SRTP mid-term report

Shen Keyi

In this report, I will review what I have learned about membrane computing since the project was launched.

First, a *membrane structure* consisting of several membranes arranged in a hierarchical structure inside a main membrane. Regions delineated by the membranes contain *multisets* of *objects.* This is an initial configuration of the system, then computation proceeds, and stops when the result is in the specific output membrane.

Membrane computing abstracts from cell structure and functioning in biology, with the aim of finding computability-relevant ideas, and the generated models and results are relevant for computer science.

The relevant literature mainly focuses on the construction of various membrane computing models based on cellular biochemical reactions and the analysis of the computing power and computing efficiency of the models, rather than the construction of membrane computing for specific problems.

But as a compartmental computational model based on communication between regions, it has the separate "processors" communicating in various ways. Since processors can deal with multisets in the synchronized, non-deterministic and parallel way, Membrane system has strong computing power, so it also has broad application prospect in other fields.

In chapter 2, I first reviewed the knowledge of Plasma Membrane and the mechanism about how neurons work together. I basically learned them when I was in high school, and this biological knowledge helped me to have an intuitive framework understanding of membrane computing. After that, I learned the basic concepts of formal language, automata theory and complexity theory as the basic and important background knowledge for membrane computing. I was amazed at the definitions and theorems of these written systems. At present, I think I can basically understand the content (such as the definitions of a large number of concepts and the possible relationships between concepts) mentioned in the textbook, but I may not use it very well. Many proofs of the theorem are difficult for me.

In chapter 3, I learned the basic model and some extensions of the symbolic object membrane system. Simply, extensions are mainly from adding more features (such as considering membrane dissolution, introducing priorities among the evolution rules) and adding additional restrictions (such as controlling the permeability of membranes, using promoters/inhibitors). Almost all of these extensions have biological origins. Through these extensions, we try to stay as close as possible to the biological reality, especially to the ways of transferring chemical compounds through membranes, and, at the same time, to keep the models as simple/elegant as possible from a mathematical point of view. After further study, I also learned that these different systems have different computing capabilities. These provide some flexibility and insight for our future applications of membrane computing. In the application, we need to add specific features and restrictions according to the actual situation to build membrane systems with symbol-objects.

Of course, none of the systems we have talked about obey the conservation laws of objects, and when we do computation by communication only, since the system never change, create, or destroy the objects, the conservation laws of objects is kept in the system. And under the mechanism of communication and object conservation, we can draw a conclusion: the system is computationally complete. The communication mechanism makes the system transform from independent computing mode between a collection of separate computing agents to cooperative computing mode. Furthermore, we can prove that purely communicative membrane systems are still computationally universal. In addition, the fourth chapter also mentioned a technical detail. In purely communicative membrane systems, in addition to the communication within the system, the communication between the system itself and the environment is also crucial. The system does this by changing the position of the membrane in which the object is located, and the number of objects does not increase (conservation). So, if we only use the finite multiset of objects inside the system initially, then the computing is also finite. We can assume that the environment contains infinite available copies of all objects. And the system can get new objects by communicating with the environment as needed.

Due to my internship in the summer vacation, I have not finished reading the introduction to membrane computing. Although I have gained some understanding of membrane computing, I still have no idea about the application of membrane computing and our project. Moreover, I lack experience in such things as "grammar"(But this is important for understanding the series of theoretical interpretations of membrane systems in the book) and feel very abstract, so I learn slowly, but I will keep working. In a conversation with JACK BAI, he proposed we can relate membrane computing to GOLANG considering the parallelism of membrane systems. I don't know much about GOLANG, just know that it has advantages in concurrent and asynchronous programming. Maybe we'll try and explore in this direction. Of course, I have learned that membrane computing has been applied in computer graphics, approximate optimization, cryptography, big data processing, and the development and research of relevant software and hardware for simulation membrane computing has also made progress.