ECS120 Introduction to the Theory of Computation
Fall Quarter 2007

Discussion Notes
Wednesday, October 16, 2007

Decision Procedures

Decision procedures are basically algorithms which provide with certainty a yes or no answer. When giving us a decision procedure make sure to (1) specify your algorithm and (2) explicitly state when your algorithm outputs a `YES` answer and when it outputs a `NO` answer.

There are two useful techniques for creating decision procedures. If you are lucky, you can break down the problem you are solving into sub-problems which already have solutions. You can then reuse those solutions in solving the main problem.

Otherwise, you need to construct an algorithm which uses what you know or can assume. If we know a language L is regular, we know it can be represented by a DFA, NFA, or regular expression. Therefore we can show how to solve our problem given a DFA, NFA, and regular expression. The catch is that we should try and provide a polynomial-time algorithm for each case.

Example 1

Let L_1 and L_2 be regular languages. Describe a decision procedure that determines if L_1 and L_2 have at least 1 string in common.

. . . .

First, we should try and break down the problem. How do we know when L_1 and L_2 share at least 1 string? This is true whenever $L_1 \cap L_2 \neq \emptyset$.

Therefore, we want to build an algorithm:

$$\textsc{Similar}(L_1, L_2) = \begin{cases} \texttt{YES} & \text{if } L_1 \cap L_2 \neq \emptyset \\ \texttt{NO} & \text{otherwise} \end{cases}$$

So now the problem has become determining if $L_1 \cap L_2 \neq \emptyset$. Notice that now we can break this into sub-problems. We need to find $L_1 \cap L_2 = L_3$, and then determine if $L_3 \neq \emptyset$.

We already know how to do $L_1 \cap L_2 = L_3$. For now, lets assume we have a procedure $\textsc{Intersect}(L_1, L_2)$.

We know that L_3 is regular. Therefore, we can also use the decision procedure discussed in class for determining if $L_3 = \emptyset$. Lets call this procedure $\textsc{IsEmpty}(L)$.

Now we can put the two together, and define our $\textsc{Similar}$ algorithm:

$\text{SIMILAR}(L_1, L_2)$:

1. Find $\text{INTERSECT}(L_1, L_2) = L_3$
2. Run $\text{ISEMPTY}(L_3)$:
   - If $\text{ISEMPTY}(L_3) =$ `YES`, then output `NO`.
   - Else, output `YES`.

Example 2

Let L_1 and L_2 be regular languages. Describe a decision procedure that determines if w is not in L_1 or in L_2.

. . . .

Essentially, this is determining if $w \notin L_1 \cup L_2$. We know how to take the union of two languages, and already discussed a decision procedure for $w \in L$ in class. Therefore we get:

$\text{EXAMPLE2}(L_1, L_2, w)$:

1. Find $\text{UNION}(L_1, L_2) = L_3$
2. Run $\text{ISMEMBER}(L_3, w)$:
   - If $\text{ISMEMBER}(L_3, w) =$ `YES`, then output `NO`.
   - Else, output `YES`.