Lab01-Algorithm Analysis

Exercises for Algorithms by Xiaofeng Gao, 2018 Spring Semester.

1. Fibonacci numbers are the numbers in the following integer sequence, called the Fibonacci sequence, and characterized by the fact that every number after the first two is the sum of the two preceding ones:

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$$

Use minimal counter example principle to prove that every integer $n \geq 1$ can be expressed as the sum of distinct Fibonacci numbers.

Solution:

Let $P(n)$ be the statement that an integer $n \geq 1$ can be a sum of distinct Fibonacci numbers.
As n=1 then, $F_i = 1$, where F_i is the ith number of fibonacci series.
if $P(n) \neq TRUE$ for $n \geq 1$, then the minimum value n=k is also $FALSE$
Since $P(1)$ is true, then $k \geq 2$
Let $P(k)$= $F_i + q$, where $F_i < k < F_{t+1}$
$q < k$ then $P(q)$ = TRUE.
Also $q < F_t + q < F_{t+1}$.
But this makes P(k) true as we can write k as the sum of F_t and Fibonacci sum of q. This cancel out the conjecture that k is the smallest number for which $P(k)$ is $false$.

2. Given the following sequence: $a_1 = 1$, $a_2 = 3$ and $a_k = a_{k-1} + 2a_{k-2}$, use strong mathematical induction to prove that a_n is odd for all integer $n \geq 1$.

Solution: Base: Show that the property is true for n=1 and n=2;

$a_1 = 1$ and $a_2 = 3$ as both 1 and 3 are odd.
Induction Hypo: If, for any integer k > 2, the property is true for all integers i such that $1 \leq i < k$,
then it is true for k;
Proof Inductive Hypothesis: Let k>2 be an integer, and suppose a_i is odd for all integers i such that $1 \leq i < k$.
Is a_k odd? , if so, the statement has been proved.
We know that $a_k = a_{k-2} + 2a_{k-1}$ by the definition of $a_1, a_2, a_3, ...$
$k - 2$ is less than k and is greater than or equal to 1 (because k >2).
By inductive hypothesis , a_{k-2} is odd.
Every term of the sequence is an integer (being a sum of products of integers),
$2a_{k-1}$ is even by definition of even (has 2 as a factor).
a_k is odd because it is the sum of an odd integer and an even integer.

3. For Algorithm 1 and Algorithm 2 shown below, answer the following questions respectively.

   (a) In Algorithm 1 and Algorithm 2, "and" and "xor" are bitwise operations*. What is the maximum number of times Line 6 is executed in Algorithm 1 when n is a power of 2? (*Hint*: Algorithm 1 is inspired by a special data structure.)

   (b) What is the time complexity of Algorithm 2 expressed in O, Ω and Θ notation?

*https://en.wikipedia.org/wiki/Bitwise_operation

Algorithm 1: COUNT1

```
  Input: n
1 count ← 0;
2 for i ← 1 to n do
3     j ← i;
4     while j ≠ 0 do
5         j ← j − (j and (j xor (j − 1)));
6         count ← count + 1;
```

Algorithm 2: COUNT2

```
Input: n
count ← 0;
if n is even then
    for i ← 1 to n do
        j ← i;
        while j ≠ 0 do
            j ←
              j − (j and (j xor (j − 1)));
            count ← count + 1;
else
    j ← ⌊n/2⌋;
    while j ≥ 1 do
        count ← count + 1;
        j ← ⌊j/2⌋;
```

<u>**Solution:**</u> For Algorithm1, The maximum number of time, when line 6 was executed when n is a power of 2 is:

$\sum_{i=1}^{n} log_{2i} = log_2(n!)$

For Algorithm 2, the Time complexity is

$f(n)$= $O\ log\frac{n}{2}$ and $\theta\ log\frac{n}{2}$

4. Given you two arrangements g_1 and g_2, decide whether $g_1 = \Omega(g_2)$ or $g_2 = \Omega(g_1)$ and give the reason.

   (a) $g_1 = n!$ and $g_2 = n^n$

   (b) $g_1 = \log_2 n$ and $g_2 = \sqrt{n}$

   (c) $g_1 = n^3$ and $g_2 = (\log_2 n)!$

   (d) $g_1 = (\lg n)^{\lg n}$ and $g_2 = 2^n$

<u>**Solution:**</u>

(a) $g_1 = n!$ = n(n-1)(n-2)...1

$g_2 = n^n = n.n...n$ where n > (n-1),(n-2)...

$g_1 < g_2 = \Omega\ (g_1)$

(b) H = $g_2 - g_1$ = $\sqrt{n}$ - $log_2 n$

$H' = \frac{1}{2}\ \frac{1}{\sqrt{n}} - \frac{1}{n.ln2} = \frac{\sqrt{n}.ln2-2}{2.nln2}$; $n \geq 4$

$H' \geq 0$, n $\geq$ 4, H is monotonic Increasing.

H $\geq$ H(4)=0

$\sqrt{n} - log_2 n = 0$

$\sqrt{n} \geq log_2 n$

$g_2 = \Omega(g_1)$

(c) m = $log_2 n \to n = 2^m$

g_1= $n^3 = 2^{3m} = 8^m$

$g_2 = m!$

$m \geq 8$ - Assume

g_1 = 8.8...8

$g_2 = m(m-8)(m-7)...1$

$g_2 > g_1$; $m \to \infty$
(d) m = logn, $n = 10^m$
$g_1 = m^m$; $g_2 = 2^{10^m}$
$log_2\ (g_1) =$ m$log_2\ m \leq m^2$
$log_2\ (g_2) = 10^m$
$log_2 m < m$; $m \to \infty$
$log(g_1) \leq log(g_2)$
$g_1 \leq g_2$ $m \to \infty$