

# パラメタライズドモナドによる依存アップデートの形式化に向けて

池守 和槻 叢 悠悠 増原 英彦 (東京工業大学)

## 背景

- Idris … 依存型を持つ純粋関数型言語
- プログラムの実行前後のエフェクトを表す型を
  - 静的に検査できる
  - 異なる型に更新できる (強アップデート<sup>[1])</sup>)
  - 返回值に依存して更新できる (依存アップデート<sup>[2])</sup>)

fの返回值の型    実行前の状態の型    実行後の状態の型

```
f : Eff () [STATE Int] [STATE Bool]
f = do
  x ← get      - Int型の状態の値をxに束縛
  let y = x < 10
  putM y       - 状態をBool型の値yで更新
```

(Idrisにおける強アップデート)

## IdrisのEffectsライブラリ

gの返回值の型

実行前の状態の型  
長さがn, 要素の型がIntのリスト

返回值okに依存した実行後の  
状態の型

```
g : Eff Bool [STATE $ Vect n Int]
  (\ok => if ok then [STATE $ Vect n Int]
         else [STATE $ Vect (S n) Bool])
g = do
  xs ← get      - Vect n Int型の状態の値をxsに束縛
  case length xs > 1 of
    True => pureM True - 状態の更新はしない
    False => do
      let ys = True::(map (\x => x < 10) xs)
      putM ys - Vest (S n) Bool型の値に状態を更新
      pureM False
```

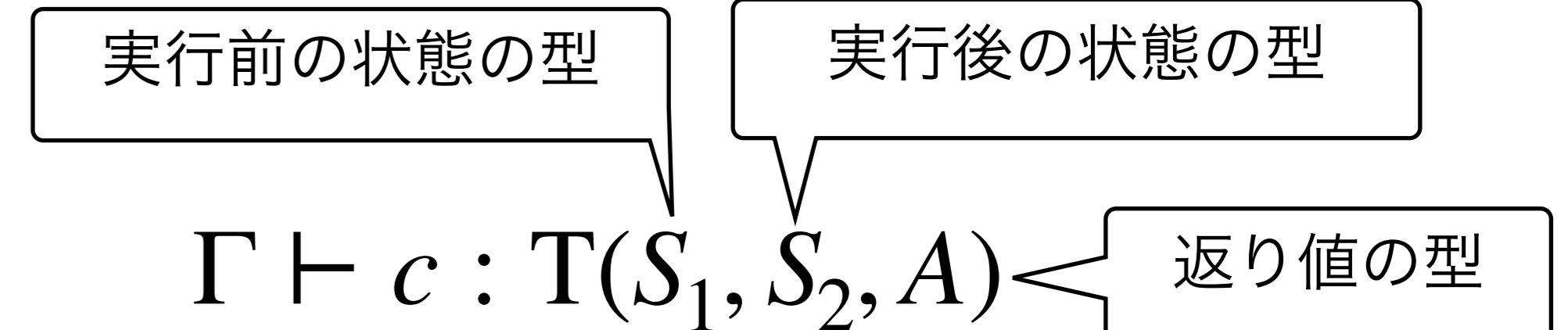
(Idrisにおける依存アップデート)

## パラメタライズドモナド

- パラメタライズドモナドのHaskellにおける型クラスの定義

```
class PMonad (T :: * -> * -> * -> *) where
  return :: a -> T s s a
  bind   :: T s1 s2 a -> (a -> T s2 s3 b) -> T s1 s3 b
```

- パラメタライズドモナド(T, return, bind)の型判断



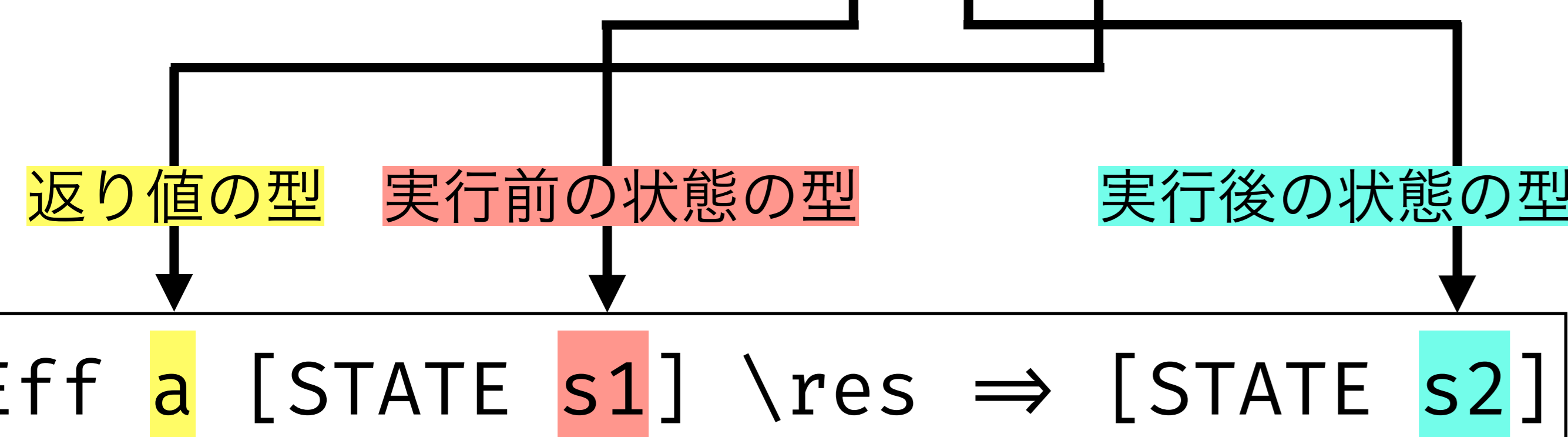
強アップデートの形式化が可能<sup>[3]</sup>

## 類似点と相違点

### 類似点

- 型の対応関係

$$\Gamma \vdash c : T(S_1, S_2, A)$$



### 相違点

- Idrisのエフェクトライブラリでは実行前後の状態の型
  - s1, s2を依存型にできる
  - s2が返回值resに依存できる

## 目標

### Idrisの依存アップデートの形式化

## アイデア

パラメタライズドモナド + 依存型

= Idrisの依存アップデート ?

## 参考文献

1. A. Ahmed, M. Fluet, and G. Morrisett. L<sup>3</sup> : A Linear Language with Locations. Fundamenta Informaticae, 2007.
2. E. Brady. Programming and reasoning with algebraic effects and dependent types, IICFP'13, 2013.
3. R. Atkey. Parameterised notions of computation. Journal of Functional Programming, 2009.

## 今後の課題

パラメタライズドモナド + 依存型の形式化

- 依存性に対応した型判断の設計

A型の項xに依存した型

$$\Gamma \vdash c_1 : T(S_1, S_2, A) \quad \Gamma, x : A \vdash c_2 : T(S_2, S_3, B)$$

$$\Gamma \vdash \text{let } x = c_1 \text{ in } c_2 : T(S_1, S_3, B)[c'/x]$$

T(S1, S2, A)型の項c1からA型の項c'を得る必要がある!

- 健全性の証明