



一种软件定义网络中基于蚁群算法的一致性路由策略

王雨薇¹, 兰巨龙¹, 席孝强¹, 范茜勉², 江逸茗¹

(1. 国家数字交换系统工程技术研究中心 郑州 450002;

2. 华北水利水电大学电力学院 郑州 450002)

摘要: 软件定义网络通过对交换机下发流表项实现数据平面的更新, 但流表项的安装具有时延, 可能导致部分数据分组被错误地处理或丢弃。对此, 提出一种基于蚁群算法的一致性路由策略, 在路由参数中加入不一致时延, 并通过控制器对流表的下发顺序做出调整, 避免了在数据分组传输过程中, 交换机控制逻辑的不一致。实验结果表明, 与最短路径算法相比, 出现不一致问题时, 该策略所选路径的实际时延小于最短路径算法的时延, 数据分组丢失率、错误转发的概率明显降低。

关键词: 软件定义网络; 控制逻辑; 一致性; 路由; 蚁群算法

doi: 10.11959/j.issn.1000-0801.2015206

An Consistent Routing Policy Based on Ant Algorithm in Software Defined Networking

Wang Yuwei¹, Lan Julong¹, Xi Xiaoqiang¹, Fan Ximian², Jiang Yiming¹

(1. National Digital Switching System Engineering & Technological R&D Center, Zhengzhou 450002, China;

2. School of Electronic Power, North China University of Water Resource and Electric Power, Zhengzhou 450002, China)

Abstract: Software defined networking updated data through distributing flow table to switches. But the installation of flow table had delay and may lead to wrong forwarding. An consistent routing policy based on ant algorithm was proposed. The inconsistent delay was introduced in routing parameter. The sequence of flow setup was adjusted through the controller, so as to avoid the inconsistency of the control logic in switches. Results show that, compared with Dijkstra algorithm, when the inconsistency problem happens, the proposed policy has less actual delay and the packet lost rate and the wrong forwarding rate have been increasingly decreased.

Key words: software defined networking, control logic, consistency, routing, ant algorithm

收稿日期: 2015-05-29; 修回日期: 2015-08-03

基金项目: 国家重点基础研究发展计划(“973”计划)基金资助项目(No.2012CB315901, No.2013CB329104), 国家自然科学基金资助项目(No.61309019, No.61372121), 国家高技术研究发展计划(“863”计划)基金资助项目(No.2013AA013505)

Foundation Items: The National Basic Research Program of China(973 Program)(No.2012CB315901, No.2013CB329104), The National Natural Science Foundation of China(No.61309019, No.61372121), The National High Technology Research and Development Program of China(863 Program)(No.2013AA013505)

论文引用格式: 王雨薇, 兰巨龙, 席孝强等. 一种软件定义网络中基于蚁群算法的一致性路由策略. 电信科学, 2015206

Wang Y W, Lan J L, Xi X Q, et al. An consistent routing policy based on ant algorithm in software defined networking. Telecommunications Science, 2015206



1 引言

目前,可重构网络^[1]、智慧协同网络^[2]、软件定义网络^[3]等新型网络体系结构为解决传统网络的僵化问题提供了新的方案。其中,软件定义网络(software defined networking, SDN)因其逻辑控制和数据转发分离、网络可编程、集中控制的特点减少了能量消耗^[4],增加了网络吞吐率^[5-8],并降低了网络管理复杂度。以上特征的实现,需要更新数据平面,向交换机下发一系列流表。传统网络中,网络状态的一致性是由分布式协议实现的。在SDN控制转发分离后,如何保证流表更新的一致性,进而保证网络状态的一致性,是SDN的主要研究方向之一。

SDN允许操控者运行自己的算法以选择新的路由策略。但是在策略改变期间,网络需要维持一些原有状态,如避免回环转发、黑洞、违反访问控制政策等。一致性更新为SDN策略转换期间的状态保持提供了机制,保证了在策略转变过程中,每一个交换机在数据分组到来时,要么只执行旧的控制逻辑,要么只执行新的控制逻辑。

SDN技术是通过交换机中流表项的更新实现网络状态变化的,交换机根据流表项中的匹配项及动作集合对数据分组做出相应的处理方式(转发、丢弃、上传到控制器等)。虽然SDN具有集中控制特性,但本质上还是分布式的、异步操作的,流表项的安装具有时延,可能导致部分数据分组被错误地处理或丢弃。目前的解决方案主要集中在以下3点。

- 通过增加交换机的自治智能,分担控制器负载,从而减少控制节点到各交换机之间的时延(如DIFANE架构^[9]等),但给交换机增加控制功能违反了SDN中控制与转发相分离的原则。
- 通过控制器的集中控制功能,对流表项进行分类。如参考文献[10]构建中间规则,保留与共同流表项相关的数据分组,将其余数据分组上传给控制器缓存,直到其余规则更改完毕。但每次对流表项进行处理后,都需要等待一个全网最大时延,增加了流表更新时间。
- 对新旧两套规则进行区分,如设置优先级或增加额外标签。参考文献[11]以额外的数据标签(VLAN、MPLS)来区分新、旧两套规则,但额外标签的使用可能导致数据分组被错误转发。

第二类问题中,参考文献[10]是构建一组中间规则,保

证任何时刻交换机中只有一组规则。参考文献[12]对流表项和交换机进行分类,根据分类结果进行更新。上述更新方式虽然能够基本保证在数据分组的传输路径中,所有交换机经历的控制逻辑一致,但每次写入流表项都需要等待一个全网最大时延,使得流表项更新时间加长。

针对上述问题,本文提出一种基于蚁群算法的一致性路由策略,该策略属于在保证一致性前提下的单路径路由选择方案,通过集中式网络的全局视角调用网络状态信息,动态调整路由计算参数,在路由参数中加入不一致时延,并通过控制器对流表的下发顺序做出调整,在保证一致性的同时减少了流表项更新时间。首先,控制器根据交换机上报的信息确定全网拓扑、时延及链路剩余带宽;其次,结合各链路的时延、控制器和交换机之间的时延、交换机的不一致时延选择数据分组的路由路径;最后,控制器根据所选择路径中各交换机的不一致时延准确下发流表,数据分组返回数据平面。

2 基于蚁群一致性的路由策略

对一致性路由问题进行了重述和建模,将问题建模成为一个考虑不一致时延的路由选择问题,继而给出相应的蚁群一致性(ant colony optimization-consistency, ACO-C)策略并对其进行了描述,ACO-C策略包括ACO-C算法和控制器流表下发机制。

2.1 基于蚁群一致性的路由策略模型

2.1.1 问题的描述及相关定义

SDN中,控制器通过向交换机下发一系列流表来实现数据平面的更新。而数据平面的更新是异步的,流表项的生效时间有先后,可能导致部分交换机在数据分组到来时同时经历新、旧控制逻辑,即经历控制逻辑不一致。如图1所示,交换机1在 t_1 时刻安装新规则,交换机2在 t_2 时刻安装新规则($t_2 > t_1$),则在时间段(t_1, t_2)内到达交换机2的数据分组因没有相应的转发规则,而处于未知的处理状态。

本文研究的问题是:为数据分组的传输选择路由路径,保证在数据分组传输过程中,每一个交换机要么只执行旧的控制逻辑,要么只执行新的控制逻辑。

具有全网视图的SDN能在很大程度上提高路由算法的性能,主要原因有:SDN控制平面具有全局性,可集中获取全网信息并根据不同的需求计算路由、有效管理和调度流量;控制平面根据数据平面实时上报的网络信息进行流量和网络状态分析,及时调整流表下发顺序或更改路由路

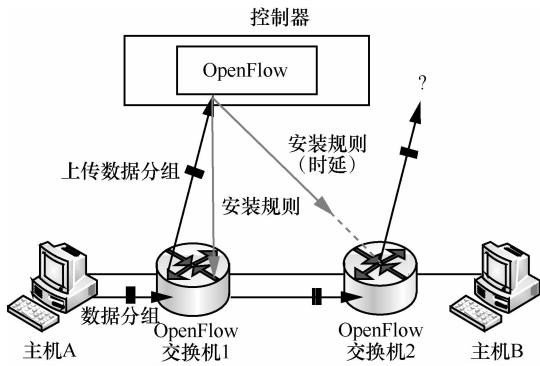


图 1 控制逻辑不一致示意

径,实现一致性转发,优化网络性能。

网络可以描述为一个无向图 $G=(V,E)$ 。其中,图的顶点表示网络节点;图的边表示网络链路; $V=(v_1, v_2, \dots, v_i, \dots, v_n)$ 为各网络节点 (OpenFlow 交换机) 的集合; $E=(e_{12}, e_{23}, \dots, e_{ij}, \dots, e_{ln})$ 为网络链路集合。

为了选择数据分组的一致性路由路径,本文给出了如下 3 个定义。

定义 1 (网络时延) 网络时延分为相邻交换机之间的时延及控制器和交换机间的时延。本文为了网络时延取值尽可能准确及计算方便,随机取一段时间内的时延进行若干次测量,最后取平均时延表示节点的网络时延。

$$d_{ij} = \frac{1}{n} \sum_{q=1}^n d_{ij}^q \quad (1)$$

其中, d_{ij} 代表相邻交换机 v_i 与 v_j 的网络时延,若 v_i 与 v_j 不相邻,则取 $d_{ij}=0$ 。 n 代表随机测量的次数, d_{ij}^q 表示对时延的第 q 次测量值。

同样,控制器 c 到交换机 v_i 的时延可表示为:

$$c_i = \frac{1}{n} \sum_{q=1}^n c_i^q \quad (2)$$

定义 2 (可达路径及剩余带宽) $p(s,d)$ 表示从交换机 v_s 到 v_d 的所有可达路径的集合。 $P_{sd} \in p(s,d)$, $P_{sd} = \langle v_s, v_m, \dots, v_d \rangle$ 表示 v_s 到 v_d 之间的一条有序可达路径。 $l_{sn} \in P_{sd}$, $v_n \in P_{sd}$ 为可达路径 P_{sd} 的一条子路径。网络各边的剩余带宽 $b_{ij} = \text{bandwidth}(e_{ij})$ 通过询问端口交换机获得。则可达路径 P_{sd} 的实际带宽为其瓶颈带宽: $\text{bandwidth}(P_{sd}) = \min\{\text{bandwidth}(e_{ij}), v_i, v_j \in P_{sd}\}$ 。带宽约束条件为 $\text{bandwidth}(P_{sd}) \geq B$ 。

定义 3 (交换机的不一致时延) $\forall v \in p(s,d), \exists v_h \in P_{sd}$, $P_{sd} = \langle v_s, \dots, v_h, \dots, v_d \rangle$, 满足:

$$c_h > c_s + \sum_{v_i, v_j \in P_{sd}} d_{ij} \quad (3)$$

则表示在该可达路径中,交换机 v_h 存在因时延而引起的控制逻辑不一致,且交换机 v_h 的不一致时延为:

$$\Delta t_h = c_h - c_s - \sum_{v_i, v_j \in P_{sd}} d_{ij} \quad (4)$$

若交换机没有经历控制逻辑不一致,则记 $\Delta t=0$ 。

则可达路径 P_{sd} 的总时延为:

$$\Delta TP_{sd} = 2c_s + \sum_{v_i, v_j \in P_{sd}} d_{ij} + \max(\Delta t) \quad (5)$$

2.1.2 模型的建立

本文在 SDN 基本构架的基础上,在控制器中 ACO-C 模块,如图 2 所示。ACO-C 模块从数据平面调取所需信息供蚁群算法使用,然后计算出最优路径和各交换机的不一致时延。

控制器利用 packet_in 和 packet_out 消息在 OpenFlow 交换机中运行发现协议。控制器定时地发送含有 LLDP 数据分组的 packet_out 消息给与其相连的 OpenFlow 交换机,并根据反馈回来的 packet_out 消息获知数据平面的信息。控制器中的交换机管理模块、主机管理模块、拓扑管理模块分别记录来自数据平面的各种信息,如网络拓扑、网络时延 (d_{ij} 与 c_i)、链路剩余带宽 b_{ij} 等,并把这些信息传递给 ACO-C 模块。当数据分组到来时,源交换机把数据分组上传给控制器进行缓存,然后 ACO-C 模块根据 ACO-C 算法计算出路由路径及各交换机的不一致时延 Δt ,并按照流表下发规则依次精准下发流表,对源交换机下发流表后,即可将数据分组返回到数据平面。

2.2 策略的相关计算

2.2.1 ACO-C 算法规则

ACO-C 算法在蚁群算法中加入了一致性时延参数。蚁群算法是意大利学者 Marco Dorigo 在受到真实蚂蚁觅食行为的启发后,提出的一种启发式算法,能够根据调整影响路由选择的参数 (如链路时延、不一致时延、时延抖动等),灵活地选择路由路径。

算法对路径的选择规则和信息素轨迹强度调整规则的定义如下。

(1) 路径选择规则

每只蚂蚁根据概率选择式由当前节点 i 选择下一节点 j :

$$p_{ij}^i = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in allowed_k} [\tau_{il}(t)]^\alpha [\eta_{il}]^\beta} & , j \in allowed_k \\ 0 & , \text{其他} \end{cases} \quad (6)$$

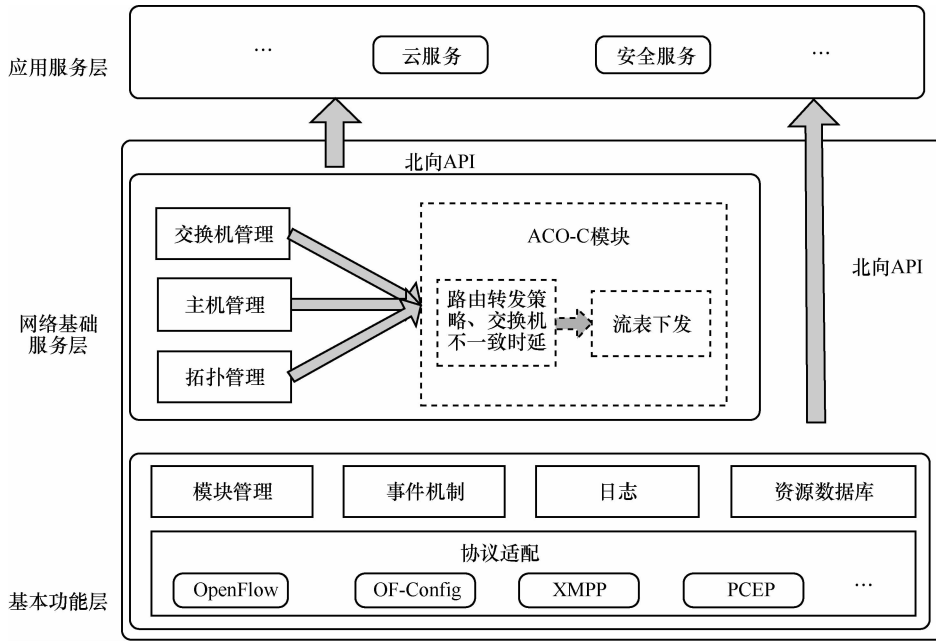


图2 加入 ACO-C 模块的控制器结构

其中, $\tau_{ij}(t)$ 表示 t 时刻边 e_{ij} 上的信息素轨迹强度; α 表示轨迹的相对重要性 ($\alpha \geq 0$); β 表示时延相对重要程度的控制参数 ($\beta \geq 0$); $allowed_k$ 表示第 k 只蚂蚁下一步可以选择的节点集, η_{ij} 为边 e_{ij} 的时延影响度。由于本文中, 影响总时延的因素包括 d_{ij} (节点的网络时延) 和 T_j^k (第 k 只蚂蚁行走至节点 j 时, 交换机 v_j 的不一致时延与前向交换机的最大不一致时延之差)。

$$T_j^k = \begin{cases} \Delta t_j - \max(\Delta t)_{si}, & \Delta t_j > \max(\Delta t)_{si} \\ 0, & \Delta t_j \leq \max(\Delta t)_{si} \end{cases} \quad (7)$$

其中, $\max(\Delta t)_{si}$ 表示子路径 l_{si} 存在的最大不一致时延, 所以选取:

$$\eta_{ij} = 1/(d_{ij} + T_j^k) \quad (8)$$

式(6)通过综合链路 e_{ij} 上的信息轨迹强度及时延影响度 η_{ij} , 有利于选择总时延更小的链路。

(2) 信息素轨迹强度调整规则

对于蚂蚁 k , 设当它成功完成一次路由后选择的路径为 P_{sd}^k , 完成一次循环需要走 n 步, 设 $e_{ij} \in P_{sd}^k$, 则 e_{ij} 上信息量根据式(9)、式(10)进行调整:

$$\tau_{ij}^k(t+n) = \rho \cdot \tau_{ij}^k(t) + \Delta \tau_{ij}^k(t, t+n) \quad (9)$$

$$\Delta \tau_{ij}^k(t, t+n) = \begin{cases} \frac{Q}{L_k}, & \text{蚂蚁 } k \text{ 在本次循环中经过路径 } e_{ij} \\ 0, & \text{其他} \end{cases} \quad (10)$$

其中, $\Delta \tau_{ij}^k(t, t+n)$ 表示第 k 只蚂蚁在时间 $(t, t+n)$ 内留在路径 e_{ij} 上的信息素量, 按照式(10)进行计算, $L_k = \Delta T P_{sd}^k - 2c_s$,

m 表示蚂蚁总数, $\Delta \tau_{ij}(t, t+n)$ 表示本次循环中 e_{ij} 信息素量的增量, $(1-\rho)$ 为信息素轨迹的衰减系数。

算法的主要流程是, 对删减了不满足带宽条件后的网络进行寻路, 在选择下一跳时考虑链路时延和交换机的不一致时延, 得到最优路径。算法首先选择源节点和目的节点; 计算所有下一跳节点的选择概率 (行(12)、行(13)); 采用转轮赌法选择蚂蚁下一步将要走的节点, 根据式(6)的计算结果, 概率大的下一跳有更大的机会被选中, 从而加快了收敛速度 (行(14)); 每只蚂蚁寻路完毕后, 进行信息素的更新, 以利于后续蚂蚁的寻路 (行(23)); 算法迭代 $N_{c \max}$ 次, 输出总时延最小的路径, 若总时延最小的路径有多条, 选择瓶颈带宽最大的路径 (行(26)~(28))。

输入: 网络拓扑 $G(V, E)$; 网络中各链路时延邻接矩阵 D 与带宽矩阵 L ; 网络中控制器与各交换机时延矩阵 c ; 确定源节点 s 与目的节点 d ; 给出带宽约束条件 B 的值; 给出参数 α, β, ρ, Q 的值; 确定迭代次数 $N_{c \max}$ 与每波蚂蚁的个数 m 。

输出: 源节点到目的节点的最优路径及最优路径的时延。

初始化: $T^k = \phi; N_c = 1; \Delta \tau_{ij} = 0; \tau_{ij}(t) = \rho h$ /* 初始化蚂蚁 k 行走的路径中每个交换机的不一致时延差 T^k , 循环计数器 N_c , 轨迹强度增量 $\Delta \tau_{ij}$; 为每条边设置一个轨迹强度 $\tau_{ij}(t)$ 初始值;

(1) 删除网络中不满足带宽要求的链路

```

(2) For( $N_c=1; N_c < N_{c\max}; N_c++$ )
(3)  $tabu_k = \phi$  /*  $tabu_k$  为禁忌表, 用来保存第  $k$  只蚂蚁行走的路径, 每次循环初始阶段, 禁忌表为空 */
(4) For( $k=1; k \leq m; k++$ )
(5) DELAY[ $k$ ]=0 /* 蚂蚁  $k$  所走过的路径的总时延 */
(6)  $p_k = s$  /* 蚂蚁  $k$  的当前节点 */
(7) While( $p_k \neq d$ )
(8) 确定蚂蚁  $k$  下一跳的节点集合  $allowed_k$ 
(9) If  $allowed_k = \phi$ 
(10) 蚂蚁  $k$  跳出循环
(11) Else
(12) 根据式(7)计算蚂蚁  $k$  下一跳不一致时间差矩阵  $T^k$ 
(13) 根据式(6)计算蚂蚁  $k$  选择下一跳各节点的概率
(14) 根据转轮赌法选择下一跳节点  $v$ 
(15)  $Path = [Path, v]$  /* 路径添加 */
(16) DELAY[ $k$ ] = DELAY[ $k$ ] + D( $s, v$ ) +  $T_v^k$  /* 路径时延累计 */
(17)  $p_k = s$  /* 蚂蚁  $k$  移动到当前节点 */
(18) 修改禁忌表
(19) end If
(20) end While
(21) end For
(22) 记录下每只蚂蚁的觅食路线和线路长度
(23) 根据式(9)、式(10)进行全局信息素的更新
(24) 清空禁忌表
(25) end For
(26) 保存总时延最小的路径
(27) 若总时延最小的路径有多条, 选择瓶颈带宽最大的路径
(28) 输出最优路径

```

2.2.2 控制器流表下发机制

在 SDN 中, 由于流表项的更新是异步的, 流表项的安装具有时延, 所有交换机上的新流表项不能同时生效。根据 SDN 控制器的集中控制特性, 在控制器中加入控制器定时器, 通过调整流表的下发时间避免各交换节点出现控制逻辑不一致的情况。

(1) 由上述算法产生的最优路径 P_{sd} 中, 对存在控制逻辑不一致的交换机节点分别计算其不一致时延 Δt , 并与此交换机点组成二元组 $(v, \Delta t)$, 把所有的二元组按照 Δt 从小到大的顺序放入栈 H 。

(2) 设置控制器定时器初始值 $NT = \max(\Delta t)$, 从栈顶取出第一个二元组, 并使控制器对其中的交换节点 v 下发流表, 同时删除第一个二元组, NT 递减一个时钟周期。

(3) 当 NT 的值与 H 中第 k 个二元组中 Δt 的值相同时, 控制器对此二元组中的交换机下发流表, NT 递减一个时钟周期。如此重复至 $NT=0$ 。

(4) 当 $NT=0$ 时控制器同时对可达路径中的其余交换机下发流表。

(5) 当控制器对源交换机 v_s 下发流表后, 即可将数据分组返回数据平面。

2.3 性能分析

本文对流表更新一致性方案提出如下评价指标: 隔离性, 即考虑是否可能对其他控制行为产生影响; 更新时间, 即完成流表项更新所需要的时间。由于参考文献[9]提出的方案违反了 SDN 中控制与转发相分离的原则, 只将本文所提方案与参考文献[10]、参考文献[11]、参考文献[12]所提方案进行比较, 见表 1。

表 1 本文所提策略与相关研究比较

方案	隔离性	更新时间
本文方案	良好	长
参考文献[10]方案	良好	较长
参考文献[11]方案	较差	短
参考文献[12]方案	良好	较长

(1) 隔离性

参考文献[11]所提方案使用额外标签区分两套流表项, 需要将标签信息添加为流表项匹配信息的一部分。如果新旧规则中, 已经将此标签作为流表项匹配信息, 则数据分组会被错误转发或丢弃。参考文献[10, 12]所提方案与本文所提策略不需要更改流表项匹配信息, 因此隔离性较好。

(2) 更新时间

参考文献[10]所提方案由于引入了中间规则, 流表项更新时间要长于参考文献[11]。参考文献[12]所提方案分阶段对交换机进行更新, 流表更新时间近似于参考文献[10]。本文所提方案仅通过控制器对流表的下发顺序进行调整, 且不需要等待全网最长时延。在已知新规则, 即不需要计算路由由路径的情况下, 流表项的更新时间大于参考文献[11]



所提方案,小于参考文献[10]、参考文献[12]所提方案。

综上,参考文献[11]所提方案受限于隔离性,而本文所提方案与参考文献[10,12]所提方案相对则更优。本文所提方案对更新时间进行优化,因此又优于参考文献[10,12]所提方案。

3 仿真与结果分析

3.1 仿真环境搭建

关于环境搭建和参数设置,本文做出以下几点说明。

(1) 拓扑的选择和链路参数的设置

本文根据 Reeves 博士的随机拓扑生成算法生成拓扑(Salama 模型)^[13]完成算法验证,其中,时延以 ms 为单位,链路时延根据交换节点间的距离等因素确定,剩余带宽在 [0,1 000] 上均匀分布,并对控制器到各节点的时延做了处理,以使得部分路径中,交换机存在控制逻辑不一致现象。图3 为随机生成的一张 8 节点网络拓扑图。各边参数分别表示时延及剩余带宽,各交换机参数表示控制器到交换机的时延。

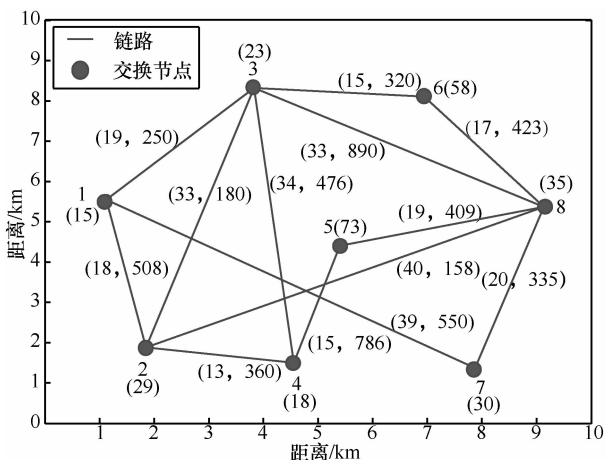


图3 仿真拓扑

(2) ACO-C 算法参数的设置

对蚁周系统的仿真结果表明, β 在 5~10 最优, α 的

最优值在 1 附近, ρ 的最优值在 0.7 附近。本文仿真试验中,取 $\alpha=1, \beta=5, \rho=0.7$, 蚂蚁只数 $m=20$, 最大迭代次数 $N_{c, \max}=200$, 各链路的信息素初始值 $ph=50$ 。

3.2 仿真分析

当前 SDN 控制器路由的实现以最短路由算法为主,本文采用控制器中的常用路由算法 Dijkstra 算法和 ACO-C 策略进行对比。为了保证二者的仿真结果的公平性,在 Dijkstra 算法开始时删减掉不满足带宽限制的边,同时在 Dijkstra 算法计算出的最短路径时延中,加入数据分组在源交换机的上传和下载时间。

3.2.1 实验一

本文使用如图 3 所示网络拓扑对 ACO-C 策略进行了验证。在仿真实验中,设置当交换机经历控制逻辑不一致时,把数据分组上传给控制器缓存,由控制器下发新的流表后再将数据分组返回网络,即无分组丢失和错误转发的情况存在。随机选取 8 组(源节点,目的节点),仿真结果见表 2。其中,路径选择为运行算法所选择出的最优路径,计算时延为算法计算出的最优路径的时延,实际时延为数据分组的实际传输时延。

表 3 为运行 Dijkstra 算法的情况示意。由表 2 可看出,对于 ACO-C 算法所选择出的路径,计算时延和实际时延保持一致。由表 3 可以看出,由 Dijkstra 算法所选择出的最优路径中,部分路径的计算时延小于实际时延。对于部分单播路由请求,分别通过两种算法求出的路由路径不一致,如(1,8)、(3,7)、(4,8)。对于部分单播路由请求,虽然通过两种算法计算出的路径一致,但路径的实际时延却不同,如(1,5)、(2,8)、(3,5)。这是由于 Dijkstra 算法不能考虑到一致性问题,仅以链路的网络时延作为选择标准,在数据分组的传输过程中,若出现不一致问题,则交换机把数据分组上传给控制器,增加了数据分组的上传和下载时间。而

表 2 ACO-C 算法实验结果

单播路由请求(源节点,目的节点)	路径选择	计算时延/ms	实际时延/ms	交换机时延调整(交换机,时延/ms)
(1,8)	1-3-8	82	82	/
(1,5)	1-2-4-5	88	88	(V5,12)
(2,6)	2-1-3-6	110	110	/
(2,8)	2-4-5-8	121	121	(V5,16)
(3,5)	3-4-5	96	96	(V5,1)
(3,7)	3-8-7	99	99	/
(4,6)	4-3-6	85	85	/
(4,8)	4-3-6-8	102	102	/

表 3 Dijkstra 算法实验结果

单播路由请求(源节点,目的节点)	路径选择	计算时延/ms	实际时延/ms	出现不一致问题的交换机
(1,8)	1-3-6-8	81	99	V6
(1,5)	1-2-4-5	76	100	V5
(2,6)	2-1-3-6	110	110	/
(2,8)	2-4-5-8	105	137	V5
(3,5)	3-4-5	95	97	V5
(3,7)	3-6-8-7	98	138	V6
(4,6)	4-3-6	85	85	/
(4,8)	4-5-8	70	150	V5

ACO-C 策略把一致性时延作为路径选择的参数之一,并通过控制器对流表的下发顺序做出调整,避免了额外的上传与下载时间。对比表 2、表 3 可明显看出,当出现不一致问题时,由 ACO-C 策略选择出的路径的实际时延小于由 Dijkstra 算法选择出的路径的实际时延。

3.2.2 实验二

在 Mininet 实验平台上对 ACO-C 策略和 Dijkstra 算法所选择出的路径进行对比分析。实验环境为 Mininet-2.1.0-130919-ubuntu-13.04-server-amd64,Mininet 内内置 Ryu 控制器。对比试验步骤如下。

(1) 将如图 3 所示拓扑用 Python 语言写入虚拟机,各节点时延与各链路带宽定义如图 3 所示,每个交换节点分别与一个主机节点相连。

(2) 每次随机选择两个主机进行通信,每次测试时清空所有交换机的流表项。

(3) 到达交换机的数据分组,若在流表项中找不到相应的匹配规则,则选择丢弃或随机转发到其他交换机(模拟控制逻辑不一致)。

(4) 分别用 ACO-C 策略和 Dijkstra 对各交换节点间的路径进行选择。

(5) 利用 Ryu 控制器对交换节点信息进行统计。
本实验中,在对交换节点进行信息统计时,对除了每次通信的源节点以外的交换机进行统计,以排除算法收敛时间的影响。表 4 列出了随机选取 1 000 次联通主机进行 UDP 通信的情况下 ACO-C 策略和 Dijkstra 算法分别出现控制逻辑不一致的统计信息。实验表明,对于和控制器之间时延较大的交换机,很有可能出现控制逻辑不一致现象,如交换机 5、6。Dijkstra 算法不考虑一致性问题,当数据分组经过这些交换机时,丢弃率和错误转发率明显增高。

4 结束语

针对 SDN 流表更新的一致性的问题,本文提出一种基于蚁群算法的一致性路由策略,通过集中式网络的全局视角调用网络状态信息,动态调整路由计算参数,在蚁群算法参数中加入不一致时延,并通过控制器对流表的下发顺序做出调整,保证了在数据分组传输过程中,交换机控制逻辑的一致。仿真结果表明,ACO-C 策略在保证数据分组一致性转发的基础上,能够选出总时延最短的最优路径。与最短路径算法相比,用 ACO-C 策略选择出的路径能

表 4 信息统计比较

交换机编号	Dijkstra 算法		ACO-C 算法	
	丢弃率	错误转发率	丢弃率	错误转发率
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0
5	8.62%	7.45%	0	0
6	7.68%	5.72%	0	0
7	0	0	0	0
8	0	0	0	0



够有效降低数据分组的丢弃率和错误转发率。

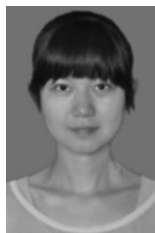
参考文献

- 1 兰巨龙,程东年,胡宇翔.可重构信息通信基础网络体系研究. 通信学报, 2014, 35(1): 128~139
Lan J L, Cheng D N, Hu Y X. Research on reconfigurable information communication basal network architecture. Journal on Communications, 2014, 35(1): 128~139
- 2 张宏科, 罗洪斌. 智慧协同网络体系基础研究. 电子学报, 2013, 41(7): 1249~1254
Zhang H K, Luo H B. Fundamental research on theories of smart and cooperative networks. Journal on Electronics, 2013, 41(7): 1249~1254
- 3 SDN: the new norm for networks. <https://www.opennetworking.org/images/stories/downloads/openflow/wpsdn-newnorm.pdf>, 2012
- 4 Heller B, Seetharaman S, Mahadevan P, *et al.* ElasticTree: saving energy in datacenter networks. Proceedings of Conference on Networked System Design and Implementation, San Jose, CA, USA, 2010
- 5 Al-Fares M, Radhakrishnan S, Raghavan B, *et al.* Hedera: dynamic flow scheduling for data center networks. Proceedings of Conference on Networked System Design and Implementation, San Jose, CA, USA, 2010
- 6 Benson T, Anand A, Akella A, *et al.* MicroTE: finegrained traffic engineering for data centers. Proceedings of International Conference on Emerging Networking Experiments and Technologies, Tokyo, Japan, 2011
- 7 Curtis A R, Mogul J C, Tourrilhes J, *et al.* DevoFlow: scaling flow management for high-performance networks. Proceedings of Special Interest Group on Data Communication, Toronto, Canada, 2011
- 8 Jain S, Kumar A, Mandal S, *et al.* B4: experience with a globally-deployed software defined WAN. Proceedings of Special Interest Group on Data Communication, Hong Kong, China, 2013
- 9 Minlan Y, Michael J, Jia W. Scalable flow-based networking with DIFANE. Proceedings of Special Interest Group on Data Communication, New Delhi, India, 2010
- 10 Mcgeer R. A safe, efficient update protocol for OpenFlow networks. Proceedings of the First Workshop on Hot Topics in Software Defined Networks, New York, USA, 2012
- 11 Reitblatt M, Foster N, Rexford J, *et al.* Abstractions for network update. Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, New York, USA, 2012
- 12 周烨, 杨旭, 李勇等. 基于分类的软件定义网络流表更新一致性方案. 电子与信息学报, 2013, 35(7): 1746~1752

Zhou Y, Yang X, Li Y, *et al.* Software defined network flow table update consistent plan based on classification. Journal of Electronics & Information Technology, 2013, 35(7): 1746~1752

- 13 Salama H F. Multicast routing for real-time communication on high-speed networks. Raleigh: North Carolina State University, 1996

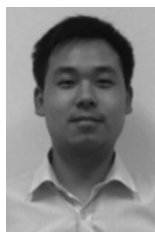
作者简介



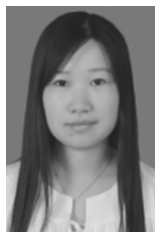
王雨薇, 女, 国家数字交换系统工程技术研究中心硕士生, 主要研究方向为新型网络体系结构。



兰巨龙, 男, 国家数字交换系统工程技术研究中心教授、博士生导师, 主要从事新一代信息网络关键理论与技术的研究工作, 目前作为首席科学家主持国家“973”计划项目“可重构信息通信基础网络体系研究”。



席孝强, 男, 国家数字交换系统工程技术研究中心硕士生, 主要研究方向为新型网络体系结构。



范茜勉, 女, 华北水利水电大学硕士生, 主要研究方向为电力系统稳定与安全运行。



江逸茗, 男, 国家数字交换系统工程技术研究中心讲师, 主要研究方向为新型网络体系结构、可重构网络、网络虚拟化。