- Parse trees are useful if further (or more detailed) processing and checking needs to be done on structured representation of the source language.

  The most 'natural' grammar may not be the most 'parsable' grammar. A top-down parser might like to save a left-branching parse tree but use a right-branching grammar (due to e.g. easier construction of semantics in the next phase)

  A translator might like to know more about internal structure of statements than the rules reveal, e.g., nesting of loops

  Order of attribute evaluation can be different than the order of rule

use in derivations (e.g., inherited attributes in a bottom-up parser or synthesized attributes in a top-down parser)

DECOUPLING OF PARSING FROM TRANSLATION

- building syntax (parse) trees:

```
mknode(op,left,right)

mkleaf(id,entry)

mkleaf(num,value)
```

```
E -> E + T    E.nptr:=mknode('+',E1.nptr,T.nptr)

E -> E - T    E.nptr:=mknode('-',E1.nptr,T.nptr)

E -> T        E.nptr:=T.nptr

T -> (E)      T.nptr:= E.nptr

T -> id       T.nptr:=mkleaf(ID,id.entry)

T -> num      T.nptr:=mkleaf(NUM,num.val)
```

- Synthesized attributes fit nicely with bottom-up parsing (and inherited attributes with top-down)

- SYNTHESIZED ATTRIBUTES IN A TOP-DOWN PARSER

$$
\begin{array}{ll}
A \rightarrow AY & \{A_0.a = g(A_1.a, Y.y)\} \\
A \rightarrow X & \{A.a = f(X.x)\}
\end{array}
$$

after removing left recursion

$$A \rightarrow X\{A.a = f(X.x)\}R$$

X must pass on it's attributes to R

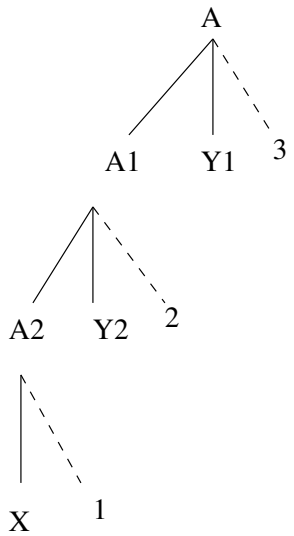
Two sets of attributes for dummy symbol R: R_i: inherited. R_S: synthesized

$$A \rightarrow X \{R.i = f(X.x)\}\ R\ \{A.a = R.s\}$$

$$R \rightarrow Y \{R_1.i = g(R_0.i, Y.y)\}\ R_1\ \{R_0.s = R_1.s\}$$

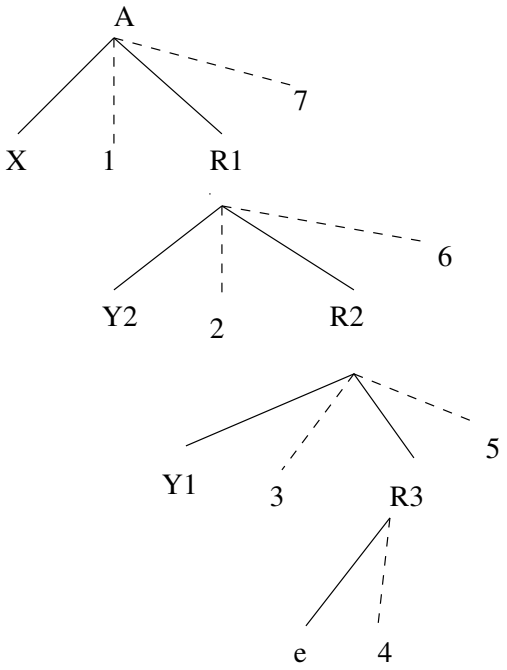
$$R \rightarrow \epsilon \{R.s = R.i\}$$

Attributes are evaluated in the same order



order: 1−2−3

A2.a=f(X.x)
A1.a=g(A2.a,Y2.y)
A.a=g(A1.a,Y1.y)



order: 1−2−3−4−5−6−7

R1.i=f(X.x)
R2.i=g(R1.i,Y2.y)
R3.i=g(R2.i,Y1.y)
R3.s=R3.i
R2.s=R3.s
R1.s=R2.s
A.a=R1.s

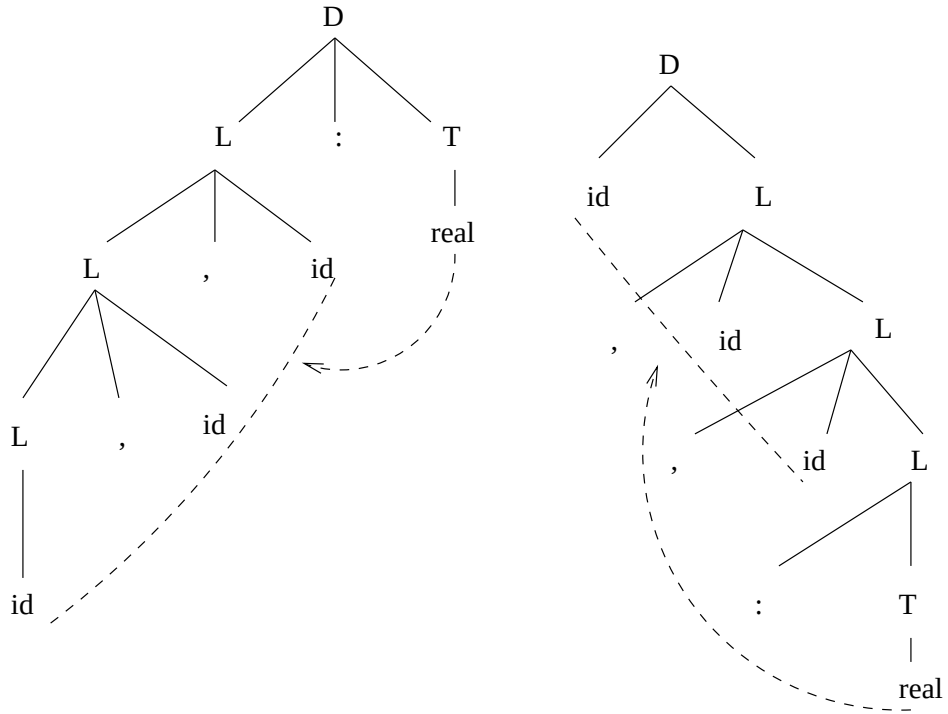
- INHERITED ATTRIBUTES IN A BOTTOM-UP PARSER

  Rewrite the grammar in such a way that attributes are synthesized

  `D` $\longrightarrow$ `L : T`

  `T` $\longrightarrow$ `integer | real | char`

  `L` $\longrightarrow$ `L , id | id`



D → id L

T → integer | real | char

L → , id L | : T