IS

情報システム学科
Department of Information Systems

研究室　Database & Software Engineering研究室　　カテゴリー　IT・IoT・AI・ロボティクス
教員名　深海　悟

モデル検査技術とその応用ー論理構造物の検証技術

1. 背景

物理構造物(例えば建築)の場合		論理構造物 (ソフトウェア)の場合
意匠設計はともかく 構造設計＆検証は 物理学＋解析学がベース	VS.	機能設計（アプリケーション領域に依存）はともかく ソフトウェアの振舞いは数理論理学や組合せ数学に基づいて設計・検証が可能のはず。しかし、現実の設計・開発の現場ではあまり・・・ 経験とカンだけで建築構造物、電気回路、機械を設計するようなもの

⇒ Formal Methods（数理論理学や組合せ数学に基づくソフトウェア設計・検証の方法論）が必要

2. モデル検査とは？

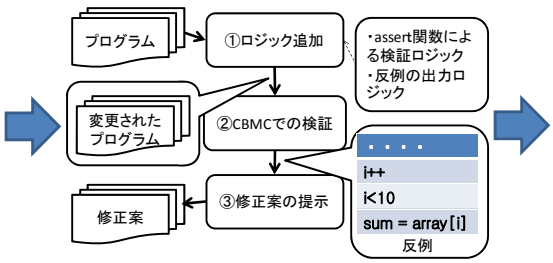
Formal Methodsの一種で、システムの状態遷移が指定された性質を満たすか否かを網羅的にチェックする技術
性質を満たさない場合、満たさない具体的な状態遷移系列（反例）を提示してくれるのが特徴
状態遷移に帰着できる問題なら何でも(論理回路、ソフトウェアの仕様、プログラム、プロトコル・・・)検証可能
検証できる性質
- Safety Property(好ましくない状態に到達することはないか)
- Liveness Property(好ましい状態に何時かは到達できるか)

3. モデル検査の応用　C言語プログラムの不具合修正方法の自動提案

C言語プログラムを解析し、配列の範囲外参照を引き起こす可能性がある場合、モデル検査システムが出力する反例を解析することによって、どうすればこれを回避できるかの修正方法を提案してくれるシステムを開発した。
モデル検査ツールとしては、既存のC言語検査ツールであるCBMCを利用している。

不具合を含むプログラム

```
#include<stdio.h>
#define NUM 10
int sortArray(int a[NUM]){
  int *ptr = a;
  int i,j;
  for(i=0;i<NUM-1;i++){
    for(j = NUM ;j>i;j--){
      if(a [ j ] < a [ j - 1 ]){
            int t=a[j];
            a[j]=a[j-1];
            a[j-1]=t;
      }
    }
  }
  return 0;
}
```



開発したシステム

出力された修正提案①

```
for(i=0;i<NUM-1;i++){
  for(j = NUM-1 ;j>i;j--){
```

出力された修正提案②

```
if(j < NUM && a [ j ] < a [ j - 1 ]){
    int t=a[j];
    a[j]=a[j-1];
    a[j-1]=t;
}else{
    break;
}
```

配列の範囲外参照 - 2つのタイプ
① いきなり範囲外を参照する場合（上記の例）
② 範囲内から範囲外に飛び出す場合

```
L1: #define NUM 10
L2: int main(void){
L3: int array[NUM];
L4: int *p;
L5: for(p = array; *p != 0; p++){
L6:         scanf( "%d" , p);
L7: }
…
```

```
int array[NUM];
int *p;
for(p = array; p < array + NUM -1
    && *p != 0; p++){
        scanf( "%d" , p);
}
```

今後改善が必要な事項
(1) 配列の参照式が四則演算を含む数式になっている場合

本システムでは検証できないケース
(1) 無限ループ内で繰り返しに依存しない不具合が存在する場合
(2) 検証対象の配列がプログラム引数になっている場合

本システムはプログラムの仕様を見ているわけではないので、提案された修正案が仕様に照らして正しい保障はない。しかし検証実験によれば多くの場合に正しい提案を含む複数の提案を出力してくれることがわかっている。