ISSN 2007-9737

Problema de membresía con matriz de adyacencia

Yolanda Moyao Martinez, Jose de Jesus Lavalle Martínez, Carlos Guillen Galban, Darnes Vilariño Ayala

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación,
México

{ymoyao,dvilarino}@cs.buap.mx

Resumen. En este articulo se propone un algoritmo, para resolver el problema de membresía en Gramáticas de Reemplazo de Hiperaristas (HRG). Given a hypergraph H with labeled nodes and hyperedges, dirigidas y enraizadas, el problema consiste en determinar si $H \in L(G)$, donde $G \in HRG$, es decir si H está en el lenguaje generado por G, para esto el análisis se lleva a cabo directamente en la Matriz de Adyacencias del hipergrafo H. Además, para el algoritmo propuesto se presenta la demostración de que es *correcto*.

Palabras clave. Problema de membresía, hipergrafo, matriz de adyacencias.

Membership Problem with Adjacency Matrix

Resumen. In this article, proposed a algorithm to solve the membership problem in Hyperedge Replacement Grammars (HRG). Given a hypergraph H with labeled nodes rooted and directed hyperedges, the problem consists in determining if $H \in L(G)$, where G is in HRG, this is to say, if H is in the language generated by G, for this the analysis is done directly in the Adjacency Matrix of the hypergraph H. For the algorithm proposed, also presents the proof of its correctness.

Palabras clave. Membership problem, hypergraph, adjacency matrix.

1. Introducción

En este artículo se presenta un algoritmo *Analizador* que soluciona el problema de membresía en Gramáticas de Reemplazo de Hiperaristas (HRG), por sus siglas en inglés, es decir, a través de las reglas de producción reconoce el hipergrafo de entrada H.

Las gramáticas de hipergrafos se han desarrollado como una extensión del concepto formal de Gramáticas Libres de Contexto, donde el concepto de reescritura de símbolos por cadenas se generaliza a reemplazo de hiperaristas por hipergrafos, a pesar de que en el caso de cadenas el reemplazo de una cadena por otra suele ser un proceso sencillo, no es así para el caso de hipergrafos, pues tratándose de este esquema se requiere encontrar una ocurrencia de un hipergrafo para ser reemplazado por otro hipergrafo [13].

Las HRG son un formalismo para la generación de lenguajes de hipergrafos, las HRG son utilizadas en diferentes áreas, tales como comprensión y generación de lenguaje natural, traducción automática basada en semántica, así como en teoría de juegos, bases de datos, inteligencia artificial, entre otras [15, 16, 17, 7].

Es bien sabido que el problema de membresía para HRG en general es intratable [1, 12] sin embargo, se han presentado algunos resultados para resolver el problema de membresía. En [13] se presenta una versión extendida "Top-Down" para *HRG* del algoritmo CKY de grámaticas libres de contexto para cadenas sobre hipergrafos de grado acotado, los cuales permiten ejecutar el algoritmo analizador en un tiempo polinomial.

Por otro lado [3] presenta un algoritmo para un analizador "Bottom-Up" en HRG representado a través de un sistema deductivo. El algoritmo se basa en una buena descomposición hiperarbórea que cumple ciertas restricciones, tales como, ancho y grado del hipergrafo acotado.

ISSN 2007-9737

También presenta un esquema de optimización que junto con las restricciones antes mencionadas permiten ejecutar el algoritmo analizador en un tiempo polinomial.

El algoritmo que presenta [15] solamente emplea gramáticas de grafos [5] para representar algunos aspectos del significado de una oración en inglés. Del mismo modo el algoritmo que presenta [8], dado que está diseñado para Gramáticas de Grafos Regulares [4], las cuales son una subfamilia de los Lenguajes de Reemplazo de Hiperaristas (HRL), por sus siglas en inglés.

El *Analizador* que aquí se propone resuelve el problema de membresía en HRG, para ello realiza el proceso de análisis sobre una representación matricial de H, en partícular sobre la matriz de adyacencias. De tal forma que se define un nuevo tipo de Matriz de Adyacencias (MA), como una herramienta más para el análisis en hipergrafos.

En el *Analizador* no se utiliza la definición de MA para grafos, es por ello, que se ha propuesto una definición para la construcción de la MA, correspondiente al hipergrafo de entrada H. Cada fila de la MA se forma con conjuntos, los cuales representan las etiquetas de las hiperaristas de cada vértice del hipergrafo H.

El *Analizador* tiene una complejidad polinomial de orden $O(l^5)$, donde l es el número de vértices del hipergrafo de entrada H

Las contribuciones que se presentan en este artículo son: el diseño del *Analizador*, cuya complejidad es polinomial y que lleva a cabo el análisis sobre la MA del hipergrafo de entrada H, para resolver el problema de membresía en HRG. Así como la definición de la MA del hipergrafo de entrada y la demostración de que el *Analizador* es correcto.

El artículo se encuentra organizado de la siguiente manera, en la sección 2 se abordan algunos conceptos relacionados con las HRG, tales como, mecanismo de reemplazo de hiperaristas por hipergrafos, la relación de derivación y su cerradura reflexiva y transitiva de la relación de derivación. También se describe el lenguaje de hipergrafos.

En la sección 3 se presenta el *Analizador* que resuelve el problema de membresía en HRG, así como el desarrollo de un ejemplo que muestra el funcionamiento del *Analizador*, también se presenta la demostración de que el *Analizador* es correcto y el análisis de la complejidad del *Analizador*. Finalmente en la sección 4 se presentan los principales resultados obtenidos, así como las aportaciones obtenidas de este artículo.

2. Hipergrafos y Gramáticas de Reemplazo de Hiperaristas

Un hipergrafo es una generalización de un grafo, donde cada **hiperarista** e [15] puede relacionar cualquier subconjunto de vértices denominados vértices adjuntos, incluyendo al conjunto vacío. Un hipergrafo se forma de un conjunto de vértices junto con un número variado de hiperaristas dirigidas. [11, 18, 6].

Definición 1 Un hipergrafo con hiperaristas etiquetadas y dirigidas es una terna $H = (V, E, lab)$, donde V es un conjunto de vértices, cada $e \in E$ es un par (R_e, D_e), tal que $R_e \subseteq V$ es el origen de e y $D_e \subseteq V \setminus R_e$ es el destino, C es un conjunto numerable de etiquetas y lab es una función de E en C.

Definición 2 Un hipergrafo con hiperaristas etiquetadas y dirigidas $H = (V, E, lab)$ es **enraizado** si:

— H es acíclico.

— $V = \{v_R\} \cup N$, $\{v_R\} \cap N = \emptyset$, v_R es un vértice distinguido que se le llama nodo raíz del hipergrafo H. N es el conjunto de vértices que no son raíz.

Cuando hablemos de hipergrafo, entiéndase que es un hipergrafo enraizado con hiperaristas etiquetadas y dirigidas.

Definición 3 [9] Sea $H = (V, E, lab)$ un hipergrafo, sea V un conjunto de vértices y $a, b \in V$. Entonces a es un vértice adyacente a b si existe una hiperarista $e \in E$ tal que $a \in R_e$ y $b \in D_e$.

2.1. Reemplazo de Hiperaristas por Hipergrafos

En esta sección, se presentan las definiciones del proceso de derivación de hipergrafos y reemplazo de una hiperarista por un hipergrafo, elementos necesarios para abordar el concepto de la HRG, el cual es fundamental en el diseño del Analizador.

Los vértices externos se definen como una lista ordenada de vértices distintos $ext \in V^*$. El vértice raíz se define como un vértice designado como la raíz del hipergrafo y cada hiperarista es dirigida desde esta raíz, ambos elementos específican cómo reemplazar una hiperarista por un hipergrafo.

Las HRG permiten manipular hipergrafos mediante la sustitución de hiperaristas. Una hiperarista $e \in H$ es reemplazada por un hipergrafo H', a través del vértice raíz y los vértices externos, inicialmente la arista e es removida y los vértices externos de H' son mapeados con los vértices adjuntos de e, es decir el i-ésimo y j-ésimo vértice externo de la hiperarista e es mapeado con el i-ésimo y j-ésimo vértice adjunto en el hipergrafo H'.

Definición 4 [18] Sean $\mathcal{H}_C$ la clase de todos los hipergrafos sobre el conjunto de etiquetas C. Sean $H \in \mathcal{H}_C$ un hipergrafo y $B \subseteq E(H)$ un conjunto de hiperaristas a ser reemplazadas. Sea $repl : B \to \mathcal{H}_C$ una función que realiza la operación de reemplazo de la siguiente manera.

El **reemplazo** de B en H hecho por $repl$ produce el hipergrafo $H[repl]$ que se obtiene al quitar B de E_H, agregando disjuntamente los vértices e hiperaristas de $repl(e)$ por cada $e \in B$ y empatando el i-ésimo vértice externo con el i-ésimo vértice adjunto de e para cada $e \in B$. Todas las hiperaristas mantienen sus etiquetas y vértices adyacentes; los vértices externos de $H[repl]$ son los de H. Si $B = \{e_1, \ldots, e_n\}$ y $repl(e_i) = R_i$, para $i = 1, \ldots, n$, entonces se escribe $H[e_1/R_1, \ldots, e_n/R_n]$ en lugar de $H[repl]$.

La Figura 1 inciso b) muestra el reemplazo de la hiperarista Y por el hipergrafo H que se muestra en el inciso a).

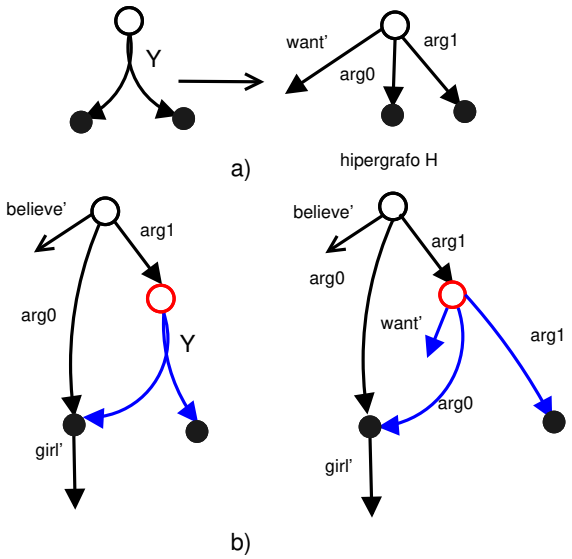

Fig. 1. a) Hipergrafo H. b) Reemplazo de hiperarista Y por H (tomada de [3])

2.2. Gramáticas de Reemplazo de Hiperaristas

Las HRG son elementos que pueden ser empleados para la generación y análisis de la representación semántica apoyada en hipergrafos [10]. En la Figura 2 inciso a) se muestra una HRG.

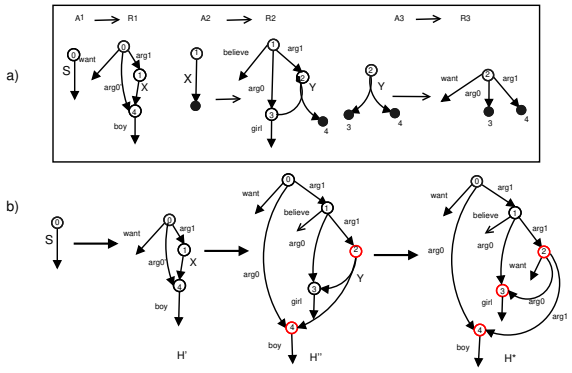

Fig. 2. a) Reglas de producción para G. b) Derivación a partir de G para el hipergrafo H^* que representa el significado "The boy wants the girl to believe that he wants her" (tomada de [3])

ISSN 2007-9737

Definición 5 [15] **Una Gramática de Reemplazo de Hiperaristas** (HRG) es una tupla $G = (N, T, S, P)$ donde:

— N es un conjunto finito de símbolos no terminales.

— T es un conjunto finito de símbolos terminales.

— $S \in N$ es un símbolo no terminal inicial.

— P es un conjunto finito de reglas de producción de la forma $p : A \rightarrow R$, donde $A \in N$, y R es un hipergrafo cuyas hiperaristas están etiquetadas por símbolos de $T \cup N$.

2.3. Derivación

Sean $e \in E$ una hiperarista etiquetada con A y R un hipergrafo. El elemento principal de las HRG es el conjunto finito de reglas de producción, se dice que estas controlan el proceso de derivación pues determinan la hiperarista e que debe ser reemplazada por un hipergrafo. Una regla de producción de las HRG es de la forma $A \rightarrow R$ para la hiperarista e que será reemplazada. A es llamado el lado izquierdo de $A \rightarrow R$ y R es llamado el lado derecho de $A \rightarrow R$. Para ser más precisos se tiene la siguiente definición.

Definición 6 [15] Sea N un conjunto de símbolos no terminales. Una **regla de producción** sobre N es de la forma $p = A \rightarrow R$, $A \in N$ y R es un hipergrafo cuyas hiperaristas están etiquetadas por símbolos de $N \cup T$.

La definición 6 se tomó de [15], sin embargo para el propósito en este artículo A corresponde a la etiqueta del hipergrafo del lado izquierdo de la regla de producción p.

El proceso de derivación inicia con el símbolo inicial S, y de forma repetida se elige un símbolo A no terminal para ser reemplazado por el hipergrafo R, a través de alguna regla de producción $A \rightarrow R$. A su vez el hipergrafo R puede tener otras hiperaristas etiquetadas con símbolos no terminales, de tal manera que el proceso de reemplazo continúa hasta que todas las hiperaristas estén etiquetadas con símbolos terminales [2].

Definición 7 [2] Sea G una HRG y $A \rightarrow R$ una regla de producción de G . La relación $H' \Rightarrow H^*$ (H^* es derivada de H' en un solo paso) se define de la siguiente forma. H' debe tener una hiperarista e etiquetada con A; sean $v_1, ..., v_k$ vértices conectados de la hiperarista. Sean $u_1, ..., u_k$ los vétices externos de R. Entonces H^* es el hipergrafo formado después de haber removido e de H', haciendo una copia isomorfa de R, e identificando v_i con las copias de u_i para cada $i = 1, ..., k$.

Sea $\overset{*}{\Rightarrow}$ la cerradura reflexiva y transitiva de $\Rightarrow$. El lenguaje genreado por G que usualmente se denota por $L(G)$ es igual al conjunto de todas las H, tales que S deriva en o o más pasos a H y donde H solamente consta de hiperaristas etiquetadas con símbolos terminales.

La Figura 2 inciso b) muestra un ejemplo del proceso de derivación a partir de G para el hipergrafo de entrada H, el cual representa "The boy wants the girl to believe that he wants her".

Inicialmente, se aplica la regla de producción $A_1 \rightarrow R_1$ para reemplazar el hipergrafo cuya hiperarista está etiquetada con el símbolo inicial S, por el hipergrafo R_1, con lo que se obtiene el hipergrafo $H^{'}$. Posteriormente en $H^{'}$ sobre la base de la regla de producción $A_2 \rightarrow R_2$ se reemplaza la hiperarista X por el hipergrafo R_2, con lo que se obtiene el hipergrafo $H^{''}$, enseguida se realiza una copia isomorfa de R_2, considerando un mapeo entre los vértices 1,4 conectados a X y los vértices externos 1,4 de R_2.

Finalmente, en $H^{''}$ sobre la base de la regla de producción $A_3 \rightarrow R_3$ se reemplaza la hiperarista Y por el hipergrafo R_3, con lo que se obtiene el hipergrafo H^*, enseguida se realiza una copia isomorfa de R_3, considerando un mapeo entre los vértices 2,3,4 conectados a Y y los vértices externos 2,3,4 de R_3.

Se muestra que en el hipergrafo resultante H^* todas las hiperaristas están etiquetadas solamente con símbolos terminales, por lo tanto se concluye que el hipergrafo de entrada H^*, sí es derivado a partir del hipergrafo $H^{'}$, a través de la aplicación de las reglas de producción de G.

Una vez presentado el concepto de la HRG, así como el proceso de derivación, se puede abordar

el concepto de lenguaje de hipergrafos, el cual se define como el conjunto de todos los hipergrafos H, de tal forma que cada uno contiene solamente hiperaristas etiquetadas con símbolos terminales y que además estos hipergrafos pueden ser derivados a partir del símbolo S aplicando producciones de P, en un número finito de pasos $S \overset{*}{\Rightarrow} H$ [14].

Definición 8 [11] **El lenguaje de hipergrafos** $L(HRG)$ generado por HRG consiste de todos los hipergrafos etiquetados con símbolos terminales, los cuales se pueden derivar de A aplicando producciones de P, donde $A \in \mathcal{H}_C$ es un hipergrafo.

$L(HRG) = \{H \in \mathcal{H} | A \Rightarrow_P^* H\}$.

3. Analizador

En esta sección se presenta el *Analizador* para resolver el problema de membresía en HRG. Este lleva a cabo el análisis en la MA del hipergrafo de entrada H. La MA para el hipergrafo de entrada H y para cada hipergrafo R de cada una de las reglas de producción de la HRG, son creadas antes de ejecutar el *Analizador*.

3.1. Matriz de Adyacencias

En el *Analizador* no se utiliza la definición de Matriz de Adyacencias (MA) para grafos, sino una en la cual las etiquetas de las hiperaristas de cada vértice quedan representadas en cada uno de las entradas de la matriz.

La MA de un hipergrafo, es una matriz cuadrada, donde los vértices de H indexan a cada fila y columna de la matriz bajo un orden arbitrario. Sin embargo el orden elegido debe ser consistente, es decir indexar las filas y columnas bajo el mismo orden.Ésto se puede ver representado en la definición 9.

Cuando hablemos de M_H, entiéndase que es la matriz de adyacencias del hipergrafo H. De igual forma, cuando hablemos de $M_{etiq(A \to R)}$, entiéndase que es la matriz de adyacencias del hipergrafo R.

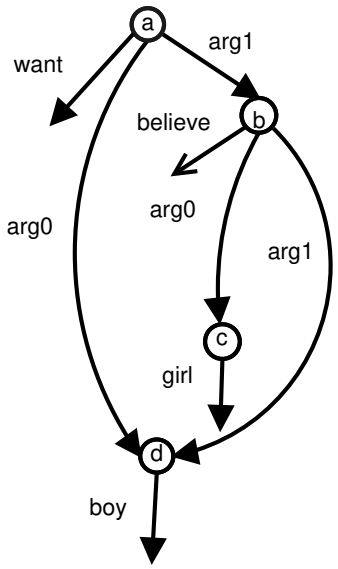

Fig. 3. Hipergrafo H (tomado de [15])

Definición 9 Sea H un hipergrafo con hiperaristas etiquetadas y dirigidas y sea MA una matriz cuadrada con entradas a_{ij}, definidas por $a_{ij} = lab(e)$ donde $i, j \in V(H)$:
$a_{ij} = lab(e)$ donde $i \in R_e, j \in D_e$ y $e \in E(H)$.

Para el propósito de este artículo, formamos la MA de la siguiente manera. Con base a la dirección de cada hiperarista dirigida, de tal forma que la primera fila, corresponde con el conjunto formado por el vértice raíz, enseguida la segunda fila, corresponde con el conjunto formado por el vértice adyacente a las siguientes raíces, ubicadas en el hipergrafo, partiendo de arriba hacia abajo y de izquierda a derecha, y así sucesivamente hasta haber representado en la MA a todas las hiperaristas del hipergrafo.

En las figuras 3 y 4, se muestra el hipergrafo de entrada H con su M_H formada por cinco filas y cinco columnas. La primera fila tiene 3 conjuntos $\neq \emptyset$, los cuales representan las etiquetas de las hiperaristas adyacentes al vértice a.

Las figuras 6 a 11 muestran las M_R para el hipergrafo R de las reglas X_0, X_3, X_2 y X_1 de la HRG dada.

El *Analizador* recibe como entrada la M_H de H, el conjunto de las M_A de los hipergrafos R

para cada una de las reglas de producción de la HRG y la M_S de R para la regla inicial $S \rightarrow R$. El *Analizador* no utiliza los vértices externos, debido a que esta parte del hipergrafo de entrada H y los vértices externos se requieren solamente durante el proceso de derivación.

Cuando hablemos de empatar dos filas, entiéndase que nos referimos a que los conjuntos de ambas filas se intersectan, esto significa que las etiquetas de las hiperaristas del hipergrafo representado por éstas, son las mismas que las del hipergrafo R de la regla.

$$\begin{array}{c} \\ \{a\} \\ \{b\} \\ \{c\} \\ \{d\} \\ \infty \end{array} \begin{array}{c} \begin{array}{ccccc} \{a\} & \{b\} & \{c\} & \{d\} & \infty \end{array} \\ \begin{pmatrix} \{\} & \{arg_1\} & \{\} & \{arg_0\} & \{want\} \\ \{\} & \{\} & \{arg_0\} & \{arg_1\} & \{believe\} \\ \{\} & \{\} & \{\} & \{\} & \{girl\} \\ \{\} & \{\} & \{\} & \{\} & \{boy\} \\ \{\} & \{\} & \{\} & \{\} & \{\} \end{pmatrix} \end{array}$$

Fig. 4. M_H del hipergrafo de la figura 3

Los algoritmos *Analizador* y *Empata* van creando una copia de las filas de M_H y de la M_R del hipergrafo R conforme se va realizando el análisis, debido a que durante el proceso de análisis los conjuntos que se intersectaron podrían ser $\emptyset$ y en caso de que las filas no empaten entre ellas, se tiene que buscar una nueva regla, por lo que es necesario recuperar los conjuntos de ambas filas,

4. Algoritmo *Analizador*

El algoritmo *Analizador* consta de tres iteraciones anidadas.

1. La iteración más externa (líneas 2 a 24) toma cada fila f_{H_i} de la matriz de adyacencias M_H del hipergrafo de entrada H.

2. [1] La iteración intermedia (líneas 2 a 37) toma cada matriz de adyacencias M_{R_j} de R para la reglas de producción, cuyas etiquetas se han almacenado en la pila.

Algorithm 1 $Empata(f_{AH}, M_R)$

```
1: Salida: {V_e, V_n, V_i, V_d, V_p}
2: f_R ← next(f_R, M_R)
3: Mientras(M_R ≠ ∅)
4:    f_AR ← f_R
5:    empataf = V_n
6:    Para cada conjunto x ≠ ∅ de f_AH
7:       i ← false
8:       Para cada conjunto y ≠ ∅ de f_AR
9:          si (x ∩ y) ≠ ∅
10:            x ← x − (x ∩ y)
11:            y ← y − (x ∩ y)
12:            i ← true
13:            rompeCiclo
14:      continua
15:      si ¬i
16:         rompeCiclo
17:      continua
18:      Para cada y ≠ ∅ de f_AR
19:         Sea V ∈ y
20:         si V ∈ T
21:            f_R ← next(f_R, M_R) ir a 3
22:         si V = etiq(M_R)
23:            empataf ← V_¬i
24:         en otro caso
25:            empataf ← V_¬d
26:         rompeCiclo ir a 35
27:         y ← y − {V}
28:      continua
29:   si i
30:      empataf ← Ve
31:      rompeCiclo
32:   en otro caso
33:      empataf ← V_p
34: continua
35: regresa empataf
```

3. [2]La iteración más interna (líneas 2 a 35) toma cada fila f_{R_k} de la matriz de adyacencias M_{R_j} para ver si empata con la fila f_{H_i}, aquí tenemos tres casos:

   a) f_{R_k} y f_{H_i} empatan (líneas 6 a 17 y 30 a 31), entonces continua con el ciclo más externo.

[1] Esta iteración corresponde al algoritmo *Procesa*

[2] Esta iteración corresponde al algoritmo *Empata*

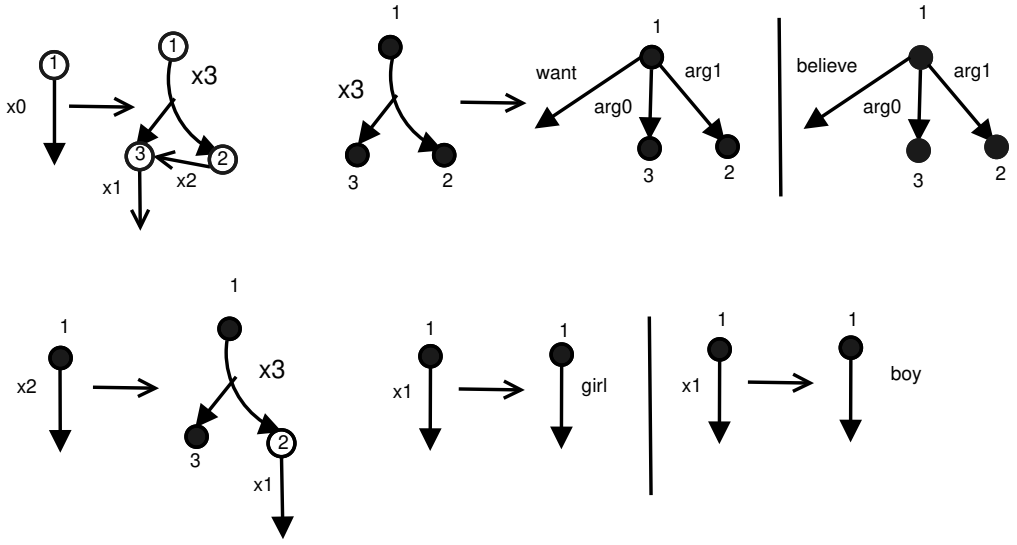

Fig. 5. HRG (tomadas de [15])

$$\begin{array}{c} \\ \{1\} \\ \{2\} \\ \{3\} \\ \infty \end{array} \begin{array}{c} \begin{array}{cccc} \{1\} & \{2\} & \{3\} & \infty \end{array} \\ \begin{pmatrix} \{\} & \{X_3\} & \{X_3\} & \{\} \\ \{\} & \{\} & \{X_2\} & \{\} \\ \{\} & \{\} & \{\} & \{X_1\} \\ \{\} & \{\} & \{\} & \{\} \end{pmatrix} \end{array}$$

Fig. 6. M_{X0} para el hipergrafo de la figura 5

$$\begin{array}{c} \\ \{1\} \\ \{2\} \\ \{3\} \\ \infty \end{array} \begin{array}{c} \begin{array}{cccc} \{1\} & \{2\} & \{3\} & \infty \end{array} \\ \begin{pmatrix} \{\} & \{arg_1\} & \{arg_0\} & \{want\} \\ \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} \end{pmatrix} \end{array}$$

Fig. 7. M_{X3} para el hipergrafo de la figura 5

$$\begin{array}{c} \\ \{1\} \\ \{2\} \\ \{3\} \\ \infty \end{array} \begin{array}{c} \begin{array}{cccc} \{1\} & \{2\} & \{3\} & \infty \end{array} \\ \begin{pmatrix} \{\} & \{arg_1\} & \{arg_0\} & \{believe\} \\ \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} \end{pmatrix} \end{array}$$

Fig. 8. M_{X3} para el hipergrafo de la figura 5

b) f_{R_k} y f_{H_i} no empatan (líneas 6 a 17 y 19 a 21) y f_{R_k} no contiene variables, entonces continua con el ciclo más interno.

c) f_{R_k} y f_{H_i} no empatan (líneas 6 a 17 y 22 a 26) y f_{R_k} contiene las variables $V_1, \ldots, V_m$, entonces para cada $V_l, 1 \leq l \leq m$ continua con el ciclo intermedio.

El algoritmo *Procesa* utiliza una pila, de tal forma que durante el análisis, se van almacenando las

$$\begin{array}{c} \\ \{1\} \\ \{2\} \\ \{3\} \\ \infty \end{array} \begin{array}{c} \begin{array}{cccc} \{1\} & \{2\} & \{3\} & \infty \end{array} \\ \begin{pmatrix} \{\} & \{X_3\} & \{X_3\} & \{\} \\ \{\} & \{\} & \{\} & \{X_1\} \\ \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} \end{pmatrix} \end{array}$$

Fig. 9. M_{X2} para el hipergrafo de la figura 5

$$\begin{array}{c} \\ \{1\} \\ \infty \end{array} \begin{array}{c} \begin{array}{cc} \{1\} & \infty \end{array} \\ \begin{pmatrix} \{\} & \{boy\} \\ \{\} & \{\} \end{pmatrix} \end{array}$$

Fig. 10. M_{X1} para el hipergrafo de la figura 5

$$\begin{array}{c} \\ \{1\} \\ \infty \end{array} \begin{array}{c} \begin{array}{cc} \{1\} & \infty \end{array} \\ \begin{pmatrix} \{\} & \{girl\} \\ \{\} & \{\} \end{pmatrix} \end{array}$$

Fig. 11. M_{X1} para el hipergrafo de la figura 5

etiquetas de todas las variables encontradas en la fila f_R de la matriz de adyacencias M_R que está siendo analizada, posteriormente estas se irán desempilando una a una conforme se van empatando las filas f_R de la matriz M_R y f_H de la matriz M_H.

El funcionamiento del Algoritmo 3 se muestra para la primera fila de la matriz M_H de la figura 4 junto con las M_{R_j} de las reglas de producción de la HRG, las cuales se muestran en las figuras 6 a 11.

En la iteración más externa, toma la fila {a} de la matriz M_H, para que enseguida tome la matriz de adyacencias correspondiente a la regla del símbolo inicial M_{X0} del hipergrafo R de la regla X_0 y en la línea 9 llame al algoritmo *Empata*$(\{a\}, M_{X0})$, con el propósito de que en la iteración más interna, verifique si la fila $\{1\}$ de M_{X0} empata con la fila {a} de M_H.

Así que el algoritmo *Empata*$(\{a\}, M_{X0})$ en las líneas 22 a 26 regresa la etiqueta de la variable X_3 contenida en la fila {1}, enseguida en la iteración intermedia, el algoritmo *Procesa*$(\{a\}, C_R, P_{Var}, X_3, M_{X0})$, en las líneas 19 a 20 apila la variable X_3 en la pila P_{Var}, enseguida, el algoritmo *Empata*$(\{a\}, M_{X0})$ en la iteración más interna en las líneas 8 a 14 y 32 a 33 valida que todos los conjuntos de la fila $\{1\}$ ya están vacíos, es decir que ya no hay más variables para apilar, entonces el algoritmo *Procesa*$(\{a\}, C_R, P_{Var}, V_p, M_{X0})$ en la iteración intermedia, en las líneas 22 a 26 toma la matriz M_{X3}.

Algorithm 2 *BuscaMA*$(CR, PVar[tope])$

1: Salida: $\{\emptyset, M_A\}$
2: $M_{AR} \leftarrow 0$
3: Para cada M'_R de C_R
4: si $etiq(M'_R) = PVar[tope]$
5: $M_{AR} \leftarrow M'_R$
6: regresa (M_{AR})

Posteriormente, el algoritmo *Empata*$(\{a\}, M_{X3})$ en la iteración más interna toma la fila $\{1\}$ de la matriz de adyacencias M_{X3} para que enseguida en la iteración que inicia en la fila 8 a 14, tome los conjuntos $\{arg_1\}$, $\{arg_0\}$ y $\{want\}$ de la fila {a}

Algorithm 3 *Analizador*(M_H, C_R, S)

1: Salida: $\{true, false\}$
2: $q \leftarrow 1$
3: $M_R \leftarrow S$
4: $push(P_{Var}[tope], S)$
5: Mientras $(q-1) < |M_H|$
6: $f_H \leftarrow next(f_H, M_H)$
7: $f_{AH} \leftarrow f_H$
8: $empatafH \leftarrow false$
9: $res \leftarrow Empata(f_{AH}, M_R)$
10: $R \rightarrow Procesa(f_H, C_R, P_{Var}, res, M_R)$
11: si $R = V_f$
12: $empatafH \leftarrow true$
13: $proc[q] \leftarrow f_H$
14: $q \leftarrow q + 1$
15: ir a 5
16: si $R = V_m$
17: ir a 7
18: si $R = V_n$
19: $empatafH \leftarrow false$
20: rompeCiclo
21: en otro caso
22: ir a 9
23: continua
24: regresa $empatafH$

y válida que se intersectan con $\{arg_1\}$, $\{arg_0\}$ y $\{want\}$ de la fila $\{1\}$, a su vez a cada conjunto de ambas filas les va restando la intersección de ambos, como se muestra en la figura 12.

Posteriormente, en la iteración intermedia en las líneas 4 a 5 y 16 a 18 el algoritmo *Procesa*$(\{a\}, C_R, P_{Var}, V_e, M_{X3})$ desempila a X_3 para tomar el siguiente elemento en la pila es decir, X_0 y en seguida el algoritmo *Analizador*(M_H, C_R, S) en las líneas 13 a 17 marca la fila {a} como ya procesada.

Nuevamente, en la iteración más externa tomaría la siguiente fila a procesar, es decir la fila $\{b\}$ de M_H, para que en la iteración intermedia tome la fila 2 de X_0.

4.1. Demostración

Demostraremos que el algoritmo *Analizador* es correcto, pero primero se demuestra una proposición más general.

Algorithm 4 *Procesa*$(f_H, C_R, P_{Var}, res, M_R)$

```
1: Salida: {M_R, V_f, V_m, V_n}
2: fin ← true resp ← V_p
3: Mientras ¬vacia(P_Var) or empatafH = false
4:     si res = V_e
5:       pop(P_Var[tope])
6:       Para cada conjunto x de f_AH
7:         si x ≠ ∅ fin ← false
8:           rompeCiclo
9:       si fin and P_Var[tope] = res_-d
10:          Mientras P_Var[tope] ≠ res_-i
11:            pop(P_Var[tope])
12:            resp ← V_m
13:          ir a 37
14:      continua
15:      resp ← V_f ir a 36
16:      M_R ← BuscaMA(C_R, P_Var[tope])
17:      pop(P_Var[tope])
18:     push(P_Var, P_Var[tope]_-i)
19:     si res = V and res ≠ P_Var[tope]
20:      push(P_Var, res)
21:     si res = V_p
22:      M_R ← BuscaMA(C_R, P_Var[tope])
23:      si P_Var[tope] ≠ P_Var[tope]_-i
24:        pop(P_Var[tope])
25:        push(P_Var[tope]_-i)
26:      ir a 37
27:     si res = V_n
28:      Mientras(eti(M_R) ≠ P_Var[tope] or C_R ≠ ∅)
29:        M_R ← next(M_R, C_R)
30:        si (eti(M_R) = P_Var[tope])
31:          M_R ← BuscaMA(C_R, P_Var[tope])
32:          ir a 37
33:        en otro caso
34:          resp ← V_n
35:          ir a 37
36:        continua
37: regresa resp
```

Proposición 1 Sean $H \in \mathcal{H}_C$ y HRG una gramática de reemplazo de hiperaristas, entonces $H \in L_A(HRG)$ si y sólo si $Analizador(M_H, C_R, A)$.

La demostración se divide en dos casos:

$H \in L_A(HRG)$**:** Se realiza una demostración por inducción sobre el número k de derivaciones.

$$\boldsymbol{f_{AH}} = \{a\} \begin{pmatrix} \{a\} & \{b\} & \{c\} & \{d\} & \infty \\ \{\} & \{arg_1\} & \{\} & \{arg_0\} & \{want\} \end{pmatrix}$$

$$\boldsymbol{M_{X3}(f_{AR})} = \{1\} \begin{pmatrix} \{1\} & \{2\} & \{3\} & \infty \\ \{\} & \{arg_1\} & \{arg_0\} & \{want\} \end{pmatrix}$$

$$\boldsymbol{f_{AH}} = \{a\} \begin{pmatrix} \{a\} & \{b\} & \{c\} & \{d\} & \infty \\ \{\} & \{\} & \{\} & \{\} & \{\} \end{pmatrix}$$

$$\boldsymbol{M_{X3}(f_{AR})} = \{1\} \begin{pmatrix} \{1\} & \{2\} & \{3\} & \infty \\ \{\} & \{\} & \{\} & \{\} \end{pmatrix}$$

Fig. 12. Empatando fila {a} con fila {1}

$k = 1 : A \Rightarrow H$, en este caso se tiene el hipergrafo H de entrada y se reconoce en un solo paso, en cuyo caso el algoritmo *Analizador*(M_H, C_R, A) realiza lo siguiente, toma la matriz de adyacencias M_H del hipergrafo de entrada H, la cual tiene la siguiente forma

$$\begin{array}{c} \\ n_1 \\ n_2 \\ \dots \\ n_{m-1} \\ \infty \end{array} \begin{array}{c} \begin{array}{ccccc} n_1 & n_2 & \dots & n_{n-1} & \infty \end{array} \\ \begin{pmatrix} h_{11} & h_{12} & \dots & h_{1n-1} & h_{1n} \\ h_{21} & h_{22} & \dots & h_{2n-1} & h_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ h_{m-11} & h_{m-12} & \dots & h_{m-1n-1} & h_{m-1n} \\ h_{m1} & h_{m2} & \dots & h_{mn-1} & h_{mn} \end{pmatrix} \end{array}$$

Fig. 13. Forma de M_H

y la matriz de adyacencias M_R del hipergrafo R de la regla de producción $A \to R$, la cual tiene la siguiente forma

$$\begin{array}{c} \\ n_1 \\ n_2 \\ \dots \\ n_{s-1} \\ \infty \end{array} \begin{array}{c} \begin{array}{ccccc} n_1 & n_2 & \dots & n_{t-1} & \infty \end{array} \\ \begin{pmatrix} r_{11} & r_{12} & \dots & r_{1t-1} & r_{1t} \\ r_{21} & r_{22} & \dots & r_{2t-1} & h_{2t} \\ \dots & \dots & \dots & \dots & \dots \\ r_{s-11} & r_{s-12} & \dots & r_{s-1t-1} & r_{s-1t} \\ r_{s1} & r_{s2} & \dots & r_{st-1} & r_{st} \end{pmatrix} \end{array}$$

Fig. 14. Forma de M_R

donde $m = s$, $n = t$.

En este caso, ambas matrices se empatan en todas sus filas, pues de acuerdo a la iteración más externa, en las líneas 2 a 24: toma cada una de las filas de M_H para que en la iteración más interna tome cada fila de M_R y en las líneas 6 a 14 y 29 a 31: verifique que ambas filas se empatan pues los conjuntos $h_{ij} = r_{ij}$, donde $i = 1, ..., m$ y $j = 1, ..., n$. Así que *Analizador* regresa el valor $true$.

La hipótesis de inducción es $A \Rightarrow^k H$ si y sólo si *Analizador*(M_H, C_R, A).

$k = i + 1$: Si el número de derivaciones es $i + 1$ entonces $A \Rightarrow^i H'$, por lo tanto debe existir una hiperarista X y una producción de la forma $X \rightarrow R$, de tal manera que $A \Rightarrow^i H'[X/R] \Rightarrow H$. Por hipótesis de inducción *Analizador*(M_H, C_R, A) regresaría $true$, pero como H' contiene una hiperarista etiquetada X y HRG contiene una regla de producción de la forma $X \rightarrow R$, entonces el algoritmo *Analizador*(M_H, C_R, A) haría lo siguiente.

Toma la matriz de adyacencias $M_{H'}$ del hipergrafo de entrada H', la cual tiene la siguiente forma

$$\begin{array}{c} \\ n_1 \\ n_2 \\ ... \\ n_{k-2} \\ n_{k-1} \\ \infty \end{array} \begin{array}{c} \begin{array}{cccccc} n_1 & n_2 & ... & n_{l-2} & n_{l-1} & \infty \end{array} \\ \begin{pmatrix} - & - & ... & - & - & - \\ - & - & ... & - & - & - \\ ... & ... & ... & - & ... & ... \\ - & - & - & X & - & - \\ - & - & ... & - & - & - \\ - & - & ... & - & - & - \end{pmatrix} \end{array}$$

Fig. 15. Forma de $M_{H'}$

y la matriz de adyacencias M_R del hipergrafo R de la regla de producción $A \rightarrow R$, la cual tiene la siguiente forma

En este caso, $k - 1$ filas de la matriz $M_{H'}$ empatan con las s filas de M_R pues la fila n_{k-2} de $M_{H'}$ no empata con ninguna fila de M_R. Esto se puede observar en en la iteración más interna, en las lineas 6 a 17

$$\begin{array}{c} \\ n_1 \\ n_2 \\ ... \\ n_{s-1} \\ \infty \end{array} \begin{array}{c} \begin{array}{ccccc} n_1 & n_2 & ... & n_{t-1} & \infty \end{array} \\ \begin{pmatrix} r_{11} & r_{12} & ... & r_{1t-1} & r_{1t} \\ r_{21} & r_{22} & ... & r_{2t-1} & h_{2t} \\ ... & ... & ... & ... & ... \\ r_{s-11} & r_{s-12} & ... & r_{s-1t-1} & r_{s-1t} \\ r_{s1} & r_{s2} & ... & r_{st-1} & r_{st} \end{pmatrix} \end{array}$$

Fig. 16. Forma de M_R

y 24 a 26: donde comprueba que ambas matrices no se corresponden debido a que $M_{H'}$ contiene a la variable X. Así que ahora toma la matriz de adyacencias M_H para el hipergrafo de entrada H, la cual tiene la siguiente forma

$$\begin{array}{c} \\ n_1 \\ n_2 \\ ... \\ N_1 \\ N_2 \\ ... \\ N_{i-1} \\ \infty \\ N_{m-1} \\ \infty \end{array} \begin{array}{c} \begin{array}{cccccccc} n_1 & ... & n_1 & ... & n_{j-1} & \infty & n_{n-1} & \infty \end{array} \\ \begin{pmatrix} - & ... & - & - & - & - & - & - \\ - & ... & - & - & - & - & - & - \\ ... & ... & - & - & - & - & & \\ - & - & x_{11} & ... & x_{1j-1} & x_{1j} & - & - \\ - & - & x_{21} & ... & x_{2j-1} & x_{2j} & - & - \\ - & - & ... & ... & ... & ... & - & - \\ - & - & x_{i-11} & ... & x_{i-1j-1} & x_{i-1j} & - & - \\ - & - & x_{i1} & ... & x_{ij-1} & x_{ij} & - & - \\ - & ... & - & - & - & - & - & - \\ - & ... & - & - & - & - & - & - \end{pmatrix} \end{array}$$

Fig. 17. Forma de M_H

y en la iteración intermedia en las líneas 21 a 26: desempila a X y toma la matriz de adyacencias M_X para el hipergrafo R de la regla $X \rightarrow R$, la cual tiene la siguiente forma:

$$\begin{array}{c} \\ \{f_1\} \\ \{f_2\} \\ ... \\ \{f_{s-1}\} \\ \infty \end{array} \begin{array}{c} \begin{array}{ccccc} \{c_1\} & \{c_2\} & ... & \{c_{t-1}\} & \infty \end{array} \\ \begin{pmatrix} x_{11} & x_{12} & ... & x_{1t-1} & x_{1t} \\ x_{21} & x_{22} & ... & x_{2t-1} & x_{2t} \\ ... & ... & ... & ... & ... \\ x_{s-11} & x_{s-12} & ... & x_{s-1t-1} & x_{s-1t} \\ x_{s1} & x_{s2} & ... & x_{st-1} & x_{st} \end{pmatrix} \end{array}$$

Fig. 18. Forma de M_X

donde: $i = s$, $j = t$.

En este caso, ambas matrices se empatan en todas sus filas, pues de acuerdo a la iteración más externa, en las líneas 2 a 24: toma cada una de las filas $N_1, ...N_i$ de M_H para que en la iteración más interna tome cada fila de M_X y en las líneas 6 a 14 y 29 a 31: verifique que ambas filas se empatan pues los

conjuntos $x_{uv} = x_{st}$, donde $u = 1, ..., i$ y $v = 1, ..., j$.. Así que *Analizador* regresa el valor $true$.

$H \notin L_A(HRG)$**:** En este caso el algoritmo *Analizador*(M_H, C_R, A) hace lo siguiente, toma la matriz de adyacencias M_H del hipergrafo de entrada H y cada una de las matrices de adyacencias M_R del hipergrafo R de la regla de producción $A \rightarrow R$ y en la iteración más interna, en las líneas 6 a 17: comprueba que ambas matrices no se corresponden, entonces en la iteración intermedia en las líneas 21 a 26: desempoila cada una de las M_R restantes y en la iteración más externa en las líneas 5 a 23: comprueba que M_H no se corresponde con ninguna de las M_R, así entonces el proceso de análisis ya no puede continuar y regresa el valor $false$.

Ahora la demostración de la corrección del algoritmo *Analizador*(M_H, C_R, S).

Corolario 1 Sean $H \in \mathcal{H}_T$ y HRG una gramática de reemplazo de hiperaritas, entonces $H \in L_S(HRG)$ si y sólo si *Analizador*(M_H, C_R, S).

Demostración. La prueba se sigue directamente sustituyendo $H \in \mathcal{H}_T$ por H y el símbolo inicial S por A en el desarrollo de la demostración de la proposición 1.

4.2. Complejidad del Analizador

La complejidad en tiempo del *Analizador* es de orden polinomial y está dada en términos del número de vértices del hipergrafo de entrada H (denotado como l) y de una constante igual al número de reglas de la HRG (denotado por p). Primero realizaremos el análisis de la complejidad para el algoritmo *Empata*, en seguida para el algoritmo *Procesa* y finalmente para el algoritmo *Analizador*.

En líneas 6 a 17. El número de entradas de f_{AH} está representado por $|f_{AH}|$ y el número de entradas de f_{AR} por $|f_{AR}|$, donde cada entrada es un conjunto. Siendo $min(|x|, |y|)$ el tiempo que lleva ejecutar el bloque de sentencias de la línea 7 a 16. Es así que el tiempo necesario para ejecutar la iteración que inicia en la línea 6 y finaliza en la línea 17, se representa en la siguiente sumatoria:

$$t_1 = \sum_{i=1}^{|f_{AH}|} \sum_{j=1}^{|f_{AR}|} min(|x|, |y|).$$

Líneas 18 a 28. El número de entradas de f_{AR} está representado por $|f_{AR}|$, donde cada entrada es un conjunto y C_1 es el costo necesario para ejecutar el bloque de sentencias que inicia en la línea 19 a 27. Es así que el tiempo necesario para ejecutar la iteración que inicia en la línea 18 y finaliza en la línea 28, se representan en la siguiente sumatoria:

$$t_2 = \sum_{j=1}^{|f_{AR}|} C_1.$$

Líneas 3 a 35. El número de filas de cada regla M_R está representado por $|M_R|$. Es así que el tiempo requerido para ejecutar las iteración que inicia en la línea 3 y termina en la línea 35, se representa en la siguiente sumatoria:

$$t_3 = \sum_{k=1}^{|M_R|} (t_1 + t_2) = |M_R|(|f_{AH}||f_{AR}|min(|x|, |y|) + |f_{AR}|C_1).$$

Sabemos que el número de filas en M_R para cualquier regla no puede ser mayor que el número de vértices de H, del mismo modo sucede para el número de conjuntos de f_{AH}, al igual que para el número de conjuntos de f_{AR} y para $min(|x|, |y|)$. Por lo tanto:

$$t_3 = l(l^3 + lC_1) = l^4 + l^2C_1.$$

Entonces, la complejidad de *Empata* es de $O(l^4)$

La complejidad para el algoritmo *Procesa* se calcula de la siguiente forma.

En las líneas 6 a 14. El número de entradas de f_{AH} está representado por $|f_{AH}|$, donde cada entrada es un conjunto y C_2 es el costo necesario para ejecutar el bloque de sentencias que inicia en la línea 7 a 13. Es así que el tiempo necesario para ejecutar la iteración que inicia en la línea

6 y finaliza en la línea 14, se representa en la siguiente sumatoria:

$$t_4 = \sum_{i=1}^{|f_{AH}|} C_2.$$

Líneas 28 a 36. El número de reglas de CR es igual a la constante p y C_3 es el costo necesario para ejecutar el bloque de sentencias que inicia en la línea 29 a 35. Es así que el tiempo requerido para ejecutar las iteración que inicia en la línea 28 y termina en la línea 36, se representa en la siguiente sumatoria:

$$t_5 = \sum_{i=1}^{|CR|} C_3 = |CR|C_3 = pC_3.$$

Líneas 3 a 37. El número de variables por cada fila de cada regla M_R está representado por $|P_{Var}|$. Es así que el tiempo requerido para ejecutar las iteración que inicia en la línea 3 y termina en la línea 37, se representa en la siguiente sumatoria:

$$t_6 = \sum_{k=1}^{|P_{Var}|} (t_4 + t_5) = |P_{Var}|(|f_{AH}|C_2 + pC_3).$$

Sabemos que el número de variables en P_{Var} para cualquier fila no puede ser mayor que el número de vértices de H, del mismo modo sucede para el número de conjuntos de f_{AH}. Por lo tanto:

$$t_6 = l(lC_2 + pC3) = l^2C_2 + lpC_3.$$

Entonces, la complejidad de *Procesa* es de $O(l^2)$.

La complejidad para el *Analizador* se calcula de la siguiente forma.

Línea 5 a 24. El tiempo requerido para ejecutar las iteración que inicia en la línea 5 y termina en la línea 24, se representa en la siguiente sumatoria:

$$t_7 = \sum_{i=1}^{|M_H|} (t_3 + t_6) = l^4 + l^2C_1 + l^2C_2 + lpC_3.$$

Como el número de vértices de H es una cota superior para M_H, es decir $l > |M_H|$, entonces el tiempo total T para *Analizador* se calcula de la siguiente forma:

$$T = l(l^4+l^2C_1+l^2C_2+lpC_3) = l^5+l^3C_1+l^3C_2+l^2pC_3.$$

Por lo tanto, la complejidad de *Analizador* es de $O(l^5)$.

5. Conclusiones

Hemos presentado el diseño de un algoritmo *Analizador correcto* para resolver el problema de membresía en $HRGs$, en este algoritmo no se utilizó la definición de MA para grafos, es por ello, que se desarrolló un método para la construcción de la MA, en el cual las etiquetas de las hiperaristas de cada vértice, quedan representadas a través de conjuntos, que forman cada fila de la M_H correspondiente al hipergrafo de entrada H.

Hemos descrito el proceso de análisis del hipergrafo H a través de su representación matricial, de tal forma que se explota el concepto de matriz de adyacencias, como una herramienta más para el análisis de hipergrafos.

Hemos presentado la demostración de que el *Analizador* es correcto y tiene una complejidad polinomial de orden $O(l^5)$, donde l es el número de vértices del hipergrafo de entrada.

Referencias

1. **Aalbersberg, I., Rozenberg, G., Ehrenfeucht, A. (1986).** On the membership problem for regular dnlc grammars. Discrete Applied Mathematics, Vol. 13, No. 1, pp. 79–85. DOI: https://doi.org/10.1016/0166-218X(86)90070-3.
2. **Aguiñaga, S., Chiang, D., Weninger, T. (2018).** Learning hyperedge replacement grammars for graph generation. CoRR, Vol. abs/1802.08068.
3. **Chiang, D., Andreas, J., Bauer, D., Hermann, K. M., Jones, B., Knight, K. (2013).** Parsing graphs with hyperedge replacement grammars. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, pp. 924–932.

4. **Courcelle, B. (1991).** The monadic second-order logic of graphs V: on closing the gap between definability and recognizability. Theor. Comput. Sci., Vol. 80, No. 2, pp. 153–202. DOI: 10.1016/0304-3975(91)90387-H.
5. **Drewes, F., Kreowski, H.-J., Habel, A. (1997).** Handbook of graph grammars and computing by graph transformation. chapter Hyperedge Replacement Graph Grammars. World Scientific Publishing Co., Inc., River Edge, NJ, USA, pp. 95–162.
6. **Engelfriet, J. (1997).** Handbook of formal languages, vol. 3. chapter Context-free Graph Grammars. Springer-Verlag New York, Inc., New York, NY, USA, pp. 125–213.
7. **Gallo, G., Scutellà, M. (1998).** Directed hypergraphs as a modelling paradigm. Decisions in Economics and Finance, Vol. 21, pp. 97–123. DOI: 10.1007/BF02735318.
8. **Gilroy, S., Lopez, A., Maneth, S. (2017).** Parsing graphs with regular graph grammars. Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017), Association for Computational Linguistics, Vancouver, Canada, pp. 199–208. DOI: 10.18653/v1/S17-1024.
9. **Gottlob, G., Grohe, M., Musliu, N., Samer, M., Scarcello, F. (2005).** Hypertree decompositions: Structure, algorithms, and applications. **Kratsch, D.**, editor, Graph-Theoretic Concepts in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 1–15.
10. **Groschwitz, J., Koller, A., Teichmann, C. (2015).** Graph parsing with s-graph grammars. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers, The Association for Computer Linguistics, pp. 1481–1490.
11. **Habel (1991).** Hyperedge Replacement: Grammars and Languages. University of Bremen, United States of America, 1st edition.
12. **Lange, K.-J., Welzl, E. (1987).** String grammars with disconnecting or a basic root of the difficulty in graph grammar parsing. Discrete Applied Mathematics, Vol. 16, No. 1, pp. 17–30. DOI: https://doi.org/10.1016/0166-218X(87)90051-5.
13. **Lautemann, C. (1990).** The complexity of graph languages generated by hyperedge replacement. Acta Informatica, Vol. 27, No. 5, pp. 399–421. DOI: 10.1007/BF00289017.
14. **Minas, M. (2000).** Hypergraphs as a uniform diagram representation model. Selected Papers from the 6th International Workshop on Theory and Application of Graph Transformations, TAGT'98, Springer-Verlag, London, UK, UK, pp. 281–295.
15. **Peng, X., Song, L., Gildea, D. (2015).** A synchronous hyperedge replacement grammar based approach for AMR parsing. Proceedings of the Nineteenth Conference on Computational Natural Language Learning, Association for Computational Linguistics, pp. 32–41. DOI: 10.18653/v1/K15-1004.
16. **Peuser, C. (2018).** From hyperedge replacement grammars to decidable hyperedge replacement games. **Mazzara, M., Ober, I., Salaün, G.**, editors, Software Technologies: Applications and Foundations - STAF 2018 Collocated Workshops, Toulouse, France, June 25-29, 2018, Revised Selected Papers, volume 11176 of Lecture Notes in Computer Science, Springer, pp. 463–478. DOI: 10.1007/978-3-030-04771-9_33.
17. **Rozenberg, G. (1997).** Handbook of graph grammars and computing by graph transformation.
18. **Rozenberg, G., Welzl, E. (1986).** Boundary NLC graph grammars basic definitions, normal forms, and complexity. Information and Control, Vol. 69, pp. 136–167. DOI: 10.1016/S0019-9958(86)80045-6.

Article received on 03/05/2021; accepted on 12/06/2021.
Corresponding author is Darnes Vilariño Ayala.