- This slide was
  - a material for the "Reading PLDI Papers (PLDIr)" study group
  - written by Kazuhiro Inaba (www.kmonos.net), under my own understanding of the papers published at PLDI
    - So, it may include many mistakes etc
- For your correct understanding, please consult the original paper and/or the authors' presentation slide!

k.inaba (稲葉 一浩), reading:

PLDIr #12
Mar 12, 2011

paper written by J. Kodumal and A. Aiken
(PLDI 2004)

The Set Constraint/CFL Reachability Connection in Practice

解きたい問題（の例）

「tainted とマークされた値が
untainted マークの変数に入らない」の静的検証

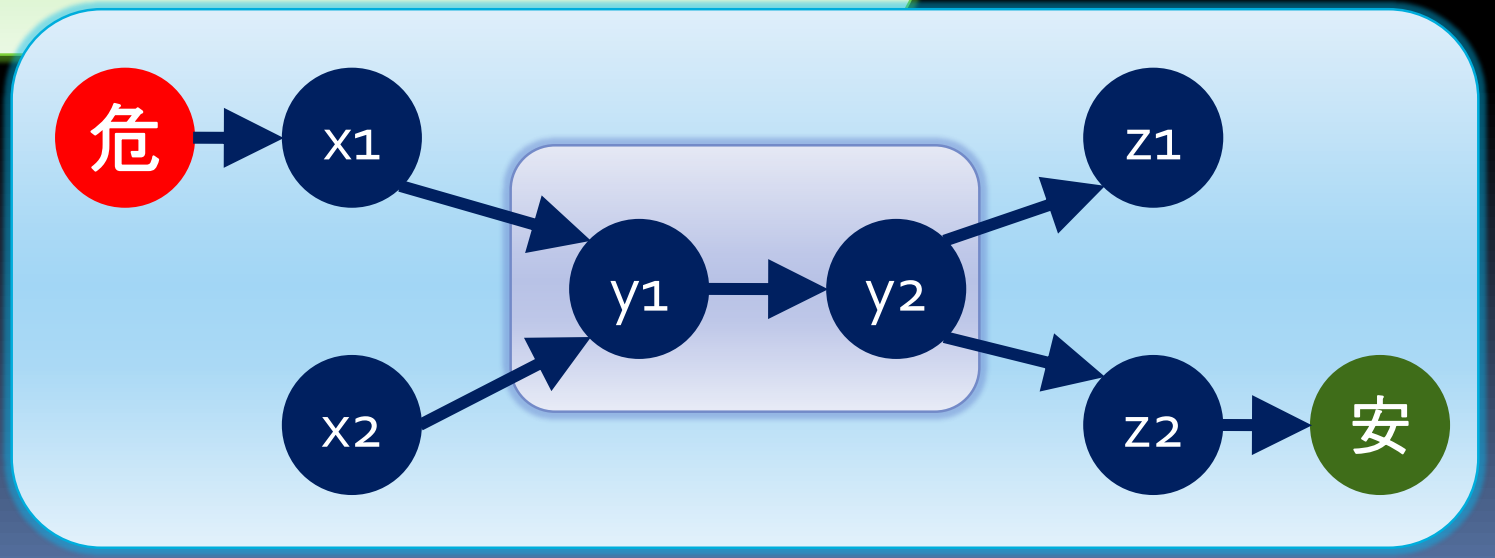
```
int id(int y1) { int y2 = y1; return y2; }

int main(void)
{
        tainted int   x1;
        int           z1, x2;
        untainted int z2;
        z1 = id(x1); // call site 1
        z2 = id(x2); // call site 2
}
```

典型手法:
グラフの到達可能性問題と見なす

```
int id(int y1){int y2=y1; return y2;}
int main(void) {
        tainted int   x1;
        int           z1, x2;
        untainted int z2;
        z1 = id(x1); // call site 1
        z2 = id(x2); // call site 2
}
```

危 から

安 に行ける？



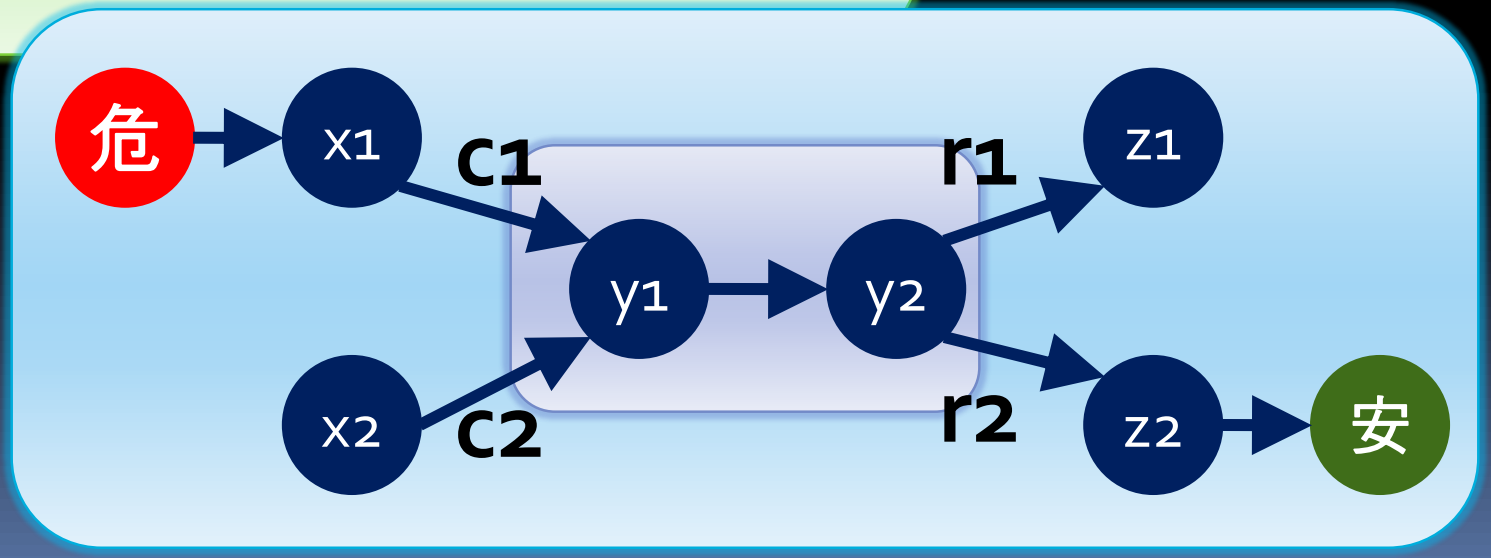
Betterな精度の典型手法:
グラフのCFL到達可能性問題と見なす

```
int id(int y1){int y2=y1; return y2;}
int main(void) {
        tainted int   x1;
        int           z1, x2;
        untainted int z2;
        z1 = id(x1); // call site 1
        z2 = id(x2); // call site 2
}
```

危 から

安 に

c1r1 | c2r2 で行ける？



CFL Reachability を解く典型手法: “Set Constraint” 問題に帰着

- CFL Reachability の計算量
  - O(|文法|^3 |グラフ|^3)
    - CYK構文解析 + Warshall-Floyd 到達可能性
  - 多項式時間だけど実用には厳しい重さ

- ヒューリスティックス Solver のある問題に帰着 ➔ “Set Constraint” 問題

“Set Constraint” 問題

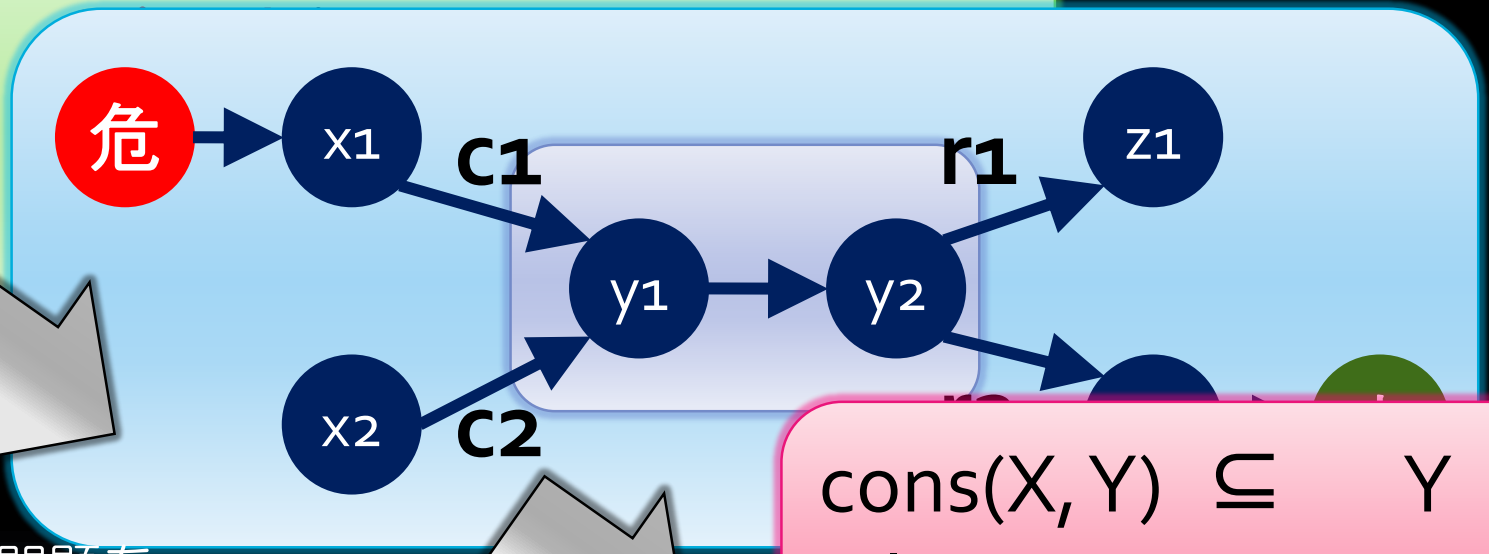
こんな連立方程式を解く問題。

cons(X, Y)	⊆	Y
nil	⊆	Y
suoɔ 1(Y)	⊆	one

- 集合Xの要素とYの要素をconsしたらYに入る
- nilというアトムは集合Yに入る
- 集合Yのconsの形の要素の第一要素はone

既存のやり方の流れ

```
int id(int y1){int y2=y1; return y2;}
int main(void) {
}
```



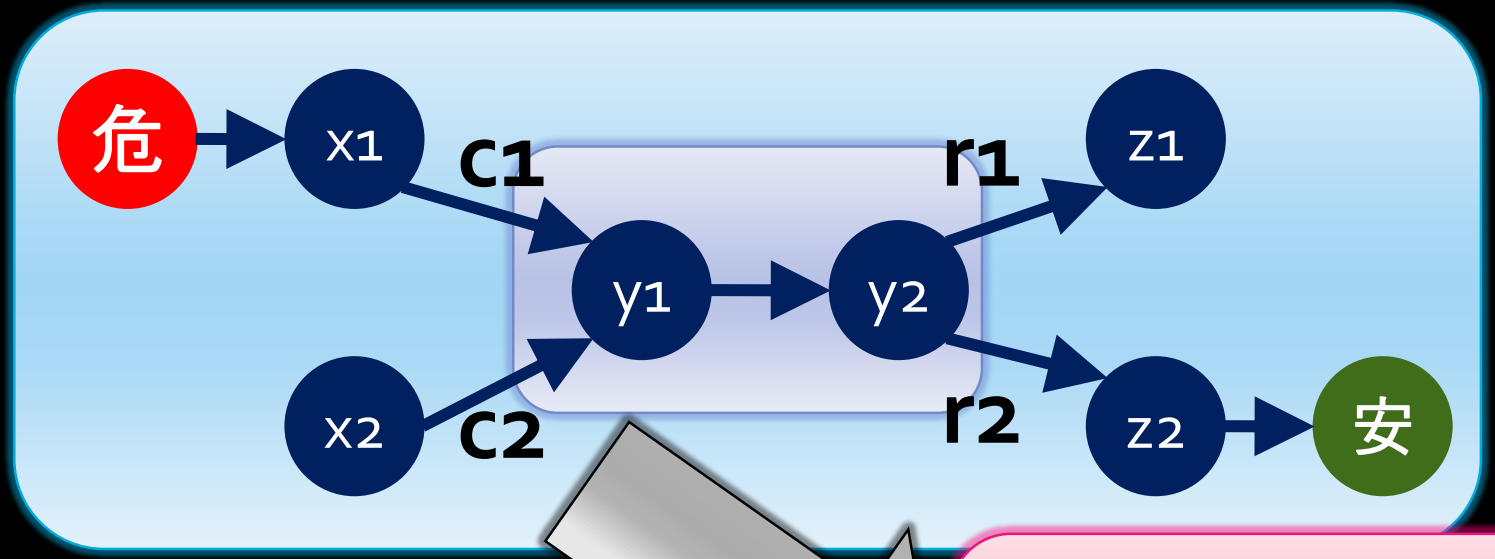
解析の問題を
CFL Reachability に

CFLReachability を
Set Constraint に

$$cons(X, Y) \subseteq Y$$
$$nil \subseteq Y$$
$$suoɔ\ 1(Y) \subseteq one$$

解く

問題点

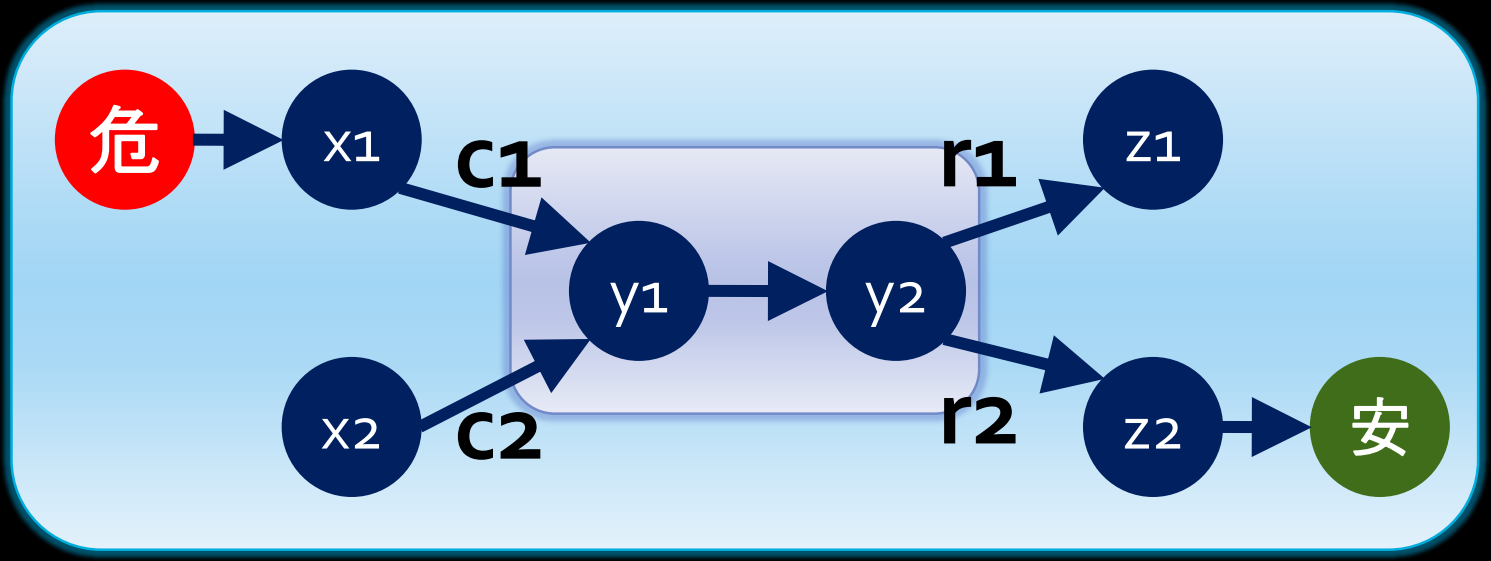


cons(X, Y)	⊆	Y
nil	⊆	Y
suoɔ 1(Y)	⊆	one

CFL Reachability を
Set Constraint に
[Melski&Reps 97]

← まだ遅い

観察



- 一般の CFLReach を解きたいわけじゃない
- プログラム解析から現れるような CFLReach が解ければよい
- “Call-Ret の対応が取れてる” を表す文法の CFLReach が解ければ十分では？

この論文のやったこと: “DyckCFL” に特化した帰着法

- k-DyckCFL
  - S ::= P*
  - P ::= $(_1$ S $)_1$ | $(_2$ S $)_2$ | … | $(_k$ S $)_k$
  - 「対応のとれた括弧の列」

- tbw

結果

- 漸近計算量
  - O(|文法|^3 |グラフ|^3) ➔ O(|文法| |グラフ|^3)