University of Strathclyde Science

Lecture 9: Dogs and Collars

Dr John Levine

CS103 Machines, Languages and Computation
October 23rd 2015

Today's Lecture

- How to find the right collar for your dog: mapping from dog names onto collar numbers and vice versa.
- Why there are more dogs than collars on Planet Zog where the dogs' names are real numbers.
- Types of sets: finite, countable, uncountable.
- Assignments 4 and 5 (in this lecture)

Mappings to Show the Size of Sets

- Consider two sets shown below:

  A = {1, 2, 3, ... , 1000}.
  B = {1, 3, 5, …, 2001}.

- Which set is bigger?
- We could count the elements of each set, but this would be tedious.
- We can make a mapping from elements of the first set to elements of the second set, and then see if that mapping accounts for all elements of both sets.

Dogs and Collars

- Consider an infinite set of dogs.
- Each dog has a name, which is a finite string of letters (all in uppercase), such as “FIDO” or “GROMIT”.
- Consider an infinite set of collars, where every collar is labelled with an integer $n \in \{0, 1, 2, \ldots\}$.
- Are there enough collars for all the dogs?
- In other words, can I find a computable mapping from the dog’s name to an integer, so that every dog gets a unique collar?
- And does every collar have a unique dog?

The Dogs and Collars Problem

- Try to find a way to map the names of the dogs onto a unique collar (i.e. an integer).
- Is there a collar for every dog?
- Try to specify how the mapping would work in reverse, i.e. how to turn the collar number into the name of the dog that would own that collar.
- Try writing a program to perform the reverse mapping.
- Is there a dog for every collar?

Numbering the Dogs

- Mapping from dog names (strings) onto non-negative integers is quite straightforward.
- Easiest way: treat the string as a integer in base 27, e.g. "ACE" $\rightarrow$ 135 (base 27)
- But this means 100 base 27 has no dog, because we can't translate the zeros to a letter.
- We can't use A = 0, because this would produce the same number for "AARON" and "RON".
- Is there a numbering scheme which maps every dog to an integer and every integer to a dog?

Numbering the Dogs

- The strings have a natural order, so we can use that.
- First, assign the number 0 to the empty string “” (The Dog With No Name)
- Next, line up all the strings of length 1 in alphabetical order: “A”, “B”, … “Z”. These are numbered 1 up to 26.
- Next, line up all the strings of length 2: “AA”, “AB”, …, “AZ”, “BA”, …, “ZZ”. There are 676 of these, so they get numbered from 27 up to 702.
- Length 3: “AAA” $\rightarrow$ 703, …, “ZZZ” $\rightarrow$ 18278, and so on

Reversing the Mapping

- Decoding the collars (i.e. mapping from collar numbers back to strings) is a bit tricky, but can be done:

```
> (map-dog-to-collar  “AAA”)
703
> (map-collar-to-dog  703)
“AAA”
> (map-dog-to-collar  “GROMIT”)
91667882
> (map-collar-to-dog  91667882)
“GROMIT”
>
```

Assignment 4

1. Which collar is given to the dog called “ODIE”?

2. What is that name of the dog that gets collar number 106503?

The Hard Bit

- On Planet Zog, they name their dogs not with strings of letters, but with real numbers.
- The real numbers are the decimal numbers, including recurring numbers like $0.33333333333\ldots$ and irrational numbers like $\sqrt{2}$ and π which never repeat themselves.
- So the Zogians have dogs called things like: 3.58, 15.6, $0.3333333333\ldots$, $\sqrt{2}$, $\sqrt{3}$, π, e, …
- If the collars are still numbered with integers from 0 upwards, are there enough collars for all the dogs?

A Possible Mapping?

- Let's try a possible mapping from N to real numbers in the range 0.0 → 1.0 (i.e. a subset of R):

$$
\begin{array}{l}
0 \leftrightarrow 0\ .\ 3\ 3\ 3\ 3\ 3\ \ldots \\
1 \leftrightarrow 0\ .\ 3\ 1\ 4\ 1\ 5\ \ldots \\
2 \leftrightarrow 0\ .\ 5\ 0\ 0\ 0\ 0\ \ldots \\
3 \leftrightarrow 0\ .\ 0\ 1\ 4\ 1\ 5\ \ldots \\
4 \leftrightarrow 0\ .\ 0\ 1\ 3\ 1\ 5\ \ldots \\
\ldots
\end{array}
$$

- Have we managed to get a collar for every Zogian dog who has a name in the range 0.0 → 1.0?

The Diagonalisation Proof

- Let's try a possible mapping from N to real numbers in the range 0.0 $\rightarrow$ 1.0 (i.e. a subset of R):

$$
\begin{array}{l}
0 \leftrightarrow 0\ .\ {\color{red}3}\ 3\ 3\ 3\ 3\ \ldots \\
1 \leftrightarrow 0\ .\ 3\ {\color{red}1}\ 4\ 1\ 5\ \ldots \\
2 \leftrightarrow 0\ .\ 5\ 0\ {\color{red}0}\ 0\ 0\ \ldots \\
3 \leftrightarrow 0\ .\ 0\ 1\ 4\ {\color{red}1}\ 5\ \ldots \\
4 \leftrightarrow 0\ .\ 0\ 1\ 3\ 1\ {\color{red}5}\ \ldots \\
\ldots
\end{array}
$$

- Form a new number by adding 1 to the digits shown:

$? \leftrightarrow 0\ .\ {\color{red}4\ 2\ 1\ 2\ 6}\ \ldots$ no collar for this dog!

Zog’s Dogs are Uncountable

- The diagonalisation proof shows that it is impossible to align the real numbers with the integers.
- No matter what mapping we try, there will clearly be a large number of Zogian dogs who will get no collar!
- This shows that the set of all real numbers is a “larger” infinite set than the set of all integers.
- The set of integers is called a *countable* infinite set.
- The set of real numbers is called an *uncountable* infinite set.

Finite and Infinite Sets

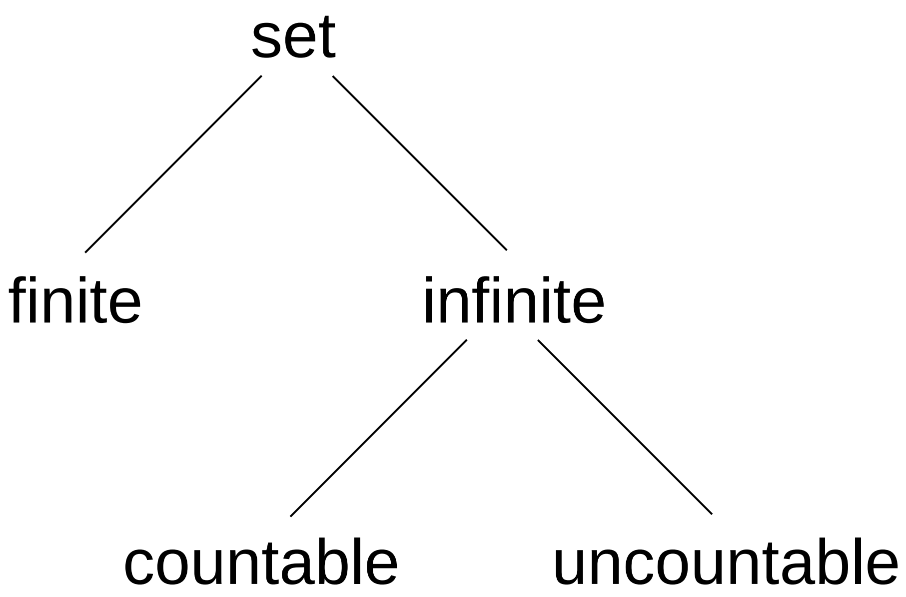
- A *finite* set is one which obeys the following property: if I list all the members of the set, I will eventually stop.
- Example: S = the set of all odd numbers < 1000.
- $S = \{1, 3, 5, 7, 9, 11, 13, 15, \ldots, 993, 995, 997, 999\}$.
- An *infinite* set is one where, if I list all the members of the set, I will never stop:
- Example: T = the set of all odd numbers > 1000.
- $T = \{1001, 1003, 1005, 1007, 1009, 1011, 1013, \ldots$

Countable and Uncountable Sets

- There are two types of infinite sets: *countable* sets and *uncountable* sets.
- Countable infinite sets can be made to align with the integers, like the dogs with strings for names.
- Example: the set of all strings of the MIU system.
- Uncountable infinite sets cannot be made to align with the integers, no matter what mapping you try.
- Example: the set of all real numbers.
- Uncountable infinite sets are “bigger” than countable infinite sets.

Countable and Uncountable Sets

- Summary of different types of sets:



Assignment 5

All sets are either finite or infinite, and all infinite sets are either countable or uncountable.

(a) Define what is mean by a *finite set.*

(b) Define what is mean by a *countable infinite set.*

(c) Define what is meant by an *uncountable infinite set*.

Assignment 5

(d) Label each of the following sets as finite or infinite, then label the infinite sets as countable or uncountable:

(i) the set of all students taking CS103

(ii) the set of all theorems of the MIU system

(iii) the set of all the prime numbers

(iv) the set $\{n \mid n \in N \text{ and } n < 5\}$

(v) the set of all the real numbers

(vi) the set of all dogs with four letter names

(vii) the set of all possible subsets of N

(viii) the set of all computer programs

(ix) the set of all functions from $N \rightarrow N$

Non-Computable Functions

- The first 5 weeks of the course will build up towards the proof that non-computable functions exist:

  If *P* is the set of all computer programs and *F* is the set of all functions $f: n \rightarrow m$ such that n,m are members of the set of natural numbers *N*, then $|F| > |P|$ and therefore there are some functions in *F* for which no computer program can exist.

Non-Computable Functions

- The first 5 weeks of the course will build up towards the proof that non-computable functions exist:

  If P is the set of all computer programs and F is the set of all functions $f\colon n \rightarrow m$ such that n,m are members of the set of natural numbers N, then $|F| > |P|$ and therefore there are some functions in F for which no computer program can exist.

Coming Up Next on CS103…

- How big is the set of all computer programs?
- How big is the set of all functions from $N \rightarrow N$?
- Are there more functions than there are programs?

- Final lecture before the test: Monday 26th October
- Class test: Friday 30th October, 10am
- Tutorials on Thursday 29th October, 10am / 1pm
- See syllabus document for what to revise!