第24卷 增刊2
2007年12月
工 程 数 学 学 报
CHINESE JOURNAL OF ENGINEERING MATHEMATICS
Vol. 24 Supp.2
Dec. 2007

文章编号:1005-3085(2007)08-0117-04

公交线路选择模型

蔡志杰[1]， 丁颂康[2]
(1- 复旦大学数学科学学院，上海 200433； 2- 上海海事大学基础部，上海 200135)

摘 要： 本文讨论公交线路选择模型，介绍集合求交方法及与之对应的搜索方法。同时给出相应的图论模型，讨论了它们的算法复杂性。

关键词： 集合求交算法；搜索方法；图论模型；算法复杂性

分类号： AMS(2000) 90B20 **中图分类号：** O157.6 **文献标识码：** A

1 问题的提出

今年全国大学生数学建模竞赛B题以奥运为背景，考虑公交线路的选择问题，是一道贴合生活实际的赛题。在评阅试卷时，我们发现很多参赛队采用了“集合求交”的方法，在描述了该方法的总体思路后，绝大多数参赛队给出了搜索算法求解这一问题。但是由于搜索算法的计算量很大，这些参赛队大多限制乘客的换乘次数在2次以内，有的参赛队还给出限制2次换乘的心理因素或相关的调查结果。然而在实际的交通网络中，即使像北京这样公交线路比较完善的大城市，也会有需要换乘3次或3次以上的站点，因此2次换乘这一约束条件会导致在某些情况下出现无解的结果，这是不合理的。那么，在没有换乘限制的约束条件下，集合求交算法是不是不能用了呢？其算法复杂性是不是非多项式的？本文主要讨论这种方法的算法复杂性及其相应的搜索方法和图论模型[1-3]。最后说明采用搜索算法，可以在很短的时间内求出最优解。

2 集合求交算法

首先引入一些记号，记 N 为公交站点总数(约为4000)，L 为公交线路(包括公汽和地铁)总数(约为500)，Δ_l 为经过每个车站的公交线路数(平均约为15)，Δ_s 为每条线路上的车站数(平均约为40)。将所有公交线路视为单行线，双行线拆分成两条单行线，环线也视为两条单行线。

记 p 为起始站，q 为终点站。集合求交算法可以描述如下：

令 $L_p^{(0)}=\{l \mid p$ 在公交线路 l 上$\}$，$L_q^{(0)}=\{l \mid q$ 在公交线路 l 上$\}$，则 $L_p^{(0)}\cap L_q^{(0)}$ 表示 p，q 在同一条线路上的公交线路集合。若公交线路 l 上的站点从起点到终点按次序依次记为 $s_1,s_2,\cdots,s_m$，则 $A^{(0)}=\{l \mid l\in L_p^{(0)}\cap L_q^{(0)}, p=s_i\in l, q=s_j\in l, i<j\}$ 即为从站点 p 可直达站点 q 的公交线路集合。值得提及的是，绝大多数采用此方法的参赛队都没有考虑 p，q 在公交线路上的次序，而直接将 $L_p^{(0)}\cap L_q^{(0)}$ 作为 p，q 直达线路的集合。如果所有的公交线路都是双行线，这是可以的，否则需要考虑站点之间的次序，即考虑公交线路行驶的方向。

如果 $A^{(0)}=\emptyset$，则 p，q 之间没有直达线路，需要考虑一次换乘。令 $S_p^{(1)}=\{s \mid l\in L_p^{(0)}, p=s_i\in l, s=s_j\in l, i<j\}$ 为 p 可直达的站点集合，$S_q^{(1)}=\{s \mid l\in L_q^{(0)}, s=s_i\in l, q=s_j\in l, i<j\}$ 为可直达 q 的站点集合，$B^{(1)}=S_p^{(1)}\cap S_q^{(1)}$ 为从 p 经一次换乘可到达 q 的转乘站点集合。

如果 $B^{(1)} = \emptyset$，则 p 经一次换乘无法到达 q，继续考虑二次换乘。令 $L_p^{(2)} = \{l \mid s$ 在公交线路 l 上，$s \in S_p^{(1)}\}$，$L_q^{(2)} = \{l \mid s$ 在公交线路 l 上，$s \in S_q^{(1)}\}$，则 $A^{(2)} = \{l \mid l \in L_p^{(2)} \cap L_q^{(2)}, s = s_i \in l, t = s_j \in l, i < j\}$ 为从 p 经二次换乘到达 q 的第二条公交线路集合。若 $A^{(2)} \neq \emptyset$，则从 p 转乘二次公交线路可到达 q；而当 $A^{(2)} = \emptyset$ 时，继续这一过程，可得到转乘 k 次到达 q 的全部线路。

但是，这种描述方式较为复杂，特别是换乘次数超过2次时，因此大多数参赛队都增加了最多换乘2次的约束条件。让我们分析一下集合求交算法的算法复杂性。

过 p 点的公交线路数平均为 $\#L_p^{(0)} = \Delta_l$ 条，过 q 点的公交线路数平均也有 $\#L_q^{(0)} = \Delta_l$ 条，因此求出直达线路集合 $A^{(0)}$ 的计算量为 $O(\Delta_l^2)$。

对一次换乘 $B^{(1)}$，从 p 出发可直达的站点数平均为 $\#S_p^{(1)} = \Delta_l\Delta_s$，可直达 q 的站点数平均也是 $\#S_q^{(1)} = \Delta_l\Delta_s$，故求出一次换乘 $B^{(1)}$ 的计算量为 $O((\Delta_l\Delta_s)^2)$。

对两次换乘 $A^{(2)}$，$\#L_p^{(2)} = \#S_p^{(1)} \cdot \Delta_l = \Delta_l^2\Delta_s$，$\#L_q^{(2)} = \Delta_l^2\Delta_s$，故求出两次换乘 $A^{(2)}$ 的计算量为 $O((\Delta_l^2\Delta_s)^2)$。

对一般的 k 次换乘，虽然算法描述很复杂，但其算法复杂性容易由上面的推导递推给出，为 $O((\Delta_l^{[\frac{k}{2}]+1}\Delta_s^{[\frac{k+1}{2}]})^2)$。

3 搜索算法

上一节我们给出了集合求交的搜索算法，方法是从起点 p 和终点 q 分别开始搜索，寻找公共的线路或站点。我们也可以从起点 p (或终点 q) 一端开始进行搜索，直到搜索到终点 q (或起点 p) 为止。这实际上就是直接搜索算法。具体算法为(不妨设从起点 p 出发开始搜索)：

记 $L(s) = \{l \mid$ 站点 s 在线路 l 上$\}$ 表示经过站点 s 的所有公交线路组成的集合，$S(l) = \{s \mid$ 站点 s 在线路 l 上$\}$表示公交线路 l 上所有站点组成的集合。

令 $L^{(0)} = L(p)$，$S^{(0)} = \{s \mid s \in S(l), l \in L^{(0)}\}$。$S^{(0)}$ 表示与 p 在同一条线路上的站点集合。为描述站点之间的先后次序，引入集合 $D(s) = \{t \mid s = s_i \in l, t = s_j \in l, l \in L(s)$，且 $i < j\}$ 表示从 s 可以直达的站点集合。由此定义，令 $D^{(0)} = D(p)$，则当 $q \in D^{(0)}$ 时，从 p 出发无需经过换乘即可直达终点 q。

若 $q \notin D^{(0)}$，则从 p 出发不能直达 q，需考虑一次换乘，令 $L^{(1)} = \{l \mid l \in L(s), s \in D^{(0)}\}$，$S^{(1)} = \{s \mid s \in S(l), l \in L^{(1)}\}$，$D^{(1)} = \{s \mid s \in D(t), t \in D^{(0)}\}$。若 $q \in D^{(1)}$，则从 p 经一次换乘可到达 q。

一般地，令 $L^{(k)} = \{l \mid l \in L(s), s \in D^{(k-1)}\}$，$S^{(k)} = \{s \mid s \in S(l), l \in L^{(k)}\}$，$D^{(k)} = \{s \mid s \in D(t), t \in D^{(k-1)}\}$。当 $q \notin D^{(j)}$ $(j = 1, 2, \cdots, k-1)$，而 $q \in D^{(k)}$ 时，从 p 出发需经 k 次换乘才能到达终点 q。

类似于上一节的分析可以得到 k 次换乘时直接搜索算法的复杂性为 $O((\Delta_l\Delta_s)^{k+1})$。

直接搜索算法与上一节给出的集合求交搜索算法本质上是一致的，当换乘次数较少时，这是多项式算法。但当 k 稍大一些，计算量将以指数形式增长。特别地，要得到全部解，最坏的情况计算量将达到 $O((\Delta_l\Delta_s)^L)$，其中 L 为公交线路总数。

4 搜索算法的图论模型

直接搜索算法可以等价地归结为二分图上修正的最短路问题。

首先由公交线路和公交站点构造一个二分图 $G = (V, E)$，其中 $V = S \cup L$ 为顶点集合，$S = \{s_1, s_2, \cdots\}$ 为所有公交站点组成的集合，$L = \{l_1, l_2, \cdots\}$ 为所有公交线路组成的集合，$E = \{e_1, e_2, \cdots\}$ 为边集，当且仅当公交站点 s 在公交线路 l 上时，s 与 l 之间有一条边。

这样 S 与 L 构成了二分图的两类顶点，$s\in S$ 的邻集是该站点可以乘坐的所有公交线路组成的集合，$l\in L$ 的邻集是在同一条公交线路上的所有站点组成的集合。

容易看到，站点 p 与 q 的邻集有交，当且仅当在二分图 G 中存在从点 p 到 q 的长度(即边数) 为 2 的路。由于 G 为二分图，与 p 和 q 相邻的顶点为 $l\in L$，这表明存在某条线路 l 同时经过 p 和 q。但是从 p 到 q 的路线必须满足一定的次序，即在线路 l 上，p 在前，q 在后。为此，必须在二分图 G 的边上增加一个标记 λ。

设边 $e=(s,l)$，s 是线路 l 上第 k 个站点，则相应的标记 $\lambda=k$。这样，从 p 无需换乘可直达 q 的充分必要条件是，存在从 p 到 q 的长度为 2 的路 $Q=pe_{j_1}le_{j_2}q$，且 $\lambda_{j_1}<\lambda_{j_2}$，其中 l 为直达公交线路，$\lambda_{j_2}-\lambda_{j_1}$ 为乘车站数。

类似地，从 p 至少需要经过一次换乘才能到达 q 的充分必要条件是，存在从 p 到 q 的最短长度为 4 的路 $Q=pe_{j_1}l_{k_1}e_{j_2}se_{j_3}l_{k_2}e_{j_4}q$，且 $\lambda_{j_1}<\lambda_{j_2}$，$\lambda_{j_3}<\lambda_{j_4}$，其中 l_{k_1}，l_{k_2} 为乘坐的两辆公交车，$\lambda_{j_2}-\lambda_{j_1}$ 和 $\lambda_{j_4}-\lambda_{j_3}$ 分别为两条线路上乘坐的站数，s 为换乘站点。

以此类推，如果从 p 到 q 的最短路长度为 $2m$，记为 $Q=pe_{j_1}l_{k_1}e_{j_2}s_{i_1}e_{j_3}l_{k_2}e_{j_4}s_{i_2}\cdots s_{i_{m-1}}e_{j_{2m-1}}l_{k_m}e_{j_{2m}}q$，且 $\lambda_{j_{2t-1}}<\lambda_{j_{2t}}$ $(t=1,2,\cdots,m)$，则从 p 至少需要经过 $m-1$ 次换乘(即乘坐 m 辆公交车)才能到达 q，所乘坐的 m 条公交线路为 l_{k_t} $(t=1,2,\cdots,m)$，乘坐站数为 $\lambda_{j_{2t}}-\lambda_{j_{2t-1}}$ $(t=1,2,\cdots,m)$，换乘站点为 s_{i_t} $(t=1,2,\cdots,q-1)$。由此，我们将集合求交方法转换成二分图 G 上修正的最短路问题。

由于有标识条件的存在，通常所用的最短路算法，如 Dijkstra 算法和 Floyd & Warshall 算法，在此都将失效。为此，我们给出以下修正的最短路算法。

记 $d(v)$ 为从起点 p 到顶点 v 的最短路长度，S_i 为从起点 p 经 i 次转乘所能到达的站点集合，$N(v)$ 为顶点 v 的邻集，T_i 为从起点 p 第 i 次转乘的线路集合。

第 1 步 置 $d(p)=0$，对于所有其它点 v，置 $d(v)=\infty$。记 $S_0=\{p\}$，$i=0$。

第 2 步 置 $T_i=\emptyset$。检查所有满足条件 $u\in S_i$，$v\in N(S_i)\subset L$ 的边 $uv\in E$。如果存在这种边 uv，则令 $d(v)=d(u)+1$，$T_i:=T_i\cup\{v\}$，并记 $\lambda(v)=\lambda_{uv}$。

第 3 步 置 $S_{i+1}=\emptyset$。检查所有满足条件 $v\in T_i$，$u\in N(T_i)\subset S$ 的边 $vu\in E$。如果存在这种边 vu，并且满足 $\lambda_{vu}>\lambda(v)$，则令 $d(u)=d(v)+1$，$S_{i+1}:=S_{i+1}\cup\{u\}$。

第 4 步 如果 $S_{i+1}=S$，则算法终止。否则令 $L:=L\backslash T_i$，$S:=S\backslash S_i$，$i:=i+1$，转第 2 步。

容易看出，算法终止时的指标 i 就是整个公交系统中，从 p 到其余公交站点需要换乘公交线路的最大次数。

5 关于搜索算法的进一步分析

第 3 节我们指出搜索算法是多项式算法。但当换乘次数较大时，计算量仍然很大。而图论算法与搜索算法是一致的，因此当换乘次数较大时，计算量也很大。

那么，能否减少计算量，使得搜索算法仍然有效呢？事实上，如果某个站点(如站点 s) 已在从 p 出发的某条线路(如线路 l) 上，即 $s\in D^{(0)}$，那么在 $D^{(1)}$ 中如果再次出现站点 s 时，这个站点就不需要再考虑了。同样地，如果 $s\in\bigcup_{i=0}^{k}D^{(i)}$，那么在 $D^{(k+1)}$ 中就无需再考虑这个站点 s 了。

对于公交线路也是如此。若某线路 $l\in\bigcup_{i=1}^{k}L^{(i)}$，则在 $L^{(k+1)}$ 中无需再考虑这条线路 l 了。

相应地，在算法中只要将 $L^{(k)}$ 和 $D^{(k)}$ 修改为 $\bar{L}^{(k)}=L^{(k)}\backslash\bigcup_{i=0}^{k-1}L^{(i)},\bar{D}^{(k)}=D^{(k)}\backslash\bigcup_{i=0}^{k-1}D^{(i)}$ 就可以了。搜索时，采用广度优先的搜索方法。在编制程序时，只要设置两个数组，分别记录各站点和线路的处理情况。初始都为 -1，当某条线路出现在 $L^{(k)}$ 时，相应的线路数组元素赋

值为 k；当某个站点出现在 $D^{(k)}$ 时，相应的站点数组元素也赋值为 k。在同一层搜索时，如果发现某条线路在数组中的值为 $j\ (0 \le j < k)$，则此条线路不予考虑；同样地，如果发现某个站点在数组中的值为 $j\ (0 \le j < k)$，则此站点也不予考虑。这样，在搜索算法中，总的搜索量仅为 $O(NL)$，其中 N 为站点总数，L 为线路总数。计算量是非常小的。

同样地，在图论模型中，我们已经考虑了这种处理，即在第4步中，$L := L \backslash T_i$ 就是在线路集合 L 中将已经处理过的线路 T_i 去掉，$S := S \backslash S_i$ 就是在站点集合 S 中将已经处理过的站点 S_i 去掉，从而在下一次迭代时，L 和 S 的规模都将缩小，达到减少计算量的目的。

6 其他目标的处理

以上讨论只涉及“转乘次数最少”单个目标。在实际情况中，乘客往往还会考虑“乘车时间最短”及“费用最少”等因素。根据实际情况，我们以换乘次数作为第一优先目标，乘车时间(或费用)作为第二优先目标，费用(或乘车时间)作为第三优先目标进行处理。这样，在图论模型中，乘车时间和费用可作为权系数引入，分别记 w_t 和 w_f。

为明确起见，我们考虑乘车时间为第二目标，费用为第三目标。记 $T_{\min}(v)$ 为从起点 p 到站点 v 的最短乘车时间，$F_{\min}(v)$ 为乘车时间为 $T_{\min}(v)$ 时的最少乘车费用。初始时，对所有的 $v \ne p$ 置 $T_{\min}(v) = \infty$，$F_{\min}(v) = \infty$。在搜索过程中，在同一级搜索中，对可达的站点分别计算相应的乘车时间 $T(v)$ 和费用 $F(v)$，如果 $T(v) < T_{\min}(v)$ 或者 $T(v) = T_{\min}(v)$ 且 $F(v) < F_{\min}(v)$，则用新线路替代原线路，并令 $T_{\min}(v) := T(v)$，$F_{\min}(v) := F(v)$。整个搜索过程无需作大的改动即可达到目的，算法复杂性仍是 $O(NL)$。

在第2节中，我们指出考虑公交线路行驶方向是必要的，因此在图论模型中引入了标记 λ，用来记录站点在线路上的位置信息。正是因为 λ 的存在，使得常用的最短路算法失效，从而导致计算量的增加。如果公交线路均为双行线，那么 λ 就不必引入了。这样，搜索算法就与一个标准的二分图的最短路问题等价，从而可以使用诸如 Dijkstra 算法或 Floyd & Warshall 算法等经典方法进行搜索计算。这是一个多项式算法，且算法复杂性不随换乘次数的增加而增加。

参考文献：

[1] Lawler E. Combinatorial Optimization: Network and Matroids[M]. New York: Holt, Rinehart and Winston, 1976

[2] Papadimitriou C H, Steiglitz K. Combinatorial Optimization: Algorithms and Complexity[M]. Prentic-Hall, Inc, Englewood Cliffs, N. J., 1982

[3] Bondy J A, Murty U S R. Graph Theory with Applications[M]. London: MacMillan Press, 1976

Models for the Public Transportation Line Selection

CAI Zhi-jie[1], DING Song-kang[2]

(1- School of Mathematical Sciences, Fudan University, Shanghai 200433;
2- Department of Foundation Shanghai Maritime University, Shanghai 200135)

Abstract: In this paper, we discuss models for the public transportation line selection and introduce set intersection algorithm and corresponding searching method. We also give a graph theory model, and discuss the complexities of algorithm of these models.

Keywords: set intersection algorithm; searching method; graph theory model; complexity of algorithm