最短路径算法加速技术研究综述

宋 青，汪小帆

(上海交通大学电子信息与电气工作学院 上海 闵行区 200240)

【摘要】最短路径的快速有效计算研究具有重要的实际意义。经典算法的高计算复杂度制约了其在大规模网络中的应用。该文从以优先队列为代表的基本加速技术、目标引导技术以及分层技术3个方面综述了该领域最新、最具代表性的一些算法，包括作者在网络分层模型的构造及其分层搜索算法设计方面的最新成果。最后展望了该领域的未来研究方向。

关 键 词 启发式; 分层; 大规模网络; 最优化; 最短路径
中图分类号 TP393；U491 文献标识码 A doi:10.3969/j.issn.1001-0548.2012.02.002

Survey of Speedup Techniques for Shortest Path Algorithms

SONG Qing and WANG Xiao-fan

(School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University Minhang Shanghai 200240)

Abstract The problem of efficient path computation arises in many domains of practical importance. Classical algorithms failed to apply to large-scale networks due to their heavy computational complexity. This paper reviews some latest representative results classified according to the underlying speedup techniques, including basic speedup techniques like priority queues, goal-directed techniques, and hierarchical approaches. Moreover, the paper introduces the most recent achievement of the authors on network hierarchy construction and hierarchical algorithm design. Finally, some future directions are discussed.

Key words heuristics; hierarchical; large-scale networks; optimization; shortest paths

最短路径问题一直是运筹学、计算机科学、地理信息科学、交通运输等领域的一个研究热点[1-2]。很多实际问题都可以通过抽象进而转化为网络中的最短路径计算问题，如道路交通网络中的出行路线选取问题，计算机网络中信息流在路由器间的最佳传输问题，以及社会关系网络中两个陌生人之间的联系紧密度计算问题等。因此，针对网络中最短路径的有效计算问题研究具有重要的理论和现实意义。继文献[3]针对Dijkstra算法的开创性工作之后，大量专家学者围绕该问题进行了深入研究。特别是近年来伴随着大规模动态网络数据的出现，最短路径的实时计算面临着新一轮的挑战。很多实际系统(如GPS导航系统以及消防、救灾等应急系统)都要求在尽可能短的时间内获取最佳路径，以更好地应对网络动态等方面的影响，为用户提供快速决策。所有这些都依赖于更高性能的最短路径算法。受人工智能等领域一些思想的启发[4-7]，一系列针对最短路径求解的启发式加速方案应运而生，经典算法与不断发展完善的计算机数据结构及各类加速技术的有效结合使得新的最短路径算法不断涌现。它们在空间复杂度、时间复杂度、易实现性及应用范围等方面各具特色[8-12]。

本文首先简单回顾了以标号算法为代表的经典最短路径算法的基本思想及其具体实现，然后根据使用的加速技术的不同对现有的最短路径算法研究进行了系统分类和详细评述。最后，针对最短路径算法的发展方向进行了探讨。

1 最短路径问题及经典算法

实际网络在数学上可以抽象成一个图$G=(V, E, W)$，其中V为节点集合，E为边集合，W为边的权重集合。每个节点代表系统中的一个确定目标，如道路交通网络中的某个道路交叉口，因特网中的某一路由器等。连边$e=(i, j)$表征节点i、j对应的网络目标之间的连接关系，可以是有向的或无向的，其权重$w_{ij}\in W$刻画了该连边上某种属性的强度。以道路交

收稿日期：2011－08－04；修回日期：2011－12－27
基金项目：国家自然科学基金(61074125)；国家973计划(2010CB731400)
作者简介：宋青(1983－)，女，博士生，主要从事复杂系统建模与优化方面的研究.

通网为例，连边表示对应的地点之间有道路相连接，而边的权重则可以描述道路的长度、车辆通过该道路的通行时间以及通行费等不同信息。假定从源节点s到目标节点t的路径P为一组首尾相接的连边序列：$(s, i), \cdots, (j, t)$，P的“长度”定义为这一系列连边的权重的累加和。最短路径问题即在给定网络中找出从源节点s到目标节点t满足“长度”最短的一条路径。

作为图论与网络优化研究的核心内容之一，最短路径问题的研究已经经历了半个多世纪的发展历程。标号算法作为目前研究最为成熟、应用最为广泛的最短路径算法族[11-12]，也是绝大多数最短路径算法的核心部分。它在路径搜索的过程中为每个节点i维护距离标号$L_{(i)}$和前驱标号$P_{(i)}$，分别用来存储当前计算的从源节点s到节点i的路径长度和该路径中节点i的前一节点，算法迭代过程中通过不断修改这些标号来求取最优路径，其主要流程如下：

1) 初始化。源节点设置为：$i=s$，$L_{(i)}=0$，$P_{(i)}$为空；所有其他点：$L_{(j)}=\infty$，$j\neq i$；候选节点集$Q=\{i\}$。

2) 节点选择。从候选集Q中选取一个节点i并将其从Q中移除。

3) 节点松弛。对以节点i为起始端的所有连边$e=(i, j)$执行松弛操作，即判断下列不等式是否成立：$L_{(i)}+w_{ij}<L_{(j)}$；如果条件满足，则更新当前存储的到节点j的路径长度为$L_{(j)}=L_{(i)}+w_{ij}$，更新节点j的前一节点$P_{(j)}=i$，并将节点j加入到候选集Q中。

4) 终止规则。如果集合Q为空，则算法完成，退出搜索；否则，转到2)再继续。

根据从集合Q中选取当前扫描节点的策略的不同，标号算法又可以分为标号设定(LS)和标号修正(LC)两大类，其代表分别为Dijkstra[3]和Bellman-Ford算法[13]。已有的绝大多数最短路径算法均可以视为在该算法流程基础上，通过采用不同的数据结构、节点选取策略以及生成树更新技术而拓展出的不同实现形式。下面将结合具体加速技术的不同对其进行分类。

2　基本加速技术

2.1　优先队列

优先队列是一个由0个或多个元素组成的集合，其中每个元素都有一个对应的优先权或值，并按优先权大小顺序存放。优先队列同时支持插入元素、删除最小值元素和修改键值几种运算，从而可以更好地实现对候选集Q中元素的管理。通过使用不同形式的优先队列来改进算法的运行结构，可以从一定程度上提高算法的执行效率。

事实上，只有当采用最原始的无序列表来存储候选集Q时，Dijkstra算法的时间复杂度才为$O(n^2)$。当采用二叉堆或Fibonacci堆来实现Dijkstra算法时，其时间复杂度分别降为$O(m\log n)$和$O(m+n\log n)$。通过利用各种高效的不同形式的堆和桶结构优先队列来实现Q，可以得到更多的LS算法[11,14-15]。同样，对于LC算法而言，采用不同的列表实现方式可以得到更多的具有不同时间复杂度的LC算法[13,16-20]。

2.2　数据预处理

数据预处理是一种典型的以空间换时间的做法。由于预先计算并存储所有节点间的最短距离将耗费巨大的存储空间，甚至会引发内存/硬盘溢出以及数据查询效率的降低，因此在实际应用中更加倾向于通过预处理少量的重要信息来加快路径的搜索：一方面，数据预处理环节要足够快速，同时预处理生成的待保存的数据量要足够少；另一方面，最短路径的实时计算过程能得到有效的加速。这两个目标本质上是相互矛盾的，而如何在这二者之间寻找一种比较好的折衷则是目前大量研究所探讨的重点。

数据预处理技术通常与其他加速技术配合使用。如在目标引导搜索算法中，通过预处理部分节点间距离来改善目标引导项[21-22]，进而减小算法的搜索空间。此外，数据预处理技术在分层搜索算法中的应用更为广泛，文献[23-26]利用了网络分层来获取其中的重要节点和边的信息，然后对最高层网络中的重要节点间的距离进行预处理和存储。

2.3　双向搜索

在经典最短路径算法中搜索是单方向的，算法的执行时间取决于搜索过程中所遍历的节点数目，对于二维情形，其搜索空间可以近似表示为以s-t为半径的圆面。如果同时从s和t开始构造最短路径树[27]，那么问题的规模就会相应地减小。双向搜索技术就是通过把初始问题分解成两个对等的子问题分别处理达到减小算法搜索空间，加速算法执行的目的。各类加速技术算法的搜索空间如图1所示。

双向最短路径算法的效率与精度受前、反向搜索的交替执行方式及算法的终止规则的影响[8]。一般来说，从初始点和目标点两边同时搜索可以改善搜索效率，但是单独应用双向搜索技术的加速效果并不理想，在某些情况下反而可能比单向搜索差[28]。

研究表明双向搜索技术与分层技术配合使用时算法的性能较好[29-30]。此外，当对算法精度要求不严格时，可以对双向搜索的终止规则进行弱化以进一步加快算法的执行效率，如文献[29, 31]将终止规则修改为前、反向最短路径树上至少有γ个公共节点，然后通过调整γ值的大小来控制算法的精度和效率，寻求它们之间的折衷。

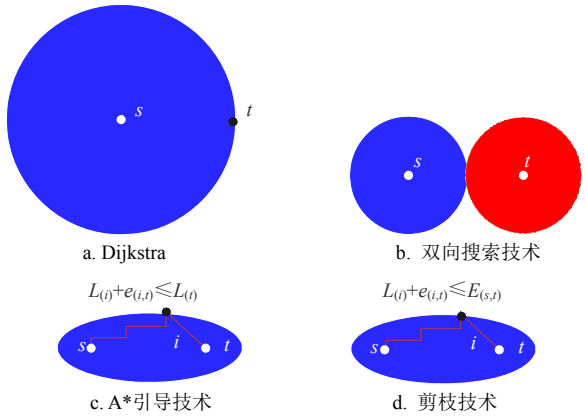


图1 基于不同加速技术的Dijkstra算法搜索空间示意图

3 目标引导技术

在经典最短路径算法中搜索是盲目的，从s出发的最短路径树沿着各个方向均匀对等地增长，从而在s-t的相反方向也同样遍历了大量节点，无论从时间还是空间上都造成了大量浪费，因此如果能够综合利用网络的结构特征、初始点与目标点的位置信息以及路径构成等先验知识来对搜索过程加以正确引导，使其沿着目标的方向逐步逼近进行搜索，那么势必节省大量的搜索空间，加快搜索算法的执行，这也是目标引导技术的出发点。目前针对最短路径问题的求解已经相继出现了一系列引导策略以不断提高搜索的效率和精度。根据使用的具体策略的不同将其划分如下：

3.1 基于A*的引导技术

A*算法[4]是一种典型的最佳优先搜索算法。它在搜索过程中使用估价函数对每一个节点进行评估，其估价函数定义为：

$$F_{(i)} = L_{(i)} + e_{(i,t)} \tag{1}$$

式中，$F_{(i)}$是节点i的估价值；$L_{(i)}$是从初始点s到节点i的最短路径值；$e_{(i,\,t)}$是从节点i到目标点t的最短路径的估计代价。当$e_{(i,\,t)}$趋于0时A*退化为标准LS算法。显然，A*的执行过程只需在上述算法流程的基础上将第3步修改为：

3) 节点松弛。对以节点i为起始端点的所有连边e=(i, j)执行松弛操作，即判断下列不等式是否成立：$L_{(i)}+ w_{ij}+ e_{(j,\,t)} < F_{(j)}$；如果条件满足，则更新当前存储的到节点$j$的路径长度为$L_{(j)}= L_{(i)}+ w_{ij}$，更新节点$j$的估价值为$F_{(j)} = L_{(i)}+ w_{ij}+ e_{(j,\,t)}$，更新节点$j$的前一节点$P_{(j)}= i$，并将节点$j$加入到候选集$Q$中。

基于A*的引导算法的关键在于选取一个恰当的估价函数。当估价值$F_{(i)}$从不高估节点s-t间的实际最优路径值时算法必能得到最优解，此时的启发式估价函数称为是可接纳的(admissible)，且估价值越接近实际最优路径值，估价函数选取得就越好，算法的搜索效率也就越高。而当估价值大于实际最优路径值时，算法搜索的节点数明显减少，搜索效率也显著提高，但是不能保证得到最优解，文献[32]给出了其解的精度与估价函数之间的近似关系。

估价函数有很多不同的定义方式，从最初的使用欧式距离加以近似，到后期的通过与其他加速技术或思想的结合来进行递推，总体上是为了让$e_{(i,\,t)}$与真实值更加逼近。文献[33]将机器学习的观点引入到A*算法中，利用先前搜索过的节点间的路径值来估计$e_{(i,\,t)}$的值。文献[34]将人工智能领域的前向启发思想融入其中，使用下一个节点到目标点的最佳路径的估价值来估计当前$e_{(i,\,t)}$的值，其计算公式为：

$$e_{(i,t)} = \min_{\forall j}\{w_{ij} + e_{(j,t)} \mid (i,j) \in E\} \tag{2}$$

文献[21]结合数据预处理技术并应用三角不等式原理来获取$e_{(i,\,t)}$的表达式，提出了一种基于地标的A*算法(ALT算法)：通过从网络中选取少量节点作为地标并预先计算、存储所有节点i与地标L间的距离$d(i, L)$和$d(L, i)$，可以由三角不等式原理$d(L, t)$–$d(L, i)\leqslant d(i, t)$和$d(i, L)d(t, L)\leqslant d(i, t)$进而推得$e_{(i,\,t)}$的表达式为：

$$e_{(i,t)} = \max_{\forall L}\{(d(L,t) - d(L,i)),\ (d(i,L) - d(t,L))\} \tag{3}$$

3.2 基于剪枝的引导技术

剪枝技术的基本思想是通过淘汰不太可能出现在最优路径上的节点来限制搜索区域，以达到减小算法计算复杂度的目的。剪枝技术的关键是如何找出这一区域，使得算法的计算过程能得到有效的加速，同时保证得到一条较优的路径。文献中关于搜索区域的限制有着不同的定义方式[22,29,31,35]，但大致可以概括为如下不等式：

$$L_{(i)} + e_{(i,t)} \leqslant E_{(s,t)} \tag{4}$$

式中，$L_{(i)}$是从初始点s到节点i的最短路径值(即i的距离标号)；$e_{(i,\,t)}$是从节点i到目标点t的最短路径估价值；$E_{(s,\,t)}$是从初始点s到目标点t的距离估计上限值。

基于剪枝技术的最短路径算法的执行过程跟经典算法大致相同，只是在“节点选择”环节需要增

加对所选取节点的位置的判断，看其是否在限定区域内部，如果不在则没有必要对该节点执行任何操作，即将上述标号算法主要流程中的第二步修改为：

2) 节点选择。从集合Q中选取一个节点i并将其从Q中移除。如果节点i满足不等式$L_{(i)}+e_{(i,t)}>E_{(s,t)}$，则跳转到4)。

显然，基于剪枝技术的最短路径算法的效率与精度取决于估价函数$e_{(i,t)}$和$E_{(s,t)}$。当$e_{(i,t)}$总是低估节点i-t间的真实距离，同时$E_{(s,t)}$总是高估节点s-t间的真实距离时，算法能保证获取最优解——估价值与真实值的偏差越大，限定的搜索区域就越大，算法的效率也会越差，当$e_{(i,t)}$趋于0而$E_{(s,t)}$趋于无穷时，算法退化为经典最短路算法，此时式(4)对搜索区域的限制不再起作用；反之，算法不能保证最优解，而且偏差越大限定的搜索区域会越小，有时会出现算法终止时搜索仍未到达目标点t的情形[31]。

文献[35]提出了一种能获取最优路径的两阶段剪枝算法：第一阶段使用A*搜索算法来计算从初始点s到目标点t的距离，并用其作为估计上限$E_{(s,t)}$，而节点i-t间的距离估计$e_{(i,t)}$则使用欧式距离进行近似；第二阶段则等同于一般的剪枝算法。文献[22]结合数据预处理技术和双向搜索技术，提出了一种利用预处理簇间距离来对搜索过程加以限制的剪枝算法——PCD算法。算法首先将网络分割成若干个簇，对任意两个簇I、T，计算它们之间的最短连接距离$d(I,T):=\min\{d(u,v)|\ u\in I,v\in T\}$，然后将该距离值连同对应的最短路径的两侧端点$s_{IT}$、$t_{IT}$保存起来。在算法执行过程中，每当搜索到达前侧端点$s_{IT}$时，更新$s$-$t$间距离估计的上限$E_{(s,t)}$；此外，在算法迭代的每一步，计算从当前节点$i$到目标点$t$的最短距离估计$e_{(i,t)}$并进行判断，如果式(4)不成立，则从节点$i$出发的搜索将被终止。PCD算法的估价函数定义为($B(T)$是$T$的边界点)：

$$e_{(i,t)}=d(I,T)+\min_{t'\in B(T)}L^{t}_{(t')} \tag{5}$$

$$E_{(s,t)}=L^{s}_{(s_{IT})}+d(I,T)+L^{t}_{(t_{IT})} \tag{6}$$

由于$e_{(i,t)}$始终小于节点i-t间的真实距离，而$E_{(s,t)}$始终大于节点s-t间的真实距离，因而PCD算法能保证得到最优解。

此外，关于搜索区域的限制还有一些不同的方式。如文献[36]使用矩形区域来限定搜索范围，在路径搜索过程中位于矩形区域外的节点则被忽略不计。该方法针对初始点与目标点间距离较近、结构相对规则的网络效果较好，而对初始点与目标点间距离较远的情形其区域限制效果并不明显。显然，基于矩形搜索区域的最短路径算法不能保证解的最优性，同时也会出现搜索过早终止而找不到目标点这一现象。

文献[29]综合利用了数据预处理、分层和双向搜索技术，提出了一种将搜索限制在部分子网络内部的方法。算法首先利用社团识别技术对网络进行分割降解，针对初始点与目标点位于同一子网络内部的情形，采用对搜索区域重构的办法来计算最优路径(重构的搜索区域由该子网络内部的节点、边以及预处理阶段修正的边构成)；而对于初始点与目标点位于不同子网络的情形，定义了子网络坐标并使用欧式距离来近似估计子网络间距离，通过选取距离较近的子网络对并将搜索范围限定在其内部，达到了加速算法执行的目的。针对网络中部分边的权值发生变化这一动态情形，利用模块度Q[37]来评判网络的分割质量，仅当模块度的增益ΔQ超出允许值时才对网络重新进行社团的划分和网络分层构造。当ΔQ在允许范围内时仅对包含权重变化边的社团进行重构，更新成本非常低，从而使得该方法可以较好地推广到动态情形。

3.3　基于路标的引导技术

基于路标的算法设计思想仿照了日常生活中路标的导向作用，即告诉行人该条道路通往哪些地方，从而使行人可以更加方便快捷地到达目的地，避免了绕道或者走错道。在路标类算法中，针对网络中的每条边引入一个路标(Signposts)，用来指示网络中经由这条边的最短路径所能到达的节点的大致范围，从而在最短路径算法的执行过程中仅需要考虑那些能通往目标节点的边，减少了不必要的搜索，实现了加速。目前文献中所普遍关注的路标类算法主要有几何容器法和边标识法两种。

几何容器法[38]的基本思想是针对网络中的每条边$e=(i,j)$预处理一个几何容器$C(e)$，使之至少容纳从节点i出发且以e为起始边的最短路径所能到达的所有节点，从而将保存每条边的这部分节点的问题转化为保存某个容器参量，进而节省了空间占用。在算法的执行过程中仅需要松弛那些对应容器中包含目标点的边。当然，几何容器的引入会同时带来部分额外节点，这在一定程度上会降低算法的搜索效率，但是并不会影响其解的最优性，如图2所示，其中白色节点表示从节点i出发且以e为起始边的Dijkstra搜索分支上的节点。大量文献针对几何容器的构造方式进行了研究[38-40]，尽可能地消除额外节

点。几何容器法可以与其他加速算法配合使用[41-42]，但是由于其预处理成本非常高，目前尚未有大规模网络方面的应用。

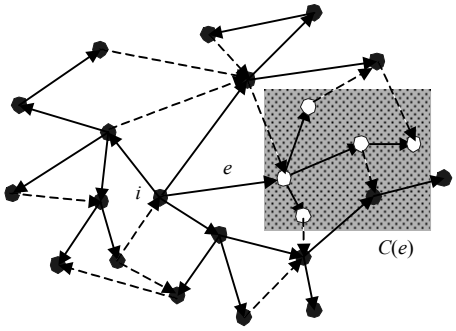


图2 几何容器示意图

文献[43-46]提出了一类可应用于大规模网络的路标算法——边标识法。算法首先将网络分割成k个区域，然后针对网络中的每条边e添加相对于所有区域r的标识$f_e(r)$→{true, false}，r∈{1, 2,…, k}。如果以e为起始边的最短路径能到达区域r中的点则令$f_e(r)$的值为真(true)，反之为假(false)。在搜索算法的执行过程中仅需要考虑那些相对于目标点t所在区域的标识为真的边，而其他的边则忽略不计。边标识法的计算性能受网络分割技术及分割后的区域数目k的影响[43-44]。随着区域数目k的增多算法的搜索空间会逐渐减小，但与此同时算法所耗费的预处理时间以及存储空间会随之增长。通过调节k值大小，可以实现在加速比与存储空间消耗二者之间的折衷。边标识法的预处理成本相比几何容器法要低，只需要从区域边界点执行单源最短路径计算即可对所有边的标识作出判断。文献[46]使用了更为巧妙快速的标识判断方法，针对每一个区域仅执行了一次单源最短路径计算。边标识法可以与其他加速技术结合使用。如文献[33]将边标识法拓展到多层网络中使得算法的效率得到进一步提高，同时占用了更少的存储空间。

3.4 基于潜在目标的引导技术

基于潜在目标的算法设计的大致思想是在初始点s与目标点t之间依次预先设定几个潜在的目标点$g_1, g_2,\cdots, g_r$，将初始时s-t间的路径计算问题降解为若干个子问题分别处理(如s-g_1, g_1-g_2,…, g_r-t)，从而达到减小整个算法复杂度的目的，在一定程度上也是一种分治的做法。潜在目标一般是那些最有希望出现在最优路径上的节点或边，在道路网最短路径问题中通常选取为重要的交通枢纽或者是根据用户偏好而指定的一些标志性建筑、中转站点等。

然而，基于潜在目标的引导算法不一定是最优的，其计算精度取决于这些潜在目标点的选取质量。文献[24]针对节点加权网络设计了一种启发式引导算法。算法首先将搜索的范围限定在一系列潜在的子网络中，然后从这些顺次相邻的子网络中依次选取距离前一点(初始时为源节点)欧式距离最近的边界点以及与它相邻的权值最小的边界点作为潜在目标点，并通过分别在对应的子网络内部的最短路径计算将整条路径拼接起来，该算法的精度尚有待于进一步验证[47]。文献[30]提出了一种预处理中转节点以获取最优路径的算法：针对网络中的每个节点，预先计算它与对应的中转节点间的距离，以及所有中转节点间的距离表，从而将最短路径的计算问题转化为已保存数据的叠加和比较过程，实现了加速，其本质是子目标法与数据预处理、双向搜索以及分层技术的结合。

4 分层技术

分层设计思想是在求解复杂问题时首先抓住问题的关键点而忽略其他次要的细节性部分，随后再对细节部分加以完善。这一过程类似于司机的路线选取行为：为了获取s、t两地间的路线，司机通常凭借经验先确定从s周边通向t周边的主要干道，然后再分别找出从s和t通往主干道的支线道路[8]。由于分层策略在降低算法复杂度和抑制问题计算时间随规模呈非线性增长等方面所表现出来的优良性能[48]，目前已成为大规模网络最短路径计算中所普遍使用的一种加速技术。

根据网络分层的具体构造方式，可以将现有的分层搜索算法大致分为以下两类。

一类是利用网络内在的特性来构造分层并进行算法设计[8, 23, 49-52]，主要以道路交通网中的路径导航类算法为代表。由于道路网通常表现出自分层特性，快速、远距离路径往往由等级高的道路例如高速公路、主要干道来提供，而等级低的道路如次要干道和乡镇公路则是为局部区域服务的，因而很多文献中都利用了这种特性来提取网络分层。如文献[49]将主要干道和高速公路抽象为高一级网络，构建了一个两层网络分层结构。文献[50]利用道路的长度作为分类依据，将长距离道路定义为高一级网络。还有文献[23]通过对道路网进行化简，去除中间节点及岔道来获取高一级网络。总之，不论采取哪种抽象方式，其构造的各级网络都必须是连通的。为了保证连通性，有时需要专家凭借经验判断作出调整[49]，因而这类分层方式的随机性因素较多，预处理过程相对复杂。

在这类分层模式下，高一级网络将其对应的低一级网络分割成若干个子网络，如图3所示，各邻接子网络之间相互交叠，其中高一级网络的道路交叉点被定义为入口点(entry points)，最短路径的搜索过程就是沿着入口点实现了各级网络分层间的切换。当然，其搜索算法的具体实现可以有不同的形式，包括与前面介绍的各类加速技术的结合。文献[53-54]采用最近入口点切换的方式，首先在原网络中分别以初始点和目标点为树根构造最短路径树直到距离它们最近的入口点已遍历，然后切换到高一级网络中来计算入口点间的最短距离。但是，基于最近入口点切换的方式不一定是最优的，有时会存在较大误差，因而文献[49]中考虑了初始点与目标点周围的所有入口点，将初始点、目标点所在的子网络与高一级网络拼接在一起，随后使用Dijkstra或A*算法来搜索最佳路径。与此类似，文献[23]对高一级网络进行扩充，将初始点和目标点通过虚拟边与所在高一级子网络的入口点连接起来(虚拟边的权重等于在原网络中子网络内部计算的最短路径值)，从而将两层网络上的最短路径搜索转化为单层问题，并通过引入剪枝技术实现了算法的进一步加速。值得注意的是，使用入口点进行搜索切换在某些情形可能会错失最优路径。如初始点与目标点位于相邻子网络内部且它们之间有小路(minor roads)相连，此时的最优路径可能并不经过入口点，而是直接由这些小路拼接而成，如文献[23]。针对此类情形，文献[23, 49, 53]给出了一定的解决方案，对初始时的道路分层结构作出进一步的调整。

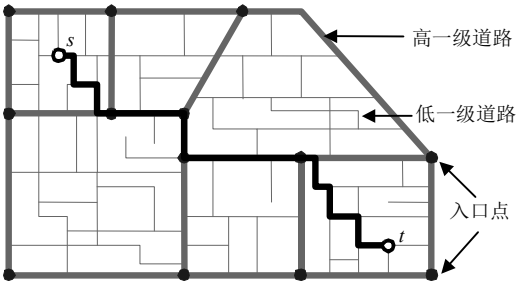
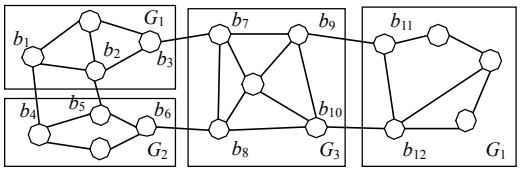


图3　基于网络特性的两层网络分层示意图

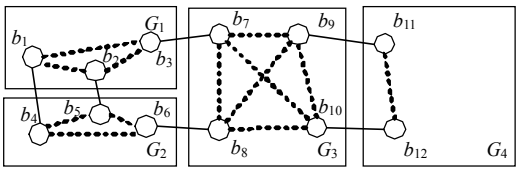
此外，仿照道路网等级特性的分层方式，文献[30, 51]提出了一种通用的(不局限于道路网)同时保证精确路径的分层构造方案：通过固定以每个节点为树根的局部最小生成树的节点个数来定义各节点的邻域，将网络中所有节点对*s*-*t*间的最短路径上不完全属于*s*和*t*的邻域范围的部分定义为“高速公路”，从而最短路径的计算过程只需要在原网络中初始点与目标点的邻域内进行，然后即可跳转到“高速公路”上继续搜索。

另外一类则是基于对网络的有效分割构造分层并进行算法设计[24-26,55-57]。有效分割的目的是为了降低算法的预处理成本、提高其执行效率，或者是满足某些特定性能要求。如文献[56]指出一个好的分割算法应该保证分割后的各个子网络的规模大致均衡，各子网络的边界点(border nodes)数目大致相等且总的边界点数最小等等。由于很多网络都近似平面结构，因此有关平面图形的分割算法通常用来对其分割降解[44]。此外，目前文献中使用的性能较好的分割技术还有SPC[55]、METIS[58]、*k*-中心聚类[22]和社团识别[37]，具体使用哪种分割技术时效果最佳因搜索算法而异。

与前面那类分层模式不同，这里分割后的各个子网络是相互独立的，子网络之间通过少量的边(intergroup arcs)联系起来，这部分边以及构造的子网络内部边共同组成了高一级网络分层，如图4所示。当然，构造的边的类型不同，分层模型的描述也不同[24-26,29,47]。同样，基于该类模型的分层搜索算法也具有不同的实现形式，但总的来说都是先在原网络中初始点与目标点所在的子网络内部搜索，然后通过边界点跳转到高一级网络中继续搜索，直到终止规则成立。算法的具体差别源于所使用的预处理方案或者是加速技术的不同。如文献[26]构造了各子网络内部所有边界点间的连边，并使用A*算法来加快高一级网络中的最短路径搜索。文献[25]结合预处理距离表优化的策略，针对各个子网络中所有节点间的距离以及所有边界点之间的距离进行预处理和存储，从而将最短路径的实时计算过程转化为已保存数据的查询、叠加和比较的过程，因而非常快速，其缺点是耗费了大量预处理时间和存储空间。



a. 原网络和4个子网络G_1-G_4，其中b_1-b_{12}为边界点



b. 高一级网络(虚线表示构造的子网络内部边)

图4　基于网络分割的两层网络分层示意图

当然，关于网络分割及其分层设计还有一些其他方式。文献[47]从一种理想的路径计算模式出发，提出并设计了最小化穿越距离比的网络分割算法，

取得了良好的性能。通过提取分割后的网络中的边界点、子网络之间的连边并构造各子网络内部边，以及通过将各子网络及其邻接关系映射为节点和边，分别构建了高一级网络分层和抽象网络分层，并基于该分层模式设计了一种通用的、能获取最优路径的HSP分层路由算法。算法综合利用了数据预处理、分层和剪枝技术，在其执行过程中通过维护和更新初始点s与目标点t之间的距离值的上限$\hat{d}(s,t)$，计算s-t间经由节点u(最短路径搜索由节点u离开各个子图)的距离估计下限$\underline{d}(s,u,t)$来实现对搜索区域的限定，一旦距离估计下限$\underline{d}(s,u,t)$超过了上限值$\hat{d}(s,t)$，那么从节点u出发的搜索以及此后所有指向节点u邻居子图的搜索即可终止。此外，还给出了该网络分割及分层方法在动态情形的拓展实现方式。

总之，分层技术已成为目前针对大规模网络最短路径计算问题进行降解、简化的重要手段，而且常与目标引导技术、数据预处理技术以及双向搜索技术结合使用。特别是在基于网络分割构造分层的这类搜索算法设计中，由于构造的高一级网络分层中节点的度通常会增大(如文献[26]中每个子网络近似为一个完全图)，如果直接采用Dijkstra算法进行搜索其效率势必降低，因此文献中普遍采用预处理距离表优化或目标引导的方法来对搜索加以限制。分层搜索算法的效率受多方面因素的影响，包括预处理阶段生成并保存的数据量的多少、网络分层的拓扑结构以及搜索规则等等；而算法的精度主要取决于目标引导项的强弱及准确程度。一般来说，算法的效率会以一定程度上牺牲精度和存储空间占用为代价，从而如何在算法的预处理/存储消耗与实时效率以及算法的实时效率与精度之间寻找一种较好的折衷，成为大多数分层算法所探讨的重点。

5 结 论

当前的最短路径算法研究正趋于向着各类加速技术、加速方法间的交叉结合的方向发展。正如文中所介绍的，很多方法单独应用时的加速效果并不显著，只有当相互间有序结合时才能使各自的优势得以充分体现。

尽管目前针对最短路径问题的快速算法研究已经取得了一些突破和成果，并伴随有大规模真实网络中的应用，但是，总体上该研究仍然处于发展中阶段，还有很多工作需要进一步探索和认识。

首先，近年来的大量研究都是以道路交通网络作为研究载体，通过利用道路网的一些内在特性来加速最短路径算法的执行。由于建立在单一网络类型之上，使得这部分算法的应用往往具有一定局限性。针对更为一般实际网络的通用最短路径算法还有待于进一步的研究。

其次，现有最短路径算法在搜索效率、计算精度以及存储消耗方面通常各有利弊，很多算法效率的提高都是以牺牲大量预处理或搜索精度为代价，而实际系统中客户端的配置往往不允许存储过多数据，同时又要给用户提供较准确的快速决策，因而如何挖掘有效的启发式引导项来更好权衡这三者之间的关系，以及如何将最短路径算法加速技术与近年来网络科学中关于网络拓扑结构的研究[59-60]结合起来以设计更为有效的算法仍然值得进一步探索。

再次，在基于网络分割构造分层的这类最短路径算法设计中，网络分割质量的好坏对后续的搜索算法的执行效率有着一定的影响作用。现有研究中普遍使用平面图形分割技术来实现对网络的分割降解，而现实世界中的网络通常呈现非平面结构，因而针对这类网络的快速有效分割技术还有待于深入研究。此外，不同网络分割下搜索算法的具体性能也值得进一步的对比分析。

最后，现有的启发式加速算法研究多数侧重于解决静态网络中的最短路径问题[8-9]。然而，实际网络中边的权值通常是动态变化的，因而如何将各类启发式加速策略合理地拓展到动态情形(包括如何对网络分割及预保存数据进行局部更新并做出快速反应)，设计适应于动态网络的快速路径算法也是未来研究的趋势所在。

参 考 文 献

[1] BERTSEKAS D P. Network optimization: continuous and discrete models[M]. Belmont, MA: Athena Scientific, 1998.
[2] TARJAN R E. Data structures and network algorithms[M]. Philadelphia, PA: SIAM, 1993.
[3] DIJKSTRA E W. A note on two problems in connexion with graphs[J]. Numerical Mathematics, 1959, 1: 269-271.
[4] HART E P, NILSSON N J, RAPHAEL B. A formal basis for the heuristic determination of minimum cost paths[J]. IEEE Trans Syst Sci Cybern, 1968, SSC-4(2): 100-107.
[5] NILSSON J N. Problem-solving methods in artificial intelligence[M]. New York: McGraw-Hill, 1971.
[6] NEWELL A, SIMON H A. Human problem solving[M]. Englewood Cliffs, NJ: Prentice-Hall, 1972.
[7] PEARL J. Heuristics: Intelligent search strategies for computer problem solving[M]. Boston, MA, USA: Addison-Wesley Publishing Company, 1984.
[8] FU L, SUN D, RILETT L R. Heuristic shortest path algorithms for transportation applications: state of the art[J].

Comput Oper Res, 2006, 33: 3324-3343.
[9] DELLING D, SANDERS P, SCHULTES D, et al. Engineering route planning algorithms[J]. In Algorithmics of Large and Complex Networks, 2009, LNCS 5515: 117-139.
[10] BAUER R, DELLING D, SANDERS P, et al. Combining hierarchical and goal-directed speed-up techniques for Dijkstra's algorithm[J]. ACM Journal of Experimental Algorithms, 2010, 15(2.3): 1-31.
[11] DEO N, PAN C Y. Shortest path algorithms: taxonomy and annotation[J]. Networks, 1984, 14: 275-323.
[12] 陆锋. 最短路径算法: 分类体系与研究进展[J]. 测绘学报, 2001, 30(3): 269-275.
LU Feng. Shortest path algorithms: taxonomy and advance in research[J]. Acta Geodaetica et Cartographica Sinica, 2001, 30(3): 269-275.
[13] BELLMAN R. On a routing problem[J]. In Quarterly of Applied Mathematics, 1958, 16(1): 87-90.
[14] FREDMAN M L, TARJAN R E. Fibonacci heaps and their uses in improved network optimization algorithms[J]. J ACM, 1987, 34(3): 596-615.
[15] CHERKASSKY B V, GOLDBERG A V, RADZIK T. Shortest path algorithms: theory and experimental evaluation[J]. Mathematical Programming, 1996, 73: 129-174.
[16] PAPE U. Implementation and efficiency of Moore-algorithm for the shortest route problem[J]. Mathematical Programming, 1974, 7(2): 212-222.
[17] PALLOTTINO S. Shortest path methods: complexity, interrelations and new propositions[J]. Networks, 1984, 14: 257-267.
[18] GLOVER F, KLINGMAN D, PHILLIPS N, et al. New polynomial shortest path algorithms and their computational attributes[J]. Management, 1985, 31: 1106-1128.
[19] GOLDBERG A V, RADZIK T. A heuristic improvement of the bellman-ford algorithm[J]. Applied Mathematics Letters, 1993, 6: 4-6.
[20] BERTSEKAS D P. A simple and fast label correcting algorithm for shortest paths[J]. Networks, 1993, 23: 704-709.
[21] GOLDBERG A V, WERNECK R F. Computing point-to-point shortest paths from external memory[C]// Proceedings of the 7th Workshop on Algorithm Engineering and Experiments (ALENEX). Philadelphia: SIAM, 2005: 26-40.
[22] MAUE J, SANDERS P, MATIJEVIC D. Goal directed shortest path queries using precomputed cluster distances[J]. ACM Journal of Experimental Algorithms, 2009, 14(3.2): 1-27.
[23] JAGADEESH G R, SRIKANTHAN T, QUEK K H. Heuristic techniques for accelerating hierarchical routing on road networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2002, 3(4): 301-309.
[24] RAJAGOPALAN R, MEHROTRA K G, MOHAN C K, et al. Hierarchical path computation approach for large graphs[J]. IEEE Transactions on Aerospace and Electronic Systems, 2008, 44(2): 427-440.
[25] WANG Z, CHE O, CHEN L, et al. An efficient shortest path computation system for real road networks[C]// Proceedings IEA/AIE. Berlin: Springer, 2006: 711-720.
[26] JUNG S, PRAMANIK S. An efficient path computation model for hierarchically structured topographical road maps[J]. IEEE Transactions on Knowledge & Data Engineering, 2002, 14(5): 1029-1046.
[27] DANTZIG G B. On the shortest route through a network[J]. Management Science, 1960, 6: 187-190.
[28] DREYFUS E S. An appraisal of some shortest path algorithms[J]. Operations Research, 1969, 17: 395-412.
[29] SONG Q, WANG X F. Efficient routing on large road networks using hierarchical communities[J]. IEEE Transactions on Intelligent Transportation Systems, 2011, 12(1): 132-140.
[30] SCHULTES D. Route planning in road networks[D]. Karlsruhe: Universität Fridericiana, 2008.
[31] FU L. Real-time vehicle routing and scheduling in dynamic and stochastic traffic networks[D]. Edmonton, Alberta: University of Alberta, 1996.
[32] NICOSIA G, ORIOLO G. An approximate A* algorithm and its application to the SCS problem[J]. Theoretical Computer Science, 2003, 290(3): 2021-2029.
[33] BANDER J L, WHITE C C. A heuristic search algorithm for path determination with learning[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part A, 1998, 28(1): 131-134.
[34] ADLER J L. A best neighbor heuristic search for finding minimum paths in transportation networks[C]//Presented at the 77th Transportation Research Board Annual Meeting. Washington DC: [s.n.], 1998.
[35] LYSGAARD J. A two-phase shortest path algorithm for networks with node coordinates[J]. European Journal of Operational Research, 1995, 87(2): 368-374.
[36] KARIMI H A. Real-time optimal route computation: a heuristic approach[J]. ITS Journal, 1996, 3(2): 111-127.
[37] FORTUNATO S. Community detection in graphs[J]. Phys Rep, 2010, 486(3-5): 75-174.
[38] WAGNER D, WILLHALM T. Geometric containers for efficient shortest-path computation[J]. ACM Journal of Experimental Algorithms, 2005, 10(1.3): 1-30.
[39] WAGNER D, WILLHALM T. Geometric speed-up techniques for finding shortest paths in large sparse graphs[C]//11th Annual European Symposium on Algorithms (ESA). New York: Springer, 2003, 2832: 776-787.
[40] SCHULZ F, WAGNER D, WEIHE K. Dijkstra's algorithm on-line: an empirical case study from public railroad transport[C]//3rd Workshop on Algorithm Engineering (WAE). New York: Springer, 1999, 1668: 110-123.
[41] HOLZER M, SCHULZ F, ANDWILLHALM T. Combining speed-up techniques for shortest-path computations[C]//Experimental and Efficient Algorithms: Third International Workshop, RIBEIRO C C and MARTINS S L, Eds. New York: Springer, 2004, 3059: 269-284.

[42] WAGNER D, WILLHALM T, ZAROLIAGIS C. Dynamic shortest path containers[C]//3rd Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS'03). New York: Elsevier, 2004, 92: 65-84.
[43] KÖHLER E, MÖHRING R H, SCHILLING H. Acceleration of shortest path and constrained shortest path computation[C]//4th International Workshop on Efficient and Experimental Algorithms (WEA). Berlin: Springer, 2005: 126-138.
[44] MÖHRING R, SCHILLING H, SCHÜTZ B, et al. Partitioning graphs to speed up Dijkstra's algorithm[J]. ACM Journal of Experimental Algorithmics, 2006, 11(2.8): 1-29.
[45] KÖHLER E, MÖHRING R H, SCHILLING H. Fast point-to-point shortest path computations with arc-flags[C]//9th DIMACS Implementation Challenge. [S.l.]: [s.n.], 2006.
[46] HILGER M. Accelerating point-to-point shortest path computations in large scale networks[D]. Berlin: Technische Universität Berlin, 2007.
[47] SONG Q, WANG X F. Partitioning graphs to speed up point-to-point shortest path computations[C]//Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference. New York: IEEE Press, 2011: 5299-5304.
[48] KORF R E. Planning as search: a quantitative approach[J]. Artificial Intelligence, 1987, 33(1): 65-68.
[49] LIU B. Route finding by using knowledge about the road network[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part A, 1997, 27(4): 436-448.
[50] CHOU Y L, ROMEIJN H E, SMITH R L. Approximating shortest paths in large-scale networks with an application to intelligent transportation systems[J]. INFORMS Journal on Computing, 1998, 10(2): 79-163.
[51] SANDERS P, SCHULTES D. Highway hierarchies hasten exact shortest path queries[C]//Proceedings ESA Mallorca. New York: Springer, 2005: 568-579.
[52] SCHULTES D, SANDERS P. Dynamic highway-node routing[C]//6th Workshop on Experimental Algorithms. Berlin: Springer, 2007: 66-79.
[53] CAR A, FRANK A U. General principles of hierarchical spatial reasoning—the case of way-finding[C]// Proceedings of the Sixth International Symposium on Spatial Data Handling. Columbia: University of South Carolina, 1994.
[54] SHAPIRO J, WAXMAN J, NIR D. Level graphs and approximate shortest path algorithms[J]. Networks, 1992, 22: 691-717.
[55] HUANG Y W, JING N, RUNDENSTEINER E A. Effective graph clustering for path queries in digital map database[C]//Proceedings CIKM Rockville. New York: ACM Press, 1996: 215-222.
[56] HABBAL M B, KOUTSOPOULOS H N, LERMAN S R. A decomposition algorithm for the all-pairs shortest path problem on massively parallel computer architectures[J]. Transportation Science, 1994, 28(4): 292-308.
[57] IDWAN S, ETAIWI W. Computing breadth first search in large graph using hMetis partitioning[J]. European Journal of Scientific Research, 2009, 29(2): 215-221.
[58] KARYPIS G, KUMAR V. A fast and high quality multilevel scheme for partitioning irregular graphs[J]. SIAM J Sci Comput, 1998, 20(1): 359-392.
[59] NEWMAN M E J. Networks: an introduction[M]. Oxford: Oxford University Press, 2010.
[60] 汪小帆, 刘亚冰. 复杂网络中的社团结构算法综述[J]. 电子科技大学学报, 2009, 38(5): 537-543.
WANG Xiao-fan, LIU Ya-bing. Overview of algorithms for detecting community structure in complex networks[J]. Journal of University of Electronic Science and Technology of China, 2009, 38(5): 537-543.

编 辑 蒋 晓