CS350 – Problems and Languages

Purpose: There is a direct relationship between two big concepts of *problems* and *languages*. A *language* is the set of *problem instances* which have the answer "yes". So now these languages can be described as *decidable* if there is a membership algorithm or *recognizable* if there is a program which accepts the strings in the language (and may reject or be undefined for strings not in the language).

The purpose of this module is to explore the relationship between *problems* and *languages* as well as the differences between *decidable* and *recognizable* languages

Knowledge: This module will help you become familiar with the following content knowledge:

- How to transform a problem into a language
- How to determine whether a language is recursive (decidable) and/or recursively enumerable (recognizable)
- How to determine if a class of languages is *closed* under an operation

Activity: With your group perform the following tasks and answer the questions. You will be reporting your answers back to the class in 60 minutes.

1. A *language* in this course is going to be a set of strings. We also specify an *alphabet* which is a set of characters that can appear in the strings.

   Suppose our alphabet is $\Sigma = \{a, b, c\}$.
   Our language $L_1 = \{w \mid w \text{ contains more b's than a's}\}$

   Determine which of these strings are contained in the language L_1 and which are not.

   (a) "abbcbba"
   (b) "cab"
   (c) "bba"
   (d) "bccccbcccc"
   (e) "bacb"

Suppose our language is $L_2 = \{a^n \mid n \text{ is even}\}$ where a^n is notation indicating a string of n a's.

Determine which of these strings are contained in the language L_2 and which are not.

(a) ϵ (which is notation for the empty string)

(b) "aa"

(c) "aabb"

(d) "aaaa"

2. Since languages are sets, we can use the regular set operation on them like union, intersection, etc.

Union $L_1 \cup L_2$ is the set containing all elements that appear either in L_1 or in L_2

Intersecion $L_1 \cap L_2$ is the set containing all elements that appear in both L_1 and L_2

Complement $\overline{L}$ is the set containing all strings that we can make from the alphabet that are not contained in L

Concatenation $L_1 L_2$ is the set containing all the strings that can be formed from a string in L_1 followed by a string in L_2

Kleene star L^* is all the strings that can be formed from concatenating strings contained in L together.

Suppose alphabet $\Sigma = \{a, b\}$, $L_1 = \{a^n \mid n \geq 0\}$ and $L_2 = \{b^n \mid n \geq 0\}$
Describe the set $L_1 \cup L_2$
Describe the set $L_1 \cap L_2$
Describe the set $L_1 L_2$

3. The textbook says that a language is *decidable* if we have a membership algorithm that can tell us whether or not a string is in the language. Many other texts use the term *recursive* language instead and we will use the terms decidable and recursive interchangeably.

   Mark each of the following statements as either true or false pertaining to a decidable language L.

   (a) There is a Program P such that for all $I \in L$, P(I)="yes" or P(I)="no" and P(I) is never undefined.

   (b) There is a decision problem D such that $L = \{I \mid D(I) = "yes"\}$

   (c) There is a program which enumerates all the strings in L (although the program might run forever if L is an infinite set)

4. The textbook says that a language is *recognizable* if there is a Program P such that for all $I \in L$, P(I)="yes" and for all $I \notin L$, P(I)="no" or P(I) is undefined. Many other texts use the term *recursively enumerable* instead and we will use the terms recognizable and recursively enumerable interchangeably.

   Mark each of the following statements as either true or false pertaining to a recognizable language.

   (a) There is a membership algorithm that can tell us if a string is in the language or not

   (b) There is a decision problem D such that $L = \{I \mid D(I) = "yes"\}$

   (c) There is a program which enumerates all the strings in L (although the program might run forever if L is an infinite set)

5. We say a set is *closed under an operation* if the operation performed on members of the set results in a value that is also contained in the set.

   (a) Is the set of all integers closed under addition?

   (b) Is the set of all integers closed under division?

   (c) Is the set of all even numbers closed under addition?

   (d) Is the set all all odd numbers closed under addition?

6. We will prove that the set of all decidable languages is closed under the union operation. In other words, if we take the union of two decidable languages we will get another decidable language. The following is an attempt at proving that. Discuss whether you think this proof is correct or incorrect.

   Given two decidable languages A and B, and programs memberA and memberB that decide them, write a program memberUnionAB to decide whether a string is in the union.

```
def memberUnionAB(inString):
    if memberA(inString)=="yes":
       return "yes"
    else:
       return memberB(inString)
```

7. We will prove that the set of all recognizable languages is closed under the union operation. In other words, if we take the union of two recognizable languages we will get another recognizable language. The following is an attempt at proving that. Discuss whether you think this proof is correct or incorrect.

   Given two recognizable languages A and B, and programs recognizeA and recognizeB that recognize them, write a program recognizeUnionAB to determine whether a string is in the union. Remember that a recognizer will return ”yes” if the string is in the language but if the string is not in the language it might return ”no” or it might be undefined.

```
def recognizeUnionAB(inString):
    if recognizeA(inString)=="yes":
       return "yes"
    else:
       return recognizeB(inString)
```

Complete the following assignments for grading. Each should be done individually but you may consult with a classmate to discuss strategies.

Assignment 1:
Complete exercises 4.20 on p70 of the text.

Criteria for Success: You have explanations for parts a) through d) and an example for part e).

Assignment 2:
Suppose that L is a finite language. Show that L^* is recursively enumerable.

Criteria for Success: One way to show that a language is recursively enumerable is to describe an enumeration procedure for it. Recall that *recursively enumerable* and *recognizable* mean the same thing.

Assignment 3:
Complete exercises 4.25 a) and 4.26 c) on p70 of the text.

Criteria for Success: You always show a closure property by showing that after performing the operation (complement, in this case) you still have a language of the same kind. Recall that *recursive* and *decidable* mean the same thing. Likewise *recursively enumerable* and *recognizable* mean the same thing.

Assignment 4:
Suppose we have a procedure for enumerating the language L in *proper order*, which is similar to dictionary ordering but we order first by the length of the strings. Show that this means that L is recursive.

Criteria for Success: One way to show that a language is recursive is to give a membership algorithm. Describe such an algorithm.

Submit your written answers in Canvas for grading or turn in on paper assignment if your prefer.