Operačné systémy

Manažment Pamäte 1.

Ing. Martin Vojtko, PhD.

2024/2025

Požiadavky

Požiadavky na pamäť

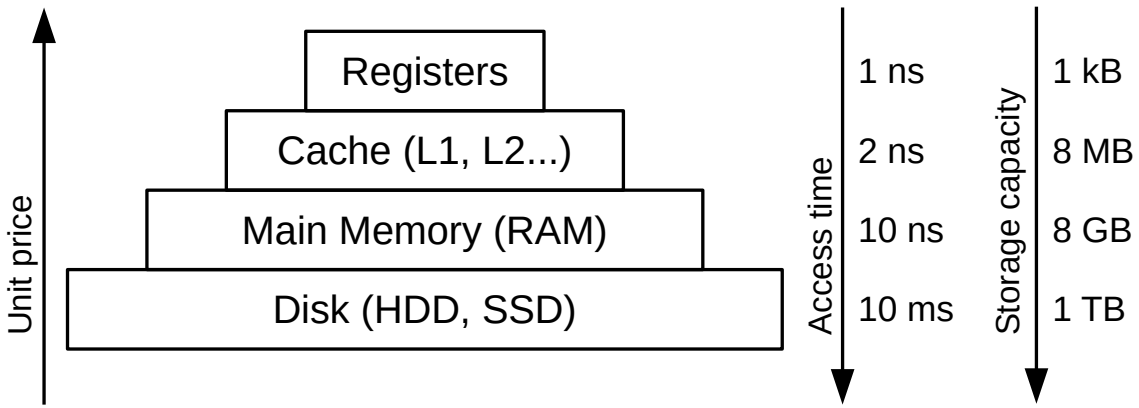
- Používateľ chce mať privátnu, nekonečne veľkú, nekonečne rýchlu, trvalú a lacnú pamäť.
  - Súčasné PC majú 10000 krát viac pamäte ako najväčšie počítače zo 60-tých rokov.
  - Napriek tomu, že množstvo dostupnej pamäte rastie nikdy jej nie je dosť.

Parkinsonov zákon

Programy majú tendenciu naplniť celú pamäť, ktorá je im k dispozícii.

Požiadavky na pamäť

- Čiastočné naplnenie požiadaviek sa rieši hierarchickým rozdelením pamäte.
- OS vie spravovať hlavnú pamäť a úložisko na pevnom disku.



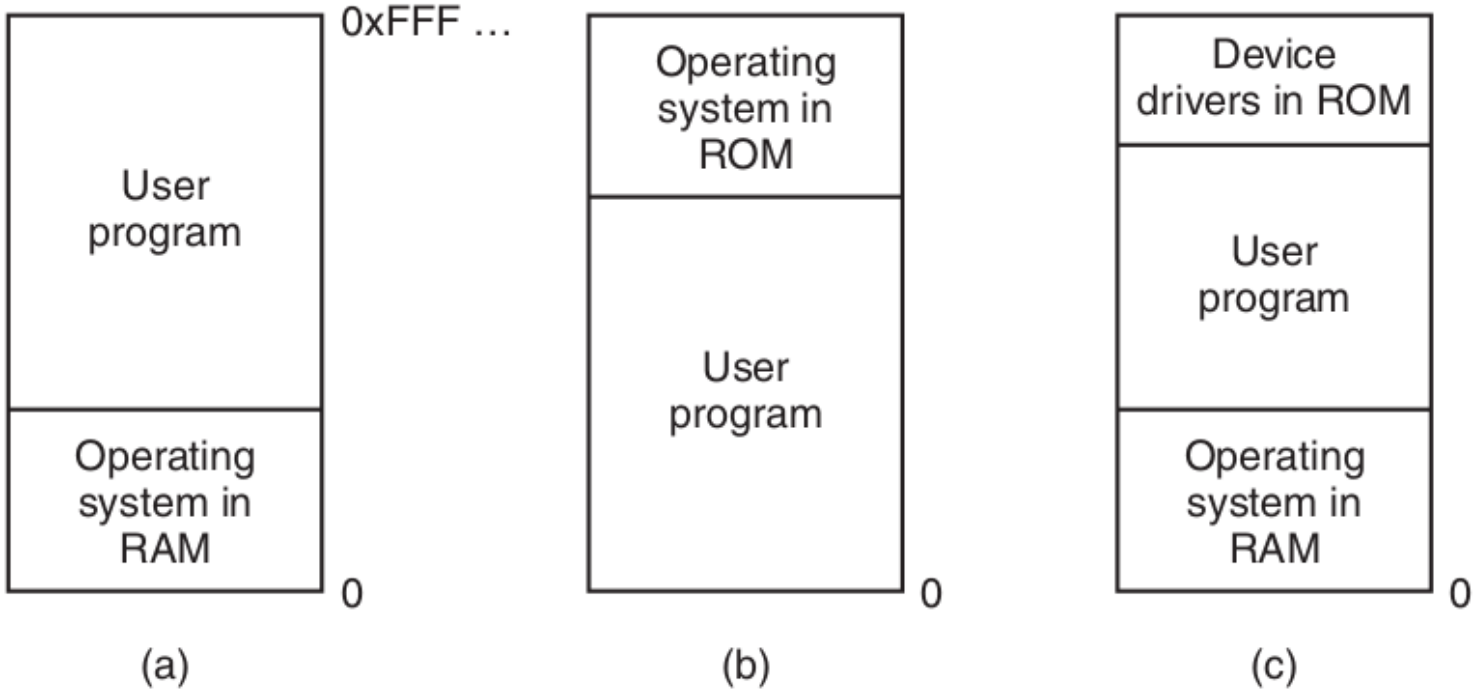
Žiadna abstrakcia

Žiadna abstrakcia

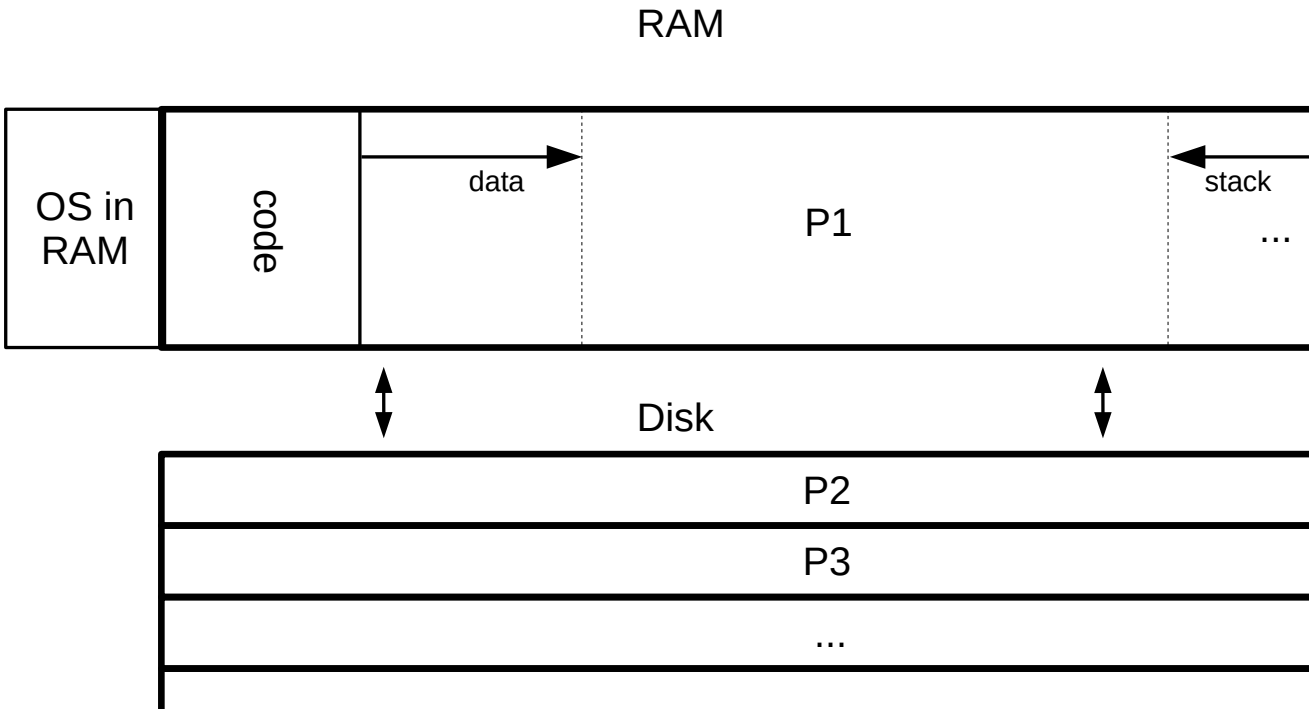
- Pre OS je najjednoduchšie nemať abstrakciu pamäte.
- Každá generácia výpočtových systémov začínala takto.
- Proces ako taký vidí fyzickú pamäť.
  - Prakticky nie je možné udržiavať dva a viac programov v pamäti.
  - Proces zapisujúci na konkrétnu fyzickú adresu bez obmedzení prepíše iný proces.
  - Proces zapisujúci na konkrétnu fyzickú adresu môže prepísať OS.
  - Pamäť nie je rozlišovaná na pamäť programu a dáta procesu.

Žiadna abstrakcia - jeden proces

- Možným riešením je:
  - vyhradenie pamäte pre proces a OS v RAM. (a) Programátor alebo kompilátor musí zaručiť, že nebude zasahovať mimo tejto pamäte.
  - vyhradenie pamäte programu OS do ROM. (b)
  - ...

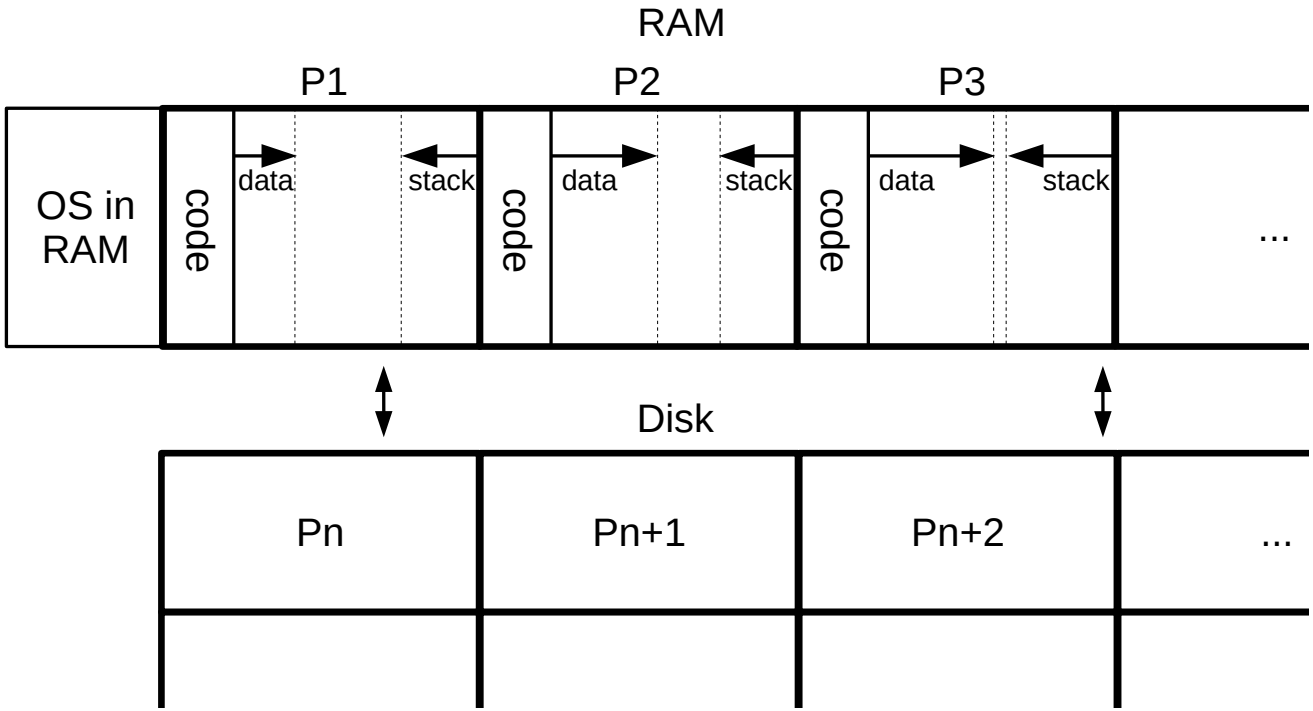


Žiadna abstrakcia - jeden proces



- Procesy sa dajú striedať prostredníctvom disku.
- Kompilátor zaručuje správne umiestnenie procesu v pamäti.
- Všetky procesy majú rovnako veľkú pamäť.
- Čítanie z disku je nákladné preto preplánovanie nie je časté.
- Na takomto princípe fungovali ranné batch systémy.

Žiadna abstrakcia - viac procesov

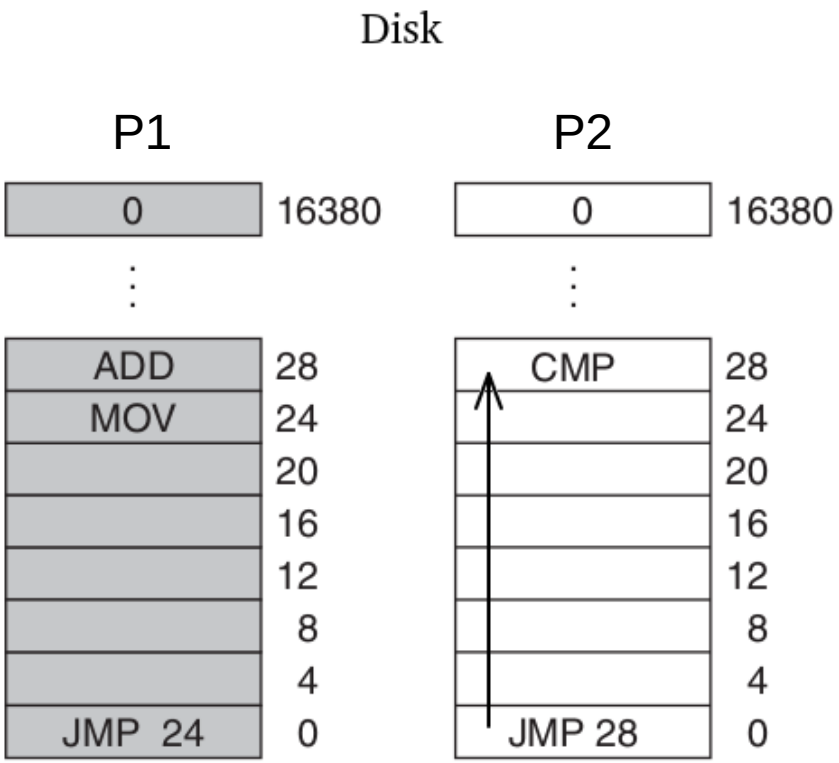


- Jeden proces v pamäti je veľmi neefektívny.
- Riešením je rozdelenie pamäte na viac blokov rovnakej veľkosti.
- V každom bloku môže byť jeden proces.
- Veľkosť procesu je limitovaná veľkosťou bloku.
- Vzniká problém relokácie procesu.

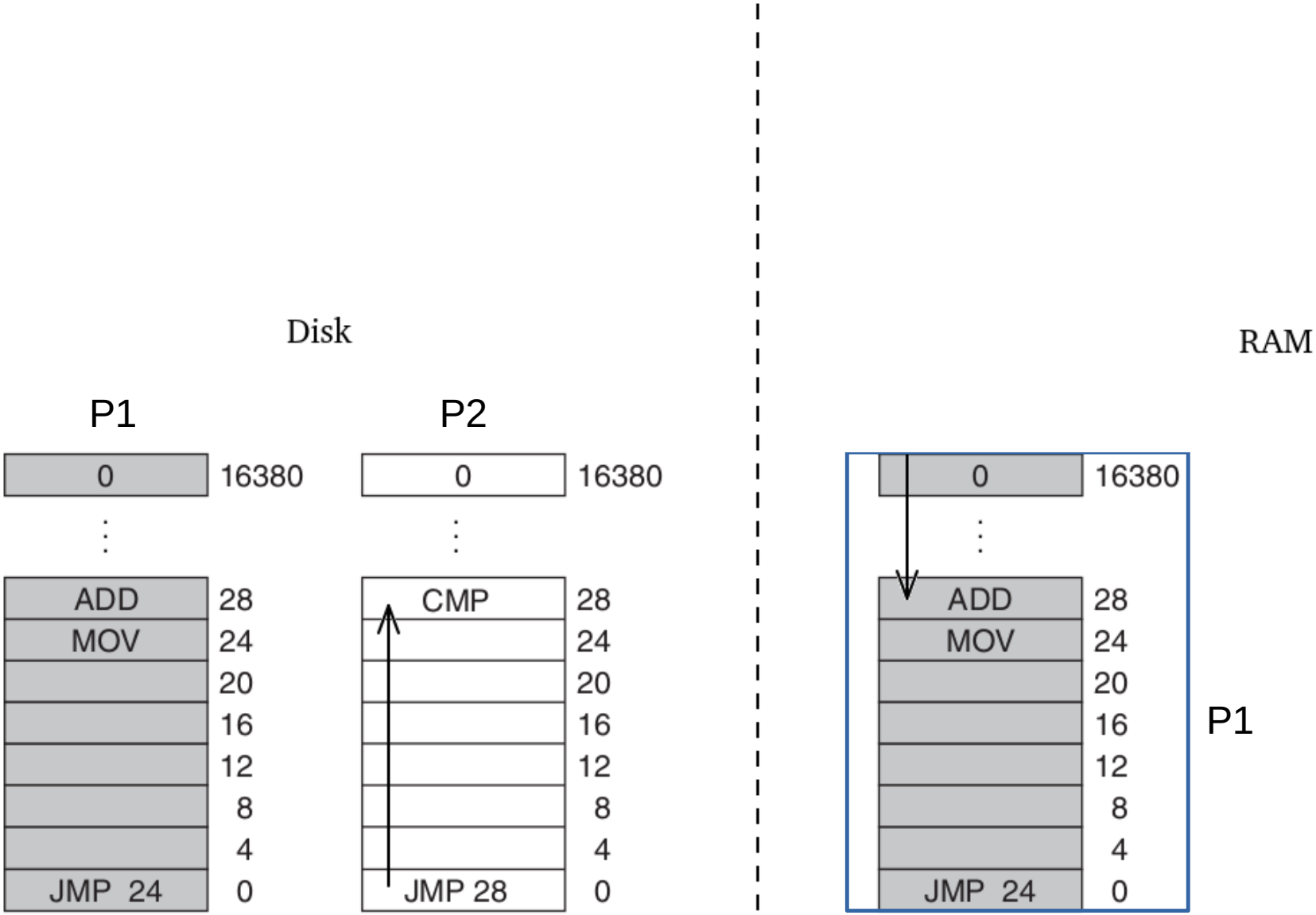
Žiadna abstrakcia - problém relokácie

- Každý program je výsledkom kompilácie.
- Kompilátor bežne kompiluje program od adresy 0.
- Hneď za pamäťou programu nasledujú dáta rastúce hore.
- Obyčajne na koniec adresného priestoru je umiestnený zásobník rastúci dolu.
- Akonáhle program načítame z disku, tak nastáva problém, pretože blok kam sme program uložili nezačína na adrese 0.

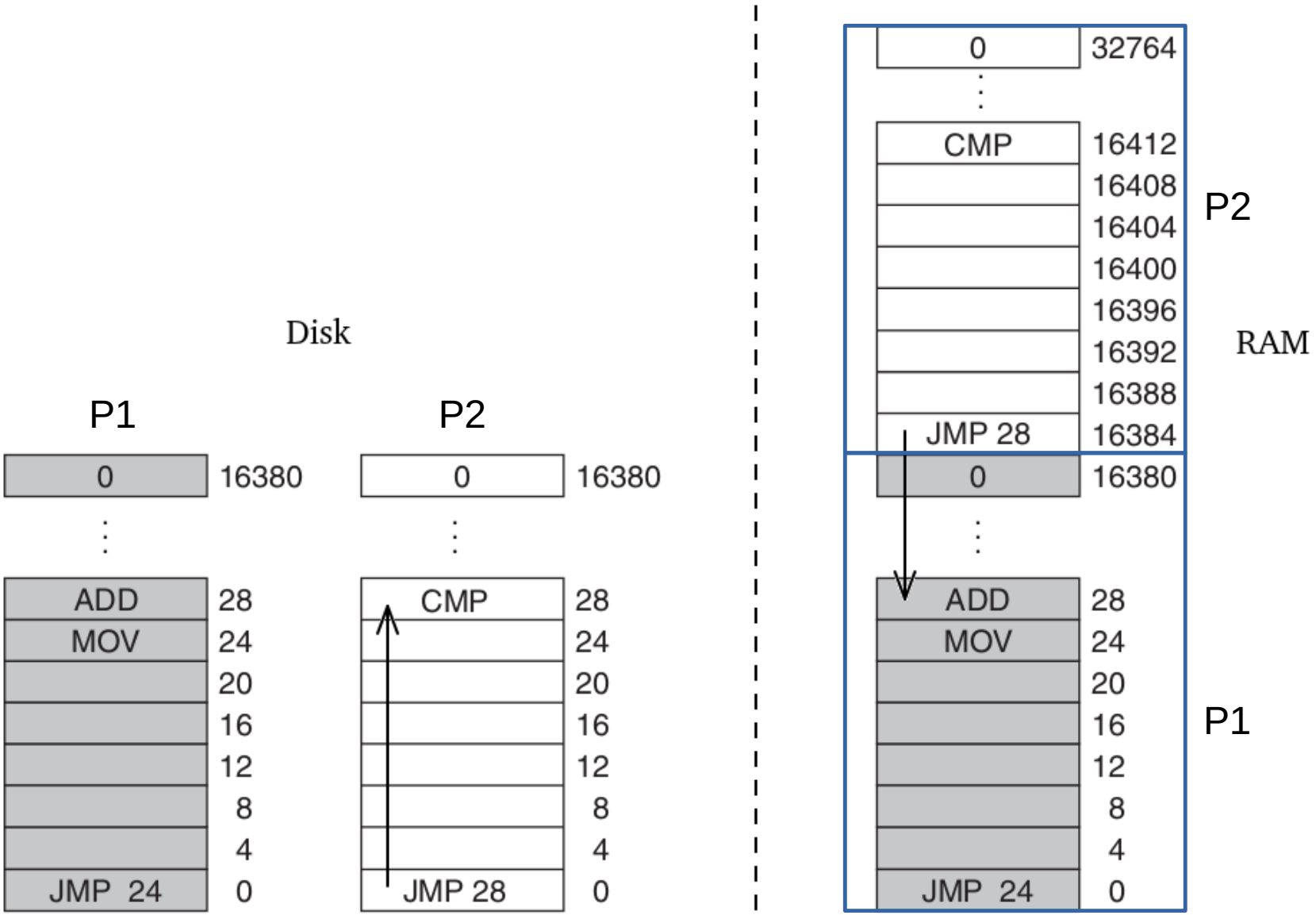
Žiadna abstrakcia - problém relokácie



Žiadna abstrakcia - problém relokácie



Žiadna abstrakcia - problém relokácie



Žiadna abstrakcia - problém relokácie - riešenie

Statická relokácia

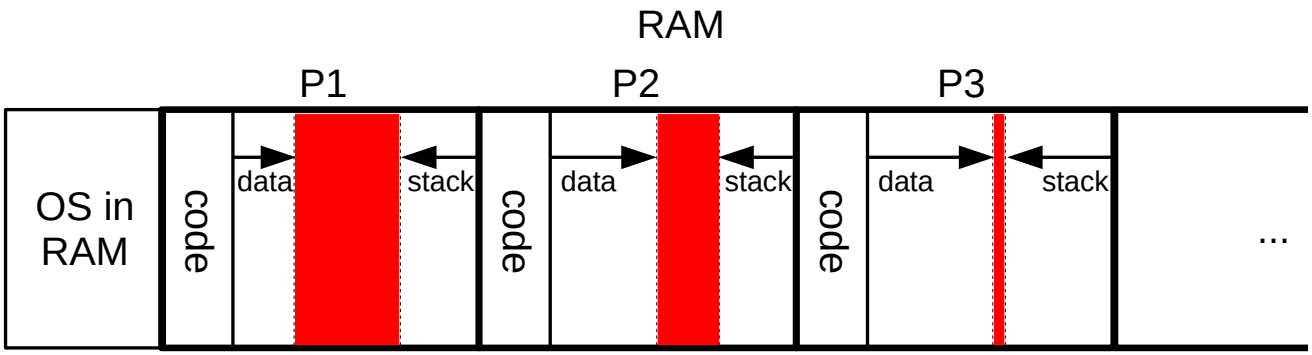
Počas načítania programu do pamäte sa kód prechádza a pripočíta sa ku každej adrese konštanta. Táto konštanta je rovná počiatočnej adrese bloku.

- Načítanie programu je výrazne dlhšie pretože OS musí najprv vykonať relokáciu.
- Načítanie programu do pamäte má za úlohu zavádač.
- Fixné bloky sú značne zaväzujúce.
  - Malé bloky obmedzujú veľkosť programu. Relokácia je rýchlejšia.
  - Veľké bloky plytvajú pamäťou. Relokácia je pomalšia.
- Program stále môže poškodiť iný program ak neexistuje HW ochrana pamäte programu.

Žiadna abstrakcia - interná fragmentácia

Interná fragmentácia

je rozdrobenie voľnej nepoužívanej pamäte vo vyhradených adresných priestoroch procesov.



Abstrakcia Pamäte

Abstrakcia pamäte

- Prístup procesov k fyzickej pamäti výpočtového systému je zaručeným receptom na problém.
- V pamäti máme veľa procesov. Tieto navyše počas výmeny musí OS neustále relokovať.
- Fixná veľkosť blokov pridelenej pamäte programu obmedzuje programátora.
- Je nevihnutná abstrakcia, ktorá:
  - čas relokácie skráti,
  - procesy obmedzí len do vyhradeného priestoru a
  - rozviaže ruky programátrom.

Adresný priestor (Address Space)

- Podobne ako proces je abstraktným CPU, OS definuje adresný priestor ako abstrakciu pamäte.

Adresný priestor

je množina adries, ktoré môže proces použiť. Každý proces má vlastný adresný priestor nezávislý od ostatných procesov.

- Izolácia adresného priestoru procesu od fyzickej pamäti zjednodušuje:
  - zavádzanie programu,
  - ochranu pred chybami.
- Procesy môžu pamäť aj zdieľať ale za pevne stanovených podmienok.

Adresný priestor (Address Space)

- Riešenie pomocou SW by bolo extrémne pomalé. OS by musel sám každú inštrukciu skontrolovať a vykonať prípadnú statickú relokáciu.
- Adresný priestor teda nie je možné realizovať bez pomoci HW.

Base register (Bázový register)

je register, ktorý definuje aktuálny počiatok adresného priestoru procesu. Umožňuje adresovať proces vždy od 0.

Limit register

je register, ktorý definuje maximálnu povolenú adresu adresného priestoru. Umožňuje vykonať kontrolu procesov.

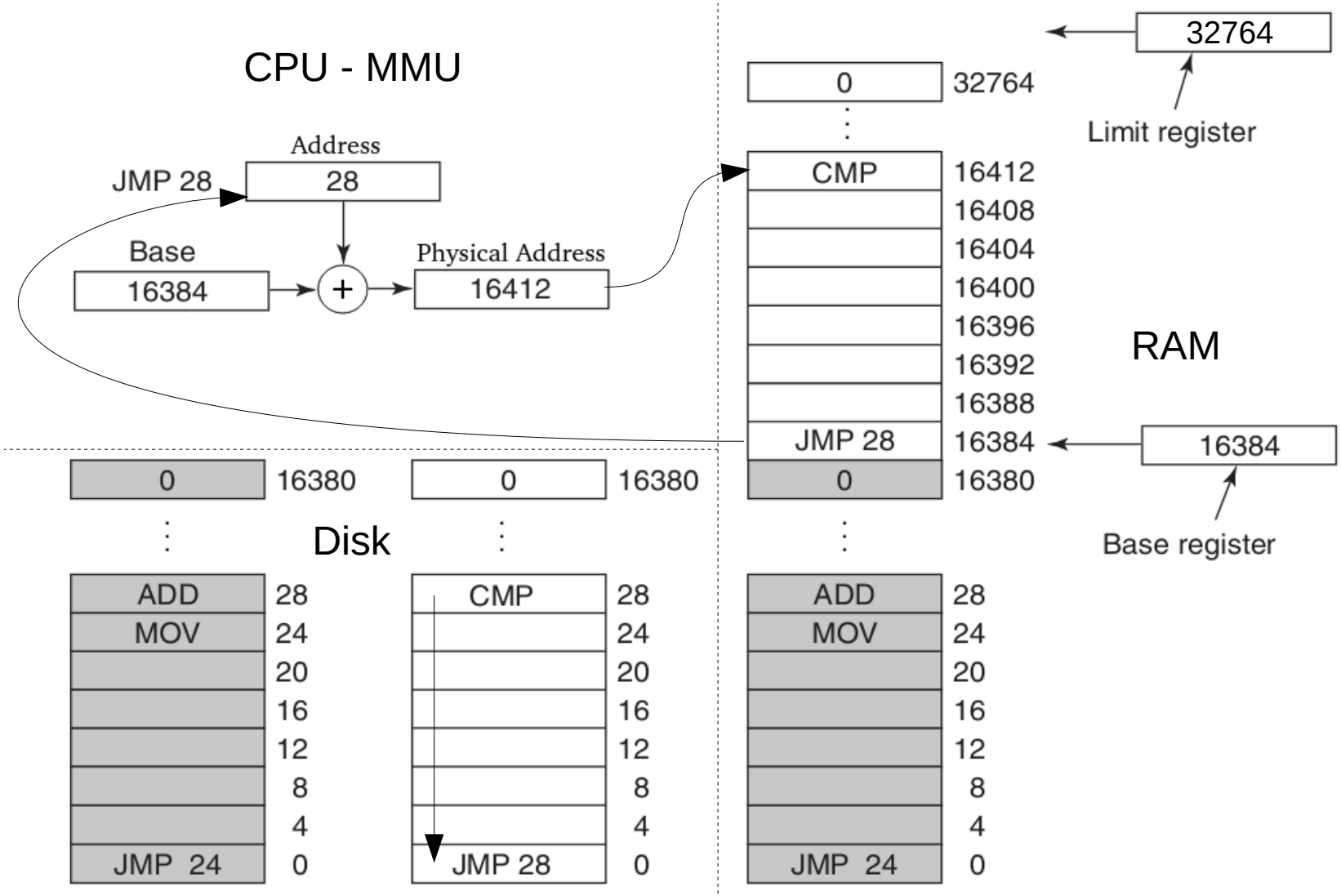
- Base a Limit register vymedzujú adresný priestor procesu.
- Tento adresný priestor môžeme označovať aj ako segment.

Adresný priestor - problém relokácie

Dynamická relokácia

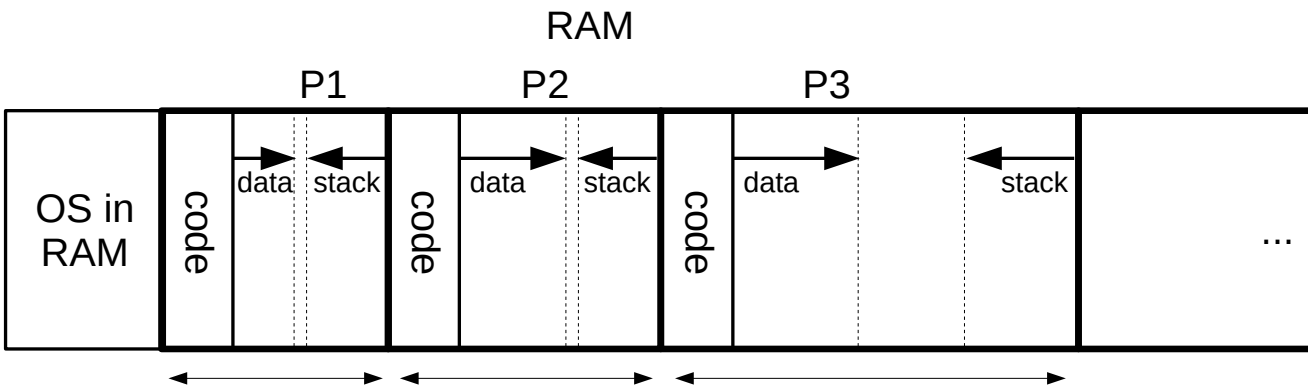
je okamžitá relokácia adries realizovaná pripočítaním Base registra.
Dynamická relokácia sa vykonáva na akúkoľvek operáciu s pamäťou.
Relokácia je vykonaná pomocou HW bez toho aby SW o tom vedel.

Adresný priestor - problém relokácie



Adresný priestor - problém relokácie

- Base a Limit registre umožňujú:
  - rýchlu relokáciu
  - procesy s rôznou veľkosťou adresného priestoru.
- Na druhú stranu vyžadujú komplikovanejší HW:
  - sčítačku pri výpočte fyzickej pamäti.
  - komparátor pri kontrole maximálnej adresy.



Odloženie procesov na disk

- Doteraz sme pracovali s predpokladom, že výpočtový systém pracuje s takým počtom procesov aký dokáže uložiť do hlavnej pamäte.
- Proces bol udržiavaný v hlavnej pamäti kým neskončil a až potom bol nahradený iným.
- Takéto riešenie je v bežnom OS neakceptovateľné. Pamäte je málo.

Odloženie do pamäte

V bežnom PC OS sa pri jeho štarte spustí od 50 do 100 procesov. Po prihlásení používateľa sa toto množstvo násobí. Pri takomto počte procesov bežná RAM nepostačuje, a preto sa vybrané procesy alebo časť ich pamäte presúva na pevný disk. To aký proces je v RAM rozhoduje aktivita daného procesu. O všetko sa postará OS.

Odloženie procesov na disk

Swapping

Pri uvoľňovaní hlavnej pamäte je odložený celý adresný priestor procesu. Jeho dáta aj jeho program. Obeťami swap-u sú blokované a málo aktívne procesy.

Virtual memory

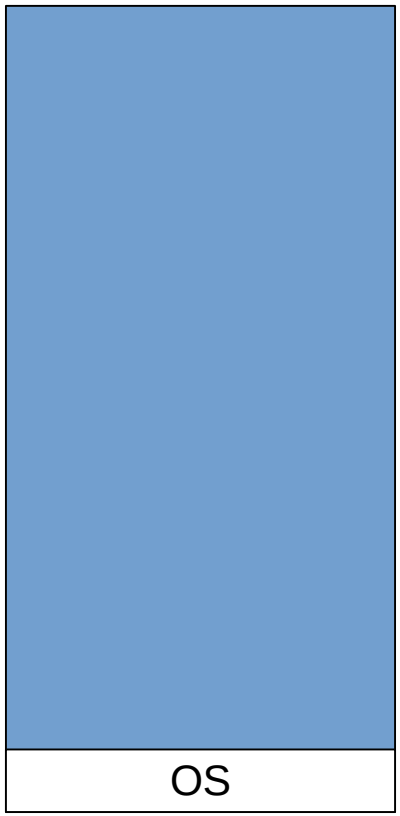
Pri uvoľňovaní hlavnej pamäte sú odkladané časti procesov, ktoré sú v danom čase nepotrebné. Proces môže byť vykonávaný i keď nie je celý v pamäti.

Swapping

- Celý proces (dáta, zásobník a kód (program)) je odložený na disk do tzv. SWAP partície.
- Na disku je vyhradený priestor, kde sa môžu odkladať procesy.
- Odloženie procesu riadi OS.
- Dôvodom na odloženie na disk môže byť:
  - Proces je dlhodobo neaktívny, čaká na prostriedok.
  - Vznikol nový proces s vyššou prioritou.
  - Pamäť dosahuje prahovú úroveň obsadenia.
- algoritmy swap-u udržujú dostatočne veľký voľný priestor v RAM. OS často začína swap-ovať už pri 60% vyťažení RAM.

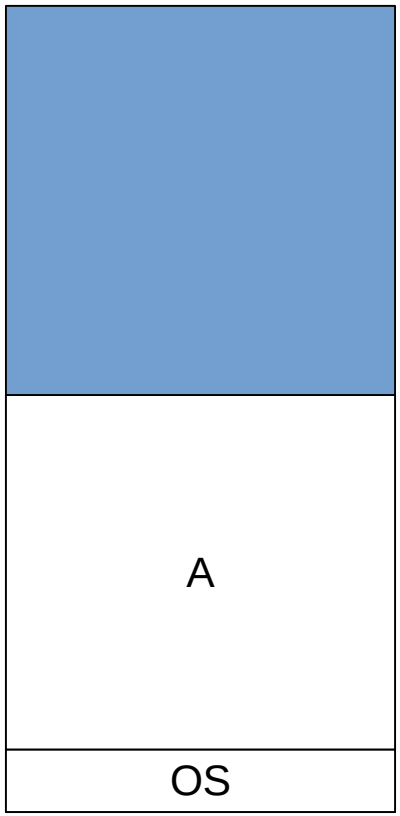
Swapping

- OS udržuje informácie o voľnej pamäti.
- Proces načítaný zo swap-u môže skončiť na inej adrese.
- Base a Limit registre problém premiestňovania odstránia.



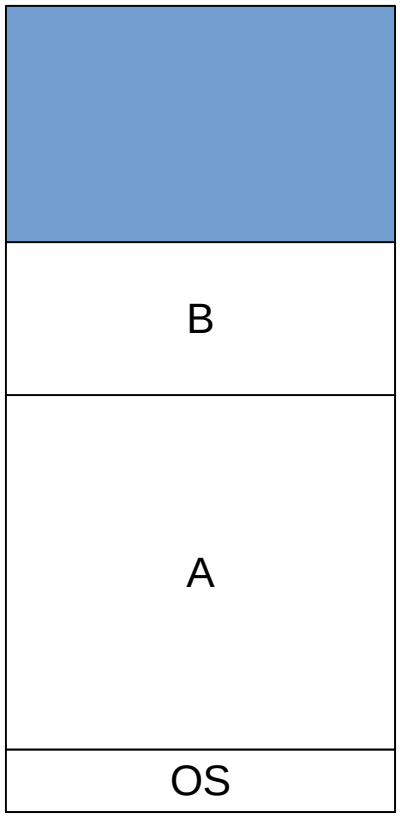
Swapping

- OS udržuje informácie o voľnej pamäti.
- Proces načítaný zo swap-u môže skončiť na inej adrese.
- Base a Limit registre problém premiestňovania odstránia.



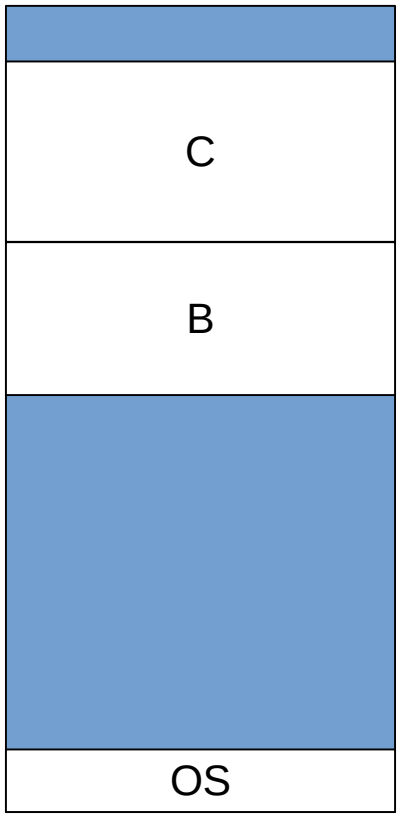
Swapping

- OS udržuje informácie o voľnej pamäti.
- Proces načítaný zo swap-u môže skončiť na inej adrese.
- Base a Limit registre problém premiestňovania odstránia.



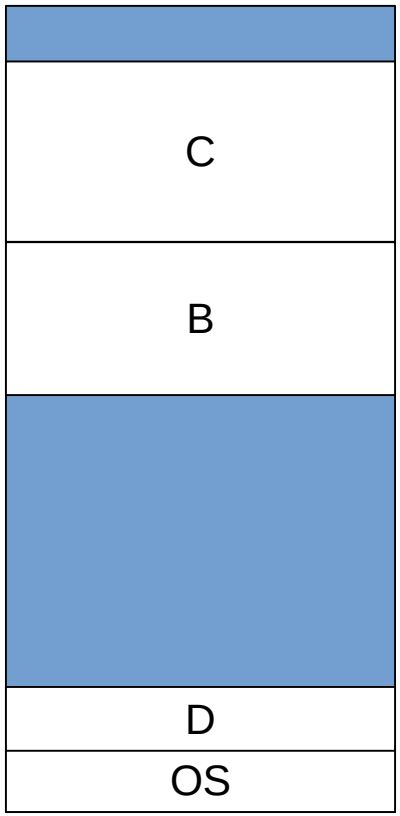
Swapping

- OS udržuje informácie o voľnej pamäti.
- Proces načítaný zo swap-u môže skončiť na inej adrese.
- Base a Limit registre problém premiestňovania odstránia.



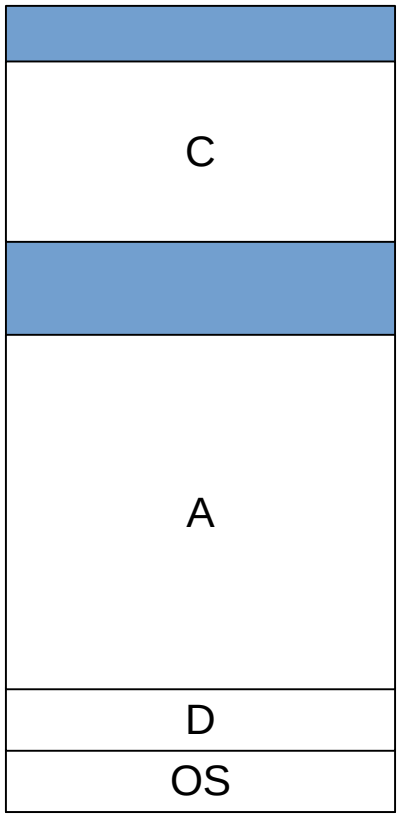
Swapping

- OS udržuje informácie o voľnej pamäti.
- Proces načítaný zo swap-u môže skončiť na inej adrese.
- Base a Limit registre problém premiestňovania odstránia.



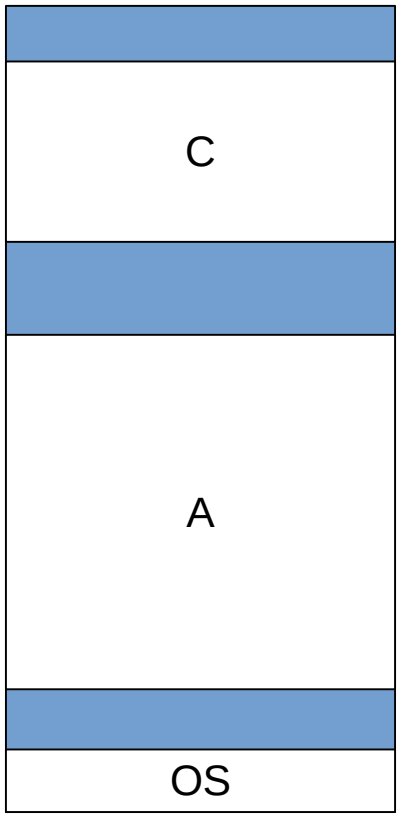
Swapping

- OS udržuje informácie o voľnej pamäti.
- Proces načítaný zo swap-u môže skončiť na inej adrese.
- Base a Limit registre problém premiestňovania odstránia.



Swapping

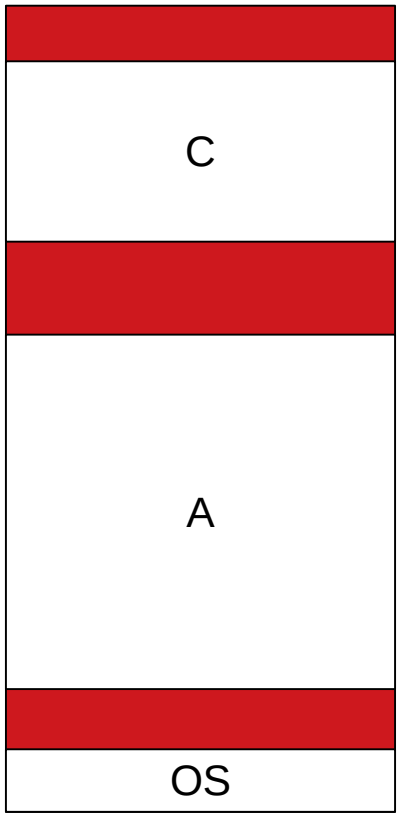
- OS udržuje informácie o voľnej pamäti.
- Proces načítaný zo swap-u môže skončiť na inej adrese.
- Base a Limit registre problém premiestňovania odstránia.



Swapping - externá fragmentácia

Externá fragmentácia

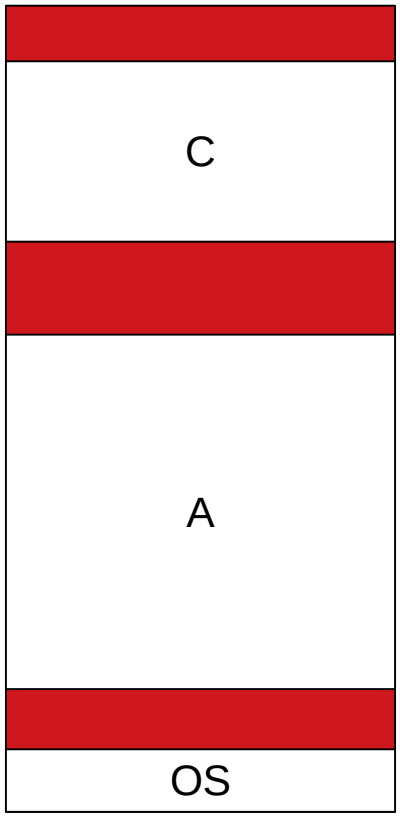
je rozdrobenie voľnej pamäte pri postupnom prideľovaní pamäte v blokoch rôznych veľkostí.



Swapping - externá fragmentácia

Kompakcia pamäte - defragmentácia

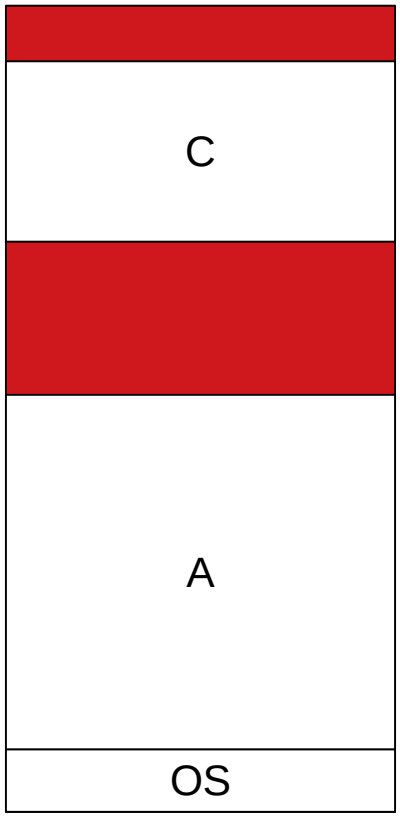
je technika spájania voľných fragmentov pamäte do jedného súvislého bloku. Pridelené bloky pamäte sa presúvajú v pamäti jeden za druhý.



Swapping - externá fragmentácia

Kompakcia pamäte - defragmentácia

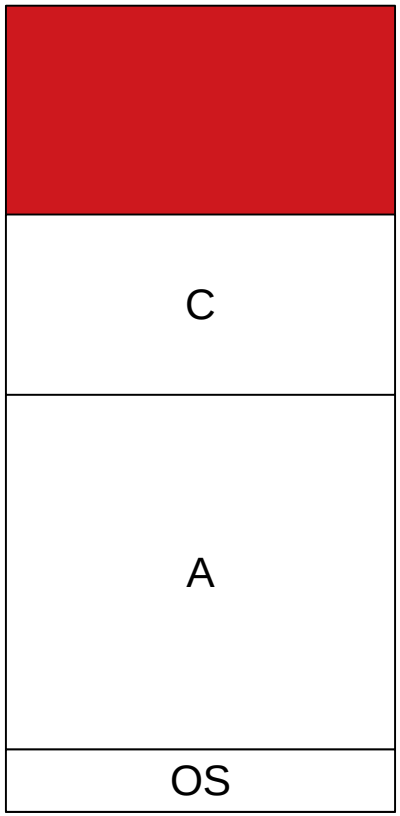
je technika spájania voľných fragmentov pamäte do jedného súvislého bloku. Pridelené bloky pamäte sa presúvajú v pamäti jeden za druhý.



Swapping - externá fragmentácia

Kompakcia pamäte - defragmentácia

je technika spájania voľných fragmentov pamäte do jedného súvislého bloku. Pridelené bloky pamäte sa presúvajú v pamäti jeden za druhý.



Swapping - externá fragmentácia

Kompakcia pamäte - defragmentácia

Kompakcia pamäte je nákladná operácia. Napríklad pre 16GB pamäte pri rýchlosti presunu 1B/ns bude kompakcia trvať 16s. Počas kompakcie OS vykonáva upratovanie. Toto upratovanie zaťažuje CPU. Procesy ktorých pamäť sa presúva nemôžu byť vykonávané.

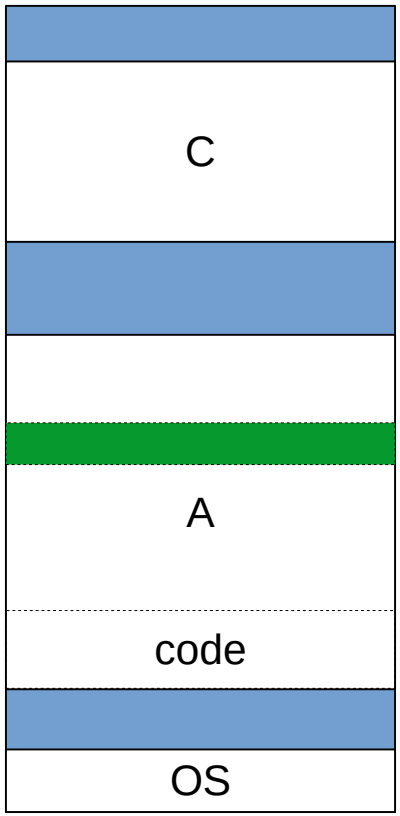
- Preto sa vo všeobecnosti kompakcia v RAM neujala.
- Našla svoje uplatnenie pri správe diskového priestoru.

Dynamický adresný priestor procesu

- Variabilné bloky pridelenej pamäte umožňujú zmenu veľkosti adresného priestoru procesu za behu.
- V procese je bežné, že dáta a zásobník rastú. Ak prerastú pridelený blok proces musí rásť do voľného fragmentu.
- Ak okolo procesu neexistuje dostatočne veľký priestor na rast:
  - môžeme ho premiestniť na inú pozíciu v pamäti.
  - susedný proces môžeme premiestniť do swap-u na disku.
  - proces môže spať kým v jeho okolí nie je dosť miesta.

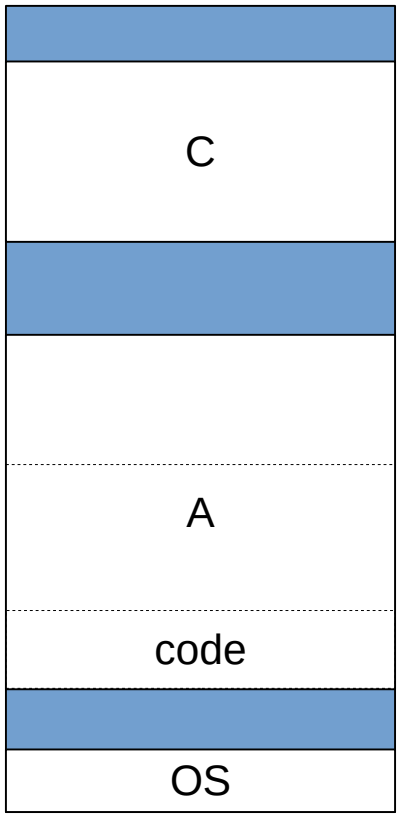
Dynamický adresný priestor procesu

- Proces bežne má pridelený priestor na rast zásobníka a dát.
- Ak sa minie je nutné zväčšiť blok procesu.
- Operácia je drahá, a preto sa vždy prideľuje väčší úsek pamäte.



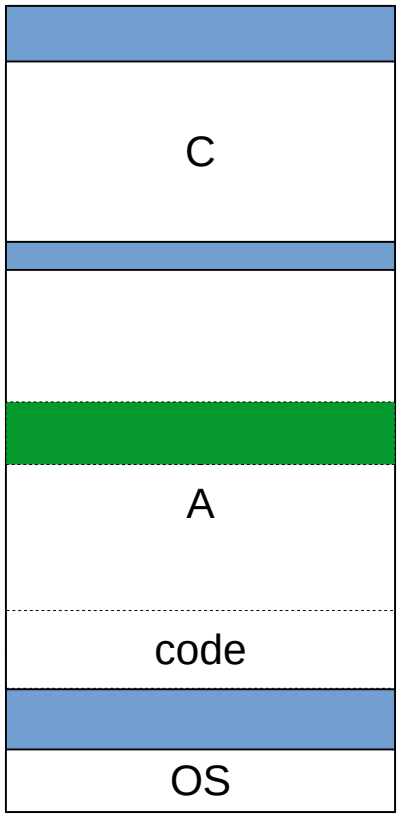
Dynamický adresný priestor procesu

- Proces bežne má pridelený priestor na rast zásobníka a dát.
- Ak sa minie je nutné zväčšiť blok procesu.
- Operácia je drahá, a preto sa vždy prideľuje väčší úsek pamäte.



Dynamický adresný priestor procesu

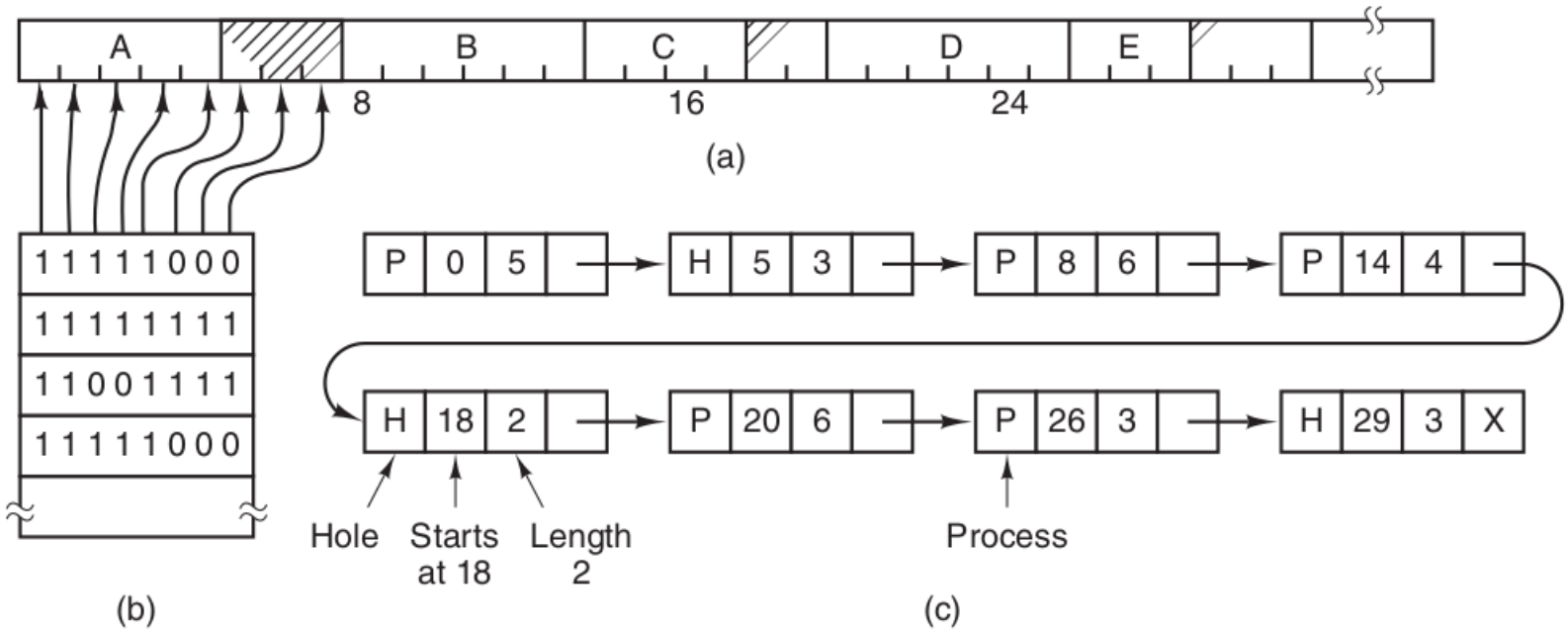
- Proces bežne má pridelený priestor na rast zásobníka a dát.
- Ak sa minie je nutné zväčšiť blok procesu.
- Operácia je drahá, a preto sa vždy prideľuje väčší úsek pamäte.



Manažment pamäte

Manažment voľnej pamäte

- Dynamické prideľovanie pamäte a swapping sú časťou správy pamäte
- Informácie o stave pamäte udržiavame buď:
  - bit mapou (b)
  - spájaným zoznamom (c)
- Prideľovanie pamäte riešime na úrovni OS a na úrovni procesov.



Manažment voľnej pamäte - bit mapa

- jeden bit v bit mape reprezentuje stav konkrétneho bloku pamäte
- veľkosť bloku definuje veľkosť bit mapy.
- bežné je stanoviť veľkosť bloku ako mocninu 2.
- požadovaný úsek pamäte sa musí vyhľadať prechádzaním bit mapy.

Bit mapa

Ak máme veľkosť bloku k = 32b (4B) a máme pamäť o veľkosti 16GB tak bit mapa bude mať veľkosť $16 * 2^{30}/(4 * 8) = 0.5GB$. Ak máme veľkosť bloku k = 4kB tak veľkosť bit-mapy bude $16 * 2^{30}/(4 * 2^{10} * 8) = 0.5MB$. Hľadanie voľných úsekov v takejto pamäti je nákladné. Ak hľadáme úsek pamäte o veľkosti $4 * k$ tak musíme nájsť taký úsek v bit-mape kde za sebou nasleduje $4 * 0$.

Manažment voľnej pamäte - spájaný zoznam

- stav úseku pamäte je uložený v položke spájaného zoznamu.
- jedna položka obsahuje:
  - stav (voľný/používaný) ??
  - adresu začiatku úseku
  - veľkosť úseku ??
  - adresu nasledujúceho úseku
  - adresu predchádzajúceho úseku ??
- položka zoznamu má dvoch susedov.
- veľkosť zoznamu nezávisí od veľkosti pamäte.
- susediace voľné úseky sa môžu zlučovať.

Manažment voľnej pamäte - algoritmy pridelenia

Pridelenie pamäte v bit mape

ak nájdem vhodný voľný úsek pamäte v mape, tak príslušné bity nastavím na 1.

Pridelenie pamäte v zozname

ak nájdem vhodný voľný úsek pamäte v zozname tak:

- zmením jeho stav na pridelený ak je veľkosť zhodná.
- zmením jeho stav a zmenším ho na požadovanú veľkosť. Zo zbytku spravím nový voľný úsek.

Manažment voľnej pamäte - algoritmy pridelenia

First Fit

Prehľadávanie zoznamu/mapy od začiatku kým nenájdem prvý vyhovujúci úsek. Nájdený úsek môže byť väčší ako treba.

Next Fit

Prehľadávanie zoznamu/mapy od posledného miesta pridelenia kým nenájdem prvý vyhovujúci úsek. Nájdený úsek môže byť väčší ako treba.

Best Fit

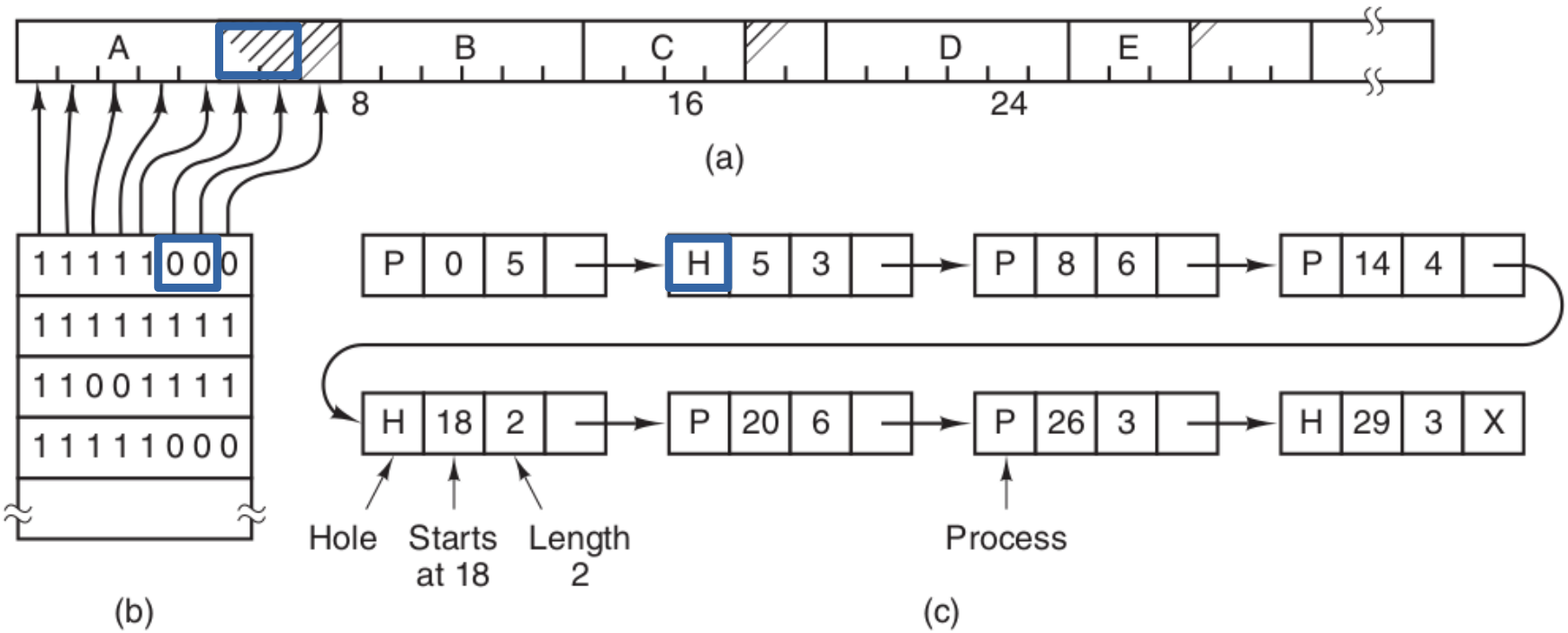
Prehľadávanie celého zoznamu/mapy kým nenájdem najmenší vyhovujúci voľný úsek. Vznikajú extrémne malé nepoužiteľné úseky.

Worst Fit

Prehľadávanie celého zoznamu/mapy kým nenájdem najväčší vyhovujúci voľný úsek.

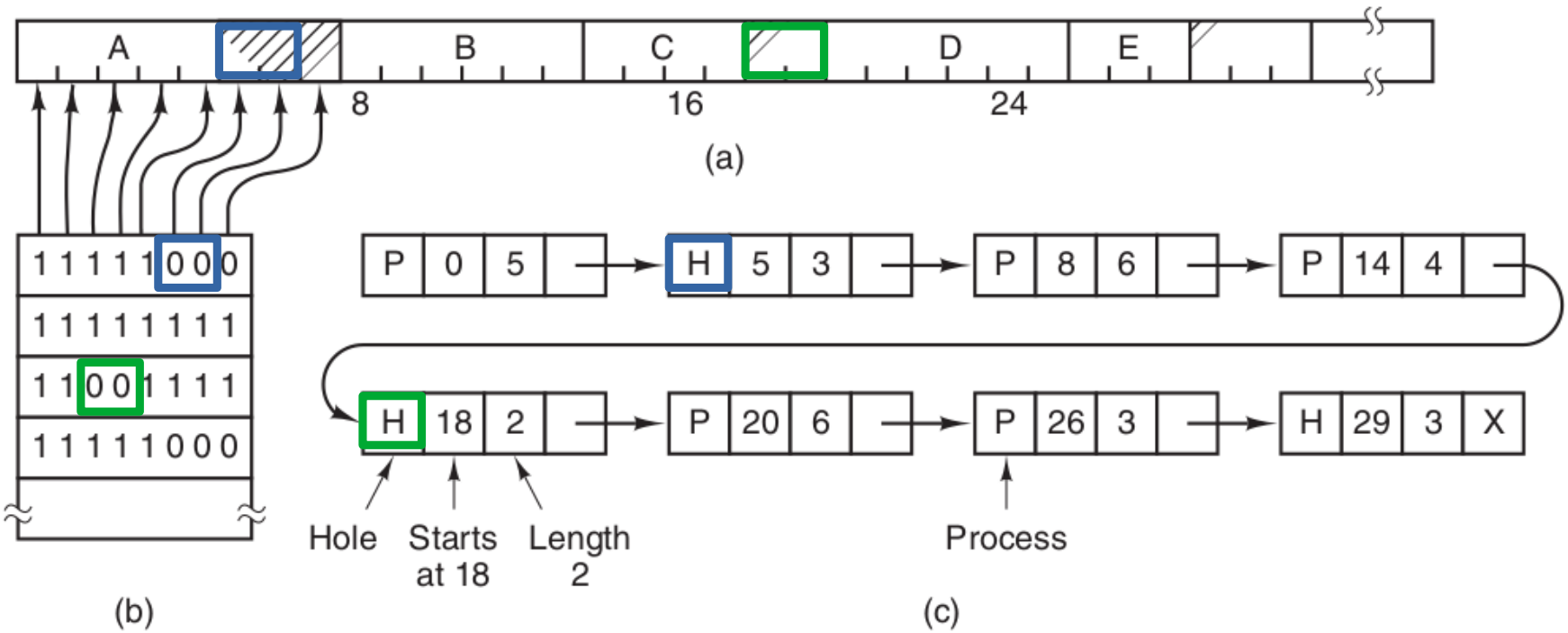
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 First Fit



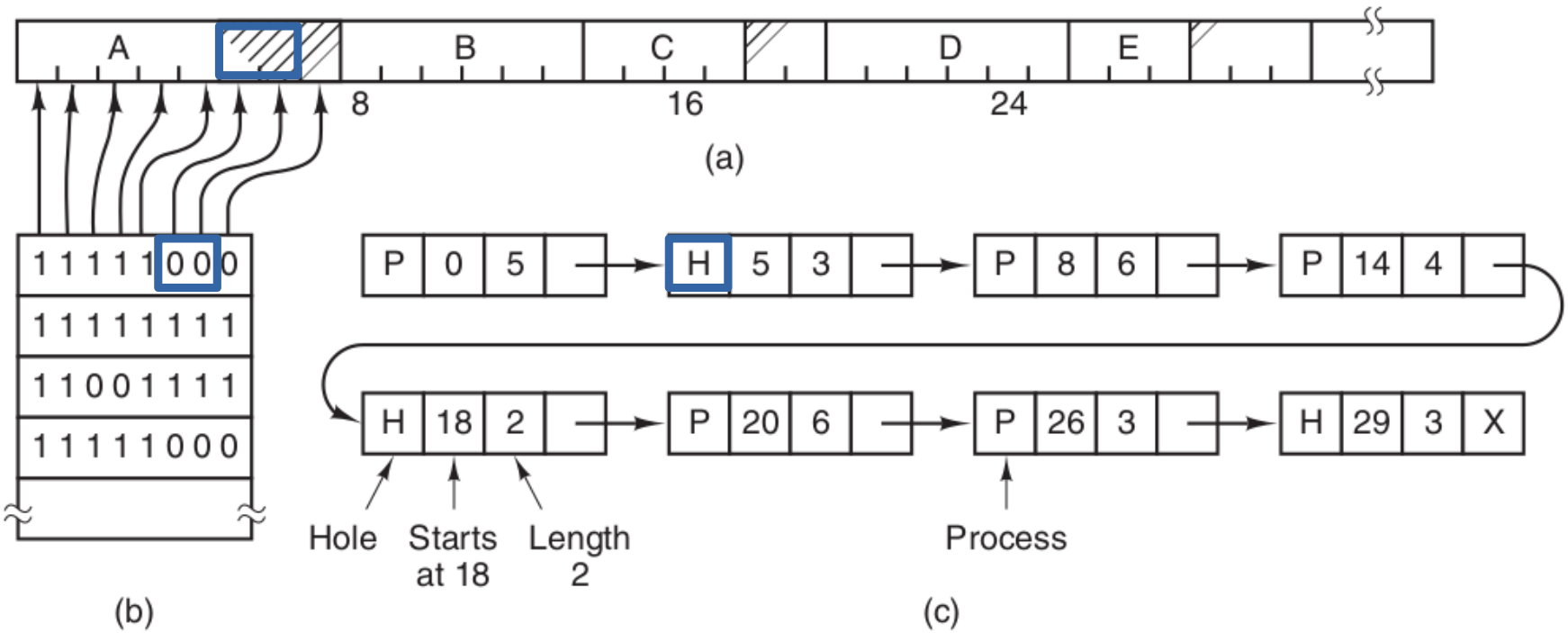
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Next Fit



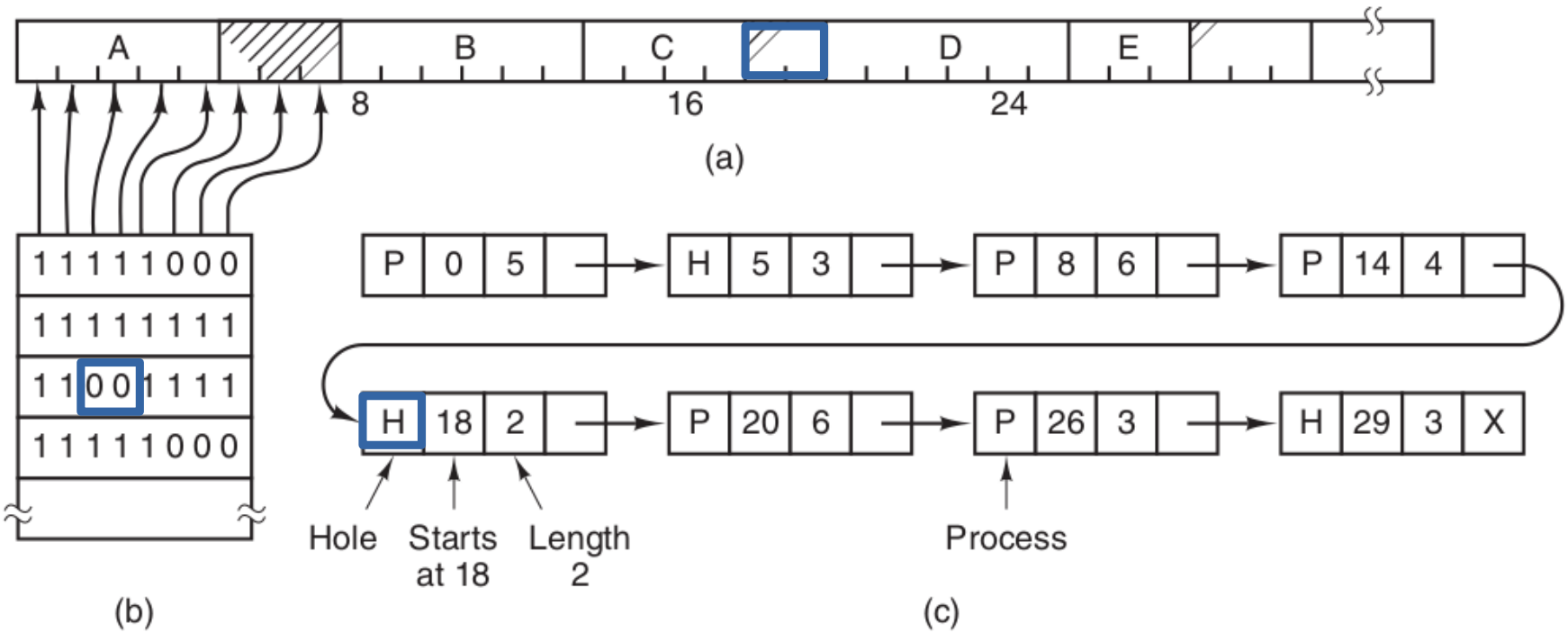
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Best Fit



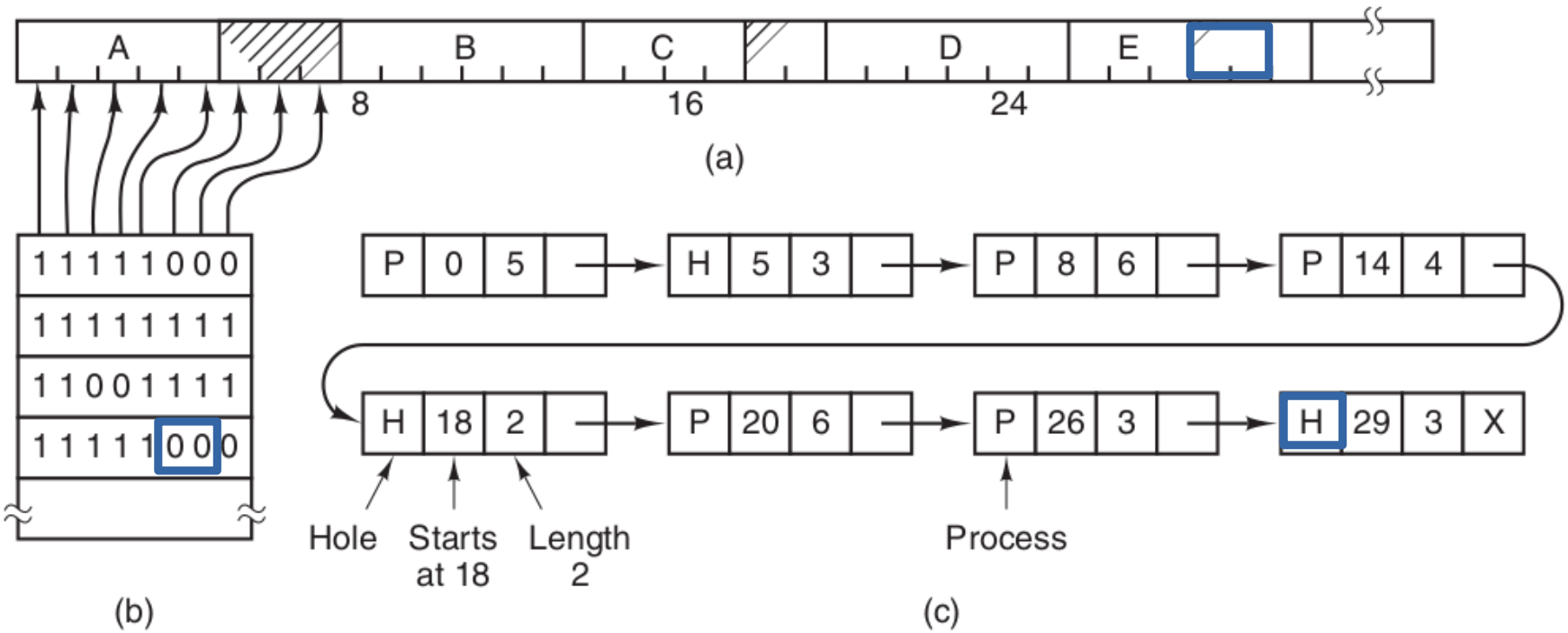
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Best Fit



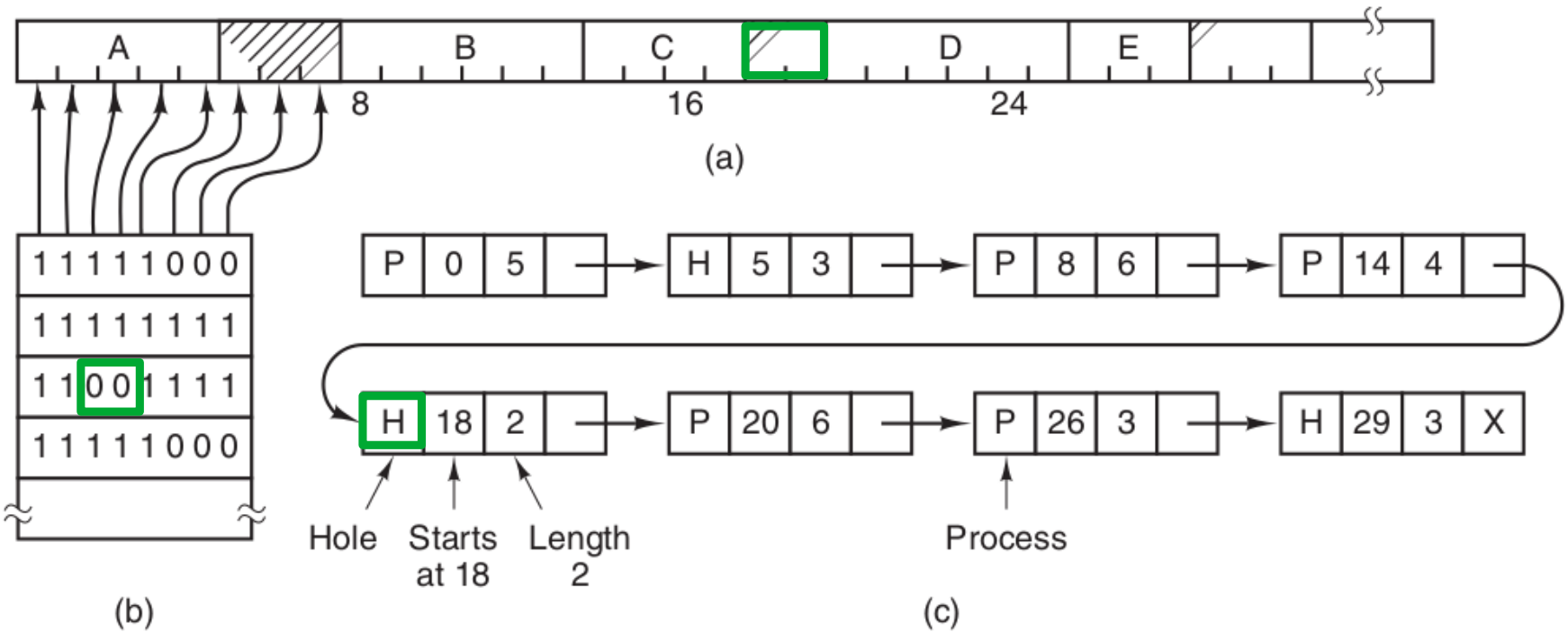
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Best Fit



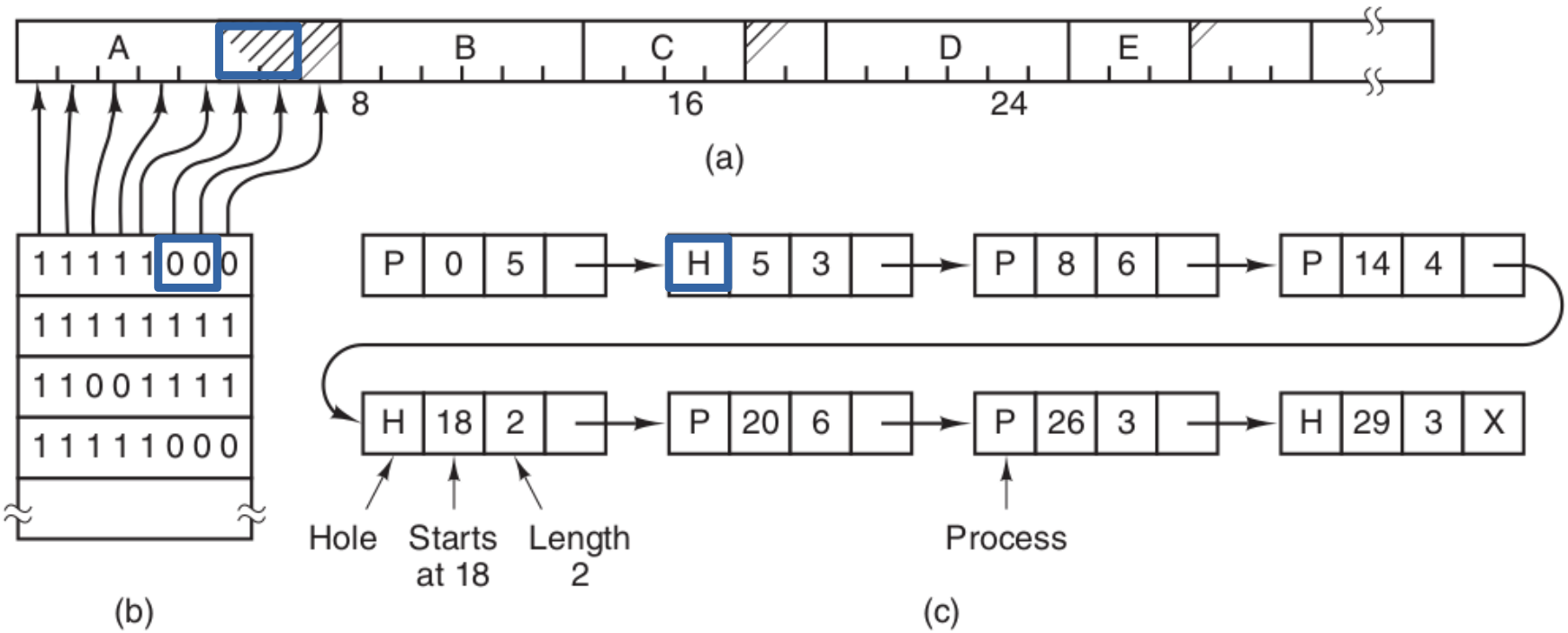
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Best Fit



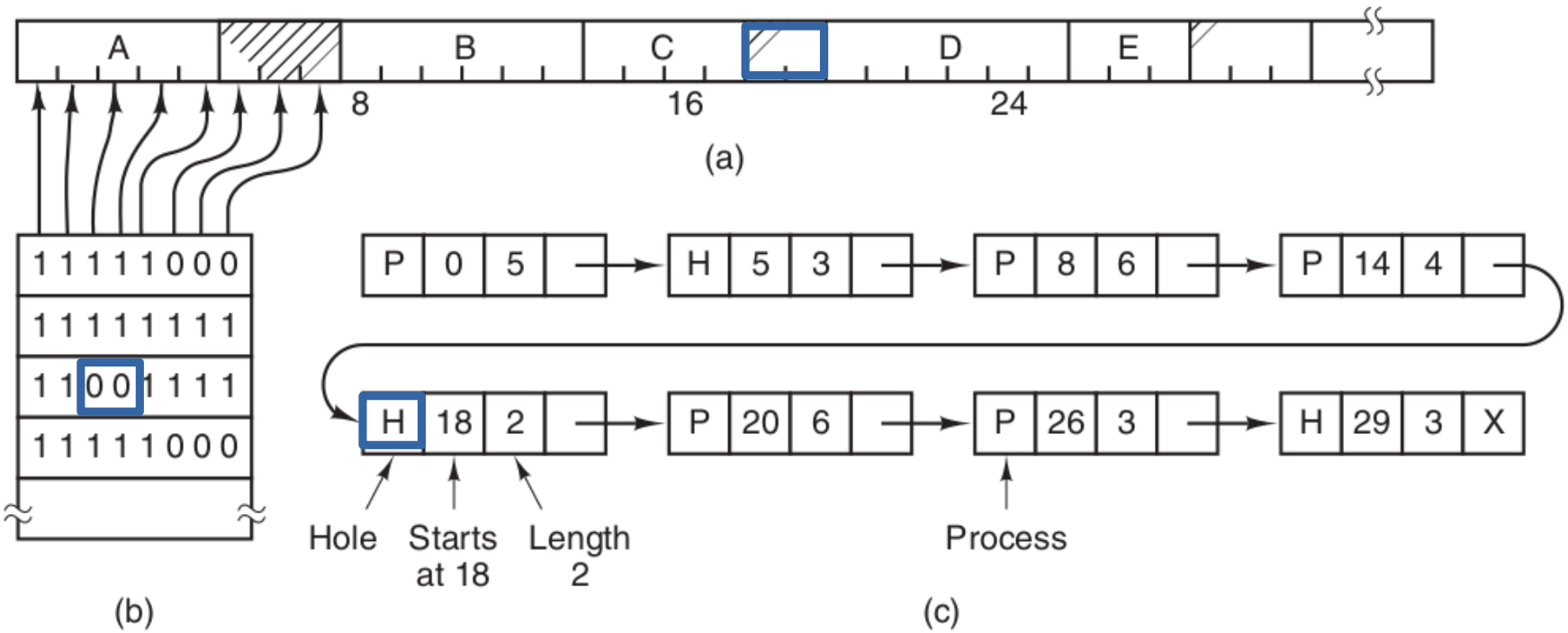
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Worst Fit



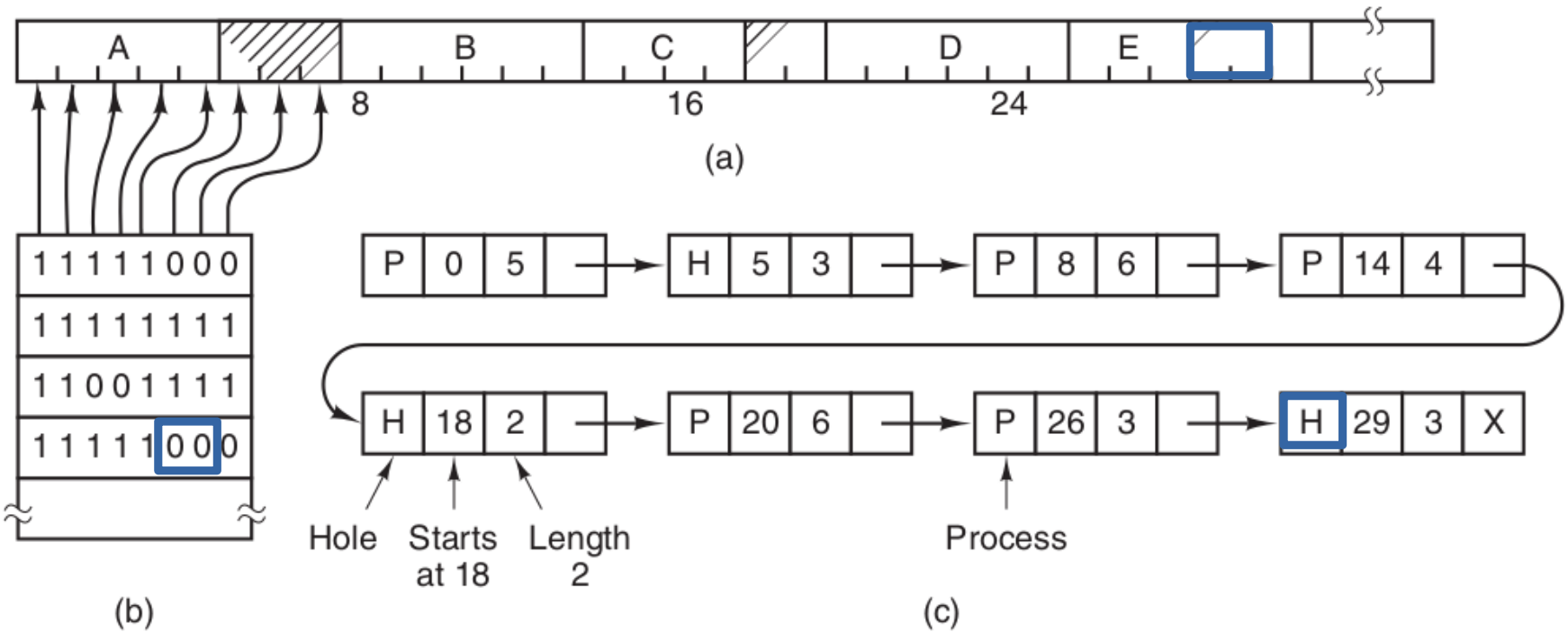
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Worst Fit



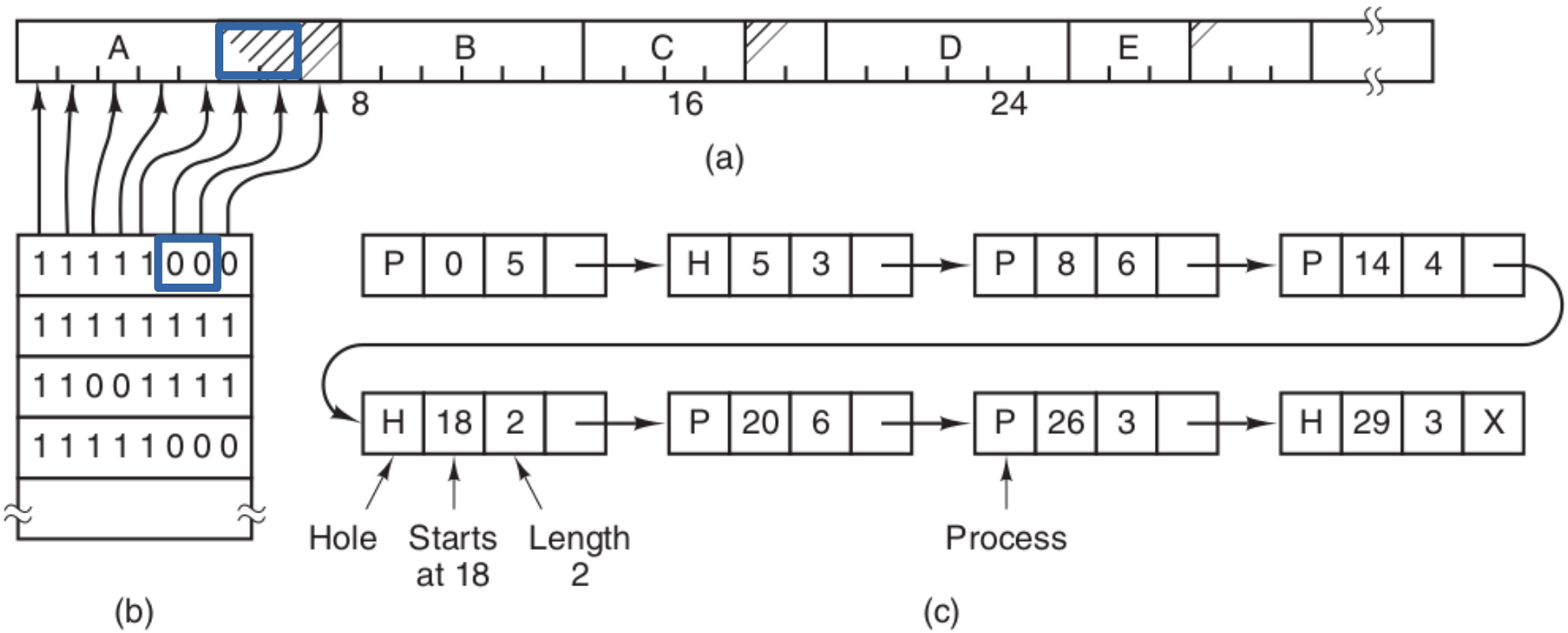
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Worst Fit



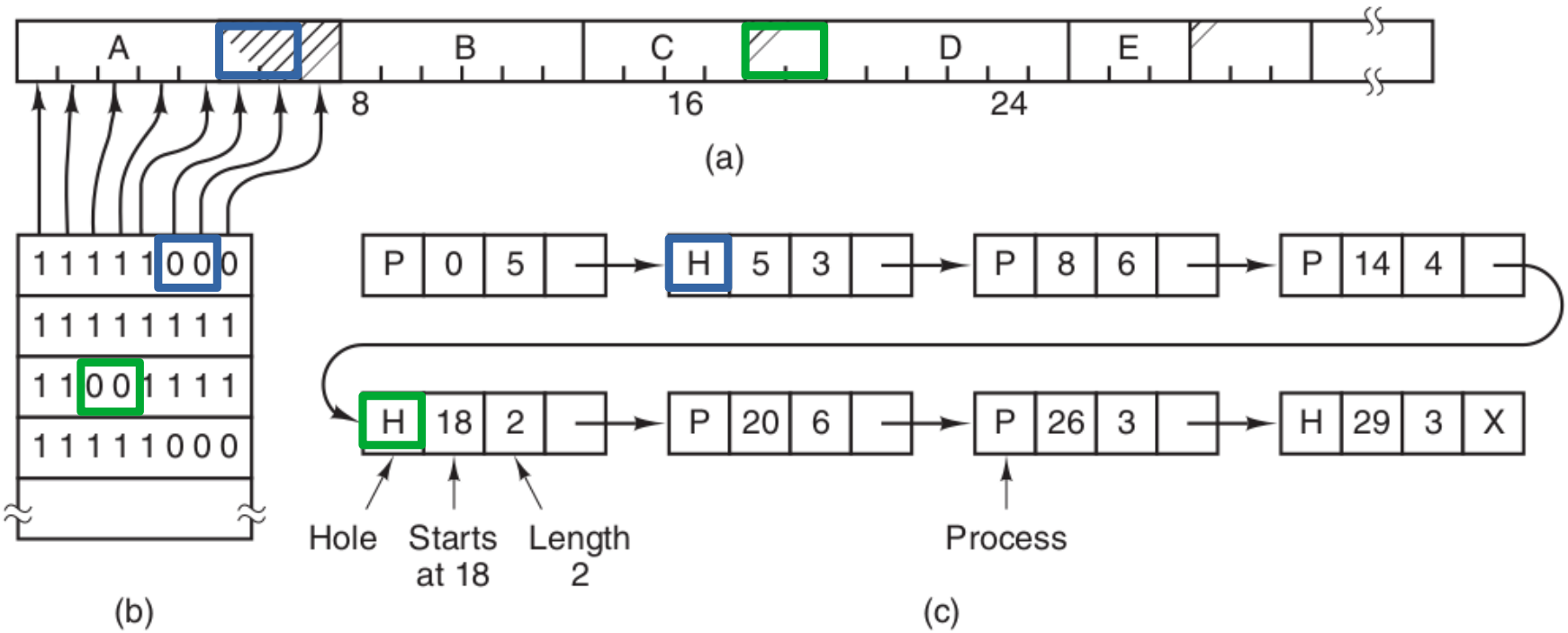
Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 Worst Fit



Manažment voľnej pamäte - algoritmy pridelenia

Allocate: 2 First Fit, Worst Fit, Next Fit, Best Fit



Manažment voľnej pamäte - optimalizácia

- Rozdelenie spájaného zoznamu na zoznam pridelených úsekov a voľných úsekov.
- Zoznam voľných úsekov môžeme:
  - udržiavať usporiadaný podľa veľkosti. (Best Fit a Worst Fit)
  - rozdeliť na viac zoznamov s položkami rovnakej veľkosti. (Quick Fit)
- usporiadanie je nákladnejšie.(Potrebujem uložiť viac informácií)

Quick Fit

ak udržujeme viac zoznamov (tried) voľných úsekov. Požadovaná pamäť sa hľadá v konkrétnej triede voľných úsekov alebo väčšej.

Zhrnutie

Zhrnutie

- Správa pamäte prebehla komplikovaným vývojom.
- Spočiatku OS nemali žiadnu abstrakciu pamäte. V pamäti bol jeden proces.
- Neskôr sa vyvinula jednoduchšia abstrakcia pamäte za pomoci Base a Limit registra:
  - V pamäti existuje viacero procesov.
  - Spočiatku máme rovnaké bloky pamäte. (interná fragmentácia)
  - Neskôr prišli bloky rôznych dĺžok, ktoré sa ale za života procesu nemenili. (externá fragmentácia)
- Požiadavky na pamäť rástli a pristúpilo sa k swap-ovaniu.
- Príbeh nekončí. Tešíme sa na segmentovanie a stránkovanie

Čo robiť do ďalšej prednášky

- Prečítať kapitolu 3.4, 3.5, 3.6 a 3.9 z Tanenbauma.