ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.9, September 2009, pp.2520−2530
doi: 10.3724/SP.J.1001.2009.03448

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

一种动态环境下的互联网服务故障诊断算法*

褚灵伟[1+], 邹仕洪[1], 程时端[1], 田春岐[2], 王文东[1]

[1](北京邮电大学 网络与交换国家重点实验室,北京 100876)

[2](同济大学 电子与信息工程学院 计算机科学与技术系,上海 200092)

Efficient Fault Diagnosis Algorithm in Dynamic Internet Service Environment

CHU Ling-Wei[1+], ZOU Shi-Hong[1], CHENG Shi-Duan[1], TIAN Chun-Qi[2], WANG Wen-Dong[1]

[1](State Key Laboratory of Networking and Switching, Beijing University of Posts and Telecommunications, Beijing 100876, China)

[2](Department of Computer Science and Technology, College of Electronic and Information Engineering, Tongji University, Shanghai 200092, China)

+ Corresponding author: E-mail: rue2004cn@sohu.com, http://bnrc.cs.bupt.cn

Chu LW, Zou SH, Cheng SD, Tian CQ, Wang WD. Efficient fault diagnosis algorithm in dynamic Internet service environment. *Journal of Software*, 2009,20(9):2520−2530. http://www.jos.org.cn/1000-9825/3448.htm

Abstract: Dynamic changes in service environment will affect fault diagnosis algorithm. In order to reduce the impact, challenges of fault diagnosis in dynamic environment are analyzed in this paper. Multi-layer management model is presented to model the service system, Bipartite Bayesian network is chosen to model the dependency relationship and binary symmetric channel is chosen to model noises. To deal with the dynamic fault set caused by fault recovery mechanism, prior fault probability is modified based on fault persistent time statistic; To deal with the dynamic model, expected model is built based on the time of observing symptoms and original models in current window. Simulation results show that this fault diagnosis algorithm is efficient in dynamic Internet service environment.

Key words: dynamic system; fault management; fault diagnosis; dependency model; bipartite Bayesian network

摘 要: 服务环境中的动态性会对故障诊断算法性能造成影响.为了降低这种影响,分析了服务环境中的动态性,提出多层管理模型建模服务系统:二分贝叶斯网络建立依赖模型和二元对称信道建模噪声.针对故障自动修复机制导致的动态故障集环境,在故障持续时间统计的基础上修正当前窗口内先验故障概率;针对动态模型环境,基于当前窗口内原始模型和观察症状时间建立期望模型.仿真结果显示,算法可以有效地诊断动态环境下的互联网服务故障.

关键词: 动态系统;故障管理;故障诊断;依赖模型;二分贝叶斯网络

中图法分类号: TP393 **文献标识码**: A

* Supported by the National Natural Science Foundation of China under Grant Nos.60603060, 60502037, 90604019 (国家自然科学基金); the Program for New Century Excellent Talents in University of China under Grant No.NECT-08-0739 (新世纪优秀人才支持计划); the National High-Tech Research and Development Plan of China under Grant Nos.2007AA12Z321, 2007AA01Z206 (国家高技术研究发展计划(863))

Received 2008-06-23; Accepted 2008-08-21

近年来,服务提供商(service providers,简称SP)意识到在Internet上提供增值服务能够带来潜在的高额利润,并开发了各种各样的Internet服务,如Web服务,IPTV,VoIP等.为了维持当前客户并吸引新客户,SPs必须为服务提供QoS(quality of service)保证.但是服务相关设备,网络和服务本身都可能发生异常,从而导致服务QoS降级,甚至服务不可用.显然,性能降级或服务不可用都将影响SP的信誉.为了保证QoS,SPs迫切需要一种有效的服务故障管理机制来监测故障发生,分析故障原因,并尽快采取修复措施.

故障管理包括故障监测、故障诊断、故障修复.故障监测的目标是及时发现问题组件导致的外部现象,也称为告警或者症状,并将症状通知管理系统.发生问题的组件称为故障(fault),异常监测结果称为负症状(negative symptom),正常监测结果称为正症状(positive symptom).故障诊断的目标是根据观察到的症状集快速、准确地推理得出故障假设.现有的故障诊断技术大体可以分为两个类别:以专家系统(如基于规则的系统[1]和基于案例的系统[2])为代表的确定性故障诊断;另一类是非确定性故障诊断,或称为概率性故障诊断.故障修复的任务是尽快修复诊断得出的故障假设,恢复服务的正常运行.

确定性依赖模型[3−5]在故障管理中被广泛采用,这种模型假设当一个故障发生时,所有与该故障相关的症状都会发生.近年来,研究者逐渐认识到故障和症状之间的联系是不确定或者概率的,故障以一定概率引起存在依赖关系的症状异常[6,7].复制组件(replicated components)、不精确门限等都可能导致不确定性.本文在不确定性模型的基础上展开工作.图论、神经网络和不确定推理可以用来描述不确定性.但是,神经网络需要长时间训练;不确定推理存在严格假设;图论技术相对更适用于概率故障诊断.为了简化建模过程,降低诊断复杂度,本文采用二分贝叶斯网络建模概率依赖.

观察到的症状集合不仅由故障集决定,而且受到其他因素的影响[8].例如,由于不可靠通信机制,症状可能在向管理系统的传输过程中被改变.故障发生导致的负症状可能会在到达管理系统前丢失,称为症状丢失(symptom loss);而正症状可能会被虚假症状(spurious symptom)干扰.本文为每个症状构造二元对称信道(binary symmetric channel)来建模上述噪声.

传统的故障诊断算法一般假设故障环境是静态的,基于静态模型诊断得出故障假设.然而,很多实际场景是动态变化的.举例来说,动态服务网络(dynamic service network)[9]中服务的加入与删除;故障修复机制[10]自动修复故障组件.在服务层以外的其他层上同样存在着动态性,如移动ad-hoc网中节点移动导致依赖路径变化[11].上述动态性可能导致一个时间窗口内依赖模型或者发生故障集的变化,传统故障诊断算法并没有对动态环境建模的机制,将受到动态性的影响而导致性能下降.

为了保证在动态环境下的诊断算法性能,本文综合考虑动态故障集和动态模型,提出了动态环境下的诊断算法DGA(greedy algorithm in dynamic environment).该算法针对动态故障集环境,在故障持续时间统计(fault persistent time statistic)的基础上修正当前窗口内先验故障概率;针对动态模型环境,基于当前窗口内原始模型和症状时间建立期望模型(expected model).仿真结果显示,DGA算法在动态环境下获得了较好的诊断性能.

本文第1节介绍相关工作.第2节分析服务场景,建模依赖关系和噪声.第3节给出动态环境下的诊断算法.第4节给出仿真过程和实验结果分析.第5节总结全文.

1 相关工作

动态贝叶斯网络(dynamic Bayesian network,简称DBN)可用于建模故障自动修复导致的发生故障集变化[12−14],由模型中节点集上一时间点的状态推理当前时间点节点先验概率.但是DBN基于马尔可夫假设,即当前节点集先验概率只与前一时间点节点集状态相关,丢失了故障发生时间到当前时间点的时间长度信息.另外,构造DBN需要对历史数据的学习,由于服务环境本身是动态的,节点可能频繁加入和离开,依赖关系和强度也会动态变化,因此,DBN的精确性很难得到保证.Rish等人还提出了sequential multifaults算法[12]处理故障集动态性,假设每个时间窗口内系统只发生一个新故障或者一次修复行为.显然,实际中很难定义窗口大小来满足假设.

模型动态变化会导致传统故障诊断算法性能的下降.Natu和Sethi提出了AFL算法[11]处理动态模型环境

下的故障诊断问题,该算法定时更新依赖模型,并根据窗口内症状集调整前后端模型,分别利用前后模型诊断出两个假设,并综合得出最终假设.Kompella 等人提出,如果在某时间内发生多个拓扑变化,则使用每个拓扑产生一个假设,从假设集中选择故障组成最终假设[15].上述两种方案的共同之处在于都使用多个模型分别诊断出假设,再根据多个假设得出最终假设.但是,多个诊断过程会导致较大的计算开销.Nguyen 和 Thiran 提出将时间窗口划分为槽(slot),假设槽足够小使槽内路由不变,将一个时间窗口内的所有路由树聚集为森林,建立确定性依赖矩阵[6].然而,实际中很难确定槽的大小,过小的槽会带来很大的模型更新开销,过大的槽又难于保证槽内路由不变.Ding 等人提出使用 DBN 建模动态 IP 网络环境[14],但是 DBN 只能建模节点概率和条件概率的变化,不能建模节点集和边集的变化.

2 服务建模

一般来说,一个服务场景由 3 部分组成:服务,网络和用户(用户也可能是调用服务的其他应用程序或者服务).场景中的 3 个部分都可能发生故障,从而导致服务无效或者性能降级.服务或者网络故障可能导致大范围的服务使用异常,而单个用户造成的影响较小,因此,本文关注服务和网络故障.

一个服务场景可能包括多个组织或者公司.他们可能有各自不同的故障管理策略和管理信息格式.试图在单一管理点上管理所有组件是不合理的.另外,子系统内部的信息安全需求也不允许外部系统管理子系统内部组件.为了满足不同的管理需求,需要对服务场景分层[16],从不同粒度和视角建模服务场景.

一个服务子系统中包含多个组件,每个组件都存在多个状态.当组件发生故障时,会影响与它存在依赖关系的服务.为了表示组件和服务之间的依赖关系,本文提出一个分层管理模型,如图 1 所示.

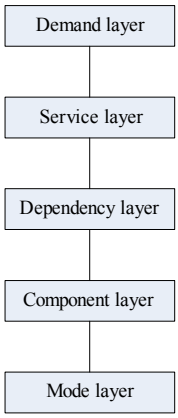

Fig.1 Multi-Layer management model
图 1 多层管理模型

需求层:该层表示对每个服务的 QoS 承诺.SP 一般从服务延时、服务不可用时间等方面提供 QoS 承诺,对不同用户提供的 QoS 可能不同.通过检查服务运行状况可以判断 SLA 是否违约,并向管理系统发送相应症状.

服务层:该层表示系统中实际存在的服务.服务与需求存在一对多的映射关系.如果根据需求发现症状,可以将每个症状映射到一个服务,表示该服务出现异常.同理,服务出现异常,必然可以映射到一个或者多个需求异常.

依赖层:该层描述上下层之间的依赖关系.本文关注基于图论技术的依赖表示方式,称为依赖图.图中节点表示服务和组件,有向边表示依赖关系,边上权重值表示依赖强度.依赖图可以扩展为描述故障-症状依赖关系的故障传播模型,具体方法如下:将每个组件扩展为多个组件模式,每个服务扩展为多个可能症状,每个依赖强度值扩展为依赖矩阵.

组件层:该层表示服务正常运行所依赖的组件,每个组件都是系统中的实际元素.一个服务一般依赖多个组件,不同服务可能依赖相同组件.

模式层:每个组件存在多个状态,也称为模式.其中一个为正常工作模式,其余表示性能下降的不同等级.假设系统 S 中的组件集为 $\{f_i,1\le i\le n\}$,组件 f_i 存在 $m_i\ge 2$ 个模式,那么全局模式空间大小为 $\prod m_i$.如果将组件模式简化为故障和正常工作,则 $m_i=2$,全局模式空间大小为 2^n.在所有可能状态的空间中,只有一个为正常工作状态,其余表示性能下降或者故障状态.

上述管理模型的核心是依赖层的依赖图.根据依赖关系,可以将依赖模型建立为二分贝叶斯网络.依赖模型的下层顶点表示组件集 $F=\{f_i,1\le i\le n\}$,$f_i=1$ 表示组件发生故障,$f_i=0$ 表示组件正常.每个组件节点存在一个先验故障概率 $P(f_i)$,该值由专家分配或者根据历史统计值评估[16].上层顶点表示服务集 $S=\{s_j,1\le j\le m\}$,$s_j=1$ 表示负症状,$s_j=0$ 表示正症状.与组件 f_i 存在因果关系的服务节点集表示为 $Child(f_i)$,与服务 s_j 存在因果关系的组件节点集表示为 $Par(s_j)$.有向边表示因果联系,每条有向边关联一个表示因果强度的条件概率 $P(s_j=1|f_i=1)$,简写为 $P(s_j|f_i)$,该值可以通过外部观察、故障注入技术[9,17]或者服务依赖发现工具[18]获得.本文中假设服务异常的可能原因相

互独立并且以逻辑运算符号 OR 结合,即 noisy-OR 模型[19].

以端到端服务(end-to-end service)[7]为例表示依赖图模型,其他服务可用类似方式建模.端到端服务表示两个节点之间的通信,由于端到端服务可能跨越多个域,需要采用多域故障管理框架 MDFM(multi-domain fault management framework)[20]将管理任务从多个域分解到特定域内,为每个域建立依赖模型.图 2 显示了如何基于拓扑和路由信息为域 AS1 建立依赖模型.本场景假设路径通信中断为症状、连接或者节点发生故障.

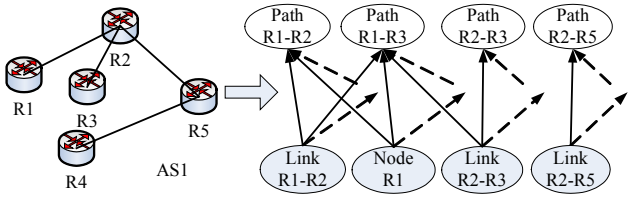

Fig.2 Build dependency model in IP layer

图 2 建立 IP 层依赖模型

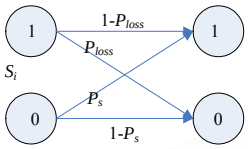

Fig.3 Binary symmetric channel

图 3 二元对称信道

监测发现症状之后,应当将症状发送到管理系统.但是,由于不可靠的通信机制(如 SNMP 协议),症状经常受到丢失和虚假症状的干扰[8].同样,其他因素也会产生干扰.举例来说,过于宽松的门限值会导致症状丢失;间歇故障和过于严格的门限会导致虚假症状.本文为每个症状构建二元对称信道来建模噪声[21].图 3 显示了一个二元对称信道例子,其中 P_{loss} 代表症状丢失概率,P_s 代表虚假症状概率.通过对历史统计数据,包丢失率或系统基线工具(system baseline tool)相关信度测量的分析可以评估这些概率值[8].

3 动态环境下的故障诊断算法

首先介绍故障诊断问题,提出故障诊断的基本算法.然后,分别介绍动态故障集和动态模型,提出相应的应对机制.最后综合上述动态性,提出动态环境下的故障诊断算法.

3.1 基本诊断算法

在观察到当前时间窗口内症状后,需要分析症状并推理得出故障假设.故障诊断的任务是找出最大概率故障子集 $H\subseteq F$,使得探针输出为当前观察到的症状集.形式化如下:

输入:

a) 依赖模型 $BG=(F,S,P_F,P_{F,S})$.F 表示故障集,S 表示症状集,P_F 表示故障先验概率,$P_{F,S}$ 表示依赖强度;

b) 症状丢失率 $P_{loss}(s)$;

c) 症状虚假率 $P_s(s)$;

d) 在某个时间窗口内观察到的症状信息:负症状集 $S_N\subseteq S_O$,正症状集 $S_P\subseteq S_O$.S_O 为可观察的症状集合(由系统的可管程度决定,即系统中嵌入的管理代码量).

假设:

a) Noisy-OR 模型,即引起同一症状的各个故障是相互独立的,并使用逻辑操作符 OR 连接;

b) 多个故障同时发生的概率比单个故障发生的概率低(这是故障诊断算法的常用假设,由于故障假设结果通常还需要进一步测试以确定确切的故障,为节省测试成本,故障假设中包含的故障数目越少越好).

输出:最可能故障假设 H,具有以下性质:

a) 每个负症状可被故障假设 H 中的至少一个故障解释;

b) 故障假设所包含的故障数量$|H|$越小越好.

故障诊断的任务是获得最大概率解释(most probable explanation,简称 MPE),公式化为

$$\arg\max_{H\subseteq F} P(H \mid S_P, S_N) \tag{1}$$

其中,$P(H|S_P,S_N)$表示在给定观察 S_P 和 S_N 的条件下,H 为当前发生故障集合的概率.

根据贝叶斯定理,

$$P(H|S_P,S_N)=P(S_P,S_N|H)\times P(H)/P(S_P,S_N) \tag{2}$$

由于 $P(S_P,S_N)$与选择的 H 无关,可以将其去除,简化为

$$\arg\max_{H\subseteq F} P(S_P,S_N \mid H)\times P(H)=\arg\max_{H\subseteq F} P(S_P \mid H)\times P(S_N \mid H)\times P(H) \tag{3}$$

其中,$P(H)$表示 H 为发生故障子集的概率,$P(S_P|H)$和 $P(S_N|H)$表示故障集为 H 时观察到当前症状结果的概率.设故障 f 的先验故障概率为 $P(f)$,当前假设中该故障的值 $f\in\{0,1\}$.

$$P(H)=\prod_{f\in H} P(f)^f\times(1-P(f))^{(1-f)} \tag{4}$$

$$P(S_P|H)=\prod_{s\in S_P} P(s=0\mid H)=\prod_{s\in S_P}(1-P_s(s))\times x+P_{loss}(s)\times(1-x) \tag{5}$$

$$P(S_N|H)=\prod_{s\in S_N} P(s=1\mid H)=\prod_{s\in S_N} P_s(s)\times x+(1-P_{loss}(s))\times(1-x) \tag{6}$$

其中,$x=\prod_{f\in H}(1-P(s\mid f))$,表示故障假设 H 未导致 s 发生异常的概率.

贝叶斯网络中的精确推理是 NP-hard 问题,即使在简化的二分贝叶斯网络中,该问题依然是 NP-hard 的.为了减小推理复杂度,可以采用贪婪算法来推理假设.其主要思想是在每次循环中,选择最可能故障(故障信度值最大)加入假设,并删除该故障的对应症状.由于负症状表示有故障发生,当负症状集为空时,可认为所有故障已加入假设,这时算法停止.

无噪声环境下,对每个故障的信度评估如下.以 S_f 表示 f 的后代节点中观察到的节点集合.

$$P(f\mid S_N,S_P)=P(f\mid S_f)=P(f,S_f)/P(S_f)=P(f)\times P(S_f\mid f)/P(S_f)=P(f)\times\prod_{s\in S_f} P(s\mid f)^s\times(1-P(s\mid f))^{1-s}/P(s) \tag{7}$$

初始化故障信度 $b(f)$为 $P(f)$,假设当前接收到症状 s,$b_{old}(f)$和 $b_{new}(f)$分别表示 f 更新前和更新后的信度值.可将式(7)改写为

$$b_{new}(f)=b_{old}(f)\times(1-P(s\mid f))/P(s=0),\quad \text{if } s=0 \tag{8}$$

$$b_{new}(f)=b_{old}(f)\times P(s\mid f)/P(s=1),\quad \text{if } s=1 \tag{9}$$

$$P(s=0)=\sum_{F'\subseteq Par(s)}\prod_{f\subseteq F'} P(f)\times(1-P(s\mid f))\prod_{f\subseteq Par(s)-F'}(1-P(f)) \tag{10}$$

$P(s)$为症状先验概率,如果选择 F'的全部组合,则其计算复杂度为 $o(2^{|Par(s)|})$.为了减小复杂度,可以从中选择部分高概率组合使之覆盖较大的概率空间即可.

存在噪声环境下,将公式(8)~公式(10)修改如下即可得到故障信度:

$$b_{new}(f)=b_{old}(f)\times(P(s\mid f)\cdot P_{loss}(s)+(1-P(s\mid f))\times(1-P_s(s)))/P'(s=0),\quad \text{if } s=0 \tag{11}$$

$$b_{new}(f)=b_{old}(f)\times(P(s\mid f)\times(1-P_{loss}(s))+(1-P(s\mid f))\times P_s(s))/P'(s=1),\quad \text{if } s=1 \tag{12}$$

$$P'(s=0)=P(s=0)\times(1-P_s(s))+P(s=1)\times P_{loss}(s) \tag{13}$$

3.2 动态故障集及其应对机制

根据故障持续时间可以将故障分为 3 类:瞬态故障,间歇故障,持续故障.其中,瞬态故障和间歇故障持续时间较短,对系统造成的性能影响较小;而持续故障一直存在系统中,直到相应的人工或者自动修复机制[10]修复该故障.本文关注对性能影响较大的持续故障,简称为故障.由于实际环境中故障修复机制的存在,故障会在发生一段时间 t 后被修复,t 受到故障管理机制、管理系统负载和故障模式的影响会有所变化.在未修复时间$(0,t)$内,故障将持续出现在时间窗口中,导致当前窗口发生的故障会以较高概率出现在之后窗口.现有的故障诊断算法一般假设系统状态在诊断期间不改变,认为故障先验概率是固定的,缺少对发生故障集动态变化的考虑.

为了提高动态环境下的故障诊断算法性能,DGA 算法统计每个故障的持续时间,对故障修复行为建模,从而修正每个时间窗口内的先验故障概率,提高故障诊断性能.对存在大量历史记录的系统,分析历史记录即可以统计出每个故障的持续时间;对不存在历史数据的系统,我们利用故障注入方法[22]主动向系统注入故障,统计出

故障的持续时间.假设故障 f 发生 n 次,在时间 t 内被修复的次数为 $m(0\le m\le n)$,则 f 在 t 时间内未被修复的概率 $P_f(t)=1-m/n$.从 $P_f(t)$定义可知,$\lim_{t\to 0}P_f(t)=1$,$\lim_{t\to\infty}P_f(t)=0$.根据已有故障持续统计值,可以使用时间序列建模 $P_f(t)$,限于篇幅,本文不描述该建模过程.

在基于主动探针[5,12]的故障管理中,监测探针一般于固定时刻被发送出去.假设监测探针发送时间间隔为 Δt,不考虑探针时延及诊断时延,可认为故障假设之间的时间间隔同样为Δt.若 t_{i-1} 时间得出的假设不包含故障 f,而 $t_i=t_{i-1}+\Delta t$ 时间得出的假设包含 f,则 f 发生时间应在(t_{i-1},t_i)内.若 t_i 至 $t_i+k\times\Delta t$ 得出的假设均包含 f,则到 $t_i+(k+1)\times\Delta t$ 时,f 已持续时间在范围$((k+1)\times\Delta t,(k+2)\times\Delta t)$内,$f$ 的先验故障概率应为该时间段内 $P_f(t)$的数学期望 $\int_{(k+1)\times\Delta\tau}^{(k+2)\times\Delta t}P_f(t)\mathrm{d}t\Big/\Delta t$.

在基于被动监测机制[19,23]及时间窗口的故障管理中,上报的症状一般出现在时间窗口的任意位置.假设时间窗口长度为Δt,若 $i-1$ 窗口得出的假设不包含故障 f,而 i 窗口得出的假设包含 f,则认为 f 发生时间应在 i 窗口内.以 i 窗口前端为时刻 0,设 f 发生时刻为 $t(t<\Delta t)$,若 i 至 $i+k$ 窗口假设均包含 f,则到 $i+k+1$ 窗口内任意时间点 y 时,f 已持续时间为 $y-t$,f 的先验故障概率应为 $P_f(t,y)=P_f(y-t)$.在 $i+k+1$ 窗口内,y 的下限为 $y1(t)=(k+1)\times\Delta t-t$,$y$ 的上限为 $y2(t)=(k+2)\times\Delta t-t$,则 f 在 $i+k+1$ 窗口内的先验故障概率为 $\int_0^{\Delta t}\int_{y_1(t)}^{y_2(t)}P_f(y-t)\mathrm{d}y\mathrm{d}t\Big/\Delta t^2$.

为了表示每个故障的已持续时间,DGA 为每个故障 f 维持数值 C_f,表示 f 已连续出现在假设中的次数.根据上一假设中故障 f 对应的 C_f,即可得出当前 f 已持续时间和先验故障概率.将修正的先验故障概率代入模型,并使用基本故障诊断算法即可诊断得出故障假设.

3.3 动态模型及其应对机制

实际服务系统中,不仅存在上述故障集动态性,还存在模型动态性.许多研究[9,11,24]指出,依赖信息一般是不准确或者不完全的,需要不断更新依赖信息.另外,服务组件的动态加入/离开/移动,动态负载均衡等都会导致难于预测的模型动态变化.传统故障诊断算法一般假设已知依赖模型的先验知识.但是在动态环境中,不能完全满足这个假设,为此在动态环境中需要周期性更新依赖模型,来逐渐获得更精确的依赖信息.

依赖模型的动态性包括:(1) 节点集动态性:组件和服务在系统运行期间可能动态加入或者删除,导致故障和症状集的动态变化;(2) 因果边动态性:节点间联系的变化可能导致因果边的消失或者增加,从而改变组件-服务间的因果联系;(3) 因果强度动态性:随着模型的不断精确,因果边上的强度可能发生改变.

为了获得精确的依赖模型,需要周期性更新模型.如果更新周期过长,可能丢失模型动态信息.如果更新周期过短,则导致更新开销过大.本文假设已经合理选取更新周期,使得周期内每个节点、依赖边及其权重只会发生一次变化.故障诊断时间窗口长度一般选择与依赖模型更新周期相同,则动态性可能导致时间窗口前后端的模型不相同.如果一个窗口前后端模型相同,可以认为模型未变化,采用基本故障诊断算法即可推理得出故障假设.如果模型发生变化,则变化必然发生于窗口内的某个时间点.由于无法确定变化发生的时间点,也就无法确定该窗口内观察到症状所依赖的模型.症状既可能依赖于前端模型,也可能依赖于后端模型.在这种动态变化的依赖环境下,采用一般诊断算法根据静态模型来推理故障假设会降低算法性能.

在动态模型环境下,需要根据时间窗口前后端模型以及观察到的症状集推理出该窗口内的故障假设.DGA 算法将每个窗口的前后端模型和观察症状的时间信息综合,建立表示当前窗口依赖模型评估值的期望模型,最后根据期望模型使用基本诊断算法推理得出故障假设.

假设窗口前端模型为 $BG_1=(F_1,S_1,P_{F_1},P_{F,S_1})$,窗口后端模型为 $BG_2=(F_2,S_2,P_{F_2},P_{F,S_2})$,观察症状集为 $S_O=S_N\cup S_P$,每个症状 s 的发生时间为 t_S,最终合成的期望模型为 $BG=(F,S,P_F,P_{F,S})$.下面介绍模型合成方法:如果 BG_1 和 BG_2 中对应项相同,则认为该项没有发生变化,简单地将 BG 中对应项置为相同值即可.下面只讨论不相同的情况.

如果 $F_1\ne F_2$,表示组件集在时间窗口内发生变化.由于前后模型中组件都可能发生故障,引发相应负症状.为了在 BG 中不丢失故障集信息,应当包括所有组件,即 $F=F_1\cup F_2$.

如果 $S_1\ne S_2$,表示症状集在时间窗口内发生变化.症状与故障存在依赖关系,为了不丢失故障引发的症状信

息,应当包含所有症状,即 $S=S_1\cup S_2$.

$P_{F,S}$ 为依赖及强度信息,强度值表示为 $P(s|f)$.若模型节点间不存在依赖,则 $P(s|f)=0$.评估期望模型中的依赖强度值需要综合考虑前后模型中对应依赖的强度值,不同环境中前后模型的重要程度可能不同,为此可以为前后依赖强度增加不同的可选权重.$P(s|f)=\alpha\times P_1(s|f)+\beta\times P_2(s|f)$,其中 $\alpha+\beta=1$.

(1) 对于当前窗口未观察到的症状,即不存在该症状的发生时间信息,可以由管理员预先设置权重值,如 $\alpha=\beta=0.5$;(2) 对于当前窗口内观察到的症状,每个症状都有一个发生时间,该时间值可以用来权衡前后模型的重要程度.假设当前时间窗口长度为 Δt,以窗口前端为时间 0,症状 s 发生时间为 $t_s(0\le t_s\le\Delta t)$.直观上感受,t_s 越接近 0,则 s 受到 BG_1 影响的可能性越大;反之,则受到 BG_2 影响的可能性越大.因此,可以预先构造合适的时间相关权重函数 $\alpha(t_s)$和$\beta(t_s)$,如 $\alpha(t_s)=t_s/\Delta t,\beta(t_s)=1-t_s/\Delta t$.

3.4 DGA算法

综合考虑动态故障集和动态模型环境,动态环境下的故障诊断算法伪代码如下:

Algorithm DGA (**greedy algorithm in dynamic environment**).

1) set $H_i=\varnothing$
2) if $F_1\neq F_2$, $F=F_1\cup F_2$; else, $F=F_1$
3) if $S_1\neq S_2$, $S=S_1\cup S_2$; else, $S=S_1$
4) foreach $s\in S$, $f\in F$, do
   if $s\in S_N$ or $s\in S_P$, $P(s|f)=\alpha(t_s)\times P_1(s|f)+\beta(t_s)\times P_2(s|f)$
   else, $P(s|f)=\alpha\times P_1(s|f)+\beta\times P_2(s|f)$
   end
5) if $f\in H_{i-1}$, $belief(f)=P(C_f,f)$; else, $belief(f)=P(f)$
6) foreach $s\in S_P$, do
   update $belief(f)$ as formula(11)
   end
7) foreach $s\in S_N$, do
   update $belief(f)$ as formula(12)
   end
8) set $SuspectedFaults=Parents(S_N)$
9) choose a fault $f\in SuspectedFaults$ that maximizes $belief(f)$;
   $H_i+=f$; $SuspectedFaults^-=f$; $S_P^-=Child(f)$; $S_N^-=Child(f)$
10) if $S_N\neq\varnothing$, return to step 5
11) foreach $f\in H_i$, do
   if $f\in H_{i-1}$, $C_f=C_f+1$; else, $C_f=1$
   end

4 仿真分析

由于不同服务场景产生的模型是类似的,本文只以端到端服务环境为仿真场景.仿真开始时,随机产生 n 个节点的树状或者网状网络[8],使用最短路径优先协议计算网络节点间路由.网络生成和路由确定后,根据第 2 节所述方法建立依赖模型.先验故障概率和条件概率随机产生,分别在[0.001,0.01]和(0,1)内均匀分布.每个仿真场景由如下参数确定:症状观察率 $OR(OR=|S_O|/|S|)$,症状丢失率 P_{loss},虚假症状率 P_s,网络节点数 n.在我们的仿真中,OR 取值 0.1 和 0.5,P_{loss} 取值 0.1,P_s 取值 0.05,n 的范围为 10~150.每个场景中,我们仿真 500 个案例,每个案例对应一个时间窗口,仿真结果均为案例结果的平均值.因为可能存在多个故障同时发生的情况,每个案例随机选择发生故障子集 $F_C\subseteq F,|F_C|\le 5$.

本文使用 4 个指标来评估故障定位算法:诊断率 DR(diagnosis rate),误判率 FPR(false positive rate),诊断率方差 VDR(variance of diagnosis rate)和故障诊断时间.指标的定义如下:

$$DR=|F_C\cap H|/F_C \tag{14}$$

$$FPR = |H - F_C| / |H| \tag{15}$$

$$VDR = \sum_{i=1}^{N} \left(DR_i - \sum_{i=1}^{N} DR_i / N \right)^2 \Big/ N \tag{16}$$

其中,F_C 表示实际发生的故障,H 表示由故障诊断算法产生的故障假设,而 N 表示在同一个故障场景中仿真案例的个数.

DGA 算法可以在同时发生故障集动态和模型动态的环境中使用.但是,为了确定 DGA 在这两个环境下的各自性能,对这两个环境分别仿真.

4.1 动态故障集环境

动态故障集环境下,需要为每个故障确定持续概率函数.简单起见,假设每个故障的持续概率 $P_f(t)=x^t$,x 为(0,1)之间的随机值.为当前窗口案例选择 F_C 时,首先从上一窗口案例 F_C 中选择持续到本窗口发生的故障,然后随机选择其余发生故障加入当前窗口的 F_C.从 F_C 对应可见症状中选择负症状子集 S_N,然后从中选择 $P_{loss}\times|S_N|$ 个症状为丢失症状并从 S_N 中移除,最后选择 $P_s\times|S_N|$个症状为虚假症状并加入 S_N,即得出最终负症状集.观察到的正症状集 $S_P=S_O-S_N$.

动态故障集环境下的仿真结果如图 4~图 7 所示,图中 GA 表示基本故障诊断算法,DGA 表示动态环境下的故障诊断算法.图 4 比较故障诊断率 *DR*.图中可见,随着 *OR* 的增大,诊断率也相应提高,这是由于更多的症状输入诊断算法,可以使诊断更精确.比较相同 *OR* 下的 GA 和 DGA 曲线可见,DGA 比 GA 的诊断率高,这是因为 DGA 考虑了故障修复行为对故障集带来的影响,使用了更准确的先验故障概率.图 5 比较故障误判率 *FPR*.随着 *OR* 的增大,误判率也相应升高.这是由于高 *OR* 导致更多观察症状,按照本文仿真方法,相应地增加了更多虚假症状,从而导致误判率升高.比较同 *OR* 值的 GA 和 DGA 曲线可知,DGA 的误判率要低于 GA.显然,DGA 采用的动态先验故障概率有助于减小误判率.图 6 比较诊断率方差 *VDR*.该值描述了故障诊断率的分散程度,可用于衡量算法的稳定性.*VDR* 的值越低,说明算法的稳定性越好.图中可见,随着 *OR* 的增大,诊断率方差相应减小,算法越稳定.比较同 *OR* 值的 DGA 和 GA 可见,DGA 比 GA 更稳定.图 7 比较算法诊断时间.随着 *OR* 值的升高,需要诊断症状数也相应增加,诊断所花时间同样增加.比较同 *OR* 值的 DGA 和 GA 曲线可见,两者所用时间几乎完全一致.

4.2 动态模型环境

为了仿真模型的动态变化,首先根据前文所述生成一个依赖模型作为窗口前端的模型,然后动态调整生成窗口后端模型.因为模型中节点集的动态变化必然导致依赖边和依赖强度的变化,所以只考虑节点集的动态性.由于窗口前后端模型在诊断算法中的地位相等,因此节点的增加和删除对算法的影响是相似的,可以简化为只考虑节点的增加.

假设模型于窗口内随机时间点 t_a 发生变化.模型变化强度由动态调整率 *AR* 确定,*AR* 取值 0.25 和 0.5.动态调整模型时,增加 $AR\times|F|$个组件节点,每个新增组件选择 m 个服务作为相关依赖节点(m 为组件的平均相关服务数),依赖强度为(0,1)内的随机数.为当前窗口选择观察症状时,为每个观察症状 s_i 关联一个随机发生时间 t_i.如果 $t_i<t_a$,则该症状与前端模型相关;反之,与后端模型相关.将异常症状加入负症状集 S_N,再从中选择 $P_{loss}\times|S_N|$个症状为丢失症状从 S_N 中移除,最后选择 $P_s\times|S_N|$个症状为虚假症状加入 S_N,即得出最终负症状集 S_N.观察到的正症状集 $S_P=S_O-S_N$.

为了简化图形,动态模型环境下的仿真只选取 OR=0.1 的情况,仿真结果如图 8~图 11 所示.图中 AFL 曲线表示文献[11]中算法的性能.AFL 的主要思想是根据接收到的症状调整前后窗口中依赖权重,然后使用窗口前后端模型分别诊断得出故障假设.为了提高诊断率,将这两个故障假设合并作为当前窗口故障假设.图 8 比较故障诊断率 *DR*.图中可见,随着 *AR* 的增大,前后模型差异也增大,算法诊断率下降.比较同 *AR* 的曲线,GA 未考虑模型动态性,诊断率最差.DGA 合成期望模型,得到了最好的诊断率.AFL 的诊断率较 DGA 稍差.图 9 比较误判率 *FPR*.比较不同 *AR* 对应的曲线,发现 *AR* 对 *FPR* 的影响不大,不同 *AR* 对应的同一算法曲线几乎相同.将不同算法曲线比较,GA 的误判率最高,AFL 次之,DGA 最低.GA 未考虑模型变化,所以误判率最高.由于 AFL 将每个窗口内得出的两个假设合并生成最终假设,虽然保证了诊断率,但是也带来了高误判率.DGA 使用一个期望模型来

诊断假设,在保证高诊断率的同时,也减小了误判率.图 10 比较诊断率方差 VDR.比较不同 *AR* 对应的同一算法曲线,高动态性环境对应的 VDR 也大,即算法诊断率不稳定.对比同 AR 对应的不同算法可知,GA 最不稳定,DGA 和 AFL 的稳定性很接近.图 11 比较故障诊断时间.不同 AR 对应的同一算法曲线非常接近,也就是说 AR 对诊断时间影响不大.比较不同算法可见,GA 只需要在一个原有模型基础上做故障诊断,因此花费时间最短.DGA 需要合成期望模型,AFL 需要动态调整前后模型并分别诊断,DGA 和 AFL 的花费时间几乎相同,超过 GA 所需时间.

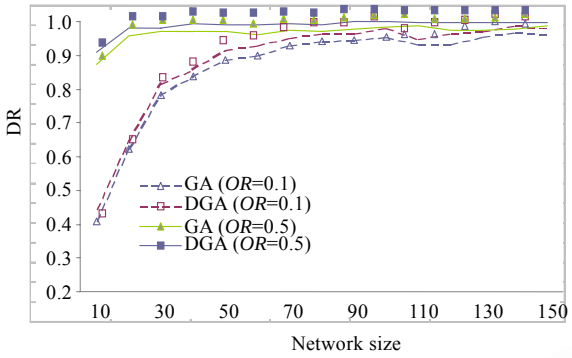

Fig.4 Comparison of DR

图 4 诊断率比较

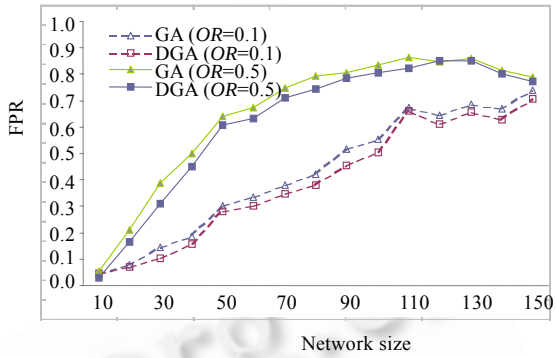

Fig.5 Comparison of FPR

图 5 误判率比较

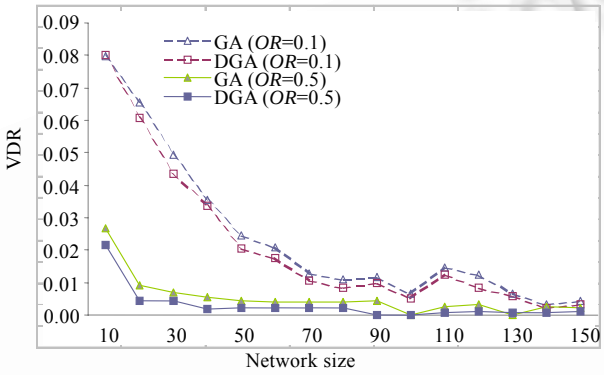

Fig.6 Comparison of VDR

图 6 诊断率方差比较

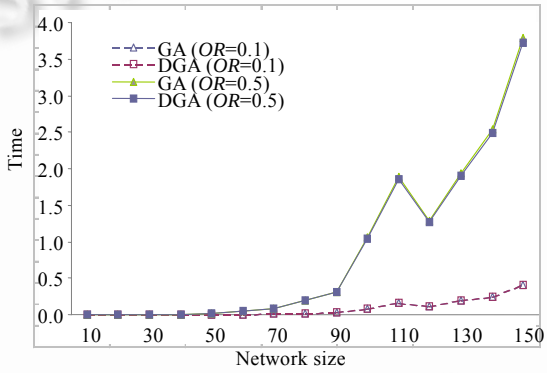

Fig.7 Comparison of time

图 7 诊断时间比较

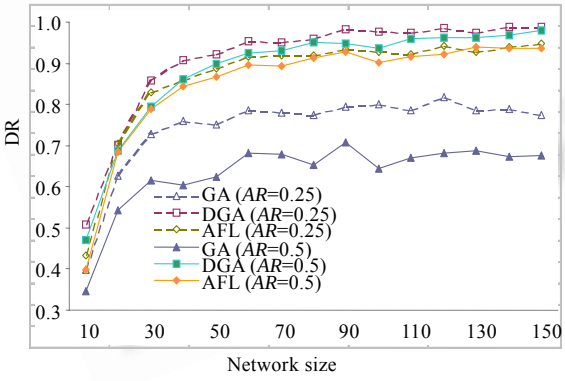

Fig.8 Comparison of DR

图 8 诊断率比较

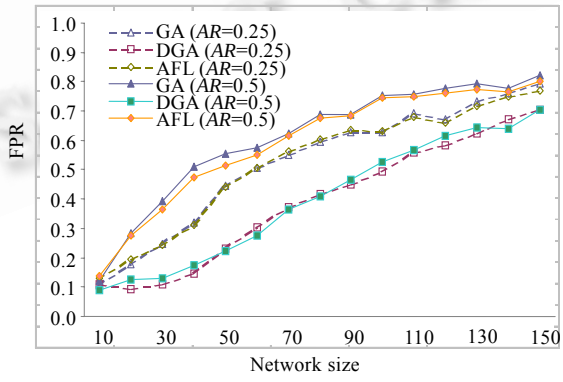

Fig.9 Comparison of FPR

图 9 误判率比较

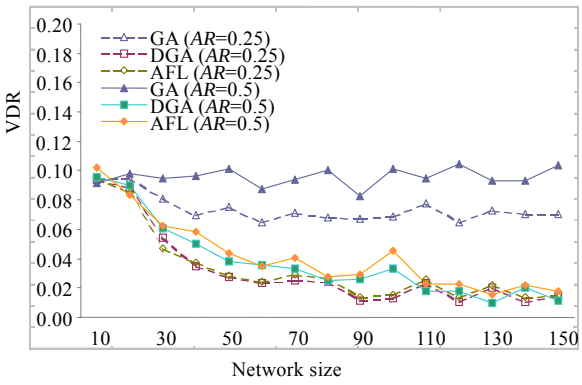

Fig.10 Comparison of VDR

图 10 诊断率方差比较

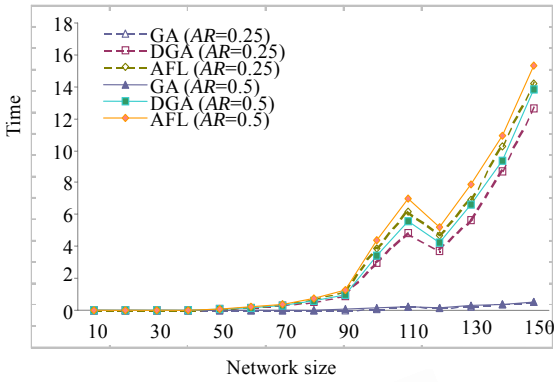

Fig.11 Comparison of time

图 11 诊断时间比较

5 结束语

本文讨论了服务环境中的动态性,分析了动态性给故障诊断算法带来的影响.针对故障集动态性,我们提出的 DGA 算法根据之前故障假设修正当前窗口内先验故障概率;针对模型动态性,DGA 生成当前窗口内的期望模型,最后采用基本诊断算法推理得出假设.仿真结果显示,DGA 算法获得了较好的诊断率、误判率和诊断率方差.由于 DGA 算法生成的期望模型较为复杂,导致诊断时间比基本诊断算法长,下一步可以改进期望模型生成方法,提高诊断时间的性能.

References:

[1] Jakobson G, Weissman M. Alarm correlation. IEEE Network, 1993,7(6):52−59.

[2] Lewis LM. A case-based reasoning approach for the resolution of faults in communication networks. In: Proc. of the 3rd IFIP/IEEE Symp. on Integrated Network Management. San Francisco: North-Holland Publishing Co., 1993. 671−682. http://portal.acm.org/citation.cfm?id=732040

[3] Li F, Thottan M. End-to-End service quality measurement using source-routed probes. In: Proc. of the IEEE INFOCOM. 2006. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4146937

[4] Chen ZX. Proactive probing and probing on demand in service fault localization. The Int'l Journal of Intelligence Control and Systems, 2005,2(2):107−113.

[5] Natu M, Sethi AS. Active probing approach for fault localization in computer networks. In: Proc. of the 4th IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services (E2EMON 2006). 2006. 25−33. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1651276

[6] Nguyen HX, Thiran P. Using end-to-end data to infer lossy links in sensor networks. In: Proc. of the 25th IEEE Int'l Conf. on Computer Communications. Barcelona: IEEE INFOCOM. 2006. 1−12. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4146924

[7] Steinder M, Sethi AS. A survey of fault localization techniques in computer networks. Science of Computer Programming. Computer Systems (AH), 2004,53(22):165−194.

[8] Steinder M, Sethi AS. Probabilistic fault diagnosis in communication systems through incremental hypothesis updating. Computer Networks, 2004,45(4):537−562.

[9] Hasselmeyer P. An infrastructure for the management of dynamic service networks. IEEE Communications Magazine, 2003,41(4):120−126.

[10] Candea G, Kiciman E, Zhang S, Keyani P, Fox A. JAGR: An autonomous self-recovering application server. In: Proc. of the 5th Int'l Workshop on Active Middleware Services. 2003. 168−177. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1210217

[11] Natu M, Sethi AS. Using temporal correlation for fault localization in dynamically changing networks. Int'l Journal of Network Management, 2007. http://portal.acm.org/citation.cfm?id=141551512

[12] Rish I, Brodie M, Ma S, Odintsova N. Adaptive diagnosis in distributed systems. IEEE Trans. on Neural Networks (Special Issue on Adaptive Learning Systems in Communication Networks), 2005,16(5):1088−1109.

[13] Lerner U, Parr R, Koller D, Biswas G. Bayesian fault detection and diagnosis in dynamic systems. In: Proc. of the 17th National Conf. on Artificial Intelligence. AAAI Press/MIT Press, 2000. 531−537. http://portal.acm.org/citation.cfm?id=72111314

[14] Ding JG, Kramer B, Xu SH, Chen HS, Bai YC. Predictive fault management in the dynamic environment of IP networks. In: Proc. of the IEEE Workshop on IP Operations and Management. 2004. 233−239.

[15] Kompella RR, Yates J, Greenberg A, Snoeren AC. Detection and localization of network black holes. In: Proc. of the 26th IEEE Int'l Conf. on Computer Communications. Anchorage: IEEE INFOCOM, 2007. 2180−2188. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4215834

[16] Huang XH, Zou SH, Wang WD, Cheng SD. Fault management for Internet service: modeling and algorithms. In: Proc. of the IEEE Int'l Conf. on Communications (ICC 2006). 2006. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4024235

[17] Bagchi S, Kar G, Hellerstein J. Dependency analysis in distributed systems using fault injection: Application to problem determination in an e-commerce environment. In: Proc. of the 12th Int'l Workshop on Distributed Systems: Operations and Management (DSOM 2001). 2001. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.106.6353

[18] Basu S, Casati F, Daniel F. Web service dependency discovery tool for SOA management. In: Proc. of the 2007 IEEE Int'l Conf. on Services Computing: SOA Industry Summit. 2007. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4278726

[19] Steinder M, Sethi AS. Probabilistic fault localization in communication systems using belief networks. IEEE/ACM Trans. on Networking, 2004,12(5):809−822.

[20] Huang XH, Zou SH, Wang WD, Cheng SD. MDFM: Multi-Domain fault management for Internet services. In: Royo JD, Hasegawa G, eds. Proc. of the 8th Int'l Conf. on Management of Multimedia Networks and Services (MMNS 2005). New York: Springer-Verlag, 2005. 121−132.

[21] Narasimha R, Dihidar S, Ji C, McLaughlin SW. Scalable fault diagnosis in IP networks using graphical models: A variational inference approach. In: Proc. of the IEEE Int'l Conf. on Communications (ICC 2007). 2007. 147−152. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4288703

[22] Hsueh MC, Tsai TK, Iyer RK. Fault injection techniques and tools. Computer, 1997,30(4):75−82.

[23] Yemini SA, Kliger S, Mozes E, Yemini Y, Ohsie D. High speed and robust event correlation. Communications Magazine, 1996, 34(5):82−90.

[24] Chandalia G, Rish I. Blind source separation approach to performance diagnosis and dependency discovery. In: Proc. of the ACM SIGCOMM Conf. on Internet Measurement. 2007. 259−264. http://portal.acm.org/citation.cfm?id=1298342

褚灵伟(1982－),男,江苏启东人,博士生,主要研究领域为服务故障管理,Web 服务技术.

田春岐(1975－),男,讲师,主要研究领域为计算机网络服务质量,P2P 信任管理与激励机制.

邹仕洪(1978－),男,博士,副教授,CCF 高级会员,主要研究方向为移动无线网络,网络服务质量与服务管理.

王文东(1963－),男,教授,CCF 高级会员,主要研究领域为高速网络理论技术.

程时端(1940－),女,教授,博士生导师,主要研究领域为网络性能评估,QoS.