面向智慧医疗云的 SDN 动态负载均衡方法

李雄英[1,2] 董庆贺[1] 何 倩[1,2] 周水明[1,2]
(桂林电子科技大学广西云计算与复杂系统重点实验室 广西 桂林 541004)[1]
(桂林电子科技大学认知无线电与信息处理教育部重点实验室 广西 桂林 541004)[2]

摘 要 文中引入软件定义网络(Software Defined Network,SDN)对智慧医疗云进行网络管理,并且针对传统 SDN 控制器存在单点失效和负载均衡的问题,设计了智慧医疗分布式 SDN 控制器系统。SDN 控制系统分为 SDN 控制器集群、数据转发平面和智慧医疗云服务系统 3 层。在此基础上,提出一种实时负载动态自调节的快速负载均衡算法 DAF(Dynamic Adaptive and Fast Load Balancing)。在该算法中,负载信息感知组件周期性地采集自己的负载信息,自动地进行控制器间的负载信息交互;控制器的负载值超过阈值时,会触发交换机迁移动作,以动态配置交换机与控制器之间的映射关系。实验结果表明,面向智慧医疗云的分布式 SDN 控制系统的性能良好,且 DAF 算法能够快速地实现 SDN 控制器间的负载均衡,提升了智慧医疗云的网络吞吐量。

关键词 智慧医疗云,软件定义网络,负载均衡,单点失效,控制器集群
中图法分类号 TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.11.010

SDN Dynamic Load Balancing Method for Smart Healthcare Cloud

LI Xiong-ying[1,2] DONG Qing-he[1] HE Qian[1,2] ZHOU Shui-ming[1,2]
(Guangxi Key Lab of Cloud Computing and Complex System,Guilin University of Electronic Technology,Guilin,Guangxi 541004,China)[1]
(Key Laboratory of Cognitive Radio and Information Processing of the Ministry of Education,Guilin University of Electronic Technology,Guilin,Guangxi 541004,China)[2]

Abstract This paper introduced software defined network (SDN) to manage smart healthcare cloud network, and designed a smart healthcare distributed SDN controller system to address the problems of single point failure and unbalanced load distribution in traditional SDN controller. This system is composed of three layers: SDN controller cluster, data forwarding plane and smart healthcare cloud service system. On this basis, this paper proposed a DAF (Dynamic Adaptive and Fast Load Balancing) algorithm. In this algorithm, the load information aware component periodically collects its own load information, and automatically interacts the load information between controllers. When the load value of controller exceeds its threshold, the switch migration action is triggered to configure the mapping relationship between switches and controllers dynamically. The experimental results show that the distributed SDN control system for smart healthcare cloud has good performance, besides, the DAF algorithm can quickly balance the load among the SDN controllers and improve the network throughput of healthcare cloud.

Keywords Smart healthcare cloud, Software defined network, Load balancing, Single point failure, Controller cluster

1 引言

随着智慧医疗云网络规模的不断增大,医疗云网络变得越来越复杂。基于传统网络架构的医疗网络体系庞大、结构复杂、更新困难,对新业务的支持日渐不堪重负,这使得医疗云网络的维护与管理变得极其困难且成本高昂。同时,随着医疗数据的增多,医疗云网络的数据传输效率也成为了一个亟待解决的问题。SDN 技术[1]作为一种逻辑集中的新网络体系架构,将传统封闭的网络体系解耦为数据平面、控制平面和应用平面,支持通过软件编程来对网络进行控制管理,提高了实现和部署网络新技术和新协议的灵活性和可操作性。研究者们开始将 SDN 技术应用于智慧医疗云,使医疗云网络的管理更简单化、自动化、智能化,通过 SDN 控制器选择数据传输路径、控制链路带宽等多种方式提高了网络资源的利用率和医疗数据的传输速率。

Bedhief 等人[2]提出了一种面向智慧医疗、智慧城市的物

到稿日期:2017-10-07 返修日期:2018-01-25 本文受国家自然科学基金(61661015),认知无线电与信息处理教育部重点实验室主任基金(CRKL160101),广西云计算与大数据协同创新基金(YD16801),广西密码学与信息安全重点实验室基金(GCIS201701)资助。
李雄英(1993—),女,硕士生,主要研究方向为 SDN 和 QoS;董庆贺(1978—),女,硕士,讲师,主要研究方向为物联网,E-mail:daphny@guet.edu.cn(通信作者);何 倩(1979—),男,博士,教授,CCF 高级会员,主要研究方向为分布式计算、信息安全;周水明(1992—),男,硕士生,主要研究方向为 SDN。

联网架构，结合 SDN 和 Docker 技术，使用一个 POX 控制器来管理 SDN 网络，提升了物联网设备之间的数据传输性能。Suzuki 等人[3]提出了一种新的构建动态 VPN 的方法，并使用 OpenFlow 控制器远程控制 VPN 路由器，保证了医疗信息在各医疗机构之间的流通安全性。Izaddoost 等人[4]利用 SDN 控制器选择医疗数据的最佳传输路径，解决了医疗云网络中的数据传输拥塞问题，提高了实时医疗数据的处理效率。采用单一的 SDN 控制器管理医疗云网络[2-4]，虽然能改善数据传输的性能，但也带来了单一控制器失效所造成的整个医疗云网络瘫痪的问题。因此，不少研究者提出使用多控制器的方式来避免单一节点失效的问题[5-7]。然而，采用多控制器将使到达控制器集群的流量是动态变化的，且控制器和交换机之间的静态配置会导致控制器之间的负载不均衡，从而降低网络性能。

针对上述问题，本文基于 SDN 控制器 Floodlight[1)]，设计了面向智慧医疗云的分布式 SDN 控制器集群系统，提出了一种基于医疗云网络中控制器节点负载信息的负载均衡算法——动态自调节的快速负载均衡算法 DAF（Dynamic Adaptive and Fast Load Balancing）。

2 相关工作

2.1 智慧医疗

近年来，智慧医疗的研究主要集中在医疗应用管理、医疗数据传输、医疗数据安全、医疗设备监控等方面。Silva 等人[8]提出了一种基于 SDN 并支持上下文感知移动方法的网络框架，该网络框架保障了巴西智慧医疗系统的网络连通性和医疗云应用的服务质量。Hu 等人[9]提出了基于 POF 的源路由协议，其改善了转发平面的效率，提高了医疗数据的传输速率。Moustafa 等人[10]基于 WebRTC 实现了远程用户、医疗设备与医疗云数据中心的通信，并对医疗云中的医疗设备进行了实时的远程监控。Boussada 等人[11]基于上下文感知，提出了一种保障医疗云中医疗数据安全性的方法，该方法能保护病人的个人隐私。

上述文献大多利用了 SDN 技术，使得医疗云的网络自动化管理和性能得到提升，但都未涉及医疗云网络中控制平面性能的研究。在医疗云的网络中，控制平面的性能对整个网络的性能起着决定性作用，而控制器作为控制平面的核心，成为了提升控制平面性能的关键。

2.2 SDN 控制器

目前，针对 SDN 多控制器的负载均衡问题的解决办法大致分为两类：集中式决策型[12-13]和分布式决策型[14-15]。Dixit 等人[12]提出了“ElastiCon”，用一个负载均衡器收集整个控制平面的负载信息，并根据负载信息相应地增加或减少控制池中控制器的数量；同时还提出了交换机迁移协议，实现了负载的迁移。但是，其对控制器的负载状态的评估及目标控制器的选择并无具体描述。Bari 等人[13]通过添加监控模块实时收集所有控制器的流量数据，并根据流量的实时变化向过载控制器发送命令，对交换机进行动态调整。但是，该方法对迁移代价的衡量较为单一。

分布式决策型，即每个控制器通过收集其他所有控制器的负载信息以获知全网控制器的负载信息，并在本地执行负载均衡策略[14-15]。Zhou 等人[14]提出了一种允许每个控制器在本地执行的负载均衡策略，实现了控制器间的负载均衡。文献[15]在文献[14]的基础上降低了控制器间负载信息的交互频率，但其控制器负载状态的评估条件单一。这类方法不需要监控模块，但控制器间负载信息的交互会增加控制器的负载，从而增加负载均衡完成的时间。

3 智慧医疗 SDN 控制器的系统架构

本文基于 Floodlight 控制器，利用若干个控制器节点组成的集群构建了分布式控制平面，从而进行智慧医疗云的网络管理。系统架构从上至下分为 3 层，第 1 层是 SDN 控制器集群，第 2 层是数据转发平面，第 3 层是智慧医疗云服务系统。系统架构如图 1 所示。

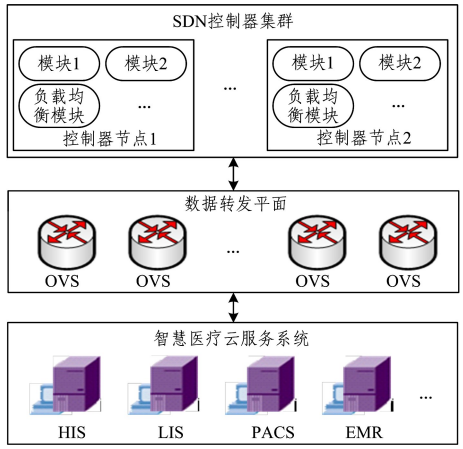

图 1 面向智慧医疗的 SDN 控制器系统架构

Fig. 1 SDN controller system architecture for smart healthcare

SDN 控制器集群由包含负载均衡模块的多个 SDN 控制器节点组成。该层是整个网络的控制核心，通过调用网络视图的全局数据，进而操作交换机，实现对整个医疗网络的管理和控制。

数据转发平面由多个 OpenFlow 交换机组成。该层基于 OpenFlow 协议，按照控制器集群下发的流表对医疗数据进行转发传输。

智慧医疗云服务系统是指部署在各医院的医疗信息化系统，主要负责采集和处理医疗数据。

本文采用动态自调节的快速负载均衡策略实现智慧医疗系统网络中控制器间的负载均衡，并在 SDN 控制器中设计了负载均衡模块用于执行负载均衡策略。负载均衡模块主要由负载信息感知、负载信息交互、负载均衡决策和交换机迁移 4 个部分组成。负载信息感知部分负责周期性地采集系统中各控制器的负载信息；负载信息交互部分负责控制器间负载信息的交互操作，即每个控制器将本地采集的负载信息发送给其他控制器；负载均衡决策部分负责根据整体控制器的负载情况做出均衡负载的决策；交换机迁移部分负责执行交换机的迁移动作。

1) http://www.projectfloodligh.org

4　动态自调节的快速负载均衡策略

动态自调节的快速负载均衡策略根据控制器的实时负载状态,对控制器间的负载进行动态调整。当系统中有过载控制器时,会触发控制器执行负载均衡策略,即由 DAF 算法选择出过载控制器、迁移交换机及接收迁移交换机的控制器,并执行交换机迁移动作,使控制器间的负载达到均衡状态。

4.1　相关定义

源控制器:需将其 master 角色的交换机迁移至其他控制器的控制器。

目标交换机:执行迁移动作的交换机。

目的控制器:接收目标交换机的控制器。

本文用到的相关符号及其含义如表 1 所列。

表 1　相关符号及其含义

Table 1　Notations and corresponding meanings

L_{now}	当前时刻的负载
L_{pre}	前一时刻的负载
$count$	控制器的数量
T	负载测试周期
R	消息到达速率
M	内存利用率
U	CPU 利用率
Rt	交换机与控制器间的往返时间
$R_{overload}$	控制器的过载率
$overLoadId$	过载控制器的 ID
$targetSwId$	目标交换机的 ID
$dstConId$	目的控制器的 ID
Pri_{des}	目的控制器的优先权值
Thr_{des}	目的控制器的负载阈值
L_{des}	目的控制器的负载
L_{mig}	迁移负载
Rs	每个交换机向控制器提出的平均 packet_in 消息请求速率
$targetRs$	目标交换机的消息到达速率
N	master 交换机数目与交换机总数目的比值
$\lambda_1,\lambda_2,\lambda_3,\lambda_4$	计算负载值的权重参数,$\lambda_1+\lambda_2+\lambda_3+\lambda_4=1$
q	交换机给控制器增加的负载
x_1,x_2	计算控制器优先权值的权重参数,$x_1+x_2=1$
μ_1	计算迁移负载的权重参数
μ_2	计算目的控制器成为新过载控制器的阈值参数

4.2　动态自调节的快速负载均衡算法

DAF 算法通过动态调节医疗云网络中控制器间的负载,快速地实现控制器间的负载均衡,提高了医疗云网络的吞吐量。DAF 算法的具体执行过程如算法 1 所示。

算法 1　DAF(L_{now},L_{pre},Thr,T,Rt,$count$,q,x_1,x_2,μ_1,μ_2)

输入:L_{now},L_{pre},Thr,T,Rt,count,q,x_1,x_2,μ_1,μ_2

输出:LoadBalanceFlag(True or False)

```
1. loadBalanceFlag=False
2. while (t == nT)
3.    loadInforming(L_now, L_pre, q, Thr )
4.    while L_now != Ø do
5.       overLoadIds←srcControllerSelect(L_now, Thr, count)
6.       targetSwId, targetRs←targetSwitchSelect(Rs, overLoadId)
7.       dstConId←dstConSelect(x_1, x_2, μ_1, μ_2, targetSwId, targetRs, Rt)
8.       将 id 为 targetSwId 的交换机从 id 为 overLoadId 的控制器迁移至 id 为 dstConId 的控制器
9.       loadBalanceFlag=True
10.   end while
11. end while
12. return loadBalanceFlag
```

首先,各控制器利用负载信息感知组件周期性地采集自己的负载信息。然后,负载信息交互部分判断控制器是否需要将自己的负载信息发送给其他控制器,以完成控制器间的负载信息交互(算法 1 第 3 行)。当控制器的负载值超过阈值时,会触发交换机迁移动作。过载控制器通过负载均衡决策部分选择出源控制器、目标交换机、目的控制器(算法 1 第 5－7 行),进而完成迁移动作,即将目标交换机从源控制器迁移至目的控制器(算法 1 第 8 行)。最后,检查控制器集群是否负载均衡,若不均衡,则继续执行迁移动作;若集群中控制器的负载已均衡,则继续更新控制器的负载信息。

4.2.1　负载感知

负载感知部分负责采集每个节点的负载信息,主要包括控制器 master 角色的交换机数与系统中总交换机数的比率 N、控制器的平均 packet_in 消息请求速率 R、控制器的 CPU 利用率 U 和内存利用率 M。根据以上信息,由式(1)可计算得到控制器的负载 L。

$$L=\lambda_1*N+\lambda_2*R+\lambda_3*U+\lambda_4*M \tag{1}$$

本文还选择了另外两个度量用于负载均衡决策:每个交换机向控制器提出的平均 packet_in 消息请求速率 Rs,每个交换机到控制器的往返时间(RTT)Rt。Rs 用于判断该交换机对控制器造成的负载值;RTT 是衡量数据传输的一个重要因素,因此将作为选择目的控制器的一个参数。因为控制器和交换机之间每一跳的延迟几乎是固定的,所以本文用控制器到交换机的跳数值代替控制器和交换机之间的 RTT 值。

4.2.2　负载信息交互

每个控制器需周期性地将自己的负载信息发送给其他控制器,并存储和处理其他控制器的负载信息,以实现控制器间的负载信息交互。虽然控制器间周期性地交互负载信息会减少负载均衡决策的时间,但会给控制器增加额外的处理开销和通信开销,特别是当控制器在相邻时刻的负载值变化较小时,向其他控制器发送负载信息是多余的操作。为了减少这些开销,我们基于文献[15]提出了一种新的抑制算法,该算法能减少负载信息交互的次数,其具体描述如算法 2 所示。

算法 2　LoadInforming(L_{now},L_{pre},q,Thr)

输入:L_{now},L_{pre},q,Thr$\{V_0=0,V_1,V_2,\cdots,V_7=Thr-q\}$,$V_m-V_{m-1}<V_{m-1}-V_{m-2}$,$m\geqslant 2$

输出:Informing load information (True or False)

```
1. LoadInformFlag=False
2. if (L_now < (Thr−q)) then
3.    for (i:1→6)
4.       if (L_now ≥ V_i && L_pre < V_i) || (L_now < V_i && L_pre ≥ V_i)
5.          LoadInformFlag=True
6.          break
7.       end if
8.    end for
9.    L_pre=L_now
10. else
11.   LoadInformFlag=True
```

12. $L_{pre}=L_{now}$
13. end if
14. return LoadInformFlag

该算法把 0 至负载阈值与 q 的差值分为 7 个区间，当控制器前一时刻的负载和当前时刻的负载值属于同一区间时，控制器将不发送负载信息给其他控制器。通过这种方法，可以减少因控制器间频繁的负载信息交互操作造成的冗余开销，但这种方法会导致某控制器在当前时刻的负载与其他控制器记录的该控制器负载值存在偏差。这种偏差对控制器的影响分为两种情况：1)该控制器的负载低，即使接收目标交换机，也仍然不会过载；2)该控制器的负载较高，它可能会因接收了目标交换机而变成新的过载控制器。为避免发生第二种情况，设计算法中的区间长度呈递减趋势，使得越接近负载阈值的控制器越频繁地发送负载信息给其他控制器。此外，还增加了一个权值 q，q 一般取值大于或等于一台交换机可能对控制器增加的负载值。当控制器的当前负载大于或等于负载阈值与 q 之间的差值时，表示该控制器的负载值发生了上述偏差，因被选为目的控制器而成为新过载控制器的概率很高，因此设计该控制器发送当前负载信息给其他控制器。通过上述两种方式，可有效避免负载偏差带来的负面影响。

4.2.3 负载均衡决策

负载均衡决策部分负责选择过载情况最严重的控制器作为源控制器，然后决策出目标交换机进行迁移，并决策出目的控制器来接收目标交换机。

1)源控制器选择策略

在控制器集群中，只会出现一台控制器过载或多台控制器过载两种情况。针对这两种情况，我们提出了一种源控制器选择算法，如算法 3 所示。

算法 3 srcControllerSelect(L_{now}, Thr, $count$)

输入：L_{now}, Thr, count
输出：overLoadId
1. if (count==1)
2. if ($L_{now}\geqslant$Thr)
3. overLoadId=this.id
4. end if
5. else
6. for (i:0→count−1)
7. if (L_{now} i$\geqslant$Thr i)
8. $R_{overload}$←getOverLoad(i)
9. add to Map<id,$R_{overload}$>overLoadCon
10. end if
11. end for
12. overLoadId←overLoadCon sort by $R_{overload}$ desc
13. end if
14. return overLoadId

该算法根据过载控制器的数量分情况分别进行处理。当只有一台控制器处于过载状态时，选择这台控制器作为源控制器(算法 3 第 1−3 行)。当同时有多台控制器处于过载状态时，若这些过载控制器选择的目的控制器相同，那么该目的控制器在接收目标交换机之后可能会成为新的过载控制器。为解决这个问题，首先通过式(2)计算每个过载控制器的过载率 $R_{overload}$(算法 3 第 6−11 行)，然后将过载控制器根据过载率进行降序排序，并选择过载率最大的控制器作为源控制器(算法 3 第 12 行)。当多台控制器的过载率相等且均为最大值时，从这些过载率最大的控制器中随机选择一台控制器作为源控制器。

$$R_{overload}=(L_{now}-Thr)/Thr \tag{2}$$

2)目标交换机选择策略

确定执行负载均衡策略的源控制器后，需在源控制器作为 master 控制器的交换机中选择目标交换机。通过负载感知部分，测量出每个交换机向源控制器提出的平均 packet_in 消息请求速率 Rs。Rs 值越大，表示源控制器需要处理该交换机的请求越多，即该交换机消耗源控制器的负载越多。为了降低源控制器的负载，将控制器管理的所有交换机按 Rs 值进行降序排序，并选择 Rs 值最大的交换机作为目标交换机。算法的具体描述如算法 4 所示。

算法 4 targetSwitchSelect(Rs, $overLoadId$)

输入：overLoadId, Rs
输出：targetSwId, targetRs
1. switchsId[]←getSwitchs(overLoadId)
2. for (i:0→switchsId[].size()−1)
3. Rs←getRs(switchsId[i])
4. Map<switchsId[i],Rs> switchPkt
5. end for
6. OrderSwitchPkt←switchPkt sort by Rs desc
7. targetSwId, targetRs←get the first key of OrderSwitchPkt
8. return targetSwId, targetRs

3)目的控制器选择策略

一个交换机可以拥有一个 master 角色的控制器和多个 slave 角色的控制器。对于每个目标交换机，其多个 slave 角色的控制器都可以接受它的迁移。为此，我们提出了一种目的控制器选择算法，用于从多个 slave 角色的控制器中选择优先权值最高的控制器作为目的控制器来接收迁移交换机。算法的具体描述如算法 5 所示。

算法 5 dstConSelect(x_1, x_2, μ_1, μ_2, $targetSwId$, $targetRs$, Rt)

输入：x_1, x_2, targetSwId, targetRs, Rt
输出：dstConId
1. dstCons[]←getdstCons(targetSwId)
2. for (i:0→dstCons[].size()−1)
3. Thr, L_{now}←getLoadInformation(i)
4. Pri=x_1*(Thr− L_{now})−x_2*Rt
5. add to Map⟨dstConts[i],Pri⟩dstConsInfo
6. end for
7. OrderdstConsInfo←dstConsInfosort by Pri desc
8. for (key:OrderdstConsInfo.keySet())
9. L_{mig}=targetRs* μ_1
10. if ($L_{mig}<\mu_2$*(Thr−L_{now}))
11. dstConId=key
12. break
13. end if
14. end for
15. return dstConId

该算法中,若选择负载值最小的控制器作为目的控制器,则可能会出现目的控制器与目标交换机之间的 RTT 最大的情况,会降低数据的传输性能。为避免发生这种情况,综合考虑控制器的负载值和目的控制器与目标交换机的 RTT 值两个因素,给每个空闲控制器分配一个优先权值 Pri:

$$Pri_{\mathrm{des}}=x_1*(Thr_{\mathrm{des}}-L_{\mathrm{des}})-x_2*R \tag{3}$$

为确保目的控制器不会因接收目标交换机而变成新的过载控制器,在选择目的控制器的过程中,控制器还需满足式(4)所示的约束条件:

$$L_{\mathrm{mig}}<\mu_2*(Thr_{\mathrm{des}}-L_{\mathrm{des}}) \tag{4}$$

4.2.4　交换机迁移

交换机迁移组件主要负责交换机的动态迁移。为了避免在迁移交换的过程中出现丢包的情况,采用图 2 所示的无缝迁移交换机方法。

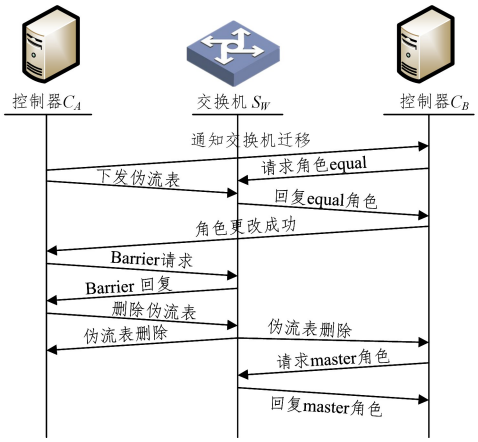

图 2　交换机动态迁移的流程图

Fig. 2　Flowchart of dynamic migration of switchboard

首先,源控制器 C_A 发送一个交换机迁移请求消息给目标控制器 C_B。为了维持 S_W 交换机流表的一致性,保证一直有且只有一个控制器处理交换机的消息,C_A 控制器下发一个伪流表至交换机 S_W。因为伪流表不匹配任何数据流,所以不会影响 OpenFlow 网络。然后,C_B 发送一个更改其角色为 equal 的消息至 S_W,C_B 会收到交换机的回复消息,并告知 C_A 自己的角色已经更改为 equal。为确保交换机中所有消息都已处理完毕,C_A 发送 barrier 消息至交换机。交换机在处理完 barrier 消息后回复 C_A。最后,C_A 下发伪流表删除消息至交换机 S_W,S_W 删除伪流表后通知所有与之相连的控制器进行角色更改。在收到通知消息后,C_A 将设置角色为 slave,C_B 将设置角色为 master,并开始负责管理 S_W。

4.2.5　算法性能分析

DAF 算法包含了 4 个子算法(即算法 2－算法 5),因而其性能取决于每个子算法的优劣。在负责负载信息交互的算法 2 中,需循环判断控制器前一时刻的负载和当前时刻的负载值是否属于同一区间,循环 6 次,时间复杂度和空间复杂度都是 O(1)。在文献[15]提出的算法中,负载信息交互需循环 $n-1$ 次,时间复杂度是 $O(n-1)>O(1)$。因此,DAF 算法中的负载信息交互算法的时间复杂度有所优化。为获取以过载率为 $Value$ 的过载控制器集合,算法 3 遍历了 n 个控制器,计算控制器的过载率,并对控制器按过载率进行排序,时间复杂度是 $O(n+n\lg n)$,空间复杂度是 $O(2n)$。算法 4 需要计算 n 个交换机的 Rs 值,时间复杂度和空间复杂度均为 $O(n)$,并采用 Collections. sort()方法对 map 进行排序,时间复杂度为 $O(n\lg n)$,空间复杂度为 $O(n)$,因而算法 4 的总时间复杂度为 $O(n+n\lg n)$,总空间复杂度为 $O(2n)$。最后,算法 5 需遍历 n 个备选目的控制器,计算并保存控制器的优先权值,并对控制器按优先权值进行排序,时间复杂度为 $O(n+n\lg n)$,空间复杂度为 $O(2n)$。

4 个子算法的时间复杂度分别为 $O(1)$,$O(n+n\lg n)$,$O(n+n\lg n)$,$O(n+n\lg n)$;空间复杂度分别为 $O(1)$,$O(2n)$,$O(2n)$,$O(2n)$。因此,DAF 算法的总时间复杂度为 $O(n\lg n)$,总空间复杂度为 $O(n)$。DAF 算法占用的内存空间少,运行效率高,能够快速地实现控制器间的负载均衡,增加网络的吞吐量。

5　实验结果与分析

本文通过仿真实验来验证 DAF 算法的有效性,比较该算法与现有的静态配置算法的系统吞吐量,并与文献[16]中的算法对比负载均衡策略的完成时间。

5.1　实验环境

本文使用 Mininet[1)] 模拟医疗云网络,并进行相关实验。实验中搭建了 3 个 OpenFlow 控制器节点和 6 个 OpenFlow 交换机。每台控制器连接不同数量的交换机,每个交换机除了连接一个 master 控制器,还连接另外两个控制器作为 slave 控制器。交换机 S_1,S_2,S_3 与控制器 $A(C_A)$,$B(C_B)$,$C(C_C)$ 相连,其中 C_A 为 master 控制器,C_B 和 C_C 为 slave 控制器。交换机 S_4,S_5 与 3 个控制器相连,其中 C_B 为 master 控制器,C_A 和 C_C 为 slave 控制器。交换机 S_6 连接的 3 个控制器中,C_C 为 master 控制器,C_A 和 C_B 为 slave 控制器。每个 OpenFlow 交换机都连接一个主机。通过 Cbench[2)] 测试可得,单个控制器每秒最多处理 14997 个数据包。设定系统负载阈值为 80%,负载监测周期 $T=1$ s,权值参数 $\lambda_1=0.1$,$\lambda_2=0.7$,$\lambda_3=0.1$,$\lambda_4=0.1$,$x_1=0.5$,$x_2=0.5$,$\mu_1=0.7$,$\mu_2=0.85$。

5.2　均衡性能

为验证 DAF 算法的可行性,本文通过注入大量数据包的方式使部分控制器的负载超过阈值,从而触发负载均衡策略。通过观察系统中各控制器的负载状态是否由不均衡状态变成均衡状态,来验证 DAF 算法实现控制器间负载均衡的可行性。同时,通过与文献[15]提出的算法进行对比,来验证 DAF 算法是否缩短了负载均衡的完成时间。

实验测试步骤:主机间模拟智慧医疗云服务系统之间发送医疗数据包,交换机将医疗数据包封装在 packet_in 报文中并发送给控制器。设定交换机发送 packet_in 消息的速率为每秒 2000 个;从第 6s 开始,调整交换机发包速率为每秒 5000

1) http://mininet.org

2) http://sourceforge.net/projects/cbench

个。实验结果如图 3 所示。

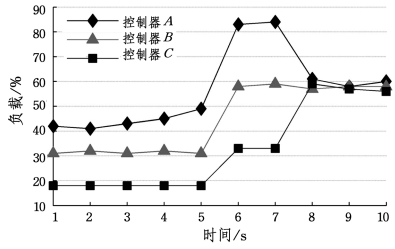

图 3 负载均衡的完成时间

Fig. 3 Completion time of load balancing

由图 3 可知,在前 5s,因为 packet_in 数据包请求量少且请求速率稳定,所以 3 个控制器的负载值基本保持不变。第 6s 时,packet_in 数据包请求量增多,引起 C_A,C_B,C_C 的负载急剧增加,导致 C_A 的负载超过了其阈值,触发了交换机迁移动作,并将部分负载转移至 C_C。因此,在第 7s 到 8s 间,C_C 的负载上升,C_A 的负载下降。在第 8s 时,负载均衡过程执行完毕,完成时间在 2s 内。记录负载均衡策略开始执行的时间和控制器达到负载均衡的时间,两者的差值即为负载均衡的完成时间。记录 10 组数据,计算其平均值即可得到负载均衡的平均完成时间为 1.087s。

文献[15]中负载均衡的完成时间在 5s 内,其控制器 A 作为 master 角色连接了 4 个交换机,但是该方案并没有明确说明负载均衡策略执行过程中迁移交换机的数目。根据控制器连接的交换机数,可知其迁移的交换机数目不大于 3。以最好情况计算,即假设 5s 迁移了 3 个交换机,那么在迁移一个交换机的情况下,完成负载均衡的平均时间是 1.67s。而在本方案中,迁移一个交换机时,负载均衡的平均完成时间是 1.087s(少于 1.67s)。因此,本文提出的 DAF 算法能够有效地减少负载均衡的完成时间。

5.3 系统吞吐量

通过与静态算法进行对比,来验证 DAF 算法是否能够有效地提高医疗云系统的数据吞吐量。静态算法的核心思想是静态配置控制器与交换机之间的映射关系。吞吐量指控制器每秒内能够处理事务的数量,它是衡量控制器负载均衡算法性能的一个重要指标。本实验将系统中所有控制器每秒处理 packet_in 消息的总数目作为系统吞吐量。

实验测试步骤:主机间模拟医疗云服务系统之间发送医疗数据包。测试开始的前 10s,每秒向 3 个控制器分别注入 8000 个 packet_in 消息。从第 11s 开始,每秒向 C_A,C_B,C_C 分别注入 15000 个、6000 个、3000 个 packet_in 数据包。实验结果如图 4 所示。

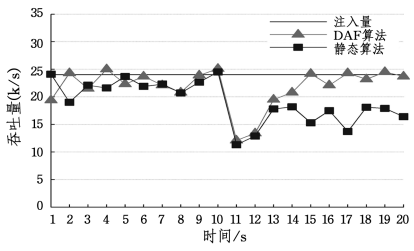

图 4 系统吞吐量

Fig. 4 System throughput

由图 4 可知,在前 10s,因为医疗云系统中控制器的负载没有超出其处理范围,所以静态算法和 DAF 算法的吞吐量近似相等。在第 11s 时,两种算法的系统吞吐量都急剧下降。这是因为 packet_in 请求量的增多导致了控制器间的负载不均衡。控制器 A 接收的 packet_in 数量超过了其处理能力,缓冲区溢出导致 packet_in 数据包丢失,从而造成系统吞吐量下降。但是,静态算法的系统吞吐量从第 11s 开始基本维持在较低水平,而 DAF 算法的系统吞吐量从第 12s 开始逐渐上升,并恢复至正常水平。这是因为 DAF 算法使得过载 C_A 的部分负载转移至 C_C,且不会造成 C_C 过载,从而增加了智慧医疗云系统的吞吐量。

结束语 为解决智慧医疗云网络中存在的单点失效问题,设计了面向智慧医疗云的分布式 SDN 控制器集群系统。针对该集群系统中多控制器的负载不均衡问题,本文提出并实现了一种基于医疗网络控制器节点负载信息的动态自调节的负载均衡算法 DAF。经实验验证,该集群系统避免了医疗云网络的单点失效问题,且通过 DAF 算法实现了控制器间的负载均衡,缩短了负载均衡完成的时间,增加了医疗系统的吞吐量。本文算法是通过 mininet 仿真软件进行验证的,并未在真实的医疗云网络中测试。此外,本文将 CPU 利用率、内存利用率、交换机比率和 packet_in 消息速率作为负载因子,仍可以进一步优化。下一步计划将本算法应用于真实的医疗云网络中,并通过优化负载因子和负载均衡决策,进一步缩短负载均衡完成的时间,优化智慧医疗云的网络性能。

参 考 文 献

[1] ZUO Q Y,CHEN M,ZHAO G S,et al. Research on OpenFlow-Based SDN Technologies[J]. Journal of Software,2013,24(5):1078-1097. (in Chinese)
左青云,陈鸣,赵广松,等. 基于 OpenFlow 的 SDN 技术研究[J]. 软件学报,2013,24(5):1078-1097.

[2] BEDHIEF I,KASSAR M,AGUILI T. SDN-based architecture challenging the IoT heterogeneity[C]// 2016 3rd Smart Cloud Networks & Systems(SCNS). Dubai,2016:787-558.

[3] LEE J S,OBI T,SUZUKI H,et al. A new method for constructing dynamic VPN cooperating with OpenFlow control technology and healthcare PKI[C]// 2015 17th Asia-Pacific Network Operations and Management Symposium (APNOM-S). Busan,2015:432-435.

[4] IZADDOOST A,MCGREGOR C. Enhance Network Communications in a Cloud-Based Real-Time Health Analytics Platform Using SDN[C]// 2016 IEEE International Conference on Healthcare Informatics (ICHI). Chicago,IL,2016:388-391.

[5] LIU L,ZHOU J T. A review of software definition of network control plane[J]. Computer Science,2017,44(2):75-81. (in Chinese)
柳林,周建涛. 软件定义网络控制平面的研究综述[J]. 计算机科学,2017,44(2):75-81.

[6] LEVIN D,WUNDSAM A,HELLER B. Logically centralized? State distribution trade-offs in software defined networks[C]// Proceedings of the ACM SIGCOMM Workshop on HotSDN. 2012:1-6.

[7] WANG M M,LIU J W,CHEN J,et al. Software definition net-

work:security model, mechanism and research progress [J]. Journal of Software,2016,27(4):969-992. (in Chinese)
王蒙蒙,刘建伟,陈杰,等. 软件定义网络:安全模型、机制及研究进展[J]. 软件学报,2016,27(4):969-992.

[8] SILVA F,CASTILLO L,LEMA A,et al. Software defined eHealth networking towards a truly mobile and reliable system [C]// 2014 IEEE 16th International Conference on e-Health Networking, Applications and Services (Healthcom). Natal, 2014:560-564.

[9] LI S,HU D,FANG W,et al. Source routing with protocol-oblivious forwarding (POF) to enable efficient e-Health data transfers[C]// 2016 IEEE International Conference on Communications (ICC). Kuala Lumpur,2016:1-6.

[10] MOUSTAFA H,SHEN G,KAMATH S,et al. Remote monitoring and medical devices control in eHealth[C]// 2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob). New York, NY,2016:1-8.

[11] BOUSSADA R,ELHDHILI M E,SAIDANE L A. QoS enabled privacy preserving solution for eHealth systems[C]// International Conference on PERFORMANCE Evaluation and Modeling in Wired and Wireless Networks. IEEE,2017.

[12] DIXIT A,FANG H,MUKHERJEE S,et al. ElastiCon:an elastic distributed SDN controller[C]// 2014 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS). Marina del Rey,CA,2014:17-27.

[13] BARI M F,ROY A R,ZHANG Q,et al. Dynamic Controller Provisioning in Software Defined Networks[C]// Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013). Zurich,2013:18-25.

[14] ZHOU Y,ZHU M,XIAO L,et al. A Load Balancing Strategy of SDN Controller Based on Distributed Decision[C]// 2014 IEEE 13th International Conference on Trust,Security and Privacy in Computing and Communications. Beijing,2014:851-856.

[15] YU J K,WANG Y,PEI K K,et al. A load balancing mechanism for multiple SDN controllers based on load informing strategy [C]// 2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS). Kanazawa,2016:1-4.

(上接第 65 页)

[2] CURRY R M,SMITH J C. A survey of optimization algorithms for wireless sensor network lifetime maximization[J]. Computers & Industrial Engineering,2016,101:145-166.

[3] HEINZELMAN W R,CHANDRAKASAN A,BALAKRISHNAN H. Energy-efficient communication protocol for wireless microsensor networks[C]// Proceedings of the 33rd Annual Hawaii International Conference on System Sciences. USA: IEEE Press,2000:1-10.

[4] CHEN Z S,SHEN H. Energy efficient wireless sensor network routing protocol [J]. Computer Science, 2015, 42(8): 90-94, 117. (in Chinese)
陈战胜,沈鸿. 能量高效的无线传感器网络路由协议[J]. 计算机科学,2015,42(8):90-94,117.

[5] YOUNIS O,FAHMY S. HEED:a hybrid,energy-efficient,distributed clustering approach for ad hoc sensor networks[J]. IEEE Transactions on Mobile Computing,2004,3(4):366-379.

[6] XU Y,HEIDEMANN J,ESTRIN D. Geography-informed energy conservation for ad hoc routing[C]// Proceedings of the 7th Annual International Conference on Mobile Computing and Networking. Rome: ACM Press,2001:70-84.

[7] JANNU S,JANA P K. A grid based clustering and routing algorithm for solving hot spot problem in wireless sensor networks [J]. Wireless Networks,2016,22(6):1901-1916.

[8] HALEEM FARMAN H J,AHMAD J,JAN B,et al. Grid-Based Hybrid Network Deployment Approach for Energy Efficient Wireless Sensor Networks[J]. Journal of Sensors, Research Gate,2016,2016(3):1-16.

[9] MENG X,SHI X,WANG Z,et al. A grid-based reliable routing protocol for wireless sensor networks with randomly distributed clusters[J]. Ad Hoc Networks,2016,51(C):47-61.

[10] ZHU M,XIAO Z,LIU H L,et al. A Clustering Routing Algorithm Based on Virtual Grid in WSN[J]. Journal of Sichuan University (Engineering Science Edition), 2012, 44(5): 143-148. (in Chinese)
朱敏,肖震,刘昊霖,等. WSN 中基于虚拟网格的分簇路由算法[J]. 四川大学学报(工程科学版),2012,44(5):143-148.

[11] ZHANG H B,SHEN H. Balancing Energy Consumption to Maximize Network Lifetime in Data-Gathering Sensor Networks [J]. IEEE Transactions on Parallel & Distributed Systems, 2008,20(10):1526-1539.

[12] AKBAR M, JAVAID N, IMRAN M,et al. A multi-hop angular routing protocol for wireless sensor networks[J]. International Journal of Distributed Sensor Networks,2016,12(9):1-13.

[13] FENG R,LI T,WU Y,et al. Reliable routing in wireless sensor networks based on coalitional game theory[J]. IET Communications,2016,10(9):1027-1034.

[14] HAN K H H,KO Y B,KIM J H. A novel gradient approach for efficient data dissemination in wireless sensor networks[C]// IEEE 60th Vehicular Technology Conference. Los Angeles: IEEE Press,2004:2979-2983.

[15] ZHENG M C,ZHANG D F,ZHAO X C. The study on the behavioral characteristics of the wireless sensor network of minimum hop routing [J]. Computer Application, 2007, 27(10): 2552-2555. (in Chinese)
郑明才,张大方,赵小超. 最小跳数路由无线传感器网络行为特征研究[J]. 计算机应用,2007,27(10):2552-2555.