Relazione Progetto Buddy Allocator

Gianluca Scanu

Matricola 1884094

Il lavoro svolto presenta una possibilità di rappresentazione di un allocatore di memoria, di tipologia Buddy, che usa strutture differenti da quelle viste a lezione durante il corso. Nello specifico l'idea fondamentale è che l'allocatore non si costruisca su delle apposite strutture per la gestione di blocchi ma invece usufruisca di una struttura ausiliaria, una bitmap, per la realizzazione di questa funzione.

Per meglio comprendere la struttura del progetto risulta conveniente dividere la sua presentazione in macro aree, che corrispondono, in effetti, a dei moduli indipendenti la cui unione permette il funzionamento dell'intero allocatore.

La bitmap. La struttura sulla quale l'intero scheletro di questa rappresentazione di un buddy allocator prende forma è una bitmap. La bitmap è idealmente corrispondente ad un array di bit la cui lunghezza può essere decisa durante la sua inizializzazione. Attraverso le due funzioni per, rispettivamente, ispezionare un bit restituendone il valore e settare un valore di un bit, permette di mantenere una vista della memoria aggiornata. I singoli bit rappresenteranno infatti i blocchi della memoria partendo da quelli più grandi (da indice 0) per arrivare ai più piccoli.

L'albero. L'insieme dei blocchi, come abbiamo appena visto, viene gestito attraverso un array di bit. Per la gestione dei bit in questo array sfrutteremo le idee di un albero binario e le connessioni tra i nodi che lo compongono. La memoria viene infatti divisa ricorsivamente in due partendo dal blocco più grande, corrispondente alla memoria stessa, fino ad arrivare ai blocchi di grandezza minima. Ogni blocco così costruito corrisponderà ad un bit dell'array e, di conseguenza, ad un nodo dell'albero binario. Il primo blocco, il più grande, coinciderà concettualmente con la radice dell'albero e i blocchi di grandezza minima con le sue foglie ed ogni nodo risulterà collegato con il classico rapporto di parentela. È da notare come, dato che si sta parlando di un albero binario rappresentato tramite array, funzioni come quelle per raggiungere il nodo padre o i nodi figli risultano in semplici calcoli matematici sugli indici dell'array.

L'allocatore. Avendone spiegato gli strumenti, si può ora passare al vero soggetto di questo progetto: l'allocatore stesso. La sua inizializzazione prevede uno stretto controllo sui dati passati come parametro. Questi ultimi infatti potrebbero portare ad un'incongruenza nella ripartizione della memoria e ad una conseguente creazione di un allocatore non funzionante o parzialmente attivo ma incapace di operare

correttamente. Una volta terminati i controlli i suoi campi, tra cui la bitmap per la gestione dei blocchi, vengono inizializzati. L'allocatore fornisce poi l'implementazione delle funzioni centrali per il suo compito: **malloc** e **free**. La **malloc** interroga il livello dell'albero dove risiedono i blocchi abbastanza grandi per contenere la porzione da restituire. Nello specifico l'interrogazione, come per la free, avviene tramite uno scorrimento dei bit nella bitmap corrispondenti ai blocchi di cui si vuole sapere lo stato (0 = Libero , 1 = Occupato). Se un blocco di tale grandezza viene trovato allora viene restituito il suo indirizzo di partenza, altrimenti viene restituito NULL. Risulta importante notare come nella determinazione della disponibilità di un nodo, oltre ad uno scorrimento orizzontale, si faccia uso di una risalita dell'albero verticale al fine di ispezionare i genitori del nodo candidato. In questo modo sapremo se il blocco da restituire non è un discendente di blocchi più grandi occupati, ovvero se non è contenuto in essi. **La free** opera in modo analogo interrogando la bitmap ma, diversamente dalla malloc, lavorando sull'albero unicamente in direzione verticale. L'obiettivo della funzione risulta essere infatti quello di risalire l'albero partendo da un blocco, liberando (stato a 0) tutti i blocchi che possono essere liberati. La condizione per la liberazione durante la risalita della free risulta essere chiara: se il bit corrispondente al blocco e quello del suo buddy sono entrambi 0 il blocco genitore può essere liberato e il procedimento può continuare dall'indice del suo bit.

Testing. Il progetto risulta dotato di un programma di test che permette di apprezzare le funzionalità dell'allocatore in vari scenari. I tre possibili test (test1, test2, test3) offrono un'idea di come l'allocatore riesca a soddisfare in diverso ordine richieste malloc e free e a gestire richieste malloc troppo grandi o free con indirizzi invalidi. Nel test3 infine è possibile visualizzare lo stato della bitmap durante le varie operazioni di malloc e free osservando i cambiamenti di stato dei bit durante il processo.