SNU 4541.664A Program Analysis
Note tb-1

Prof. Kwangkeun Yi

타입 시스템 Type-Based Analysis

개괄: 프로그램 분석으로서

타입 시스템 = 형식논리의 증명규칙 + 구현 알고리즘

타입 시스템

프로그램 분석으로서:

- 분석 성질: 프로그램 부품들이 계산해 내는 값들의 타입은?
- 분석의 안전성(*soundness*): 분석한 타입의 값만을 실행중에 계산한다.
- 분석의 알고리즘: 동일화(*unification*) 알고리즘

유의

- 타입 시스템 = 특정 성질(프로그램 타입)에 특화된 분석기
- 안전성 증명방법 = 특화된 증명기술
- 알고리즘 = 그 분석기에 특화된 알고리즘

- 타입 시스템 = 프로그램 부품들의 타입을 결정하는
  - 증명규칙/추론규칙(*inference rules, proof rules*)
- 안전성 증명방법 = 그 증명규칙들로 증명되는
  - 모든 결론은 믿을만함(*soundness*)를 증명, 즉
  - 결론대로 프로그램이 실행됨을 증명
- 알고리즘 = 증명규칙들로 만들어지는 결론과
  - 같은 결론을 계산하는 알고리즘

타입 시스템 = 요약해석의 한 예

모든 타입 시스템들은 요약해석(*abstract interpretation*). 따라서, 프로그램 분석에는 요약해석만 알면 된다?

- CPO, 고정점(*fixpoints*), 갈로아 연결(*Galois connection*), 축지법/좁히기(*widening/narrowing*), 고정점 알고리즘(*fixpoint algorithm*)

그러나,

- 요약해석은 우리의 종교가 아니다
- 논리적으로 문제 없는 방법이라면 뭐든지 상관없다
- 다양한 접근 방법을 익혀야
  - 타입 시스템으로 분석을 디자인 하는 장단점은 실제의 경우를 보고나서 리뷰해보자

타입 시스템 = 증명규칙? 복습

형식논리와 추론

- 논리식 집합의 정의
- 논리식 의미의 정의
- 참논리식 집합의 정의
- 참논리식 추론의 방법
- 추론방법 평가하기

논리식 집합

$$
\begin{array}{rcl}
f & \rightarrow & T \mid F \\
 & \mid & \neg f \\
 & \mid & f \wedge f \\
 & \mid & f \vee f \\
 & \mid & f \Rightarrow f
\end{array}
$$

논리식 의미

조립식 정의 compositional definition

$$
\begin{array}{rcl}
[\![T]\!] & = & \text{true} \\
[\![F]\!] & = & \text{false} \\
[\![\neg f]\!] & = & \text{not}[\![f]\!] \\
[\![f_1 \wedge f_2]\!] & = & [\![f_1]\!] \text{ andalso } [\![f_2]\!] \\
[\![f_1 \vee f_2]\!] & = & [\![f_1]\!] \text{ orelse } [\![f_2]\!] \\
[\![f_1 \Rightarrow f_2]\!] & = & [\![f_1]\!] \text{ implies } [\![f_2]\!]
\end{array}
$$

임의의 논리식 f의 의미가 정의 된 셈.

참 논리식 집합의 정의

쌍 $(\{f_1, \cdots, f_n\}, f)$(표기: "$\Gamma \vdash f$") 들의 집합

$$\frac{}{\Gamma \vdash T} \quad \frac{}{\Gamma \vdash f}\ f \in \Gamma$$

$$\frac{\Gamma \vdash F}{\Gamma \vdash f} \qquad \frac{\Gamma \vdash \neg\neg f}{\Gamma \vdash f}$$

$$\frac{\Gamma \vdash f_1 \quad \Gamma \vdash f_2}{\Gamma \vdash f_1 \wedge f_2}$$

$$\frac{\Gamma \vdash f_1 \wedge f_2}{\Gamma \vdash f_1}$$

$$\frac{\Gamma \vdash f_1}{\Gamma \vdash f_1 \vee f_2}$$

$$\frac{\Gamma \vdash f_1 \vee f_2 \quad \Gamma \cup \{f_1\} \vdash f_3 \quad \Gamma \cup \{f_2\} \vdash f_3}{\Gamma \vdash f_3}$$

$$\frac{\Gamma \cup \{f_1\} \vdash f_2}{\Gamma \vdash f_1 \Rightarrow f_2}$$

$$\frac{\Gamma \vdash f_1 \Rightarrow f_2 \quad \Gamma \vdash f_1}{\Gamma \vdash f_2}$$

$$\frac{\Gamma \cup \{f\} \vdash F}{\Gamma \vdash \neg f}$$

$$\frac{\Gamma \vdash f \quad \Gamma \vdash \neg f}{\Gamma \vdash F}$$

참논리식 집합 정의 = 증명들의 집합을 정의

증명들의 집합을 만드는 귀납규칙 ("증명규칙" inference rules).

- 예를 들어, 증명규칙

$$\frac{\Gamma \vdash f_1 \quad \Gamma \vdash f_2}{\Gamma \vdash f_1 \wedge f_2}$$

  은 증명을 만드는 귀납 규칙
- $\Gamma \vdash f_1$와 $\Gamma \vdash f_2$의 증명들을 가지고 $\Gamma \vdash f_1 \wedge f_2$의 증명을 만든다.

증명 나무

$$\dfrac{\dfrac{}{\{p \to \neg p, p\} \vdash p} \quad \dfrac{\dfrac{\dfrac{}{\{p \to \neg p, p\} \vdash p \to \neg p} \quad \dfrac{}{\{p \to \neg p, p\} \vdash p}}{\{p \to \neg p, p\} \vdash \neg p}}{\{p \to \neg p, p\} \vdash F}}{\{p \to \neg p\} \vdash \neg p}$$

증명 규칙의 평가

기계가 만드는 $\{g_1, \cdots, g_n\} \vdash f$ 는 어떤 것들인가? 예) $[\![g_1 \wedge \cdots \wedge g_n \Rightarrow f]\!] = \text{true}$ 인가?

- 기계의 안전성 soundness:

$$\Gamma \vdash f \text{ 이면 } [\![\Gamma \Rightarrow f]\!] = \text{true}$$

- 기계의 완전성 completeness:

$$\Gamma \vdash f \text{ 면이 } [\![\Gamma \Rightarrow f]\!] = \text{true}$$

타입 시스템에서 논리식

논리식 = 프로그램 식과 그 타입을 표현하는 식

- 타입들은

$$\{int, real, bool, int*, int \to bool, int \times int, \cdots\}$$

즉

$$\begin{array}{rcll} Type & \tau & \to & \iota & \text{primitive type} \\ & & | & \tau \times \tau & \text{product type} \\ & & | & \tau \to \tau & \text{function type} \\ & & | & \tau* & \text{pointer type} \end{array}$$

- 프로그램 식들은 대상 언어의 문법 정의로 부터

$$\begin{array}{rcl} E & \to & n \mid x \mid \&\, x \mid x\, E \\ & | & x := E \mid E\,;\, E \\ & | & \texttt{while}\, E\, E \mid \cdots \end{array}$$

즉, 논리식의 생김새는

$$E : \tau$$

타입 시스템에서 참인 논리식

논리식 의미의 정의

$$[\![E : \tau]\!] = \text{true}$$
$$iff$$
E runs ok and its, if any, value has type τ

참인 타입식 증명 규칙

"Γ안에 있는 논리식들을 가정하면 $E : \tau$ 이다"

$$\Gamma \vdash E : \tau$$

를 증명하는 규칙:

$$\frac{}{\Gamma \vdash n : int} \qquad \frac{}{\Gamma \vdash x : \tau} \; x : \tau \in \Gamma$$

$$\frac{\Gamma \vdash x : \tau}{\Gamma \vdash \& x : \tau *} \qquad \frac{\Gamma \vdash x : \tau_1 \to \tau_2 \quad \Gamma \vdash E : \tau_1}{\Gamma \vdash x\, E : \tau_2}$$

$$\vdots$$

증명규칙의 안전성

증명 규칙이 결론내리는(만드는) $\Gamma \vdash E : \tau$ 는 안전한가?

- 임의의 프로그램 식 E에 대해서

$$\Gamma \vdash E : \tau \text{ 이면 } [\![\Gamma]\!] \Rightarrow [\![E : \tau]\!] = \mathsf{true}$$

  인가?
- 증명: 모든 $\Gamma \vdash E : \tau$에 대해서 증명 (귀납법, why?)

구현 알고리즘

알고리즘 $A(E, \Gamma)$

$$A(E, \Gamma) = \tau \quad \Longleftrightarrow \quad \Gamma \vdash E : \tau$$

아니면 적어도

$$A(E, \Gamma) = \tau \quad \Longrightarrow \quad \Gamma \vdash E : \tau$$

- $A(E, \Gamma)$는 E로 부터 타입식에 대한 연립 방정식을 도출하고 그 방정식을 푼다
- 방정식 푸는 방법이 고정점 알고리즘보다는 타입식에 특화
  - 동일화(*unification*) 알고리즘