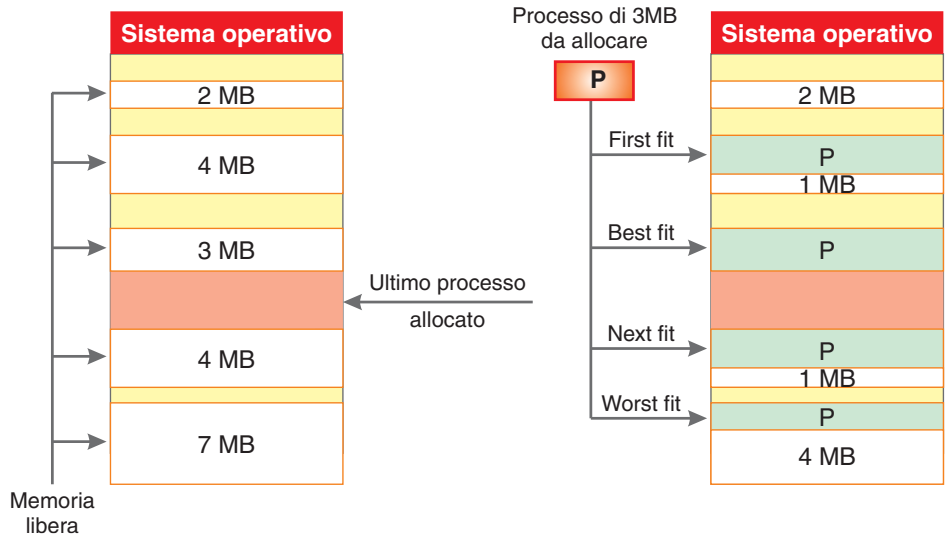
Algoritmi di allocazione dei processi nei sistemi a partizioni variabili

Con la gestione della memoria a partizioni dinamiche bisogna risolvere il problema dell'**allocazione dinamica della memoria** che consiste nello scegliere lo spazio di memoria libera dove collocare un processo. I diversi algoritmi di allocazione devono cercare di limitare la frammentazione della memoria.
Si consideri la situazione di una memoria parzialmente occupata dai processi, indicati con rettangoli colorati, e con diversi spazi liberi dei quali è specificata la dimensione. L'ultimo processo collocato in memoria è evidenziato in un colore differente. Supponiamo di dover creare una nuova partizione di 3MB per collocare in memoria il processo **P**.



La scelta della collocazione di **P** in memoria può essere fatta con diversi criteri che provocano l'inserimento di **P** in zone differenti della memoria.

- L'algoritmo **First Fit** (*il primo dove ci sta*) prevede di scandire la memoria dall'inizio sino a trovare uno spazio libero di dimensione maggiore o uguale alla dimensione del processo.
- L'algoritmo **Best Fit** (*la taglia migliore*) richiede di scandire tutta la memoria per identificare lo spazio libero più piccolo in grado di contenere P. L'idea alla base del *best fit* è di ridurre lo spazio di memoria sprecata, creando frammenti di dimensione minima.
- Il **Next Fit** (*il prossimo dove ci sta*) è una variante del *first fit*. Con il *next fit* la ricerca del primo spazio libero di memoria in grado di accogliere **P** viene fatta partendo dallo spazio libero successivo all'ultimo processo allocato. L'algoritmo *next fit* compensa la tendenza del *first fit* di allocare i processi all'inizio della memoria, cercando di distribuirli in tutta la memoria.
- L'algoritmo **Worst Fit** (*la taglia peggiore*) colloca il processo nello spazio di memoria libera di dimensione maggiore. È l'opposto del *best fit* ed è basato sul presupposto che se lo spazio libero di memoria è grande, lo spazio che rimane libero dopo avere allocato un processo è abbastanza grande per contenere altri processi e per non essere sprecato.

Per determinare l'efficienza dei diversi metodi sono stati condotti molti studi e simulazioni sul comportamento di questi algoritmi. Le conclusioni sono che il *first fit* è di norma ritenuto il più efficiente, insieme alla variante *next fit*, seguito dal *best fit* e dal *worst fit*. Le prestazioni decrescono sia per il tempo necessario ad eseguire gli algoritmi, sia per l'efficienza nell'utilizzazione della memoria.

Domanda

Abbina a ciascun algoritmo di allocazione dinamica della colonna di sinistra il corrispondente criterio scegliendo nella colonna a destra:

a) First Fit — 1) Dall'ultimo spazio libero allocato con gli spazi liberi ordinati secondo la collocazione in memoria

b) Best Fit — 2) Dall'inizio della memoria con gli spazi liberi ordinati secondo la collocazione in memoria

c) Next Fit — 3) Con gli spazi liberi ordinati per valori decrescenti di ampiezza

d) Worst Fit — 4) Con gli spazi liberi ordinati per valori crescenti di ampiezza.

(Risposta: a2, b4, c1, d3)

Problemi

1. Usare gli algoritmi di allocazione first fit, best fit e worst fit per posizionare un processo di 800 KB in una memoria avente spazi liberi di 2MB, 3MB, 50MB, 13MB, 1MB. Gli spazi liberi sono elencati in base al posizionamento in memoria.
2. Scrivere un metodo per realizzare gli algoritmi di allocazione first fit, next fit, best fit, worst fit, ipotizzando una tabella con gli spazi liberi ordinati in base al posizionamento in memoria.