Relazione Accordo Bizantino (Protocollo MonteCarlo)

Pezzano Enrico

In questo laboratorio abbiamo ripreso gli argomenti riguardanti del Problema del Consenso Bizantino (utilizzando un protocollo di tipo Monte Carlo); ci sono *n* generali, di questi *n-1* è un traditore.

PROBLEMA:
Le scelte dei quattro processi sono salvate in una matrice di 3x4, in cui le righe rappresentano le quattro scelte che riceve ogni processo onesto (compresa la propria) e le colonne rappresentano le scelte di ogni singolo processo sotto forma di uni o zeri.

	Processo 1	*Processo 2*	*Processo 3*	*Processo 4*
Processo 1	*0*	*1*	*0*	*1*
Processo 2	*0*	*1*	*0*	*0*
Processo 3	*0*	*1*	*0*	*1*

La tabella soprastante indica che il *Processo 1* sceglie *0* e riceve *1* dal *Processo 2*, *0* dal *Processo 3* e *1* dal *Processo 4*.

Il quarto è quello ***villano*** e sceglierà il *contrario* della scelta del processo a cui si riferisce la riga. Per esempio, nella riga del *Processo 1*, il *Processo 4* sceglierà il contrario del *Processo 1*, nella riga del *Processo 2* sceglierà il contrario del *Processo 2* e nella riga del *Processo 3* il contrario di ciò che ha scelto il *Processo 3*.
Si otterrà un accordo nel momento in cui ogni processo ***onesto*** riceverà almeno 3 bit uguali e con il medesimo valore per ogni processo.

	Processo 1	*Processo 2*	*Processo 3*	*Processo 4*
Processo 1	*1*	*1*	*1*	*0*
Processo 2	*1*	*1*	*1*	*0*
Processo 3	*1*	*1*	*1*	*0*

Mostra una condizione di accordo tra i processi onesti (fine dell'algoritmo)

Al turno 0, le scelte di ogni processo sono casuali e può succedere, circa il 25% dei casi ì, che i processi leali si trovino già in accordo. Così facendo, il problema si risolve in 0 turni.
Altrimenti, al turno 1 sceglie il bit analizzando quelli ricevuti da ogni processo e lo comunica agli altri processi. Decisione in base all'algoritmo in sottostante:

$maj(i) \leftarrow$ valore maggioritario tra i ricevuti (incluso il proprio)

$tally(i) \leftarrow$ numero dei valori uguali a $maj(i)$

$\texttt{if } tally(i) \geq 2t + 1$

$\quad \texttt{then } b(i) \leftarrow maj(i)$

$\texttt{else if } testa$

$\quad \texttt{then } b(i) \leftarrow 1$

$\texttt{else } b(i) \leftarrow 0$

Se tally non è ≥ (del doppio del numero dei processi birbanti)+1, allora si procede con il lancio della moneta globale; il risultato verrà comunicato a tutti i processi. L'algoritmo verrà iterato finchè non si raggiungerà un accordo.

CODICE:
Nel *main()* creo le strutture dati necessarie ad immagazzinare le scelte dei vari processi ed il numero di round necessari per raggiungere un accordo. Successivamente è implementato un ciclo *for* che itera l'algoritmo ITERATIONS volte e aggiunge al vettore dei risultati il numero di round utilizzati ad ogni run.
L'algoritmo utilizza la funzione *start()* per inizializzare la matrice delle scelte con valori casuali e attraverso un while che pone la condizione *consensus() == false*, dove la funzione *consensus()* si occuperà di controllare se si è raggiunto un accordo ed esegue la funzione do*_round().*
Quest'ultima calcola *tally* e la *maggioranza* delle scelte ricevute da ogni processo, attraverso le relative funzioni, e, in base all'algoritmo visto precedentemente, sceglie cosa mandare agli altri processi. Ad ogni esecuzione di do*_round()* viene incrementata la variabile *round_number_counter* *(*indica il numero di round in quel momento).

Una volta concluso il *for* si calcola *media*, *varianza* ed il numero di round dopo il quale la probabilità di raggiungere un accordo è maggiore del 99.9%.

CONSIDERAZIONI FINALI:
Iterando il codice per 10^5 volte si hanno i seguenti risultati:

```
[enrico@Macbook-Air-M1 4.1 Protocollo MonteCarlo % ./a.out

Numero di round dopo il quale la probabilità che l'accordo è raggiunto è più gra
nde del 99.9%: 10

Media del numero di round necessari per il consenso: 1.50987

Varianza del numero di round necessari per il consenso: 2.27962
```

Sappiamo inoltre che se $n \geq 3t+1$ (n numero di processi e t numero di processi inaffidabili) il consenso può essere sempre raggiunto in un numero di round dell'ordine di $\boldsymbol{O(t+1)}$, che nel nostro caso è 2; quindi il valore che risulta come media conferma la stima.

La probabilità di raggiungere un accordo è maggiore del 99.9% dopo 10 round, perché, ad ogni round con probabilità(*testa*)=probabilità(*croce*)=*1/2* dopo l'esito del lancio della prima moneta, tutti i processi affidabili avranno raggiunto il consenso; infatti 1-(0.5^10) = 0.999.