ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΙΚΑ
ΜΑΘΗΜΑΤΑ

Σχεδίαση Γλωσσών & Μεταγλωττιστές

Ενότητα 4: Συντακτική Ανάλυση

Επ. Καθ. Π. Κατσαρός
Τμήμα Πληροφορικής

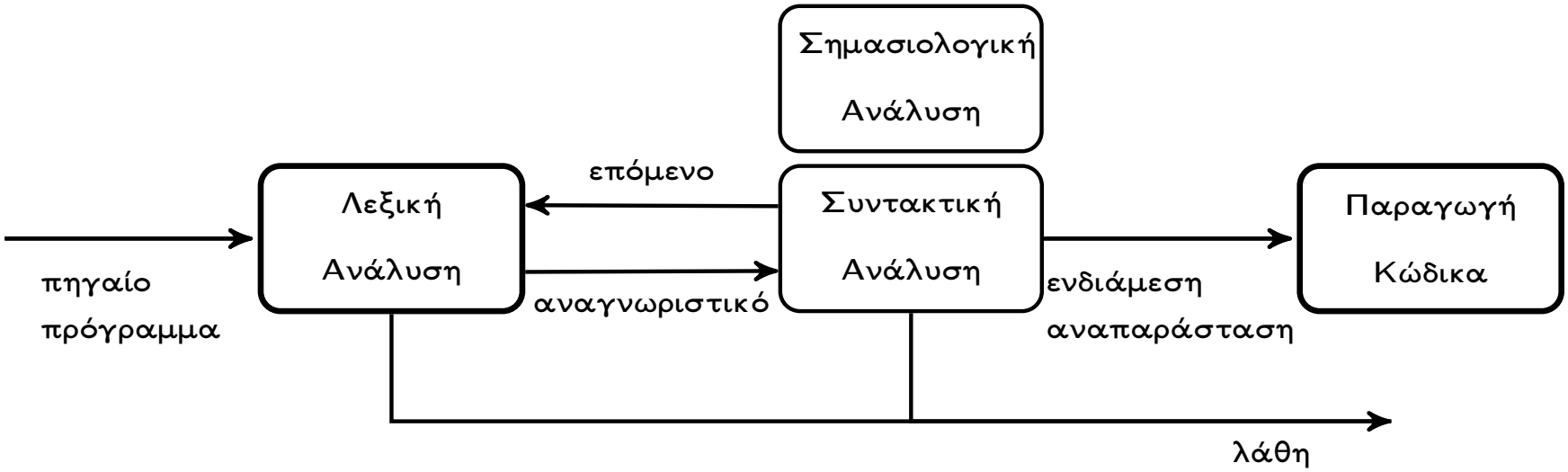
Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδεια χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο«Ανοικτά Ακαδημαϊκά Μαθήματα στο Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης» έχει χρηματοδοτήσει μόνο την 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.

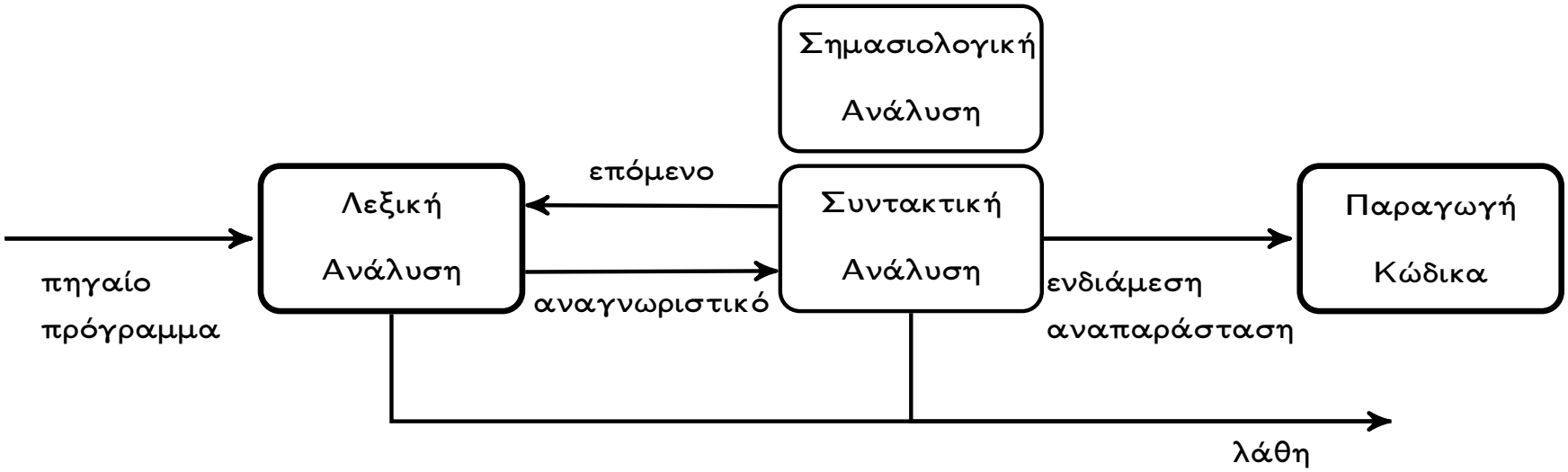
Βασικές έννοιες



Η Συντακτική Ανάλυση δέχεται μια ροή αναγνωριστικών, για την οποία καλείται να ανιχνεύσει τη συντακτική δομή του πηγαίου προγράμματος.

- Αν η σύνταξη συμμορφώνεται στον ορισμό της Γ.Π., τότε αυτή **μπορεί** να αποτυπώνεται σε δομή δέντρου, την **ενδιάμεση αναπαράσταση**. Υπάρχουν και γραμμικές ενδιάμεσες αναπαραστάσεις.
- Αν η σύνταξη δε συμμορφώνεται στον ορισμό της Γ.Π., τότε ο αλγόριθμος Σ.Α. εντοπίζει ένα **συντακτικό λάθος** και στη συνέχεια εκτελεί μια διαδικασία **ανάνηψης**, ώστε να συνεχίσει την ανάλυση μέχρι το τέλος του προγράμματος.

Βασικές έννοιες



- Η Σ.Α. ενεργοποιεί κατά την ανίχνευση τους ελέγχους σημασίας του προγράμματος (π.χ. κανόνες εμβέλειας, τύπων κ.α.) και γι΄ αυτό αντλούνται πληροφορίες από το δέντρο και από τον πίνακα συμβόλων.
- Το δέντρο αντικατοπτρίζει τη σειρά με την οποία ανιχνεύεται η θέση των αναγνωριστικών στη συντακτική δομή.
- Αν μία μόνο διάσχιση του δέντρου αρκεί για την εκτέλεση των σημασιολογικών ελέγχων, τότε δεν αποθηκεύεται δομή δεδομένων στη μνήμη. Εκτελείται **μετάφραση σε ένα πέρασμα** του κώδικα.
- Για πιο απαιτητικούς σημασιολογικούς ελέγχους αποθηκεύεται η δεντρική δομή και εκτελείται **μετάφραση σε περισσότερα περάσματα**.

Συντακτική δομή Γ.Π.: Γραμματικές Χωρίς Συμφραζόμενα

Η συντακτική δομή των Γ.Π. περιγράφονται από γραμματικές χωρίς συμφραζόμενα.

Ορισμός 1 (Γραμματική Χωρίς Συμφραζόμενα)

Γραμματική Χωρίς Συμφραζόμενα G είναι μία τετράδα (V, Σ, R, S) όπου:

- V είναι ένα πεπερασμένο σύνολο συμβόλων, το **αλφάβητο** της G
- $\Sigma \subseteq V$ περιλαμβάνει τα **τερματικά σύμβολα**
- R ένα πεπερασμένο σύνολο **κανόνων παραγωγής**, που ορίζεται ως υποσύνολο του $(V - \Sigma) \times V^*$. Κάθε κανόνα $(A, u) \in R$ με $A \in V - \Sigma$ και $u \in V^*$ τον γράφουμε ως
$$A \longrightarrow u$$
- $S \in V - \Sigma$ ένα **μη τερματικό σύμβολο** που καλείται **αρχή** της G

Συντακτική δομή Γ.Π.: Γραμματικές Χωρίς Συμφραζόμενα

Αν η συμβολοσειρά $w \in \Sigma^*$ παράγεται από το S μέσω κανόνων της G, τότε γράφουμε $S \Rightarrow^* w$ και $w \in L(G)$, δηλαδή η w **συμμορφώνεται στη σύνταξη των προτάσεων της γλώσσας** της G.

- Στη ΓΧΣ που ορίζει της σύνταξη μιας Γ.Π. **τερματικά σύμβολα είναι τα αναγνωριστικά που παράγει η Λ.Α.**
- Μη τερματικά είναι τα σύμβολα που εκφράζουν τις συντακτικές δομές της Γ.Π., ενώ η αρχή της ΓΧΣ αντιστοιχεί σε ένα οποιοδήποτε πρόγραμμα.

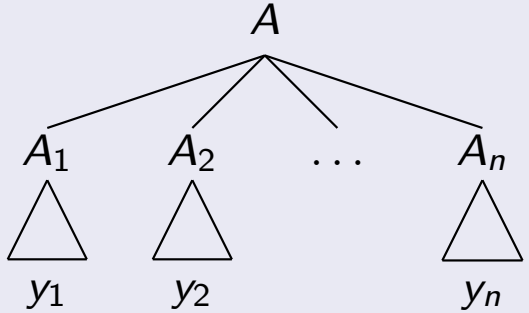
Η σύνταξη ενός προγράμματος εκφράζεται από ένα **παράγωγο δέντρο**.

Σύνταξη προγράμματος: Παράγωγο Δέντρο

Ορισμός 2 (Παράγωγο δέντρο)

Για μια ΓΧΣ $G = (V, \Sigma, R, S)$:

- Για κάθε $\alpha \in \Sigma$ ο γράφος με ένα μόνο κόμβο, α είναι παράγωγο δέντρο με ρίζα και μοναδικό φύλλο τον κόμβο α.
- Αν $A \longrightarrow \epsilon$ είναι κανόνας του R (ϵ η κενή συμβ/ρά), τότε το $\begin{array}{c} A \\ | \\ \epsilon \end{array}$ είναι παράγωγο δέντρο με ρίζα τον κόμβο A και μοναδικό φύλλο τον ϵ.
- Αν τα 

είναι παράγωγα δέντρα με ρίζες $A_1, A_2, ..., A_n$ και με παραγόμενες συμβ/ρές τις $y_1, y_2, ..., y_n$ και αν $A \longrightarrow A_1 A_2 ... A_n$ είναι κανόνας του R, τότε το 

είναι παράγωγο δέντρο με ρίζα A και παραγόμενη συμβ/ρά την $y_1 \cdot y_2 ... \cdot y_n$.

Παράδειγμα ΓΧΣ και παράγωγου δέντρου

ΓΧΣ για εντολές εκχώρησης & παράγωγο δέντρο για την a[4] :=3×5.

assign-stmt → *var* **:=** *expr*	(1)
expr → *term*	(2)
| *expr* + *term*	(3)
| *expr* - *term*	(4)
term → *prim-expr*	(5)
| *term* × *prim-expr*	(6)
| *term* / *prim-expr*	(7)
prim-expr → *var*	(8)
| **NUM**	(9)
| **(** *expr* **)**	(10)
var → **ID**	(11)
| **ID** [*sbscr-lst*]	(12)
sbscr-lst → *expr*	(13)
| *sbscr-lst*, *expr*	(14)

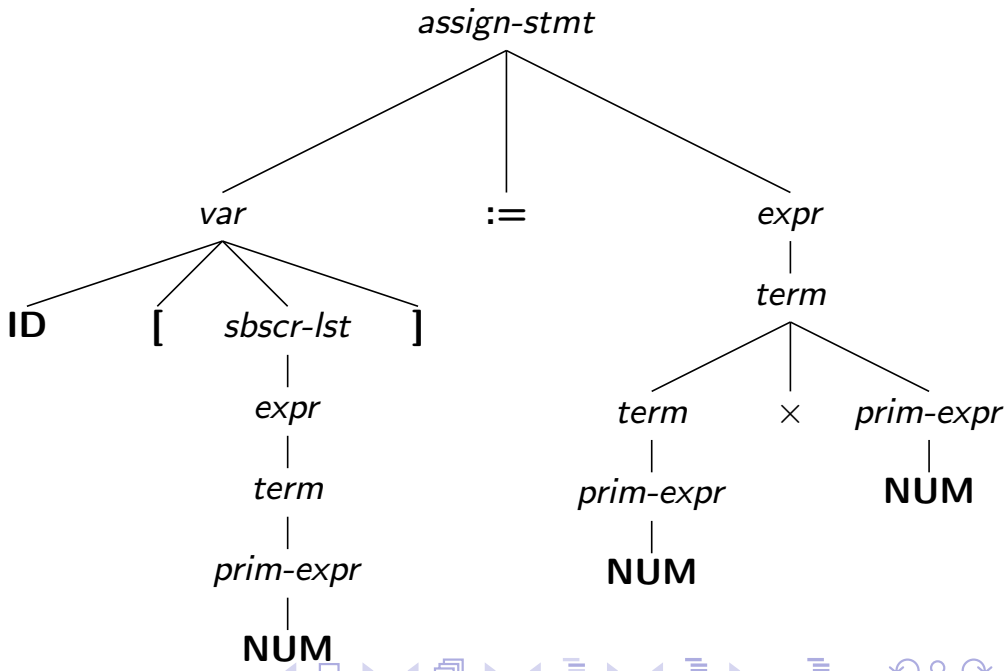
assign-stmt $\overset{(1)}{\Rightarrow}$ *var*:=*expr* $\overset{(12)}{\Rightarrow}$ **ID**[*sbscr-lst*]:=*expr* $\overset{(13)}{\Rightarrow}$ **ID**[*expr*]:=*expr*

$\overset{(2)}{\Rightarrow}$ **ID**[*term*]:=*expr* $\overset{(5)}{\Rightarrow}$ **ID**[*prim-expr*]:=*expr* $\overset{(8)}{\Rightarrow}$ **ID**[**NUM**]:=*expr*

$\overset{(2)}{\Rightarrow}$ **ID**[**NUM**]:=*term* $\overset{(6)}{\Rightarrow}$ **ID**[**NUM**]:=*term*×*prim-expr*

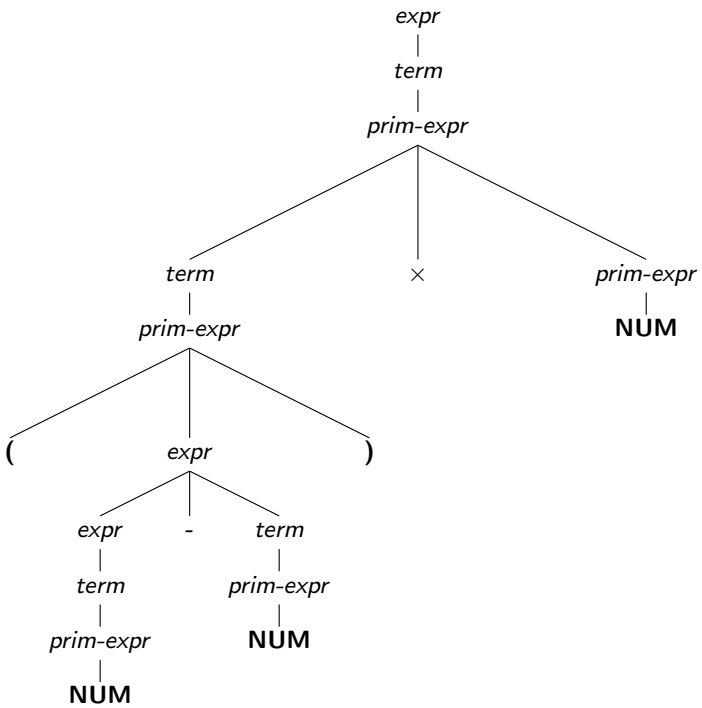
$\overset{(6)}{\Rightarrow}$ **ID**[**NUM**]:=*prim-expr*×*prim-expr*

$\overset{(9)}{\Rightarrow}$ **ID**[**NUM**]:=**NUM**×*prim-expr* $\overset{(9)}{\Rightarrow}$ **ID**[**NUM**]:=**NUM**×**NUM**

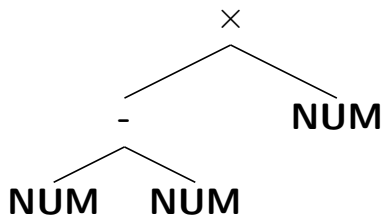


Ενδιάμεση αν/σταση: Αφαιρετικό Συντακτικό Δέντρο

- Τα παράγωγα δέντρα διατηρούν όλη την πληροφορία που χρησιμοποιείται κατά τη συντακτική ανάλυση.
- Κάποιες από αυτές τις πληροφορίες δεν επηρεάζουν τη σημασία της Γ.Π. και θα μπορούσαν να παραλειφθούν, για να απλουστευθεί η επεξεργασία (διάσχιση) του δέντρου.
- Η αφαιρετική σύνταξη δεν περιλαμβάνει όλες τις λεπτομέρειες



Π.χ. μπορούμε να παραλείψουμε τις παρενθέσεις στις εκφράσεις και οι πράξεις να ομαδοποιούνται απευθείας στη δομή του δέντρου.



Σύνταξη και σημασία προγραμμάτων

Ορισμός 3 (Διφορούμενη ή ασαφής ΓΧΣ)

Υπάρχει συμβ/ρά της γλώσσας με τουλάχιστο δύο παράγωγα δέντρα (δύο διαφορετικές αριστερές ή δύο διαφορετικές δεξιές παραγωγές)

ΓΧΣ με πρόβλημα στον ορισμό της **προτεραιότητας των τελεστών**:

$$
\begin{array}{lll}
expr \rightarrow & expr\ OP\ expr & (1)\\
\mid & \textbf{(}\ expr\ \textbf{)} & (2)\\
\mid & \textbf{NUM} & (3)\\
OP \rightarrow & + & (4)\\
\mid & - & (5)\\
\mid & * & (6)\\
\mid & / & (7)
\end{array}
$$

Αριστερή παραγωγή της συμβ/ράς $\mathbf{27 - 5 * 8}$

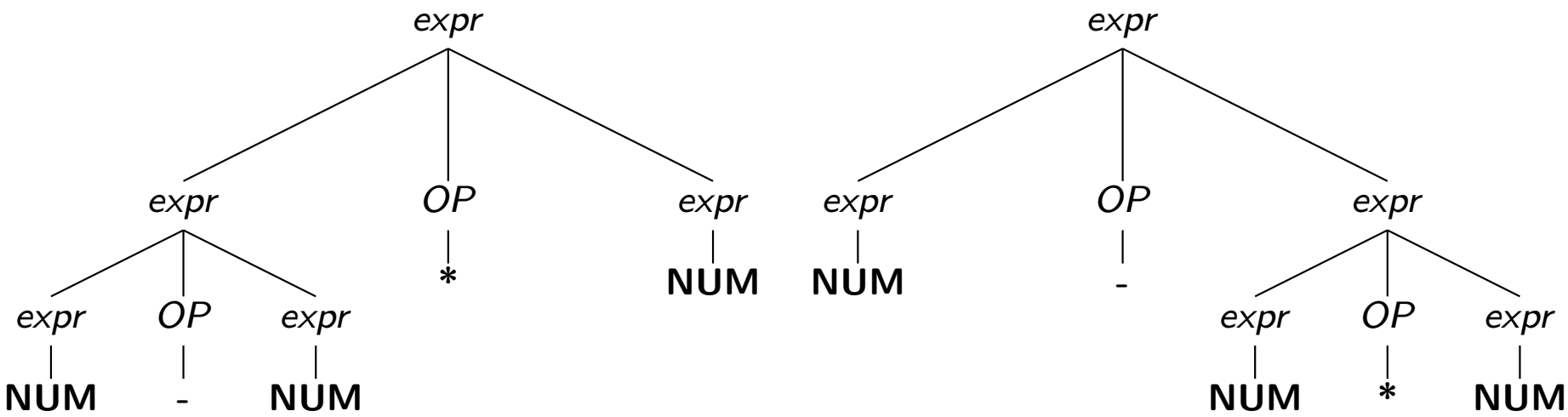
$expr \overset{(1)}{\Rightarrow} expr\ OP\ expr \overset{(1)}{\Rightarrow} expr\ OP\ expr\ OP\ expr$
$\overset{(3)}{\Rightarrow} \textbf{NUM}\ OP\ expr\ OP\ expr \overset{(5)}{\Rightarrow} \textbf{NUM - }\ expr\ OP\ expr$
$\overset{(3)}{\Rightarrow} \textbf{NUM - NUM}\ OP\ expr \overset{(6)}{\Rightarrow} \textbf{NUM - NUM *}\ expr$
$\overset{(3)}{\Rightarrow} \textbf{NUM - NUM * NUM}$

Δεύτερη αριστερή παραγωγή της ίδιας συμβ/ράς

$expr \overset{(1)}{\Rightarrow} expr\ OP\ expr \overset{(3)}{\Rightarrow} \textbf{NUM}\ OP\ expr$
$\overset{(5)}{\Rightarrow} \textbf{NUM - }\ expr \overset{(1)}{\Rightarrow} \textbf{NUM - }\ expr\ OP\ expr$
$\overset{(3)}{\Rightarrow} \textbf{NUM - NUM}\ OP\ expr \overset{(6)}{\Rightarrow} \textbf{NUM - NUM *}\ expr$
$\overset{(3)}{\Rightarrow} \textbf{NUM - NUM * NUM}$

Σύνταξη και σημασία προγραμμάτων

Δέντρα της ΓΧΣ για τις δύο αριστερές παραγωγές της $\mathbf{27 - 5 * 8}$ (η σωστή σημασία/αποτέλεσμα δίνεται από το δεξί δέντρο):



Η σύνταξη μιας γλώσσας επηρεάζει τη σημασία της

Η ΓΧΣ που περιγράφει τη σύνταξη δεν πρέπει να είναι διφορούμενη ή αλλιώς πρέπει ο αλγόριθμος μεταγλώττισης να χειρίζεται όλες τις ασάφειες κατά τρόπο συμβατό με τη σημασία της γλώσσας.

Σύνταξη και σημασία προγραμμάτων

Διορθωμένη ισοδύναμη ΓΧΣ (παράγει την ίδια γλώσσα με ένα μόνο δέντρο για τη συμβ/ρά $\mathbf{27 - 5 * 8}$):

$$
\begin{aligned}
\mathit{expr} &\rightarrow \mathit{expr}\ \mathit{ADOP}\ \mathit{expr} \\
&\mid \mathit{term} \\
\mathit{term} &\rightarrow \mathit{term}\ \mathit{MULOP}\ \mathit{term} \\
&\mid \mathit{factor} \\
\mathit{factor} &\rightarrow \mathbf{NUM} \\
&\mid \mathbf{(}\ \mathit{expr}\ \mathbf{)} \\
\mathit{ADOP} &\rightarrow + \\
&\mid - \\
\mathit{MULOP} &\rightarrow * \\
&\mid /
\end{aligned}
$$

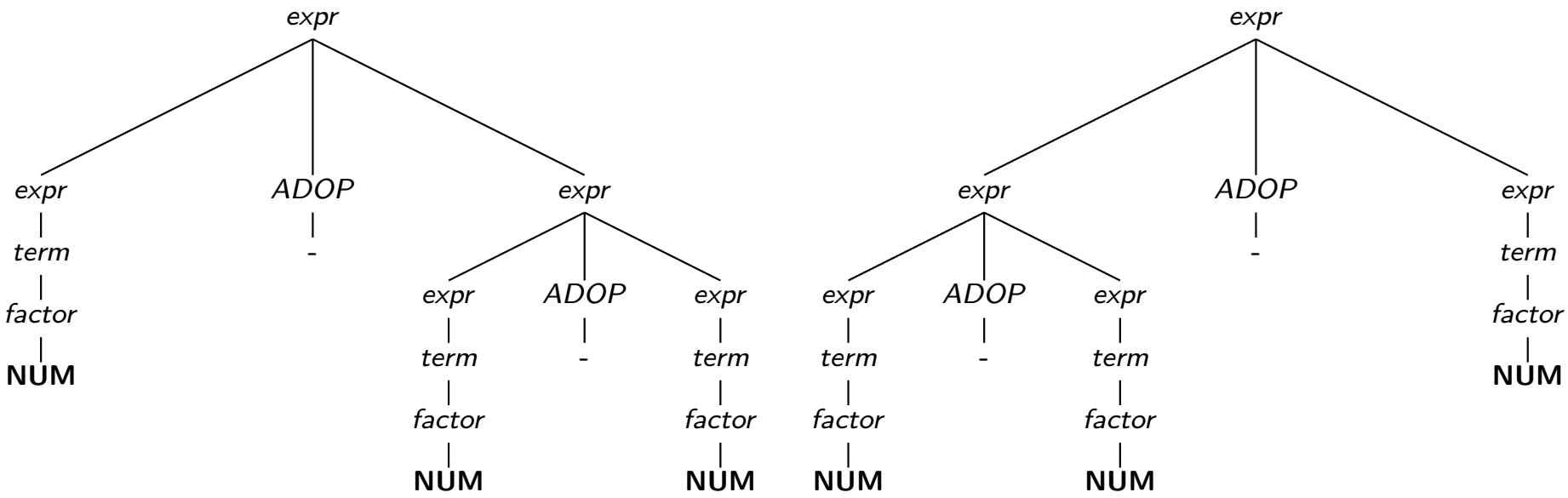
Προτεραιότητα τελεστών στις ΓΧΣ

Ομαδοποιούμε τους τελεστές που έχουν ίδια προτεραιότητα σε κανόνα με ένα κοινό μη τερματικό σύμβολο.

Σύνταξη και σημασία προγραμμάτων

Η διορθωμένη ΓΧΣ είναι διφορούμενη ως προς την **προσεταιριστικότητα των τελεστών** της ίδιας προτεραιότητας.

- Δέντρα της διορθωμένης ΓΧΣ για δύο αριστερές παραγωγές της **27 − 5 − 8** (η σωστή σημασία δίνεται από το δεξί δέντρο):



Σύνταξη και σημασία προγραμμάτων

Διορθωμένη ισοδύναμη ΓΧΣ (παράγει την ίδια γλώσσα με ένα μόνο δέντρο για τη συμβ/ρά $\mathbf{27 - 5 - 8}$):

$$
\begin{aligned}
\mathit{expr} &\rightarrow \mathit{expr}\ \mathit{ADOP}\ \mathit{term} \\
&\mid \mathit{term} \\
\mathit{term} &\rightarrow \mathit{term}\ \mathit{MULOP}\ \mathit{factor} \\
&\mid \mathit{factor} \\
\mathit{factor} &\rightarrow \mathbf{NUM} \\
&\mid \mathbf{(}\ \mathit{expr}\ \mathbf{)} \\
\mathit{ADOP} &\rightarrow + \\
&\mid - \\
\mathit{MULOP} &\rightarrow * \\
&\mid /
\end{aligned}
$$

Αριστερή Προσεταιριστικότητα στις ΓΧΣ

Ένας κανόνας ΓΧΣ παράγει δέντρο με αριστερή προσεταιριστικότητα αν αυτός είναι **αποκλειστικά αριστερά αναδρομικός**.

Σύνταξη και σημασία προγραμμάτων

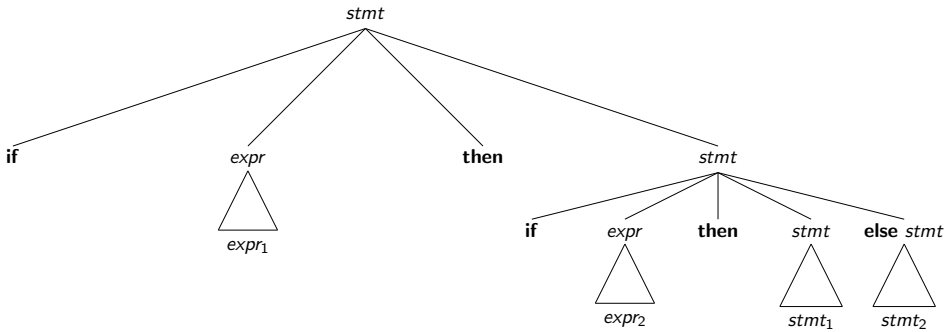
Διφορούμενη ΓΧΣ με το πρόβλημα του μετέωρου else:

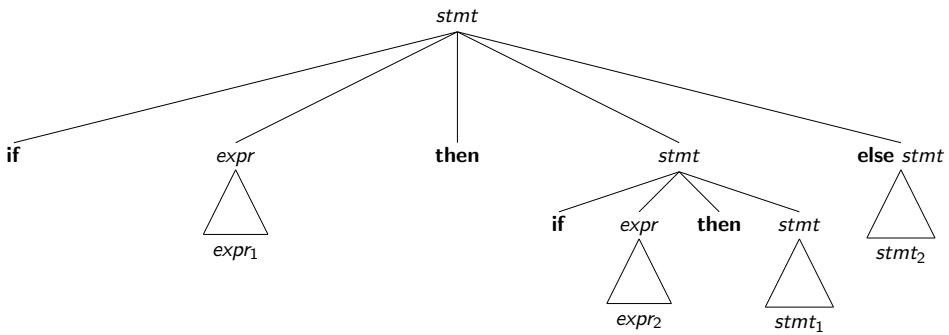
$$\begin{aligned} \mathit{stmt} \rightarrow\ & \textbf{if}\ \mathit{expr}\ \textbf{then}\ \mathit{stmt} \\ \mid\ & \textbf{if}\ \mathit{expr}\ \textbf{then}\ \mathit{stmt}\ \textbf{else}\ \mathit{stmt} \end{aligned}$$

Για το πρόγραμμα

if $expr_1$ **then if** $expr_2$ **then** $stmt_1$ **else** $stmt_2$

έχουμε δύο πιθανά δέντρα με διαφορετική σημασία εκτέλεσης:





Σύνταξη και σημασία προγραμμάτων

Η αρχή του πλησιέστερου if

Όταν δεν γίνεται ένθεση εντολών if με χρήση διαχωριστών της γλώσσας, τότε κάθε else εξορισμού αναφέρεται στο πλησιέστερο προαναφερόμενο if.

Διορθωμένη ΓΧΣ, που υλοποιεί τη σημασία του πλησιέστερου if

$$
\begin{aligned}
\mathit{stmt} &\rightarrow \mathit{bal\_stmt} \\
&\mid \mathit{unbal\_stmt} \\
\mathit{bal\_stmt} &\rightarrow \textbf{if}\ \mathit{expr}\ \textbf{then}\ \mathit{bal\_stmt}\ \textbf{else}\ \mathit{bal\_stmt} \\
\mathit{unbal\_stmt} &\rightarrow \textbf{if}\ \mathit{expr}\ \textbf{then}\ \mathit{stmt} \\
&\mid \textbf{if}\ \mathit{expr}\ \textbf{then}\ \mathit{bal\_stmt}\ \textbf{else}\ \mathit{unbal\_stmt}
\end{aligned}
$$

Μετασχηματισμός ΓΧΣ

Κάθε μετασχηματισμός ΓΧΣ που αποσκοπεί σε απαλοιφή ασάφειας έχει ως συνέπεια η νέα γραμματική να περιγράφει μια πιο σύνθετη και επομένως δυσκολότερα κατανοητή σύνταξη.

Δηλωτικές & Αλγοριθμικές Γλώσσες

Περιπτώσεις δηλωτικών (declarative) γλωσσών:

- Συναρτησιακές Γλώσσες: Haskell, ML κ.α.
- Γλώσσες Λογικού Προγραμματισμού: Prolog κ.α.
- Γλώσσες Δεδομένων (dataflow): Lucid κ.α.
- Γλώσσες Κανόνων (rule-based): CLIPS κ.α.
- Γλώσσες Περιγραφικής Επισημείωσης (descriptive markup): HTML, SGML, XML κ.α.

Περιπτώσεις αλγοριθμικών (imperative) γλωσσών:

- Διαδικασιακές Γ.Π.: FORTRAN, C, Pascal κ.α.
- Αντικειμενοστρεφείς Γ.Π.: Smalltalk, C++, Java κ.α.
- Γλώσσες Πολυπρογραμματισμού & Πολυεπεξεργασίας: Erlang, Ada, Java κ.α.
- Γλώσσες Σεναρίων & Διαδικτύου: ECMAScript, Perl, Python, PHP κ.α.
- Γ.Π. Ενσωματωμένων Συστημάτων - Πραγματικού Χρόνου: nesC, Esterel, Real-Time Java κ.α.
- Γλώσσες Υψηλής Ακεραιότητας: SPARKAda
- Συμβολικές Γλώσσες Μηχανής: Assembly

Παράδειγμα κειμένου HTML

```
<!DOCTYPE html>
<html>
<body>

<h4> One row and three columns: </h4>
<table border="1">
<tr>
  <td>100</td>
  <td>200</td>
  <td>300</td>
</tr>
</table>

<p><b> This text is bold </b></p>

<p>
<a href="http://www.w3.org/" > W3C </a>
This is a link to a website on the World Wide Web.
</p>

</body>
</html>
```

Υπολογισμός αριθμών Fibonacci σε Haskell

```
module Main( main ) where

import System( getArgs )

fib 0 = 0

fib 1 = 1

fib n = fib (n-1) + fib (n-2)


f n = (if n<0 && even n then -1 else 1) * fib (abs n)


f_print n = print(show n ++ "th Fibonacci number is " ++ show (f n))


main = do

    args <- getArgs

    if (length args /= 1)

        then putStr "Usage: f1a <n>"

        else (f_print (read (head args)))

    putStr "\n"
```

Σύνταξη Αλγοριθμικών Γλωσσών

- Ορισμοί τύπων: εισάγουν νέους τύπους.
- Δηλώσεις μεταβλητών: εισάγουν ονόματα δεδομένων.
- Εντολές εκχώρησης: προκαλούν αλλαγή στην τιμή ενός δεδομένου.
- Εντολές I/O
- Σύνθετες εντολές: ακολουθία/block εντολών
- Εντολές ελέγχου ροής: επηρεάζουν την προκαθορισμένη γραμμική ροή της εκτέλεσης
- Άλλες εντολές

Υπολογισμός αριθμών Fibonacci σε FORTRAN 2003

```
program f1a
use f1a_module
integer:: i,k
character:: arg*128
if (command_argument_count() == 1) then
    call get_command_argument(1, arg)
    if (verify(arg, "0123456789 ")==0) then
        read(arg, fmt=*) i
        k = f(i)
        print "(i0,a,i0)",i,"th Fibonacci number is ",k
        stop
    endif
endif
print *,"Usage: f1a <n>"
stop
end program
```

```
module f1a_module
public :: f
contains
integer pure recursive function fib(n) result(ret)
integer,intent(in):: n
if (n>1) then
    ret = fib (n-1) + fib (n-2)
else
    ret = n
endif
end function fib
integer elemental function f(n)
integer,intent(in):: n
f = sign(1,n)**(n+1) * fib(abs(n))
end function f
end module f1a_module
```

Υπολογισμός αριθμών Fibonacci σε ANSI C

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <inttypes.h>

int64_t fib(int n) {
    return n<2 ? n : fib(n-1)+fib(n-2);
}

int64_t f(int n) {
    if(n<0)
        return n%2 ? fib(-n) : -fib(-n);
    else
        return fib(n);
}

void f_print(int n) {
    printf("%dth Fibonacci number is %" PRId64 "\n",n,f(n));
}

int main(int argc, char** argv) {
    if (argc!=2 || (atoi(argv[1])==0 && strcmp(argv[1],"0")!=0)) {
        printf("Usage: f1a <n>\n");
    else
        f_print(atoi(argv[1]));
}
```

Υπολογισμός αριθμών Fibonacci σε Java

```
class f1a {
 private static long fib(long n) {
        return n<2 ? n : fib(n-2)+fib(n-1);
 }

 private static long f(long n) {
    if(n<0)
        return (n%2==0) ? -fib(-n) : fib(-n);
    else
        return fib(n);
 }

 private static void fib_print(int n) {
  System.out.println(n + "th Fibonacci number is " + f(n));
 }

 public static void main(String argv[]) {
  try {
      if(argv.length == 1) {
          fib_print(Integer.parseInt(argv[0]));
          System.exit(0);
      }
  } catch (NumberFormatException e) { }
  System.out.println("Usage: java f1a ");
  System.exit(1);
 }
}
```

Υπολογισμός αριθμών Fibonacci σε Perl

```
sub fib {
    return @_[0] < 2 ? @_[0] : fib(@_[0]-1) + fib(@_[0]-2)
}

sub f {
    if (@_[0] < 0) {
        return @_[0] % 2 ? fib(-@_[0]) : -fib(-@_[0]);
    } else {
        return fib(@_[0]);
    }
}
```

```
sub fib_print {
    print @_[0], "th Fibonacci number is ", f(@_[0]), "\n";
}

if ( $#ARGV || int($ARGV[0]) ne $ARGV[0]) {
    die "Usage: perl ", $0, " <n>\n";
} else {
    fib_print($ARGV[0]);
}
```

Υπολογισμός αριθμών Fibonacci σε UNIX shell

```
#!/bin/bash
fib () {
  if test $1 -lt 2; then
    echo $1
  else
    echo $(($(fib $(($1-2)))+$(fib $(($1-1)))))
  fi
}
f () {
  if test $1 -lt 0; then
    if test $(($1%2)) -eq 0; then
      echo $((0-$(fib $((0-$1)))))
    else
      echo $(fib $((0-$1)))
    fi
  else
    echo $(fib $1)
  fi
}

main () {
  if [ "$#" -ne 1 ]; then
    echo "Usage: $0 <n>"
  else
    RET=$(f $1)
    echo $1 "th Fibonacci number is" $RET
  fi
}
# entry point
main $1
```

Καθοδική και Ανοδική Ανάλυση

Αλγόριθμοι Καθοδικής Συντακτικής Ανάλυσης

- Το παράγωγο/αφαιρετικό συντακτικό δέντρο που αντιστοιχεί στο πηγαίο πρόγραμμα **διατρέχεται από τη ρίζα** (αρχή γραμματικής) **προς τα φύλλα** (τερματικά σύμβολα από τη Λ.Α. του προγράμματος).
- Δύο οικογένειες αλγορίθμων: οπισθοδρόμησης (backtracking) και πρόγνωσης. Πιο διαδεδομένοι είναι οι αλγόριθμοι πρόγνωσης recursive descent και LL(1).
- Εφαρμόσιμη όταν η γραμματική χωρίς συμφραζόμενα δεν είναι αριστερά αναδρομική.
- Τα βήματα της ανάλυσης αντιστοιχούνε σε αριστερές παραγωγές.

Αλγόριθμοι Ανοδικής Συντακτικής Ανάλυσης

- Το παράγωγο/αφαιρετικό συντακτικό δέντρο που αντιστοιχεί στο πηγαίο πρόγραμμα **διατρέχεται από τα φύλλα προς τη ρίζα**.
- Πιο διαδεδομένοι είναι οι αλγόριθμοι πρόγνωσης της οικογένειας LR.
- Εφαρμόζονται σε μία ευρύτερη οικογένεια γραμματικών χωρίς συμφραζόμενα.
- Τα βήματα της ανάλυσης αντιστοιχούνε σε δεξιές παραγωγές.

Αλγόριθμοι οπισθοδρόμησης και αλγόριθμοι πρόγνωσης

Αλγόριθμοι Οπισθοδρόμησης

- Επιλέγεται ad hoc μια σειρά γραμματικών κανόνων και αν η ανάλυση φθάσει σε αδιέξοδο, τότε αναιρεί έναν ή περισσοτέρους κανόνες και επιλέγει άλλους κανόνες για τα ίδια μη τερματικά (διαφορετική σειρά).
- Συγκριτικά υψηλό υπολογιστικό κόστος μεταγλώττισης.
- Η αναγνώριση συντακτικού λάθους καθυστερεί και αυτό περιπλάκει την ανάνηψη από λάθη.
- Τα περισσότερα - αν όχι όλα - τα χαρακτηριστικά Γ.Π. μπορούν να περιγραφούν από Γ.Χ.Σ. με περιορισμένες ή ανύπαρκτες απαιτήσεις οπισθοδρόμησης.

Αλγόριθμοι Πρόγνωσης

- Διαβάζεται ένα ή περισσότερα τερματικά (αναγνωριστικά) στην είσοδο και με βάση αυτά επιλέγεται ποιος κανόνας θα χρησιμοποιηθεί από αυτούς που αναφέρονται σε ένα μη τερματικό σύμβολο.
- Συγκριτικά αποδοτικότερη μεταγλώττιση και πιο απλή ανάνηψη από λάθη.

Ονοματολογία αλγορίθμων Σ.Α.

XY(n)

Το X αναφέρεται στη φορά ανάγνωσης των συμβόλων του πηγαίου προγράμματος

- Για X=L έχουμε left-to-right ανάγνωση του προγράμματος, αν αυτό βρίσκεται τοποθετημένο σε μία μαγνητική ταινία.

Το Y αναφέρεται στον τύπο των παραγωγών που ακολουθεί ο αλγόριθμος

- Για Y=L έχουμε αριστερές left παραγωγές και αυτό είναι χαρακτηριστικό των αλγορίθμων καθοδικής ανάλυσης.
- Για Y=R έχουμε δεξιές right παραγωγές και αυτό είναι χαρακτηριστικό των αλγορίθμων ανοδικής ανάλυσης.

Το *n* αναφέρεται στον αριθμό των τερματικών συμβόλων που διαβάζονται στην είσοδο για να επιλεγεί ο κανόνας που θα χρησιμοποιηθεί. Στην ανοδική ανάλυση οι αλγόριθμοι LR(1) είναι επαρκείς για την ανάλυση όλων των ντεντερμινιστικών Γ.Χ.Σ.

Τέλος ενότητας

Επεξεργασία: Εμμανουέλα Στάχτιαρη
Θεσσαλονίκη, 23/07/2014