26th Nov

Theorem: For any integer $a > 0$ and $n \geq 0$, the algorithm Power(a, n) correctly computes a^n

Proof: Induction on n.

Basis Step: $n = 0$.
Power(a,n) return 1. Hence, the output is correct.

Induction Hypothesis: For every integer $0 \leq n \leq k$, Power(a,n) correctly outputs a^n.

Induction Step: Consider $n = k+1$.
$k+1 \geq 1$.

Case-(i): $(k+1)$ is odd
Then $p = \frac{k}{2}$ is an Integer.
By induction hypothesis Power$(a, k/2)$ correctly outputs $a^{k/2}$.

Subsequently the algorithm outputs $a \cdot a^{k/2} \cdot a^{k/2} = a^{k+1}$.
Hence, the algorithm works correctly when n is odd.

Algorithm:
(i) Proof of Correctness $\quad a^n$
(ii) Time complexity $\quad n \geq 0$

$a^n \quad a > 0$

Power(a,n)
```
if(n = 0) then return 1;
if(n is odd) then
    p = (n-1)/2;
    x = Power (a, p);
    Return a*x*x;
else (n is even)
    p = n/2;
    x = Power (a, p);
    return x*x;
endif.
```

Case-(ii): $(k+1)$ is even.
Then $p = \frac{k+1}{2}$ is integer.
By induction hypothesis, Power $(a, \frac{k+1}{2})$ correctly outputs $a^{(k+1)/2}$.
Subsequently the algorithm outputs $(a^{(k+1)/2})^2 = a^{k+1}$
Hence, the algorithm works correctly when n is even.

Since the cases are mutually exhaustive, this completes the proof of correctness of Power (a, n) for any integer $a > 0$ and $n \geq 0$.

MergeSort: (A, n)

If $(n = 1)$ then return A.

$A_L = A[1, 2, \ldots, n/2]$;

$A_R = A[n/2 + 1, \ldots, n]$;

MergeSort $(A_L, n/2)$;

MergeSort $(A_R, n/2)$;

Combine $(A_L, A_R, n/2, n/2)$;

Soln. **Power (a, n)**

$T(n) = T(n/2) + \alpha$

α is a constant independent of n.

$$T(n) = T(n/2) + \alpha$$
$$= T(n/4) + \alpha + \alpha$$
$$= \vdots$$
$$= T\left(\frac{n}{2^t}\right) + t\alpha.$$

$n/2^t = 1$

$t = \log_2(n+1)$

$1 + \alpha . \log_2 (n+1)$

$O(\log_2 n)$. $\leq 2 . \alpha \log_2 n$

$(n = 1)$ then

$f_0 := 0$

$f_1 = 1$

$f_n = f_{n-1} + f_{n-2}$

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

$T(n, k)$

$= T(n-1, k-1)$
$+ T(n-1, k)$.

Permutation of n numbers

$n = 6$

$5, 4, 3, 2, 1, 6$

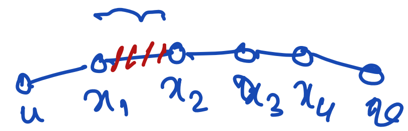
$2, 3, 4, 5, 6, 1 \rightarrow$ derangement

Exercise: If P is a shortest simple path between u and v in G, then any subpath $\hat{P}$ of P between x and y is a shortest simple path between x and y.

(Prove it)

Exercise: Prove or disprove:

If P be a longest simple path between u and v in G, then any subpath $\hat{P}$ of P between x and y is a longest simple path between x and y in G.



$x_1 - x_2 - x_3$

$u - x_1 - x_2$

$x_3 - x_4 - v$

A simple path with maximum possible number of edges (maximum total weight in case of weighted graph) is called LONGEST PATH between u and v.