

# The Definition of Snail Programming Language

letexpr

2020 年 7 月 16 日

## 1 はじめに

Snail は静的型付けの関数型プログラミング言語である.

主な特徴として,

- Bounded Linear Type によるリソースの制御
- Effect System / Coeffect System (未実装)
- 軽量の依存型 (indexed type) (未実装)

が挙げられる.

本文では Snail について Core 言語を定義し, Core 言語への脱糖規則, Core 言語の型付け規則, 操作的意味論を定義することにより Snail に定義を与える.

本文中ではメタ変数として以下のようなものを用いる.

- $\Gamma, \Delta, \Theta \dots$  型環境上を動くメタ変数.
- $A, B, C \dots$  型の上を動くメタ変数.
- $Ac, Bc, Cc \dots$  コンストラクタ上を動くメタ変数.
- $x, y, z \dots$  変数上を動くメタ変数.
- $p, q, r \dots$  resource semiring 上を動くメタ変数.
- $e \dots$  項の上を動くメタ変数.
- $s \dots$  文字列リテラル上を動くメタ変数.
- $n \dots$  自然数上を動くメタ変数.
- $i \dots$  整数上を動くメタ変数.
- $f \dots$  小数上を動くメタ変数.
- $b \dots$  論理値リテラル上を動くメタ変数.

## 2 Snail の構文定義

EBNF 記法を用いて Snail の具象構文を以下に示す.

```
toplevel ::= let [rec] x {y [ : <type>]] : <type> = <term> {<mutual-recursion-let>}  
          | typedef A = [ | ] {<Ac [of <type>]] | } <Bc [of <type>]] {<mutual-recursion-type>}
```

```
mutual-recursion-type ::= and A = [ | ] {<Ac [of <type>]] | } <Bc [of <type>]]
```

```
type ::= <type> → <type>  
       | ! '[' <expmod> ']' '{' <type> '}'  
       | <type> <type>  
       | '(' <type> ')'   
       | A
```

```
expmod ::= n | ∞
```

```
pattern ::= <pattern> <pattern>  
          | <pattern> binop <pattern>  
          | '(' <pattern> ')'   
          | x | Ac  
          | list      (組み込みリストの構文糖衣)  
          | [ ]  
          | -
```

```
mutual-recursion-let ::= and x {y [ : <type>]] : <type> = <term>
```

```
term ::= <term> <term>  
       | let [rec] x {y [ : <type>]] : <type> = <term> {<mutual-recursion-let>} in <term>  
       | fun {x [ : <type>]] → <term>  
       | match <term> with [ | ] {<pattern> → <term> | } <pattern> → <term>  
       | if <term> then <term> else <term>  
       | fix x.<term>  
       | '(' <term> [ : <type>] ')'   
       | ! <term>  
       | i | f | s | b | x | Ac | [ ] | list
```

### 3 Snail の Core 言語

Snail の Core 言語は Snail のプログラムを脱糖する事により得ることができる.

#### 3.1 Core 言語の構文

Core 言語は次のような構文を持つ.

$$\begin{aligned}
 e ::= & \text{let } !x = e_1 \text{ in } e_2 \\
 & | \text{ i } | \text{ f } | \text{ s } | \text{ b } | \text{ x } | !e \\
 & | \text{ match } e \text{ with } \{ \text{pat} \rightarrow e \mid \} \text{ pat} \rightarrow e \\
 & | e_1 e_2 \mid \lambda x. e \mid \text{fix } x. e
 \end{aligned}$$

$$\text{pat} ::= \text{pat}_1 \text{ pat}_2 \mid x$$

#### 3.2 Core 言語の型システム

Core 言語の型付け規則を次に示す.

##### 3.2.1 型付け規則

|   |          |   |       |
|---|----------|---|-------|
| $\frac{}{\vdash i : \text{Int}}$                                    | (INT)    | $\frac{\Delta \vdash e : B \quad \Gamma <: \Delta}{\Gamma, \Theta \vdash e : B}$                                      | (SUB) |
| $\frac{}{\vdash f : \text{Float}}$                                  | (FLOAT)  |   |       |
| $\frac{}{\vdash s : \text{String}}$                                 | (STRING) | $\frac{\Gamma, x : A \vdash e : B}{\Gamma \vdash \lambda x. e : A \multimap B}$                                       | (ABS) |
| $\frac{}{\vdash b : \text{Bool}}$                                   | (BOOL)   | $\frac{\Gamma \vdash e : A \multimap B \quad \Delta \vdash e' : A}{\Gamma + \Delta \vdash e e' : B}$                  | (APP) |
| $\frac{}{x : A \vdash x : A}$                                       | (ID)     |   |       |
| $\frac{[\Gamma] \vdash e : B}{r \star [\Gamma] \vdash !e : !_r B}$  | (PR)     | $\frac{[\Gamma], x : [A]_p \vdash e : A \quad 1 + p \star q \preceq q}{q \star [\Gamma] \vdash \text{fix } x. e : A}$ | (FIX) |
| $\frac{\Gamma, x : A \vdash e : B}{\Gamma, x : [A]_1 \vdash e : B}$ | (DER)    | $\frac{\Gamma \vdash e : !_r A \quad \Delta, x : [A]_r \vdash e' : B}{\Gamma + \Delta \vdash e e' : B}$               | (LET) |

### 3.2.2 アルゴリズム的型付け規則

### 3.2.3 部分型付け規則

$$\frac{}{A <: A} \quad (\text{O-I})$$

$$\frac{A <: B \quad q \preceq p}{!_p A <: !_q B} \quad (\text{O-B})$$

$$\frac{A' <: A \quad B <: B'}{A \multimap B <: A' \multimap B'} \quad (\text{O-L})$$

$$\frac{A <: B \quad q \preceq p}{[A]_p <: [B]_q} \quad (\text{O-D})$$

$$\frac{}{\Gamma <: \Gamma} \quad (\text{O-IC})$$

$$\frac{\Gamma <: \Delta \quad A <: B}{\Gamma, x : B <: \Delta, x : A} \quad (\text{O-C})$$

## 4 参考文献