CSE 291 Homework 2: Sorting Around the World

Qing Zhang

PID: A53235512 GitHub ID: 32892259

1 Latency Experiment

The first group of experiments has been conducted to measure the round-trip latency, where `ping()` is called three times for each pair of the cities. Tab 1.1[1] lists the results of latency measure. It can be observed that on average the latency for each pair of cities is stable and the length is closely related to the distance between cities. In Table 2.1, I list the distance between cities with the help of Google maps and find that the latency is directly proportional to the distance. However, the latency between Seoul and Ireland is longer than that of Ireland and Sao Paulo, while the distance of former is shorter than the latter. That is probably because of other factors, such as the number of routers, network traffic and other users.

Further, I also conducted an experiment to measure the latency by call `ping()` upon connection. The results are displayed in Table 1.2. The measurement turns out to be much longer than the previous experimental results, which might because that overhead induced by building the connection or local cache of IP address.

Table 1.1: Results of Latency Experiment (on average / after the first call).

Source City	**Destination City**	**Date/Time**	**Exp. 1 (ms)**	**Exp. 2 (ms)**	**Exp. 3 (ms)**
Seoul	Ireland	5/05/2018, 4:23pm	252.440642	252.101974	253.593445
Ireland	Sao Paulo	5/05/2018, 5:20pm	191.215437	191.855698	190.24646
Sao Paulo	Mumbai	5/05/2018, 2:47pm	309.54034	306.696571	306.18090
Mumbai	Seoul	5/05/2018, 3:46pm	163.192896	162.900802	162.855991

Table 1.2: Results of Latency Experiment (call `ping()` upon connection).

Source City	**Destination City**	**Date/Time**	**Exp. 1 (ms)**	**Exp. 2 (ms)**	**Exp. 3 (ms)**
Seoul	Ireland	5/05/2018, 4:23pm	1022.331431	1023.917656	1009.138885
Ireland	Sao Paulo	5/05/2018, 5:20pm	903.093246	901.780606	895.835927
Sao Paulo	Mumbai	5/05/2018, 2:47pm	1324.153344	1155.289933	1153.736149
Mumbai	Seoul	5/05/2018, 3:46pm	724.988137	727.043714	719.098645

[1] Here we use the dense table to show the results instead of the table style required in the handout, since we only measure 4 out of 16 cities pairs.

2 APPLICATION-LEVEL THROUGHPUT EXPERIMENT

The second group of experiments has been conducted to measure the application throughput. The throughput estimates are obtained by dividing the number of integers sorted with the total invocation time. The results are shown in Table 2.2. The number of integers sorted is selected to ensure that the RPC call takes about 20 seconds or so (Table 2.1). Similar conclusion can be achieved that the performance is inversely proportional to the distance between cities and fluctuates due to other factors.

As for the application throughput with respect to bytes sorted per second, since in Java `int` type is 4 bytes width, the throughput(bytes/sec) can be easily obtained by multiplying the results in Table 2.2 with 4.

Table 2.1: Number of Integers used in the Experiments.

Source City	**Destination City**	**Distance (mi)**	**Number of Integers**
Seoul	Ireland	5589.92	2200000
Ireland	Sao Paulo	5,809.52	2800000
Sao Paulo	Mumbai	8,464.05	2000000
Mumbai	Seoul	3,473.20	3500000

Table 2.2: Results of Application-Level Throughput experiment

Source City	**Destination City**	**Date/Time**	**Exp. 1 (int/ms)**	**Exp. 2 (int/ms)**	**Exp. 3 (int/ms)**
Seoul	Ireland	5/05/2018, 4:23pm	112.77494452	112.05936465	112.07073763
Ireland	Sao Paulo	5/05/2018, 5:20pm	145.79915812	142.40511399	146.16857288
Sao Paulo	Mumbai	5/05/2018, 2:47pm	90.0227134	91.73731616	94.64683741
Mumbai	Seoul	5/05/2018, 3:46pm	169.44133374	165.22275912	166.67167568

3 NETWORK-LEVEL THROUGHPUT EXPERIMENT

The last group of experiments has been carried out to measure the one-way network throughput. The results are shown in Table 3.1. The throughput is inversely proportional to the distance between cities and affected by other factors in the mean time. The one-way network throughput is sightly double higher than the application throughput, which indicates that the data transfer between server and client consumes most of time in the GlobeSort application compared to the sorting itself.

Table 3.1: Results of Network-Level Throughput Experiment.

Source City	**Destination City**	**Date/Time**	**Exp. 1 (int/ms)**	**Exp. 2 (int/ms)**	**Exp. 3 (int/ms)**
Seoul	Ireland	5/05/2018, 4:23pm	237.82091523	240.07616191	236.04188681
Ireland	Sao Paulo	5/05/2018, 5:20pm	314.95492477	315.17828462	314.25790264
Sao Paulo	Mumbai	5/05/2018, 2:47pm	198.45656836	194.34885313	198.93712423
Mumbai	Seoul	5/05/2018, 3:46pm	369.4427397	370.99490588	374.58759486