

# 計算効果の導入

勝股 審也

国立情報学研究所

2022/08/30



# 略歴

## 研究分野

- プログラミング言語の意味論 (=数理モデル)
- ソフトウェア検証の数学的基盤

## 略歴

- エジンバラ大学理 工学部 PhD (2005)
- 数理解析研究所 助手/助教 (2004-2016)
- 国立情報学研究所 特任准教授 (2017~)  
ERATO 蓮尾メタ数理プロジェクト 研究総括補佐  
同 メタ理論的統合グループ リーダー

## 計算効果 (computational effect) とは

... プログラムを実行することで起きる様々な作用や現象の総称  
私が論文をいくつか調べたところ

- (~1997 年) side effect (副作用)、effect、monadic effect 等
- (1998 年~) computational effect が使われる
  - ▶ Philip Wadler: The Marriage of Effects and Monads. ICFP 1998: 63-74
  - ▶ Carsten Führmann: Direct Models for the Computational Lambda Calculus. MFPS 1999: 245-292

追加情報お待ちしています

これにない、本講演も最初は副作用、後に計算効果に移行

# 副作用(計算効果)とは具体的には?

## 副作用のないプログラムの例: ユークリッドの互除法 (Java)

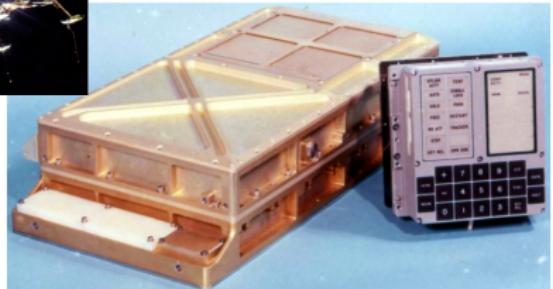
```
int gcd(int x, int y) {  
    if (y == 0) return y;  
    else return gcd(y, x % y);  
}
```

## 副作用のあるプログラムの例: 足し算ゲーム (Basic)

すぐやみつる、こんにちはマイコン、p.171 より

```
10 INPUT A  
20 INPUT B  
30 C = A + B  
40 INPUT D  
50 IF C = D THEN PRINT "セイカイ"; GOTO 80  
60 PRINT "マチガイ"  
70 GOTO 40  
80 END
```

副作用を起こすプログラムは至るところで動いている



左: 着陸船イーグルとアポロ誘導コンピュータ (1966-)



# 計算効果の種類

## 出力と入力 (python)

```
print("Hello world!")  
val = input()
```

## 疑似乱数 (python)

```
rand = random.random()
```

## 無限ループ (python)

```
while(true):  
    x = x + 1
```

## 例外 (python)

```
raise ZeroDivisionError
```

他にも大域・局所メモリ、名前生成、確率的選択、非決定的選択、…

# 計算効果: 継続

プログラムの実行のある時点における継続とは、**その時点以降にやるべきこと**

10:00 - 10:30	オープニング、計算効果の導入
10:30 - 12:00	限定継続を使った計算効果プログラミング
12:00 - 13:30	昼休み
13:30 - 15:00	計算効果の推論技術
15:00 - 15:30	休憩
15:30 - 17:00	計算効果の数理モデル

継続をデータとして扱えるとジャンプや計算の復帰が可能になる

## 継続制御命令 (scheme)

```
(call/cc (lambda (k) ...))
```

- ⇒ 様々な計算効果の実現
- ⇒ 古典論理との関連

# 計算効果を確認するには?

int f(void) に計算効果はあるか?

以下の二つの関数の動作に差があるならば、f は計算効果を持つ

```
void disc1() {           void disc2() {  
    int y = f();          return;  
    retrun;              }  
}  
注意: 差が無い場合は f に計算効果があるかはわからない
```

- int f() {printf("hello"); return 0;}
- int f() {while(true){x = x + 1;};}

# 計算効果を確認するには?

int f(void) に計算効果はあるか?

以下の二つの関数の動作に差があるならば、f は計算効果を持つ

```
int copy1() {  
    int y1 = f();  
    int y2 = f();  
    printf("%d, %d", y1, y2);  
}  
  
int copy2() {  
    int y = f();  
    printf("%d, %d", y, y);  
}
```

注意: 差が無い場合は f に計算効果があるかはわからない

- int f() {printf("hello"); return 0;}
- int f() {return rand();}
- int f() {while(true){x = x + 1;}}

# 計算効果の表現と理解に対する取り組み

- 計算効果をどのように実装するか?
- 計算効果のあるプログラムをどのように検証するか?
- 多岐にわたる計算効果をどのようにモデルするか?

## 計算効果を理解・記述し、検証・分析するための様々な概念

モナド、継続、CPS変換、エフェクトシステム、ホーア論理、分離論理 (separation logic)、線形型、オーナーシップ型、モデルチェックング、…

計算効果には数多くのトピックがあり、様々なアプローチで研究されている。本サマースクールではそれらのいくつかを紹介したい。

- (叢先生) 継続による実行フローの制御と計算効果のプログラミング
- (関山先生) 計算効果のあるプログラムに関する推論
- (勝股) モナドによる計算効果の数学的モデル

# 理論 $\iff$ 実装: Haskell のモナド型

Moggi のモナドによる計算効果のモデルと Haskell への応用

- Eugenio Moggi: Computational Lambda-Calculus and Monads. LICS 1989: 14-23
- Eugenio Moggi: Notions of Computation and Monads. Inf. Comput. 93(1): 55-92 (1991)
- Philip Wadler: Comprehending Monads. Math. Struct. Comput. Sci. 2(4): 461-493 (1992)

モナドの型クラス (Haskell)

```
class Monad m where
  (»=) :: m a -> (a -> m b) -> m b
  return :: a -> m a
```

Haskell のモナドを用いて作られたシューティングゲーム

- Monadius <https://hackage.haskell.org/package/Monadius>

## 継続の理論的研究

- Matthias Felleisen: The Theory and Practice of First-Class Prompts. POPL 1988: 180-190
- Olivier Danvy, Andrzej Filinski: Abstracting Control. LISP and Functional Programming 1990: 151-160
- John C. Reynolds: Definitional Interpreters for Higher-Order Programming Languages. High. Order Symb. Comput. 11(4): 363-397 (1998) 他多数

## 実装

- OchaCaml  
(<http://pllab.is.ocha.ac.jp/~asai/OchaCaml/demo/>)
- Scala (shift-reset; continuations plugin)
- OCaml (shift, control, shift0, control0; delimcc library)
- Haskell (control0; GHC proposal)
- Racket (さまざまな限定 / 非限定継続命令; control package)

# 理論 $\iff$ 実装: ハンドラー

## 例外処理 (Python)

```
try ...  
except ZeroDivisionError:  
    return 0
```

## 代数的エフェクトとエフェクトハンドラーの理論

- Gordon D. Plotkin, John Power: Adequacy for Algebraic Effects.  
FoSSaCS 2001: 1-24
- Gordon D. Plotkin, Matija Pretnar: Handlers of Algebraic Effects.  
ESOP 2009: 80-94 他

## 実装

- OCaml5 で予定あり
- Idris (<https://docs.idris-lang.org/en/latest/effects/index.html>)
- Haskell の extensible effect library  
(<https://hackage.haskell.org/package/extensible-effects>)
- Effekt language, Koka, Frank, Eff, ...