Report 3: Loggy

Chen, Zidi

September 24, 2020

1 Introduction

This assignment constructs a small system, in which a few workers send messages to each other and a logger tries to print the messages in the right order. The process uses Lamport time stamp to identify the order, and also the vector time stamp is tried.

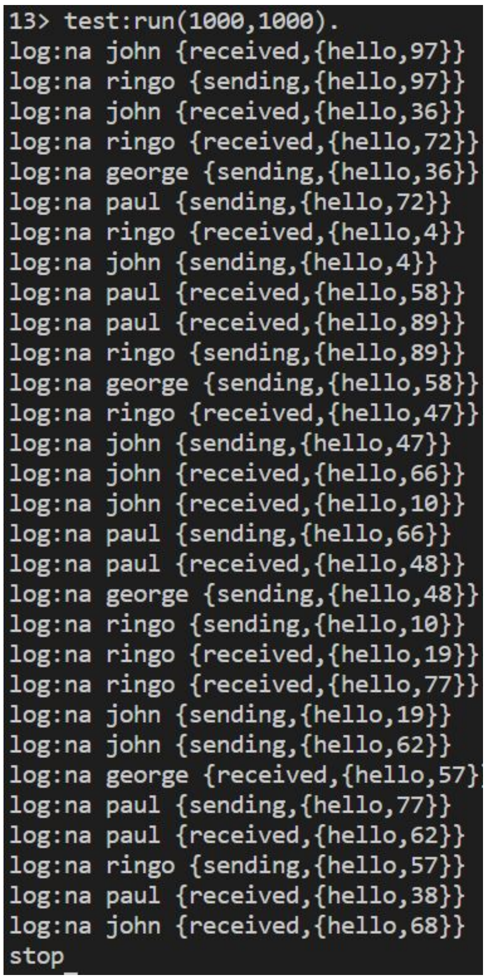
2 Main problems and solutions

When I was constructing the loggy(logger) module, I found the most difficult thing is to figure out how the messages flow. The programming part is not so difficult since there is already skeleton code and erlang pre-defined functions. Then, I followed the small introduction of TA in the help session. It was really helpful. I understood how the message flows in the process after the help section.

3 Evaluation

Here I implemented the process without any clock, the process with Lamport clock and the process with vector clock.

3.1 How it looks like without time stamp



```
13> test:run(1000,1000).
log:na john {received,{hello,97}}
log:na ringo {sending,{hello,97}}
log:na john {received,{hello,36}}
log:na ringo {received,{hello,72}}
log:na george {sending,{hello,36}}
log:na paul {sending,{hello,72}}
log:na ringo {received,{hello,4}}
log:na john {sending,{hello,4}}
log:na paul {received,{hello,58}}
log:na paul {received,{hello,89}}
log:na ringo {sending,{hello,89}}
log:na george {sending,{hello,58}}
log:na ringo {received,{hello,47}}
log:na john {sending,{hello,47}}
log:na john {received,{hello,66}}
log:na john {received,{hello,10}}
log:na paul {sending,{hello,66}}
log:na paul {received,{hello,48}}
log:na george {sending,{hello,48}}
log:na ringo {sending,{hello,10}}
log:na ringo {received,{hello,19}}
log:na ringo {received,{hello,77}}
log:na john {sending,{hello,19}}
log:na john {sending,{hello,62}}
log:na george {received,{hello,57}
log:na paul {sending,{hello,77}}
log:na paul {received,{hello,62}}
log:na ringo {sending,{hello,57}}
log:na paul {received,{hello,38}}
log:na john {received,{hello,68}}
stop_
```

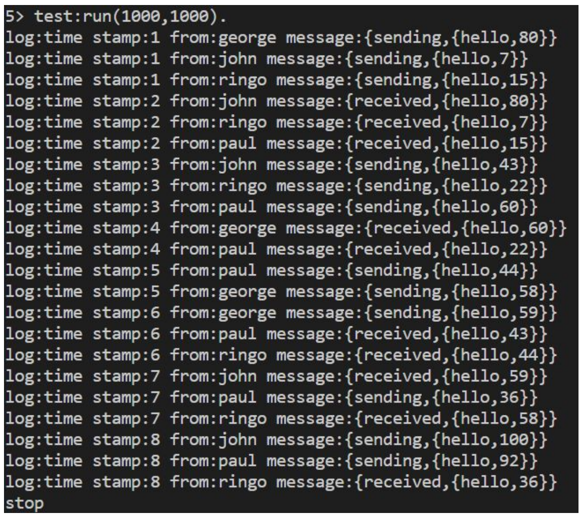
Figure 1: result without time stamp

When the workers sending message without time stamp, there were some errors. The most obvious one is that John received the message before anyone sent the message. This should not be possible.

3.2 Process with Lamport clock

The time module implements the Lamport clock. It can initialize, increase and merge the time stamps of the message which is handled by workers. It can also initialize and update the clock in the loggy module. In addition, it can check if the message is safe to print. Most of the functions are realized with the help of lists in erlang with predefined functions.

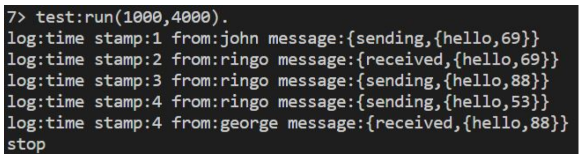
Here is the result of using Lamport clock. Now there is no received message appears before sending message. The time stamps are also in order. But it is still not enough to fully recognize the order because there is message with the same time stamp.



```
5> test:run(1000,1000).
log:time stamp:1 from:george message:{sending,{hello,80}}
log:time stamp:1 from:john message:{sending,{hello,7}}
log:time stamp:1 from:ringo message:{sending,{hello,15}}
log:time stamp:2 from:john message:{received,{hello,80}}
log:time stamp:2 from:ringo message:{received,{hello,7}}
log:time stamp:2 from:paul message:{received,{hello,15}}
log:time stamp:3 from:john message:{sending,{hello,43}}
log:time stamp:3 from:ringo message:{sending,{hello,22}}
log:time stamp:3 from:paul message:{sending,{hello,60}}
log:time stamp:4 from:george message:{received,{hello,60}}
log:time stamp:4 from:paul message:{received,{hello,22}}
log:time stamp:5 from:paul message:{sending,{hello,44}}
log:time stamp:5 from:george message:{sending,{hello,58}}
log:time stamp:6 from:george message:{sending,{hello,59}}
log:time stamp:6 from:paul message:{received,{hello,43}}
log:time stamp:6 from:ringo message:{received,{hello,44}}
log:time stamp:7 from:john message:{received,{hello,59}}
log:time stamp:7 from:paul message:{sending,{hello,36}}
log:time stamp:7 from:ringo message:{received,{hello,58}}
log:time stamp:8 from:john message:{sending,{hello,100}}
log:time stamp:8 from:paul message:{sending,{hello,92}}
log:time stamp:8 from:ringo message:{received,{hello,36}}
stop
```

Figure 2: result with Lamport time stamp

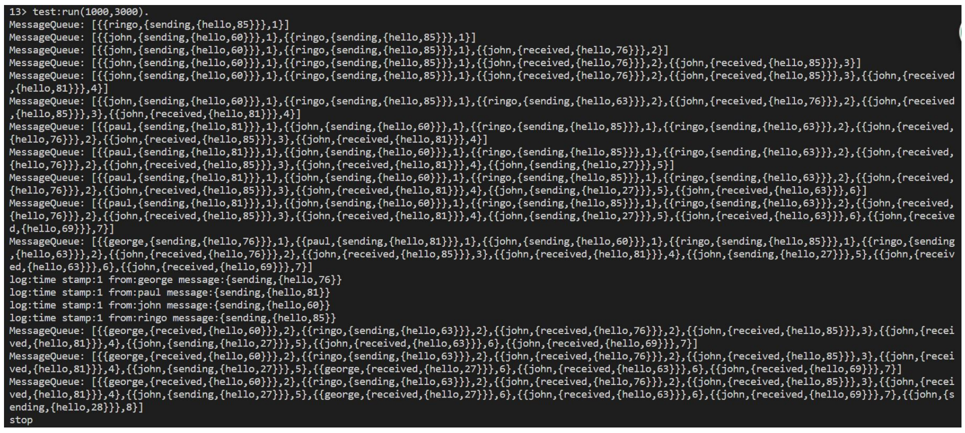
In figure 3, I increased the jitter. The workers were then less active.



```
7> test:run(1000,4000).
log:time stamp:1 from:john message:{sending,{hello,69}}
log:time stamp:2 from:ringo message:{received,{hello,69}}
log:time stamp:3 from:ringo message:{sending,{hello,88}}
log:time stamp:4 from:ringo message:{sending,{hello,53}}
log:time stamp:4 from:george message:{received,{hello,88}}
stop
```

Figure 3: result with Lamport time stamp with bigger jitter

Figure 4 prints the hold back queue in the log. The max queue has 10 entities when sleep is 1000 and jitter is 4000.

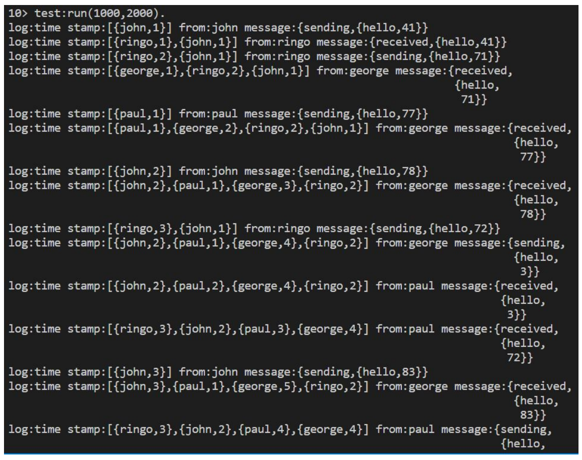


```
13> test:run(1000,3000).
MessageQueue: [{{ringo,{sending,{hello,85}}},1}]
MessageQueue: [{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1}]
MessageQueue: [{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{john,{received,{hello,76}}},2}]
MessageQueue: [{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{john,{received,{hello,76}}},2},{{john,{received,{hello,85}}},3}]
MessageQueue: [{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{john,{received,{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{received
,{hello,81}}},4}]
MessageQueue: [{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{ringo,{sending,{hello,63}}},2},{{john,{received,{hello,76}}},2},{{john,{received
,{hello,85}}},3},{{john,{received,{hello,81}}},4}]
MessageQueue: [{{paul,{sending,{hello,81}}},1},{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{ringo,{sending,{hello,63}}},2},{{john,{received,
{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{received,{hello,81}}},4}]
MessageQueue: [{{paul,{sending,{hello,81}}},1},{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{ringo,{sending,{hello,63}}},2},{{john,{received,
{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{received,{hello,81}}},4},{{john,{sending,{hello,27}}},5}]
MessageQueue: [{{paul,{sending,{hello,81}}},1},{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{ringo,{sending,{hello,63}}},2},{{john,{received,
{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{received,{hello,81}}},4},{{john,{sending,{hello,27}}},5},{{john,{received,{hello,63}}},6}]
MessageQueue: [{{paul,{sending,{hello,81}}},1},{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{ringo,{sending,{hello,63}}},2},{{john,{received,
{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{received,{hello,81}}},4},{{john,{sending,{hello,27}}},5},{{john,{received,{hello,63}}},6},{{john,{receive
d,{hello,69}}},7}]
MessageQueue: [{{george,{sending,{hello,76}}},1},{{paul,{sending,{hello,81}}},1},{{john,{sending,{hello,60}}},1},{{ringo,{sending,{hello,85}}},1},{{ringo,{sending
,{hello,63}}},2},{{john,{received,{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{received,{hello,81}}},4},{{john,{sending,{hello,27}}},5},{{john,{receiv
ed,{hello,63}}},6},{{john,{received,{hello,69}}},7}]
log:time stamp:1 from:george message:{sending,{hello,76}}
log:time stamp:1 from:paul message:{sending,{hello,81}}
log:time stamp:1 from:john message:{sending,{hello,60}}
log:time stamp:1 from:ringo message:{sending,{hello,85}}
MessageQueue: [{{george,{received,{hello,60}}},2},{{ringo,{sending,{hello,63}}},2},{{john,{received,{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{recei
ved,{hello,81}}},4},{{john,{sending,{hello,27}}},5},{{john,{received,{hello,63}}},6},{{john,{received,{hello,69}}},7}]
MessageQueue: [{{george,{received,{hello,60}}},2},{{ringo,{sending,{hello,63}}},2},{{john,{received,{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{recei
ved,{hello,81}}},4},{{john,{sending,{hello,27}}},5},{{george,{received,{hello,27}}},6},{{john,{received,{hello,63}}},6},{{john,{received,{hello,69}}},7}]
MessageQueue: [{{george,{received,{hello,60}}},2},{{ringo,{sending,{hello,63}}},2},{{john,{received,{hello,76}}},2},{{john,{received,{hello,85}}},3},{{john,{recei
ved,{hello,81}}},4},{{john,{sending,{hello,27}}},5},{{george,{received,{hello,27}}},6},{{john,{received,{hello,63}}},6},{{john,{received,{hello,69}}},7},{{john,{s
ending,{hello,28}}},8}]
stop
```

Figure 4: result with Lamport time stamp with queue print

3.3 Process with vector clock

Here is the process with vector clock. There is also no received message which exists before the sending message. I also found that it is clearer about the message order compared with Lamport clock. For example, in the figure, it is clear that john sent a message and george received at the last three lines.



```
10> test:run(1000,2000).
log:time stamp:[{john,1}] from:john message:{sending,{hello,41}}
log:time stamp:[{ringo,1},{john,1}] from:ringo message:{received,{hello,41}}
log:time stamp:[{ringo,2},{john,1}] from:ringo message:{sending,{hello,71}}
log:time stamp:[{george,1},{ringo,2},{john,1}] from:george message:{received,
                                                                    {hello,
                                                                     71}}
log:time stamp:[{paul,1}] from:paul message:{sending,{hello,77}}
log:time stamp:[{paul,1},{george,2},{ringo,2},{john,1}] from:george message:{received,
                                                                             {hello,
                                                                              77}}
log:time stamp:[{john,2}] from:john message:{sending,{hello,78}}
log:time stamp:[{john,2},{paul,1},{george,3},{ringo,2}] from:george message:{received,
                                                                             {hello,
                                                                              78}}
log:time stamp:[{ringo,3},{john,1}] from:ringo message:{sending,{hello,72}}
log:time stamp:[{john,2},{paul,1},{george,4},{ringo,2}] from:george message:{sending,
                                                                             {hello,
                                                                              3}}
log:time stamp:[{john,2},{paul,2},{george,4},{ringo,2}] from:paul message:{received,
                                                                           {hello,
                                                                            3}}
log:time stamp:[{ringo,3},{john,2},{paul,3},{george,4}] from:paul message:{received,
                                                                           {hello,
                                                                            72}}
log:time stamp:[{john,3}] from:john message:{sending,{hello,83}}
log:time stamp:[{john,3},{paul,1},{george,5},{ringo,2}] from:george message:{received,
                                                                             {hello,
                                                                              83}}
log:time stamp:[{ringo,3},{john,2},{paul,4},{george,4}] from:paul message:{sending,
                                                                           {hello,
```

Figure 5: result with vector time stamp