경보 메시지간의 종속 관계를 안전하게 계산하는 방법

김유일, 이우석
서울대학교

오류 정적 분석기의 거짓 경보

표: Airac 분석 사례

분석 대상	소스 크기	경보 수	오류 수
Software 1	109,878	64	1
Software 2	17,885	18	9
Software 3	3,254	57	0
Software 4	29,972	140	112
Software 5	19,263	100	3
Software 6	36,731	48	4
Software 7	138,305	147	47
Software 8	233,536	162	6
Software 9	47,268	273	1

Y. Jung et al., “Taming False Alarms from a Domain-Unaware C Analyzer by a Bayesian Statistical Post-Analysis, SAS’05.

해결 방안은?

중요한 경보를 먼저 보여주는 방법

더 정확한 분석을 적용하는 방법

중복되는 경보를 감추는 방법은 없을까?

연결된 오류 경보

```
position_set grps[256];

MALLOC(grps[ngrps].elems, position, d->nleaves);
grps[ngrps].nelem = 1;
grps[ngrps].elems[0] = pos;
```

A code snippet from Grep 2.5.1

```
while (*optarg && *optarg >= '0' && *optarg <= '9')
    val = val * 8 + *optarg++ - '0';
```

A code snippet from Wu-ftpd 2.6.2

복잡하게 연결된 오류 경보

```
position_set grps[256];

MALLOC(grps[ngrps].elems, position, d->nleaves);
grps[ngrps].nelem = 1;
ngrps++;
...
ngrps--;
grps[ngrps].elems[0] = pos;
```

문제 정의

하나의 오류 경보가 거짓으로 판명되었을 때,
자연스럽게 거짓임이 드러나는 또 다른 경보들을
어떻게 찾을 수 있을까?

기대 효과

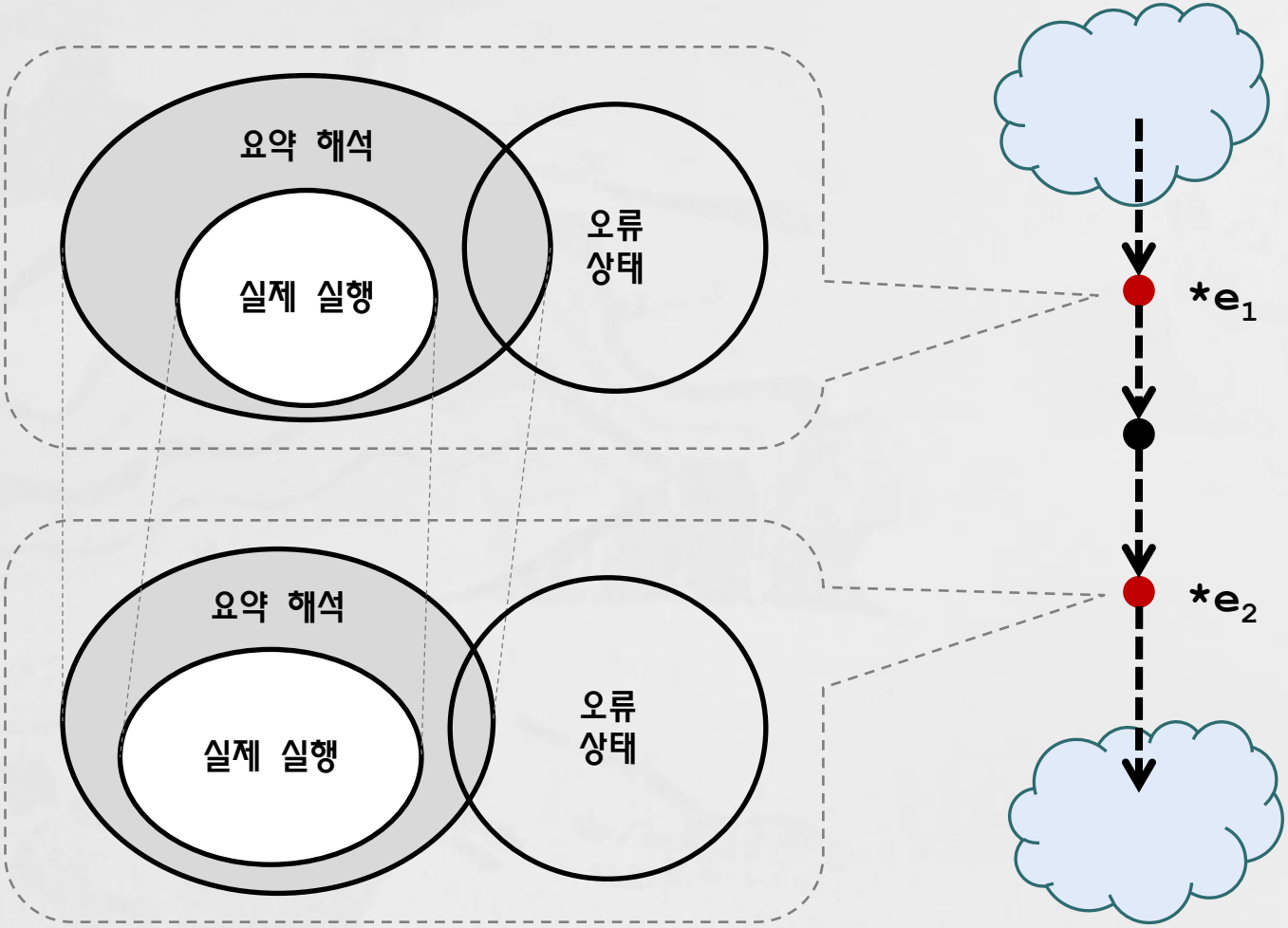
50% 절약

기존 기술들과 잘 어울림

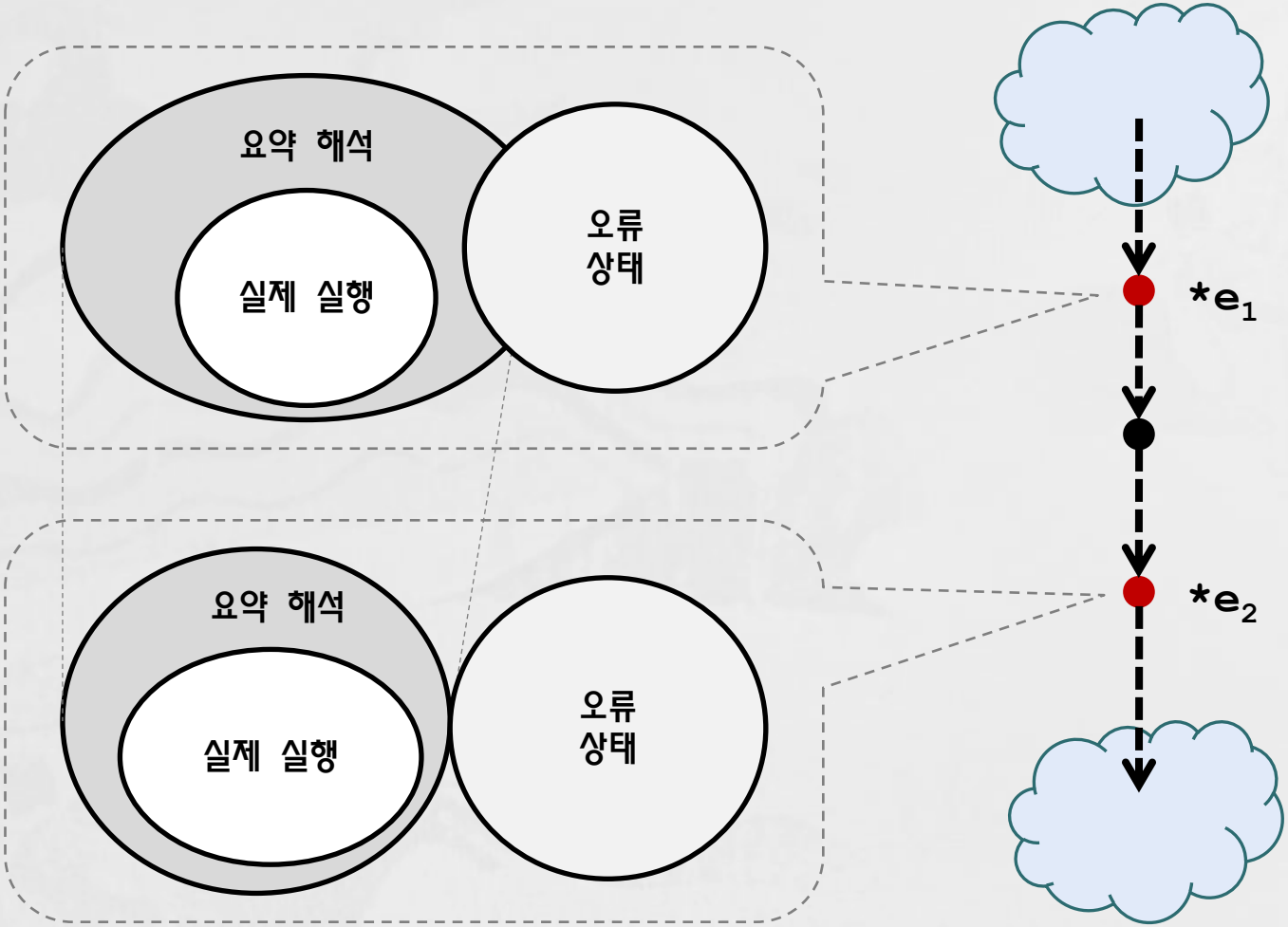
이론

경보 메시지간의 종속 관계를 안전하게 계산하는 틀

거짓 경보란?



거짓 경보간의 종속 관계



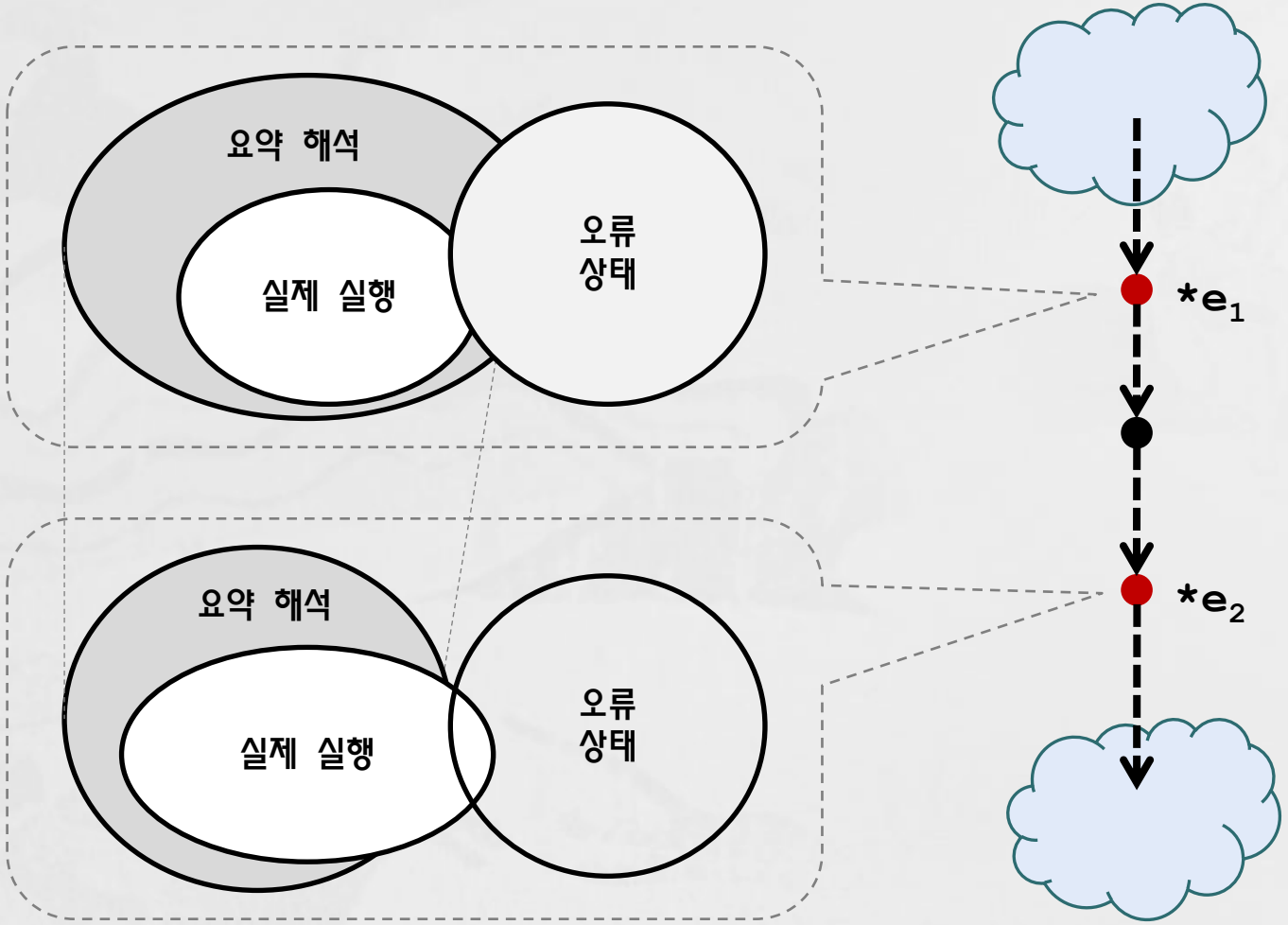
경보간의 종속 관계

가정: 앞의 조건을 만족하는 경우에,

정리 1: 거짓 경보간의 종속 관계
경보 1이 가짜면, 경보 2도 가짜다.

정리 2: 오류간의 종속 관계
경보 2가 진짜면, 경보 1도 진짜다.

오류간의 종속 관계



실제

Airac 경보 메시지 출력을 개선한 구현 사례

구현

- 분석 도메인 확장

$$\hat{Mem} = \hat{Addr} \rightarrow \hat{Val}$$
$$\hat{Addr} = Var + AllocSite$$
$$\hat{Val} = \hat{\mathbb{Z}} \times 2^{\hat{Addr}} \times 2^{AllocSite \times \hat{\mathbb{Z}} \times \hat{\mathbb{Z}}}$$

$$\hat{Mem} = \hat{Addr} \rightarrow \hat{Val} \times AlarmTag$$
$$\hat{Addr} = Var + AllocSite$$
$$\hat{Val} = \hat{\mathbb{Z}} \times 2^{\hat{Addr}} \times 2^{AllocSite \times \hat{\mathbb{Z}} \times \hat{\mathbb{Z}}}$$

구현

- 분석 과정

```
position_set grps[256];

/* ngrps = [0, +oo] */
MALLOC(grps[ngrps]①.elems, position, d->nleaves);


/* ngrps = [0, +oo] */
grps[ngrps]②.nelem = 1;


/* ngrps = [0, +oo] */
grps[ngrps]③.elems[0] = pos;
```

구현

- 분석 과정

```
position_set grps[256];

/* ngrps = [0, +oo] */
MALLOC(grps[ngrps]①.elems, position, d->nleaves);
REFINE(0 <= ngrps < 256, Alarm1);

/* ngrps = [0, +oo] */
grps[ngrps]②.nelem = 1;
REFINE(0 <= ngrps < 256, Alarm2);

/* ngrps = [0, +oo] */
grps[ngrps]③.elems[0] = pos;
REFINE(0 <= ngrps < 256, Alarm3);
```

구현

- 분석 과정

```
position_set grps[256];

/* ngrps = [0, +oo] */
MALLOC(grps[ngrps]①.elems, position, d->nleaves);
REFINE(0 <= ngrps < 256, Alarm1);

/* ngrps = [0, 255], {Alarm1} */
grps[ngrps]②.nelem = 1;
REFINE(0 <= ngrps < 256, Alarm2);

/* ngrps = [0, 255], {Alarm2} */
grps[ngrps]③.elems[0] = pos;
REFINE(0 <= ngrps < 256, Alarm3);
```

실험 결과: Good

분석 대상	SLOC	본래 경보 수	최종 경보 수	감소 비율	그룹 수	평균 그룹 크기	최대 그룹 크기
polymorph-0.4.0	1,357	9	3	67%	1	6.0	6
ncompress-4.2.4	2,195	14	9	36%	3	1.7	2
129.compress	5,585	61	41	33%	5	4.0	15
archimedes-0.7.0	6,959	978	488	50%	163	3.5	12
man-1.5h1	7,232	54	32	41%	10	2.5	7

평균 49% 감소

연결된 경보 메시지들

```
while (1) {
    *(htab_p - 16) = m1;      *(htab_p - 15) = m1;
    *(htab_p - 14) = m1;      *(htab_p - 13) = m1;
    *(htab_p - 12) = m1;      *(htab_p - 11) = m1;
    *(htab_p - 10) = m1;      *(htab_p - 9) = m1;
    *(htab_p - 8) = m1;       *(htab_p - 7) = m1;
    *(htab_p - 6) = m1;       *(htab_p - 5) = m1;
    *(htab_p - 4) = m1;       *(htab_p - 3) = m1;
    *(htab_p - 2) = m1;       *(htab_p - 1) = m1;
    htab_p -= 16;
    i -= 16L;
    if (! (i >= 0L)) {
      break;
    }
  }
  i += 16L;
```

A code snippet from 129.compress (SPEC95)

연결된 경보 메시지들

```
for(m=1;m<=MN3;m++){
  for(j=ND+1-io;j<=NYE-io;j++)
    for(i=ND+1-io;i<=NXE-io;i++)
      bufx2d[i][j]=0.25*(h2d[i][j][m]+h2d[i+1][j][m]
        +h2d[i][j+1][m]+h2d[i+1][j+1][m])
        +0.0625*(ux2d[i][j][m]-ux2d[i+1][j][m]
        +ux2d[i][j+1][m]-ux2d[i+1][j+1][m]
        +uy2d[i][j][m]+uy2d[i+1][j][m]
        -uy2d[i][j+1][m]-uy2d[i+1][j+1][m])
        +dtodx2*(f2d[i][j][m]-f2d[i+1][j][m]
        +f2d[i][j+1][m]-f2d[i+1][j+1][m])
        +dtody2*(g2d[i][j][m]+g2d[i+1][j][m]
        -g2d[i][j+1][m]-g2d[i+1][j+1][m]);
  for(j=ND+1;j<=NYE;j++)
    for(i=ND+1;i<=NXE;i++)
      h2d[i][j][m]=bufx2d[i-io][j-io];
}
```

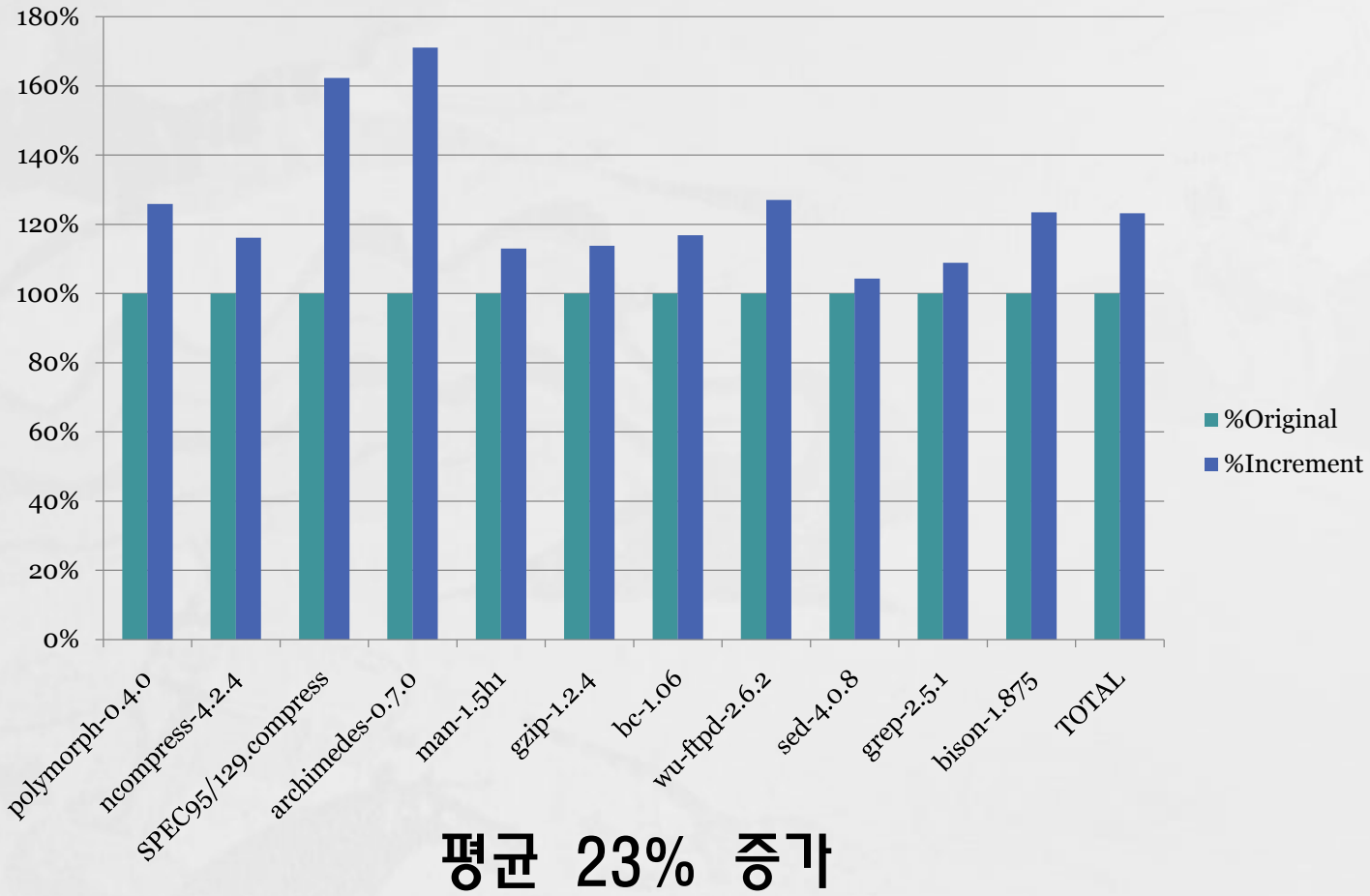
A code snippet from archimedes-0.7.0

실험 결과: Bad

분석 대상	SLOC	본래 경보 수	최종 경보 수	감소 비율	그룹 수	평균 그룹 크기	최대 그룹 크기
gzip-1.2.4	11,213	335	282	16%	40	2.4	5
bc-1.06	12,830	461	443	4%	4	7.0	18
wu-ftpd-2.6.2	18,071	517	464	10%	29	2.6	7
sed-4.0.8	18,687	311	289	7%	6	2.2	6
grep-2.5.1	20,843	30	26	13%	2	3.0	4
bison-1.875	31,203	337	328	3%	6	3.0	4

평균 8% 감소. 안 되는 이유는?

추가적인 분석 시간



평균 23% 증가

연구 계획 & 요약

결과를 개선할 수 있는 두 가지 아이디어

연구 계획: 가벼운 관계 도메인

```
while (*(p + i) && *(p + i) >= '0' && *(p + i) <= '9')
    val = val * 8 + *(p + i)++ - '0';
```

```
i = [0, +oo]
p = {offset: [0, +oo], size: [0, +oo]}
```

⬇

```
REFINE(p.offset + i < p.size);
```

⬇

```
i = [0, +oo]
p = {offset: [0, +oo], size: [0, +oo]}
```

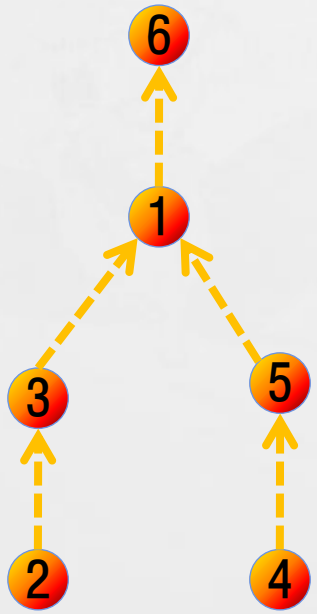
현재의 분석 도메인에서 오류 조건을 잘라낼 수 없다.

연구 계획: 역방향 분석

```
char original[MAX], newname[MAX];

for(i=0;i<strlen(original);i++){
    if( isupper( original[i]① ) ){

        newname[i]② = tolower( original[i]③ );
        continue;
    }
    newname[i]④ = original[i]⑤;
}
newname[i]⑥ = '\0';
```



경보 1과 경보 6의 종속 관계를 알려면 역방향 분석이 필요

요약

- 오류 정적 분석 결과에서 경보간의 종속 관계
  - 이 경보가 가짜면, 저 경보도 가짜다.
  - 이 경보가 진짜면, 저 경보도 진짜다.
- 경보간의 종속 관계를 안전하게 계산하는 틀
  - 완전히 새로운 방법이고,
  - 믿을 수 있고,
  - 기존의 기술과 잘 어울림