面向 SDN/NFV 架构的 VNF 硬件加速资源编排机制

段通，兰巨龙，胡宇翔，范宏伟
（国家数字交换系统工程技术研究中心，河南 郑州 450002）

摘 要：SDN/NFV 架构中虚拟网络功能（VNF, virtual network function）的性能受限问题，使 VNF 的硬件加速机制成为研究热点。在部署硬件加速资源后，如何实现对硬件加速资源的统一管控和最优编排是亟待解决的问题。首先，提出了基于分离式控制的硬件加速资源统一管控架构；然后，将传统网络资源和硬件加速资源统一到网络模型中，并将硬件加速资源编排问题建模成基于线性约束的多目标优化问题；最后，设计了加速卡优先部署的启发式算法对问题进行求解。实验结果表明，与现有研究相比，所提机制能有效整合硬件加速资源，降低了近 30% 的处理时延。

关键词：SDN/NFV；虚拟网络功能；硬件加速；资源编排
中图分类号：TP393
文献标识码：A
doi: 10.11959/j.issn.1000-436x.2018108

Orchestration mechanism for VNF hardware acceleration resources in SDN/NFV architecture

DUAN Tong, LAN Julong, HU Yuxiang, FAN Hongwei
National Digital Switching System Engineering & Research Center, Zhengzhou 450002, China

Abstract: The hardware acceleration mechanism for VNF (virtual network function) is recently a hot research topic in SDN/NFV architecture because of the low processing performance of VNF. Once hardware acceleration resources have been plugged into the network, how to optimally mange and orchestrate these resources under service requirements remains a question to be solved. Firstly, a unified management architecture based on separated control for hardware acceleration resources was proposed. Then, traditional network topology together with hardware acceleration resources were modeled into a unified network model and then the hardware acceleration resource orchestration problem was transferred into a multi-objective linear programming problem. Finally, a hardware-accelerator-card-prior' heuristic algorithm was designed. Experimental results show that compared with existing methods, the proposed orchestration mechanism can efficiently manage hardware acceleration resources and reduce the processing latency by 30%.

Key words: SDN/NFV, virtual network function, hardware acceleration, resource orchestration

1 引言

当前，互联网体系正在酝酿着巨大变革，以软件定义网络（SDN, software defined networking）和网络功能虚拟化（NFV, network function virtualization）技术为典型代表的新型网络体系随之涌现。SDN 将网络控制与转发逻辑解耦合，通过网络控制接口的开放，支持上层应用对网络的管控；NFV 将

收稿日期：2017-11-08；**修回日期**：2018-05-23
通信作者：段通，duantong21@126.com
基金项目：国家网络空间安全专项基金资助项目（No.2017YFB0803204）；国家高技术研究发展计划（“863”计划）基金资助项目（No.2015AA016102）；国家自然科学基金资助项目（No.61521003）
Foundation Items: The National Network Security Special Program of China (No.2017YFB0803204), The National High Technology Research and Development Program of China(863 Program) (No.2015AA016102), The National Natural Science Foundation of China (No.61521003)

传统网络硬件设备的功能软件化，转移到数据中心的通用服务器中，从而实现网络功能的灵活部署和快速升级。由于 SDN 和 NFV 技术能够显著提升网络资源利用率、降低网络运维成本、加快网络业务部署，其引起了产业界和学术界的重点关注。然而，在 SDN/NFV 架构中，虚拟网络功能（VNF, virtual network function）一般部署在服务器上的虚拟机（VM, virtual machine）或容器（docker）中，由于虚拟 I/O 接口速率和软件处理速率的限制，其在转发性能上远远落后于传统的专用功能硬件设备。在处理网络流量较大、时延敏感的业务时，VNF 相对低下的转发处理能力使 NFV 往往难以提供相应的服务保障。

因此，针对 VNF 的硬件加速机制成为研究热点[1-2]。当前，针对 NFV 加速处理的研究可归纳为 2 个方面：1) 在服务器端对 VNF 进行加速，多通过 FPGA 加速卡[3-4]等实现；2) 借助转发节点（通常为 OpenFlow 交换机）的硬件处理能力，将 VNF 的一部分处理逻辑以表项形式加载到交换机上，实现对 VNF 的硬件处理加速[5-6]。然而，网络业务往往需要对数据流进行多重功能处理，即“服务功能链”（SFC, service function chain）；针对不同的网络业务需求，需要通过合适的功能部署和编排，实现对数据流相应功能的处理。因此，在网络中加入硬件加速资源后，如何统一地管控这些加速资源使网络在业务请求下最优地部署、编排这些硬件加速资源，以实现对服务功能链的加速处理，是亟待解决的问题。

在现有研究方面，网络功能的部署和编排问题一直是 NFV 的研究热点。Chi 等[7]针对 VNF 在数据中心内部的 3-tire 树形结构，设计了 VNF 部署和流量调度算法。随后，Xia 等[8]和汤红波等[9]分别将该问题扩展到多数据中心网络和移动核心网中，研究了 VNF 的部署和流量调度问题。Cohen 等[10]将 VNF 部署问题统一建模为线性约束下的最优化问题，并针对该问题进行了详细的数学算法分析。在 VNF 部署之后，需要针对不同的业务需求进行功能实例选取和路径计算。段通等[11]就功能实例选取及组合问题进行了理论建模，旨在实现网络效用函数的最大化。Gushchin 等[12]研究了硬件功能部署后的流量传输算法。随后，Dwaraki 等[13]将算法扩展至虚拟功能的路由。Bari 等[14]和刘彩霞等[15]对 NFV 功能部署和编排问题进行了统一建模，并借鉴维特比算法思想，将路径开销与部署开销统一到维特比状态转移表的状态转移权值中，从而同时求解出功能部署位置以及流量传输路径。Li 等[16]将功能部署、实例选取、路由 3 个部分统一成 NFV 编排问题进行了建模并提出了启发式算法，从整体上对 NFV 功能部署和编排问题进行了建模求解。Ma 等[17]对该问题进行了更深一步的研究，加入了对流量经过 VNF 后速率变化情况的考虑以及功能链无序或部分有序情况的考虑，并设计了相应的启发式算法求解。然而，由于网络加速资源的形态与通用的 NFV 服务器和 OpenFlow 交换机均有一定差异，其部署和编排方法也与基于虚拟机的 VNF 编排方法有所不同，但以上研究中的网络功能部署和编排算法均未考虑网络加速资源的部署和编排。就目前研究来看，只有华为在架构层面上提及了加速资源的部署编排问题[18]，但未给出具体的模型及算法实现。

基于以上分析可知，针对网络硬件加速资源的编排管理机制仍需进一步研究。因此，本文将服务器加速板卡和 OpenFlow 交换机统一为硬件加速资源，研究了针对硬件加速的资源管控和编排机制，从而使网络在上层业务的需求下，能够以最优的方式调度、编排资源，以实现对服务功能链的最优承载。具体地，本文的创新点包括以下 3 个方面。

1) 在架构层面，通过分析当前硬件加速资源管控方法所存在的结构性矛盾，提出了网络转发和 VNF 处理分离的硬件加速资源管控架构，实现了对硬件加速资源的一致管控。

2) 在建模层面，将传统网络资源与硬件加速资源统一到同一网络功能部署和编排问题模型中，通过理论分析确定该问题为 NP-难问题，并设计了相应的启发式算法进行求解。

3) 在实现层面，提出了可行性方案，即针对目前 OpenFlow 交换机设计不支持分离式管控的问题，利用 OpenFlow 1.3 协议中的多控制器技术来实现对硬件加速资源的单独管控。

2　加速资源统一管控架构

SDN 和 NFV 分别打破了原本封闭僵化的网络设备和功能设备，使整个网络体系朝着开放虚拟化的方向发展。从架构上看，SDN 和 NFV 分别作用于网络的不同部分且略有交叉，在实用上存在互补关系，使两者的结合变得顺理成章。2014

年，全球开放网络基金会提出了基于 OpenFlow 的 SDN/NFV 架构，旨在结合 NFV 在功能部署和 SDN 在流量调度方面的优势，构建灵活开放的新型网络体系架构。然而，现有 SDN/NFV 架构并未考虑在嵌入硬件加速资源（如 FPGA、SmartNIC、OpenFlow 交换机等）之后如何管理使用这些加速资源。

华为提出了硬件资源统一化编排管理平台[18]，将网络中的所有硬件设备进行统一编排控制，包括硬件网络设备、FPGA 加速卡、网卡、GPU、多核板卡等。在这种架构下，NFV 基础设施由 OpenFlow 交换机和通用服务器扩展到几乎所有现有的网络硬件设备和专用加速卡，为 SDN/NFV 数据平面描绘了宏伟蓝图，但其并未给出具体的实现细节。另外一种通用的思想是利用 SDN 控制器来控制硬件加速资源，文献[5]提出了基于 SDN 的 NFV 功能加载机制，将 VNF 的部分功能加载到 OpenFlow 交换机中，并采用 SDN 控制器对这部分功能逻辑进行管理。但 OpenFlow 交换机的硬件处理是无状态的，状态管理依然运行于上层 VNF 中。因此，清华大学毕军教授团队[6]提出了带状态处理的加速架构设计，通过在交换机的硬件中加入状态表，使上层 VNF 的功能几乎可以全部加载到交换机中，从而减少硬件上传数据分组的频次。然而，这种 SDN 担负部分 VNF 处理的架构会带来 2 个方面的问题：首先，为管理 OpenFlow 交换机内的功能表项，VNF 往往运行在 SDN 控制器之上，这会使原本就负责网络状态管理的控制器的负载进一步加大，造成控制器过载；其次，由于 VNF 功能表项与 SDN 转发表项同时混杂在交换机的流表里，极易导致交换机内的流表表项冲突。经以上分析可知，现有研究为实现对硬件加速资源的管控，将本应属于 VNF 所控制的硬件加速资源划归到专门负责网络转发的 SDN 控制器下，由此产生的结构性矛盾必然会导致网络转发和 VNF 处理的管控混叠以及 SDN 控制器的任务过载。

基于此，本文提出了基于 VNF 与转发分离控制的硬件加速资源管理架构（MSC, management based on separated control），如图 1 所示。在传统 SDN/NFV 架构下，虚拟资源管理器（VIM, virtual infrastructure manager）负责虚拟资源的控制，包括对网络资源的管理（NC, network controller）以及对计算、存储资源的管理（CSC, compute & storage control），在上层用户需求的驱动下，为网络服务分配相应的资源。VNF 管理器（VNFM, VNF manager）负责 VNF 的管理平面，负责对 VNF 的功能进行描述、注册、实例化以及生存时间的管理。编排器（NFVO, NFV orchestrator）则在用户服务需求下对网络功能进行编排，包括计算网络功能的部署位置以及网络流量的转发路径等。

MSC 在基于传统 SDN/NFV 架构的基础上进行了改进：1)在 VIM 中添加了硬件加速资源管理器（HAM, hardware accelerator manager），作为所有硬件加速资源的管理器，包括转发元件中的硬件加速资源和 VNF 服务器中的硬件加速卡资源；2)为实现 HAM 对硬件加速表的管理，底层转发元件的架构也应做出相应调整，即 OpenFlow 交换机的流表分成了 2 个部分，即为硬件加速表和转发表，2 个类型的流表分别承载 VNF 下发的处理规则和 SDN 控制器下发的流量转发规则，并分别被 HAM 和 NC 所控制；3) VNFM 对 VNF 的描述（VNFD, VNF description）文件中加入了对 VNF 硬件加速需求的

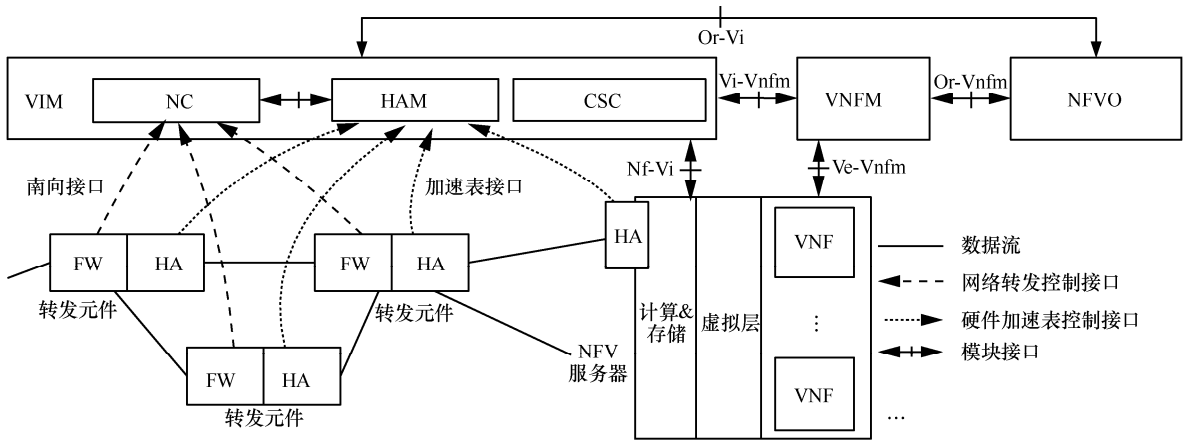


图 1 基于 VNF 与转发分离控制的硬件加速资源管理架构

描述，包括对硬件加速资源类型（如 L3 层或 L7 层处理等）、容量（如能承载的规则数量等）、处理流程的描述，通过 Vi-Vnfm 接口向 VIM 通告，然后 HAM 即可根据 VNFD 以及 VNF 实例化需求来为其分配相应的硬件加速资源进行承载。通过以上 3 种改进，网络转发控制和 VNF 加速资源管控实现了分离，这种混叠的消除使网络可以对各种类型的硬件加速资源进行统一管控，从而能在较大程度上缓解 SDN 控制器负载的同时，充分利用网络中的加速资源，提升网络功能处理的加速效果。

3　问题建模

MSC 硬件加速资源统一管控架构可以使网络根据 VNF 的加速需求，选取相应的硬件加速资源对其进行承载。基于此，本文重点研究在网络中部署硬件加速资源后，如何对硬件加速资源进行编排以实现对 SFC 的处理加速。首先，定义本文所要解决的功能部署和硬件加速资源编排问题。

定义 1　支持硬件加速的功能部署和编排问题（VDOP-HA, VNF deployment and orchestration problem enabling hardware acceleration）。网络中已经部署了交换机、服务器以及硬件加速资源，在用户的服务功能链请求下，首先，最优地部署 VNF，并将有硬件加速需求的 VNF 处理逻辑映射至相应的硬件加速资源中进行承载；然后，选择一条从源节点到目的节点的最优转发路径，使流量顺序流经服务链的各个 VNF 或承载其处理逻辑的硬件加速资源，实现对服务功能链的最优承载。

3.1　符号描述

1) 网络拓扑

VDOP-HA 网络拓扑模型示意如图 2 所示，网络中包含转发节点和 VNF 服务器节点，其中，转发节点采用 MSC 架构中的交换机结构，VNF 服务器节点则是可承载多个 VNF 运行的商用服务器。硬件加速资源包括分布于转发节点中的硬件加速表和插在 VNF 服务器中的硬件加速卡。网络拓扑记为无向图 $G(V,E)$，其中，V 是转发节点集，E 是链路集；链路代价权值为 $w(e_{ij})$，链路带宽容量为 $B(e_{ij})$，其中，$e_{ij}\in E$，$i,j\in V$；网络服务器节点集为 S，其中，服务器 s 的计算存储资源容量表示为 C_s；网络服务器与网络转发节点的连接关系用 $h(s,v)\in\{0,1\}$（$s\in S$，$v\in V$）表示。

2) 网络加速资源

各个服务器中插入的硬件加速卡资源容量记为 H_s（$s\in S$，若无硬件加速资源，则 H_s 为 0），通常表现为所能承载的表项数量或所能处理的功能类型；网络转发节点的硬件加速资源记为 H_v，通常表现为所能承载的加速表项容量。

3) 网络功能

网络功能集合为 $F=\{F_S, F_H, F_N\}$，其中，F_S、F_H、F_N 分别表示能够映射至硬件加速卡资源、能够映射至转发节点硬件加速表以及不能够进行硬件

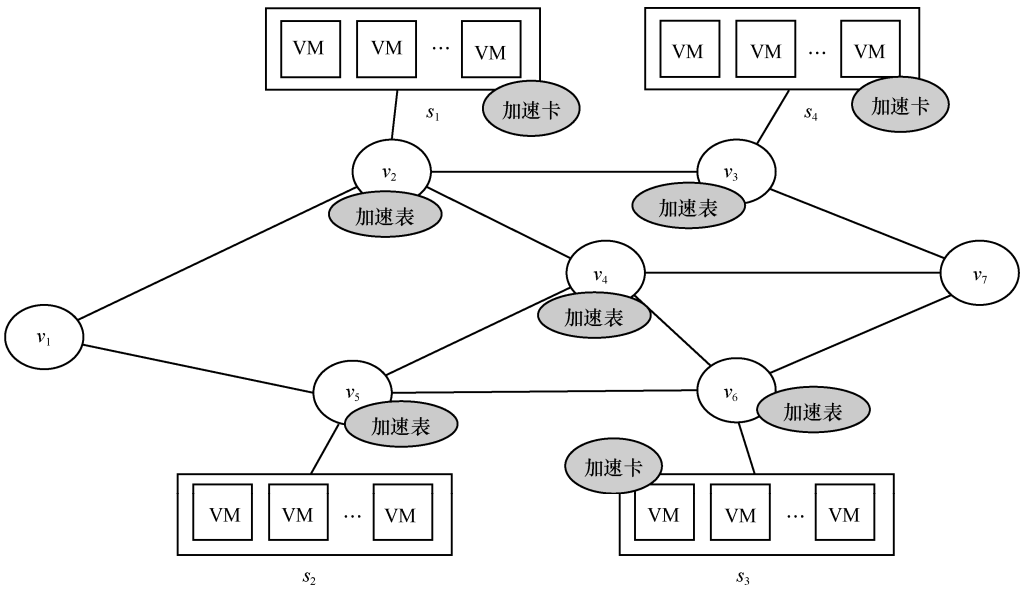


图 2　VDOP-HA 网络拓扑模型示意

加速处理的 VNF 类型，功能 $f \in F$ 的计算存储需求为 $c(f)$，加速资源需求为 $h(f)$。

4) 业务需求

用户流量处理需求记为 $T=\{v_s, v_t, (f_1 f_2, \cdots, f_n), B_t | s, t \in V, f_i \in F, 1 \leqslant i \leqslant n\}$，其中，$v_s$ 和 v_t 分别为流量传输的源、目的节点，B_t 为该流量的带宽需求。例如，图 2 所示的网络拓扑，用户需求为 $T=\{v_1, v_7, (f_1, f_2, f_3), 3$ Mbit/s$\}$，表示用户的业务需求是对从 v_1 到 v_7 的流量进行 $f_1 \rightarrow f_2 \rightarrow f_3$ 的顺序功能处理，带宽需求为 3 Mbit/s。那么就需要将(f_1, f_2, f_3)部署于合适的服务器中，并选择一条合适的路径依次流经功能 f_1、f_2、f_3 所部署的转发节点（以硬件加速资源方式部署）或 NFV 服务器。

3.2 优化目标

为简化模型，本文假设用户需求的网络功能中，能够被硬件加速处理的功能均可映射至相应的硬件加速资源。

假设 1 对$\forall f_i \in \{F_S, F_H\}, i \in \{1, 2, \cdots, n\}$，功能的硬件加速需求均可满足。由于不能被满足硬件加速处理需求的功能可以当作 $f_i \in \{F_N\}$进行部署，因此，该假设并未改变该问题的性质以及求解方法，而仅仅是简化了模型复杂度。

功能部署问题和流量路由问题通常是独立的，但在 SDN/NFV 网络环境下，这 2 个问题往往不是独立的：首先，功能的部署往往影响流量的转发顺序及路径；其次，流量转发路径的长度往往又影响功能的部署位置。为体现功能部署和流量传输的联合处理，问题的目标一般选为最小化总开销，包括链路开销和功能部署开销。此时，VDOP-HA 可表述为

$$\min \ \alpha \sum_{i \in V} \sum_{j \in V} [y(i,j) \times w(e_{ij})] + \beta \sum_{s \in S} \sum_{i=1}^{n} [x(f_i, s) \times c(f_i)] \tag{1}$$

$$\text{s.t.} \ \ x(f_i, s) \in \{0,1\}, i \in (1,2,\cdots,n), s \in S \tag{2}$$

$$y(i,j) \in \{0,1\}, i, j \in V, e_{ij} \in E \tag{3}$$

$$\sum_{i=1}^{n} x(f_i, s) \times c(f_i) \leqslant C_s \tag{4}$$

$$y(i,j) \times B_t \leqslant B(e_{ij}), \forall i, j \in V, e_{ij} \in E \tag{5}$$

$$\sum_{f_k \in F_S} x(f_k, s) \times h(f_k) \leqslant H_s, \ \text{if} \sum_{f_k \in F_S} x(f_k, s) \times h(s, v) = 1; \forall v \in V, s \in S \tag{6}$$

$$y(i,v) = y(v,j) = 1, \ \text{if} \sum_{f_k \in F_S} x(f_k, s) \times h(s, v) = 1; \forall v \in V, s \in S \tag{7}$$

$$\sum_{f_k \in F_H} z(f_k, v) \times h(f_k) \leqslant H_v, \ z(f_k, v) \in \{0,1\} \tag{8}$$

$$y(i,v) = y(v,j) = 1, \ \text{if} \sum_{f_k \in F_H} z(f_k, v) = 1 \tag{9}$$

其中，式(1)中的 α 和 β 是权重调节系数，用于将链路开销和功能部署开销统一到同一个优化目标中；式(2)和式(3)是所要求解的功能部署和路径选择参数；式(4)表示服务器的计算存储资源约束；式(5)表示链路带宽约束；式(6)和式(7)表示采用硬件加速卡资源做加速的功能部署约束以及流量传输约束，即流量一定流经与功能部署服务器相连的转发节点；式(8)和式(9)表示采用转发节点中加速表做加速的功能部署约束以及流量传输约束，即流量一定流经与功能表项部署的转发节点，其中，$z(f_k, v)$表示功能 f_k 是否将功能表项部署于转发节点 v 中的加速表内。

在 VDOP-HA 问题中，不仅要求解出 VNF 所要部署的服务器位置以及流量的转发路径，还需求解出 VNF 部署于服务器之后将软件功能的处理逻辑映射至硬件加速资源的映射参数。例如，对于图 2 所示的拓扑而言，VDOP-HA 需要求解的参数如图 3 所示。

首先，所有的功能都需要以 VM（或 docker）的形式部署于服务器上，这是所需要求解的第一层参数，即 VM 部署参数 $x(f_i, s)$；其次，部署于 VM 中的功能有一部分是可以将其功能表项或处理逻辑映射至硬件加速资源的，这是所需求解的第二层参数，即功能到加速资源的映射参数 $z(f_k, v)$，其中，对于 $f_i \in F_S$ 的功能，其加速资源就直接位于其所在服务器的加速卡中，因此，这里的映射参数只是到交换机加速表的映射参数；最后，数据流需要（依次）流经这些功能，形成一条从源到目的节点的路径，这是所需求解的第三层参数，即路径选取参数 $y(i, j)$，其中，若功能被映射至加速资源，则流量要流经其加速资源所在节点而非其 VM 部署的节点。

4 加速资源编排算法

首先，证明了 VDOP-HA 问题是 NP-难问题；其次，针对该问题进行了启发式算法设计；最后，分析了算法的时间复杂度。

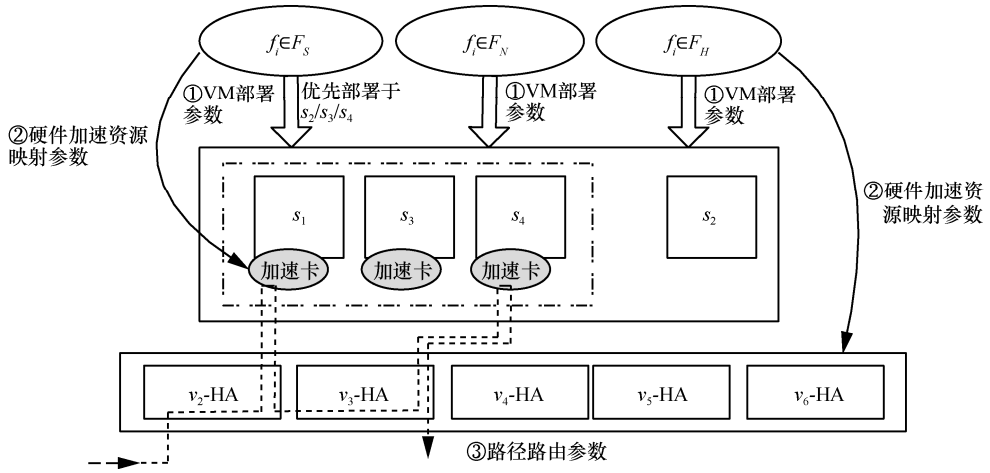


图 3　VDOP-HA 问题求解过程示意

4.1　问题分析

为简化 VDOP-HA 问题，本节首先将 VDOP-HA 模型进行简化，假设网络链路的带宽总是满足业务需求。

假设 2　网络的链路带宽总满足业务需求。由于在算法运行时可将不满足带宽需求的链路从拓扑中去掉（该转化过程在多项式时间内即可完成）并进行求解，因此，该假设并未改变该问题的性质以及求解难度，而仅仅是简化了求解步骤。

定理 1　VDOP-HA 问题是 NP-难问题。

证明　已知哈密顿环路问题（Hamiltonian cycle problem）是 NP-难问题[19]，即对于有向图 $G=(V, E)$，判断其是否存在包含所有节点且只包含一次的简单环路。对于图 $G=(V, E)$的 VDOP-HA 问题，其一个特例可在多项式时间内归约到哈密顿环路问题，如图 4 所示。

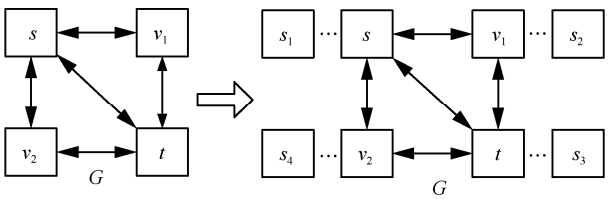


图 4　VDOP-HA 到哈密顿环的归约示例

1) 每个节点 v 均有 NFV 服务器相连，即对于$\forall s \in S$，$v \in V$，均有 $h(s,v)=1$。

2) 每个功能 f 所需要的计算存储需求为 $c(f)$，加速资源需求为 $h(f)$均相同。

3) 每个服务器节点的计算资源容量 $C_s=c(f)$，加速卡资源容量 $H_s=h(f)$。

4) 用户流量需求 $T=\{v_s, v_t, (f_1,f_2,\cdots,f_n), B_t|s, t \in V, f_i \in F, 1 \leqslant i \leqslant n\}$，其中 $n=|V|$。

此时，如果 G 存在哈密顿环路，则 VDOP-HA 问题的特例存在最短路径，且该路径即为哈密顿环，此时，各个功能分别部署于各个节点的服务器中；反之亦然。因此，VDOP-HA 问题也是 NP-难问题。

VDOP-HA 问题是 NP-难问题，难以在多项式时间内求出问题的最优解。现有功能部署和编排的启发式算法主要求出的是功能所要部署的服务器位置参数以及流量的转发路径参数，但在 VDOP-HA 问题中，还需求解出功能部署于服务器之后将软件功能的处理逻辑映射至硬件加速资源的映射参数。因此，在现有研究的基础上，本文主要考虑求解 VNF 到硬件加速资源的映射参数以及流量的路由参数，即仅考虑能够映射至硬件加速资源的功能（$f_i \in \{F_S, F_H\}$）的部署与编排问题。

4.2　算法设计

需要加速资源映射的功能分为 2 种：1) 可映射至服务器加速卡的功能，2) 可映射至交换机内加速表资源的功能。对这 2 种功能的映射原则表述如下。

1) 当有空闲硬件加速资源时，将能够映射至硬件加速资源的功能优先映射至相应硬件加速资源。由于功能经过硬件加速处理后的处理速率相比基于软件的处理速率要提升几十甚至上百倍，因此，虽然将功能映射至加速资源进行处理可能会改变传输路径，从而带来额外的路径时延，但是相比于硬件加速带来的处理性能提升和处理时延降低，其

额外的路径时延可以忽略不计。

2) 当功能既可映射至交换机加速表又可映射至服务器加速卡时，优先将功能映射至交换机加速表。这主要是由于：首先，服务器加速卡资源的成本更高；其次，转发节点的节点数多且其加速表资源容量更大；最后，加速卡相比交换机内的加速表的灵活性更高。因此，将其承载交换机加速表无法承载的功能处理将带来更高的效用。

基于以上原则，本节设计了加速卡优先部署的启发式（HCPD, HA card-prior deployment）算法，如算法 1 所示。由于服务功能链一般不超过 5 个功能，功能 $f_i \in F_H$ 的个数一般不超过 3 个，而且转发节点的加速表 $f_i \in F_H$ 一般均能够满足功能的映射需求，因此，可优先满足功能 $f_i \in F_S$ 的硬件加速需求。算法的基本思路是：首先，依次搜索确定能够承载功能 $f_i \in F_S$ 的加速卡所在的服务器位置，当有多个服务器均满足要求时，选取相距较近的服务器进行承载，此时，从源节点到最后一个所选服务器的路径也随之确定；然后，计算出一条从最后一个所选服务器到目的节点的（最短）路径；最后，再将功能 $f_i \in F_H$ 的规则表项映射至路径上满足功能链功能顺序约束的转发节点的加速表中。

算法 1 HCPD 算法

1) $N_S=|\{f|f\in F_S\}|$, $N_H=|\{f|f\in F_H\}|$, i=1, j=1, v_temp=v_s, path =[]; // path 记录流量流经的路径

2) while $i \leqslant N_S$ ($f_i \in F_S$) //将 $f_i \in F_S$ 的处理逻辑映射至服务器的加速卡资源

3) start_node = v_temp; //从 start_node 开始找距离最近的可映射的硬件加速卡

4) deploy_fs (start_node, f_i, s); //deploy_fs 函数选取邻近的服务器部署 f_i

5) $i = i$ +1, v_temp=(v|$h(s,v)$=1);

6) Addroute(path, v_temp}; //将搜索的路径加入 path 中

7) Delete(G, path); //并将该链路从图 G 中删去以保证路径无重复使用

8) end while

9) Findroute (G, v_{v_temp}, v_t, v_set);//从最后一个服务器节点到目的节点的最短路径，G 已删除已用链路

10) while $j \leqslant N_H$ ($f_j \in F_H$) //将 $f_i \in F_H$ 的规则表项映射至转发节点内的加速表资源

11) start node = v_s;

12) deploy_fh (start_node, f_j, v); //在路径上将 f_j 依顺序部署于相应的转发节点中的加速表中

13) $j = j$ +1;

14) end while

其中，deploy_fs 函数依次搜索离 start_node 最近的能够满足功能加速需求的服务器部署 f_i，若无加速卡可承载 f_i，则选取相近的服务器部署（此时该功能并未被加速），具体步骤如下。

Function deploy_fs (start_node, f_i, s)

1) find nearest s, s.t. $H_s \geqslant h(f_i)$ //寻找离 start_node 最近的满足功能硬件加速需求的加速卡所在的服务器

2) if exist s //如果存在，则将功能映射至服务器的加速卡中

3) $x(f_i,s)$=1;

4) else //如果不存在，则将功能就近部署

5) find nearest s, $x(f_i,s)$=1;

6)end if

对于图 2 所示的网络拓扑，假设各个服务器只能部署一个功能，如果用户需要一条从 v_1 到 v_7 包含 3 个功能（L7 Firewall–NAT–DPI）的服务功能链，首先，L7 Firewall 功能和 DPI 功能属于集合 $f_i \in F_S$，优先部署于含有加速卡的服务器 $s_1/s_3/s_4$，从 v_1 开始依次搜索含有加速卡的服务器，得到 s_1 和 s_4 承载 L7 Firewall 功能和 DPI 功能，此时，流经 2 个功能的路径也随搜索过程而确定（$v_1 \rightarrow v_2 \rightarrow v_3$）；然后，计算从 v_3 到 v_7 的最短路径，最终得到 $v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_7$ 的流量传输路径；最后，将 NAT 功能部署于含有加速表资源的转发节点 v_2 或 v_3 中（部署在 2 个节点中的任意一个均可保证服务链的顺序）。

4.3 算法性能分析

HCPD 算法中 $f \in F_S$ 的映射部分按距离从小到大的顺序依次搜索加速卡资源，最短距离可采用 Dijkstra 算法计算，其基于斐波那契堆实现的时间复杂度为 $O(|E|+|V|\log|V|)$，一次搜索最多遍历$|S|$个服务器节点，因此，该部分的时间复杂度为 $O(|E|+|S|+|V|\log|V|)$；选路的 Findroute()部分也可采用 Dijkstra 算法求解，时间复杂度为 $O(|E|+|V|\log|V|)$；功能 $f \in F_H$ 的映射部分采用普通搜索即可，时间复杂度为 $O(|V|)$。HCPD 算法的 $f \in F_S$ 映射部分需要搜索 n 次，因此，忽略低阶项后 HCPD 算法的时间复杂度为 $O(n(|E|+|S|+|V|\log|V|))$，其中，n 为用户需求的服务功能链长度。

除了时间复杂度分析外，编排成功率也是衡量 HCPD 算法的重要指标。编排成功率，即在 VDOP-HA 问题有解情况下，HCPD 计算出解（不一定是最优解）的概率。影响 HCPD 算法编排成功率的决定性因素是网络拓扑以及加速资源的分布位置，因为将功能 $f \in F_S$ 映射至加速资源时采用贪心策略，即最短路径策略搜索，所以在映射一个功能完成之后，并不能够保证该功能映射到的服务器加速卡在不复用链路的情况下，能够到达下一个部署功能的服务器节点，而且也不能保证从 $f \in F_S$ 的最后一个功能部署服务器节点到目的节点有路径。例如，在图 2 所示的拓扑中，从 v_1 到 v_7 的流请求中，有 3 个功能 $f \in F_S$ 的映射需求，按照 HCPD 算法搜索到了 s_1 和 s_4 中的加速卡承载前 2 个功能，但由于转发链路不可复用，因此，从 s_4 已经不能够再到达下一个加速卡所在的服务器 s_3；而事实上，这个问题的最优解是 $v_1 \rightarrow v_5 \rightarrow v_6(s_3) \rightarrow v_4 \rightarrow v_2(s_1) \rightarrow v_3(s_4) \rightarrow v_7$。

5　性能仿真

本节分别对所提算法的时间开销和编排成功率进行了仿真验证，并基于 SDN 控制器和 NetFPGA-10G[20]平台进行了原型系统仿真。

5.1　算法性能仿真

采用如图 5 所示的网络拓扑，网络中有 11 个转发节点（v_1~v_{11}），6 个服务器节点（s_1~s_6，其中 5 个节点包含加速卡资源）。为不失一般性，假设所有转发节点的硬件加速表资源容量相同。仿真平台采用 Matlab 搭建，计算机处理器为 Inter® Core™ i7-3770、4 GB RAM。设有 10 种功能，其中，5 种功能 $f_i \in F_S$，加速资源需求服从[1,10]之间的随机分布；5 种功能 $f_i \in F_H$，加速资源需求服从[5,15]之间的随机分布。令每个转发节点的加速资源容量为 20，服务器内加速卡资源容量为 10。

考虑到现有硬件加速资源管理编排的研究缺乏算法层面的讨论，因此，本文只能与现有服务功能链部署选路方面的算法进行对比。本文选取了具有代表性的 2 种思路进行对比：ProvisionTraffic[14]以功能链为导向，将不同功能所能够部署的节点分列到不同的级中，各级各节点之间的传输权值由节点之间的最短路径确定，然后采用维特比算法求解出最佳部署位置及转发路径；Greedy 则以路径为导向，采用广度或深度优先的搜索思路依次按照服务链上的功能搜索，选取最近的可部署节点进行部署。仿真不同服务链长度下算法的时间开销，随机生成数据流传输请求，重复 100 次，得到 3 种算法的平均时间开销，如图 6(a)所示。

实验结果表明，3 种算法的处理时延均与服务功能链长度呈线性关系，结果符合上文对算法复杂度的理论分析，且 ProvisionTraffic 算法的处理时延最高，因为其算法要计算各级各节点之间的最短路径，这最多需要计算 $\frac{|S|^2}{2}$ 次，而 HCPD 算法和 Greedy 算法最多计算 n 次，其中，n 为用户需求的服务功能链长度。Greedy 算法的处理时延比 HCPD 高近 30%，因为 HCPD 区分了 $f \in F_S$ 和 $f \in F_H$ 的情况，对功能 $f \in F_H$ 的映射进行了简化处理，而 Greedy 算法不加区分地进行搜索，带来了额外的时间开销。

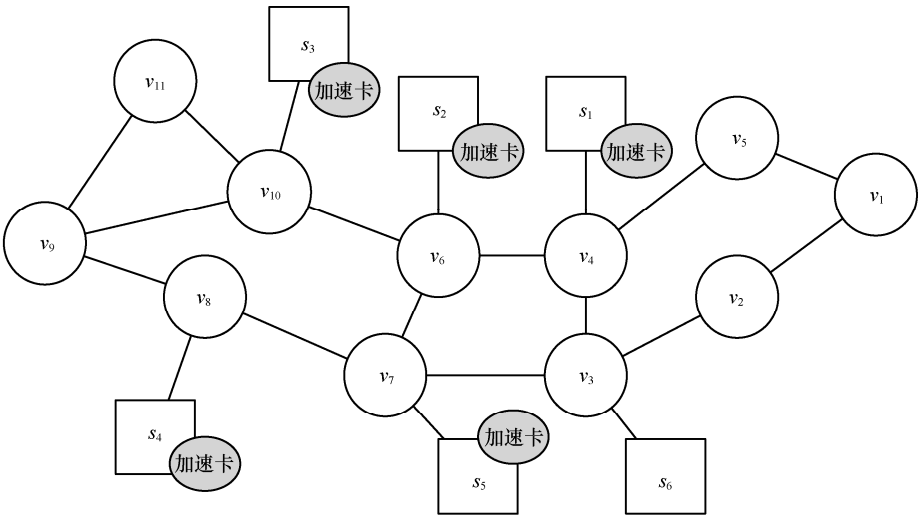


图 5　算法性能仿真网络拓扑

为仿真算法的编排成功率，随机生成 1 000 次数据流请求，得到 3 种算法的编排成功率，即能够完成硬件加速资源映射并找到一条从源到目的地址传输路径的概率，实验结果如图 6(b)所示。由仿真结果可知，在相同请求次数下，3 种算法的编排成功率均随功能链长度增长而降低。这是由于功能越多，其所需要的服务器节点也就越多，满足功能顺序的条件下选路成功的概率也就越低。在相同功能链长度下，ProvisionTraffic 与 Greedy 算法的编排成功率相当，HCPD 算法的编排成功率要比这 2 种算法高出近 30%。这是由于前 2 种算法的思想相近，针对功能链上每一个功能依次进行贪心搜索；而 HCPD 则根据功能类型进行有差别搜索，且其搜索的次数要低于前 2 种算法，因此成功率要高。

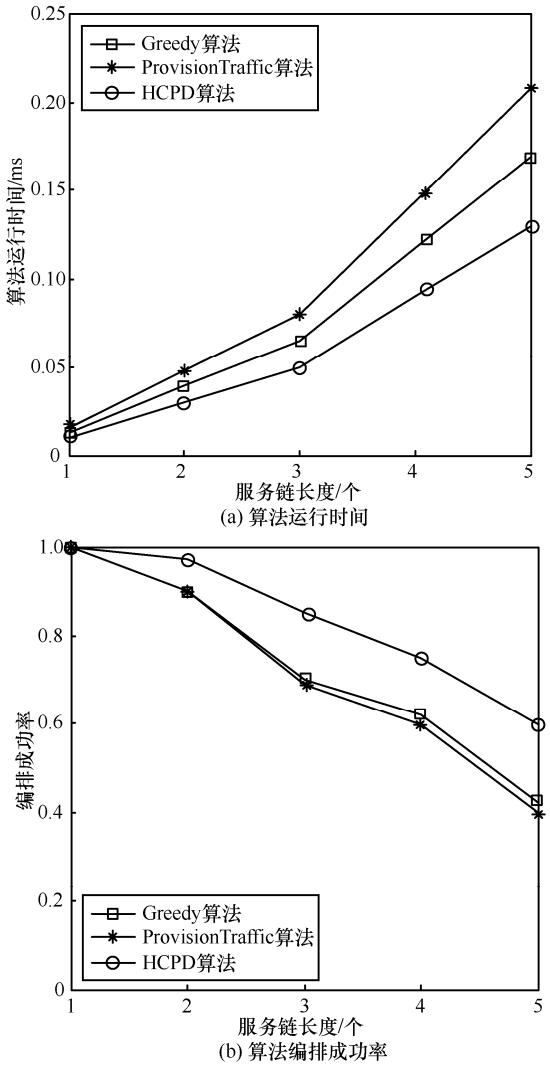


图 6 算法性能对比

从 3 种算法的对比结果可以看出，在相同请求下，与现有算法相比，HCPD 算法能在更短的处理时延下得到更高的编排成功率。综合时间复杂度和编排成功率来看，HCPD 算法性能更优。

5.2 原型系统仿真

为实现对交换机中硬件加速表的管控，需要有单独的 HAM 连接交换机，而目前 OpenFlow 交换机设计并不支持 HAM 的连接。因此，本文利用 OpenFlow 1.3 协议中的多控制器技术来实现 MSC 的原型系统。多控制器技术允许有多个控制器同时控制转发元件，其中，有一个主控制器（MASTER）和多个从控制器（EQUAL）。多个控制器均可向转发元件发送指令，交换机则可向任意控制器发送回复或错误信息。多控制器技术的提出最初是为了提升控制器的可靠性，使交换机与一个控制器连接的中断不会影响对交换机处理行为的控制。在 MSC 的原型实现中，这种多控制器技术正好提供了 HAM 的一个实现方案，即主控制器实现 SDN 控制器的任务，包括网络状态收集、处理网络事件、确保网络连接、网络流量调度等；从控制器实现 HAM，用于配置交换机中的加速表。

MSC 原型系统仿真环境如图 7 所示，SDN 控制器和 HA 管理器都基于 Floodlight 控制器实现；OpenFlow 交换机和 NFV 服务器均采用 NetFPGA-10G 板卡实现，其中，OpenFlow 交换机中的加速表最大支持容纳 128 条加速规则，NFV 服务器加速卡最大可容纳 64 条自定义加速规则，并支持 32 bit 的正则匹配。为简化与现有算法的对比，本文假设算法中的拓扑信息已经给定，不需要通过控制器来获取，算法仅用来根据数据流的需求来计算功能映射位置及转发路径。

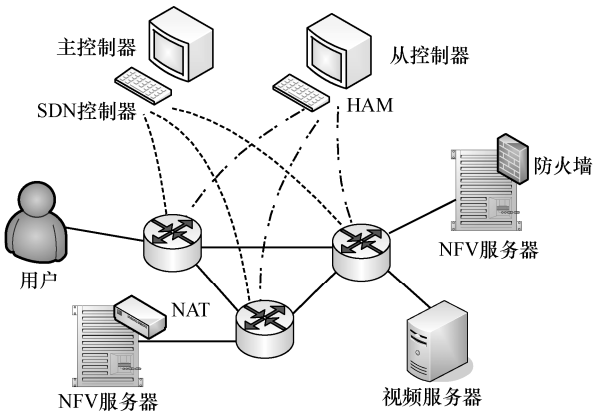
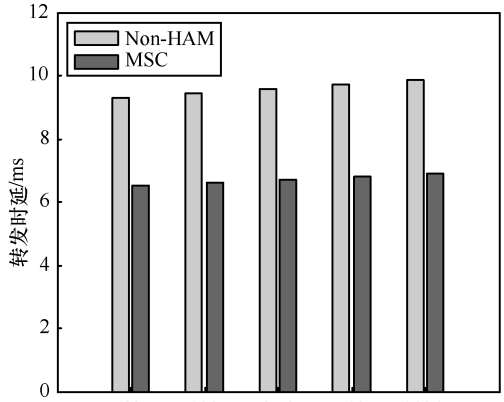
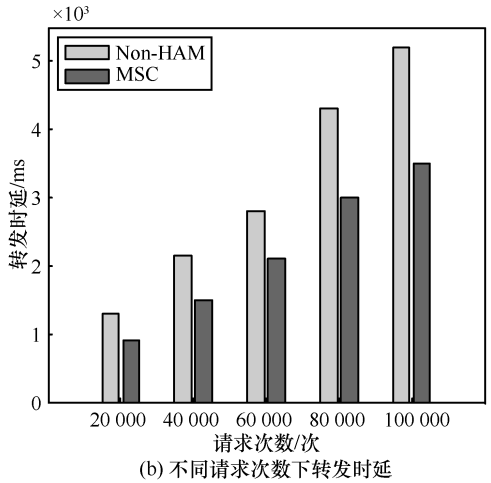


图 7 MSC 原型系统仿真环境

原型系统仿真选取从用户到视频服务器的数据流，针对有 MSC 以及未使用 HAM 的加速架构的端到端转发性能进行了测试。首先测试了不同数据分组长度下用户机到视频服务器的传输时延，然后测试了不同请求次数下的传输时延，实验结果如图 8 所示。其中，未使用 HAM 的架构是当前通用的 NFV 加速架构，其 VNF 运行于 SDN 控制器之上，这种架构可以看作 HAM 和 SDN 控制器集成到一个控制器之中。由于文献[6]已经给出了防火墙的实现，为简化实验复杂度，选取防火墙和 NAT 作为数据流的功能处理需求。由图 8 仿真结果可知，HAM 的引入使 MSC 的转发时延相比 DPAK 降低了约 30%，这是由于不添加 HAM 进行分离式控制时，VNF 处理的数据流会经过控制器进入上层的 VNF，使原本负责网络状态控制和网络连接的 SDN 控制器的负载增大，导致流时延增大。实验仅采用 2 个 VNF 进行仿真，可以预知，当功能数量增多时，MSC 的性能优势将会更加明显。



(a) 不同数据分组长度下转发时延



(b) 不同请求次数下转发时延

图 8　有 MSC 以及未使用 HAM 的端到端转发性能对比

6　结束语

本文在所提硬件加速资源统一管控平台的基础上，研究了如何将上层用户的功能需求映射到相应的硬件加速资源中进行加速处理的问题。首先，将包括 NFV 服务器节点中的加速卡资源和转发节点中的加速表资源在内的网络硬件加速资源统一到网络功能部署和编排模型中；然后，对该问题模型进行了理论分析，得出了该问题为 NP-难问题的结论；最后，通过对该模型简化并设计了相应的启发式算法，对功能部署和编排问题进行了求解。仿真实验和原型系统仿真表明，所提机制具有更低的时间开销，能够获得更高的编排成功率，且对未来 SDN/NFV 网络中硬件加速资源的管控和编排具有一定实用价值。

参考文献：

[1] HAN B, GOPALAKRISHNAN V, JI L, et al. Network function virtualization: challenges and opportunities for innovations[J]. IEEE Communications Magazine, 2015, 53(2): 90-97.

[2] MIJUMBI R, SERRAT J, GORRICHO J, et al. Network function virtualization: state-of-the-art and research challenges[J]. IEEE Communications Surveys & Tutorials, 2016, 18(1): 236-262.

[3] GE X, LIU Y, DU D, et al. OpenANFV: accelerating network function virtualization with a consolidated framework in Openstack[J]. ACM Computer Communications Review, 2014, 44(4): 353-354.

[4] LI B J, TAN K, LUO L Y, et al. ClickNP: highly flexible and high performance network processing with reconfigurable hardware[C]// ACM SIGCOMM'16. 2016: 1-14.

[5] MARIAS J, GARAY J, TOLEDO N, et al. Toward an SDN-enabled NFV architecture[J]. IEEE Communications Magzine, 2015, 53(4): 187-193.

[6] BI J, ZHU S Y, SUN C, et al. Supporting virtualized network functions with stateful data plane abstraction[J]. IEEE Network, 2016, 30(3): 40-45.

[7] CHI P, HUANG Y, LEI C. Efficient NFV deployment in data center networks[C]//IEEE ICC. 2015: 5290-5295.

[8] XIA M, SHIRAZIPOUR M, ZHANG Y, et al. Network function placement for NFV chaining in packet/optical data centers[J]. Journal of Lightwave Technology, 2015, 33(8): 1565-1570.

[9] 汤红波, 袁泉, 卢干强, 等. 一种支持节点分割的 vEPC 虚拟网络功能部署模型[J]. 电子与信息学报, 2017, 39(3): 546-553.
TANG H B, YUAN Q, LU G Q, et al. A model for virtualized network function deployment based on node-splitting in vEPC[J]. Journal of Electronics & Information Technology, 2017, 39(3): 546-553.

[10] COHEN R, LEWIN L, NAOR J, et al. Near optimal placement of virtual network functions[C]//IEEE Conference on Computer Communications (INFOCOM). 2015: 1346-1354.

[11] 段通, 兰巨龙, 程国振, 等. 基于元能力的 SDN 功能组合机制[J]. 通信学报, 2015, 36(5): 156-166.
DUAN T, LAN J L, CHENG G Z, et al. Functional composition in software-defined network based on atomic capacity[J]. Journal on Communications, 2015, 36(5): 156-166.

[12] GUSHCHIN A, WALID A, TANG A. Scalable routing in SDN- enabled networks with consolidated middleboxes[C]// HotMiddlebox'15. 2015: 55-60.

[13] DWARAKI A, WOLF T. Adaptive service-chain routing for virtual network functions in software-defined networks[C]// HotMiddlebox'16. 2016: 32-37.

[14] BARI M., CHOWHURY S., AHMED R, et al. On orchestrating virtual network functions[C]//International Conference on Network and Service Management. 2015: 50-56.

[15] 刘彩霞, 卢干强, 汤红波, 等. 一种基于 Viterbi 算法的虚拟网络功能自适应部署方法[J]. 电子与信息学报, 2016, 38(11): 2922-2930.
LIU C X, LU G Q, TANG H B, et al. Adaptive deployment method for virtualized network function based on Viterbi algorithm[J]. Journal of Electronics & Information Technology, 2016, 38(11): 2922-2930.

[16] LI Y, ZHENG F, CHEN M, et al. A unified control and optimization framework for dynamical service chaining in software-defined NFV system[J]. IEEE Wireless Communications, 2015, 22(6): 15-23.

[17] MA W, SANDOVAL O, BELTRAN J, et al. Traffic aware placement of interdependent NFV middleboxes[C]//IEEE INFOCOM. 2017: 1-12.

[18] BRONSTEIN Z, ROCH E, XIA J, et al. Uniform handling and abstraction of NFV hardware accelerators[J]. IEEE Network, 2015, 29(3): 22-29.

[19] CORMEN T, LEISERSON C, RIVEST R, et al. Introduction to algorithms(3rd)[M]. MIT Press, 2009: 878-879.

[20] ZILBERMAN N, AUDZEVICH Y, KALOGERIDOU G. NetFPGA-rapid prototyping of networking devices in open source[C]// SIGCOMM'15. 2015: 363-364.

[作者简介]

段通（1992-），男，河南驻马店人，国家数字交换系统工程技术研究中心博士生，主要研究方向为网络功能虚拟化、可编程网络硬件等。

兰巨龙（1962-），男，河北张家口人，国家数字交换系统工程技术研究中心教授、博士生导师，主要研究方向为未来信息通信网络关键理论与技术。

胡宇翔（1982-），男，河南周口人，博士，国家数字交换系统工程技术研究中心副教授，主要研究方向为未来网络关键技术、网络智慧化等。

范宏伟（1994-），男，河南长葛人，国家数字交换系统工程技术中心硕士生，主要研究方向为网络功能虚拟化、硬件加速等。