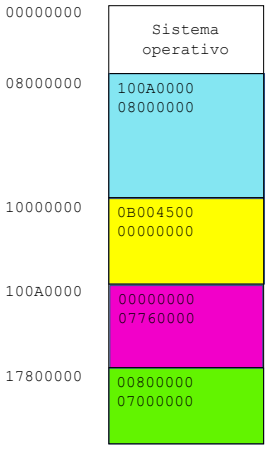
Programmazione di Sistema (JZ-ZZ) 2022/23

Memory Management

Esercizi proposti tratti da temi di esame

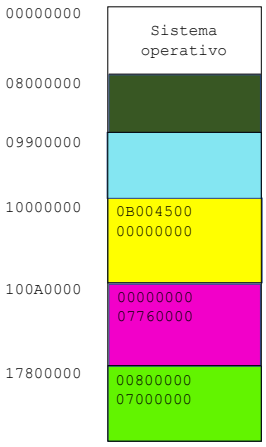
Soluzione 1 *(da 05/07/2011)*



Nell'istante iniziale, la configurazione è la seguente:
Free List: (08000000, 08000000), (100A0000, 07760000)
P10: (10000000, 000A0000)
P11: (17800000, 08800000)

- Allocazione di P12:

Servono 25 MB (in esadecimale 01900000 B), che vengono allocati nella partizione libera più grande (strategia worst-fit), ovvero la partizione (08000000, 08000000). La partizione libera risultante dall'allocazione di P12 avrà SIZE ($SIZE_{\text{partizione libera}} - SIZE_{P12}$) 08000000 – 01900000 = 06700000 B e indirizzo di partenza pari a 10000000 – 06700000 = 09900000.

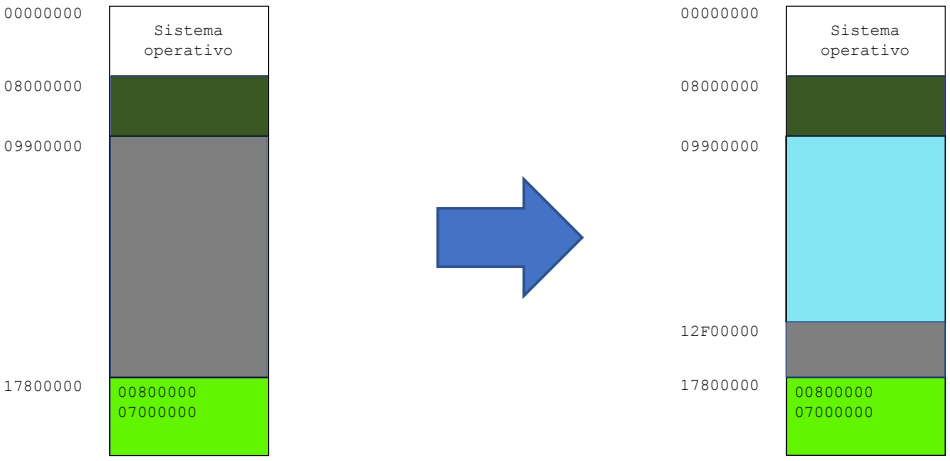


La nuova configurazione è la seguente:
Free List: (100A0000, 07760000), (09900000, 06700000)
P10: (10000000, 000A0000)
P11: (17800000, 08800000)
P12: (08000000, 01900000)

P.S.: Notare che, essendo la Free List ordinata in ordine decrescente da testo dell'esercizio, la partizione libera residua dopo l'allocazione di P12 va inserita in coda.

- Allocazione di P13:

Servono 150 MB (in esadecimale 09600000 B). Nessuna delle partizioni è grande abbastanza.
Optiamo quindi per uno swap-out di P10, generando così un'unica partizione di dimensione SIZE + SIZE + SIZE (06700000 + 000A0000 + 07760000) = 0DF00000 B. Qui sarà allocato il processo P13. La partizione libera risultante dalla allocazione di P13 avrà SIZE pari a 0DF00000 – 09600000 = 04900000
La configurazione risultante è la seguente:
Free List: (12F00000, 04900000)
P11: (17800000, 08800000)
P12: (08000000, 01900000)
P13: (09900000, 09600000)



Dopo lo swap-out di P10 Dopo l'allocazione di P13

- **Termine di P11:**

Il processo P11 viene terminato e la sua partizione liberata, generando così un unico blocco nella Free List di dimensione = 04900000 + 08800000 = 0D100000

La nuova configurazione è:

Free List: (12F00000, 0D100000)

P12: (08000000, 01900000)

P13: (09900000, 09600000)

Soluzione 2 *(da 01/09/2008)*

- Con *hit-ratio* si intende la percentuale di volte in cui la page richiesta viene trovata nel TLB
- Ipotizzando che ogni entry nelle tabelle delle pagine utilizzi 32 bit (4 byte), una page table di secondo livello avrà 2^{11} caselle (11 da p2) e dimensione $2^{11}*4$ Byte, mentre il primo livello avrà 2^{10} caselle e dimensione $2^{10}*4$ Byte.
- La frammentazione esterna è nulla per definizione a causa della page table. La frammentazione interna per P1 è anche essa nulla, essendo lo spazio virtuale multiplo di 2^{11} = 2 KB (la dimensione delle pagine/frame, d = 11)
- EAT = 0.99 * 300 + 0.01 * 3 * 300 = 306 ns. Si noti che nel caso di TLB Miss, gli accessi sono 3 (table page a due livelli e RAM)

Soluzione 3 *(da 25/06/2018)*

Vantaggi:

- Risparmio di memoria: la tabella ha le dimensione della RAM fisica invece che quella dello spazio di indirizzamento virtuale.
- C'è un'unica tabella per tutti i processi

Svantaggi:

- Lentezza: occorre cercare una pagina anziché accedervi in modo diretto.

Si trascurano eventuali bit di validità e modifica.

Standard Page Table:

N_{pagine} = 32 GB / 1 KB = 32 M (numero di indici di frame nella page table)

Dimensione Page Table = 32 M * 4B = 128 MB

Inverted Page Table:
Ogni riga dell'IPT contiene un indice di pagina (4B) e il pid del processo (2B)
Nframe = 8GB / 1 KB = 8M (per IPT si usa l'intera memoria RAM)
Dimensione IPT = 8M * (4B + 2B) = 48 MB

Il numero massimo di pagine virtuali di un processo è limitato dalla dimensione degli indici di pagina (32 bit), cioè 2^{32} = 4 GB. Siccome ogni pagina ha dimensione 1 KB, lo spazio di indirizzamento virtuale ha dimensione massima 4 GB * 1 KB = 4 TB