Análisis de Lenguajes de Programación

Semántica

8 de Septiembre de 2025

Definición formal de un lenguaje

- La sintaxis especifica cómo se construye un lenguaje. Vimos
  - cómo generar un lenguaje CF a partir de una CFG,
  - y cómo definir un analizador sintáctico (parser) que reconozca un lenguaje generado por una CFG.
- La semántica da significado a cada programa gramaticalmente correcto del lenguaje.

Beneficios de la semántica formal

- **Implementación**
  - Ayudan a escribir compiladores
  - Permiten garantizar que las optimizaciones son correctas.
- **Análisis estático** Dado que la semántica provee la base para razonar sobre programas, es necesaria para probar propiedades
  - Propiedades de correctitud
  - Propiedades de seguridad
- **Diseño de lenguaje**
  - Resolver ambiguedades de construcciones del lenguaje.
  - El uso de las matemáticas puede sugerir estilos de programación.

Enfoques clásicos de semánticas

Operacional: Captura el significado de un programa como una relación que describe cómo se ejecuta.

Denotacional: El significado de un programa es modelado por objectos matemáticos que representan el efecto de ejecutar las construcciones y pertenecen a un dominio $\mathcal{D}$.
Función de interpretación: $[\![\_]\!] : \mathcal{T} \to \mathcal{D}$

Axiomática: Especifica propiedades sobre el efecto de ejecutar programas. Enfoque basado en la lógica matemática, la más conocida es la lógica de Hoare usada para probar la correcctitud de programas imperativos.

Tipos de semántica operacional

La **semántica operacional** nos proporciona un modelo para la implementación de un intérprete o compilador del lenguaje. Se distinguen dos tipos de acuerdo a los detalles de ejecución que brindan:

paso chico (small-step): la evaluación se hace paso a paso.

paso grande (big-step): oculta cómo se llega al resultado.

Ejemplo de semántica operacional

- Ambos tipos de semántica se definen mediante una **relación de transición**.
- Definimos el conjunto de términos $\mathcal{T}$ como:

  $t \ ::= \ \mathbf{T} \mid \mathbf{F} \mid \mathbf{if}\ t\ \mathbf{then}\ t\ \ \mathbf{else}\ t$

  y el conjunto de valores $\mathcal{V}$ ($\mathcal{V} \subseteq \mathcal{T}$):

  $v \ ::= \ \mathbf{T} \mid \mathbf{F}$

- t y v son metavariables que denotan términos y valores respectivamente.
- La semántica *big-step* va a relacionar un término con un valor.

Ejemplo de semántica *big-step*

Definimos $\Downarrow \subseteq \mathcal{T} \times \mathcal{V}$ como la menor relación que satisface estas reglas:

$$\frac{}{v \Downarrow v} \qquad \text{(B-Val)}$$

$$\frac{t_1 \Downarrow \mathtt{T} \qquad t_2 \Downarrow v}{\mathtt{if}\ t_1\ \mathtt{then}\ t_2\ \mathtt{else}\ t_3 \Downarrow v} \qquad \text{(B-IfTrue)}$$

$$\frac{t_1 \Downarrow \mathtt{F} \qquad t_3 \Downarrow v}{\mathtt{if}\ t_1\ \mathtt{then}\ t_2\ \mathtt{else}\ t_3 \Downarrow v} \qquad \text{(B-IfFalse)}$$

Observaciones

- La primer regla es un axioma.
- Cada regla define un **esquema** de reglas, dado que t_1, t_2, t_3 y v son metavariables. A partir de ellas se deducen infinitas reglas, por ejemplo:

$$\frac{}{\mathtt{T} \Downarrow \mathtt{T}}$$

$$\frac{\mathtt{T} \Downarrow \mathtt{F} \qquad \mathtt{F} \Downarrow \mathtt{F}}{\mathtt{if\ T\ then\ T\ else\ F} \Downarrow \mathtt{F}}$$

Árbol de derivación

- Cuando $(t, v) \in \Downarrow$, es decir (t, v) pertenece a la relación de evaluación decimos t **deriva a** v y escribimos $t \Downarrow v$.

- Mostraremos que $t \Downarrow v$ mediante un **árbol de derivación**, cuyas hojas van a ser instancias de los axiomas y los nodos internos instancias de las reglas.

Ejemplo de Árbol de Derivación

Probamos que

$$\texttt{if}\ \left(\texttt{if F then T else T}\right)\ \texttt{then F else T} \Downarrow \texttt{F}$$

$$\dfrac{\dfrac{\dfrac{}{\texttt{F} \Downarrow \texttt{F}}\ (\textsc{B-Val}) \qquad \dfrac{}{\texttt{T} \Downarrow \texttt{T}}\ (\textsc{B-Val})}{\texttt{if F then T else T} \Downarrow \texttt{T}}\ (\textsc{B-IfFalse}) \qquad \dfrac{}{\texttt{F} \Downarrow \texttt{F}}\ (\textsc{B-Val})}{\texttt{if}\ \left(\texttt{if F then T else T}\right)\ \texttt{then F else T} \Downarrow \texttt{F}}\ (\textsc{B-IfTrue})$$

Ejercicio: Probar que

$$\texttt{if F then T else}\ \left(\texttt{if T then F else T}\right) \Downarrow \texttt{F}$$

Relación de Evaluación de Paso Chico

- La semántica se da por una relación entre "estados" de una máquina abstracta.
- Definimos la relación de evaluación $\to \subseteq \mathcal{T} \times \mathcal{T}$

$$\texttt{if T then } t_2 \texttt{ else } t_3 \to t_2 \quad (\text{E-IfTrue})$$

$$\texttt{if F then } t_2 \texttt{ else } t_3 \to t_3 \quad (\text{E-IfFalse})$$

$$\frac{t_1 \to {t_1}'}{\texttt{if } t_1 \texttt{ then } t_2 \texttt{ else } t_3 \to \texttt{if } {t_1}' \texttt{ then } t_2 \texttt{ else } t_3} \; (\text{E-If})$$

- $t \to t'$ se lee "t evalúa a t' en un solo paso".

Acerca de la relación de Evaluación de Paso Chico

- Notar que `T` y `F` no evalúan a nada.
- Las reglas a veces se dividen en reglas de computación (E-IfTrue y E-IfFalse) y reglas de congruencia (E-If.)
- La relación de evaluación fija una **estrategia de evaluación**.
  - En `if` t_1 `then` t_2 `else` t_3, se debe evaluar t_1 antes de evaluar t_2 o t_3.

Ejercicio

Modificar la relación de evaluación

$$\texttt{if T then } t_2 \texttt{ else } t_3 \rightarrow t_2 \qquad (\text{E-IfTrue})$$

$$\texttt{if F then } t_2 \texttt{ else } t_3 \rightarrow t_3 \qquad (\text{E-IfFalse})$$

$$\frac{t_1 \rightarrow t_1'}{\texttt{if } t_1 \texttt{ then } t_2 \texttt{ else } t_3 \rightarrow \texttt{if } t_1' \texttt{ then } t_2 \texttt{ else } t_3} \ (\text{E-If})$$

para que en $\texttt{if } t_1 \texttt{ then } t_2 \texttt{ else } t_3$ se evalúe primero t_2, luego t_3 y finalmente t_1.

Recordar que usamos la metavariable v para representar valores (en este caso `T` o `F`.)

Ejemplo: Árbol de Derivación

Sea

$$\begin{aligned}
s &= \texttt{if T then F else T}\\
t &= \texttt{if } s \texttt{ then T else T}\\
u &= \texttt{if F then T else T}
\end{aligned}$$

entonces podemos justificar que

$$\texttt{if } t \texttt{ then F else F} \to \texttt{if } u \texttt{ then F else F}$$

con el árbol

$$\dfrac{\dfrac{\dfrac{}{s \to \texttt{F}}\ (\textsc{E-IfTrue})}{t \to u}\ (\textsc{E-If})}{\texttt{if } t \texttt{ then F else F} \to \texttt{if } u \texttt{ then F else F}}\quad (\textsc{E-If})$$

Inducción sobre una derivación

- Las propiedades sobre la relación de evaluación son probadas generalmente por inducción sobre la derivación de evaluación.

- Probaremos que un predicado $\mathcal{P}$ vale para cualquier dericación $\mathcal{D}$, probando que:

  "Para cada derivavión D, suponiendo que vale $\mathcal{P}(\mathcal{C})$, donde $\mathcal{C}$ es una derivación inmediata de $\mathcal{D}$, vale $\mathcal{P}(\mathcal{D})$."

Ejemplo: Determinismo de evaluación de paso chico

Teorema

Si $t \to t'$ y $t \to t''$, entonces $t' = t''$.

Pasos a seguir

1. Elegimos sobre qué derivación vamos a hacer inducción (por ej, $t \to t'$).
2. Hacemos análisis por casos sobre la última regla aplicada en la derivación. Pudiendo usar la HI en los casos que contienen subderivaciones.

Ejemplo

Demostración

- Haremos inducción sobre la derivación $t \to t'$.

- Si la última regla aplicada es:
  - $E - IFTRUE$, entonces la forma de t es **if** T **then** t_2 **else** t_3 y $t' = t_2$.

    En la derivación $t \to t''$ la última regla aplicada no puede ser $E - IFFALSE$, por la forma de t.

    Tampoco puede ser $E - IF$, ya que esa regla requiriría que $T \to t_1'$, y T es un valor.

    Por lo tanto la única regla aplicada es $E - IFTRUE$.
    Concluímos entonces que $t'' = t_2$, con lo cual $t' = t''$.
  - $\mathtt{E} - \mathtt{IFFALSE}$, el razonamiento es similar al caso anterior

continuación ejemplo

- $\mathtt{E}-\mathtt{IF}$, t tiene la forma **if** t_1 **then** t_2 **else** t_3, donde $t_1 \to t_1'$ y $t' =$ **if** t_1' **then** t_2 **else** t_3.

  En la derivación $t \to t''$ la última regla no pudo ser $E - IFTRUE$, dado que $t_1 \to t_1'$, es decir, no puede ser T.

  Tampoco pudo aplicarse $E - IFFALSE$.

  La única regla aplicada es $E - IF$, es decir que $t_1 \to t_1''$ y $t'' =$ **if** t_1'' **then** t_2 **else** t_3.

  Por HI, dado que $t_1 \to t_1'$ es una subderivación de $t \to t'$ y $t_1 \to t_1''$, $t_1' = t_1''$, con lo cual $t' = t''$

Forma Normal

- Un término t está en **forma normal** si no se le puede aplicar ninguna regla de evaluación.
- O sea, t está en forma normal si no existe t' tal que $t \rightarrow t'$.
- Para nuestro lenguaje simple las formas normales son `T` y `F` (los valores).
- **Teorema**: Todo valor está en forma normal.
- En general el converso no vale (por ej. errores de ejecución), pero para nuestro lenguaje tenemos el siguiente teorema:
- Si t está en forma normal, entonces t es un valor.
  - Prueba: por inducción estructural sobre t en el contrarecíproco.

Evaluación de pasos múltiples

- La relación de **pasos múltiples** $\rightarrow^*$ es la clausura reflexivo-transitiva de $\rightarrow$.
- Es decir es la menor relación tal que

$$\frac{t \rightarrow t'}{t \rightarrow^* t'} \qquad \frac{}{t \rightarrow^* t} \qquad \frac{t \rightarrow^* t' \qquad t' \rightarrow^* t''}{t \rightarrow^* t''}$$

Teorema (Unicidad de Formas Normales)

Si $t \rightarrow^* u$ *y* $t \rightarrow^* u'$, *donde* u *y* u' *son formas normales, entonces* $u = u'$.

Teorema (La evaluación termina)

Para todo término t *hay una forma normal* t' *tal que* $t \rightarrow^* t'$.

Más resultados

- La evaluación de paso grande tiene propiedades similares

Teorema (Determinismo)

Si $t \Downarrow v$ *y* $t \Downarrow v'$ *entonces* $v = v'$.

Teorema (Terminación)

Para todo término t*, existe* v *tale que* $t \Downarrow v$.

- Relación entre las dos semánticas

Teorema (Equivalencia de paso grande y chico)

Para todo término t *y valor* v*,* $t \Downarrow v$ *sii* $t \rightarrow^* v$.

Semántica del Lenguaje de Expresiones Aritméticas

- Trabajamos ahora con el lenguaje de expresiones aritméticas completo.

$$
\begin{aligned}
t ::= &\ \mathtt{T} \mid \mathtt{F} \mid \mathtt{if}\ t\ \mathtt{then}\ t\ \mathtt{else}\ t \\
\mid &\ 0 \mid \mathtt{succ}\ t \mid \mathtt{pred}\ t \mid \mathtt{iszero}\ t
\end{aligned}
$$

- Para definir los valores agregamos una nueva categoría sintáctica de valores numéricos:

$$
\begin{aligned}
v &::= \mathtt{T} \mid \mathtt{F} \mid nv \\
nv &::= 0 \mid \mathtt{succ}\ nv
\end{aligned}
$$

- Vamos a definir la relación de evaluación para el lenguaje completo, agregando reglas a las existentes.

Nuevas reglas de evaluación de paso chico

$$\frac{t_1 \to t_1'}{\texttt{succ}\ t_1 \to \texttt{succ}\ t_1'} \qquad (\textsc{E-Succ})$$

$$\texttt{pred}\ 0 \to 0 \qquad (\textsc{E-PredZero})$$

$$\texttt{pred}\ (\texttt{succ}\ nv_1) \to nv_1 \qquad (\textsc{E-PredSucc})$$

$$\frac{t_1 \to t_1'}{\texttt{pred}\ t_1 \to \texttt{pred}\ t_1'} \qquad (\textsc{E-Pred})$$

$$\texttt{iszero}\ 0 \to \texttt{T} \qquad (\textsc{E-isZeroZero})$$

$$\texttt{iszero}\ (\texttt{succ}\ nv_1) \to \texttt{F} \qquad (\textsc{E-IsZeroSucc})$$

$$\frac{t_1 \to t_1'}{\texttt{iszero}\ t_1 \to \texttt{iszero}\ t_1'} \qquad (\textsc{E-IsZero})$$

Acerca de las nuevas reglas

- Notar el rol que juega la categoría sintáctica nv en la estrategia de evaluación.
- Por ejemplo, no se puede usar $\text{E-P}\textsc{red}\text{S}\textsc{ucc}$ para concluir que $\texttt{pred}\ (\texttt{succ}\ (\texttt{pred}\ 0)) \to \texttt{pred}\ 0$.
- Notar que términos como $\texttt{succ}\ \texttt{F}$ son formas normales, pero **no son valores**.
- Si t es una forma normal pero no es un valor, decimos que t está **atascado** (stuck).
- Un término atascado se puede pensar como error de run-time. No se puede seguir la ejecución porque se llegó a un estado sin sentido.

Ejercicios

- Probar que la relación de evaluación es determínistica. O sea que si $t \to t'$, y $t \to t''$, entonces $t' = t''$.
- Probar que todo valor es una forma normal.

Resumen

- Diferentes formas de especificar la semántica de lenguajes.
- Semántica operacional de paso grande y de paso chico
  - Valores,relación de evaluación, árbol de derivación, forma normal, términos atascados.
  - Propiedades: determinismo, valores como forma normal, unicidad de formas normales, terminación.

- **Referencias:** Types and Programming Languages. Benjamin Pierce. Capítulo 3.