiB, virt. address 32 bit, fiz. adrs 30 bit

- Proces powinie posiadać 2 MB pamięci przeznaczonej do danych, co pozwoli na zapisanie 1 MB struktury. Jako że struktura jest skojarzona z danymi, to pojęcie struktury nie ma sensu.

a) $21 \text{ MiB} \rightarrow 21 \text{ MiB} / 8 \text{ kB} = 2628$ Stück pro 8 kB

$\text{offset} = 13 \text{ b}$, 2. úton 11 b , 1 úton 8 b

- jedna struktura adresująca strukturę mówiącą o 2048 pozycjach \Rightarrow powinno być 2688 struktur potocznych dla położenia 2.

b) 1MiB zapisoblikt \Rightarrow $1\text{MiB} / 8\text{KiB} = 128$ struktur po 8KiB \Rightarrow 1 struktura adresu struktury

c) Zadlu stísnět plochou sdušoví

$$\text{Sticky plot dots} = 2688 + 128$$

$$\text{Stísky plo kčii} = 2+1+1$$

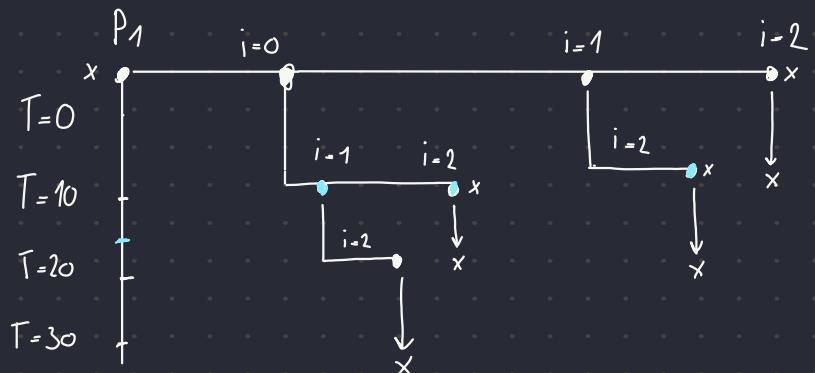
\uparrow \uparrow \uparrow
 a b c

8

V čase $t=0$ byl spuštěn následující program:

```
int main ( void )
{
    pid_t ch;
    int i, staus;

    for ( i = 0; i < 3; i ++ )
    {
        ch = fork ();
        if ( ch == 0 ) sleep ( 10 );
    }
    wait ( &status );
    return ( 0 );
}
```



a) kolit buck process culture = point = 7

b) Učenici prvního stupně hrají

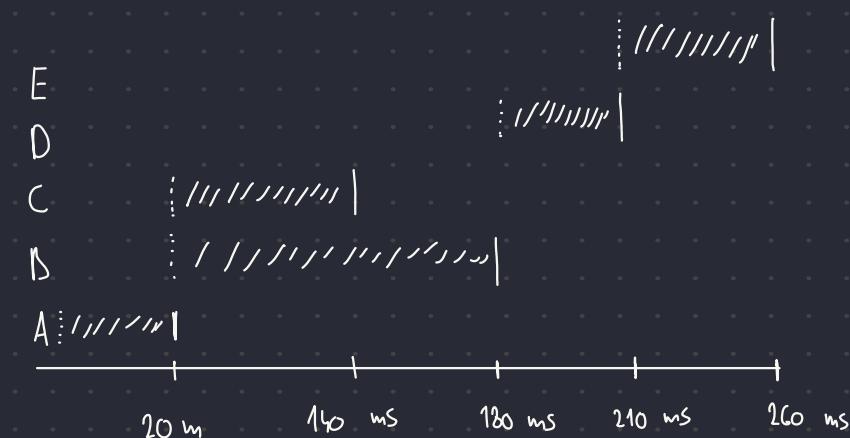
$$c) \text{ kohlenstoff } + 15 = 3$$

d) kdy se utoní poslední? $T = 30$

Příklady na procesy

OS používá statické priority pro plánování procesů. Má k dispozici jedno CPU bez hyperthreadingu. Časové kvantum je 10ms, režie přepnutí kontextu je zanedbatelná. Téměř ve stejný okamžik jsou spuštěné procesy A, B, C, D a E s dobou používání CPU podle tabulky níže. Pro každý proces určete, kdy dojde.

Proces	A	B	C	D	E
Statická prioritá	10	8	8	4	2
CPU time	20	100	60	30	50

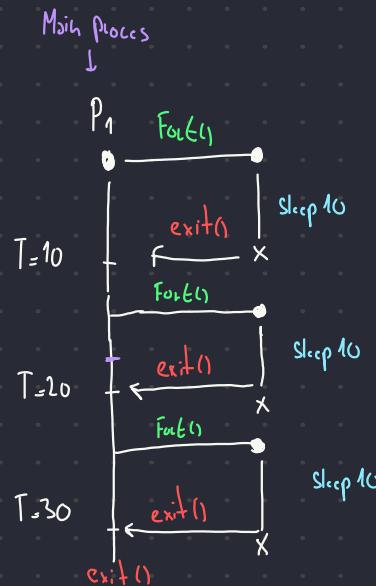


	Start	End
A	0	20
B	20	120
C	20	80
D	20	110
E	20	100

9)

Následující kód zkompilujeme a spustíme v čase t .

```
int main( int argc, char * argv [] ) {
...
n = 3;
for ( int i = 0; i < n; i ++ ) {
    child_pid = fork ();
    if ( child_pid == 0 ) {
        execvp("sleep", "sleep", "10", (char *) NULL);
    }
    else { wait(&status); }
}
exit(0);
}
```



- a) Kolké procesy vzniknou? $3 + 1 = 4$ (exec mohlo být ve VAS a žežku ho vykonávat)
- b) Kdy se ukončí první? Po 10 sekundách
- c) Kolké procesy počítí v $T=15$? 2 main + jeden child
- d) Kdy se ukončí poslední? v $T=30$ main process

(10)

Plánování výstav

Čas Caché procesu! NE doba finální výstav

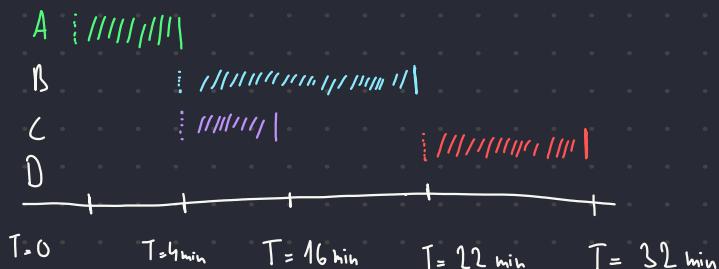
Proces	Počet vláken	Priorita vláken	CPU čas sek.výpočtu [minuty]
A	1	90	4
B	3	60	15
C	1	60	3
D	2	10	10

Časové členění = 20 ms

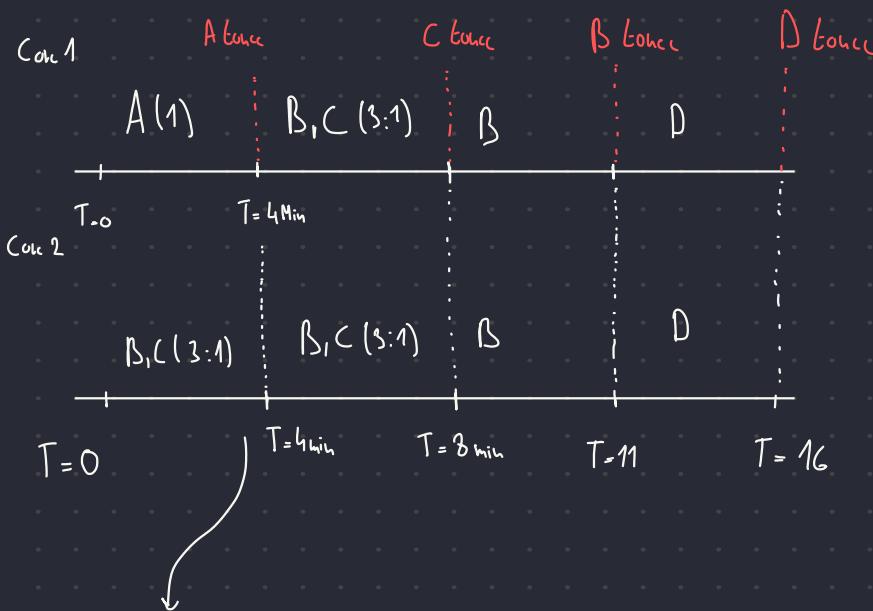
viz počet vláken
↓

a) Zjednodušené výstav skončí na 1 jádru

B,C se budou střídat v poměru 3:1

Po 12 min bude z B pryž 9 min a z C 3
⇒ C skončí z B 2 bude 6 min

b) Zjednodušené výstav skončí na 2 jádrách



v T=4 A Hotovo, C = 3-1, B = 15-3 = 12

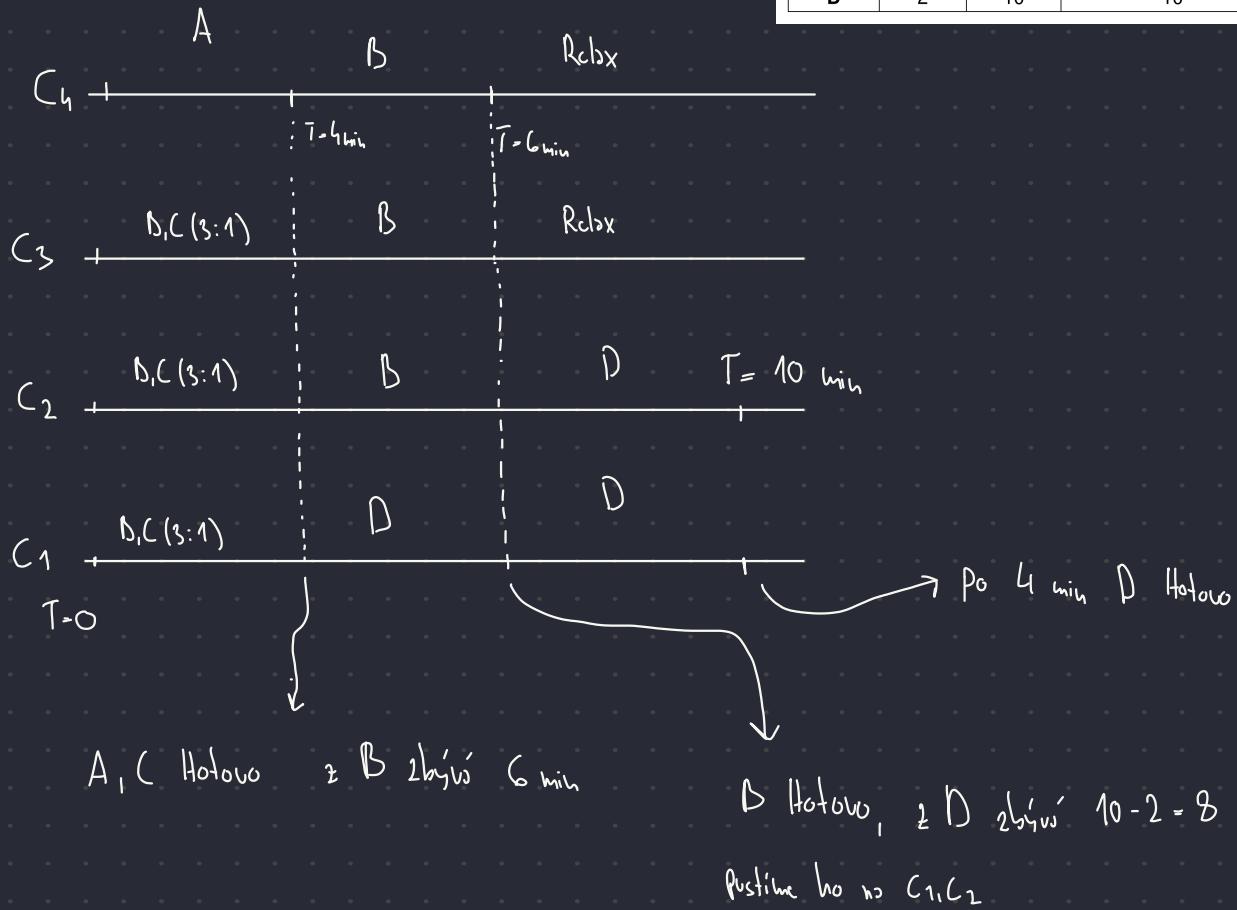
v T=8 C Hotovo, B = 12-6 = 6 min

v T=11 B Hotovo, nasadím D

v T=16 D Hotovo

c) 2, jde o dlouho stojící na 4 vláknech

Proces	Počet vláken	Priorita vláken	CPU čas sek.výpočtu [minuty]
A	1	90	4
B	3	60	15
C	1	60	3
D	2	10	10



Doddlock - Bentéjev algoritmus

- matice R = požadavky vláken
- matice A = přidělkové prostředky
- vektor F = aktuální volné prostředky

$$R = \begin{bmatrix} 3 & 3 & 2 & 0 \\ 1 & 4 & 0 & 5 \\ 2 & 0 & 2 & 1 \\ 1 & 2 & 1 & 0 \\ 6 & 1 & 1 & 7 \end{bmatrix}$$

$$A = \begin{bmatrix} 2 & 0 & 1 & 0 \\ 1 & 2 & 0 & 5 \\ 2 & 0 & 2 & 0 \\ 1 & 1 & 0 & 0 \\ 4 & 1 & 1 & 2 \end{bmatrix}$$

$$F = \begin{bmatrix} P_1 & P_2 & P_3 & P_4 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

$P_1 \ P_2 \ P_3 \ P_4$ Uspořejení

tsdby = procesy

sloupcy = prostředky

Zj. systém v horizontu sloupců?

$$M = R - A = \begin{bmatrix} 1 & 3 & 1 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 2 & 0 & 0 & 5 \end{bmatrix} \begin{array}{l} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}$$

Uspořájme 3, $F_{free} = [2, 0, 2, 5]$ (pořízené co se uvalilo z R)

5 $F_{free} = [6, 1, 3, 7]$

4 $F_{free} = [7, 2, 3, 7]$

2 $F_{free} = [8, 4, 3, 12]$

1 $F_{free} = [10, 4, 4, 12]$

(11) Strukčovský + virtuální přístup (1. čtvrtový Tobulka)

a) Kdo předkládá Virtual → Fyzický MMU

b) Co se při tom používá? Page Table, TLB



c)

Popis systému

- Mějme systém s 32-bitovým CPU, který podporuje virtuální stránkovou paměť s 4KB stránkami a používá klasickou tabulkou stránek.
- Na systému je nainstalovaný 32-bitový OS.
- 32-bitový proces A má na nejnižších adresách VAS alokováno 8MB pro kód a haldu, na nejvyšších adresách 16MB pro zásobníky a běží v něm 2 vlákna.

Otázky

- Kolik tabulek si musí OS udržovat, pokud je právě spuštěno celkem m procesů a n vláken?
- Jaká je struktura logické adresy včetně velikosti?
- Jaká je struktura fyzické adresy včetně velikosti?
- Kolik řádek bude mít tabulka?
- Jaké informace jsou uloženy na řádce a kolik zabírájí místa?
- Kolik místa zabírá celá tabulka pro nás proces A?

1) Přet přes \rightarrow m

$$\text{Offset} = 4 \text{ kB} = 2^{12}$$

20 b 12 b



4) Počet řádců = 2^{20} jeho řádků pro stránku

5) Číslo stránky + present, dirty, ... Zajistitelného míst v řádu $32b = 4B$

6) $2^{20} \cdot 4 = 4 \text{ MiB}$

\uparrow \uparrow
počet řádců velikost řádky

12

Stříhaný + virtuální počet (2 úrovní tabulek)

Popis systému

- Máme systém s 32-bitovým CPU, který podporuje virtuální stránkovou paměť s 4KB stránkami a používá dvouúrovňovou tabulku stránek (stejný počet bitů pro indexy do první i druhé úrovně).
- Na systému je nainstalovaný 32-bitový OS.
- 32-bitový proces A má na nejnižších adresách VAS alokováno 8MB pro kód a haldu, na nejvyšších adresách 16MB pro zásobníky a běží v něm 2 vlákna.

Otázky

- Kolik tabulek si musí OS udržovat, pokud je právě spuštěno celkem m procesů a n vláken?
- Jaká je struktura logické adresy včetně velikosti?
- Jaká je struktura fyzické adresy včetně velikosti?
- Kolik tabulek si OS musí udržovat v nejhorší případě?
- Kolik řádek budou mít tyto tabulky?
- Jaké informace jsou uloženy na řádkách těchto tabulek a kolik zabírají místa?
- Kolik míst zabírají tabulky pro náš proces A?

$$\text{Offset} = 4 \text{KiB} - 2^{12} \Rightarrow 12$$

1) pro proces, má v součtu odrazy w dolní podle využití VAS

16b 10b 12b



20b

12b



4) počet tabulek v nejhorším případě?

$$- 1 + 2^{10}$$

↑ ↑

Hlavní 2. úrovni

$$5) \text{Hlavní } 2^{10} \text{ a } 2. \text{ úrovni } 2^{10}$$

6) 1. úl. Adresa dolní tab. present, dirty ... (32b)

2. úl. číslo řádku, present, dirty ... (32b)

7)

$$1 \text{úrovni} = 4 \text{KiB}$$

$$2 \text{úrovni} = (2+1) \cdot 4 \text{KiB}$$

$$2 \text{MiB} \quad \uparrow \quad \uparrow \quad 16 \text{MiB}$$

Memory

Počet str 2.úl - (dota / stisk), / počet řádků 2. úrovni

$$\left[(8 \text{MiB} / 4 \text{KiB}) / 2^{10} \right] =$$

$$2 \text{MiB} / 4 \text{KiB} = 2^{11} \text{ stránek}$$

\Rightarrow Tabulka 2 úrovni 2^{10} řádků \Rightarrow patřícíme 2

$$16 \text{MiB} / 4 \text{KiB} = 2^{12} \Rightarrow 4$$

(12) Stránkování + virtuální paměť (Inverovaná Tabulka)

Popis systému

- Mějme systém s 32-bitovým CPU, který podporuje virtuální stránkovovanou paměť s 4KB stránkami a používá invertovanou tabulkou stránek.
- Na systému je nainstalovaný 32-bitový OS.
- 32-bitový proces A má na nejnižších adresách VAS alokováno 8MB pro kód a haldu, na nejvyšších adresách 16MB pro zásobníky a běží v něm 2 vlákna.

Oázky

- ① Kolik tabulek si musí OS udržovat, pokud je právě spuštěno celkem m procesů a n vláken?
- ② Jaká je struktura logické adresy včetně velikostí?
- ③ Jaká je struktura fyzické adresy včetně velikostí?
- ④ Kolik řádek bude mít tabulka?
- ⑤ Jaké informace jsou uloženy na řádce a kolik zabírají místa?
- ⑥ Kolik místa zabírá celá tabulka pro náš proces A?

1) Pouze 1 řádek celé sloučené

20 b 12 b

Číslo	stránky	offset
-------	---------	--------

20 b 12 b

Číslo	Rámce	offset
-------	-------	--------

4) 2^{20} (řádků pro rámce)

5) Číslo rámce, plus rozdíly ... (32 bit)

6) $2^{20} \times 4 = 4 \text{ MiB}$

↑ ↑

Počet řádků: 32b = velikost řádky

K čemu slouží a co obsahuje TLB?

"Cache", která obsahuje informace o stránkách a rámcích posledně překládaných adres.

Kolik má TLB řádek a jaká je zde informace?

Počet řádek je daný hardwarem. Řádka obsahuje: číslo stránky, číslo rámce, ID procesu, ...

Jak se hledá v TLB?

Na základě čísla stránky a ID procesu chceme zjistit číslo rámce, ve kterém je daná stránka nahrána.

Co znamená TLB hit?

Číslo rámce bylo nalezeno v TLB. MMU může rovnou přeložit logickou adresu na fyzickou.

Co znamená TLB miss?

Číslo rámce nebylo nalezeno v TLB. MMU bude hledat v další tabulce/dalších tabulkách.

Co znamená Page fault?

Číslo rámce nebylo nalezeno ani v dalších tabulkách. MMU žádá o pomoc OS.

(13) Page Fault - Optimální

Čas přístupu	0	3	6	8	11	13	15	17	19	21	24	27	30	32	35	37	39	41	43	46	48
Typ přístupu	r	r	r	r	w	r	r	r	w	r	r	r	w	r	r	r	w	r	r	r	w
Číslo stránky	0	5	1	1	9	0	6	1	10	0	5	2	10	1	9	7	9	0	4	10	9

Rámce a, b, c, d

0, 3, 6, 8, 11, 13, 15, 17, 19, 21, 24, 27, 30, 32, 35, 37, 39, 41, 43, 46, 48

A	0				0			0								0	4			
B		5								5	2			9	9					9
C			1	1			1						1		7					
D				9	6	10					10							10		

10x Page Fault (počet x)

- jdu po časech, pokud je stránka možnáho k rámci opisu, jde o podíl na doba výkonu
- které stránky když nejsou uvedeny a přípisu funkce provedou, příčtu Page Fault

(14) Page Fault NRU (Not Recently Used)

R - referenced bit, M - modified bit - při náhlístí stránky nastaveno na 0, 1 pakže k/w

- OS periodicky kontrole
- pokud vylákáme stránku k nahrazení vybereme tu s nízkou hodnotou RM

Číslo stránky	2	3	2	1	5	2	4	5	3	2	5	2
Typ přístupu	r	r	w	r	r	r	r	r	r	w	w	r
Čas	1	4	9	10	15	17	20	21	22	25	27	30

R - bít hostující výzvy, M - párc při write

	1	4	9	10	15	17	20	21	22	25	27	30	31	32	37
a	Students	2		2	2		2	2		2	2		2		2
	R	1		1	0		0	1		0	1		1		1
	M	0		1	1		1	1		1	1		1		1
b	Students	3		3	5	5	4	3	3						
	R	1		0	1	0	1	1	0	1	0				
	M	0		0	0	0	0	0	0	0	0				
c	Students			1		1		5		5		5			
	R			1		0		1		0		1			
	M			0		0		0		0		1			
		X	X		X	X		X	X	X	X				

=> 7x Page Fault

⑯ F:FO => vyhodím tu co je tam nejdále

Příklad

- Ke stránkám se přistupovalo v pořadí: 2,3,2,1,5,2,4,5,3,2,5,2.
- Fyzická paměť se skládá ze tří prázdných rámců a, b, c.
- Jak se bude měnit obsazení rámců v průběhu času?

Číslo stránky	2	3	2	1	5	2	4	5	3	2	5	2
Rámec	a	2		2	5		5	3				
	b		3			2			2	5		
	c			1		4				2		
Začátek seznamu	a	a	a	a	b	c	a	a	b	b	c	a
Výpadek stránky	x	x	x	x	x	x	x	x	x	x	x	x

- Přístup bez výpadku stránky/přístup s výpadkem stránky.
- Pokud je více možností, zvolíme rámec ze začátku abecedy.
- Počet výpadků: 9.

(16) Clock Algorithmus

- používá učebateli P k záčtu hysterezy v první řadce co se hodí

Příklad Page Fault
 1) P uložuje do řadky kde $R \neq 0$, hystereza je 0, posune se do dálky, opakuje
 2) P uložuje do řadky kde $R=0$, hystereza stříháku, posune se o 1 směrem

Přístup k stránkám: 2 3 2 1 5 2 4 5 3 2 5 2
 + / , , | | | | | |

a	Stříháku	2	2	5		5	3			
	R	1	1	1		1	1			
b	Stříháku		3		3	2		2	2	2
	R		1		6	1		0	1	1
c	Stříháku			1	1	4		4	5	
	R			1	0	1		0	1	
clock		a	a	a	a	b	c	a	b	b
Fault		x	x	x	x	x	x	x	x	x

$\delta \times \text{Page Fault}$

(17) Page Fault - LRU (Least Recently Used)

- hodisíme tu stránku, ke které se nejdále nepřistupovalo

2 3 2 1 5 2 4 5 3 2 5 2

a	2	2		2	2	3				
b		3		5		5		5		
c			1		4		2		2	
	x	x	x	x	x	x	x	x	x	

$\tau \times \text{Page Fault}$

Aging Algo? To snad ne :/ h) Fitwiki to málo falešný Místnost

(17)

Předpokládejte následující konfiguraci diskového pole:

- Typ disků: Seagate Savvio 10K.6 (SAS, 6 Gib/s)
- Počet a velikost disků: 22 x 450 GiB
- Rychlosť disku: 10000 rpm, 340 MiB/s
- Spolehlivost disků: 1172000 h MTBF
- Typ diskového pole: RAID 1
- Řadič: 1x, pracuje s rychlosťí 1700 MiB/s, k tomuto řadiči jsou přímo pripojené všechny disky
- Pripojení pole: Ethernet 10GBASE-T (10 Gbit/s) x 1 rozhraní

a) Pst řeš po 32 měsících budec došlo OK

$$\text{Disk OK za } 32 \text{ měsíčů} \rightarrow R(T) = e^{-\frac{T}{MTBF}}$$

$$\left(1 - \underbrace{\left(1 - R(32) \right)^2}_{\text{obs. } F_{\text{fail}}} \right) \quad \begin{matrix} \uparrow \\ \text{11 \leftarrow počet disků} \end{matrix} \quad - \left(1 - \left(1 - e^{-\frac{365 \cdot 32 \cdot 24 / 12}{1172000}} \right)^9 \right) = 99,555\%$$

Složení jednaží

Jst dleto bude třet obnovu?

↓

- rychlosť normální 340 MiB/s, $340 - 20 = 320 \text{ MiB/s}$

- obnovujeme 450 GB

$$\Rightarrow \frac{450 \cdot 2^{10} \text{ MiB}}{320} = 1440$$

- Typ disků: Seagate Savvio 10K.6 (SAS, 6 GiB/s)
- Počet a velikost disků: 18 x 750 GiB
- Rychlosť disku: 10000 rpm, 330 MiB/s
- Spolehlivost disků: 1520000 h MTBF
- Typ diskového pole: RAID 10 (mirror then strip)
- Radič: 1x, pracuje s rychlosťou 1320 MiB/s, k tomuto radiči jsou přímo připojené všechny disky
- Připojení pole: Ethernet 10GBASE-T (10 Gbit/s) x 2 rozhraní



$$1) Kapacita = 18 \times 750 / 2 = 6750$$

$$2) Read = \min(18 \times 330, 1320) = 1320$$

$$3) Write = \min(0.5 \times 18 \times 330, 0.5 \times 1320) = 660$$

4) 1 (pokud 2 z mirrora => problym)

5) $\Rightarrow 45$ měsíců bude jít z finančního dluhu

$$\left(1 - \left(1 - e^{-\frac{24 \times 365 + 45/12}{1520000}} \right)^2 \right)^9 = 99,59\%$$

6) Obnovu 1 disku. 20 MiB provoz

$\Rightarrow 20$ MiB se kopíruje mezi 9 zbylých stripů. 2.22 zážádů k jednomu disku

Tedy číslo $330 - 2.22$ na zapisujeme 330 $\Rightarrow 327.78$ obnovic

- můžeme cítit a zapisovat současně počet 2 + 327.78 < rychlosť shémuice

\Rightarrow což je $\Rightarrow 327.78$ MiB/s

$$\frac{750 \cdot 2^{10}}{327.78} = 2343 \text{ s}$$

- Typ disků: WD RE SAS (SAS, 12 GiB/s)
- Počet a velikost disků: 14 x 1500 GiB
- Rychlosť disku: 7200 rpm, 150 MiB/s
- Spolehlivosť disků: 720000 h MTBF
- Typ diskového pole: RAID 6, 1 řadič pracující s rychlosťou 270 MiB/s, k tomuto řadiči jsou přímo připojené všechny disky
- Připojení pole: FibreChannel 1600 (1600 MiB/s) x 2 rozhraní

$$1) Kapacita = (14 \times 2) \times 1500 = 18000$$

$$2) R_{read} = \min(12 \times 150, 270) = 270$$

$$3) W_{write} = 12 / 14 + \min(14 \times 150, 270) = 231,428$$

↑
výpočetní

4) Inhou vypočítat 2 disky

5) Bez záloh dat 37 dní.

$$\text{pst 1 disk 0\%} = \frac{-24 \times 365 + 37 / 12}{e^{-720000}} = 0,9632$$

Aby byly data 0% musí vypočítat 0,1 mezi 2 disky

$$(0,9632)^{14} + \binom{14}{1} \cdot (1 - 0,9632)^1 \cdot (0,9632)^{13} + \binom{14}{2} \cdot (1 - 0,9632)^2 \cdot (0,9632)^{12} = 98,66\%$$

↑
výpočetní 0\%
↑ Fail
↑ Fail

6) Doba obnovy jednoho disku. Záložní 10 MiB/s

- potřebujeme čist 12 a m 1 záložní 12:1 zálohování souborů
- fcdy 270 / 13 MiB/s zápis

$$\frac{1500 \cdot 2^{10}}{270 / 13} = 76800$$

Používáte následující konfiguraci diskového pole:

- Typ disků: WD Caviar Red (SATA, 5 GiB/s)
- Počet a velikost disků: 12 x 1500 GiB
- Rychlosť disku: 7200 rpm, 90 MiB/s
- Spolehlivost disků: 773000 h MTBF
- Typ diskového pole: RAID JBOD (concatenation), 1 řadič pracující s rychlosťou 180 MiB/s, k tomuto řadiči jsou přímo připojené všechny disky
- Připojení pole: Ethernet 1000BASE-T (1 Gbit/s) x 1 rozhraní

akou bude mít takové pole kapacitu (v GiB):

$$1) K_{\text{kapacit}}} = 12 * 1500 = 18000$$

$$2) R_{\text{read}} = 90 \text{ MiB}$$

$$3) W_{\text{rite}} = 90 \text{ MiB}$$

$$4) V_{\text{rychlosti}} = 0$$

$$5) \text{pst} \text{ dato ob } 2,43 \text{ m}$$

$$\text{Pst ob } = e^{-\frac{365 * 24 * 24}{773000}} = 0,96$$

$$\Rightarrow (0,96)^{12} = 61,27\%$$

$$1) 9000$$

$$2) \min(18 * 90, 180) = 170$$

$$3) \min(18 * 90 + 0,5, 170 + 0,5) = 90$$

$$4) \text{jednou}$$

$$5) 24 \text{ m} \quad 1 - \left(1 - \left(e^{-\frac{365 * 24 * 24}{853000}} \right)^9 \right)^2 =$$

$$0,98 \text{ pst OK}$$

$$2) \text{odloč ob } (0,98)^9 = 0,83375$$

$$2) \text{odloč ob } = 1 - 0,83375 = 0,16625$$

$$\text{ohc jso u hrubk } (0,16625)^2 = 0,027625$$

$$\text{odpom 1 zjdc } = 1 - 0,027625 = 0,97236 \neq 95,13 \text{ dle zadání}$$

6) Záleží 10 MiB/s. Musíme obnovit 9 GiB aby čist / zapísat poslední řádky 9 * 90 MiB

ale sběrnic pouze 120 MiB - 10 MiB zbylé = 170 / 2 = 85

$$\frac{9000 \cdot 2^{10}}{80} = 108423$$

čist / zapis

potřebuju aby pětka celá
byla když plus stíha



Stripe Stripe

Fit WiFi

2023 26.5

1)a

Máme k dispozici pevný disk s následujícími parametry:

- Kapacita: 1 TiB
- Velikost sektoru: 4096 B
- Počet hlav: 2
- Počet cylindrů: 32768
- Průměrný počet sektorů na cylindr: 8192
- Rychlosť: 10000 rpm
- Průměrná doba vystavení hlaviček: 9 ms
- Doba vystavení hlaviček na sousední cylindr: 3 ms

1. Jak dlouho bude trvat přečíst soubor o velikosti 7819264 B z disku, pokud je jeho obsah uložen na jednotlivých sektorech náhodně umístěných po disku?

Čas zadejte v ms, tolerance 1%.

Pozor, při výpočtu používáme jednotky KiB, MiB, ..., platí: 1 KiB = 1024 B, 1 MiB = 1024 KiB, ...

$$\text{doba} = 7819264 \text{ B} \quad \text{náhodně na disku}$$

$$\text{počet sektorů} = 7819264 / 4096 = 1909$$

$$\text{jedno otáčení} = \frac{1}{\text{rps}} = \frac{1}{\frac{10000}{60}} = 0,006 \text{ s} = 6 \text{ ms}$$

$$\text{Spisový sektor (půl točítka)} = 3 \text{ ms}$$

$$\text{čas jednoho sektoru} = \frac{\text{jedno otáčení}}{\text{co jsou písací}} = \frac{6}{8192}$$

$$\begin{aligned} \text{Doba zápisu} &= \text{počet sektorů} * (\text{Spisový sektor} + \text{čas jednoho sektoru} + \text{vystavení hlaviček}) \\ &= 1909 * (3 + \frac{6}{8192} + 9) = 22909,4 \text{ ms} \end{aligned}$$

2 cylindru

1)b

Máme k dispozici pevný disk s následujícími parametry:

- Kapacita: 1 TiB
- Velikost sektoru: 4096 B
- Počet hlav: 2
- Počet cylindrů: 32768
- Průměrný počet sektorů na cylindr: 8192
- Rychlosť: 10000 rpm
- Průměrná doba vystavení hlaviček: 9 ms
- Doba vystavení hlaviček na sousední cylindr: 3 ms

1. Jak dlouho bude trvat přečíst soubor o velikosti 7819264 B z disku, pokud je jeho obsah uložen na jednotlivých sektorech ~~náhodně~~ umístěných po disku?

Čas zadejte v ms, tolerance 1%.

Pozor, při výpočtu používáme jednotky KiB, MiB, ..., platí: 1 KiB = 1024 B, 1 MiB = 1024 KiB, ...

$$\text{sektory} = 7819264 / 4096 = 1909$$

$$\text{počet cylindrů} = 1909 / 4096$$

$$\text{jedno otáčení} = 1 / \frac{10000}{60}$$

$$\text{Spisový sektor} = \text{jedno otáčení} / 2$$

$$\text{čas celkový} = \text{vystavení} + \text{počet cylindrů} * \text{jedno otáčení} + \text{počet cylindrů} * \text{vystavení soused} + \text{počet cylindrů} * \text{spisový sektor}$$

$$\begin{aligned} &\uparrow \\ -1 \text{ nahoře dolní část} \end{aligned}$$

$$\begin{aligned} &\downarrow \\ \text{horní celá část} \end{aligned}$$



Na systému je nainstalován operační systém Oracle Solaris 11 (64b) a nastaveny následující limity:

- maximální velikost zásobníku každého procesu je nastavena na 7 MiB a
- běžný uživatel smí spustit celkem maximálně 160 vláken (po dosazení tohoto limitu budou funkce využívající nová vlákna vracet chybou).

Program „bin:sleep“ bude po spuštění n sekund ve stavu BLOCKED a potom se ukončí. Podobně funkce sleep(n) způsobí uspání volajícího procesu na n sekund. Dobu potřebnou na provedení ostatních systémových a knihovních volání zanedbejte. Předpokládejme, že běžný uživatel, kterému na systému běží pouze login shell, v čase T_0 spustí přeloženou verzi následujícího programu /home/user/prog.

```
int main ()
{
    char a[745531];
    char *ptr_char = (char *) malloc (3700961);

    for ( int i = 0; i < 5 ; i++ )
    {
        pid_t child_pid = fork ();
        if (child_pid == 0)
        {
            int status;
            execvp("/bin/sleep", "sleep", "22", (char *) NULL);
            wait(& status);
        }
        else
        {
            sleep(11);
        }
    }
    free ( ptr_char );
    return 0;
}
```

Uvažujte následující příkaz:

\$./home/user/prog

1. Kolik procesů se celkem vytvoří v důsledku spuštění tohoto příkazu?
2. Za kolik sekund po spuštění programu /home/user/prog dobehne poslední proces?
3. Kolik procesů poběží v čase $T_0 + 52.734$ s?
4. Kolik míst si bude alokovat každý proces na zásobníku (zaokrouhlete na MiB nahoru)?
5. Kolik míst si bude alokovat každý proces na haldě (zaokrouhlete na MiB nahoru)?

def:
↓ ↓ ↴
1) Pouze 5+1, exec hodičí plugin

2) $44+22 = 66$

\uparrow \uparrow
4x sleep main 1x sleep

3) Kolik jich běží v $T_0 + 52.734$ s?

$$\text{main} + C_4 + C_5 = \underline{\underline{3}}$$

	Konec	Start
C1	22	0
C2	11+22	11
C3	$2.11+22 = 44$	22
C4	$33+22 = 55$	33
C5	$44+22 = 66$	44

4) Allokace zásobníku $(745531 + 8 + 4 + 4 + 4) / 2^{20} = 1\text{ MiB}$

5) Allokace haldy $3700961 / 2^{20} = 4\text{ MiB}$

2b)

```
int main ()
{
    char a[2034618];
    char *ptr_char = (char *) malloc (3212961);

    for ( int i = 0; i <= 3 ; i++ )
    {
        pid_t child_pid = fork ();
        if (child_pid == 0)
        {
            execvp("/bin/sleep", "sleep", "7", (char *) NULL);
        }
        else if (child_pid >= 0)
        {
            execvp("/home/user/prog", "prog", (char *) NULL);
        }
    }
    sleep(3);
    free ( ptr_char );
    return 0;
}
```

Uvažujte následující příkaz:

\$./home/user/prog

1. Kolik procesů se celkem vytvoří v důsledku spuštění tohoto příkazu?
2. Za kolik sekund po spuštění programu /home/user/prog dobehne poslední proces?
3. Kolik procesů poběží v čase $T_0 + 52.734$ s?
4. Kolik míst si bude alokovat každý proces na zásobníku (zaokrouhlete na MiB nahoru)?
5. Kolik míst si bude alokovat každý proces na haldě (zaokrouhlete na MiB nahoru)?

1) 207 (Limit je 203 ale už běží sleep)

2) 7 sekund

3) 206 (main už neprobízí, tak užívá po 3 sekundách)

4) $[2034618 / 2^{20}] = 2\text{ MiB}$

5) $[3212961 / 2^{20}] = 4\text{ MiB}$

3)

Bod připojení	Disk	Typ systému souborů	Parametry připojení
/	/dev/sda1	UFS	read/write, noatime
/home	/dev/sda2	UFS	read/write, noatime
/fat1	/dev/sdb1	FAT32	read/write
/fat2	/dev/sdc1	FAT32	read/write

- Parametr atime znamená, že se u souborů/adresářů aktualizuje čas přístupu při každém přístupu.
- Parametr noatime znamená, že se u souborů/adresářů neaktualizuje čas přístupu.
- Systém souborů FAT32 nepoužívá informaci o přístupu, tedy atribut atime/noatime není použit.
- Při výpočtu používáme jednotky KiB, MiB, ..., platí: 1 KiB = 1024 B, 1 MiB = 1024 KiB, ...

Systém souborů UFS pracuje s i-nodey, které mají 12 odkazů přímých a po jednom jedonásobném, dvojnásobném a trojnásobném neprímém odkazu na datové bloky. Velikost datového bloku je 16 KiB, odkazy na datový blok jsou 64-bitové. Atributy i-nodej jsou: typ souboru, velikost souboru, čítací pravou, vlastník a skupina, čas modifikace, čas přístupu, symbolický link se vždy zapisuje do datového bloku. Informace o volných objektech UFS (inodech a blocích) jsou uloženy v bitové mapě na disku, která se při připojení systému souborů načítá do paměti.

Systém souborů FAT pracuje s datovými bloky (clustery) o velikosti 16 KiB a odkazy na clustery jsou 64-bitové.

Předpokládejme, že obsah adresáře nepřesáhne velikost bloku 16 KiB. Dále předpokládejte, že v paměti jsou načtené pouze superbloky/boot sektoře jednotlivých připojených disků. Dále pro jednoduchost předpokládejme, že operace uspějí (cesty existují), soubory lze číst/zapisovat, je dostatek místa, všimněte si, že operace prováděme pod uživatelem root, tedy nemáme omezená přístupová práva). Pro zpracovaný soubor vše, že zobrazení by vypadalo takto:

```
# ls -l /var/boot/etc/doc/local/etc
...
-rw-rw-rw- 1 root root 1422467929837 Apr 7 00:05 index.html
...
```

V takové konfiguraci se provede následující příkaz shellu:

```
# rm -f /var/boot/etc/doc/local/etc/index.html
```

Uvažujte pouze efekt tohoto příkazu (tedy neuvažujte režii spojenou s vlastním natažením tohoto příkazu do paměti).

- Kolik i-nodů se musí načíst?
- Kolik i-nodů se musí zapsat?
- Kolik datových bloků se musí načíst?
- Kolik datových bloků se musí zapsat?
- Kolik operací při alokování/uvolňování bloků a i-nodů musí být provedeno? Každou operaci alokace/uvolnění bloků/i-node započtěte zvlášť.

1) Odstřižení = celý cesta včetně Souboru => 8 i-nodes

2) 1 - do posledního adresáře

3) ? TODO

4) 1 do posledního adresáře

5) ? TODO

4) Otázka: 4 (8 bodů)

Zadání:

Systém používá virtuální paměť se stránkováním. Velikost stránky/rámce je 16384 B, velikost logické adresy je 36 bitů, velikost fyzické adresy je 36 bitů. Systém používá TLB o velikosti 512 položek (fádek). Na systému právě běží 983 uživatelských procesů. Každému uživatelskému procesu je alokováno pouze na nejnížších adresách 41059304 bytů pro kod a na další a na nejvyšších adresách 12894208 bytů pro zásobník. Pro nahrazení stránek se používá algoritmus Second chance.

Předpokládejme, že systém používá invertovanou tabulku stránek. Která z následujících tvrzení jsou pro tento systém s danou konfigurací pravdivá?

- Logická adresa obsahuje offset 14 bitů
 - Logická adresa obsahuje číslo stránky o velikosti 16 bitů
 - Logická adresa obsahuje číslo rámce o velikosti 22 bitů
 - Logická adresa obsahuje číslo rámce o velikosti 22 bitů (Poznámká: ano, otázka byla doprovody duplicitná)
- Fyzická adresa obsahuje číslo stránky o velikosti 22 bitů
 - Fyzická adresa obsahuje číslo rámce o velikosti 22 bitů
 - Fyzická adresa obsahuje číslo rámce o velikosti 36 bitů
 - Fyzická adresa obsahuje offset o velikosti 14 bitů
- Při překladu adresy hledáme v tabulce rádku, ve které je záznam o dané stránce
 - Při překladu adresy hledáme číslo stránky používá jako index do invertované tabulky stránek
 - Při překladu adresy hledáme v tabulce rádku, ve které je záznam o dalším rámci
 - Při překladu adresy se číslo rámce používá jako index do TLB
- Jádro systému si musí pamatovat jenom jednu tabulku stránek
 - Jádro systému si musí pamatovat pouze tabulky, které nejsou v TLB
 - Jádro systému si nemusí pamatovat žádnou tabulkou vše je uloženo v TLB
 - Jádro systému si musí pamatovat pro každý proces právě jednu tabulku
- Na rádce invertované tabulky stránek je uložený Present bit
 - Na rádce invertované tabulky stránek musí být vždy uložené číslo rámce
 - Na rádce invertované tabulky stránek je uložen odkaz na umístění v TLB
 - Na rádce invertované tabulky stránek je uložený offset
- Na rádce TLB je uložené číslo rámce
 - Na rádce TLB je uložené číslo stránky
 - Na rádce TLB je uložený offset
 - Na rádce TLB je uložený Modify bit
- Cache hit ratio TLB se může zvýšit, pokud systém bude používat pro proces menší stránky
 - Současně serverové a desktopové procesory umožňují používat různé velké stránky pro různé procesy
 - Současně serverové a desktopové procesory umožňují aplikacím za běhu měnit velikost TLB podle potřeby
 - Cache hit ratio TLB se může zvýšit, pokud systém bude používat pro proces větší stránky
- Kolik rádek bude obsahovat tabulka stránek?

$$\text{Offset} = \log_2(16384) = 14$$



$$\text{Velikost Rámce} = 2^{22} = 4194304$$

5)

[-] Otázka: 5 (3 body) ↗

Zadání:

Vyberte pravdivé tvrzení:

1. Kernel běží typicky v režimu supervisor, ale pokud běží ve virtuálním stroji, běží pouze v režimu user.
2. Kernel běží typicky v režimu supervisor, ale jeho moduly a ovladače běží pouze v režimu user.
3. Kernel běží typicky v režimu user, ale jeho moduly a ovladače běží v režimu supervisor.
4. Kernel běží typicky v režimu user, ale v případě potřeby se automaticky přepíná do režimu supervisor.
5. Kernel běží typicky v režimu supervisor.

f_{ho} : 5

Ne: ostatní

6)

Zadání:
Předpokládejme následující systém:

- Procesor: 4x Intel® Xeon™ Processor (počet jader na procesor: 1).
- Počet vláken na jádro: 1 (tedy celkem však 4 na procesor).
- Architektura: SMPUMA (desktop).
- Paměť: 2 GB DDR.

Na systému je mimořádný 32-bitový operační systém, který používá pro plánování uživatelských procesů statickou prioritu (vyšší hodnota znamená vyšší prioritu). Jádro/procesor se přiděluje jednotlivým vlákňům maximálně na dobu 30 ms a režii přepínání kontextu zanedbáváme. Dále předpokládáme:

- jádro a ostatní procesy (kromě P₀, ..., P₅) procesor nezatekají,
- plánovac se snaží prodloužit životnost CPU tím, že uděluje rovnoměrnou teplostu všem jádrom, tedy výpočetní zátěž rovnoměrně rozkládá na jednotlivá jádra,
- pro každého běžného uživatele je nastaven limit, který dovolí uživateli sputit maximálně 640 vláken, po dosažení limitu bude funkce vytvářející nová vlákna vracet chybu,
- všichni běžní uživateli z následující tabulky spustí své procesy v čase T₀,
- u každého procesu je uvedena statická priorita, počet vláken, která provádí výpočet, a celková doba výpočtu (doba výpočtu procesu, pokud by běžel sekvenčně na jednom nezajištěném jádru). Pokud je výpočet v procesu realizován pomocí více vláken na dostatečném počtu, pak předpokládáme lineární zrychlení výpočtu, a režii na vytváření a synchronizaci vláken zanedbáme.

Uživatel	Proces	Statická priorita	Počet vláken	Celková doba výpočtu [sec]
xwagner	P ₀	0	3	108
trdlicka	P ₁	26	1	54
trdlicka	P ₂	80	3	270
zdarek	P ₃	71	1	45
zdarek	P ₄	0	4	72
soch	P ₅	26	2	72

1. Za kolik sekund po T₀ se ukončí proces P₇? (Výsledek zadejte i s desetinnou částí, je tolerovaná malá odchylnka.)
2. Za kolik sekund po T₀ se fakticky začne zpracovávat proces P₇? (Výsledek zadejte i s desetinnou částí, je tolerovaná malá odchylnka.)
3. Za kolik sekund po T₀ se ukončí poslední proces? (Výsledek zadejte i s desetinnou částí, je tolerovaná malá odchylnka.)

4 priory = 4 vlákn

Proces	Statická priorita	Počet vláken	Celková doba výpočtu [sec]	Doba výpočtu každého vlákna [sec]
P ₂	80	3	270	90
P ₃	71	1	45	45
P ₁	26	1	54	54
P ₅	26	2	72	36
P ₀	0	3	108	36
P ₄	0	4	72	18

Upřesnit si tabulku =>

- Seředit podle priorit, přidat výp. vlákn

$$T=0 \quad C_{1,2,3} \Rightarrow P_2$$

$$\vee T=45 \text{ do dle } P_3, \text{ z } P_2 \text{ zbyl } 45s$$

$$C_4 \Rightarrow P_3$$

1) 45

2) 90

3) TODO

$$T=45$$

$$C_{1,2,3} \Rightarrow P_2$$

$$\vee T=90 \quad P_2 \text{ hotovo, z } P_1 \text{ zbylo } 54 - 45/3 = 39s$$

$$C_4 \Rightarrow P_1, P_5 (1:2)$$

$$\text{z } P_5 \text{ zbylo } 36 - \frac{2}{3} \cdot 45 = 30s$$

$$T=90$$

2 body 2. Z doporučení koncu to fikt dikt výhodu

2023 6. 2

1)

[>] Oázka 1, Řešení (3 body) *

Vyber jednu z možnosti:

- Uživatelské procesy běží typicky v user mode
- Uživatelské procesy běží typicky v user mode, ale pokud se uživatel přepne jako root, tak běží v režimu supervizor
- Uživatelské procesy běží typicky v režimu supervizor
- Další podobné

- Pwrc *

2a)

Máme procesor na kterém lze spustit 8 vláken. Na něm spustíme proces, který vytvoří 11 vláken (dohromady tedy 12 vláken). Tyto vládkna jsou většinu času ve stavu READY nebo RUNNING. Proces se ukončí po 11727 milisekundách. Jak dlouho proces poběží na procesoru se 3 vláknou?

8 jadrou, proces s 11 vláknami (většinu času Ready / Running) dobačku je 11727

$$\text{Na jednoum vláknu} = 11727 \div 8 = 1466$$

$$\text{Na třech} = 1466 \div 3 = 488,66$$

2b)

Máme procesor na kterém lze spustit 4 vláken. Na něm spustíme proces, který vytvoří 6 vláken (dohromady tedy 7 vláken). Tyto vládkna jsou většinu času ve stavu BLOCKED. Proces se ukončí po 7230 milisekundách. Jak dlouho proces poběží na procesoru se 2 vláknou?

4 jadru, 7 vláknů proces většinu času Blocked.

Na procesoru s 4 jadry? Stojí (většinu času je kódování) 7230

3)

Zadaný kód: (symbolicky)

```
int g_counter = 100;
sem_t g_sem;

void foo(int a) {
    sem_wait();
    g_counter += a;
    sem_post();
}

int main() {
    sem_init(g_sem, 0, 2) // 2 vládkna

    thread_t t1(foo, -10);
    thread_t t2(foo, -8);
    thread_t t3(foo, -7);
    thread_t t4(foo, 10);

    t1.join();
    t2.join();
    t3.join();
    t4.join();
}
```

Vyber všechny správné:

- Program obsahuje časové závislosti chybou
- Program se může dostat do deadlocku
- Program se může dostat do livelocku
- Na konci programu může mit čítač hodnotu x
- Na konci programu může mit čítač hodnotu y
- Program neobsahuje časové závislosti chybou

sem_init(g_sem, 0, 2)
↳ init countlu

hodnota mezi více procesy Ccountlu = 2

Ano pod sem wait mohou být 2 vládku hojednu

4-6 hujou vyfoceny

1)

Program je navržen jako vícevláknový, má hlavní vlákno a vytvoří 9 vláken. Všechna vlákna jsou téměř po celou dobu běhu procesu ve stavu blocked. Program spustíme na PC, který má 3 jádra, proces skončí za 17280ms (real time). Velikost časového kvanta je 10ms. Dobu potřebnou pro přepínání kontextu zanedbáme. Jaká bude doba běhu (v ms), pokud by PC měl 1 jádro navíc a každé jádro by bylo 2x výkonnejší?

$$\text{ho jednomu vláknu} \quad 17280 * 3 = 51840 \text{ s}$$

$$\text{ho 4 vláknach} \quad 51840 / 4 = 12960$$

$$\text{když 2 vlákn je } 2 \times \text{výkonejší} \rightarrow 12960 / 2 = 6480$$

2)

Instrukce které jsou na procesoru x86 k dispozici v režimu user?

1. LIDT •
2. CLI •
3. OUT dx, eax (zapiše data do perif.) •
4. MOV ax, bx (zkopíruje obsah registru) ♀
5. SHL eax,3 (bitový posuv vlevo) •

3)

```
int g_x = 200;
sem_t g_sem;

void foo(int val) {
    sem_wait(&g_sem);
    g_x += val;
    sem_post(&g_sem);
}

int main() {
    sem_init(&g_sem, 0, 2);
    thread t1(foo, -4);
    thread t2(foo, 8);
    thread t3(foo, 9);
    thread t4(foo, 9);

    t1.join();
    t2.join();
    t3.join();
    t4.join();
}
```

$$200 \rightarrow 209 \quad \text{pokud 2 op. zálohování mohou se jít do ohlášení}$$

$$T_3 \text{ holt } 200, \text{ připočítává } T_1 T_2, T_4, \dots T_3 \text{ uloží } 200 + 9$$

$$200 \rightarrow 220 \quad 220 \text{ naposledy}$$

4-6 neni'

2019 28.5

1)

```

Na systému je nainstalován operační systém Debian Wheezy 7.5 (64b) a nastaveny následující limity:
• maximální velikost zásobníku každého procesu je nastavena na 8 MiB a
• běžný uživatel smí spustit celkem maximálně 80 vláken (po dosažení tohoto limitu budou funkce vytvářející nová vlákna vracet chybu).

Program "/bin/sleep" bude po spuštění  $n$  sekund ve stavu BLOCKED a potom se ukončí. Podobně funkce "sleep( $n$ )" způsobí uspání volajícího procesu na  $n$  sekund. Dobu
Predpokládejme, že běžný uživatel, kterému na systému běží pouze login shell, v čase  $T_0$  spustí přeloženou verzi následujícího programu /home/user/prog.

int main ()
{
    char a[771167];
    char *ptr_char = (char *) malloc (5939908);

    for ( int i = 0; i < 3 ; i++ )
    {
        pid_t child_pid = fork ();
        if (child_pid == 0)
        {
            sleep(6);
        }
        else
        {
            int status;
            wait(&status);
            execvp("/bin/sleep", "sleep", "7", (char *) NULL);
            sleep(6);
        }
    }
    free ( ptr_char );
    return 0;
}

Kolik procesů se celkem vytvoří v důsledku spuštění následujícího příkazu
$ ./home/user/prog

```

$$1) Počet procesů = \text{main} + 3 = 4$$

$$2) \text{Kdy dožívne posledný} = 3 \cdot 7$$

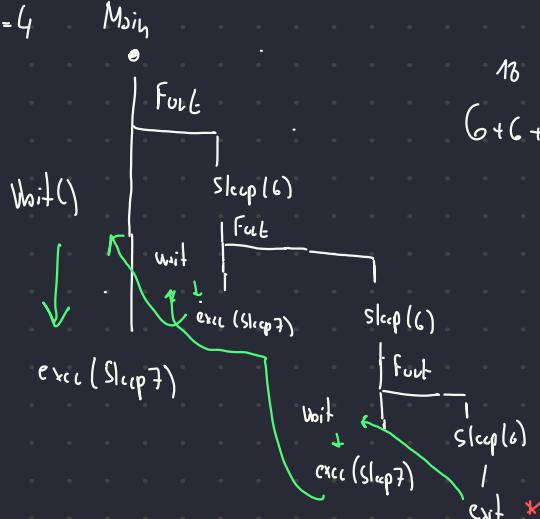
$$3) \text{kolik bude v T} + 20 \cdot 7 =$$

$$\Rightarrow 3$$

* Stanovil po 13 s

$$4) Zásobník 771167 / 1^{16} = 1 \text{ MiB}$$

$$5) Hlava, 5939908 / 2^{20} = 6 \text{ MiB}$$



$$16 + 21 \\ 6 + 6 + 6 + 7 + 7 + 7 = 39$$

2)

Předchozí otázka	
Řádné odevzdání	
Hodnocení:	0.0000
Máme k dispozici pevný disk s následujícími parametry:	
<ul style="list-style-type: none"> Kapacita: 2 TiB Velikost sektoru: 4096 B Počet hlav: 1 Počet cylindrů: 65536 Průměrný počet sektorů na cylinder: 8192 Rychlosť: 10000 rpm Průměrná doba vystavení hlaviček: 9 ms Doba vystavení hlaviček na sousední cylinder: 1 ms 	
ak dlouho bude trvat přečíst soubor o velikosti 402653184 B z disku, pokud soubor není vůbec fragmentovaný? Čas zadějte v ms, tolerance 1%.	
soubor, při výpočtu používáme jednotky KiB, MiB, ..., platí: 1 KiB = 1024 B, 1 MiB = 1024 KiB, ...	
I povíd" (celé desítkové číslo):	119

Je to souborů všechno sebe

$$\text{Počet sektoriů} = 402\ 653\ 184 / 4096 = 98\ 304$$

$$\text{Počet cylindrů} = 98\ 304 / 8192 = 12$$

$$\text{Čas otáčky} = 1 / \frac{10\ 000}{60} = 6 \text{ ms}$$

$$\text{Splínkový sektor} = 3 \text{ ms}$$

$$\text{čtení souboru} = \frac{6}{8192}$$

$$\text{Celkově} = \text{vystavení} + \text{Počet cylindrů} * \text{čas otáčky} + (\text{Počet cylindrů} - 1) * \text{vystavení soused} + \text{Počet cylindrů} * \text{splínkový sektor}$$

$$9 + 12 * 6 + (12 - 1) * 1 + 12 * 3 = 128$$

3)

Předchozí otázka

Rádce otevřené
Hodnocení: 0.0000

Systém používá virtuální pamět se strukturou. Velikost stránky/rámce je 65536 B, velikost logické adresy je 52 bitů, velikost fyzické adresy je 52 bitů. Systém používá TLB o velikosti 256 pravděpodobně, že systém používá invertovanou tabulku stránek. Který z následujících tvrzení jsou pro tento systém v danou konfiguraci pravdivá?

Logická adresa obsahuje číslo rámce velikosti 36 bitů

Logická adresa obsahuje číslo rámce velikosti 36 bitů

Logická adresa obsahuje číslo stránky o velikosti 36 bitů

Logická adresa obsahuje offset 16 bitů

Fyzická adresa obsahuje číslo stránky velikosti 36 bitů

Fyzická adresa obsahuje offset velikosti 16 bitů

Fyzická adresa obsahuje číslo stránky velikosti 36 bitů

Fyzická adresa obsahuje číslo rámce velikosti 52 bitů

Při překladu adresy se číslo rámce používá jako index do TLB

Při překladu adresy se číslo stránky používá jako index do TLB

Při překladu adresy hledáme číslo rámce, do kterého se stránka nahrála

Při překladu adresy lze použít rozptylovací funkci k urychlení překladu

Jádro systému si nemusí pamatovat žádnou tabulkou vše je uloženo v TLB

Jádro systému si musí pamatovat jenom jednu tabulku stránek

Jádro systému si musí pamatovat pro každý proces právě jednu tabulku

Jádro systému si musí pamatovat alespoň 255 tabulek stránek

Na rádce invertované tabulky stránek musí být vždy uloženo číslo rámce

Na rádce invertované tabulky stránek je uložený Reference bit

Na rádce invertované tabulky stránek nesmí být uloženo číslo rámce

Pokud na rádce invertované tabulky stránek uložíme číslo rámce, lze pro vyhledávání v tabulce použít efektivnější algoritmus

Na rádce TLB je uložený offset

Na rádce TLB je uložený Modify bit

Na rádce TLB je uložené číslo stránky

Na rádce TLB je uložené číslo rámce

Cache hit ratio TLB se může zvýšit pokud systém bude používat pro proces menší stránky

Současné serverové a desktopové procesory umožňují používat různě velké stránky pro různé procesy

Cache hit ratio TLB se může zvýšit pokud systém bude používat pro proces větší stránky

Současné serverové a desktopové procesory umožňují aplikacím za běhu měnit velikost TLB podle potřeby.

jolik rádek bude obsahovat tabulku stránek?

Odpověď (celé desítkové číslo): 137438953472

$$\begin{aligned} & \text{Fyz - off} \\ & 2 \\ & = 52 - 16 = 36 \\ & = 2 = 2 \end{aligned}$$

4)

Předchozí otázka

Rádce otevřené
Hodnocení: 0.0000

Na UNIXovém systému máme připojeny následující soubory:

Bod připojení	Disk	Typ systému souboru	Parametry připojení
/	/dev/sda1	UFS	read-write, nosetime
/home	/dev/sda2	UFS	read-write, nosetime
/fat1	/dev/sdb1	FAT32	read-write
/fat2	/dev/sdc1	FAT32	read-write

- Parametr atime známená, že se u souborů/adresářů aktualizuje čas přístupu při každém přístupu.
- Parametr nosetime známená, že se u souborů/adresářů neaktualizuje čas přístupu.
- Systém souborů FAT32 nepoužívá informaci o přístupu, tedy atribut atime/nosetime není použit.
- Při výpočtu používáme jednotky KB, MB, ..., plati: 1 KB = 1024 B, 1 MiB = 1024 KB, ...

Systém souboru UFS pracuje s i-nodey a bloky (clusters) o velikosti 2 KiB a odkazy na clustery jsou 32-bitové.

Předpokládejme, že obsah adresáře nepevnělé velikost bloku 2 KiB. Dále předpokládejme, že v paměti jsou načtené pouze superbloky/boot sektory jednotlivých připojení ke čističkám, že je dostatek místa, všechny si, že operace prováděme pod uživatelem root, tedy nemáme omezena na přístupovou právu. Pro zpracovávaný soubor vše.

```
# ls -l /var/etc/share/boot/include/libz.so.1.2.7
...
-rw-rw-rw- 1 root root 9560847947 May 25 15:16 libz.so.1.2.7
...
```

V takové konfiguraci se provede následující příkaz shellu:

```
# rm -f /var/etc/share/boot/include/libz.so.1.2.7
```

Uvažujte pouze efekt tohoto příkazu (tedy neuvažujte režim spojenou s vlastním natažením tohoto příkazu do paměti).

1) Kolik i-node musí načítat?

Odpověď (celé desítkové číslo): 7

2) Kolik i-node se musí zapsat?

Odpověď (celé desítkové číslo): 0

3) Kolik datových bloků se musí načíst?

Odpověď (celé desítkové číslo):

4) Kolik datových bloků se musí zapsat?

Odpověď (celé desítkové číslo):

5) Kolik operací alokování/uvolňování bloků a i-nodů musí být provedeno? Každou operaci alokace/uvolnění bloku/i-node započte zvlášť.

Odpověď (celé desítkové číslo): 2

fit-wiki.cz/_media/skola/predmety/bi-osy/img_1979.jpg?cache=1

1) od / až 6 souborů 7

2) 1 poslední sdružit

3) TODO

4) 1

5) TODO

5)

Předchozí otázka

Rádce otevřené
Hodnocení: 0.0000

Předpokládejme následující systém:

- Processor(y): 1x Intel® Core™ i5-3330 (počet jader na procesor: 4), Počet vláken na jádro: 1 (tedy celkem vláken na procesor: 4), Počet paměti: 8 GB DDR3
- Na systému je nastavileno 32-bitové operační systém, který používá pro plánování uživatelských procesů statickou prioritu (vyšší hodnota znamená vyšší priority). Je kontextu zanedbatelné. Dále předpokládáme:

- pro každého uživatele je určena jedna statická priorita pro procesor (max 4),
- plánování se snadí přidělit životnosti CPU tim, že udržuje rovnomořnou teplostu všech jader, tedy výpočetní zátěž rovnomořně rozkládá na jednotlivá jádra,
- pro každého uživatele je nastaven limit, který dovolí uživateli spuštít maximálně 832 vláken, po dosažení limitu budou funkce vytvářející nová vláka zavrhovány,
- u každého procesu je uvedena statická priorita, počet vláken, která provádějí výpočet, a celková doba výpočtu (doba výpočtu procesu, pokud by běžel sekvenčně na dosud záčatém počtu jader, pak předpokládáme lineární zrychlení výpočtu), a režim na vytváření a synchronizaci vláken zanedbáme.

Uživatel	Proces	Statická priorita	Počet vláken	Celková doba výpočtu [sec]
xvagner	P ₁	71	2	169
trdlicka	P ₂	77	4	162
sooch	P ₃	6	4	108
xvagner	P ₄	71	1	324
trdlicka	P ₅	71	1	18
sooch	P ₆	6	4	252

Za kolik sekund po T₀ se ukončí proces P₁? (Výsledek zadejte i s desetinnou částí, je tolerována malá odchylnka.)

Odpověď (celé nebo desítkové číslo): 256.5

Za kolik sekund po T₀ se fakticky začne zpracovávat proces P₄? (Výsledek zadejte i s desetinnou částí, je tolerována malá odchylnka.)

Odpověď (celé nebo desítkové číslo): 108

Za kolik sekund po T₀ se ukončí poslední proces? (Výsledek zadejte i s desetinnou částí, je tolerována malá odchylnka.)

Odpověď (celé nebo desítkové číslo):

Pločas	Pliu	Vláska	Calka	Ces po vlnkou
P ₂	77	4	108	27
P ₁	71	2	162	81
P ₄	71	1	18	18
P ₀	6	3	184	63
P ₃	6	4	324	81
P ₅	6	4	252	63

$$T=0 \quad c_{1,2,3,4} \Rightarrow P_2 \quad T = \underline{27} \quad P_2 \text{ Hotov}$$

$$T=27 \quad c_{1,2} \quad P_1 \quad T = 27 + 18 = 45 \quad P_4 \text{ Hotov}$$

$$c_3 \quad P_4$$

$$c_4 \quad P_0, P_3, P_5 \quad (3:4:4)$$

$$2 \quad P_1 \quad 2b_{yw} \quad b_1 - 18 = 63$$

$$2 \quad P_0 \quad 2b_{yw} \quad T_{0,0}$$

$$2 \quad P_3 \quad 2b_{yw} \quad T_{0,0}$$

$$2 \quad P_5 \quad 2b_{yw} \quad T_{0,0}$$

$$T=45 \quad c_1, c_2 \quad P_1$$

$$c_1, c_4 \quad (3:4:4)$$

$$P_0, P_3, P_5$$

$$T=45 + c_3 = 108 \quad \text{sloučit } P_1$$

=

Konec výsledku T_{0,0}

Výsledek B

Pločas	Pliu	Vláska	Calka	Ces po vlnkou
P ₂	80	4	180	45
P ₃	73	3	81	27
P ₄	73	3	216	72
P ₀	33	2	90	45
P ₁	4	1	90	45
P ₅	4	1	54	54

X

4. jídelna

1:1

□ □ □ □

$$T=0 \quad c_{1,2,3,4} \quad P_2 \quad \Rightarrow \quad T=45 \quad P_2 \text{ je Hotovo}$$

P_{1,0} výška kruva

Řádné odevzdání

6.0000

Hodnocení:

Máme k dispozici pevný disk s následujícimi parametry:

- Kapacita: 3 TiB
- Velikost sektoru: 4096 B
- Počet hlav: 1
- Počet cylindrů: 98304
- Průměrný počet sektorů na cylinder: 8192
- Rychlosť: 15000 rpm
- Průměrná rychlosť vystavení hlaviček: 4 ms / 100 cyl

V systému jsou následující požadavky na diskové operace (zadané v tomto pořadí):

Cylinder	Počet za sebou jdoucích sektorů	Operace
55700	3000	W
53100	3200	W
38600	400	R
82800	5800	W
11300	1000	R

Hlavičky se na počátku nacházejí na cylindru 48300 a (pokud je to zadanou strategii potřeba) uvažujte jejich pohyb směrem k vyšším cylindrům.

Jak dlouho bude trvat vyřízení těchto požadavků, pokud uvažujeme strategii SCAN Algorithm (elevator alg.)? Výsledek zadejte v ms, tolerance 1%.

Pozor, při výpočtu používáme jednotky KiB, MiB, ..., platí: 1 KiB = 1024 B, 1 MiB = 1024 KiB, ...

Odpověď (celé nebo desetinné číslo):

4254,88

48300 → 53100 → 55700 → 82800 → 38600 → 11300
 192ms 104ms 108ms 1768 1092

$$\text{oběhu hl.} = \underline{\underline{4240 \text{ ms}}} \quad \text{je to spavidlo } \left(1 / \frac{15000}{60}\right) / 2 = 2 \text{ ms, } \text{čemu } 5 \times \Rightarrow 5 \times 2$$

$$\text{Celkový čas} = 3000 + 3200 + 400 + 5800 + 1000 = 13400$$

$$\text{Čas jedné sekce} = \frac{6}{8192}$$

$$\text{Čas všechny} = 13400 \times \frac{6}{8192} = 9,81445$$

$$\text{Celkový čas} = 4240 + 10 + 9,81445 = 4259,8 \quad \text{Tolerance 1% dobrý OK}$$

Máme k dispozici pevný disk s následujícími parametry:

- Kapacita: 2TB
- Počet hlav: 1
- Velikost sektoru: 512 B
- Počet cylindrů 524288
- Průměrný počet sektorů na cylinder: 8192
- Rychlosť: 6000 rpm
- Vystavění hlavičky: 7ms
- Vystavění hlavičky na sousední cylindr: 1ms

Jak dlouho bude trvat přečíst soubor o velikosti 20135936 B který je fragmentovaný po 8192B blocích

$$jedna\ otáčka = 1 / \frac{6000}{60} = 10\ ms$$

$$tot.\ spojčení = 5\ ms$$

$$počet\ bloců = 20135936 / 8192 = 2458$$

$$Sčítání\ v\ blocu = 8192 / 512 = 16$$

$$\text{čtení jedné sčítané} = \frac{10}{8192}$$

$$Doba\ čtení\ bloku = 16 \times 10 / 8192$$

$$\text{Čtení souboru} = \text{počet\ bloců} * (\text{vystavění\ hlavičky} + \text{rotacioní\ spojčení} + \text{doba\ čtení\ bloku})$$

$$2458 * \left(7 + 5 + \frac{16 * 10}{8192} \right) = 29544$$

V systému jsou k dispozici celkem 4 rámce, které pro přehlednost označíme **a**, **b**, **c** a **d**. Pro konfiguraci platí:

- Na začátku jsou tyto rámce prázdné.
- Pokud algoritmus pro náhradu stránek může umístit stránku do více rámců, vždy zvolí rámec, který je první v abecedním pořadí.
- Pokud algoritmus periodicky modifikuje R bit, pak se tak děje v časech $i \times 30$, kde $i \in \{0, 1, 2, \dots\}$.
- Pokud algoritmus periodicky modifikuje M bit, pak se tak děje v časech $j \times 125$, kde $j \in \{0, 1, 2, \dots\}$.
- Ke stránkám paměti se přistupuje v následujícím pořadí:

time	11	17	29	42	53	55	68	73	77	88	96	104	110	118
R/W	r	w	w	w	w	r	r	w	r	w	r	w	r	r
page #	2	0	3	3	9	2	5	1	1	5	3	9	8	9

Ke kolika výpadkům stránky (page fault) by došlo, pokud by systém používal pro náhradu stránek **algoritmus LRU**?

LRU - Least Recently Used - vyhodnocuje tu co bylo nejdílčí používání

	11	17	29	42	53	55	68	73	77	88	96	104	110	118
a	2					2					9			9
b		0					5		5					
c			3	3				1	1				3	
d					9						3			
	x	x	x	x			x	x			x	x	x	

=> 9x Page Fault, 9, 5, 8, 3

```
for( int i = 0; i < 5; i++ ){
    if( fork() == fork() )
        break;
}
```

- rozložit pro jednu iteraci, vznikne 3 nové procesy, tedy bude 3 potřebných

- výsledek $1 + 3 + 3^2 + \dots + 3^h$

↑
mnoh

2018 / 2019

Důležitější plánování

a další bloky ↴

i-hodiny

Proces	Celkový počet požadovaných prostředků				Již přidělené prostředky			
	T_0	T_1	T_2	T_3	T_0	T_1	T_2	T_3
P_0	5	5	4	4	1	0	2	2
P_1	1	1	1	1	1	0	0	1
P_2	4	4	6	4	3	3	0	1
P_3	3	3	8	4	0	2	2	2
P_4	1	0	0	1	1	0	0	1

A část prostředků je ještě volná, viz. následující tabulka:

Typ prostředku	T_0	T_1	T_2	T_3
Počet prostředků	0	1	5	1

$$A = \begin{bmatrix} 1 & 0 & 2 & 2 \\ 1 & 0 & 0 & 1 \\ 3 & 3 & 0 & 1 \\ 0 & 2 & 2 & 2 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$M = Q \cdot A$$

$$F = [0, 1, 5, 1]$$

Request
↓

$$Q = \begin{bmatrix} 5544 \\ 1111 \\ 4464 \\ 3384 \\ 1001 \end{bmatrix}$$

$$M = \begin{bmatrix} 4 & 5 & 2 & 2 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 6 & 3 \\ 3 & 1 & 6 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 8 \end{matrix}$$

$$5 \text{ Proc} \quad F = [1, 1, 5, 1]$$

$$2 \text{ uspořejení} \quad F = [6, 1, 5, 2]$$

=> hic dsl kde uspořejit => nebezpečný stav

Coffmanovy podmínky : Všechny vyloučení
 Neodkladitelnost
 Dílčí cestují] hnutí podmínky

Podmínky k ukládání cestují

Předpokládejme následující systém:

- Procesor(y): Intel® Core™ i5-3330 (počet jader na procesor: 4)
- Počet vláken na jádro: 1 (tedy celkem vláken na procesor: 4)
- Architektura: SMP/UMA (desktop)
- Paměť: 4 GiB DDR3

Na systému je nainstalovaný 64-bitový operační systém, který používá pro plánování uživatelských procesů statickou prioritu (vyšší hodnota znamená vyšší prioritu). Jádro/procesor se přiděluje jednotlivým vláknům maximálně na dobu 10 ms a režii přepínání kontextu zanedbáváme. Dále předpokladáme:

- jádro a ostatní procesy (kromě P_0, \dots, P_5) zatěžují procesor z **20%**

$$1 \times 1 \times 4 = 4 \text{ jádra}$$

pouze 80% využití

$$\text{více lze dle fakt } * \frac{100}{80}$$

Uživatel	Proces	Priorita	Počet vláken	Celková doba výpočtu [sec]
xvagner	P_0	48	3	162 - 202,5
zdarekj	P_1	13	2	180 - 225
trdlicka	P_2	13	2	72 - 90
trdlicka	P_3	48	4	72 - 90
xvagner	P_4	13	3	135 - 168,75
soch	P_5	13	2	72 - 90

$$T = 0 \quad P_0, P_3 \quad \text{na 4 jádrách} \quad P_3 \text{ dokáže po } \frac{q_0 \times \frac{7}{4}}{4} = 39 \cdot 375 \quad T = 39 \cdot 375$$

$$\text{celkový plísek} = 39 \cdot 375 \cdot 4 = 157.5$$

$$P_0 \text{ stíhlo} = 157.5 - 90 = 67.5$$

$$2 P_0 \text{ zůstalo} = 157.5 - 67.5 = 135$$

$$T = 39 \cdot 375 \quad C_{1,2,13} \quad P_0 \quad P_0 \text{ } 135 / 3 = 45 \text{ dokáže } P_0 \Rightarrow T = 84 \cdot 375$$

$$C_{1,2,13} \quad P_1, P_2, P_5 \quad \text{celkový plísek} = 45 \times 4 = 180$$

$$P_2 \text{ } P_0 = 180 - 135 = 45$$

$$2 P_1 \text{ odplácavího } \frac{2}{9} \cdot 45 = 10 \Rightarrow 225 - 10 = 215$$

$$P_2 \quad \frac{2}{9} \cdot 45 = 10 \Rightarrow q_0 - 10 = 80$$

$$P_4 \quad \frac{3}{9} \cdot 45 = 15 \Rightarrow 168.75 - 15 = 153.75$$

$$P_5 \quad \frac{1}{9} \cdot 45 = 10 \Rightarrow q_0 - 10 = 80$$

$$T = 84.375 \quad c_{1,2,3,4} \\ p_{1,2,4,5} \quad P_0 \quad \frac{80 * \frac{9}{2}}{4} = 90 \quad \text{dolzhue } P_2, P_5 \quad T = 174.375$$

$c_{1,2,3,4}$ prisic = $90.4 - 360$

$$2 p_1 \quad 360 \cdot \frac{2}{4} = 30 \Rightarrow 215 - 30 = 185$$

$$P_4 \quad 360 \cdot \frac{3}{4} = 120 \Rightarrow 153.75 - 120 = 33.75$$

$$T = 174.375 \quad c_{1,2,3,4} \quad P_0 \quad \frac{33.75 * \frac{5}{2}}{4} = 14.0625 \quad \text{dolzhue } P_4 \quad T = 188.4375$$

$P_1, P_4 \quad (2:3)$

$$2 p_1 \quad 14.0625 * 4 * \frac{2}{5} = 22.5 \Rightarrow 135 - 22.5 = 112.5$$

$$T = 188.4375 \quad c_{1,2} = P_1 \quad P_0 \quad \frac{112.5}{2} = 56.25 \quad \text{dolzhue } P_1$$

$c_{3,4} = \text{Idle}$

$$T = 188.4375 + 56.25 = 244.6875$$

1) Postit processi \rightarrow Fork + Main = 8

2) Postcond $T + 30$

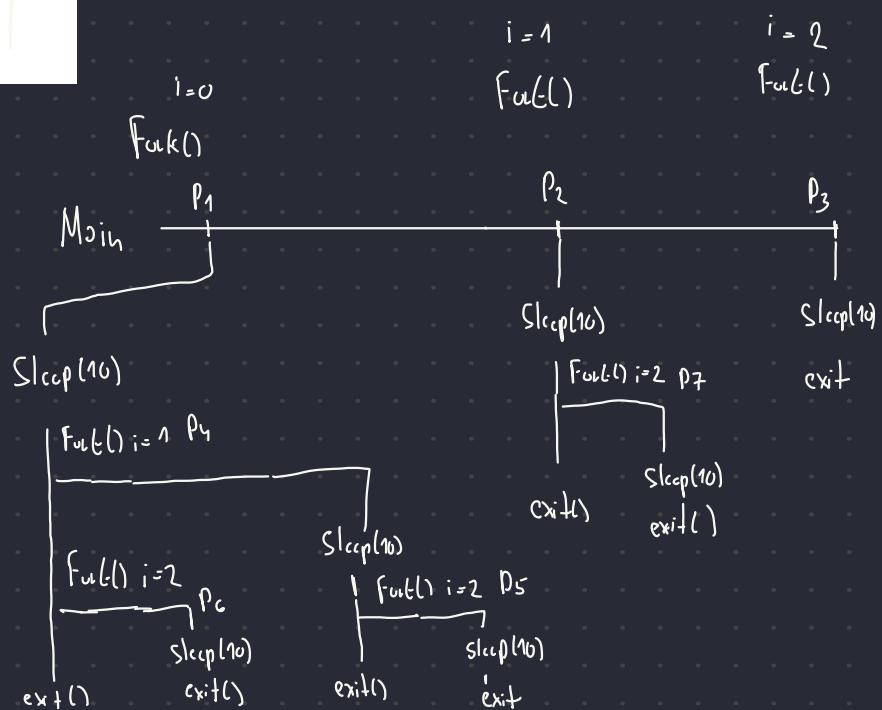
3) $\vee T + 15 \quad P_1, P_2, P_3 \Rightarrow 3$

4) 2500000 $\approx 4797902 / 2^{20} \approx 5 \text{ MiB}$

5) Holes $= 3700521 / 2^{20} \approx 4 \text{ MiB}$

```
int main()
{
    char a[4797902];
    char *ptr_char = (char *) malloc (3700521);

    for (int i = 0; i < 3; i++)
    {
        child_pid = fork();
        if (child_pid == 0)
        {
            sleep(10);
        }
        free(ptr_char);
        return 0;
    }
}
```



$$3 + 1 + 2 = 6 + 1 = 7$$

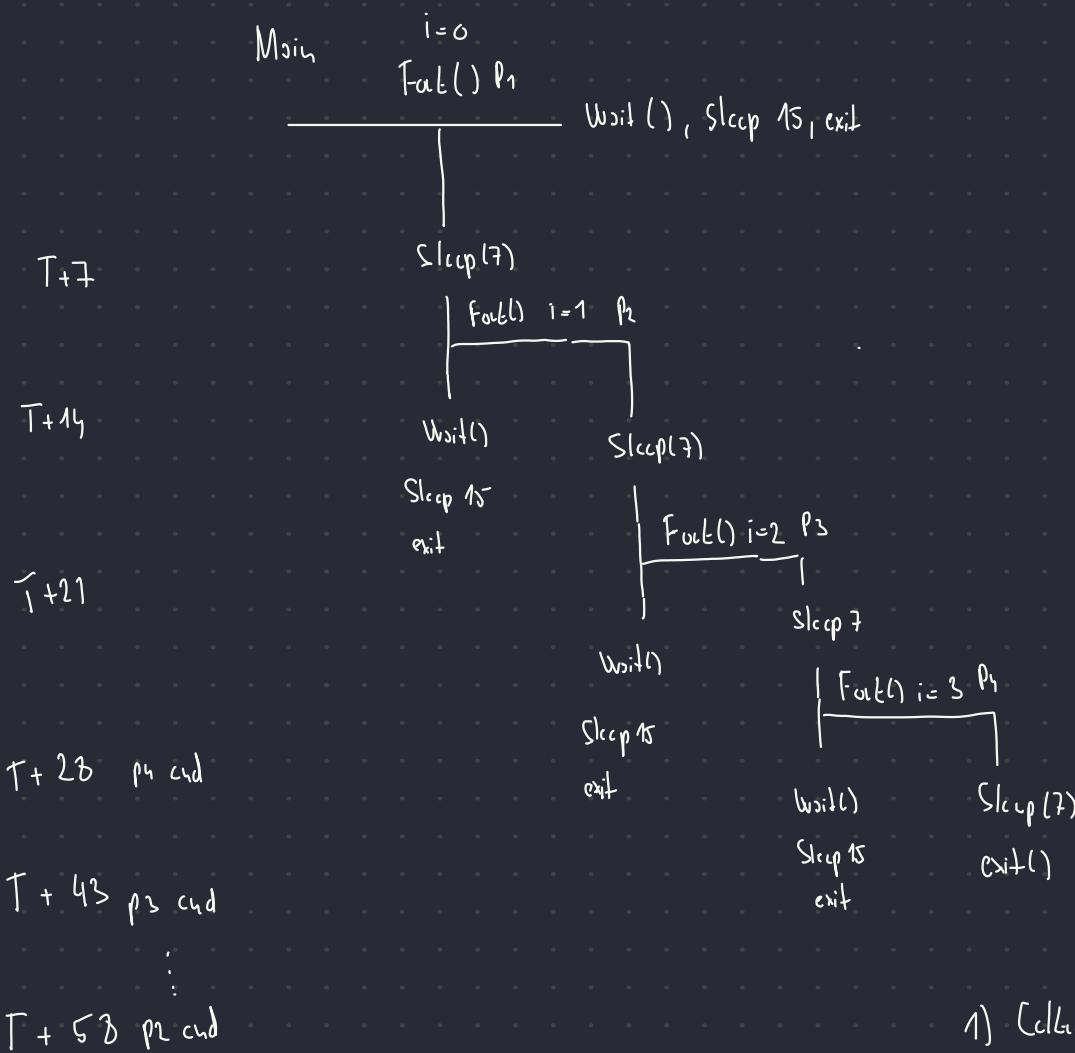
↑
main

```

int main()
{
    char a[1779061];
    char *ptr_char = (char *) malloc (7179004);

    for (int i = 0; i < 4; i++)
    {
        pid_t child_pid = fork();
        if (child_pid == 0)
        {
            sleep(7);
        }
        else
        {
            int status;
            wait(&status);
            execlp("/bin/sleep", "sleep", "15", (char *) NULL);
            sleep(7);
        }
    }
    free(ptr_char);
    return 0;
}

```



1) Callum 5 Procesos

$$2) \text{ Postulante } t = 4*7 + 4*15$$

$$= 60 + 28 = 88$$

3) Kort 5 T = 45, a 77

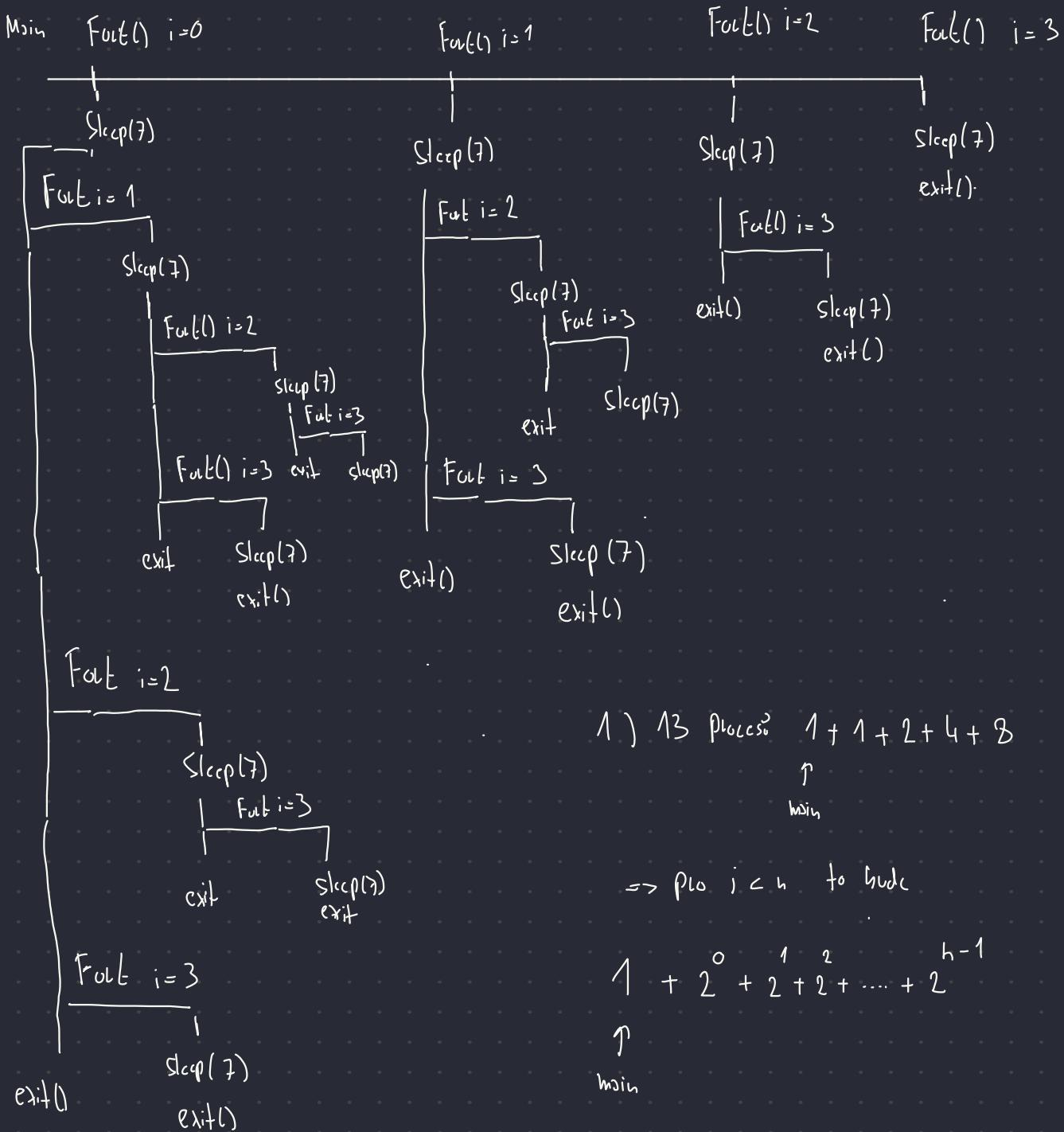
Vec Groupe p4, p5 \Rightarrow 3 procesos

```

...
int main()
{
    char a[1775297];
    char *ptr_char = (char *) malloc (8233219);

    for (int i = 0; i < 4; i++)
    {
        pid_t child_pid = fork();
        if (child_pid == 0)
        {
            sleep(7);
        }
    }
    free(ptr_char);
    return 0;
}

```



$$1) 13 \text{ process } 1 + 1 + 2 + 4 + 8$$

↑
main

\Rightarrow pro. $i < n$ to handle

$$1 + 2^0 + 2^1 + 2^2 + \dots + 2^{h-1}$$

↑
main

2) 28 diff. diff. diff.

$$3) T + 11.5 \Rightarrow \text{main} + 5 = 6$$

Bod připojení	Disk	Typ systému souborů	Parametry připojení
/	/dev/sda1	UFS	read/write, noatime
/home	/dev/sda2	UFS	read/write, noatime
/fat1	/dev/sdb1	FAT32	read/write
/fat2	/dev/sdc1	FAT32	read/write

- Parametr atime znamená, že se u souborů/adresářů aktualizuje čas přístupu při každém přístupu.
- Parametr noatime znamená, že se u souborů/adresářů neaktualizuje čas přístupu.
- Systém souborů FAT32 nepoužívá informaci o přístupu, tedy atribut atime/noatime není použit.
- Při výpočtu používáme jednotky KiB, MiB, ..., platí: 1 KiB = 1024 B, 1 MiB = 1024 KiB, ...

Systém souborů UFS¹ pracuje s **i-nody**, které mají 12 odkazů přímých a po jednom jednonásobném, dvojnásobném a trojnásobném nepřímém odkazu na datové bloky. Velikost datového bloku je 2 KiB, odkazy na datový blok jsou 32-bitové. Atributy i-nodu jsou: typ souboru, velikost souboru, čítač pevných linků, přístupová práva, vlastník a skupina, čas modifikace, čas přístupu, symbolický link se vždy zapisuje do datového bloku. Informace o volných objektech UFS (i-nodes a blocích) jsou uložené v bitové mapě na disku, která se při připojení systému souborů načítala do paměti.

Systém souborů FAT32 pracuje s datovými bloky (clusters) o velikosti 2 KiB a odkazy na clustery jsou 32-bitové.

¹ Unix File System

Předpokládejte, že obsah adresáře nepřesáhne velikost bloku 2 KiB. Dále předpokládejte, že v paměti jsou načtené pouze **superbloky/boot sektry** jednotlivých připojených disků. Dále pro jednoduchost předpokládáme, že operace uspějí (cesty existují, soubory lze číst/zapisovat, je dostatek místa, všimněte si, že operace provádíme pod uživatelem root, tedy nemáme omezená na přístupová práva). Pro zpracovávaný soubor víte, že zobrazení by vypadalo takto:

```
# ll /var/include/dev/share/boot/boot
...
-rw-rw-rw- 1 root root 309788851 Feb 4 12:57 ntpd.conf
...
V takové konfiguraci se provede následující příkaz shellu:
# cp /var/include/dev/share/boot/boot/ntp.conf /home/log/boot/games/ld.so.conf
Uvažujte pouze efekt tohoto příkazu (tedy neuvažujte režii spojenou s vlastním natažením tohoto příkazu do paměti).
```



1) Počet hmotných i-nodes

$$\Rightarrow / \text{nl} / \text{include} / \dots = 8$$

$$/ \text{home} / \text{log} / \dots = 4$$

↑

hmotných 2x /

2) Počet zapojených i-nodes = 2

Díl souboru + Soubor

3) kolik dat Bloků

Adressář 7+4 (/ se nachází)

Nepřímí odkazy v bloku = $2 \text{KiB} / 4 = 2^9$ odkazů

Soubor zálohován = $309788851 / 2 \text{KiB} = 151265$ Bloků

$$151265 - 12 = 151253$$

↑
přímé

$$1 \text{úložný } 2^9 = 512 \Rightarrow 151253 - 512 = 150741$$

$$2 \text{ úložný } 2^9 \cdot 2^9 \Rightarrow 262144 < 150741$$

$$\frac{150741}{2^9} = 294.4 \Rightarrow 295$$

$$\uparrow \text{ záložní } 2 + \boxed{1}$$

$$\begin{aligned}
 \text{CetLcm} &= \text{Soubor} + \text{odískovac} + 2 \cdot \text{?} + 1 \text{účtu} + 2 \text{účtu} \\
 &= 151\ 265 + 11 + 295 + 1 + 1 = 151\ 573
 \end{aligned}$$

4) Zapsat bloky

$$\text{kopiirov}' = 295 + 1 + 1 = 297$$

$$\text{Soubor} = 151\ 265$$

Příklad adresář

$$\rightarrow 297 + 151\ 265 + 1 = 151\ 563$$

5) Počet ovladačů \Rightarrow Stojan

\varnothing_{10} MV

1) Cesta bez koncových souborů

$$\Rightarrow 10$$

2) Zapsat i-nodes 2 soub. a soubor

```
# mv /usr/log/games/dev/src/index.php /var/local/games/home/index.html
```

3) Kolik dat. bloků se musí vytvořit? Počet Adresářů $\Rightarrow 6 + 4 = 10$

4) Kolik zapsat dat bloků sic soub., dat soub. $\Rightarrow 2$

5) Allokovaný / unallokován $\Rightarrow \bigcirc$