Project Three Design Report

1. Introduction of server type

For server type, my design consists of three types of server: *Frontend main server*, *Frontend work server,* and *Apptier work server*:

1.Frontend main server

Frontend server is launched by the given Cloud program with vmid == 1. This server is responsible for dealing with getting requests from load balancer queue, monitor current request interval and adjust the number of *Frontend work server* and *Apptier work server*. Before the running of one *apptier work server*, it will also process requests to reduce timeout.

2. Frontend work server

Frontend work server is launched by *Frontend main server* dynamically and responsible for getting requests from load balancer queue

3. Apptier work server

Apptier work server is launched by *Frontend main server* dynamically and responsible for processing requests submitted by Frontend servers

2. Introduction of the general procedure

Firstly, the Cloud program will launch vmid = 0, 1 as a database server and work server respectively. We use mid =1 as *Frontend main server*.

Secondly, the *frontend main server* will get the first two requests to roughly determine the client interval to initialize the number of *Frontend work servers* and *Apptier work servers*.

Thirdly, frontend main server also acts as a Frontend servers to get requests. After some time, it will check the current average client interval to adjust the number of *Frontend work servers* and *Apptier work servers*.

3. Data Structure

AppQueue is a synchronized queue to store the requests submitted by frontend servers in Frontend main serve. Every apptier servers will connect to *Frontend main server* to poll a request form this queue and process it.

Frontend_machine_List and *Apptier_machine_List* are two lists to store the current existed work servers vmid. These are used to calculate the number of work servers, start and end server and for each work server to find its role.

4. Communication between servers

1.work server get the role

Frontend main server will insert vmid of work server to corresponding machine list when it launches a server to work. Work server will connect to Frontend main server to find out which list its vmid is in and know its role (Apptier or Frontend).

2.transmit and get result

Frontend work server will connect to *Frontend main server* to submit its result to *Appqueue*. *Apptier work* server will connect to *Frontend main server* poll requests from *Appqueue*.

3.scale in and scale out

Use the notify and kill mechanism. *Frontend main server* decide which server should be terminated and notify it. After this server finishes its ongoing job, notify the main server to terminate it immediately, go to sleep and wait for being terminated.

5. Adjust Policy

After many experiments, I divided different according to client interval(i.e., c-xxx-1, xxx in this parameter) and find a table of the optimal number of two types of server corresponding to each client interval. Evey time main server adjust the number of work servers will follow this table to allocate work servers.

6. Cache implementation

The cache is a *ConcurrentHashMap* in my implementation. I override three methods(i.e., get, set, transaction) in Cloud.DatabaseOps in my program. The new get, set, transaction methods are inserted by cache implementation to store the information browsed before. It is worth mentioning that after every transaction, we should update the 'item_qty' to a new one according to the quantity user buy.

04/08/2020
Zihao ZHOU