SNU 4541.664A Program Analysis
Note 19

Prof. Kwangkeun Yi

요약해석 VS 타입 스타일

집합 제약식 분석 Constraint-based Analysis

타입 스타일 VS 요약해석

타입 스타일의 약점

- 요약해석만큼 재사용가능한 정리들이 없음
  - "subject reduction lemma" = 안전성 정리 자체
  - AI: 갈로아 연결, "fixpoint transfer theorem" = 안전성의 조건
- 다양한 정확도의 분석을 디자인하는 공통된 방법이 없음
  - 다형타입(*polymorphism*), 물타기()등 별개의 예
  - AI: 요약의 정도 즉 요약 함수의 순서
- 일반적인 알고리즘이 없음
  - 동일화(*unification*) 알고리즘을 이용, 매번 안전성 증명해야
  - AI: 고정점 반복(*fixpoint iterations*), 명백히 안전함

타입 스타일 VS 요약해석

타입 스타일의 장점

- 분석 디자인 = 논리시스템 디자인
- 형식논리에 익숙하면 유리: 분석의 안전성 증명 = 논리시스템의 안전성
- 분석이 전체 프로그램이 없어도 가능하도록 준비되있다:
  - "$\Gamma \vdash e : \tau$"의 유추, 즉
  - 가정 Γ을 만들고 이용하는 규칙들, 즉
  - 없는 부분에 대한 가정을 다루는 것이 배어있다
- 분석이 빠를 수 있다: 동일화(*unification*) 알고리즘의 과감함

타입 스타일 VS 요약해석

경향/관성

- 타입 시스템을 아는 사람는 타입 시스템으로
- 요약해석을 아는 사람은 요약해석으로

두가지를 모두 알고있는 사람은 억지부리지 않고 균형있게

요약해석 VS 타입 스타일

집합 제약식 분석 Constraint-based Analysis

집합 제약식을 가지고 프로그램 분석

과정

- 프로그램을 훑어서 집합 제약식들을 도출
- 도출된 집합 제약식을 푼다

다른 분석기술도 모두 이런식으로 볼 수 있으나

- 요약해석 = 제약식 도출; 풀기
- 타입시스템 = 제약식 도출; 풀기
- 등등 = 제약식 도출; 풀기

특별한 집합 제약식 분석을 다룬다:

- 집합 제약식을 $O(n^3)$에 풀 수 있는 (일반적으로 NEXPTIME-complete)
- 집합 제약식을 푸는 방식이 또 다르다

집합 제약식

제약식

$$\varphi \supseteq se$$

"φ 집합은 집합식(*set expression*) se 가 의미하는 집합을 포함한다."

$$\begin{array}{rcll} se & \rightarrow & \varphi & \text{집합변수} \\ & | & f(\varphi, \cdots, \varphi) & \text{구성} \\ & | & f_i^{-1}(\varphi) & \text{파괴} \\ & | & se \cap se & \end{array}$$

프로그램으로 부터 제약식들

$$\wedge_i (\varphi_i \supseteq se_i)$$

이 나오고 풀이결과는 제약식들을 모두 만족시키는 φ_i 집합들.

예

각 프로그램식들이 가지는 값들의 집합:

$$(\lambda_0 x.(\lambda_1 y.y\ (x\ 1)))\ (\lambda_2 z.\lambda_3 w.w + z)\ (\lambda_4 a.a\ 2)$$

예

각 프로그램식들이 가지는 값들의 집합, 혹은 각 변수가 가지는 값들의 집합:

```
x := cons(1,cons(2,nil))
y := car(x)
x := cdr(x)
```

예

각 변수가 가지는 값들의 집합:

```
x := nil
while -
  x := cons(1, x)
```