UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR08008 INFORMATICS 2A: PROCESSING FORMAL AND NATURAL LANGUAGES

Wednesday 21st August 2013

14:30 to 16:30

INSTRUCTIONS TO CANDIDATES

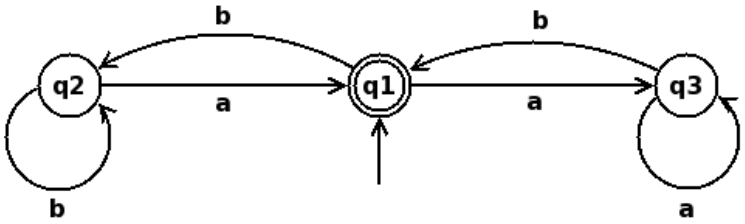
1. **Answer all five questions in Part A, and two out of three questions in Part B. Each question in Part A is worth 10% of the total exam mark; each question in Part B is worth 25%.**
2. **Use a single script book for all questions.**
3. **Calculators may be used in this exam.**

Convener: J Bradfield
External Examiner: A Preece

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

PART A
ANSWER ALL QUESTIONS IN PART A.

1. Consider the DFA below over the alphabet $\Sigma = \{a, b\}$.



   (a) Based on the DFA, write down simultaneous Kleene-algebra equations in three variables X_1, X_2 and X_3, representing the languages accepted if the DFA is run starting in states $q1$, $q2$ and $q3$ respectively. [*4 marks*]

   (b) Use Arden's Rule to solve the simultaneous equations and produce a regular expression for the language accepted by the DFA (whose start state is $q1$). [*6 marks*]

2. Consider a pushdown automaton with two control states $Q = \{q1, q2\}$, input alphabet $\Sigma = \{a, b\}$, stack alphabet $\Gamma = \{\bot\}$, transition relation:

$$\begin{aligned} q1 &\xrightarrow{a, \bot : \bot\bot} q1 \\ q1 &\xrightarrow{\epsilon, \bot : \epsilon} q2 \\ q2 &\xrightarrow{b, \bot : \epsilon} q2 \\ q2 &\xrightarrow{\epsilon, \bot : \epsilon} q2 \end{aligned}$$

   and start state $q1$. The automaton accepts on empty stack.

   (Here, we use the general notation

$$q \xrightarrow{s, x : \alpha} q'$$

   to mean that when the automaton is in control state $q \in Q$ and $x \in \Gamma$ is popped from the top of the stack, the input symbol or empty string $s \in \Sigma \cup \{\epsilon\}$ can be read to reach control state $q' \in Q$ with $\alpha \in \Gamma^*$ pushed onto the stack.)

   (a) Describe in detail an execution of the above PDA that accepts the string

$$aab$$

   [*8 marks*]

   (b) Give a precise mathematical definition of the language accepted by the PDA above. [*2 marks*]

3. A context-free grammar for *clauses* in propositional logic is given below using two nonterminals Lit for *literals* and Clause for clauses. The terminal symbols are

$$\mathsf{Var} \quad \vee \quad \neg$$

where Var is a lexical class of *propositional variables*, $\vee$ is the disjunction (or) operation, and $\neg$ is the negation (not) operation. The grammar has productions:

$$\begin{aligned} \mathsf{Lit} &\rightarrow \mathsf{Var} \mid \neg\ \mathsf{Var} \\ \mathsf{Clause} &\rightarrow \mathsf{Lit} \mid \mathsf{Clause} \vee \mathsf{Clause} \end{aligned}$$

The start symbol is Clause.

(a) This grammar is *structurally ambiguous*. State precisely what this means, and provide an example phrase, accompanied by parse trees, that illustrates the ambiguity. [*6 marks*]

(b) Construct an LL(1) grammar that is equivalent to the grammar for clauses above. [*4 marks*]

4. Consider the grammar

$$\begin{aligned} \mathsf{S} &\rightarrow \mathsf{NP}\ \ \mathsf{V} \\ \mathsf{NP} &\rightarrow \mathsf{N} \mid \mathsf{N}\ \mathsf{N} \end{aligned}$$

where S is the start symbol, and N and V are terminals denoting parts of speech as follows:

$$\mathsf{N} = \{ \text{ bread, price } \} \qquad \mathsf{V} = \{ \text{ rises, falls } \}$$

Use the *Earley algorithm* to parse the sentence

bread price rises

You should show the execution of the algorithm as a table with a row for each step. Each row should include the start and end position of the portion of input processed, and a letter P, S or C to indicate whether the step is due to the predictor, scanner or completer. To get you started, the first row of your table should be

$$\mathsf{S} \rightarrow \bullet\ \mathsf{NP}\ \ \mathsf{V} \qquad [0,0] \qquad \mathrm{P}$$

[*10 marks*]

5. (a) Consider the following context-free grammar and associated semantics:

S →	Name is a NP	{ NP.Sem(Name.Sem) }
NP →	N Rel	{ λx.N.Sem(x) ∧ Rel.Sem(x) }
Rel →	who V Name	{ λy.V.Sem(x,Name.Sem) }
Rel →	Name V	{ λy.V.Sem(Name.Sem,y) }
V →	likes	{ λuv.*likes*(u,v) }
N →	child	{ λz.*child*(z) }
Name →	Anna	{ *Anna* }
Name →	Bill	{ *Bill* }

Write down the lambda expression that this yields for the parsed sentence

(S (Name Bill) is a (NP (N child) (Rel (Name Anna)(V likes))))

[*5 marks*]

(b) Show how your lambda expression from part (b) reduces via a series of β-reductions to a formula of first order logic. [*5 marks*]

PART B
ANSWER TWO QUESTIONS FROM PART B.

6. The LL(1) grammar below provides a grammar for a simplified fragment of XML (Extensible Mark-up Language) documents. The five terminal symbols are

   <a>   text

   In the context of XML, these represent opening tags (<a> and), and closing tags (and) for two elements (a and b), and a lexical class text for the text content of XML elements.[1]

   The grammar has just one nonterminal symbol, Doc. The productions are:

   Doc → ϵ | text | <a> Doc Doc | Doc Doc

   We write L_{XML} for the language recognised by the above grammar.

   (a) Is the language L_{XML} regular? If it is regular then justify your answer by producing an NFA or regular expression that recognises the language. If it is not regular then prove this using the pumping lemma. [*7 marks*]

   (b) Calculate the *first* and *follow* sets for the nonterminal symbol Doc. [*6 marks*]

   (c) Construct the parse table for the grammar. [*6 marks*]

   (d) State what goes wrong when the LL(1) predictive parsing algorithm is applied to the phrases below, none of which is in L_{XML}. In each case, give an appropriate error message that the parser might return at the point at which it gets stuck.

   i. <a> text
   ii. <a> text
   iii. <a> text

   [*6 marks*]

[1] Don't worry if you have not met XML before. You do not need any knowledge of XML to answer this question.

7. In this question we will consider how probabilities may be assigned to context-free grammar rules using a corpus of parsed sentences. The grammar we will consider has non-terminals S, VP, NP, PP with S as the start symbol. The terminals are the English words 'the' and 'with', along with the identifiers Name, N, V for parts of speech defined as follows:

$$\begin{aligned} \mathsf{N} &= \{\text{man, bird, wood, chainsaw, telescope}\} \\ \mathsf{V} &= \{\text{knew, cut, saw}\} \\ \mathsf{Name} &= \{\text{John, Susan, Alice, Peter, Mary}\} \end{aligned}$$

The rules of the grammar are:

$$\begin{aligned} \mathsf{S} &\rightarrow \mathsf{Name\ \ VP} \\ \mathsf{VP} &\rightarrow \mathsf{V\ NP} \mid \mathsf{V\ NP\ PP} \\ \mathsf{NP} &\rightarrow \text{the}\ \ \mathsf{N} \mid \text{the}\ \ \mathsf{N\ PP} \\ \mathsf{PP} &\rightarrow \text{with}\ \ \mathsf{NP} \end{aligned}$$

We shall consider the following corpus of five parsed sentences:

(S (Name John)(VP (V knew)(NP the (N man)(PP with (NP the (N chainsaw))))))
(S (Name Susan)(VP (V cut)(NP the (N wood))(PP with (NP the (N chainsaw)))))
(S (Name Alice)(VP (V saw)(NP the (N wood))))
(S (Name Peter)(VP (V saw)(NP the (N man))))
(S (Name Mary)(VP (V saw)(NP the (N bird))(PP with (NP the (N telescope)))))

(a) Briefly explain how, in general, probabilities for (non-lexicalized) context-free rules may be calculated from a parsed corpus. [*2 marks*]

(b) Apply this method to obtain a probability for each of the above grammar rules using the given corpus. [*5 marks*]

(c) Now suppose we use our probabilistic grammar to analyse the sentence:

John saw the man with the telescope

Draw all possible parse trees for this sentence. Calculate the probability for each of these tree structures ignoring the expansion of the terminals Name, N, V to particular lexical items (in other words, using only the rule probabilities obtained in part (b)). [*6 marks*]

(d) Suppose now that in the above grammar, we *lexicalize* VP with respect to its head verb, and NP with respect to its head noun. (We do not lexicalize S or PP.) Write down the rules for expanding the lexicalized non-terminals VP[saw], NP[man] and NP[telescope]. Use the given corpus to assign a probability to each of these rules. Also use the corpus to assign probabilities to the head selection rules for these four non-terminals (e.g. $\mathsf{VP} \rightarrow \mathsf{VP}$[saw]). [*8 marks*]

QUESTION CONTINUES ON NEXT PAGE

(e) From each of the parse trees obtained in part (c), we now obtain a corresponding parse tree for the lexicalized grammar. Calculate the probability of each of these trees according to the lexicalized rule probabilities. This time, you should take account of the expansion of the terminal Name, assuming all five names are equally probable. [*4 marks*]

8. The children's memory game *Granny went to market* has the following pattern. The first player tells the others about an item that Granny bought when she went to market, by saying, for example,

   Granny went to market and she bought some apples.

   Successive players then continue by adding items to the list of purchases made by Granny, always repeating all previous items, maintaining their order. For example, play might continue:

   Granny went to market and she bought some apples and some bananas.
   Granny went to market and she bought some apples and some bananas and some carrots.

   Consider the following grammar for generating the full text of legitimate plays of the game. Suppose we have a fixed (finite) collection of terminals representing *items* that Granny might buy; for example it might include:

   $$\mathsf{apples}\ \ \mathsf{bananas}\ \ \mathsf{carrots}\ \ \mathsf{dates}\ \ \mathsf{eggs}\ \ \mathsf{figs}\ \ldots$$

   In addition, we have further terminals:

   $$\mathsf{Granny}\ \ \mathsf{went}\ \ \mathsf{to}\ \ \mathsf{market}\ \ \mathsf{and}\ \ \mathsf{she}\ \ \mathsf{bought}\ \ \mathsf{some}\ \ \mathsf{.}$$

   We write Σ for the set of *all* terminals, and I for the subset of *item* terminals.

   Our grammar is now specified using nonterminals

   $$\mathsf{S}\ \ \mathsf{E}\ \ \mathsf{G}\ \ \mathsf{B_i}$$

   where there is one nonterminal $\mathsf{B_i}$ for each *item* terminal i. The start symbol is S and the productions are:

   $$\begin{array}{rcll} \mathsf{S} & \to & \mathsf{E\ B_i} \mid \mathsf{E\ S\ B_i} & \\ \mathsf{E\ B_i} & \to & \mathsf{G\ i\ .} & \mathsf{i} \in I \\ \mathsf{t\ B_i} & \to & \mathsf{B_i\ t} & \mathsf{i} \in I,\ \ \mathsf{t} \in \Sigma - \{\mathsf{bought}\} \\ \mathsf{G\ B_i} & \to & \mathsf{B_i\ G\ i\ and\ some} & \mathsf{i} \in I \\ \mathsf{G} & \to & \mathsf{Granny\ went\ to\ market\ and\ she\ bought\ some} & \end{array}$$

   Here, we use i to range over item terminals, and t to range over all terminals (including item terminals) except for bought. When the same symbol (i or t) occurs more than once in the same production, each occurrence of the symbol has to be instantiated in the same way. So, for example, the following are both valid instances of the third production above

   $$\begin{array}{rcl} \mathsf{.\ B_{apples}} & \to & \mathsf{B_{apples}\ .} \\ \mathsf{bananas\ B_{apples}} & \to & \mathsf{B_{apples}\ bananas} \end{array}$$

(a) What type of grammar is the above an example of? Explain your answer. Based on the form of the grammar, what can you say about where the language of legitimate plays lies in the Chomsky hierarchy? [*3 marks*]

(b) Give full derivations of

i. Granny went to market and she bought some apples **.** [*3 marks*]

ii. Granny went to market and she bought some apples **.** Granny went to market and she bought some apples and some bananas **.** [*7 marks*]

We now consider two variations of the game. In Variation 1, players, in turn, declare a single item that Granny bought at the market. At the end, the last player, instead of introducing a new item, recalls all the items previously declared, in the order in which they were originally given. For example, the following is the full text of a possible play of Variation 1.

> Granny went to market and she bought some apples **.** Granny went to market and she bought some bananas **.** Granny went to market and she bought some apples and some bananas **.**

In Variation 2, play proceeds as in Variation 1, except that the last player has to recall all the items previously declared in reverse order. For example, the following is a legitimate play of Variation 2.

> Granny went to market and she bought some apples **.** Granny went to market and she bought some bananas **.** Granny went to market and she bought some bananas and some apples **.**

We assume that there are at least two players. An example of a 2-player play (of both Variation 1 and Variation 2) is:

> Granny went to market and she bought some apples **.** Granny went to market and she bought some apples **.**

We write L_1 for the language of all legitimate plays of Variation 1, and L_2 for the language of all legitimate plays of Variation 2.

(c) Just one of the languages L_1 and L_2 is context free. Say which! Provide a context-free grammar for that language. [*6 marks*]

(d) For the remaining non-context-free language (L_1 or L_2), provide a noncontracting grammar for the language. [*6 marks*]