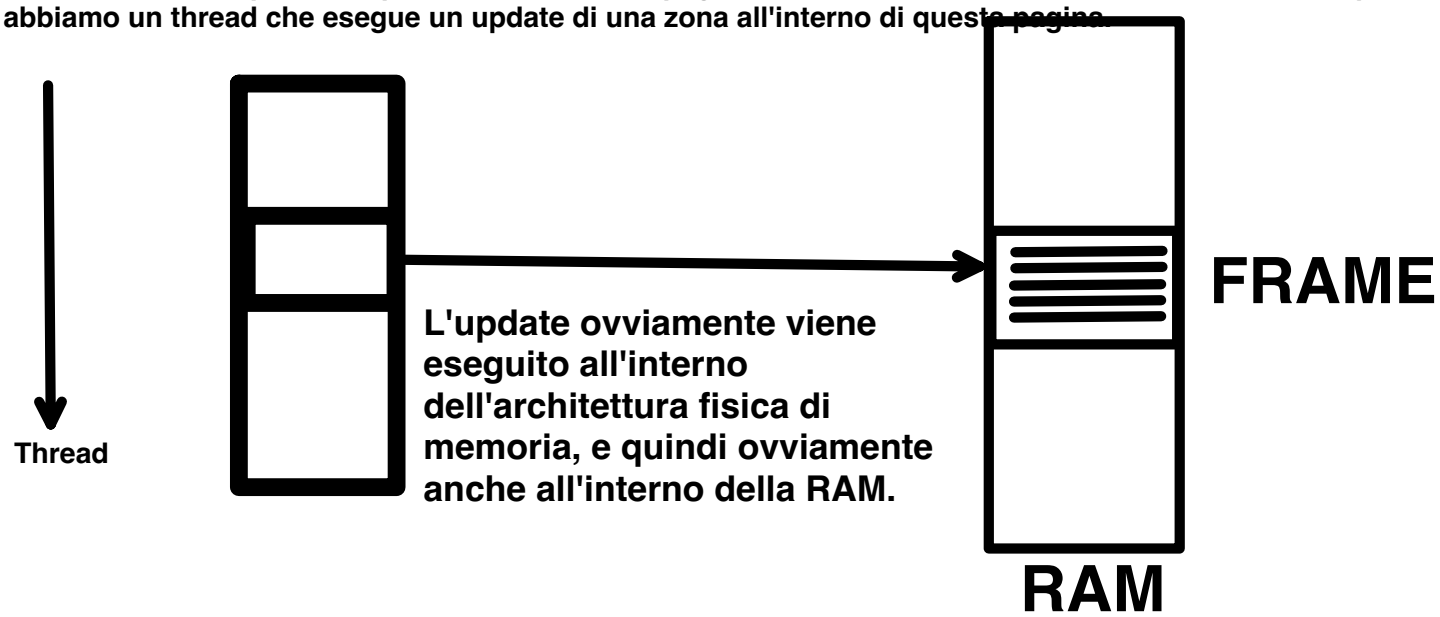
Tener traccia delle modifiche

giovedì 18 maggio 2023 11:28

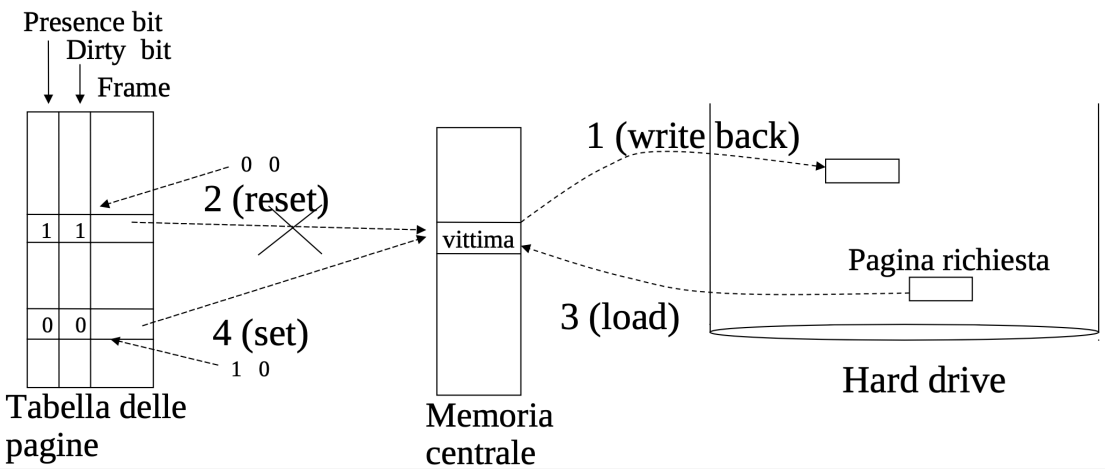
Tener traccia delle modifiche

Per i 3 aspetti di cui parlavamo prima, uno semplice è quello di "come" tenere traccia delle modifiche che le pagine subiscono dal software, quando queste sono in memoria di lavoro.
Dato un address space, noi possiamo avere una pagina che è caricata all'interno della RAM in uno specifico FRAME, e poi abbiamo un thread che esegue un update di una zona all'interno di queste pagine.



Come facciamo a tener conto, che l'informazione presente in RAM, è stata aggiornata da questo thread?
Questa cosa qui viene fatta utilizzando un'altra facilities che ci mettono a disposizione le Page Table, e in particolare, per ogni entry della page table associata ad una specifica pagina logica, non abbiamo solo il Presence Bit che ci dice se l'informazione che riguarda il posizionamento della pagina logica all'interno della RAM, effettivamente è valida, ma abbiamo anche il dirty bit, che implica dire che la prima volta che accediamo in scrittura ad una specifica pagina logica, e ovviamente questo accesso in scrittura prevede che il sottosistema di controllo vada comunque nella entry per pescare il frame e caricarlo nel TLB per andare a verificare se questa entry sia valida, quale sia il frame da accedere (in particolare identifichiamo un frame all'interno della memoria) ed eventualmente se stiamo accedendo in scrittura aggiorniamo anche il Dirty Bit, abbiamo la possibilità di mantenere traccia del fatto che una pagina è stata aggiornata ALMENO con una scrittura, magari più di una ma non fa niente, noi impostiamo il dirty bit a 1 per indicare che almeno una scrittura è avvenuta all'interno di questa pagina.

- la tabella delle pagine mantiene un bit per ogni entry denominato **dirty bit**
- il dirty bit viene posto ad 1 (dal firmware) su una scrittura sulla corrispondente pagina
- se il dirty bit associato alla vittima è pari a 1, la vittima viene ricopiata su hard-drive



È molto interessante quando dobbiamo prendere il contenuto del frame, e selezionarlo come VITTIMA. Perché se stiamo selezionando lui come vittima, per far spazio in quella zona della memoria centrale a qualche altra pagina, dobbiamo sicuramente avere il WriteBack che ci porta il contenuto del frame nella Swap Area - in modo tale da avere questa informazione aggiornata se la dovessimo ricaricare nella RAM - , sicuramente la entry che associava la pagina logica su cui volevamo lavorare a quel frame deve essere resettata per andare ad indicare che non è più valida, poi (nel momento in cui effettuiamo questa sostituzione) dobbiamo caricare una pagina RICHIESTA, poteva essere swapped out dal frame, e infine (punto 4) dobbiamo aggiornare questa entry associata alla nuova pagina logica che noi stiamo cercando di utilizzare ora, in modo tale che ci sia l'indicazione che la nuova pagina è finita esattamente all'interno di quel frame.
La stessa cosa poteva capitare se questa nuova pagina, invece di doverla ricaricare dall'hard drive, dovevamo semplicemente materializzarla perché acceduta per la prima volta dai thread di questa applicazione.
Nella nuova entry inizialmente sul presence bit abbiamo zero, perché originariamente questa entry della page table NON era valida.
Quando andiamo a caricare la pagina in memoria il software del S.O effettua un aggiornamento del presence bit da 0 a 1, che indica che quella pagina da ora in poi è presente ma chiaramente non è dirty se l'accesso per esempio era in lettura.
Il che implica dire che poi questi bit verranno aggiornati successivamente anche dal sotto-sistema di controllo nel momento in cui dovessimo eseguire una scrittura su questa stessa pagina che abbiamo caricato.
L'identificazione della vittima si basa comunque sulla strutturazione della Page Table per il mantenimento di ulteriori bit di gestione/controllo all'interno della page table stessa per identificare il fatto di aver scritto.

Per il secondo aspetto, l'idea di selezionare delle vittime nel momento in cui dobbiamo far spazio all'interno della RAM per altre informazioni che devono essere materializzate in RAM e qualcosa che è presente correntemente in RAM deve essere eliminato, è un aspetto che riguarda anche la definizione di algoritmi.

Algoritmi di selezione della vittima

All'interno dei sistemi operativi noi abbiamo realmente degli algoritmi che implementano la selezione della vittima. E quando noi abbiamo degli algoritmi dobbiamo anche essere in grado di valutarli, per andare ad indicare quale algoritmo possa essere migliore di un altro per eseguire queste attività di selezione della vittima.

Una delle metriche che classicamente è utilizzata per valutare questi algoritmi è la "Frequenza di Page Fault".

Se questi algoritmi sostituiscono le informazioni all'interno della RAM, facendo si che globalmente le mie applicazioni generino un numero ridotto di Major Fault, per riportare nuovamente in RAM altre cose, significa dire che la sostituzione è fatta bene.<
Quindi noi avevamo collocato e lasciato all'interno della RAM cose che effettivamente ci servivano.
Tipicamente o possiamo utilizzare delle tracce reali d'esecuzione di applicazioni, oppure possiamo utilizzare approcci casuali (quindi riferimenti all'interno di un address space generati in modo randomico.

Metodo di valutazione

- si utilizzano particolari sequenze di riferimenti ad indirizzi logici, generate in modo casuale oppure in base a tracce reali di esecuzione

C'è sicuramente il concetto di aspettativa che noi vorremmo avere nel momento in cui lavoriamo con algoritmi di selezione della vittima che sono buoni. In particolare se noi consideriamo un certo numero di FRAME, con un certo algoritmo e con un certo carico di lavoro, osserviamo una certa frequenza di page fault, l'aspettativa per un algoritmo buono è che se noi spendiamo soldi per acquistare altra RAM e quindi aumentiamo il numero dei frame, si ha una riduzione della frequenza dei page fault.

Algoritmi di selezione della vittima

Metrica di valutazione

- frequenza di page fault (major fault)

Metodo di valutazione

- si utilizzano particolari sequenze di riferimenti ad indirizzi logici, generate in modo casuale oppure in base a tracce reali di esecuzione

Aspettativa

- all'aumentare del numero di frames della memoria centrale, per una data sequenza di riferimenti ad indirizzi logici, la frequenza di page fault dovrebbe decrescere monotonamente