6.033 DP

lopezv

March 2016

1 Evaluation

1.1 Connection Time

Our group was able to determine that Happynet would be able to have a connection time of approximately 400 ms in the average use case. This time exceeds our acceptable margin of under two seconds and should be quick enough so that clients are unable to notice any significant overhead. We were able to compute this connection time by considering the worst case delays in the connection process. Recall that Happynet first communicates with all nearby APs when attempting to connect, and that the delay in connection comes from waiting for packets to be transferred from the clients to the Aps. Thus the time to connect (TTC) can be given be the following formula:

$$TTC = Channels \cdot (HeartbeatTime + ResponseTime) \tag{1}$$

1.2 Performance in Common Use Cases

1.2.1 Single User in Underutilized Area

This use case is a very common, and Happynet is able to satisfy clients in underutilized areas. These areas can be found all over campus in some of the more secluded spaces on campus such as certain study areas and offices. Hence since there is not much threat of overloading the system in areas like this, our primary concern would be Happynet's performance.

TODO: Insert image of scenario 1

1.2.2 Many User in a Single Area

Whenever a group of users aggregate to a single area such as a classroom or public dining space, Happynet must be able to attempt to optimally connect and distribute users over the available APs.

TODO: Insert image of scenario 2

1.2.3 High Churn

Since one of Happynet's criteria for connecting to an AP is the AP's available throughput it is imperative that AP's always have an accurate measure of both the desired throughput of its clients and the amount of clients connected to it. Our system handles this by considering clients who have failed to send a packet within the past

TODO: Insert image of scenario 3

1.3 CacheTTL

When clients receive location suggestions, Happynet must be sure to be able to provide as accurate information as possible to clients. Otherwise users may end up walking up to 500ft to attempt to connect to an AP when there is not any available to support to them in the recommend location. However, the system should also be able to off load these recommendations to the AP so that the server does not get overloaded with location requests and so that APs can provide service while the server is offline. Therefore it is important that our cache TTL accommodates for both these issues.

1.4 Memory

Given that APs are nearly continuously collecting and sending data for storage to the server, Happynet must be able to handle the memory constraints imposed by the hardware that we are using.

1.4.1 Server Database Size

The central database stores throughput and client count information for every AP in the system. This has the potential to grow very rapidly since we would like to compute information for every AP for approximately every second. Our current estimations hold that the current hard drive provided by IS&T would be able to contain up to 10 years of storage.

TODO: Include calculations

1.5 Potential Bottlenecks

1.5.1 Data Gathering

If the number of APs were to grow considerably (e.g. if MIT were to expand or Happynet were to be implement on a much larger scale), data gathering can stand to become a very limiting bottleneck as APs send their data directly to the central server. The most effective solution were this to be the case would be to implement a distributed network of servers which maintain the data sent by APs.

1.5.2 AP Density

Since current specifications only allow for up to 11 APs to be available in close proximity, should any particular area become incredibly popular our system would be unable to handle just a dense amount of clients. This is also exasperated by the fact that the number of client devices per user is increasing as well with people carrying multiple devices such as their laptops, smart phones, tablets, etc. Unfortunately, our system is unable to scale to solve this issue due to technological constraints.

1.5.3 Load Balancing

When load balancing Happynet optimizes for available throughput in the network, however since users take a non-instant amount of time to relocate to their suggested AP, it is possible that multiple APs recommend the same location and since they do not communicate with each other about their connection.

Another issue that is possible when it comes to load balancing is ensuring that loads are distributed evenly amongst the APs,

1.5.4 Expected Unhappiness

Our design currently assumes that users will be unhappy at most 6-8 times per hour. However, should user increase their unhappiness dramatically, Happynet would be in a similar issue when it comes to load balancing like above.