Lecture 1

Notes for ECS 20 (Prof. Rogaway)

Scribe: Blake T.

TODAY:

1. **Introduction**
2. **Some Example Problems**
3. **Sentential Logic** (but we didn't get to this)

1. Introduction

Prof. Phil Rogaway introduced himself.
The TAs, Tung and Min-Eu, introduced themselves.
The course homepage is **www.cs.ucdavis.edu/~rogaway**.
A course information sheet is there, and you need to read it.
Problem Set 1 is due on **Monday** (see the course information sheet).

- Discrete mathematics deals with finite and countably infinite sets
- Seems to be a term rarely used by mathematicians, who say what the are doing more specifically.
- Some branches of discrete mathematics are:
  - Combinatorics (how to count things, how to make combinatorial objects that have desired properties)
  - Graph theory (points and two-elements subsets of them)
  - Logic
  - Set theory (normally dealt with in a class like this, but much modern set theory is *not* dealing with finite or countably infinite sets)
  - Probability (again, routinely treated in discrete math classes, but only when we assume that the underlying "probability space" is finite or countably infinite).
  - And much more

Helpful Techniques for Solving Discrete Math Problems

1. Generalize the problem (in the right way!)
2. Introduce variables (e.g., substituting n for 100 in Ex. 0)
3. Group terms cleverly (e.g., the algebraic analysis of Ex. 0)
4. Name the things that you are interested in.
5. Think recursively.
6. Solve small cases by hand and look for emerging patterns.
7. Substitute repeatedly to simplify "recurrence relations".
8. Use contradiction (this is demonstrated in Ex. 2).

9. Follow your nose (often only one natural path to go down) (eg, Ex. 2)

2. Some Example Problems

Example 0:
Find the following sum:

$$1 + 2 + \ldots + 100$$

or more generally

$$1 + 2 + \ldots + n$$

Solution:

This problem can be viewed algebraically by writing the list of numbers forwards and backwards, i.e.

$$\begin{array}{lccccccc} & 1 & + & 2 & + \ \ldots & + (n-1) & + & n \\ + & n & + & (n-1) & + \ \ldots & + \ 2 & + & 1 \\ \hline & (n+1) & + & (n+1) & + \ \ldots & + (n+1) & + & (n+1) \end{array}$$

which is $n(n + 1)$

Since this result is really twice the sum we are looking for it follows that

$$1 + 2 + \ldots + n = (n(n + 1)) / 2$$

This example can also be solved geometrically by writing twice 1 + 2 + … + n points in the following way:

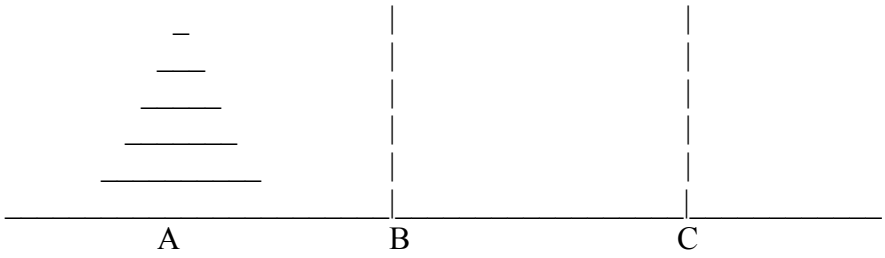
```
*   o   o   o   o
*   *   o   o   o
*   *   *   o   o        n points   (n=5 in the picture)
*   *   *   *   o
*   *   *   *   *
```

n + 1 points (n=5 in the picture)

So he total number of * points is

(n (n + 1)) / 2, which is the sum we are after.

Example 1: The Towers of Hanoi



Five rings of increasing diameter are placed on peg A. The pegs must be moved from A to C using the following rules:

- Only the topmost ring on a peg can be moved.
- A bigger ring cannot be placed on a smaller one.

Problem: Find a function that describes the least number of moves needed to solve the problem when you have n rings.

How many moves do you think it takes to move the five rings? Nobody guessed it or figured it out. Rogaway claims that the answer is 31.

We want a formula that specifies a number of moves that is both:

- **Sufficient**: there **is** a solution to the game using this number of moves.
- **Necessary**: no solution can use **fewer** moves than this.

Solution:

First, let's define what we're interested in. Let

T_n = The minimum number of moves needed to move the n rings from peg A to peg C (obeying the rules of the game).

Actually, this isn't good. We need to generalize the definition to do the job. So, instead, let

T_n = The minimum number of moves needed to move n rings from some one specified peg to some other specified peg (obeying the rules of the game).

Sufficiency:

Think recursively. Assume a "black box" algorithm can move the first n - 1 rings from any peg to any other peg. Solving the problem this way requires that the first n -1 rings be moved, then the largest ring be moved once, then the smaller rings be moved on to the largest ring. This number of moves can be represented by:

$$T_n \leq T_{n-1} + 1 + T_{n-1} = 2T_{n-1} + 1$$

Necessity:

Now we have to reason about *any* algorithm that solves the puzzle.
Any solution must move the largest ring to the final peg for the very last time. That takes one move. But before that happened, we had to get the n-1 rings that were formerly on top of the start peg and move them off to a free peg. That takes at least T_{n-1} moves. After we got the biggest ring to its destination peg, we had to move the n-1 smaller rings from the free peg where they were at to the final peg. That takes at least T_{n-1} moves. So, all in all, any solution needs to spend at least

$$T_n \geq 1 + T_{n-1} + T_{n-1} = 2T_{n-1} + 1$$

moves.

Putting together the two inequalities we have that

$$T_n = 2T_{n-1} + 1$$

We know that in order to move zero rings to their final location requires zero moves, so

$T_0 = 0$

Using this as our base value we can then determine that:

T_1	T_2	T_3	T_4	T_5	…	T_n	
1	3	7	15	31		$2^n - 1$	(apparently)

One way to get the general formula is by repeated substitution:

$$T_n = 2\,T_{n-1} + 1$$
$$= 2\,[2\,T_{n-2} + 1] + 1 = 2^2\,T_{n-2} + (1 + 2)$$
$$= 2^2\,[2\,T_{n-3} + 1] + (1 + 2) = 2^3\,T_{n-3} + (1 + 2 + 4)$$

$$= 2^n * 0 + (1 + 2 + 2^2 + \ldots + 2^{n-1})$$
$$= (1 + 2 + 2^2 + \ldots + 2^{n-1}) = 2^n - 1$$

Binary Representation

Example 2:

Prove that $x = \sqrt{2}$ is **irrational**.

Definition:

$x \in \mathbf{R}$ is **rational** if $x = p/q$ for some integers p and $q \in \mathbf{Z}$, $q \neq 0$.

$x \in \mathbf{R}$ is **irrational** if it is not rational.

Note: "if" in definitions mean "if and only if", or "exactly when"

Here we prove by contradiction.
Assume for contradiction that x *is* rational, ie,

$x = p/q$ for some integers p and q, $q \neq 0$

Without a loss of generality, it can be assumed that either p is odd or q is odd
(if they're both even, cross out the common factors of 2 until one of the numbers is odd).
Additionally, an odd number squared is still an odd number.

$\sqrt{2} = p/q$ square both sides:
$\rightarrow \quad 2 = p^2/q^2$ multiply through by the denominator:
$\rightarrow \quad 2q^2 = p^2$

From this we know that p is even, because the square of an odd number is odd (why?!). But then q is odd, because we know that at least one of p and q is odd. So we can write

$p = 2j$ for some $j \in \mathbf{Z}$ and

$q = 2i + 1$ for some $i \in \mathbf{Z}$

$\rightarrow 2(2i + 1)^2 = (2j)^2$

$\rightarrow (2i + 1)^2 = 2j^2$

$\rightarrow 4i^2 + 4i + 1 = 2j^2$

$\rightarrow 4(i^2 + i) + 1 = 2j^2 \qquad \rightarrow\leftarrow$

The contradiction is that an odd number (the LHS) can equal an even number (the RHS). We can conclude that our original assumption is wrong: $x = \sqrt{2}$ is **not** rational, which, by definition, means that it is **irrational**.

Example 3:

We claim that : 5 shuffles of a deck of cards is **not** enough to randomize the cards.

Here by **shuffles** I mean the usual "rifle shuffle." Prof. Rogaway demonstrates one with an imaginary deck.

Assume a starting sequence of

1, 2, 3, …, 51, 52

Even though we won't define what it means to randomize the cards, clearly a deck cannot be well randomized unless you can get **any** resulting sequence of cards, including, for example, the sequence:

52, 51, …, 3, 2, 1

We are going to show that 5 shuffles of this deck will **never** transform the specified starting sequence to the specified final sequence. So it can't do a good job of mixing the deck.

We didn't finish doing this, but we introduced an idea [*corrected in these notes*] that Prof. Rogaway claimed would help.

To illustrate the idea, let's look at a "small" deck of cards, say 8 cards, that, for example, happens to be in the order:

S = 3, 6, 7, 2, 1, 8, 4, 5

This sequences contains four "subsequences" of successive integers, namely,

3, 4, 5 and
6, 7, 8 and
1 and
2.

These are called the **rising sequences** of S**.** A rising sequence R from S is a *sequence of successive integers that appear as a (not necessarily consecutive) subsequence of S and that has the property that it cannot be extended, either to the left or to the right, while remaining a sequence of successive integers appearing as a subsequence of S.* In other words, a rising sequence in S is any **maximal** subsequence of S consisting of successive integers.

We're interested in **how many** rising sequences it takes to "cover" some given sequence S. For the example above, the answer was 4. Once we "remove" the rising sequence 3,4,5 we are left with 6,7,2,1,8; then we might remove the rising sequence 6,7,8 and we'll be left with 2,1; then we might remove the rising sequence 2 and we'll have just 1 left; and then we'd have to remove 1 and we'd be done. No matter what order you remove the rising sequences from S, they'll always be 4 of them to remove until you exhaust S. That's because the rising sequences of S are **uniquely determined** by S. To see this, note that if you name a number i in S then there's going to be a *unique* rising sequence containing i—the one determined in the obvious way by going to the left and right from i and grabbing whatever you can.

How many rising sequences does it take to cover the sequence 1, 2, …, 51, 52 ? That's easy: just 1. Namely S **itself** is a rising sequence that covers S.

How many rising sequences does it take to cover the sequence 52, 51, …, 2, 1? That's the other end of the spectrum: 52. Each rising sequence is just a single number i.

Now what on earth does any of this rising sequence stuff have to do with proving that five shuffles is inadequate to randomize a deck of cards? We'll find out next time!