Parsing Chart Extensions 1

- Chart extensions
- Dynamic programming methods
  - CKY
  - Earley

Parsing Chart Extensions 2

- Constraint propagation
  - CSP (Quesada)
- Bidirectional chart parsing
  - Head driven
  - functors driven
  - Island driven
- Using Charts for unification-based formalisms
  - Restrictors
  - Backbone grammars

Parsing Chart Extensions 3

Constraint propagation

- CSP J.F Quesada
- Grammar ambiguity (external) vs parser ambiguity (internal)
- 2 strategies:
  - Basic mechanism bidirectional BU
  - TD predictions
    - Partial derivations parciales
    - adjacencies
- Horizontal mechanism for constraint propagation

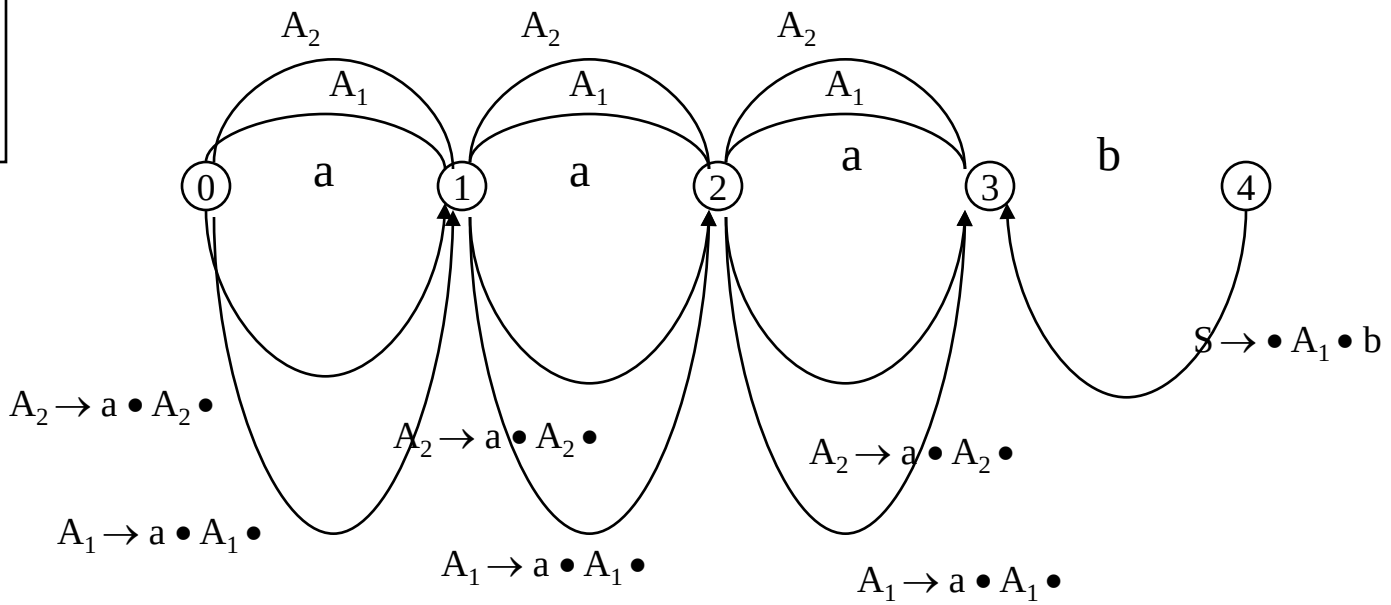
Parsing Chart Extensions 4

$S \rightarrow A_1\ b$
$S \rightarrow A_2\ c$
$A_1 \rightarrow a$
$A_1 \rightarrow a\ A_1$
$A_2 \rightarrow a$
$A_2 \rightarrow a\ A_2$

grammar

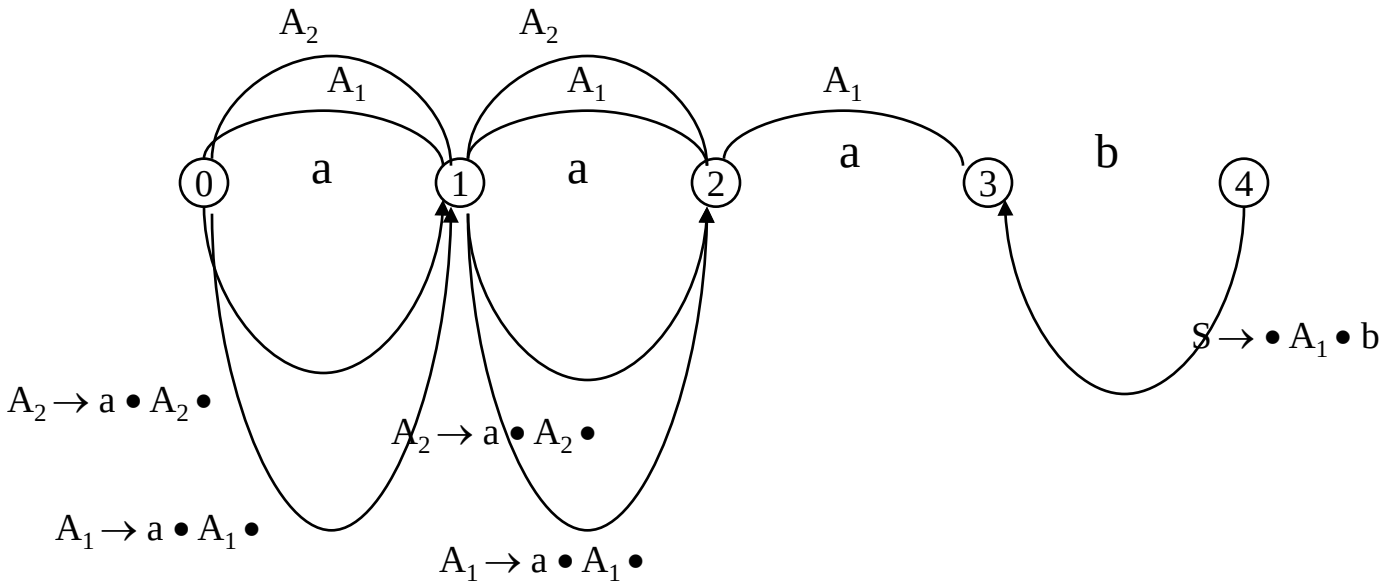
a a a b

sentence



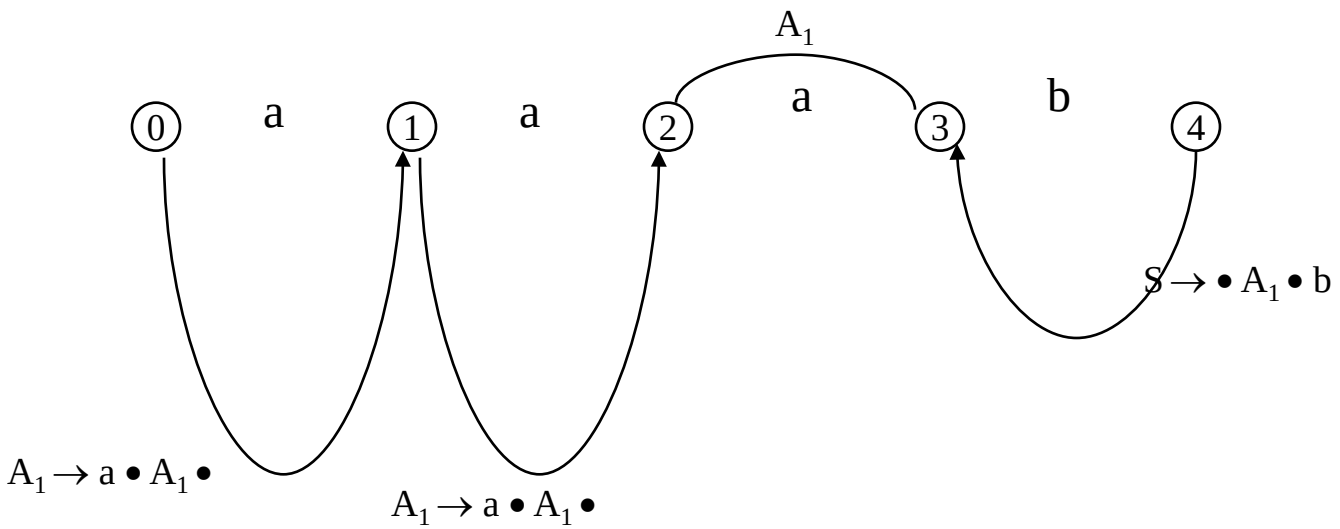
Parsing Chart Extensions 5

Constraints in position 3



Parsing Chart Extensions 6

propagation to positions 2 and 1



Parsing Chart Extensions 7

Head driven chart parsing

- Start rule application from the most informative/restrictive component
  - Linguístic motivation : HG, HPSG
  - Automatic acquisition
  - Generalization of left corner (LC) parsing
  - Generalización of left to write strategies
  - Extended Head Grammar

Parsing Chart Extensions 8

Island driven chart parsing

- Parsing starts BU from some dynamically determined positions of the input string (islands)
- How select them
  - Non ambiguous words
  - base NPs
  - Segments detected with high precision (speech)
- How to extend them
  - local approach
  - neighbouring approach

Tabular methods 1

- Dynamic programming
- CFG
- CKI (Cocke, Kasami, Younger,1967)
  - Grammar in CNF
- Earley 1969
- Extensible to unification, probabilistic, etc...

Tabular methods 2

$G = \langle N, \Sigma, P, S \rangle$, $G \in CNF$, $w = a_1 \ldots a_n$

CKY

$\langle X, H, D \rangle$

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

$H = \{[A, j-1, j] \mid A \rightarrow a_j \in P_G \wedge 1 \leq j \leq n\}$

$D = \{[B, i, j], [C, j, k] \Rightarrow [A, i, k] \mid A \rightarrow BC \in P_G \wedge 0 \leq i < k < j\}$

$V(D) = \{[A, i, j] \mid A \Rightarrow^* a_{i+1} \ldots a_j\}$

domain, set of items

set of de hypothesis

set of valid entities

set of deductive steps

Tabular methods 3

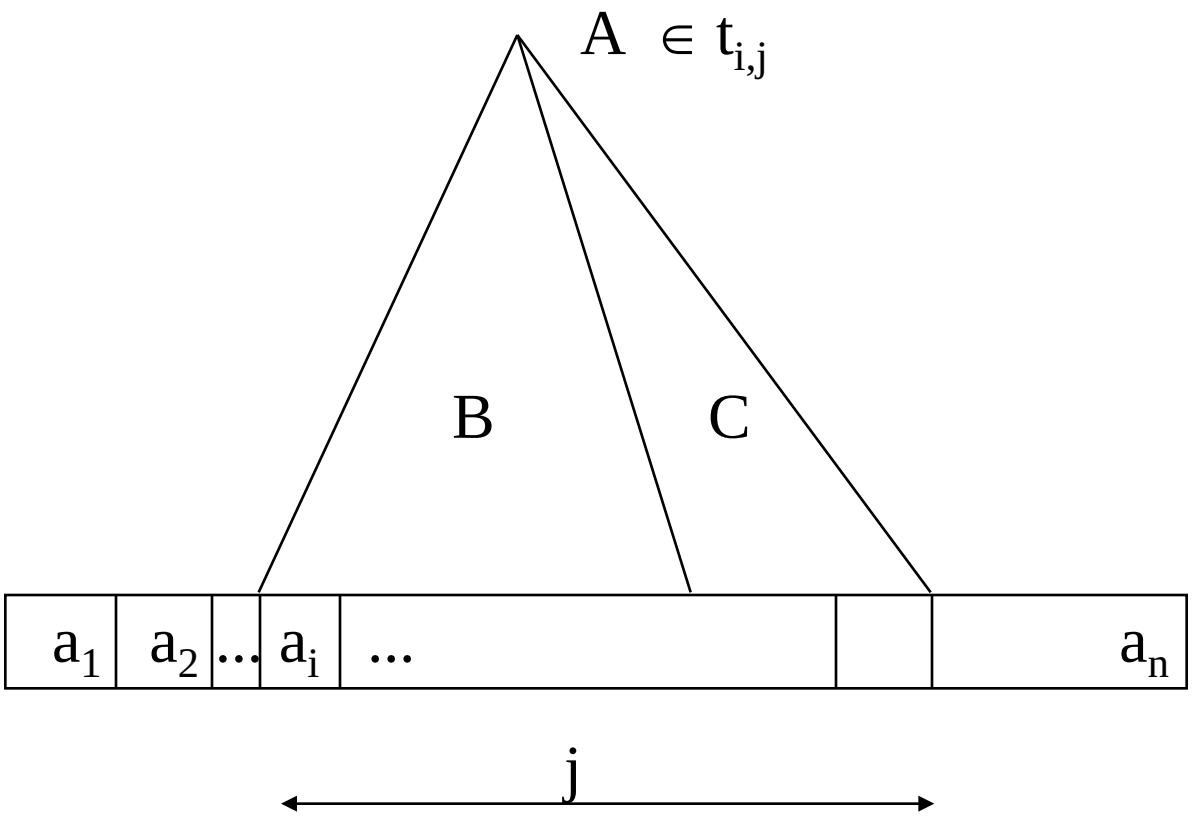
CKY

spatial cost $O(n^{2)}$
temporal cost $O(n^{3)}$
CNF
Dynamically build the parsing table

t_{ij} $\quad 1 \leq i \leq |w|, 1 \leq j \leq |w| - i + 1$

where $w = a_1, \ldots a_n$ is the input string

Tabular methods 4



Where A -> BC is a binary production of the grammar

Tabular methods 5

That A is in cell t_{ij} means that from A the text fragment a_i, ... a_{i+j-1} (string of length j starting in i-esim position) can be derived.

The grammaticality condition is that the initial symbol of the grammar (F) satisfies $f \in t_{1|w|}$

Tabular methods 6

The table is built BU
First row 1 is built using only the unary rules of the grammar:

row j=1
$t_{i1} = \{A | [A \text{ --> } a_i] \in P\}$

Next rows j=2,... are built. The key of the algorithm is that when row j is built all the previous ones (from 1 to j-1) are already built:

row j > 1
$t_{ij} = \{A | \exists k, 1 \leq k < j, [A \text{ --> } BC] \in P, B \in t_{ik}, C \in t_{i+k,j-k}\}$

Tabular methods 7

1. Add the lexical edges
2. for w = 2 to N:
   for i = 0 to N-w:
   for k = 1 to w-1:
   if:
   - A→BC ***and***
   - B→α ∈ chart[i,i+k] ***and***
   - C→β ∈ chart[i+k,i+w]

   then:
   - add A→BC to chart[i,i+w]
3. If S∈chart[0,N], return the corresponding parse

taken from Loper

Tabular methods 8

sentence	$\rightarrow$ NP, VP
NP	$\rightarrow$ A, B
VP	$\rightarrow$ C, NP
A	$\rightarrow$ det
B	$\rightarrow$ n
NP	$\rightarrow$ n
VP	$\rightarrow$ vi
C	$\rightarrow$ vt

Tabular methods 9

the cat eats fish sentence			
the cat eats sentence	*cat eats fish* sentence		
the cat NP	*cat eats* sentence	*eats fish* VP	
the (det) A	*cat (n)* B, NP	*eats (vt, vi)* C, VP	*fish (n)* B, NP

Tabular methods 10

Earley

$P_{earley} = \langle I, H, D \rangle$

$I = \{[A \rightarrow \alpha \bullet \beta, i, j] \mid A \rightarrow \alpha\beta \in P_G \wedge 0 \leq i \leq j\}$

$D = D^{init} \cup D^{scan} \cup D^{compl} \cup D^{pred}$

$H = \{ [a_1, 0, 1], \ldots, [a_n, n-1, n]\}$

Initialization

$D^{init} = \{\Rightarrow [S \rightarrow \bullet\gamma, 0, 0]\}$

scanning

$D^{scan} = \{[A \rightarrow \alpha \bullet a\beta, i, j], [a, j, j+1] \Rightarrow [A \rightarrow \alpha a \bullet \beta, i, j+1]\}$

completion

$D^{compl} = \{[A \rightarrow \alpha \bullet B\beta, i, j], [B \rightarrow \gamma \bullet, j, k] \Rightarrow [A \rightarrow \alpha B \bullet \beta, i, k]\}$

prediction

$D^{pred} = \{[A \rightarrow \alpha \bullet B\beta, i, j] \Rightarrow [B \rightarrow \bullet\gamma, j, j]\}$

$V = \{[A \rightarrow \alpha \bullet \beta, i, j] \mid \alpha \Rightarrow^* a_{i+1} \ldots a_j \wedge S \Rightarrow^* a_1 \ldots a_i A \gamma\}$

$F = \{[S \rightarrow \gamma \bullet, 0, n]\}$

$C = \{[S \rightarrow \gamma \bullet, 0, n] \mid \gamma \Rightarrow^* a_1 \ldots a_n\}$

Tabular methods 11

spatial cost $O(n^2)$
temporal cost $O(n^3)$, $O(n^2)$ for non ambiguous grammars
CNF not needed
TD strategy

S	$\rightarrow$	NP, VP.
NP	$\rightarrow$	*N.
NP	$\rightarrow$	*DET, *N.
VP	$\rightarrow$	*V, NP.

Tabular methods 12

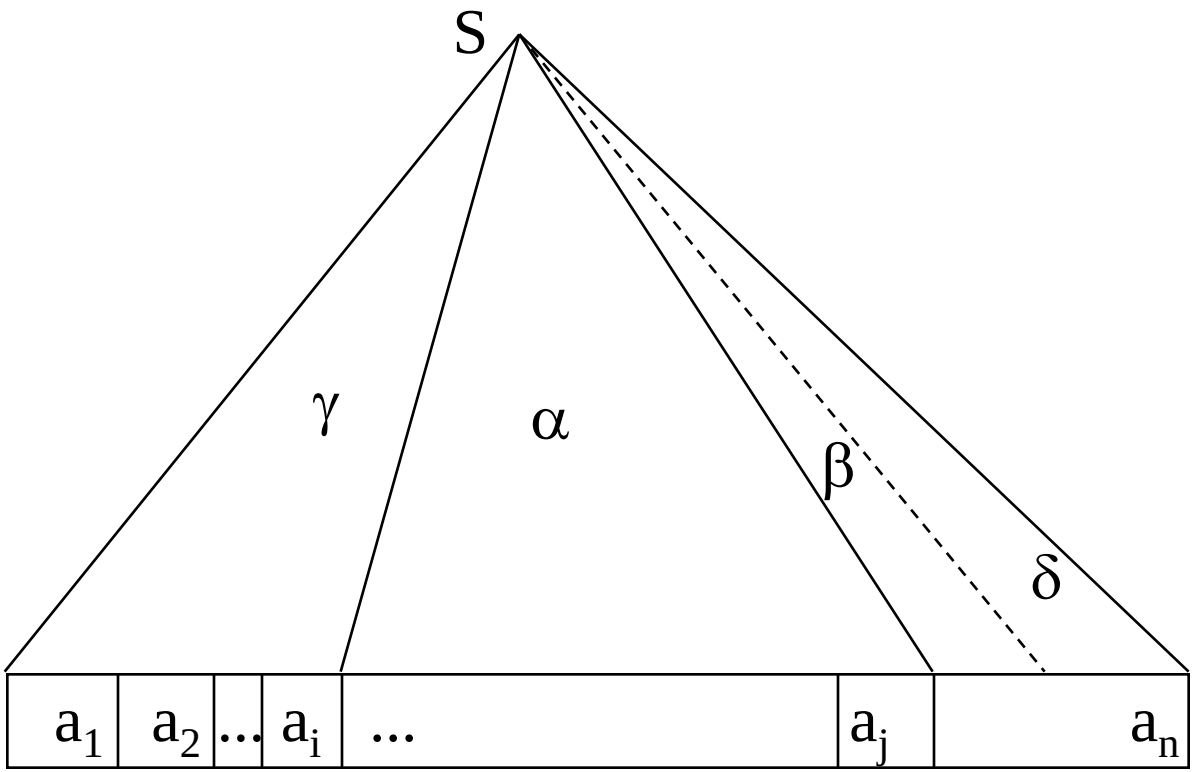
Each state corresponds to consuming one word from the input string: $I_0, I_1, \ldots I_i, \ldots I_{|w|}$

Each state contains sets of "parse list" indicating the process of application of a rule (as a dotted rule) for reaching the state

$[A \dashrightarrow \alpha \bullet \beta, i] \in I_j$ means that $S \dashrightarrow \gamma A \delta$ where
$\gamma \dashrightarrow a_1 \ a_2 \ \ldots a_i \qquad \alpha \dashrightarrow a_{i+1} \ a_{i+2} \ \ldots a_j$

Gramaticality condition is $[S \dashrightarrow \alpha \bullet, 0] \in I_{|w|}$

Tabular methods 13



S --> γ A δ

[A --> α • β, i] ∈ I_j

Tabular methods 14

Iterative process:

- Building I_0
- Building (extension) the rest of states

operations:

- TD prediction
- scanning
- completion (end of a rule)

Tabular methods

Building I_0:

1) if [S--> α] $\in$ P then [S--> $\bullet$ α, 0] $\in I_0$

2) if [B--> $\gamma \bullet$, 0] $\in I_0$

(only possible if [B--> $\bullet$, 0] $\in I_0$)

and [A--> $\alpha \bullet B\beta$, 0] $\in I_0$ then [A--> $\alpha B \bullet \beta$, 0] $\in I_0$

3) if [A--> $\alpha \bullet B\beta$, 0] $\in I_0$ and [B--> γ] $\in$ P

then [B--> $\bullet$ γ, 0] $\in I_0$

Tabular methods 16

Extension:

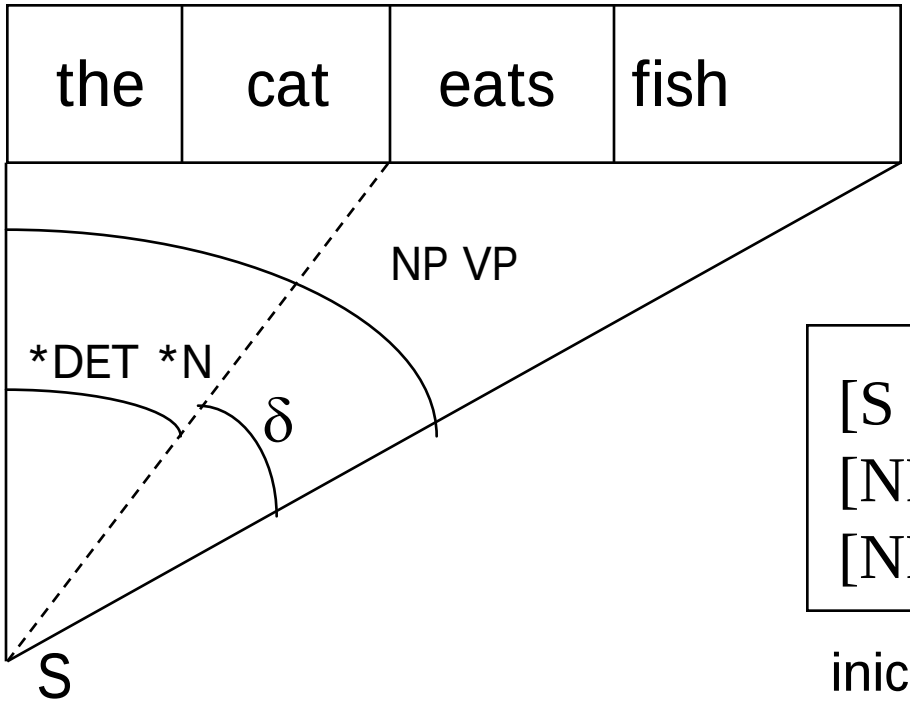
4) if [B--> $\alpha \bullet a\beta$, i] $\in I_{j-1}$ y $a = a_j$ then [B--> $\alpha a \bullet \beta$, i] $\in I_j$
(**scanning**)

5) if [A--> $\gamma \bullet$, i] $\in I_j$ and [B--> $\alpha \bullet A\beta$, k] $\in I_i$ then
[B--> $\alpha A \bullet \beta$, k] $\in I_j$
(**completion**)

6) if [A--> $\alpha \bullet B\beta$, i] $\in I_j$ and [B--> γ] $\in P$
then [B--> $\bullet \gamma$, j] $\in I_j$
(**prediction**)

Tabular methods 17

example:



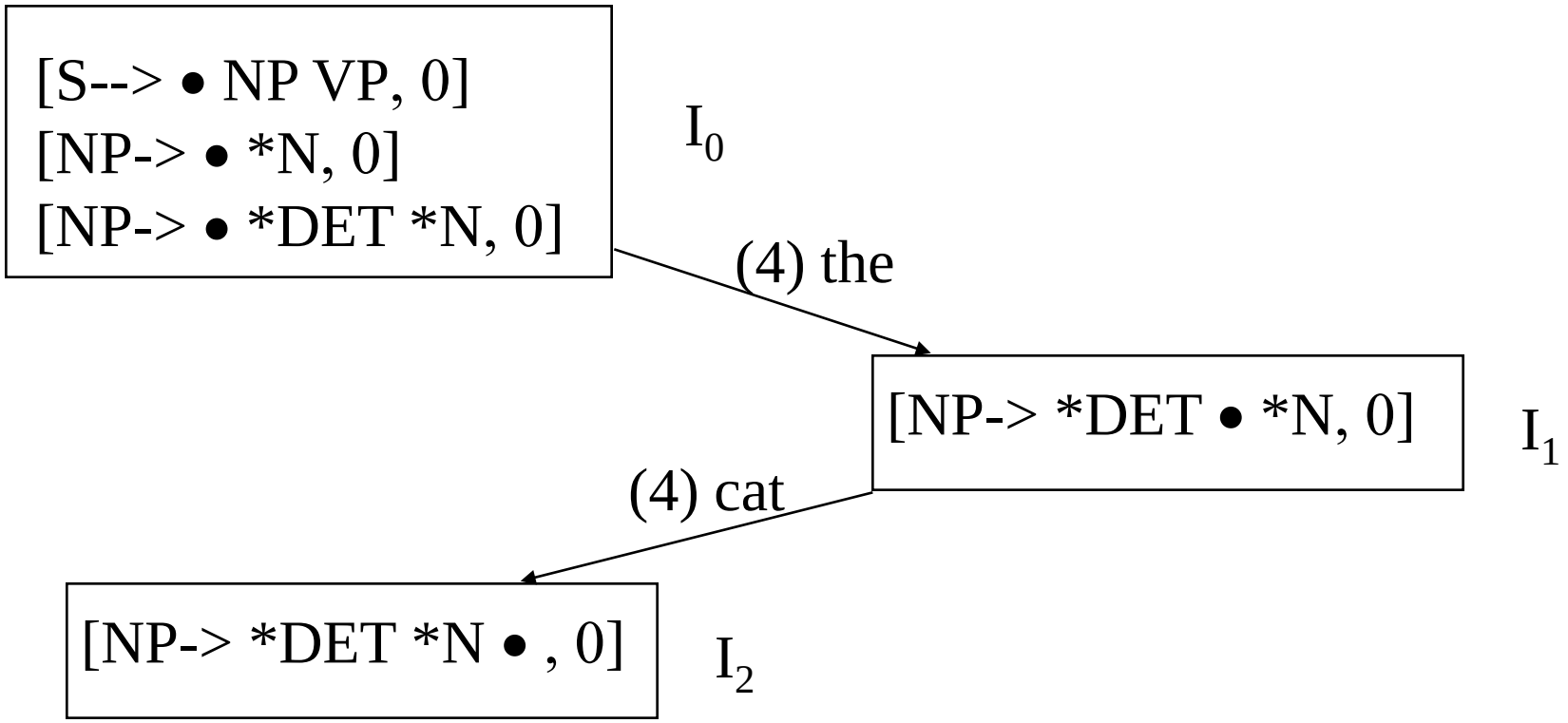
$[S \rightarrow \bullet\ NP\ VP, 0] \in I_0$
$[NP \rightarrow \bullet\ *N, 0] \in I_0$
$[NP \rightarrow \bullet\ *DET\ *N, 0] \in I_0$

inicialization

prediction

prediction

Tabular methods 18



Tabular methods 19