SNU 4541.664A Program Analysis Note 20

Prof. Kwangkeun Yi

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 분석만을 다룬다:

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

일반 집합 제약식

집합 제약식

$$se \supseteq se'$$

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

$$\begin{array}{rcll}
\varphi & \in & V & \text{집합 변수 집합} \\
f & \in & C & \text{구성자}(\textit{constructor})\text{ 집합} \\
se & \to & \varphi & \text{집합변수} \\
 & \mid & f(se, \cdots, se) & \text{구성} \\
 & \mid & f^{-i}(se) & \text{파괴} \\
 & \mid & se \cap se & \text{교집합} \\
 & \mid & se \cup se & \text{합집합} \\
 & \mid & \neg se & \text{여집합}
\end{array}$$

프로그램 분석에 사용할 집합 제약식

집합 제약식

$$\varphi \supseteq se$$

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

$$\begin{array}{rcll}
\varphi & \in & V & \text{집합 변수 집합} \\
f & \in & C & \text{구성자(\textit{constructor}) 집합} \\
se & \to & \varphi & \text{집합변수} \\
 & \mid & f(\varphi, \cdots, \varphi) & \text{구성(\textit{construction})} \\
 & \mid & f^{-i}(\varphi) & \text{파괴(\textit{deconstruction})} \\
 & \mid & se \cap se & \text{교집합} \\
 & \mid & \top \mid \bot &
\end{array}$$

프로그램으로 부터 제약식들

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

이 나오고 풀이결과는 제약식들을 모두 만족시키는 φ_i 집합들.

집합식의 의미 (1/2)

$$
\begin{array}{rcll}
\varphi & \in & V & \text{집합 변수 집합} \\
f & \in & C & \text{구성자}(\textit{constructor})\text{ 집합} \\
se & \rightarrow & \varphi & \text{집합변수} \\
 & | & f(\varphi, \cdots, \varphi) & \text{구성}(\textit{construction}) \\
 & | & f^{-i}(\varphi) & \text{파괴}(\textit{deconstruction}) \\
 & | & se \cap se & \text{교집합} \\
 & | & \top \mid \bot &
\end{array}
$$

- 구성자가 필요로하는 인자수(*arity*)는 정해져 있고, 인자가 필요없는 구성자는 상수
- 집합식의 의미는 낱말(*term*)들의 집합 $\in 2^H$
- 모든 낱말(*term*) t 들의 집합 H(Herbrand universe)는

$$
\begin{array}{rcrcll}
H & \ni & t & \rightarrow & f & \text{상수 낱말}(\textit{term}) \\
 & & & | & f(t, \cdots, t) & \text{인자로 구성한 낱말}(\textit{term})
\end{array}
$$

$$
\begin{array}{lcll}
se & \to & \varphi & \text{집합변수} \\
 & \mid & f(\varphi, \cdots, \varphi) & \text{구성}(\textit{construction}) \\
 & \mid & f^{-i}(\varphi) & \text{파괴}(\textit{deconstruction}) \\
 & \mid & se \cap se & \\
 & \mid & \top \mid \bot &
\end{array}
$$

se의 의미 $[\![se]\!]_\sigma$는

$$\sigma \in V \to 2^H$$

에 기대어

$$
\begin{array}{rcl}
[\![\varphi]\!]_\sigma & = & \sigma(\varphi) \\
[\![f(\varphi_1, \cdots, \varphi_n)]\!]_\sigma & = & \{f(t_1, \cdots, t_n) \mid t_i \in [\![\varphi_i]\!]_\sigma\} \\
[\![f^{-i}(\varphi)]\!]_\sigma & = & \{t_i \mid f(t_1, \cdots, t_n) \in [\![\varphi]\!]_\sigma\} \\
[\![se \cap se']\!]_\sigma & = & [\![se]\!]_\sigma \cap [\![se']\!]_\sigma \\
[\![\top]\!]_\sigma & = & H \\
[\![\bot]\!]_\sigma & = & \emptyset
\end{array}
$$

집합 제약식의 해

유한개의 연립 제약식들

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

의 해는

$$\sigma \in V \to 2^H,$$

각 제약식들을 모두 만족시키는:

$$\wedge_i([\![\varphi_i]\!]_\sigma \supseteq [\![se_i]\!]_\sigma).$$

- 연립 집합 제약식 $\mathcal{C}$의 해 σ를 "모델"(*model*)이라고 부르고
- 이렇게 $\sigma \models \mathcal{C}$ 쓴다.

집합 제약식 해/모델

연립 집합 제약식

$$C \stackrel{\text{let}}{=} \wedge_i(\varphi_i \supseteq se_i)$$

의 해 $\sigma \in V \to 2^H$는 항상 존재:

- 쓸모없는 $\{\varphi \mapsto H \mid \varphi \in V\}$

해 중에서 가장 작은 해가 항상 존재:

- $\mathcal{M}$가 모든 해들의 집합 $\mathcal{M} = \{\sigma \mid \sigma \models C\}$ 일 때, 최소의 해는

$$\bigcap \mathcal{M} \stackrel{\text{def}}{=} \{\varphi \mapsto \bigcap_{\sigma \in \mathcal{M}} \sigma(\varphi)\}$$

집합식 설탕: $se \Rightarrow se'$

$$[\![\varphi_1 \Rightarrow \varphi_2]\!]_\sigma = \begin{cases} \emptyset & \text{if } [\![\varphi_1]\!]_\sigma = \emptyset \\ [\![\varphi_2]\!]_\sigma & \text{otherwise} \end{cases}$$

따라서,

$$\begin{array}{rcl} \varphi_1 \Rightarrow \varphi_2 & \equiv & f^{-1}(f(\varphi_2, \varphi_1)) \\ se_1 \Rightarrow se_2 & \equiv & ? \end{array}$$

집합 제약식 분석

- 대상 프로그래밍 언어에 적당한 집합 제약식을 정의
  - 집합변수 $\varphi \in V$ 들이 프로그램의 무엇과 연계되는지
  - 구성자(*constructor*) 집합 C 즉, 낱말(*term*) 집합 H은 무엇이 되는지
- 프로그램으로 부터 집합 제약식 도출 방법 정의
- 도출된 집합 제약식의 해를 계산
  - 도출된 집합 제약식을 모두 참으로 만드는 $\sigma \in V \rightarrow 2^H$
  - $\boxed{\sigma(\varphi)\text{를 집합 제약식들로 표현}}$

집합 제약식 분석 예

리스트 처리 명령형 언어

$$
\begin{array}{rcl}
c & \rightarrow & x \mathrel{\texttt{:=}} e \\
  & | & c \,\texttt{;}\, c \\
  & | & \texttt{while}\ e\ c \\
  & | & \texttt{case}\ e\ \texttt{nil:}\ \ c\ \texttt{cons:}\ \ c \\
e & \rightarrow & n \mid \texttt{nil} \mid x \\
  & | & \texttt{cons(}e\texttt{,}e\texttt{)} \\
  & | & \texttt{car(}e\texttt{)} \mid \texttt{cdr(}e\texttt{)}
\end{array}
$$

분석 목표: 각 변수가 가지는 값들의 집합; 각 명령문 실행후 변수가 가지는 값들의 집합

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

집합 제약식 분석 예

분석 목표: 각 변수가 가지는 값들의 집합; 각 명령문 실행후 변수가 가지는 값들의 집합

```
x := nil;
while y
  x := cons(1, x);
  y := cdr(x)
```

집합 제약식 분석 예

분석 목표: 각 변수가 가지는 값들의 집합; 각 명령문 실행후 변수가 가지는 값들의 집합

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

집합 제약식 분석 예

분석 목표: 각 변수가 가지는 값들의 집합; 각 명령문 실행후 변수가 가지는 값들의 집합

```
case x of
  nil:   y := cons(1, x)
  cons:   y := cdr(x)
```

집합 제약식 분석 예

람다 함수형 언어

$$\begin{array}{rcl} e & \rightarrow & n \mid x \\ & \mid & \lambda x.e \mid e\, e \end{array}$$

각 식들이 가지는 값들의 집합:

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

집합 제약식의 해를 구하는 알고리즘 얼개

- 프로그램 pgm을 훑어서 집합 제약식 집합 C를 도출하는 방법

$$pgm \triangleright C$$

결정

- 집합 제약식을 풀어가는 규칙(*constraint resolving rules*) R을 정의: R은 새로운 제약식을 더해가는 규칙
- 주어진 C에서 R로 풀 수 있는 데 까지 푼다. 즉,

$$lfp \lambda X.C \cup \{a \mid X \vdash_R a\} \stackrel{\text{let}}{=} R^*(C)$$

를 계산.

- 해는 $R^*(C)$에서 각 변수 φ의 제약식들 중에서

해는 $R^*(C)$에서 각 변수 φ의 제약식들 중에서 최소 알갱이로 풀려진 제약식(*atomic constraints*)

$$
\begin{array}{rcl}
\varphi & \supseteq & ae \\
ae & \rightarrow & \top \mid \bot \\
 & \mid & f(\varphi, \cdots, \varphi)
\end{array}
$$

들로 구성된다.

해 = 문법

$R^*(C)$에서 φ의 풀려진 제약식들

$$\begin{array}{rcl} \varphi & \supseteq & ae_1 \\ & \vdots & \\ \varphi & \supseteq & ae_n \end{array}$$

를 만족하는 집합 φ는 다음의 문법이 만드는 집합이다:

$$\begin{array}{rcl} \varphi & \rightarrow & ae_1 \\ & \vdots & \\ & \mid & ae_n \end{array}$$

읽기

[읽을 논문] Constraint-based Analysis의 논문 4편.