论文题目：支持可编程数据平面的网络模拟器设计与开发

学生姓名：况鹏

指导教师：毕军、赵鹏

摘　要

从网络模拟器角度写：

网络模拟广泛应用于网络研究、教育、工业的各个方面，然而在进行网络模拟之前，传统的网络模拟器需要开发网络模拟行为模型，而这需要开发人员熟悉网络模拟器内部运行机制、函数库等。此外，由于开发的网络模拟行为模型紧密耦合于模拟器内部实现，导致开发代码不能直接迁移到真实的网络设备。最近，新兴起的以 P4 为主流的可编程数据平面技术允许操作员直接定义网络设备的行为逻辑而不用考虑网络设备的底层实现机制，正逐渐地引起学术界和工业界的关注。

受到可编程数据平面技术的驱动，我们提出 NS4，一个支持可编程数据平面技术的网络模拟器。NS4 通过将 P4 整合到传统网络模拟器 ns3 中，充分利用 P4 协议无关性、目标独立性优势，简化了网络模拟行为模型的开发，解除了开发代码与模拟器内部实现耦合性，使得开发代码能直接迁移到真实网络设备上，解决了传统网络模拟器存在的问题。此外，不同于网络仿真器，NS4 进行模拟过程中，占用系统资源有限，可为 P4 程序的有效性以及实现性能提供大规模网络拓扑级别的模拟测试。

在这篇文章中，我们设计并实现了 NS4，它由用于处理数据包行为的数据平面模块以及用于与 P4 流水线交互的控制平面模块组成。我们展现了 SilkRoad 案例来评估 NS4 有效性，构建了 NS3 与 NS4 延迟对比实验、大规模 Fattree 网络拓扑测试实验来评估 NS4 模拟效率，通过使用 P4 和 C++开发几个有代表性网络应用程序来衡量 NS4 的开发复杂度。实验结果表明 NS4 能够以较小的代价模拟大规模的 P4 驱动的网络。

从可编程数据平面角度写：

以 P4 为主流的可编程数据平面技术通过对数据包解析、处理行为进行编程，可快速支持网络新协议的设计以及新应用的开发等，在促进网络创新等方面发挥着重要作用，然而，随着 P4 程序种类、数量增加，对 P4 程序有效性及实现性能进行测试正逐渐地引起学术界和工业界的关注。

现有的 P4 程序的网络仿真软件如 Mininet、BMv2、P4Runtime 等受到仿真系统硬件资源的限制，难以扩展到大规模网络拓扑级别，导致无法对 P4 程序在大规模拓扑下进行测试。怎么引出 ns3

关 键 词：网络模拟；P4；可编程数据平面

Title: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Name: XXXXX
Supervisor: XXXXX

ABSTRACT

英文摘要的内容、格式和字号必须与中文摘要一致。
居中编排“ABSTRACT”（三号 Times New Roman）。
摘要正文，英文每段开头左对齐顶格编排，段与段之间空一行。小四号字。

The key parts in drip irrigation facilities are emitters. The structural design parameters of emitters can directly affect its performance and the function of the whole drip irrigation system ……

1. Because……

2. Only ……

3. To support ……

KEY WORDS: XXX; XXX; XXX; XXX; XXX
每个关键词组的第一个字母大写，其余为小写，每一关键词之间用分号隔开，最后一个关键词后无标点符号。例如：Drip irrigation emitter; RP&M; Hydraulics; Labyrinth flow channel

目　录

1 绪论

随着互联网规模的不断增长，网络架构变得越来越复杂，新的网络协议以及网络功能越来越难以直接部署在实际网络中进行测试和验证。因此，学术界和工业界设计了很多网络模拟系统用于对网络协议和网络功能的有效性和性能进行评估。

现有的网络模拟系统有 OPNet、REAL、PFPSim、ns 等。OPNet[1]是面向对象的通用目的网络模拟器，它使用动态进程分配技术构建虚拟电路传输模型，以便进行离散事件的模拟，它基于 Proto-C 语言，可实现几乎所有网络功能及协议，它除了规范模拟开发环境外，还为网络模拟提供了图形化的 GUI 界面。REAL[2]是一个用于研究流动态行为以及拥塞控制的网络模拟器，它使用 NetLanguage 描述网络拓扑、协议、数据和控制参数，目前已提供 30 多个模块来进行流控制协议的模拟。PFPSim[3]是一个基于可编程转发平面结构的、用于数据包处理应用程序分析和验证的网络模拟器，使用转发结构描述语言（FAD）定义转发设备体系结构，使用 C++或 P4 定义应用程序代码，它关注的是转发设备体系结构的建模。Crys-talNet[102] 是一个具有极高可扩展性的网络模拟器，它能够模拟数千个网络设备。ns 模拟工具集，由 ns1、ns2[29]、ns3[103]组成，是由不同团体和组织开发的一套基于离散事件的开源网络模拟器。其中 ns3 由 C++、Python 编写，广泛用于网络研究和教学领域，里面包含了大量由开源社区人员开发的扩展插件，可支持很多网络场景[104-106]。

在上述网络模拟系统中，ns3 应用最为广泛，操作员在使用 ns3 进行网络模拟时，需要遵循以下几个步骤：首先，根据网络协议或网络实体的设计需求开发一个网络行为模型作为模拟器的内部模块。其次，根据实际场景需要搭建网络拓扑，定义模拟过程网络任务。然后，设置模拟时间，激发网络模拟，验证模拟行为是否与预期一致。最后，当模拟行为符合预期时，需要重写模拟代码，部署到测试床或厂商设备。在这样的过程之中，网络模拟器 ns3 存在着几个重要问题：

D1：开发网络模拟行为模型是一项耗时、费劲的工作。它需要开发人员熟悉网络模拟器的内部运行机制、函数库等，而这不可避免地带来了不小的学习负担。

D2：开发的模拟代码很难迁移到真实的网络设备。由于开发的代码是由 C++编写的，大量依赖于模拟器提供的函数库，而真实网络设备很多都是用硬件描述语言（如 VHDL、Verilog）或集成电路领域中的 FPGA、ASIC 开发设计的，导致开发代码不能直接迁移到真实的网络设备，而这不可避免地带来了代码转化的过程，增加了大量重复性工作，也增大了程序出错的机率。

D1 和 D2 的本质在于网络模拟行为模型的实现紧耦合于模拟器内部结构机制，所以迫切地需要一种解除它们之间耦合性的方法。最近，新兴起的以 P4 为主流的可编程数据平面技术提供了底层设备的抽象转发模型，允许操作员直接定义网络设备行

为逻辑而不用考虑底层设备的实现细节，似乎为解除网络模拟行为模型与模拟器平台之间耦合性提供了一个可行的方法。

受到这个想法的启发，本文提出了 NS4，一个由 P4 驱动的，支持大规模网络拓扑构建的网络模拟器。通过将 P4 整合到 ns3 中，NS4 解除了行为模型与潜在模拟平台之间的耦合性。正如表 1 所示，NS4 通过 P4 定义网络模拟行为模型，不需借助任何模拟器内部函数库（D1 解决办法），此外，由于 P4 具有目标独立性优势，使得 P4 开发的行为模型可直接移植到支持 P4 的网络设备，这样避免了大量代码重写的工作（D2 解决办法）。另外由于 NS4 是基于 ns3 网络模拟平台，不同于网络仿真器，消耗系统资源有限，容易扩展到大规模网络拓扑级别，可为 P4 程序研究开发提供大规模网络拓扑级别的模拟测试。

表 1-1 ns3 与 NS4 的对比

特点	ns3	NS4
行为模型开发语言	C++	P4
耦合性	√	×
可移植性	×	√

然而在实现 NS4 的过程中，面临着如下的挑战：

C1：如何基于 ns3 模拟平台构建可编程交换机转发平面的数据包处理行为模型。可编程交换机转发平面支持多种复杂的数据包处理行为，例如重提交、再循环、克隆、多播等，里面牵涉到复杂的缓冲器、队列调度系统，如何构建出支持这些复杂机制的可编程交换机转发平面的数据包处理行为模型是一个重大挑战。

C2：如何将转发平面运行时的流表操作转换为 ns3 网络模拟系统中的离散事件。ns3 是一个基于离散事件的网络模拟器，即里面的操作是提前预计好的、由事件驱动的。然而 P4 程序的流表操作是由控制平面应用程序管理的，不是离散型的。而目前，现存的 P4 运行时控制器如 P4Runtime[32]和 ONOS-BMv2[31,62]都不能够直接与 ns3 网络模拟系统进行交互。

C3：如何基于当前网络拓扑结构设计自动流表规则。由于网络中的每个 P4 设备都需要安装流表项以执行相应的网络策略，然而当网络规模达到一定程度时，手工配置并下发每一个 P4 设备的流表项将是一件十分费力并且容易出错的工作，因此需要提供自动下发流表项机制以减轻繁重的人力负担。但是不同的网络模拟任务可能处在不同的网络拓扑环境之下，如何基于不同网络拓扑结构构建相应的自动流表规则是一个重大挑战。

NS4 内部构建了一个 P4 流水线，里面包括了缓冲器、队列的完整设计，以建立真实 P4 设备的模型。为了适应 ns3 模拟系统离散事件处理机制，NS4 提供了一个控制模块用于运行时离散操作。此外，NS4 借用了 OpenFlow[14]实时流表下发的想法，提供了一个模块用于实时计算和下发流表项，以减轻繁琐的流表规则配置工作。

整体而言，本文做了如下工作：

（1） 设计并实现了 NS4，第一个支持可编程数据平面技术的网络模拟器，源

代码可从 Github[15]访问。

（2） 设计了 P4 流水线内部缓冲器、队列系统结构来构建真实 P4 设备模型以正确地处理数据包行为，设计了几个控制模块来进行流表规则实时计算、流表下发、流表查询操作。

（3） 构建了 NS3 与 NS4 延迟对比实验、大规模 Fattree 网络拓扑测试实验以评估 NS4 有效性，使用了 P4 和 C++开发几个有代表性网络应用程序以衡量 NS4 的开发复杂度，展现了 L4 层负载平衡器 SilkRoad 以评估 NS4 可用性。

本文接下来的内容按照如下结构展开。TO DO

1.1 研究背景

随着互联网规模的不断增长，网络架构变得越来越复杂，新的网络协议以及网络功能越来越难以直接部署在实际网络中进行测试和验证。因此，学术界和工业界设计了很多网络模拟系统用于对网络协议和网络功能的有效性和性能进行评估。

现有的网络模拟系统有 OPNet、REAL、PFPSim、ns 等。OPNet[1]是面向对象的通用目的网络模拟器，它使用动态进程分配技术构建虚拟电路传输模型，以便进行离散事件的模拟，它基于 Proto-C 语言，可实现几乎所有网络功能及协议，它除了规范模拟开发环境外，还为网络模拟提供了图形化的 GUI 界面。REAL[2]是一个用于研究流动态行为以及拥塞控制的网络模拟器，它使用 NetLanguage 描述网络拓扑、协议、数据和控制参数，目前已提供 30 多个模块来进行流控制协议的模拟。PFPSim[3]是一个基于可编程转发平面结构的、用于数据包处理应用程序分析和验证的网络模拟器，使用转发结构描述语言（FAD）定义转发设备体系结构，使用 C++或 P4 定义应用程序代码，它关注的是转发设备体系结构的建模。Crys-talNet[102] 是一个具有极高可扩展性的网络模拟器，它能够模拟数千个网络设备。ns 模拟工具集，由 ns1、ns2[29]、ns3[103]组成，是由不同团体和组织开发的一套基于离散事件的开源网络模拟器。其中 ns3 由 C++、Python 编写，广泛用于网络研究和教学领域，里面包含了大量由开源社区人员开发的扩展插件，可支持很多网络场景[104-106]。

P4 是一种用于描述可编程交换机如何处理数据包行为的领域特定语言。

1.2 研究问题

1.3 本文核心想法

1.4 实现挑战

1.5 本文工作内容

2 网络模拟器 ns-3

2.1 ns-3 概况

ns-3 首次发布于 2008 年，是一个开源的基于离散事件的网络模拟器，用于网络研究和教育。ns-3 使用 C++和 Python 编写，关注于构建网络协议、网络应用程序的工作环境，因此提供了许多构造网络工作环境的模块以及用于进行模拟实验的模拟引擎。区别于其他的网络模拟工具，ns-3 提供了一系列函数库用于与外部软件库联合使用。尽管 ns-3 也支持数据分析以及可视化工具，但更推荐用户工作于命令行环境。ns-3 主要是用于 Linux 系统，也提供了对 FreeBSD、Cygwin 的支持。

2.2 ns-3 特征

在 ns-3 中，Node 是基本计算设备的抽象，可以使用 Node 添加应用程序、协议栈、外部网卡及其对应的驱动器来模仿真实网络结构。Application 是产生将被模拟的活动的应用程序的抽象，Application 提供了管理用户级别的应用程序的接口，开发者可以使用面向对象编程的方法继承 Application，实例化它们特定需求的应用程序。基本通讯子网的抽象使用 Channel 表示，它提供了管理通讯子网对象以及与节点 Node 连接的方法。网络设备抽象使用 NetDevice 表示，它提供了管理节点 Node 与通道 Channel 连接的方法。

使用 ns-3 进行模拟需要进行搭建网络拓扑，创建网络设备，赋予 Mac 地址，将网络设备安装到节点上，配置节点的协议栈，连接网络设备和通道，给节点安装应用程序，激发模拟等操作。

2.3 ns-3 典型应用

目前使用 ns-3 进行模拟的应用有很多。

3 领域特定语言 P4

3.1 P4 概况

P4 是一门用于描述网络转发设备（例如交换机、路由器、网卡、或者其他网络应用程序）如何表达数据包处理行为的语言。它基于一个由解析器、匹配行为表、逆解析器等元素组成的抽象转发模型。解析器用于提取进入的数据包的头部信息，匹配行为表会对匹配上特定流表项的数据包执行对应的行为操作，逆解析器将数据包头部与数据包内容重新组合成数据包。

使用 P4 程序，编程人员无需关注潜在底层硬件设备差异性（目标独立性），不用限制于任何特定的网络协议（协议无关性），只需按照业务需求、行为逻辑编写数据包解析、处理程序，此外，编程人员还可在将程序部署到硬件设备后修改数据包处理行为（可重构性），可满足网络更新的需求。

3.2 P4 抽象转发模型

P4 的抽象转发模型泛化了数据包是怎样被不同的技术（例如，固定功能交换机 ASICs，NPUs，可重配置交换机，软件交换机，FPGAs）在不同的转发设备（例如，以太网交换机，负载平衡器，路由器）下被处理的过程。基于抽象转发模型，开发人员可编写出目标独立的程序，再借助编译器可匹配到各种不同的转发设备。

抽象转发模型由配置、下发两种不同类型的操作控制。配置操作定义解析器，设置每个阶段中匹配动作表顺序，明确每个阶段中匹配动作表应该匹配的头部字段以及执行的动作集。配置操作决定了应支持哪种协议以及如何处理数据包。下发操作通过对特定的匹配动作表添加或删除流表项来决定对哪个数据包执行怎样的操作，以满足上层网络策略的需求。

如图 3-2 所示，抽象转发模型由解析器、入口流水线、缓存器、出口流水线、逆解析器等组成。到达的数据包首先经解析器提取头部字段，被提取的头部字段然后被传递到入口和出口流水线中的匹配动作表，被特定流表项匹配上的数据包将执行与之对应的动作。

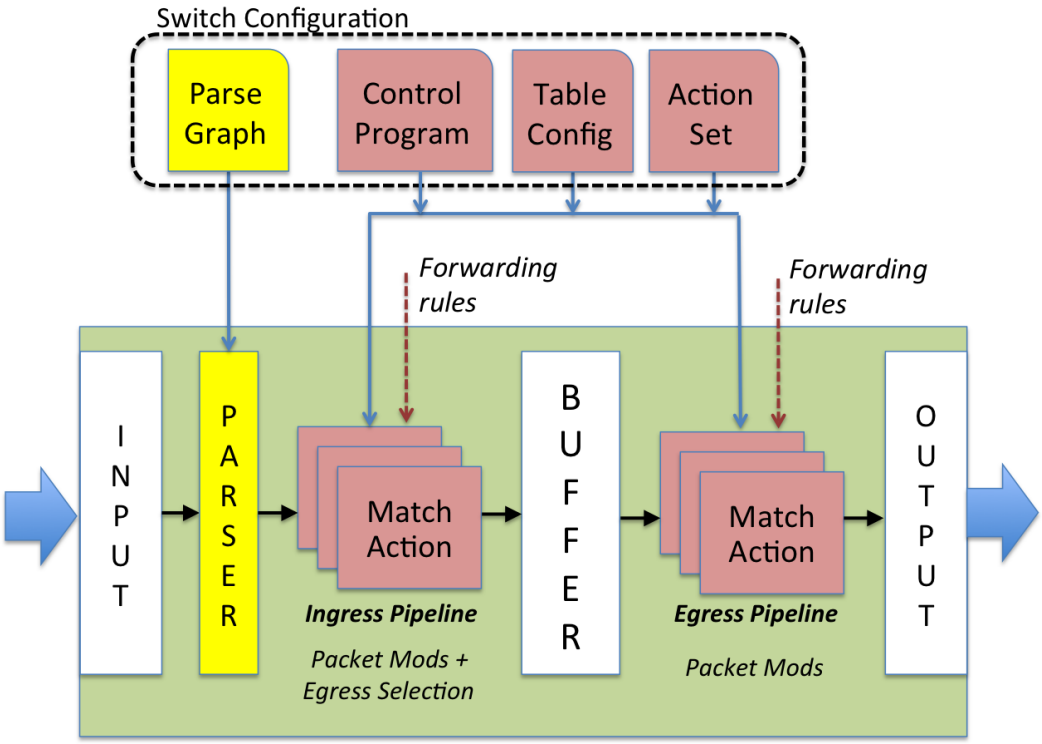


图 3-1 抽象转发模型

3.3 P4 语言要素

P4 程序包含了以下几个关键的语言要素。

3.3.1 头部

头部定义了一系列字段的顺序和结构，包括字段的宽度以及字段取值范围的限定。

3.3.2 解析器

解析器明确了怎样鉴别数据包头部以及头部中的有效顺序。P4 使用状态机来实现解析器，解析开始于起始状态，逐次按照解析定义遍历数据包头部，提取相应的头部字段，改变解析器状态，直到到达停止状态才结束解析行为，提取的数据包头部字段值将被发送到匹配动作表以供处理。

3.3.3 表

匹配动作表定义了表将要匹配的字段以及可能执行的动作，它允许编译器决定内存类型以及需要使用多少内存来实现表，它是进行数据包处理的核心机制。

3.3.4 动作

动作是描述数据包头部和元数据如何被处理的代码段，P4 支持从简单的协议无关

的原始动作集中构造复杂的动作，以实现复杂的数据包处理逻辑，P4 假定在一个复杂动作功能中的原始动作是可以并行执行的。

3.3.5 控制程序

控制程序决定了匹配动作表应用到数据包的顺序。控制程序通过函数、条件表达式、表的引用来实现控制流定义。

3.4 P4 底层架构

RMT 架构 TO DO

3.5 P4 设备（硬件、软件）

TO DO

3.6 P4 应用

TO DO

4 NS4 的设计

4.1 NS4 的结构

图 4-1 展示了 NS4 的整体架构图，NS4 由控制平面和数据平面两部分组成。控制平面包括网络路由控制器以及流表管理器，网络路由控制器用于产生数据平面中每个 NS4 网络设备的路由流表项，它包含了网络拓扑管理器、路由协议配置器、实时计算器、流表生成器四个组件；流表管理器用于下发并查询数据平面中每个 NS4 网络设备的流表，它由配置器、策略管理器、数据收集器三个组件组成。数据平面包含了用于模拟可编程交换机处理数据包行为的 NS4 网络设备模块，NS4 网络设备模块包含了 P4 配置器、流表操作接口、通道管理器、P4 流水线这几个部分。

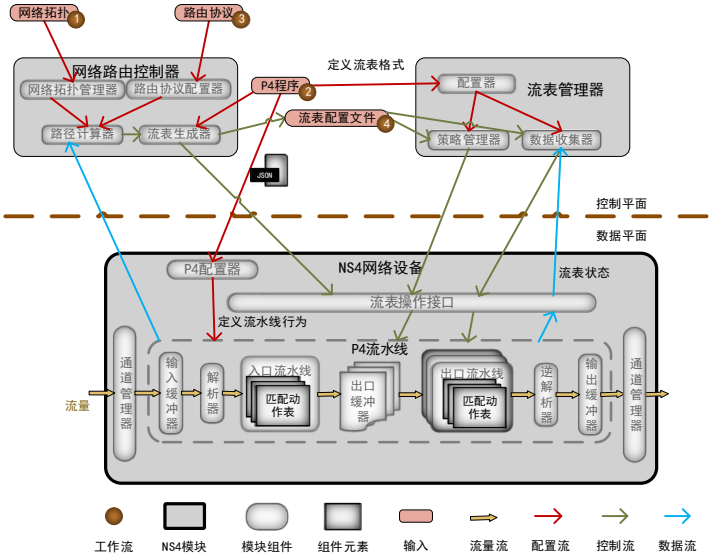


图 4-1 NS4 系统架构图

管理员可以通过网络实例化 NS4 网络设备来加载 P4 程序以及通过控制平面模块下发匹配规则来实现数据平面的转发逻辑功能。NS4 网络设备模块在实现上继承了 ns-3 模拟系统中的网络设备模块，因此，可以通过 ns-3 提供的基础组件，将 NS4 网络设备模块连接到其他 NS4 网络设备或传统网络设备（如路由器或交换机）上。利用了 ns-3 模拟系统占用系统资源较小的优势，可以基于 NS4 网络设备模块构建大规模网络拓扑以进行关于 P4 程序网络级别的模拟测试。

尽管 ns-3 本身的设计支持与外部设备交互通信，但是本文并不推荐通过模拟系统外部的控制器控制 NS4 网络设备。由于外部控制器的性能依赖于外部系统的资源及负载，使用外部控制器与 NS4 网络设备进行通信，会使得 NS4 的模拟结果依赖于外部系统，这意味着即使相同的 P4 程序在不同的外部系统的环境下会得到不同的模拟结果。为了在不引入任何外部控制器的情况下实现数据平面的大规模网络拓扑模拟，NS4 设计了多个内部控制平面模块，便于管理员在模拟系统内部下发流表规则，避免因外部系统的不稳定性、差异性影响模拟结果的准确性。

网络模拟前，管理员需要提供一个流表配置文件，该文件会包含数据平面上匹配动作表的流表操作规则。在模拟开始时，该配置文件会被 NS4 中流表管理器模块转化为 NS4 系统内部的离散事件，NS4 将在指定时间执行对应的符合流表规则的匹配动作操作。但当网络规模增大到一定程度时，手工配置每个 NS4 网络设备的配置文件无疑会耗费大量的人力成本以及时间成本。为了降低大量配置工作开销，NS4 设置了网络路由控制器用于自动化产生 NS4 网络设备的流表规则。

利用 NS4 进行网络模拟的工作流如图 4-1 所示：（1）管理员首先根据网络场景需求定义网络拓扑结构；（2）将每个 NS4 网络设备需要的 P4 程序编译成 json 格式文件，通过 P4 配置器加载到 NS4 网络设备内部 P4 流水线中，以配置数据平面的逻辑行为；（3）根据网络路由协议自动产生数据平面中 NS4 网络设备的流表配置文件，如果需

要，还可手工添加其余额外的流表规则；（4）定义数据收集任务，查询数据平面中 NS4 网络设备的流表下发状态；（5）安装应用程序，触发网络模拟执行操作。

4.2 NS4 的数据平面

NS4 网络设备被设计来构建 P4 设备的模型，是 NS4 可编程数据平面的基本模块。通过提供连接其他网络设备的接口，NS4 网络设备确保操作员能够简单地构建一个由多个网络设备组成的支持 P4 的网络。通过提供流表操作接口，NS4 网络设备可与控制平面中流表管理器联系，以配置 NS4 网络设备中 P4 流水线的网络策略。NS4 网络设备的基本目标是模拟真实世界 P4 设备的行为。通过将完整的 P4 流水线整合到 NS4 网络设备中，NS4 网络设备可按照 P4 程序的定义执行相应的数据包处理行为。为了确保对 P4 程序重提交、克隆、多播、再循环等复杂行为的支持，NS4 设计了对应的队列、缓冲器结构。

4.2.1 流表操作接口

为了与控制平面的流表管理器进行联系，以方便地进行流表下发、查询等操作，NS4 基于 P4 流水线抽象出了一个流表操作接口。流表操作接口包含了两类流表操作类型，一类用于给 NS4 网络设备添加特定流表项，以执行高层网络策略，包括添加、修改、删除、清空流表规则，设置测量器速率、设置寄存器、设置计数器等操作，另一类用于查询 NS4 网络设备中特定匹配动作表的流表状态，包括查询流表数量、查询流表内容、读取测量器速率、读取计数器、读取寄存器等操作。

4.2.2 P4 流水线

P4 流水线是 NS4 网络设备的核心。为了配置 P4 流水线的功能逻辑，NS4 提供了 P4 配置器以及流表操作接口来配置 P4 流水线的功能、设置相应的网络策略机制。在 NS4 网络设备的初始化阶段，P4 配置器会将已编译完的 P4 程序导入到 P4 流水线中，来设置 P4 流水线中的数据包处理行为。在运行阶段，流表操作接口将被控制平面中的流表管理器调用以进行添加、删除、修改、查询流表条目的操作。此外，NS4 在 P4 流水线的外部添加了一个通道管理器来隐藏潜在通道的细节，并为 P4 流水线提供了一个统一的接口，使得 NS4 网络设备可与各种不同网络设备相连接，由此扩展了 NS4 可模拟的 P4 网络设备的规模。

图 4-2-1 展示了 P4 流水线内部结构。P4 流水线由输入缓冲器、解析器、入口流水线、出口缓冲器、出口流水线、逆解析器、输出缓冲器组成。输入缓冲器用于缓存进入 P4 流水线中的数据包。解析器用于提取数据包的头部字段，生成数据包头部向量，以供入口流水线和出口流水线中的匹配动作表进行处理。位于入口流水线中的匹配动作表会产生一个出口定义，用于决定数据包将要转发的端口。出口缓冲器用于缓存即将进入出口流水线的数据包，其内部实现了复杂的队列机制。出口流水线也可对数据包头部字段进行修改，但默认不可更改数据包的物理目的地址。逆解析器会将数

据包头部向量以及数据包携带的数据重新组合成数据包。输出缓冲器用于缓存从逆解析器流出的数据包，并将其转发至其他网络设备。

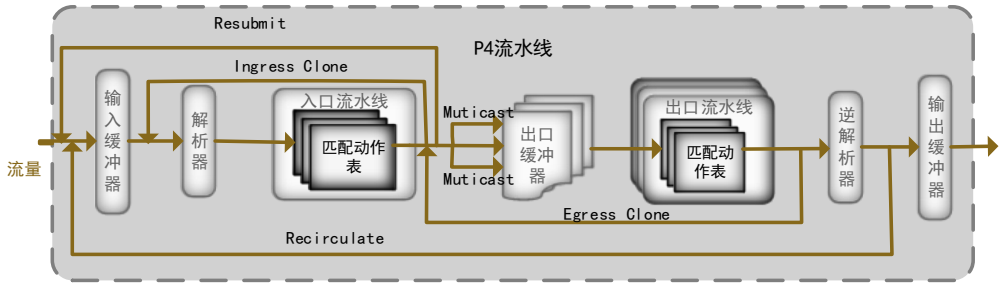


图 4-2 P4 流水线

4.2.3 队列系统

为了支持 P4 程序中重提交、克隆、多播、再循环等复杂机制，NS4 提供了缓冲器、队列系统的设计。如图 4-2-1 所示，NS4 内部的 P4 流水线中包含输入缓冲器、出口缓冲器、输出缓冲器三种不同缓冲器。输入缓冲器与输出缓冲器内部均由先进先出队列实现，只对数据包提供了一个缓存功能。而出口缓存器内部由优先级队列实现，且每个出口缓存器包含多个优先级队列，位于高优先级队列中的数据包会优先被处理。此外，为了模拟真实网络设备行为，可对优先级队列中的数据包设置延迟时间，还可利用优先级队列实现限速功能，以限制队列处理数据包最大速率。为了减少数据包重排序时间，优先级队列会按照发送时间对其中的数据包进行排序。（删除？）出口缓冲器数量一致于出口流水线的数量，从每个出口缓冲器流出的数据包会进入对应的出口流水线。（需不需要说明出口流水线的实现机制，比如线程）

基于设计的缓冲器、队列结构，NS4 提供了入口克隆和出口克隆两种克隆机制。克隆会产生一个新的数据包实例，并不影响被克隆的数据包，入口克隆会克隆从入口流水线流出的数据包，并将其发送给解析器处理，出口克隆则将出口流水线流出的数据包克隆到出口缓冲器前。除了克隆机制以外，NS4 还提供了对重提交、再循环、多播机制的支持。重提交会将从入口流水线流出的数据包以及相应的元数据重新提交到输入缓冲器，再次执行解析操作；不同于重提交，再循环则将来自于逆解析器之后的，被入口流水线和出口流水线处理完的数据包发送至输入缓冲器，以再次进行流水线处理操作；（TO DO：补充下每种机制的应用案例）多播发生在入口流水线之后，会将原数据包复制多份发送至多个不同的出口缓冲器，以完成上层策略的广播、多播等机制。

4.3 NS4 的控制平面

NS4 的控制平面主要有两个设计目标：（1）将流表配置文件转化为离散事件，并在指定时间触发事件；（2）根据网络拓扑结构以及所设置的路由协议，计算数据包路由

路径，并根据 P4 程序定义，自动产生特定格式的流表项，以满足网络路由需求。

针对目标 1，NS4 设计了流表管理器，它会根据流表配置文件描述，在指定的时间点给对应的 NS4 网络设备下发给定的流表项。针对目标 2，NS4 设计了网络路由控制器，里面包含了网络拓扑管理器、路由协议配置器、路径计算器、流表生成器等组件。

4.3.1 流表管理器

为了满足网络上层策略需求，使得 NS4 网络设备能够针对特定数据包执行特定的动作，NS4 控制平面提供了流表管理器，用于给 NS4 网络设备下发流表操作。流表管理器由配置器、策略管理器、数据收集器组成，配置器会根据 P4 程序定义确定下发流表格式，具体包括匹配动作表的匹配参数个数、匹配类型、行为参数个数、行为类型，测量器类型、计数器类型等。策略管理器用于给 NS4 网络设备下发添加、删除、修改等影响 NS4 网络设备内部流表项数目或内容的规则，数据收集器用于给 NS4 网络设备下发查询流表状态的规则。

为了符合于 ns-3 基于离散事件的执行机制，流表管理器会将流表配置文件转化为离散事件，如表 4-3-1 所示，NS4 离散事件描述包括时间戳、NS4 网络设备 ID、命令类型以及命令参数。时间戳决定了什么时候执行该离散事件，NS4 网络设备 ID 为执行该事件的对应目标设备。表 4-3-2 展示了离散事件支持的命令类型及相应的参数，包括匹配动作表的增删改查以及计数器、测量器、寄存器的读写等操作。其中匹配动作表的操作事件由策略管理器完成，匹配动作表的状态读取事件由数据收集器完成。为了记录和调度离散事件，策略管理器和数据收集器的内部均维护了一个优先级队列，其中事件的时间戳为队列的优先级，优先级队列会根据事件的时间戳以及当前的模拟时间来调度执行事件，一旦模拟时间与事件的时间戳相一致，该事件就会被调度执行。

表 4-1 NS4 离散事件描述

Time stamp	Device ID	Command Type	Parameters

表 4-2 NS4 流表命令描述

命令类型	参数	描述	执行组件
table_set_default	<table name> <action name> <action parameters>	设置匹配动作表默认行为	策略管理器
table_add	<table name> <action name> <match fields> => <action parameters> [priority]	给匹配动作表增加一条流表项	策略管理器
table_modify	<table name> <action name> <entry handle> [action parameters]	修改匹配动作表特定流表项行为参数	策略管理器
table_delete	<table name> <entry handle>	删除匹配动作表特定流表项	策略管理器
meter_set_rates	<name> <index> <rate_1>:<burst_1> <rate_2>:<burst_2> ...	设置测量器速率	策略管理器
counter_reset	<name>	重置计数器	策略管理器
register_write	<name> <index> <value>	设置寄存器值	策略管理器
table_num_entries	<table name>	读取匹配动作表流表数	数据收集器

table_dump_entry	<table name> <entry handle>	读取匹配动作表流表内容	数据收集器
meter_get_rates	<name> <index>	读取测量器速率	数据收集器
counter_read	<name> <index>	读取计数器	数据收集器
register_read	<name> [index]	读取寄存器	数据收集器

4.3.2 网络路由控制器

为了解决手工配置流表规则过于繁琐的问题，NS4 提供了网络路由控制器，用于自动产生路由流表项。网络路由控制器由网络拓扑管理器、路由协议配置器、路径计算器、流表生成器组成。网络拓扑管理器内部维护一张网络拓扑结构图，图中节点为主机或转发设备，边为主机到转发设备、主机到主机、转发设备到转发设备之间的连接链路，节点包括 IP 地址、Mac 地址等信息，边包括链路带宽、链路延迟等信息。路由协议配置器用于配置网络路由协议，包括距离向量路由协议（NixVector）、等价路由协议（ECMP）等，以决定数据包选择路径方式。路径计算器根据网络拓扑结构以及所配置的路由协议，为通讯对计算一条路由路径。流表生成器会根据所计算的路由路径，以及该路径中每一个 NS4 网络设备的 P4 程序，自动产生每个 NS4 网络设备需要的流表项。

网络路由控制器提供了实时计算、提前预置两种工作模式。实时计算借鉴了 OpenFlow 首包上传被动式下发匹配规则的思想，具体是将 P4 流水线中匹配动作表的默认无匹配动作行为，修改为将数据包上传到网络路由控制器，网络路由控制器在接收到该数据包之后，会根据该数据包的源地址和目标地址以及网络拓扑结构信息计算出相应的路由路径，自动实时产生所需要的流表项，并将其下发到 P4 流水线中。实时计算虽然避免了繁琐的流表配置问题，但不可避免的引入了控制路径—即 P4 流水线到网络路由控制器以及网络路由控制器到 P4 流水线—的通讯时间开销，可能会影响网络模拟结果的可靠性。为了解决实时计算模式引入控制路径的问题，网络路由控制器还提供了提前预置这一种工作模式，相比于实时计算在模拟运行时实时产生路由流表项，提前预置模式下，主机通讯对所需要的路由流表项，则是在网络模拟前产生。提前预置需要输入主机通讯对的信息，以便计算通讯主机之间的路由路径。提前预置不适用于模拟前不知道通讯主机对的场景（例如，随机选择通讯对），尽管可以通过输入所有可能通讯对信息来产生所有主机通讯之间的路由流表项，以实现该场景的主机通讯，但无疑会造成大量流表项浪费，耗费过多内存资源。

实时计算的优点在于只下发所需要的路由流表项，节省了内存资源，缺点则是引入了控制路径，带来了一定的时间开销，也损失了一定的网络模拟可靠性。提前预置通过牺牲空间换时间的方式解决了实时计算控制路径的问题，但可能会耗费过多的内存资源。

5 NS4 的实现

NS4 是基于开源的 ns-3 网络模拟系统编写的、支持 P4 的、可模拟大规模 P4 网络

的模拟系统。NS4 主要是使用 C++编写，里面也包含由 python、P4 编写的部分，代码编写风格遵循于 GNU 编码标准，目前包括近万行代码，可成功在 Ubuntu16.04 上运行，代码完全公开，可从 https://ns-4.github.io/获取。

在实现上，NS4 将 bmv2[]中的 P4 流水线嵌入到 ns-3 交换机中，使其能够按照 P4 程序定义对数据包进行处理，以支持 P4 语言功能。为了实现 P4 程序中的队列机制以及模拟出口流水线的行为，NS4 引入了优先级队列以及多线程编程模式。

5.1 内部模块

为了便于维护以及易于扩展，NS4 采用面向对象编程模式，对很多内部结构进行抽象。整体而言，NS4 内部包括 P4 流水线，P4 网络设备、P4 交换机接口、流表操作管理、网络路由控制、异常处理等模块，以下将对各个模块进行简单地介绍。

5.1.1 P4 流水线模块

P4 流水线模块是 NS4 的核心模块，里面定义了一个 P4Model，其继承于 bmv2 中的 Switch 原型，里面实现了 P4 流水线的完整处理过程，包括队列系统处理机制。（TO DO：详细说明下队列系统？）

5.1.2 P4 网络设备模块

P4 网络设备模块是沟通 ns-3 与 P4 流水线的桥梁，里面定义了一个 P4NetDevice，其继承于 ns-3 中的 NetDevice，以便与 ns-3 其余网络设备进行联系。P4NetDevice 内部包含了 P4Model，以进行 P4 流水线操作。

5.1.3 P4 交换机接口模块

P4 交换机接口模块是沟通 P4 流水线与流表管理器的重要组件，里面定义了 P4SwitchInterface，其对 P4 流水线的流表操作方式进行了抽象，提供了增删读改等流表操作方法的接口。

5.1.4 流表管理模块

流表管理模块是位于控制平面的，用于给数据平面的 P4 流水线下发流表以完成上层网络策略需求的一个模块，里面定义了 FlowTableManager，其实现了更改匹配动作表的流表项以及查询匹配动作表的流表状态这两种类型的方法。（TO DO：添加 StrategyManager 以及 StateAttainManager？以及及其优先级队列信息）

5.1.5 网络路由控制模块

网络路由控制模块用于计算数据包路由路径，里面定义了 RouteController 以及 TopoReader，TopoReader 用于读取网络拓扑，位于 RouteController 内部，RouteController 提供了多种计算路由路径的方式，比如距离向量（NixVector），等价路由（ECMP）等等。

5.1.6 异常处理模块

异常处理模块内部定义了 P4Exception，主要是对下发流表操作进行异常检查并处理，具体包括对调用函数名称、参数个数及类型做验证，以便快速定位具体出错位置，方便运行及调试。

5.2 NS4 支持特征

NS4 将 ns-3 与 bmv2 相结合，致力于打造一个支持 P4 程序的、可进行大规模网络模拟测试的网络模拟系统。NS4 需要运行在 ns-3 以及 bmv2 的部署环境下，目前可成功运行在 Ubuntu16.04 中，理论上还可支持其余能安装 ns3 以及 bmv2 的 Linux 操作系统或 Linux 容器工具。

通过完整地将 bmv2 中的 P4 流水线整合到 ns-3 中，NS4 提供了等同于 bmv2 对 P4 语言的支持力度。可支持 P414[]以及 P416[]两种 P4 语言规范，以及由其定义的寄存器、计数器、测量器等带状态的操作，和克隆、重提交、再循环、多播等数据包处理行为机制。

（TO DO：添加一张表，写明 NS4 支持的特征）

5.3 NS4 的扩展性

（TO DO）

5.4 NS4 的局限性

（TO DO）

6 NS4 的评估

为了评估 NS4 的有效性、模拟效率和开发复杂度，本文设计了如下实验进行评估：（1）搭建 Fattree[4]网络拓扑，并在核心交换机上实现使用 P4 语言开发的负载平衡器 SilkRoad[38]来评估 NS4 的有效性。(2)分别在只含有单个交换机的网络拓扑以及 Fattree 网络拓扑下，构建 ns-3 与 NS4 的模拟对比实验，从执行时间以及资源利用率两个方面评估 NS4 的模拟效率。（3）将几个有代表性的位于数据平面的网络应用程序作为测试案例，通过对比 C++和 P4 开发的代码行数，衡量 ns-3 模拟与 NS4 模拟的实现复杂度。以上所有的实验均是在 Dell R730xd PowerEdge 服务器[108] 上进行，该服务器配有 64GB 内存，两个 Intel Xeon-2620 CPU，和一个千兆网卡。

6.1 SilkRoad 应用案例

为了评估 NS4 的有效性，本文使用 NS4 实现了负载均衡器 SilkRoad 的应用案例。在云数据中心中，很多流量都带有一个虚拟 IP 地址（Virtual IP Address，VIP），并且需要通过负载均衡器映射到具有直接 IP 地址（Direct IP Address，DIP）的服务器。目前，云数据中心网络使用软件负载均衡器（SLB）实现了上述地址映射功能[109]，但是带来了巨大的性能开销。通过利用 P4 语言在可编程数据平面实现负载均衡功能，SilkRoad 实现了数据包线性转发性能。然而，由于缺乏有效的 P4 程序的大规模网络级别的模拟验证系统，研究者无法评估 SilkRoad 在大规模网络拓扑下的正确性和性能。NS4 的提出弥补了这项空白，使得管理员可以验证数据平面程序在大规模网络拓扑下的有效性和性能。

为了评估 SilkRoad 的表现性，本文实现了 SilkRoad 原型，并将其部署在 k=8 的 Fattree 网络拓扑下。为了与 SilkRoad 进行对比，本文在 ns-3 平台下实现了软件负载均衡器 SLB 的应用程序。如图 6-1 所示，为了更好地展示模拟过程，本文使用了 k=4 的 Fattree 网络拓扑。为了对数据中心流量进行模拟，本文按照指数随机分布 on-off 模式[110]生成 VIP 流量。

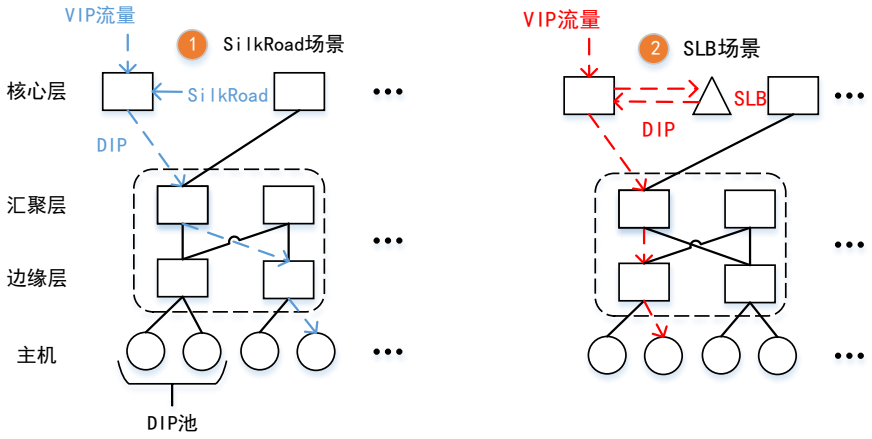
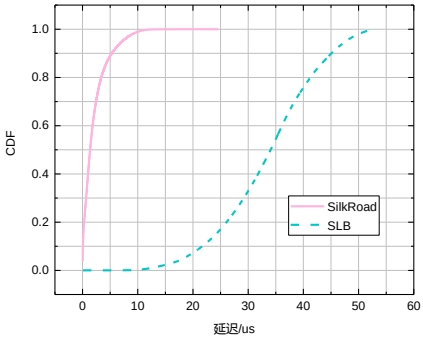


图 6-1 SilkRoad 部署网络

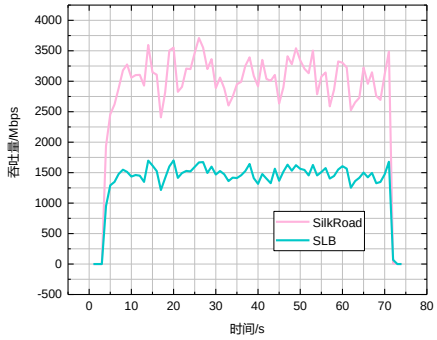
连接到同一边缘交换机的主机被视为具有相同VIP的服务器池(Server Pool)并提供特定服务。接收到携带 VIP 的数据包后，如图 6-1 中的 SilkRoad 场景，SilkRoad 交换机会根据 P4 程序定义的负载均衡机制进行相应的地址映射；而普通交换机则必须将数据包转发到软件交换机 SLB 以获取数据包的 DIP，之后普通交换机将数据包发送至相应的服务器，这个过程如图 6-1 中的 SLB 场景。在模拟过程中，本文测量了流完成时间(Flow Completion Time,FCT)、延迟时间和吞吐量等指标以比较 SilkRoad 与 SLB 的性能。

图 6-2(a)展示了两种场景下数据包延迟的累积分布函数(Cumulative distribution function,CDF)图。由于 SilkRoad 直接在数据平面实现了负载均衡的功能，避免了将数据包发往 SLB 以及从 SLB 发回数据包过程，因此 SilkRoad 场景中的数据包比 SLB 场景中的数据包具有更小的延迟时间。图 6-4(b)展示了两种场景下系统吞吐量随时间变化情况图，从图中可以发现，两种场景下系统吞吐量均会随着模拟开始而增加，并在模拟结束之前降至零。在模拟过程中，SilkRoad 系统吞吐量会始终优于 SLB 系统，这是因为 ASIC 处理数据包的速度远高于 SLB。图 6-2(c)和 6-2(d)

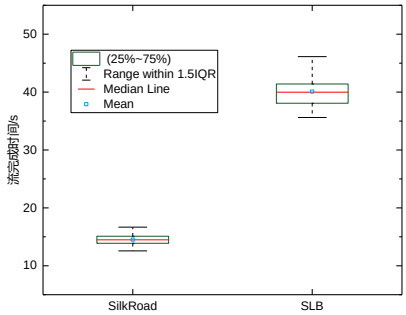
展示了两种场景下数据流完成时间和吞吐量。可以看到，在 SilkRoad 场景下，数据包直接在交换机上处理，相比于 SLB，数据流完成时间更短，数据流吞吐量更大。本节实验证明了 NS4 对于支持研究人员在大规模网络上模拟 P4 程序并进行有效性验证和性能评价的可行性。



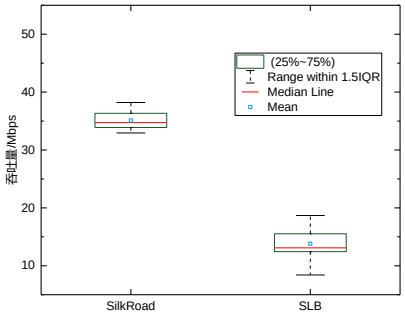
(a) 延迟累计分布



(b) 系统吞吐



(c) 流完成时间



(d) 流吞吐

图 6-2 SilkRoad 与 SLB 模拟性能对比

6.2 模拟表现性

6.2.1 P4 流水线延迟效应

为了评估 NS4 中 P4 流水线的延迟效应，我们搭建了一个简单的由一个交换机连接两个主机的网络拓扑，并选择其中一个主机作为发送端，向另一个主机发送 UDP 数据包，通过更改发送 UDP 数据包的数量，来分别测量 ns-3 与 NS4 场景下的模拟时间，以评估 NS4 中 P4 流水线的延迟效应。

如图 6-3 所示，随着发送数据包数量的增大，两种场景下的模拟时间均会线性的增加，并且 NS4 场景下的模拟时间均高于 ns-3，这是因为在 NS4 场景下，数据包均需要经过 P4 流水线处理，因此会带来一定的延迟时间。

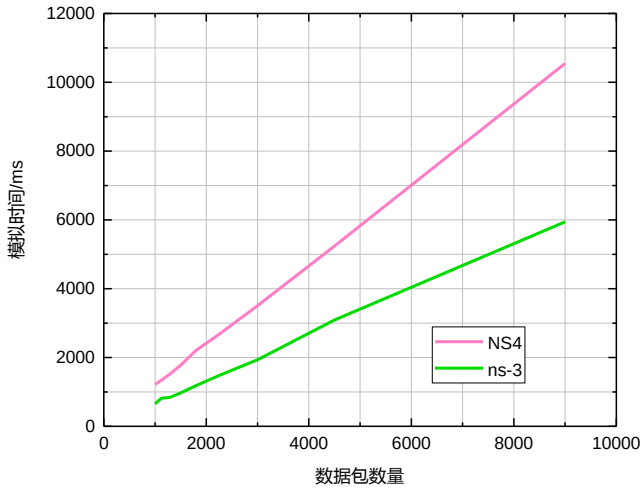


图 6-3 NS4 中 P4 流水线延迟效应

6.2.2 NS4 系统性能

为了评估 NS4 模拟大规模网络的性能，本文使用 NS4 模拟具有不同 K 值的 Fattree 网络。为了使得主机通讯覆盖整个网络范围，以评估整个网络功能，我们按照图 6-4 的方式选择通讯对，假设网络中主机数为 n，则第 i 个主机和第 n-i+1 个主机进行通讯。为了缩短模拟系统运行时间，我们将系统模拟时间设置为 10 秒，并且让每个通讯对每隔一秒发送一个数据包。为了让 NS4 系统正常运行，我们提前计算好了每个通讯对的路由路径，并且提前将所需流表项下发到每个交换机中。我们将 Fattree 网络的 K 值从 2 逐渐增大到 24，并且测量了 CPU 利用率、内存使用量、模拟系统的执行时间以评估模拟系统的性能，图 6-5 展示了 NS4 与 ns-3 模拟系统的性能对比。

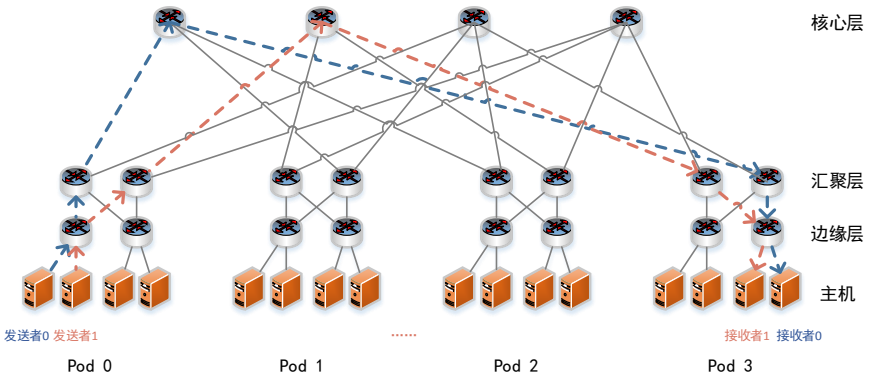
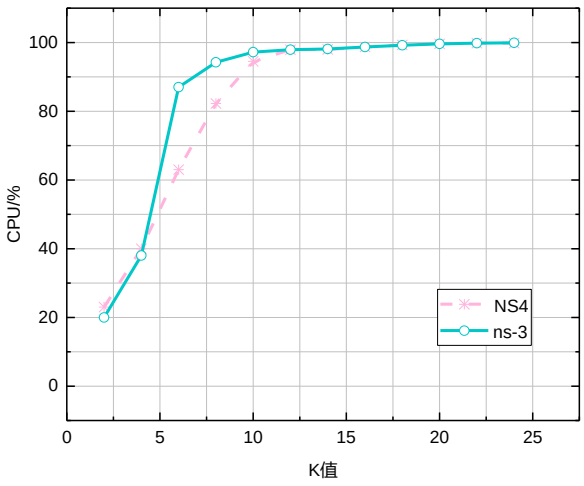


图 6-4 K 等于 4 的 Fattree 网络

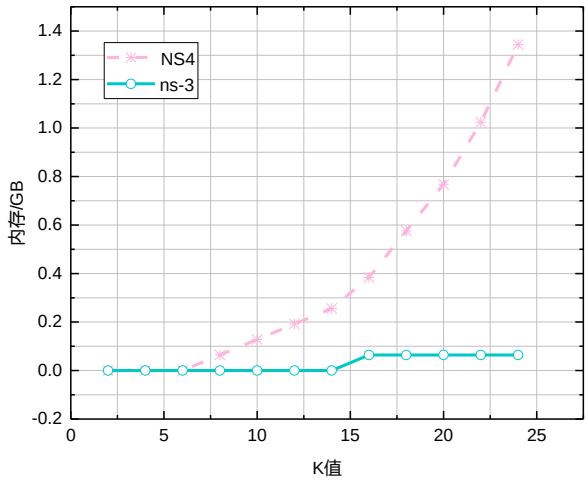
从图 6-5(a)和图 6-5(b)可知，当 Fattree 的 K 值逐渐增大时，NS4 与 ns-3 模拟系统的 CPU 使用率变化情况相差不大，但 NS4 对内存的消耗却要远高于 ns-3。NS4 内存使用量从 65MB 逐渐增大到 1376MB，而 ns-3 内存使用量基本维持在 65MB 左右，这是因为 NS4 与 ns-3 采取了两种不同的路由策略。NS4 通过提前计算路由路径，并且提前下发路由流表项到交换机的方式，导致了更大的内存消耗(0(k^3)数量级内存使用量)，但相应地带来了更短的执行时间；作为对比，ns-3 采取了实时计算的路由策

略，需要在模拟过程中实时计算数据包的路由路径，所以尽管会减少内存使用量，但不可避免地需要更长的执行时间。

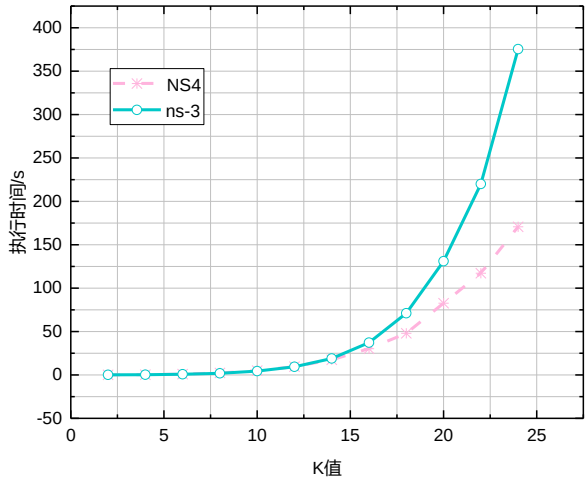
图 6-5(c)展示了两种模拟系统的执行时间对比。从局部变化情况来看，当 Fattree 的 K 值从 2 逐渐增大至 12 时，NS4 的执行时间要略高于 ns-3，当 K 值从 14 增大至 24 时，ns-3 的执行时间要高于 NS4，且它们执行时间的差值变得越来越大。这是由于 NS4 中 P4 流水线带来的时间延迟，与 NS4 提前下发路由流表项避免实时计算路由路径从而减少时间消耗相互平衡的结果。当 K 值小于或等于 12 时，由于网络中主机与交换机数较小，实时计算所有通讯主机之间路由路径时间要小于 P4 流水线带来的时间延迟，所以 ns-3 的执行时间要略低于 NS4。但当 K 值从 14 逐渐增大到 24 时，由于网络中主机数(O(k^3)数量级)和交换机数(O(k^2)数量级)迅速增加，导致网络中链路数(O(k^3)数量级)也相应地增加，从而使得实时计算所有通讯主机之间路由路径时间要逐渐地高于流经 P4 流水线带来的时间延迟，所以这个时候 ns-3 的执行时间要高于 NS4。从整体变化趋势来看，ns-3 执行时间的增长速度要高于 NS4，这是因为实时计算通讯主机路由路径的时间主要由网络中链路数决定，其为 O(k^3)数量级，而流经 P4 流水线时间由网络中交换机数决定，其为 O(k^2)数量级。可以预见，当网络规模更大时，NS4 的执行时间要远低于 ns-3，这也表明 NS4 有着强于 ns-3 的网络扩展性。



(a)CPU 利用率



(b) 内存使用量



(c) 执行时间

图 6-5 NS4 模拟系统性能

6.3 开发复杂度

为了对比评估 NS4 与 ns-3 的开发复杂度，本文选取了几个有代表性的数据平面的网络应用程序。本文选取的基础应用程序是网络中最常见的用于转发数据包的二层/三层交换机(L2/L3 Switch)，在它的基础之上，本文构建了其他完成特定网络功能的应用程序，包括：用于过滤特定数据包的访问控制列表(Access Control List, ACL)，用于将私有 IP 地址转化为合法 IP 地址的网络地址转换(Network Address Translation, NAT)，用于对端口转发的报文进行过滤控制，防止非法报文入侵端口的源防护(Source Guard, SG)，以及用于防止数据包洪泛的风暴控制(Storm Control, SC)等。由于 P4 程序的头部定义和解析器定义是通用的，并且编写它们是一次性的行为，所以本文只计算 P4 程序

里面的匹配动作表以及控制流代码行数。如表 6-1 所示，随着支持网络功能的数量增加，两个模拟系统所需要编写的代码行数也逐渐增加，得益于 P4 在描述数据平面网络设备行为的高适用性，使用 P4 编写的网络应用程序的代码行数要远小于 ns-3 中使用 C++编写的，此外，由于 P4 具有目标独立性的优势，开发者只需专注于实现数据平面数据包处理的行为逻辑，而无需关注模拟系统内部的开发框架以及函数库，且开发代码可直接移植到其余支持 P4 的硬件设备，使得网络模拟过程变得更加简单高效。

表 6-1 网络应用程序开发复杂度

案例	网络应用程序	ns-3	NS4
1	L2/L3 Switch	598	165
2	L2/L3 Switch, ACL	803	252
3	L2/L3 Switch, ACL, NAT	1038	494
4	L2/L3 Switch, ACL, NAT, SG, SC	1219	637

7 NS4 的使用（TO DO：找一个小型的网络应用场景）

7.1 场景说明

7.2 P4 程序的定义

7.3 网络拓扑定义

7.4 模拟程序定义

7.5 流表规则定义

7.6 模拟

8 结论与展望

结论与展望：结论包括对整个研究工作进行归纳和综合而得出的总结；所得结果与已有结果的比较；联系实际结果，指出它的学术意义或应用价值和在实际中推广应用的可能性；在本课题研究中尚存在的问题，对进一步开展研究的见解与建议。结论集中反映作者的研究成果，表达作者对所研究课题的见解和主张，是全文的思想精髓，是全文的思想体现，一般应写得概括、篇幅较短。撰写时应注意下列事项：

- 结论要简单、明确。在措辞上应严密，但又容易被人领会。
- 结论应反映个人的研究工作，属于前人和他人已有过的结论可少提。
- 要实事求是地介绍自己研究的结果，切忌言过其实，在无充分把握时，应留有余地。

8.1 标题 2

8.1.1 标题 3

公式按章重新编号：

$$\frac{n!}{r!(n-r)!}\frac{1}{2} \tag{12-1}$$

公式（12-1）说明，……（公式在正文中的引用）

图题注：

图 8-1　XXXXXX

致　谢

致谢：对于毕业设计（论文）的指导教师，对毕业设计（论文）提过有益的建议或给予过帮助的同学、同事与集体，都应在论文的结尾部分书面致谢，言辞应恳切、实事求是。应注明受何种基金支持（没有可不写）。

参考文献

（此上两空行不能删除，是为 EndNote 的参考文献列表所预留）

文后著录的参考文献务必实事求是。论文中引用过的文献必须著录，未引用的文献不得出现。应遵循学术道德规范，避免涉嫌抄袭、剽窃等学术不端行为。

参考文献一般应是作者亲自考察过的对学位论文有参考价值的文献，除特殊情况外，一般不应间接引用。

参考文献应有权威性，要注意引用最新的文献。

参考文献的数量：

毕业论文一般不得少于 10 篇，其中外文文献不得少于 2 篇。

参考文献的著录格式应符合国家标准 GB/T 7714-2005《文后参考文献著录规则》。参考文献中每条项目应齐全。

文献中的作者不超过三位时全部列出，超过三位时，一般只列前三位，中文的后面加“等”字，英文的后面加“et al”，作者姓名之间用逗号分开。

外国人名一般采用姓在前，名在后的著录法，姓全写且第一个字母大写，名简写成单个大写字母且不加标点，姓和名之间空 1 格，如：“Metcalf SW”。也可采用名在前，姓在后的著录法，姓全写且第一个字母大写，名简写成单个大写字母且不加标点，名和姓之间空 1 格，如：“SW Metcalf”。

中文人名的英文表达方式：

简写时，采用姓在前，名在后的著录法，姓全写且第一个字母大写，名简写成单个大写字母且不加标点，如，“钱学森”，简写为“Qian XS ”。

全拼时，名在前，姓在后的著录法，名的第一个字母大写，名连写，名后空 1 格写姓，姓的第一个字母大写。如，“钱学森”，写为“Xuesen Qian”。

文后参考文献著录格式范例样板，采用五号。

具体要求如下：

A 专著（包括普通图书[M]、论文集和会议录[C]、科技报告[R]、学位论文[D]、标准[S]）

主要责任者. 文献题名[文献类型标志]. 其他责任者. 版本项(第 1 版不标注) . 出版地：出版者，出版年：引文页码. 获取和访问路径.

B 专著中的析出文献

析出文献主要责任者. 析出文献题名[文献类型标志]. 析出文献其他责任者//专著主要责任者. 专著题名：其他题名信息. 版本项(第 1 版不标注) . 出版地：出版者，出

版年：析出文献的起止页码．获取和访问路径．

C 连续出版物

主要责任者．题名:其他题名信息［文献类型标志］．年，卷（期）—年，卷（期）．出版地：出版者，出版年．获取和访问路径．

D 连续出版物中的析出文献（包括期刊中析出的文献[J]、报纸中析出的文献[N].）

析出文献主要责任者．析出文献题名［文献类型标志］．连续出版物题名：其他题名信息，年，卷（期）：页码．获取和访问路径．

E 专利文献

专利发明者/专利申请者或所有者．专利题名：专利国别,专利号［文献类型标志］．公告日期或公开日期．获取和访问路径．

F 电子文献（包括专著或连续出版物中析出的电子文献）

主要责任者．题名：其他题名信息[文献类型标志/载体类型标志]．出版地：出版者，出版年（更新或修改日期）．获取和访问路径．

表 2-2 文献类型和标志代码

文献类型	标志代码	文献类型	标志代码
普通图书	M	会议录	C
汇编	G	报纸	N
期刊	J	学位论文	D
报告	R	标准	S
专利	P	数据库	DB
计算机程序	CP	电子公告	EB

表 2-3 电子文献载体和标志代码

载体类型	标志代码	载体类型	标志代码
磁带（magnetic tape）	MT	磁盘（disk）	DK
光盘（CD-ROM）	CD	联机网络（online）	OL

样例：

[1] 刘国钧,郑如斯．中国书的故事［M］．北京：中国青年出版社，1979：110-115．

[2] 昂温 G． 外国出版史[M]. 陈生铮译．北京：中国书籍出版社，1988．

[3] 辛希孟．信息技术与信息服务国际研讨会论文集：A 集［C］．北京:中国社会科学出版社,1979．

[4] 冯西桥．核反应堆压力容器的 LBB 分析［R］．北京：核能技术设计研究院，1997．

[5] 张和生．地质力学系统理论［D］．太原：太原理工大学，1998．

[6] 全国文献工作标准化技术委员会第七分委员会．GB/T 5795-1986．中国标准书号［S］．北京:中国标准出版社，1986．

[7] 罗云．安全科学理论体系的发展及趋势探讨[M]//白春华，何学秋，吴宗之．21世纪安全科学与技术的发展趋势．北京：科学出版社，2000：1-5．

[8] 钟文发．非线性规划在可燃毒物配置中的应用[C]//赵玮．运筹学的理论与应用：中国运筹学会第五届大会论文集．西安：西安电子科技大学出版社，1996：468－471．

[9] 高义民，张凤华，邢建东等．颗粒增强不锈钢基复合材料冲蚀磨损性能研究[J]． 西安交通大学学报，2001，35(7)：727-730．

[10] Papworth A, Fox P, Zeng GT, et al. Ability of aluminum alloy to wet alumina fibres by addition of bismuth[J]. Mater Sci & Technol,1999,15(4):419-428.

[11] 丁文祥．数字革命与竞争国际化[N]．中国青年报，2000－11－20(15)．

[12] 姜锡洲．3一种温热外敷药制备方案：中国，881056078［P］．1989-07-26．

[13] Koseki A,Momose H,Kawahito M,et al Complier:US,828402［P/OL］2002-05-25 [2002-05-28].http://FF&p.

[14] Online Computer Library Center, Inc. History of OCLC[EB/OL].[2000-01-08]. http://www. clc.org/about/history/default.htm.

[15] 江向东．互联网环境下的信息处理与图书管理系统解决方案[J/OL]．情报学报,1999,18(2):4[2000-01-18].http://www.chinainfo.gov.cn/periodical/qbxb．

[16] Scitor C. Project scheduler[CP/DK].Sunnyvale,Calif.:Scitor Corp, 1983.

[17] Metcalf SW. The Tort Hall air emission study[C/OL]//The International Congress on Hazardous Waste, Marquis Hotel, Atlanta,Georgia,June 5-8,1995: impact on human and ecological health[1998-09-22]. http://atsdrl.atsdr.cdc.gov:8080/cong95. html.

参考文献里面标点符号：英文文献用半角,中文文献用全角。

附　录

附录编号依次编为附录 A，附录 B。附录标题各占一行，按一级标题编排。每一个附录一般应另起一页编排，如果有多个较短的附录，也可接排。附录中的图表公式另行编排序号，与正文分开，编号前加“附录 A-”字样。