

# 1. Introducción

En la clase previa hicimos una presentación preliminar de las *Minimalist Grammars* siguiendo la formalización de [Stabler \(2011\)](#). Revisamos brevemente una serie de definiciones formales y nos centramos, en particular, en la función *merge*, que tiene dos variantes: (i) *external merge* [**em**] es la función responsable de la creación de la estructura sintáctica; e (ii) *internal merge* [**im**] es la función responsable del movimiento de constituyentes. Ambos tipos de *merge* son desencadenados por dos clases de rasgos diferentes: rasgos de selección categorial [=X] en **em** y rasgos de licenciamiento [ $\pm$ x] en **im**. Vimos, asimismo, la implementación en Prolog del parser minimalista `mgpx` y el parser de secuenciación léxica `lpx`, en tres gramáticas básicas.

En esta clase, veremos con mayor profundidad el funcionamiento de estas gramáticas y revisaremos algunas restricciones y soluciones en relación con el ordenamiento lineal de los constituyentes. Luego, revisaremos de gramáticas más complejas, que involucran el movimiento de núcleos.

## 2. Algunos supuestos básicos

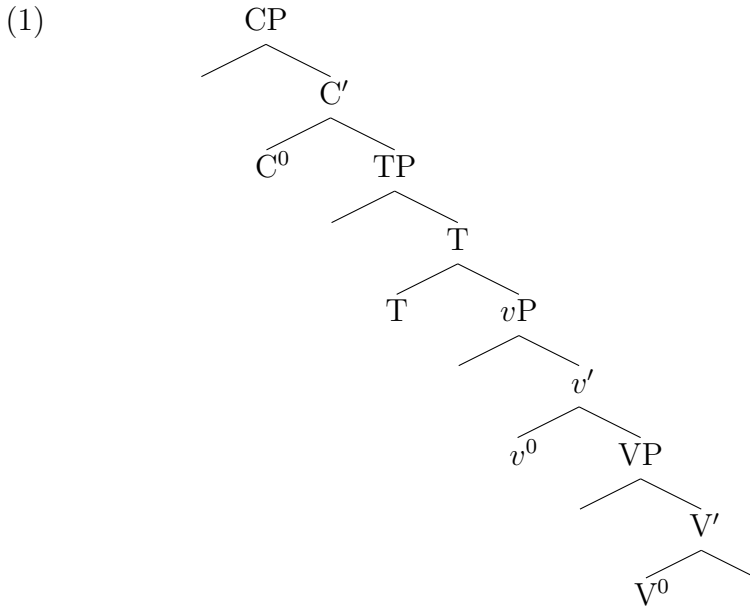
En estas clases adoptamos, en general, el supuesto estándar en el marco generativo actual según el cual el esqueleto funcional básico de la cláusula incluye la proyección de tres núcleos:

$v^0$  es el núcleo responsable de la transitividad del verbo, introduce el argumento externo (AE) y determina la asignación de caso acusativo.

$T^0$  es el núcleo asociado a la flexión verbal, que determina el caso nominativo.

$C^0$  es el núcleo responsable de la complementación. Determina la finitud de la oración y aloja diferentes constituyentes movidos asociados con la estructura de la oración o la modalidad.

La estructura básica de la cláusula aparece respresentada en forma de árbol en 1.



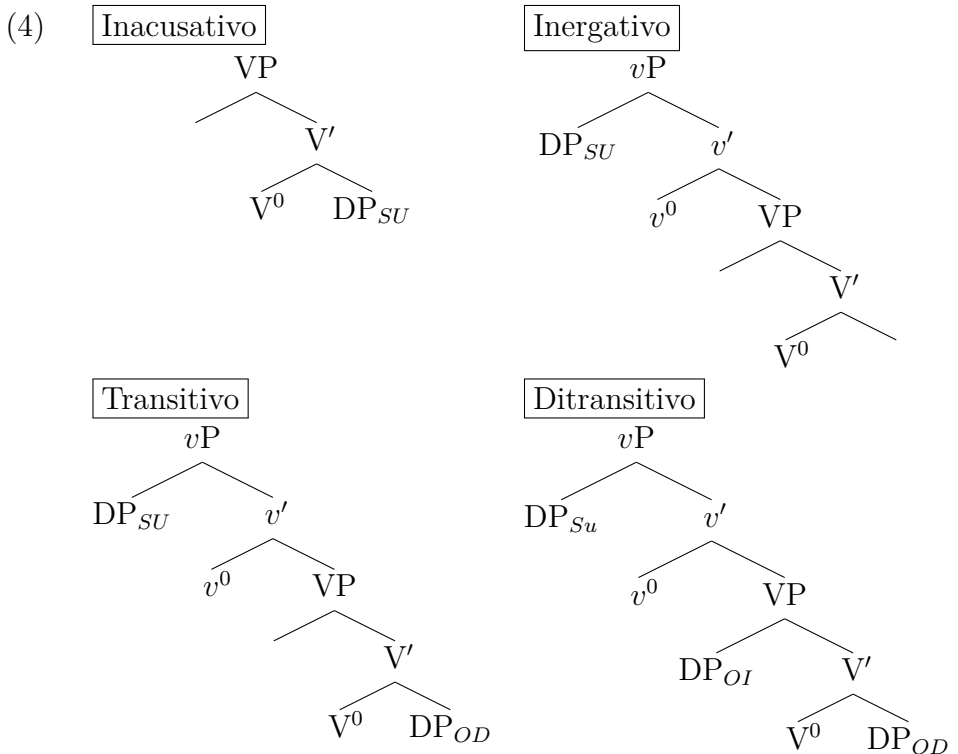
Asimismo, seguimos el supuesto de que los argumentos nominales son proyecciones de núcleos determinantes. En este sentido, asumiremos que tanto los sustantivos determinados (2a), como los nombres propios (2b), los pronombres personales (2c), interrogativos (2d) y relativos (2e) (entre otros) son determinantes, *i.e.*, núcleos D<sup>o</sup>.

- (2)
- a. *la/una manzana*
  - b. *María, Ana, Pedro, Ficciones, Rayuela, ...*
  - c. *él, ella, yo, nosotros, me, lo, le, ...*
  - d. *qué, quién, cuál, ...*
  - e. *que, quien, el/la, cual, ...*

En lo que sigue, vamos a adoptar, fundamentalmente, una perspectiva lexicalista fuerte –aunque tomaremos una visión un poco más débil cuando veamos *affix hopping*–. Adoptar esta perspectiva implica que las palabras no se forman en el componente sintáctico, sino en un componente independiente. Así, los ítems léxicos pueden ser representados como matrices de rasgos sintácticos, fonológicos y semánticos no ordenados.

$$(3) \begin{bmatrix} \text{cat:} & V \\ \text{sem:} & \dots \\ \text{fon:} & /cocinar/ \\ \text{sel:} & \text{---} \{DP, NP\} \end{bmatrix} \begin{bmatrix} \text{cat:} & D \\ \text{sem:} & \dots \\ \text{fon:} & /la/ \\ \text{sel:} & \text{---} NP \\ \text{lic:} & -\text{case} \end{bmatrix} \begin{bmatrix} \text{cat:} & v \\ \text{sem:} & \dots \\ \text{fon:} & \epsilon \\ \text{sel:} & DP \text{ ---} VP \\ \text{lic:} & +\text{case} \end{bmatrix}$$

Con estos supuestos en mente, podemos asumir, sin controversias, las siguientes estructuras básicas para verbos inacusativos, inergativos, transitivos y ditransitivos:



## 2.1. Bare Phrase Structure

En el marco del *Programa Minimalista*, la Teoría de X-barra es abandonada y reemplazada por la teoría de *Bare Phrase Structure*. Desde esta perspectiva, los niveles de barra son eliminados como tales. Así, la estructura de frase se forma por al operación *merge* (ensamble), que básicamente se encarga de:

- combinar dos objetos sintácticos,

- etiquetar al objeto sintáctico resultante de la combinación.

Como discutiremos en breve, *merge*, eventualmente, podría ser responsable del orden lineal. Repasemos el funcionamiento de esta operación.

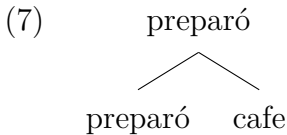
Supongamos que tenemos dos objetos sintácticos (dos elementos insertados en la derivación) *preparó* y *café*. La operación *merge* toma ambos objetos y crea un objeto sintáctico nuevo que es el resultado de la combinación de estos dos elementos:

- (5) a. *café*, *preparó*  
 b. *merge*(*preparó*, *café*)  $\Rightarrow$  {*preparó*, *café*}

En la representación de (5b) falta el etiquetado de la estructura resultante. En principio, podemos asumir que la identificación de la etiqueta está determinada por el núcleo que tiene los rasgos de selección, en este caso, sería *preparó* y no *café*. Así las cosas, el resultado de *merge* en (5b) en debería ser (6), en el que se inserta la etiqueta que identifica al núcleo de la estructura.

- (6) {*preparó*{*preparó*, *café*}

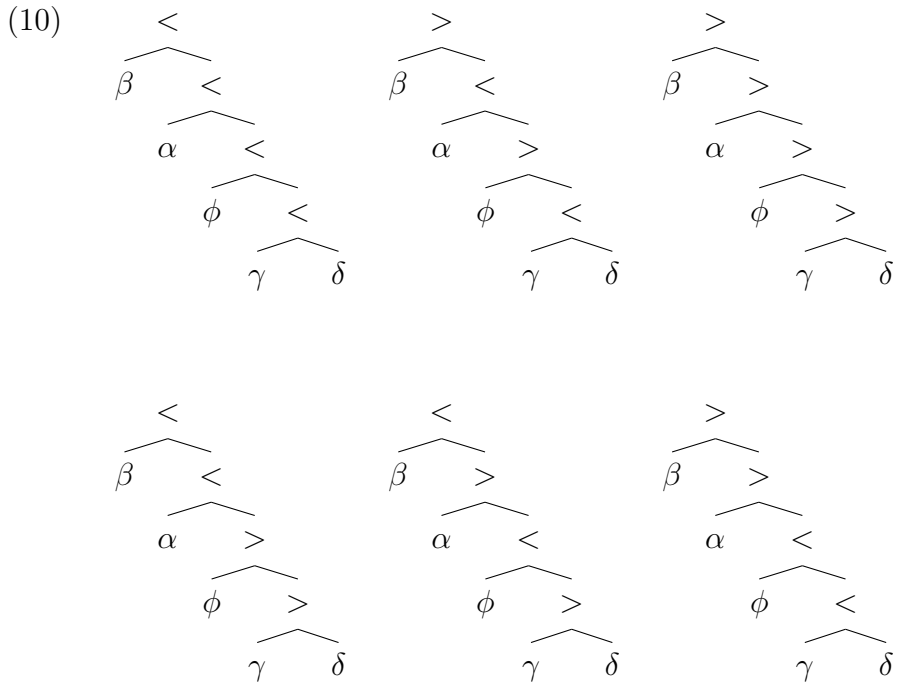
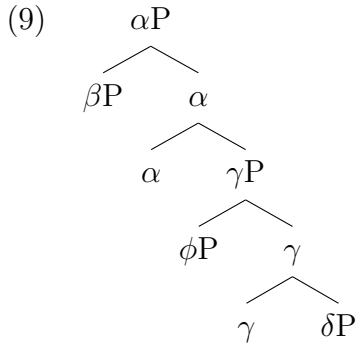
La representación en términos de la teoría de conjuntos de (6) es equivalente a la siguiente representación en forma de árbol:



El sistema de [Stabler \(2011\)](#) sigue la propuesta de eliminación de las etiquetas, pues introducen una redundancia innecesaria. Esto se debe a que lo relevante es identificar los núcleos y sus propiedades, que determinan la posibilidad de participar en sucesivas operaciones de *merge*. Las etiquetas, entonces, son reemplazadas por los símbolos  $>$  y  $<$  que señalan al núcleo. Así (7) se puede representar como en (8):



**Ejercicio 1** ¿Cuál de las estructuras que aparecen a continuación es el *bare tree* apropiado para el siguiente árbol en la notación de X-barra?



## 2.2. Movimiento

Uno de los objetivos centrales de la gramática generativa es el de explicar el hecho de que ciertos constituyentes se interpretan en posiciones diferentes de las que se oyen. Sin ir más lejos, en (11), el

pronombre interrogativo *qué* se pronuncia junto al verbo principal *creer* pero se interpreta como el objeto de *comprar*, que tiene dos niveles de incrustación con respecto a la oración principal.

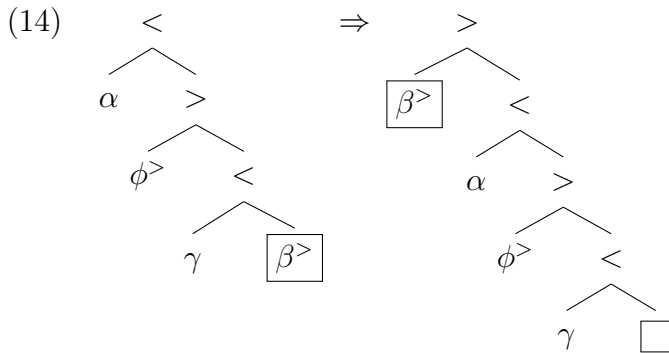
- (11) [¿Qué creés [O que dijo María [O que compró Juan \_\_\_?]]]
- 

Esta característica se conoce como la propiedad de dislocación de las lenguas naturales. En el marco generativo, esta propiedad se explica mediante la operación de movimiento. Dentro del Programa Minimalista, el movimiento se reinterpreta como un tipo particular de *merge*, que consiste en tomar un constituyente que es parte de un árbol y se ensambla en otra posición.

- (12)
- 

En el marco de la propuesta de Stabler que vimos la clase pasada, el movimiento supone ensamblar un subárbol como especificador de otro nodo y reemplazar la posición original por el nodo vacío  $\epsilon$ .

- (13)
-



### 3. Minimalist Grammar (Básicas)

Recordemos los “ingredientes” de las gramáticas minimalistas que vimos la última clase.

Vocabulario	{casa, juan, comió, ε, ...}
Tipos de items	{:, : }
Rasgos sintácticos	C, T, V, N, ... =C, =T, =V, =N, ... +wh, +focus, +caso, ... -wh, -focus, -caso, ...
Árboles/expresiones	{>, < }

Operaciones

$$em(t_{1[=x]}, t_{2[x]}) = \begin{cases} < & \text{if } |t_1|=1 \\ \begin{array}{c} \wedge \\ t_1 \quad t_2 \end{array} & \\ > & \text{otherwise} \\ \begin{array}{c} \wedge \\ t_2 \quad t_1 \end{array} & \end{cases}$$

$$im(t_{1[+x]}) = \begin{array}{c} > \\ \wedge \\ t_2^M \quad t_{1\{t_2[-x]^M \rightarrow \varepsilon\}} \end{array} \quad \text{si SMC}$$

Con estos elementos es posible construir ítems léxicos análogos a los (3), pero, con una diferencia central, porque en este caso el

ordenamiento de los rasgos es relevante. Recordemos que al aplicarse *external merge* o *internal merge*, los rasgos que inducen la aplicación de la operación son borrados. El borrado de los rasgos es necesario porque de otro modo los otros rasgos no resulta visibles para las siguientes operaciones. Veamos el punto con mayor detalle.

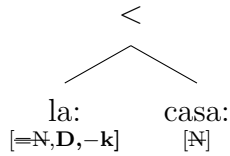
El léxico que aparece en (15) permite derivar la expresión *falso español* de (16):

- (15) a. casa :: N  
 b. la :: =N, D, -k  
 c. pintó :: =D,+k,V

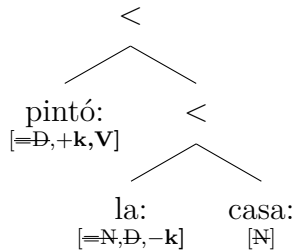
(16) la casa pintó

Revisemos como se construye de (16) paso a paso. Los rasgos recuadrados son los que desencadenan *merge* en cada estado y los rasgos tachados indican que han sido borrados.

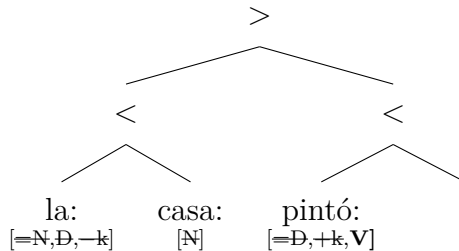
- (17) a.  $\text{em}(\text{la}::\boxed{=N}, D, -k + \text{casa}::\boxed{N}) \Rightarrow$



- b.  $\text{em}(\text{pintó}::\boxed{=D}, +k, V + \{\text{la}::\boxed{=N}, \boxed{D}, -k \dots\}) \Rightarrow$



- c.  $\text{im}(\text{pintó}::\boxed{=D}, \boxed{+k}, V \{\dots\}) \Rightarrow$





**Ejercicio 2** ¿Qué sucedería si los rasgos de *la* estuvieran ordenados de otra manera? Construya las gramáticas correspondientes y pruébelas en la implementación de Stabler. (Agregue las líneas necesarias en la sección de gramáticas en el archivo `setup.pl`).

```
(18) % file: gprueba1.pl
```

```
[pintó] :: [= 'D', +k, 'V'].
[casa]  :: ['N'].
[la]   :: [= 'N', 'D', -k]. % <-- Línea relevante

startCategory('V').
```

```
(19) % file: gprueba2.pl
```

```
[pintó] :: [= 'D', +k, 'V'].
[casa]  :: ['N'].
[la]   :: ['D', = 'N', -k]. % <-- Línea relevante

startCategory('V').
```

```
(20) % file: gprueba3.pl
```

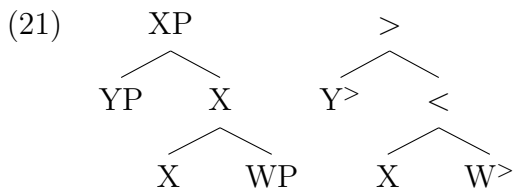
```
[pintó] :: [= 'D', +k, 'V'].
[casa]  :: ['N'].
[la]   :: [-k, 'D', = 'N']. % <-- Línea relevante

startCategory('V').
```

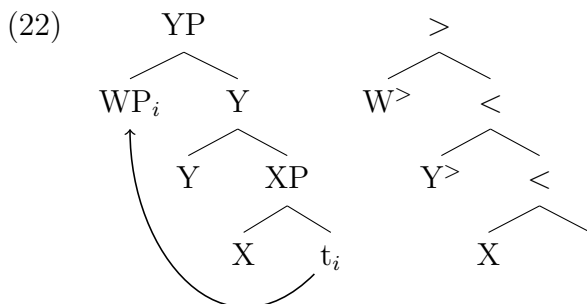
Esta gramática mínima es similar a las gramáticas `g0.pl` y `g0spanish.pl`, revisadas la clase pasada, por el hecho de que emplean rasgos de licenciamiento  $[\pm f]$ . Estos rasgos inducen la operación *internal merge* cuya consecuencia es el cotejo/chequeo de rasgos.

### 3.1. Observaciones sobre el orden lineal

Es interesante notar que las implementaciones que estamos revisando construyen estructuras con núcleo inicial *-i.e.*, núcleo a la izquierda-, como (21).



Las estructuras de núcleo final *-i.e.*, núcleo a la derecha- son configuraciones derivadas (26):



Más allá de la discusión en torno de si el componente sintáctico determina el orden lineal (véase Kayne, 1994; Kayne, 2011), la operación *external merge* que emplean estas gramáticas coloca necesariamente el complemento a la derecha del núcleo y el especificador a la izquierda, como parece desprenderse de la definición formal de esta operación. Los movimientos de licenciamiento, en cambio, colocan necesariamente a los consituyentes desplazados en una posición en que preceden al núcleo. En consecuencias, Las estructuras de (21) da como resultado la secuencia de (23), mientras que la de (26) produce el orden lineal de (24):

$$(23) \quad Y \frown X \frown W \quad (\text{ver } 21)$$

$$(24) \quad W \frown Y \frown X \quad (\text{ver } 26)$$

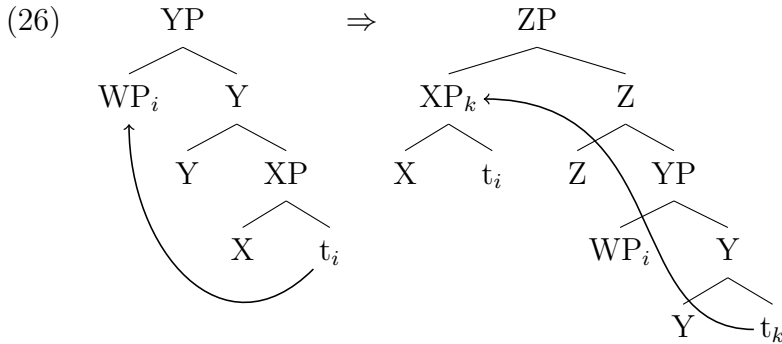
En pocas palabras, si el objeto se mueve para cotejar y eliminar su rasgos de caso *-i.e.*, *-k*, *-case*, etc., entonces siempre se genera la estructura cuya secuencia lineal es la de (25).

$$(25) \quad (\text{Sujeto}) \frown \text{Objeto} \frown \text{Verbo}$$

Si las gramáticas que estamos revisando no emplean movimiento nuclear, la pregunta que surge es cómo se obtienen, entonces, estructuras SVO.

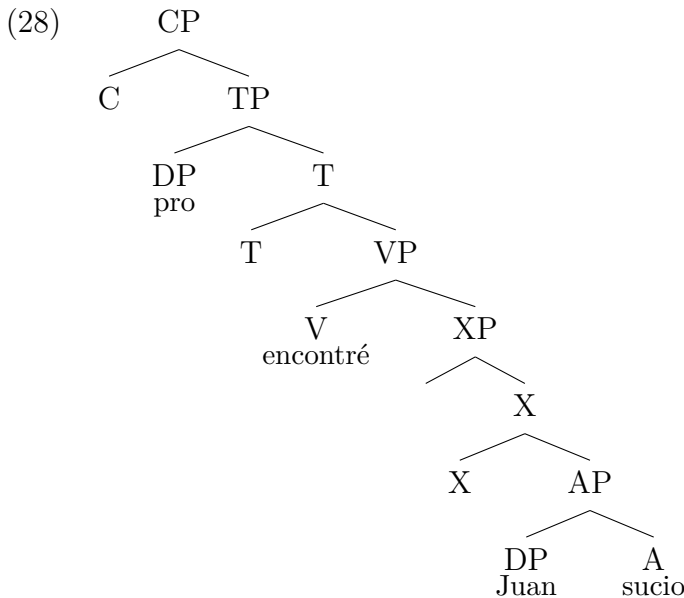
**¿Cómo se obtienen estructuras SVO, si la gramática emplea rasgos de licenciamiento?**

Estas gramáticas adoptan de manera generalizada un tipo de movimiento conocido como movimiento de remanente [*remnant movement*]. En términos sencillos, este movimiento consiste en el desplazamiento de una porción de la estructura oracional después de haber extraído un constituyente de su interior.

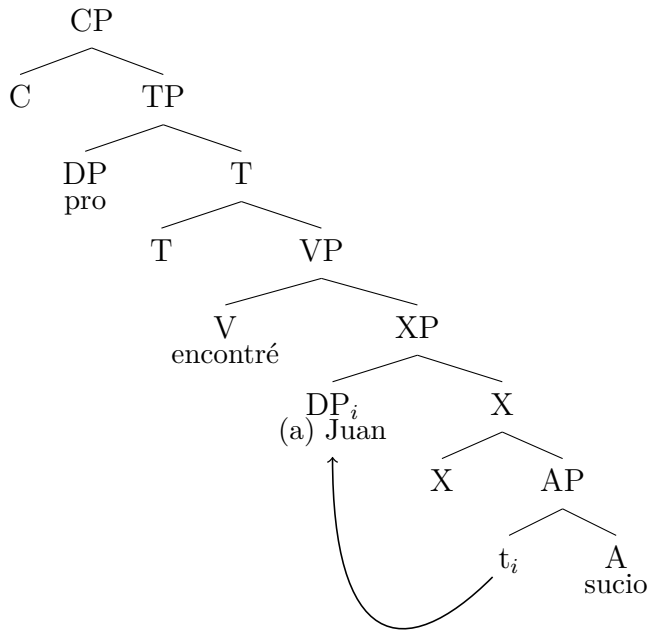


Por ejemplo, este tipo de movimiento podría ser considerado para explicar las alternativas de (27), que derivarían de la estructura básica de (28):

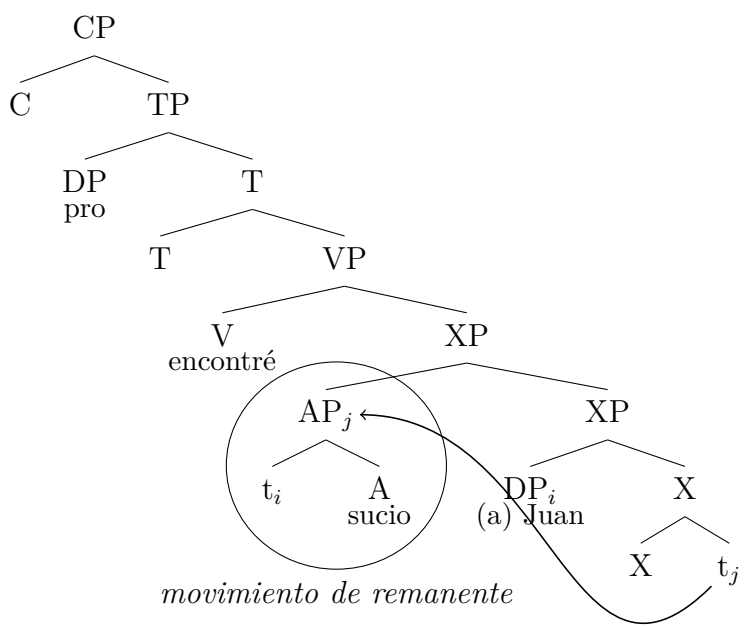
- (27) a. Encontré a Juan sucio.  
 b. Encontré sucio a Juan.




(29)



(30)

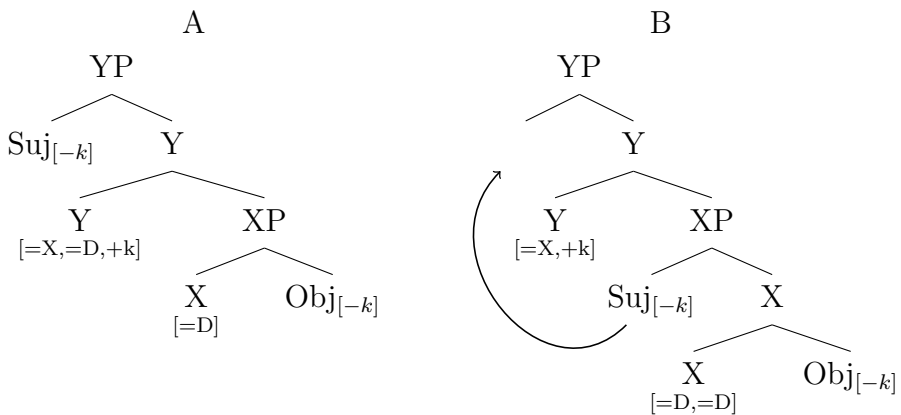


 Revisemos la gramática `g-ne.pl` de [Stabler](#) para ver el funcionamiento de la gramática con rasgos de licenciamiento  $\pm wh$  y  $\pm k$

Siguiendo la estructura de `g-ne.pl`, elaboramos la gramática `g-nSP.pl`, que permite derivar algunas oraciones del español. Un punto importante para tener en cuenta es que estas gramáticas contemplan el licenciamiento de caso, es decir que los argumentos tienen que portar el rasgo  $[-k]$  o  $[-case]$ .

**Ejercicio 3** Observen con detenimiento las gramáticas `g-ne.pl` y `g-nSP.pl`. Indique de qué manera están distribuidos los rasgos para insertar al objeto y el sujeto en la derivación y cotejar los rasgos de caso  $[-k]$ .

**Ejercicio 4** Las estructuras que aparecen a continuación resultan conflictivas para el tipo gramáticas que acabamos de revisar.

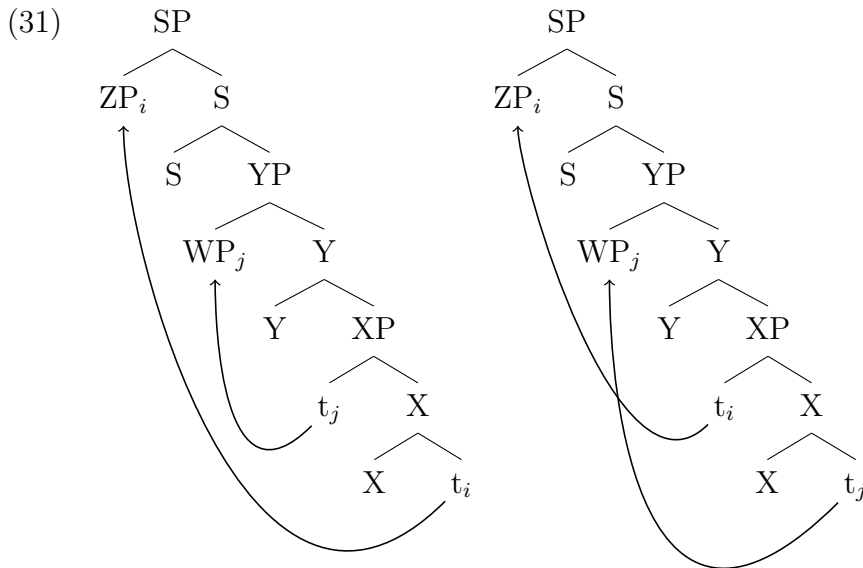


1. Construya los items léxicos correspondientes para intentar inducir estas derivaciones parciales.

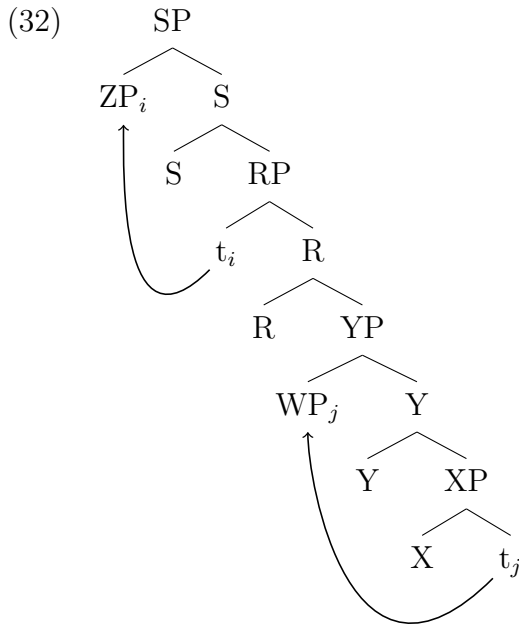
2. Pruebe cómo funcionarían los items léxicos creados, haciendo las modificaciones necesarias en la gramática `g-nSP.pl`.
3. Explique cuál es el problema que presentan estas modificaciones.

### 3.2. Anidamiento y entrecruzamiento

Como acabamos de ver, las gramáticas `g-ne.pl` y `g-nSP.pl` no pueden generar los movimientos de licenciamiento por  $[\pm k]$  que involucren trayectos anidados o cruzados:




El problema reside en que en cualquiera de estas alternativas quedarían rasgos sin satisfacer. La solución de las gramáticas `g-ne.pl` y `g-nSP.pl` es que el cotejo del rasgo  $-k$  del objeto ocurra en una posición por debajo de punto en que se introduce el sujeto:



Con todo, es interesante notar que Chomsky (1995, p. 171) observa respecto del cotejo de caso que “[...] crossing and not nesting is the only permissible option in any language”.

**Ejercicio 5** Tomando el conjunto de supuesto que hemos considerados hasta el momento, de qué manera podrían ser reformuladas las gramáticas `g-ne.pl` y `g-nSP.pl` de modo tal que admitan el cruzamiento de movimientos para el cotejo de caso del objeto y del sujeto.

 Revisemos qué alternativa encuentra la gramática `g0sp-caso.pl` para permitir movimientos cruzados.

## 4. Lenguaje $a^n b^n c^n$ ( $n \geq 1$ )

Hasta el momento vimos cómo la propuesta de [Stabler](#) permite construir una gramáticas capaces de implementar *external merge*, pero fundamentalmente *internal merge*.

En lo que sigue veremos que estas herramientas permiten diseñar una gramática capaz de generar el lenguaje  $a^n b^n c^n$  ( $n \geq 1$ ).

**Recordemos** El lenguaje  $a^n b^n c^n$  puede ser generado por una gramática sensible al contexto como la siguiente:


- (33) I.  $S \rightarrow aSBC$   
II.  $S \rightarrow aBC$   
III.  $CB \rightarrow BC$   
IV.  $aB \rightarrow ab$   
V.  $bB \rightarrow bb$   
VI.  $bC \rightarrow bc$   
VII.  $cC \rightarrow bcc$

- (34) 1. S  
2. aBC II  
3. abC III  
4. abc VI

- (35) 1. S  
2. aSBC I  
3. aaBCBC II  
4. aaBBCC III  
5. aabBCC IV  
6. aabbCC V  
7. aabbcC VI  
8. aabbcc VII



**Ejercicio 6** Derive la oración aaabbbccc.

 Veamos la gramática `anbncn.pl` desarrollada por Stabler.

## Referencias

Chomsky, N. (1995). *The minimalist program*. MIT Press, Cambridge, Massachusetts. Manejamos la traducción de Juan Romero Morales, Madrid: Alianza, 1999.

Kayne, R. S. (1994). *The antisymmetry of syntax*, volumen 25. MIT press.

Kayne, R. S. (2011). Why are there no directionality parameters. En *Proceedings of WCCFL*, volumen 28, pp. 1–23.

Stabler, E. P. (2011). Computational perspectives on minimalism. En Boeckx, C., editor, *Oxford handbook of linguistic minimalism*, pp. 617–643. Oxford.