Tema 2 - Monitorizarea Traficului(A)

Vasile Alexandru, anul II, grupa A3

December 2019

1 Introducere

In aceasta tema vor fi prezentate cateva aspecte ale proiectului propus, arhitectura acestuia, tehnologiile folosite, modul de functionare, precum si caracterstricile sale. Acest proiect va fi implementat folosind limbajul C, iar ca baza de date vor fi folosite fisier text obisnuite. Scopul acestui proiect este de a implementa un sistem de monitorizare al traficului, care va transmite informatii utile participantilor la trafic, precum traseul cel mai scurt, incidente in trafic, limite de viteza, informatii despre vreme/evenimente sportive etc.

2 Tehnologii utilizate

Pentru realizarea proiectului voi folosi modelul TCP-concurent(orientat conexiune), pentru a putea servi mai multi clienti simultan. Acest lucru va fi realizat intr-o bucla infinita, in care serverul asteapta realizarea conexiunii cu un nou client. Daca aceasta conexiune se realizeaza cu succes, serverul va crea un proces copil prin primitiva fork(), proces copil care va prelua comenzile clientului. De asemenea, in momentul apelului primitivei fork(), procesul copil va mosteni un numar de ordine(contor incrementat in procesul parinte dupa fiecare acceptare a unui client nou). Prin acest numar de ordine procesul copil va sti carui client din vectorul de descriptori sa transmita informatii, iar la nivel de server, numarul de ordine va fi folosit si pentru a instiinta procesul parinte care dintre procesele copil doreste sa transmita un mesaj(pentru a putea comunica clientilor noi update-uri de la server, vom salva toti descriptorii de socket intr-un vector de clienti, iar fiecare proces fiu va mosteni index-ul clientului pe care il serveste).

Comunicarea proces copil - proces parinte in server se va realiza prin pipe-uri, avand cate un pipe pentru fiecare proces copil/client. De asemenea vor exista 2 pipe-uri suplimentare, iar procesul parinte va citi din primul pipe cand primeste primul semnal definit(SIGUSR1) si din al doilea pipe cand va primi al doilea semnal definit(SIGUSR2). Scopul ambelor pipe-uri este acela ca, atunci cand un proces copil din server trimite un semnal clientului, parintele va citi prima data din unul din cele 2 pipe-uri suplimentare numarul de ordine al procesului copil,

apoi citeste informatia propriu-zisa din din pipe-ul din matricea de pipe-uri cu numarul de ordine respectiv.

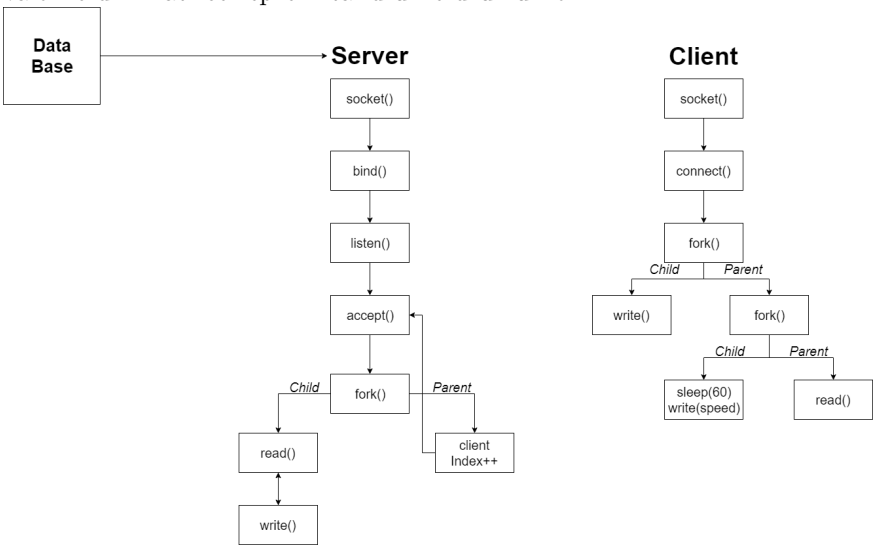
Am ales folosirea protocolului de comunicatie TCP, deoarece consider ca este necesara conexiune sigura, in care trimiterea corecta a pachetelor, precum si ordinea acestora, la toti clientii, este asigurata, deoarece:

-nu exista o trimitere continua de pachete intre client si server, lucru ce ar putea duce la cresterea semnificativa a overhead-ului(datorita pachetelor ACK), ceea ce ar putea face folosirea UDP-ului mai favorabile. In schimb, pachetele sunt transmise periodic, la fiecare 60 de secunde pentru actualizarea vitezei si atunci cand clientul transmite o comanda noua serverului sau serverul trebuie sa transmita o informatie noua clientilor.

-integritatea pachetelor, precum si ordinea in care acestea sunt trimise, este necesara pentru a asigura faptul ca informatii precum traseul calatoriei si alte informatii despre trafic sunt transmise in mod corect; altfel ar putea aparea cazuri in care de exemplu un client ar putea primi un traseu gresit catre destinatia dorita, datorita unui incident in trafic nereceptionat/receptionat in oridine gresita.

3 Arhitectura aplicatiei

Serverul va avea acces la o colectie de fisiere care va reprezenta o mapa virtuala, iar la nivel de program, aceast mapa va fi reprezentat prin mai multe matrici de adiacenta, precum matricea idDrum, unde liniile si coloanele reprezinta locatii, iar valorile din matrice reprezinta id-urile drumurilor.



In procesul server, vom accepta multipli clienti, carora le vom asigna cate un proces copil fiecaruia, care se va ocupa de comenzile clientului, intr-o structura repetitva in care, se va asteapta citirea unei comenzi de la client prin apelul blocant read(), comanda fiind apoi procesata, procesul fiu comunicand cu parintele

prin semnale si pipe-uri in cazul in care anumite informatii valabile pentru toti clientii sunt actualizate(ex: Daca un client raporteaza o strada blocata, acel proces fiu va trimite un semnal care procesul parinte, care va astepta citirea index-ului procesul copil dintr-un pipe, iar apoi informatia propriu-zisa va fi citita din pipe-ul cu indicele respectiv. Dupa aceea aceasta informatie va fi transmisa tuturor clientilor conectati.).

In client, procesul parinte va avea 2 procese copil. Primul proces copil se va ocupa de transimiterea periodica a vitezei catre server(la intervale de 60 de secunde, aces proces est automat), al doilea proces copil va citi informatii primite de la server, pe care le va afisa sau cu ajutorul carora va trimite noi cereri catre server. Procesul parinte va fi responsabil pentru citirea comenzilor date direct de catre user si transimiterea acestora catre server.

4 Detalii de implementare

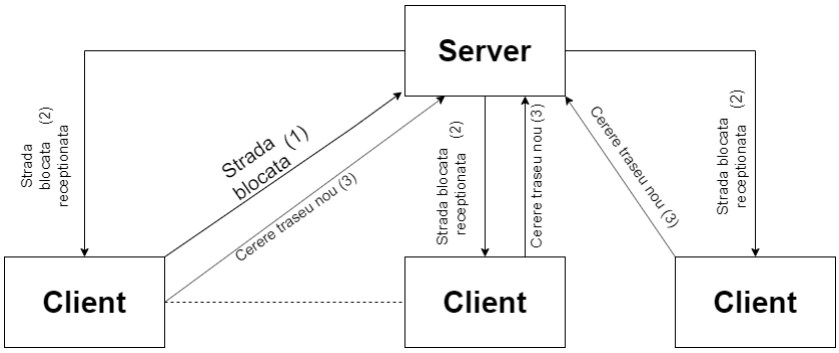
Implementarea diferitelor functionalitati:

- calculul traseului de lungime minima

  Dupa ce clientul va transmite serverului locatiile de plecare, respectiv sosire, serverul va calcula drumul de lungime minima intre aceste 2 locatii, luand in considerare strazile care nu sunt disponibile(blocate). Drumul va fi salvat intr-un vector cu id-urile locatiile ce apartin traseului. Apoi serverul va accesa sistemul de fisiere si va transmite clientului pe rand numele locatiilor cu id-urile din vector si numele strazilor dintre acele locatii.

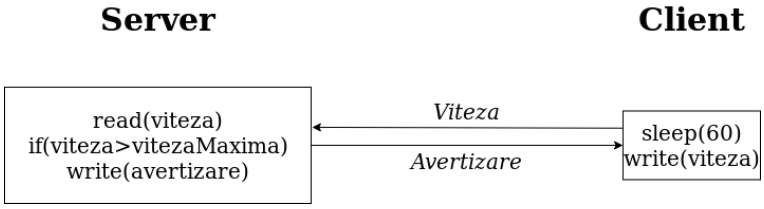
- raportarea unei strazi blocate/deblocate

  Un client poate raporta daca o strada este blocata/deblocata, informatie ce va fi trimisa procesului copil din server, care va trimite un semnal procesului parinte din server, iar acesta va trimite mesajul tuturor clientilor, moment in care fiecare client va cere automat serverului un nou traseu (care poate fi identic cu cel vechi daca schimbarile nu il influenteaza in vreun fel) si va trimite automat catre server viteza cu care circula. De calcularea noului traseu pentru fiecare client se va ocupa procesul copil corespondent din server, dupa ce acesta din urma primeste de la procesul parinte din server informatiile noi despre mapa.



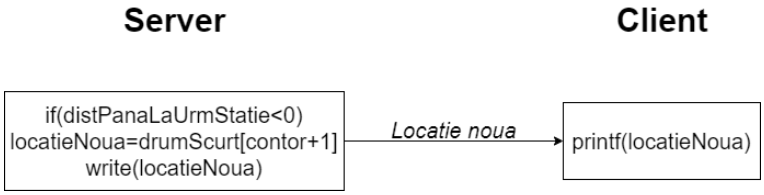
- raportarea automata a vitezei

  Toti clientii vor trimite automat viteza cu care circula la fiecare 60 de secunde. Acestia vor fi informati de catre server despre limita de viteza pe strada respectiva, precum si daca acestia incalca viteza legala. De asemnea, de fiecare data cand un incident este raportat, viteza fiecarui sofer va fi transmisa automat catre server.



- actualizarea locatiei curente

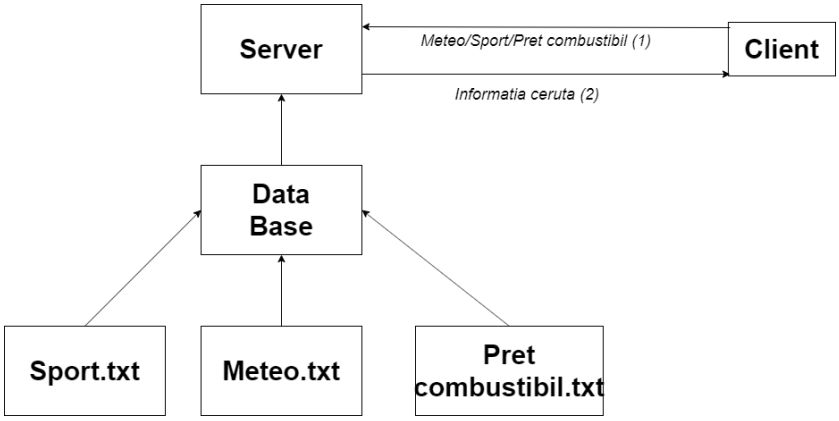
  Serverul memoreaza intotdeauna distanta pe care clientul o are de parcurs pana la urmatoarea locatie. Cu ajutorul vitezei trimise de la client, serverul calculeaza distanta pe care o va parcurge clientul in urmatoarele 60 de secunde si actualizeaza distanta. Daca distanta ajunge la 0 sau devine negativa inseamna ca am ajuns la urmatoarea locatie din traseu, moment in care un mesaj cu locatia respectiva este trimis catre client. La ultima locatie, clientul va primi un mesaj in care este anuntat ca a ajuns la destinatie, moment in care acesta se inchide.



- informatii despre vreme, evenimente sportive, pret combustibil la statiile peco

  Clientii care vor alege la inceputul traseului optiunea de a putea primi informatii suplimentare, vor avea la dispozitie comenzi pentru afisarea a

diverse informatii utile, precum stiri sportive, evenimente meteo, preturile la combustibil la statiile peco, informatii ce vor fi stocate intr-un sistem de fisiere text la care serverul va avea acces si le va transmite clientilor informatiile clientilor. Cei care nu au ales aceasta optiune nu vor avea acea acces la aceste informatii



- posibilitatea de a face un detur catre o statie peco

  Prin comanda "Statie peco" clientilor li se va transmite la cati km distanta sunt de cel mai apropiat peco si vor avea optiunea de a schimba traseul catre acea statie, iar serverul le va comunica cel mai scurt traseu din punctul curent catre statia peco daca clientii doresc acest lucru. Daca un client a facut un ocol catre o statie peco, dupa ce a ajuns la acea statie, va avea optiunea de a finaliza traseul sau de a relua traseul catre destinatia initiala. Daca clientul alege cea de-a doua varianta, serverul ii va recomanda din nou traseul cel mai scurt de la acea statie catre destinatia finala.

Cateva exemple de cod semnificativ:

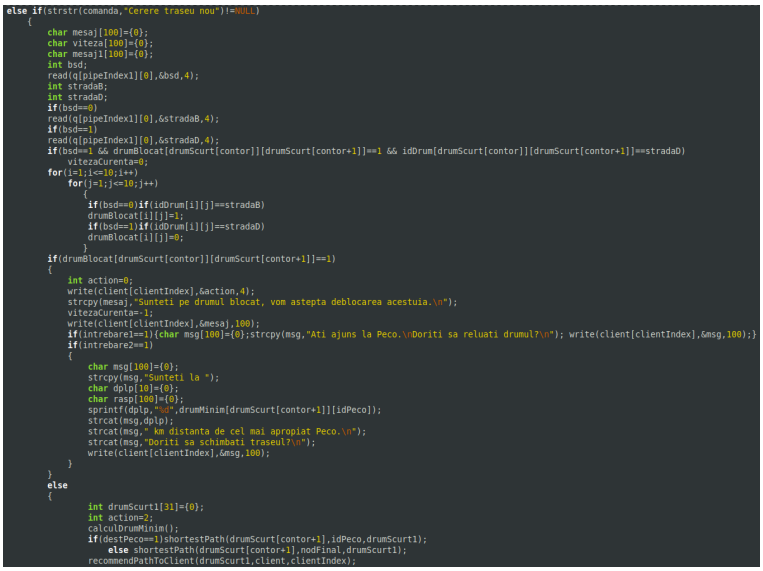
Functia shorthestPath foloseste o matrice de cost minim, obtinuta folosing algoritmul lui Warshall, construind traseul minim de la punctul de plecare la punctul de sosire, calculand la fiecare pas o noua locatie ce apartine traseului. Daca o locatie se va afla in traseul de lungime minima, atunci distanta minima de la ultima locatie pe care am calculat-o la pasul anterior la destinatie trebuie sa fie egala cu suma dintre: -lungimea dintre locatia curenta(care stim ca apartine traseului de cost minim) si locatia despre care vrem sa determinam daca se afla in traseul minim -lungimea minima dintre locatia despre care dorim sa determinam apartenenta la traseu si destinatie

```
174 void shortestPath(int punctCurent,int sosire,int* drumScurt)
175 {
176     int c=0;
177     int k,ok=0;
178     drumScurt[c]=punctCurent;
179     c++;
180     while(punctCurent!=sosire && ok==0 )
181     {
182         for(k=1;k<=10;k++)
183             if(ok==0)
184                 if(lungimeDrum[k][punctCurent]+drumMinim[k][sosire]==drumMinim[punctCurent][sosire] && punctCurent!=k)
185                     {
186                         drumScurt[c]=k;
187                         c++;
188                         punctCurent=k;
189                         if(punctCurent==sosire)ok=1;
190                             break;
191                     }
192     }
193 }
```

Clientul trimite la fiecare 60 de secunde viteza sa catre server. Cu ajutorul acestei informatii serverul calculeaza distanta parcursa de client cu viteza respectiva pe parcursul unui minut si actualizeaza distanta ramasa pana la statia urmatoare(variabila dplus). Daca aceasta aceasta distanta devine negativa inseamna ca statia urmatoare a fost parcursa, deci trimitem un mesaj clientului cu locatia pe langa care a trecut. Daca acea locatie coincide cu destinatia clientului, serverul trimite user-ului mesajul "Ati ajuns la destinatie". Cand clientul primite acest mesaj, acesta stie ca a ajuns la destinatie, iar inainte de a se inchide trimite mesajul "kill" catre server, pentru ca acesta din urma sa stie ca acel client nu mai este conectat la server

```
dplus=dplus-(float)vitezaCurenta/60;
if(dplus<=0 && drumScurt[contor+2]!=0)
  {
      char locatie[100]={0};
      idToLocatie(drumScurt[contor+1],locatie);
      strcat(locatie,"\n");
      write(client[clientIndex],&locatie,100);
      contor++;
      drumCurent=idDrum[drumScurt[contor]][drumScurt[contor+1]];
      dplus=(float)lungimeDrum[drumScurt[contor]][drumScurt[contor+1]]
  }
 else if(dplus<=0 && drumScurt[contor+2]==0)
      {
      char locatie[100]={0};
      idToLocatie(drumScurt[contor+1],locatie);
      write(client[clientIndex],&locatie,100);
      if(destPeco==0)
      {
          strcat(locatie,"\nAti ajuns la destinatie");
          write(client[clientIndex],&locatie,100);
      }
      else if(destPeco==1)
      {
          vitezaCurenta=-1;
          char lc[100]={0};
          strcpy(lc,"Ati ajuns la Peco.Doriti sa reluati traseul?\n");
          intrebare1=1;
          write(client[clientIndex],&lc,100);
      }
  }
```

Comanda "Cerere traseu nou" este trimisa automat de client catre server, atunci cand clientul receptioneaza informatii despre o strada blocata/deblocata. Procesul copil va fi city din pipe daca strada primita este blocata sau deblocata, va actualiza matricea drumurilor, va recalcula matricea de cost minim si va folosi functiile shortestPath si recommendPathToClient pentru a transmite clientului traseul cerut.



5 Concluzii

Implementarea proiectului ar putea fi imbunatatia prin folosirea unei baze de date SQL, fisiere XML, precum si prin extinderea mapei virtuale puse la dispozitia serverului. De asemenea, utilizarea altor tehnologii precum thread-urile ar putea aduce anumite avantaje proiectului, datorita proprietatii acestora de memorie partajata. Proiectul ar putea fi imbunatatit si prin implementarea de noi functionalitati(calculul procentului de combustibil ramas in rezervor, durata de timp in care o strada este blocata, reglarea climei etc).

6 Bibliografie

https://www.geeksforgeeks.org/floyd-warshall-algorithm-dp-16/
https://stackoverflow.com/questions/28636260/
https://www.quora.com/How-do-I-print-a-colored-output-in-C
https://drive.google.com/file/d/1J96c88-sMGgk6wffgAltB_5nKD_L6hMB/view
https://drive.google.com/file/d/1zLL8yGFZbI0nSE-c9KpFiDBN29K1E-xR/view
https://hub.packtpub.com/understanding-network-port-numbers-tcp-udp-and-icmp-on-an-operatir