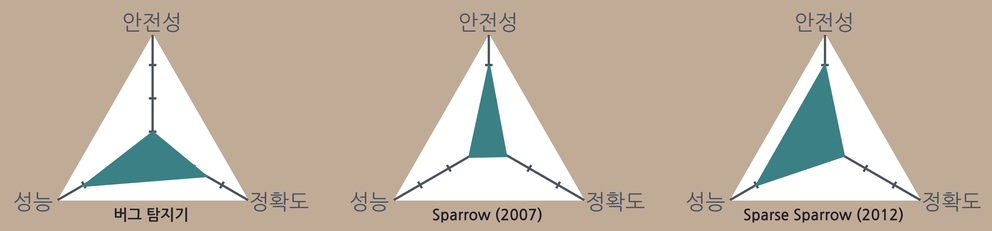
부분적으로 함수 호출 문맥을 구분하는 분석

오학주[1] 이원찬[1] 양홍석[2] 이광근[1]

[1]서울대학교 [2]University of Oxford

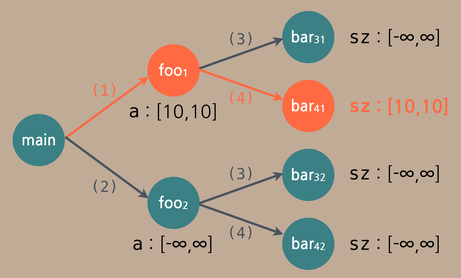
동기

★ 안전하게, 큰 프로그램을, 정확하게 분석하기 어렵다.



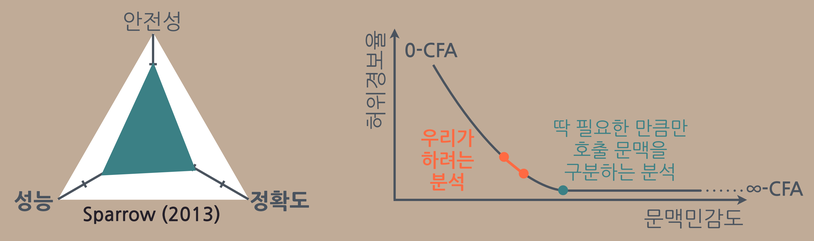
★ 정확도를 위해 함수 호출 문맥을 일괄적으로 구분하면 (k-CFA) 비효율적이다.

```
char * bar(int sz)
{
  return malloc(sz);
}

char * foo(int a)
{
  char *x = bar(*);      // (3)
  return bar(a);         // (4)
}

int main()
{
  char *ptr1 = foo(10); // (1)
  *ptr1 = 0;            //(p1)
  char *ptr2 = foo(*);  // (2)
  *ptr2 = 0;            //(p2)
}
```



- 1-CFA 이하는 (p1)에서 허위경보를 냄
- 2-CFA는 필요 이상으로 호출 문맥을 많이 구분

★ 비용이 크지 않으면서 정확한 분석을 하자!



★ 함수 호출 문맥을 구분하면 정확해질 경로만 구분하여 분석하자!

목표

결과

★ 7개의 GNU 프로그램에 대한 실험

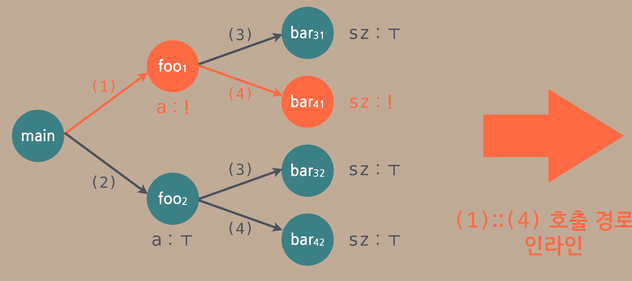
프로그램	호출 문맥 구분 X		부분적으로 호출 문맥 구분				Δ경보	Δ시간	Δ크기
	경보	본분석	경보	전분석	본분석	구분된 호출수			
spell	58	0.6	30	0.3	0.9	25/124 (20.2%)	48%	1.8x	1.47x
bc	606	15.6	483	6.5	15.3	29/777 (3.7%)	20%	1.4x	1.03x
tar	940	43.8	799	11.8	43.5	56/1,218 (4.6%)	15%	1.3x	1.05x
less	654	131.1	561	11.9	184.7	59/1,522 (3.9%)	14%	1.5x	1.22x
make	1,500	89.3	1,002	20.3	124.2	85/1,050 (8.3%)	33%	1.6x	1.20x
wget	1,307	72.0	905	29.9	126.1	111/1,973 (5.6%)	30%	2.2x	1.74x
a2ps	3,682	125.0	2,004	205.3	343.6	263/2,450 (10.7%)	46%	4.4x	1.93x

➥ 함수 호출를 **일부만 구분해도**(7%) 허위경보를 **크게 줄일 수 있다**(34%).

★ 전분석으로 요약값의 정확도를 보존하는 (잃어버릴 수 있는) 경로를 식별

- 요약 도메인: $Var \to \hat{V}_{\text{pre}}$ $\quad \hat{V}_{\text{pre}} = \top$ 부정확한 값 / ! 정확한 값 / $\bot$
- 전분석은 본분석에 대해 안전: $\gamma(\text{lfp}F) \subseteq \gamma_{\text{pre}}(\text{lfp}F_{\text{pre}})$
- 전분석 요약 도메인은 배분법칙이 성립: $F_{\text{pre}}(X \sqcup Y) = F_{\text{pre}}(X) \sqcup F_{\text{pre}}(Y)$
  ➥ 모든 호출 문맥을 구분하는 분석을 효율적으로 구현 가능 (RHS 알고리즘)

★ 식별된 경로에 나타난 함수 호출을 모두 인라인 후 분석



- 정확도를 보존하는 경로: **(1)::(4)**
- 정확도를 잃는 경로: (1)::(3),(2)::(3),(2)::(4)

```
char * bar(int sz)
{
  return malloc(sz);
}

char * foo(int a)
{
  char *x = bar(*);      // (3)
  return bar(a);         // (4)
}

int main()
{
  char *x = bar(*);
  char *ptr1 = malloc(10);
  *ptr1 = 0;             //(p1)
  char *ptr2 = foo(*);   // (2)
  *ptr2 = 0;             //(p2)
}
```

방법