CS 520 Homework 3 Abdullah Samarkandi   
--------------------------- -----------------------------

*1. Does the distance between the adjacent buses remain the same? If not, what should be done to ensure that it be the same?*

The time of drive of the Bus between the two adjacent stops and passenger boarding time are not changing, the distance between two buses also doesn’t change. Also, arrival time of passenger are random, there can be any number of passenger arriving at stop. This will result in delay of bus leaving the stop if there are too many passengers to board.

To keep uniform distance between buses a constant number of passengers need to be boarded into the buses.

*2. What is the average size of a waiting queue at each stop (and what are its maximum and minimum)? (You may provide this information on an hourly [simulation time] base.)*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | stops | | | | | | | | | | | | | | | |
| hours |  | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 0 | 1 | 1 | 1 | | 0 | 1 | 1 | 1 | 0 | 2 | 1 | 0 | 1 | 1 | 0 | 2 |
| 1 | 36 | 2 | 90 | | 40 | 20 | 90 | 47 | 39 | 108 | 51 | 11 | 99 | 37 | 19 | 112 |
| 2 | 73 | 24 | 69 | | 28 | 88 | 106 | 55 | 12 | 91 | 49 | 9 | 88 | 12 | 177 | 119 |
| 3 | 134 | 69 | 42 | | 105 | 221 | 172 | 159 | 101 | 99 | 38 | 9 | 188 | 66 | 23 | 129 |
| 4 | 101 | 109 | 51 | | 24 | 199 | 171 | 178 | 92 | 32 | 218 | 298 | 245 | 219 | 214 | 142 |
| 5 | 419 | 389 | 354 | 312 | | 289 | 255 | 229 | 151 | 109 | 105 | 77 | 53 | 14 | 49 | 210 |
| 6 | 251 | 155 | 141 | 110 | | 55 | 19 | 533 | 466 | 432 | 395 | 384 | 370 | 319 | 279 | 249 |
| 7 | 594 | 493 | 400 | 354 | | 402 | 298 | 294 | 198 | 175 | 154 | 98 | 52 | 9 | 265 | 552 |
| 8 | 152 | 104 | 69 | 12 | | 605 | 599 | 554 | 532 | 449 | 435 | 403 | 330 | 294 | 281 | 222 |

Average waiting queue size at each stop is 193 passengers per minute,

maximum value is 605, minimum value is 0.

The horizintal line: the Bus-stops. The verticle line: # people in queue.

Chnging the varibles of the experiment:

- Decreasing the mean arrival rate of passengers led to more passengers arrive. Length of waiting queue is inversely proportional to mean interval arrival time.

- Decreasing the driving time of bus lead to the bus arraiving early at each stop, the wait queue at bus stops were longer.

- Changing the number of passenger randomly means the time spent by bus on any stop is greater in this case

The code:

Compile with: gcc -std=c++11 bus.cpp -o bus -lstdc++

It works with linux lab

#include <stdio.h>

#include <stdlib.h>

#include <iostream>

#include <algorithm>

#include <queue>

#include <random>

#include <string>

using namespace std;

struct event{

double time;

char type;

int busNumber,

stopNumber;

friend bool operator>(const event& lhs, const event& rhs)

{

return lhs.time > rhs.time;

}

};

int main(){

double timer=0.0;

double simulationTime= 8 \* 60 \* 60;

int numBuses = 5;

int boarding\_time= 2;

int meanArrivalRate= 5;

int time\_to\_next\_stop= 2 \* 60;

int countBusStop= 15;

long checkpoint=0;

int line[15] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

priority\_queue<event, vector<event>, greater<event> > q;

random\_device rd;

mt19937 rng(rd());

uniform\_int\_distribution<> dist(1,countBusStop);

for (int i=0; i< numBuses ; i++){

struct event arrivalEvent;

// strcpy (arrivalEvent.type, "arrival");

arrivalEvent.type = 'a';

arrivalEvent.time = 0;

arrivalEvent.busNumber = i;

arrivalEvent.stopNumber= countBusStop/numBuses % countBusStop;

q.push(arrivalEvent);

cout << "arrival\n";

}

for (int i=0; i< countBusStop; i++){

struct event personEvent;

// strcpy (personEvent.type, "person");

personEvent.type = 'p';

personEvent.time = 0;

personEvent.busNumber = 0;

q.push(personEvent);

cout << "person\n";

}

while(!q.empty()){

struct event x = q.top();

q.pop();

printf("%c\t%f\t%d\t\n",x.type,x.time,x.stopNumber);

switch (x.type) {

case 'a':

cout << "case a \n";

if(line[x.stopNumber] > 0){

struct event boarderEvent; //boarding

boarderEvent.type = 'b';

int temp = timer + boarding\_time;

boarderEvent.time = temp ;

int theBus = rand() % 5 ;

boarderEvent.busNumber = theBus;

boarderEvent.stopNumber = x.stopNumber;

q.push(boarderEvent);

//cout<<"Generated a boarder event \n";

cout<<"stopNumber "<<boarderEvent.stopNumber<<" busNumber "<<boarderEvent.busNumber

<< " time "<< boarderEvent.time<<endl;

//generate boarder event

}

else{

struct event arrivalEvent; //arrival

arrivalEvent.type = 'a';

arrivalEvent.time = timer + time\_to\_next\_stop;

cout <<"a timer " <<arrivalEvent.time<< endl;

//cout << "arrivalEvent.time"<< x.time <<"\n";

x.stopNumber= x.stopNumber + 1 % countBusStop;

q.push(arrivalEvent);

//generate arrival event

}

break;

case 'b':

cout << "case b\n"; //arrival

if(line[x.stopNumber] == 0){

struct event arrivalEvent;

arrivalEvent.stopNumber= (int)(line[x.stopNumber] + 1) % countBusStop;

int theBus = rand() % 5 ;

arrivalEvent.busNumber = theBus+1 % numBuses;

arrivalEvent.stopNumber = x.stopNumber +1 % countBusStop;

q.push(arrivalEvent);

//generate arrival event at x.stop\_num+1

}

else{

while (line[x.stopNumber] != 0) { //boarding

cout <<"boarding at stopNumber " <<x.stopNumber <<" busNumber "<< x.busNumber

<<"time"<<x.time<< "\n";

timer = x.time +boarding\_time;

cout <<"before boarding, the line is: " <<line[x.stopNumber]<< "\n";

line[x.stopNumber]= line[x.stopNumber]-1;

cout <<"after boarding, the line is: " <<line[x.stopNumber]<< "\n";

//generate boarder event at time = clock + boarding\_time

}

}

break;

case 'p':

cout << "new person\n";

struct event personEvent;

int xx = rand() % 30 ;

//double y= log(1- xx)/5;

//time + xx

personEvent.time = timer + xx ;

line[personEvent.stopNumber]++;

cout <<"p timer " <<personEvent.time<< endl;

personEvent.stopNumber =x.stopNumber;

q.push(personEvent);

//line[x.stopNumber]+1;

//generate person event at clock + random()

break;

}

}

}

//printf("%c\t%f\t%d\t\n",x.type,x.time,x.stopNumber);