

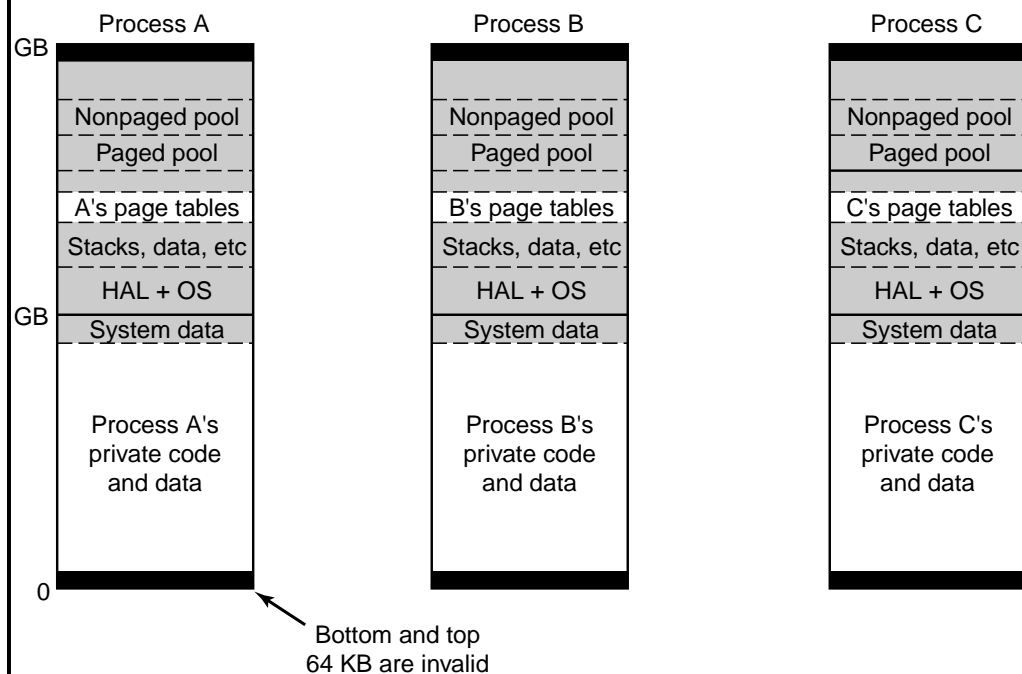
Windows: Gestione della Memoria

1

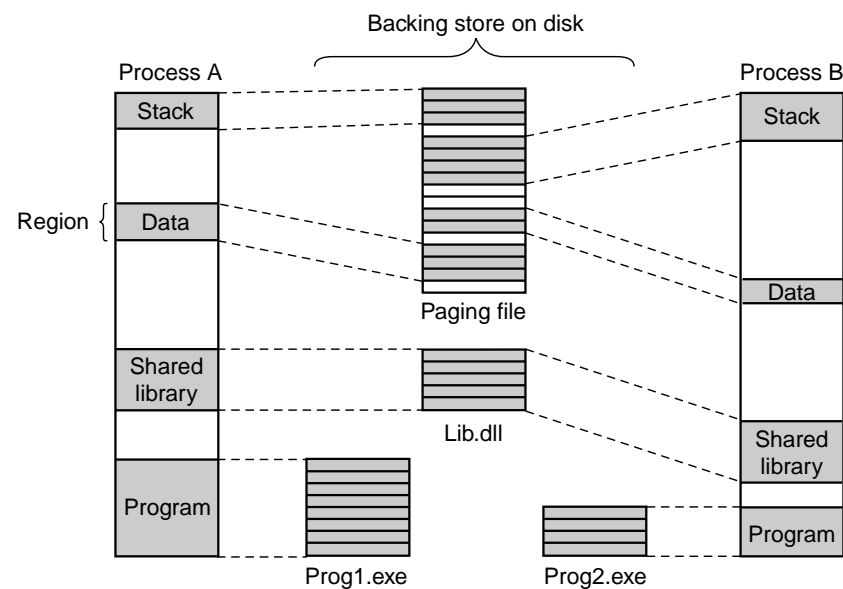
Modello della memoria in Windows 2000

- Ogni processo utente di Windows riceve uno spazio indirizzi virtuali di 4G, diviso in due parti
 - codice e dati, nella parte bassa (< 2G) Liberamente accessibile.
 - kernel, HAL (strato di astrazione hardware) e strutture di sistema (comprese le page tables) nella parte alta La maggior parte di questo spazio non è accessibile, neanche in lettura.
 - i primi ed ultimi 64kb sono invalidi per individuare rapidamente errori di programmazione (puntatori a 0 e -1).
- La memoria è puramente paginata (senza prepaging), con copy-on-write.
- Il caricamento dei segmenti è basato fortemente sul memory mapping.
- Mappare il kernel nello spazio dei processi utenti aumenta l'efficienza delle chiamate di sistema: un thread che passa in modo kernel non deve cambiare tabelle per gestire la rilocazione

2



Mappatura dei segmenti in memoria reale e su file



3

Gestione della paginazione

Nessuna forma di prepaging: tutte le pagine vengono caricate su page fault.

Le pagine possono essere in diversi *stati* dai quali dipendono le sorti di un page fault

Windows utilizza la strategia del working set per gestire la paginazione

4

Stati di una pagina

- *Available*: pagina non usata da nessun processo. Si divide in tre possibilità:
 - Free: riusabile
 - Standby: rimossa da un ws ma richiamabile (buffering)
 - Zeroed: riusabile e in più tutta azzerata (si cancellano i dati per ragioni di sicurezza)
- *Reserved*: riservata da un processo ma non ancora usata (ad es. spazio per lo stack di un thread). Non fa parte del ws fino a che non viene veramente usata.
- *Committed*: usata da un processo e associata ad un blocco su disco (*mappata*): la pagina fisica potrebbe non essere in memoria

5

Casi di page fault

- La pagina riferita non è committed
 - ⇒ Terminazione del processo
- Violazione di protezione
 - ⇒ Terminazione del processo
- Scrittura su una pagina condivisa
 - ⇒ Copy-on-write su una pagina reserved
- Crescita dello stack
 - ⇒ Allocazione di una pagina azzerata
- La pagina riferita è committed (ha un indirizzo fisico associato) ma non attualmente caricata in memoria
 - ⇒ il vero page fault: pagein della pagina mancante

6

Algoritmo di rimpiazzamento di pagina

La paginazione è basata sul modello del Working Set per i *processi*

- ogni processo ha un working set (insieme delle pagine mappate in memoria) e una dim. minima *min* e massima *max* (si può scendere sotto *min* e salire sopra *max*)
- Tutti i processi iniziano con lo stesso *min* e *max* (risp. 20-50 e 45-345, in proporzione alla RAM; modificabile dall'amministratore del sistema)
- Ad un page fault ($ws = \text{dimensione del working set}$)
 - se $ws < max$, la pagina viene allocata ed aggiunta al working set.
 - se $ws > max$, una pagina vittima viene scelta nel working set (politica di rimpiazzamento *locale*)
- I limiti possono cambiare nel tempo: se un processo sta paginando troppo (thrashing locale), il suo *max* viene aumentato
- vengono mantenute sempre libere almeno 512 pagine

7

Le pagine allocate vengono prelevate dalla *free list*:

- ogni secondo parte un thread del kernel (*balance set manager*)
- se la *free list* è troppo corta, parte il *working set manager* che esamina i *working sets* per liberare pagine
 - prima i processi più grandi e idle da più tempo; il processo in foreground è considerato per ultimo
 - se un processo ha $ws < min$ o ha avuto molti page fault recentemente, viene saltato
 - altrimenti una o più pagine vengono rimosse
- si ripete sempre più aggressivamente finché la *free list* ritorna accettabile
- anche parte del kernel può essere paginata
- Eventualmente un *ws* può scendere sotto il *min*
- non esiste completo swapout di processi

4. Azzerate (zeroed): libere e con contenuto azzerato

- le pagine nelle liste 1 e 2 si possono recuperare quando il processo corrispondente le richiede
- la lista 3 contiene pagine di processi che hanno terminato l'esecuzione
- Le pagine possono cambiare lista:
 - dei demoni di scrittura di pagine mappate/modificate spostano pagine da 1 a 2
 - come effetto di una deallocazione una pagina può andare da 2 a 3
 - il demone *zero page thread* (gira a priorità più bassa) sposta pagine da 3 a 4: se la CPU è inattiva vengono azzerate delle pagine (più utili di pagine sporche)

Gestione della memoria fisica

- I frame liberi sono organizzati in diverse liste mantenute dal sistema operativo
- Windows 2000 utilizza strategie ed euristiche complesse (che non vedremo) per gestire la scelta della pagina fisica da allocare per una richiesta
- Le liste sono 4:
 1. Pagine modificate: pagine eliminate da un *ws*, associate ad un processo, ma che devono essere copiate su disco
 2. Pagine in attesa (standby): eliminate da un *ws*, assoc. ad un processo, consistenti con immagine su disco
 3. Pagine libere: non sono più associate ad alcun processo