

MS2a, Week 6

Rune Lyngsø

November 24, 2011

A Stochastic Context Free Grammars

- a. Consider the context free grammar G with variables $\{S\}$, alphabet $\{(\,)\}$ (i.e. left and right parentheses), start variable S , and productions

$$S \rightarrow (S) \mid SS \mid ()$$

For each of the following four strings, determine whether the string can be generated from G . If the string can be generated from G , provide a derivation generating the string.

- $()()$
- ϵ (the empty string)
- $(())$
- $()(())$

- b. Assume that Pr assigns a probability to each of the productions of G , with

$$Pr(S \rightarrow (S)) = 0.5 \quad Pr(S \rightarrow SS) = 0.3 \quad Pr(S \rightarrow ()) = 0.2$$

What is the probability of generating the string $(())$?

What is the probability of generating the string $()()$?

- c. In the grammar of question a, the string $()()$ can be derived both as $S \Rightarrow SS \Rightarrow ()S \Rightarrow ()()$ and as $S \Rightarrow SS \Rightarrow S() \Rightarrow ()()$. These two derivations are essentially the same, though, the only difference is whether we choose to first replace the first S in SS , or first replace the second S . A *leftmost derivation* is one where we always replace the leftmost variable in the current string. Only the first of the above derivations is leftmost. A grammar for which we can find a string that has at least two different

leftmost derivations, or equivalently two different parse trees, is called *ambiguous*. For each of the following three grammars, determine whether they are ambiguous. For each ambiguous grammar, provide a string and two different leftmost derivations, or parse trees, of that string, proving the ambiguousness.

- G_1 has variables $\{S\}$, alphabet $\{(,)\}$, start variable S , and productions

$$S \rightarrow (S) \mid SS \mid \epsilon$$

(remember that ϵ denotes the empty string).

- G_2 has variables $\{S, A\}$, alphabet $\{(,)\}$, start variable S , and productions

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

$$A \rightarrow (S).$$

- G_3 has variables $\{S, A\}$, alphabet $\{(,)\}$, start variable S , and productions

$$S \rightarrow AS \mid A$$

$$A \rightarrow (S) \mid \epsilon.$$

- G_4 has variables $\{W, V, I, U\}$, alphabet $\{l, u, r\}$, start variable W , and productions

$$W \rightarrow \epsilon \mid Wu \mid WV$$

$$V \rightarrow lUr \mid lUVUr \mid lIUr$$

$$I \rightarrow V \mid Iu \mid uI \mid II$$

$$U \rightarrow \epsilon \mid uU.$$

Note the close resemblance between this grammar and the recursions in equations (6)–(8) of the RNA lecture notes. An investigation into the undesirable features of ambiguity in RNA secondary structure grammars is part of [3]. There are no algorithms that for all context-free grammars can determine whether they are ambiguous, but it can be determined for large classes of context-free grammars [1].

B Grammar Design

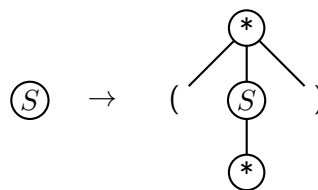
- d. We can design a context-free (even regular) grammar capturing matching two sequences

$$S \rightarrow S_{\tau}^{\sigma} \mid \epsilon$$

with $\sigma, \tau \in \Sigma \cup \{\epsilon\}$ where Σ is the alphabet of the sequences and ϵ correspond to the empty string. Notice that we use ϵ instead of a gap character, as we want the grammar to generate the two sequences rather than an alignment of the two sequences. To extract an alignment from a derivation we would just have to replace ϵ with gaps.

Can you modify this grammar such that blocks can be matched in reverse, i.e. the first characters in one sequence are matched against the last characters of the other sequence within the block?

- e. In a regular grammar, all productions are required to have the variable on the same side of the terminal symbol, i.e. we can only use productions of one of the types $U \rightarrow \sigma V$ and $U \rightarrow V\sigma$. For linear grammars we can use both. Design a linear grammar generating exactly the set of strings $\{a^i g^i \mid i > 0\}$.
- f. In tree adjoining grammars, the object we are modifying is the derivation tree rather than a string. This is done by allowing special nodes with a single parent and a single child to be expanded with a small tree, where the root and a specially designated leaf in this small tree get connected to the parent and child of the special node. In a context-free grammar, these replacements always happen at the leaves of the derivation tree. For example, we could replace the context-free rule $S \rightarrow (S)$ with the tree adjoining rule



meaning that a node of type S can be replaced with the tree shown, using the nodes marked with $*$ for connections. The string of a derivation from a tree adjoining grammar is the sequence of terminal symbols at the leaves, read from left to right. Design a tree adjoining grammar that generates

the set of strings $\{s \mid s = xx \wedge x \in \{a, g\}^*\}$, i.e. all strings of a's and g's that consist of two copies of the same string.

C Genome Rearrangement

- g. One model for genome rearrangements is sorting by reversals. In this model we assume that we can identify homologous blocks in two (or more) genomes, and rearrangement events correspond to reversing a subsequence of blocks in the current configuration. E.g. if we have four identified blocks, represented by the configuration $(1, \underbrace{2, 3}, 4)$, we can reach the configuration $(1, 3, 2, 4)$ by reversing the indicated subsequence in the original configuration. In [2] eight blocks in the X chromosome are identified as homologous between mouse and human, with the ordering in human represented by $(4, 6, 1, 7, 2, 3, 5, 8)$ when the mouse ordering is $(1, 2, 3, 4, 5, 6, 7, 8)$. What is the shortest sequence of reversals you can find that converts one into the other?
- h. The model captures the event of inverting a part of a chromosome, corresponding to replacing a stretch with its reverse complement. This means that in addition to observing the ordering we can also observe the directionality of each block, i.e. the parity of the number of times it has been inverted. We can extend the model to include this by adding a sign to each block, with negative numbers corresponding to reversed direction. Repeating the above example, we can get from the configuration $(+1, \underbrace{-2, -3}, +4)$ to the configuration $(+1, +2, +3, +4)$ by the single reversal indicated. In the mouse and human X chromosome example, the configuration becomes $(-4, -6, +1, 7, 2, -3, +5, +8)$ when extended with directionality, where the missing signs on blocks 2 and 7 indicate unknown directionality. What is now the shortest sequence of reversals you can find, and does parsimony allow you to make any statements about the directionality of blocks 2 and 7 in human compared to mouse? .

References

- [1] C. Brabrand, R. Giegerich, and A. Møller. Analyzing ambiguity of context-free grammars. *Science of Computer Programming*, 75(3):176–191, 2010.
- [2] S. D. M. Brown, P. Avner, Y. Boyd, V. Chapman, S. Rastan, L. Sefton, J. D. Thomas, and G. E. Herman. Mouse X chromosome. *Mammalian Genome*, 4:S269–S281, 1993.

- [3] R. Dowell and S. R. Eddy. Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction. *BMC Bioinformatics*, 5:71, 2004.