Finite automata theory and formal languages (DIT321, TMV027)

Nils Anders Danielsson

2019-02-25

Today

- Some equivalences.
- Ambiguity.
- Designing grammars.

A bug

- Last time I told you that the following equivalence is valid:

$$\alpha \Rightarrow^* \beta \quad \Leftrightarrow \quad \alpha \Rightarrow^*_{\mathsf{lm}} \beta$$

- However, it is not.
- My mistake was to trust the course text book, which contains a similar error.
- Do not trust everything that you read.
- The following proposition is valid, trust me:

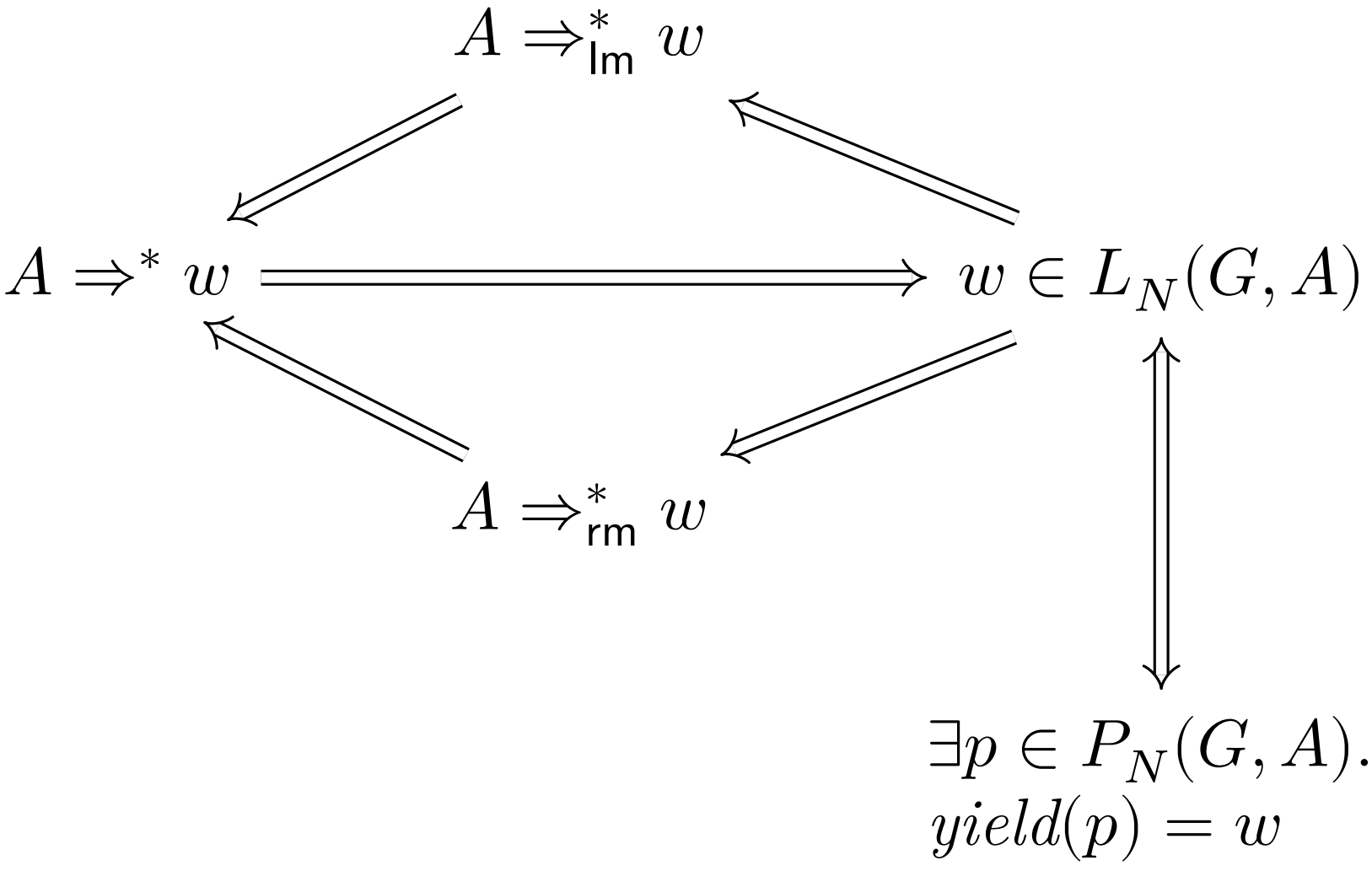
$$\forall \alpha \in (N \cup \Sigma)^*, w \in \Sigma^*.$$
$$\alpha \Rightarrow^* w \quad \Leftrightarrow \quad \alpha \Rightarrow^*_{\mathsf{lm}} w$$

Construct a grammar $G = (N, \Sigma, P, S)$
and strings $\alpha, \beta \in (N \cup \Sigma)^*$
such that $\alpha \Rightarrow^* \beta$ but not $\alpha \Rightarrow^*_{\mathsf{lm}} \beta$.

Some equivalences

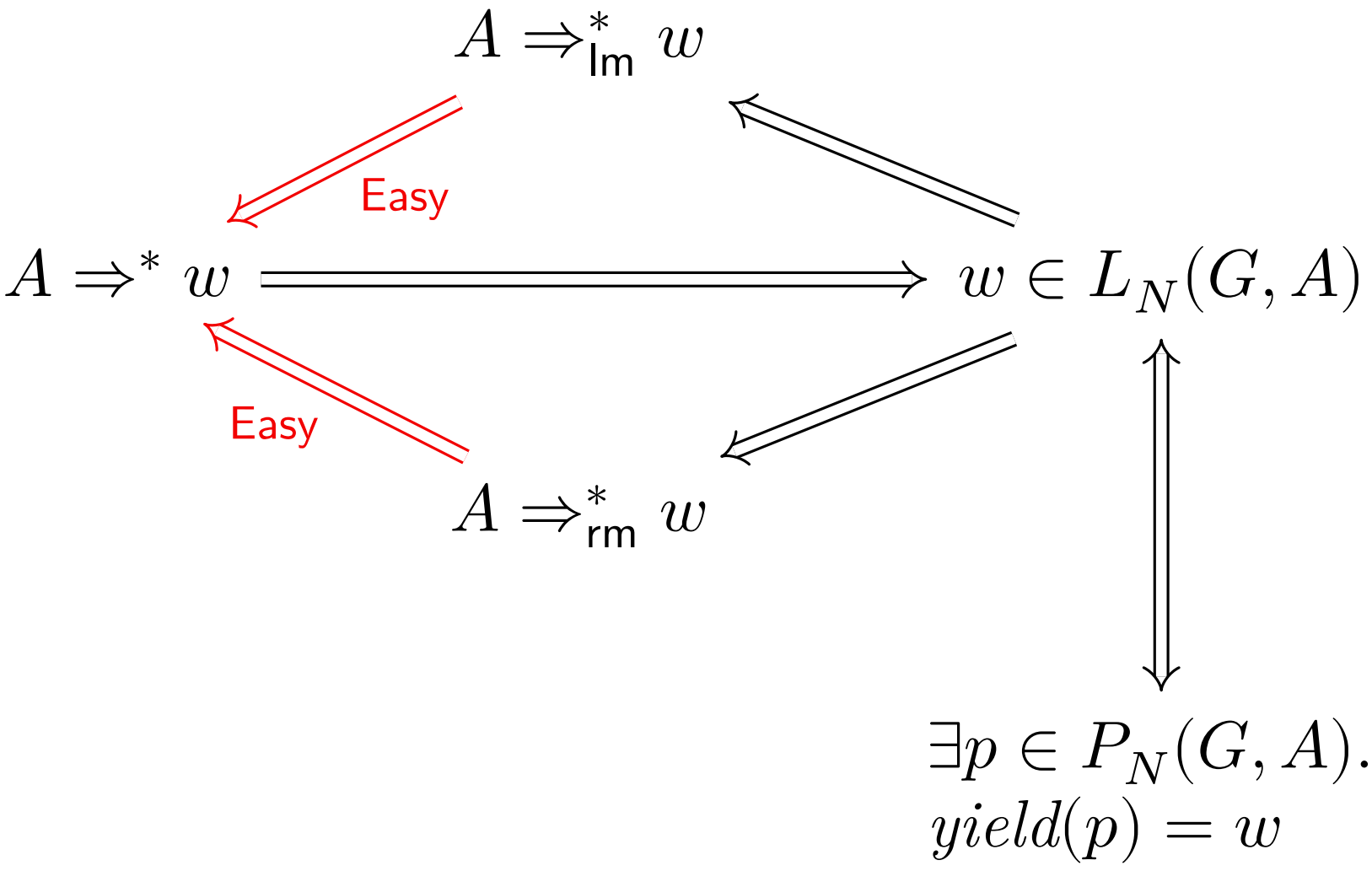
Some equivalences

With $w \in \Sigma^*$:



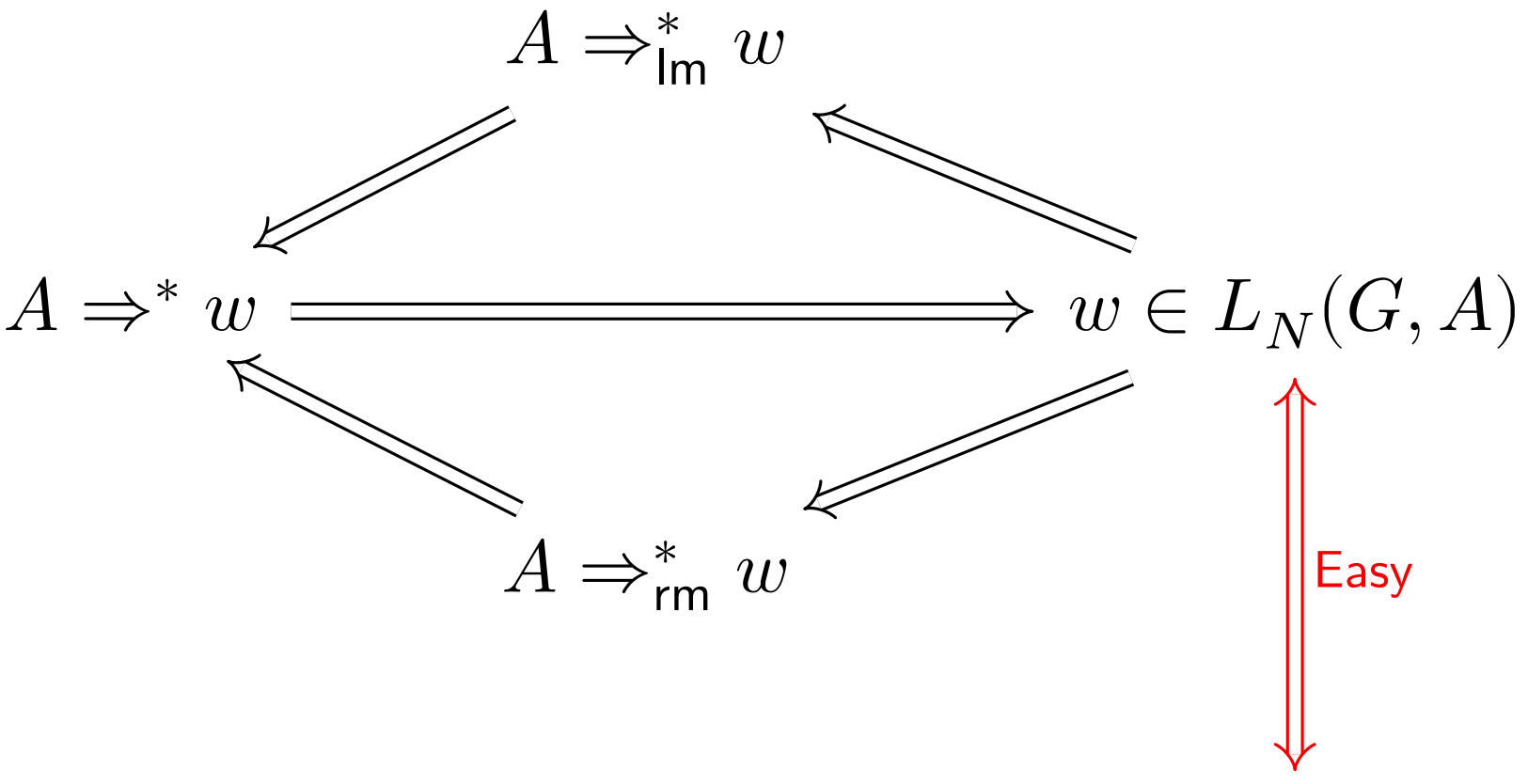
Some equivalences

With $w \in \Sigma^*$:



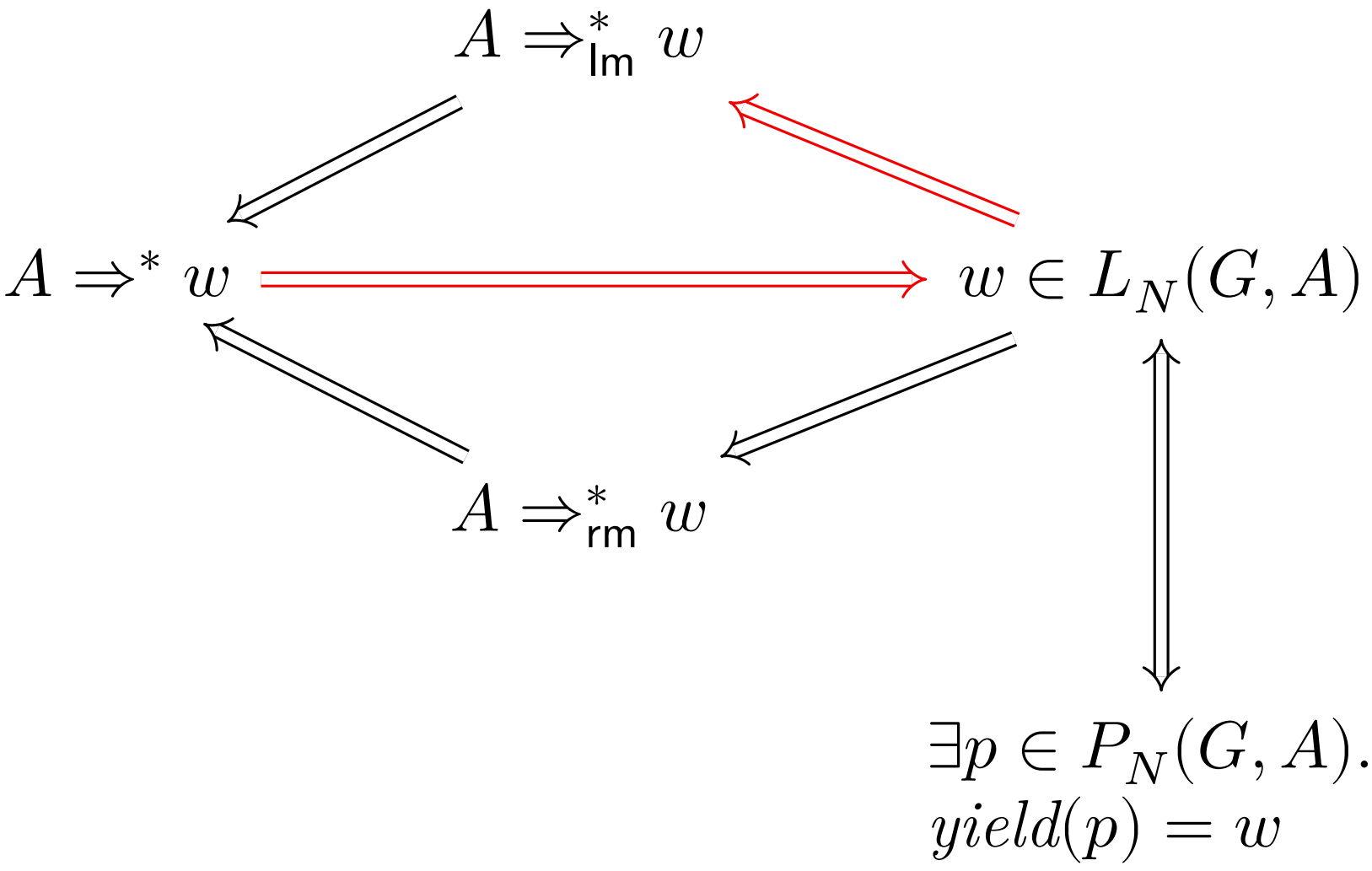
Some equivalences

With $w \in \Sigma^*$:



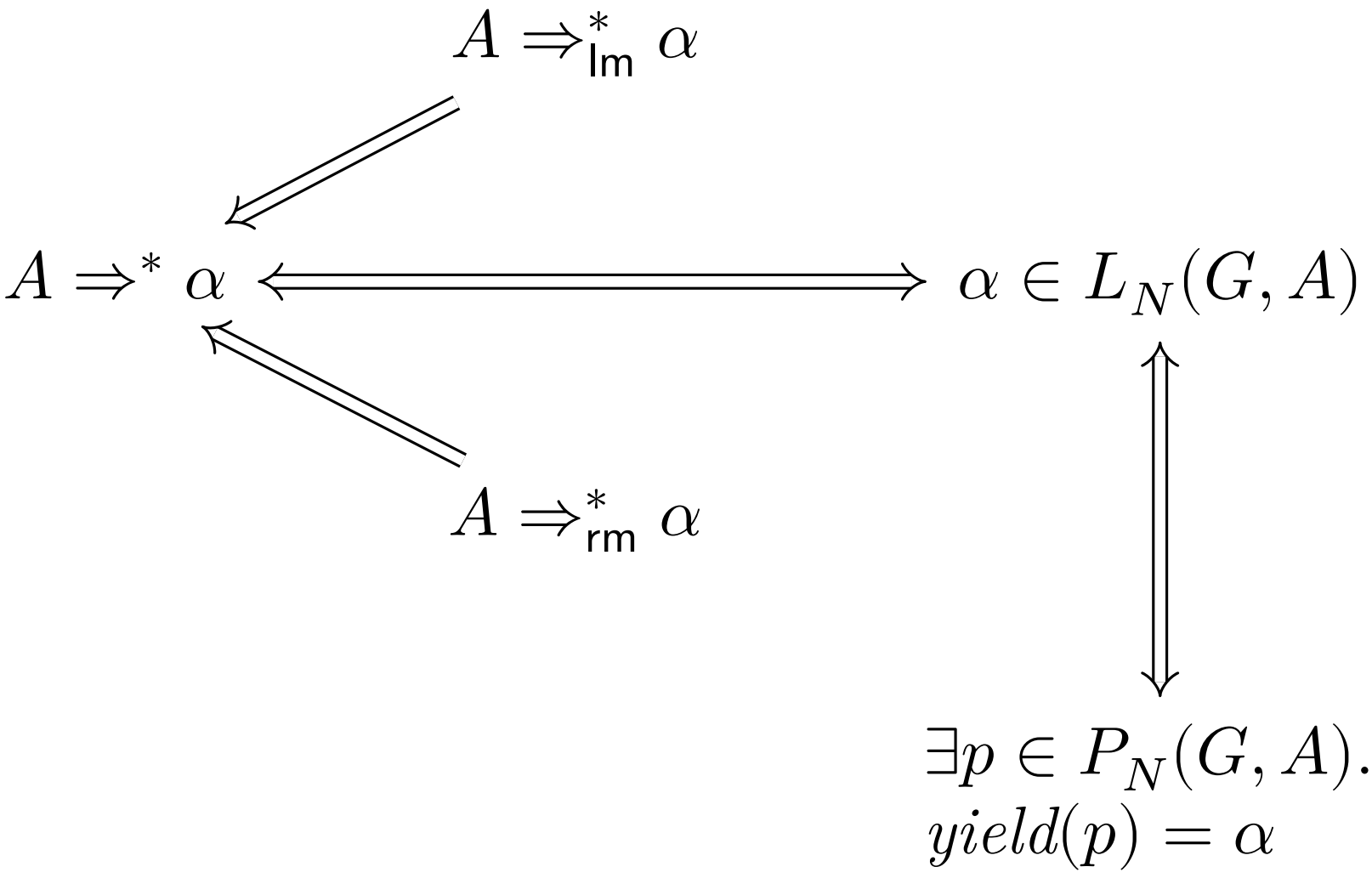
Some equivalences

With $w \in \Sigma^*$:



Some equivalences

With $\alpha \in (N \cup \Sigma)^*$:



Some equivalences

The property

$$\forall \alpha, \beta \in (N \cup \Sigma)^*.\ \alpha \Rightarrow^* \beta \ \Rightarrow\ \beta \in L_N^*(G, \alpha)$$

can be proved by induction on the structure of the derivation, using the following lemmas:

- $\alpha \in L_N^*(G, \alpha)$
- $\beta \in L_N^*(G, \alpha) \wedge \gamma \in L_N^*(G, \beta) \Rightarrow$ $\gamma \in L_N^*(G, \alpha)$
- $\alpha \Rightarrow \beta \ \Rightarrow\ \beta \in L_N^*(G, \alpha)$

Some equivalences

The property

$$\forall \alpha, \beta \in (N \cup \Sigma)^*.\ \alpha \Rightarrow \beta \ \Rightarrow\ \beta \in L_N^*(G, \alpha)$$

can be proved using the following additional lemmas:

- $\alpha \in L_N(G, A) \Rightarrow \alpha \in L_N^*(G, A)$
- $\beta \in L_N^*(G, \alpha) \wedge \beta' \in L_N^*(G, \alpha') \Rightarrow \beta\beta' \in L_N^*(G, \alpha\alpha')$

Prove

$$\forall \alpha, \beta \in (N \cup \Sigma)^*.\ \alpha \Rightarrow \beta \ \Rightarrow\ \beta \in L_N^*(G, \alpha).$$

Some equivalences

The property

$$\forall A \in N, w \in \Sigma^*.\ w \in L_N(G, A) \ \Rightarrow\ A \Rightarrow^*_{\mathsf{lm}} w$$

can be proved by induction on the structure of the recursive inference, using the following lemmas:

- $\alpha \Rightarrow^*_{\mathsf{lm}} \alpha' \ \Rightarrow\ \alpha\beta \Rightarrow^*_{\mathsf{lm}} \alpha'\beta$
- $\alpha \Rightarrow^*_{\mathsf{lm}} \alpha' \ \Rightarrow\ w\alpha \Rightarrow^*_{\mathsf{lm}} w\alpha'$
- $\alpha \Rightarrow^*_{\mathsf{lm}} \beta \wedge \beta \Rightarrow^*_{\mathsf{lm}} \gamma \ \Rightarrow\ \alpha \Rightarrow^*_{\mathsf{lm}} \gamma$

Ambiguity

Ambiguity

A grammar $G = (N, \Sigma, P, S)$ is ambiguous if there is a string $w \in \Sigma^*$ such that there are two different...

- ...parse trees in $P(G, S)$ with yield w.
- ...leftmost derivations $S \Rightarrow_{\mathsf{lm}}^* w$.
- ...rightmost derivations $S \Rightarrow_{\mathsf{rm}}^* w$.
- ...derivations of $w \in L(G, S)$.

Ambiguity

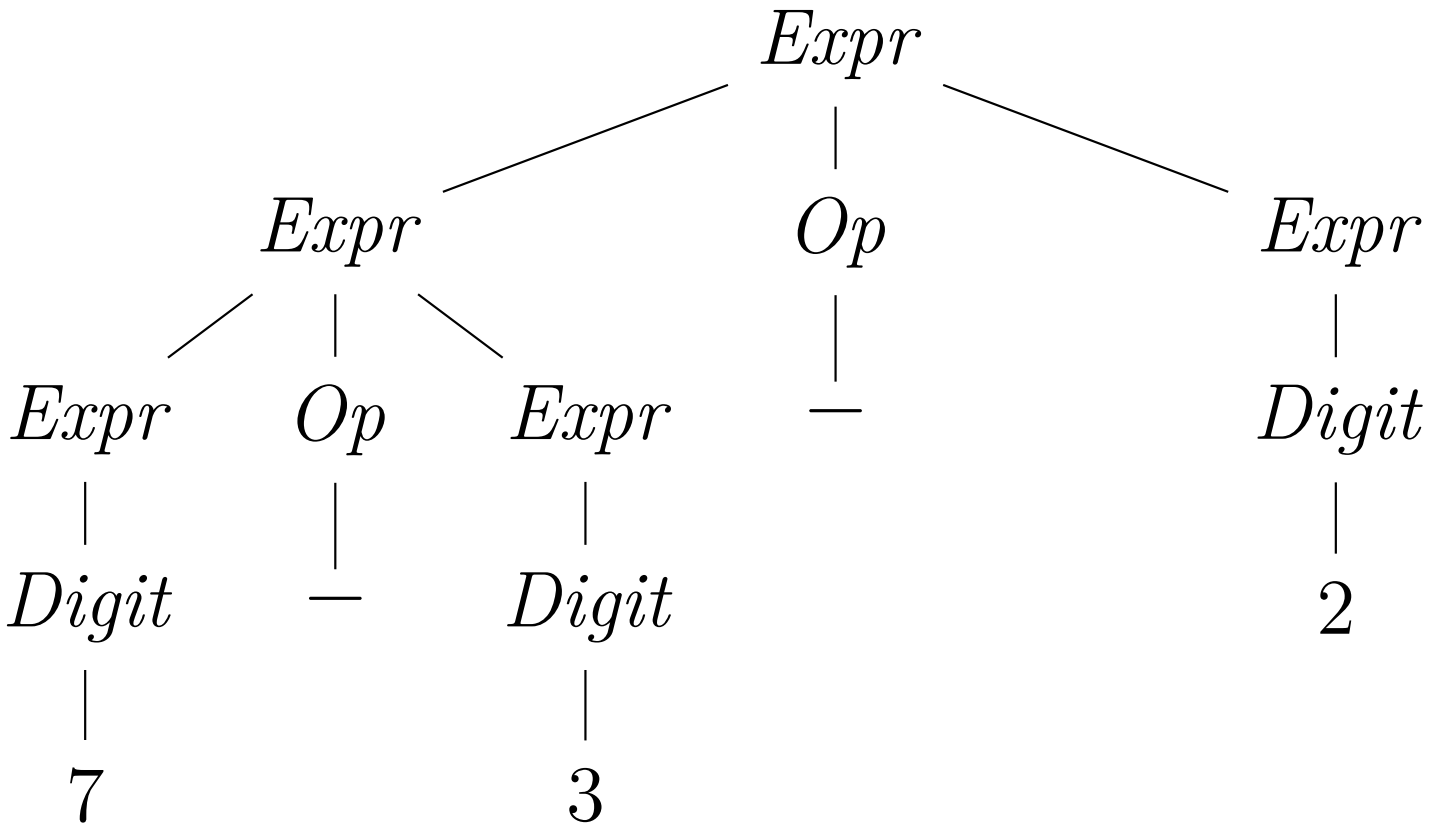
Consider the following (underspecified) context-free grammar over $\{ +, -, \cdot, /, (,) \} \cup \{ 0, 1, ..., 9 \}$:

$$
\begin{aligned}
\mathit{Expr} &\rightarrow \mathit{Expr}\ \mathit{Op}\ \mathit{Expr} \mid \mathit{Digit} \mid (\ \mathit{Expr}\ ) \\
\mathit{Op} &\rightarrow + \mid - \mid \cdot \mid / \\
\mathit{Digit} &\rightarrow 0 \mid 1 \mid ... \mid 9
\end{aligned}
$$

How should $7 - 3 - 2$ be interpreted?

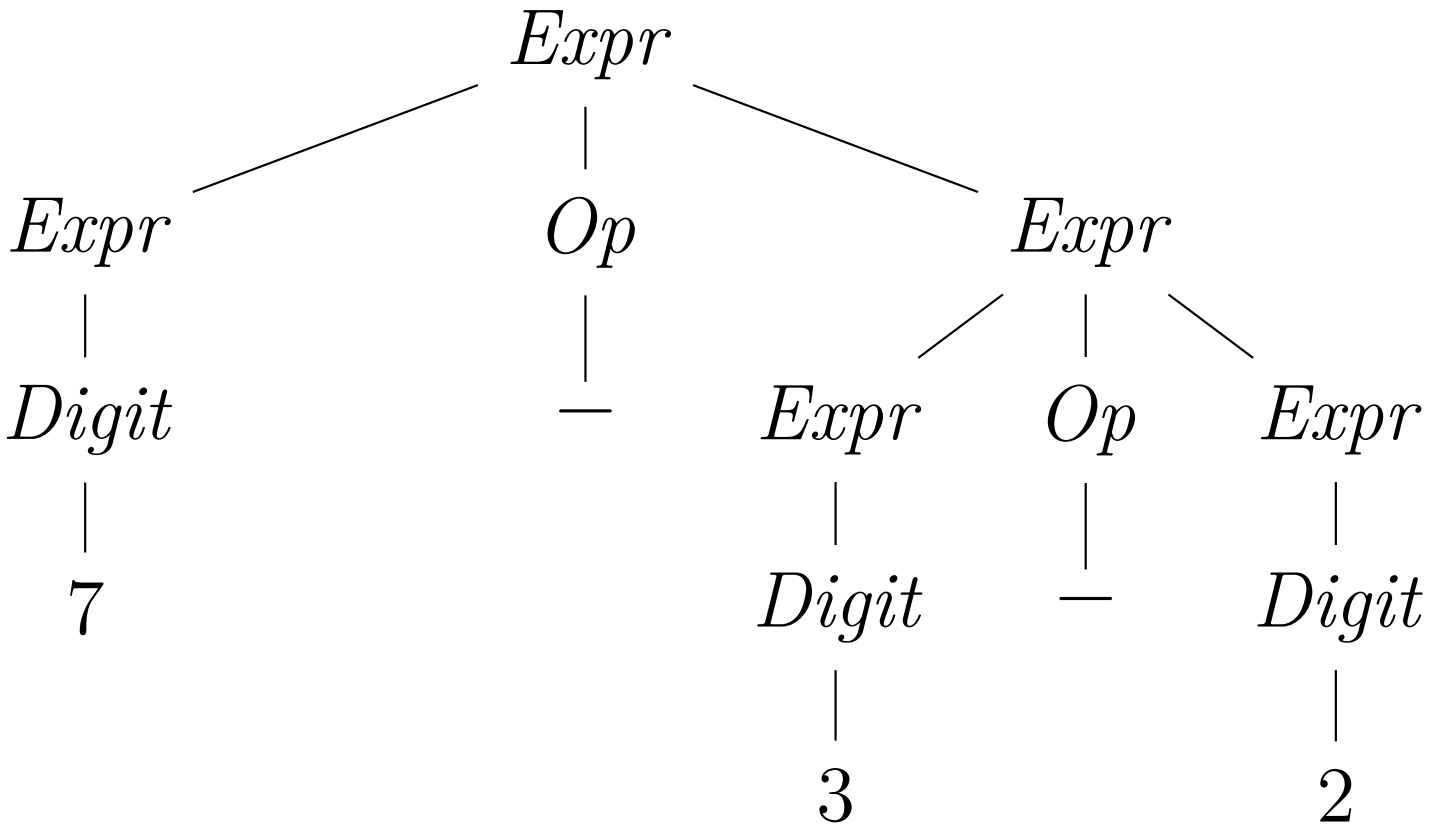
Ambiguity

A parse tree for $7 - 3 - 2$:



Ambiguity

Another parse tree for $7 - 3 - 2$:



Ambiguity

- The values differ: $(7 - 3) - 2 = 2$, but $7 - (3 - 2) = 6$.
- If a grammar is used to determine how to interpret an expression, then it may be unclear how to interpret an ambiguous string.

For which of the following sets of productions P is $(\{ S, A \}, \{ 0, 1 \}, P, S)$ an ambiguous grammar?

1. $S \to S$
2. $S \to S \mid \varepsilon$
3. $S \to 1S1 \mid 0S0 \mid \varepsilon$
4. $S \to 1S1 \mid 1A1 \mid \varepsilon, A \to 1A1 \mid S$
5. $S \to 1S1 \mid 1A1 \mid \varepsilon, A \to 0S0$

Ambiguity

- It is common to interpret $7 - 3 - 2$ as $(7 - 3) - 2$.
- The minus operator is said to "associate to the left".
- Exponentiation typically associates to the right: $3^{3^3} = 3^{(3^3)}$.

Ambiguity

- It is also common to interpret $7 \cdot 3 - 2$ as $(7 \cdot 3) - 2$, and not $7 \cdot (3 - 2)$.
- The multiplication operator is said to "bind tighter than" the subtraction operator, or to have "higher precedence".

Ambiguity

The following (underspecified) context-free grammar over $\{ +, -, \cdot, /, (,) \} \cup \{ 0, 1, ..., 9 \}$ is unambiguous:

$$
\begin{aligned}
\mathit{Expr} &\rightarrow \mathit{Term}\ \mathit{Add\text{-}op}\ \mathit{Expr} \mid \mathit{Term} \\
\mathit{Term} &\rightarrow \mathit{Term}\ \mathit{Mul\text{-}op}\ \mathit{Factor} \mid \mathit{Factor} \\
\mathit{Factor} &\rightarrow \mathit{Digit} \mid (\ \mathit{Expr}\ ) \\
\mathit{Add\text{-}op} &\rightarrow + \mid - \\
\mathit{Mul\text{-}op} &\rightarrow \cdot \mid / \\
\mathit{Digit} &\rightarrow 0 \mid 1 \mid ... \mid 9
\end{aligned}
$$

Use this grammar to parse the following string. Compute the value of the expression, using the parse tree to guide the evaluation.

$$3 \cdot 5 - 3 \cdot 8/4/2 - 6 \cdot 2$$

Ambiguity

- It is undecidable whether a context-free grammar is ambiguous.
- However, several parser generators use restricted context-free grammars that are guaranteed to be unambiguous.
- If such a tool complains about a “conflict”, then the problem might be that the grammar is ambiguous.

Suggest some replacement for ??? that ensures that $3 \hat{} 3 \hat{} 3$ is a valid string that is interpreted as $3 \hat{} (3 \hat{} 3)$. The start symbol is E_0.

$$
\begin{array}{ll}
E_0 & \rightarrow E_0 \; \mathit{Add\text{-}op} \; E_1 \mid E_1 \\
E_1 & \rightarrow E_1 \; \mathit{Mul\text{-}op} \; E_2 \mid E_2 \\
E_2 & \rightarrow ??? \\
E_3 & \rightarrow \mathit{Digit} \mid (\; E_0 \;) \\
\mathit{Add\text{-}op} & \rightarrow + \mid - \\
\mathit{Mul\text{-}op} & \rightarrow \cdot \mid / \\
\mathit{Digit} & \rightarrow 0 \mid 1 \mid ... \mid 9
\end{array}
$$

Ambiguity

- There are context-free languages for which there are no unambiguous context-free grammars.
- Such languages are called *inherently ambiguous*.
- See the book for an example.

Designing grammars

Define a grammar for some simple (context-free) language, perhaps a tiny programming language. Try to make the grammar unambiguous.

Designing grammars

If you want to know more about the use of grammars in the specification and implementation of programming languages you might be interested in the course *Programming language technology*.

Today

- Some equivalences.
- Ambiguity.
- Designing grammars.

Next lecture

- Something about normal forms.
- The pumping lemma for context-free languages.

- Deadline for the next quiz: 2019-02-28, 10:00.
- Deadline for the fifth assignment: 2019-03-03, 23:59.