Final report on the Management & Content Delivery for Smart Networks: Algorithms & Modelling laboratories

Luca Gioacchini (s257076), Riccardo Sappa (s263241),
Stefano Picerno (s265406)

A.A.2018/19

Contents

1 Simulation on Content Delivery Network

A Content Delivery Network (CDN) is a geographically distributed network of proxy servers, which speed up the content distribution between the internet users by storing a local replica of accessible objects in different servers. The CDN design is a hard task, since different factor must be taken into account, such as the servers' maintaining cost or the clients serving policy.

In order to evaluate a CDN performances, different tools can be used, for example the discrete-event simulation framework SimPy. In this section the results of two CDN design choices are provided: a *static* servers one and a *dynamic* one.

1.1 CDN Design Choices

The static servers CDN is characterized by a starting number of always-on servers in five countries (China, Brazil, Japan, USA and India). If a client finds all the servers busy, a new one is deployed by using three criteria:

- The *distance* based one, with which the server is placed in the client hosting country;
- The *cost* based one, with which the server is placed in the country providing the minimum maintaining cost;
- The *people* based one, with which the server is placed in the country identified with the maximum number of inhabitants. The countries are roundly chosen until all of them don't have the same number of servers.

The dynamic servers CDN is characterized by a fixed number of server in the same countries of the static case. At the beginning, a number of servers is active, whereas the others are in idle. When the server's available capacity exceed a threshold, it can wake up an idling device, whereas, if the capacity decreases under a second threshold and then is increased over a third one, the server stops receiving new requests, finishes the pending ones and is set in idle(a minimum number of devices is always-on). The waking up process follows three different policies:

- The *distance* based one, with which a server is woken up in the same country of the saturated one. If all the local devices are active, the new server is chosen in the client hosting country;
- The *cost* based one, with which the woken up server belongs to the country providing the minimum maintaining cost;
- The *people* based one, with which the server is placed in the country identified with the maximum number of inhabitants. The countries are roundly chosen until all of them doesn't have the same number of active servers.

The clients' arrival is described by a Poisson process:

$$P(x) = e^{-\lambda}\frac{\lambda^x}{x!} \tag{1}$$

where λ is the arrival rate, determined as the 0.1% of the daily number of internet users for each country by taking into account the relative time zones. In this way, the generated traffic showed in Figure 1 follows a day-night pattern so the day can be divided into a high traffic hours range and a low traffic one.

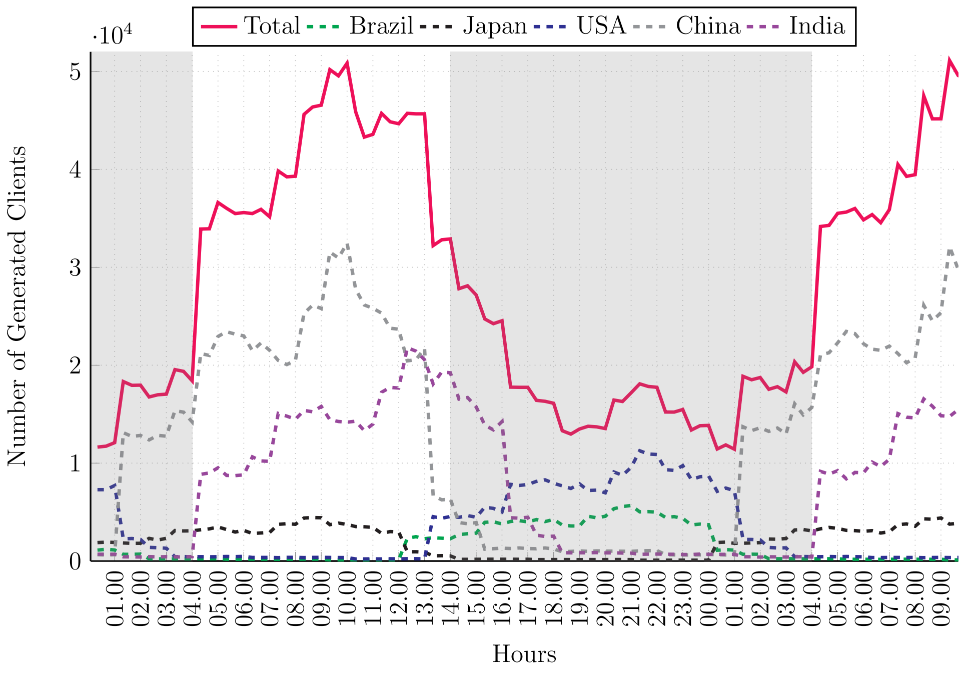


Figure 1: Country specific and total generated traffic trend. The highlighted areas are the low traffic hours ranges, the white areas are the high traffic ones.

1.2 Servers

In order to reduce the CDN maintaining cost, the servers dispose of a dynamic double threshold system: one for the low traffic hours, and one for the high traffic ones. According to Figure 2, when the traffic is low, a new server is woken up if the available capacity of a device goes under the 30% of the total one. Moreover, if the capacity goes under the 93% of the total one, the server might be in idle and from now on, it stops accepting new requests if the capacity increases by reaching the 95% of the total.

On the other hand, during the high traffic period, the minimum capacity needed to wake up a server is the 50% of the total. This value is the same one which can send a server in idle and the 95% of the total capacity remains the maximum threshold beyond which the device stops receiving new requests.

The server's maintaining costs per hour are modeled on the Amazon CloudFront ones and the server's capacity is fixed at $10Gb$.

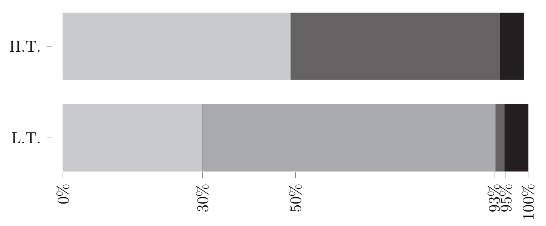


Figure 2: Low and High traffic thresholds in terms of server's capacity percentage

1.3 Clients

Regarding the clients, they are characterized by a random number of requests uniformly distributed between 10 and 100. Although the size of a CDN requests belongs in the $[400KB, 4MB]$ range, in order to reduce the computational power and time, it has been set in the $[4MB, 40MB]$ range. This is a good compromise for the small amount of people joining the CDN.

When a client enters the system, it sends its requests in succession until all of them are not served. The time needed to finish its service is called *service time* and it is one of the performance evaluation parameters.

The serving policy is based on the distance: a client checks for locally active servers, if they are all busy, it searches for an active server in the nearest country and so on. In the dynamic case, the client continues its research until a free server is not found.

The serving time for a single requests depends on three factors:

- A small server latency: a discrete random variable uniformly distributed between 1 ms and 10 ms;
- A round-trip-time (RTT) from the client to the server equal to the time the light takes to travel from the capital city of the country where the client is located to the city where server is hosted;
- A transfer delay, which is determined by the size of the response divided by the capacity allocated to the request in the server.

1.4 Experimental Results

The CDN simulation has been performed by considering 24 hours of service (from 3:00 am to 3:00 am), with 3 additional hours of warm up (from 12:00 am to 3:00 am) in order to make the simulator operative. The performance evaluating parameters are extracted every 20 simulated minutes and the values obtained during the warm up are discarded.

In the static case, the starting number of servers is initialized as one server per country, whereas in the dynamic case each country has 15 servers. The minimum number of active devices is 2 in Brazil, USA and Japan, 4 in India and 5 in China. At the simulation beginning 6 devices are active in China, 4 in India, 2 in USA, Brazil and Japan.

According to Figure 3, in the static case the number of server grows linearly when passing from the low traffic zone, to the high traffic one. Once the optimal number of devices has been found, it remains constant, meaning that all the clients are successfully served. Regarding the deployment policy, the distance based one seems to be the best one with respect to the number of servers, and consequently to the maintaining cost. However, even if the number of servers stops growing, the percentage of the locally served requests has a decrease from the 6:00 pm (from the 90% to the 82-83%), whereas the other two deploying policies lead to a similar trend except for an increase from the 50%, up to the 77-78%. In terms of average session time, this leads to a dual behavior: the cost based and the people based policy are characterized with very high times (from 14s to 30s) in the first part of the day and a decrease down to 2s in the last part of the day. On the other hand, the distance based case leads to lower session times in the first part of the day and higher ones (from 18s to 24s) in the last part, coherently with the percentage of local served requests decrease.

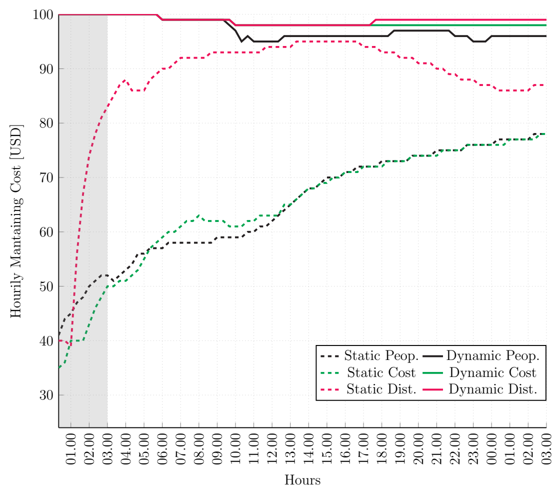
Regarding the dynamic design strategy, the number of active servers clearly reflects the traffic patterns: when it transits from low to high, more servers become active, whereas they are set in idle when the traffic decreases. The maintaining costs has quite the same trend of the number of devices and the cheapest wake up strategy seems to be the people based one, whereas the cost based and distance based policies follow quite the same pattern during the high traffic period. However, in the low traffic one, the cost based strategy leads better performances than the other two. Thanks to the on/off transition of the servers, the percentage of active requests stands around the 95-100%. This leads to quite acceptable average session times around 2s with all the three strategies, even if the distance based one has a 10s peak in the rush hours and the other two have 12 seconds peaks from the 9:00 pm to the 12:00 am.

	Static			Dynamic		
	Distance	Cost	People	Distance	Cost	People
Active Servers	24	36	36	40	37	31
Daily Cost [USD]	1.75	2.62	2.60	2.94	2.72	2.39
Session Time [s]	9.32	7.76	8.52	1.59	1.73	2.68
Local Requests [%]	90.92	68.59	68.08	98.77	98.32	96.83

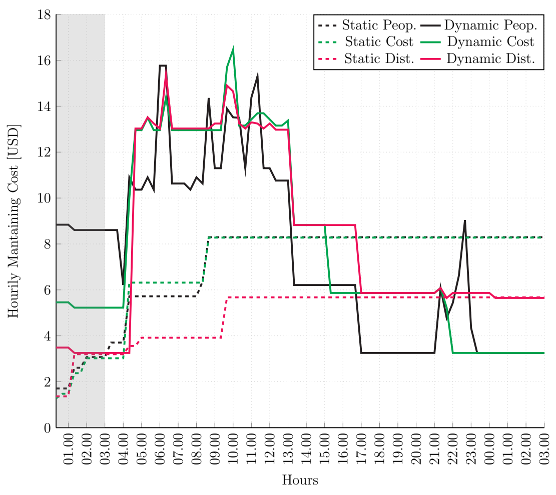
Table 1: Daily average parameters

By comparing the two design choices, the low number of active servers and the hourily maintaining costs during the low traffic period obtained in the static cases is balanced in the high traffic hours by the dynamic ones. Only on the basis of the average daily cost shown in Table 1 the first strategy seems to be the best one. In fact, thanks to the dynamic server allocation, the greater number of active devices allows to obtain an acceptable average daily session time (1.59s and 1.73s are the best cases), whereas, by considering that in the first design strategy a new server is deployed only if all the others are saturated and that the transfer delay depends on the available capacity, a fixed number of always on servers leads to unacceptable service time (7.76s in the best case and 9.32s in the worst case). This behaviour is evidenced also by the percentage of locally served requests (the RTT depends on the client-server distance).

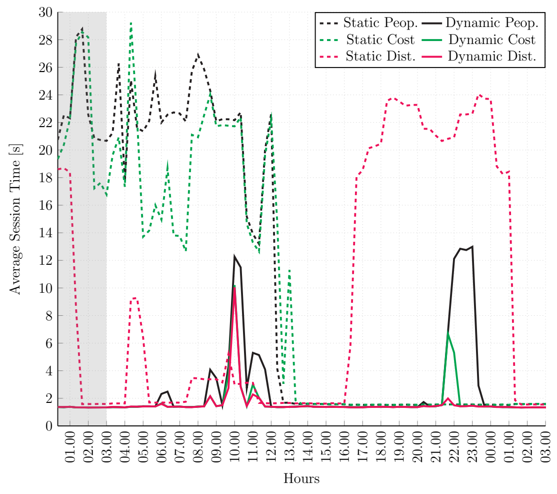
According to what has been said, depending on the deploying requirements, if the session time is negligible, the static servers strategy is the best one with respect to the hourily maintaining costs, especially if the new servers are placed with respect to the client hosting country. On the other hand, if the cost limitation are not tight, the



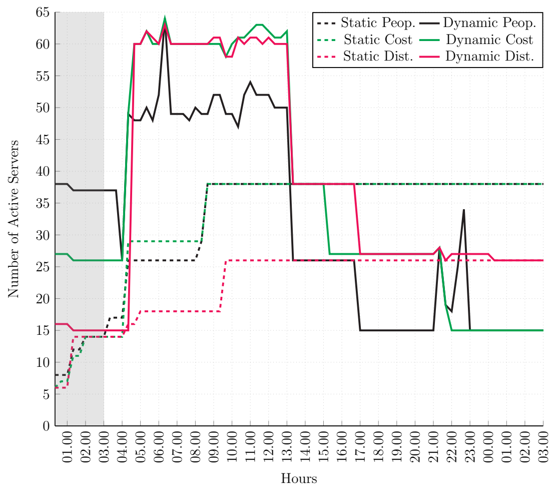
(a) Percentage of locally served requests



(b) Hourily mantaining cost trend



(c) Average session time



(d) Number of active servers

Figure 3: Performance evaluation parameters. The highlighted area is the simulation warm up. 24 simulated hours of service

dynamic server allocation is the best designing strategy, together with the people based wake up policy.

Finally, the cost based dynamic case can be considered as the best compromise between the main two performances evaluating parameters with a daily average maintaining cost of 2.72 USD and an average session time of 1.73s, in consequence of an average 98.32% of locally served requests.