Project
SNU 4541.664A

Kwangkeun Yi

- 이 프로젝트의 목적은 강의에서 다루는 정적분석 이론(요약해석 abstract interpretation)을 사용해서 구체적인 정적분석기를 디자인하고 구현해보는 것이다.
- 앞으로 숙제들은 이 프로젝트를 실현해가는 과정이 될 것이다.

대상언어 D

D라고 부르기로한 아래의 언어를 생각하자. 정수입력을 받고, 변수의 주소도 값으로 다룰 수 있도록 했고, 타겟이 실행중에 계산되는 점프명령이 있다.

$$
\begin{array}{rcl}
C & \to & x \texttt{ := } E \mid \texttt{*}x \texttt{ := } E \\
& \mid & C \texttt{ ; } C \\
& \mid & \texttt{if } E\ C\ C \\
& \mid & \texttt{repeat } C\ E \\
& \mid & \texttt{goto } E \\
E & \to & n \quad (n \in \mathbb{Z}) \\
& \mid & E \texttt{ + } E \\
& \mid & E \texttt{ * } E \\
& \mid & \texttt{-} E \\
& \mid & E \texttt{ < } E \\
& \mid & x \mid \texttt{*}x \mid \texttt{\&}x \\
& \mid & \texttt{readInt} \quad \text{(정수입력)}
\end{array}
$$

점프명령어 "`goto` E"에서 식 E를 계산하면 점프할 명령어(C)의 번호가 된다. 그 명령어로 점프한다. 프로그램의 모든 명령어마다 고유의 자연수가 명령어 번호로

붙는다고 가정한다. 고유 번호 ℓ이 붙은 명령어 $\ell : C$는 다음과 같이 만들어진다:

$$
\begin{array}{rcl}
C & \rightarrow & \ell : C' \\
C' & \rightarrow & x \texttt{ := } E \mid \texttt{*}x \texttt{ := } E \\
& \mid & C \texttt{ ; } C \\
& \mid & \texttt{if } E\ C\ C \\
& \mid & \texttt{repeat } C\ E \\
& \mid & \texttt{goto } E
\end{array}
$$

이 프로젝트에서는 D 프로그램을 정적분석하는 "watercheck"(watermark checker)이라는 분석기를 디자인하고 구현하게 된다.

분석기 워터첵(watercheck)의 배경과 목표

Natural이라는 회사는 D 프로그램 자동생성 서비스를 제공한다. 이 서비스에서 생성하는 D 프로그램에는 Natural AI가 자동 생성한 코드임을 알 수 있는 고유의 식별 코드 조각이 포함되있다.

> 그 식별 코드가 실행중에 하는 일은, 어느 지점에서는 늘 어떤 변수에 특정한 성질의 정수가 저장되어 있는 것이다. 그 성질이란 250401(Natural설립일)로 나누어 나머지가 항상 123이라는 것이다.

주어진 D 프로그램이 Natural AI의 작업인지 아닌지를 확인하려면 간단치가 않다. 프로그램의 모든 입력의 경우마다 일일이 돌려보고 그런 성질의 변수가 실행중에 어느 지점에서 발생하는지 찾아내야 하는데, 이 과정이 비용이 많이 들거나 심지어는 아예 불가능하게 된다. 입력의 경우가 무한하거나 너무 많은 경우도 그렇고, 프로그램이 사용하는 많은 변수들의 값들을 실행을 통해서 매순간 일일이 추적해야 하는 어려움도 그렇고.

한 방법은 정적분석 기술을 이용하는 것이다. 정적분석기술을 이용하면 주어진 소프트웨어가 어느 지점에서 위와 같은 변수를 가지고 있는지를 자동으로 안전하게 판단할 수 있게 된다. 정수 변수들이 실행중에 가지는 값들이 250401로 나누어 어떤 나머지 값들을 가지는지를 추적하면 된다.

예를들어, 정적분석기는 아래의 D 프로그램을 분석해서 Natural AI가 만든 코드인지를 확인해 줄 수 있어야 한다. 변수 `xp`는 프로그램의 8번줄이 끝나면 늘 250401으로 나누어 나머지가 123이기 때문이다.

```
1:            port = 12; len = 20; pos = 139;
```

```
2:            fd = 11; xp := 0; i = &xp;
3:            index = 15;
4:            repeat
5:                 len = len + port;
6:                 pos = pos + (*i);
7:                 xp = *i + port + fd + 501;
8:                 *i = *i + 250000;
9:                 if (pos < index) (xp = xp * 2) (index = index + (-1));
10:           (pos < index)
```

□