CS 70 FALL 2006 — DISCUSSION #3

D. GARMIRE, L. ORECCHIA & B. RUBINSTEIN

1. Administrivia

(1) Course Information
  - The third homework is due September 15^{th} at 4pm in 283 Soda Hall.
  - You should already have received your graded homework #1.

2. Strong Induction: Sums of Fibonacci & Prime Numbers

Repeated from last week's sections.

Many of you may have heard of the Fibonacci sequence. We define $F_1 = 1, F_2 = 1$, and then define the rest of the sequence recursively: for $k \geq 3$, $F_k = F_{k-1} + F_{k-2}$. So the sequence ends up looking like:

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots$$

While not all positive integers are Fibonacci (e.g. 4), surprisingly we can express any positive integer as the sum of distinct terms in the Fibonacci sequence.

Theorem 1. *Every positive integer n can be expressed as the sum of distinct terms in the Fibonacci sequence.*

Proof. Let $P(n)$ be the statement that n can be expressed as the sum of distinct terms in the Fibonacci sequence. We begin with the base case $n = 1$. Since 1 is a term in the Fibonacci sequence (namely F_1), then $P(1)$ is true.

Now we proceed to the inductive step. We wish to show that $P(1) \wedge P(2) \wedge \cdots \wedge P(n) \implies P(n+1)$. So assume that $P(1), P(2), \ldots, P(n)$ hold. Now we consider $n+1$. There are two cases:

(1) $n + 1$ is itself a Fibonacci number.
(2) $n + 1$ is not a Fibonacci number.

If the former holds, then we're done. If the latter holds, then there must exist some positive integer k such that

$$F_k < n + 1 < F_{k+1}.$$

Since $F_k < n+1$, we may decompose $n+1$ into $F_k + (n+1-F_k)$. But by definition, $(n + 1 - F_k) < n + 1$ so by the inductive hypothesis we know that $P(n + 1 - F_k)$ is true, hence it may be expressed as such:

$$n + 1 - F_k = F_{i_1} + F_{i_2} + \cdots + F_{i_m}$$

Date: September 17, 2006.

The authors gratefully acknowledge Chris Crutchfield and Amir Kamil for the use of their previous notes, which form part of the basis for this handout.

where the subscripts are distinct. Moreover, since $n+1-F_k < F_k$ (since $n+1 < F_{k+1}$ implies $n+1-F_k < F_{k-1} < F_k$) it is not possible that any of the F_{i_j} could be equal to F_k. Therefore we have

$$n + 1 = F_k + F_{i_1} + F_{i_2} + \cdots + F_{i_m}$$

and $P(n + 1)$ holds. Thus by strong induction, $P(n)$ holds for all $n \geq 1$. □

Similarly one might attempt to prove the analogous result with primes (repeats allowed).

Exercise 1. Prove that all integers greater than one can be expressed as the sum of primes. □

3. Bad Induction Proofs

Sometimes we can mess up an induction proof by not proving our inductive hypothesis in full generality. Take, for instance, the following proof:

Theorem 2. *All acyclic graphs must have at least one more vertex than the number of edges.*

Proof. This proof will be by induction. Let $P(n)$ be the proposition that an acyclic graph of n vertices has at most $n - 1$ edges.

- Base Case: $n = 1$.
  Clearly a graph with no cycles and 1 vertex must not have any edges.
- Inductive Step: Suppose we have an acyclic graph G with n vertices.
  Adjoin a vertex v by edge e to the graph G to create the graph G'. Now we have an acyclic graph with $n + 1$ vertices. By the inductive hypothesis, the subgraph G has at most $n - 1$ edges. Therefore G' has at most n edges and we're done.

□

Exercise 2. What exactly went wrong with the proof?

To fix the proof, first prove that any acyclic graph must have at least one vertex of degree less than 2. Then prove that any acyclic (connected) graph with n vertices and at least one vertex of degree less than 2 has $n - 1$ edges.

4. Induction and Recursion

It is natural to prove facts about recursive functions using induction. Let's look at an example now.

A string over an alphabet Σ is a sequence of letters $a_1a_2 \ldots a_n$ such that each $a_i \in \Sigma$. The length of such a string as the previous one is n. There also exists the empty string λ with length 0. Consider the following recursive function:

```
len(a=a₁a₂...aₙ)
  if (a=λ) then return 0;
  else return 1 + len(a₂a₃...aₙ);
```

Exercise 3. Prove that `len()` correctly computes the length of any string.

5. The Stable Marriage Problem

Recall from class the Stable Marriage Problem, and the associated propose and reject (a.k.a. the Traditional Marriage) algorithm. We have proven the following facts already:

Facts 3. *For the case when men propose and women accept/reject:*

(i) (Improvement Lemma) if a girl dates a boy on day t, then she will date/be married to someone at least as good, every day from t onwards

(ii) (Corollary of the Improvement Lemma) each girl will marry her favorite amongst the boys she dated throughout the duration of the propose-rejection algorithm

(iii) No boy can be rejected by all the girls

(iv) The propose-reject algorithm terminates in at most n^2 days

(v) The propose-reject algorithm always produces a stable pairing, that is male-optimal and female-pessimal

You will need to use some of these facts on the homework. For today, let's practice with an example set of preference lists.

Exercise 4. Come up with 3 stable-marriage problems (i.e. three sets of boys and girls preference lists) and try running the propose-accept algorithm them. Think about why some of the above facts are true for the resulting pairings.