CMPS 102 — Quarter Spring 2017 – Homework 2

VLADOI MARIAN

April 28, 2017

I have read and agree to the collaboration policy.Vladoi Marian
Name of students I worked with: Victor Shahbazian and Mitchell Etzel

Solution to Problem 2: Divide and Conquer

The pseudocode for the Guessing Number Algorithm:
Suppose we have a function Guess (x) . If you pass a integer for x , were x is your guess, this function will return : warmer, colder or you guessed it!.
Suppose that the integer that you have to guess is in the range 1 to n. All the integers in this range are stored in an array A that has size n. a = the first number in the range; b = the last number in the range; c = the number we guess.

GuessNumber (A, a =1, b = n, c = 1)

if (Guess (c) == " you guess it ") **then**
return "It is c" ;
end if

d = a + b - c; (d = the next guess)
m= $\lceil \frac{(c+d)}{2} \rceil$. (m = the point were we divide the array))

if ($d > c$) **then**
if (Guess(d) == "you guess it ") **then**
return "It is d" ;
else if (Guess(d) == "warmer") **then**
GuessNumber (A, m, b, d) (Call recursively the function. We are looking for values in the range $A_m \; to \; A_b$)
else
GuessNumber (A, a, m , d) (Call recursively the function. We are looking for values in the range $A_a \; to \; A_m$)
end if
end if

if ($d < c$) **then**
if (Guess(d) == "you guess it ") **then**
return "It is d" ;

else if (Guess(d) == "warmer") **then**

GuessNumber (A, a, m , d) (Call recursively the function. We are looking for values in the range $A_a \; to \; A_m$)

else

GuessNumber (A, m, b , d) (Call recursively the function. We are looking for values in the range $A_m \; to \; A_b$)

end if

end if

Our first guesses are 1 and n.We consider that (c+d)/2 = (a+b)/2 . Solving for d = a + b - c. The way we defined c forces us to guess values between -n and 2n. c is the last number we guess and d is the new number. Using the return value of the function Guess(x), and divide and conquer aproach, each call to function GuessNumber discard half of the values in our range. The recurrence relation is T(n/2) + O(1). Solving the reccurrence relation with Master Theorem, case 3 , we get the running time of our algorithm to be O($log_2(n)$). This algorithm takes ($log_2(n)$) + O(1) guesses in the worst case scenario. The Space Complexity is O(n) because we can use only an Array of size n to reprezent the range of values.

Proving by induction that the algorithm works.

Claim 1. *Let P(n) be the assertion that algorithm guess the corect number for values in the range 1 to n,* $\forall n$.

Proof. Base case: In the case where n=1 , the algorith works. We have only one value in the array and this is our guess

Inductive Step: We assume that the algorithm works for values in the range of 1 to n , where $1 < n \leq k$; We prove that the algorithm works for values in the range 1 to k + 1;
When range of the values is 1 to k+1 , we have 3 cases:
1. if the range has only one element (the algorithm works)
2. If (Guess(x) == warmer) we call recursively our function GuessNumber on (k+1)/2 values.Assuming that the recursive calls work correctly, this call works too.
3. If (Guess(x) == colder) we call recursively our function GuessNumber on (k+1)/2 values.Assuming that the recursive calls work correctly, this call works too.
The algorithm works correctly in all cases, we can conclude by induction that the algorithm guess the corect number $\forall n$. □