CS331 More on Languages

July 25, 2001 **Shariq Rizvi**

1 The Jargon

We will use the following terminology:

1. Finite Set: Cardinality of the set is an element of N, the set of natural numbers. Given any finite set, there exists a natural no. that is larger than the cardinality of the set. For example, the set {4,5,7,8} is finite.

2. Countably Infinite Set : There exists a bijection from the given set to the set N of natural nos. There is no natural number larger than the cardinality of the set.

3. Uncountable Set : No. of elements is more than in the set of natural nos.

We have already seen what alphabets, strings and languages are. To recapitulate, an alphabet is a finite set of symbols. A string is a concatenation of symbols from a given alphabet. A language is a set of strings. A point to note is that a string may be finite or countably infinite in length. Any concatenation of the symbols in the alphabet will be called a string. So, for the alphabet $\{a, b\}$, the following are some strings - $a, aba, bbb, abbaaab, b, abababab \ldots$

It may be observed that the last string listed above is countably infinitely long and is thus not finite in length.

A natural question that arises is that given an alphabet Σ (which by definition is a finite set of symbols), is the set S of strings formed from symbols in Σ finite, countably infinite or uncountable? We saw some intuitive arguments for resolving this question in the previous class. Let's try to do some precise mathematical reasoning now.

S definitely cannot be finite. Even for a single symbol alphabet, say $\{a\}$, possible strings are : $a, aa, aaa, aaaa, \ldots$ Surely, the number of such strings (and hence the cardinality of S) cannot be finite.

Will S be countably infinite? The answer is *Yes* for a one symbol alphabet and *No* for a larger alphabet. In the former case, say the alphabet is $\{a\}$. Then S will be : $\{a, aa, aaa, \ldots\} \cup \{aaa\ldots\}$. The second set contains the string which has countably infinite number of a's. There is only one such string. Thus S is countable with a bijection between itself and $\mathbf{N}$ being defined as under :

$$f(s) = \begin{cases} 1 & if\ s\ =\ aaaa\ldots \\ length(s) + 1 & otherwise \end{cases}$$

Note that the string a will be mapped to 2, aa to 3 and so on. The infinite string $aaa\ldots$ will be mapped to 1. Also, given any natural number, it has a pre-image in S. Thus S is countably infinite.

Now consider the alphabet $\{a, b\}$. The set of strings is $S = \{a, b, aa, ba, abab, \ldots\}$ $\cup$ $\{aaa\ldots, bbb\ldots, abab\ldots, \ldots\}$. The first is the set of all finite strings and the second is the set of all countably infinitely long strings. In this case, the set S can be shown to be uncountable as follows.

Consider only the second constituent set (that of infinitely long strings). Suppose this set is countable. Then we can assign to each member of this set a unique natural number. Sensing the possibility of using the Diagonalisation argument, we now construct an infinite two-dimensional matrix as under, having in the i^{th} row, the infinite string mapped to i.

$$\begin{bmatrix} \mathbf{a} & a & b & a & a & b & . & . & . \\ b & \mathbf{a} & b & a & b & a & . & . & . \\ b & b & \mathbf{b} & b & b & b & . & . & . \\ . & . & . & \mathbf{a} & . & . & . & . & . \\ . & . & . & . & \mathbf{b} & . & . & . & . \end{bmatrix}$$

The string on the diagonal ($aabab\ldots$) when complemented (replace each 'a' by 'b' and vice versa) gives a string $bbaba\ldots$ which must also occur in the matrix (as it is an infinite concatenation of symbols from our alphabet). However, this string cannot be the i^{th} row of the matrix for any i, as it will conflict with the i^{th} character of this row ($i \in$ N). Thus, the string cannot be present in the matrix. So, we end up addled by a contradiction.

Thus, the set Σ^ω of infinite strings is **not** countably infinite but uncountable. S being a superset of this, must also be uncountable.

Gist : The set S of finite and countably infinitely long strings formed from an alphabet Σ with at least 2 symbols is uncountable.

We have seen earlier that a language is merely a collection of strings. A language thus, is just a subset of S, the set of strings. For example, for the alphabet $\{a, b\}$, the sets $\{a, ab, baa\}$ and $\{abb, b, e, ababab\dots, bbbaaaa\dots\}$ are two of various languages possible. Note that a language may also contain strings of S that are countably infinitely long.

Given a particular alphabet, what is the number of languages possible? The answer is - uncountably infinite. Reasoning - As proved above, S is either countably infinite or uncountable, depending on whether Σ has one or more symbols. The powerset of S, i.e., the set of all languages, is thus uncountable.

Gist : Given any non-empty alphabet Σ, uncountable languages originate from it.

2 Our Focus : Σ^*

It will be of interest to see how restricting ourselves to just finite strings (no uppper bound on length though) affects the study. So, we partition S, the set of all strings as follows: $S = \Sigma^* \cup \Sigma^\omega$, where Σ^* is the set of all finite strings and Σ^ω is the set of all countably infinitely long strings. Here, we consider Σ^* to also include the special string of length 0, denoted ϵ. This is a special string that when concatenated to any string w leaves it unchanged. Similarly, any string w concatenated to ϵ gives the same string w. In other words, $w.\epsilon = \epsilon.w = w$ for any string w (including ϵ).
For example, $ababab$ and $\epsilon \in \Sigma^*$ while $ababab\dots \in \Sigma^\omega$.

We are interested only in the set of finite strings Σ^*. Is this set countable or uncountable?
Proving the countability of Σ^* : Let our alphabet be $\{a_0, a_1, ..., a_n\}$. Then the following function f is an injection from N to Σ^* :

$f(i) = a_0a_0a_0\dots$(concatenated i times)

and g defined below is an injection from Σ^* to N:

$g(a_{l_1}a_{l_2}\ldots a_{l_m}) = 2^{l_1}.3^{l_2}.5^{l_3}\ldots p_m^{l_m}$ for non empty strings, and
$g(\epsilon) = 1$.

In the definition of g above, the bases on the RHS are the consecutive prime numbers (p_m being the m^{th} prime). As the string is finite, RHS can be evaluated for any element in Σ^*. The injection of g follows from the unique prime factorization property of natural numbers. As we have an injection both ways, we also have a bijection given by the Schroeder-Bernstein theorem.

Gist : Σ^* is countably infinite for any non-empty alphabet Σ.

2.1 Languages from Σ^*

Now that the set of strings Σ^* is known to be countably infinite, what impact is made on $\mathcal{L}$, the set of all languages formed from finite length strings? Is $\mathcal{L}$ countable?
Since $\mathcal{L}$ is nothing but the power set of Σ^*, its cardinality is the same as that of the power set of N ($|\Sigma^*| = |N|$). Thus, $\mathcal{L}$ is uncountable.

3 Where the computer fails....

Can a $C++$ program (a finite set of C++ instructions) generate all the languages possible for an alphabet Σ even if we give it all the time it needed?

What does "all the time it needed" mean? : Can the program generate languages in such a way that any language is generated at some time (no matter how large that time is).

Answer : **No**

Argument 1 : If the program generated all languages in the above mentioned fashion, then we could index the languages in the order they were generated. First language mapped to 1, second to 2 and so on. Also, if as claimed, any language is listed at some time, then every language is mapped to a natural number. So, we have set up a bijection between $\mathcal{L}$ can $\mathbf{N}$. Which contradicts the uncountability of $\mathcal{L}$.

Argument 2 : The $C++$ alphabet is finite, so the number of finite strings ($C++$ programs) is at best countably infinite. Suppose a program does the job, then each language's generation can be represented by a smaller sub program of this program. So, again we have a bijection between $\mathcal{L}$ and a subset of all $C++$ programs. The latter is countable, while the former uncountable. So, we contradict ourselves again.

So, no computer (based on the prevalent model of computation) can list all such languages - a problem where computers fail!

4 Introduction to Finite Automata

A Turing machine represents the essence of a computer, as we know it today. It consists of a Finite State machine with a tape and a read/write head. In general the head can read from as well write to the tape, and can move either to the left or to the right on the tape. One of the most profound results of Computer Science is that this basic model represents the full power of the present paradigm of computation. We'll study this later in the course. For our present venture, we assume the following :

1. Head moves only to its right on the tape.
2. It can only perform the read operation.
3. Number of states is finite.
4. The tape is of finite length.

This model is called the **Finite Automata**. A finite automata (M) is completely specified by 5 parameters : set of states (Q), the alphabet of symbols (Σ), the state transition function ($\delta : Q \times \Sigma \longrightarrow Q$), initial state ($q_0 \in Q$), and the set of accepting or final states ($F \subseteq Q$). Thus, $M = (Q, \Sigma, \delta, q_0, F)$.