

A*算法

A*算法是对BFS的一种改进，定义起点为 s 终点为 t 那么需要找到从起点开始到某一个点 n 的估值函数 $g(n)$ 和从这个点到终点的估值函数 $h(n)$ 然后定义函数 $f(n)=g(n)+h(n)$ 用 $f(n)$ 重载运算符加入优先队列，然后和BFS相似的搜索方式。

例题

[洛谷p1379](#)

八数码问题

选取估值函数 $g(n)$ 为当前图案和最终图案有多少个方块不用。

代码

```
#include <algorithm>
#include <cstdio>
#include <cstring>
#include <queue>
#include <set>
using namespace std;
const int dx[4] = {1, -1, 0, 0}, dy[4] = {0, 0, 1, -1};
int fx, fy;
char ch;
struct matrix {
    int a[5][5];
    bool operator<(matrix x) const {
        for (int i = 1; i <= 3; i++)
            for (int j = 1; j <= 3; j++)
                if (a[i][j] != x.a[i][j]) return a[i][j] < x.a[i][j];
        return false;
    }
} f, st;
int h(matrix a) {
    int ret = 0;
    for (int i = 1; i <= 3; i++)
        for (int j = 1; j <= 3; j++)
            if (a.a[i][j] != st.a[i][j]) ret++;
    return ret;
}
struct node {
    matrix a;
    int t;
    bool operator<(node x) const { return t + h(a) > x.t + h(x.a); }
} x;
priority_queue<node> q;
set<matrix> s;
```

```
int main() {
    st.a[1][1] = 1;
    st.a[1][2] = 2;
    st.a[1][3] = 3;
    st.a[2][1] = 8;
    st.a[2][2] = 0;
    st.a[2][3] = 4;
    st.a[3][1] = 7;
    st.a[3][2] = 6;
    st.a[3][3] = 5;
    for (int i = 1; i <= 3; i++)
        for (int j = 1; j <= 3; j++) {
            scanf(" %c", &ch);
            f.a[i][j] = ch - '0';
        }
    q.push({f, 0});
    while (!q.empty()) {
        x = q.top();
        q.pop();
        if (!h(x.a)) {
            printf("%d\n", x.t);
            return 0;
        }
        for (int i = 1; i <= 3; i++)
            for (int j = 1; j <= 3; j++)
                if (!x.a.a[i][j]) fx = i, fy = j;
        for (int i = 0; i < 4; i++) {
            int xx = fx + dx[i], yy = fy + dy[i];
            if (1 <= xx && xx <= 3 && 1 <= yy && yy <= 3) {
                swap(x.a.a[fx][fy], x.a.a[xx][yy]);
                if (!s.count(x.a)) s.insert(x.a), q.push({x.a, x.t + 1});
                swap(x.a.a[fx][fy], x.a.a[xx][yy]);
            }
        }
    }
    return 0;
}
```

From:
<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:qgjyf2001:a_%E6%90%9C%E7%B4%A2

Last update: 2020/08/21 13:27