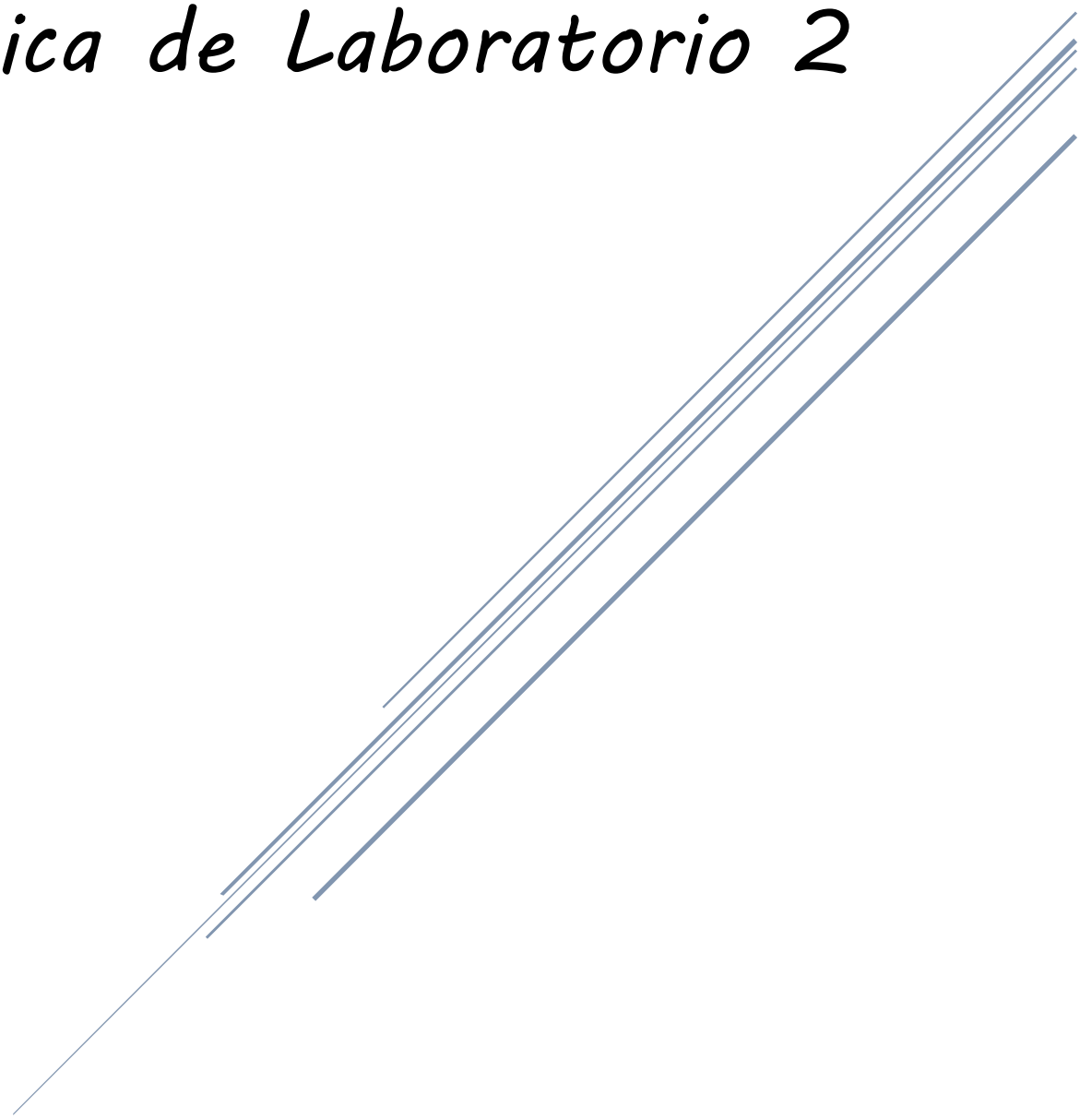
Conocimiento y Razonamiento Automatizado

Práctica de Laboratorio 2



Raúl Moratilla Núñez

Abel López Martínez

ÍNDICE

REPARTO DE TAREAS

El reparto de tareas en la práctica de laboratorio ha sido llevado a cabo de manera colaborativa por ambos integrantes del grupo. Durante el desarrollo de la práctica, se realizaron diversas tareas que fueron distribuidas equitativamente entre los miembros del equipo. En muchos casos, estas tareas fueron realizadas de manera simultánea, lo que permitió que se avanzara en la práctica de manera más eficiente y en un menor tiempo.

Es importante destacar que, en todo momento, se mantuvo una comunicación fluida y constante entre los integrantes del grupo para asegurarnos de que cada uno estaba realizando su tarea de manera adecuada y para coordinar los trabajos que se estaban llevando a cabo simultáneamente. Además, se fomentó el trabajo en equipo y la colaboración para que todos pudiéramos aprender de la experiencia y obtener los mejores resultados posibles.

En resumen, podemos afirmar que el reparto de tareas en la práctica de laboratorio fue realizado por ambos integrantes del grupo de manera equitativa y colaborativa, con muchas de ellas llevadas a cabo en trabajo simultáneo para optimizar el tiempo y obtener los mejores resultados.

GRADO DE CUMPLIMIENTO DE LOS REQUISITOS

Hemos completado todos los requisitos obligatorios de la práctica de manera satisfactoria. Hemos seguido cuidadosamente todas las instrucciones y criterios establecidos, prestando atención a los detalles y asegurándonos de cumplir con cada uno de ellos. Hemos trabajado juntos de manera rigurosa y metódica, lo que ha permitido lograr los resultados deseados.

ESTRUCTURA DE DATOS: COMPOUND

Para poder realizar análisis sintáctico de oraciones de todo tipo siguiendo la sintaxis del castellano, usaremos una estructura de dato nativa de prolog (es decir, no implementada por nosotros).

Un compound es una estructura de datos que consta de un functor y uno o más argumentos. El functor es un átomo que representa el nombre del compound y los argumentos pueden ser cualquier término en Prolog, incluyendo otros compounds. Por ejemplo, el compound "padre(juan, maria)" consta del functor "padre" y dos argumentos "juan" y "maria".

Los compounds se utilizan comúnmente en Prolog para representar objetos complejos y estructuras de datos, como listas y árboles. Por ejemplo, una lista de números enteros puede representarse como un compound con el functor "lista" y los argumentos que son los elementos de la lista: "lista(1,2,3,4,5)".

Por otro lado, un functor es una estructura de datos que consta de un átomo y un número de argumentos, pero a diferencia de los compounds, los functors no tienen nombres específicos para cada argumento. En cambio, se utilizan para definir predicados y relaciones en Prolog. Por ejemplo, el functor "mujer/1" define un predicado que es verdadero si su único argumento es una mujer.

Los functors se utilizan en Prolog para definir relaciones y predicados, lo que permite al programador definir la lógica de su programa en términos de reglas y hechos. Por ejemplo, la regla "madre(X,Y) :- mujer(X), padre(Z,Y)" define que X es la madre de Y si X es una mujer y Z es el padre de Y.

En resumen, tanto los compounds como los functors son tipos de datos utilizados en Prolog para estructurar la información y definir relaciones y predicados. Los compounds se utilizan para representar objetos complejos y estructuras de datos, mientras que los functors se utilizan para definir reglas y hechos en la lógica del programa.

ESTRUCTURA DEL CÓDIGO

La práctica se descompone en los siguientes archivos de código ejecutable en Prolog:

diccionarioESP.pl

Es un set de palabras en español diferenciadas por su tipo morfológico, es decir, determinantes (det), sustantivos (n), verbos(v), adjetivos (adj), conjunciones (conj), adverbios (adv) y preposiciones (prep). A parte, cada grupo morfológico tiene una regla al inicio, que permite definir el compound de ese tipo de palabra.

diccionarioTRA.pl

Es el mismo set de palabras que el anterior, pero además añadiendo la traducción al inglés y las reglas para pasar de una palabra en español a una en inglés.

draw.pl

Archivo Prolog descargado desde BlackBoard que implementa la impresión de un árbol de compounds, esto nos sirve para mostrar el árbol sintáctico que queda de analizar oraciones y así poder mostrar este análisis de una manera visual.

ejecutar.pl

Es el archivo base para ejecutar el resto de la parte principal, se compone de:

- Consults

Realización de consult al resto de archivos utilizados en este (equivalente a importar una librería o programa externo en otro leguaje). En este caso consult del archivo drawl.pl para poder dibujar el árbol sintáctico tras analizar las oraciones y consult de diccionarioESP.pl para acceder a todas las palabras de las que se dispone, así como de su tipo morfológico.

- Regla o_prueba
  Almacena mediante un índice y una lista de palabras cada una de las frases de entrada que se pedían como casos en el pdf de enunciado de la práctica

- Regla ejecutar_pruebas
  Itera para la frase i-ésima de la regla anterior, y para ella llama a ejecutar_oracion.

- Regla ejecutar_oracion
  Saca el análisis sintáctico mediante compounds, dibuja el árbol de salida llamando a draw, separa las oraciones compuestas en oraciones simples y las muestra

gramatica.pl

Las reglas gramaticales son las siguientes

- Regla oracion
- Regla compuesta
- Regla simple
- Regla coordinada
- Regla subordinada
- Regla complementos
- Regla g_verbal
- Regla g_nominal
- Regla g_adjetival
- Regla g_adverbial
- Regla g_preposicional

procesamiento.pl

- Regla aplanar_args
  Aplana el argumento N del compound de X (todo lo que tenga comp(...)) y los mete en Y.

- Regla aplanar_comp
  Aplana el compound X (todo lo que tenga comp(...)) y lo devuelve en Y.

- Regla aplanar_iter
  Hace una iteración de aplanar el compound X (todo lo que tenga comp(...)) y lo devuelve en Y.

- Regla args_aplanados
  Devuelve el número de argumentos que tendrá el compound aplanado.

- Regla add_args
  Añade a Y todos los compounds de X que no sean comp(...), los comp(...) los aplana y los añade a Y.

- Regla buscar_subordinada
  Dentro del árbol sintáctico pasado busca una oración de relativo (subordinada) tanto en sus nodos hijos como en sus nodos hermanos.

- Regla ajustar_compuestas_args
  Sirve para ayudar a iterar el árbol sintactico buscando si hay y en su caso donde se encuentra una posible oración de relativo, sirve de ayuda para la regla ajustar_compuestas y para ello hace uso de la regla anterior buscar_subordinada.

- Regla ajustar_compuestas
  Con las reglas sintácticas que hemos definido, no podemos ver donde hay una oración subordinada dentro de otra en una primera pasada, por lo que estas siempre se toman como oraciones simples, pese a ser compuestas, por lo que con esta función podemos realizar una pasada posterior recorriendo todo el árbol sintáctico, y con la ayuda de la regla anterior buscar si hay oraciones subordinadas para cambiar la oración padre de esta de simple a compuesta.

- Regla separar
  Otro de los requisitos de la práctica era “La simplificación de oraciones coordinadas, subordinadas de relativo o compuestas consistirá en la conversión de cualquiera de estas oraciones en tantas oraciones simples como se componga la original.”. Para esto sirve esta regla, que busca oraciones simples dentro de una oración compuesta mayor (oración padre) y mediante esto separar en tantas oraciones simples como sea necesario. Es posible que se encuentren guiones “-” entre las oraciones simples de salida, lo cual no significa nada, pero era necesario por la implementación realizada de la gramática.

- Regla separar_args
  Sirve de ayuda para la regla anterior, con el fin de poder iterar el árbol tanto en niveles horizontales (nodos hermanos) como en niveles verticales (nodos hijos).

- Regla quitar_subordinada
  Elimina las oraciones subordinadas de X obteniendo en X1 el resto de la oración.

- Regla buscar_sujeto
  Busca el sujeto de una oración buscando un grupo nominal a la izquierda, en caso de haber este es el sujeto, y luego devuelve true.

- Regla get_sujeto
  Llama a buscar sujeto para que se pueda iterar de forma recursiva.

- Regla poner_sujeto
  Itera cada una de las oraciones, mientras no aparezca un sujeto potencial, añade las oraciones sin sujeto (ya que será oración con sujeto omitido o impersonales). Cuando aparece un sujeto este pasa a ser el sujeto de esa y las siguientes oraciones mientras que no aparezca otro que lo reemplace. (Explicado en mayor profundidad en el apartado de “Aspectos a tener en cuenta”)

- Regla formatear_oraciones
  Formatea las oraciones para que se puedan imprimir de forma correcta (en listas) y ponerles el sujeto.

- Regla get_oraciones
  Obtiene del compound oraciones(...) cada una de las oraciones en listas.

- Regla get_oracion
  Obtiene del compound oracion(de cualquier tipo)(...) la oración en una lista.

- Regla copy_compound
  Sirve para copiar desde el primer compound pasado como argumento hacia el segundo compound pasado como argumento, pero eligiendo desde que posición quieres copiar en el primero y desde que posición quieres pegar en el segundo.

- Regla concatenar_compound
  Mediante la ayuda de la anterior regla concatena dos compounds de entrada devolviéndolo en el tercer argumento de la regla.

traductor.pl

Es el archivo de una de las mejoras de la práctica que permite traducir oraciones de español a inglés y viceversa. Su estructura es la siguiente:

- Consults
  En este caso consult del archivo diccionarioTRA.pl para acceder a todas las palabras de las que se dispone tanto en español como en inglés, así como de su tipo morfológico.

- Reglas gramaticales
  Las mismas que en el archivo analisis.pl, pero con estas tanto para palabras en español, como para palabras en inglés.

- Regla traducir

  Que dado el idioma desde el que se quiere traducir, hacia el que se quiere traducir y una oración, pasa esta de un idioma a otro, mostrándolo por pantalla.

- Reglas o_pruebaESP, o_pruebaENG y ejecutar_pruebasESP y ejecutar_pruebasENG
  Son igual que en el archivo anterior, con la diferencia de que, en vez de mostrar el árbol del análisis sintáctico de cada oración, la traduce del español al inglés y viceversa, mostrándola en ambos idiomas.

EJECUCIÓN

Para ejecutar la práctica hemos creado un archivo (ejecutar.pl) que es el que se debe consultar para poder ejecutar la práctica. Este archivo permite probar las funcionalidades de la práctica (de la parte de sintaxis) mediate un único predicado:

\> ejecutar_pruebas(1, 15)

Este predicado realiza internamente las funciones de analizar sintácticamente la frase (oracion(X, ORACION, [])), dibujarla (draw(X)), separar y formatear las oraciones simples cada una con su sujeto (separar(X, C), formatear_oraciones(C, L)) y las muestra por pantalla. A cada una de estas funciones se le puede llamar de forma manual con la oración que se desee y además si se quieren aplicar todas estas funciones a la vez a una misma oración se puede llamar al predicado (ejecutar_oracion(ORACION)).

Para ejecutar el traductor de oraciones se debe realizar un consult a traductor.pl y así se puede ejecutar (traducir(DESDE, HASTA, ORACION)) que traduce la ORACION del idioma DESDE al idioma HASTA, o se puede llamar a (ejecutar_pruebasESP(1, 15)) y (ejecutar_pruebasENG(1, 15)) que traducen las 14 oraciones de prueba del español al inglés y del inglés al español respectivamente.

ASPECTOS A TENER EN CUENTA

Complejidad de las subordinadas

Las oraciones subordinadas acompañan a un nombre, un verbo y otros tipos de palabras, es decir, tienen dependencia de otros grupos de la oración que contiene a la subordinada y tienen una función de sintagma. Es por ello, que no nos ha sido posible identificar una oración compuesta que tuviera dentro una subordinada en primera instancia. En su lugar, nuestra implementación trata estas oraciones compuestas como oraciones simples y después de haber identificado todos los elementos de la oración, en caso de que dentro haya alguna subordinada, la oración simple se recalifica como oración compuesta.

Selección de sujetos oracionales

Una vez analizadas las oraciones, como ya hemos explicado, se pueden descomponer las oraciones compuestas en oraciones simples, además a cada una de estas oraciones simples se le añade el sujeto que tiene de la siguiente forma.

Se escoge la primera oración, si no tiene, se escribe directamente, ya que significa que o tiene sujeto omitido o es oración impersonal. En caso de que sí tenga sujeto, se escribe tal cual y se selecciona ese sujeto como sujeto potencial para la siguiente oración. A partir de aquí se va iterando con las siguientes oraciones se escribe con el sujeto que tenga y se selecciona como sujeto potencial de la siguiente. Y en caso de que no tenga sujeto se escribe el sujeto potencial de la anterior.

Oraciones coordinadas

Para que nuestro código sea aún más flexible en relación con poder añadir todas las oraciones coordinadas que se desee hemos decidido que las oraciones coordinadas que tengas más de dos oraciones simples se representen como una oración coordinada que contiene una oración simple y una oración compuesta que será coordinada, y así de forma recursiva hasta que se acaben las oraciones.

MEJORAS REALIZADAS

Las mejoras extras realizadas respectos a los elementos obligatorios de la práctica son los siguientes.

Ampliación del diccionario

Se ha realizado una ampliación al diccionario con más de 500 palabras de diferentes tipos morfológicos, sacadas de la página web incluida en la bibliografía. Nos hemos ayudado de un pequeño script Python que hemos realizado para poder clasificar manualmente estas palabras. El script no se adjunta en la entrega de la práctica, pero en caso de ser solicitado por el profesor, se le puede enviar sin problemas.

Flexibilización de la gramática

Realizando la práctica nos hemos dado cuenta de que para cumplir con la sintaxis de varias de las oraciones propuestas era necesario incluir predicados muy específicos y estrictos con las oraciones de entrada (es decir, que no servirían para otras oraciones), es por ello por lo que decidimos crear un nuevo predicado que nos permita tener más flexibilidad pudiendo añadir sintagmas de forma recursiva.

El predicado creado es complementos/3. La idea de este predicado es incluir todos los sintagmas de los que se pueda acompañar tanto un verbo como un sintagma nominal y que estos se puedan añadir de forma recursiva e ilimitada.

Nos hemos encontrado con un gran problema al realizar esta implementación, y es que a la hora de imprimir el árbol sintáctico nos imprimía los complementos como un elemento sintáctico más, añadiendo ramas que poco sentido tenían. Por tanto, hemos tenido que procesar el compound del análisis sintáctico después de haberlo realizado.

El post-procesamiento consiste en aplanar el árbol cada vez que apareciera un compound de tipo comp(…) dejando todas esas ramas al mismo nivel.

Sin embargo, este esfuerzo ha merecido la pena, ya que nuestro programa es capaz de analizar oraciones con una cantidad de grupos sintácticos muy elevada, que sería imposible de analizar si tuviéramos que hacer un predicado por cada uno de esos casos.

Traductor

Se ha realizado la implementación de un traductor entre español e inglés, aunque no habría que hacer mucha modificación para que este fuera entre dos idiomas cualesquiera, además de añadir diccionarios entre ambos dos lenguajes.

Nuestra implementación está capacitada para traducir una oración cualquiera entre estos idiomas siempre y cuanto las palabras que componen la oración aparezcan en el diccionario adjunto. Además, la ejecución del programa permite la traducción de todas las oraciones de prueba que aparecían en el enunciado de la práctica tanto de español a inglés como de inglés a español, haciendo todas de golpe y mostrando su salida por pantalla.

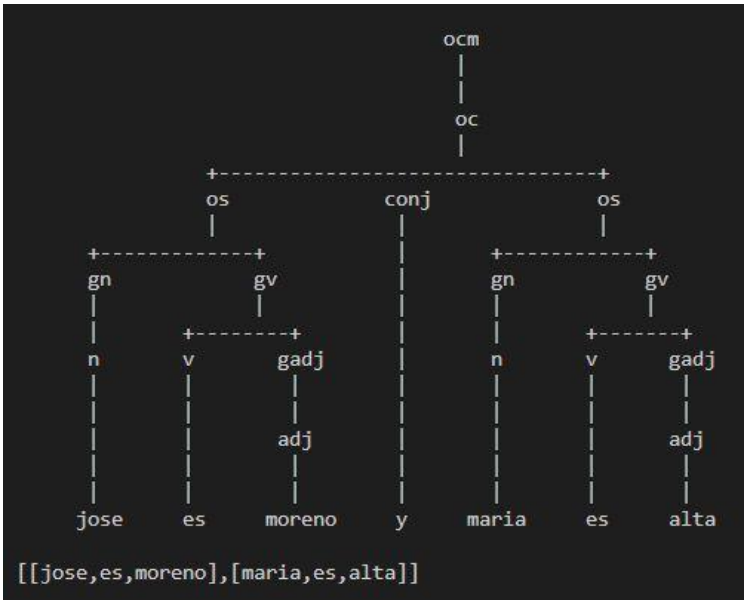
Flexión de palabras

Se ha realizado un script en Python que permite la flexión de las distintas palabras que se le pasen, sacando de esto la palabra tanto en masculino y femenino, como en plural y singular, esto lo hemos aplicado a parte del set de palabras del que ya disponíamos previamente, haciendo crecer este aún más.

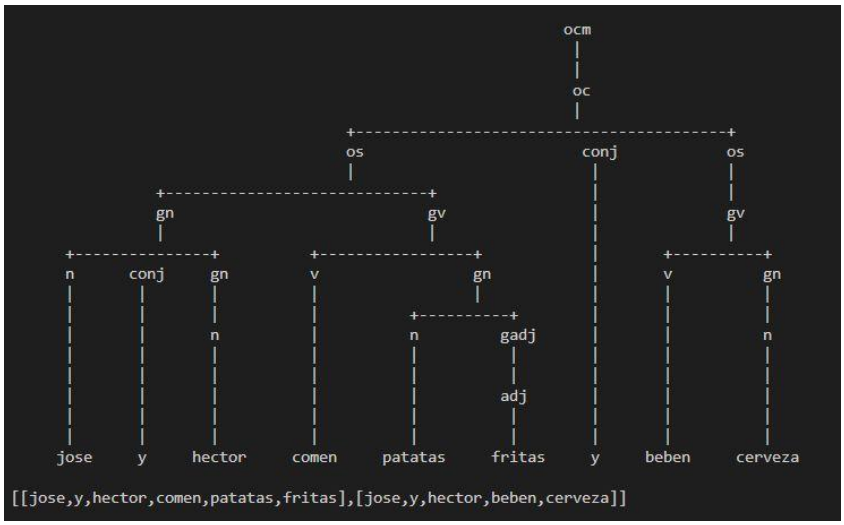
Este script de Python al igual que el de la ampliación del diccionario no se han adjuntado en la entrega, ya que no era el lenguaje de programación evaluado en la asignatura, pero este queda a disposición del profesor en caso de que lo solicite.

EJEMPLOS DE LA EJECUCIÓN

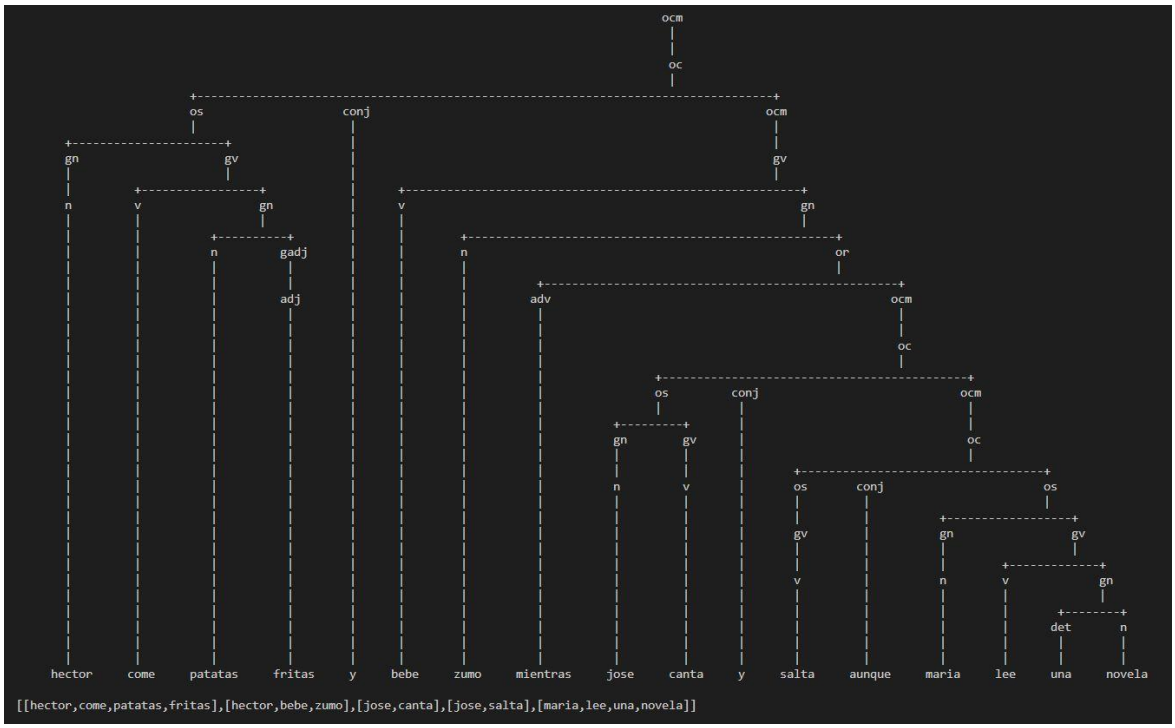
Analizador sintáctico



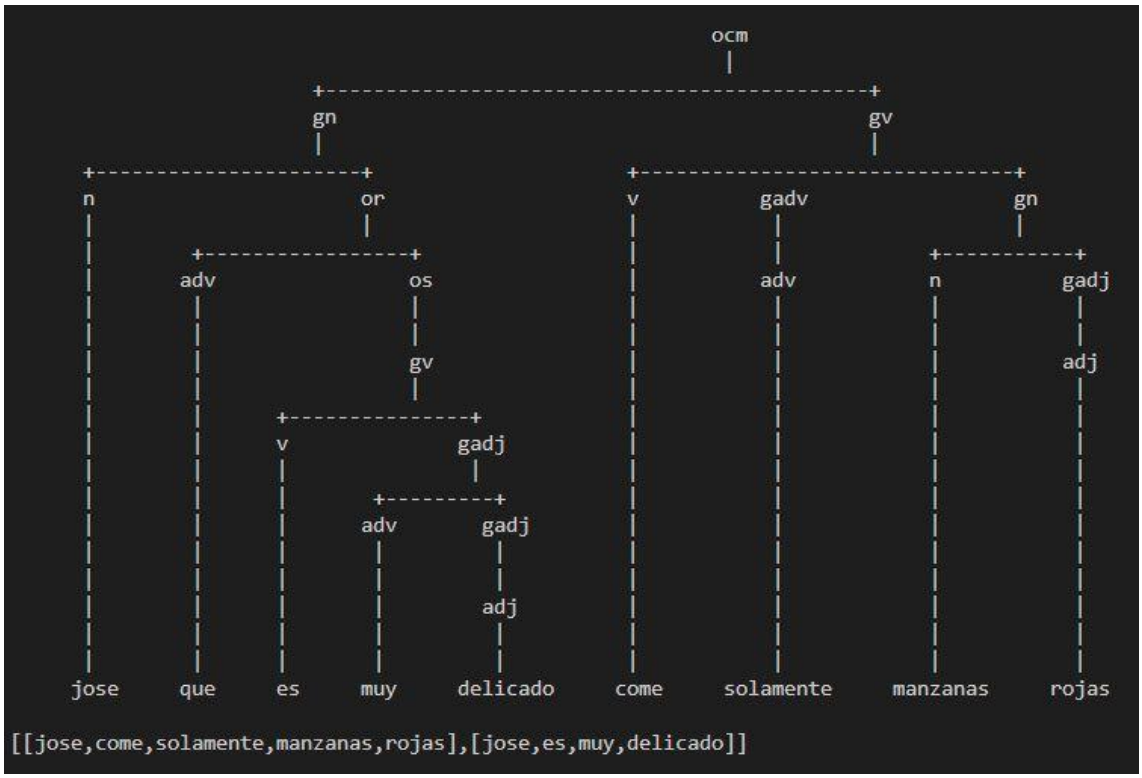
Ejemplo 1: En esta oración podemos ver una oración coordinada de dos oraciones simples, como podemos observar esta oración es catalogada como oración compuesta y la conjunción no forma parte de ninguna de las dos oraciones.



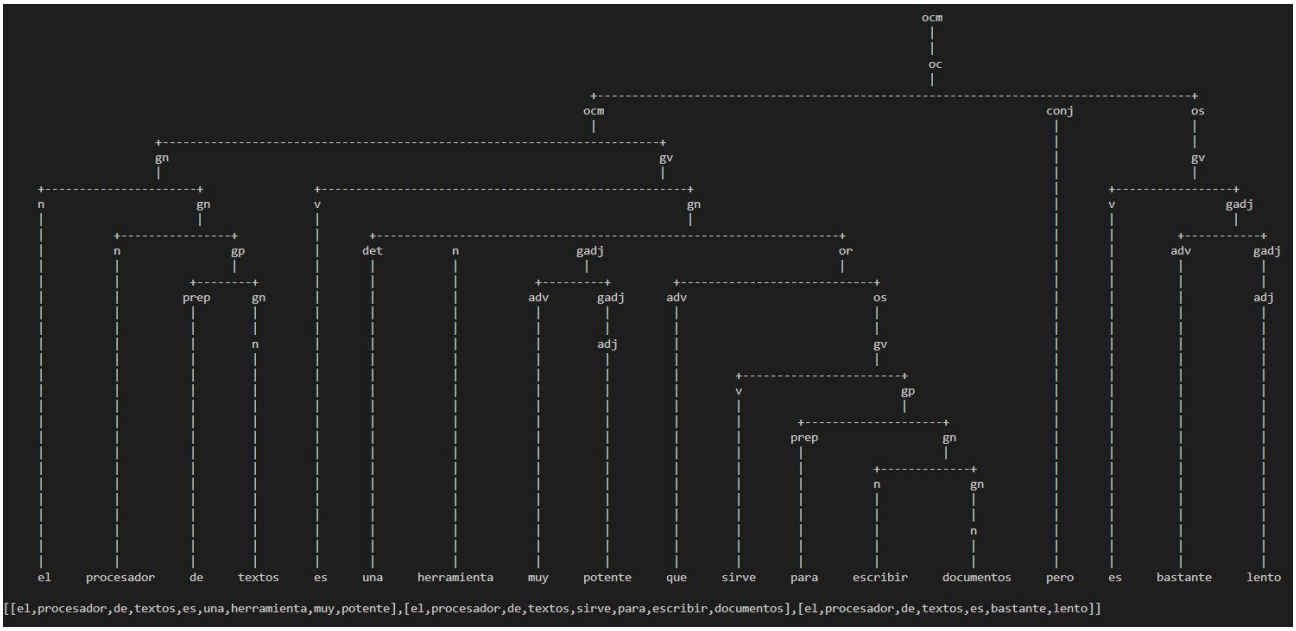
Ejemplo 5: La finalidad de la selección de esta oración es el uso de las conjunciones en sintagmas nominales, y la muestra de como funciona la asignación de sujetos, como podemos observar “José” y “Héctor” son el sujeto también de la segunda oración.



Ejemplo 8: Esta oración es bastante más compleja que las anteriores, consta de varias oraciones coordinadas y de una subordinada. Como podemos comprobar, la oración que contiene a la subordinada está catalogada también como compuesta. Aquí también se puede observar el funcionamiento de la selección del sujeto, siendo “José” tanto el sujeto de canta como de salta.

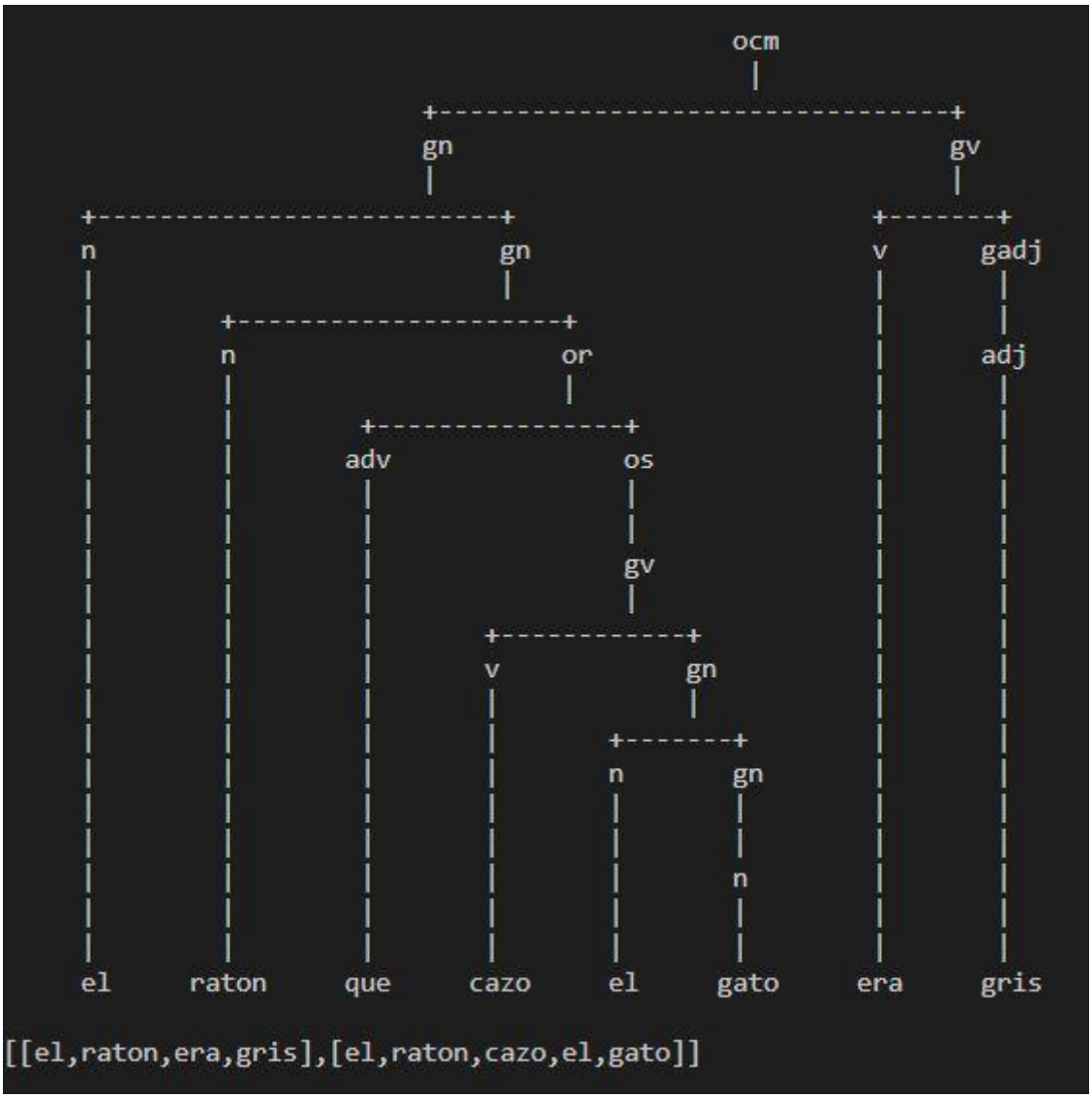


Ejemplo 10: en este ejemplo aparece una oración subordinada lo que nos permite mostrar cómo se han considerado las oraciones que componen una oración compuesta subordinada. Como podemos observar, la primera oración (la principal) es la oración completa, pero con la rama de la oración subordinada podada, y la segunda oración es la oración subordinada con el sujeto de la principal.



Ejemplo 12: Esta oración se ha seleccionado por su complejidad y por la mezcla de tipos de oraciones, además se observa la labor realizada con el tema de los complementos en el grupo

nominal “una herramienta muy potente que sirve para escribir documentos”, el núcleo “herramienta” está acompañado por un grupo adjetival y por una oración subordinada. Si no hubiéramos realizado el predicado de los complementos, tendríamos que haber realizado un predicado específico para este caso, que es poco común.



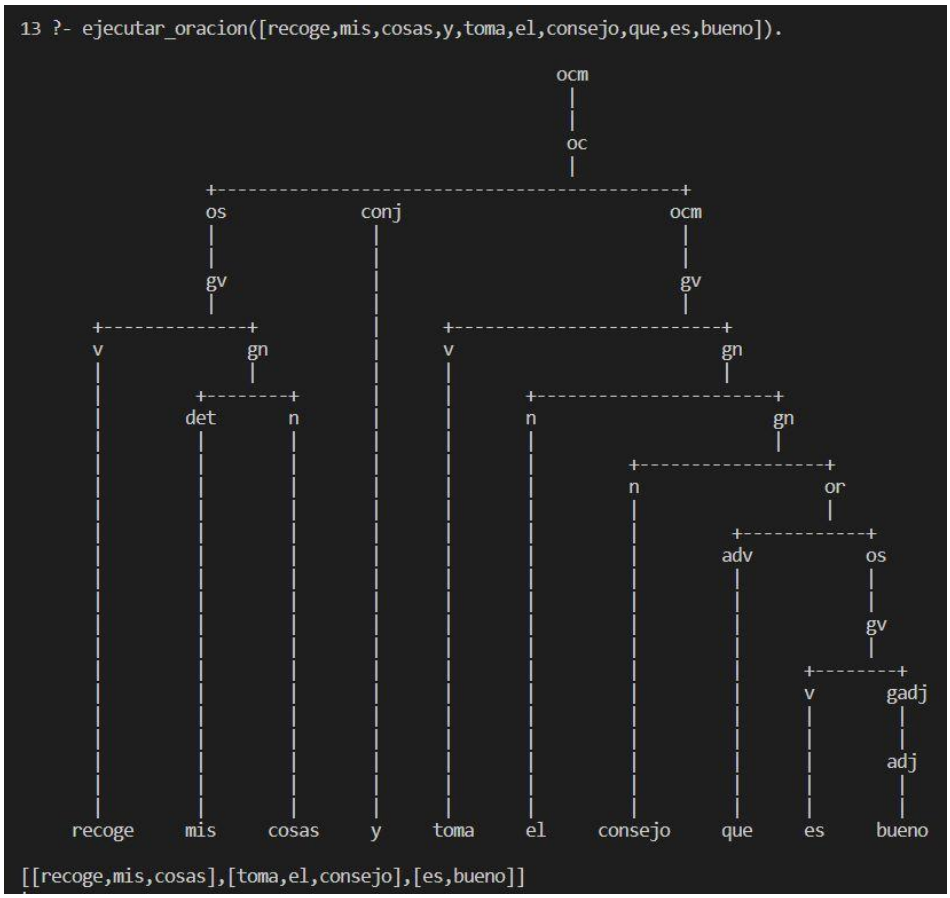
Ejemplo 13: Esta oración está en voz pasiva, lo que significa que el sujeto no realiza la acción, sino que la acción se realiza sobre él. Esto significa que la selección del sujeto es complicada, ya que habría que pasarla a voz activa para determinarlo correctamente. De hecho, la segunda oración tiene el sujeto y el complemento directo intercambiados ya que es el gato quien caza al ratón.

Traductor

```
2 ?- ejecutar_pruebasESP(1, 15).
Oracion en esp: [jose,es,moreno,y,maria,es,alta]
Oracion en eng: [joseph,is,brown,and,mary,is,tall]

Oracion en esp: [jose,estudia,filosofia,pero,maria,estudia,derecho]
Oracion en eng: [joseph,studies,philosophy,but,mary,studies,law]

Oracion en esp: [maria,toma,un,cafe,mientras,jose,recoge,la,mesa]
Oracion en eng: [mary,takes,a,coffee,while,joseph,collect,the,table]

Oracion en esp: [jose,toma,cafe,y,lee,el,periodico]
Oracion en eng: [joseph,takes,coffee,and,reeds,the,newspaper]

Oracion en esp: [jose,y,hector,comen,patatas,fritas,y,beben,cerveza]
Oracion en eng: [joseph,and,hector,eat,potatoes,fried,and,drink,beer]

Oracion en esp: [jose,come,patatas,fritas,pero,maria,prefiere,paella,aunque,hector,toma,cafe,e,irene,lee,una,novela]
Oracion en eng: [joseph,eats,potatoes,fried,but,mary,prefers,paella,although,hector,takes,coffee,and,irene,reeds,a,novel]

Oracion en esp: [irene,canta,y,salta,mientras,jose,estudia]
Oracion en eng: [irene,sings,and,jumps,while,joseph,studies]

Oracion en esp: [hector,come,patatas,fritas,y,bebe,zumo,mientras,jose,canta,y,salta,aunque,maria,lee,una,novela]
Oracion en eng: [hector,eats,potatoes,fried,and,drinks,juice,while,joseph,sings,and,jumps,although,mary,reeds,a,novel]

Oracion en esp: [jose,que,es,agil,escala,en,el,rocodromo,por,las,tardes]
Oracion en eng: [joseph,that,is,agile,climbs,in,the,climbing-wall,by,the,afternoon]

Oracion en esp: [jose,que,es,muy,delicado,come,solamente,manzanas,rojas]
Oracion en eng: [joseph,that,is,very,delicate,eats,only,apples,red]

Oracion en esp: [el,procesador,de,textos,que,es,una,herramienta,bastante,potente,sirve,para,escribir,documentos]
Oracion en eng: [the,processor,of,texts,that,is,a,tool,quite,powerful,serves,to,writing,documents]

Oracion en esp: [el,procesador,de,textos,es,una,herramienta,muy,potente,que,sirve,para,escribir,documentos,pero,es,bastante,lento]
Oracion en eng: [the,processor,of,texts,is,a,tool,very,powerful,that,serves,to,writing,documents,but,is,quite,slow]

Oracion en esp: [el,raton,que,cazo,el,gato,era,gris]
Oracion en eng: [the,mouse,that,hunted,the,cat,was,gray]

Oracion en esp: [el,hombre,que,vimos,ayer,era,mi,vecino]
Oracion en eng: [the,man,that,saw,yesterday,was,my,neighbor]
```

Así queda la ejecución de la traducción de todas las oraciones del enunciado de ingles a español, mostrando la oración de entrada y la de salida tras traducir. Si el comando usado hubiese sido \> ejecutar_pruebasENG(1, 15). Estas hubieran salido traducidas de inglés a español.

Nuestro ejemplo de oración



Aquí podemos observar que el programa admite nuevas palabras que no estaban inicialmente y como funciona una oración sin sujeto.

Nuestro ejemplo de traducción

```
3 ?- traducir(esp,eng,[recoge,mis,cosas,y,toma,el,consejo,que,es,bueno]).
Oracion en esp: [recoge,mis,cosas,y,toma,el,consejo,que,es,bueno]
Oracion en eng: [collect,my,things,and,takes,the,advice,that,is,good]
```

Aquí podemos observar que el traductor admite nuevas palabras, siempre que estas aparezcan en el diccionario. Se puede observas que “take” no lo conjuga bien, ya que no podemos distinguir entre el número de personas que se está hablando en la oración.

ERRORES Y ASPECTOS NO IMPLEMENTADOS

Se informa que todos los aspectos solicitados en la práctica han sido implementados adecuadamente y sin errores. Se ha prestado atención a los detalles y se ha garantizado que se cumplan todas las especificaciones establecidas. No se ha encontrado ningún error o aspecto solicitado que no haya sido implementado correctamente, lo que demuestra un alto nivel de calidad en el trabajo realizado. En conclusión, se puede afirmar que la práctica ha sido completada con éxito en todos sus aspectos.

BIBLIOGRAFÍA

Apuntes de teoría y laboratorio disponibles en Blackboard.

Página web del diccionario: https://espanolaldia.wordpress.com/2016/07/08/las-500-palabras-mas-usadas-en-espanol/