CIT 596 Homework 6

Steven Tomcavage
stomcava@seas.upenn.edu

April 17, 2011

1 Exercise 5.1

Show that EQ_{CFG} is undecidable.

Proof.

1. Assume EQ_{CFG} is decidable.
2. Let S be a CFG and T be a CFG that accepts all possible strings from the tokens used by S.
3. To determine if $S = T$ requires deciding ALL_{CFG}, where ALL_{CFG} accepts when S generates all possible strings from its tokens.
4. By Theorem 5.13 in Sipser, ALL_{CFG} is undecidable, and therefore, EQ_{CFG} is undecidable. $\square$

2 Exercise 5.3

Find a match in the following instance of the Post Correspondance Problem.

$$\left\{ \left[\frac{ab}{abab}\right], \left[\frac{b}{a}\right], \left[\frac{aba}{b}\right], \left[\frac{aa}{a}\right] \right\}$$

Here a Prolog program to find a solution to a given Post Correspondance problem:

```
/*
    Program: postCorrespondance.pl
Description: Finds a solution to the Post Correspondance Problem, if a solution
             is available. Prints the solution as a list of item positions.
             Input the problem as a list of tuples where each tuple has the
             top value followed by the bottom value.
*/

% Try to find a one-item solution, if that doesn't work, recursively expand
% possible solution size until a solution is found.
solve(Problem, Solution)    :- solve(Problem, Solution, 1).
solve(Problem, Solution, N) :- get_solution(Problem, ("",""), Solution, N).
solve(Problem, Solution, N) :- M is N + 1, solve(Problem, Solution, M).

% Using an item as the start point, recursively go through the problem space
% and try to find a combination of items that are a solution.
get_solution(_, ("",""), [], 0).
```

```
get_solution(Problem, Answer, [SolutionI|SolutionRest], N) :-
    N > 0,
    nth1(SolutionI, Problem, Item),
    append_top_bottom(Answer, Item, AnswerAppended),
    check_answer(AnswerAppended, AnswerChecked),
    M is N - 1,
    get_solution(Problem, AnswerChecked, SolutionRest, M).

% Append the given item to the top and bottom of the current answer
append_top_bottom((Top1, Bottom1), (Top2, Bottom2), (Top3, Bottom3)) :-
    append(Top1, Top2, Top3),
    append(Bottom1, Bottom2, Bottom3).

% Check for match in top and bottom of all positions in the current answer
check_answer(([], B), ([], B)) :- !.
check_answer((A, []), (A, [])).
check_answer(([X|A], [X|B]), (Ao, Bo)) :- check_answer((A, B), (Ao, Bo)).
```

Running this program under SWI-Prolog on the given problem yields the following solutions:

```
pdt_reload('c:/dropbox/usr/mcit/prolog/postcorrespondance.pl').
|    c:/dropbox/usr/mcit/prolog/postcorrespondance.pl
% c:/dropbox/usr/mcit/prolog/postcorrespondance compiled 0.00 sec, 2,872 bytes
true.

17 ?- solve([("ab", "abab"), ("b", "a"), ("aba", "b"), ("aa", "a")], Solution).
Solution = [4, 4, 2, 1] ;
;
Solution = [1, 1, 3, 2, 2, 4, 4] .
```

So the solutions are:

$$\left\{\left[\frac{aa}{a}\right], \left[\frac{aa}{a}\right], \left[\frac{b}{a}\right], \left[\frac{ab}{abab}\right]\right\}$$

$$\left\{\left[\frac{ab}{abab}\right], \left[\frac{ab}{abab}\right], \left[\frac{aba}{b}\right], \left[\frac{b}{a}\right], \left[\frac{b}{a}\right] \left[\frac{aa}{a}\right], \left[\frac{aa}{a}\right]\right\}$$

3 Exercise 5.4

If $A \leq_m B$ and B is a regular language, does that imply that A is a regular language? Why or why not?

No, it does not imply that A is a regular language. B is the set of strings generated by $f(w)$ where f is a function and w is a string generated by A. The mapping is not necessarily a one-to-one relationship between w and $f(w)$. Thus, there can exist a function, f, that maps from a non-regular language to a regular language.

4 Exercise 5.5

Show that A_{TM} is not mapping reducible to E_{TM}.

Proof.

1. Assume that $A_{TM} \leq_m E_{TM}$.
2. Let M be the TM for $\langle E_{TM}, w \rangle$.
3. Build a TM, N, so that on w, if M rejects, then A accepts, and if M accepts, then A rejects.
4. But, because E_{TM} is undecidable, then A_{TM} must be undecidable.
5. Therefore, A_{TM} is not mapping reducible to E_{TM}. □

5 Exercise 5.6

Show that leq_m is a transitive relation.

Proof.

1. Let $A \leq_m B$. This means there is a function f_1 that accepts a w in A and the output of f_1 is in B.
2. Let $B \leq_m C$. This means there is a function f_2 that accepts a w in B and the output of f_2 is in C.
3. Thus, since B is a subset of A and C is a subset of B, then C must be a subset of A and there must be a function f_3 that accepts a w in A and the output of f_3 is in C.
4. Therefore, leq_m is a transitive relation. □

6 Exercise 5.7

Show that if A is Turing-recognizable and $A \leq_m \not{A}$, then A is decidable.

Proof.

1. Let B be the language $\not{A}$.
2. Assume that B is Turing-recognizable.
3. Since A is mapping reducable to B, then given a w in the laguage of A, if B accepts w, A will reject w, and if B rejects w, then A will accept w.
4. Thus, B is a co-Turing recognizer for A.
5. Therefore, A is Turing-recognizable and B is co-Turing recognizable for A, then A is decidable. □

7 Exercise 5.9

Let $T = \{\langle M \rangle \mid M$ is a TM that accepts w^R whenever it accepts $w\}$. Show that T is undecidable.

This is a restatement of the halting problem. Since we've already proved that the halting problem is undecidable, then this problem is undecidable.

8 Exercise 5.17

Show that the Post Correspondence problem is decidable over the unary alphabet $\Sigma = \{1\}$.

Proof.

1. Given that the Post Corrspondence problem requires each charcter in the top and bottom strings to match and for the top and bottom strings to be the same length.
2. Using a unary alphabet, each character on the top is guaranteed to match its corresponding character on the bottom.
3. Determining whether the items can be arranged so the top and bottom lengths are the same is a matter of trying all possible combinations of items without repeating items. If a combination is found where the length of the top matches the length of the bottom, then the machine accepts. Otherwise the machine rejects.
4. Therefore the Post Correspondence problem is decidable over the unary alphabet. $\square$

9 Exercise 5.19

In the silly Post Correspondence Problem, SPCP, in each pair the top string has the same length as the bottom string. Show that the SPCP is decidable.

Proof.

1. Given that the Post Corrspondence problem requires each charcter in the top and bottom strings to match and for the top and bottom strings to be the same length.
2. If all items have the same length between the top and bottom, then the machine needs to search for any item where the characters on the top match the characters on the bottom. If one item like that is found, then the machine accepts. Otherwise the machine will never be able to build a correct string, and it rejects.
3. Therefore the Silly Post Correspondence problem is decidable. $\square$

10 Exercise 5.28

Let P be a language consisting of Turing machine descriptions where P fulfills two conditions. First P is nontrivial - it contains some, but not all, TM descriptions. Second, P is a property of the TM's language - whenever $L(M_1) = L(M_2)$, we have $\langle M_1 \rangle \in P$ iif $\langle M_2 \rangle \in P$. Here, $M1$ and $M2$ are any TMs. Prove that P is an undecidable language.

TODO

11 Exercise 5.29

Show that both conditions in Exercise 5.28 are necessary for proving that P is undecidable.

TODO

12 Exercise 6.1

Give an example in the spirit of the recursion theorem of a program in a real programming language that prints itself out.

Here is a program written in Clojure which prints itself out. To run it (assuming the code is in a file named quine.clj and Clojure is installed on the machine), start the Clojure REPL with the command `java -cp clojure.jar clojure.main`. Then load and execute the file with the command `(load-file "./quine.clj")`.

```
;     Program: quine.clj
; Description: A program that outputs a copy of itself. The program consists of
;              an anonymous function that takes the string "w" and prints
;              "((w (quote w)))". The argument "w" to the anonymous function
;              is the full text of the anonymous function itself as a quoted
;              string. This all works because code is data is code in Clojure.

((fn [w] (list w (list (quote quote) w)))
  (quote (fn [w] (list w (list (quote quote) w)))))
```

13 Exercise 6.3

Show that if $A \leq_T B$ and $B \leq_T C$ then $A \leq_T C$.

Proof.

1. The statement $A \leq_T B$ means that A is a decider if there exists some language B that A reduces to.
2. Likewise, $B \leq_T C$ means that B is a decider if there exists some language C that B reduces to.
3. Assuming from the first statement that a language B exists, and from the second statement that B is a decider, then $A \leq_m B \leq_T C$.
4. Since Exercise 5.6 shows that $\leq_m$ is a transitive relation, then $A \leq_m B \leq_T C$ means that $A \leq_T C$.
5. Therefore, if $A \leq_T B$ and $B \leq_T C$ then $A \leq_T C$. $\square$

14 Exercise 6.5

Is the statement $\exists x \forall y[x + y = y]$ a member of $\text{Th}(N, +)$? Why or why not? What about $\exists x \forall y[x + y = x]$?

Assuming that $0 \in N$, then the first statement, $\exists x \forall y[x + y = y]$, is a member of $\text{Th}(N, +)$ because it is true when $x = 0$. So for all y, $0 + y = y$.

The second statement, $\exists x \forall y[x+y = x]$, is not a member of $\text{Th}(N, +)$, because it's only true where $y = 0$. In that case, $x + 0 = x$, but in all other cases in N, it is false.

15 Exercise 6.6

Describe two different Turing machines, M and N, that, when started on any input, M outputs $\langle N \rangle$ and N outputs $\langle M \rangle$.

The machine M contains the description of the machine N. For each configuration in M, the machine reads a step from the description of N, outputs that step, and moves right to the next step in the description of N.

Likewise, the machine N contains a description of M and opterates on that description in the same manner as M opperates on the description it contains.

16 Exercise 6.10

Give a model of the sentence provided in the text (omitted).

The model is $(\{\mathbf{C}, \mathbf{R}\}, =)$.

17 Exercise 6.17

Let A and B be two disjoint languages. Say that language C seperates A and B if $A \subseteq C$ and $B \not\subseteq C$. Describe two Turing-recognizable languages that aren't separated by any decidable language.

This problem is searching for a C that is undecidable where $A \leq_T C$ and $B \leq_T \not C$. I'm not sure where I can find such a C.

18 Exercise 6.25

Show that for any c, some strings x and y exist where $K(xy) > K(x) + K(y) + c$.

To show this, set $x \geq 3c$ and $y \geq 3c$. Thus, $K(9c) > K(3c) + K(3c) + K(c)$.

19 Bonus Problem

Provide a program that prints itself out.

See Exercise 6.1 for a listing of Clojure code that prints itself out when run.