基于子树丢包模式的链路丢包率快速推断算法

尹文涛,杨京礼,姜守达,魏长安
(哈尔滨工业大学自动化测试与控制系,黑龙江哈尔滨 150080)

摘 要: 为提高网络链路丢包率的测量速度,本文提出一种基于子树丢包模式的链路丢包率推断算法. 该算法通过选择合理的链路丢包率初始值以减少迭代次数;根据端到端测量结果将网络拓扑划分为传输状态确定性区域和非确定性区域,避免确定性区域冗余分解造成的时间开销;通过对非确定性区域子树丢包模式按层分解,以子树丢包模式为基本计算单元,减少非确定性区域链路丢包的重复分解过程,提高链路丢包率计算速度. 仿真结果表明,该算法能在不损失测量精度的前提下,减少链路丢包率测量总时间,提高测量速度.

关键词: 网络测量; 网络层析成像; 链路丢包率; 丢包模式
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112 (2016)03-0565-07
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2016.03.011

A Fast Network Link Loss Inference Algorithm Based on Subtree Loss Pattern

YIN Wen-tao, YANG Jing-li, JIANG Shou-da, WEI Chang-an
(*Automatic Test and Control Institute, Harbin Institute of Technology, Harbin, Heilongjiang* 150080, *China*)

Abstract: In order to improve the efficiency of link loss inference algorithm, a novel inference algorithm based on subtree loss pattern is proposed. The iterations of the algorithm are reduced by obtaining a more appropriate initialization of loss rates. According to the outcomes of end-to-end measurements, this algorithm partitions the network topology into one area in which the transmission state is determinate, and several areas in which the transmission state is indeterminate. By decomposing all the indeterminate areas, a subtree loss pattern database is constructed. The loss rate is calculated based on the loss pattern. Through reducing the redundancy decomposition process, it can speed up the process of the inference of the link loss rate. Simulation results show that the algorithm can reduce the time of link loss rate inference with identical accuracy.

Key words: network measurement; network tomography; link packet loss rate; loss pattern

1 引言

随着计算机网络规模的扩大,以及网络安全需求的不断提高,传统的基于网络中间节点协作的网络测量方法面临巨大挑战. 网络层析成像技术[1]将医学上的计算机层析成像思想引入到计算机网络测量中,根据在网络边界上获得的端到端的测量数据来分析和推断网络拓扑结构[2]、链路丢包率[3,4]、链路时延[5]等网络内部性能参数,对网络中的故障链路进行诊断和定位. 网络层析成像技术可以在无需中间节点协作的条件下估计网络内部链路参数,克服了传统方法的不足,是目前大规模网络性能测量最有前景的方法之一.

目前,有些学者已经提出一些基于网络层析成像技术的链路丢包率推断算法. 文献[3,6]提出两种基于极大似然估计的链路丢包率推断算法. 其中文献[3]提出的DE(Direct Estimator)算法通过求解高阶方程计算链路丢包率;文献[6]提出采用EM(Expectation Maximization)算法通过执行一个迭代过程逐步搜索高维解空间进行求解. 该类算法精度最高,但当网络规模较大时,算法复杂度极高. 文献[7,9]首先搜索网络中的拥塞链路,通过将非拥塞链路从路由矩阵中去除,以化简路由矩阵,然后进行链路丢包率求解. 该类算法具有较低的算法复杂度,但在链路丢包率计算过程中将部分丢包率不为零的链路从路由矩阵中去除,因此会产生一定的误差,且该类算法均建立在网络中拥塞链路比例较小的假设条件之下. 文献[10]提出一种自底向上的链路丢包率估计算法,首先估计叶链路丢包率,然后再自底向上逐层求解链路丢包率. 该算法利用兄弟链路之间的相关性,直接给出解析解,极

收稿日期:2014-05-08;修回日期:2014-08-27;责任编辑:梅志强
基金项目:黑龙江省博士后基金(No. LBHZ11171)

大地降低了算法复杂度,但在逐层求解的过程中引入累积误差,导致算法精度有所降低. 文献[11]对自底向上的方法进行改进,在计算链路 k 的成功传输概率统计父节点收到探测包个数时,忽略了节点 k 的父节点收到,但同时在链路 k 和链路 k 的兄弟节点都丢包的探测包,影响了算法精度. 文献[12]将 EM 算法和自底向上算法合并,首先采用自底向上方法直接求解叶链路丢包率,然后将求解的叶链路丢包率作为初始值,采用 EM 算法推断内部链路丢包率. 该算法解决了自底向上方法内部链路丢包率估计引入累积误差的问题,但估计内部链路时,仍通过执行迭代过程搜索高维解空间进行求解,因而算法复杂度仍较高. 文献[13]提出的 FPA(Fast Path-based Approach)算法利用子树丢包之间的相关性,直接求解链路丢包率,该算法具有较低的算法复杂度,但由于该算法在求解过程中忽略了部分全局信息,算法精度有所降低.

本文根据网络丢包的特性,提出了一种基于子树丢包模式的链路丢包率推断算法,在保证精度的前提下,提升推断过程的计算速度,减少链路丢包率测量的总时间.

2 模型与假设

2.1 网络模型

用 $T=(V,L)$ 表示逻辑树状网络拓扑模型[3,13],其中 V 为节点集合,代表网络中的路由器和主机,L 为连接节点的链路集合. 节点 $O\in V$ 为 T 的根节点. 节点集合 $R\subset V$(无子节点的节点)表示所有的叶节点,即探测报文接收节点,$|R|$ 为叶节点个数. 每个非叶节点 k 都至少有一个子节点,节点 k 的子节点集合用 $c(k)$ 表示. 每个非根节点 k 有且仅有一个父节点,用 $f(k)$ 表示. 链路 $(f(k),k)\in L$ 记为链路 k. 定义 $f^1=f$ 且 $f^n(k)=f(f^{n-1}(k))$,其中 n 为正整数. 用集合 $a(k)=\{i\in V\mid \exists n>0, i=f^n(k)\}$ 表示节点 k 的祖先节点,用集合 $U=V\backslash\{O\}$ 表示非根节点集合,用集合 $I=U\backslash R$ 表示非叶节点非根节点集合,即网络内部节点集合. $T(k)=(V(k),L(k))$ 表示以节点 k 为根的子树,其叶节点集合为 $R(k)=R\cap V(k)$;

2.2 丢包模型假设

与大多数文献[10~13]一样,本文假设:

(1)在探测周期内,网络拓扑结构不发生变化;

(2)链路丢包过程为相互独立的伯努利(Bernoulli)过程,即同一探测包经过不同的链路的丢包事件相互独立,不同探测包经过同一链路的丢包事件相互独立.

对每一个链路 $k\in L$,用 $\alpha_k(1)\in[0,1]$ 表示探测包从节点 $f(k)$ 经过链路 k 被成功传输的概率,即 $\alpha_k(1)=P\{x_k=1\mid x_{f(k)}=1\}$;$\alpha_k(0)=1-\alpha_k(1)$ 表示探测包经过链路 k 被丢失的概率,即链路丢包率. 定义 $|L|$($|L|$ 为链路个数)维向量 $\boldsymbol{A}=(\alpha_1(0),\alpha_2(0),\cdots,\alpha_{|L|}(0))$ 为全部链路的丢包率,$\boldsymbol{A}$ 即为要通过统计方法推测的参数.

从根节点向叶节点发送 N 个多播探测包,用 x_k^i($k\in V,1\leqslant i\leqslant N$)表示各个节点的接收情况. 当节点 k 成功接收到第 i 个探测包时,$x_k^i=1$,否则 $x_k^i=0$. $X^i=(x_k^i,k\in V)$ 表示第 i 个探测包在所有节点的分布. N 次探测结果用 $\boldsymbol{X}=\{X^i,1\leqslant i\leqslant N\}$ 表示. 设节点 k 收到的探测包个数为 n_k,N 次探测结果的对数似然函数为:

$$\begin{aligned}L(\boldsymbol{X};\boldsymbol{A})&=\lg(P(\boldsymbol{X};\boldsymbol{A}))\\&=\lg\prod_{i=1}^{N}P(X^i;\boldsymbol{A})\\&=\sum_{k\in U}(n_k\lg\alpha_k(1)+(n_{f(k)}-n_k)\lg\alpha_k(0))\end{aligned}\tag{1}$$

通过极大似然估计,可得链路丢包率估计值为:

$$\hat{\alpha}_k(0)=1-n_k/n_{f(k)}\tag{2}$$

2.3 基于 EM 的链路丢包率估计

在每次探测中,仅有叶节点的探测结果为可观测数据(用 $\boldsymbol{Y}=\boldsymbol{X}_R=(x_k;k\in R)$ 表示),而网络内部节点的探测结果是未知的,因此不能使用式(2)直接求解链路的丢包率. 文献[6]提出用 EM 算法求解链路丢包率,主要通过以下两步进行求解:

E(Expectation)步:根据当前的链路丢包率估计值 $\hat{\boldsymbol{A}}^{(l)}$,计算 N 次探测结果的对数似然函数:

$$\begin{aligned}Q(\hat{\boldsymbol{A}};\hat{\boldsymbol{A}}^{(l)})&=E_{\hat{\boldsymbol{A}}^{(l)}}[\lg P(\boldsymbol{X};\hat{\boldsymbol{A}}^{(l)})\mid\boldsymbol{Y}]\\&=\sum_{k\in U}(\hat{n}_k\lg\hat{\alpha}_k^{(l)}(1)+(\hat{n}_{f(k)}-\hat{n}_k)\lg\hat{\alpha}_k^{(l)}(0))\end{aligned}\tag{3}$$

其中

$$\begin{aligned}\hat{n}_k&=\sum_{i=1}^{N}P_{\hat{\boldsymbol{A}}^{(l)}}(x_k^i=1\mid Y^i)\\&=\sum_{y\in\Omega_R}n(y)\sum_{x\in\Lambda_y}1_{\{x_k=1\}}P_{\hat{\boldsymbol{A}}^{(l)}}(x\mid y)\end{aligned}\tag{4}$$

其中 Ω_R 表示所有可能的叶节点探测结果,y 为其中一种叶节点探测结果,$n(y)$ 为该探测结果出现的次数. Λ_y 表示叶节点探测结果 y 对应的所有可能的全部节点探测结果,$P_{\hat{\boldsymbol{A}}^{(l)}}(x\mid y)$ 表示叶节点探测结果为 y 时,网络全部链路的探测结果为 x 的概率.

M(Maximization)步:通过最大化 $Q(\hat{\boldsymbol{A}};\hat{\boldsymbol{A}}^{(l)})$:

$$\hat{\boldsymbol{A}}^{(l+1)}=\arg\max_{\hat{\boldsymbol{A}}}Q(\hat{\boldsymbol{A}};\hat{\boldsymbol{A}}^{(l)})\tag{5}$$

可得链路 k 的丢包率估计值为:

$$\hat{\alpha}_k^{(l+1)}(0)=1-\hat{n}_k/\hat{n}_{f(k)}\tag{6}$$

3 基于子树丢包模式的链路丢包率推断算法

3.1 链路丢包率初始值计算

现有 EM 算法链路丢包率初始值一般任意给定,迭代次数较多. 由于更精确的链路丢包率初始值能减少 EM 算法迭代次数,从而提高算法速度,本文提出一种快速链路丢包率估计算法,利用该算法求得的结果对

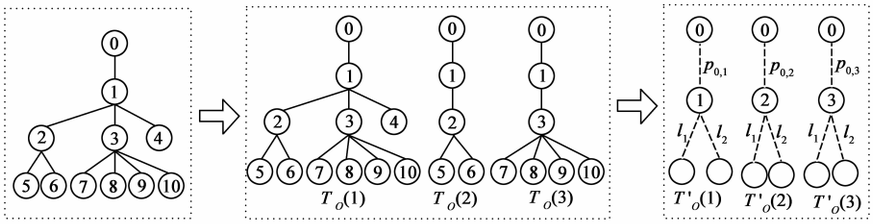
EM 算法链路丢包率进行初始化.

对每一个内部节点 $k(k\in I)$,都存在一个由根节点到节点 k 的路径上所有节点与子树 $T(k)$ 组成的子树,用 $T_O(k)$ 表示. 如图 1(a)所示拓扑,3 个内部节点(节点 1、2、3)对应的 $T_O(1)$、$T_O(2)$ 和 $T_O(3)$ 如图 1(b)所示. 节点 k 的每一个子节点对应子树 $T_O(k)$ 的一个分支. 分支的任一个叶节点接收到探测包,则视为该分支成功接收到探测包. 选取接收到探测包个数最多的两个分支,分别记为 l_1,l_2,将路径 $p_{0,k}$ 和分支 l_1,l_2 均视为虚链路,则构成一个以节点 k 为分叉节点的二叉树,记为 $T'_O(k)$. $T_O(1)$、$T_O(2)$ 和 $T_O(3)$ 对应的二叉树 $T'_O(1)$、$T'_O(2)$ 和 $T'_O(3)$ 如图 1(c)所示. 用 $P_{0,k}$ 表示探测包在公共路径 $p_{0,k}$ 上被成功传输的概率,根据文献[3]中式(14),可得 $P_{0,k}$ 极大似然估计值为:

$$\hat{P}_{0,k}=\frac{n_{r_1}n_{r_2}}{n(x_{r_1}=1,x_{r_2}=1)} \tag{7}$$

其中,n_{r_1} 和 n_{r_2} 分别表示被分支 r_1、r_2 接收到的探测包个数,$n(x_{r_1}=1,x_{r_2}=1)$ 表示同时被分支 r_1、r_2 成功接收到的探测包个数. 对所有的内部节点 k,都可以根据式(7)计算出探测包在路径 $p_{0,k}$ 上的成功传输概率. 对于叶节点 r,由于叶节点探测结果是可以观测到的,故路径 $p_{0,r}$ 上的成功传输概率估计值可以直接求出:

$$\hat{P}_{0,r}=\frac{n_r}{N} \tag{8}$$



(a) 树型拓扑　　(b) $T_O(k)$　　(c) $T'_O(k)$

图1　二叉树构造

用式(7)和(8)求出所有非根节点的路径成功传输概率后,即可根据父子节点的路径成功传输概率关系求出所有链路的丢包率:

$$\hat{\alpha}_k(0)=1-\frac{\hat{P}_{0,k}}{\hat{P}_{0,f(k)}} \tag{9}$$

本文采用式(9)得出的链路丢包率估计值,作为 EM 算法链路丢包率初始值.

3.2　网络拓扑传输状态独立区域划分

由于仅有叶节点的探测结果是可以观测到的,而网络内部节点的探测结果是不能观测到的,故对于一次探测过程,仅有叶节点和成功接收到该探测包的叶节点的祖先节点的探测结果是可以确定的,其他节点的探测结果是不能确定的. 对任意链路 $k=(f(k),k)$,根据叶节点探测结果可以得到该链路对应的父子节点的探测结果有如表 1 所示五种情况. 其中第 1、2 种情况链路传输状态是确定的,第 3、4、5 种情况链路的传输状态是不确定的. 根据链路传输状态是否确定,将整个网络链路分为两个部分,传输状态确定性区域 $A(L;y)$ 和传输状态非确定性区域 $B(L;y)$. 在该次探测过程中,探测包在 $A(L;y)$ 中的所有链路都被成功传输或丢失,而在 $B(L;y)$ 中的所有链路上的传输状态是不确定的. 如图 2 所示拓扑,当叶节点探测结果为 $y=(1,0,1,0,0,0,0,1,0,0,0,0,0)$ 时,探测包在图中实线表示的链路上被成功传输,在虚线表示的链路上被丢失,在点划线表示的链路上探测包的传输状态不能确定. 故 $A(L;y)$ 为图中实线和虚线表示的链路集合,$B(L;y)$ 为图中点划线表示的链路集合.

表 1　节点探测结果及对应链路传输状态

类型	探测结果	链路状态
1	$x_{f(k)}=1,x_k=1$	传输成功
2	$x_{f(k)}=1,x_k=0$	传输失败
3	$x_{f(k)}=1,x_k$ 不确定	不确定
4	$x_{f(k)}$ 不确定,$x_k=0$	不确定
5	$x_{f(k)}$ 不确定,x_k 不确定	不确定

当节点 k 的探测结果不确定时,其全部非叶子孙节点的探测结果都是不确定的,由表 1 可得,当链路 k 的传输状态不确定时,其全部的子孙链路的传输状态都是不确定的. 因此,$B(L;y)$ 中链路可构成一系列没有公共节点的子树. 用 $T_f(k)$ 表示节点 k 的父节点 $f(k)$ 与子树 $T(k)$ 构成的子树,则图 2 中传输状态不确定链路构成子树 $T_f(2)$ 和 $T_f(10)$. 由链路丢包的独立性假设,可得这些没有公共节点的子树传输状态相互独立. 令 $B_k(L;y)=L(T_f(k))$,可将 $B(L;y)$ 进一步划分成一系列传输状态非确定性独立区域 $\{B_k(L;y)\}_k$. 图 2 中 $B(L;y)$ 可以划分成独立区域 $B_2(L;y)$ 和 $B_{10}(L;y)$,如图 3 所示.

将整个网络拓扑划分为传输状态确定性区域 $A(L;y)$ 和一系列传输状态非确定性独立区域

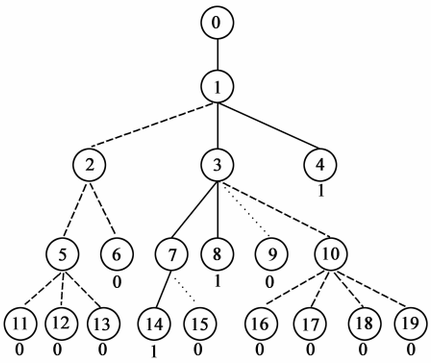


图2 链路区域划分

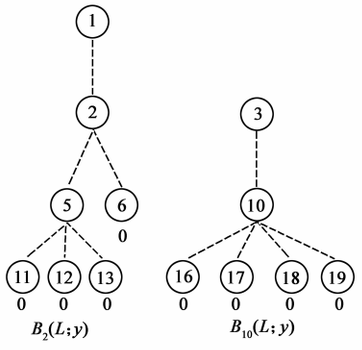


图3 传输状态非确定性区域

$\{B_k(L;y)\}_k$ 后,在利用式(4)计算各个节点收到的探测包个数 n_k 时,即可根据各个节点所属区域分别进行计算. 当链路 k 属于 $A(L;y)$ 时,若节点 k 为内部节点则 $P_{\hat{A}^{(l)}}(x_k=1|y)=1$,若节点 k 为叶节点,由于叶节点探测结果是可以观察到的,故 $P_{\hat{A}^{(l)}}(x_k=1|y)$ 也可以直接根据探测结果给出;当链路 k 属于 $B(L;y)$ 时,$P_{\hat{A}^{(l)}}(x_k=1|y)$ 值不能直接确定. 本文对于属于 $B(L;y)$ 中链路,首先建立一个子树丢包模式数据库,然后使用子树丢包模式数据库计算 $P_{\hat{A}^{(l)}}(x_k=1|y)$,以减少冗余计算,提高算法效率.

3.3 子树丢包模式

由 3.2 节分析可得,每一个独立区域 $B_k(L;y)$ 都构成一个子树 $T_f(k)$,该子树的根节点 $f(k)$ 成功接收到探测包,而全部叶节点 $R(k)$ 均未收到探测包. 对任意节点 k 所对应的子树 $T_f(k)$,从该子树根节点 $f(k)$ 向子树的叶节点发送的多播探测包,其全部叶节点 $R(k)$ 均未收到该探测包的事件用 $\delta_k^{\{0\}}$ 表示,其概率记为 $\rho_k(\delta_k^{\{0\}},\hat{\boldsymbol{A}}^{(l)})$. $\delta_k^{\{0\}}$ 可以视为一个丢包模式,将丢包模式按层次自上而下分解,可以构建一个子树丢包模式数据库 $\{\delta_k^{\{0\}}\}$,从而避免冗余计算. 如图 4 为 $\delta_2^{\{0\}}$ 的分解过程,从上到下经过 2 层分解后,到达叶节点,分解结束.

对叶节点 k,根据 $\delta_k^{\{0\}}$ 定义可得 $\rho_k(\delta_k^{\{0\}},\hat{\boldsymbol{A}}^{(l)})$ 为:

$$\rho_k(\delta_k^{\{0\}},\hat{\boldsymbol{A}}^{(l)})=\hat{\alpha}_k^{(l)}(0),k\in R \tag{10}$$

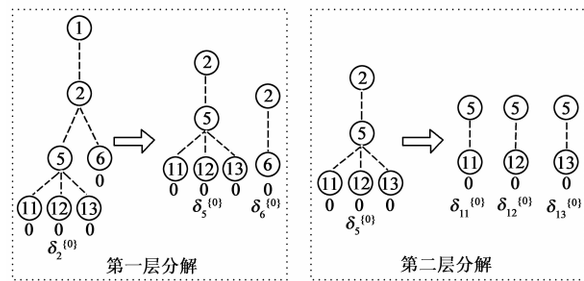


图4 丢包模式分解

设子树 $T(k)$ 的所有叶节点都未收到探测包的事件的概率为 $R_k^{\{0\}}$,即

$$R_k^{\{0\}}=P(x_k=0:k\in R(k)) \tag{11}$$

对非叶节点对应的丢包模式 $\delta_k^{\{0\}}$,其概率 $\rho_k(\delta_k^{\{0\}},\hat{\boldsymbol{A}}^{(l)})$ 与其子节点 $k'\in c(k)$ 对应的丢包模式 $\delta_{k'}^{\{0\}}$ 存在如下关系:

$$\begin{aligned}\rho_k(\delta_k^{\{0\}},\hat{\boldsymbol{A}}^{(l)})&=P(x_k=0)P(R_k^{\{0\}}|x_k=0)\\&\quad+P(x_k=1)P(R_k^{\{0\}}|x_k=1)\\&=\alpha_k^{(l)}(0)+\alpha_k^{(l)}(1)\prod_{k'\in c(k)}\rho_{k'}(\delta_{k'}^{\{0\}},\hat{\boldsymbol{A}}^{(l)})\end{aligned} \tag{12}$$

故对于非叶节点所对应的丢包模式,用式(10)计算出叶节点对应丢包模式的概率后,即可按照式(12)自底向上计算出所有内部节点对应的丢包模式概率.

设从子树 $T_f(k)$ 根节点 $f(k)$ 向该子树的叶节点发送的多播探测包,当所有叶节点均未收到该探测包时,节点 k 成功接收到探测包的概率为 $\gamma_k(\delta_k^{\{0\}},\boldsymbol{A}^{(l)})$,即

$$\begin{aligned}\gamma_k(\delta_k^{\{0\}},\boldsymbol{A}^{(l)})&=P(x_k=1|\delta_k^{\{0\}},\boldsymbol{A}^{(l)})\\&=\begin{cases}0, & k\in R\\ \dfrac{\alpha_k^{(l)}(1)\prod\limits_{k\in c(k)}\rho_{k'}(\delta_k^{\{0\}},\boldsymbol{A}^{(l)})}{\rho_k(\delta_k^{\{0\}},\boldsymbol{A}^{(l)})}, & k\notin R\end{cases}\end{aligned} \tag{13}$$

采用式(10)和(12)计算出所有丢包模式的概率后,带入式(13)即可计算出所有丢包模式对应的 $\gamma_k(\delta_k^{\{0\}},\boldsymbol{A}^{(l)})$.

3.4 链路丢包率计算

划分独立区域,并建立丢包模式数据库后,本文将网络节点分为 3 类:叶节点,属于传输状态确定性区域的内部节点和属于传输状态非确定性区域的内部节点. 在计算 $\hat{n}_k$ 时分别针对三种类型的节点采用不同方法进行计算.

对于叶节点 r,由于叶节点探测结果是可以观测到的,故叶节点接收到的探测包个数:

$$\hat{n}_k = \sum_{i=1}^{N} x_k^i, k \in R \tag{14}$$

令

$$\hat{n}_k(y) = n(y) P_{\hat{\boldsymbol{A}}^{(l)}}(x_k = 1 \mid y) \tag{15}$$

代入式(4)可得

$$\hat{n}_k = \sum_{y \in \Omega_R} \hat{n}_k(y) \tag{16}$$

对于网络内部节点 $k \in I$,若对应链路 $k \in A(L;y)$ 时,$P_{\hat{\boldsymbol{A}}^{(l)}}(x_k = 1 \mid y) = 1$,则有

$$\hat{n}_k(y) = n(y) P_{\hat{\boldsymbol{A}}^{(l)}}(x_k = 1 \mid y) = n(y), k \in I \text{ 且 } k \in A(L;y) \tag{17}$$

对于网络内部节点 $k \in I$,若对应链路 $k \in B(L;y)$ 时,根据 $\rho_k(\delta_k^{\{0\}}, \hat{\boldsymbol{A}}^{(l)})$ 的定义,$\hat{n}_k(y)$ 和 $\hat{n}_{f(k)}(y)$ 存在如下关系:

$$\hat{n}_k(y) = \hat{n}_{f(k)}(y) \gamma_k(\delta_k^{\{0\}}, \boldsymbol{A}^{(l)}), k \in I \text{ 且 } k \in B(L;y) \tag{18}$$

按照式(18)即可从上到下计算出各节点对应的 $\hat{n}_k(y)$.

求出所有的 $\hat{n}_k(y)$ 后,代入式(16)即可得到全部内部节点对应的 $\hat{n}_k$ 值,然后利用式(6)计算第 $(l+1)$ 次迭代结果 $\boldsymbol{A}^{(l+1)}$. 重复整个迭代求解过程,直到迭代终止条件满足,即可得到链路丢包率的估计值.

3.5　算法复杂度分析

由 2.3 节可知,链路丢包率推断算法复杂度主要取决于 $\hat{n}_k$ 求解过程. 现有的 EM 算法先找出 y(叶节点探测结果)对应的所有可能的 x(全部节点探测结果),然后进行计算,其算法复杂度主要体现在计算所有的 x 的概率. 以图 2 所示的网络拓扑为例,当叶节点探测结果为 $y^1 = (1,0,1,0,0,0,0,1,0,0,0,0,0)$,$y^2 = (1,0,1,1,0,0,0,1,0,1,0,1,1)$ 时,对应的所有可能的 x 的个数为 9,每次迭代过程计算所有 y 对应的全部 x 的概率需要的乘法计算次数为 $19 \times 9 = 171$ 次. 而本文提出的基于子树丢包模式的链路丢包率推断算法(Loss Inference Algorithm Based on Subtree Loss Pattern,LIALP)先建立子树丢包模式数据库,以子树模式为基本单元计算 $\hat{n}_k$ 值,其计算复杂度主要取决于建立丢包模式数据库过程中 $\rho_k(\delta_k^{\{0\}}, \hat{\boldsymbol{A}}^{(l)})$ 和 $\gamma_k(\delta_k^{\{0\}}, \boldsymbol{A}^{(l)})$ 的计算. 表 2 给出了每次迭代需要计算的丢包模式及计算相应的 ρ_k 和 γ_k 所需的乘法次数. 从表中可以看出,算法每次迭代需要的乘法次数为 18,与现有的 EM 算法每次迭代需要的乘法计算次数 171 次相比,效率有了很大的提高. 该示例为探测包个数为 2 时的情况,随着探测包个数的增加,这种优势会越来越明显. 另外,利用 3.1 节提出的链路丢包率快速估计算法为 EM 算法进行初始化能有效地减少迭代次数,进一步提高算法效率.

表 2　丢包模式及计算相应变量所需乘法次数

丢包模式	ρ_k	γ_k
$\delta_2^{\{0\}}$	2	2
$\delta_5^{\{0\}}$	3	3
$\delta_6^{\{0\}}$	0	0
$\delta_{10}^{\{0\}}$	4	4
$\delta_{11}^{\{0\}}$	0	0
$\delta_{12}^{\{0\}}$	0	0
$\delta_{13}^{\{0\}}$	0	0
$\delta_{16}^{\{0\}}$	0	0
$\delta_{17}^{\{0\}}$	0	0
$\delta_{18}^{\{0\}}$	0	0
$\delta_{19}^{\{0\}}$	0	0

4　仿真

为对本文提出 LIALP 算法性能进行验证,本文从链路丢包率测量精度与速度两个方面对算法进行考察,并与现有 EM 算法[6]和 FPA 算法[13]进行比较.

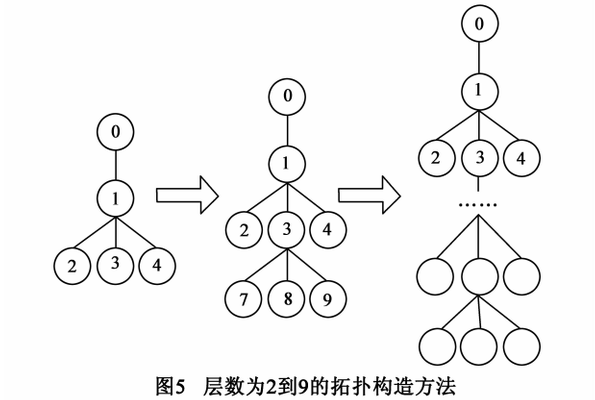


图5　层数为2到9的拓扑构造方法

4.1　MATLAB 模型仿真

为考察算法计算时间和误差随拓扑层数的变化,本文按图 5 所示方法构造每层节点数目为 3,拓扑层数分别为 2 至 9 的 8 个拓扑. 所有拓扑的链路丢包率服从均匀分布:$\alpha_k(0) \sim \text{Uniform}[0, 0.2]$. 对每个拓扑,均产生 100 个丢包率组合,重复进行 100 次实验,探测包个数为 500. 图 6 – 7 分别给出了本文提出的 LIALP 算法、EM 算法和 FPA 算法链路丢包率测量计算时间和相对误差随网络拓扑层数的变化. 从图 6 可以看出,本文提出的 LIALP 算法链路丢包率测量计算时间较现有 EM 算法有明显降低,而且随着拓扑层数的增加,这种优势更加明显,当网络拓扑层数为 8 时,LIALP 算法计算时间(0.1182s)仅为 EM 算法计算时间(9.3563s)的 1.26%. FPA 算法利用子树丢包之间的相关性直接求

解链路丢包率,不需要迭代计算,故算法计算速度最快.但由于 FPA 算法在求解的过程中,将网络拓扑拆分成单个子树后进行计算,忽略了部分全局信息,使得算法精度有所降低.如图 7 所示,当网络拓扑层数为 8 时,FPA 算法精度较 LIALP 算法降低 19.5%.

为考察在网络不同拥塞程度下算法精度的变化,本文采用图 2 所示拓扑,所有链路丢包率服从均匀分布:$\alpha_k(0) \sim \text{Uniform}[\alpha_{ave}-0.02, \alpha_{ave}+0.02]$,其中 α_{ave} 为丢包率均值,$\alpha_{ave}=0.05, 0.10, \cdots, 0.30$.探测包个数为 500,重复 100 次试验.图 8 给出了链路丢包率测量相对误差随链路丢包率均值的变化情况.从图可以看出,LIALP 算法和 EM 算法测量相对误差明显小于 FPA 算法,三种算法测量相对误差都随链路丢包率均值的增大而增大,且 FPA 算法增大的更快.

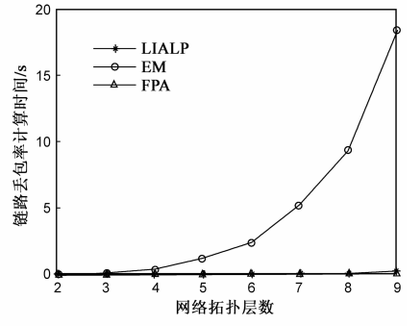


图6 链路丢包率计算时间随网络拓扑层数的变化

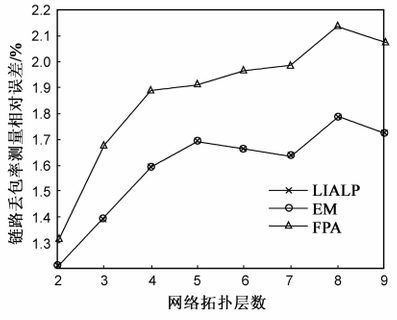


图7 链路丢包测量相对误差随网络拓扑层数的变化

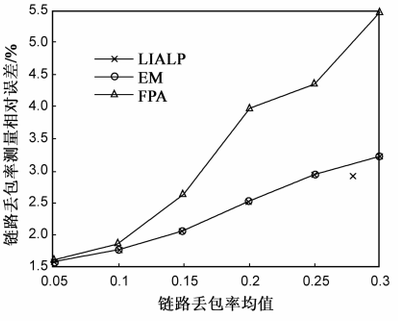


图8 链路丢包测量相对误差随链路丢包率均值的变化

4.2 NS2 仿真

增加探测包个数即增加采样点数,能减小链路丢包率测量误差,但是增加探测包个数会导致探测时间增加,从而导致测量总时间增加.本文采用 NS2 仿真进一步比较算法精度随探测包个数的变化,以及在相同精度条件下链路丢包率测量总时间.采用 NS2 构建如图 2 所示的网络.该网络包含 20 个节点和 19 条链路.所有链路的带宽均为 10Mbps,每条链路的物理传播时延为 10ms,队列长度为 50,排队模型为 FIFO(First In First Out),拥塞避免算法采用尾部丢弃(Drop-tail).仿真背景流由两部分组成:(1)服从 Pareto 分布的 On/Off 模型 120 条 UDP 流和 120 条 TCP 流.发送节点和接收节点在所有节点中随机选择.(2)叠加在每条链路上的服从 Pareto 分布的 On/Off 模型 UDP 流和 TCP 流.每条链路上的 UDP 流和 TCP 流个数均为 15 至 30 之间的随机整数.UDP 流和 TCP 流的 On 期和 Off 期均为 200ms,UDP 流的速率为 0.5Mbps,TCP 流的速率为 0.2Mbps.探测包为从根节点向全部叶节点发送的恒定速率多播包,其大小为 50Byte,发送速率为 0.2Mbps.

表 3 和图 9 分别给出了探测包个数为 500、1000、…、10000 时,三种算法链路丢包率测量计算时间和相对误差的变化情况.从中可以看出,三种算法计算时间都随探测包个数的增加而增大,相对误差都随着探测包的个数的增加而降低.当探测包个数相同时,LIALP 算法和 EM 算法精度最高,FPA 算法计算时间最短,但误差最大,故在相同测量精度条件下,FPA 算法需要发送的探测包最多.所需探测包个数越多,则探测时间越长,从而导致链路丢包率测量总时间越长.表 4 给出了当相对误差为 1%时,三种算法所需探测包个数、链路丢包率计算时间和测量总时间.从表中可以看出,在误差相同的条件下,FPA 算法的链路丢包率计算时间(0.13s)最短,但需要的探测包个数(5000)较 LIALP 算法和 EM 算法(3300)多 51.5%.LIALP 算法链路丢包率测量速度最快,其测量总时间(7.23s)仅为现有 EM 算法(103.95s)的 6.96%,FPA 算法(10.13s)的 71.37%.

表 3 三种算法计算时间(s)随探测包个数的变化

探测包个数	LIALP	EM	FPA
500	0.21	31.84	0.01
1000	0.32	50.00	0.03
1500	0.41	60.36	0.04
2000	0.49	71.57	0.05
2500	0.55	82.28	0.07
3000	0.60	91.21	0.08
3500	0.65	101.44	0.09
4000	0.71	109.19	0.11
4500	0.75	116.93	0.12
5000	0.78	124.05	0.13
5500	0.81	129.02	0.14
6000	0.86	133.78	0.16
6500	0.89	139.27	0.17
7000	0.92	144.86	0.19
7500	0.95	149.75	0.20
8000	0.97	154.72	0.21
8500	1.00	159.67	0.22
9000	1.02	163.80	0.24
9500	1.05	168.60	0.25
10000	1.18	173.03	0.26

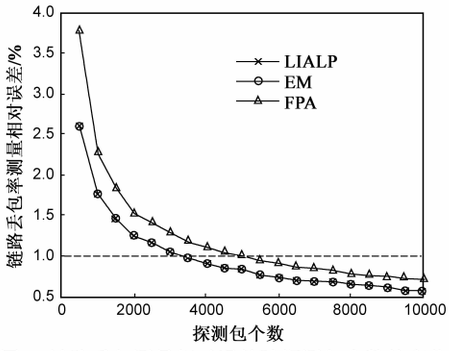


图9　链路丢包测量相对误差随探测包个数的变化

表 4　三种算法探测包个数、计算时间和测量总时间

算法	探测包个数	计算时间(s)	测量总时间(s)
LIALP	≈3300	≈0.63	≈7.23
EM	≈3300	≈97.35	≈103.95
FPA	≈5000	≈0.13	≈10.13

5　结论

本文根据网络链路丢包的特性,提出了一种基于子树丢包模式的链路丢包率推断算法. 该算法能在保持现有 EM 算法较高精度的条件下,有效减少链路丢包率测量总时间,提高测量速度. 在相同测量精度(相对误差为 1%)下,该算法测量速度较现有 EM 算法提高约 13.4 倍,较 FPA 算法提高 40%.

参考文献

[1] Coates A, Hero III A O, Nowak R, et al. Internet tomography[J]. IEEE Signal Processing Magazine, 2002, 19(3):47 – 65.

[2] 张润生,李艳斌,李啸天. 基于合并分层聚类的网络拓扑推断算法[J]. 电子学报,2013,41(12):2346 – 2352.
Zhang Run-sheng, Li Yan-bin, Li Xiao-tian. Agglomeration hierarchical clustering based algorithm for network topology inference[J]. Acta Electronica Sinica, 2013, 41(12): 2346 – 2352. (in Chinese)

[3] Cáceres R, Duffield N G, Horowitz J, et al. Multicast-based inference of network-internal loss characteristics[J]. IEEE Transactions on Information Theory, 1999, 45(7): 2462 – 2480.

[4] Fei G, Hu G. Accurate and effective inference of network link loss from unicast end-to-end measurements[J]. IET Communications, 2012, 6(17): 2989 – 2997.

[5] Zhang Zhi-yong, Fei Gao-lei, Pan Shenli. Afast link delay distribution inference approach under a variable bin size model[J]. IEICE Transactions on Communications, 2013, 96(2):504 – 507.

[6] Bu T, Duffield N, Presti F L, et al. Network tomography on general topologies[A]. ACM SIGMETRICS Performance Evaluation Review[C]. New York: ACM, 2002. 21 – 30.

[7] Nguyen H X, Thiran P. Network loss inference with second order statistics of end-to-end flows[A]. Proceedings of ACM SIGCOMM'07[C]. New York: ACM, 2007. 227 – 240.

[8] Qiao Yan, Qiu Xue-song, Meng Luo-ming, et al. Efficient loss inference algorithm using unicast end-to-end measurements[J]. Journal of Network and Systems Management, 2013, 21(2): 169 – 193.

[9] 顾然,邱雪松,乔焰,等. 基于非线性规划的链路丢包率推理算法[J]. 电子与信息学报,2012,34(6):1425 – 1431.
Gu Ran, Qiu Xue-song, Qiao Yan, et al. Link loss inference algorithm with nonlinear programming[J]. Journal of Electronics & Information Technology, 2012, 34(6): 1425 – 1431. (in Chinese)

[10] Zhu W, Geng Z. A bottom-up inference of loss rate[J]. Computer Communications, 2005, 28(4): 351 – 365.

[11] Li Y J, Cai W D. Afast multicast-based approach to inferring loss performance[J]. Journal of Communication and Computer, 2006, 3(3): 19 – 24.

[12] Zhang Jian-zhong, Lin Wen, Lin Jun-wu. Multicast-based inference of network internal loss from end-to-end data[A]. IEEE 9th International Conference on Signal Processing[C]. Beijing: IEEE, 2008. 2045 – 2048.

[13] Su H, Li Y, Lin S, et al. Inference of link loss rates by explicit estimation[J]. IET Communications, 2010, 4(5): 540 – 550.

作者简介

尹文涛　男,1983 年生于湖北孝感,哈尔滨工业大学自动化测试与控制系博士研究生. 主要研究方向为网络测量与网络层析成像技术.
E-mail: huayichu@163com

杨京礼　男,1984 年生于山东日照,哈尔滨工业大学自动化测试与控制系讲师. 主要研究方向为网络测量与网络层析成像技术.
E-mail: icehit0615@163com

姜守达(通信作者)　男,1964 年出生黑龙江伊春,哈尔滨工业大学自动化测试与控制系教授. 主要研究方向为虚拟试验技术,网络测量技术,数字信号处理等.
E-mail: jsd@hit.edu.cn

魏长安　男,1981 年生于河北承德,哈尔滨工业大学自动化测试与控制系讲师. 主要研究方向为虚拟试验技术,自动测试技术等.