Esercizi d'esempio per la prima provetta – Sistemi Operativi per la Robotica

Schedulazione Real Time

- Si abbiano i seguenti 5 task tutti arrivati all'istante 0, descritti con la coppia (durata, deadline):
  T1(2,4), T2(3,5), T3(1,7), T5(1,6)
  Valutare la loro schedulabilita' in tempo reale con schedulazione non preemptive e fare il corrispondente diagramma temporale.

- Si abbiano i seguenti 5 task arrivati in vari istante, descritti con la terna (arrivo, durata, deadline):
  T1(0,2,4), T2(1,3,5), T3(2,1,7), T5(3,1,8)
  Valutare la loro schedulabilita' in tempo reale con schedulazione preemptive e fare il corrispondente diagramma temporale.

- Modificare il problema precedente utilizzando una schedulazione non pre-emptive (Bratley)

- Si abbiano i seguenti 6 task tutti arrivati all'istante 0, descritti con la coppia (durata, deadline):
  T1(2,5), T2(3,6), T3(1,8), T5(1,8), T6(2,10)
  I task sono sottoposti a questi vincoli di precedenza: A è il primo, da cui dipendono B e C. D, E dipendono da B ed F dipende da C.
  Determinare la loro schedulazione in tempo reale non preemptive usando LDF e fare il corrispondente diagramma temporale.

- Si abbiano i seguenti 5 task periodici, descritti con la coppia (durata, periodo):
  T1(2,4), T2(3,5), T3(1,5), T5(1,4)
  Valutare la loro schedulabilita' in tempo reale con schedulazione Rate Monotonic e fare il corrispondente diagramma temporale.

- Definire la schedulazione Polling Server e dire in cosa si differenzia con il Deferrable Server

- Definire un task sporadico.

- Definire la condizione di ottimalità in uno schedulatore

- Definire il limite superiore minimo di uno schedulatore Rate Monotonico. Perchè un insieme di task periodici è schedulabile se il fattore di utilizzazione è minore del limite superiore minimo?

- Siano dati tre task periodici:

	periodo	durata
T1	2	1
T2	4	1
T3	8	2

  Dare due differenti spiegazioni sul perché i tre task sono o non sono schedulabili con RM.

- Verificare con che algoritmo (RM o EDF) è possibile schedulare i seguenti task periodici:

	periodo	durata
T1	3	1
T2	4	2
T3	6	1

  Tracciare il diagramma temporale di alcune schedulazioni possibili.

- Un sistema consiste nei tre task periodici (7)

	periodo	durata
T1	3	1
T2	5	2
T3	8	3

  - Qual'è la totale utilizzazione del processore?
  - Qual è il minimo aumento del periodo del task T3 tale che l'insieme di task sia schedulabile con l'algoritmo EDF?

- Un sistema consiste nei seguenti tre task periodici: (7)

	periodo	durata
T1	5	1
T2	8	2
T3	14	4

- Verificare, usando il metodo del tempo massimo di risposta, se i task sono schedulabili con RM. In caso affermativo, si supponga che i primi 'x' istanti del task T2 siano non interrompibili. Se 'x' vale una unità temporale, i task restano schedulabili con RM?

- Un sistema consiste in tre task che condividono una risorsa X utilizzata in modalità non interrompibile. La priorità del task Ti è maggiore di quella del task Tj se i<j. Se l'uso delle risorse è quello indicato in tabella:

	arrivo	Durata totale	Uso risorsa X Start-durata d'uso
T1	2	4	3-2
T2	5	5	
T3	0	6	1-5

  - Qual è la massima durata del tempo di bloccaggio dei task dovuta alla inversione di priorità? Si disegni il diagramma temporale di questa schedulazione.

Programmazione Posix

- Cos'è POSIX 1003? quali parti sono maggiormente note?
- Che risultato fornisce questo programma?

```
#include    <sys/types.h>
#include    <sys/wait.h>
int main(void)
{
      pid_t pid;
      if ( (pid = fork()) < 0) err_sys("fork error");
      else if (pid == 0)      /* primo figlio */
            if ( (pid = fork()) < 0) err_sys("fork error");
            else if (pid > 0) Exit(0);
            /* qui sono il secondo figlio (processo generato) */
            sleep(1);
            printf("Sono il secondo figlio creato da = %d\n", getppid());
            exit(0);


   /* aspetta la terminazione del primo figlio */
      if (waitpid(pid, NULL, 0) != pid) syerr_sys("waitpid error");

      /* Ora sono il processo originale */

      exit(0);
}
```

Programmazione Posix 1003.1b

- Questo programma dà un errore di inserimento modulo. Trovare e giustificare l'errore.

```
#include <linux/kernel.h>
#include <linux/module.h>
MODULE_LICENSE("GPL");

int init_module(void)  //entry point
{
            print("Hello world!\n");
            return 0;
}

void cleanup_module(void)     //exit point
{
            print("Goodbye world!\n");
            return;
}
```

Programmazione Posix 1003.1c

- Questo programma ha il seguente comportamento: esce subito terminando i suoi threads. Trovare l'errore.

```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#define NUM_THREADS     5
```

```
void *stampa(void *threadid)
{
   int i;
   double myresult=0.0;
   printf("sono il thread=%ld. Start\n", threadid);
   sleep(1);
   printf("sono il thread=%ld. End\n",threadid);
   pthread_exit(NULL);
}

int main(int argc, char *argv[])
{
      pthread_t threads[NUM_THREADS]; //qui  memorizzo i thread ID
      int rc;
      long t;
      for(t=0;t<NUM_THREADS;t++){
        printf("Sono il main. Sto creando il thread %ld\n", t);
        rc = pthread_create(&threads[t], NULL, stampa, (void *)t);
        if (rc){ printf("ERRORE nr. %d\n", rc); exit(-1); }
      }
 }
```