文章编号:1001 -4098(2006)09-0001-05

不确定情形下通信网络最短路径关键点问题*

闫化海[1], 徐寅峰[1,2], 刘 明[1]

(1. 西安交通大学 管理学院, 西安 陕西 710049;
2. 机械制造系统工程国家重点实验室, 西安 陕西 710049)

摘 要: 在通信网络中, 因突发事件造成通信路由节点毁坏或者中断的现象时有发生, 传输的数据包不得不从中断处沿着最短的替代路径行进到数据包的接收节点, 在这种情形下, 哪个路由节点中断使得数据包实际行进的总路程最长呢?从通信网络管理的角度来看这是一个非常重要的问题。对该问题, 以前的文献都是从确定情形(事先具有节点中断的完全信息)下进行研究的, 本文从不确定情形(只有数据包行进到中断节点的邻接点时才获得该节点中断的信息)的角度重新考虑这个问题。本文首先定义了不确定情形下的最短路径关键点概念, 给出了计算不确定情形下最短路径关键点的算法及其时间复杂性分析。结合实际通信网络的算例分析, 比较了确定情形下最短路径关键点和不确定情形下最短路径关键点问题, 指出了不确定情形下最短路径关键点问题更具有实际意义。

关键词: 关键点; 不确定情形; 最短路径; 算法
中图分类号: C931; O221 **文献标识码**: A

1 引言

在通信网络中, 因突发事件造成通信路由节点毁坏或者中断的现象时有发生, 传输的数据包不得不从中断处沿着最短的替代路径行进到数据包的接收节点, 致使数据包传输的距离或者时间增大, 在这种情形下, 哪个路由节点中断使得数据包实际行进的总路程或总时间最长呢?在某些特定情形下, 出于对数据包信息保密或信息安全的考虑, 会对数据包在通信网络中的传输时间有一个严格的限制, 例如ATM 机系统, 如果传输时间超出这个限制, 该数据包就会被系统自动拒收或者被视为无效(垃圾)数据包。在这种情形下, 从通信网络管理的角度来看, 研究不确定情形下的最短路径关键点问题具有十分重要的现实意义。

最短路径关键点问题最初由 Corley 和 Sha 提出, 更准确地讲, 他们考虑了更一般的情形, 即在一个至少(k+1)-Nodes 连通的网络 $G(V, E)$ 中, 同时删除 k 个节点, 使得起点 s 至终点 t 的最短路径长度增至最大, 则称这 k 个点为最短路径的 k 关键点, 对于 k=1 的情形, 给出了时间复杂性为 $O(mn + n^2\log n)$ 的算法, n 和 m 表示图中节点和边的个数[1]。此后对于 k=1 的情形, E. Nardelli、G. Proietti 和 P. Widmyer 给出了时间复杂性为 $O(m + n\log n)$ 的算法[2]; A. Bar-Noy 等证明了对于任意的 k, 最短路径 k-关键边问题是NP-Hard 问题[3]。国内研究方面, 李引珍和郭耀煌给出了 $O(n^3)$ 时间复杂性的算法[4]。

在类似的情形下, 节点是稳定的, 而边有可能中断, 以往关于关键边的相关文献主要包括最短路径关键边和最长绕行路关键边两方面。最短路径关键边问题首先由 Corley 和 Sha 提出, 即在给定一个至少2-Edges 连通的网络 $G(V, E)$ 中, 则至少存在一条边 e^*, 当从 $G(V, E)$ 中删除 e^* 时, 使得起点 s 至终点 t 的最短路径长度增至最大, 则称边 e^* 为该最短路径的关键边[1]; 李引珍和郭耀煌给出了计算最短路关键边的子树连通算法, 该算法的时间复杂性为 $O(n^2)$[5]; Malik 等利用斐波纳契堆(Fibonacci Heaps)给出了时间复杂性为 $O(m + n\log n)$ 的算法[6]; 此后 E. Nardelli、G. Proietti 和 P. Widmyer 又将算法的时间复杂性改进到 $O(m \cdot \alpha(m, n))$, 其中 α 为阿克曼(Ackermann)函数的反函数[7]。

最长绕行路关键边问题是由 E. Nardelli、G. Proietti 和 P. Widmyer (1998)提出的, 即: 在一个至少2-Edges 连通的网络 $G(V, E)$ 中, 对最短路径上的任意一条边为 e=

* 收稿日期: 2006-03-21
基金项目: 国家杰出青年科学基金资助项目(70525004); 国家自然科学基金资助项目(70471035)
作者简介: 闫化海(1974-), 男, 山东人, 西安交通大学管理学院, 研究方向: 交通管理。

(u, v), 其中 u 点更靠近起点, 当从 G 中删除边 $e=(u, v)$ 时, 在 u 点进行绕路到达终点, 使得绕行路的路长满足 $d_{G-e^*}(u^*, t)-d_G(u^*, t)\geq d_{G-e}(u, t)-d_G(u, t)$, 则称边 $e^*=(u^*, v^*)$ 为最长绕行路的关键边。E. Nardelli 等给出时间复杂性为 $O(m+n\log n)$ 的算法[8]; 此后他们又将算法的时间复杂性改进到 $O(m\cdot\alpha(m, n))$ [7]。

现有关于最短路径关键点问题的研究都是基于确定情形下的, 即在出发前就已获得节点中断的完全信息, 然而在实际中, 由于突发事件的不可预见性, 只有到达中断节点的邻接点处才获得该节点中断的信息, 事先并不知道该节点已经中断。针对以上研究不足, 本文提出了不确定情形下的最短路径关键点问题, 从具有不确定性的角度出发重新考虑最短路径关键点问题。

本文第2节为问题的描述和不确定情形下最短路关键点的定义; 第3节给出具体的算法及其时间复杂性分析; 第4节结合实际通信网络给出算例, 比较分析确定情形和不确定情形下的最短路径关键点问题的差异; 第5节指出进一步研究的方向。

2 问题描述及定义

2.1 问题描述

将现实中的通信网络抽象成以路由器为节点, 两节点间的连结线为边的网络, 边长为实际连结线的长度。我们将通信网络记为 $G(V, E)$, 其中 $V=\{s, v_1, v_2, \cdots, v_{n-2}, t\}$ 为节点集, s 表示数据包的发出节点, t 表示数据包的接收节点, $E=\{e_1, e_2, \cdots, e_m\}$ 为边集, 其中 m 为 G 中边的个数, n 为节点的个数。给定起点 s 和终点 t, $P_G(s, t)$ 表示从起点 s 至终点 t 的最短路径, 最短路径的长度记为 $d_G(s, t)$, $\omega(x, y)$ 表示边 (x, y) 的长度,$\{s, v_1, v_2, \cdots, v_k, t\}$ 表示最短路径 $P_G(s, t)$ 上的节点集, 其中 k 表示除起点 s 和终点 t 之外的最短路径上节点的个数, 且节点 v_{i-1} 比 v_i 节点更靠近起点。

为了研究方便, 本文的假设条件为:

①数据包的传输总是沿着出发点 s 至接收点 t 的最短路径上行进, 在本文中假设两点间的最短路径是唯一的;

②节点中断仅发生在 s 至 t 的最短路径上, 且节点中断的信息并不扩散到网络的其他节点上, 只有数据包行进到最短路径上该节点的邻接节点时才获得该节点中断的详细信息;

③在数据包传输过程中, 只发生一次节点中断的事件, 节点中断时与其相连的边同时视为中断。

2.2 不确定情形下的最短路径关键点定义

以往关于最短路径关键点问题的文献研究都是基于确定情形的, 即在出发前就已获得节点中断的完全信息。而在实际情形中, 数据包只有到达中断处才获得节点中断的信息, 因此在本文我们考虑节点中断位置不确定的情形, 定义不确定情形下的最短路径关键点概念。

定义: 在一个至少2-Node 连通的网络 $G(V, E)$ 中,$\{s, v_1, v_2, \cdots, v_k, t\}$ 是最短路径 $P_G(s, t)$ 上的节点集, 从 G 中依次删除节点 $v_i(i=1,2, \cdots, k)$, 使得数据包从节点 v_{i-1} 开始沿着 $\{G-v_i\}$ 中点 v_{i-1} 至 t 的最短路径绕行到终点 t, 使得数据包实际行进的总路程满足 $d_{G-v_i^*}(s, t)=\max\{d_{G-v_i}(s, t)\}$, 则称满足最大路程的节点 v^* 为不确定情形下的最短路径关键点。其中 $d_{G-v_i^*}(s, t)=d_G(s, v_{i-1}^*)+d_{G-v_i^*}(v_{i-1}^*, t)$, $d_{G-v_i}(s, t)=d_G(s, v_{i-1})+d_{G-v_i}(v_{i-1}, t)$, $d_G(s, v_{i-1})$ 为获得节点 v_i 中断信息时已经走过的路程, $d_{G-v_i}(v_{i-1}, t)$ 为在 $G(V-v_i, E)$ 中从 v_{i-1} 点到终点 t 绕行路的路长。

3 不确定情形下最短路径关键点的计算

下面我们给出计算不确定情形下最短路径关键点问题的具体算法, 根据子树连通的思想, 只要我们能分别计算得到以终点 t 和中断节点邻接点 v_{i-1} 为根的最短路树, 使其子树连通, 就可以利用终点 t 至各点的最短距离进行计算。

首先调用 Bellman-Ford 算法计算以终点 t 为根的最短路树 $S_G(t)$, 假设最短路径是唯一的。节点 $v_i(i=1,2, \cdots, k)$ 是最短路 $P_G(s, t)$ 上的节点(起点 s 和终点 t 除外), 且节点 v_{i-1} 比节点 v_i 更靠近起点 s 依次删除节点 v_i 及其相连的边, 最短路树 $S_G(t)$ 中的点被分置在3个不同点集 $M_t(v_i)$、$N_t(v_i)$ 和 $Q_t(v_i)$ 中, 包含终点 t 的点集为 $M_t(v_i)$, 包含节点 v_{i-1} 和 s 的点集为 $N_t(v_i)$, 其余节点集合统称为 $Q_t(v_i)=\{V-M_t(v_i)-N_t(v_i)-v_i\}$, 如图1所示。$M_t(v_i)$ 是在 $S_G(t)$ 中从终点可直接到达该点而不通过节点 v_i 的节点集合, 而 $N_t(v_i)$ 是 $S_G(t)$ 的一个子树, 也可以被认为是以 v_{i-1} 为根的最短路子树。为了构造节点 v_{i-1} 至节点 t 的最短路, 将 $M_t(v_i)$ 和 $Q_t(v_i)$ 点集合并, 视为一个集合 $M_t(v_i)\cup Q_t(v_i)$, 至此就可以根据子树连通的思想, 利用点集 $N_t(v_i)$ 和 $M_t(v_i)\cup Q_t(v_i)$ 来构造计算不确定情形下最短路径关键点的算法。但是, 由于节点 y 可能属于 $M_t(v_i)$ 或 $Q_t(v_i)$ 两个不同的点集, 无法得到 $d_{G-v_i}(y, t)$, 为了计算 $d_{G-v_i}(y, t)$, 假设存在一个节点 u 在 t 至 y 的最短路径上, 点 u 与点 y 相邻且更靠近节点 t. 下面根据节点 u 和 y 的不同位置, 分别讨论所有三种可能情形下的 $d_{G-v_i}(y, t)$ 计算。

情形1: 节点 u 和 y 在 $M_t(v_i)$ 中, 即 $u, y\in M_t(v_i)$, 此时 $d_{G-v_i}(t, y)=d_G(t, y)$。

情形2: 两个节点 u 和 y 中一个在 $M_t(v_i)$ 中, 一个在 $Q_t(v_i)$ 中, 即 $u\in M_t(v_i)$, $y\in Q_t(v_i)$, 此时可得 $d_{G-v_i}(t, y)=\min\limits_{u\in M_t(v_i), y\in Q_t(v_i)}\{d_{G-v_i}(t, u)+w(u, y)\}$, 由于 $u\in M_t(v_i)$, 所以有 $d_{G-v_i}(t, u)=d_G(t, u)$, 可得 $d_{G-v_i}(t, y)=\min\limits_{u\in M_t(v_i), y\in Q_t(v_i)}\{d_G(t, u)+w(u, y)\}$。

情形3: 两节点 u 和 y 在 $Q_t(v_i)$ 中, 即 $u, y \in Q_t(v_i)$, 此时边 (u, y) 为 $Q_t(v_i)$ 中的一条边。此时在情形2已经计算得到的结果基础上, 在点集 $Q_t(v_i)$ 中, 利用Dijkstra 算法计算 t 至 $Q_t(v_i)$ 中各点最短路径的距离 $d_{G-v_i}(t, y)$。

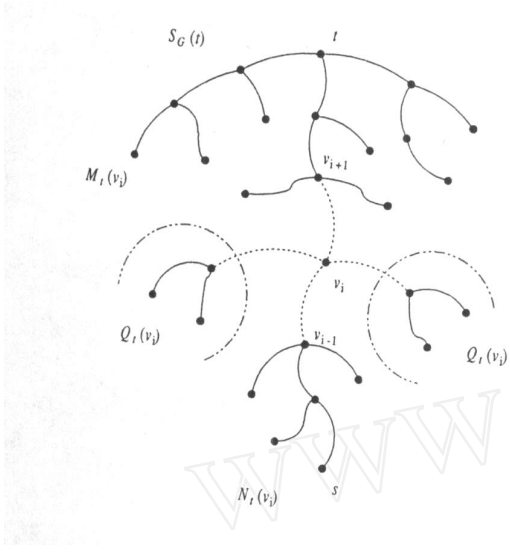

图1　节点 v_i 及其邻接边删除后, 图中节点被分置在 $M_t(v_i)$、$N_t(v_i)$ 和 $Q_t(v_i)$ 三个点集中

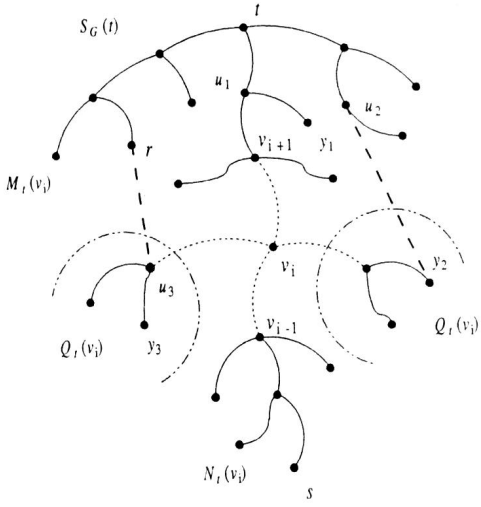

图2　节点 t 到节点 y 的三种不同情形

通过上述三种情形的分析, 可以得到 $d_{G-v_i}(t, y)$。至此, 可以得到以 v_{i-1} 为根的最短路子树和以 t 为根的最短路树, 根据子树连通的思想, 对于节点 v_{i-1} 至终点 t 的最短路径的距离 $d_{G-v_i}(v_{i-1}, t)$, 如图3所示, 有下列等式成立

$$d_{G-v_i}(v_{i-1}, t) = \min_{x \in N_t(v_i), y \in M_t(v_i)} \{d_{G-v_i}(v_{i-1}, x) + w(x, y) + d_{G-v_i}(y, t)\}$$

实际上, 因为 $x \in N_t(v_i)$, 所以 $d_{G-v_i}(v_{i-1}, x) = d_G(v_{i-1}, x) = d_G(t, x) - d_G(t, v_{i-1})$。所以有

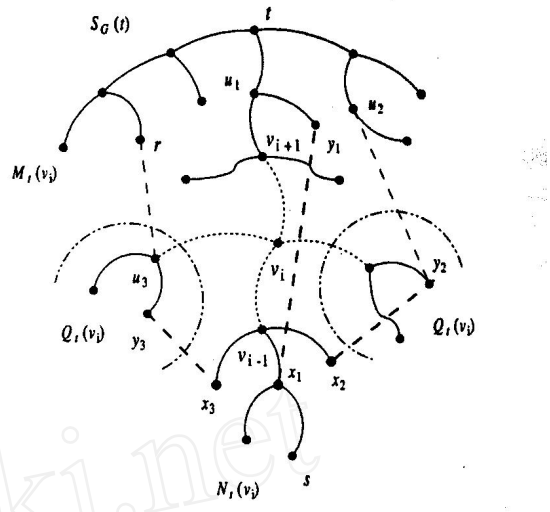

图3　割边 $w(x, y)$ 的三种不同情形

$$\begin{aligned} & d_{G-v_i}(v_{i-1}, t) \\ = & \min_{x \in N_t(v_i), y \in M_t(v_i)} \{d_{G-v_i}(v_{i-1}, x) + w(x, y) + d_{G-v_i}(y, t)\} \\ = & d_{G-v_i}(v_{i-1}, t) \\ = & \min_{x \in N_t(v_i), y \in M_t(v_i)} \{d_G(t, x) - d_G(t, v_{i-1}) + w(x, y) + d_{G-v_i}(y, t)\} \end{aligned}$$

再加上绕行前已经走过的路程 $d_G(s, v_{i-1})$, 可以得到实际行进的总路程

$$\begin{aligned} & d_{G-v_i}(s, t) \\ = & d_G(s, v_{i-1}) + d_{G-v_i}(v_{i-1}, t) \\ = & d_G(s, t) - d_G(t, v_{i-1}) \\ & + \min_{x \in N_t(v_i), y \in M_t(v_i)} \{d_G(t, x) - d_G(t, v_{i-1}) + w(x, y) + d_{G-v_i}(y, t)\} \end{aligned}$$

3.1　算法

根据上述算法思想, 我们给出下面的具体算法:

步骤1: 利用 Bellman-Ford 算法计算 $G(V, E)$ 中 t 至所有其他顶点的最短路径, 此时便得到了 $S_G(t)$ 和 $P_G(s, t)$, 并记录 $P_G(s, t)$ 上节点的个数 k (节点 s, t 除外);

步骤2: 令 i=1;

步骤3: 从 $P_G(s, t)$ 删除节点 v_i, 生成 $M_t(v_i)$、$N_t(v_i)$ 和 $Q_t(v_i)$;

步骤4: 计算 $d_{G-v_i}(t, y)$;

步骤5: 计算 $d_{G-v_i}(v_{i-1}, t) = \min_{x \in N_t(v_i), y \in M_t(v_i)} \{d_G(t, x) - d_G(t, v_{i-1}) + w(x, y) + d_{G-v_i}(y, t)\}$;

步骤6: 计算 $d_{G-v_i}(s, t) = d_G(s, v_{i-1}) + d_{G-v_i}(v_{i-1}, t)$;

步骤7: $i = i+1$, 如果 $i \le k$, 则回到步骤3, 否则到步骤8;

步骤8: 求 $d_{G-v_i}(s, t)$ 的最大值, 所对应的中断节点即为不确定情形下最短路径关键点。

3.2　算法复杂性分析

对顶点数为 n、边的个数为 m 的通信网络而言, 上述

算法的时间复杂性分析如下:

①步骤1中调用Bellman-Ford 算法, 计算的时间复杂性为$O(mn)$;

②步骤3的时间复杂性为$O(1)$;

③步骤4中, 第一和第二种情形中连接计算的时间复杂性为$O(m)$, 第三种情形调用Dijkstra 算法计算的时间复杂性为$O(n^2)$;

④步骤5和步骤6为连接计算, 各步骤计算的时间复杂性为$O(m)$;

⑤算法中第2～7步为循环计算, 循环的次数为k。因为$k \leq n-2$, 所以2-7 步的时间复杂性为$O(n^3)$;

⑥步骤8中计算的时间复杂性为$O(n)$。

所以本文所给计算不确定情形下最短路径关键点的算法时间复杂性为$O(n^3)$。

4 实例分析

图4为某银行部分自动取款机(ATM)的布点网络图, t表示该银行的ATM 处理中心所在位置, 其余节点为ATM 机的所在位置。现在有位顾客在节点s的取款机上取款, 取款机发出取款的指令通过数据包的形式发送到处理中心进行确认, 然后将确认的客户信息反馈到客户端, 包含顾客信息的数据包从点s沿着最短路径传送到处理中心。本文假设在数据包的传输过程中发生仅一次节点中断事件, 节点中断迫使数据包绕行到处理中心, 这将会延长数据包传输的时间, 致使顾客等待的时间变长, 从而降低该银行客户服务的满意度。银行从提高服务质量的角度出发, 为了保证因突发性的节点中断迫使银行服务延迟的时间在一个可接受的范围之内, 就需要确定哪个节点中断会造成最长的延迟时间, 从而在实际的通信网络管理中采取有效措施避免这种最坏情形的发生。

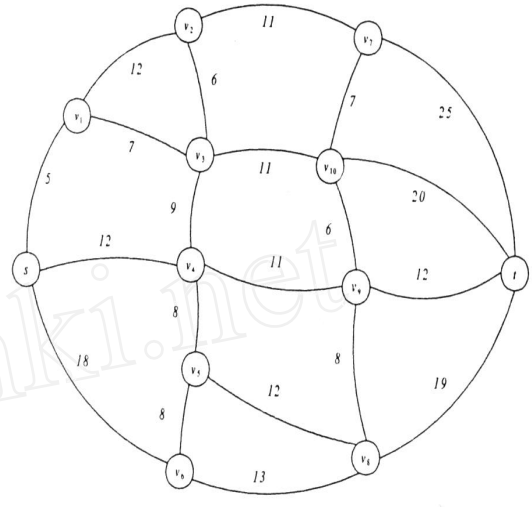

图4 某银行ATM 机布点网络图

对于该问题, 其实质是要求在不确定情形下, 最短路径上的哪个节点中断时致使数据包传输的总路程最长?如上图所示, 起点s与终点t之间的最短路径为$s \rightarrow v_4 \rightarrow v_9 \rightarrow t$, 对应的最短路径的长度为35。为了比较分析确定和不确定情形下的最短路径关键点的异同, 下面分别计算这两个问题, 具体计算结果如表1所示。

表1 计算结果

	确定情形下最短路径关键点		不确定情形下最短路径关键点			
	$d_{G-v_i}(s,t)$	实际行走线路	$d_{G-(s,v_{i-1})}$	$d_{G-v_i}(v_{i-1},t)$	$d_{G-v_i}(s,t)$	实际行走线路
节点v_4中断	41	$s \rightarrow v_1 \rightarrow v_3 \rightarrow v_{10} \rightarrow v_9 \rightarrow t$	0	41	41	$s \rightarrow v_1 \rightarrow v_3 \rightarrow v_{10} \rightarrow v_9 \rightarrow t$
节点v_9中断	43	$s \rightarrow v_1 \rightarrow v_3 \rightarrow v_{10} \rightarrow t$	12	39	51	$s \rightarrow v_4 \rightarrow v_5 \rightarrow v_8 \rightarrow t$
关键点	节点		节点			

比较上述结果可以发现:

①当节点v_4中断时, 确定情形下的总路程等于不确定情形下的总路程, 两者的距离都等于41, 行走的线路同为$s \rightarrow v_1 \rightarrow v_3 \rightarrow v_{10} \rightarrow v_9 \rightarrow t$, 这是由于$v_4$中断时, 即$i=1$的情形, 此时节点$v_{i-1}=s$, 相当于$d_G(s, v_{i-1})=0$的情形, 所以二者距离相等。

②当节点v_9中断时, 确定情形下的总路程等于43, 线路为$s \rightarrow v_1 \rightarrow v_3 \rightarrow v_{10} \rightarrow t$; 而不确定情形下的总路程等于51, 实际行程为$s \rightarrow v_4 \rightarrow v_5 \rightarrow v_8 \rightarrow t$, 比在确定情形时多走了8个单位的距离。

③两种情形下的关键点同为节点v_9, 这说明节点v_9对该网络是十分重要的。此外不确定情形下实际行走的距离比在确定情形时多了8个单位的距离, 这是由于在确定情形时具有节点中断的完全信息, 而在不确定情形下时, 只有行进到中断节点的邻接点时才获得节点中断的信息, 因此不确定情形行走的距离要大于确定情形。

④此外对于不同的网络结构, 两种情形下的最短路径关键点不一定相同。在本文给出的实例中, 两种不同情形下的关键点是相同的, 但是如果网络结构发生改变, 两种情形下的关键点就有可能不同, 这与网络的结构特性紧密相关。

在现实情形下, 通信网络的节点中断是突发性的, 是

不可预见的, 也就是说不确定情形下的最短路径关键点问题更符合现实的情形。通过上述结果的比较分析, 斡们可以看出, 不确定情形下的最短路径关键点问题可以更好地解释实际情形。

5　结论

在ATM 机的通信网络中, 经常由于各种突发事件导致路由节点被破坏的现象时有发生, 为了避免突发性事件给通信系统造成的最坏情形发生, 非常有必要研究不确定情形下最短路径关键点问题, 因此本文研究具有十分重要的现实意义。本文在定义不确定最短路径关键点概念的基础上, 给出了计算不确定情形下最短路径关键点的改进算法, 该算法的时间复杂性为$O((n^2)$, 其中n为节点数。通过实际通信网络的算例分析, 说明不确定情形下最短路径关键点问题可以更好地解释实际情形。此外, 不确定情形下的最短路径k关键点问题可以作为进一步研究方向。

参考文献:

[1] Corley H W, Sha D Y. Most vital links and nodes in weighted networks [J]. Operation Research Letters, 1982, (1): 157～161.

[2] Nardelli E, Proietti G, Widmyer P. Finding the most vital node of a shortest path [J]. Theoretical Computer Science, 2003,296:167 ～177.

[3] Bar-Noy A, Khuller S, Schieber B. The complexity of finding most vital arcs and nodes. Technical Report CS-TR-3539[R]. Institute for Advanced Computer Studies, University of Maryland, MD, 1995.

[4] 李引珍, 郭耀煌　运输网络最短路径关键点问题研究[J]. 铁道学报, 2004,26 (6):106～111.

[5] 李引珍, 郭耀煌　交通运输网络最短路径关键边问题研究[J]. 中国管理科学, 2004,12 (4):69～73.

[6] Malik K, Mittal A K, Gupta S K. The k most vital arcs in the shortest path problem [J]. Operation Research Letters, 1989, (8):223～227.

[7] Nardelli E, Proietti G, Widmyer P. A faster computation of the most vital edge of a shortest path between two nodes [J]. Information Processing Letters, 2001,79 (2):81～85.

[8] Nardelli E, Proietti G, Widmyer P. Finding the detour critical edge of a shortest path between nodes [J]. Information Processing Letters, 1998,67 (1):51～54.

[9] Ball M O, Golden B L, Vohra R V. Finding the most vital arcs in a network [J]. Operations Research Letters, 1989, (8):73～76.

The Most Vital Node of the Shortest Path under Uncertainty in Communication Networks

YAN Hua-hai[1], XU Yin-feng[1,2], LIU Ming[1]

(1. School of Management, Xi'an Jiaotong University, Xi'an 710049, China;
2. The State Key Lab for Manufacturing Systems Engineering, Xi'an 710049, China)

Abstract: In communication networks, data packages often travel longer than the shortest path from source node to destination node due to sudden node failure caused by unexpected events. From the network management point of view, it is important to find the node whose removal results in the longest travel distance between source node and destination node in a network. The most vital node (MVN) problem of the shortest path in term of certainty were extensively studied in the past. This paper aims at the most vital node of the shortest path problem under uncertainty (MVN-U). Firstly, this paper states the concept of the most vital node of the shortest path under uncertainty. Secondly, it presents an algorithm of computing the MVN-U and analyses its time complexity, and then a numerical result of ATM networks is given. In the end, by comparing the result of MVN-U and MVN problem, we conclude that the MVN-U problem has more practical significance.

Key words: Most Vital Node; Uncertainty; Shortest Path; Algorithm