

一种面向航空集群机载网络的分布式更新方法

陈 坤, 吕 娜, 朱海峰, 方 宇

(空军工程大学信息与导航学院, 西安, 710077)

摘要 针对当前集中式网络更新方法更新时间长且过度依赖控制器的问题, 为降低网络更新时间和减缓控制器负荷, 提出一种面向航空集群机载网络的分布式更新方法。设计改进的更新消息, 控制器通过生成更新依赖图进行更新消息的构造, 将更新依赖关系编码到更新消息中下发到数据平面; 而后交换机分布式执行更新操作, 利用询问和通知消息进行更新顺序协调, 从而在数据平面实现分布式网络更新。仿真实验表明: 与现有集中式更新方法相比, 所提分布式更新方法能够通过降低更新期间的节点交互次数来减少更新协调时间, 从而降低业务流更新时间以及整个网络更新时间。

关键词 航空集群; 机载网络; 软件定义网络; 数据平面; 分布式更新

DOI 10.3969/j.issn.1009-3516.2020.05.016

中图分类号 TP302 **文献标志码** A **文章编号** 1009-3516(2020)05-0099-07

A Distributed Updating Method Geared to the Airborne Network of Aviation Swarm

CHEN Kun, LYU Na, ZHU Haifeng, FANG Yu

(Information and Navigation College, Air Force Engineering University, Xi'an 710077, China)

Abstract Aimed at the problems that the current centralized network update method remains to be long in update time and excessively dependent on the controller, and in order to save the network update time and slow down the controller load, a distributed update method geared to the airborne network of aviation swarm is proposed. A modified update message is designed. The controller carries out the construction of update message by generating the update dependency graph, encodes the update dependency relationship into the update message, and sends it to the data plane. And then the switch distributed update operation is made by using the query and notification message to update the order coordination, achieving the distributed network update in the data plane. The simulation results show that compared with the existing centralized update methods, the proposed distributed update method can reduce the update coordination time by reducing the number of node interactions during the update period, saving much update time of the business flow and the entire network.

Key words aviation swarm; airborne network; software-defined networking; data plane; distributed update

收稿日期: 2020-04-17

基金项目: 国家自然科学基金(61703427)

作者简介: 陈 坤(1996—), 男, 河南固始人, 硕士生, 主要从事航空数据链研究。E-mail: 648315977@qq.com

通信作者: 吕 娜(1970—), 女, 陕西西安人, 教授, 主要从事航空数据链研究。E-mail: lvnn2007@163.com

引用格式: 陈坤, 吕娜, 朱海峰, 等. 一种面向航空集群机载网络的分布式更新方法[J]. 空军工程大学学报(自然科学版), 2020, 21(5): 99-105. CHEN Kun, LYU Na, ZHU Haifeng, et al. A Distributed Updating Method Geared to the Airborne Network of Aviation Swarm[J]. Journal of Air Force Engineering University (Natural Science Edition), 2020, 21(5): 99-105.

航空作战力量在现代战争中往往占据着一定战略地位,建设和发展航空网络的重要性不言而喻。现代军事思想不断衍生发展,未来航空集群作战的理念已被广泛认同^[1-2]。为满足航空集群多平台多任务灵巧协同作战的需求,有研究人员将软件定义网络(Software-defined Networking, SDN)的思想运用在机载网络中,提出软件定义航空集群机载网络(Software-defined Airborne Network of Aviation Swarm, SDAN-AS),赋予了机载网络灵活的网络控制和管理、自动化可编程的业务部署,从而提供与多任务灵活耦合的网络服务^[3]。SDN 的引入能够突破单个作战平台的效能瓶颈,多平台之间展开的灵活、高效协同能够弥补单一平台的弱点,从而形成优势互补的强大体系作战能力^[1-3]。

在 SDAN-AS 中,也同样避免不了 SDN 固有的网络一致性更新问题,一致性更新是所有集中式控制的网络都面临的问题,即如何在改变网络配置期间保持网络的一致属性(如无路由黑洞、无转发循环和无链路拥塞)^[4-6]。但由于控制器与交换机之间存在着固有延迟,即使同时发送更新指令,指令也难以同时到达所有的交换机。即使同时到达所有的交换机,完成更新的时间也难以同步。因此无法同时进行整个网络的更新,现有更新方法往往需要通过多个轮次来进行^[7-8]。

网络的一致性更新近些年来受到广泛关注,研究人员针对更新期间的各种一致属性开展了各类研究。文献[9]首次提出两阶段更新方法,实现了更新过程中的数据包一致属性,保证转发路径不会新旧两套转发规则的混合。文献[10]基于两阶段更新提出增量两阶段更新方法,牺牲更新时间来降低规则开销。文献[11]提出 SWAN 更新系统,采用预留链路带宽的方法实现无链路拥塞的更新。文献[12]提出 Dionysus 动态更新系统,通过构建更新依赖图减少网络更新时间,从而加快更新速度。以上几种现有更新方法在一定程度上能够较好地保持网络更新期间的各种一致属性,但都属于集中式更新方法,往往分为很多个轮次,通过控制器与数据平面的频繁交互来协调交换机的更新顺序。控制器下发更新指令到网络中的部分交换机,收到更新指令的交换机完成更新后向控制器发送确认消息,控制器在接收到所有的确认消息后完成更新的一个轮次,开始进入下一个轮次,下发新的更新指令到网络中的交换机。在集中式更新方法中,控制器与交换机之间的频繁交互将会大大增加更新的完成时间^[13-14]。同时集中式更新方法在更新过程中全程依赖控制器的计算与调度,SDAN-AS 中的控制器搭载于资源

有限的空中平台,难以承载较高的负荷。

本文针对上述集中式更新方法的缺点,提出一种面向航空集群机载网络的分布式更新方法,设计改进的更新消息格式,将更新依赖关系编码到更新消息中,采用询问和通知 2 种消息实现数据平面的分布式交互,将控制器的更新顺序协调功能转移到数据平面上进行,从而实现快速一致的分布式更新,有效降低网络更新时间。对于追求高实时性、高动态组网的 SDAN-AS 而言,降低网络更新时间从而减少更新期间的网络性能下降具有重要意义。

1 问题描述

1.1 网络模型

为方便描述,首先引入本文的网络模型抽象。在 SDAN-AS 中,通常由大型空中平台担任网络控制器。由若干小型空中平台担任网络交换机共同构成网络的数据平面。因此可以将 SDAN-AS 的数据平面抽象为由若干个节点和链路连接起来的无向图 $G = (V, E)$ 。其中其中 $V = \{v_1, v_2, \dots, v_m\}$ 表示网络中交换机的集合, $E = \{e_1, e_2, \dots, e_n\}$ 表示链路集合。对于 $\forall e \in E$, C_e 表示该链路的容量。 $\mu = \{\mu_{e_1}, \mu_{e_2}, \dots, \mu_{e_n}\}$ 表示网络链路利用率的集合。

本文采用标准的流模型, $F = \{f_1, f_2, \dots, f_m\}$ 表示网络中的业务流集合,其中每条业务流 f 表示在一个源节点和一个目的节点之间的一组数据包的集合,其流量需求表示为 d_f 。控制器可以通过定期收集数据平面信息或根据带宽资源分配计算出网络中业务流需求的估计值。业务流采用基于隧道的转发,对于每条业务流,采用 VLAN 标签建立多条从源节点到目的节点的隧道,隧道上的每个节点根据匹配 VLAN 标签的流表规则进行相应的转发。

1.2 网络更新问题

G 表示网络状态,即决定网络中所有业务流的数据包转发规则的集合。网络控制器定期执行流量工程,根据业务需求变化和网络拓扑信息,执行路由算法或流量调度算法计算出新的路由转发规则, $P_o(f)$ 表示流 f 在当前网络状态 G_o 中的旧转发路径, $P_n(f)$ 表示流 f 在目标网络状态 G_n 中的新转发路径。执行网络更新将网络从状态 G_o 转变为状态 G_n ,将流量从旧转发路径迁移到新转发路径上。

由于更新的过程不是瞬态,而是异步分布式的过程。需要谨慎地协调交换机流表规则的更新顺序,否则交换机流表规则的不一致将会导致网络配置的不一致,从而产生如路由黑洞、循环转发、链路

拥塞等问题^[15-16],严重影响更新期间的网络性能。为了避免网络配置的各种不一致,需要网络更新算法满足对应的一致属性。本文所设计的网络更新方法主要满足以下3种一致属性:

1)无路由黑洞一致性:更新期间网络中所有数据包都不会被丢弃。

2)无转发循环一致性:更新期间网络中所有数据包的转发不会产生环路^[17]。

3)无链路拥塞一致性:更新期间网络中所有链路的负载都不会超过其容量^[11-18]。

由于网络更新是异步的过程,可以将每一次网络更新 U 序列化为一系列的更新操作的集合,表示为 $U=\{o_1, o_2, \dots, o_n\}$,其中每一个更新操作 o 表示插入、修改或删除一条流表规则。为了保持更新期间的网络配置一致属性,最小化更新操作之间产生的冲突,需要安排更新操作以合适的顺序执行。由于更新操作之间具有相互依赖的关系,根据依赖关系可以建立起更新依赖图,从而根据更新依赖图计算出合适的更新顺序。

2 更新消息设计

为减缓控制器负载,直接在数据平面实现网络的分布式更新,本节设计了改进的更新消息格式,将更新依赖关系编码到更新消息中。

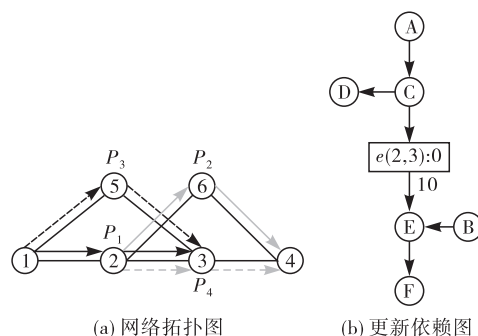
2.1 更新依赖图

为了更好地阐述改进的更新消息设计,首先引入更新依赖图的概念。更新依赖图是描述网络更新操作之间依赖关系的有向图。根据当前网络状态 G_o 以及目标网络状态 G_n ,可以计算出从 G_o 更新到 G_n 的过程中所需要执行的一系列更新操作。而后根据更新操作之间的相互依赖关系、网络链路带宽资源以及更新操作对链路带宽资源的需求等信息,可以为每一个给定当前网络状态 G_o 和目标网络状态 G_n 的完整网络更新过程构建出满足其一致性要求的更新依赖图。

如图1所示,图1(a)中存在2条业务流,用不同的颜色表示,分别是源节点为1、目的节点为3的 f_1 和源节点为2、目的节点为4的 f_2 。每条业务流的需求都为10个单位,网络中的每条链路的链路容量为10个单位。其中实线部分表示当前网络状态 G_o ,虚线部分表示目标网络状态 G_n 。

根据图1(a)的网络拓扑图生成的更新依赖图如图1(b)所示,其中圆形节点表示操作节点,操作节点与操作的对应关系如表1所示;矩形节点表示链路资源节点,表示链路 $e(2,3)$ 上的可用链路带宽

资源为0,指向性直线表示节点之间的依赖关系。操作C的执行将会在链路 $e(2,3)$ 上释放10个单位的链路带宽资源,操作E的执行需要链路 $e(2,3)$ 上具有10个单位的可用链路带宽资源。



(a) 网络拓扑图 (b) 更新依赖图

图1 更新依赖图示例

表1 依赖图节点含义表示

序号	操作
A	在节点5上添加路径 p_3
B	在节点3上添加路径 p_4
C	在节点1上修改 f_1 的流量分配权重
D	删除节点2上的路径 p_1
E	在节点2上修改 f_2 的流量分配权重
F	删除节点6上的路径 p_2

2.2 改进型更新消息

为了实现分布式的快速一致网络更新,本小节设计了改进型更新消息格式,将操作之间的依赖关系编码到更新消息中。网络控制器采用改进型的更新消息来进行数据平面的网络配置,通过节点之间的分布式协调来执行网络更新。

改进型更新消息中包含以下3个字段的信息。

1)操作:表示一个任意的更新操作;

2)前置条件:表示执行该更新操作所需要预先执行的前置更新操作的组合,以逻辑布尔表达式呈现,更新操作之间采用逻辑关系式进行连接,只有前置条件满足之后才可执行该操作。

3)后置操作:表示依赖于该更新操作执行的一系列更新操作的集合。

控制器根据从数据平面收集到的网络状态信息为网络更新中的每一个更新操作计算出一个对应该操作的更新消息。控制器计算完更新消息后,同时下发所有的更新消息到对应的网络节点。节点接收到更新消息后按照以下规则进行执行。

对于一个收到更新消息的节点 v :

1)只有当该更新消息中的前置条件被满足之后,才能够执行该更新消息中的操作;

2)在更新消息中的操作被执行之后,节点 v 向其后置操作中的所有更新操作所对应的节点发送通知消息。

前置条件中的逻辑布尔表达式由单个或多个更新操作通过 $\&$ 和 $\parallel$ 2 种逻辑运算符连接起来。

定义 1 运算符 $\&$: 对于任意的操作 o_1 和 o_2 , $o_1 \& o_2$ 表示 o_1 与 o_2 都被执行。

定义 2 运算符 $\parallel$: 对于任意的操作 o_1 和 o_2 , $o_1 \parallel o_2$ 表示 o_1 与 o_2 至少存在一个被执行。

例如,若更新操作 o_1 依赖于更新操作 o_2 和 o_3 的执行,则 o_1 的前置条件为 $o_2 \& o_3$, o_2 和 o_3 的后置操作都为 $\{o_1\}$ 。前置条件和后置操作的计算在第 3 节中的算法 2 进行详细介绍。

2.3 分布式更新协调

在本文的分布式更新方法中,为数据平面上的分布式更新顺序协调定义了两种消息:询问消息、通知消息。取代了集中式更新方法中控制器的更新顺序协调功能,数据平面中的节点通过询问和通知两种消息进行分布式交互,协调更新顺序,从而减少控制器与数据平面之间的频繁交互所带来的延迟。

询问消息: 当一个交换机接收到控制器发送的一条更新消息后,将会发送询问消息到其前置条件中的所有更新操作所对应的交换机,询问这些更新操作是否已执行。

通知消息: 在一条更新消息中的操作被其对应的节点执行后,该节点将会发送通知消息到其后置条件中的所有更新操作所对应的交换机,从而通知这些交换机该更新操作已经执行。

询问和通知 2 种消息能够确保更新操作在数据平面被分布式地执行。操作之间的执行顺序完全由数据平面完成,从而取代控制器的更新顺序协调功能。网络控制器只需根据更新依赖图为所有的更新操作构造对应的更新消息并下发到数据平面,即使更新消息到达的时间不同,也能够保证更新操作按照更新依赖图中的依赖关系进行执行。

对于图 1 中的情况,控制器能够根据图 1(b) 中的更新依赖图构造出每一个更新操作所对应的更新消息,如第 3 节中图 1 的更新消息构造如表 2 所示。

当网络控制器将所有的更新消息下发到数据平面的各交换机后,由于更新操作 A、B 的前置条件均不存在,因此节点 3 接收到更新消息 4 后可直接执行更新操作 B,并向其后置操作 E 所对应的节点 2 发送通知消息。同理,节点 5 在接收到更新消息 5 后可直接执行更新操作 A,并向节点 1 发送通知消息。节点 1 接收到更新消息 1 后,不能立即执行更新操作 C,只有在更新操作 A 执行之后才能更新,因此节点 1 需要向其前置条件(即操作 C)对应的节点 5 发送询问消息,当节点 1 接收到节点 5 发送的通知消息后,即可执行更新操作 C,并向其后置操作

D、E 所对应的节点 2 发送通知消息。

表 2 更新消息示例

节点序号	消息序号	操作	前置条件	后置操作
1	1	C	A	D,E
2	2	D	C	F
	3	E	B&C	
3	4	B		E
5	5	A		C
6	6	F	E	

数据平面的交换机之间利用询问和通知两种消息进行分布式交互来协调更新顺序,最终完成整个网络的分布式更新。

3 分布式更新方法

3.1 更新方法流程描述

阶段 1 控制器根据从数据平面收集到的网络拓扑信息,将网络更新分解为一系列的更新操作。

阶段 2 根据网络状态信息将一系列更新操作生成更新依赖图。再根据更新依赖图计算出每一个更新操作所对应的前置条件和后置操作,构造更新消息并下发到该更新操作对应的交换机。

阶段 3 数据平面中的交换机接收到控制器发送的更新消息后,根据更新消息中的前置条件和后置操作开始执行分布式更新,采用询问、通知两种消息进行更新顺序的协调,从而完成数据平面的分布式网络更新。

3.2 算法描述

3.2.1 更新依赖图生成

对于更新依赖图的生成,首先根据网络所需要满足的一致属性,提取出约束条件,然后根据约束条件进行更新依赖图的生成。本文主要关注无路由黑洞、无循环转发和无拥塞 3 种一致属性。

无路由黑洞一致性约束: 要确保添加隧道的操作在流量从入口交换机输入之前执行,删除隧道的操作要在流量迁移出该隧道之后执行。即修改入口交换机流量分配权重的操作依赖于添加隧道的操作,删除隧道的操作依赖于修改入口交换机流量分配权重的操作。

无循环转发一致性约束: 由于网络采用基于隧道的转发,只有建立起一条新转发路径的隧道,才能够在其上进行数据包的转发,因此一定满足无循环转发的一致性。

无链路拥塞一致性约束: 要确保更新操作的执行必须具有相应的可用链路带宽,即更新操作依赖于链路可用带宽资源,同时更新操作也会释放一些

链路带宽资源。

生成更新依赖图算法如算法1所示。

算法1:更新依赖图生成算法

输入:流集合 F ; 路径集合 P ; 流需求集合 d_f ;

输出:更新依赖图

1. for each $f \in F$ do
2. 寻找业务流 f 的入口交换机 s
3. 生成“修改 s 上的流量分配权重”操作节点, 记为 o^*
4. for each $v \in P_n(f) - s$ do
5. 生成“在 v 中添加隧道 $P_n(f)$ ”的操作节点, 记为 o_{add}
6. 添加一条从 o_{add} 指向 o^* 的边
7. end for
8. for each $v \in P_o(f) - s$ do
9. 生成“在 v 中删除隧道 $P_o(f)$ ”的操作节点, 记为 o_{del}
10. 添加一条从 o^* 指向 o_{del} 的边
11. end for
12. for each $e \in P_n(f)$ do
13. 生成对应链路 e 的资源节点, 记为 r^-
14. 添加一条附加数值 d_f , 从 r^- 指向 o^* 的边
15. end for
16. for each $e \in P_o(f)$ do
17. 生成对应链路 e 的资源节点, 记为 r^+
18. 添加一条附加数值 d_f , 从 o^* 指向 r^+ 的边
19. end for
20. end for

3.2.2 构造更新消息

在生成更新依赖图之后, 控制器需要为每一个更新操作构造出一条更新消息, 下发到数据平面中相对应的交换机。

更新消息构造的关键是计算前置条件和后置操作, 前置条件为一组更新操作的组合, 表现为逻辑布尔表达式的形式; 后置操作为一组更新操作的集合。更新消息构造算法如算法2所示。

算法2:更新消息构造算法

输入:更新依赖图

输出:更新消息

1. 初始化所有操作的前置条件和后置操作
2. for each o_i do
3. if $\exists o_j \in o_i$ 的父节点且 $o_j \in U$ then
4. o_i 的前置条件 = o_i 的前置条件 $\& o_j$
5. end if
6. end for
7. for each r do
8. if r 的可用资源 $< r$ 子节点的总需求 then
9. 计算 r 子节点优先级
10. 将 r 子节点按照优先级从高到低排列
11. for each $o \in r$ 子节点 do
12. r 的可用资源 = r 的可用资源 - o 的链路需求
13. if r 的可用资源 < 0 then

14. 根据 r 的父节点计算合适的调度 S

15. o 的前置条件 = o 的前置条件 $\& S$

16. end if

17. end for

18. end if

19. end for

20. for each o_i do

21. if $\exists o_j \in o_i$ 的前置条件 then

22. o_j 的后置操作 = o_j 的后置操作 $\cup o_i$

23. end if

24. end for

4 仿真实验及分析

本文的仿真实验搭建在 CPU 为 Intel Xeon, 主频为 3.2 GHz, 内存 16 GB 的商用服务器上。软件定义航空集群机载网络的场景采用 Exata 5.1 网络仿真平台进行搭建。本节与现有集中式网络更新方法的典型代表 Dionysus 更新系统进行对比, 从而验证分析所提方法的性能。

4.1 仿真参数设置

仿真参数设置如表3所示。节点按照控制器计算并下发的流表规则进行数据包转发和处理。控制器在每个流量工程周期内根据当前网络拓扑重新计算新的路由转发, 执行网络更新。共进行 1 000 次完整的网络更新过程, 对每一次完整网络更新数据进行统计、处理和分析。

表3 仿真参数设置

仿真参数	数值
场景区域/(km $\times$ km)	1 000 $\times$ 1 000
节点数目/个	20~100
节点通信半径/km	200
节点移动速率/(km $\cdot$ h $^{-1}$)	200~1 200
业务流数目/条	40~200
业务流需求/Mbps	$N(1, 1)$
链路容量/Mbps	5

4.1.1 更新完成时间

图2为网络更新完成时间的比较。从仿真结果可以看出, 计算时间只在更新完成时间中占据了很小的比例, 更新完成时间主要表现在更新顺序协调所消耗的协调时间。由于分布式更新方法采用询问与通知2种消息直接在数据平面的交换机之间进行分布式交互, 避免了集中式更新中依赖控制器进行更新调度的“控制器-交换机-控制器”模式所带来的频繁交互产生的高延迟, 因此能够显著降低更新协调时间。与 Dionysus 更新系统相比, 本文提出的分布式更新方法在更新计算时间上表现近似, 但在更

新协调时间上降低了约 40%，从而显著降低了整个更新完成时间。

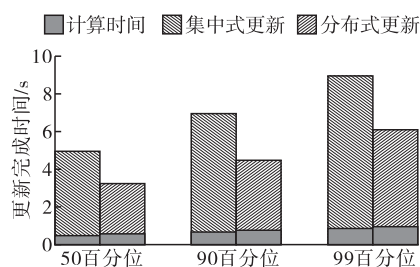


图2 更新完成时间比较

4.1.2 更新协调距离

图3为本文分布式更新方法在仿真实验中更新协调距离的累积密度函数。更新协调距离表示一个更新操作执行之后向被其依赖的更新操作所对应的节点发送通知消息所需要的跳数,其累积密度函数表示不大于该更新协调距离的操作在操作总数目中所占比例。从仿真结果可以看出,在分布式更新中约20%的更新协调距离为0,即相互依赖的更新操作在相同的交换机上执行;约36%的更新协调距离为1,即相互依赖的更新操作在相邻的交换机上执行。而在集中式更新中,需要控制器与数据平面之间反复交互,每个操作的更新协调距离都至少为2。分布式更新方法利用相邻交换机之间的直接交互,大大降低了消息交互所带来的更新时间,从侧面验证了分布式更新方法在降低更新时间上的优势。

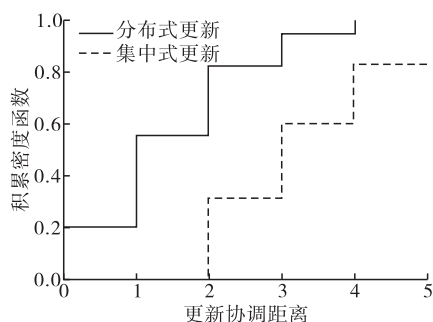


图3 更新协调距离比较

4.1.3 业务流更新时间

图4为业务流更新时间的累积密度函数对比。业务流更新时间表示单条业务流更新完成所需要的时间,其累积密度函数表示不大于该更新时间的业务流数目在业务流总数目中所占比例。从仿真结果中可以看出,在集中式更新方法中,最长的业务流更新时间约为800 ms,90%的业务流在400 ms内完成更新;而分布式更新方法中最长的业务流更新时间约为600 ms,90%的业务流在250 ms内完成更新。在同等累积密度函数下,分布式更新方法的业务更新时间比集中式更新方法更短;在同等业务流更新时间的限制下,分布式更新方法完成了更多的

业务流更新。可以看出分布式更新方法在提高业务流更新速度方面表现较好。

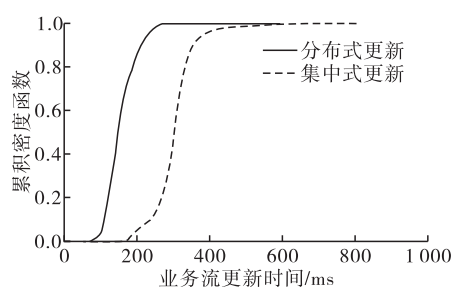


图4 业务流更新时间

5 结语

本文针对集中式网络更新方法更新时间长、过于依赖控制器的的问题,提出一种应用于软件定义航空集群机载网络的分布式更新方法。

与集中式更新方法相比,本文所提方法通过实现数据平面的分布式交互有效降低更新协调距离,减少了更新协调所需的交互次数,从而降低更新协调时间。仿真实验表明,本文所提方法与集中式更新方法相比,更新计算时间大致保持相同,更新协调时间降低了约40%,从而显著减少整个网络的更新完成时间。

参考文献

- [1] 梁晓龙,何吕龙,张佳强,柏鹏. 航空集群构型控制及其演化方法[J]. 中国科学:技术科学, 2019, 49(3): 277-287.
- [2] 梁一鑫,程光,郭晓军,等. 机载网络体系结构及其协议栈研究进展[J]. 软件学报, 2016, 27(1): 96-111.
- [3] 赵尚弘,陈柯帆,吕娜,等. 软件定义航空集群机载战术网络[J]. 通信学报, 2017, 38(8): 140-155.
- [4] MCKEOWN N, ANDERSON T, BALAKRISHNAN H, et al. OpenFlow: Enabling Innovation in Campus Networks[J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69-74.
- [5] MONSANTO C, REICH J, FOSTER N, et al. Composing Software Defined Networks[C]//10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13). CA: USENIX, 2013: 1-13.
- [6] KREUTZ D, RAMOS F M V, ESTEVES V P, et al. Software-Defined Networking: A Comprehensive Survey[J]. Proceedings of the IEEE, 2014, 103(1): 10-13.
- [7] FOERSTER K T, SCHMID S, VISSICCHIO S. Survey of Consistent Software-Defined Network Updates[J]. IEEE Communications Surveys & Tutorials, 2018, 21(2): 1435-1461.
- [8] KOMAJWAR S, KORKMAZ T. Challenges and Solutions to Consistent Data Plane Update in Software De-

- fined Networks[J]. Computer Communications, 2018, 130: 50-59.
- [9] REITBLATT M, FOSTER N, REXFORD J, et al. Abstractions for Network Update [J]. ACM SIGCOMM Computer Communication Review, 2012, 42(4): 323-334.
- [10] KATTA N P, REXFORD J, WALKER D. Incremental Consistent Updates[C]//Proceedings of The Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. NY: ACM, 2013: 49-54.
- [11] HONG C Y, KANDULA S, MAHAJAN R, et al. Achieving High Utilization with Software-Driven WAN [C]//ACM SIGCOMM Computer Communication Review. NY: ACM, 2013, 43(4): 15-26.
- [12] JIN X, LIU H H, GANDHI R, et al. Dynamic Scheduling of Network Updates [C]//ACM SIGCOMM Computer Communication Review. NY: ACM, 2014, 44(4): 539-550.
- [13] Li G, Qian Y, Zhao C, et al. Ddp: Distributed Network Updates in Sdn[C]//2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2018: 1468-1473.
- [14] Nguyen T D, Chiesa M, Canini M. Decentralized Consistent Updates in SDN[C]//Proceedings of the Symposium on SDN Research. NY: ACM, 2017: 21-33.
- [15] MAHAJAN R, WATTENHOFER R. On Consistent Updates in Software Defined Networks[C]//Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks. NY: ACM, 2013: 1-7.
- [16] MATTOS D M F, DUARTE O C M B, PUJOLLE G. Reverse Update: A Consistent Policy Update Scheme for Software-Defined Networking[J]. IEEE Communications Letters, 2016, 20(5): 886-889.
- [17] LUDWIG A, MARCINKOWSKI J, SCHMID S. Scheduling Loop-Free Network Updates: It's Good to Relax! [C]//Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing. NY: ACM, 2015: 13-22.
- [18] WANG W, HE W, SU J, et al. Cupid: Congestion-Free Consistent Data Plane Update In Software Defined Networks[C]//IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications. NJ: IEEE, 2016: 1-9.

(编辑:韩茜)

(上接第98页)

的方法和基于 ECF 的方法分别是所提方法的特例。通过仿真分析和验证了尺度因子对最终成像性能的影响。与基于 CF 的方法相比,所提方法能更有效地抑制伪影。通过选择合适的尺度因子,所提方法比基于 ECF 的方法更适用于多目标情况。在实际应用中,如何更好地自适应选择尺度因子需要进一步研究。

参考文献

- [1] 许强,金添,邱磊. 基于多特征结合的 MIMO 穿墙雷达“鬼影”抑制[J]. 现代电子技术, 2015, 38(19): 1-7.
- [2] FENG W, FRIEDT J M, NICE G, et al. 3-D Ground-Based Imaging Radar Based on C-Band Cross-MIMO Array and Tensor Compressive Sensing[J]. IEEE Geoscience and Remote Sensing Letters, 2019, 16(10): 1585-1589.
- [3] YANG J, JIN T, HUANG X, et al. Sparse MIMO Array Forward-Looking GPR Imaging Based on Compressed Sensing in Clutter Environment [J]. IEEE Transactions on Geoscience and Remote Sensing, 2013, 52(7): 4480-4494.
- [4] 王怀军,黄春琳,陆珉,等. MIMO 雷达反向投影成像算法[J]. 系统工程与电子技术, 2010, 32(8): 1567-1573.
- [5] FENG W, YI L, SATO M. Near Range Radar Imaging Based on Block Sparsity and Cross-Correlation Fusion Algorithm[J]. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2018, 11(6): 2079-2089.
- [6] ZHUGE X, YAROVVOY A G. Sparse Multiple-Input Multiple-Output Arrays for High-Resolution Near-Field Ultra-Wideband Imaging[J]. IET Microwaves, Antennas Propagation, 2011, 5(13): 1552-1562.
- [7] BURKHOLDER R J, BROWNE K E. Coherence Factor Enhancement of Through-Wall Radar Images[J]. IEEE Antennas and Wireless Propagation Letters, 2010(9): 842-845.
- [8] LU B, SUN X, ZHAO Y, et al. Phase Coherence Factor for Mitigation of Sidelobe Artifacts in Through-The-Wall Radar Imaging[J]. Journal of Electromagnetic Waves and Applications, 2013, 27(6): 716-725.
- [9] LIU J, JIA Y, KONG L, et al. Sign-Coherence-Factor-Based Suppression for Grating Lobes in Through-Wall Radar Imaging[J]. IEEE Geoscience and Remote Sensing Letters, 2016, 13(11): 1681-1685.
- [10] NILSEN C I C, HOLM S. Wiener Beamforming and the Coherence Factor in Ultrasound Imaging[J]. IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, 2010, 57(6): 1329-1346.
- [11] HU J, ZHU G, JIN T, et al. Grating Lobe Mitigation Based on Extended Coherence Factor in Sparse MIMO UWB Array[C]//2014 12th International Conference on Signal Processing (ICSP). IEEE, 2014: 2098-2101.
- [12] FORTUNY-GUASCH J. A Fast and Accurate Far-Field Pseudopolar Format Radar Imaging Algorithm [J]. IEEE Transactions on Geoscience and Remote Sensing, 2009, 47(4): 1187-1196.

(编辑:陈斐)