1. Recapitulación

Antes de emepazar con gramáticas de dependencias, recordemos algunos puntos sobre las gramáticas independientes de contexto que nos conviene tener en a mano.

- Las CFG están constituidas por:
  - Un vocabulario de símbolos terminales.
  - Un vocabulario de símbolos no terminales.
  - Un elemento raíz que se correspondía con la categoría gramatical más compleja (O).
  - Un conjunto de reglas de derivación que indican cómo un símbolo del vocabulario no terminal se reescribe como una serie de no terminales o un símbolo no terminal.
- Las CFG pueden representarse por medio de grafos acíclicos dirigidos (árboles), cuyos nodos están regidos por axiomas de Dominancia y Precedencia.
- La noción de *dominancia exhaustiva* define la relación entre nodos que forman un constituyente. El tipo de constituyente formado es el del nodo que domina la estructura, categorizado por un símbolo no terminal.
- Las gramáticas independientes de contexto pertenecen a la teoría de constituyentes inmediatos (IC) y a las gramáticas de estructura de frase (PSG).

2. Gramáticas de dependencias

La Gramática de Dependencias es un marco teórico desarrollado principalmente por Tesnière (1959), libro aparecido cinco años después de la muerte de su autor; y ya se encuentra un adelanto de este marco en Tesnière (1934). Desde un punto de vista formal, las características más importantes de su planteo son las siguientes:

- Las palabras establecen distintos tipos de relaciones entre sí. Entre estos, las llamadas relaciones de "dependencia" entre un regidor y un dependiente son las principales.

- La estructura de las oraciones es representada mediante grafos acíclicos dirigidos denominados *stemmas*. Visualmente son semejantes a las gramáticas independientes de contexto con la diferencia de que no poseen nodos no terminales.

- No existe en la teoría una unidad de nivel superior equivalente a lo que se conoce en las gramáticas independientes de contexto como "constituyentes".

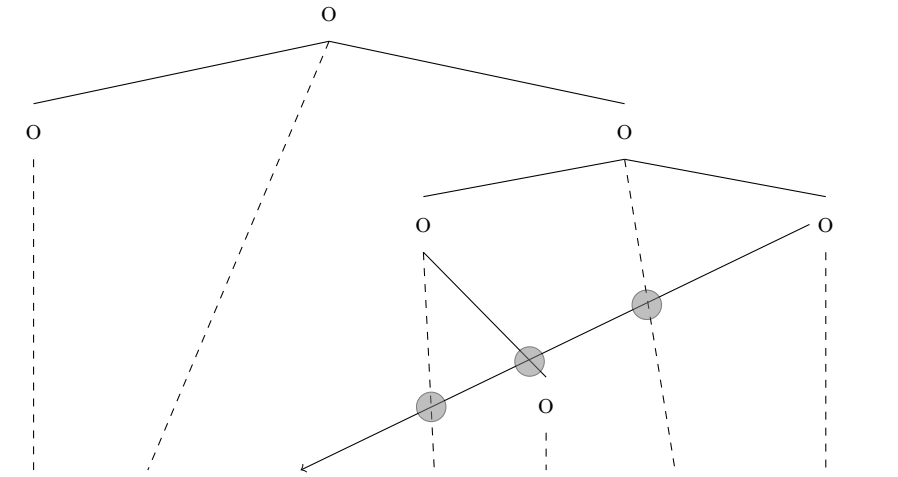
2.1. Motivación de las gramáticas de dependencias

Hasta esta clase, asumimos un acercamiento a las representaciones sintácticas en el que la unidad primitiva de la gramática es el constituyente y en el que las relaciones semánticas derivan de la estructura jerárquica de los constituyentes.

Sin embargo, como indica (Carnie, 2010, Capítulo 9), este no es el único acercamiento posible a una teoría que dé cuenta de la generación de infinitas oraciones dado un conjunto finito de elementos. Existen otras propuestas que describen y otorgan estructura a las oraciones por fuera de las teoría de constituyentes, y una de ellas es la gramática de dependencias.

Al prescindir de la noción de constituyente, las GD no requieren que las palabras presenten un agrupamiento contiguo, lo que las vuelve preferibles a la hora de explicar dependencias no locales. Recordaremos que, para las gramáticas de constituyetes, un núcleo y un elemento regido debían pertenecer al mismo sintagma y guardar una relación de contigüidad. Para poder dar cuenta de los casos en los que un elemento ajeno al consituyente interrumpía dicha relación, era necesario apelar a postulados adicionales, tales como el movimiento o las categorías nulas. Al no utilizar la noción de consituyente, las gramáticas de dependencias se liberan también de la necesidad de recurrir a tales postulados. Esto hizo que tomaran un especial impulso en la descripción de las lenguas de orden libre. Mel'čuk (2011) exhibe esta problemática con un fragmento de un poema de Catulo, en latín.

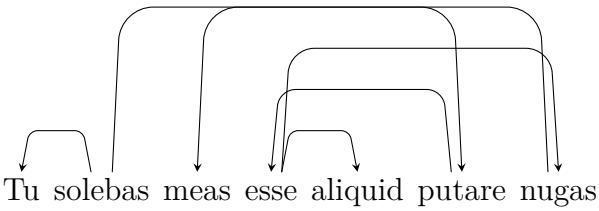
(1)



Tu	*solebas*	*meas*	*esse*	*aliquid*	*putare*	*nugas*
tú	soler	mi	ser	algo	pensar	bagatela
nom	2sg	fem.pl.acc	inf	nom	inf	fem.pl.acc

"Tú solías pensar que mis bagatelas eran algo."

(2)



Mel'čuk (2011) relata su primer acercamiento a las GD motivado por la necesidad de un cambio de paradigma en la traducción automática entre lenguas con diferente orden de palabra, así como diferentes dependencias morfológicas de caso, genéro o número.

Al mismo tiempo, como explica Bauer (1979), habría incluso razones para preferir las DG (*Dependency Grammars*) a las gramáticas de estructura de frase como base de una gramática transformacional. Algunas de las razones presentadas en Bauer son que las DG:

- Permiten una representación directa de la relación entre núcleo y modificadores.

- Ofrecen una descripción más clara de las frases con caso.

- Son débilmente equivalentes a las teorías de constituyentes, lo que significa que ambas gramáticas pueden dar cuenta de las mismas cadenas de texto. Adicionalmente, ambas teorías (la de constituyentes y la de dependencias) son débilmente equipotentes, dado que todas las gramáticas de ambas teorías presentan dicha equivalecia débil (Hays, 1964).

2.2. La noción de dependencia

Ver ejercicio en 3.1

La intuición detrás de la noción de dependencia es que, en la oración, cada palabra dependen de otra, si exceptuamos a la raíz, que no depende de ninguna.

La dependencia es una relación binaria R que se establece entre los elementos de una oración (W):

- $\mathrm{R} \subset \mathrm{W} \times \mathrm{W}$

Esta relación R está regida por una serie de axiomas:

- R es acíclica:

  $\forall w_1, w_2, ..., w_{k-1}, w_k \in W : \langle w_1, w_2 \rangle \in R, ..., \langle w_{k-1}, w_k \rangle \in R : w_1 \neq w_k$

- Existe un único elemento raíz que no pertenece a R:

  $\exists! w_1 \in W : \forall w_2 \in W : \langle w_1, w_2 \rangle \notin R$

- Todos los elementos de R tienen un único regidor:

  $\forall w_1, w_2, w_3 \in W : \langle w_1, w_2 \rangle \in R \wedge \langle w_1, w_3 \rangle \in R \rightarrow w_2 = w_3$

De los axiomas anteriores se sigue que:

- R es anti-reflexiva:

  $\forall \mathrm{w}_1 \in \mathrm{W}$: $\langle \mathrm{w}_1, \mathrm{w}_1 \rangle \notin \mathrm{R}$.

- R es anti-simétrica:

  $\forall \mathrm{w}_1 \mathrm{w}_2 \in \mathrm{W}$: $\langle \mathrm{w}_1, \mathrm{w}_2 \rangle \in \mathrm{R} \rightarrow \langle \mathrm{w}_2, \mathrm{w}_1 \rangle \notin \mathrm{R}$.

- R no es transitiva:

  $\forall w_1 w_2 w_3 \in W$: $\langle w_1, w_2 \rangle \in R \wedge \langle w_2, w_3 \rangle \in R \rightarrow \langle w_1, w_3 \rangle \notin R$.

La relación de dependencia puede ser representada por una flecha: Ver ejercicio en 3.2

$$\text{Regidor} \rightarrow \text{Dependiente}$$

Si bien bajo esta representación la relación de dependencia puede resultar parecida a la noción de derivación de CFG, debemos notar las diferencias. En CFG, entre los símbolos a izquierda y derecha de la flecha de derivación teníamos una relación de dominancia entre los nodos. Y, a diferencia de la relación de dependencia entre nodos, la dominancia era reflexiva y transitiva.

Ahora, desde la intuición, ¿cuál es la raíz en estas oraciones y qué palabra depende de cuál?

Alexis mira Alejo y Valentina. 

Alexa escucha música y podcasts. 

Saoko, papi, saoko. 

2.2.1. Tipos de dependencias

En pleno auge de las gramáticas transformacionales, Hays (1964) señala la posibilidad de extender las gramáticas de dependencias en

la dirección de una gramática transformacional o de una gramática de estratos en la que cada estrato posee su propio alfabeto. En ese sentido, plantea la existencia de un alfabeto terminal, con elementos terminales (*i.e.* cadenas), y de un alfabeto auxiliar, con nombres o símbolos que caractericen las ocurrencias de Las cadenas terminales. Estas, observa Hays, son sintácticamente ambiguas, en tanto pueden desempeñar distintas funciones sintácticas (un nombre, por ejemplo, puede ser tanto sujeto como objeto en una oración). Los símbolos auxiliares, en cambio, son precisos, no contienen ambigüedad. Entre ambos tipos de elementos (terminales y auxiliares) se da una relación muchos-a-muchos (*many-many*), en tanto un terminal puede ser caracterizado por más de un auxiliar al cumplir distintas funciones, y un auxiliar puede corresponderse con una lista de terminales, todos a los que pueda caracterizar en al menos una oración.

Mel'čuk (2011) propone una extensión de la gramática de dependencias por estratos en la que cada estrato posee un tipo de dependencia. Todas las combinaciones posibles entre estos tipos de dependencia describirían casi el total de las dependencias linguisticas.

- **Dependencias semánticas**: Las dependencias semánticas serían las encargadas de establecer relaciones entre las unidades mínimas del estrato semántico (semantemas) y corresponderían a una relación de predicado–argumento, donde el regidor es el predicado.

- **Dependencias sintácticas**: Las dependencias sintácticas establecen las relaciones entre la red representada por el estrato semántico y la cadena de morfemas del estrato morfológico en la forma de un árbol de dependencias.

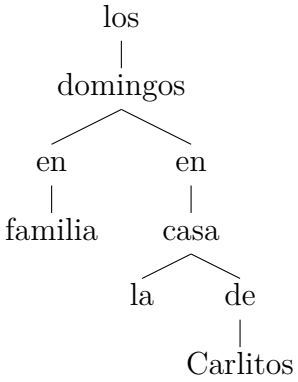
- **Dependencias morfológicas**: Las dependencias morfológicas dependen del idioma y no es necesario que unan a todos los elementos del estrato morfológico de una oración, pero son usadas por las reglas sintácticas y se corresponden con las relaciones de *Government* y *Agreement*.

2.3. El Stemma

Tesnière (1934)[1] desarrolla la idea de *stemma* para representar de manera gráfica las conexiones que vinculan a las palabras en la oración. El *stemma* en Tesnière toma la forma de un árbol en el que el regidor domina a sus dependientes y en el que se pierde el orden lineal de la oración.

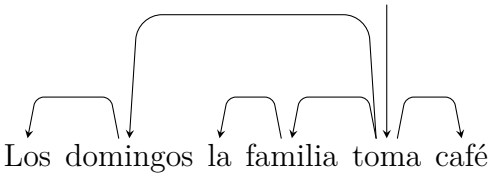
Ver ejercicio en 3.3

(3)



Hays (1964) y Gaifman (1965) proponen el árbol de dependencias, que mantiene el orden lineal y la estructura del *stemma* y usa categrías en vez de palabras. Las palabras aparecen relacionadas con su categoría por medio de lineas punteadas.

Por último, Hudson (1984) propone una notación diferente, donde se mantiene el orden lineal de palabras y el uso de flechas indica la relación entre el regidor y su dependiente:



Esta última forma de representación, alternativamente con la categoría como etiqueta de la flecha, es la que más veremos representada en las implementaciones computacionales, sea en las visualizaciones como en los *papers*.

Ver ejercicio en 3.4

[1] Tomado aquí a partir de Carranza (2016).

2.4. Axiomas de las gramáticas de dependencias

Robinson (1970) formuló cuatro axiomas que rigen la formación de las estructuras de dependencias.

- **Raíz única**: Como vimos, la relación de dependencia es binaria, anti reflexiva, anti simétrica y no transitiva, lo que conlleva que un elemento de la oración no depende de nadie. Ese elemento es la raíz (*root*) y solo un elemento en la oración puede ser independiente. La raíz suele estar indicada en las representaciones planas, como la de Hudson, por una flecha que no tiene nodo de procedencia.

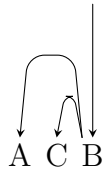
- **Conectividad**: Todos los elementos de la oración dependen directamente de otro elemento. No habrá elementos intermedios en las estructuras de dependencias, por lo que únicamente habrá tantos nodos como palabras en el grafo correspondiente.

- **No multidominancia**: Ningún elemento depende de más de un elemento. Un regidor podrá regir varios dependientes, pero cada dependiente puede tener únicamente un regidor.

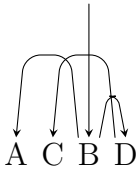
- **Proyectividad**: Si un elemento A depende directamente de un elemento B y un elemento C interviene en el orden lineal entre ellos, ese elemento C depende directamente de A o de B. La proyectividad expresa una condición similar a la *non-tangling condition* de las CFG. Sin embargo, Tesnière no imponía esta condición, como lo veremos a continuación.

2.5. No proyectividad

Tesnière reconoce la posibilidad de que un dependiente no aparezca inmediatamente antes ni inmediatamente después de su regidor. A esta linearización la llama *orden brisé* y corresponde a los casos en que un dependiente A está separado de su regidor B por otro dependiente C al que también rige B:



Pero este orden también se corresponde con los casos en que A depende de B pero un elemento C, que no depende de B ni de A, sino de D, se encuentra linearmente entre A y B:



Este último ejemplo es el que rompe el axioma de proyectividad. Las gramáticas de dependencias que dan cuenta de fenómenos de esta forma son gramáticas no proyectivas.

Si la gramática no proyectiva es irrestricta, resulta en una gramática intratable computacionalmente. Para evitar esto, habilitando al mismo tiempo la riqueza expresiva de la no proyectividad, algunas implementaciones incluyen restricciones a la no proyectividad, logrando aplicaciones levemente no proyectivas capaces de parsear una oración en tiempo polinómico.

2.6. Parsers

Los parsers para estas gramáticas fueron ganando espacio en la industria en tres acercamientos posibles: “graph based” (basado en grafos), “transition based” (basado en transiciones) o una mezcla de los dos algoritmos.

Los dos algoritmmos tienen en común el usar un modelo supervisado de aprendizaje automático para la predicción de cierta información. Para los basados en transiciones, el modelo indicará la próxima transición. Para los basados en grafos, los pesos usados por los arcos del grafos son aprendidos previamente por el modelo.

- **"transition based"** es un tipo de parser de dependencia que, dada una cadena, busca predecir la serie de transiciones necesarias para concluir en un árbol exitoso. El resultado de cada transición es la relación entre dos palabras de la cadena, o entre una de ellas y un símbolo "root" agregado por el algoritmo (transición que asigna la raíz). Un tipo de parser muy usado con este algoritmo es el "shift reduce". Como sabemos, este parser consta de:

  - un pila (*stack*) donde podrá almacenar elementos de la cadena a la espera de una order;

  - una orden para agregar otro elemento de la cadena a la pila (*shift*);

  - una orden para unir (*reduce*) elementos ya presentes en la pila. Para este tipo de parser, es habitual que los tipos de orden posibles para la unión estén basados en el *arc-standard algorithm.* Este algoritmo solo puede predecir parsers proyectivos y consta de dos transiciones posibles:

    - *left-arc* crea un dependencia entre la última palabra agregada al stack y la siguiente en la pila (la de abajo) y saca esta última de la pila, es decir, que en el árbol resultante se creará un arco a la izquierda.

    - *right-arc* crea un dependencia entre la palabra de abajo del stack y la última agregada y saca esta última de la pila, es decir, que en el árbol resultante se creará un arco a la derecha.

  El parser se inicia con una configuración que consta de un árbol vacío y un input, llama a un clasificador que predice la próxima transición que tomará el parser (*shift*, *left-arc* o *right-arc*) y aplica este proceso hasta obtener un stack vacío y un árbol completo.

  El tiempo de procesamiento de una cadena de n elementos mediante este procedimiento es de $2n$-1.

- **"graph based"** son los parsers donde el árbol resultante es el grafo que maximiza una puntuación sobre todos los posibles grafos para una cadena. Una de las puntuaciones más usuales en este tipo de parser es *edge-factored*, es decir, basado en el puntaje de los arcos entre palabras y alcanzando la mayor

puntuación aquel árbol que une los arcos con mayor puntaje. Para eso es necesario:

- un modelo para asignar puntajes a los arcos;

- un algoritmo de búsqueda del camino con mayor puntuación.

Uno de los algoritmos de búsqueda más usados en estos parsers es el *Maximum Spanning Tree*. Este algoritmo construye un grafo dirigido *edge-factored* donde los vertices son las palabras de la cadena input y los arcos entre ellas representan posibles dependencias. Cada arco está pesado según su mayor o menor probabilidad de representar una relación regidor-dependiente válida.

Los arcos de estos grafos son dirigidos, por lo que el modelo que asigna puntajes puede asignarle un puntaje a la relación $\langle w_1, w_2 \rangle$ y otro puntaje a $\langle w_2, w_1 \rangle$.

Este tipo de parser es preferible al *transition based* para dependencias más largas, ya que las transiciones suelen perder precisión si los elementos a unir están alejados simplemente porque la oración es más larga. Al proponer un acercamiento donde el árbol completo es puntuado, los parsers *graph based* evitan este efecto *greedy* de resolver dependencias de corta distancia.

2.7. Implementaciones

- **spaCy** es una librería *open source* de Python para procesamiento de lenguaje natural (como NLTK). Entre las herramientas que provee la librería (NER, PoS tag, etc.) hay un parser de dependencias basado en transiciones pero que permite una transformación pseudo proyectiva basada en Nivre y Nilsson (2005) para predecir árboles no proyectivos. Pasemos a ver el parser de spaCy en acción en la notebook de esta clase.

3. Ejercitación

3.1. Funciones en la oración

Viene de la página 4

Indicá qué categoría gramatical y qué funciones sintácticas pueden tener los siguientes *ítems* léxicos.

1. Estudiar
   Categoría: Funciones:
2. Mesa
   Categoría: Funciones:
3. Contra
   Categoría: Funciones:
4. Caminando
   Categoría: Funciones:
5. Limpio
   Categoría: Funciones:
6. Cultural
   Categoría: Funciones:

3.2. Relación núcleo–elemento regido

Viene de la página 5

¿Qué dependencias podrían tener los siguientes *ítems* léxicos? Indicá su categoría gramatical y proponé un ejemplo concreto.

1. Ver
   Categoría: Dependencias:
2. Con
   Categoría: Dependencias:
3. Casa
   Categoría: Dependencias:
4. Preguntar
   Categoría: Dependencias:
5. Dudar
   Categoría: Dependencias:

3.3. Árbol de dependencias

Armá el árbol de dependencias para las siguientes oraciones:

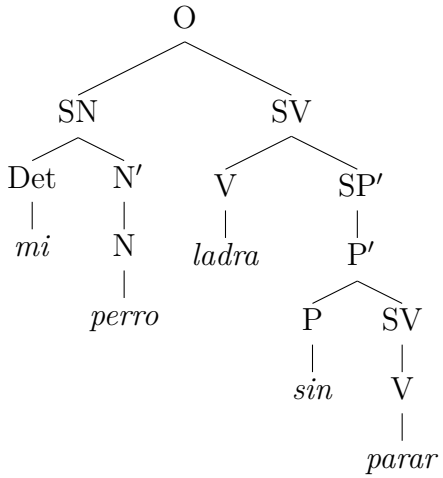
Viene de la página 7

1. Fernando miró al perro con rabia.
2. Macarena saludaba a la vecina con la escoba.
3. Pablo estornudó fuertemente.

3.4. De CFG a DG

Transformá el siguiente árbol en uno de dependencias.

Viene de la página 7



Referencias

Bauer, L. (1979). Some thoughts on dependency grammar.

Carnie, A. (2010). *Constituent structure*. Oxford University Press, Oxford.

Carranza, F. (2016). Tesnière y su gramática de dependencias: continuidades y discontinuidades. *RAHL: Revista argentina de historiografía lingüística*, 8(2):59–78.

Gaifman, H. (1965). Dependency systems and phrase structure systems. *Information and Control*, (8):304–337.

Hays, D. G. (1964). Dependency theory: A formalism and some observations. *Language*, 40(4):511–525.

Hudson, R. A. (1984). *Word grammar*. Blackwell Oxford.

Mel'čuk, I. (2011). Dependency in language-2011. En *Proceedings of International Conference on Dependency Linguistics*, pp. 1–16. Citeseer.

Nivre, J. y Nilsson, J. (2005). Pseudo-projective dependency parsing. En *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pp. 99–106, Ann Arbor, Michigan. Association for Computational Linguistics.

Robinson, J. J. (1970). Dependency structures and transformational rules. *Language*, pp. 259–285.

Tesnière, L. (1934). Comment construire une syntaxe. *Bulletin de la Faculté des lettres de Strasbourg*, 12(7).

Tesnière, L. (1959). *Eléments de syntaxe structurale*. Klincksieck, Paris.