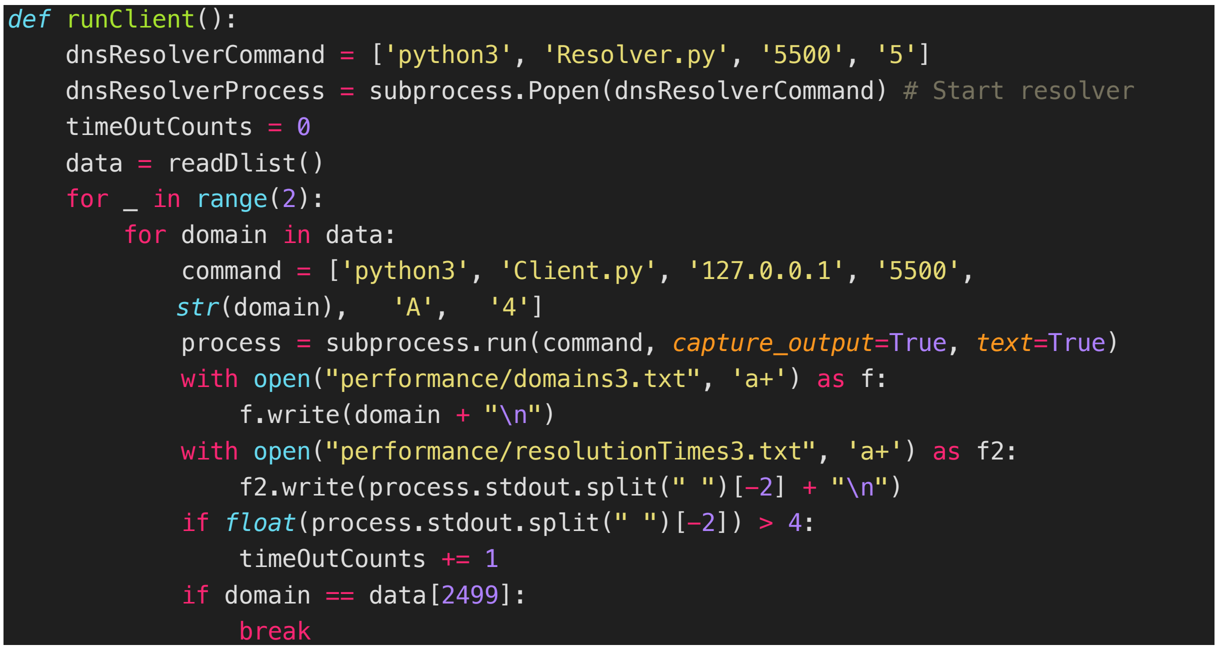
COMP3331 Assignment 1 - Performance Analysis z5358739

1. How was the experiment conducted?

The experiment was conducted by comparing the resolver developed for the assignment, Resolver.py against existing standard public DNS resolvers such as the Google DNS, 8.8.8.8 and the Cloudflare DNS, 1.1.1.1. Three experiments were run, which each independently sent 5000 queries to their respective DNS Resolver

Analysis of resolving times was conducted by calculating the time taken between the sending of a query by the client and receiving an answer from the resolver using the python standard time module. By using the Client.py command line interface to send DNS Queries and measuring the resolving time when 5000 independent A RR queries are sent to each of the respective resolvers, the individual times taken for each query were written to an external text file, as seen in the performance folder provided in my submission. This process was automated through an external python file, performanceRunner.py that iterated through a list of 2500 domain names twice and using the standard subprocess module to run the terminal commands necessary to execute the client until termination. The time taken for each query was then extracted from the print() output produced by the client using the stdout variable available from a process instance.



```python
def runClient():
    dnsResolverCommand = ['python3', 'Resolver.py', '5500', '5']
    dnsResolverProcess = subprocess.Popen(dnsResolverCommand) # Start resolver
    timeOutCounts = 0
    data = readDlist()
    for _ in range(2):
        for domain in data:
            command = ['python3', 'Client.py', '127.0.0.1', '5500',
            str(domain),   'A',   '4']
            process = subprocess.run(command, capture_output=True, text=True)
            with open("performance/domains3.txt", 'a+') as f:
                f.write(domain + "\n")
            with open("performance/resolutionTimes3.txt", 'a+') as f2:
                f2.write(process.stdout.split(" ")[-2] + "\n")
            if float(process.stdout.split(" ")[-2]) > 4:
                timeOutCounts += 1
            if domain == data[2499]:
                break
```

Figure 1: runClient function used to automated client query execution.

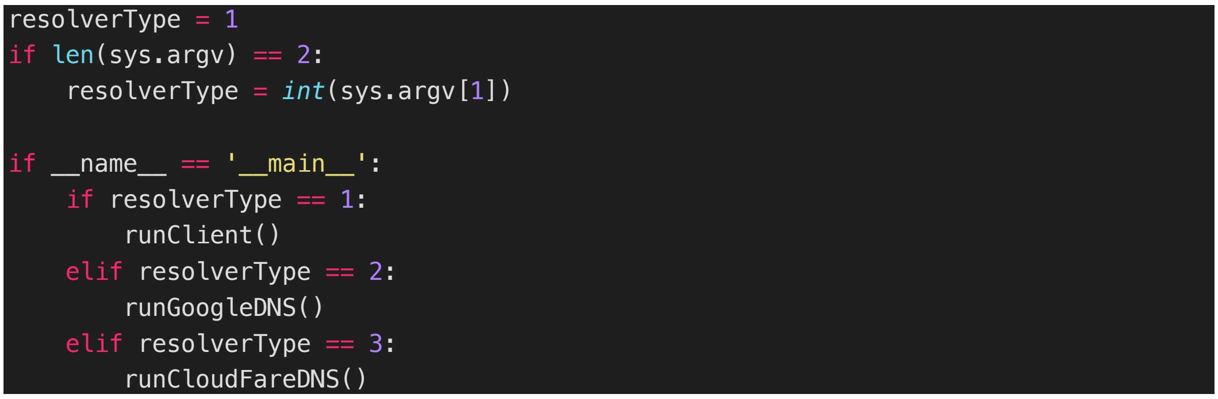
Similar methods were run for the google DNS Resolver and the Cloudflare DNS Resolver that queried each respective resolver, sending 5000 queries in total each.

Domain names used remained constant amongst all three experiments alongside the query type, which was an A record query. A constant timeout value of 4 seconds was set amongst all three experiments to limit performance issues and avoid cases with outlier resolver times that would skew the dataset.

For further clarification on automation process, please refer to performanceRunner.py in the provided submission.

2. How can the experiment be repeated?

The experiment can easily be repeated due to the performanceRunner.py file that was developed. This file acts as an external script that performs the task of executing the client, Client.py and sequentially sending 5000 queries to each DNS resolver, and then collecting the query resolving time from the output of the client and writing it to an external text file. As seen in figure 1, the program executes the Resolver and Client with pre-set command line arguments, to ensure that the experiment variables can remain constant and to maintain data integrity and consistent results. Furthermore, a constant timeout period is used to place a cap on the query times in order to avoid outliers amongst all three experiments. Three functions have been developed that independently run the experiment on the three DNS Resolvers. This was done to ensure each DNS Resolver's experiment is run independently and that the queries are not dependent on one another or clashing with one another due to parallel queries. Furthermore, for each query sent, the performanceRunner waits for the full execution of the Client.py to ensure no parallel queries. The data was stored in an external text file for ease of use, and easy comparisons between data sets through tools such as excel.



```
resolverType = 1
if len(sys.argv) == 2:
    resolverType = int(sys.argv[1])

if __name__ == '__main__':
    if resolverType == 1:
        runClient()
    elif resolverType == 2:
        runGoogleDNS()
    elif resolverType == 3:
        runCloudFareDNS()
```

Figure 2: Usage of performanceRunner.py

However, without use of the performanceRunner.py file, the experiment can be repeated by through the following general steps:

1. Gather a list of domain names to send to the Resolver.

2. For each domain name, send a query through Client.py to the Resolver.

3. Capture the response and extract the resolving time from Client.py output.

4. Combine data into a single dataset and repeat experiment for other resolvers to compare datasets.

3. Show the results presented in tabular form and in graphical form

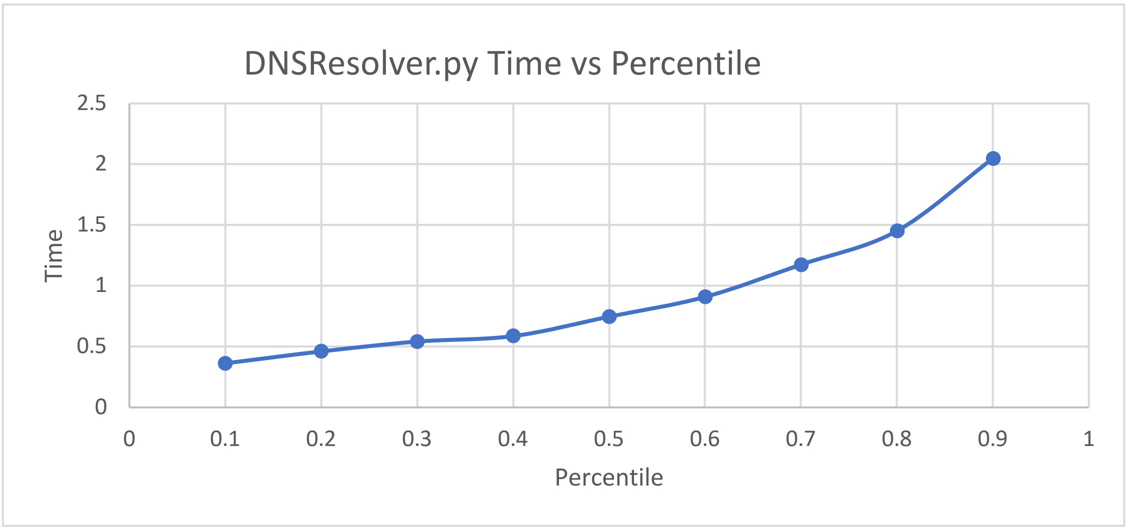


Figure 3: Time for query response vs Percentiles results for Resolver.py (n = 5000)

Percentile	Time
10th	0.3629
20th	0.46042
30th	0.54166
40th	0.58658
50th	0.7462
60th	0.90886
70th	1.1745
80th	1.45156
90th	2.04809

Figure 4: Percentile vs Time tabular data for Resolver.py (n = 5000)

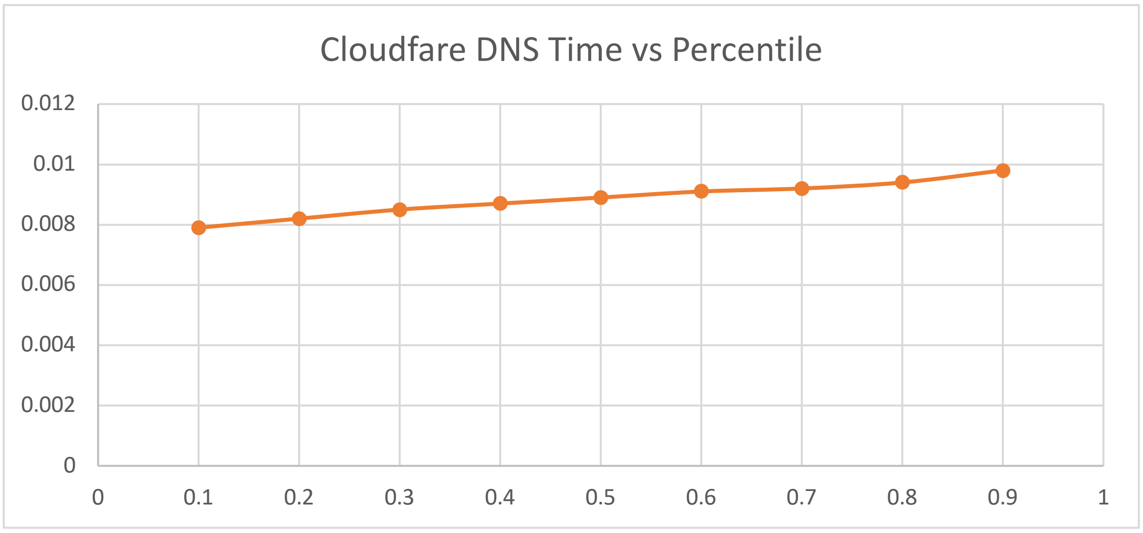


Figure 5: Time for query responses vs Percentile for Cloudfare DNS Resolver (1.1.1.1) (n=5000)

Percentile	Time
10th	0.0079
20th	0.0082
30th	0.0085
40th	0.0087
50th	0.0089
60th	0.0091
70th	0.0092
80th	0.0094
90th	0.0098

Figure 6: Percentile vs Time tabular data for Cloudfare DNS Resolver (1.1.1.1) (n = 5000)

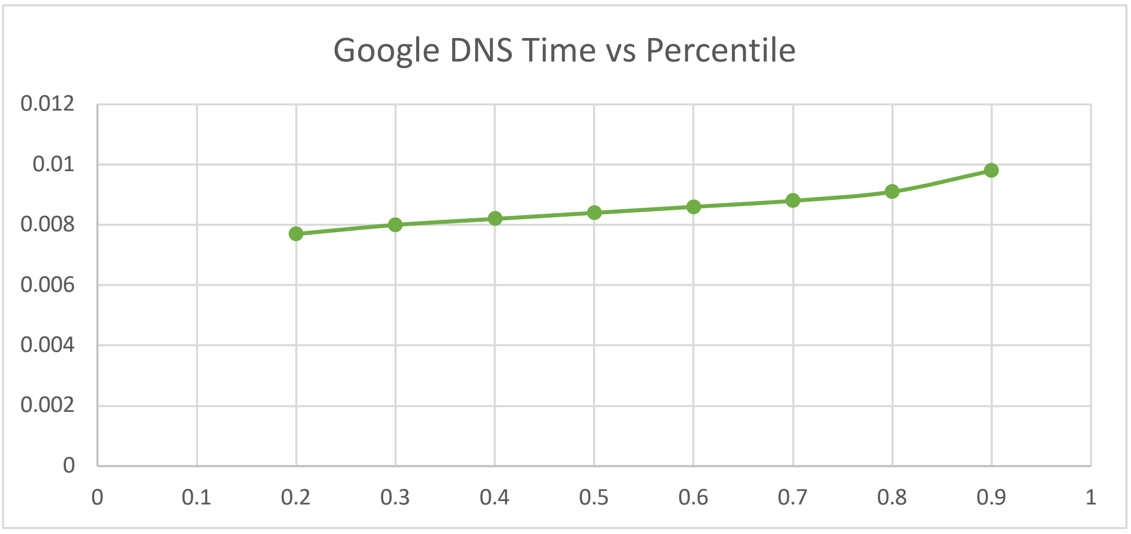


Figure 7: Time for query responses vs Percentile for Google DNS Resolver (8.8.8.8) (n=5000)

Percentile	Time
10th	0.0079
20th	0.0082
30th	0.0085
40th	0.0087
50th	0.0089
60th	0.0091
70th	0.0092
80th	0.0094
90th	0.0098

Figure 8: Percentile vs Time tabular data for Google Resolver (8.8.8.8) (n = 5000)