

# MS2a, Exercises Week 8, Model Solution

Rune Lyngsø

December 2, 2009

## A Hidden Markov Model Use

- a. Consider a hidden Markov model emitting sequences over the alphabet  $\{A, C, G, T\}$ . The model has two states, 0 and 1, that are equiprobable start states. Transition probabilities are 0.75 for remaining in a state and 0.25 for switching to the other state. In state 0 A and G are emitted with probability 0.45 while C and T are emitted with probability 0.05. In state 1 A and G are emitted with probability 0.05 while C and T are emitted with probability 0.45. What is the probability of observing the sequence ACTG? Observe that we do not have an end state providing explicit termination, so the model will not model a sequence length distribution. Rather, for every sequence length it models a distribution over sequence content. **(3 points)**

Unfortunately there was a typo in the exercises sent out, as it was stated that state 1 emits A and T with probability 0.45 – this should of course read that state 1 emits C and T with probability 0.45, as the emission probability for A has already been given as 0.05.

The probability of the observation is the sum of the probabilities of all possible ways the hidden Markov model can generate ACTG. Hence we need to invoke the forward algorithm, finally summing over all possible states we could end in rather than just look up the probability for ending in the end state. The table computed by the forward algorithm is

1	0.025	0.03375	0.012375	0.000473
0	0.225	0.00875	0.00075	0.001645
	A	C	T	G

The sum of probabilities in the last column is  $2.12 \cdot 10^{-3}$ .

- b. What is the most likely sequence of hidden states, and how probable is it? **(2 points for probability, 2 points for annotation)**

Here we need to find the most likely path in the HMM generating ACTG, rather than the sum over all paths, and hence it is the Viterbi algorithm that should be applied. Doing this, we get the following table with backtrack of the largest value in the last column shown by arrows:

1	0.025	0.025313 ←	0.008543 ↘	0.000320
0	0.225 ↗	0.008438	0.000316	0.000961
	A	C	T	G

So the most likely annotation of the sequence with states is  $\overset{0}{A}\overset{1}{C}\overset{1}{G}\overset{0}{T}$  which has probability  $9.61 \cdot 10^{-4}$ . Note that each symbol is annotated with the state most likely to emit it. This is not surprising if we take a closer look at the probabilities: the ratio between emission probabilities is about thrice the ratio between the transition probabilities. Still, the most likely path contributes less than half the total probability of the sequence.

- c. What is the most likely hidden state at position 2, summing over all possible paths, and how probable is it? **(3 points)**

We have just determined the state at position 2 in the most likely sequence of hidden states. But here we need to sum over all possible sequences of hidden states. Hence we need to combine the probability of all paths generating the sequence up to position 2 with all paths generating the sequence after position 2. I.e. we need to invoke the backward algorithm. The table computed by the backward algorithm, where we for convenience omit the emission probability of the first symbol, is

1	0.018938	0.055	0.15	1
0	0.007313	0.03	0.35	1
	A	C	T	G

Multiplying with the corresponding entries of the forward table we get a posterior probability of state 0 of  $2.625 \cdot 10^{-4}$  and a posterior probability of state 1 of  $1.856 \cdot 10^{-3}$  (observe that these two probabilities sum to

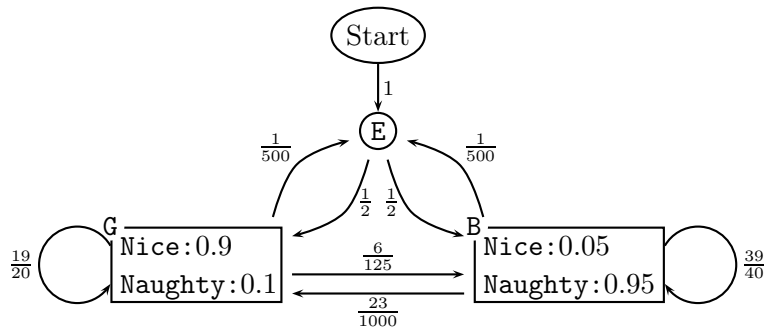
the total probability of ACTG, as they should). Hence state 1 is the most probable at position 2, carrying about 88% of the probability mass.

## B Hidden Markov Model Design

- a. The occasionally dishonest casino is a standard HMM example. Given the season and the problems ludomania causes in modern society, we will consider a rephrased version of this example. **Infinity Road** is an endless row of houses with a child living in most. Given the length of the street, Santa and his elves cannot constantly check the behaviour of all the children in the road, but checks just one deed per child each year to see whether it is **Naughty** or **Nice**. However, even a **Nice** child may transgress and perform a **Naughty** act (with probability 10% i.i.d. for all **Nice** children), and a **Naughty** child may inadvertently find himself doing something that can be classified as **Nice** (with probability 5%, again i.i.d. for all **Naughty** children). Santa would like to do better than basing his judgement on just a single observation, and it just so happens that **Infinity Road** segments into **Good** neighbourhoods and **Bad** neighbourhoods, both of lengths that are geometrically distributed and with an expected length of a **Good** neighbourhood of 20 houses and an expected length of a **Bad** neighbourhood of 40 houses. All children living in a **Good** neighbourhood are **Nice**, and all children living in a **Bad** neighbourhood are **Naughty**. It is equally likely that **Infinity Road** starts with a **Good** neighbourhood as with a **Bad** neighbourhood. Houses with no children living in them are distributed uniformly at random, with on average one out of every 500 houses not having a child living in it. Neighbourhoods either side of one or more childless houses are uncorrelated, such that the neighbourhood starting after a childless house has equal chance of being **Good** and **Bad**. Design a hidden Markov model that can help Santa use the observations for all the children to annotate each child as either **Naughty** or **Nice** – don't worry that it would normally take infinitely long time to annotate an infinitely long sequence. **(3 points)**

The annotation problem is one of annotating each deed, whether **Naughty** or **Nice**, with a neighbourhood state, either **Good** or **Bad**, as the neighbourhood state yields the overall inclination of a child regardless of the character of the observed deed. If we start with the childless houses, there are no observations for these so a *silent* state seems appropriate (alternatively, one can choose to also have **Empty** as a possible observation, only emitted from childless houses; this will improve annotation as

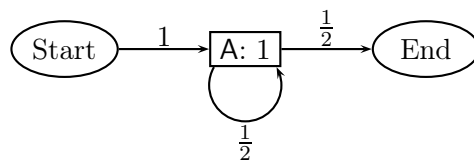
the location of empty houses does provide further information). From all states, we need to go to this silent state with probability  $\frac{1}{500}$ . This also holds for the state itself, but as it is silent we may as well skip the self-loop on this state and just transition to either a Good or a Bad neighbourhood with equal probability  $\frac{1}{2}$ . This means that we may also use this for the initial distribution. Otherwise we need two *non-silent* states, one for each type of neighbourhood, and the size expectations immediately specifies the probability of the self-loop transition for each. The remaining probability is assigned the transition switching to the other type of neighbourhood. These observations give the following HMM



where the state **G** models houses in **Good** neighbourhoods, the state **B** models houses in **Bad** neighbourhoods, and the state **E** models childless houses.

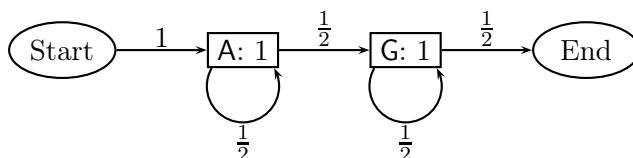
- b. Construct a HMM that generates the sequence  $A^i$ , i.e. the sequence of  $i$   $A$ s, with probability  $2^{-i}$  for  $i \geq 1$ , if possible. Otherwise argue it is not possible. **(1 point)**

This is just a geometric distribution on  $i$  with parameter  $1/2$ , so we just need a single state emitting the  $A$ 's, with probability  $1/2$  of staying in the state after an emission and probability  $1/2$  of going to the end state:



- c. Construct a HMM that generates sequences over the alphabet  $\{A, G\}$  with probability  $(k-1)2^{-k}$  for generating a sequence of length  $k \geq 2$ , and for which all sequences that are generated of length  $k$  are on the form  $A^i G^{k-i}$  and equiprobable. (1 point)

Consider the HMM



Evidently it can only generate sequences on the form  $A^i G^{k-i}$ ,  $k \geq 2, 1 \leq i < k$ , and for a given  $k$  all sequences that can be generated of length  $k$  have probability  $2^{-k}$ : there is a unique run generating a particular sequence of length  $k$ , and in this run all emissions have probability 1 while there is one transition with probability 1 and  $k$  transitions with probability  $\frac{1}{2}$ . It follows that the total probability of generating a sequence of length  $k$  is  $(k-1)2^{-k}$ .

- d. Construct a HMM that generates the sequence  $A^i G^i$  with probability  $2^{-i}$  for  $i \geq 1$ , if possible. Otherwise argue that it is not possible. (1 point)

For this to work, we would need to know how many A's were emitted when we start emitting the G's. But this means that we need to be able to remember arbitrarily far back, which should tell us we will be violating the Markov property.

More formally, assume that there is a hidden Markov model  $M$  with  $n$  states that emits sequences according to the required distribution. Consider a run emitting the sequence  $A^n G^n$ . This will pass through  $2n$  states, so there must be a state  $q$  that is visited at least twice. We can write  $A^n G^n$  as  $xyz$ , where the non-empty sequence  $y$  is the part of  $A^n G^n$  emitted between the first visit to  $q$  and the last visit to  $q$ . Evidently the sequence  $xyyz$  can be generated with non-zero probability, as we could repeat the path taken between the first visit to  $q$  and the last visit to  $q$  one more time. As  $M$  cannot emit sequences with G's preceding A's,  $y$  must be on the form  $A^i G^j$ . But if  $i \neq j$ , then  $xyyz$  does not contain the same number of A's and G's and should thus be emitted with probability zero. Hence  $i = j > 0$ , as  $y$  is non-empty. But then  $xyyz = A^n G^i A^i G^n$ , which conflicts the fact that G's cannot



What is the probability of generating the string  $()()$ ? **(1 point)**

This string can only be generated by the derivation tree shown above, which contains one application of the rule  $S \rightarrow (S)$  and one application of the rule  $S \rightarrow ()$ , hence the probability is  $0.5 \cdot 0.2 = 0.1$ .

What is the probability of generating the string  $()()()$ ? **(1 point)**

This string has two possible derivations. Apart from the one shown above, it can also be generated by the derivation tree that is the mirror image (so where it is the second  $S$  from the first application of the  $S \rightarrow SS$  rule that is expanded using this rule once more, rather than the first  $S$ ). Both derivation trees have two applications of the  $S \rightarrow SS$  rule and three applications of the  $S \rightarrow ()$  rule, so the total probability of generating this string is  $2 \cdot 0.3^2 \cdot 0.2^3 = 1.44 \cdot 10^{-3}$ .

- 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 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 of that string that proves the ambiguousness.

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

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

(remember that  $\epsilon$  denotes the empty string). **(1 point)**

This grammar is ambiguous as e.g. the empty string can be generated by infinitely many leftmost derivations, two of which are  $S \Rightarrow \epsilon$  and  $S \Rightarrow SS \Rightarrow \epsilon S \Rightarrow \epsilon$ .

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

$$\begin{aligned} S &\rightarrow AS \mid \epsilon \\ A &\rightarrow (S). \end{aligned}$$

**(1 point)**

This grammar is unambiguous. We can prove this by induction. The basis is that for all strings of length 0 that can be generated by  $G_2$ , there is only one way to generate them. This follows from variable  $A$  always generating strings containing at least two symbols, so to generate the empty string we must apply the  $S \Rightarrow \epsilon$  derivation straight away.

Now assume that we want to prove there is only one way to generate the string  $s$  and we know the claim holds for all shorter strings. If  $s$  is not the empty string, then the first production used must be  $S \Rightarrow AS$ . The next step in any leftmost derivation is  $AS \Rightarrow (S)S$ , so  $s = (x)y$  where both  $x$  and  $y$  are derived from  $S$  and shorter than  $s$ . Hence, for any split of  $s = (x)y$  there is only one derivation by the induction hypothesis. One can further prove that we can only derive balanced strings of parentheses from  $S$ . It follows that there is only one way to split  $s$  into  $(x)y$  such that both  $x$  and  $y$  can be derived from  $S$ , namely by splitting at the end of the first prefix of  $s$  that is itself a balanced string of parentheses.

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

$$\begin{aligned} S &\rightarrow AS \mid A \\ A &\rightarrow (S) \mid \epsilon. \end{aligned}$$

**(1 point)**

Again we have a grammar with an infinite number of derivations for the empty string, e.g.  $S \Rightarrow A \Rightarrow \epsilon$  and  $S \Rightarrow AS \Rightarrow \epsilon S \Rightarrow A \rightarrow \epsilon$ .

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

$$\begin{aligned} W &\rightarrow \epsilon \mid Wu \mid WV \\ V &\rightarrow lUr \mid lUVUr \mid lIUr \\ I &\rightarrow V \mid Iu \mid uI \mid II \\ U &\rightarrow \epsilon \mid uU. \end{aligned}$$

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 [2]. 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].

(1 point)

This grammar is ambiguous. The problem is with the productions of  $I$ , where we can add a  $u$  to either side of  $I$ . This means that we can add  $u$ 's on both sides of  $I$  by first adding the one to the left and then the one to the right, or vice versa. The  $I \rightarrow II$  production is also a problem, as we can choose to double either the lefthand or the righthand  $I$  in a string with two neighbouring  $I$ 's obtaining the same new string. In derivation terms this becomes  $W \Rightarrow WV \Rightarrow \epsilon V \Rightarrow IIR \Rightarrow IVIr \Rightarrow IUrIr \Rightarrow I\epsilon rIr \Rightarrow IruIr \Rightarrow IruIur \Rightarrow IruVur \Rightarrow IruUur \Rightarrow Iru\epsilon ur = IruIur = IruUur \Leftarrow IruUur \Leftarrow IruVur \Leftarrow IruIur \Leftarrow IruIur \Leftarrow I\epsilon rIr \Leftarrow IUrIr \Leftarrow IVIr \Leftarrow IIR \Leftarrow \epsilon V \Leftarrow WV \Leftarrow W$ .

## References

- [1] C. Brabrand, R. Giegerich, and A. Møller. Analyzing ambiguity of context-free grammars. In *Proc. CIAA '07*, volume 4783 of *LNCS*. Springer-Verlag, 2007.
- [2] R. Dowell and S. R. Eddy. Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction. *BMC Bioinformatics*, 5:71, 2004.