Countable & Countably Infinite Sets

Countability is defined as the one-to-one correspondence between some set and the natural numbers.

Let $\omega = \{0, 1, 2, 3, \ldots\}$ be the set of all natural numbers.

Definition: We say that some set X is ***countable*** iff there exists a (total) injective function $f : X \to \omega$. If the function f is also bijective then we say that X is ***countably infinite***.

Example: The set $A = \{a, b, c\}$ is countable, since we can construct an injective function $f : A \to \omega$ such that $a \mapsto 0, b \mapsto 1, c \mapsto 2$.

Example: The set of all non-zero natural numbers $\mathcal{N} = \omega - \{0\}$ is countably infinite, since we can construct a bijective function $f : \mathcal{N} \to \omega$ where $f(n) = n - 1$, with $n \in \mathcal{N}$.

NOTE: The real numbers are uncountable because in any interval between two real numbers there exists an infinity of real numbers. (Therefore, it is impossible to bring the real numbers into a one-to-one correspondence with the natural numbers, all of the natural numbers would be used up in trying to count the number of elements in the first infinitesimal interval on the real number line.)

Constructing Sets

Countable sets are usually constructed by enumerating their members.

We can use this to construct languages.

Example: The language $L = \{aa, ab, ba, bb \mid a, b \in \Sigma\}$.

Example: Let $\Sigma = \{1, 0\}$, the language $L = \{s \mid |s| \leq 3$ and $s_i \in \Sigma\}$.

Constructing Sets

Since it is impossible to enumerate countably infinite sets, languages that are countably infinite are usually describe via a ***recursive definition***.

Example: Let $\Sigma = \{a\}$, then $L = \{s \mid s$ begins with an a and has an even length$\}$. Formally we can describe L as follows:

Basis: $aa \in L$.

Recursive step: if $u \in L$, then $uaa \in L$.

Closure: $u \in L$ only if it can be obtained from the basis by a finite number of applications of the recursive step.

NOTE: L is countably infinite; $f: L \rightarrow \omega$ where $f(l) = |l|/2 - 1$, with $l \in L$.

NOTE: Even though the set L is infinite, each member of L is considered finite, in the sense that each member is constructed by a finite number of applications of the recursive step starting at the basis.

NOTE: L is considered a well-founded set: no infinite chains from an element to the basis.

Constructing Sets

We can use regular expressions as a short hand for describing countably infinite regular languages.

Example: Let $\Sigma = \{a\}$ and $L = \{s \mid s$ begins with an a and has an even length$\}$, then

$$L = aa(aa)^*$$

We can use context-free grammars as a short hand for describing countably infinite context-free languages.

Example: Let $\Sigma = \{a, b\}$ and
$L = \{s \mid s$ has one b and has a pre- and postfix of equal number of a's$\}$, then

$$\begin{aligned} A &\rightarrow aAa \\ A &\rightarrow b \end{aligned}$$

Infinite Languages & FAs

Lemma (Infinite Languages): A FA recognizing a countably infinite language must have cycles.

Proof: Let k be the number of states in some FA M recognizing the countably infinite language L. Since L is countably infinite, we can always construct a string s with $|s| \geq k$ using the recursive step an appropriate number of times. Assume that the machine M can accept this string s without cycles. But a machine with k states can at most recognize a string of length $k - 1$ without repeating a state. This is a contradiction, therefore the FA M has at least one cycle.

NOTE: This gives rise to the pumping lemma.

Finite Languages & FAs

Lemma (Finite Languages): A FA recognizing a countable language must not have cycles.

Proof: Let M be a FA recognizing the countable language L. Also assume that M has a cycle. But notice if we use the machine as a generator for the strings in the language then we can use the cycle in the machine as a recursive step in generating strings in such a way that the resulting language is countably infinite; a contradiction. Therefore, M cannot have cycles.

The Pumping Lemma

Thereom (The Pumping Lemma): Let L be a regular language accepted by DFA M with p states. Then given any string s of length at least p, it maybe divided into three pieces, $s = xyz$, such that

1. for each $i \geq 0$, $xy^iz \in A$,
2. $\mid y \mid > 0$, and
3. $\mid xy \mid \leq p$.

Here, p is called the *pumping length*.

Proof: We break the proof into two parts.

(a) If L is a regular language and countably infinite, then the pumping lemma follows from our lemma on infinite languages.

(b) If L is a countable regular language then the pumping lemma holds vacuously. To see this we need to realize that implication is true even if the antecedent is false; assume $P = false$ and $Q = anything$ then $(P \rightarrow Q) = true$. We now show that the antecedent for the pumping lemma is false for countable regular languages. From our lemma on finite languages we know that a machine recognizing a finite language cannot have cycles, therefore no string that M accepts can be longer than $p - 1$, thus we will be never able to fulfill the antecedent $|s| \geq p$.

□