

学校代号 _____
分 类 号 _____
10532
TP391

学 号 _____
密 级 _____
S151000891
普通



湖南大学
HUNAN UNIVERSITY

硕士学位论文

SDN/NFV网络架构可生存性算法研究

学位申请人姓名 _____ 陶恒
培 养 单 位 _____ 信息科学与工程学院
导师姓名及职称 _____ 谢鲲 教授
学 科 专 业 _____ 计算机科学与技术
研 究 方 向 _____ 计算机网络
论文提交日期 _____ 2018年 5月 8日

学校代号: 10532
学 号: S151000891
密 级: 普通

湖南大学硕士学位论文

SDN/NFV网络架构可生存性算法研究

学位申请人姓名: 陶恒
导师姓名及职称: 谢鲲 教授
培养单位: 信息科学与工程学院
专业名称: 计算机科学与技术
论文提交日期: 2018年 5月 8日
论文答辩日期: 2018年 5月 19日
答辩委员会主席: 李智勇 教授

The Research on SDN/NFV Network Architecture Survivable Algorithm

by

Heng TAO

B.E. (Hunan University of Science and Technology) 2015

A thesis submitted in partial satisfaction of the

requirements for the degree of

Master of engineering

in

Computer Science and Technology

in the

Graduate school

of

Hunan University

Supervisor

Professor Kun Xie

April, 2018

湖南大学

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律后果由本人承担。

作者签名： 签字日期： 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权湖南大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于

- 1、保密，在____年解密后适用于本授权书
2、不保密。
(请在以上相应方框内打”√”)

作者签名： 签字日期： 年 月 日
导师签名： 签字日期： 年 月 日

摘要

现如今网络整体规模和网络功能类型复杂度越来越大，5G、VR技术等对网络性能需求不断提高，传统的网络架构已经无法满足按需调动、快速配置等需求，软件定义网络（SDN）的提出就是为了应对现有网络架构无法解决的问题和网络性能瓶颈的限制。利用网络功能虚拟化(NFV)技术将网络资源进行虚拟化，在很大程度上可以解决现有网络无法解决的问题和到达现有网络无法达到的网络性能需求。网络服务运行在实际的物理设备上，这将必然产生不可避免的网络故障，SDN/NFV新型的网络架构在网络的可生存性领域比传统网络架构更加高效灵活的检测、处理和恢复网络故障。

传统分布式网络通过复杂的MPLS协议来实现对源节点和目的节点之间提供多条QoS路径，以提高网络的可生存性，这一做法已不能满足在网络连接出现故障时保持业务高效灵活迅速的提供可生存性保护需求。SDN在光网络和Overlay网络应用中高效实现不相交路径算法，快速的在源节点和目的节点之间寻找满足一定QoS约束的不相交路径（链路不相交,节点不相交或SRLG 不相交）。当主用路径出现故障时，将其承载的业务流转换到备用路径上，从而实现快速的业务恢复。因此，快速不相交路径算法的研究具有很高的实用价值。

SDN作为新型网络架构，网络虚拟化作为未来网络研究的重要领域，两者的结合具有很高的研究意义。在网络虚拟化过程中，不可避开的技术就是虚拟网络嵌入算法以及可生存性保护的研究，嵌入算法主要是考虑虚拟网络节点和链路向底层物理网络的映射，如何为虚拟网络在底层物理网中找到满足条件且最优的嵌入，并且可生存性保护需求可以保证底层物理网资源失效的情况下，保证原本虚拟网络业务正常运行。

本文结合实际项目，基于SDN/NFV网络架构下以可生存性算法作为课题的研究方向，主要研究了以下四个方面的内容：

首先，研究网络可生存性技术，对网络故障失效环境进行了研究，并且探讨了已有各种情形的路径保护算法和不相交路径算法。

其次，光网络和Overlay网络中，考虑共享风险链路组（SRLG）不相交的约束条件，提出一个特殊的边集合，通过创新性的容量设置来获得这个边集合，以这个边集合来分而治之的解决SRLG不相交路由问题，并且与已有算法进行了对比分析，提出的算法优于现有其它算法。

最后，在虚拟网络向底层物理网络映射的过程中，考虑节点和链路映射以及可生存性嵌入算法的设计，在可生存性嵌入算法的设计中，考虑到节点带有特定功能约束条件，本文结合已有算法的不足，设计出星型分割动态规划分配的可生

存性虚拟网络嵌入算法，并且都与已有算法进行了对比分析，提出的算法优于现有其它算法。

关键词：软件定义网络； 网络功能虚拟化； 不相交路径； 虚拟网络嵌入； 可生存性； 风险共享链路组

Abstract

Nowadays, the overall scale of the network and the complexity of the network function type are becoming more and more complex. The demand for network performance is constantly increasing for 5G and VR technology, and the traditional network architecture has been unable to meet the needs of on-demand deployment and rapid configuration. Software defined Network (SDN) is proposed to deal with the existing network architecture can not solve the problems and network performance bottlenecks. Virtualization of network resources by using network function virtualization technology can, to a large extent, solve the problems that cannot be solved by the existing network, and reach the network performance requirements that the existing network cannot achieve. Network services run on actual physical devices, which will inevitably lead to inevitable network failures. SDN / NFV new-style network architecture is more efficient and flexible to detect, deal with and recover network failures than traditional network architecture in the field of network survivability.

Traditional distributed network provides multiple QoS paths between source node and destination node through complex MPLS protocol, which improves the survivability of the network. This approach can no longer meet the requirement of maintaining efficient, flexible and fast survivability protection in the event of network connection failure. SDN can efficiently implement disjoint path algorithm in optical network and Overlay network applications. A disjoint path (link disjoint, node disjoint or SRLG disjoint) that satisfies certain QoS constraints is quickly found between the source node and the destination node. When the main path fails, the service flow is transferred to the backup path to achieve fast service recovery. Therefore, the research of fast disjoint path algorithm has high practical value.

SDN as a new network architecture, network virtualization as an important field of network research in the future, the combination of the two has very high research significance. In the process of network virtualization, the technology that can not be avoided is the virtual network embedding algorithm and the research of survivability protection. The virtual network embedding algorithm mainly considers the virtual network nodes and links mapping to the underlying physical network. how to find the optimal embedding for the virtual network in the underlying physical network, and the survivability protection requirements can guarantee the failure of the underlying physical network resources to ensure the normal operation of the original virtual network service.

In this paper, combining with the actual project, based on the research direction of sur-

vivability algorithm under the SDN/NFV network architecture, the following four aspects are mainly studied:

Firstly, the network survivability technology is studied, and the fault failure environment is studied, and the existing path protection algorithms and disjoint path algorithms are discussed.

Secondly, in optical networks and Overlay networks, a special edge set is proposed considering the disjoint constraint of shared risk link group SRLG, and the edge set is obtained by innovative capacity setting. This edge set is used to solve the SRLG disjoint routing problem, and it is compared with the existing algorithms. The proposed algorithm is superior to other existing algorithms.

Finally, when the virtual network is mapped to the underlying physical network, the design of node and link mapping and survivability embedding algorithm is considered. In the design of survivability embedding algorithm, the condition that nodes have specific functional constraints is considered. Considering the shortcomings of the existing algorithms, this paper designs a survivability virtual network embedding algorithm for star partitioning dynamic programming assignment, and compares it with the existing algorithms. The proposed algorithm is superior to other existing algorithms.

Key Words: Software Defined Network(SDN); Network Function Virtualization(NFV); Disjoint Path; Virtual Network Embedding(VNE); Survivability; Shared Risk Link Group(SRLG)

目 录

学位论文原创性声明和学位论文版权使用授权书	I
摘要	II
Abstract	IV
目录	VI
插图索引	IX
附表索引	XI
第 1 章 绪论	1
1.1 研究背景及意义	1
1.1.1 课题研究背景	1
1.1.2 课题研究目的和意义	4
1.2 相关的研究及发展趋势	6
1.2.1 不相交路径可生存性算法	6
1.2.2 可生存性虚拟网络嵌入算法	7
1.3 论文的主要研究内容和组织结构	8
1.3.1 论文的主要研究内容	8
1.3.2 论文的组织结构与贡献	9
第 2 章 网络可生存性基本原理	11
2.1 网络的可生存性概述	11
2.2 网络故障分类	12
2.3 网络可生存性度量指标	12
2.4 网络故障处理	13
2.5 网络保护策略	14
2.5.1 网络连通	14
2.5.2 网络增强	15
2.5.3 路径保护	15
2.6 小结	16
第 3 章 不相交路径问题	17
3.1 预备知识	17
3.2 不相交路径	18
3.3 可靠性不相交路径	19
3.4 最大不相交路径	21

3.5 域不相交路径	21
3.6 地区不相交路径	22
3.7 不相交路径对	23
3.8 共享风险链路组不相交路径	23
3.9 本章小结	24
第 4 章 共享风险链路组不相交路径对问题	26
4.1 问题描述	26
4.2 原有算法概述	26
4.3 复杂度规模	28
4.4 陷阱问题	29
4.5 分而治之的快速SRLG不相交路径对算法	30
4.5.1 分而治之	30
4.5.2 SRLG冲突链路集合	31
4.5.3 算法步骤	36
4.5.4 算法时间复杂度	36
4.6 实验环境与评价指标	38
4.7 算法性能评估及比较	39
4.8 小结	41
第 5 章 虚拟网络嵌入问题	43
5.1 网络虚拟化概述	43
5.2 网络虚拟化的技术特征	45
5.3 虚拟网络嵌入算法	45
5.4 可生存性虚拟网络嵌入算法	51
5.4.1 共享vs非共享	51
5.4.2 主动保护vs被动恢复	52
5.4.3 算法评价度量	52
第 6 章 物理节点单故障可生存性虚拟网络嵌入问题	55
6.1 问题描述	55
6.1.1 虚拟网络嵌入	55
6.1.2 可生存性虚拟网络嵌入	57
6.2 星型图分解动态规划节点嵌入算法	58
6.2.1 基于星型图的分解	58
6.2.2 构建星型二分图	59
6.2.3 星型二分图问题定义	60
6.2.4 动态规划方法	62

6.3	添加备份资源的略	64
6.4	多节点故障情形	66
6.5	完整算法步骤	66
6.5.1	将增广的备份资源嵌入到底层网络中	67
6.5.2	算法步骤	68
6.6	时间复杂度	68
6.7	仿真配置与评价指标	69
6.7.1	实验环境	69
6.7.2	评价指标	70
6.8	算法性能评估及比较	71
6.8.1	接受率	71
6.8.2	活动节点	71
6.8.3	链路与路径长度	72
6.8.4	成本、收入和成本/收入比	73
6.9	小结	74
第 7 章	总结和展望	75
7.1	本文工作总结	75
7.2	未来研究工作与展望	75
参考文献	77
致 谢	91
附录A	发表论文和参加科研情况说明.....	92

插图索引

图 3.1	Bhandari算法实例.....	20
图 3.2	最大不相交路径的实例.....	21
图 3.3	多域网络实例	22
图 3.4	地区不相交路径的实例.....	22
图 3.5	不相交路径对实例	23
图 3.6	共享风险链接组(SRLG)实例	24
图 4.1	SRLG不相交路径对实例	26
图 4.2	二进制搜索法求最优解实例	29
图 4.3	演示KSP算法效率低下的实例	29
图 4.4	分而治之的解决方案	31
图 4.5	图 G^* 实例	33
图 4.6	最大流最小割定理实例.....	33
图 4.7	图 G^* 的最小割实例	35
图 4.8	路径权重	41
图 4.9	路径跳数	41
图 4.10	运行时间	41
图 4.11	运行时间(无CoSE)	41
图 4.12	核加速比	41
图 4.13	算法加速比	42
图 4.14	算法加速比(无SCLS)	42
图 4.15	效率.....	42
图 5.1	未来互联网的资源配置.....	46
图 5.2	在线VNE算法中重新嵌入VNR	48
图 6.1	虚拟网络请求 $G(V, E)$	55
图 6.2	底层物理网络 $G(S, L)$	56
图 6.3	节点映射和链路映射的实例	57
图 6.4	1+1-保护机制	58
图 6.5	当物理节点 s_1 失效时VirtualStar(v_i)和PhysicalStar(s_j)	61
图 6.6	动态规划方法的演示	63
图 6.7	当物理节点 s_2 失效时VirtualStar(v_i)和PhysicalStar(s_j)	66
图 6.8	当物理节点 s_1 和 s_2 同时失效时VirtualStar(v_i)和PhysicalStar(s_j)	67

图 6.9	增广的备份资源图	68
图 6.10	节点 s_4 失效实例	68
图 6.11	成功的VNE请求数和成功的SVNE请求数.....	71
图 6.12	接受率.....	71
图 6.13	虚拟网络的平均虚拟节点数.....	72
图 6.14	底层物理网络平均启动的节点数	72
图 6.15	底层物理网络启动的节点数与虚拟网络虚拟节点数之比	72
图 6.16	虚拟网络链路数.....	73
图 6.17	物理网络链路数.....	73
图 6.18	物理网络链路数与虚拟网络链路数之比.....	73
图 6.19	底层物理网络消耗资源的平均成本	73
图 6.20	虚拟网络获得需求的平均收益	73

附表索引

表 3.1	不相交路径算法比较	25
表 4.1	SRLG拓扑数据.....	38
表 5.1	虚拟网络嵌入算法分类.....	48
表 5.2	虚拟网络嵌入的度量分类.....	53
表 5.3	可生存性虚拟网络嵌入算法比较	54
表 6.1	SNDlib拓扑数据	69

第1章 绪论

1.1 研究背景及意义

1.1.1 课题研究背景

1.1.1.1 软件定义网络SDN

众所周知，相比发展迅速的计算机产业，网络产业的创新十分缓慢。每一个创新都需要等待数年才能完成技术标准化。为了解决这个问题，SDN创始人Nick McKeown教授对网络产业的创新模式和计算机产业的创新模式进行了对比和研究。在分析了计算机产业的创新模式之后，他总结出支撑计算机产业快速创新的三个主要因素。

1. 计算机工业找到了一个面向计算的通用硬件底层：通用处理器，使得计算机的功能可以通过软件定义的方式来实现。
2. 计算机软件的开源模式，加速了软件开发的进程，催生了大量的开源软件，推动了整个计算机产业的快速发展，Linux开源操作系统就是最好的证明。
3. 计算机功能的软件定义方式带来了更加灵活的编程能力，使得软件应用的种类得到爆炸式的增长。

相比之下，传统的网络设备与上世纪60年代的IBM大型机类似，网络设备硬件、网络应用和操作系统三部分紧耦合在一起组成一个封闭的系统。这三部分相互依赖，通常隶属于同一家网络设备厂商，每一部分的演进和创新都要求其余部分做出同样的升级。这样的架构严重阻碍了网络创新进程的开展。如果网络产业能像当今计算机产业一样，也具备通用硬件底层、开源模式和软件定义功能三要素，一定能获得更快的创新速度，最终像计算机产业一样取得空前的发展。

正是在这种思路的影响下，McKeown教授团队提出了个新的网络体系结构SDN^[1]：在SDN架构中，网络的数据平面与控制平面相分离，数据平面将变得更加通用化，变得与计算机通用硬件底层类似，不再需要具体实现各种网络协议的控制逻辑，而只需要接收控制平面的操作指令并执行即可。网络设备的控制逻辑转而由软件实现的SDN应用和SDN控制器来定义，从而实现网络功能的软件定义化。随着开源SDN开放接口和开源SDN控制器的出现，网络体系结构也拥有了通用底层硬件、开源模式和支持软件定义三个要素。

现代电信网络提供多种高速通信，通过大规模的面向连接和分组交换的网络提供服务运行在光网络，数字用户线路(DSL)，电缆连接甚至无线地面和卫星连

接。因为社会很大程度上依赖于现代电信网络，已经做了很多工作来防止网络故障，例如，通过改善设备环境和材料的物理方面。然而，过去一个世纪的网络故障情形显示，高速骨干网络链路即使短暂的故障也会造成大量的数据包丢弃，严重影响通信质量。根据对Internet服务提供者（ISP）的观察，骨干网链路一年内大约有30%的概率会出现故障^[2]。不论采取了哪些预防性保护措施，网络节点和链接将最终概率性的失灵并停止运作。链路故障是网络中较为普遍的现象。因此，加快故障的恢复速度，降低故障造成的业务丢弃已成为当前研究亟待解决的问题。

在面向连接的网络中，例如波长路由网络中，由于网络节点或链路的故障而造成的网络服务中断通常可以通过从源节点分配至少两个不相交的路径到每个网络连接的目的节点来防止中断^[3]。然后监视从源节点到目标节点的连接状态，当网络连接的主路径故障失效时，可以重新配置连接以使用其备份路径继续保持工作。还可以同时在连接的主路径和备份路径上发送通信信息，因此不需要在主路径故障失效时进行重新配置。虽然找到从源节点到目的节点的一对(最小和)Min-Sum不相交路径是多项式可解的^[4,5]，但由于存在陷阱拓扑^[6]，返回的路径可能远远大于节点之间的最短路径^[6]。另一种方法是找到一对Min-Min不相交的路径，其中主路径的权重最小化，而不是两条路径的权重之和(最小和)，网络业务主要运行在主路径上，当出现故障时备份路径主要是保证业务非中断，而不一定保证原来主路径一样的服务质量，所以对主路径的权重代价尽量小，另外必须保证存在备份路径，但对于备份路径的权重不做严格的需求。然而Min-Min不相交路径问题是NP-hard^[7]。

在分组交换网络中，流量可以沿着主要路径的一部分重新路由，这在面向连接的网络中是不可能的。主路径上的每个中间节点在必要时具有通过另一个链路接口转发数据包的能力。此外，在分组经过故障重路由后，分组被定向到到达目的地的最短剩余路径，可能的方法是跟踪未受故障影响的(初始)主路径的其余部分。然而，配置全网配置需要复杂的转发规则构造，传统的分布式路由协议在计算和内存容量较低的嵌入式系统上难以实现。而新兴的软件定义的网络(SDN)方法可以促进这种网络功能的实现。

1.1.1.2 网络功能虚拟化NFV

多样化的专有网络设备增加了服务提供商的资金和运营费用，同时也引起了网络僵化的问题。网络功能虚拟化（NFV）提出解决了这些问题，实现网络功能的纯软件商品通用的硬件。NFV允许灵活的配置，部署和集中管理虚拟网络功能。结合SDN软件定义NFV架构进一步提供灵活的流量操作和网络功能和资源的联合优化。这种体系结构有利于广泛的应用程序(例如服务链)，并且正在成

为NFV的主要形式。

目前的网络服务依赖于专有设备和不同的网络设备，这些设备是多种多样和专门创建的^[8-10]。这种情况引发了所谓的网络僵化问题，阻碍了业务的增加和网络的升级。为了解决这一问题并减少资本支出和业务支出，虚拟化已成为一种将网络软件功能/应用程序与其支持的硬件分离的方法，并允许网络服务作为软件^[11-13]加以实施。利用虚拟化技术，ETSI工业规范组提出了网络功能虚拟化NFV，虚拟化以前的一些专有专用硬件执行的网络功能^[14,15]，通过将网络功能与底层硬件设备分离，NFV在优化共享的物理基础设施之上提供了基于软件的网络功能的灵活配置。它通过利用低成本商品服务器来解决管理和控制这些封闭和专有设备的运营成本问题。

另一方面，随着软件定义网络SDN的发展，随着网络体系结构^[16-18]的引入，将SDN与NFV(软件定义的NFV体系结构)集成以实现各种网络控制和管理目标的趋势得到了明显的发展。当将SDN应用于NFV时，可以帮助应对动态的挑战，资源管理，智能服务编排和故障恢复。通过SDN/NFV可以动态地为特定类型的服务链创建虚拟服务环境，从而避免了专用硬件和复杂的工作来提供新的服务请求。在使用SDN的同时，NFV进一步实现了实时动态功能配置和灵活的业务转发。

为了获得当前和未来云计算的成功，网络虚拟化也是其中的关键，而SDN是网络可编程性的关键，SDN由于其集中控制的特性，它最适合的领域应该是数据中心中网络虚拟化的应用。SDN要求集中控制，网络虚拟化也同样要求集中化控制，网络虚拟化可以为数据中心的每个“租户”提供自己的网络拓扑并控制其流量，这种特性使得两者很自然地结合到一起了。SDN可以在控制器应用和交换机转发表之间提供标准接口，因此，是网络虚拟化的自然平台。然而，支持具有不同拓扑和控制器应用的许多租户提出了可扩展的挑战性，所以SDN虚拟网络的概念应运而生，其中，虚拟网络服务是SDN虚拟网络的一部分，也是部署防火墙、均衡负载、网络路由和虚拟专用网(Virtual private network, VPN)等按需求提供功能的应用软件。在软件中，虚拟化网络服务消除了对昂贵的物理、专有和固定网络的需要，这些设备都缺乏数据中心所需的灵活性和可扩展性。并且，现在的云计算技术大大减缓了创建新网络服务的压力，同时，诸如网络创新的全球实验环境可以让研究人员在共享基础设施的分片上进行大规模的实验，让租户共享物理资源，虚拟化技术势必成为了这些基础架构中的关键。

网络虚拟化^[13]技术通过对物理网络资源进行抽象、隔离和分配，可支持多个虚拟网络(Virtual Network, VN)共存于同一物理网络中。虚拟网络映射^[19](Virtual Network Embedding, VNE)是实现网络虚拟化的关键，旨在研究如何合理地将租户定制的虚拟网络请求(Virtual Network Request, VNR)映射到底层物理网络(Substrate Network, SN)中，获取满足虚拟网络服务所需的物理网络资源，

从而提高虚拟网络请求接受率和网络资源利用率。对于不同的虚拟网络和物理网拓扑，设计不同的映射算法，会得到不同效率的映射方案，如何映射以确保网络资源的高效利用以及网络的负载均衡，也是现在和未来网络研究的方向。在虚拟网络映射算法中，默认分配给租户的虚拟网络服务可以一直正常运行，但现实中，由于设备自身或者环境等问题有时会引发物理网络故障，进而影响租户的虚拟网络服务的可生存性。因此，为向租户提供一个高可生存性的虚拟网络服务，一些研究者提出了多种可生存性虚拟网络映射算法^[20]。

可生存性虚拟网络映射主要研究节点故障、链路故障或者同时考虑这两种故障的情况。在物理网络中，节点故障对于虚拟网络可用性的影响更大，所以本文将研究基于物理节点故障的可生存性虚拟网络映射（Survivable Virtual Network Embedding, SVNE)算法。

1.1.2 课题研究目的和意义

共享风险链路组(SRLG)是网络生存性分析中的一个逻辑概念。SRLG是一组共享公共物理资源(电缆、管道、节点或子结构)的网络链路，其故障将导致该组所有链路的故。现如今SDN/NFV网络架构Overlay网络体系结构可分为两层：逻辑层和物理层。物理层包括光纤跨度(例如电缆、管道等)、光交换节点(例如OADM、OXC)和光纤跨度节点(即光纤跨越的位置)。逻辑层由链路和节点(物理层中节点的子集)组成。逻辑层中的链路是物理层中的连接，它可以穿越多个光纤跨度和/或光纤跨越节点。此外，几个这样的链路可能通过相同的光纤跨度和光纤跨度节点。通过同一光纤跨度的多个链路因此构成了SRLG。物理层中的光纤跨度等单一故障可能导致多个链路故障。

为了保护网络的两个节点之间的逻辑连接不受单个SRLG故障的影响，通常为连接分配两条不同的路径。其中一种称为主路径，在正常情况下使用。当活动路径失败时，将使用第二个称为备份路径。由于这两个路径不应该同时失败主路径上的链接不能与备份路径上的链接共享任何公共SRLG。在这种情况下，这两条路径是完全不相交的，简称SRLG不相交，如果有主路径，不存在完全SRLG不相交的备份路径，可以找到一个最大的SRLG不相交的备份路径，它是网络中具有主动路径的公共SRLG最少的路径。同样，这两条路径可以是链路不相交的，也可以是节点不相交的。由于一个网络链路可能属于几个SRLG，所以找到一对SRLG不相交的路径比找到一对链路/节点不相交的路径要复杂得多。

在SDN控制器中，往往需要考虑在多约束下的路由问题^[21]。特别在业务发放的场景中，为了到达抗故障的效果，需要为业务寻找工作与保护两条路径。而且为了保证网络的通信质量，通常需要考虑时延，跳数，以及工作与保护路径间的分离约束。所以多约束下的路由问题对实现网络资源实现高效快速调配和利用具

有重要的意义。

现如今有关SRLG完全不相交路径的研究还有很大的空间，其中对两条路径的限制条件的研究将适应多种实际网络应用场景，特别是限制第一条路径的权重尽量小，对在业务中可以更加的提高业务的QoS服务质量，和当故障发生时必定保证存在另一条备份路径来维持原先业务基本的服务质量。研究**Min-Min SRLG不相交路径问题**是现有网络可生存性算法领域的重要方面。

可生存性虚拟网络嵌入随着存储器和服务器的不断虚拟化，带动了网络规模的增大和网络服务流畅性的提高，从而进一步加大了对自动化、多租户和多路径的求迫切性，也带来了网络虚拟化的需求。虚拟化的主要思想就是要创建一个运行在被抽象的实际物理实体之上的更高层次的抽象。网络虚拟化的出现，网络管理员不仅可以随时随地选择创建网络，还能够任意扩展和收缩已经存在的网络。智能的虚拟化软件在完成这些任务时，位于上层的虚拟网络不需要知道底层物理拓扑或者配置发生了什么变化，上层虚拟网络在向底层虚拟网络映射时，为了达到低成本高收益的效果，需要找到最有效的嵌入算法。

而由于SDN的集中控制性能，相比传统的分布式算法，集中控制器可以更好 地执行最优嵌入算法的计算和提供特殊QoS服务需求，这是因为：

- 可以考虑更多的因素，包括当前带宽的负载情况等
- 有一个更稳定的网络全局视图
- 可以在服务器性能较高的内存和处理器上执行计算，而不是在网络设备能力有限的可用内存和处理器上执行

为了支持具有不同拓扑结构的虚拟网络，需要一种嵌入方法将虚拟网络映射到物理设施，并将物理网故障如链路或交换机故障映射到虚拟网络组件，任何虚拟化解决方案都必须快速执行这些映射操作，以便每个租户可以对其虚拟网络提供实时控制。在将虚拟网络映射到物理网络时，嵌入算法是其中的关键，在将虚拟网络的节点和链路进行映射时，需要找到最佳分配完成资源的配置，现有的嵌入算法已经很多，对于不同的应用场景，很难评判算法性能的好坏，较好的解决办法是针对不同的应用场景设计满足要求的算法，使算法的性能和效率能够优化。另外，在虚拟网络的映射过程中，为了防止底层物理资源出现故障失效的情况，可以从用户层面和网络层面进行可生存性的研究。

研究物理节点单故障可生存性虚拟网络嵌入问题，有利于提高网络的可靠性，容错性，鲁棒性和可生存性，当物理层的网络节点出现故障时，能快速完全的保证原网络服务需求正常运行，这对网络的可生存性研究领域有重要的研究含义。

1.2 相关的研究及发展趋势

1.2.1 不相交路径可生存性算法

网络故障主要表现为链路故障，链路故障恢复策略可分为主动式和被动式两种^[22,23]。被动式策略在网络故障后自适应动态地进行全网资源重分配，但路由重新收敛花费较多的时间而不可接受。因此目前故障快速恢复研究以主动式策略为主，通过提前对网络进行资源规划和预留，使得故障时能迅速切换，如基于备份路径和基于多拓扑^[24]的故障恢复技术。基于备份路径的故障恢复技术提供端到端路径重路由，在全局范围内进行流量分配，易于基于现有协议实现；多拓扑技术需要配置多个拓扑子层，路由存储消耗大。因此，备份路径技术是当前故障恢复领域研究的热点^[25–29]。

故障恢复的本质在于维持故障链路承载流量的传输，因此应从流量持续传输角度解决故障恢复问题。大部分故障恢复工作集中在如何选择可靠备份路径上，且一般利用单一备份路径进行故障恢复。然而当流量超出备份路径可用带宽时，单一备份路径无法满足故障恢复的要求。受到这一启发，文献^[26]将多路径技术引入到故障恢复中，采用多条备份路径共同承担流量，减少了流量的丢弃。在此基础上，文献^[27]考虑了不同故障状况，通过将重路由流量分配给有跳数限制的多条备份路径，在网络投入运行之前就设计好应对各种故障场景的最低容量备份网络。文献^[28]提出一种结合故障恢复与流量工程的网络结构，在多路径故障恢复基础上进行流量工程优化，其目的是进行负载均衡，且假设备份路径可用带宽满足故障恢复需求。文献^[29]提出一种跨层故障恢复模型，考虑了备份链路的可靠性，提升了故障恢复成功率。然而，上述故障恢复算法大都以最大化重路由为目标，未考虑恢复后的流量是否满足用户的需求。但是经由备份路径的重路由流量即使最终传输成功，由于链路过载、时延超时等原因，也无法满足业务的服务质量需求(Quality of Service, QoS)，属于无效流量。虽然文献^[30]提出了一种满足QoS约束的自适应调整的多路径路由，但未考虑故障恢复问题。因此，目前已提出的大部链路故障恢复算法不能很好地确保业务的服务质量。

网络服务质量需求是多约束不同属性的条件，可以对这些条件加权值和归一化统一成单一约束条件，网络大部分工作流量集中主路径上，备份路径是当主路径出现故障时，临时使用来恢复故障的路径，所以对备份路径的服务质量需求并不需要特别苛刻，而应该尽量提高主路径的服务质量需求。而网络故障一般为链路故障，SDN/NFV架构下多租户数据中心Overlay网络和传输层光网络的故已经不是单独一条链路的故障，而是拓扑图逻辑上几条链路同时故障，此时引申到风险链路组的故障。综上，求一对风险链路组不相交满足服务质量需求的路径对问

题是现今网络故障可生存性算法的研究重点。

1.2.2 可生存性虚拟网络嵌入算法

在考虑虚拟网络的嵌入过程中，常见的算法主要由两类：首先第一种是节点映射和链路映射彼此独立进行处理，将虚拟节点映射到物理节点上，然后由于此时已经映射的节点在底层拓扑的位置是已知的，因此接下来的链路映射就可以概述为多商品流问题^[31](Multi-commodity Flow Problem, MCF)，这种算法比较简单，但是由于节点和链路映射是独立进行的，只考虑到局部最优解无法获得全局最优解；第二种是节点和链路映射同时考虑，被称为虚拟网络嵌入协调过程，虽然它们都考虑到了全局情况，但是由于算法本身原理的限制，存在着成本比较高和结果不够精确的缺点。在虚拟网络的嵌入算法中，一般会考虑动态映射和静态映射两种情况，有效的嵌入算法在降低成本，提高资源使用率这一方面就显得尤为重要。

现有基于物理节点故障恢复的可生存性虚拟网络嵌入SVNE算法主要采用主动保护^[32-35] 和被动恢复^[36-38]两类策略。主动保护策略在进行虚拟网络请求映射前为虚拟网络预置备份资源以供后续故障恢复使用，被动恢复策略在进行虚拟网络请求映射前不预置备份资源，在后续故障发生后再使用节点迁移或者现有的物理网络资源进行故障恢复。采用主动保护策略的算法包括FI-EVN^[32]、FD-EVN^[33] LC-SVNE^[35]等，采用被动恢复策略的算法包括NB-SVNE^[38]、MR-SVNE^[37]等。

可生存性虚拟网络嵌入SVNE算法的研究从基于主动保护和被动恢复两个方面提出了相应的故障恢复方法，但是依旧存在下面的一些不足：

- 现有基于传统网络的SVNE算法中的主动保护策略比较固化，在物理网络无法为VNR提供全部所需备份资源的时候，会拒绝该VNR，进而导致虚拟网络请求接受率和网络运营收益偏低，而SVNE算法中的被动保护策略易造成故障恢复效率低和故障恢复成功率抖动大的情况。
- 现有基于SDN网络的SVNE相关算法中采用的也是主动保护策略，也存在主动保护策略的请求接受率和网络运营收益低的问题。虽然算法在此基础上增加了备份资源重分配机制和虚拟网络映射资源来增网络运营收益加和网络资源利用率，但是资源动态调整机制不仅会增加算法的时间复杂度，还易造成虚拟网络抖动，影响虚拟网络服务质量。考虑到算法时间复杂度问题，本文暂不考虑使用资源重映射机制。

综上所述，目前的研究对于虚拟网络的保护策略不够灵活，或多或少都会出现性能短板问题，如主动保护策略中的虚拟网络请求接受率和收益低，被动恢复策略故障恢复效率低，传统网络下的SVNE算法无法对物理资源进行集中管控，需要在物理设备间进行大量的通信，这会带来大量的通信开销，因此提出一个根据

物理网络资源现状灵活实现虚拟网络冗余备份的可生存性虚拟网络映射算法，可用以提高虚拟网络请求接受率、故障恢复效率和成功率，这非常有意义。

因此，为提高算法的故障恢复成功率和虚拟网络请求接受率，使运营商的物理网络能够为租户提供更为可靠的虚拟网络服务，本文提出了星型分解动态规划分配的可生存性虚拟网络映射算法，此算法可以应用在主动保护和被动恢复两种策略的情形，与现有算法的比较，具有更高的可生存性接受率，资源利用率和收益，并且故障恢复效率更高。并且能拓展到应有于地域受限的虚拟网络嵌入问题。

1.3 论文的主要研究内容和组织结构

1.3.1 论文的主要研究内容

SDN作为新型网络架构，NFV作为未来网络研究的重要领域，两者的结合具有新的研究意义。在网络可生存性领域中，对业务主路径提供当网络故障发生时的备份路径是一个重要研究，在网络虚拟化的过程中，不可避开的一个问题就是虚拟网络嵌入算法的研究。

1.3.1.1 Min-Min SRLG 不相交路径对问题

基于图论在寻找SRLG不相交路径时，对陷阱问题提供了见解。具体来说，对于带有陷阱问题的AP，我们观察到AP路径上存在一组链路，任何通过所有这些“问题”链路AP都不能找到一个SRLG不相交的BP。我们称之为SRLG冲突链路集。一旦遇到陷阱问题，建议寻找SRLG冲突链路集，而不是搜索所有可能的替代路径，基于最大流最小割定理^[39]。进一步提出了一种分而治之的min-min SRLG不相交路由算法，将原路由问题划分为多个子问题并行执行，找到可行的AP和BP对。研究的主要内容如下：

- 提出了一种新的方案，通过巧妙地设置链路容量来构造一个新的流图，以便于发现SRLG冲突链路集。
- 提出了一种寻找最小SRLG冲突链路集的算法，这有助于减少搜索替代SRLG- 不相交路径对的复杂性。
- 根据SRLG图的风险共享的特性，将SRLG冲突链路最小查找问题转化为子集覆盖问题，使我们能够应用通用算法在不同的复杂SRLG场景(包括一个或多个SRLG 模式的链路)中寻找最小SRLG冲突链路集。
- 提出了一种新的分而治之算法，该算法能在遇到陷阱问题时将原有的min-min SRLG不相交路由问题划分为多个并行执行子问题。与现有技术相比，这样的解决方案搜索过程可以利用现有的AP搜索结果和并行执行更

快的路径查找。

- 在一个多核CPU平台上进行了广泛的仿真，以评估所提出的算法。仿真结果表明，该算法能够以更快的速度在不同的网络场景中找到最佳解。

1.3.1.2 物理节点单故障可生存性虚拟网络嵌入问题

嵌入算法主要考虑的是虚拟网络节点和链路向底层物理网的节点和链路的映射，考虑如何为虚拟网络在底层物理网中找到满足条件的最优嵌入，并且尽可能地提高网络资源的利用率。鉴于此，本文做了如下研究：

1. 首先，研究SDN虚拟网络技术，对SDN网络虚拟化的环境进行了研究，并且探讨和总结了虚拟网络向底层物理进行嵌入时的不同嵌入算法。
2. 在虚拟网络向底层物理网络映射时，考虑节点和链路的映射以及路由算法的设计，在嵌入算法的设计中，假设节点已经映射并且节点具有功能类型的约束。分别设计算法实现此时提供可生存性的虚拟网络嵌入算法。

本课题将结合位置约束的概念，以虚拟网络请求的接受率、物理网络资源用率和故障恢复率等指标作为算法性能评价指标，提出一个星型分解动态规划分配的可生存性虚拟网络嵌入算法。为了实现课题目标，需要进行以下研究工作：

1. 调研现有的可生存性虚拟网络映射算法，以及基于SDN的虚拟网络映射算法和可生存性虚拟网络映射算法，分析不同算法的优缺点，以及存在的共性问题。
2. 提出一种星型分解动态规划分配的可生存性虚拟网络嵌入算法。
3. 设计和搭建仿真平台，对提出的嵌入算法进行仿真，并实现相关的对比方案。
4. 对实验结果进行分析，评估本文方案和对比方案的性能优劣。

1.3.2 论文的组织结构与贡献

本文依据网络可生存性、不相交路径以及虚拟网络嵌入等方面的研究现状与发展趋势，并结合自身对以上的探索和研究，将本文分为七章，各章主要内容如下：

- 第一章为绪论部分，主要介绍了本课题的研究背景及其意义并且简要地分析了国内外研究现状和发展趋势。
- 第二章介绍了网络可生存性相关原理，概述了网络的常见故障和网络故障恢复的两大机制，接着详细介绍了网络故障保护策略。
- 第三章主要是Min-Min风险共享链路组不相交路由算法的设计与实现，在存在陷阱问题的情况下，我们提出了一种求解Min-Min 风险共享链路组不相交路由问题的有效算法。为了降低搜索复杂性，我们提出了一种分而治

之的解决方案，将原Min-Min风险共享链路组不相交路由问题划分为多个子问题，该子问题基于从AP路径遇到陷阱问题导出的SRLG冲突链路集。我们的算法利用现有的AP搜索结果和并行执行来实现更快的路径查找。我们在一个多核CPU平台上使用拓扑跟踪进行了广泛的仿真，仿真结果表明在比较搜索速度较高的情况下，我的算法的性能优于其它算法。

- 第四章主要是物理网络单节点故障可生存性虚拟网络嵌入算法的设计与实现，当一个虚拟请求已经嵌入和运行在底层物理网络中时，突然一个底层物理节点随机独立的出现故障失效，我提出的星型分解动态规划分配的算法可生存性虚拟网络嵌入算法可以预先分配备用资源来预防故障失效，我们做了多种算法指标度量的仿真，仿真结果表明我的算法的在多数性能上优于其它算法。
- 第五章对现有的工作和研究成果进行了总结，并进一步对其中的某些技术问题探讨了改进方案。

第2章 网络可生存性基本原理

为了增强网络性能、保证网络的健壮性，需要对SDN网络中的许多问题进行优化设计，这些问题主要包括：网络可生存性设计问题，业务恢复问题等。随着SDN网络的兴起，在各个领域涌现出来的网络业务在广泛的增长，网络的可生存性问题已成为SDN网络关注的热点。目前业内提出了许多关键词，比如可靠性、容错性、抗毁性、鲁棒性和可信赖性等，它们的本质都是以网络可生存性的研究为出发点。在本章中我们主要就网络可生存性问题展开讨论。

2.1 网络的可生存性概述

软件定义网络中控制层中心控制器管理每条信道，而当某条信道发生故障时，则必须在短时间内恢复。因此在软件定义网络的网络控制层路由方面，研究路由的可生存性变得尤为重要。

关于网络可生存性(survivability)具体定义有多种说法^[40]。Neumann^[41]等人首先是提出了网络系统领域的可生存性定义：基于计算机通信系统的应用在任意的不利条件下都应具有持续满足保持用户需求的基本能力，其中用户需求包含安全性、可靠性、实时响应和正确性等需求；Ellison^[42]等人进一步研究来完善了网络系统可生存性的定义：当网络系统在遭受故障、攻击和意外事故的情况下都能及时完成任务的能力，属于网络完整性的一部分。

考虑到今天通信系统和基础设施的重要性，网络应该在设计和操作上考虑到系统或者设施出现故障能够被解决。例如，由于恶意攻击，自然灾害，突然间的电缆断裂，计划维修，设备故障等等，网络节点/链接可能出现故障。弹性，容错性，可生存性，可靠性，鲁棒性和可信赖性，是已经被使用的不同术语。这些术语是面对网络故障时网络维护运行保持通信能力的术语。如作者^[40]所指，不同的术语有重叠的意思和一定的歧义。在本文中，我们将使用可生存性网络一词是指在网络中当一个网络组件出现故障，可以通过找到替代路径来避免出现故障的网络组件。

就国内外目前的研究成果来看，网络可生存性设计主要基于两种模式^[43]，即被动式和主动式。简单来说，第一种模式是在故障产生后对系统模型的故障进行处理，目前还处于研究阶段；而第二种模式采用了冗余入侵的策略，它具有高效的监测功能，一旦故障出现，便调用冗余资源来对系统的异常处理。第二种模式的成本较第一种模式要高一些，并且现阶段这方面的研究还不是很成熟，目前比较经典的是主动式模式，它通过对网络故障的快速检测、定位和恢复使其可生存

性提高。

随着人们对可生存性技术的关注，国内外的许多研究学者和机构都开展对可生存性问题的研究，其研究重点可概括为可生存性的基本概念^[3]、可生存性体系结构和系统模型^[44]、可生存性风险评估^[45]等。

现阶段，网络可生存性研究已经有一定的开展，许多知名学者就络可生存性问题的不同方面不同领域作了深入研究，问题涉及故障分类、可生存性问题建模分析和故障恢复技术等许多子领域，并取得了一些成果。

2.2 网络故障分类

由于网络遭受故障的因素及出现故障的网络元素不同，网络故障也是多种多样，按照网络中故障的表现方式不同，将网络故障分为软故障和硬故障；按照故障产生的元素不同可分为信道故障、节点故障和链路故障。概念解释如下：

所谓软故障是指网络信息传输过程中由于信号逐步衰减，而造成信息丢失的事件，如光纤损耗增大等；硬故障是指某些意想不到的随机事件致使传输信道出现中断，如地震、光纤断裂、收发元器件老化失效等。

不难看出，硬故障对网络业务影响较大，但容易检测出来并且处理方便；软故障对网络业务的影响范围比较小，但出现机率极高，不容易被立即发现和处理，并且难以对故障精确定位。

因而，从科研角度出发，网络中的故障按拓扑位置区分是更利于研究：

- 信道故障是由于该信道对应的底层物理设备出现故障而引起。
- 链路故障主要是由于两物理节点间的通信链路（如光纤）断裂而引起。
- 节点故障主要是由于底层物理节点（如路由器）断电或者物理破坏而引起。

2.3 网络可生存性度量指标

对网络进行可生存性设计时，要评价是否达到的最理想的结果或者状态：对于给定的网络拓扑结构，能够在最短的时间内使故障元素获得最大程度的恢复，并同时也要保证最大的资源利用率^[46]。然而由于任何事件都是此消彼长的互斥性，很难同时达到所有的需求标准，所以需要根据不同的业务特性或者用户需求甚至网络本身的特点，采取提高网络生存性的措施，从而满足网络的可生存性指标需求。

常用的网络可生存性指标度量主要有：

- 故障恢复时间：指从网络故障发生时，到故障被处理业务恢复正常传输所需的时间。该项指标是最直接也最能体现网络的可生存性能。众所周知，

对于网络的用户来说，传输过程是完全透明的，他们所能感受到的服务质量就是网络提供的业务服务质量，而其中最敏感的就是故障恢复时间。

- 业务请求接收率：是指被接收的网络业务数与总体业务请求数的比值。可生存性网络研究的目标是最大化请求接收率。
- 网络资源利用率：这是衡量任何一个网络优劣的关键指标，提高网络利用率是网络运营商追求利益降低成本的目标。
- 平均网络负载：网络负载是指网络节点或者链路中的流量负荷，一旦节点或者链路负荷过重，将直接影响整体网络的连通性能。因此，最小化网络负载将有利于网络可生存性的研究。
- 业务平均跳数：某些网络有时以步长跳数来衡量代价成本，因此最小化路径长度是网络优化指标之一。
- 鲁棒性：即健壮性或者可生存性，是指经历过一次网络故障后，网络再次承受故障不受故障影响的能力。它主要用来衡量网络业务的可持续生存性。

2.4 网络故障处理

网络的可生存性实现机制根据是否进行重路由计算和是否预留备用资源，通常把故障处理策略分为保护(protection，主动式)和恢复(restoration，被动式)^[47]。保护措施和恢复措施均是在网络故障情况下，使停止的业务得以重新运行。原则上，两者都要利用重路由方式(重新选择新的路由来代替故障路由)，继续保证故障业务数据的传输。但就具体实施方法而言，保护和恢复方法又有所不同。两种方案各有长短，保护机制可以提供更快的恢复时间，有利于网络服务质量的保障；恢复机制能够提高网络资源的利用效率降低网络资源的消耗。网络恢复策略是在网络发生故障后动态自适应地进行全网资源重分配，但分布式网络路由重新收敛需要花费较多的收敛时间，这样服务质量是不可接受。因此目前故障快速恢复研究以网络保护策略为主。

保护方案是指事先为业务(如虚拟网络请求、服务链请求等)分配好预留的备用资源，主要是通过节点之间预留的备用资源来实现网络保护。当故障发生时，将工作通路上的通信信号切换到备份通路上，使工作信号通过预留的备份通路维持业务正常传输；而恢复方案指，并不事先为网络业务分配预留的备份资源，在检测到故障时，动态地从网络中寻找替代路由，来承载受故障影响业务。如果不可避免的找不到不符合需求的路由，则该工作路径上携带的业务就会丢弃。

保护方案和恢复方案两者不是互斥关系，因此，我们在实际网络的可生存性设计过程中，往往会同时考虑保护和恢复方案折中的方案，取两种方案中的优点合并，从而为客户提供多种服务级别和服务质量，尽力做到在恢复时间、可生存性保障、资源效率及成本收益之间取得平衡。同时保证故障发生的情况下，及时性和网络资源合理化利用，即保证在一定的约束条件下为工作通路预留好事先准备的保护通路，这就是网络可生存性基于约束条件的资源优化问题。

2.5 网络保护策略

在实际的网络操作中，由于网络资源的不足，通常以保护机制为基础，來保障一些可预料的故障，（如光纤断裂等交换机故障），然后再使用恢复策略进行加强，保障整网范围内的服务质量。根据不同需求的业务可以选择不同类型的网络保护策略以保证网络的可生存性，比如对于实时业务，可以使用节点/链路保护，即预先建立预留资源和保护通道的保护方式；而对于“best in effort”的业务，就按需建立保护通道或者依靠高层的恢复机制。保护机制与恢复机制相比，保护机制具有更快速的恢复能力、更高的可靠性和更好的可生存性，这更适用于规模大的传输网络，因此本文主要研究的是保护机制。

其中提高网络可生存性主要的三种方法如下所示。

1. 网络连通，即可生存性好的网络应该是具有图论中高连通性的。
2. 网络增强，即可能需要新的链路/节点以增加网络的连通性。
3. 路径保护，即寻找替代方案的过程，失效故障时的替代路径。

现实网络中网络故障是不可避免的，因此必须不断加强网络可生存性研究，可以采取对网络故障进行快速准确的检测、定位和恢复的方法，来保证网络的可生存性能。

2.5.1 网络连通

网络通常被表示为图 $G = (\mathbb{V}, \mathbb{E})$ ，其中 \mathbb{V} 是 $|\mathbb{V}|$ 个节点的集合（例如表示路由器）， \mathbb{E} 是 $|\mathbb{E}|$ 条链路的集合（例如表示光纤线路或无线电信道）。链路可以用它们的容量、延迟、长度、成本和/或失效概率的权重为特征。如果图中的每对节点之间存在路径则图是连通(connected)，否则该图被称为非连通。在可生存性的含义中，连通性的概念可以进一步拓展为 k -连通(k -connected)，其中在每对节点之间存在至少 k 条不相交路径。根据这些路径是节点不相交还是链路不相交，我们区分为节点连通性和链路连通性。图 G 的边连通度 $\lambda(G)$ 是删除 G 的最小边数使得图边不连通。相应地，图的节点连通性 $\kappa(G)$ 是删除 G 的最小节点数使得图节点不连通。

2.5.2 网络增强

测试出网络的连通度的结果显示出网络可能是不够健壮(连通)。重新布线(覆盖)网络可以提高其健壮性^[48]。因此通常通过向网络添加新的链路和可能的节点来提高网络的性能或网络的健壮性。添加链路或节点可能代价很高(这能通过链路/节点权重来反映)，因此新的链路/节点应该被明智地放置，以便以最少的链路/节点数量获得所需的网络属性，或者添加固定数量的链接/节点使所需的网络属性最大化。这类问题被称为(网络)增强(network augment)问题，然而在这类问题中，问题仅在其目标上有所不同。例如， k -连通性是网络健壮性的一个重要属性，通过添加链路来达到 k -连通性就是这样的目标之一。代数连通性增强是一个NP-hard问题^[49]。类似地，添加最小数量的链路使一个图是弦(chordal)也是NP-hard^[50](如果一个图的四个或更多个节点之间都有一条边相连则称这个图为弦)。

2.5.3 路径保护

在Internet上部署诸如OSPF这样的网络协议，以获得正确的拓扑视图，并在发生变化(如链路故障)时，将路由收敛到新的(不受干扰的情况)。但是这个过程并不快，应用程序在性能上可能仍然面临不可接受的干扰。结合MPLS技术，可以使用MPLS快速重路由机制，它提供了从失效的主路径在亚秒时间切换到备份路径的能力。这个快速重路由机制在RFC 4090^[51]，其被提出并且已经被几个供应商实现了。这一概念也已扩展到纯IP网络，并被称为IP快速重路由^[24]。RFC 4090 定义了RSVP-TE扩展来建立备份标签交换路径(LSP)隧道，用于对LSP隧道进行本地修复。备份路径可以为防止链接或节点故障而配置。由于备份路径是预先计算的，所以在计算备份路径或在发生故障时执行信令时不会浪费时间。因此，迫切需要有效的算法来计算不相交的路径。根据备份路径是在主路径失效之前还是之后计算，可生存性技术可以广泛地分为恢复技术或保护技术。目前，许多文献对于路径保护技术进行了大量的研究，根据不同的方式或功能总结如下：

- 从重路由的角度：基于路径(通道)保护、基于链路的保护及区段保护。
- 根据备用资源的预留方式：共享保护和专用保护。
- 按照路由的计算方式：实时计算和预计算。

2.5.3.1 通道保护

通道(路径)保护是指业务故障恢复由通道两端的终端节点来实现。具体来说，基于通道保护机制是指对工作路由事先预留一条备份路由，在故障发生后，用预留备份通道来传输故障通道中的业务数据流，从而取代原先的故障通道，实现业务的重路由。当发生故障时，其切换过程只涉及源、目的节点，与中间节点无关，

由于是源节点和目的节点启动保护切换，因而对故障的具体定位要求并不高。

通道保护又分为共享通道保护（备用资源能同时为多条工作通道提供保护）和专用通道保护（备用资源为某条工作通道专用），共享保护是指1: N保护方式，专用保护是指1+1通路保护和1: 1通路保护。

2.5.3.2 链路保护

链路保护是指业务请求经过的每一条链路，都有一条备用保护路径对这条链路进行保护，一旦链路出现故障，业务将跳过故障点，直接切换到保护路径上。在与故障点邻接的两点间，为该故障链路寻找一条可不经过该故障点的备用路径。显然，链路保护方案中参与保护切换的节点数较少，因而具有更快的恢复速度，同时由于为每条链路进行保护，资源浪费过高资源利用低。

链路保护方案中，对于不同链路中的业务只要不同时刻发生网络故障，不同链路共享相同的保护路径，因此也分为共享链路保护和专用链路保护两种。前者是指对于某一条链路，提供专门的保护路径，其他链路则不得使用该条链路的专用保护路径；而后者允许不同的链路的保护路径在其重叠的链路或者节点部分实现资源共享，后者比前者资源利用率要高，而前者较后者的抗故障性更好。

2.5.3.3 区段保护

区段保护是对通路保护和链路保护折中的保护机制，考虑了通路保护和链路保护各自的特点而得到的一种保护方式。区段保护是指在一对节点之间出现故障时，对该段链路中的业务切换到这两个节点之间的另一段路径中去。如果两个相邻节点之间发生故障，则类似于链路保护。

2.6 小结

网络可生存性设计的目的是提高网络的健壮性，由于网络故障不可避免，那么故障后的及时修复成为网络性能的一个重要方面，比较得出主动式方案优于被动式保护方案，同时通道保护方案优于链路保护方案。

第3章 不相交路径问题

不相交路径(Disjoint Path)问题可以看作是最短路径问题的一个扩展，是网络故障恢复路径保护方法的主要技术。不相交路径是计算出几条不共享任何公共点/边的路径。提供不相交路径将提高网络连接的可靠性和网络流量，并相应地提高网络的可生存性。网络可生存性被定义为在网络组件(例如，节点/链路)发生故障时提供持续服务的能力^[52]。不相交路径对问题是不相交路径问题的另一个变体。

不相交路径具有广泛的应用领域。例如，对于通信网络中的业务具有多条不相交的路径将提高其传输可靠性。通过在多个不相交路径上并发发送流量，路径的失效不会影响其他路径的性能，并且流量仍将到达其目的地。在交通网络中，预先计算出的不相交路径数将使卡车司机能够按照不同的路径改变路线，而不是总是坚持最短的路径。

本章的其余部分按以下方式组织。在**不相交路径**部分，给出了不相交路径的形式化定义，讨论不相交路径问题的附加条件限制及其相应的时间复杂性，描述了几种有代表性的不相交路径算法。在**基于可靠性的不相交路径**部分，我们将介绍路径可靠性的概念及其与不相交路径的关系。在**最大不相交路径**部分将阐述不相交路径可能部分重叠的情况下而不是完全不相交。在**域不相交路径**章节，我们继续在多域网络中寻找域不相交路径。因为多个链接(或多个节点)可能在共享风险下同时失效，在**共享风险链路组不相交路径**部分介绍共享风险链接组SRLG的概念。确保不相交的路径不会同时失效由于单个链接(或节点)失败。风险也可能影响基于地区的网络，因此**地区不相交路径**部分讨论了几种基于地区的风险模型。与不相交路径问题对应的不相交路径对问题将在**不相交路径对**部分中讨论，将讨论不同情况下的复杂性。最后，我们在最后一节对本章进行了简要的总结。

3.1 预备知识

网络在我们的日常生活中非常普遍。我们的身体是由突触连接的神经元网络组成的。我们的运输网络使我们能够轻松地往返于不同的地方。互联网是我们巨大的信息门户也是世界内的计算机网络。电网提供电力，而如果没有电力那我们现在社会都可能会停止运作。我们的社交网络让我们和朋友与家人联系。由于网络的重要性，网络特性的多样性已经得到了广泛的研究，尤其是在图论领域。

在图论中，网络被看作是一种通过节点互相链接的。节点表示网络的关节点，例如，通信网络中的路由器，海运网络或交通网络中的城市。链路表示将关键点连接在一起的连接器，例如，通信网络中的电缆，海运中的贸易路线，运输网络中的网络或公路。

图论中研究最多的课题之一是最短路径问题，即在网络中的两个节点，使得路径上链路权重之和最小化。传统的最短路径算法是Dijkstra算法^[53]以及Bellman-Ford算法^[39,54]。使用最短路径在一个通信网络的两个路由器间信号可以在最小延迟之间交换，货物可以在海运网络中两个港口之间的以燃油成本最低的代价发送，而且在运输网络中我们可以更快地往返于城市之间。

网络通常表示为图 $G(V, E)$ ，其中 V 是 $|V|$ 个节点的集合(例如，节点表示路由)和 E 是 $|E|$ 条链路的集合(例如，链路代表光纤线路或无线电信道)。链接可能带有延迟、长度或成本等属性。对于每条链路 e_i ， w_{e_i} 表示链路的权重。路径 P 的权重表示为路径 P 中每条链路的权重之和 $w_P = \sum_{e_i \in P} w_{e_i}$ 。

3.2 不相交路径

不相交路径问题被定义成如下：

定义 3.1(不相交路径问题) 给定 $|V|$ 个节点集 V 和 $|E|$ 条加权链路集 E 组成的有向网络 $G(V, E)$ ，两个特殊节点 $s, d \in V$ 。给定一个整数 $k > 0$ ，求 s 到 d 的 k 条路径 P_1, P_2, \dots, P_k ，使路径间不共享任何公共链路(或节点)。

对不相交的路径添加附属目标条件，例如：

- 最小-最大(Min-Max)不相交路径问题-所有链路权重之和最大的路径最小化。
- 最小-最小(Min-Min)不相交路径问题-所有链路权重之和最小的路径最小化。
- 有界(Bounded)不相交路径问题-每条路径的所有链路权重之和应小于给定权值 Δ 。
- 最小和(Min-Sum)不相交路径问题- k 条路径的所有链路权重之和的总和最小化。

算法3.2.1给出了求解不相交路径问题的一种简单的启发式算法，称为迭代DP算法。迭代DP算法基于 k 个连续最短路径的计算。

Bhandari^[55]提出了一种不受到陷阱拓扑影响的Min-Sum不相交路径算法。虽然有一种称为Suurballe算法^[4]也可以绕过陷阱拓扑问题，但我们关注Bhandari算法，因为它更简单。算法3.2.2给出了Bhandari算法的伪码，我们演示了使用Bhandari算法在图3.1 (a)所示的网络查找 $k = 3$ 条链路不相交的路径(从节点1到节

算法 3.2.1 迭代DP算法

- 1: **for** $i \leftarrow 1$ to M **do**
 - 2: 寻找从节点 s 到节点 d 的最短路径 P_i
 - 3: 从 G 中删除 P_i 的中间节点链路
 - 4: **end for**
-

点5)。从节点1到节点5的第一条初步最短路径是 $P_1 = 1 - 2 - 3 - 4 - 5$ (权重4)。 P_1 的组成链路反转其链路权重取负如图3.1 (b)所示。然后，计算的另一条初步最短路径 $P_2 = 1 - 4 - 3 - 5$ (权重8)，如图3.1 (c)所示。为了得到两条链接不相交的路径 P_1 和 P_2 ，不包括重叠链路，在计算另一条初始路径 $P_3 = 1 - 3 - 2 - 6$ (权重9)之前，第3-5行再次重复，如图3.1 (d) 所示。再次，将重叠链路排除在外，以获得最后三条链路不相交的路径，如图3.1 (e)所示。

算法 3.2.2 Bhandari(G, s, t, k)

- 1: 寻找从节点 s 到节点 d 的最短路径 P_1
 - 2: **for** $i \leftarrow 2$ to M **do**
 - 3: 对于节点不相交的路径，拆分所有 P_x 的中间节点，其中 $x < i$
 - 4: 用反向边替换在原图中所有 P_x ($x < i$)路径的每条链路
 - 5: 寻找从节点 s 到节点 d 的最短路径 P_i
 - 6: 删除所有重叠链接，以获得不相交的路径 P_x ，其中 $x \leq i$
 - 7: **end for**
-

3.3 可靠性不相交路径

路径可靠度是指路径在未来随机时间内处于运行状态的概率^[56]。路径 P 的可靠度(A)可以通过乘以其所有组成链路的可靠度来计算：

$$A = \prod_{y \in P} a_y \quad (3.1)$$

其中 a_y 是链路 y 的可靠度，它依赖于链路的平均故障间隔时间(MTBF)和平均修复时间(MTTR)。

$$a = \frac{MTBF}{MTBF + MTTR} \quad (3.2)$$

$$MTBF = \frac{\text{total operating time}}{\text{number of failures}} = \frac{1}{\text{failure rate}} \quad (3.3)$$

$$MTTR = \text{failure localization time} + \text{failure repair time} \quad (3.4)$$

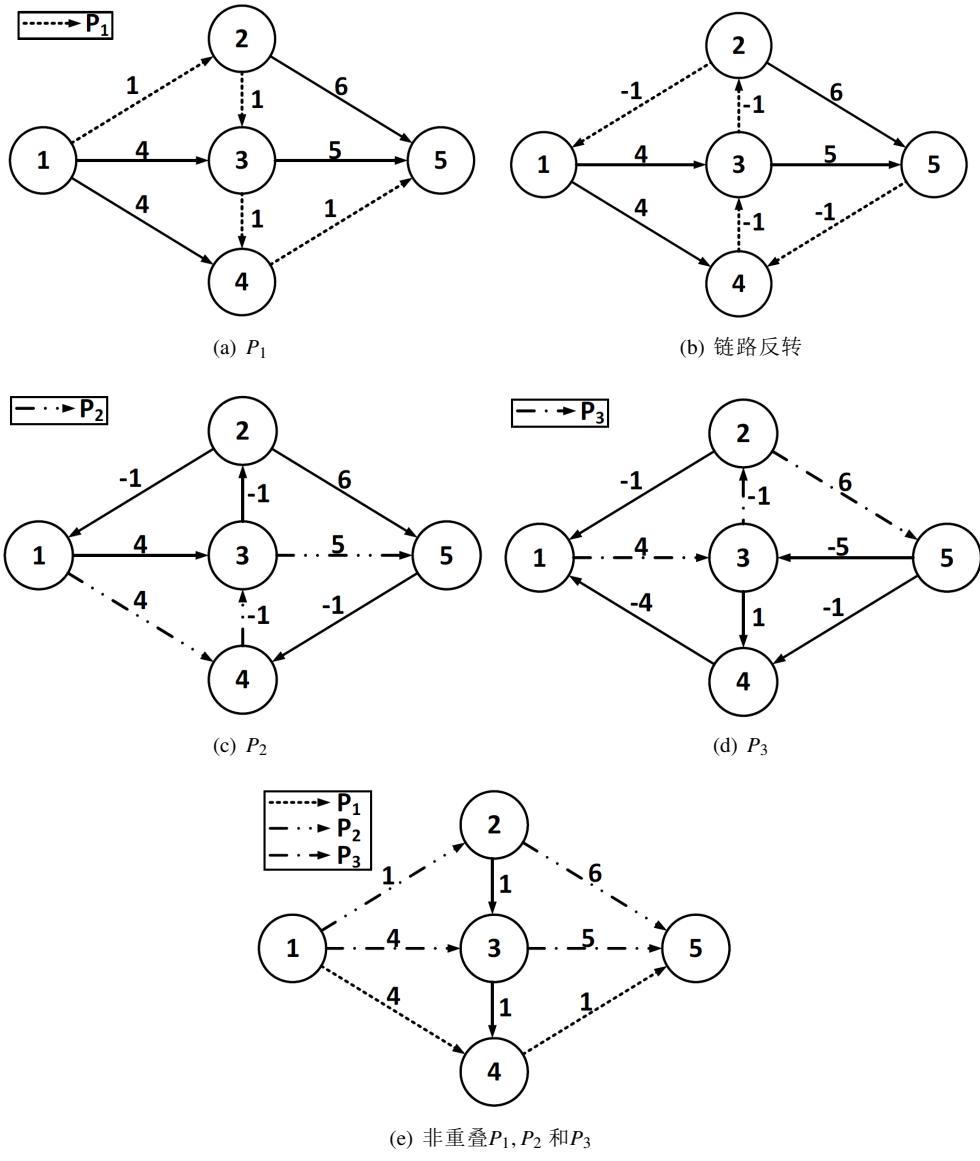


图 3.1 Bhandari 算法实例

k 条不相交路径的总可靠度可以计算为:

$$A_t = 1 - \prod_k (1 - A_k) \quad (3.5)$$

可靠性不相交路径问题被定义成如下:

定义 3.2(可靠性不相交路径) 给定 $|V|$ 个节点集 V 和 $|E|$ 条加权链路集 E 组成的有向网络 $G(V, E)$, 两个特殊节点 $s, d \in V$, 给定一个整数 $k > 0$ 和小数 δ , 每条链路的权重即为每条链路的可靠度, 求 s 到 d 的 k 条路径 P_1, P_2, \dots, P_k , 使路径间不共享任何公共链接(或节点)并且 k 条路径总的可靠度 A_t 至少为 δ 。

3.4 最大不相交路径

最大不相交路径问题被定义成如下:

定义 3.3(最大不相交路径问题) 给定 $|V|$ 个节点集 V 和 $|E|$ 条加权链路集 E 组成的有向网络 $G(V, E)$, 两个特殊节点 $s, d \in V$, 给定一个整数 $k > 0$, 求 s 到 d 的 k 条路径 P_1, P_2, \dots, P_k , 使路径间共享最少的公共链接(或节点)。

如果不存在完全不相交的路径, 则最大不相交路径是有用的。如果两个路径所共有的(节点)链路的数目最小, 则一对路径是最大不相交的。如图3.2所示, 因为不存在完全不相交的路径对, 从节点1到节点4的路径 P_1 和 P_2 都使用链路(3,4)。然而, 如果共享节点或链路失效, 最大不相交路径仍然容易出现同步路径失效。例如, 如果链接(3,4)失效, 则 P_1 和 P_2 路径都将失效。

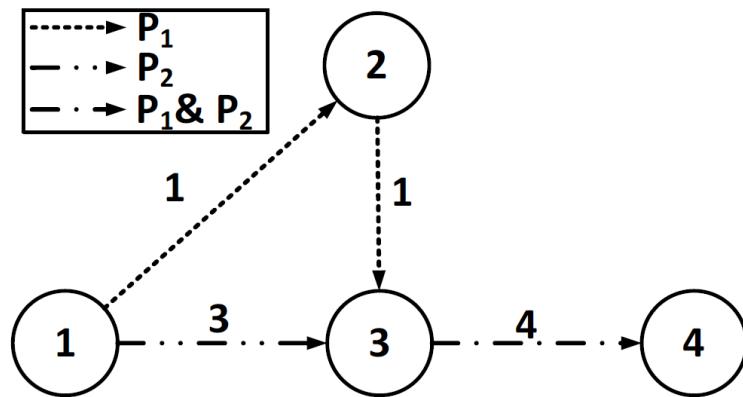


图 3.2 最大不相交路径的实例

3.5 域不相交路径

域不相交路径问题被定义成如下:

定义 3.4(域不相交路径问题) 给定 $|V|$ 个节点集 V , $|E|$ 条加权链路集 E , $|D|$ 个域集 D , $|L|$ 条域内加权链路集 L 组成的有向网络 $G(V, E, D, L)$, 两个特殊节点 $s, d \in V$ 。

每个域 $d \in \mathbb{D}$ 由一组点集 $\mathbb{V}_d \subseteq \mathbb{V}$ 和一组由点集 \mathbb{V}_d 构成的链路集 $\mathbb{E}_d \subseteq \mathbb{E}$ 构成。给定一个整数 $k > 0$, 求 s 到 d 的 k 条路径 P_1, P_2, \dots, P_k , 使路径间不共享任何公共域。

域是由一组节点和链路组成。图3.3所示了一个具有四个域的网络, 这些域通过域间链路相互连接。域可以表示光网络中的行政区域、交通网络中城市的一个省等。

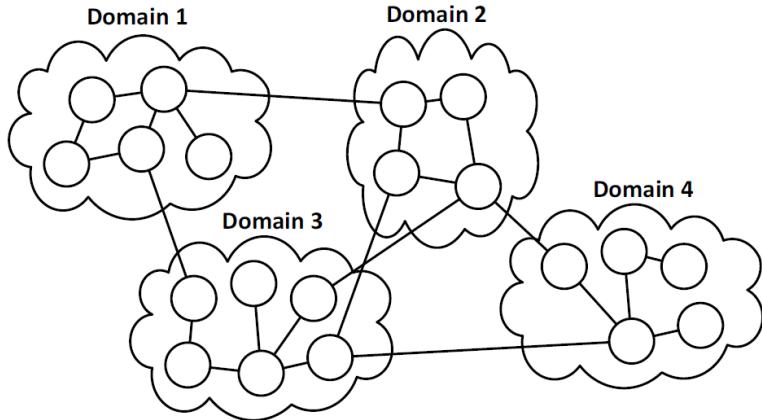


图 3.3 多域网络实例

3.6 地区不相交路径

地区不相交路径问题被定义成如下:

定义 3.5(地区不相交路径问题) 给定 $|\mathbb{V}|$ 个节点集 \mathbb{V} 和 $|\mathbb{E}|$ 条加权链路集 \mathbb{E} 组成的有向网络 $G(\mathbb{V}, \mathbb{E})$, 并且这些点和边被嵌入到二维平面中, 两个特殊节点 $s, d \in \mathbb{V}$ 和一个直径 $D > 0$ 。给定一个整数 $k > 0$, 求 s 到 d 的 k 条路径 P_1, P_2, \dots, P_k , 使路径间不会受到直径 D 的单一地区失效的影响(除了在点 s, d 中的失效)。

例如, 如图3.4所示的网络链路的权重代表其相邻节点之间公里距离。考虑当 $D = 15km$ 时从节点1到节点6找一对地区不相交路径的问题。由于链路(2,4)和(3,5)之间的距离小于D, 路径 P_1 和 P_2 不是该问题的可行解决方案。

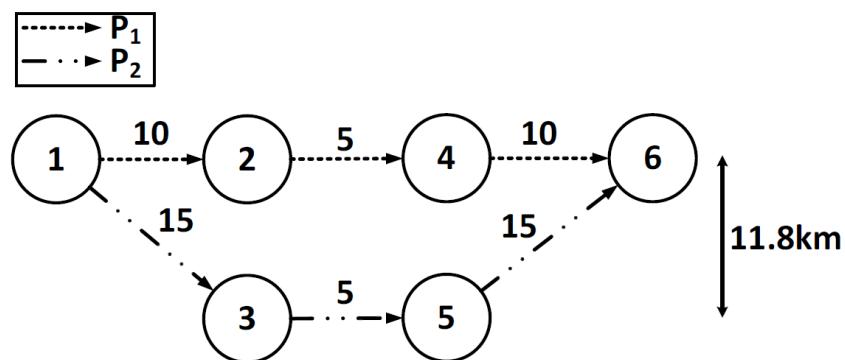


图 3.4 地区不相交路径的实例

3.7 不相交路径对

不相交路径对问题被定义成如下：

定义 3.6(不相交路径对) 给定 $|V|$ 个节点集 V 和 $|E|$ 条加权链路集 E 组成的有向网络 $G(V, E)$, k 对特殊节点对 $s_k, d_k \in V$ 。给定一个整数 $k > 0$, 求 k 条路径 P_1, P_2, \dots, P_k 分别从其对应的源节点 s_k 到其对应的目的节点 d_k , 使路径间不共享任何公共链接(或节点)。

例如, 图3.5所示的不相交路径对 P_1, P_2, P_3 和 P_4 都是不相交的。

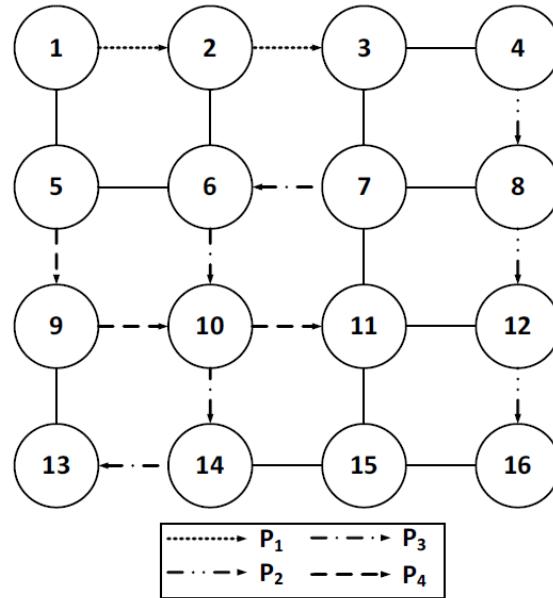


图 3.5 不相交路径对实例

3.8 共享风险链路组不相交路径

共享风险链接组(SRLG)是一组链路共享同一个组件, 该组件的故障会导致所有链路的故障。一条链接可以属于多个共享风险链接组里。举个例子, 在光网络的导管中^[57]可以放置多条光链路, 如图3.6 所示, 链路(1,2),(3,2) 和(3,4) 放置在一个导管中, 同时链路(3,2)和(3,4)也共同放置在另一个导管中。如果某个导管被切断, 相应导管的链路将失效。每个导管对应一个共享风险链接组。其它共享风险链接组的应用是交通网络的相关拥塞和电网的级联故障^[58]。

设 \mathbb{R} 为网络中的风险集(故障)。每个风险可能对应于导管断开、光纤断裂、在一个节点上驱动故障、软件故障或这些因素的任何组合。对每一个共享风险链路组 $r_i \in \mathbb{R}$ 是指与其风险 r_i 相关的链路集合 \mathbb{R}_{r_i} , $1 \leq i \leq \chi$ 和 $\chi = |\mathbb{R}|$ 是共享风险链路组集合的个数。如图4.1(a) 所示, 该图包含五个共享风险链路组集合 $\mathbb{R}_{r_1} = \{e_1, e_9\}$, $\mathbb{R}_{r_2} = \{e_2, e_3, e_{19}\}$, $\mathbb{R}_{r_3} = \{e_2, e_4, e_{11}, e_{17}\}$, $\mathbb{R}_{r_4} = \{e_5, e_{13}\}$, $\mathbb{R}_{r_5} = \{e_{15}, e_{18}\}$ 。在这个例子中, 链路 e_2 同属两个共享风险链路组集合里 \mathbb{R}_{r_2} 和 \mathbb{R}_{r_3}

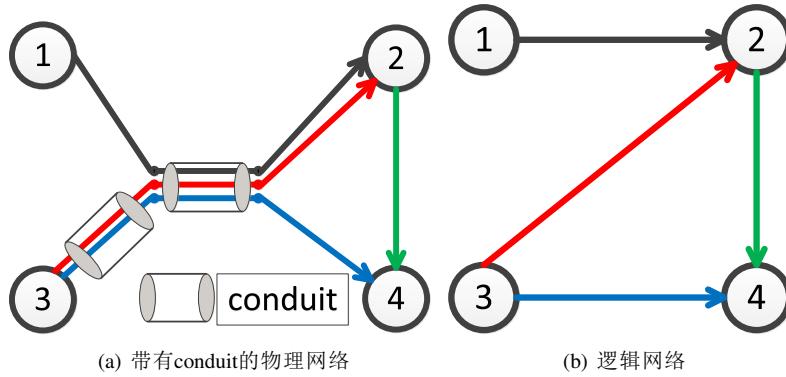


图 3.6 共享风险链接组(SRLG)实例

共享风险链路组不相交路径问题定义如下：

定义 3.7(共享风险链路组不相交路径问题) 给定 $|V|$ 个节点集 V 和 $|E|$ 条加权链路集 E 组成的有向网络 $G(V, E)$ 和 $|R|$ 个风险组 R , 两个特殊节点 $s, d \in V$ 。任何链路 $(u, v) \in E$, 给定一个整数 $k > 0$, 求 s 到 d 的 k 条路径 P_1, P_2, \dots, P_k , 使路径间不共享任何公共风险组。

风险组里的资源可以是节点而不一定是链路，这就引申出共享风险节点组不相交路径问题。共享风险节点组不相交路径可以通过节点转移方法成共享风险链路组不相交路径问题。

3.9 本章小结

如表3.1概述了所考虑的不相交路径问题的变体。每个问题都根据其约束条件、实际应用和时间复杂度进行分类。

表 3.1 不相交路径算法比较

问题	约束条件	实际应用	时间复杂度
不相交路径	路径不共享公用链路或节点	为运输网络中的卡车司机提供不同的候选路径	多项式可解(例如Bhandari算法 ^[55])
可靠性不相交路径	路径不共享公共链接或节点, 具有可靠性约束	提高电信网络中业务的连接可靠性	多项式可解(例如Bhandari算法 ^[55])
最大不相交路径	路径共享最小的公共链路或节点	在两个地点安全地转移一个非常重要的人	NP-hard和很难有近似算法其因子为 $2^{\log^{1-\epsilon} N}$ ($\epsilon > 0$)。当k=2时, 这个问题是多项式可解(例如MADSWIP算法 ^[59])
域不相交路径	路径不共享公共域	提高跨多个光网络域的业务传输可靠性。	NP-hard ^[60]
地区不相交路径	路径不能受到直径D的单一地区失效的影响(除了包括源或目标节点失效)	确保地区灾难不会同时影响不同不相交的道路	NP-hard和很难有近似算法 ^[61] , 除非所有成对的网络节点距离大于D
不相交路径对	路径不共享公共链路或节点(路径间可能有不同的源节点和目标节点)。	在芯片上互连指定的通道, 没有任何不同引脚的电线相互接触	当 $k \geq 2$ 在一般有向网络中为NP-hard ^[62] , 但在无向网络 ^[63] 、有向平面网络 ^[64] 和有向无圈网络中 ^[62] 是多项式可解的
风险共享链路组不相交路径	路径不共享共同的风险组	确保不同的不相交路径不使用属于光网络中同一管道的链路。	NP-hard ^[65] 和难以近似 ^[58] , 除非所有SRLG遵循星型性质且所有SRLG的数目为常数 ^[66] , 或当所有SRLG遵循星性且最大节点度最多为4 ^[66] 时, 或者当所有SRLG遵循星型性质且网络是有向无圈图 ^[66] 时, 或在特定跨度共享拓扑 ^[57] 中

第4章 共享风险链路组不相交路径对问题

4.1 问题描述

SRLG不相交路径在它们之间没有任何共同的风险资源，也就是说，由于风险而导致的路径失败不会影响其他路径。图4.1(b)显示两条SRLG 不相交路径对，表示为AP 和BP。因为这两条路径没有共同的风险资源，如果AP 失败，BP仍然可以工作。本章主要讨论了两条不相交的路径，即可以描述如下。

Min-Min SRLG不相交路径对问题。给定一个图 $G(V, E)$ ，每条链路 $e_i \in E$ 相关联一个权重 w_{e_i} ，一个源节点 s 和一个目的节点 d ，找到一对 s 到 d 的SRLG不相交路径对(表示为AP和BP)，而且要求这两条不相交路径中路径权重较小的那条路径权重最小化，形式化如下：

$$\begin{aligned} & \underset{AP, BP}{\text{minimize}} \quad \min(w_{AP}, w_{BP}) \\ & \text{subject to} \quad r_{AP} \cap r_{BP} = \emptyset \end{aligned} \quad (4.1)$$

$$AP \cap BP = \emptyset$$

即 w_{AP} 和 w_{BP} 是AP和BP的路径权重，AP 和BP分别是路径AP和BP 上的链路集， r_{AP} 和 r_{BP} 分别是影响路径AP和BP的SRLG 集。

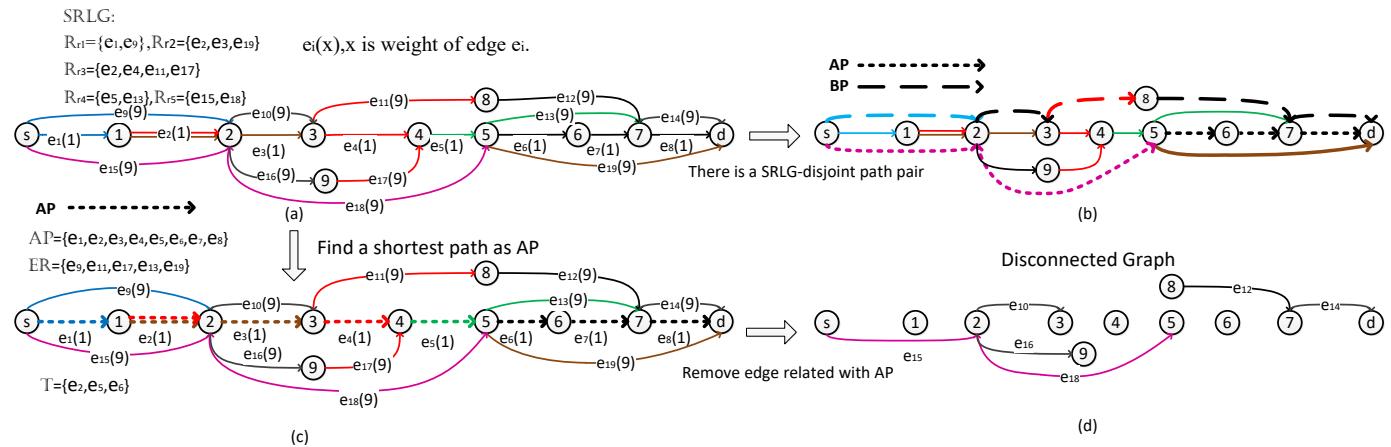


图 4.1 SRLG不相交路径对实例

4.2 原有算法概述

共享风险链路组(SRLG)是一组链路共享相同的一个组件，该组件的故障会导致在这个组里所有链路同时发生故障。就路径保护而言，尽管某些链路或者节点

不相交路径算法^[5,7,55,65,67-70]已经提出来，SRLG不相交路径问题是比较棘手的，而且这些原有研究是限制在一定领域的。比如当每个SRLG只包含一条链路时，这个SRLG不相交路由问题可以简化为链路不相交路径问题，而通过节点分裂方法(node split method)^[39]节点不相交路径问题可以转化为链路不相交路径问题。因为SRLG组通常包括的链路超过一条，并且网络中的链路通常可以属于多个SRLG组里，以至于求一对SRLG不相交路径问题比求一对链路或者节点不相交路径问题要困难得多。

为了解决SRLG不相交路径问题，一种可能的方法是0-1整数线性规划(ILP)^[65]，通过分支限界法(branch-and-bound)来搜索选择最优的主路径和备份路径。该方法时间复杂度高，不适用于大型网络。为了降低算法的复杂度，基于APF的启发式算法^[71-73]能够求Min-Min SRLG不相交路径问题的近似最优解。首先使用Dijkstra算法(或任何其他最短路径算法)求出主路径，求主路径时不考虑其相应的备用路径情况，在删除AP沿线的链路并且与AP共风险的节点和链路后，再利用最短路算法求的备用路径。

然而，使用APF启发式算法的有一个主要缺陷，一旦求得路径AP后也可能无法找到相对的SRLG不相交路径BP，即使网络中确实存在一对不相交路径。这就是所谓的“陷阱”问题，在节4.4将详细介绍陷阱问题产生的原因，即使稠密网络中^[74]这也是可能发生，在一个稀疏连接的网络中当然是不能被忽略。陷阱问题分为两种：不可避免的陷阱和可避免的陷阱。不可避免的陷阱是受拓扑约束的，任何算法都无法解决。如果网络不是2-边连通度的，则没有算法可以保证在拓扑中存在两个SRLG不相交路径。另一方面，当两个节点之间存在SRLG不相交路径对，但由于路由算法的缺陷而找不到时，就会出现一个可避免的陷阱，在本章中只考虑了可避免的陷阱。

对简单的APF算法扩展，提出了KSP(K-最短路径)算法来处理节点/链路不相交路径的陷阱问题。虽然它是处理陷阱问题最有效的算法之一，但它在大型网络中的性能受到影响，因为KSP可能会涉及多路径搜索测试(K测试)，直到它找到不相交路径。当前候选的路径AP遇到陷阱问题后，仅根据路径长度选择下一个要测试的候选AP，而不考虑当前候选AP的那条链路(或那些链路)导致查找不到不相交路径BP失败。因此，为了找到一对不相交路径对，需要对大量的路径进行测试，这就引入了KSP算法中与K相关的时间复杂度。对于遇到陷阱问题的AP，我们应用从AP路径导出的SRLG冲突链路集来指导将来的AP路径测试。这在很大程度上有助于减少寻找替代路径的时间复杂度。

其它SRLG不相交路径算法^[75-79]，搜索最大SRLG不相交路径对，并且路径间共享最小数目的公共链路。由于AP和BP可能具有相同的风险资源组，通过这种方法找到的解决方案是不可靠的。我们的算法目标是寻找完全SRLG不相交路径。

Xu^[80]试图找到完全SRLG不相交路径。但他的算法减少了问题的搜索空间，加快了路径搜索的速度。然而，它可能会以较大的代价返回路径，因为在削减后的搜索空间可能会失去最优解。相反，为了大大加快搜索过程，我们利用SRLG冲突链路集将原问题划分为多个子问题，这些子问题可以并行执行。因此，我们的算法可以运行得更快，返回主路径成本非常低。

Datta^[77]提出方法是将SRLG不相交路径问题转化为链路不相交路径问题，然后利用链路不相交路径算法来解决。然而，只有特殊的SRLG星型模式可以转换为链路不相交，这样就限制了该算法的广泛应用。当AP遇到陷阱问题时，CoSE^[76]算法试图找到一个SRLG集合，任何AP路径包含了这个SRLG集合里的所有SRLG，则必定找不到任何与其对应的SRLG不相交路径BP。CoSE首先通过多轮搜索查找多个AP共享的SRLG，并且组成一个SRLG集合，然后根据SRLG集合来划分原始问题以搜索SRLG不相交路径对。而不使用SRLG中链路之间共享风险的特性，CoSE方法的这种穷尽搜索需要非常高的计算开销。

4.3 复杂度规模

定理 4.1 Min-Min SRLG-不相交路径对问题是NP-complete问题.

证明：根据^[81]，Min-Min链路不相交路径对问题是NP-complete的。Min-Min链路不相交路径问题是Min-Min SRLG不相交路径问题的子问题。设Min-Min SRLG不相交路径问题的复杂性为C(A)，则NP-complete≤C(A)。

为了求Min-Min SRLG不相交路径问题的时间复杂度，我们首先假设了一个问题B(问题B的复杂度表示为C(B))，当找到两条的SRLG不相交路径并且路径较小的路径其权重小于或等于M(M是大于零的整数)。Min-Min SRLG不相交路径问题A与问题B等价，我们知道M必须大于零并且小于 $\sum_{e_i \in E} w_{e_i}$ 。例如，我们假设 $0 \leq M \leq 10$ 和 $m=6$ 是最优解，通过经典的二分法(binary search method)其时间复杂度为O(log(N))，如图4.2所示，通过二分法我们得到了两条不相交的路径，较小的路径其权重为m，因此问题A与问题B等价。假设程序X在输入问题B时，如果问题B没有解，则程序Y立即停止，否则程序B继续执行并获得问题B的解，因此问题B可以归结为NP-hard问题。因此C(B)≤NP-hard。

此外，给定任意两条路径，很容易在多项式时间内判别这两条路径是否为SRLG不相交路径，较小的路径其权重小于或等于M，从而使得C(B)≤NP-complete。当B的复杂度等于A时，我们有C(A)=C(B)≤NP-complete。因此，A=NP-complete。 □

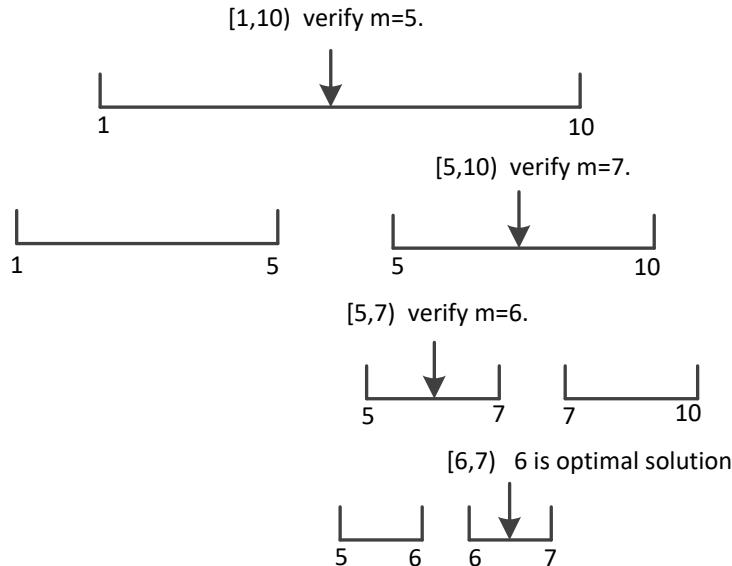


图 4.2 二进制搜索法求最优解实例

4.4 陷阱问题

基于APF的启发式算法可能会陷入“陷阱”问题。也就是说，当一个AP被确定时，即使网络中确实存在一对不相交路径对，它也可能无法找到SRLG不相交的BP路径。图ref{fig:CompositeGraph.(c),(d)}说明了陷阱问题，虚线表示一条AP路径，其链路集为 $\text{AP} = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$ ，在删除AP上的链路和与AP共享风险的链路后，图4.1.(d)所示的不存在从s到d的路径，因此找不到BP。

虽然KSP算法被认为是解决陷阱问题的有效算法，但它可能面临着效率低下的问题。如图4.3所示假设 e_1, e_2, e_3, e_4 的链路权重比其他链路大得多。此外，在 e_1, e_2, e_3, e_4 中， e_1 和 e_2 的链路权重远小于 e_3, e_4 。然后，在KSP算法多次找从s到d的K短路时，总是包含 e_1, e_2 （虚线表示）。则最短AP总会遇到陷阱问题，因为 e_1 和 e_4 具有相同的风险，因此无法找到BP。为了避免陷阱问题，必须将K设为一个大值，这给KSP带来了很高的时间复杂度。

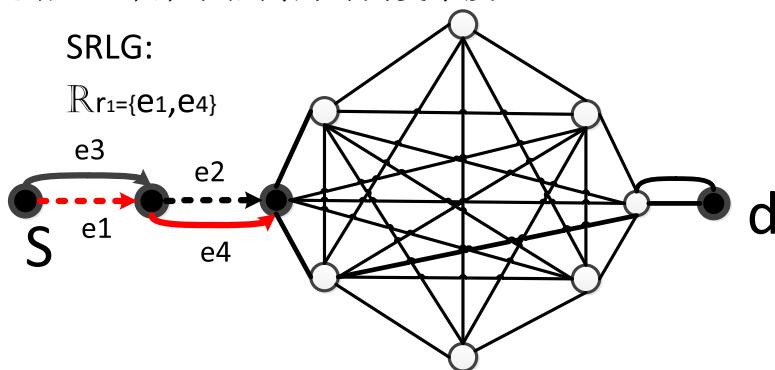


图 4.3 演示KSP算法效率低下的实例

4.5 分而治之的快速SRLG不相交路径对算法

当一个陷阱问题发生，并且对于给定的AP没有SRLG不相交路径BP时，AP中可能存在一个子链路集，这样任何通过这个链路集里所有的这些“问题”链路的AP都不能找到一条相对应SRLG不相交BP路径，我们称之为**SRLG冲突链路集**。与KSP不同，当最短AP路径遇到陷阱问题时，我们将通过两个主要步骤来解决这个问题。如图4.3所示的例子中，我们将首先找到图4.3中的SRLG冲突链路集合，然后应用分而治之算法将原问题划分为两个子问题 $\mathcal{P}(\emptyset, \{e_1\})$ 和 $\mathcal{P}(\{e_1\}, \{e_2\})$ 。这两个子问题可以在多核CPU平台上并行执行，快速得到SRLG不相交路径对。

4.5.1 分而治之

在得到SRLG冲突链路集后，设计了一种分而治之的算法，将原Min-Min SRLG不相交路径问题划分为多个子问题，并行执行，加快求SRLG不相交路径对的过程。

为了便于问题划分，我们首先定义了两个不相交的链路集合 \mathbb{I} 和 \mathbb{O} ，其中称 \mathbb{I} 为包含集集合， \mathbb{O} 称为排除集集合。由 $\mathcal{P}(\mathbb{I}, \mathbb{O})$ 表示的Min-Min SRLG不相交问题，用于寻找一对AP和BP，其中AP是所有可能的AP中最短的，其中路径AP必须经过 \mathbb{I} 集合里的所有链路和不经过 \mathbb{O} 集合里的所有链路。

最初，让 $\mathbb{I} = \emptyset$ $\mathbb{O} = \emptyset$ ，原来的Min-Min SRLG不相交路径对问题可以用 $\mathcal{P}(\emptyset, \emptyset)$ 表示。给定SRLG冲突链路集 $\mathbb{T} = \{e_1, e_2, \dots, e_{|\mathbb{T}|}\}$ ，原问题 $\mathcal{P}(\emptyset, \emptyset)$ 可按以下步骤划分。

1. 首先， $\mathcal{P}(\emptyset, \emptyset)$ 能被划分成两个子问题 $\mathcal{P}(\emptyset, \{e_1\})$ 和 $\mathcal{P}(\{e_1\}, \emptyset)$ 。
2. 类似， $\mathcal{P}(\emptyset, \{e_1\})$ 能被划分成两个子问题 $\mathcal{P}(\{e_1, e_2\}, \emptyset)$ 和 $\mathcal{P}(\{e_1\}, \{e_2\})$ 。
3. 这个划分步骤持续直到步骤 $|\mathbb{T}|$ ，问题 $\mathcal{P}(\{e_1, e_2, \dots, e_{|\mathbb{T}|-1}\}, \emptyset)$ 进一步的拆分成两个子问题 $\mathcal{P}(\{e_1, e_2, \dots, e_{|\mathbb{T}|-1}, e_{|\mathbb{T}|}\}, \emptyset)$ 和 $\mathcal{P}(\{e_1, e_2, \dots, e_{|\mathbb{T}|-1}\}, e_{|\mathbb{T}|})$ 。注意到，子问题 $\mathcal{P}(\{e_1, e_2, \dots, e_{|\mathbb{T}|-1}, e_{|\mathbb{T}|}\}, \emptyset)$ 是无解的。

除了子问题 $\mathcal{P}(\{e_1, e_2, \dots, e_{|\mathbb{T}|}\}, \emptyset)$ 外，我们将试图求其它每个子问题的最优解。然后选择最好的路径对(即主路径最短的路径对)，作为原问题 $\mathcal{P}(\emptyset, \emptyset)$ 的最终(最优)解。如果这些子问题都没有解，则我们可以得出原问题没有任何解，因为我们子问题包括了所有的可能的不相交路径对。

就时间复杂性而言，解决子问题所需的时间比原来的问题应该花费的更少。因为一条链路(来自集合 \mathbb{T})将在计算AP的路径时被去除，这也确保了不同的AP路径将被测试且是否存在一个SRLG不相交BP路径。

当遇到陷阱问题时，我们的解决方案将划分原来的问题，并测试每个子问题以寻找到最终的解。在我们的分而治之方法中，子问题是由于SRLG冲突链路集而

得，而这个SRLG冲突链路集却是当AP路径遇到陷阱问题生成的。与现有的算法相比较，该算法在不考虑现有结果和问题的情况下，可以在很大程度上降低算法的计算量。对于图4.4中的例子所示，SRLG冲突链路集是 $\mathbb{T} = \{e_2, e_5, e_6\}$ 。拆分过程过程如图4.4所示。根据SRLG冲突链路集，我们应该测试总共3个子问题 $\mathcal{P}(\{e_2, e_5\}, \{e_6\})$, $\mathcal{P}(\{e_2\}, \{e_5\})$ 和 $\mathcal{P}(\emptyset, \{e_2\})$ ，其中选择AP路径权重最低的最优子问题作为原问题 $\mathcal{P}(\emptyset, \emptyset)$ 最终的(最优)解。

注意，我们不需要解决子问题 $\mathcal{P}(\{e_2, e_5\}, \emptyset)$ 和 $\mathcal{P}(\{e_2\}, \emptyset)$ ，因为它们的解已经包含在其它的子问题中。第一个解空间由两个子问题 $\mathcal{P}(\{e_2, e_5, e_6\}, \emptyset)$ 和 $\mathcal{P}(\{e_2, e_5\}, \{e_6\})$ 组成。由于SRLG冲突链路集为 $\mathbb{T} = \{e_2, e_5, e_6\}$ ，显然，子问题 $\mathcal{P}(\{e_2, e_5, e_6\}, \emptyset)$ 是没有解的。因此， $\mathcal{P}(\{e_2, e_5\}, \{e_6\})$ 的解空间等于 $\mathcal{P}(\{e_2, e_5\}, \emptyset)$ 的解空间。同样， $\mathcal{P}(\{e_2\}, \emptyset)$ 的解空间包括 $\mathcal{P}(\{e_2\}, \{e_5\}, \emptyset)$ 和 $\mathcal{P}(\{e_2\}, \{e_5\})$ 的解空间。

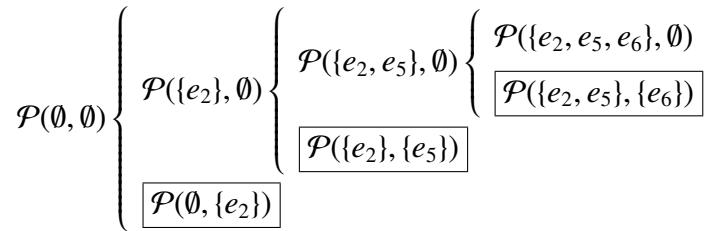


图 4.4 分而治之的解决方案

4.5.2 SRLG冲突链路集合

在本节中，我将描述了如何找到一个SRLG冲突链路集合，当在网络 G 中给定一个AP路径并且没有SRLG不相交的BP路径。

4.5.2.1 通过新奇的边容量设置准则构造一个新图 G^*

如4.5.2.2节介绍的，如果在图 G 中去除所有在边割集 \mathbb{L}_Φ 的所有边，则 $|f| = 0$ 。也就是说，不存在任何流能从 s 到 d 。在本文中，我们试图基于割集的概念找到SRLG冲突链路集合。如果AP从 s 到 d 流通，经过的链路与割集 \mathbb{L}_Φ 共享风险，则找不到任何与其对应的SRLG不相交路径BP，因为没有一条在割集中的链路可以被BP选择。

基于割集基础为了找到SRLG冲突链路集，我们构造了一个新图 G^* ，如下所示。

1. G^* 与 G 的节点和链路拓扑关系一样。
2. 跟每条链路 e_i 相关的链路权重 w_{e_i} 是跟其相对应图 G 中边的权重一样的。
3. 我们使用公式4.2的准则设置每条边 $e_i \in \mathbb{E}$ 相关的容量 c_{e_i} 。

$$c_{e_i} = \begin{cases} 1 & e_i \in AP \\ |\text{AP}| + 1 & e_i \in ER \\ |\text{AP}| + (|\text{AP}| + 1) \times |\text{ER}| + 1 & otherwise \end{cases} \quad (4.2)$$

AP 指在图 G 中较小权重路径 AP 上所有链路的集合， ER 指不属于路径 AP 上的边但是与路径 AP 上的边共享风险的链路集合。

如图4.1(c)所示，路径 AP 的边集合 $\text{AP} = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$, $\text{ER} = \{e_9, e_{11}, e_{17}, e_{13}, e_{19}\}$ 。 $|\text{AP}| = 8$, $|\text{ER}| = 5$, $|\text{AP}| + 1 = 9$ 和 $|\text{AP}| + (|\text{AP}| + 1) \times |\text{ER}| + 1 = 54$ 。我们产生一个新图 G^* 如图4.5所示，在图 G^* 中边的容量是根据公式(4.2)所设置。

r_P 表示影响路径 P 上的风险集合，即 $r_P = \{r \in \mathbb{R}: \text{路径 } P \text{ 包含的链路在 } \mathbb{R}_r \text{ 中}\}$ 。如图4.1(c)所示，在路径 AP 上的边集 $\text{AP} = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$, 并且 $e_1 \in \mathbb{R}_{r_1}$, $e_2 \in \mathbb{R}_{r_2}$, $e_3 \in \mathbb{R}_{r_3}$, $e_4 \in \mathbb{R}_{r_3}$, $e_5 \in \mathbb{R}_{r_4}$, 路径 AP 的风险集合是 $r_{AP} = \{r_1, r_2, r_3, r_4\}$ 。 ER 代表不属于 AP 上的链路但是与 AP 共享相同的风险集合的链路。如图4.1(c)所示， $\text{ER} = \{e_9, e_{11}, e_{17}, e_{13}, e_{19}\}$ 。

4.5.2.2 最小割最大网络流定理

设 $G = (\mathbb{V}, \mathbb{E})$ 是一个网络(其中 \mathbb{V} 是 $|\mathbb{V}|$ 个节点的集合， \mathbb{E} 是 $|\mathbb{E}|$ 条链路的集合)，其中 $s \in \mathbb{V}$ 和 $d \in \mathbb{V}$ 分别指源节点和终节点。链路 e_i 的容量表示该条链路的最大流量。链路的流 f_{e_i} 应该满足以下两个限制：

1. 容量限制: $\forall e_i \in \mathbb{E}: f_{e_i} \leq c_{e_i}$.
2. 流量守恒: $\forall u \in \mathbb{V} - \{s, d\}: \sum_{v \in \mathbb{V}} f_{(v,u)} = \sum_{v \in \mathbb{V}} f_{(u,v)}$, (v, u) 和 (u, v) 代表链路 $e(v, u)$ 和 $e(u, v)$.

流的值定义为 $|f| = \sum_{v \in \mathbb{V}} f_{(s,v)}$, 其中 s 是源节点。它表示从 s 节点到 d 节点的流量。

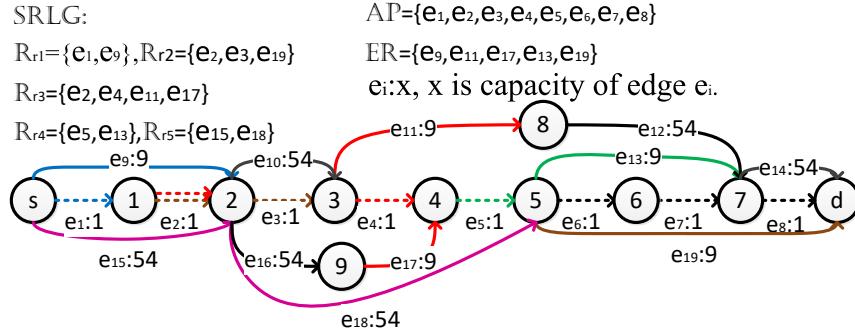
最大流量问题: 尽可能的求从 s 节点到 d 节点的最大流量值 $|f|$ 。

一个 s - d 割 $\Phi = (\mathbb{S}, \mathbb{D})$ 是节点 \mathbb{V} 的划分且 $s \in \mathbb{S}$ 和 $d \in \mathbb{D}$, Φ 的割集合 \mathbb{L}_Φ 是一个包含边的集合。

$$\mathbb{L}_\Phi = \{(u, v) \in \mathbb{E} : u \in \mathbb{S}, v \in \mathbb{D}\}. \quad (4.3)$$

如果在割集合 \mathbb{L}_Φ 中的边被去除，那么在原图中的流值 $|f| = 0$ 。即没有流能从 s 节点到 d 节点。一个 s - d 割 $\Phi = (\mathbb{S}, \mathbb{D})$ 的容量被定义成 $c(\Phi) = \sum_{e_i \in \mathbb{L}_\Phi} c_{e_i}$ 。最小子-d割 Φ 问题，最小化 $c(\Phi)$ 即决定点集 \mathbb{S} 和 \mathbb{D} 使得 s - d 割 $\Phi = (\mathbb{S}, \mathbb{D})$ 的($c(\Phi)$)最小化。

最小割最大流定理: 一个 s - d 流的最大值等于 s - d 割的最小割。如图4.6所示，在图 G 中的流量 $|f| = f_{(s,v_1)} + f_{(s,v_2)}$ 。这个割 $\Phi(\mathbb{S}, \mathbb{D})$ 是 $\mathbb{S} = \{s, v_1, v_2, v_4\}$ 和 $\mathbb{D} = \{v_3, d\}$ ，它是最小的割其容量为 $c(\Phi) = c_{(v_1, v_3)} + c_{(v_4, v_3)} + c_{(v_4, d)} = 12 + 7 + 4 = 23$. 显然，

图 4.5 图 G^* 实例

$|f| = c(\Phi)$, 即 $s-d$ 最大流等于所有 $s-d$ 割中最小的容量。

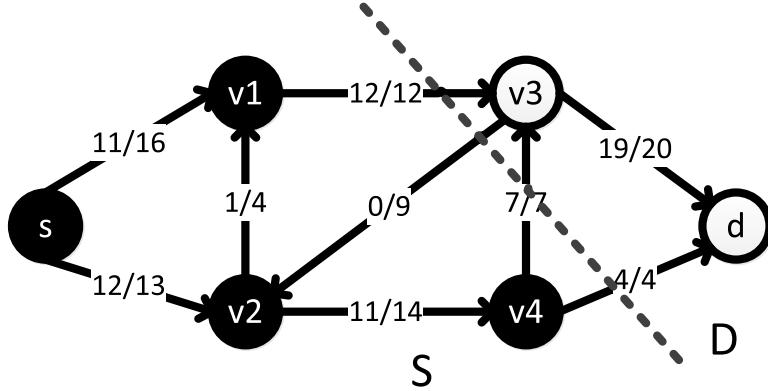


图 4.6 最大流最小割定理实例

4.5.2.3 新图 G^* 中最小割集性质

最大流最小割定理指出，在网络中，从源节点 s 到目的节点 d 的最大流值等于最小边割集的总容量，即最小的链路总容量，如果删除最小边割集上的边，将断开目的节点 d 与源节点 s 的连通。我们的算法基于图的最小割集的链路集来求得最小SRLG冲突链路集。我们将首先展示我们重建新图 G^* 的一些优良特性为求得最小SRLG冲突链路集。

引理 4.1 在图 G^* 中任何从 s 到 d 的路径必须经过在AP或者ER中边集合的一条边。

证明：用矛盾法证明这条引理。假设对于某条路径AP，有另一条路径从 s 到 d 在图 G^* 中不与AP共享风险，即这条路径不通过AP或者ER的任何一条链路。可以很容易得出这样的结论，这条路径是与路径AP对应的SRLG不相交路径BP。这与AP没有对应的SRLG不相交路径的说法相矛盾。□

引理 4.2 图 G^* 的任何一条最大流的流值最多为 $|AP| + (|AP| + 1) \times |ER|$ 。

证明：假设 G^* 的最大流 f 为 $|f| = k$ 。在 G^* 中， f 可以被划分为 k 个从 s 到 d 的1单元流。根据引理4.1，这些1-单位流中的每一个都必须通过通过AP或者ER的至少一条链路。请注意，AP或者ER中链路的容量分别为1或者 $|AP|+1$ 。根据AP和ER中链路

的容量设置，因为AP中的链路只能承载1-单位流量，而在ER中一条链路在ER中能最多承载 $|AP|+1$ 单位流量。因此，最多只能有 $|AP| + (|AP| + 1) \times |ER|$ 个从s到d的单位流量。 \square

引理 4.3 图 G^* 最小割 Φ 的边割集 L_Φ 上所有链路都在AP 或者ER 中。

证明：根据最大流最小割定理， $c(\Phi)$ 表示的最小切割 Φ 的容量，其应该等于最大流量值，根据引理4.2最大流量最多为 $|AP| + |ER| \times (|AP| + 1)$ 。根据容量设定原则如公式4.2所示，一条链路既不在AP 也不在ER ($e_i \notin AP$ 和 $e_i \notin ER$)，则这条链路的容量为 $c_{e_i} = |AP| + (|AP| + 1) \times |ER| + 1$ ，这条链路是大于 $|AP| + (|AP| + 1) \times |ER|$ 。因此，这条链路不可能在 L_Φ 中。因此，割集 L_Φ 中的所有链路都必须属于边集合AP 或者ER。 \square

定理 4.2 如果在图 G 中一个单元流阻塞了全部边割集 L_Φ 的所有边，则在原图中不会存在任何流经过这个图的割 Φ 。

证明：如果在图 G 中一个单元流阻塞了全部边割集 L_Φ 的所有边，则在原图中没有流能使用边割集 L_Φ 里的边和没有任何流能经过这个图的割 Φ 。 \square

定理4.2提供了找到SRLG冲突链路集的可能性。即当AP路径遇到陷阱问题时，我们找到AP 路径边集的子集能够阻塞原有在边割集 L_Φ 的所有边，所以这个子集合即为SRLG 冲突链路集。当一条路径包含SRLG冲突链路集里的所有边，则没有任何流能经过割集 Φ ，因此没有与其对应的SRLG不相交路径BP。

4.5.2.4 SRLG冲突链路集的集合覆盖问题

虽然AP路径上的所有链路一起也可以构成SRLG冲突链路集，我们感兴趣的是尽量规模较小的SRLG冲突链路集，SRLG 冲突链路集的大小确定相互间互斥的子问题规模。

根据定理4.2，最小SRLG冲突链路集问题可以描述为：查找AP 链路集上的最小链路子集，这些链路可以阻塞边割集 L_Φ 。

对于任何链路 e_i ， \mathbb{SR}_{e_i} 表示与链路 e_i 共享风险的链路集。显然， \mathbb{SR}_{e_i} 包括 e_i 本身和所有包含链路 e_i 风险贡献链路组SRLG集合的所有边。例如，如图4.7所示， e_i 是在两个SRLG中 $\mathbb{R}_{r_2} = \{e_2, e_3, e_{19}\}$ ， $\mathbb{R}_{r_3} = \{e_2, e_4, e_{11}, e_{17}\}$ ，因此， $\mathbb{SR}_{e_2} = \{e_2, e_3, e_{19}, e_4, e_{11}, e_{17}\}$ 。

对每条在AP上的路径 e_i ，我们定义每条边的cut-block-link集合为 $\mathbb{B}_{e_i} = \mathbb{SR}_{e_i} \cap L_\Phi$ ，这个集合是边割集 L_Φ 的子集，能通过 e_i 而堵塞这个集合的所有边。

因此，最小SRLG冲突链路集问题可以定义为一个集合覆盖问题：给定AP(路

径AP上的链路集合)、边割集 \mathbb{L}_Φ 和cut-block-link集合 $\mathbb{B}_{e_1}, \mathbb{B}_{e_2}, \dots, \mathbb{B}_{e_{|\mathbb{A}\mathbb{P}|}}$ 。我们求出最小cut-block-link集合集，其交集是边割集 \mathbb{L}_Φ ，即最小规模的 $\mathbb{T} \subseteq \{e_i | e_i \in \mathbb{A}\mathbb{P}\}$ 以至于 $\cup_{e_i \in \mathbb{T}} \mathbb{B}_{e_i} = \mathbb{L}_\Phi$ 。

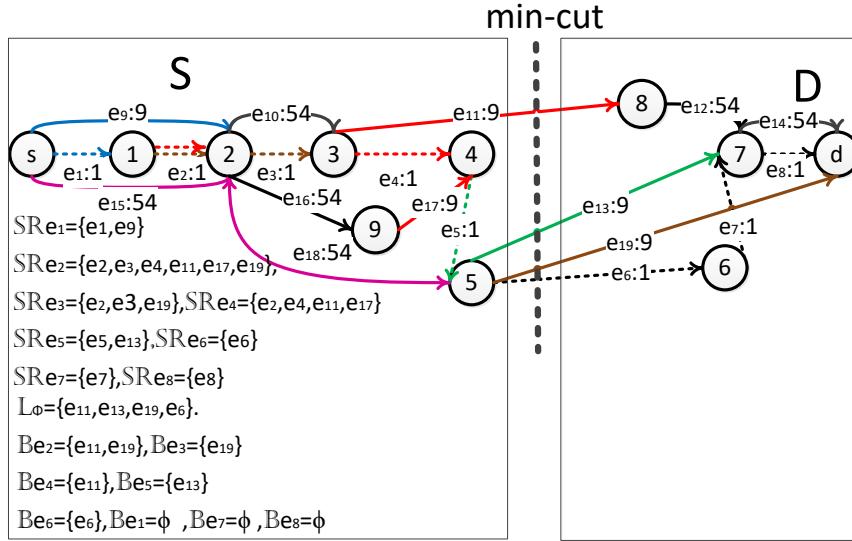


图 4.7 图 G^* 的最小割实例

集合覆盖问题通常是一个NP-hard问题。它的复杂性取决于元素的大小(表示 (N))。在我们的最小SRLG 冲突链路集问题中， $n = |\mathbb{L}_\Phi|$ ，即割集 \mathbb{L}_Φ 的边数。因为本文的重点不是改进集合覆盖问题算法，我们应用^[82]中提出的算法。在复杂度 $O(\log(|\mathbb{L}_\Phi|))$ 的情况下求出最小的SRLG 冲突链路集，通常即使是在大规模的网络，最短的路径作为AP都没有大跳数 $n = |\mathbb{L}_\Phi|$ 。因此，用集合覆盖问题来求最小值SRLG冲突链路集代价不是很大。

为了说明如何找到最小SRLG冲突链路集，如图4.7 所展示了一个例子，求最小 $\Phi(\mathbb{S}, \mathbb{D})$ ， $\mathbb{S} = \{s, 1, 2, 3, 4, 5, 9\}$ 和 $\mathbb{D} = \{d, 6, 7, 8\}$ ，边割集 $\mathbb{L}_\Phi = \{e_{11}, e_{13}, e_{19}, e_6\}$ 。对于AP路径上的所有链路，cut-block-link集合是： $\mathbb{B}_{e_1} = \emptyset$, $\mathbb{B}_{e_2} = \{e_{11}, e_{19}\}$, $\mathbb{B}_{e_3} = \{e_{19}\}$, $\mathbb{B}_{e_4} = \{e_{11}\}$, $\mathbb{B}_{e_5} = \{e_{13}\}$, $\mathbb{B}_{e_6} = \{e_6\}$, $\mathbb{B}_{e_7} = \emptyset$ and $\mathbb{B}_{e_8} = \emptyset$ 。为了覆盖 \mathbb{L}_Φ ，最少的cut-block-link集合是 $\mathbb{B}_{e_2} = \{e_{11}, e_{19}\}$, $\mathbb{B}_{e_5} = \{e_{13}\}$, $\mathbb{B}_{e_6} = \{e_6\}$ 。因此，最小SRLG冲突链路集是 $\mathbb{T} = \{e_2, e_5, e_6\}$ 。

如图4.7所示的例子中，虽然 $|\mathbb{L}_\Phi| = 4$ ，但是最小SRLG冲突链路集 $|\mathbb{T}| = |\{e_2, e_5, e_6\}| = 3$ 是小于 $|\mathbb{L}_\Phi| = 4$ 。这是因为 e_2 属于 \mathbb{R}_{r_2} 和 \mathbb{R}_{r_3} ，并能阻塞在割集 \mathbb{L}_Φ 中的两条链路 e_{11} 和 e_{19}

根据SRLG的拓扑类型^[77]，SRLG的拓扑类型：星型类型和非星型类型。对星型类型SRLG，所有链路都从同一个节点开始或结束在同一个节点。例如，如图4.7所示， e_1 和 e_9 来自相同的节点 s ， \mathbb{R}_{r_1} 是星型SRLG。对非星型类型，并不是SRLG 中的所有链路都是从相同节点开始或结束于同一节点。如图4.7 所示， \mathbb{R}_{r_2} , \mathbb{R}_{r_3} , \mathbb{R}_{r_4} 和 \mathbb{R}_{r_5} 是非星型类型。而且，即使在图4.7 所示包括星型SRLG 和无星

型SRLG，我们的算法高效且有效地通过解决集合覆盖问题来解决冲突集。

因此，与现有的一些研究不同的是，^[77]只能处理单个SRLG类型，这样的简单场景中一条链路只属于一个SRLG，我们的算法能更有效的处理多种情形。在更一般的情况下，链路可以属于一个或多个SRLG具有更多不同的SRLG类型。

4.5.3 算法步骤

算法4.5.1显示了完整的min-min SRLG不相交路由算法。算法中的输入参数包括网络图(G)、源节点(s)、目的节点(d)、包含链路集应包括在AP(\mathbb{I})中，而排除链路集应不包含在AP(\mathbb{O})中。算法的输出结果是SRLG不相交路径对(AP, BP)。

为了寻找SRLG不相交路径，首先通过步骤2中的FIND_AP($G, s, d, \mathbb{I}, \mathbb{O}$)搜索网络中最小权重AP路径，其中 $\mathbb{I} = \emptyset, \mathbb{O} = \emptyset$ ，然后在步骤4中通过FIND_SRLG_Disjoint_BP(G, s, d, AP)搜索BP。特别是，可以通过Dijkstra算法找到AP路径。为了计算BP，FIND_SRLG_Disjoint_BP(G, s, d, AP)包括两个步骤。首先，对于AP上的所有链接，删除与这些链接有共同风险的链接。第二，Dijkstra的算法再次运行在网络其余的链路上，计算从 s 到 d 的第二条最短路径BP。

如果我们能找到一个SRLG不相交的BP路径，则解决Min-Min SRLG不相交的路由问题，并如步骤6所示返回找到的路径对。否则，就会出现陷阱问题。为了处理陷阱问题，步骤8首先找到SRLG冲突链路集 T ，第10步将原Min-Min SRLG不相交路由问题划分为基于冲突集 T 的 $|T|$ 个子问题。所有子问题都可以并行执行。步骤11使用集合 F 存储满足 $AP_i \neq \emptyset$ 和 $BP_i \neq \emptyset$ 的可行解。在 F 中的所有可行解中，选择AP路径权重最低的路径对作为原Min-Min SRLG不相交路由问题的最优解。

我们以图4.1中的例子来说明我们的算法4.5.1。根据步骤2，我们的算法首先通过Dijkstra算法搜索路径权重最小的路径 $AP = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$ ，路径显示为图4.1(c)中的虚线。在删除AP上的链路以及与AP共享共同风险的链路之后，我们得到图4.1(d)，这是一个不连通图没有BP路径。然而，如图4.1(b)所示，拓扑中实际存在一对SRLG不相交路径。因此，陷阱问题就会发生。在步骤8中找到SRLG冲突链路集 $\{e_2, e_5, e_6\}$ 之后，我们采用分而治之的方法将原问题 $P(\emptyset, \emptyset)$ 划分为三个子问题，根据步骤8，并行执行这些子问题 $P(\emptyset, \{e_2\})$ ， $P(\{e_2\}, \{e_5\})$ 和 $P(\{e_2, e_5\}, \{e_6\})$ 后，返回具有最小AP权重的路径对。

4.5.4 算法时间复杂度

当AP遇到陷阱问题时为了找到SRLG不相交路径对，我的算法首先计算出SRLG冲突链路集，然后通过把原问题划分成 T 个子问题来求解原问题。如4.5.2.4节所述边割集 L_Φ 的边数规模通常不是很大，因此，本算法寻找SRLG冲突链路集不会带来太多的时间成本。因此，我们关注的是路径查找过程的计算开

算法 4.5.1 Min-Min SRLG不相交路径对算法

Input: G : 网络图

s : 源节点

d : 目的节点

\mathbb{I} : 必过链路集

\mathbb{O} : 必不过链路集

Output: AP: 主路径

BP: 备用路径

1: $AP = \emptyset, BP = \emptyset, \mathbb{I} = \emptyset, \mathbb{O} = \emptyset$

2: $AP \leftarrow \text{FIND_AP}(G, s, d, \mathbb{I}, \mathbb{O})$

3: **if** $AP \neq \emptyset$ **then**

4: **return** $BP \leftarrow \text{FIND_SRLG_Disjoint_BP}(G, s, d, AP)$

5: **if** $BP \neq \emptyset$ **then**

6: **return** 路径对(AP, BP)

7: **else**

8: 找到SRLG冲突链路集 \mathbb{T}

9: $\mathbb{T} \leftarrow \mathbb{T} - (\mathbb{I} \cup \mathbb{O})$

10: 分而治之的并行执行

$$(AP_1, BP_1) = \text{Min-Min}(G, s, d, \mathbb{I}, \mathbb{O} \cup \{t_1\}),$$

$$(AP_2, BP_2) = \text{Min-Min}(G, s, d, \mathbb{I} \cup \{t_1\}, \mathbb{O} \cup \{t_2\}),$$

$$(AP_3, BP_3) = \text{Min-Min}(G, s, d, \mathbb{I} \cup \{t_1, t_2\}, \mathbb{O} \cup \{t_3\}),$$

...

$$(AP_{|\mathbb{T}|}, BP_{|\mathbb{T}|}) = \text{Min-Min}(G, s, d, \mathbb{I} \cup \{t_1, t_2, \dots, t_{|\mathbb{T}|-1}\}, \mathbb{O} \cup \{t_{|\mathbb{T}|}\})$$

11: $F \leftarrow \text{FIND_FEASIBLE}((AP_1, BP_1), \dots, (AP_{|\mathbb{T}|}, BP_{|\mathbb{T}|}))$

12: **if** $F \neq \emptyset, \emptyset$ **then**

13: **return** 路径对(AP, BP) 满足条件 $AP = \arg \min_{AP} \{F\}$

14: **end if**

15: **end if**

16: **end if**

销。

一般来说，对于一个有 $|\mathbb{E}|$ 条链路和 $|\mathbb{V}|$ 个节点的网络，求最小权重路径问题的时间复杂性是 $(|\mathbb{E}| + |\mathbb{V}|) \times \log(|\mathbb{V}|)$ 。为了解决陷阱问题，我们的路径查找问题与最初的最小权重路径问题有点不同。我们在路径查找的过程中引入了一些约束条件。例如，查找AP路径必须通过必过链路集 \mathbb{I} 和必不过链路集 \mathbb{O} 。由于这些链路集通常并不大，这些约束在成本计算上几乎没有差别。因为不同的子问题有不同的链路集，为了使描述简单明了，我们仍然使用 $(|\mathbb{E}| + |\mathbb{V}|) \times \log(|\mathbb{V}|)$ 作为一次路径搜索时间复杂度。而我们算法将原问题分成 $|\mathbb{T}|$ 个子问题，算法复杂度为 $|\mathbb{T}| \times (|\mathbb{E}| + |\mathbb{V}|) \times \log(|\mathbb{V}|)$ 。

对于不同算法复杂性的比较，我们也展示了在CoSE^[76]和KSP^[73]在路径查找过程中的复杂性。

CoSE试图找到一个冲突的SRLG集合，而不是一个冲突链路集。但是他们寻找冲突的SRLG集的方法是穷尽的查找而且成本很高。在这我们主要分析研究路径查找过程时间成本。由于我们的SRLG冲突链路集是由最小割和集合覆盖问题

导出的， $|T|$ 是最小的SRLG冲突链路集规模。因此，在CoSE中的冲突SRLG集至少是 $|T|$ ，并且我们表示SRLG集合为 $\{SRLG_1, SRLG_2, \dots, SRLG_{|T|}\}$ ，由于每个SRLG路径包含多条链路，因此CoSE的子问题应该比我们的要大得多。在AP路径上的一个SRLG的必过链路集合和不过链路集会产生 $|SRLG|$ 个子问题。所以一个SRLG集合 $\{SRLG_1, SRLG_2, \dots, SRLG_{|T|}\}$ 将引入 $\prod_{i=1}^{|T|} |SRLG_i|$ 个子问题。因此CoSE的复杂性是 $\prod_{i=1}^{|T|} |SRLG_i| \times (|\mathbb{E}| + |\mathbb{V}|) \times \log(|\mathbb{V}|)$ ，这是比我们的算法那复杂度大得多的。

对于KSP算法^[73]，路径查找复杂度为 $K \times ((|\mathbb{E}| + |\mathbb{V}|) \times \log(|\mathbb{V}|))$ ，其中K是在发现SRLG不相交路径对之前应该测试的路径次数。然而，由于KSP没有从前面的路径搜索过程利用前面的信息，在最坏的情况下，KSP可能尝试从源节点s到目的节点d的所有路径。因此，最糟糕的K值是 $2^{|\mathbb{E}|}$ ，这会带来很大的计算成本。

4.6 实验环境与评价指标

因为我们找不到任何拓扑带有SRLG链路属性，所以我们通过注入SRLG信息生成一个合成的数据集。拓扑数据有7种不同的拓扑，具有不同的节点数目、链路数目、链路权重(表示链路的延迟或其他参数)，如表4.1所示显示了拓扑的基本属性。这7个拓扑中的节点、链路)。

表 4.1 SRLG拓扑数据

拓扑	1	2	3	4	5	6	7
点	527	521	521	2023	451	521	449
边	4158	4052	4152	4142	2780	4052	2778
No.SRLG	132	86	89	207	210	128	88
SRLG边比率	9.66%	6.16%	6.18%	14.94%	22.55%	9.65%	9.53%

为了进行性能比较，除了我们的算法(名为SCLS)，我们还实现了另外四个SRLG不相交路径对算法。算法如下：

1. **ILP:** ^[65] 中的工作旨在寻找SRLG不相交路径对。通过整数线性规划方程使这两条路径总的权重最小化。我们没有找到其他通过整数线性规划方程寻找Min-Min SRLG不相交路径对的方法的研究。因此，从^[65]的整数规划方程，通过改变目标函数构造Min-Min SRLG 不相交路径对问题的整数线性规划方程。
2. **IQCP:** 因为任意0-1整数线性规划方程，其中所有变量为0或1，原问题的整数线性规划方程可以表示为一个二次约束方程，我们还设计了Min-Min SRLG 不相交路径问题成一个整数二次约束规划(IQCP)^[65]。
3. **KSP^[73]:** 它在源节点s和目的节点d中找到第K短路作为候选AP路径，一个接一个地测试候选的AP路径是否有相应的SRLG不相交路径BP，直到我们发现了这样的BP，算法才结束。

4. CoSE^[76]: 当AP遇到陷阱问题时, CoSE尝试进行简单而详尽的搜索, 以找到一个SRLG集合。任何AP路径通过这个SRLG集都无法找到SRLG不相交的BP路径。基于这个SRLG集合, 它划分原问题并设计算法来求SRLG不相交路径对。

前两者(ILP和IQCP)是基于整数规划模型的。在我们的实现中, 工具GUROBI 7.0^[83]用于解决这两个整数规划问题。六种性能指标用于评估不同的SRLG分路径算法:

- 路径权重: 路径中链路权重的总和。
- 路径跳数: 路径中的跳数。
- 运行时间: 查找SRLG不相交路径对的归一化平均时间。
- 算法加速比: 给定两种不同的算法(alg_1 和 alg_2)的计算时间, 表示为 T_1 和 T_2 , 算法 alg_1 对于算法 alg_2 计算时间上的加速比是 $alg_1: S_{1-2} = T_1/T_2$ 。
- 核加速比: 并行程序的核心加速比^[84]通常定义为 $S_p = \frac{T_1}{T_p}$, 其中p是处理器内核和 T_1 和 T_p 表示在1核和p核上的运行时间。
- 效率^[84]: 定义为 $E_p = \frac{S_p}{p} = \frac{T_1}{pT_p}$, 它是百分比(0, 1)的范围内。

所有实现都是在linux服务器上运行的, 这个服务器配置Intel(R) Xeon(R) CPU E5-2620 2.00GHz(24核)和32.00GB内存。为了测量计算时间, 我们在所有实现的算法中插入一个定时器。

我们通过在拓扑数据集里注入SRLG链路属性产生了两种SRLG类型, 星型类型和非星型类型。在光纤网络中, SRLG是星型类型, 而在其他网络类型中, 例如一个Overlay网络中, SRLG可以是非星型的。每个SRLG组是通过随机选择2-5链路组成的。在五种SRLG不相交路径对算法中, 只有CoSE和我们SCLS是并行算法。尽管ILP、IQCP和KSP不是并行算法, 我们仍然在实现它们来体现出我算法设计所获得的速度增益, 归一化^[85]所有拓扑数据的结果为最终结果。

4.7 算法性能评估及比较

从4.5.4节分析, 在KSP下求第一条K最短路的计算复杂度是 $2^{|E|} \times ((|E| + |V|) \times \log(|V|))$, 最坏的情况下这将是 $K \times ((|E| + |V|) \times \log(|V|))$ 。与分析一致, 当我们使用7个拓扑运行KSP, 没有仿真结果能在1小时内返回, 而其他算法则可以在11秒内。计算时间长使得KSP难以在实践中使用。因此, 我们不在结果显示KSP。

1. 路径权重: 如图4.8所示, 显示AP路径权重、BP路径权重和AP和BP的总和路径权重。显然, 所有的算法SCLS、CoSE、ILP和IQCP实现了相同的AP路径权重。但是不同的算法有不同的BP权重, 因此它们有不同的路径权重

和。因为所有算法都解决了SRLG不相交路径对问题，尽管他们发现不同的SRLG不相交路径对，它们都能达到找到最小权重相等的AP路径。然而，这两个基于整数线性规划的算法，ILP 和 IQCP，主要是找到最小化AP的权重但是其随意找到其它任何SRLG不相交路径BP，因此这两个算法搜索到的BP路径是不同的。

2. 路径跳数：图4.8显示AP路径跳数、BP路径跳数和AP和BP路径跳数之和，因为所有算法的目标都是最小化SRLG不相交路径对的最小路径权重。在路径跳数中，它们具有相同的AP权重(如图4.8所示)但是他们有不同的AP路径跳数(如图4.9所示)。尽管所有算法的AP路径权重都小于BP路径权重如图4.8所示，在图4.9中，AP跳数可能并不总是少于BP路径跳数。
3. 运行时间：如图4.10所示，通过改变使用的CPU核数来展示不同算法下的运行时间。由于CoSE下的运行时明显大于其他算法的运行时，为了更清楚地展示其他算法的结果，我们在图4.11 中通过排除CoSE来进一步绘制运行时间的结果。由于ILP和IQCP不是并行算法，这些算法在不同核数下的运行时间大致相等。我们的SCLS和CoSE的运行时间随着处理器核数的增加而减少，因为这两种算法可以将原问题划分为多个子问题来并行执行，并利用多核CPU的并行性来加快路径搜索的速度，虽然CoSE是一种并行算法，但计算时间比ILP和IQCP还要大。一些可能的原因包括：1)在CoSE中冲突SRLG集合的搜索过程效率不高；2)由于一个SRLG 通常包含多条链路，基于冲突SRLG问题的划分将带来大量的子问题需要解决，这也将带来大量的计算量。与CoSE不同的是，当AP上遇到陷阱问题时我们的SCLS根据图中的最小割集理论来求SRLG冲突链路集，并达到图4.10中所示的最低时间消耗。这说明了我们的冲突链路集查找算法是有效的，并且我们提出的分治算法和基于SRLG冲突链路集的智能AP搜索过程，可以大大降低计算量。
4. 算法加速比：如图4.13所示，我们进一步比较了它们的计算速度。特别地，为了找出使用不同算法寻找所需路径时所获得的加速比，我们使用CoSE作为基准算法，并设置 $alg_1 = \text{CoSE}$ 。与图4.10中的结果相似，在图4.13中SCLS的加速度是CoSE的1000倍以上。在图4.13中由于CoSE的运行速度明显小于其他算法，很难在图4.13中观察到，我们在图4.14中排除了最大的SCLS 数据来进一步绘制了算法加速比结果。
5. 核加速比：与使用算法加速比来比较所有算法的总体运行速度不同，这个度量“核加速比”是来评估CPU中的核数如何影响给定算法的运行速度。图4.12绘制了所有实现的算法的核加速比。算法ILP和IQCP下的核加速比在任意核数下近似等于1，因为它们不是并行算法。当核数小于4时，我们

的SCLS的核心加速比随着核数的增加而增加，当超过4核时，SCLS的核心加速比保持稳定，说明4核对SCLS是足够的。这一结果与Amdahl定律^[86]是一致的，即理论核加速比确定的上界限定于问题的规模大小。但是，即使核心数等于8，CoSE下的核加速比也继续增加，这是数字4的两倍。结果表明，即使是8核CPU也不能满足COSE的并行性要求。这是因为CoSE发现的冲突SRLG集包含了大量的链路，这进一步导致了大量的子问题，从而导致了较大的问题规模和计算成本。

6. 效率：与图4.12相似，随着更多的核增多效率值会降低，因为一个大的核心数会带来更多的成本来协调进程。如图4.15所示，所有算法的效率值都随着核心数的增加而降低。与图4.12中的结果一致，由于CoSE引入的子问题比我们的SCLS多，在核心数达到4之后，CoSE下的效率大于SCLS。然而，如图4.13所示我们的SCLS实现了更大的算法加速比。

全部仿真结果表明，在搜索速度较快的情况下，SCLS算法的性能优于其他算法，因为我们算法发现的冲突链路集可以方便有效的执行并行算法并且计算量小。

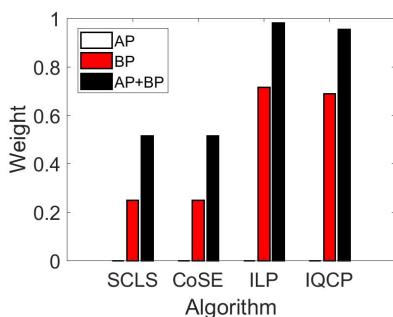


图 4.8 路径权重

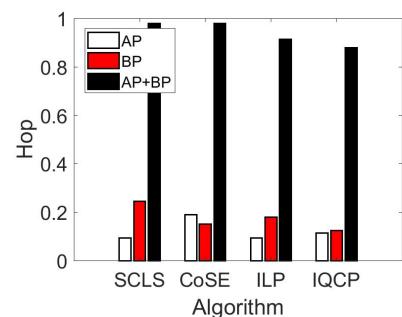


图 4.9 路径跳数

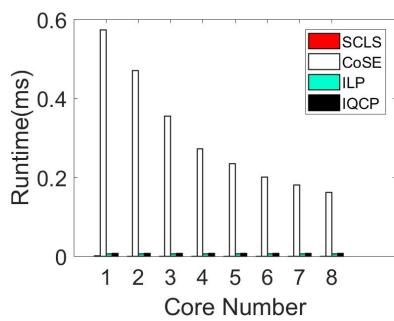


图 4.10 运行时间

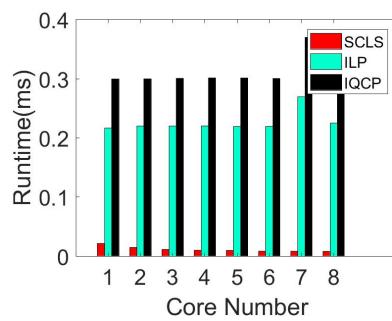


图 4.11 运行时间(无CoSE)

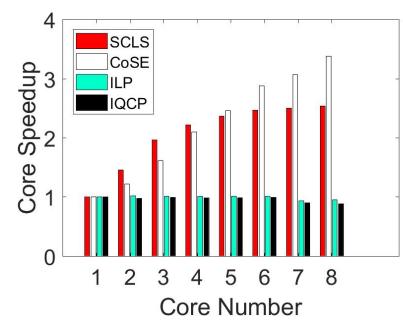


图 4.12 核加速比

4.8 小结

本章提出了一种创新的分而治之算法，该算法能在遇到陷阱问题时将原有的min-min SRLG 不相交路由问题划分为多个并行执行子问题。与现有技术相比，这样的解决方案在搜索过程可以利用现有的AP 搜索结果，并且并行执行来加快

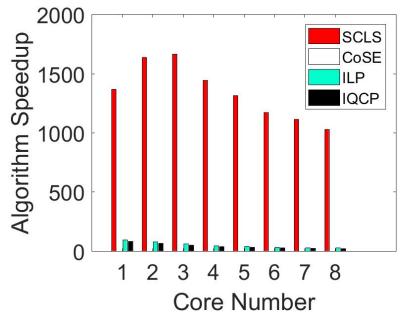


图 4.13 算法加速比

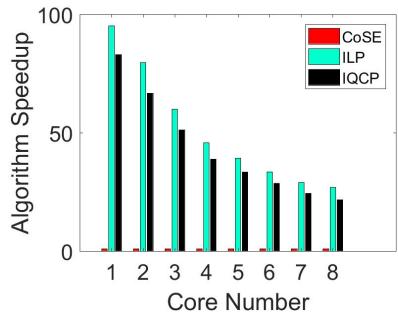


图 4.14 算法加速比(无SCLS)

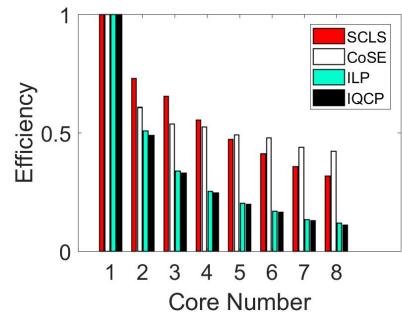


图 4.15 效率

路径查找。

第5章 虚拟网络嵌入问题

5.1 网络虚拟化概述

现如今未来互联网架构提案存在两种主要方法：一元论和多元论。首先，一元论模型中网络有一个整体架构，架构具有高度灵活性，以支持所有将会出现的新兴网络应用。在一元论网络架构方法中，每次当且仅当一个协议栈运行在物理层上。而多元论的思想是基于互联网必须同时支持多个网络，每个网络具有各自运行协议栈的能力，该协议栈是适合一项给定应用需求的。这种针对特定服务而创建特定网络的做法，简化了例如移动性、安全性或服务质量等特性新应用的部署，为提供不同服务而设计多个网络的做法比为同时处理不同服务而设计一个特殊独特网络的做法要更加简单高效。因此，多元论方法可以被理解为是一种“分而治之”方法的。

因为寻找涵盖所有可能需求的单一解决方案是非常困难的，构造支持这些需求的网络就成为了一种可行的替代方法，这是多元论架构的首要特点，多元论被广泛接受的原因是多元论方法不仅能解决所有已知的互联网问题，而且也能要求在未来网络的演进过程中依然能够持续前进的动力。此外，多元论的另一个关键优势就是其固有的后向兼容性，因为当前的互联网是并行运行的。多元论提案是基于多个网络运行在同一物理网络基础设施之上的思想，即使它们在寻址方案、报文格式和协议方面存在差异，也是如此，因为这些网络共享一个物理层，例如链路和路由器等，所以对基础介质的多种访问方式必须由同一个检测器进行编排。这个检测器就是一个特殊应用的软件，它将共享介质虚拟化后提供给运行在其上的多个网络，因此每个网络的运行就像它正在使用给定的物理资源一样，这些网络就是虚拟网络，但是为了能够共享一个物理物理，这将不得不面临网络性能的挑战。

虚拟化是以软件模拟硬件平台如服务器、网络资源或存储设备等功能，所有功能都与硬件解耦分离，并被模拟为“虚拟实例”，具有像传统硬件解决方案一样的操作能力。此外可以使用单个硬件平台来支持多个虚拟机器或设备，这些虚拟机器或设备根据需求容易改变，因此与传统的基于硬件的解决方案相比，虚拟化解决方案的可扩展性，可移植性和成本效益更高。

计算机虚拟化主要是应用在数据中心，可以在一台物理机上同时运行几台服务器。计算机虚拟化与网络虚拟化比较类似，支持网络虚拟化共享的资源是网络。将虚拟化技术应用到网络的最主要动机是尽量在每个虚拟网络也即虚拟分片内运行各自定制的协议栈。在计算机中，网络虚拟化是将软件和硬件的网络功能

和网络资源组合成基于软件的单个管理实体。

网络虚拟化包括平台虚拟化，通常与资源虚拟化相结合。另外，网络虚拟化通常分为内部虚拟化和外部虚拟化。内部虚拟化是为单个网络服务器上的软件容器提供类似网络的功能，使用伪接口或软件容器配置单个系统，以使用软件仿真物理网络，这可以通过将应用程序隔离到伪接口或单独容器来提高单个系统的效果。外部虚拟化将网络或网络的部分组合成虚拟单元，将一个或多个局域网(LAN)组合为虚拟网络，可以提高数据中心或大型网络的效率，虚拟局域网(VLAN)和网络交换机包括关键组件。使用此技术，系统管理员可以将在物理上连接到同一本地网络的系统配置为单独虚拟网络。相反，管理员可以将单独的局域网(LAN)上的系统组合成跨越大网络段的单个VLAN。

数据中心网络需要虚拟化，就像计算和存储需要虚拟功能一样，这样可以很高效方便地隔离共存的多个租户。现有的数据中心网络虚拟化经历了两个阶段：第一阶段是使用简单的VLAN和MAC地址来实现隔离；第二阶段的是使用IP覆盖网络，这两个阶段的方法都有各自对应的缺陷，首先，基于VLAN和MAC地址的方法难以配置和管理，并且将虚拟机网络与物理基础设施直接绑定会使得虚拟机的放置和挪动不够灵活。其次，IP覆盖网络对参与虚拟网络的虚拟机数量有限制，并且出现的故障问题难以调试。现在行业提出了一种新的方法VXLAN技术，利用多层标签来实现隔离网络，并且可以指定通过数据中心网络的路由，分组标签可以使用SDN控制平面，控制平面实现了数据中心交换机的OpenFlow控制。

当虚拟化应用于网络时，虚拟化可以创建硬件和软件网络资源(路由器、交换机等)的基于软件化的逻辑视图。物理网络设备简单地负责分组的转发，而虚拟网络提供智能抽象，这样有利于部署以及管理网络服务和物理网络资源。因此，网络虚拟化可以调整网络以更好地支持虚拟化环境。

在网络虚拟化环境中，多个异构的虚拟网络架构都共享物理物理网设施，这种共享是对物理物理网络设施进行隔离、抽象而形成的。不同的虚拟网络在很多方面都是不同的：服务的提供、网络拓扑、技术的使用等。网络虚拟化主要是想要极大地发挥资源共享的优势，支持未来网络发展的渐进部署，进行灵活的配置与管理。相比于现有的互联网，这种体系架构优劣分明：它提供了确保了安全性和服务质量，更灵活的资源供使用；但与此同时，我们也需要设计一套更加完善和复杂的虚拟网络管理机制来保证这种性能改善。

在虚拟网络中有三个角色：基础设施提供商、服务提供商和客户。每个角色都有不同的目标：基础设施提供商通常专注于平衡负载，最大化收入和最小化成本；服务提供商通常关注最大化收入，例如，尽可能多地支持客户或请求，他们是客户和基础设施提供商之间的中介；客户通常专注于服务质量，例如下载时间或比特率。也就是说，网络虚拟化的一个优点是虚拟网络可具有异构资源以满足

各种定制需求，虚拟化也可以提供实验环境为研究人员评估新的协议。当服务提供商生成虚拟网络请求时，它可以为每个虚拟节点和链路指定资源。因此，当将虚拟网络请求嵌入到物理网络中特定的物理节点和链路上时，其必须满足对虚拟节点和虚拟链路的约束。

5.2 网络虚拟化的技术特征

网络虚拟化也是现在网络领域的一个研究方向，总的来说，它具有如下特征^[12]:

1. 独立性：网络虚拟化平台的运行不受网络硬件的限制，而且未来出现的网络架构会更好地支持虚拟化，从而能够进一步改善云的成本代价。
2. 隔离性：物理网络和虚拟网络之间要进行安全隔离，通过共享物理物理资源，能够大大地提高计算能力，并且可以优化存储资源和网络资源的使用率。网络虚拟化平台在进行资源整合的同时还必须提供隔离，同时也要隔离虚拟网络的地址和物理网络。
3. 灵活性：网络虚拟化要遵循计算领域虚拟化的模式，在计算领域中，虚拟机能够被作为一个软件状态来进行处理，这是该领域虚拟化的一个主要特征，网络虚拟化也具有同样的灵活性。
4. 多功能性：可以很好地复制物理网络的服务模式，对于运行在任一物理环境下的任一工作负载，网络虚拟化平台必须能够提供很好的支持，此外，对于现有的网络服务，比如负载均衡，WAN优化以及ACL等也能够提供。网络虚拟化具有云的规模和性能，网络虚拟化要有能力支持大规模的网络部署环境，这样不仅可以允许存在大量的租户，而且还可以支持很多服务，如数据中心的租用等。网络虚拟化还不应该允许网络中存在故障点，并且可以支持故障切换功能和多路由功能
5. 兼容性，网络虚拟化平台要能够与任何其它类似的网络平台兼容。

5.3 虚拟网络嵌入算法

在网络虚拟化过程中，主要的实体是虚拟网络(Virtual Network, VN)。VN是顶部的主动式和被动式网络元素(网络节点和网络链路)的组合。虚拟节点通过虚拟链路互连形成虚拟网络拓扑，通过虚拟化底层物理网络(Substrate Network SN)的节点和链路资源，多个虚拟网络可以在同一的物理硬件上创建和共同管理具有广泛改变特性的网络拓扑。此外，抽象资源虚拟化机制的引入允许网络运营商以更加高度灵活和动态的方式管理和配置网络。

网络虚拟化技术通过对物理网络资源进行抽象、隔离和分配，可支持多个虚

拟网络共存于同一物理网络中。在物理网络中嵌入虚拟网络是网络虚拟化面临的主要问题是资源分配问题，通常被称为虚拟网络嵌入(Virtual Network Embedding, VNE)问题。通过动态嵌入将虚拟资源转移到物理硬件上，可以最大限度地利用现有硬件的好处。虚拟网络嵌入是实现网络虚拟化的关键，旨在研究如何合理地将租户定制的虚拟网络请求嵌入到物理物理网络中，获取满足虚拟网络服务所需要的物理网络资源，从而提高网络资源利用率和虚拟网络请求接受率。最优的动态资源分配，导致未来网络的自配置和组织，为终端用户提供定制的端到端保证服务。从QoS、经济效益、生存性到网络的安全性等不同目标，这种最优性可以根据不同的目标来计算。如图5.1所示网络虚拟化如何使用嵌入算法，以便以最佳方式在物理基础设施上分配虚拟资源。VNO 使用嵌入算法来决定从哪个虚拟资源请求。VNP 通过使用InPS底层物理网络资源来实例化它们。

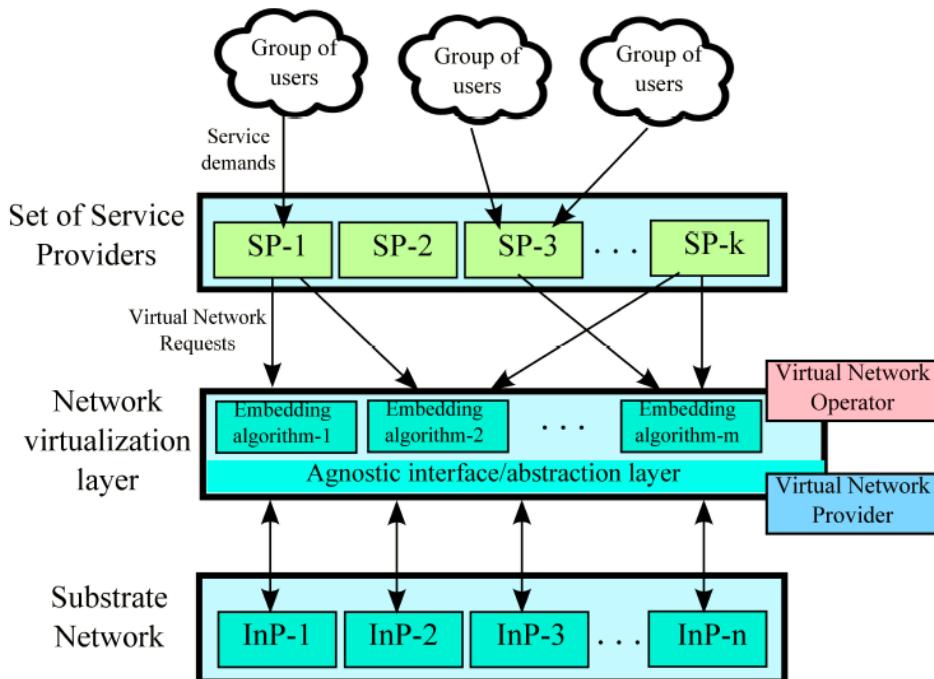


图 5.1 未来互联网的资源配置

虚拟网络嵌入问题涉及节点和链路中虚拟资源的分配。因此，它可以分为两个子问题：虚拟节点嵌入(VNoM)，其中虚拟节点必须在物理节点中分配。虚拟链路嵌入(VLiM)，其中连接这些虚拟节点的虚拟链路必须嵌入到物理网络中相应节点的路径。虚拟网络的一个节点能够被嵌入到物理网络拓扑中的任意一个物理节点，并且一条虚拟链路也可能被嵌入到物理网络的多条物理路径，因此，任意一个虚拟网络被嵌入到底层物理网络时，都会有多种嵌入方案。所以怎么把业务供应商的虚拟网络请求与物理网络进行嵌入就显得尤为重要。当虚拟网络的链路和节点有约束条件时，此时的嵌入算法属于NP难问题^[87]，即使是考虑最简单的没有约束条件的情况，在虚拟网络拓扑给定的情况下，也仍然是NP难问题^[87]。所以现在在大多的研究中如表5.1所列，基本是利用启发式算法来求解虚拟网络

嵌入问题尽量优化解决方案。

子图同构是虚拟网络嵌入相关问题的基础^[88]，没有考虑物理设备的故障和复杂的基础设施构成，也没有考虑虚拟网络请求的在线达到。在将虚拟链路嵌入到物理物理路径上时，如果允许路径分割，那么一条虚拟链路将会嵌入到物理物理网络的多条路径上，这些路径的源和目的节点相同，并且在链路嵌入时，物理链路需要有足够的带宽^[89]。在一次虚拟网络请求嵌入到物理网之后，物理网络剩余节点的计算资源和链路的带宽容量将会相应地减少。在虚拟网络的请求是已知的情况下，此时的虚拟网络嵌入问题可以转化为多路分离器(multiway separator)问题求解，是一个NP难问题^[90]。

VNE问题有多种不同变体。因此，文献中提出的所有VNE方法都可以根据它们是静态的还是动态的、集中的还是分布的、简洁的还是冗余的进行分类。这六个概念将在这里描述。

1. 静态Static vs 动态Dynamic: 在大多数现实世界中，VNE 必须作为网络问题来解决，也就是说，不会事先知道虚拟网络请求VNR 的到来。相反，虚拟网络动态得到达系统，并且可以在网络中停留任意时间。为了实际起见，VNE算法必须在VNR到达时处理它们，而不是一次加入一组VNR(离线VNE)。虽然原则上所有方法都可以在线操作，但静态VNE 方法并不考虑重新嵌入更多VNR之一的可能性，以提高SN中嵌入的性能。一些影响导致需要重新安置部分(甚至是完整的)虚拟网络：比如物理网络资源的破碎化，虚拟网络的变化，物理网络的故障等问题。

动态VNE方法试图重新配置嵌入的虚拟网络请求，以便重新组织资源分配并优化物理网络资源的使用。例如，图5.2显示了在不同时间到达并离开物理网络的三个在线嵌入的VNR。如果物理网络资源不是第一次通过重新配置已嵌入的VNR2来去碎片，则它们中的最后一个VNR3不能嵌入到物理网络。

2. 集中式Centralize vs 分布式Distribution: VNE问题可以通过集中式或分布式方式解决。每一种方法都有其各自的优点和缺点，从根本上说是不同的。

3. 简洁Concise vs 冗余Redundant: 单个物理网络实体的失败将影响嵌入到它的所有虚拟实体。因此，在部署在虚拟网络内的环境中故障敏感的应用程序，可以设置备份资源，在相应的的主要资源失败的情况下，这些资源可以用作回退资源。要做到这一点，嵌入结果本身可以是冗余的，对于节点和/或链路故障是有可生存性的。否则，如果没有冗余嵌入结果被称为“简洁”。

对VNE方法的每个类别都是相互独立的。例如，一个算法可以同时具有集中

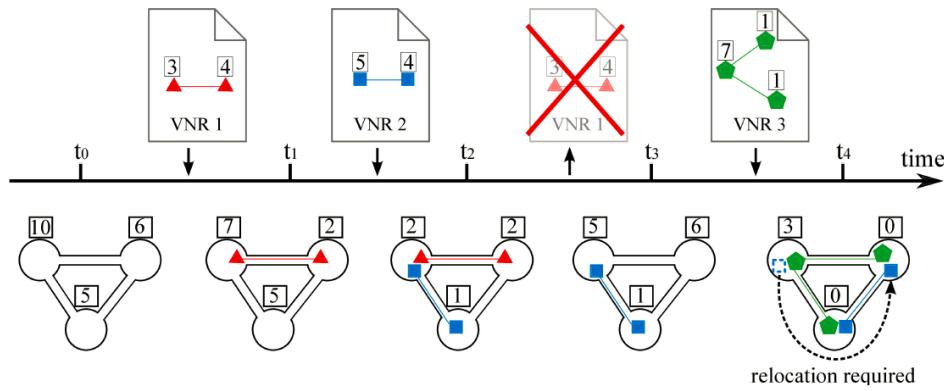


图 5.2 在线VNE算法中重新嵌入VNR

式、动态性和冗余性。在此基础上，可以导出一个泛型表示法来确定用以下语法描述：

$$[C|D]/[S|D]/[C|R]$$

第一个字符表示算法是集中的还是分布式的。同样，第二个字符表示算法是静态的还是动态的。最后，第三个字符表示该算法是简洁的还是冗余的。因此，表示为C/D/R的算法将是一种集中式的、动态的、冗余的算法。这样就可以对任何给定的算法进行快速分类，并与类似的方法进行适当的比较。

总之，如表5.1所示给出了当目前为止虚拟网络嵌入算法简明的分类总结。

表 5.1 虚拟网络嵌入算法分类

类型	论文	优化	协调	贡献
	Inführ and Raidl ^[91]	Exact	One Stage	提供延迟、位置和路由约束
	Liu et al. ^[92]	Exact	One Stage	基于对应矩阵的精确VNE
	Trinh et al. ^[93]	Exact	One Stage	具有SLA QoS保证的精确VNE问题
	Pages et al. ^[94]	Exact/Metaheuristic	One Stage	介绍了光网络中的VNE
	Lischka and Karl ^[88]	Heuristic	One Stage	提供基于子图同构的一阶段VNE
	Di et al. ^[95]	Heuristic	One Stage	改进 ^[88] 方法
	Ghazar and Saaman ^[96]	Heuristic	One Stage	介绍SN的分层管理
	Yun et al. ^[97,98]	Heuristic	One Stage	第一种在无线多跳网络中的VNE方法,介绍无线VNE的度量和可行性措施
	Chen et al. ^[99]	Heuristic	One Stage	减少物理网络资源碎片
	Yu et al. ^[100]	Heuristic	One Stage	增加协调的一阶段VNE
	Liu et al. ^[101]	Heuristic	Two Stages	基于节点邻近的改进节点与链路嵌入
C/S/C	Sheng et al. ^[102,103]	Heuristic	Two Stages	概率资源共享处理负载抖动
	Li et al. ^[104]	Heuristic	Two Stages	拓扑感知来加强VNE节点与链路嵌入
	Lu and Turner ^[105]	Heuristic	Uncoordinated	嵌入特定骨干星型VN拓扑

表 5.1 (续表)

类型	论文	优化	协调	贡献
	Yu et al.* ^[89]	Heuristic	Uncoordinated	将KSP算法 ^[73] 应用于VLiM
	Razzaq and Siraj ^[106]	Heuristic	Uncoordinated	基于VLiM 的KSP 算法
	Razzaq et al. ^[107]	Heuristic	Uncoordinated	研究瓶颈节点对VNE影响
	Nogueira et al. ^[108]	Heuristic	Uncoordinated	考虑SN 资源异质性的VNE
	Leivadeas et al.* ^[109]	Heuristic	Uncoordinated	无线网络测试VNE
	Botero et al. ^[110,111]	Heuristic	Uncoordinated	引入隐性跳数约束
	Zhu and Ammar* ^[112]	Heuristic	Uncoordinated	在SN中提供均衡的链路和节点压力
	Fajjari et al. ^[113]	Metaheuristic	One Stage	Max-Min 蚁群元启发式VNE方法
	Cheng et al. ^[114]	Metaheuristic	One Stage	拓扑感知节点排序 ^[115] 对PSO算法加速收敛的VNE元启发式方法
	Zhang et al. ^[116]	Heuristic	Uncoordinated	嵌入一个虚拟节点到多个物理网络节点上
	Di et al. ^[117]	Heuristic	One Stage	通过谨慎选择要嵌入的第一个虚拟节点来协调VNE节点和链路嵌入的冲突, p减少回溯的次数
	Abedifar and Es-hghi ^[118]	Heuristic	Uncoordinated	在光网络中引入VNE, 以尽量减少每个链路的 λ 数。
	Aris Leivadeas et al. ^[119]	Heuristic	Coordinated	考虑虚拟节点对嵌入的重要性
	Tae-Ho Lee et al. ^[120]	Heuristic	InterInP	多提供商环境下虚拟网络的聚类
	Fajjari et al. ^[121]	Heuristic	One Stage	瓶颈相邻链路的节点迁移
	Bienkowski et al. ^[122,123]	Heuristic	Two Stages	当服务访问位置更改时进行迁移
C/D/C	Zhu and Ammar* ^[112]	Heuristic	Uncoordinated	减少定期重新配置的成本
	Fan and Ammar ^[124]	Heuristic	Uncoordinated	减少VNRs 重新配置的成本
	Cai et al. ^[125]	Heuristic	Uncoordinated	基于SN演化的重构
	Shun-li and Xue-song ^[126]	Heuristic	Uncoordinated	用非最佳嵌入虚拟节点和链路以节省SN资源
	Sun et al. ^[127]	Heuristic	Uncoordinated	介绍VNRs演进的VNE 问题
D/S/C	Houidi et al. ^[128,128]	Heuristic	Uncoordinated	第一个分布式方法来解决VNE, 提出了一种管理物理节点间通信的VNE协议
	Xin et al. ^[129]	Heuristic	InterInP	介绍了用于网络云的Inter InP VNE
	Lv et al. ^[130]	Heuristic	InterInP	使用层次虚拟资源组织的InterInP VNE
	Houidi et al.* ^[131]	Exact/Metaheuristic	InterInP	VNR 被拆分成多个子VN, 在不同的InPs中分配每个子VN, 提供精确和启发式的分割方法

表 5.1 (续表)

类型	论文	优化	协调	贡献
	Leivadeas et al.* ^[132]	Heuristic	InterInP	用一个启发式集成min k-割算法和子图同构方法的图分割InterInP
D/D/C	Marquezan et al. ^[133]	Heuristic	Uncoordinated	第一种分布式动态方法，在多个VN要求更改时重新组织SN
	Houidi et al.* ^[131]	Exact	One Stage	提供ILP 精确解的第一种方法
	Zhang et al. ^[134]	Exact	One Stage	实现增强QoS嵌入的最佳可生存性解决方案，提供多种物理网络备份路径
	Botero et al. ^[135]	Exact	One Stage	介绍了能量感知的VNE
	Wang and Wolf ^[136]	Exact	One Stage	将VNR 重新定义为流量矩阵
	Shamsi ^[137-139]	Heuristic	One Stage	通过提供中间节点的备份路径来恢复链路故障
	Koslovski et al. ^[140]	Heuristic	One Stage	引入可靠性作为INP提供的服务,基于子图同构检测的可靠VNEs
	Yu et al. ^[141]	Heuristic	One Stage	引入依赖于故障的保护，并为每个区域故障提供备份解决方案。
	Lv et al. ^[142]	Heuristic	One Stage	在无线Mesh网络中引入组播VNE
	Chowdhury. ^[143,144]	Heuristic	Two Stages	基于VLiM 的多路径VNE的协调
C/S/R	Rahman et al. ^[36]	Heuristic	Two Stages	一旦出现故障，通过预先为SN链路中的备份预留带宽配额，可以将经济损失降到最低。
	Butt et al.* ^[145]	Heuristic	Two Stages	网络瓶颈资源的VNE 感知
	Yeow et al. ^[146]	Heuristic	Two Stages	引入备份资源之间的共享，减少分配给冗余的资源
	Sun et al. ^[147]	Heuristic	Two Stages	优化嵌入代价降低计算复杂度的可生存性VNE
	Yu et al. ^[32]	Heuristic	Two Stages	分析物理节点失效的可生存性VNE
	Yu et al.* ^[89]	Heuristic	Uncoordinated	介绍了VLiM 的多径方法
	Gao et al. ^[148]	Heuristic	Uncoordinated	改进方法 ^[144]
	Yang et al. ^[149]	Heuristic	Uncoordinated	在区域中划分SN以降低VNE的复杂度
	Zho et al. ^[150]	Heuristic	Uncoordinated	将一个虚拟节点嵌入到多个物理节点
	Chen et al. ^[151]	Heuristic	Uncoordinated	在线VNE过程中只考虑物理链路故障，针对被动故障的可生存性保护方法
	Yu et al. ^[152]	Heuristic	Uncoordinated	对高stress链路提供保护以防止SN链路主动故障的VNE方法
	Sun et al. ^[153]	Heuristic	Uncoordinated	向VNE引入随机带宽需求
	Lu et al. ^[154]	Heuristic	Uncoordinated	在链路中引入负载平衡
	Guo et al. ^[155]	Heuristic	Uncoordinated	主动可生存性VLiM 方法共享备份路径

表 5.1 (续表)

类型	论文	优化	协调	贡献
	Cheng et al. ^[115]	Metaheuristic	Two Stages	在VNE中引入拓扑感知
	Sheng et al. ^[156]	Metaheuristic	Two Stages	嵌入时间取决于VNR生存时间,使用模拟退火启发
	Zhang et al. ^[157]	Metaheuristic	Two Stages	引入粒子群启发算法
	Sun et al. ^[158]	Metaheuristic	Two Stages	在多数据中心环境中引入VNE
	Lv et al. ^[159]	Metaheuristic	Uncoordinated	在无线Mesh网络中引入VNE
	Leivadeas et al.* ^[132]	Heuristic	Two Stages	使用方法 ^[143] 求解任意资源池的VNE
	Masti and Raghavan ^[160]	Heuristic	Two Stages	考虑物理链路剩余容量的VNE
	Zhang et al. ^[161]	Exact/Heuristic	One Stage	恢复链路故障, 提供不相交的SN备份路径
C/D/R	Butt et al.* ^[145]	Heuristic	Two Stages	被动的虚拟链路和节点重新配置导致较小可能拒绝关键的SN区域
	Yu et al.* ^[89]	Heuristic	Uncoordinated	通过改变多径VLiM解决方案中的分裂率来重新配置嵌入
	Schaffrath et al. ^[162]	Exact	One Stage	以ILP为基础的VNE, 动态地重新配置现有嵌入
	Chen et al. ^[163]	Heuristic	Two Stages	高利用率SN节点的周期性重构
D/S/R	Chowdhury et al. ^[164]	Heuristic	InterInP	第一次提出InP VNE算法。在InP和SP收益之间进行协调。VNR在不同InPs中划分和在各自的本地InPS嵌入
D/D/D	Houidi et al. ^[165]	Heuristic	Two Stages	针对节点和链路故障的可生存性VNE

5.4 可生存性虚拟网络嵌入算法

5.4.1 共享vs非共享

可以通过在底层物理网络内集成回退资源来发挥VNE方面的可生存性。可以为可能故障失效的某些特定主节点/链路建立备份节点/链路。在任何时候, 必须保证网络拓扑的一致性, 特别是关于被定义为对故障有可生存性的资源。对于用户来说, 从故障中恢复应该是透明性的, 也就是说, 他不应该注意到网络切换到备份资源。即使在使用时间敏感的应用程序时, 用户也不应觉察到发生了错误。这尤其要求, 为了选择备份资源, 必须考虑主要实体的所有QoS要求。

备份资源本身可以是专用的, 也可以是共享的^[155]。专用意味着每个虚拟网络都可以建立完整的备份网络和备份资源。完全致力于虚拟网络, 相互独立。但

是，这是资源效率低下的，因为对于每个虚拟资源都需要获得一个已经嵌入的专用底层实体。在某些情况下，共享和重用备份资源也是可以接受的，以便减少附加备份资源在底层物理网络上的占用。通常，较高程度的重用备份资源会导致可靠性降低，反之亦然。

此外，(共享)备份资源可以预先分配(即在第一个虚拟网络嵌入请求到达之前)或“按需”(即为每个嵌入请求分配)。共享按需备份资源可以在嵌入时间分配，即每次虚拟网络请求到达^[32,146,155]时。然而，共享预分配算法在配置阶段定义了一些特定的备份资源，即在任何虚拟网络请求到达之前^[36]。

5.4.2 主动保护vs被动恢复

根据是否对租户的虚拟网络提供备份资源，可以将SVNE算法划分为两类：基于主动保护策略的算法和基于被动恢复策略的算法^[20]。

在基于主动保护策略的算法中，在嵌入虚拟网络请求前，为虚拟网络预置备份资源，以供后续故障恢复使用。该策略可以提供较高的虚拟网络可生存性保障，但是对于备份资源消耗较大。代表性算法如:FI-EVN^[32]、FD-EVN^[33] LC-SVNE^[35] 等。

在基于被动恢复策略的算法中，在嵌入虚拟网络请求前，不预置备份资源，在后续故障发生后再使用节点迁移或者现有的物理网络资源进行故障恢复。该策略的优势在于不需要额外提供虚拟网络备份资源，但是相对而言，故障恢复成功率不稳定且恢复效率较低。代表性算法如：MR-SVNE^[37]、NB-SVNE^[38] 等。

SVNE算法的两种备份策略均存在性能问题，其中主动保护策略需要提供虚拟网络全部重要节点及其关联链路的备份资源，所以该策略的虚拟网络请求接受率较低，被动恢复策略每次进行故障发生时需要使用现有网络资源进行故障恢复，故障恢复效率低下。

5.4.3 算法评价度量

可生存性虚拟网络嵌入算法的最终目标是实现网络运营收益最大化和成本消耗最小化。因此，为了实现网络运营收益最大化，需要在提高虚拟网络嵌入收益的同时，保障较高的故障恢复成功率和恢复效率以降低故障成本。

度量标准对于评估成功嵌入的质量是非常重要的。它们用于比较不同的VNE方法，并量化优化方面的进展。在本文中，有关VNE的不同的度量被分成以下四大类别：

通过对虚拟网络嵌入(VNE)算法文献的调研，选取了算法性能分析常用的四组评价指标作为实验性能参数，分别是：服务质量度量，成本相关度量，可生存性度量和其他度量。如表5.2所示概述了本文讨论的各种指标。根据应用程序的

场景，可以为大多数度量计算平均值、最大值或最小值。

表 5.2 虚拟网络嵌入的度量分类

优化目标	度量	描述
服务质量	Path Length	虚拟链路跨越的物理链路数
	Stress level	物理网络实体实现的虚拟实体的数量
	Utilization	使用的全部物理资源和物理网络资源总数的比值
	Throughput	虚拟节点之间可实现的数据速率
	Delay	一个数据包通过虚拟链路所用的时间
	Jitter	虚拟链路上数据包到达时间的变化
资源支出	Cost	嵌入全部VNR的所有物理网络资源之和
	Revenue	全部VNR所需资源之和
	Cost/Revenue	物理网络资源与虚拟网络需求的比值
	Acceptance ratio	嵌入成功的虚拟网络请求数和全部虚拟网络请求数目的比值
可生存性	Number of backups	可用备份资源的数量
	Path redundancy	多路径嵌入中路径的多样性
	Cost of survivability	维护可生存性所需的额外成本
	Recovery blocking probability	不可恢复的故障场景与所有故障情景的比值
	Number of migrations	在发生故障时必须移动的虚拟节点数
其它	Failure Recovery ratio	恢复成功的虚拟节点数和发生故障的虚拟节点数目的比值
	Runtime of the algorithm	VNE算法在嵌入一定大小数据时所需的时间
	Number of coordination messages	为了完成嵌入而必须在分布式环境中交换的消息数量
	Active substrate nodes	为了实现管理虚拟基础设施，必须打开的物理节点数

总之，如表5.3所示给出了可生存虚拟网络嵌入算法的总结。

表 5.3 可生存性虚拟网络嵌入算法比较

算法（论文）	故障类型	优化目标	处理机制
Survivable virtual network embedding ^[166]	单个物理链路故障	为基础设施提供最大限度的收入	被动，故障后(Restoration)
Shared backup network provision for virtual network embedding ^[155]	单个物理链路故障	最大化收益/VN接受率	主动，故障前(Protection)
Migration based protection for virtual infrastructure survivability for link failure ^[167]	单个物理链路故障	最小化成本	Protection
QoSMap: Achieving Quality and Resilience through Overlay Construction ^[139]	单个物理链路故障	尽量减少额外备份资源和延迟	Protection
An overlay mapping model for achieving enhanced QoS and resilience performance ^[134] ; An overlay mapping model for achieving enhanced QoS and resilience performance ^[134]	单个物理链路故障	尽量减少额外备份资源和延迟	Protection
Cost efficient design of survivable virtual infrastructure to recover from facility node failures ^[32]	单个虚拟节点故障	最小化成本	Protection
A novel two-step approach to surviving facility failures ^[168]	单个虚拟节点故障	尽量减少资源/总成本	Protection
Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures ^[141]	单个区域故障	最小化成本	Protection
Location-constrained survivable network virtualization ^[35]	单个虚拟节点故障	尽量减少资源	Protection
Designing and embedding reliable virtual infrastructures ^[146]	单个物理节点故障	尽量减少所用资源	Protection
Survivable virtual infrastructure mapping in virtualized data centers ^[169]	单个服务器故障	最小化运营成本	Protection
Adaptive virtual network provisioning ^[165]	单个节点或者链路故障	最小化通信成本	Restoration

第6章 物理节点单故障可生存性虚拟网络嵌入问题

6.1 问题描述

在这一部分中，我们首先介绍了虚拟网络嵌入的基本概念，然后提出了我们的可生存性虚拟网络嵌入问题。

6.1.1 虚拟网络嵌入

我们将虚拟网络VN表示为无向图 $G(V, E)$ ，其中 V 和 E 分别是虚拟节点和虚拟链路的集合。每个虚拟链路 e_{ij} 具有带宽需求 d_{ij} 。每个虚拟节点 v_i 具有计算容量需求 d_i 。对于虚拟节点 v_i ，需要在虚拟节点上执行的虚拟功能表示为 $f(i)$ 。如图6.1所示虚拟网络 $G(V, E)$ 具有虚拟节点集 $V = \{v_1, v_2, v_3, v_4\}$ 和虚拟边集 $E = \{e_{12}, e_{13}, e_{14}, e_{23}\}$ 。需要在这些虚拟节点上执行的虚拟函数是 $f(1) = f_1, f(2) = f_2, f(3) = f_3, f(4) = f_4$ 。

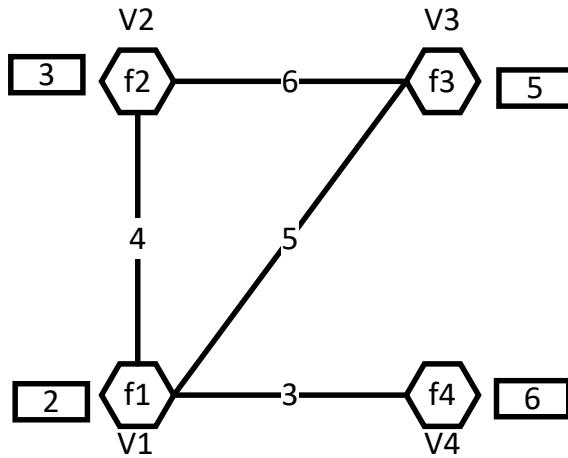


图 6.1 虚拟网络请求 $G(V, E)$

我们将底层物理网络建模为一个无向图 $G(S, L)$ ，其中 S 和 L 分别是物理节点和物理链路的集合。对于物理节点 s_i ，我们使用 $F(i)$ 和 c_i 分别表示可以在该节点上执行的一组可行的虚拟功能和可用的计算能力。每个物理链路 l_{ij} 都有可用带宽 b_{ij} 。如图6.2所示的物理网络 $G(S, L)$ ，物理节点集合 $S = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$ ，链路集合 $L = \{l_{12}, l_{13}, l_{14}, l_{15}, l_{23}, l_{25}, l_{35}, l_{36}, l_{37}, l_{47}, l_{58}\}$ ，每一个物理节点的可行的虚拟功能 $F(1) = \{f_1\}, F(2) = \{f_2, f_3\}, F(3) = \{f_3\}, F(4) = \{f_4\}, F(5) = \{f_1, f_2\}, F(6) = \{f_1, f_4\}, F(7) = \{f_2, f_3\}, F(8) = \{f_2\}$ 。

在给定VN请求 $G(V, E)$ 的情况下，虚拟网络嵌入问题的目的是将该请求映射到物理网络 $G(S, L)$ 上，同时提供所需的足够资源。一个可行的嵌入应该满足节点容量约束、链路带宽约束和功能类型约束这三个约束条件。对于一个虚拟节点 v_i ，

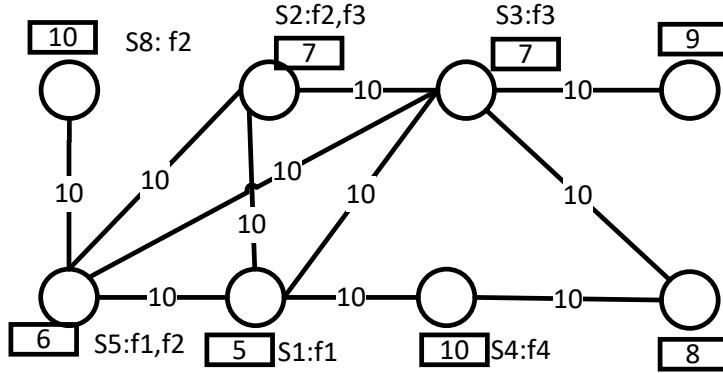


图 6.2 底层物理网络 \$G(S, L)\$

物理节点 \$s_j\$ 只在节点容量约束和功能类型两种情况满足约束条件（即 \$f_i \in F_j\$）下被映射这个虚拟节点。即节点的容量请求应满足 \$d_i \leq c_j\$ 的物理节点，虚拟节点上需要执行的虚拟功能属于物理节点 \$s_j\$ 上可以执行的虚拟功能集 \$f_i \in F_j\$。如果物理节点 \$s_j\$ 满足这两个约束，则节点映射是可行的，并且我们表示这样的映射为 \$\phi(v_i) = s_j\$。

对于在两个已经映射到两个物理节点 \$s_{i'}\$ 和 \$s_{j'}\$（即 \$\phi(v_i) = s_{i'}\$ 和 \$\phi(v_j) = s_{j'}\$）的虚拟节点之间的虚拟链路 \$e_{ij}\$，在链路带宽约束下 \$d_{ij} \leq b_{i'j'}\$（其中 \$b_{i'j'}\$ 是连接物理节点 \$s_{i'}\$ 和 \$s_{j'}\$ 的可用带宽），这条虚拟链路 \$e_{ij}\$ 可以映射到物理路径 \$p_{\phi(v_i)\phi(v_j)}\$，则表示可行的链路映射为 \$\rho(e_{ij}) = p_{\phi(v_i)\phi(v_j)}\$。

显然，为了找到一个可行的虚拟网络嵌入，我们需要找到两个映射函数 \$\phi\$ 和 \$\rho\$ 来将所有虚拟节点映射到物理节点，以及将所有虚拟链路映射到物理路径。

如图6.3所示一个可行的虚拟网络嵌入，将图6.1中的虚拟网络嵌入到图6.2中的物理网络中，其中虚拟节点 \$v_1\$ 嵌入到物理节点 \$s_1\$，虚拟节点 \$v_2\$ 嵌入到物理节点 \$s_2\$，虚拟节点 \$v_3\$ 嵌入到物理节点 \$s_3\$，虚拟节点 \$v_4\$ 嵌入到物理节点 \$s_4\$。在图6.2中，我们还展示了在这样映射之后物理网络中已经占用的和可用的资源。例如，对于物理节点 \$s_1\$，其占用的计算容量为2，可用容量为5。

给定VN请求 \$G(V, E)\$ 和物理网络 \$G(S, L)\$，对于可行映射，我们将映射的物理图表示为 \$G(\hat{S}, P)\$，其中 \$\hat{S}\$ 是容纳虚拟节点的物理节点集，其中 \$\hat{S} = \{s_{i'} : \phi(v_i) = s_{i'}, \text{for all } v_i \in V, s_{i'} \in S\}\$ 和 \$P\$ 是路径集，其中每个路径都持有一个虚拟链路 \$P = \{p_{\phi(v_i)\phi(v_j)} : \rho(e_{ij}) = p_{\phi(v_i)\phi(v_j)}, \text{for all } e_{ij} \in E\}\$。由于每个虚拟链路对应一个物理网络路径，该路径由多个物理链路组成，因此我们还将 \$G(\hat{S}, \hat{L})\$ 表示为占领的物理网络 \$\hat{L} = \{l_{pg} : l_{pg} \in p_{s_{i'}s_{j'}}, \rho(e_{ij}) = p_{\phi(v_i)\phi(v_j)}, \text{for all } e_{ij} \in E, l_{pg} \in L\}\$。

已经有了许多关于虚拟网络嵌入问题^[19]的研究；，因为本文的重点不是虚拟网络嵌入算法，我们采用了^[88] 中的算法作为基本的虚拟网络嵌入算法。

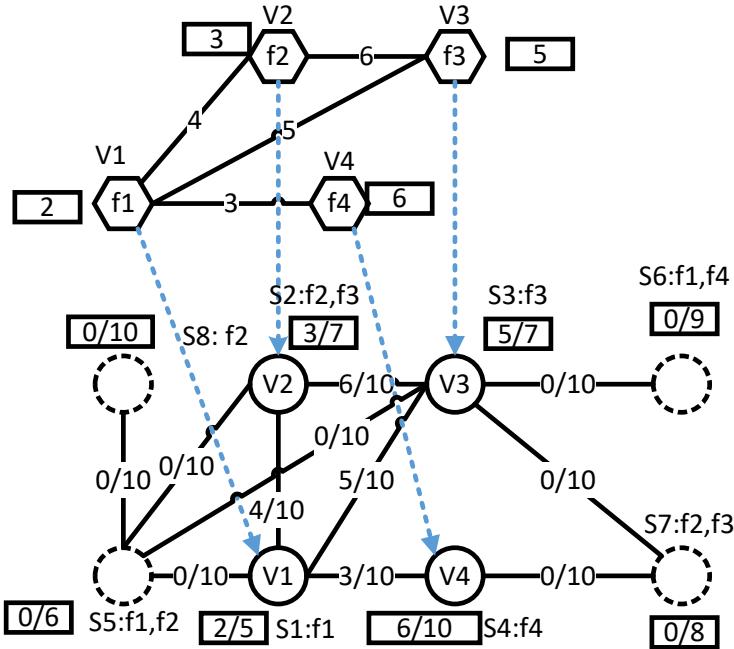


图 6.3 节点映射和链路映射的实例

6.1.2 可生存性虚拟网络嵌入

由于恶意攻击、自然灾害、无意中断电缆、计划维护、设备故障、物理节点承载的虚拟节点可能会遭受不可避免的故障。当单个或多个物理网络组件发生故障时，VN 中可能会发生故障，从而造成财务损失。一般来说，多个物理节点的同时失效是相互独立的，单个节点的失效通常发生在大多数时间^[146]。本文研究了单节点失效的可生存性虚拟网络嵌入问题。在本章中，我们将讨论如何将我们的算法扩展到多节点故障的场景中。

对于VN请求 $G(V, E)$ 和物理网络 $G(S, L)$ ，给出了其占用物理网络的可行映射，在任何一个物理节点发生故障时，增加最小备份物理资源以提供可生存性的网络服务。

节点故障不仅影响运行在失效的物理节点上的可视化服务，而且会终止通过该节点的所有通信。物理节点 $s_i \in S$ 的失效导致相邻物理链路的失效 $L_i = \{l_{ik} : k \in N(i)\}$ ，其中 $N(i)$ 是节点 s_i 的邻居节点。物理链路的失效场景可以通过增加中间节点的方法等价转换为节点故障的场景。

由于我们无法预测哪个节点将失效，即使我们知道多个节点不会同时失效，为了处理单节点故障，一个直接的方式是为VN请求中的每个虚拟节点提供专用备份资源，也称为1+1保护方案。如图6.4所示利用一个例子来说明这种直截了当的方法。在该示例中，将具有4个虚拟节点的虚拟网络映射到物理网络，其中4个物理节点参与了这样的嵌入。为了提供1 + 1 保护方案，在图中添加了4个备份节点、8个备份链路。

1+1保护方案虽然实现简单，但需要大量的备份资源。提去算法的目的是以最小的备份资源成本提供可生存性的网络服务。

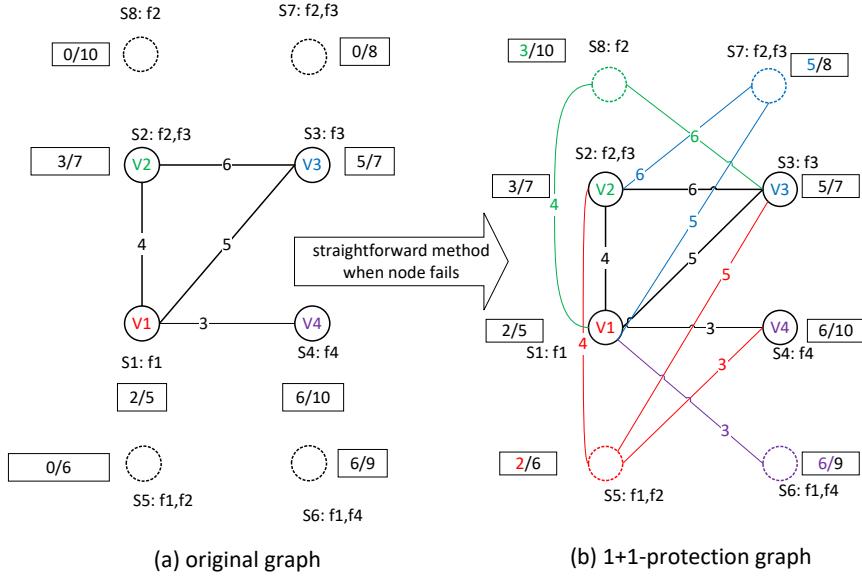


图 6.4 1+1-保护机制

6.2 星型图分解动态规划节点嵌入算法

可生存性虚拟网络嵌入需要添加备份资源，以保证当任何一个物理节点失效时，剩余的物理资源加上备份资源仍然可以支持原先的虚拟网络请求。为了便于在节点失效时找到可行的嵌入，本节首先对虚拟网络进行分解，物理网络以星型结构为基础的局部组件。在此基础上，提出了一种新的二分图，并将具有最小备份资源的可生存性虚拟网络嵌入问题转化为一个基于定义良好的二分图的虚拟星图分配问题。

6.2.1 基于星型图的分解

将虚拟网络分解为虚拟局部星型图。每个虚拟局部星型图与一个虚拟节点相关联。给出虚拟节点 v_i ，相应的虚拟局部星型图被定义为属性化单层根树，并如公式6.1所示。

$$\text{VirtualStar}(v_i) = (v_i, \phi(v_i), d_i, f_i, D_i, N_i) \quad (6.1)$$

其中， N_i 是虚拟网络中的相邻节点集，并与节点 v_i 相关联的带宽需求集 $D_i = \{d_{ij} | v_j \in N_i\}$ 。注意， $\text{VirtualStar}(v_i)$ 包括节点映射信息 $\phi(v_i)$ 。因为我们希望最小化备份资源以提供可生存性的网络服务，在节点失效前重复使用映射可能减少额外资源，使系统保持稳定的一个很好的选择。在虚拟星型图结构中，根节点和它的邻居节点存在链路，但是根节点的邻居节点之间不存在链路。

物理网络被分解为物理局部星型图。同样，每个物理局部星型图与一个物理节点相关联。给定物理节点 s_j ，对应的物理局部星型图被定义为属性化单层根树，并表示如公式6.2所示。

$$PhysicalStar(s_j) = (s_j, \phi^{-1}(s_j), c_j, F(j), \phi(N(\phi^{-1}(s_j))), a) \quad (6.2)$$

其中 $\phi^{-1}(s_j)$ 是映射到物理节点 s_j 的虚拟节点集， $N(\phi^{-1}(s_j))$ 是 $\phi^{-1}(s_j)$ 中虚拟节点的相邻节点， $\phi(N(\phi^{-1}(s_j)))$ 是承载这些邻居的物理节点， c_j 是节点容量， $F(j)$ 是 s_j 支持的虚拟功能， a 是一个1位的单比特，它的值为0或1，以指示这个物理节点是否开启和设置了虚拟机。与虚拟星型图类似，在物理星型图结构中，根节点与相邻节点之间存在边，相邻节点之间不存在边。定义在6.1和6.2中的虚拟星型图和物理星型图很好地捕获了隐藏在VN请求中的局部结构，以保持节点与其相邻节点之间的关系。基于虚拟局部星型图和物理虚拟星型图，虚拟网络和物理网络可以分解为多个组件。

6.2.2 构建星型二分图

构造了一个二分图 $G = \{V_1, V_2, E\}$ 来表示虚拟网络与物理网络之间的关系。 V_1 和 V_2 分别表示虚拟星型图和物理星型图的集合。如果 $VirtualStar(v_i)$ 的虚拟功能 f_i 能在具有 $f_j \in F_j$ 的物理节点 s_j 上执行，则在边集 E 中添加边 $e(i, j)$ ，以连接 $VirtualStar(v_i)$ 和 $PhysicalStar(s_j)$ 。

我们的目标是最小化备份资源以提供可生存性的服务。为了服务于目标，给定边 $e(i, j)$ ，我们定义了边权 $w(i, j)$ 作为备份资源成本。当节点发生故障时，将虚拟星型图(v_i)映射到物理星型图(s_j)。根据虚拟节点 v_i 映射到物理节点 s_j 节点是否失效，如公式6.3所示在两种不同的情况下定义了边权重 $w(i, j)$ 。

$$w(i, j) = \begin{cases} \alpha \sum_{\phi(v_k) \notin \phi(N(\phi^{-1}(s_j)))} d_{ik} & v_i \in \phi^{-1}(s_j), v_k \in N(i) \\ \alpha \sum_{k \in N(i)} d_{ik} + \beta M_m + \lambda c_i + \theta & v_i \notin \phi^{-1}(s_j), v_k \in N(i) \end{cases} \quad (6.3)$$

在公式(6.3)中， θ 被定义如下。

$$\theta = \begin{cases} C_s & a = 0 \\ 0 & a = 1 \end{cases} \quad (6.4)$$

在第一种情况下($v_i \in \phi^{-1}(s_j)$)，由于虚拟节点 v_i 映射到的物理节点 s_j 没有失效，节点容量需求已经得到满足，因此当将虚拟星型图(v_i)映射到物理星型图(s_j)时只需要带宽备份成本。对于每个相邻的 $v_k \in N(i)$ ，如果物理节点 $\phi(v_k) \notin \phi(N(\phi^{-1}(s_j)))$ 包含失效虚拟节点 v_k ，则应添加具有带宽 d_{ik} 的新路径作为备份资源。因此，在这种情

况下，备份成本只包括带宽成本，并表示为 $\alpha \sum_{\phi(v_k) \notin \phi(N(\phi^{-1}(s_j)))} d_{ik}$ ，其中 α 是单位带宽成本。

在第二种情况下($v_i \notin \phi^{-1}(s_j)$)，由于虚拟节点 v_i 在节点失效之前没有映射到物理节点 s_j ，因此需要将虚拟节点 v_i 迁移到物理节点 s_j 。因此，边权重 $w(i, j)$ 包括节点容量成本、路径带宽成本和迁移成本，以便在物理节点失效时将虚拟节点从物理节点迁移到另一个物理节点。此外，如果备份物理节点 s_j 之前不包含任何虚拟节点，则将虚拟节点迁移到该物理节点也会引入虚拟机启动成本，其表示为 C_s 。

如图6.5所示显示了当物理节点 s_1 失效时这种二分图的一个例子。这种二部图的边权可以用矩阵6.2.2表示。

	R_{S_1}	R_{S_2}	R_{S_3}	R_{S_4}	R_{S_5}	R_{S_6}	R_{S_7}
L_{V_1}	∞	∞	∞	∞	$C_s + M_m + (2) + 12$	$C_s + M_m + (2) + 12$	∞
L_{V_2}	∞	4	∞	∞	$C_s + M_m + (3) + 10$	∞	$C_s + M_m + (3) + 10$
L_{V_3}	∞	$M_m + (5) + 11$	5	∞	∞	∞	$C_s + M_m + (5) + 11$
L_{V_4}	∞	∞	∞	3	∞	$C_s + M_m + (6) + 3$	∞

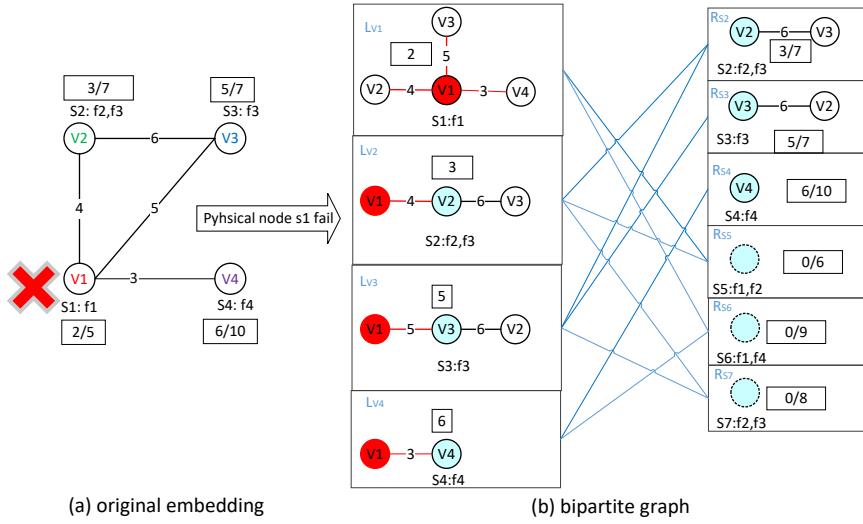
对于虚拟节点 v_i ，如果它的虚拟功能不能在具有 $f(i) \notin F(i)$ 的物理节点 s_j 中执行，则没有边连接在 V_1 中的 $VirtualStar(v_i)$ 和在 V_2 中的 $PhysicalStar(s_j)$ 。为了便于问题表述，我们将边权设为 ∞ 。例如，由于 v_1 的虚拟功能是 f_1 ，所以它不能在物理节点 s_2 中执行。因此，与 $VirtualStar(v_1)$ 和 $PhysicalStar(s_2)$ 连接时，不添加任何边，我们设置 $w(1, 2) = \infty$ 。

对于 $VirtualStar(v_2)$ ，因为 v_2 最初由 s_2 持有，但是，当物理节点 s_1 (最初持有 v_1)失效时，虚拟链路 e_{12} 无法映射到物理网络。我们应该找到一条连接 $\phi(v_2)$ 和 $\phi(v_1)$ 满足带宽需求 $d_{12} = 4$ 的新路径。因此，边权系数 $VirtualStar(v_2)$ 和 $PhysicalStar(s_2)$ 的值为4。

当节点 s_1 失效时，它直接影响虚拟节点 v_1 。由于 s_5 可以执行虚拟功能 f_1 ，我们可以添加一个边来连接 $VirtualStar(v_1)$ 和 $PhysicalStar(s_5)$ 。然而，在由于 s_5 以前没有设置虚拟机，因此还引入了虚拟机的启动成本 C_s 。因此，链路权重为 C_s (启动成本)+ M_m (迁移成本)+3(节点容量成本)+10(带宽成本)。

6.2.3 星型二分图问题定义

为了提供可生存性的服务，每个虚拟星型图(根虚拟节点和连接根节点及其相邻节点的虚拟链路)都应该映射到物理星型图。假设虚拟网络由n个虚拟节点组成，因此有n个虚拟星型图。物理网络由m个物理节点组成，从而构成m个物理星

图 6.5 当物理节点 s_1 失效时 $\text{VirtualStar}(v_i)$ 和 $\text{PhysicalStar}(s_j)$

型图。在等式6.5中，我们使用二进制位 M_{ij} 来表示第*i*个虚拟星型图是否映射到第*j*物理星型图。

$$M_{ij} = \begin{cases} 1 & \text{map } v_i \text{ to } s_j \\ 0 & \text{otherwise} \end{cases} \quad (6.5)$$

当物理节点发生故障时，为了使备份资源成本最小化，可生存性虚拟网络嵌入问题可以定义如下：

$$\begin{aligned} \min_{M_{ij}} \quad & \sum_{i=1}^n \sum_{j=1}^m M_{ij} w_{ij} \\ \text{s.t.,} \quad & \sum_{i=1}^n d_i M_{ij} \leq c_j \\ & \sum_{j=1}^m M_{ij} \leq 1 \\ & M_{ij} = \{0, 1\} \end{aligned} \quad (6.6)$$

其中 $\sum_{i=1}^n \sum_{j=1}^m M_{ij} w_{ij}$ 表示当物理节点失效时将所有虚拟星型图映射到物理星型图的总备份资源成本。在公式6.6中， $\sum_{i=1}^n d_i M_{ij} \leq c_j$ 是物理节点的容量约束，也就是说，即使允许将多个虚拟节点映射到一个物理节点，总容量需求不应大于物理节点的容量。 $\sum_{j=1}^m M_{ij} \leq 1$ 表示一个虚拟星型图只映射到一个物理星型图。

显然，公式6.6中定义的问题是一个二进制ILP问题，根据Karp的21个NP-完全问题^[170]，它是一个一般的NP-complete问题。

定理 6.1 公式6.6是一个NP-complete问题。

证明: 如果只有一个物理节点 $m=1$, 则我们的可生存性虚拟网络嵌入问题可以退化为单背包问题, 即NP-完全问题。在实际应用中, m 通常大于1, 单背包问题是可生存性虚拟网络嵌入问题的子问题, 给出的可行解在多项式时间内很容易验证, 根据计算机复杂性领域上的可归约性定理^[171], 得出我们在公式6.6中定义问题是NP-complete的结论。 \square

6.2.4 动态规划方法

虽然求解ILP公式会得到最小成本的可生存性虚拟网络嵌入, 但其指数时间复杂度使得这种方法在大型物理网络中嵌入虚拟网络是不可行的。在这一部分中, 我提出了一种基于动态规划的算法, 该算法仅具有多项式时间复杂度, 因此对于实际的网络系统是可行的方法。

为了描述基于动态规划的算法, 我们定义了 $dp[i][x_1][x_2] \dots [x_m]$ 表示以最小备份资源代价将前面 $i(0 \leq i \leq n)$ 个虚拟星型图放置到物理网络中的 m 个物理星型图中, 其容量限制为 x_1, x_2, \dots, x_m 。

第 i 个虚拟节点可以选择放置在任何一个存活的物理星型图上。设 $\theta(i, j)$ 表示已经将原先 $i-1$ 个虚拟节点最佳放置后再将第 i 个虚拟节点放置到第 j 个物理节点的备份资源成本。 $\theta(i, j)$ 表示如下:

$$\theta(i, j) = \begin{cases} dp[i-1][x_1][x_2] \dots [x_j - d_i] \dots [x_m] + w_{ij} & (x_j \geq d_i, f_i \in F_j) \\ \infty & otherwise \end{cases} \quad (6.7)$$

在等式6.7中, 如果物理节点 x_j 的容量限制大于容量需求 d_i 并且在物理节点 $f_i \in F_j$ 能执行虚拟功能 f_i , 则 $\theta(i, j)$ 是已经放置前面 $i-1$ 个最佳虚拟星型图的代价和(即 $dp[i-1][x_1 - d_i][x_2] \dots [x_m]$)和将虚拟星型图(v_i)映射到物理星型图(s_1)的成本(即 w_{i1})。基于 $\theta(i, j)$, $dp[i][x_1][x_2] \dots [x_m]$ 可通过以下动态规划函数计算。

$$dp[i][x_1][x_2] \dots [x_m] = \min\{\theta(i, 1), \theta(i, 2), \dots, \theta(i, j), \dots, \theta(i, m)\} \quad (6.8)$$

基于动态规划的算法如伪码6.2.4所示。我们还以如图6.6所示中的一个例子来说明该算法。

在本例中为了清晰地表示, 需要将三个虚拟星型图放置到两个可用的物理星型图以实现最小的备份资源成本。物理节点下的可用容量分别为 $c_1 = 5$ 和 $c_2 = 4$ 。这三个虚拟星型图的容量需求分别为 $d_1 = 2$, $d_2 = 1$ 和 $d_3 = 2$ 。在这些虚拟星型图中需要执行的虚拟功能是 $f(1) = f_1$, $f(2) = f_1$, $f(3) = f_2$ 。这两个物理节点支持的虚拟功能是 $F(1) = \{f_1, f_2\}$ 和 $F(2) = \{f_1\}$ 。

如图6.6所示, x 和 y 轴分别表示物理星型图 s_1 和 s_2 的容量极限。虚拟星型图与物理星型图的连接边的权重 $w_{11} = 1$, $w_{12} = 2$, $w_{21} = 3$, $w_{22} = 1$, $w_{31} = 1$, $w_{32} = \infty$ 。

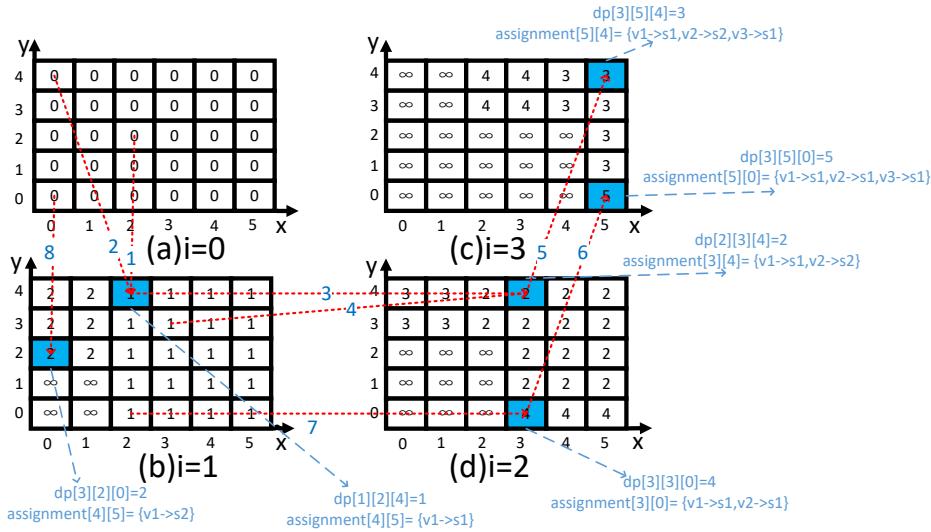


图 6.6 动态规划方法的演示

算法 6.2.1 基于动态规划方法二分星型图匹配算法

Input: $dp[i][x_1][x_2] \dots [x_m] = 0 (1 \leq i \leq n, 0 \leq x_1 \leq c_1, 0 \leq x_2 \leq c_2, \dots, 0 \leq x_m \leq c_m)$ 首先被定义成无穷大 ∞ , $dp[0][x_1][x_2] \dots [x_m] = 0 (0 \leq x_1 \leq c_1, 0 \leq x_2 \leq c_2, \dots, 0 \leq x_m \leq c_m)$, m : 物理节点的数量。 $M[x_1][x_2] \dots [x_m] = \mathbf{0}_{n \times m}$ 每个节点的映射矩阵

Output: 当节点容量为 c_1, c_2, \dots, c_m 时最小代价的节点映射

```

1: for all  $i$  such that  $1 \leq i \leq n$  do
2:   for all  $x_1, x_2, \dots, x_m$  such that  $c_1 \geq x_1 \geq d_i, c_2 \geq x_2 \geq d_i, c_3 \geq x_3 \geq d_i, \dots, c_m \geq x_m \geq d_i$  do
3:      $dp[i][x_1][x_2] \dots [x_m] = \min\{\theta(i, 1), \theta(i, 2), \dots, \theta(i, j), \dots, \theta(i, m)\}$ 
4:      $j' = \arg \min\{\theta(i, 1), \theta(i, 2), \dots, \theta(i, j), \dots, \theta(i, m)\}, .$ 
5:      $M[x_1][x_2] \dots [x_m] = M[x_1][x_2] \dots [x_{j'} - d_i] \dots [x_m]$ 
6:      $M[x_1][x_2] \dots [x_m]_{i, j'} = 1$ 
7:   end for
8: end for
9: return  $dp[i][c_1][c_2] \dots [c_m]$  and  $M[c_1][c_2] \dots [c_m]$ 
```

最初，在图6.6(a)中，由于没有对任何物理星型图放置虚拟星型图，所有容量限制情况下的备份成本($x_1=0,1,2,3,4$ 和 $x_2=0,1,2,3,4,5$)都是0。特别是，即使设置了4和5的容量限制， $dp[0][4][5]=0$ 。如图6.6(b)所示，当将容量要求为 $d_1 = 1$ 的第一个虚拟节点 v_1 放置到这两个物理节点时，可以在两个物理节点中执行 f_1 。因此，我们有

$$dp[1][x_1][x_2] = \min\{\theta(1, 1), \theta(1, 2)\} \quad (6.9)$$

特别是，如果这两个物理节点的容量极限分别为 $x_1=2$ 和 $x_2=0$ ，则 $dp[1][2][0] = dp[0][0][0] + w_{11} = 2$ 。如果这两个物理节点的容量极限分别为 $x_1=2$ 和 $x_2=4$ ，则 $\theta(1, 1) = dp[0][0][2] + w_{11}$ 和 $\theta(1, 2) = \infty$ ，从而 $dp[1][2][4] = \min\{dp[0][0][4] + w_{11}, dp[0][2][2] + w_{12}\} = 1$ 。

同样，当将容量要求 $d_2 = 2$ 的第二个虚拟节点放置到这两个物理节点时，所有容量限制下的成本结果如图6.6(d)所示。因为 f_2 可以在两个物理节点中执行，

因此 v_2 可以放置在两个物理节点中，所以我们有

$$dp[2][x_1][x_2] = \min\{\theta(2, 1), \theta(2, 2)\} \quad (6.10)$$

特别是，如果这两个物理节点的容量极限分别为 $x_1=3$ 和 $x_2=4$ ，则 $dp[2][3][4] = \min\{dp[1][2][4] + w(2, 1), dp[1][3][3] + w(2, 2)\} = 2$ 。如果这两个物理节点的容量极限分别为 $x_1=3$ 和 $x_2=0$ ，则 $dp[2][3][0] = dp[1][2][0] + w(2, 1) = 4$ 。

如图6.6(d)显示了将容量要求为 $d_3 = 1$ 的第三个虚拟节点放置到这两个物理节点时的最小资源成本结果。由于 f_3 只能在物理节点 s_1 中执行，所以我们有 $\theta(3, 2) = \infty$ 。特别是，如果这两个物理节点的容量极限分别为 $x_1=5$ 和 $x_2=0$ ，则 $dp[3][5][0] = dp[2][3][0] + w(3, 1) = 4$ 。由于这两个物理节点的节点能力分别为5和4，我们得到了 $dp[3][5][4] = dp[2][3][4] + w(3, 1) = 3$ ，如图6.6(d)所示，在 $dp[3][5][4]=3$ 处得到了最佳位置，节点映射为 $v_1 \rightarrow s_1, v_2 \rightarrow s_2, v_3 \rightarrow s_1$ 。

例如，如图6.5和等式6.2.2所示，其最小资源成本结果的最优节点映射矩阵如矩阵6.11所示。

$$M_{ij} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (6.11)$$

6.3 添加备份资源的略

如果一个节点失败，则在应用算法6.2.4计算最小额外资源后，添加新的物理节点和物理路径来维护网络可生存性服务。如果不包含任何虚拟节点的物理节点在节点失效前就已经添加为备份节点，我们应该将这个物理节点设置为 $a=1$ ，以指示该物理节点已经容纳过虚拟节点。

由于我们的可生存性虚拟网络嵌入问题是为了在任何一个节点失效时最小化增加的备份资源，而不是给定的一个节点故障，我们应该逐一测试每个节点的故障，并添加足够的备份资源。作为备份资源只需要当节点失效时，备份资源应在不同节点失效时共享。如果我们直接应用公式6.3中定义的成本来计算另一个物理节点失效时的备份资源，因为公式6.3中定义的成本不考虑备份资源共享，则会导致重复添加备份资源的问题。

为了解决这一问题，我们将 $M(j)$ 加入到物理星型结构中，表示在节点失效的

情况下迁移到物理节点 s_j 中的虚拟节点集。

$$PhysicalStar(s_j) = (s_j, \phi^{-1}(s_j), c_j, F(j), \phi(N(\phi^{-1}(s_j))), a, M(j)) \quad (6.12)$$

由于备份资源应该只添加一次，为了便于表示该约束，我们定义了以下函数。

$$\mu(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases} \quad (6.13)$$

当我们测试另一个节点失效时，而不是等式6.3中定义的代价，我们定义了一个新的代价函数，考虑到不同节点失败时的备份资源共享。

如等式6.14所示，资源成本是根据两种情况设置的。如果虚拟节点 v_i 最初由物理服务器 s_j 持有，或者 v_i 被迁移到物理服务器，即 $v_i \in (\phi^{-1}(s_j) \cup M(j))$ ，当它的邻居 $v_k \in N(i) \setminus \phi^{-1}(s_j)$ 失效时，路径带宽成本为 $\sum_{k \in N(i) \cap N(i')} \mu(d_{ik} - d_{i'k}) + \sum_{k \in (N(i) - N(i'))} d_{ik}$ 。

否则，在原先节点失效的测试前这个虚拟节点 v_i 没有被一个物理节点持有和迁移到其他物理节点，除了这个带宽成本为 $\sum_{k \in N(i) \cap N(i')} \mu(d_{ik} - d_{i'k}) + \sum_{k \in (N(i) - N(i'))} d_{ik}$ ，节点的容量成本为 $\lambda * \mu(d_i - \max_{v_{i'}}(d_{i'}))$ ，虚拟服务器迁移的代价为 $\beta * M_m$ 和 θ 是从映射虚拟星型图 v_i 到物理星型图 s_j 的代价。如 $\mu(d_i - \max_{v_{i'}}(d_{i'}))$ 而言，如果备份容量已分配的最大值 $\max_{v_{i'}}(d_{i'})$ 大于当前映射(即 d_i)，不需要任何资源，否则，备份资源还缺少 $d_i - \max_{v_{i'}}(d_{i'})$ 。

对于 $VirtualStar(v_2)$ ，由于 v_3 最初由 s_3 持有，但是当物理节点 s_1 失效时，我们可以将虚拟节点 v_3 迁移到物理节点 s_2 ，因为 s_2 是以前已经安装过虚拟机，因此，成本包括虚拟链路带宽成本11，节点容量成本5，虚拟机迁移成本 M_m 。

基于节6.2.4的方法，在物理节点 s_1 失效时得到最佳节点映射： $v_1 \rightarrow s_5$ ， $v_2 \rightarrow s_2$ ， $v_3 \rightarrow s_3$ ， $v_4 \rightarrow s_4$ 。物理节点 s_5 开始为虚拟节点 v_1 建立和应用2个节点计算资源，并分别找到带宽约束为4、5、3的对应的虚拟链路 (v_1, v_2) ， (v_1, v_3) ， (v_1, v_4) 的三个物理路径。

如图6.7展示了二分图的一个例子，物理节点 s_2 在物理节点 s_1 失效之后再失

$$w(i, j) = \begin{cases} \sum_{k \in N(i) \cap N(i')} \mu(d_{ik} - d_{i'k}) + \sum_{k \in (N(i) - N(i'))} d_{ik} & v_i \in (\phi^{-1}(s_j) \cup M(j)), v_{i'} \in M(j), k \notin \phi(\phi^{-1}(s_j)) \\ \sum_{k \in N(i) \cap N(i')} \mu(d_{ik} - d_{i'k}) + \sum_{k \in (N(i) - N(i'))} d_{ik} + \lambda f(d_i - \max_{v_{i'}}(d_{i'})) + \beta M_m + \theta & v_i \notin (\phi^{-1}(s_j) \cup M(j)), v_{i'} \in M(j) \end{cases} \quad (6.14)$$

效。这个二分图的边权矩阵可以用矩阵6.3表示。

(a) original embedding

R_{S_1}	R_{S_2}	R_{S_3}	R_{S_4}	R_{S_5}	R_{S_6}	R_{S_7}	
L_{V_1}	4	∞	∞	∞	M_m	$C_s + M_m + (2) + 12$	∞
L_{V_2}	∞	∞	∞	∞	$M_m + (1) + 5$	∞	$C_s + M_m + (3) + 10$
L_{V_3}	∞	∞	5	∞	∞	∞	$C_s + M_m + (5) + 11$
L_{V_4}	∞	∞	∞	3	∞	$C_s + M_m + (6) + 3$	∞

(b) bipartite graph

图 6.7 当物理节点 s_2 失效时 VirtualStar(v_i) 和 PhysicalStar(s_j)

6.4 多节点故障情形

如图6.8所示给出了当物理节点 s_1 和 s_2 同时失效时星型二分图的一个例子。这种二分图的边权矩阵可以用矩阵6.4表示。

	R_{S_1}	R_{S_2}	R_{S_3}	R_{S_4}	R_{S_5}	R_{S_6}	R_{S_7}
L_{V_1}	∞	∞	∞	∞	$C_s + M_m + (2) + 12$	$C_s + M_m + (2) + 12$	∞
L_{V_2}	∞	∞	∞	∞	$C_s + M_m + (3) + 10$	∞	$C_s + M_m + (3) + 10$
L_{V_3}	∞	∞	5	∞	∞	∞	$C_s + M_m + (5) + 11$
L_{V_4}	∞	∞	∞	3	∞	$C_s + M_m + (6) + 3$	∞

6.5 完整算法步骤

在这一部分中，我们首先给出了SeVN图的增广资源分配过程。下面将详述更多细节。

此外，对于节点嵌入和虚拟链路嵌入，由于单个节点失效时并不是所有的虚

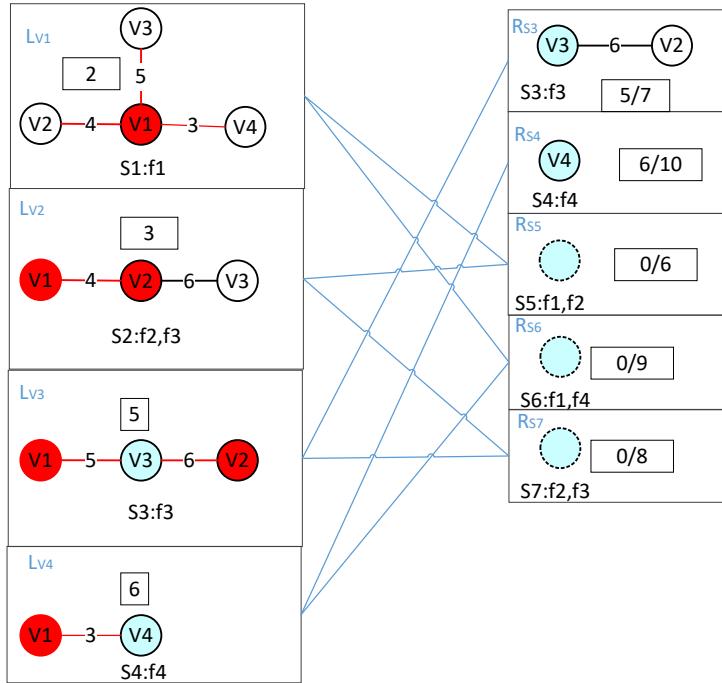


图 6.8 当物理节点 s_1 和 s_2 同时失效时VirtualStar(v_i)和PhysicalStar(s_j)

拟链路或它们的带宽都会同时使用，所以如果将虚拟链路嵌入在同一底层物理链路上，则一些虚拟链路可以共享底层物理资源。减少总底层物理带宽所需。简单地说，底层物理链路对应不同虚拟网络节点失效时可生存性请求的备份带宽可以相互共享。

6.5.1 将增广的备份资源嵌入到底层网络中

从节6.2.4中，我得到了节点 v_i 失效的节点映射和链路映射关系，和计算出每个节点需要重新分配的计算量，以及每个链路在底层物理网络SN中应该重新分配的带宽，如图6.9所示。分别在物理节点 $s_1, s_2, s_3, s_4, s_5, s_6, s_7$ 中增加计算资源0、2、0、0、3、6、0。查找带宽分别为1, 4, 6, 3, 6, 6与物理路径 $p_{s_1s_2}, p_{s_1s_3}, p_{s_1s_5}, p_{s_1s_6}, p_{s_2s_5}$ 对应的五条路径。尽管虚拟网络嵌入算法很多，但相对于我们的算法，节点映射阶段已经完成，接下来的步骤描述了链路映射阶段。我们使用标准最短路径算法Dijkstra 算法^[172]来请求底层物理节点的扩展路径，并为这种虚拟网络生存的请求的部分带宽重新分配给这些路径。一些研究^[89]集中在路径嵌入和路径分割嵌入，以实现高的链路利用率、链路压力和其他目标，但本文没有重点讨论链路映射问题。

最后，迭代的重复图分解过程和多背包问题求解过程，处理每个嵌入虚拟节点的物理节点失效情况，重新构造增强图，以保证虚拟网络的生存请求。如图6.10所示，连续使用不同颜色标记SeVN的部分增广资源并且连续显示每个节点故障。值得注意的是，部分增广资源的每一步都是基于前一步的资源增强。

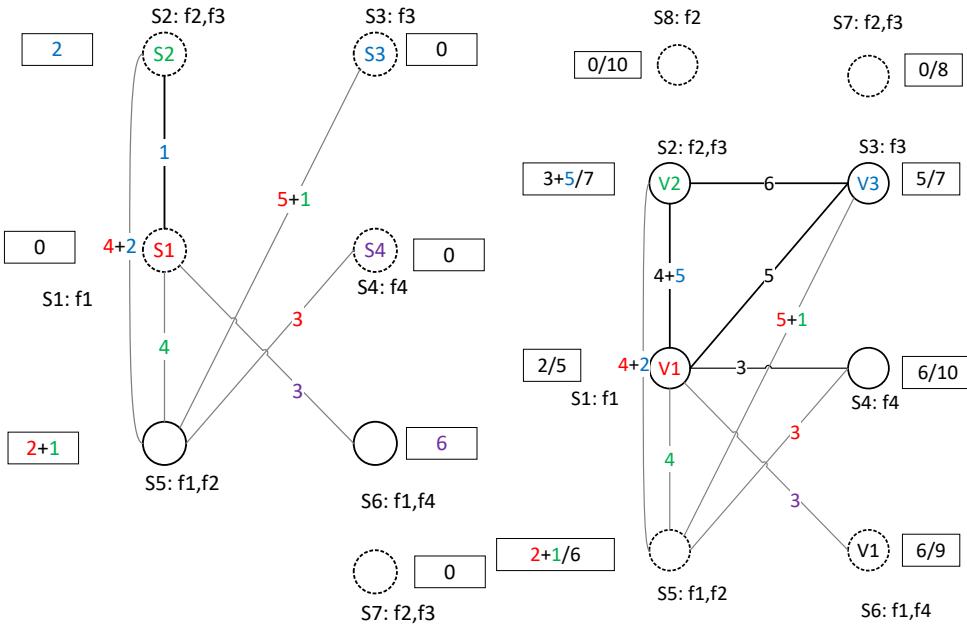


图 6.9 增广的备份资源图

图 6.10 节点 s_4 失效实例

6.5.2 算法步骤

在本节中，我们描述了SeVN问题完整的算法步骤，如伪码6.5.2所示
算法 6.5.1 可生存性虚拟网络嵌入算法

Input: $G(V, E)$: 虚拟网络请求; $G(S, L)$: 物理网络。

Output: 生成SeVN并增加资源将SeVN嵌入到底层物理网络中。

- 1: 将虚拟网络 G^V 嵌入^[92]到底层网络 G^S 中
- 2: 从与此VN嵌入请求对应的SN中提取嵌入的虚拟网络 $eVNG(\hat{S}, \hat{L})$
- 3: **for all** v_i 并且 $v_i \in G(\hat{S}, \hat{L})$ **do**
- 4: 从 图 $G(\hat{S}, \hat{L})$ 中 分 解 $eVNG(\hat{S}, \hat{L})$ 成 两 个 星 型 结 构 集 $VirtualStar(v_i)$ 和 $PhysicalStar(s_j)$
- 5: 构建 items 根据星型结构集 $VirtualStar(v_i)$.
- 6: 构建 knapsacks 根据星型结构集 $PhysicalStar(s_j)$
- 7: 构建边权值矩阵 6.14.
- 8: 根据节 6.2.4 所描述的动态规划方法解决多背包问题
- 9: 添加新节点，连接新的边，重新分配节点计算资源和链路的带宽资源到 $G(\hat{S}, \hat{L})$ 中构建新的 $G(\hat{S}, \hat{L})$
- 10: **end for**
- 11: 从 $G(\hat{S}, \hat{L})$ 中嵌入增广资源到物理网络 $G(S, L)$

6.6 时间复杂度

对SeVN请求问题分析，该问题是优化领域中的二次规划问题，而SeVN问题解的实质是一系列排列矩阵，决定了每个节点失效后的迁移重分配过程。虚拟网络 $|V|$ 节点全部都依次失效，每个节点的迭代时间复杂度为 $O[|V| * |S| * \prod_{i=1}^{|S|} c^i]$ 。因此，其总体计算复杂度为 $O[|V| * |S| * |V| * \prod_{i=1}^{|S|} c^i]$ ，其中 $|V|$ 为失效节点数。

6.7 仿真配置与评价指标

首先描述了仿真配置，然后给出了评价指标。

6.7.1 实验环境

对于任何VN请求，VN节点的数目是由3到10之间的均匀分布随机得到的，每一对虚拟节点都是以概率0.5随机连接的。VN节点的计算需求从1到5之间随机分布，并且VN上的带宽从1到10之间随机分布。

VN请求的到达服从泊松过程(平均每1时间单位请求15次)。请求的持续时间服从指数分布，平均100个时间单位，高的请求率和较长的租用时间保证了物理基础设施的高利用率。

根据^[141]，即 $\lambda/\alpha = 3$ ，节点的计算资源与链路的带宽资源的相对成本为3。使用的SN拓扑是SNDlib拓扑数据^[173]如表6.1所示。底层物理节点(链路)的计算(带宽)资源都是整数的。分别分布在10至20(50至100)之间。为了对设备节点失效场景进行建模，我们选择了底层物理网络中的所有底层物理设备节点逐个失效，并统计了每40个时间单元的迁移频率。

表 6.1 SNDlib拓扑数据

数据集	节点	链路
cost266	37	57
geant	22	36
germany50	50	88
giul39	39	172
janos-us-ca	39	122
janos-us	26	84
nobel-eu	28	41
norway	27	51
pioro40	40	89
ta1	24	55
ta2	65	108
zib54	54	81

为了进行性能比较，除了我提出的方案(由STAR命名)之外，我还实现了其他可生存的算法RVN^[146]如下：

1. *RVN*：算法将备份节点与关键节点之间、备份节点与备份节点之间的连接起来，而不连接关键节点与关键节点之间，关键节点表示嵌入物理节点中的虚拟节点，备份节点表示的增广节点。
2. *oVN*：当与虚拟节点映射的底层物理上的每个节点失效时，一种简单的方

法是增加一个新的物理节点，用于迁移失败的虚拟节点，并将新的链路与失败的物理节点连接起来。

此外，我们还与虚拟网络嵌入算法^[92]进行了比较，后者是虚拟网络请求第一次到达时的嵌入方法，其中VN没有可生存性要求，虚拟网络嵌入算法请求(标记为**bVN**)作为衡量可生存性所消耗的扩展资源量的标准算法，在实验评估图中标记为“algorithm shared”或者“algorithm Noshared”。我们假设当一个虚拟网络嵌入到底层物理网络中时，对于**bVN**算法来说，就需要有成功的可生存性保证。

基于底层物理网络的资源，无论是共享的还是非共享的(专用的)^[105]，我们在资源分配模式上独立地实现了这两种情况。每个虚拟网络请求的相应的备份需求相应的底层物理资源可以相互共享，并具有可生存性保证。“专用的”意味着对每个虚拟网络是一个完整的备份网络，备份资源完全用到虚拟网络中，相互独立。但是这是资源效率低下的，因为对于每个获得嵌入式的虚拟资源，都需要一个专用的底层物理实体。在某些情况下，共享和重用备份资源也是可以接受的，以便减少附加备份资源在底层物理网络上的占用。通常，较高程度的重用备份资源会导致可靠性降低，反之亦然。

每个虚拟网络的请求都存在一个生命周期，统计记录虚拟或底层物理板网络在实时情况下的性能指标。由于每个算法的性能度量的实验数据波动，我们将每个虚拟或底层网络性能度量的实验数据逐个记录下来，然后通过成功接受的虚拟网络嵌入或可生存的嵌入式虚拟网络作为分母来对实验数据进行平均。

所有仿真都运行在服务器上，服务器配置为Intel(R) Xeon(R) CPU E5-2620 2.00GHz (24 Cores) 和32.00GB RAM。

6.7.2 评价指标

本节介绍了启发式算法在平均接受率、活动节点和其他度量方面的不同度量^[19]如下所述。

1. 接受率：与阻塞概率的概念相反。虚拟网络请求的比率是成功嵌入物理网络的虚拟网络请求数除以虚拟网络嵌入请求数，表示为 A_{eVN}/A_{VN} 。为可生存性保证而成功增广和嵌入到物理网络的虚拟网络请求数表示为 A_{SeVN}/A_{VN} ，当虚拟网络嵌入到底层网络中时该虚拟网络成功可生存性保护的比率被表示为 A_{SeVN}/A_{eVN} 。
2. 活动节点：表示底层物理网络开启的节点数和可生存性虚拟网络的节点数，这种度量在节能的VNE算法中特别有用。
3. 链路或路径长度：链路长度度量虚拟网络中链路的数量，和路径长度度量是两个互连虚拟节点映射到两个底层节点之间的物理链路数量。
4. 成本、收入和成本/收益：成本：使用的物理网络节点计算能力或链路带宽

资源。收益：虚拟网络的节点计算能力或链路带宽需求。成本/收益：这个比率表示虚拟化开销。

A_{VN} 表示单位时间 t 时虚拟网络请求数。如图6.11所示，嵌入到物理网络中的虚拟网络请求的成功数目表示为 A_{eVN} ，以及成功增广资源并且嵌入到物理网络中表示为 A_{SeVN} 。

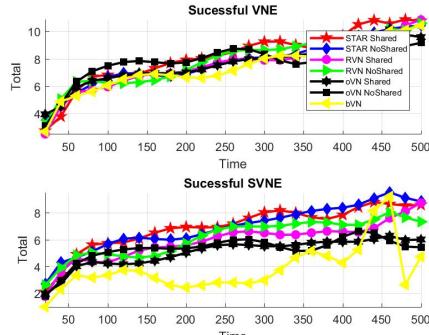


图 6.11 成功的VNE请求数和成功的SVNE请求数

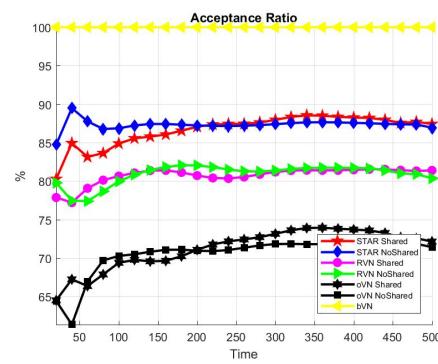


图 6.12 接受率

考虑所有虚拟节点的情况下，评估了SeVN问题启发式方法STAR在分配资源时的性能。我们特别关注物理网络使用的资源和虚拟网络请求的资源情况，以及可生存虚拟网络请求的可接受率。

6.8 算法性能评估及比较

6.8.1 接受率

在这一部分中，如图6.12所示比较了SeVN问题不同算法可生存性虚拟请求的接受率。此外， bVN 是所有其它算法前期的虚拟嵌入算法，在可生存性增广嵌入过程中，我们假设计算法 bVN 没有可生存性资源增广而已经达到可生存性功能需求。以 bVN 作为标准比较算法来度量其它算法的性能，特别是可生存性需求导致的对底层物理网络额外资源消耗量。如图6.12所示，当系统中运行到250单元时间后接受率相对稳定和保持一致，STAR 算法比RVN 算法有更高的接受率，接受率提高了近4%，这是因为STAR算法当可生存性需求时资源增广分配过程中比RVN 算法消耗更少的底层资源。STAR-NoShared 比STAR-Shared的接受率低，直观地说，这是因为STAR-NoShared的为了可生存性的保证而需要更多的资源，备份资源之间都相互不共享，为了可生存性需求而资源增广都需要申请大量的额外资源，由于冗余资源消耗的原因，接受率至少比STAR-Shared 损失了6%。

6.8.2 活动节点

为了承载所有虚拟网络而需要使用的逻辑节点数如图6.13所示，STAR 算法比其它算法使用更少的虚拟节点数这将使得STAR 算法比其它算法消耗更少的底

层资源。

如图6.14所示，为了承载所有虚拟网络而需要启动和被嵌入虚拟节点的底层物理网络节点数，底层物理网络需要启动节点数与底层物理网络上承接的虚拟网络路径的平均长度有关，因为使用额外的节点来转发端节点之间的通信数据，因此，选择未启动的节点来转发数据的概率增加了。在能源效率方面，嵌入所需的底层物理网络节点数可以粗略估计能源消耗。

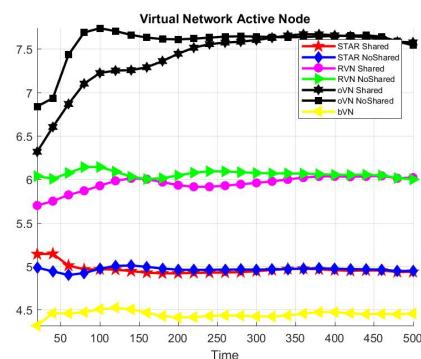


图 6.13 虚拟网络的平均虚拟节点数

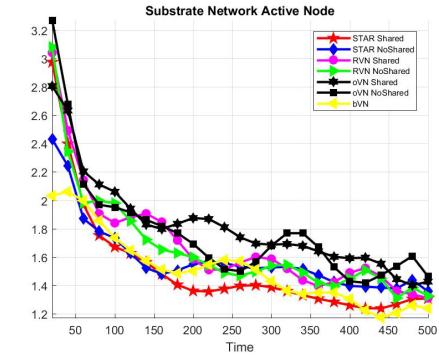


图 6.14 底层物理网络平均启动的节点数

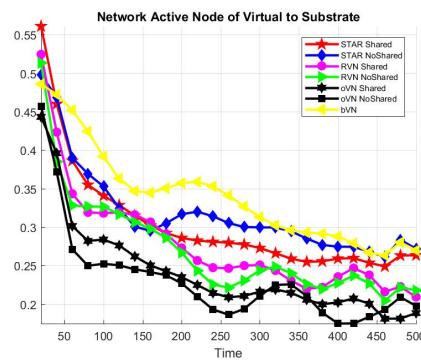


图 6.15 底层物理网络启动的节点数与虚拟网络虚拟节点数之比

底层物理网络启动的节点数与虚拟网络的虚拟节点数之比如图6.15所示。

6.8.3 链路与路径长度

为了承载所有虚拟网络而需要使用的虚拟链路数如图6.16所示，虚拟网络链路对应的物理路径越长，就需要为嵌入虚拟链路预留更多的资源。由于作为路径的一部分所以每个底层物理网络节点(接收节点除外)都需要一些时间来转发通过该路径发送的包，因此服务质量受路径长度的影响。通常，包延迟随着路径长度的增加而增加。

为了承载所有虚拟网络而需要使用的底层物理网络链路数如图6.17所示。物理网络链路数与虚拟网络链路数之比如图6.18所示。我提出的算法STAR获得的虚拟网络链路数和物理网络链路数与算法bVN最接近，算法STAR比其它算法RVN和oVN具有更优的虚拟网络链路数和物理网络链路数。

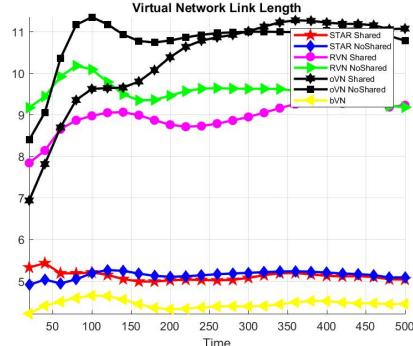


图 6.16 虚拟网络链路数

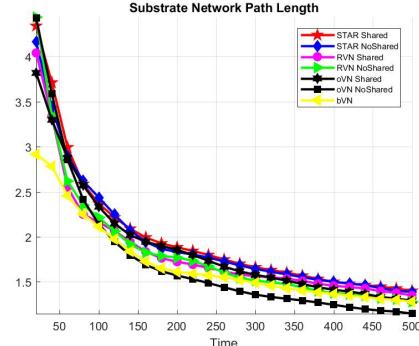


图 6.17 物理网络链路数

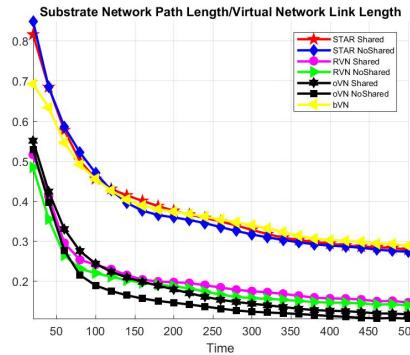


图 6.18 物理网络链路数与虚拟网络链路数之比

6.8.4 成本、收入和成本/收入比

在本工作中，成本是为满足SEVN可生存性需求而消耗的底层物理资源(即所有底层物理设备节点的节点计算能力、所有光纤链路上的链路带宽)。底层物理网络消耗资源的平均成本如图6.19所示。我提出的算法*STAR*获得的成本和收益

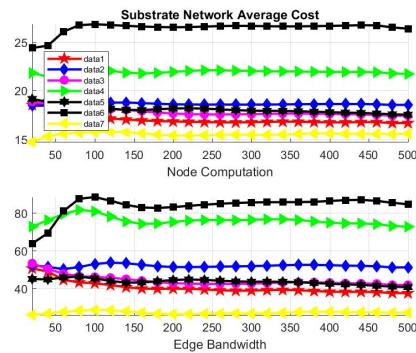


图 6.19 底层物理网络消耗资源的平均成本

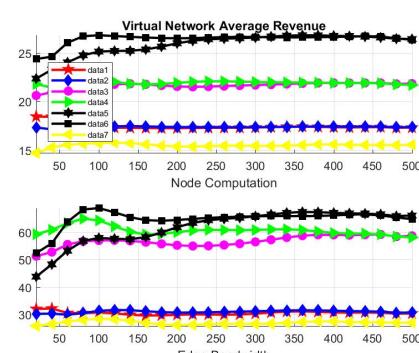


图 6.20 虚拟网络获得需求的平均收益

与算法**bVN**最接近，算法*STAR*比其它算法*RVN*和*oVN*具有更优的成本和收益。虚拟网络的平均收益如图6.20所示。

6.9 小结

本节提出了一种有效的算法*STAR*, 用于解决在具有可生存性保证请求的嵌入虚拟网络中, 通过冗余节点和链路来有效地分配SeVN问题的增广资源(启动新的物理节点、链路带宽或节点计算)。资源分配方法考虑的是冗余节点是被动的还是主动的, 只要在从故障恢复时有充足的资源可用。由于一个物理基础设施(底层网络)承载多个VN, 因此在VN之间共享冗余节点是更有效的资源。若要减少分配给可生存性保证的资源, 请执行以下操作。我们引入了一种启发式方法来共享这些独立的和相关的故障类型, 并在星型构造分解和动态规划的基础上实现了资源的最优分配。我们的算法采用了一种新的资源分配方法来减少新的启动主机, 在有限的进程时间内最小化资源, 而且即使问题的规模相对较小, 线性规划也无法计算出最优解。仿真结果表明, 该算法在节约资源、实现高接受率、高资源利用率和低启动节点数量等方面具有显著的效果。

第7章 总结和展望

7.1 本文工作总结

本文针对网络故障的自身特点，对如何在SDN/NFV网络架构中提出时效快，低开销，高可生存性的保护算法，做出了多层次，多角度，多方位的深入研究和分析。本文针对SDN/NFV网络架构中可生存性算法研究做出了以下贡献

1. 首先，为了提高主路径传输的可生存性，全面研究了现在网络故障存在的各种故障类型的特点，包括主动式和被动式。针对这些故障恢复机制方式，设计了快速的分而治之的快速SRLG不相交路径对算法。
2. 本文提出了一种拓扑图在存在陷阱问题情况下求解Min-Min SRLG不相交路由问题的高效算法。为了降低搜索的复杂性，我们创新性提出了一种分而治之的解决方案，将原Min-Min SRLG 不相交路由问题划分为多个子问题，该子问题基于从AP路径上遇到陷阱问题时导出的SRLG冲突链路集。我提出的算法利用现有的AP搜索结果和并行执行来实现更快的路径查找。
3. 并且本文在一个多核CPU平台上使用合成的拓扑进行了广泛的模拟。仿真结果表明，在搜索速度比较下，该算法的查找性能优于其它现有算法。
4. 其次，我们分析虚拟网络嵌入问题的本质特征，从而提出适合星型分解动态规划嵌入的可生存性虚拟网络嵌入算法，以解决原先虚拟网络嵌入算法无法解决的时间复杂性和低资源利用率等问题。
5. 本文在可生存性虚拟网络嵌入问题中，引入网络节点带有特定功能类型的限制条件，创新性提出一种星型分割动态分配嵌入的启发式算法，在虚拟和物理星型图之间的权重设置上，考虑网络资源的利用率和网络节点开启的代价。
6. 我提出的算法能快速的实现虚拟网络嵌入的可生存性需求，仿真结果表明，我提出的算法与其他现有算法效果比，虚拟嵌入可生存性请求的成功率更高，嵌入的物理资源消耗更低，物理资源的利用率更高。

7.2 未来研究工作与展望

由于自身条件和时间的限制，我们对网络中可生存性算法的研究工作还不够深入也不够完善。在分析和总结本文工作的同时，未来我还可以从以下几个方面继续研究：

1. 对于SRLG不相交问题，可以考虑路径上的附属限制条件不单单只是权重

值一个累加性属性，可以同时考虑多限制条件，并且限制条件是多属性的，对不相交条件的限制不再是单单的共享风险链路组不相交，可以同时考虑点或者边不相交的混合情形。

2. 本文提出的Min-Min SRLG不相交路径对算法考虑的是单备份路径，算法可以通过迭代法和引申SRLG冲突集的概念拓展到求多条Min-Min SRLG不相交路径的算法。
3. 对可生存性虚拟网络嵌入问题研究中，未来可以考虑对多个物理节点发生故障的情形，研究多个故障点发生的概率分布和影响特征，同样，我提出的算法能容易的实现多故障点的情形。和考虑节点和链路更多的需求约束条件。
4. 本文提出的星型分解动态规划嵌入的可生存性算法，没有考虑链路的映射的优化问题，我们可以在本文的算法的基础上做进一步改进，提出适合前面节点映射的链路映射算法，来和谐节点和链路映射两部分的资源消耗。

参考文献

- [1] Nick McKeown, Tom Anderson, Hari Balakrishnan, et al. OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 2008, 38(2):69–74
- [2] Christian Doerr, Fernando A Kuipers. All quiet on the internet front? *IEEE Communications Magazine*, 2014, 52(10):46–51
- [3] Fernando A Kuipers. An overview of algorithms for network survivability. *ISRN Communications and Networking*, 2012, 2012
- [4] JW Suurballe. Disjoint paths in a network. *Networks*, 1974, 4(2):125–145
- [5] John W Suurballe, Robert Endre Tarjan. A quick method for finding shortest pairs of disjoint paths. *Networks*, 1984, 14(2):325–336
- [6] D Anthony Dunn, Wayne D Grover, Mike H MacGregor. Comparison of k-shortest paths and maximum flow routing for network facility restoration. *IEEE Journal on selected areas in Communications*, 1994, 12(1):88–99
- [7] Longkun Guo, Hong Shen. On finding min-min disjoint paths. *Algorithmica*, 2013, 66(3):641–653
- [8] Justine Sherry, Shaddi Hasan, Colin Scott, et al. Making middleboxes someone else’s problem: network processing as a cloud service. *ACM SIGCOMM Computer Communication Review*, 2012, 42(4):13–24
- [9] Zhaoguang Wang, Zhiyun Qian, Qiang Xu, et al. An untold story of middleboxes in cellular networks. In: Proc of ACM SIGCOMM Computer Communication Review, volume 41. ACM, 2011, 374–385
- [10] Michael Walfish, Jeremy Stribling, Maxwell N Krohn, et al. Middleboxes No Longer Considered Harmful.. In: Proc of OSDI, volume 4. 2004, 15–15
- [11] Gregor Schaffrath, Christoph Werle, Panagiotis Papadimitriou, et al. Network virtualization architecture: Proposal and initial prototype. In: Proc of Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures. ACM, 2009, 63–72
- [12] NM Mosharaf Kabir Chowdhury, Raouf Boutaba. A survey of network virtualization. *Computer Networks*, 2010, 54(5):862–876
- [13] NM Mosharaf Kabir Chowdhury, Raouf Boutaba. Network virtualization: state of the art and research challenges. *IEEE Communications magazine*, 2009, 47(7)

- [14] Margaret Chiosi, Don Clarke, Peter Willis, et al. Network functions virtualisation: An introduction, benefits, enablers, challenges and call for action. In: Proc of SDN and OpenFlow World Congress. 2012, 22–24
- [15] Frank Yue. Network functions virtualization-everything old is new again. F5 Neworks, 2013.
- [16] Antonio Manzalini, Roberto Saracco, Cagatay Buyukkoc, et al. Software-defined networks for future networks and services. In: Proc of White Paper based on the IEEE Workshop SDN4FNS. 2014
- [17] Soheil Hassas Yeganeh, Amin Tootoonchian, Yashar Ganjali. On scalability of software-defined networking. *IEEE Communications Magazine*, 2013, 51(2):136–141
- [18] Xiaohu Ge, Hui Cheng, Mohsen Guizani, et al. 5G wireless backhaul networks: challenges and research advances. *IEEE Network*, 2014, 28(6):6–11
- [19] Andreas Fischer, Juan Felipe Botero, Michael Till Beck, et al. Virtual network embedding: A survey. *IEEE Communications Surveys & Tutorials*, 2013, 15(4):1888–1906
- [20] Sandra Herker, Ashiq Khan, Xueli An. Survey on survivable virtual network embedding problem and solutions. In: Proc of International Conference on Networking and Services, ICNS. 2013
- [21] Ian F Akyildiz, Ahyoung Lee, Pu Wang, et al. A roadmap for traffic engineering in SDN-OpenFlow networks. *Computer Networks*, 2014, 71:1–30
- [22] Amund Kvalbein, Audun Fosselie Hansen, Stein Gjessing, et al. Fast IP network recovery using multiple routing configurations. In: Proc of in INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings. Citeseer, 2006
- [23] Ning Qi, Bin-Qiang Wang, Zhi-Ming Wang. Research on reconfigurable service carrying network resilient construction algorithms. *Dianzi Yu Xinxi Xuebao(Journal of Electronics and Information Technology)*, 2012, 34(2):468–473
- [24] Mike Shand, Stewart Bryant. IP fast reroute framework. 2010.
- [25] Baohua Yang, Junda Liu, Scott Shenker, et al. Keep forwarding: Towards k-link failure resilient routing. In: Proc of INFOCOM, 2014 Proceedings IEEE. IEEE, 2014, 1617–1625
- [26] Ye Wang, Hao Wang, Ajay Mahimkar, et al. R3: resilient routing reconfiguration. In: Proc of ACM SIGCOMM Computer Communication Review, volume 40. ACM, 2010, 291–302

- [27] Ron Banner, Ariel Orda. Designing low-capacity backup networks for fast restoration. IEEE, 2010
- [28] Martin Suchara, Dahai Xu, Robert Doverspike, et al. Network architecture for joint failure recovery and traffic engineering. In: Proc of Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems. ACM, 2011, 97–108
- [29] Qiang Zheng, Guohong Cao, Thomas F La Porta, et al. Cross-layer approach for minimizing routing disruption in IP networks. IEEE Transactions on Parallel and Distributed Systems, 2014, 25(7):1659–1669
- [30] Satyajayant Misra, Guoliang Xue, Dejun Yang. Polynomial time approximations for multi-path routing with bandwidth and delay constraints. In: Proc of INFOCOM 2009, IEEE. IEEE, 2009, 558–566
- [31] Shimon Even, Alon Itai, Adi Shamir. On the complexity of time table and multi-commodity flow problems. In: Proc of Foundations of Computer Science, 1975., 16th Annual Symposium on. IEEE, 1975, 184–193
- [32] Hongfang Yu, Vishal Anand, Chunming Qiao, et al. Cost efficient design of survivable virtual infrastructure to recover from facility node failures. In: Proc of Communications (ICC), 2011 IEEE International Conference on. IEEE, 2011, 1–6
- [33] Zhiming Wang, Jiangxing Wu, Yu Wang, et al. Survivable virtual network mapping using optimal backup topology in virtualized SDN. China communications, 2014, 11(2):26–37
- [34] Gang Sun, Hongfang Yu, Lemin Li, et al. Efficient algorithms for survivable virtual network embedding. In: Proc of Asia Communications and Photonics Conference and Exhibition. Optical Society of America, 2010, 79890K
- [35] Qian Hu, Yang Wang, Xiaojun Cao. Location-constrained survivable network virtualization. In: Proc of Sarnoff Symposium (SARNOFF), 2012 35th IEEE. IEEE, 2012, 1–5
- [36] Muntasir Raihan Rahman, Issam Aib, Raouf Boutaba. Survivable virtual network embedding. In: Proc of International Conference on Research in Networking. Springer, 2010, 40–52
- [37] Zhu Qiang, WangHui Qiang, FengGuang Sheng, et al. Heuristic survivable virtual network embedding based on node migration and link remapping. In: Proc of Information Technology and Artificial Intelligence Conference (ITAIC), 2014 IEEE 7th Joint International. IEEE, 2014, 181–185

- [38] LU Bo, Tao Huang, Xiao-chuan SUN, et al. Dynamic recovery for survivable virtual network embedding. *The Journal of China Universities of Posts and Telecommunications*, 2014, 21(3):77–84
- [39] Lester Randolph Ford Jr, Delbert Ray Fulkerson. *Flows in networks*. Princeton university press, 2015
- [40] Mohamed Al-Kuwaiti, Nicholas Kyriakopoulos, Sayed Hussein. A comparative analysis of network dependability, fault-tolerance, reliability, security, and survivability. *IEEE Communications Surveys & Tutorials*, 2009, 11(2)
- [41] BA Hollway, PG Neumann. Survivable computer-communication systems: The problem and working group recommendations. Washington: US Army Research Laboratory, 1993.
- [42] Robert J Ellison, David A Fisher, Richard C Linger, et al. Survivable network systems: An emerging discipline. Technical report, Carnegie-mellon Univ Pittsburgh PA Software Engineering Inst, 1997
- [43] 韩建军, 张迎. 网络可生存性研究综述. *今日科苑*, 2007, (18):15–15
- [44] 卢兴华, 刘增良, 林憬锵. 信息系统生存性评估方法研究. *微计算机信息*, 2006, 22(3):39–41
- [45] 林雪纲, 许榕生. 信息系统生存性分析模型研究. *通信学报*, 2006, 27(2):153–159
- [46] 王秀君. 网络中可靠路由算法的研究: [Thesis]. 山东师范大学, 2008
- [47] Amund Kvalbein, Audun Fosselie Hansen, Tarik Čičić, et al. Multiple routing configurations for fast IP network recovery. *IEEE/ACM Transactions on Networking (TON)*, 2009, 17(2):473–486
- [48] Piet Van Mieghem, Huijuan Wang, Xin Ge, et al. Influence of assortativity and degree-preserving rewiring on the spectra of networks. *The European Physical Journal B*, 2010, 76(4):643–652
- [49] Damon Mosk-Aoyama. Maximum algebraic connectivity augmentation is NP-hard. *Operations Research Letters*, 2008, 36(6):677–679
- [50] Mihalis Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM Journal on Algebraic Discrete Methods*, 1981, 2(1):77–79
- [51] Ping Pan, George Swallow, Alia Atlas. Fast reroute extensions to RSVP-TE for LSP tunnels. Technical report, 2005
- [52] Dongyun Zhou, Suresh Subramaniam. Survivability in optical networks. *IEEE network*, 2000, 14(6):16–23

- [53] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1959, 1(1):269–271
- [54] Paolo Toth, Daniele Vigo. The vehicle routing problem. SIAM, 2002
- [55] Ramesh Bhandari. Optimal physical diversity algorithms and survivable networks. In: Proc of Computers and Communications, 1997. Proceedings., Second IEEE Symposium on. IEEE, 1997, 433–441
- [56] Matthieu Clouqueur, Wayne D Grover. Availability analysis of span-restorable mesh networks. *IEEE journal on selected areas in communications*, 2002, 20(4):810–821
- [57] Ramesh Bhandari. Optimal diverse routing in telecommunication fiber networks. In: Proc of INFOCOM’94. Networking for Global Communications., 13th Proceedings IEEE. IEEE, 1994, 1498–1508
- [58] David Coudert, Pallab Datta, Stéphane Pérennes, et al. Shared risk resource group complexity and approximability issues. *Parallel Processing Letters*, 2007, 17(02):169–184
- [59] Nina Taft-Plotkin, Bhargav Bellur, Richard Ogier. Quality-of-service routing using maximally disjoint paths. In: Proc of Quality of Service, 1999. IWQoS’99. 1999 Seventh International Workshop on. IEEE, 1999, 119–128
- [60] Chengyi Gao, Mohammad M Hasan, Jason P Jue. Domain-disjoint routing based on topology aggregation for survivable multidomain optical networks. *Journal of Optical Communications and Networking*, 2013, 5(12):1382–1390
- [61] Stojan Trajanovski, Fernando A Kuipers, Aleksandar Ilić, et al. Finding critical regions and region-disjoint paths in a network. *IEEE/ACM Transactions on Networking (TON)*, 2015, 23(3):908–921
- [62] Steven Fortune, John Hopcroft, James Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, 1980, 10(2):111–121
- [63] Neil Robertson, Paul D Seymour. Graph minors. XIII. The disjoint paths problem. *Journal of combinatorial theory, Series B*, 1995, 63(1):65–110
- [64] Alexander Schrijver. Finding k disjoint paths in a directed planar graph. *SIAM Journal on Computing*, 1994, 23(4):780–788
- [65] Jian Qiang Hu. Diverse routing in optical mesh networks. *IEEE Transactions on Communications*, 2003, 51(3):489–494
- [66] Jean-Claude Bermond, David Coudert, Gianlorenzo D’Angelo, et al. SRLG-diverse routing with the star property. In: Proc of Design of Reliable Communication Networks (DRCN), 2013 9th International Conference on the. IEEE, 2013, 163–170

- [67] Chung-Lun Li, S Thomas McCormick, David Simchi-Levi. The complexity of finding two disjoint paths with min-max objective function. *Discrete Applied Mathematics*, 1990, 26(1):105–115
- [68] Yuchun Guo, Fernando Kuipers, Piet Van Mieghem. Link-disjoint paths for reliable QoS routing. *International Journal of Communication Systems*, 2003, 16(9):779–798
- [69] Dahai Xu, Yang Chen, Yizhi Xiong, et al. On finding disjoint paths in single and dual link cost networks. In: Proc of INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, volume 1. IEEE, 2004
- [70] AA Beshir, FA Kuipers. Variants of the min-sum link-disjoint paths problem. IEEE/SCVT, 2011
- [71] Eiji Oki, Nobuaki Matsuura, Kohei Shiromoto, et al. A disjoint path selection scheme with shared risk link groups in GMPLS networks. *IEEE Communications letters*, 2002, 6(9):406–408
- [72] Guangzhi Li, Bob Doverspike, Chuck Kalmanek. Fiber span failure protection in mesh optical networks. *Optical Networks Magazine*, 2002, 3(3):21–31
- [73] David Eppstein. Finding the k shortest paths. *SIAM Journal on computing*, 1998, 28(2):652–673
- [74] P Laborczi, J Tapolcai, Pin-Han Ho, et al. Solving asymmetrically weighted optimal or near-optimal disjoint path pair for the survivable optical networks. In: Proc of Third International Workshop On Design Of Reliable Communication Networks (DRCN’01). 2001
- [75] Mohammad Javad Rostami, Azadeh Alasadat Emrani Zarandi, Seyed Mohamad Hosseini Nasab. MSDP with ACO: A maximal SRLG disjoint routing algorithm based on ant colony optimization. *Journal of Network and Computer Applications*, 2012, 35(1):394–402
- [76] Mohammad Javad Rostami, Siavash Khorsandi, Ali Asghar Khodaparast. CoSE: A SRLG-disjoint routing algorithm. In: Proc of Universal Multiservice Networks, 2007. ECUMN’07. Fourth European Conference on. IEEE, 2007, 86–92
- [77] Pallab Datta, Arun K Somani. Graph transformation approaches for diverse routing in shared risk resource group (SRRG) failures. *Computer Networks*, 2008, 52(12):2381–2394
- [78] Dahai Xu, Yizhi Xiong, Chunming Qiao. A new PROMISE algorithm in networks with shared risk link groups. In: Proc of Global Telecommunications Conference, 2003. GLOBECOM’03. IEEE, volume 5. IEEE, 2003, 2536–2540

- [79] Ajay Todimala, Byrar Ramamurthy. IMSH: An iterative heuristic for SRLG diverse routing in WDM mesh networks. In: Proc of Computer Communications and Networks. Proceedings. 13th International Conference on. IEEE, 2004, 199–204
- [80] Dahai Xu, Yizhi Xiong, Chunming Qiao, et al. Trap avoidance and protection schemes in networks with shared risk link groups. *Journal of Lightwave Technology*, 2003, 21(11):2683
- [81] Randeep Bhatia, Murali Kodialam, TV Lakshman. Finding disjoint paths with related path costs. *Journal of Combinatorial Optimization*, 2006, 12(1-2):83–96
- [82] Vasek Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 1979, 4(3):233–235
- [83] Gurobi Optimization, et al. Gurobi optimizer reference manual. URL: <http://www.gurobi.com>, 2012, 2:1–3
- [84] Ananth Grama. Introduction to parallel computing. Pearson Education, 2003
- [85] DM Tax, RP Duin. Feature scaling in support vector data descriptions. *Learning from Imbalanced Datasets*, 2000. 25–30
- [86] Gene M Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In: Proc of Proceedings of the April 18-20, 1967, spring joint computer conference. ACM, 1967, 483–485
- [87] Edoardo Amaldi, Stefano Coniglio, Arie MCA Koster, et al. On the computational complexity of the virtual network embedding problem. *Electronic Notes in Discrete Mathematics*, 2016, 52:213–220
- [88] Jens Lischka, Holger Karl. A virtual network mapping algorithm based on subgraph isomorphism detection. In: Proc of Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures. ACM, 2009, 81–88
- [89] Minlan Yu, Yung Yi, Jennifer Rexford, et al. Rethinking virtual network embedding: substrate support for path splitting and migration. *ACM SIGCOMM Computer Communication Review*, 2008, 38(2):17–29
- [90] David G Andersen. Theoretical approaches to node assignment. Computer Science Department, 2002. 86
- [91] Johannes Inführ, Günther R Raidl. Introducing the virtual network mapping problem with delay, routing and location constraints. In: Proc of Network optimization. Springer, 2011: 105–117
- [92] Wenzhi Liu, Yang Xiang, Shaowu Ma, et al. Completing virtual network embedding all in one mathematical programming. In: Proc of Electronics, Communications and Control (ICECC), 2011 International Conference on. IEEE, 2011, 183–185

- [93] Tri Trinh, Hiroshi Esaki, Chaodit Aswakul. Quality of service using careful overbooking for optimal virtual network resource allocation. In: Proc of Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2011 8th International Conference on. IEEE, 2011, 296–299
- [94] Albert Pages, Jordi Perello, Salvatore Spadaro, et al. Strategies for virtual optical network allocation. *IEEE Communications Letters*, 2012, 16(2):268–271
- [95] Hao Di, Lemin Li, Vishal Anand, et al. Cost efficient virtual infrastructure mapping using subgraph isomorphism. In: Proc of Asia Communications and Photonics Conference and Exhibition. Optical Society of America, 2010, 79890L
- [96] Tay Ghazar, Nancy Samaan. Hierarchical approach for efficient virtual network embedding based on exact subgraph matching. In: Proc of Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE. IEEE, 2011, 1–6
- [97] Donggyu Yun, Yung Yi. Virtual network embedding in wireless multihop networks. In: Proc of Proceedings of the 6th international conference on future internet technologies. ACM, 2011, 30–33
- [98] Donggyu Yun, Jungseul Ok, Bongjhin Shin, et al. Embedding of virtual network requests over static wireless multihop networks. *Computer Networks*, 2013, 57(5):1139–1152
- [99] Xuzhou Chen, Yan Luo, Jie Wang. Virtual network embedding with border matching. In: Proc of Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on. IEEE, 2012, 1–8
- [100] Hongfang Yu, Vishal Anand, Chunming Qiao, et al. A cost efficient design of virtual infrastructures with joint node and link mapping. *Journal of Network and Systems Management*, 2012, 20(1):97–115
- [101] Jiang Liu, Tao Huang, Jian-ya Chen, et al. A new algorithm based on the proximity principle for the virtual network embedding problem. *Journal of Zhejiang University SCIENCE C*, 2011, 12(11):910
- [102] Sheng Zhang, Zhuzhong Qian, Bin Tang, et al. Opportunistic bandwidth sharing for virtual network mapping. In: Proc of Global telecommunications conference (GLOBECOM 2011), 2011 IEEE. IEEE, 2011, 1–5
- [103] Sheng Zhang, Zhuzhong Qian, Jie Wu, et al. An opportunistic resource sharing and topology-aware mapping framework for virtual networks. In: Proc of INFOCOM, 2012 Proceedings IEEE. IEEE, 2012, 2408–2416
- [104] Xiao-ling Li, Huai-min Wang, Chang-guo Guo, et al. Topology awareness algorithm for virtual network mapping. *Journal of Zhejiang University SCIENCE C*, 2012, 13(3):178–186

- [105] Jing Lu, Jonathan Turner. Efficient mapping of virtual networks onto a shared substrate. 2006.
- [106] Adil Razzaq, Muhammad Siraj Rathore. An approach towards resource efficient virtual network embedding. In: Proc of Evolving Internet (INTERNET), 2010 Second International Conference on. IEEE, 2010, 68–73
- [107] Adil Razzaq, Peter Sjödin, Markus Hidell. Minimizing bottleneck nodes of a substrate in virtual network embedding. In: Proc of Network of the Future (NOF), 2011 International Conference on the. IEEE, 2011, 35–40
- [108] Joao Nogueira, Márcio Melo, Jorge Carapinha, et al. Virtual network mapping into heterogeneous substrate networks. In: Proc of Computers and Communications (ISCC), 2011 IEEE Symposium on. IEEE, 2011, 438–444
- [109] Aris Leivadeas, Chrysa Papagianni, Evripidis Paraskevas, et al. An architecture for virtual network embedding in wireless systems. In: Proc of Network Cloud Computing and Applications (NCCA), 2011 First International Symposium on. IEEE, 2011, 62–68
- [110] Juan Felipe Botero, Xavier Hesselbach, Andreas Fischer, et al. Optimal mapping of virtual networks with hidden hops. *Telecommunication Systems*, 2012, 51(4):273–282
- [111] Juan Felipe Botero Vega, Xavier Hesselbach Serra, Michael Duelli, et al. Flexible VNE algorithms analysis using ALEVIN. In: Proc of Proceedings. 2011, 47–48
- [112] Yong Zhu, Mostafa H Ammar. Algorithms for assigning substrate network resources to virtual network components.. In: Proc of INFOCOM, volume 1200. 2006, 1–12
- [113] Ilhem Fajjari, Nadjib Ait Saadi, Guy Pujolle, et al. VNE-AC: Virtual network embedding algorithm based on ant colony metaheuristic. In: Proc of Communications (ICC), 2011 IEEE International Conference on. IEEE, 2011, 1–6
- [114] Xiang Cheng, Sen Su, Zhongbao Zhang, et al. Virtual network embedding through topology awareness and optimization. *Computer Networks*, 2012, 56(6):1797–1813
- [115] Xiang Cheng, Sen Su, Zhongbao Zhang, et al. Virtual network embedding through topology-aware node ranking. *ACM SIGCOMM Computer Communication Review*, 2011, 41(2):38–47
- [116] Sheng Zhang, Jie Wu, Sanglu Lu. Virtual network embedding with substrate support for parallelization. In: Proc of Global Communications Conference (GLOBECOM), 2012 IEEE. IEEE, 2012, 2615–2620
- [117] Hao Di, Hongfang Yu, Vishal Anand, et al. Efficient online virtual network mapping using resource evaluation. *Journal of Network and Systems Management*, 2012, 20(4):468–488

- [118] Vahid Abedifar, Mohammad Eshghi. A novel routing and wavelength assignment in virtual network mapping based on the minimum path algorithm. In: Proc of Ubiquitous and Future Networks (ICUFN), 2012 Fourth International Conference on. IEEE, 2012, 204–208
- [119] Aris Leivadeas, Chrysa Papagianni, Symeon Papavassiliou. Socio-aware virtual network embedding. *IEEE Network*, 2012, 26(5)
- [120] Tae-Ho Lee, Shahnaza Tursunova, Tae-Sang Choi. Graph clustering based provisioning algorithm for virtual network embedding. In: Proc of Network Operations and Management Symposium (NOMS), 2012 IEEE. IEEE, 2012, 1175–1178
- [121] Ilhem Fajjari, Nadjib Aitsaadi, Guy Pujolle, et al. Vnr algorithm: A greedy approach for virtual networks reconfigurations. In: Proc of Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE. IEEE, 2011, 1–6
- [122] Marcin Bienkowski, Anja Feldmann, Dan Jurca, et al. Competitive analysis for service migration in vnets. In: Proc of Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures. ACM, 2010, 17–24
- [123] Marcin Bienkowski, Anja Feldmann, Johannes Grassler, et al. The wide-area virtual service migration problem: A competitive analysis approach. *IEEE/ACM Transactions on Networking (ToN)*, 2014, 22(1):165–178
- [124] Jinliang Fan, Mostafa H Ammar. Dynamic topology configuration in service overlay networks: A study of reconfiguration policies. In: Proc of INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings. Citeseer, 2006, 1–12
- [125] Zhiping Cai, Fang Liu, Nong Xiao, et al. Virtual network embedding for evolving networks. In: Proc of Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE, 2010, 1–5
- [126] Shun-li Zhang, Xue-song Qiu. A novel virtual network mapping algorithm for cost minimizing. *Journal of Selected Areas in Telecommunications*, 2011, 29(1):1–9
- [127] Gang Sun, Hongfang Yu, Vishal Anand, et al. A cost efficient framework and algorithm for embedding dynamic virtual network requests. *Future Generation Computer Systems*, 2013, 29(5):1265–1277
- [128] Ines Houidi, Wajdi Louati, Djamel Zeghlache. A distributed virtual network mapping algorithm. In: Proc of Communications, 2008. ICC’08. IEEE International Conference on. IEEE, 2008, 5634–5640

- [129] Yufeng Xin, Ilia Baldine, Anirban Mandal, et al. Embedding virtual topologies in networked clouds. In: Proc of Proceedings of the 6th international conference on future internet technologies. ACM, 2011, 26–29
- [130] Bo Lv, Zhenkai Wang, Tao Huang, et al. Virtual resource organization and virtual network embedding across multiple domains. In: Proc of Multimedia Information Networking and Security (MINES), 2010 International Conference on. IEEE, 2010, 725–728
- [131] Ines Houidi, Wajdi Louati, Walid Ben Ameur, et al. Virtual network provisioning across multiple substrate networks. *Computer Networks*, 2011, 55(4):1011–1023
- [132] Aris Leivadeas, Chrysa Papagianni, Symeon Papavassiliou. Efficient resource mapping framework over networked clouds via iterated local search-based request partitioning. *IEEE Transactions on Parallel and Distributed Systems*, 2013, 24(6):1077–1086
- [133] Clarissa C Marquezan, Lisandro Z Granville, Giorgio Nunzi, et al. Distributed autonomic resource management for network virtualization. In: Proc of Network operations and management symposium (NOMS), 2010 IEEE. IEEE, 2010, 463–470
- [134] Xian Zhang, Chris Phillips, Xiuzhong Chen. An overlay mapping model for achieving enhanced QoS and resilience performance. In: Proc of Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2011 3rd International Congress on. IEEE, 2011, 1–7
- [135] Juan Felipe Botero, Xavier Hesselbach, Michael Duelli, et al. Energy efficient virtual network embedding. *IEEE Communications Letters*, 2012, 16(5):756–759
- [136] Cong Wang, Shashank Shanbhag, Tilman Wolf. Virtual network mapping with traffic matrices. In: Proc of Communications (ICC), 2012 IEEE International Conference on. IEEE, 2012, 2717–2722
- [137] Jawwad Shamsi, Monica Brockmeyer. QoSMap: QoS aware mapping of virtual networks for resiliency and efficiency. In: Proc of Globecom Workshops, 2007 IEEE. IEEE, 2007, 1–6
- [138] Jawwad Shamsi, Monica Brockmeyer. Efficient and dependable overlay networks. In: Proc of Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on. IEEE, 2008, 1–8
- [139] Jawwad Shamsi, Monica Brockmeyer. QoSMap: Achieving quality and resilience through overlay construction. In: Proc of Internet and Web Applications and Services, 2009. ICIW’09. Fourth International Conference on. IEEE, 2009, 58–67

- [140] Guilherme Koslovski, Wai-Leong Yeow, Cedric Westphal, et al. Reliability support in virtual infrastructures. In: Proc of Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on. IEEE, 2010, 49–58
- [141] Hongfang Yu, Chunming Qiao, Vishal Anand, et al. Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures. In: Proc of Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE. IEEE, 2010, 1–6
- [142] Pin Lv, Zhiping Cai, Jia Xu, et al. Multicast service-oriented virtual network embedding in wireless mesh networks. *IEEE Communications Letters*, 2012, 16(3):375–377
- [143] Mosharaf Chowdhury, Muntasir Raihan Rahman, Raouf Boutaba. Vineyard: Virtual network embedding algorithms with coordinated node and link mapping. *IEEE/ACM Transactions on Networking (TON)*, 2012, 20(1):206–219
- [144] NM Mosharaf Kabir Chowdhury, Muntasir Raihan Rahman, Raouf Boutaba. Virtual network embedding with coordinated node and link mapping. In: Proc of INFOCOM 2009, IEEE. IEEE, 2009, 783–791
- [145] Nabeel Farooq Butt, Mosharaf Chowdhury, Raouf Boutaba. Topology-awareness and reoptimization mechanism for virtual network embedding. In: Proc of International Conference on Research in Networking. Springer, 2010, 27–39
- [146] Wai-Leong Yeow, Cédric Westphal, Ulaş Kozat. Designing and embedding reliable virtual infrastructures. In: Proc of Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures. ACM, 2010, 33–40
- [147] Gang Sun, Hongfang Yu, Lemin Li, et al. The framework and algorithms for the survivable mapping of virtual network onto a substrate network. *IETE Technical Review*, 2011, 28(5):381–391
- [148] Xiujiao Gao, Hongfang Yu, Vishal Anand, et al. A new algorithm with coordinated node and link mapping for virtual network embedding based on LP relaxation. In: Proc of Asia Communications and Photonics Conference and Exhibition. Optical Society of America, 2010, 79881Y
- [149] Fan Yang, Zhen-kai Wang, Jian-ya Chen, et al. Vlb-vne: A regionalized valiant load-balancing algorithm in virtual network mapping. In: Proc of Wireless Communications, Networking and Information Security (WCNIS), 2010 IEEE International Conference on. IEEE, 2010, 432–436

- [150] Ye Zhou, Yong Li, Depeng Jin, et al. A virtual network embedding scheme with two-stage node mapping based on physical resource migration. In: Proc of Communication Systems (ICCS), 2010 IEEE International Conference on. IEEE, 2010, 761–766
- [151] Yang Chen, Jianxin Li, Tianyu Wo, et al. Resilient virtual network service provision in network virtualization environments. In: Proc of Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on. IEEE, 2010, 51–58
- [152] Yu Yang, Shan-zhi Chen, Xin Li, et al. RMap: An algorithm of virtual network resilience mapping. In: Proc of Wireless Communications, Networking and Mobile Computing (WiCOM), 2011 7th International Conference on. IEEE, 2011, 1–4
- [153] Gang Sun, Hongfang Yu, Lemin Li, et al. Exploring online virtual networks mapping with stochastic bandwidth demand in multi-datacenter. Photonic Network Communications, 2012, 23(2):109–122
- [154] LÜ Bo, Tao Huang, Zhen-kai Wang, et al. Adaptive scheme based on status feedback for virtual network mapping. The Journal of China Universities of Posts and Telecommunications, 2011, 18(5):87–94
- [155] Tao Guo, Ning Wang, Klaus Moessner, et al. Shared backup network provision for virtual network embedding. In: Proc of Communications (ICC), 2011 IEEE International Conference on. IEEE, 2011, 1–5
- [156] Sheng Zhang, Zhuzhong Qian, Song Guo, et al. FELL: A flexible virtual network embedding algorithm with guaranteed load balancing. In: Proc of Communications (ICC), 2011 IEEE International Conference on. IEEE, 2011, 1–5
- [157] Zhongbao Zhang, Xiang Cheng, Sen Su, et al. A unified enhanced particle swarm optimization-based virtual network embedding algorithm. International Journal of Communication Systems, 2013, 26(8):1054–1073
- [158] Gang Sun, Hongfang Yu, Vishal Anand, et al. Optimal provisioning for virtual network request in cloud-based data centers. Photonic Network Communications, 2012, 24(2):118–131
- [159] Pin Lv, Xudong Wang, Ming Xu. Virtual access network embedding in wireless mesh networks. Ad Hoc Networks, 2012, 10(7):1362–1378
- [160] Sarang Bharadwaj Masti, Serugudi V Raghavan. Vna: An enhanced algorithm for virtual network embedding. In: Proc of Computer Communications and Networks (ICCCN), 2012 21st International Conference on. IEEE, 2012, 1–9
- [161] Xian Zhang, Xiuzhong Chen, Chris Phillips. Achieving effective resilience for QoS-aware application mapping. Computer Networks, 2012, 56(14):3179–3191

- [162] Gregor Schaffrath, Stefan Schmid, Anja Feldmann. Optimizing long-lived cloudnets with migrations. In: Proc of Proceedings of the 2012 IEEE/ACM fifth international conference on utility and cloud computing. IEEE Computer Society, 2012, 99–106
- [163] Dongdong Chen, Xuesong Qiu, Zhaowei Qu, et al. Algorithm for virtual nodes re-configuration on network virtualization. 2011.
- [164] Mosharaf Chowdhury, Fady Samuel, Raouf Boutaba. Polyvine: policy-based virtual network embedding across multiple domains. In: Proc of Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures. ACM, 2010, 49–56
- [165] Ines Houidi, Wajdi Louati, Djamal Zeghlache, et al. Adaptive virtual network provisioning. In: Proc of Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures. ACM, 2010, 41–48
- [166] Muntasir Raihan Rahman, Raouf Boutaba. SVNE: Survivable virtual network embedding algorithms for network virtualization. *IEEE Transactions on Network and Service Management*, 2013, 10(2):105–118
- [167] Hongfang Yu, Vishal Anand, Chunming Qiao, et al. Migration based protection for virtual infrastructure survivability for link failure. In: Proc of Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference. IEEE, 2011, 1–3
- [168] Chunming Qiao, Bingli Guo, Shanguo Huang, et al. A novel two-step approach to surviving facility failures. In: Proc of Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference. IEEE, 2011, 1–3
- [169] Jielong Xu, Jian Tang, Kevin Kwiat, et al. Survivable virtual infrastructure mapping in virtualized data centers. In: Proc of Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on. IEEE, 2012, 196–203
- [170] Richard M Karp. On the computational complexity of combinatorial problems. *Networks*, 1975, 5(1):45–68
- [171] Derick Wood, Derrick Wood. Theory of computation. 1987.
- [172] S Skiena. Dijkstra’s algorithm. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, Reading, MA: Addison-Wesley, 1990. 225–227
- [173] Sebastian Orlowski, Roland Wessäly, Michal Pióro, et al. SNDlib 1.0 —Survivable network design library. *Networks*, 2010, 55(3):276–286

致 谢

转瞬间，三年的研究生学习生涯落下帷幕，在这个三年的苦读学习中我得到了来自我的导师，我的朋友，我的同学们的无私的关怀和无私的帮助，让我在求学的道路上不断前行。值此论文完成之时，谨向帮助和关心过我的人致以最诚挚的谢意！

首先，我向我的恩师谢鲲教授表示我最深的敬意和感谢！非常感谢她在我攻读硕士学位的三年时间里，对我学习上给予的充分全面的指导，让我在个人能力上和学术能力都提升了一个台阶。谢老师宽厚待人的品格，高尚的师德，都作为一种深人格魅力，深深地影响着我对待人生对待学术的态度，这一切必将会让我终身受益，我在此向谢老师表示深深的谢意！谢老师用她渊博的知识对我们进行全面而专业的指导，她对待学术执着，细心，一丝不苟的精神对我形成一种教诫，让我督促自己也要像她那样对待学术，要通过不断地学习来提升自我能力。祝愿谢老师及家人平安，快乐，健康！

感谢基地实验室已经毕业的各位师姐师兄在我研究生一二年级时对我生活上和学习上的指导与帮助。特别要感谢宁雪萍师姐、黄俊师兄、施文师兄、郑哲师兄、陈宇翔师兄、王乐乐师兄和陈振华师兄在我学术刚入门时，对我的热情的帮助，无论是在学科的学习上还是论文的写作，他们的提点总会让我受益匪浅，让我从最初的迷茫期很快地过度到了前进期，非常感谢他们的指导和帮助。感谢彭灿、潘海娜、赵彦彦和李晓灿，我们一起探讨，学习上相互指导，生活上相互帮助，希望我们能够同门之情长存！感谢我的好朋友们胡海洋、何展、黎孟、侯宇凡、秦光睿和王思娟，跟你们一起相处，给我枯燥和压力的硕士学习过程增添了不少的欢笑，愿我们友谊长久！

深深地感谢我的爱人和我的父母，在我漫漫的求学路程中不断的激励我默默地支持我，生活中给我无微不至的关心，你们一直是我不断前进的主要动力。我会以自己的实际成绩向你们感恩和回馈！

感谢信息科学与工程学院的答辩委员会和评审论文的各位教师，感谢您们在百忙之中拿出宝贵的时间对我的论文进行建设性的指导和提出关键性的意见！

最后，我想感谢湖南大学、感谢我的学校。惟楚有才、于斯为盛，我有幸能够在千年学府里不断追溯着先贤的脚步和思想，在这里度过人生中珍贵的三年光阴。难忘的是那一次次呼朋引伴结伴而行的岳麓山之行、那一场场湘江边盛大的美丽烟火，爱晚亭的枫、湘江的水还有那麓山南路的风，将永远吹拂我的这颗恒心。

附录A 发表论文和参加科研情况说明

(一) 发表的学术论文

- [1] Kun Xie, Heng Tao, Xin Wang, Gaogang Xie, Jigang Wen, Jiannong Cao, Zheng Qin. Divide And Conquer For Fast SRLG Disjoint Routing[C]. DSN 2018: International Conference on Dependable Systems and Networks, Luxerbourg(CCF B).

(二) 申请及已获得的专利

- [1] 陶恒, 谢鲲. 一种求完全风险共享链路组分离路径对的方法及系统: 中国。