CS70 In Simpler Terms - Note 3

Kevin Liu

July 3, 2017

1 Infinity and Countability

Countability may be a fairly new concept for many of you, and it is easy to confuse computability and countability. Here I will summarize what you need to know as well as a few tips that will guarantee you a few basic points on the exam.

- 2 sets have the same *cardinality*/size if we can demonstrate a bijection between the two sets(show onto and one-to-one)
- Set S is *countable* if there is a bijection between S and N
- $|\mathbb{N}| = |\mathbb{Z}| = |\mathbb{Q}|$ (all countable sets)
- if $|A| \leqslant |B|$ and $|B| \leqslant |A| \Rightarrow |A| = |B|$
- Binary strings of any finite length: $\{0,1\}$* (each digit in a binary string is from the set {0,1})
- Ternary strings: $\{0,1,2\}$*
- Lexicographic order: numerically increasing order
- Cantor's Diagonalization proof: proves that $\mathbb{R}$ is not countable by adding 2 (mod 10) to each of the values in the diagonal, and noticing that the diagonal number can't exist in the set. This proof can't be used on $\mathbb{Q}$ since adding 2 (mod 10) to a rational number does not guarantee that it wil still be a rational number.

2 Computability and the Halting Problem

In this section, I will highlight a few of the most common examples when dealing with computability problems. All of these examples rely on the fact that the Halting problem program does not exist. **General Halting Problem approach:**

Assume by contradiction that program P exists.
define Halt
Modify $F \Rightarrow F^{'}(x)$
Use P as a subroutine

If the original program halts, P returns true, otherwise false

Example problem:
Consider a program P that takes in F, input x, output y, returns true if $F(x)$ outputs y, and returns false otherwise.

```
def Halt(F, x):
    y = 0
    def F'(x):
        F(x)
        return y
    return P(F', x, y)
```

- Can a computer program print all rational numbers?
  Yes, since you can enumerate $\mathbb{Q}$ so you can print them.
- There is NO program DEAD which takes P, x, n and determines if the nth line is executed when you run $P(x)$.
- There exists a program H that determines whether a program P on input x that outputs the value $x + 42$ after executing 42 statements or *steps*. True statement.
- ****IMPORTANT**** You can count the number of steps that a Program has taken, but you can't determine whether a line has been executed.