Computer Science 513

Chomsky Hierarchy III: Context-Sensitive Languages

Instructor: Wayne Eberly

Department of Computer Science
University of Calgary

Lecture #23

Goals for Today

Goals for Today:

- Introduction to ***Context-Sensitive Grammars and Languages***

Reference:

- Michael A. Harrison
  Introduction to Formal Language Theory
  Addison-Wesley, 1978

Context-Sensitive Grammars

Consider a phrase-structure grammar

$$G = (V, \Sigma, \Pi, S)$$

as this is defined in Lecture #8.

Definition: *G* is a ***context-sensitive grammar*** if each of the productions has one of the following forms:

- $\alpha A \beta \rightarrow \alpha \gamma \beta$, where $A \in V$, $\alpha, \beta, \gamma \in (V \cup \Sigma)^\star$, $\gamma \neq \lambda$, and the start variable, S, does not appear in any of the strings, α, β, or γ;
- $S \rightarrow \lambda$.

A language $L \subseteq \Sigma^\star$ is a ***context-sensitive language*** if it is the language of a context-sensitive grammar.

Context-Sensitive Grammars

Example: Consider a phrase-structure grammar $G = (V, \Sigma, \Pi, S)$ such that

$$\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}, \quad V = \{S, B, C, T, W, Z\},$$

and Π includes the following productions.

$S \to T$	$CB \to CZ$	$\mathtt{a}B \to \mathtt{ab}$
$S \to \lambda$	$CZ \to WZ$	$\mathtt{b}B \to \mathtt{bb}$
$T \to \mathtt{a}BC$	$WZ \to WC$	$\mathtt{b}C \to \mathtt{bc}$
$T \to \mathtt{a}TBC$	$WC \to BC$	$\mathtt{c}C \to \mathtt{cc}$

Context-Sensitive Grammars

G is a context-sensitive grammar — and it will be shown, during the lecture presentation, that

$$L(G) = \{a^n b^n c^n \mid n \in \mathbb{N}\}.$$

Thus this language is a context-sensitive language.

Context-Sensitive Grammars

Claim #1: Every context-free language is a context-sensitive language.

Proof: Let $L \subseteq \Sigma^\star$ be a context-free language.

- Then $L = L(G)$, for some context-free grammar that is in ***Chomsky Normal Form***.
- Comparing definitions of "context-free grammars in Chomsky normal form" and "context-sensitive grammars", one can see that the above grammar, G, is a context-sensitive grammar.
- Thus L is context-sensitive. Since L was an arbitrarily chosen context-free language, this establishes the claim. □

Context-Sensitive Grammars

Claim: #2: There exists a language $L \subseteq \Sigma^\star$ (for some alphabet Σ) such that L is context-sensitive, but L is not context-free.

Proof: Let $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ and let

$$L = \{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \mid n \in \mathbb{N}\} \subseteq \Sigma^\star.$$

Then L is context-sensitive — as argued above — but L is not context-free, as shown in the previous lecture — as required to establish the claim. □

Noncontracting Grammars

Definition: A phrase-structure grammar $G = (V, \Sigma, \Pi, S)$ is a ***noncontracting grammar*** if every production in Π has the form

$$\alpha \to \beta$$

for $\alpha, \beta \in (V \cup \Sigma)^\star$ such that $|\beta| \geq |\alpha| \geq 1$. A phrase-structure grammar $G = (V, \Sigma, \Pi, S)$ is an ***essentially noncontracting grammar*** if the only productions in Π have one of the following forms

(i) $\alpha \to \beta$ for $\alpha, \beta \in (V \cup \Sigma)^\star$, where $|\beta| \geq |\alpha| \geq 1$, and the variable S does not appear in β, or

(ii) $S \to \lambda$.

Noncontracting Grammars

Claim #3: Let $L \subseteq \Sigma^\star$.

(a) L is a context-sensitive grammar if and only if L is the language of an essentially noncontracting grammar.

(b) If $\lambda \notin L$ then L is a context-sensitive grammar if and only if L is the language of a noncontracting grammar.

How This is Proved:

- Note that any context-sensitive grammar $G = (V, \Sigma, \Pi, S)$ is an essentially noncontracting grammar — and if $\lambda \notin L(G)$ then G is a noncontracting grammar as well — so one direction, in each part of the claim, is trivial.

Noncontracting Grammars

- In order to prove the rest of the claim, consider a production

$$\alpha \to \beta$$

in an essentially noncontracting grammar $G = (V, \Sigma, \Pi, S)$, that is not the production "$S \to \lambda$" — so that $\alpha, \beta \in (V \cup \Sigma)^\star$ and $|\beta| \geq |\alpha| \geq 1$.

- Suppose, in particular, that $|\alpha| = k$, $|\beta| = \ell$, and

$$\alpha = A_1 A_2 \ldots A_k \quad \text{and} \quad \beta = B_1 B_2 \ldots B_\ell$$

for symbols $A_1, A_2, \ldots, A_k, B_1, B_2, \ldots, B_\ell \in V \cup \Sigma$.

Noncontracting Grammars

- Let $C_1, C_2, \ldots, C_k$ be symbols that are *not* in $V \cup \Sigma$ — and suppose we include these symbols in a new set of variables $\widehat{V} = V \cup \{C_1, C_2, \ldots, C_k\}$ in a new set of variables, $\widehat{V}$, that replaces V.
- Let us ***replace*** the rule with "$\alpha \to \beta$" with the following:
  - Let us add the rule $A_1 A_2 \ldots A_k \to C_1 A_2 A_3 \ldots A_k$.
  - For each integer i such that $1 \leq i \leq k-2$ let us also introduce the rule

    $$C_1 C_2 \ldots C_i A_{i+1} A_{i+2} \ldots A_k \to C_1 C_2 \ldots C_i C_{i+1} A_{i+2} \ldots A_k.$$

  - Add the rule

    $$C_1 C_2 \ldots C_{k-1} A_k \to C_1 C_2 \ldots C_{k-1} B_k B_{k+1} \ldots B_\ell.$$

  - For each integer i such that $2 \leq i \leq k-1$ add the rule

    $$C_1 C_2 \ldots C_{i-1} C_i B_{i+1} B_{i+2} \ldots B_\ell \to C_1 C_2 \ldots C_{i-1} B_i B_{i+1} \ldots B_\ell.$$

  - Add the rule $C_1 B_2 B_3 \ldots B_\ell \to B_1 B_2 \ldots B_\ell$.

Noncontracting Grammars

- It can be shown that the new grammar has the same language as the old one.
- If this process is applied to every production that cannot be concluded in a context-sensitive grammar then the result is a context-sensitive grammar with the same language as the essentially noncontracting grammar that we started with — as needed to prove part (a) of the claim.
- The same process can be used to produce a context-sensitive grammar with the same language as any given noncontracting grammar — as needed to prove part (b) of the claim, as well.

Kuroda Normal Form

Definition: A noncontracting grammar $G = (V, \Sigma, \Pi, S)$ is in ***Kuroda normal form*** if every rule in Π has one of the following forms.

- $AB \to CD$, for $A, B, C, D \in V$
- $A \to CD$, for $A, C, D \in V$
- $A \to B$, for $A, B \in V$
- $A \to \mathrm{a}$, for $A \in V$ and $\mathrm{a} \in \Sigma$.

It can be shown that every language $L \subseteq \Sigma^\star$, such that L is context-sensitive, and such that $\lambda \notin L$, is the language of a grammar in Kuroda normal form.

Linear Bounded Automata

Definition: A ***linear bounded automaton*** is space-bounded nondeterministic Turing machine that is formally defined as a 7-tuple

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$$

where

- Q is a finite, nonempty set of ***states***.
- Σ is the ***input alphabet***.
- Γ is the ***tape alphabet***. $\Sigma \subseteq \Gamma$, and Γ also includes two special symbols, $\#_L$ and $\#_R$, where $\#_L \notin \Sigma$ and $\#_R \notin \Sigma$. $Q \cap \Gamma = \emptyset$.
- q_0, q_{accept} and q_{reject} are distinct states in Q.
- M's transition function, δ, and its tape, are as described next.

Linear Bounded Automata

- As for one-tape nondeterministic Turing machines (whose tape heads are allowed to stay where they are), the ***transition function*** is a function

$$\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{\mathtt{L}, \mathtt{R}, \mathtt{S}\}).$$

- However, the symbols "$\#_L$" and "$\#_R$" cannot be replaced with other symbols, and new copies of these can be written on the tape (so that other symbols in Γ) would be overwritten.
- Furthermore, the tape head can never move to the left past a copy of "$\#_L$" and it can never move to the right past a copy of "$\#_R$".
- Thus — for a state $q \in Q$ and symbol $\sigma \in \Gamma$ — $\delta(q, \sigma)$ should not include any transitions requiring these changes to the tape contents or movements of the tape head.

Linear Bounded Automata

- If $\omega \in \Sigma^\star$ then the ***initial configuration*** of M, for ω, is one such that the tape stores the string

$$\#_L \, \omega \, \#_R$$

  with the tape head pointing to the copy of "$\#_L$" — that is, pointing to the leftmost symbol in the above string — and the machine in state q_0.
- It follows, by the above, that if $|\omega| = n$ then M's tape head can never move outside the region, including $n + 2$ cells, that include "$\#_L$", the input string, and "$\#_R$", during its execution.

Linear Bounded Automata

- The computation(s) of a linear bounded automaton, on a string ω, be be represented by a ***computation tree***, just as they can for a nondeterministic Turing machine.
- Just like for a nondeterministic Turing machine...
  - *M* ***accepts*** ω if *M*'s computation tree for ω includes at least one "accepting configuration" – that is, a configuration in which *M* is in its accepting state, q_{accept}.
  - *M* ***rejects*** *M* if *M*'s computation tree for ω is finite and does not include an accepting configuration.
  - *M* ***loops on*** ω if *M*'s computation tree for ω is infinite does not include an accepting configuration.

  Thus *M* either ***accepts***, ***rejects*** or ***loops on*** every string $\omega \in \Sigma^\star$ (but does not do more than one of these things, for any string $\omega \in \Sigma^\star$).

Linear Bounded Automata

If $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ is a liinear bounded automaton then the ***language of M***, ***L*(*M*)**, is the set of strings $\omega \in \Sigma^\star$ such that M accepts ω.

Clam #4: Let $L \subseteq \Sigma^\star$. Then L is context-sensitive if and only if L is a language of a linear bounded automaton.

A proof of Claim #4 is given in Chapter 9 of Harrison's text, *Introduction to Formal Language Theory*.

Languages That are *Not* Context-Sensitive

- Claim #4 can be used to prove that the set of context-sensitive languages is the same as the ***complexity class*** "NSPACE(n)".
- Results from ***computational complexity theory*** (studied in a course like CPSC 511) can be used to prove the existence of languages that are not context-sensitive, and to identify some of these. For example, any language that is "PSPACE-complete" cannot be context-sensitive, because it is provably not in NSPACE(n).
- It also follows, by such results, that every context-sensitive language is ***decidable***, but there exist decidable languages that are not context-sensitive.

Closure Properties

- Like the set of context-free languages, the set of context-sensitive languages is closed under the "regular operations" — union, concatenation, and Kleene star. It is also closed under reversal.
- *Unlike* the set of context-free languages, the set of context-sensitive languages is *also* closed under intersection and complementation!
- Most of these closure properties are easily proved using Claim #4, The exception — closure under intersection — follows from a nontrivial result in computational complexity, the "Immerman-Szelepsényi Theorem", which states that various nondeterministic space-bounded complexity classes are closed under complementation.

Decidable Problems and Computable Functions

- Given a context-sensitive grammar $G = (V, \Sigma, \Pi, S)$, and a string $\omega \in \Sigma^\star$ one can decide whether $\omega \in L(G)$ by carrying out an ***exhaustive search*** for derivations of ω — using the fact that G is an "essentially noncontracting" grammar to terminate searches when needed.
- Since the set of all context-sensitive languages is the same as the computational complexity class NSPACE(n) — and since it is reasonably easy to convert a context-sensitive grammar into a linear bounded automaton with the same language — it would be extremely surprising if an *efficient* (that is, deterministic, polynomial-time) algorithm, to decide membership in a given context-sensitive language, existed.

Undecidable Problems

- Given any context-free grammar $G = (V, \Sigma, \Pi, S)$, one can do the following:
  - Convert this into a context-free grammar $\widehat{G}$, in Chomsky normal form, with the same language.
  - Notice that $\widehat{G}$ is also a ***context-sensitive*** grammar with the same language as G.
- This can be used to argue that various undecidable problems, concerning the languages of context-free grammars, have corresponding generalizations — concerning the languages of *context-sensitive grammars* — that must also be undecidable.
- For example, given a pair of context-sensitive grammars G_1 and G_2 with the same input alphabet, there is no algorithm to decide whether $L(G_1) \cap L(G_2)$