README

Dependencias

Son necesarias la instalación de ciertas dependencias para poder ejecutar el proyecto correctamente, más concretamente son requeridos ***python3*** y ***pip3***

$ sudo apt-get install python3

$ sudo apt-get install python3-pip

Adicionalmente serán utilizados los siguientes módulos de python: ***tabulate*** y ***tkinter***

$ pip3 install tabulate

$ sudo apt-get install python3-tk

Ejecución del proyecto

El proyecto puede ser ejecutado manualmente o mediante el uso de scripts, a continuación procederemos a explicar los dos métodos respectivamente.

Ejecución Manual

Para una correcta ejecución del sistema distribuido, como paso previo, deberemos ejecutar el archivo "***ratf_setup.py***" el cual genera por defecto un conjunto de archivos de configuración para cinco diferentes servidores y tres clientes. En caso de necesitar cambiar estos valores, es posible modificarlos, pasando como parámetros la cantidad de servidores y la cantidad de clientes, en ese respectivo orden.

El sistema asumirá que está compuesto por la cantidad de servidores ingresados anteriormente, independientemente si estos están activos o no.

Comando a ejecutar:

$ *python ratf_setup.py [#server] [#client]*

Una vez hecho esto, y teniendo los archivos de configuración necesarios, proseguimos a ejecutar los servidores (cada uno en una terminal diferente) con el siguiente comando:

$ python server.py configs/server-n.json (donde n es el número de servidor a ejecutar).

Para el ejemplo por defecto de 5 servidores, será necesario ejecutar al menos tres de ellos para que el sistema funcione correctamente, esto es debido a que al ser un sistema de cinco nodos en total, se requieren como mínimo tres votos de diferentes servidores para lograr un consenso.

Para interactuar con los servidores será necesario ejecutar un cliente con el siguiente comando:

$ python cliente-n.json (donde n es el número de cliente a ejecutar).

La ejecución del cliente, abrirá una interfaz gráfica, la cual nos permitirá enviar comandos del tipo SET y GET a los servidores del sistema distribuido.

Ejemplo de ejecución:

1. $ python raft_setup.py
2. $ python server.py configs/server-1.json
3. $ python server.py configs/server-2.json
4. $ python server.py configs/server-3.json
5. $ python server.py configs/server-4.json
6. $ python server.py configs/server-5.json
7. $ python cliente-1.json

Este ejemplo crea un sistema compuesto por 5 servidores (1.) en el cual todos ellos se encuentran ejecutándose (2., 3. ,4. ,5. ,6.), y como paso final ejecuta un cliente (7.)

Ejecución con script

Existen dos scripts, ***run-raft-linux*** (para OS linux) y ***run-raft-windows*** (para OS windows).

- Para el caso de ***run-raft-linux*** *este* acepta dos parámetros de entrada, el primero es la cantidad de servidores por la cual se encuentra compuesto en el sistema, el segundo la cantidad de servidores a ejecutar y el tercer parámetro es la cantidad de clientes a ejecutar. En caso de no ingresar estos parámetros, por defecto el script crea y ejecuta cinco servidores en todo el sistema y dos clientes.

  Comando a ejecutar:

  $ sh run-raft-linux.sh [#max_server] [#server] [#client]

  Ejemplo de ejecución:

  $ sh run-raft-linux.sh 4 3 1

  Este ejemplo crea un sistema compuesto por 4 servidores, en el cual 3 de ellos se encuentran ejecutándose, y finalmente se ejecuta un cliente.

  Nota: el script usa terminales gnome (gnome-terminal) para ejecutar los servidores y clientes.

- En caso de ejecutar **run-raft-windows,** una vez haya ocurrido esto, se le pedirá ingresar por terminal la cantidad de servidores por la cual se compondrá el sistema, paso seguido, la cantidad de servidores a ejecutar y finalmente la cantidad de clientes a ejecutar.

  Comando a ejecutar:

  $ run-raft-windows