

Análisis sintáctico

Miguel A. Alonso Carlos Gómez Jesús Vilares

Departamento de Computación, Facultad de Informática, Universidade da Coruña

1 Introducción

- Análisis basado en constituyentes
- El problema de la ambigüedad
- Análisis basado en dependencias

2 Esquemas de análisis sintáctico

3 El algoritmo de Earley

- Hipótesis
- Ítems
- Pasos deductivos
- Ítems finales
- Ejemplo

4 Análisis sintáctico probabilístico

Índice

1 Introducción

- Análisis basado en constituyentes
- El problema de la ambigüedad
- Análisis basado en dependencias

2 Esquemas de análisis sintáctico

3 El algoritmo de Earley

- Hipótesis
- Ítems
- Pasos deductivos
- Ítems finales
- Ejemplo

4 Análisis sintáctico probabilístico

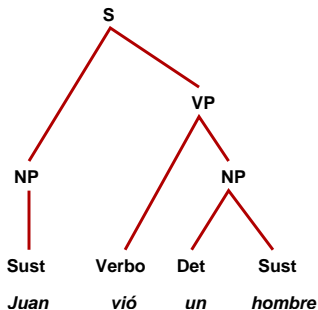
La estructura de las frases

- La **sintaxis** describe la **estructura** de las frases de un lenguaje
- Una frase bien formada se puede descomponer en constituyentes de acuerdo a unas **reglas sintácticas**
- Las reglas sintácticas se describen mediante un **formalismo gramatical**
- Un **analizador sintáctico** es un programa de ordenador que obtiene la estructura asociada a una frase dada de acuerdo con una gramática.

Ejemplo: análisis sintáctico con CFG

- Frase: “Juan vio un hombre”
- Gramática: $S \rightarrow NP VP$
 $NP \rightarrow Sust$
 $NP \rightarrow Det Sust$
 $VP \rightarrow Verbo NP$

- Estructura:



Problema: la ambigüedad

Las gramáticas de los lenguajes naturales son ambiguas: se pueden asociar varias estructuras a una misma frase

$S \rightarrow NP VP$

$S \rightarrow S PP$

$NP \rightarrow Sust$

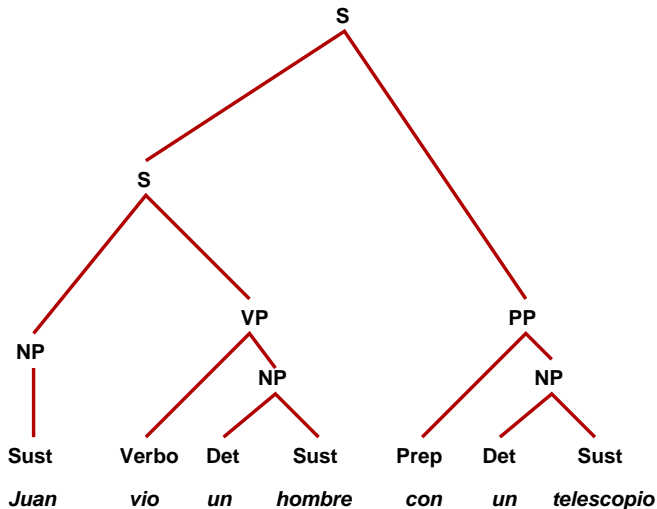
$NP \rightarrow Det Sust$

$NP \rightarrow NP PP$

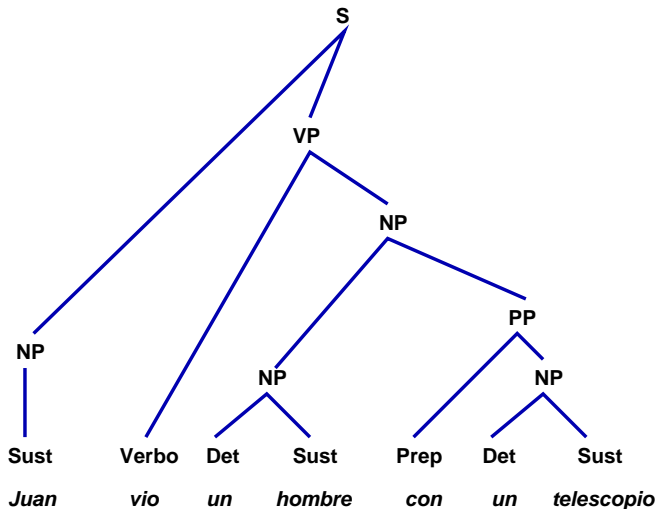
$PP \rightarrow Prep NP$

$VP \rightarrow Verbo NP$

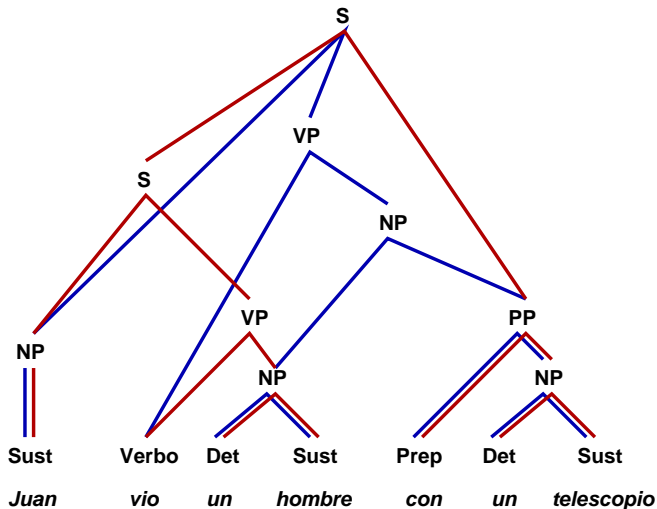
Problema: la ambigüedad



Problema: la ambigüedad



Problema: la ambigüedad



¿Cómo manejar la ambigüedad del análisis?

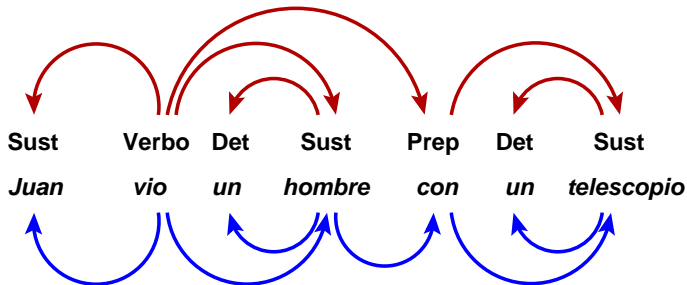
Dos grandes enfoques:

- Guardar todos los posibles análisis de forma eficiente:
 - Evitar la realización de cálculos redundantes
 - Compartir partes de la frase entre varios análisis
 - Unir ramas de análisis que difieren en ciertas subestructuras
- Trabajar con un sólo análisis:
 - Selección del mejor análisis mediante cálculos probabilísticos
 - y/o seleccionar el análisis que cubre una parte mayor de la frase
 - y/o análisis superficial de *chunks*
 - y/o heurísticas
 - y/o ...

Análisis basado en dependencias

Un enfoque diferente del tradicional que está ganando popularidad

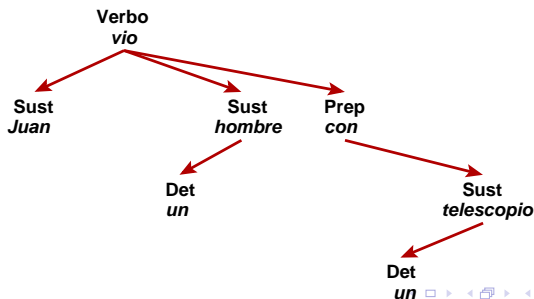
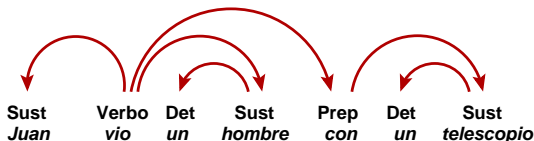
- Trabajar con las relaciones existentes entre las palabras de la frase



- Las dependencias pueden estar anotadas (tener nombre)
- **Aprenden** de un corpus anotado sintácticamente (Treebank de dependencias)
- (normalmente) no hay una gramática

Árbol de dependencias

El conjunto de dependencias de una frase es más manejable en forma de árbol de dependencias. Los nodos del árbol son las palabras y las aristas indican dependencias.



Análisis basado en dependencias

Ventajas:

- Siempre se obtiene un análisis
- ... y sólo uno
- Algunos analizadores de dependencias son muy rápidos

Inconvenientes:

- Necesidad de un treebank
- La fase de entrenamiento/aprendizaje es muy lenta
- para el entrenamiento/aprendizaje se emplean modelos estadísticos de clasificación complejos (SVM, ...) que hacen asunciones sobre los datos que se sabe que no se cumplen y cuyo comportamiento no se comprende bien.
- Algunos analizadores de dependencias son muy lentos

Lectura recomendada para el tema

Carlos Gómez-Rodríguez,

Parsing Schemata for Practical Text Analysis,

volume 1 of *Mathematics, Computing, Language, and Life: Frontiers in Mathematical Linguistics and Language Theory* series,

Imperial College Press and World Scientific, London, 2010.

ISBN 978-1-84816-560-1.

- 1 Introducción
 - Análisis basado en constituyentes
 - El problema de la ambigüedad
 - Análisis basado en dependencias
- 2 Esquemas de análisis sintáctico
- 3 El algoritmo de Earley
 - Hipótesis
 - Ítems
 - Pasos deductivos
 - Ítems finales
 - Ejemplo
- 4 Análisis sintáctico probabilístico

Algoritmos más conocidos

- Basados en *backtracking* (estilo Prolog, ineficientes)
- CYK (Cocke, Younger, Kasami)
- Earley
- Left-Corner
- Generalized LR

Programación dinámica

- Los algoritmos de análisis sintáctico eficiente hacen uso de la **programación dinámica**:
 - almacenando los resultados intermedios en **ítems**
 - combinando ítems mediante **pasos deductivos** para obtener análisis de porciones cada vez mayores de la cadena de entrada
 - evitando la realización de cálculos redundantes
- Complejidad temporal $\mathcal{O}(n^3)$
- Complejidad espacial $\mathcal{O}(n^2)$
- donde n es la longitud de la frase.

Esquemas de análisis sintáctico

Formalismo declarativo para la definición de algoritmos de análisis sintáctico

- Independiente de las estructuras de datos
- Independiente de los algoritmos de manipulación

Proporciona un entorno homogéneo de alto nivel (abstracto) para la definición de algoritmos

- Permite identificar algoritmos que son iguales salvo por aspectos concretos de implementación
- Permite derivar unos algoritmos a partir de otros mediante transformaciones simples bien definidas
- Permite crear una **red de relaciones** entre analizadores sintácticos
- Facilita las demostraciones de corrección
- Hace evidentes los costes (complejidad temporal y espacial)

Esquemas de análisis sintáctico

- Las palabras de la frase a analizar constituyen las **hipótesis** o ítems iniciales
- Los resultados intermedios se almacenan en forma de **ítems**
- Los ítems se combinan mediante **pasos deductivos**, reglas de inferencia que permite obtener nuevos ítems a partir de ítems e hipótesis existentes.
- La presencia de **ítems finales** nos indica si el análisis tuvo éxito.

Ejemplo del algoritmo ya conocido: El esquema CYK

Estrategia ascendente pura para gramáticas en Forma Normal de Chomsky

- *Hipótesis*

$$H = \{[a_i, i - 1, i] \mid 0 < i \leq n\}$$

- Conjunto de ítems

$$I = \{[A, i, j] \mid A \in N \wedge 0 \leq i < j\}$$

- Pasos deductivos

$$D(1) : \frac{[a, i - 1, i]}{[A, i - 1, i]} A \rightarrow a \in P$$

$$D(2) : \frac{[B, i, j] \quad [C, j, k]}{[A, i, k]} A \rightarrow BC \in P$$

- Ítems finales

$$I_f = \{[S, 0, n]\}$$

Compilación de esquemas de análisis sintáctico

Los esquemas de análisis sintáctico **no son directamente ejecutables**.

Dos opciones:

- Implementación específica
- Implementación genérica: [máquina deductiva de análisis sintáctico](#)

Es posible transformar automáticamente (compilar) un esquema de análisis sintáctico a una implementación ejecutable eficiente basada en la máquina deductiva de análisis sintáctico.

Detalles:

*Carlos Gómez-Rodríguez, Jesús Vilares and Miguel A. Alonso,
"A compiler for parsing schemata",
Software: Practice and Experience, 39(5):441-470, 2009.
ISSN 0038-0644. DOI 10.1002/spe.904*

Máquina deductiva de análisis sintáctico

- Entradas: una cadena de entrada
una gramática
- Salida: un conjunto de ítems que representa el proceso de análisis
- Estructuras de datos auxiliares: una **tabla** de ítems
una cola de ítems (**agenda**)
- Proceso:
 - 1 Aplicar los pasos deductivos que no tienen ítems antecedentes, almacenando los ítems resultantes en la tabla y en la agenda
 - 2 Repetir hasta que la agenda esté vacía:
 - 1 Sacar un ítem de la agenda
 - 2 Aplicar todos los pasos deductivos que lo utilicen como antecedente
 - 3 Almacenar los ítems consecuentes en la tabla y en la agenda
- En los pasos 1) y 2.c) se comprobará que los ítems no estén repetidos

Índice

- 1 Introducción
 - Análisis basado en constituyentes
 - El problema de la ambigüedad
 - Análisis basado en dependencias
- 2 Esquemas de análisis sintáctico
- 3 El algoritmo de Earley
 - Hipótesis
 - Ítems
 - Pasos deductivos
 - Ítems finales
 - Ejemplo
- 4 Análisis sintáctico probabilístico

El algoritmo de Earley

- Estrategia mixta:
 - Fase **predictiva** (descendente) que garantiza que sólo se utilizan aquellas reglas que son derivables a partir del axioma de la gramática.
 - Fase **ascendente** que combina análisis aprciales
 - No establece restricciones en la forma de la gramática
 - Misma complejidad que CYK:
 - $\mathcal{O}(n^3)$ en tiempo
 - $\mathcal{O}(n^2)$ en espacio
- donde n es al longitud de la frase

Notación

- Sea una gramática $G = (N, \Sigma, P, S)$
- N es el conjunto de símbolos **no-terminales** (denotados por letras mayúsculas)
- Σ es el conjunto de símbolos **terminales** (denotados por letras minúsculas)
- P es el conjunto de **producciones** o reglas de la forma $A \rightarrow \alpha$ (donde α es una cadena de terminales y no terminales)
- $S \in N$ es el **axioma** (el símbolo no-terminal que etiqueta la raíz de un árbol sintáctico completo para una frase)

El algoritmo clásico

```

S = array [0..n] of state sets;
for i = 0 ... n { S[i] = ∅; } //initialize the n+1 sets to ∅
for each rule S → α ∈ P //initialize S[0]
  S[0] = S[0] ∪ {(S → α, 0, 0)};
for i = 0 ... n {
  process the members of S[i] in order, executing each of these operations on
  each state (A → α, j, f) until no more of them can be applied:
  1) Predictor:
    X = j + 1th symbol in α;
    if X exists and is a nonterminal
      for each production of the form X → β in P
        S[i] = S[i] ∪ {(X → β, 0, i)};
  2) Completer:
    if X does not exist //(j + 1 > |α|)
      for each state (B → β, l, g) in S[f] {
        Y = l + 1th symbol in β;
        if Y exists ∧ Y = A
          S[i] = S[i] ∪ {(B → β, l + 1, g)};
      }
  3) Scanner:
    if X exists and is a terminal
      if X = ai+1
        S[i + 1] = S[i + 1] ∪ {(A → α, j + 1, f)};
}

```

if S[n] contains a state of the form (S → γ, |γ|, 0) return true; else return false;

El algoritmo clásico

La idea es sencilla pero el pseudocódigo es...

- relativamente complejo
- poco declarativo
- los detalles de las estructuras de datos y de control dificultan la comprensión de los mecanismos básicos del algoritmo
- además, no es la única implementación posible del algoritmo (incluso se cuestiona que sea la más eficiente)

En resumen, *los árboles no dejan ver el bosque*

El esquema Earley

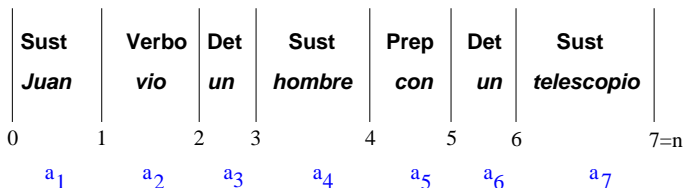
Madiante la definición del esquema de análisis sintáctico de Earley trataremos de ...

- centrarnos en la idea, abstraernos de los detalles de implementación
- comprender los principios que rigen el algoritmo

Hipótesis

El conjunto de hipótesis o ítems iniciales es el estándar

$$H = \{[a_i, i - 1, i] \mid 0 < i \leq n\}$$

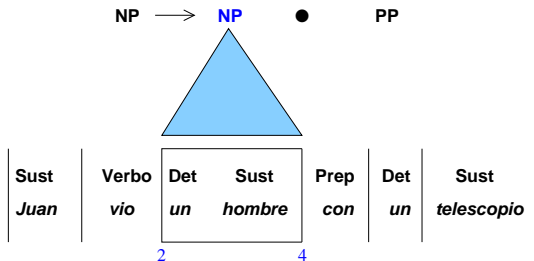


Ítems

$$I = \{[A \rightarrow \alpha \bullet \beta, i, j] \mid A \rightarrow \alpha\beta \in P \wedge 0 \leq i < j \wedge S \xrightarrow{*} A \wedge \alpha \xrightarrow{*} a_{i+1} \dots a_j\}$$

- $A \rightarrow \alpha\beta$ es una producción de la gramática
- A es derivable desde el axioma
- A partir de α se deriva la parte contigua de la frase formada por $a_{i+1} \dots a_j$
- A partir de β no sabemos lo que se derivará, pero para poder completar el análisis será necesario que sea una parte contigua de la cadena que empiece en la posición j

Ejemplo concreto de ítem

$$[NP \rightarrow NP \bullet PP, 2, 4]$$


Además sabemos que $S \xrightarrow{*} NP$ y el algoritmo va a garantizar que este ítem forma parte de un análisis viable en el sentido de que la parte leída de la frase (*Juan vio un hombre*) es el prefijo de alguna frase válida en el lenguaje generado por la gramática (**Propiedad del prefijo válido**)

Paso deductivo Initter

$$\text{INITTER: } \overline{[S \rightarrow \bullet \alpha, 0, 0]}$$

Inicialmente sólo sabemos que:

- el análisis debe comenzar por la raíz del árbol de análisis sintáctico (axioma)
- comenzaremos a leer la frase a partir del comienzo (posición 0)
- Intentaremos todas las producciones que deriven del axioma

Paso deductivo Predictor

$$\text{PREDICTOR: } \frac{[A \rightarrow \alpha \bullet B\beta, i, j]}{[B \rightarrow \bullet \gamma, j, j]}$$

Si

- hemos activado una regla $A \rightarrow \alpha B\beta$
- la parte α deriva $a_{i+1} \dots a_j$
- para avanzar en la producción, necesitamos derivar un trozo de cadena a partir del no-terminal B y de la posición j de la frase

entonces

- activaremos todas las producciones que tengan B como no-terminal de su lado izquierdo, indicando que comenzaremos a leer la frase en la posición j

Paso deductivo Completer

$$\text{COMPLETER: } \frac{[A \rightarrow \alpha \bullet B\beta, i, j] \quad [B \rightarrow \gamma \bullet, j, k]}{[A \rightarrow \alpha B \bullet \beta, i, k]}$$

Si

- hemos activado una regla $A \rightarrow \alpha B\beta$
- la parte α deriva $a_{i+1} \dots a_j$
- para avanzar en la producción, necesitamos derivar un trozo de cadena a partir del no-terminal B y de la posición j de la frase

y

- B deriva una parte de la frase que comienza en la posición j y termina en la k

entonces

- unimos la parte derivada por α con la derivada por B , lo que resulta en que αB deriva la parte de la frase que comienza en la posición i y termina en la k

Paso deductivo Scanner

$$\text{SCANNER: } \frac{[A \rightarrow \alpha \bullet a\beta, i, j] \quad [a, j, j + 1]}{[A \rightarrow \alpha a \bullet \beta, i, j + 1]}$$

Si

- hemos activado una regla $A \rightarrow \alpha a\beta$
- la parte α deriva $a_{i+1} \dots j$
- para continuar avanzando en la producción, necesitamos leer un terminal a
- la siguiente palabra de la frase es a

entonces

- leemos a , lo que implica que avanzamos una posición en la lectura de la frase

Ítems finales

$$[S \rightarrow \gamma \bullet, 0, n]$$

- Hemos derivado toda la frase (desde la posición 0 hasta la n , que es la ultima) a partir del axioma de la gramática

Juan vio un hombre con un telescopio

agenda

[Sust, 0, 1]
Juan

[Verbo, 1, 2]
vio

[Det, 2, 3]
un

[Sust, 3, 4]
hombre

[Prep, 4, 5]
con

[Det 5, 6] [Sust, 6, 7]
un telescopio

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

[S->.NP VP,0,0]

[Sust, 0, 1]
Juan[Verbo, 1, 2]
vio[Det, 2, 3]
un[Sust, 3, 4]
hombre[Prep, 4, 5]
con[Det 5, 6] [Sust, 6, 7]
un telescopio

agenda

[S->.S PP,0,0]
[S->.NP VP,0,0]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

[S->.NP VP,0,0]

[Sust, 0, 1]
Juan[Verbo, 1, 2]
vio[Det, 2, 3]
un[Sust, 3, 4]
hombre[Prep, 4, 5]
con[Det 5, 6] [Sust, 6, 7]
un telescopio**agenda**[S->.S PP,0,0]
[S->.NP VP,0,0]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

[S->.NP VP,0,0]

[NP->.Sust,0,0]

[Sust, 0, 1]
Juan[Verbo, 1, 2]
vio[Det, 2, 3]
un[Sust, 3, 4]
hombre[Prep, 4, 5]
con[Det 5, 6] [Sust, 6, 7]
un telescopio

agenda

[S->.NP VP,0,0]

[NP->.Sust,0,0]

[NP->.Det Sust,0,0]

[NP->.NP PP,0,0]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

[S->.NP VP,0,0]

[NP->Sust.,0,1]

[NP->.Sust,0,0]

[Sust, 0, 1]

Juan

[Verbo, 1, 2]

vio

[Det, 2, 3]

un

[Sust, 3, 4]

hombre

[Prep, 4, 5]

con

[Det 5, 6] [Sust, 6, 7]

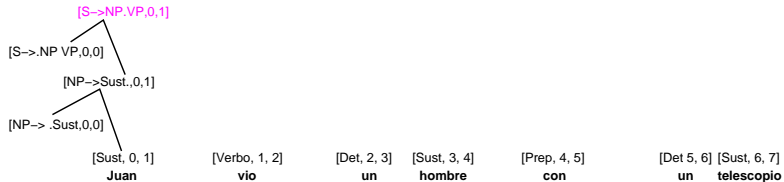
un telescopio

agenda

[NP->.Sust,0,0]
 [NP->.Det Sust,0,0]
 [NP->.NP PP,0,0]
 [NP->Sust.,0,1]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

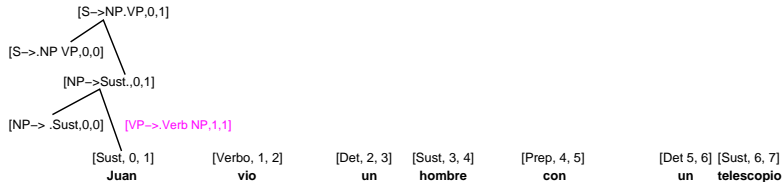


agenda

[NP->Sust.,0,1]
 [S->NP.VP,0,1]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

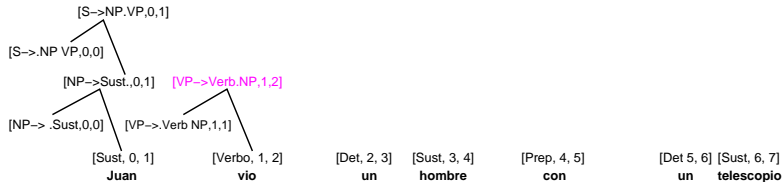


agenda

[S->NP.VP,0,1]
 [VP->.Verb NP,1,1]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

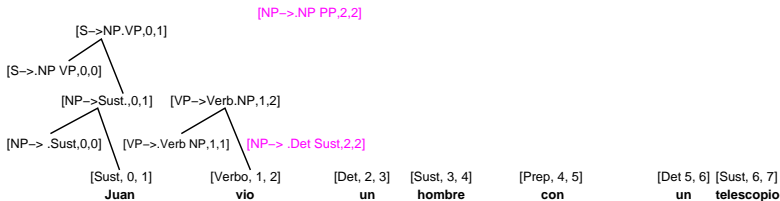


agenda

[VP->.Verb NP,1,1]
 [VP->Verb.NP,1,2]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]



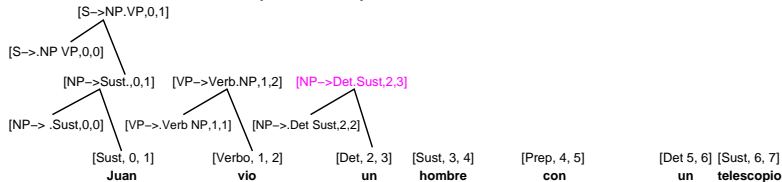
agenda

[VP->Verb.NP,1,2]
 [NP->.Sust,2,2]
 [NP->.Det Sust,2,2]
 [NP->.NP PP,2,2]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

[NP->.NP PP,2,2]



agenda

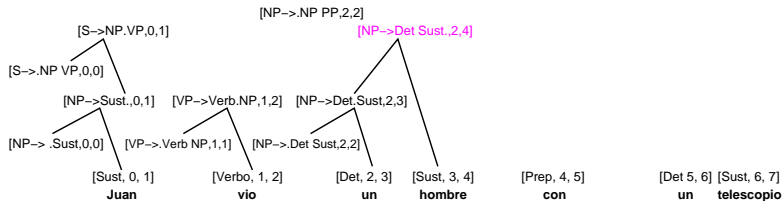
[NP->.Det Sust,2,2]

[NP->.NP PP,2,2]

[NP->.Det.Sust,2,3]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]



agenda

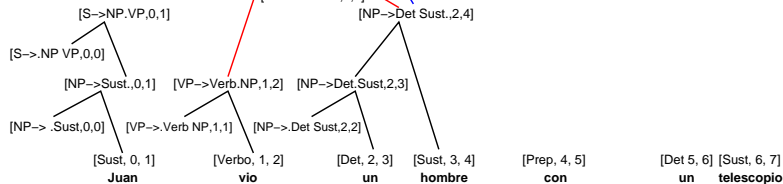
[NP->Det.Sust,2,3]

[NP->Det Sust.,2,4]

Juan vio un hombre con un telescopio

[S->.S PP,0,0]

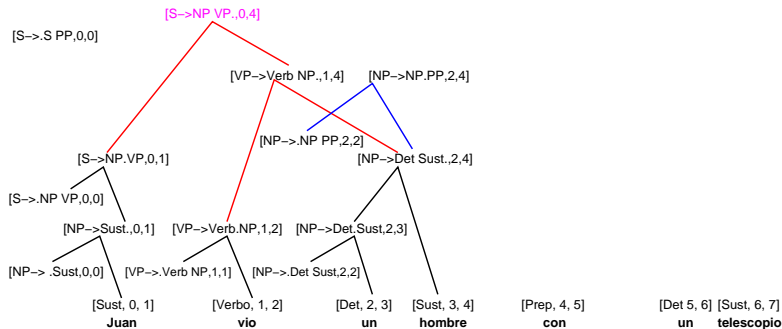
[VP->Verb NP.,1,4] [NP->NP.PP,2,4]



agenda

[NP->Det Sust.,2,4]
 [VP->Verb NP.,1,4]
 [NP->NP.PP,2,4]

Juan vio un hombre con un telescopio



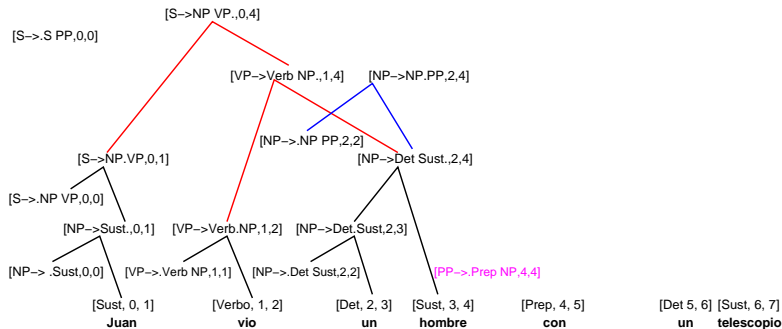
agenda

[VP->Verb NP.,1,4]

[NP->NP.PP,2,4]

[S->NP.VP.,0,4]

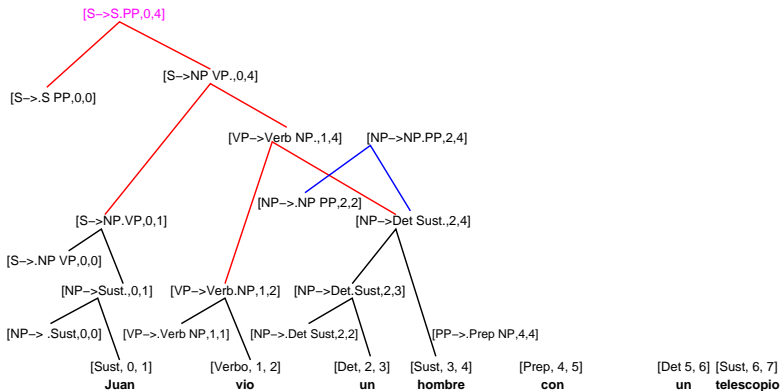
Juan vio un hombre con un telescopio



agenda

- [NP->NP.PP,2,4]
- [S->NP VP.,0,4]
- [PP->.Prep NP,4,4]

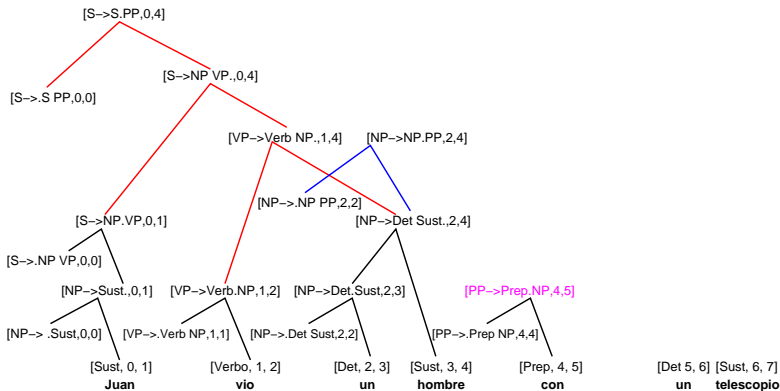
Juan vio un hombre con un telescopio



agenda

 $[S \rightarrow NP VP., 0, 4]$ $[PP \rightarrow .Prep NP, 4, 4]$ $[S \rightarrow S.PP, 0, 4]$

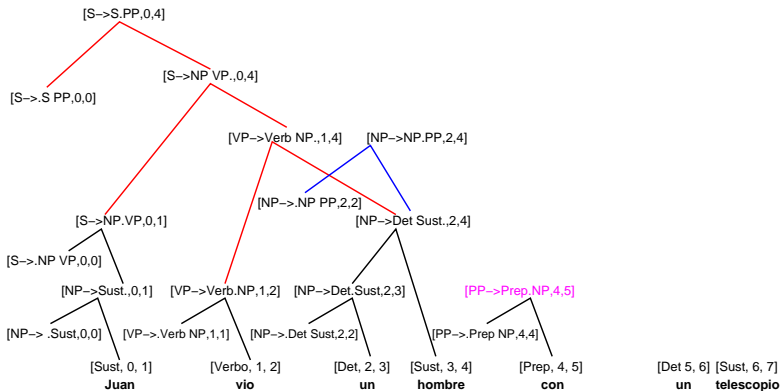
Juan vio un hombre con un telescopio



agenda

- [PP->.Prep NP,4,4]
- [S->S.PP,0,4]
- [PP->Prep.NP,4,5]

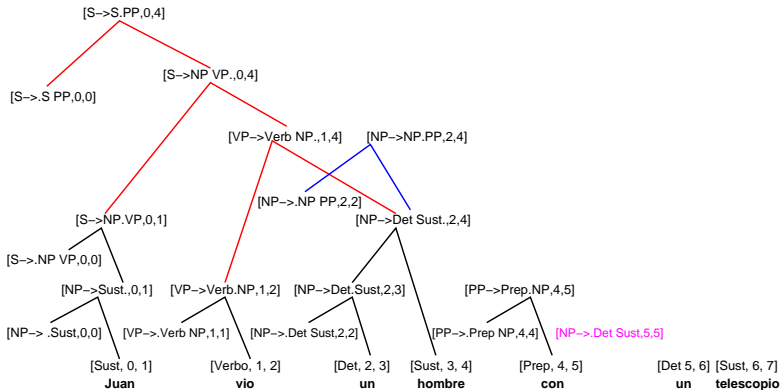
Juan vio un hombre con un telescopio



agenda

$[S \rightarrow S.PP, 0, 4]$
 $[PP \rightarrow Prep.NP, 4, 5]$

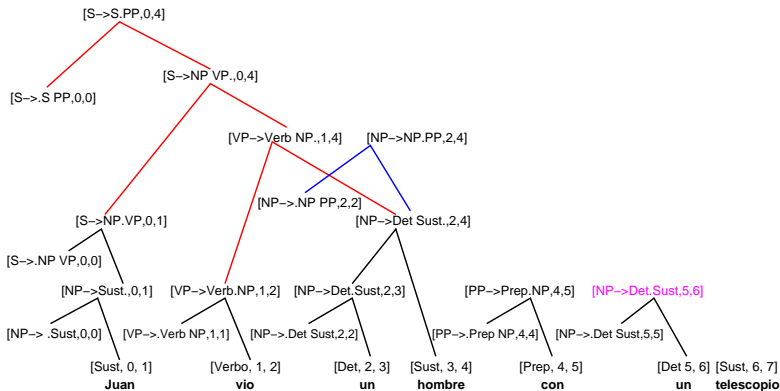
Juan vio un hombre con un telescopio



agenda

- [PP->Prep.NP,4,5]
- [NP->.Sust,5,5]
- [NP->.Det Sust,5,5]
- [NP->.NP PP,5,5]

Juan vio un hombre con un telescopio



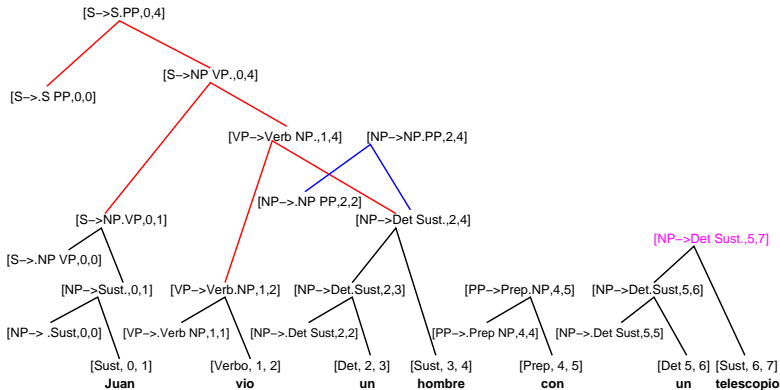
agenda

[NP->.Det Sust,5,5]

[NP->.NP PP,5,5]

[NP->Det.Sust,5,6]

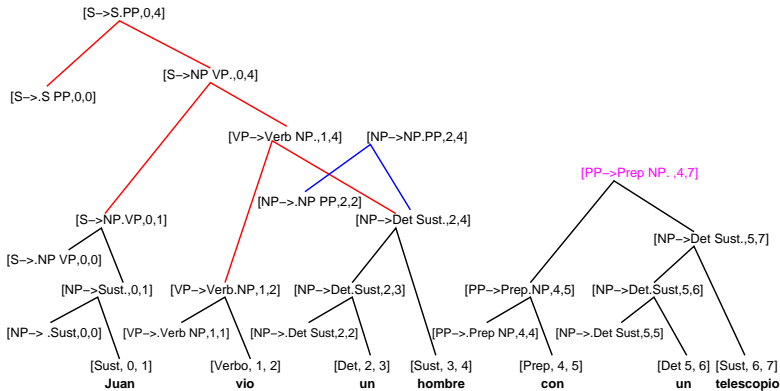
Juan vio un hombre con un telescopio



agenda

- [NP->Det.Sust,5,6]
- [NP->Det.Sust.,5,7]

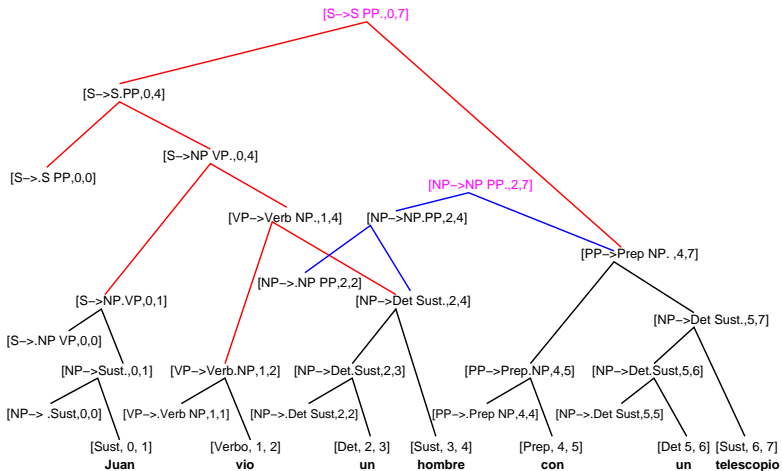
Juan vio un hombre con un telescopio



agenda

- [NP->Det Sust.,5,7]
- [PP->Prep NP.,4,7]

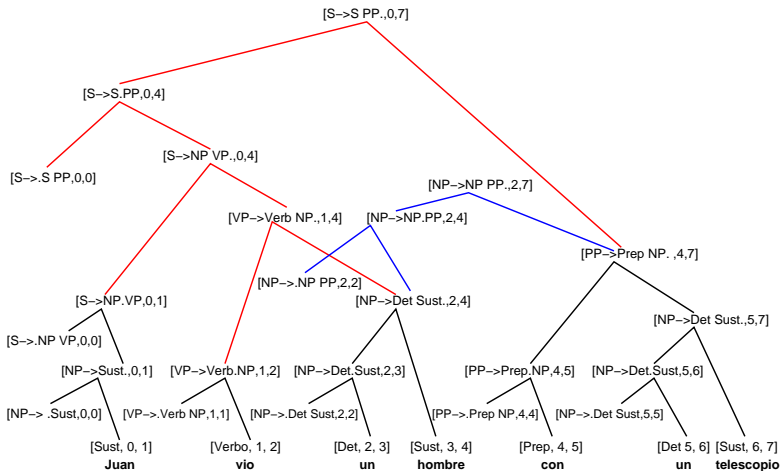
Juan vio un hombre con un telescopio



agenda

- [PP->Prep NP, 4, 7]
- [S->S PP, 0, 7]
- [NP->NP PP, 2, 7]

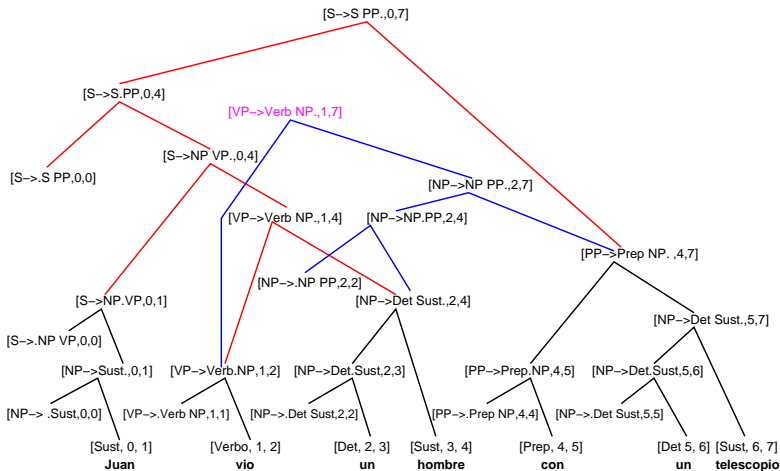
Juan vio un hombre con un telescopio



agenda

 $[S \rightarrow S PP., 0, 7]$ $[NP \rightarrow NP PP., 2, 7]$

Juan vio un hombre con un telescopio

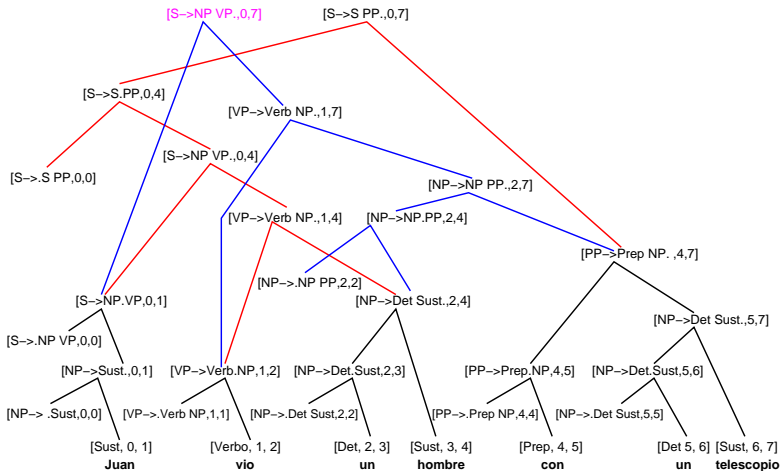


agenda

[NP->NP PP.,2,7]

[VP->Verb NP.,1,7]

Juan vio un hombre con un telescopio

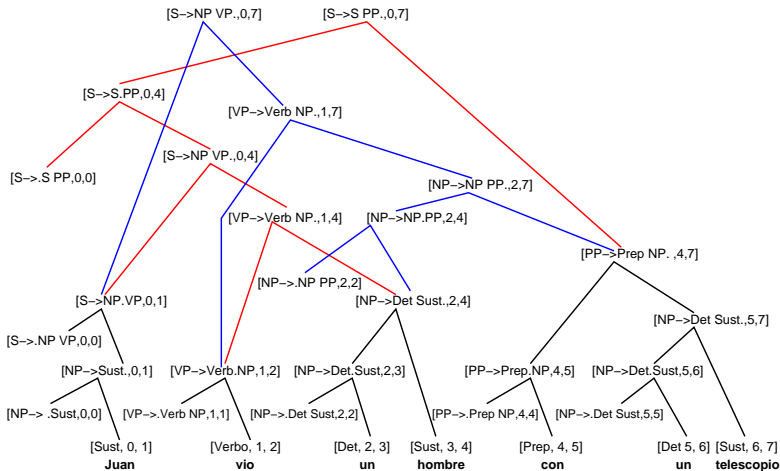


agenda

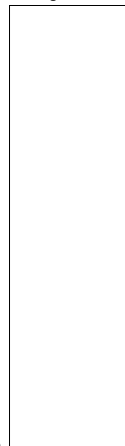
$[VP \rightarrow Verb NP, 1, 7]$

$[S \rightarrow NP VP, 0, 7]$

Juan vio un hombre con un telescopio



agenda



Índice

- 1 Introducción
 - Análisis basado en constituyentes
 - El problema de la ambigüedad
 - Análisis basado en dependencias
- 2 Esquemas de análisis sintáctico
- 3 El algoritmo de Earley
 - Hipótesis
 - Ítems
 - Pasos deductivos
 - Ítems finales
 - Ejemplo
- 4 Análisis sintáctico probabilístico

Gramática independiente del contexto estocástica

- Cada producción tiene asociada una probabilidad, y este conjunto de probabilidades verifica:

$$\sum_{\alpha} P(A \rightarrow \alpha) = 1, \quad \forall A \in N.$$

- Cuando escribimos $P(A \rightarrow \alpha)$, lo que realmente queremos decir es $P(A \rightarrow \alpha|A)$. Por tanto, para cada símbolo no terminal A , la gramática proporciona la distribución de probabilidad de todas sus posibles transformaciones α .

Probabilidad de una frase

- Probabilidad de una frase s de acuerdo con una gramática G

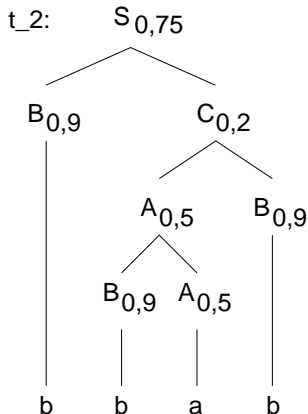
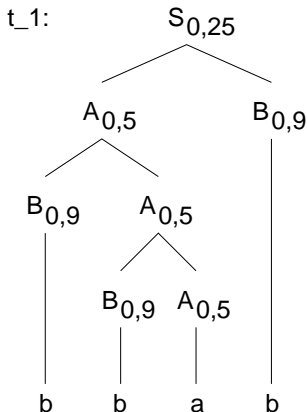
$$P(s) = \sum_t P(s, t) = \sum_{t: \text{frontera}(t)=s} P(t),$$

donde la variable t recorre el espacio de todos los posibles árboles cuya frontera coincide con la frase s .

- La probabilidad de un nodo cualquiera de un árbol t es el producto de las probabilidades de sus subárboles locales y de la probabilidad de la regla de producción de G que los une.
- La probabilidad de un árbol t viene dada, por tanto, por la probabilidad de su nodo raíz.

Ejemplo

$S \rightarrow A B$ (0,25), $A \rightarrow B A$ (0,5), $B \rightarrow C C$ (0,1), $C \rightarrow A B$ (0,2),
 $S \rightarrow B C$ (0,75), $A \rightarrow a$ (0,5), $B \rightarrow b$ (0,9), $C \rightarrow a$ (0,8),



$$P(t_1) = 0,9 \times 0,9 \times 0,5 \times 0,5 \times 0,5 \times 0,9 \times 0,25 = 0,02278$$

$$P(t_2) = 0,9 \times 0,9 \times 0,5 \times 0,5 \times 0,9 \times 0,2 \times 0,75 = 0,02733$$

$$P(s) = P(t_1) + P(t_2) = 0,02733 + 0,02278 = 0,05011$$

Ventajas y limitaciones

- En caso de ambigüedad, da una medida de la plausibilidad de cada árbol
- Precisan un corpus anotado sintácticamente para extraer las probabilidades (entrenar el modelo)
- No es imparcial: se favorecen los árboles “bajitos”
- No tienen en cuenta el contexto en el que se aplican las reglas. En algunos casos, los modelos de n-gramas modelan mejor el lenguaje

Consistencia

- Se trata de obtener modelos probabilísticos del lenguaje
- Debería cumplirse que

$$\sum_{s \in L(G)} P(s) = \sum_t P(t) = 1.$$

pero sólo se cumple si la masa de probabilidad de las reglas se acumula en un número finito de árboles

Ejemplo de gramática inconsistente

$$S \rightarrow a \left(\frac{1}{3}\right) \quad S \rightarrow S S \left(\frac{2}{3}\right)$$

genera frases de la forma a, aa, aaa, \dots

$$P(a) = \frac{1}{3}$$

$$P(aa) = \frac{2}{3} \times \frac{1}{3} \times \frac{1}{3} = \frac{2}{27}$$

$$P(aaa) = \left(\frac{2}{3}\right)^2 \times \left(\frac{1}{3}\right)^3 \times 2 = \frac{8}{243}$$

$$\vdots$$

$$P(L(G)) = \frac{1}{3} + \frac{2}{27} + \frac{8}{243} + \dots \text{ que tiende a } \frac{1}{2}$$

¡¡La mitad de la masa de probabilidad ha desaparecido en el conjunto infinito de árboles que no generan frases de este lenguaje!!

Afortunadamente la consistencia no es importante para comparar las probabilidades de dos árboles de análisis

Análisis sintáctico estocástico

$$I = \{[A \rightarrow \alpha \bullet \beta, i, j, p] \mid A \rightarrow \alpha \beta \in P \wedge 0 \leq i < j \wedge S \xrightarrow{*} A \wedge \alpha \xrightarrow{*} a_{i+1} \dots a_j\}$$

$$\text{INITTER: } \frac{}{[S \rightarrow \bullet \alpha, 0, 0, P(S \rightarrow \alpha)]}$$

$$\text{PREDICTOR: } \frac{[A \rightarrow \alpha \bullet B \beta, i, j, p]}{[B \rightarrow \bullet \gamma, j, j, P(B \rightarrow \gamma)]}$$

$$\text{COMPLETER: } \frac{[A \rightarrow \alpha \bullet B \beta, i, j, p] \quad [B \rightarrow \gamma \bullet, j, k, q]}{[A \rightarrow \alpha B \bullet \beta, i, k, p * q]}$$

$$\text{SCANNER: } \frac{[A \rightarrow \alpha a \beta, i, j, p] \quad [a, j, j + 1]}{[A \rightarrow \alpha a \bullet \beta, i, j + 1, p]}$$

Máquina deductiva estocástica

Obtener todos los análisis posibles requiere del uso de estructuras algebraicas complejas para manejar árboles cíclicos con infinitas probabilidades (si no hay ciclos en la gramática, el problema es sólo de eficiencia)

Lo más habitual es querer obtener sólo el árbol más probable.

Para ello, cuando se va a meter un ítem I en la tabla, se busca sin tener en cuenta las probabilidades:

- Si no se encuentra un ítem equivalente, se inserta I
- Si se encuentra un ítem equivalente J y $P(J) \geq P(I)$, no se hace nada (I no se mete en la tabla y tampoco en la agenda)
- Si se encuentra un ítem equivalente J y $P(J) < P(I)$, se inserta I en la tabla y en la agenda. Por cuestión de rendimiento es conveniente eliminar J de la tabla y de la agenda.