Performance Evaluation Report

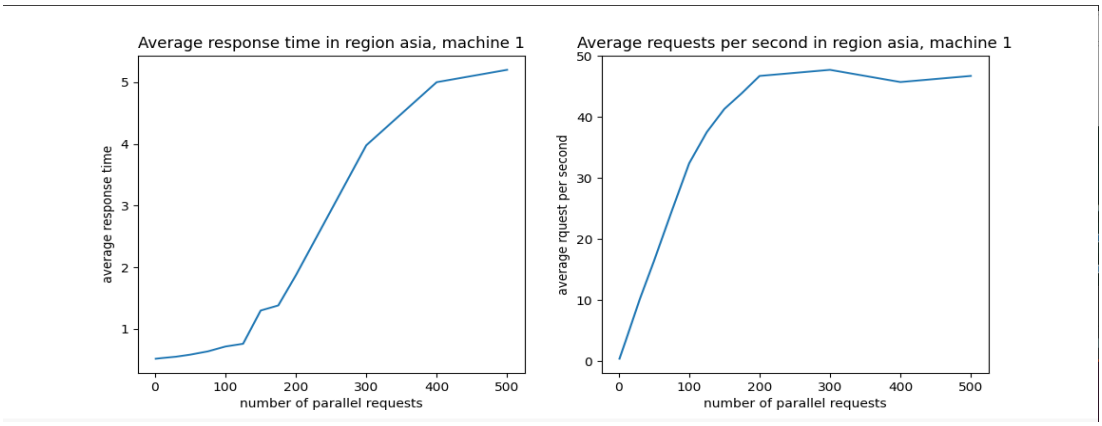
Caroi Narcis-Florin
343C2
Parts Completed:

- Prerequisites
- Implementation
- Evaluation
- Documentation

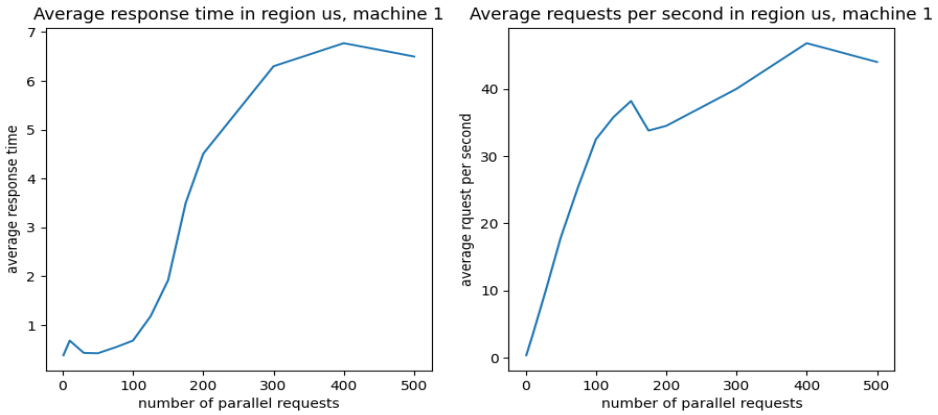
A. System Limits Analysis

How many requests can be handled by a single machine?

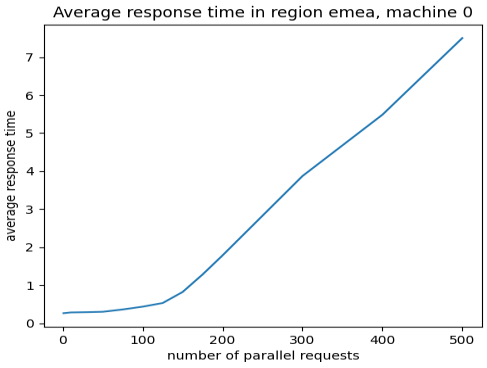
- asia – I tested on worker 1. Request that can be handled on this machine is between 400-500. When I tested with 500 requests I got the error message: RemoteDisconnected('Remote end closed connection without response')

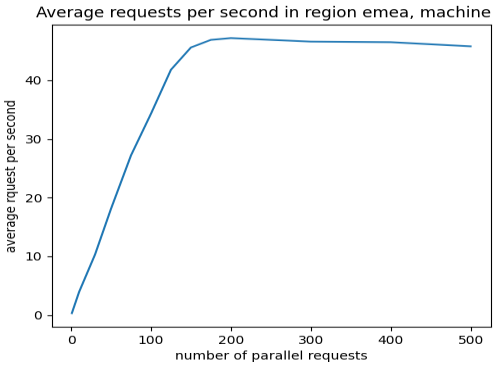


- us – I tested on worker 1. Request that can be handled on this machine is between 400-500. When I tested with 500 requests I got the error message: RemoteDisconnected('Remote end closed connection without response')



- emea - I tested on worker 0. Request that can be handled on this machine is between 400-500. When I tested with 500 requests I got the error message: RemoteDisconnected('Remote end closed connection without response')





P.S: I got the results with locust, that is a performance testing tool

What is the latency of each region?

I tested it with locust with 1 user, and I got the next results:

- us – average 380 ms
- asia – average 450 ms
- emea – average 263 ms

What is the computation time for a work request?

Using curl command, I got average working time for all of them (us,asia,emea) the same time 20ms.

What is the the response time of a worker when there is no load?

Using locust with one user, I got average response time for us 380ms, asia 520 ms and emea 263 ms.

What is the latency introduced by the forwarding unit?

Using locust directly on Heroku app I got the next average results: asia1 65ms , us1 70ms, emea 58ms. So the latency introduced by the forwarding unit is between 200-500ms.

How many requests must be given in order for the forwarding unit to become the bottleneck of the system? How would you solve this issue?

When I was testing with Locust, I noticed that when I send between 400-500 parallel request I got an error message RemoteDisconnected('Remote end closed connection without response') and since I sent between 300-400 parallel requests I almost got to the bottleneck of 40-45 requests per second. To solve this I would change the forwarding unit with another that is better, or I would introduced more forwarding units with load balancer.

What is your estimation regarding the latency introduced by Heroku?

Using locust directly on Heroku app I got the next average results: asia1-65ms , us1-70ms, emea 65ms.

What downsides do you see in the current architecture design?

I think the most disadvantage fact is the forwarding unit that must be replaced with a better one or should be added more forwarding units.

B. Load Balancing Policies Comparison

Random Policy

The requests are distributed random to each worker.

Round Robin

The requests are distributed to each worker in order. First request to the first worker so on until the fifth request to the fifth worker, again and again until the requests are finished.

Weighted Round Robin

Almost the same idea as round robin. Unlike Round Robin, I saw that emea is the fastest, then us and then asia the slowest. So, I chose that emea should get 3 request, us 2 request per worker with total of 4, and asia 1 request per worker with total of 2.

Least Connection

It should be something like, see the worker that got the least request and send requests to it, but it works almost like Round Robin

Fastest Time

Found the fastest worker and all requests are sent to it.