Chomsky Normal Form

Homework

- Homework #3 returned
- Homework #4 due today
- Homework #5
  - Pg 169 -- Exercise 4
  - Pg 183 -- Exercise 4c,e,i (use JFLAP)
  - Pg 184 -- Exercise 10
  - Pg 195 – Exercise 5
  - Pg 196 – Exercise 15
- Due 10 / 14

Announcements

- Final Exam Dates have been announced
  - Tuesday, November 11
  - 12:30 – 2:30 pm
  - Room TBA
- Conflicts? Let me know.

Before We Start

- Exam 1 will be returned next class
- Any questions?

Plan for today

- 1st half
  - Chomsky Normal Form
- 2nd half
  - Pushdown Automata

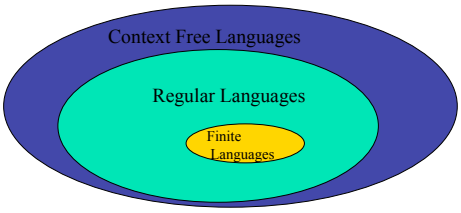
Exercises to discuss

- For after class
  - CFG from last time
  - Pumping Lemma from exam
    - Will discuss next time.
  - Algorithm from HW#3.

Languages

- Recall.
  - What is a language?
  - What is a class of languages?

The Language Bubble



Context Free Languages

- Context Free Languages(CFL) is the next class of languages outside of Regular Languages:
  - Language / grammar: Context Free Grammar
  - Machine for accepting: Pushdown Automata

Grammars

- Let's formalize this a bit:
  - A grammar is a 4-tuple: (V, T, P, S) where
    - V is a set of variables
    - T is a set of terminals
    - P is a set of production rules
    - V and T are disjoint (I.e. $V \cap T = \varnothing$)
    - $S \in V$, is your start symbol

General Grammars

- Production Rules
  - Of the form $A \rightarrow B$
    - A is a string of terminals and variables
    - B is a string of terminals and variables
    - To apply a rule, replace any occurrence of A with the string B.

Grammars

- Let's formalize this a bit:
  - Production rules
    - We say that γ can be derived from α in one step:
      - $A \rightarrow \beta$ is a rule
      - $\alpha = \alpha_1 A \alpha_2$
      - $\gamma = \alpha_1 \beta \alpha_2$
      - $\alpha \Rightarrow \gamma$
    - We write $\alpha \Rightarrow^* \gamma$ if γ can be derived from α in zero or more steps.

Context Free Grammars

- Production Rules
  - Of the form A → B
    - A is a variable
    - B is a string, combining terminals and variables
    - To apply a rule, replace an occurrence of A with the string B.
  - We say that the grammar is context-free since this substitution can take place regardless of where A is.

Context Free Grammars

- The language generated by a grammar
  - Let G = (V, T, P, S)
  - The language generated by G, L(G)
    - $L(G) = \{ x \in T^* \mid S \Rightarrow^* x \}$
- A language L is a Context Free Language (CFL) iff there is a CFG G, such that
  - L = L(G)

Chomsky Normal Form

- Chomsky Normal Form
  - A context free grammar is in Chomsky Normal Form (CNF) if every production is of the form:
    - A → BC
    - A → a
    - Where A,B, and C are variables and a is a terminal.

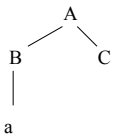
Theory Hall of Fame

- Noam Chomsky
  - The Grammar Guy
  - 1928 –
  - b. Philadelphia, PA
  - PhD – UPenn (1955)
    - Linguistics
  - Prof at MIT (Linguistics) (1955 - present)
  - Probably more famous for his leftist political views.

http://www.chomsky.info

Chomsky Normal Form

- If we can put a CFG into CNF, then we can calculate the "depth" of the longest branch of a parse tree for the derivation of a string.



At most 2 branches at every node

Chomsky Normal Form

- 3 Step process:
  1. Remove λ- Productions
  2. Remove Unit Productions
  3. Remove Useless Symbols

Removing λ-Productions

- A λ-Productions is a production of the form
  - $A \rightarrow \lambda$
- Basic idea
  - Find the set of all variables A such that $A \Rightarrow^* \lambda$ (set of nullable variables)
  - For all productions that contain a nullable variable on the right hand side, add a production that eliminates the nullable from the right hand side

Removing λ-Productions

- We must be a bit careful here
  - If λ is in a CFL, then the production $S \rightarrow \lambda$ must be in the production set.
  - The algorithm to be described will generate $L - \{\lambda\}$

Removing λ-Productions

- Step 1: Find the set of nullable variables:
  - Example:
    - $S \rightarrow AB$
    - $A \rightarrow aAA \mid \lambda$
    - $B \rightarrow bBB \mid \lambda$
    - All variables are nullable
      - A and B are nullable since $A \rightarrow \lambda$ and $B \rightarrow \lambda$
      - S is nullable since $S \rightarrow AB$ and A and B are nullable

Removing λ-Productions

- Step 2: Remove nullable variables
  - For all productions $A \rightarrow \beta$ where β contains nullable variables, add a new production with each nullable removed from β

Removing λ-Productions

Step 2: Remove nullable variables
Example:

- $S \rightarrow AB$
- $A \rightarrow aAA \mid \lambda$
- $B \rightarrow bBB \mid \lambda$
- All variables are nullable

Removing λ-Productions

- Step 2: Remove nullable variables
Example:
  - Consider: $S \rightarrow AB$
    - Add to P: $S \rightarrow A$ and $S \rightarrow B$
  - Consider: $A \rightarrow aAA$
    - Add to P: $A \rightarrow aA$ and $A \rightarrow a$
  - Consider: $B \rightarrow bBB$
    - Add to P: $B \rightarrow bB$ and $B \rightarrow b$

Removing λ -Productions

- Step 2: Remove nullable variables
  - Our grammar now looks like:
    - S → AB | A | B
    - A → aAA | aA | a | λ
    - B → bBB | bB | b | λ

Removing λ-Productions

- Step 3: Remove your λ-Productions
  - Example:
    - Remove A → λ and B → λ
    - Our final grammar looks like:
      - S → AB | A | B
      - A → aAA | aA | a
      - B → bBB | bB | b
  - Questions?

Removing Unit Productions

- A Unit Productions is a production of the form
  - A → B where A and B are variable
  - Basic idea
    - Very similar to removing λ productions
    - For each variable A, find the set of all variables B such that $A \Rightarrow^* B$ by just following unit productions (A-derivable)
    - For all variables B that are A derivable and for all productions B → α, add the production A → α

Removing Unit Productions

- Step 0: Remove λ-Productions using the previous algorithm.

Removing Unit Productions

- Step 1: For all variables A find the set of A-derivable variables:
  - Recursive definition of A-derivable
    1. If A → B then B is A-derivable
    2. If C is A derivable and C → B (and B ≠ A), then B is A derivable
    3. No other variables are A-derivable.

Removing Unit Productions

- Step 1: For all variables A find the set of A-derivable variables:
  - Example:
    - S → S + T | T
    - T → T * F | F
    - F → (S) | a
    - Let's find the set of S-derivable variables:
      - T is S derivable since S → T
      - F is S derivable since T → F and T is S derivable

Removing Unit Productions

- Step 1: For all variables A find the set of A-derivable variables:
  - Example:
    - S → S + T | T
    - T → T * F | F
    - F → (S) | a
    - S-derivable = {T, F}
    - T-derivable = {F}
    - F-derivable = ∅

Removing Unit Productions

- Step 2: For each variable A, if B is A-derivable, for each non-unit production B → β, add the production A → β

Removing Unit Productions

- Step 2:
  - Example:
    - S → S + T | T
    - T → T * F | F
    - F → (S) | a
    - S-derivable = {T, F}
    - T-derivable = {F}
    - Add to P: S → T * F, S → (S) | a
    - : T →(S) | a

Removing Unit Productions

- Step 2:
  - Our new grammar now looks like:
    - S → S + T | T * F | (S) | a | T
    - T → T * F | (S) | a | F
    - F → (S) | a

Removing Unit Productions

- Step 3: Remove Unit Productions
  - Our final grammar looks like:
  - Our new grammar now looks like:
    - S → S + T | T * F | (S) | a
    - T → T * F | (S) | a
    - F → (S) | a
    - Remove S → T, T → F
  - Questions

Removing Useless Symbols

- A symbol X is <u>useful</u> for a grammar G = (V, T, P, S) if
  - $S \Rightarrow^* \alpha X \beta \Rightarrow^* w$ where $w \in L(G)$
- In other words, a <u>useful symbol</u> will be used somewhere in the derivation of a string in the language.
- Any symbol that is not useful is <u>useless</u>.
- Useless symbols do not add to the language generated by a grammar, so it's okay to remove them.

Removing Useless Symbols

- Definitions:
  - We say a symbol X is generating if:
    - $X \Rightarrow^* w$ for some $w \in L(G)$
  - We say a symbol X is reachable if:
    - $S \Rightarrow^* \alpha X \beta$ for some α, β
- Symbols that are useful must be both generating and reachable.
  - Such symbols (and assoc. productions) can be removed

Removing useless symbols

- Algorithm:
  1. Eliminate all non generating symbols
  2. Eliminate all non reachable symbols from resultant grammar.

Removing useless symbols

- Finding generating symbols
  1. All symbols in T are generating
  2. If $A \rightarrow \alpha$ and all symbols in α are generating, then A is generating.
  3. No other symbols are generating.

Removing useless symbols

- Finding reachable symbols
  1. S is reachable
  2. If A is reachable, and $A \rightarrow \alpha$, then all variables in α are reachable.

Removing Useless Symbols

- Example:

  $S \rightarrow AB \mid a$

  $A \rightarrow b$

  B is useless since it is not generating

  Eliminate it

Removing useless symbols

- Example:

  $S \rightarrow a$

  $A \rightarrow b$

  - Now A is not reachable, eliminate it!

  $S \rightarrow a$

  Note that you must eliminate non-generating symbols before non-reachable symbols.

Recall our goal

- Chomsky Normal Form
  - A context free grammar is in Chomsky Normal Form (CNF) if every production is of the form:
    - $A \rightarrow BC$
    - $A \rightarrow a$
    - Where A,B, and C are variables and a is a terminal.

Chomsky Normal Form

- Given a CFG G, there is an equivalent CFG, G' in Chomsky Normal form such that
  - $L(G') = L(G) - \{\lambda\}$

Chomsky Normal Form

- Step 1:
  - Remove λ-Productions
- Step 2:
  - Remove Unit Productions
- Step 3:
  - Remove useless symbols

Chomsky Normal Form

- After steps 1 – 3 :
  - All productions are of the form:
    - $A \rightarrow a$ where A is a variable and a is a terminal
    - $A \rightarrow \beta$ where $|\beta| \geq 2$ and β contains variables and/or terminals.
  - Step 4: Derive terminals from new variables:
    - For all productions of the 2nd type: $A \rightarrow \beta$, for all terminals a in β, create a new variable X_a
    - Add a new production $X_a \rightarrow a$
    - Replace a in β with X_a

Chomsky Normal Form

- Step 4:
  - Let's go back to our first example:
    - $S \rightarrow AB \mid A \mid B$
    - $A \rightarrow aAA \mid aA \mid a$
    - $B \rightarrow bBB \mid bB \mid b$
  - Removing unit transitions:
    - $S \rightarrow AB \mid aAA \mid aA \mid a \mid bBB \mid bB \mid b$
    - $A \rightarrow aAA \mid aA \mid a$
    - $B \rightarrow bBB \mid bB \mid b$
  - Note that S, A, and B are all useful.

Chomsky Normal Form

- Step 4:
  - Define new productions: $X_a \rightarrow a$ and $X_b \rightarrow b$ and replace instance of a with X_a, similarly for b
    - $S \rightarrow AB \mid aAA \mid aA \mid a \mid bBB \mid bB \mid b$
    - $A \rightarrow aAA \mid aA \mid a$
    - $B \rightarrow bBB \mid bB \mid b$
  - New:
    - $S \rightarrow AB \mid X_a AA \mid X_a A \mid a \mid X_b BB \mid X_b B \mid b$
    - $A \rightarrow X_a AA \mid X_a A \mid a$
    - $B \rightarrow X_b BB \mid X_b B \mid b$
    - $X_a \rightarrow a$
    - $X_b \rightarrow b$

Chomsky Normal Form

- After steps 1 – 4 :
  - All productions are of the form:
    - A → a where A is a variable and a is a terminal
    - A → β where | β | ≥ 2 and β contains only variables.
  - Step 5:
    - For all productions of type 2 where | β | > 2 , replace the production with a series of new productions each having exactly 2 variables on the right
    - Best illustrated with an example

Chomsky Normal Form

- Step 4:
  - The production:
    - A → BCDBCE
  - Would be replaced with
    - A → BY_1
    - Y_1 →CY_2
    - Y_2 →DY_3
    - Y_3 → BY_4
    - Y_4 → CE

Chomsky Normal Form

- Step 4:
  - Back to our example
    - S → AB | $\underline{X_a}\,\underline{AA}$ | X_a A | a | $\underline{X_b}\,\underline{BB}$ | X_b B | b
    - A → $\underline{X_a}\,\underline{AA}$ | X_a A | a
    - B → $\underline{X_b}\,\underline{BB}$ | X_b B | b
    - X_a → a
    - X_b → b
  - Add productions
    - Y_1 → AA
    - $\underline{Y_2}$ →BB

Chomsky Normal Form

- Step 4:
  - Our final grammar
    - S → AB | $\underline{X_a}\,Y_1$ | X_a A | a | $X_b\,\underline{Y_2}$ | X_b B | b
    - A → $X_a\,Y_1$ | X_a A | a
    - B → $X_b\,\underline{Y_2}$ | X_b B | b
    - Y_1 → AA
    - $\underline{Y_2}$ → BB
    - X_a → a
    - X_b → b
  - Questions

CNF

- Any grammar can be placed into CNF
- Why bother?
  - Means to simplify grammars
  - Gives upper limit on size of parse tree
    - And we'll need this factoid next week.

Questions?

- Break.