ΓΛΩΣΣΙΚΗ ΤΕΧΝΟΛΟΓΙΑ

ΣΥΝΤΑΞΗ: ΟΡΘΟΛΟΓΙΚΗ ΠΡΟΣΕΓΓΙΣΗ (FORMAL SYNTAX)

Χρηματοδότηση

Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.

Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Ιόνιο Πανεπιστήμιο**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.

Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

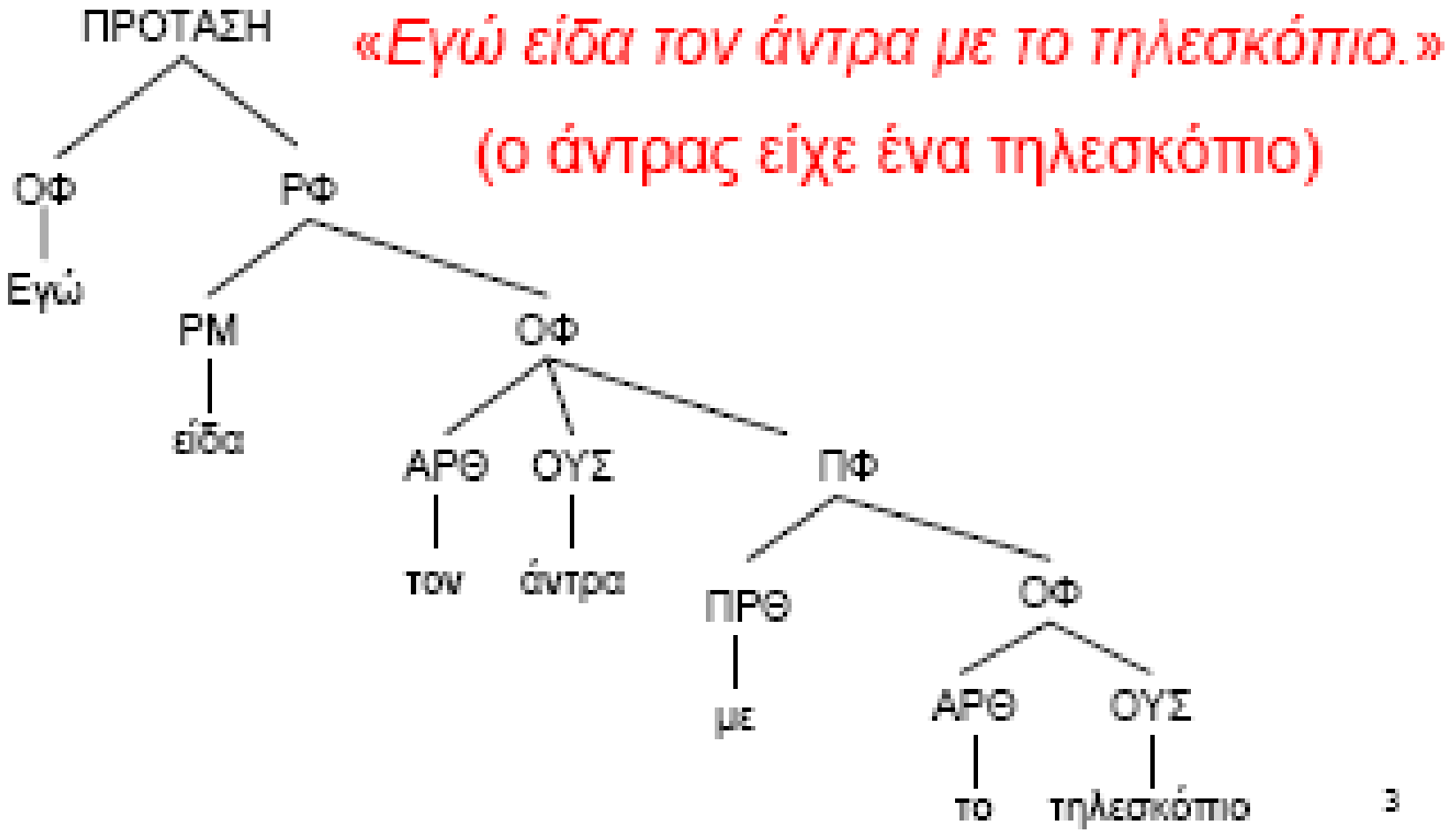
Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons

Τι είναι η Συντακτική Ανάλυση;

- Μία πρόταση φυσικής γλώσσας μετατρέπεται σε μία ιεραρχημένη δομή που ανταποκρίνεται στη διασύνδεση των δομικών στοιχείων της πρότασης
- Η όλη διαδικασία καλείται ανάλυση (parsing)
- Για μία πρόταση η ανάλυση μπορεί να δίνει πολλαπλά πιθανά αποτελέσματα (parses)
- Η πιο απλή μορφή δομής είναι ένα συντακτικό δέντρο (parse tree)
- Η έξοδός της οδηγείται στη σημασιολογική ανάλυση

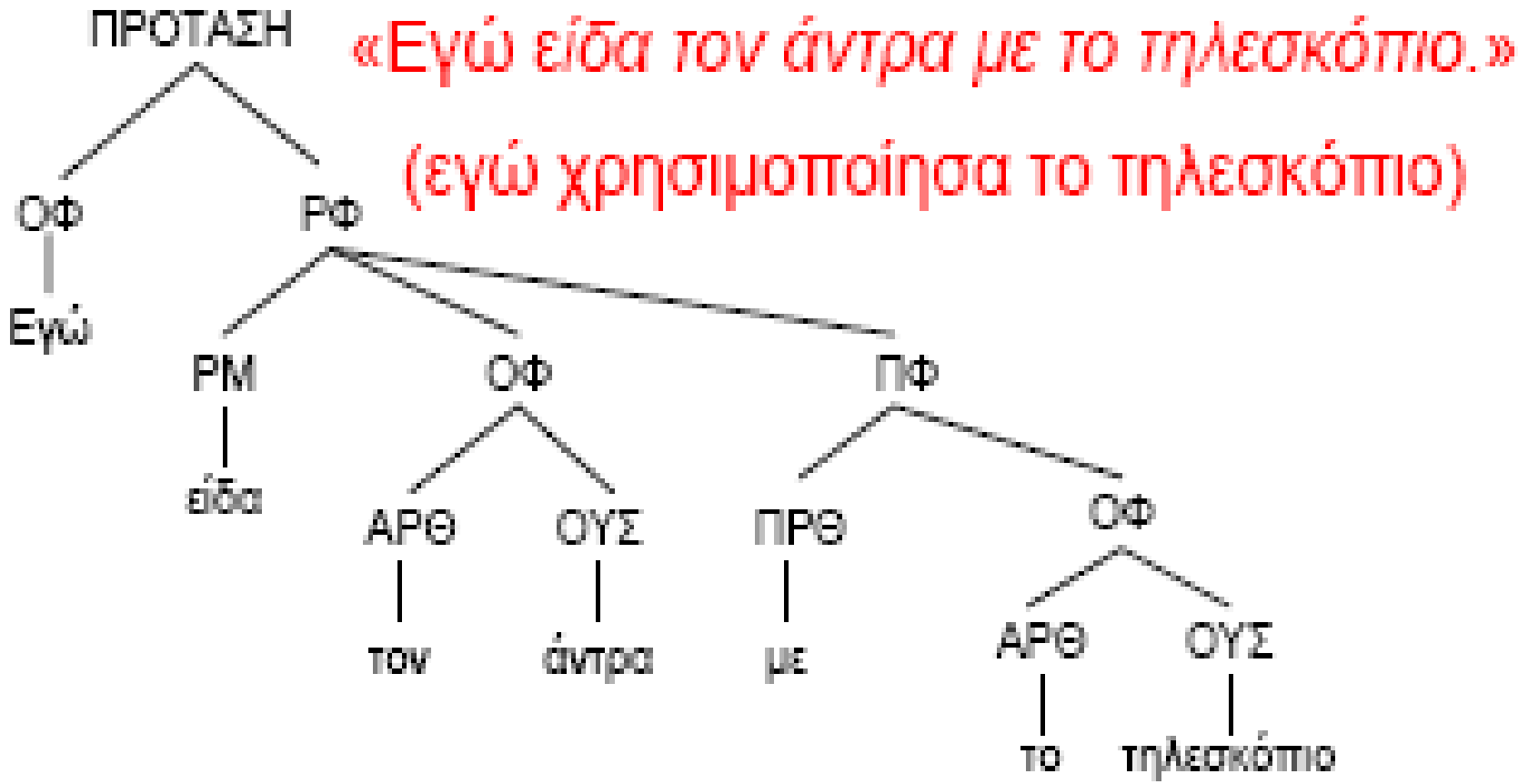
Παράδειγμα: Parse 1



«Εγώ είδα τον άντρα με το τηλεσκόπιο.»

(ο άντρας είχε ένα τηλεσκόπιο)

Παράδειγμα: Parse 2



Οι πολλαπλές αναλύσεις

- Η απόδοση πολλών πιθανών συντακτικών αναλύσεων σε μία πρόταση αποτελεί μεγάλο πρόβλημα
  - "*List the sales of the products produced in 1973 with the products produced in 1972.*"
  - 455 parses (Martin et al., 1987)
- Η συντακτική ανάλυση συχνά παραλείπεται ή ενοποιείται με την σημασιολογική
- Μερική ανάλυση (partial parsing)

Συντακτική Ανάλυση

- Σχεδόν όλα τα υπάρχοντα συστήματα έχουν δύο συστατικά
  - Γραμματική (grammar): ρητή αναπαράσταση των συντακτικών κανόνων της γλώσσας
  - Αναλυτής (parser): συγκρίνει τη γραμματική με τις προτάσεις εισόδου και παράγει συντακτικές δομές

Μια απλή Γραμματική

S → NP VP
NP → the NP1
NP → PRO
NP → PN
NP → NP1
NP1 → ADJS N
ADJS → ∈ | ADJ ADJS
VP → V
VP → V NP
N → file | printer
PN → Bill
PRO → I
ADJ → short | long | fast
V → printed | created | want

- Κανόνες επανεγγραφής (rewrite rules)
- X → Y Z : Το X μπορεί να αντικατασταθεί από τα Y και Z
- X → Y | Z : Το X μπορεί να αντικατασταθεί από το Y ή το Z
- Αρχικό σύμβολο: S
- Τερματικό σύμβολο: συγκεκριμένη λέξη (file, Bill)
- Μη-τερματικό σύμβολο: αναλύεται σε τερματικά (NP, V)
- ∈: το κενό

Μια απλή Γραμματική

S → NP VP
NP → the NP1
NP → PRO
NP → PN
NP → NP1
NP1 → ADJS N
ADJS → ϵ | ADJ ADJS
VP → V
VP → V NP
N → file | printer
PN → Bill
PRO → I
ADJ → short | long | fast
V → printed | created | want

Παραδείγματα προτάσεων:

I printed the long file.

Bill created the printer.

Bill created I.

Bill created Bill.

File want.

Τύποι Γραμματικών: Ιεραρχία του Chomsky (Chomsky Hierarchy)

- **Κανονική γραμματική (Regular Grammar) ή τύπου 3:** στα αριστερά ένα μη-τερματικό σύμβολο, στα δεξιά ένα τερματικό σύμβολο και πιθανώς ένα μη-τερματικό
  - A → aB ή
  - A → a
- **Γραμματική ελεύθερης σύνταξης (Context-free Grammar) ή τύπου 2 :** στα αριστερά ένα μη-τερματικό σύμβολο, στα δεξιά οτιδήποτε
- **Γραμματική ευαίσθητης σύνταξης (Context-Sensitive Grammar) ή τύπου 1** : κανόνες της μορφής X A Y → X B Y
- **Ελεύθερη Γραμματική (Free Grammar) ή τύπου 0:** κανόνες Οτιδήποτε
- Κάθε γραμματική τύπου 3 είναι και τύπου 2, 1 και 0. Κάθε γραμματική τύπου 2 είναι και τύπου 1 και 0, κ.ο.κ.

Μια καλή Γραμματική

- Μία καλή γραμματική μιας φυσικής γλώσσας πρέπει να:
  - Αναγνωρίζει τις συντακτικά ορθές προτάσεις ("Bill printed the long file.")
  - Απορρίπτει τις συντακτικά λάθος προτάσεις ("Printed long the Bill file.")
  - Αποδίδει λογικές συντακτικές δομές στις προτάσεις
    [(Bill) printed (the long file)] σωστό
    [(Bill) (printed the long) file] λάθος

Γραμματικές Οριστικών Προτάσεων (Definite Clause Grammars -DCGs)

- Αποτελούν επέκταση των Γραμματικών Ελεύθερης Σύνταξης και μπορούν να εκφραστούν εύκολα και να «τρέξουν» σε Prolog.
- Εκφράζονται απευθείας σαν όροι της Prolog (Prolog terms), οι οποίοι είναι της μορφής:

  A → B

  που μεταφράζεται ως: «Μια πιθανή μορφή του A είναι η B»

Παράδειγμα: DCG και PROLOG

```
s --> np, vp.
np --> art, n.
vp --> v, np.
art --> [the].
n --> [cat].
n --> [dog].
v --> [ate].
```

```
?-phrase(s, [the, dog, ate, the,
   cat]).
yes

?-phrase(s, Sentence).
Sentence = [the, cat, ate, the, cat]
Sentence = [the, cat, ate, the, dog]
Sentence = [the, dog, ate, the, cat]
Sentence = [the, dog, ate, the, dog]
yes
```

Επέκταση: Επίθετα

```
np --> art, adjs, n.
adjs --> [].
adjs --> adj, adjs.
art --> [the].
adj --> [tall].
adj --> [dark].
adj --> [handsome].
n --> [man].
```

- Μία ονοματική φράση μπορεί να έχει κανένα ή ένα ή πιο πολλά επίθετα.

```
?-phrase(np, [the, man]).
yes
?-phrase(np, [the, tall,
   dark, handsome, man]).
yes
```

Επέκταση: Μορφολογική Συμφωνία

```
s --> np(N), vp(N).
np(N) --> art, n(N).
vp(N) --> v(N), np(_).
art --> [the].
n(sing) --> [cat].
n(plur) --> [cats].
v(sing) --> [eats].
v(plur) --> [eat].
```

- Επιβάλλουμε τη συμφωνία στον αριθμό όπου χρειάζεται.
- Ορίζουμε τις τιμές του αριθμού για το κάθε τερματικό σύμβολο.

```
?-phrase(s, [the, cats,
  eats]).

no
```

Επέκταση: Αναφορικές Προτάσεις

```
+
np --> art, n, relpn, vp.
relpn --> [that].
relpn --> [who].
```

- "*The man that likes Mary eats an apple.*"
- "*John likes the girl who eats the apple.*"

Άλλες επεκτάσεις

- Μεταβατικά και αμετάβατα ρήματα.
- Πιο σύνθετη μορφολογική συμφωνία (πτώση, γένος).
- Ερωτήσεις και εντολές.
- Ποσοδείκτες (π.χ. all, those).
- Βοηθητικά ρήματα (π.χ. may, have).
- ...
- Καμία γραμματική δεν μπορεί να καλύψει όλες τις πιθανές προτάσεις μιας γλώσσας

Το λεξικό

- Όσο επεκτείνουμε τη γραμματική, χρειάζεται να αποθηκεύσουμε περισσότερη πληροφορία για την κάθε λέξη.

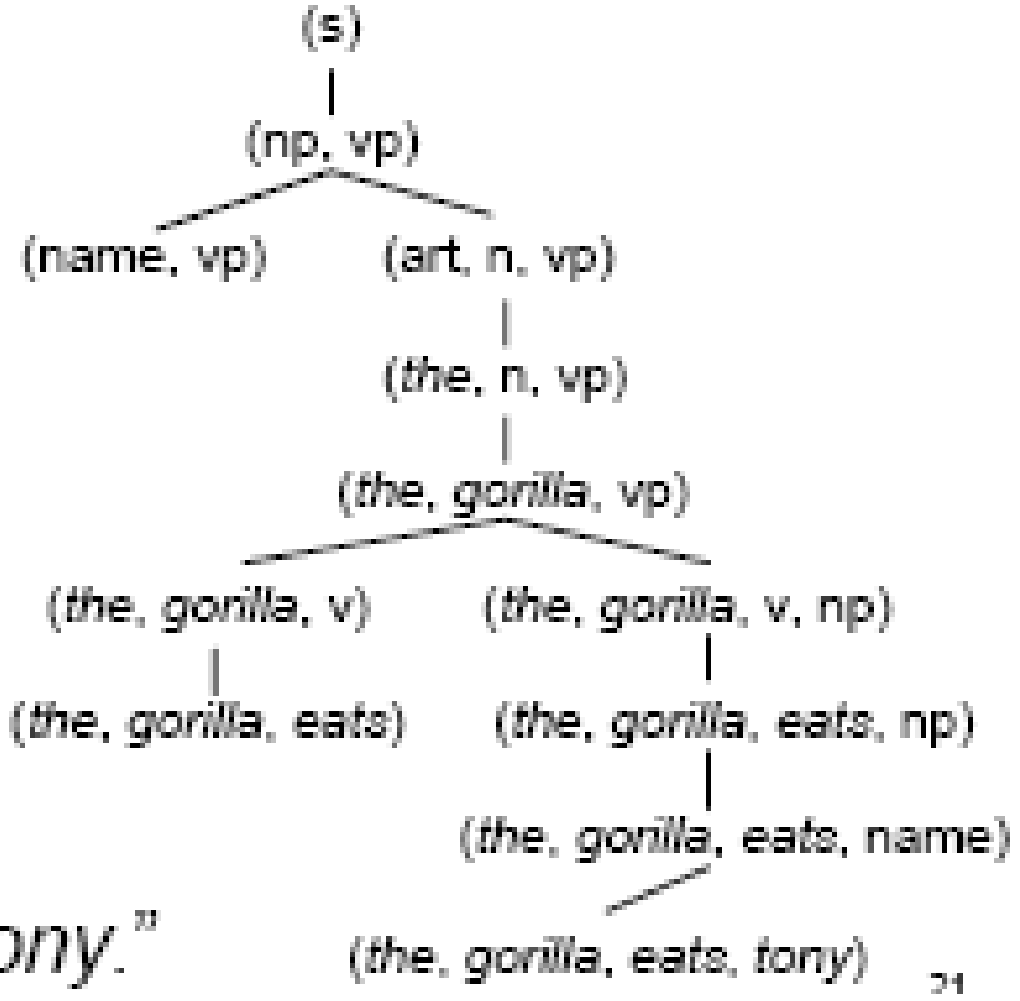
```
verb(transitive, sing) --> [smells].
verb(intransitive, sing) --> [smells].
noun(plural) --> [smells].
```

Είδη Αναλύσεων

- Top-down
  - Ξεκινώντας από ένα αρχικό σύμβολο (S), εφαρμόζουμε τους γραμματικούς κανόνες προς τα εμπρός (ταιριάζοντας το αριστερό μέρος) μέχρι τα τερματικά σημεία του δέντρου να αντιστοιχούν σε λέξεις της πρότασης.
- Bottom-up
  - Ξεκινώντας από την πρόταση εφαρμόζουμε τους γραμματικούς κανόνες αντίστροφα (ταιριάζοντας το δεξί μέρος) μέχρι να φτάσουμε σε ένα αρχικό σύμβολο.

Top-down ανάλυση

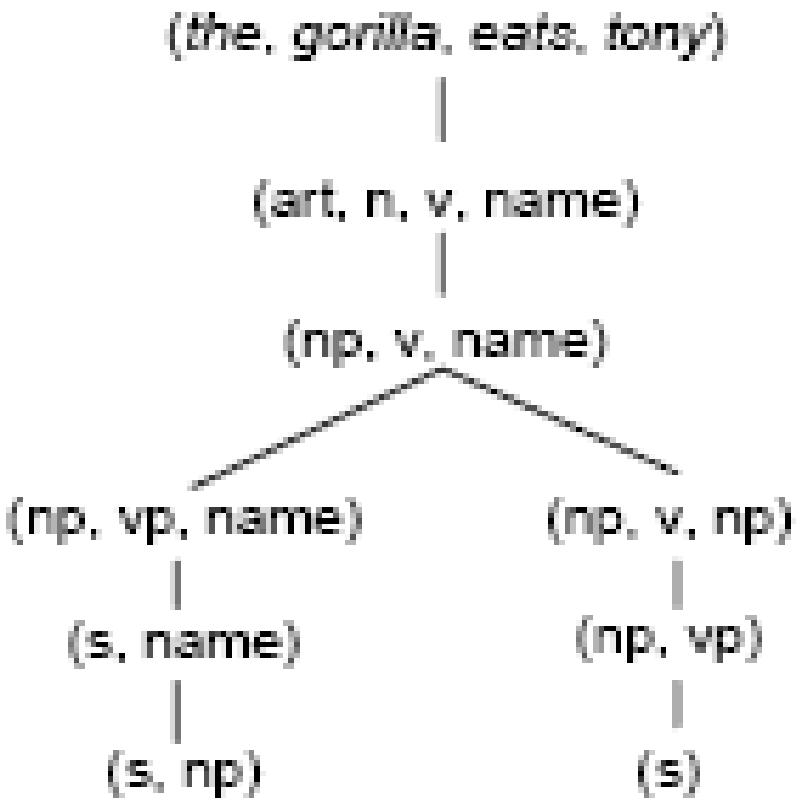
```
s --> np, vp.
np --> name.
np --> art, n.
vp --> v.
vp --> v, np.
art --> [the].
n --> [gorilla].
name --> [tony].
v --> [eats].
```



"*The gorilla eats Tony.*"

Bottom-up Ανάλυση

```
s --> np, vp.
np --> name.
np --> art, n.
vp --> v.
vp --> v, np.
art --> [the].
n --> [gorilla].
name --> [tony].
v --> [eats].
```



"*The gorilla eats Tony.*"

Σύγκριση top-down και bottom-up ανάλυσης

- Η ανάλυση top-down δεν χάνει χρόνο εξερευνώντας δέντρα που δεν οδηγούν σε πρόταση.
- Αντίθετα η bottom-up ανάλυση σπαταλάει πολύ χρόνο στο να φτιάχνει υπο-δέντρα (sub-trees) τα οποία δεν έχουν καμια ελπίδα να οδηγήσουν σε πρόταση.
- Η top-down ανάλυση σπαταλάει όμως πολύ χρόνο στην εξερεύνηση δέντρων που οδηγούν μεν σε πρόταση, αλλά είναι άσχετα με την πρόταση προς ανάλυση.

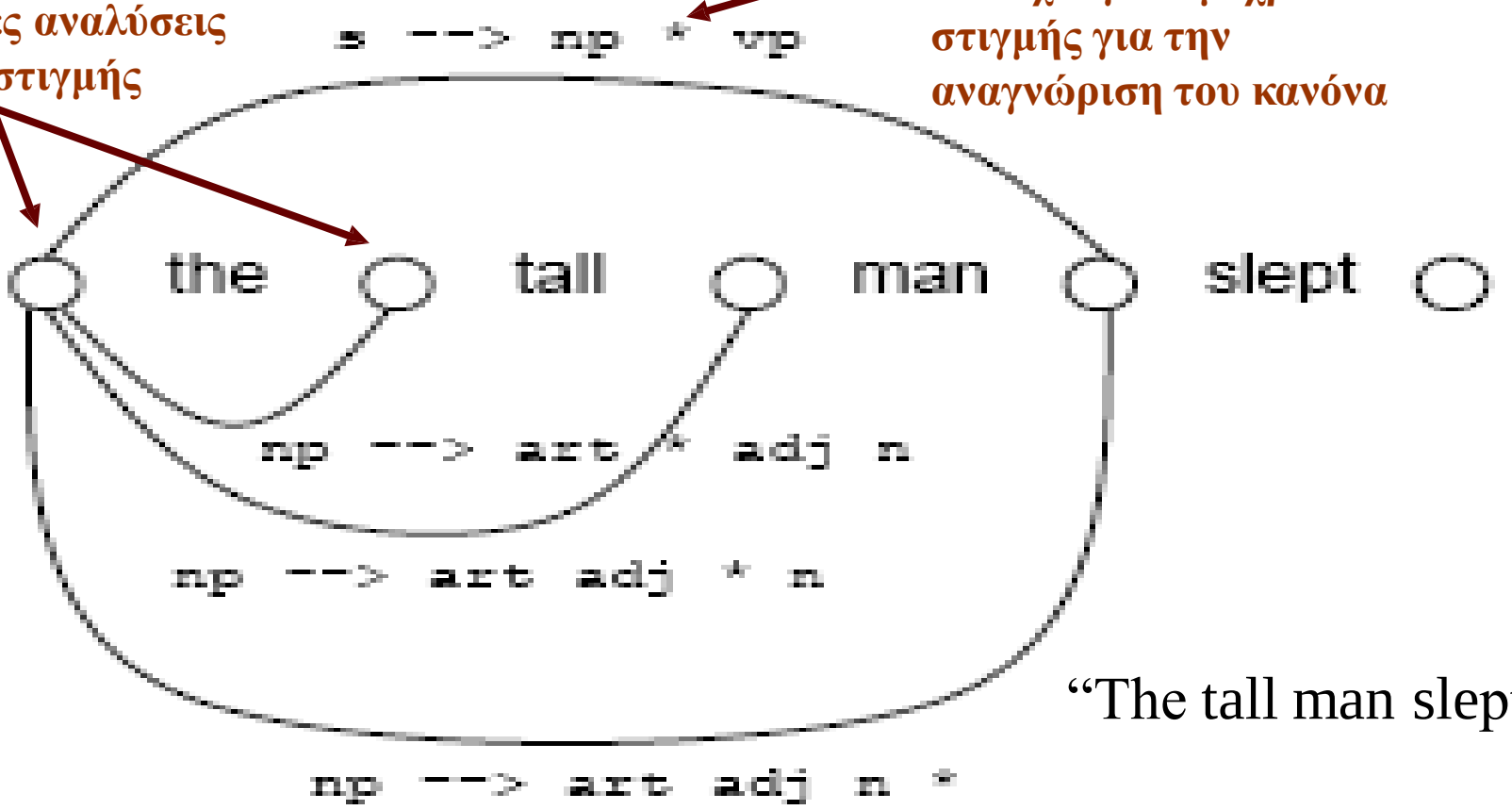
Chart Parsing (Διαγραμματική Ανάλυση - Earley, 1970)

- Αποθηκεύει τα αποτελέσματα ανάλυσης κομματιών της πρότασης για την αποφυγή άσκοπων επαναλήψεων
  - "the tall man" = np
- Χρησιμοποιεί μία δομή δεδομένων που καλείται διάγραμμα (chart)
  - Για N λέξεις το chart έχει N+1 κόμβους
  - Ολοκληρωμένη σύνδεση: τμήμα κειμένου που αναλύθηκε επιτυχώς
  - Ατελής σύνδεση: αρχή νέας ανάλυσης
  - Dotted rules: S → np * vp

Chart Parsing: Παράδειγμα

Καταστάσεις (states) του διαγράμματος: αναπαριστούν τις μερικές αναλύσεις μέχρι στιγμής

Σεσημασμένος κανόνας: το * δείχνει την πρόοδο που έχει γίνει μέχρι στιγμής για την αναγνώριση του κανόνα



"The tall man slept"

Chart Parsing: Άλλο παράδειγμα

S → NP VP
S → AUX NP VP
S → VP
NP → DET N
NP → PN
VP → V
VP → V NP
AUX → [does | did | do]
N → [book | flight | meal | money]
PN → [John | Michael | George]
DET → [that | this | a]
V → [book | include | prefer]

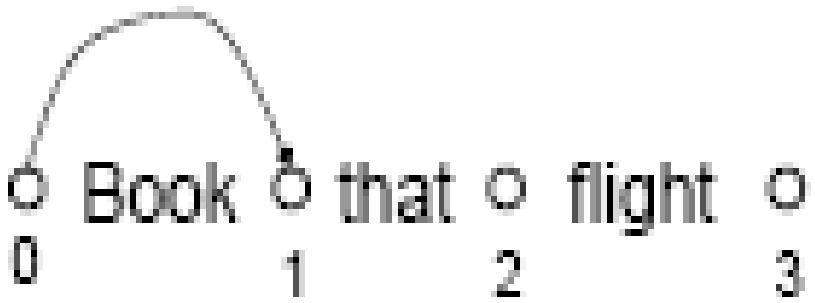
Παράδειγμα: Πρώτο Βήμα

S → * NP VP
S → * AUX NP VP
S → * VP
NP → * DET N
NP → * PN
VP → * V
VP → * V NP
AUX → * [does | did | do]
N → * [book | flight | meal | money]
PN → * [John | Michael | George]
DET → * [that | this | a]
V → * [book | include | prefer]

Παράδειγμα: Δεύτερο Βήμα

S → NP VP
S → AUX NP VP
S → VP
NP → DET N
NP → PN
VP → V
VP → V NP
AUX → [does | did | do]
N → [book | flight | meal | money]
PN → [John | Michael | George]
DET → [that | this | a]
V → [book | include | prefer]

S → VP *	3
VP → V *	2
VP → V * NP	2
N → book *	1
V → book *	1



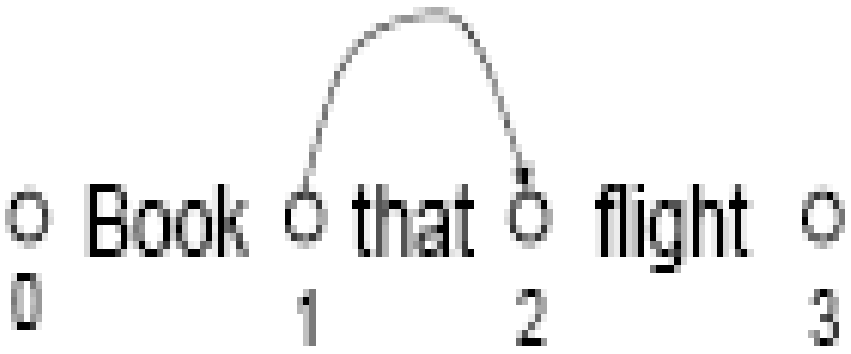
Παράδειγμα: Τρίτο Βήμα

S → NP VP
S → AUX NP VP
S → VP
NP → DET N
NP → PN
VP → V
VP → V NP
AUX → [does | did | do]
N → [book | flight | meal | money]
PN → [John | Michael | George]
DET → [that | this | a]
V → [book | include | prefer]

Από πριν: **VP → V * NP**

Καινούρια: NP → DET * N 2

DET → that * 1

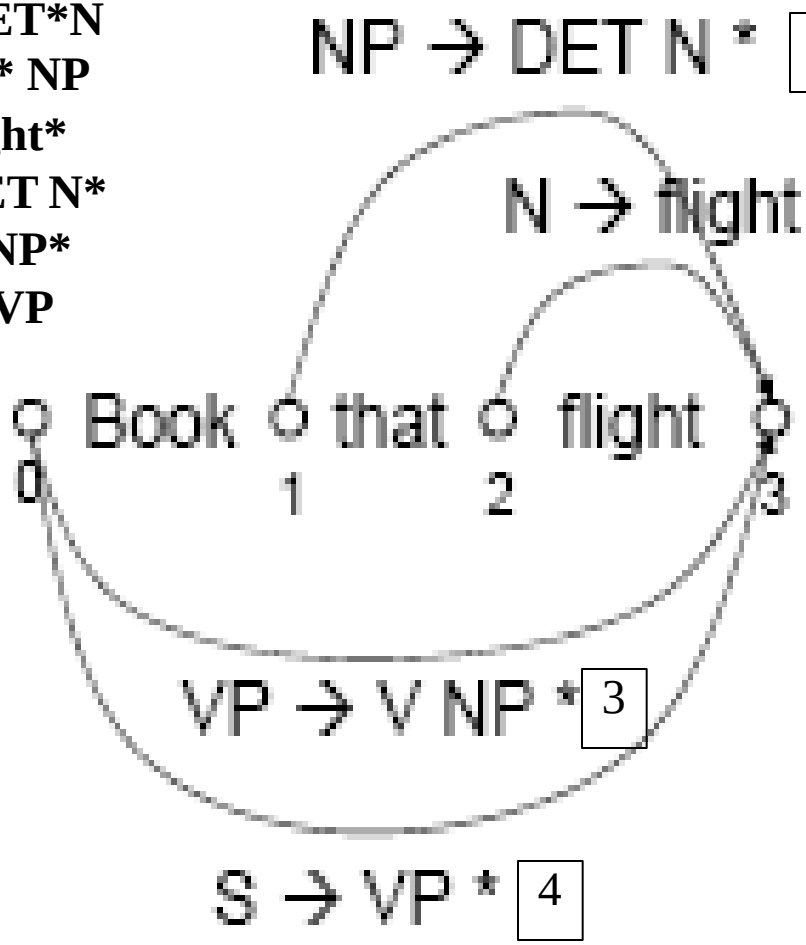


Παράδειγμα: Τέταρτο Βήμα

S → NP VP
S → AUX NP VP
S → VP
NP → DET N
NP → PN
VP → V
VP → V NP
AUX → [does | did | do]
N → [book | flight | meal | money]
PN → [John | Michael | George]
DET → [that | this | a]
V → [book | include | prefer]

Απο πριν: **NP → DET*N**
VP → V * NP

Καινούρια: **N → flight***
NP→DET N*
VP→V NP*
S→NP*VP
S→VP*



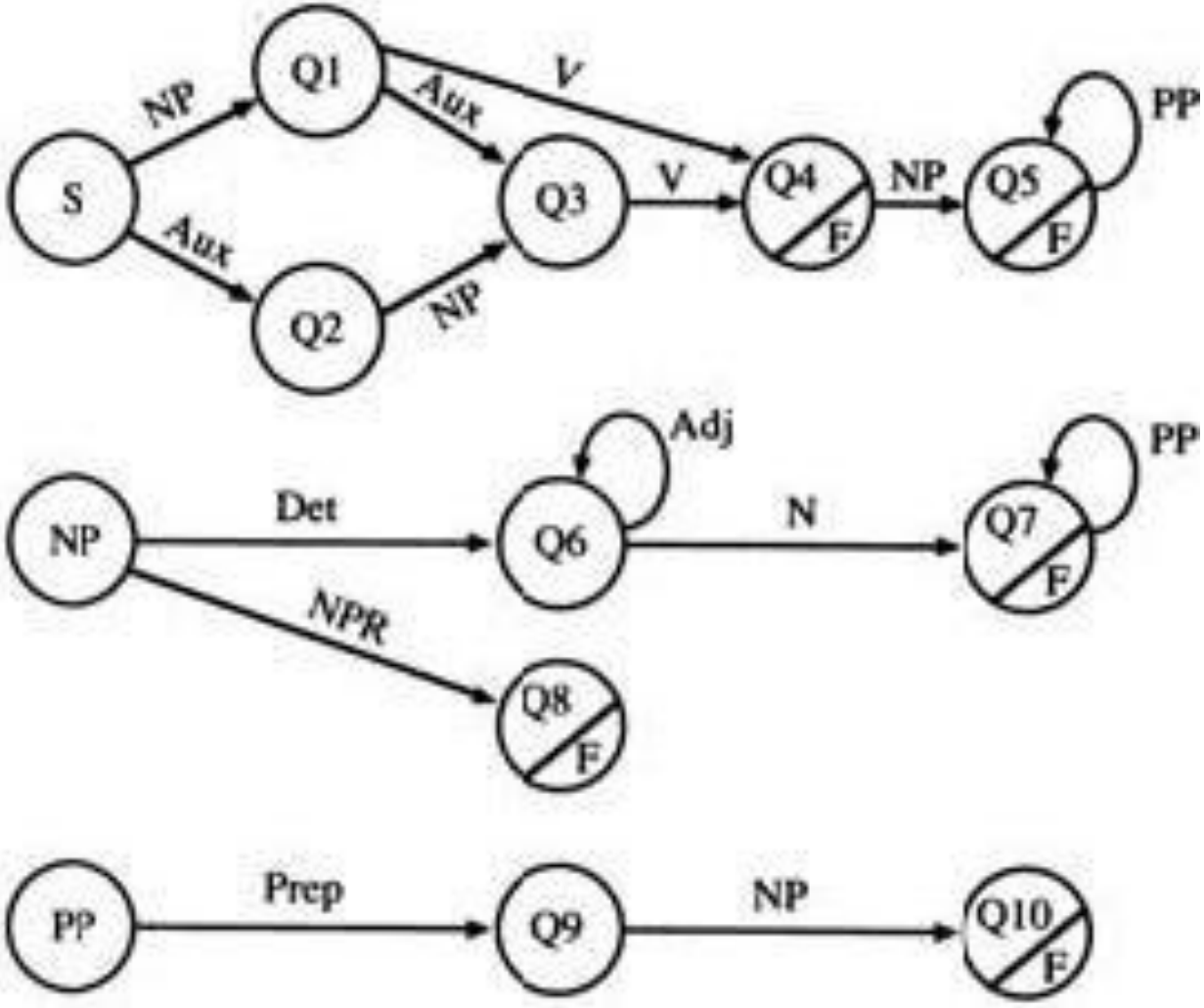
Αποτελεσματικότητα Chart Parsing

- Το chart parsing είναι σημαντικά πιο αποτελεσματικό σε σχέση με τις απλές μεθόδους αναζήτησης που δεν κρατούν κομμάτια της ανάλυσης.
- Για σχετικά μεγάλες προτάσεις και περίπλοκες γραμματικές το chart parsing υπερτερεί κατά πολύ.

Επαυξημένα Δίκτυα Μετάβασης (Augmented Transition Networks)

- Επιτρέπουν τον συνδυασμό διαφόρων ειδών γνώσης
- Είναι παρεμφερή με τα αυτόματα πεπερασμένων καταστάσεων
  - Αποτελούνται από ένα αριθμό κόμβων (καταστάσεις) και τόξα που συνδέουν τις καταστάσεις και αντιστοιχούν:
    - Σε συγκεκριμένες λέξεις
    - Σε κατηγορίες λέξεων
    - Σε κλήσεις σε άλλα δίκτυα
    - Σε διαδικασίες ελέγχου

Παράδειγμα ATN Γραμματικής



Διαδικασία: “The long file has printed.”

Η εκτέλεση εξελίσσεται ως εξής:

1. Ξεκίνησε από την κατάσταση S.
2. Κάλεσε (κάνε μετάβαση) στο NP.
3. Κάνε έλεγχο κατηγορίας (μέρος του λόγου) για να διαπιστωθεί αν το “the” είναι άρθρο.
4. Ο έλεγχος επιτυγχάνει, θέσε τον καταχωρητή DETERMINER (άρθρο) σε DEFINITE (οριστικό) και προχωρά στην κατάσταση Q6.
5. Κάνε έλεγχο κατηγορίας για να διαπιστωθεί αν το “long” είναι επίθετο.
6. Ο έλεγχος επιτυγχάνει, έτσι προσαρτάται στη λίστα που περιέχεται στον καταχωρητή ADJS (αυτή η λίστα ήταν προηγουμένως άδεια). Αναμονή του συστήματος στην κατάσταση Q6.

Διαδικασία: “The long file has printed.”

7. Κάνε έλεγχο κατηγορίας για να διαπιστωθεί αν το “file” είναι επίθετο. Αυτός ο έλεγχος αποτυγχάνει.
8. Κάνε έλεγχο κατηγορίας για να διαπιστωθεί αν το “file” είναι ουσιαστικό. Ο έλεγχος επιτυγχάνει, έτσι θέσε τον καταχωρητή NOUN σε “file” και προχώρα στην κατάσταση Q7.
9. Μετάβαση στο PP.
10. Κάνε έλεγχο κατηγορίας για να διαπιστωθεί αν το “has” είναι πρόθεση. Ο έλεγχος αποτυγχάνει και έτσι εκτέλεσε μια λειτουργία επιστροφής (pop) και σηματοδότησε την αποτυχία.

Διαδικασία: “The long file has printed.”

11. Δεν υπάρχει κάτι άλλο που να μπορεί να γίνει πέρα από την κατάσταση Q7, έτσι εκτέλεσε μια λειτουργία επιστροφής και επέστρεψε τη δομή

    (NP (FILE (LONG) DEFINITE))

    Με την επιστροφή αυτή, η μηχανή οδηγείται στην κατάσταση Q1.

12. Κάνε έλεγχο κατηγορίας για να διαπιστωθεί αν το “has” είναι ρήμα. Υπάρχει επιτυχία και ο καταχωρητής AUX τίθεται στη μηδενική τιμή NIL ενώ ο καταχωρητής V σε “has”.

13. Μετάβαση στην κατάσταση Q4.

    Προωθήσου στην κατάσταση NP. Αφού η επόμενη λέξη “printed” δεν είναι άρθρο ούτε ουσιαστικό, η NP θα επιστρέψει αποτυχία.

Διαδικασία: “The long file has printed.”

14. Το μόνο που μπορεί πλέον να γίνει στην κατάσταση Q4 είναι ο τερματισμός. Όμως, αφού παραμένει ακόμη είσοδος και δεν έχει βρεθεί η πλήρης ανάλυση, η οπισθοδρόμηση είναι απαραίτητη.
15. Το τελευταίο σημείο επιλογής βρισκόταν στην κατάσταση Q1, οπότε επέστρεψε εκεί. Οι καταχωρητές AUX και V πρέπει να καθαρισθούν.
16. Κάνε έλεγχο κατηγορίας για να διαπιστωθεί αν το “has” είναι βοηθητικό ρήμα. Ο έλεγχος επιτυγχάνει, έτσι θέσε το AUX σε “has” και εκτέλεσε μετάβαση στην κατάσταση Q3.
17. Κάνε έλεγχο κατηγορίας για να διαπιστωθεί αν το “printed” είναι ρήμα. Ο έλεγχος επιτυγχάνει, και θέσε τον καταχωρητή V σε “printed”. Κάνε μετάβαση στην κατάσταση Q4.
18. Τώρα, μια και δεν απομένει άλλη είσοδος, η κατάσταση Q4 είναι μια αποδεκτή τελική κατάσταση. Εκτέλεσε μια λειτουργία επιστροφής (pop) και επέστρεψε τη δομή:

    (S DCL (NP(FILE(LONG)DEFINITE))
    HAS
    (VP PRINTED))

Αυτή η δομή είναι και η έξοδος της ανάλυσης.

Deductive Route

- Σε αυτή τη διάλεξη
  - υλοποιήσαμε ένα μοντέλο κανόνων (γραμματική) και
  - χρησιμοποιήσαμε συμπερασμό με βάση το μοντέλο κανόνων (rule-based reasoning) για να αναλύσουμε καινούρια δεδομένα (προτάσεις)