A Tale of Cats and Mouse

Hou, Jue

jue.hou@helsinki.fi

1. Solution

* Mouse keeps an open socket and response to any coming message. For “MEOW”, it will wait 12 seconds then send “OUCH”. For other message, it will response to confirm its existence.
* Chase cat, Jazzy and Catty, will send an arbitrary message to mouse to test if mouse is in the current node. Then they will send the result to Listy. There are three result in all: “F”, “NF” and “G”. Each of them represent for Found, Not Found and Got. The format of the message is as followed:

(result) (ukkonode) (catname)

* Listy keeps an open socket too. Receiving message from chase cat, it will add a timestamp to each message and then write it to the file “cmsg”. The format of message which is actual recorded in the “csmg” file is as followed:

(result) (ukkonode) (catname) (timestamp)

* Cordy will periodically read the “cmsg” file. By keeping a timestamp itself, it can response to the new messages and update its own timestamp. Cordy also keeps a list of ukko nodes and a list of chase cats. There are three state for ukkonodes: “u”, “f”, “s”, representing for unknown , found and under searching. For chase cats, there are three states as well: “p”, “w” and “s”, representing for pending , waiting for double check and searching. According to the new messages, state of cat and the state of nodes, cordy will assgin a cat, double check a node or give a attack command.

1. Executing Process

* Mouse:
  + Read from port\_number and start to listen to the port.
  + Response to a coming message. If it is a “MEOW”, wait 12 seconds and send a “OUCH”. After that, terminate itself. If the message is not a “MEOW”, send a “Mouse Confirmed” message to sender.
* Listy:
  + Read from port\_number and start to listen to the port.
  + After receiving message from either of chase cats, add a timestamp to each message and write it all to the “cmsg” file.
  + If the message is a G message, terminate itself
* Chase Cats (Jazzy amd Catty):
  + Get the name and command from command line interface. Here, Command is refer to searching and attacking.
  + Get the mouse port from port\_number and send a test message to mouse.
  + Sleep 12 seconds and then send a message to test its existence. Or sleep 6 seconds and then send “MEOW” to attack. Both actions will get a result.
  + The result is “NF” by default. If mouse response to search, set the result to “F”. If the mouse send back “OUCH”, which means it has been attacked, set the result to “G”
  + Send the result along with the current node name and cat name to the Listy.
* Cordy:
  + Built a list for cats with their states and a list for ukkonodes with their states.
  + Read the “cmsg” file every 2 seconds. Process the file line by line.
  + Compare the timestamp in each message and its own timestamp, ignore all the messages with a smaller and execute operation for messages which have a larger timestamp. Operations are as followed:
  + For NF message, if no cat is in “w” state, assign the cat mentioned in this message to another arbitrary node. Remove the node mentioned in the message from the ukkonode list. If there is already cat in “w” state. Assign this cat to search the node marked as “f”.
  + For F message, mark the current cat as “w”. Mark the node as “f” as well if it is not in “f” state. If it is already in “f” state, assign “Jazzy” to have an attack.
  + For G message, terminate itself .

1. Instruction

* Environment: python 3
* **Please put every file in the same directory and follow the exact order as listed below to start the scripts.**

1. Start the mouse script by typing “python3 mouse.py” in the command line interface at one of the nodes recorded in the “ukkonodes” file.
2. Start the Listy script y typing “python3 listy.py” in the command line interface at one of the nodes recorded in the “ukkonodes” file as well. Specify the Listy node in the “listy\_location”.
3. Start the Cordy script by typing “python3 cordy.py” at an arbitrary ukko node.