doi: 10.3969/j.issn.1005-3026.2017.03.007

支持流量感知的软件定义网络高效路由方案

祝烈煌, 张琼宇, 沈 蒙, 王明钟
(北京理工大学 计算机学院, 北京 100081)

摘 要: 针对目前软件定义网络的实现方案中,路由策略所采取的最短路径模型无法保证网络信息交付延迟最小的问题,以信息交付延迟作为路由方案效率的衡量指标,提出了一种支持流量感知的高效路由方案. 基于全网流量信息,本方案采用多元优化方案综合考虑转发路径的可用带宽、丢包率、延迟、交换机无效服务率以及路由长度,做出路由决策以提高路由效率. 实验证明,与目前 OpenFlow 框架下采用的路由方案相比,本方案可以将数据交付效率提高 90% . 此外,本方案有利于维护网络负载均衡.

关 键 词: 软件定义网络;路由;流量优化;路由模型;传输延迟
中图分类号: TP 393.0 **文献标志码**: A **文章编号**: 1005-3026(2017)03-0335-06

Efficient Traffic-Aware Routing Scheme for Software Defined Networks

ZHU Lie-huang, *ZHANG Qiong-yu*, *SHEN Meng*, *WANG Ming-zhong*
(School of Computer Science & Technology, Beijing Institute of Technology, Beijing 100081, China. Corresponding author: SHEN Meng, E-mail: shenmeng@bit.edu.cn)

Abstract: Routing decision in software defined networks (SDN), which was based on the shortest path model, was unable to guarantee to minimize the delivery delay in the routing path. An efficient traffic-aware routing scheme was proposed, in which the information delivery latency was taken as the measurement of the routing efficiency. Based on the network-wide traffic information, various traffic indicators were considered by this scheme, including available bandwidth, packet loss probability, delay, switches invalidity probability, and routing path length, to find out the optimal routing path to enhance the efficiency of data delivery in networks. Experiments show that the scheme outperforms current scheme applied in OpenFlow framework with up to 90% improvement on efficiency. What's more, the scheme is helpful to maintain the load balance of the networks.

Key words: software defined networks(SDN); routing; traffic optimization; routing module; delivery delay

软件定义网络(SDN)[1-2]被认为是解决企业网络和数据中心网络规模及复杂度扩张的最有前景的技术. SDN 将控制平面与数据平面分离,通过控制器对网络实施逻辑上的集中控制.

SDN 中,控制器的全局视角和全网流量信息被广泛应用于路由优化和提高网络服务效率. 例如,SDN 边界网关协议(BGP)[3]通过实现不同域之间迅速及时的信息交换,提高了网络服务效率; SDN 基于内容的路由协议通过准确快速地更新发布/订阅消息提高网络服务效率. 然而,现有研究均采用传统网络中常见的路由方案(如最短路径方案)作为路由策略,没有关注路由方案本身的效率改进.

现有路由方案大多数仅仅关注单一流量指标的优化,比如最小化转发路径长度或者最大化带宽. 它们本质上都是基于网络拓扑的最短路径问题[4]. 然而,优化单一指标并不能找出最高效的转发路径. 换言之,最短路径问题模型并不适合高

收稿日期: 2015-10-28
基金项目: 北京市自然科学基金资助项目(4164098); 国家自然科学基金资助项目(61602039,61272512,61300177).
作者简介: 祝烈煌(1976-),男,浙江衢州人,北京理工大学教授.

效路由方案,尤其是在动态网络环境中.

因此,基于 SDN 环境中的全网流量信息,本文提出了一种支持流量感知的软件定义网络高效路由方案,提高数据传输的速度和成功率. 该方案试图找出一条最优的转发路径,实现多种流量因素整体优化,包括可用带宽、丢包率、延迟、交换机无效服务率以及路由长度.

1 相关工作

软件定义网络(SDN)是一种网络架构革新[1-2],它将网络控制从数据转发中分离出来. 在 SDN 中,网络对于上层应用层而言是一个逻辑上的或者虚拟的实体,对于用户而言则是可编程的.

目前国内外已有大量的针对 SDN 的研究. Bennesby 等[3]提出了 Inter - AS 路由组件,用于优化边界网关协议(BGP),使其能够更好地应用于 SDN 环境,同时解决了一些 BGP 在传统网络中存在的问题. Li 等[4]关注 SDN 的能量节约问题,Karl 等[5]、Egilmez 等[6-8]基于 OpenFlow 框架[9]提出了一种最优多媒体路由方案,可以改善流媒体服务中的用户体验. Adrichem 等[10]关注 SDN 网络的路由可靠性问题,提出了故障快速恢复机制. 上述研究基本都需要利用全局视角和全网流量信息[11]. 尽管它们提高了某些特殊应用或者传输层协议的网络服务效率,但是它们均未提高下层路由方案的效率,本文尝试解决这一问题.

2 支持流量感知的高效路由方案

本节将详细地介绍并解释支持流量感知的高效路由方案. 首先,给出该方案的系统架构;然后介绍该方案中涉及到的优化指标;最后,陈述该方案的算法设计细节.

2.1 系统架构

本方案的系统架构如图 1 所示,基于 OpenFlow 框架,交换机将维护本地流表以存储控制器分配的转发规则[12]. 交换机需要转发数据包时,首先依据包头信息在本地流表寻找合适的转发规则,如果找到了就依照规则进行数据转发;否则,交换机将向控制器(controller)发出请求. 交换机获得来自控制器的转发规则后,按照规则转发数据包. 由此过程可知,SDN 网络中交换机仅负责数据转发,而不需要执行任何路由算法. 因此,只需实现高效路由方案并将其应用于控制器即可.

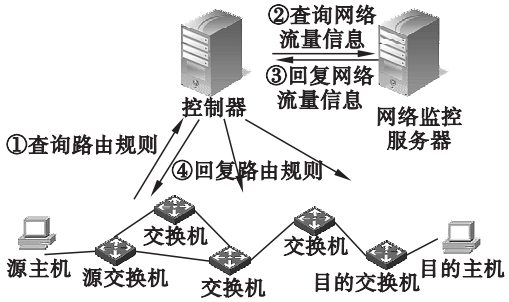

图 1 系统架构

Fig. 1 System architecture

本方案中,控制器将周期性地从网络监控服务器中获取流量信息并存储在本地. 当控制器接到路由规则的请求时,控制器从本地内存中获取当前网络的流量信息. 然后,基于流量信息,依据本文所提出的路由算法获取最优的路由规则. 随后,控制器从路由规则中分离出转发路径上每个交换机的转发规则,并将规则发送到相应的交换机. 本文基于流量信息实现一种高效的路由算法,帮助控制器找到最优转发路径,进而提高数据传输效率.

2.2 优化指标

转发路径由链路和网络节点组成. 因此,转发路径的性能取决于链路和网络节点的性能[13-14]. 就链路而言,流量相关的性能主要有延迟、丢包率和可用带宽(即吞吐量). 至于网络节点(即交换机),其无效服务率可以用于性能衡量. 这里无效服务是指交换机由于设备故障、缓存不足、处理能力不足或者其他原因导致无法在预期时间内完成数据传输,其实质反映了转发设备所承受的压力.

网络服务效率取决于信息交付的速度和成功率. 在上述指标中,延迟、可用带宽影响着信息传输的速度,进而影响转发路径的信息交付速度. 而丢包率和交换机无效服务率则决定了转发路径的信息交付的成功率.

2.3 支持流量感知的高效路由算法

本文提出的支持流量感知的高效路由方案通过综合优化多个流量指标提高网络服务效率,本质上是采用多目标优化策略解决路由问题.

本文采用效用函数来衡量转发路径 p 的效率,即转发延迟. 假设一条路径主要流量参数如下:可用带宽为 $B(p)$,丢包率为 $L(p)$,交换机无效服务率为 $C(p)$,传播延迟为 $D(p)$,路由长度为 $N(p)$. 那么在数据传输过程中,能够使用的带宽平均为 $B(p)\times(1-C(p))\times(1-L(p))$. 假设转发数据大小为 M,那么数据将被分成

$\frac{M}{B(p)\times(1-C(p))\times(1-L(p))}$批传输. 考虑最糟糕的情况,即每一批数据逐个发送,则数据 M 的传输延迟可以被估算为 $\frac{M\times D(p)\times N(p)}{B(p)\times(1-C(p))\times(1-L(p))}$,即这实际上是 M 传输延迟的最大值. 本文用传输 1 个单位的数据量的用时来衡量传输效率,则转发路径 p 的传输效率为

$$f(p)=\frac{B(p)\times(1-C(p))\times(1-L(p))}{N(p)\times D(p)}.$$

这里所有变量的值都经过归一化处理为无量纲变量.

假设有网络 N,p 是其中一条转发路径. 令 PL 表示路径 p 上的链路集合,S 表示路径 p 上的交换机集合,TL 表示网络 N 中的链路集合,TS 表示网络 N 中交换机集合,则效用函数中各个变量计算如下:

1) 交换机无效服务率 $C(p)$ 表示路径 p 上至少有一个交换机无法提供有效服务的概率,计算如下:

$$C(p)=1-\prod_{i\in S}(1-c_i).$$

其中 c_i 表示 S 中第 i 个交换机的无效服务率. 那么 $\prod_{i\in S}(1-c_i)$ 就是链路 p 上所有交换机均可以提供有效服务的概率.

2) $L(p)$ 即路径 p 的丢包率,计算如下:

$$L(p)=1-\prod_{i\in \mathrm{PL}}(1-l_i).$$

其中 l_i 表示 PL 中第 i 条链路的丢包率,因此,$\prod_{i\in \mathrm{PL}}(1-l_i)$ 可以表示链路 p 上所有链路都不丢包的概率.

3) $D(p)$ 即路径 p 的延迟,计算如下:

$$D(p)=\frac{\sum_{i\in \mathrm{PL}} d_i}{\sum_{j\in \mathrm{TL}} d_j}.$$

路径 p 的延迟的理论值是 p 上所有链路的延迟之和,因此,路径 p 的延迟为 $\sum_{i\in \mathrm{PL}} d_i$,这里 d_i 表示 PL 中第 i 条链路的延迟. 此外,网络 N 中的延迟最大值是网络中所有的链路延迟之和,用这个最大值作为分母对延迟进行归一化处理. 公式中 d_j 是 TL 中第 j 条链路的延迟.

4) $B(p)$ 即路径 p 的可用带宽,计算如下:

$$B(p)=\frac{\min_{i\in L} b_i}{\max_{j\in \mathrm{TL}} b_j}.$$

可用带宽即单位时间内(通常是 1s)通过链路上某一点的数据比特数. 因此,路径 p 的可用带宽取决于路径上各链路可用带宽的最小值. 在上述公式中,b_i 是 PL 中第 i 条链路的可用带宽,b_j 是 TL 中第 j 条链路的可用带宽. 此外,相对于其他变量,带宽的取值往往比较大,为了避免带宽过多地影响链路性能的评估,使用网络 N 中所有链路可用带宽的最大值作为分母对带宽进行归一化处理.

5) $N(p)$ 即路径 p 的长度,计算如下:

$$N(p)=\frac{|S|}{|\mathrm{TS}|}.$$

p 的实际长度是 $|S|$,即路径 p 中交换机的数量,使用网络 N 中路径长度的最大值(由网络 N 中交换机数量总和估算)进行归一化处理.

基于上述效用函数,本文基于分支限界策略设计了路由算法,用于找出源主机到目的主机之间能够最大化效用函数的最优转发路径.

由现有的 OpenFlow 控制器提供的转发路径 S_{orig} 的效用值 $f(S_{\mathrm{orig}})$ 作为算法的初始界限 bound. 如果算法无法找到更优的转发路径,那么初始的转发路径 S_{orig} 将被选作最优路径 S_{opt}.

该算法可分为 3 个步骤:

步骤 1(初始化):令候选解集 C_{set} 为空,将原始的 OpenFlow 控制器提供的转发路径 S_{orig} 作为最优解. 同时,令当前解 S_{cur} 是一条仅包含源交换机的路径.

步骤 2(扩展候选解集):设交换机 L_n 是当前解路径上的最后一个交换机. 如果 L_n 就是目的交换机,那么将当前解 S_{cur} 作为最优解 S_{opt}. 否则,令所有与 L_n 连接的交换机形成一个交换机集合,对集合中的每个交换机作如下处理:将其添加到当前解中,形成一个新的解 S_{temp},如果这个新的解的效用函数 $f(S_{\mathrm{temp}})$ 值比界限值 bound 大,则将这个新解 S_{temp} 添加到候选解集 C_{set} 中,否则不作处理.

步骤 3(更新当前解):在候选解集 C_{set} 中找出效用值最大的候选解,将其作为当前解 S_{cur}.

重复步骤 2 和步骤 3,直到步骤 2 中交换机 L_n 是目的交换机,或者候选解集 C_{set} 变成空集,然后返回最优解 S_{opt}.

在一个 n 节点的网络中,路由算法有两重循环,外层循环次数由路径长度决定,记为 $k_1 n$,其中 k_1 为一个小于 1 的常数,两个内层循环的循环次数表示为 $k_2 n$,其中 k_2 为常数. 因此,在 1 个由 n 个节点组成的网络中,路由算法的时间复杂度为 $O(n^2)$.

3 性能测试

为了评估支持流量感知的高效路由算法的效率,本文使用 Mininet 作为数据平台、Floodlight 作为控制器搭建一个 SDN 仿真环境. 实验运行环境是 Intel Core – i5 处理器,4 GB 内存.

在实验中,本文首先在 SDN 环境中将本方案与 SDN 架构下的路由算法(具体实现由 Floodlight 控制器提供,后文中称之为"传统路由方案")进行对比. 此外,对于支持流量的路由算法本身,选择了一种传统网络中同样支持流量感知的动态多径路由协议 SPEF(shortest-paths penalizing exponential flow-splitting[15])与本方案进行对比.

3.1 与 SDN 路由方案对比

3.1.1 仿真模型

在 SDN 环境下,与 SDN 路由方案进行对比,网络拓扑结构选自 Topology Zoo[16] 数据集. 为了对不同的网络环境进行测试,本文采用拓扑规模和交换机无效服务率作为网络环境参数. 对于网络拓扑,选择拥有 61 台交换机的 GARR 作为小规模网络. 对于交换机无效服务率(记作 $C(N)$),由于网络环境越糟糕,网络的数据传输压力越大,从而交换机的压力就越大,导致其无效服务率将比较高. 因此,在网络环境良好时,令 $0 < C(N) < 5\%$;而在网络环境不佳时,令 $5\% < C(N) < 10\%$. 在仿真实现时,令设备以 $C(N)$ 的概率断开连接,然后再重新与控制器建立连接.

基于选自 Topology Zoo 数据集的网络拓扑,本文随机分配网络流量信息,包括每条链路的可用带宽、延迟、丢包率以及每个交换机的无效服务率. 分配规则如下:每条链路的可用带宽取值范围是 0 ~ 5 MB/s;每条链路的延迟取值范围为 10 ~ 30 ms;每条链路的丢包率取值范围为 0 ~ 5%;每个交换机的无效服务率取值范围如上文所述. 所有参数都满足独立随机分布.

在 GARR 中,本文执行两种仿真实验:一种是衡量最短路径长度不同时,文件传输效率的提升;另一种是衡量不同大小文件传输效率的提升.

在上述两种实验中,传输效率的提升(记为 Δ)用传输用时减少百分比来衡量:

$$\Delta = \frac{t_{\text{ori}} - t_{\text{new}}}{t_{\text{ori}}}.$$

其中:t_{ori} 表示传统路由方案下平均传输用时;t_{new} 表示本文的路由方案下平均传输用时.

3.1.2 仿真结果

3.1.2.1 最短路径长度不同时的文件传输

在 GARR 实验中,最短路径长度取值范围 2 ~ 6. 对于每一种长度,选择 5 组不同的源主机与目的主机对,进行 10 次重复实验.

图 2a 和图 2b 是 GARR 的实验结果. 图 2a 是网络环境良好(即 $0 < C(N) < 5\%$)时的结果,而图 2b 是网络环境不佳(即 $5\% < C(N) < 10\%$)时的结果. 图中散列点是每轮实验的具体传输效率提高比例,折线是不同最短路径长度传输效率提高比例的均值. 由图 2 可知,不同网络环境中,传输效率提高比例均为正,并且传输效率平均提高了 60%,最高可达到 80% 的提升.

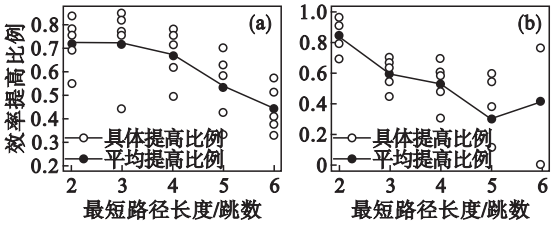

图 2 GARR 中不同最短路径长度效率提升

Fig. 2 Efficiency improvements with different shortest routing path lengths in GARR

(a)—$0 < C(N) < 5\%$;(b)—$5\% < C(N) < 10\%$.

为了更加详细地展示实验结果,本文从上述实验中不同的最短路径长度的源主机与目的主机对中随机选择一对出来,画出了它们的传输时间分布. 图 3a 和图 3b 是文件传输用时分布的蜡烛图. 对于不同的源主机与目的主机对,文件传输用时差异较大,因此,图 3 中文件传输用时取以 10 为底的对数.

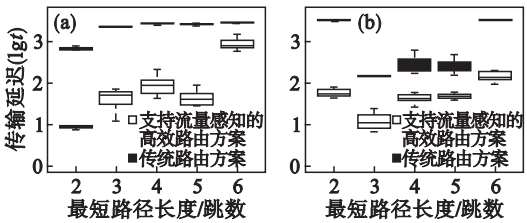

图 3 GARR 中不同最短路径长度传输时间分布

Fig. 3 Time distribution with different shortest routing path lengths in GARR

(a)—$0 < C(N) < 5\%$;(b)—$5\% < C(N) < 10\%$.

蜡烛图的 5 个值从上到下分别是最大值、3/4 值、中位数、1/4 值、最小值. 比较发现,支持流量感知的高效路由算法的蜡烛图,整体位于传统方案下方. 也就是说,在 GARR 中,采用支持流量感知的高效路由方案,在不同最短路径上传输相同文件效率高于传统方案.

3.1.2.2　不同大小文件传输

实验中选择 5 组不同的源主机与目的主机对，使其具有相同的最短路径长度. 在 GARR 中该长度取 4. 对每个源主机与目的主机对，从源主机依次发送大小为 256 KB，512 KB，1 MB 和 1.5 MB 的文件到目的主机，每次实验重复 10 次.

在网络环境良好（即 $0 < C(N) < 5\%$）时的实验结果如图 4a 所示，而网络环境不佳（即 $5\% < C(N) < 10\%$）时的实验结果如图 4b 所示. 图中散列点是每轮实验的具体传输效率提高比例，折线是不同大小的文件传输效率提高比例的均值. 由图 4 可知，不同网络环境中，传输效率提高比例均为正，并且在网络环境良好的情况下，传输效率平均提高了 55%，最高可有 75% 的提升，而在网络环境不佳时传输效率平均提高了 30%，最高时为 65%.

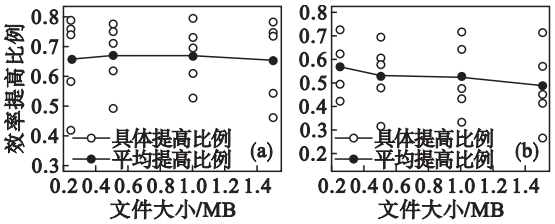

图 4　GARR 中不同大小文件传输效率提升

Fig. 4　Efficiency improvements with different file sizes in GARR

(a)—$0 < C(N) < 5\%$；(b)—$5\% < C(N) < 10\%$.

不同网络环境下，文件传输时间分布分别如图 5a 和图 5b 所示. 实验中同样对文件传输时间进行以 10 为底取对数操作.

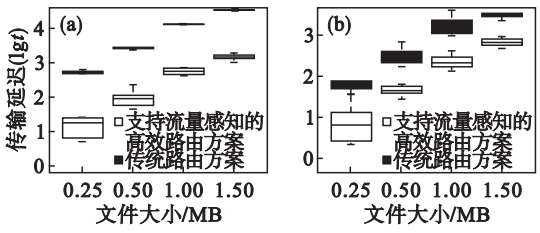

图 5　GARR 中不同大小文件传输时间分布

Fig. 5　Time distribution with different file sizes in GARR

(a)—$0 < C(N) < 5\%$；(b)—$5\% < C(N) < 10\%$.

通过对比，可以发现支持流量感知的高效路由算法的蜡烛图整体位于传统方案下方. 因此，使用支持流量感知的高效路由方案相比传统方案，对于不同大小的文件的传输效率均有所提高.

3.2　与 SPEF 对比

3.2.1　仿真环境

本实验中采用 GARR 网络作为拓扑结构，为了与 SPEF 方案对比，实验设置链路带宽在 1 ~ 10 GB/s之间随机分布，链路丢包率取值范围为 0 ~ 5%，链路延迟在 10 ~ 30 ms 间随机取值，设备无效服务率设为 0 ~ 5%. 同时在网络中以 30% 的概率在任意两点间随机产生一个取值在 100 ~ 500 MB 间的数据流. 最后统计不同路由方案下数据流传输延迟的累计分布情况.

3.2.2　实验结果

实验结果如图 6 所示，本方案中数据流延迟基本都在 120 ms 以内，而 SPEF 则在 140 ms 以内. 由此可见，相较于 SPEF，本文方案有更好的性能.

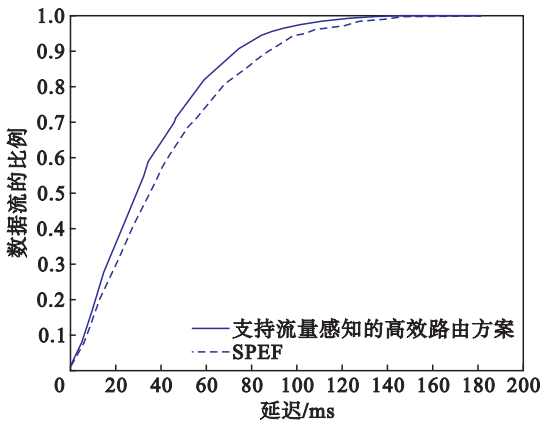

图 6　数据流延迟 CDF 图

Fig. 6　CDF of delay for data flows

由于 SPEF 关注点在于保证网络负载均衡，因此在降低数据流传输延迟方面的性能不如本方案.

4　结　　论

本文提出了一种支持流量感知的软件定义网络高效路由方案. 实验结果证明本文的方案可以很好地提高网络环境下的路由效率，同时能够减小最大带宽利用率，提高平均带宽利用率，而且方案本身的时间代价在毫秒级别. 本文的工作是对 OpenFlow 路由方案的直接扩展. 接下来，作者计划将本文方案部署到真实的 SDN 环境中.

参考文献：

[1] Open Networking Foundation. Software defined networking: the new norm for networks[EB/OL]. (2012 - 10 - 15)[2015 - 04 - 30]. https://www.opennetworking.org/images/ stories/downloads/openflow/wpsdnnewnorm.pdf.

[2] McKeown N, Anderson T, Balakrishnan H, et al. Openflow: enabling innovation in campus networks[J]. *SIGCOMM Computer Communication Review*, 2008, 38(2): 69 - 74.

[3] Bennesby R, Fonseca P, Mota E, et al. An inter-as routing component for software-defined networks[C]// IEEE/IFIP Network Operations and Management Symposium. Maui, 2012: 138 - 145.

[4] Li D, Shang Y, Chen C. Software defined green data center network with exclusive routing[C]//IEEE Conference on Computer Communications (INFOCOM). Toronto, 2014: 1743 – 1751.

[5] Karl M, Gruen J, Herfet T. Multimedia optimized routing in openflow networks[C]//IEEE International Conference on Networks (ICON). Singapore, 2013: 1 – 6.

[6] Egilmez H, Civanlar S, Tekalp A. An optimization framework for QoS-enabled adaptive video streaming over openflow networks[J]. *IEEE Transactions on Multimedia*, 2013, 15(3): 710 – 715.

[7] Egilmez H, Gorkemli B, Tekalp A, et al. Scalable video streaming over openflow networks: an optimization framework for QoS routing[C]//IEEE International Conference on Image Processing (ICIP). Brussels, 2011: 2241 – 2244.

[8] Egilmez H, Civanlar S, Tekalp A. A distributed QoS routing architecture for scalable video streaming over multi-domain openflow networks[C]//IEEE International Conference on Image Processing (ICIP). Orlando, 2012: 2237 – 2240.

[9] OpenFlow. OpenFlow switch specification version 1. 1. 0. [EB/OL]. (2011 – 02 – 28)[2015 – 04 – 10]. http://archive. openflow. org/ documents/openflow-specv1. 1. 0. pdf.

[10] Adrichem N, Asten B, Kuipers F. Fast recovery in software-defined networks[C]// Third European Workshop on Software Defined Networks (EWSDN). Budapest, 2014: 61 – 66.

[11] Adrichem N, Doerr C, Kuipers F. Opennetmon: network monitoring in openflow software-defined networks[C]//IFIP Network Operations and Management Symposium. Krakow, 2014: 1 – 8.

[12] Sharma S, Staessens D, Colle D, et al. Automatic configuration of routing control platforms in openflow networks[J]. *ACM SIGCOMM Computer Communication Review*, 2013, 43(4): 491 – 492.

[13] Hu F, Hao Q, Bao K. A survey on software-defined network and OpenFlow: from concept to implementation[J]. *IEEE Communications Surveys and Tutorials*, 2014, 16(4): 2181 – 2206.

[14] Shin S, Yegneswaran V, Porras P, et al. Avant-guard: scalable and vigilant switch flow management in software-defined networks[C]//ACM SIGSAC Conference on Computer & Communications Security. Berlin, 2013: 413 – 424.

[15] Xu K, Shen M, Liu H, et al. Achieving optimal traffic engineering using a generalized routing framework[J]. *IEEE Transactions on Parallel and Distributed Systems*, 2016, 27(1): 51 – 65.

[16] University of Adelaide. The Internet topology zoo [EB/OL]. (2013 – 04 – 16)[2015 – 04 – 10]. http://www. topology-zoo. org/ dataset. html.

(上接第 324 页)

4 结 论

本文结合卫星拓扑变化特点,设计网络容量分析方法,刻画比例公平性约束,构建资源分配优化模型,改进燕子群算法,提出一种低轨道星间功率带宽资源联合分配方法. 仿真分析得出卫星瞬时吞吐容量和网络瞬时容量变化具有周期性和随机性,所提资源分配算法通过牺牲网络容量获得卫星间资源分配的公平性. 在实际应用背景下验证并改进本文方法的实用性是今后研究工作的重点.

参考文献:

[1] Nakahira K, Abe J, Mashino J. Novel channel allocation algorithm using spectrum control technique for effective usage of both satellite transponder bandwidth and satellite transmission power [J]. *IEICE Transactions on Communications*, 2012, 95(11): 3393 – 3403.

[2] Wu X L, Chen Y Y, Gao L Q, et al. A utility-based OFDM resource allocation scheme for LEO small satellite system [C]// International Conference on Cyberspace Technology. Stevenage: IET, 2014: 68 – 73.

[3] Ji Z, Wang Y Z, Feng W, et al. Delay-aware power and bandwidth allocation for multiuser satellite downlinks[J]. *IEEE Communications Letters*, 2014, 18(11): 1951 – 1954.

[4] Wang H, Liu A J, Pan X F. Optimization of joint power and bandwidth allocation in multi-spot-beam satellite communication systems [J]. *Mathematical Problems in Engineering*, 2014, 2014: 1 – 9.

[5] Wang X W, Wang X Y, Che H, et al. An intelligent economic approach for dynamic resource allocation in cloud services [J]. *IEEE Transactions on Cloud Computing*, 2015, 3(3): 275 – 289.

[6] Wang X W, Cheng H, Li K Q, et al. A cross-layer optimization based integrated routing and grooming algorithm for green multi-granularity transport networks[J]. *Journal of Parallel and Distributed Computing*, 2013, 73(6): 807 – 822.

[7] Wang X W, Cheng H, Huang M. Multi-robot navigation based QoS routing in self-organizing networks [J]. *Engineering Applications of Artificial Intelligence*, 2013, 26(1): 262 – 272.

[8] Neshat M, Sepidnam G, Sargolzaei M. Swallow swarm optimization algorithm: a new method to optimization[J]. *Neural Computing and Applications*, 2013, 23(2): 429 – 454.

[9] Kaveh A, Bakhshpoori T, Afshari E. An efficient hybrid particle swarm and swallow swarm optimization algorithm [J]. *Computers & Structures*, 2014, 143: 40 – 59.