TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Giải thuật tiến hóa đa mục tiêu giải bài toán định tuyến trong mạng ảo hóa

Đỗ Đức Anh
anh.dd193976@sis.hust.edu.vn

Ngành: Khoa học máy tính

Giảng viên hướng dẫn: PGS.TS Huỳnh Thị Thanh Bình _____

Chữ kí GVHD

Khoa: Khoa học máy tính

Trường: Công nghệ thông tin và Truyền thông

HÀ NỘI, 06/2022

LỜI CẢM ƠN

Trong quá trình học tập tại Đại học Bách Khoa Hà Nội, ngoài những nỗ lực cá nhân, việc hoàn thành đồ án một cách toàn diện của tôi cũng thể thiếu sự hỗ trợ, hướng dẫn và sự giúp đỡ vô giá của các thầy cô giáo. Vì vậy, tôi muốn bày tỏ lòng biết ơn chân thành đến toàn bộ giáo viên tại Đại học Bách Khoa Hà Nội, đặc biệt là các giảng viên thuộc Trường Công nghệ thông tin và Truyền thông, đã truyền đạt kiến thức quý báu thông qua các bài giảng và giúp tôi hoàn thành thành công luận văn tốt nghiệp.

Đặc biệt, tôi muốn bày tỏ lòng biết ơn sâu sắc đến PGS.TS Huỳnh Thị Thanh Bình, một giảng viên tận tâm tại Đại học Bách Khoa Hà Nội, vì sự chuyên nghiệp trong việc hướng dẫn, chỉ dẫn và đóng góp ý kiến quý báu trong suốt hành trình hoàn thành luận văn tốt nghiệp này.

Hơn nữa, tôi muốn bày tỏ lòng biết ơn chân thành đến các anh, chị, em, bạn ở trong Phòng nghiên cứu Mô hình hóa, mô phỏng và tối ưu hóa (MSO Lab), mọi người đã tích cực lắng nghe và chia sẻ ý kiến và quan điểm chân thành của mình. Đặc biệt, tôi xin được chân thành cảm ơn T.S Nguyễn Thị Tâm, anh Trần Công Đạo, anh Trần Huy Hùng đã luôn nhiệt tình chỉ bảo và giúp đỡ tôi trong suốt quá trình học tập và nghiên cứu.

Cuối cùng, tôi xin gửi lời cảm ơn đặc biệt đến bố mẹ, người thân và bạn bè đã luôn động viên và hỗ trợ liên tục trong quá trình nghiên cứu và hoàn thành luận văn tốt nghiệp này. Mong rằng mọi người luôn có sức khỏe dồi dào, may mắn và thành công.

Tôi nhận thức rằng luận văn tốt nghiệp này còn một số hạn chế cần khắc phục, chủ yếu do thời gian và nguồn lực hạn chế. Do đó, tôi mong mỏi nhận được những ý kiến và phản hồi từ các giảng viên nhằm nâng cao chất lượng và hoàn thiện luận văn này.

TÓM TẮT NỘI DUNG ĐỒ ÁN

Ảo hóa chức năng mạng (Network Function Virtualization - NFV) đang nhận được sự quan tâm của các nhà nghiên cứu trong thời gian gần đây vì nó là một giải pháp hiệu quả cho phép thay thế các phần cứng chuyên dụng phức tạp bằng các phần mềm chạy trong môi trường ảo hóa. NFV có nhiều lợi ích như tính linh hoạt, hiệu suất, khả năng mở rộng, chu kỳ triển khai ngắn, và nâng cấp dịch vụ. Những lợi ích này đã đẩy mạnh sự phát triển của NFV trong kiến trúc mạng, mở ra cánh cửa cho việc cung cấp dịch vụ mạng sáng tạo và hiệu quả trong các lĩnh vực khác nhau. Trong NFV, dịch vụ được yêu cầu được thực hiện thông qua một chuỗi các chức năng mạng ảo (Virtual Network Function - VNF). Những chức năng mạng ảo này có thể chạy trên máy chủ thông thường nhờ sử dụng công nghệ ảo hóa. Các VNF này được sắp xếp theo một thứ tự đã định trước qua đó dữ liệu được truyền qua, cũng được biết đến với tên gọi chuỗi dịch vụ (Servive Funciton Chain - SFC).

Phân bổ tài nguyên trong NFV là một vấn đề phức tạp khi kích thước mạng và số lượng yêu cầu dịch vụ ngày tăng lên. Việc phân bổ tài nguyên hợp lý có thể làm nâng cao chất lượng dịch vụ của mạng. Bài toán phân bổ tài nguyên bao gồm bài toán đặt các VNF và định tuyến các chuỗi dịch vụ và các biến thể của bài toán này. Trong nghiên cứu này, đồ án tập trung vào giải quyết bài toán định tuyến cho các chuỗi dịch vụ để tối ưu hóa hai mục tiêu: i) tối đa số lượng chuỗi dịch vụ được đáp ứng; ii) tối đa việc sử dụng tài nguyên trong mạng. Để giải quyết bài toán tối ưu đa mục tiêu này, đồ án nghiên cứu các giải thuật tiến hóa đa mục tiêu để tìm được tập các lời giải không bị trội lẫn nhau. Kết quả thực nghiệm được tiến hành trên nhiều bộ dữ liệu khác nhau cho thấy sự hiệu quả của các tiếp cận dựa trên giải thuật tiến hóa đa mục tiêu so với cách tiếp cận đơn mục tiêu để giải quyết bài toán.

Sinh viên thực hiện
(Ký và ghi rõ họ tên)

ABSTRACT

Network Function Virtualization (NFV) has been attracting the attention of researchers recently as it is an efficient solution that replaces complex dedicated hardware with software running in a virtualized environment. NFV offers numerous benefits, such as flexibility, performance, scalability, short deployment cycles, and service upgrades. These advantages have propelled the development of NFV in network architectures, opening doors for innovative and efficient network services in various fields. In NFV, the requested service is implemented through a sequence of Virtual Network Functions (VNFs) that can run on regular servers using virtualization technology. These VNFs are arranged in a predefined order through which data flows, known as Service Function Chaining (SFC).

Resource allocation in NFV becomes complex as the network size and the number of service requests increase. Proper resource allocation can enhance the quality of network services. The resource allocation problem includes placing VNFs and routing service chains and its variants. This study focuses on solving the routing problem for service chains to optimize two objectives: i) maximizing the number of satisfied service chains, and ii) maximizing the utilization of resources in the network. To tackle this multi-objective optimization problem, the study applies multi-objective evolutionary algorithms to find a set of non-dominated solutions. Experimental results conducted on various datasets demonstrate the effectiveness of the multi-objective evolutionary algorithm approaches compared to single-objective approaches in addressing the problem.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Từ viết tắt	**Viết đầy đủ**	**Ý nghĩa**
GA	Genetic Algorithm	Giải thuật di truyền
MOTLBO	Multi-Objective Teaching-Learning-Based Optimization	Tiến hóa đa mục tiêu dựa trên cơ chế dạy học
MOEA/D	Multi-objective Evolutionary Algorithm based on Decomposition	Thuật toán tiến hóa đa mục tiêu dựa trên phân rã
MOEAs	Multi-objective Evolutionary Algorithms	Các thuật toán tiến hóa đa mục tiêu
MOP	Multi-objective Optimization Problem	Bài toán tối ưu hóa đa mục tiêu
NFV	Network Function Virtualization	Ảo hóa chức năng mạng
NSGA2	Non-dominated Sorting Genetic Algorithm II	Thuật toán di truyền sắp xếp không trội
SPEA2	Pareto 2 Strength Pareto Evolutionary Algorithm	Thuật toán tiến hóa dựa trên tiềm năng biên
SFC	Service Function Chain	Chuỗi dịch vụ
VNF	Virtual Network Function	Chức năng mạng ảo

LỜI MỞ ĐẦU

Trong vòng ba thập kỷ vừa qua, ngành công nghệ thông tin ngày càng phát triển, đặc biệt là các lĩnh vực về mạng. Những phát triển này không chỉ diễn ra trong khả năng tính toán, mà còn trong khả năng truyền tải và lưu trữ dữ liệu. Sự kết hợp giữa việc kết nối các trung tâm dữ liệu quy mô lớn và mô hình tính toán đám mây đã mang lại nhiều lợi ích. Các tổ chức có thể chia sẻ và truy cập dữ liệu từ nhiều nguồn khác nhau thông qua mạng kết nối đám mây, đồng thời tận dụng khả năng mở rộng linh hoạt của đám mây để đáp ứng nhu cầu tính toán biến đổi. Điều này mang lại hiệu quả cao hơn, tiết kiệm chi phí và tăng tính linh hoạt trong quản lý dữ liệu và công việc tính toán. Tuy nhiên, khi những xu hướng này diễn ra, cũng có một loạt các thách thức về quản lý và điều phối đã xuất hiện. Các thách thức này gây ra những khó khăn và làm phức tạp hóa trong việc quản lý và phối hợp hiệu quả các hệ thống tiên tiến này.

Cho đến gần đây, các thiết bị mạng vật lý truyền thống, ví dụ như bộ chuyển mạch (switch) và bộ định tuyến (router), thường được vận hành với các hệ thống cấu hình điều khiển độc lập và nội bộ chuyên dụng; tức là, độc quyền của nhà cung cấp. Các thiết lập này dẫn đến tăng độ phức tạp trong quản lý do các chuỗi dịch vụ có cấu hình riêng lẻ và mức độ phụ thuộc cao vào nhà cung cấp. Do đó, nhiều nhà cung cấp dịch vụ đã gặp khó khăn khi xây dựng và cung cấp các máy chủ bằng các thiết bị vật lý thông thường. Để khắc phục những khó khăn trên, các tổ chức khác nhau đã bắt đầu nỗ lực đơn giản hóa thiết kế nút mạng bằng cách tách mặt phẳng điều khiển (control plane) và mặt phẳng dữ liệu (data plane) hay được gọi là mạng do phần mềm xác định (software-defined networking - SDN). Mục tiêu chính của SDN là cải thiện khả năng kiểm soát mạng bằng cách cho phép các doanh nghiệp và nhà cung cấp dịch vụ đáp ứng nhanh chóng khi các nhu cầu kinh doanh thay đổi. Trong mạng SDN, tất cả các thiết bị mạng trong 1 hệ thống được điều khiển thông qua một giao diện điều khiển quản lý tập trung (Centralized Control Controller) là các phần mềm chạy trên máy chủ và tách biệt hẳn với thiết bị mạng. Tổng thể, SDN cố gắng giảm chi phí hoạt động (OpEx) và vốn (CapEx) bằng cách giảm độ phức tạp của việc quản quản lý và tăng tính linh động. Ngoài ra, việc áp dụng các nút SDN thúc đẩy tính độc lập với nhà cung cấp.

Mặc dù việc tách mặt phẳng điều khiển và mặt phẳng dữ liệu mang lại những lợi thế đã nêu trên, việc triển khai các dịch vụ mạng truyền thống vẫn còn phức tạp và tốn kém. Hơn nữa, phần lớn các dịch vụ mạng truyền thống được quản lý và triển khai bởi các hệ thống nhúng độc quyền của nhà cung cấp. Vì vậy, ảo hóa chức năng

mạng (Network Function Virtualization - NFV), lần đầu tiên được đề xuất vào năm 2012, là một sự thay thế tiên tiến cho các mô hình mạng. NFV là một phiên bản cải tiến hơn của các hộp trung gian dựa trên phần cứng (hardware-based middleboxes) thông thường để giảm độ phức tạp trong quản lý dịch vụ mạng và cải thiện khả năng (tái) triển khai. Trong NFV, phần mềm quản lý mạng hoạt động trong một môi trường ảo hóa thay vì bị hạn chế bởi phần cứng. Nhờ công nghệ ảo hóa, mỗi dịch vụ được triển khai dưới dạng một loạt chức năng mạng ảo (Virtual Network Function - VNF) đảm nhận vai trò như phần cứng chuyên biệt và có thể hoạt động trên bất kỳ máy chủ tiêu chuẩn nào. Dữ liệu phải được truyền tải trên một luồng dữ liệu xác định, đi qua một chuỗi các VNF theo đúng trình tự yêu cầu, gọi chuỗi dịch vụ chức năng (Service Function Chain - SFC). Không giống như các mạng thông thường, các mạng hỗ trợ NFV mang lại nhiều lợi thế, bao gồm định tuyến chuỗi dịch vụ linh hoạt, chi phí triển khai chuỗi dịch vụ hợp lý, tính mở rộng cao và quản lý tài nguyên hiệu quả. Kiến trúc cơ sở này đã trở thành một tiêu chuẩn phổ biến trong các ứng dụng trong thế giới thực khác nhau như ảo hóa các trạm cơ sở di động, nền tảng như một dịch vụ (PaaS), mạng phân phối nội dung (CDN), truy cập cố định và môi trường gia đình.

Mặc dù NFV đem lại nhiều lợi ích như trên, một trong những thách thức chính là phân bổ tài nguyên mạng để thiết lập các SFC cần thiết một cách hiệu quả. Phân bổ tài nguyên trong NFV là một vấn đề tối ưu hóa phức tạp với một số biến thể với các mục tiêu khác nhau. Bài toán phân bổ tài nguyên bao gồm 2 vấn đề chính: đặt các VNF được yêu cầu của mỗi người dùng trên mạng và định tuyến cho các chuỗi dịch vụ SFC để không vi phạm các ràng buộc. Đồ án này tập trung nghiên cứu bài toán định tuyến trong mạng ảo hóa theo hướng tiếp cận tiến hóa đa mục tiêu.

Các đóng góp chính của đồ án bao gồm:

- Đồ án nghiên cứu bài toán định tuyến trong mạng ảo hóa dưới dạng bài toán tối ưu hóa đa mục tiêu Multi-objective Optimization Problem (MOP) có tên gọi là MO-TRR. Mục tiêu đầu tiên là tối đa hóa số lượng chuỗi dịch vụ được đáp ứng. Mục tiêu thứ hai là tối thiểu lượng tài nguyên mạng được sử dụng. Hai mục tiêu này xung đột lẫn nhau và cần sự cân bằng hợp lý trong lời giải.
- Áp dụng bốn giải thuật tiến hóa đa mục tiêu khác nhau để giải bài toán MO-TRR bao gồm: thuật toán di truyền sắp xếp không trội II (Non-dominated Sorting Genetic Algorithm II - NSGA-II) [1], thuật toán tiến hóa dựa trên tiềm năng biên (Pareto 2 Strength Pareto Evolutionary Algorithm - SPEA2)[2], thuật toán tiến hóa đa mục tiêu dựa trên phân rã (Multi-objective Evolutionary Algorithm based on Decomposition - MOEA/D) [3] và tiến hóa đa mục

tiêu dựa trên cơ chế dạy học (Multi-Objective Teaching-Learning-Based Optimization - MOTLBO)[4].

- Để đánh giá hiệu quả của thuật toán nghiên cứu. Đồ án tiến hành thực nghiệm trên các kịch bản dữ liệu khác nhau, sử dụng các độ đo đa mục tiêu phổ biến.

Từ những mục tiêu nghiên cứu trên, ngoài phần Mở đầu và Kết luận, nội dung của đồ án bao gồm các chương như sau:

Chương 1 trình bày về cơ sở lý thuyết bao gồm kiến thức tổng quan về bài toán phân bổ tài nguyên trong mạng ảo hóa, bài toán tối ưu và các giải thuật tiến hóa. Ngoài ra chương này cũng giới thiệu qua một số các tiếp cận các bài toán đa mục tiêu.

Chương 2 giới thiệu về các nghiên cứu liên quan và mô hình cụ thể của bài toán định tuyến định tuyến trong mạng ảo hóa.

Chương 3 trình bày đóng góp chính của đồ án bao gồm các giải thuật đa mục tiêu và mô tả chi tiết các thành phần của thuật toán.

Chương 4 trình bày các so sánh, đánh giá kết quả thực nghiệm nhằm đánh giá hiệu quả của các thuật toán tối ưu đa mục tiêu và so sánh với các giải thuật mới nhất hiện nay.

CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

Chương này trình bày các cơ sở lý thuyết về ảo hóa chức năng mạng, bài toán tối ưu bao gồm tối ưu đơn mục tiêu và tối ưu đa mục tiêu. Sau đó, tổng quan về giải thuật tiến hóa và các thành phần của giải thuật di truyền để giải quyết bài toán tối ưu sẽ được trình bày. Phần cuối chương sẽ thảo luận các phương pháp để giải bài toán tối ưu đa mục tiêu.

1.1 Ảo hóa chức năng mạng

Khái niệm về ảo hóa chức năng mạng (Network Function Virtualization - NFV) ra đời vào tháng 10 năm 2012 khi một số nhà cung cấp dịch vụ viễn thông hàng đầu thế giới đã cùng viết một sách trắng (white paper) kêu gọi các hoạt động công nghiệp và nghiên cứu. Vào tháng 11 năm 2012, bảy nhà điều hành này (AT&T, BT, Deutsche Telekom, Orange, Telecom Italia, Telefonica và Verizon) đã chọn Viện Tiêu chuẩn Viễn thông Châu Âu (ETSI) để trở thành nơi tổ chức Nhóm Quy định Công nghiệp cho NFV (ETSI ISG NFV). Hiện nay, với độ phức tạp tương đối cao của mô hình NFV, ISG-NFV cũng đã định nghĩa một số nhóm làm việc (working groups- WG) để xử lý các lĩnh vực cụ thể của việc đặc tả NFV.

- Giao diện và Kiến trúc (Interfaces and Architecture - IFA): Đặc tả kiến trúc tổng thể của NFV và yêu cầu về giao diện để hỗ trợ khả năng tương thích.
- Tiến hóa và Hệ sinh thái (Evolution and Ecosystem -EVE): Chịu trách nhiệm đặc tả giao diện giữa NFV và các tiêu chuẩn hoặc công nghệ khác có liên quan đến NFV. Một ví dụ, WG này đã định nghĩa các yêu cầu và trường hợp sử dụng tương tác NFV-SDN khác nhau.
- Giải pháp (Solutions - SOL): Tập trung vào đặc tả và phát triển các giao thức để hỗ trợ tương thích NFV.
- Độ tin cậy (Reliability - REL): Tập trung vào cải thiện tính tin cậy và khả dụng của hệ thống NFV.
- Tính bảo mật (Security - SEC): Điều chỉnh các vấn đề liên quan đến bảo mật cho NFV, ví dụ như bảo mật triển khai.
- Kiểm thử và cài đặt (Testing and Implementation - TST): Tập trung vào triển khai các bài kiểm tra NFV để kiểm thử và chứng minh khả năng của công nghệ này. Do đó, WG này rất quan trọng trong việc thúc đẩy sự áp dụng NFV của các nhà điều hành.

Do đó, việc xác định các khía cạnh cụ thể của NFV được xử lý bởi các WG và

xác định những chủ đề có quan tâm cao đối với người dùng là rất quan trọng. Dưới đây là mô tả tóm tắt các chủ đề đó:

- Kiến trúc: Phân tích và đề xuất cải tiến cho kiến trúc NFV.
- MANO: Tập trung vào yêu cầu và quản lý tài nguyên vật lý và ảo, tương quan các chức năng mạng, kiểm soát vòng đời, thực hiện và quản lý chính sách.
- Các trường hợp sử dụng: Phát triển các trường hợp triển khai NFV tiên tiến.
- Định vị: Giải quyết vấn đề định vị VNF bằng cách phân tích và/hoặc đề xuất các mô hình định vị VNF mới.

Các nhà nghiên cứu đã đưa ra nhiều chủ đề nghiên cứu khác nhau trong mạng NFV. Trong nghiên cứu [5], [6], các tác giả cung cấp một cái nhìn tổng quan về NFV bao gồm: kiến trúc và triển khai NFV, phân bổ tài nguyên trong NFV, và ứng dụng của NFV trong các lĩnh vực khác nhau như điện toán đám mây (Cloud Computing) hay mạng do phần mềm xác định (Software Defined Networking). Trong nghiên cứu này, đồ án tập trung vào các nghiên cứu liên quan đến bài toán phân bổ tài nguyên trong mạng NFV.

Trong mạng ảo hóa, mỗi yêu cầu được biểu diễn bởi một trình tự các chức năng mạng ảo (Virtual Network Functions - VNF) có thể chạy trên các máy chủ chung bằng cách tận dụng công nghệ ảo hóa. Các VNFs này được sắp xếp với một thứ tự xác định dựa trên luồng dữ liệu, nó còn được gọi là chuỗi dịch vụ (Service Function Chaining - SFC). Tận dụng công nghệ ảo hóa, SFC có thể được thiết lập bằng cách đặt các VNF đã được yêu cầu trên mạng một cách hiệu quả và nhanh chóng. Hơn nữa, một hoặc nhiều VNF có thể thêm hoặc xóa linh hoạt với chi phí rất nhỏ và hiệu quả cao trong trường hợp các SFC bị thay đổi. Ảo hóa chức năng mạng cho phép phân bổ tài nguyên mạng trong một cách linh hoạt hơn, cung cấp một cơ chế quản lý và vận hành hiệu quả, do đó có thể giảm chi phí hoạt động (OpEx) và vốn (CapEx) cho các nhà cung cấp dịch vụ mạng.

Đi kèm với những lợi ích mà ảo hóa chức năng mạng mang lại, một câu hỏi quan trọng là làm cách nào chúng ta có thể sử dụng tài nguyên mạng (năng lượng, băng thông, độ trễ, ...) một cách hiệu quả để có thể đáp ứng các SFCs. Bài toán phân bổ tài nguyên mạng bao gồm bài toán ngoại tuyến và bài toán trực tuyến. Thường thì trong hệ thống cung cấp dịch vụ ngoại tuyến (offline provisioning), tập yêu cầu R được cung cấp trước và mục tiêu là thiết kế giải pháp tối ưu dựa trên R. Ngược lại, trong hệ thống cung cấp dịch vụ trực tuyến (online provisioning), giả định là các yêu cầu trong R đến mạng theo cách thức trực tuyến, tức là phải đáp ứng từng yêu cầu, vì không biết được các yêu cầu tương lai sẽ đến như thế nào. Theo nghiên cứu

[7], trường hợp tổng quát của bài toán ngoại tuyến có thể được mô hình hóa thành bài toán đặt và định tuyến (VNF Placement and Traffic Routing - VPTR). Trong bài toán VPTR, một mạng máy tính được biểu diễn dưới dạng đồ thị các nút mạng (máy chủ, máy tính, bộ định tuyến, switch, ...) và các kết nối giữa chúng. Các nút mạng có thể có các tài nguyên CPU hoặc bộ nhớ giới hạn, trong khi các kết nối có khả năng truyền tải băng thông hữu hạn. Mỗi yêu cầu gửi đến mạng phải được xử lý bởi một chuỗi SFC hoàn chỉnh. Mục tiêu của VPTR là triển khai các VNF tại các nút mạng phù hợp và xây dựng các SFC cho các yêu cầu sao cho yêu cầu chuỗi VNF được đáp ứng và không vi phạm ràng buộc về khả năng.

Trong nghiên cứu [8], các tác giả đã chứng minh bài toán VPTR là bài toán NP-khó. Họ đề xuất mô hình quy hoạch nguyên tuyến tính (Integer Linear Programming - ILP) để tìm lời giải chính xác. Các tác giả trình bày một mô hình tối ưu hóa chứa mạng cơ sở, yêu cầu lưu lượng và VNFs trong nghiên cứu [9]. Mỗi yêu cầu có một số lượng đường đi khả thi, và ILP có thể chọn con đường tốt nhất. Trong công bố [10], tác giả tập trung vào việc truyền tải VNF và xử lý độ trễ. Họ sử dụng quy hoạch tuyến tính nguyên hỗn hợp để biểu diễn kết hợp bài toán lập lịch VNF và định tuyến lưu lượng. Một số nghiên cứu khác cố gắng đơn giản hóa VPTR hoặc áp dụng các phương pháp heuristics để đạt được các giải pháp gần tối ưu. Hai biến thể đáng chú ý nhằm đơn giản hóa bài toán VPTR bao gồm bài toán đặt VNF (VNF Placement - VNFP) và định tuyến chuỗi dịch vụ (Traffic Routing - TRR). VNFP cung cấp các đường đi đã được xác định trước cho tất cả các yêu cầu, và chỉ cần đặt VNF tại các nút mạng phù hợp [11]. Ngược lại, các VNF trong các bài toán TRR đã được triển khai sẵn, và cần phải tìm các đường đi phù hợp cho các yêu cầu [12]. Trong nghiên cứu [13], các tác giả đã đề xuất cách tiếp cận tham lam cho bài toán VPTR mà tập các VNF có thứ tự bộ phận và ràng buộc vị trí. Các tác giả trong [14] đề xuất một kỹ thuật dựa trên làm tròn kết hợp với ILP để giảm thiểu năng lượng tiêu thụ của nhiều yêu cầu. Hơn nữa, trong trường hợp lưu lượng được định tuyến bởi các luồng khác nhau, thuật toán trong [15] có thể tìm các giải pháp gần đúng cho VPTR. Các công trình này chỉ bao gồm những sửa đổi nhỏ trong VPTR và đưa ra các giải pháp tốt trong thời gian tính toán hợp lý.

Thay vì giải các bài toán đơn mục tiêu trong bài toán định tuyến trong mạng ảo hóa như các nghiên cứu trước, đồ án xây dựng một bài toán tối ưu đa mục tiêu để đạt được sự cân bằng tốt hơn giữa các mục tiêu. Sau đó, đồ án này áp dụng một số giải thuật tiến hóa đa mục tiêu để giải quyết bài toán đã được xây dựng.

1.2 Bài toán tối ưu

Trong các công việc thực tế như kinh doanh, trồng trọt, xây dựng, ..., con người thường xuyên phải đưa ra các quyết định sao cho kết quả đạt được ở mức tốt nhất trong các điều kiện cho phép. Những bài toán thực tế như vậy thường có thể được mô hình hóa dưới dạng các bài toán tối ưu. Trong khoa học máy tính, bài toán tối ưu hóa là bài toán tìm ra lời giải tốt nhất trong các tập các lời giải khả thi. Việc phân loại và nhận dạng bài toán tối ưu là rất quan trọng trước khi bắt tay vào việc giải quyết bài toán đó. Do cấu trúc của không gian lời giải là khác nhau, giải quyết mỗi loại bài toán lại đòi hỏi những kĩ thuật, phương pháp đặc trưng riêng. Có nhiều cách phân loại các bài toán tối ưu, nếu xét theo số lượng mục tiêu, các bài toán tối ưu được sắp xếp vào hai nhóm chính như sau:

Tối ưu hóa đơn mục tiêu

Bài toán tối ưu đơn mục tiêu là một bài toán tối ưu hóa trong đó chỉ có một mục tiêu duy nhất cần được tối ưu. Mục tiêu này được biểu diễn dưới dạng hàm mục tiêu và thường là một hàm số toán học.

$$\begin{aligned} &minimize\, f(x) \\ &\text{với } x \in \mathbb{D} \end{aligned}$$

trong đó $\mathbb{D} \subseteq \mathbb{R}^n$ là miền xác định, $x = (x_1, x_2, ..., x_n)^T \in \mathbb{D}$ là một lời giải chấp nhận được, $f : \mathbb{D} \longrightarrow \mathbb{R}^n$ là hàm chi phí, hay hàm mục tiêu của bài toán. Nếu tập $\mathbb{D} \equiv \mathbb{R}^n$ ta nói bài toán là tối ưu không ràng buộc. Ngược lại, nếu $\mathbb{D} \subset \mathbb{R}^n$ bài toán tối ưu được gọi là có ràng buộc.
Việc giải bài toán tối ưu chính là đi tìm phần tử $x^* \in \mathbb{D}$ ứng với giá trị $f(x^*)$ nhỏ nhất. Điểm x^* khi ấy được gọi là nghiệm tối ưu toàn cục và $f(x^*)$ là giá trị tối ưu toàn cục của bài toán.

Tối ưu hóa đa mục tiêu

Trong thực tế, việc tìm lời giải cho các vấn đề tối ưu hóa đòi hỏi phải thỏa mãn cùng lúc nhiều tiêu chí (mục tiêu) tối ưu khác nhau [16]. Giả sử mọi mục tiêu đều là cực tiểu hóa, bài toán có thể biểu diễn như sau:

$$\begin{aligned} &minimize\, (f_1(x), f_2(x), ..., f_m(x)) \\ &\text{với } x \in \mathbb{D} \end{aligned} \tag{1.1}$$

trong đó $m \geq 2$ là số hàm mục tiêu của bài toán, $\mathbb{D} \subseteq \mathbb{R}^n$ là miền xác định thay đổi

tùy thuộc vào ràng buộc của bài toán đặt ra, $x = (x_1, x_2, ..., x_n)^T \in \mathbb{D}$ là một lời giải chấp nhận được.

Tính trội và tối ưu Pareto

Các mục tiêu của bài toán đa mục tiêu trong thực tế thường xung đột lẫn nhau, tức là nếu muốn cải thiện tính tối ưu của một mục tiêu thì phải giảm tính tối ưu của một mục tiêu khác. Chính vì vậy khái niệm về tính "trội" được đưa ra như sau.

Định nghĩa 1 (Tính trội Pareto [17]). Xét hai lời giải hợp lệ $x^1, x^2 \in \mathbb{D}$. Lời giải x^1 được gọi là "trội" hơn x^2 nếu thỏa mãn 2 điều kiện sau đây:

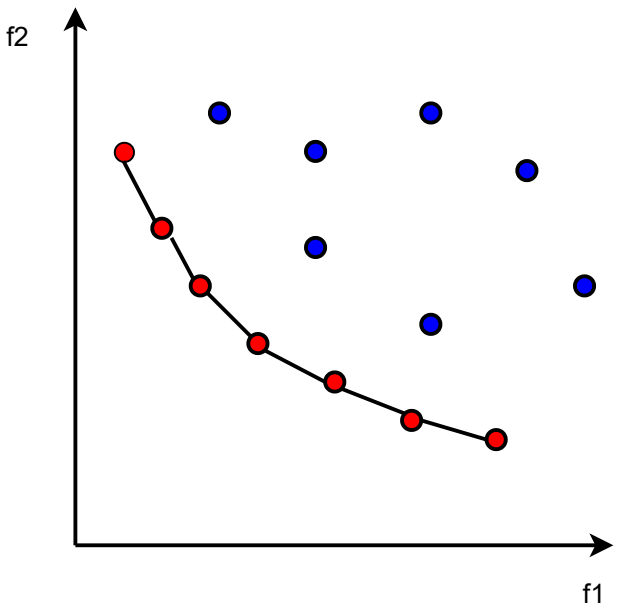
- $f_i(x^1) \leq f_i(x^2)\ \forall i \in 1, 2, ..., m$
- $\exists j \in \{1, 2, ..., m\} \mid f_j(x^1) < f_j(x^2)$

Từ đó, một số khái niệm dựa trên tối ưu Pareto (Pareto optimal [17]) sẽ được sử dụng như sau:

- Tập nghiệm không bị trội (non-dominated solution set) là tập tất cả các lời giải mà trong đó mỗi lời giải đều không trội lẫn nhau.
- Biên Pareto (Pareto front) là biên tạo bởi giá trị các hàm mục tiêu tương ứng của tập nghiệm không trội.
- Tập nghiệm tối ưu Pareto (Pareto-optimal set) là tập nghiệm không trội tốt nhất trong không gian tìm kiếm.
- Biên tối ưu Pareto (Pareto-optimal front) là biên tạo bởi giá trị các hàm mục tiêu tương ứng của tập nghiệm tối ưu Pareto.

Một biên Pareto trong bài toán cực tiểu hóa hai mục tiêu được minh họa trong Hình 1.1. Mỗi lời giải trong biên này đều có thể xem là "tối ưu" với mức độ tối ưu của các mục tiêu khác nhau. Với các thuật toán đa mục tiêu dựa trên khái niệm tối ưu Pareto, tập lời giải hình thành nên biên Pareto chính là đầu ra mong muốn của bài toán.

Mục tiêu cuối cùng của một thuật toán tối ưu đa mục tiêu là xác định các lời giải trong tập Pareto tối ưu. Tuy nhiên, việc xác định toàn bộ tập Pareto tối ưu, đối với nhiều bài toán đa mục tiêu, thực tế là không thể do kích thước của nó. Hơn nữa, đối với nhiều bài toán, đặc biệt là một số các bài toán tối ưu tổ hợp, việc chứng minh tính tối ưu của lời giải là gần như không thể. Do đó, một phương pháp tối ưu đa mục tiêu thực tế là điều tra một tập lời giải (tập Pareto tốt nhất đã biết) mà càng giống tập Pareto tối ưu càng tốt. Với những quan ngại này, ba mục tiêu chính của một phương pháp tối ưu đa mục tiêu bao gồm:



Hình 1.1: Minh họa biên pareto cho bài toán cực tiểu hóa hai mục tiêu. Những điểm màu xanh là các lời giải bị trội, điểm màu đỏ là lời giải không bị trội và đường nối các điểm màu đỏ là biên tối ưu Pareto.

- Pareto tốt nhất đã biết càng gần với Pareto tối ưu thực sự càng tốt. Lý tưởng nhất, tập Pareto tốt nhất đã biết nên là một phần tập Pareto tối ưu.
- Các lời giải trong tập Pareto tốt nhất đã biết nên phân bố đồng đều và đa dạng trên tập Pareto.
- Pareto tốt nhất đã biết nên bao phủ toàn bộ phổ Pareto. Điều này đòi hỏi tìm ra các lời giải ở hai đầu cực của không gian hàm mục tiêu.

Với giới hạn thời gian tính toán cho trước, mục tiêu đầu tiên được đạt tốt nhất bằng cách tập trung (tăng cường) tìm kiếm vào một vùng cụ thể của Pareto. Ngược lại, mục tiêu thứ hai đòi hỏi nỗ lực tìm kiếm phân bố đồng đều trên Pareto. Mục tiêu thứ ba nhằm mở rộng Pareto ở cả hai đầu, khám phá các lời giải cực đại mới.

1.3 Giải thuật tiến hóa

Do nhu cầu giải quyết các bài toán kích thước lớn ngày càng tăng, các giải thuật tiến hóa (Evolutionary Algorithms - EAs) đã và đang được áp dụng để giải các bài toán khó và trả về các lời giải gần đúng trong một khoảng thời gian chấp nhận được. Lớp các thuật toán này được lấy cảm hứng từ quá trình tiến hóa trong tự nhiên, trong đó các cá thể tốt hơn và thích nghi tốt hơn với môi trường sẽ có khả năng sinh tồn và sinh sản nhiều hơn. Quá trình tiến hóa diễn ra qua nhiều thế hệ, trong đó các lời giải tiềm năng được tạo ra và cải thiện dần theo thời gian. Sự "tiến hóa" của quần thể dựa trên các cơ chế như tái sản xuất, đột biến, tái tổ hợp và chọn lọc là mấu chốt tạo ra lời giải tối ưu hơn. Lớp các giải thuật tiến hóa bao gồm:

Giải thuật di truyền (Genetic Algorithm - GA), Chiến lược tiến hóa (Evolutionary Stragy - ES), Lập trình di truyền (Genetic Programing - GP), ... Tùy từng bài toán, chúng ta có thể lựa chọn các giải thuật tiến hóa khác nhau nằm trong lớp giải thuật hóa để giải chúng một cách hiệu quả. Tiếp theo, đồ án sẽ trình bày chi tiết về giải thuật di truyền - một trong các đại diện lâu đời và phổ biến nhất của lớp thuật toán tiến hóa, và các toán tử tiến hóa thường được dùng trong giải thuật di truyền.

Khái niệm về giải thuật di truyền (Genetic Algorithm - GA) được phát triển bởi Holland và đồng nghiệp của ông trong những năm 1960 và 1970 [18]. GA được truyền cảm hứng từ lý thuyết tiến hóa giải thích nguồn gốc của các loài. Trong tự nhiên, các loài yếu và không phù hợp với môi trường của chúng đối diện với sự tuyệt chủng thông qua quá trình lựa chọn tự nhiên. Các loài mạnh mẽ có cơ hội lớn hơn để truyền gen của mình cho các thế hệ tiếp theo qua quá trình sinh sản. Trong dài hạn, các loài mang sự kết hợp chính xác trong gen của mình trở thành loài ưu thế trong quần thể của chúng. Đôi khi, trong quá trình tiến hóa chậm rãi, sự thay đổi ngẫu nhiên có thể xảy ra trong gen. Nếu những thay đổi này mang lại lợi thế bổ sung trong cuộc chiến sinh tồn, các loài mới sẽ tiến hóa từ những loài cũ. Những thay đổi không thành công sẽ bị loại bỏ thông qua quá trình lựa chọn tự nhiên.

Trong giải thuật di truyền, mỗi cá thể được biểu diễn dưới dạng một chuỗi gen, tương ứng với các thông số của lời giải. Quá trình tiến hóa diễn ra thông qua các toán tử như lai ghép (crossover) và đột biến (mutation), trong đó các gen được kết hợp và thay đổi để tạo ra cá thể con mới. Mỗi cá thể trong quần thể được đánh giá bằng cách sử dụng một hàm mục tiêu (objective function) để đo lường chất lượng hoặc hiệu suất của lời giải. Các cá thể có độ thích nghi cao được ưu tiên để tiếp tục vào thế hệ tiếp theo, trong khi các cá thể yếu hơn có khả năng bị loại bỏ. Qua nhiều thế hệ, giải thuật di truyền tìm kiếm và tối ưu hóa các lời giải tiềm năng trong quần thể. Nó đã được ứng dụng rộng rãi trong nhiều lĩnh vực, bao gồm tối ưu hóa hàm số, tối ưu hóa tổ hợp, thiết kế mạng neural, lập lịch và nhiều bài toán tối ưu hóa khác. Với tính linh hoạt và khả năng khám phá, giải thuật di truyền là một công cụ mạnh mẽ để giải quyết các bài toán phức tạp và tìm kiếm các lời giải tối ưu.

Biểu diễn cá thể

Biểu diễn cá thể là bước đầu tiên và quan trọng trong giải thuật di truyền. Trong giải thuật di truyền, mỗi cá thể trong quần thể được biểu diễn dưới dạng một chuỗi gen, còn được gọi là chromosome. Biểu diễn cá thể là cách mà các thông tin gen của nó được mã hóa và đại diện cho một lời giải tiềm năng cho bài toán được giải. Một biểu diễn tốt sẽ giúp tìm kiếm không gian lời giải hiệu quả và có khả năng tìm ra các lời giải tối ưu trong quần thể. Một số cách mã hóa phổ biến có thể kể

đến như: mã hóa hoán vị, mã hóa số thực, mã hóa nhị phân, mã hóa Prufer, mã hóa Cayley, ... [19].

Khởi tạo quần thể

Khởi tạo quần thể trong giải thuật di truyền là quá trình tạo ra một tập hợp ban đầu các cá thể (các lời giải tiềm năng cho bài toán được giải) để bắt đầu quá trình tiến hóa. Quá trình khởi tạo quần thể nhằm đảm bảo tính đa dạng và đại diện của các cá thể trong không gian tìm kiếm. Điều này đảm bảo rằng quần thể có khả năng tìm kiếm các vùng khác nhau của không gian tìm kiếm và tăng cơ hội tìm ra các lời giải tốt hơn. Việc lựa chọn phương pháp khởi tạo quần thể phù hợp có thể ảnh hưởng đáng kể đến hiệu suất và khả năng tìm kiếm của giải thuật di truyền.

Có nhiều phương pháp khác nhau để khởi tạo quần thể, bao gồm:

1. Khởi tạo ngẫu nhiên: Các cá thể trong quần thể được tạo ra một cách ngẫu nhiên, bằng cách chọn các giá trị gen hoặc thông số ngẫu nhiên cho mỗi cá thể.
2. Khởi tạo thông minh: Sử dụng kiến thức về bài toán hoặc các phương pháp thông minh như tìm kiếm cục bộ để tạo ra các cá thể ban đầu có khả năng gần lời giải tối ưu hơn.
3. Khởi tạo dựa trên kinh nghiệm: Sử dụng thông tin từ quần thể hoặc các lời giải đã biết trước đó để khởi tạo các cá thể ban đầu.

Hàm thích nghi

Hàm thích nghi (fitness function) là một công thức toán học được sử dụng để đánh giá chất lượng hoặc sự phù hợp của một cá thể trong quần thể. Cá thể có độ thích nghi cao sẽ cho thấy nó là cá thể có tốt và nên được giữ lại cho thế hệ tiếp theo. Bằng cách đánh giá và so sánh độ thích nghi của các cá nhân trong một quần thể theo cách lặp lại, thuật toán tối ưu hóa có thể tiến triển đến các lời giải cải thiện qua các thế hệ. Thiết kế một hàm thích nghi phù hợp và tính toán độ thích nghi một cách chính xác là những yếu tố quan trọng để đạt được kết quả tối ưu hóa thành công. Có nhiều cách chọn hàm thích nghi khác nhau tùy vào bài toán, một cách chọn hàm thích nghi phổ biến là sử dụng luôn hàm mục tiêu trong mô hình bài toán đã đề ra.

Phép lai ghép

Phép lai ghép là một toán tử quan trọng trong thuật toán tiến hóa để tạo ra các cá thể con mới bằng cách kết hợp thông tin di truyền từ hai cá thể cha mẹ. Phép lai ghép thường được áp dụng trên các lời giải được biểu diễn bằng chuỗi gen hoặc cấu trúc tương tự. Quá trình lai ghép bắt đầu bằng việc chọn ngẫu nhiên một vị trí

lai ghép trong chuỗi gen của hai cha mẹ. Sau đó, một phần gen từ một cha mẹ được truyền sang cá thể con, trong khi phần gen còn lại được truyền từ cha mẹ còn lại. Quá trình lai ghép này tạo ra sự kết hợp và đa dạng hóa di truyền trong quần thể, cho phép khám phá không gian tìm kiếm một cách hiệu quả. Một vài phép lai ghép thường được sử dụng:

- **Lai ghép 1 điểm cắt (Single-Point Crossover)**: Là phép lai ghép đơn giản nhất, trong đó một điểm cắt ngẫu nhiên được chọn và các phần gen trước điểm cắt được lấy từ một cha mẹ, và các phần gen sau điểm cắt được lấy từ cha mẹ còn lại.
- **Lai ghép đồng nhất (Uniform Crossover)**: Mỗi gen trong cá thể con được chọn từ cha mẹ ngẫu nhiên. Các gen có thể được trao đổi hoặc duy trì nguyên vị trí dựa trên một xác suất xác định.
- **Lai ghép nhị phân (Simulated Binary Crossover)**: Thường được sử dụng trong các bài toán tối ưu giá trị thực. Nó mô phỏng toán tử chéo nhị phân bằng cách áp dụng hàm phân phối xác suất để tạo con cái.

Phép đột biến

Ngoài phép lai ghép, phép đột biến thường được sử dụng để tăng độ đa dạng trong quần thể. Trong quá trình đột biến, một hoặc nhiều gene của cá thể con được thay đổi ngẫu nhiên để tạo ra một cá thể mới. Nếu lai ghép được sử dụng để khai thác tiềm năng từ các lời giải hiện thời, đột biến đem lại khả năng khám phá những khu vực mới trong không gian tìm kiếm, giúp việc tìm kiếm thoát khỏi điểm tối ưu cục bộ. Một vài phép đột biến hay được sử dụng:

- **Đột biến đảo bit (Bit-Flip Mutation)**: Trong phép đột biến này, một hoặc nhiều bit trong chuỗi gen của cá thể con được ngẫu nhiên chuyển đổi từ 0 sang 1 hoặc từ 1 sang 0.
- **Đột biến hoán vị (Swap Mutation)**: Trong phép đột biến này, hai gen trong chuỗi gen của cá thể con được hoán đổi vị trí.
- **Đột biến Gaussian (Gaussian Mutation)**: Toán tử đột biến này làm xáo trộn các giá trị của gen trong nhiễm sắc thể bằng cách sử dụng phân bố Gaussian. Các giá trị mới được lấy mẫu ngẫu nhiên xung quanh các giá trị ban đầu.

Chọn lọc cá thể cho thế hệ tiếp

Chọn lọc là quá trình chọn lựa các cá thể để tham gia vào quá trình tiến hóa tiếp theo. Một số phương pháp chọn lọc phổ biến trong giải thuật di truyền là:

- **Chọn lọc theo mức độ cạnh tranh (Tournament Selection)**: Các cá thể được

chia thành các nhóm nhỏ và các cá thể trong từng nhóm tham gia một cuộc thi. Các cá thể có hiệu suất tốt nhất trong từng nhóm được chọn để tiếp tục vào quá trình tiến hóa.

- **Chọn lọc theo phần tử tốt nhất (Elitism Selection)**: Một số cá thể tốt nhất trong quần thể được chọn trực tiếp và giữ lại trong quần thể cho đến các thế hệ tiếp theo. Điều này giúp bảo toàn những lời giải tốt nhất và ngăn chặn việc mất mát thông tin quan trọng trong quá trình tiến hóa.

Điều kiện dừng của thuật toán

Điều kiện dừng (termination condition) trong giải thuật di truyền được thiết lập để quyết định thời điểm thuật toán nên dừng lại và trả về lời giải tốt nhất hiện tại. Thông thường, giải thuật di truyền sẽ kết thúc sau một số thế hệ đã được xác định trước hoặc dừng khi sau một số thế hệ không cải thiện được hàm mục tiêu của bài toán. Việc chọn trước số thế hệ giúp tiết kiệm chi phí tính toán vì giải thuật sẽ dừng lại sau một số bước lặp xác định. Hiện nay, trong lớp các giải thuật tiến hóa, số lần đánh giá tối đa cho phép thường được sử dụng làm điều kiện dừng để đảm bảo tính công bằng khi so sánh với các giải thuật khác. Ngoài ra, giải thuật di truyền có thể dừng sau một số thế hệ không tìm được lời giải cải thiện hơn, nhằm dừng giải thuật khi khả năng tìm kiếm lời giải tốt bị giảm. Tuy nhiên, việc dừng giải thuật khi các toán tử di truyền vẫn có khả năng tạo ra các lời giải tốt có thể ảnh hưởng đến chất lượng của lời giải đạt được. Do đó, việc lựa chọn điều kiện dừng hợp lý là cân nhắc giữa chi phí thời gian và chất lượng lời giải.

1.4 Phân loại các phương pháp giải bài toán tối ưu đa mục tiêu

Các thuật toán tiến hóa đa mục tiêu có thể được phân loại dựa trên đặc điểm và chiến lược của chúng. Trong phần này, đồ án sẽ trình bày ba hướng tiếp cận phổ biến và các ưu nhược điểm của ba cách tiếp cận.

1.4.1 Phương pháp tổng có trọng số

Phương pháp tổng có trọng số là phương pháp trực quan nhất để giải quyết các bài toán tối ưu đa mục tiêu. Mục tiêu của phương pháp này là tổng hợp tuyến tính toàn bộ hàm mục tiêu thành một hàm mục tiêu duy nhất. Để làm điều này, thuật toán sử dụng các trọng số để tính tổng có trọng số của các hàm mục tiêu. Một bài toán tối ưu với m mục tiêu $f_1, ..., f_m$ được chuyển đổi thành một bài toán tối ưu mục tiêu duy nhất có hàm mục tiêu như sau:

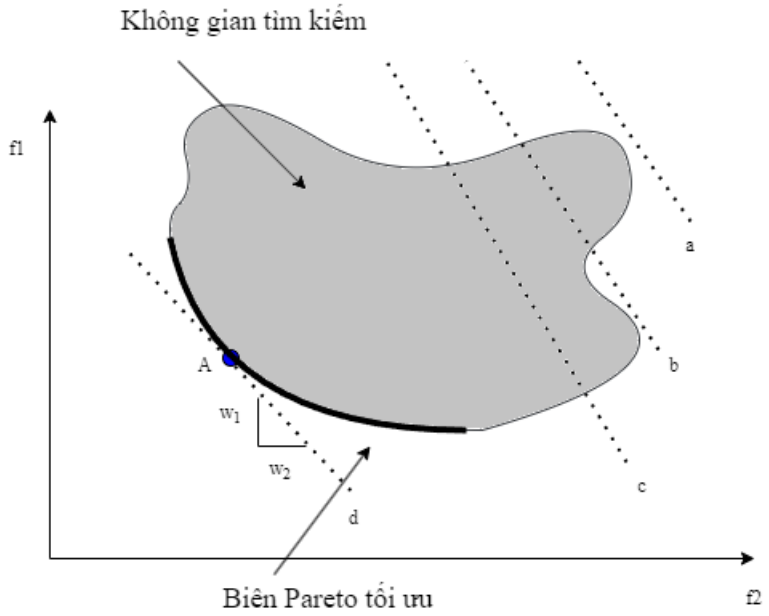
$$\min_{x \in \mathbb{D}} f(x) = \sum_{i=1}^{m} w_i f_i(x) \tag{1.2}$$

trong đo $w_i \neq 0$ và $w_i \in [0, 1]$, thường được gọi là trọng số hay độ quan trọng của các mục tiêu, $w_1 + w_2 + ... + w_m = 1$.

Tiếp theo, đồ án minh họa cách phương pháp tổng trọng số có thể tìm ra các lời giải tối ưu Pareto của bài toán gốc. Để đơn giản, đồ án chỉ xem xét bài toán tối ưu hai mục tiêu được thể hiện trong hình 1.2. Không gian mục tiêu khả thi và tập hợp các lời giải tối ưu Pareto tương ứng được hiển thị. Với hai mục tiêu, có hai trọng số w_1 và w_2, nhưng chỉ một trọng số là độc lập. Biết một trọng số, ta có thể tính toán trọng số còn lại bằng phép trừ đơn giản. Rõ ràng từ hình vẽ rằng sự lựa chọn của một vector trọng số tương ứng với một lời giải tối ưu đã được xác định trước trên mặt tối ưu Pareto, như được đánh dấu bằng điểm A. Bằng cách thay đổi vector trọng số, một điểm tối ưu Pareto khác có thể được thu được. Tuy nhiên, có một số khó khăn với phương pháp này:

- Việc chọn đồng nhất các vector trọng số không nhất thiết dẫn đến tìm một tập hợp đồng nhất các lời giải tối ưu Pareto trên mặt tối ưu Pareto [20].
- Phương pháp này không thể được sử dụng để tìm các lời giải tối ưu Pareto nằm trên phần không lồi của mặt tối ưu Pareto.

Vấn đề đầu tiên làm cho việc áp dụng phương pháp tổng trọng số trở nên khó khăn và không đáng tin cậy cho bất kỳ bài toán nào để tìm một tập hợp đại diện tốt các lời giải tối ưu Pareto. Vấn đề thứ hai phát sinh do việc một lời giải nằm trên mặt tối ưu Pareto không lồi không thể bao giờ là lời giải tối ưu của bài toán được đưa ra trong công thức 1.2.



Hình 1.2: Hình vẽ mình họa giải thuật tổng trọng số trong bài toán tối ưu 2 mục tiêu.

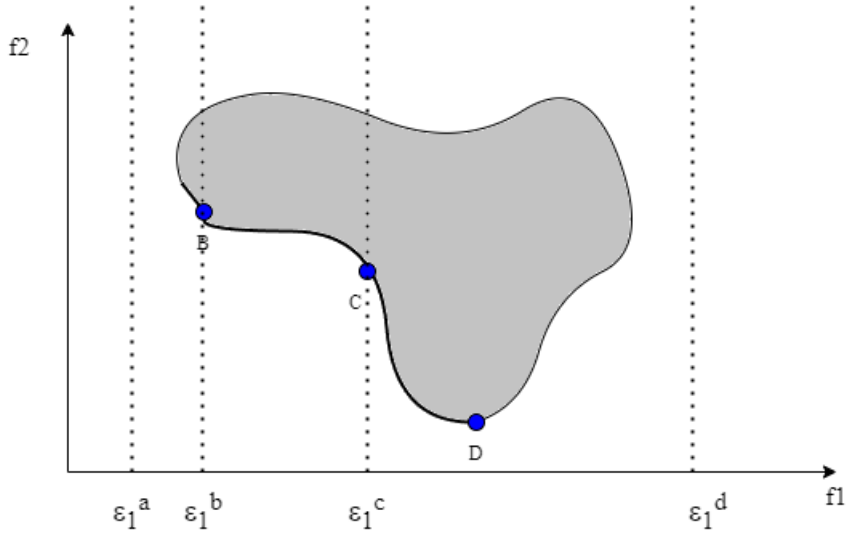
1.4.2 Phương pháp ràng buộc ε

Để giảm bớt khó khăn gặp phải bởi phương pháp tổng trọng số trong việc giải quyết các bài toán có không gian mục tiêu không lồi, phương pháp ràng buộc ε được sử dụng. Haimes [21] đã đề xuất chuyển đổi bài toán tối ưu đa mục tiêu với m mục tiêu và n ràng buộc thành một bài toán tối ưu đơn mục tiêu dưới sự ràng buộc của $n + m - 1$ ràng buộc bằng cách giới hạn việc tối ưu hóa $m - 1$ mục tiêu cho một lượng được xác định trước phụ thuộc vào mục tiêu. Bài toán được chuyển đổi như sau:

$$\begin{aligned} &\min_{x \in \mathbb{D}} f_k(x) \\ &\text{với} \quad g_i(x) \leq 0 \quad 1 \leq i \leq n \\ &f_j(x) \leq \varepsilon_j \quad 1 \leq j \leq m \quad and \quad j \neq k \end{aligned} \tag{1.3}$$

Giả sử chúng ta giữ lại f_2 như một mục tiêu và xem f_1 như một ràng buộc: $f_1(x) \leq \varepsilon_1$. Hình 1.3 hiển thị bốn tình huống với các giá trị ε_1 khác nhau. Hãy xem xét tình huống thứ ba với $\varepsilon_1 = \varepsilon_1^c$ trước. Bài toán kết quả với ràng buộc này chia không gian mục tiêu khả thi ban đầu thành hai phần, $f_1 \leq \varepsilon_1^c$ và $f_1 > \varepsilon_1^c$. Phần bên trái trở thành lời giải khả thi của bài toán kết quả được nêu trong Công thức 1.3. Bây giờ, nhiệm vụ của bài toán kết quả là tìm lời giải giảm thiểu trong vùng khả thi này. Từ Hình 1.3, rõ ràng rằng lời giải tối thiểu là C. Điều này cho phép thu được các lời giải Pareto trung gian trong trường hợp các bài toán không gian mục tiêu không lồi bằng phương pháp ràng buộc ε.

Một trong những khó khăn của phương pháp này là lời giải cho bài toán được nêu trong Công thức 1.3 phụ thuộc lớn vào vector ε được chọn. Hãy xem lại Hình 1.3. Thay vì chọn ε_1^c, nếu chọn ε_1^a, không có lời giải khả thi cho bài toán được nêu ra. Do đó, sẽ không tìm thấy lời giải nào. Trong khi đó, nếu sử dụng ε_1^d, toàn bộ không gian tìm kiếm đều khả thi. Bài toán kết quả có giá trị nhỏ nhất tại D. Hơn nữa, khi số lượng mục tiêu tăng lên, có nhiều thành phần hơn trong vector ε, do đó yêu cầu nhiều thông tin hơn từ người dùng.



Hình 1.3: Hình vẽ mình họa giải thuật ràng buộc ε trong bài toán tối ưu 2 mục tiêu.

1.4.3 Phương pháp đa mục tiêu tiến hóa

Hầu hết các phương pháp truyền thống như ở trên đều cần dựa vào tri thức của con người. Các kỹ thuật như vậy có thể xấp xỉ biên Pareto tối ưu bằng cách lặp lại quá trình tìm kiếm sau khi điều chỉnh các tham số tổng hợp (trọng số hay gọi là mức độ quan trọng của mục tiêu). Tuy nhiên, biên Pareto thu được chưa chắc đã là biên Pareto tối ưu toàn cục. Hơn nữa, phân bố các lời giải trên biên phụ thuộc vào hiệu suất của phương pháp giải quyết bài toán tối ưu đơn mục tiêu được sử dụng. Mặc dù các kỹ thuật này tương đối đơn giản để triển khai, chúng thường không hiệu quả và đôi khi nhạy cảm với hình dạng của biên Pareto.

So sánh với hai phương pháp cổ điển trên, một số thuật toán metaheuristic thiết kế nhằm trực tiếp tạo ra biên Pareto bằng cách tối ưu hóa đồng thời các mục tiêu riêng lẻ. Các thuật toán dựa trên bầy đàn có lợi thế trong việc đánh giá đồng thời nhiều lời giải tiềm năng trong một lần lặp. Ngoài ra, chúng cung cấp sự linh hoạt lớn cho người ra quyết định, đặc biệt là trong những trường hợp không có thông tin tiên quyết có sẵn như là trường hợp của hầu hết các vấn đề tối ưu đa mục tiêu trong thực tế. Tuy nhiên, thách thức đặt ra là làm thế nào để hướng dẫn tìm kiếm đến tập biên Pareto tối ưu và làm thế nào để duy trì một quần thể đa dạng để ngăn chặn sự hội tụ sớm.

Như đã trình bày trong phần trước, tính toán tiến hóa mô phỏng quá trình tiến hóa sinh học. Một quần thể các cá thể đại diện cho các lời giải khác nhau đang tiến hóa để tìm ra các lời giải tối ưu. Các cá thể mạnh mẽ nhất được lựa chọn, áp dụng các phép đột biến và lai ghép, từ đó tạo ra thế hệ mới (con cái). Chúng thường được kết hợp với các kỹ thuật truyền thống được trình bày trong phần trước. Cần tránh nhầm lẫn giữa các kỹ thuật dựa trên thuật toán tiến hóa kết hợp và những kỹ thuật

hướng đến xác định các lời giải Pareto hiệu quả được trình bày trong phần này.

Các kỹ thuật đa mục tiêu tiến hóa (Evolutionary Multi-Ojective Otimization - EMO) giải quyết thành công những hạn chế của các phương pháp cổ điển khi tạo ra biên Pareto. Bởi vì chúng cho phép khám phá đồng thời các điểm khác nhau trên biên Pareto, chúng có thể tạo ra nhiều lời giải trong một lần chạy duy nhất. Quá trình tối ưu hóa có thể được thực hiện mà không cần thông tin trước về mức độ quan trọng tương đối của các mục tiêu. Các kỹ thuật này có thể xử lý các vấn đề không chính xác với các mục tiêu không thể so sánh. Ngoài ra, chúng không bị ảnh hưởng bởi hình dạng của mặt Pareto. Hạn chế chính của nhóm lời giải này là hiệu suất giảm khi số lượng mục tiêu tăng lên, vì không tồn tại phương pháp tính toán hiệu quả tính thứ hạng Pareto. Ngoài ra, chúng yêu cầu các tham số bổ sung như hệ số chia sẻ (sharing factor) hoặc số lượng mẫu Pareto cần điều chỉnh.

CHƯƠNG 2. BÀI TOÁN ĐỊNH TUYẾN TRONG MẠNG ẢO HÓA

Phần này sẽ trình bày các nghiên cứu liên quan về bài toán định tuyến các chuỗi dịch vụ trong mạng ảo hóa và đưa ra mô hình đa mục tiêu cho bài toán. Bài toán MO-TRR sẽ bao gồm hai mục tiêu: tối đa hóa tỉ lệ chuỗi dịch vụ được đáp ứng và tối đa hóa khả năng sử dụng tài nguyên mạng.

2.1 Các nghiên cứu liên quan

Do các tính năng vốn có và nguyên tắc thiết kế của NFV, vấn đề định tuyến khác với vấn đề cơ bản vì khi các VNF được yêu cầu được đặt trên mạng, ràng buộc SFC đòi hỏi tuyến đường từ nguồn tới đích phải đi qua mỗi VNF theo một thứ tự xác định. Sallam Wang và các cộng sự [22] đề xuất một thuật toán phân tán (Alternating Direction Method of Multipliers - DMM) để giải quyết bài toán cân bằng tải (load balancing) trong vấn đề TRR cho nhiều chuỗi dịch vụ. Thuật toán được đề xuất trong [22] đã được chứng minh có tỷ lệ phủ cố định. Trong nghiên cứu [23], nhóm tác giả đề xuất một thuật toán có độ phức tạp trong thời gian đa thức để giải bài toán Đường đi ngắn nhất (Shortest Path) với ràng buộc SFC thông qua một đồ thị phụ trợ. Ràng buộc SFC trong bài toán Luồng cực đại (Maximum Flow) yêu cầu mỗi dòng dữ liệu đi qua một tập hàm mạng theo một thứ tự đã quy định trước trước khi đến đích. Sau đó, họ đề xuất một mô hình quy hoạch nguyên tuyến tính (Integer Linear Programming - ILP) để giải quyết việc đặt các VNF sao cho giá trị luồng cực đại với ràng buộc SFC bằng giá trị luồng cực đại ban đầu mà không có ràng buộc SFC. Trong bài toán định tuyến đa điểm (Multicast Traffic Routing - MTRR) trong NFV, có một nút nguồn và nhiều nút đích, và bài toán là tìm một cây đa điểm từ nguồn tới các đích sao cho mỗi đường đi từ nguồn tới đích đi qua SFC yêu cầu. Z.Xu và các cộng sự trong nghiên cứu [24] đã đề xuất một thuật toán xấp xỉ để giải bài toán MTRR bằng cách chuyển đổi nó thành bài toán cây Steiner trong một đồ thị phụ trợ vô hướng.

Ngoài ra, các tác giả trong nghiên cứu [25] và [26] đề xuất một heuristics dựa trên đồ thị lớp để giải quyết bài toán TRR (Traffic Routing and Resource Allocation) nhằm tối ưu hóa độ trễ. Ý tưởng chung là xây dựng một đồ thị đa tầng bằng cách tạo ra $|F|$ bản sao của đồ thị gốc, trong đó $|F|$ đại diện cho số lượng VNF được yêu cầu. Các liên kết giữa các tầng được tạo ra để kết nối cùng một nút trong các tầng khác nhau, và các nút giữa các tầng được tạo ra để đại diện cho các vị trí khả dụng để đặt mỗi VNF trong mỗi tầng. Bằng cách gán trọng số độ trễ cho các liên kết, thuật toán được đề xuất tìm đường đi ngắn nhất từ nút nguồn vào trong tầng 1 đến nút đích trong tầng $|F|$. Xie và các cộng sự trong [27] nghiên cứu bài toán

định tuyến đa nguồn đa đích theo độ trễ trong mạng ảo hóa, trong đó có nhiều nút nguồn và nút đích. Xie et al. [27] trình bày một thuật toán heuristics dựa trên cây bao trùm tối thiểu (minimum spanning tree) luôn cố gắng tìm các liên kết chung để các SFC triển khai có thể được chia sẻ bởi nhiều yêu cầu (người dùng).

Trong trường hợp mạng có một số dịch vụ xảy ra sự cố, Yu và các cộng sự [12] đưa ra định nghĩa về mức độ đáng tin cậy như sau: trong trường hợp một dịch vụ đơn lẻ tùy ý xảy ra sự cố, lượng băng thông bị mất tối đa là r_j, do đó cần sử dụng nhiều đường đi để truyền dữ liệu giữa mỗi cặp VNF liền kề. Giả sử rằng VNF đã được đặt trong mạng và một tập các đường đi giữa cặp nút đã được cung cấp, nhóm tác giả trong nghiên cứu [12] đề xuất cách tìm một tập con các đường đi khả thi giữa mỗi cặp VNF liền kề, sao cho yêu cầu độ tin cậy được đáp ứng. Họ đề xuất một giải thuật Fully-Polynomial Time Approximation Scheme (FPTAS) dựa trên lý thuyết đối ngẫu để giải quyết bài toán này.

2.2 Bài toán định tuyến đa mục tiêu trong mạng ảo hóa

Một mạng ảo hóa được mô hình hóa bằng một đồ thị vô hướng $G(V, E)$, trong đó V và E lần lượt là tập hợp các nút và liên kết. Trong mạng vật lý, chúng ta sử dụng $u, v \in V$ để chỉ ra hai nút và $e_{uv} \in E$ để biểu thị liên kết kết nối nút u và v. Tập V có thể được chia ra thành hai tập con bao gồm: tập các nút vật lý V_p (physical nodes) và tập các nút máy chủ V_s (server nodes). Trong đó các nút vật lý chịu trách nhiệm chuyển tiếp dữ liệu đến các nút lân cận và các nút máy chủ không chỉ phụ trách chuyển tiếp dữ liệu mà cho phép triển khai các chức năng mạng ảo VNF.

Trong mạng, mỗi nút $v_i \in V$ được đặc trưng bởi hai thông số: dung lượng bộ nhớ cap_i và khả năng tính toán cpu_i để có thể sử dụng được các dịch vụ mạng ảo hóa (VNF) khác nhau. Giả định rằng các VNF chỉ có thể được đặt ở trên các nút máy chủ . Các nút vật lý sẽ chỉ được sử dụng cho định tuyến và chuyển mạch thông thường. Vì vậy, với các nút máy chủ ta sẽ chỉ xét đến khả năng tính toán và bỏ qua dung lượng bộ nhớ và ngược lại với các nút vật lý. Nói một cách khác, cap_i được cho bằng vô hạn nếu v_i là nút máy chủ và cpu_i được cho bằng 0 nếu v_i là nút vật lý. Mỗi cạnh $e_{ij} \in E$ giữa 2 đỉnh v_i và v_j có lượng băng thông khả dụng w_{ij}.

Giả định rằng k loại VNF được triển khai ví dụ như tường lửa, IDS, proxy, cân bằng tải, và các loại khác. Trong đồ án, tập hợp các VNF được ký hiệu là $F = \{f_1, f_2, ..., f_k\}$, trong đó f_i là loại VNF thứ i. Trong bài toán đang xét, tất cả các VNF đã được triển khai. Ký hiệu cn_{ij} là một biến nhị phân, trong đó cn_{ij} = 1 cho biết rằng VNF f_i được cài đặt trên nút v_j; ngược lại, giá trị của nó là 0.

Một dịch vụ mạng là một chuỗi yêu cầu/chuỗi dịch vụ (Service Chain Function - SFC) hoàn chỉnh được cung cấp cho người dùng bởi các nhà điều hành mạng.

Thông thường, nó bao gồm một số chức năng mạng ảo phải được thực thi theo một thứ tự xác định. Mỗi chuỗi dịch vụ từ đầu đến cuối $r \in R$ như một bộ 6 phần tử $(s^r, t^r, \alpha^r, \beta^r, \gamma^r, F^r)$, trong đó s^r và t^r lần lượt đại diện cho đỉnh bắt đầu và đỉnh kết thúc. Các thông số α^r, β^r và γ^r lần lượt đại diện cho yêu cầu về tài nguyên bộ nhớ, tính toán và băng thông. $F^r = \{f_1^r, f_2^r, ..., f_t^r\}$ là một tập hợp có thứ tự các VNF mà SFC cần phải đáp ứng trong quá trình định tuyến.

Bảng 2.1: Danh sách các ký hiệu trong bài toán.

Ký hiệu	Mô tả
Mạng vật lý	
$G(V, E)$	V: tập đỉnh gồm 2 loại nút bao gồm nút vật lý và nút máy chủ E: tập các cạnh
V_p	Tập các nút vật lý
V_s	Tập các nút máy chủ
w_{ij}	Băng thông giữa cạnh nối hai đỉnh v_i và v_j
cpa_i	Tài nguyên bộ nhớ của nút i
cpu_i	Tài nguyên tính toán của nút i
F	Tập các VNFs, $F = f_1, f_2, \ldots, f_k$
R	Tập các SFC cần phục vụ, $R = \{r_1, r_2, \ldots, r_m\}$
k	Số lượng VNF
m	Số lượng SFC
Chuỗi dịch vụ SFC	
s^r	Đỉnh đầu vào của SFC r
t^r	Đỉnh đầu ra của SFC r
α^r	Tài nguyên về bộ nhớ yêu cầu của SFC r
β^r	Tài nguyên về tính toán yêu cầu của SFC r
γ^r	Tài nguyên về băng thông yêu cầu của SFC r
$count_i^r$	Số lần SFC r đi qua nút vật lý i
F^r	Tập có thứ tự VNFs mà SFC cần đáp ứng
Biến nhị phân	
cn_{ij}	Quyết định đặt VNF f_i trên nút v_j
cp_{ij}^r	Quyết định SFC r sử dụng VNF f_i trên nút v_j
x_r	Quyết định SFC r có được đáp ứng
z_{ij}^r	Quyết định SFC r có đi qua cạnh nối giữa hai nút i, j

Phần tiếp theo sẽ trình bày chi tiết mô hình toán học của bài toán tối ưu đa mục tiêu định tuyến trong mạng ảo hóa.

Đầu vào

- Mạng ảo hóa $G(V, E)$ và tập các chức năng k loại VNF ở trong mạng.
- Tập các chuỗi dịch vụ SFC cần đáp ứng $R = \{r_1, r_2, ..., r_m\}$.

Đầu ra

- Đường định tuyến cho từng chuỗi dịch vụ được đáp ứng trong R được ký hiệu $p^r = \{p_1^r, p_2^r, ..., p_{l_r}^r\}, \forall r = 1, 2, ..., m$
- Tập các biến quyết định cp_{ij}^r, trong đó nếu $cp_{ij}^r = 1$ tức là chuỗi dịch vụ r có sử dụng VNF f_j ở nút v_i và $cp_{ij}^r = 0$ sẽ ngược lại.

Ràng buộc

- Mỗi chuỗi dịch vụ r phải bắt đầu từ s_r và kết thúc tại t_r: $p_1^r = s_r$, $p_l^r = t_r$.
- Giới hạn về tài nguyên bộ nhớ: Tài nguyên bộ nhớ của nút vật lý i bị giới hạn; bởi vậy tổng bộ nhớ của các chuỗi dịch vụ do nút i cung cấp không được vượt quá dung lượng của nút này.

$$\sum_{r=1}^{m} x_r * \alpha_r * count_i^r \leq cap_i \quad \forall i \in V_p \tag{2.1}$$

  trong đó $count_i^r$ là số lần SFC r đi qua nút vật lý i.

- Giới hạn về tài nguyên tính toán: Tương tự như nút vật lý, tài nguyên tính toán nút máy chủ i cung cấp bị giới hạn không được vượt quá dung lượng của nút này.

$$\sum_{r=1}^{m} x_r * \beta_r * \sum_{j=1}^{K} cp_{ij}^r \leq cpu_i \quad \forall i \in V_s \tag{2.2}$$

- Giới hạn về băng thông: Phương trình 2.3 đảm bảo rằng lưu lượng đi qua liên kết nào đều không được vượt quá dung lượng của liên kết này. Biến nhị phân $z_{ii'}^r$ để biểu diễn nếu đường định tuyến p^r cho chuỗi dịch vụ r đi qua liên kế giữ hai đỉnh v_i và $v_{i'}$.

$$\sum_{r=1}^{m} x_r(\alpha_r z_{ii'}^r) \leq w_{ii'} \quad \forall i, i' \in \{1, ..., n\}. \tag{2.3}$$

- Phương trình 2.4 đảm bảo rằng VNF chỉ được thực hiện nếu nó đã được triển khai.

$$\frac{\sum_{r=1}^{m} x_r cp_{ij}^r}{m} \leq cn_{ij} \forall i = 1, 2, ..., n; j = 1, 2, ..., K \tag{2.4}$$

  trong đó $cn_{ij} = 1$ nếu VNF f_j được đặt trên máy chủ v_i, $cn_{ij} = 0$ nếu ngược lại.

Mục tiêu

- Tối đa số lượng chuỗi dịch vụ được đáp ứng

$$\sum_{r=1}^{m} x_r \rightarrow max \tag{2.5}$$

trong đó $x_r = 1$,nếu chuỗi dịch vụ SFC $r - th$ được đáp ứng và $x_r = 0$ nếu ngược lại.

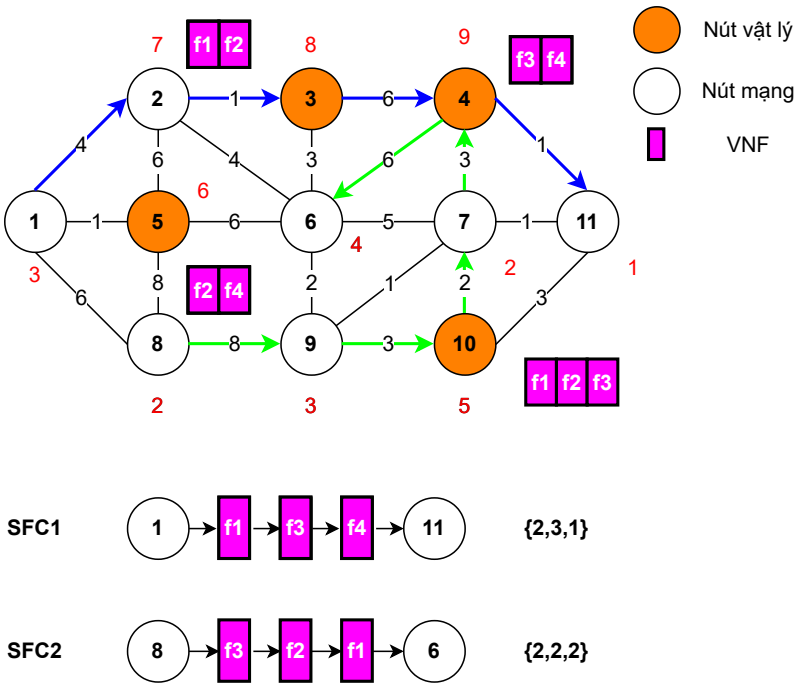
- Tối đa hóa khả năng sử dụng tài nguyên mạng

Sử dụng tài nguyên là mối quan tâm chính trong việc phân bổ tài nguyên trong các VNF. Giá trị này phụ thuộc vào yêu cầu tài nguyên và tài nguyên có sẵn trong mạng nền. Đối với tập hợp các chuỗi dịch vụ SFC được chấp nhận, giá trị này được tính toán dựa trên tài nguyên có sẵn của các nút và liên kết theo phương trình 2.6.

$$L_b = 1 - \frac{rs_1 + rs_2 + rs_3}{3} \rightarrow max \tag{2.6}$$

trong đó rs_1 lượng băng thông tối đa của các liên kết. rs_2 and rs_3 lần lượt là tỉ lệ lượng tài nguyên bộ nhớ và tài nguyên tính toán tối đa được sử dụng trong mạng. Giá trị L_b càng lớn tức là lượng tài nguyên còn lại trong mạng càng nhiều.

Hai mục tiêu này thường xung đột lẫn nhau và cần được cân bằng một cách hiệu quả.



Hình 2.1: Hình vẽ minh họa bài toán đa mục định tuyến trong mạng ảo hóa với 2 SFC.

Hình vẽ 2.1 minh họa một bài toán định tuyến trong mạng ảo hóa. Trong hình vẽ là một mạng ảo hóa với 11 nút bao gồm 7 nút vật lý (màu trắng) và 4 nút máy chủ (màu cam) và các thông số về tài nguyên của chúng (ký tự màu đỏ). Một tập các VNF $F = \{f_1, f_2, f_3, f_4\}$ được triển khai ngẫu nhiên trên các nút máy chủ trong mạng vật lý này. Có 2 chuỗi dịch vụ cần được đáp ứng trong mạng là $SFC1$ và $SFC2$. Ví dụ với biểu diễn như trên thì $SF1$ bắt đầu ở nút 1 và kết thúc ở nút 11 và nó phải đi qua 3 VNF theo thứ tự lần lượt f_1, f_3, f_4, ba phần tử đằng sau tương ứng lần lượt là tài nguyên về bộ nhớ, tính toán và băng thông $SFC1$ cần sử dụng. Tương tự thì $SF2$ bắt đầu ở nút 8 và kết thúc ở nút 6 và phải đi lần lượt qua f_3, f_2, f_1. Ngoài ra hai đường màu xanh dương và xanh lá cây là hai đường đi hợp lệ cho $SFC1$ và $SFC2$. Lưu ý, $SFC2$ không thể dụng f_1 tại nút 10 vì không đủ tài nguyên tính toán vì vậy phải đi vòng lên nút 4 trước khi về nút 6.

CHƯƠNG 3. CÁC GIẢI THUẬT TIẾN HÓA ĐA MỤC TIÊU

Hiện nay, các vấn đề về tìm kiếm và tối ưu hóa, đặc biệt là các vấn đề liên quan đến các hàm mục tiêu và ràng buộc phi tuyến, phi lồi và không khả vi, vẫn là một thách thức khó khăn. Cho đến hiện tại, chưa có thuật toán toán học được biết đến để giải quyết các vấn đề này một cách tối ưu. Trong những trường hợp như vậy, việc sử dụng các phương pháp tối ưu hóa meta-heuristic như thuật toán tiến hóa, thuật toán mô phỏng luyện kim (simulated annealing), tìm kiếm tabu (tabu search) và các phương pháp khác được khám phá từ hiện tượng tự nhiên hoặc vật lý đã được áp dụng phổ biến.

Các thuật toán tiến hóa truyền thống thường được sử dụng để giải quyết các vấn đề có một mục tiêu hoặc mục đích duy nhất. Tuy nhiên, hầu hết các vấn đề thực tế đều có nhiều mục tiêu mâu thuẫn tức là tối ưu một mục tiêu sẽ ảnh hưởng đến độ tối ưu của các mục tiêu còn lại. Các phương pháp cổ điển để giải quyết các vấn đề như vậy chủ yếu tập trung vào việc biến đổi nhiều mục tiêu thành một mục tiêu duy nhất, các phương pháp này yêu cầu áp dụng lặp lại một thuật toán để tìm ra nhiều lời giải Pareto tối ưu và đôi khi các ứng dụng như vậy thậm chí không đảm bảo tìm thấy bất kỳ lời giải Pareto tối ưu nào. Ngược lại, các thuật toán tiến hóa đa mục tiêu (Multi-Objective Evolutionary Algorithm - MOEA) là một cách hiệu quả để tìm ra nhiều lời giải Pareto tối ưu cùng một lúc trong một lần mô phỏng duy nhất. Điều này cung cấp cho người ra quyết định một loạt các lựa chọn khả thi, giúp họ đưa ra quyết định thông minh dựa trên sở thích và ưu tiên của mình.

Một lợi ích khác của MOEAs là sự đa dạng của các lời giải tìm được. Bằng cách khám phá các khu vực khác nhau của Pareto front, MOEAs tạo ra một tập hợp đa dạng các lời giải bao gồm nhiều sự cân đối khác nhau giữa các mục tiêu. Điều này giúp người ra quyết định có được cái nhìn toàn diện về không gian lời giải của vấn đề và tạo điều kiện để nhận biết những sự cân đối mới và không trực quan. Thêm vào đó, MOEAs cũng mang lại sự linh hoạt và đa dạng. Các thuật toán MOEAs có thể xử lý các vấn đề có cả biến quyết định liên tục và rời rạc, làm cho thuật toán trên áp dụng được cho một loạt các tình huống thực tế. Ngoài ra, MOEAs cũng có tính linh hoạt và độ ổn định trong việc xử lý không chắc chắn và nhiễu trong hàm mục tiêu. Các thuật toán MOEAs có thể xử lý các hàm mục tiêu không chính xác hoặc hộp đen mà không yêu cầu kiến thức cụ thể về vấn đề. Điều này làm cho MOEAs phù hợp cho các vấn đề thực tế nơi các hàm mục tiêu có thể thay đổi hoặc bị lỗi đo lường. Việc song song hóa và mở rộng cũng là lợi ích đáng chú ý của MOEAs, cho phép khám phá hiệu quả không gian lời giải và mở rộng cho các vấn đề có số chiều

cao. Bằng cách sử dụng tài nguyên tính toán phân tán, MOEAs có thể tăng tốc quá trình tối ưu hóa và giải quyết các vấn đề phức tạp một cách hiệu quả hơn.

Tổng quát, những lợi ích của thuật toán tiến hóa đa mục tiêu làm cho chúng trở thành công cụ quý giá trong việc giải quyết các vấn đề tối ưu hóa với nhiều mục tiêu. Khả năng xử lý xung đột, cung cấp các lời giải đa dạng, phù hợp với các loại vấn đề khác nhau và hỗ trợ quyết định làm cho chúng phù hợp với việc giải quyết những thách thức phức tạp trong thế giới thực.

Do các lý do nêu trên, đồ án sẽ áp dụng bốn thuật toán MOEAs bao gồm: thuật toán di truyền sắp xếp không trội II (Non-dominated Sorting Genetic Algorithm II - NSGA-II) [1], thuật toán tiến hóa dựa trên tiềm năng biên (Pareto 2 Strength Pareto Evolutionary Algorithm - SPEA2)[2], thuật toán tiến hóa đa mục tiêu dựa trên phân rã (Multi-objective Evolutionary Algorithm based on Decomposition - MOEA/D) [3] và tiến hóa đa mục tiêu dựa trên cơ chế dạy học (Multi-Objective Teaching-Learning-Based Optimization - MOTLBO)[4] để giải quyết bài toán MO-TRR.

3.1 Mã hóa cá thể

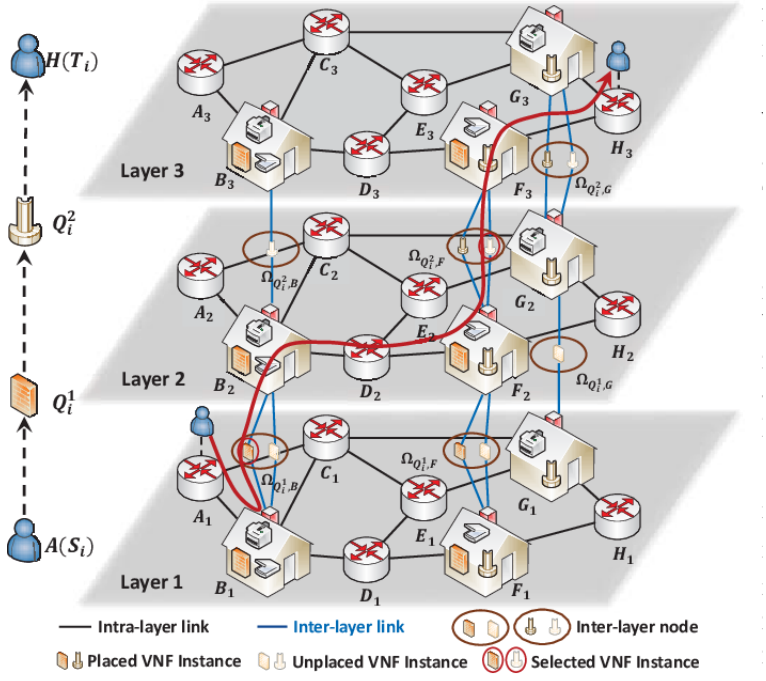
Quá trình mã hóa cá thể đóng vai trò quan trọng trong các thuật toán dựa trên di truyền. Trong bài toán MO-TRR, thứ tự xử lý các chuỗi dịch vụ đóng vai trò quan trọng trong việc phân bổ tài nguyên trong mạng. Ví dụ, nếu đáp ứng một chuỗi dịch vụ có yêu cầu về bộ nhớ hoặc băng thông lớn, tài nguyên mạng còn lại sẽ không đủ để đáp ứng cho các chuỗi dịch vụ còn lại. Do đó, quy trình xây dựng lời giải bao gồm hai bước: i) Quyết định thứ tự được xử lý của các chuỗi dịch vụ; ii) Định tuyến cho từng chuỗi dịch vụ theo trạng thái hiện tại của mạng.

Ở bước đầu tiên, mỗi cá thể được biểu diễn dưới dạng vector là hoán vị của danh sách các chuỗi dịch vụ cần đáp ứng. Giả sử có n chuỗi dịch vụ cần đáp ứng, mỗi cá thể sẽ là một véc tơ n chiều $x = (x_1, x_2, ..., x_n)$, trong đó x_i tương ứng với thứ tự thực hiện của chuỗi dịch vụ thứ i.

Ví dụ một bài toán cần yêu cầu định tuyến 5 chuỗi dịch vụ bao gồm: A, B, C, D, và E. Một cá thể hợp lệ có thể được biểu diễn là $[4, 1, 3, 2, 5]$, có nghĩa là sẽ đáp ứng chuỗi dịch vụ B trước, sau đó lần lượt thực hiện định tuyến các chuỗi dịch vụ D, C, A và cuối cùng là E.

Tiếp theo, m ỗi chuỗi dịch vụ SFC được định tuyến bằng cách xây dựng một đồ thị đa tầng như trong thuật toán [26]. Nhóm tác giả đã đề xuất xây dựng một đồ thị đa tầng bao gồm nhiều bản sao mạng vật lý được và các tầng liên tiếp nhau. Tất cả các cạnh trong cùng một tầng được gọi là cạnh trong tầng. Các cạnh và nút được sử dụng để kết nối các nút máy chủ trong các tầng liền kề được gọi là cạnh giữa các

tầng và nút giữa các tầng. Trong đồ thị đa tầng, các nút giữa các tầng được chọn và sắp xếp theo ràng buộc thứ tự của các VNF mà một chuỗi dịch vụ SFC cần triển khai. Các tài nguyên của các cạnh trong tầng, các nút vật lý và các nút máy chủ trong đồ thị đa tầng bằng với tài nguyên tương ứng của chúng trong đồ thị vật lý. Ngoài ra, băng thông của các cạnh giữa các nút liên tầng và các nút vật lý trong đồ thị đa tầng được đặt là 0.



Hình 3.1: Minh họa một chuỗi dịch vụ SFC trong đồ thị đa tầng [26].

Trong hình vẽ 3.1, một SFC i xuất phát từ đỉnh A và kết thúc tại đỉnh H đi qua lần lượt VNF_a và VNF_b. Trong đồ thị đa tầng, đỉnh bắt đầu và kết thúc của SFC i lần lượt là A_1 và H_3. Giả sử lời giải tối ưu tìm được cho SFC i là đường màu đỏ: $A_1 \rightarrow B_1 \rightarrow \Omega^1_{Q_i^1,B} \rightarrow B_2 \rightarrow D_2 \rightarrow F_2 \rightarrow \Omega^2_{Q_i^2,F} \rightarrow F_3 \rightarrow H_3$, trong đó $\Omega^1_{Q_i^1,B}$ đại diện cho VNF_a đặt ở nút B và $\Omega^2_{Q_i^2,F}$ đại diện cho VNF_b đặt ở nút F. Do đó lời giải tối ưu trong đồ thị gốc là $A \rightarrow B \rightarrow D \rightarrow F \rightarrow H$, trong đó sử dụng VNF_a ở B và VNF_b ở F.

3.2 Các toán tử di truyền

Bước sinh ra quần thể con bằng các toán tử di truyền đóng vai trò quan trọng trong việc khai thác và khai phá không gian tìm kiếm. Hai toán tử di truyền chính thường được sử dụng là phép lai ghép và đột biến.

Lai ghép

Lai ghép được thực hiện trên từng cặp cá thể với xác suất p_c. Với mỗi cặp, giả sử x_1 là cha và x_2 là mẹ, ta lai ghép chúng bằng phép lai ghép thứ tự (Ordered

Crossover). Ý tưởng cơ bản của phép lai ghép thứ tự là trao đổi một phần tử con của cá thể cha mẹ và duy trì thứ tự của các phần tử còn lại. Quá trình lai ghép bắt đầu bằng việc chọn hai điểm cắt ngẫu nhiên trên chuỗi, tạo thành một đoạn con cắt. Cụ thể, phép lai ghép thứ tự thực hiện các bước sau:

- Chọn hai điểm cắt ngẫu nhiên trên chuỗi cha và chuỗi mẹ.
- Sao chép đoạn con cắt từ cha vào vị trí tương ứng trong cá thể con.
- Xóa các phần tử đã được sao chép từ cha khỏi chuỗi mẹ.
- Bắt đầu từ vị trí cuối của đoạn con cắt trong chuỗi mẹ, lần lượt thêm các phần tử còn lại từ chuỗi mẹ vào cá thể con theo thứ tự gốc.
- Lặp lại quá trình trên cho cá thể con kế tiếp.

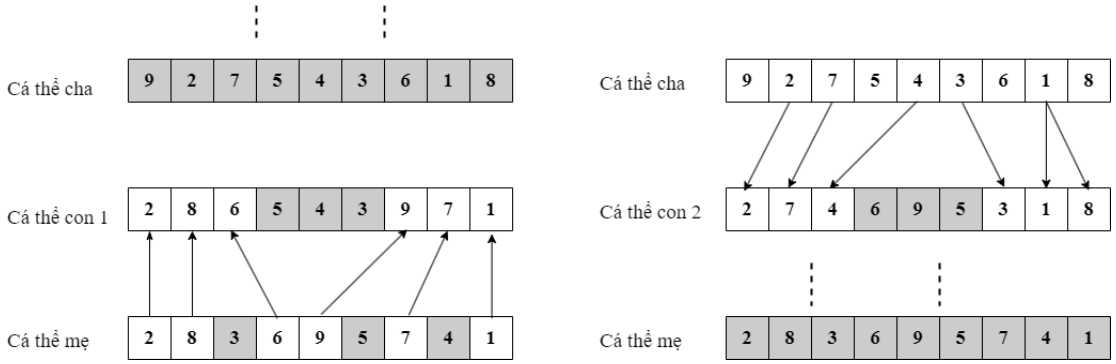
Phép lai ghép thứ tự giúp tạo ra sự đa dạng gen trong quần thể và kết hợp thông tin từ cả hai cha mẹ. Đồng thời, phép lai ghép này duy trì tính chất thứ tự trong chuỗi, giúp bảo toàn các thuộc tính quan trọng của cá thể gốc.

Đột biến

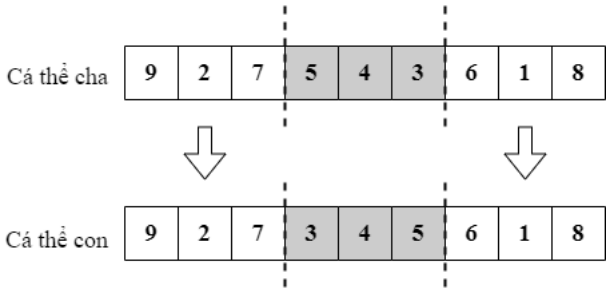
Để đảm bảo độ đa dạng của quần thể, mỗi cá thể con c sinh ra sẽ có xắc suất bị đột biến là p_m. Phép đột biến được chọn là đột biến đảo ngược (Inversion Mutation).Cụ thể, phép đột biến đảo ngược thực hiện các bước sau:

- Chọn một điểm bắt đầu và một điểm kết thúc ngẫu nhiên trong chuỗi.
- Đảo ngược thứ tự của các phần tử trong đoạn con từ điểm bắt đầu đến điểm kết thúc.
- Cập nhật cá thể con bằng chuỗi đã được đột biến.

Hình 3.2 và 3.3 minh họa chi tiết 2 toán tử.



Hình 3.2: Ví dụ minh họa toán tử lai ghép.



Hình 3.3: Ví dụ minh họa toán tử đột biến.

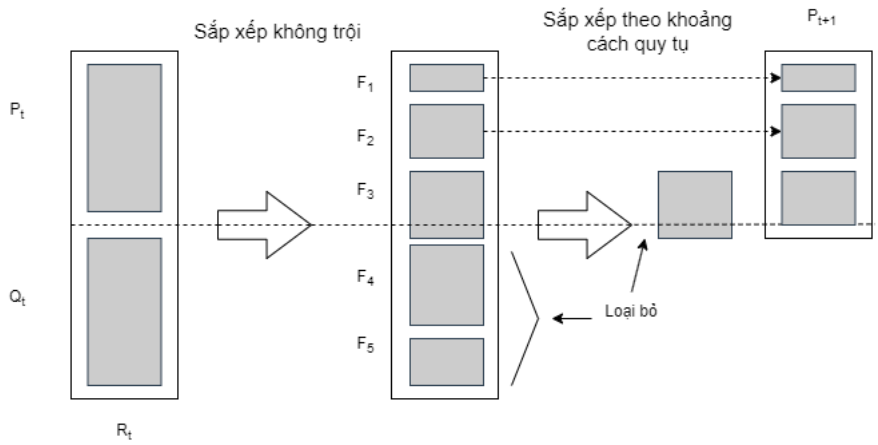
3.3 Các giải thuật tiến hóa đa mục tiêu

3.3.1 Thuật toán di truyền sắp xếp không trội II

Thuật toán di truyền sắp xếp không trội II (Non-dominated Sorting Genetic Algorithm II - NSGA-II) được đề xuất bởi Deb và đồng nghiệp vào năm 2002 [1] để tìm kiếm nhiều lời giải Pareto tối ưu trong một bài toán tối ưu đa mục tiêu (MOP) có ba đặc điểm sau:

1. Sử dụng chọn lọc tinh hoa (elitist selection): Giữ lại các lời giải tốt nhất từ thế hệ cha mà không bị thay thế bởi thế hệ con.
2. Sử dụng cơ chế duy trình độ đa dạng (explicit diversity preserving mechanism): Đảm bảo rằng quần thể duy trì một mức độ đa dạng cao bằng cách đánh giá sự khác biệt giữa các lời giải.
3. Tập trung vào các lời giải không bị trội (non-dominated solutions): Ưu tiên lựa chọn các lời giải không bị trội, tức là không có lời giải khác tốt hơn trên tất cả các mục tiêu.

Trong NSGA-II, quần thể con Q_t được tạo ra bằng cách sử dụng quần thể cha P_t và các toán tử di truyền thông thường giống thuật toán di truyền đã đề cập ở mục trước. Sau đó, hai quần thể được kết hợp để tạo thành quần thể R_t với kích thước $2N$. Sau đó, sắp xếp không trội được sử dụng để phân loại toàn bộ quần thể R_t. Sau khi quá trình kết thúc, quần thể mới được chia thành các biên Pareto có rank khác nhau. Quá trình chọn lọc bắt đầu bằng biên Pareto tốt nhất (có rank là 0) và tiếp



Hình 3.4: Hình vẽ minh họa quá trình chọn lọc trong NSGA-II.

tục bằng các lời giải của biên Pareto thứ hai, tiếp theo là biên Pareto thứ ba, và cứ tiếp tục như vậy. Vì kích thước tổng quát của quần thể R_t là $2N$, không phải tất cả các biên đều có thể được chứa trong N vị trí có sẵn trong quần thể mới. Tất cả các biên không được chọn thì sẽ bị xóa đi. Khi xem xét biên cuối cùng được chấp nhận, có thể tồn tại nhiều lời giải hơn trong biên cuối cùng so với số lượng vị trí còn lại trong quần thể mới. Tình huống này được minh họa trong Hình 3.4. Thay vì loại bỏ ngẫu nhiên một số cá thể từ biên cuối cùng được chấp nhận, những lời giải làm tăng tính đa dạng của quần thể sẽ được chọn. Sơ đồ thuật toán được trình bày trong Thuật toán 1.

Algorithm 1: Thuật toán NSGA-II

Input: Số lượng quần thể ban đầu N, số lượng thế hệ G
Output: Tập hợp các lời giải gần tối ưu
Khởi tạo quần thể ban đầu với N cá thể ngẫu nhiên;
Đánh giá hàm mục tiêu cho từng cá thể trong quần thể;
for $g \rightarrow 1$ **to** G **do**
    Tạo tập hợp con cá thể con bằng cách sử dụng toán tử chéo và lai ghép;
    Đánh giá hàm mục tiêu cho cá thể con;
    Gộp quần thể cha mẹ và quần thể con để tạo thành quần thể mới với kích thước $2N$;
    Áp dụng toán tử chọn lọc dựa trên biên Pareto và khoảng cách quy tụ để chọn ra N cá thể cho thế hệ tiếp theo;
end

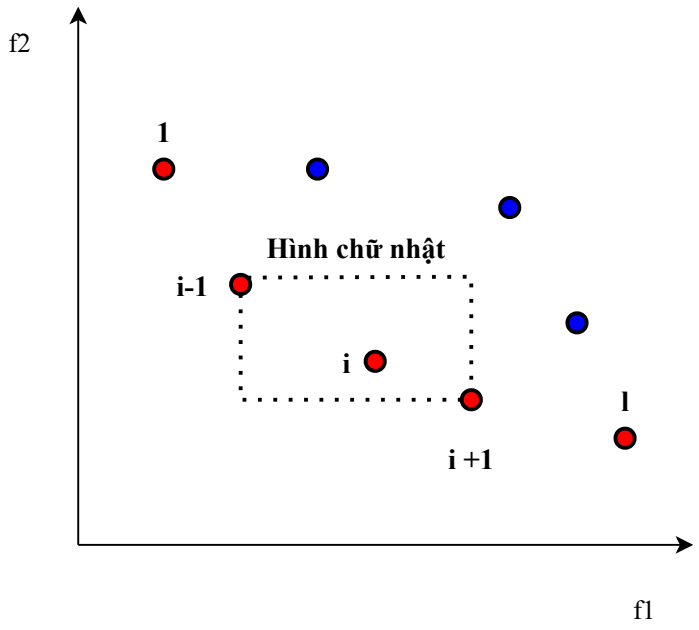
Khoảng cách quy tụ (Crowding distance)

Khoảng cách quy tụ thường được sử dụng để đo khoảng cách giữa các giải pháp láng giềng với nhau trong không gian mục tiêu, nhằm thúc đẩy sự đa dạng và phân tán của các giải pháp. Việc sử dụng khoảng cách quy tụ giúp thuật toán duy trì sự

cân bằng giữa hội tụ (tìm các giải pháp gần với biên Pareto tối ưu) và đa dạng (các lời giải phân bố đều trên biên tối ưu) trong việc tối ưu đa mục tiêu. Khoảng cách quy tụ d_i của một lời giải i là một đo lường của không gian tìm kiếm chuẩn hóa xung quanh i mà không được chiếm bởi bất kỳ lời giải khác trong quần thể. Dựa trên hai thuộc tính này, chúng ta có thể định nghĩa toán tử lựa chọn lời giải như sau:

1. Gọi số lượng lời giải trong F là $l = |F|$. Đối với mỗi i trong tập hợp, gán ban đầu $d_i = 0$.
2. Đối với mỗi hàm mục tiêu $m = 1, 2, ..., M$, sắp xếp tập hợp theo thứ tự tồi hơn của f_m hoặc tìm vector chỉ số đã được sắp xếp: $I^m = sort\,(f_m, >)$
3. Đối với $m = 1, 2, ..., M$, gán một khoảng cách lớn cho các lời giải biên, tức là $d_{I_1^m} = d_{I_l^m} = \infty$ và đối với tất cả các lời giải khác $j = 2$ đến $(l - 1)$, gán

$$d_{I_j^m} = d_{I_j^m} + \frac{f^{(I_{j+1}^m)_m} - f^{(I_{j-1}^m)_m}}{f_m^{max} - f_m^{min}}$$



Hình 3.5: Ví dụ minh họa khoảng cách quy tụ.

Chỉ số I^j đại diện cho chỉ số lời giải thứ j trong danh sách đã được sắp xếp. Do đó, đối với bất kỳ mục tiêu nào, I_1 và I_l đại diện cho giá trị hàm mục tiêu thấp nhất và cao nhất, tương ứng. Thành phần thứ hai ở phía bên phải của phương trình cuối cùng là hiệu giữa giá trị hàm mục tiêu của hai lời giải láng giềng ở hai phía của lời giải Ij. Do đó, chỉ số này đại diện cho một nửa chu vi của khối chữ nhật bao quanh với các lời giải láng giềng được đặt tại các đỉnh của khối chữ nhật (Hình 3.5). Một điểm thú vị là đối với bất kỳ lời giải i nào, hai lời giải $(i + 1)$ và $(i - 1)$ không nhất thiết phải là láng giềng trong tất cả các mục tiêu, đặc biệt là đối với $M \geq 3$. Các

tham số $\overset{max}{m}$ và f_m^{min} có thể được đặt là giá trị lớn nhất và nhỏ nhất trong quần thể của hàm mục tiêu thứ m.

3.3.2 Thuật toán tiến hóa dựa trên tiềm năng biên

Thuật toán tiến hóa dựa trên tiềm năng biên (Pareto 2 Strength Pareto Evolutionary Algorithm - SPEA2) là một phiên bản cải tiến của SPEA [28] được đề xuất vào năm 2001 [2]. SPEA2 sử dụng một chiến lược đánh giá cá thể mới, một kỹ thuật ước lượng mật độ và một phương pháp cắt giảm tập lưu trữ· nâng cao. Sơ đồ của thuật toán được nêu chi tiết trong Thuật toán 2. Về cơ bản thuật toán có cấu trúc khá giống với NSGA-II, tuy nhiên có sự khác nhau trong cách đánh giá cá thể và quá trình chọn lọc cá thể sau mỗi thế hệ.

Đánh giá cá thể

Mỗi cá thể i trong quần thể P và tập lưu trữ A đều có chỉ số $S(i)$, đại diện cho số cá thể nó trội hơn:

$$S(i) = |\{j | j \in (P \cup A), i \ominus j\}|$$

trong đó $|.|$ đại diện số cá thể trong tập hợp, $i \ominus j$ tức cá thể i trội hơn cá thể j. Dựa vào giá trị S, độ thích nghi gốc của cá thể $R(i)$ được tính như sau:

$$R(i) = \sum_{j \in (P \cup A), i \ominus j} S(j)$$

Mật độ $D(i)$ tương ứng với cá thể i được định nghĩa bởi:

$$D(i) = \frac{1}{\sigma_i^k + 2}$$

Trong đó, σ_i^k là phần tử thứ k trong danh sách khoảng cách (trong không gian mục tiêu) từ cá thể i đến tất cả các cá thể j còn lại sau khi được sắp xếp theo thứ tự tăng dần.

Cuối cùng, việc thêm D(i) vào độ thích nghi gốc $R(i)$ để cho ra độ thích nghi cuối cùng của cá thể i:

$$F(i) = R(i) + D(i)$$

Chọn lọc cá thể

Sao chép tất cả các cá thể không bị trội, tức là những cá thể có độ thích nghi nhỏ hơn 1, từ tập lưu trữ và quần thể gốc vào quần thể của thế hệ tiếp theo:

$$P_{t+1} = \{i | i \in P_t \cup A \wedge F(i) < 1\}$$

Nếu biên không bị trội khớp chính xác với quần thể mới $|P_{t+1}| = N$, bước chọn môi trường đã hoàn tất.

Nếu quần thể mới kích thước quá nhỏ $|P_{t+1}| < N$, $N - |P_{t+1}|$ cá thể bị trội tốt

nhất trong tập lưu trữ trước và quần thể gốc được sao chép vào quần thể mới.

Nếu quần thể mới có kích thước quá lớn $|P_{t+1}| > N$, một quy trình cắt giảm được kích hoạt, loại bỏ các cá thể từ P_{t+1} cho đến khi $|P_{t+1}| = N$, tức là cá thể có khoảng cách nhỏ nhất đến cá thể khác được chọn ở mỗi giai đoạn; nếu có nhiều cá thể có khoảng cách nhỏ nhất, thì việc xử lý đồng nhất được thực hiện bằng cách xem xét khoảng cách nhỏ thứ hai và tiếp tục như vậy.

Algorithm 2: Thuật toán SPEA2

Đầu vào: Quần thể P, tập lưu trữ· A
Đầu ra : Các cá thể không bị trội trong A
Khởi tạo quần thể P với các cá thể ngẫu nhiên;
Khởi tạo tập lưu trữ· A trống;
while *điều kiện dừng chưa thỏa mãn* **do**
  Đánh giá độ thích nghi của các cá thể trong P và gán giá trị độ thích nghi và mật độ;
  Cập nhật tập lưu trữ· A bằng cách chọn các cá thể không bị xếp hạng từ P và A;
  Tạo quần thể con trống Q;
  while $|Q| < |P|$ **do**
    Chọn các cá thể cha mẹ từ P và A bằng phương pháp chọn đấu loại nhị phân;
    Thực hiện quá trình lai ghép và đột biến để tạo ra các cá thể con;
    Thêm các cá thể con vào Q;
  end
  Kết hợp P và Q để tạo thành R;
  Thực hiện quá trình lựa chọn môi trường để điền P với các cá thể tốt nhất từ R và A;
end
Trả về các cá thể không bị trội trong A là kết quả cuối cùng;

3.3.3 Thuật toán tiến hóa đa mục tiêu dựa trên phân rã

Nhược điểm chính của các kỹ thuật tiến hóa đa mục tiêu chung là chúng xem xét một bài toán tối ưu đa mục tiêu như một "hộp đen", tức là không sử dụng kiến thức cụ thể về bài toán, điều này có thể gây ra những tìm kiếm không cần thiết và quá trình lai ghép phá hủy, ảnh hưởng tiêu cực đến hiệu suất tổng thể của chúng. Do đó, việc tích hợp kiến thức cụ thể về bài toán trong MOEA để chỉ đạo quá trình tìm kiếm vào các khu vực triển vọng của không gian tìm kiếm có thể mang lại lợi ích. Tuy nhiên, thiết kế các toán tử cụ thể cho toàn bộ một bài toán tối ưu đa mục tiêu (MOP) thường khó khăn. Giải thuật Tiến hóa đa mục tiêu dựa trên phân rã (Multi-objective Evolutionary Algorithm based on Decomposition - MOEA/D) [3] giảm bớt khó khăn này bằng cách phân rã MOP thành nhiều bài toán con thuần

nhất được tối ưu song song, bằng cách sử dụng thông tin hàng xóm và các kỹ thuật đơn vị. Tại mỗi thế hệ, quần thể bao gồm các lời giải tốt nhất được tìm thấy cho đến nay (tức từ khi bắt đầu chạy thuật toán) cho mỗi bài toán con. Mối quan hệ hàng xóm giữa các bài toán con này được xác định dựa trên khoảng cách giữa các vector hệ số tổng hợp của chúng. Các lời giải tối ưu cho hai bài toán con hàng xóm nên rất tương tự nhau. Mỗi bài toán con (tức hàm tổng hợp đơn vị) được tối ưu trong MOEA/D bằng cách sử dụng thông tin chỉ từ các bài toán con hàng xóm của nó. Khung tổng quát của MOEA/D được trình bày như sau [3].

Cho $\lambda_1, ..., \lambda_N$ là một tập các vector trọng số phân bố đều và $z^* = (z_1^*, ..., z_n^*)$ là điểm tham chiếu. Vấn đề xấp xỉ của biên Pareto của MOP có thể được phân rã thành các bài toán tối ưu hóa đơn vị bằng cách sử dụng phương pháp Tchebycheff và hàm mục tiêu của bài toán con thứ j là:

$$g^{te}(x|\lambda^j, z^*) = max_{1 \leq i \leq n}\{\lambda_i^j | f_i(x) - z_i^*\} \tag{3.1}$$

trong đó $z^j = (z_1^j, ..., z_n^j)^T$ là vec tơ trọng số, i.e, $\forall i = 1, ..., n : \lambda_i \geq 0$ và $\sum_{i=1}^{n} \lambda_i = 1$. MOEA/D tối ưu toàn bộ N hàm mục tiêu này đồng thời trong 1 lần chạy. Ở mỗi thế hệ t, MOEA/D với phương pháp Tchebycheff duy trì: một quần thể gồm N điểm $x^1, ..., x^N \in \Omega$, trong đó x^i là lời giải hiện tại cho bài toán con thứ i, $FV^i, ..., FV^N$ trong đó $FV^i = F(x^i) \forall i = 1, ..., N$; và $z = (z_1, ..., z_n)^T$ trong đó z_i là giá trị tốt nhất đã tìm được cho mục tiêu f_i. Ngoài ra, MOEA/D duy trì một quần thể ngoài (external population - EP), dùng như một kho lưu trữ những cá thể không bị trội trong suốt quá trình tìm kiếm. Khung cơ bản của thuật toán có thể tìm thấy ở trong Thuật toán 3.

Algorithm 3: Thuật toán MOEA/D

Input: N: số lượng bài toán con, T: số lượng thế hệ, PopSize: kích thước quần thể, λ: ma trận N x n chứa các vector trọng số, z^*: điểm tham chiếu, Problem: bài toán tối ưu hóa đa mục tiêu

Output: P: quần thể tốt nhất, EP: external population

```
Khởi tạo quần thể P kích thước PopSize với các lời giải ngẫu nhiên;
Khởi tạo external population EP rỗng;
for t = 1 to T do
    for i = 1 to N do
        Lựa chọn lân cận cho sub-problem i dựa trên λ;
        Tính các hệ số phân bố (fitness) cho các lời giải trong lân cận;
        Chọn các lời giải cha để tiến hóa (parent selection);
        Áp dụng các toán tử tiến hóa (crossover, mutation) để tạo ra con cái;
        Thêm con cái vào quần thể P;
    end
    Cập nhật EP bằng cách thêm các lời giải nondominated từ P;
    Cắt tỉa EP nếu kích thước vượt quá kích thước tối đa;
end
```

3.3.4 Thuật toán tiến hóa đa mục tiêu dựa trên cơ chế dạy học

Thuật toán tiến hóa đa mục tiêu dựa trên cơ chế dạy học (Multi-Objective Teaching-Learning Based Optimization -MOTLBO [4]) là một thuật toán tối ưu đa mục tiêu mở rộng từ thuật toán TLBO. Nó kết hợp quá trình giảng dạy và học tập với các kỹ thuật tối ưu để tìm ra một tập hợp các lời giải biểu thị sự cân đối giữa nhiều mục tiêu xung đột. Phương pháp TLBO chia quá trình tối ưu thành hai giai đoạn chính: Giai đoạn giảng dạy (Teacher Phase) và giai đoạn học từ học sinh (Learner Phase).Sơ đồ của thuật toán được trình bày cụ thể trong Thuật toán 4

Giai đoạn giảng dạy (Teacher Phase)

Vector trung bình M được tính toán dựa trên giá trị trung bình của các giá trị của học sinh cho mỗi biến thiết kế.

Giáo viên, được đại diện bởi học sinh tốt nhất, cố gắng di chuyển giá trị trung bình gần về giá trị tối ưu. Giá trị trung bình mới M' được tính toán theo công thức sau:

$$M' = M + TF * (rand() - 0.5) * (M - Teacher)$$

trong đó TF là một yếu tố giảng dạy được chọn ngẫu nhiên có giá trị là 1 hoặc 2, và rand() sinh ra một số ngẫu nhiên trong khoảng [0, 1]. Các lời giải của học sinh được cập nhật bằng công thức:

$$Learner' = Learner + rand() * (M' - Learner)$$

trong đó Learner đại diện cho lời giải của học sinh.

Giai đoạn học từ học sinh (Learner Phase)

Mỗi học sinh tương tác ngẫu nhiên với một học sinh khác, và nếu học sinh kia có lời giải tốt hơn, học sinh hiện tại sẽ chấp nhận lời giải đó.

Công thức cập nhật cho học sinh trong giai đoạn học sinh là:

$$Learner' = Learner + rand() * (OtherLearner - Learner)$$

trong đó OtherLearner đại diện cho lời giải của học sinh khác.

Algorithm 4: Thuật toán MOTLBO

Input: Kích thước quần thể N, Số thế hệ G
Output: Tập hợp các lời giải không bị trội
Khởi tạo: Tạo ngẫu nhiên quần thể ban đầu có kích thước N;
for $g = 1$ **to** G **do**
  Giai đoạn Giáo viên:;
  Tính vector trung bình M là trung bình của các lời giải của các học sinh;
  Chọn học sinh tốt nhất làm giáo viên T;
  Cập nhật vector trung bình M' bằng công thức
   $M' = M + TF \cdot (rand() - 0.5) \cdot (M - T)$;
  Giai đoạn Học sinh:;
  for *mỗi học sinh* H **do**
    Chọn một học sinh khác ngẫu nhiên H';
    Cập nhật lời giải của học sinh H' bằng công thức
     $H' = H + rand() \cdot (M' - H)$;
    Đánh giá và so sánh giá trị tối ưu của H' với H;
    Lưu giữ lời giải tốt hơn giữa H và H';
  end
  Áp dụng phân loại không bị trội để chọn ra những học sinh tốt nhất dựa trên giá trị tối ưu của chúng;
end
Chọn các lời giải không bị trội là tập hợp lời giải cuối cùng;

CHƯƠNG 4. ĐÁNH GIÁ THỰC NGHIỆM

Chương này trình bày các thực nghiệm đánh giá hiệu quả của các thuật toán tiến hóa đa mục trong việc giải quyết các bài toán đa mục tiêu định tuyến trong mạng ảo hóa. Các thuật toán được thực hiện trên các bộ dữ liệu với kịch bản khác nhau để chứng minh tính hiệu quả của các thuật toán tiến hoá đa mục tiêu.

4.1 Dữ liệu thực nghiệm

Để kiểm tra sự hiệu quả các giải thuật, đồ án này sử dụng dữ liệu thực nghiệm giống trong nghiên cứu [29]. Bộ dữ liệu được xây dựng dựa trên 3 mô hình kiến trúc mạng trong thực tế để xây dựng mạng vật lý bao gồm: NSF (14 đỉnh, 21 cạnh) [8], CONUS (75 đỉnh, 99 cạnh) [15], COGENT (104 đỉnh, 116 cạnh) [24]. Với mỗi kiến trúc mạng trong ba mạng vật lý nói trên, các nút mạng sẽ được chọn dựa trên bốn kịch bản khác nhau như sau:

1. Centers: Trong kịch bản này giả định giống như kiến trúc mạng trung tâm tức là chỉ một vài trung tâm dữ liệu lớn có thể xử lý dịch vụ cho các yêu cầu trong mạng. Do đó, chỉ 10% nút có bậc cao nhất được chọn là nút mạng, các nút còn lại đều là nút vật lý.
2. Uniform: 30% số nút ở trong mạng được chọn một cách ngẫu nhiên để làm nút mạng. Sau đó các VNFs được đặt một cách ngẫu nhiên lên các nút mạng này. Mỗi nút mạng có từ 2 đến 3 VNF.
3. Rural: Trong kịch bản này, VNFs được giả định rằng chỉ được triển khai ở các khu vực nông thôn và ngoại thành. Điều này có nghĩa là các nút mạng sẽ là 30% nút có bậc thấp nhất trong mạng vật lý. Ngoài ra, các VNF cũng phân phối thưa nên mỗi nút mạng chỉ chứa 1 VNF.
4. Urban: Ngược lại với kịch bản Rural, kịch bản này sẽ chọn 30% nút có bậc cao nhất để làm nút mạng.

Mỗi bộ dữ liệu sẽ được ký hiệu x_y_z, trong đó x là tên của kiến trúc mạng ($x \in$ [COGENT, CONUS, NSF]), y là hai chữ cái đầu trong tên loại kịch bản ($y \in$ [ce, un, ru, ur]) và z là số lượng chuỗi dịch vụ cần đáp ứng. Ví dụ nsf_ur_30 để chỉ tới bộ dữ liệu sử dụng kiến trúc mạng NSF, kịch bản Urban và cần đáp ứng 30 chuỗi dịch vụ. Trong mỗi bộ dữ liệu, tài nguyên về bộ nhớ của nút vật lý, tài nguyên tính toán của nút mạng và băng thông tối đa đều được đặt là 100. Để sinh ra một chuỗi dịch vụ thì đầu tiên sẽ sinh ra tập các chức năng mạng ảo F, bao gồm 10 VNF riêng biệt. Mỗi SFC được xây dựng bằng cách chọn lần lượt một cách ngẫu nhiên các VNF ở trong F, trong khi các nút bắt đầu và kết thúc được chọn ngẫu nhiên

trong các nút vật lý ở trong mạng. Tài nguyên về bộ nhớ α, tài nguyên về khả năng tính toán β và lượng băng thông cần thiết để chuyển tiếp dữ liệu γ được sinh ngẫu nhiên trong khoảng $[1, 10]$.

4.2 Các tiêu chí đánh giá

Để có thể so sánh và đánh giá hiệu năng của các thuật toán tiến hóa đa mục tiêu, có nhiều độ đo được đề xuất để giải quyết vấn đề này. Chỉ sử dụng một độ đo đơn thuần không thể phản ánh được chất lượng lời giảm bao gồm độ hội tụ và độ đa dạng mà các thuật toán tìm được. Vì vậy, trong nghiên cứu này, ba độ đo đã được sử dụng để đánh giá tổng thể hiệu suất của các thuật toán như sau.

1. Độ đo số cá thể (NDS metric - [30], [31]): đánh giá số lượng lời giải trong biên tối ưu mà lời giải tìm được. Số lượng lời giải tìm được càng lớn cho thấy rằng thuật toán đó có khả năng tìm kiếm đa dạng.

   $$n_s = |A|$$

   trong đó A là tập lời giải không bị trội tìm được.

2. Độ đo bao phủ (Convergence metric - [1],[32]): đánh giá mức độ hội tụ của biên Pareto tìm được. Độ đo này so sánh tập hai lời giải của hai thuật toán khác nhau và tính tỷ lệ lời giải trong tập hợp thứ hai mà có ít nhất một lời giải tốt hơn hoặc bằng trên mỗi mục tiêu so với tập hợp thứ nhất. Cụ thể độ đo được định nghĩa như sau: Cho A, B là tập lời giải của hai thuật toán đang xét.

   $$C(A, B) = |b \in B | \exists a \in A : a \ominus b|$$

   Giá trị của $C(A, B) = 1$ nghĩa là tất cả các lời giải trong B bị trội bởi ít nhất một lời giải trong A. Ngược lại, nếu $C(A, B) = 0$ đại diện trường hợp mà không lời giải nào trong B bị trội bởi A. Tuy nhiên, $C(A, B)$ không nhất thiết phải bằng $1 - C(B, A)$ nên cần phải xét cả hai thông số.

3. Độ đo Hypervolume (S measure - [33]): đo lường vùng không gian mục tiêu bị trội bởi tập các lời giải tìm được và điểm Nadir. Độ đo này thường được sử dụng vì có thể đánh giá đồng thời đội hội tụ và độ đa dạng của biên Pareto. Hypervolume cũng có các tính chất toán học tốt hơn nhiều so với các độ đo khác; mặc dù tính toán chính xác giá trị của hypervolume khó khăn, nhưng đã được đề xuất nhiều thuật toán nhanh để có được một giá trị gần đúng. Ngoài ra, người ta đã chứng minh rằng hypervolume đạt giá trị tối đa khi và chỉ khi tập hợp các lời giải chỉ chứa các điểm Pareto tối ưu. Ví dụ minh họa về độ đo này được biểu diễn ở hình.

4.3 Tham số cài đặt thực nghiệm

Tất cả các thuật toán được cài đặt bằng ngôn ngữ Python 3 và được chạy 10 lần độc lập trên máy tính có cấu hình như sau:System: Windows 10, CPU: Intel(R) Core(TM) i5-7500 3.4GHz, Main memory: 8GB. Tham số của các thuật toán được cài đặt như trong bảng 4.1.

Bảng 4.1: Bảng tham số các thuật toán.

Tham số	NSGA2	SPEA2	MOEA/D	MOTLBO
Kích thước quần thể	100	100	-	100
Số thế hệ	100	100	100	100
Chọn lọc giao đấu	2	2	-	-
Tỉ lệ lai ghép	0.9	0.9	0.9	-
Tỉ lệ đột biến	0.1	0.1	0.1	-
Số bài toán con	-	-	100	-
Số hàng xóm	-	-	3	-

4.4 Kết quả thực nghiệm

4.4.1 So sánh độ hiệu quả của các thuật toán tối ưu đa mục tiêu

Để có thể đánh giá được độ hiệu quả của bốn thuật toán trên các bộ dữ liệu khác nhau ba độ đo được nêu ra ở phần trên đã được sử dụng. Bảng 4.2 so sánh số lượng cá thể trên biên của bốn thuật toán trong từng kịch bản. Kết quả thực nghiệm cho thấy rằng giá trị độ đo số cá thể của NSGA2 và SPEA2 xấp xỉ nhau trong hầu hết các bộ dữ liệu và lớn hơn giá trị của hai giải thuật còn lại. Cụ thể, trên các bộ dữ liệu có kiến trúc mạng COGENT, NSGA2 và SPEA2 tìm được khoảng 5 cá thể trên biên, trong khi hai thuật toán còn lại chỉ tìm được trung bình 4 điểm. Tương tự như vậy, đối với 2 kiến trúc mạng CONUS và NSF, số lượng cá thể tìm được của 2 thuật toán NSGA2 và SPEA2 khoảng 3 và 6, trong khi kết quả của các thuật toán MOEA/D và MOTLBO chỉ khoảng 2 và 5. Tóm lại, có thể kết luận rằng SPEA2 và NSGA2 có khả năng tìm nhiều hơn 1 lời giải so với hai thuật toán còn lại.

Đồ án cũng so sánh kết quả thu được của các thuật toán dựa trên độ đo HV. Bảng 4.3 cung cấp giá trị trung bình của độ đo Hypervolume sau 10 lần chạy của mỗi thuật toán trên các kịch bản thực nghiệm khác nhau. Độ đo này thường được sử dụng vì nó cho thấy chất lượng của lời giải cả về độ đa dạng và độ hội tụ. Giá trị độ đo Hypervolume càng cao thì thuật toán càng tốt. Nhìn chung không có sự sai khác nhiều về kết quả của 4 thuật toán. Tuy nhiên, 2 thuật toán SPEA2 và NSGA2 vẫn cho kết quả tốt hơn. Dựa trên các giá trị trong bảng 4.3, kết quả trung bình của thuật toán NSGA2 và SPEA2 trên bộ dữ liệu có kiến trúc mạng COGENT là 0.43, giá trị này của thuật toán MOEA/D và MOTLBO lần lượt là 0.422 và 0.390. Trong

các bộ dữ liệu với kiến trúc mạng CONUS, thuật toán NSGA2 vẫn cho kết quả tốt nhất với giá trị trung bình 0.487, nhiều hơn 0.076 so với kết quả của MOTLBO. Tương tự như vậy, quy luật này cũng không có sự thay đổi đáng kể trong bộ dữ liệu có kiến trúc mạng NSF: kết quả của NSGA2 và SPEA2 là 0.328 và 0.338, trong khi kết quả của MOEA/D và MOTBLO là 0.327 và 0.287. Kết quả của thuật toán MOEA/D hơi thấp hơn so với hai thuật toán trên dù thuật toán này thường được chứng minh tốt hơn hai thuật thuật toán trên trong các nghiên cứu gần đây. Nguyên nhân có thể là do trong thuật toán MOEA/D có khá nhiều hyper-parameter (ví dụ như số hàng xóm, số bài toán con, ...) nên việc chọn các tham số này sẽ ảnh hưởng độ hiệu quả của thuật toán. Chọn các tham số lớn có thể dẫn đến kết quả tốt tuy nhiên sẽ đánh đổi bằng độ phức tạp cao. Ngoài ra giá trị độ đo Hypervolume cho thuật toán MOTLBO là thấp nhất trong toàn bộ tập dữ liệu. Điều này có thể dự đoán trước do MOTLBO được thiết kế cho các bài toán tối ưu liên tục nên khi áp dụng cho bài toán tối ưu tổ hợp thì thuật toán sẽ có độ hiệu quả thấp hơn so với ba giải thuật còn lại.

Bảng 4.2: So sánh độ đo số cá thể giữa các giải thuật MOEAs.

Kiến trúc mạng	Kịch bản	NSGA2	SPEA2	MOEA/D	MOTLBO
COGENT	Centers	**2.87**	**2.87**	2.67	2.90
	Rural	5.43	**5.55**	4.95	5.00
	Uniform	**5.93**	5.47	4.83	4.63
	Urban	4.87	**4.93**	4.65	4.08
	Trung bình	**4.78**	4.71	4.28	4.15
CONUS	Centers	**3.13**	2.93	2.79	2.34
	Rural	**4.01**	3.93	3.53	2.82
	Uniform	**2.86**	2.81	2.51	2.34
	Urban	**2.52**	2.47	2.31	1.98
	Trung bình	**3.13**	3.03	2.79	2.37
NSF	Centers	7.07	**7.15**	6.65	5.82
	Rural	**6.03**	5.74	5.18	4.38
	Uniform	5.03	**5.13**	4.79	3.84
	Urban	4.83	**5.74**	4.98	4.32
	Trung bình	5.74	**5.94**	5.40	4.59

Ngoài ra, kết quả thu được từ bốn thuật toán trong các bộ dữ liệu cũng được đánh giá theo cặp bằng độ đo bao phủ. Kết quả được đưa ra trong bảng 4.4. Độ đo này đánh giá mức độ không bị trội của tập lời giải tìm được bởi hai thuật toán. Từ bảng 4.4 có thể thấy SPEA2 tốt hơn NSGA2 ở bộ dữ liệu có kiến trúc mạng NSF nhưng tệ hơn trong 2 kiến trúc mạng còn lại. Ngoài ra, ba thuật toán NSGA2, SPEA2, MOEA/D hoàn toàn vượt trội khi so với MOTLBO. Tổng hợp kết quả thì có thể nói rằng NSGA2, SPEA2 tốt hơn hai thuật toán còn lại.

Bảng 4.3: So sánh bốn thuật toán MOEAs theo độ đo HV.

Kiến trúc mạng	Kịch bản	NSGA2	SPEA2	MOEA/D	MOTLBO
COGENT	Centers	**0.400**	**0.400**	0.391	0.375
	Rural	**0.397**	0.396	0.390	0.382
	Uniform	**0.449**	0.445	0.434	0.384
	Urban	**0.480**	0.479	0.472	0.417
	Trung bình	**0.431**	0.430	0.422	0.390
CONUS	Centers	**0.459**	0.458	0.446	0.387
	Rural	0.470	**0.473**	0.458	0.397
	Uniform	**0.483**	0.474	0.467	0.402
	Urban	**0.537**	0.536	0.526	0.460
	Trung bình	**0.487**	0.485	0.474	0.411
NSF	Centers	**0.348**	0.347	0.342	0.304
	Rural	**0.365**	0.363	0.350	0.304
	Uniform	**0.309**	0.308	0.295	0.258
	Urban	0.289	**0.333**	0.320	0.282
	Trung bình	0.328	**0.338**	0.327	0.287

Bảng 4.4: So sánh các thuật toán MOEAs sử dụng độ đo bao phủ.

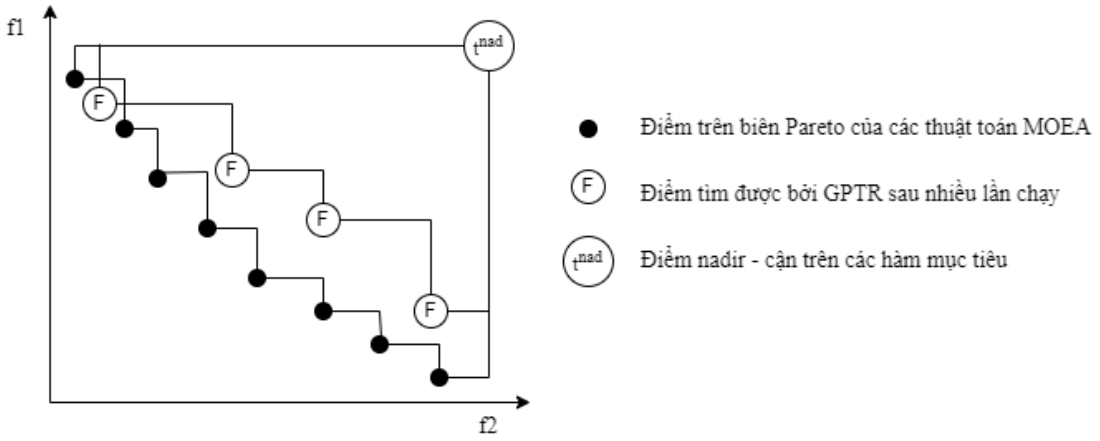
Kiến trúc mạng	Độ đo bao phủ				
	Thuật toán	NSGA2	SPEA2	MOEA/D	MOTLBO
COGENT	NSGA2	-	0.21	0.19	0.04
	SPEA2	0.25	-	0.19	0.04
	MOEA/D	0.31	0.35	-	0.06
	MOTLBO	0.66	0.68	0.63	**-**
CONUS	NSGA2	-	0.31	0.28	0.03
	SPEA2	0.41	-	0.36	0.06
	MOEA/D	0.47	0.4	-	0.12
	MOTLBO	0.85	0.82	0.76	**-**
NSF	NSGA2	-	0.28	0.24	0.08
	SPEA2	0.24	-	0.24	0.07
	MOEA/D	0.31	0.33	-	0.1
	MOTLBO	0.68	0.69	0.64	**-**

4.4.2 So sánh độ hiệu quả của các thuật toán tối ưu đa mục tiêu với thuật toán tối ưu đơn mục tiêu

Phần này tiến hành so sánh các thuật toán MOEAs với thuật toán trong nghiên cứu [29] - một phương pháp khác cũng được đề xuất cho cùng mô hình. Các tác giả trong nghiên cứu [29] đã sử dụng giải thuật lập trình di truyền có tên gọi GPTR để đưa ra một cách định tuyến hiệu quả. Tuy nhiên, GPTR đưa bài toán MO-TRR về bài toán đơn mục tiêu bằng cách sử dụng vec tơ trọng số. Do đó để có thể so sánh các thuật toán MOEAs với GPTR, phương pháp SOGA-3 trong nghiên cứu

[34] được sử dụng. Chọn $d = 20$, phương pháp này tạo 21 vec tơ trọng số bao gồm $(0, 1), (0.05, 0.95), ..., (1, 0)$ từ đó cho ra tối ra 21 lời giải khác nhau để hợp lại thành 1 biên Pareto. Biên này sẽ được đem đi so sánh với biên tìm được bằng các thuật toán MOEAs như hình 4.1. Để việc so sánh dễ dàng, độ đo mới là "số lượng cải thiện" δ chính là số bộ dữ liệu mà trong đó thuật toán A trả về giá trị hypervolume tốt hơn B sẽ được sử dụng.

$$\delta(A, B) = \sum_{i=1}^{no.instances} (HV(A, r) > HV(B, r))$$



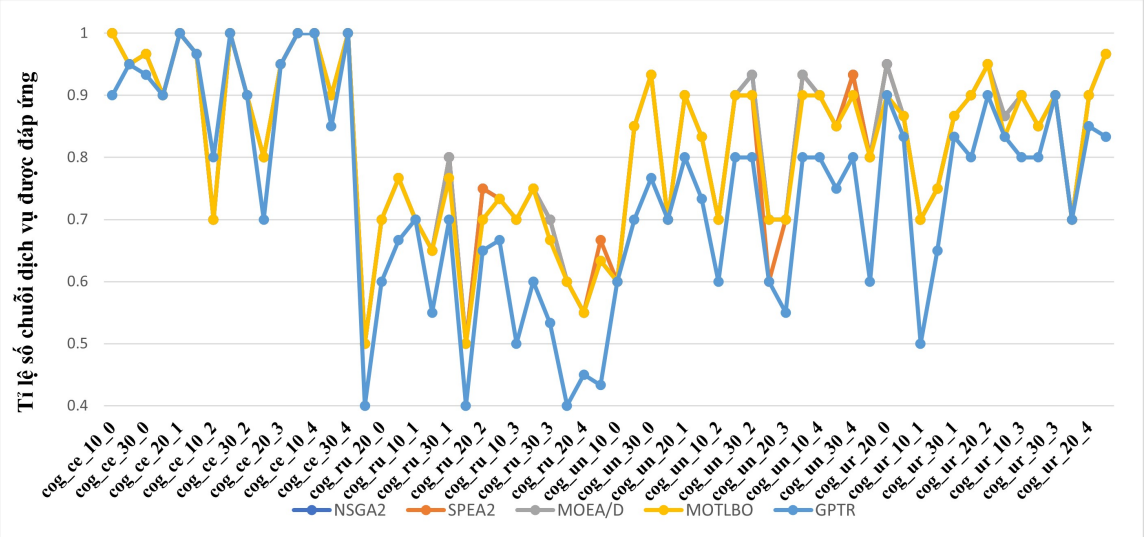
Hình 4.1: So sánh đơn mục tiêu với đa mục tiêu bằng hypervolume.

Kết quả thu được ở bảng 4.5. Như vậy kết quả thu được các thuật toán MOEAs hoàn toàn vượt trội so với GPTR về độ đa dạng của biên Pareto tìm được. Hầu hết với các bộ dữ liệu, khi sử dụng thuật toán GPTR mặc dù với các bộ vec tơ trọng số khác nhau nhưng vẫn chỉ tìm được 1 lời giải duy nhất, điều này nghĩa là việc áp dụng các thuật toán MOEAs có thể tìm được nhiều lời giải hơn để người quyết định có thể lựa chọn.

Bảng 4.5: So sánh độ đo δ giữa các thuật toán MOEAs và GPTR.

Kiến trúc mạng	Kịch bản	Số bộ dữ liệu	NSGA2	SPEA2	MOEA/D	MOTLBO
COGENT	Centers	15	15	15	15	15
	Rural	15	15	15	15	15
	Uniform	15	15	15	15	15
	Urban	15	15	15	15	15
CONUS	Centers	15	15	15	15	15
	Rural	15	15	15	15	15
	Uniform	15	13	14	13	12
	Urban	15	15	15	15	15
NSF	Centers	15	15	15	15	15
	Rural	15	15	15	15	15
	Uniform	15	15	15	15	15
	Urban	15	15	15	15	15

Ngoài ra đồ án lần lượt so sánh hai mục tiêu bao gồm tỉ lệ số chuỗi dịch vụ được đáp ứng và tỉ lệ sử dụng tài nguyên mạng của các thuật toán MOEAs và thuật toán GPTR.



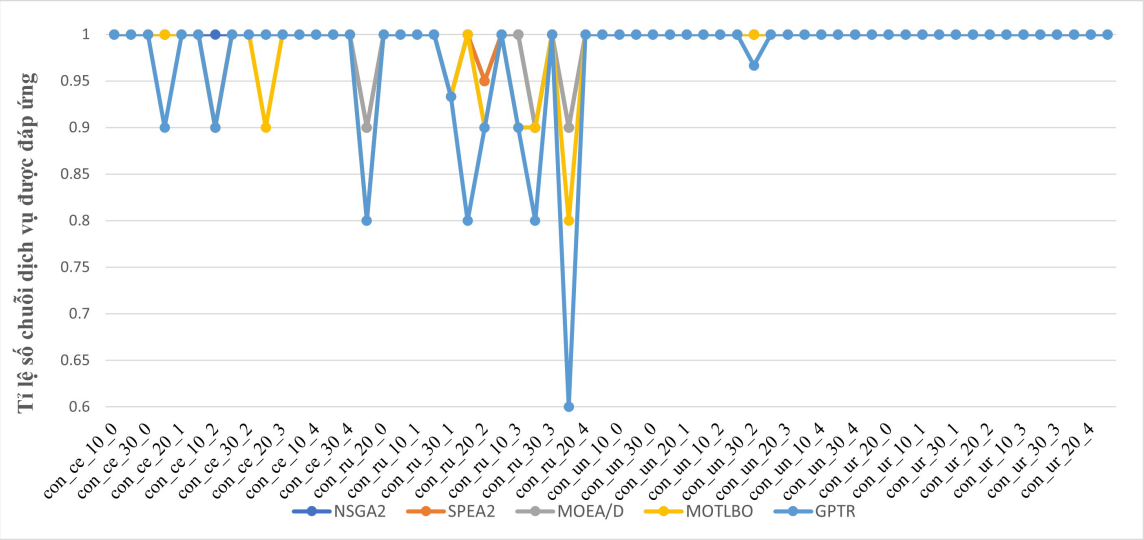
Hình 4.2: Tỉ lệ số chuỗi dịch vụ được đáp ứng ở kiến trúc mạng COGENT.

Đầu tiên, đồ án so sánh tỉ lệ chuỗi dịch vụ được đáp ứng của các thuật toán MOEAs với thuật toán GPTR trên các kiến trúc mạng khác nhau trong hình 4.2, 4.3 và 4.4. Dễ thấy rằng các thuật toán MOEAs cho kết quả vượt trội hơn thuật toán GPTR trong những bộ dữ liệu đang xét.

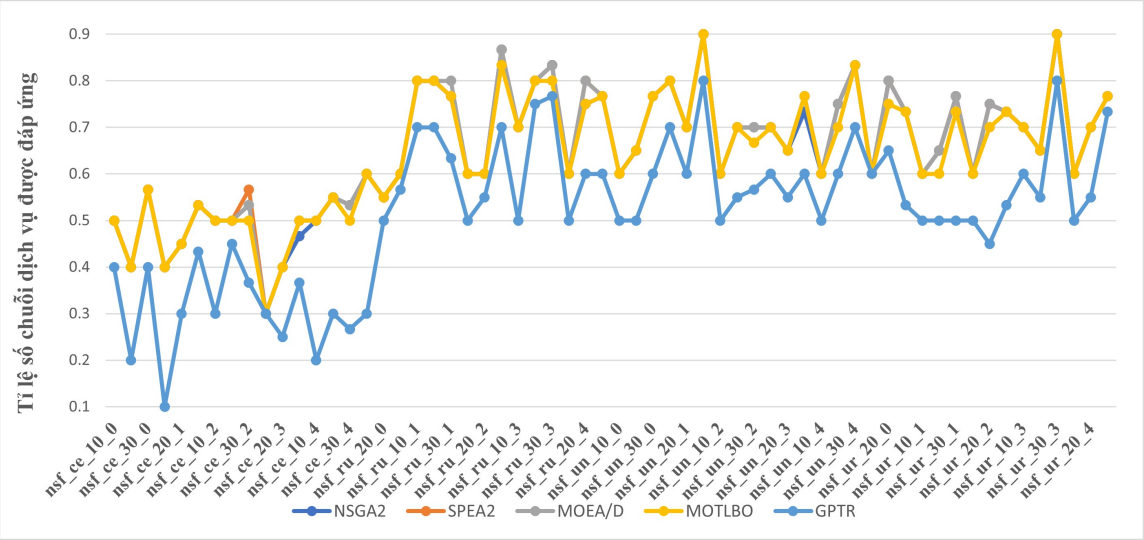
Trong hình 4.2, có thể thấy NSGA2, SPEA2 và MOEA/D là ba thuật toán hoạt động tốt nhất. Cụ thể, kết quả trung bình của bốn thuật toán trong kịch bản Centers đều là 0.935 trong khi kết quả của GPTR là 0.923. Trong các kịch bản Rural, Uniform và Urban kết quả trung bình tốt nhất mà các thuật toán MOEAs đạt được lần lượt là 0.671, 0.824 và 0.858 so với kết quả thu được của thuật toán GPTR là 0.55, 0.72 và 0.782.

Kết quả của thuật toán GPTR trong kiến trúc mạng CONUS trong hình 4.3 lần lượt là 0.987, 0.915, 0.998, 1. Với các thuật toán MOEAs, các giá trị tốt nhất lên tới 1, 0.972, 1, 1. Điều này cho thấy các thuật toán MOEAs hiệu quả với mọi kịch bản với kiến trúc mạng CONUS.

Cuối cùng, với bộ dữ liệu với kiến trúc mạng NSF, kết quả của các thuật toán MOEAs đều lớn hơn 0.473 với kịch bản Centers, so với GPTR là 0.308. Trong các kịch bản còn lại tỉ lệ số chuỗi dịch vụ được chấp nhận của NSGA2, SPEA2 và MOEA/D, MOTLBO đều lớn hớn 0.7 so với kết quả xấp xỉ 0.59 của thuật toán GPTR.



Hình 4.3: Tỉ lệ số chuỗi dịch vụ được đáp ứng ở kiến trúc mạng CONUS.



Hình 4.4: Tỉ lệ số chuỗi dịch vụ được đáp ứng ở kiến trúc mạng NSF.

Tiếp theo, các thuật toán được so sánh theo mục tiêu thứ hai là tỉ lệ sử dụng tài nguyên mạng trong bảng 4.6. Có thể thấy trong hầu hết kết quả của các thuật toán MOEAs gần như tương đương nhau và vượt trội hơn hẳn GPTR. Cụ thể trong bộ dữ liệu có kiến trúc mạng COGENT, tỉ lệ sử dụng năng lượng trung bình của NSGA2 và SPEA2 đều xấp xỉ 0.399, nhiều hơn 0.15 so với kết quả của GPTR. Tương tự như vậy, đối với các bộ dữ liệu có kiến trúc mạng CONUS và NSF, tỉ lệ sử dụng năng lượng khoảng 0.355 và 0.399. Trong khi tỉ lệ sử dụng năng lượng của GPTR trong các bộ dữ liệu này chỉ được trung bình 0.229 và 0.189.

Bảng 4.6: So sánh tỉ lệ sử dụng năng lượng trong các bộ dữ liệu khác nhau.

Kiến trúc mạng	Kịch bản	NSGA2	SPEA2	MOEA/D	MOTLBO	GPTR
COGENT	Centers	**0.303**	0.301	0.294	0.283	0.131
	Rural	**0.445**	0.441	0.441	0.430	0.311
	Uniform	**0.410**	0.407	0.399	0.384	0.256
	Urban	0.420	**0.421**	0.416	0.404	0.276
	Trung bình	**0.395**	0.393	0.388	0.375	0.244
CONUS	Centers	**0.332**	0.331	0.326	0.309	0.195
	Rural	**0.356**	0.355	0.350	0.335	0.197
	Uniform	**0.339**	**0.339**	0.337	0.314	0.242
	Urban	**0.394**	0.390	0.390	0.371	0.280
	Trung bình	**0.355**	0.354	0.351	0.332	0.229
NSF	Centers	**0.561**	**0.561**	0.556	0.544	0.381
	Rural	0.382	**0.385**	0.376	0.353	0.147
	Uniform	**0.304**	0.300	0.295	0.282	0.096
	Urban	**0.349**	**0.349**	0.333	0.325	0.133
	Trung bình	**0.399**	**0.399**	0.390	0.376	0.189

KẾT LUẬN

Bài toán phân bổ tài nguyên trong mạng ảo hóa đang thu hút sự quan tâm ngày càng nhiều từ nhiều nhà nghiên cứu. Các phương pháp được nghiên cứu hiện tại dường như chỉ xem xét số lượng chuỗi dịch được chấp nhận là mục tiêu duy nhất cần tối ưu mà ít quan tâm đến những khía cạnh quan trọng khác, ví dụ như cân bằng tải. Đồ án đã tóm tắt những kiến thức cơ bản nhất về ảo hóa chức năng mạng NFV đồng thời khảo sát một số nghiên cứu trước đó về vấn đề định tuyến trong mạng ảo hóa. Thêm vào đó, đồ án cũng nghiên cứu một cách khái quát lớp thuật toán tiến hóa bầy đàn nói chung và các giải thuật đa mục tiêu nói riêng.

Từ những kiến thức đã được tiếp thu, đồ án đã đưa ra mô hình bài toán đa mục tiêu MO-TRR nhằm tối ưu hai mục tiêu xugn đột là tỉ lệ số chuỗi dịch vụ được đáp ứng và khả năng sử dụng tài nguyên trong mạng ảo hóa. Tiếp theo, đồ án tiến hành áp dụng bốn giải thuật tiến hóa đa mục tiêu khác nhau bao gồm NSGA2, SPEA2, MOEA/D và MOTLBO nhằm giải quyết bài toán nói trên. Đồ án cũng tiến hành so sánh kết quả với giải thuật đơn mục tiêu GPTR mới được đề xuất trong nghiên cứu [29]. Sau khi tiến hành các thực nghiệm theo nhiều kịch bản khác nhau và phân tích các kết quả đạt được, giải thuật NSGA2 cho kết quả tốt nhất trong bốn thuật toán và các thuật toán tiến hóa bầy đàn có khả năng tìm kiếm các lời giải đa dạng hơn giúp cho người quyết định có nhiều lựa chọn.

Tuy nhiên đồ án vẫn chỉ mang tính chất nghiên cứu các thuật toán tiến hóa đa mục tiêu và áp dụng các thuật toán tiến hóa đa mục tiêu để giải bài toán tối ưu. Trong tương lai, tác giả sẽ dự định thử nghiệm các các cách kết hợp và tìm ra thuật toán định tuyến có hiệu suất cao để làm giảm độ phức tạp tính toán và nâng cao chất lượng lời giải. Ngoài ra, tác giả cũng dự định giải quyết các bài toán phức tạp hơn bao gồm cả việc định tuyến SFC và đặt VNF cũng như là các bài toán online khi các yêu cầu không đến một cách đồng thời.

Công bố

N. T. Tam, D. D. Anh, et al. "Genetic programming for resource allocation in network function virtualization." 2023 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2023.

TÀI LIỆU THAM KHẢO

[1] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[2] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm," *TIK-report*, vol. 103, 2001.

[3] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.

[4] F. Zou, L. Wang, X. Hei, D. Chen, and B. Wang, "Multi-objective optimization using teaching-learning-based optimization algorithm," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, pp. 1291–1300, 2013.

[5] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications surveys & tutorials*, vol. 18, no. 1, pp. 236–262, 2015.

[6] B. Yi, X. Wang, K. Li, M. Huang, *et al.*, "A comprehensive survey of network function virtualization," *Computer Networks*, vol. 133, pp. 212–262, 2018.

[7] S. Yang, F. Li, S. Trajanovski, R. Yahyapour, and X. Fu, "Recent advances of resource allocation in network function virtualization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 2, pp. 295–314, 2020.

[8] W. Ma, O. Sandoval, J. Beltran, D. Pan, and N. Pissinou, "Traffic aware placement of interdependent nfv middleboxes," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, IEEE, 2017, pp. 1–9.

[9] M. Hamann and M. Fischer, "Path-based optimization of nfv-resource allocation in sdn networks," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, IEEE, 2019, pp. 1–6.

[10] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Transactions on communications*, vol. 64, no. 9, pp. 3746–3758, 2016.

[11] S. Agarwal, F. Malandrino, C.-F. Chiasserini, and S. De, "Joint vnf placement and cpu allocation in 5g," in *IEEE INFOCOM 2018-IEEE conference on computer communications*, IEEE, 2018, pp. 1943–1951.

[12] R. Yu, G. Xue, and X. Zhang, "Qos-aware and reliable traffic steering for service function chaining in mobile networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2522–2531, 2017.

[13] B. Spinnewyn, P. H. Isolani, C. Donato, J. F. Botero, and S. Latré, "Coordinated service composition and embedding of 5g location-constrained network functions," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1488–1502, 2018.

[14] I. Jang, D. Suh, S. Pack, and G. Dán, "Joint optimization of service function placement and flow distribution for service function chaining," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2532–2541, 2017.

[15] H. Feng, J. Llorca, A. M. Tulino, D. Raz, and A. F. Molisch, "Approximation algorithms for the nfv service distribution problem," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, IEEE, 2017, pp. 1–9.

[16] K. Miettinen, *Nonlinear multiobjective optimization*. Springer Science & Business Media, 1999, vol. 12.

[17] S Brisset and F Gillon, "Approaches for multi-objective optimization in the ecodesign of electric systems," *Eco-Friendly Innovation in Electricity Transmission and Distribution Networks*, pp. 83–97, 2015.

[18] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[19] A. E. Eiben and J. E. Smith, *Introduction to evolutionary computing*. Springer, 2015.

[20] K. Deb, "Nonlinear goal programming using multi-objective genetic algorithms," *Journal of the Operational Research Society*, vol. 52, no. 3, pp. 291–302, 2001.

[21] Y. Haimes, "On a bicriterion formulation of the problems of integrated system identification and system optimization," *IEEE transactions on systems, man, and cybernetics*, no. 3, pp. 296–297, 1971.

[22] T. Wang, H. Xu, and F. Liu, "Multi-resource load balancing for virtual network functions," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2017, pp. 1322–1332.

[23] G. Sallam, G. R. Gupta, B. Li, and B. Ji, "Shortest path and maximum flow problems under service function chaining constraints," in *IEEE INFO-*

COM 2018-IEEE Conference on Computer Communications, IEEE, 2018, pp. 2132–2140.

[24] Z. Xu, W. Liang, M. Huang, M. Jia, S. Guo, and A. Galis, "Approximation and online algorithms for nfv-enabled multicasting in sdns," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2017, pp. 625–634.

[25] A. Dwaraki and T. Wolf, "Adaptive service-chain routing for virtual network functions in software-defined networks," in *Proceedings of the 2016 workshop on Hot topics in Middleboxes and Network Function Virtualization*, 2016, pp. 32–37.

[26] J. Pei, P. Hong, K. Xue, and D. Li, "Efficiently embedding service function chains with dynamic virtual network function placement in geo-distributed cloud system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 10, pp. 2179–2192, 2018.

[27] K. Xie, X. Zhou, T. Semong, and S. He, "Multi-source multicast routing with qos constraints in network function virtualization," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, IEEE, 2019, pp. 1–6.

[28] E. Zitzler and L. Thiele, "An evolutionary algorithm for multiobjective optimization: The strength pareto approach," *TIK report*, vol. 43, 1998.

[29] N. T. Tam, D. D. Anh, T. H. Hung, P. V. Hanh, H. T. T. Binh, and L. T. Vinh, "Genetic programming for resource allocation in network function virtualization," in *2023 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2023, pp. 01–07.

[30] D. Lei, M. Li, and L. Wang, "A two-phase meta-heuristic for multiobjective flexible job shop scheduling problem with total energy consumption threshold," *IEEE transactions on cybernetics*, vol. 49, no. 3, pp. 1097–1109, 2018.

[31] N. T. Tam, T. H. Hung, H. T. T. Binh, *et al.*, "A decomposition-based multiobjective optimization approach for balancing the energy consumption of wireless sensor networks," *Applied Soft Computing*, vol. 107, p. 107 365, 2021.

[32] M. Laszczyk and P. B. Myszkowski, "Survey of quality measures for multi-objective optimization: Construction of complementary set of multi-objective quality measures," *Swarm and Evolutionary Computation*, vol. 48, pp. 109–133, 2019.

[33] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[34] H. Ishibuchi, Y. Nojima, and T. Doi, “Comparison between single-objective and multi-objective genetic algorithms: Performance comparison and performance measures,” in *2006 IEEE International Conference on Evolutionary Computation*, IEEE, 2006, pp. 1143–1150.