由“汽车问题”浅谈深度搜索的一个方面

————搜索对象与策略的重要性

问 题 描 述

有一个人在某个公共汽车站上，从 12:00 到 12:59 观察公共汽车到达本站的情况，该站被多条公共汽车线路所公用，他依次记下公共汽车到达本站的时刻。

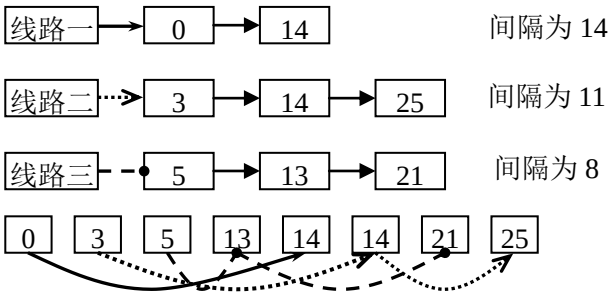
- 在 12:00－12:59 期间，同一条线路上的公共汽车以相同的时间间隔到站。
- 时间单位用“分”表示，从 0 到 59 。
- 每条公共汽车线路至少有两辆车到达本站。
- 公共汽车线路数 K 一定≤17，汽车数目 N 一定小于 300。
- 来自不同线路的公共汽车可能在同一时刻到达本站。
- 不同公共汽车线路的车首次到站时间和到站的时间间隔都有可能相同。

请为公共汽车线路编一个调度表，目标是：公共汽车线路数目最少的情况下，使公共汽车到达本站的时刻满足输入数据的要求。

例如：

汽车编号	1	2	3	4	5	6	7	8
到达时间	0	3	5	13	14	14	21	25

那就可能存在这样一个解，由以下 3 条汽车线路组成：



解　　析

经过一系列的分析，我们决定用深度搜索（由于通篇讨论的是深度搜索，以下就统一简称搜索）解这道题目。

对这样一个问题，首先提取出三个关键要素：**时间、车、路线**。

☆车辆的特征是时间，

☆路线的特征是“首发车时间”和“间隔时间”，这等效于“第一辆车”和“第二辆车”。

面对这三个关键要素，下面就要从中确定搜索对象和搜索策略。可以看出，**题目要求的是车和线路的关系，而时间在其中起的是描述作用和条件制约作用**，因此，本题的搜索对象应该是车或线路这两个关键要素。

● 分析搜索对象及策略

1．对象—→ 车

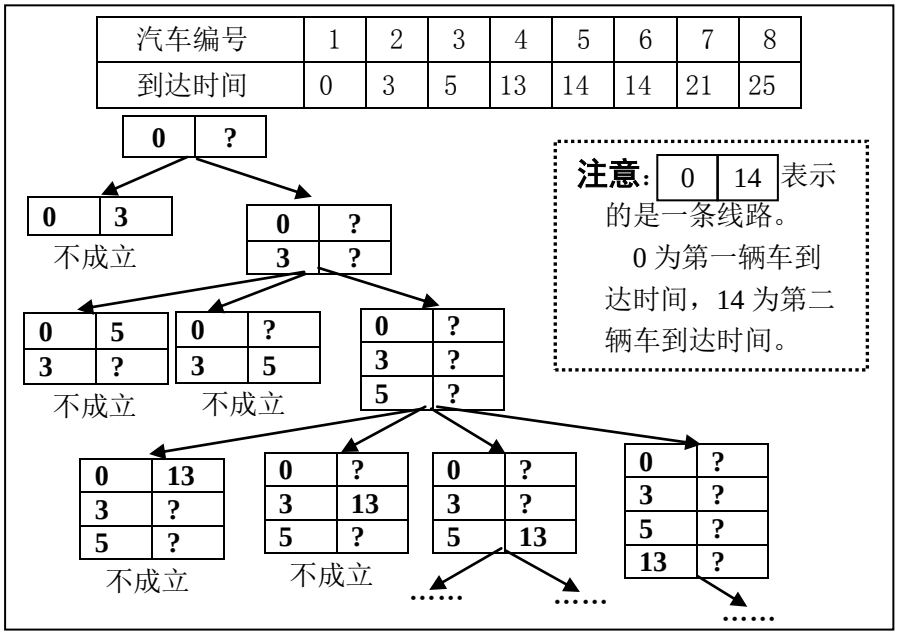
由此对象而产生的搜索策略是：按到站时间顺序，依次枚举每辆车属于哪条路线。

注意路线的特征，若一路线的**第一辆车**和**第二辆车**确定了，那该路线也就确定了。

大致搜索方案为：

按到达时间顺序，依次对于那些没有确定属于哪条线路的车进行枚举，该车属于某新线路的**第一辆车**或属于某已有线路的**第二辆车**，若为后一种选择，则可确定路线上其他所有的车辆。

还是看先前给的那个简单例子来构造搜索树大致如下：



观察该搜索树，发现：随着搜索树层数的递增，每层节点所扩展出的树叉数目逐渐增大。从直观上说：该搜索树从根开始，“分叉”越来越多，“枝叶”越来越茂盛。

这就是搜索对象为车的搜索树的典型特点。

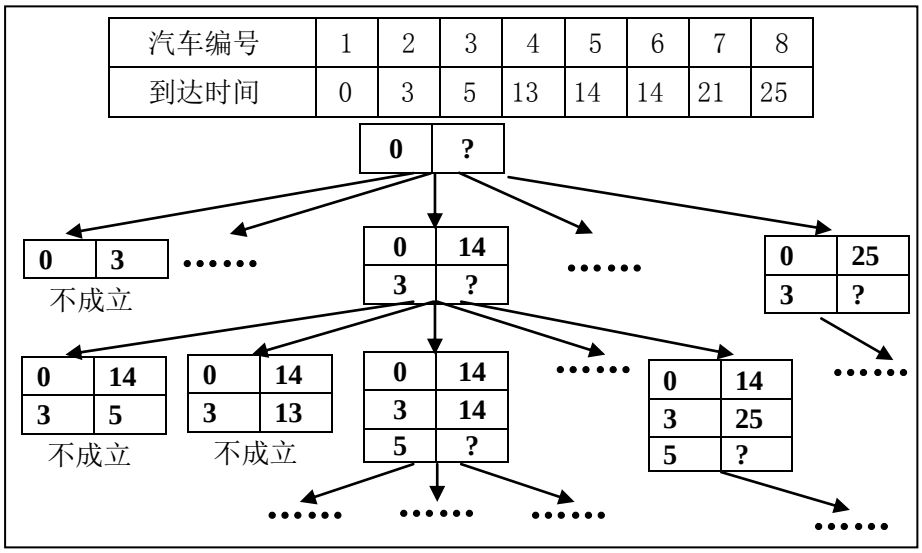
2．对象—→线路

由此对象而产生的搜索策略是：枚举每条路线包含哪些车，确定该路线。

实际上，对每条路线，我们只要枚举其特征：**第一辆车**和**第二辆车**。再根据“有序化”思想，固定所有路线是按照第一辆车的到达时间为关键字排序的。

大致搜索方案为：

搜索每层都要确定一条路线：将未确定归属路线的到达时间最小的车**固定**为新路线的第一辆车，其后枚举这条路线的第二辆车，从而确定该路线。



同样，根据先前给的那个简单例子我们也构造出了方法二的搜索树（见前页）。

观察这个搜索树，和方法一的搜索树对比，我们发现，两者特性截然相反，该搜索树从根开始，“分叉”越来越少。

两种搜索对象及策略是完全不同的，搜索树特性又截然相反。从宏观上，搜索树上节点多少，两者相差无几。如何抉择呢？这时就要从微观上比较：

● 比较哪个搜索对象和策略更优

既然是比较，就要有比较的标准。这里，确定了两个标准：

谁易于优化剪枝　；　谁的操作量小

★关于**谁易于优化剪枝**的比较：

本题的主要剪枝有三种，如下逐一分析。

◇ 可行性剪枝

当路线的**特征**确定了，就可以判断该路线是否成立。

方法一，搜索中，每层枚举当前车是哪条路线的第二辆车，都要用到该判断；方法二，每层是确定一条路线，也用到该判断。关键是，根据两者搜索树，方法一一旦剪枝，将剪去的是一大片“茂盛”的树枝，显然，相比之下，方法二剪枝的效果就差了许多。

故在此剪枝应用上，方法二比方法一逊色许多。

◇ 与已知最优解比较剪枝

这就要看谁能很快找到解了。显然，由于方法一可行性剪枝的优点，每次剪枝都能删去**很多**的不可行的节点，找到解的速度就不比方法二慢了。

此剪枝，方法二不比方法一要好。

◇ 排除重复剪枝

注意到题目中**时间**这个关键要素的范围为 0~59，而车辆数目可达 300，说明，在同一时间到达的车辆数目很多！前面那个简单例子中，就出现了两个 14，而到达时间为 14 的两辆车各属于那条路线是等效的，这就有重复。

方法一对于同时到达的且未确定归属的车，若编号小的车为某路线的第一辆车，则编号大的车为也必为一条路线的第一辆车。

方法二，对到达时间相同的且未确定归属的车，固定只选编号最小的车为第二辆车。

两者剪去的都是重复的枝，所以，效果是一样的，故此剪枝上，双方平分秋色。

结论：由于方法一搜索树的良好特性，使得方法一在剪枝优化方面前景更广阔。

★关于**谁的操作量小**的比较：

操作量是“主递归程序操作量”的简称，由主递归程序的枚举循环和剪枝函数决定的。

◇主递归程序的枚举循环

方法一，每层利用循环来枚举一辆车是属于新路线的第一辆车还是已知路线的第二辆车，而且搜索树上这个循环枚举量是由未确定“第二辆车”的线路数目决定的，最大为 17。

方法二，每层枚举哪辆车是第二辆车。由于用了排除重复剪枝，这个循环量最大为 60。

直观上看两者的最大界限就已知道，方法二不比方法一好。

◇剪枝函数消耗时间

由于本题特殊性，主要剪枝函数基本上差异不大，消耗时间也差不多。

结论：操作量大小方面，方法二不比方法一好。

最终结论：

通过以上微观理论分析，对两种方法进行了比较，得出最终结论是：方法一比方法二好！选定了搜索对象和策略，刚才又分析了实现中的主要问题：如何剪枝，编写程序自然就得心应手了。

事实也证明了我们的结论。两种程序运行比较如下，可以发现两者优劣差异巨大。

	car.in1 K=3	car.in2 K=5	car.in3 K=8	car.in4 K=10	car.in5 K=15	car.in6 K=17
方法一用时	0.01s	0.01s	0.05s	0.11s	1.48s	1.76s
方法二用时	0.01s	0.01s	9.07s	>100s	>100s	>100s

总　　结

就本题而言，从宏观上看，很难知道两种方法效率的差异，为什么方法一更好呢？关键原因在于：它适合程序上的**剪枝优化**，且**操作量小**。

那为什么选择这两个方面为标准呢？

我们知道：

深度搜索消耗时间 ≈ 每个节点操作系数 × 节点个数

从上面一个公式，我们很显然地能从微观上看出，要减少消耗时间，

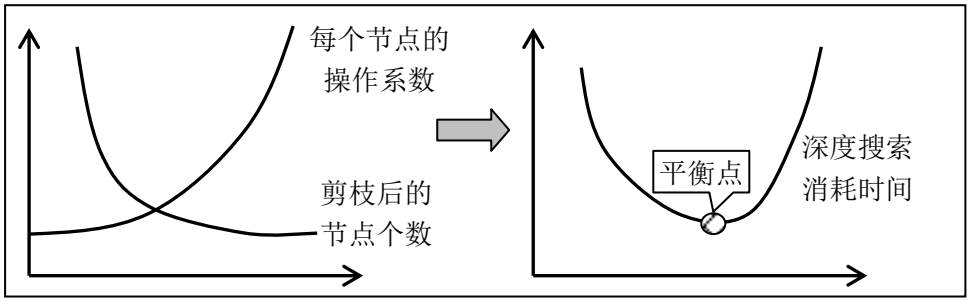
一是减少节点个数——这就是我们所说地**剪枝优化**；

二是减少每个节点的操作系数——即刚才分析的**程序操作量**。

为了提高搜索效率，根据这个公式，我们往往在以上两方面反复进行改进。殊不知，**从宏观前提上，如何才能使我们“可以、充分、有效”剪枝，如何才能使我们“可以、充分、有效”降低程序操作量也都是很重要的！**

于是，搜索对象和策略为剪枝，降低操作系数创造前提条件的好坏就成了我们的标准！

但是，在以这两方面为标准比较的时候，我们要注意到：这两个标准紧密关联，要剪枝多，就要有好的复杂的剪枝函数，但这就增大了每个节点操作系数。如图所示：



两者在目的上是统一的，效果上却是对立的。在以这两者为标准的时候，要把握好“如何协调，找准两者**平衡点**”。

总的来说，对深度搜索题目，一个好的搜索对象和策略是十分重要的。本文通过对“汽车问题”的分析，充分说明了这点，并且根据**深度搜索消耗时间公式**提出了比较搜索对象和策略的标准：优化剪枝与操作系数。

同时，对**深度搜索消耗时间公式**的分析也启发我们，**为了更好的解决问题达到目的，仅仅在微观上进行变动更新是不够的，还要首先为这个目的去创造良好的宏观条件！**

好的搜索对象和策略正是成功解搜索题的宏观条件，这也是本文的主旨。