多约束条件路径规划方案设计

一、概述

在该多约束条件路径规划问题中，最优路径或是无解时的参考路径中所有必经节点（包括起始点、终止点、必经点、必经边端点）必然是按照一定顺序排列的。本算法通过寻找一条总跳数（经过的节点数目）最小的所有必经节点顺序链，进而找到最优解或者次优解。

二、方案设计

整体方案主要流程如框图 1 所示。首先，我们根据输入文件将图的相关信息保存到起来。然后，对图进行规模压缩，得到任意两个必经节点间的跳数最小的路径的相关信息，并保存起来。接着，我们初始化了一条可插入边链，使用双向邻近节省法将必经点和必经边按照一定的规律选择最优位置插入到源节点 S 和目的节点 E 之间，得到一条总跳数较小的所有必经节点顺序链。利用 2opt 算法对这条所有必经节点顺序链进行优化，使得路径总跳数进一步减小。最后，比较 2opt 优化后总跳数与阈值的关系，输出最优路径或者是在无解时将跳数、必经边、必经点这三个约束条件按不同优先级排序的参考路径。其中，双向临近节省插入法是本算法的核心。

图1

2.1 相关词语的定义解释

① 阈值：约束条件中给出的允许路径经过的最大节点数目；

② 跳数：点 a 到点 b 的跳数，是指一条以 a，b 作为起点和终点的路径中，所经过的所有节点的数目（包括点 a、b）；

③ 必经节点：包括起始节点，终止节点，必经点，必经边的端点；

④ 所有必经节点顺序链：一条经过所有必经点、必经边，以起始节点作为源点，终止节点作为终点的路径（只包含所有必经节点）。我们只关注这条路径任意两个必经节点间的跳数和必经节点排列的顺序，而不关注到底经过了哪些节点。

2.2 拓扑图相关信息存储设计

① 读取测试文件，以邻接表的形式存储该拓扑图。

② 创建必经边链表，保存必经边。

③ 创建必经点链表，保存必经点。

④ 创建所有必经节点（包括起始节点，终止节点，必经点和必经边的端点）集合。

⑤ 在邻接表中删除禁止边。

2.3 规模压缩

本算法核心在于得到一条总跳数最小的所有必经节点顺序链表，为方便快捷的得到经过任意两个必经节点所需的最小跳数，需将 2.2 中得到的原始数据进行提炼。结合广度优先和按层遍历的思想，将存储了整个拓扑图的相关信息的邻接表转化为只存储任意两个必经节点间的最小跳数(经过的节点数)、跳数等于该最小跳数的路径数目、相应的路径及路径花费的二维矩阵 pPathMatrix，数据结构如图 2 所示。

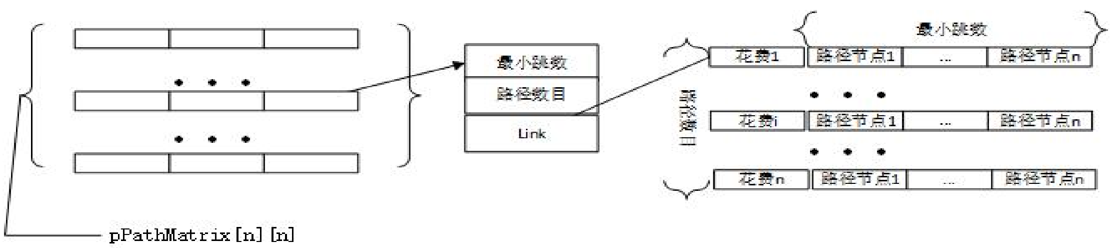


图2

因为算法后面可能会反复使用任意两个必经节点的跳数的相关数据，将数据存储起来可以减少计算时间。规模压缩的时间复杂度是 $O\left(m^2\ (n+e)\right)$，其中 m 为所有必经节点的个数，n 为图的顶点个数，e 为图的边的个数。

2.4 双向邻近节省插入法

双向邻近节省插入算法的目的是根据规模压缩得到的二维结构体矩阵 pPathMatrix 中的相关信息，得到一条经过所有必经点，必经边且总跳数较小的路径。此时，我们只关注这条路径的跳数，不关注这条路径经过的非必经节点具体是哪些节点。因此，这个问题可以转化为一笔画问题：在一个顶点为所有必经节点，边的权重为两个必经节点间的最小跳数的全联通无向图中，如何找到一条经过所有顶点且花费较小的路径？我们可以通过这样的方法找到：构建一条从起始节点指向终止节点的路径，每次选择一个必经点或者必经边插入到路径中，插入时选择合适的位置使路径花费的增加尽可能的小，直到将所有的必经点和必经边都插入。算法的时间复杂度：$O\left((m+k)^2\right)$，其中 m 为必经节点的个数，k 为必经边的个数。算法具体实现流程如图 3 所示：

① 初始化可插入边链，将起始节点与终止节点作为第一条可插入边加入到可插入边链中。

② 每次选择一个需要插入的必经边或者必经点：

在未被插入的必经点中选择一个与 no 节点之间跳数最小的点 Node, 并记录其最小跳数 hopNode;

在未被插入的必经边中选择一个与 no 节点之间跳数最小的边 Edge,并记录其最小跳数 hopEdge;

比较 hopNode 与 hopEdge 的值，若 hopNode 较小则插入点 Node，反之则插入边 Edge。

③ 寻找最优插入位置：

使用节省插入法将选择的需要插入的必经边或者必经点插入到路径中。

节省插入法：比较将必经点或必经边插入到每条可插入边后路径增加的跳数，选择跳数增加最小的一条可插入边，将必经点或者必经边插入插入。图 5 示意了将边 c-d 或点 c 插入可插入边 a-b 时路径增加的跳数和可插入边链的更新。

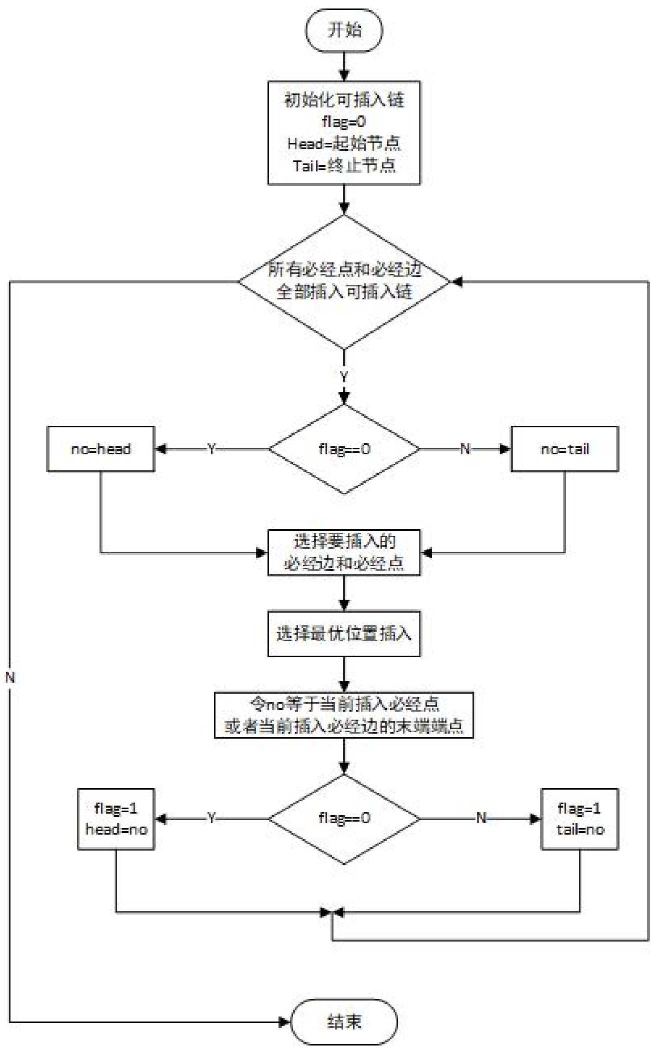


图 3、双向邻近节省插入法流程图

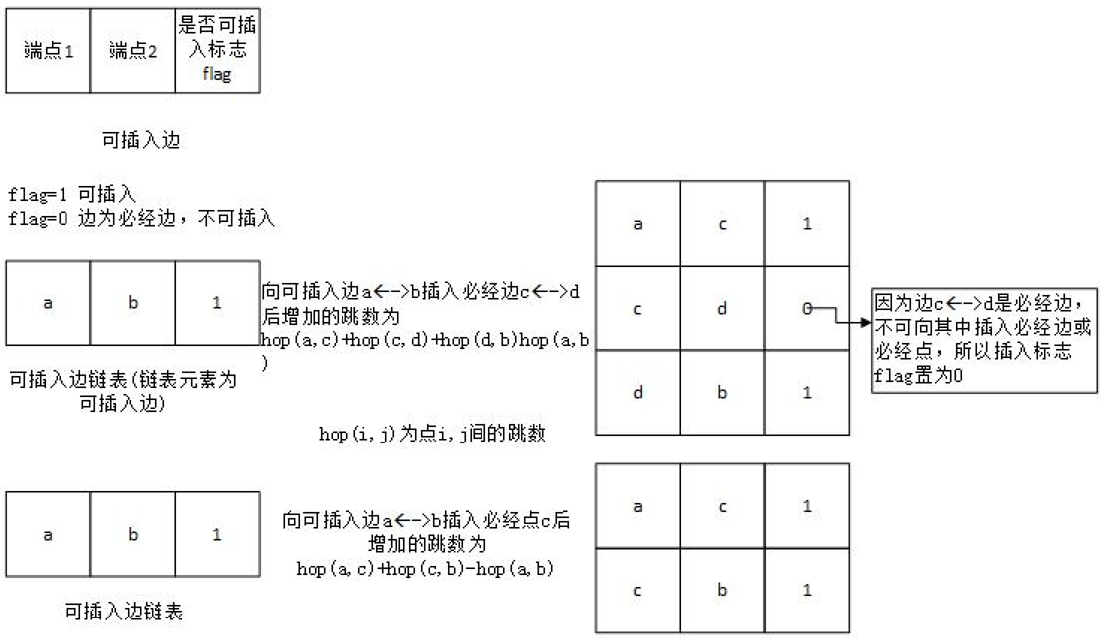


图5

2.5 2-opt 优化

通过以上步骤，我们得到了一条所有必经节点的顺序链，利用 2opt 优化算法对其进行跳数上的优化。首先对任意两个必经节点 i、k 进行交换，交换过程如下：

取 i = 4, k = 7;

原顺序链: A ==> B ==> C ==> D ==> E ==> F ==>G ==> H ==>I;

交换 i,k 后：

新顺序链: A ==> B ==> C==> (G ==> F ==> E ==> D)==>H==>I

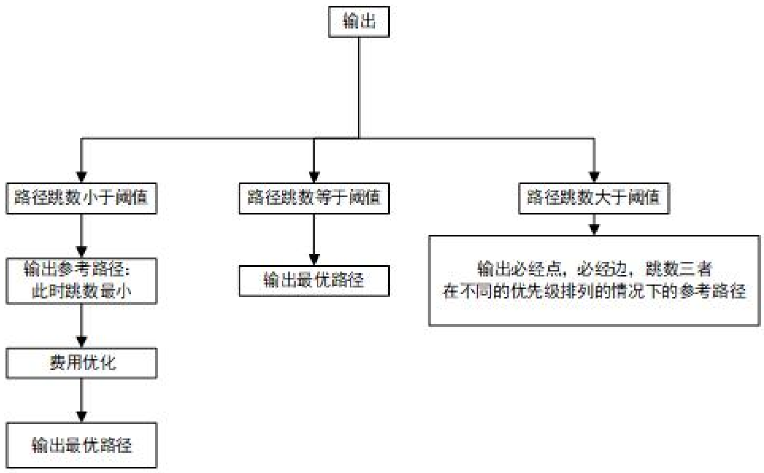
若 C==>D,G==>H 不是必经边，且交换后的路径总跳数减小，则用新路径来替换原路径。否则，保持原路径不变。

时间复杂度：$O\left((m+k)^2\right)$ m 为必经节点的个数，k 为必经边的个数

2.6 输出路径

将完成 2opt 优化的所有必经节点顺序链根据规模压缩后得到的二维矩阵中的相关信息转化为一条完整的路径。若遇到两条必经节点间跳数最小的路径不止一条，选择花费最小的路径。得到的该完整路径是所有经过全部的必经点、必经边并且跳数最小的路径中，花费最小的路径。

当路径跳数等于阈值时，这条路径就是最优解。当路径跳数小于阈值时，题目有解。首先输出该路径，该路径是在跳数最小的条件下花费最小的路径，继而在着重考虑费用的情况下，通过费用优化算法将它优化为花费最小且跳数不超过阈值的最优解。



当路径跳数大于阈值时：按照优先级排列的情况的不同，输出参考路径。

① 当跳数<必经点<必经边、跳数<必经边<必经点时，我们不对 2opt 优化后得到的完整路径进行修改，直接输出。此时的路径是当前图所有经过必经点和必经边的路径中，在跳数最小条件下花费最小的路径。

② 当跳数>必经点=必经边时，我们选择放弃经过一些必经边、必经点。此时的路径是当前图所有跳数满足小于阈值的路径中，经过的必经点和必经边最多的。

③ 当跳数>必经点>必经边时，我们首先选择放弃经过一些必经边使得跳数满足阈值约束条件，若放弃所有的必经边后得到的路径跳数仍然大于阈值，我们继续选择放弃经过一些必经点使路径跳数进一步减小以满足阈值约束条件。

④ 当跳数>必经边>必经点时，我们首先选择放弃经过一些必经点使得跳数满足阈值约束条件，若放弃所有的必经点后得到的路径的跳数仍然大于阈值，我们继续选择放弃经过一些必经边。

⑤ 当必经点>跳数>必经边时，我们选择放弃经过一些必经边。此时的路径是当前图所有经过所有的必经点的路径中，跳数最小且经过的必经边最多的。

⑥ 当必经边>跳数>必经点时，我们选择放弃经过一些必经点。此时的路径是当前图所有经过所有的必经边的路径中，跳数最小且经过的必经点最多的。

2.6.1 费用优化：

费用优化算法是在路径跳数不大于阈值的条件下，通过插入一些节点进行松弛操作的方式达到减少整条路径花费的目的。2opt 操作之后得到的路径是在路径跳数最小情况下花费最小的路径，因此若想进一步减小路径花费，必然需要插入一些节点来松弛路径中花费较大的边。我们希望每增加一个节点，减少的花费都是当前情况下减少的最多的。而花费越大的边，减小的路径花费越有可能大。因此，我们先选定一定数目花费最大的边，以替换性价比(替换减小的花费与增加跳数的比值)由大到小的顺序松弛这些花费较大边，直到下次增加节点跳数会超过阈值为止。具体过程如下：

选择权值较大的需要替换的边：为了保证余量，在 2-opt 优化后的顺序链中选取 2*K 条权值最大的边并保存在最大堆数组中。其中 K 值计算式如下：

$$K = \frac{N}{1.8}\left(N = 阈值 - 路径跳数\right)$$

计算公式说明：

我们选取任意 20 张图，测试统计替换权值较大的边时替换性价比最高时插入节点的个数。在 40%的情况下，替换路径只增加了 1 个节点；40%的情况下，替换路径增加了 2 个节点；15%的情况下，替换路径增加了 3 个节点；5%的情况下，替换路径增加的节点数>3 个。

在忽视小概率情况下，假设完成费用优化需要替换的边的个数 K，则其中 40%的边替换时路径跳数会增加 1；40%的边替换时路径跳数会增加 2；20%的边替换时路径跳数会增加 3。

显然：$N \le 0.4K + 0.4K \times 2 + 0.2K \times 3 \Rightarrow K \ge N/1.8$

在替换依次替换这些花费较大的边时，我们需要一个标准来排列这些边的替换顺序。这个标准就是性价比，公式如下：

$$性价比 = \frac{减少的花费}{增加的跳数}$$

分别计算出最大堆数组中保存的所有花费较大的边插入 1，2，3 个节点进行松弛操作的性价比，保存性价比最高计算结果（性价比、减小的花费。增加的跳数）。将最大堆数组中的边按性价比从大到小排序。在替换后路径总跳数不超过阈值的前提下，每次选择一个性比价最高的边,用花费较小的路径代替这条边在原来路径中的位置。在具体替代过程中需要同双向邻近节省插入法和 2opt 算法，保证必经边不被打破。

2.6.2 根据约束条件优先级，选择放弃经过一些必经边、必经点

在没有最优解（经过所有必经点、必经边的路径最小跳数大于阈值）的情况下，我们根据约束条件的优先级选择舍弃一些约束条件寻找次优解。当跳数的优先级小于必经点、必经边时，上述总跳数最小的路径即为次优解。当跳数的优先级大于必经边或者必经点的优先级时，我们需要通过选择放弃经过一些必经边或者必经点使得新得到的路径的总条数小于等于阈值。在选择放弃经过的必经边或者必经点的时候希望每次的选择都能够使路径跳数的减少最大。因此，引入收益（舍弃该必经点、必经边路径跳数的减小）作为标准来选择这些放弃经过的必经点和必经边。将在双向邻近节省插入某必经点或必经边使路径跳数的增量和在 2opt 优化后最小跳数路径中删除该必经点或必经边路径跳数的减少量作为收益的两个影响因素，即：$\text{收益} = \text{插入时增加的跳数} + \text{删去时减少的跳数}$，我们优先选择收益较高的必经边或必经边放弃经过。具体过程如下：

计算 N，N=路径跳数-阈值。此时，程序根据优先级，每次选择成本最高的必经点或者必经边放弃加入到路径中，直到选择的必经点或者必经边的收益综合大于二倍的 N，保证之后得到的新路径的跳数一定小于阈值。

利用双向邻近节省插入法和 2opt 算法得到一条经过剩余的必经边和必经点的路径。为了防止之前放弃经过的必经点或者必经边过多，在得到新的完整路径后，我们尝试重新向路径中插入已经删除的收益较低的必经点或者必经边，将路径跳数调整为最接近阈值时为止。

三、测试及分析

1、不同插入算法的效果比较测试

表 1 的数据是经过不同的插入方法得到经过所有必经点和必经边路径的跳数。具体数值是利用我们自己编写的测试用例生成软件分别取顶点数为 100/300/500 的测试用例 20 组，求得的平均值。显然，双向邻近节省插入法+2opt 优化得到的路径的跳数最小，并且 2opt 的优化效果随顶点个数的增加明显的增大。

表 1、不同插入算法的效果比较

跳数 / 顶点个数	双向邻近节省插入+2opt 优化	双向临近节省插入	临近插入	节省插入（先边后点）	节省插入法（先点后边）	节省插入法（点边交替）
100	27	27.7	31.7	29.15	28.65	28.65
300	124.1	128.14	157.95	131.9	134.9	134.9
500	281.42	300.85	383.81	306.71	312.57	311.85

2、费用优化测试

我们利用自己编写的测试用例生成软件生成测试用例，每组测试 20 次，因为篇幅的影响，截取其中的 3 次。从表 2 的数据可以看出，费用优化确实能够实现在路径条数不超过阈值的前提下，对路径的花费尽可能的减小的作用。

表 2、费用优化测试效果对比

顶点个数	阈值	优化前跳数	优化前花费	优化后跳数	优化后花费
100	46	30	1252	46	959
	46	27	1420	42	1148
	37	31	1205	36	1088
300	322	127	6396	220	4910
	262	120	5639	194	4568
	268	124	6036	203	4867
500	783	288	13984	456	10672
	759	285	14298	480	11219
	510	284	14595	462	11450

3、输出次优解参考路径测试

表 3 的数据为次优解参考路径测试输出的数据，结果显示，本算法确实能够根据优先级的不同，删除一些必经点和必经边使路径跳数小于等于阈值。

表 3、次优解参考路径测试

跳数	路径跳数	阈值	不经过的必经点数	不经过的必经边数	花费
必经点<必经边<跳数	203	207	24	39	9686
	213	225	31	30	10001
	205	213	26	36	8951
必经边<必经点<跳数	205	207	0	53	9404
	212	225	0	57	9916
	205	213	0	54	8947
必经边<跳数<必经点	204	207	0	47	10443
	223	225	0	43	10443
	213	213	0	45	9656
必经点<跳数<必经边	274	207	24	0	13358
	276	225	31	0	12854
	274	213	26	0	12974
必经点=必经边<跳数	204	207	0	47	10443
	223	225	0	43	10443
	213	213	0	45	9656
跳数<必经点/必经边	287	207	0	0	14645
	287	225	0	0	13577
	284	213	0	0	13936

4、总结

经过试验测试验证，本算法能够利用双向邻近节省插入法和 2opt 优化算法得到一条经过所有必经点、必经边的在跳数最小条件下花费最小的路径。费用优化测试验证在在路径跳数小于阈值的情况下费用优化算法确实能够起到优化路径、减小花费的作用。次优解参考路径测试验证了本算法能够在无解的情况根据约束条件的优先级输出不同的参考路径。以上测

试方法证明了我们最初的设想：将原问题可以转化为一笔画问题：在一个顶点为所有必经节点，边的权重为两个必经节点间的最小跳数的全联通无向图中，如何找到一条经过所有顶点且花费较小的路径？我们可以通过这样的方法找到：构建一条从起始节点指向终止节点的路径，每次选择一个必经点或者必经边插入到路径中，插入时选择合适的位置使路径花费的增加尽可能的小，直到将所有的必经点和必经边都插入。再针对这条得到的路径，进行优化和处理，得到最优解或者是在无解的情况下输出按约束条件优先级参考路径。

四、团队分工

团队名称：Jlu

成员名称：张希明　620321199512310315，吉林大学，分工：算法及编程

成员名称：刘宏宇　210402199412062043，吉林大学，分工：算法及测试

成员名称：吴鹏　　220282199506291712，吉林大学，分工：建模