EECS 2001N: Introduction to the Theory of Computation
Tutorial 1, Jan 10, 2:30 pm
Problems

1. In each part below, a recursive definition is given of a subset of $\{a, b\}^*$. Give a simple non-recursive definition in each case. Assume that each definition includes an implicit last statement: "Nothing is in L unless it can be obtained by the previous statements."

   (a) $a \in L$; for any $x \in L$, xa, xb are in L.

   Solution: This is the set of all strings starting with a. An informal justification is this: a is in the language and since the other 2 rules add a, b to the right side (or end) of a string, the rules can be applied to generate all strings starting with a.

   (b) $a \in L$; for any $x \in L$, bx, xb are in L.

   Solution: This is the set of all strings containing exactly one a. An informal justification is this: a is in the language and since the other 2 rules add b to the left and right side of a string, the rules can be applied to generate all strings with a single a. Alternatively we can say that the set consists of all strings of the form $b^m ab^n$ where $m, n \in \mathbb{Z}$ and $m, n \geq 0$.

   (c) $a \in L$; for any $x \in L$, ax, xb are in L.

   Solution: This set consists of all strings of the form $a^m ab^n$ where $m, n \in \mathbb{Z}$ and $m, n \geq 0$. An informal justification is this: a is in the language and since the other 2 rules add a to the left and b to the right side of a string. Alternatively we can say that the set consists of all strings $a^m b^n$ where $m, n \in \mathbb{Z}$ and $m \geq 1, n \geq 0$.

   (d) $a \in L$; for any $x \in L$, ax, bx, xb are in L.

   Solution: Note that the set of rules are the union of rules for parts (b), (c), but the language produced is not the union of the languages in parts (b), (c)! This set consists of all strings of the form sab^n where $n \in \mathbb{Z}$ and $n \geq 0$, and s is any string over $\{a, b\}$. An informal justification is this: a is in the language and since the first 2 rules add a, b to the left and can thus produce any string s to left of a. The third rule adds zero or more b's (but not a's) to the right of a.