Problem to resolve

We are going to design and implement an elevator system

Background and pre-request

1. There is a building with 11 floors, from 1 to 11, no basement;
2. There are 3 same elevators in this building;
3. Elevator’s speed is 1 floor per time unit, no matter going up or going down;
4. no time cost on passenger getting in/out; but when elevator picks passenger, or passenger gets out of elevator, elevator must stay at the same floor;
5. once passenger gets in an elevator at T0, his/her destination floor must be input at T0 + 1; This is guaranteed;
6. an elevator can carry at most 20 passengers;
7. initial position of all elevators is 1st floor;
8. clock of elevator world is controlled by client, clock ticks when API request comes;
9. elevator going up 1 floor requires 0.6 power unit, and going down 1 floor needs 0.4 power unit;
10. all passengers are rational;
11. please ignore the weight of passenger;
12. both user experience and power cost should be considered;

API requirement

1. API POST ENDPOINT/reset, which resets the elevator system to initial status, and return the elevator stats;
2. API POST ENDPOINT/workload, increases world time by +1, and sends user requests if available. The parameter to this API is a JSON string, which stands for workload data, is converted by below format objects:

{

'request': [

[user1, startFloor, direction],

[user2, startFloor, direction],

],

'goto': [

[user3, destFloor],

[user4, destFloor],

]

}

Be noted:

Any single field of above workload data is integer value;

For direction, 1 means going up and 0 means going down;

The workload data stands for new requests happen in current time unit concurrently;

The workload data could be empty, means no request coming, just increase the world time;

Fields 'request' and 'goto' are optional;

1. Both APIs must return stats of all 3 elevators through a JSON string, which is converted by below objects:

{

'1': {

'floor': integer,

'users': [user1, user2,...]

},

'2': {

'floor': integer,

'users': [user3, user4,...]

},

'3': {

'floor': integer,

'users': [user5, user6,...]

}

}

Such as: {u'request': [[0, 1, 1]], u'goto': []}, and client gets {1: {'users': [0], 'floor': 1}, 2: {'users': [], 'floor': 1}, 3: {'users': [], 'floor': 1}}

Then: {u'request': [], u'goto': [[0, 3]]}, and result is {1: {'users': [0], 'floor': 2}, 2: {'users': [], 'floor': 1}, 3: {'users': [], 'floor': 1}}

Be noted:

fields '1', '2', '3' are required to present elevator's name;

'floor' of output is required to present current position of the elevator;

'users' is used to present who are in this elevator, it can be empty if no one is carried;

when client detects a user already in an elevator at the first time, the 'goto' request of this user must be sent in next API call, this is guaranteed;

Test client

1. We prepared a test client in python, which can be found at http://pa-dbc1110.eng.vmware.com/jjin/testclient/
2. This test client requires python 2.6 or 2.7, python3 is not supported;
3. It covers 70% final test cases, through assertion;
4. Default timeout for a single API call is 100ms, no retry during verification.

Referee

1. Two kinds of workloads are prepared to verify the elevator system, including:
2. Predictable case, which covers 8:30 – 9:00 am, 12:00 – 1:00 pm, 5:30 – 7:30 pm
3. Purely random case simulates pure random requests

Both workloads will be processed by elevator system, and we take a sum of 0.9 \* predicable\_score + 0.1 \* random\_score.

1. Failed run gets 0 point;
2. How to evaluate user experience:

The latter a user arrives, the more his/her satisfaction decreases

AVG\_USER\_SATISFACTION\_RATE = sum(USER\_SATISFACTION\_RATE) / USER\_COUNT

1. How to evaluate power cost:

WORST\_POWER\_COST = sum(abs(dest\_floor – start\_floor) \* (0.6 or 0.4 depends on the direction))

POWER\_COST\_RATE = ACTUAL\_POWER\_COST / WORST\_POWER\_COST

1. Score for a single workload:

30 + max(70 \* AVG\_USER\_SATISFACTION\_RATE / POWER\_COST\_RATE, 0)

30 is the base score for a full pass run.

1. We provide 1 free run with final workload. If you want to try more, 5 points will be taken from final score for each try.

Best Score from Committee

Average customer satisfaction rate: 79.077%

Score: 307.87

Average customer satisfaction rate: 93.786%

Score: 87.27

Total score: 285.81