Pokročilé operační systémy

Alokace paměti

Petr Krajča

Katedra informatiky
Univerzita Palackého v Olomouci

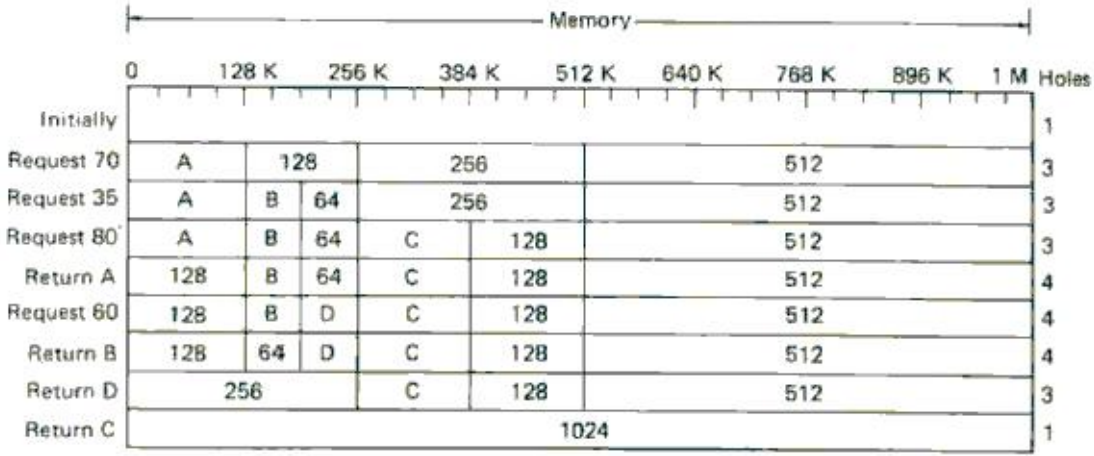
Alokace paměti na úrovni jádra

1. alokace větších souvislých bloků paměti (zarovnané na stránky)
2. alokace menších bloků (řádově desítky bytů)

- oddělené, ale provázané problémy
- buddy alokátor (pro větší úseky)
- SLAB alokátor (pro menší bloky)

Buddy alokace paměti

- kompromisní řešení z pohledu vnější a vnitřní fragmentace
- jednoduché na implementaci
- přiděluje pamět po blocích velikost 2^K, kde $L \leq K \leq U$, kde 2^L je nejmenší blok, který chceme přiřadit a 2^U je největší možný blok (celá paměť)
- algoritmus alokace paměti velikosti s
  1. najdi blok velikosti $2^{k-1} < s \leq 2^k$, příp. pokud je $s < 2^L$, blok velikosti 2^L
  2. pokud takový není, rozděl nejmenší blok větší než 2^k na půl
  3. přejdi na krok jedna
- uvolnění paměti
  1. uvolni blok paměti
  2. je jeho soused volný?
  3. pokud ano, sluč je dohromady, pokračuj krokem 2
- snižuje míru vnější fragmentace, má střední míru vnitřní fragmentace
- spojení volný bloků je rychlé


Memory
0
128 K
256 K
384 K
512 K
640 K
768 K
896 K
1 M
Holes
Initially
1
Request 70
A
128
256
512
3
Request 35
A
B
64
256
512
3
Request 80
A
B
64
C
128
512
3
Return A
128
B
64
C
128
512
4
Request 60
128
B
D
C
128
512
4
Return B
128
64
D
C
128
512
4
Return D
256
C
128
512
3
Return C
1024
1

Implementační poznámky

- nutné evidovat velikosti jednotlivých bloků
- pro velikost objektů se používá řád (všechny alokované úseky mají velikost 2^k, stačí znát k)
- snadné nalézt souseda, stačí otočit nejnižší bit pro adresy daného řádu
- pro každý řád se eviduje seznam volných bloků (lze použít volné bloky)
- výhodné mít oboustranný spojový seznam
- ostatní informace vhodné mít v pomocných strukturách (pro každou stránku/rámec jednu)

Slab alokátor

- vychází z předpokladu, že objekty je potřeba inicializovat
  - nastavit tabulku metod,
  - nastavit výchozí hodnoty,
  - inicializovat synchronizační primitiva, atd.
- nezanedbatelná režie
- přichystání objektů do cache a jejich recyklace po uvolnění
- pro každý typ objektů samostatná cache (de facto alokátor)
- stránka (slab) rozdělena na objekty stejné velikosti + hlavička
- každá cache má tři seznamy slabů (oboustranný spojový seznam)
  - plné
  - částečně zaplněné
  - prázdné
- slaby získány jako jednotlivé stránky např. buddy alokací
- možnost vrátit paměť (volných slabů)
- konstruktory a destruktory objektů volány při alokaci/uvolnění celého slabu

Slab alokátor, dokončení

- problém s vnější fragmentací (řeší různě velké slaby)
- lepší využití cache (lze řešit umístěním volného místa)

Alokace objektů bez potřeby inicializace

- `kmalloc(size)`, `kfree(obj)`
- lze použít Slab alokátor bez konstruktoru/destruktoru
- nachystané pro velikosti 8, 16, 32, ...

Úkol

- Naimplementujte funkce `kmalloc` a `kfree`, které budou schopny pracovat s objekty o velikosti 8 B až 512 B.

Bonusový úkol

- Naprogramujte alokátor paměti pro standardní knihovnu C.
- Použijte stejnou strategii jako ptmalloc.
- Použijte 32 front pro objekty o velikost 8 až 256 B na 32bitové architektuře, 16 až 512 B na 64bitové architektuře.
- Ostatní objekty pro jednoduchost umístěte do jedné fronty, ve které se bude vyhledávat formou first-fit.