와이드닝을 조금 더 똑똑하게 해보기

2014. 07. 29
ROSAEC Workshop
ROPAS 김솔

와이드닝의 목표 : 빨리 가는 것 & 도착하는 것

```
1: int i = 0, buf[15]
2: while (...)
3:     ...
4:     i++
5: buf[i]
```

와이드닝 없이 : [0,0] > [0,1] > [0,2] > .. > ??

일반적인 와이드닝 : [0,0] > [0, +oo)

와이드닝의 목표 : 빨리 가는 것 & 도착하는 것

```
1: int i = 0, buf[15]
2: while (...)
3:     ...
4:     i++
5: buf[i]
```

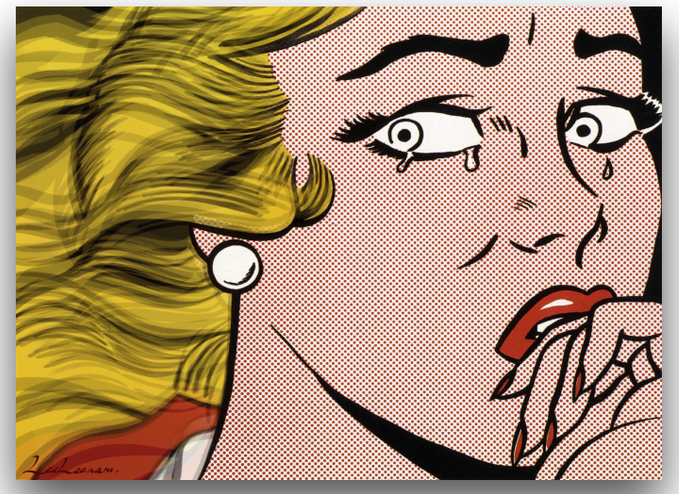
와이드닝 없이 : [0,0] > [0,1] > [0,2] > .. > ??

일반적인 와이드닝 : [0,0] > [0, +oo)

딜레이를 이용한 와이드닝 : [0,0] > [0,1] > [0,2] > [0, +oo)

Threshold를 이용한 와이드닝 : [0,0] > [0, 8] > [0,17] > [0, +oo)

멈추어야 할 곳을 지나쳐 버릴 때가 있습니다.



No threshold Threshold

```
1: int i = 0, buf[15]
2: while (...)      // [0, +oo)  [0, 10]
3:    if (i == 10)
4:       break      // [10, 10]  [10, 10]
5:    else { }      // [0, +oo)  [0, 9]
6:    i++
7: buf[i]           // [0, +oo)  [0, 10]
```

멈추어야 할 곳을 지나치는 경우는
크게 두 종류로 분류할 수 있습니다.

종류 1 : 조건문을 통해 반복문을 빠져나오는 경우

종류 2 : 반복되면서 연산하다보면 스스로 안정되는 값들

종류 1. 조건문을 통해 반복문을 빠져나오는 경우

```
                          No threshold Threshold
1: int i = 0, buf[15]
2: while (...)      // [0, +oo)  [0, 10]
3:     if (i == 10)
4:         break    // [10, 10]  [10, 10]
5:     else { }     // [0, +oo)  [0, 9]
6:     i++
7: buf[i]           // [0, +oo)  [0, 10]
```

종류 2. 반복되면서 연산하다보면 스스로 안정되는 값들

```
                                        No threshold Threshold
1: int i = 0, buf[6] = {1,2,3,1,2,3}
2: while (...)
3:     i = buf[i]    // [0, +oo)  [0, 3]
```

알람 및 시간 차이

No threshold Threshold

프로그램	알람	시간	알람	시간	알람 차이	시간 차이
gzip	444	1.40	433	2.12	11	0.72
bc	610	5.75	586	80.22	24	74.47
chess	2081	5.13	1496	9.04	585	3.91
grep	988	1.84	986	2.18	2	0.34
tar	1292	5.88	1284	8.01	8	2.13
less	1718	53.41	1716	63.38	2	9.97
make	2136	35.25	2136	42.14	0	6.89
euler	4479	18.89	4473	20.32	6	1.43
wget	1518	11.73	1518	15.44	0	3.71
screen	7634	816.80	7622	858.39	12	41.59
fluidsynth	1192	3.57	1192	4.89	0	1.32

알람 및 시간 차이

줄어든 알람

약 2.7%

늘어난 분석 시간

약 15%

연구를 통해 보이고 싶은 것

- 와이드닝은 생각보다 정확도를 크게 떨어트리지 않는다.
- 하지만 와이드닝을 대충 해서 인해 생겨난 허위 알람들은 와이드닝을 덜 대충 하는 방법 이외에는 없앨 수 있는 방법이 없다.
- Threshold 값들을 잘 뽑아서 사용하면 적은 비용으로 와이드닝으로 인해 생긴 허위 알람들을 없앨 수 있다.