# Süntaksanalüüs

# Süntaksanalüüs

- Süntaksanalüüs kontrollib programmi struktuuri vastavust keele grammatikale:
  - saab sisendina, skanneri poolt genereeritud, lekseemide jada;
  - väljastab programmi esitava (abstraktse) süntaksipuu;
  - süntaktiliste vigade korral, teeb kindlaks nende asukoha;
  - ... teavitab võimalikest vea põhjustest;
  - ... püüab veast toibuda ja jätkata analüüsi (et järgnevaid vigu avastada).

- Süntaksanalüüsi kutsutakse parsimiseks (parsing) ning vastavat analüsaatorit nimetatakse parseriks (parser).

# Grammatikad

- Keelte süntaksi kirjeldatakse reeglina kontekstivaba grammatika abil.
- Grammatika on nelik $G = \langle N, T, P, S \rangle$, kus
  - $N$ on lõplik mitteterminaalide tähestik;
  - $T$ on lõplik terminaalsümbolite tähestik;
  - $N \cap T = \emptyset$ ja $V = N \cup T$;
  - $P \subset \{\alpha \to \beta \mid \alpha \in V^+,\ \beta \in V^\star\}$ on lõplik produktsioonireeglite hulk;
  - $S \in N$ on algsümbol.
- Grammatika on kontekstivaba (context-free), kui produktsioonireeglid on kujul $A \to \alpha$, kus $A \in N$ ja $\alpha \in V^\star$.

# Grammatikad

- Jada $w \in V^\star$ nimetatakse lausevormiks (sentential form).
- Lausevorm $v \in V^\star$ on otsetuletatav (directly derivable) lausevormist $u \in V^\star$ (tähistus $u \Longrightarrow v$), kui leiduvad $w_1, w_2, \alpha, \beta \in V^\star$ sellised, et $u = w_1 \alpha w_2$, $v = w_1 \beta w_2$ ja $\alpha \to \beta \in P$.
- Relatsiooni $\Longrightarrow$ refleksiivset transitiivset sulundit (tähistus $\Longrightarrow^\star$) nimetatakse derivatsiooniks (derivation) ehk tuletuseks.
- Grammatika $G = \langle N, T, P, S \rangle$ genereerib keele

$$L(G) = \{w \in T^\star \mid S \Longrightarrow^\star w\}$$

- Grammatikad $G_1$ ja $G_2$ on ekvivalentsed, kui $L(G_1) = L(G_2)$.

# Grammatikad

## Chomsky hierarhia:

| | **Produktsioonid** | **Keelte tüüp** | **Automaat** |
|---|---|---|---|
| $L_0$ | $\alpha \to \beta$ | Semi-Thue süsteemid | Turingi masin |
| $L_1$ | $\alpha A \beta \to \alpha \gamma \beta$ | Kontekstist sõltuvad | Tõkestatud Turingi masin |
| $L_2$ | $A \to \alpha$ | Kontekstivabad | Magasinmäluga automaat |
| $L_3$ | $A \to w,\ A \to wB$ | Regulaarsed | Lõplik automaat |
| $(L_4)$ | $A \to w$ | Lõplikud | Tsükliteta lõplik automaat |

kus $A, B \in N,\ \alpha, \beta, \gamma \in V^\star$ ja $w \in T^\star$.

Lemma: Chomsky hierarhia on range; so.:

$$(L_4) \subset L_3 \subset L_2 \subset L_1 \subset L_0$$

# Kontekstivabad grammatikad

- Edaspidi käsitleme ainult kontekstivabu grammatikaid.
- Kontekstivabade grammatikate produktsioonireegleid esitatakse tavaliselt Backus-Naur'i kujul (BNF).
- Näide: olgu $N = \{\mathrm{Exp}\}$ ja $T = \{+, *, (, ), id\}$, siis

$$\begin{array}{rcl} \mathrm{Exp} & \to & \mathrm{Exp} + \mathrm{Exp} \\ & | & \mathrm{Exp} * \mathrm{Exp} \\ & | & (\ \mathrm{Exp}\ ) \\ & | & id \end{array}$$

esitab produktsioonireeglite hulka

$$\begin{array}{rcll} P & = & \{ & \mathrm{Exp} \to \mathrm{Exp} + \mathrm{Exp}, \quad \mathrm{Exp} \to (\ \mathrm{Exp}\ ), \\ & & & \mathrm{Exp} \to \mathrm{Exp} * \mathrm{Exp}, \quad \mathrm{Exp} \to id \quad \}. \end{array}$$

# Kontekstivabad grammatikad

- Mitteterminaal $A$ on produktiivne (productive), kui leidub $w \in T^\star$ selline, et $A \Longrightarrow^\star w$.
- Mitteterminaal A on saavutatav (reachable), kui leiduvad lausevormid $u, v \in V^\star$ sellised, et $S \Longrightarrow^\star uAv$.
- KV-grammatika $G = \langle N, T, P, S \rangle$ on taandatud (reduced), kui tema iga mitteterminaal on produktiivne ja saavutatav.
- Lemma: Iga KV-grammatika saab teisendada temaga ekvivalentseks taandatud KV-grammatikaks.

# Kontekstivabad grammatikad

- Reeglina saab ühte ja sama lauset tuletada paljudel eri viisidel.
- Kanoonilised derivatsioonid:
  - vasakderivatsioon – derivatsiooni igal sammul asendatakse vasakpoolseim mitteterminaal;
  - paremderivatsioon – derivatsiooni igal sammul asendatakse parempoolseim mitteterminaal.
- Näide:

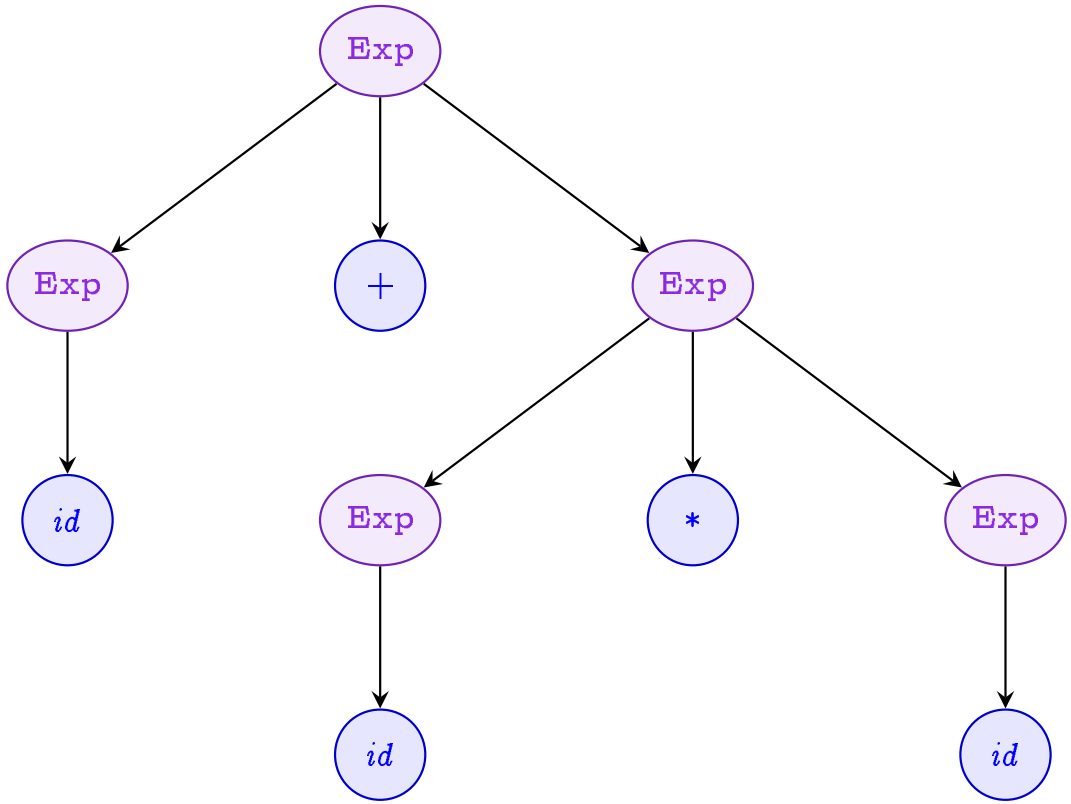
$$
\begin{aligned}
\mathrm{Exp} &\Longrightarrow_{lm} \mathrm{Exp} + \mathrm{Exp} \\
&\Longrightarrow_{lm} id + \mathrm{Exp} \\
&\Longrightarrow_{lm} id + \mathrm{Exp} * \mathrm{Exp} \\
&\Longrightarrow_{lm} id + id * \mathrm{Exp} \\
&\Longrightarrow_{lm} id + id * id
\end{aligned}
\qquad
\begin{aligned}
\mathrm{Exp} &\Longrightarrow_{rm} \mathrm{Exp} + \mathrm{Exp} \\
&\Longrightarrow_{rm} \mathrm{Exp} + \mathrm{Exp} * \mathrm{Exp} \\
&\Longrightarrow_{rm} \mathrm{Exp} + \mathrm{Exp} * id \\
&\Longrightarrow_{rm} \mathrm{Exp} + id * id \\
&\Longrightarrow_{rm} id + id * id
\end{aligned}
$$

# Kontekstivabad grammatikad

- Iga derivatsioon määrab üheselt süntaksipuu (syntax-tree, parse-tree), mis on järjestatud tippudega puu, kus:
  - puu juur on märgendatud algsümboliga S;
  - vahetipud on märgendatud mitteterminaalidega;
  - lehed on märgendatud terminaalsümboliga või tühisümboliga $\varepsilon$;
  - kui vahetipp on märgendatud mitteterminaaliga $A$ ja tema alampuude (vasakult paremale) $t_1, \ldots, t_n$ juured on märgendatud vastavalt $A_1, \ldots, A_n$, siis $A \to A_1 \ldots A_n \in P$.
- Tuletatud lause saadakse lugedes puu lehtede märgendid vasakult paremale.
- Süntaksipuu määrab üheselt kasutatud produktsiooni-reeglid, kuid mitte nende rakendamise järjekorda.

# Kontekstivabad grammatikad

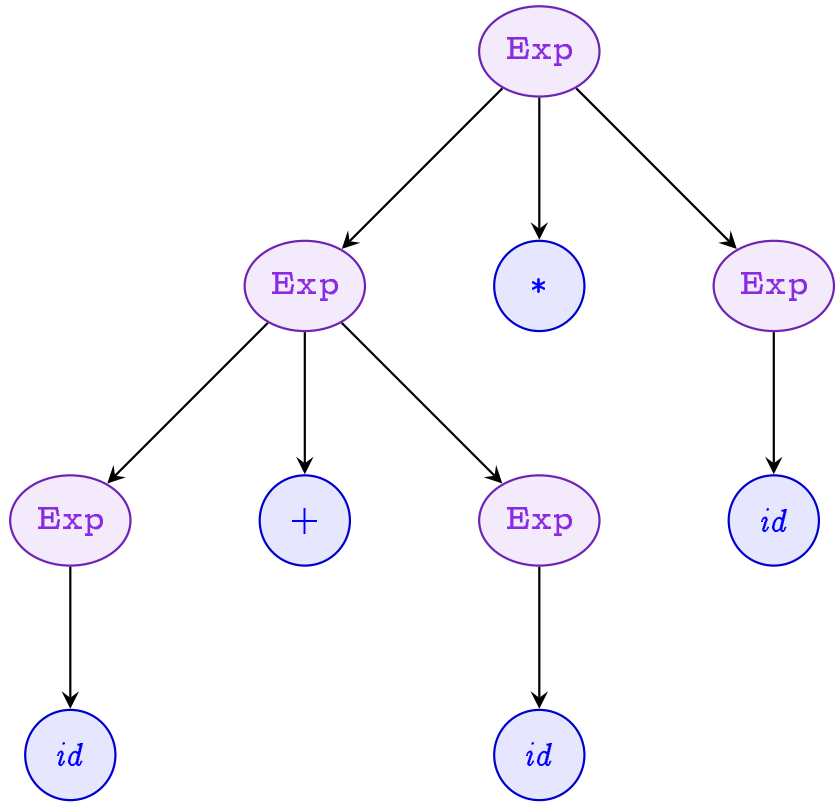
Näide: eelnevalt toodud vasak- ja paremderivatsioonile vastab mõlemal juhul sama süntaksipuu



# Kontekstivabad grammatikad

NB! Ühel lausel võib olla mitu erinevat süntaksipuud!

$$
\begin{aligned}
\mathrm{Exp} &\Longrightarrow_{lm} \mathrm{Exp} * \mathrm{Exp} \\
&\Longrightarrow_{lm} \mathrm{Exp} + \mathrm{Exp} * \mathrm{Exp} \\
&\Longrightarrow_{lm} id + \mathrm{Exp} * \mathrm{Exp} \\
&\Longrightarrow_{lm} id + id * \mathrm{Exp} \\
&\Longrightarrow_{lm} id + id * id
\end{aligned}
$$



# Kontekstivabad grammatikad

- KV-grammatika on mitmene (ambiguous), kui ühe lause jaoks leidub mitu süntaksipuud.
- Iga süntaksipuu jaoks leidub täpselt üks vasak- ja üks paremderivatsioon; seega:
  - ühesel lausel on täpselt üks vasak- ja üks paremderivatsioon;
  - mitmesel lausel on vähemalt kaks vasak- ja paremderivatsiooni.
- Lause erinevad süntaksipuud vastavad reeglina lause semantiliselt erinevatele tõlgendusvõimalustele.
- Mitmese grammatika saab teatud juhtudel (aga mitte alati) teisendada ekvivalentseks üheseks grammatikaks.

# Kontekstivabad grammatikad

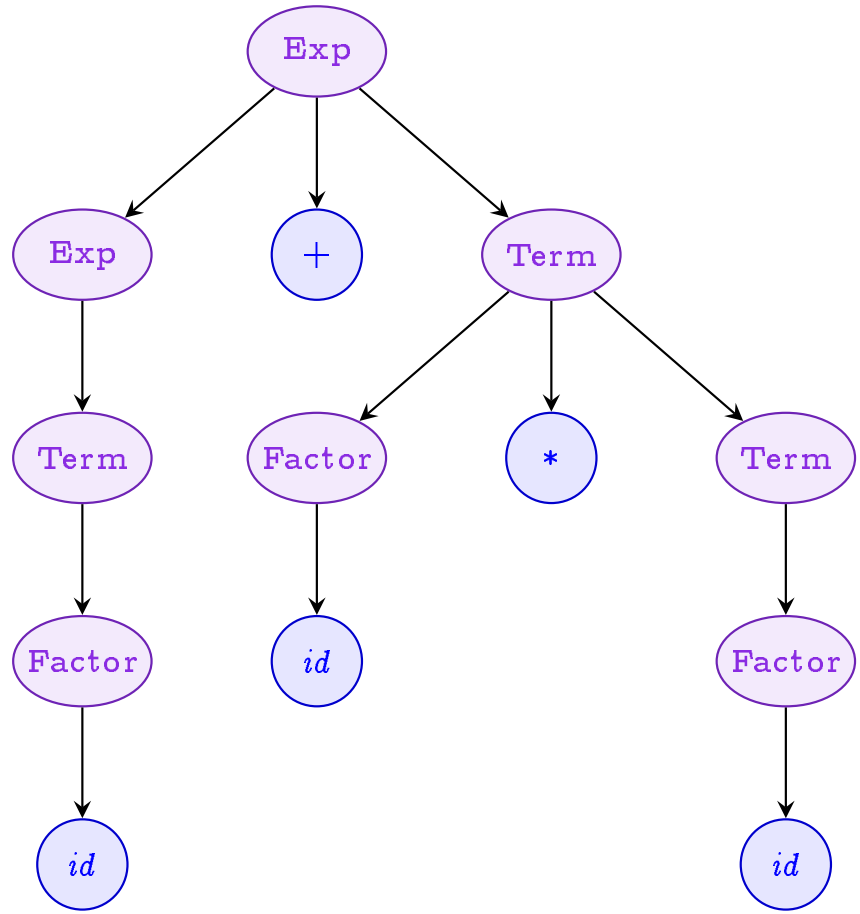
- Mitmesuse eemaldamine – binaarsed operaatorid:
  - iga prioriteeditaseme jaoks toome sisse uue mitteterminaali;
  - vasakassotsiatiivse operaatori korral kasutame vasakrekursiivset; paremassotsiatiivse korral paremrekursiivset produktsioonireeglit;
  - tugevama prioriteediga operaatoritele vastavad reeglid paigutame "sügavamale".
- Näide:

$$
\begin{array}{lcl}
\mathrm{Exp} & \rightarrow & \mathrm{Exp} + \mathrm{Term} \\
 & | & \mathrm{Term} \\
\mathrm{Term} & \rightarrow & \mathrm{Factor} * \mathrm{Term} \\
 & | & \mathrm{Factor} \\
\mathrm{Factor} & \rightarrow & (\ \mathrm{Exp}\ ) \\
 & | & id
\end{array}
$$

# Kontekstivabad grammatikad

Näide:

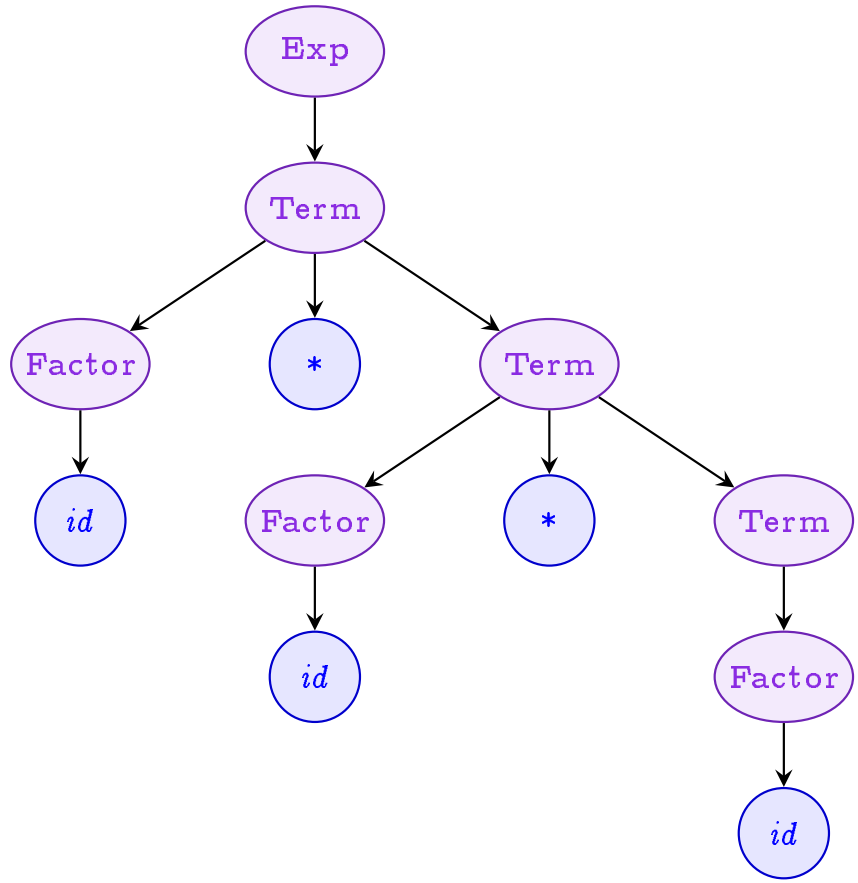
$$
\begin{aligned}
\mathtt{Exp} &\Longrightarrow_{lm} \mathtt{Exp} + \mathtt{Term} \\
&\Longrightarrow_{lm} \mathtt{Term} + \mathtt{Term} \\
&\Longrightarrow_{lm} \mathtt{Factor} + \mathtt{Term} \\
&\Longrightarrow_{lm} id + \mathtt{Term} \\
&\Longrightarrow_{lm} id + \mathtt{Factor} * \mathtt{Term} \\
&\Longrightarrow_{lm} id + id * \mathtt{Term} \\
&\Longrightarrow_{lm} id + id * \mathtt{Factor} \\
&\Longrightarrow_{lm} id + id * id
\end{aligned}
$$



# Kontekstivabad grammatikad

Näide:

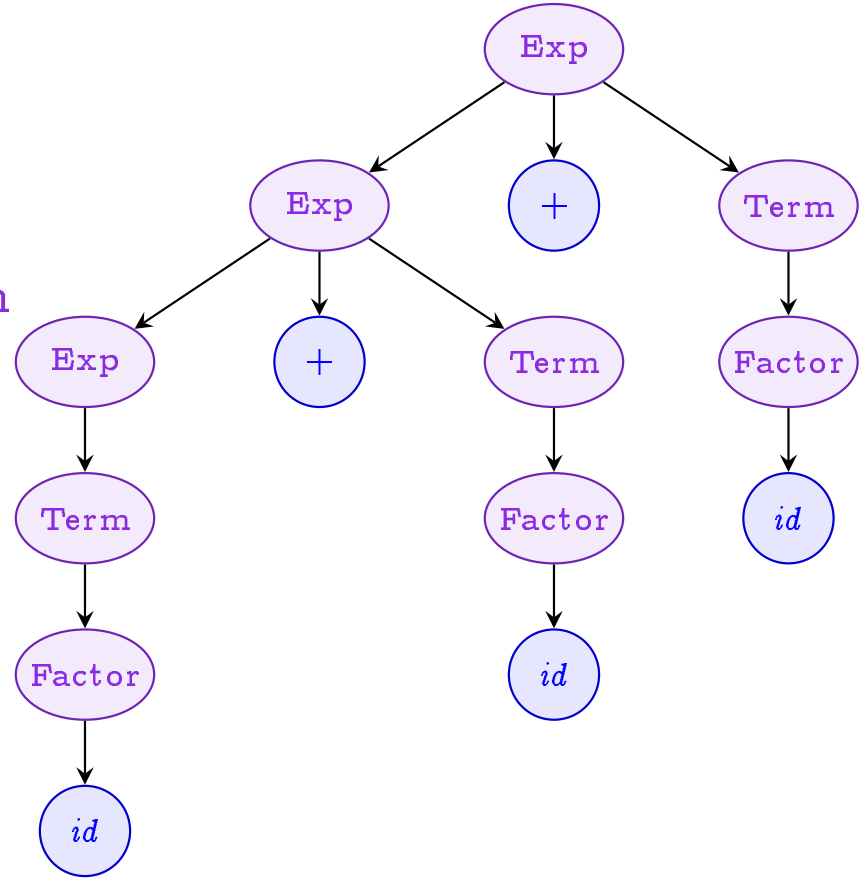
$$
\begin{aligned}
\mathrm{Exp} &\Longrightarrow_{lm} \mathrm{Term} \\
&\Longrightarrow_{lm} \mathrm{Factor} * \mathrm{Term} \\
&\Longrightarrow_{lm} id * \mathrm{Term} \\
&\Longrightarrow_{lm} id * \mathrm{Factor} * \mathrm{Term} \\
&\Longrightarrow_{lm} id * id * \mathrm{Term} \\
&\Longrightarrow_{lm} id * id * \mathrm{Factor} \\
&\Longrightarrow_{lm} id * id * id
\end{aligned}
$$



# Kontekstivabad grammatikad

Näide:

$$
\begin{aligned}
\text{Exp} &\Longrightarrow_{lm} \text{Exp} + \text{Term} \\
&\Longrightarrow_{lm} \text{Exp} + \text{Term} + \text{Term} \\
&\Longrightarrow_{lm} \text{Term} + \text{Term} + \text{Term} \\
&\Longrightarrow_{lm} \text{Factor} + \text{Term} + \text{Term} \\
&\Longrightarrow_{lm} id + \text{Term} + \text{Term} \\
&\Longrightarrow_{lm} id + \text{Factor} + \text{Term} \\
&\Longrightarrow_{lm} id + id + \text{Term} \\
&\Longrightarrow_{lm} id + id + \text{Factor} \\
&\Longrightarrow_{lm} id + id + id
\end{aligned}
$$



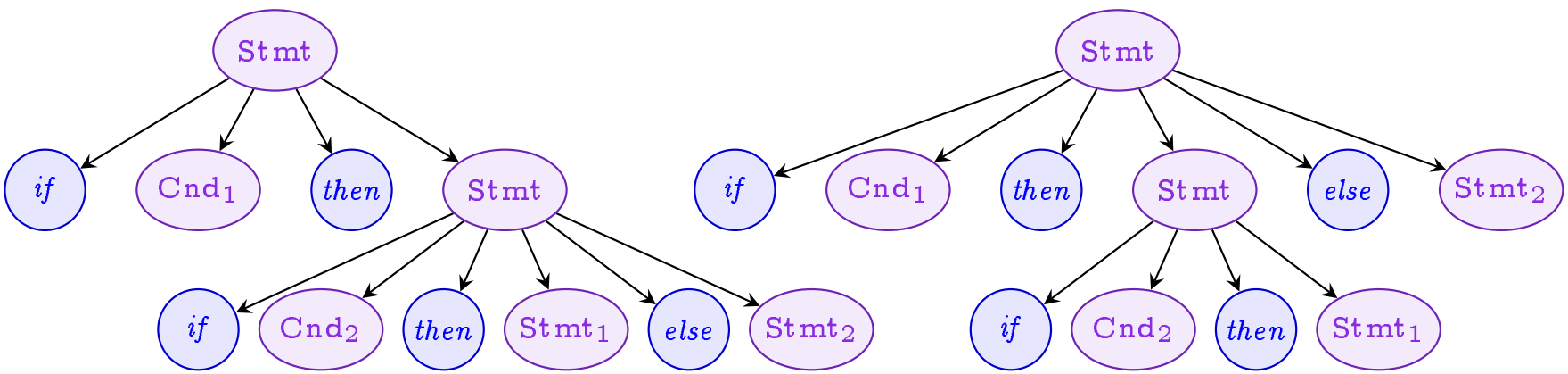
# Kontekstivabad grammatikad

- Mitmesuse eemaldamine – tingimuslaused:

$$\begin{aligned} \text{Stmt} &\rightarrow \textit{if}\ \text{Cnd}\ \textit{then}\ \text{Stmt} \\ &\mid \textit{if}\ \text{Cnd}\ \textit{then}\ \text{Stmt}\ \textit{else}\ \text{Stmt} \\ &\mid \text{Other} \end{aligned}$$

- Järgnev lausevorm omab kahte süntaksipuud:

$$\textit{if}\ \text{Cnd}_1\ \textit{then}\ \textit{if}\ \text{Cnd}_2\ \textit{then}\ \text{Stmt}_1\ \textit{else}\ \text{Stmt}_2$$

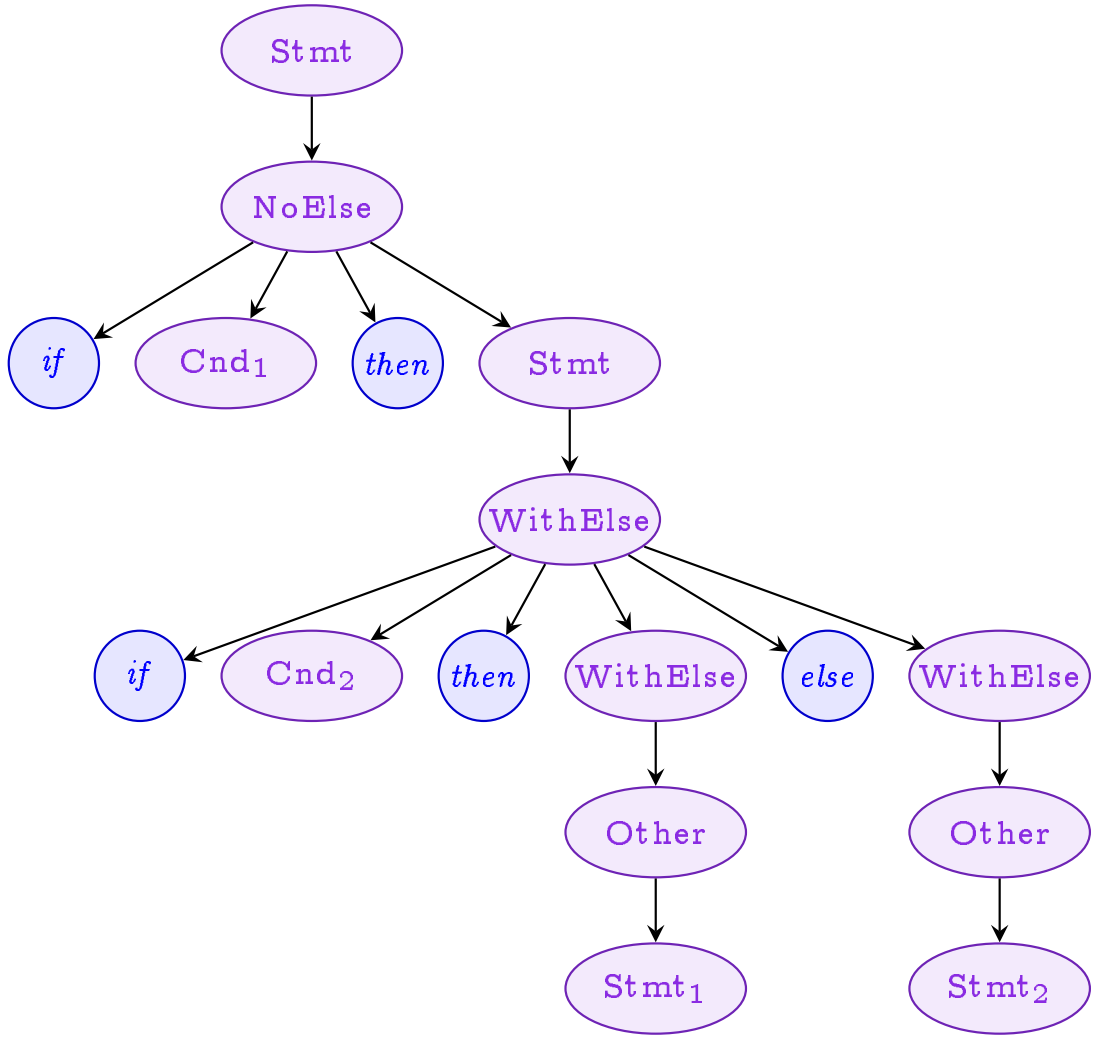


# Kontekstivabad grammatikad

Tavaliselt loetakse korrektseks esimest süntaksipuud; so. *else* kuulub alati kõige sisemise võimaliku tingimuslause juurde:

$$
\begin{array}{lcl}
\text{Stmt} & \rightarrow & \text{WithElse} \\
 & | & \text{NoElse} \\
\text{WithElse} & \rightarrow & \textit{if}\ \text{Cnd}\ \textit{then}\ \text{WithElse}\ \textit{else}\ \text{WithElse} \\
 & | & \text{Other} \\
\text{NoElse} & \rightarrow & \textit{if}\ \text{Cnd}\ \textit{then}\ \text{Stmt} \\
 & | & \textit{if}\ \text{Cnd}\ \textit{then}\ \text{WithElse}\ \textit{else}\ \text{NoElse}
\end{array}
$$

# Kontekstivabad grammatikad



# Parsimistehnikad

Ülalt-alla parsimine (top-down parsing):

- alustab süntaksipuu ehitamist juurest ning kasvatab seda lehtede suunas;
- igal sammul valib produktsioonireegli ning üritab seda sobitada sisendsõnaga;
- mittesobiva reegli korral toimub tagasipöördumine (backtracking);
- reeglina annab vasakpoolse derivatsiooni.

Alt-üles parsimine (bottom-up parsing)

- alustab süntaksipuu ehitamist lehtedest ning kasvatab seda juure suunas;
- rakendab sobivaid reegleid paremalt vasakule kuni jõuab algsümbolini;
- reeglina annab parempoolse derivatsiooni.

# Ülalt-alla parsimine

Ülalt-alla parsimise üldine algoritm:

- konstrueerime juurtipu, märgendame selle algsümboliga ja jätkame puu konstrueerimist lehteded suunas sügavuti vasakult paremale;
- kui vaatluse all olev tipp on mitteterminaal $A$, siis valime mingi produktsioonireegli kujul $A \to \alpha$, konstrueerime reegli paremale poolele vastavad tipud, ning jätkame vasakpoolseima alamtipuga;
- kui tipp on terminaalsümbol mis ei sobi sisendsümboliga, siis teostame tagasipöördumise produktsioonireegli valikuni mis selle tipu konstrueeris ja jätkame sealt uuesti valides teise reegli;
- kui tipp on terminaalsümbol mis sobib sisendsümboliga, siis jätkame järgmise vasakpoolseima läbivaatamata tipuga.

# Ülalt-alla parsimine

Näide:

| | | |
|---|---|---|
| E | → | E + T |
| | \| | E − T |
| | \| | T |
| T | → | T * F |
| | \| | T / F |
| | \| | F |
| F | → | ( E ) |
| | \| | *id* |
| | \| | *num* |

| | |
|---|---|
| | • *id* − *num* ∗ *id* |
| *E*.1 | • *id* − *num* ∗ *id* |
| *E*.3 | • *id* − *num* ∗ *id* |
| *T*.3 | • *id* − *num* ∗ *id* |
| *F*.2 | • *id* − *num* ∗ *id* |
| | *id* • − *num* ∗ *id* |
| | *id* − • *num* ∗ *id* |
| *T*.1 | *id* − • *num* ∗ *id* |
| *T*.3 | *id* − • *num* ∗ *id* |
| *F*.3 | *id* − • *num* ∗ *id* |
| | *id* − *num* • ∗ *id* |
| | *id* − *num* ∗ • *id* |
| *F*.2 | *id* − *num* ∗ • *id* |
| | *id* − *num* ∗ *id* • |

# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \ / \ \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

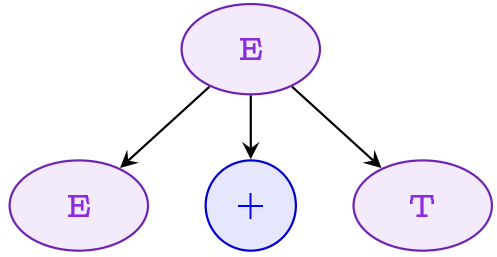
$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.1 & id \bullet - num * id \\
E.3 & id - \bullet\ num * id \\
T.3 & id - num \bullet * id \\
F.2 & id - num * \bullet\ id \\
 & id - num * id \bullet
\end{array}
$$

# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.1 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

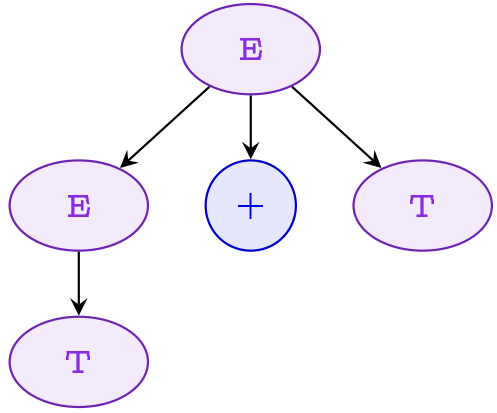


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.1 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

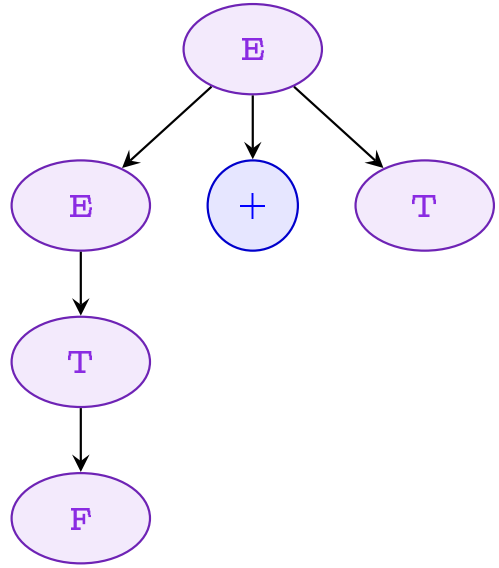


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \ / \ \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.1 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

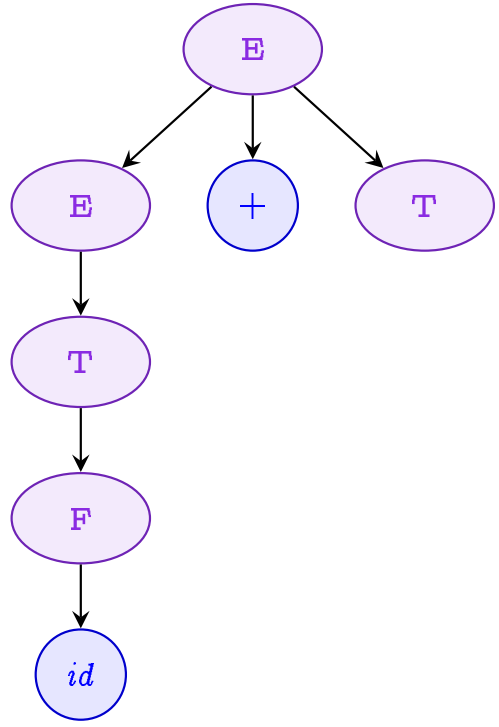


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.1 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

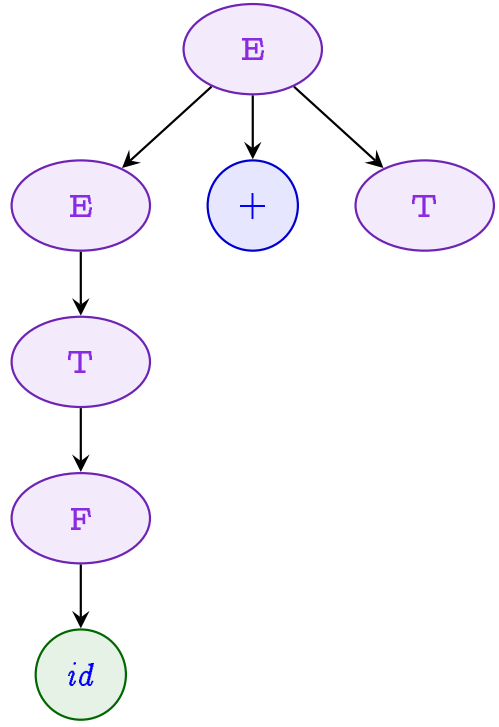


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \to & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \to & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \to & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.1 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

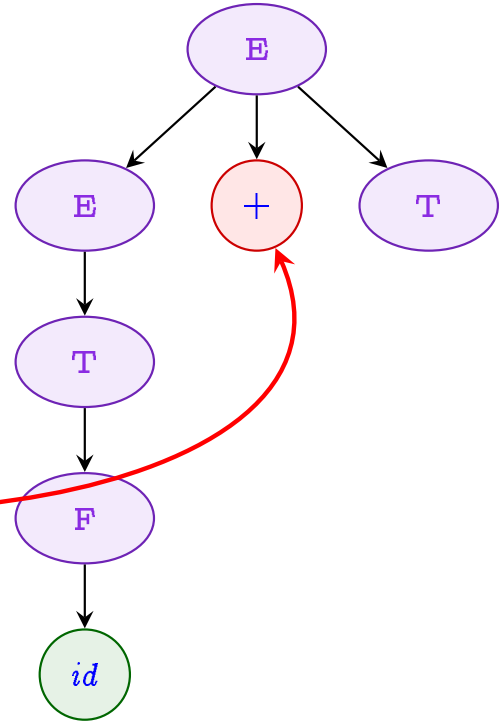


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.1 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$



# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \ / \ \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

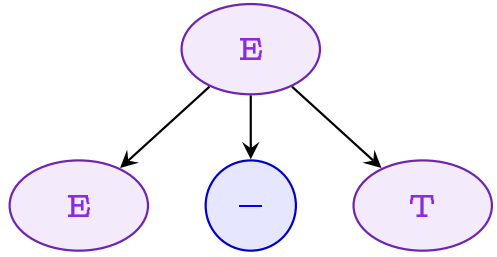
$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.1 & id \bullet - num * id \\
E.3 & id - \bullet\ num * id \\
T.3 & id - num \bullet * id \\
F.2 & id - num * \bullet\ id \\
 & id - num * id \bullet
\end{array}
$$

# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \ / \ \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

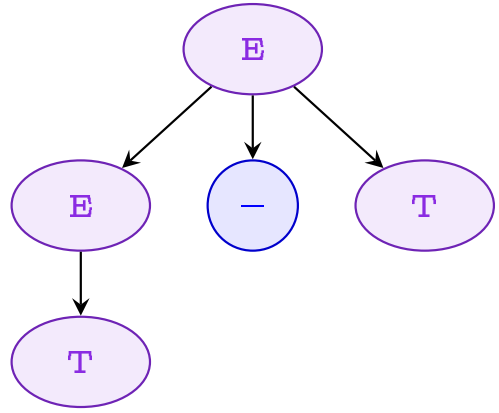


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \ / \ \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

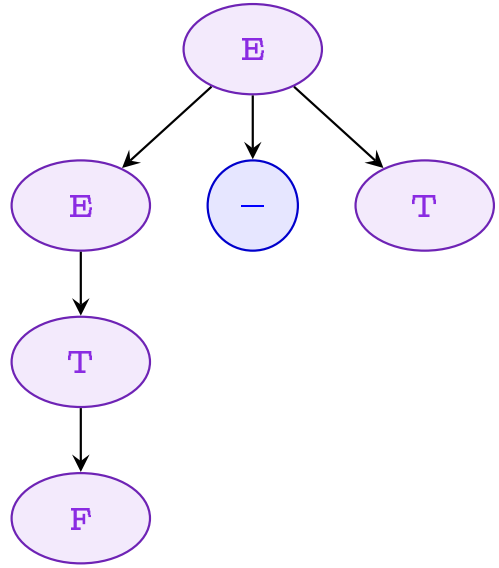


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \ / \ \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

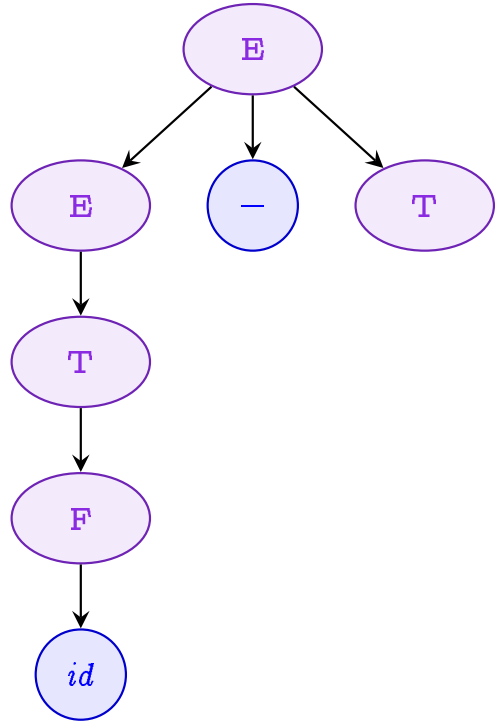


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

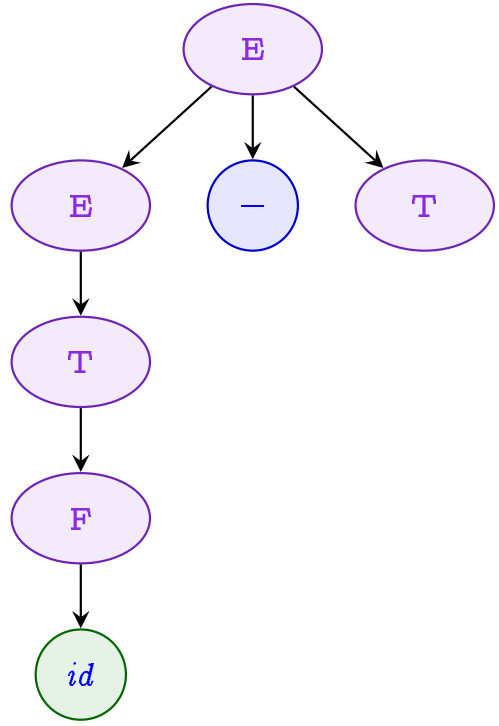


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet\ num * id \\
T.1 & id - \bullet\ num * id \\
T.3 & id - \bullet\ num * id \\
F.3 & id - \bullet\ num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet\ id \\
F.2 & id - num * \bullet\ id \\
 & id - num * id \bullet
\end{array}
$$

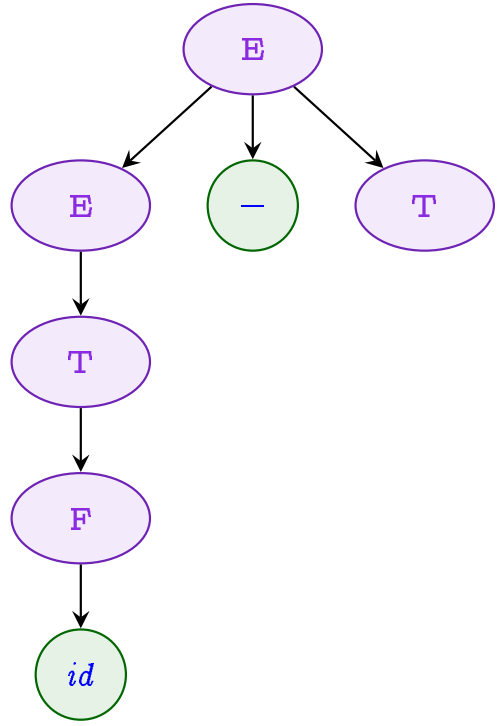


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

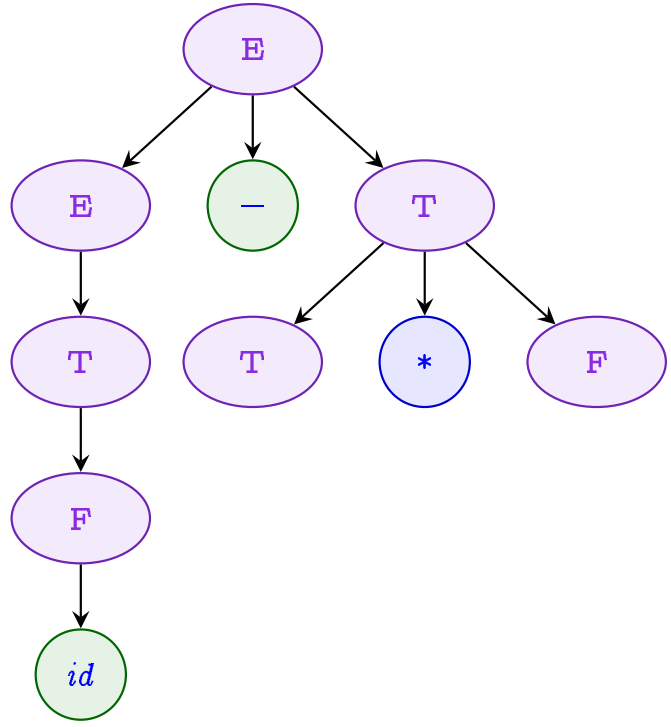


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

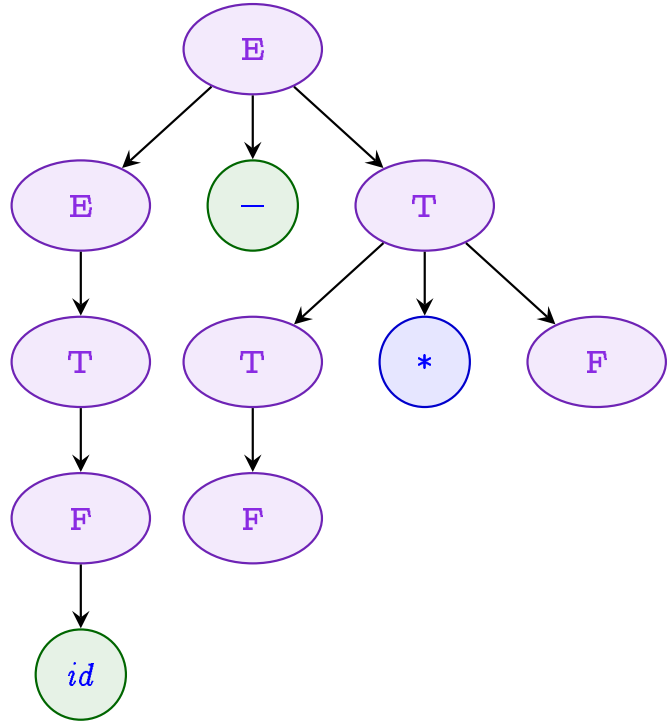


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

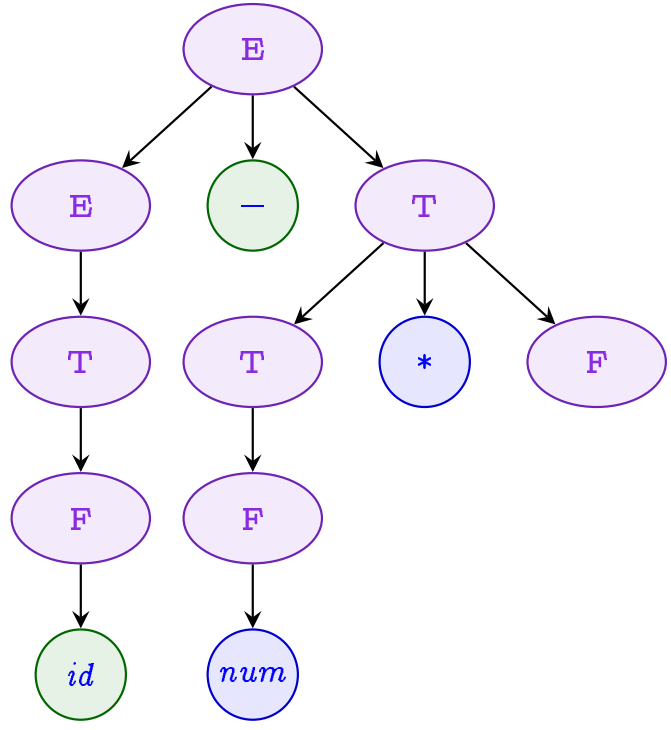


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

| | |
|---|---|
| | $\bullet\ id - num * id$ |
| $E.2$ | $\bullet\ id - num * id$ |
| $E.3$ | $\bullet\ id - num * id$ |
| $T.3$ | $\bullet\ id - num * id$ |
| $F.2$ | $\bullet\ id - num * id$ |
| | $id \bullet - num * id$ |
| | $id - \bullet num * id$ |
| $T.1$ | $id - \bullet num * id$ |
| $T.3$ | $id - \bullet num * id$ |
| $F.3$ | $id - \bullet num * id$ |
| | $id - num \bullet * id$ |
| | $id - num * \bullet id$ |
| $F.2$ | $id - num * \bullet id$ |
| | $id - num * id \bullet$ |

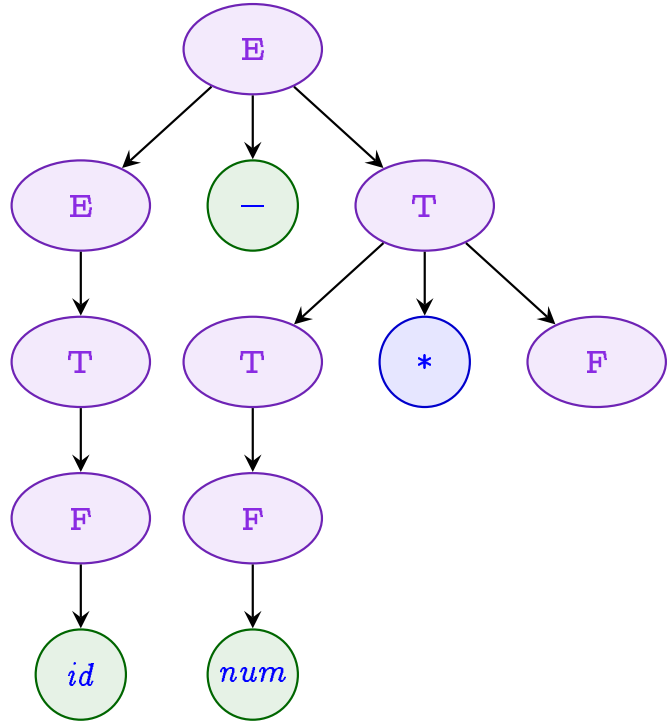


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

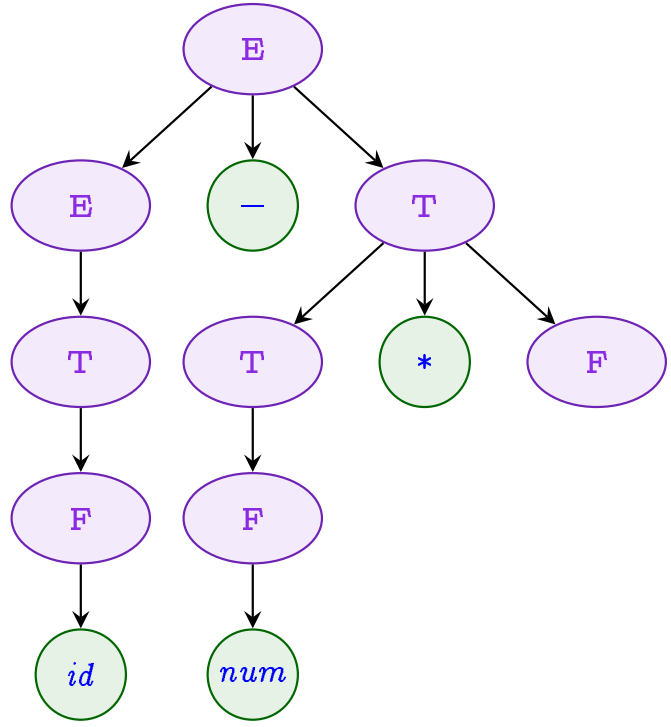


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \ / \ \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

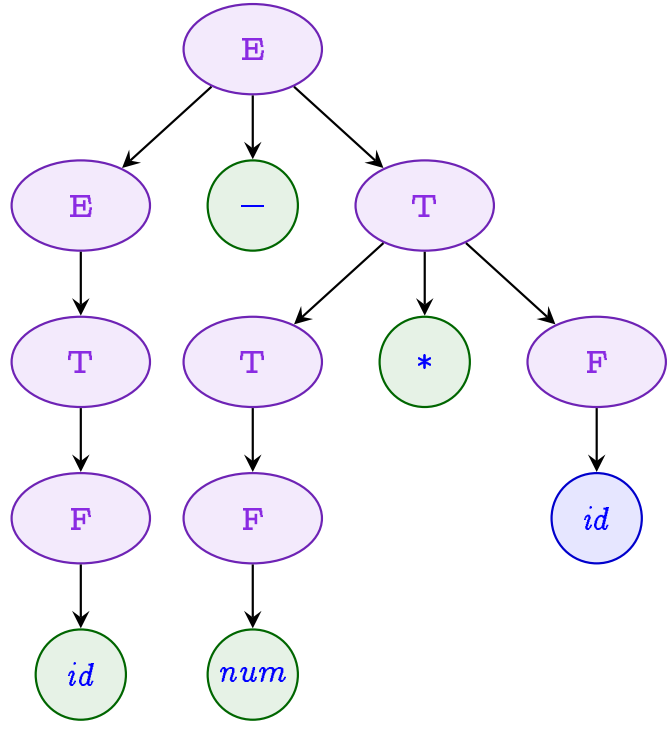


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \rightarrow & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \rightarrow & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \ / \ \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \rightarrow & (\ \mathtt{E}\ ) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$

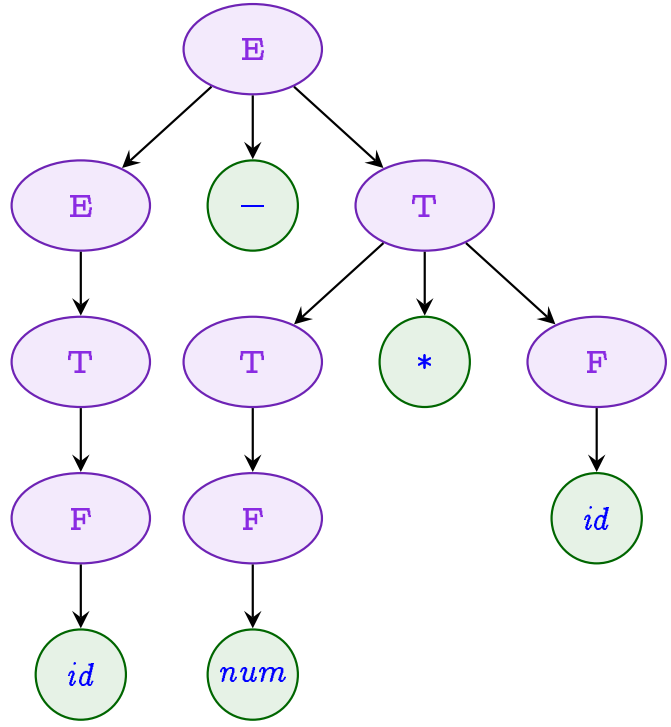


# Ülalt-alla parsimine

Näide:

$$
\begin{array}{lcl}
\mathtt{E} & \to & \mathtt{E} + \mathtt{T} \\
 & | & \mathtt{E} - \mathtt{T} \\
 & | & \mathtt{T} \\
\mathtt{T} & \to & \mathtt{T} * \mathtt{F} \\
 & | & \mathtt{T} \;/\; \mathtt{F} \\
 & | & \mathtt{F} \\
\mathtt{F} & \to & (\; \mathtt{E} \;) \\
 & | & id \\
 & | & num
\end{array}
$$

$$
\begin{array}{ll}
 & \bullet\ id - num * id \\
E.2 & \bullet\ id - num * id \\
E.3 & \bullet\ id - num * id \\
T.3 & \bullet\ id - num * id \\
F.2 & \bullet\ id - num * id \\
 & id \bullet - num * id \\
 & id - \bullet num * id \\
T.1 & id - \bullet num * id \\
T.3 & id - \bullet num * id \\
F.3 & id - \bullet num * id \\
 & id - num \bullet * id \\
 & id - num * \bullet id \\
F.2 & id - num * \bullet id \\
 & id - num * id \bullet
\end{array}
$$



# Ülalt-alla parsimine

- Parsimise efektiivsus sõltub tugevasti õige produktsioonireegli valikust.
- Vale reegli valimine põhjustab hilisema tagasipöördumise vajaduse.
- Vasakrekursiivsete reeglite olemasolul võib ülalt-alla parsimine mittetermineeruda.

- $id - num * id$

$E.2$ • $id - num * id$

$E.2$ • $id - num * id$

$E.2$ • $id - num * id$

...

# Ülalt-alla parsimine

- Parsimise efektiivsus sõltub tugevasti õige produktsioonireegli valikust.
- Vale reegli valimine põhjustab hilisema tagasipöördumise vajaduse.
- Vasakrekursiivsete reeglite olemasolul võib ülalt-alla parsimine mittetermineeruda.


- $id - num * id$

$E.2$
- $id - num * id$

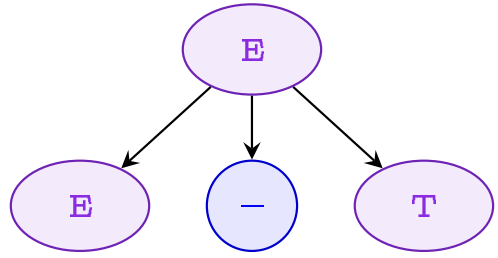
$E.2$
- $id - num * id$

$E.2$
- $id - num * id$

...

# Ülalt-alla parsimine

- Parsimise efektiivsus sõltub tugevasti õige produktsioonireegli valikust.
- Vale reegli valimine põhjustab hilisema tagasipöördumise vajaduse.
- Vasakrekursiivsete reeglite olemasolul võib ülalt-alla parsimine mittetermineeruda.




- $id - num * id$

$E.2$ • $id - num * id$

$E.2$ • $id - num * id$

$E.2$ • $id - num * id$

...

# Ülalt-alla parsimine

- Parsimise efektiivsus sõltub tugevasti õige produktsioonireegli valikust.
- Vale reegli valimine põhjustab hilisema tagasipöördumise vajaduse.
- Vasakrekursiivsete reeglite olemasolul võib ülalt-alla parsimine mittetermineeruda.

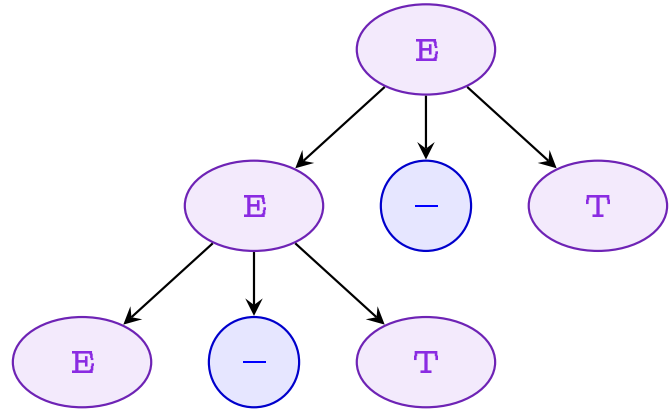
- $id - num * id$

$E.2$ • $id - num * id$

$E.2$ • $id - num * id$

$E.2$ • $id - num * id$

...



# Ülalt-alla parsimine

- Parsimise efektiivsus sõltub tugevasti õige produktsioonireegli valikust.
- Vale reegli valimine põhjustab hilisema tagasipöördumise vajaduse.
- Vasakrekursiivsete reeglite olemasolul võib ülalt-alla parsimine mittetermineeruda.

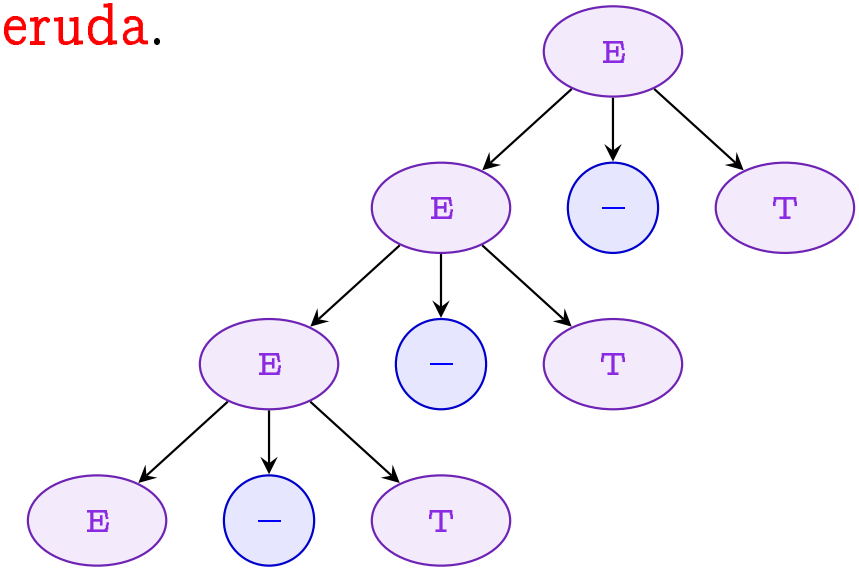
- $id - num * id$

$E.2$ • $id - num * id$

$E.2$ • $id - num * id$

$E.2$ • $id - num * id$

. . .

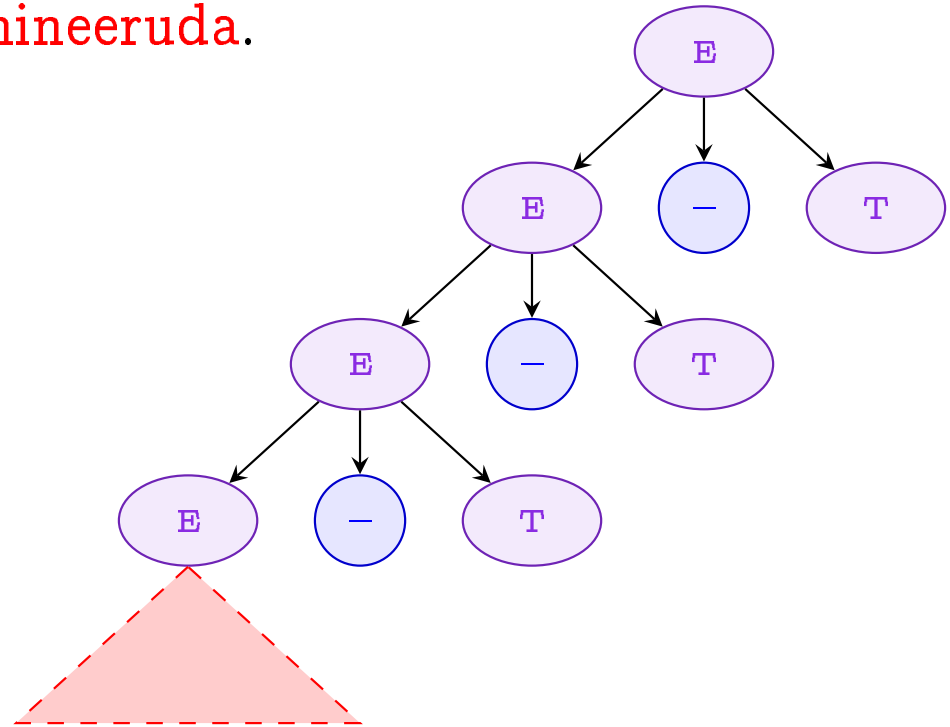


# Ülalt-alla parsimine

- Parsimise efektiivsus sõltub tugevasti õige produktsioonireegli valikust.
- Vale reegli valimine põhjustab hilisema tagasipöördumise vajaduse.
- Vasakrekursiivsete reeglite olemasolul võib ülalt-alla parsimine mittetermineeruda.

- $id - num * id$
- $E.2$ $id - num * id$
- $E.2$ $id - num * id$
- $E.2$ $id - num * id$

...



# Vasakrekursioon

- Grammatika on vasakrekursiivne, kui leidub mitteterminal $A \in N$ selline, et

$$A \Longrightarrow^{+} A\,\alpha,$$

  kus $\alpha \in V^{\star}$.
- Vasakrekursioon on vahetu, kui leidub reegel kujul $A \to A\,\alpha$.
- Vastasel korral on vasakrekursioon kaudne.

# Vasakrekursiooni elimineerimine

Vahetu vasakrekursiooni eemaldamine:

- Toome sisse uue mitteterminali ning asendame vasakrekursiooni paremrekursiooniga

$$A \rightarrow A\ \alpha \mid \beta \quad \longrightarrow \quad \begin{array}{lll} A & \rightarrow & \beta\ A' \\ A' & \rightarrow & \alpha\ A' \mid \varepsilon \end{array}$$

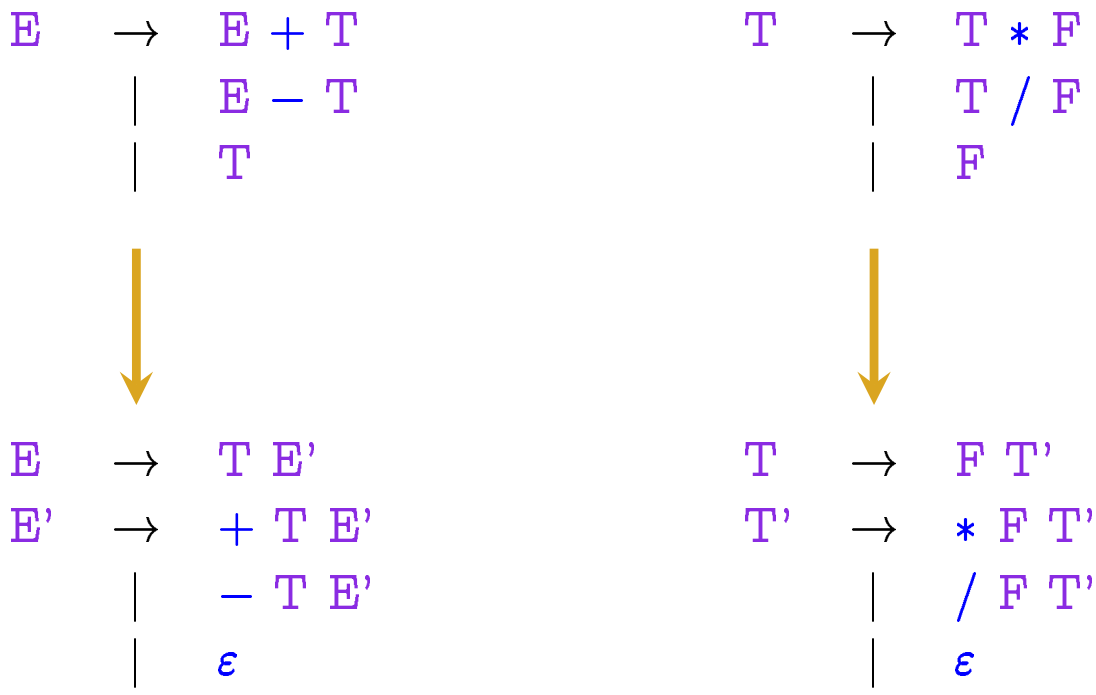
- Üldjuhul

$$A \rightarrow A\ \alpha_1 \mid A\ \alpha_2 \mid \ldots \mid \beta_1 \mid \beta_2 \mid \ldots$$

$$\longrightarrow \quad \begin{array}{lll} A & \rightarrow & \beta_1\ A' \mid \beta_2\ A' \mid \ldots \\ A' & \rightarrow & \alpha_2\ A' \mid \alpha_2\ A' \mid \ldots \mid \varepsilon \end{array}$$

# Vasakrekursiooni elimineerimine

Näide:




- Uus grammatika genereerib sama keele, kuid on natuke vähem intuitiivne.
- Mõlemates grammatikates on operaatorid vasakassotsiatiivsed.

# Vasakrekursiooni elimineerimine

Kaudse vasakrekursiooni eemaldamine:

- Näide:

$$\begin{aligned} A_1 &\rightarrow A_2\ \alpha \mid \beta \\ A_2 &\rightarrow A_1\ \gamma \mid A_2\ \delta \end{aligned}$$

- Teisendame kaudse vasakrekursiooni vahetuks.
- Asendame mitteterminali $A_2$ reeglite paremates pooltes kõik $A_1$ esinemised tema definitsiooniga:

$$\begin{aligned} A_1 &\rightarrow A_2\ \alpha \mid \beta \\ A_2 &\rightarrow A_2\ \alpha\gamma \mid \beta\gamma \mid A_2\ \delta \end{aligned}$$

- Eemaldame vahetu vasakrekursiooni:

$$\begin{aligned} A_1 &\rightarrow A_2\ \alpha \mid \beta \\ A_2 &\rightarrow \beta\ \gamma\ A_2' \\ A_2' &\rightarrow \alpha\ \gamma\ A_2' \mid \delta\ A_2' \mid \varepsilon \end{aligned}$$

# Vasakrekursiooni elimineerimine

Üldine algoritm kaudse vasakrekursiooni eemaldamiseks:

Järjestame mitteterminalid mingisse järjekorda $A_1, \ldots, A_n$

**for** $i \leftarrow 1$ **to** $n$

**for** $j \leftarrow 1$ **to** $i - 1$

Asendame iga produktsiooni kujul $A_i \to A_j\ \alpha$
produktsioonidega $A_i \to \beta_1\ \alpha \mid \beta_2\ \alpha \mid \ldots \mid \beta_k\ \alpha$,
kus $A_j \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_k$ on kõik $A_j$ produktsioonid

Eemaldame vahetu vasakrekursiooni mitteterminali $A_i$
produktsioonidest

NB! Eeldab, et algses grammatikas pole ei $\varepsilon$-produktsioone, ega tsükleid (so. $A_i \Longrightarrow^+ A_i$).

# Vasakrekursiooni elimineerimine

Näide:

$$
\begin{array}{lcl}
A & \to & C\ \alpha \mid A\ \alpha \\
B & \to & A\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi
\end{array}
$$

$\longrightarrow$

$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & A\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$

$\longrightarrow$

$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$

$\longrightarrow$

$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & C\ \alpha\ A'\ \beta\ \delta \mid \gamma\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$

$\longrightarrow$

$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & \alpha\ A'\ \beta\ \delta\ C' \\
A' & \to & \alpha\ A' \mid \varepsilon \\
C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{array}
$$

$\longrightarrow$

$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
C & \to & \alpha\ A'\ \beta\ \delta\ C' \\
A' & \to & \alpha\ A' \mid \varepsilon \\
C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{array}
$$

# Vasakrekursiooni elimineerimine

Näide:

$$\begin{array}{lcl} A & \to & C\ \alpha \mid A\ \alpha \\ B & \to & A\ \beta \mid \gamma \\ C & \to & B\ \delta \mid \varphi \end{array} \longrightarrow \begin{array}{lcl} A & \to & C\ \alpha\ A' \\ B & \to & A\ \beta \mid \gamma \\ C & \to & B\ \delta \mid \varphi \\ A' & \to & \alpha\ A' \mid \varepsilon \end{array}$$

$$\longrightarrow \begin{array}{lcl} A & \to & C\ \alpha\ A' \\ B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\ C & \to & B\ \delta \mid \varphi \\ A' & \to & \alpha\ A' \mid \varepsilon \end{array} \longrightarrow \begin{array}{lcl} A & \to & C\ \alpha\ A' \\ B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\ C & \to & C\ \alpha\ A'\ \beta\ \delta \mid \gamma\ \delta \mid \varphi \\ A' & \to & \alpha\ A' \mid \varepsilon \end{array}$$

$$\longrightarrow \begin{array}{lcl} A & \to & C\ \alpha\ A' \\ B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\ C & \to & \alpha\ A'\ \beta\ \delta\ C' \\ A' & \to & \alpha\ A' \mid \varepsilon \\ C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon \end{array} \longrightarrow \begin{array}{lcl} A & \to & C\ \alpha\ A' \\ C & \to & \alpha\ A'\ \beta\ \delta\ C' \\ A' & \to & \alpha\ A' \mid \varepsilon \\ C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon \end{array}$$

# Vasakrekursiooni elimineerimine

Näide:

$$
\begin{array}{lcl}
A & \to & C\ \alpha \mid A\ \alpha \\
B & \to & A\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi
\end{array}
$$



$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & A\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$

$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$



$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & C\ \alpha\ A'\ \beta\ \delta \mid \gamma\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$



$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & \alpha\ A'\ \beta\ \delta\ C' \\
A' & \to & \alpha\ A' \mid \varepsilon \\
C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{array}
$$



$$
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
C & \to & \alpha\ A'\ \beta\ \delta\ C' \\
A' & \to & \alpha\ A' \mid \varepsilon \\
C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{array}
$$

# Vasakrekursiooni elimineerimine

Näide:

$$
\begin{array}{lcl}
A & \to & C\ \alpha \mid A\ \alpha \\
B & \to & A\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi
\end{array}
\quad\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & A\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$

$$
\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
\quad\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & C\ \alpha\ A'\ \beta\ \delta \mid \gamma\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$

$$
\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & \alpha\ A'\ \beta\ \delta\ C' \\
A' & \to & \alpha\ A' \mid \varepsilon \\
C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{array}
\quad\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
C & \to & \alpha\ A'\ \beta\ \delta\ C' \\
A' & \to & \alpha\ A' \mid \varepsilon \\
C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{array}
$$

# Vasakrekursiooni elimineerimine

Näide:

$$
\begin{array}{lcl}
A & \to & C\ \alpha \mid A\ \alpha \\
B & \to & A\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi
\end{array}
\quad\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & A\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$

$$
\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & B\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
\quad\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & C\ \alpha\ A'\ \beta\ \delta \mid \gamma\ \delta \mid \varphi \\
A' & \to & \alpha\ A' \mid \varepsilon
\end{array}
$$

$$
\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
B & \to & C\ \alpha\ A'\ \beta \mid \gamma \\
C & \to & \alpha\ A'\ \beta\ \delta\ C' \\
A' & \to & \alpha\ A' \mid \varepsilon \\
C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{array}
\quad\longrightarrow\quad
\begin{array}{lcl}
A & \to & C\ \alpha\ A' \\
C & \to & \alpha\ A'\ \beta\ \delta\ C' \\
A' & \to & \alpha\ A' \mid \varepsilon \\
C' & \to & \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{array}
$$

# Vasakrekursiooni elimineerimine

Näide:

$$
\begin{aligned}
A &\to C\ \alpha \mid A\ \alpha \\
B &\to A\ \beta \mid \gamma \\
C &\to B\ \delta \mid \varphi
\end{aligned}
\longrightarrow
\begin{aligned}
A &\to C\ \alpha\ A' \\
B &\to A\ \beta \mid \gamma \\
C &\to B\ \delta \mid \varphi \\
A' &\to \alpha\ A' \mid \varepsilon
\end{aligned}
$$

$$
\longrightarrow
\begin{aligned}
A &\to C\ \alpha\ A' \\
B &\to C\ \alpha\ A'\ \beta \mid \gamma \\
C &\to B\ \delta \mid \varphi \\
A' &\to \alpha\ A' \mid \varepsilon
\end{aligned}
\longrightarrow
\begin{aligned}
A &\to C\ \alpha\ A' \\
B &\to C\ \alpha\ A'\ \beta \mid \gamma \\
C &\to C\ \alpha\ A'\ \beta\ \delta \mid \gamma\ \delta \mid \varphi \\
A' &\to \alpha\ A' \mid \varepsilon
\end{aligned}
$$

$$
\longrightarrow
\begin{aligned}
A &\to C\ \alpha\ A' \\
B &\to C\ \alpha\ A'\ \beta \mid \gamma \\
C &\to \alpha\ A'\ \beta\ \delta\ C' \\
A' &\to \alpha\ A' \mid \varepsilon \\
C' &\to \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{aligned}
\longrightarrow
\begin{aligned}
A &\to C\ \alpha\ A' \\
C &\to \alpha\ A'\ \beta\ \delta\ C' \\
A' &\to \alpha\ A' \mid \varepsilon \\
C' &\to \gamma\ \delta\ C' \mid \varphi\ C' \mid \varepsilon
\end{aligned}
$$

# Ennustav parsimine

- Vale produktsioonireegli valimine põhjustab tagasipöördumise.
- Tihti on võimalik valitava reegli õigsuse üle otsustada mingi arvu sisendsümbolite ettevaatamise teel.
- Üldjuhul on tarvis ette vaadata piiramata arv sümboleid.
  - Näit.: Cocke-Younger-Kasami või Earley algoritmid.
- Ennustav parsimine (predictive parsing) on ülalt-alla parsimine, kus alati on võimalik ige reegel valida nii, et pole vaja tagasipöörduda.
  - Grammatika peab olema selline, et järgmine sisendsümbol (või mingi fikseeritud arv sümboleid) määrab unikaalselt ära valitava reegli.

# Ennustav parsimine

- Iga lausevormi $\alpha \in (N \cup T)^\star$ jaoks defineerime hulga:

$$\begin{aligned} \mathit{first}(\alpha) &= \{a \in T \mid \alpha \Longrightarrow^\star a\,\beta\} \\ &\quad \cup \{\varepsilon \mid \alpha \Longrightarrow^\star \varepsilon\} \end{aligned}$$

  kus $\beta \in (N \cup T)^\star$.

- Iga mitteterminali $A \in N$ jaoks defineerime hulga:

$$\begin{aligned} \mathit{follow}(A) &= \{a \in T \mid S \Longrightarrow^\star \alpha\,A\,a\,\beta\} \\ &\quad \cup \{\$ \mid S \Longrightarrow^\star \alpha\,A\} \end{aligned}$$

  kus $\alpha, \beta \in (N \cup T)^\star$ ja \$ on spetsiaalne sisendilõpu marker.

# Ennustav parsimine

Näide:

$$
\begin{array}{lcl}
S & \to & A\,B\,C \\
A & \to & a\,A \mid \varepsilon
\end{array}
\qquad\qquad
\begin{array}{lcl}
B & \to & b \mid \varepsilon \\
C & \to & c \mid d
\end{array}
$$

$$
\begin{array}{lcl}
\mathit{first}(C) & = & \{c, d\} \\
\mathit{first}(B) & = & \{b, \varepsilon\} \\
\mathit{first}(A) & = & \{a, \varepsilon\} \\
\mathit{first}(S) & = & \mathit{first}(A\,B\,C) \\
 & = & (\mathit{first}(A) \setminus \{\varepsilon\}) \\
 & & \cup\ (\mathit{first}(B) \setminus \{\varepsilon\}) \\
 & & \cup\ \mathit{first}(C) \\
 & = & \{a, b, c, d\}
\end{array}
\qquad\qquad
\begin{array}{lcl}
\mathit{follow}(C) & = & \{\$\} \\
\mathit{follow}(B) & = & \mathit{first}(C) \\
 & = & \{c, d\} \\
\mathit{follow}(A) & = & (\mathit{first}(B) \setminus \{\varepsilon\}) \\
 & & \cup\ \mathit{first}(C) \\
 & = & \{b, c, d\}
\end{array}
$$

# Ennustav parsimine

- Kui grammatikas on reeglid $A \to \alpha$ ja $A \to \beta$ sellised, et $first(\alpha) \cap first(\beta) = \emptyset$, siis on esimese sisendsümboli põhjal võimalik otsustada kumba reeglit valida.
- NB! Kehtib ainult juhul, kui $\varepsilon \notin \{first(\alpha) \cup first(\beta)\}$.
- Vastasel korral tuleb vaadelda ka hulka $follow(A)$.
- Defineerime iga reegli $A \to \alpha$ jaoks hulga:

$$first^+(\alpha) = \begin{cases} (first(\alpha) \setminus \{\varepsilon\}) \cup follow(A), & \text{kui } \varepsilon \in first(\alpha) \\ first(\alpha), & \text{kui } \varepsilon \notin first(\alpha) \end{cases}$$

- Grammatika on LL(1), kui iga (paarikaupa erineva) produktsioonireegli $A \to \alpha$ ja $A \to \beta$ korral

$$first^+(\alpha) \cap first^+(\beta) = \emptyset$$

# Ennustav parsimine

- NB! LL(1) grammatika ei saa olla ei vasakrekursiivne ega mitmene!
- Näide:

$$S \to S\,a \mid \beta$$

- Kui $\beta \neq \varepsilon$
  - Siis $\mathit{first}(\beta) \subseteq \mathit{first}(S) = \mathit{first}(S\,a)$
  - Seega $\mathit{first}^+(S\,a) \cap \mathit{first}^+(\beta) \neq \emptyset$
- Kui $\beta = \varepsilon$
  - Siis $a \in \mathit{first}(S\,a)$ ja $a \in \mathit{follow}(S) = \{a, \$\}$
  - Seega $\mathit{first}^+(S\,a) \cap \mathit{first}^+(\beta) \neq \emptyset$

# Ennustav parsimine

- Paljusid grammatikaid on võimalik teisendada LL(1) kujule kasutades:
  - vasakrekursiooni elimineerimist;
  - vasakfaktoriseerimist;
  - halvimal juhul üldistame natuke grammatikat (ning kontrollime eemaldatud kitsendusi pärast parsimist).
- Vasakfaktoriseerimine (left-factoring) asendab ühise prefiksiga reeglid uute reeglitega, kus ühine prefiks on ainult ühes parempooles.
- Näide:

$$\begin{array}{lcl} A & \to & B\,a\,C\,D \\ & | & B\,a\,C\,E \end{array} \quad \longrightarrow \quad \begin{array}{lcl} A & \to & B\,a\,C\,Z \\ Z & \to & D \\ & | & E \end{array}$$

# Ennustav parsimine

Vasakfaktoriseerimise algoritm:

1. Iga mitteterminali $A \in N$ korral leiame pikima prefixi $\alpha$, mis esineb kahes või rohkemas $A$ produktsioonireegli parempooles.
2. Kui $\alpha \neq \varepsilon$, siis asendame kõik $A$ produktsioonid

$$A \rightarrow \alpha\,\beta_1 \mid \alpha\,\beta_2 \mid \ldots \mid \alpha\,\beta_n \mid \gamma$$

   produktsioonidega

$$\begin{aligned} A &\rightarrow \alpha\,Z \mid \gamma \\ Z &\rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n \end{aligned}$$

   kus $Z \in N$ on uus mitteterminal.
3. Kordame protsessi, kuni ükski parempool ei oma ühist prefiksit.

# Ennustav parsimine

- Rekursiivselt laskuv (recursive descent) parsimine on ennustava parsimise meetod, kus:
  - iga mitteterminali kohta realiseeritakse üks protseduur, mis tunneb ära sellele mitteterminalile vastavad lausevormid;
  - iga protseduur valib sõltuvalt sisendist reegli ning kutsub (rekursiivselt) välja reegli paremas pooles asuvatele mitteterminalidele vastavad protseduurid.
- Rekursiivselt laskuv parsimine on levinuim meetod (lihtsate keelte) parserite käsitsi kirjutamiseks.

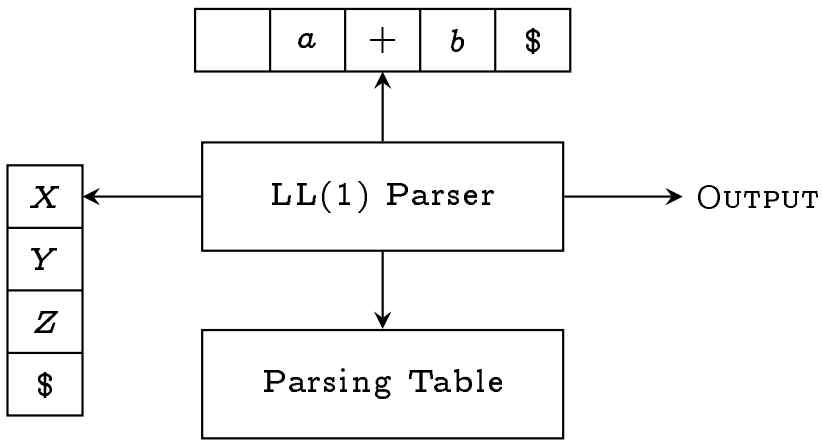
# Ennustav parsimine

Näide: olgu $A$ produktsioonireeglid $A \to a\,B \mid b\,C\,A \mid D\,E$

```
parseA() {
  if token = a then {
    token := nextWord(); parseB();
  } else if token = b then {
    token := nextWord(); parseC(); parseA();
  } else {
    parseD(); parseE();
  }
}
```

# Ennustav parsimine

- Automaatselt genereeritavate LL(1) parserite korral kasutatakse tavaliselt tabeljuhitavat LL(1) parsimist:
  - moodustatakse maatriks $M$, kus read ja veerud on vastavalt indekseeritud mitteterminalide ja terminalidega;
  - tabeli pesades on produktsioonireeglid, mida antud mitteterminali ja sisendsümboli korral valida.
- Tabeljuhitava LL(1) parseri struktuur:



# Ennustav parsimine

LL(1) parsimise algoritm:

$push(\$);\ push(S);$
$token := nextWord();$
**while** $stack \neq empty$ **do** {
  $A := pop();$
  **if** $A \in N$ **then** {
    **if** $M[A, token] = B_1 \ldots B_n$ **then** {
      $push(B_n); \ldots; push(B_1);$
    } **else** $error();$
  } **else if** $A = token$ **then** {
    $token := nextWord();$
  } **else** $error();$
}

# Shift-reduce parsimine

Shift-reduce parsimine on üldine meetod alt-üles süntaksanalüüsiks:

- puu konstrueerimine toimub lehtedest juure suunas eesmärgiga "taandada" sisendstring algsümboliks;
- parsimise ajal on korraga terve mets puid, mis vastavad erinavatele, juba äratuntud, alamstringidele;
- kaks baasaktsiooni:
  - shift loeb uue sisendsümboli;
  - reduce taandab mingi produktsioonireegli parempoolele vastava (juba loetud/taandatud sisendsümbolite ja mitteterminalide) jada reegli vasakpoolel olevaks mitteterminaliks;
- konstruktsioon vastab paremderivatsioonile.

# Shift-reduce parsimine

Näide:

Input String: $\bullet\, a\, b\, c\, c\, d\, e$

$$
\begin{array}{lcl}
S & \rightarrow & a\, A\, B\, e \\
A & \rightarrow & b\, c\, A \mid c \\
B & \rightarrow & d
\end{array}
$$




shift
shift
shift
shift
reduce $A \rightarrow c$
reduce $A \rightarrow b\, c\, A$
shift
reduce $B \rightarrow d$
shift
reduce $S \rightarrow a\, A\, B\, e$

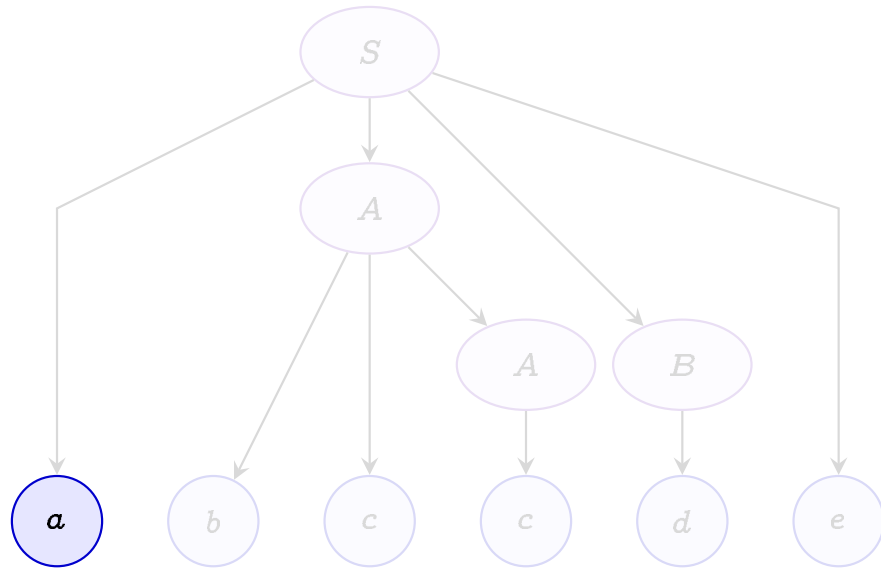
$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\, A\, B\, e & \Longrightarrow_{rm} & a\, A\, d\, e \\
 & \Longrightarrow_{rm} & a\, b\, c\, A\, d\, e & \Longrightarrow_{rm} & a\, b\, c\, c\, d\, e
\end{array}
$$

# Shift-reduce parsimine

Näide:

Input String: $a \bullet b\,c\,c\,d\,e$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$




**shift**

shift

shift

shift

reduce $A \rightarrow c$

reduce $A \rightarrow b\,c\,A$

shift

reduce $B \rightarrow d$

shift

reduce $S \rightarrow a\,A\,B\,e$

$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

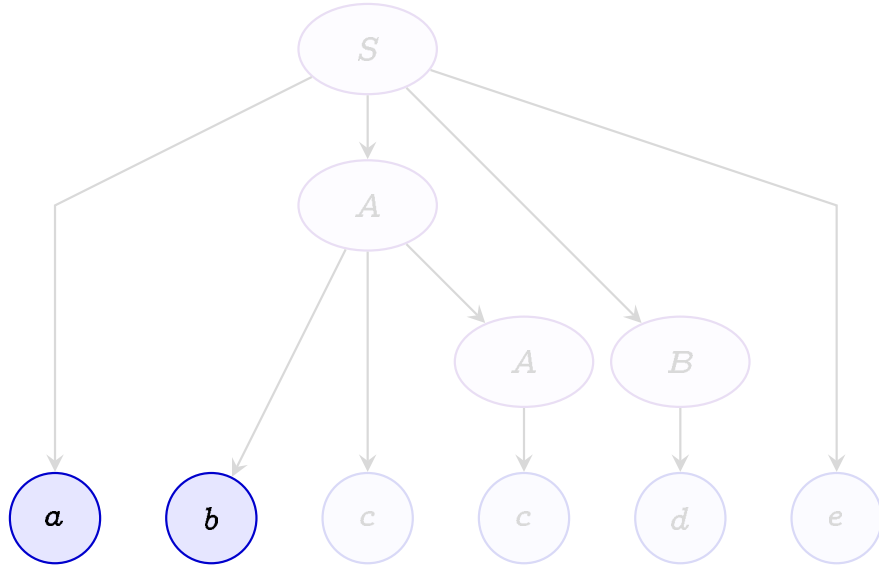
# Shift-reduce parsimine

Näide:

Input String: $a\,b \bullet c\,c\,d\,e$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
shift
shift
reduce $A \rightarrow c$
reduce $A \rightarrow b\,c\,A$
shift
reduce $B \rightarrow d$
shift
reduce $S \rightarrow a\,A\,B\,e$




$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

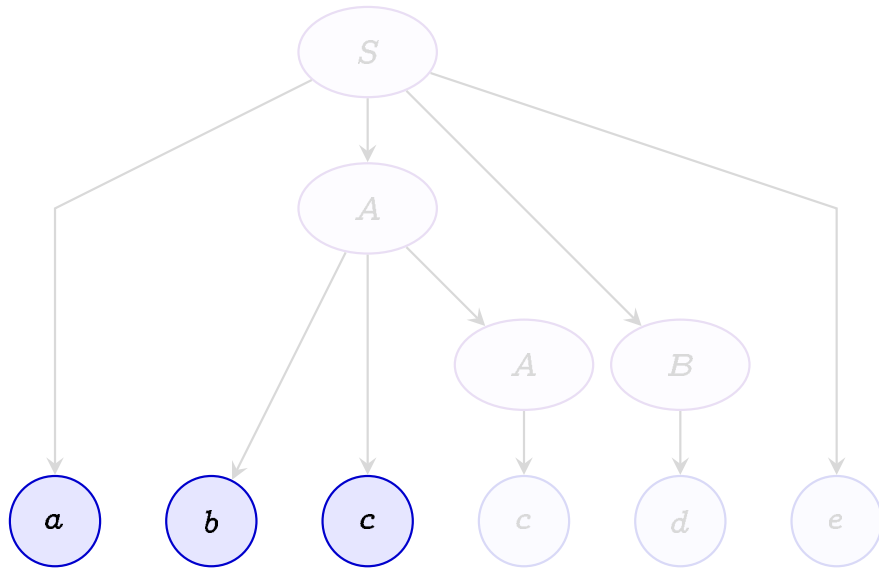
# Shift-reduce parsimine

Näide:

Input String: $a\,b\,c \bullet c\,d\,e$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
**shift**
shift
reduce $A \rightarrow c$
reduce $A \rightarrow b\,c\,A$
shift
reduce $B \rightarrow d$
shift
reduce $S \rightarrow a\,A\,B\,e$




$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

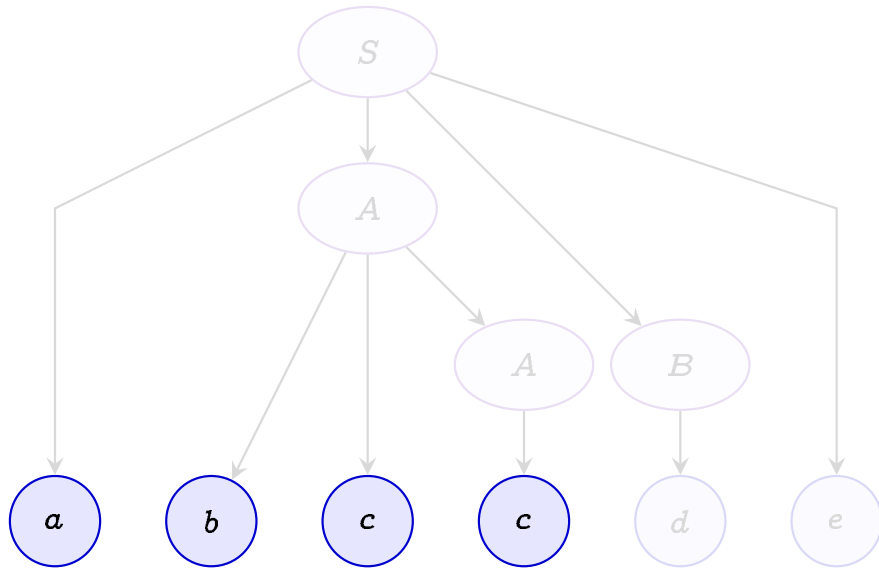
# Shift-reduce parsimine

Näide:

Input String: $a\,b\,c\,c \bullet d\,e$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
**shift**
**shift**
reduce $A \rightarrow c$
reduce $A \rightarrow b\,c\,A$
shift
reduce $B \rightarrow d$
shift
reduce $S \rightarrow a\,A\,B\,e$




$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

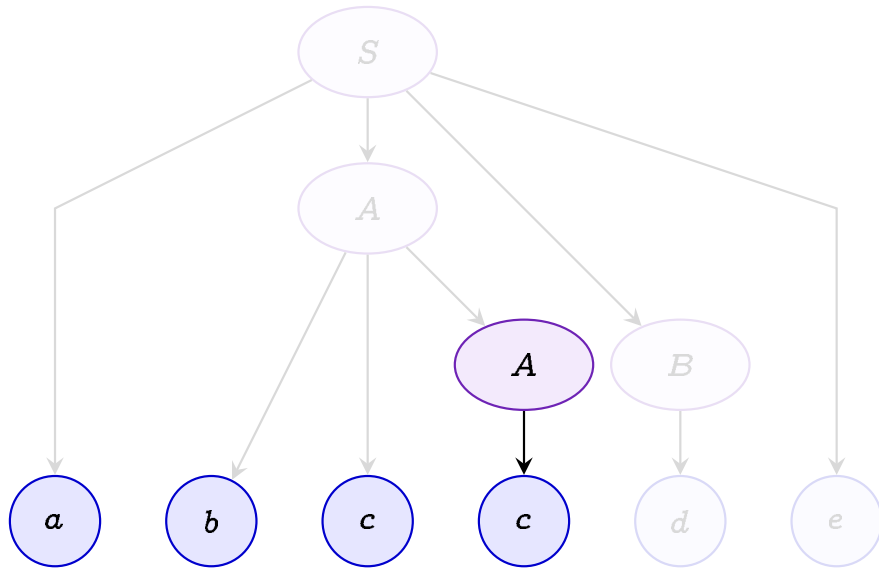
# Shift-reduce parsimine

Näide:

Input String: $a\,b\,c\,c \bullet d\,e$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
**shift**
**shift**
**reduce** $A \rightarrow c$
reduce $A \rightarrow b\,c\,A$
shift
reduce $B \rightarrow d$
shift
reduce $S \rightarrow a\,A\,B\,e$




$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

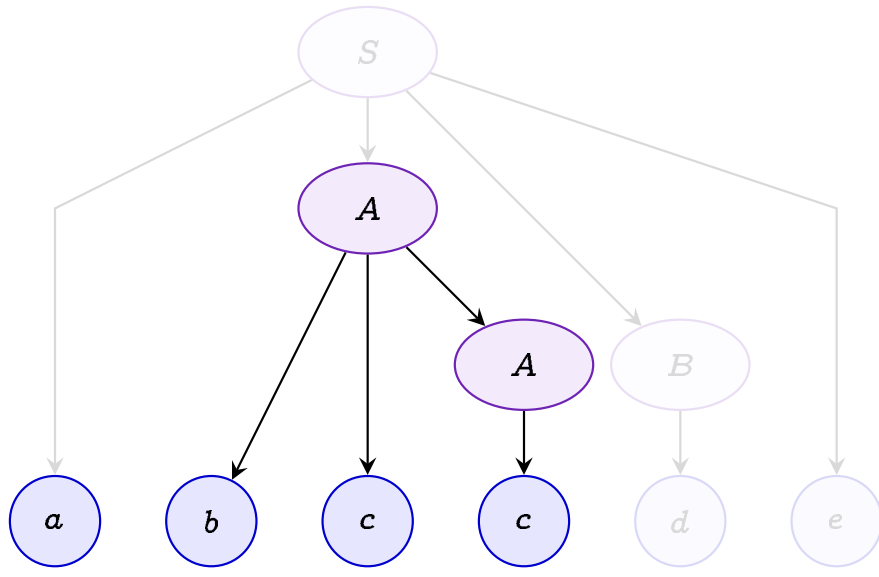
# Shift-reduce parsimine

Näide:

Input String: $a\,b\,c\,c \bullet d\,e$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
**shift**
**shift**
**reduce** $A \rightarrow c$
**reduce** $A \rightarrow b\,c\,A$
**shift**
**reduce** $B \rightarrow d$
**shift**
**reduce** $S \rightarrow a\,A\,B\,e$



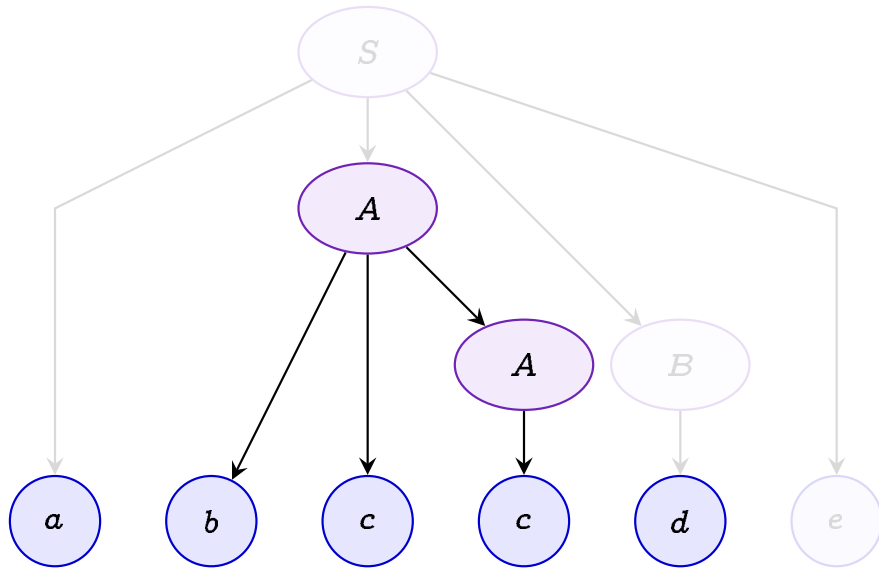

$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

# Shift-reduce parsimine

Näide:

Input String: $a\,b\,c\,c\,d \bullet e$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$




**shift**
**shift**
**shift**
**shift**
**reduce** $A \rightarrow c$
**reduce** $A \rightarrow b\,c\,A$
**shift**
**reduce** $B \rightarrow d$
**shift**
**reduce** $S \rightarrow a\,A\,B\,e$

$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

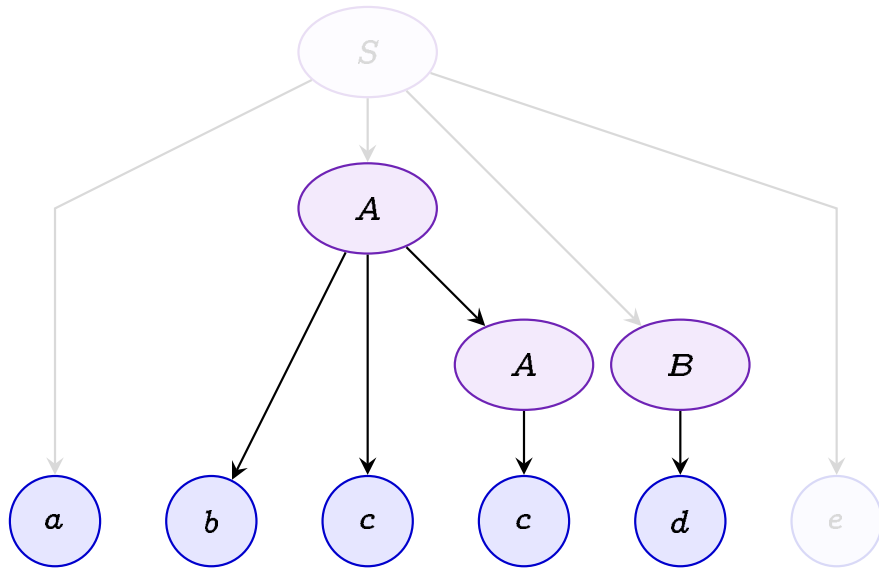
# Shift-reduce parsimine

Näide:

Input String: $a\,b\,c\,c\,d \bullet e$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
**shift**
**shift**
**reduce** $A \rightarrow c$
**reduce** $A \rightarrow b\,c\,A$
**shift**
**reduce** $B \rightarrow d$
**shift**
**reduce** $S \rightarrow a\,A\,B\,e$




$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

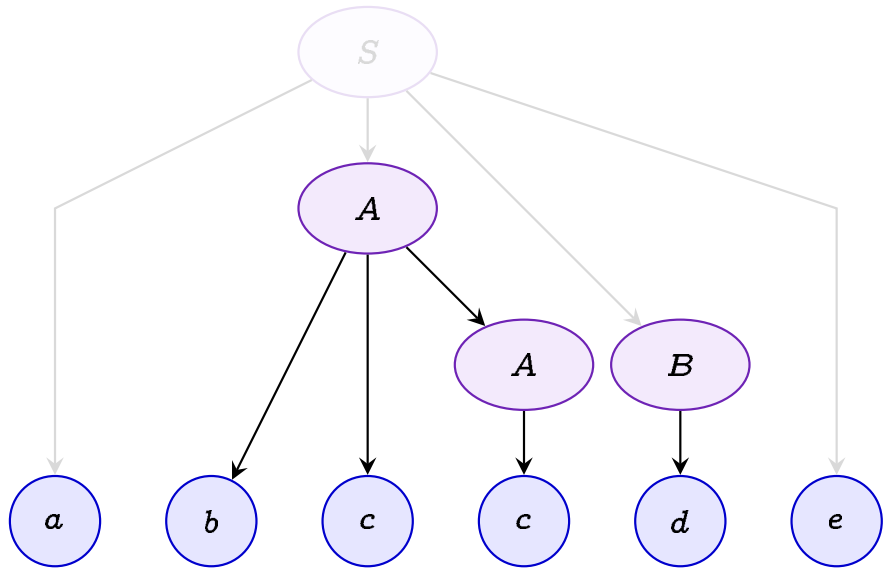
# Shift-reduce parsimine

Näide:

Input String: $a\,b\,c\,c\,d\,e\,\bullet$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
**shift**
**shift**
**reduce** $A \rightarrow c$
**reduce** $A \rightarrow b\,c\,A$
**shift**
**reduce** $B \rightarrow d$
**shift**
**reduce** $S \rightarrow a\,A\,B\,e$




$$
\begin{array}{lll}
S & \Longrightarrow_{rm} a\,A\,B\,e & \Longrightarrow_{rm} a\,A\,d\,e \\
 & \Longrightarrow_{rm} a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} a\,b\,c\,c\,d\,e
\end{array}
$$

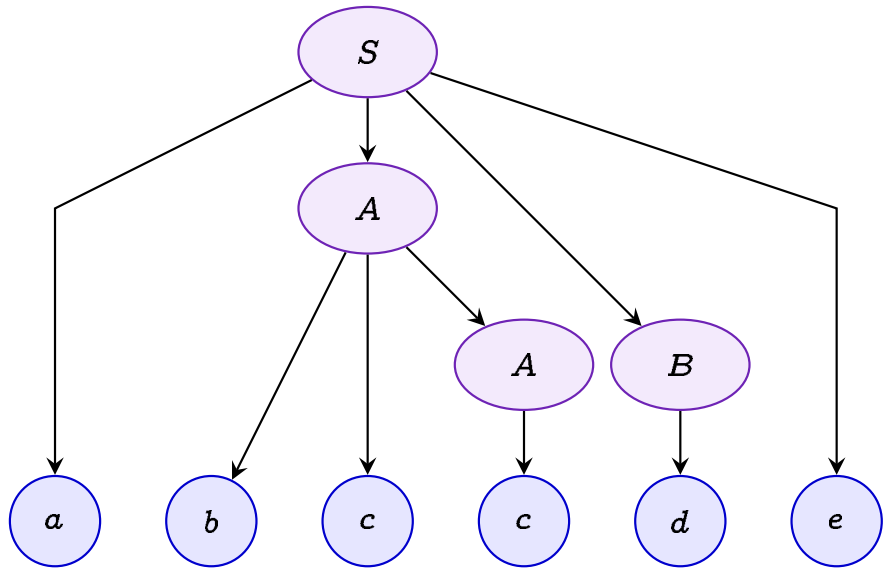
# Shift-reduce parsimine

Näide:

Input String: $a\,b\,c\,c\,d\,e\,\bullet$

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
**shift**
**shift**
**reduce** $A \rightarrow c$
**reduce** $A \rightarrow b\,c\,A$
**shift**
**reduce** $B \rightarrow d$
**shift**
**reduce** $S \rightarrow a\,A\,B\,e$




$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

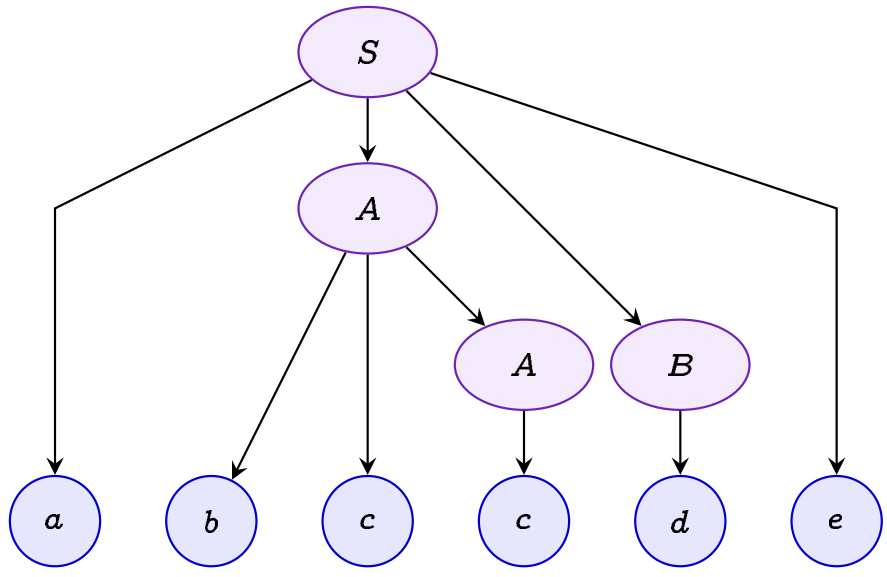
# Shift-reduce parsimine

Näide:

$$
\begin{array}{lcl}
S & \rightarrow & a\,A\,B\,e \\
A & \rightarrow & b\,c\,A \mid c \\
B & \rightarrow & d
\end{array}
$$

**shift**
**shift**
**shift**
**shift**
**reduce** $A \rightarrow c$
**reduce** $A \rightarrow b\,c\,A$
**shift**
**reduce** $B \rightarrow d$
**shift**
**reduce** $S \rightarrow a\,A\,B\,e$

Input String:




$$
\begin{array}{lllll}
S & \Longrightarrow_{rm} & a\,A\,B\,e & \Longrightarrow_{rm} & a\,A\,d\,e \\
 & \Longrightarrow_{rm} & a\,b\,c\,A\,d\,e & \Longrightarrow_{rm} & a\,b\,c\,c\,d\,e
\end{array}
$$

# Shift-reduce parsimine

- Lausevormi nimetatakse paremlausevormiks (right-sentential form), kui ta on tuletatav paremderivatsiooni abil.
- Paremlausevormi $\gamma$ pide (handle) on produktsioonireegel $A \to \beta$ ja sõnas $\gamma$ alamssõna $\beta$ esinemise positsioon, mis paremderivatsiooni eelmises lausevormis on asendatud mittetermi-naliga $A$.
- Ekvivalentselt, paremlausevormi $\gamma$ pide on alamstring $\beta$, selline et $S \Longrightarrow_{rm}^{\star} \delta A w \Longrightarrow_{rm} \delta \beta w = \gamma$, kus $\beta, \gamma, \delta \in V^{\star}$ ja $w \in T^{\star}$.
- Pidemete leidmise ja nende taandamise protsessi nimetatakse "pidemete kärpimiseks" ("handle pruning").
- NB! Ühese grammatika korral on paremderivatsioon, ja seega ka pidemed, unikaalsed.

# Shift-reduce parsimine

Näide: olgu antud grammatika

$$\begin{array}{lcl} S & \to & a\,A\,B\,e \\ A & \to & b\,c\,A \mid c \\ B & \to & d \end{array}$$

ja paremderivatsioon

$$S \Longrightarrow_{rm} a\,A\,B\,e \Longrightarrow_{rm} a\,A\,d\,e \Longrightarrow_{rm} a\,b\,c\,A\,d\,e$$

Paremlausevormi $abcAde$ pide on $bcA$.