CS120 Project Report

赵泽宇、戚嘉宸 - ShanghaiTech

Project 1

Task 1

We learned that the following content must be implemented using Java or C++, and it would be more convenient to use Java. Therefore, we ultimately decided to use Java to complete the subsequent project content.

Task 1 requires us to store the sound signal and play it again after a certain period of time. Initially, we discovered ExampleHost.java in *“JAsioHost”* because it was the only file at that time that we could comprehend. Although we roughly understand its operating logic, we still don't know how to use *“ASIO”* classes or functions. Upon realizing that this part did not strictly require the use of *“ASIO”*, we decided to use other Java sound collection and playback libraries, which can also achieve the ultimate goal.

Task 2

Task 2 requires us to output two sinusoidal signals. Since the example in ExampleHost.java is to output a sinusoidal signal, we only need to add it into two, and make the maximum value of |f(t)| not exceed 1 (we don’t know if it is useful). Then it can complete the goal.

Task 3

Task 3 requires us to use acoustic signals to transmit 10,000 "0" or "1" from node 1 to node 2 within a certain period of time, and ensure a certain accuracy.

Given that we cannot control the playback and reception conditions on the day of the check, we are unable to employ the ASK method to analyze the sound signal. As we did not anticipate a practical implementation method for PSK, we ultimately opted for the FSK method to analyze the acoustic signal.

Firstly, we need to understand the process of obtaining the frequency of a sound signal. To achieve this, we developed a new class, FFT, designed to convert the float array storing the acoustic signal sampling points into a frequency array through Fourier transformation. Subsequently, we compare the internal values of the array to identify the peak value, representing the frequency of the acoustic signal.

This method proves effective in parsing the frequency when dealing with an acoustic signal that transmits a single frequency and remains constant. However, when the frequency undergoes a change, the approach involves transmitting an acoustic signal with a length of bufferSize and a frequency of 1378Hz, followed by another acoustic signal with the same length. In situations where the acoustic signal with a frequency of 1378Hz is replaced by an acoustic signal with a frequency of 2756Hz, we encounter challenges in effectively analyzing the frequency of the latter period. In essence, aligning the acoustic signals becomes problematic, preventing us from ensuring that the detected range each time corresponds precisely to the same frequency segment of the acoustic signal. Faced with this dilemma and lacking a satisfactory solution, we eventually decided to abandon the task.

Project 2

Task 1

Task 1 mandates the use of an external sound card and a wired transmission method to achieve more efficient transmission based on Project 1. Given that task 3 of our Project 1 remains incomplete, we are required to initiate this task from the beginning.

Upon confirming that FSK will be utilized, we embarked on solving the alignment problem. Our devised solution involves saving the received acoustic signal in a large array, formatted as a float array, and then analyzing it after all the data has been transmitted. During parsing, a pointer called Head is employed to mark the starting position of parsing. Upon completing the parsing of a signal, Head is incremented by TransSize (where TransSize represents a unit length of the same frequency). Since the frequency obtained by FFT is consistently 0 before parsing the signal, as soon as a non-zero frequency is parsed, the data segment is entered. At this point, we aim to align the data to ensure smooth parsing of subsequent data. Therefore, we continue to increment Head by 1, i.e., move the sampling points one by one, and then perform FFT. When the obtained frequency changes, it indicates that the boundary of the original frequency change has been reached, signifying rough alignment completion. Subsequently, if Head+=TransSize is ensured, there is a high probability that the obtained frequency can be parsed normally. Upon later inspection, we discovered that by locating the sampling point where the frequency changed, moving back TransSize/4 sampling points, and then using FFT, the accuracy of the obtained frequency was higher.

However, uncertainties arise regarding whether the frequencies of the first two data of the check data are different. Consequently, we employ another frequency, distinct from the one representing the data, to mark the alignment request. When the receiver detects the alignment frequency, it gradually shifts the sampling points back until the frequency representing the data is obtained through analysis, and then shifts back TransSize/4 bits, completing the alignment.

As data is collected, the sampling can become biased again, necessitating multiple alignment attempts. We opt to increase the alignment frequency by two checkSize sizes after every 200 pieces of data and perform alignment using the aforementioned method. To meet the specified time constraints, we must either reduce the checkSize or increase the amount of data transmitted at each frequency. Unfortunately, due to the limitations of our self-built FFT, which can only identify 12 fixed frequencies, we cannot detect any frequency accurately. In the end, we decided to use one frequency to represent 3 bits (utilizing 2^3=8 frequencies plus an alignment frequency). However, the transmission was not completed within the stipulated 10 seconds, but rather within 15 seconds.

Task 2

Task 2 necessitates achieving 100% accuracy, implying the use of acknowledge (ack) to confirm the accurate parsing of data.

This requirement entails converting and parsing the sample points obtained in real-time, comparing the preamble content to determine whether retransmission is necessary. Instead of collecting all sample points and then performing unified conversion, we adopt a strategy of transmitting every 200 pieces of data as a package. We store the obtained sampling points in a float array with a large capacity and analyze each new sampling point until the remaining sampling points cannot support a complete FFT transformation.

After successfully achieving this part, we need to use the CRC verification method to check whether the obtained package is correct. If it is correct, we need to send back an ack to inform node 1.

Our strategy involves ceasing package transmission after sending one and resuming only when the ack is received. If the ack is not received after more than two Round-Trip Times (RTTs), we initiate retransmission. However, during testing, we observed that node 1 stopped transmitting after sending a package, and node 2 received and transmitted an ack. Still, node 1 failed to receive the returned ack, leading to program failure. Despite ensuring the opening of input and output channels for both nodes, we were unable to resolve the issue, ultimately resulting in the abandonment of this task.

Project 3

Task 1

Task 1 necessitates utilizing the completion of Project 2 to transmit ICMP packets between the two nodes for a ping connection. We encountered problems in Project 2 that required us to start over for resolution. Initially, we used a mixer to connect two nodes, but upon reevaluation, we found it unnecessary. We suspected that the remaining signal in the channel mixed with the acknowledge (ack) was causing node 1 to incorrectly analyze the frequency of the ack. To address this, we opted not to use the mixer, resulting in successful reception. Subsequent content was also easily resolved.

Returning to task 1, we simply need to initiate a ping request on node 1. The array generated in the form of a request packet is then converted into binary and transmitted to node 2. Node 2 parses the received request packet and responds with a reply packet. Node 1 records the time difference between sending and receiving packets, printing it out to complete the task.

Task 2

Task 2 requires us to extend the ping connection between node 3 and node 1, where node 2 and node 3 are connected to the network.

This implies that the packets sent back from node 2 to node 3 must adhere to the ICMP protocol. To address this, we employed the *"pcap4j"* library, successfully resolving this aspect of the problem.

However, challenges arose in the process of transmitting from the physical layer to the IP layer. Using two programs to handle the transmission of the physical layer and the IP layer separately led to complications. After node 3 transmitted the request packet to node 2 through the IP layer and received it, node 2 could not pass the packet to node 1. Similarly, after node 1 passed the reply package to node 2, node 2 could no longer pass the package to node 3.

To overcome this issue, we attempted to use *"socket"* to transmit data between the two programs. However, due to our unfamiliarity with the library or the incompatibility of this library, we were unable to successfully transmit data between the programs, resulting in a time delay that exceeded the tolerable range. Consequently, the task ended in failure.

Project 4

Task 1

Task 1 needs to build upon Project 3, but despite our concerted efforts, we struggled to devise a viable solution to the previous problem. Regrettably, we were compelled to abandon this project.

Suggestions

1. We propose the introduction of more discussion classes to guide and instruct on the usage of critical tools such as *“JAsioHost”* or *“Jucer”* and *“Pcap4J”*.
2. Including practical applications of physical layer transmission and data parsing in the curriculum could be beneficial, given the significant time investment required, even if not deemed the most crucial aspect of the course.
3. We hope to provide us with external sound cards as soon as possible to reduce the impact of our own computer speakers and microphones.
4. I hope there will be a certain number of in-class tests to monitor whether we understand the teaching content well. Just the final exam will make it difficult for us to evaluate our absorption of knowledge in a timely manner.
5. Based on the previous suggestion, TA can set up exercise classes and conduct reviews for each test or assignment.

Acknowledgment

I express my sincere gratitude to Professor Yang Zhice and Professor Chen Haoxian for their vivid, engaging, and logically clear lectures. Their insightful teaching has greatly enriched my understanding of the subject matter. I also extend my thanks to the TAs for their patient guidance throughout the course. Finally, I appreciate the course CS120 for providing me with interesting and meaningful knowledge.