Esempio

venerdì 7 aprile 2023 16:55

☆ ESEMPIO CONCRETO

1:07:00 29-03-2021

Abbiamo un'applicazione multi-thread per acquisire stringhe, il thread che acquisisce stringhe lancia un thread per far si che il thread che viene ad essere attivato va ad inserire la stringa in maniera ordinata all'interno di un'array di puntatori.

```c
#include <unistd.h>
#include <stdio.h>
#include <pthread.h>
#include <string.h>
#include <stdlib.h>

#define MAX_CHAR    1000
#define MAX_STRING      2000

int string_number=0;

char * string_array[MAX_STRING];

void *status;

void * OrderThread(void * new_string) {

  int i,j;

  //printf("Ordering thread active\n");
  for (i=0 ; i < string_number; i++) {
    if (strcmp(new_string,string_array[i]) <= 0) {
      for (j=0; j<(string_number-i); j++) {
    string_array[string_number - j] = string_array[string_number - j - 1];
      }
      break;
    }
  }
  string_array[i] = new_string;
  string_number++;
  pthread_exit((void *)NULL);
}

int main () {
  int i;
  pthread_t tid = 0;
  char * old_buffer;
  void * return_code;
  char buffer[MAX_CHAR];
  int notfirst = 0;

  while(1) {
    printf("Insert string: ");
    scanf("%s", buffer);

    if (strcmp(buffer, "quit") == 0 && notfirst == 0) return 0;
    if (strcmp(buffer, "quit") == 0) break;
    old_buffer = strdup(buffer);

    if (notfirst) pthread_join(tid, &status);
    else notfirst = 1;

    i=pthread_create(&tid, NULL, OrderThread, (void *)old_buffer);
    if (i) {
      printf("cannot create thread for error %d\n", i);
      exit(EXIT_FAILURE);
    }

  }

  pthread_join(tid, &return_code);

  for (i=0; i< string_number; i++) printf("String %d: %s\n", i,string_array[i]);

  return 0;
}
```

Il thread vive in una funzione che è fatta così: ritorna eventualmente un puntatore, dopodiché prende in input un puntatore e questo puntatore identifica la posizione di memoria che dobbiamo inserire all'interno del puntamento che facciamo tramite l'array di puntatori string_array, e per tutte le stringhe andiamo a comparare se la nuova stringa è minore o uguale alfabeticamente alla stringa puntata da quell'array di puntatori. Nel caso in cui la cosa sia vera spostiamo i puntatori per liberare un elemento e andiamo a scrivere il puntatore alla nuova stringa in quell'elemento.

Poi andiamo ad aggiornare il numero delle stringhe e ritorniamo il codice numerico 0 come puntatore status.

Nel main c'è un while(1) dove chiediamo una stringa, la leggiamo, e la mettiamo in buffer.

**Se la stringa è uguale alla stringa quit ed eventualmente non siamo passati all'interno di questo ciclo, usciamo direttamente.
Se viene messa la stringa qui ma eravamo già passati all'interno del while, ovviamente usciamo dal while con un break.**

**In ogni caso duplichiamo la stringa con strdup(buffer), eventualmente attendiamo che l'ultimo thread che è stato attivato abbia completato la sua esecuzione, questo lo facciamo con un join, in particolare passando TID, e poi ovviamente andiamo a creare un nuovo thread e l'id del nuovo thread lo registriamo in Tid, e questa tid nel giro dopo del while, la utilizziamo come parametro nella join appunto.
Il terzo parametro è la funzione Order_Thread, che deve riordinare questo database di stringhe che stiamo mantenendo, e infine nell'ultimo parametro gli passiamo ovviamente il puntatore in type-casting a (void)* della stringa che avevamo duplicato.**

Dopo questa duplicazione, buffer è di nuovo riutilizzabile in questa scanf() all'interno del ciclo while, per poter acquisire questa nuova stringa.

Abbiamo anche un check sugli errori di I. E ora passiamo nella parte esterna al ciclo.

Nella parte esterna al ciclo while(1), ATTENDIAMO l'ultimo thread lanciato e poi andiamo ad emettere in output tutte le stringhe che sono state inserite come puntate all'interno di quell'array di puntatori.

Per compilare un'applicazione che utilizza queste API per la gestione dei thread all'interno dei sistemi UNIX, dobbiamo indicarlo a GCC: -lpthread

```
AND-THREADS/UNIX> gcc threads-sort-strings.c -lpthread
AND-THREADS/UNIX> ./a.out
Insert string: xxx
Insert string: fff
Insert string: ddd
Insert string: aaa
Insert string: quit
String 0: aaa
String 1: ddd
String 2: fff
String 3: xxx
```