Manhattan 空间有障碍的最短路径和 3-Steiner 树算法*

周 智[1,2], 蒋承东[1,2+], 黄刘生[1,2], 顾 钧[3]

[1](中国科学技术大学 计算机科学与技术系,安徽 合肥 230027)
[2](国家高性能计算中心(合肥),安徽 合肥 230027)
[3](香港科学技术大学 计算机科学系,香港)

On Optimal Rectilinear Shortest Paths and 3-Steiner Tree Routing in Presence of Obstacles

ZHOU Zhi[1,2], JIANG Cheng-Dong[1,2+], HUANG Liu-Sheng[1,2], GU Jun[3]

[1](Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)
[2](National High Performance Computing Center at Hefei, Hefei 230027, China)
[3](Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, China)
+ Corresponding author: Phn: 86-551-3603747, Fax: 86-551-3601013, E-mail: cdjiang@mail.ustc.edu.cn
http://www.nhpcc.ustc.edu.cn
Received 2002-04-10; Accepted 2002-12-24

Zhou Z, Jiang CD, Huang LS, Gu J. On optimal rectilinear shortest paths and 3-steiner tree routing in presence of obstacles. *Journal of Software*, 2003,14(9):1503~1514.
http://www.jos.org.cn/1000-9825/14/1503.htm

Abstract: Finding networks with minimal cost to connect points is a key problem in VLSI design, which can be described as obstacle-avoiding shortest path and minimum Steiner tree problem according to whether the number of points is greater than 2. Connection graphs, such as track based on graphs G_C and G_T and free area based graphs G_F and G_G, are effective tools for the shortest path problem, which is the foundation of the Steiner tree problem. The contribution of this paper includes three points: The dynamic algorithms for querying the shortest path between two points on each connection graph are designed and analyzed for the first time; secondly, all algorithms for Steiner problem on each connection graph are analyzed; The number of candidate Steiner points is reduced from $O((e+p)^2)$ to $O((t+p)^2)$ in the 3-Steiner algorithm on G_C, where e, t, p presents the number of edges, extreme edges of obstacles and terminals; An average $\Theta(t)$ algorithm for 3-Steiner problem are designed on G_G.
Key words: VLSI design; connection graph; shortest path; minimal Steiner tree

摘 要: 在 VLSI 设计中,多点互连是物理设计阶段的关键问题之一,而互连的点数等于 2 或大于 2 分别对应于 Manhattan 空间上有障碍时的最短路径问题和最小 Steiner 树问题,显然前者是后者的基础.连接图是研究最短路径问题的有效工具,已有的典型连接图包括基于轨迹的 G_C 和 G_T 以及基于自由区的 G_F 和 G_G.工作包括 3 个方面:设计

* Supported by the National Grand Fundamental Research 973 Program of China under Grant No.G1998030401 (国家重点基础研究发展规划(973))

第一作者简介: 周智(1976－),湖南娄底人,博士生,讲师,主要研究领域为复杂性理论,算法设计.

并分析了在各种连接图上实现动态的点对之间的最短路径查询算法;分析了在各个连接图上构造 3-Steiner 树的算法,对于已有的 G_C 上的 3-Steiner 算法,将其 Steiner 顶点的候选集合规模从 $O((e+p)^2)$降低到了 $O((t+p)^2)$,其中 e,t,p 分别表示边数、障碍极边数和顶点数;设计了在 G_G 上的 3-Steiner 树构造算法,其平均情况时间复杂度只有 $\Theta(t)$.

关键词: VLSI 设计;连接图;最短路径;最小 Steiner 树

中图法分类号: TP301 **文献标识码**: A

在有障碍时求两点间的最短路径是机器人设计、VLSI 布线设计和地理信息系统中的基本问题.由于设计规则和工艺的要求,VLSI 设计中的布线问题有以下特性:

(1) 布线空间为 $m\times n$ 的有限平面网格;

(2) 电路元件和已布好的线成为新的布线的障碍;

(3) 两点间路径和障碍的边均为水平或竖直边.

现有的算法分为两类:走迷宫(maze-running)算法和线搜索(line-search)算法.

Lee 算法[1]为第 1 个走迷宫算法,采用的是广度优先搜索方法.此算法的缺点在于其时间和空间复杂度均为 $O(mn)$,且每个节点需 $O(\log L)$ 字节存储空间,其中 L 为源节点到目标节点的最短路径长度.Lee 算法有许多改进版本:Akers[2]引进一种编码策略,使每个节点的存储空间只需 2 字节而与 L 无关;Soukup[3]提出一种将深度优先搜索与广度优先搜索相结合的方法,以“不改变方向”为启发策略,提高算法性能,但无法保证得到最优解.

迷宫算法基于单位格点搜索,故在时间和空间上是低效的.为此,人们提出了线搜索算法.基本思想是构造一个代表障碍和节点位置的图,通过在图中寻找最短路径得到原格点中的最短路径.此类算法有以下特点:

(1) 构造的图比原单元格点图稀疏,因此而导致算法时间复杂度降低;

(2) 构造的图中的路径与原网格中的路径可以对应起来.对源和目标节点,构造的图中存在与原网格中的最短路径相对应的路径.

较早的此类算法由 Hightower[4]提出.后来有一些较新的线搜索算法基于计算几何的方法,如 Clarkson[5]等,计算几何算法的时间复杂度较低但实现过于复杂而难以在实际中应用.Zheng[6]引入了一个强连接图,其算法时间复杂度为 $O(e^2)$,其中 e 为所有障碍的总边数,它同时采用 A^* 算法和“不改向”启发策略,有较好的实用性.Wu[7]引入一个很小的连接图,称为轨迹图(track graph).它不是强连接图,但对例外情况有简单而有效的处理.该算法的时间和空间复杂度分别为 $O((e+k)\log(t))$ 和 $O(e+k)$,其中 t 为障碍的极边数,k 为轨迹的交点数,且 $k=O(t^2)$.在最坏情况下,$t=e,k=e^2$.文献[8,9]中给出了基于自由区定义的连接图,其顶点数和边数分别为 $O(t)$ 和 $O(t\log t)$,为现有的线搜索算法中复杂度最低的一种.

而连接图需要完成的工作不仅包括两点之间的最短路径,连接多个顶点的最小 Steiner 树[10]是进一步需要研究的对象.在无障碍的 Manhattan 空间中,有一系列关于最小 Steiner 树的性质[11,12]和求解的近似算法[13],而在有障碍的情况下,文献[10]给出了在矩形障碍区域下的对应性质,并给出了几个基本算法,这些算法所使用的连接图等价于边连接图 G_G,并且采用基于对候选点集合进行枚举的方法效率很低,无法满足 VLSI 中实际运用的需要.显然,有必要采用高效连接图来提高求解效率.

本文第 1 节简要描述了本文中涉及的概念(并在文献[12,14]中给出严格和详细的描述)和已有的结论.第 2 节分析了在各个连接图上实现点对之间的路径查询的方法.第 3 节分别在 G_F,G_T,G_G 上设计求 3 点之间 Steiner 树的算法.第 4 节总结了分析和比较的结果,并给出了进一步的研究方向.

1 基本定义及特性

1.1 最短路径和连接图

在 Manhattan 空间上,两点之间的距离定义为其横、纵坐标差值之和,所有路径只能沿水平方向和竖直方向设置,这使得在该空间上对路径、多边形、凹凸性等的定义方式和性质与欧氏空间大不相同(关于这些概念的严格定义和基本性质参见文献[14]).一个最重要的结论是:两点之间的最短路径的长度若等于其距离,则不包含

改变方向的边,即极边,如图 1 中 s_1,t_1 所示:从 s_1 到 t_1 其水平边均为从左到右,竖直边均为从下到上;在有多边形障碍的时候两点之间的最短路径要么等于其距离,如图 1 中 s_2,t_2 所示,要么必然包含障碍的极边,如图 1 中 s_3,t_3 所示.这些结论使得任意最短路径可以用很少的极边来描述,用障碍上的边特别是极边构造出的连接图上可以表示任意两点之间的最短路径,从而使得构造与搜索区域面积无关的最短路径算法成为可能.

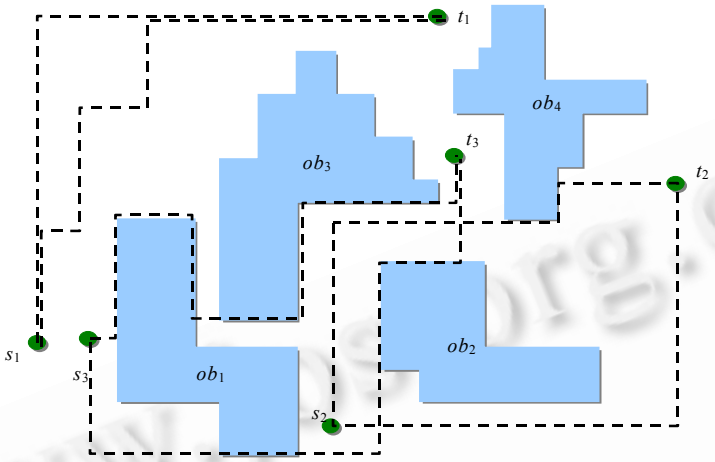
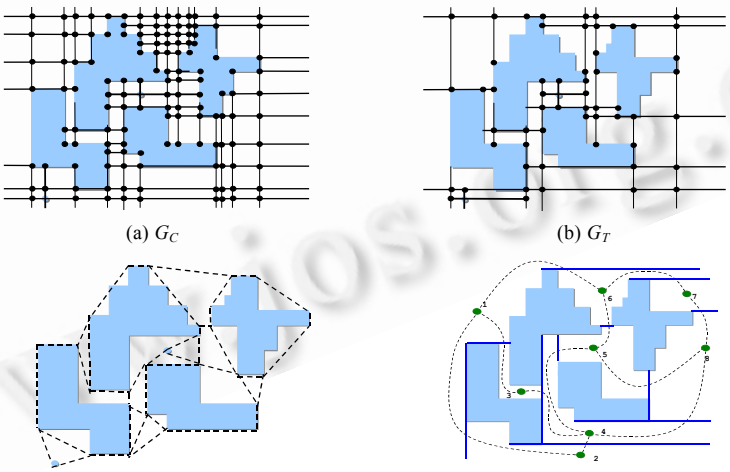


Fig.1 Rectilinear shortest paths in presence of obstacles

图 1 有障碍的 Manhattan 空间上点对间的最短路径

现有的 4 种典型的连接图如图 2 所示:G_C 由所有障碍的边界上的边所在的轨迹的交点所构成的网格定义,G_T 由障碍上的极边上的轨迹的交点所构成的网格定义,G_F 基于极顶之间的矩形自由区定义,而 G_G 定义在广义自由区的概念上.其性能比较见表 1(其中 e 为障碍上的边数,t 为极边数),由于求两点之间的最短路径与连接图是否为强连接图以及连接图的边数直接相关,显然在性能上的排序应该是 $G_C<G_T<G_F<G_G$.



(a) G_C (b) G_T

(c) G_F (d) G_G

Fig.2 Four kinds of connection graphs

图 2 4 种典型的连接图

可以看到,对连接图的设计和性能分析都是基于两点之间的最短路径,而没有对多点路由的性能进行分析比较,显然是不足的.本文将分析它们在这方面的特性,其基础为多点间互连的最小 Steiner 树算法对连接图的要求.

Table 1 Comparison among four kinds of connection graphs (G_C,G_T ,G_F,G_G)

表 1 4 种典型的连接图 G_C,G_T ,G_F 和 G_G 的对比

	G_C	G_T	G_F	G_G
Number of vertices	$O(e^2)$	$O(t^2)$	$O(t)$	$\Theta(t)$
Number of edges	$O(e^2)$	$O(t^2)$	$O(t\log t)$	$\Theta(t)$
Maximum degree of vertices	$O(1)$	$O(1)$	$O(t)$	Planar graph
Construction complexity	$O(e^2\log e)$	$O(t^2\log t)$	$O(t\log t)$	$O(t\log t)$
Uniqueness	Yes	Yes	Yes	No
Strongly connected graph	Yes	No	Yes	No
Planar graph	Yes	Yes	No	Yes
Track based	Yes	Yes	No	No
Free area based	No	No	Yes	Yes

1.2 最小Steiner树

定义 1(一般的最小 Steiner 树问题). 给定无向加权图 $G=(V,E,w)$ 和其顶点子集 $U\subset V$,ST 为 G 的子树,且 $U\subset V(ST)$,则称 ST 为点集 U 在图 G 上的 Steiner 树,记其集合为 $Steiner(G,U)$, $v\in V(ST)-U$ 称为 ST 上的 Steiner 点,记为 $Steiner(ST)$;记 Steiner 树的权值 $w(ST)$ 为所有边权之和.若 $w(ST_{\text{opt}})=\min\limits_{ST\in Steiner(G,U)} w(ST)$,则称 ST_{opt} 为 U 在 G 上的最小 Steiner 树,记为 $ST_{\text{opt}}(G,U)$,记 U 在 G 上的最小生成树为 $MST(U)$.若图 G 定义在 Manhattan 空间上,即顶点集 V 为 Manhattan 空间上的点,两点之间的边的权值等于其 Manhattan 距离,则称 $ST_{\text{opt}}(G,U)$ 为 $MRST(U)$ (minimal rectilinear Steiner tree),称顶度数大于 2 的 Steiner 点为非平凡 Steiner 点(以下除非特殊说明,图都定义在 Manhattan 空间上,且 Steiner 点均指非平凡的 Steiner 点).MRST 性质如下:

定理 1[12]. MRST 问题是 NP-Hard 问题.

定理 2[11]**(Hanan 定理)**. $MRST(U)$ 上的 Steiner 点均在经过 U 的水平线和竖直线的交点上,如图 3 所示.

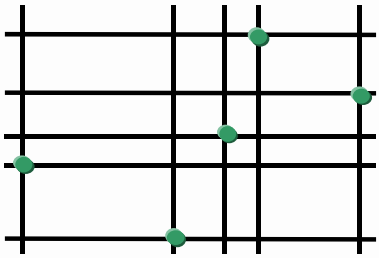


Fig.3 Hanan theory

图 3 Hanan 定理

定理 3[11]. $w(MRST(U))\ge(2/3)w(MST(U))$.

定义 2(问题定义). 对 $SPP=(OB,PIN)$ [10],其中 OB 为障碍集合, $PIN\subset N^2-OB$ 为顶点集合, $|PIN|=p$,令 $G_{\text{SPP}}=(V_{\text{SPP}},E_{\text{SPP}})$ 为 G_0 在 N^2-OB 上的生成子图(不失一般性,我们限制障碍为凸多边形(对凹多边形的情况讨论请参见文献[9]).同时,为描述方便,设 G_{SPP} 定义在 N^2-OB' 上[14],其中 $OB'=\bigcup_{ob_i\in OB} in(ob_i)$),则在包含障碍 OB 的网格面 N^2 上求连接点集 PIN 的最小 Steiner 树问题可描述为:求 PIN 在 G_{SPP} 上的最小 Steiner 树 $ST_{\text{opt}}(G_{\text{SPP}},PIN)$,简记为 OAMRST(SPP)(obstacle-avoiding minimum rectilinear Steiner tree),同样定义 OAMST(SPP)为 G_{SPP} 上连接 PIN 的最小生成树.如图 4 所示,对 $PIN=\{v_1,v_2,v_3,v_4\}$,实线为 OAMST,虚线为 OAMRST,其中方块点 s_1,s_2 为 Steiner 点.

有几个基本结论如下:

定理 4[10]. OAMRST 为 NP-Hard 问题.

定理 5[10]. OAMRST(SPP)上的 Steiner 点在 G_C(SPP)上(称 G_c 为 Escaping Graph[10],同时,障碍集合仅定义为矩形区域,互连点集合定义在障碍的边界上).

定理 6[11]. $2\times w(OAMRST(U))\ge w(OAMST(U))$.

在定理 5 的基础上,我们又可以给出更简化的约束条件,首先有如下定义:

定义 3(顶点退化的实例). 对 PIN 上的顶点 pin 视为退化的多边形障碍 $ob(pin)$,即其上下左右极边

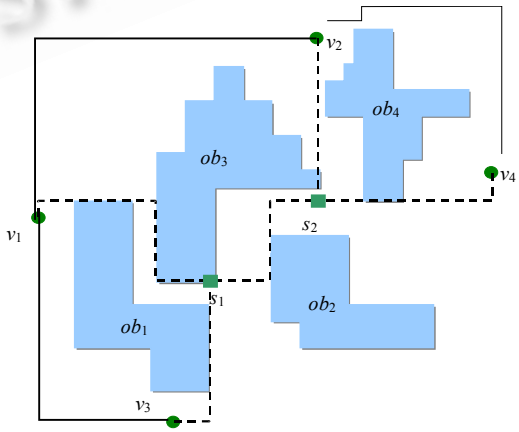


Fig.4 Minimum Steiner tree in presence of obstacles

图 4 有障碍情况下的最小 Steiner 树示意图

重合在该顶点上,得到实例 $SPP_{de}=(OB\cup(\bigcup_{pin\in PIN}ob(pin)),\varnothing)$,称为对 SPP 退化后的实例.

仍然有定理 4 的结论,定理 7 给出了最简单的完全构造算法,注意到,Steiner 树上的 Steiner 顶点至多为 $p-2$ 个,故将所有可能的方案枚举一遍的时间为 $O\left(\sum_{i=0}^{p-2}\binom{(e+p)^2}{i}t_{MST}(p+i)\right)$,其中 $t_{MST}(k)$ 为求 k 个顶点的最小生成树的时间.注意到,定理 2 给出在无障碍情况下算法的时间为 $\sum_{i=0}^{p-2}\binom{p^2}{i}t'_{MST}(p+i)$.在有障碍的情况下,连接方案的数量和求单个方案的代价都极大地增加了,问题的难度相应也有所增加,算法所需的时间往往超出实际能接受的范围,人们对该问题所得到的算法结果十分有限.显然,对每对顶点都在连接图上来求其最短路径是不合算的,因而研究在不同的连接图上点对之间的最短路径算法就成为了一个基本任务.

2 点对之间最短路径的查询

在连接图上求所有点对之间的最短路径有两种方法,其一,在连接图(除 G_G 以外,因为其构造与 PIN 上的顶点无关)上考虑 PIN 上的顶点,求出 PIN 中的点对之间的最短路径;其二,只考虑障碍得到连接图,求该图上关键的顶点之间的最短路径,对任意不在障碍上的顶点对之间的最短路径,通过某种算法用关键点对之间的最短路径求得,即查询得到.对给定的无向非负加权图 $G=(V,E,w)$,求点对之间的最短路径可用 Johnson 算法和用 Fibonacci 堆实现的优先队列在 $O(|V|^2\log|V|+|V||E|)$ 时间内完成.而若图 G 为平面图,则可用 Frederickson 的快速算法[15]在 $O(|V|^2)$ 时间内得到.对于第 1 种方法,存储空间需要 $O(p^2)$,而 G_C,G_T 和 G_F 所需的时间分别为 $O((e+p)^4)$,$O((t^2+p)^2)$ 和 $O(t^2\log t)$.然而在 VLSI 中,每个网络的顶点数 p 很小,而网格数 n 极大,显然这种方法不可取.以下我们分别对各个连接图考虑第 2 种方法.

2.1 基于轨迹的连接图G_C和G_T

注意到这两个图的共同之处非常明显,都是基于障碍上的边界的轨迹所构成的网格,所不同的只是后者将边界限制在极边范围内.因而我们首先分析轨迹的特性.

定义 4(相邻轨迹). 对竖直轨迹(关于轨迹等的基本概念请参考文献[14])集合 $TV=\{tv_i=(x,d,u)\}$ 按其 x 坐标排序为 $x(tv_i)\le x(tv_{i+1})$,其中 $u(tv_i),d(tv_i)$ 分别为 tv_i 的 y 坐标的最大和最小值,记轨迹区间为 $in(tv_i)=[d(tv_i),u(tv_i)]$;对 tv_i,tv_j,记 $in(tv_i,tv_j)=in(tv_i)\cap in(tv_j)$;若两条轨迹 $x(tv_i)<x(tv_j)$,$in(tv_i,tv_j)\ne\varnothing$,且在矩形区域 $[x(tv_i),x(tv_j)]\times in(tv_i,tv_j)$ 中不包含障碍中的点,同时不存在第 3 条轨迹 tv_k,使得 $x(tv_i)<x(tv_k)<x(tv_j)$,且 $in(tv_i,tv_j)\cap(in(tv_i,tv_k)\cup in(tv_k,tv_j))\ne\varnothing$,则称 tv_i,tv_j 相邻,记为 $tv_i\prec_x tv_j$;称矩形区域 $[x(tv_i),x(tv_j)]\times in(tv_i,tv_j)$ 为 tv_i,tv_j 的邻接区域,记为 $AF(tv_i,tv_j)$.对于水平轨迹集合 $TH=\{th_i=(y,l,r)\}$,同样定义 $th_i\prec_y th_j$ 和 $AF(th_i,th_j)$ 等对应概念.

引理 1. 对 G_C 上的竖直轨迹 $TV=\{tv_i\}$ 和水平轨迹 $TH=\{th_i\}$,$\bigcup_{tv_i\prec_x tv_j}(AF(tv_i,tv_j)-OB)=\bigcup_{th_i\prec_y th_j}(AF(tv_i,tv_j)-OB)=N^2-OB$,即 G_C 上水平轨迹和竖直轨迹所构成的邻接区域集合分别将障碍外的点完全覆盖.

引理 2. 对 G_T 上的竖直轨迹 $TV=\{tv_i\}$ 和水平轨迹 $TH=\{th_i\}$,$\bigcup_{tv_i\prec_x tv_j}(AF(tv_i,tv_j)-OB)\cup\bigcup_{ob_i\in OB}ExR(ob_i)=\bigcup_{th_i\prec_y th_j}(AF(tv_i,tv_j)-OB)\cup\bigcup_{ob_i\in OB}ExR(ob_i)=N^2-OB$,即除部分空极区中的点以外,对 G_T 有和 G_C 相同的结论(空极区的概念请参考文献[14]).

定义 5(一般顶点在轨迹图上的邻接区和邻接顶点). 对任意点 $s\in N^2-OB$,若 s 不被包含在任意一条轨迹上,由引理 1 可知,存在 $tv_{i1}\prec_x tv_{j1}$ 和 $th_{i2}\prec_y th_{j2}$,使得 $s\in AF(th_{i2},th_{j2})\cap AF(tv_{i1},tv_{j1})$,称 $AF(th_{i2},th_{j2})\cap AF(tv_{i1},tv_{j1})$ 为 s 的邻接区,记为 $AF(s)$;否则(对图 G_T 所定义的轨迹集合),s 在障碍的空极区内,则定义包含 s,边界为轨迹和障碍边界的最小区域为 s 的邻接区 $AF(s)$.称 $AF(s)$ 包含在轨迹上的顶点为 s 的邻接顶点,记为 $AV(s)$.对于 s 被包含在某一轨迹上的情况,将该轨迹视为退化的两条平行轨迹处理.

引理 3. $\bigcup_{s\in N^2-OB}AF(s)=N^2-OB$,且对任意 $s,t\in N^2-OB$,或者 $AF(s)=AF(t)$,或者 $AF(s)\cap AF(t)=\varnothing$,或者 $AF(s)\cap AF(t)$ 为一条轨迹上的线段或边.

引理 4. 对任意 $s \in N^2 - OB$,都有 $|AV(s)| \in \{2,3,4\}$.

引理 5. 在图 G_C 中,对任意 $s \in N^2 - OB$,都有 $AV(s) \subset V(G_C)$.对 G_T 有相同的结论.

从而我们可以依赖已经得到的连接图上的点对之间的信息得到任意点对间的最短路径.

定理 7(基于轨迹的连接图上点对间最短路径查询). 对 $s,t \in N^2 - OB$,在 G_C 或 G_T 中,若 $\exists tv_i \prec_x tv_j$,使得 $s,t \in AF(tv_i, tv_j)$,则连接 s,t 的最短路径 $SP(s,t)$ 在该区域中且路径长度等于两点之间的距离(如图 5(a)所示);否则,如图 5(b)所示, $\rho(SP(s,t)) = \min_{s_i \in AV(s), t_j \in AV(t)} \{\rho(s,s_i) + \rho(SP(s_i,t_j)) + \rho(t_j,t)\}$,即 s,t 之间的最短路径可以通过连接各个顶点在该连接图上的邻接顶点以及邻接顶点之间的最短路径得到.

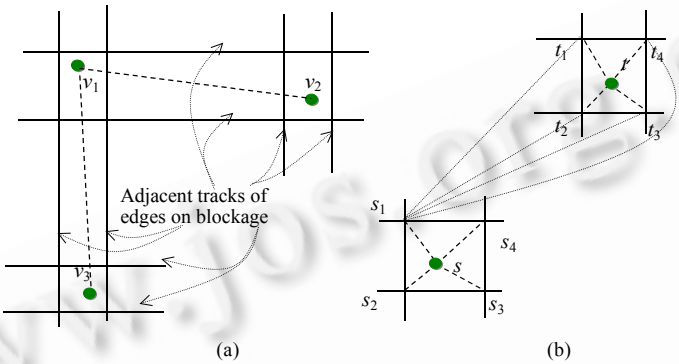


Fig.5 Querying the shortest path in G_C and G_T

图 5 G_C 和 G_T 上点对之间最短路径的查询

对于含有 $O(n)$ 条轨迹的连接图,显然其顶点数和边数均为 $O(n^2)$,且注意到该图为平面图,从而建立该图上点对之间的最短路径长度查询的矩阵需要 $O(n^4)$,存储需要 $O(n^4)$ 空间.利用在存储池中实现的邻接图来表示同方向的轨迹的相邻特性,按 x,y 坐标将连接图上的顶点散列在存储池中,则判断两点是否处在同一个相邻区域中只需 $O(\log n)$ 时间,同时得到顶点的邻接顶点需要 $O(\log n)$ 时间,由引理 5 可知,每个点的邻接顶点数为常数,从而查询两点之间的最短路径需要 $O(\log n)$ 时间.

2.2 基于自由区的连接图 G_F

已知 $G_F(OB)$,显然该图的组织形式与上面的基于轨迹的连接图极不一致,由于 G_F 的顶点均为极顶,故要利用 G_F 的信息必须要求两点处在不同的自由区内,从而,对 $s,t \in N^2 - OB$ (不妨设其均不为 G_F 的顶点),求最短路径分为两步:

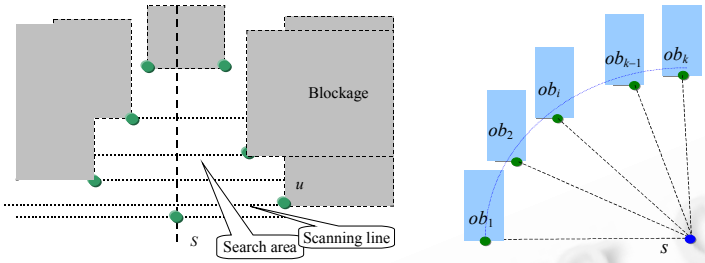
(1) 判断以 s,t 为对角顶点的矩形区域是否为自由区,即其中是否含有极顶.若没有,则显然该矩形区域内任意没有极边的路径为连接它们的最短路径,即有 $\rho_{G_F}(s,t) = \rho(s,t)$;

(2) 定义 $AV(s)=\{v|\langle s,v\rangle \in E(G_F(OB \cup \{ob(s)\}))\}$ 为顶点 s 在 $G_F(OB)$ 上的邻接顶点,即为将 s 视为退化的障碍得到的自由区连接图上该顶点的相邻顶点,显然这些顶点均为极顶.令 $\rho_{G_F}(s,t) = \min_{s_i \in AV(s), t_j \in AV(t)} \{\rho(s,s_i) + \rho(SP_{G_F}(s_i,t_j)) + \rho(t_j,t)\}$,其中 $SP_{G_F}(v,u)$ 为图 G_F 上极顶 u,v 之间的最短路径.显然,(1)所定义的情况对应于在图 $G_F(OB \cup \{ob(s), ob(t)\})$ 上 s,t 之间存在边的情况.又 $G_F(OB)$ 为定义在退化的 SPP_{de} 上的强连接图[8],故有结论如下:

定理 8(自由区连接图上的点对间路径查询). $\rho_{G_F}(s,t) = \rho(SP(s,t))$,即按上面方式得到的最短路径是正确的.

$G_F(OB)$ 上 $|V| = O(t)$, $|E| = O(t \log t)$,故对(1)的情况作判断需要 $O(\log t)$ 时间;对于情况(2),关键在于求任意点的邻接顶点,显然可以按照构造 G_F 的方式得到,即用扫描线方法构造,其主体思想为:对 s 的 4 个象限上的构造显然是完全类似的,如图 6(a)所示,若在顶点 s 的右上方向有极顶 t,则在阴影区域($x > x(u)$, $y > y(u)$)内的极顶不可能和 s 得到自由区,在下一次 y 扫描($y > y(u)$)中无须考虑 $x > x(u)$ 的顶,如此直到构造结束.该算法的正确性在文献[8,9]中得到保证.显然,构造相邻顶点集合的复杂度为 $O(\log t + |AV(s)|)$,查询的复杂度为 $O(|AV(s)|^2)$.注

意到 $|AV(s)|=d_{G_F(OB\cup\{ob(s)\})}(s)$,故 $|AV(s)|$ 的期望值(构造平均复杂度)为 $O(\log t)$,而在最坏情况下,$|AV(s)|=O(t)$,如图 6(b)所示,故在 G_F 上查询两点之间距离的最坏情况复杂度为 $O(t^2)$,而平均情况为 $O(\log^2 t)$.



(a) Plane scanning algorithm
(a) 扫描构造算法

(b) Worst case
(b) 最坏情况构造

Fig.6 Adjacent vertex of s in G_F
图 6 点 s 在 G_F 上的邻接顶点

2.3 基于正规划分的广义连接图G_G

类似于上面的情况,当用 G_G 查询时,我们同样试图尽量利用在该图上已经能够得到的最短路径信息,为此,必须将 G_G 上广义自由区之间的关系转化为点之间的关系,有定义如下:

定义 6(正规顶点). 对于正规划分[14] $P_g=\{F_g\}\in UG(N^2-OB)$,顶点 $s\in N^2-in(OB)$,若存在 $F_{g1},F_{g2}\in P_g$ 使得 $e=F_{g1}\cap F_{g2}\neq\varnothing$(由正规划分的定义,知其为一条线段),且 s 为 e 的端点,则称 s 为正规划分 P_g 的正规顶点,记为 $s\in UV(P_g)$,如图 7 所示.注意到,相邻的广义自由区至多有一条公共边,所以,$|UV(P_g)|=2|E(G_G)|=\Theta(t)$,从而有:

引理 7. $|UV(P_g)|=\Theta(t)$,求 $UV(P_g)$ 中任意两点之间的最短距离的时间复杂度为 $O(t^2)$.

基于广义连接图的最短路径查询算法.

对目标顶点 $s,t\in N^2-OB$ 和广义连接图 G_G 的一个正规划分 P_g(在这里考虑用扫描线算法构造的划分),首先定义邻接区域和邻接顶点如下:

(1) 若存在 $F_g\in P_g$ 使得 $s\in\mathrm{in}(F_g)$,则定义 s 的邻接区域 $AF(s)=F_g$;

(2) 否则,存在 $F_{g1},F_{g2}\in P_g$ 使得 $s\in F_{g1}\cap F_{g2}$,也就是说,s 在广义自由区的边界上,则定义 s 的邻接区域 $AF(s)=F_{g1}\cap F_{g2}$;

(3) 定义邻接区域上的正规顶点为 s 的邻接顶点,即 $AV(s)=AF(s)\cap UV(P_g)$.

在广义连接图 G_G 上,可以采用与其他连接图类似的"查询"方式得到最短路径.首先计算出所有正规顶点两两之间的最短路径,构造一个查询矩阵.查询算法如下:

(1) 以 s,t 为对角顶点定义矩形区域 $R(s,t)$;

(2) 检查 s,t 是否在同一个广义自由区中,若存在 $F_g\in P_g$ 使得 $AF(s),AF(t)\subseteq F_g$,即两个顶点处于同一个广义自由区内或其边界上,根据广义自由区的定义,连接 s,t 之间的最短路径等于其距离,即 $\rho(SP(s,t))=\rho_{G_G}(s,t)=\rho(s,t)$,如图 7 中点对 v_1,v_2 和 v_3,v_4 所示;

(3) s,t 分属不同的广义自由区,且 $R(s,t)$与 $AF(s)$的边界相交的边完全包含在两个自由区的相邻边界上:即存在 $F_g\in P_g$ 使得 $AF(s)\cap R(s,t)\subset AF(s)\cap F_g$,如图 7 中 v_2,v_4 和 v_3,v_5 所示.此时补充 s 的邻接顶点为 $AF(s)\cap R(s,t)$ 的两个端点,定义为 $AV_t'(s)$.令 s 的新的邻接顶点集合为 $AV_t(s)=AV(s)\cup AV_t'(s)$.对 t 也有同样的定义 $AV_s(t)$.

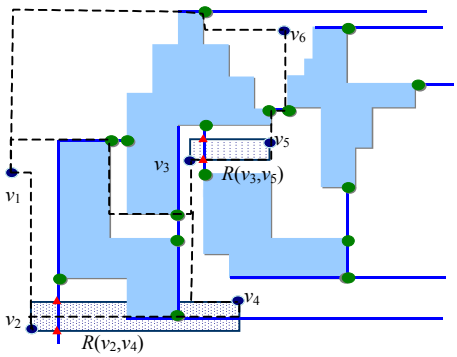


Fig.7 Formal vertex and shortest path in G_G
图 7 G_G 上的正规顶点和最短路径

(4) 其他情况下邻接顶点集合不变.此时从 s 出发到 t 的路径查询就是完备的了.可以构造 s,t 之间的最短路径,使其经过各自的邻接顶点.采用下式就可以得到 s,t 的最短路径:

$$\rho_{G_G}(s,t)=\min_{s_i\in AV_t(s),t_j\in AV_s(t)}\{\rho(s,s_i)+\rho(SP_{G_G}(s_i,t_j))+\rho(t_j,t)\}.$$

根据 G_G 的构造方法,有如下结论:

定理 9(基于广义自由区的广义连接图的点对间最短路径查询). 对任意两个顶点 $s,t\in N^2-OB$,按上述算法构造的路径为连接 s,t 的最短路径,即 $\rho_{G_G}(s,t)=\rho(SP(s,t))$.

从复杂度上分析,对图 G_G,$|V|=\Theta(t)$,$|E|=\Theta(t)$,构造查询矩阵的预处理时间为 $O(t\log t)$.上述(2)的操作只需 $\Theta(\log t)$时间,而(3)和(4)的操作时间取决于 $AF(s)$边界的复杂度.在最坏情况下,$AV(s)=O(t)$,如图 8 所示,而在平均情况下为 $O(1)$;故查询两点间的最短距离的最坏情况复杂度为 $O(t^2)$,而平均情况复杂度为 $O(\log t)$.

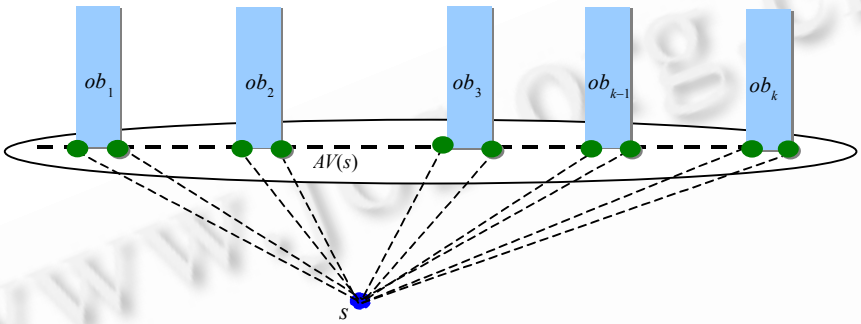


Fig.8 Worst case of querying shortest path in G_G

图 8 G_G 上点对路径查询的最坏情况

3 3 点 Steiner 树的构造

见表 2,在 VLSI 设计中涉及到的最小 Steiner 树的平均顶点数为 3.5 左右,其中 3 顶点和 4 顶点的网络占所有两个以上的顶点网络的 74%.从而,求 Steiner 树的一个自然的想法是用完全算法得到 3 个顶点的互连方案,然后作为一个基本单元应用到更多顶点的 Steiner 树实例上,如 G3S 算法和 B3S 算法[10].3 顶点的 Steiner 树只有两种方案,其一为不增加 Steiner 点的 3 点互连,其二为增加一个 Steiner 点连接单个顶点.利用定理 5,基于连接图 G_C 求最优 3 点 Steiner 树需要的计算时间为 $O(e^2t_s)$,其中 t_s 为求两点之间的最短路径的时间.以下我们分析在其他连接图上对应的性质、算法和复杂度.

Table 2 Percentage of number of network vertices in SIGDA library of standard instances

表 2 在 SIGDA 标准实例库上的网络的端点数的比例

Number of vertices	2	3	4	5	6	7	8	9	10	>10
Percentage in library	52.8	27.5	7.4	2.7	3.1	1.4	0.2	1.0	0.1	3.8

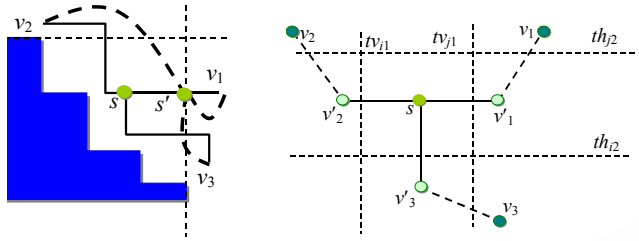
3.1 基于极边的连接图G_T

与 G_C 不同,G_T 不是强连接图,原因在于对非障碍点只连接到最近的极边轨迹上.显然,若将完整的轨迹加入连接图中将得到强连接图,即有如下结论:

定理 10. $G_T(SPP_{de})$ 为强连接图.

定理 11. 存在 OAMRST(SPP)使得其中的 Steiner 点均为 $G_T(SPP_{de})$ 的顶点.

证明:设 T 为一棵定义在 $SPP=(OB,PIN)$ 上的最小 Steiner 树,设其中存在 Steiner 顶点 s 不在 $G_T(SPP_{de})$ 上,设 $d_T(s)=3$,则不妨设与其相邻的顶点 v_1,v_2,v_3 分别从 x 正向、x 负向和 y 负向与 s 相连接(如若两个顶点到 s 的方向是一致的,则可以增加 Steiner 顶点,使得整个树的代价减小,与 T 为最小 Steiner 树相矛盾).注意到 $AF(s)$不会出现在某个空极区内(否则,如图 9(a)所示,注意到,由于空极区中点有两个方向上的布线必然会有相同的"后续方向",如图中 v_1,v_3 所示.故至多有两个方向的布线可以相互独立,最小 Steiner 树在该区域的 Steiner 点的度数至多为 2,与我们的假设矛盾),故可设存在 $tv_{i1}\prec_x tv_{j1}$,$th_{i2}\prec_y th_{j2}$,使得 s 处在这 4 条轨迹形成的矩形区域中,不失一般性地假设 s 的邻接区 $AF(s)$就是这个矩形区域,如图 9(b)所示.考虑如下两种情况:



(a) (b)

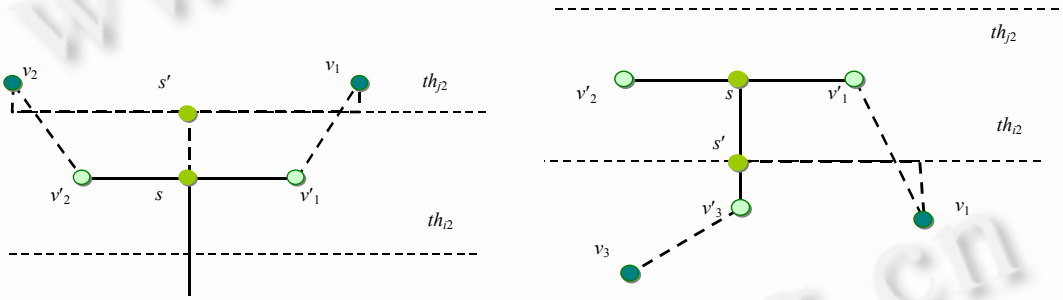
Fig.9 Probable distributions of Steiner points outside of graph $G_T(SPP_{de})$

图 9 不在图 $G_T(SPP_{de})$ 上的 Steiner 顶点的可能分布

(a) v_1,v_2 都不在 th_{i2},th_{j2} 之间:

(a1) 若子路径 sv_1,sv_2 均首先与 th_{j2} 相交,如图 10(a)所示,则将 s 移动到 s',使得 $s'\in th_{i2}$, $Dirt(ss')=U$ (ss' 的方向向上),由于 s 的邻接区 $AF(s)$中不包含障碍的点,故所得到的 Steiner 树是合法的,显然其代价为 $w(T)-\rho(s,s')$,与 T 为最小 Steiner 树相矛盾;

(a2) 否则,不妨设 sv_1 首先与 th_{i2} 相交,如图 10(b)所示,设 sv_3 的第 1 条边(由假设,该边为竖直边)上的另一个端点为 v'_3,令 $s'=(x(s),\max(y(th_{i2}),y(v'_3))$,则将 Steiner 顶点 s 移动到 s' 上而得到的 Steiner 树的代价为 $w(T)-\rho(s,s')$,同样与 T 为最小 Steiner 树矛盾;



(a) (b)

Fig.10 Two sub-cases of (a) in proof of theory 11

图 10 定理 11 证明中情况(a)的两种子情况

(b) v_1,v_2 至少有一个在 th_{i2},th_{j2} 之间(如图 11 所示):不妨设为 v_1,则 v_1 必然是一个 Steiner 顶点,否则其轨迹使得 $th_{i2}\prec_y th_{j2}$ 不能成立.取 T 在 th_{i2},th_{j2} 之间的那些点集为 S,则有 $S\in Steiner(T)$, $v_1,s\in S$.令

$S_u=\{v\in V(T)-S\mid \exists u\in S,s.t.\langle u,v\rangle\in E(T),$且$\langle u,v\rangle$首先与$th_{j2}$相交$\}$, 令 $\Delta_u=\min\limits_{s\in S}(y(th_{j2})-y(s))$,

$S_d=\{v\in V(T)-S\mid \exists u\in S,s.t.\langle u,v\rangle\in E(T),$且$\langle u,v\rangle$首先与$th_{i2}$相交$\}$, 令 $\Delta_d=\min\limits_{s\in S}(y(s)-y(th_{i2}))$.

显然有 $\Delta_d>0,\Delta_u>0$.定义点集 $S_\Delta=\{(x,y)\mid (x,y-\Delta)\in S\}$ 为将 S 中的顶点位移Δ得到的顶点集合,则保持拓扑连接不变,当 $-\Delta_d\le\Delta\le\Delta_u$ 时得到的点集 S_Δ 与原先的边构成合法的 Steiner 树 $T_{S,\Delta}$,其边权代价为

$$w(T_{S,\Delta})=w(T)-|S_u|\times\Delta+|S_d|\times\Delta.$$

因为 T 为最小 Steiner 树,故对于 $\forall-\Delta_d\le\Delta\le\Delta_u$,都有 $w(T_{S,\Delta})\le w(T)$,显然可以得到 $|S_u|=|S_d|$,即 $w(T_{S,\Delta})=w(T)$.此时将 S 整体上移 Δ_u 或整体下移 Δ_d ,即减少了在 th_{i2},th_{j2} 之间的 Steiner 顶点的数量,不断重复上面的过程,直到 $S=\varnothing$为止.

对竖直轨迹做类似的操作,可将 Steiner 顶点移动到连接图的顶点上,同时不增加所得到的 Steiner 树的代价.上面的证明方式也可以简单地用于 $d(s)=4$ 的情况. □

对于 $|PIN|=p$,定理 11 使得 Steiner 点的候选集合的规模由 $O((e+p)^2)$ 降低到 $O((t+p)^2)$,注意到,当障碍完全由矩形区域构成时,两者是一致的,而对一般的多边形障碍,这极大地减小了候选集合的规模.显然,对于 3 点

Steiner 树,如果已经建立了点对之间的最短路径查询,连接图 G_T 上的构造算法的复杂度为 $O(t^2)$,否则为 $O(t^4)$.

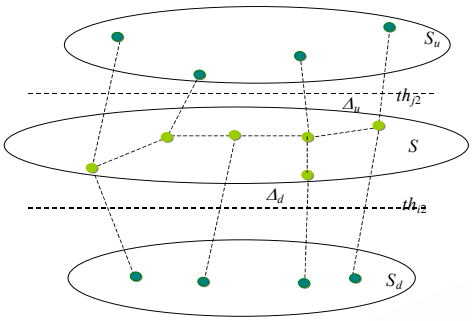


Fig.11 Case (b) in proof of theory 11
图 11 定理 11 证明中情况(b)的示意图

3.2 基于自由区的连接图G_F

由于该图为强连接图,且注意到若需要互连的顶点之间的最短路径与其距离不相等时必然经过 G_F 的顶点,故若将 Steiner 点限制在图 G_F 的顶点上,可以得到对最小 Steiner 树的一种近似.图 G_F 上的顶点数为 $O(t)$,根据上面对点对之间路径的查询的结论,该近似算法的最坏情况复杂度为 $O(t^2)$,而平均复杂度为 $O(t\log t)$.显然,这种近似没有充分地利用 Manhattan 空间中路径的信息,需要作一些改进,这些改进涉及到 Manhattan 空间上路径的区域表示和图 G_F 上的对应表示的问题,参见文献[9],此处不再详述.

3.3 基于正规划分的广义连接图G_G

G_G 有如下两点性质:

(1) 注意到一个正规划分 P_g 能完全覆盖整个没有障碍的区域,对连接顶点 v_1,v_2,v_3 的最小 Steiner 树,若其中包含 Steiner 顶点 s,则存在 $F_g \in P_g$ 使得 $s \in F_g$.

(2) 定义点 v 到线段 e 的最短距离为 $\rho(v,e)=\rho(v,u_{v,e})$,其中点 $u_{v,e} \in e$ 使得 $\rho(v,u_{v,e})=\min_{u\in e}\rho(SP(v,u))$,这可以由规范路径得到(请参见文献[14]).

求 3 点间最小 Steiner 树的算法如下:

(1) 若顶点 v_1,v_2,v_3 在同一个广义自由区中,即存在 $F_g \in P_g$,使得 $\{v_1,v_2,v_3\} \subset F_g$,由于广义自由区是凸区域,故可直接按照无障碍时的情况求解;

(2) 否则,对于正规划分 P_g 的每个广义自由区 F_g,定义如下点集:

(a) 对 $v \in \{v_1,v_2,v_3\}-F_g$,令 $B(v,F_g)=\{u_{v,e} \mid \exists F_g' \in P_g, s.t. F_g' \cap F_g = e\}$,即为自由区外的顶点 v 到达自由区 F_g 所可能经过的那些顶点的集合,如图 12 所示.

(b) 对 $v \in \{v_1,v_2,v_3\} \cap F_g$,令 $B(v,F_g)=\{v\}$,在同一个广义自由区的顶点 $\{u,v,w\}$,其最小 Steiner 树上的 Steiner 点记为 $StPt(\{u,v,w\})$(对没有 Steiner 点的情况,设 $StPt(\{u,v,w\})$ 为 3 点中任意一点).

(3) 枚举各种互连方案,在这些方案中选取代价最小的,如图 12 所示.

$$\rho_{ST_G}=\min_{F_g\in P_g,\, u_{i,ji}\in B(v_i,F_g)}\sum_{i=1}^{3}\rho(v_i,u_{i,ji})+\rho(u_{i,ji},StPt(\{u_{1,j1},u_{2,j2},u_{3,j3}\})).$$

图 12 中为一种互连方案:v_1,v_2,v_3 之间的 Steiner 树由广义自由区 F_g 内部和自由区外部的路径组成.自由区 F_g 的边界为 $\{e_{j1},e_{j2},e_{j3}\}$,边界上的点 $u_{i,ji} \in B(v_i,e_{ji}), i=1,2,3$.

定理 12. $\rho(ST_{opt}(OB,\{v_i \mid i=1,2,3\})=\min\{\rho(MST(\{v_i\})),\rho_{ST_G}\}$,其中 $MST(\{v_i\})$ 为 3 个顶点在有障碍情况下的最小生成树,ST_G 为上面基于正规划分所求得的 Steiner 树.

得到一个顶点 v 到所有的广义自由区边界的最短路径可以简单地在 $\Theta(t)$ 时间内得到;对于一个广义自由区 F_g 中的 Steiner 树构造方案,直接与其边界的数量即 F_g 在 G_G 中的顶点度数相关,最坏情况下有 $O(t^3)$种,而平均情况仅为 $O(1)$.从而整个构造算法的最坏情况复杂度为 $O(t^4)$,平均情况为 $O(t)$.

注意到,最坏情况的产生在于构造方案的数量过大,从而一个有用的近似是:对不在广义自由区 F_g 内的顶点 v,取顶点 v 到广义自由区的边界 $bound(F_g)$ 上最近的顶点 u_{v,F_g} 作为在此广义自由区中构造最小 Steiner 树的顶

点,即 $\rho(SP(u_{v,F_g},v))=\min_{u\in bound(F_g)}\rho(SP(v,u))$;否则,令 $u_{v,F_g}=v$.取 $\rho'_{ST_G}=\min_{F_g\in P_g}\sum_{i=1}^{3}\rho(v_i,u_{i,F_g})+\rho(u_{i,F_g},StPt(\{u_{1,F_g},u_{2,F_g},u_{3,F_g}\}))$.这虽然不能保证得到最优解,但显然有较好的性能.注意到,该算法对每个广义自由区只求一次 Steiner 树,故其复杂度为 $\Theta(t)$.

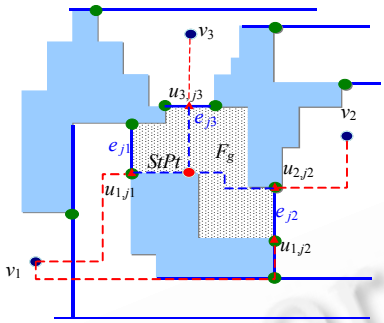


Fig.12 Construction of 3-Steiner tree based on formal partitioning
图 12 基于正规划分的 3 点间 Steiner 树的构造

4 结论和进一步的工作

本文中在各个连接图上所设计的算法的复杂度总结见表 3,可以看到,基于自由区的连接图在算法的平均性能和对空间的要求方面明显优于基于轨迹的连接图.进一步的工作方向包括:

(1) G_G 上求点对之间最短路径的查询以及 3 点 Steiner 树的最坏情况复杂度很高,是由于顶度数过大,能否设计出顶点度数为常数或对数的广义连接图,使得在最坏情况和平均情况下均有很好的性能;

(2) 算法的高效实现和实验分析;

(3) 在有障碍情况下求一般的最小 Steiner 树的快速算法.

Table 3 Comparison among four kind of connection graphs

表 3 4 种典型连接图的性能比较

		G_C	G_T	G_F	G_G
Querying shortest paths	Pretreatment	$O(e^4)$	$O(t^4)$	$O(t^2\log t)$	$O(t\log t)$
	Space complexity	$O(e^4)$	$O(t^4)$	$O(t^2)$	$O(t^2)$
	Length querying (worst)	$O(1)$	$O(1)$	$O(t^2)$	$O(t^2)$
	Length querying (average)	$O(1)$	$O(1)$	$O(\log^2 t)$	$O(\log t)$
	Path querying	$O(e^2)$	$O(t^2)$	$O(t\log t)$	$O(\log t)$
3-Steiner tree optimal algorithm	Time complexity (worst)	$O(e^2)$	$O(t^2)$		$O(t^4)$
	Time complexity (average)	$O(e^2)$	$O(t^2)$		$O(t)$
	Space requirement	$O(e^4)$	$O(t^4)$		$O(t^2)$
	Pretreatment	$O(e^4)$	$O(t^4)$		No need
Approximate algorithm	Time complexity (worst)			$O(t^2)$	$\Theta(t)$
	Time complexity (average)			$O(t\log t)$	$\Theta(t)$
	Space requirement			$O(t^2)$	$\Theta(t)$
	Pretreatment			$O(t^2\log t)$	No need

References:

[1] Lee CY. An algorithm for path connections and its applications. IRE Transactions on Electronic Computers, 1961,EC-10(2): 346~365.

[2] Akers SB. A modification of Lee's path connection algorithm. IEEE Transactions on Electronic Computers, 1967,EC-16:97~98.

[3] Soukup J. Fast maze router. In: Proceedings of the 15th Design Automation Conference. Las Vegas: ACM Press, 1978. 100~102.

[4] Hightower DW. A solution to line routing problems on the continuous plan. In: Proceedings of the 6th Annual Design Automation Workshop. ACM Press, 1969. 1~24.

[5] Clarkson KL, Kapoor S, Vaidya PM. Rectilinear shortest path through polygonal obstacles in $O(n\log^2 n)$ time. In: Proceedings of the 3rd Annual Symposium on Computational Geometry. New York: ACM Press, 1987. 251~257.

[6] Zheng SQ, Lim JS, Iyengar SS. Finding obstacle-avoiding shortest paths using implicit connection graphs. IEEE Transactions on Computer Aided Design, 1996,15(1):103~110.

[7] Wu YF, Widmayer P, Schlag MDF, Wong CK. Rectilinear shortest paths and minimum spanning trees in the presence of rectilinear obstacles. IEEE Transactions on Computers, 1987,36(3):321~331.

[8] Zhou Z. Obstacle-Avoiding minimum rectilinear Steiner tree problem [MS. Thesis]. Hefei: University of Science and Technology of China, 1998 (in Chinese with English abstract).

[9] Zhou Z, Chen GL, Gu J. Finding obstacle-avoiding shortest path using connection graph with $O(t\log t)$ edges. Chinese Journal of Computers, 1999,22(5):519~524 (in Chinese with English abstract).

[10] Ganley JL, Cohoon JP. Routing a multi-terminal critical net: Steiner tree construction in the presence of obstacles. In: Proceedings of the IEEE International Symposium on Circuits and Systems. London: IEEE Computer Society Press, 1994. 113-116.

[11] Hanan M. On Steiner's problem with rectilinear distance. SIAM Journal on Applied Mathematics, 1966,14(2):255~265.

[12] Garey MR, Johnson DS. The rectilinear Steiner tree problem is NP-complete. SIAM Journal on Applied Mathematics, 1977,32(4): 826~834.

[13] Ganley JL, Cohoon JP. A faster dynamic programming algorithm for exact rectilinear Steiner minimal trees. In: Proceedings of the 4th Great Lakes Symposium on VLSI. 1994. 238~241.

[14] Zhou Z, Jiang CD, Huang LS, Gu J. Finding obstacle-avoiding shortest path using generalized connection graph with $\Theta(t)$ edges. Journal of Software, 2003,14(2):166~174 (in Chinese with English abstract).

[15] Frederickson GN. Fast algorithms for shortest paths in planar graphs with applications. SIAM Journal on Computing, 1987,16(6): 1004~1022.

附中文参考文献:

[8] 周智.有障碍的 Manhattan 空间中的最小 Steiner 树问题[硕士学位论文].合肥:中国科学技术大学,1998.

[9] 周智,陈国良,顾钧.用 $O(t\log t)$的连接图求有障碍时的最短路径.计算机学报,1999,22(5):519~524.

[14] 周智,蒋承东,黄刘生,顾钧.用$\Theta(t)$的广义连接图求有障碍时的最短路径.软件学报,2003,14(2):166~174.

第 1 届全国人工智能教育学术研讨会

征文通知

由中国人工智能学会主办，首都师范大学承办，北京航空航天大学、北京工商大学、天津师范大学协办的第一届全国人工智能教育学术研讨会将于 2003 年 12 月 13~15 日在北京召开。这次会议是中国人工智能学会主办的首届全国人工智能教育学术研讨会，也是我国人工智能教育工作者在新世纪的第一次盛会。它一定会对我国人工智能教育事业的发展起到积极的促进作用。

会议主题：信息时代的智能教育

主要内容：

1. 国内外人工智能教育的现状及我国人工智能教育改革的讨论；
2. 不同层次、不同专业的人工智能教学要求、教学计划和教学大纲的讨论；
3. 人工智能课程的教学经验和教材建设经验交流；
4. 智能教学软件和教学系统交流；
5. 关于试办人工智能本科专业的问题的讨论；
6. 其他一些与人工智能教育有关的问题；
7. 人工智能基础、理论、技术和应用研究的进展。

报到地点：北京航空航天大学招待所 **报到时间：**12 月 13 日全天

时间安排：12 月 14 日上午开幕式、特邀报告,下午主题发言和分组交流
12 月 15 日上午分组交流,下午自由交流或参观

联系方式：

联系地址：100037 北京市西三环北路 105 首都师范大学信息工程学院
联 系 人：谢达 王万森 葛庆平 联系电话：010-68416830， 68903443， 68901041
E-mail: xieda89@263.net wangwansen@263.net wangws@mail.cnu.edu.cn