Achieving cost-efficiency of fine-tuned path-sensitive static analyzers through data-driven path selection

김진영

SoFA Show & Tell
2019. 4. 19.

요약

- 사람이 정교하게 다듬은 정적분석기의 성능을 데이터에 기반해 개선
- ‘경로 선별 휴리스틱’ 갈아끼우기
  - 데이터 기반 (분석기 + 벤치마크 프로그램)
  - 생성은 자동으로, 준비에 드는 노력은 적게
- 기존 대비 38%의 시간에 91%의 오류를 보고하도록 개선함
- 오늘은: 추가 실험들

선별하는 경로 비교

- 질문: 학습한 경로 선별 휴리스틱과 기존과의 차이는?
- 답: 알람이 날 가능성을 예측하고 이에 기반해 선별

	기존	학습한
보고하는 알람 수	5,563	5,055 (90.8%)
탐색하는 경로 수	7,718,329	3,615,791 (46.8%)
이 중, 계속 탐색하면 알람이 날 수 있는 경로의 비율	3.2%	4.0%
경로/알람	1,387	715 (x0.52)

기존 휴리스틱의 비용-성능 조절과 비교

- 질문: 기존 휴리스틱을 유지하며 비용-성능을 조절한 것보다 좋은가?
- 답: 그렇다. 비용을 조절하려면,
  - 분석 도중 선택하는 최대 경로 개수 제한 (21)
    - 39%의 시간에 81%의 오류를
  - ~~함수별 분석 시간 제한, 전체 분석 시간 제한~~
- 학습한 휴리스틱이 더 적은 시간에 더 많은 오류를 찾음

적은 수의 프로그램으로만 학습

- 질문: 분석기를 정교하게 다듬는 데 쓰인 적은 수의 벤치마크로만 학습해도 일반적으로 잘 동작하는가?
- 답: 벤치마크가 대상 알람을 골고루 포함하고 있다면 그렇다

- 튜닝에 사용된 5개 프로그램으로만 학습 뒤 나머지 12개로 평가
  - 3종의 오류에 대해서는 쓸만한 휴리스틱을 학습해냄
  - 37%의 시간에 87%의 오류를 보고

한 프로그램의 이전 리비전으로 학습

- 질문: 한 프로그램의 이전 리비전들로 학습하면, 이후 리비전들에 대해서 잘 동작하는가?
- 답: 타겟 프로그램을 하나로 한정하여 학습하면,
  - 대체로는 (bash, bison, make, tar)
    - 일반적으로 학습한 것과 비슷하거나 조금 못하는 수준
    - 이질적인 오류가 새로 등장하면 약함
  - 오류의 갯수가 적고, 리비전 전후의 오류가 비슷하면 더 좋기도 함
    - bison: (1.x로 학습, 2.x로 평가) 34%의 시간에 96%의 오류를
    - emacs: (25.x로 학습, 26.x로 평가) 21%의 시간에 97%의 오류를

정리

- 기존 휴리스틱의 비용을 조절한 것보다 더 적은 시간에 더 많은 알람을 찾음
- 기존 휴리스틱 대비
  - 절반 이하의 경로만 탐색
  - 평균적으로 한 알람을 내기 위해 탐색하는 경로의 수도 절반
- 대상 알람들이 고루 포함되어 있다면 학습셋이 작아도 잘 작동
  - 분석기 튜닝에 사용된 적은 수의 벤치마크 프로그램 (5개)
  - 대상 프로그램이 하나라면, 이전 리비전