Project Retrospective - P1 - Brian Sung

1. Given the same goals, how would you complete the project differently if you didn't have any restrictions imposed by the instructor? This could involve using a particular library, programming language, etc. Be sure to provide sufficient detail to justify your response.

I think I would prefer to implement the project with Python. In my opinion, Python simplifies the overall syntax and would give me a shouter implementation period. I used a lot of type converting in the code and it's just much easier to do so in Python.

Here are some Java-vs-Python example with the code in my project:

Java	Python
`String.valueOf(int)`	`str(int)`
`Integer.parseInt(String);`	`int(str)`
`if (something != null) {}`	`if not something:`
`System.out.println("Chunk [" + i + "] = [" + chunkName + "], hash = [" + hash + "]");`	`print(f'Chunk [{i}] = [{chunkName}] , hash = [{hash}]')`

`import StorageMessage`
`StorageMessages.Message m = StorageMessages.Message.newBuilder().setType(StorageMessages.messageType.REQUEST).setFileName(filename).setChunkId(i).setHash(ByteString.copyFrom(hash.toByteArray())).build();`
	`import storage_message`
	`m = storage_message.Message()`
	`m.type = storage_message.messageType.REQUEST`
	`m.filename = filename`
	`m.chunk_id = i`
	`m.hash = str(hash)`

Those are the huge optimization in syntax in my opinion, and there are more.

2. The system we designed is decentralized, but this comes at a cost. Discuss this trade-off.

In order to achieve truly decentralized, there are some trade-off of stability. When a new node joins the network or a node crash or leave the network, there is a short period that the network is not stable. During the period, the replication process is proceeding and not all of the nodes realize that there is a new node. So when a client try to send a request to the network, some nodes could route the request to a wrong node. Also, if a node stores a large dataset, the replication process cloud take very long to complete and return to a stable stage.

3. Let's imagine that your next project was to improve and extend P1. What are the features/functionality you would add, use cases you would support, etc? Are there any weaknesses in your current implementation that you would like to improve upon? This should include at least three areas you would improve/extend.

First, I didn’t get a chance to add “removing” functionality into client side so I would add this feature into the system. Although StorageNode can remove redundant chunks after replication, there is still a need for client to do such operation.

Second, currently, my chord network can handle “some” failures. If two StorageNode crash in the same time and they happen to be side-by-side nodes, the network will crash since I only maintain two successors in each node. (The recovering process requires access to successors.) However, from the resources I read, they have a whole list of successors recorded in each StorageNode. Thus, if multiple successors crash, they could always recover from the “next” live successor. I would have to make some changes in my chord implementation to achieve that.

Finally, my current approach for client to access StorageNodes is to maintain a buffer that records a list of StorageNode that the client has accessed, and each request from client will be randomly sent to one of the StorageNode in the buffer. (That is for failure handling mostly.) There is no formal load balancing for this approach so some nodes could handle more requests than others. That would be a good improvement to add to my system.

4. Give a rough estimate of how long you spent completing this assignment. Additionally, what part of the assignment took the most time?

For average, I spent 20 hours per week. The whole project took me about 5.5 weeks so, in total, it took me 110 hours proximately. The implementation of chord protocol and its failure handling took the most of the time since it is entirely new for me and I had to read a lot of papers and resources.

5. What did you learn from completing this project? Is there anything you would change about the project?

I am very satisfied with the project. Although I have implemented a zero-hub DHT previously, this project gave me a whole new experience implementing a different and **truly** decentralized DHT. I learned that you don’t necessary need to maintain all information in all nodes, instead, we can achieve a lot of functionalities with a mathematical way, which gave us a better optimization of system resource and no overshot.