Delft University of Technology

Quantitative Evaluation of Embedded Systems

IN4390

Lab 1 (Optional Part)

Authors: Jiaxuan Zhang(5258162)
Yiting Li(5281873)
Group ID: (26)

December 14, 2020

1 Global Clarification and Acknowledge

1.1 Some Statements

Several global configuration that are used while dealing with these 4 questions should be clarified:

1. Question 6 and 8 are implemented in the same computer, while Question 5 and 7 are implemented in another computer. Question6's remote communication is done with Group50.

2. Same as previous experiment, during each test, abnormal data can be found in the beginning or in the middle. Reasons for unexpected value in the beginning may be that the build up of communication need time during the early stage of communication, it can be partly illustrated by "cold start" phenomenon. Reasons for unexpected value in the middle may be that some turbulence(e.g. some unrelated programs or tasks) occurred during the experiment. Method to eliminate the influence of these unexpected value can be discarding or substituting them with other data. Details will be mentioned in each chapter.

1.2 Acknowledgements

We would like to express our appreciation to all those who provided us the possibility to complete this experiment. They are Tianrui Mao and Yuan Fu from Group 50. Together with them, we completed the experiment 6, discussed the results, and solved some problems.

2 Question 5

2.1 Background and Obtaining Data

The goal of question 5 is to check the relation between ROS2 communication latency in two nodes when CPU frequency scaling, by deriving a box plot diagram.

To get the data we need

1. repeat experiment in Question1 with CPU frequency scaling is off.

2. turn on the CPU frequency scaling, repeat experiment in Question 1.

2.2 Modifying, Displaying Data

In fact, those data are not so smooth, we can substitute those abnormal data with original median.

After modification, a box plot of transporting time of each data size can be drawn. The box plot is shown as follow,

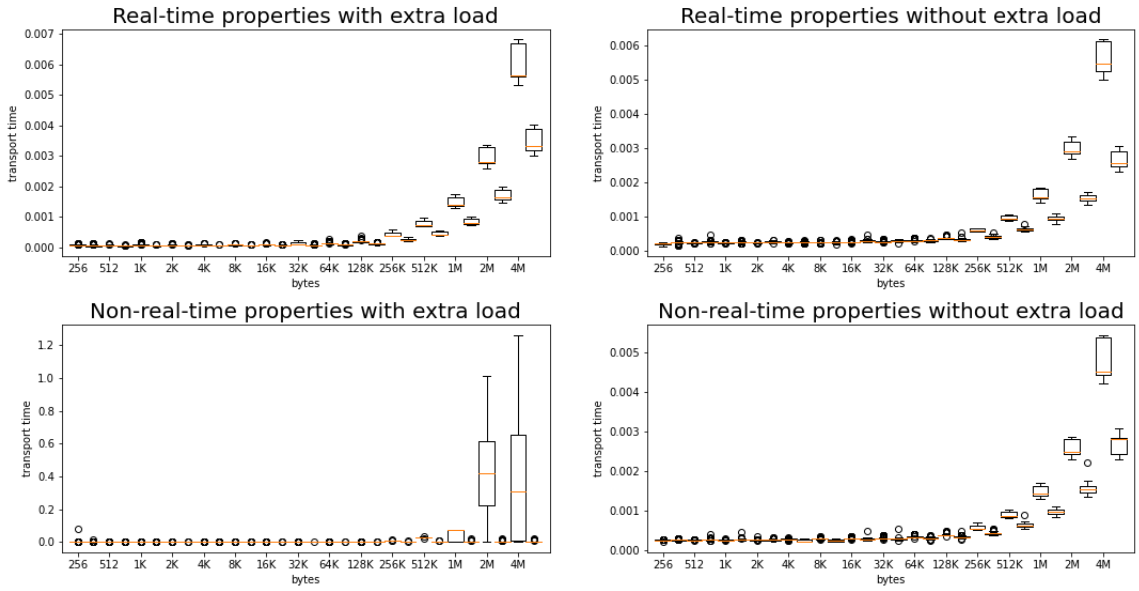
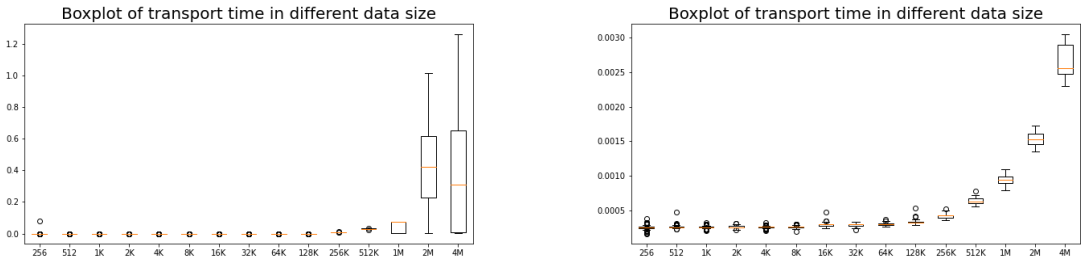


Figure 1: box-plot of transport time in different data size

Due to the huge difference between two experiment, the left bottom figure is not clear, the detail is shown as follow,



(a) non-real property with CPU frequency scaling off and extra work load

(b) non-real property with CPU frequency scaling on and extra work load

Figure 2: transport time of nodes when non-real property and extra work load present

2.3 Interpreting Data and Conclusions

In the Figure1, under each value in the x-coordinates of each picture, the picture on the left represents situation where CPU frequency scaling is off, and the picture on the right represents situation where CPU frequency scaling is on.

From Figure 1 and Figure 2, the performance will be improved when CPU frequency scaling is on. It may because the CPU will adjust its frequency to feat tasks, when transported data size is large, CPU will increase it frequency to satisfy the transmission requirement. Therefore, we can obtain some conclusions that:

1. Real-time property can shorten the transport time, the absence of extra work load can also lead to a smaller transport time.

2. Turning on the CPU frequency scaling can accelerate the communication, leading to overall small delivering time.

3 Question 6

3.1 Background

In this question, there are three parts to be finished:

1. Measure and draw the box plot of the latency of each message size;
2. Draw a curve to show the average latency and the 80% confidence interval of each point.

3.2 Implementation

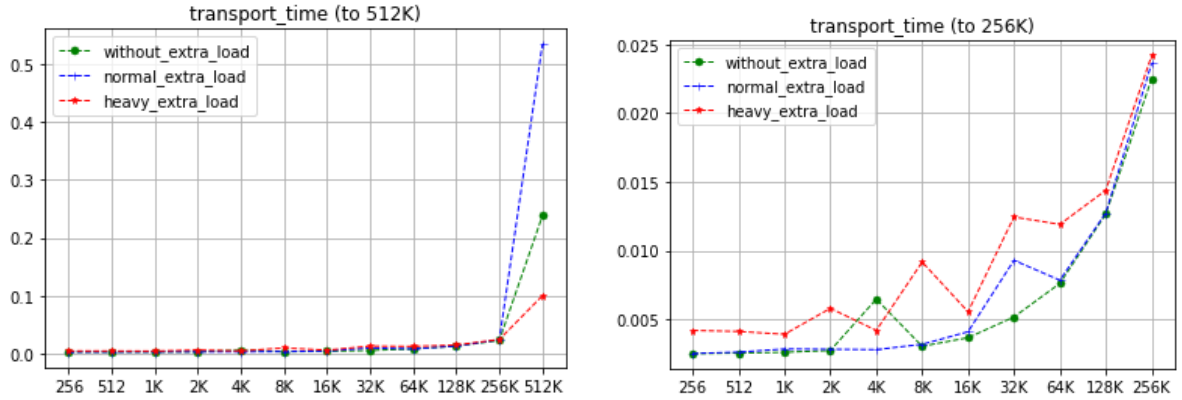
We measured the latency in two machines, one with SSD and the other with HDD. We use DDS method Opensplice to measure the delay, and set the QoS to "reliable", and sample 300 points for each size. We tested the WIFI environment using online WIFI speed test, the average download speed is as high as 76.8 Mbps, and the average upload speed is 94.6 Mbps. So we implement the measurements in 4 steps:

1. First of all, we measure the delay when there are only two machines under this WIFI, one of which is talker and the other is listener.
2. Next, we add a normal extra workload with several telephone interfaces in this WIFI, and watch some short videos
3. Then another computer and another mobile phone joined WIFI, and started to download the game package, with the download speed almost 15 Mbps.
4. Finally, we analyze the data using Python.

.

3.3 Result and Analysis

The curve of mean of time latency is shown in Figure 3 .When measuring, we found that message with size larger than 512 KByte has a very large latency and it can not be done even in several hours if we want more than 120 sample points. Through the system monitor, when we set sample points to 100, we found that the total upload size from the speaker increased by more than 1Gbyte. This phenomenon is unexpected and we don't know how to handle it. Therefore, we only measured the delay for message less than 1 MB. There is a big jump on the latency from 256 Kbytes to 512 Kbytes, to make the graph more clear, we also draw charts for message size less than 512 Kbytes.

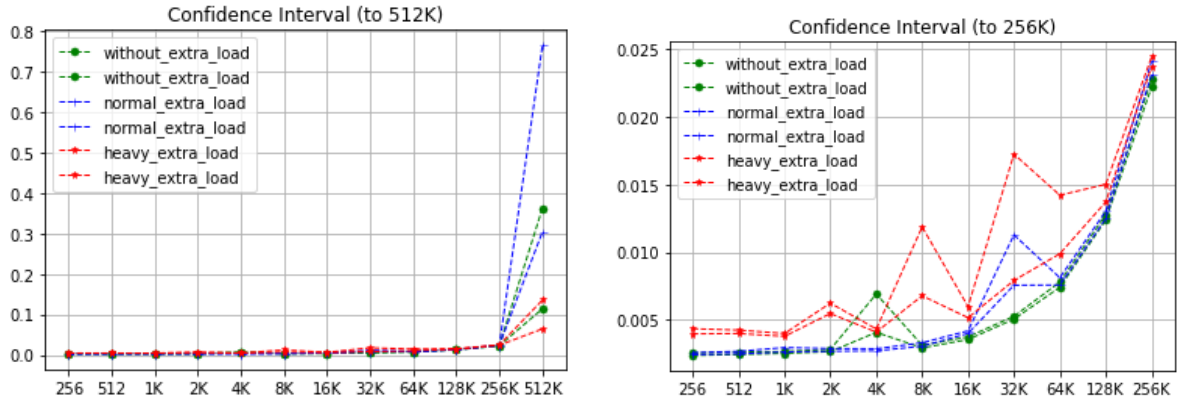


(a) Transport Time to 512Kbyte

(b) Transport Time to 256Kbyte

Figure 3: Transport Time

The confidence Interval of these points are shown in Figure 5. The box plots are shown in Figure **??**.



(a) Confidence Interval to 512Kbyte

(b) Confidence Interval to 256Kbyte

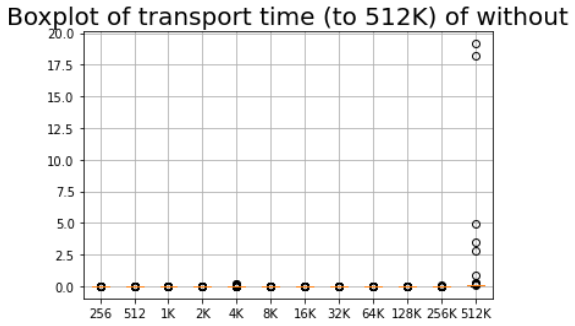
Figure 4: Confidence Interval

In the Figures, when the message size changes, the extra workload under the Internet has very different effects.

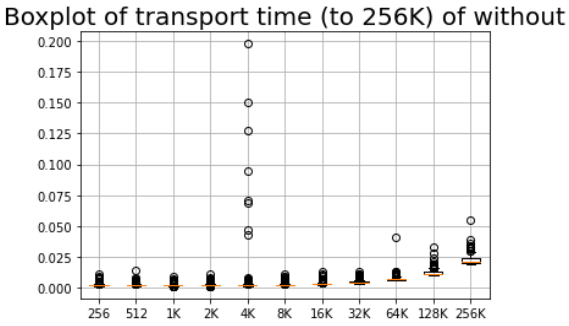
1. When the message size is relatively small (less than 512 kilobytes), the workload always deteriorates the performance. It can be seen from the box diagram that with the increase of extra workload, the number and amplitude of outliers are also increasing. Besides, in the confidence interval graph, the greater the extra workload, the greater the difference between the upper bound and the lower bound. In the mean graph, the increment of the mean also has such a trend.

2. At 512k, however, the trend is different. The case with the maximum additional workload has the smallest confidence interval and the lowest delay. From the box diagram, the heaviest extra workload will even reduce the outlier. For the normal extra workload, the delay will increase and the fluctuation will become more serious.

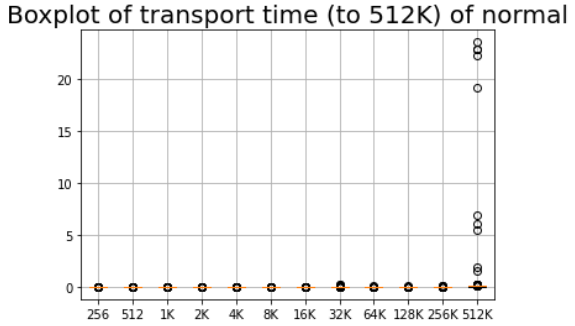
This phenomenon is very strange, and we tried to find some possible explanations. We think this may be due to the dynamic resource allocation strategies of network. It seems that the dynamic allocation of the network bandwidth occurs not only when several machines are connected to the same WIFI, but also between different routers in a large building such as student apartments.



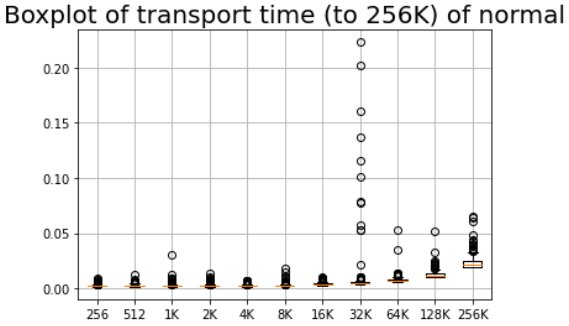
(a) Box 512 Without



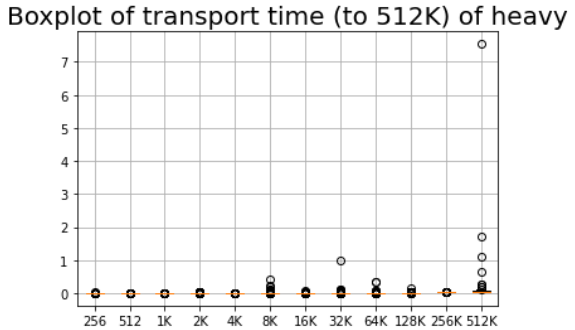
(b) Box 256 Without



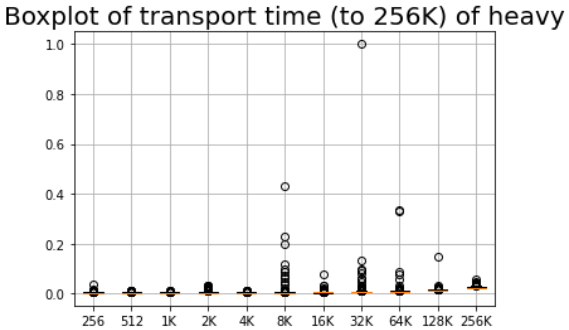
(c) Box 512 Normal



(d) Box 256 Normal



(e) Box 512 Heavy



(f) Box 256 Heavy

Figure 5: Box Plot

1. When the message size is large and there is heavy extra workload interference in the same WIFI, more resources will be allocated to the router. It explains the phenomenon that when the message size is 512 Kbyte, the latency is the most stable and the lowest with the heaviest extra workload.

2. When the message size is small, the addition of the demand for DDS and the heaviest extra workload is still not large enough for being allocated for more resources. Meanwhile, the extra load will occupy the limited resource, deteriorating the DDS communication with small message size.

3. The normal extra workload is not heavy enough for applying for more resources, so as the previous reason, the extra load squeezes the limited resource, deteriorating the latency.

It seems that this interpretation can partly explain the phenomenon.

4 Question 7

4.1 Background and Obtaining data

The communication between multiple nodes may lay a impact on the latency of ROS's communication. Although the communication in ROS is subscribing and publishing, which means that multiple listeners can subscribe one topic at the same time, in general idea, as the amount of listeners increase, the latency will still grow. The bias due to multiple nodes communication should be evaluated. To achieve that, extra nodes should be created.

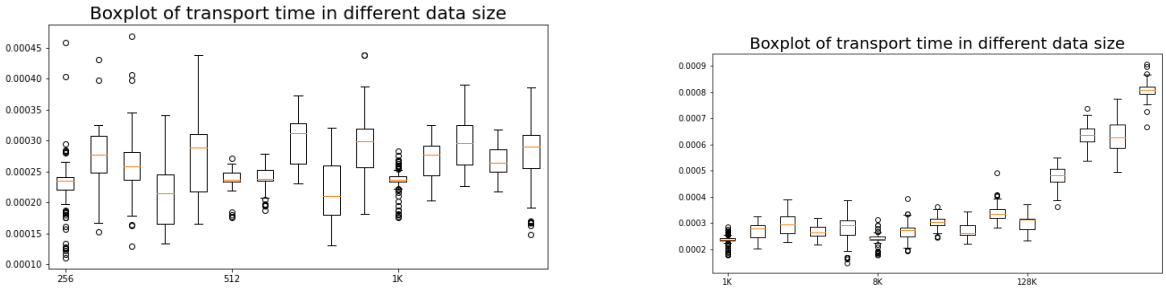
The original listener node can be copied many times to create a certain amount of node. As the Fast RTPs may have trouble when dealing with task that has a large data size, in this question we choose OpenSplice protocol. We increased the amount of listener nodes from 1 to 10, then repeat the experiment mentioned in Question1, the transport time should be stored.

4.2 Modifying, Displaying and Interpreting Data

Same as the Question1, some data are so abnormal that brings a huge bias to final result, we eliminated those data by using median to replace abnormal data.

After experiment and data modification, 10 groups of data is presented, the difference between groups should be presented first. Displaying all data won't be possible because there are 55 nodes in total, therefore we chose node0 in group0 (group$n$ means that n nodes are subscribing at the same time), node1 in group2, node3 in group5, node7 in group7, node5 in group9.

As all nodes subscribe topic at the same time, randomly picking one node in group is an appropriate way. We drew a box-plot to analyse them, the diagram is shown as follow:



(a) small data size

(b) middle data size

Figure 7: comparison among different data size

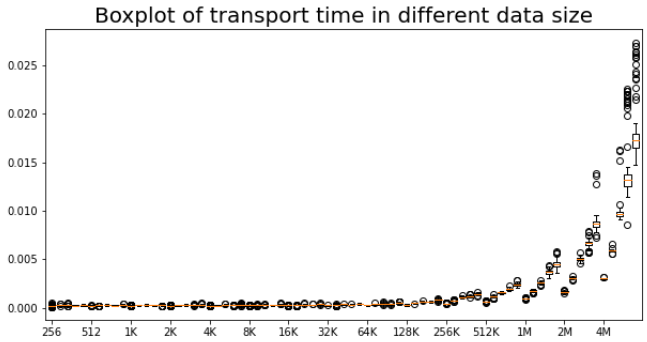


Figure 6: comparison of all data

From the Figure 8(within label"4M", from left to right is node0 in group0, node1 in group2, node3 in group5, node7 in group7, node5 in group9), the increasing trend of transport time is easy to notice, the first trend is that transport time will increasing when data size that is been conveyed increases. The second trend is that when listeners increases, the latency of communication also increases.

The reason for second trend is that when ROS facing multiple destination, it takes time to prepare each subscriber to connect to the published topic. Therefore, as destination grows, communication delay will grow.
When we check the Figure8 carefully, we found that the difference between multiple destination is more apparent as data size grows up, as shown in the Figure7(a) and Figure7(b). This trend show that ROS has a good real-time properties when facing small-data-multiple-destination task. Transport time of large number of destination is still same as that of small number of nodes.

Other properties can be searched further is the latency of each node communication when the overall number of listeners is fixed. We drew the box plot of 4 nodes in group4, the figure is shown as follow,

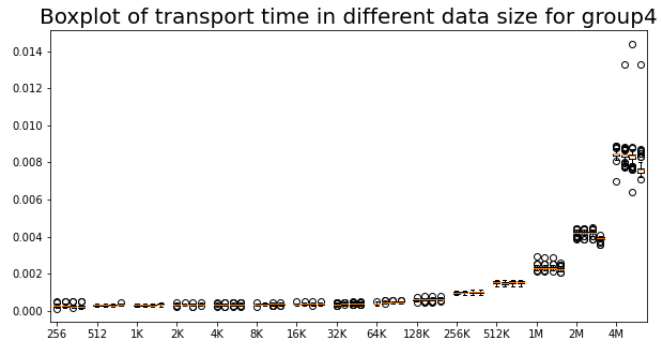


Figure 8: comparison of all data

While the growing trend of transport time is still obvious, there is one thing interesting, that is all node has the same transport time when dealing with different data size task, despite some small fluctuation. The ROS1 doesn't have this properties, ROS makes a improvement. The possible reason for this property is that ROS2 wants to maintain the synchronization of multiple destination communication.

4.3 Conclusion

We drew a conclusion that:

1. ROS2 can preserve the synchronization when facing multiple destination communication task. Each node consumes almost same time to fully subscribe the content from the published topic.

2. While delivering small size data, the amount of nodes will not affect much on the transport time of communication. While transporting big size data, the consuming time will grow fast(maybe exponentially) as the number of nodes grows linearly.

5 Question 8

5.1 Background

In this question, there are two parts to be finished:

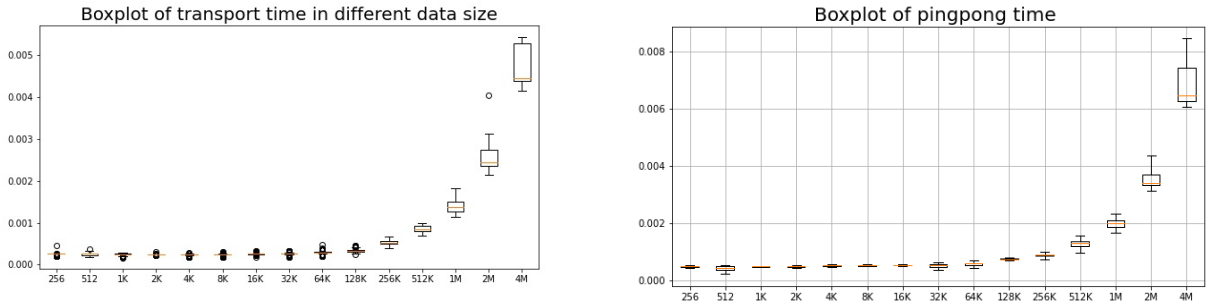
1. Implement ping-pong communication between two nodes, measure and compare with results from Question 1.

2. Implement a chain communication consists of five nodes and measure the end-to-end latency and the latency between individual steps.

5.2 Implementation of ping-pong communication

In this part, two nodes `talker_pingpong` and `listener_pingpong` was implemented and the communication latency with DDS method FastRTPS and message size from 256Byte to 4MByte are measured.

In the `talker_pingpong` node, it has a publisher and a subscriber. There are two threads in this node, one thread keep publishing on the topic `pingpong_test`, similar to the talker node in Question 1. While in another thread, by using `rclcpp::spin()`, the subscriber monitors the topic `pingpong_ack` and output the receiving time into corresponding documents. the acknowledge message is a simple message `x_ack` where x is a number smaller than 14, so the size of the acknowledge message keeps almost the same.



(a) ping communication

(b) ping-pong communication

Figure 9: ping-pong communication and ping communication

In the `listener_pingpong` node, it has a publisher and a subscriber. but different with `talker_pingpong` node, there is only one thread. By using `rclcpp::spin()`, the subscriber monitors the topic `pingpong_test` and record the receiving time into corresponding documents. The publisher will only publish a acknowledgement on the topic `pingpong_ack` when the callback function of the subscriber is waken up. In the callback function, the publisher will publish message as soon as the receive time of subscriber is calculated, so we take the publish time the same value as subscribe time.

5.3 Implementation of chain communication

In this part, 5 nodes `talker_A`, `listener_B`, `listener_C`, `listener_D`, `listener_E` are designed. `listener_B`, `listener_C`, `listener_D` has the same structure while subscribing or publishing on different topics. `talker_A`, `listener_E` are nearly the same with the nodes in Question 1.

`listener_B`, `listener_C`, `listener_D` have nearly the same structure with `listener_pingpong`. The topics of these five nodes is shown in Table 1

	talker_A	listener_B	listener_C	listener_D	listener_E
subscribe		A2B	B2C	C2D	D2E
publish	A2B	B2C	C2D	D2E	

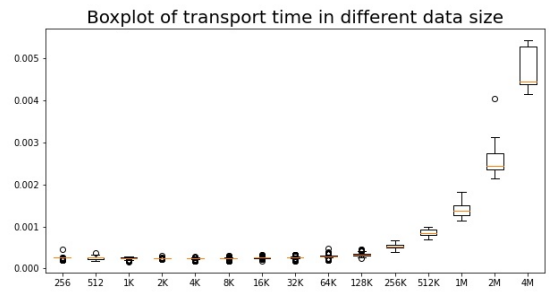
Table 1: Topics of five nodes in chain communication

5.4 Measurements

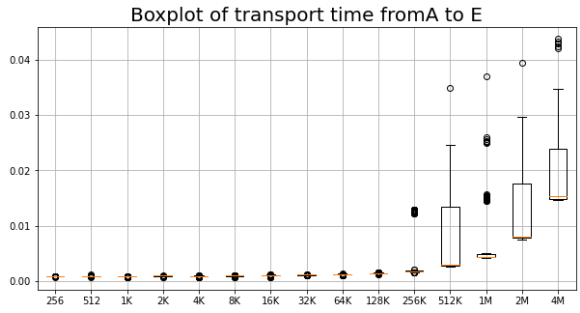
In each part, 500 samples points are recorded, and considering the "cold start" phenomenon, the initial 50 points are not calculated in each part.

5.4.1 ping-pong communication

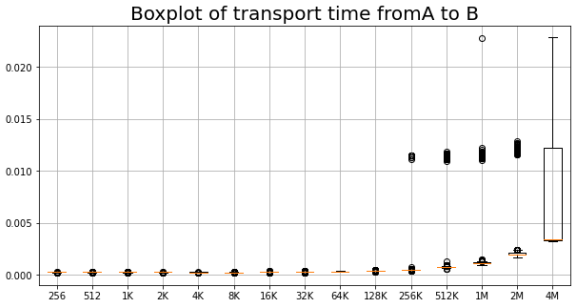
The measurements of latency of ping-pong communication is in Figure 9. It can be seen that when the message size is small, the increase of delay is slight. As the increasing of the message size, the impact of the round trip become obvious. Although the size of the acknowledgement message remains unchanged, the increment is different. The larger the data size, the longer the increment of latency. This may because larger message sizes will bring a greater burden to the system, so the latency of acknowledgement message may increase.



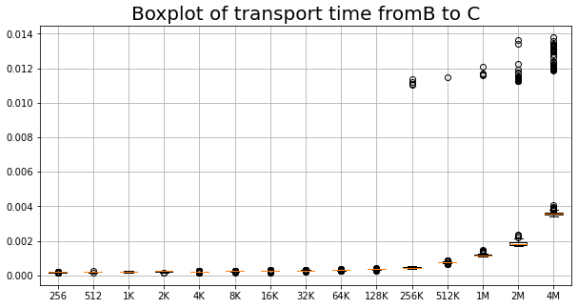
(a) ping communication



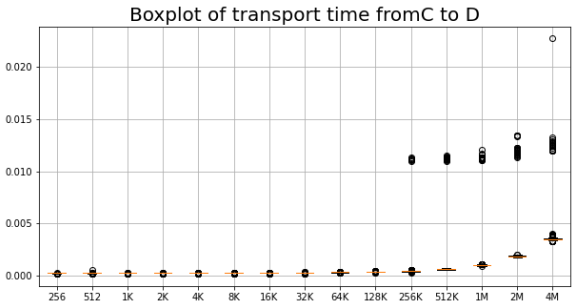
(b) A2E



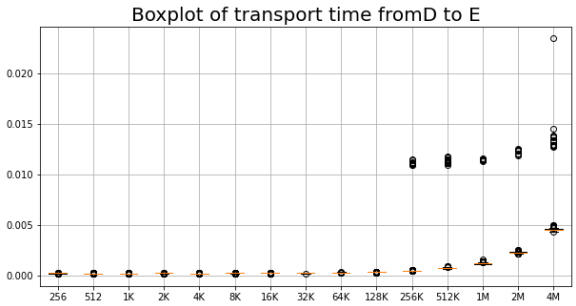
(c) A2B



(d) B2C



(e) C2D



(f) D2E

Figure 10: chain communication.

5.4.2 chain communication

The measurements of latency of chain communication is in Figure 10. According to the box plots, there are some phenomenons.

1. The median of each step of chain communication and the ping communication in Question 1 are almost the same.
2. The median of each step of chain communication are almost the same.
3. In each step of chain communication, there are more outliers than the result from Question 1.
4. The median of latency of A to E is almost the linear addition of the median of time of the same step.

The first and second phenomenon seem strange, but after looking at the performance benchmarks on FastRTPS website. The main feature that may increase latency is the number of subscribers on a single topic. In chain communication, there is only one subscriber and one publisher per topic, so most sample points have the same performance, as in the ping communication of question 1, even though

they are in different steps,.

Meanwhile, these nodes do run on the same machine simultaneously. Multiple steps will possibly bring extra burden to the computer and make the stability worse. It can be used to explain the increase of anomaly points.