Computation Models and Languages

1 Introduction 09/30

Today, we define the form to express the syntax.

- BNF
  - This is a form to write the syntax
  - How to express the natural number in BNF
- BNF can be interpreted strictly
  - recursive definition and induction principle

2 operatinal semantics 10/21

- Proof equivalence between BNF's definition as context free language and its definition as a recursive definition
- evaluational context and instruction
- How to evaluate int expressions, bool expressions, and branch expressions

3 Program and specification 10/28

We learned how to know whether a program moves as the creators' intentions. Creators' intentions are written as the specification of programs.

First, we interpret the specification into the mathematical expressions in order to make it easy to be discussed.

We have two ways to prove the correctness of the program. One is to use operational semantics. The other is to prove partial correctness and halting.

4 Hoare logic 11/07

Hoare logic is a set of the inference rules of the relationships, partial correctness and halting.

Hoare logic has soundness and relative completeness.

We exercise about how we construct a proof tree in Hoare logic.

5 Hoare logic soundness and completeness 11/11

Today, we proved the soundness and relative completeness of the Hoare logic.

Soundness is easy to be proved. We use structural recursion to each rule.

To prove completeness, first, we consider the weakest preconditions. If the weakest preconditions are written in the logical expressions, we can complete proof of the relative completeness.

6 Denotational semantics 11/13

consider programs as the functions from input to output.

Define the denotational semantics of the arithmetic expressions, boolean expressions, if, while, and recursive function.

7 lalmbda calculus 11/18

We define lambda calculus $M ::= x|\lambda x.M|M_1M_2$, which expresses the same thing as the Turing machines. In addition, we practice how to calculate it.

8 lambda calculus 11/25

we defined β redux and η redux. In addition, we learned how to express the set of natural numbers and their calculations, how to express pairs, and how to express the recursive functions.

9 lambda calculus and chrch rosser theorem 12/02

we recap how to express the recursive functions in lambda calculus. Then, we learned Chrch Rosser theorem. From Chrch Rosser theorem, we can say the uniqueness of the beta normal form. Also, we proof the Chrch Rosser theorem.

10 typed lambda calculus 12/09

First, we learned the difference between call by name and call by value. Then, we learned typed lambda calculus, which is lambda calculus with type information. Type is unique. If the type is decided, the reduction must finish.

11 type inference and polymorphism 12/16

we learned type inference, which bases on the algorithms using constraints. In addition, we learned the importance of polymorphism.

12 ML and Curry-Howard 12/23

we learned ML's type system. What kind of formulae are not accepted in ML and what is accepted. we also learned the Curry-Howard correspondence, which is the relationship between computer programs and mathematical proofs.

13 Logic programing 12/25

We learned prolog.