同济大学计算机系

人工智能课程 8 数码实验报告

姓名学号 李辉 1652286、刘兵 1752397、孙文丽 1752844

专　　业 计算机科学

授课老师 武妍

一、 实验概述

(一) 实验目的

熟悉和掌握启发式搜索的定义、估价函数和算法过程，并利用 A*算法求解 8 数码难题，理解求解流程和搜索顺序。

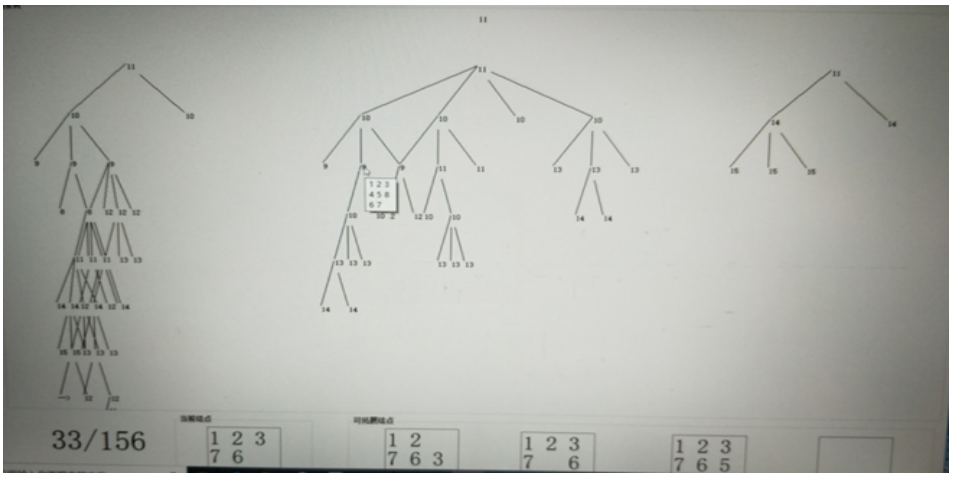
(二) 实验内容

1.以 8 数码问题为例实现 A*算法的求解程序（编程语言不限），要求设计两种不同的估价函数。

2.设置相同初始状态和目标状态，针对不同的估价函数，求得问题的解，并比较它们对搜索算法性能的影响，包括扩展节点数、生成节点数和运行时间。画出不同启发函数 h(n)求解 8 数码问题的结果比较表，进行性能分析

3. 要求界面显示初始状态，目标状态和中间搜索步骤。

4. 画出图示所示的搜索生成的树，在每个节点显示对应节点的 f*(n)值，以显示搜索过程，以红色标注出最终结果所选用的路线。



二、 实验方案设计

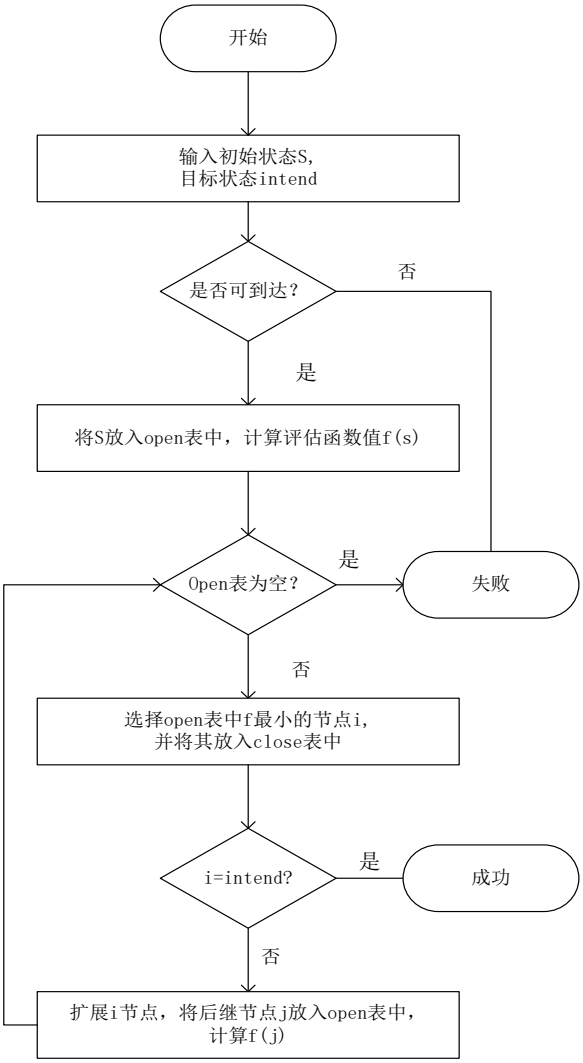
(一) 总体设计思路与总体架构

采用 A*搜索算法，分别使用两种评估函数（f(n)=g(n)+h(n)）搜索最短路径。

利用优先队列存放节点。open 表存放下一步可扩展的节点，closed 表用于存放已扩展的节点。用 STL 向量实现优先队列。

如果节点可到达，则执行 A*搜索算法：每次选择 open 表中评估函数值最小的节点进行扩展，直到 open 表为空（搜索失败）或者找到目标节点。

最后打印选取的最佳搜索路径、扩展节点数、生成节点数和运行时间。



（二）　核心算法及基本原理

①A*算法属于有信息搜索（启发式搜索），根据评估函数估计每一个节点效用值，选择最优位置并进行下一步扩展，是基于树或图求出最低通过成本的算法。该算法可以省略大量无意义的搜索路径，更为高效，并且具有完备性和最优性。

A*算法最为核心的部分，在于评估函数的设计：f(n)=g(n)+h(n)

其中 f(n)为当前节点到目标节点的最小路径代价的估计值；

g(n)是从初始节点到当前节点的实际代价；

h(n)是当前节点到目标节点最小代价的估计函数，只依赖于函数状态，并具有一致性和可采纳性，有以下两种方案：

a）　$h_1(n)$=棋子不在位数之和

b）　$h_2(n)$=每个棋子与目标位置的距离（曼哈顿距离）

g(n)所占的比重越大，越趋向于宽度优先或等代价搜索；

反之，h(n)的比重越大，启发性能就越强。

②算法描述

把起始节点 S 放到 open 表中；

如果 open 是空表，则失败退出，无解；

计算 open 表中节点的评估函数值；

从 open 表中选择一个 f 值最小的节点 q[n](n≥1), 若其中有目标节点，则搜索成功，否则生成该节点所有的后继节点；

将扩展后的节点放入 close 表中；

回到步骤 2。

③复杂度分析

A*的时间复杂度在最大绝对误差下是指数级。 考虑每步骤代价均为常量，我们可以把这记为 $O(b^{\varepsilon d})$, 其中 d 是解所在深度。考虑绝大多数实用的启发式，绝对误差至少是路径代价 h(n)的一部分，所以 ε 是常量或者递增的并且时间复杂度随 d 呈指数级别的增长。

A*的空间复杂度为 O(n)，n 是节点总数。

（三） 模块设计

①数据结构设计

用 node 类记录节点状态、评估函数值、后继节点和父结点等信息；

用 STL 向量表示优先队列。

②功能说明

函数名	作用域	功能
void valuation(int index)	node 成员函数	记录八数码状态
int dis()	node 成员函数	计算 f(n)
bool isend()	node 成员函数	判断是否为目标状态
bool isequal(node q)	node 成员函数	判断当前节点与节点 q 是否相等
bool isexpansive(node &n)	全局函数	判断是否可扩展，避免重复
bool isempty()	全局函数	判断是否还有可访问点
bool able_to_end()	全局函数	判断是否可到达目标状态
vector<int> find_min()	全局函数	找到 open 表中评估值最小节点
void breath(int index)	全局函数	找到后继节点
void print(int index, vector<node>&rstep_v)	全局函数	打印到 index 节点的路径
void process()	全局函数	定义算法执行过程

(四)创新内容

网上一些资料显示，在选择评估函数值最小的节点过程中，如果出现 f(n)相等的情况，将会任选其中一个进行扩展。本实验采用的方法是扩展所有 f(n)最小的节点，这样

做的优势是可以避免评估函数对节点区分度不大的缺陷，一定程度上减少不必要的搜索过程，缺点是牺牲了空间复杂度。

三、 实验过程

（一） 环境说明

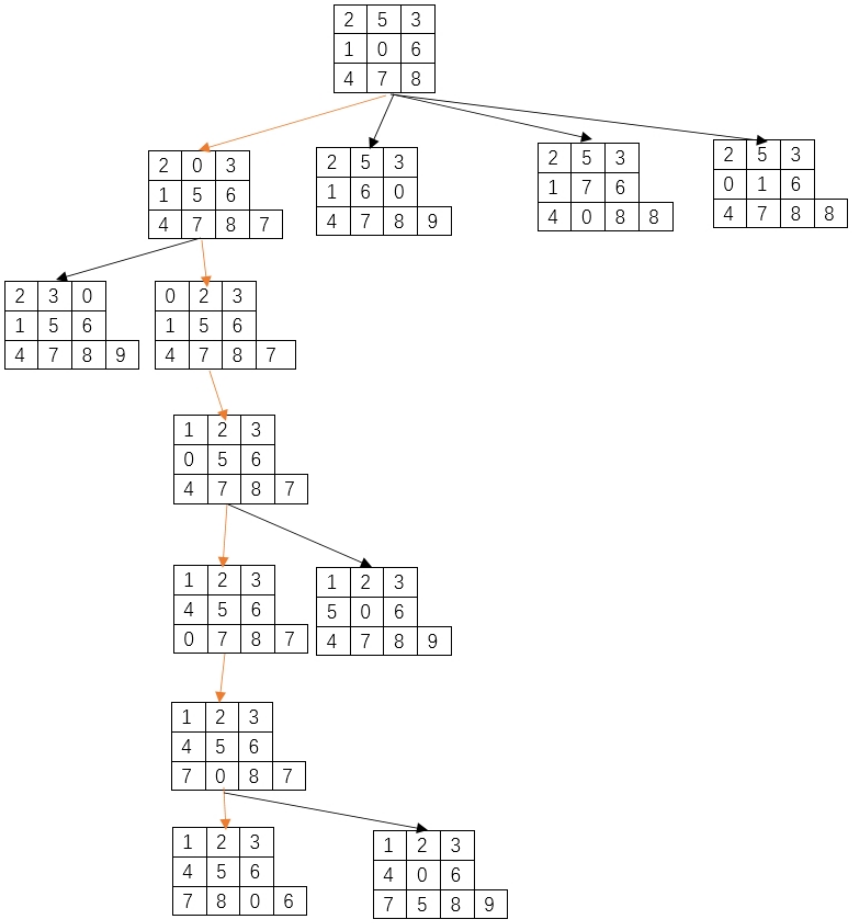
操作系统：Windows；
语言：C++;
开发环境：Dev-C++
核心使用库：iostream,vector,string,time.h,cmath,queue,stdlib.h,stdio.h

（二） 源代码文件清单

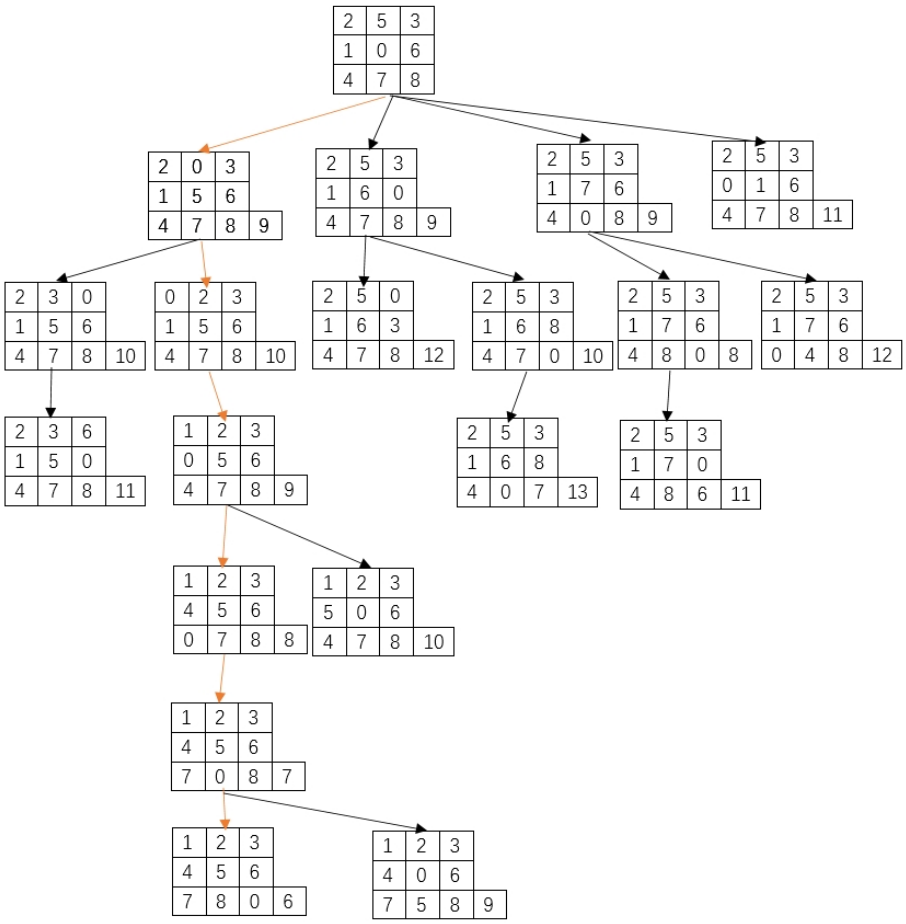
八数码.cpp：评估器源文件，用于实现 A*算法，求解 8 数码难题。

（三） 实验结果展示

1. 不在位数搜索树



2. 与目标距离和搜索树



注：初始状态为 1 2 3
4 5 6
7 8 0

目标状态 启发函数	8 2 3 4 5 7 6 1 0	2 3 4 5 7 8 1 0 6	4 3 1 6 8 7 2 0 5	3 4 6 7 8 1 0 5 2	5 2 3 4 6 7 0 8 1
$h_1(n)$=不在位数之和	扩展节点数:24 生成节点数:25882 操作时间为: 23448.0000ms	扩展节点数:19 生成节点数:3635 操作时间为: 1553.0000ms	扩展节点数:23 生成节点数:18948 操作时间为: 24410.0000ms	扩展节点数:18 生成节点数:1816 操作时间为: 669.0000ms	扩展节点数:20 生成节点数:5280 操作时间为: 2079.0000ms
$h_2(n)$=与目标位置的距离	扩展节点数:24 生成节点数:2916 操作时间为: 926.0000ms	扩展节点数:19 生成节点数:789 操作时间为: 551.0000ms	扩展节点数:23 生成节点数:2198 操作时间为: 926.0000ms	扩展节点数:18 生成节点数:167 操作时间为: 210.0000ms	扩展节点数:20 生成节点数:871 操作时间为: 418.0000ms

性能分析：A*算法本身具有完备性和最优性。根据输入-运行时间表，与目标位置的距离作为启发函数时间复杂度更低。

四、 总结

（一） 实验中存在的问题及解决方案

问题：如何处理 open 表中存在多个评估函数值最小的节点

解决方案：扩展所有评估函数值最小的节点，这样可以避免评估函数对节点区分度不大的缺陷，一定程度上减少不必要的搜索过程，缺点是牺牲了空间复杂度。我们选择通过向量表示的一个优先队列解决了该问题。

（二） 心得体会

通过本次实验，我更深刻地体会到 A*算法优势。相比广度优先搜索等无信息搜索，启发式搜索的优势在于省略了很多无意义的搜索路径。除此之外，通过对比发现启发式函数与 A*算法的复杂度直接相关。在小组合作方面，我们组分工明确，执行力很强，效率较高，合作愉快。但在实验初期我们也走过一些弯路，由于没有完全理解实验要求，过于重视可视化，还好及时问了助教，省去了一部分工作量。

做完这个实验加深了我对 A*算法搜索过程的理解。当然程序还有需要提升的地方，例如通过图形界面展示搜索过程。在 8 数码实验中，我们小组三人协作十分愉快，互帮互助，互相提高。过程中我有不懂的地方都向其他两位同学请教，他们都会和我一起探索并提出可行意见。我基础比较薄弱，很感谢他们给我机会负责代码部分并给予指导。相信我们在之后的实验中会继续紧密协作，高效高质的完成每一个实验，好好掌握人工智能基础理论知识。

（三） 后续改进方向

一是改进启发函数，在保证完备性和最优性的前提下争取更低的时间复杂度和空间复杂度。比如设计不相交的模式数据库，这样可以大大减少生成的节点。

二是完善图形界面，在可视化方面下功夫，可以让搜索过程更加清晰明了，用户界面更友好。

（四） 总结

本次实验是 A*算法的一个实际应用。通过此次实验，理解了启发式函数对于 A*的重要性，具象体会了八数码问题的解决过程（搜索树），更加深刻地认识了人工智能技术的强大。

参考文献

https://blog.csdn.net/u012283461/article/details/79078653
https://www.cnblogs.com/guanghe/p/5485816.html

成员分工与自评

成员	分工
李辉	搜索过程的实现，编写代码
刘兵	搜索树的可视化
孙文丽	总结实验，撰写实验报告，制作 PPT

1. 李辉

我主要负责搜索过程的实现，编写代码这一部分。程序可以实现打印出搜索路径，显示所有扩展出的节点个数和路径节点个数，还能显示搜索过程的时间。主要通过调用 STL 向量存储扩展过程中的节点，一个向量 v 用于存储过程中所有扩展出来的节点，另一个向量 rstep_v 用于存储最优搜索路径路程中的节点。选用了两种启发式函数（9 宫格中不在位数和曼哈顿距离——每个数字与目标状态的距离之和）。遇到的主要困难是当遇到多个评估值相同的待扩展节点时如何进行同步扩展，这个地方卡了很长时间，最后我选择了通过向量表示的一个优先队列解决了这个问题。

2. 刘兵

本次实验中，我负责搜索树的可视化。要做到将搜索树完整展示出来，并且标注出最终的搜索路径，需要对 A*算法的求解过程有一个比较清晰的了解，并且熟悉启发函数的应用。通过这个实验，我明白 A*算法的关键在于设计一个合适的启发函数，这样对于目标的搜索将变得事半功倍，空间占用也会更小。

3. 孙文丽

我负责总结实验，撰写实验报告，制作 PPT。通过总结实验，对 A*算法的理解更为透彻，参与了整个实验进程，对 8 数码问题有更深刻的思考，而且锻炼了文案水平。