TaPL

omosan0627

August 21, 2023

3 型無し算術式

抽象構文? 帰納的定義・証明? 評価? 実行時エラーのモデル化?
Chapter5: 型無しラムダ、名前束縛、代入 Chapter8: 型システム、静的型付け
Chapter9: 静的型付けラムダ

3.1 導入

文法: 本書では BNF
構文: 項、値などの組かな.
項: 計算の構文的表現 (つまりメタ変数 t に代入することができる表現)
式: あらゆる種類の構文的表現 (項式、条件式など)??
メタ変数: メタ言語の変数. (対象言語の変数ではなく). 何らかの項のための placement holder
抽象構文:?
値: 項の部分集合で、評価の結果

3.2 構文

帰納的な項の定義: 推論規則を満たす最小の集合具体的な項の定義: 前提を持つ規則を 1 回適用した項を集める。それを有限回繰り返して得られる集合
完全帰納法は全て帰納ステップになっているとみなせる。

3.3 項に関する帰納法

構造的帰納法: 「各項 s に対して、s の任意の直接の部分項 r に対して $P(r)$ がなりたつとき、$P(s)$ が証明できる」ならば、全ての s に対して $P(s)$ が成立するこれは具体的な項の定義から証明できる. これは一般的な帰納法になっている.

3.4 意味論のスタイル

表示的意味論 (モデル理論っぽ)、公理的意味論 (ホーア論理とか?) もあるが、操作的意味論をこの本では扱う

3.5 評価

評価関係: 関係は (項, 項) の集合であることに注意.
インスタンス: 規則の結論や前提のメタ変数それぞれに対し、一貫して同じ項による置き換えを行ったものである.
規則がある関係によって満たされる: 規則の任意のインスタンスについて, 結論がその関係に属するか, または前提の内の一つが属さないことである.
1 ステップ評価関係:規則を満たす最小の二項関係. これは項の定義同様具体的に構成できるし、構造的帰納法も使える. 導出に関する帰納法と言う.
1 ステップ評価の決定性: Coq での証明ができません. 項 t が正規形:t $\rightarrow$ t' となる t' が存在しないとき. 全ての値は正規形である. 正規形は値とは限らず、そうでないとき行き詰まりという。状態実行時エラーの解析に使われるかも.
多ステップ評価関係 $\rightarrow^*$: 1 ステップ評価の反射的推移的閉包. つまり有限回の 1 ステップ評価で到達できる項の関係. これも推論規則から定義できます.
停止尺度: 評価の停止性の証明で使われる関数のこと. 項について単調減少. 構文要素ってなんだ?

型無しラムダ計算

基礎

一般的な手続き、関数を引数に値を渡すことで具体化する.
ラムダ計算: 全てが関数. 変数、ラムダ抽象、関数適用の 3 種類の項のみで構成.

抽象構文

具象構文: プログラマが直接読み書きする文字列抽象構文: 単純な内部的表現. ラベル付き木 (抽象構文木 AST) として表現される.
字句解析器: 文字列-¿トークン列
構文解析器: トークン列-¿抽象構文木
優先順位や結合法則を決めておくと、括弧を減らせる。
本書では抽象構文木を念頭に置くが、関数適用は左結合、ラムダ抽象の本体はできるだけ右に展開する.
$\lambda x.\lambda y.xyx$ は $(\lambda x.(\lambda y.(xy)))x$ と解釈することができるが、$\lambda x.(\lambda y.((xy)x))$ と解釈することにする。
変数とメタ変数の違いをここからは意識する必要がある. ただ文脈から大体わかるね.

スコープ