Test

Výpadky stránek

co je vypadek stranky
7) Co je to vypadok stranky (1b)
a) vypadnutie listu zo zosita
b) nenajdenie kluca v asociativnej pamati
c) vyhodenie stranky z pamati
d) udalost, ktora nastane pri neexistencii mapovania (správně)

kolik vypadku stranek muze max. nastat - 2B instrukce, 4kB stranky, dvouurovnove strankovani, presouva se 8MB (4108)

kolik vypadku stranek muze max. nastat - 2B instrukce, 4kB stranky, jednourovnove strankovani, presouva se 8MB
(spravne je 4100) (100%)

2 bajtova instrukcia, 8 MB dat, 4 KB stranka, 3 urovnove strankovanie, tie tabulky maju 1024 poloziek, max pocet vypadkov (4114)

- max. počet výpadků stránky při jednoúrovňové stránkování, instrukce 2B, data 8kB, stránka 4kB - 8

kolik vypadku stranek muze max. nastat - 2B instrukce, 4kB stranky, dvouurovnove strankovani, presouva se 8kB
a) 8
b) 10
c) 14
d) 12
(celkem 14 vypadku)

- max. počet výpadků stránky při tříúrovňovém stránkování, instrukce 2B, data 8kB, stránka 4kB - 20 (100%)

Výpadky při LRU

Urcite pocet vypadkov stranok. K dispozicii su 4 ramce (na
zaciatku
vsetky prazdne), pridelovacia strategia je LRU.
Poziadavky na stranky
prichadzaju v tomto poradi : 1 2 3 4 5 6 2 4 2 3 1.

1. (vyp) 1
2. (vyp) 1 2
3. (vyp) 1 2 3
4. (vyp) 1 2 3 4
5. (vyp) 2 3 4 5
6. (vyp) 3 4 5 6
7. (vyp) 4 5 6 2
8. 5 6 2 4
9. 5 6 4 2
10.(vyp) 6 4 2 3
11.(vyp) 4 2 3 1

tz. 9 vypadku (100%)

LRU: 1 3 5 7 2 3 4 5 2 4 6 8
A) 5
B) 9 (správně)
C) 10
D) 8

Proces postupne pouziva stranky v nasledujicim poradi: 1, 2, 3, 4, 5, 6, 3, 6, 2, 5, 1.
K dispozici jsou pouze 4 ramce. LRU
a) 5
b) 9
c) 10
d) 8 (Správně)

RAID

kolik disků může vypadnout v raid0+1 (vhodná polovina) (100%)
kolik disku muze vypadnout v RAID0 (aby to neposkodilo data)(0) (100%)
kolko diskov moze vypadnut v raid 1 (spravne "vhodne volena polovica") (100%)
kolik disku muze vypadnout v RAID6 (aby to neposkodilo data)(dva)
kolko diskov moze vypadnut v RAID 5 (1) (100%)
RAID 50 (1 z kazdej polovice)

Zablokování

Prevedeno do mechanickeho algoritmu - namalujes si kolecka A B C D (procesy) a ctverecky 1 2 3 4 (zdroje). Algoritmus je tento:

1. Je li v instrukci napr. "A zada 1", pak:
- pokud z 1 nevede sipka, namaluj sipku 1->A
- pokud z 1 vede sipka, namaluj sipku A->1
2. Je li v instrukci napr. "A uvolnuje 1", pak:
- smaz sipku 1->A
- vede-li do 1 nejaka sipka/sipky, tak jednu vyber a otoc ji
3. Je-li ted v grafu cyklus, je to deadlock.
4. Cti dalsi instrukci & goto 1

Procesy A,B,C,D zarizeni 1,2,3,4, 12 kroku s popisem zadosti a uvolnovani zarizeni, kdy se D dostane do deadlocku

4zarizeni, 4 procesy, dana posloupnost zadosti/uvolneni, kdy dojde k zablokovani

-pri ktere instrukci dojde poprve k deadlocku (nasledoval vycet A zada o 2 , B zada o 1... B zada o 2, A zada o 1 -> deadlock u seste instrukce)

Mame procesy A,B,C,D a prostriedky 1,2,3,4. Urcte, v ktorom kroku sa proces D dostane do deadlocku?
A ziada 2; A ziada 3; B ziada 1; A uvolnuje 2; C ziada 4;

D ziada 4;
C ziada 2; C uvolnuje 4; B ziada 4; D ziada 3; A ziada 1; C ziada 1
a) 6
b) 10
c) 11 (správně)
d) 9

Procesy A, B, C, D hospodari s prostredky 1, 2, 3, 4 podle nasledujiciho
scenare:
1) A zada 2
2) A zada 3
3) B zada 1
4) A zada 4
5) D zada 4
6) C zada 4
7) A uvolnuje 4
8) D zada 1
9) B zada 3
10) A zada 4
11) A zada 1
12) C zada 1
Ve kterem bode se proces D stava jednim z procesu v deadlocku?
a) 6
b) 10 (spravne)
c) 11
d) 9

Mezi Coffmanovy podminky nepatri
a, nepreemptivne pridelovanie
b, cakanie v kruhu
c, preemptivne pridelovanie
d, postupne pridelovanie
(odpoved: preemptivni pridelovani) (100%)

Co je Randezvous?

(nastudujte si zpravy, SEND, RECEVIE a jak to vypada, kdyz ma schranka velikost 0 zprav)

Randezvous je:
- Setkani 2 entit na 1 miste v kodu pomoci synchronizacnych primitiv ...
- Zablokovani 2 procesu pomoci synchr. ...
- Nejaka uplna blbost
- Rande
(ja som dal to setkani, aj ked sa mi to celkom nezdalo)

Co je to randezvous (1b) (Francuzi nebite ma, tak to bolo napisane v zadani)
(asi myšleno rendez-vous)
a) cakanie na spravu
b) odoslanie spravy
c) odosielanie bez vyrovnavacich front sprav
d) odosielanie s vyrovnavacimi frontami sprav
MARTIN - c)

"Zajímavým případem je situace, kdy je schránka nulové velikosti. Pokud je odesilatelem proveden SEND a jeste tam není zádný príjemce cekající operací RECEIVE, musí odesilatel pockat. Podobne príjemce ceká na odesilatele, az neco zasle. Po predání zprávy, tj. je provedena operace SEND i RECEIVE na jednu schránku, se oba procesy opet rozbehnou. Tento případ se nazývá dostavenícko (randezvous)."

Stránkování aj.

Co resi tabulky stranek?
(prevod cisla stranky na cislo ramce) (100%)

aktivni cekani: pro predpokladane kratke doby cekani

kedy nastava zablokovani u nezaporneho semaforu pri DOWN (citac rovny 0)
kdy se zablokuje DOWN u semaforu (kdyz je <=0)
kedy sa proces zablokuje na semafore, citac cele cislo (aka je hodnota citaca pred operaciou DOWN) (<= 0, tu som mal mozno chybu)

vyhladoveni
co je to vyhladoveni (ze to nerobi nic uzitocne napriek tomu ze to nieco robi, vid filozofovia)

Asociativni pamet je:
a) pamet s postupnym prohledavanim podle klice
b) pamet s paralelnim pristupem na adresu
c) pamet s paralelnim hledanim podle klice (správně)
d) pamet s dvojitym paralelnim prohledavanim cele pameti

co se používá na velmi rychlý přístup ke kritické sekci (spinlock) (100%)

1) Jake jsou zakladni pozadavky na HW pro podporu strankovani?
A) prepocet cisla stranky na cislo ramce, rozpoznani vypadku stranky
B) prepocet cisla stranky na cislo ramce, rozpoznani vypadku stranky, asociativni pamet
C) rozpoznani vypadku stranky, viceurovnove strankovani
D) prepocet cisla stranky na cislo ramce, asociativni pamet
SPRAVNE - a)

Tabulka stranek resi
A) Prepocet realne adresy na fyzickou
B) Prepocet fyzicku adresu na virtualni
C) Prepocet cisla ramca na cislo stranky
D) Prepocet cisla stranky na cislo ramca (správně)

Alg Bankere je jednou z moznosti k:
A) Urceni stranky, kt. bude odstranena z pameti
B) Detekce zablokovani a zotaveni ze zablokovani
C) Predchazeni (napadani Coffmanovych podminek) zablokovani
D) Predchazeni (vyhnuti se) zablokovani (správně)

Psací otázky

procesy a vlakna
-proces
 -paměť, prostředky, práva
 -hierarchie procesů
 -identifikace procesů (PID)
-vlákna
 -uživatelský prostor
 -jádro
 -hybrid (m:n)
-Linux nezná vlákna, jsou to procesy které sdíly stejný adr.prostor.

 Definujte co jsou to zpravy a k nim prislusna systemova volani.
Zprávy
Systémová volání SEND a RECEIVE pro zasílání a príjem zpráv. Zpráva je nejaká prenásená informace mezi odesilatelem a príjemcem.

SEND zasle zprávu. Nikdy se nezablokuje. Pokud na zprávu ceká príjemce operací
RECEIVE, je mu zpráva dorucena a príjemce odblokován.

RECEIVE zprávu prijímá. Pokud zádná zpráva není dostupná, prijímající proces je
zablokován.

Je treba vyresit problém adresace, tj. urcení odesilatele a príjemce.
Lze resit napr. identifikací procesu na daném pocítaci.
Jiným resením je zavedení schránek (mailboxu) a adresace pomocí identifikace schránky.

Schránka je vyrovnávací pamet typicky pevné délky. Do ní jsou zprávy ukládány operací SEND a z ní vybírány operací RECEIVE. Schránka modifikuje operaci SEND: pokud je schránka plná, zablokuje se i SEND.

zprávy a synchronizační fce k nim (posilani zprav a synchronizacni funkce)
spravy+funkcie ktore sa pouzivaju
 - tam mi chybělo uvést, jak se zprávy adresují, resp. že se vůbec adresují, a pak ještě že send/receive musí být atomické (a to jsem věděl, a pak jsem si řekl, že to je tak samozřejmý, že to tam psát nebudu ;e)) -> bod dolů

Zprávy
-data, informace přenášená od odesílatele k příjemci
-implementace OS
-atom.operace SEND a RECEIVE
-SEND-odešle zprávu - pokud na ni čeká příjemce RECEIVE, přijme ji a odblokuje se
-RECEIVE-přijímá zprávu, pokud žádná není, zablokuje se
-adresace-pomocí id procesu (jen mezi 2 procesy)
  -schránky(velikost a id)-pracuje se s id, při plné schránce SEND zablokován
  -více procesů spolupracuje na jedné schránce
-rendez-vous-schránka 0 velikosti-příjemce/odesilatel-jeden čeká na druhého-po předání odblokovány oba

architektury os

tak tohle jsem ale naprosto vůbec netušil. tak jsem tam napsal něco jako jednouživatelský/víceuživatelský, single/multitasking ... to šlo úplně mimo
pak jsem tam dal monolitický x modulární x mikrokernelový systém, to už byla strefa do černého, za to jsem dostal celé tři body (ještě mi mr. yaghob řekl, že si myslí, že to je docela štědré ... to má sice na jednu stranu pravdu, ale jestli by mi chyběly čtyři nebo jeden bod na jedničku, to už je myslím docela jedno ;e)) ... pak tam mělo být něco o VMkách údajně ... neví někdo, co to má být?
architektury OS (monolit, mikrokernel, hybrid/modularny)

-Monolitický systém(nejstarší,nejrychlejší,dodnes používaný, UNIX, Windows)
-Virtuální stroje-původně(VM pro IBM 360,OS má dvě úlohy-multiprogramming,extended machine)
  -dnes(definovaný abstraktní stroj,nezáleží na skutečném HW,pomalejší)
-mikrojádro-(nejnovější,experimentální,co nejmenší,architektura klient/server,komunikace mezi procesy,vhodný pro distribuované OS,dnes jediný komerční OS založený na mikrojádře (Chorus))

Charakterizace zarizeni

(odnimatelnost, rychlost, velikost, pristup R,W atd.)
-druhy - blokové(disk, síťová karta)
  - znakové(klávesnice, myš)
-přístup - sekvenční(datová páska)
  - náhodný (disk, CD)
-synchronní(disk)-na žádost
-asynchronní(síťovka)-poskytuje i nevyžádaná data
-sdílení-sdílené(síťovka)-preemtivní
  -vyhrazené(tiskárna)-nepre.,dedikované
-rychlost/směr dat -RW

Strankovani a segmentace

rozdil mezi strankovanim a segmentaci

základy stránkování
 - tam jsem měl jen velmi letmo naznačené výpadky stránek, tedy to, že ta stránka tam být může a nemusí ... to jsem nějak nepochopil, nicméně za to jeden bod dolů, a druhý za to, že jsem napsal, že se "vyhledává" v tabulce stránek, a ono se to přitom indexuje. na tomhle by mě prej při souborce hezky vydusili ;e)

(co resi, VAP, FAP, prepocet adresy, vypadek stranky, ... vzdyt to znate)

zaklady strankovania (VA -> FA, indexovanie v strankovacej tabulke, PTE, riesenie velkosti - inverzne (hashovane) + viacurovnove, TLB, riesi fragmentaciu,

|VAP| > |FAP|, ...)

stránkování
-VAP rozdělen na úseky stejné délky – stránky(velikost je mocnina 2)
-FAP rozdělen na úseky stejné délky jako stránky – rámce
-převod adres stránkovací tabulkou(příznak existence mapování) výpadek stránky
-výpadek-potřeba znovu načíst stránku do rámce, zavést mapování a obnovit kontext
-str.tabulky-v hl.paměti(problém-velké,pomalé)->víceúrovňové str.
-nulaúrovňové str. využívá pouze TLB(řízena i OS)
-inverzní str.tab - FAP menší než VAP (64-bit CPU) - pomocí hashování
-hashTable-velikost jako fyz.paměti procesu
-může nastat kolize v hashT-spoják,když heldám stránku, musím ho projít(spoj. krátké 1-2)

segmentace
Stránkování chápe virtuální adresový prostor jako jednorozmerný. Segmentace zavádí dvojrozmerný virtuální adresový prostor(FAP 1D). Segment je nezávislý adresový prostor sestávající z adres od 0 do limitu segmentu. Segmenty mohou mít ruzné velikosti, a ty se mohou menit v prubehu casu. Segment má své umístení ve fyzické pameti (neviditelné pro procesy). Kazdá adresa se pak sestává z dvojice (s, d), kde s je císlo segmentu a d je adresa v segmentu. Segmenty mohou být prítomny nebo neprítomny a dochází k výpadkum segmentu (obdoba výpadku stránky).

při výpadku nutné vyměnit celý segment a ty mohou být velké
lze sesypat segmenty ve FAP
nelze mít segment větší než FAP

Kombinace segmentace+stránkování
-odstraňuje nevýhody segmentace
-neprovádí se výpadky segmentů

NRU(Not Recetly Used)
-každá stránka příznaky A (Accessed) a D (Dirty)-uloženo v TLB
-typicky implementovány HW
-lze simulovat SW(pokud CPU nepodporuje A,D)
-jednou za čas se smažou všechna A
-při výpadku rozdělím stránky podle A, D
-vyberu stránku z nejnižší neprázdné třídy

LRU(Least Recently Used)
-ne pro fyz paměť, ale např. pro asoc.
-často používané str budou v nejbližších okamžicích znovu použity
-lze implementovat v HW
-64-bitový čítač, zápis při použití
-matice n x n bitů (složité)

druhy a obsluha preruseni
- druhy - synchronní(záměrné - trap, výjimky - nesprávné chování procesu)
  - asynchronní (vnější události)
  - polling (kontrola stavu zařízení)
- obsluha - OS se ujme řízení
  - uloží se stav CPU
  - analýza přerušení
  - vyvolání přísl.obsluhy
  - obslouží se přerušení
  - obnova stavu CPU (může přeplánovat)
  - aplikace pokračuje

Definuj semafor

-implementovan OS
-každý vlastní frontu
-čítač a fronta uspaných procesů
-atom. operace DOWN/UP
-DOWN-když >0 dec a pokrač, =0 zablokování, proces do fronty čekajících
-UP-nepr.fronta -probudí nějaký proces za DOWN, jinak inc(čítač)
NEBO
-DOWN-vždy dec(čítač), když >=0, pokrač, jinak zablokování
-UP-vždy inc(čítač), <=0 a nepr.fronta-odblokuje libovolný proces

algoritmy pridelovani pameti

First-fit-první volný dostatečné velikosti(rychlí,ale občas rozdělí velkou díru)
Next-fit-jako first-fit(začíná se na poslední prohledávané pozici)-jakoFF ale rychlejší
Best-fit-nejmenší volný dostatečné velikosti(pomalý,velké díry nechává vcelku, ale malé kam se nic nevejde)
Worst-fit-největší volný(pomalý-prohledá celý seznam, dělí velké díry)

Klasicke synchronizacni procesy

strucna charakterizace utoku na kryptograficke systemy

Model utocnika podle Doleva a Yao
Může získat libovolnou zprávu putující po síti
Je právoplatným uživatelem sítě a tudíž může zahájit komunikaci s jiným uživatelem
Může se stát příjemcem zpráv kohokoliv
Může zasílat zprávy komukoliv zosobněním se za jiného uživatele
Nemůže odhadnout náhodné číslo z dostatečně velkého prostoru
Bez správného klíče nemůže nalézt zprávu k šifrované zprávě a nemůže vytvořit platnou šifrovanou zprávu z dané zprávy, vše vzhledem k nějakému šifrovacímu algoritmu

strucna charakterizace utoku na kryptograficke systemy

utoky na kryptograficke protokoly
-útok na veřejné klíče

-přehrání zpráv-M odposlouchá zprávy, a pak totéž udělá sám(řešeno nějakým časovým razítkem)
-muž uprostřed-M se vydává z obou stran za toho druhého-řešeno ověřením, že A je A -paralelní spojení-několik běhů protokolů prováděných současně pod řízením M
-odražení (reflection)-A zahájí komunikaci, M zachytí zprávu, upraví ji, aby nebyl poznat původní A a pošle ji zpět A
-prokládání-několik běhů protokolu prováděných současně pod řízením M, zprávy z jednoho se použijí u dalšího, atd.
-chyba typu-nedodržení přesného sémantického významu zprávy
-vypuštění jména-pokud v protokolu není poznat, kdo za to může
-chybné použití šifrovací služby
-špatný algoritmus použitý na nevhodném místě

Druhy kryptografickych systemu

(symetricke/asymetricke)
symetricky klíčované šifry-blokové šifry (DES, IDEA, AES),proudové šifry (SEAL)
+šifry s vysokou datovou propustností,krátké klíče,slučování více šifer vytváří silnější šifru
-klíče na obou koncích musí zůstat utajeny,je třeba často měnit klíče
šifry s asymetrickým klíčem-RSA, DSA (ElGamal)
+pouze privátní klíč je tajný,klíče je možné měnit méně často
-mnohem pomalejší než symetrické šifry,klíče mnohem delší,o žádném schématu veřejného klíče nebylo dokázáno, že je bezpečné

Kryptograficky system
-důvěrnost,celistvost,autentikace(vím,od koho to je),nemožnost popření(když potvrdím, nelzepopřít)
-kr.alg,kr.protokoly
-kr.systémy-symetrické(sdílený klíč),asymetrické (veřejný klíč)

ukladanie suborov na disk

-souvislá alokace (souvislý sled bloků-kde uloženo stačí č.1. bloku)
 -lepší práce s diskem
 -problém při hledání volného místa
 -problém při zvětšování souborů
-spojovaná alokace
 -pospojování bloků použitých pro soubor
 -modifikace FAT – přemístění spojového seznamu do speciální oblasti disku
-Indexová alokace - UNIX a i-node

K cemu slouzi souborovy system
Ukoly souboroveho systemu.
Význam souborových systémů
 - uchovat prezistentní data (data přežijí konec procesu)
 - udržovat informaci o volném místě
 - vytvořit adresářovou strukturu(uložení atributů)
 - sdílení informace mezi procesy
 - správa souborů-odkaz na data která jsou v souboru umístěna
 - pojmenování souborů-zvednutí abstrakce

pametove mapovane soubory

vrstvy I/O systemu, zpravy
Vrstvy I/O subsystemu (patri sem to, co je na slajdech oznaceno jako I/O software)
 -Uživatelský I/O software
 -I/O nezávislý subsystém
 -Ovladače zařízení
 -Obsluha přerušení
 -HW
-poskytuje uživatelským programům jednotné rozhraní
-skrývá rozdíly mezi jednotlivými zařízeními
-provádí pomocné úkoly pro ovladače-alokace paměti,hlášení chyb,vyrovnávání (buffering)

zneni znamych synchronizacnich problemu

-na alegoriích ukazují časté problémy při synchronizaci
-problém producent-konzument
-problém obědvajících filosofů
-problém ospalého holiče

planovani procesu

 -entita(proces/vlákno)
 -přidělování CPU entitám
 -plánovač (scheduler)-rozdává procesor
 -preemt.plánování - větší OS
 -nepre. pl. (kooperativní multitasking)
   -OS nemá prostředky aby odstřih proces od procesoru
   -apl. komunikuje s CPU
 cíle
 -každý proces dostane CPU
 -efektivnost - co nejvíc dávat CPU aplikacím
 -rychlá doba odezvy
 kriteria k naplánování
 -práce s CPU ? I/O
 -priorita =(static-důležitost + dynamic-spravedlnost)
 SMP(symetr.multiprocesor)
 -fronta CPU čeká na procesy(aktivně/pasivně-až je nědo vzbudí)
 Real time
 -plnávání NP úplný problém
 -aplikace řízené událostmi
 -každý úkol realný čas k dokončení

planovaci algoritmy
 -FIFO - nepreemptivní,kdo dřív přijde, dříve mele
 -Round Robin (RR) - preemptivní, podobné jako FIFO, ale plánovač přiřazuje procesu čas (time slice)
 -více front se zpětnou vazbou(FIFO,RR)

reseni zablokovani
-pštrosí alg.-vše se nechá na uživateli
-detekce hledání kružnice v grafu
-zotavení-opatrné odebrání prostředků(na přechodnou

dobu)
-zabíjení procesů-zrušení cyklu
-rollback-jeden proces se vrátí

Definujte monitor.

Monitory
Pro zjednodusení práce a zamezení chyb pri pouzití semaforu bylo navreno synchronizacní primitivum monitor.
Monitor je soubor funkcí, promenných a datových struktur, který jsou sdruzeny do zvlástního balíku. Procesy smí volat funkce z monitoru, kdykoliv chtejí, ale nemají prístup k promenným monitoru z funkcí mimo monitor. Navíc monitor zarucuje, ze v jednom okamziku muze být aktivní nejvýse jeden proces v jedné instanci monitoru.

Monitor je konstrukcí programovacího jazyka a prekladac tudíz ví, ze funkce monitoru je treba prekládat jinak. Závisí na prekladaci, jak implementuje monitory (typicky pomocí semaforu). Protoze vzájemné vyloucení zarizuje prekladac a ne programátor, zabrání se mnoha chybám. Stací pouzevedet, ze vsechny kritické operace je treba provádet v monitorech.
Blokování procesu uvnitr funkcí monitoru pomocí podmínených promenných a operacemi WAIT a SIGNAL na nich definovaných. Podmínené promenné nejsou cítace (na rozdíl od semaforu, je tam fronta zablokovaných procesů).
Kdyz funkce monitoru zjistí, ze nemuze pokracovat, volá operaci WAIT a zablokuje aktivní proces v monitoru. To umozní jinému procesu vniknout do monitoru.
Jiný proces muze být probuzen pomocí operace SIGNAL (jeden zcekajících z mnoziny zablokovaných procesu na dané podmínené promenné). Tady ale vzniká problém, ze v jednom okamziku jsou dva procesy v jedné instanci monitoru.

Existují dve resení:
1. Spustit probuzený proces a druhý uspat
2. Proces volající SIGNAL okamzite opustí monitor.

Monitor stručně
-pro zjednodušení a zamezení chyb semaforu
-obdoba třídy v C++ (promenné privátní,funkce i public)
-v jedné instanci m. může být aktivní jeden proces
-implementovan pomoci semaforu (není čítač, ale fronta blok. procesů)
-vzajemne vylouceni zařizuje překladač - zabrání se chybám
-všechny kritické operace třeba provádět v monitorech
-když fce motitoru nemůže pokračovat, volá Wait - blokace procesu
-jiný proces pak může vniknout do monitoru
-nebo probuzeni čekajícího procesu pomocí SIGNAL
-problém 2 v jednom monitoru
1)spustit probuzený, druhý uspat
2)probouzející okamžitě opustí monitor

Coffmanovy podmínky + zablokování

Zablokování
Mnozina procesu je zablokována, jestlize kazdý proces z této mnoziny ceká na událost, kterou muze zpusobit pouze jiný proces z této mnoziny.

Ctyri nutné podmínky pro zablokování (Coffman) (musí platit všechny aby bylo zablokováno)
Vzájemné vyloucení - Kazdý prostredek je bud pridelen práve jednomu procesu, nebo je volný.
Drz a cekej - Procesy aktuálne vlastnící nejaký prostredek mohou zádat nové prostredky.
Neodnímatelnost - Pridelené prostredky nemohou být vzaty nazpátek hrubou silou. Musí být vráceny explicitne procesem, který je vlastní.
Cekání do kruhu - Existuje kruhový retez procesu, kde kazdý z nich ceká na
prostredek, který je drzen dalsím clánkem retezu.

Prostredky
Prostredek je cokoliv, co muze být v jednom casovém okamziku pouzito nejvýse jedním procesem. Odnímatelné mohou být procesu odejmuty bez jakýchkoliv následku Neodnímatelné nelze je odejmout bez vázných následku (vcetne selhání procesu) Zablokování zpusobují neodnímatelné prostredky. Zablokování na odnímatelných prostredcích muze být odstraneno prevedením pozadovaných prostredku z jednoho procesu na jiný.)

Asociativní pamet

Reší problém rychlosti prístupu do stránkovacích tabulek v pameti. Vyuzívá lokality programu, tj. program v jistém casovém úseku využívá pouze nekolik
stránek pameti. Asociativní pamet zabranuje pruchodu stránkovacími tabulkami pri
kazdém pametovém odkazu. Obsahuje nekolik položek (rádove desítky) a každá
položka je delena na dve cásti: klíc a hodnotu. V klíci jsou císla stránek a
hodnoty jsou císla rámcu. Pri prevodu virtuální adresy na fyzickou adresu se nejprve paralelne prohledává asociativní pamet na císlo stránky podle klíce. Pokud tento dotaz na
asociativní pamet uspeje, již se nepristupuje do stránkovacích tabulek. Pokud v
asociativní pameti hledané císlo stránky není, použije procesor stránkovací tabulky pro nalezení mapování. Pokud mapování uspeje, je zapsáno do asociativní pameti (na úkor jiné položky). Pokud ne, nastane výpadek stránky.