

7章: エラーの推定

ANDREAS ZELLER 著 中田秀基 監訳

デバッグの理論と実践

-なぜプログラムはうまく動かないのか-

学籍番号 60206299

氏名 吉富 楓雅

自動デバッグテクニック(再掲)

今後の章の紹介

- ・入力の単純化
- ・プログラムスライス
- ・状態の観察
- ・状態の監視
- ・アサーション
- ・異常
- ・原因と結果の連鎖
- ・ミスから学ぶ



目次

1. 値の起源の特定
2. 制御フローの理解と依存関係の追跡
3. プログラムスライス
4. 静的解析の限界
5. まとめ

今回は
「プログラムスライス」
についてです

値の起源の特定



起こってはいけない問題が発生

プログラムの実行時に
何が起こり得る・起こり
得ないかを推測



プログラルスライス

障害発生!!!



KernighanとPikeの言葉

ありえない現象が起こった.

確かなのは, 事象が実際に起こったという事実だけだ.

そこで, 事象が起こった理由を見つけるために, 今, 目の前にある結果から,
過去に遡って考えなければならない.

制御フローの理解 依存関係の追跡



C言語でフィボナッチ数列

```
#include <stdio.h>
```

```
int fib(int n)
```

```
{
```

```
    int f, f0 = 1, f1 = 1;
```

```
    while (n > 1) {
```

```
        n = n - 1;
```

```
        f = f0 + f1;
```

```
        f0 = f1;
```

```
        f1 = f;
```

```
    }
```

```
    return f;
```

```
}
```

```
int main()
```

```
{
```

```
    int n = 9;
```

```
    while (n > 0)
```

```
    {
```

```
        printf("fib(%d) = %d¥n", n, fib(n));
```

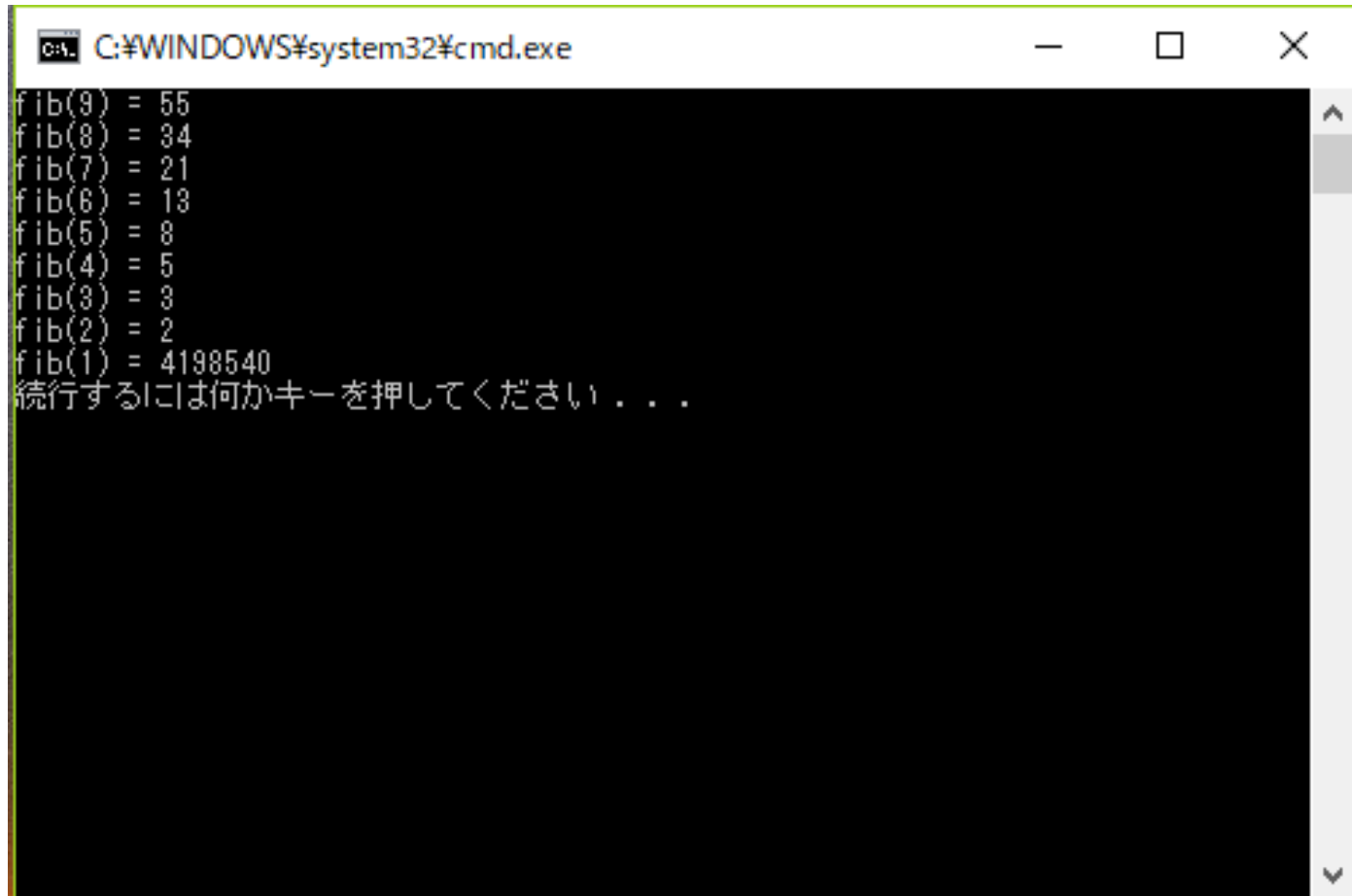
```
        n = n - 1;
```

```
    }
```

```
    return 0;
```

```
}
```

実行結果



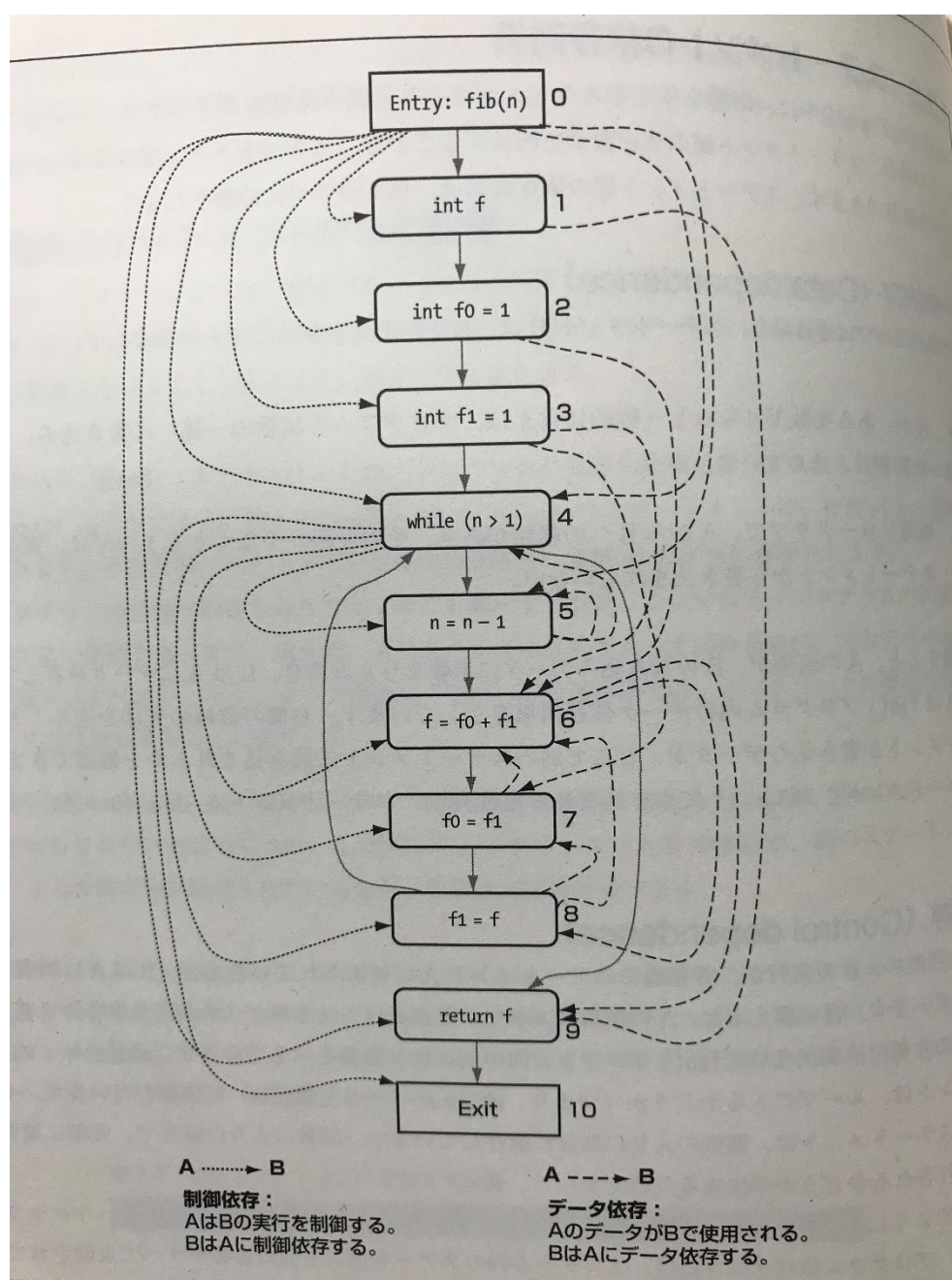
```
C:\WINDOWS\system32\cmd.exe
fib(9) = 55
fib(8) = 34
fib(7) = 21
fib(6) = 13
fib(5) = 8
fib(4) = 5
fib(3) = 3
fib(2) = 2
fib(1) = 4198540
続行するには何かキーを押してください...
```

制御フローグラフ

1	Entry: fib(n)
2	int f
3	int f0 = 1
4	int f1 = 1
5	while (n > 1)
6	n = n - 1
7	f = f0 + f1
8	f0 = f1
9	return f
10	Exit

プログラムスライス





fib()のプログラム 依存グラフ

プログラムスライスとは

プログラムスライス：

プログラムまたはプログラムの実行のうち、障害に関係する部分を切り取るテクニック

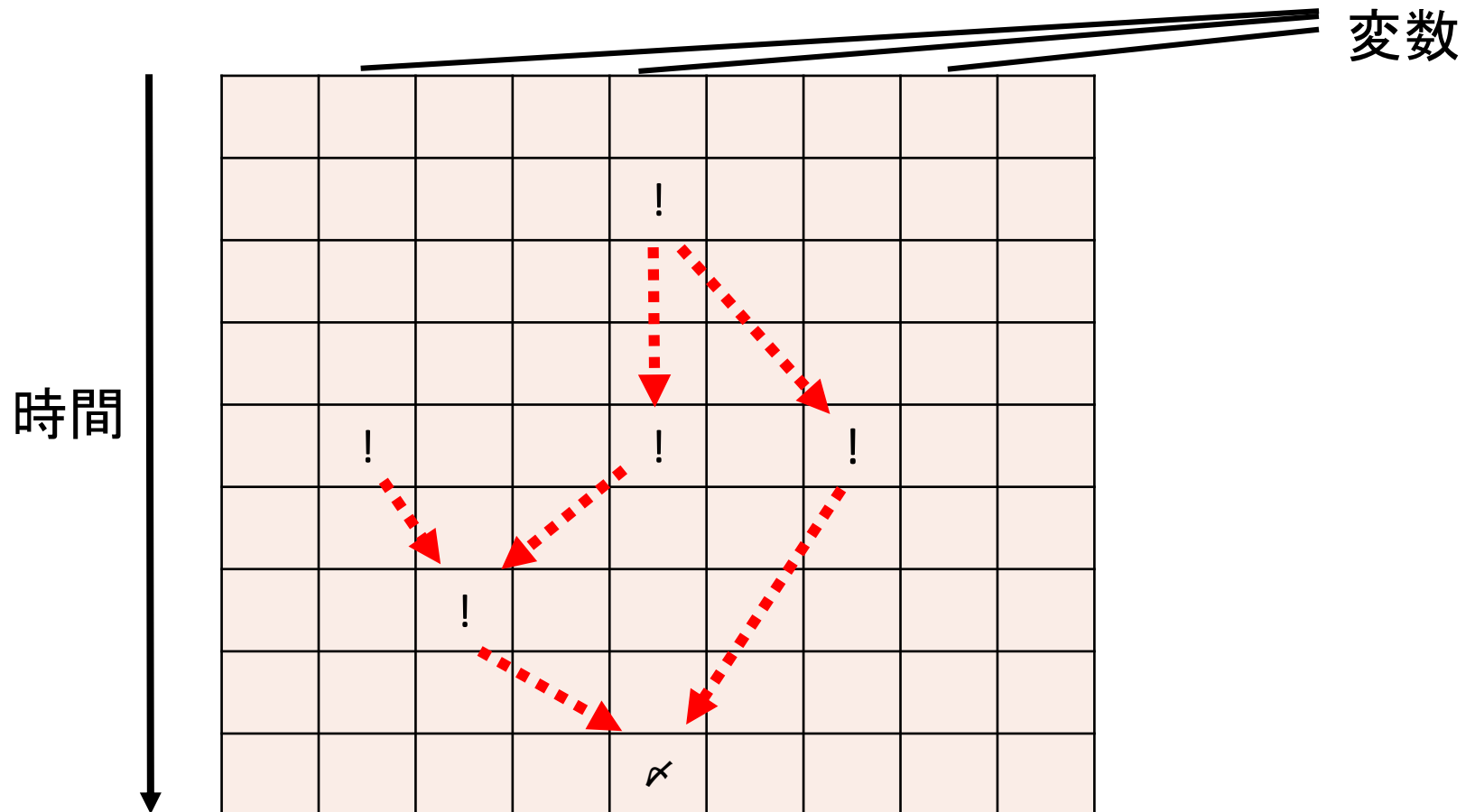
・スライス = サブセット

サブセット：

あるステートメントに影響を与えた部分またはあるステートメントから影響を受ける部分

・スライシング = スライスを抽出する動作

プログラムのスライスのイメージ



プログラムスライスの種類

- ・ **フォワードスライス** $S^F(A)$:

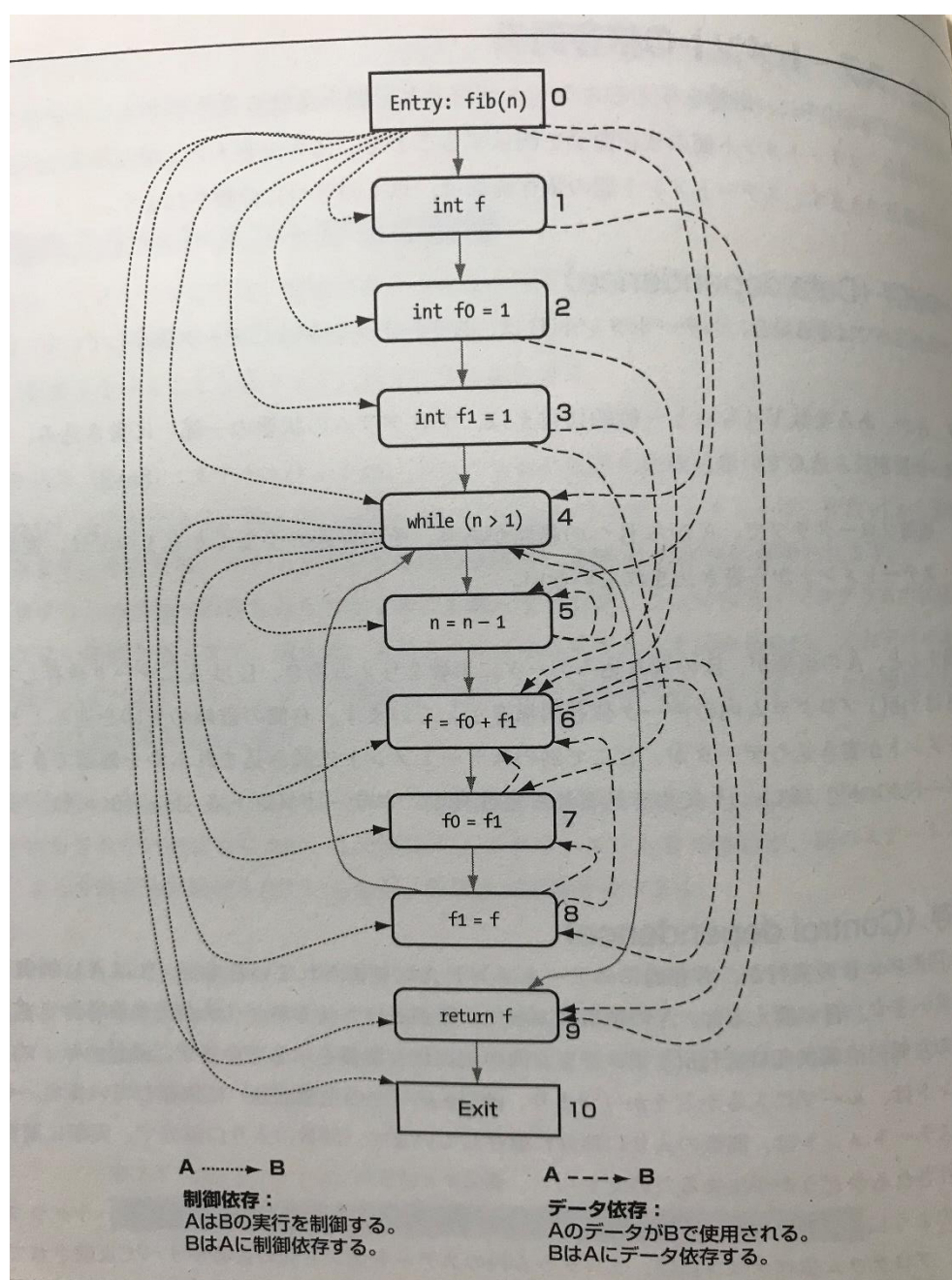
$$S^F(A) = \{ B \mid A \rightarrow^+ B \}$$

とあるステートメントAからすべての依存関係を追跡する

- ・ **バックワードスライス** $S^B(B)$:

$$S^B(B) = \{ A \mid A \rightarrow^* B \}$$

依存関係に沿って, プログラムコードを後ろから前へ辿る



※再掲

フィボナッチ数列を例に

fib()のプログラム依存グラフについて

- ・ステートメント2についてフォワードスライス考えると

$$S^F(2) = \{2, 6, 7, 8, 9\}$$

- ・ステートメント9についてバックワードスライス考えると

$$S^B(9) = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

スライスの操作

- ・ **チョップ (chop)** :

フォワードスライスとバックワードスライスの論理積

フォワードスライスの起点が、バックワードスライスの起点にどのような影響を与えるのか

- ・ **バックボーン (backbone)** :

2つのバックワードスライスの論理積

アプリケーション内で複数の値の算出に関与した箇所を知る

- ・ **ダイス (dice)** :

2つのバックワードスライスの差分

ある変数のバックワードスライスが別のものとどう違うのかを知る

プログラムの無関係な部分の除去

プログラム全体

```
1 int main() {  
2   int a, b, sum, mul;  
3   sum = 0;  
4   mul = 1;  
5   a = read();  
6   b = read();  
7   while (a <= b) {  
8     sum = sum + a;  
9     mul = mul * a;  
10    a = a + 1;  
11  }  
12  while(sum);  
13  write(mul);  
14 }
```

バックワードスライス $S^B(13)$

```
1 int main() {  
2  
3  
4   mul = 1;  
5   a = read();  
6   b = read();  
7   while (a <= b) {  
8  
9     mul = mul * a;  
10    a = a + 1;  
11  }  
12  
13  write(mul);  
14 }
```

プログラムの無関係な部分の除去

バックボーン $S^B(12) \cap S^B(13)$ ダイス $S^B(13) \times S^B(12)$

```
1  int main() {  
2  
3  
4  
5    a = read();  
6    b = read();  
7    while (a <= b) {  
8  
9  
10       a = a + 1;  
11    }  
12  
13  
14 }
```

```
1  int main() {  
2  
3  
4    mul = 1;  
5  
6  
7  
8  
9       mul = mul * a;  
10  
11  
12  
13    write(mul);  
14 }
```

怪しいコードの例

- 未初期化の変数の読み込み
- 未使用の値
- メモリリーク
 - : 実行中のプログラムが確保したメモリ領域の開放を
忘れたまま放置してしまうこと
- インタフェースの誤使用
- Nullポインタ

参考：C言語におけるメモリ領域開放

```
int func()
{
    int err = 0;

    int *ptr = NULL;  // 関数の先頭で確保
    ptr = (int *)malloc(sizeof(int) * 1024);
    if (!ptr) err = 1;
    if (!err) { ... } // 関数の内容1
    if (!err) { ... } // 関数の内容2
    if (ptr) {free(ptr), ptr = NULL;} // 関数の最後で開放
    return err;
}
```

静的解析の限界



チューリングの停止性問題

- ・FingBugsなどのツールで, 実際の偽陽性検知率は50%

プログラミング言語には数多くの構成要素が存在し, これらの要素が正確なデータ依存関係の検出を難しくする



チューリングの停止性問題

静的解析の限界を示す例

- ・奇数の完全数の存在を確認するプログラム

```
int x;
```

```
for(i = j = k = 1; --j || k; k = j ? i % j ? k : k - j : (j = j + 2));
```

```
write(x);
```

- ・完全数：すべての正の約数の合計がその数自身と等しい数

(例) $28 = 1 + 2 + 4 + 7 + 14$

- ・現在, 奇数の完全数の存在は証明されていない

参考：チューリングの停止性問題とは

・1936年にアラン・チューリングによって証明された解決不可能問題の1つ

「任意のプログラムに1つの入力値を与えて実行した場合、このプログラムは最終的に停止するか？停止せずに永遠にループするか？」



このチューリング機械の存在を仮定すると「自身が停止すると判定したならば無限ループを行い、停止しないと判定したならば停止する」別のチューリング機械が構成でき、矛盾

まとめ



まとめ

- ・プログラムスライスとは, 障害に関係する部分を切り取るテクニック
- ・2種類のプログラムスライス
- ・静的解析には停止性問題がある