Grammaires attribuées

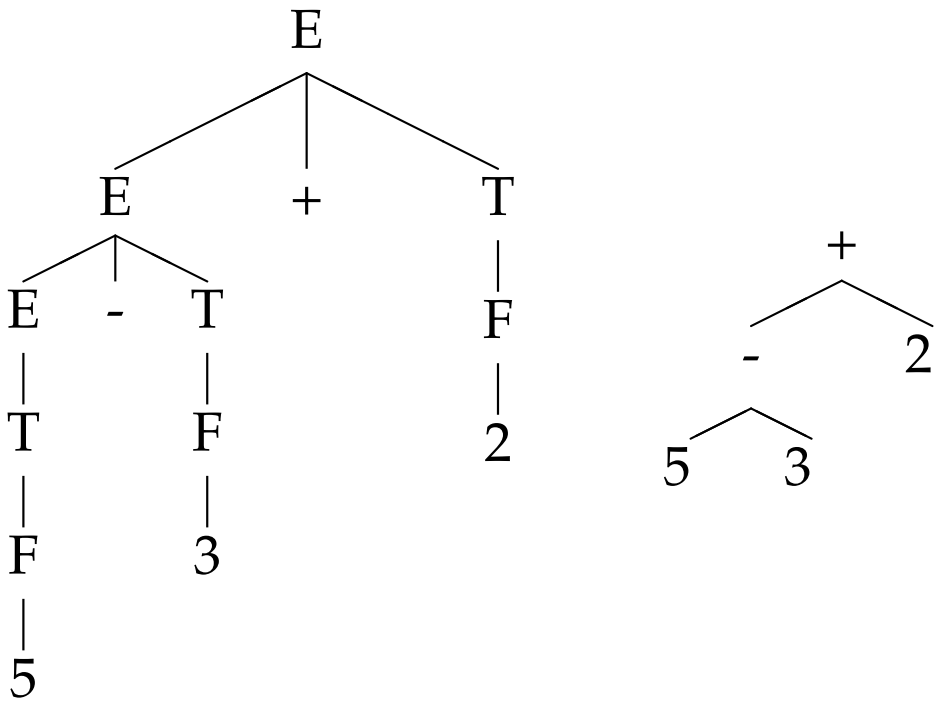
Alexis Nasr
Carlos Ramisch
Manon Scholivet
Franck Dary

Compilation – L3 Informatique
Département Informatique et Interactions
Aix Marseille Université

Arbre de dérivation v/s arbre abstrait

- L'arbre de dérivation produit par l'analyse syntaxique possède des nœuds superflus, qui ne véhiculent pas d'information.
- De plus, la mise au point d'une grammaire nécessite souvent l'introduction de règles dont le seul but est de simplifier l'analyse syntaxique.
- Un arbre abstrait constitue une interface plus naturelle entre l'analyse syntaxique et l'analyse sémantique, elle ne garde de la structure syntaxique que les parties nécessaires à l'analyse sémantique et à la production de code.
- L'arbre abstrait peut être construit :
  - lors de l'analyse syntaxique
  - lors d'un parcours de l'arbre de dérivation.

Arbre de dérivation v/s arbre abstrait



Grammaire attribuée

- Une grammaire attribuée est une grammaire avec des actions sémantiques attachées aux règles
- Les actions sémantiques sont des affectations qui manipulent des attributs
- Un attribut est une variable quelconque associée à une construction du langage décrit par la grammaire, par exemple :
  - le type d'une expression
  - la valeur d'une expression (p.ex. calculette)
  - la ligne du programme (pour débogage)
  - un noeud de l'arbre abstrait
- On associe les attributs aux symboles terminaux et non terminaux de la grammaire.
- Notations : $A.t$ est l'attribut t associé au symbole A.

Actions sémantiques

règle			action sémantique		
A	$\rightarrow$	$A\ a$	$A.n$	$=$	$A_1.n+1$
A	$\rightarrow$	ε	$A.n$	$=$	0

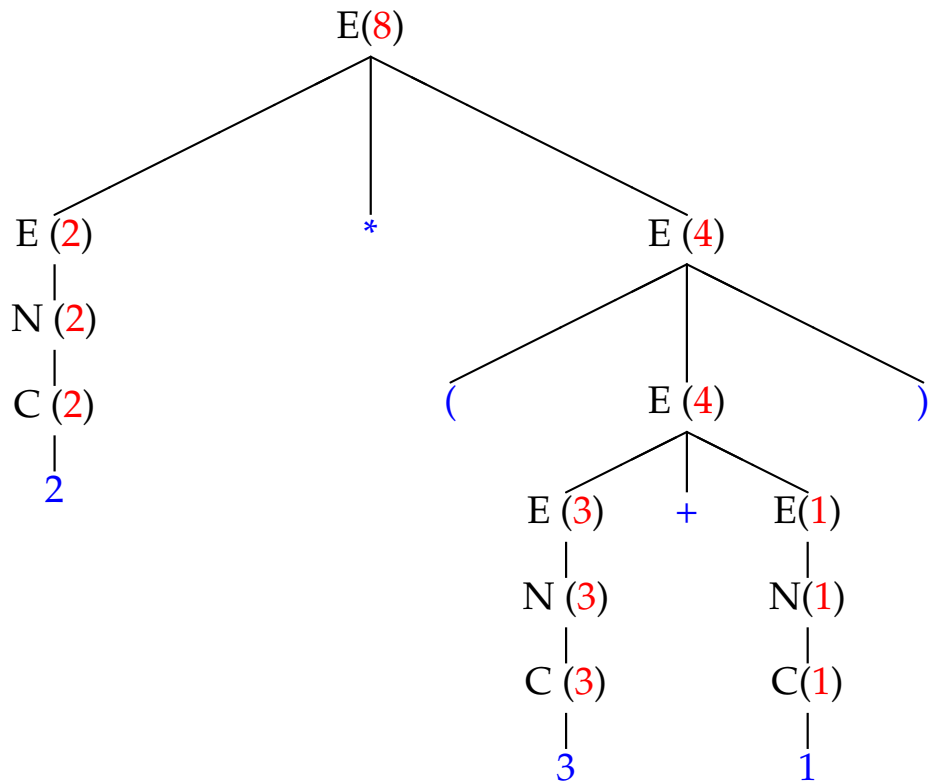
- L'attribut n est défini pour le symbole A
- $A.n$ est initialisé à 0 quand la feuille ε est créée lors de l'analyse
- À chaque fois qu'on voit un a, la valeur de $A.n$ est incrémentée de 1 par rapport à sa valeur en partie droite de production $A_1.n$
- La valeur de $A.n$ à la racine de l'arbre est le nombre de a du mot $\Longrightarrow$ cet attribut "compte" le nombre de a qui se suivent
- Convention : quand une règle contient plusieurs instances du même symbole, les instances en partie droite sont numérotées 1, 2, ... pour distinguer les symboles. Le symbole en partie gauche est celui sans indice. Il s'agit bien du même symbole A et non pas de symboles différents.

Exemple de grammaire attribuée : calculette

règle			action sémantique		
E	$\rightarrow$	E + E	$E.t$	=	$E_1.t + E_2.t$
E	$\rightarrow$	E * E	$E.t$	=	$E_1.t * E_2.t$
E	$\rightarrow$	(E)	$E.t$	=	$E_1.t$
E	$\rightarrow$	N	$E.t$	=	$N.t$
N	$\rightarrow$	N C	$N.t$	=	$10 * N_1.t + C.t$
N	$\rightarrow$	C	$N.t$	=	$C.t$
C	$\rightarrow$	0	$C.t$	=	0
C	$\rightarrow$	1	$C.t$	=	1
C	$\rightarrow$	2	$C.t$	=	2
	...			...	
C	$\rightarrow$	9	$C.t$	=	9

Attention : E_1 et E_2 sont des instances du non terminal E et non des symboles différents. Il est nécessaire de les distinguer afin de distinguer les traductions qui leurs sont associées.

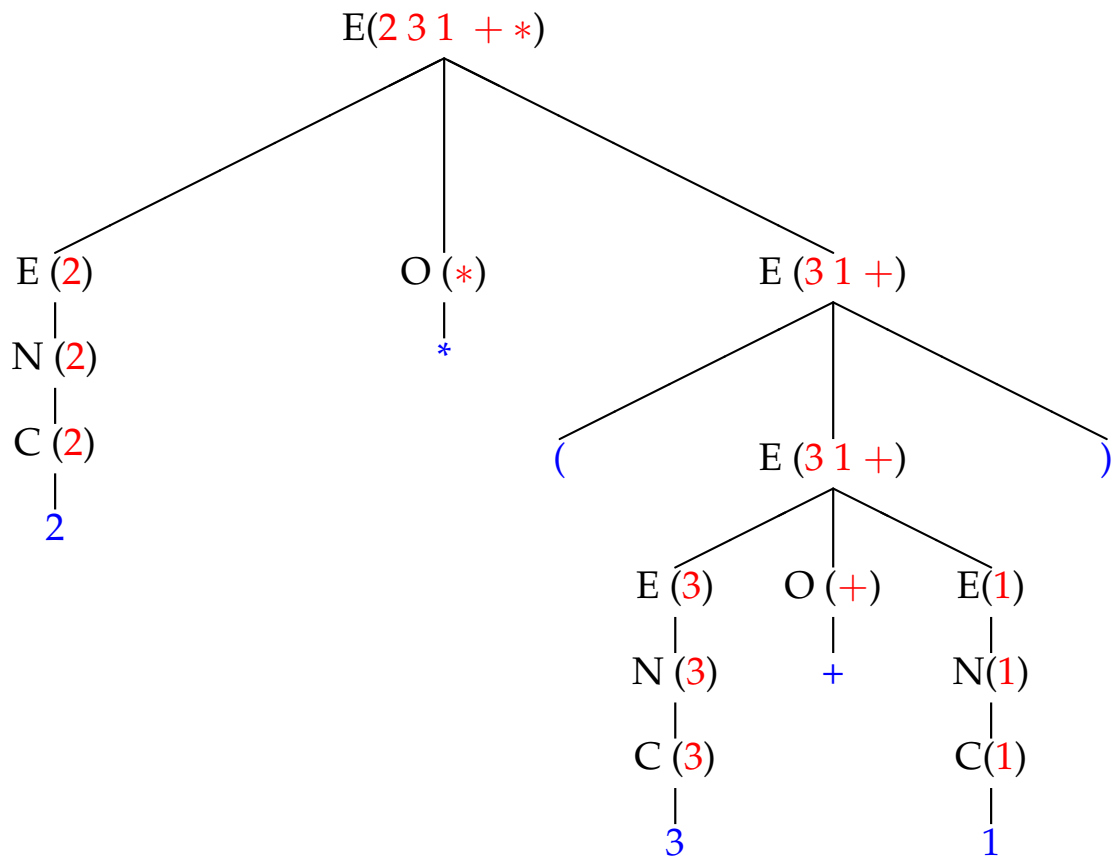
Calculette



Exemple de grammaire attribuée : notation post-fixée

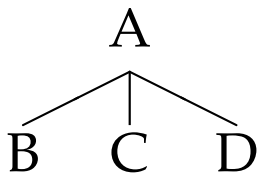
règle			action sémantique		
E	$\rightarrow$	E O E	$E.t$	$=$	$E_1.t \mid\mid E_2.t \mid\mid O.t$
E	$\rightarrow$	(E)	$E.t$	$=$	$E_1.t$
E	$\rightarrow$	N	$E.t$	$=$	$N.t$
O	$\rightarrow$	$+$	$O.t$	$=$	$+$
O	$\rightarrow$	$-$	$O.t$	$=$	$-$
N	$\rightarrow$	C N	$N.t$	$=$	$C.t \mid\mid N_1.t$
N	$\rightarrow$	C	$N.t$	$=$	$C.t$
C	$\rightarrow$	0	$C.t$	$=$	0
C	$\rightarrow$	1	$C.t$	$=$	1
C	$\rightarrow$	2	$C.t$	$=$	2
	$\dots$			$\dots$	
C	$\rightarrow$	9	$C.t$	$=$	9

Notation post-fixée



Attributs synthétisés

Un attribut est dit synthétisé si sa valeur au niveau d'un nœud A d'un arbre d'analyse est déterminée par les valeurs de cet attribut au niveau des fils de A et de A lui même.



$$A.t = f(B.t, C.t, D.t)$$

- Les attributs synthétisés peuvent être évalués au cours d'un parcours ascendant de l'arbre de dérivation.
- Un tel parcours peut être effectué simultanément à la construction de l'arbre (lors de l'analyse syntaxique).
- Dans l'exemple, $B.t$, $C.t$ et $D.t$ doivent être calculés avant de calculer $A.t$.
- La grammaire est dite *S-attribuée*.

Exemple S-attribué : calculette

- La grammaire attribuée évalue la valeur des expressions
- La valeur finale se retrouve à la racine de l'arbre de dérivation
- Implémente précédences et associativité gauche
- La grammaire est S-attribué (uniquement attributs synthétisés)
- On peut évaluer l'expression en même temps qu'on la dérive

règle			action sémantique		
E	$\rightarrow$	E + T	$E.v$	=	$E_1.v + T.v$
E	$\rightarrow$	E - T	$E.v$	=	$E_1.v - T.v$
E	$\rightarrow$	T	$E.v$	=	$T.v$
T	$\rightarrow$	T * F	$T.v$	=	$T_1.v \times F.v$
T	$\rightarrow$	T / F	$T.v$	=	$T_1.v \div F.v$
T	$\rightarrow$	F	$T.v$	=	$F.v$
F	$\rightarrow$	(E)	$F.v$	=	$E.v$
F	$\rightarrow$	n	$F.v$	=	n

Test 1 : précédence des opérateurs

```
5 + 3 * 2
```

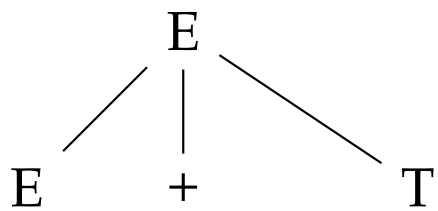
$5 + (3 \times 2) = 11$

E

Test 1 : précédence des opérateurs

5 + 3 * 2

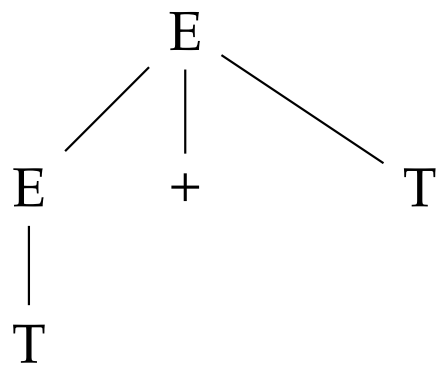
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

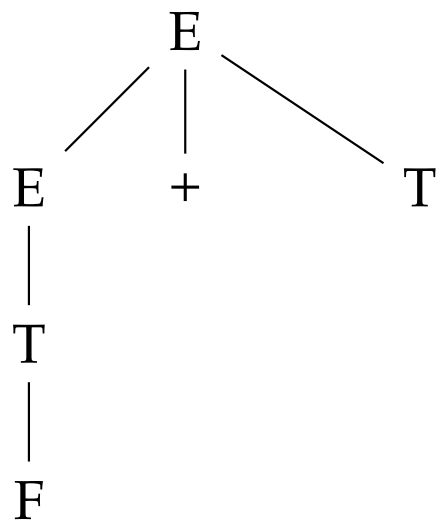
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

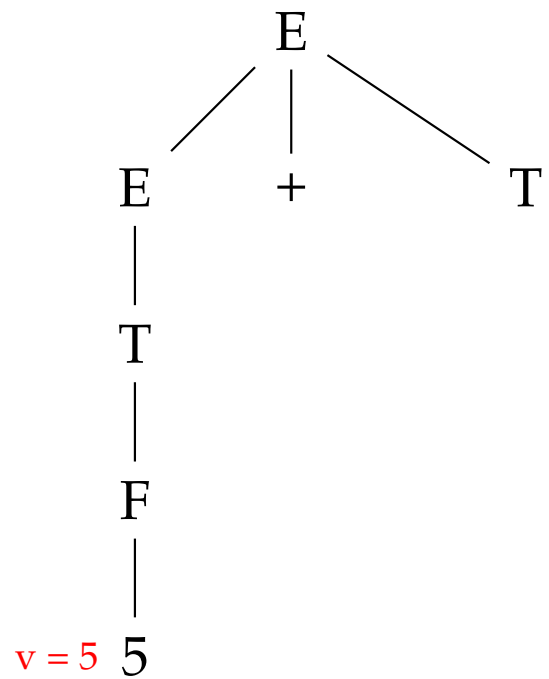
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

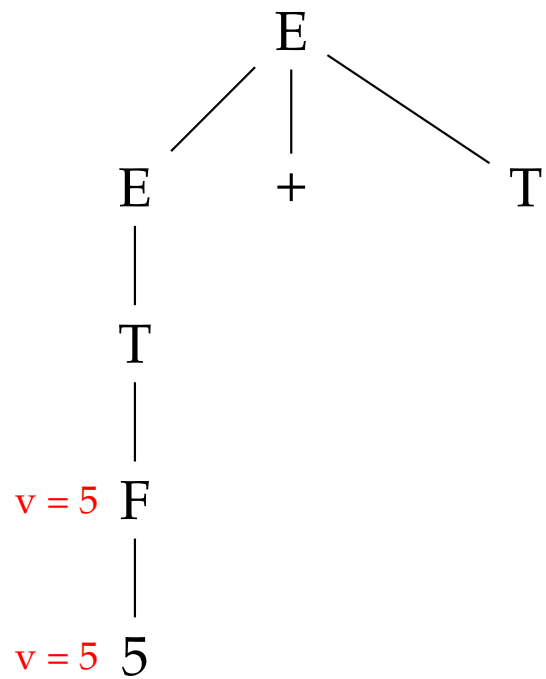
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

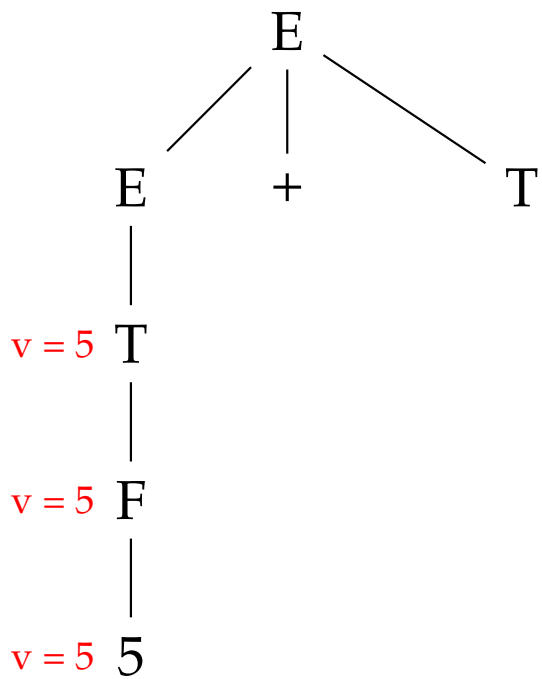
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

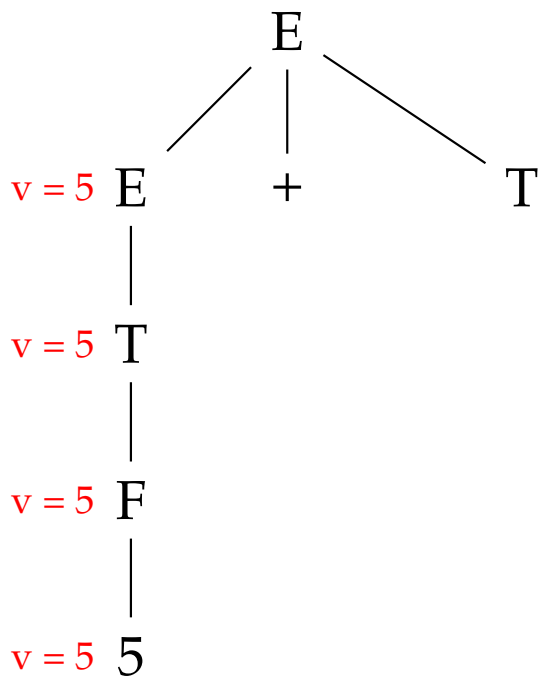
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

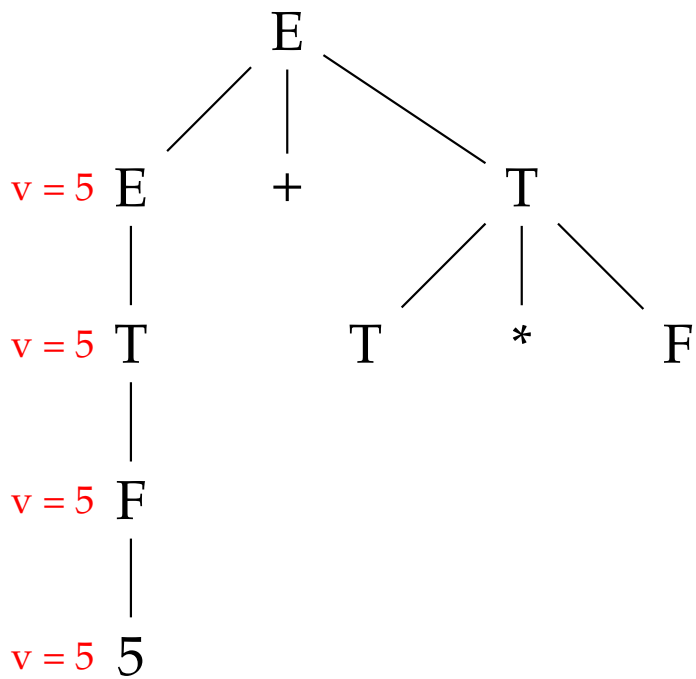
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

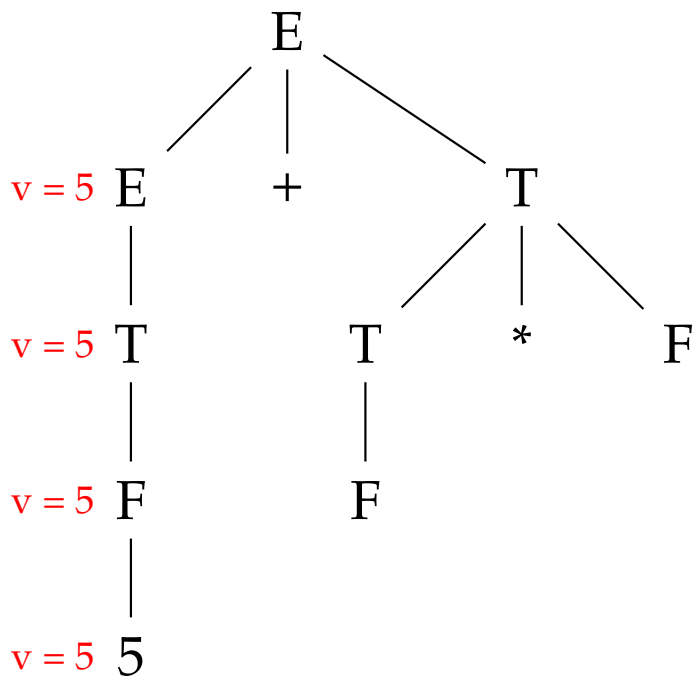
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

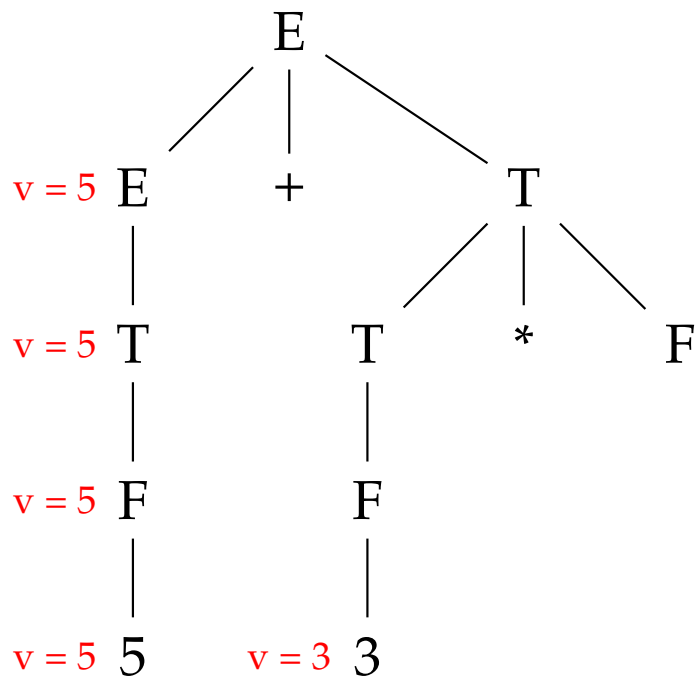
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

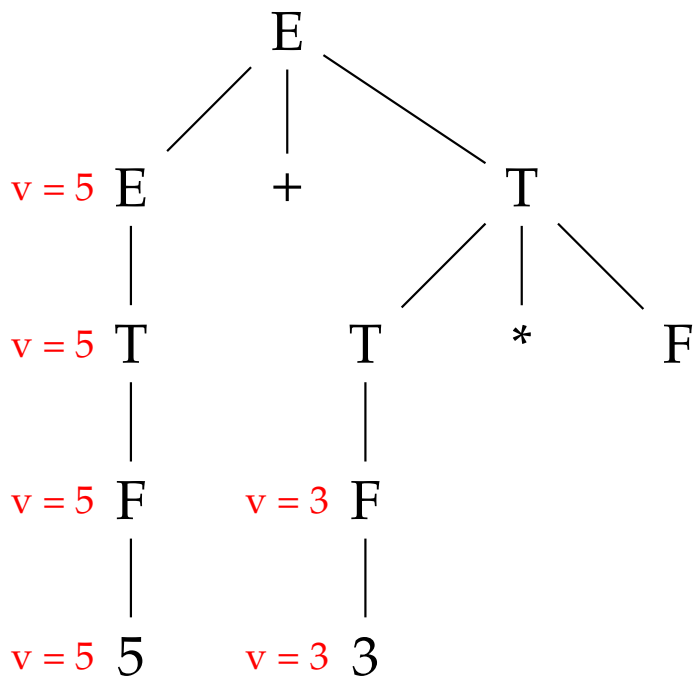
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

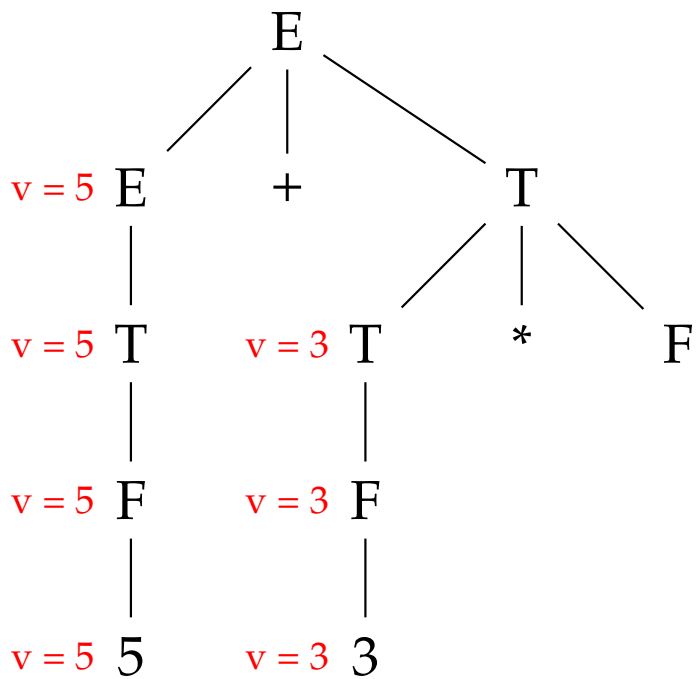
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

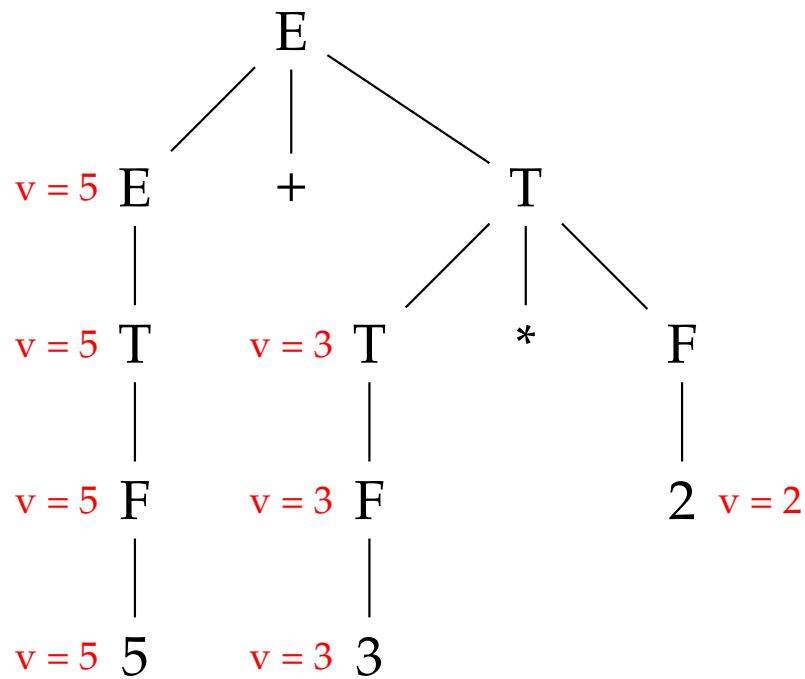
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

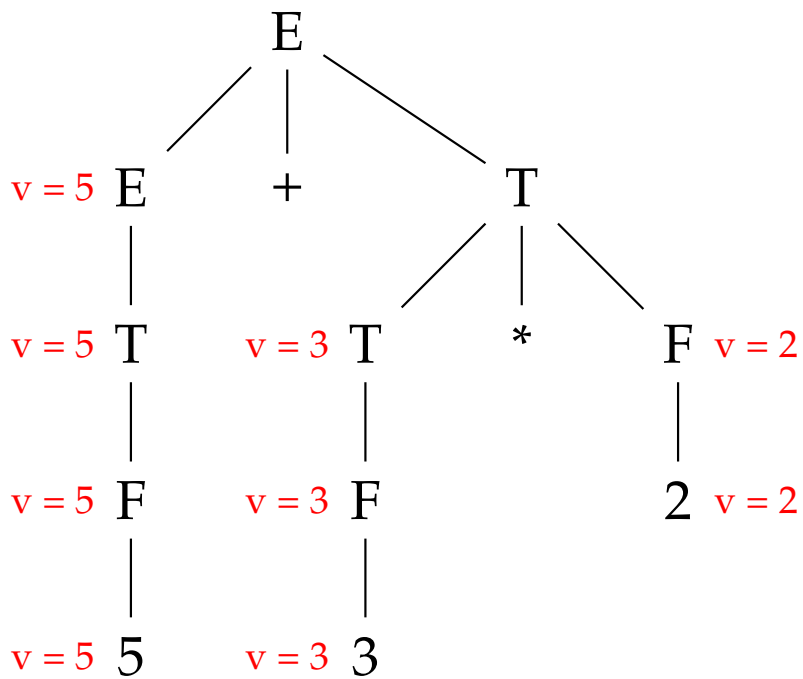
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

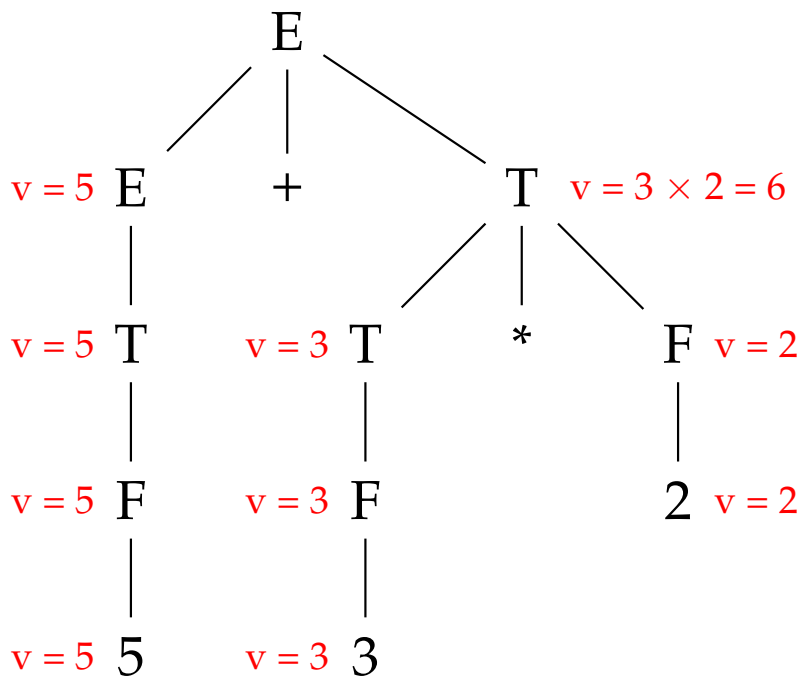
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

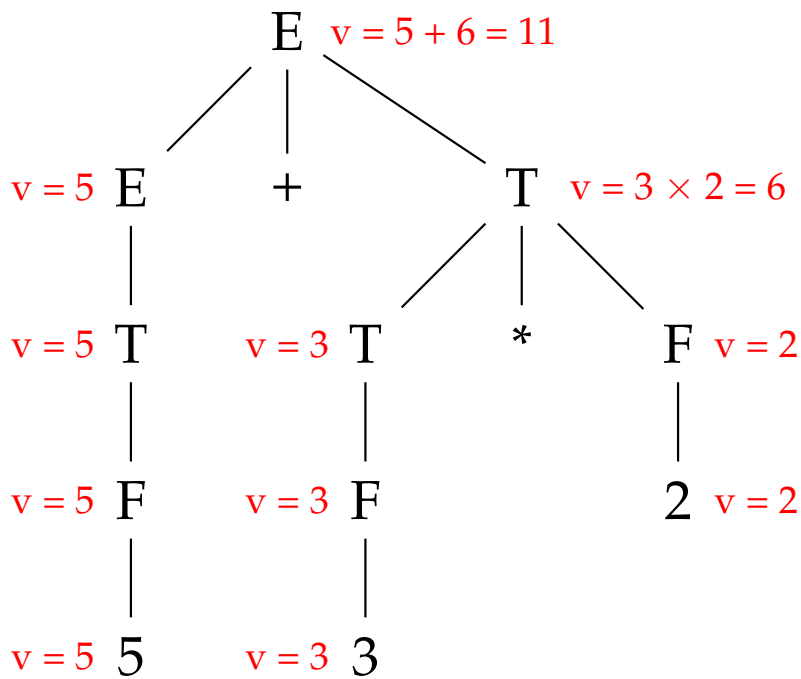
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

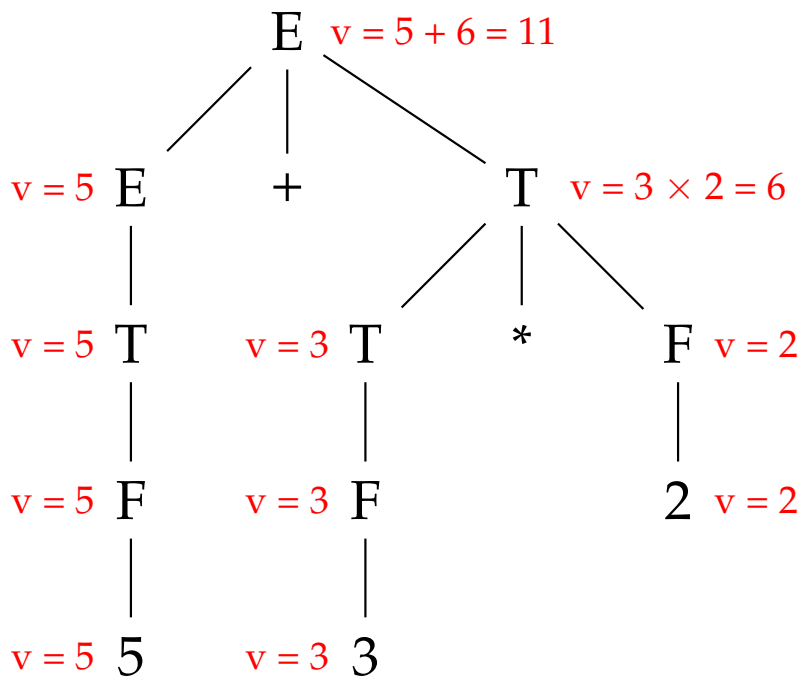
$5 + (3 \times 2) = 11$



Test 1 : précédence des opérateurs

5 + 3 * 2

$5 + (3 \times 2) = 11$



OK

Test 2 : associativité à gauche

5 - 3 + 2

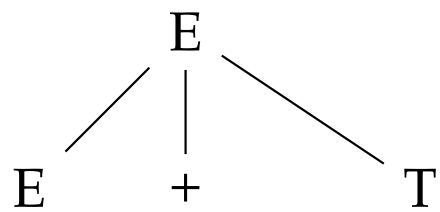
$$(5 - 3) + 2 = 4$$

E

Test 2 : associativité à gauche

5 - 3 + 2

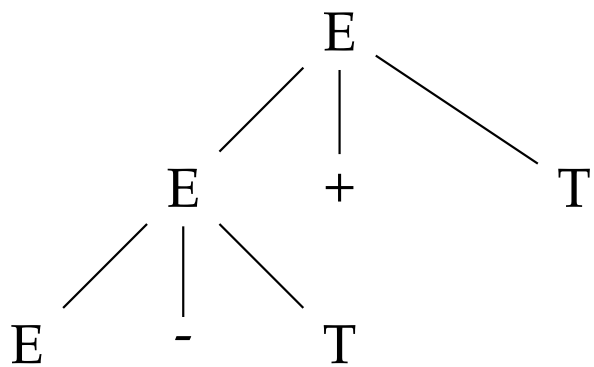
$$(5 - 3) + 2 = 4$$



Test 2 : associativité à gauche

5 - 3 + 2

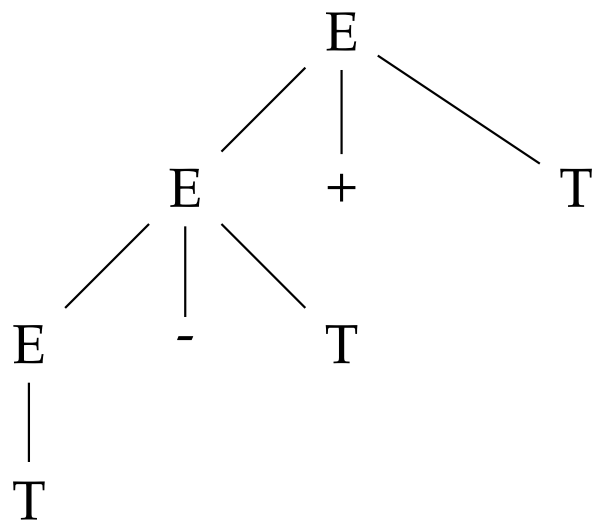
$$(5-3)+2=4$$



Test 2 : associativité à gauche

5 - 3 + 2

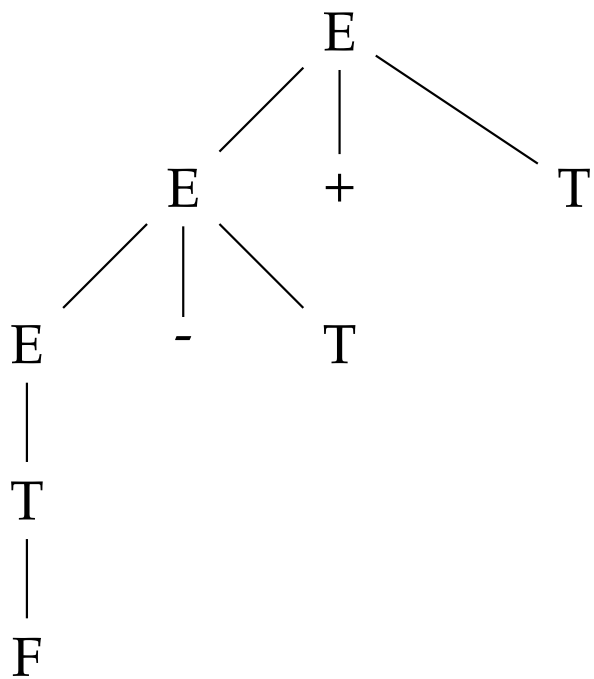
$$(5-3)+2=4$$



Test 2 : associativité à gauche

5 - 3 + 2

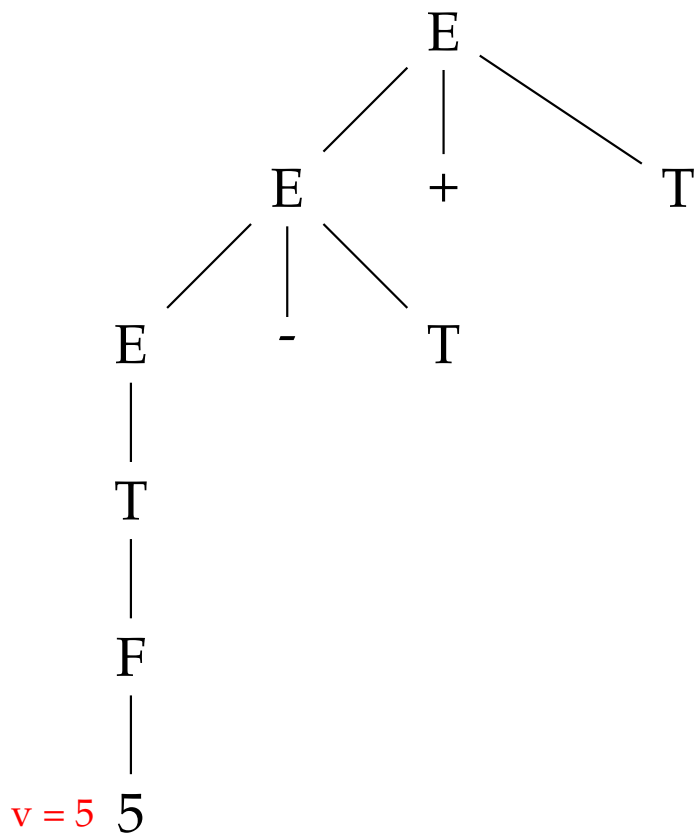
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

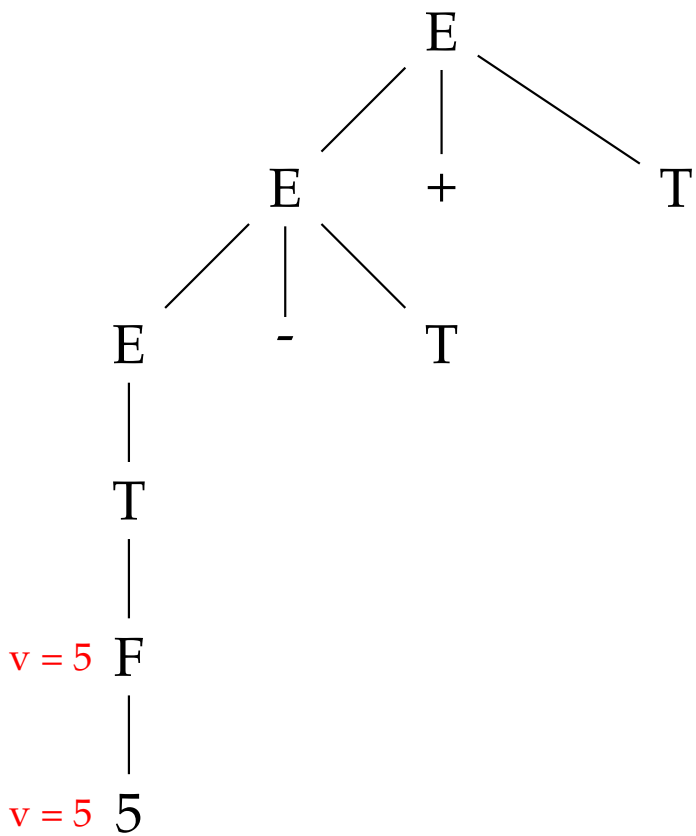
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

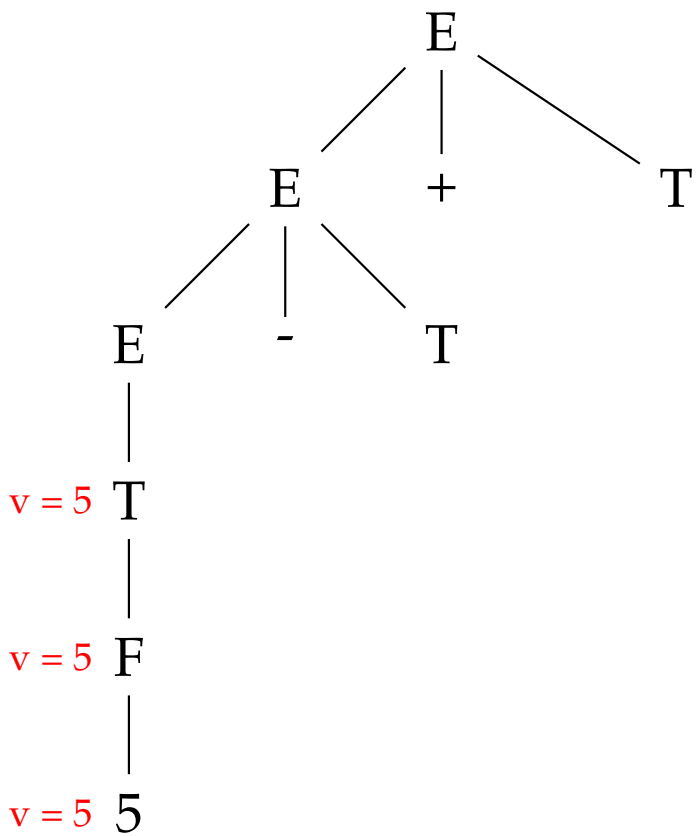
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

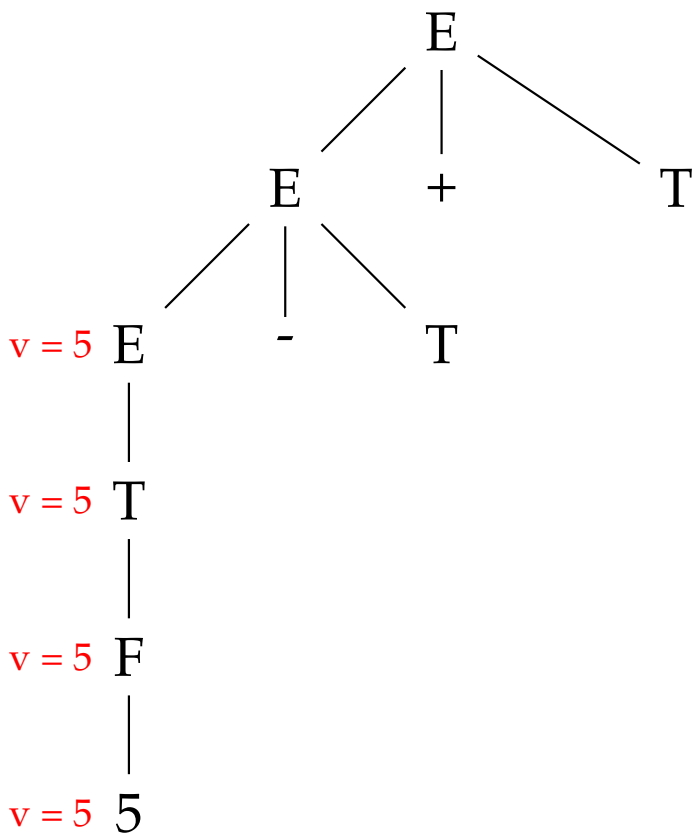
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

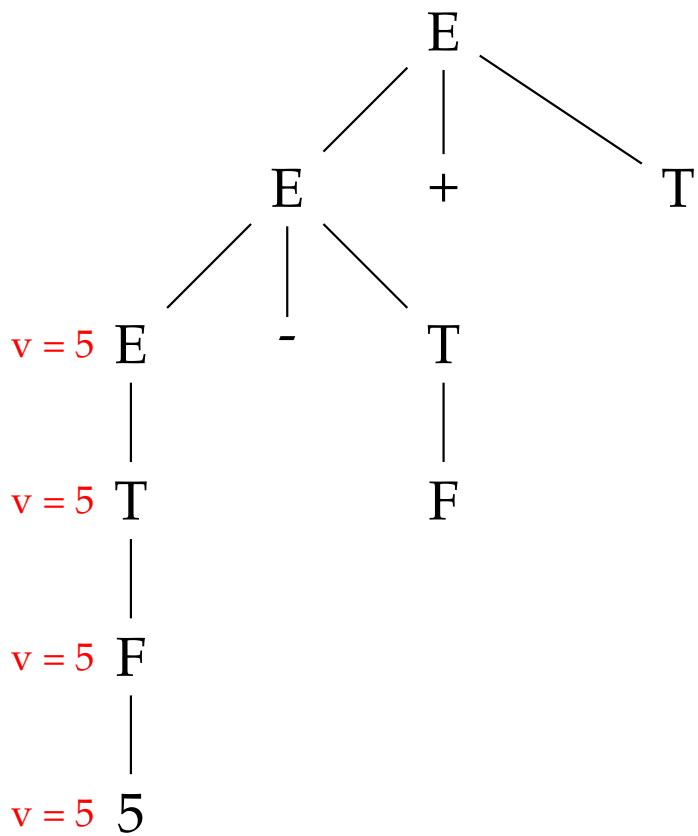
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

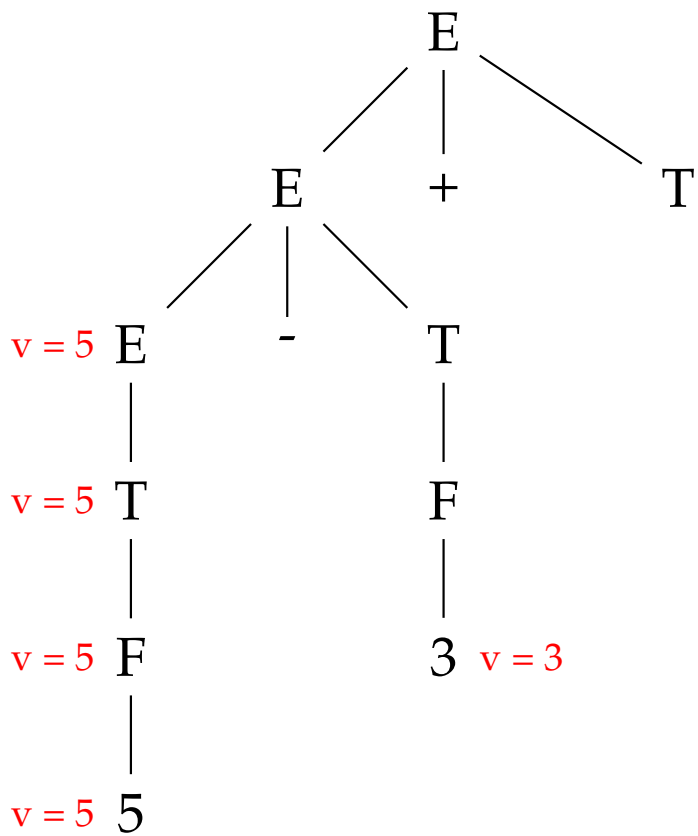
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

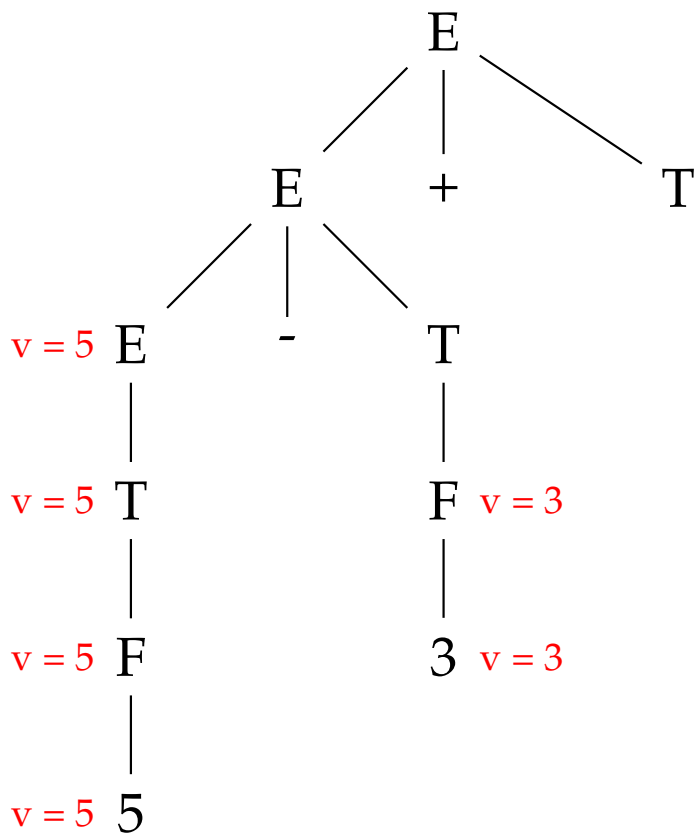
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

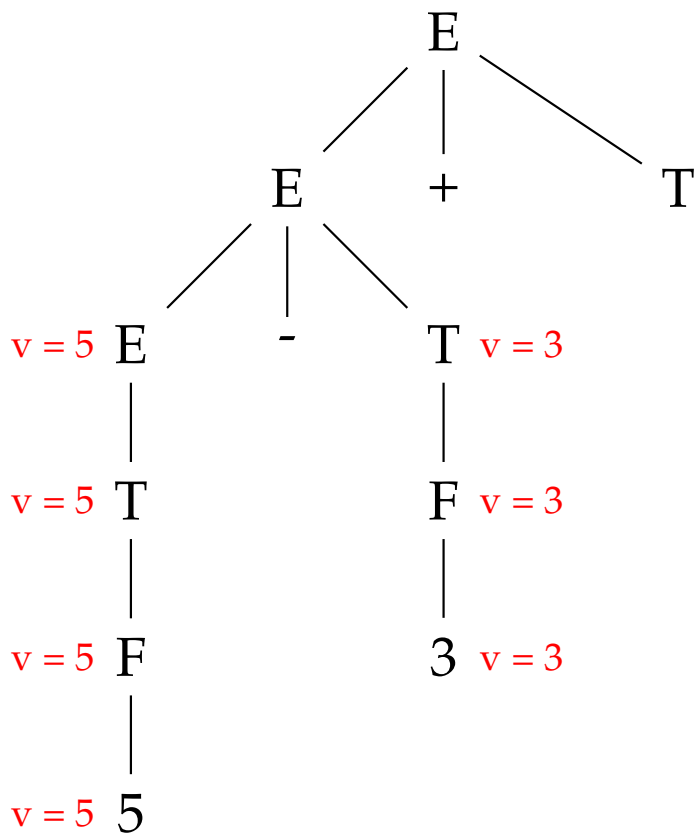
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

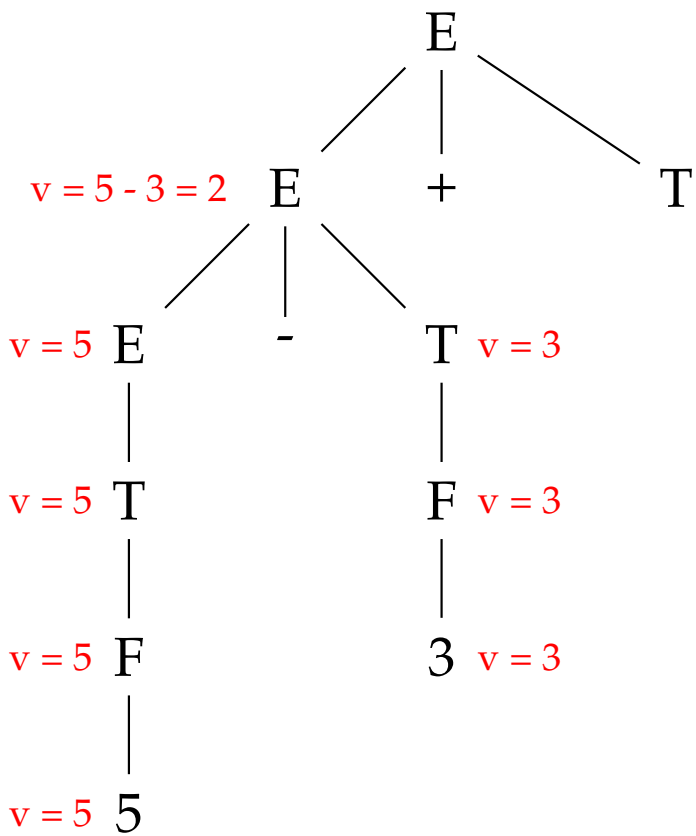
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

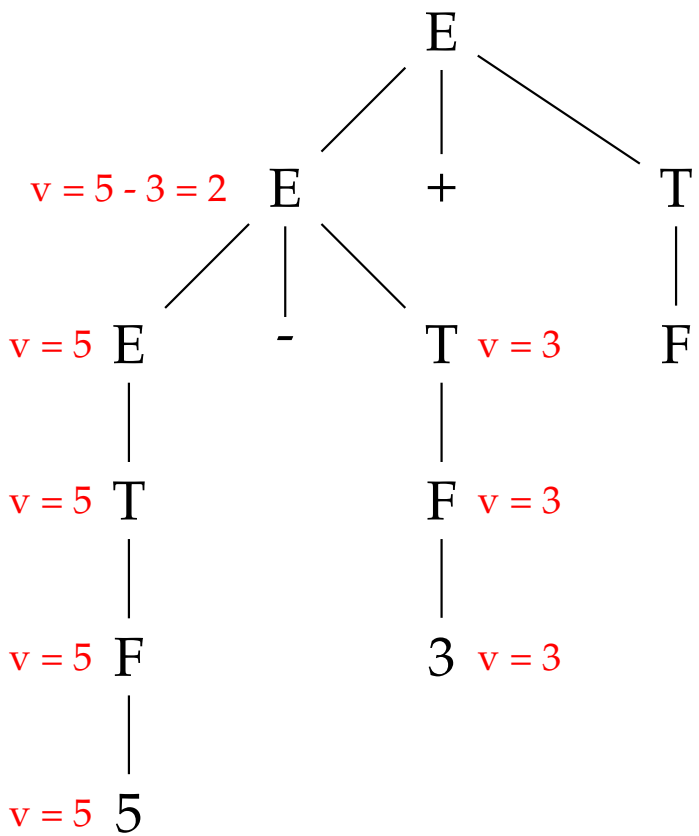
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

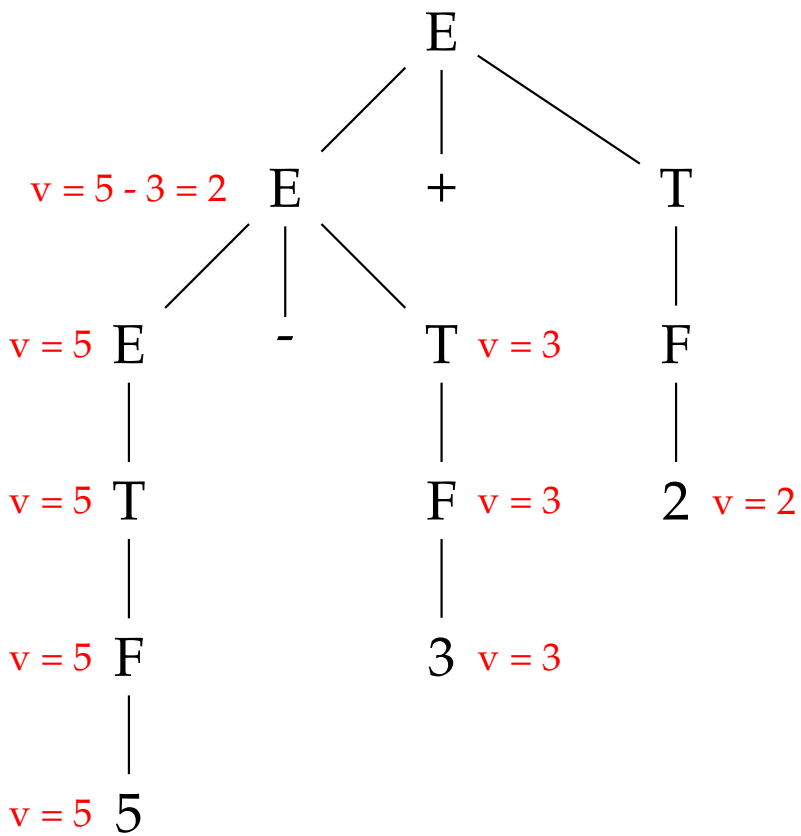
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

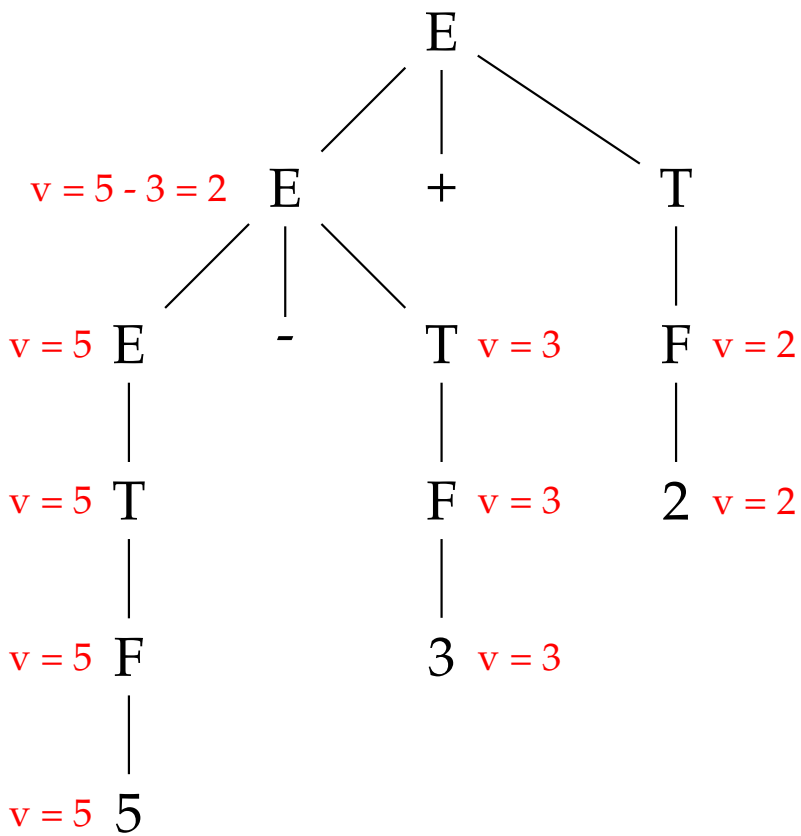
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

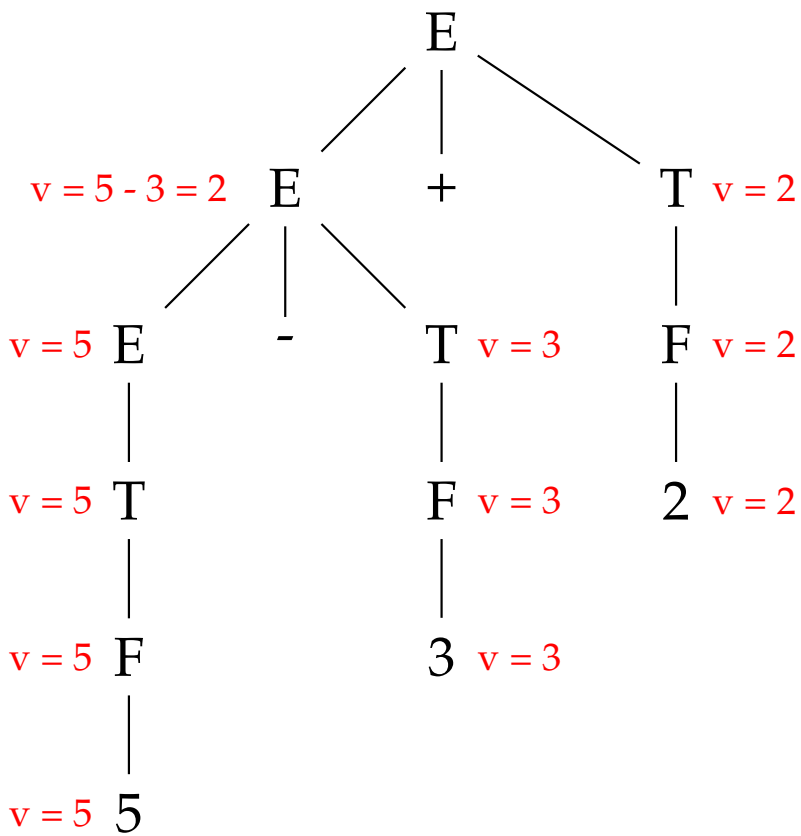
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

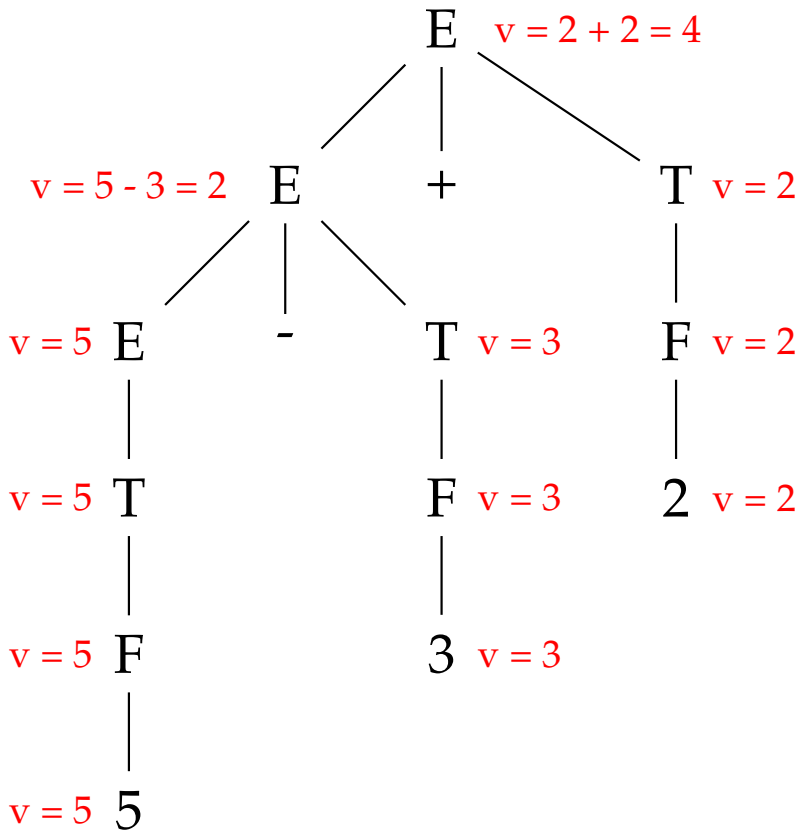
$(5-3)+2=4$



Test 2 : associativité à gauche

5 - 3 + 2

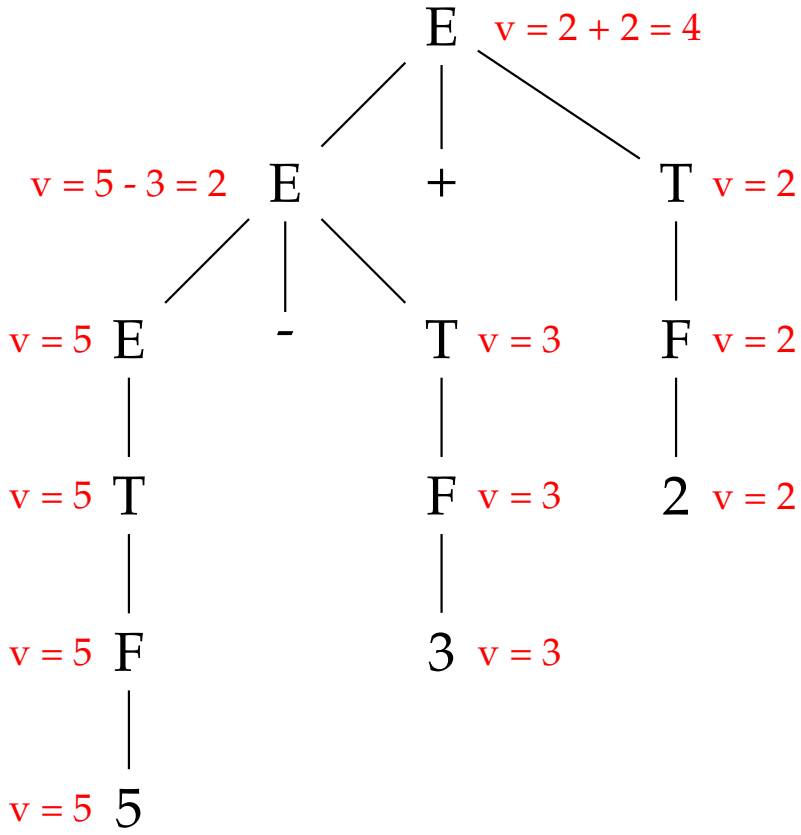
$(5 - 3) + 2 = 4$



Test 2 : associativité à gauche

5 - 3 + 2

$$(5-3)+2=4$$



OK

Attributs hérités

A l'inverse, un attribut est hérité s'il dépend de son père.

- Problème : éviter les attributions circulaires !

$$A \rightarrow B \qquad \begin{array}{lcl} A.s & = & B.h \\ B.h & = & A.s + 1 \end{array}$$

$$A.s \circlearrowright B.h$$

- On se restreint généralement aux attributions
  - synthétisées,
  - héritées d'un frère *gauche*,
  - héritées du père, mais sans cycle.

  Les grammaires sont alors appelées grammaires *L-attribuées*.
- On peut alors calculer les attributs selon un parcours en profondeur gauche de l'arbre.

Exemple L-attribué : typographie

- Langage de composition de formules mathématique.
- Exemple de formule : $a_0 + a_1 x^1 + a_2 x^2 + a_3 x^3$
- On se limitera ici au traitement des indices et des indices d'indices.
- Chaque lettre est écrite dans une boîte et les boîtes sont combinées pour produire de nouvelles boîtes

Grammaire des boîtes

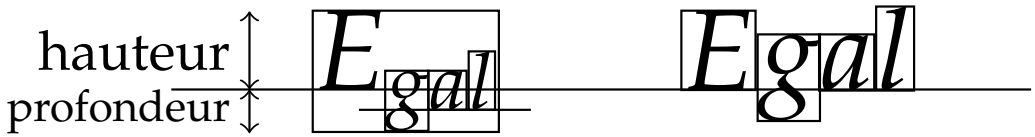
- Une boîte peut être constituée de :
  - deux boîtes juxtaposées
  - une boîte et une boîte en indice ; la seconde apparaît dans une taille inférieure, au-dessous et à droite de la première
  - une boîte entre accolades pour regrouper des boîtes et des indices
  - un caractère
- Chaque mode de combinaison de boîtes est représenté par une règle de grammaire :

$$B \to BB \mid B \ \textbf{sub} \ B \mid \{B\} \mid \text{car}$$

- On considère que l'indiçage et la juxtaposition sont associatifs à droite et que l'indiçage est prioritaire sur la juxtaposition.
- Exemples :
  - `x sub i` correspond à x_i
  - `x sub i + y sub i` correspond à $x_i + y_i$
  - `x sub {i + j}` correspond à x_{i+j}

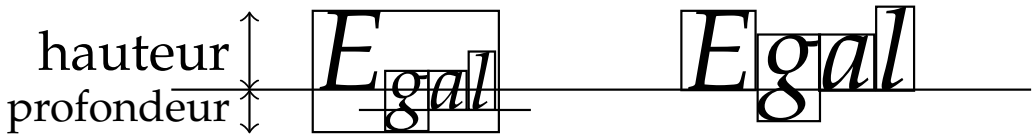
Composition

Pour composer la formule, il faut calculer la taille, la hauteur et la profondeur des caractères



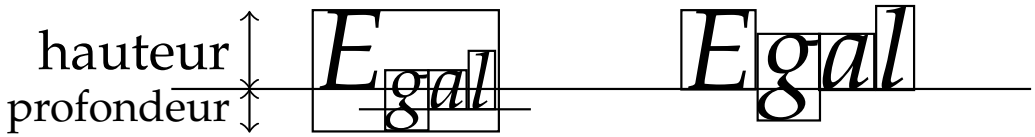
- la taille d'un caractère est la distance entre son point le plus haut et son point le plus bas
- chaque boîte a une ligne de pied qui est une ligne imaginaire sur lesquels sont "posés" les caractères
- hauteur d'une boîte : distance qui sépare le sommet de la boîte à sa ligne de pied
- profondeur d'une boîte : distance qui sépare le bas de la boîte à la ligne de pied

Quelques règles de composition



- règles de juxtaposition
  - deux sous-boîtes d'une boîte héritent d'elle la même taille.
  - la hauteur de la boîte est le maximum des hauteurs des sous-boîtes
  - la profondeur de la boîte est le maximum des profondeurs des sous-boîtes
- règles de la mise en indice
  - la taille d'une boîte en indice est 70% celle de sa mère
  - la ligne de pied d'une boîte en indice est descendue de 25% de la taille de sa mère

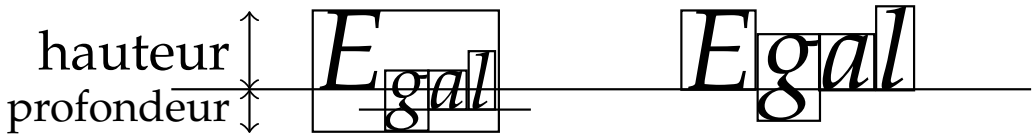
Attributs



On définit trois attributs pour le symbole B :

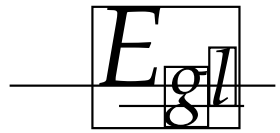
- la taille ($B.t$), hérité,
- la hauteur ($B.h$), synthétisé,
- la profondeur ($B.p$), synthétisé.

Exemple : typographie

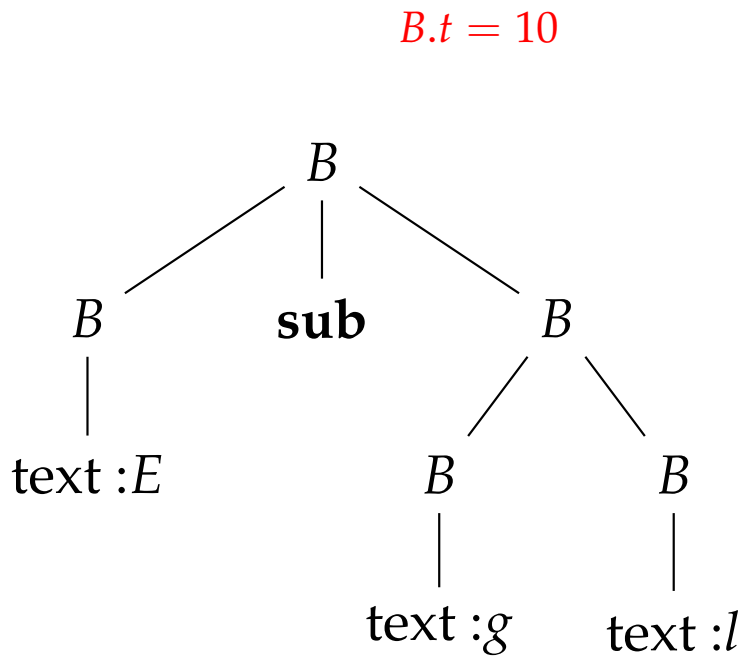


Productions	Règles sémantiques
$S \to B$	$B.t = 10$
$B \to BB$	$B_1.t = B_2.t = B.t$ $B.h = \max(B_1.h, B_2.h)$ $B.p = \max(B_1.p, B_2.p)$
$B \to B$ **sub** B	$B_1.t = B.t$ $B_2.t = \frac{7}{10} B.t$ $B.h = \max(B_1.h, B_2.h - \frac{1}{4} B.t)$ (la ligne est abaissée) $B.p = \max(B_1.p, B_2.p + \frac{1}{4} B.t)$
$B \to \text{car}$	$B.h = \text{hauteur}(B.t, \text{car})$ $B.p = \text{profondeur}(B.t, \text{car})$
$B \to \{B\}$	$B_1.t = B.t, B.h = B_1.t, B.p = B_1.p$

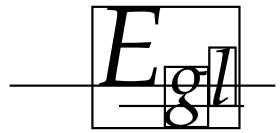
Exemple : typographie



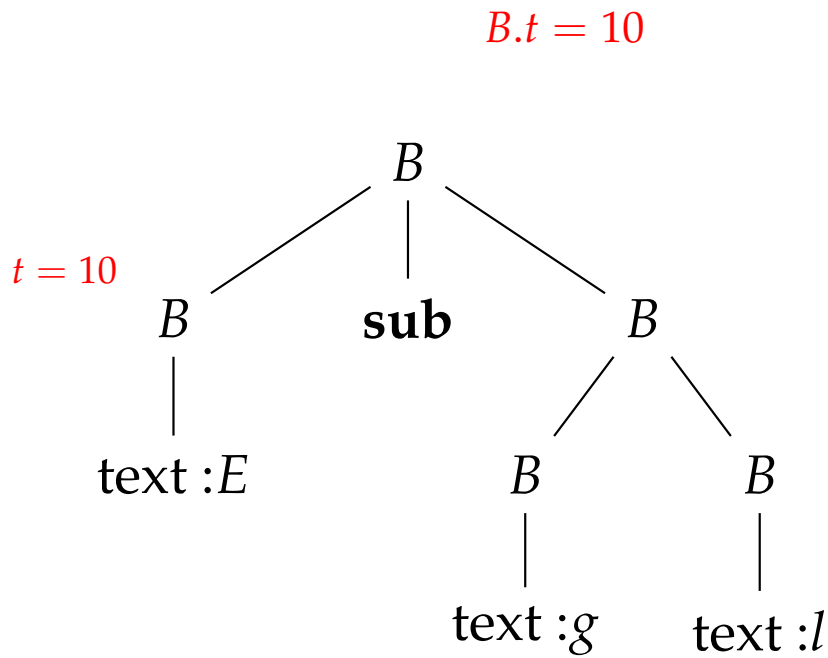
Parcours de l'arbre en profondeur, de gauche à droite :



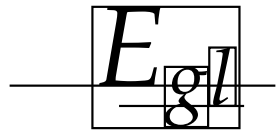
Exemple : typographie



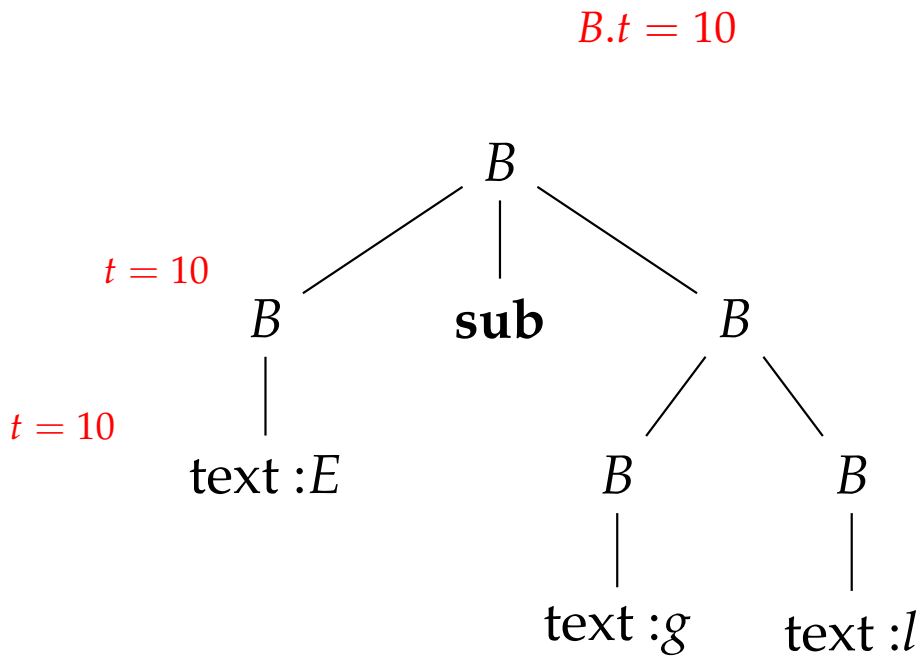
Parcours de l'arbre en profondeur, de gauche à droite :



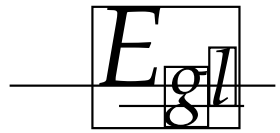
Exemple : typographie



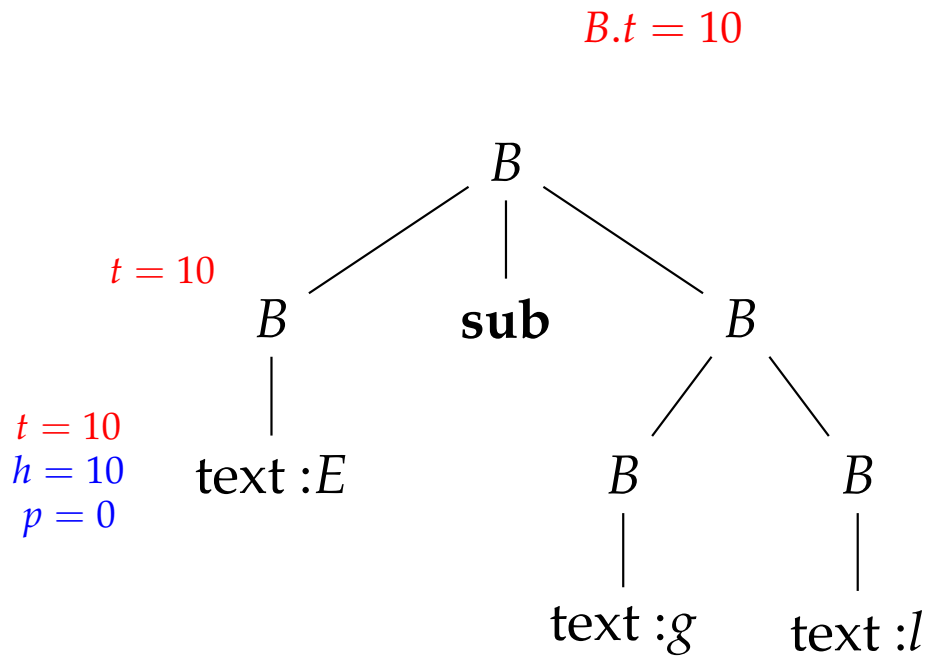
Parcours de l'arbre en profondeur, de gauche à droite :



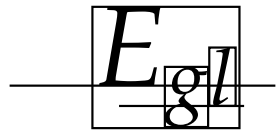
Exemple : typographie



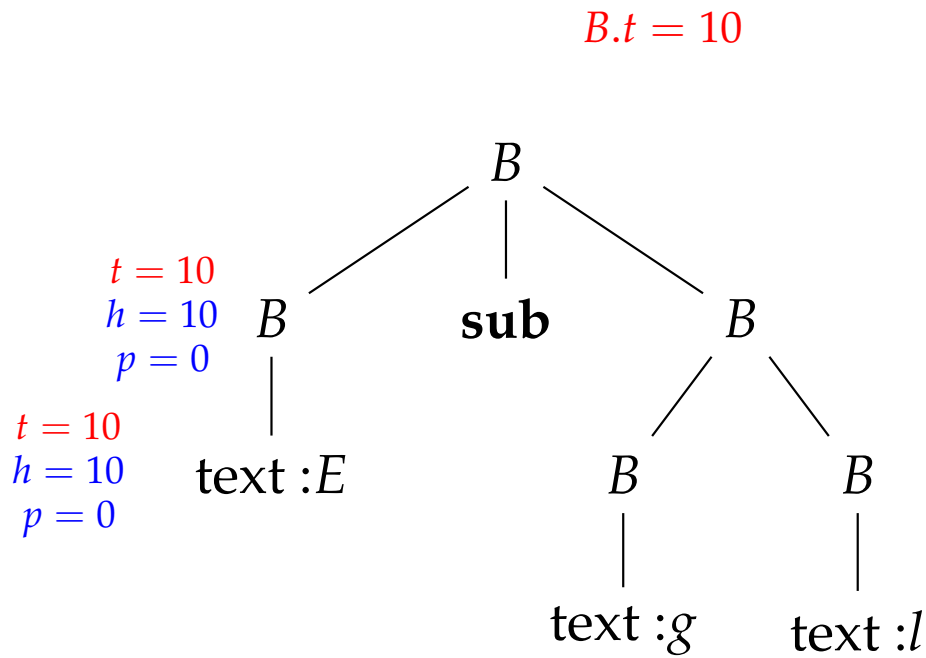
Parcours de l'arbre en profondeur, de gauche à droite :



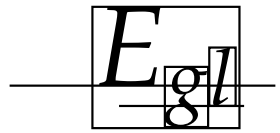
Exemple : typographie



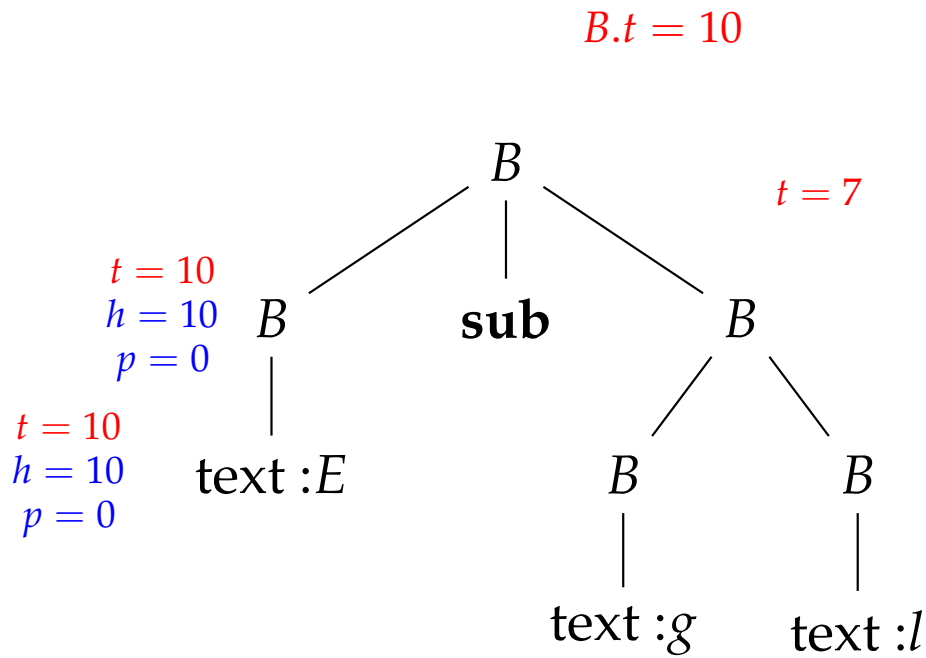
Parcours de l'arbre en profondeur, de gauche à droite :



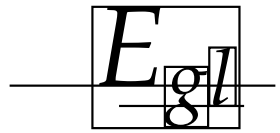
Exemple : typographie



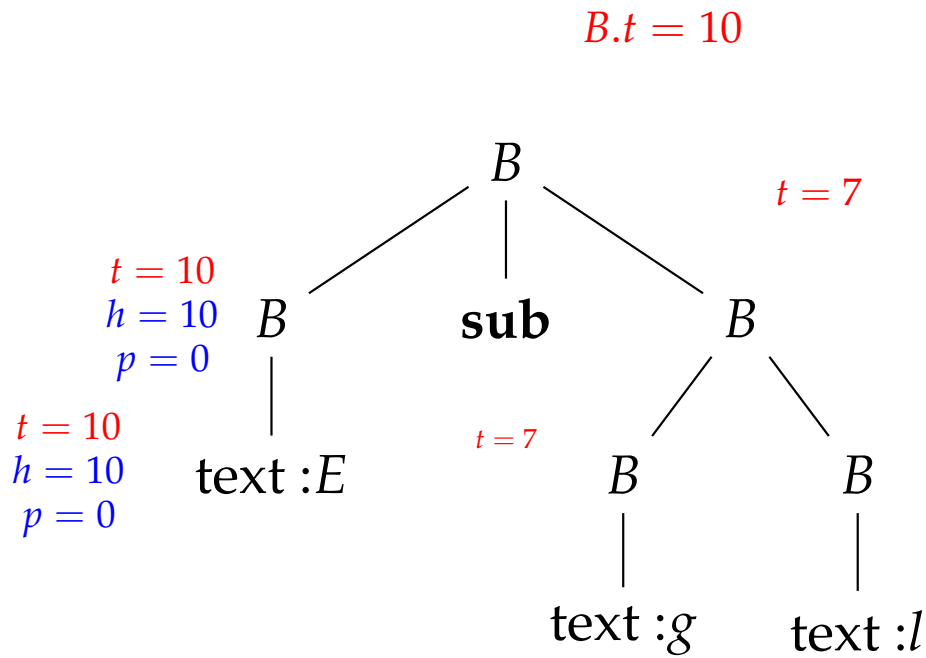
Parcours de l'arbre en profondeur, de gauche à droite :



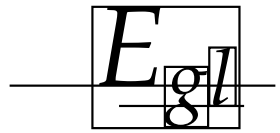
Exemple : typographie



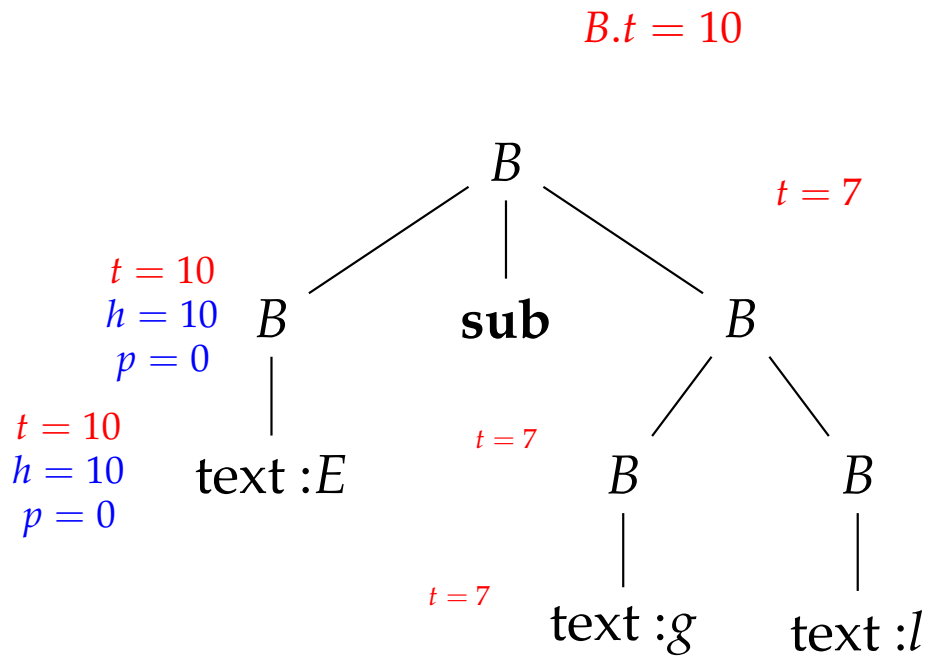
Parcours de l'arbre en profondeur, de gauche à droite :



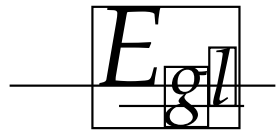
Exemple : typographie



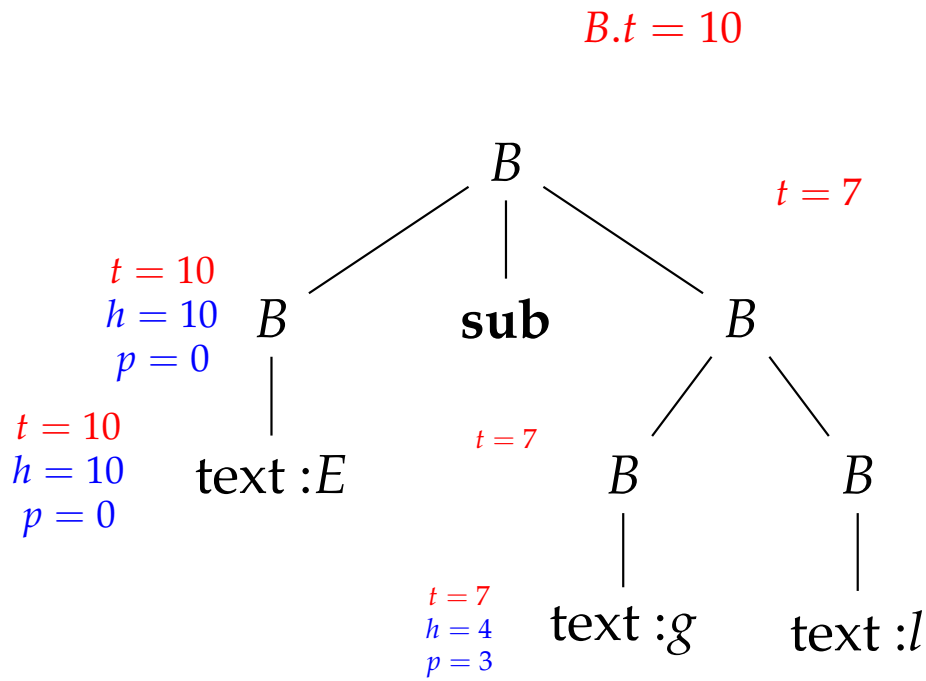
Parcours de l'arbre en profondeur, de gauche à droite :



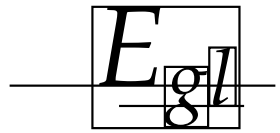
Exemple : typographie



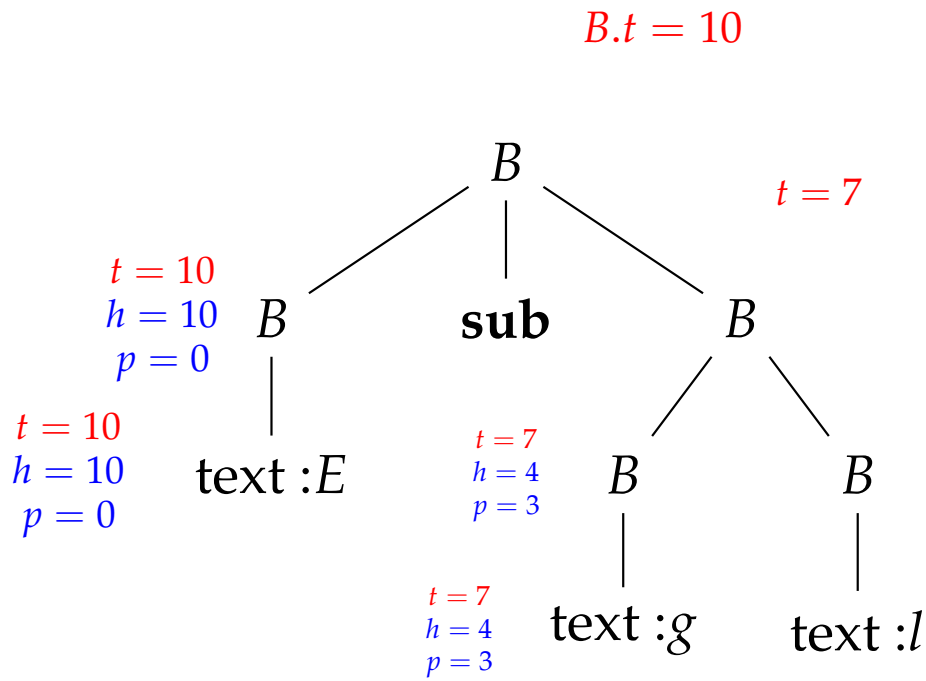
Parcours de l'arbre en profondeur, de gauche à droite :



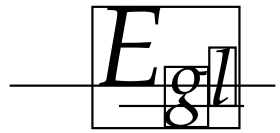
Exemple : typographie



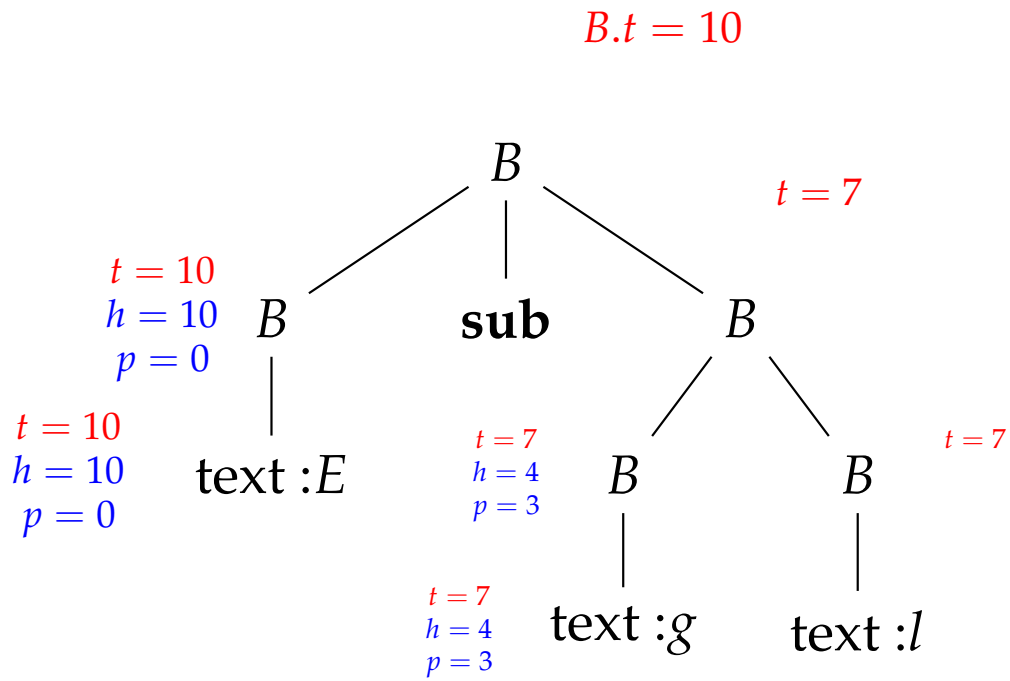
Parcours de l'arbre en profondeur, de gauche à droite :



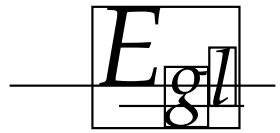
Exemple : typographie



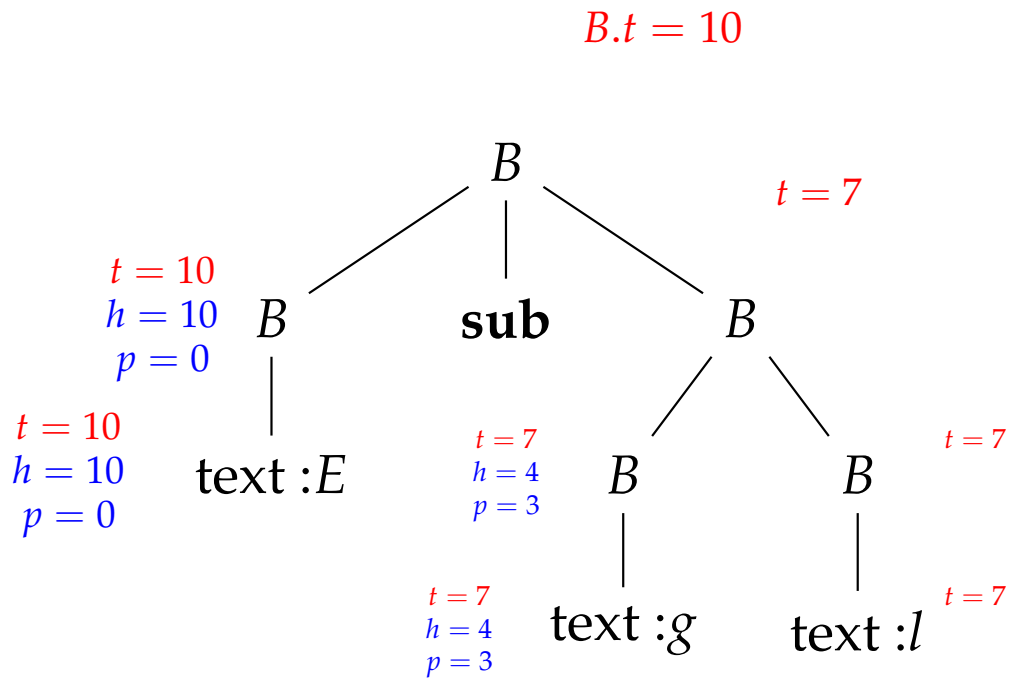
Parcours de l'arbre en profondeur, de gauche à droite :



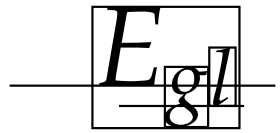
Exemple : typographie



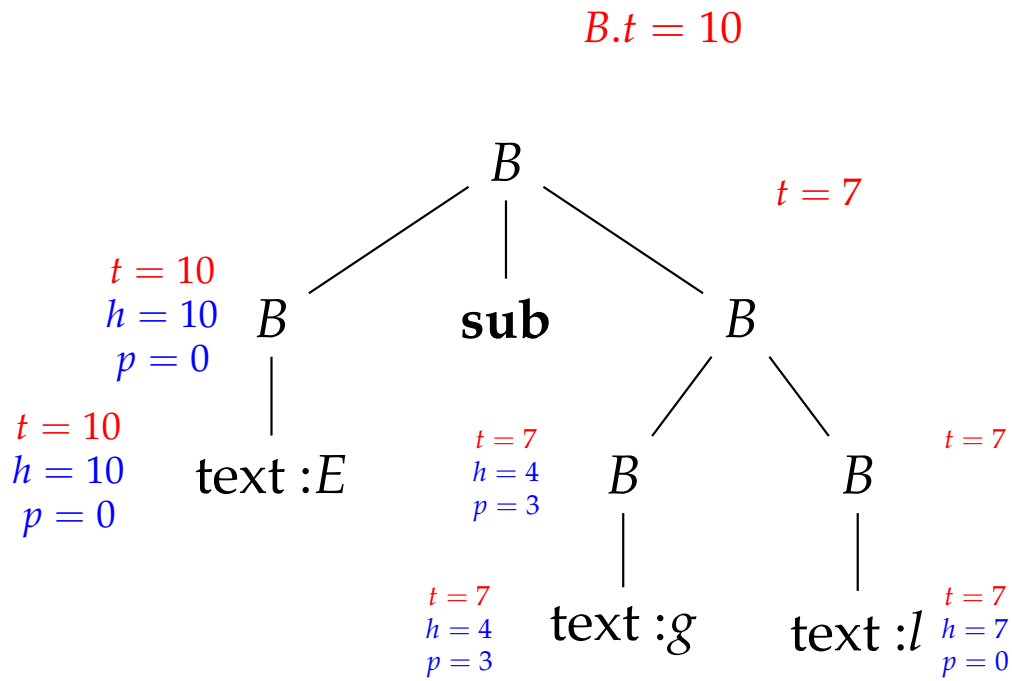
Parcours de l'arbre en profondeur, de gauche à droite :



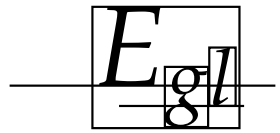
Exemple : typographie



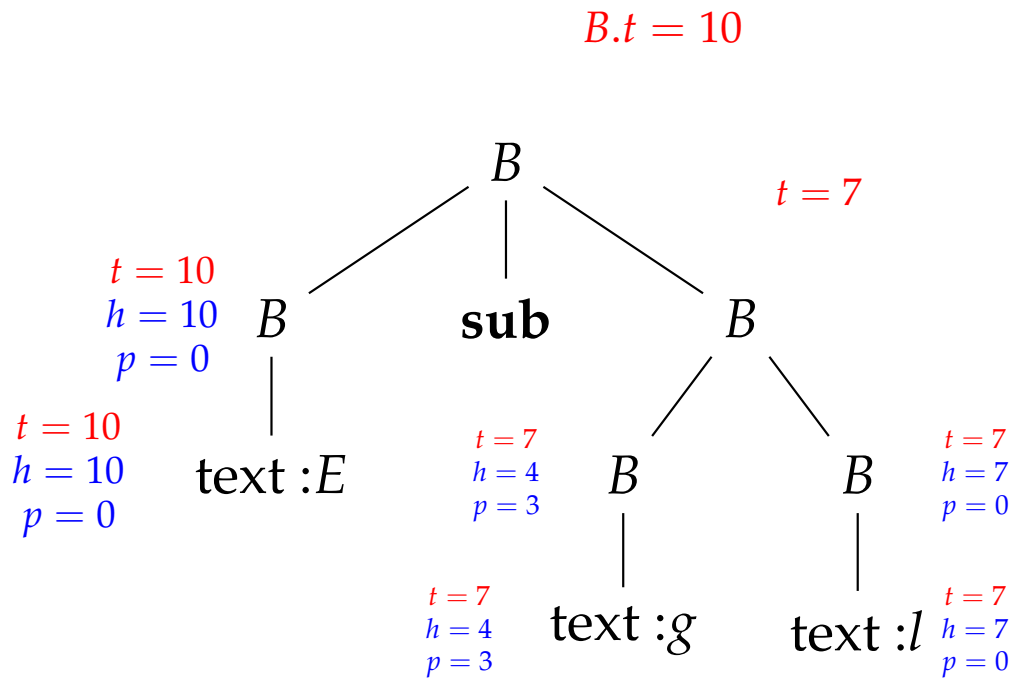
Parcours de l'arbre en profondeur, de gauche à droite :



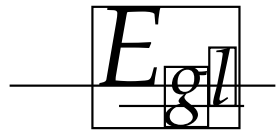
Exemple : typographie



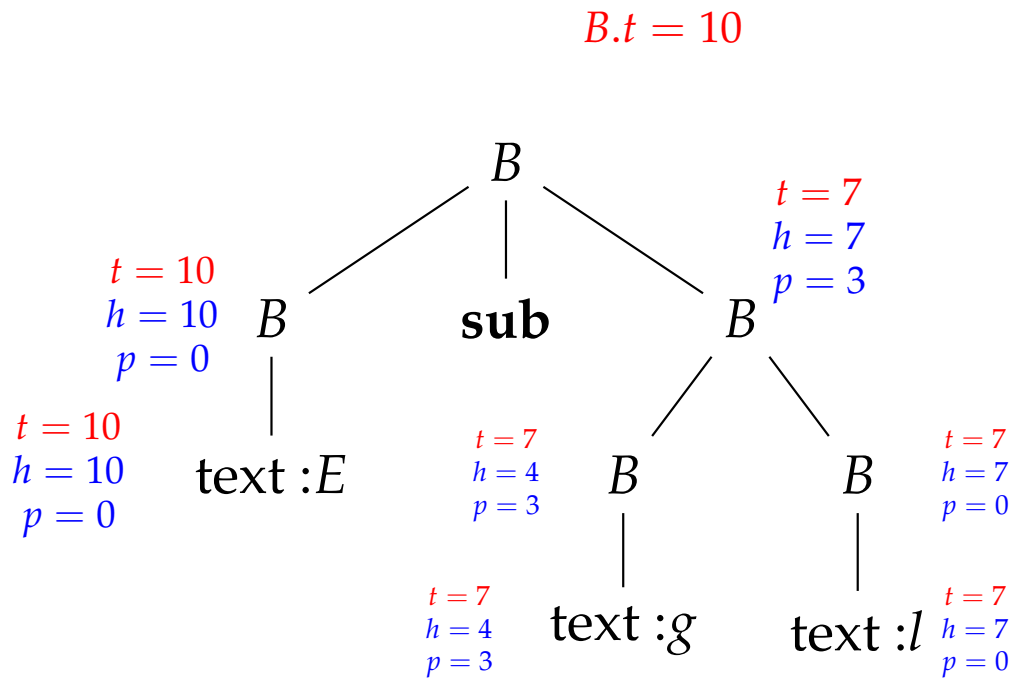
Parcours de l'arbre en profondeur, de gauche à droite :



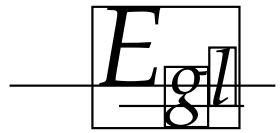
Exemple : typographie



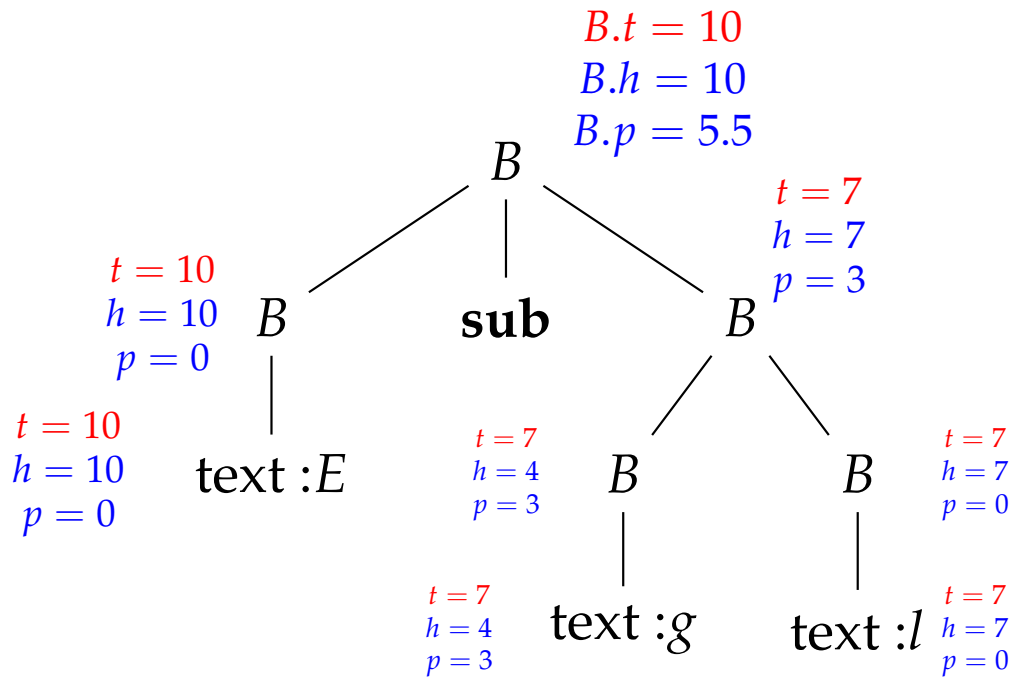
Parcours de l'arbre en profondeur, de gauche à droite :



Exemple : typographie



Parcours de l'arbre en profondeur, de gauche à droite :



S- ou L-attribuée ?

- Puisqu'on doit *descendre* dans l'arbre de dérivation pour hériter, les traductions L-attribuées sont plutôt réservées aux analyses *descendantes* (notamment LL(k)).
- Les analyses *ascendantes* préfèrent les traductions S-attribuées (LR(k), `yacc`).

Retour à l'arbre abstrait

Construction de l'arbre abstrait à l'aide d'une grammaire attribuée.

règle			action sémantique		
InstrSi	$\rightarrow$	`si` Exp `alors` Instr	InstrSi.n	=	noeud(SI, Exp.n, Instr.n)
Listeinstr	$\rightarrow$	Instr Listeinstr	Listeinstr.n	=	noeud(LI, Inst.n, Listeinstr1.n)
Listeinstr	$\rightarrow$	ε	Listeinstr.n	=	`NULL`

...

Construction de l'arbre abstrait

Objectif : construire l'arbre abstrait pendant la dérivation
Attribut : n noeud de l'arbre abstrait
Constructeur : *noeud(op, g, d)*

- $op \rightarrow$ opérateur
- $g \rightarrow$ fils gauche
- $d \rightarrow$ fils droit

règle			action sémantique		
E	$\rightarrow$	E + T	E.n	=	noeud($+$, E_1.n, T.n)
E	$\rightarrow$	E - T	E.n	=	noeud($-$, E_1.n, T.n)
E	$\rightarrow$	T	E.n	=	T.n
T	$\rightarrow$	T * F	E.n	=	noeud($\times$, T_1.n, F.n)
T	$\rightarrow$	T / F	E.n	=	noeud($\div$, T_1.n, F.n)
T	$\rightarrow$	F	T.n	=	F.n
F	$\rightarrow$	(E)	F.n	=	E.n
F	$\rightarrow$	n	F.n	=	n

Construction de l'arbre abstrait

Objectif : construire l'arbre abstrait pendant la dérivation
Attribut : n noeud de l'arbre abstrait
Constructeur : *noeud(op, g, d)*

- $op \rightarrow$ opérateur
- $g \rightarrow$ fils gauche
- $d \rightarrow$ fils droit

règle			action sémantique		
E	$\rightarrow$	E + T	E.n	=	noeud($+$, E_1.n, T.n)
E	$\rightarrow$	E - T	E.n	=	noeud($-$, E_1.n, T.n)
E	$\rightarrow$	T	E.n	=	T.n
T	$\rightarrow$	T * F	E.n	=	noeud($\times$, T_1.n, F.n)
T	$\rightarrow$	T / F	E.n	=	noeud($\div$, T_1.n, F.n)
T	$\rightarrow$	F	T.n	=	F.n
F	$\rightarrow$	(E)	F.n	=	E.n
F	$\rightarrow$	n	F.n	=	n