

CS2020A Discrete Mathematics

TUTORIAL 4 SUBMISSION

Submitted By

| | |
|---------------------------|------------------|
| Kalakuntla Parjanya | 112401018 |
| Prachurjya Pratim Goswami | 102401023 |
| Chavva Srinivasa Saketh | 112401009 |
| Madeti Tarini | 112401020 |
| Vedant Singh | 142401041 |

Prove the following or give a counterexample.

Theorem 1. For every natural number n ,

$$\sum_{i=0}^n f_i = f_{n+2} - 1$$

where f_0, f_1, \dots is the Fibonacci sequence defined as $f_0 = 0, f_1 = 1$ and for every $k \geq 2$,

$$f_k = f_{k-1} + f_{k-2}.$$

Proof:

Applying the Common Induction Principle on n ;

Base Case: For $n = 0$, $f_0 = 0$ and $f_2 - 1 = 1 - 1 = 0$. Thus the theorem is true for base case, $n = 0$.

Induction Hypothesis: Let the theorem be true for $n = k$. Then,

$$\sum_{i=0}^k f_i = f_{k+2} - 1$$

Then for $n = k + 1$,

$$\begin{aligned} \sum_{i=0}^{k+1} f_i &= f_{k+1} + \sum_{i=0}^k f_i \\ &= f_{k+1} + f_{k+2} - 1 \\ &= f_{k+3} - 1 \\ &= f_{(k+1)+2} - 1 \end{aligned}$$

The statement is true for $k + 1$

\therefore By mathematical induction, we can say that the theorem is true for all $n \in \mathbb{N}$

Theorem 2. Every natural number can be expressed as the sum of a unique set of powers of two.

Proof:

Let k be the smallest natural number that cannot be expressed as the sum of powers of 2.

Let 2^p be the maximum power of 2 $\leq k$.

$$k = 2^p + m \quad \text{for an } m \in \mathbb{N}$$

It is quite obvious that $m < k$. As k is the least counterexample for the theorem, m satisfies Theorem 2. Hence, m can be expressed as a sum of powers of 2.

Also,

$$\begin{aligned} k &< 2^{p+1} \\ 2^p + m &< 2^{p+1} \\ m &< 2^{p+1} - 2^p \\ m &< 2^p \end{aligned}$$

distinct

So, if $m = 2^{a_1} + 2^{a_2} + \dots + 2^{a_k}$, for some a_1, a_2, \dots, a_k , then

$$m = 2^{a_1} + 2^{a_2} + \dots + 2^{a_k} < 2^p$$

This means that none of a_1, a_2, \dots, a_k can be greater than or equal to p .

So, k can be written as, $k = 2^p + 2^{a_1} + 2^{a_2} + \dots + 2^{a_k}$.

Hence, any number can be written as the sum of powers of 2.

Now to show the uniqueness of this set of powers, assume that a number can be written as a sum of powers of 2 in two different ways.

Let $k = 2^{p_1} + 2^{p_2} + \dots + 2^{p_m} = 2^{q_1} + 2^{q_2} + \dots + 2^{q_n}$

Where p_1 to p_m and q_1 to q_n are in increasing order.

If p_1 is the least power of them all, then take 2^{p_1} as common and cancel on both sides.

$$\Rightarrow 1 + 2^{p_2-p_1} + \dots + 2^{p_m-p_1} = 2^{q_1-p_1} + 2^{q_2-p_1} + \dots + 2^{q_n-p_1}$$

For this to satisfy, there should be 1 on right side $\Rightarrow q_1 = p_1$.

Similarly, we can continue this process concluding that the number of terms and powers are same on both sides.

Having different number of terms will reach to a point where there are $m-n$ (if $m > n$) terms on one side and 0 on the other side which is not possible for powers of two.

Hence, k can be represented as sum of powers of two in only one way.

Just show LHS is odd & RHS is even for contradiction.

Theorem 3. The number of subsets of $\{1, \dots, n\}$ which do not contain any pair of consecutive numbers is f_{n+2} . (where f_n is defined in the first task)

Proof: On applying strong induction on n :

Base Case: For $n = 0$, the total number of subsets which do not contain any pair of consecutive numbers is $1 = f_2 = f_{0+2}, \{\emptyset\}$

Induction hypothesis: Let the given statement be true for all $i \in \mathbb{N}$, where $i \leq k$.

\therefore The number of subsets of $\{1, 2, 3, \dots, i\}$ is f_{i+2} for all $i \leq k$

Now for $k+1$:

There are two cases, either the subset will contain $k+1$ or it won't.

Case 1 :

If the subset does not contain $k+1$, the number of subsets with no consecutive elements will be the same as the number of subsets created using k elements.

So the number of subsets will be f_{k+2}

Case 2 :

Now the number of subsets which contain $k+1$ will definitely not contain k , so it will be same as the subsets of $\{1, 2, \dots, k-1\}$, with $k+1$ being added to all the subsets. So the number of subsets will be $f_{k-1+2} = f_{k+1}$ (Using strong induction hypothesis)

So the total number of subsets of $\{1, 2, 3, \dots, k+1\}$ is :

$$\begin{aligned} & f_{k+1} + f_{k+2} \\ &= f_{k+3} \\ &= f_{k+1+2} \end{aligned}$$

So, the statement is true for $k+1$ as well.

The statement is true for all $n \in \mathbb{N}$

\therefore So the number of subsets of $\{1, 2, \dots, n\}$ which do not contain any pair of consecutive numbers is f_{n+2} .

Theorem 4. For any two natural numbers a and b ,

$$\gcd(a, b) = \gcd(b, a \bmod b),$$

where $a \bmod b$ is the remainder obtained when dividing a by b .

Prove the correctness of the following algorithms using the principle of induction. Also argue why the algorithms will terminate.

Algorithm 1.

```
def gcd(a, b):
    # Input: Two natural numbers a and b
    # Output: The greatest common divisor of a and b
    if b == 0:
        return a
    else:
        return gcd(b, a % b)
```

Algorithm 2.

```
def gcd_ext(a, b):
    # Input: Two natural numbers a and b
    # Output: d, x, y, where d = gcd(a, b) and d = ax + by
    if b == 0:
        return a, 1, 0
    else:
        d, x, y = gcd_ext(b, a % b)
        return d, y, x - (a//b)*y
```

Solution:

Proof of Theorem 4:

Let a be any natural number and $b > 1$,

$$a = bq + r \quad (0 \leq r < b) \text{ and } r = a \bmod b \quad \checkmark$$

Now, to prove $\gcd(a,b) = \gcd(b,r)$,

Case 1: Let $d = \gcd(a,b) \implies d|a$ and $d|b \implies d|(a - bq) \implies d|r$ since $r = a - bq$

Case 2: Let $d = \gcd(b,r) \implies d|b$ and $d|r \implies d|(bq + r) \implies d|a$

$\implies d$ is a common divisor of a,b . Hence, $\gcd(b,r) \leq \gcd(a,b)$

$\therefore \gcd(a,b) = \gcd(b,r) \implies \gcd(a,b) = \gcd(b,a \bmod b)$

Algorithm 1:

To prove the correctness of Algorithm 1, we need to show that the recursive step and the termination step are valid.

→ Show base case holds.

Recursive Step: We can validate the recursive step using Theorem 4, proved above.

Termination Step: When $b = 0$, it means that in the previous step, $a \bmod b = 0$. This means that a is divisible by b . So, the gcd of any number divisible by b and b is b itself. This b is our a in the next step, which is what we are returning. Also, the second argument in each call, i.e; $a \bmod b$, is always strictly less than the previous second argument b . Hence, the algorithm terminates by reaching base case $b = 0$.

Algorithm 2:

To prove the correctness of Algorithm 2, we again need to show that the recursive step and the termination step are valid.

→ Proof for base case ??

Recursive Step: The output of the code is supposed to be (in order): d, x and y such that $d = \gcd(a,b)$ and $d = ax+by$.

$$a = bq + r \quad (\text{where } 0 \leq r < b \text{ and } q = a//b)$$

In the recursive step,

d is set to be the gcd of b and $a \bmod b$, while some new variables, say x_1 and y_1 are assigned such that now $d = bx_1 + ry_1$.

And then the function returns d as the gcd of the current step and y_1 and $x_1 - (a//b)y_1$ as the values of x and y , respectively, of the current step.

From the proof of Theorem 4, we have already proved that $\gcd(a,b) = d = \gcd(b,r)$. Now,

$$\begin{aligned} d &= bx_1 + ry_1 \quad (\text{for some } x_1, y_1 \in \mathbb{Z}) \\ \implies d &= bx_1 + (a - bq)y_1 \\ \implies d &= ay_1 + b(x_1 - qy_1) \\ \implies d &= ay_1 + b(x_1 - (a//b)y_1) \quad (\text{i}) \end{aligned}$$

But, since d is also the gcd of a and b ,

$$d = ax + by \quad (\text{ii})$$

From (i) and (ii), we get

$$x = y_1 \text{ and } y = x_1 - (a//b)y_1$$

where x and y are the coefficients of a and b (from the current step) and x_1 and y_1 are the coefficients of the previous (recursive) step.

Termination Step: Similar to Algorithm 1, the second argument of the function must continuously decrease to reach $b = 0$ case. As $\gcd(a,b)$, when $b = 0$ is a , $d = 1 \cdot a + 0 \cdot b$. Thus we return x, y as 1, 0.