基于ARA*算法的动态迷宫搜索

1. 背景知识

ARA*算法，是一种高效的任意时刻规划算法，采用迭代运行膨胀启发式A*算法的思想，但是在保留次优性因子的前提下重用了之前计算的结果。下面将依次介绍A*算法、膨胀启发式A*、ARA*。

1.1 A*算法

A*算法是一种启发式搜索算法，对于一个节点 `n` ，它使用 到达节点n的实际成本`g(n)` 和 到达目标的估计剩余成本`h(n)` 的组合来评估 综合成本`f(n)=g(n)+h(n)` 。该算法维护要探索的节点的优先级队列，其中具有 最低综合成本`f(n)` 的节点首先被探索。
A*算法在运算过程中，每次从优先队列中选取 `f(n)` 值最小（优先级最高）的节点作为下一个待遍历的节点。另外，A*算法使用两个集合来表示待遍历的节点，与已经遍历过的节点，这通常称之为 `open_set` 和 `close_set` 。
完整的A*算法描述如下：

- 初始化 `open_set` 和 `close_set` ；
- 将起点加入 `open_set` 中，并设置 `f(n)` 为 `0` （优先级最高）；
- 如果 `open_set` 不为空，则从 `open_set` 中选取优先级最高的节点 `n` ：
  - 如果节点 `n` 为终点，则：
    - 从终点开始逐步追踪 `parent` 节点，一直达到起点；
    - 返回找到的结果路径，算法结束；
  - 如果节点 `n` 不是终点，则：
    - 将节点 `n` 从 `open_set` 中删除，并加入 `close_set` 中；
    - 遍历节点 `n` 所有的邻近节点：
      - 如果邻近节点 `m` 在 `close_set` 中，则：
        - 跳过，选取下一个邻近节点
      - 如果邻近节点 `m` 也不在 `open_set` 中，则：
        - 设置节点 `m` 的 `parent` 为节点 `n`
        - 计算节点 `m` 的优先级
        - 将节点 `m` 加入 `open_set` 中

`h(n)` 是A*算法的启发函数， `h(n)` 的计算方法将直接决定A*算法的正确性和速度。

- 如果 `h(n)` 始终小于等于节点 `n` 到终点的代价，则A*算法保证一定能够找到最短路径。但是当 `h(n)` 的值越小，算法将遍历越多的节点，也就导致算法越慢。在极端情况下，当启发函数 `h(n)` 始终为0，则将由g(n)决定节点的优先级，此时算法就退化成了Dijkstra算法。
- 如果 `h(n)` 完全等于节点 `n` 到终点的代价，则A*算法将找到最佳路径，并且速度很快。可惜的是，并非所有场景下都能做到这一点。因为在没有达到终点之前，我们很难确切算出距离终点还有多远。
- 如果 `h(n)` 的值比节点n到终点的代价要大，则A*算法不能保证找到最短路径，不过此时会很快。
  对于 `h(n)` 的计算方式最常见的有：曼哈顿距离、对角距离、欧几里得距离。可根据需要自行选择。

1.2 膨胀启发式A*算法

膨胀启发式A*算法（A* search with inflated heuristics）是一种被证明在多数情况都更快但是不能保证找到最优解的算法。它在传统A*算法的启发式函数中乘上一个大于 `1` 的膨胀因子ϵ。
一般来说A*会将一个一致性启发式函数作为输入，启发式函数要满足：

- 对于s的后继节点s'，当$s \neq s_{goal}$时，$h(s) \leq c(s, s') + h(s')$，当$s = s_{goal}$时，$h(s) = 0$

这里的$c(s, s')$是指s和s'的代价距离（必须为正）。一致性是指启发式函数$h(s)$的值不大于s到目标点的真实代价距离，这保证了启发式函数的可采纳性（admissible）。膨胀式启发函数（指用大于一的数ϵ与启发式函数$h(n)$相乘）可以减少扩展节点的数量，使得搜索更快。但是这样就丧失了算法的可采纳性，得到的结果不再能被保证是最优解。

下面是膨胀启发式A*算法的伪代码：

$01\ g(s_{start}) = 0; OPEN = \emptyset;$
02 insert s_{start} into $OPEN$ with $f(s_{start}) = \epsilon * h(s_{start});$
03 while(s_{goal} is not expanded)
04 remove s with the smallest f-value from $OPEN$;
05 for each successor s' of s
06 if s' was not visited before then
07 $f(s') = g(s') = \infty;$
08 if $g(s') > g(s) + c(s, s')$
09 $g(s') = g(s) + c(s, s');$
10 $f(s') = g(s') + \epsilon * h(s');$
11 insert s' into $OPEN$ with $f(s');$

1.3 ARA*算法

ARA*（Anytime Repairing A*）算法多次执行膨胀启发式A*算法，首先选取一个较大的ϵ进行计算，之后在计算之前将ϵ逐渐减小直到$\epsilon = 1$。这样的好处是在任意时刻我们都可以保证有一条到达终点的路径。
具体的ARA*算法实现请学习提供的文章：
ARA*: Anytime A* with Provable Bounds on Sub-Optimality

2. 题目描述

妮可校园被魔法师变成了一个巨大的迷宫，在迷宫的深处埋藏着坤坤收集的宝藏。不怕苦难的勇敢小妮可想要找到坤坤埋藏的宝藏，成为妮可校园的王者。但是魔法师却想将坤坤的宝藏占为己有，所以他会在小妮可探索途中改变迷宫样子。但是善良的坤坤会阻止魔法师将宝藏封存起来，也就是说，总是会存在一条通往宝藏的康庄大道。
坤坤为了考察小妮可，会在任何时刻向小妮可发出询问，看看小妮可是否知道前往宝藏的路线。如果小妮可不能告诉他，他将进入伤心食堂进食。如果坤坤进食，会因为伤心食堂的饭菜而陷入沉思，就不能阻止魔法师了，魔法师将把宝藏占为己有，小妮可就再也找不到通往宝藏的康庄大道了。
TOO SAD /(◕‿‿◕)\

注意，小妮可还会被魔法师追赶，所以他不能在一个地方停留太久，也不能不停的在迷宫中奔跑。现在就请你来帮助小妮可尽快的找到宝藏吧。

当然，聪明的小妮可自身也是可以抵御魔法师的攻击的，所以如果魔法师妄图向小妮可所在的位置发动进攻，这次攻击将被视为无效。

Input and Output

当程序开始运行时，您需要向程序传递一个参数。参数有两种选择:terminal和gui。terminal意味着您应该在终端中打印所需的信息。GUI意味着你应该显示一个窗口。

Input：

第1行：n m e，表示迷宫M的行数和列数，以及ARA*算法迭代开始的权值（默认迷宫起始位置为(0,0)，宝藏位置为(n-1,m-1)，n，m<=100）；
第2~n+1行：每行m个数，用空格隔开，$M_{ij} = 0$表示第i行j列个迷宫格子没有障碍，$M_{ij} = 1$表示第i行j列个迷宫格子有障碍；
第n+2行：p，表示魔法师施展魔法的次数；
第n+3~n+2+p行：每行有3个数t，i，j，表示当e=t时，在第i行第j列设置障碍，即令$M_{ij} = 1$（如果此时小妮可刚好位于M_{ij}，这此次攻击无效）；
第n+3+p行：k，表示坤坤询问小妮可的次数；
第n+4+p~n+3+p+k行：每行一个数t，表示当e=t时，坤坤向小妮可发出询问。

对于M_{ij}，$0 <= i < n, 0 <= j < m$。

注意：对于同一时刻，不同行动的优先级是：设置障碍>计算路径>询问>移动。也就是说如果在时间t,魔法师与坤坤同时做出行动，那么你应该先设置障碍改变地图，然后使用新的地图寻找你的路径，然后用新的路径回答坤坤，所有的做完之后再向下一个格子移动。（可移动的方向包含周围八个方向而非仅上下左右四个，还包括左上，右上，左下，右下，无论是哪个方向，都算是走了1步）

Output：

2*k行。假设e=t时，小妮可在(a_0, b_0)，此时他前往宝藏的可能路径是$(a_0, b_0) \to (a_1, b_1) \to ... \to (n-1, m-1)$共L个点，则你在此时先输出一行一个数：
L
再输出一行共$2 * L$个数，用空格隔开：
$a_0\ b_0\ a_1\ b_1\ ...\ n-1\ m-1$

提示

每当e-1，小妮可都必须前进一步，然后再重新对路径迭代。

此时可能存在很多条路径，我们会确定一个最优路径范围[low,high]，其中如果你的L满足$low <= L <= high$，你将被判定为正确结果。所以请尽可能优化你的评估函数。请不要试图用传统最短路方法trick，因为每次询问都是有时间限制的，超过时间限制的结果将不被接受。

请严格遵守上述要求。当我们测试您的程序时，我们将使用输入数据运行您的程序并进行比较您的结果与标准输出数据。如果您的程序没有遵循上述规定限制和打印其他东西，比较将失败，你将失去测试分数。

如果您对输入/输出格式有任何疑问，您可以询问学生助理或老师。

3. 项目要求

项目的实际工作包含两个部分:

- 编写上面描述的系统。(50分)
- 报告。(20分)

3.1 编写上面描述的系统

首先，你应该建立这样一个系统。

- (10分)程序应该能够从标准输入读取初始状态；
- (10分)程序应能在合理的时间内找出问题的解决方案；
- (10分)程序应能在终端模式下运行，并以标准输出显示结果；
- (10分)该程序应该能够在GUI模式下运行，并显示小妮可和宝藏在迷宫中的位置，并且标出被魔法师攻击过的地方（包括可能失效的攻击）；
- (10分)提供一个按钮查看询问结果(小妮可从当前位置前往宝藏位置的路径)，即：
  - 第一次按动按钮，时间从初始时间跳到第一次询问时间，展示第一次询问结果的路线；
  - 第二次按动按钮，时间从第一次询问时间跳到第二次询问时间，展示第二次询问结果；
  - 中间以此类推；
  - 最后一次按动按钮，时间从最后一次询问时间跳到小妮可到达终点。
    每次时间跳转的中间时间段，你应该正常执行移动和加入障碍，并更新地图和小妮可位置。 时间增加一个单位，小妮可移动一格，直至小妮可找到宝藏。

3.2 报告

请包括你想让我们知道的关于你的项目的一切。例如项目的基本结构、项目中使用的数据结构、优化性能的方式，以及使您的项目与其他项目不同的任何内容。

请包括对样例数据或者你自己生成的数据的运行结果。向我们展示它是否能得出正确的答案。

在报告中清楚地说明如何运行程序。当我们测试你的程序时，这是必要的。

但是，我们对报告的长度没有要求。请不要使它不必要的长。记得写上小组成员和学生编号。另外，请记住以“pdf”格式上传报告。

报告中英文均可。

3.3 运行测试

你可能会注意到，以上部分加起来是70分。最后，我们将在您的程序上运行一些我们准备好的测试。这是你项目的最后30点。这只能在提交代码之后完成。我们将用一些案例来测试你的程序。通过指定初始状态，程序的结果应该与标准答案相同。每个测试用例的难度是不同的。有些例子很小的地图。然而，有些情况下包含的内容远不止这些。你需要尝试最小化算法所花费的时间。每个测试用例都有一些分数，你的分数取决于你通过的测试用例的数量。你的算法必须能够通过GUI向我们展示这个过程。

在这个项目中可以使用和不能使用的东西是有限制的。

你可以使用各种工具来推进你的项目，但第三方物理引擎、游戏引擎和AI工具是不允许的。例如，你可以使用OpenGL和OpenCL，但不允许在项目中使用Unity、Unreal或Torch。

您可以使用algs4库。如果你想使用某个库，但又不确定，你可以询问。

4. Bonus

随机生成可解输入状态。(最多10分)

你不能从网上或其他组的同学抄袭代码，否则我们将按照CS部门的抄袭政策处理这种情况。