スマートコントラクトのためのファジングのHoare論理による効率化

大倉功士　末永幸平
京都大学

ブロックチェーン上に記録され，自動的に契約を実行するコード
課題：コード内にバグがあると悪用され続ける
一度デプロイするとコード変更が困難
よって、デプロイ前の検証が重要

ソフトウェアのランダムテスト手法で，最近スマートコントラクト検証でよく用いられる
ランダムに入力を与え，挙動を観察
✕ランダム入力に頼るため，希少なバグだと発見が困難

本研究：ファジングを希少なバグでも見つけられるように効率化

従来のファジング：入力空間とバグに行き着く空間に大きなギャップが存在
提案手法：入力空間を**Hoare論理**を用いて狭め，バグに到達しやすくする
1. cを性質φの成立を検査する命令assert(φ)を含む命令列とする
2. c'をc中のassert(φ)をassert($\neg\varphi$)に変換した命令列とする
3. c'の事前条件をHoare論理を用いて求め，それにより入力空間を狭める

$\neg\varphi$に行き着く入力を見つけるのがファジングの役割

Hoare論理を用いたファジングの例

```
int64 x;
function f1() public{
  if (x < 50) {
    x = x * 2;
  } else{
    x = x - 10;
  }
}
function f2() public{
  if (x < 70){
    x = x + 50;
  } else{
    x = -x;
  }
}
function f3() public{
  if (x < -100 || 150 < x) {
    x = x / 2;
  } else {
    x = -x;
  }
}
```

・*f1, f2, f3*はif文を含み，*x*を書き換える関数
・*f1*(); *f2*(); *f3*();を順に実行後，*x*の範囲をテストする
テスト内容は，assert($x < 50 \,||\, 80 < x$)とする

・*f1*(); *f2*(); *f3*(); assert($50 \leq x \leq 80$);の事前条件を求める
1. assert($50 \leq x \leq 80$);の事前条件は，
$$50 \leq x \leq 80$$
2. これを事後条件としてもつ*f3*();の事前条件は，
$$\big((x < -100 \,\|\, 150 < x) \wedge (100 \leq x \leq 160)\big) \vee ((-100 \leq x \leq 150) \wedge (-80 \leq x \leq -50))$$
まとめると，
$$(-80 \leq x \leq -50) \vee (150 < x \leq 160)$$
弱化(範囲を緩める)を行い，
$$-80 \leq x \leq 160$$
3. *f2*();と*f1*();に関しても同様に行うと，
$$-65 \leq x \leq 90$$
入力範囲をこれに狭める.

通常のファジングの場合の入力範囲は
$-2^{63} \leq x \leq 2^{63} - 1$

ファジング実行結果

・通常のファジング

テスト実行回数	36	32	117	45	66

・Hoare論理＋ファジング

テスト実行回数	12	1	12	3	9

テスト試行回数が減った！

まとめと今後の課題

まとめ
・スマートコントラクトにおけるファジングは、入力空間とバグに行き着く空間に差がある
・今回はHoare論理を用いて入力空間を狭めた

今後の課題
・Hoare論理の健全性証明
・実装