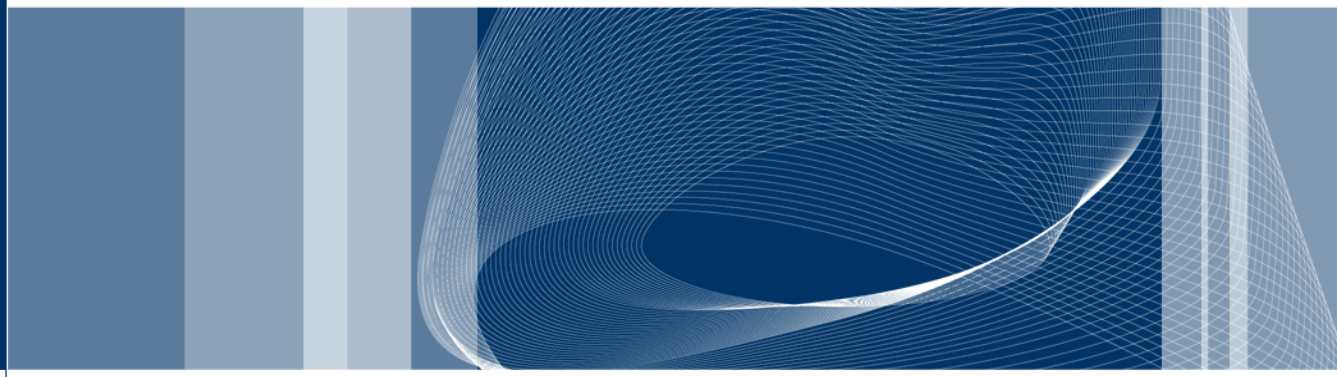
POLITECNICO DI MILANO



Esercizi Threads

Algoritmi e Calcolo Parallelo

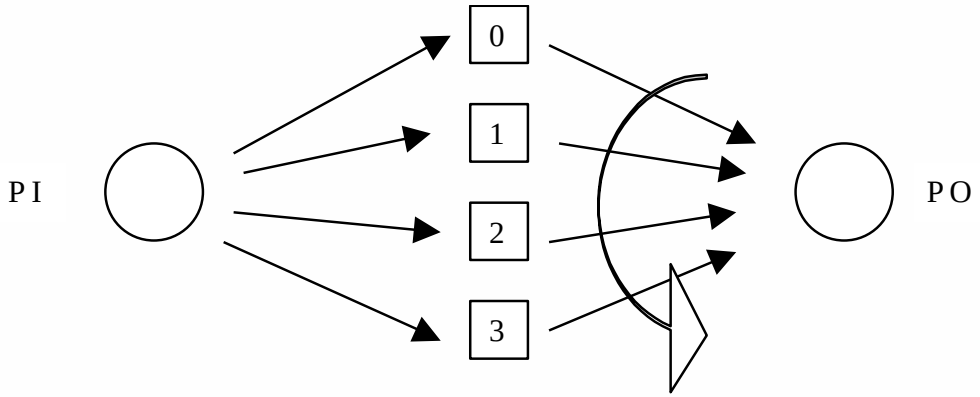
Daniele Loiacono

Esercizio 1

- Si utilizzino i thread per modellizzare la movimentazione di un conto bancario.
- Si implementino i seguenti thread:
  - Un thread *timer* che scandisce il tempo a partire da 1/1/2010 (si ipotizzi per semplicità che tutti i messi abbiano 31 giorni). Per ogni giorno il thread aggiorna gli interessi applicando al capitale corrente un tasso pari al 2% annuo. Gli interessi vengono sommati al capitale l'ultimo giorno dell'anno.
  - Un thread *stipendio* che incrementa il capitale di 1650 euro il giorno 27 ogni 2 mesi.
  - Un thread *mutuo* che decrementa il capitale di 800 euro il giorno 10 di ogni mese.
- Si implementino i thread in modo da consentire un accesso esclusivo alla variabile capitale.

Esercizio 2

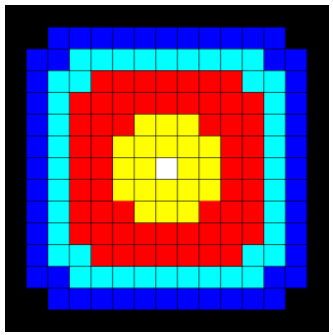
- Il thread PI esegue ripetutamente le seguenti operazioni:
  - legge da tastiera una coppia di valori <i, ch>, dove i è un numero tra 0 e 3, ch un carattere
  - inserisce il carattere ch nel buffer i (ognuno dei quattro buffer contiene al più un carattere)
- Il thread PO considera a turno in modo circolare i quattro buffer e preleva il carattere in esso contenuto, scrivendo uscita la coppia di valori <i, ch> se ha appena prelevato il carattere ch dal buffer i
- Implementare i thread in C++ in modo che l'accesso a ognuno dei buffer sia in mutua esclusione (PI rimane bloccato se il buffer a cui accede è pieno, PO se è vuoto)

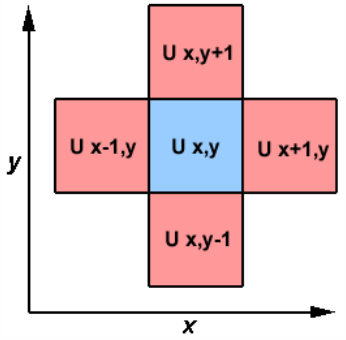


Esercizio 3

- Proporre una soluzione basata su pthreads per parellizzare su una macchina con M processori il calcolo di una semplice equazione di calore su una matrice NxN:

$$U_{x,y} = U_{x,y} + C_x * (U_{x+1,y} + U_{x-1,y} - 2 * U_{xy}) + C_y * (U_{x,y+1} + U_{x,y-1} - 2 * U_{x,y})$$



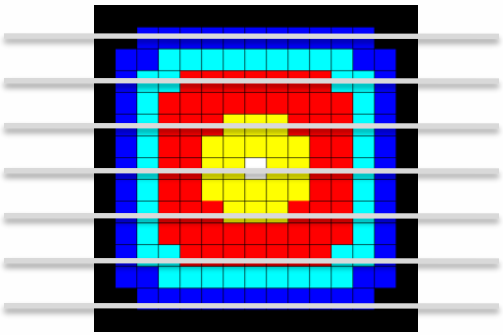


- La soluzione seriale del problema può essere schematizzata come:

```
for (x=1; x < N-1; x++)
   for (y=1; y < N-1; y++)
        u[x][y] = u[x][y] + cx*(u[x+1][y] + u[x+1][y]
        - 2* u[x][y]) + cy*(u[x][y+1] + u[x][y-1]
        - 2* u[x][y]
```

Esercizio 3 – Soluzione

- Si può scomporre la matrice in blocchi ed utilizzare ciascun thread per aggiornare un diverso blocco della matrice



- Ciascun thread può svolgere i propri calcoli in maniera totalmente indipendente.

Esercizio 3bis

- Estendere l'esercizio precedente affinchè vengano svolte K iterazioni di aggiornamento della matrici
- Problemi
  - Sincronizzazione fra i threads tra un'iterazione e quella successiva
  - Gestione efficiente della matrice risultato