PERFORMANCE EVALUATION OF SERVICE FUNCTIONS CHAIN PLACEMENT ALGORITHMS IN EDGE CLOUD

Dinh Xuan Lam*, Duong Thuy Huong
TNU - University of Information and Communication Technology

ARTICLE INFO

Received: 12/10/2021
Revised: 29/11/2021
Published: 30/11/2021

KEYWORDS

Network Function Virtualization
Service Function Chain
Placement
Optimization
Edge Cloud

ABSTRACT

The emergence of the Network Function Virtualization (NFV) paradigm has become a potential solution dealing with the rapid growth of global Internet traffic in the last decades. There, network appliances are transformed into Virtual Network Functions (VNF) running on a standard server. This promises to significantly reduce overall cost and energy consumption. Additionally, a hardware-based network function chain is replaced by a chain of the VNFs, called Service Function Chain (SFC). The expected benefit of SFC is the reduction in complexity when deploying heterogeneous network services. However, the considerable drawback of SFC is the distribution of the VNFs over different hosts. An inefficient placement of VNFs can induce a high latency within the chain and wasted server resources. In this work, we design four placement algorithms that aim to efficiently place the SFC in servers with regard to minimizing service response time and resource utilization. Herein, heuristic approaches are evaluated against optimal solutions for the placement problems, which are formulated by using Integer Linear Programming. We evaluate and compare these placement strategies in a simulator. Our result shows that the optimized solutions produce the lowest service response time and least server utilization in all types of simulated SFCs. On the other hand, the heuristic algorithms are also able to come close to the optimum by simple placing rules.

MÔ PHỎNG VÀ ĐÁNH GIÁ MỘT SỐ THUẬT TOÁN PHÂN BỐ VỊ TRÍ CỦA CHUỖI CHỨC NĂNG MẠNG ẢO HOÁ TRONG MÔ HÌNH ĐIỆN TOÁN BIÊN

Đinh Xuân Lâm*, Dương Thúy Hường
Trường Đại học Công nghệ thông tin và Truyền thông – ĐH Thái Nguyên

THÔNG TIN BÀI BÁO

Ngày nhận bài: 12/10/2021
Ngày hoàn thiện: 29/11/2021
Ngày đăng: 30/11/2021

TỪ KHÓA

Mạng ảo hóa
Chuỗi chức năng
Thuật toán phân bố vị trí
Tối ưu hóa
Điện toán biên

TÓM TẮT

Sự xuất hiện của mô hình NFV đã trở thành một giải pháp tiềm năng đối phó với tốc độ tăng trưởng nhanh chóng của lưu lượng Internet toàn cầu trong những thập kỷ qua. Ở đó, các thiết bị mạng được chuyển đổi thành chức năng mạng ảo (VNF) chạy trên một máy chủ tiêu chuẩn. Ngoài ra, một chuỗi chức năng mạng dựa trên phần cứng được thay thế bằng một chuỗi VNF, được gọi là chuỗi chức năng dịch vụ (SFC). Lợi ích của SFC là giảm độ phức tạp khi triển khai các dịch vụ mạng không đồng nhất. Trong bài báo này, tác giả đã nghiên cứu và xây dựng bốn thuật toán nhằm sắp xếp và đặt SFC một cách hiệu quả trong các máy chủ nhằm đến việc giảm thiểu thời gian đáp ứng dịch vụ và sử dụng tài nguyên. Ở đây, các thuật toán truyền thống được so sánh với một mô hình tính toán tối ưu dựa trên quy hoạch tuyến tính số nguyên. Tác giả đánh giá và so sánh các chiến lược phân bố vị trí này trong một chương trình mô phỏng. Kết quả cho thấy, các giải pháp tối ưu tạo ra thời gian đáp ứng dịch vụ thấp nhất và sử dụng máy chủ ít nhất trong tất cả các kịch bản mô phỏng. Mặt khác, các thuật toán truyền thống cũng có thể chấp nhận được bằng các quy tắc phân bổ dựa trên thuật toán sắp xếp đơn giản.

DOI: https://doi.org/10.34238/tnu-jst.5142

* Corresponding author. *Email: dxlam@ictu.edu.vn*

1. Giới thiệu

Sự phát triển bùng nổ của các dịch vụ mạng Internet và công nghệ truyền thông hiện đại đã mang đến cho người dùng những trải nghiệm tốt hơn. Chính vì vậy, hạ tầng mạng Internet không ngừng được cải tiến để duy trì và đáp ứng tốt hơn nữa nhu cầu ngày càng tăng cao của người dùng cuối. Ngày nay, mạng Internet chủ yếu được xây dựng từ những thiết bị phần cứng trung gian như bộ định tuyến, tường lửa hay cân bằng tải. Những thiết bị phần cứng này thường được lắp đặt tập trung trong trung tâm máy chủ và hoạt động đồng bộ với nhau. Dữ liệu ra vào được xử lý tuần tự tại mỗi thiết bị trước khi được gửi đến thiết bị của người dùng. Mỗi thiết bị trên được gọi là một chức năng mạng hay *network function*. Một chuỗi các chức năng mạng này kết hợp với nhau để cung cấp một dịch vụ nào đó được gọi là chuỗi chức năng dịch vụ hay *Service Function Chain* (SFC) [1], [2]. Trên thực tế, các chuỗi chức năng này được xây dựng bằng các thiết bị phần cứng và nó có nhiều hạn chế như sự đồng bộ hóa của chuỗi khó thay đổi, khó thay thế thiết bị, chi phí đầu tư, vận hành và bảo trì lớn, thời gian lắp đặt, triển khai phần cứng thường kéo dài và mất nhiều công đoạn kiểm thử trước khi được vận hành, khó có khả năng mở rộng, điều này dẫn đến sự chậm trễ triển khai dịch vụ mới đến người dùng.

Để khắc phục những hạn chế này, một công nghệ mạng mới và thông minh hơn đã ra đời được gọi là *Network Functions Virtualization* (NFV) [3]-[5]. Khái niệm về NFV lần đầu tiên được giới thiệu trong hội nghị "*Software - Defined Networking (SDN) and OpenFlow*" vào tháng 10 năm 2012. NFV hướng tới mục tiêu giải quyết những vấn đề trên bằng cách ứng dụng công nghệ ảo hóa. NFV là một cách tiếp cận mới trong mô hình kiến trúc của mạng truyền thông, trong đó các chức năng mạng được phần mềm hóa và có thể triển khai động ở hầu hết các vị trí trong mạng máy tính, thay thế các thiết bị mạng chuyên dụng nằm ở các vị trí được định trước. Các phần mềm này được gọi là *Virtual Network Function* (VNF). Ý tưởng chính của mô hình này là tách các chức năng mạng (VNF) khỏi phần cứng vật lý của chúng. Các chức năng mạng được triển khai bởi các phần mềm có thể chạy đồng thời trên nhiều chủng loại phần cứng máy chủ, có khả năng di chuyển, khởi tạo ở các vị trí khác nhau trong mạng theo yêu cầu mà không cần phải lắp đặt thiết bị phần cứng mới [6], [7]. Đặc biệt, nhiều chức năng khác nhau có thể dễ dàng kết hợp tạo thành chuỗi SFC để phục vụ nhiệm vụ nào đó.

Các dịch vụ được triển khai dưới dạng chuỗi chức năng dịch vụ SFC là sự kết hợp của một số chức năng cơ bản, thường chạy trong một số dạng môi trường ảo (máy ảo hoặc Docker[1]). Về cơ bản, 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. Hiện nay, nhiều cấu hình mạng ảo hóa có thể cùng tồn tại trên cùng một cơ sở hạ tầng vật lý. Với công nghệ NFV, các SFC dựa trên phần mềm có thể được khởi tạo, kiểm soát, cập nhật dễ dàng bằng cách cài đặt trực tiếp trên máy chủ, điều đó làm giảm thiểu thời gian phản hồi của dịch vụ. Ngược lại, một chuỗi chức năng phần cứng muốn nâng cấp thường phải thay thế thiết bị và cấu hình lại cấu trúc vật lý một cách thủ công [8], [9].

Tuy nhiên, một trong những vấn đề chính liên quan đến việc triển khai SFC là việc xác định và áp dụng các cấu hình SFC là khá phức tạp. Thứ nhất, mỗi chức năng mạng đơn lẻ VNF trong một chuỗi SFC có thể phải phân bố tại các máy chủ vật lý khác nhau trong mạng dẫn tới chiều dài của chuỗi tăng lên. Điều này trực tiếp làm tăng độ trễ truyền dữ liệu trong toàn bộ chuỗi gây giảm hiệu năng mạng. Thứ hai, các VNF thuộc các chuỗi khác nhau có thể được triển khai trong cùng một máy chủ và chiếm tài nguyên máy chủ này, tuy nhiên một số VNF không tồn tại liên tục mà có thể được tắt đi và khởi tạo nhiều lần dẫn tới việc tối ưu hóa tài nguyên máy chủ là một vấn đề quan trọng. Các vấn đề trên cho thấy cần phải giải một bài toán tối ưu hóa sự phân bố các VNF trên một hoặc nhiều máy chủ khác nhau để đảm bảo độ trễ tối thiểu hoặc sử dụng tối đa tài nguyên máy chủ tránh lãng phí.

2. Phương pháp nghiên cứu và thuật toán

[1] https://www.docker.com

2.1. Đặc tính kỹ thuật của chuỗi SFC

Để giải bài toán tối ưu hóa độ trễ đầu cuối và tối ưu hóa tài nguyên máy chủ, tác giả đã nghiên cứu và xây dựng bốn thuật toán bao gồm hai thuật toán sắp xếp đơn giản có tên là: "*Sắp xếp tập trung - Centralization"* và *"Sắp xếp điều phối - Orchestration"*, và hai hàm mục tiêu thuộc mô hình tính toán tối ưu dựa trên quy hoạch tuyến tính số nguyên *(Integer Linear Programming - ILP)*. Các thuật toán này được thiết kế nhằm mục đích phân bố tự động các VNF trong các máy chủ để giảm thiểu độ trễ toàn bộ chuỗi hoặc tối ưu tài nguyên máy chủ. Để thực hiện mô phỏng các chuỗi SFC có 3 VNF thành phần, giải bài toán ILP tự động và so sánh tính hiệu quả của các thuật toán trên, tác giả đã cải tiến phần mềm mô phỏng EdgeNetworkCloudSim [10] để tích hợp công cụ giải bài toán ILP bằng mô-đun CPLEX Optimizer.

2.2. Các thuật toán phân bố vị trí chuỗi chức năng mạng SFC

Trong phần này tác giả giới thiệu bốn thuật toán phân bố vị trí chuỗi chức năng mạng SFC, đó là thuật toán sắp xếp tập trung *Centralization* (CEN), thuật toán sắp xếp điều phối *Orchestration* (ORC), tối ưu hóa thời gian dịch vụ *Service Time Optimization* (STO) và tối ưu hóa tài nguyên *Resource Optimization* (RO).

2.2.1. Thuật toán sắp xếp tập trung Centralization Algorithm và sắp xếp điều phối Orchestration

Thuật toán sắp xếp tập trung (CEN) cố gắng đặt tất cả các máy ảo của một chuỗi SFC càng gần người dùng càng tốt, nghĩa là thuật toán này có độ trễ thấp nhất giữa các máy ảo và người dùng. Thuật toán CEN hữu ích trong điện toán đám mây cổ điển, nơi các máy ảo độc lập của người dùng phải được đặt gần người dùng nhất. Tuy các máy ảo có thể gần người dùng nhất, nhưng độ dài của chuỗi có thể là điểm yếu lớn nhất của thuật toán này.

Thuật toán sắp xếp điều phối (ORC) khác với thuật toán sắp xếp tập trung CEN, trong đó chỉ máy ảo đầu tiên trong chuỗi được cố gắng đặt càng gần người dùng càng tốt. Các máy ảo tiếp theo được đặt càng gần máy ảo trước đó càng tốt. Dựa trên điều này, ORC cố gắng rút ngắn độ dài của chuỗi để giảm độ trễ trong chuỗi.

2.2.2. Thuật toán tối ưu hóa thời gian đáp ứng dịch vụ và tối ưu hóa tài nguyên

Bảng 1 trình bày các ký hiệu được sử dụng để xây dựng bài toán tối ưu hóa.

Bảng 1. *Tóm tắt các tham số được sử dụng trong mô hình ILP*

Tham số	Mô tả
T	Tập hợp các thành phần của ứng dụng
C	Tập hợp các kênh giữa các thành phần của ứng dụng, $C \subseteq T \times T$
H	Tập hợp các máy chủ trong một trung tâm máy chủ
L	Tập hợp các liên kết giữa các máy chủ, $L \subseteq H \times H$
S	Tập hợp các thành phần của ứng dụng người dùng được cấp phát tĩnh, $S \subset T$
H^S	Tập hợp các máy của người dùng để chứa các thành phần ứng dụng người dùng $H^S \subset H, f: S \rightarrow HS \mid \forall h' \in H^S, \exists t' \in S: h' = f(t')$ (f is subjective)
R	Tập hợp các tài nguyên của máy chủ
R'	Tập hợp các tài nguyên của các liên kết
M	Tập hợp các tiêu chí giám sát tài nguyên tại máy chủ
M'	Tập hợp các tiêu chí giám sát tài nguyên tại các liên kết
a_t^r	Lượng tài nguyên r được yêu cầu bởi thành phần ứng dụng t
i_h^r	Dung lượng tài nguyên r tại máy chủ h
β_h^r	Lượng tài nguyên r sẵn sàng tại máy chủ h
m_h^k	Giá trị đo được của tiêu chí k tại máy chủ h
c_{sd}^r	Lượng tài nguyên r được yêu cầu bởi kênh (s, d)
b_{uv}^r	Lượng tài nguyên r có sẵn tại liên kết (u, v)
μ_{uv}^k	Giá trị đo được của tiêu chí k tại liên kết (u, v)

Dựa trên các ký hiệu đã trình bày tại Bảng 1 và thuộc tính của chuỗi SFC, bài toán tối ưu hóa được xây dựng như dưới đây. Tài nguyên máy chủ được định nghĩa, bao gồm:

R = {CP U, Memory}

R' = {Bandwidth}

Hàm mục tiêu 1: Thời gian đáp ứng dịch vụ (STO) của chuỗi dịch vụ là nhỏ nhất:

$$Objective1 = \sum_{(s,d)\in C} \pi_{uv,sd}\, \mu_{uv}^{Delay},\ \forall (u,v) \in L. \quad (1)$$

Hàm mục tiêu 2: Sử dụng tài nguyên (RO) của chuỗi dịch vụ là nhỏ nhất:

$$Objective2 = \sum_{h\in H} (min\{\sum_{t\in T} \sigma_{ht}, 1\} \frac{100\beta_h^{CPU}}{i_h^{CPU}}). \quad (2)$$

Mục tiêu (2) nhằm mục đích tối ưu hóa số lượng máy chủ được sử dụng tại một vị trí và tỷ lệ với CPU khả dụng. Quy trình tối ưu hóa này sẽ cố gắng sắp xếp các máy ảo trong một máy chủ nhưng sẽ có ưu tiên đối với một máy chủ đang được sử dụng và vẫn còn tài nguyên thừa. Do đó, các máy chủ đang không sử dụng sẽ không được kích hoạt cho đến khi các máy chủ khác đã được sử dụng hết. Bằng cách này, các máy chủ không sử dụng có thể được đặt ở trạng thái chờ để tiết kiệm năng lượng. Tuy nhiên, tại lúc khởi tạo, tất cả các máy chủ sẽ có cùng xác suất được chọn.

Điều kiện và rằng buộc:

$$\pi_{uv,sd} \in \{0, 1\},\ (s, d) \in C,\ (u, v) \in L. \quad (3)$$

Trong phương trình (3), $\pi_{uv,sd}$ là biến quyết định và bằng 1 nếu kênh dữ liệu (*s, d*) được định tuyến từ liên kết (*u, v*), ngược lại là bằng 0.

$$\sigma_{ht} \in \{0, 1\},\ h \in H,\ t \in T. \quad (4)$$

Trong ràng buộc (4), σ_{ht} là một biến quyết định và bằng 1 nếu nhiệm vụ *t* được gán cho máy chủ *h*, ngược lại là bằng 0.

$$\sum\nolimits_{h\in H} \sigma_{ht} = 1,\ \forall t \in T, \quad (5)$$

$$\sigma_{ht} = 1,\ h \in H^S,\ t \in S, \quad (6)$$

$$\sum_{t\in T\setminus S} \sigma_{ht} = 0,\ \forall h \in H^S, \quad (7)$$

Ràng buộc (5) đảm bảo rằng một tác vụ (hoặc thành phần ứng dụng) chỉ được gán cho một máy chủ. Vị trí tĩnh của tác vụ người dùng được định nghĩa trong phương trình (6) được đưa ra như một đầu vào cho bài toán và không được quyết định. Trong khi, ràng buộc (7) chỉ ra rằng các ứng dụng của người dùng chỉ được đặt trong các máy chủ đã được chỉ định trước đó.

$$\sum_{t\in T} \sigma_{ht}\, a_t^r \le \beta_h^r,\ \forall r \in R,\ \forall h \in H^S. \quad (8)$$

Công thức (8) ràng buộc rằng máy chủ *h* phải có đủ tài nguyên để cấp phát thành phần cho ứng dụng *t*.

$$\sum\nolimits_{(u,h)\in L} \pi_{uh,sd} + \sigma_{hs} = \sum\nolimits_{(h,v)\in L} \pi_{hv,sd} + \sigma_{hd} \quad (9)$$

Ràng buộc (9) thể hiện trong một phương trình các nội dung sau:

- Ràng buộc luồng không thể phân chia: Một kênh sử dụng một liên kết đi duy nhất từ nguồn và một liên kết đến duy nhất tại đích và không phân chia,

$$\sum_{(u,h)\in L} \pi_{uh,sd} = 1\ if\ \sigma_{us} = 1,$$

$$\sum_{(h,v)\in L} \pi_{hv,sd} = 1 \; if \; \sigma_{vd} \; = \; 1,$$

• Sự sắp xếp các nhiệm vụ: Không bắt buộc phải có đường truyền trong trường hợp cả *s* và *d* được gán cho cùng một máy chủ (và không có kiểm tra dung lượng),

$$\sigma_{hs} = \sigma_{hd},$$
$$\pi_{uu,sd} = 0,$$

• Ràng buộc bảo toàn luồng: Không có lưu lượng nào được lưu trữ trong một nút trừ khi nút này là nguồn hoặc đích hoặc nguồn và đích được sắp xếp,

$$\sum_{(u,h)\in L} \pi_{uh,sd} = \sum_{(h,v)\in L} \pi_{hv,sd}$$
$$\forall h \in H : \sigma_{hs} = 0, \sigma_{hd} = 0.$$
$$\sum\nolimits_{(u,h)\in L} \sigma_{hs}\pi_{uh,sd} = 0. \quad (10)$$

Ràng buộc (10) đảm bảo rằng không có vòng lặp nào trong đường dẫn trước khi đến đích,

$$\pi_{uv,sd} = \pi_{vu,sd}, (s,d), (d,s) \in C, (u,v), (v,u) \in L. \quad (11)$$

Như được xác định trong Công thức (11), giao tiếp hai chiều giữa hai tác vụ được định tuyến thông qua cùng một đường hai chiều. Thêm vào đó, nguồn và đích của luồng không được định tuyến riêng biệt.

$$\sum\nolimits_{(s,d)\in C} \pi_{uv,sd} \, C_{sd} \le b_{uv}, \;\; \forall (u,v) \in L. \quad (12)$$

Ràng buộc (12) đảm bảo rằng liên kết (*u*, *v*) phải có đủ tài nguyên băng thông theo yêu cầu của các kênh (*s*, *d*).

3. Phương pháp mô phỏng

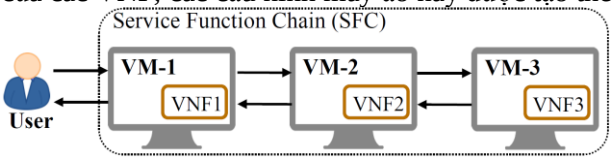
3.1. Mô hình điện toán biên

Như đã trình bày trong phần trước, tác giả hướng tới mô hình điện toán biên cho các kịch bản mô phỏng. Điện toán biên là một mô hình điện toán phân tán, đưa việc xử lý tính toán và lưu trữ dữ liệu đến gần vị trí cần thiết hơn để nâng cao tốc độ và tiết kiệm băng thông *(Edge Computing)* [11]. Điện toán biên là một mạng lưới các trung tâm xử lý và lưu trữ dữ liệu cục bộ trước khi dữ liệu được gửi đến trung tâm dữ liệu hoặc đưa lên điện toán đám mây *(cloud)*. Nó tối ưu hóa các hệ thống truyền dẫn để tránh gián đoạn hoặc làm chậm việc gửi và nhận dữ liệu. Mọi thứ được tính toán để xử lý ngay tại các biên *(edge)* của hệ thống mạng. Như vậy, mô hình có thể loại bỏ phần lớn hiện tượng tắc nghẽn mạng và giảm rất nhiều độ trễ đầu cuối thường thấy trong các mô hình điện toán đám mây truyền thống. Trong nghiên cứu này, tác giả xây dựng kịch bản mô phỏng thiết bị của người dùng kết nối trực tiếp với máy chủ điện toán biên.

3.2. Phần mềm mô phỏng

Trong nghiên cứu này, tác giả sử dụng phần mềm EdgeNetworkCloudSim để mô phỏng một cấu trúc liên kết mạng cố định trong mô hình điện toán biên. Trong đó, các máy chủ có thể được phân bố tại các vị trí địa lý khác nhau gần người dùng và được thiết lập để sẵn sàng vận hành các chuỗi dịch vụ SFC được thiết lập trước bao gồm *Video streaming*, *Database* hoặc dịch vụ *Web*, mỗi dịch vụ này được đặc trưng bởi tài nguyên máy chủ nó cần như số lượng CPU và bộ nhớ RAM. Người dùng được mô phỏng gửi các yêu cầu ngẫu nhiên tới chuỗi dịch vụ bao gồm ba VNF được đặt trên các máy chủ khác nhau. Các thuật toán phân bố vị trí VNF sau đó tính toán và tự động đặt các phần mềm này tại các máy chủ khả dụng để tối ưu độ trễ hoặc tài nguyên.

Hình 1 biểu diễn mô hình kết nối của chuỗi SFC trên phần mềm mô phỏng bao gồm 3 máy ảo có tên là VM-1, VM-2, VM-3 các máy này được cài đặt để chứa các chức năng mạng ảo hóa VNF. Bảng 2 trình bày các cấu hình máy ảo khác nhau căn cứ theo nhu cầu sử dụng tài nguyên của các VNF, các cấu hình máy ảo này được tạo theo tiêu chuẩn của Amazon EC2 Cloud[2].



Hình 1. *Mô hình kết nối của các VNF trong chuỗi SFC*

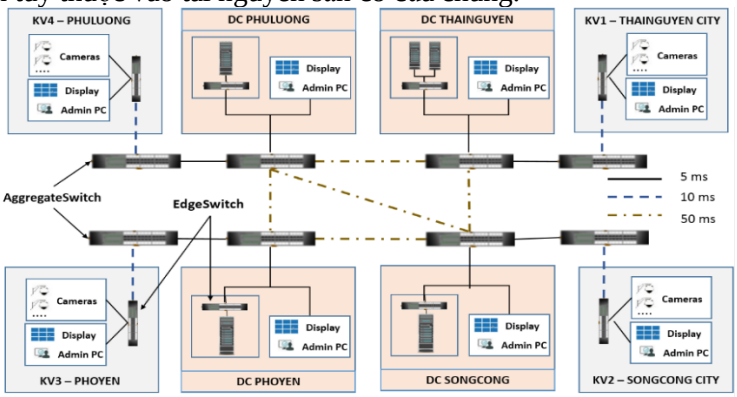
Bảng 2. *Cấu hình các máy ảo trong mô phỏng*

VM Type	CPU	RAM
T2Nano	1	1024MB
T2Small	2	2048MB
T2Large	4	4096MB

Theo mô hình kết nối ở Hình 1, dữ liệu yêu cầu của người dùng được truyền theo chiều mũi tên đến lần lượt các máy ảo chứa VNF để xử lý tuần tự, dữ liệu phản hồi từ dịch vụ cũng được truyền ngược lại lần lượt qua các VNF đến người dùng. Cấu hình này hoàn toàn tương tự như cấu hình tại cái máy chủ và chức năng mạng vật lý trên thực tế [1].

3.3. Kịch bản mô phỏng

Trong chương này, giả định rằng một hệ thống camera giám sát giao thông thông minh được triển khai tại một số khu vực có mật độ giao thông lớn bao gồm 4 đơn vị hành chính thuộc địa phận tỉnh Thái Nguyên, bao gồm thành phố Thái Nguyên, thành phố Sông Công, thị xã Phổ Yên và huyện Phú Lương. Mỗi khu vực trong hệ thống camera giám sát giao thông này bao gồm nhiều camera, các thiết bị này được quản lý bởi máy tính thông thường và liên tục gửi dữ liệu *Video*, *Data*base và *Web* lên các trung tâm máy chủ điện toán biên là DC PHULUONG, DC THAINGUYEN, DC PHOYEN và DC SONGCONG. Hình 2 thể hiện cấu trúc liên kết trong mô phỏng (topology). Trong topology này, các máy ảo của SFC được phân phối trên các máy chủ của điện toán biên tùy thuộc vào tài nguyên sẵn có của chúng.



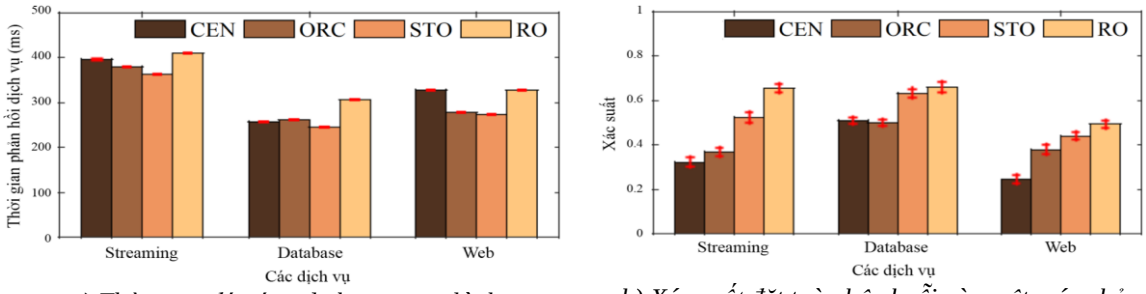
Hình 2. *Cấu trúc liên kết điện toán biên trong mô phỏng*

3.4. So sánh các thuật toán phân bố vị trí về độ trễ đầu cuối

Hình 3a cho thấy thời gian đáp ứng dịch vụ trung bình của ba SFC với các thuật toán phân bố vị trí khác nhau. Trục *x* cho biết ba dịch vụ, trục *y* thể hiện thời gian phản hồi trung bình của dịch vụ tính bằng mili giây. Các cột trong biểu đồ với các màu khác nhau thể hiện cho thời gian phản hồi trung bình của dịch vụ theo từng thuật toán phân bố vị trí khác nhau với độ tin cậy 95%.

Hình 3a cho thấy việc gửi nhiều đoạn video làm tăng thời gian phản hồi trung bình của một chuỗi dịch vụ *Video Streaming* [12]. Ngược lại, dịch vụ *Database* có thời gian phản hồi trung bình thấp nhất khoảng 250ms. Vì nó yêu cầu tổng kích thước của tất cả các máy ảo thấp hơn, các thuật toán phân bố vị trí có cơ hội cao để đặt các máy ảo trong một máy chủ mong muốn, điều này làm giảm thời gian phản hồi dịch vụ tổng thể.

[2] https://aws.amazon.com/ec2/instance-types



a) Thời gian đáp ứng dịch vụ trung bình *b) Xác suất đặt toàn bộ chuỗi vào một máy chủ*

Hình 3. *Thời gian đáp ứng dịch vụ trung bình và xác suất phân bố của các SFC*

Về thời gian phản hồi dịch vụ được tạo ra bởi các thuật toán phân bố vị trí khác nhau, STO đạt được thời gian phản hồi dịch vụ thấp nhất trong tất cả các dịch vụ, tiếp theo là ORC, CEN và RO. Ví dụ, khi sử dụng STO làm thuật toán phân bố vị trí cho dịch vụ Web, ta chỉ mất 272,8ms từ khi người dùng yêu cầu dịch vụ cho đến khi nhận được phản hồi của nó. Trong khi đó, bằng cách sử dụng các thuật toán ORC, CEN hoặc RO, mất lần lượt là 277,8ms, 328ms và 327,1ms cho mỗi phản hồi. Điều này không khó hiểu vì thuật toán STO được thiết kế để tính toán vị trí cho mục tiêu giảm thiểu thời gian đáp ứng dịch vụ bằng cách sử dụng mô hình ILP và xem xét trạng thái toàn hệ thống. Lưu ý rằng, trong topology của chúng ta chỉ có bốn trung tâm dữ liệu, nên thời gian xử lý của mô-đun CPLEX Optimizer là không đáng kể. Tuy nhiên, với cấu trúc liên kết mạng lớn hơn, không gian nghiệm được tạo bởi trình tối ưu hóa CPLEX cũng lớn, thời gian tìm ra vị trí phân bố tối ưu trong chuỗi có thể làm tăng thời gian phản hồi dịch vụ tổng thể. Vì vậy, thời gian giải bài toán ILP có thể ảnh hưởng tiêu cực tới tổng thời gian phản hồi dịch vụ, bởi vì mô hình ILP trong phạm vi bài báo này một dạng bài toán *NP-khó*.

Chính vì vậy, các phương pháp tiếp cận dựa trên kinh nghiệm cũng phải được nghiên cứu, chẳng hạn như ORC đạt thời gian đáp ứng dịch vụ thấp thứ hai cho các dịch vụ Streaming và Web. Thuật toán ORC cố gắng giảm thiểu độ dài của chuỗi, tất cả các máy ảo trong chuỗi được đặt sao cho khoảng cách giữa chúng ngắn nhất. Máy ảo đầu tiên trong chuỗi được đặt càng gần người dùng càng tốt. Trên thực tế, giải thuật ORC phải kiểm tra tất cả các máy chủ với nhiều vòng lặp để tìm vị trí tốt nhất cho các máy ảo. Các máy chủ được chọn cũng phải có đủ tài nguyên cho tất cả các máy ảo. Thao tác kiểm tra này được thực hiện mỗi lần cho một dịch vụ mới được yêu cầu khởi tạo. Do đó, với số lượng yêu cầu ngày càng tăng từ người dùng, nó cũng ảnh hưởng đến thời gian phản hồi dịch vụ tổng thể.

Trên thực tế, vị trí lý tưởng cho thời gian phản hồi dịch vụ thấp nhất là khi tất cả các máy ảo trong chuỗi được đặt trong một máy chủ. Trong trường hợp này, độ trễ giữa ba máy ảo trong chuỗi bằng 0, điều này làm giảm đáng kể thời gian phản hồi dịch vụ tổng thể. Hình 3b cho thấy xác suất xảy ra tình huống này. Trục *x* cho biết ba loại dịch vụ, trục *y* cho biết xác suất có thể đặt toàn bộ chuỗi vào một máy chủ. Các cột với các màu khác nhau đại diện cho giá trị trung bình của các thuật toán khác nhau với khoảng tin cậy 95%.

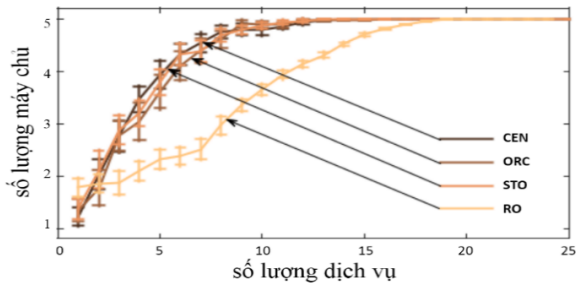
Có thể thấy rằng, thuật toán RO thể hiện tỉ lệ đặt toàn bộ chuỗi vào một máy chủ cao hơn so với các thuật toán khác, vì nó được thiết kế đặc biệt để tối ưu hóa tài nguyên. Điều này được mô tả chi tiết hơn trong phần tiếp theo. Về các thuật toán khác, hình 3b cho thấy STO có giá trị cao hơn đáng kể so với các thuật toán ORC và CEN. Trong dịch vụ Streaming, giải thuật STO có 52,44% khả năng đặt toàn bộ chuỗi dịch vụ vào một máy chủ, trong khi con số này ở ORC và CEN lần lượt là 36,86% và 32,29%. Xu hướng này cũng gặp phải trong dịch vụ Database và Web. Điều này là hợp lý vì giải thuật STO tính toán sắp xếp SFC dựa trên tổng độ trễ tối thiểu. Do đó, việc đặt tất cả các máy ảo trong một máy chủ là một tùy chọn có mức độ ưu tiên cao nhất.

3.5. So sánh các thuật toán phân bố vị trí về mức độ sử dụng tài nguyên máy chủ

Như đã trình bày trong phần trước, mục tiêu thứ hai của mô hình ILP là giảm thiểu việc sử dụng tài nguyên (thuật toán RO) trong việc xác định vị trí phân bố của một SFC. Hình 3b trong

phần trước chỉ ra rằng RO có xác suất đặt tất cả các máy ảo của một SFC vào một máy chủ cao hơn khi so sánh với các thuật toán khác. Để đạt được mục tiêu này, hàm mục tiêu cố gắng giảm thiểu số lượng máy chủ được sử dụng cho SFC liên quan đến tài nguyên sẵn có của chúng. Điều này có nghĩa là thuật toán RO sẽ cố gắng đặt càng nhiều SFC càng tốt trong một máy chủ và nó sẽ ưu tiên lựa các máy chủ đang được sử dụng và vẫn còn tài nguyên thừa. Bằng cách này, các máy chủ không sử dụng có thể được đặt ở chế độ dự phòng hoặc tắt để tiết kiệm năng lượng.

Để đánh giá hiệu suất của thuật toán phân bố vị trí này, chúng ta tính toán số lượng các máy chủ đã sử dụng cùng với số lượng SFC được khởi tạo đồng thời. Ở đây, một máy chủ được coi là được sử dụng khi ít nhất một CPU được cấp cho một máy ảo của một SFC. Hình 4 cho thấy mối tương quan giữa số lượng máy chủ và số lượng dịch vụ cũng chạy đồng thời. Trục *x* hiển thị số lượng dịch vụ đồng thời, trục *y* cho biết số lượng trung bình của các máy chủ được sử dụng tương ứng, có nghĩa là tỷ lệ sử dụng máy chủ. Các đường có màu khác nhau thể hiện tỷ lệ sử dụng máy chủ trung bình được tạo ra bởi các thuật toán phân bố vị trí khác nhau.



Hình 4. *Số lượng máy chủ và dịch vụ sử dụng đồng thời*

Hình 4 cho thấy rằng các thuật toán phân bố vị trí STO, ORC và CEN tạo ra tỷ lệ sử dụng máy chủ tương tự nhau, tất cả các máy chủ được sử dụng khi có nhiều hơn 10 SFC được khởi tạo đồng thời. Điều này là hợp lý, vì các thuật toán này cố gắng giảm thiểu thời gian phản hồi của dịch vụ bằng cách chọn các máy chủ gần nhất với người dùng. Vì người dùng đang gửi yêu cầu từ các vị trí khác nhau như đã thể hiện trong topology, các máy chủ gần nhất nhanh chóng được sử dụng. Tuy nhiên các thuật toán ORC, CEN và STO chiếm dụng máy chủ nhiều hơn nhiều so với RO, thuật toán phân bố vị trí RO có tỷ lệ chiếm dụng máy chủ thấp hơn nhiều, tỉ lệ này được biểu thị bằng đường màu vàng riêng biệt trên biểu đồ Hình 4. Có thể thấy rằng, để khởi tạo trung bình 10 dịch vụ đồng thời, thuật toán RO chỉ sử dụng 3,6 máy chủ. Tất cả các máy chủ được sử dụng hết chỉ khi có từ 21 dịch vụ được khởi tạo cùng một lúc. Để khai thác hết số máy chủ thì số lượng dịch vụ được khởi tạo đồng thời bởi RO cao gấp đôi các thuật toán phân bố vị trí khác. Điều này là do RO là thuật toán luôn ưu tiên sắp xếp các máy ảo trong một máy chủ trước khi xem xét các máy chủ khác. Hơn nữa, khi một SFC đã hoàn thành nhiệm vụ và giải phóng tài nguyên máy chủ, máy chủ này có mức độ ưu tiên được chọn cao hơn cho dịch vụ đến tiếp theo so với những máy chủ đang không sử dụng. Dựa trên điều này, số lượng máy chủ cần dùng sẽ được giảm thiểu và các máy chủ khác có thể được đặt ở trạng thái chờ. Điều này có thể giảm đáng kể mức tiêu thụ điện năng và tiết kiệm năng lượng. Có thể thấy rằng các thuật toán sắp xếp đơn giản dựa trên kinh nghiệm STO, ORC và CEN đã trình bày cho thấy một hiệu suất tối ưu về thời gian đáp ứng dịch vụ, nhưng vẫn cần cải thiện về việc sử dụng tài nguyên.

4. Kết luận

Trong mô hình NFV, việc sử dụng các SFC hứa hẹn sẽ làm giảm sự phức tạp trong việc triển khai dịch vụ. Tuy nhiên, việc phân bố vị trí VNF trên các máy chủ khác nhau sẽ làm tăng độ trễ tổng thể và tỷ lệ sử dụng máy chủ. Trong bài báo này, tác giả đã nghiên cứu, xây dựng và đánh giá bốn thuật toán phân bố vị trí trong chuỗi SFC trong mô hình điện toán biên. Các thuật toán này nhằm mục đích tối ưu hóa thời gian phản hồi dịch vụ hoặc tối ưu hóa việc sử dụng tài nguyên. Để đánh giá và so sánh hiệu suất của các thuật toán phân bố vị trí này, tác giả sử dụng

trình mô phỏng EdgeNetworkCloudSim mở rộng. Qua đó, các trung tâm dữ liệu được đặt trong điện toán biên và các VNF của SFC được đặt trong các máy chủ của trung tâm dữ liệu theo một thuật toán sắp xếp vị trí cụ thể. Trong khi các thuật toán CEN và ORC phân bố vị trí các VNF dựa trên giải thuật sắp xếp đơn giản để tìm các vị trí thích hợp, thì 2 thuật toán STO và RO là các giải pháp được tối ưu hóa bằng cách giải một mô hình ILP.

Về thời gian phản hồi dịch vụ, kết quả cho thấy thuật toán STO hoạt động tốt hơn các thuật toán khác trong tất cả các loại dịch vụ trong chuỗi. Điều này chứng tỏ rằng, việc sử dụng mô hình ILP có thể tính toán một giải pháp tối ưu. Đặc biệt, xác suất đặt tất cả các máy ảo của một chuỗi trong một máy chủ cao hơn so với các thuật toán CEN và ORC, dẫn đến giảm thời gian phản hồi dịch vụ. Tuy nhiên, thời gian xử lý của trình tối ưu hóa CPLEX là một nhược điểm đáng kể do thời gian tính toán có thể tăng lên khi cấu trúc liên kết lớn và phức tạp hơn.

Mục tiêu thứ hai của mô hình ILP là tối ưu việc sử dụng tài nguyên máy chủ. Vị trí của SFC được tối ưu để sử dụng số lượng máy chủ ít nhất. Kết quả của mô phỏng cho thấy rằng, trong trường hợp 10 SFC đồng thời được kích hoạt, thuật toán RO chỉ yêu cầu một nửa tài nguyên máy chủ. Ngược lại, các thuật toán khác sử dụng tất cả các trung tâm dữ liệu để xử lý một số lượng SFC đồng thời tương ứng. Nghiên cứu này có thể được sử dụng làm tài liệu tham khảo cho các nhà nghiên cứu cùng lĩnh vực hoặc các nhà phát triển dịch vụ điện toán biên, dịch vụ chuỗi chức năng mạng và đặc biệt là công nghệ NFV.

Lời cảm ơn

Nghiên cứu này được hỗ trợ bởi đề tài nghiên cứu khoa học cấp cơ sở Trường Đại học Công nghệ thông tin và Truyền thông – Đại học Thái Nguyên (Mã số: T2021-07-19).

TÀI LIỆU THAM KHẢO/ REFERENCES

[1] J. Halpern and C. Pignataro, *Service function chaining (SFC) architecture*, RFC, Tech. Rep. 7665, Oct. 2015.

[2] W. John, K. Pentikousis, G. Agapiou, E. Jacob, M. Kind, A. Manzalini, F. Risso, D. Staessens, R. Steinert, and C. Meirosu, "Research directions in network service chaining," in *SDN for Future Networks and Services (SDN4FNS)*, Trento, Italy: IEEE, Nov 2013.

[3] M. Chios, D. Clarke, P. Willis, A. Reid, J. Feger, M. Bugenhagen, W. Khan, M. Fargano, C. Cui, and H. Deng, "Network functions virtualisation: An introduction, benefits, enablers, challenges and call for action," In *SDN and OpenFlow World Congress*, Darmstadt-Germany, October 22-24, 2012.

[4] R. Mijumbi, J. Serrat, J. Gorricho, N. Bouten, F. D. 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.

[5] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90 - 97, 2015.

[6] E. Masanet, A. Shehabi, J. Liang, L. Ramakrishnan, X. Ma, V. Hendrix, B. Walker, and P. Mantha, *The energy efficiency potential of cloud-based software: A us case study*, Technical Report, Ernest Orlando Lawrence Berkeley National Laboratory (LBNL), Berkeley, CA (United States), 2013.

[7] I. Chih-Lin, J. Huang, R. Duan, C. Cui, J. X. Jiang, and L. Li, "Recent progress on c-ran centralization and cloudification," *IEEE Access*, vol. 2, pp. 1030-1039, 2014.

[8] D. Bhamare, R. Jain, M. Samaka, and A. Erbad, "A survey on service function chaining," *Journal of Network and Computer Applications*, vol. 75, pp. 138-155, 2016.

[9] N. Huin, A. Tomassilli, F. Giroire, and B. Jaumard, "Energy-efficient service function chain provisioning," *Journal of Optical Communications and Networking*, vol. 10, no. 3, pp. 114-124, 2018.

[10] M. Seufert, B. K. Kwam, F. Wamser, and P. Tran-Gia, "EdgeNetworkCloudsim: Placement of service chains in edge clouds using networkcloudsim," In *IEEE Conference on Network Softwarization (NetSoft 2017)*, IEEE, Bologna, Italy, Jul. 2017, pp. 1-6.

[11] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of things journal*, vol. 3, no. 5, pp. 637-646, 2016.

[12] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A survey on quality of experience of HTTP adaptive streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469-492, 2014.