ICPC Notebook

Gustavo Guedes

May 11, 2022

Contents

1 Teste

Testando

1.1 Hello World

```
#include<iostream>

using namespace std;

int main(){
    cout << "Hello World\n";

    return 0;
}
```

2 String

2.1 Suffix Array

```
#include<bits/stdc++.h>
#define MAX 112345
#define _ra(i) (i<n? ra[i]:0)
using namespace std;

typedef pair<int, int> ii;
int n, k;
string s;
int sa[MAX], ra[MAX], tmp[MAX];

//Construct Suffix Array - O(m log n)
bool cmp(int a, int b){
    return ii(ra[a],_ra(a+k))<ii(ra[b],_ra(b+k));
}
void build_sa(){
    for(int i=0; i<n; i++){sa[i]=i; ra[i]=s[i];}
    for(k=1; k<n; k<<=1){
        sort(sa, sa+n, cmp);
        tmp[sa[0]]=0;
        for(int i=1; i<n; i++){
            tmp[sa[i]] = tmp[sa[i-1]];
            if( !(ra[sa[i]]==ra[sa[i-1]] &&
                    _ra(sa[i]+k)==_ra(sa[i-1]+k)) )
                tmp[sa[i]]++;
        }
        for(int i=0; i<n; i++) ra[i]=tmp[i];
    }
}

//Search - O(m log n)
int bsearch(int i, int f, string P, int lower){
    if(i>=f) return i;
    int m = (i+f)/2;
    int c = s.compare(sa[m],P.length(),P);

    if(lower && c<0) return bsearch(m+1,f,P,lower);
    else if(lower) return bsearch(i,m,P,lower);
    if(c>0) return bsearch(i, m, P, lower);
    return bsearch(m+1, f, P, lower);
}
ii search(string P){
    int l = bsearch(0, n-1, P, 1); //find first
    int u = bsearch(0, n-1, P, 0); //find next from last
    if(l==-1) return ii(-1,-1);
    return ii(l,u);
}

int lcp[MAX], plcp[MAX], phi[MAX];
//Compute Longest Common Prefix - O(n)
void compute_lcp(){
    int L=0;
    phi[sa[0]]=-1;
    for(int i=1; i<n; i++) phi[sa[i]]=sa[i-1];
    for(int i=0; i<n; i++){
        if(phi[i]==-1) {plcp[i]=0;continue;}
        while(s[i+L]==s[phi[i]+L]) L++;
        plcp[i]=L;
        L = max(L-1,0);
    }

    for(int i=0;i<n; i++)lcp[i]=plcp[sa[i]];
}

//Finding Longest Repeated Substring - O(n)
int lrs(){
    int m=0;
    for(int i=0; i<n; i++) m = max(m, lcp[i]);
    return m;
}

//Finding Longest Common Substring - O(n)
int owner[MAX];
int lcs(){
    int m=0;
    for(int i=1; i<n; i++){
        if(owner[sa[i]]==owner[sa[i-1]]) continue;
        m = max(m, lcp[i]);
    }
    return m;
}

int main(){
    ios_base::sync_with_stdio(0);cin.tie(0);
    //-----
    s="BANANABAN$";
    n = (int) s.length();
    build_sa(); //9 5 7 3 1 6 0 8 4 2
    compute_lcp(); //0 0 1 2 3 0 3 0 1 2
    ii f = search("AN"); // (2,5)
    int r = lrs(); //3 ("ANA", "BAN")

    //-----
    string s1 = "GATAGACA";
    string s2 = "CATA";
    s = s1+"$"+s2+"#";
    n = (int) s.length();

    build_sa(); //13 8 12 7 5 3 10 1 6 9 4 0 11 2
    compute_lcp(); //0 0 0 1 1 1 1 3 0 2 0 2 0 2
    f = search("GA"); //10 12

    int n1 = (int) s1.length();
    for(int i=0; i<n; i++)
        owner[i]=i<=n1?1:2; //Only if >1 strings
    r = lcs(); //3 ("ATA")
}
```