

面向大规模卫星网络协议设计和性能评估的 实时仿真系统设计及验证

曹琦轩, 史 琰, 周连伟, 丁新旺
(西安电子科技大学通信工程学院, 陕西 西安 710071)

摘要: 针对当前仿真工具存在的问题, 研究并开发实时网络仿真测试系统SiT4NET。首先, SiT4NET利用容器技术在保证节点独立性的基础上解决了有限仿真资源实现大规模网络节点部署问题, 提升了仿真系统的资源利用率; 其次, 提出基于数据单元的节点间分组传输和系统级数据单元申请/回收机制, 为网络的实时仿真提供核心支撑; 再次, 通过标准高层协议接口和内部分组格式及可重构的分组处理机制, 降低了协议的仿真实现与目标平台之间转换的复杂度; 最后, 通过SiT4NET和OPNET平台搭建相同的卫星网络环境, 设置多组仿真场景, 对仿真结果进行比较和分析, 说明SiT4NET可以满足大规模卫星网络性能仿真需求。

关键词: 容器技术; 实时仿真; 虚拟分组传输; 分级数据单元管理

中图分类号: TP927.2

文献标志码: A

doi: 10.11959/j.issn.2096-8930.2023004

Design and Verification of Real-Time Simulation System for Protocol Design and Performance Evaluation of Large-Scale Satellite Network

CAO Qixuan, SHI Yan, ZHOU Lianwei, DING Xinwang
School of Telecommunication Engineering, Xidian University, Xi'an 710071, China

Abstract: Aiming at the problems above, a real-time network simulation test system named SiT4NET was studied and developed. Firstly, SiT4NET used container technique to solved the problem of large-scale network node deployment with limited resources on the basis of ensuring nodes' independence, thus improved the resource utilization of the simulation system. Secondly, the packet transmission between nodes based on data units and the mechanism for requesting/retrieving system-level data units were innovatively proposed to provided a strong foundation for real-time network simulation. And thirdly, through the standard high-level protocol interface, internal packet format and reconfigurable mechanism of packet processing, SiT4NET reduced the complexity of conversion between the simulation implementation of the protocol and the target platform. Finally, by set up the same satellite network environment on SiT4NET and OPNET platforms, set up multiple groups of simulation scenarios, compared and analyzed the results, it showed that this method could met the performance requirements of large-scale satellite network.

Keywords: container technology, real-time simulation, virtual packet transmission, graded data unit management

0 引言

大规模低轨卫星网络具有传播时延低、全球覆盖、信号质量好等优势, 会成为未来移动通信网络的重要组成部分。低轨卫星网络存在规模庞大、网络拓扑复杂等特点, 相关算法和路由协议测试、验证困难, 需要专门

研制大规模卫星网络仿真与评估系统。目前国内外主要从以下4个方面对卫星网络仿真进行相关研究: 面向卫星网络的数学理论模型、基于实物测试床的卫星、基于离散事件模拟的卫星网络数字仿真、基于虚拟化技术的卫星网络仿真。

收稿日期: 2022-11-15; **修回日期:** 2023-02-10

基金项目: 国家重点研发计划资助项目(No.2020YFB1806100)

Foundation Item: National Key Research&Development Program of China (No.2020YFB1806100)

(1) 面向卫星网络的数学理论模型

参考文献[1]根据星间链路建链过程中卫星相对运动特性实时地计算出波束指向所需的方位角和俯仰角信息。参考文献[2]深入研究了天线指向特性,同时对星间链路以及轨道构型进行建模和分析。参考文献[3]根据研究的GNII传输时路径选择的约束要素与约束关系展开建模分析。参考文献[4]研究了LEO网络星间链路模型,探讨了星间链路几何参数的动态特性变化规律和对整体性能以及设计的影响。以上阐述的算法和仿真模型为进行高性能卫星网络仿真提供了重要的理论基础,但是理论分析需要与实际仿真测试结果进行对比进而验证理论的正确性。

(2) 基于实物测试床的卫星

参考文献[5]提出了一个时延容忍网络(DTN)应用和协议的网络测试平台,该平台基于通用的无线网络仿真测试平台QOMB开发,采用和目标环境强一致的实物测试环境。该平台依赖于硬件资源,在大规模场景下,存在着成本高、灵活性低和可维护性差的问题。此外,实物测试难以实现实验环境的可重复性以及可控性,仿真测试与性能评估通常需要在特定环境下进行大量的实验。因此,相比于软件仿真,实物测试不利于对网络协议和上层应用从统计学原理的角度进行客观测试与分析。

(3) 基于离散事件模拟的卫星网络数字仿真

采用离散事件机制的仿真软件测试网络协议是目前主流的仿真方法。参考文献[6]提出了一种基于改进的OPNET-HLA的大规模网络仿真方法,该方法利用OPNET网络仿真软件进行分布式仿真,但是又进一步增加了网络体系结构设计的复杂度。参考文献[7]使用网络仿真工具NS3对基于SDN的卫星LTE融合网络模型进行仿真搭建,并对不同仿真场景下的网络性能进行测试和分析。参考文献[8]基于网络模拟软件NS2模拟卫星链路并搭建GSM链路,在NS2中实现了Abis接口,并评估了在高时延以及损耗卫星信道上的性能。基于离散事件模拟的仿真软件进行测试,存在各层网络协议以及上层应用无法直接移植的问题,需要在仿真软件内重新实现后方可使用,一定程度上增加了仿真实现的复杂度。此外,上述卫星网络数字仿真软件的仿真推进依赖于对离散事件的调度,不具备实时性,在大规模仿真场景下无法有效加载真实业务流量,仿真的拟真度较差。

(4) 基于虚拟化技术的卫星网络仿真

参考文献[9]利用虚拟化技术搭建卫星互联网仿真平台,在SGIN-HPET方法的基础上,进一步提出卫星互联网动态联动仿真技术。参考文献[10]提出了一种基于容器技术的虚拟网络数据通信交互方法,为Docker节点虚拟出网卡并且配置IP,实现Docker节点间的连通性。

参考文献[11]以OpenStack为基础设计了DTN仿真平台EmuStack,使用多台物理机模拟数百节点的大规模网络拓扑,不会像基于离散事件的网络模拟器一样等待事件,而是作为分布式实时仿真平台。基于虚拟化技术的卫星网络仿真技术综合了实物测试平台和网络模拟器的仿真优势,已成为当前卫星网络仿真的主流技术^[12],不仅能够运行真实业务流量,而且仿真结果具有较高逼真度和可扩展性。但是,现有基于虚拟化的网络仿真平台采用虚拟网络技术实现数据通信,存在大量用户态和内核态的数据拷贝,不仅会占用过高的计算机资源,严重影响平台的有效仿真规模,而且会降低数据传输效率,无法有效地模拟大规模场景下的高吞吐量业务。

1 大规模卫星网络实时仿真平台关键技术

1.1 基于容器的仿真网络节点大规模部署技术

虚拟化是一种资源管理技术,可以在一台计算机上为大规模卫星节点提供相互隔离且资源受控的运行环境,在节约成本的同时提高计算机的资源利用率。目前用于为应用程序提供独立且可移植的计算环境的虚拟化技术主要有虚拟机和容器。由于创建容器的系统开销远小于虚拟机,因此SiT4NET平台使用主流的容器管理软件Docker就可以在物理机上创建更多的虚拟化环境,从而实现更大规模的仿真卫星节点的部署。此外,考虑到仿真卫星节点部署规模受限于物理机的物理资源,本平台采用集群架构,将容器节点部署到多台物理机内,进一步提高有效仿真规模。

1.2 实时仿真的关键技术

实时仿真主要体现在仿真系统的时间推进与实际时间保持一致,要求仿真中事件的处理速率与该事件在现实中的处理速率保持一致。大规模卫星网络实时仿真的主要任务是模拟真实数据业务的流动,在保证仿真的实时性和有效性的前提下,需要高性能的节点间数据传输速率来提高平台的仿真规模和性能。为了提高仿真卫星节点间数据的传输速率,本平台提出了基于共享内存的虚拟数据单元传输技术,可以避免传统socket方案导致的进程切换与数据拷贝开销。此外,本平台还提出了支持多核并行处理的分级数据单元管理机制,进一步提高分组的传输速率。

1.2.1 虚拟数据单元传输技术

利用共享内存实现分组传输,首先需要发送节点将待传输的分组保存到共享内存上,并通知接收节点分组在共享内存中的位置和大小。在共享内存上划分空间保存分组的方式主要有固定分区和动态分区。相比于动态分区,固定分区在共享内存中划分出足够容纳分组的固定大小的单元,不需要根据分组大小调整分区,不存在外部碎片,

分配和回收更简单高效。因此,本平台在共享内存中采用固定分区方式的虚拟数据单元来保存待传输的分组。

虚拟数据单元是指一种在共享内存中创建的结构体,用于保存仿真参数信息、自定义数据帧头部信息以及真实业务数据。图1展示了节点通过共享内存上的虚拟数据单元实现传输分组的过程。节点将需要传输的分组保存到虚拟数据单元中,并在发送队列中传递指向保存分组的虚拟数据单元地址。

1.2.2 支持多核并行处理的分级数据单元管理机制

虚拟数据单元是在共享内存中划分的有限的资源,需要一种管理机制实现对使用完的数据单元进行回收以循环利用,从而保证仿真可以长期有效运行。此外,在大规模节点并行传输分组时,虚拟数据单元管理机制需要充分利用多核并行处理机制,保证大规模高性能仿真的实时性和有效性。

第一种设计方案考虑从机器层面创建一个保存所有空闲数据单元地址的队列来管理所有虚拟数据单元。该方案

中各容器节点并行发送分组时会向数据单元管理队列频繁申请数据单元,需要采取并发安全机制避免多线程数据单元出队冲突,会使得各节点申请与回收虚拟数据单元的操作串行化,无法充分发挥物理机多核的能力。第二种设计方案考虑从容器层面管理数据单元。该方案中每个容器节点自己创建数据单元管理队列,并在传输分组时直接从自己的数据单元管理队列获取与回收数据单元,无须进行并发安全控制。但是容器节点中保存的数据单元数量是有限的,无法保障系统长期高速仿真。综上所述,单级数据单元管理机制存在缺陷,如果仅从整台机器角度进行管理,会导致各节点发送分组串行化;如果仅从容器角度对数据单元进行管理,会造成数据单元资源失衡现象。因此本文将上述两个方案结合,设计了分级数据单元管理机制。

分级数据单元管理机制不仅在宿主机上创建机器级数据单元管理队列,还在每一个容器中创建容器级数据单元管理队列。图2展示了容器级数据单元管理队列与机

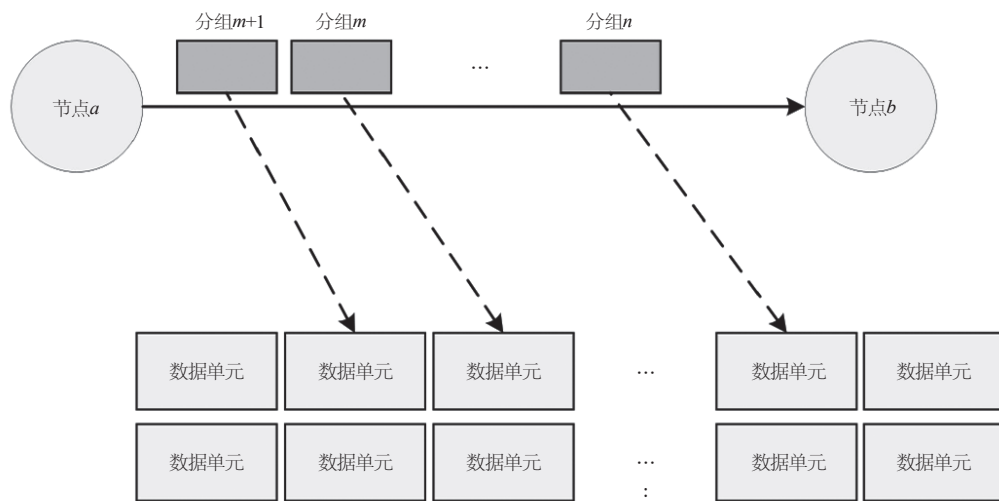


图1 节点通过共享内存上的虚拟数据单元实现传输分组的过程

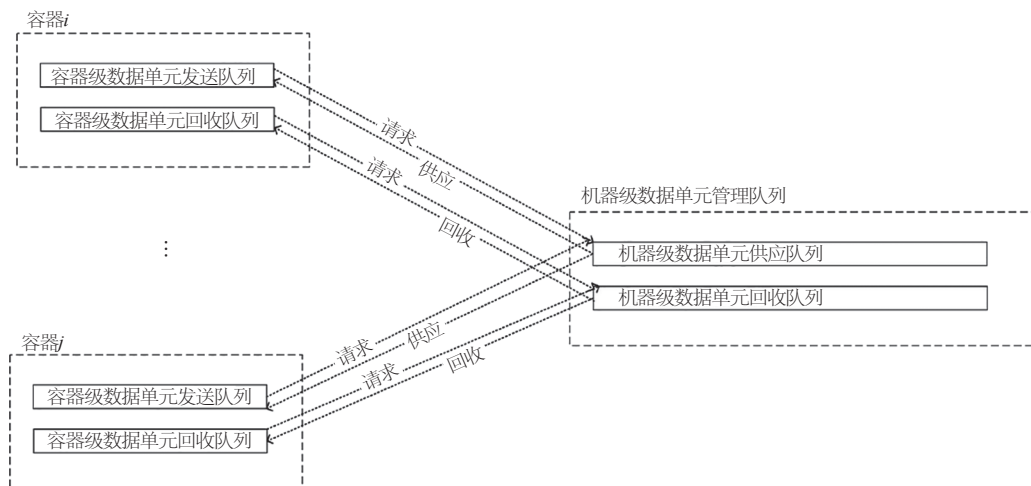


图2 容器级数据单元管理队列与机器级数据单元管理队列的关系

器级数据单元管理队列的关系。

虚拟数据单元以组的形式划分,每一组单元的地址保存在单元块中。机器级数据单元管理队列中保存了多个单元块的块号,并在容器级数据单元管理队列需要补充数据单元时提供整个单元块。容器节点在发送分组时从容器级数据单元管理队列请求虚拟数据单元,并在发现队列为空时,向机器级数据单元管理队列重新申请一个单元块。同样,节点在接收分组时,需要将虚拟数据单元回收进单元块,并在单元块存满时,将单元块号回收进机器级数据单元管理队列中。分级数据单元管理机制通过机器级数据单元管理队列为节点补充虚拟数据单元,从而解决节点间由于业务流传输导致的数据单元资源失衡问题。下面将建立排队模型,分析分级数据单元管理机制的数据单元出队效率。

设节点*i*在单位时间内产生的业务量为 λ_i ,管理队列单位时间内供应数据单元数为 μ ,整网内节点总个数为 k 。图3为单级管理队列的排队模型示意。

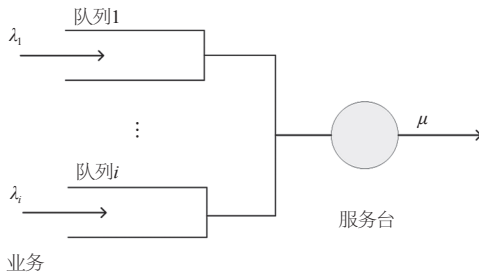


图3 单级管理队列排队模型示意

对于从机器层面管理虚拟数据单元的单级管理队列,如果所有节点都在产生业务,整网单位时间内产生

的业务量设为 λ_k ,可得式(1)

$$\lambda_k = \sum_{i=1}^k \lambda_i \quad (1)$$

设单个业务平均等待时间为 W_{qs} ,如式(2)所示

$$W_{qs} = \frac{\lambda_k}{\mu(\mu - \lambda_k)} = \frac{1}{\mu} \cdot \left(\frac{1}{\frac{\mu}{\lambda_k} - 1} \right) \quad (2)$$

对于本文提出的分级数据单元管理机制,在每个节点内都有容器级数据单元管理队列,不同容器内服务相互独立,如图4所示。

设 W'_{qi} 为容器*i*内单业务平均等待时间,得

$$W'_{qi} = \frac{\lambda_i}{\mu(\mu - \lambda_i)} = \frac{1}{\mu} \cdot \left(\frac{1}{\frac{\mu}{\lambda_i} - 1} \right) \quad (3)$$

设容器级数据单元管理队列大小为 N ,当该队列数据单元都用完时会去机器级数据单元管理队列申请补充。容器*i*单位时间内申请补充数据单元的次数为 λ_{si} ,得

$$\lambda_{si} = \frac{\lambda_i}{N} \quad (4)$$

整网单位时间内申请补充数据单元次数为 λ_s ,得

$$\lambda_s = \sum_{i=1}^k \frac{\lambda_i}{N} = \frac{\lambda_k}{N} \quad (5)$$

容器*i*等待数据单元补充的平均等待时间为 W''_{qi} ,如

$$W''_{qi} = \frac{\lambda_s}{\mu(\mu - \lambda_s)} = \frac{1}{\mu} \cdot \left(\frac{1}{\frac{\mu N}{\lambda_k} - 1} \right) \quad (6)$$

因此,对于两级管理队列,单个业务总的平均等待

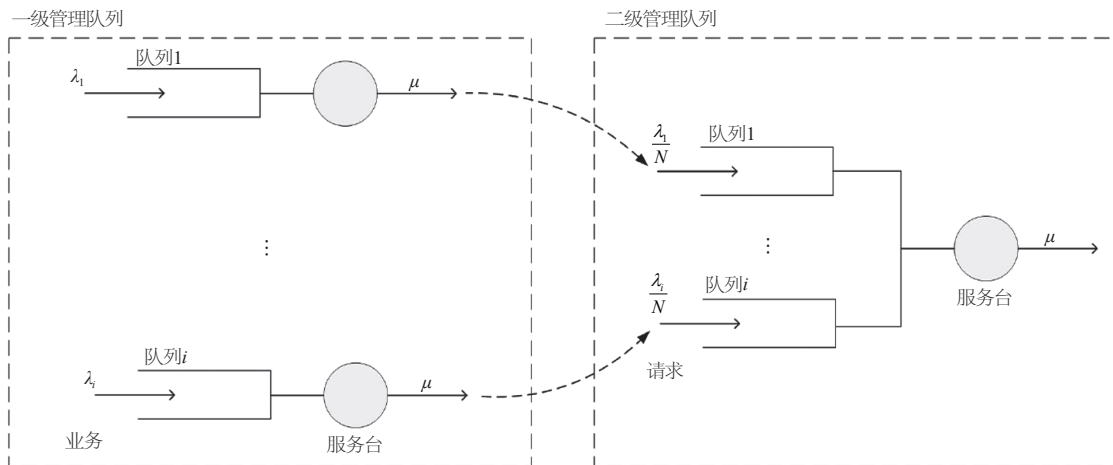


图4 双级管理队列排队模型示意

时间 W_{q_i} 如下

$$W_{q_i} = W_{q_i}' + W_{q_i}'' = \frac{1}{\mu} \cdot \left(\frac{1}{\frac{\mu}{\lambda_i} - 1} + \frac{1}{\frac{\mu N}{\lambda_k} - 1} \right) \quad (7)$$

当容器级数据单元管理队列较长，也就是 N 较大时，机器级数据单元管理队列补充数据单元所耗时间可以忽略，如下

$$W_{q_i} = W_{q_i}' + W_{q_i}'' \approx W_{q_i}' = \frac{1}{\mu} \cdot \left(\frac{1}{\frac{\mu}{\lambda_i} - 1} \right) \quad (8)$$

比较式(2)与式(8)可得，相比于只从机器层面管理虚拟数据单元的单机管理队列，采用分级数据单元管理机制使得每个节点只有在容器级管理队列为空时才会以块为单位向机器级管理队列请求数据单元，可以显著降低机器级管理队列补充数据单元的时间开销，提高分组传输的整体速率，进一步保证大规模高性能仿真的实时性和有效性。

1.3 仿真目标与仿真实现的快速转换技术

作为一个网络仿真平台，本平台的任务是将目标平台上运行的实例转换为可以在本平台上运行的仿真模型，从而在仿真场景中实现对目标实例中运行的协议、功能等的仿真验证。目前已有的一些仿真软件，存在可移植性差、仿真实现复杂、用户定制不灵活等缺点。例如，在OPNET中，用户需要将整个目标实例根据OPNET的3层建模机制进行实现，增加了额外的仿真开发周期。此外，用户在OPNET中定义的网络节点模型的运行依赖于其自身的离散事件仿真引擎，只有将仿真实现整体重写才能在其他平台上运行。本平台针对现有仿真软件存在仿真实现与目标平台运行的实例之间快速转换困难的问题，通过在仿真平台中使用标准的POSIX多线程进行实时仿真，降低了仿真实现移植到目标平台的复杂度。此外，本平台实现了通用的网络模型，并设计标准高层协议接口实现上层协议应用与仿真平台的解耦，降低了用户对协议的仿真开发难度。最后，本平台提出了开放的分组格式和处理机制，在通用的网络节点模型中为用户提供了灵活定制网络功能的方法。

1.3.1 标准的POSIX可移植操作系统接口

可移植操作系统接口(POSIX)是由IEEE发布的一套操作系统对外接口标准，意在获得源代码级别的软件可移植性^[13]。本平台通过POSIX进行开发，利用Linux提供的POSIX多线程机制推动网络节点的运行，使得平台代码可以在兼容POSIX标准的操作系统上直接重新编译

运行，无须对平台代码整体重写，从而使得本平台的仿真实现具有较好的移植性。

1.3.2 标准的高层协议接口

网络节点中主要包含数据链路层模型、网络层模型和高层协议。其中，高层协议部分主要由用户开发，根据不同的需求，其实现存在多样性。为了降低用户的开发成本，本平台实现了如图5所示通用的网络节点模型。在网络节点模型中，定义了网络层与数据链路层中的基本功能，并定义了与高层协议应用之间的标准接口，从而实现高层协议应用与仿真进程之间的解耦，使用户专注于高层协议的仿真开发。

用户在实现不同高层协议应用时，无须对网络层和数据链路层进行整体重写，直接调用高层协议接口就可以向节点模型中的网络层和数据链路层下发数据，使得本平台的网络节点模型具有较强的通用性。例如，上层的路由协议应用运行时，首先会调用初始化函数向节点进程注册自身信息，接着获取向节点进程传输数据的缓冲区，最后通过调用标准的数据发送和接收函数，实现与网络层之间的分组传输。

1.3.3 开放的分组格式和分组处理机制

由于在覆盖范围、移动模型、业务特性、链路特性等方面的差异，不同网络节点的实现可能存在较大区别，主要体现在网络节点模型中分组处理的差别。本平台提供了如图6所示的开放分组格式和分组处理机制。

首先，本平台在内部定义了统一格式的分组。当网络节点从链路上接收到用户自定义格式的分组后，会通过用户实现的转换函数将用户自定义格式的分组或帧转换为统一的分组格式，后续在网络层与数据链路层中处理统一格式的分组，从而保证节点模型在处理不同网络分组时的通用性。此外，在节点模型中，与分组处理相关的功能采用可重构函数实现。当用户需要根据网络需求定制网络层或数据链路层中某些功能时，只需要对该功能所对应的处理函数进行重构，无须重写整个节点模型，从而在网络节点模型实现的通用性和面向网络功能修改的灵活性之间达到了较好的平衡。

2 大规模实时仿真平台的搭建与性能评估

2.1 大规模卫星网络实时仿真平台的搭建

SiT4NET平台部署在由高性能交换机和8台高性能物理机组成的计算集群中。本文利用Docker软件在8台高性能物理机上创建大量的容器节点来运行模拟卫星，并在每一台物理机上运行控制仿真运行的机器级仿真引擎。本文采用上文提出的分级数据单元传输技术实现了同一台物理机内容器间高效的数据传输引擎。针对

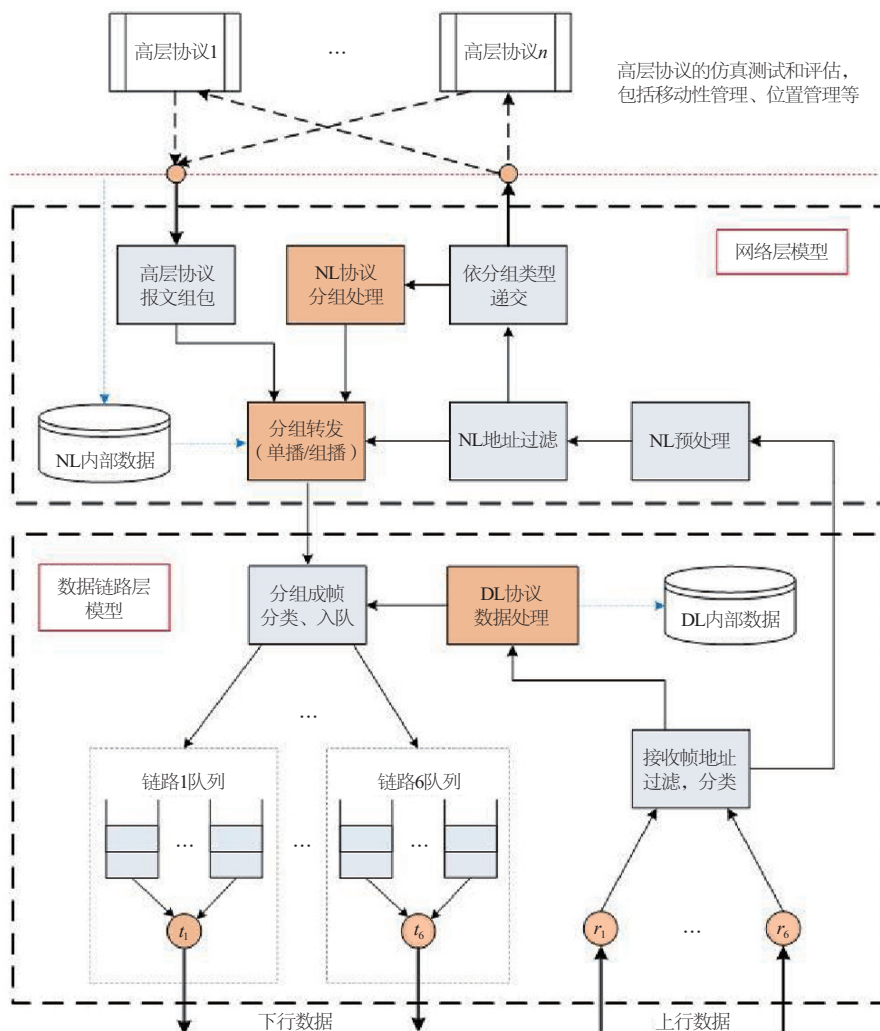


图5 通用的网络节点模型

跨物理机的容器节点，本文使用远程直接存取器访问（Remote Direct Memory Access, RDMA）技术提高数据在物理机之间的传输效率，保证跨物理机的容器节点之间通信的实时性。在部署SiT4NET平台的高性能物理机上，采用100 Gbit/s速率的RDMA网卡与交换机联通，利用RDMA的零拷贝、内核旁路等特性，降低跨物理机链路的模拟对CPU资源的占用，显著提高跨物理机数据传输速率。

SiT4NET平台主要为大规模卫星网络仿真服务，支持实时仿真、虚实结合仿真，为用户提供可灵活定制的通用节点模型，且仿真实现具有较强的可移植性。本平台可以有效部署的卫星星座规模达到20 000颗，单链路有效仿真速率为500 000 pps（Packet Per Second），全网仿真总吞吐量在8台物理机组成的集群中可以达到640 Gbit/s。

2.2 平台性能评估

为了评估SiT4NET平台的仿真性能，本文分别测试了SiT4NET平台大规模卫星网络节点部署能力、仿真时长和网络吞吐量，对平台的性能进行了评估。

2.2.1 仿真平台大规模卫星网络节点部署能力

本文在8台物理机上共创建了20 000个容器，并在容器内运行卫星网络仿真进程snpsia，从而实现20 000个卫星节点的部署。单物理机内网络节点进程部署的资源占用结果统计如图7所示。

由于仿真资源主要关注CPU占有率和内存占用，本文在仿真节点部署前后查询了物理机的CPU使用率和内存占用情况。分析图7，在仿真节点部署前，物理机的CPU有98.9%的空闲时间占用百分比，内存占用为38 494 MB；在仿真节点部署后CPU空闲时间百分比为93.9%，内存占用

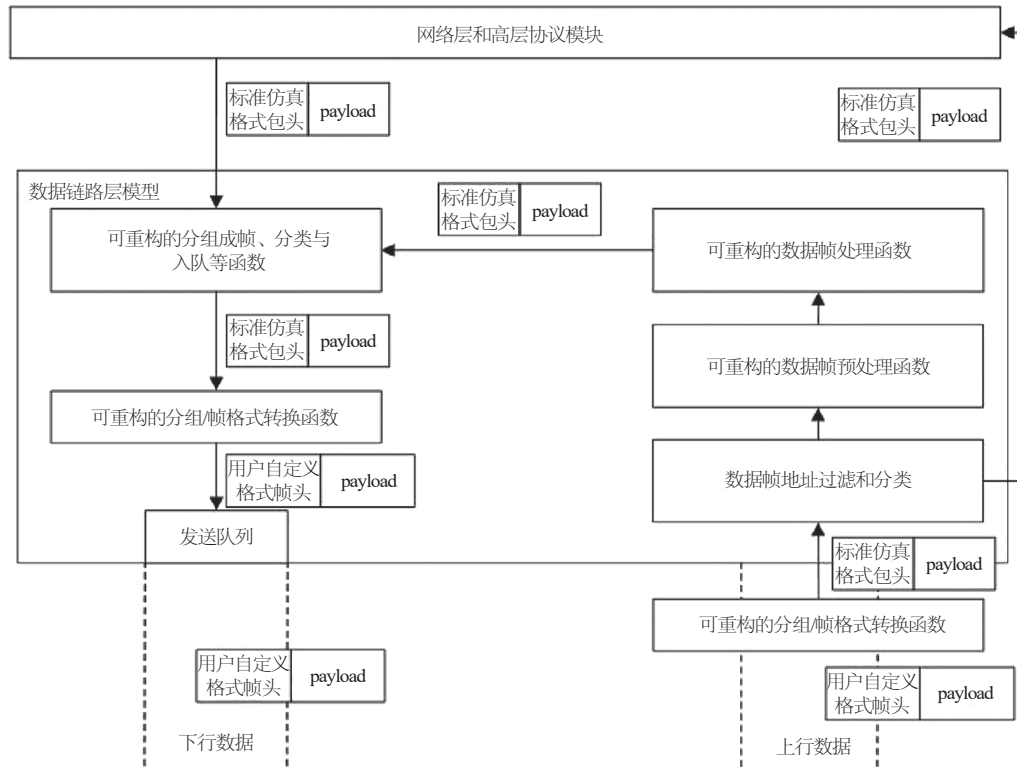


图6 开发的分组格式与分组处理机制

```
[root@worker1 command1008wxy]# ps -ef | grep ./snpsia | grep -v grep | grep 'docker exec' | wc -l
0
[root@worker1 command1008wxy]# top|grep Cpu
%Cpu(s):  0.4 us,  0.7 sy,  0.0 ni, 98.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st

[root@worker1 command1008wxy]# free -m
              total        used         free      shared  buff/cache   available
Mem:           257433         38494        201754          443        17185        217631
Swap:              0              0              0
```

(a) 大规模网络节点部署前物理机资源占用

```
[root@worker1 command1008wxy]# ps -ef | grep ./snpsia | grep -v grep | grep 'docker exec' | wc -l
2500
[root@worker1 command1008wxy]# top|grep Cpu
%Cpu(s):  3.2 us,  2.9 sy,  0.0 ni, 93.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st

[root@worker1 command1008wxy]# free -m
              total        used         free      shared  buff/cache   available
Mem:           257433        176944         62149         1569        18339        78056
Swap:              0              0              0
```

(b) 大规模网络节点部署后物理机资源占用

图7 单物理机内网络节点进程部署的资源占用结果统计

为176 944 MB。对比大规模网络节点部署前后的CPU占有率只相差5%，说明本文通过容器部署大规模节点不会占用过高的处理器资源。仿真部署过程中会创建大量的节点进程，并为进程分配内存空间。仿真部署前后，物理机的内存占用增加了128 GB，平均为每一个节点分配

了52 MB的内存空间，其内存占用较低，进一步说明本文的仿真平台可以通过8台物理机有效完成20 000个节点的仿真部署。

2.2.2 仿真时长测试

OPNET采用离散事件驱动机制推进仿真，随着事件

量的增多, 仿真消耗的时间也随之增多。本实验设置仿真时长为100 s, 在高业务量场景下, 本平台与OPNET平台仿真耗费时间对比如图8所示。

分析图8, 本平台实际仿真时间和设置的仿真时长基本一致, 而OPNET平台所用时间明显要大于仿真时长, 可以说明本仿真平台能够有效进行实时仿真测试。

2.2.3 网络吞吐量性能测试

本文在单台物理机内测试了SiT4NET平台可以模拟的星间链路吞吐量和网络吞吐量性能。为了满足多种测试场景的需求, 探究发包速率对星间链路吞吐量的影响, 本文在单对收发卫星节点上设置了多种数据包发包速率进行仿真测试, 测试结果如图9(a)所示。此外, 在实际场景下往往会存在多条业务流, 为了研究仿真中多个节点同时发送分组对本平台星间链路吞吐量的影响, 本文增设了收发卫星节点对数这一变量, 同时不设置发包间隔, 保证发包速率可以达到最大, 以达到测试平台可以承载的网络吞吐量极限性能的目的, 测试结果如图9(b)所示。

分析图9(a), 随着发包速率不断增加, 卫星网络吞吐量也在相应地提升。在发包速率较小时, 平台吞吐

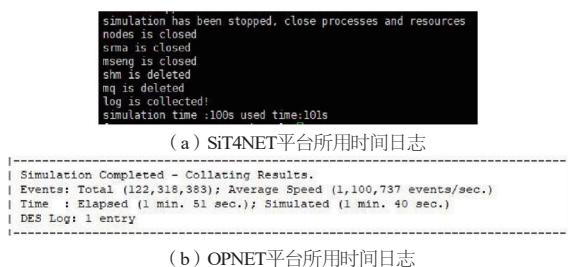
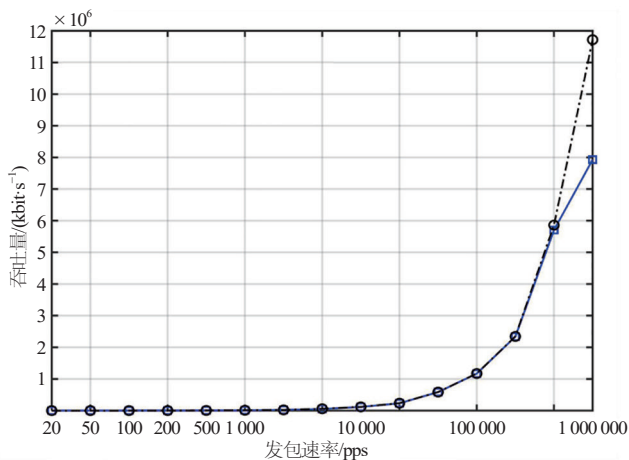


图8 SiT4NET平台与OPNET平台仿真消耗时间对比



(a) 发包速率测试结果

量测试数据曲线与理论测试曲线几乎重合; 而随着发包速率的逐渐增大, 平台吞吐量仿真值与理论值存在些许偏差, 特别是当发包速率达到1 000 000 pps时, 吞吐量仿真值明显比理论值要小, 说明当前情况下已达到平台性能极限。即当业务量较少时, 卫星节点可以及时地处理接收到的数据; 但是当业务量较高时, 卫星节点处理能力有限, 导致一些数据无法及时处理。

分析图9(b), 随着收发卫星节点对数的增多, 卫星网络整网吞吐量也在不断增加, 但是其并不与一对收发卫星节点所得结果呈倍数关系, 并且随着收发卫星节点对数的增加, 吞吐量的增加逐渐变少。实时仿真在运行过程中很大程度上会受到物理机硬件设备以及计算性能的影响, 在流量过高, 超出了计算机性能瓶颈时无法达到预期结果。通过测试, 在收发卫星节点对数为22对时, 整网吞吐量达到80 Gbit/s, 可以满足对卫星网络吞吐量的需求。

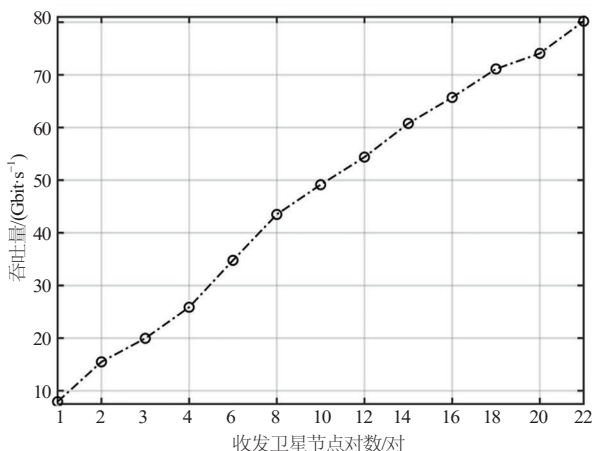
2.3 仿真实验移植到目标硬件平台的验证

为了验证平台仿真实验的移植性, 将在仿真系统中实现的路由协议移植到了星载路由交换机的VxWorks操作系统中运行, 并与仿真平台进行联合仿真。

仿真平台与星载交换机通过网线互联, 并运行相同的路由协议。如图10所示, 仿真平台发送的hello包可以被星载交换机正常接收, 表明从仿真平台上移植的路由协议可以在星载交换机上正常运行, 证明在本平台上进行仿真开发具有较好的移植性。

3 仿真分析

前文分析了本平台针对大规模卫星网络仿真使用的



(b) 多源多目的场景下平台吞吐量测试数据

图9 单对收发卫星节点不同发包速率下的吞吐量仿真测试结果

技术,提出了优化设计方案,并搭建仿真平台,测试验证了本平台性能。下面将在SiT4NET平台与OPNET平台上搭建相同的测试场景,通过对比仿真结果验证本平台仿真的正确性。

3.1 仿真场景设计

本文设计的卫星星座模型存在12个轨道,每个轨道共有12颗卫星。首先本文使用STK卫星工具包获取卫星星座数据,生成卫星网络中每分钟整网卫星的位置和状态等信息,接着对数据进行处理,生成每分钟全网路由表信息。在OPNET仿真软件中,卫星节点可以直接读取指定路径下的路由表文件;在SiT4NET仿真平台中,本文将全网路由表信息放到各容器挂载的共享内存映射文件中,容器中的仿真卫星节点会将全网路由表中自己的路由信息记录到本地。

3.2 路由协议介绍

无论是OPNET还是大规模卫星网络仿真与评估平台都是验证新的协议和算法,因此本文将同一个路由协议放到两个平台上运行,并对结果进行分析。在此仿真测试场景中,本文使用快照路由协议^[14]作为卫星节点间的路由协议。快照路由协议使用一系列的静态网络拓扑来描述动态变化的卫星网络拓扑。快照路由协议需要获取全网的卫星信息,因此该操作主要放在地面控制中心进行处理。地面控制中心处理全网卫星信息,周期性地生成全网路由表,计算出路由更新包扩散至全网的路径,并选取一个卫星节点作为源节点上注更新信息。地面控制中心生成全网路由表采用的是Floyd算法。Floyd算法采用动态规划的思想来计算加权图中任意两点之间的最短路径。

3.3 仿真测试结果比较与验证

为了验证仿真平台的有效性,本文在同一仿真场景

下测试了本平台与OPNET平台,并分别从时延和吞吐量两方面分析仿真结果。

(1) 时延

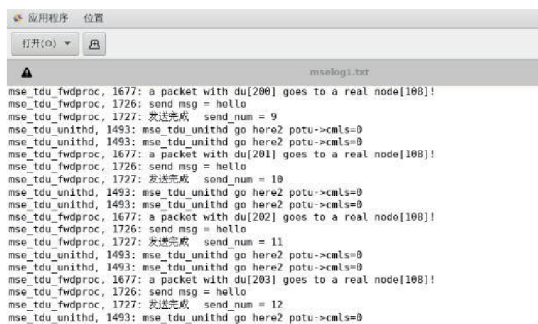
在两个平台中,选取相同的源卫星节点和目的卫星节点,从源节点向目的节点发送数据包,并对结果进行对比分析。时延仿真环境主要配置信息见表1。

表1 时延仿真环境主要配置信息

参数	值
卫星总数	144颗
发射机发包速率	1 Gbit/s
数据包大小	1 500字节
发包间隔	10 ms
仿真时间	100 s

为了让路由跳数更多,中间经过更多的卫星节点来进行时延仿真测试,本文选取51号和90号卫星分别作为源节点和目的节点。数据包一共需要经过7跳才会从源节点到达目的节点。51号卫星和90号卫星之间不存在直连链路,需要查找根据路由协议生成的路由表信息,通过7跳转发将数据包从51号卫星传输到90号卫星。经过在OPNET平台以及本平台仿真测试,数据包均会经过相同的转发路径到达目的卫星节点。根据上述仿真环境,本文对数据包传输过程的时延进行模拟,并分别在两个平台上进行100次链路时延仿真测试,其测试结果见表2。

分析表2,OPNET平台多次仿真结果完全一致,这是因为离散时间仿真严格按事件序列调度事件,而事件序列依赖于仿真系统提供的随机数函数和种子。在随机数种子相同的情况下,事件发生的时间和顺序都相同,且仿真时钟只按照事件序列中的时刻推进,不会受到物理机性能等外部条件影响,因此多次仿真结果具有强一致性。SiT4NET平台的实时仿真机制必然存在物理机的处理时延,单跳误差为27 μ s,该误差在可接受范围内。



(a) 仿真平台发送hello包打印



(b) 星载交换机接收hello包打印

图10 仿真平台向星载交换机发送hello包

表2 单对收发卫星节点时延仿真测试结果

仿真次数/次	OPNET平台测试平均时延/ms	SiT4NET平台测试平均时延/ms
20	62.701	62.882
40	62.701	62.881
60	62.701	62.881
80	62.701	62.884
100	62.701	62.883

表3 吞吐量仿真测试结果

仿真次数/次	OPNET测试平均吞吐量/(Mbit·s ⁻¹)	SiT4NET测试平均吞吐量/(Mbit·s ⁻¹)
20	1.144	1.129
40	1.144	1.128
60	1.144	1.129
80	1.144	1.129
100	1.144	1.129

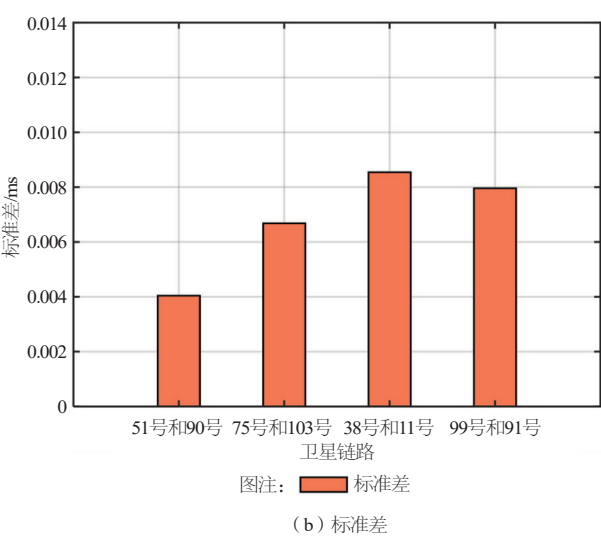
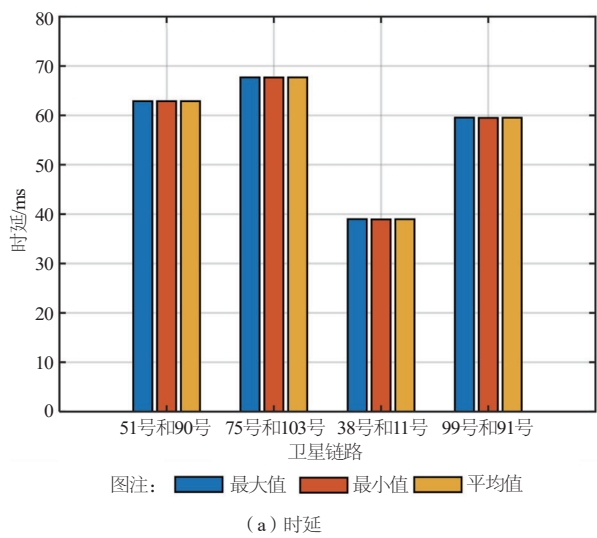


图11 单对收发卫星节点链路时延分析

除了51号和90号卫星节点，为了验证SiT4NET平台多次仿真结果的稳定性，本次实验又分别选取75号和103号卫星节点、1号和38号卫星节点以及91号和99号卫星节点作为仿真节点，得到图11。

分析图11可知，本仿真评估平台在同一仿真环境下多次实验所得结果基本相同，同时标准差结果也表明测试数据的离散程度不高，说明本平台的仿真测试稳定性强，可以进行重复实验。

(2) 吞吐量

与时延仿真相同，本文在OPNET平台以及本仿真评估平台上搭建相同的仿真场景，其中仿真时长设置为30 min。表3展示了在上述仿真环境下吞吐量仿真测试结果。

在实时仿真中，模拟数据包发送速率的精确度依赖于操作系统控制发送数据包的线程休眠时间的精度，因此实时仿真吞吐量必然会与理想情况有一定差距。分析表3中的结果，本平台测试的平均吞吐量与OPNET平台相仿，说明本平台测得的结果准确性高，验证了本平台设计的合理性与正确性。同时通过观察结果，在进行多次实验的情况下本平台测得的数据波动依旧很小（基本一致），同样说明本平台稳定性较高，可以进行反复测试实验。

参考文献：

[1] 滕云, 万里, 王跃科,等. 星间链路建链指向算法研究与性能验证[J]. 仪器仪表学报, 2014, 35(S2): 96-100.
TENG Y, WANG L, WANG Y K, et al. Inter-satellite link directivity algorithm research and performance validation[J]. Chinese Journal of Scientific Instrument, 2014, 35(S2): 96-100.

[2] 林琪, 李智. 星间链路天线指向建模与仿真分析[J]. 计算机仿真, 2008, 25(7): 33-37.
LIN Q, LI Z. Modeling and simulation analysis of antenna orientation on inter-satellite link[J]. Computer Simulation, 2008, 25(7): 33-37.

[3] 易先清, 张耀鸿, 钟涛, 等. 基于星间链路的导航卫星网络自适应路由策略与机制研究方法[J]. 装备学院学报, 2014, 25(1): 50-54.
YI X Q, ZHANG Y H, ZHONG T, et al. Research method about strategy and mechanism of path adaptive selection of GNII transmission based on ISL[J]. Journal of Academy of Equipment, 2014, 25(1): 50-54.

[4] 王亮, 张乃通, 刘晓峰. 低轨卫星通信网络星间链路几何参数动

- 态特性[J]. 哈尔滨工业大学学报, 2003, 35(2): 184-187.
- WANG L, ZHANG N T, LIU X F. Dynamic characters of inter-satellite links in LEO networks[J]. Journal of Harbin Institute of Technology, 2003, 35(2): 184-187.
- [5] BEURAN R, MIWA S, SHINODA Y. Network emulation testbed for DTN applications and protocols[C]//Proceedings of 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). Piscataway: IEEE Press, 2013: 151-156.
- [6] 权东晓. 基于改进的OPNET-HLA的大规模网络仿真方法[D]. 西安: 西安电子科技大学, 2016.
- QUAN D X. Large-scale network simulation method based on improved OPNET-HLA[D]. Xi'an: Xidian university, 2016.
- [7] 尹先兵, 徐展琦, 肖永伟, 等. 基于SDN的卫星LTE融合网络及其性能仿真[J]. 无线电通信技术, 2017, 43(6): 20-23.
- YIN X B, XU Z Q, XIAO Y W, et al. Converged satellite and terrestrial LTE network based on SDN and its performance simulation[J]. Radio Communications Technology, 2017, 43(6): 20-23.
- [8] MUSABEKOV S B, IBRAIMOV R R. NS-2 network performance evaluation of Abis interface over DVB-S2 in the GSM over satellite network[C]//Proceedings of 2009 First Asian Himalayas International Conference on Internet. Piscataway: IEEE Press, 2009: 1-5.
- [9] 叶海洋. 基于虚拟化的天地一体化网络仿真技术研究[D]. 无锡: 江南大学, 2019.
- YE H Y. Research on emulation technology based on virtualization for the space-ground integrated network[D]. Wuxi: Jiangnan University, 2019.
- [10] 李大辉, 高锦, 李伦, 等. 基于容器技术的虚拟网络数据通信交互方法与系统: CN110995561A[P]. 2020-04-10.
- LI D H, GAO J, LI L, et al. Virtual network data communication interaction method and system based on container technology: CN110995561A[P]. 2020-04-10.
- [11] LI H F, ZHOU H C, ZHANG H K, et al. EmuStack: an OpenStack-based DTN network emulation platform[C]//Proceedings of 2016 International Conference on Networking and Network Applications (NaNA). Piscataway: IEEE Press, 2016: 387-392.
- [12] 陈鸣, 陶小妹, 胡超, 等. 基于网络功能虚拟化的网络试验平台的设计与实现[J]. 计算机学报, 2018, 41(9): 2016-2028.
- CHEN M, TAO X M, HU C, et al. Design and implementation of network testing platform based on network function virtualization[J]. Chinese Journal of Computers, 2018, 41(9): 2016-2028.
- [13] ANSI/IEEE Std1003.1b-1993, 信息技术. 可移植操作系统接口(POSIX). 第1部分: 系统应用程序接口(API). 修改件1: 实时扩展语言(C语言)[S]. 2010.
- ANSI/IEEE Std1003.1b-1993, Information Technology. Portable operating system interface (POSIX)-Part 1: system application programming interface (API); Amendment 1: real-time extensible language (C language)[S]. 2010.
- [14] FISCHER D, BASIN D, ECKSTEIN K, et al. Predictable mobile routing for spacecraft networks[J]. IEEE Transactions on Mobile Computing, 2013, 12(6): 1174-1187.

[作者简介]



曹琦轩 (1998-), 男, 西安电子科技大学硕士生, 主要研究方向为空间信息网络。



史琰 (1975-), 男, 博士, 西安电子科技大学通信工程学院教授, 博士生导师, 主要研究方向为智能网络、空间信息网络、高性能通信与计算协同。



周连伟 (1997-), 男, 西安电子科技大学硕士生, 主要研究方向为空间信息网络。



丁新旺 (1998-), 男, 西安电子科技大学硕士生, 主要研究方向为空间信息网络。