Solutions to Old Final Exams (For Fall 2007)

CS 381 (Fall 2002, Fall 2004, Fall 2005, Fall 2006)

Yogi Sharma

Disclaimer: I, Yogi Sharma, do not claim these solution to be complete, or even to be absolutely correct. Use at your own risk. I just wrote them hurriedly, and, although the chances are small, they might contain some error(s).

1 Fall 2002

1.1 Problem 1

The set remains regular. Given an automaton for R, you could make automaton for the set which has two b's erased. The new automaton will behave similar to the automaton for R but will guess twice during its run that a b was present in the original string at the point of guessing, and in the end it checks that the string is accepted.

1.2 Problem 2

The resulting set might not even be context-free. Consider $\{(abc)^*\}$. After rearrangement, it becomes $\{a^n b^n c^n \mid n \geq 0\}$ which is not context-free.

1.3 Problem 3

It is context free. The pda guesses whether $i = k$ or $j = l$. Once this guess it made, it ignores the irrelevent symbols. For example, if it guesses that $j = l$, then it ignore a's, pushes b's, ignores c's and pop b's on seeing d's.

1.4 Problem 4

(a) It is not. The language $\{a^n b^n c^m\}$ is context free, so is $\{a^m b^n c^n\}$. But their intersection is our favorite non-context-free language, that is $\{a^n b^n c^n\}$.

(b) It is closed under homomorphism. This can be seen by proving that CFL's are closed under substitution and homomorphism is a special case of substitution. (See book page 284 for proof.)

(c) It is closed under inverse homomorphism too. The construction is a little involved though, although the idea is simple. See page 289 in the book.

1.5 Problem 5

The halting problem (HP) is undecidable. We did this in the review session on Monday. The idea is to prove that if HP were decidable, then L_D will be recursive, a disaster and a mathematical calamity.

1.6 Problem 6

(a) The set of finite automaton that accept no string.

(b) Does not exist.

(c) The same as in first part, the set if RE (in fact it is recursive), and the complement is recursive.

(d) The same as in first part. Although if a set and its complement both are RE, then both must be recursive.

(e) $\{(M, w) \mid M \text{ halts on } w\}$.

(f) $\{M \mid L(M) \text{ is finite}\}$, i.e., the set of all Turing machines whose language is finite.

2 Fall 2003

The solutions are available on the website.

3 Fall 2004

3.1 Problem 1

The set is still regular. Let us say the old automaton for S is M and we want to construct the automaton N for the new set. Note that if there is an adjacent pair of 1 in the new string, the old string could have an even block of

0's between them. N behaves similar to M, but on ϵ-moves, the automaton N "inserts" the block of even length (0 or more) between adjacent 1's. It checks that the string is accepted (by M) in the end.

3.2 Problem 2

No. If it were, then CFL's would be closed under intersection using De Morgan's law, a contradiction.

$$A \cap B = (A^c \cup B^c)^c$$

3.3 Problem 3

This was in your prelim 2, as well in homework.

3.4 Problem 4

The proof is in the book. The idea is the following. Let $\mathcal{P}$ be a non-trivial property of RE sets such that the empty set (empty language $\emptyset$) does not have the property $\mathcal{P}$. (The case when empty set has the property is similar, we consider $\mathcal{Q} = \bar{\mathcal{P}}$.) Now, we reduce the halting problem to deciding the question whether the language accepted by Turing machine M has property $\mathcal{P}$. Since the property $\mathcal{P}$ is non-trivial, there is a language L (and a corresponding Turing machine M_L with $L(M_L) = L$) such that $\mathcal{P}(L) = \text{true}$. The reduction is as follows. Given an instance of the halting problem (M, w), the new machine N (which we will feed to the hypothetical decider of $\mathcal{P}$) simulates M on w. If the simulation does not halt, machine N goes into infinite computation. If M halts on w, then machine N simulates M_L on its input.

Now we need to prove that this reduction works. It is instructive to do this, please go over the proof yourself.

3.5 Problem 5

(a) It means that any problem in NP can be reduced to the NP-complete problem via a polynomial time reduction. Also, an NP-complete problem is itself in NP.

(b) Given (G, k) (a graph G and a number k), is there a clique of size k in the graph?

(c) See page 448 in the book. It proves that the indepedent set problem is NP-complete. The instance (G, k) for clique and (H, k) for indepedent set are same, if H is the complement graph of G, that is a edge is there in H if and only if it is not there in G. This proves that the clique problem is NP-complete.

(Note: I am not sure whether this was done in the class this time or not. You should ask Prof. Hopcroft whether this type of questions are included in the exam or not.)

4 Fall 2005

4.1 Problem 1

The automaton contains four states. I am not going to draw it, but the idea is very similar to one of the homework problems that asked you to accept all strings in which number of 0's was a multiple of 5 and number of 1's was a multiple of 3 (or something like that).

4.2 Problem 2

$R_{i,j}^k$ denoted a regular expression, such that $L\left(R_{i,j}^k\right)$ was the set of all strings x which can lead the automaton from state i to state j without ever going through states higher than k. (The automaton can pass through states i and/or j even if they are greater than k, but no other state higher than k.)

4.3 Problem 3

It is not context free. If n is the integer of the pumping lemma, then use the string $a^n b^{2^n}$. As it turns out, a much smaller string also suffices, that is $a^{\lceil \log n \rceil} b^{2^{\lceil \log n \rceil}}$, but never mind.

4.4 Problem 4

It is context free. The strings of the language are of the form $a^i b^j c^k$ such that either $i \neq j$ or $j \neq k$. You can write grammar for both of the sets, and union them. For writing the grammer for $i \neq j$, consider two cases: $i < j$ and $i > j$ and write grammar for both of them, and union them.

4.5 Problem 5

(a) L_1 cannot be expressed in terms of L_2 (after applying whatever is mentioned). The reason being that L_2 is context free, L_1 is not, and all the operations specified preserve "context-free-ness."

(b) L_2 can be expressed in terms of L_1. I am not going to write all the steps, but the idea is the following. First put hats on some b (and not on some other) using inverse homomorphism. Keep only those string with one unhatted b (intersection with regular sets). Map a to 0, b to 1, $\hat{b}$ to ϵ, and c to 0 (using homomorphism).

4.6 Problem 6

No, they are not. For example, halting problem is RE, but complement of halting problem in non-RE (since if it were RE, then halting problem would have been decidable, a contradiction).

4.7 Problem 7

See Fall 2002 final, Problem 5.

4.8 Problem 8

If we could reduce an NP-complete problem to a problem in P, then NP will be equal to P. If the reduction required exponential time, then there is not special consequence. (In fact any NP-complete problem can be reduced to a polynomial time solvable problem using exponential time reduction.)

5 Fall 2006

5.1 Problem 1

$b^*(abb^*)^*a^*$

5.2 Problem 2

It is not context free. (As a heuristic, if there are two conditions to be checked, or two numbers to be compared, the language is not CFL. Of course, this is not a proof, but it might give a good starting point.)

We will prove this by contradiction. Let it were cfl. Then there exists n such that blah, blah... (see page 275 for what is blah, blah). We will

choose the string $a^n b^n c^n$. Now the advesary gives us a partition. The string vw could be of the following five forms a^+, a^+b^+, b^+, b^+c^+, or c^+. In this five case, pump down, pump down, pump down, pump up, and pump up respectively to show that the resulting string is not in the language. (In the exam, you will of course need to write more detail.)

5.3 Problem 3

Let the states of pda be Q and states of dfa be R. Then you could make new state set equal to $Q \times R \times \{0, 1, 2\}$. Start in the start state of pda and start state of dfa, and 0 as the third component. In every move, the third component will increase by 1 modulo 3. When the third component is 0, you make a move in the first component (corresponding to pda), otherwise, you make a move in the second component (according to dfa). accepting states are $F \times G \times \{0\}$, if F is the set of accepting states of pda, and G is the accepting states of dfa.

5.4 Problem 4

The idea is the following: puts hats on some symbols (and not other others) using inverse homomorphism. Retain hats on everything except first two IDs (using intersection with regular set). Then map all hatted symbols to ϵ (using homomorphism).

5.5 Problem 5

It is non-RE. This is the complement of the halting problem, see problem 6 in Fall 2005 final (above) for a quick argument.

5.6 Problem 6

(a) NP-hard. (At least as hard as NP-complete problem.)

(b) B could be anything.

(c) B could be anything.

(d) B is at least as hard as A, but nothing more can be said.

(e) B must requires polynomial time for deciding.

(f) B could be anything.

5.7 Problem 7

It is a boolean formula prefixed by $\forall$ and $\exists$ quantifiers which are over the variables. For example, $\forall x_1 \exists x_2 (x_1 \wedge x_2)$.

QBF is in PSPACE (in fact it is complete for PSPACE). See page 483 in the text book.

(Note: I am not sure whether this is covered in the class this time or not. Talk to Prof. Hopcroft.)