1. 选题的意义

近年来，随着云计算、大数据产业的快速发展，衍生除一系列网络应用，给网络架构带来了巨大挑战。全新的网络应用对网络设备的灵活性与可扩展性提出了更高的要求。针对此类问题，斯坦福的学者们提出了SDN(software-defined network)的概念及模型，旨在打破传统网络设备的封闭性，加大网络的开放性。SDN网络架构将网络设备上的控制权分离出来，由集中的控制器管理，提出了控制平面和转发平面的概念。

在SDN网络中，由于控制平面和转发平面的分离，控制器成为了整个网络处理数据的核心装置。这种将控制逻辑集中起来的优势虽然明显，但是在大规模的网络部署环境中，由于控制器需要集中管理成千上万的交换设备，而单台设备的控制能力有限，在负载压力过重的情况下，网络设备的寻路请求将无法得到及时的响应。因此，正是这种控制层集中化的设计直接限制了SDN网络的大规模部署。

为了解决SDN控制平台的扩展性问题，业界提出了逻辑上集中、物理上分布的多SDN控制器架构，但是该架构中SDN控制器和SDN交换机之间是静态的映射关系，无法动态适应网络中流量的变化，造成了控制器负载不平衡。为了有效解决多SDN控制器部署方案中控制器之间负载不平衡的问题，需要实时监控和共享SDN控制器之间的负载情况，从而设计出一种机制使得在控制器失效或控制器负载过大时能够将SDN交换机无缝迁移到其他正常的或负载较轻的控制器，避免了控制器单点失效或者负载过大。

2. 国内外研究现状

Brandon Heller等人首次提出迁移交换机的思想，以交换机和控制器之间的时延的时延作为迁移度量，研究了不同控制器部署位置对于交换机迁移的影响，但只是进行了理论性分析，未作算法上的具体描述。Dixit等人提出了一种弹性分布式控制器架构ElastiCon，增加了负载适应决策模块，基于双门限值，当控制器负载超过上、下门限值时，通过交换机的动态迁移，实现了控制器和交换机之间的动态配置，提高了控制器的资源利用率，不足之处在于未考虑多控制器同时超载的情况。Barim等人注重于网络中的流量变化，并设定相应的流量监视模块，根据交换机的实时流请求速率对交换机进行动态调整。KOPONEN等人提出了一种两层控制器架构；在底层，控制器群并不进行互相通信，对网络信息也毫不知情；上层采用一个逻辑上集中的控制器来维持全局网络状态。两层间的控制器需要不断进行通信，整个网络的开销较大。Eugen等人提出了多控制器部署的优化方案，重点在于控制器负载均衡和控制器间时延等因素，权衡了多个相互矛盾的优化目标，但是对于迁移交换机的数量和位置未作要求，容易造成多交换机迁移冲突问题。在Guang等人利用率最低迁移策略中，提出选取资源利用率最低的控制器作为交换机的迁移目标，虽然提高了迁移效率，但目标控制器的搜索构成较为复杂，控制器消耗了大量资源进行通信交互，降低了控制器的流处理性能。综上所述，在分布式控制器的环境下，为了动态实现控制器的负载均衡，需要设计一种代价较小的交换机迁移方案。

3. 选题所要设计、研究的内容及可行性论证

3.1 设计的内容

3.1.1 SDN网络环境搭建

本次设计主要针对多控制器的SDN网络环境，在opendaylight开源分布式控制器的基础上进行交换机迁移方案的模块开发，使用mininet网络仿真器模拟SDN网络。因此，多控制器的网络环境是本次设计的内容和基本开发、应用场景。

3.1.2 交换机迁移策略的设计

为了使开发的迁移机制无缝衔接源网络控制器系统，尽可能使该功能与系统解耦，本方案将交换机迁移策略分为四个模块，分别是控制器评估、交换机评估、迁移决策、迁移触发。

- 控制器评估模块

  实时跟踪控制器的负载状况，监测交换机的流处理状态。

- 交换机评估模块

  对网络中各个交换机进行分析，当检测到迁移需求时，就根据交换机的负载状态给出迁移对象。

- 迁移决策模块

  决策模块综合网络域和控制层的信息，制定具体的迁移策略，并触发迁移执行模块。

- 迁移执行模块

  执行相应的交换机迁移动作。

3.2 可行性论证

3.2.1 技术可行性

目前，针对云计算、大数据等相关新兴应用的出现，业界涌现了多种分布式控制器的解决方案，如Hyperflow、ONIX、ONOS等项目，多控制器的SDN网络搭建技术已趋于成熟。本次设计采用的Open-DayLight控制器遵循OSGI标准，并提供了二次开发的java接口和相关文档，极大降低了开发的难度。

3.2.2 经济可行性

本次设计基于开源控制器进行上层开发，能在各网站上查找相关资料，无需支付任何费用。

4. 主要关键技术、工艺参数和理论依据

4.1 主要关键技术

4.1.1 Mininet网络模拟器

Mininet是斯坦福大学的研究人员基于进程虚拟化技术开发的一款网络仿真平台。利用这种进程虚拟化技术，Mininet 仿真平台可以短时间内子在同一台电脑上模拟一个完整的网络主机、链接和交换机，便于开发测试和演示。Mininet对最新的 OpenFlow 协议提供完整支持，可以创建出 OpenFlow 控制器、交换机和主机以及它们相互连接而成的网络。通过使用 Mininet 仿真平台，可以及其方便的进行网络应用测试，验证应用的有效性，缩短开发周期，还可以将仿真平台中的代码迁移到真实的网络部署中。Mininet 的主要特性如下：

- 方便快捷的创建控制器、交换机和主机及其一起组成的各种网络
- 支持 Open Flow 协议规范
- 提供 CLI 接口，及其方便的对各种测试参数进行配置
- 提供可扩展的 Python API，支持自定义网络拓扑
- 内置多种网络测试软件，如 Wireshark 抓包软件和窗口监测软件等

4.1.2 OpenDayLight控制器

OpenDaylight(ODL)是一个模块化的开放平台，用于定制和自动化任意规模的网络。OpenDaylight重点关注网络可编程性，可解决现有网络环境中的各种使用场景。ODL拥有一套模块化、可插拔灵活地控制平台作为核心，这个控制平台基于Java开发，理论上可以运行在任何支持Java的平台上。ODL控制器采用OSGI框架，SGI框架是面向Java的动态模型系统，它实现了一个优雅、完整和动态的组件模型，应用程序（Bundle）无需重新引导可以被远程安装、启动、升级和卸载，通过OSGI捆绑可以灵活地加载代码与功能，实现功能隔离，解决了功能模块可扩展问题，同时方便功能模块的加载与协同工作。ODL控制平台引入了SAL，SAL北向连接功能模块，以插件的形式为之提供底

层设备服务，南向连接多种协议，屏蔽不同协议的差异性，为上层功能模块提供一致性服务，使得上层模块与下层模块之间的调用相互隔离。SAL可自动适配底层不同设备，使开发者专注于业务应用的开发。

4.1.3 Openflow协议

OpenFlow协议作为应用最广泛的南向接口协议，目前正在不断地演进当中。OpenFlow协议一直由ONF组织负责维护和发布，目前最新的版本为OpenFlow v1.5。OpenFlow v1.0是最早发布的版本，拥有安全通道、单流表匹配模式和一些简单的动作指令，基本上满足了SDN数控分离的要求。针对OpenFlow v1.0存在着匹配模式过于简单无法应对复杂的网络需求这一不足，OpenFlow v1.1设计了"流水线+组表"模式。Openflow v1.2版本针对多控制器架构设计了与多控制器相关的数据处理方式，同时为适应IPv6网络添加了对IPv6基本头的支持，另外对匹配域和端口概念进行了修改。OpenFlow v 1.3是所有版本中比较重要和稳定的一个版本，它进一步扩展了对IPv6基本头的支持，同时重新定义了流的计量方式以测量、控制流的速率。Openflow v1.4增加了流表同步机制，设置流表项的重要级并规定了当流表满载时允许增加新的流表项或者删除重要级别低的流表项。Openflow v1.4还加强了对多控制器的支持，制定了一个交换机可以同时连接多个控制器的规则。对于交换机来说只能拥有一个Master控制器，但是可以有多个Equal控制器或者Salve控制器。Openflow v1.5不仅新添加了出口表功能，还增强了流水线处理等功能。

4.2 理论依据

Cheng等人将交换机迁移问题命名为SMP（Switch Migration Problem），并针对SMP提出了一种负载均衡算法DHA（Distributed Hopping Algorithm）。DHA首先描述了交换机与控制器之间的关系，并将SMP定义为网络效用最大化问题（Network Utility Maximization，NUM），在此基础上设计了负载均衡优化梭型，包括目标函数和约束条件。目标函数旨在寻求最大化的网络效用，约束条件包括；(1)控制器负载不能够超出其负载闽值；(2)每个交换机设备都设置了一个负载上限，避免出现小部分交换机就能消耗掉控制器资源的现象；(3)每个控制器的管理范围不能重叠，即一个交换机同一时间只能受一个控制器的管理。

在此模型下，控制器能承担的负载压力越大其网络效用越高。DHA在每个控制器上部署负载估测模块、DHA决策模块和分布式数据存储模块。在负载估测模块中设置了负载上限和负载下限两个重要的阀值，指示何时启动DHA决策模块。分布式存储模块存储了数据平面内所有的交换机信息，当然也包括负载估测模块和DHA决策模块中的相关函数值。在两个真实的网络拓化上进行了实验验证，实验结果表明DHA在流表建立时间、通信开销和资源平均利用率方面有明显的优势。

陈飞宇等人提出了一种交换机动态迁移算法DSMA，该算法把交换机与控制器之间的映射关系建模成0-1规划问题Psi。Openflow协议1.4版本中规范了交换机与控制器的连接关系，规定一个交换机同一时间只能连接一个Master控制器，但是可同时连接多个Equal控制器或者Salve控制器。由此可知交换化与Master控制器之间确实存在0-1关系，如果某个控制器是交换机的Master控制器，那么这个控制器和交换机之间的关系为1；反之，若某个控制器是交换机的Equal控制器或者Salve控制器，那么这个控制器和交换机之间的关系为0。DSMA基于这个0-1关系利用单目标优化免疫粒子群算法首先得到满足控制平面负载均衡要求的一组解，然后考虑控制器与交换机么间的通信开销，在送一组解中选取最合适的交换机控制器部署方案。

5. 设计的研究特色和创新之处

5.1 设计的研究特色

本次设计主要研究数据中心级SDN网络在多控制器下负载均衡策略的应用，即基于交换机迁移实现控制器之间的负载均衡，为了设计和实现更好的解决方案，本设计着重于以下几个方面的研究：

- 基于Openflow的SDN技术研究

首先，通过对计算机网络发展相关文献的研究，了解到传统网络 TCP/IP 架构的缺陷，特别是目前网络业务需求陡增的情况下，导致网络管理的难度越来越大。在这种背景下，SDN 概念提出。通过对 SDN 转发控制分离架构的研究，探讨如何在基于 OpenFlow 的 SDN 架构下解决网络的管理与创新问题。然后，通过对 Open Flow 相关技术的研究，熟悉 OpenFlow 交换机的工作原理和 OpenFlow协议，分析 OpenFlow 交换机的流表结构和匹配规则，理清 OpenFlow 协议的各个组成部分。最后，对 OpenFlow 网络实现控制转发分离的核心部分—OpenFlow控制器进行研究，分析各种控制器的架构特点和组件系统构成以及对交换机的管理流程。

- 基于多控制器网络环境下交换机迁移算法设计与实现

  通过分析现有的交换机迁移算法，并总结其优点，针对各算法的缺陷和可改进之处，为多控制器环境下解决网络的负载均衡问题提供一种代价较低的解决方案。

- 仿真实验设计

  利用mininet在仿真平台上模拟高负载场景，并捕获实时数据，分析实验结果，验证设计的性能。

5. 2 设计的创新之处

本次设计对数据中心网络的负载均衡技术进行分析，认为目前的多控制器网络中主要影响网络性能的是控制器的负载均衡以及交换机的负载均衡。本文通过控制交换机的迁移实现整个网络流表处理的负载均衡，实时检测交换机与控制器的负载情况并以web方式呈现。本次设计中，我们主要从交换机的全局视图角度研究控制器的负载，建立数学模型，通过网络负载情况的实时数据找出低冲突迁移方案的局部最优解。

6. 主要参考文献

[1]胡延楠. 软件定义网络关键技术及相关问题的研究[D].北京邮电大学,2015.

[2]陈飞宇,汪斌强,孟飞,王雨薇.一种软件定义网络中交换机动态迁移算法[J].计算机应用研究,2016,33(05):1446-1449+1480.

[3]黄小曼,沈苏彬.一种基于集群的SDN控制器负载均衡方案[J].计算机应用与软件,2016,33(06):130-133+162.

[4]刘必果. 基于交换机迁移的SDN控制平面负载均衡研究[D].安徽大学,2017.

[5]胡涛,张建辉,毛明.SDN中基于迁移优化的控制器负载均衡策略[J].计算机应用研究,2018,35(02):559-563.

[6]李婉,沈苏彬,吴振宇.一种基于多SDN控制器的交换机迁移机制[J].计算机技术与发展,2018,28(01):89-94+99.

[7]朱国晖,张瑞,郭嘉.基于SDN控制器的负载均衡策略[J].西安邮电大学学报,2017,22(04):38-42.

[8]赵季红,张彬,王力,曲桦,郑浪.SDN中基于Q-learning的动态交换机迁移算法[J].电视技术,2016,40(06):68-72+110.

[9]童俊峰,闫连山,邢焕来,崔允贺.基于自优化的SDN交换机动态迁移机制[J].计算机系统应用,2017,26(11):82-88.

[10]黄小曼. SDN网络控制器负载均衡技术研究与实现[D].南京邮电大学,2015.

[11]Klamka K , Dachselt R . Elasticcon: Elastic Controllers for Casual Interaction[C]// International Conference on Human-computer Interaction with Mobile Devices & Services. ACM, 2015.

实施方案和时间安排

第 3-4 周： 确定毕业设计题目， 完成开题报告和文献翻译。 第 5 周： 迁移策略的各模块逻辑设计。 第 6-7 周： 算法模块的功能实现与控制器应用集成。 第 8-9 周： 部署分布式控制器环境并准备中期检查报告。 第 10 周： 完成迁移算法性能测试。 第 11 周： 完成论文的初稿。 第 12-13 周： 修改论文形成最终的论文。 第 14-16 周： 编写相关文档， 准备答辩 PPT。