Kompilátory
Syntaxou riadený preklad

Ján Šturc

Zima 2009

Syntaxou riadené definície

- Symbolom gramatiky sú priradené atribúty, ktoré nesú dodačnú informáciu o konštrukciách rozpoznávaných syntaxou.
- Hodnoty atribútov sa vypočítavajú pomocou *sémantických rutín* asociovaných so syntaktickými pravidlami.
- Sémantické rutiny môžu:
  - generovať medzijazyk,
  - ukladať a vyberať informáciu z tabuľky symbolov,
  - vykonávať kontrolu typov,
  - produkovať správy o chybách,
  - vykonávať nejaké iné činnosti.
  - Môžu to byť akékoľvek programy, poradie ich výpočtu určuje prekladová schéma.
- *Atribút* je premenná ľubovoného typu:
  - reťazec, číslo, smerník, zložený typ (record).

Syntaxou riadené definície a prekladové schémy

- Pre programovaciu paradigmu syntaxou riadeného prekladu používame dva pojmy:
  - Syntaxou riadené definície
  - Prekladové schémy
- Syntaxou riadená definícia:
  - Je čosi medzi špecifikáciou a programom.
  - Abstrahuje od mnohých implementačných detailov, hlavne od poradia výpočtu.
  - K pravidlám gramatiky sa pridajú sémantické rutiny a nič sa nehovorí o poradí ich výpočtu.
- Prekladová schéma:
  - Určuje poradie volania sémantických rutín a spôsob komunikácie medzi nimi.
  - Poradie volania sémantických rutín môže a nemusí súvisieť s metódou syntaktickej analýzy.
  - Pre jednu metódu syntaktickej analýzy sa dá použiť (vymyslieť) viacero prekladových schém.

Syntaxou riadené definície

- *Syntaxou riadená definícia* je zovšeobecnenie bezkontextovej gramatiky:
  - Každému symbolu gramatiky je priradená množina atribútov.
  - Z množiny atribútov vydelujeme dve významné podmnožiny:
    - **syntetizované** atribúty (atribúty sa propagujú od synov otcovi)
    - **dedičné** atribúty (dedia sa od otca a súrodencov).
- *Prekladová schéma* je množina pravidiel, ktoré určujú graf závislosti medzi atribútmi a poradie ich vyhodnotenia.
  - Môže byť lokálna, v rámci pravidla
  - Globálna, spoločná pre všetky pravidlá
- *Graf závislostí* určuje aj poradie volania sémantických rutín.
- Sémantické rutiny počítajú hodnoty atribútov, môžu mať vedlajšie efekty (napr. zápis do tabuľky symbolov).

Atribútové gramatiky

Každému pravidlu gramatiky $A \rightarrow \alpha$ priradíme množinu sémantických rutín tvaru:

$$b := f(c_1, c_2, \ldots, c_n);$$

kde f je funkcia a pre b a $c_1, c_2, \ldots, c_n$ platí jeden z nasledujúcich výrokov:

➔ b je syntetizovaný atribút A $c_1, c_2, \ldots, c_n$ sú atribúty symbolov v pravidle ($A \rightarrow \alpha$).

alebo

➔ b je zdedený atribút niektorého zo symbolov α (pravej strany pravidla) a $c_1, c_2, \ldots, c_n$ sú atribúty symbolov v pravidle ($A \rightarrow \alpha$).

Všeobecná metóda – Annotated parse tree

- Syntaktický strom s uzlami „ozdobenými atribútmi“ sa nazýva *annotated parse tree*.
- Proces výpočtu hodnôt atribútov v uzloch sa nazýva *annotating* (alebo *zdobenie*) syntaktického stromu.
- Poradie týchto výpočtov je určené grafom závislostí atribútov, indukovaným prekladovou schémou.
- Vlastne ani nepotrebujeme prekladovú schému, stačí predpokladať, že používame globálne mená atribútov a výpočet atribútu je tvaru:

$$b := f(c_0, c_1, \ldots, c_{n-1});$$

  Graf závislostí konštruujeme z výpočtových pravidiel, hrana $b \leftarrow c_i$ znamená, že b sa počíta pomocou c_i.
- Topologické utriedenie grafu závislostí implikuje poradie výpočtu.

Atribútové gramatiky

- Sémantické pravidlo $b := f(c_1, c_2, \ldots, c_n)$ hovorí, že the atribút b závisi od atribútov $c_1, c_2, \ldots, c_n$.

- V syntaxou riadených dafiníciach, sémantické rutiny (pravidlá) môžu okrem výpočtu atribútova mať vedlajšie efkty (side effects) napr.: zápis do tabuľky, výstup a pod.

- Atribútová gramatika je syntaxou riadená definícia, v ktorej sémantické rutiny nemajú vedlajší efekt. Sú to funkcie, ktoré počítajú atribúty.

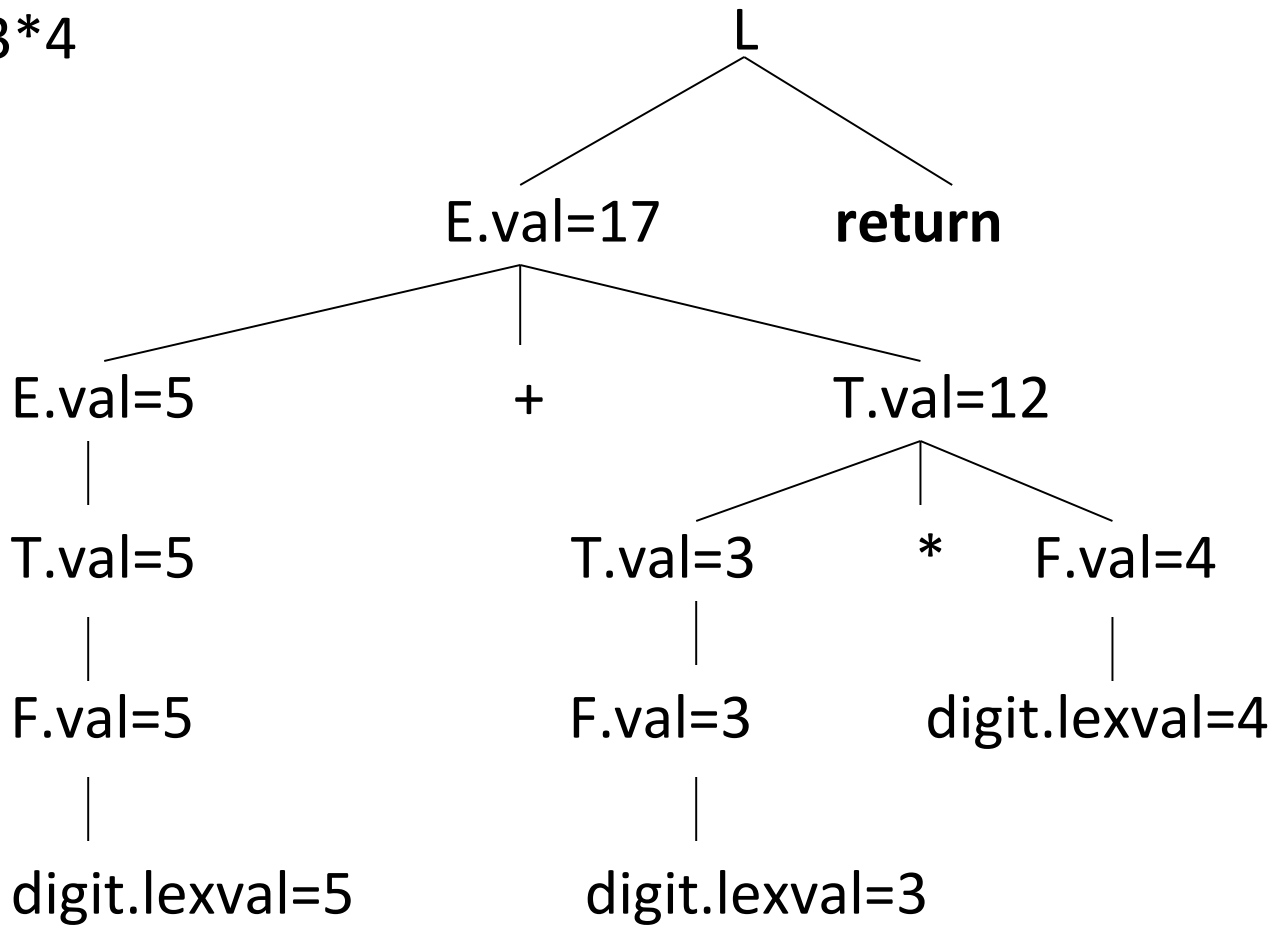
Syntaxou riadená definícia – príklad

Gramatika	Sémantické rutiny
$L \rightarrow E$ **return**	print(E.val)
$E \rightarrow E_1 + T$	$E.val = E_1.val + T.val$
$E \rightarrow T$	E.val = T.val
$T \rightarrow T_1 * F$	$T.val = T_1.val * F.val$
$T \rightarrow F$	T.val = F.val
$F \rightarrow (E)$	F.val = E.val
$F \rightarrow$ **digit**	F.val = **digit**.lexval

9. Symbolom E, T a F je priradený (syntetizovaný) atribút *val*.
10. Token **digit** má atribút *lexval* (Predpokladáme, že hodnotu mu priradila lexikálna analýza.)

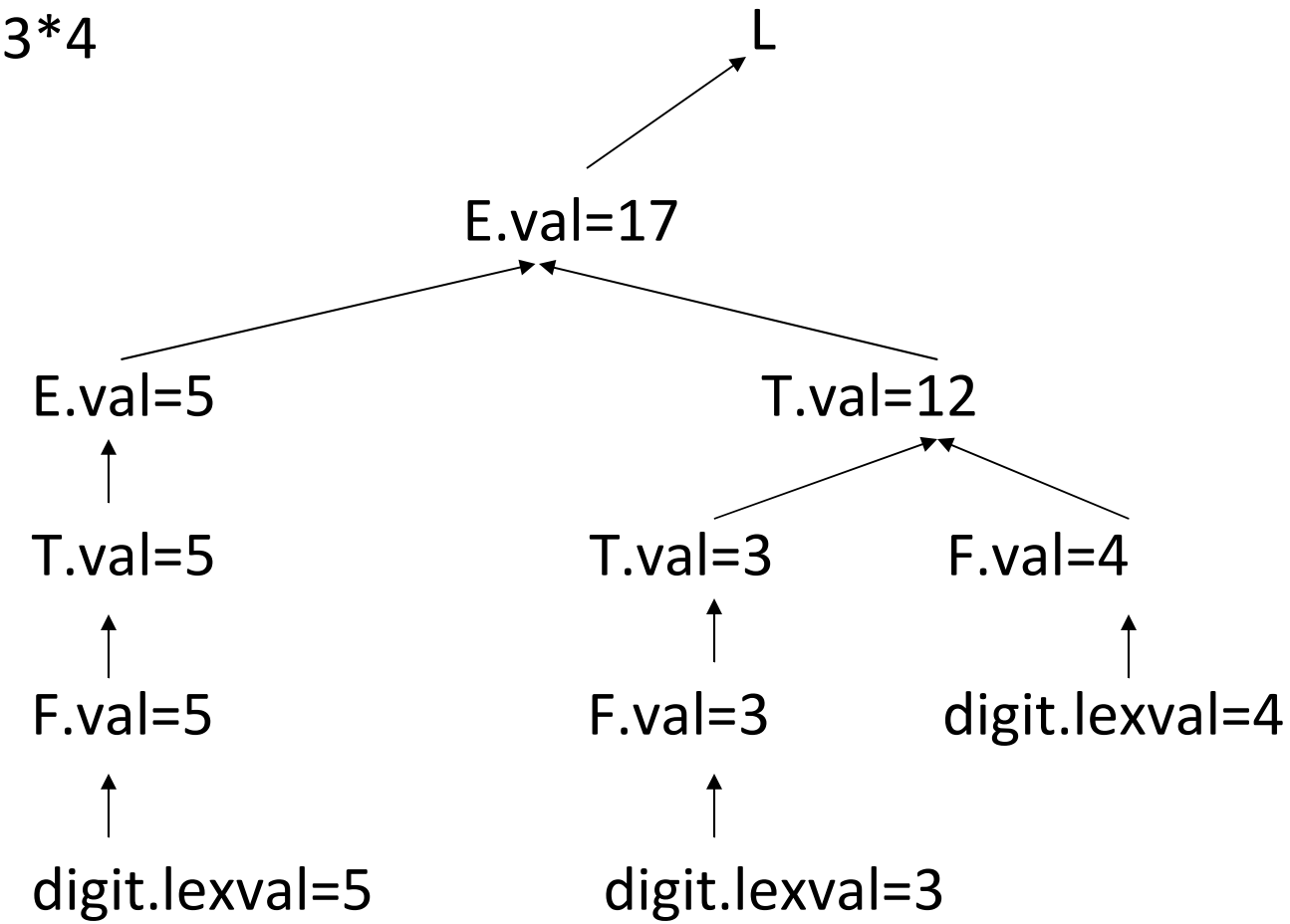
Ozdobený syntaktický strom – príklad

Vstup: 5+3*4



Graf závislosti atribútov – príklad

Vstup: 5+3*4



Iný príklad – generovanie kódu

Gramatika	Semantické rutiny
$E \rightarrow E_1 + T$	E.loc=newtemp();
	E.code = E_1.code ‖ T.code ‖ add E_1.loc,T.loc,E.loc
$E \rightarrow T$	E.loc = T.loc; E.code=T.code
$T \rightarrow T_1 * F$	T.loc=newtemp();
	T.code = T_1.code ‖ F.code ‖ mult T_1.loc, F.loc, T.loc
$T \rightarrow F$	T.loc = F.loc, T.code=F.code
$F \rightarrow (E)$	F.loc = E.loc, F.code=E.code
$F \rightarrow$ **id**	F.loc = **id**.name, F.code=""

- Symboly E, T, a F majú dva syntetizované atribúty *loc* a *code*.
- Token **id** má a syntetizovaný atribút *name* (predpokladáme, že ho priradila lexikálna analýza).
- ‖ označuje zreťazenie.

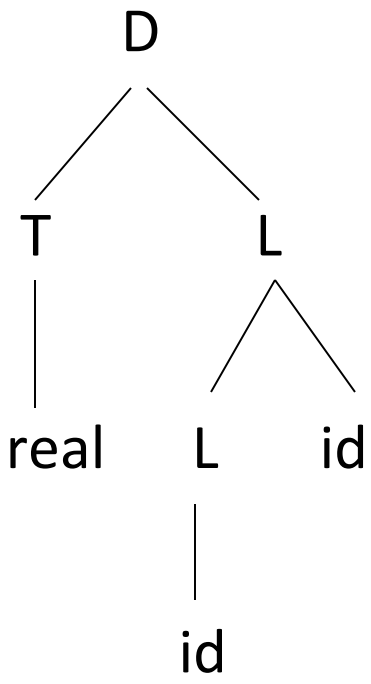
Syntaxou riadené definície – zdedené atribúty

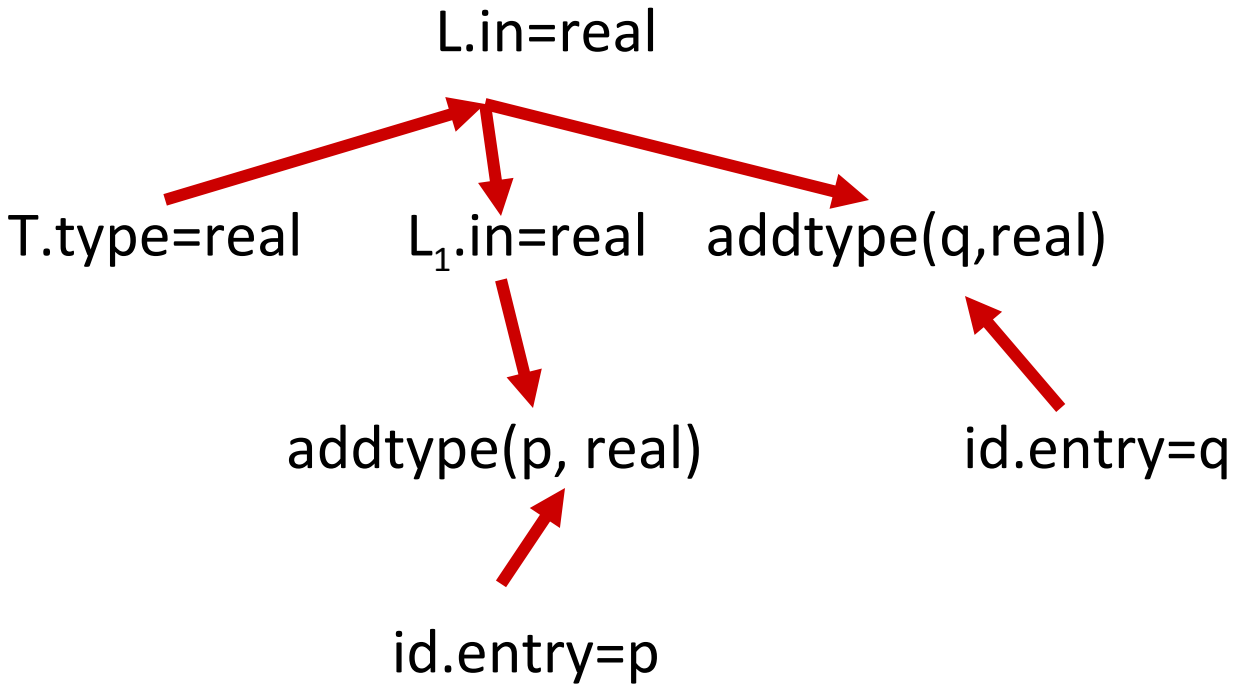
Gramatika	Sémantické rutiny
D → T L **;**	L.in:= T.type;
T → **int**	T.type:= integer;
T → **real**	T.type:= real;
L → L_1 **, id**	L_1.in = L.in; addtype(**id**.entry, L.in);
L → **id**	addtype(**id**.entry, L.in);

- Symbol T má syntesizovaný atribút *type*.
- Symbol L má zdedený atribút *in.*

Graf závislosti atribútov – zdedené atribúty

Vstup: `real p q`





syntaktický strom

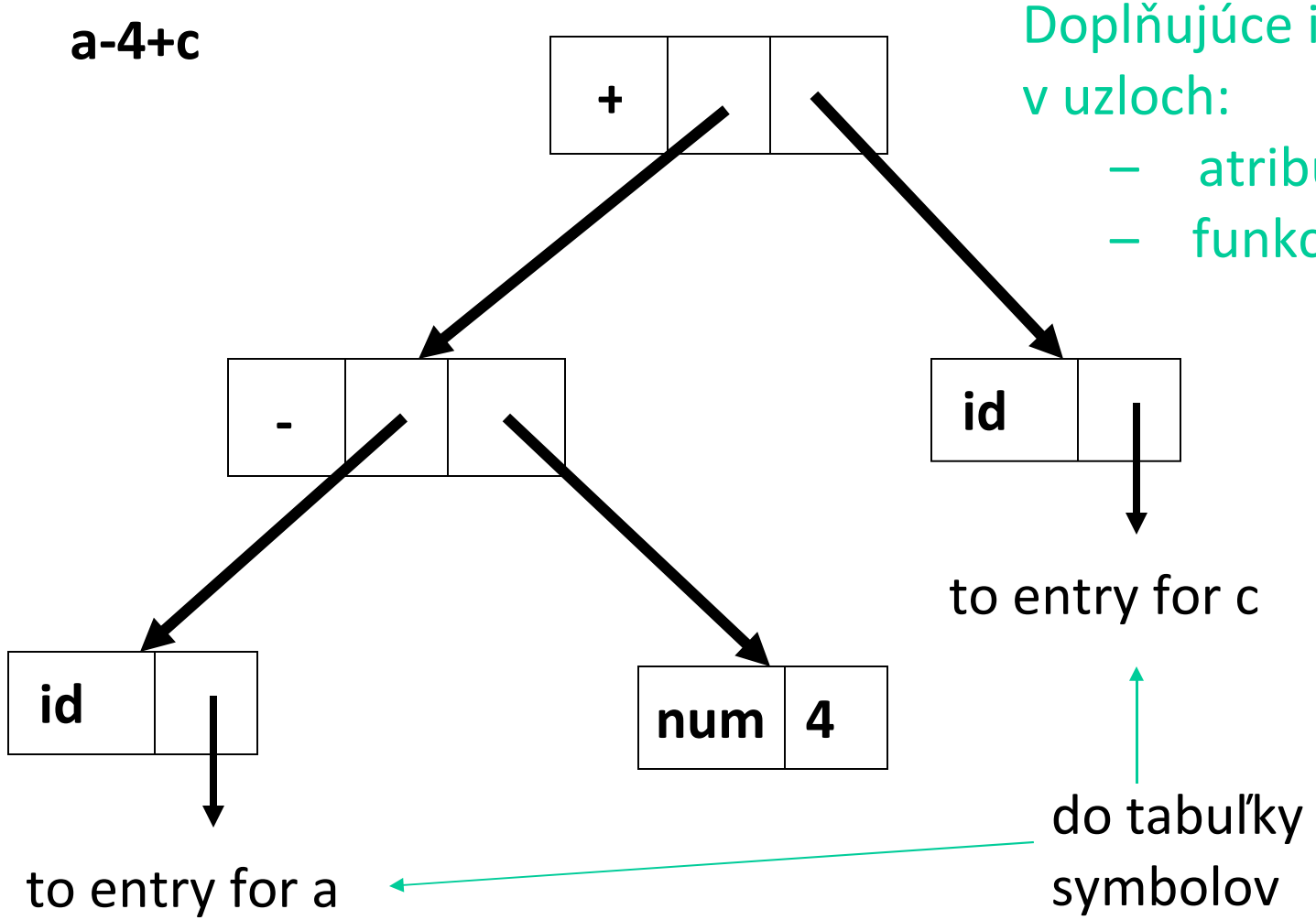
graf závislosti atribútov

Syntaktické stromy ako vnútorná reprezentácia

- Oddelenie prekladu od syntaktického stromu
- Syntaktický strom: je vnútorna reprezentácia vstupu prvý postupný cieľ prekladu.
- Nástroje: ptn function *mknode(op:op, ptn:ls, ps);*
  ptn function *mkleaf(a:token, b:lexval);*
  syntetizovaný atribút *nptr* (ptn: pointer to node)

Gramatika	Sémantické rutiny
$E \rightarrow E_1 + T$	E.*nptr* = *mknode*("+",E_1.*nptr* ,T.*nptr*)
$E \rightarrow E_1 - T$	E.*nptr* = *mknode*("-",E_1.*nptr* ,T.*nptr*)
$E \rightarrow T$	E.*nptr* = T.*nptr*
$T \rightarrow (E)$	T.*nptr* = E.*nptr*
$T \rightarrow id$	T.*nptr* = *mkleaf*(id, id.*lexval*)
$T \rightarrow num$	T.*nptr* = *mkleaf(num,* num.*lexval)*

Reprezentácia syntaktických stromov dátova štruktúra



Doplňujúce informácie v uzloch:

- atribúty
- funkcie

Vytvorenie grafu závislostí medzi atribútami

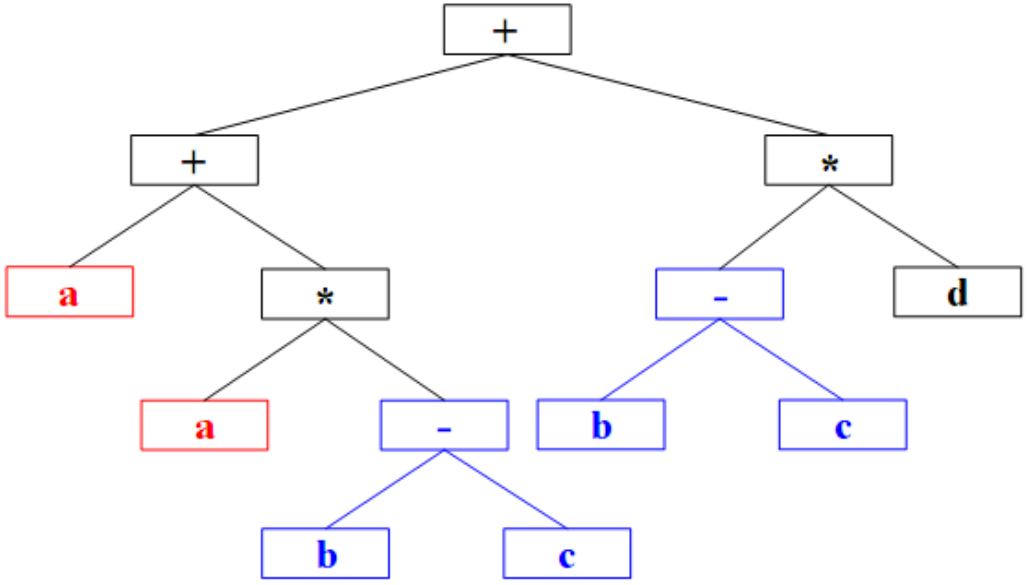
- D = <V, E>, kde V je množina vrcholov a E množina hrán

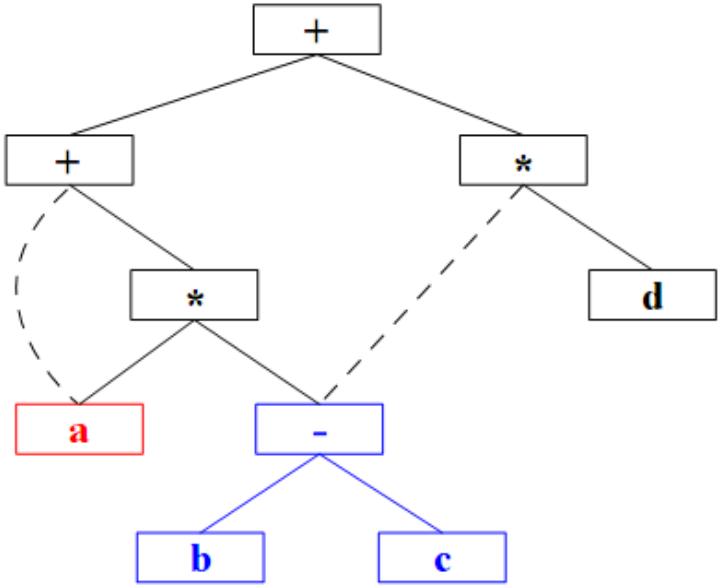
```
for každý vrchol n v strome odvodenia do
    for každý atribút a symbolu gramatiky v n do
        vytvor vrchol pre a;
for každý vrchol n v strome odvodenia do
    for každé sémantické pravidlo b := f(c_1, ..., c_k) asociované s pravidlom v n do
        for i := 1 to k do
            pridaj hranu z c_i do b
```

- Ak graf D neobsahuje cyklus, jeho topologickým utriedením získame poradie výpočtu atribútov
- Nevýhodou všeobecnej metódy je potreba konštrukcie syntaktického stromu a grafu závisloti.

Stromy a dagy výrazov

Výraz: $a + a * (b - c) + (b - c) * d$





Strom výrazu, hoci je veľmi podobný syntaktickému stromu, nie je syntaktický strom. (Vzájomná transformácia je „skoro izomorfizmus“).

Infixová, prefixová a sufixová notácia majú rôzne syntaktické stromy, ale ten istý strom výrazu.

Prekladové schémy

1. Konštrukciou ozdobeného syntaktického stromu a grafu závislostí
   - Práve bola popísaná.
   - Má viac prechodov.
   - Počas kompilácie
2. Bez explicitnej konštrukcie grafu závislostí. Poradie je určené metódou syntaktickej analýzy.
   - S-atribútové definície
   - L-atribútove definície
3. Cez globálne dátové štruktury. Sémantické rutiny okrem atribútov pracujú aj s globálnymi štruktúrami.
   - Syntaktický analyzátor volá sémantické rutiny podľa potreby.
   - Globálne dátové štruktúry umožňujú obchádzať zásobníkovú disciplínu odovzdávania atribútov.

Ohraničenia na syntaxou riadené definície

1. S-atribútové definície:
   - Používajú len syntetizované atribúty.
   - Syntetizované atribúty sú dobré. Syntaktická analýza zdola-nahor ich môže počítať pri redukcii.
2. Zdedené atribúty sa vo všeobecnosti nedajú vypočítať počas syntaktickej analýzy.
   - Pri analýze zhora dolu sú problémom atribúty symbolov napravo od práve prezeraného symbolu.
3. L-atribútové definície:
   - Okrem syntetizovaných atribútov používajú aj
   - zdedené atribúty, ale dediť sa môže len od otca a ľavých súrodencov.
   - Dajú sa vyhodnotiť pri analýze zhora nadol. (Syntetizovaný atribút otca vyhodnotíme ako zdedený atribút posledného symbolu pravidla.)

- S-atribútové a L-atribútové definície sa dajú vyhodnotiť v jedinom prechode súčasne so syntaktickou analýzou.

Vyhodnotenie S-atribútových definícii zdola nahor

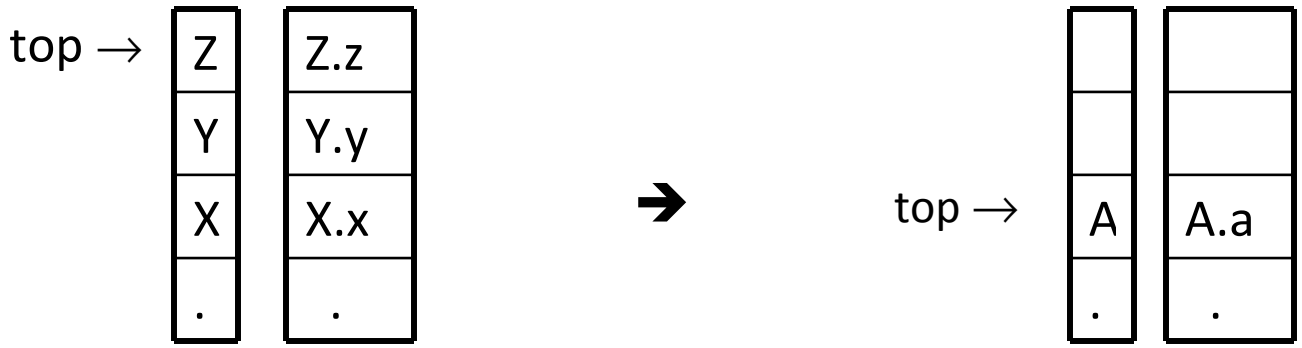
- Syntetizované atribúty symbolov gramatiky ukladáme do paralelného zásobníku.
  - Predstavujeme si zásobník ako

stav	symbol	atribúty

  - Keď zásobník syntaktického anlyzátora obsahuje symbol X (terminál alebo neterminál), zodpovedajúca položka paralelného zásobníku obsahuje syntetizované atribúty symbolu X.
- Výpočet sémantickej rutiny sa realizuje súčasne s redukciou.

A $\rightarrow$ XYZ *A.a:= f(X.x,Y.y,Z.z),* všetky atribúty sú syntetizované.

zásobník (abstrahujeme od stavov)



Príklad – implementácia S-atribútovej definície

Pravidlá	Sémantické rutiny
L → E **return**	print(val[top-1])
E → E_1 + T	val[ntop] = val[top-2] + val[top]
E → T	
T → T_1 * F	val[ntop] = val[top-2] * val[top]
T → F	
F → (E)	val[ntop] = val[top-1]
F → **digit**	

- Pri operácii shift sa lexikálny atribút lexval tokenu (digit.*lexval*) na vstupe natlačí (push) do paralelného zásobniku. Ak token (+,*) nemá atribúty, môže to byť hocičo, resp. stačí posunúť smerník na vrchol zásobníku.
- Pri redukcii podľa pravidla A → β sa vrchol zásobníka nastaví na ntop = top – |β| + 1 .

L-atribútové definície

- Syntaxou riadená definícia je L-atribútová, ak pre ľubovolné pravidlo $A \rightarrow X_1X_2 \ldots X_n$ platí, že každý dedičný atribút symbolu X_i z pravej strany pravidla je závislý iba na:
  - atribútoch symbolov $X_1, \ldots, X_{i-1}$, ktoré sú v pravidle na ľavo od X_i
  - a dedičných atribútoch symbolu A.
- Príklad SRD, ktorá nie je L-atribútová

Gramatika	Sématické rutiny	
$A \rightarrow LM$	$L.i := l(A.i)$	
	$M.i := m(L.s)$	
	$A.s := f(M.s)$	
$A \rightarrow QR$	$R.i := r(A.i)$	
	$Q.i := q(R.s)$	← Nesplňuje definíciu
	$A.s := f(Q.s)$	

Pozn.: Každá S-atribútová SRD je aj L-atribútová.

Vyhodnotenie L-atribútových definícii

- Atribúty sa dajú vyhodnotiť prehľadávaním syntaktického stromu do hĺbky (depth-first search).

```
procedure dfvisit(n:node);
begin
    for každé dieťa m vrcholu n zľava doprava do begin
        vyhodnoť dedičné atribúty vrcholu m;
        dfvisit(m)
    end;
    vyhodnoť syntetizované atribúty vrcholu n
end
```

- Implementácia pri syntaktickej analýze
  - Zhora nadol: Všetky SRD založené na LL(1) gramatikách.
  - Zdola nahor: Všetky SRD založené na LL(1) gramatikách a veľká časť SRD založených na LR(1) gramatikách.

Prekladové schémy I

- Prekladová schéma je konkretizácia syntaxou riadenej definície.
- Špecifikuje naviac poradie výpočtu atribútov.
- Volanie sémantických rutín sa zapisuje v krútených zátvorkách do vnútra pravej strany pravidla na miesto, kde sa má vykonať.
- Príklad – preklad aritmetických výrazov do sufixovej formy

  E → T R

  R → **addop** T *{emit(***addop**.*lexval)}* R_1 |ε

  T → F Q

  Q → **mulop** F *{emit(***mulop**.*lexval)}* Q_1 |ε

  F → **(**E**)** | **id** *{emit(***id**.*lexval)}*
- Tajne predpokladáme, že syntaktická analýza sa bude robiť zhora nadol a v tom poradí sa vyhodnocujú aj sémantické rutiny.

Prekladové schémy II

- Prakticky jediným obmedzením je, že sémantická rutina sa nesmie odkazovať na ešte nevyhodnotený atribút.
- Pre S-atribútové definície je jednoduché sémantické rutiny prídu nakoniec.
- Príklad ten istý

  E → E **addop** T *{emit(***addop**.*lexval)}*| T

  T → T **mulop** F *{emit(***mulop**.*lexval)}* | F

  F → **(**E**)** | **id** *{emit(***id**.*lexval)}*
- Tentokrát na syntaktickú analýzu používame posunovo redukčný automat.

Prekladové schémy pre L-atribútové SRD

1. Dedičný atribút symbolu na pravej strane pravidla musí byť vypočítaný v sémantickej rutine naľavo od neho.
2. Sémantická rutina nesmie používať syntetizované atribúty symbolov napravo od nej.
3. Syntetizovaný atribút ľavej strany pravidla sa dá vypočítať až potom, čo boli vypočítané všetky atribúty, na ktorých je závislý.
   - Obvykle sémantickú rutinu, ktorá ho počíta, umiestňujeme na konci pravej strany príslušného pravidla.

- L-atribútové prekladové schémy sa prirodzene implementujú pre LL(1) gramatiky.
- Ľavo rekurzívne schémy používajúce len syntetizované atribúty vieme „mechanicky“ prerobiť na pravo rekurzívne.

Transformácia ľavo rekurzívnej schémy

- Vstupná prekladová schéma:

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

- Po odstránení ľavej rekurzie z CF gramatiky získame novú gramatiku:

$$
\begin{array}{lcl}
A & \rightarrow & X\ R \\
R & \rightarrow & Y\ R \mid \varepsilon
\end{array}
$$

- Po prepísaní sémantických akcií získame výslednú prekl. schému:

$$
\begin{array}{lcll}
A & \rightarrow & X & \{R.i := f(X.x)\} \\
 & & R & \{A.a := R.s\} \\
R & \rightarrow & Y & \{R_1.i := g(R.i, Y.y)\} \\
 & & R_1 & \{R.s := R_1.s\} \\
R & \rightarrow & \varepsilon & \{R.s := R.i\}
\end{array}
$$

Príklad - kalkulačka

$E \rightarrow E_1$ **+** T {E.val := E_1.val + T.val} | T {E.val := T.val}

$T \rightarrow T_1$ ***** F {T.val := T_1.val * F.val} | F {T.val := F.val}

$F \rightarrow$ (E) {F.val := E.val}

$F \rightarrow$ **num** {F.val := **num**.lexval}

$E \rightarrow T$ {R.i := T.val} R {E.val := R.s}

$R \rightarrow$ **+** T {R_1.i := R.i + T.val} R_1 {R.s := R_1.s}

| ε {R.s := R.i}

$T \rightarrow F$ {Q.i := F.val} Q {T.val := Q.s}

$Q \rightarrow$ ***** F {Q_1.i := Q.i * F.val} Q_1 {Q.s := Q_1.s}

| ε {Q.s := Q.i}

$F \rightarrow$ **(** E **)** {F.val := E.val}

| **num** {F.val := **num**.lexval}

Rekurzívny zostup pre L-atribútové schémy

- Neterminálnym symbolom priradíme funkcie.
  - Argumenty funkcie priradenej neterminálnemu symbolu A sú zdedené atribúty symbolu A.
  - Funkcia priradená symbolu A vracia syntetizované atribúty symbolu A.
- Funkcia A musí plniť obe úlohy
  - Robiť syntaktickú analýzu
    - Zistiť, ktoré pravidlo treba použiť
    - Testovať terminálne symboly na vstup (match)
  - Počítať atribúty
    - Uchovať v lokálnych premenných hodnoty atribútov potrebné pre výpočet zdedených atribútov neterminálnych symbolov v tele alebo syntetizovaných atribútov A.
    - Volať funkcie pre neterminály tela (pravých strán pravidiel) so správnymi argumentami. (Pretože, sa jedná o L-atributovú syntaxou riadenú definíciu, tieto už boli vypočítané a mohli byť uchované v lokálnych premenných.)

Implementácia pre LL(1) syntaktický analyzátor

1. L-atribútová definícia sa dá zorganizovať tak, že „skutočné“ atribúty majú len neterminálne symboly. Terminálne symboly (tokens) majú len svoje lexikálne atribúty dostupné prostredníctvom tabuľky symbolov.
2. Sémantické akcie ukladáme do zásobniku formou dvoch typov položiek (records)
   - Záznam pre zdedené atribúty obsahuje zdedené atribúty, neterminálu V, ktorý je v zásobníku bezprostredne pod ním. Táto položka obsahuje aj referenciu na sémantickú rutinu, ktorá počíta atribúty neterminálu V.
   - Záznam pre syntetizované atribúty, nachádza sa v zásobníku bezprostredne pod neterminálom V, ktorému tieto atribúty patria. Môže tie obsahovať referenciu na sémantickú rutinu.
3. Vytváranie záznamov a prenosy atribútov sa vykonajú počas náhrady neterminálu na vrchole zásobníku. Program vie, ako hlboko v zásobniku sa potrebný záznam nachádza.

Implementácia L-atribútových SRD a LR parsing

1. Budeme uvažovať L-atribútovú SRD, ktorá má syntaktické rutiny pre výpočet zdedených atribútov pred neterminálnymi sybolmi a rutiny pre výpočet syntetizovaných atribútov na konci pravidla pre daný neterminál.
2. Upravíme gramatiku tak, že namiesto každej sémantickej rutiny vo vnútri pravidla vložíme nový, nikde sa nevyskytujúci, neterminál (M_1, M_2, ...). Pre všetky značkovacie neterminály pridáme pravidlá $M_i \rightarrow \varepsilon$.
3. Modifikujeme sémantické rutiny. Nech pôvodné $A \rightarrow \alpha\{a\}B\beta$ je modifikované na $A \rightarrow \alpha MB\beta$. Pridáme pravidlo $M_i \rightarrow \varepsilon\{a'\}$. Kde a':
   - Kopíruje atribúty A a symbolov α potrebné k výpočtu a ako zdedené atribúty neterminálu M.
   - Vypočíta to isté ako a (zdedené atribúty B) ako syntetizované atribúty M.
   - Je to porušenie zásobníkovej disciplíny. Doteraz sa používali len atribúty symbolov v rámci pravidla. Nevadí, potrebné atribúty sú v LR zásobniku.
   - Pri kontext senzitívnej analýze tento problém nenastane ($\alpha MB \rightarrow \alpha B$).

Metóda nedodržujúca zásobníkovú disciplínu

- Vo všeobecnosti nie je potrebné dodržať zásobníkovú disciplínu.
- V tomto prípade syntaktická analýza a sémantika pracujú ako „coroutines“ – dva programy, z ktorých každý ma svoje dátové štruktúry a svoj stav.
  - Syntaktická analýza, môže kedykoľvek prerušiť svojú prácu a požiadať sémantiku, aby vykonala nejakú rutinu.
  - Sémantika ju vykoná a výsledok si zapamätá vo svojich štruktúch. (V podstate sémantika je objekt, ktorý ponúka svoje metódy syntaktickej analýze.)
  - Po skončení (akceptovaní) sémantika, alebo má výsledok, alebo ho vygenerovala v priebehu práce.
- Dá sa urobiť viac ako L-atribútové prekladové schémy.
  - Použitie metódy je silne závislé od kvality programátora.

Zacyklenie – cirkulárne SRD

- Môže sa stať, že sa pre niektoré slová vypočet atribútov zacyklí. Zacyklené atribúty sa vo všeobecnosti nedajú vypočitať.
- Testovanie, či sa výpočet zacyklí pre dané slovo je jednoduché:
  - Vytvoríme ozdobený syntaktický strom, graf závislosti atribútov, zistíme, či graf obsahuje cyklus.

Def.: Hovoríme, že atribútová gramatika (SRD) je *cirkulárna*, ak jazyk generovaný gramatikou obsahuje slovo, na ktorom sa výpočet atribútov zacyklí.

- Je to rozhodnuteľné. Stačí vyskúšať konečný počet slov. Pumpovacia lema. Ak k zacykleniu nedôjde pri vyskušaní slov $\alpha\upsilon^k\beta\omega^k\gamma$, pre $k \in \{0,1,2\}$, už k nemu nedôjde nikdy.
- Je to ťažké, problém je NP-úplný.
- Otázky cirkularity majú len teoretický význam pre celkový stav oblasti. Z hľadiska praxe sú nedôležité.