Theoretical Foundations of Computer Science

2015 Assignment

This assignment will not be submitted, but the questions in the assignment test will be based directly on the questions below. Note that while some questions may be identical, the general idea is that the concepts covered in the questions are the same. In other words you need to understand the answers to these questions, not just memorize them.

# Section 1 - Classification

For each of the following problems, classify it into a category (Regular, Context-Free, Decidable, Undecidable) and prove its membership of that category. You can use Tier 1-3 and 4+ instead of these categories if you wish.

To prove membership, do the following:

* Regular – provide a DFA, NFA or RE that solves the problem
* Context-Free – provide a proof (preferably using the pumping lemma) that it is not Regular and provide a PDA or CFG that solves the problem
* Turing Decidable – provide a proof (preferably using the pumping lemma for CFGs) that it is not Context-Free and provide a high-level description of a Turing Machine M and prove that M is a decider. If the format of the problem precludes the application of PDAs or CFGs the pumping lemma proof can be omitted and the format problems should instead be used as an argument.

OR you can prove that a problem is Turing Decidable by reduction to and from a known Turing Decidable problem.

OR some sensible combination of the two options above.

* Turing Undecidable – prove (using reduction from ATM) that the problem is undecidable

If you cannot prove or disprove membership formally, give a convincing argument. If you need to make assumptions, state these clearly. If you are not sure of a term, make sure that you look up a definition from a reliable source.

## PROBLEM 1:

Compute whether a directed graph, G, is connected.

## PROBLEM 2:

A machine designed for counting coins is to have an additional circuit added. The existing machine will send a bit string representing each coin to the new circuit (11 is for $2, 10 for $1, 01 for 50c and 00 for 20c – the machine does not accept 10c or 5c coins). The new circuit is to check whether the machine receives more than twice as many $1 and $2 coins (combined) as $50c coins in order to determine whether the container for 50c coins needs to be decreased.

## PROBLEM 3:

The controller for a railroad crossing light receives input from a senor on the railway line some distance away. Once per five seconds, the pressure pad sends a digit to the controller; 1 if a train has been sensed passing and 0 if all is clear. Normally the controller is in the ‘inactive’ state, meaning that the boom gates are up and the alerts are turned off. When a 1 is received in such a state, the controller switches to a ‘warning’ state causing alerts to sound. Five seconds later the controller switches to a ‘closed’ state, meaning that the boom gate lowers while the alerts continue. After 30 seconds in the ‘closed state’ in which no 1s have been received, the controller returns to ‘inactive’ state, causing the boom gate to raise and the alarms to stop. Any 1s received during the ‘closed’ state cause the 30 second countdown to reset.

## PROBLEM 4:

A somewhat superstitious Linux programmer who has read too much Asimov has decided to run an experiment; he converted a number of his program source files into binary and attempted to determine whether these binary strings formed a regular language on the suspicion that coders are part of some experiment of either alien or divine nature. Failing in this, he decided to instead write some software that takes some source code as input and determines whether the accepted inputs to this source code (the ones that result in a successful computation) themselves form a regular language.

## PROBLEM 5:

The program YACC (Yet Another Compiler Compiler) receives an input a string of symbols that have been tokenized by a lexical analyser such as Lex. The syntax for the tokens is given below. As each token is received, YACC takes one of four actions; shift, reduce, accept or error. Shifting means storing the current token for later in a FILO (First-In Last-Out) structure. Reducing means combining the current input with the last information stored in the structure, in order to produce a new token. This token can then be stored in the structure or acted upon as if it were the new input.

The problem is to determine whether YACC would determine that the input meets the following syntax:

expr: term '+' expr

| term

;

term: '(' expr ')'

| term '!'

| number

;

number: digit (digit)\*

digit: 0|1|2|3|4|5|6|7|8|9

## PROBLEM 6:

For a ternary code (Σ = {a,b,c}), find all strings of the form { aibjck }, 0<*k*<*j*<*i*.

## PROBLEM 7:

As part of creating an AI programming machine, your team is tasked with writing a universal syntax checker. The program should load a set of syntax specifications for a programming languages (such as C or Java) and a program, and test to see whether the program meets the syntax specification.

## PROBLEM 8:

A conspiracy theorist has decided that Garfield cartoons form a secret code. The idea is that each panel in each cartoon is assigned a ‘character number’, which is the number of main characters (Garfield, Odie and Jon) who are clearly visible in that panel. The character numbers then form a string, which is then translated into binary and hence into English.

A linguistics expert has pointed out that if such a code were to exists, certain groupings would naturally occur more than any other. In particular, the sequence 011 would occur roughly three times as often as 003 and twice as often as 100.

The problem is to write a program that checks a string of ‘character numbers’ to see if the expected frequency is upheld.

## PROBLEM 9:

The language of binary strings that contain the sub-string 01*n*0 and the sub-string 10*n*1 where *n*>0.

## PROBLEM 10:

The language of binary strings that consist of 1*n*0*m*1*m*0*n* where *m*,*n*>0.

# Section 2 - Complexity

For each of the following problems, classify the problem into one or more time complexity categories (P, NP, NP-complete, NP-hard). If you cannot prove membership formally, give a convincing argument. If you need to make assumptions, state these clearly.

When using reduction to show a problem is NP-complete or NP-Hard, relate the problem to SAT, 3-SAT, or 3-COLOR.

## PROBLEM 1:

An agent from a security agency that must remained un-named requires that your group assist in checking the work of a second group, who are referred only as the S-team. The S-team has created a program that takes a graph that represents a fiber-optic network, and computes a minimal set of nodes (called the ‘nodes of interest’) in the network such that every link in the network is adjacent to at least one of the nodes. The idea is that the security agency can then install ‘appropriate measures’ in these nodes in order to have access to all traffic in the area.

Your team’s job is to write some software that takes the network (in the form of a graph) and the size of the answer from the S-team’s program and checks whether the network contains an appropriate set of ‘nodes of interest’ that is one smaller than the provided answer. For example, if the S-team’s answer has 300 nodes, your program would check if a 299 node answer exists.

Interestingly enough, the contract doesn’t specify that you actually need to give the 299 node answer, if it exists.

## PROBLEM 2:

A Sudoku buff has just come across Addoku for the first time. Sometimes referred to as “sum doku” or “killer Soduko”, this game involves a 3x3 grid of 3x3 cells much like Soduko. Unlike Soduko, there are no numbers pre-inserted in the squares, but instead each square is part of exactly one contiguous grouping of cells and each such grouping has its sum given. The buff had hear that Sudoku is NP-Hard (although he doesn’t know what it means) and asks you whether that is true and whether this new game might be even harder.

## PROBLEM 3:

A marketing ‘genius’ has the idea of buying all of the buildings in a block surrounded by only three roads that meet at certain angles, demolishing them, and replacing them with a gigantic, glass pyramids with triangular bases. You’ll get paid whether or not his idea works, and your task is to analyse Australia’s road transport network and look for all ‘triangle blocks’. You may assume that these can be located on the map as cliques of size 3.

## PROBLEM 4:

A computing lecturer who is keen on setting high standards asks his students to not only learn about Bayesian networks, but to actually write a program to infer the value of the hidden variables of any network given as input; formally the task is to compute the exact probabilistic inference of the Bayesian network.

Hint: You’ll probably have to do some research for this one.

## PROBLEM 5:

The nation’s security forces are hunting a mad bomber, who has sworn to set off the least amount of bombs, to do the maximum amount of damage to Australia’s computer communication network (by cutting as many communication paths as possible) after not being able to get the desired hat for his Team Fortress character. You have been given a graph of the Australian copper and fibre network (which are the known targets of the bomber) and the way in which they connect to each other, in order to determine where he is most likely to strike. It is not thought that he will attempt to strike any of the nodes, since they are currently being protected.

Hint: Consider various Cut problems.

## ADDITIONAL - PROBLEM 6:

The problem SAT, but with the added complication that each variable used must be quantified at the beginning of the formula by either ∀ or ∃.

**Note:** This question is here to teach you how to look questions up difficult questions. I don’t expect a proof for this one; an argument (possibly with references) will do. If you manage to prove the classification for this one on your own you probably deserve 110% in this unit.