Giải thuật di truyền giải bài toán tối ưu đặt và định tuyến trong mạng ảo hóa

Lê Đăng Nguyên[1], Trần Bá Tuấn[2], Nguyễn Thị Tâm[2], Lê Trọng Vĩnh[2]
[1] Trường Đại học Hải Phòng, Hải Phòng, Việt Nam
[2] Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Hà Nội, Hà Nội, Việt Nam
Tác giả liên hệ: Lê Trọng Vĩnh, vinhlt@gmail.com
Ngày nhận bài: 19/01/2024, ngày sửa chữa: 06/06/2024, ngày duyệt đăng: 24/06/2024
Định danh DOI: 10.32913/mic-ict-research-vn.v2024.n2.1254

Tóm tắt: Ngày nay, hạ tầng mạng Internet không ngừng được cải thiện để đáp ứng nhu cầu sử dụng ngày càng cao của người dùng. Việc lắp đặt các thiết bị phần cứng trong trung tâm máy chủ của mạng gặp phải nhiều hạn chế như: khó thay thế thiết bị; chi phí đầu tư, vận hành lớn,... Công nghệ ảo hóa các chức năng mạng (*Network Function Virtualization - NFV*) ra đời đã khắc phục các nhược điểm nêu trên. Trong công nghệ này, các phần cứng chuyên dụng sẽ được triển khai thay thế bằng các phần mềm chạy trong môi trường ảo hoá. Phân bổ tài nguyên là một vấn đề quan trọng trong NFV. Việc phân bổ tài nguyên hợp lý có thể giúp tăng chất lượng dịch vụ và giảm chi phí triển khai. Tối ưu việc phân bổ tài nguyên có thể được thực hiện thông qua việc đặt các chức năng mạng ảo (VNF) trên các máy chủ và tìm đường định tuyến hợp lý cho các dịch vụ mạng. Bài báo này nghiên cứu bài toán đặt các chức năng mạng ảo và định tuyến cho các chuỗi dịch vụ để đáp ứng hai mục tiêu: i) giảm chi phí triển khai; ii) giảm độ trễ của các chuỗi dịch vụ. Tối ưu đa mục tiêu được chuyển về tối ưu đơn mục tiêu (SO-VPTR) bằng cách tiếp cận tổng có trọng số. Nghiên cứu đề xuất mô hình quy hoạch tuyến tính nguyên hỗn hợp (MILP) để giải tối ưu hoá đơn mục tiêu này. Mô hình MILP có thể đưa ra lời giải chính xác cho các bộ dữ liệu có kích thước nhỏ. Để tìm lời giải cho các bộ dữ liệu có kích thước lớn hơn, nghiên cứu đề xuất thuật toán di truyền để giải bài toán. Kết quả thực nghiệm trên các bộ dữ liệu khác nhau đã chứng minh hiệu quả của thuật toán đề xuất.

Từ khóa: *ảo hóa chức năng mạng, thuật toán di truyền, phân bổ tài nguyên.*

Title: **Genetic Algorithm for Optimizing the Virtual Network Functions Placement and Traffic Routing Problem.**

Abstract: Recently, the Internet infrastructure is constantly being improved to meet the increasing needs of users. Installing hardware devices in the network server center faces many limitations such as: difficulty in replacing devices; large investment and operating costs,... Network Function Virtualization (NFV) technology was invented to overcome the disadvantages. Specialized hardware will be replaced by software running in visualized environments in this technology. Resource allocation is an important problem in NFV. Reasonable resource allocation can increase service quality and reduce network deployment costs. Optimizing resource allocation can be accomplished through placing virtual network functions (VNFs) on servers and finding appropriate routing for network services. This research explores the network resource allocation to achieve two objectives: i) reduce deployment costs; ii) reduce the delay of the service chain in the network. We transfer the multi-objective problem to single-objective problem (SO-VPTR) by the weighted-sum approach. This research proposes a mixed-integer linear programming (MILP) model to solve the single-objective optimization problem. The MILP model can provide accurate solutions for small datasets. In order to find solutions for larger sized datasets, the research proposes a genetic algorithm to solve the problem. The experimental results on different datasets demonstrated the effectiveness of the proposed algorithm.

Keywords: *network function virtualization, genetic algorithm, resource allocation.*

I. GIỚI THIỆU

Ảo hóa các chức năng mạng (*Network Function Virtualization - NFV*) là một khái niệm trong lĩnh vực mạng máy tính, nhằm tối ưu hóa, cải tiến và đơn giản kiến trúc mạng. NFV dựa trên các kĩ thuật ảo hoá để cho phép các chức năng mạng ảo có thể được triển khai động trên hạ tầng của NFV mà không cần lắp đặt các thiết bị phần cứng mới. Mạng ảo hoá về cơ bản dựa trên các công nghệ ảo hoá đã có như các công nghệ đã phát triển cho điện toán đám

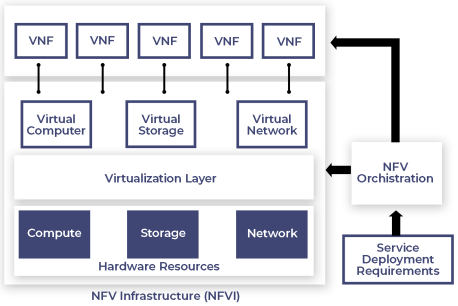
mây, bao gồm ba thành phần chính: các chức năng mạng ảo (*Virtual Network Function - VNF*), hạ tầng mạng ảo hoá (*Network Function Virtualization Infrastructure - NFVI*) và hệ thống quản trị ảo hoá chức năng mạng (*Network Function Virtualization Management - NFVM*). Hình 1 trình bày tổng quan hệ thống mạng ảo hoá.

Trong NFV, các dịch vụ được yêu cầu được thực hiện bởi một chuỗi VNF có thể được chạy trên các máy chủ bằng cách tận dụng công nghệ ảo hóa. Các VNF này được thực hiện với thứ tự xác định trước và được gọi là chuỗi chức năng dịch vụ (*Service Function Chaining - SFC*). Hay nói một cách khác, một SFC tương ứng với một chuỗi các VNF, thông qua chuỗi này, một luồng lưu lượng dữ liệu có thể đi qua từ nguồn đến đích của nó. Hình 2 minh hoạ một chuỗi dịch vụ đi qua 3 VNF là: tường lửa (*Firewall - FW*), hệ thống phát hiện xâm nhập (*Intrusion Detection System - IDS*) và cân bằng tải (*Load Balancer - LB*). Hiện nay, nhiều cấu hình mạng ảo hóa có thể cùng tồn tại trên một cơ sở hạ tầng vật lý. Các SFC dựa trên phần mềm có thể được khởi tạo và cập nhật dễ dàng bằng cách đặt trực tiếp trên máy chủ thay vì sử dụng cấu trúc vật lý một cách thủ công nếu muốn nâng cấp các phần cứng. Như vậy, có thể thấy rằng NFV phân bổ tài nguyên mạng một cách linh động hơn.

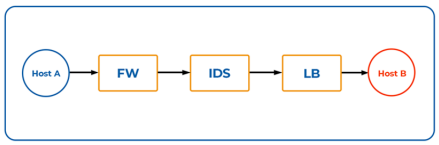
Từ những ưu điểm mà NFV mang lại, một câu hỏi được đặt ra là làm như thế nào để phân bổ nguồn tài nguyên cho các SFC được yêu cầu? Điều này phụ thuộc vào việc giải quyết hai bài toán: định tuyến cho các chuỗi dịch vụ (*Traffic Routing Problem - TRR*), bài toán đặt VNF và định tuyến (*VNF Placement and Traffic Routing - VPTR*) và các biến thể của hai bài toán này. Mục tiêu của các bài toán là đặt các VNF trên mạng và định tuyến giữa các VNF sao cho các ràng buộc về khả năng của mỗi nút và băng thông trên các liên kết được thỏa mãn.

Do các đặc tính sẵn có và các nguyên tắc thiết kế của NFV, bài toán VPTR cũng có một số khác biệt so với các bài toán đã có: bài toán định tuyến và đặt dịch vụ trong nghiên cứu [1], bài toán định tuyến và đặt các máy ảo trong nghiên cứu [2, 3]. Ví dụ, khi các VNF yêu cầu được đặt trên mạng, tuyến đường đi qua từng VNF sẽ theo một thứ tự xác định trước. Ngoài ra, các tham số như chất lượng dịch vụ (*Quality of Service - QoS*) cũng được xem xét trong bài toán VPTR. Ví dụ, nếu chúng ta coi độ trễ đầu cuối của các gói đi qua mỗi VNF trong một SFC là một thước đo đánh giá chất lượng dịch vụ của NFV, để đảm bảo một mạng đáp ứng dịch vụ, các nhà cung cấp dịch vụ cần phải thiết kế mạng sao cho độ trễ nằm trong một ngưỡng cho phép. Bên cạnh độ trễ, khả năng phục hồi cũng là một tham số quan trọng khác để đo chất lượng dịch vụ. Khả năng phục hồi xác định mức độ mà dịch vụ được cung cấp có thể tồn tại khi gặp sự cố. Nếu một nút hoặc một liên kết trong SFC bị lỗi, toàn bộ SFC không thể hoạt động và do đó, dịch vụ được cung cấp sẽ bị tạm ngừng. Có thể thấy rằng, bài toán VPTR và các biến thể của bài toán này sẽ trở nên phức tạp hơn khi tham số chất lượng dịch vụ được xem xét. Nghiên cứu này sẽ tập trung vào giải quyết các bài toán tối ưu hóa việc phân bổ tài nguyên trong NFV có xem xét đến chất lượng dịch vụ của mạng. Nghiên cứu cũng đề xuất các giải thuật gần đúng (xấp xỉ, heursitic và metaheuristic) để giải quyết các bài toán được đưa ra.

Phần còn lại của bài báo được tổ chức như sau: Phần II trình bày một số nghiên cứu liên quan đến bài toán đặt và định tuyến trong mạng ảo hóa. Phần III trình bày mô hình hoá bài toán và đưa ra mô hình quy hoạch nguyên hỗn hợp cho bài toán. Thuật toán đề xuất và thực nghiệm được trình bày lần lượt trong phần IV và Phần V. Cuối cùng, phần VI sẽ tổng kết lại một số kết quả đạt được của bài báo cũng như trình bày một số hướng nghiên cứu trong tương lai.



Hình 1. Hệ thống mạng ảo hoá.

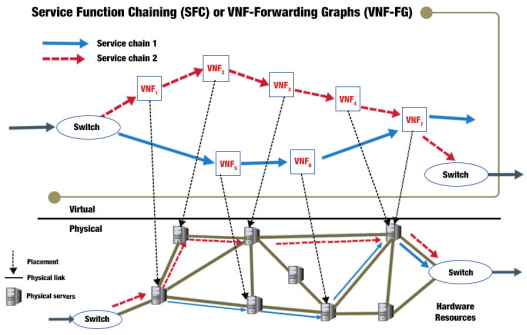


Hình 2. Minh họa một chuỗi chức năng dịch vụ với FW, IDS và LB.

II. CÁC NGHIÊN CỨU LIÊN QUAN

Bài toán phân bổ tài nguyên trong mạng NFV đã và đang nhận được sự quan tâm của các nhà nghiên cứu trong và ngoài nước trong thời gian gần đây. Hầu hết các nghiên cứu đối với bài toán này đều quan tâm đến các tham số chất lượng dịch vụ khác nhau như chi phí, độ trễ, tính sẵn có, năng lượng,... Trong phần này, nghiên cứu sẽ khảo sát dựa trên các bài toán tổng quát đã được trình bày ở phần trên: bài toán VPTR, bài toán đặt VNF (*VNF Placement - VNFP*) như hình 3 và bài toán TRR.

Ma và các cộng sự trong nghiên cứu [4] nghiên cứu bài toán VPTR với mục tiêu cực tiểu hóa giá trị tải lớn nhất của



Hình 3. Minh họa bài toán đặt và định tuyến chuỗi chức năng dịch vụ.

các liên kết. Các tác giả chứng minh bài toán là NP-hard ngay cả trong trường hợp VNF không có thứ tự bằng cách quy dẫn về bài toán tìm chu trình Hamilton. Các tác giả cũng đề xuất thuật toán LFLG (*Least First Greatest Last*) dựa trên thuật toán định tuyến min-max để giải bài toán này. Trong trường hợp VNF không có thứ tự, Cohen [5] đề xuất một thuật toán xấp xỉ sử dụng kĩ thuật làm tròn (*Rouding Technique*) để giải quyết bài toán VPTR với mục tiêu cực tiểu hóa tổng chi phí của hệ thống. Chi phí của hệ thống ở đây được tính bằng chi phí cài đặt chức năng f trên nút n và tổng khoảng cách giữa các máy khách và các nút nhận dịch vụ. Ghaznavi [6] trình bày cách phân bố các VNF trên nhiều nút để giảm thiểu tỉ lệ tải của các liên kết (hoặc tối đa thông lượng). Sau đó, các tác giả trình bày phương pháp tìm kiếm địa phương sử dụng tham số điều chỉnh để cân bằng giữa độ chính xác và thời gian để giải quyết bài toán VPTR. Spinney và cộng sự [7] xem xét bài toán VPTR với ràng buộc thứ tự bộ phận. Họ mô hình hóa bài toán về mô hình quy hoạch tuyến tính nguyên (*Integer Linear Programming - ILP*) dựa trên cây VNF được tăng cường với các SFC đáp ứng về độ ưu tiên. Sau đó, họ đề xuất thuật toán heuristic để giải quyết bài toán. Murray và cộng sự trong nghiên cứu [8] đã trình bày bài toán đặt và định tuyến VNF dựa trên thuật toán sinh cột (*Column Generation*) để giải bài toán với việc xem xét các ràng buộc mức độ dịch vụ (*Service Level Agreement - SLA*). Ngoài ra, Golkarifard và cộng sự [9] đã đề cập tới bài toán VNF động với đặt, phân bổ tài nguyên và định tuyến qua việc xây dựng mô hình quy hoạch phi tuyến nguyên và đề xuất một thuật toán heuristic gần tối ưu - MaxSR.

Kuo [10] quan sát và đưa ra nhận xét rằng: việc sử dụng nhiều máy ảo hơn có thể dẫn đến một tuyến đường dài hơn và do đó, có thể tốn băng thông liên kết hơn. Feng [11] đã đề xuất một thuật toán xấp xỉ $O(\epsilon)$ trong thời gian $O(1/\epsilon)$ để tìm giải pháp tối thiểu cho bài toán VPTR với một tập các yêu cầu cho trước. Li và các cộng sự [12] mô hình bài toán đặt VNF sang bài toán cặp ghép trong đồ thị và đề xuất một thuật toán heuristic cân bằng tải để giải quyết bài toán này. Ma [4] đề xuất một thuật toán chính xác trong thời gian đa thức để giải bài toán VNFP với trường hợp các VNF không có thứ tự. Các tác giả cũng chứng minh bài toán VNFP có thứ tự toàn bộ và bộ phận là bài toán NP-hard. Pham [13] đề xuất thuật toán xấp xỉ Markov dựa trên mẫu để tối thiểu hóa chi phí cho bài toán VNFP.

Wang [14] đề xuất thuật toán ADMM (*Alternating Direction Method of Multipliers*) để giải bài toán cân bằng tải TRR cho nhiều yêu cầu. Ràng buộc SFC yêu cầu đường đi từ nguồn đến đích phải đi qua mỗi VNF theo một thứ tự nào đó. Sallam [15] đưa ra một thuật toán thời gian đa thức để giải quyết bài toán đường đi ngắn nhất có ràng buộc SFC thông qua đồ thị luồng. Ràng buộc SFC trong bài toán luồng cực đại yêu cầu mỗi luồng đi qua một tập hợp các chức năng mạng theo một thứ tự xác định trước khi đến đích. Sau đó, họ đề xuất phương pháp quy hoạch nguyên để giải quyết bài toán đặt các VNF sao cho giá trị của luồng cực đại với ràng buộc SFC bằng giá trị của luồng cực đại ban đầu không có ràng buộc của SFC. Đối với bài toán định tuyến đa điểm (*Multicast Traffic Routing Problem - MTRR*) trong NFV, có một nút nguồn và nhiều nút đích, vấn đề là tìm một cây xuất phát từ một nguồn đến nhiều điểm đích sao cho đi qua các SFC yêu cầu. Xu và các cộng sự [16] đề xuất một thuật toán gần đúng để giải quyết bài toán MTRR bằng cách đưa bài toán này về bài toán cây Steiner trong một đồ thị vô hướng.

Qu và các cộng sự [17] xem xét đến độ trễ khi truyền và xử lý các VNF trong bài toán VNFP, đồng thời mô hình bài toán này về mô hình quy hoạch tuyến tính nguyên hỗn hợp (*Mixed Integer Linear Programming - MILP*). Tuy nhiên, họ giả định rằng các liên kết ảo giữa hai nút vật lý có thể xử lý nhiều nhất một luồng dữ liệu tại một thời điểm. Zhang và cộng sự [18] đưa ra một thuật toán dựa trên ADMM để giải quyết bài toán VPTR với độ trễ. Tuy nhiên, họ không xem xét đến độ trễ của các nút. Li và các cộng sự [19] giải quyết bài toán VPTR có xem xét đến độ trễ.

Nghiên cứu [20] và [21] đề xuất một thuật toán xấp xỉ dựa trên đồ thị phân lớp để giải bài toán TRR có xem xét đến độ trễ. Ý tưởng chính của cách tiếp cận này là xây dựng một đồ thị mới bằng cách tạo ra $|F|$ bản sao từ đồ thị ban đầu, $|F|$ là số lượng các VNF được yêu cầu. Các liên kết giữa các lớp được tạo ra để kết nối cùng một nút trong các lớp khác nhau và các nút xen kẽ được tạo ra để đại diện cho các vị trí có thể lưu trữ mỗi VNF trong mỗi lớp. Xie [22] trình bày một thuật toán heuristic dựa trên cây bao trùm tối thiểu (*Minimum Spanning Tree*) để tìm một liên kết chung mà các SFC đã triển khai có thể được chia sẻ bởi nhiều yêu cầu.

Chen [23] đề xuất một thuật toán heuristic dựa trên chuỗi Markov ẩn để đặt VNF để cực tiểu hóa cả chi phí và độ trễ

cho một tập các luồng cho trước. Trong [24], bài toán phân bổ tài nguyên CPU và VNFP được mô hình hóa như một mô hình tối ưu không lồi, bài toán ban đầu sẽ được chia thành hai bài toán con. Các tác giả đề xuất một thuật toán heuristic để giải bài toán VNFP có xem xét đến độ trễ và đề xuất thuật toán trong thời gian đa thức để giải bài toán phân bổ tài nguyên CPU bằng cách dẫn về điều kiện KKT (Karush-Kuhn-Tucker).

Khi ràng buộc về độ trễ SFC được đưa thêm vào, chúng ta có thể thấy rằng bài toán VPTR và các biến thể của các bài toán này sẽ khó hơn những bài toán được đưa ra ban đầu trong phần III. Hơn thế nữa, một số nghiên cứu thường áp dụng lý thuyết về hàng đợi để giải quyết bài toán VPTR bằng cách kết hợp cả độ trễ của hàng đợi và độ trễ khi xử lý. Tuy nhiên, lý thuyết về hàng đợi thường giả định rằng tỉ lệ luồng dữ liệu đến tuân theo phân phối Poisson và thời gian xử lý ở mỗi nút tuân theo phân phối mũ. Điều này không phải lúc nào cũng đúng trong thực tế (ví dụ trong trường hợp mà luồng dữ liệu đến một cách dồn dập).

Hạn chế của các nghiên cứu liên quan:

- Một số nghiên cứu đề xuất các thuật toán để định tuyến tối ưu cho một dịch vụ mạng. Tuy nhiên, trong thực tế, mạng cần xử lý nhiều dịch vụ một cách đồng thời.
- Một số nghiên cứu giải quyết bài toán định tuyến cho nhiều dịch vụ mạng. Tuy nhiên, việc chọn thứ tự định tuyến cho các dịch vụ sẽ ảnh hưởng đến kết quả của bài toán.
- Một số nghiên cứu chỉ xem xét bài toán tối ưu đơn mục tiêu hoặc đảm bảo cân bằng tải hoặc tối đa số lượng dịch vụ mạng được đáp ứng.

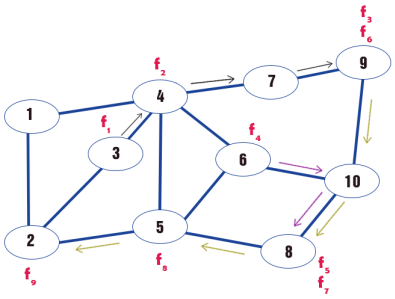
Đóng góp chính của nghiên cứu này như sau:

- Đề xuất mô hình cho bài toán đặt và định tuyến cho các dịch vụ mạng thỏa mãn hai mục tiêu: i) tối thiểu chi phí triển khai mạng; ii) tối thiểu độ trễ của các chuỗi dịch vụ trong mạng.
- Đưa bài toán tối ưu hoá đa mục tiêu về bài toán tối ưu đơn mục tiêu ((bài toán SO-VPTR)) sử dụng phương pháp tổng có trọng số.
- Đề xuất mô hình quy hoạch tuyến tính nguyên hỗn hợp cho bài toán SO-VPTR.
- Đề xuất thuật toán di truyền để giải bài toán SO-VPTR.
- Tiến hành thực nghiệm thuật toán đề xuất trên các bộ dữ liệu khác nhau để chứng minh tính hiệu quả của thuật toán.

III. MÔ HÌNH HÓA BÀI TOÁN

1. Đầu vào

Mạng vật lý mô hình hóa như một đồ thị $G = (P, E)$, trong đó P là tập các máy chủ vật lý trong mạng, với mỗi



Hình 4. Ví dụ một bài toán đặt và định tuyến chuỗi chức năng dịch vụ.

Bảng I
ĐƯỜNG ĐỊNH TUYẾN CỦA CÁC CHUỖI DỊCH VỤ.

SFC	Đường dẫn ảo	Đường dẫn vật lý
1	$f_1 - f_2 - f_3$	$v_3 - v_4 - v_7 - v_9$
2	$f_4 - f_5$	$v_3 - v_6 - v_{10} - v_8$
3	$f_6 - f_7 - f_8 - f_9$	$v_9 - v_{10} - v_8 - v_5 - v_2$

máy chủ vật lý $p \in P$ được đặc trưng bởi các thành phần c_p^P, r_p^P, h_p^P, d_p^P là tài nguyên CPU, RAM, DISK và thời gian trễ ảo hóa; E là tập các liên kết vật lý. Mỗi liên kết vật lý được đặc trưng bởi b_{pq}^E, d_{pq}^E là tài nguyên băng thông và thời gian trễ khi lan truyền trên nó. Với mỗi VNF $v \in V$ được đặt trên các máy chủ vật lý (P) bao gồm các thành phần c_v^V, r_v^V, h_v^V là tài nguyên sử dụng CPU, RAM, DISK.

Mỗi chuỗi chức năng dịch vụ $s \in S$ gồm tập các VNF V_s và tập các liên kết ảo L. Mỗi liên kết ảo $(v, u) \in L$ bao gồm tài nguyên sử dụng băng thông b_{vu}^L và thời gian trễ khi lan truyền d_{vu}^L.

2. Đầu ra

Sắp xếp 1 tập V các VNF vào 1 tập máy chủ vật lý (PM) P và điều hướng chúng theo các các chuỗi SFC, đồng thời thỏa mãn các ràng buộc của bài toán.

Các biến quyết định bao gồm:

$$x(v,p) = \begin{cases} 1, & \text{nếu } v \text{ được đặt vào } p \\ 0, & \text{ngược lại} \end{cases}$$

$$y((v,u),(p,q)) = \begin{cases} 1, & \text{nếu } (u,v) \in L \text{ được đặt trên } (p,q) \\ 0, & \text{ngược lại} \end{cases}$$

3. Các mục tiêu

Mục tiêu đầu tiên xét đến là **tối thiểu hóa chi phí tổng thể** (chi phí phát sinh trong hệ thống cho tất cả các chuỗi dịch vụ) được mô tả dưới đây

$$CO(r,t) = CO(r) + CO(t) \quad (1)$$

trong đó, $CO(r)$ là chi phí tài nguyên (CPU, RAM, DISK) và $CO(t)$ là chi phí giao tiếp (băng thông).
Coi các máy chủ vật lý (PM) có chi phí không đồng nhất, việc đặt VNF tại mỗi máy chủ vật lý phát sinh một chi phí tính toán khác nhau. Chi phí cho các tài nguyên với mỗi chuỗi dịch vụ $s \in S$ được tính là chi phí cho CPU, RAM, DISK của các máy chủ vật lý mà các VNF của SFC đặt trên đó.

$$CO(r) = \sum_{p \in P} \sum_{v \in V} (co_c^p c_v^V + co_r^p r_v^V + co_h^p h_v^V).x(v,p) \quad (2)$$

Chi phí giao tiếp được tính bởi tổng chi phí băng thông sử dụng của liên kết ảo (VL) khi (v,u) đặt trên liên kết vật lý (PL) (p,q).

$$CO(t) = \sum_{(p,q) \in E} \sum_{(v,u) \in L} co_b^p b_{vu}^L.[y((v,u),(p,q)) + y((v,u),(q,p))] \quad (3)$$

Kết hợp các phương trình (1 - 3), chi phí tổng thể cho tất cả các chuỗi dịch vụ là:

$$CO(r,t) = \sum_{p \in P} \sum_{v \in V} (co_c^p c_v^V + co_r^p r_v^V + co_h^p h_v^V).x(v,p) + \sum_{(p,q) \in E} \sum_{(v,u) \in L} co_b^p b_{vu}^L.[y((v,u),(p,q)) + y((v,u),(q,p))] \quad (4)$$

Tùy thuộc vào cấu trúc liên kết của hệ thống mà các liên kết vật lý có độ trễ khác nhau khi xử lý chuỗi dịch vụ. Ngoài ra, việc tập hợp các VNF trên một số lượng giới hạn các máy ảo gây tình trạng tắc nghẽn mạng dẫn đến tăng độ trễ của chuỗi dịch vụ. Hơn nữa, dựa trên QoS và thỏa thuận mức độ dịch vụ (*Service Level Agreement - SLA*) dự kiến giữa người dùng và nhà cung cấp thì mỗi chuỗi dịch vụ có thể có thời hạn độ trễ. Do đó, chúng ta sẽ có thêm mục tiêu là **tối thiểu hóa độ trễ** và nó được xác định bởi công thức sau:

$$D(c,p,v,q) = D(c) + D(p) + D(v) + D(q) \quad (5)$$

trong đó, $D(c)$ là độ trễ lan truyền, $D(p)$ là độ trễ xử lý, $D(v)$ thể hiện cho độ trễ ảo hóa và $D(q)$ đại diện cho độ trễ hàng đợi.
Độ trễ lan truyền là độ trễ trên liên kết vật lý và độ trễ trên liên kết ảo. Với mỗi liên kết vật lý, liên kết ảo sẽ có một độ trễ nhất định.

$$D(c) = \sum_{s \in S} \sum_{(p,q) \in E} \sum_{(v,u) \in L_s} (d_{pq}^E + d_{vu}^L).[y((v,u),(p,q)) + y((v,u),(q,p))] \quad (6)$$

Độ trễ xử lý là thời gian chạy VNF trên các nút vật lý, tính bằng thương số giữa tổng số tài nguyên yêu cầu chia cho tổng số tài nguyên tại máy chủ vật lý đó (coi độ trễ xử lý chỉ ảnh hưởng bởi tài nguyên CPU).

$$D(p) = \sum_{s \in S} \sum_{p \in P} \sum_{v \in V_s} \frac{c_v^V}{c_p^P}.x(v,p) \quad (7)$$

Khi thực hiện một chức năng ảo trên một máy chủ vật lý sẽ có một khoảng thời gian trễ ảo hóa nhất định

$$D(v) = \sum_{s \in S} \sum_{p \in P} \sum_{v \in V_s} d_p^P.x(v,p) \quad (8)$$

Độ trễ hàng đợi của chuỗi dịch vụ phụ thuộc vào việc sử dụng liên kết và có thể được tính bằng thương số của tổng yêu cầu bằng thông và giới hạn bằng thông của tất cả các máy chủ vật lý đang đặt các VNF.

$$D(q) = \sum_{s \in S} \sum_{(p,q) \in E} \sum_{(v,u) \in L_s} \frac{b_{vu}^L}{b_{pq}^E}.[y((v,u),(p,q)) + y((v,u),(q,p))] \quad (9)$$

Chi phí triển khai chuỗi dịch vụ và độ trễ đều là những mục tiêu quan trọng trong bài toán đặt và định tuyến chuỗi dịch vụ. Tuy nhiên, các mục tiêu đó mâu thuẫn nhau. Ví dụ, việc giảm thiểu số lượng các máy chủ vật lý đang hoạt động làm tăng lưu lượng mạng gây tắc nghẽn mạng, do đó có thể làm tăng độ trễ.

Vì vậy, việc kết hợp tối ưu chi phí và độ trễ bằng cách sử dụng trọng số $\alpha \in [0,1]$ được biểu diễn công thức dưới đây:

$$min \quad \alpha.CO(r,t) + (1-\alpha).D(c,p,v,q) \quad (10)$$

Có thể điều chỉnh α để cân bằng giữa tối ưu chi phí và độ trễ. Nếu coi mục tiêu tối ưu chi phí là quan trọng, đặt α lớn, khi $\alpha = 1$ chỉ tối ưu hóa chi phí và bỏ qua tác động của độ trễ. Mặt khác, α càng nhỏ càng nhấn mạnh việc tối ưu độ trễ, nếu $\alpha = 0$ tối ưu chỉ dựa trên độ trễ.

4. Các ràng buộc bài toán

Bài toán đặt và định tuyến chuỗi chức năng dịch vụ chúng tôi xem xét các ràng buộc dưới đây:

$$\sum_{v \in V} c_v^V x(v,p) \le c_p^P, \forall p \in P \quad (11)$$

$$\sum_{v \in V} r_v^V x(v,p) \le r_p^P, \forall p \in P \quad (12)$$

$$\sum_{v \in V} h_v^V x(v,p) \le h_p^P, \forall p \in P \quad (13)$$

$$\sum_{(v,u) \in L} b_{vu}^L.[y((v,u),(p,q)) + y((v,u),(q,p))] \le b_{pq}^E \quad \forall (p,q) \in E, p \ne q \quad . \quad (14)$$

$$\sum_{p \in P} x(v,p) = 1, \forall v \in V \quad (15)$$

$$\sum_{q \in P, (p,q) \in E} y((v,u),(p,q)) \le 1, \forall p \in P \quad (16)$$

$$\sum_{q \in P, (q,p) \in E} y((v,u),(q,p)) \le 1, \forall p \in P \quad (17)$$

$$x(v,p) \le x(u,p) + \sum_{q\in P,(p,q)\in E} y((v,u),(p,q)), \forall p \in P \tag{18}$$

$$x(u,p) \le x(v,p) + \sum_{q\in P,(q,p)\in E} y((v,u),(q,p)), \forall p \in P \tag{19}$$

$$x(v,p) - x(u,p) = \sum_{q\in P,(p,q)\in E} y((v,u),(p,q)) - \sum_{q\in P,(q,p)\in E} y((v,u),(q,p)), \forall p \in P \tag{20}$$

$$\sum_{(p,q)\in E}\sum_{(v,u)\in V_s}(d^E_{pq}+d^L_{vu}).[y((v,u),(p,q))+y((v,u),(q,p))] + \sum_{s\in S}\sum_{p\in P}\sum_{v\in V_s}\frac{c^V_v}{c^P_p}.x(v,p) + \sum_{s\in S}\sum_{p\in P}\sum_{v\in V_s} d^P_p.x(v,p) + \sum_{s\in S}\sum_{(p,q)\in E}\sum_{(v,u)\in L_s}\frac{b^L_{vu}}{b^E_{pq}}.[y((v,u),(p,q))+y((v,u),(q,p))] \le d^{max}_s, \forall s \in S \tag{21}$$

Trong đó, công thức 11, 12 và 13 là ràng buộc về tài nguyên, tức là CPU, RAM và DISK của các VNF đặt trên các máy chủ vật lý không vượt quá CPU, RAM, DISK của các máy chủ vật lý. Ràng buộc băng thông qua mỗi liên kết vật lý không được vượt quá khả năng của liên kết vật lý được trình bày ở công thức 14. Công thức 15 là ràng buộc mỗi VNF được đặt duy nhất vào một máy chủ vật lý (PM). Ràng buộc về luồng được mô tả từ công thức 16 đến 20, mỗi SFC được phục vụ thông qua một đường dẫn nối từ nguồn tới đích theo đúng thứ tự yêu cầu. Ràng buộc cuối cùng được trình bày công thức 21 là về độ trễ, tức là mỗi SFC không được vượt quá thời gian trễ cho phép.

Các ký hiệu sử dụng trong mô hình bài toán được mô tả trong Bảng *II*.

IV. THUẬT TOÁN DI TRUYỀN

Thuật toán di truyền được xây dựng dựa trên quy luật tiến hoá sinh học của một quần thể sống. Các cá thể trải qua một quá trình phát triển và sinh sản để tạo ra các cá thể ở thế hệ tiếp theo.

Trong quá trình sinh trưởng và phát triển, những cá thể xấu tức là những cá thể không thích nghi được với môi trường sẽ bị đào thải. Ngược lại, những cá thể tốt sẽ được giữ lại (đây là quá trình chọn lọc) và được lai ghép để tạo ra những cá thể ở thế hệ mới cho thế hệ sau. Những cá thể mới được sinh ra sẽ mang những tính trạng, đặc điểm của cá thể cha - mẹ (hiện tượng di truyền).

Bảng II
CÁC KÝ HIỆU ĐƯỢC SỬ DỤNG.

Ký hiệu	Mô tả
P	Tập các máy chủ vật lý PM
E	Tập các liên kết vật lý PL
V	Tập các VNF
S	Tập các SFC
V_s	Tập các VNF của $s \in S$
L_s	Tập các liên kết ảo của $s \in S$
L	Tập các liên kết ảo, $L = \cup_{s\in S} L_s$
c^P_p	Tài nguyên CPU của máy chủ vật lý ($p \in P$)
r^P_p	Tài nguyên RAM của máy chủ vật lý ($p \in P$)
h^P_p	Tài nguyên DISK của máy chủ vật lý ($p \in P$)
d^P_p	Thời gian trễ ảo hóa của máy chủ vật lý ($p \in P$)
b^E_{pq}	Tài nguyên băng thông của liên kết vật lý ($(p,q) \in E$)
d^E_{pq}	Thời gian trễ khi lan truyền trên liên kết vật lý ($(p,q) \in E$)
c^V_v	Tài nguyên sử dụng CPU của VNF ($v \in V$)
r^V_v	Tài nguyên sử dụng RAM của VNF ($v \in V$)
h^V_v	Tài nguyên sử dụng DISK của VNF ($v \in V$)
b^L_{vu}	Tài nguyên sử dụng băng thông của liên kết ảo ($(v,u) \in L$)
d^L_{vu}	Thời gian trễ khi lan truyền trên liên kết ảo ($(v,u) \in L$)
co^p_c	Chi phí sử dụng 1 CPU
co^p_r	Chi phí sử dụng 1 GB RAM
co^p_b	Chi phí sử dụng băng thông cho mỗi Kbps
$CO(r)$	Tổng chi phí tài nguyên
$CO(t)$	Tổng chi phí giao tiếp
d^{max}_s	Thời gian trễ tối đa của chuỗi SFC
$D(c)$	Độ trễ lan truyền
$D(p)$	Độ trễ xử lý
$D(v)$	Độ trễ ảo hóa
$D(q)$	Độ trễ hàng đợi

Các cá thể mới này được tạo trong quá trình lai ghép có thể xảy ra hiện tượng đột biến và tạo ra những gen (cá thể) khác với cha mẹ. Lai ghép và đột biến là hai hiện tượng có vai trò như nhau trong quá trình tiến hoá mặc dù phép đột biến xảy ra với xác suất thấp hơn so với lai ghép. Chi tiết các bước thực hiện của thuật toán di truyền được trình bày trong thuật toán 1.

1. Mã hóa cá thể và khởi tạo quần thể

Mỗi cá thể Ind trong quần thể được biểu diễn bởi một dãy các số nguyên có độ dài $size = \sum_{s\in S} V_s$.

$$Ind = \{ind_1, ind_2, ..., ind_{size}\}$$

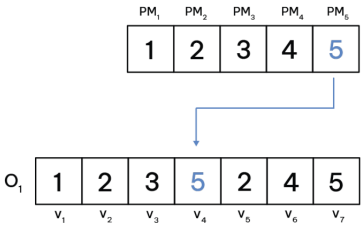
trong đó, ind_i là một giá trị nguyên tương ứng với máy chủ đặt VNF, $ind_i \in \{1, 2, ..., N\}$. Hình 5 minh hoạ cách biểu diễn cá thể cho một mạng gồm 5 nút máy chủ, 7 loại VNF $\{v_1, v_2, ..., v_7\}$ và 2 SFC. Trong đó, SFC thứ nhất yêu cầu phải đi qua tập các VNF là $\{v_1, v_2, v_3, v_4\}$, SFC thứ hai yêu cầu đi qua tập các VNF là $\{v_5, v_6, v_7\}$.

Mỗi cá thể trong quần thể được khởi tạo ngẫu nhiên. Sau quá trình khởi tạo, đường đi của SFC $s \in S$ được xác định là đường đi ngắn nhất thỏa mãn các ràng buộc từ điểm nguồn đến điểm đích.

Thuật toán 1: Thuật toán di truyền (Genetic Algorithm)

Input: Tập hợp chuỗi dịch vụ S, tập hợp máy chủ vật lý P, tập hợp liên kết vật lý E.
Output: Các VNF được đặt trên các máy chủ vật lý
Đường định tuyến cho các chuỗi dịch vụ
Kích thước quần thể $pSize$, số thế hệ $generation$, xác suất đột biến r_m, xác suất lai ghép r_c

$t \leftarrow 0$;
Khởi tạo quần thể $Pop_t = \{pop_1, pop_2, ..., pop_{pSize}\}$;
while $t \leq generation$ **do**
    Tính độ thích nghi của các cá thể trong Pop_t;
    $Q_t \leftarrow$ Lai ghép trên Pop /*Xem thuật toán 2*/
    $R_t \leftarrow$ Đột biến trên Q_t /*Xem thuật toán 3 */
    Tạo ra Pop_{t+1} bằng cách chọn lọc $pSize$ cá thể từ $Q_t \cup R_t \cup Pop_t$
end



Hình 5. Biểu diễn cá thể.

2. Hàm đánh giá cá thể

Độ thích nghi là khả năng phù hợp của cá thể (lời giải) đối với môi trường (bài toán). Việc xây dựng hàm đánh giá độ thích nghi là một trong những bước quan trọng trong thuật toán di truyền. Trong nghiên cứu này, hàm đánh giá độ thích nghi của mỗi cá thể chính là hàm mục tiêu và được tính theo công thức 10.

3. Toán tử di truyền

- **Lai ghép (Crossover)**: thực hiện để khám phá không gian lời giải mới bằng cách trao đổi các đoạn trong chuỗi giữa cha, mẹ được chọn. Chúng tôi lựa chọn lai ghép đồng dạng, tức là mỗi đoạn của cá thể con được chọn ngẫu nhiên với xác suất ngang nhau giữa các đoạn tương ứng của cha mẹ. Để tránh lặp vô hạn, chúng ta đặt điều kiện bằng cách sử dụng hằng số (β), xác định số lần tối đa mà thuật toán tạo ra một cá thể thỏa mãn ràng buộc, nếu không, một trong các cặp bố mẹ được chọn thêm vào quần thể mới.
- **Đột biến (Mutation)**: nếu như phép lai ghép tạo ra các cá thể mới có thể kế thừa được những gen tốt của cha mẹ thì phép đột biến sẽ giúp duy trì sự đa dạng của quần thể bằng cách thêm vào các gen mới một cách ngẫu nhiên. Điều này có thể ngăn chặn sự hội tụ sớm và giúp cho quần thể tránh khỏi những lời giải tối ưu địa phương. Phép đột biến thường xảy ra với xác suất nhỏ vì nếu xác suất đột biến lớn thì thuật toán di truyền sẽ trở thành quá trình tìm kiếm ngẫu nhiên. Tương tự với phép lai ghép, cá thể mới sẽ được kiểm tra, nếu không đột biến sẽ được lặp lại β lần.

Thuật toán 2: Lai ghép (Crossover)

for $i = (eliteSize + 1)$ **to** $pSize$ **do**
    Chọn cặp cha mẹ p, q bằng kỹ thuật Tournament
    for $j = 1$ **to** $cSize$ **do**
        $r \leftarrow randomize(0, 1)$
        if $r \leq F(q)/(F(p) + F(q))$ **then**
            $o_{ij} \leftarrow e_i^q$
        else
            $o_{ij} \leftarrow e_i^p$
        end
    end
    if o_i *không khả thi* **then**
        if $repeat \leq \beta$ **then**
            Quay lại dòng 2
        else
            $newPop_i \leftarrow$ cha mẹ có $F(c)$ tốt hơn
        end
    end
end
return $newPop$

4. Chọn lọc

Chọn lọc là quá trình chọn những cá thể từ quần thể hiện tại để tạo ra thế hệ tiếp theo. Theo thuyết tiến hoá của Darwin, những cá thể xấu sẽ bị đào thải, những cá thể tốt sẽ được giữ lại. Như vậy, độ thích nghi của các cá thể trong quần thể sẽ được hoàn thiện hơn sau nhiều thế hệ. Có nhiều phương pháp để chọn lọc cá thể, trong nghiên cứu này, chúng tôi sử dụng phương pháp chọn lọc giao đấu (*Tournament Selection*). Phương pháp chọn lọc này đã khắc phục được vấn đề về quy mô của phương pháp dựa trên giá trị thích nghi và làm cho các cá thể tốt ít bị loại bỏ. Phương pháp này chỉ lấy giá trị tương đối của hàm đánh độ thích nghi.

V. THỰC NGHIỆM

Thuật toán 3: Đột biến (Mutation)

for $i = (eliteSize + 1)$ **to** $pSize$ **do**
  $r \leftarrow randomize(0, 1)$
  $k \leftarrow randomize(1, cSize)$
  if $r > \gamma$ **then**
    $o_{ik} \leftarrow randomize(1, |P|)$
    if o_i *không khả thi* **then**
      if $repeat \leq \beta$ **then**
        Quay lại dòng 5
      end
    else
      $newPop_{ik} \leftarrow v$
    end
  end
end
return $newPop$

1. Tham số thực nghiệm

Chương trình thực nghiệm giải bài toán đặt và định tuyến các chuỗi chức năng dịch vụ tối ưu chi phí và độ trễ sử dụng mô hình quy hoạch tuyến tính nguyên hỗn hợp (MILP) và thuật toán di truyền được triển khai bằng ngôn ngữ lập trình Python, hệ điều hành Window 11 với CPU AMD Ryzen 7 5825U và RAM 16 Gb.

Mô hình quy hoạch tuyến tính nguyên hỗn hợp (MILP) được cài đặt dựa trên thư viện Gurobi Optimization và được đặt giới hạn trong thời gian 1 giờ đồng hồ.

Thuật toán di truyền được cài đặt với các tham số như Bảng III.

Bảng III
THAM SỐ CỦA THUẬT TOÁN DI TRUYỀN

Tên tham số	Mô tả	Giá trị
$generation$	Số lượng thế hệ	50
$pSize$	Kích thước quần thể	200
r_c	Xác suất lai ghép	0.3
r_m	Xác suất đột biến	0.1

2. Dữ liệu thực nghiệm

Dữ liệu được sử dụng trong nghiên cứu này được tạo ra thông qua việc sử dụng SND Lib (Survivable Network Design), một kho lưu trữ chứa các phiên bản thiết kế mạng viễn thông thực tế. Tuy nhiên, cần lưu ý rằng dữ liệu từ SND Lib không bao gồm thông tin về tài nguyên mạng, yêu cầu của chuỗi dịch vụ, và các thông số liên quan khác. Vì vậy, để hoàn thiện và làm phong phú hơn dữ liệu, các thông tin thiếu này được bổ sung từ mạng topo thu thập được từ SND Lib được đề cập trong Bảng *IV*. Quá trình này giúp nghiên cứu xây dựng một bộ dữ liệu đa dạng, kết hợp thông tin chi tiết về cấu trúc mạng từ SND Lib với các thông số quan trọng về tài nguyên mạng và yêu cầu của chuỗi dịch vụ.

Bảng IV
GIÁ TRỊ MỘT SỐ THAM SỐ CHO BÀI TOÁN.

Physical Machine	Số lượng nhân CPU	[16-64]
	Dung lượng Ram (GB)	[8-64]
	Dung lượng ổ cứng (GB)	[500-1000]
	Độ trễ ảo hóa (ms)	[5-10]
	Độ trễ lan truyền trên liên kết vật lý (ms)	[5-20]
	Băng thông tại liên kết vật lý (Gbps)	[4-40]
	Chi phí băng thông ($)	[5-10]
	Chi phí tài nguyên cho mỗi CPU ($)	[5-10]
	Chi phí tài nguyên cho mỗi GB Ram ($)	[5-10]
	Chi phí tài nguyên cho mỗi GB ổ cứng ($)	[0.05-0.1]
SFC	Độ trễ tối đa (ms)	[50-300]
	Số lượng VNF ở mỗi SFC	[2-10]
VNF	Tài nguyên CPU yêu cầu	[4-16]
	Tài nguyên RAM yêu cầu (GB)	[4-16]
	Tài nguyên ổ cứng yêu cầu (GB)	[100-200]
	Băng thông yêu cầu (Gbps)	[4-15]
	Độ trễ lan truyền trên liên kết ảo (ms)	[5-10]

Để kiểm tra tính hiệu quả của các phương pháp đưa ra, xét 5 đồ thị khác nhau từ SND Lib, mỗi đồ thị có đặc điểm, yêu cầu khác nhau. Bảng *V* tóm tắt các đồ thị được sử dụng trong nghiên cứu này.

Bảng V
DỮ LIỆU THỰC NGHIỆM.

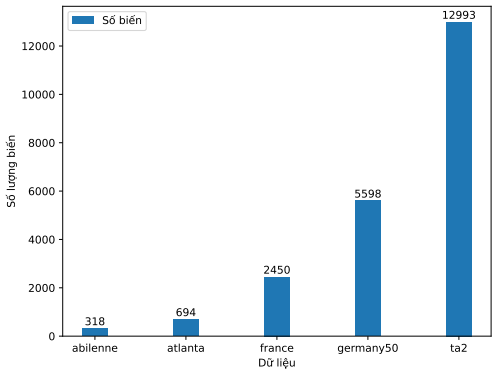
Dữ liệu	Số lượng PM	Số lượng PL	Số lượng SFC	Số lượng VNF
abilene	12	15	2	9
ttlanta	15	22	3	14
france	25	45	6	26
germany 50	50	88	8	30
ta2	65	108	14	57

3. Kết quả thực nghiệm

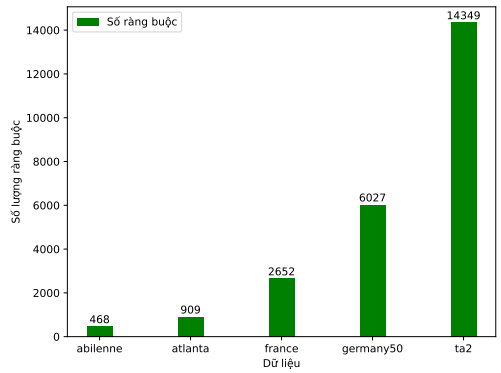
Dựa vào dữ liệu thực nghiệm thì dưới đây là so sánh về số lượng biến và ràng buộc trên mô hình quy hoạch tuyến tính nguyên hỗn hợp (MILP) của các bộ dữ liệu *abilene*, *atlanta*, *france*, *germany50*, *ta2* của SND Lib được mô tả trong Hình 6.

Hình 7, 8 thể hiện kết quả và đánh giá hiệu suất của việc thực thi hai phương pháp trên các bộ dữ liệu *abilene*, *atlanta*, *france*, *germany50*, *ta2* của SND Lib. Dựa vào kết quả thực nghiệm có thể nói rằng thuật toán di truyền tương đối tốt so với việc sử dụng mô hình quy hoạch tuyến tính nguyên hỗn hợp khi giải quyết bài toán. Thuật toán di truyền chỉ kém hơn trung bình khoảng 12% về chi phí và 11% về độ trễ nhưng thời gian chạy nhanh hơn đáng kể.

Hình 9 cho thấy chi phí ($) và độ trễ của mô hình quy hoạch tuyến tính nguyên hỗn hợp (MILP) và thuật toán di

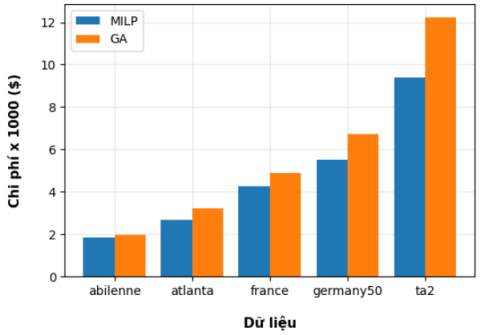


(a) Thống kê số lượng biến.

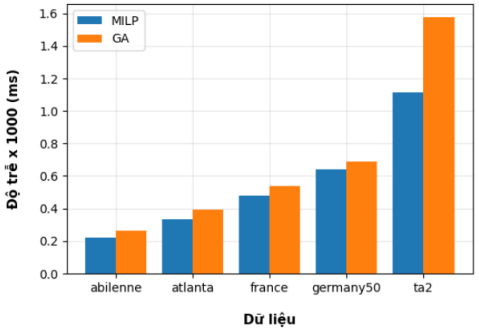


(b) Thống kê số lượng ràng buộc

Hình 6. So sánh về số lượng biến, ràng buộc trên mô hình MILP của các dữ liệu.

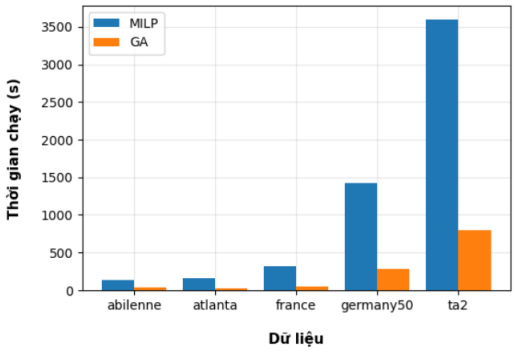


(a) So sánh về chi phí.



(b) So sánh về độ trễ.

Hình 7. So sánh chi phí và độ trễ giữa MILP và thuật toán di truyền (với $\alpha = 0.5$).

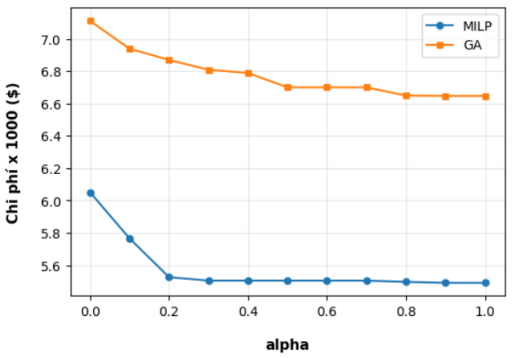


Hình 8. Thời gian chạy giữa MILP và thuật toán GA.

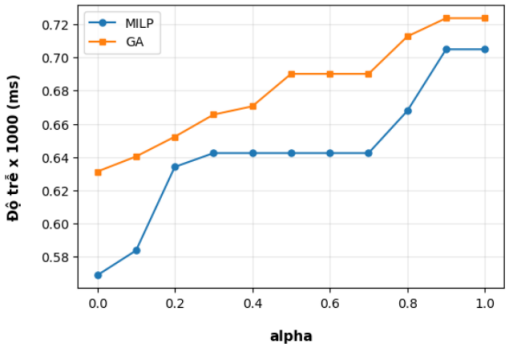
truyền (GA) cho các giá trị khác nhau trong khoảng từ 0 đến 1. Nhìn chung, cả hai giải pháp đều thu được kết quả gần đúng ở tất cả các giá trị nhưng MILP tốt hơn trong mọi trường hợp không quá 12% xảy ra với $\alpha = 0.1$. Chi phí giảm khi tăng α. Với α là trọng số nhấn mạnh vào mức tối ưu chi phí hơn là độ trễ, do đó, khi giá trị α cao hơn, thuật toán sẽ cố gắng tìm giải pháp có chi phí nhỏ hơn. Ngược lại, khi giá trị α nhỏ hơn, thuật toán sẽ cố gắng tìm giải pháp có độ trễ nhỏ hơn. Do đó, nếu $\alpha = 1$ thì có giải pháp mà chi phí là nhỏ nhất, nếu $\alpha = 0$ thì có giải pháp mà độ trễ là nhỏ nhất. Như vậy, cần chọn giá trị α một cách phù hợp để cân bằng giữa tối ưu chi phí và độ trễ hoặc theo yêu cầu phía người dùng.

VI. KẾT LUẬN

Nghiên cứu đã tiến hành một phân tích sâu rộng về bài toán đặt và định tuyến chuỗi chức năng dịch vụ với mục tiêu tối ưu hóa cả chi phí và độ trễ, đồng thời phải thỏa mãn các ràng buộc về tài nguyên, luồng, độ trễ và băng thông. Kết quả của nghiên cứu đã chỉ ra rằng bài toán này không chỉ khó khăn với phiên bản đơn giản, mà còn đặt ra những thách thức nổi bật. Để giải quyết bài toán này, nghiên cứu đã sử dụng mô hình quy hoạch tuyến tính nguyên hỗn hợp để tìm được lời giải chính xác. Đặc biệt, thuật toán di truyền đã được đề xuất và triển khai để giải quyết bài toán trong trường hợp các bộ dữ liệu có kích thước lớn. Thuật toán được thử nghiệm trên nhiều bộ dữ liệu khác nhau để chứng minh hiệu suất và độ chính xác của nó. Kết quả thực nghiệm đã minh họa sự hiệu quả của thuật toán đề xuất trong việc giải quyết bài toán đặt và định tuyến



(a) So sánh về chi phí.



(b) So sánh về độ trễ.

Hình 9. Chi phí và độ trễ giữa MILP và thuật toán di truyền các giá trị α khác nhau (Dữ liệu germany50).

cho các hệ thống mạng phức tạp. Thời gian tới, chúng tôi sẽ mở rộng nghiên cứu bài toán, tập trung vào đặt và định tuyến chức năng mạng ảo trong môi trường mạng thay đổi liên tục. Bài toán động đặt ra thách thức lớn hơn và làm tăng tính ứng dụng và tính linh hoạt của giải pháp thực tế.

LỜI CẢM ƠN

Nghiên cứu này được tiến hành trong khuôn khổ đề tài QG.23.05 của ĐHQGHN.

TÀI LIỆU THAM KHẢO

[1] Q. Zhang, Q. Zhu, M. F. Zhani, R. Boutaba, and J. L. Hellerstein, "Dynamic service placement in geographically distributed clouds," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 12, pp. 762–772, 2013.

[2] J. W. Jiang, T. Lan, S. Ha, M. Chen, and M. Chiang, "Joint vm placement and routing for data center traffic engineering," in *2012 Proceedings IEEE INFOCOM*. IEEE, 2012, pp. 2876–2880.

[3] S. Yang, P. Wieder, R. Yahyapour, S. Trajanovski, and X. Fu, "Reliable virtual machine placement and routing in clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 10, pp. 2965–2978, 2017.

[4] 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.

[5] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 1346–1354.

[6] M. Ghaznavi, N. Shahriar, S. Kamali, R. Ahmed, and R. Boutaba, "Distributed service function chaining," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2479–2489, 2017.

[7] 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.

[8] A. Murray, A. Arulselvan, M. Roper, M. Cashmore, S. K. Mohalik, I. Burdick, and S. David, "The cost of quality of service: Sla aware vnf placement and routing using column generation," in *2023 13th International Workshop on Resilient Networks Design and Modeling (RNDM)*. IEEE, 2023, pp. 1–8.

[9] M. Golkarifard, C. F. Chiasserini, F. Malandrino, and A. Movaghar, "Dynamic vnf placement, resource allocation and traffic routing in 5g," *Computer Networks*, vol. 188, p. 107830, 2021.

[10] T.-W. Kuo, B.-H. Liou, K. C.-J. Lin, and M.-J. Tsai, "Deploying chains of virtual network functions: On the relation between link and server usage," *IEEE/ACM Transactions On Networking*, vol. 26, no. 4, pp. 1562–1576, 2018.

[11] 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.

[12] C. You *et al.*, "Efficient load balancing for the vnf deployment with placement constraints," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[13] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient vnf placement for service chaining: Joint sampling and matching approach," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 172–185, 2017.

[14] 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.

[15] G. Sallam, G. R. Gupta, B. Li, and B. Ji, "Shortest path and maximum flow problems under service function chaining constraints," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 2132–2140.

[16] 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.

[17] 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.

[18] Z. Zhang, Z. Li, C. Wu, and C. Huang, "A scalable and distributed approach for nfv service chain cost minimization," in *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*. IEEE, 2017, pp. 2151–2156.

[19] Q. Li, Y. Jiang, P. Duan, M. Xu, and X. Xiao, "Quokka: Latency-aware middlebox scheduling with dynamic resource allocation," *Journal of Network and Computer Applications*, vol. 78, pp. 253–266, 2017.

[20] 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.

[21] 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.

[22] 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.

[23] H. Chen, X. Wang, Y. Zhao, T. Song, Y. Wang, S. Xu, and L. Li, "Mosc: A method to assign the outsourcing of service function chain across multiple clouds," *Computer Networks*, vol. 133, pp. 166–182, 2018.

[24] 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.

Lê Đăng Nguyên tốt nghiệp Đại học năm 1996 tại Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Hà Nội, chuyên ngành Tin học, Thạc sĩ năm 2005 tại Trường Đại học Công nghệ, ĐHQGHN chuyên ngành Công nghệ phần mềm, Tiến sĩ năm 2016 tại Trường Đại học Khoa học Tự nhiên, ĐHQGHN, chuyên ngành Cơ sở Toán học cho Tin học. Hiện là giảng viên chính tại Khoa Công nghệ Thông tin, Trường Đại học Hải Phòng. Hướng nghiên cứu chính: thuật toán tiến hóa giải các bài toán tối ưu tổ hợp.
Email: nguyenld@dhhp.edu.vn

Trần Bá Tuấn Nhận bằng Cử nhân ngành Máy tính và khoa học thông tin tại Trường Đại học Khoa học Tự nhiên vào năm 2022, hiện đang công tác tại Khoa Toán - Cơ - Tin học, Trường Đại học Khoa học Tự nhiên, ĐHQGHN. Hướng nghiên cứu chủ yếu bao gồm Khoa học máy tính, Trí tuệ nhân tạo.

Email: batuan24122@gmail.com

Nguyễn Thị Tâm tốt nghiệp Thạc sĩ chuyên ngành Cơ sở toán cho tin học tại Đại học Khoa học Tự nhiên vào năm 2015. Tiến sĩ chuyên ngành Khoa học máy tính tại Đại học Bách khoa Hà Nội vào năm 2022. Hiện là đang là giảng viên tại Trường Đại học Khoa học Tự nhiên, ĐHQGHN. Hướng nghiên cứu chính: Computational Intelligence, Artificial Intelligence, và Memetic Computing.
Email: tamnt@vnu.edu.vn

Lê Trọng Vĩnh tốt nghiệp Đại học năm 1994 tại trường Đại học Tổng hợp Hà Nội, chuyên ngành Tin học, Thạc sĩ năm 1997 tại trường ĐH KHTN, ĐHQG Hà Nội chuyên ngành Đảm bảo Toán học cho Máy tính và các Hệ thống tính toán, Tiến sĩ năm 2006 tại Viện Khoa học và Công nghệ Tiên tiến Nhật Bản chuyên ngành Công nghệ Thông tin. Được phong Phó Giáo sư năm 2012. Hiện là giảng viên cao cấp tại Khoa Toán - Cơ - Tin học, Trường Đại học Khoa học Tự nhiên, ĐHQGHN. Hướng nghiên cứu chính là các thuật toán tiến hóa giải các bài toán tối ưu tổ hợp, đặc biệt là các bài toán liên quan đến mạng máy tính.
Email: vinhlt@gmail.com