硕士学位论文

全局视角下的 SDN 动态升级策略研究

Research on SDN Migration Strategy
from a Dynamic and Global Perspective

作者：李小乐

导师：郑宏云

北京交通大学

2021 年 6 月

学位论文版权使用授权书

本学位论文作者完全了解北京交通大学有关保留、使用学位论文的规定。特授权北京交通大学可以将学位论文的全部或部分内容编入有关数据库进行检索，提供阅览服务，并采用影印、缩印或扫描等复制手段保存、汇编以供查阅和借阅。同意学校向国家有关部门或机构送交论文的复印件和磁盘。学校可以为存在馆际合作关系的兄弟高校用户提供文献传递服务和交换服务。

（保密的学位论文在解密后适用本授权说明）

学位论文作者签名：　　导师签名：

签字日期：　　年　　月　　日　　签字日期：　　年　　月　　日

学校代码：10004 密级：公开

北京交通大学

硕士学位论文

全局视角下的 SDN 动态升级策略研究

Research on SDN Migration Strategy
from a Dynamic and Global Perspective

作者姓名：李小乐 学　　号：18120101

导师姓名：郑宏云 职　　称：副教授

学位类别：工学 学位级别：硕士

学科专业：通信与信息系统 研究方向：信息网络

北京交通大学

2021 年 6 月

致谢

连雨不知春去，一晴方觉夏深，还没有看够明湖的鸭子摇摇晃晃，还没有感受够图书馆前凉爽的夏风，也还没有吃够实验室的零食，食堂的饭菜，就要毕业了，在此感谢许多人的帮助。

由衷地感谢我的导师郑宏云老师，您说话时温柔随和，亲切温婉，却句句条理清晰，逻辑缜密。学术科研上，您专注踏实，对学生严谨要求，悉心指导。在论文的写作修改过程中，您不厌其烦的和我一起反复修改，字斟句酌。生活中，您对学生关怀备至，尊重并鼓励学生的想法和决定。真心感谢郑老师在这三年对我的培育和关爱。

感谢郭宇春老师、赵永祥老师、李纯喜老师、和陈一帅老师。在每一次组会中，您们对于专业领域清晰敏锐的洞察力和严谨求真的治学态度让我钦佩。感谢各位老师在学习阶段给予我的无私帮助和关怀，在此向各位老师表示诚挚的谢意。感谢陈老师带我接触人工智能相关领域。

感谢实验室的张琛玥、张虎信、赵映南、高志鹏、李文雯、张庆贺同学以及各位师弟师妹，能够与大家相遇相识是我的荣幸。在实验室一起并肩奋斗的时光是我未来最怀念的。

感谢一直支持鼓励我的家人，尤其是我的姐姐李小欢，忍受我无时无刻的叨扰，是他们作为我人生中最坚实的后盾，帮助我顺利完成学业。

感谢我的挚爱王子吉安在学业和生活中给予我的支持和陪伴。何时杖尔看南雪，我与梅花两白头。愿你一切顺遂。

摘要

软件定义网络（Software Defined Network，SDN）因其转控分离和网络可编程性的特点，是 5G 网络的关键技术以及最具潜力的下一代网络解决方案之一。业界对其研究势头有增无减，在骨干网、承载网、接入网中开展了越来越多的研究和落地工作，本文聚焦于骨干网的 SDN 方案。但是，建设 SDN 网络不是一个革命过程，而是从原有的传统 IP 网络逐步升级到 SDN 网络的演进过程，这样的动态升级过程需要经过多个升级阶段，每个阶段决策升级哪些(W*hich*)交换机、交换机升级的顺序（W*hen*）以及如何部署控制器（*ho*W）三个方面(3W)的内容，从而完成从传统 IP 网络过渡到混合 SDN，再到纯 SDN 网络的升级过程。

最优的动态升级策略既要保证每一步升级带来的网络可编程性最大化和控制器部署后的混合 SDN 网络性能最优，称为局部双目标，也要确保升级后持续运营的纯 SDN 网络的性能最优，称为全局单目标。但是，每一步的局部目标与最终的全局目标存在时序正反两个方向上的复杂影响，使得同时考虑上述三个方面(3W)，得出最优动态升级策略具有难度。本文发现和研究了上述最优动态升级问题。

论文的第一个创新工作是同时考虑 3W 问题，将最优动态升级问题建模为一个双目标动态优化模型，在整个升级时序中（*when*），同时优化交换机升级（*which*）与控制器部署（*how*）；通过引入随时间变化的退化因子，刻画局部双目标到全局单目标的变化过程。与已有的交换机和控制器联合部署的双目标优化模型不同，本文提出的双目标是动态变化的；与经典动态优化模型不同，本优化模型在时序的正反两个方向上都存在动态性。

论文的第二个创新工作是，通过引入反向惩罚项，将双目标动态优化模型转化为一系列可以直接求解的单目标模型。进而提出启发式算法，通过改进遗传算法，得到基于 SDN 匹配关系的遗传算法（SM-GA）和基于互补交叉变异的遗传算法（CCM-GA）分别求解惩罚项与最优动态升级策略。

第三个创新工作是，将升级过程看作马尔可夫决策过程，提出了基于强化学习的求解算法。该算法通过学习自动习得全局收益对单步局部收益的影响，而不必像启发式算法那样通过经验进行定义，而经验定义未必能获得最优值。

论文使用真实的网络拓扑数据，对上述问题和算法进行仿真实验，并与基线算法进行对比。结果表明，本文所提的求解算法均更好地实现了局部目标与全局目标的平衡，升级得到的纯 SDN 网络性能更优。在大规模真实网络拓扑中，基于深度强化学习的求解算法较基于改进遗传算法的启发式算法能够获得更优的全局收益。

关键词：混合 SDN；动态升级；最优化；控制器联合部署；改进遗传算法；强化学习

ABSTRACT

Software Defined Network (SDN) is the key technology of 5G network and one of the most potential next-generation network solutions due to its separation of control and forwarding and network programmability. The research popularity is growing, and more and more companies choose to research and implement SDN in backbone networks, bearer networks, and access networks. This paper focuses on SDN solutions for backbone networks. However, the construction of SDN is not a revolutionary process, but an evolutionary process from the traditional IP network to an SDN network, which requires multiple steps. The entire dynamic upgrade process needs to solve three proble ms (3W): which switches to upgrade (Which), the switch upgrade sequence (When) and how to deploy the controller (hoW), to complete the upgrade process from the traditional IP network to the hybrid SDN, and then to the pure SDN network.

The optimal dynamic upgrade strategy not only needs to ensure the network programmability benefits brought by the upgraded switches and the network performance brought by the deployed controllers in each step, called the local dual-objective, but also needs to ensure the performance of the pure SDN which is continuous operated after the upgrade, called the global single objective. However, the local goal of each step and the final global goal have complex effects in both the positive and negative directions of the time sequence, which makes it difficult to obtain the optimal dynamic upgrade strategy by considering the above three aspects (3W) at the same time. This paper discovers and studies the above-mentioned optimal dynamic upgrade problem.

The first innovative work of the paper is to consider the 3W problem at the same time, and model the optimal dynamic upgrade problem as a dual-objective dynamic optimization model. During the entire upgrade sequence (when), simultaneously optimize the benefits of switches upgrade switch (which) and controller deployment (how). By introducing a degradation factor that changes with time, the transformation process from a local dual-objective to a global single objective is described. Different from the existing dual-objective optimization model for joint deployment of switches and controllers, the dual-objective proposed in this paper is dynamic. Different from the classic dynamic optimization model, this optimization model is dynamic in both directions of the timing sequence.

The second innovative work of the paper is to transform the dual-objective dynamic

optimization model into a series of single-objective integer linear programming proble ms which can be solved directly by introducing a reverse penalty term to consider both local and global benefits during single-step upgrades. Then a heuristic algorithm is proposed. By improving the genetic algorithm, a genetic algorithm based on SDN matching relationship (SM-GA) and a genetic algorithm based on complementary crossover mutation (CCM-GA) are obtained to solve the penalty term and the optimal upgrade strategy respectively.

The third innovative work is to regard the upgrade process as a Markov decision process and propose a solution algorithm based on reinforcement learning. The algorithm automatically learns the impact of global returns on single-step local returns through learning, instead of defining through experience like heuristic algorith ms, and empirical definitions may not be able to obtain the optimal value.

The third innovative work is to regard the upgrade process as a Markov decision process and propose a solution algorithm based on reinforcement learning. Compared with the heuristic algorithm, the solution algorithm based on reinforcement learning does not need to define the impact of the global benefits on the local decision, but learns the optimal value automatically through learning, while the empirical definitions may not be able to obtain the optimal value.

The paper uses real network topology data to conduct simulation experiments on the above proble ms and algorith ms, and compares them with the baselines. The results show that the solution algorith ms proposed in this paper better achieve the balance between the local and the global goal. Besides, the pure SDN obtained by the upgrade has better performance. In the large-scale real network topology, the solution algorithm based on deep reinforcement learning can obtain better global benefits than the heuristic algorithm based on the improved genetic algorithm.

KEYWORDS： Hybrid SDN; Dynamic Migration; Optimization; Joint Controller Deployment; Improved Genetic Algorithm; Reinforcement Learning

目录

1 引言

1.1 研究背景和意义

随着移动互联网的发展以及视频业务的活跃，互联网服务和信息内容逐渐多样化，移动通信流量持续增长[1]，给网络带来了巨大的压力，使得传统 IP 网络需要改进自身架构以适应现有的网络流量传输需求。软件定义网络（Software Defined Network, SDN）作为一种全新的网络范式，通过改进网络结构，分离转发平面与控制平面，可用于解决传统网络技术存在的固有结构僵化、配置复杂等缺陷[2]，被认为是一种可行的未来网络架构[3]。此外，软件定义网络的转控分离与可编程特点，有效满足了未来 5G 网络架构的主要技术特征，是 5G 网络中的一项关键技术。随着 5G 网络的不断发展，SDN 网络的研究热度也进一步提升。

SDN 网络的特点是转控分离和可编程性。转控分离是指 SDN 网络将 IP 网络中的控制功能与转发功能分离，原有转发功能仍由交换机进行，新的交换机称为 SDN 交换机，而原有的计算、控制功能则通过引入新的网络设备，改由控制器承担，实现控制平面与数据平面分离。SDN 的体系架构如图 1-1 所示，在数据平面，SDN 交换机只需要根据给定的转发表将收到的数据流进行转发，处理日益增长的流量需求，在控制平面，控制器收集全网拓扑信息并计算，通过南向接口进行转发表下发，便于进行网络的管理与配置，北向接口则屏蔽了底层设备的差异，为 SDN 网络应用提供资源虚拟化。可编程性是指网络可以通过命令行对设备进行配置，SDN 网络通过开放强大的编程接口，提供了灵活的网络可编程性，使得对流经 SDN 交换机的流量进行控制，实现负载均衡[4]。

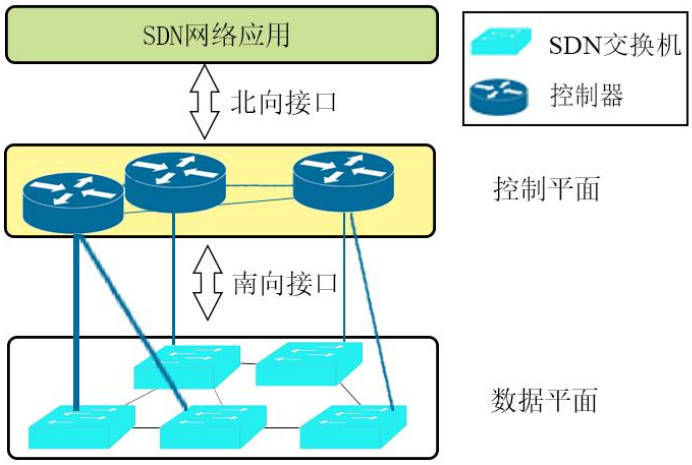


图 1-1 SDN 体系架构

Figure 1-1 The architecture of SDN

SDN 网络的转控分离与可编程特点使得网络可以对流经 SDN 交换机的流量通过控制器进行编程管理，带来灵活的网络配置能力，因此将传统 IP 网络改造升级为 SDN 网络已经成为业界共识，用以利用 SDN 网络的可编程性[5]、网络控制能力[6]、网络灵活性[7]和安全性[8]等。具体来说，将传统交换机升级为 SDN 交换机，可以对流经 SDN 交换机的流量进行编程，改变流量转发路径，以应对网络突发失效或对流量进行负载均衡。现有研究采用可编程路径数[5]、可编程流量[9]、单链路恢复能力[10]等指标衡量升级为 SDN 交换机带来的收益，本文采用已升级节点的度之和衡量，并称其为度收益。为了保证 SDN 交换机的正常工作，还需要引入 SDN 控制器来进行转发表的计算与下发。控制器的引入带来了控制器与交换机之间通信的控制时延以及控制器之间同步网络状态的同步时延，为了衡量控制器带来的时延，采用控制时延与同步时延之和体现控制器引入的损失，称为时延损失。

但是受限于工程造价和建设能力，由传统 IP 网络升级为 SDN 网络不是一蹴而就，而需要几年时间分步完成[11]，例如在文献[12]中，AT&T 网络在 2016 年、2017 年与 2020 年分别将其网络的 34%、55%和 75%升级为 SDN 网络。因此 SDN 网络升级是一个经历多步的升级过程，每一次升级称为一个升级步或时间步，在一次升级时，选择部分交换机升级为 SDN 交换机，并选择部分已升级交换机位置部署控制器，此时网络中会同时存在传统 IP 交换机与 SDN 交换机，该网络称为混合 SDN。图 1-2 为混合 SDN 的示例说明，在数据平面，仅有部分 IP 交换机升级为 SDN 交换机，此时网络为混合 SDN 网络，其中 SDN 交换机可以编程控制流经的流量，因此在升级交换机时，应使得可控流量尽可能多。在控制平面，控制器仅控制 SDN 交换机，SDN 交换机会向控制器发送请求等，控制器也需要向交换机下发转发表，控制器与交换机之间存在控制时延。控制平面还存在多个控制器的情况，此时多控制器之间需要同步信息确保网络的一致性，控制器之间通信存在同步时延。因此，在部署控制器时，应使得控制时延与同步时延尽可能小。

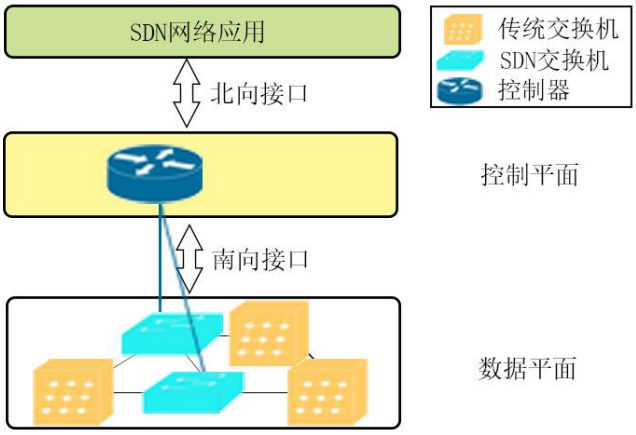


图 1-2 混合 SDN 体系架构

Figure 1-2 The architecture of hybrid SDN

在整个升级过程中，需要确定在哪一步（when）选择哪些交换机（which）进行升级，以及如何部署控制器并确定交换机和控制器之间的连接关系（how）。其中选择交换机和确定升级步是指首先升级哪些交换机，随后升级哪些交换机，需要根据升级带来的收益进行决策；部署控制器和确定连接匹配关系是指在已升级的SDN交换机中，选择其中部分位置部署控制器，并将已升级的SDN交换机分配给控制器，使得每一交换机都被控制器控制，需要根据控制器引入的时延损失进行决策。由于整个升级过程中，when、which以及how问题的求解都与升级利益相关，需要同时进行回答才能最大化整个升级过程的收益。

遗憾的是，据我们目前掌握的资料来看，现有研究的升级方案只回答了以上的一个或两个问题。如文献[13]仅考虑升级过程中的交换机选择问题，即只回答了which问题，文献[14]在纯SDN网络中进行控制器的部署，只回答了how问题。文献[15]考虑完整的SDN升级过程，在每次升级时进行交换机的选择，即同时回答了when和which的问题，但没有考虑控制器部署问题。同时考虑交换机升级与控制器部署的研究只有文献[16]，但该文献只回答了which和how的问题，忽视了升级步之间的关联性，也即没有回答when的问题，并不能得出完整的最优动态升级策略。本文首次对三个问题同时进行回答与求解，得出完整的升级策略。

在整个升级过程中同时考虑并解决上述三个问题，并非易事。待升级的初始网络为纯IP网络，在升级过程中，由于部分传统交换机升级为SDN交换机，网络变为混合SDN[17]，即传统交换机和SDN交换机混合存在的网络。升级完成后，网络演变为纯SDN网络，即只包含SDN交换机的网络。由IP网络过渡为混合SDN网络，再逐步过渡为纯SDN网络的过程存在双向时序约束，使得在升级策略求解时需要同时考虑。双向时序约束具体是指，由传统网络升级为SDN网络是一脉相承的，单步升级过程中部署的控制器会存在于最终纯SDN网络，影响纯SDN网络的性能，即两者存在正序影响；反过来，纯SDN网络是最终的升级目标，为了追求更有利的纯SDN网络布局，会迫使在单步升级过程中，尽可能将控制器部署在全局最优位置上，即两者存在逆时序影响。可以看出，将三个问题同时求解并不是简单地将单步升级策略进行迭代，而需要解决双向时序影响带来的挑战。

本文发现和研究了全局视角下的SDN动态升级问题，也即同时求解升级步确定（when）、交换机选择（which）以及控制器部署（how）问题。但同时求解以上三个问题存在难度，本文首先通过引入退化因子，对问题进行了最优化建模与转化，之后引入惩罚项，将双目标动态优化模型转化为可直接求解的单目标模型。考虑到问题为NP-hard问题，在大规模场景下，提出基于改进遗传算法的启发式算法求解近似解；最后，将升级过程看作马尔可夫决策过程，提出基于强化学习的求解算法，通过自学习得到全局收益对单步局部收益的影响。

1.2 研究现状

本文提出的 SDN 网络动态升级问题需要同时解决升级时间步确定（when）、交换机升级（which）以及控制器部署匹配（how）三个问题，而现有的研究仅考虑了单个或两个问题。其中，仅解决控制器部署的问题（how）被称为多控制器部署问题，仅考虑交换机升级的问题（which）被称为静态升级问题，同时考虑交换机升级与时间步（when & which）的问题被称为动态升级问题，同时求解交换机升级与控制器部署（which & how）的问题被称为联合部署问题。本节将从以上四个研究方向进行研究现状的分析。

(1) SDN 多控制器部署问题

SDN 网络研究之初，主要利用单个控制器进行全网交换机的控制与管理[18]。随着 SDN 网络的发展，网络规模逐步扩大，单个控制器无法满足全网的信息控制与交换机管理，因此，多控制器布局方案被提出[14]。多控制器布局方案是指在网络中，部署多个控制器分别控制网络中的部分交换机，并保证所有的交换机都被控制器控制。多控制器部署可以提升网络处理信息的能力，提高网络的可扩展性，同时避免了单一交换机出现故障时引起整个网络瘫痪的情况，提升了网络的鲁棒性。多控制器布局需要计算得出控制器的个数与部署位置以及控制器和交换机之间的匹配控制关系。因此，如何在 SDN 网络中进行多控制器的部署被广泛研究，这些研究主要从控制器通信开销[19]、可靠性[20-21]、控制器引入的时延[22-23]、负载均衡[24-25]等方面进行控制器的部署，以提高控制器部署带来的网络性能提升，减少控制器引入的弊端。

在考虑网络时延的多控制器布局方案中，文献[22]首次研究了 Internet2 网络中部署控制器所引入的时延，定义了平均时延与最坏时延，发现控制器的部署策略会影响节点和控制器之间的时延，并展示了不同控制器部署方式对平均时延和最坏时延的影响。文献[26]通过最小化控制器与交换机之间的控制时延以及控制器与控制器之间的同步时延，确定控制器的最优位置，并基于 k-means 算法与 Dijkstra 算法提出一种求解的启发式算法。

文献[21]则考虑控制器部署的可靠性，其中可靠性是指网络自身在规定条件下正常工作的能力。作者将整个网络所有节点之间的连通度定义为网络可靠性，通过对平均流规则下发时延、控制器负载方差以及连通度进行建模，提出基于高可靠性的控制器部署模型。在时延满足服务质量的前提下，以最大化交换机到控制器的平均可靠度为目标，求解控制器的最佳个数、部署位置以及管辖的交换机区域。实验结果表明，所提出的算法较于随机算法与粒子群算法，可靠度更优。

文献[20]定义了控制开销作为控制器部署的新指标，即控制器对交换机进行控

制产生的通信开销，包括统计信息收集开销，下发规则开销以及控制器同步开销。将最优化目标设定为最小化通信开销来计算控制器部署位置，也即在至多部署k个控制器的情况下，找到最优的控制器部署方式使得通信总开销最小。作者在真实网络拓扑中进行实验，实验发现通过最小化时延部署的控制器位置比随机部署位置的控制开销减少了 60%，并论证了随着控制器数目的增加，控制开销逐渐减少，但随着控制器数目的增多，减少的速度变慢，带来影响减少。

文献[27]主要考虑安全性问题，作者通过考虑链路故障的场景下进行控制器的部署提升 SDN 网络的安全性，通过在单链路故障和多链路故障下分别进行 SDN 网络控制器的部署，利用遍历算法与蒙特卡罗算法搜寻最优解得到部署策略。

除了以上对单指标的考虑外，不少研究还对多个指标同时进行考虑，文献[28]同时考虑时延和可靠性，提出了基于最小覆盖的控制器部署算法（MCC），在使用最少控制器的条件下，满足时延和可靠性需求，通过与其他算法对比，验证了 MCC 算法相较于传统算法，具有最好的可靠性。文献[29]也同时考虑了网络控制时延和可靠性两种性能指标进行控制器部署，用节点不可达百分比期望值作为可靠性指标，在控制网络可靠性最优的情况下，遍历所有解的时延，选择时延最优解。得到基于时延约束的高可靠性控制器部署模型。文献[30]则同时考虑控制时延与负载均衡，并通过对 k-means 算法进行改进，以适应多控制器部署的场景。改进的 k-means 算法首先对网络进行初步分类，使得平均控制时延最小，之后使用模拟退火算法重新分配处于边界的节点，使得控制器之间达到负载均衡，最后通过标签传递算法找出未被分配的点，进行重新分配，解决跨域通信问题。最终仿真结果表明，该算法能够使平均控制时延最小，同时控制器负载差异度最小。文献[31]同时考虑控制器和交换机之间的最差控制时延、控制器之间的最差同步时延、负载均衡以及对节点故障的容错性。在满足时延需求的前提下，选择负载均衡与容错性更高的点作为控制器部署位置。

在算法求解过程中，以上文献大多搭建最优化模型并直接求解进行控制器部署位置的计算。此外，还有文献使用聚类[32]、二部图匹配[33]、博弈论[34]、遗传算法[35]等方法进行求解。

聚类算法是将网络根据某一指标聚为与控制器个数相同的类数，每一类为一控制器管辖域，选择聚类中心点或其他位置点作为控制器部署位置。如文献[32]提出利用 k-means 算法，在最小化子网内控制时延的目标下，将网络划分为多个子网，每个子网作为一个控制器管理域，在子网内选择最小化控制时延的点作为控制器。文献[36]主要考虑控制器与交换机之间的控制时延，引入控制器容量、交换机与控制器关联等限制条件，建立最优化模型，所建立的模型为非线性整数规划问题，难以求解，作者将该问题分解为控制器与交换机关联子问题和控制器容量匹配子

问题，并进行基于距离的聚类求解，得到控制器部署和容量匹配策略。文献[37]针对扁平化的多控制器部署问题提出了基于密度聚类的控制器部署算法，先计算传输的控制时延和通信开销的决策图，之后通过寻找决策图拐点的方法计算最佳控制器数量，将大于拐点值的位置作为最优控制器布局点。最后通过实验仿真表明这一方法在最大时延、平均时延、最大消耗和平均消耗等方面都有较优的性能。

此外，还有文献采用博弈论的方法进行控制器布局。文献[34]使用博弈论与 k-means 算法进行控制器部署问题的求解。作者提出了一种基于网络划分的控制器布局策略，称为合作 k-means 算法，首先利用 k-means 算法将网络划分成多个子网，之后利用合作博弈论的思想，将所有子网作为博弈的参与者，通过试图与其他子网中的交换机进行联盟，实现自身价值的最大化，也即实现最小化控制时延。

文献[38]建立最优化模型，将最优化目标设为最小化控制时延、同步时延与负载均衡指标，使用基于变异函数的粒子群优化多目标遗传算法求解最优化问题的帕累托最优解。文献[39]利用遗传算法研究了 SDN 网络中控制器与交换机匹配关系的分配问题，通过最小化交换机的平均事件响应时间，建模最优化模型，确定匹配关系。

(2) SDN 静态升级问题

SDN 网络静态升级问题是指在传统 IP 网络中，选择部分交换机进行升级，得到混合 SDN 网络，使得整个网络在不需要全部升级为 SDN 交换机的情况下，就可以享有类似 SDN 网络的控制和管理[40]。这一研究问题是在整个升级过程中只考虑一个固定的时间步，在该时间步下，进行交换机的升级。现有研究大多将升级后混合 SDN 网络的性能作为升级目标，进行交换机的选择，包括网络可编程性[40]、流量工程[41]、故障恢复[42]、资源消耗[43-44]、安全问题[45]等。

文献[5]将混合 SDN 网络的可编程性作为网络升级的指标，SDN 交换机的引入使得通过 SDN 交换机的通信流量可以根据控制器的转发表规则进行控制，以进行拥塞避免，维护网络安全等，因此作者将最大化网络的可编程路径数作为升级的目标，进行交换机的选择。文献[9]也选择 SDN 网络的可编程性作为网络升级指标，不同的是，作者除了考虑可编程路径数外，还考虑了每条路径上的流量，将网络可编程性定义为可编程流的数目，将原有的每条路径上流量相同的约束去掉，进一步扩展了模型的适用范围。

文献[6]主要考虑控制器的影响，在企业网络中，由于企业预算有限，只能单次升级部分交换机为 SDN 交换机，形成混合 SDN 网络，升级交换机时主要考虑控制器对整个网络流量的控制范围尽可能的大。因此作者将混合 SDN 部署问题定义为一个受预算、链路容量、SDN 交换机流表大小和传统交换机 VLAN 大小约束的优化问题，其中最优化目标是使 SDN 控制器尽可能多的控制有最短路由流经的交

换机，以充分发挥控制器对流量的管控作用。

文献[10]主要从故障恢复的角度进行交换机的升级，由于 SDN 交换机可以在控制器的控制下，将数据通过非最短路进行转发，因此对于单点链路故障的情况，可以在经过 SDN 交换机时利用非最短路进行转发，对数据起到保护作用。因此作者提出交换机升级的又一指标，单链路故障恢复，通过在网络中选择最少数量的 IP 交换机升级为 SDN 交换机，以保护所有的单链路故障。同时，在满足单链路故障恢复的条件下，最小化恢复路径长度，以减少重路由报文的延迟，并节省网络带宽。

文献[7]考虑 SDN 网络控制器的能力与网络灵活性，作者在最小化成本的前提下，最大化网络控制能力和网络灵活性，通过定义网络中流经 SDN 交换机的流量为网络控制能力，通过引入偏心来表示网络灵活性，偏心值越小，该点到其他点距离越近，会有更多的流量流经该点，因此该点更适合被升级为 SDN 交换机。

(3) SDN 动态升级问题

SDN 动态升级问题是指将升级过程看作整体，确定每一升级步选择哪些交换机，最终形成完整的升级策略。受限于预算、技术和网络所面临的架构变更风险等，整个动态升级过程会在几年时间内，分多个时间步逐步进行[11]。在升级过程中所使用的指标包括可编程路由数[46]、升级开销[15]、SDN 可迂回路径数[13]、用户 QoE[25]等。

文献[13]论证了从 IP 网络过渡为 SDN 网络需要经过多年的时间，并指出升级过程不应该一步完成，将最优化目标设定为首先最大化可编程流量，之后最大化可迂回路径数来考虑流量工程的灵活性。

文献[47]将 SDN 网络升级问题建模为流量工程最优化问题，通过最小化最大链路利用率，利用遗传算法和贪婪算法搜索近似解，得到完整的升级过程。

文献[48]从流量工程的角度出发，最大限度地减少链路利用率，得到由传统网络设备升级为 SDN 交换机的最优增量部署顺序，从而确定每一步升级时最合适的迁移设备。在研究过程中，将资源和预算约束作为约束条件，在维持网络的负载均衡的同时最小化所升级的交换机数目。最终结合粒子群优化算法和蚁群优化算法，在每个迁移阶段根据路由器的位置、最优权值设置和流量分割率来确定该最优序列。

(4) SDN 联合部署问题

SDN 联合部署问题是指由 IP 网络升级过渡为 SDN 网络过程中，在某一升级步下，同时考虑交换机升级和控制器部署问题。在初期研究中，由于网络规模较小，只需要一个控制器就可以实现全网的控制，因此以往的研究只考虑了交换机的选择问题，而随着网络规模不断增大，多控制器部署问题被提出，因此有少部分研究在升级过程中同时考虑交换机升级和控制器部署问题，也即联合部署问题。

文献[16]同时考虑交换机的升级与控制器的放置，将混合 SDN 网络升级建模为双目标优化问题，在最大化可编程流的同时，最小化控制时延，并将升级预算作为约束，构建最优化问题，利用最优化求解器直接求解这一整数线性规划问题，得到最优的单步升级策略。

在 SDN 网络动态升级策略研究中，静态方法主要考虑在给定网络拓扑下，如何选择部分交换机使得升级后的混合 SDN 网络收益最大，动态方法主要考虑整个升级过程中交换机的选择，目的在于整个升级过程的收益。据我们掌握的资料，只有一篇研究工作[16]同时考虑了交换机升级与控制器部署问题，进行 SDN 网络的联合部署研究，但是并考虑到所处升级时间步的影响，只是将单步升级方案迭代到后续每一升级步中。

综上所述，现有工作缺少对完整 SDN 网络升级轨迹同时进行交换机升级和控制器部署的研究。而在 SDN 网络升级轨迹的研究中，同时求解交换机升级、时间步确定以及控制器部署三个问题会使得研究更为复杂。

1.3 论文工作和组织结构

本文将从全局视角出发，进行 SDN 网络的最优动态升级策略研究，同时求解时间步确定（when）、交换机选择（which）以及控制器部署（how）问题。本文首先将 SDN 动态升级问题建模为包含时间因素的双目标动态优化问题，在每一升级步，同时考虑交换机升级的度收益与控制器部署的时延损失，为了求解这一问题，引入惩罚项和偏好因子将优化问题转化为一系列可以直接求解的整数线性规划问题，同时考虑单步升级过程中的局部收益以及最终纯 SDN 网络的全局收益，在小规模网络下直接进行求解。在大规模网络场景下，该问题属于 NP-hard 问题，无法直接求解，通过对遗传算法进行改进，提出基于改进遗传算法的启发式算法，求解大规模场景下的近似解。最后，针对 SDN 升级问题与强化学习算法的契合度，提出利用强化学习进行升级问题的求解，在小规模网络和大规模网络场景下，分别利用 Q-learning 算法与 DQN 算法进行求解。具体来说，本文完成了以下三方面的工作。

(1) 将最优动态升级问题建模为一个双目标动态优化模型，在整个升级时序中（when），同时优化交换机升级（which）与控制器部署（how）的收益。通过引入随时间变化的退化因子，刻画局部双目标到全局单目标的变化过程。通过引入惩罚项将双目标优化问题转换为可以直接求解的整数线性规划问题，在小规模网络下，直接进行问题的求解，并验证建模的可行性与有效性。

(2) 在大规模网络中，提出启发式算法求解近似解。通过改进遗传算法，得到

基于 SDN 匹配关系的遗传算法（SM-GA）和基于互补交叉变异的遗传算法（CCM-GA）分别求解惩罚项与最优动态升级策略。

(3) 将升级过程看作马尔可夫决策过程，提出了基于强化学习的求解算法。在小规模网络下，利用 Q-learning 算法进行价值计算，在大规模网络下，引入深度神经网络，利用 DQN 算法进行升级问题的求解。

本文的结构安排如下：

第一章 论述 SDN 网络最优动态升级策略的研究背景与意义，介绍相关工作的研究现状，说明本文是对现有研究空白的补充。

第二章 介绍本文研究过程中用到的理论与技术，包括 0-1 规划、遗传算法与强化学习。

第三章 介绍问题提出的动机，根据问题特点进行问题建模与模型转换，得到可直接求解的最优化模型，并在小规模网络中进行算法验证。

第四章 针对第三章所得模型，提出大规模场景下的启发式算法，通过改进遗传算法，得出可求解大规模场景下 SDN 网络动态升级策略的启发式算法，并进行验证分析。

第五章 针对第三章所提问题的马尔可夫性，提出基于强化学习的求解算法，将 SDN 网络动态升级问题映射为强化学习模型，并基于 Q-learning 算法与 DQN 算法进行问题求解。

第六章 总结本文工作，指出下一步研究的可行方向。

2 相关技术及理论基础

2.1 0-1 规划

0-1 规划是一类特殊的整数规划，其决策变量的取值为 0 或 1，通常是线性规划。其中，整数规划中所有决策变量的取值只能是整，而线性规划是指目标函数与约束条件均为线性的规划问题。0-1 规划可以方便地描述二元逻辑关系，例如开关、有无等。因此 0-1 规划的应用十分广泛，适用于解决工厂选址、人员安排等问题[49]。

0-1 规划问题的解法一般考虑穷举法和隐枚举法，线性规划的解法一般考虑单纯形法，对于小规模问题，在问题规模及有效时间允许的情况下可以采用分支定界法等求解。由于 0-1 线性规划问题可以看作是混合整数线性规划（Mixed Integer programming, MIP）的特殊形式，现有的商用线性规划求解器可以在小规模网络下直接求解，因此采用商用求解器直接求解是一种高效的方式。而在大规模网络下，整数规划问题属于 NP-hard 问题，这就决定了随着决策变量个数 n 的增大，0-1 规划问题的计算量将急剧增加。目前，学者们普遍采用启发式算法来处理大规模问题。

指派问题是 0-1 规划的特殊情况，也称为分配问题[50]。指派问题最初是指人员和任务之间的安排问题，在已知人员列表和任务列表的情况下，通过指派哪个人完成哪项任务，使得完成任务的时间最短或收益最大。在指派问题中，还可能存在效率矩阵不是方阵的情况，这是因为人员和任务不一定是一一对应的，可能出现人多任务少或者人少任务多的情况，此时需要对问题做出适当的改进，再进行求解。具体改进方法是，当效率矩阵不是方阵时，可以通过虚设几列或几行，转变为方阵，虚设的元素根据实际情况进行填充。

由于指派问题既可视作 0-1 规划，又可以看作是运输问题，所以有很多解法，其中匈牙利数学家科尼格根据指派问题的特点，给出了求解的一种便捷有效且常用的算法，称为匈牙利算法[51]，该算法可以在多项式时间内进行求解。匈牙利算法的基本原理是，求解指派问题可以看作是在二部图内求解最大匹配问题，进而可以看作是在二部图中不断寻找增广路径，直至图中没有增广路径为止。

2.2 遗传算法

遗传算法（Genetic Algorithm，GA）是美国教授 J.Holland 基于自然选择理论与遗传基因的机理提出的一套相对完整的算法理论体系[52]。遗传算法以概率搜索

为基础，通过模拟自然环境中交叉变异、优胜劣汰的机制进行最优解的求解，具有良好的自适应能力与强大的搜索能力，同时还有隐含并行性和全局搜索特性两大显著特征，能够在解空间内高效的搜索并且通过选择机制不断逼近最优解。在组合优化、生产调度、自动控制、图像处理等领域上都有应用[53]。

遗传算法的流程如图 2-1 所示，包括编码、产生初始父代、交叉与变异、适应度评价与择优选择、终止条件等部分。首先将待求解问题的可行解转化为遗传算法可以处理的形式，对问题进行编码。之后种群产生初始父代，并对父代中的个体进行交叉和变异操作，产生新的个体，并对新生成的种群根据适应度函数进行子代择优，在新的择优子代内再次重复交叉变异与择优的过程，循环的终止条件一般情况下可以为遗传代数或是前后两次种群的差异等。当循环终止时，遗传结束，输出具有最优适应度函数值的个体作为最优解，完成遗传算法。在实际利用遗传算法求解时，需要根据问题的特点对基因进行编码，并设计交叉变异规则，选定适应度函数。

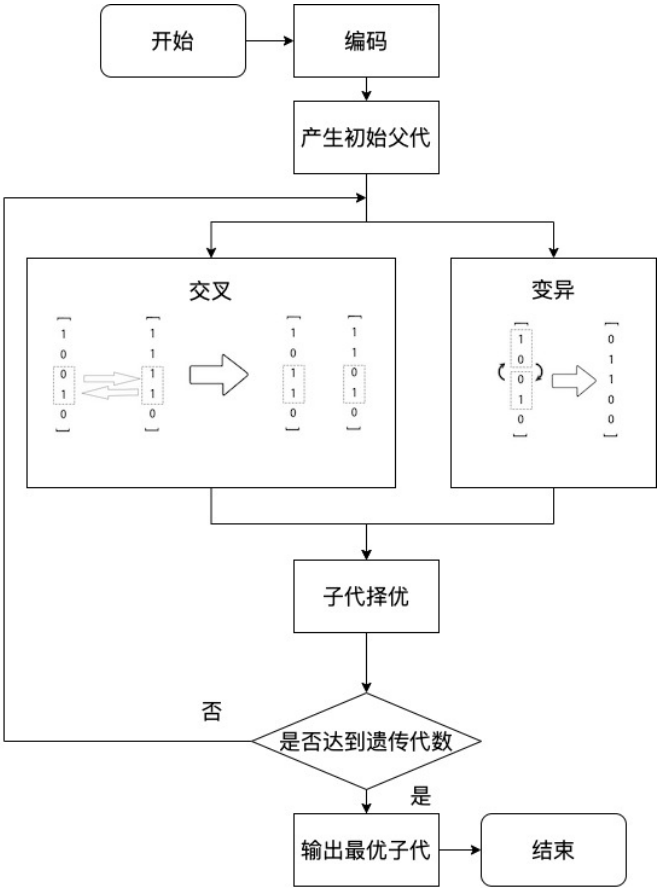


图 2-1 遗传算法流程

Figure 2-1 Genetic algorithm flow

遗传算法具有两个鲜明的特点，其一是在基因层面的染色体遗传规律，包含染色体的交叉、变异等操作，实现了基因片段的交换。其二是在种群层面上的利用“优胜劣汰，适者生存”的自然规律进行筛选。前者实现了在解空间内随机产生一些新的可行解，并且这些可行解蕴含优秀的基因片段；后者依据个体的适应度，在种群中筛选出优秀的个体。二者的共同作用可以有效的在解空间内不断的搜索更逼近最优解的可行解。

2.3 强化学习

2.3.1 马尔可夫决策过程

马尔可夫决策过程（Markov Decision Process, MDP）是指在具有马尔可夫性的环境中序贯做出决策的过程。其中马尔可夫性是指某一状态蕴含了所有相关的历史信息，即当前状态可以决定未来，就认为该状态具有马尔可夫性，具有马尔可夫性的随机过程称为马尔可夫过程，对马尔可夫过程序贯地做出决策就称为马尔可夫决策过程。马尔可夫决策过程为强化学习问题提供了理论框架，智能体在环境中根据每个时间步观察到的状态s，从可用的行动集合中选取一个动作a，环境在a的作用下，转换至新状态s'，再重复以上过程。由于马尔可夫过程的无后效性，下一时间步状态s'仅与当前状态s和a有关，此刻之前的状态和动作不对其有任何影响[54]。

在求解马尔可夫决策过程时，需要用到贝尔曼方程（Bellman Equation），贝尔曼方程描述了当前值函数和它的后继函数的关系，这样就可以通过迭代的方式求解值函数，进而引出强化学习中的 Q-learning 算法。贝尔曼方程是价值函数的期望，如公式（2-1）所示，价值函数的期望可以分解为两部分，第一项是该状态下，做出动作的立即奖励的期望，该项是常数项，所以即时奖励的期望等于即时奖励本身，第二项是下一状态值函数的折扣期望，表示未来收益对当前收益的折现。利用贝尔曼公式可以通过不断迭代得到状态—动作对的值函数[55]。

$$\begin{aligned} Q_\pi(s,a) &= E_\pi[G_t|S_t=s,A_t=a] \\ &= E_\pi[R_{t+1}+\gamma Q_\pi(S_{t+1},A_{t+1})|S_t=s,A_t=a] \end{aligned} \quad (2\text{-}1)$$

2.3.2 强化学习的基本概念

强化学习通过模仿人类和动物学习的试错机制，与环境进行交互，学习状态到行为的映射关系，以获得最大累积期望回报[56]。在强化学习模型中主要包括智

能体和环境两大对象。智能体也被称作代理，是在环境中进行操作和读取的个体，环境则是可以与智能体进行交互的外部因素。

智能体与环境的关系如图 2-2 所示，智能体从环境中读取到当前状态，在环境中选择动作进行执行，使得环境变为下一状态，并反馈给智能体一个奖励，如此循环下去，智能体与环境不断进行交互，得到更多的数据，并利用数据改善智能体自身的行为。智能体在环境中并不会被告知确定性选择哪个动作，只能通过不断尝试动作，获得环境对动作的反馈，改善自己的行为，以适应环境。

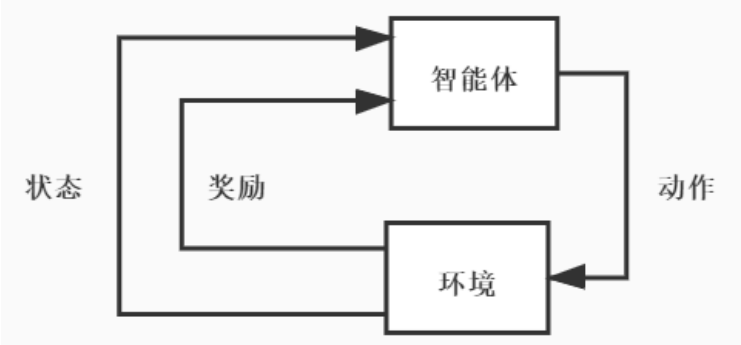


图 2-2 强化学习示意图

Figure 2-2 Reinforcement learning schematic chart

强化学习中包含三个主要部分，分别是状态、动作和奖励。

(1) 状态：状态是指智能体所处的环境信息，可以用向量、矩阵或是像素等表示，所有可能的状态构成状态空间。

(2) 动作：动作是智能体在环境中感知到环境状态后要采取的行动，需要在动作空间中选择，选择后环境改变为对应的下一状态。

(3) 奖励：智能体完成动作后，环境根据实际场景定义的奖励机制，对智能体的动作给出即时奖励，包括正向奖励和负向奖励。正向奖励会激励智能体趋向于学习该动作，负向奖励则相反。

智能体经过在环境中不断试错，决定自己的策略和值函数，策略是决定智能体行为的机制，是状态到动作的映射，用$\pi(a|s)$来表示，它定义了智能体在各个状态下的各种可能的行为方式和概率大小。强化学习的目标就是通过收集到的各种试错信息得出策略函数。值函数表示在不同状态下采取某个动作的好坏程度，是智能体得出策略的依据，如选择值函数最大的动作就可以看作是一个策略。状态行为价值函数表示在执行策略π时，对当前状态s执行某一具体行为a所获得的期望回报，包括当前动作带来的即时收益与未来预期收益对当前动作步的折现。

2.3.3 Q-learning 算法

Q-learning 算法是利用时序差分的离线策略，是模拟试错行为的算法[57]。智能体在特定状态下不断尝试各种动作，根据它收到的即时奖励以及所处状态来评估后果，将后果记录为这一状态—动作对的价值。通过反复尝试所有状态下的所有动作，智能体可以为自身建立一张状态—动作价值表，将每一状态下，不同动作的价值进行记录。在得到状态—动作价值表后，可以在所处状态下选择执行价值最高的动作，完成动作的选择。

其中，价值函数的计算公式如公式（2-2）与公式（2-3）所示，为了计算状态—动作对的价值，需要利用迭代函数，将当前动作的奖励，与潜在的未来奖励对当前行动的影响进行加权求和。

$$\begin{aligned} Q^{\pi}(s,a) &= E[r_{t+1} + \lambda r_{t+2} + \lambda^2 r_{t+3} + \cdots | s,a] \\ &= E_{s'}[r + \lambda Q^{\pi}(s',a') | s,a] \end{aligned} \tag{2-2}$$

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha(R_{t+1} + \lambda \, max_a \, Q(S_{t+1},a) - Q(S_t,A_t)) \tag{2-3}$$

在价值计算时，需要先收集升级策略，利用收集到的策略，执行动作并更新 Q 值，也就是状态—动作对的价值。其中策略的生成方式有两种，一种是随机生成动作；一种是根据当前的 Q 值表，计算出最优的动作。其中，使用随机动作的方案是探索（exploration），可以探索未知的动作会产生的效果，有利于更新 Q 值，获得更好的策略。而选择最优动作的方案是利用（exploitation），可以更好的利用已有的经验。将两者结合起来就是$\varepsilon - greedy$策略，ε一般是一个很小的值，作为选取随机动作的概率值。可以更改ε的值从而得到不同的 exploration 和 exploitation 比例。

Q-learning 在状态数和动作数较少时可以用表格列举出每一状态下执行每一个动作所获得的奖励，根据 Q 值更新公式，多次迭代计算更新表格中的值，也就是所有的状态—动作的价值。最后，决策者可以通过表格在所处状态下，选择执行当前价值最高的动作，反复执行，直至到达最终状态，获得最优结局。

综合以上推导分析过程，Q-learning 算法的训练过程可以描述为以下步骤。首先初始化 Q-learning 状态表，也即 Q 值表（Q-table），将状态表的动作价值随机初始化。之后通过训练更新价值，也即让智能体在每一状态下，或从初始状态开始，以某一策略进行动作选择，重复选择动作，观察奖励，进入新状态的过程，并利用公式（2-4）对 Q 值进行更新，其中，r_t是从状态s_t移动到状态s_{t+1}时收到的奖励，α是学习率，$Q(s_t,a_t)$是更新前的值，$max_a Q(s_{t+1},a)$是信息，通过多次学习，得到训练后的 Q 表。

$$Q^{\pi}(s_t,a_t) \leftarrow (1-\alpha) \cdot Q(s_t,a_t) + \alpha \cdot (r_t + \gamma \cdot max_a Q(s_{t+1},a)) \tag{2-4}$$

2.3.4 DQN 算法

利用 Q 值表存储价值来实现强化学习的方法会随着状态数的增多，导致 Q 值表变大，此时查询表的时间会非常长，且占用内存，Q-learning 算法在实时性上不再有意义。为了解决这一问题，可以使用神经网络近似函数来代替 Q 值表，不仅解决了状态数过多的问题，而且可以求解连续状态情况[58]。

深度 Q 网络（Deep Q Network，DQN）是一种将神经网络和 Q-learning 结合的方法，是一种无监督的机器学习算法，其核心是用神经网络$q(s,a;w), s \in S, a \in A$来代替动作价值函数。DQN 在 Q-learning 的基础上做出了三方面的改进。一是使用卷积神经网络近似逼近动作值函数，即用神经网络代替 Q-table，计算每个状态—动作对的价值；二是使用了经验回放，将历史的状态、动作、奖励等作为经验存储起来，再在存储的经验中按一定的规则采样；三是在评估网络的基础上，引入目标网络 target-Q，并修改网络的更新方式，不把刚学习到的网络权重马上用于后续的自益过程[58]。这些改进以及$\varepsilon - greedy$策略的引入解决了大规模场景或连续状态场景下存在的问题。

DQN 的算法流程如图 2-3 所示。首先随机初始化价值函数的权重，初始化状态序列和动作序列。之后在当前状态s下以ε的概率随机选择动作a，或以$1-\varepsilon$的概率选择当前状态下有最大价值的动作a，执行动作后，在环境中观测得到奖励r和下一状态s'，将(s,a,r,s')作为训练样本用于经验回放，并将训练样本存储到经验池 D 中，当存储到一定数量时，存储池进行回放。如果该动作后没有下一状态，将得到的奖励值作为神经网络训练的目标值y，否则利用贝尔曼方程，迭代得到当前状态的实际价值。最后通过梯度下降进行模型训练，更新神经网络参数。每隔一段时间，同步更新目标网络参数。

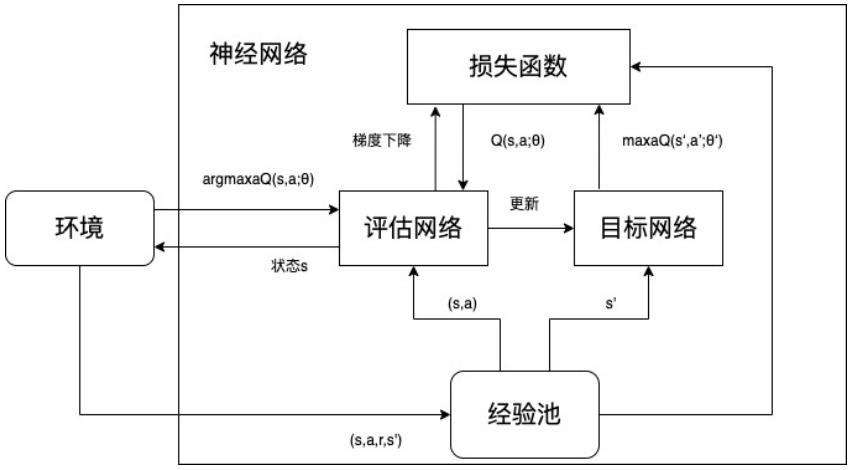


图 2-3 DQN 算法流程

Figure 2-3 DQN algorithm flow

2.4 本章小结

本文所研究的问题属于 0-1 规划，因此首先介绍 0-1 规划与整数线性规划，说明 0-1 规划问题的特点与常见解法。之后对遗传算法进行了介绍，说明了遗传算法的适用场景与范围，以及遗传算法的求解过程。最后介绍了强化学习算法，通过马尔可夫过程说明强化学习算法的原理，并介绍了强化学习中常用的两种算法，Q-learning 算法与 DQN 算法。

3 全局视角下的 SDN 最优动态升级问题

本章首先提出 SDN 最优动态升级问题，通过示例网络与理论分析说明本文的研究动机与难点，并对 SDN 最优动态升级问题进行完整描述。之后将 SDN 最优动态升级问题进行最优化建模，通过引入惩罚项与偏好因子，将其转化为可直接求解的整数线性规划模型。最后，在小规模真实网络拓扑中，求解转化后的最优化模型，并与基线算法进行对比，验证本章所提模型的有效性。

3.1 问题提出

SDN 最优动态升级策略需要同时解决在哪一时间步（when）、选择哪些交换机（which）以及如何部署控制器（how）的问题。在整个升级过程中，网络由传统 IP 网络，过渡为混合 SDN 网络，最终升级为纯 SDN 网络。3.1.1 小节将用示例网络说明研究动机，3.1.2 小节将对问题进行抽象，提出待求解问题的假设与约定，对问题进行描述。

3.1.1 研究动机

本小节将首先介绍 SDN 网络升级问题，之后通过示例网络说明同时求解以上三个问题存在的难点。

SDN 网络升级问题是将传统 IP 网络升级为 SDN 网络的过程，如图 3-1 所示。在升级过程中，基于现有网络拓扑，所有的传统 IP 交换机将升级为 SDN 交换机，并在某些节点部署控制器，用来控制和管理交换机，维持网络的正常运行。

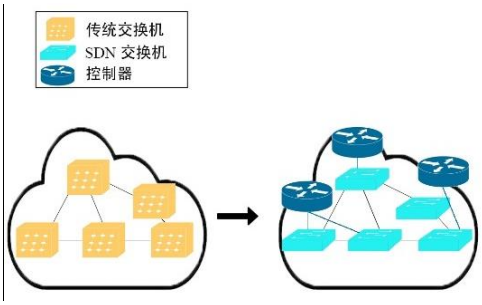
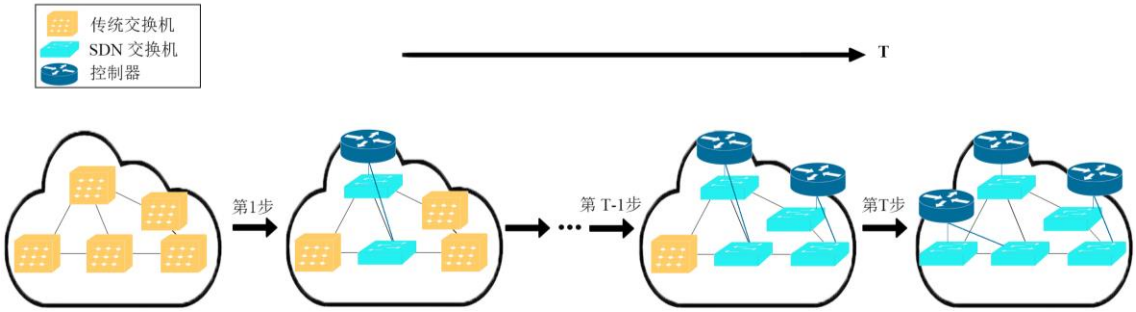


图 3-1 传统 IP 网络与 SDN 网络

Figure 3-1 Traditional IP network and SDN network

在网络升级工程中，受到资金、工期等的限制，SDN 的升级往往不是一步到位，而是分多步进行的，如图 3-2 所示。整个过程可以用多步时间序列表示，在每

一升级步，需要基于当前的网络拓扑，选择某些传统交换机升级为 SDN 交换机，并且选择一些已经升级为 SDN 交换机的节点位置放置控制器，之后为每一个 SDN 交换机分配控制器。图 a)所示的网络中只有传统交换机，是 IP 网络；通过在 a)中选择部分交换机进行升级，并部署控制器，网络升级为传统交换机与 SDN 交换机共存的网络，即混合 SDN 网络(hybrid-SDN network)，如 b)与 c)所示；最终，所有的交换机都升级为 SDN 交换机，网络最终演变为纯 SDN 网络(pure-SDN network)，如 d)所示。



a) IP 网络　b) 混合 SDN　c) 混合 SDN　d) 纯 SDN

a) IP network　b) Hybrid SDN　c) Hybrid SDN　d) Pure SDN

图 3-2 SDN 网络分步升级示意图

Figure 3-2 Migration process of SDN

在升级过程中，需要考虑在哪一步（when）升级哪些交换机（which）以及如何部署控制器（how），使得整个网络升级过程获得最大的收益。为了保证整个升级过程的收益，需要制定合适的动态升级策略，追求升级收益的最大化。其中升级收益包括单步升级过程中混合 SDN 的收益，以及升级完成后纯 SDN 的收益。

混合 SDN 网络需要进行交换机的选择以及控制器的部署，本文定义最大化节点度收益为交换机选择的目标，定义最小化控制器引入的时延为控制器部署的目标，因此单步升级需要同时最大化升级节点度收益与最小化控制器引入的时延，该局部目标是双目标，称为局部双目标，并将混合 SDN 网络的性能称为单步收益或单步利益。纯 SDN 网络中所有交换机均为 SDN 交换机，不再需要考虑节点度收益，因此纯 SDN 网络下仅需要考虑控制器部署的目标，即最小化控制时延与同步时延之和，该全局目标是单目标，称为全局单目标，并将纯 SDN 网络的性能称为全局收益。

节点的度收益是所有已升级为 SDN 交换机节点的度之和，体现了升级这些交换机带来的网络可编程性能提升。受益于 SDN 的可编程性，SDN 交换机在转发数据时可以根据当前网络状态自由地选择转发路径，将 SDN 交换机放置在信息交换频发、可选择路由更多的节点上会带来更大的收益，也即 SDN 交换机节点的度之和越大，收益越大。因此将最大化节点度之和作为交换机升级的目标。

控制器引入的时延包括控制时延和同步时延，体现了网络针对请求的响应速度。交换机需要向控制器请求转发表，控制器需要计算转发表并下发给交换机，控制器与交换机之间的通信时延称为控制时延；同时控制器之间需要进行通信来同步网络拓扑等信息，控制器与控制器之间的时延被称为同步时延。两种时延都会影响网络的整体通信效率，因此将最小化控制时延与同步时延之和作为控制器部署的目标。

可以看出，整个升级过程中网络由 IP 网络过渡为混合 SDN 网络，最终成为纯 SDN 网络，升级目标由混合 SDN 网络的双目标变化为纯 SDN 网络的单目标。这两个网络的收益并非独立无关，而是相互影响的，需要在升级过程中同时考虑。具体来说，单步升级过程中部署的控制器会留存于最终纯 SDN 网络中，影响后续 SDN 网络的时延性能，所以局部双目标与全局单目标存在正向时序影响；而最终纯 SDN 网络的性能是整个升级过程的最终目标，驱使单步升级过程中，将控制器尽可能部署于最终全局最优位置上，因此两个网络的目标还存在逆向时序影响。如果直接按照正向时序，仅考虑单步升级收益，依序决定每一升级步升级哪些交换机和部署哪些控制器，会忽视逆向时序影响，最终可能导致升级完成后的纯 SDN 网络表现性能不佳，影响未来十几年的网络运营状态。

接下来通过一个示例网络分析说明，由于上述局部和全局目标在正逆两个时序两个方向上的相互制约关系，同时回答“什么时候、选择哪个、如何部署”三个问题存在的难点。示例网络如图 3-3 所示，网络包含 16 个节点，23 条边，其中边上的数字代表数据经该边转发的时延，单位为 ms。为了对比升级方案的优劣，首先根据时延最短计算出全局最优控制器为：D、E、K，如图 3-4 所示，且该布局的控制时延为 102.5 ms，同步时延为 53 ms，总时延为 155.5 ms，升级结果如表 3-1 所示。在升级时规定，每次只能升级 6 个节点，放置 1 个控制器，每个控制器只可以控制 6 个交换机，且控制器的部署位置在交换机处选择，即物理空间上，控制器和交换机放在一起。

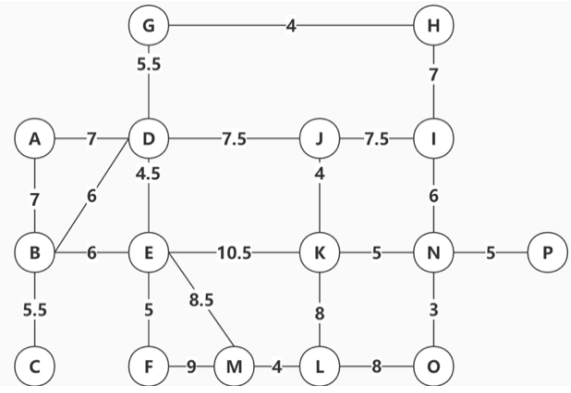


图 3-3 示例网络拓扑

Fig 3-3 Topology of example network

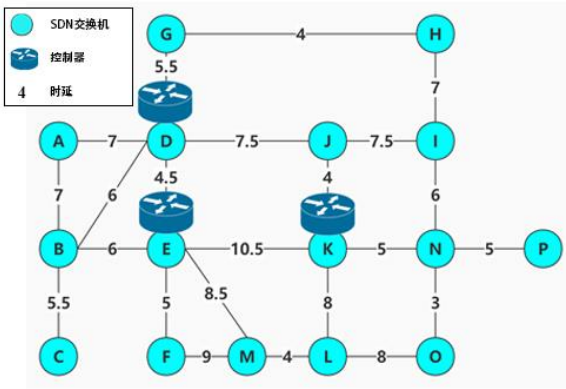
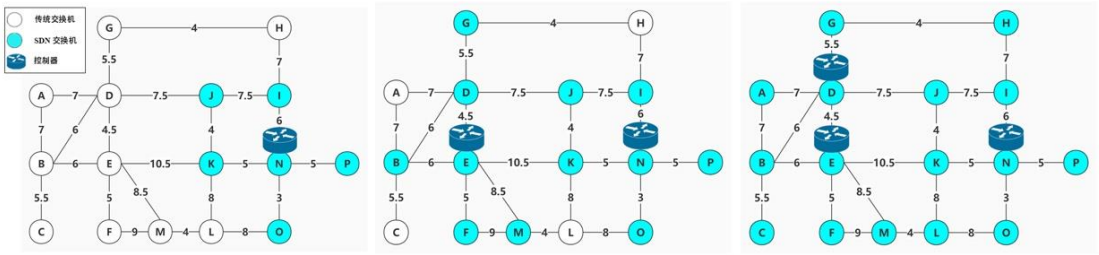


图 3-4 示例网络最优控制器布局

Figure 3-4 Global optimal controller layout of the example network

为了展示仅考虑单步升级利益存在的问题，在示例网络中按照时间顺序，依次在每一升级步内，最大化单步升级的度收益与最小化控制器引入的两种时延之和，也即采用文献[16]中的联合部署方案，依序求解最优结果，得到完整升级方案。所得的升级结果如表 3-1 与图 3-5 所示，升级后的 SDN 交换机用蓝色的点表示，控制器用图标表示。

该升级方案最终选择的控制器为 D、E、N，控制时延为 90.5 ms，同步时延为 73 ms，总时延为 163.5 ms。而全局最优控制器布局为 D、E、K，总时延为 155.5 ms。可以看出，追求单步最优得到的结果并非全局最优。若想达到最优时延，需要对现有控制器进行拆除，在全局最优位置进行重建，这会导致成本增加，若不拆除非最优控制器，则需要忍受非最优布局带来的额外通信时延。



a) 第一步　　b) 第二步　　c) 第三步

a) Step 1　　b) Step 2　　c) Step 3

图 3-5 示例网络在联合部署方案下的逐步升级图

Figure 3-5 Migration process of example network using the Joint Deployment

表 3-1 示例网络升级结果对比

Table 3-1 Comparison of results for example network

	部署结果	控制时延(ms)	同步时延(ms)	总时延(ms)
最优布局	D、E、K	102.5	53	155.5
逐步升级布局	D、E、N	90.5	73	163.5

通过示例网络的升级结果可以得出，SDN 网络升级过程中，升级目标由双目标转化为单目标，且升级过程中，局部双目标与全局单目标之间的影响是双向的。如果只考虑正序影响，将导致单步升级的利益与最终升级完成后纯 SDN 网络的利益存在冲突，得到的纯 SDN 网络控制器布局并非最优解。而纯 SDN 网络是升级的最终目标，将会维持未来 10-20 年的正常运行，所以希望在升级过程中同时考虑单步利益与升级完成后纯 SDN 网络的利益，即同时考虑局部双目标与全局单目标。

3.1.2 问题描述

本小节将 SDN 最优动态升级问题描述为数学问题，并约定升级问题需要满足的约束条件。

首先约定 SDN 最优动态升级问题中的网络拓扑。将网络拓扑抽象为一个加权无向图，用$G(N,E,\omega(E))$来表示，其中点集N中的元素为交换机节点，数量为n，边集E中的元素为交换机之间的网络拓扑连边，数量为m，权重集$\omega(E)$中的元素为节点间通信时延，即数据经所选边转发所花费的时间，且整个升级过程中，网络拓扑保持不变。

接着，对整个升级过程进行数学描述。完整的升级过程如图 3-2 所示，在每一次升级时，选择一些传统 IP 交换机进行升级并在已升级交换机中选择部分位置部署控制器，网络迁移为混合 SDN。之后从剩余的 IP 交换机中选择部分交换机进行后续升级，直至所有交换机完成升级，最后一步获得纯 SDN 网络，此时升级过程结束，所有的升级步骤构成了完整的升级策略。在升级问题中，定义分T个时间步来完成升级，在第t个升级时间步，选择s^t个交换机进行升级，放置r^t个控制器，其中控制器的部署位置需要从已升级的 SDN 交换机位置中进行选择，且约定控制器一旦放置，不可再更改其位置。

中间升级步结束后得到混合 SDN 网络，其升级目标是最优化网络可编程性和延迟，即最大化所升级节点的度收益，同时最小化控制时延与同步时延之和。而最后一次升级得到的网络是纯 SDN 网络，只需要考虑纯 SDN 网络的时延性能，因此全局单目标是最小化控制时延与同步时延之和。随着升级的进行，升级目标由局部双目标退化为全局单目标。

3.2 问题建模

本节首先对 SDN 网络升级的最优化问题进行建模，得出最优化目标与约束条件。为了求解模型，将双目标模型转化为单目标模型，在转换过程中，通过引入惩

罚项，在单步升级过程中考虑最终纯 SDN 网络的利益。为了求解惩罚项，还需要对纯 SDN 网络中控制器最优位置进行最优化模型建模并求解。

3.2.1 原始的双目标模型

为了体现升级过程中，局部双目标向全局单目标的转化，本小节引入随时间变化的退化因子进行描述。同时添加约束条件，限制升级步之间的时序约束。

SDN 网络升级过程中需要决定在每一次升级时选择哪些交换机，在哪些位置部署控制器，以及控制器与交换机的匹配关系，因此引入决策变量x_i^t、y_i^t、z_{ij}^t分别表示在第t次升级时，交换机位置、控制器位置以及匹配关系，这些变量均为 0-1 变量，i和j为交换机或控制器的序号。

$$x_i^t \in \{0,1\} : t \in [1,2,3 \dots T], i \in N \quad (3\text{-}1)$$

$$y_i^t \in \{0,1\} : t \in [1,2,3 \dots T], i \in N \quad (3\text{-}2)$$

$$z_{ij}^t \in \{0,1\} : t \in [1,2,3 \dots T], i,j \in N \quad (3\text{-}3)$$

SDN 网络升级过程需要进行交换机的选择与控制器的部署。选择交换机升级为 SDN 交换机需要以最大化交换机升级带来的度收益为目标，部署控制器需要以最小化控制器部署引入的控制时延与同步时延之和为目标，据此得出升级过程中的两个最优化目标。

最大化交换机升级带来的度收益如公式（3-4）所示，表示最大化 SDN 交换机的节点度之和。其中d_i为节点i升级为 SDN 交换机带来的度收益，若第t次升级时，交换机i升级为 SDN 交换机，则x_i^t为 1，否则为 0。这一目标只在中间升级过程中存在，在最后一次升级时，所有的交换机都将被升级为 SDN 交换机，不需要考虑度收益，所以这个目标是随着升级时间步逐渐退化的，通过引入退化因子β^t，来表示这一优化目标随升级时间步的变化。第一次升级与中间升级步骤中，退化因子为 1，在最后一次升级时，退化因子退化为 0。

$$obj_1 : max \sum_{i \in N} \beta^t * d_i * x_i^t \quad (3\text{-}4)$$

最小化控制器部署引入的控制时延和同步时延如公式（3-5）所示，表示在给定时间步t，有匹配关系的控制器与交换机之间的时延和以及控制器之间的时延和最小，前一部分为控制时延，后一部分为同步时延。若第t次升级时，交换机i处放置了 SDN 控制器，则y_i^t为 1；若第t次升级时，控制器i控制 SDN 交换机j，则z_{ij}^t为 1，w_{ij}为节点i和j之间的通信时延。这一目标在整个升级过程中都存在。

$$obj_2 : min \sum_{i,j \in N} w_{ij} * z_{ij}^t + \sum_{i,j \in N} w_{ij} * y_i^t * y_j^t \quad (3\text{-}5)$$

obj_1与obj_2中的变量求解存在约束关系，obj_2中的y_i^t需要在obj_1中的x_i^t求解完成后，从其中已升级的 SDN 交换机中挑选，也就是说只有在$x_i^t = 1$时，y_i^t才可能为 1。因此这两个最优化目标是一个整体，不能分别求解，需要在同一时间步下，对应求解。同时，由于升级过程是一个整体，前一升级步中升级的交换机和放置的控制器在下一升级步中仍然存在，升级过程有时序关系，而后面的升级结果也会迫使前面的升级过程中尽可能选择有利于最终纯 SDN 网络时延较小的控制器进行放置，在时序上两个方向的互相影响致使该优化问题需要作为整体进行求解，不能依次固定时间步，求给定时间步下的最优解。

得到优化目标后，根据模型的实际背景为最优化模型添加约束条件。

(1) 升级交换机个数限制：第t次升级s^t个交换机，最终所有交换机都升级为 SDN 交换机。

$$\sum_{i\in N} x_i^t = s^t, \sum_t s^t = n, \forall t \in [1,T] \tag{3-6}$$

(2) 放置控制器个数限制：第t次升级放置r^t个控制器，最终的控制器总数为m。

$$\sum_{i\in N} y_i^t = r^t, \sum_t r^t = m, \forall t \in [1,T] \tag{3-7}$$

(3) 控制器能力约束：只有控制器可以控制其他交换机，且每个控制器所管辖的交换机要少于其管理能力。

$$\sum_{j\in N} z_{ij}^t \le c * y_i^t, \forall i, \forall t \in [1,T] \tag{3-8}$$

(4) 交换机被控制约束：每个交换机若被控制器所管，则该交换机必须先为 SDN 交换机，且只被一个控制器控制。

$$\sum_{i\in N} z_{ij}^t = x_j^t, \forall j, \forall t \in [1,T] \tag{3-9}$$

(5) 每个控制器部署位置的交换机必须是 SDN 交换机。

$$y_i^t \le x_i^t, \forall i, \forall t \in [1,T] \tag{3-10}$$

(6) $t-1$时刻升级的交换机，在t时刻仍处于升级状态，即交换机升级状态不可回退，每个交换机只能被升级一次。

$$x_i^{t-1} \le x_i^t, \forall t \in [2,T] \tag{3-11}$$

(7) $t-1$时刻部署的控制器，在t时刻仍为控制器，即控制器部署后不可拆除。

$$y_i^{t-1} \le y_i^t, \forall t \in [2,T] \tag{3-12}$$

此外，所有的决策变量都是 0-1 变量。

$$x_i^t, y_i^t, z_{ij}^t \in \{0,1\}, \forall i,j, \forall t \in [1,T] \tag{3-13}$$

综上，得出 SDN 网络动态升级过程的最优化模型如下。

$$obj: max \sum_{i \in N} \beta^t * d_i * x_i^t$$

$$min \sum_{i,j \in N} w_{ij} * z_{ij}^t + \sum_{i,j \in N} w_{ij} * y_i^t * y_j^t$$

$$s.t. \sum_{i \in N} x_i^t = s^t, \sum_t s^t = n, \forall t \in [1,T]$$

$$\sum_{i \in N} y_i^t = r^t, \sum_t r^t = m, \forall t \in [1,T]$$

$$\sum_{j \in N} z_{ij}^t \leq c * y_i^t, \forall i, \forall t \in [1,T]$$

$$\sum_{i \in N} z_{ij}^t = x_j^t, \forall j, \forall t \in [1,T]$$

$$y_i^t \leq x_i^t, \forall i, \forall t \in [1,T]$$

$$x_i^{t-1} \leq x_i^t, \forall t \in [2,T]$$

$$y_i^{t-1} \leq y_i^t, \forall t \in [2,T]$$

$$x_i^t, y_i^t, z_{ij}^t \in \{0,1\}, \forall i, j, \forall t \in [1,T] \qquad (P_1)$$

最优化模型中用到的参数如表 3-2 所示，其中q_i与η将在之后模型转换中用到。

表 3-2 模型参数表

Table 3-2 Notations of the model

参数	含义
n	交换机节点个数
m	部署的控制器个数
T	总升级时间步数
c	控制器的管理能力
N	网络拓扑节点集合
i	交换机编号，$i \in N$
j	控制器编号，$j \in N$
t	升级时间步序号，$t \in [1,T]$
d_i	节点i的度
q_i	非全局最优因子，表示节点i是否为全局最优控制器
w_{ij}	节点i和j之间最短路的时延
s^t	第t时间步升级的交换机个数
r^t	第t时间步部署的控制器个数
γ^t	第t时间步的惩罚因子
β^t	第t时间步的退化因子
η	度收益与时延损失之间的偏好因子

3.2.2 变形的单目标模型

根据上一小节的分析，最优化模型（P_1）在求解时要看作一个整体，将所有时间步下的决策变量同时进行求解，也即$\forall i,j,t$下，x_i^t, y_i^t, z_{ij}^t需要同时进行求解，这样会导致决策变量过多。同时，由于两个最优化目标之间有依存关系，需要同时求解两个目标，若直接固定时间步求解，会导致单步升级时忽略全局利益，（P_1）模型无法直接求解，需要将（P_1）模型进行简化，转化为不存在时间依赖的单目标优化函数，即可在固定时间步下求解整数线性规划问题。

在转化过程中，需要同时考虑升级过程的双向影响。其中正序影响可以通过引入时间上标依序求解，并对相邻升级步进行约束来体现；逆序影响则需要引入额外的优化目标，在单步升级过程中，同时考虑最终纯 SDN 网络的利益。考虑到升级后纯 SDN 网络的运行时间相较于整个升级过程的时间更长，最终纯 SDN 网络带来的长期收益比升级过程中的短期收益更重要。因此，在单步升级过程中，希望单步升级部署的控制器尽可能位于全局最优控制器部署点，兼顾单步收益与全局收益。本文在分步升级过程中引入惩罚项，若控制器在分步升级时被放置在非最优位置，就要受到相应的惩罚。同时引入惩罚因子作为惩罚项与收益项的比重，平衡当前时间步下混合 SDN 网络的性能以及最终纯 SDN 网络的性能，也即平衡当前时间步收益与未来收益。引入惩罚项后，单步升级时同时考虑了局部双目标与全局单目标，在单步收益中允许少量的控制器偏离当前的最佳位置，以兼顾升级完成后的网络性能，从而获得一个较好的控制器布局方案，维持整个 SDN 网络在未来 10 年到 20 年的正常高效运行。

首先引入偏好因子将双目标问题转化为单目标问题，并引入新的变量和约束将问题转化为整数线性规划问题。转换后的模型如式（P_1^*）所示。

$$obj:\ max\ (1-\eta)\sum_{i\in N}\beta^t * d_i * x_i^t - \eta * \left(\sum_{i,j\in N} w_{ij} * z_{ij}^t + \sum_{i,j\in N} w_{ij} * y_{ij}^t\right)$$

s.t.

$$y_{ij}^t \le y_i^t, \forall i,\mathrm{j},\forall t \in [1,T]$$

$$y_{ij}^t \le y_j^t, \forall i,\mathrm{j},\forall t \in [1,T]$$

$$y_{ij}^t \ge y_i^t + y_j^t - 1, \forall i,\mathrm{j},\forall t \in [1,T]$$

（3-6）（3-7）（3-8）（3-9）（3-10）（3-11）（3-12）（3-13） （P_1^*）

由于度收益与时延损失单位不同，不同网络中两者的重要程度也不同，因此引入偏好因子来表示两者的重要程度占比，定义η为度收益与时延损失之间的偏好因

子，也即获取升级为 SDN 交换机带来的收益和避免控制器引入的时延之间的偏好程度。若η为 0，则只考虑 SDN 网络获得的度收益，若η为 1，则仅考虑 SDN 网络中控制器引入的时延，引入偏好因子后得到的单目标与（P_1）模型中的双目标等价[16]。之后将最优化目标中的非线性项$y_i^t * y_j^t$通过引入新的变量$y_{ij}^t = y_i^t * y_j^t$转换为线性项，并添加线性约束表示y_{ij}^t与y_i^t的关系，便于直接使用商用线性规划求解器求解。由于y_i^t和y_j^t均为 0-1 变量，仅当两者都为 1 的情况下，y_{ij}^t为 1，否则为 0。

引入偏好因子仅使得双目标问题转换为单目标问题，还需进一步解决时间依赖性问题。约束条件（3-11）和（3-12）规定了正时序依赖关系，后一时间步的升级方案受限于前一时间步。但时序依赖包含两个方向，正序是时间顺序，升级过程的进行是随着时间有先后顺序的；逆序是利益影响，单步升级时还要考虑未来纯 SDN 网络的收益。逆时序依赖不能简单的通过添加约束条件实现，本文引入惩罚项，在每一时间步下，对部署在非全局最优位置的控制器进行惩罚，表示未来收益对当前时间步的约束。由于前序升级步中部署的控制器将一直延续到最终纯 SDN 网络中，直接影响纯 SDN 网络的表现性能，因此将惩罚项定义为非全局最优控制器带来的惩罚。具体来说，单步升级时部署的控制器应尽可能处于纯 SDN 网络中的控制器最优位置，因此对升级过程中部署在非全局最优位置的控制器给予一定的惩罚，迫使在中间升级步中部署的控制器尽量处于纯 SDN 网络下的最优位置。其中纯 SDN 网络中的控制器最优位置是指最小化控制时延与同步时延之和的控制器部署位置，本文也称为控制器全局最优位置。

引入惩罚项后，最优化目标如公式（3-14）所示。第一部分为所有已升级 SDN 交换机节点的度之和，也即度收益，第二部分为控制时延与同步时延之和，第三部分为惩罚项。该最优化目标表示升级过程需要同时最大化升级的度收益，最小化控制器部署引入的时延和，以及最小化惩罚。此时最优化模型可以依次给定时间步，依序求解单步升级策略，得到完整的最优动态升级策略。

$$max\ \sum_{i\in N}\beta^t * d_i * x_i^t - \eta * (\sum_{i,j\in N} w_{ij} * z_{ij}^t + \sum_{i,j\in N} w_{ij} * y_{ij}^t) - \sum_{i\in N}\gamma^t * q_i * y_i^t \quad (3\text{-}14)$$

其中，惩罚项中非全局最优因子q_i表示位置i是否为纯 SDN 网络最优控制器位置，若是，则为 0，否则为 1。如果位置i不是最优位置，且在该位置部署了控制器，则$q_i = 1$，$y_i^t = 1$，对该放置做出惩罚，q_i的计算将在下一小节中说明。γ^t为局部利益与全局利益的协调因子，也即惩罚系数。由于在靠前时间步中部署的控制器将存在更久的时间，对整个升级过程的影响更大，因此希望γ^t随着时间逐渐减小，且与升级所处的时间步相关，所以定义γ^t为t时刻布局控制器存在的时间与整个网络持续的总时间之比，如公式（3-15）所示。

$$\gamma = \frac{t_{max}+t_{block*（T-t）}}{t_{max}+t_{block*（T-1）}} \tag{3-15}$$

t_{max}为升级完成后纯 SDN 网络的最大寿命，t_{block}为两次升级过程中间隔的时间。该系数的含义为，第一次部署的控制器存在的时间比第二次部署的控制器存在时间长，相较于第二次更希望第一次部署的控制器为全局最优控制器，所以惩罚系数随着升级次数的增加而减小。

以上完成了最优化模型的转换，转换后的整数线性规划模型如（P_1^{**}）所示。

$$obj:\ \max(1-\eta)\sum_{i\in N}\beta^t * d_i * x_i^t - \eta * \left(\sum_{i,j\in N} w_{ij} * z_{ij}^t + \sum_{i,j\in N} w_{ij} * y_{ij}^t\right) - \sum_{i\in N}\gamma^t * q_i * y_i^t$$

$$s.t.\ \sum_{i\in N} x_i^t = s^t, \sum_t s^t = n, \forall t \in [1,T]$$

$$\sum_{i\in N} y_i^t = r^t, \sum_t r^t = m, \forall t \in [1,T]$$

$$\sum_{j\in N} z_{ij}^t \le c * y_i^t, \forall i, \forall t \in [1,T]$$

$$\sum_{i\in N} z_{ij}^t = x_j^t, \forall j, \forall t \in [1,T]$$

$$y_i^t \le x_i^t, \forall i, \forall t \in [1,T]$$

$$x_i^{t-1} \le x_i^t, \forall t \in [2,T]$$

$$y_i^{t-1} \le y_i^t, \forall t \in [2,T]$$

$$y_{ij}^t \le y_i^t, \forall i, \mathrm{j}, \forall t \in [1,T]$$

$$y_{ij}^t \le y_j^t \forall i, \mathrm{j}, \forall t \in [1,T]$$

$$y_{ij}^t \ge y_i^t + y_j^t - 1 \forall i, \mathrm{j}, \forall t \in [1,T]$$

$$x_i^t, y_i^t, z_{ij}^t \in \{0,1\}, \forall i, j, \forall t \in [1,T] \tag{P_1^{**}}$$

3.2.3 惩罚项求解模型

为了求得惩罚项中的非全局最优因子q_i，需要先计算出纯 SDN 网络下的控制器最优位置，之后转化为q_i。本小节首先搭建求解纯 SDN 网络中最优控制器部署位置的模型，之后给出由控制器全局最优位置得出非全局最优因子q_i的方法。

搭建最优化模型时，与上一小节的变量定义一致，定义y_i为是否在位置i部署控制器，定义z_{ij}为控制器与交换机之间的所属关系，表示交换机i是否被控制器j控

制。

构建最优化模型如下，其目标为最小化控制时延与同步时延之和。

$$obj:\ min\sum_{i,j\in N} w_{ij} * z_{ij} + \sum_{i,j\in N} w_{ij} * y_i * y_j \quad (3\text{-}16)$$

该模型的约束有以下三类。

(1) 控制器数量约束：在整个拓扑中，所放置的控制器数量为固定值m。

$$\sum_{i\in N} y_i = m, \forall i \quad (3\text{-}17)$$

(2) 控制器能力约束：每个控制器都有其能力限制，每个控制器所控制的交换机不能超过其能力限制。同时在控制关系中，只有控制器才可以控制其他交换机。

$$\sum_{i\in N} z_{ij} < c * y_j, \forall j \quad (3\text{-}18)$$

(3) 交换机被控约束：每个交换机只能被一个控制器控制。

$$\sum_{j\in N} z_{ij} = 1, \forall i \quad (3\text{-}19)$$

最后，所有的决策变量都是 0-1 变量。

$$y_i, z_{ij} \in \{0,1\}, \forall i, j \quad (3\text{-}20)$$

此时得到的最优化模型中含有非线性项$y_i * y_j$，非线性项的引入会导致最优化问题求解难度增加，因此考虑引入新的变量$y_{ij} = y_i * y_j$替换原有的非线性项，将控制器全局最优位置求解问题转换为整数线性规划问题，从而利用现有的商用求解器直接求解。在引入新变量后，还需要增加新的约束来满足变量之间的关系。由于y_i和y_j都是 0-1 变量，当且仅当两个变量都为 1 时，$y_{ij} = 1$，否则$y_{ij} = 0$。因此引入以下三个约束：

$$y_{ij} \le y_i \quad (3\text{-}21)$$

$$y_{ij} \le y_j \quad (3\text{-}22)$$

$$y_{ij} \ge y_i + y_j - 1 \quad (3\text{-}23)$$

最终形成最优化模型：

$$obj: \cdot min \sum_{i,j\in N} w_{ij} * z_{ij} + \sum_{i,j\in N} w_{ij} * y_i * y_j$$

$$s.t. \sum_{i\in N} y_i = m, \forall i$$

$$\sum_{i\in N} z_{ij} < c * y_j, \forall j$$

$$\sum_{j\in N} z_{ij} = 1, \forall i$$

$$y_{ij} \le y_i$$

$$y_{ij} \le y_j$$
$$y_{ij} \ge y_i + y_j - 1$$
$$y_i, z_{ij} \in \{0,1\}, \forall i,j \qquad (P_2)$$

通过解该线性规划，可以得到决策变量的解z_{ij}，以及全局最优控制器y_i的部署位置，非全局最优因子q_i的计算公式如下。

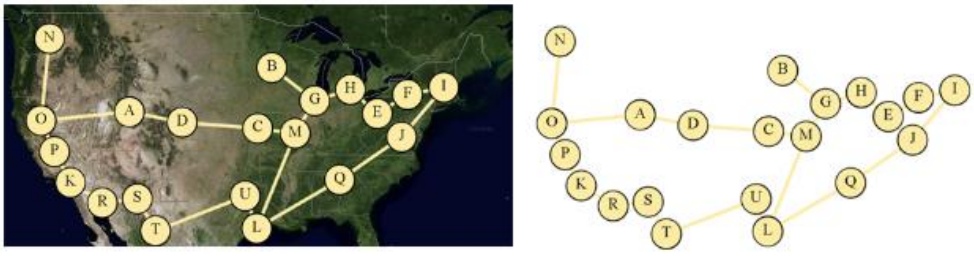
$$q_i = 1 - y_i \qquad (3\text{-}24)$$

3.3 模型验证

上一节对 SDN 网络升级问题进行了最优化模型建模与转化，构建可以直接求解的最优化模型。本节将在美国真实骨干网络拓扑 ATMnet 中进行模型验证，并与其他基线算法进行比较，分析本文模型的有效性。

3.3.1 数据集与算法说明

本节选用的数据集为美国真实骨干网络 ATMnet，拓扑数据来源于 Topology Zoo[59]，该网络中有 21 个点和 22 条边，是小规模网络，拓扑图如图 3-6 所示，a)图为拓扑图在真实地图上的展示，b)图为抽离出的拓扑图。约定升级分 3 次完成，每次升级 7 个交换机，放置 1 个控制器，每个控制器最多可以管理 10 个交换机，且规定两次升级之间时间跨度为 1 年，最终升级所得的纯 SDN 网络可以正常运行 10 年。本文采用源点（source）和终点（destination）之间的实际地理距离除以光速对时延进行估计。



a) 带地图的网络拓扑　　b) 网络拓扑

a) Topology with map　　b) Topology without map

图 3-6 ATMnet 网络拓扑图

Figure 3-6 Topology of ATMnet

选用三种不同的升级方案对网络进行升级，其中前两种方案为传统方案，作为本文的基线对比方案，第三种方案为本文提出的升级方案，以此来分析本文所提出的升级策略的有效性。

朴素升级方案（Naïve deployment）：朴素升级方案来源于文献[16]，将优化目标分别考虑，首先以升级所获得的度收益最大作为优化目标进行交换机升级，再以控制时延与同步时延之和最短进行控制器的部署。

联合升级方案（Joint deployment）：联合升级方案来源于文献[16]，这一升级方案同时考虑交换机升级和控制器部署，但不考虑整个升级过程中的时序依赖性，具体实现方式是在每一个升级步中，同时最大化度收益和最小化控制时延与同步时延，也即模型（P_1^*）中去掉时间维度，逐次迭代。

全局升级方案（Global deployment）：全局升级方案是本文提出的升级方案，这一方案综合考虑全局利益和局部利益，在固定时间步下，同时最大化度收益、最小化时延、最小化惩罚，也即求解模型（P_1^{**}）。

朴素升级方案的基本思想是首先进行交换机选择，之后在已升级的交换机中进行控制器部署，是将现有的控制器部署策略直接应用于 SDN 动态升级问题中，其优点在于在每次升级时可以充分考虑交换机升级带来的度收益，但朴素升级方案并没有均衡升级带来的收益与部署控制器引入的时延。联合升级方案的基本思想是在朴素升级方案的基础上，同时考虑交换机升级与控制器部署，均衡了两种收益，但其缺点在于没有考虑升级步之间的双向时间关联性，仅将单步升级策略直接应用于后续升级，忽视了最终纯 SDN 网络的全局收益对单步升级的影响。

3.3.2 求解结果与对比

求解 SDN 动态升级策略需要先求出全局最优控制器部署位置，进而得出非全局最优因子，代入惩罚项。因此首先对网络拓扑全局最优控制器位置进行求解，结果如图 3-7 所示，其中有相同颜色的节点被同一个控制器控制，全局最优控制器布局点为节点 G（芝加哥）、节点 M（圣路易斯）以及节点 P（加利福尼亚），纯 SDN 网络下的控制时延为 65 ms，同步时延为 52 ms，总时延为 117 ms。

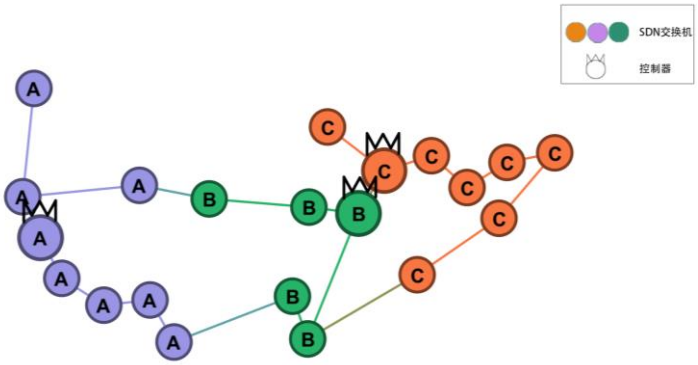


图 3-7 全局最优控制器布局图

Figure 3-7 Global optimal controller layout

通过依次求解模型（P_1）中的两个目标函数得到朴素升级方案；按照时序依次给定时间步下，求解模型（P_1^*）的最优解得到联合升级方案；将求解得到的控制器最优布局方案作为惩罚项中非全局最优因子q_i输入到模型（P_1^{**}）中求解，得到全局升级方案。

以上三个升级方案最优化模型均可转换为整数线性规划，利用 cplex 库函数直接进行求解，求解得到的升级方案如图 3-8 所示，升级结果如表 3-3 所示。其中（a）（b）（c）依次为朴素升级方案求得的逐步升级结果，最终的控制器部署位置为节点 C（丹佛）、节点 F（费城）以及节点 O（奥克兰），最终纯 SDN 网络的控制时延为 58 ms，同步时延为 68 ms，总时延为 126 ms；（d）（e）（f）依次为联合升级方案求得的逐步升级结果，最终的控制器部署位置为 E（匹兹堡）、K（休斯顿）、L（洛杉矶），最终纯 SDN 网络下的控制时延为 58 ms，同步时延为 70 ms，总时延为 128 ms；（g）（h）（i）依次为全局升级方案求得的逐步升级结果，也即本文提出的方案，最终的控制器部署位置为 G（芝加哥）、节点 K（休斯顿）以及节点 L（洛杉矶），纯 SDN 网络下的控制时延为 57 ms，同步时延为 62 ms，总时延为 119 ms。

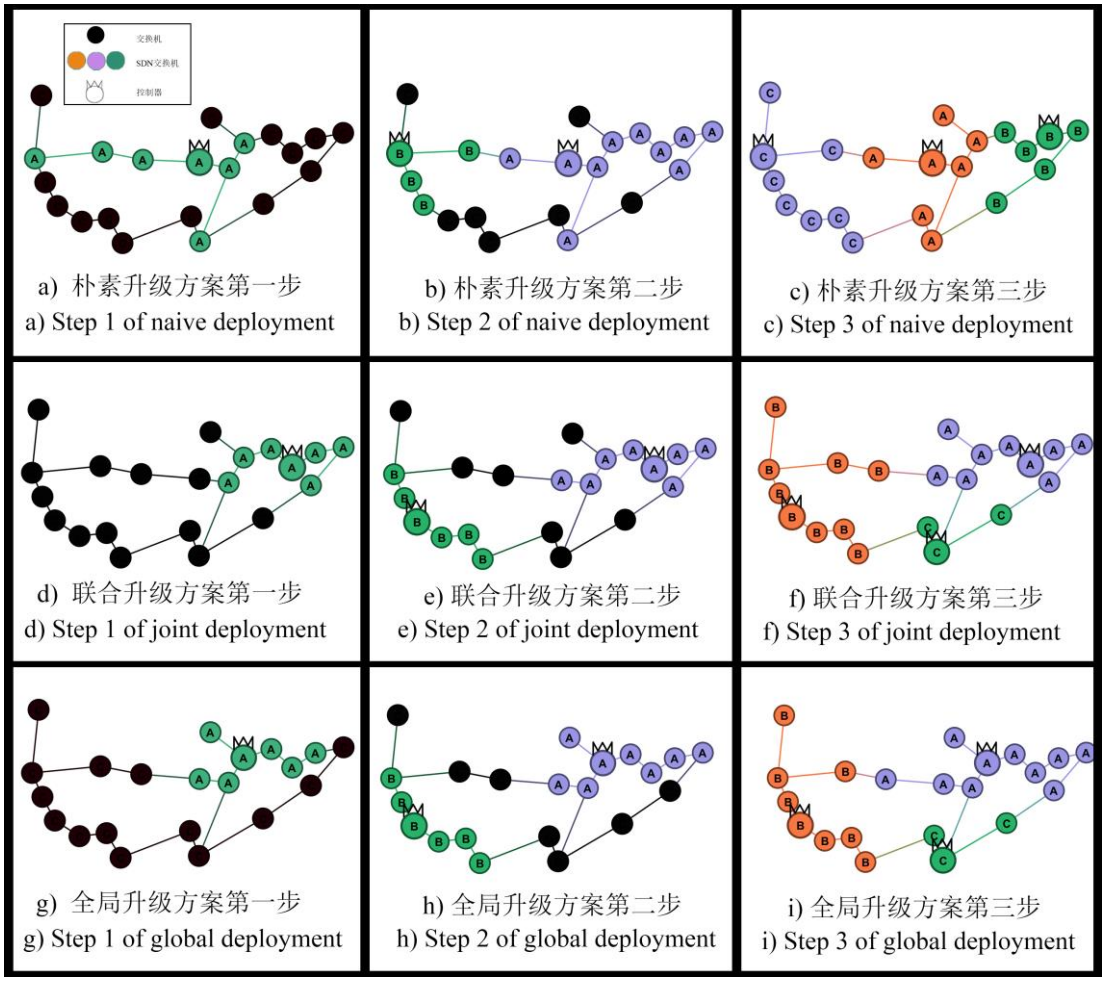


图 3-8 三种升级方法的升级结果图

Figure 3-8 Migration process of the three upgrade schemes

表 3-3 三种升级方案结果对比

Table 3-3 Comparison of results for three upgrade schemes

	部署结果	控制时延(ms)	同步时延(ms)	总时延(ms)
朴素升级方案	C、F、O	58	68	126
联合升级方案	E、K、L	58	70	128
全局升级方案	G、K、L	57	62	119

由升级结果图可以看出，朴素升级方案中，所升级的点会在散布在拓扑中，而联合升级方案与全局升级方案，交换机会成团的升级，形成簇状。在综合考虑局部利益和全局收益时，逐步升级所得的控制器布局方案最接近全局最优控制器部署方案。

3.3.3 指标定义与分析

升级结果图直观显示了三种升级方案的结果差异，但未给出具体的数值比较，因此本小节将在定义性能指标的基础上，定量分析升级方案的性能优劣。性能包括两个方面，一是单步收益，即每一升级步得到的度收益和时延损失；二是整体收益，即整个升级过程的总度收益和总时延损失。

定义 3.1：度收益

度收益是所升级的交换机节点的度之和，代表了所升级的 SDN 交换机在路由选择时，可选路径数的多少。一般来说，度收益越大，可选择的非最短路径数越多，在网络拥塞的情况下对流量的分配效果越好，计算公式如式（3-25）所示。

$$degree\_gain = \sum_{i \in N} d_i * x_i^t \quad (3\text{-}25)$$

定义 3.2：时延损失

时延损失是所部署控制器带来的控制时延与同步时延之和。时延越小，信令传递越快，信息传递效率越高，全网对于突发事件的反应越迅速。时延损失包括控制时延与同步时延，控制时延是指控制器控制其范围内的交换机的时延总和，代表了全网拓扑控制信息传递的时延损失，同步时延是指控制器之间相互传递信息的时延总和，代表了全网同步拓扑信息的时延损失，计算公式如式（3-26）所示。

$$time\_loss = \sum_{i,j \in N} w_{ij} * z_{ij}^t + \sum_{i,j \in N} w_{ij} * y_i^t * y_i^t \quad (3\text{-}26)$$

定义 3.3：总度收益

总度收益为每一升级步的度收益的时间积分，即每次升级的度收益乘以该次升级持续的时间并求和。由于升级过程中产生的混合 SDN 网络以及最终的纯 SDN

网络都会在实际升级建造中存在一段时间，因此每一次升级带来的利益对全局的贡献可以利用该次升级所存在的时间进行衡量，将每次所得利益进行时间积分可以有效地反映出这一升级方案的全局收益大小，计算公式如式（3-27）所示。

$$total\_degree\_gain = \int_1^T gain = \sum_{t=1}^{T}(time * \sum_{i \in N} d_i * x_i^t) \quad (3\text{-}27)$$

定义 3.4：总时延损失

总时延损失为每一升级步的时延损失的时间积分，即该次升级的时延损失乘以该次升级持续的时间并求和，计算公式如式（3-28）所示。

$$total\_time\_loss = \int_1^T loss = \sum_{t=1}^{T}(time * (\sum_{i,j \in N} w_{ij} * z_{ij}^t + \sum_{i,j \in N} w_{ij} * y_i^t * y_i^t)) \quad (3\text{-}28)$$

度收益与时延损失这两个指标分别反映了升级带来的收益与损失，可以较好的反映出升级方案对于 SDN 网络利弊的均衡情况，度收益对比与时延损失对比图分别如图 3-9 和图 3-10 所示，三种升级方案计算得到单步升级收益与全局升级收益结果如表 3-4 所示。

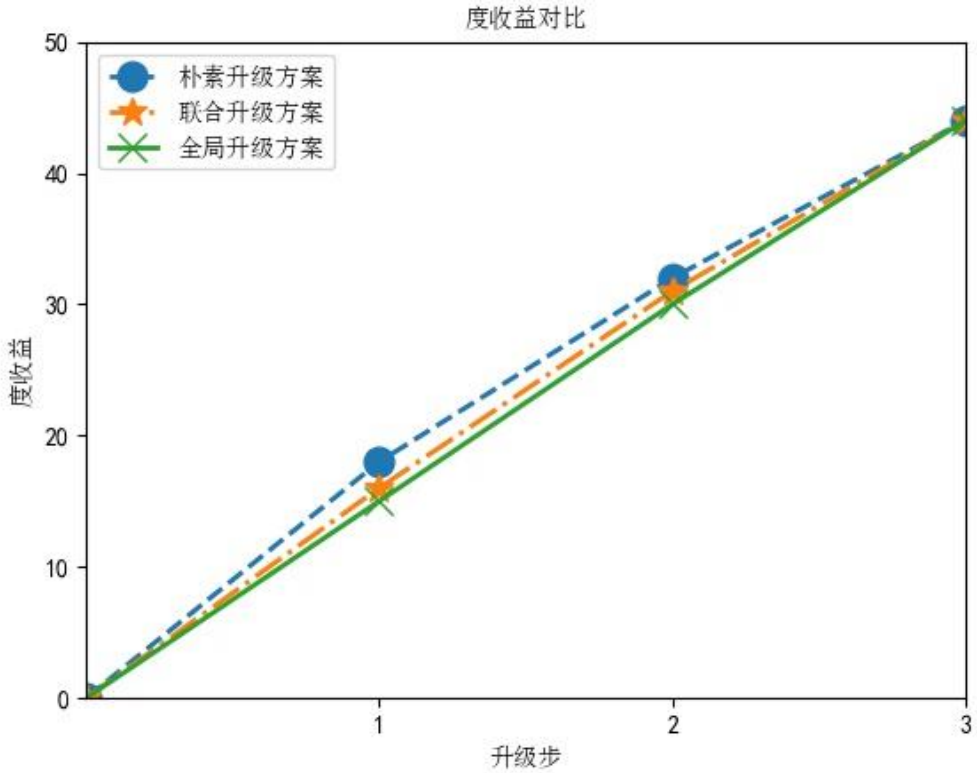


图 3-9 三种升级方案度收益对比

Figure 3-9 Degree gain comparison of the three upgrade schemes

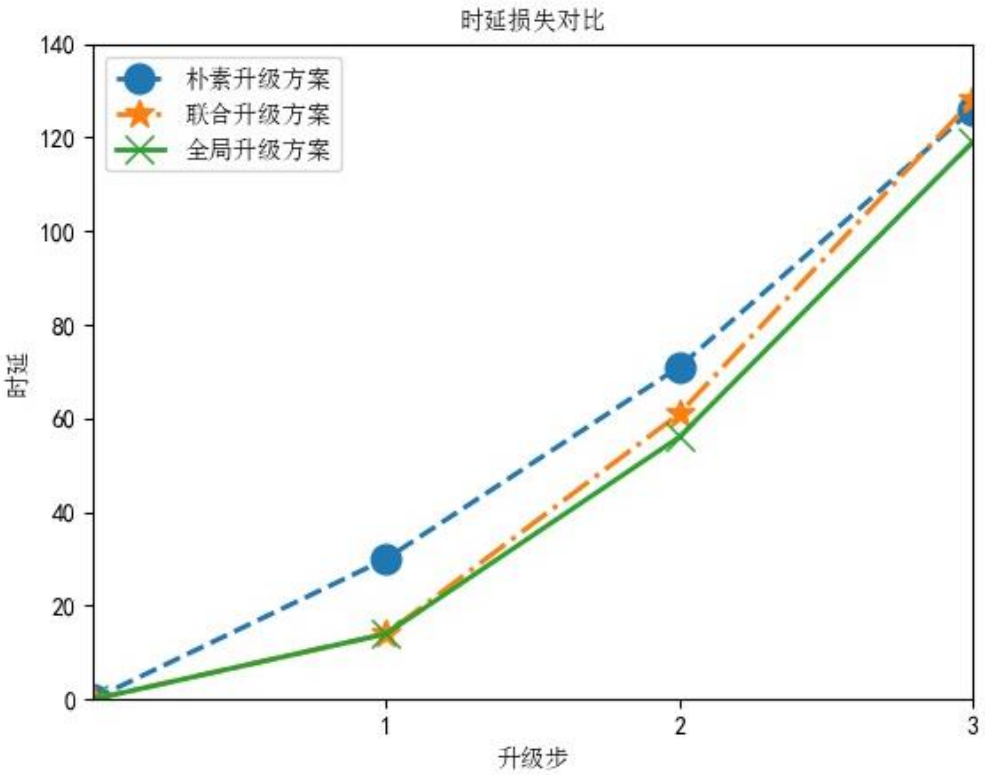


图 3-10 三种升级方案时延损失对比图

Figure 3-10 Delay loss comparison of the three upgrade schemes

表 3-4 三种升级方案结果对比

Table 3-4 Comparison of results for three upgrade schemes

升级步	度收益			时延损失(ms)		
	朴素①	联合②	全局③	朴素①	联合②	全局③
1	18	16	15	30	14	14
2	32	31	30	71	61	56
3	44	44	44	126	128	119
总计	402	399	397	1109	1099	1022

可以看出三种方案的度收益相差并不大，但是本文提出的全局升级方案的时延损失明显低于其他两种基线方案，在任一升级步的时延损失都是最小。全局升级方案牺牲了较小的中间升级度收益，但收获了每一时间步的最短时延，尤其是在升级过程中最重要的最终纯 SDN 网络布局中，全局升级方案获得的控制器布局时延最小。

由于度收益和时延损失的单位不一致，无法对全局升级方案相较于其他两种方案导致的度收益牺牲与带来的时延损失减少之间进行数值直接比较，因此引入收益牺牲和时延减少指标。

定义 3.5：收益牺牲

收益牺牲表示方案b相对于方案a来说，获得的度收益减少的百分比，是负向收益，计算公式如（3-29）所示。

$$degree\_sacrifice = \frac{degree\_gain_b - degree\_gain_a}{degree\_gain_a} \quad (3\text{-}29)$$

定义 3.6：时延减少

时延减少表示方案b相对于方案a的时延减少的百分比，是正向收益。计算公式如公式（3-30）所示，

$$delay\_reduction = \frac{time\_loss_b - time\_loss_a}{time\_loss_a} \quad (3\text{-}30)$$

根据公式（3-29）和（3-30），并根据表 3-2 计算可得，在第一次升级过程中，全局升级方案③相较于朴素升级方案①在牺牲了 16%的度收益下，获取了 53%的时延减少，全局升级方案③相较于联合升级方案②牺牲了 6%的度收益。在第二次升级过程中，全局升级方案③相较于朴素升级方案①在牺牲了 6%的度收益下，获取了 21%的时延减少，全局升级方案③相较于联合升级方案②牺牲了 3%的度收益下，获取了 8%的时延减少。在最终布局下，由于三个升级方案都将所有节点进行了升级，所以三种方案所得到的度收益相同，而在时延方面，全局升级方案③考虑了全局最优布局，其时延最小，对比结果如图 3-11 所示。

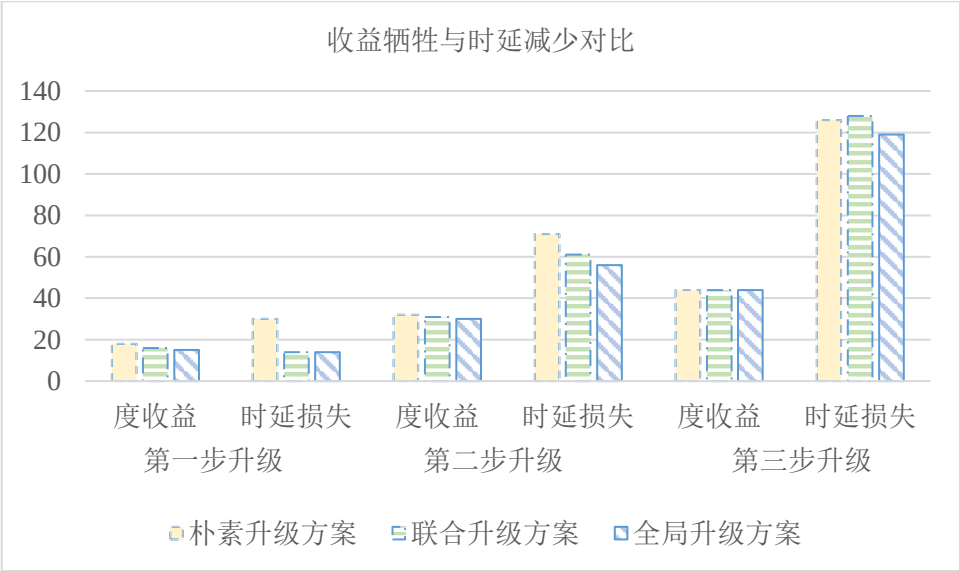


图 3-11 三种升级方案收益牺牲与时延减少对比图

Figure 3-11 Gain sacrifice and delay reduction comparison of the three upgrade schemes

因此，全局升级方案③相较于朴素升级方案和联合升级方案，可以在牺牲较小的度收益情况下获取较大的时延减少，且最终纯 SDN 网络的控制器布局时延是最小的，而最终时延影响着该网络在未来十几年的性能，更为重要。

在时间积分后，三种方案的总度收益相差不大，对比图如图 3-12 所示。全局升级方案③相较于朴素升级方案①，牺牲了 1%的总度收益，换取了 8%的总时延减少；全局升级方案③相较于联合升级方案②，牺牲了 0.5%的总度收益，换取了 7%的总时延减少。由此可见，本文所提出的方案③可以在牺牲较小总度收益下，换取更大的总时延减少。

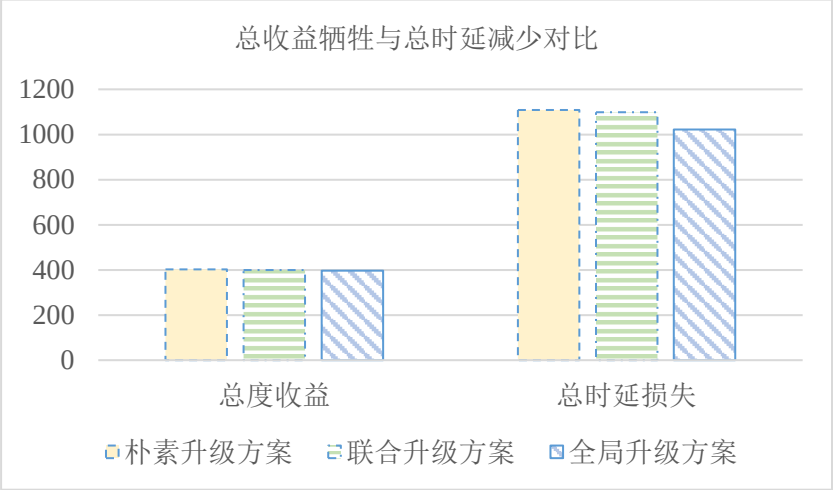


图 3-12 三种升级方案总收益牺牲与总时延减少对比图

Figure 3-12 Total gain sacrifice and total delay reduction comparison of the three upgrade schemes

本文提出的全局升级方案，获得的时延减少优于朴素升级方案和联合升级方案，且最终纯 SDN 网络下，全局升级方案的效果要比前两种方案更好，而且最终纯 SDN 网络的效果更重要。这是 SDN 网络运营时间远短于 SDN 网络建设时间，且 SDN 网络的升级部署最终目标是在未来运营过程中达到良好性能。因此本文提出的全局升级方案相较于现有方案，可以更好的在单步利益与全局利益之间达到权衡，牺牲小部分单步利益以换取长远来看更有利的全局收益。

3.4 本章小结

本章对 SDN 网络动态升级策略依次进行了问题分析、问题建模与模型验证。在问题分析中，通过示例网络与理论分析，发现 SDN 动态升级问题需要同时求解升级时间步确定、交换机选择以及控制器部署的问题。本章对单步利益和全局利益进行分析建模，首先将问题映射为数学问题，定义问题的目标与约束，之后引入退化因子，分别对交换机升级与控制器部署设定最优化目标，并添加相应约束条件，完成了 SDN 网络升级的最优化模型建模。由于该模型的正序时间依赖性与逆序利益依赖性无法直接求解，本章通过引入偏好因子和惩罚函数将模型转换为可以直接求解的整数线性规划问题。引入的惩罚项可以平衡两种利益，更全面的考虑了 SDN 网络动态升级的利益最大化。最后，本文在真实网络拓扑中进行升级策略求

解，得到由纯 IP 网络到纯 SDN 网络的完整升级过程，并与现有算法进行对比分析，验证了本文提出的全局升级方案可以更好的考虑整个 SDN 网络升级过程的利益，通过牺牲小部分单步收益换取长远的全局收益。

4 基于改进遗传算法的启发式算法

第 3 章对全局视角下的 SDN 最优动态升级问题进行建模，并在小规模网络下直接求解升级策略，但该问题属于 NP-hard 问题，在大规模场景下无法直接求解。本章提出一种基于改进遗传算法的启发式算法，针对问题约束条件多的特点，对遗传算法中的编码以及交叉变异进行改进，以提高产生子代的合格率。首先提出 SM-GA 算法求解全局近似最优控制器布局，之后将其解转化为惩罚项；然后提出 CCM-GA 算法求解近似最优升级策略，最后在真实网络拓扑中验证启发式算法的有效性。

4.1 算法描述

SDN 最优动态升级问题为 NP-hard 问题[16]，当网络拓扑点数增多时，模型变量和约束条件增多，商用求解器无法在有效时间内求得精确解，本章提出一种启发式算法在大规模网络下，对 3.2 节中提出的双目标动态优化模型（P_1）进行近似求解。由于P_1模型被分解为两个单目标模型：纯 SDN 网络下的控制器最优布局模型（P_2）和变形的单目标升级策略模型（P_1^{**}），所以，该启发式算法包括两个子算法：基于 SDN 匹配关系的遗传算法（SM-GA）和基于互补交叉变异的遗传算法（CCM-GA），分别用于对P_2和（P_1^{**}）进行求解。该启发式算法的整体框图如图 4-1 所示。

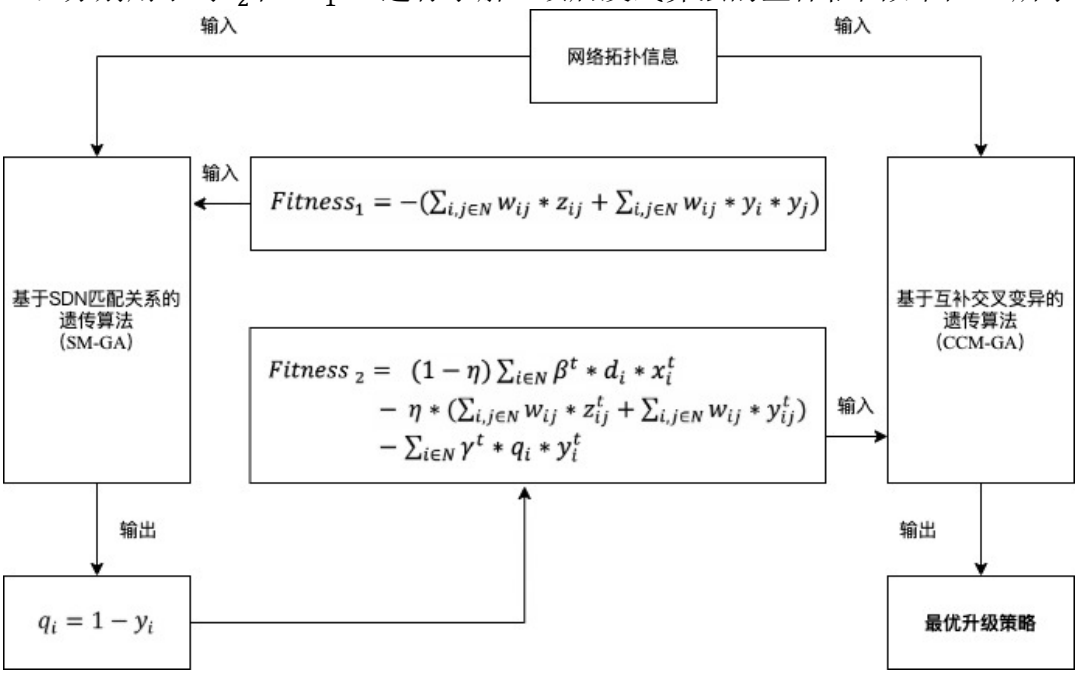


图 4-1 SDN 最优动态升级问题的启发式算法求解框图

Figure 4-1 Block diagram of heuristic algorithm for SDN migration problem

图中，左框图为基于 SDN 匹配关系的遗传算法（SM-GA），用于求解纯 SDN 网络下的控制器最优布局，以确定惩罚项中非全局最优因子q_i。该算法的输入是网络拓扑信息和适应度函数$Fitness_1$，适应度函数为控制器引入的控制时延与同步时延和。该算法的输出是最优化模型（P_2）的解y_i，代入公式即可求出非全局最优因子q_i。右框图为基于互补交叉变异的遗传算法（CCM-GA），用于求解动态升级策略。该算法的输入为网络拓扑以及适应度函数$Fitness_2$，适应度函数为模型（P_1^{**}）的最优化目标，其中非全局最优因子q_i为 SM-GA 算法求解结果。该算法的输出是模型（P_1^{**}）的解，也即动态升级策略。SM-GA 将在 4.2 节进行介绍，CCM-GA 算法将在 4.3 节进行介绍。

4.2 基于 SDN 匹配关系的遗传算法

4.2.1 SM-GA 算法框图

为了求解惩罚项q_i，本节提出基于 SDN 匹配关系的遗传算法（genetic algorithm based on SDN matching relation, SM-GA），求解纯 SDN 网络下的控制器全局最优布局方案，进而得出惩罚项中的非全局最优因子q_i，便于后续 SDN 升级策略的求解。纯 SDN 网络控制器最优位置的确定问题定义为，在一个有s个点的网络拓扑中部署r个控制器，使得控制时延与同步时延之和最小，也即 3.2.4 节中的（P_2）模型。基于 SDN 匹配关系的遗传算法框图如图 4-2 所示，左图为遗传算法的流程，右图为 SM-GA 算法对流程各部分的具体定义。

在遗传算法中，首先需要对问题进行可行解定义，也即通过编码确定遗传算法中个体的基因结构。SM-GA 算法定义个体的解格式为$n*n$维矩阵，并给出个体解需满足的条件。

之后根据约束条件产生初始父代，作为遗传算法的初始种群。由于 SM-GA 算法的个体基因约束较多，随机产生个体会导致父代的合格率很低，SM-GA 算法设计了特殊的产生父代规则，以提升产生父代的合格率。

基因片段是交叉和变异的基础，SM-GA 算法中基因为$n*n$维矩阵，可选的基因片段包括矩阵行、矩阵列以及矩阵块，也即基因行、基因列以及基因块。其中基因行包含的信息是该行所代表的交换机被哪个控制器控制，基因列包含的信息是该列所表示的位置是否放置控制器，若放置了控制器，还可以得出该控制器控制哪些交换机，基因块则在该问题中没有实际的意义。

交叉和变异是产生新的基因类型的方法，交叉与变异是遗传算法产生新的基因类型，使种群趋于最优的手段。其中交叉是指将两个个体的对应基因片段互换，产

生两个新个体的过程；变异是指对单个个体的某一基因片段进行突变，产生一个新个体的过程，SM-GA 算法针对个体基因约束条件多的问题，设计交叉变异规则。

在新种群内需要根据适应度函数进行种群择优，适应度函数可选择为所求问题的优化目标，利用适应度函数可以计算每一个个体的适应值，之后选取适应值最好的一批个体作为下一次遗传的父代，进行后续遗传。SM-GA 算法选择模型（P_2）的优化目标，即控制时延与同步时延的和，如公式（4-1）所示。

$$Fitness_1 = -(\sum_{i,j\in N} w_{ij} * z_{ij} + \sum_{i,j\in N} w_{ij} * y_i * y_j) \qquad (4\text{-}1)$$

通过多轮遗传迭代，直至种群最优值不再更新或是达到遗传代数，完成遗传，输出最优子代。

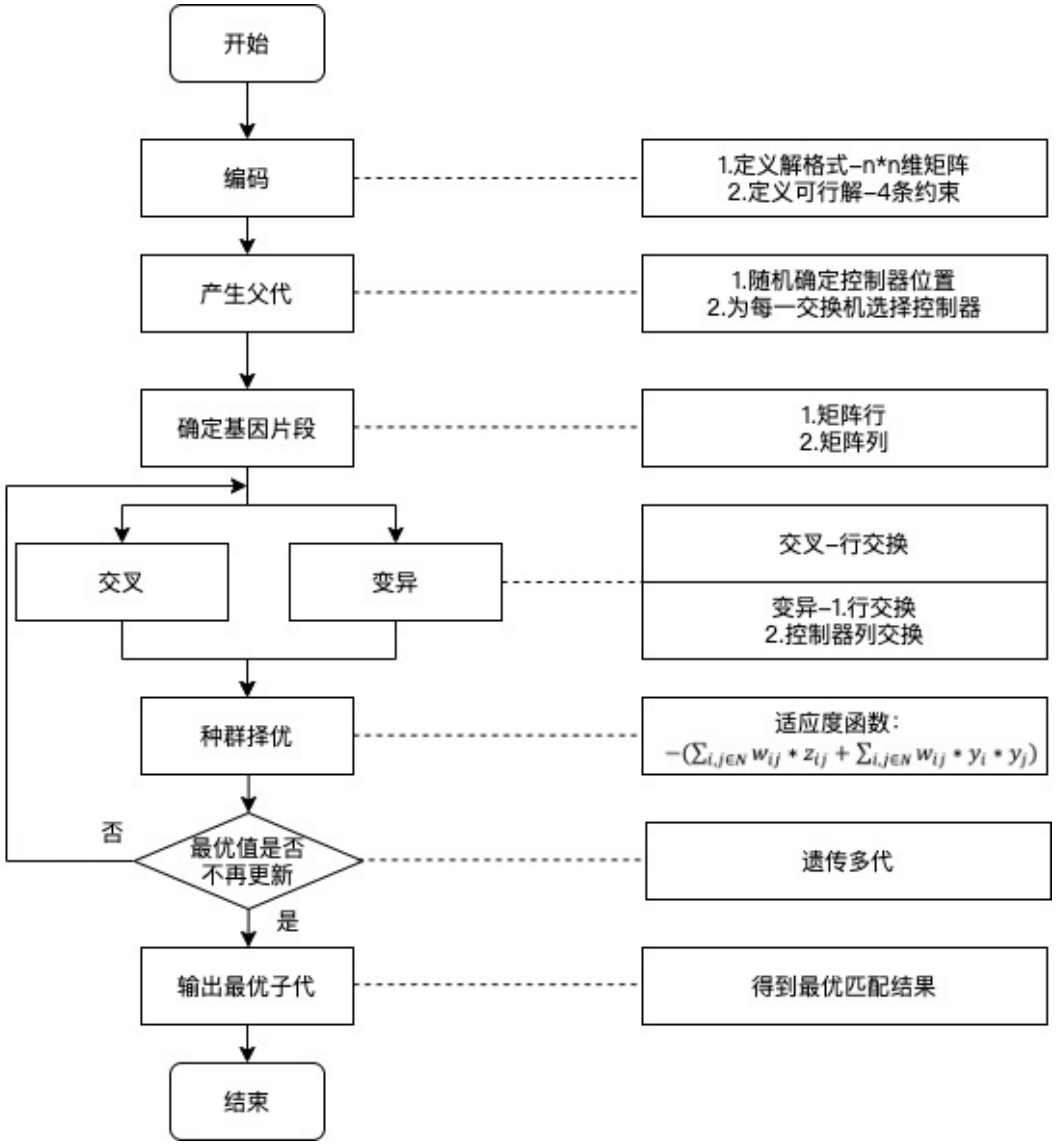


图 4-2 SM-GA 算法框图

Figure 4-2 Block diagram of SM-GA

SM-GA 算法对遗传算法中编码以及交叉变异进行了改进，以适应问题的特点，

其中编码与产生父代将在 4.2.2 节进行说明，改进的交叉变异规则将在 4.2.3 节进行说明。

4.2.2 编码与产生父代

本小节将对 SM-GA 算法中可行解的定义进行介绍，并说明产生初始父代的方法，也即定义遗传算法中的编码和产生父代过程。

在控制器全局最优位置问题中，可行解的定义需要包含控制器的部署情况以及控制器与交换机之间的控制匹配情况，其中，控制器部署信息包含于控制器与交换机之间的匹配关系中，因此用$n*n$维 0-1 匹配关系矩阵表示个体，如公式（4-2）所示，矩阵中n为交换机个数，m为控制器个数。该匹配关系矩阵中第i行第j列的元素z_{ij}表示交换机i是否被控制器j控制，若控制器j控制交换机i，则$z_{ij}=1$，否则$z_{ij}=0$。通过匹配关系矩阵可以分析得出控制器位置，如果某行对角元素为 1，则该行代表的位置部署了控制器，即若$z_{ii}=1$，则位置i部署了控制器。

$$\begin{matrix} z_{11} & \cdots & z_{1n} \\ \vdots & \ddots & \vdots \\ z_{n1} & \cdots & z_{nn} \end{matrix} \tag{4-2}$$

此外，并不是所有的$n\times n$维 0-1 矩阵都是可行解，还需要对可行解进行约束，具体的约束有以下五条。

(1) 矩阵的每一行最多有 1 个 1，即每个交换机只能被一个控制器控制；

$$\sum_j z_{ij}=1 \tag{4-3}$$

(2) 矩阵的每一列中，1 的个数不能超过控制器的控制能力c，即控制器所控制的交换机个数不能超过其控制能力；

$$\sum_i z_{ij}\le 1 \tag{4-4}$$

(3) 只有对角元素为 1 时，该列其他位置才可以为 1，即只有该位置部署控制器，才有控制其他交换机的能力；

$$if\ z_{jj}=0,\ z_{ij}=0 \tag{4-5}$$

(4) 整个矩阵中 1 的个数为交换机总数，即交换机个数约束。

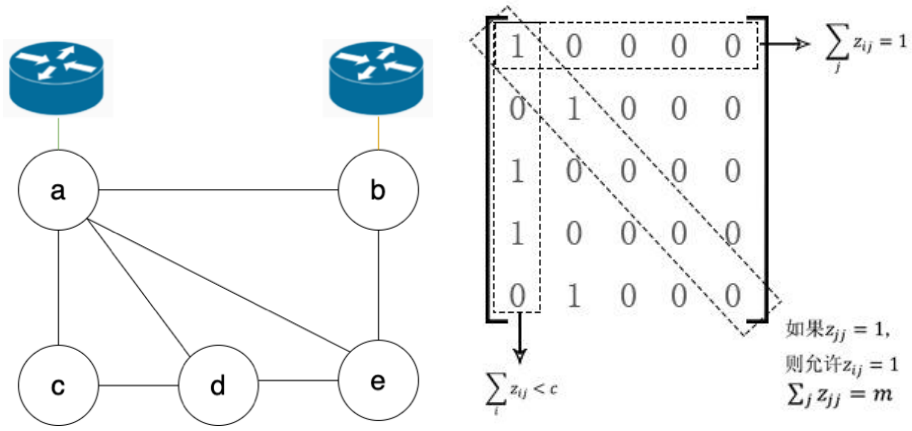
$$\sum_i\sum_j z_{ij}=n \tag{4-6}$$

(5) 对角线中 1 的个数为控制器个数，即控制器个数约束。

$$\sum_j z_{jj}=m \tag{4-7}$$

根据遗传算法个体的定义，进行初始父代的产生，产生后的父代将作为后续遗传的初始种群。然而如果直接随机生成父代，会导致产生的父代合格率很低，因此需要设计特殊的产生规则，以提升产生父代的合格率。设计产生父代的步骤如下。

首先随机选择r个点作控制器，将对应的对角线元素置为 1，即若i为控制器，则$z_{ii}=1$；其次，随机确定每个控制器控制的交换机个数，此时除控制器外，还有$s-r$个交换机待分配给r个控制器，所以将$s-r$分成r份，每份不超过$c-1$；最后确定每个控制器控制哪些交换机，也即矩阵中每行哪些位置为 1，根据上一步中确定的每行 1 的数目要求，在对应行中，随机选择不含 1 的列置为 1。这样就随机产生了符合要求的可行父代解，可行解的形式如图 4-3 所示，左图 a)为示例拓扑，在交换机 a 和 b 处部署控制器，分别控制交换机 a、c、d 和交换机 b、e，该拓扑对应的矩阵如图 b)所示。



a) 示例拓扑　　b) 示例可行解

a) Example topology　　b) Example of feasible solution

图 4-3 SM-GA 父代可行解示例

Figure 4-3 Example of feasible solution for parent for SM-GA

4.2.3 交叉与变异

直接进行基因片段的交叉和变异会导致产生的后代中合格率较低。原因在于可行解的要求较为苛刻，对每一行、每一列中 1 的个数和依赖关系都有较强的定义，直接交换某些行或某些列会违反这些规则，产生无效解，因此需要在遗传算法交叉变异操作时，设计特殊的交叉变异规则，使得产生的子代也尽可能是可行解。

(1) 交叉

定义基因交叉的基本片段为基因行。在交叉时，随机选择一行，将两个基因的对应行进行互换即完成一次交叉，如图 4-4 所示，选择矩阵 1 和矩阵 2 的第 5 行进行互换，产生两个新个体。交叉后，产生的新个体还需要验证是否满足可行解的约束，需要多次交叉时则重复多次交换。而若交换基因列，会导致一行中 1 的个数超过 1，容易产生非法解，如在图 4-5 中，选择矩阵 1 和矩阵 2 的第 1 列进行交叉，交换后得到矩阵 3 和 4，矩阵 3 中交换机个数不符合要求，不满足约束（4-3）和

约束（4-6），矩阵 4 中，第三行有两个 1，不满足每行只有 1 个 1 的可行解要求，违反了约束（4-3）和约束（4-6）。因此，选择基因行作为 SM-GA 算法中交叉的基本基因片段。

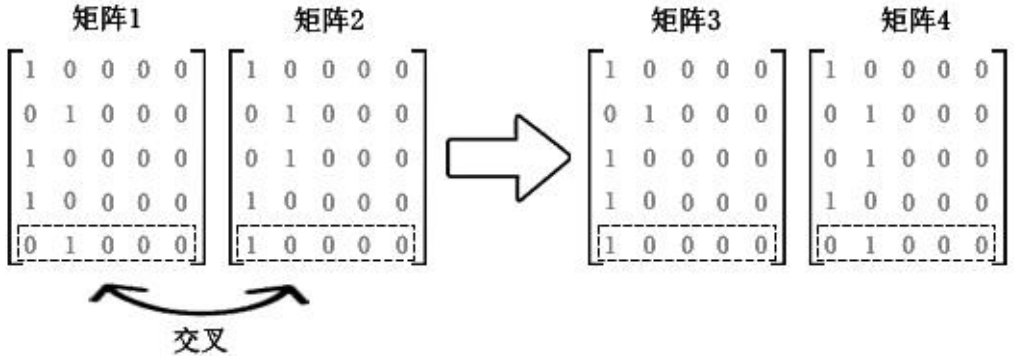


图 4-4 匹配矩阵交叉规则示例

Figure 4-4 Example of match matrix crossover rule

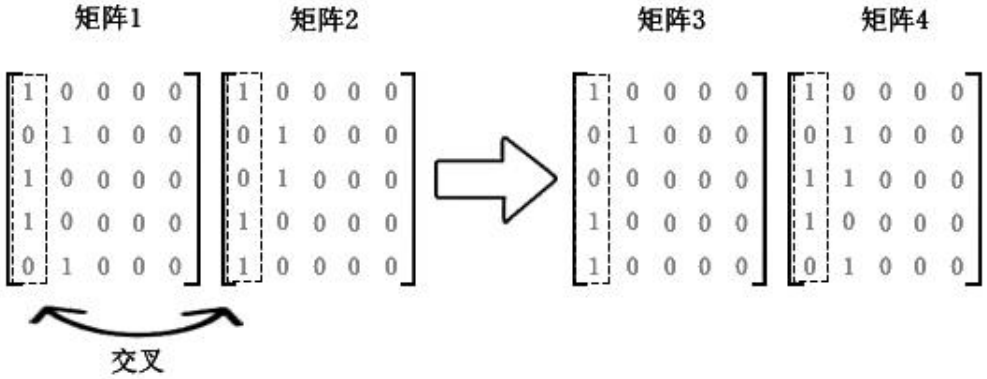


图 4-5 匹配矩阵交叉错误规则示例

Figure 4-5 Wrong example of match matrix crossover rule

(2) 变异

定义基因变异的基本片段是基因行与基因列。变异是指某些基因片段发生突变，变为其他取值，产生未出现过的个体，但在本问题中，矩阵为 0-1 矩阵，直接改变一个位置的值，只能是从 0 变为 1 或是从 1 变为 0，由于可行解的要求是 1 的个数为拓扑点数，即确定值，单个基因的变异会违反这一规则，产生的子代都是非法解，因此考虑在矩阵内部进行行列交换。基因变异的目的是产生原本不存在的控制器匹配关系和控制器放置位置。

为了产生新的控制器匹配关系，选择交换个体中的随机两行。矩阵行可以得出交换机由哪个控制器控制，通过互换两行，可以改变交换机所属的控制器方案，同时不超过控制器的匹配能力，因此可以对矩阵中的任意两行进行交换以改变匹配关系，如图 4-6 所示，在矩阵中，选择第 3 行与第 5 行进行交换，得到新的匹配关系矩阵 2，在交换后需要判断新生成的解是否为可行解。

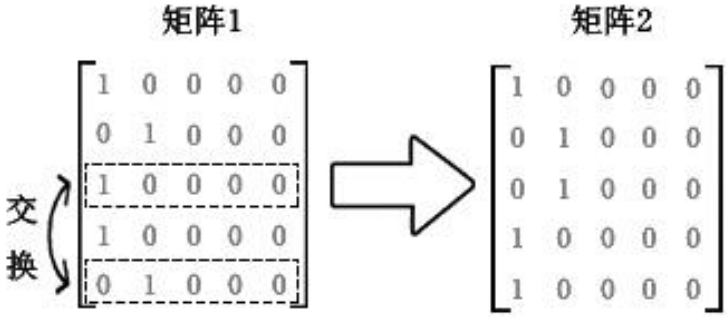


图 4-6 匹配矩阵变异规则 1 示例

Figure 4-6 Example of match matrix mutation rule 1

但图 4-6 所示的变异规则只能产生新的匹配方案，无法生成新的控制器部署方案，为了得到新的控制器部署方案，需要更换对角线元素，由于控制器个数有限制，可以通过交换控制器行的方式来进行变异。首先随机选择控制器列第j列，之后在第j列中随机寻找为 1 的行，例如选择$c_{ij}=1$，交换第i列与第j列。这样，将控制器从部署位置j换到了部署位置i，且原来由控制器j控制的交换机改由控制器i负责，保证了控制器个数不发生变化，使得产生的解尽可能是可行解。如图 4-7 所示，原始控制器部署位置为 1 和 2，在矩阵 1 中首先选择控制器列第 1 列，第 1 列中含有 1 的位置为第 1、3、4 行，从中随机选择 3，将第 1 列与第 3 列进行交换，得到新的控制器部署位置为 2 和 3，如矩阵 2 所示。

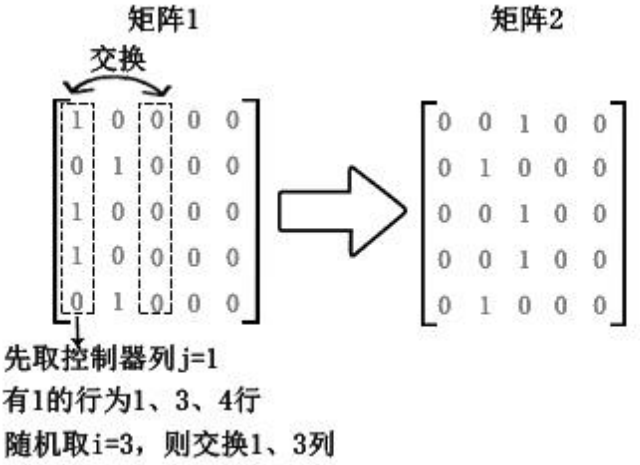


图 4-7 匹配矩阵变异规则 2 示例

Figure 4-7 Example of match matrix mutation rule 2

基于以上规则，对父代进行交叉变异得到子代，对子代中所有个体利用适应度函数计算适应值，选取适应值表现较好的个体作为下一次遗传的父代。多次重复交叉变异、择优遗传的过程，在遗传完成后，从最后一批子代中选择适应度函数值最优的个体作为模型的解，得到近似最优的控制器部署匹配方案。

4.3 基于互补交叉变异的遗传算法

4.3.1 CCM-GA 算法框图

为了求解 SDN 升级问题，本节提出基于互补交叉变异的遗传算法（genetic algorithm based on complementary crossover mutation, CCM-GA），在已知惩罚项非全局最优因子的前提下，代入 SDN 最优动态升级模型中，作为该遗传算法择优的适应度函数。SDN 升级问题是指在一个有n个点的网络拓扑中，分T次进行升级，每次选择s^t个交换机升级，部署r^t个控制器。在单步升级时，同时最大化升级带来的度收益，最小化控制时延与同步时延，最小化非全局最优控制器引入的惩罚，也即求解 3.2.3 节的（P_1^{**}）模型。基于互补交叉变异的遗传算法框图如图 4-8 所示，左图为遗传算法的流程，右图为 CCM-GA 算法对流程各部分的具体定义。

CCM-GA 算法定义个体的解格式为$2 \times n$维矩阵，并给出个体解需满足的条件。

之后根据约束条件产生初始父代，作为遗传算法的初始种群。由于 CCM-GA 算法的个体基因约束较少，可直接根据约束随机产生父代。

基因片段为交叉和变异的基础，SM-GA 算法中基因为$2 * n$维矩阵，包含交换机行与控制器行，可分别对两行进行交叉变异，此时基因片段为单个位。

CCM-GA 算法针对个体基因的特点，设计交叉变异规则。

CCM-GA 算法选择模型（P_1^{**}）的优化目标，如公式（4-8）所示。选其中控制时延需要根据控制器与交换机匹配方案得出，在已知 SDN 交换机列表和控制器列表时，求最优匹配方案可以看作指派问题，利用匈牙利算法进行求解，可基于 Gurobi 求解器直接进行这一指派问题的求解。

$$Fitness_2 = \sum_{i \in N} \beta^t * d_i * x_i^t - \eta * \sum_{i,j \in N} w_{ij} * z_{ij}^t - \sum_{i \in N} \gamma^t * q_i * y_i^t \quad (4\text{-}8)$$

通过多轮遗传迭代，直至种群最优值不再更新或是达到遗传代数，完成遗传，输出最优子代。

CCM-GA 算法对遗传算法中编码以及交叉变异进行了改进，以适应问题的特点，其中编码与产生父代将在 4.3.2 节进行说明，改进的交叉变异规则将在 4.3.3 节进行说明。

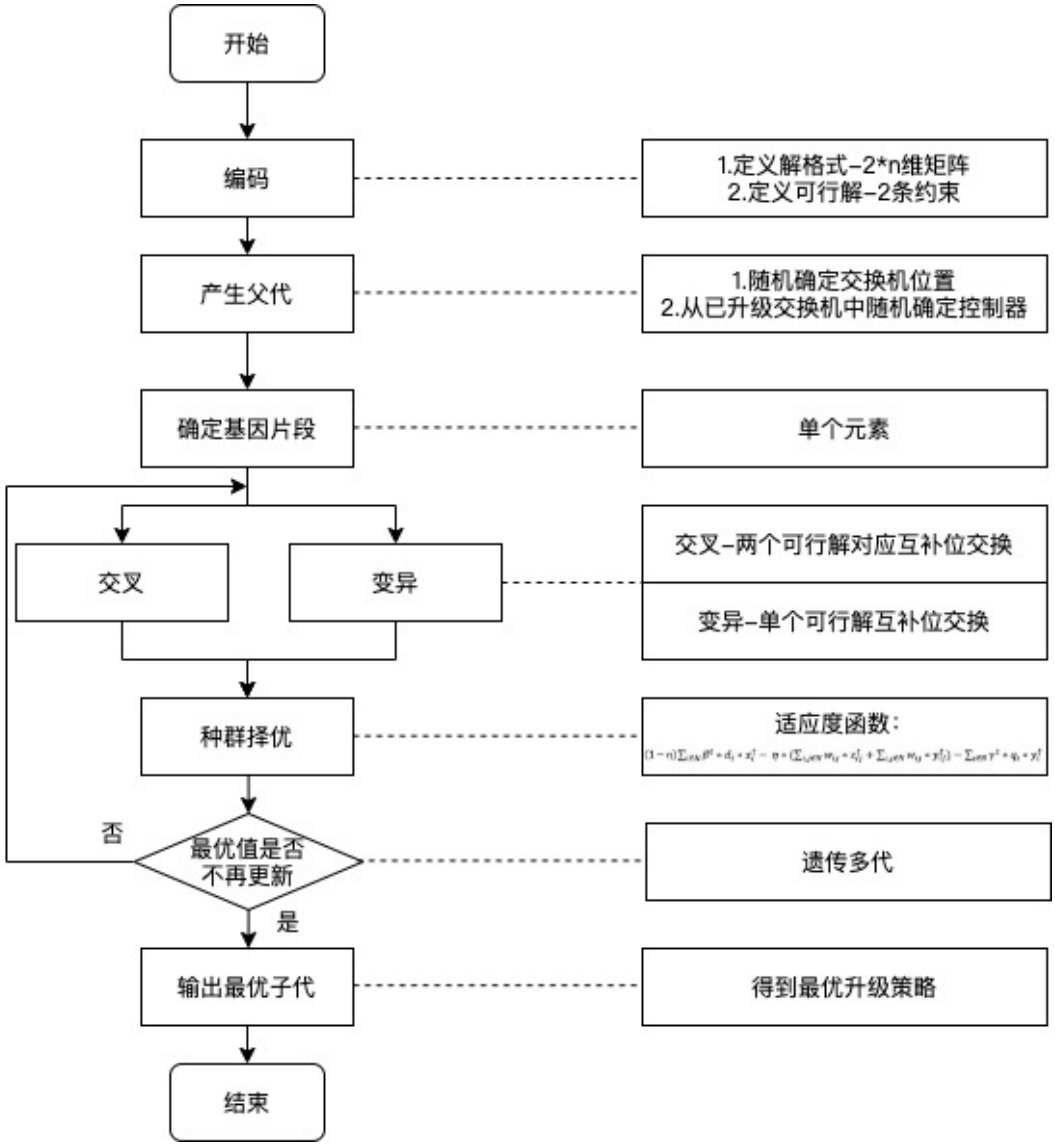


图 4-8 CCM-GA 算法框图

Figure 4-8 Block diagram of CCM-GA

4.3.2 编码与产生父代

在基于互补交叉变异的遗传算法中，个体的定义需要包括交换机的升级状态、控制器部署位置以及两者之间的匹配关系，这会导致待求解的变量增多。如果继续使用匹配关系矩阵作为可行解的基因，虽然可以根据匹配矩阵分析得出交换机升级情况与控制器部署情况，但变量增多会导致匹配矩阵的约束增多，此时进行交叉变异产生非法解的概率更大，因此需要重新定义个体基因。

在 SDN 网络升级策略模型中，选用交换机升级情况和控制器部署情况的$2 \times n$维矩阵作为基因表示。如公式（4-9）所示，a_i表示交换机i是否为 SDN 交换机，若

是则为 1，否则为 0；b_i表示位置i是否放置控制器，若放置则为 1，否则为 0。根据$2 \times n$维矩阵无法得到控制器与交换机的匹配关系，需要引入匈牙利算法得到最优的匹配方案以及最短控制时延。将最短控制时延代入适应度函数，计算每一个体的适应值，并进行择优筛选，得出最优函数值的升级方案。

$$\begin{bmatrix} a_1 & a_2 & \cdots & a_n \\ b_1 & b_2 & \cdots & b_n \end{bmatrix} \tag{4-9}$$

此外，并不是所有的$2 \times n$维 0-1 矩阵都是可行解，还需要对可行解进行约束，具体的约束有以下三条。

(1) a_i行中 1 的个数为该步所升级的交换机个数；

$$\sum_i a_i = s \tag{4-10}$$

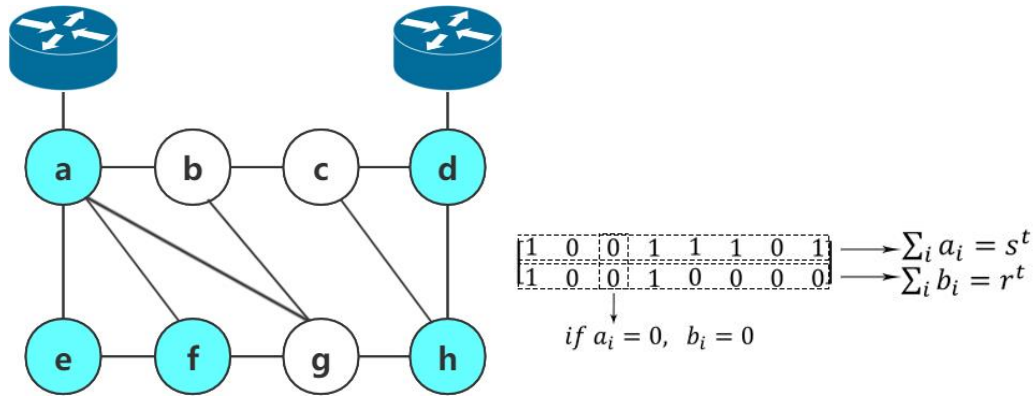
(2) b_i行中 1 的个数为该步所部署的控制器个数；

$$\sum_i b_i = t \tag{4-11}$$

(3) 只有a_i为 1 时，b_i才可以为 1，即只有 SDN 交换机位置可以部署控制器。

$$if\ a_i = 0,\ b_i = 0 \tag{4-12}$$

随机产生父代时，首先确定升级哪些交换机，即从所有点中随机选择s^t个交换机升级为 SDN 交换机，得到 SDN 交换机行，也即可行解的第一行。之后从已升级 SDN 交换机列表中，随机挑选出r^t个控制器，得到控制器行，也即可行解的第二行，将两个向量组合得到父代个体，父代可行解示例如图 4-9 所示，左图 a)为示例拓扑，交换机 a、d、e、f、h 交换机升级为 SDN 交换机，在交换机 a 和 d 处部署控制器，分别控制交换机 a、e、f 和交换机 d、h，该拓扑对应的可行父代如图 b)所示。



a) 示例拓扑　　b) 示例可行解

a) Example topology　　b) Example of feasible solution

图 4-9 CCM-GA 父代可行解示例

Figure 4-9 Example of feasible solution for parent for CCM-GA

4.3.3 交叉与变异

由于个体基因中包括交换机信息与控制器信息，两者之间存在约束关系，将基因中的两行分别进行交叉变异可以提高产生后代的合格率，因此在交叉和变异时分别对交换机行与控制器行进行交叉和变异。

交叉包括交换机行交叉与控制器行交叉，交换机行交叉需要保证交换后的两个个体依然符合所升级的 SDN 交换机个数要求，因此先随机选择两个个体，挑选出两个个体中互补的片段，选择两个互补的片段进行交换，其中互补是值两个个体按位相减后，差为 1 与差为-1 的两位。如图 4-10 中第二列与第四列是互补的两位，通过互补片段相交换，把个体 1 的“10”片段与个体 2 的“01”片段进行交换，保证了两个个体交叉后仍符合每一步升级的 SDN 交换机个数限制，同时更改所升级的交换机。多次执行互补基因位交换就可以实现基因片段的交叉。控制器行交叉需要在同一种升级情况下选择两种不同的控制器部署方案进行交叉，这样可以保证不会产生非 SDN 交换机被选为控制器的非法情况。首先随机选择一种交换机升级方案，得到一个交换机升级行，在该交换机行下，选择两个控制器部署方案，同样选择其中互补的片段进行互换，如图 4-11 所示，多次互换可实现控制器基因段的交叉。

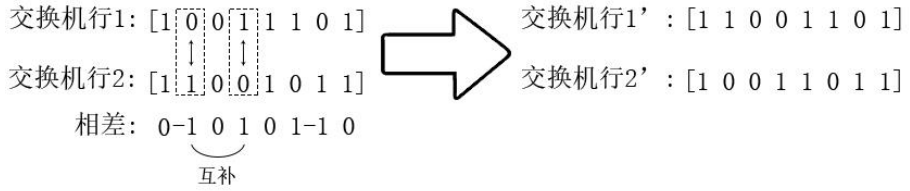


图 4-10 交换机行交叉示例

Figure 4-10 Example of switch row crossover

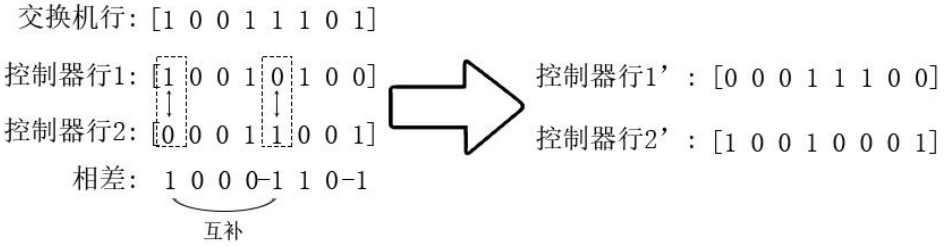


图 4-11 控制器行交叉示例

Figure 4-11 Example of controller row crossover

变异分为交换机行变异与控制器行变异。交换机行变异时需要保证所升级的交换机个数正确，因此选择在该行中随机选择两位进行互换，如图 4-12 所示，将第二列的 1 与第四列的 0 进行交换，多次执行互换得到实现多次变异。

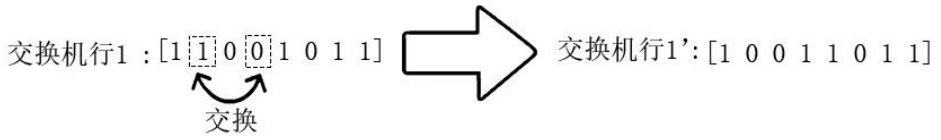


图 4-12 交换机行变异示例

Figure 4-12 Example of switch sequence mutation

控制器行的变异在保证控制器个数正确的同时，还要保证新产生的控制器同时是 SDN 交换机，因此在控制器行中随机选择为 1 的一位，再从已升级交换机中选择不是控制器的一位，将两位交换，如图 4-13 所示，将控制器的第 4 列与第 5 列交换，得到新的控制器行，与原有的交换机行重新组成新的个体。控制序列变异可以将原来的一个控制器移除，在其他 SDN 交换机位置处放置一个新的交换机。

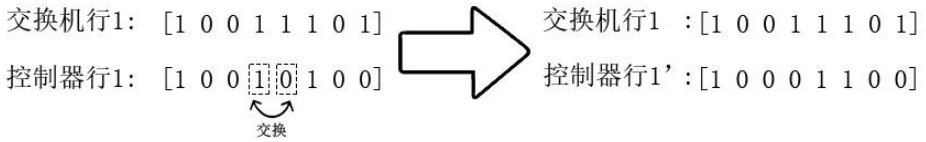


图 4-13 控制器序列变异示例

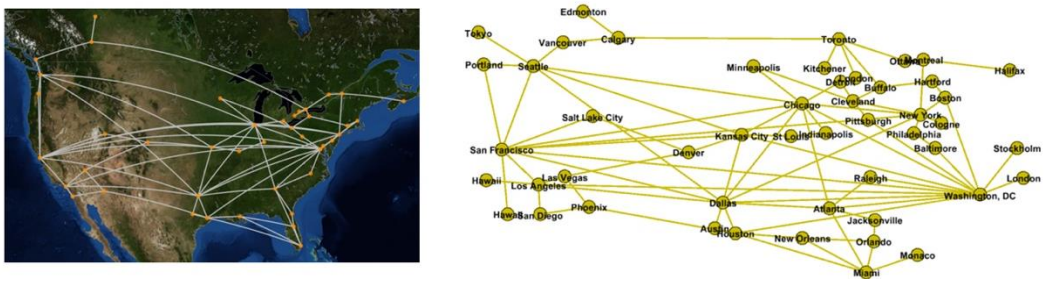
Figure 4-13 Example of controller sequence mutation

交换机行交叉变异后，重新生成新的控制器行，组合生成新的子代，控制器子列交叉变异后，与原先相对应的交换机子列组合生成新的子代，将这些子代连同父代利用适应度函数进行择优遗传。经过多代遗传后，选择适应度最优的个体作为最终解，得到交换机升级行和控制器部署行，利用匈牙利算法求解得到最优匹配方案，即得到该步的升级策略。对每一步升级都利用遗传算法求解，最终得到完整的 SDN 动态升级策略。

4.4 算法验证

本节将利用基于 SM-GA 与 CCM-GA 启发式算法求解真实网络拓扑的 SDN 动态升级策略，首先在网络拓扑中利用 SM-GA 算法求解全局最优控制器布局，再计算惩罚项，输入到 CCM-GA 算法的适应度函数中，求解 SDN 动态升级策略。

选用美国真实骨干网 UUNET 进行遗传算法模型的求解，网络拓扑数据来源于 Topology Zoo[59]。UUNET 拓扑包含 44 个点，47 条边，是 IP 层客户网，网络拓扑如图 4-14 所示，a)图为拓扑图在真实地图上的展示，b)图为抽离出的拓扑图。



a) 带地图的网络拓扑　　b) 网络拓扑

a) Topology with map　　b) Topology without map

图 4-14 UUNET 网络拓扑图

Figure 4-14 Topology of UUNET

4.4.1 全局最优控制器布局求解

首先利用遗传算法求解纯 SDN 网络下控制器最优布局。设定参数如下，网络中部署 5 个控制器，每个控制器的控制能力为 10 个交换机，父代数目、子代数目、交叉次数、变异次数均为 100。遗传算法中的适应度函数值随着遗传代数的表现如图 4-15 所示，随着遗传进行，控制器布局总时延逐渐变小，直至不再改变，子代趋于近似最优。

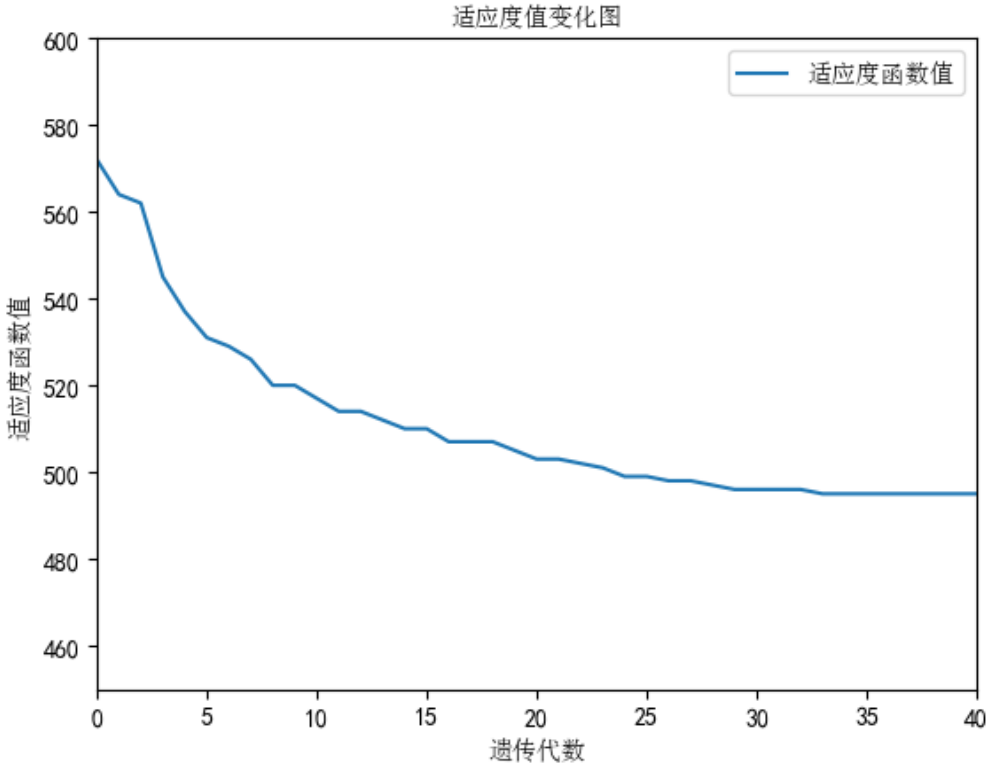


图 4-15 适应度函数值变化曲线

Figure 4-15 The curve of fitness function value

将所有点编号为 1-49，纯 SDN 网络的最优控制器位置及匹配方案如图 4-16 所示，同色同字母圆点表示归属于同一控制器控制。为了对比分析遗传算法求得的纯 SDN 网络控制器布局与真实最优布局之间的差异，由于全局最优布局变量较少，可以直接利用最优化求解器 Gurobi 求解全局最优控制器部署位置，最优布局拓扑如图 4-17 所示。两种算法得到的控制器布局并不相同，遗传算法的控制器较为集中，可以降低同步时延，最优化求解器求得的控制器布局相较来说比较分散，但所控制的交换机有明显的社区聚类，可以很好地降低控制时延。

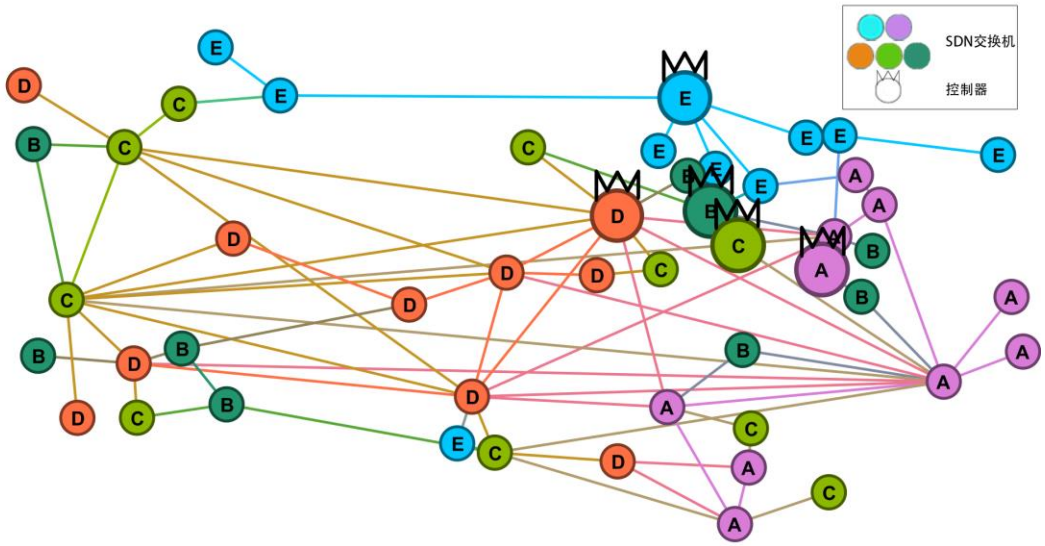


图 4-16 基于 SM-GA 的 UUNET 全局最优控制器布局图

Figure 4-16 The global optimal controllers layout of UUNET of SM-GA

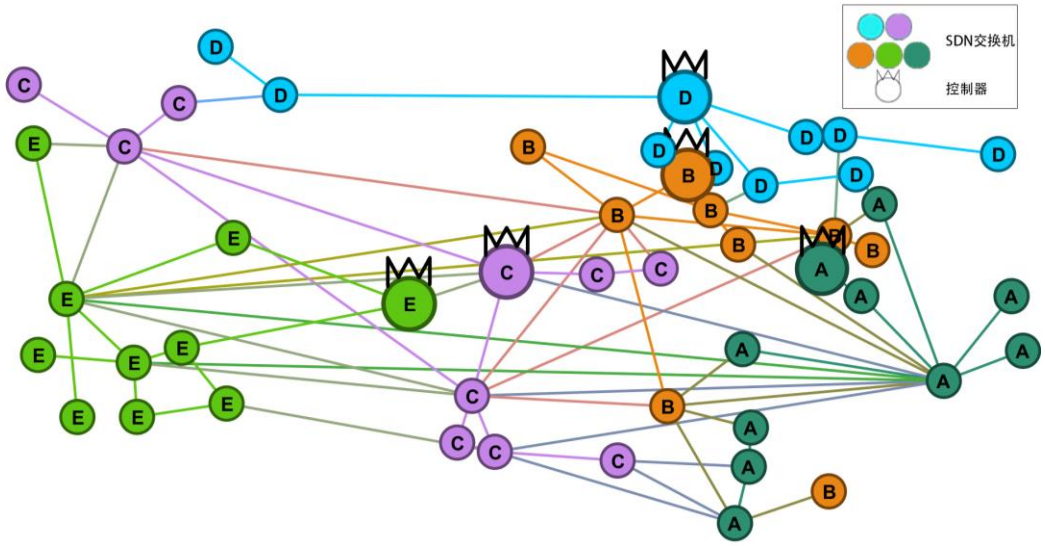


图 4-17 UUNET 全局最优控制器布局图

Figure 4-17 The global optimal controllers layout of UUNET

进一步，计算两种布局的时延值，结果如表 4-1 所示。其中，遗传算法求得的控制器布局网络的控制时延为 471 ms，同步时延为 24 ms，时延和为 495 ms；最优

化求解器求得的控制器布局的控制时延为 375 ms，同步时延为 96 ms，总时延为 471 ms。遗传算法的最终总时延表现差于最优布局，误差为 0.051。

表 4-1 两种算法网络时延对比

Table 4-1 The comparison of delay between two algorith ms

算法	控制时延(ms)	同步时延(ms)	总时延(ms)
SM-GA	472	24	495
最优布局	375	96	471

4.4.2 SDN 动态升级策略求解

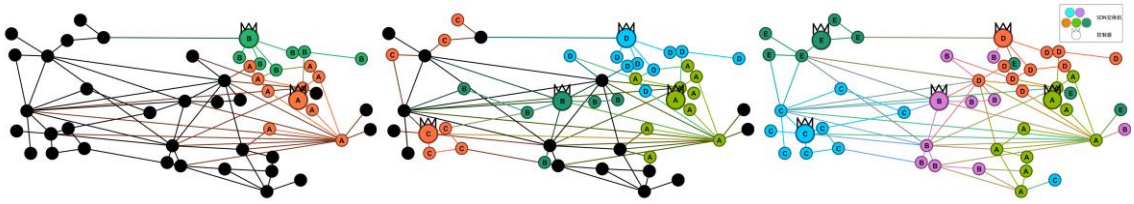
根据全局最优控制器位置，计算惩罚项中非全局最优因子，作为 CCM-GA 算法的输入，求解 SDN 网络升级策略。第一次升级时升级交换机 17 个，放置控制器 2 个。第二次升级时升级交换机 16 个，放置控制器 2 个，第三次升级时升级交换机 16 个，放置控制器 1 个。升级参数如表 4-2 所示。

表 4-2 升级参数

Table 4-2 Migration parameters

时间步	交换机个数	控制器个数
1	17	2
2	16	2
3	16	1

最终求解得到的逐步升级方案如图 4-18 所示，同色同编号的圆表示由相同的控制器控制。可以看出，利用遗传算法求解得到的逐步升级方案在初始升级时会选择成簇的交换机进行升级，以使得控制时延较小。最终得到的 SDN 网络的时延和为 552 ms。而 SDN 网络全局最优化控制器部署方案的最终总时延为 471 ms，近似解与最优解的误差为 0.172。



a) 第一步　　b) 第二步　　c) 第三步

a) Step 1　　b) Step 2　　c) Step 3

图 4-18 基于 CCM-GA 的 UUNET 逐步升级图

Figure 4-18 UUNET migration process with CCM-GA

计算得到每个升级步的时延与度收益如表 4-3 所示。分析可得，虽然遗传算法求解的升级方案得到的最终纯 SDN 网络并不是最优布局，但与最优时延相差 0.172，是近似最优解。因此遗传算法为大规模网络提供了升级策略求解的可行性，且最终的求解结果为近似最优解。

表 4-3 分步升级收益损失表

Table 4-3 Gain and loss of the migration

时间步	度收益	控制时延(ms)	同步时延(ms)	总时延(ms)
1	57	12	4	16
2	99	77	98	175
3	168	356	196	552

4.5 本章小结

本章提出了在大规模网络下求解最优控制器布局的改进启发式算法，包括基于 SDN 匹配关系的遗传算法（SM-GA）以及基于互补交叉变异的遗传算法（CCM-GA），两种算法都是对现有遗传算法的改进。

在求解过程中，SM-GA 算法首先将全局最优控制器布局的可行解编码为$n \times n$维矩阵，并规定可行解需满足的条件，进而根据可行解的要求改进交叉变异规则。CCM-GA 算法将可行解编码为$2 \times n$维矩阵，并利用商用最优化求解器求解匹配问题，得出最优匹配关系。分别对交换机行与控制器行改进交叉变异规则。

对遗传算法中的编码、交叉、变异进行改进可以适应问题的特殊约束，提高子代合格率，并在实际网络中进行算法求解，验证了改进的启发式算法可以有效得求出近似最优解。

5 基于强化学习的求解算法

第 3、4 章中惩罚项的引入与求解是本文同时回答三个问题（3W）的关键，但人为定义惩罚项并非最优，而强化学习很好地契合了本研究问题的双向时序约束特性，可自学习未来收益对当前的折现，因此本章提出基于强化学习的求解算法。首先介绍该算法的研究动机，并对算法进行描述，之后基于 Q-learning 算法与 DQN 算法分别针对小规模和大规模场景进行强化学习中的价值计算，最后在真实网络拓扑中验证两种算法，说明基于强化学习求解 SDN 最优动态升级策略的可行性。

5.1 研究动机

SDN 网络升级过程中，每一次升级选择时，决策都只与当前时间点的网络状态有关，而与之前的升级状态无关，具有马尔可夫性。而强化学习的大多数算法是以马尔可夫决策过程为基础发展起来的，可用于求解马尔可夫决策过程，故本节研究基于强化学习的方法求解 SDN 网络动态升级策略。

SDN 网络升级过程中有两种目标，一是局部双目标，即单步升级过程中最大化升级的度收益与最小化控制器部署引入的控制时延与同步时延和；二是全局单目标，即最小化纯 SDN 网络的控制时延与同步时延之和。在整个升级过程中存在局部双目标与全局单目标的相互影响，在单步升级时既需要考虑局部双目标，也需要考虑全局单目标。而强化学习在某一状态下选择动作时，既考虑当前动作的即时奖励，也考虑未来预期收益对当前的折现。因此本文问题与强化学习的内涵相一致，都是选择动作来最大化当前收益与未来可能的预期收益，选择动作时可能为了长远利益牺牲眼前的利益。在强化学习算法中，全局收益已经隐含于未来收益中，不再需要人为定义惩罚项，可以直接通过自学习得到惩罚项。

综上，本文考虑基于强化学习进行 SDN 网络动态升级策略的研究。首先搭建强化学习模型，定义模型中的状态、动作以及奖励。之后分别利用 Q-learning 算法与 DQN 算法进行价值的计算与模型的求解。最后，将模型与算法结合，在真实网络中进行算法的仿真验证。

5.2 算法描述

基于强化学习的求解算法需要将 SDN 网络动态升级问题映射为强化学习模型，定义其中的状态、动作以及奖励函数，便于在后续章节中利用强化学习算法进行价

值计算与问题求解。

本文提出的强化学习求解算法的整体模型如图 5-1 所示。其中，SDN 升级智能体维护着一张 Q-table 来记录每一状态—动作对的价值，或是利用神经网络来计算每个状态下，所有可能动作的价值。SDN 升级智能体与 SDN 升级环境进行交互，输入选择动作后可以得到相应的网络状态与收益奖励，形成一系列训练样本，将这些状态—动作的价值序列输入智能体，可以对 Q-table 进行更新或是对神经网络进行训练，计算状态—动作对的价值。训练好的智能体可以为环境制定策略。

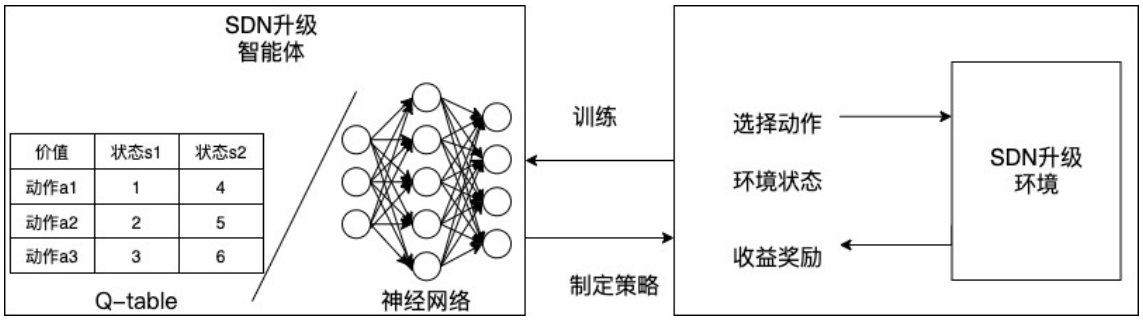


图 5-1 SDN 升级强化学习模型图

Figure5-1 Reinforcement learning model of SDN migration

SDN 升级环境包含状态与动作之间的转移关系，也即 SDN 逐步升级的过程。SDN 网络动态升级问题分T次将所有的n个交换机升级为 SDN 交换机，在第t次升级时，选择s^t个交换机进行升级，并放置r^t个控制器，升级后得到新的网络状态，在新的网络状态下，再次选择待升级的交换机和部署的控制器，直至所有的交换机均完成升级，即得到完整的升级策略。

为了说明本文选定的状态空间与动作空间，此处使用 5 个点的示例网络。示例网络拓扑如图 5-2 所示，连边上的数字表示数据经过该路径时的时延，规定整个网络分 3 次进行升级，每次最多升级 2 个交换机，部署 1 个控制器。

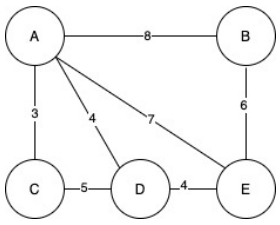


图 5-2 5 点示例网络拓扑图

Figure5-2 Example topology of 5 nodes

在定义状态空间与动作空间时，首先要保证定义的状态或动作与真实环境情况一一对应，状态的定义唯一确定一种真实网络情况，且状态空间完整表示所有可能的网络情况，动作空间也需满足这一要求。其次要尽可能缩小空间，减少状态数与动作数，以减少状态—动作对，提高强化学习运算效率。

对强化学习模型的定义如下。

(1) 网络状态（state）：将状态定义为网络升级情况，包括已升级的交换机以及已部署的控制器，用向量$[a_1, a_2, a_3, \cdots a_n; b_1, b_2, b_3, \cdots b_n]$表示，其中$a_i$表示交换机$i$是否升级，$b_i$表示交换机$i$的位置是否放置控制器。

$$s = [a_1, a_2, a_3, \cdots a_n; b_1, b_2, b_3, \cdots b_n],\ a_i, b_i \in \{0,1\} \quad (5\text{-}1)$$

(2) 选择动作（action）：将动作定义为选择哪一个网络节点，用向量$[c_1, c_2, c_3, \cdots c_n]$来表示，其中$c_i$表示在当前时间步是否升级交换机$i$或是否在该点部署控制器，动作数为$n$。

$$a = [c_1, c_2, c_3, \cdots c_n],\ c_i \in \{0,1\} \quad (5\text{-}2)$$

(3) 收益奖励：将奖励定义为状态—动作对所产生的新网络升级情况的收益。具体来说，单步选择的奖励为所升级 SDN 交换机节点的度之和，如公式（5-3）所示，单步升级的奖励为度收益与时延损失的加权和，如公式（5-4）所示，单步升级包含多个细分的单步选择。最终升级的纯 SDN 网络奖励为负的时延损失，如公式（5-5）所示。

$$reward_{temp} = \sum_{i \in N} d_i * x_i^t \quad (5\text{-}3)$$

$$reward_{step} = (1-\eta)\sum_{i \in N} d_i * x_i^t - \eta * (\sum_{i,j \in N} w_{ij} * z_{ij}^t + \sum_{i,j \in N} w_{ij} * y_i^t * y_i^t) \quad (5\text{-}4)$$

$$reward_{final} = -(\sum_{i,j \in N} w_{ij} * z_{ij} + \sum_{i,j \in N} w_{ij} * y_i * y_j) \quad (5\text{-}5)$$

强化学习中状态数与动作数过多，会给强化学习中价值的计算带来压力，大规模网络场景下无法计算，且深度强化学习中输出层神经元的个数为动作数，所以动作数应尽可能小且固定。因此在定义状态与动作时需要考虑状态数与动作数的多少。

状态的定义有多种表示方式，如可将状态定义为网络中交换机和控制器匹配关系，用$n \times n$维矩阵表示。但这一定义方式会使状态数增多，状态空间过大，给强化学习中价值的计算带来压力。本文将状态定义为网络中已升级的 SDN 交换机向量以及已部署的控制器向量，即给定时间步下，网络中哪些交换机已经被升级为 SDN 交换机，哪些位置已经被部署了控制器。用$2 \times n$维向量$[a_1, a_2, a_3, \cdots a_n; b_1, b_2, b_3, \cdots b_n]$来表示 SDN 网络升级情况，其中$a_i$表示交换机$i$是否已升级，$b_i$表示交换机$i$的位置是否已经部署了控制器。但交换机向量和控制器向量相比于匹配关系矩阵所含的信息量变少，不能直接得到匹配关系，也无法直接计算得到该状态下的时延。在实际计算中，由于给定交换机向量和控制器向量可以唯一确定该配置下的最优控制器匹配方案，进而求得控制时延与同步时延，因此在

求动作的奖励值时，首先根据状态求得当前动作的最优控制器匹配方案，之后计算给定匹配方案下的时延值。

动作有多种表示方法，如可将动作定义为某一升级步所升级的所有交换机及控制器的可能组合，此时动作数为$C_n^{s^t} \times C_{s^t}^{r^t}$。如在 5 点示例网络中，第一次升级动作可能为升级交换机 A 和 B，并在 A 位置部署控制器，动作用向量$[1,1,0,0,0; 1,0,0,0,0]$表示。但依照这一动作定义方式，在该状态下的动作数为$C_5^2 \times C_2^1 = 20$，表示从 5 个待选交换机中选择 2 个进行升级，有$C_5^2 = 10$种可能的选择方式，之后再从已升级的 SDN 交换机中选择 1 个作为控制器，有$C_2^1 = 2$种可能的选择方式，共 20 种可能的动作，此种动作定义方式会导致动作空间数呈阶乘倍的增加，且动作数不固定。为了缩小动作空间，本文将单步升级进一步细分，每一升级步可细分为s^t次选择交换机与r^t次选择控制器，称为单步选择，可以将动作空间缩小为n。例如在 5 点示例网络中，第一次升级 2 个交换机，部署 1 个控制器，可以再细分为选择交换机 A，选择交换机 B，部署控制器 A，共 3 次选择。但每次选择时只能从 5 个点中选择 1 个，可选动作数为 5。因此定义动作为n维向量$[c_1, c_2, c_3, \cdots c_n]$，动作数为$n$，该动作表示升级交换机还是部署控制器则由当前网络状态决定。如果当前所升级的交换机已满足单步升级的交换机个数限制，但控制器数目不满足要求，则该动作表示部署控制器，否则表示升级交换机。

奖励是对状态—动作对的奖励，由于在本问题中，每一状态—动作对唯一确定一个下一状态，因此奖励是对下一状态网络升级情况的收益计算。由于动作将单步升级进行了细分，所以不同状态的奖励计算方式不同。如果下一状态并未完成单步升级，如 5 点网络第一步升级，选择动作$[1,0,0,0,0]$，也即选择了交换机 A 升级，下一状态为$[1,0,0,0,0; 0,0,0,0,0]$，此时并没有完成第一步的升级，而是处于中间过渡状态，定义其奖励为已升级交换机的度之和，如公式（5-3）所示。如果下一状态完成了单步升级，如 5 点网络，在状态$[1,1,0,0,0; 0,0,0,0,0]$下选择动作$[1,0,0,0,0]$，也即选择 A 点部署控制器，下一状态为$[1,1,0,0,0; 1,0,0,0,0]$，此时完成第一步的升级，定义其奖励为该升级布局下，SDN 交换机节点的度收益与控制时延及同步时延的加权和，如公式（5-4）所示。当网络完成升级时，所有交换机都是 SDN 交换机，此时定义其奖励为该布局下负的控制时延与同步时延之和，如公式（5-5）所示。此外，还有一些动作为非法动作，如对已升级的交换机再次进行升级，对已放置的控制器再次放置，或是在非 SDN 交换机位置部署控制器，非法动作的奖励定义为负无穷。

价值（value）是对每一状态下，选择每一动作的当前收益与未来预期收益的加权和，所以每一个状态—动作对都有其价值，所有的状态—动作对构成价值表。价值与奖励的不同之处在于，奖励（reward）定义的是当前动作的即时回报，而价值

（value）则定义的是未来一系列动作的平均回报。强化学习在学习阶段，通过环境反馈的奖励，迭代计算出每一状态—动作对的价值，在评估阶段，根据所处状态，选择该状态下最优价值的动作进行执行。

对 SDN 升级问题的强化学习环境定义如图 5-3 所示。

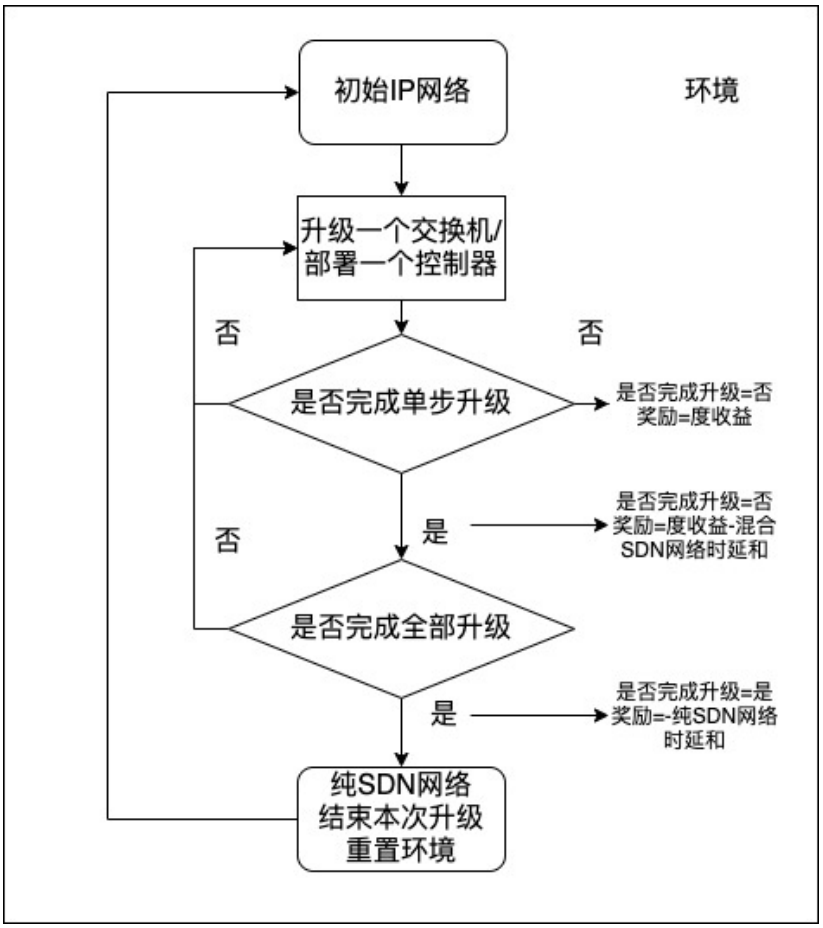


图 5-3 SDN 升级的强化学习环境

Figure 5-3 Reinforcement learning environment for SDN migration problem

5.3 价值计算

强化学习要解决的问题是：智能体要学习一个策略π，这个策略π定义了从状态到动作的映射关系$\pi: S \to A$，也就是说，决策者在任意状态s_t下所能执行的动作为$a_t = \pi(s_t)$。如何选择策略是强化学习的最终目的，如果计算出每一个状态—动作对的价值，那么在每一次选择动作时，可以采用贪婪的决策方法，选择最大价值的动作去执行，即可得到策略。其中利用表格存储状态—动作的价值的方法是 Q-learning 算法，利用深度神经网络来拟合价值函数的方法是 DQN 算法，这两种算法分别适用于小规模网络和大规模网络，本文将采用这两种算法进行价值计算。

5.3.1 基于 Q-learning 的价值计算

小规模网络的状态、动作数较少，可以利用 Q-learning 算法计算出每一状态—动作对的价值，进而根据价值得到问题的解。此时将动作定义为单步升级，不再细分单步选择。根据 2.3.3 节的推导分析，得出动作价值函数的迭代公式如（5-7）所示，$Q(s_t, a_t)$是该状态的当前价值。

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(R_{t+1} + \lambda max_a Q(S_{t+1}, a) - Q(S_t, A_t)) \quad （5\text{-}7）$$

利用 Q-learning 算法进行强化学习的示意图如图 5-4 所示，Q-learning 算法的流程如图 5-5 所示[56]。

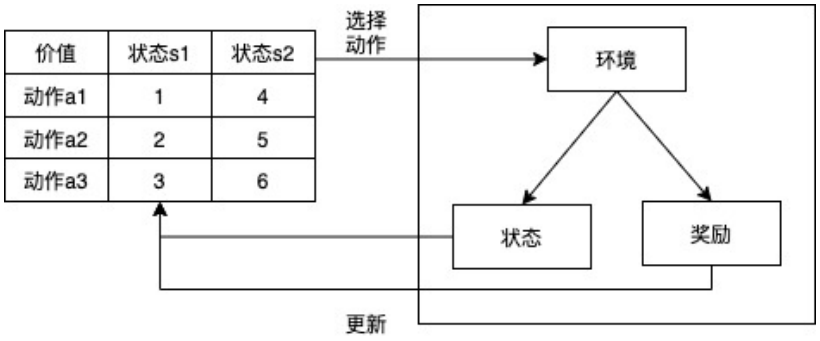


图 5-4 基于 Q-learning 算法的强化学习

Figure 5-4 Reinforcement learning based on Q-learning algorithm

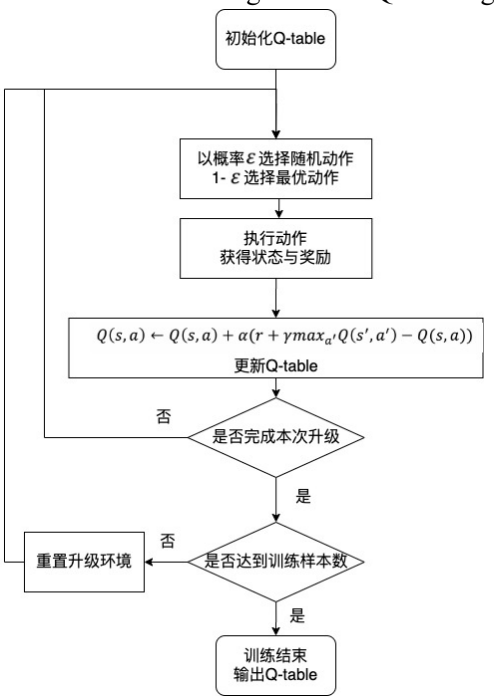


图 5-5 Q-learning 算法流程

Figure 5-5 Q-learning algorithm flow

利用 Q-learning 算法进行模型求解时，算法的状态数会随着网络节点数的增多指数增加。当网络节点为n，每次升级s^t个交换机，部署r^t个控制器时，状态数计

算公式如（5-8）所示，此时 Q-table 过大，会导致计算时间变长，占用空间过多，甚至无法计算。此外，Q-table 复杂度变高时，算法的搜索效率也会大大下降，下一小节将提出 DQN 算法，利用神经网络代替 Q-table，进行大规模网络下，求解强化学习模型中的价值。

$$num = \sum_{t=0}^{T} c(n,\ \sum_{i=0}^{t} s^i) * c(\sum_{i=0}^{t} s^i,\ \sum_{i=0}^{t} r^t) \qquad (5\text{-}8)$$

5.3.2 基于 DQN 的价值计算

当利用 Q-learning 算法进行大规模网络升级策略求解时，模型的状态数会呈指数增加，导致 Q-table 过大，无法进行 Q-table 的存储与更新计算，所以 Q-learning 适用于状态和动作空间较小的场景。另一方面，如果一个状态从未出现，Q-learning 是无法处理的，也即 Q-learning 没有预测能力与泛化能力。将深度强化学习（Deep Reinforcement Learning， DRL）应用于网络升级策略求解中能够很好地解决这些问题，本小节将介绍如何利用 DQN 算法进行价值的计算。

深度 Q 网络（Deep Q Network， DQN）是一种深度强化学习方法，是建立在传统强化学习方法 Q-learning 的基础上，利用深度神经网络代替 Q-table 得出每一状态—动作价值，并在此基础上，引入经验回放、固定目标网络等策略，实现深度神经网络的训练与预测。

5.2 节对 SDN 升级问题进行了强化学习建模，定义了状态、动作以及奖励，使得智能体在环境中可以独立的完成一次升级。将状态、动作、奖励以及动作-状态转移关系与奖励函数计算进行集成，搭建 SDN 网络升级环境，SDN 网络升级环境可以在输入动作时，根据环境所处的状态，输出下一状态与奖励。

之后搭建神经网络，并利用 SDN 升级环境中得到的样本训练神经网络。在 DQN 算法中，目标网络与评估网络结构一致，但参数不同步更新，因此搭建两个结构完全一样的网络，分别是评估网络（Q-evaluation）和目标网络（Q-target）。其中，评估网络的参数随着训练不断更新，而目标网络的参数在一段时间内是固定的，可以看作是评估网络的一个历史版本，训练一定次数后，利用评估网络的参数更新目标网络。在搭建神经网络时，本文选择三层全连接神经网络，输入为状态，输出层的神经元个数与动作数一致，输出的结果是在当前状态下，每一个可能动作的 Q 值。这样就可以得到每一状态下，不同动作的 Q 值，也即$Q(s,a)$值。

DQN 算法通过智能体与环境的交互获得训练样本，训练神经网络，预测每一状态下各动作的价值，DQN 算法示意图如图 5-6 所示，算法流程如图 5-7 所示[58]。首先智能体在环境中采集训练样本，从初始状态出发，使用ε-贪婪策略在环境中探索，以ε的概率随机选择动作，或是以$1-\varepsilon$的概率选择当前具有最大 Q 值的动作进

行执行，执行后在 SDN 升级环境中获得该动作的奖励与下一状态，直至到达最终纯 SDN 网络，也即最终状态，此时将状态重置为初始状态，重复以上过程，并将每次动作的奖励信息保存到经验池，用于智能体的训练。其中ε是随着训练逐渐变小的，用以逐渐降低对环境的探索，增加对现有经验的利用。当达到训练所需样本数后，通过经验回放来训练模型，在经验池中选择一定数目的样本进行批训练，训练目标是使评估 Q 网络逼近目标 Q 网络，每隔一定训练步，用评估 Q 网络的参数更新目标 Q 网络。

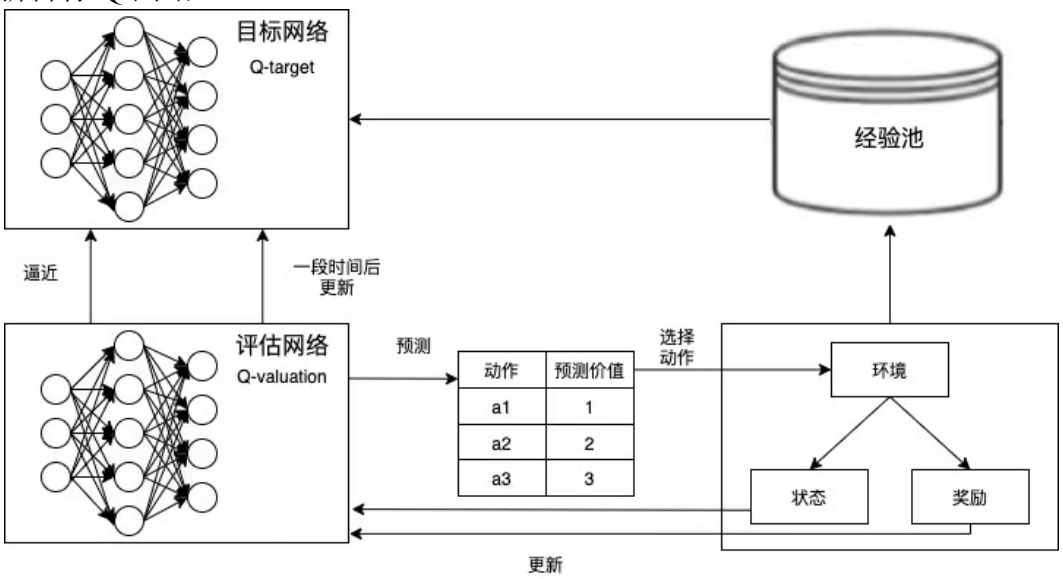


图 5-6 DQN 算法示意图

Figure 5-6 DQN algorithm schematic chart

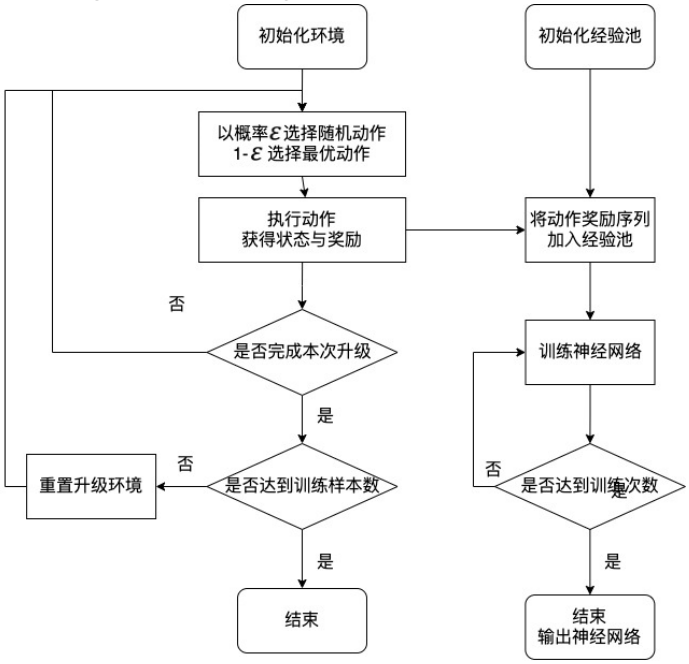


图 5-7 DQN 算法流程

Figure 5-7 DQN algorithm flow

训练完成后，可以利用训练好的神经网络进行模型的验证，将状态置为初始状态，智能体在神经网络中选择当前所处状态下价值最大的动作进行执行，直至到达最终状态，得到该算法下 SDN 网络的最优升级策略。

5.4 算法验证

本节在小规模网络与大规模网络中，分别使用 5 个点的示例网络与 49 点真实网络拓扑 UUNET 进行验证。

5.3 节基于 Q-learning 算法与 DQN 算法得到了 Q 值表或训练后的神经网络，可以根据 Q 值制定相应的策略，如本文采用贪婪策略，从初始状态开始，在每一状态下选择具有最大价值的动作进行执行，升级相应的交换机或部署相应的控制器，更新状态为下一状态。重复以上选择动作与更新状态的步骤，直至状态更新为最终状态，网络升级为纯 SDN 网络，得到完整的动态升级策略。

5.4.1 基于 Q-learning 的算法验证

本小节选用图 5-2 所示的 5 点示例网络，基于 Q-learning 算法计算 Q-table 并求解升级策略，Q-learning 算法中最小动作定义为单步升级。

首先初始化环境，将状态置为全 0，此时网络为 IP 网络。第一次升级，从 5 个传统交换机中选择 2 个进行升级，有$C_5^2 = 10$种可能的交换机选择方式，之后从所升级的交换机中选择部署控制器，每种升级方式下有$C_2^1 = 2$种可能的控制器部署方式，共$C_5^2 \times C_2^1 = 20$个可能动作，升级后有 20 个可能状态。第二次升级，从剩余 3 个交换机中选择 2 个进行升级，每个状态有$C_3^2 = 3$种可能的选择方式，在已升级但不是控制器的 3 个点中选择 1 个作控制器，有$C_3^1 = 3$种方案，共$C_3^2 \times C_3^1 = 9$个动作。第三次升级时，在已升级但不是控制器的 3 个点中选择 1 个作控制器，有$C_3^1 = 3$种方案，有 3 个可选动作。根据以上分析，计算得到 Q 值表，本小节仅展示升级决策涉及到的单步升级 Q 值表。

初始状态下的 Q 值表如表 5-1 所示，可以看出，动作 2[1,0,1,0,0,1,0,0,0,0]具有最高的价值，因此在第一步升级时选择动作 2，升级交换机 A 和 C，在 A 点部署控制器，得到下一状态[1,0,1,0,0,1,0,0,0,0]。

表 5-1 第一次升级时的 Q 值表

Table 5-1 Q-table for the first migration

动作 \ 状态	状态 [0,0,0,0,0,0,0,0,0,0]
动作 1 [1,1,0,0,0,1,0,0,0,0]	0.7006806
动作 2 [1,0,1,0,0,1,0,0,0,0]	1.7718585
动作 3 [1,0,0,1,0,1,0,0,0,0]	1.2798297
动作 4 [1,0,0,0,1,1,0,0,0,0]	-0.11142493
动作 5 [0,1,1,0,0,0,1,0,0,0]	0.08875761
动作 6 [0,1,0,1,0,0,1,0,0,0]	1.3602384
动作 7 [0,1,0,0,1,0,1,0,0,0]	-0.58119243
动作 8 [0,0,1,1,0,0,0,1,0,0]	0.29974574
动作 9 [0,0,1,0,1,0,0,1,0,0]	0.4715262
动作 10 [0,0,0,1,1,0,0,0,1,0]	-0.7456275
动作 11 [1,1,0,0,0,0,1,0,0,0]	-1.4221205
动作 12 [1,0,1,0,0,0,0,1,0,0]	0.03842248
动作 13 [1,0,0,1,0,0,0,0,1,0]	-1.1838018
动作 14 [1,0,0,0,1,0,0,0,0,1]	-1.4911616
动作 15 [0,1,1,0,0,0,0,1,0,0]	-0.73743576
动作 16 [0,1,0,1,0,0,0,0,1,0]	-0.7269001
动作 17 [0,1,0,0,1,0,0,0,0,1]	-0.8168862
动作 18 [0,0,1,1,0,0,0,0,1,0]	-0.83889884
动作 19 [0,0,1,0,1,0,0,0,0,1]	-0.3326847
动作 20 [0,0,0,1,1,0,0,0,0,1]	-0.71401703

在状态[1,0,1,0,0,1,0,0,0,0]下的 Q 值表如表 5-2 所示。该状态下具有最大价值的动作是动作 2[0,0,0,1,1,0,0,0,1,0]，执行该动作，升级交换机 D 和 E，并在交换机 D 处部署控制器，单步升级后状态为[1,0,1,1,1,1,0,0,1,0]。

表 5-2 第二次升级时的 Q 值表

Table 5-2 Q-table for the second migration

动作 \ 状态	状态 [1,0,1,0,0,1,0,0,0,0]
动作 1 [0,0,0,1,1,0,0,1,0,0]	8.084151
动作 2 [0,0,0,1,1,0,0,0,1,0]	15.449854
动作 3 [0,1,0,1,0,0,0,1,0,0]	12.795225
动作 4 [0,1,0,1,0,0,1,0,0,0]	-1.9951631
动作 5 [0,0,0,1,1,0,0,0,0,1]	-0.8300159
动作 6 [0,1,0,0,1,0,0,1,0,0]	8.731811
动作 7 [0,1,0,0,1,0,0,0,0,1]	-6.318956
动作 8 [0,1,0,0,1,0,1,0,0,0]	3.3720253
动作 9 [0,1,0,1,0,0,0,0,1,0]	0.9588636

在状态[1,0,1,1,1,1,0,0,1,0]下的 Q 值表如表 5-3 所示。第三次升级时选择动作 2[0,1,0,0,0,0,0,1,0,0]，升级交换机 B，并在 C 处部署控制器，最终状态为 [1,1,1,1,1,1,0,1,1,0]，即最终在交换机 A、C、D 处部署控制器。

表 5-3 第三次升级时的 Q 值表

Table 5-3 Q-table for the third migration

动作 \ 状态	状态 [1,0,1,1,1,1,0,0,1,0]
动作 1 [0,1,0,0,0,0,1,0,0,0]	8.2535515
动作 2 [0,1,0,0,0,0,0,1,0,0]	12.424673
动作 3 [0,1,0,0,0,0,0,0,0,1]	11.086104

以上过程采用贪婪策略，执行所处状态下具有最大价值的动作。最终得到的升级方案为，第一次升级，升级交换机 A 和 C，在 A 处部署控制器，度收益为 6，时延和为 3 ms；第二次升级交换机 D 和 E，部署控制器 D，度收益为 12，时延和为 11 ms。第三次升级，升级交换机 B，在 C 处部署控制器，最终时延为 24 ms，升级方案如表 5-4 所示。而通过求解最优化方程所得的升级方案为，第一步升级 A、C，在 A 处部署控制器；第二步升级 D、E，在 D 处部署控制器；第三步升级 B，在 C 处部署控制器，升级方案如表 5-5 所示。因此强化学习方案与最优方案一致，利用强化学习进行 SDN 动态升级策略求解是可行的。

表 5-4 基于 Q-learning 算法与最优算法求得的升级策略

Table 5-4 The migration strategy based on Q-learning algorithm

升级步	交换机	控制器	度收益	时延和（ ms）
1	A、C	A	6	3
2	D、E	D	12	11
3	B	C	14	24

表 5-5 基于最优算法求得的升级策略

Table 5-5 The migration strategy based on optimal algorithm

升级步	交换机	控制器	度收益	时延和（ ms）
1	A、C	A	6	3
2	D、E	D	12	11
3	B	C	14	24

5.4.2 基于 DQN 的算法验证

本小节选用 4.3 节中的 49 点真实网络拓扑 UUNET 进行 DQN 算法中模型的训练与验证，说明模型在大规模网络中的可行性。选择训练次数为 3000 次。

在 49 点网络中，利用 DQN 算法对神经网络进行训练，训练过程中损失函数随训练次数的变化曲线如图 5-8 所示。随着训练的进行，损失函数总体呈下降趋势，但也存在部分突变，这是由于训练样本会以ε的概率选择随机动作而非最优动作导致的。

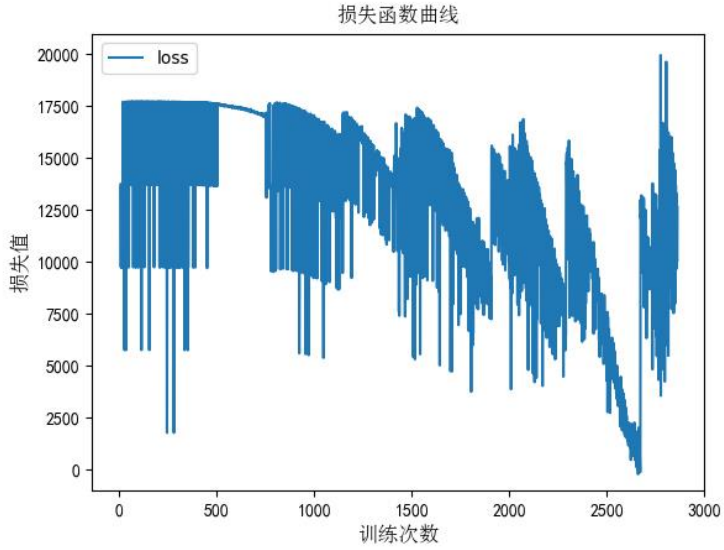
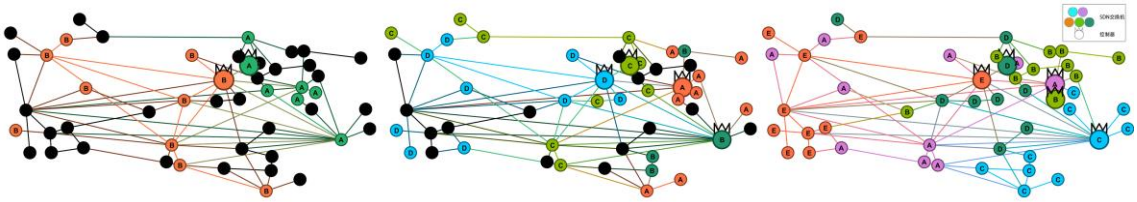


图 5-8 DQN 算法损失函数曲线

Figure 5-8 Loss function curve of DQN algorithm

训练完成后，根据所处状态计算每一动作的价值，并选择执行该状态下的最优动作，直至升级完成，升级过程如图 5-9 所示。最终得到纯 SDN 网络的时延为 507 ms，根据 4.3.1 节计算得到的最优网络时延为 471 ms，误差为 0.076，与 4.3.2 节基于改进遗传算法的启发式算法求解得到的最终纯 SDN 网络时延 552 ms 相比，基于强化学习的结果更接近最优解。



a) 第一步　　b) 第二步　　c) 第三步

a) Step 1　　b) Step 2　　c) Step 3

图 5-9 基于 DQN 算法的 UUNET 逐步升级图

Figure 5-9 UUNET migration process with DQN algorithm

基于强化学习的求解算法（以下简称为“强化学习”）与基于改进遗传算法的启发式算法（以下简称“启发式算法”）求解结果如表 5-6 所示，总计结果为时间积分结果。两种升级算法的度收益与时延损失对比分别如图 5-10 与图 5-11 所示。

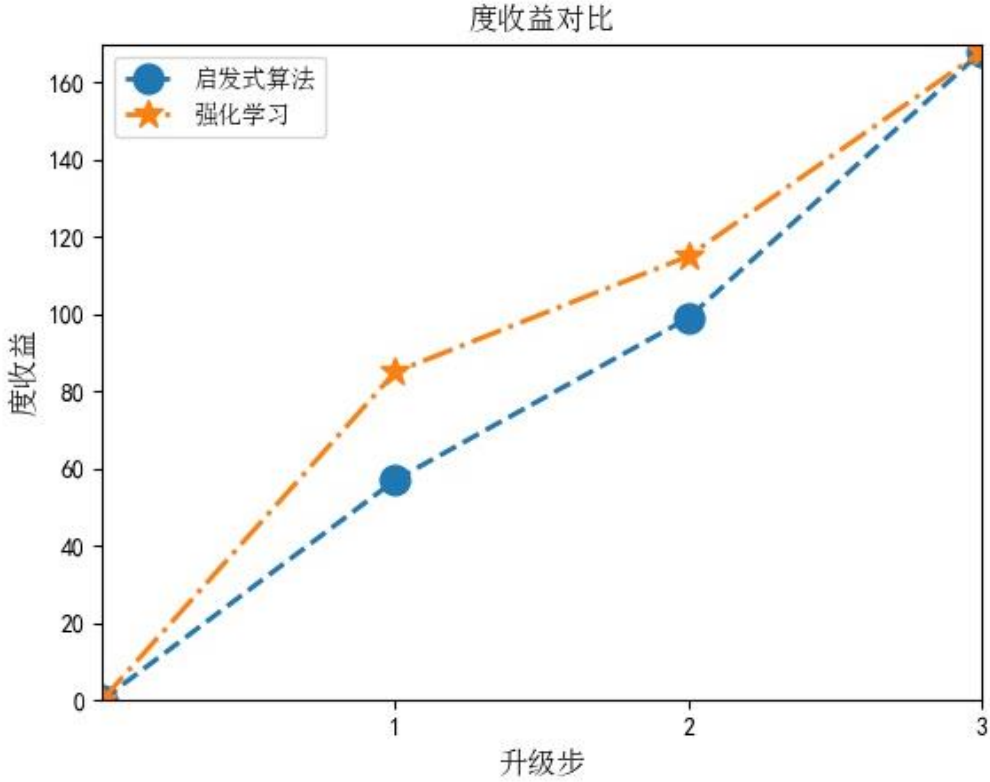


图 5-10 两种升级算法度收益对比

Figure 5-10 Degree gain comparison of the two upgrade algorith ms

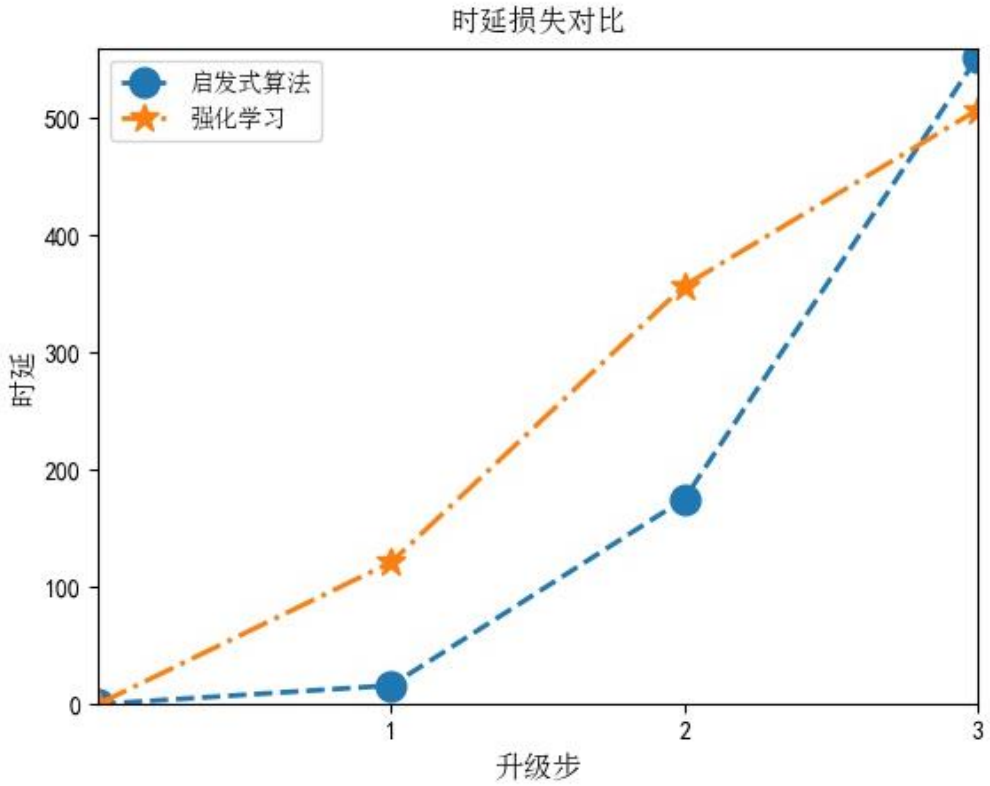


图 5-11 两种升级算法时延损失对比图

Figure 5-11 Delay loss comparison of the two upgrade algorith ms

表 5-6 两种升级算法结果对比

Table 5-6 Comparison of results for two upgrade algorith ms

升级步	度收益			时延损失(ms)		
	全局最优	启发式算法	强化学习	全局最优	启发式算法	强化学习
1	—	57	85	—	16	121
2	—	99	115	—	175	357
3	—	168	168	471	552	507
总计	—	1500	1544	—	4607	4534

分析图 5-10 与图 5-11 可以得出，强化学习在每一步的度收益都优于遗传算法的结果，在时延对比图中，启发式算法在单步升级过程中的时延损失小于强化学习的时延，在最终纯 SDN 网络下时延损失大于强化学习的时延。经过时间积分，最终强化学习得到的结果在度收益与时延损失上均优于启发式算法。

以上网络升级模型的求解验证了基于深度强化学习求解 SDN 网络动态升级策略的可行性，深度强化学习可以求解出近似最优的升级策略，且在 UUNET 网络拓扑中求得的结果优于基于改进遗传算法的启发式算法结果。

5.5 本章小结

本章首先分析了 SDN 最优动态升级问题的特点与强化学习的适用范围与原理，得出 SDN 网络升级过程中需要同时考虑单步收益与最终纯 SDN 网络的全局收益，与强化学习的内涵相一致，均是在一系列随时间进行的动作中，既要考虑当前动作带来的奖励，也要考虑未来收益对当前时间步的影响。因此，本章选择基于强化学习进行 SDN 网络动态升级策略的求解。

首先对 SDN 升级问题进行强化学习建模，定义强化学习模型中的状态、动作以及奖励。强化学习的求解要点在于价值的计算，本章选用 Q-learning 算法与 DQN 算法分别用于小规模网络与大规模网络的价值求解。其中 Q-learning 算法通过迭代更新每一个状态—动作对的 Q 值，得到状态—动作价值表，即 Q-table。DQN 算法利用神经网络拟合函数代替 Q-table，并将训练过程中得到的状态—动作序列存入经验池中用作经验回放，训练模型，训练完成后可以得到价值近似函数。

得到价值函数后，利用贪婪策略在所处状态下选择价值最大的动作进行状态更新，得到完整升级方案。本章在 5 点示例网络中验证了 Q-learning 算法可以达到或近似达到最优解，在 49 点真实网络中验证了 DQN 算法可以近似得到模型的最优解，且在本文所选的验证拓扑中，DQN 算法的结果优于遗传算法所得结果。

6 总结与展望

6.1 本文主要工作

本论文的主要工作包括以下三部分，全局视角下的 SDN 最优动态升级问题分析与建模、基于改进遗传算法的启发式算法、基于强化学习的求解算法。

(1) 全局视角下的 SDN 最优动态升级问题

本文对 SDN 最优动态升级问题进行分析与建模，主要工作包括以下三方面。

1) 对 SDN 网络升级过程中交换机升级与控制器部署问题进行分析。通过理论分析与示例网络说明，将单步升级的最优策略直接应用于升级过程，得到的最终纯 SDN 网络控制器布局并不是最优的控制器布局，论证了本研究的意义与难点。

2) 对整个升级过程进行最优化建模。引入随时间变化的退化因子描述双目标向单目标的转化，将最大化升级带来的度收益与最小化控制器部署引入的控制时延与同步时延作为最优化目标，将相应的升级约束与部署约束作为约束条件得到双目标动态优化模型。这一优化问题无法直接进行求解，通过引入惩罚项和协调因子，将问题转化为整数线性规划问题。

3) 利用商用最优化求解器对整数线性规划问题进行求解，在真实网络拓扑中，与现有升级算法进行对比分析，验证了本文提出的算法同时考虑了局部双目标与全局单目标，权衡了升级过程中混合 SDN 网络的利益与最终纯 SDN 网络的利益，牺牲较小局部收益以获得更大的全局收益。

(2) 基于改进遗传算法的启发式算法

当网络规模较大时，无法直接利用最优化求解器得出升级问题的精确解，本文采用基于改进遗传算法的启发式算法得出大规模网络下的近似解。通过对遗传算法进行改进，以使得产生的子代合格率更高，提高遗传算法求解效率。

1) 针对纯 SDN 网络中最优控制器布局问题，改进遗传算法，得到基于 SDN 匹配关系的遗传算法（SM-GA）。根据最优控制器布局问题的特点，选定匹配关系矩阵为可行解的形式，并约定了可行解需满足的条件，定义交叉为两可行解对应行的交换，变异为某一选定解任意两行的交换或是对应互补列的交换。对真实网络拓扑进行遗传算法求解，对求解结果分析对比可得，遗传算法求得的近似解相较于最优解来说，总时延增加了 5.1%，接近最优解。

2) 针对 SDN 网络最优动态升级问题，改进遗传算法，得到基于互补交叉变异的遗传算法（CCM-GA）。根据 SDN 网络动态升级问题的特点，选定交换机升级行

与控制器部署行的组合作为可行解的形式，并约定了可行解需满足的条件。交叉与变异包括交换机行交叉/变异与控制器行交叉/变异，定义交叉为两解的互补位置进行互换，定义变异为某一选定解任意两互补位置的交换。对真实网络拓扑利用遗传算法求得完整升级策略，最终得到的纯 SDN 网络控制器布局相较于最优解，总时延增加了 17.2%，验证了改进遗传算法的有效性。

(3) 基于强化学习的求解算法

本文针对强化学习与所研究问题的契合度，基于强化学习对 SDN 网络最优动态升级策略进行求解。

1) 搭建 SDN 最优动态升级问题的强化学习模型。将 SDN 最优动态升级问题中选择交换机和部署控制器定义为动作，将网络升级情况定义为状态，将升级带来的度收益与时延损失加权和作为奖励，搭建强化学习环境。

2) 基于 Q-learning 算法与 DQN 算法进行价值计算，并分别在小规模示例网络与大规模真实网络拓扑中进行验证。利用 Q-learning 算法在示例网络中得到的升级策略与最优升级策略一致；利用 DQN 算法在 49 点真实网络中升级得到的最终网络与最优时延相差 7.6%，接近最优解，且相较于遗传算法所得的近似解，利用 DQN 算法在所选拓扑中求得的结果更优。

6.2 未来工作展望

本文的研究工作还可以从以下两方面进行提升。

(1) 本文约定每次升级的交换机个数与部署的控制器个数是固定值，在此约束下得到最优升级策略。在未来研究中，每次升级的个数也可作为变量，得到最优的控制器个数，与满足工程造价约束下所升级的最优交换机个数。

(2) 本文选用度收益与时延损失作为 SDN 网络升级过程中收益的评价指标。在未来研究中，可以对交换机升级带来的收益与控制器部署带来的损失重新定义，如引入负载均衡、可靠性、可编程流等指标，使得问题考虑更加全面。

参考文献

[1] Morocho-Cayamcela M E, Lee H, Lim W. Machine Learning for 5G/B5G Mobile and Wireless Communications: Potential, Limitations, and Future Directions[J]. IEEE Access, 2019, 7(99):137184-137206.

[2] Cao Z, Panwar S S, Kodialam M, et al. Enhancing Mobile Networks With Software Defined Networking and Cloud Computing[J]. IEEE/ACM Transactions on Networking, 2017, 25(3):1431-1444.

[3] Masoudi R, Ghaffari A. Software Defined Networks: A Survey[J]. Journal of Network and computer Applications, 2016, 67: 1-25.

[4] Mao Q, Shen W. A Load Balancing Method Based on SDN[C]. Shenzhen: 2015 Seventh International Conference on Measuring Technology and Mechatronics Automation (ICMTMA). IEEE, 2015: 18-21..

[5] Dan L, Canini M, Schmid S, et al. Panopticon: Reaping the Benefits of Partial SDN Deployment in Enterprise Networks[C]. San Diego: 2014 Annual Technical Conference (USENIX). 2014: 333-345.

[6] Chen M H, Wang W M, Chung I H, et al. Incremental Hybrid SDN Deployment for Enterprise Networks[C]. Orlando: 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech). 2017: 1143-1149

[7] Fakhteh A H, Sattari-Naeini V, Naji H R. Increasing the Network Control Ability and Flexibility In Incremental Switch Deployment for Hybrid Software-Defined Networks[C]. Shanghai: 2019 9th International Conference on Computer and Knowledge Engineering (ICCKE). 2019: 263-268.

[8] Bates A, Butler K, Haeberlen A, et al. Let SDN Be Your Eyes: Secure Forensics in Data Center Networks[C]. San Diego: Workshop on Security of Emerging Networking Technologies. 2014: 1-8.

[9] Barakat O L, Emadinia T, Koll D, et al. Paving the Way towards Enterprise SDN Adoption: New Selection Strategies for Hybrid Networks[C]. Paris: 2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN). 2019: 322-328.

[10] Ze, Yang, Kwan, et al. SDN Candidate Selection in Hybrid IP/SDN Networks for Single Link Failure Protection[J]. IEEE/ACM Transactions on Networking, 2020, 28(1):312-321.

[11] Jain S, Kumar A, Mandal S, et al. B4: Experience with a Globally-deployed Software Defined WAN[J]. ACM SIGCOMM Computer Communication Review, 2013, 43(4): 3-14.

[12] As Providers Push NFV/SDN, 3 Key Issues Remain. Accessed: Apr. 23, 2017. [OL]. Available:https://www.rcrwireless.com/20170423/opinion/readerforum/reader-forum-nfv-sdn-issues-tag10.

[13] Poularakis K, Iosifidis G, Smaragdakis G, et al. Optimizing Gradual SDN Upgrades in ISP Networks [J]. IEEE/ACM transactions on networking, 2019, 27(1): 288-301.

[14] Benamrane F, Mamoun M B, Benaini R. Short: A Case Study of the Performance of an OpenFlow Controller[C] Berlin: International Conference on Networked Syste ms. 2014: 330-334.

[15] Das T, Caria M, Jukan A, et al. Insights on SDN migration trajectory[C]. Netherlands: 2015 IEEE International Conference on Communications (ICC). IEEE, 2015: 5348-5353.

[16] Guo Z, Chen W, Liu Y F, et al. Joint Switch Upgrade and Controller Deployment in Hybrid Software-Defined Networks [J]. IEEE Journal on Selected Areas in Communications, 2019, 37(5): 1012-1028.

[17] Sinha Y, Haribabu K. A Survey: Hybrid SDN[J]. Journal of Network and Computer Applications, 2017, 100: 35-55.

[18] McKeown N. Software-Defined networking[C]. Rio de Janeiro: International Conference on Computer Communications (INFOCOM) 2009, 17(2): 30-32.

[19] 陆骏. 基于软件定义网络的多控制器部署问题研究[D]. 大连:大连理工大学, 2015.

[20] 黄亭. 基于可靠性的 SDN 控制器部署研究[D]. 长沙:长沙理工大学, 2018.

[21] Ren W, Sun Y, Luo H, et al. A Novel Control Plane Optimization Strategy for Important Nodes in SDN-IoT Networks[J]. IEEE Internet of Things Journal, 2018, 6(2): 3558-3571.

[22] Heller B, Sherwood R, McKeown N. The Controller Placement Problem[J]. ACM SIGCOMM Computer Communication Review, 2012, 42(4): 473-478.

[23] Rahman S, Kim G H, Cho Y Z, et al. Deployment of an SDN-based UAV network: Controller Placement and Tradeoff Between Control Overhead and Delay[C]. Chengdu:2017 International Conference on Information and Communication Technology Convergence (ICTC). IEEE, 2017: 1290-1292.

[24] Tao P, Ying C, Sun Z, et al. The controller placement of software-defined networks based on minimum delay and load balancing[C]. Athens: 2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech). IEEE, 2018: 310-313.

[25] Ksentini A, Bagaa M, Taleb T. On Using SDN in 5G: The Controller Placement Problem [C]. Washington:2016 IEEE Global Communications Conference (GLOBECOM). IEEE, 2016: 1-6.

[26] Lei Z, Rong C, Chen Q. Control Plane Delay Minimization based SDN Controller Placement Scheme[C]. Nanjing: 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP). 2017: 1-6.

[27] Yang S, Cui L, Chen Z, et al. An Efficient Approach to Robust SDN Controller Placement for Security[J]. IEEE Transactions on Network and Service Management, 2020, 17(3): 1669-1682.

[28] Zhong Q, Wang Y, Li W, et al. A Min-cover based Controller Placement Approach to Build Reliable Control Network in SDN[C]. Istanbul: NO MS 2016-2016 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2016: 481-487.

[29] 邓春雪. 软件定义网络中多控制器部署策略的研究[D]. 北京:北京邮电大学, 2017.

[30] 杨耀通. 基于 SDN 的控制器部署问题研究[D]. 天津:天津大学, 2018.

[31] Naning H S, Munadi R, Effendy M Z. SDN Controller Placement Design: For Large Scale Production Network[C]. Bandung: 2016 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob). IEEE, 2016: 74-79.

[32] Wang G, Zhao Y, Huang J, et al. A K-means-based Network Partition Algorithm for

Controller Placement in Software Defined Network[C]. New York: 2016 IEEE International Conference on Communications (ICC). IEEE, 2016: 1-6.

[33] Yuan T, Huang X, Ma M, et al. Balance-based SDN Controller Placement and Assignment with Minimum Weight Matching[C]. Kansas City: 2018 IEEE International Conference on Communications (ICC). IEEE, 2018: 1-6.

[34] Killi B P R, Reddy E A, Rao S V. Cooperative Game Theory based Network Partitioning for Controller Placement in SDN[C]. Bengaluru: 2018 10th International Conference on Communication Syste ms & Networks (CO MSNETS). IEEE, 2018: 105-112.

[35] Bai Y, Wu M, Shen J, et al. Optimization Strategy of SDN Control Deployment Based on Simulated Annealing-Genetic Hybrid Algorithm[C]. Chengdu: 2018 IEEE 4th International Conference on Computer and Communications (ICCC). IEEE, 2018: 2238-2242.

[36] 朱磊. 软件定义网络控制器部署算法研究[D]. 重庆:重庆邮电大学, 2018.

[37] 候肖兰. 软件定义网络中可扩展性相关问题的研究[D]. 北京:北京邮电大学,2020.

[38] Liao L, Leung V C M. Genetic Algorith ms with Particle Swarm Optimization based Mutation for Distributed Controller Placement in SDNs[C]. Berlin: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). IEEE, 2017: 1-6.

[39] Han B, Yang X, Wang X. Dynamic Controller-Switch Mapping Assignment with Genetic Algorithm for Multi-controller SDN[C]. Berlin: 2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC /PiCom /CBDCom /CyberSciTech). IEEE, 2019: 980-986.

[40] Khorsandroo S, Sánchez A G, Tosun A S, et al. Hybrid SDN Evolution: A Comprehensive Survey of the State-of-the-Art[J]. Computer Networks, 2021: 107981.

[41] Hong D K, Ma Y, Banerjee S, et al. Incremental Deployment of SDN in Hybrid Enterprise and ISP Networks[C]. Berlin: Proceedings of the Symposium on SDN Research. 2016: 1-7.

[42] Chu C Y, Xi K, Luo M, et al. Congestion-aware Single Link Failure Recovery in Hybrid SDN Networks[C]. Hong Kong: 2015 IEEE Conference on Computer Communications (INFOCOM). IEEE, 2015: 1086-1094.

[43] Jia X, Jiang Y, Guo Z, et al. Intelligent Path Control for Energy-saving in Hybrid SDN Networks[J]. Computer networks, 2017, 131: 65-76.

[44] Wang H, Li Y, Jin D, et al. Saving Energy in Partially Deployed Software Defined Networks[J]. IEEE Transactions on Computers, 2015, 65(5): 1578-1592.

[45] Feng W, Guo Z, Liu C, et al. BAGUETTE: Towards a Secure and Cost-effective Switch Upgrade in Hybrid Software-Defined Networks[C]. Dublin: ICC 2020-2020 IEEE International Conference on Communications (ICC). IEEE, 2020: 1-6.

[46] Caria M, Jukan A, Hoffmann M. A Performance Study of Network Migration to SDN-enabled Traffic Engineering[C]. Austin: 2014 IEEE Global Communications Conference (GLOBECOM). IEEE, 2014: 1391-1396.

[47] Guo Y, Wang Z, Yin X, et al. Incremental Deployment for Traffic Engineering in Hybrid SDN Network[C]. Nanjing: 2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC). IEEE, 2015: 1-8.

[48] Kelkawi A, Mohammed A, Alyatama A. Incremental Deployment of Hybrid IP/SDN

Network with Optimized Traffic Engineering[C]. Online: 2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). IEEE, 2020: 57-63.

[49] Yu Z, Chen X, Li X. A Dynamic Programming Approach for Generalized Nearly Isotonic Optimization[J]. arXiv preprint arXiv:2011.03305, 2020.

[50] 哈姆迪·A·塔哈, 刘德刚. 运筹学导论[M]. 北京:中国人民大学出版社, 2014.

[51] Kuhn H W. The Hungarian Method for the Assignment Problem[J]. Naval research logistics quarterly, 2010, 52(1-2): 7-21.

[52]Kazarlis S A, Bakirtzis A G. A genetic algorithm solution to the unit commitment problem[J]. Power Systems IEEE Transactions on, 1996, 11(1):83-92.

[53] Vikhar P A. Evolutionary Algorithms: A Critical Review and its Future Prospects[C]. India: 2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC). IEEE, 2016: 261-265.

[54] D Hu. An Introduction to Markov Process in Random Environment[J]. Acta Mathematica Scientia, 2010, 230(30):xiv,171.

[55] Barron E N, Ishii H. The Bellman equation for minimizing the maximum cost[J]. Nonlinear Analysis Theory Methods & Applications, 1989, 13(9):1067-1090.

[56] Sutton R S, Barto A G. Reinforcement Learning: An Introduction[M]. MIT press, 2018.

[57] Watkins C J C H, Dayan P. Q-learning[J]. Machine Learning, 1992, 8(3-4): 279-292.

[58]Mnih V, Kavukcuoglu K, Silver D, et al. Playing Atari with Deep Reinforcement Learning[J]. Computer Science, 2013.

[59] Knight S, Nguyen H X, Falkner N, et al. The Internet Topology Zoo[J]. IEEE Journal on Selected Areas in Communications, 2011, 29(9): 1765-1775.

作者简历及攻读硕士学位期间取得的研究成果

一、作者简历

李小乐，女，1995 年 11 月生；

2014 年 9 月至 2018 年 6 月，就读于北京交通大学通信工程（理科试验班）专业，取得工学学士学位；

2018 年 8 月至 2021 年 6 月，就读于北京交通大学通信与信息系统专业，取得工学硕士学位。

二、发表论文

[1] Li X, Zheng H, Guo Y. Migrating to SDN for Mobile Core Networks: A Dynamic and Global Perspective[C]. Online: IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2020.

三、参与科研项目

[1] 基于人工智能的网络流量预测与资源管理技术

.

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作和取得的研究成果，除了文中特别加以标注和致谢之处外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京交通大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

学位论文作者签名：　　签字日期：　　年　　月　　日

学位论文数据集

表 1.1： 数据集页

关键词*	密级*	中图分类号	UDC	论文资助
混合 SDN；动态升级；最优化；控制器联合部署；改进遗传算法；强化学习	公开			
学位授予单位名称*		学位授予单位代码*	学位类别*	学位级别*
北京交通大学		10004	工学	硕士
论文题名*		并列题名		论文语种*
全局视角下的 SDN 动态升级策略研究				中文
作者姓名*	李小乐		学号*	18120101
培养单位名称*		培养单位代码*	培养单位地址	邮编
北京交通大学		10004	北京市海淀区西直门外上园村 3 号	100044
学科专业*		研究方向*	学制*	学位授予年*
通信与信息系统		信息网络	3	2021
论文提交日期*	2021.6.1			
导师姓名*	郑宏云		职称*	副教授
评阅人	答辩委员会主席*		答辩委员会成员	
	郭宇春			
电子版论文提交格式 文本（） 图像（ ） 视频（ ） 音频（ ） 多媒体（ ） 其他（ ） 推荐格式：application/ msword；application/pdf				
电子版论文出版（发布）者		电子版论文出版（发布）地		权限声明
论文总页数*	83			
共 33 项，其中带*为必填数据，为 21 项。				