Reti di calcolatori

De Gaetano Giovanni	0124002431
Sparaco Asia Maria	0124002519

Descrizione del progetto

Una navicella spaziale deve evitare i detriti provenienti da una tempesta di meteoriti. La navicella (client) entra nel settore spaziale dei meteoriti (server) connettendosi in UDP. La tempesta di meteoriti genera in modo casuale *n* pacchetti UDP ogni 2 secondi che rappresentano i detriti in una griglia MxM. La navicella riceve un alert nel caso in cui il detrito spaziale si trovi sulla sua stessa posizione e può spostarsi prima dell'impatto. Ogni spostamento della navicella viene notificato al server il quale genera nuovi detriti nella direzione della navicella stessa.

Dettagli implementativi dei client/server

Il progetto funziona tramite un'architettura client/server. All'inizio, viene inviato un pacchetto UDP vuoto al server, al fine di catturare l'indirizzo del mittente; successivamente, esso riceve dal server il parametro M, che determina la grandezza della griglia. A questo punto, il client invia al server la posizione della navicella N all'interno della griglia. Inizia così un ciclo infinito, tramite il quale il server genera n detriti con posizione casuale (ogni 2 secondi); esso controlla, inoltre, se uno dei detriti si trova nella stessa posizione della navicella. In tal caso, viene generato un messaggio di ''alert'', il quale viene inviato al client. La navicella, a questo punto, si sposta in una delle 8 posizioni adiacenti (essa non può uscire dalla griglia), la quale viene comunicata al server. Infine, il server si occupa di generare i detriti nell'intorno 3x3 della navicella.

Parti rilevanti del codice

Client:

```
Sendto(sfd,buffer,sizeof(buffer),0,(struct sockaddr*)&server,sizeof(server));
Recvfrom(sfd,&M,sizeof(M),0,(struct sockaddr*)&server,&len);

pos[0] = rand() % M;
pos[1] = rand() % M;

Sendto(sfd,pos,sizeof(pos),0,(struct sockaddr*)&server,sizeof(server));
```

Server:

```
Recvfrom(sfd,buffer,sizeof(buffer),0,(struct sockaddr*)&client,&len);

Sendto(sfd,&M,sizeof(M),0,(struct sockaddr*)&client,sizeof(client));

Recvfrom(sfd,pos,sizeof(pos),0,(struct sockaddr*)&client,&len);
```

```
while(1){
    for(int i=0;i<n;i++){
        if(changePos){
            generateDet(pos,detrito);
        }
        else{
            detrito[0] = rand() % M;
            detrito[1] = rand() % M;
        }
        griglia[detrito[0]][detrito[1]] = 'x';
        Sendto(sfd,detrito,sizeof(detrito),0,(struct sockaddr*)&client,sizeof(client));

        if((detrito[0] == pos[0]) && (detrito[1] == pos[1])){
            crash = 1;
        }

    }
```

```c
if(crash){
    crash = 0;
    char alert[10] = "alert";
    Sendto(sfd,alert,strlen(alert),0,(struct sockaddr*)&client,sizeof(client));
    Recvfrom(sfd,pos,sizeof(pos),0,(struct sockaddr*)&client,&len);
    changePos = 1;
    griglia[pos[0]][pos[1]] = 'n';
}
sleep(2);
```

```c
void generateDet(int pos[],int detrito[]){
    int offsetRow,offsetCol;
    if(pos[0] == 0){
        offsetRow = rand() % 2; // tra 0 e 1;
    }
    else if(pos[0] == M-1){
        offsetRow = (rand() % 2)-1; // tra -1 e 0
    }
    else{
        offsetRow = (rand() % 3) - 1; // tra -1,0 e 1
    }
    if(pos[1] == 0){
       offsetCol = rand() % 2; // tra 0 e 1
    }
    else if(pos[1] == M-1){
       offsetCol = (rand() % 2)-1; // tra -1 e 0
    }
    else{
        offsetCol = (rand() % 3) - 1; // tra -1,0 e 1
    }
    detrito[0] = pos[0] + offsetRow;
    detrito[1] = pos[1] + offsetCol;
}
```

Client:

```c
while(1){
    Recvfrom(sfd,buffer,sizeof(buffer),0,(struct sockaddr*)&server,&len);
    if((strcmp(buffer,"alert")) == 0){
        changePos();
        printf("La navicella si sposta\n");
        Sendto(sfd,pos,sizeof(pos),0,(struct sockaddr*)&server,sizeof(server));
    }
}
```

```
void changePos(){
    int offsetRow;
    if(pos[0] == 0){
        offsetRow = rand() % 2; // tra 0 e 1;
    }
    else if(pos[0] == M-1){
        offsetRow = (rand() % 2)-1; // tra -1 e 0
    }
    else{
        offsetRow = (rand() % 3) - 1; // tra -1,0 e 1
    }
    pos[0] += offsetRow;
    if(pos[1] == 0){
        if(offsetRow == 0)
            pos[1] += 1;
        else
            pos[1] += rand() % 2; // tra 0 e 1
    }
    else if(pos[1] == M-1){
        if(offsetRow == 0)
            pos[1] += -1;
        else
            pos[1]+= (rand() % 2)-1; // tra -1 e 0
    }
    else{
        if(offsetRow == 0)
            pos[1] += 2 * (rand() % 2) -1; //tra -1 e 1
        else
            pos[1] += (rand() % 3) - 1; // tra -1,0 e 1
    }
}
```

Compilazione ed esecuzione del progetto

Compilazione:

1. gcc client.c -o client
2. gcc server.c -o server

Esecuzione:

1. ./server <size M> &
2. ./client <Server's IP>