NOMBRE: Matías Duhalde

Nº LISTA: -

PUNTAJE:

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2223 — Teoría de Autómatas y Lenguajes Formales — 2' 2021

Tarea 4 – Respuesta Pregunta 2

1.

Se define el siguiente algoritmo

Algorithm 1 Determinar si X es alcanzable desde S en el grafo (V, E)

```
function IsReachable(grafo = (V, E), S, X)
    Q ← Queue()
    Q.enqueue(S)
    visited ← {S}
    while Q is not empty do
        Y ← Q.dequeue()
        if X = Y then
            return TRUE
        end if
        adjacent ← {A ∈ V | (Y, A) ∈ E}
        for A ∈ adjacent do
            if A ∉ visited then
                visited ← visited ∪ {A}
                Q.enqueue(A)
            end if
        end for
    end while
    return FALSE
end function
```

PD: IsReachable es correcto

Se puede notar que el algoritmo descrito corresponde a recorrer con BFS el grafo, comenzando desde el nodo S, que representa también la variable inicial de la CFG $G = (V, \Sigma, P, S)$, hasta llegar a un nodo $X \in V$.

Por definición del algoritmo, tenemos que si IsReachable($grafo$, S, X) = `TRUE`, entonces, existe un camino en el grafo que va desde S hasta X. Por construcción del grafo (V, E), si $(A, B) \in E$, entonces existen $\alpha, \beta \in (V \cup \Sigma)^*$ tal que $A \to \alpha B \beta \in P$.

Sea el camino encontrado en BFS para retornar `TRUE` el siguiente:

$$S \to A_1 \to A_2 \to ... \to X$$

Como caso base podemos ver que si S es adyacente con A_1 en el grafo, entonces existe existe $\alpha, \beta \in (V \cup \Sigma)^*$ tal que $S \to \alpha A_1 \beta \in P$, y por lo tanto A_1 es alcanzable.

Usando inducción, si asumimos que A_i es alcanzable, y existe un nodo A_{i+1} adyacente a A_i, entonces existen $\alpha, \beta \in (V \cup \Sigma)^*$ tal que $A_i \to \alpha A_{i+1} \beta \in P$, y A_{i+1} también es alcanzable. Con esta tesis inductiva, podemos concluir que X es alcanzable.

Por lo tanto, si IsReachable retorna `TRUE` con X, entonces X es alcanzable.

De manera análoga, se demuestra que si X es alcanzable, entonces por definición existe una derivación $S \to \alpha_1 A_1 \beta_1 \to ... \to \alpha_n X \beta_n$, con $A_i \in V$ y $\alpha_i \in (V \cup \Sigma)^*$.

Como caso base, tenemos que como $S \to \alpha_1 A_1 \beta_1$ es parte de la derivación, la regla pertenece P. Por definición del grafo, la arista $(S, A_1) \in E$. Usando inducción, asumiendo que la arista $(A_{i-1}, A_i) \in E$, si la $\alpha_i A_i \beta_i \to \alpha_{i+1} A_{i+1} \beta_{i+1}$ se encuentra en la derivación, por construcción del grafo nuevamente también $(A_i, A_{i+1}) \in E$. De esta manera, se puede concluir que si existe $\alpha_{n-1} A_{n-1} \beta_{n-1} \to \alpha_n X \beta_n$ en la derivación, entonces existe una arista (A_n, X), y al mismo tiempo, existe un camino que une S con X, por lo que el algoritmo retornaría `TRUE`.

Por lo tanto, si X es alcanzable, entonces IsReachable retorna `TRUE` con X. $\square$

2.

Para un hipotético algoritmo que cumpliera lo descrito en el enunciado, se pueden dar los siguientes casos:

1. $X \in T$
2. $X \notin T$

Para el primer caso, sería trivial para un algoritmo comprobar que S es generadora, debido a que tendría una regla directa a w, por lo que retornaría `TRUE` en ese caso.

Sin embargo para el segundo caso, no sería posible asumir directamente que X sea una variable generadora o no. Para esto, se pueden dar dos casos más:

1. No existe un camino en el grafo que lleve de X a algún $A \in T$
2. Existe un camino en el grafo que lleve de X a algún $A \in T$

Revisar estos casos sería simple de implementar usando un algoritmo que recorra el grafo desde X (como BFS, usado en la parte 1). Para el primero, se puede concluir que X no es generadora, por lo que el algoritmo retornaría `FALSE`.

Para el segundo caso, el hecho de existencia de un camino no implica necesariamente que X sea generadora. Los siguientes ejemplos pueden presentarse en este caso, para algún $D \notin T$.

$$X \to A \tag{9}$$

$$X \to AD \mid D \qquad D \to D \tag{10}$$

Para el ejemplo (1), es fácil darse cuenta que X sería generadora, debido a que $A \in T$ y existe una regla $A \to w$, con $w \in \Sigma^*$. Para (2), se puede comprobar que X no es generador, debido a que todas sus reglas llevan a la variable D, y D tampoco es generadora, dado que su única regla lleva a D, y $D \notin T$. Para poder distinguir entre estos dos ejemplos, no es posible hacerlo solamente con el grafo y con T, se requeriría más información sobre P (que no es proporcionado al algoritmo) para poder decidir. De esta manera, no se puede implementar un algoritmo que logre responder `TRUE` con para X generadora, con sólo el grafo y T. $\square$