ACM/ICPC 模板

XXXXXX

2023 年 2 月 15 日

目录

1 图论

1.1 Dijkstra

1.1.1 堆优化的 Dijkstra

```cpp
void dij(int s) {
    memset(dis, 0x3f, sizeof dis);
    dis[s] = 0; //初始化dis数组
    priority_queue<pair<int, int>> q; //一个优先队列
    q.push({0, s}); //c++中的pair<int, int>默认先按第一维比较，所以把dis值放在第一维
    while(!q.empty()) {
        int x = q.top().second; q.pop(); //取堆顶
        if(flag[x]) continue; //判断是否已经在S集合中，如果已经在S集合中，那么直接continue
        flag[x] = true; //标记x在S集合中
        for(int i = fr[x], y; i; i = nxt[i]) { //遍历所有与x相连的边
            y = to[i];
            if(dis[y] > dis[x] + w[i]) { //判断是否可以松弛。
                dis[y] = dis[x] + w[i];
                if(!flag[y]) q.push({-dis[y], y}); //c++中的优先队列默认为大根堆，而我们需要小根堆，为了方便可以直接取相反数
            }
        }
    }
}
```

1.1.2 朴素的 Dijkstra

```cpp
void dij(int s) {
    memset(dis, 0x3f, sizeof dis);
    dis[s] = 0; //初始化dis数组
    for(int k = 1, t; k <= n; ++k) { //一共n轮，每次找不在S集合中的点中dis值最小的点，将其放入S集合中并用它去松弛其他点
        t = -1;
        for(int i = 1; i <= n; ++i)
            if(!flag[i] && (t == -1 || dis[i] < dis[t])) t = i; //如果（不在S中 && （t还没有赋值 || i的dis值比t更小））， 那么t = i
        flag[t] = true; //标记t在S集合中
        for(int i = 1; i <= n; ++i) //松弛操作
            dis[i] = min(dis[i], dis[t] + a[t][i]);
    }
}
```

2 字符串

2.1 KMP

```
// 未经验证
int nxt[MAXN];
void getNext(char s[]) { // 模式串next数组
    int n = strlen(s + 1);
    for(int i = 0; i <= n; ++i) nxt[i] = 0;
    for(int i = 2, j = 0; i <= n; ++j) {
        while(j && s[i] != s[j + 1]) j = nxt[j];
        if(s[i] == s[j + 1]) ++j;
        nxt[i] = j;
    }
}
int kmp(char s[], char t[]) { //返回第一次匹配的地方，否则返回-1
    int n = strlen(s + 1), m = strlen(t + 1);
    for(int i = 1, j = 0; i <= m; ++j) {
        while(j && t[i] != s[j + 1]) j = nxt[j];
        if(t[i] == s[j + 1]) ++j;
        if(j == n) return i - m + 1;
    }
    return -1;
}
```

2.2 后缀数组

```
struct SA {
    static const int MAXN = 1e5 + 50;
    static char tmpStr[MAXN];
    static int n, sa[MAXN], rk[MAXN], cnt[MAXN], x[MAXN], y[MAXN];
    static int height[MAXN], st[MAXN][35]; // 不需要可以删除，注意必须连同下面初始化一起删除
    static void work(string s) {
        strcpy(tmpStr + 1, s.c_str());
        work(tmpStr);
    }
    static void work(char s[]) {
        int m = 255; n = strlen(s + 1);
        for(int i = 1; i <= n; ++i) ++cnt[x[i] = s[i]];
        for(int i = 2; i <= m; ++i) cnt[i] += cnt[i - 1];
        for(int i = n; i >= 1; --i) sa[cnt[x[i]]--] = i;
        for(int k = 1, p = 0; k <= n; k <<= 1, m = p, p = 0) {
            for(int i = n - k + 1; i <= n; ++i) y[++p] = i;
            for(int i = 1; i <= n; ++i) if(sa[i] > k) y[++p] = sa[i] - k;
            for(int i = 1; i <= m; ++i) cnt[i] = 0;
            for(int i = 1; i <= n; ++i) ++cnt[x[i]];
            for(int i = 2; i <= m; ++i) cnt[i] += cnt[i - 1];
            for(int i = n; i >= 1; --i) sa[cnt[x[y[i]]]--] = y[i], y[i] = 0;
            swap(x, y), x[sa[1]] = p = 1;
            for(int i = 2; i <= n; ++i)
                x[sa[i]] = (y[sa[i]] == y[sa[i - 1]] && y[sa[i] + k] == y[sa[i - 1] + k]) ? p :
                    ++p;
            if(p == n) break;
        }
        for(int i = 1; i <= n; ++i) rk[sa[i]] = i;
        // height数组：引理 height[rk[i]] >= height[rk[i - 1]] - 1
        for(int i = 1, k = 0; i <= n; ++i) {
            if(rk[i] == 1) continue;
            if(k) --k;
            int j = sa[rk[i] - 1];
            while(j + k <= n && i + k <= n && s[i + k] == s[j + k]) ++k;
            height[rk[i]] = k;
```

```
        }
        // lcp数组的ST表：lcp(sa[i], sa[j]) = min(height[i.....j])
        for(int i = 1; i <= n; ++i) st[i][0] = height[i];
        for(int i = n; i >= 1; --i)
            for(int j = 1; i + (1 << j) - 1 <= n; ++j)
                st[i][j] = min(st[i][j - 1], st[i + (1 << j) - 1][j - 1]);
    }
    static int lcp(int i, int j) { // ST表求LCP
        int a = rk[i], b = rk[j];
        if(a == b) return 1;
        if(a > b) swap(a, b);
        int len = log2(b - a);
        return min(st[a + 1][len], st[b - (1 << len) + 1][len]);
    }
};
char SA::tmpStr[MAXN];
int SA::n, SA::sa[MAXN], SA::rk[MAXN], SA::cnt[MAXN], SA::x[MAXN], SA::y[MAXN];
int SA::height[MAXN], SA::st[MAXN][35]; // 这里
```