Project Report for DSA

Junjie Ding Xingcheng Dong

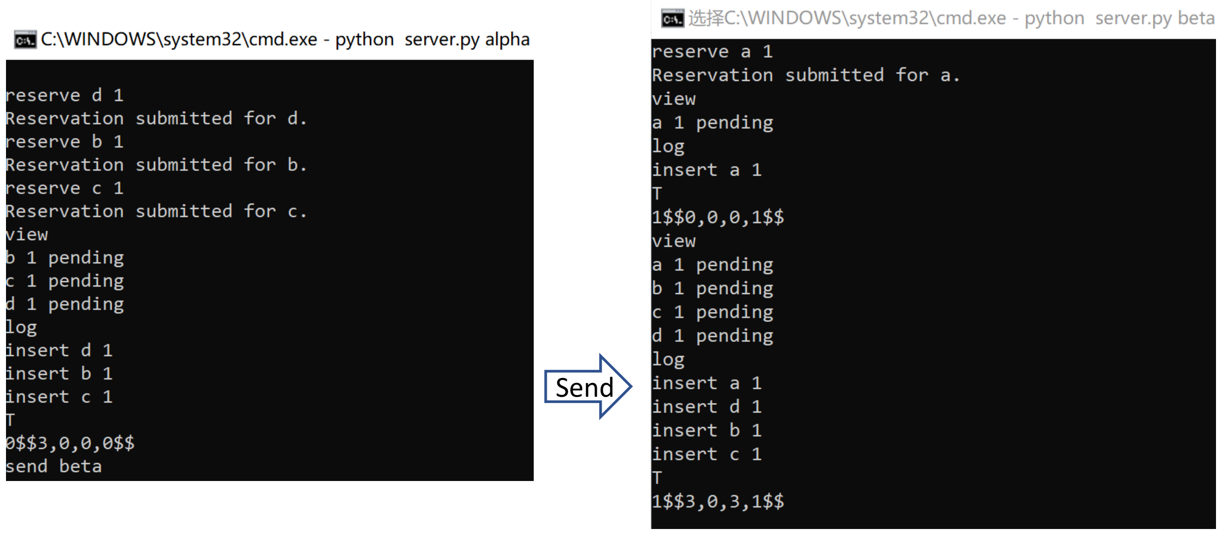
Design and Implementation:

For each site in our system, we start a thread to keep listening on the UDP port to see if there is any message sending to the current site. The main function includes a while loop to keep asking for the input and send the messages. We used a python dictionary to store the local copy of reservation list, the key is the client name and the value is the list of the flights. The log is a python list to keep track of the events and the operations(insert/delete/confirm) of the events.

Description of what events on send and sendall.

When we get a input 'send <sent_id>', we will send a message to the specific site. The message is a serialized object of a long string. We loop through the log of current site, for each event R in the log, we will add that event in our sent message if the current site knows that current site knows the site of sent_id does not know the event R. Each message will also include the current site id and the T matrix.

How to determine reservation to be confirmed:


C:\WINDOWS\system32\cmd.exe - python server.py alpha
Send
选择C:\WINDOWS\system32\cmd.exe - python server.py beta

```
reserve d 1
Reservation submitted for d.
reserve b 1
Reservation submitted for b.
reserve c 1
Reservation submitted for c.
view
b 1 pending
c 1 pending
d 1 pending
log
insert d 1
insert b 1
insert c 1
T
0$$3,0,0,0$$
send beta
```

```
reserve a 1
Reservation submitted for a.
view
a 1 pending
log
insert a 1
T
1$$0,0,0,1$$
view
a 1 pending
b 1 pending
c 1 pending
d 1 pending
log
insert a 1
insert d 1
insert b 1
insert c 1
T
1$$3,0,3,1$$
```

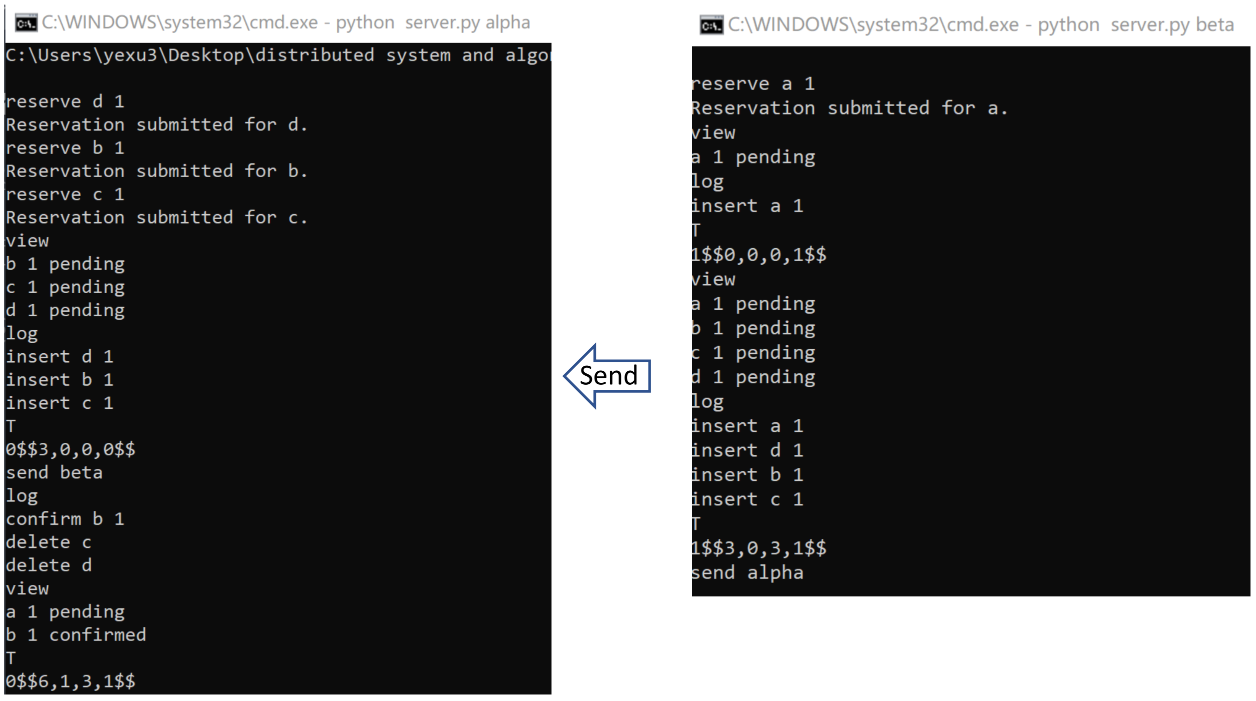
3, 0

0, 0

3, 0

3, 1

1. We have four steps to confirm one reservation: (1) this process which reserve a seat should send its PL and T to all the other process, and get all the other processes massages back. Basically, the algorithm will check the T matrix to see if current sites know all other sites knows about the reseravation. (2) after sending this reservation and getting massages back from all the other processes, the process could decide whether to confirm this reservation or cancel it, because at that time, the process finally know whether the seats are available or the seats has been taken by other processes. (3) before we get feedback from all the other processes, all the process reserved about all the flight will still be in the partial log and all the reserved information will be in dictionary with status of "pending". (4) after we decide whether confirm the reservation or cancel the reservation, we will build a new event and put the new confirm or delete event in partial log



C:\WINDOWS\system32\cmd.exe - python server.py alpha

```
C:\Users\yexu3\Desktop\distributed system and algo

reserve d 1
Reservation submitted for d.
reserve b 1
Reservation submitted for b.
reserve c 1
Reservation submitted for c.
view
b 1 pending
c 1 pending
d 1 pending
log
insert d 1
insert b 1
insert c 1
T
0$$3,0,0,0$$
send beta
log
confirm b 1
delete c
delete d
view
a 1 pending
b 1 confirmed
T
0$$6,1,3,1$$
```

C:\WINDOWS\system32\cmd.exe - python server.py beta

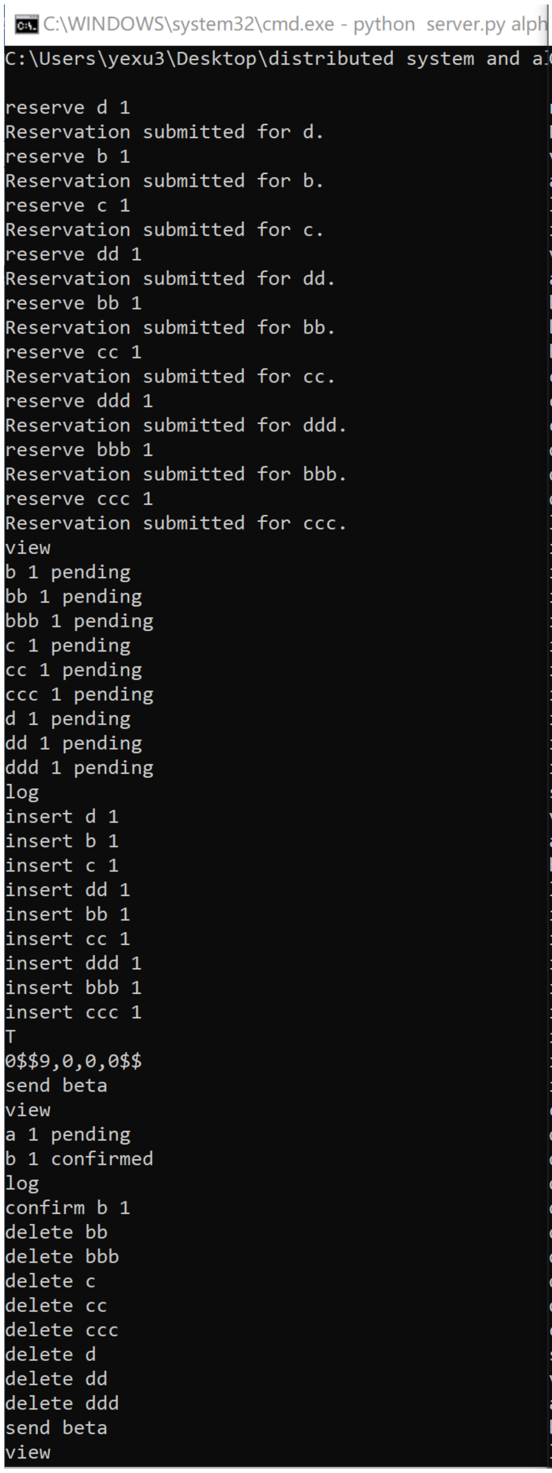
```
reserve a 1
Reservation submitted for a.
view
a 1 pending
log
insert a 1
T
1$$0,0,0,1$$
view
a 1 pending
b 1 pending
c 1 pending
d 1 pending
log
insert a 1
insert d 1
insert b 1
insert c 1
T
1$$3,0,3,1$$
send alpha
```

6, 1	3, 0
3, 1	3, 1

2. The reservation could be confirmed under either of two conditions: (1) no other processes have taken the seats, and the seats are available now. (2) our clients for the seats are alphabetically smaller than the other processes who have taken the seats without confirming their reservation.

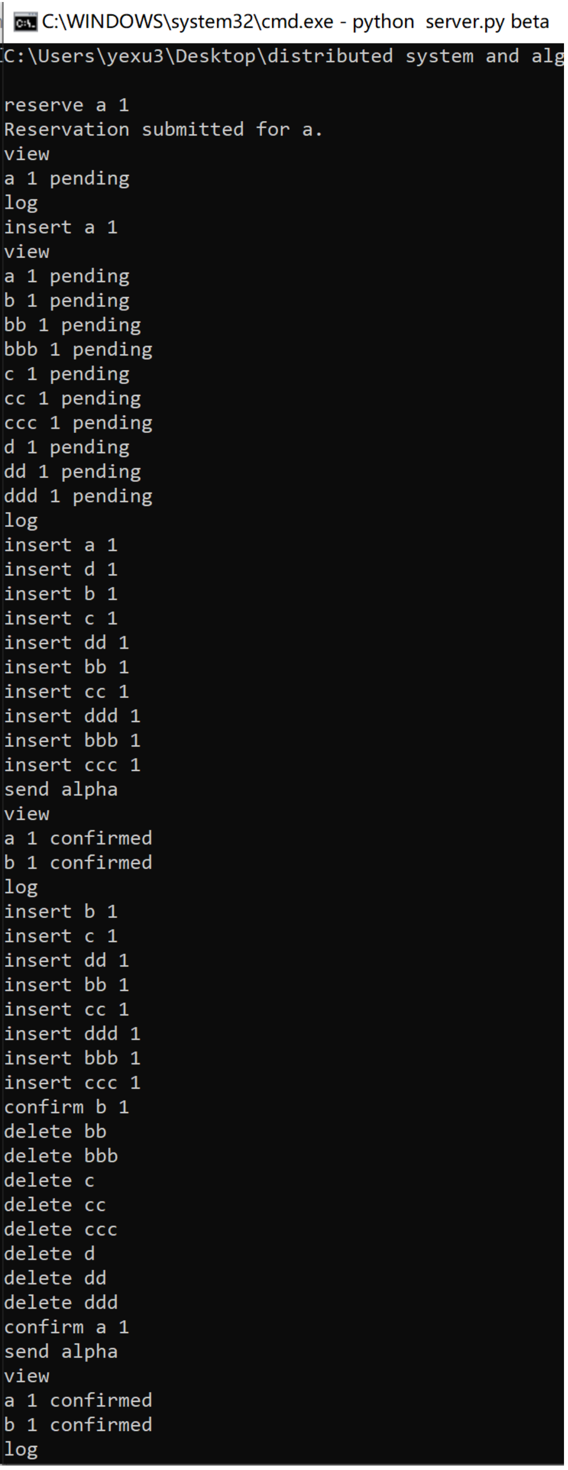
How to test projects(A description of how you tested your project (be sure to consider failures, recovery, and message loss).

1. Corner test: over reserve the seats on one flight by two different process, and keep sending massage between these two process until they get the same dictionary.



C:\WINDOWS\system32\cmd.exe - python server.py alph

```
C:\Users\yexu3\Desktop\distributed system and a

reserve d 1
Reservation submitted for d.
reserve b 1
Reservation submitted for b.
reserve c 1
Reservation submitted for c.
reserve dd 1
Reservation submitted for dd.
reserve bb 1
Reservation submitted for bb.
reserve cc 1
Reservation submitted for cc.
reserve ddd 1
Reservation submitted for ddd.
reserve bbb 1
Reservation submitted for bbb.
reserve ccc 1
Reservation submitted for ccc.
view
b 1 pending
bb 1 pending
bbb 1 pending
c 1 pending
cc 1 pending
ccc 1 pending
d 1 pending
dd 1 pending
ddd 1 pending
log
insert d 1
insert b 1
insert c 1
insert dd 1
insert bb 1
insert cc 1
insert ddd 1
insert bbb 1
insert ccc 1
T
0$$9,0,0,0$$
send beta
view
a 1 pending
b 1 confirmed
log
confirm b 1
delete bb
delete bbb
delete c
delete cc
delete ccc
delete d
delete dd
delete ddd
send beta
view
```

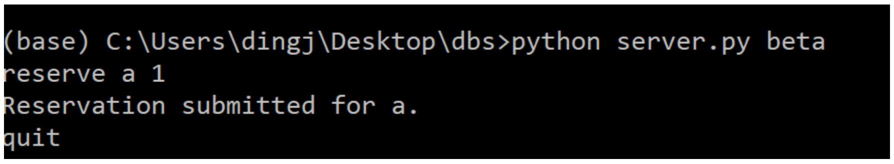


C:\WINDOWS\system32\cmd.exe - python server.py beta

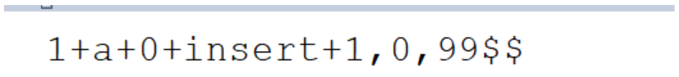
```
C:\Users\yexu3\Desktop\distributed system and alg

reserve a 1
Reservation submitted for a.
view
a 1 pending
log
insert a 1
view
a 1 pending
b 1 pending
bb 1 pending
bbb 1 pending
c 1 pending
cc 1 pending
ccc 1 pending
d 1 pending
dd 1 pending
ddd 1 pending
log
insert a 1
insert d 1
insert b 1
insert c 1
insert dd 1
insert bb 1
insert cc 1
insert ddd 1
insert bbb 1
insert ccc 1
send alpha
view
a 1 confirmed
b 1 confirmed
log
insert b 1
insert c 1
insert dd 1
insert bb 1
insert cc 1
insert ddd 1
insert bbb 1
insert ccc 1
confirm b 1
delete bb
delete bbb
delete c
delete cc
delete ccc
delete d
delete dd
delete ddd
confirm a 1
send alpha
view
a 1 confirmed
b 1 confirmed
log
```

2. Site crashes:

Each time when we have a delete/insert operation, we will save it to the log (operation + event) and we will save the information to a local text file called 'back.txt' , each time when we start a process, the first thing we need to do is check that if there is anything in the back.txt, we will repeat all the operations recorded in the log and we will empty the back file. The following shows when we reserve a and the changes in the backup file

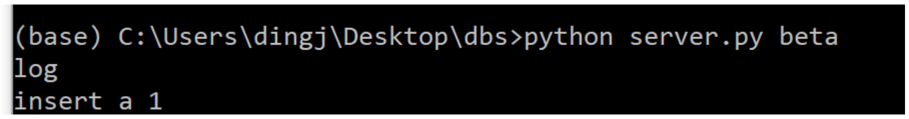


```
(base) C:\Users\dingj\Desktop\dbs>python server.py beta
reserve a 1
Reservation submitted for a.
quit
```



```
1+a+0+insert+1,0,99$$
```

Then we crashes the site using 'quit', we start this site again we can use the log to see the logs are recovered.

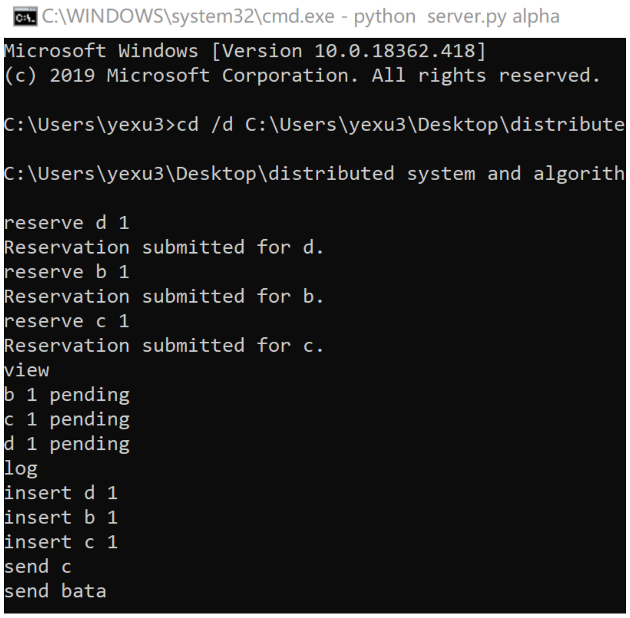


```
(base) C:\Users\dingj\Desktop\dbs>python server.py beta
log
insert a 1
```

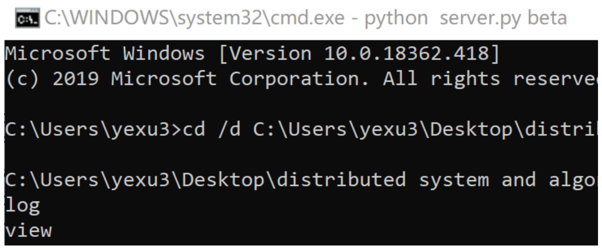
3. message loss:

Design: This algorithm is designed to prevent message loss. If some message loss happens in process alpha to process beta, we have a lot of ways to make it up. First, process beta can get message from other site which has got message from process alpha. Second, if process beta send message to process alpha, process alpha will realize that the process beta lost some message. Third, as long as process beta didn't send message to process alpha, process alpha will keep sending message which include the lost part to process beta, every time process alpha send message to process beta.

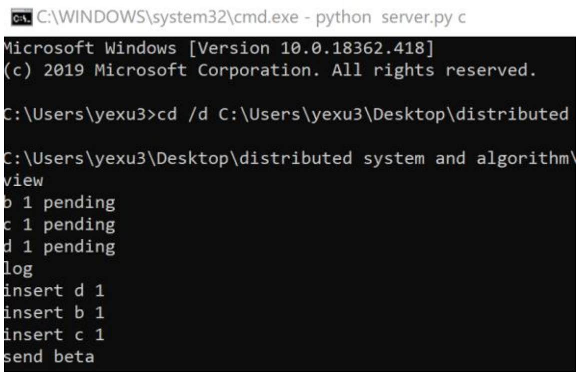
Test: I implement 3 process, and site alpha reserve one event and send it to all the other process. "By mistake" process alpha send message to process "bata" rather than beta. Then by either way, process



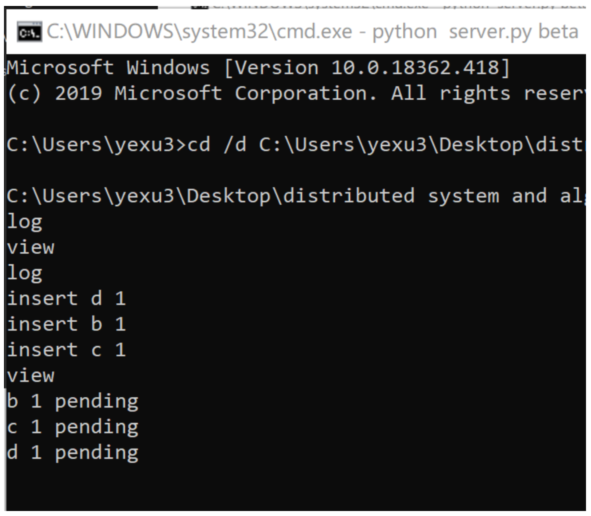
```
C:\WINDOWS\system32\cmd.exe - python  server.py alpha
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\yexu3>cd /d C:\Users\yexu3\Desktop\distribute

C:\Users\yexu3\Desktop\distributed system and algorith

reserve d 1
Reservation submitted for d.
reserve b 1
Reservation submitted for b.
reserve c 1
Reservation submitted for c.
view
b 1 pending
c 1 pending
d 1 pending
log
insert d 1
insert b 1
insert c 1
send c
send bata
```



```
C:\WINDOWS\system32\cmd.exe - python  server.py beta
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reserve

C:\Users\yexu3>cd /d C:\Users\yexu3\Desktop\distri

C:\Users\yexu3\Desktop\distributed system and algo
log
view
```



```
C:\WINDOWS\system32\cmd.exe - python  server.py c
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\yexu3>cd /d C:\Users\yexu3\Desktop\distributed

C:\Users\yexu3\Desktop\distributed system and algorithm\
view
b 1 pending
c 1 pending
d 1 pending
log
insert d 1
insert b 1
insert c 1
send beta
```



```
C:\WINDOWS\system32\cmd.exe - python  server.py beta
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reser

C:\Users\yexu3>cd /d C:\Users\yexu3\Desktop\dist

C:\Users\yexu3\Desktop\distributed system and al
log
view
log
insert d 1
insert b 1
insert c 1
view
b 1 pending
c 1 pending
d 1 pending
```