λ 計算の基本

1 導入

この講義の核心は、「計算とは関数適用の繰り返しであり、関数適用の実体は置換である」という点にある。

λ 計算は、変数・関数の定義・関数の適用という 3 つの概念だけで、すべての計算可能な操作を表現できる最小の計算モデルである。

2 中心課題

$(\lambda x.\lambda y.x)\ a\ b$ はどのように計算が進み、何に至るか。「計算が進む」とはどういうことか。

3 用語

- λ 計算: 変数・λ 抽象・適用の 3 規則で定義される計算モデル
- β 簡約: $(\lambda x.t)\ s \to t[x := s]$ という計算の 1 ステップ
- 簡約基: β 簡約が適用できる部分の形 $(\lambda x.t)\ s$
- 正規形: これ以上 β 簡約できない項
- η 変換: $\lambda x.(f\ x) =_{\eta} f$（$x \notin \mathrm{FV}(f)$ のとき）

4 方針

まず λ 項の構文（BNF）を確認する。つぎに β 簡約の規則を一段ずつ追う。最後に η 変換の意味（外延性）を加える。評価戦略（どの簡約基を先に選ぶか）はこの講義では扱わず、別講義に委ねる。

5 直感的な説明

$(\lambda x.t)\ s$ は「t の中の x を s で置き換える」ことで計算が一歩進む。これが β 簡約である。

Display

(λ x. λ y. x) a b

→β (λ y. a) b ← 外側の λ x に a を渡した（x を a に置換）

→β a ← 内側の λ y に b を渡した（y は出現しないので b は捨てられる）

「計算が進む」とは「簡約基を選び、置換を実行する」ことである。計算が止まるとは「正規形に到達する」ことである。

6 厳密な説明

6.1 1. λ 項の構文（再掲）

$$t ::= x \mid \lambda x.t \mid t_1\ t_2$$

6.2 2. β 簡約

$$\frac{}{(\lambda x.t)\ s \underset{\beta}{\to} t[x := s]}\ (\beta)$$

この規則は項のどこに現れる簡約基にも適用できる。

文脈での簡約: $t \underset{\beta}{\to} t'$ ならば

$$\frac{t \underset{\beta}{\to} t'}{t\ s \underset{\beta}{\to} t'\ s},\quad \frac{t \underset{\beta}{\to} t'}{s\ t \underset{\beta}{\to} s\ t'},\quad \frac{t \underset{\beta}{\to} t'}{\lambda x.t \underset{\beta}{\to} \lambda x.t'}$$

6.3 3. η 変換

$$\lambda x.(f\ x) \underset{\eta}{=} f \quad (x \notin \mathrm{FV}(f))$$

η 変換は「f と $\lambda x.(f\ x)$ は外延的に同じ関数だ」という主張である。どんな引数を渡しても同じ結果になるなら、「関数として等しい」とみなせる。

6.4 4. α・β・η のまとめ

操作	意味
α 変換	束縛変数の改名（意味を変えない）
β 簡約	関数適用の展開（計算の 1 歩）
η 変換	外延的同一性（関数の等価性）

7 最小の具体例

7.1 例 1: 恒等関数

$(\lambda x.x)\ a \underset{\beta}{\to} a$

$I = \lambda x.x$（恒等関数）と名前を付けると $I\ a \underset{\beta}{\to} a$。

7.2 例 2: 定数関数

$(\lambda x.\lambda y.x)\ a\ b$

$\underset{\beta}{\to} (\lambda y.a)\ b$

$\underset{\beta}{\to} a$

一番目の引数だけを返す関数。$K = \lambda x.\lambda y.x$ と書く（K コンビネータ）。
K combinator

7.3 例 3: 正規形に至らない項

$\Omega = (\lambda x.x\ x)\ (\lambda x.x\ x)$

$\underset{\beta}{\rightarrow} (\lambda x.x\ x)\ (\lambda x.x\ x) = \Omega$

Ω は β 簡約するとまた Ω に戻る。正規形が存在しない項の例である。型無し λ 計算では任意の計算が停止するとは限らない。

8 別の見方

λ 計算はチューリング機械と計算能力が等しい（チャーチ–チューリングテーゼ）。チューリング機械が「テープの状態を遷移させる」計算モデルであるのに対し、λ 計算は「式の変形規則の連鎖」という計算モデルである。前者は物理的な操作に近く、後者は数学的な書き換えに近い。

9 見分け方

- $(\lambda x.\ldots.)\ s$ という形が β 簡約基である
- $\lambda x.(f\ x)$ の形で x が f に現れないなら η 変換の対象
- 正規形かどうかは「β 簡約基が存在するか」で判定する

10 一言でいうと

λ 計算とは「変数・関数定義・関数適用」の 3 要素だけで計算を記述する体系であり、計算の 1 ステップは $(\lambda x.t)\ s \underset{\beta}{\rightarrow} t[x := s]$ という置換で与えられる。

11 関連リンク

→ 講義 **置換と α 同値の基本** lecture information programming-languages
https://study.bem130.com/lecture/information/programming-languages/foundation/置換と α 同値の基本-講義/

→ 講義 **β 簡約と η 変換の基本** lecture information programming-languages
https://study.bem130.com/lecture/information/programming-languages/lambda-calculus/β 簡約と η 変換の基本-講義/

→ 講義 **コンビネータ計算の基本** lecture information programming-languages
https://study.bem130.com/lecture/information/programming-languages/lambda-calculus/コンビネータ計算の基本-講義/