# 搬运工问题的启示

重庆外语学校 刘汝佳

# 二 搬运工问题及其特点

在对状态空间搜索算法有一定了解之后，我们来看看我们的搬运工问题。究竟用什么方法比较好呢？让我们先来看看该问题的特点。

## 1. 搬运工问题

我们在前面已经介绍过搬运工问题，这里我只是想提一些和解题有关的注意事项。首先，我们考虑的搬运工问题的地图规模最大是 20*20，这已经可以满足大部分关卡了。为了以后讨论方便，我们把地图加以编号。从左往右各列称为 A，B，C…，而从上往下各行叫 a,b,c…。而由于不推箱子时的走路并不重要，我们在记录解的时候忽略了人的位置和移动，只记录箱子的移动。人的动作很容易根据箱子的动作推出来。下面是包含解答的标准关卡第一关。


*He-Ge, Hd-Hc-Hd, Fe-Ff, Fh-Gh-Hh-Ih-Jh-Kh-Lh-Mh-Nh-Oh-Ph-Qh-Rh-Rg,*
*Ff-Fg-Fh-Gh-Hh-Ih-Jh-Kh-Lh-Mh-Nh-Oh-Ph-Qh-Qi-Ri,*
*Fc-Fd-Fe-Ff-Fg-Fh-Gh-Hh-Ih-Jh-Kh-Lh-Mh-Nh-Oh-Ph-Qh-Qg,*
*Ge-Fe-Ff-Fg-Fh-Gh-Hh-Ih-Jh-Kh-Lh-Mh-Nh-Oh-Ph-Qh-Rh,*
*Hd-He-Ge-Fe-Ff-Fg-Fh-Gh-Hh-Ih-Jh-Kh-Lh-Mh-Nh-Oh-Ph-Pi-Qi,*
*Ch-Dh-Eh-Fh-Gh-Hh-Ih-Jh-Kh-Lh-Mh-Nh-Oh-Ph-Qh*

呵呵，怎么样，第一关都要那么多步啊…以后的各关，可是越来越难。

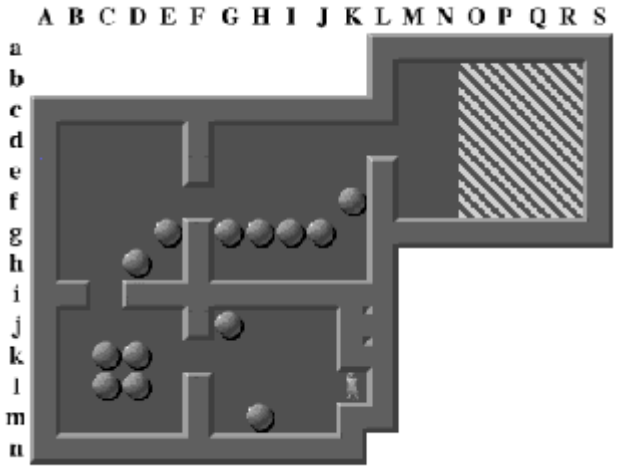
## 2. 搬运工问题的特点

我在前言里吹了这么半天，我想你即使以前没有玩，现在也已经玩过了吧：)。

有什么感觉呢？是不是变化太多了，不好把握？不仅人不好把握，连编程序也变得困难了很多。我们不妨拿它与经典的 8 数码问题作一个比较。

**1.死锁！**

初学者很快就会学到什么是死锁 – 一旦他（她）把一个箱子推到角上。显然，这样的布局再继续玩下去是没戏了，不管以后怎么推都不可能把这个箱子推离那个角。不少玩家都总结了不少死锁的经验，但是要比较系统的解决这个问题并不是一件容易的事。我们将用整整一章（其实也不长啦）的篇幅来分析这个问题。




典型的死锁。想一想，为什么：）我们再看一下 8 数码问题。它没有死锁，因为每一步都是可逆的。在这一点上，搬运工问题要令人头疼得多了。
容易看出，这样的状态空间不是无向图，而是有向图。

**2.状态空间。**

8 数码问题每次最多有 4 中移动方法，最多的步数也只有几十步。而搬运工问题呢？困难一点的关卡可以是一步有 100 多种选择，整个解答包括 600 多次推箱子动作。分支因子和解答树深度都这么大，状态空间自然就非同小可了。

**3.下界估计**

在启发式搜索中，我们需要计算 h 值，也就是需要对下界进行估计。8 数码问题有很多不错的下界函数（如“离家”距离和），但是搬运工问题又怎么样呢？我们不能直接计算“离家”距离，因为谁的家是哪儿都不清楚。很自然，我们可以做一个二分图的最佳匹配，但是这个下界怎么样呢？

**a.准确性**

对于 A*及其变种来说，下界与实际代价越接近，一般来说算法效率就越高。我们这个最佳匹配只是“理想情况”，但是事实上，在很多情况下箱子相互制约，不得已离开目标路线来为其他箱子腾位置的事情是非常普遍的。例如我们的标准关卡第 50 关，有的箱子需要从目标格子穿过并离开它来为其它箱子让路。我们的下界函数返回值是 100，但是目前的最好结果是 370。多么大的差别！

**b.效率**

由于下界函数是一个调用非常频繁的函数，其效率不容忽视。最佳匹配的时间渐进复杂度大约是 O(N^3),比 8 数码的下界函数不知大了多少…我们将会在后

面给出一些改进方法，但是其本质不会改变。

# 3. 如何解决搬运工问题

已经有人证明了搬运工问题是 NP-Hard，看来我们还是考虑搜索吧。回想一下上一节提到过的状态空间搜索，用哪一种比较好呢？

既然是智力游戏，可用的启发式信息是非常丰富了，我们不仅是要用，而且要用得尽量充分,所以应该用启发式搜索。而前面已经提到了，搬运工问题的状态空间是非常大的，A*是没有办法了，因此我们选择了 IDA*算法：实现简单，空间需求也少。

既然搬运工问题这么难，为什么有那么多人都解决了相当数量的关卡呢（标准的 90N 年以前就被人们模透了）。因为人聪明嘛。他们会预测，会安排，会学习，有直觉的帮助，还有一定的冒险精神。他们（也包括我啦，呵呵）常用的是一些“高层次”的解题策略，既有效，又灵活。(Srbga:想学吗? Readers:当然想!!) 可惜这些策略不是那么简单易学，也不是很有规律的。在后面的章节中，我将尽力模仿人的思维方式给我们的程序加入尽量多的智能。