计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210800045

FDSR:一种面向 SD-MANET 的快速转发规则下发方法

张耕强　谢　钧　杨章林
陆军工程大学指挥控制工程学院　南京 210007
(1204995726@qq.com)

摘　要　针对软件定义网络在移动自组织网络里部署传输路径时,需要控制器为路径上所有的节点下发相关流表项,从而造成传输开始需等待较长时间的问题,文中基于分段路由提出了一种快速下发数据转发规则的方法 FDSR。控制器会以分段路由的方式,通过在数据包上添加转发路径对应标签的方式下发数据转发规则,并在处理最大堆栈深度问题时,利用标签粘连技术,将整条路径分为多个标签栈,通过算法选择后分别下发给能与控制器节点快速交互的转发节点,以减少路径配置时间。实验结果表明,相较于 OpenFlow 流表下发方式,FDSR 在 SD-MANET 中能减少路径部署时长以及流表开销,并能有效应对 SR 的 MSD 问题,提升控制器部署长路径的速度。

关键词:SDN;MANET;转发规则下发;路径部署时间;分段路由
中图法分类号　TP393

Accelerating Forwarding Rules Issuance with Fast-Deployed-Segment-Routing(FDSR) in SD-MANET

ZHANG Geng-qiang, XIE Jun and YANG Zhang-lin
College of Command & Control Engineering, Army Engineering University of PLA, Nanjing 210007, China

Abstract　Aiming at the problem that the deployment of transmission paths in MANET(mobile ad hoc network) for SDN(software defined network) requires the controller to issue related flow entries for all nodes on the path, thereby causing that the transmission have to wait for a long-term period, a mechanism for quickly issuing forwarding rules, FDSR(fast-deployed-segment-routing), is proposed. The controller will issue data forwarding rules by adding labels corresponding to the forwarding path on the data packets in the way of segmented routing, and when dealing with the max stack depth problem, use label adhesion technology to divide the entire path into multiple label stacks by algorithm, sending to corresponding forwarding nodes that can quickly interact with the controller node to reduce path configuration time. Experiments show that compared with the OpenFlow distribution method, FDSR in SD-MANET (software defined mobile ad hoc network) can reduce path deployment time and flow table overhead, and can effectively deal with the MSD problem of SR and improve the long path deployment speed of the controller.

Keywords　Software defined network, Mobile ad hoc network, Forwarding rules issuance, Paths configuration time, Segment routing

1　引言

软件定义网络(Software-Defined Networking,SDN)近年来成为提高移动自组织网络(Mobile Ad hoc Network,MANET)业务能力的一种有效方案。SDN 拥有强大的网络可视化能力以及灵活的流量调控能力,能根据上层应用需求,从全局网络角度高效分配资源和选择路径,并在捕捉到网络变化后迅速进行资源再配置,动态调整数据传输路径。利用 SDN 协助 MANET 进行路由决策,能为应用流量提供高质量的 QoS 路由。

SDN 将网络的控制面和数据面完全分离,构造了集中式控制,以获取对网络资源的全局规划,这使得 SDN 设备需要完全按照控制器下发的流表进行数据包头部匹配,对流量进行转发、丢弃等相应动作。在使用 OpenFlow 协议的常用 SDN 结构里,控制器为流设置新路径时需要和该路径上的所有节点交互,配置每个节点的相关流表项。这种方式有两个缺点:一是在流数目较多时,扩展性较差;二是在网络存在频繁链路变化时,控制器需要在多个交换设备间实现快速流表更新,以减少网络收敛时间[1]。然而相较于有线网络,MANET 没有稳定充足的带宽资源支持,并且内存容量较低,很难完美适配以高速交换机为基础设施层对象的 OpenFlow 协议[2],此外,控制器所部署的节点和不同位置的节点之间的交互时长存在一定的差异,部分节点需要通过多跳连接和控制器进行数据的交换[3]。这些特点都极有可能增加路径的配置

到稿日期:2021-08-04　返修日期:2021-10-18
基金项目:国家自然科学基金(61971439)
This work was supported by the National Natural Science Foundation of China(61971439).
通信作者:谢钧(xiejun73@189.cn)

时间和传输时延，并影响控制器更新流表的有效性。然而现有工作大部分都集中在路由和数据流量的平衡上，较少涉及数据层与控制层间的高效率交互研究[4]。

分段路由(Segment Routing，SR)近年来被用于改善 SDN 控制器和数据面间的交互。SR 是源路由的一种形式，它拥有在数据包头中添加指令列表(段)的能力，可以支持流量工程、服务功能链和虚拟专用网等高级服务[5]。其利用源路由的思想，在源节点数据包上配置完毕发送路径，从而减少为中间节点配置流表项产生的开销和时延；同时充分结合 SDN 的集中式控制，避免传统源路由寻路过程造成的较长时延，能实现更简单灵活的路由[6]。但目前 SR 技术大多被部署在有线网络，鲜有在 SD-MANET(Software Defined Mobile Ad hoc Network)的应用研究。此外，受 MANET 多跳特点影响，SR 在实际应用中也会受到设备最大标签栈深度的限制，在部署跳数较多的路径时需要使用标签粘连技术，而如何选择各粘连标签的第一跳也是决定路径部署速度的一个重要因素。

面向 SD-MANET，基于 SR 提出了一种快速转发规则下发机制 FDSR (Fast-Deployed-Segment-Routing)，以提升转发规则的更新速度。控制器在利用分段路由下发转发规则从而减小 MANET 里的流表资源消耗的同时，能加快转发规则更新的速度，以匹配网络变化速度。此外，还提出一种标签分割算法，在需要使用粘贴标签时，通过分析控制器到各转发节点间的距离时延选择每个分段列表的第一跳，以实现最快速的转发规则下发。

2 相关工作

文献[7]通过在手机和电脑上部署 SDN 网络，验证了部署多控制器无法很好地解决 SD-MANET 的可靠性问题，为此设计了一种结合分段路由的混合式控制架构，即数据平面节点根据控制器下发的段列表确定下一跳，通过传统路由协议进行路由。文献[8]利用分段路由进行路径的重新配置，当网络变化导致控制器需要更改并重新配置路径时，控制器通过下发段列表暂时引导源节点的数据传输，并在此期间配置新的传输路径上的所有节点的流表，配置完毕后停止段列表的下发并启动新路径传输，从而实现了轻便快速的路由更新。文献[9]针对机载骨干网(Airborne Backbone Network，ABN)，在使用层次式控制加强 SD-ABN 可靠性的基础上，应用了分段路由进行转发规则的下发。该文献对流表下发进行了详细的设计实现，当收到节点转发请求后，平台控制器会将路由策略下发至请求节点对应的 SDN 节点，无线电模块通过访问无线信道将策略发送至 ABN 控制器，ABN 控制器计算出所需的多条可靠路径后，将路由策略反向下发给请求节点。实验结果表明，SR 在网络规模扩大时，能有效提升控制器更新效率并减少网络收敛时间和控制开销。

上述研究都利用 SR 技术进行 SD-MANET 的路由，但都是对 SR 技术的直接应用，并没有考虑在实际中会限制 SR 的 MSD(Max Stack Depth)问题，也未提出进一步的解决方案。针对 MSD 问题，文献[10]提出了一种基于关键节点的标签栈压缩算法 LSC-K，该算法通过利用全局视图分析网络拓扑关系找到关键节点，并利用松散路径原理，删减转发路径上的无效节点，从而实现标签堆栈的压缩。实验结果表明，LSC-K 算法可以有效压缩传输路径的标签堆栈数目，从而扩大受控网络的规模，提高网络资源利用率。然而，该算法的核心思想是选择关键节点对路径进行压缩，故在路径较长、关键节点较多而难以完全压缩时依旧需要引入标签粘连技术。文献[11]提出了分段列表管理算法 LMRP(Longest Match Relay Push)。LMRP 算法将路径转换为简化的段列表，并选择源节点和若干个中继节点作为压入相应段列表指令的流表项的对象，路径上的所有节点都根据各自的段路由转发表独立进行数据包的匹配转发。实验结果表明，LMRP 可以减少标签资源和流表项资源的消耗，实现任意长路径的数据流转发，同时不受 MSD 限制。然而，这些研究虽然针对 MSD 问题提出了解决方法，但其对象并不是 SD-MANET，也并未考虑 SD-MANET 中各节点与控制器间的较大交互时延差异，因此当被应用至 SD-MANET 时，其对转发规则下发速度的提升效果可能较为有限。

综上所述，目前虽然有研究将 SR 技术应用至 SD-MANET 中，但都未考虑到在实际中会影响规则下发速度的 MSD 问题；而解决 MSD 问题的研究又并不是针对 SD-MANET 的，并未考虑到带宽匮乏、多跳连接等网络特点，难以完全应用至 SD-MANET 中。因此本文提出 FDSR，该机制能在应对 MSD 问题的同时，有效提升 SD-MANET 中转发规则的下发速度。

3 SD-MANET 以及分段路由

3.1 SD-MANET

SDN 近年来在数据中心、广域网以及服务链等领域的成功应用，主要归功于其对网络底层设备的集中控制。这一思想被借鉴应用至 MANET 中，利用 SDN 的集中式控制统一管理设备，弥补 MANET 对等式结构难以合理管理网络资源的缺陷[12]。传统 MANET 路由协议中，节点完全根据自己收集的本地信息进行分布式路由决策，当需要传输数据时，MANET 节点会首先查询自身流表，在无匹配结果后向 SDN 控制器发出路径请求，由其进行路径计算。与传统 MANET 路由协议不同，SD-MANET 控制器在根据自身维持的网络全局视图计算完路径后，将路径转换为数据包转发规则下发至数据层 MANET 节点。这种集中式的控制能有效提升 MANET 在链路资源分配、拥塞避免以及多域网络融合等方面的表现。SD-MANET 原理如图 1 所示。

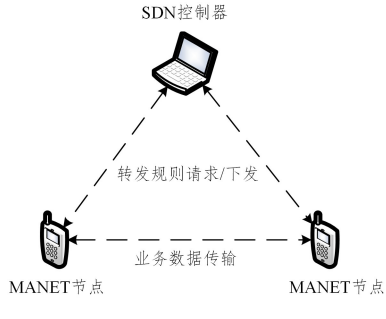


图 1 SD-MANET 原理图

Fig. 1 SD-MANET schematic diagram

然而，平面的分离意味着平面之间存在额外的通信需求，这在一定程度上会增加转发的网络流量的延迟。SDN 设备完全按照控制器下发的流表匹配数据包，对流量进行转发、丢弃等相应动作。一般地，SDN 控制器在收到路径请求后，首先会根据全局视图计算出传输路径，然后需要和该路径上的所有转发节点交互，下发每个节点的相关流表项。然而这种转发规则下发方式在 MANET 里可能会产生较大的初始配置时延，其原因在于源节点在发送路径请求至控制器后，需要等待一定的时间，直到路径上的所有节点都得到了控制器下发的转发规则后才能开始传输。

除了初始配置时延外，由于 MANET 节点的移动性较强，网络易频繁发生链路状态变化，使两端传输路径需要被频繁调整。受 SDN 集中式控制影响，转发节点在发现链路变化后对节点移动所导致的路径失效问题的及时应对能力不强，通常需要再请求控制器更新流表以部署新的路径。因此在 SD-MANET 服务具有时延要求的应用流量时，网络必须具有快速更新转发规则以及调整传输路径的能力。SD-MANET 面临的主要挑战之一，即控制器需要在多个 MANET 节点间实现快速的规则下发或更新，以减少网络收敛时间，保证控制器更新流表的有效性[3]。

3.2 分段路由

有限的流表空间导致 SDN 在灵活性和可扩展性方面面临许多挑战。传统的基于 OpenFlow 或 MPLS(Mutilple Label Switch) 的流转发方案可能会因流表溢出或 MPLS 标签负载过重而导致性能下降[13]。分段路由是 MPLS 的改进。SR 基于源路由的思想，通过在源节点发送的数据包上附加表示传输路径的多标签，实现在源节点配置完毕发送路径，减少为中间节点配置流表项所带来的开销和时延；同时，充分结合 SDN 的中心化控制，避免传统源路由寻路过程造成的较长时延，实现更简单灵活的路由。

SR 原理如图 2 所示。当发送方 A 请求服务时，控制器根据全局视图计算出合适路径$\{A,B,D,C\}$，并以分段列表(Segment List，SL)的方式告知 A 需经过的链路。A 在数据包头部附上标签$\{1007,2004,3006\}$，而后交付给对应的下一跳节点 B。B 收到该数据包后无需查询路由表，仅需要识别数据包标签对应的链路，通过默认流表匹配交付给相应端口，直到传输给接收方 C。

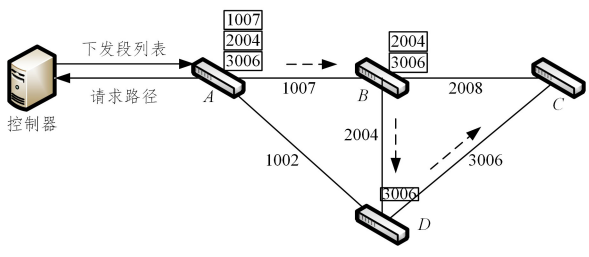


图 2 分段路由原理图

Fig. 2 Segment routing schematic diagram

然而，SR 在实际应用中存在一定的限制。为了达到线速传输，目前的大部分设备标签栈深度为 3～5 层[14]，这就使得控制器在处理跳数较多的 MANET 节点间路径时需要使用标签粘连[15]技术。而在应用标签粘连技术将路径分为多个标签栈时，若直接按最大标签栈深度依次分割，则会产生一个问题，即分割后的某段 SL 可能会被下发至距离控制器节点间具有较多跳数或者较大时延的数据转发节点，进而造成转发规则下发时间过长，影响传输质量。

如图 3 所示，假设最大标签栈深度为 5，P 为粘连标签，邻接 SIDs 每格内代表该链路的 SIDs。由于 SL 长度为 9，超出最大栈深度 5，因此控制器需要对某传输路径的邻接 SIDs 进行分割。控制器在进行分割时，若以常规方式按最大栈深度进行分割，则最大时延为 max{20，42，30}，最终得到 42；而最优情况下可以使下发时延下降至 max{20，22，20}，即 22。由此可见，对分段列表进行分割设计可以在一定程度上缩短转发规则下发的时间，从而使控制器能快速响应数据节点的转发请求，并最小化链路变化对传输造成的影响。

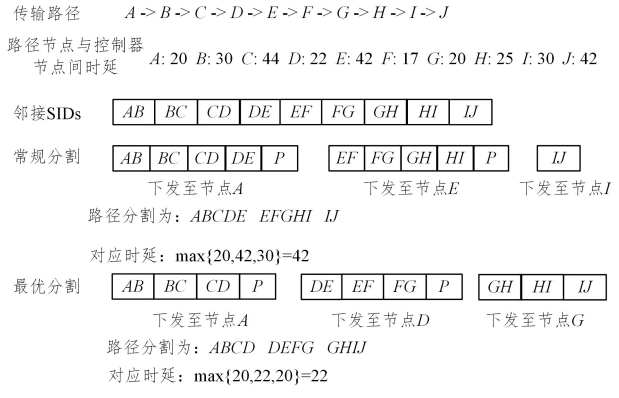


图 3 标签分割问题

Fig. 3 Label segmentation problem

4 FDSR：基于分段路由的转发规则下发方法

为了提高 SD-MANET 的转换规则下发速度，本文提出了一种转发规则下发方法 FDSR。FDSR 基于 SR 技术，以源路由的方式进行传输路径的部署。此外，针对 SR 的 MSD 问题，FDSR 应用一种标签分割算法，以解决按标签栈深度进行标签分割的常规方法的弊端，进一步提升转发规则的下发速度。

4.1 基于 SR 的 SD-MANET

网络结构分为控制层与数据层。控制器在收到数据层节点的传输请求时，首先根据维持的全局网络视图计算出最优路径，而后将该路径转换为 SL。当 SL 长度超出数据层节点最大标签栈深度后，SDN 控制器按 4.2 节所提标签分割算法将 SL 分割为多个子 SL，并提取各子 SL 对应的第一个节点作为接收对象，下发该子 SL 至该节点。源节点在将自己接收到的 SL 压入数据包标签栈后，对最外层标签进行匹配，查询自身默认流表得到对应链路的端口，而后将最外层标签弹出，从匹配端口发送至对应链路上，开始数据传输。中间转发节点按照数据包最外层标签栈进行默认流表项的匹配、弹出、转发等动作，其余收到 SL 的节点在识别到粘连标签后，将相关联的 SL 再次压入数据包标签栈，同上述流程进行数据包转发，最终传输至目的节点处。具体流程如图 4 所示。

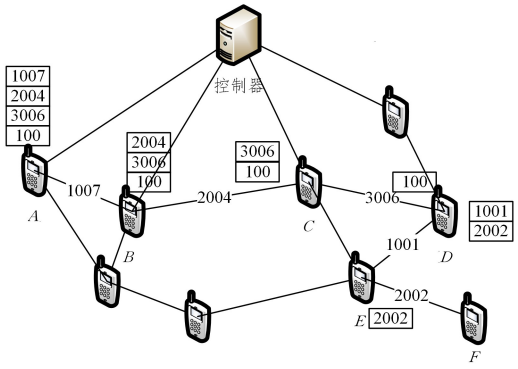


图 4 基于 SR 的 SD-MANET

Fig. 4 SR-based SD-MANET

步骤 1 控制器收到节点 A 到 F 的传输请求后，通过自身维持的全局视图计算出最佳路径 $ABCDEF$，对应{1007，2004，3006，1001，2002}。由于超出栈深度，控制器将该路径分割为{1007，2004，3006，100}，{1001，2002}两部分，其中 100 为粘连标签，分别下发给节点 A 与 D。

步骤 2 源节点 A 为数据报文添加标签栈{1007，2004，3006，100}，然后根据栈顶的标签 1007 匹配链路，结果为链路 AB，之后节点 A 将标签 1007 弹出，将携带标签栈{2004，3006，100}的数据包通过链路 AB 向下一跳节点 B 转发。

步骤 3 转发节点 B 收到报文后，根据栈顶的标签 2004 匹配到链路 AB。之后节点 B 将标签 1006 弹出，将携带标签栈{3006，100}的数据包通过链路 BC 向下一跳节点 C 转发。

步骤 4 转发节点 C 收到后也按上述方式转发数据包，发送给节点 D。

步骤 5 节点 D 收到报文后，识别出栈顶的标签 100 为粘连标签，将粘连标签 100 交换为与其关联的标签栈{1001，2002}，然后根据新的栈顶的标签匹配链路，找到对应链路 DE，之后将标签 1001 弹出。报文携带标签栈{2002}，通过链路 DE 传输至节点 E。

步骤 6 节点 E 收到报文后，按栈顶标签继续进行弹出转发动作。最后一个标签 2002 被弹出，数据最终被转发至目的节点 F。

4.2 标签分割

为了提高 SDN 下发转发规则的速度，本文基于 SR 提出标签分割算法，从传输路径中选择距离控制器较近的节点作为接收 SL 的节点，提升转发规则下发速度。标签分割算法的伪代码如算法 1 所示。

算法 1 标签分割算法

输入：路径 P 上交换机到控制器的 SID 集合 $S=\{(s_1),(s_2),\cdots,(s_k)\}$

输出：路径 P 分割后的 SID 集合 $res=\{(s_1,\cdots,s_i),(s_{i+1},\cdots,s_j),\cdots,(s_m,\cdots,s_k)\}$

```
1. if len(S)<=MSD
2.    res=S
3. else
4.    While(True)
5.     SL1=S. find_max_delay_SL()
6.      if s1. delay == SL1. delay
7.         Break;
8.      else
9.         SL2 = SL1. find_frontal_SL()
10.        if len(SL1)+len(SL2)<= MSD-1
11.           S. Combine(SL1,SL2)
12.        else
13.           Break;
14.  res=S
15. Return res
```

如图 5 所示，$D0-D6$ 为对某一路径运行标签分割算法的流程，$D7$ 为最后减少 SL 数量的额外合并步骤。

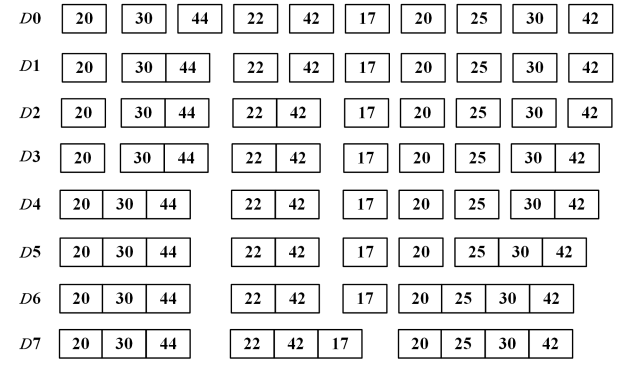


图 5 标签分割算法流程示意图

Fig. 5 Label segmentation algorithm flow diagram

对于路径 p 的转发节点到控制器节点的时延集合 $D0$ {{20}，{30}，{44}，{22}，{42}，{17}，{20}，{25}，{30}，{42}}，设置最大标签栈深度为 5，除去粘连标签外，最大标签栈深度即为 4。根据标签分割算法，控制器首先找到最大值所在集合{44}，而后将其与前一项{30}合并，得到集合{30，44}，该集合对应的时延大小为 30，更新集合得到 $D1$；然后再次进行循环，找到此时 $D1$ 的最大值 42 所在集合{42}，将其与前一项合并得到{22，42}，更新 $D1$ 得到 $D2$；以此类推直到 $D6$，时延最大值 22 所在集合{22，42}，由于加上前一项{20，30，44}超过了标签栈深度限制 4，因此无法继续合并，循环就此终止。

最后对运行标签分割算法得到的结果进行额外处理，将其他可以合并的集合如{17}进行合并，以减少 SL 数目，得到最终结果 $D7$，以此为各 SL 添加粘连标签并下发至 SL 的第一项元素对应的节点。

该算法需要每次更新后重新寻找最大值所在块，将其与前一项合并。设路径长度为 n，即初始一共 n 块。每次循环合并两个相邻块后开始下次循环，故循环每次会减少一个块，即算法在整个过程中遍历块数为 n，$n-1$，…，$n-m$，因此该算法的时间复杂度为 $O(n^2)$。该算法由于主要面向网络传输路径，问题规模并不大，因此在实际运行时能较快得到优化结果。

FDSR 的标签分割算法所得结果一定是最优解，即不存在比其所得结果具有更小时延的路径分割结果。

定理 1 FDSR 的标签分割算法所得最终解 R 为最优解。

证明：首先给出如下定义。

定义 1(元素) 单个节点对应的时延值为一个元素，即 M_i 为一元素。

定义 2(块) 多个元素所组成的集合，即(M_i，M_{i+1}，…，M_j)为一个块。

规则 1 由算法原理可得，所有路径在按算法进行标签分割后，每个块内元素的前一项一定小于或等于后一项(对于

长度小于 MSD 或者源节点具有到控制器节点最大时延的路径，由于每个块内只有一个元素，因此该规则成立）。

假设存在比 R 更优的解 R'，设 R 各块值的最大值为 M_i，分为以下两种情况：

（1）当 M_i 为第一个块第一元素（即源节点）时

由于无论怎么分割，源节点必会收到第一个块对应的 SL，因此该情况不存在更优解 R'。

（2）当 M_i 不为第一个块第一元素，即 $R=\{\cdots,(M_h,\cdots,M_{i-1}),(M_i,\cdots,M_j),\cdots\}$ 时

由于 R' 为更优解，因此 R' 各块对应值的最大值一定比 M_i 更小。

此外，根据规则 1，R 内 M_i 所在块的后面元素必须和 M_i 属于同一块，否则将产生另一个第一个元素值不小于 M_i 的块，这与 R' 具有更小时延值矛盾。

因此 M_i 一定被前面某一个小于 M_i 的元素 M_{h+k+1} 为第一项的块合并，即 R' 为 $\{\cdots,(M_{h+k+1},\cdots,M_{i-1},M_i,\cdots,M_j),\cdots\}$，而 R 为 $\{\cdots,(M_h,\cdots,M_{h+k},M_{h+k+1},\cdots,M_{h+k+m},\cdots,M_{i-1}),(M_i,\cdots,M_j),\cdots\}$。

对于 R，当最大值为 M_i 时不再向前合并分两种情况：

（1）M_i 所在块元素未满

由于算法是按每一次循环后第一个最大值所在块作为合并对象，而 M_i 所在块在 R 中未与前面元素合并，说明 M_{i-1} 一定大于或等于 M_i（若小于，则 M_{i-1} 在合并 M_i 前一定单独成块，进而下一循环一定会被合并成 $(M_{i-1},M_i,\cdots,M_j)$，与 R 不符合）。

由于 $M_h<M_i$，且 M_{h+k+1} 小于 M_i，则在集合 $(M_{h+k+1},\cdots,M_{i-1})$ 内必存在两个连续元素 M_{h+a} 和 M_{h+a+1} 使得 $M_{h+a}<M_i\leqslant M_{h+a+1}$。

由定理 1 可知，算法在运行过程中必有一合并过程结果为：

$\{\cdots,(M_{h+a}),(M_{h+a+1},\cdots,M_{i-1}),(M_i,\cdots,M_j),\cdots\}->\{\cdots,(M_{h+a},M_{h+a+1},\cdots,M_{i-1}),(M_i,\cdots,M_j),\cdots\}$

此时最大值为 M_i，而由于合并后长度超过限制，块 $(M_i,\cdots,M_j)$ 无法继续向前合并 $(M_{h+a},\cdots,M_{i-1})$，因此块 $(M_i,\cdots,M_j)$ 最终成为了结果 R 的一个元素集合。

然而，R' 为 $\{\cdots,(M_{h+k+1},\cdots,M_{i-1},M_i,\cdots,M_j),\cdots\}$，这说明 $(M_{h+k+1},\cdots,M_{i-1})$ 加上 $(M_i,\cdots,M_j)$ 的长度未超过最大长度。而 $(M_{h+a},\cdots,M_{i-1})$ 是 $(M_{h+k+1},\cdots,M_{i-1})$ 的一个子集，两者相互矛盾，故该情况不符合。

（2）M_i 所在块元素已满

此种情况下，对于 R'，M_i 所在块 $(M_{h+k+1},\cdots,M_i,\cdots M_j)$ 长度超出最大标签栈深度限制，与前提条件矛盾，故该情况不符合。

综上，假设不成立，不存在更优解。因此 FDSR 的标签分割算法得到的最终解 R 即为最优解。

5 实验分析

首先，使用 Matlab 进行时延评估，实验拓扑选用文献[15]提供的真实网络数据集，相关时延、距离等数据选用文献[16]在利用该拓扑进行实验时提供的开源数据。网络由包括控制器节点在内的共 60 个节点组成，该拓扑能为评估 FDSR 算法在不同长度路径下所带来的时延改进提供足够数量的不同跳数路径。其次，使用 Mininet[17] 的无线网络扩展版本 Mininet-wifi[18] 构造同一网络拓扑，通过部署控制器 Ryu 实现 SL 以及流表的下发，分析 FDSR 的流表开销。

5.1 时延评估

本部分实验使用 Matlab 实现。实验所用路径集由设定拓扑内所有节点两两间路径组成，一共有 1～10 跳 10 种路径长度，每种路径数量最少为 50 条，最多达到了 600 多条。

图 6 为实验结果的 3 种类别，按时延大小排序。类别 1：OpenFlow＞SR＞FDSR；类别 2：OpenFlow＝SR＞FDSR；类别 3：OpenFlow＝SR＝FDSR。其中，类别 1 数量最多，为优化的一般结果。类别 2 中 OpenFlow 和 SR 下发转发规则时延相同，由于在利用 SR 下发多个 SL 时，部分 SL 被下发至距控制器有最长时延的节点，最终导致时延未能得到优化，而 FDSR 针对这一点额外进行了 SL 接收节点的选取，降低了下发时延。出现类别 3 结果的原因为控制器下发规则至源节点的时间已经长于其到路径上其余所有节点的时间，故 SR 与 FDSR 未能起到优化效果。

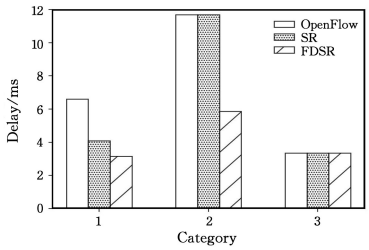


图 6 实验结果类别

Fig. 6 Experimental result category

图 7 为在不同跳数的路径下控制器按 OpenFlow，SR 以及 FDSR 3 种方法在 1～10 跳路径下进行下发的平均值统计图。可以看出，SR 和 FDSR 由于在数据包上压入路径，仅需要配置更少节点的流表，对于多数路径而言其更新时延比在每个节点配置流表的 OpenFlow 方法更低。而在 1～3 跳内，由于路径跳数未超过标签栈深度，控制器仅需要在源节点将整条路径压入，因此 SR 与 FDSR 有着相同的时延；而在路径跳数大于 3，需要使用标签粘合技术时，相较于按栈深度分割 SL 的 SR，FDSR 具有更好的转发规则下发速度，时延更低。

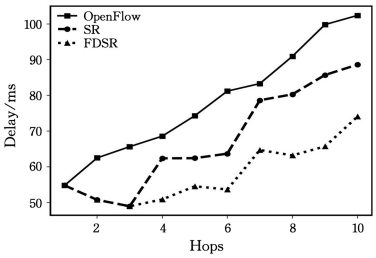


图 7 OpenFlow，SR，FDSR 时延对比

Fig. 7 OpenFlow，SR，FDSR delay comparison

5.2 开销分析

本部分通过 Mininet-wifi 以及 Ryu 实现，其中 Mininet-wifi 负责构造所用实验拓扑，Ryu 以 OpenFlow，SR，FDSR 这 3 种方式下发转发规则。

实验结果如图 8 所示。可以看出，由于 SR 仅需要部署少数节点的 pop/push 流表项（用于将 SL 压入数据包标签

栈),而转发规则只需要使用默认流表项目,因此相较于OpenFlow,SR大大减小了流表项开销。而FDSR受益于SR技术,在流表项开销上相较于OpenFlow也有了较大提升。FDSR相较于SR流表项开销有一点增加的主要原因是在分割路径时进行了节点选择,导致有些SL未能压满,因此FDSR会增加一定SL的数目,最终导致流表项开销增多。但这种开销数目较小,相较于OpenFlow,FDSR在优化流表项开销的同时大大提升了路径部署速度,整体性能较为良好。

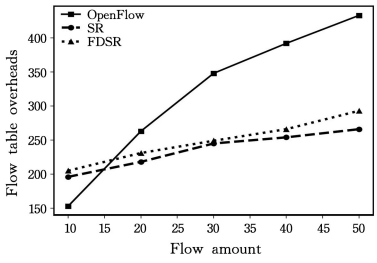


图8 流表开销对比

Fig. 8 Flow table overheads comparison

结束语 针对SDN转发规则下发需要更新路径上所有节点流表,过程较为繁琐且在MANET里易增加传输时延的问题,本文提出了一种基于SR快速下发转发规则的机制FDSR。控制器通过SR下发转发规则,并在处理MSD问题时,利用标签粘连技术将分段列表划分为多部分下发给路径上距控制器较近的转发节点,以减少路径配置时间。实验结果表明,相较于OpenFlow和SR,FDSR在SD-MANET中能降低路径的部署时延,并能有效应对SR的MSD问题,提升控制器部署长路径的速度。此外,受益于SR,FDSR相较于OpenFlow优化了流表开销。

参 考 文 献

[1] ABDULLAH Z N, AHMAD I, HUSSAIN I. Segment Routing in Software Defined Networks: A Survey[J]. IEEE Communications Surveys & Tutorials, 2019, 21(1): 464-486.

[2] HAWBANI A, WANG X, ZHAO L, et al. Novel architecture and heuristic algorithms for software-defined wireless sensor networks[J]. IEEE/ACM Transactions on Networking, 2020, 28(6): 2809-2822.

[3] ZAHMATKESH A, KUNZ T. Software defined multihop wireless networks: promises and challenges[J]. Journal of Communications & Networks, 2017, 19(6): 546-554.

[4] POULARAKIS K, QIN Q, MA L, et al. Learning the optimal synchronization rates in distributed SDN control architectures [C]// IEEE INFOCOM 2019-IEEE Conference on Computer Communications. IEEE, 2019: 1099-1107.

[5] ABDELSALAM A, VENTRE P L, SCARPITTA C, et al. Srperf: a performance evaluation framework for ipv6 segment routing[J]. IEEE Transactions on Network and Service Management, 2020, 18(2): 2320-2333.

[6] ZHANG G Y, MA J C. Talking about the application of segmented routing in SDN[J]. Post and Telecommunications Design Technology, 2016(11): 77-80.

[7] POULARAKIS K, QIN Q, MARCUS K M, et al. Hybrid SDN Control in Mobile Ad Hoc Networks[C]// 2019 IEEE International Conference on Smart Computing (SMARTCOMP). IEEE, 2019: 110-114.

[8] LUO L, YU H, LUO S, et al. Achieving Fast and Lightweight SDN Updates with Segment Routing[C]// 2016 IEEE Global Communications Conference (GLOBECOM). Washington, DC, 2016: 1-6.

[9] CHEN K, ZHAO S, LV N, et al. Segment Routing Based Traffic Scheduling for the Software-Defined Airborne Backbone Network[J]. IEEE Access, 2019, 7: 106162-106178.

[10] SHI H W, HUANG F Z. Segment routing label stack compression algorithm based on key nodes[J]. Electronic Technology and Software Engineering, 2020(12): 39-43.

[11] ZHOU J, ZHANG Z, ZHOU N. A Segment List Management Algorithm Based on Segment Routing[C]// 2019 IEEE 11th International Conference on Communication Software and Networks (ICCSN). IEEE, 2019.

[12] CHEN X, WU T, SUN G, et al. Software-Defined MANET Swarm for Mobile Monitoring in Hydropower Plants[J]. IEEE Access, 2019, 7: 152243-152257.

[13] LI Z, HU Y. Pasr: An efficient flow forwarding scheme based on segment routing in software-defined networking[J]. IEEE Access, 2020, 8: 10907-10914.

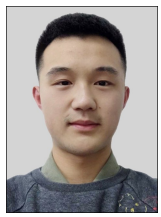
[14] HUANG L R, SHEN Q G, SHAO W J, et al. Optimizing Segment Routing With the Maximum SLD Constraint Using Openflow[J]. IEEE Access, 2018, 6(1): 30874-30891.

[15] NEUMANN A, LOPEZ E, NAVARRO L. An evaluation of BMX6 for community wireless net|works[OL]. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1001.5946&rep=rep1&type=pdf.

[16] QIN Q, POULARAKIS K, IOSIFIDIS G, et al. SDN Controller Placement With Delay-Overhead Balancing in Wireless Edge Networks[J]. IEEE Trans. Network and Service Management, 2018, 15(4): 1446-1459.

[17] FONTES R, ROTHENBERG C E. Mininet-WiFi: A Platform for Hybrid Physical-Virtual Software-Defined Wireless Networking Research[C]// Proceedings of the 2016 ACM Conference on Special Interest Group on Data Communication (SIGCOMM 2016). 2016: 607-608.

[18] LANTZ B, HELLER B, MCKDEOWN N. A Network in a Laptop: Rapid Prototyping for Software-Defined Networks[J]. Acm Sigcomm Hotnets Workshop, 2010.



ZHANG Geng-qiang, born in 1997, postgraduate. His main research interests include software defined network and mobile ad hoc network.

XIE Jun, born in 1973, Ph.D, professor, Ph.D supervisor. His main research interests include computer network and intelligent information processing.

(责任编辑:李亚辉)