Project 2
DNS Simulation

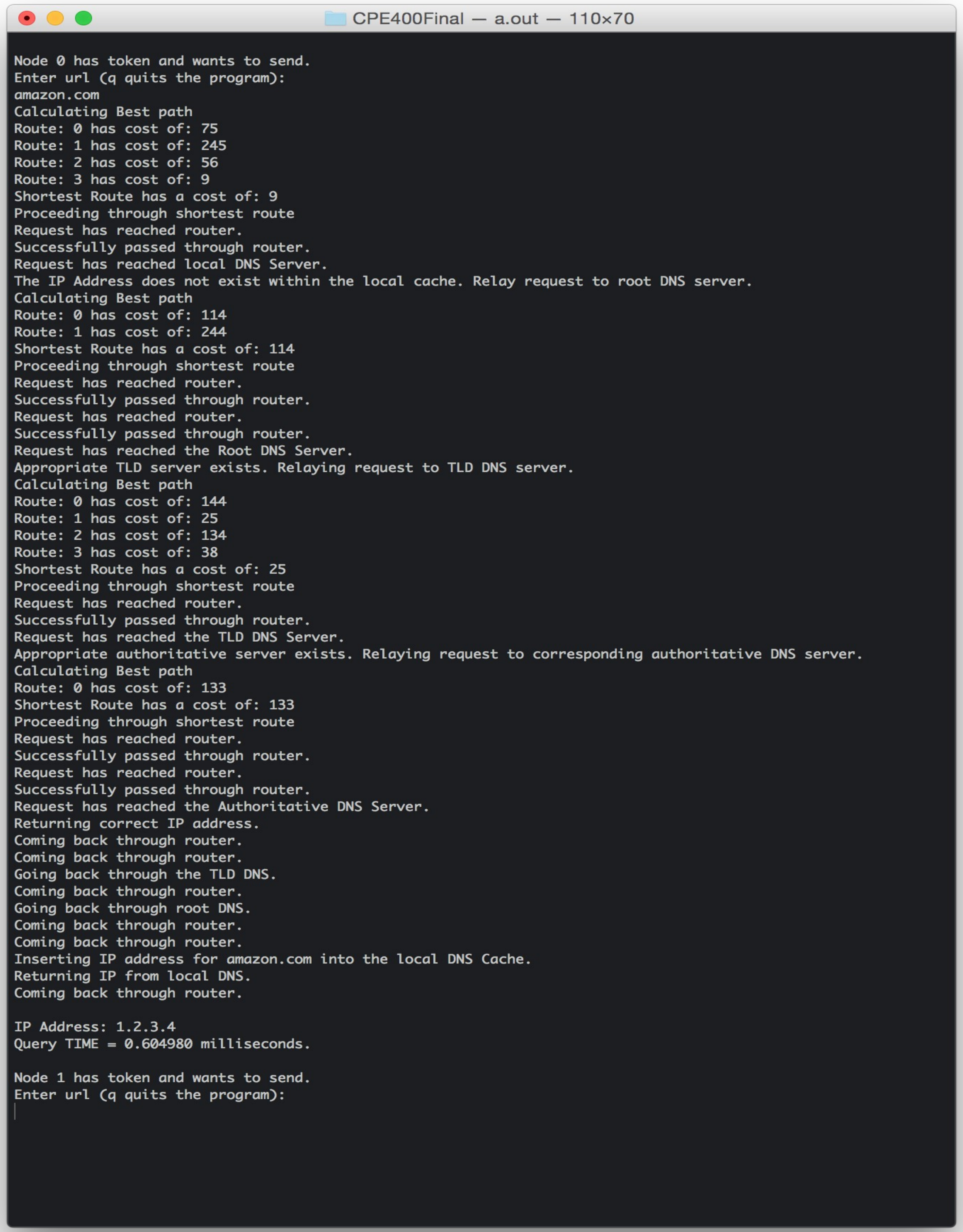
CPE 400

Renee Iinuma
Wesley Kepke
Ernest Landrito
Kyle Lee

Dr. Shamik Sengupta
December 8, 2014

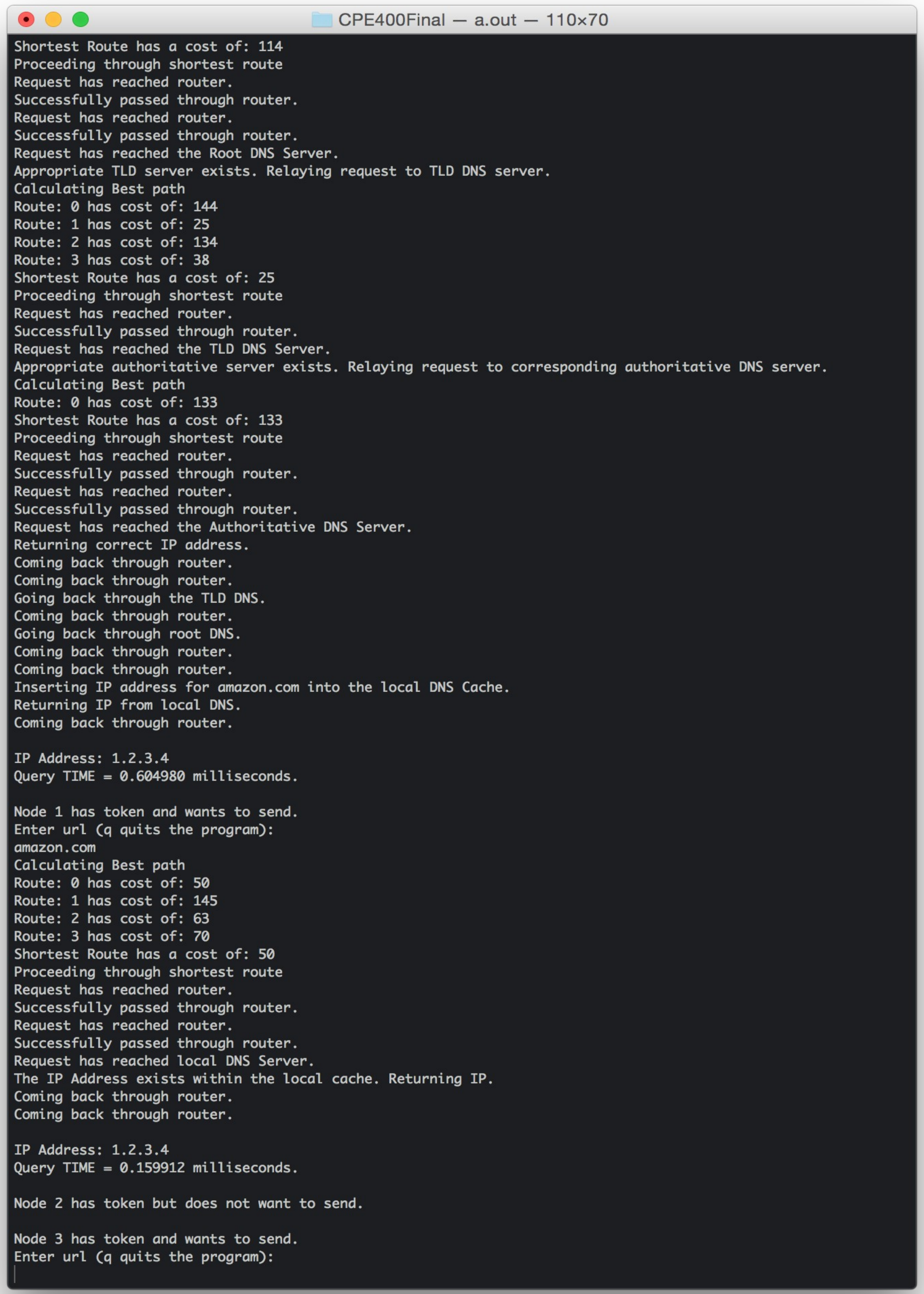
NOTE: The code for this project was written in C++ and compiles under g++. The executable is named a.out.

Screen Shots:



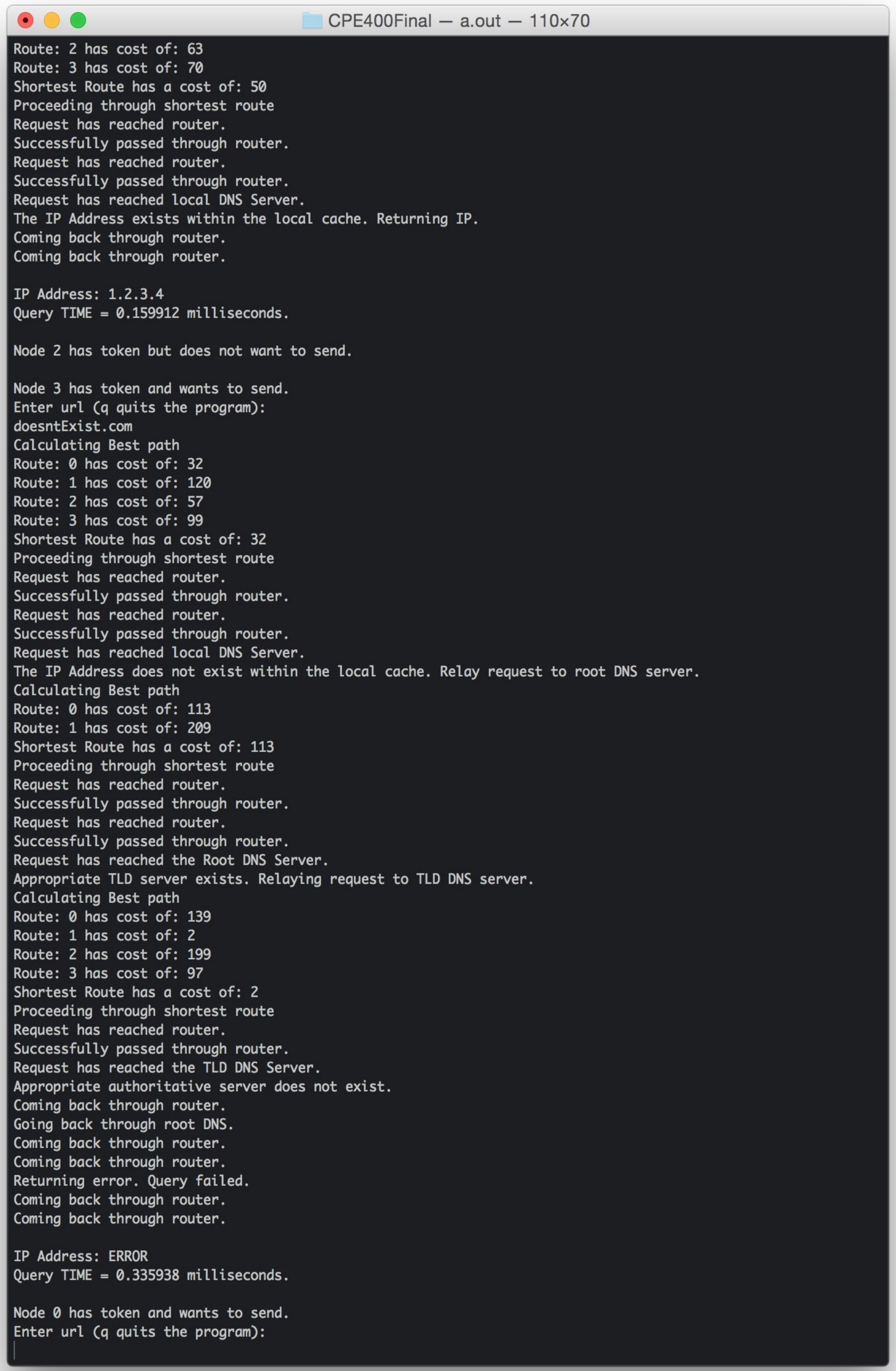
```
Node 0 has token and wants to send.
Enter url (q quits the program):
amazon.com
Calculating Best path
Route: 0 has cost of: 75
Route: 1 has cost of: 245
Route: 2 has cost of: 56
Route: 3 has cost of: 9
Shortest Route has a cost of: 9
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached local DNS Server.
The IP Address does not exist within the local cache. Relay request to root DNS server.
Calculating Best path
Route: 0 has cost of: 114
Route: 1 has cost of: 244
Shortest Route has a cost of: 114
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached router.
Successfully passed through router.
Request has reached the Root DNS Server.
Appropriate TLD server exists. Relaying request to TLD DNS server.
Calculating Best path
Route: 0 has cost of: 144
Route: 1 has cost of: 25
Route: 2 has cost of: 134
Route: 3 has cost of: 38
Shortest Route has a cost of: 25
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached the TLD DNS Server.
Appropriate authoritative server exists. Relaying request to corresponding authoritative DNS server.
Calculating Best path
Route: 0 has cost of: 133
Shortest Route has a cost of: 133
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached router.
Successfully passed through router.
Request has reached the Authoritative DNS Server.
Returning correct IP address.
Coming back through router.
Coming back through router.
Going back through the TLD DNS.
Coming back through router.
Going back through root DNS.
Coming back through router.
Coming back through router.
Inserting IP address for amazon.com into the local DNS Cache.
Returning IP from local DNS.
Coming back through router.

IP Address: 1.2.3.4
Query TIME = 0.604980 milliseconds.

Node 1 has token and wants to send.
Enter url (q quits the program):
```

Fig. 1: This screenshot demonstrates the result of querying amazon.com, and iterates through the entire DNS process because amazon.com has not yet been cached in the local DNS server. The IP address is returned, as seen in the screenshot. Furthermore, note that the time has been output and can be used to compare against other DNS queries, particularly when the query is cached in the local DNS server. Finally, please note that the token has now moved to node 1; therefore node 1 can now submit a request.



```
Shortest Route has a cost of: 114
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached router.
Successfully passed through router.
Request has reached the Root DNS Server.
Appropriate TLD server exists. Relaying request to TLD DNS server.
Calculating Best path
Route: 0 has cost of: 144
Route: 1 has cost of: 25
Route: 2 has cost of: 134
Route: 3 has cost of: 38
Shortest Route has a cost of: 25
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached the TLD DNS Server.
Appropriate authoritative server exists. Relaying request to corresponding authoritative DNS server.
Calculating Best path
Route: 0 has cost of: 133
Shortest Route has a cost of: 133
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached router.
Successfully passed through router.
Request has reached the Authoritative DNS Server.
Returning correct IP address.
Coming back through router.
Coming back through router.
Going back through the TLD DNS.
Coming back through router.
Going back through root DNS.
Coming back through router.
Coming back through router.
Inserting IP address for amazon.com into the local DNS Cache.
Returning IP from local DNS.
Coming back through router.

IP Address: 1.2.3.4
Query TIME = 0.604980 milliseconds.

Node 1 has token and wants to send.
Enter url (q quits the program):
amazon.com
Calculating Best path
Route: 0 has cost of: 50
Route: 1 has cost of: 145
Route: 2 has cost of: 63
Route: 3 has cost of: 70
Shortest Route has a cost of: 50
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached router.
Successfully passed through router.
Request has reached local DNS Server.
The IP Address exists within the local cache. Returning IP.
Coming back through router.
Coming back through router.

IP Address: 1.2.3.4
Query TIME = 0.159912 milliseconds.

Node 2 has token but does not want to send.

Node 3 has token and wants to send.
Enter url (q quits the program):
```

Fig. 2: This screenshot demonstrates what happens when a node sends a query for amazon.com. Since node 0 had already requested for amazon.com, the IP address is now stored in the local DNS cache. As a result, node 1's query need only reach the local DNS to retrieve the IP address. Since this data has been cached and the DNS process is much shorter that the initial request (node 0's request), we observe a time that is considerably smaller than node 0's request for amazon.com. Finally, please note that node 2 is given the token next but has no data to transmit, so the token then passes to node 3 who has data to transmit, which can now safely occur because node 3, and no other nodes, now has the token.



```
Route: 2 has cost of: 63
Route: 3 has cost of: 70
Shortest Route has a cost of: 50
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached router.
Successfully passed through router.
Request has reached local DNS Server.
The IP Address exists within the local cache. Returning IP.
Coming back through router.
Coming back through router.

IP Address: 1.2.3.4
Query TIME = 0.159912 milliseconds.

Node 2 has token but does not want to send.

Node 3 has token and wants to send.
Enter url (q quits the program):
doesntExist.com
Calculating Best path
Route: 0 has cost of: 32
Route: 1 has cost of: 120
Route: 2 has cost of: 57
Route: 3 has cost of: 99
Shortest Route has a cost of: 32
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached router.
Successfully passed through router.
Request has reached local DNS Server.
The IP Address does not exist within the local cache. Relay request to root DNS server.
Calculating Best path
Route: 0 has cost of: 113
Route: 1 has cost of: 209
Shortest Route has a cost of: 113
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached router.
Successfully passed through router.
Request has reached the Root DNS Server.
Appropriate TLD server exists. Relaying request to TLD DNS server.
Calculating Best path
Route: 0 has cost of: 139
Route: 1 has cost of: 2
Route: 2 has cost of: 199
Route: 3 has cost of: 97
Shortest Route has a cost of: 2
Proceeding through shortest route
Request has reached router.
Successfully passed through router.
Request has reached the TLD DNS Server.
Appropriate authoritative server does not exist.
Coming back through router.
Going back through root DNS.
Coming back through router.
Coming back through router.
Returning error. Query failed.
Coming back through router.
Coming back through router.

IP Address: ERROR
Query TIME = 0.335938 milliseconds.

Node 0 has token and wants to send.
Enter url (q quits the program):
```

Fig. 3: This screenshot demonstrates what happens when an invalid hostname (doesntExist.com) is queried from the client. An error is returned, thus terminating the DNS query. If the query contains an invalid website suffix, the query will encounter an error when the root tries to determine which TLD server to go to next. However, if an invalid or non-existent website is queried, then the process will terminate upon selection of an authoritative DNS server. Since an error occurred, the token is then passed to the next node. Please also note that even though the DNS query terminated, the time is still output to the console.

Network Layer Simulation:

1. Application Layer:

Because we focused on the behavior of DNS, the focal point of our simulation is on the application layer. To simulate the application layer, we begin at the client (there are multiple clients – more on this later) and traverse through DNS servers to find the IP address of the requested website. Specifically, in the local DNS server, it will check the local DNS server's cache for the IP address. If found, this IP address is returned and the DNS process discontinues. Otherwise, if that IP address is not found in local cache, then the request will traverse though the root, TLD, and authoritative DNS servers. To simulate this process, we created objects that model how DNS would handle an IP request. These objects include a router that would simply pass the IP request along to another device, a local DNS server that would query its cache or query the root, a root DNS server which would determine which TLD DNS server to query, a TLD DNS server which would determine which authoritative DNS server to query, and an authoritative DNS server which would return the corresponding IP address if valid. Our implementation follows the recursive structure of a DNS query; therefore, the process recursively returns the IP address through each of the DNS servers until the result is stored in the local DNS cache.

2. Network Layer:

Due to the structure of our application layer, the reader should be aware that we have multiple clients connected to a local DNS server, a local DNS server that connects to a root DNS server, a root DNS server that is connected to several TLD DNS servers, each of which is then connected to a plethora of authoritative DNS servers. However, in order to simulate a more realistic network layer, we have decided to include multiple routes (of routers) among the previously mentioned DNS servers. The reason why we decided to incorporate this addition is because we wanted to force the DNS server to determine the least-cost route that it would need (and want) to take to reach the next destination in the DNS query process. The addition of this decision is not only important but feasible because the user wants to load their requested web-page as soon as possible. By implementing a route selection protocol that provides the most efficient DNS query possible (in regards to our simulation), we not only provide functionality to the network layer component of this project, but attempt to model a real-world implementation that is synonymous with the DNS protocol.

3. Link Layer:

To simulate the network layer, we created a simple network of routers to interconnect the DNS servers. Additionally, we chose to implement the token ring system. A token ring system uses a “token” that is only given to a node when that node is allotted a chance to submit a DNS query. We are aware that a token ring network has particular disadvantages, such as a a single point of failure and inefficiency in regards to the maintenance of the token. However, since we did not cover this type of MAC protocol extensively in lecture, we decided that this would be a sufficient addition to the three-layer requirement of this project. Our simulation includes four nodes, each of which is passed over repeatedly in a circular fashion until the user terminates the process. **The DNS simulation is terminated by querying 'q'.**