软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(5):1536−1548 [doi: 10.13328/j.cnki.jos.005675]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于最小路径交叉度的域内路由保护方案*

耿海军[1], 施新刚[2], 王之梁[2], 尹 霞[3], 胡治国[4]

[1](山西大学 软件学院,山西 太原 030006)

[2](清华大学 网络科学与网络空间研究院,北京 100084)

[3](清华大学 计算机科学与技术系,北京 100084)

[4](山西大学 计算机科学与技术系,山西 太原 030006)

通讯作者: 施新刚, E-mail: shixg@cernet.edu.cn

摘 要: 已有的路由保护方案面临下面两个问题:(1) 默认路径和备份路径包含的公共边数量较高,如 ECMP 和 LFA 等;(2) 为了计算两条包含公共边数量较少的路径,限制默认路径不能使用最短路径,如红绿树方案等.针对上述两个问题,首先将计算默认路径和备份路径描述为一个整数规划问题,然后提出采用启发式方法求解该问题,接着介绍了转发算法,最后通过仿真实验和真实实验对算法进行了测试.实验结果表明,该算法不仅具有较低的计算复杂度,而且可以降低默认路径和最短路径包含的公共边的数量,提升网络可用性.

关键词: 路由保护;不相交路径;默认路径;备份路径;网络故障

中图法分类号: TP393

中文引用格式: 耿海军,施新刚,王之梁,尹霞,胡治国.基于最小路径交叉度的域内路由保护方案.软件学报,2020,31(5):1536−1548. http://www.jos.org.cn/1000-9825/5675.htm

英文引用格式: Geng HJ, Shi XG, Wang ZL, Yin X, Hu ZG. Intra-domain routing protection scheme based on minimum intersection paths. Ruan Jian Xue Bao/Journal of Software, 2020,31(5):1536−1548 (in Chinese). http://www.jos.org.cn/1000-9825/5675.htm

Intra-domain Routing Protection Scheme Based on Minimum Intersection Paths

GENG Hai-Jun[1], SHI Xin-Gang[2], WANG Zhi-Liang[2], YIN Xia[3], HU Zhi-Guo[4]

[1](School of Software Engineering, Shanxi University, Taiyuan 030006, China)

[2](Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China)

[3](Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

[4](Department of Computer Science and Technology, Shanxi University, Taiyuan 030006, China)

Abstract: The existing routing protection schemes are facing the following two problems: (1) the default path and the backup path contain a large number of common edges, such as ECMP and LFA; (2) in order to calculate two paths which have a small number of common edges, the shortest path cannot be used in the network, such as red-green tree method. To solve the above two problems, this study first describes the problem of computing backup paths and default paths as an integer programming model, and then a heuristic method is proposed to solve the problem. Next, the forwarding algorithm is introduced. Finally, the proposed algorithm is tested by simulation experiment and real experiment. Experiments show that the proposed algorithm not only has lower computational complexity,

* 基金项目: 国家自然科学基金(61702315, 61872226); 山西省高等学校科技创新项目(201802013); 国家重点研发计划(2018YFB1800401); 山西省自然科学基金(201701D121052); 山西省重点研发计划(国际科技合作)(201903D421003)

Foundation item: National Natural Science Foundation of China (61702315, 61872226); Scientific and Technological Innovation Programs of Higher Education Institutions in Shanxi Province (201802013); National Key Research and Development Program of China (2018YFB1800401); Natural Science Foundation of Shanxi Province (201701D121052); Key Research and Development Program of Shanxi Province (International Science and Technology Cooperation Project) (201903D421003)

收稿时间: 2017-10-30; 修改时间: 2018-08-09; 采用时间: 2018-10-16

but also reduces the number of common edges contained in the default paths and the shortest paths, and greatly improve the network availability.

Key words: routing protection; disjoint path; default path; backup path; network failure

随着互联网的普及和快速发展,部署在其上的应用程序发生了很大的变化.之前以发送邮件和传输文件等非实时应用[1]占部署应用的主流,而现在,随着实时应用需求越来越大,在线视频和股票交易等实时应用[2−4]占互联网部署应用的主流.这就需要互联网服务提供商提供无中断的网络服务[5,6]和快速重路由机制[7]来保证网络的可用性.OSPF[8,9]协议通过被动恢复方案来解决网络中出现的各种故障,但是整个收敛过程需要几秒甚至几十秒的时间来完成.而网络中单故障(节点、链路)[10−12]频繁发生,如果在 50ms 之内无法完成,就无法满足现实时应用对网络快速收敛的需求.所以业界普遍利用路由保护方案[13−17]来解决网络中的故障问题.

总体来说,路由保护方案可被分为两种方式:逐跳转发和非逐跳转发.逐跳转发主要有 ECMP 和 LFA 等,非逐跳转发主要有 Not-Via、多协议标签交换[18]和隧道技术[19]等.等价多路径路由(equal cost multiple paths,简称 ECMP)[20]是路由保护中最简单的一种方案.虽然 ECMP 实现简单并且容易部署,但是 ECMP 的故障覆盖率较低.IETF 在 RFC5286 标准文档中公布了快速重路由的基本框架(IP fast re-route,简称 IPFRR)[21].基于 IPFRR 提出的基本框架,文献[22]中提出利用路由偏转方案来实现报文的无环路转发,但是路由偏转方案的实现方式比较复杂.LFA 是基于 IPFRR 框架提出的一种解决算法,LFA 以其简单、容易部署而受业界关注,并且已经在部分厂商的路由器上实现并且投入使用.在网络状态发生改变时,为了迅速找到适合的备用路径,文献[23]中提出了多配置路由方案(multiple routing configurations,简称 MRC).MRC 通过给每个路由器配置多个拓扑图,利用配置的拓扑图计算备份路径,但是该方案需要的配置较多,不容易实际部署.文献[24]中提出了分组携带故障信息(failure carrying packet,简称 FCP).该方案将故障信息存储在报文的首部,当报文被转发到某个路由器时,路由器立刻检查报文首部的故障信息,然后利用故障信息生成新的网络拓扑,最后在新的网络拓扑中构造最短路径.但是 FCP 改变了报文的头部,对报文的修改较大,无法直接部署在互联网中.文献[25]介绍了一种利用 Not-Via 地址保护网络中链路和节点的路由保护方案,但是 Not-Via 改变了互联网的转发方式,并且计算复杂度较高,所以很难实际部署.但是,以上所有的方案在备份路径和最短路径中边的交叉度方面考虑不足.针对此问题,文献[26–29]中提出计算不相交的路径来提高路由可用性.然而,由于这些方案中最短路径受到了限制,在兼容性方面目前还无法融入到域内路由协议中.

所以,本文将研究重点放在如何为网络中任意的节点对之间计算备份路径,并使备份路径和最短路径的交叉度之和降到最低.本文提出的路由保护方案是在路由系统进行路由学习和路由表生成时计算备份路径,当网络拓扑发生变化时,在路由收敛的过程中不需要再重新计算备份路径,而是利用备份路径转发受拓扑变化影响的报文.当路由重新收敛完成后,在新的拓扑中重新计算备份路径.本文的贡献点主要有 3 个方面的内容:(1) 我们首先将计算默认路径和备份路径的问题描述成为一个整数规划问题,然后提出两种启发式方案(DeleteLink 和 B-DeleteLink)来求解该问题;(2) 提出了一种与互联网兼容的转发算法;(3) 实验结果表明,B-DeleteLink 不仅具有较低的计算开销,而且计算出的备份路径和最短路径具有较少的公共边,极大地提升了路由可用性.

1 网络模型和问题描述

本节首先定义一个网络模型,然后在该模型的基础上描述需要解决的问题.为了便于读者阅读,我们将文中使用的部分符号总结在表 1 中.

Table 1 Notations

表 1 符号

符号	$G=(V,E)$	$w(i,j)$	$P(o,d,G)$	$D(o,d,G)$	$P(o,d,G')$	$D(o,d,G')$	$K(o,d,e)$	$L(o,d)$	$R(G,G')$
含义	无向连通图	边$(i,j)\in E$对应的代价	(o,d)的最短路径	路径$P(o,d,G)$的代价	(o,d)的备份路径	路径$P(o,d,G')$的代价	e是否同时属于路径$P(o,d,G)$和$P(o,d,G')$	(o,d)的路径交叉度	路径交叉度

1.1 网络模型

我们把网络描述成无向连通图 $G=(V,E)$,其中,V 在网络中表示路由器(节点)集合,E 在网络中表示边(链路)的集合.对于网络中任意一条边,表示为$\forall e=(i,j)\in E$,而对应边的代价用 $w(e)$或者 $w(i,j)$来表示,其中,网络中所有的边的代价是对称的,用 $w(i,j)=w(j,i)$来表示.在给定的网络拓扑 $G=(V,E)$中,$P(o,d,G)$代表节点对(o,d)的最短路径,$D(o,d,G)$代表 $P(o,d,G)$对应的代价,$P(o,d,G')$代表节点对(o,d)的备份路径,用 $D(o,d,G')$代表 $P(o,d,G')$的代价,其中,G'是在 G 的基础上计算出的扩展拓扑结构.关于如何从 G 得到 G',以及为什么在 G'中计算出来的节点对之间的最短路径就是节点对(o,d)之间的备份路径这个问题,我们会在后续章节中进行阐述.下面将说明如何表示节点对(o,d)之间的最短路径和备份路径的交叉度.如果 $e\in P(o,d,G)$和 $e\in P(o,d,G')$同时成立,则 $K(o,d,e)=1$;否则,$K(o,d,e)=0$.即

$$K(o,d,e)=\begin{cases}1, & e\in P(o,d,G)\text{并且}e\in P(o,d,G')\\ 0, & \text{否则}\end{cases} \tag{1}$$

定义 1(节点对交叉度). 我们将节点 o 和节点 d 之间的交叉度定义为二者之间的最短路径和备份路径中同时含有的相同边的数量,用公式(2)表示.

$$L(o,d)=\sum_{e\in E}K(o,d,e) \tag{2}$$

定义 2(路径交叉度). 路径交叉度可以用公式(3)表示.

$$R(G,G')=\sum_{o,d\in V}L(o,d) \tag{3}$$

下面通过一个简单的例子来说明上述的定义.图 1 中左边的图形表示网络拓扑 G,有 5 个节点和 6 条边组成;右边的图形表示其对应的扩展网络拓扑 G'.二者的区别是删除了链路(c,b).在 $G=(V,E)$中,节点的集合可以表示为 $V=\{o,a,b,c,d\}$,对于节点对(o,d),它们之间的最短路径可以表示为 $P(o,d,G)=\{o,a,c,d\}$,备份路径可以表示为 $P(o,d,G')=\{o,b,d\}$,因此 $L(o,d)=0$,即节点对(o,d)之间的最短路径和备份路径包含的公共边的数量为 0,即二者的交叉度为 0.从该例子可以看出,如果节点对的交叉度为 0,二者之间的最短路径和备份路径没有公共边.假设该节点对之间的某条链路出现了故障,备份路径中一定不包含该条链路,则该节点对之间转发的报文不会受该故障的影响,极大地提高了网络的可靠性.

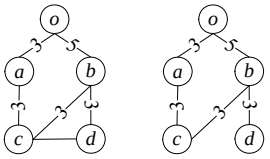

Fig.1 Network topology G and its extended network topology G'

图 1 网络拓扑 G 和其对应的扩展网络拓扑 G'

1.2 问题描述

本文解决的关键科学问题可以概括为:已知网络结构 $G=(V,E)$,如何构造 $G'=(V,E')$,最小化 $R(G,G')$的数值.下面我们首先描述如何计算 $R(G,G')$,其具体过程包括下面几个步骤.

(1) 根据 G 构造最短路径树,从而计算出所有节点对之间的最短路径.

(2) 根据 G'构造最短路径树,从而计算出所有节点对之间的备份路径.

(3) 根据上述两个步骤中计算出的最短路径和备份路径计算 $R(G,G')$.

从上述步骤可知,本文需要解决的关键问题为如何在初始网络拓扑 G 的基础上,计算出其对应的扩展网络拓扑 G',从而使得 $R(G,G')$最小.为了降低 $R(G,G')$的数值,一种较为直观的方法是在原来拓扑 G 的基础上,通过删除某些链路得到其对应的扩展网络拓扑 $G'=(V,E')$,即初始网络拓扑和扩展网络拓扑的节点的集合相同,但是边的集合不同.然后利用该扩展网络拓扑计算出备份路径,因为计算备份路径的时候将不再使用删除掉的这些链路,因此网络的交叉度会降低.本文通过删除网络中的链路获得初始网络拓扑 $G=(V,E)$对应的扩展网络拓扑

$G'=(V,E')$,在实际中并不是真正地删除这些链路,而仅仅是在计算备份路径时不再使用这些链路,初始网络拓扑并不会发生变化.下面通过定理 1 和定理 2 来说明上述方法的可行性.

定理 1. 给定网络 $G=(V,E)$,如果删除网络中的任意一条链路 $l\in E,G'=(V,E'),E'=E-\{l\}$,则 $R(G,G')$必定减小.

证明:当删除网络中的任意一条链路 $l\in E$ 时,新的拓扑中将不再包含该链路,因此根据该拓扑 $G'=(V,E')$计算出的备份路径将不再包含该链路.对于网络中的任意源-目的节点对 o 和 d,如果链路 $l\notin P(o,d,G)$,则该节点对的交叉度将不会受到影响,当 $l\in P(o,d,G)$时,则该节点对之间的交叉度将会至少减少 1.因为在网络拓扑 $G=(V,E)$中,对于任意一条链路 $l\in E$,该链路一定会出现在最短路径中,因此,上述定理成立. □

定理 2. 给定网络 $G=(V,E)$,如果删除网络中的一组链路 $L\subset E,G'=(V,E'),E'=E-L$,则 $R(G,G')$必定减小.

证明:由定理 1 可知,当删除网络中的任意一条链路时,$R(G,G')$必定减少.因此,当删除多条链路时,$R(G,G')$必定减少,上述定理成立. □

我们可以将该问题具体表达为:给定网络 $G=(V,E)$,如何删除一组链路 L,从而使得 $R(G,G')$最小.其中,$G'=(V,E'),E'=E-L,d(i,j)=\infty,(i,j)\in L,d(i,j)$表示该链路的权值.上述问题是一个整数线性规划(integer linear programming,简称 ILP)问题,即

$$\min R(G,G') \tag{4}$$

s.t.

$$D(u,u,G)=0,u\in V \tag{5}$$

$$D(u,u,G')=0,u\in V \tag{6}$$

$$w(i,j)+D(i,d,G)-D(j,d,G)\geqslant 0,i,j,d\in V \tag{7}$$

$$w(i,j)+D(i,d,G')-D(j,d,G')\geqslant 0,i,j,d\in V \tag{8}$$

$$x(i,j,d)\in\{0,1\},i,j,d\in V \tag{9}$$

$$x(i,j,d)=1,(i,j)\in P(i,d,G) \tag{10}$$

$$x(i,j,d)=0,(i,j)\notin P(i,d,G) \tag{11}$$

$$y(i,j,d)\in\{0,1\},i,j,d\in V \tag{12}$$

$$y(i,j,d)=1,(i,j)\in P(i,d,G') \tag{13}$$

$$y(i,j,d)=0,(i,j)\notin P(i,d,G') \tag{14}$$

$$x(i,j,d)+w(i,j)+D(i,d,G)-D(j,d,G)\geqslant 1,i,j,d\in V \tag{15}$$

$$x(i,j,d)+\frac{(w(i,j)+D(i,d,G)-D(j,d,G))}{M}\leqslant 1,i,j,d\in V \tag{16}$$

$$y(i,j,d)+w'(i,j)+D(i,d,G')-D(j,d,G')\geqslant 1,i,j,d\in V \tag{17}$$

$$y(i,j,d)+\frac{(w'(i,j)+D(i,d,G')-D(j,d,G'))}{M}\leqslant 1,i,j,d\in V \tag{18}$$

$$w(i,j)=w(j,i),w(i,j)\in\{1,2,\ldots,\max\},i,j\in V \tag{19}$$

$$z(i,j)\in\{0,1\},i,j\in V \tag{20}$$

$$z(i,j)=1,(i,j)\in E,i,j\in V \tag{21}$$

$$z(i,j)=0,(i,j)\notin E,i,j\in V \tag{22}$$

$$f(i,j)\in\{0,1\},i,j\in V \tag{23}$$

$$f(i,j)=1,(i,j)\in L,i,j\in V \tag{24}$$

$$f(i,j)=0,(i,j)\notin L,i,j\in V \tag{25}$$

$$f(i,j)+z(i,j)=1,(i,j)\in E,i,j\in V \tag{26}$$

下面,我们将解释该ILP模型.在上述模型中,公式(4)为本文的目标函数,即求解包含公共边数量最小的默认路径和备份路径.公式(5)和公式(6)说明在网络中节点到自己的最小代价是 0.公式(7)和公式(8)说明网络中所有节点遵循最短路径原则.公式(9)~公式(11)中的变量 $x(i,j,d)$表明,在 G 中链路(i,j)是否包含在 i 到 d 的最短路径中:如果包含在最短路径中,则该值为 1;否则为 0.公式(12)~公式(14)中,变量 $y(i,j,d)$表明在 G'中,链路(i,j)是否包含在

i 到 d 的最短路径中:如果包含在最短路径中,则该值为 1;否则为 0.公式(15)和公式(16)是在 G 中的松弛条件,在公式(15)中,假如 $x(i,j,d)=1$,则公式(15)和公式(7)是一样的,假如 $x(i,j,d)=0$,公式(15)将会变形为 $w(i,j)+D(i,d,G)-D(j,d,G)\geqslant 1$;在公式(16)中,假如 $x(i,j,d)=1$,公式(16)将会变形为 $w(i,j)+D(i,d,G)-D(j,d,G)\leqslant 0$,因此合并公式(14)和公式(16),当 $x(i,j,d)=1$ 时,$w(i,j)+D(i,d,G)-D(j,d,G)=0$,如果 $x(i,j,d)=0$,公式(16)将会变形为 $w(i,j)+D(i,d,G)-D(j,d,G)\leqslant M$,其中,$M=2\times\max(w(i,j))(i,j\in E)$.公式(17)和公式(18)是在 G' 中的松弛条件,在公式(17)中,假如 $y(i,j,d)=1$,则公式(17)和公式(8)是一样的,假如 $y(i,j,d)=0$,则公式(17)将会变形为 $w(i,j)+D(i,d,G')-D(j,d,G')\geqslant 1$;在公式(18)中,假如 $y(i,j,d)=1$,公式(18)将变为 $w(i,j)+D(i,d,G')-D(j,d,G')\leqslant 0$,因此合并公式(17)和公式(18),当 $y(i,j,d)=1$ 时,$w(i,j)+D(i,d,G')-D(j,d,G')=0$.假如 $y(i,j,d)=0$,公式(18)将变为 $w(i,j)+D(i,d,G')-D(j,d,G')\leqslant M$,其中,$M=2\times\max(w(i,j))(i,j\in E)$.公式(19)说明链路的代价是对称的.公式(20)~公式(22)中,变量 $z(i,j)\in\{0,1\}$,$z(i,j)\in\{0,1\}$,$i,j\in V$ 表示链路(i,j)是否属于集合 E:如果属于集合 E,则该值为 1;反之,则该值为 0.公式(23)~公式(25)中,变量 $f(i,j)\in\{0,1\}$,$i,j\in V$ 表示链路(i,j)是否属于集合 L:如果属于集合 L,则该值为 1;反之,则该值为 0.公式(26)表示网络中的任意链路(i,j)不能同时属于集合 E 和集合 L.

2 算 法

上述描述的 ILP 问题是一个 NP 难题,因此计算复杂度较高.如果在小型网络中(如 Abilene),可以利用CPLEX计算最优解,但是,如果在大型网络(如Sprint),利用CPLEX则无法在有限时间内计算出正确的结果.因此在大型网络中,通常利用启发式方法计算近似解.下面将介绍本文如何巧妙地计算该 NP 难题.

2.1 DeleteLink算法

本节将介绍如何采用模拟退火算法解决上述问题.算法的基本思路为:每次选择一条性能最优的链路从网络中删除,直到满足目标条件.算法 1 介绍了 DeleteLink 是如何运行的.将 G'的值初始化为 G,固定初始温度 T_0,设置删除链路集合的初始值 $L=\varnothing$,用变量 M 记录网络拓扑中边的集合,计算路径交叉度(算法第 1 行~第 5 行).为了获得删除链路的集合 L,需要执行一系列的循环过程,直到 $R(G,G')=0$,$T=0$ 或者 $M=\varnothing$之一成立.$nei(E)$的作用为随机删除一条链路$(p,q)\in E$,将链路的代价调整为 $w(m,n)=\infty$,判断网络连通性:如果网络连通,则返回该链路,将 E'修改为 $E'=E'-\{p,q\}$;如果网络不连通,则撤销删除链路的操作.函数 $\arg\min_{(p,q)\in nei(E)}(R(G,G'))$ 的作用为返回 $R(G,G')$ 的值最小时对应的链路(m,n)(算法第 7 行).当执行完上述函数后,更新网络拓扑 G'、变量 M 和路径交叉度(算法第 8 行~第 11 行).当 $R(G,G')<currentDisjoint$ 或者此时系统中的温度大于利用随机函数产生的温度时,将链路(m,n)加入到集合 L 中(算法第 12 行~第 14 行);否则将撤销删除链路(m,n)的操作(算法第 15 行),最后返回删除链路集合 L(算法第 20 行).

算法 1. DeleteLink.

Input: $G=(V,E)$, T_0.

Output: L.

1: $G'=G$
2: $T=T_0$
3: $L=\varnothing$
4: $M=E$
5: $currentDisjoint=originalDisjoint\leftarrow R(G,G')$
6: **While** $currentDisjoint>0$ and $T>0$ and $M>0$
7: $(m,n)\leftarrow \arg\min_{(p,q)\in nei(E)}(R(G,G'))$
8: $E'=E'-\{m,n\}$
9: $G'=(V,E')$
10: $M=M-\{(m,n)\}$

11: *currentDisjoint*=*R*(*G*,*G*′)
12: **If** *originalDisjoint*<*currentDisjoint* or *T*>*random*(T_0) **then**
13: *currentDisjoint*←*R*(*G*,*G*′)
14: *L*=*L*∪(*m*,*n*)
15: **else**
16: *E*′=*E*′∪{*m*,*n*}
17: **EndIf**
18: *T*←*T*−1
19: **EndWhile**
20: **Return** *L*

2.2 B-DeleteLink算法

上述算法是一种典型的贪心算法.为了删除一条链路,该算法需要经过数次的迭代过程,该算法的时间复杂度较高.因此,为了降低上述算法的时间复杂度,我们提出了一种高效的删除链路的方案.该方案的核心思想是:首先对网络中的链路按照关键度进行排序,然后按照链路的关键度从大到小依次删除链路.

从上述的讨论可知,在计算备份路径的时候,删除的链路将不会被使用.为了使得路径交叉度最小,删除最短路径中边的介数最大的链路将会使路径交叉度减小的最多,因此我们用介数来衡量链路的关键度.链路 l 的介数为网络中所有最短路径经过该链路的次数,可以形式化表示为:

$$B(l)=\sum_{\forall o,d}k(l),$$

$$k(l)=\begin{cases}1, & l\in P(o,d,G)\\ 0, & \text{otherwise}\end{cases}.$$

在解决了链路关键度问题的基础上,算法 2 详细描述了基于关键链路的算法的具体执行过程.计算网络中所有链路的介数,并且按照介数的大小对链路进行降序排序,将排序后的结果存储在集合 M 中(第 1 行~第 3 行).设 $G'=G$,计算初始路径交叉度(第 4 行、第 5 行),设置集合 L 的初值为空集(第 6 行).算法每次从集合 M 中选择一条链路 l 删除,更新 G',计算删除该链路后网络的交叉度(第 8 行~第 11 行).如果删除该链路后网络依然连通,则将该链路从集合 M 中删除,加入到集合 L 中(第 11 行~第 13 行);否则,该链路无法从网络中删除,将该链路重新插入到 G'中,(第 15 行).最后返回删除链路集合 L.

算法 2. B-DeleteLink.
Input: *G*=(*V*,*E*).
Output: *L*.
1: 计算网络中所有链路的介数
2: 按照链路介数对链路进行降序排列
3: 将排序后的节点存储在链表 *M* 中
4: *G*′=*G*
5: *currentDisjoint*=*originalDisjoint*←*R*(*G*,*G*′)
6: *L*=∅
7: **While** *M* is not empty
8: *M*=*M*−{*l*}
9: *E*′=*E*′−{*l*}
10: *G*′=(*V*,*E*′)
11: *currentDisjoint*←*R*(*G*,*G*′)
12: **If** *connect*(*G*′) **then**

13: $L=\{l\}\cup L$

14: **else**

15: $E'=E'\cup\{l\}$

16: **EndWhile**

17: **Return** L

2.3 算法讨论

定理 3. 算法 DeleteLink 的时间复杂度为 $\min(T_0,|E|)\times|E|\times O(|V|\lg|V|+|E|+\lg|E|)$.

证明:为了计算出最终需要删除的边的集合,算法需要执行 $\min(T_0,|E|)$次函数 $\arg\min_{(p,q)\in nei(E)}(R(G,G'))$,该函数的时间复杂度为$|E|\times(O(|V|\lg|V|+|E|)+O(\lg|E|))$,因此,算法 DeleteLink 的时间复杂度可以表示为

$$\min(T_0,|E|)\times|E|\times O(|V|\lg|V|+|E|+\lg|E|).$$ □

定理 4. B-DeleteLink 算法的时间复杂度为$|V|\times O(|V|\lg|V|+|E|)+|E|\times O(|V|+|E|)$.

证明:B-DeleteLink 需要计算网络中边的介数,该算法的时间复杂度为$|V|\times O(|V|\lg|V|+|E|)$.在删除链路时,需要判断图的连通性,该执行过程的时间复杂度为$|E|\times O(|V|+|E|)$,因此,该算法的时间复杂度为

$$|V|\times O(|V|\lg|V|+|E|)+|E|\times O(|V|+|E|).$$ □

上面介绍的两种算法都没有考虑备份路径的拉伸度.如果在实际中需要考虑备份路径的拉伸度,只需对上述算法做微小的调整,就可以计算出符合条件的备份路径.在算法 1 中,只需要在第 6 行加入路径拉伸度限制条件即可;在算法 2 中,只需在第 12 行加入路径拉伸度限制条件即可.

3 转发机制

在本节中,我们将详细介绍网络中报文的转发机制.对于网络中的节点$\forall v\in V$,该节点在其 FIB 表中存储两个到达所有目的的下一跳,其中一个为默认下一跳,另一个为备份下一跳.为了降低网络的额外开销,本文利用纯 IP 路由协议进行报文的转发,使用 IP 报文头部的 TOS 字段,利用该字段记录报文转发过程中是否遇到过故障:如果遇到过,则该字段的值为 1;否则为 0.下面将描述当网络中的某个节点收到一个报文时,该节点如何转发该报文.图 2 介绍了报文的具体转发过程.

(1) 如果接受到的报文的头部中 TOS 的数值为 0,则执行步骤(1.1)或者步骤(1.2).

(1.1) 如果该节点到达目的节点的默认下一跳没有出现故障,则将该报文直接转发到该节点的默认下一跳;

(1.2) 如果该节点到达目的节点的默认下一跳出现故障,则将报文头部的 TOS 字段的数值修改为 1,然后将该报文直接转发到该节点的备份下一跳.

(2) 如果接受到的报文的头部 TOS 的数值为 1,则将该报文直接转发到该节点的备份下一跳.

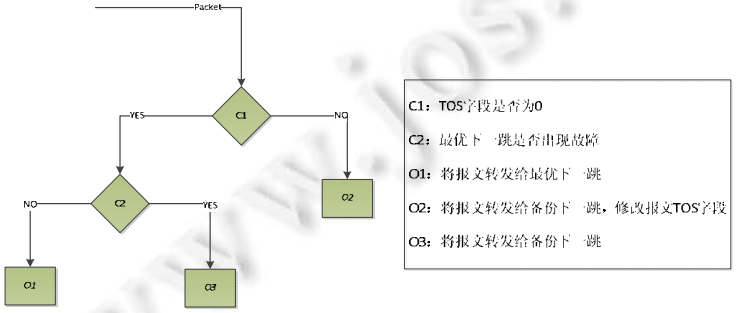

Fig.2 Forwarding mechanism

图 2 转发机制

4 实验及结果分析

本节将通过路径交叉度比率、网络可用性和计算效率来比较算法 DeleteLink、B-DeleteLink、S-DeleteLink(按照顺序删除链路),R-DeleteLink(按照随机方法删除链路)和 LFA 的性能.在比较的过程中,为了体现公正性:(1) 利用LFA算法时,随机选择一条路径作为备份路径.因为LFA算法可以计算出多个备份下一跳,而本文提到的算法只为每一个源-目对计算一个备份下一跳.这就类似于面对同一个问题时,LFA 拿出多个解决方案来解决一个问题,而本文的算法只有一个方案解决问题,这会使得解决掉问题的概率在解决问题之前就有所不同,从而导致比较不公平.(2) 本文提出的算法不与红绿树之类的算法进行比较.因为红绿树之类的方法无法直接部署在互联网中,所以两者之间没有进行比较的必要.

4.1 网络拓扑

为了使比较结果更加准确和具有一般性,本文在不同拓扑中分别运行了算法 DeleteLink、B-DeleteLink、S-DeleteLink、R-DeleteLink 和 LFA,由此来证明 B-DeleteLink 算法的高效性.3 种拓扑类型分别为 Abilene[30]、Rocketfuel[31]测量的拓扑(见表 2)和使用 Brite[32,33]生成的拓扑(参数见表 3).在使用 Brite 生成拓扑时,假设链路权值具有对称性[34],Brite 的模型设置为 Waxman,节点数量为 50~1 000,*alpha* 和 *beta* 的数值分别为 0.15 和 0.2,节点平均度设置为 2~10,模式设置为路由器,节点位置服从重尾分布,链路带宽的大小为 10~1 024,链路的代价为带宽的倒数.

Table 2 Rocketfuel topology

表 2 Rocketfuel 拓扑

自治系统编号	自治系统名称	路由器数量	边数量
1221	Telstra	108	153
1239	Sprint	315	972
3257	Tiscali	161	328
3967	Exodus	79	147

Table 3 Parameters for generated topologies using Brite

表 3 Brite 生成拓扑的参数设置

生成拓扑的模型	Waxman
路由器数量	50~1 000
alpha	0.15
beta	0.2
网络度	2~10
生成拓扑的模式	路由器
路由器的位置	重尾分布
链路带宽大小	10.0~1 024.0

4.2 路径交叉度比率

我们将执行算法后的路径交叉度除以执行算法前的路径交叉度定义为路径交叉度比率.图 3 是不同算法在真实拓扑和 Rocketfuel 测量拓扑中运行的结果.从该图中可以看出,我们提出的所有算法对应的路径交叉度比率明显低于 LFA.DeleteLink 和 B-DeleteLink 具有相同的性能,明显优于 S-DeleteLink、R-DeleteLink 的性能.这是因为 S-DeleteLink 和 R-DeleteLink 在删除链路时没有任何依据,而 DeleteLink 和 B-DeleteLink 根据链路的关键度来删除链路,因此可以达到较好的结果.例如在 Sprint 拓扑中,LFA 的路径交叉度比率接近 70%,而 DeleteLink 和 B-DeleteLink 的值为 11%,S-DeleteLink 和 R-DeleteLink 的值分别为 21%和 23%.在 Abilene 中,我们提出的算法的数值基本接近,这是因为该拓扑仅仅由 14 条链路组成,最多可以删除 4 条链路,所以每种方法删除的链路基本一致.

图 4 描绘了不同算法对应的路径交叉度比率随着网络拓扑大小的变化情况,图 5 说明了不同算法对应的路径交叉度比率随着网络节点平均度的变化情况.根据图 4 和图 5 可知,DeleteLink 和 B-DeleteLink 的性能是最优的,LFA 的性能是最差的,随着网络节点平均度的增加,各种算法的性能都有明显的提升.当网络节点平均度增加

时,网络中的链路数量将会增加,因此节点间存在不相交路径的概率将会随之增加.利用启发式算法虽然可以加快求解问题的速度,但是也会损失计算的精度.为了验证 B-DeleteLink 和最优解之间的差距,我们利用 CPLEX 计算出了 ILP 问题在 Abilene 拓扑的最优解.结果表明,在 Abilene 中,B-DeleteLink 和最优解是相同的.由于在大型网络中 CPLEX 的计算速度是特别慢的,几乎无法求解出最优解,所以只能在小规模网络中对 B-DeleteLink 和最优解的差距进行验证.

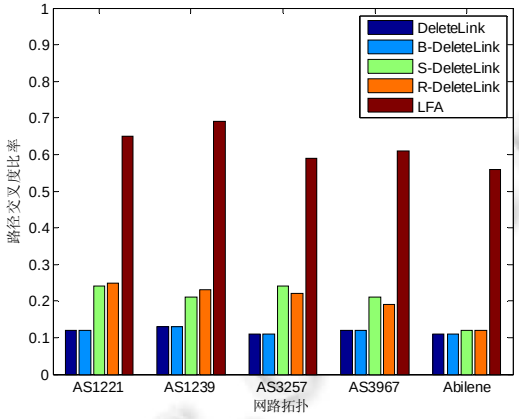

Fig.3 Path intersection ratio on Abilene and measured topologies for different algorithms

图 3 不同算法在 Abilene 和测量拓扑中的路径交叉度比率的结果

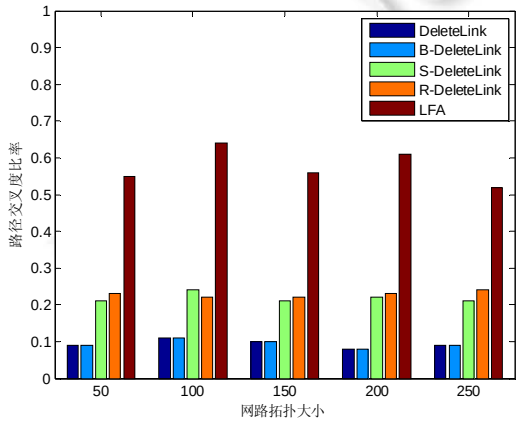

Fig.4 Path intersection ratio vs. network topology size on Brite topologies for different algorithms when average node degree is 4

图 4 当节点平均为 4 时,不同算法在 Brite 拓扑中对应的路径交叉度比率随网络拓扑大小的变化情况

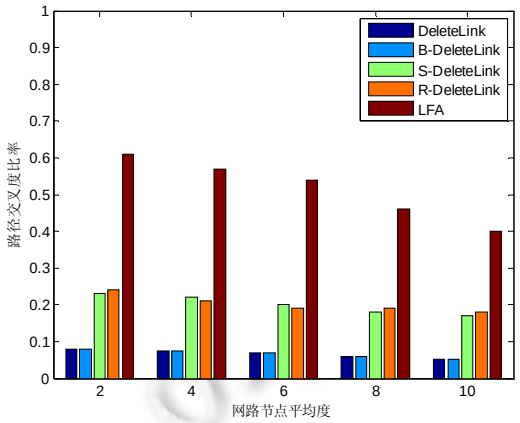

Fig.5 Path intersection ratio vs. average node on Brite topologies for different algorithms when network topology size is 1000

图 5 当拓扑大小为 1 000 时,不同算法在 Brite 拓扑中对应的路径交叉度比率随网络节点平均度的变化情况

4.3 网络可用性

本节用网络断开概率来衡量网络中报文的丢失率.网络断开概率(disconnect fraction)可以表示为:网络中每条链路的失效概率相同时,受故障影响的节点对的数量和网络中所有节点对之间的比值.图 6~图 8 分别表示在 Abilene、Ebone 和 Sprint 拓扑中,不同算法对应的网络断开概率.从这几个图形中我们可以看出,随着链路的失效概率增加,网络断开概率随之增加.DeleteLink 和 B-DeleteLink 具有相似的性能,其性能远远优于另外 3 种算

法.当网络中链路的失效概率增加时,每条链路断开的可能性将会增加,在这种情况下,最短路径和备份路径有可能都会失效.但是因为 B-DeleteLink 计算的最短路径和备份路径具有较小的路径交叉度,所以 B-DeleteLink 的可用性明显优于另外几种算法.例如,当网络中所有的链路失效概率均为 0.1 时,对于 Sprint 拓扑结构而言,算法 DeleteLink、B-DeleteLink、S-DeleteLink、R-DeleteLink 和 LFA 的对应的网络断开概率分别为 4%,4%,12%,12%和 16%.因为在 Brite 生成拓扑中的实验结果与上述结果类似,所以我们省略了该部分的结果.

为了进一步验证不同算法在真实流量矩阵下报文的丢失情况,我们在 Abilene 上进行了实验,流量数据的采集时间为 2004 年 3 月 8 日.图 9 描述了当链路的断开概率为 0.1 时,不同算法在 Abilene 中真实流量情况下报文丢失情况.图 9 的横坐标为时间间隔,纵坐标为 Disconnect Fraction.从图 9 可以看出,该图中对应的 Disconnect Fraction 明显小于图 6 中的 Disconnect Fraction.这是因为在图 6 中假设所有的节点对之间都相互发送了报文,但是在真实情况下,在某些时间段内并不是所有节点对之间都进行了报文的发送,因此在真实的流量矩阵条件下,不同算法对应的 Disconnect Fraction 会明显降低.

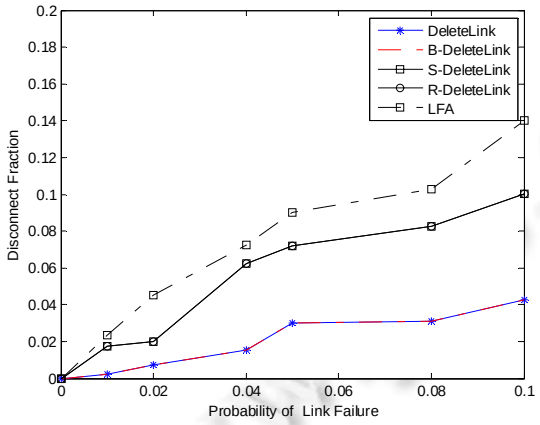

Fig.6 Results on Abilene for different algorithms

图 6 不同算法在 Abilene 中的结果

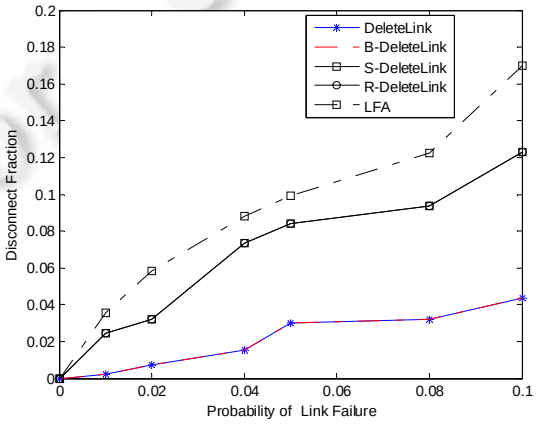

Fig.7 Results on Exodus for different algorithms

图 7 不同算法在 Exodus 中的结果

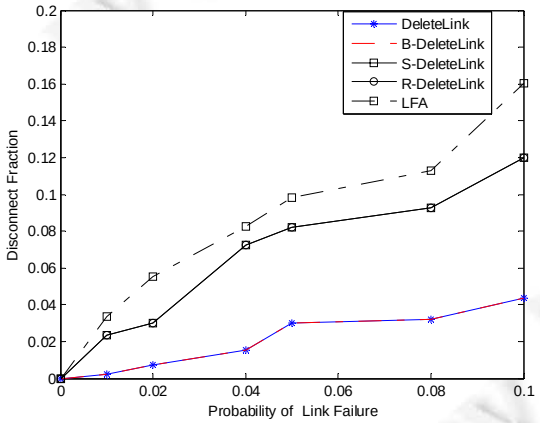

Fig.8 Results on Sprint for different algorithms

图 8 不同算法在 Sprint 中的结果

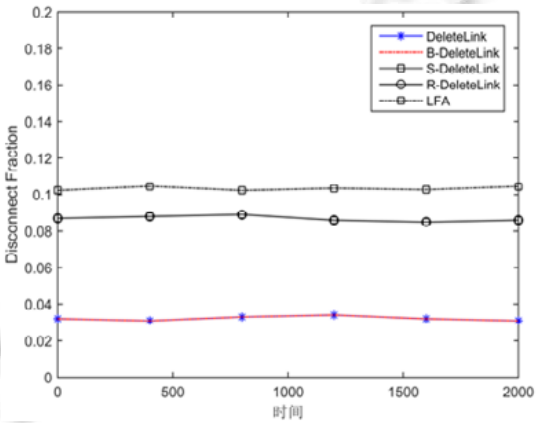

Fig.9 Results on Abilene for different algorithms when the probability of link failure is 0.1

图 9 当链路失效概率为 0.1 时,不同算法在 Abilene 中的结果

4.4 计算效率

本节利用实际计算时间来比较 DeleteLink 和 B-DeleteLink 对应的计算效率.该实验运行在处理器为酷睿 i5 和内存 2GB 的台式电脑上.图 10 描述了上述两种算法在真实拓扑和测量拓扑中的实际执行时间.从图中我们可以得出,B-DeleteLink 的计算时间仅仅是 DeleteLink 的计算时间的 1/10000.在 Sprint 拓扑结构中,DeleteLink 需要 2 天的时间才能计算出结果,而 B-DeleteLink 仅仅需要 18s 就可以得到最后解决方案.目前,骨干网中部署的路由器的配置和该台式机的配置基本接近,因此在该配置环境中进行计算效率的实验是合理的.B-DeleteLink 极大地提高了算法的计算效率,降低了算法开销,更容易实际部署.

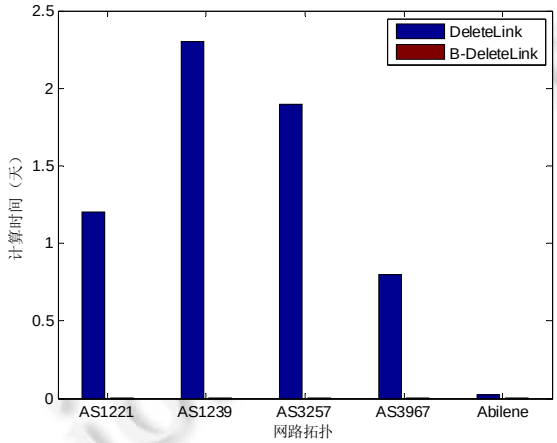

Fig.10 Computation time for different algorithms on public topologies

图 10 不同算法在公开拓扑中的计算时间

我们对算法进行了实际部署,在实验中,我们首先在 11 台电脑上安装了路由器软件 Quagga 和 Click 模拟路由器,然后部署了 DeleteLink 和 B-DeleteLink 算法,最后按照 Abilene 的拓扑结构对电脑进行了连接.实验结果表明,在模拟实验中,DeleteLink 和 B-DeleteLink 算法的实际计算时间分别为 12s 和 0.01s.在实际部署实验中,DeleteLink 和 B-DeleteLink 算法的实际计算时间分别为 15s 和 0.012s.从该实验可以看出,对于一个小型网络 Abilene,B-DeleteLink 的计算速度比 DeleteLink 的计算速度快 100 倍.

5 结束语

为了更好地将域内路由保护方案部署在实际的互联网中,克服已经存在的方案的不足,本文为所有节点对计算两条路径,分别是默认路径和备份路径,以提升路由可用性,降低报文丢失率.在文中,我们首先将计算默认路径和备份路径描述为一个整数规划问题,然后分别提出利用模拟退火算法(DeleteLink)和关键链路算法(B-DeleteLink)解决该问题.然而 DeleteLink 的计算复杂度较高,不适合在实际网络中部署.B-DeleteLink 不仅具有较小的计算开销,并且与互联网是兼容的,因此是一种具有较强竞争力的域内路由保护方案.

References:

[1] Clark, D. The design philosophy of the DARPA Internet protocols. ACM Sigcomm Computer Communication Review, 1988,18(4): 106−114.

[2] Varshney U, Snow A, Mcgivern M, *et al*. Voice over IP. Communications of the ACM, 2002,45(1):89−96.

[3] Geng H, Shi X, Wang Z, *et al*. A hop-by-hop dynamic distributed multipath routing mechanism for link state network. Computer Communications, 2018,116:225−239.

[4] Goode B. Voice over internet protocol (VoIP). Proc. of the IEEE, 2002,90(9):1495−1517.

[5] Krist P. Scalable and efficient multipath routing: Complexity and algorithms. In: Proc. of the Int'l Conf. on Network Protocols (ICNP). IEEE, 2015. 376−385.

[6] Zheng J, Xu H, Zhu X, *et al*. We've got you covered: Failure recovery with backup tunnels in traffic engineering. In: Proc. of the Int'l Conf. on Network Protocols (ICNP). IEEE, 2016. 1−10.

[7] Hartert R, Vissicchio S, Schaus P, *et al*. A declarative and expressive approach to control forwarding paths in carrier-grade networks. ACM Sigcomm Computer Communication Review, 2015,45(5):15−28.

[8] Moy J. RFC 2328: OSPF Version 2, 1998. http://www.rfc-editor.org/info/rfc1583

[9] Basu A, Riecke J. Stability issues in OSPF routing. ACM Sigcomm Computer Communication Review, 2001,31(4):225−236.

[10] Markopoulou A, Iannaccone G, Bhattacharyya S, *et al*. Characterization of failures in an operational IP backbone network. IEEE/ACM Trans. on Networking, 2008,16(4):749−762.

[11] Hou M, Wang D, Xu M, *et al*. Selective protection: A cost-efficient backup scheme for link state routing. In: Proc. of the IEEE Int'l Conf. on Distributed Computing Systems (ICDCS). IEEE, 2009. 68−75.

[12] Geng HJ, Shi XG, Yin X, Wang ZL, Yin SP. Algebra and algorithms for multipath QoS routing in link state networks. Journal of Communications and Networks, 2017,19(2):189−200.

[13] Yallouz J, Rottenstreich O, Babarczi P, *et al*. Optimal link-disjoint node-"somewhat disjoint" paths. In: Proc. of the Int'l Conf. on Network Protocols (ICNP). IEEE, 2016. 1−10.

[14] Kwong KW, Gao L, Zhang ZL. On the feasibility and efficacy of protection routing in IP networks. IEEE/ACM Trans. on Networking, 2011,19(5):1543−1556.

[15] Gopalan A, Ramasubramanian S. IP fast rerouting and disjoint multipath routing with three edge-independent spanning trees. IEEE/ACM Trans. on Networking, 2016,24(3):1336−1349.

[16] Antonakopoulos S, Bejerano Y, Koppol P. Full protection made easy: The DisPath IP fast reroute scheme. IEEE/ACM Trans. on Networking, 2015,23(4):1229−1242.

[17] Xu A, Bi J, Zhang B. Failure inference for shortening traffic detours. In: Proc. of the Int'l Symp. on Quality of Service. IEEE, 2016. 1−10.

[18] Sommers J, Barford P, Eriksson B. On the prevalence and characteristics of MPLS deployments in the open Internet. In: Proc. of the 2011 ACM SIGCOMM Conf. on Internet Measurement Conf. ACM, 2011. 445−462.

[19] Yang Y, Xu M, Li Q. Fast rerouting against multi-link failures without topology constraint. IEEE/ACM Trans. on Networking, 2018,26(1):384−397.

[20] Sridharan A, Guerin R, Diot C. Achieving near-optimal traffic engineering solutions for current ospf/is-is networks. IEEE/ACM Trans. on Networking, 2005,13(2):234−247.

[21] Atlas A, Zinin A. Basic specification for IP fast reroute: Loop-free alternates. RFC5286, 2008. http://www.rfc-editor.org/info/rfc5286

[22] Yang X, Wetherall D. Source selectable path diversity via routing deflections. In: Proc. of the ACM Special Interest Group on Data Communication (SIGCOMM). 2006. 159−170.

[23] Kvalbein A, Hansen AF, Čičić T, *et al*. Fast IP network recovery using multiple routing configurations. In: Proc. of the Int'l Conf. on Computer Communications (INFOCOM). IEEE, 2006. 1−11.

[24] Lakshminarayanan K, Caesar M, Rangan M, *et al*. Achieving convergence-free routing using failure-carrying packets. In: Proc. of the ACM Special Interest Group on Data Communication (SIGCOMM). 2007. 241−252.

[25] Enyedi G, Rétvári G, Szilágyi P, Császár A. IP fast ReRoute: Lightweight not-via without additional addresses. In: Proc. of the INFOCOM. 2009. 2771−2775.

[26] Jayavelu G, Ramasubramanian S, Younis O. Maintaining colored trees for disjoint multipath routing under node failures. IEEE/ACM Trans. on Networking, 2009,17(1):346−359.

[27] Kini S, Ramasubramanian S, Kvalbein A, Hansen AF. Fast recovery from dual link failures in IP networks. In: Proc. of the Int'l Conf. on Computer Communications (INFOCOM). IEEE, 2009. 1368−1376.

[28] Medard M, Finn S, Barry R, Gallagher R. Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs. IEEE/ACM Trans. on Networking, 1999,7(5):641−652.

[29] Lee YO, Narasimha Reddy AL. Constructing disjoint paths for failure recovery and multipath routing. Computer Networks, 2012, 56(2):719−730.

[30] Network AB. Advanced networking for research and education. 2003. http://abilene.internet2.edu

[31] Spring N, Mahajan R, Wetherall D, *et al*. Measuring ISP topologies with rocketfuel. IEEE/ACM Trans. on Networking, 2004,12(1): 2−16.

[32] http://www.cs.bu.edu/brite/

[33] Heckmann O, Piringer M, Schmitt J, *et al*. Generating realistic ISP-level network topologies. Communications Letters IEEE, 2003, 7(7):335−336.

[34] Gjoka M, Ram V, Yang X. Evaluation of IP fast reroute proposals. In: Proc. of the Communication Systems Software and Middleware. 2007. 1−8.

耿海军(1983－),男,山西晋中人,博士,副教授,主要研究领域为软件定义网络,路由算法和网络体系结构.

施新刚(1980－),男,博士,副研究员,主要研究领域为路由协议和网络测量.

王之梁(1978－),男,博士,副教授,CCF 专业会员,主要研究领域为下一代互联网和路由算法.

尹霞(1972－),女,博士,教授,博士生导师,CCF 高级会员,主要研究领域为下一代互联网和协议测试.

胡治国(1977－),男,博士,副教授,CCF 专业会员,主要研究领域为网络测量和路由算法.