

CSCI 256: Theory of Computing

CHAPTER 5 Context-Free Grammars and Languages

Dr. Benjoe A. Juliano

Tel. 530 898-4619 (office)
530 898-6442 (dept)
Fax. 530 898-5995

Juliano@ecst.csuchico.edu
http://www.ecst.csuchico.edu/~juliano

B. Juliano © 2002, based on notes by J. Ullman



1

From Figure 5.1 of ITLC, Hopcroft, Motwani, & Ullman, 2001.

Context-Free Grammars

An Informal Example

- $L_{pal} = \{w \in \{0,1\}^* \mid w = w^R\}$
- A **context-free grammar** for palindromes:

1. $P \rightarrow \epsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

B. Juliano © 2002, based on notes by J. Ullman



2

Context-Free Grammars

Definition

- A **context-free grammar** G is described by a four-tuple, $G = (V, T, P, S)$ where
 - V , a finite set of **variables** (or *nonterminals*, or *syntactic categories*), represents a language; i.e. a set of strings;
 - T , a finite set of **terminals** (or *symbols*, or *terminal symbols*), form the strings of the language being defined;
 - $P \subseteq V \times (VNT)^*$, a finite set of **productions** or *rules*, represent the recursive definition of a language; and
 - $S \in V$, the **start symbol**.

B. Juliano © 2002, based on notes by J. Ullman



3

Context-Free Grammars

Example

- $G = (\{S, A\}, \{0,1\}, P, S)$ is a CFG that generates strings of 0's and 1's such that each block of 0's is followed by at least as many 1's, where P is denoted by:

$$S \rightarrow AS \mid \epsilon$$

$$A \rightarrow 0A1 \mid A1 \mid 01$$

B. Juliano © 2002, based on notes by J. Ullman



4

From Figure 5.2 of A.T.L.C. Hopcroft, Motwani, & Ullman, 2001.

Context-Free Grammars

Example

- $G = (\{E, I\}, \{+, *, (,), a, b, 0, 1\}, P, E)$ is a CFG for simple expressions, where P is denoted by:

1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$

B. Juliano © 2002, based on notes by J. Ullman



Context-Free Grammars

Derivations using a Grammar

- Recursive inference** takes strings known to be in the language and then uses the rules from body to head.
- Derivations** use the productions from head to body; the language of the grammar is all strings of terminals obtained this way.
 - $\alpha A \beta \Rightarrow \alpha \gamma \beta$ whenever there is a production $A \rightarrow \gamma$.
 - $\alpha \xRightarrow{*} \beta$ means string α can become β in zero or more derivation steps.

B. Juliano © 2002, based on notes by J. Ullman



Context-Free Grammars

Sentential Forms

- If $G = (V, T, P, S)$ is a CFG, then any string α in $(VNT)^*$ such that $S \xRightarrow{*} \alpha$ is a **sentential form**.

Leftmost and Rightmost Derivations

- A **leftmost derivation** always replaces the *leftmost* variable in a sentential form.
 - Notation: \Rightarrow_{lm} or $\xRightarrow{*}_{lm}$
- A **rightmost derivation** defined analogously.
 - Notation: \Rightarrow_{rm} or $\xRightarrow{*}_{rm}$

B. Juliano © 2002, based on notes by J. Ullman



Context-Free Grammars

Language of a CFG

- If $G = (V, T, P, S)$ is a CFG, the **language** of G , denoted $L(G)$, is defined as

$$L(G) = \{ w \in T^* \mid S \xRightarrow{*} w \}$$

$L(G)$ is a **context-free language**, or CFL.

Theorem 5.7

- $L(G_{pal})$, where G_{pal} is the grammar in Slide #2, is the set of palindromes over $\{0, 1\}$.

B. Juliano © 2002, based on notes by J. Ullman



Context-Free Grammars

Other Sentential Forms

- Given CFG $G=(V,T,P,S)$ and $\alpha \in (VNT)^*$
 - If $S \xRightarrow{lm}^* \alpha$, then α is a *left-sentential form*.
 - If $S \xRightarrow{rm}^* \alpha$, then α is a *right-sentential form*.
 - The CFL $L(G)$ are those sentential forms that are in T^* ; *i.e.* they consist only of terminals.

B. Juliano © 2002, based on notes by J. Ullman

9

Parse Trees

Definition

- A *parse tree*, or *derivation tree*, is a tree representation for derivations from CFG $G=(V,T,P,S)$ where
 - Nodes are labeled by $x \in VNTN\{\epsilon\}$.
 - interior nodes* are labeled by $v \in V$
 - leaves* are labeled by $t \in TN\{\epsilon\}$; a leaf can be labeled ϵ if it is the only child of its parent
 - If an interior node is labeled A , and its children are labeled X_1, X_2, \dots, X_k respectively, from left to right, then $A \rightarrow X_1 X_2 \dots X_k$ where $X_i \in VNT$ is a production in P .

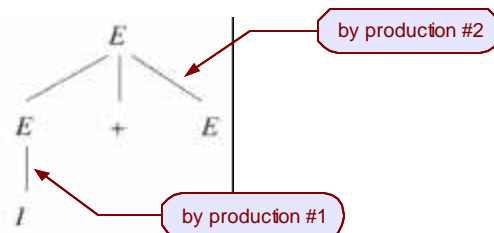
B. Juliano © 2002, based on notes by J. Ullman

10

Parse Trees

Example

- A parse tree that uses the *expression grammar* of Slide #5 showing the derivation $E \xRightarrow{*} I+E$.



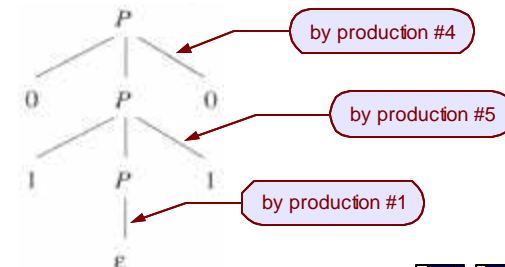
B. Juliano © 2002, based on notes by J. Ullman

11

Parse Trees

Example

- A parse tree for the *palindrome grammar* of Slide #2 showing the derivation $P \xRightarrow{*} 0110$.



B. Juliano © 2002, based on notes by J. Ullman

12

Parse Trees

Yield of a Parse Tree

- The **yield** of a parse tree is the string $w \in T^*$ derived by concatenating, from the left, the labels of the leaves of the parse tree.
- Let P denote the set of all parse trees whose yields are strings in the language of the underlying grammar. Then, each $p \in P$ has the following features:
 - The yield of p is a terminal string; *i.e.*, all leaves are labeled either with a terminal or ϵ .
 - The root of p is labeled by the start symbol.

B. Juliano © 2002, based on notes by J. Ullman

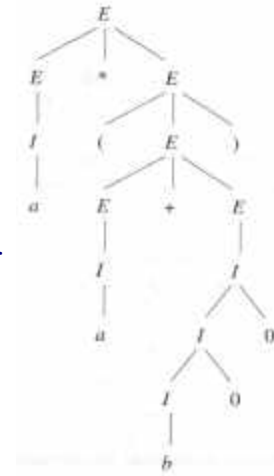
13

From Figure 5.6 of *ITLC*, Hopcroft, Motwani, & Ullman, 2001.

Parse Trees

Example

- Parse tree showing $a^*(a+b00)$ is in the language of the expression grammar of Slide #5.



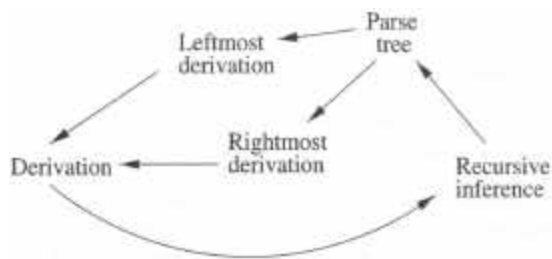
B. Juliano © 2002, based on notes by J. Ullman

14

Parse Trees

Inference, Derivations, and Parse Trees

- Plan for proving the equivalence of certain statements about grammars:



B. Juliano © 2002, based on notes by J. Ullman

15

Parse Trees

From Inferences to Trees

Theorem 5.12

- Let $G=(V,T,P,S)$ be a CFG. If the recursive inference procedure tells us that $w \in T^*$ is in the language of $A \in V$, then there is a parse tree with root A and yield w .
- Proof:**
 - By induction on the number of steps used to infer that $w \in T^*$ is in the language of $A \in V$.

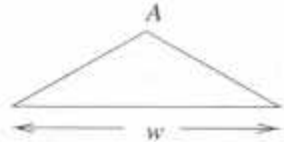
B. Juliano © 2002, based on notes by J. Ullman

16

Parse Trees

From Figure 5.8 of ATLC, Hopcroft, Motwani, & Ullman, 2001.

- From Inferences to Trees, *continued* ...
 - Proof
 - BASIS: Only one step is used to infer that $w \in T^*$ is in the language of $A \in V$.



- There must be a production $A \rightarrow w$ whose parse tree is given above.

B. Juliano © 2002, based on notes by J. Ullman

17

Parse Trees

- From Inferences to Trees, *continued* ...
 - Proof
 - INDUCTIVE HYPOTHESIS: Suppose that for all strings $x \in T^*$ and $B \in V$ where the membership of x in the language of B was inferred using n or fewer steps, there exists a parse tree with root B and yield x .
 - INDUCTIVE STEP: Consider the case where $n+1$ inference steps are used to infer that $w \in T^*$ is in the language of $A \in V$.

B. Juliano © 2002, based on notes by J. Ullman

18

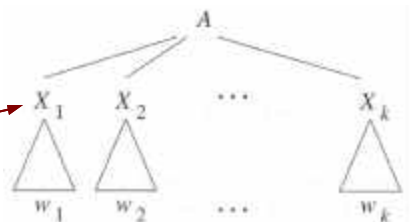
Parse Trees

From Figure 5.9 of ATLC, Hopcroft, Motwani, & Ullman, 2001.

- From Inferences to Trees, *continued* ...
 - Proof
 - INDUCTIVE STEP, *continued*: Let the last step of the inference that $w \in T^*$ is in the language of $A \in V$ use the production $A \rightarrow X_1 X_2 \dots X_k$ where $X_i \in VNT$.

Case 1: $X_i \in T$.
Then $w_i = X_i$, and the yield of the subtree is w_i .

Case 2: $X_i \in V$.
Then by IH there is a parse tree with root X_i and the yield of the subtree is w_i .



B. Juliano © 2002, based on notes by J. Ullman

19

Parse Trees

- From Trees to Derivations
 - Theorem 5.14**
 - Let $G = (V, T, P, S)$ be a CFG, and suppose there is a parse tree with root labeled $A \in V$ and with yield $w \in T^*$. Then there is a leftmost derivation $A \xRightarrow{*}_{lm} w$ in grammar G .
 - Proof
 - By induction on the height of the parse tree.

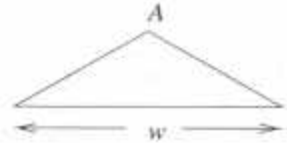
B. Juliano © 2002, based on notes by J. Ullman

20

From Figure 5.8 of (ATC, Hopcroft, Motwani, & Ullman, 2001).

Parse Trees

- From Trees to Derivations, *continued* ...
 - Proof
 - BASIS: The parse tree with root labeled $A \in V$ and yield $w \in T^*$ has height 1.



- There must be a production $A \rightarrow w$; thus, $A \xRightarrow{lm} w$ is a one-step, leftmost derivation of w from A .

B. Juliano © 2002, based on notes by J. Ullman

21

Parse Trees

- From Trees to Derivations, *continued* ...
 - Proof
 - INDUCTIVE HYPOTHESIS: Suppose that for all parse trees with root labeled $A \in V$, yield $w \in T^*$, and height n or fewer, there is a leftmost derivation $A \xRightarrow{lm}^* w$ in grammar G .
 - INDUCTIVE STEP: Consider the case where a parse tree with root labeled $A \in V$ and yield $w \in T^*$ has height $n+1$.

B. Juliano © 2002, based on notes by J. Ullman

22

From Figure 5.9 of (ATC, Hopcroft, Motwani, & Ullman, 2001).

Parse Trees

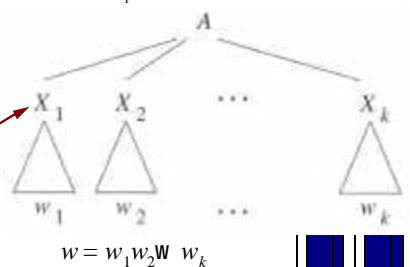
- From Trees to Derivations, *continued* ...
 - Proof
 - INDUCTIVE STEP, *continued*: The root of the parse tree is $A \in V$ with children X_1, X_2, \dots, X_k from the left, and where $X_i \in VNT$.

Case 1: $X_i \in T$.

Then $w_i = X_i$.

Case 2: $X_i \in V$.

Then by IH X_i is root of a parse tree with yield of w_i , and height at most n , so there is a leftmost derivation $X_i \xRightarrow{lm}^* w_i$.



$$w = w_1 w_2 \dots w_k$$

B. Juliano © 2002, based on notes by J. Ullman

23

Parse Trees

- From Trees to Derivations, *continued* ...
 - Proof
 - INDUCTIVE STEP, *continued*: Construct a leftmost derivation of w beginning with $A \xRightarrow{lm} X_1 X_2 \dots X_k$; then, for each $1 \leq i \leq k$, in order,

$$A \xRightarrow{lm}^* w_1 w_2 \dots w_i X_{i+1} X_{i+2} \dots X_k$$

By induction on i :

A Basis: $i=0$. We know $A \xRightarrow{lm} X_1 X_2 \dots X_k$.

A IndHyp: Assume $A \xRightarrow{lm}^* w_1 w_2 \dots w_{i-1} X_i X_{i+1} \dots X_k$

B. Juliano © 2002, based on notes by J. Ullman

24

Parse Trees

From Trees to Derivations, *continued* ...

Proof

INDUCTIVE STEP, *continued*:

" By induction on i :

A IndStep: Consider $A \xRightarrow{lm}^* w_1 w_2 W w_{i-1} X_i X_{i+1} W X_k$

• If $X_i \in T^*$, then $A \xRightarrow{lm}^* w_1 w_2 W w_{i-1} X_{i+1} X_{i+2} W X_k$

• If $X_i \in V$, continue with a derivation of w_i from X_i . If this derivation is

$$X_i \xRightarrow{lm} \alpha_1 \xRightarrow{lm} \alpha_2 \xRightarrow{lm} W \xRightarrow{lm} w_i$$

...

Parse Trees

From Trees to Derivations, *continued* ...

Proof

" By induction on i :

... proceed with

$$w_1 w_2 W w_{i-1} X_i X_{i+1} W X_k \xRightarrow{lm}$$

$$w_1 w_2 W w_{i-1} \alpha_1 X_{i+1} W X_k \xRightarrow{lm}$$

$$w_1 w_2 W w_{i-1} \alpha_2 X_{i+1} W X_k \xRightarrow{lm}$$

...

$$w_1 w_2 W w_i X_{i+1} X_{i+2} W X_k$$

There is a derivation $A \xRightarrow{lm}^* w_1 w_2 W w_i X_{i+1} X_{i+2} W X_k$

Parse Trees

From Trees to Derivations

Theorem 5.16

• Let $G=(V,T,P,S)$ be a CFG, and suppose there is a parse tree with root labeled $A \in V$ and with yield $w \in T^*$. Then there is a rightmost derivation $A \xRightarrow{rm}^* w$ in grammar G .

Proof

• Similar to proof of Theorem 5.14 ...

Parse Trees

From Derivations to Recursive Inferences

Theorem 5.18

• Let $G=(V,T,P,S)$ be a CFG, and suppose there is a derivation $A \xRightarrow{*} w$ where $w \in T^*$. Then the recursive inference procedure applied to G determines that w is in the language of $A \in V$.

Proof

• By induction on the length of the derivation

$$A \xRightarrow{*} w.$$

Parse Trees

- From Derivations to Recursive Inferences *cont'd*
 - Proof
 - BASIS: If the length of the derivation $A \xRightarrow{*} w$ is one step, then $A \rightarrow w$ must be a production. Since $w \in T^*$, w is in the language of A .
 - INDUCTIVE HYPOTHESIS: Suppose that for any derivation of length at most n , the recursive inference procedure applied to CFG $G=(V,T,P,S)$ determines that $w \in T^*$ is in the language of $A \in V$.

29

B. Juliano © 2002, based on notes by J. Ullman

Parse Trees

- From Derivations to Recursive Inferences *cont'd*
 - Proof
 - INDUCTIVE STEP: Consider the case where the derivation takes $n+1$ steps. Write the derivation as $A \Rightarrow X_1 X_2 w$ $X_k \xRightarrow{*} w \in T^*$ where $X_i \in VNT$. Break w as $w = w_1 w_2 w_k$, where
 - If $X_i \in T$, then $w_i = X_i$.
 - If $X_i \in V$, then $X_i \xRightarrow{*} w_i$. Since the first step of the derivation $A \xRightarrow{*} w$ is surely not part of the derivation $X_i \xRightarrow{*} w_i$, we know the derivation $X_i \xRightarrow{*} w_i$ is of n or fewer steps.

30

B. Juliano © 2002, based on notes by J. Ullman

Parse Trees

- From Derivations to Recursive Inferences *cont'd*
 - Proof
 - INDUCTIVE STEP, *continued*:
 - The IH applies to the derivation $X_i \xRightarrow{*} w_i$, and so w_i is inferred to be in the language of X_i .
 - So, we have a production $A \rightarrow X_1 X_2 w_k$ with w_i either equal to X_i or known to be in the language of X_i . In the next round of the recursive inference procedure, $w_1 w_2 w_k$ is in the language of A . Since $w_1 w_2 w_k = w$, we have shown that w is inferred to be in the language of A .

31

B. Juliano © 2002, based on notes by J. Ullman

CFG Applications

- Context-free grammars were originally conceived by **Noam Chomsky** (in 1956) as a way to describe natural languages.
 - Grammars are used to describe programming languages. Language descriptions in CFG form can be converted into a **parser**.
 - An essential part of **XML** (Extensible Markup Language) is the **Document Type Definition** (DTD), which is essentially a CFG. XML is widely predicted to facilitate e-commerce by allowing participants to share documents.

32

B. Juliano © 2002, based on notes by J. Ullman

Ambiguity

Ambiguous Grammars

- A CFG $G=(V,T,P,S)$ is *ambiguous* if there is at least one string $w \in T^*$ for which we can find two different parse trees, each with root labeled S and yield w . If each string has at most one parse tree in the grammar, then the grammar is *unambiguous*.

B. Juliano © 2002, based on notes by J. Ullman

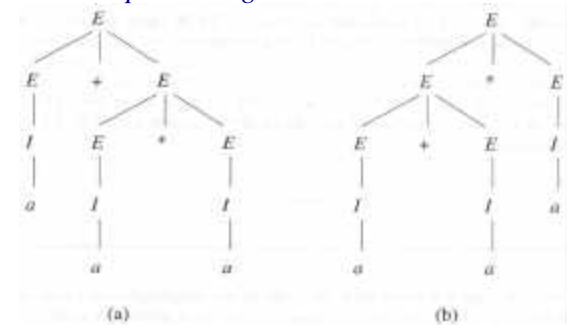
33

From Figure 5.16 of (A.T.L.C., Hopcroft, Motwani, & Ullman, 2001).

Ambiguity

Example

- The string $a+a*a$ has two derivations from E of the *expression grammar* of Slide #5.



B. Juliano © 2002, based on notes by J. Ullman

34

Ambiguity

Removing Ambiguity in Grammars

- There is no algorithm whatsoever that can tell us whether a CFG is ambiguous.
- There are CFLs that have nothing but ambiguous CFGs; for these languages, removal of ambiguity is impossible.
- Alas, for the sorts of constructs that appear in programming languages, there are well-known techniques for eliminating ambiguity.

B. Juliano © 2002, based on notes by J. Ullman

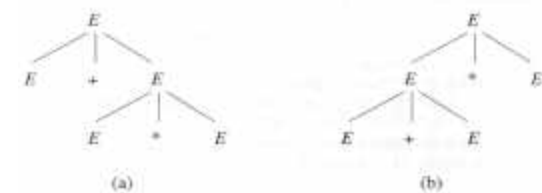
35

From Figure 5.17 of (A.T.L.C., Hopcroft, Motwani, & Ullman, 2001).

Ambiguity

Example:

- Consider *expression grammar* from Slide #5.
 - Precedence of operators* not embodied.



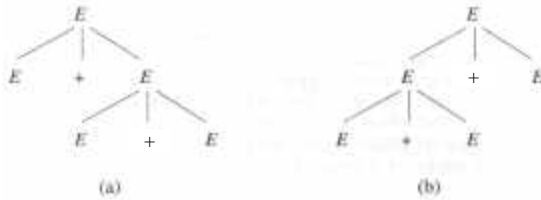
B. Juliano © 2002, based on notes by J. Ullman

36

Ambiguity

From Figure 5.17 of A TLG, Hopcroft, Motwani, & Ullman, 2001.

- Example, *continued*:
 - Consider *expression grammar* from Slide #5.
 - Left associativity for identical operators not embodied.



B. Juliano © 2002, based on notes by J. Ullman

37

Ambiguity

- Example, *continued*:
 - Consider *expression grammar* from Slide #5.
 - Solution? Introduce several different variables, each representing some “binding strength”.
 - A **factor** is an expression that cannot be broken apart by any adjacent operator.
 - A **term** is an expression that cannot be broken by the + operator; hence, a term is a product of one or more factors.
 - An **expression** will refer to any possible expression.

B. Juliano © 2002, based on notes by J. Ullman

38

Ambiguity

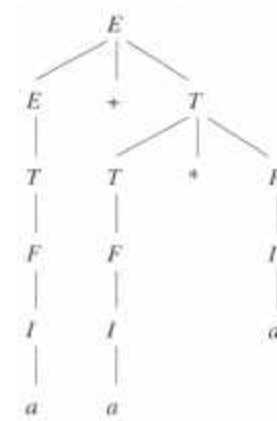
- Example, *continued*:
 - Consider *expression grammar* from Slide #5.
 - An unambiguous grammar that generates the same language as the (original) *expression grammar* of Slide #5:
 - $I \rightarrow a \mid b \mid Ia \mid Ib \mid IO \mid II$
 - $F \rightarrow I \mid (E)$
 - $T \rightarrow F \mid T * F$
 - $E \rightarrow T \mid E + T$

B. Juliano © 2002, based on notes by J. Ullman

39

Ambiguity

From Figure 5.20 of A TLG, Hopcroft, Motwani, & Ullman, 2001.



- Example:
 - Sole parse tree for $a+a*a$.

B. Juliano © 2002, based on notes by J. Ullman

40

Ambiguity

- Leftmost Derivations as a Way to Express Ambiguity

- Theorem 5.29**

- For each grammar $G=(V,T,P,S)$ and string $w \in T^*$, w has two distinct parse trees if and only if w has two distinct leftmost derivations from the start state S .



41

B. Juliano © 2002, based on notes by J. Ullman

Ambiguity

- Inherent Ambiguity**

- A CFL L is said to be *inherently ambiguous* if all its grammars are ambiguous. If even one grammar for L is unambiguous, then L is an unambiguous language.
- So what?**
 - Ambiguity of a grammar implies at least some strings in its language have different structures (parse trees).



42

B. Juliano © 2002, based on notes by J. Ullman

Ambiguity

- Inherent Ambiguity**

- Ambiguous grammars are unlikely to be useful for a programming language, because two different structures for the same thing (program) implies two different meanings (executable equivalent programs).
- An inherently ambiguous language would be absolutely unsuitable as a programming language because there would be no way to fix a unique structure for all its programs!



43

B. Juliano © 2002, based on notes by J. Ullman

Ambiguity

- There are inherently ambiguous languages.*

- Example**

$$L = \{a^n b^n c^m d^m \mid n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n \geq 1, m \geq 1\}$$

- L consists of strings in $a^+ b^+ c^+ d^+$ such that either
 - There are as many a 's as b 's and as many c 's as d 's; or
 - There are as many a 's as d 's and as many b 's as c 's.
- L is a CFL.



44

B. Juliano © 2002, based on notes by J. Ullman

Ambiguity

- Example, continued ...

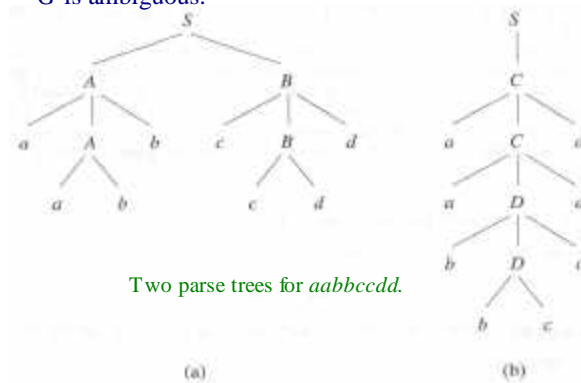
- A grammar, G , for L :

- $S \rightarrow AB \mid C$
- $A \rightarrow aAb \mid ab$
- $B \rightarrow cBd \mid cd$
- $C \rightarrow aCd \mid aDd$
- $D \rightarrow bDc \mid bc$

Ambiguity

- Example, continued ...

- G is ambiguous.



Copyright and Intellectual Property Notice

This document and its contents are the *Intellectual Property* (IP) of Dr. Benjoe A. Juliano of the Department of Computer Science at California State University, Chico (CSUC). Dr. Juliano claims exclusive moral rights of ownership under current *Copyright Laws* (Title 17 of the United States Code and 1998 Digital Millennium Copyright Act) and *IP Policies/Guidelines* (CSUC EM83-08, EM97-07, and Article 39 of the CFA/CSU Contract) including, but not limited to:

- the exclusive right to copy, reproduce, and/or distribute this document;
- the right to be identified as the creator of this work (the right of attribution);
- the right to take action against false attribution; and
- the right to object to derogatory treatment of this work (the right of integrity).