Mini-Quiz 9

1. Write your name:

2. (Rosen, Section 1.7, Problem 35) Show that if A is an uncountable set and $A \subseteq B$, then B is uncountable.

Massachusetts Institute of Technology
6.042J/18.062J: Mathematics for Computer Science
Professors David Karger and Nancy Lynch

Handout 33
April 7, 2000

Tutorial 9 Problems

Problem 1

(a) How many ways can $2n$ people be divided into n pairs?

Solution.

$(2n)!/n!2^n$.

Consider all the ways of ordering the $2n$ people in a line. There are $(2n)!$ ways of doing this. Now group them into pairs based on the linear order, i.e. group the first and second persons, group the third and fourth persons,..., group the $(2n-1)$th and $2n$th persons. Thus each linear order gives us a unique grouping into pairs. But observe that two different linear orders could give rise to the same grouping into pairs. So let us now count the number of linear orders that give rise to the same grouping into pairs. Given a grouping into pairs, to generate a linear order we need to order each pair (there are 2^n ways to do this) and then order the n pairs (there are $n!$ ways to do this). Hence $n!2^n$ linear orders give the same grouping into pairs. Thus the total number of ways $2n$ people can be divided into n pairs is $(2n)!/n!2^n$.

$\square$

(b) How many ways can you choose n out of $2n$ objects, given that n of the $2n$ objects are identical?

Solution.

The answer is 2^n, since you can pick any subset from the n nonidentical objects, and make up the rest with the identical ones. And the number of subsets of n different objects is 2^n.

$\square$

Problem 2 Calculate the number of ways to place 11 *indistinguishable* balls in four *distinguishable* boxes so that each box contains at least one, but no more than four, balls.

Solution.

First place one ball in each box. Then we must distribute 7 balls in four boxes in such a way that no more than three balls are in the same box.

We use inclusion/exclusion

Number of ways to place 7 balls in 4 boxes: $\binom{7+4-1}{7} = \binom{10}{7}$.

Number of ways to place 7 balls in 4 boxes such that there are four balls in a box = number of ways to choose box with four balls * number of ways to place remaining three balls in remaining three boxes = $4 \cdot \binom{5}{3}$.

Number of ways to place 7 balls in 4 boxes such that there are five balls in a box = $4 \cdot \binom{4}{2}$.

Number of ways to place 7 balls in 4 boxes such that there are six balls in a box = $4 \cdot \binom{3}{1}$.

Number of ways to place 7 balls in 4 boxes such there are seven balls in a box = 4.

So the total number of ways is

$\binom{10}{7} - 4\left(\binom{5}{3} + \binom{4}{2} + \binom{3}{1} + 1\right) = 40$

□

Problem 3 In this problem we will prove the remarkable fact that there exist mathematical functions that computers, no matter how powerful, simply cannot compute. We will do this through countability arguments.

(a) Show that the set of finite length binary strings is countable.

Solution.

Let B_n denote the set of binary strings of length n. This set has exactly 2^n elements. Therefore the set of finite length binary strings $B = \bigcup_{n \in \mathbb{N}} B_n$ is a union of countable number of finite sets. We know from Lecture 13 that such union is countable.

Namely, one can easily list all elements of B by first listing the elements of B_1, then the elements of B_2, then the elements of B_3, etc,. We can do this since each set B_i has a finite number of elements.

□

(b) From your answer to the previous part, what can you conclude about the countability of the set of all computer programs?

Solution.

Since every computer program can be represented as a finite string of bits (e.g. machine language code), it follows that the set of all computer programs is countable.

Namely, the mapping which represents computer programs as finite strings of bits is an injection. Hence there is a surjection from the set of finite strings of bits to the set of computer programs. From (a) we know that there is a surjection from the set of natural numbers to the set of finite strings of bits. By transitivity of surjective mapping, there is a surjection between natural numbers and computer programs. Hence the set of all comptuer programs is countable.

□

(c) Show that the set of *infinite* length binary strings is uncountable.

Solution.

We construct a bijection from the set of infinite length binary strings to the power set of the natural numbers. Since the power set of the natural numbers is uncountable, it will then follow that the set of all infinite length binary strings is uncountable.

We need to associate a subset of the naturals with each infinite length binary string. Let $b = b_0b_1b_2b_3\dots$ be an infinite length binary string (here each $b_i \in \{0,1\}$). We associate with this string the set $f(b) = S = \{i \mid b_i = 1\}$. That is, we include i in $S = f(b)$ if and only if the i-th bit position (from the left) of b is 1. Such mapping f is a bijection: It is an injection because if two infinite length binary strings are different, then clearly they map to different subsets. Furthermore, it is a surjection because for every subset $S \subset \mathbb{N}$, there is a binary string b s.t. $f(b) = S$, namely $b = b_0b_1b_2b_3\dots$ where b_i is defined as 0 if $i \notin S$ and 1 if $i \in S$.

Alternative Solution: One can also show that the set B' is uncountable directly by a diagonalization argument.

Assume B' is countable and let $b^{(1)}, b^{(2)}, b^{(3)}, b^{(4)}, ...$ be a list of all elements of B'. By diagonalization, we will construct an element $b = b_0b_1b_2b_3... \in B'$ s.t. $b \neq b^{(i)}$ for all $i \in \mathbb{N}$ as follows: j-th bit of b is defined as the opposite of the j-th bit of string $b^{(j)}$. Obviously, so defined b is not equal to any $b^{(i)}$ from the list because for every i, the i-th bit of $b^{(i)}$ is different than the i-th bit of b. Therefore, the list $b^{(1)}, b^{(2)}, b^{(3)}, b^{(4)}, ...$ is *not* a list of all elements in B', and thus the assumption that B' is countable leads to a contradiction.

$\square$

(d) A function is a *decision function* if it maps finite length bit strings into the range $\{0,1\}$. Let F be the set consisting of *all possible* decision functions. Show that set F is uncountable.

Solution.

From part (a), we know that there is a bijection from the set of all finite length bit strings to the set of natural numbers, hence we can think of decision functions as mapping natural numbers to single bits.

It is an easy bijection between a set of such mappings and a power set of the natural numbers. Namely, we associate with a decision function f a set $S = \{i \in \mathbb{N} \text{ s.t. } f(i) = 1\}$. Clearly, this is an injection and a surjection. Therefore, since the power set of the set of natural numbers is uncountable then so is the set of decision functions.

Alternative Solution: We can give a bijection between the set of all decision functions and the set of all infinite length bit strings, which by part (c), implies that the set of decision functions is uncountable. Let f be a decision function. We map f to an infinite length string as follows. We set bit i of the string to be $f(i)$. That is, our infinite length bit string will be $f(1)f(2)f(3)\dots$. Now, observe that this mapping is a bijection. In particular, suppose we have two different decision functions f and g. Since they are different, they must give a different output on some particular input. Suppose that this particular input is the number j. That is, $f(j) \neq g(j)$. Then, the corresponding infinite length binary strings will differ in the $j-th$ position. This shows that the function is injective. Now, if we have an infinite length bit string $b_0b_1b_2\dots$, then a decision function f defined as $f(0) = b_0, f(1) = b_1, f(2) = b_2, \dots,$

will be mapped to that string. Therefore, the mapping function is surjective. Since it's injective and surjective, it must be bijective. **Alternative Solution 2:**

Some students gave a diagonalization argument for why F is uncountable: Assume otherwise and let $\{f_1, f_2, f_3, ...\}$ be the list of all elements of F. From part (a) we know that B, the set of all finite length binary strings is countable, so there must exist a bijection $g : B \to \mathbb{N}$. Now, define a decision function $f : B \to \{0, 1\}$ as follows:

$$f(s) = \text{ the opposite of } f_{g(s)}(s)$$

Then f is different from all f_i's because for all $i \in \mathbb{N}$, h differs from f_i on string $s = g^{-1}(i)$.

Alternative Solution 3:

One can also argue, as some students did, that F is uncountable by showing the bijection between F and the set, $P(B)$, of all subsets of B. The bijection associates with $f \in F$ a set of strings $b \in B$ s.t. $f(b) = 1$. Then, since by part (a) there is a bijection between B and $\mathbb{N}$, there must be a bijection between $P(B)$ and $P(\mathbb{N})$. By transitivity of bijective relation between sets, there is a bijection then between F and $P(\mathbb{N})$. Hence F is uncountable.

$\square$

(e) A function is computable if there is a computer program that computes it. From your answers to the previous parts, prove that there is a decision function that is not computable.

Solution.

Since there are only countably many computer programs, but uncountably many decision functions, there can be no surjection from the set of computer programs to the set of decision functions (In fact, this means that there is *more* decision functions than computer programs). In particular, if we associate a computer program to a decision function which this program is computing[1], this association is not a surjection either. Therefore there must be some decision function which is not computed by any computer program.

$\square$

[1]If a program is not computing any decision function, associate it with a trivial decision function $f(b) = 0$, for all finite bit strings b.