Java 検証器 Regnant の帰還

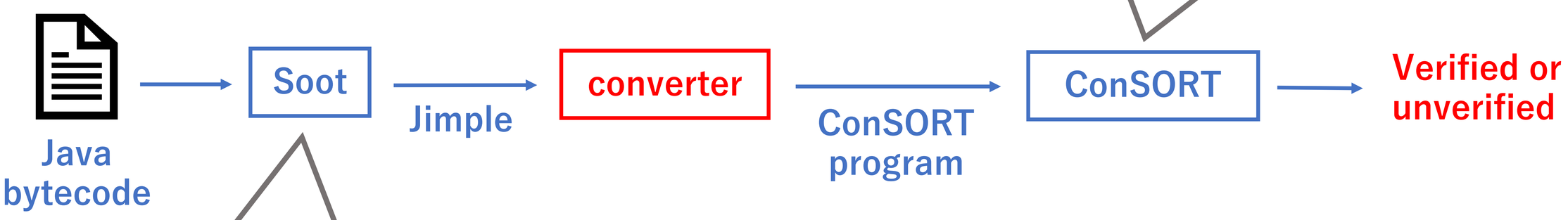
小林 亮太（M1）　五十嵐 淳　末永 幸平
京都大学

- Java のための篩型に基づく検証器 Regnant を実装
- Java バイトコードから ConSORT [Toman et al. ’20] プログラムに変換することで検証

ConSORT

- 分割可能所有権と篩型を用いた，ポインタのある手続型プログラムの検証器

アプローチ



Soot

- Java のための解析・最適化フレームワーク
- Jimple は Soot で用いられる中間言語の1つ

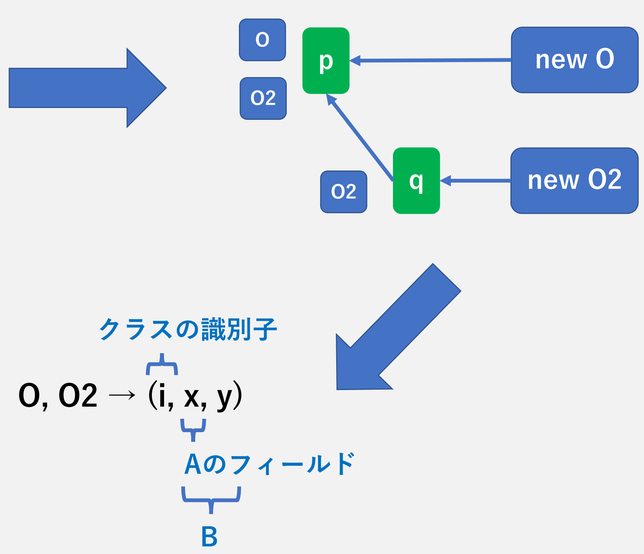
アイデア

- **オブジェクトをタプルへ変換**
  - Java におけるオブジェクトは、クラスの識別子とフィールドを要素として持つタプルに変換
  - プログラムから pointer assignment graph を生成し、その情報からタプルの形を決定するため、継承も扱える

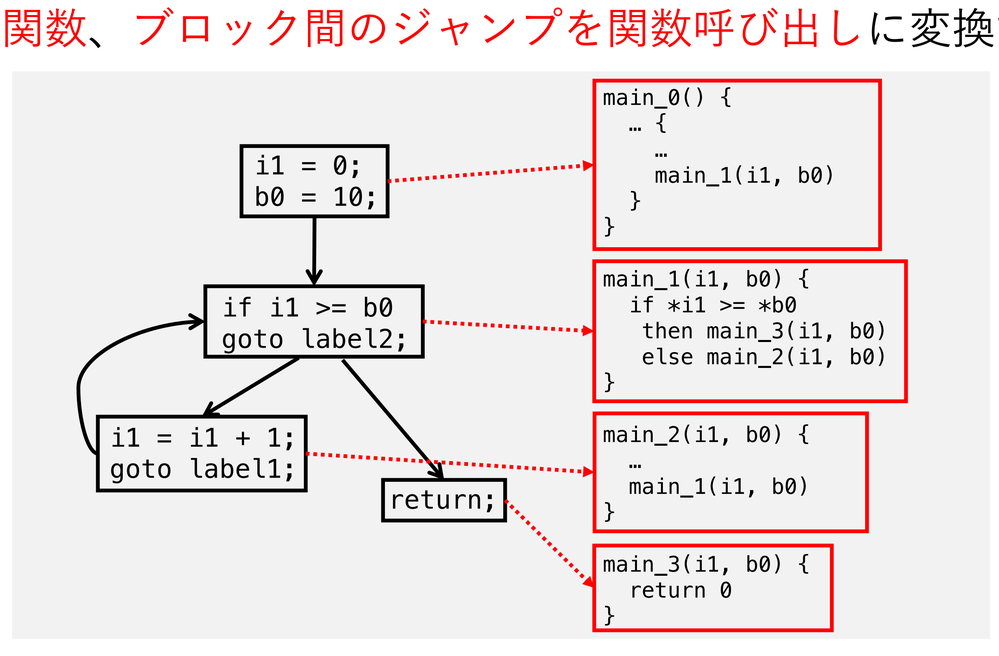
```
public class Inheritance {
   public static void main(String[] args) {
      O p = new O();
      O2 q = new O2();
      p = q;
      …
   }
}


class O {
   int x;
   …
}

class O2 extends O {
   int y;
   …
}
```



- **基本ブロックを関数へ変換**
  - ConSORT プログラムにはループ構文が存在しないため、Jimple コードを基本ブロックに分解し、基本ブロックを関数、ブロック間のジャンプを関数呼び出しに変換する



検証例

```
public class CallMethod {
    public static int abs(int x) {
    if (x < 0) {
      return -1 * x;
    } else {
      return x;
    }
  }

  public static void main(String[] args) {
    int x = -3;
    x = 1 + 2 * abs(x);
    assert(x == 7);
  }
}
```

Safe となる例

```
import java.util.Random;

public class RandomObj {
  public static void main(String[] args) {
    int x = new Random().nextInt();
    O p = new O(x);
    assert (p.isPos());
  }
}

class O {
  int x;

  O(int x) {
    this.x = x;
  }

  boolean isPos() {
    return x > 0;
  }
}
```

Unsafe となる例

今後の課題

- アノテーションの自動挿入
- 検証の高速化