Lecture-5 Main Points

- λ notation for higher order functions
  - BHK interpretation involves higher order functions naturally as constructions.
  - Standard mathematics does not have adequate notation to deal with such functions.
  - We use λ notation invented by A. Church in λ calculus.
  - Meaning of λ abstraction.
  - All functions are functions of one variable only.
  - Free variables, bound variables and closed terms.
- Examples of constructions using $\lambda-$notation.
  - $\lambda x : A.\lambda y : B.x : A \to (B \to A)$
  - $\lambda x : A.\lambda y : \neg A.yx : A \to \neg\neg A$
  - $\lambda x : A \to B.\lambda y : \neg B.\lambda z : A.y(xz) : (A \to B) \to (\neg B \to \neg A)$
- Example of building the construction along the deduction tree.
  - $$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{Ax}{y : \neg B \vdash y : \neg B} \qquad \dfrac{\dfrac{Ax}{x : A \to B \vdash x : A \to B} \quad \dfrac{Ax}{z : A \vdash z : A}}{x : A \to B, z : A \vdash xz : B}}{x : A \to B, y : \neg B, z : A \vdash y(xz) : \bot}}{x : A \to B, y : \neg B \vdash \lambda z : A.y(xz) : \neg A}}{x : A \to B \vdash \lambda y : \neg B.\lambda z : A.y(xz) : \neg B \to \neg A}}{\vdash \lambda x : A \to B.\lambda y : \neg B.\lambda z : A.y(xz) : (A \to B) \to (\neg B \to \neg A)}$$
  - **Remarks**
    1. We have used function application of constructions in '$\to$e' rule.

2. For other elimination rules we need to introduce new syntactic constructs to manipulate constructions.

- Correspondence between logic and functional programming.

Minimal Logic	Functional Programming
Propositions	Types
Construction terms	Programs
Annotated Derivations	Type assignment
Proof Normalization	Program Evaluation

  - This correspondence is known as **Curry-Howard correspondence**.
  - Proof normalization is not part of this course. Last row of the table is shown for completeness.
  - 'Construction terms' refers to representation of constructions annotating propositions in a minimal logic derivation as shown in example above.
  - 'Construction terms' when simplified/evaluated yield 'canonical constructions' given in the definition of BHK interpretation.

- Main Theorem

  - **Theorem:** For every **Nm** derivable judgement $A_1, \dots, A_m \vdash A$, there is a construction which on input constructions $a_1, \dots, a_m$ (with $a_i : A_i$) outputs a construction of A.
  - **Proof:** The proof is by induction on the depth of judgement $A_1, \dots, A_m \vdash A$. Induction step considers various cases depending on the last rule in the derivation of judgement $A_1, \dots, A_m \vdash A$. We treat only one such case here, when the last rule applied in the derivation is $\vee$- elimination as shown below.

    $$\frac{\dfrac{\mathcal{D}_1}{A_1, \dots, A_j \vdash B \vee C} \qquad \dfrac{\mathcal{D}_2}{A_{j+1}, \dots, A_m, B \vdash A} \qquad \dfrac{\mathcal{D}_3}{A_{j+1}, \dots, A_m, C \vdash A}}{A_1, \dots, A_m \vdash A}$$

    Let c_k be the constructions obtained by I.H. on $\mathcal{D}_k$, for $1 \le k \le 3$.

    Following is the description of the desired construction.

    * Let input to this construction be $a_i : A_i$, for $1 \le i \le m$.
    * Evaluate $c_1 a_1 \dots a_i$, it is either $inl(t)$ or $inr(t)$.

    * In the first case our construction outputs $c_2\ a_{j+1} \dots a_m\ t$ in the second case it outputs $c_3\ a_{j+1} \dots a_m\ t$. $\square$

- As a corollary, we get that for any derivable judgement $\vdash A$ in minimal logic, A admits BHK interpretation.

- **Remarks**

    1. In proof of the main theorem, we have treated constructions as semantic idea rather than a concrete representation of it.
    2. In general, throughout the lecture, we have not made a clear distinction between constructions and representation of a construction. This distinction is important for treating evaluation of our construction terms (programs).