·人工智能及识别技术· 文章编号:1000-3428(2018)05-0146-09 文献标志码:A 中图分类号:TP391

基于 k 最短路径算法优化与负载均衡的虚拟网络映射机制

高 斐[1],陈德礼[1],洪家军[1],于 智[2],田 甜[3]

(1. 莆田学院 信息工程学院,福建 莆田 351100;
2. 浙江大学 计算机科学与技术学院,杭州 310027;3. 审计署驻上海特派员办事处,上海 200051)

摘 要: 针对当前虚拟网络映射存在局部区域的节点和链路负载压力过大、节点和相邻链路传输时产生报文抖动和资源浪费等问题,设计一种基于全网负载均衡的虚拟网络映射算法。将节点和相邻链路资源差异性考虑到节点映射中,对 k 最短路径算法的邻接矩阵进行优化,将矩阵转换成反映链路负载均衡的映射矩阵。通过对节点和链路资源的动态调整,分析虚拟网络映射时出现的瓶颈问题。实验结果表明,与随机算法和贪婪算法相比,该算法具有更好的虚拟网络映射率和网络负载均衡性。

关键词: 虚拟网络映射;负载均衡;抖动;网络瓶颈;k 最短路径算法

中文引用格式: 高 斐,陈德礼,洪家军,等. 基于 k 最短路径算法优化与负载均衡的虚拟网络映射机制[J]. 计算机工程,2018,44(5):146-154.

英文引用格式: GAO Fei, CHEN Deli, HONG Jiajun, et al. Virtual Network Mapping Mechanism Based on k Shortest Path Algorithm Optimization and Load Balance[J]. Computer Engineering, 2018, 44(5):146-154.

Virtual Network Mapping Mechanism Based on k Shortest Path Algorithm Optimization and Load Balance

GAO Fei[1], CHEN Deli[1], HONG Jiajun[1], YU Zhi[2], TIAN Tian[3]

(1. College of Information Engineering, Putian University, Putian, Fujian 351100, China;
2. College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China;
3. Shanghai Agency of National Audit Office, Shanghai 200051, China)

【Abstract】 As the existing nodes and links of local areas are overweight, the phenomenon of packet jitter and resource waste generates when packets transmits between nodes and adjacent links. A virtual network embedding algorithm is designed for load balancing of the whole network in this paper. It considers the differences of nodes and adjacent links resources and put the differences into the node embedding process. The k shortest path algorithm of the adjacency matrix is optimized. The matrix is transformed into the new matrix which could reflect link load balancing. Finally, it analyzes the bottleneck problem of virtual network embedding by dynamically adjusting the resources of nodes and links. Experimental results show that the method could achieve better virtual network embedding ratio and network load balancing than randomized algorithm and greedy algorithm.

【Key words】 virtual network mapping; load balance; jitter; network bottleneck; k shortest path algorithm

DOI: 10.19678/j.issn.1000-3428.0048549

0 概述

随着网络技术的快速发展,互联网已经成为现代社会的重要基础设施,越来越多的行业需要借助互联网传递信息实现数据交换和资源共享,而互连网在区域扩大或资源使用方面出现了瓶颈,物理网络资源的稀缺问题慢慢浮现出来。为了提高物理网络资源的使用率,一些研究机构开始利用虚拟网络映射技术提升底层物理网络的资源利用率。虚拟网络映射技术即将物理资源按照节点和链路资源分配给不同的虚拟用户,满足不同用户的服务需求,使得底层物理网络能够最大化地将资源分配给网络用户。虚拟网络技术既减少了底层物理资源的浪费,又减少了重新构建物理网络而投入的新成本,有效解决了互联网发展过程中遇到的僵化问题。目前,国外的研究机构 GENI[1] 对全球网络环境进行创新

基金项目: 国家自然科学基金(61502417);福建省自然科学基金(2016J01759)。

作者简介: 高 斐(1982—),女,讲师、硕士,主研方向为数据挖掘、人工智能;陈德礼、洪家军,副教授、硕士;于 智,博士;田 甜,硕士。

收稿日期: 2017-09-05 **修回日期:** 2017-10-12 **E-mail:** gaofei8237@163.com

性改革,一个大规模的分布式的虚拟网络环境正在逐步完善。本文从全网的节点和链路负载均衡性展开研究,提出一种基于 k 最短路径算法优化和负载均衡的虚拟网络映射机制。

1　相关工作

由于物理网络本身存在节点和链路映射的地理位置限制,资源大小限制,用户对服务质量要求限制,以及 NP 难等问题[2-3],虚拟网络映射技术一直是科研人员长期攻克的研究课题。文献[4]将信任关系和信任度引入到虚拟网络资源分配中,分析网络虚拟化环境中的安全问题,提出基于信任感知的虚拟网络映射算法,该方法从用户的服务安全性来考虑。文献[5]针对网络设备出现能耗浪费,资源利用率不足等问题,提出通过将流量负载不敏感的节点进行资源整合,利用休眠技术实现网络节能,该方法从满足网络资源节能和提升资源使用率的角度来考虑。除此之外,还有从提高物理网络总收益的角度[6],从网络拓扑特性,从降低映射成本[7]为目标来展开研究。它们都是从不同角度对虚拟映射进行了研究。虚拟网络映射过程大致分为 2 种,一种是基于最优化的启发式算法[8-10],它通过满足某个条件使模型参数达到最优,通常可在合理的时间找出问题的最优解。从数学意义上来说,它是研究如何选出最合适的变量取值,使得目标函数实现最大化或最小化的问题[11]。另一种是根据不同的环境建立模型,从而满足虚拟用户的服务质量需求,该方法既可以实现局部最优解,也可以实现全局最优解。本课题研究的内容鉴于后者。

在虚拟映射过程中,按照节点和链路映射的顺序来划分可以分为一步映射和两步映射。一步映射主要指虚拟节点和链路同时映射,一步映射是进行节点和链路的同时映射,但此种算法的时间和空间复杂度较高。而两步映射算法在算法复杂度和映射效果方面有着较好的平衡,因此,本文选用两步映射策略。为了提高虚拟用户的成功映射率,很多文献运用贪婪思想优先考虑资源最大的节点资源进行映射,或者节点和相邻链路资源总和最大的局部区域进行映射[12-13],但此方法没有考虑全网节点和链路负载的均衡性。

如果依赖贪婪思想获得最大资源的节点或链路进行映射,在映射过程中,可能会在固定节点或固定区域进行多次映射,不仅造成其他未映射区域资源的浪费,而且还会使得在该区域周边的链路由于多次映射,使得周围链路带宽快速陷入极小值,本文采用最短路径优先映射算法时,在每次寻找路径时,都要寻找最小链路的那条路径,使得某一条或几条链路带宽无法满足虚拟用户的带宽要求,导致虚拟用户后期映射拒绝率逐渐增高。因此,在考虑节点映射负载能力的同时,还要考虑周边链路映射的负载能力。关于链路映射,很多文献采用穷举法[12-13],找出两点间的所有无环路径,将映射路径中跳数(Hop)引入到映射成本中,由于该方法必须搜索出所有的无环路径,将每条路径所有链路带宽总和除以对应路径跳数,选出比值最小的路径进行映射。此种算法虽然考虑了路径节点的转发成本,但随着网络规模增大,服务对用户的响应时间无法满足用户对服务的时间要求,即 NP 难问题中提到的无法在合理时间内找到有效解。另外,在虚拟网络映射过程中还会存在一个忽略性的问题是当节点和相邻链路间的带宽差异较大时,存在节点和相邻链路资源的浪费以及报文在它们间传输过程中产生的抖动现象。

鉴于以上研究,本文利用 MatLab 平台实现网络拓扑的映射过程,通过邻接矩阵反应网络拓扑的结构性,它不同于虚拟映射研究平台 GM-ITM。此种映射方法可以转化矩阵对应关系,找出适合算法的矩阵数据。

节点基于以下规则进行映射:1)基于节点和相邻链路间的带宽资源差异性,差异越小,该节点越优先映射;2)节点的 CPU 带宽剩余率和相邻链路带宽剩余率总和越大,说明这个区域的节点和链路负载压力越小,该节点越优先选择;3)节点 CPU 剩于带宽占所有节点 CPU 剩余带宽的比例,比例越大表明剩余带宽越大,映射成功率越高,越优先选择。链路映射算法采用 k 最短路径算法(k-shortest)[9,14-15],将该算法映射的邻接矩阵转换成反应链路负载均衡的映射矩阵,减少链路映射负载的差异性,有效平衡网络节点和链路的负载压力。

2　问题描述与网络模型

如何更有效地实现多用户复用同一个物理网络,在映射均衡性的基础上减少网络出现的瓶颈问题,从而有效提高底层网络的映射率。目前需要解决的问题主要有:

1)虚拟网络在使用 k 最短路径算法映射链路过程中容易出现映射范围局部化,局部链路带宽陷入极小值,使得无法保证一条完整路径中每条链路带宽都能满足虚拟用户对链路的带宽资源要求。

2)当节点和相邻链路带宽比例较大时,节点和链路间传输报文时,容易产生报文抖动,报文丢失和资源浪费等现象,即使足够的缓存可以减少报文抖动性,当网络流快速占满路由器整个缓冲区时,造成

数据报文丢失,产生报文重传,使得报文无法持续传输,将会产生报文抖动现象,后者往往被人们忽视。

3)当用户请求的节点或链路带宽资源高于物理节点或链路已有带宽时,无法请求到底层资源,是否可以使一些节点或链路的资源进行释放,等到虚拟用户资源释放后,新到的用户得到资源再映射,而具备什么样特性的节点和链路的资源需要调整是待解决的问题。

4)虚拟用户是否可以降低数据带宽要求,达到数据传输的畅通性,满足基本的传输服务,而不影响服务的质量。

综上所述,以上是目前需要进一步思考和解决的问题。

为了对虚拟网络映射有更深入的了解,首先需要建立虚拟网络模型,对各个参数进行定义和描述。虚拟网络模型如图 1 所示。

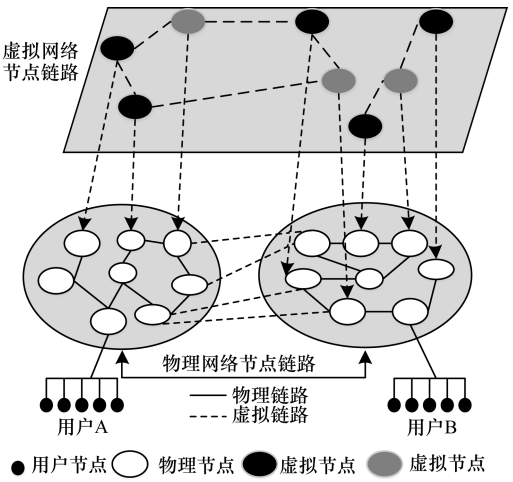

图 1 虚拟网络映射

一个完整的底层物理网络被抽象为一个无向图:

$G^{\alpha}=(N^{\alpha},L^{\alpha})$。将虚拟网络映射到物理网络设置为 $M(V^{\beta}):(V^{\beta},n^{\beta},I^{\beta})\rightarrow(G^{\alpha},N^{\alpha},L^{\alpha})\in(V^{\beta}\leqslant G^{\alpha})\cap(n^{\beta}\leqslant N^{\alpha})\cap(I^{\beta}\leqslant L^{\alpha})$ 即虚拟网络映射的最基本要求如式(1)所示。

$$V^{\beta}\leqslant G^{\alpha},n^{\beta}\leqslant N^{\alpha},l^{\beta}\leqslant L^{\alpha} \tag{1}$$

其中,$V^{\beta}\leqslant G^{\alpha}$ 表示虚拟网络映射的物理节点是底层物理网络拥有节点集合的子集,$n^{\beta}\leqslant N^{\alpha}$ 表示映射的物理节点资源必须满足虚拟用户请求的节点资源大小,$l^{\beta}\leqslant L^{\alpha}$ 表示映射的物理链路资源必须满足虚拟用户请求的链路资源大小。本文要求同一虚拟用户映射的物理节点是不同的网络节点,同一个虚拟用户映射的链路可以映射到相同或不同的物理链路上,每次映射的虚拟链路可以用某一条路径上的各条链路共同分担。具体的符号对照信息如表 1 所示。

表 1 虚拟网络映射符号

含义	符号
物理网络	G^{α}
物理节点	N^{α}
物理链路	L^{α}
物理最短路径	$Path^{\alpha}$
跳数限制	Hop
虚拟网络	V^{β}
虚拟节点	n^{β}
虚拟链路	I^{β}

3 虚拟网络映射机制

3.1 节点映射

随着虚拟网络请求数目增多,物理节点和物理链路上的负载分布将趋于不平衡,尤其采用 k 最短路径算法时,会快速陷入局部链路极小值,使后续用户无法继续映射。为此,本文提出一种基于节点和链路负载均衡以及 k 最短路径算法映射矩阵转换的虚拟网络映射算法。

定义 1 节点和相邻链路带宽差异性。

当节点的处理能力超过了相邻链路的传输能力,节点会出现存储空间浪费,CPU 利用率下降等现象。同样,当相邻链路的传输能力超过了节点的处理能力,会使链路带宽浪费,链路上的报文快速到达节点,当节点缓冲区不足,造成报文丢失和抖动等现象。为此,本文提出第一个指标是选择节点和相邻链路带宽相近的区域进行映射。物理网络拓扑简易图如图 2 所示。

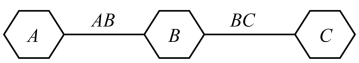

图 2 物理网络拓扑简易图

如图 2 所示,设 m 表示链路 AB 和节点 B 的处理能力的比值,n 表示节点 B 和链路的 BC 处理能力比值。m 和 n 如式(2)所示。

$$m=\frac{L_{AB}(t_i,A,B)}{CPU(N^{B}(t_i))},n=\frac{CPU(N^{B}(t_i))}{L_{BC}(t_i,B,C)} \tag{2}$$

当 $m=n=1$ 时,节点 B、链路 AB 及 BC 处于最佳状态,带宽差异为 0。这里设 t_i 时刻,节点 B 和相邻链路的带宽差异为 $\Omega(N^{B}(t_i))$,$\Omega(N^{B}(t_i))$ 与节点剩余带宽和相邻链路 AB 和 BC 的剩余带宽有关。

1)当 $m>n>1$ 时,$\Omega(N^{B}(t_i))=m+n$。m 或 n 越大,说明节点 B 越容易出现报文排队滞留及数据报文丢失,处理能力取决于链路 BC 的带宽。

2)当 $m<n<1$ 时,$\Omega(N^{B}(t_i))=\frac{1}{m}+\frac{1}{n}$。说明节点 B 受到链路 AB 的制约,链路 BC 受到节点 B 的制约,处理能力取决于链路 AB 的带宽。

3)当 $m<1$ 且 $n>1$ 时,$\Omega(N^{B}(t_i))=n+\frac{1}{m}$。处理能力取决于链路 AB 和 BC 带宽中的最小链路。

4)当 $m>1$ 且 $n<1$ 时,$\Omega(N^{\beta}(t_i))=\frac{1}{n}+m$。处理能力取决于节点 B 的带宽。

通过以上分析,节点和相邻链路带宽差异接近的区域传输效果最好。在选择物理节点进行映射时,尽量选择物理网络节点和相邻链路带宽接近的拓扑区域进行映射,减少节点和相邻链路带宽差异性引起的资源浪费和报文抖动。

定义 2　节点和相邻链路的处理能力差异。

设 t_i 时刻,物理链路 BC 带宽剩余率为 $F(L^{\alpha}(t_i,B,C))$,设链路的起点为 B,终点为 C。$L^{\alpha}(t_i,B,C)$ 为 t_i 时刻链路剩余带宽,$L^{\alpha}(t_0,B,C)$ 为 t_0 时刻链路初始带宽。

第 t_i 时刻物理链路 BC 的带宽剩余率如式(3)所示。

$$F(L^{\alpha}(t_i,B,C))=\frac{L^{\alpha}(t_i,B,C)}{L^{\alpha}(t_0,B,C)} \tag{3}$$

其中,$F(L^{\alpha}(t_i))\in[0,1]$,当 $F(L^{\alpha}(t_i))=0$ 时,说明带宽已耗尽,无法映射。当 $F(L^{\alpha}(t_i))=1$ 时,说明带宽未使用,链路带宽处于最佳状态。

影响节点处理能力的因素与节点的 CPU 带宽和内存空间有关,这里仅考虑 CPU 的带宽。

设第 t_i 时刻物理节点 N^{α} 的 CPU 带宽剩余率,如式(4)所示。

$$F(N^{\alpha}(t_i))=\frac{CPU(N^{\alpha}(t_i))}{CPU(N^{\alpha}(t_0))} \tag{4}$$

其中,$CPU(N^{\alpha}(t_i))$ 为 t_i 时刻物理节点 N^{α} 的 CPU 剩余带宽,$CPU(L^{\alpha}(t_0))$ 为 t_0 时刻物理节点 N^{α} 的 CPU 初始带宽。设节点 N^{α} 的相邻链路共有 q 条。节点 N^{α} 的相邻链路带宽剩余率平均值为 $RF(L^{\alpha}(t_i))$,如式(5)所示。

$$RF(L^{\alpha}(t_i))=\frac{\sum_{q=1}^{q}F(L^{\alpha}(t_i))}{q} \tag{5}$$

$$W(N^{\alpha}(t_i))=\theta_1F(N^{\alpha}(t_i))+\theta_2RF(L^{\alpha}(t_i)) \tag{6}$$

如式(6)所示,$W(N^{\alpha}(t_i))$ 为节点剩余率和它的相邻链路平均剩余率分别乘以相应的系数得到的总和。θ_1 和 θ_2 表示节点和相邻链路处理能力的权重关系。

定义 3　节点 CPU 剩余带宽比例。

设节点 N^{α} 的 CPU 剩余带宽占整个网络节点 CPU 剩余带宽的比例,如式(7)所示。

$$RCPU(N^{\alpha}(t_i))=\frac{CPU(N^{\alpha}(t_i))}{\sum_{s=1}^{s=\alpha}CPU(N^{S}(t_i))} \tag{7}$$

通过以上 3 个指标作为物理节点的选取依据,即物理节点映射的评估模型如式(8)所示,按照 $SNode\_Rank(N^{\alpha}(t_i))$ 从大到小进行排序。其中,μ 表示节点和相邻链路带宽差异指标的权重系数,η 表示节点剩余带宽指标的权重系数,$\Omega(N^{\alpha}(t_i))'$ 表示 $\Omega(N^{\alpha}(t_i))$ 归一化后的数值。

$$SNode\_Rank(N^{\alpha}(t_i))=W(N^{\alpha}(t_i))+\mu\cdot(1-\Omega(N^{\alpha}(t_i))')+\eta\cdot RCPU(N^{\alpha}(t_i))$$
$$0\leqslant\mu\leqslant1,\ 0\leqslant\eta\leqslant1 \tag{8}$$

同样,虚拟网络节点映射顺序如式(9)所示。

$$VNode\_Rank(n^{\beta}(t_i))=\mu\cdot(1-\Omega(n^{\beta}(t_i))')+\eta\cdot RCPU(n^{\beta}(t_i)) \tag{9}$$

由于虚拟网络节点不存在剩余率,因此 $\theta_1=\theta_2=0$。其中,$\Omega(n^{\beta}(t_i))'$ 是 $\Omega(n^{\beta}(t_i))$ 归一化后的数值。虚拟网络节点映射顺序按照 $VNode\_Rank(n^{\beta}(t_i))$ 排序从大到小选取。

3.2　链路映射

常用的链路映射算法是最短路径算法中的基于 k 最短路径映射算法。算法可以按照用户的要求选择合适的 k 值,依次选择第 k 条最短路径。本文设定映射路径经过的节点个数,即跳数(Hop),跳数越多,成本越高。本文将该算法映射的拓扑图邻接对称矩阵进行变换,将邻接对称矩阵转换成反应链路负载均衡的映射矩阵。转换后,可以解决 k 最短路径算法对原始邻接对称矩阵寻找最短路径时出现的快速陷入极小值的问题,该问题会造成路径中的某些链路剩余带宽太低,造成后续的虚拟用户映射率下降。如果将邻接对称矩阵进行变换,使得邻接对称矩阵中的链路带宽数据转换成本文设定的反应链路负载均衡的映射矩阵,无论是虚拟映射率还是链路负载均衡性都具有很好的映射效果。

如式(10)所示,矩阵 $\boldsymbol{A}_1$ 代表无向图的邻接对称矩阵,矩阵 $\boldsymbol{A}_2$ 代表转换后的反应链路负载均衡的链路映射矩阵。n 表示节点个数,$a_{xy}(t_i)$ 表示在 t_i 时刻无向拓扑图邻接矩阵元素,其中,$x\geqslant1$ 且 $x\leqslant n$,$y\geqslant1$ 且 $y\leqslant n$。当 $x\neq y$ 且 $a_{xy}(t_i)=a_{yx}(t_i)$ 时,a_{xy} 表示起点为 x,终点为 y 的链路带宽;当 $x=y$ 时,$a_{xy}(t_i)$ 表示对角线上的元素,即节点 CPU 的带宽。在每次搜索第 k 条最短路径时,使算法映射到经过转换的矩阵 $\boldsymbol{A}_2$ 上。在矩阵 $\boldsymbol{A}_2$ 中,设 $F(a_{xy}(t_i))$ 表示链路 $a_{xy}(t_i)$ 在 t_i 时刻的带宽剩余率,则 $1-F(a_{xy}(t_i))$ 表示链路 a_{xy} 在 t_i 时刻的带宽使用率,对它进行归一化后的数值表示为 $f(a_{xy}(t_i))$。$f(a_{xy}(t_i))$ 越小,说明链路负载压力越小,越容易选择作为映射的对象。另一个指标是链路使用带宽占所有链路使用带宽的比例,设为 $R(a_{xy}(t_i))$,比例越小,说明链路剩余带宽越大,映射成功率越高,越容易作为优先选择映射的对象。将 2 个指标结合在一起作为 k 最短路径算法的

映射矩阵元素。反之,如果单选链路压力大小作为选择对象,则映射次数较少且带宽较小的链路就会被选中作为映射的对象,会造成虚拟映射率下降。如果仅考虑链路剩余带宽大小作为一个指标,它与前面提到的节点映射采用最大资源节点映射的贪婪思想是一样的,虽然暂时可以完成虚拟用户的映射,但没有考虑链路负载压力的均衡性,随着时间推移,链路映射率陷入局部极小值,映射率逐渐下降。因此,在执行 k 最短路径算法之前需要做如下矩阵变换:

$$\boldsymbol{A}_1=\begin{bmatrix} a_{11}(t_i) & a_{12}(t_i) & \cdots & a_{1n}(t_i) \\ a_{21}(t_i) & a_{22}(t_i) & \cdots & a_{2n}(t_i) \\ \vdots & \vdots & & \vdots \\ a_{n1}(t_i) & a_{n2}(t_i) & \cdots & a_{nn}(t_i) \end{bmatrix} \xrightarrow{\text{k 最短路算法的接矩行}}$$

$$\boldsymbol{A}_2=\begin{bmatrix} a_{11}(t_i) & f(a_{12}(t_i))+R(a_{12}(t_i)) & \cdots & f(a_{1n}(t_i))+R(a_{1n}(t_i)) \\ f(a_{21}(t_i))+R(a_{21}(t_i)) & a_{22}(t_i) & \cdots & f(a_{2n}(t_i))+R(a_{2n}(t_i)) \\ \vdots & \vdots & & \vdots \\ f(a_{n1}(t_i))+R(a_{n1}(t_i)) & f(a_{n2}(t_i))+R(a_{n2}(t_i)) & \cdots & a_{nn}(t_i) \end{bmatrix} \tag{10}$$

虚拟网络映射伪代码如下:

输入 底层网络为 $G^{\alpha}(N^{\alpha},L^{\alpha})$,底层网络有 r_1 个节点;虚拟网络请求为 $V(n^{\beta},I^{\beta})$,虚拟网络有 r_2 个节点,$r_2<r_1$

输出 剩余底层网络 $G_S^{\alpha}(N_S^{\alpha},L_S^{\alpha})$ 设虚拟网络有 w 条链路,e 为每个时段用户到达的个数,用户到达个数服从泊松分布函数 $Poissrn(t_i)$,生存时间窗口为向量 $\boldsymbol{T}$

1. % 记录每个时段用户到达时间;%
$t_1,t_2,\cdots,t_{i-1},t_i$;
2. For i = 1:100
3. e < -Poissrn(i);
4. For j = 1:e
5. 计算物理节点指标 $\Omega(N^{\alpha}(t_i))$、$W(N^{\alpha}(t_i))$ 和 RCPU($N^{\alpha}(t_i)$),得出 SNode_Rank($N^{\alpha}(t_i)$);
6. 计算虚拟节点指标 $\Omega(N^{\beta}(t_i))$ 和 RCPU($N^{\beta}(t_i)$),得出 VNode_Rank($n^{\beta}(t_i)$);
7. S1←Descend(VNode_Rank($n^{\beta}(t_i)$));
8. S2←Descend(SNode_Rank($n^{\beta}(t_i)$));
9. If $\{n^{\beta}\leqslant N^{\alpha}\mid n^{\beta}\in S1,N^{\alpha}\in S2\}$;% 物理节点资源满足虚拟节点;%
10. $L^{\alpha}(a,b)\leftarrow S2$;% 保存相邻节点对的链路,S2(a)和S2(b)是相邻的节点;%
11. Else Continue % 否则,进行下一个虚拟用户的请求;%
12. End If
13. $A_1->A_2$;% 邻接矩阵转换;%
14. c = 0;
15. For d = 1:w
16. $A_2(a,b)\xrightarrow{\text{graphkshortestpaths(k)}}$ $Path^{\alpha}(a,b)$;
% 生成物理节点对应的最短路径;%
17. If length($Path^{\alpha}(a,b)$)$\leqslant$Hop&&
$\{1^{\beta}\leqslant L^{\alpha}\mid L^{\alpha}\in Path^{\alpha}(a,b)\}$
18. c + +;
19. Continue;
% 链路资源匹配,继续检测下一条链路;%
20. Else Break;
21. End If
22. End for
23. If c < w
24. Refuse;
% 虚拟网络链路资源不完全匹配,拒绝请求;%
25. Else
% 记录用户服务的结束时间;$f(t_1),f(t_2),\cdots f(t_i)$;%
26. End If
27. If $f(t_i)-t_i<=T$
28. Embed($CPU(n^{\beta})\rightarrow CPU(N^{\alpha}\mid)$);% 虚拟网络请求成功,物理节点映射完成;%
29. $BW(L^{\alpha})\leftarrow BW(L^{\alpha})-BW(1^{\beta})$;% 虚拟网络请求成功,更新链路带宽资源;%
30. Else Refuse;
% 超时,拒绝请求;%
31. End If
32. End For
33. End For

本文将时间分成 100 个时段,每个时段用户按照泊松分布函数产生到达个数,到达函数的时间复杂度为 $O(100\times e)$,虚拟映射算法的时间复杂度为 $O(r_2\times r_1^2)$,虚拟映射算法的空间复杂度为 $O(r_1\times r_2)$。随着节点个数增加,算法的时间和空间复杂度按照多项式曲线增长,算法从时间和空间上是可行的。

4 实验模拟

网络拓扑结构的邻接矩阵采用 Matlab 工具生

成。物理网络节点个数随机生成 30 个 ~ 50 个。虚拟网络请求的节点 CPU 和链路带宽服从 2 ~ 5 的均匀分配。VN 拓扑结构中包含的节点个数为 3 个 ~ 7 个,VN 中节点的度数服从 0 ~ 5。图 3 是基于节点度数的网络拓扑单次案例图,图中有 30 个物理节点,圆点的大小反映物理节点度数的多少。基于节点带宽大小的网络拓扑如图 4 所示。

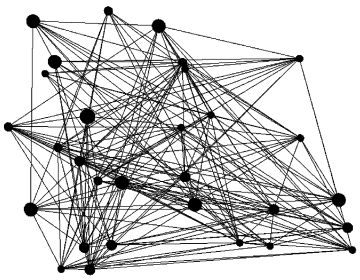

图 3　基于节点度数大小的网络拓扑单次案例

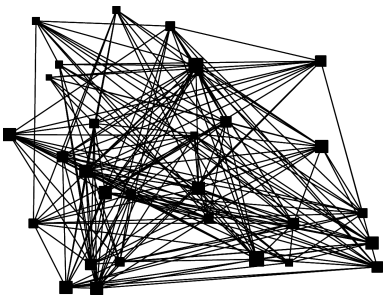

图 4　基于节点带宽大小的网络拓扑单次案例

方形点的大小反映节点带宽资源的多少,这两个图是同一时刻基于不同角度的拓扑案例图,通过两个图的对比可以提供网络不同区域的节点剩余的带宽分布情况。本文把实验完成的过程划分成 1 个 ~ 100 个时段,每个时段记录一次实验数据。采用间隔抽样法,从 100 个时段中抽取 20 个采样点,每隔5 个时段抽取 1 次作为样本数据。设生存时间窗口足够大。对参数进行赋值:$\eta = 1, \mu = 1, \theta_1 = 1, \theta_2 = 1$。将本文算法与其他两种算法进行比较,一种是随机算法,它对节点映射采用随机抽取方法,链路映射采用 k 最短路径算法;另一种算法是文献[9,13]基于贪婪思想的局部节点和相邻链路带宽最大化的节点映射算法与 k 最短路径映射算法相结合的虚拟链路映射算法。

为了更好地说明本文算法在映射过程优于其他映射算法,使用以下指标进行衡量。

1)节点和链路负载压力匹配度

为了体现节点和链路负载均衡性,引入负载压力匹配度 C,计算公式如下:

$$C = \sqrt{\frac{\sum_{\alpha=1}^{m_1}\left(F(L_\alpha(t_i)) - \overline{F(L_\alpha(t_i))}^2\right)}{m_1}} + \sqrt{\frac{\sum_{\alpha=1}^{n_1}\left(F(N^\alpha(t_i)) - \overline{F(N^\alpha(t_i))}^2\right)}{n_1}} \tag{11}$$

其中,m_1 为链路个数,n_1 为节点个数。负载压力匹配度 C 越大,说明节点与节点间,链路与链路间负载均衡效果越好,节点和链路映射负载差异越小,网络平衡性越好,映射率越高。如图 5 所示,从曲线变化趋势看,本文算法使得物理网络整体负载差异变化最稳定,差异最小。

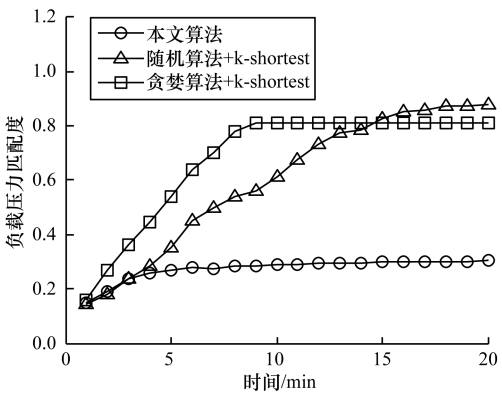

图 5　负载压力匹配度

2)最大节点负载压力和最大链路负载压力

最大节点压力和最大链路压力反映网络哪些节点和链路遇到瓶颈问题。最大节点压力和最大链路压力的变化反映整个网络是否处于资源短缺状态。图 6 是 3 种算法在映射过程中最大节点压力变化曲线。图 7 是 3 种算法在映射过程中最大链路压力变化曲线。可以看出,本文映射算法中的最大节点和最大链路压力比较稳定,一直小于其他算法,可见本文算法的负载均衡性较好。而其他 2 种算法由于没有考虑节点负载的均衡性,以及对链路映射矩阵未进行更好的优化,使它们的节点和链路很快出现压力过大,在后期映射中,全网负载压力不均衡,虚拟映射率下降。表 2 反映 3 种算法负载参数对应的数据,从平均负载压力匹配度和节点链路的负载压力上看,本文算法都优于其他 2 种算法,虽然随机算法最大压力节点的最大值和最小值小于本文算法,但整体执行过程的平均压力不如本文算法,而本文对基于最短路径算法的邻接矩阵优化,链路的负载均衡性都优于其他 2 种算法。

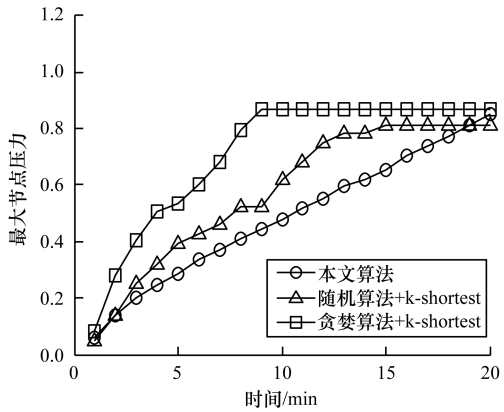

图 6 物理网络的最大节点压力

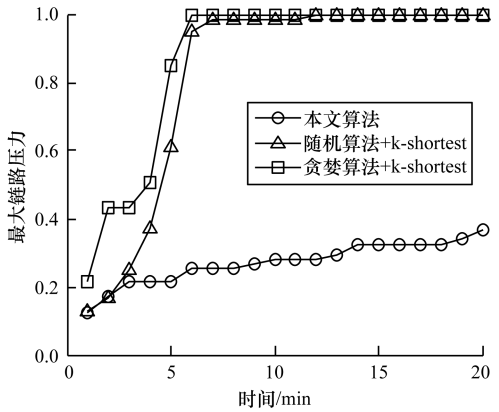

图 7 物理网络的最大链路压力

表 2 3 种算法负载平衡性的比较

算法	负载压力匹配度			最大节点压力			最大链路压力		
	最大值	最小值	平均值	最大值	最小值	平均值	最大值	最小值	平均值
本文算法	0.305 5	0.146 9	0.278 2	0.889 7	0.058 8	0.504 7	0.365 9	0.122 0	0.278 3
随机算法 + k-shortest	0.880 7	0.145 4	0.614 8	0.808 3	0.049 4	0.592 5	1.000 0	0.129 0	0.835 0
贪婪算法 + k-shortest	0.811 3	0.159 3	0.694 3	0.866 7	0.084 0	0.729 7	1.000 0	0.216 2	0.892 5

3)收益

网络映射的主要目标是物理网络在消耗相对较小资源成本和满足用户服务质量的前提下,映射尽量多的虚拟网络用户。本文分 2 种情况来考虑映射收益[16]:一个指标只考虑虚拟用户映射总收益,它包括虚拟用户节点和链路的带宽资源的请求大小;另一个指标将成本引入到映射收益中,即映射成本收益。映射成本反映虚拟映射在实际网络条件的情况下,映射收益和成本间的关系。映射成本主要体现在链路映射的路径长度,路径越长,占用的链路带宽越多。

设总收益为 $R(V(t))$,它的定义如式(12)所示。λ_1 和 λ_2 分别是调整节点收益和链路收益的权重参数,$\sum_{e_v\in E_V} W(e_v(t))$ 表示虚拟链路的收益,$\sum_{v_v\in V_V} C(v_v(t))$ 表示虚拟节点的收益。

$$R(V(t)) = \lim_{t\to|t_i-t_{i-1}|}\left[\lambda_1\sum_{e_v\in E_V}W(e_v(t)) + \lambda_2\sum_{v_v\in V_V}C(v_v(t))\right] \tag{12}$$

$$RC(V(t)) = \lim_{t\to|t_i-t_{i-1}|}\frac{R(V(t))}{\alpha_1 Cost(V(t))} \tag{13}$$

其中,$RC(V(t))$是总收益和成本的比例关系。物理网络接受虚拟请求所消耗的成本如式(14)所示。

$$Cost(V(t)) = \alpha_2\cdot\sum_{e_v\in l_\alpha}\sum_{e_v\in l_\alpha} length(e_v(t),Hop)\cdot W(e_v(t)) + \alpha_3\cdot\sum_{v\in N^\alpha}C(V_v(t)) \tag{14}$$

式(14)中的 $length(e_v(t),Hop)$ 表示分配给虚拟链路的物理路径中经过节点的个数。$W(e_v(t))$ 表示分配给该链路的路径上的占用链路总带宽。$C(V_v(t))$ 表示底层网络节点为虚拟节点分配的 CPU 带宽。当虚拟链路映射物理链路时,分配的路径中经过的节点个数越多,映射成本越大。图 8 表示本文算法分别与随机算法和贪婪算法在总收益上的比较。

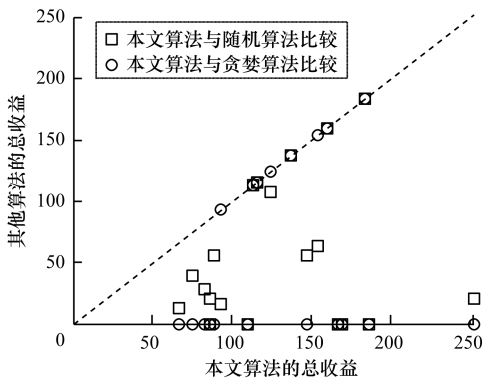

图 8 2 种情况下的总收益比较

点在对角线下面说明本文算法总收益更大,在对角线上面说明其他 2 种算法总收益更大,对角线上的点表示本文算法与其他算法总收益相同。图 7 本文算法总收益总体上要高于其他 2 种算法。图 9 是本文算法总收益与收益成本间的关系。黑色线条

是 $RC(V(t))$,白色线条是 $R(V(t))$。这里设初始参数为 $\alpha_1=0.1$,$\alpha_2=\alpha_3=1$。

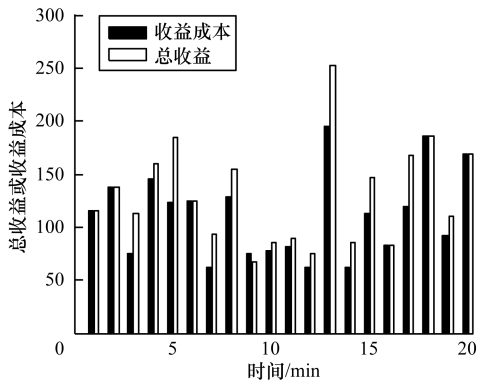

图 9　本文算法的总收益和收益成本的关系

4)虚拟网络映射率(VMR)

设虚拟用户成功映射个数为 $Receive(num(V(t)))$ 与虚拟用户请求个数 $Request(num(V(t)))$ 的比值关系如式(15)所示。

$$VMR(t)=1\underset{t\to|t_i-t_{i-1}|}{im}\frac{Receivenum(V(t))}{Requestnum(V(t))} \tag{15}$$

从图 10 可以看出,本文算法高于其他 2 种算法的虚拟网络映射率。

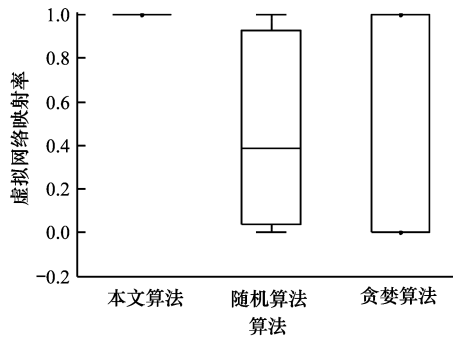

图 10　3 种算法的虚拟网络映射率

表 3 给出了 3 种算法虚拟映射率的对比数据,它通过分析 100 个时段 3 种算法对等量和等结构的虚拟用户进行映射,本文算法的虚拟用户映射率相比随机算法提高了 0.48,相比贪婪算法提高了 0.61。随机算法比贪婪算法效果更好,因为每次映射物理节点时,节点的选取是随机的,负载平衡性比贪婪算法好,而贪婪算法中的节点映射会选取节点和相邻链路带宽资源最大的区域进行映射,可能存在一个带宽较大的区域被映射多次,使得周围区域中的链路资源基本耗尽。而随机算法的节点选择的区域随机性较强,因此,随机算法的节点映射的区域变化性比基于贪婪算法的节点映射区域变化性大。但随机算法每次随机选取的节点资源不一定保证满足虚拟用户资源要求,同时链路映射没有考虑负载均衡性,造成后期虚拟网络映射率下降。

表 3　3 种算法的虚拟网络映射率比较

算法	最大值	最小值	平均值
本文算法	1	1	1.000 0
随机算法 + k-shortest	1	0	0.512 2
贪婪算法 + k-shortest	1	0	0.384 2

5)资源调整

当链路资源不足时,如果将虚拟节点迁移到其他节点上,随着网络规模增加会出现以下问题,迁移距离越大,迁移成本越高。如果把虚拟节点只迁移到相邻节点,由于节点周边链路资源可能稀缺,改变的差异较小,映射率提高的效果较轻微。因此,本文满足路径跳数限制的基础上调整 k 值,使路径重新规划。如图 11 所示,白色线条表示 k 为 1 时的虚拟映射率。黑色线条是 k 为 8 时的虚拟映射率。当虚拟链路不满足要求时,k 值增加到 8。从第 4 次到第 7 次抽样时,即第 20 到第 35 个时刻,可以看出,通过调整 k 值提高了虚拟网络映射率。

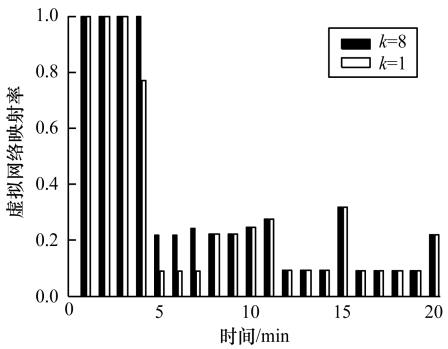

图 11　调整 k 值的虚拟网络映射率比较

当节点资源不足时,本文采用节点资源的动态调整。首先找出瓶颈节点,然后对其进行资源释放。瓶颈节点主要指最大压力节点和最大介数节点。通过实验结论得出释放最大介数节点上的资源,效果更好。介数[17]很高的节点或链路对于网络起着重要的支撑作用。节点介数定义为拓扑中经过该节点最短路径数目占所有节点最短路径总数的比例。如果该节点介数很高并且出现带宽资源短缺,则链接它的子网将无法交换数据,造成经过该节点的链路映射失败。在没有释放资源之前始终处于僵持状态。如图 12 所示,图 12 中用 2 种算法进行比较。黑色线条表示最大介数节点资源释放的虚拟映射率,白色线条表示最大压力节点资源释放的虚拟映

射率。假设 2 种节点释放的资源大小相同。在大多数时段,通过释放最大介数节点资源的虚拟映射率比释放最大压力节点资源的虚拟映射率效果更好,虚拟网络映射率提升的更快。

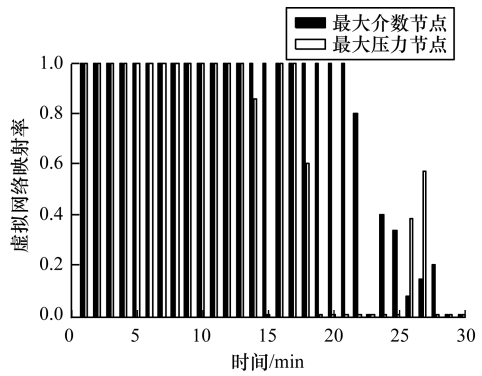

图 12 最大介数节点和最大压力节点比较

5 结束语

本文提出一种结合节点和链路资源负载均衡的高效映射方案。通过对邻接矩阵进行变换,将原来的邻接矩阵转换成反映链路负载均衡的映射矩阵,有效提高虚拟映射率。对网络出现的资源不足采取了对应的资源调整措施,即最大介数节点的资源释放和基于 k 值调整的链路重规划。下一步将研究节点和链路资源的动态调整,针对不同的虚拟用户进行定义,考虑哪些虚拟用户在降低数据带宽要求的前提下,可以实现数据的基本传输,满足基本的服务质量,从而进一步提高虚拟网络映射率。

参考文献

[1] MARK B,JEFFREY S C,LAWRENCE L,et al. GENI:a federated testbed for innovative network experiments[J]. Computer Networks,2014,61(14):5-23.

[2] 蔡志平,刘 强,吕 品,等. 虚拟网络映射模型及其优化算法[J]. 软件学报,2012,23(4):864-877.

[3] LEONARDO R B,RODRIGO R O,LUCIANA S B,et al. Security-aware optimal resource allocation for virtual network embedding[C]//Proceedings of the 8th International Conference on Network and Service Management. Washington D. C.,USA:IEEE Press,2012:378-384.

[4] 龚水清,陈 靖,黄聪会. 信任感知的安全虚拟网络映射算法[J]. 通信学报,2015,36(11):180-189.

[5] 陈晓华,李春芝,陈良育,等. 主动休眠节点链路的高效节能虚拟网络映射[J]. 软件学报,2014,25(7):1416-1431.

[6] HSU W H,SHIEH Y P. Virtual network mapping algorithm in the cloud infrastructure[J]. Journal of Network and Computer Applications,2013,36(6):1724-1734.

[7] FAJJARI I,AITSAADI N,PIORO M,et al. A new virtual network static embedding strategy within the cloud's private backbone network[J]. Computer Network,2014,62:69-88.

[8] 黄彬彬,林荣恒,彭 凯. 基于粒子群优化的负载均衡的虚拟网络映射[J]. 电子与信息学报,2013,35(7):1753-1759.

[9] ZHANG Min,TANG Xi Jie. Hop-limit path mapping algorithm for virtual network embedding[J]. Wireless Personal Communications,2016,95(3):1-16.

[10] 李小玲,郭长国,李小勇. 一种基于约束优化的虚拟网络映射方法[J]. 计算机研究与发展,2012,49(8):1601-1610.

[11] 徐冬冬,郑淑丽,曹 敏. 基于 openflow 网络的虚拟网络映射研究[J]. 合肥工业大学学报(自然科学版),2016,39(10):1352-1356.

[12] JIANG Liu,TAO Huang,CHEN Jianya,et al. A new algorithm based on the proximity principle for the virtual network embedding problem[J]. Journal of Zhejiang University-SCIENCE C,2011,12(11):910-918.

[13] 董永彬,吕光宏,李立龙. 基于节点分割的两阶段虚拟网络映射算法[J]. 四川大学学报(自然科学版),2005,52(2):287-292.

[14] EPPSTEIN D. Finding the k shortest paths[J]. SIAM Journal on Computing,1994,28(2):652-673.

[15] YU M,YI Y,REXFORD J,et al. Rethinking virtual network embedding:substrate support for path splitting and migration[J]. Computer Communication REVIEW,2008,38(2):17-29.

[16] 张 翔. 成本与能效优化的虚拟网络映射算法研究[D]. 南京:南京邮电大学,2013.

[17] FREEMAN L C. A set of measures of centrality based upon betweenness[J]. Sociometry,1977,40(1):35-41.

编辑 刘 冰