# Bus-RSN: Giải pháp tô-pô mạng liên kết dạng lai cho các trung tâm dữ liệu cỡ vừa, tiết kiệm chi phí và đáp ứng không gian mở

Kiều Thành Chung[1] [3], Vũ Quang Sơn[2], Phạm Đăng Hải[3], Nguyễn Khanh Văn[3]
[1]Trường Cao đẳng nghề Công nghệ cao Hà Nội
[2]Học viện Công nghệ Bưu chính Viễn thông
[3]Trường Đại học Bách khoa Hà Nội

Tác giả liên hệ: Kiều Thành Chung, kieuthanhchung@gmail.com
Ngày nhận bài: 08/05/2020, ngày sửa chữa: 21/06/2020
Định danh DOI: 10.32913/mic-ict-research-vn.vyyyy.nx.xyz

**Tóm tắt:** Xây dựng và phát triển các trung tâm dữ liệu (DC – Data Center) kích thước vừa và nhỏ đang được các doanh nghiệp Việt nam quan tâm. Một thách thức lớn ở đây là thiết kế tô-pô liên kết có chi phí triển khai thấp mà đảm bảo tính linh hoạt cao như dễ mở rộng và đáp ứng điều kiện hạn chế về không gian như: kết hợp nhiều phòng máy chủ có diện tích sàn bị giới hạn, không liền kề (đôi khi ở các tầng khác nhau trong tòa nhà). Các kiến trúc tô pô mạng liên kết được ứng dụng phổ biến trên thế giới là không thực sự phù hợp cho những điều kiện thực tế đặc thù này. Trong bài báo này, chúng tôi đề xuất một giải pháp tô-pô mạng liên kết dạng lai, ký hiệu Bus-RSN, có thiết kế đặc thù phù hợp cho việc lắp đặt các DC cỡ vừa và nhỏ nhằm thỏa mãn các mục tiêu trên. Tô-pô đề xuất được tạo ra bằng cách lai kết hợp tô-pô dạng Bus và tô-pô RSN (Random Shortcut Networks)[8], trong đó thành phần bus đóng vai trò là đường trục kết nối các vùng máy chủ liên tục (tương ứng một phòng hay sàn) mà mỗi vùng được liên kết bằng một cấu trúc RSN, phù hợp với giới hạn không gian riêng biệt của mỗi phòng máy chủ. Qua so sánh với các giải pháp tô-pô hiện có đáng quan tâm thông qua thực nghiệm với công cụ phần mềm[1], chúng tôi thấy giải pháp đề xuất có thể đem lại một lựa chọn có tính thực tiễn cao: giảm được chi phí thiết bị đáng kể trong khi yếu tố hiệu năng quan trọng nhất (đỗ trễ truyền tin) bị ảnh hưởng giảm tương đối nhỏ. Chẳng hạn với một cấu trúc không gian gồm 2 sàn lắp đặt riêng biệt, Bus-RSN có thể tiết kiệm chi phí thiết bị mạng đến 26% so với tô pô hiện đại hàng đầu là JellyFish[2] mà chỉ thua kém 12% về độ trễ truyền tin.

**Từ khóa:** *tô-pô mạng, mạng liên kết dạng lai, thuật toán định tuyến, mạng ngẫu nhiên.*

---

**Title:** **Bus-RSN: An Interconnect Topology for Medium-Size Data Centers, Saving in Cable and Fitting to Incremental Expansion to Non-continuous Space**

**Abstract:** The demand of building Industrial Data Centers in small and medium sizes is growing significantly in Vietnam. One faces a challenging task in designing interconnection topologies that are flexible and incremental, of initial cheap cost, and suitable for being deployed in separate rooms. Existing popular interconnect topologies are not good enough in these mentioned special conditions. In this paper, we propose a hybrid interconnect, dubbed Bus-RSN, that is specially designed for these mentioned purposes. Simplistically speaking, this interconnect is created by combining a Bus structure and a random shortcut network (RSN[8]) wherein the bus is used as a backbone to connect the separate server rooms while each server room is locally connected by a RSN structure. Using our simulation-based software tool[1] we compare our proposed interconnect with popular suitable existing ones, including JellyFish[2], and R3[3], and obtain encouraging results: ours loses a bit in latency (12%) but saves a lot in cable cost (26%) comparing to JellyFish, the popular for fast communication.

**Keywords:** *network topology, hybrid-network, routing algorithm, random network.*

---

## I. GIỚI THIỆU

Nhu cầu xây dựng và phát triển riêng các hệ thống DC (Data Center, viết tắt: DC) cỡ vừa và nhỏ tại một số doanh nghiệp Việt nam đang trở nên phổ biến, ví dụ như tại các ngân hàng, thư viện dữ liệu số, trung tâm báo chí. Ngoài phương án thuê địa điểm đặt máy chủ tại các DC lớn của các doanh nghiệp công nghệ hàng đầu như Viettel, VNPT,

FPT ..., các doanh nghiệp đủ lớn cũng đang mở rộng thêm phương án đầu tư xây dựng mới các DC cỡ nhỏ (khoảng vài chục đến vài trăm máy chủ), nhằm chủ động trong việc đầu tư, khai thác hệ thống. Ban đầu các DC này có thể chỉ phục vụ hoạt động riêng của doanh nghiệp nhưng sau đó nó có thể mở rộng lên khi doanh nghiệp mở rộng kinh doanh và có thể kết hợp cho bên ngoài thuê bao khai thác. Vì thế, nét đặc thù của xu hướng này là việc thiết lập không gian cho DC cần mang tính linh hoạt, mềm dẻo để có thể mở rộng dần, thích hợp với việc các DC có thể tăng trưởng kích thước liên tục (từ cỡ nhỏ, vài chục đến hàng trăm máy, lên cỡ vừa, hàng nghìn máy).

Khác với việc xây dựng các trung tâm dữ liệu lớn thường bao gồm việc xây dựng những tòa nhà hay khu vực không gian biệt lập với các điều kiện lý tưởng, việc phương án xây dựng các DC vừa và nhỏ thường ưu tiên tính kinh tế bằng cách khai thác các khoảng không gian còn trống hay được chuyển đổi chức năng trong các tòa nhà doanh nghiệp đã có sẵn. Những không gian mang tính huy động gom lại này thường có những hạn chế như không đủ rộng lớn và liên tục, có thể là sự kết hợp của nhiều phòng hay mặt sàn sử dụng tách rời nhau (thậm chí có thể nằm trên nhiều tầng tách biệt của một tòa nhà). Ở một số tình huống khác, một DC cỡ nhỏ ban đầu có thể được phát triển dần dần về qui mô và có thể "phình ra" vượt quá sự cho phép của không gian thiết kế ban đầu. Thường là do khả năng huy động nguồn vốn ban đầu, DC có thể có kích thước nhỏ và bố trí lắp đặt trong một không gian vừa phải, nhưng nếu sau đó hoạt động hiệu quả, doanh nghiệp có thể huy động thêm vốn và muốn tìm cách mở rộng kích thước DC một cách linh hoạt. Do đó không gian lắp đặt ban đầu có thể không còn chỗ và phải tiến hành xem xét việc mở rộng sang khu vực mới (mặt sàn mới) và cần tìm phương án giải quyết việc kết nối giữa 2 khu vực sao cho hiệu quả nhất.

Các DC có thể được mô hình hóa dưới dạng đồ thị $G(V, E)$ với $G$ là tập các đỉnh biểu diễn cho các nút mạng và $E$ là tập các cạnh biểu diễn cho các liên kết giữa chúng, được tạo ra bởi các luật liên kết được định trước. Trong bài báo này, với $N = |V|$ đỉnh cho trước, chúng tôi nghiên cứu tìm cách sắp xếp $N$ đỉnh này vào không gian không liên tục và luật liên kết giữa chúng để xây dựng một mô hình mạng liên kết phù hợp nhất, tức là đạt được hiệu năng và tính kinh tế thích hợp, cho dạng DC có tính *dễ mở rộng tăng dần* đồng thời có thể *bố trí lắp đặt trong một không gian thiếu liên tục,* tức là có thể là sự kết hợp của nhiều mặt sàn (không gian lắp đặt liên tục) tách rời và có khoảng cách nhất định giữa chúng. Chúng tôi trước hết khảo sát các mô hình mạng liên kết đã được đề xuất thuộc thiết kế tô-pô mạng liên kết hiệu năng cao, từ đó tìm cách phát triển một dạng tô-pô kiểu mới phù hợp với thực trạng doanh nghiệp nhỏ và vừa.

Theo kết quả của một quá trình khảo sát tương đối công phu chúng tôi nhận thấy có 3 cách tiếp cận chính trong thiết kế tô-pô mạng liên kết của một DC: theo dạng tô-pô chuẩn tắc (regular topology), hoặc tô pô mạng ngẫu nhiên (random network topology), hoặc tô-pô lai (hybrid topology) là sự kết hợp những đặc thù từ cả hai dạng trước đó.

Các tô-pô dạng chuẩn tắc phải tuân theo các qui ước nghiêm ngặt như tính đối xứng giữa các đỉnh, hay các tính chất cấu trúc đặc biệt riêng nên việc mở rộng qui mô là gần như không thể; nói cách khác, các mạng liên kết đã được thiết kế theo tô-pô dạng chuẩn tắc là sẽ gần như đóng với mọi sự thay đổi. Với các tô-pô chuẩn tắc truyền thống, nếu như bắt buộc phải thay đổi để tăng qui mô, gần như người ta sẽ phải xây dựng lại cả hệ thống mạng. Các tô-pô thế hệ mới, được đề xuất gần đây như Fat-Tree[4][5], có một số cải thiện về khả năng mở rộng tuy nhiên vẫn còn kém linh hoạt do vẫn cần đảm bảo tính đối xứng và bậc của nút mạng là khá lớn. Muốn mở rộng qui mô, doanh nghiệp sẽ cần phải thay thế toàn bộ các thiết bị chuyển mạch cũ bằng loại mới với số cổng (port) cao hơn, thường là đắt tiền hơn nhiều, dẫn đến chi phí mở rộng rất lớn. Ngoài ra do Fat-Tree có bậc đỉnh cao, yêu cầu xây dựng DC trong không gian là kết hợp của một số mặt sàn rời nhau sẽ gây rất tốn kém về cáp mạng.

Tiếp cận tô-pô dạng mạng ngẫu nhiên ra đời chính nhằm mục đích đảm bảo khả năng mở rộng qui mô tăng dần (incremental growth), trong đó tô-pô JellyFish[2] là một điển hình và được giới học thuật quan tâm nhiều. Tuy nhiên các tô-pô dạng này vẫn ít nhiều có tính đối xứng và bậc đỉnh cao (JellyFish là một đồ thị ngẫu nhiên có bậc đỉnh hằng số $r$ cho trước) nên cũng đòi hỏi huy động cáp mạng rất lớn nếu không gian lắp đặt không là một mặt sàn liên tục duy nhất.

Một nhóm các giải pháp tô-pô lai cũng đã được đề xuất, với tô-pô dựa trên cấu trúc siêu khối (hyper-cube) và thực hiện đệ quy để mở rộng, ví dụ R3[3], BCN[6], DCell[7]. Do dựa trên cấu trúc siêu khối, nên một hạn chế đáng kể của các dạng tô-pô này chính là sự mở rộng, ví dụ, nếu bổ sung thêm một vài nút mạng trong một tô-pô thành phần, thì các tô-pô thành phần khác cũng phải được bổ sung thêm số lượng nút tương ứng để đảm bảo tính chuẩn và tính đệ quy. Điều này có thể gây ra sự dư thừa nút mạng, gia tăng không cần thiết về chi phí thiết bị và triển khai cài đặt mở rộng. Hơn nữa, các tô-pô thành phần phải được triển khai trên các phòng máy chủ có không gian đồng đều nhau.

Theo khi khảo sát kỹ các tô-pô mạng liên kết đã biết theo cả 3 nhóm tiếp cận nói trên, đặc biệt chú ý khảo sát các đại biểu của mỗi nhóm là Fat-Tree[4], JellyFish[2] và R3[3], chúng tôi rút ra kết luận là các thiết kế tô-pô mạng được đề xuất trước đây hướng tới việc xây dựng các DC

cỡ lớn và mặc dù không đề cập đến vấn đề không gian lắp đặt của các phòng máy chủ, có một giả thiết ngầm định là các cơ sở vật chất (không gian lắp đặt) được xây dựng sao cho phù hợp với thiết kế tô-pô mạng tương ứng. Do đó hầu hết các tô-pô mạng phổ biến, bao gồm cả các tô-pô lai loại hiện đại hướng tới tính mở rộng, sẽ gặp khó khăn lớn trong việc triển khai trên một địa bàn có tính chất thiếu liên tục và không đồng nhất như đề cập trên.

Trong bài báo này, chúng tôi đề xuất Bus-RSN, một mô hình tô-pô lai (hybrid topology) như là một giải pháp chuyên dụng, đề xuất riêng cho bài toán xây dựng DC với những yếu tố đặc thù nói trên. Kiến trúc này là sự kết hợp hai dạng tô-pô: 1) Bus – cổ điển và tối giản, và 2) RSN (random shortcut networks)[8]; sự ra đời của nó là xuất phát từ những quan sát thực tế phát triển DC trong nước của chúng tôi. Với một trung tâm DC cỡ vừa hoặc nhỏ, có thể lắp đặt trong không gian hợp thành từ nhiều phòng (sàn) rời rạc; các ứng dụng khai thác cũng thường yêu cầu một số lượng máy chủ phục vụ không lớn, phù hợp triển khai trong một phòng máy chủ nào đó. Vì vậy giao thông[1] qua lại của một ứng dụng khai thác thường xuyên sẽ là giữa các máy chủ cùng phòng, là không lớn. Do đó, chúng tôi đề xuất sử dụng cấu trúc Bus như là một "đường xương sống" để liên kết các phòng máy chủ, đủ cho các nhiệm vụ quản lý, kiểm soát mạng đồng thời thiết kiệm chi phí cáp mạng và thiết bị liên quan. Đối với mỗi phòng máy chủ, chúng tôi áp dụng mô hình liên kết ngẫu nhiên RSN (phối hợp giữa cấu trúc lưới và các liên kết ngẫu nhiên) để làm giảm đường kính mạng và vẫn cho phép tiết kiệm cáp mạng. Bên cạnh đó, chúng tôi cũng thêm vào thiết kế một dạng liên kết ngẫu nhiên giữa các phòng máy, được bổ sung vào các nút mạng đặc biệt (gọi là bus-node) giữ nhiệm vụ như các nút trung tâm của các khối đơn vị thành phần của các mặt sàn phòng máy liên tục (chúng tôi dùng các thuật ngữ block và zone để gọi các cấu thành mạng nói trên). Những liên kết ngẫu nhiên loại này cho phép rút ngắn đường kính mạng (graph diameter) và đảm bảo rằng ngay cả khi các ứng dụng triển khai trên nhiều hơn một mặt sàn (zone) thì vẫn được phục vụ hiệu quả.

Thông qua đánh giá bằng thực nghiệm, chúng tôi thấy BUS-RSN đạt được những hiệu quả tốt hơn đáng khích lệ khi so sánh với những tô-pô cụ thể đại biểu quan trọng của các nhóm nói trên (Fat-Tree[4][5], JellyFish[2], R3[3]) khi cùng thiết lập với những điều kiện đặc thù cụ thể đã đề cập. Chẳng hạn như đối với một yêu cầu thiết lập mạng 128 nút chuyển mạch (switch), lắp đặt trên 2 mặt sàn (zone) biệt lập (cách nhau 15 mét) mỗi sàn có 4 khối đơn vị (block), Bảng I cho thấy, thiết lập theo JellyFish (JF-KSPR-2) sẽ đạt được hiệu năng cao nhất về độ trễ ước lượng (estimated latency) nhưng hơn thiết lập theo BUS-RSN (BR-HRA-2-4) đứng thứ hai chỉ khoảng 12%; trong khi đó BR-HRA-2-4 là tiết kiệm nhất về cáp mạng, chỉ là 18% khi so với JF-KSPR-2, và do đó tổng giá thành dự kiến chỉ là 74% của JF-KSPR-2.

Bảng I
KẾT QUẢ THỰC NGHIỆM BUS-RSN ĐƯỢC THIẾT LẬP 2 ZONE, MỖI ZONE 4 BLOCKS TRÊN MẠNG 128 NÚT.

| 128 nút mạng | ARPL | Latency | Cable | Total cost |
|---|---|---|---|---|
| BR-HRA-2-4 | 2,95 | 326,45 | 1528,20 | 731.804,30 |
| JF-KSPR-2 | 2,90 | 289,89 | 8273,20 | 986.491,50 |
| R3-SPR-8 | 4,78 | 534,62 | 3605.2 | 826.216,60 |
| RSN-SPR-2 | 3,12 | 344,28 | 3813,00 | 784.523,55 |

Rõ ràng những kết quả thực nghiệm như thế này cho thấy giải pháp đề xuất của chúng tôi là rất đáng được xem xét và có thể là một lựa chọn hữu ích cho các doanh nghiệp trong nước đang xem xét thiết kế DC với các điều kiện đặc thù đề cập.

Phần còn lại của bài báo được tổ chức như sau: phần II trình bày các nghiên cứu liên quan, phần III trình bày giải pháp chính, phần IV trình bày các kết quả thực nghiệm cùng với một số thiết lập mạng cụ thể theo điều kiện đặc thù quan tâm và phần kết luận được trình bày trong phần V.

## II. CÁC NGHIÊN CỨU LIÊN QUAN

### 1. Tô-pô mạng liên kết

Cấu trúc tô-pô thể hiện sự sắp đặt các nút mạng[2] và các liên kết giữa chúng. Trong phạm vi bài báo này, chúng tôi tập trung vào tô-pô áp dụng cho DC mà trong đó các máy chủ được kết nối với nhau thông qua cấu trúc mạng chuyên dụng, sao cho đảm bảo các mục tiêu thiết kế cụ thể như là chi phí thiết bị thấp, khả năng đáp ứng cao, và mở rộng nhanh chóng.

Các tô-pô truyền thống trước đây được đề xuất áp dụng cho DC với kích thước mạng nhỏ, ví dụ Grid, Torus, HyberCube . Các dạng tô-pô này được gọi là các tô-pô chuẩn tắc với các quy tắc chặt chẽ về mặt kết nối và đảm bảo bậc đỉnh đều trên mọi nút. Tuy nhiên, khi kích thước mạng (số nút mạng) tăng lên nhanh chóng do sự phát triển mạnh của DC, các tô-pô truyền thống trở nên kém hiệu quả khi không đáp ứng được khả năng mở rộng và tính mềm dẻo. Cụ thể là việc bổ sung thêm các nút mạng, tăng/giảm kích thước mạng có thể được tiến hành thường xuyên, nhiều lần với quy mô đa dạng, nhưng vẫn phải đảm bảo hiệu năng mạng. Với kích thước mạng ngày càng tăng, tính mềm dẻo đòi hỏi cao thì các vấn đề như tiết kiệm chi phí thiết bị và cáp kết nối, tiết kiệm năng lượng khi tải thấp, trở thành những chủ đề đáng quan tâm. Có thể thấy

[1] Sự di chuyển của các gói tin giữa các nút mạng

[2] ví dụ như các máy tính toán hoặc các thiết bị chuyển mạch – switch

vấn đề này được đề cập trong hàng loạt nghiên cứu gần đây về việc thiết kế mạng liên kết trong DC như HELIOS[9], BCube[10], Dcell[7], Scafida[11], MDCube[12], VL2[13]. Do vậy, các nhà nghiên cứu tích cực đề xuất những dạng kiến trúc tô-pô mới phù hợp hơn cho các yêu cầu hiện đại như Smallworld DataCenter[14], JellyFish[2], Space Shuffle[15], Fat-Tree[4][5], SkyWalk[16], Dragonfly[17].

Một vài đề xuất gần đây đối với các mạng dung lượng cao khai thác cấu trúc cụ thể cho tô-pô và thuật toán định tuyến tương ứng. Chúng bao gồm folded-Clos (hay còn gọi là Fat-Tree)[4][5][13][18], một vài thiết kế có sử dụng các máy chủ trong việc chuyển tiếp gói tin[7][10][12] và các thiết kế sử dụng công nghệ mạng kết nối quang[9][19]. Đối với một số tô-pô cụ thể, việc bổ sung thêm số lượng máy chủ trong khi phải đảm bảo các thuộc tính chặt chẽ của cấu trúc sẽ phải thay thế một số lượng lớn các thành phần mạng cũng như thực hiện lại việc thi công lại tuyến cáp. MDCube[12] cho phép mở rộng mạng thêm vài nghìn máy chủ, trong khi DCell[7] và BCube[10] cho phép mở rộng đến các kích thước mạng dự kiến, nhưng phải thiết kế bổ sung các cổng kết nối dự phòng, điều này dẫn đến sự dư thừa và chi phí lắp đặt tăng lên.

Trong số các đề xuất, Fat-Tree[4][5] được xem là một trong những mô hình hiệu quả đối với DC có kích thước mạng lớn (ví dụ, có thể hỗ trợ kết nối lên đến 65536 máy chủ với thiết bị switch có 64 cổng) với việc tổ chức cấu trúc mạng thành 3 tầng riêng biệt, bao gồm tầng lõi (core layer), tầng trung gian (aggregation layer) và tầng kết nối trực tiếp các host (edge layer). Mặc dù bậc đỉnh của các nút mạng trong mô hình Fat-Tree không đều nhau, tuy nhiên, xét trong cùng một tầng, các nút có cùng bậc đỉnh và chúng được kết nối với nhau theo quy luật một cách chặt chẽ. Do vậy, Fat-Tree được xem là một regular topology thế hệ sau so với các regular topology truyền thống (ví dụ như Torus, Grid, HyperCube). Thiết kế của Fat-Tree đảm bảo tham số oversubcription đạt 1:1, có nghĩa là đảm bảo tất cả các host có khả năng kết nối tới bất kỳ host khác với băng thông đường truyền kết nối đạt tối đa. Mặc dù Fat-Tree có nhiều ưu điểm như đường kính mạng thấp, độ trễ truyền tin thấp, *oversubcription*[3] đạt 1:1, tuy nhiên, nó cũng tồn tại một số hạn chế đáng kể. Do cấu trúc chặt chẽ, nên khả năng mở rộng của Fat-Tree phụ thuộc vào số cổng thiết bị switch được sử dụng trong việc kết nối mạng. Ví dụ, kích thước 3456, 8192, 27648 và 65536 tương ứng với số cổng switch là 24, 32, 48 và 64. Trong trường hợp giả định có thể sử dụng switch 50 cổng, có thể bổ sung thêm 3602 máy chủ nhưng phải thay thế toàn bộ các thiết bị switch ban đầu (48 cổng). Đây chính là hạn chế về khả năng mở rộng và làm gia tăng chi phí thiết bị của Fat-Tree.

Một cách tiếp cận mới và hấp dẫn gần đây là sử dụng các mô hình mạng ngẫu nhiên trong việc thiết kế các tô mô mạng liên kết mà có thể đạt hiệu quả để cung cấp chất lượng hiệu năng quan trọng như là tính linh hoạt (flexibility), tính mở rộng (scalability) và tính thích nghi (adaptivity) mà chúng là các đòi hỏi quan trọng đối với các tô pô DC hiện đại như đã được đề cập. Thông thường, tô pô ngẫu nhiên được xây dựng bằng việc bổ sung thêm các liên kết ngẫu nhiên đối với một đồ thị cơ bản đơn giản và chính tắc như dạng lưới 2-D; các liên kết ngẫu nhiên được tạo ra bởi một phân bố xác suất nào đó, nhưng thường là phân bố đều. Ví dụ như đồ thị liên kết ngẫu nhiên RSN, được tạo ra bởi việc bổ sung các liên kết ngẫu nhiên giữa các nút trên đồ thị cơ sở dạng lưới (grid). Mô hình xây dựng mạng này được chỉ ra tại[8], có thể đạt được đường kính đồ thị (và độ trễ truyền thông) giảm đáng kể so với các tô pô mạng truyền thống (với cùng kích thước và cùng bậc đỉnh). Hơn nữa, các tính chất quan trọng của việc mở rộng một cách tự nhiên với mô hình mạng ngẫu nhiên, do đó, nó hấp dẫn đối với mạng DC Jellyfish[2], Scafida[11]. Đề xuất JellyFish sử dụng mô hình mạng ngẫu nhiên để hỗ trợ mở rộng mạng và giá trị oversubcription đạt tỉ lệ 1:1. Tuy nhiên, JellyFish phải sử dụng thuật toán định tuyến -SPR với kích thước bảng định tuyến lớn và sử dụng các switch có bậc đỉnh cao (ví dụ, 48 cổng).

Mặc dù cả Fat-Tree[4][5] và JellyFish[2] đều đạt được giá trị oversubcription 1:1, tuy nhiên, bậc đỉnh của hai đại diện này khá cao, dẫn đến chi phí thiết bị mạng cao. Fat-Tree còn tồn tại bất lợi trong việc mở rộng mạng trong khi vẫn phải đảm bảo các tính chất chặt chẽ của cấu trúc mặc dù có lợi thế trong việc thực hiện định tuyến dễ dàng. Ngược lại, JellyFish có nhược điểm về việc xây dựng luật định tuyến do tính chất ngẫu nhiên của các liên kết, tuy nhiên, JellyFish lại có khả năng hỗ trợ mở rộng mạng (đảm bảo tính co giãn quy mô). Vậy có tồn tại một mô hình tô-pô có thể kết hợp và khai thác được những ưu điểm của những tô-pô trên (regular và ir-regular), đồng thời hạn chế những nhược điểm của chúng?

Một dạng mô hình với ý tưởng dựa trên tô-pô lai đang được quan tâm hiện nay là đồ thị hỗn hợp (compound graph). Compound graph cấp 1 $G(G1)$ là mạng liên kết được tạo ra bởi 2 regular graph $G$ và $G1$, bằng cách thay thế các nút của $G$ bằng $G1$ và các liên kết của $G$ được thay thế bởi các liên kết mới giữa 2 bản sao của $G1$ tương ứng với 2 nút của $G$ mà nó thay thế. Compound graph cấp cao hơn có thể được tạo ra từ các compound cấp thấp theo phương pháp đệ quy. Ngoài việc giữ được tính chất ban đầu

[3]Các thiết kế DC đề xuất oversubcription như là một tham số để làm giảm tổng chi phí của thiết kế đó. Oversubcription là tỉ lệ giữa tổng băng thông, trong trường hợp tồi nhất, có thể đạt được giữa các hosts đối với tổng băng thông cực đại của một cấu trúc tô-pô cụ thể. Ví dụ, giá trị oversubcription 5:1 có nghĩa là băng thông của host chỉ đạt 20%. Thông thường, các DC được thiết kế oversubcription từ 2.5:1 (tương đương 400Mbps) đến 8:1 (tương đương 125Mbps) đối với băng thông đường truyền là cho mạng sử dụng switch thông thường Ethernet

của $G$, compound graph được bổ sung các thuộc tính tốt của $G1$. Vì vậy có nhiều DC được đề xuất theo hướng sử dụng mô hình này, ví dụ BCN[6], DCell[7], R3[3]... Trong đó, R3 là tô-pô đáng chú ý khi có cấu trúc khá phù hợp với bài toán mà chúng tôi đưa ra.

R3 là tô-pô được tạo ra dựa trên mô hình compound graph, bằng cách kết hợp 2 tô-pô: generalized hybercube (GHC) và random r-regular graph (RRG, trong R3 được gọi là cluster). Với GHC, R3 khá linh hoạt trong việc xây dựng quy mô DC với số nút mạng cho trước. Ngoài ra, RRG là tô-pô có đặc tính đường kính mạng nhỏ, linh hoạt trong việc mở rộng... Trong khi sở hữu các tính chất trên, R3 vẫn tồn tại một số nhược điểm. Do được tạo bởi GHC nên R3 cần có sự đánh đổi giữa đường kính mạng và bậc của nút. Hơn nữa, do cấu trúc random của RRG và phải liên kết nhiều RRG theo cấu trúc của GHC, nên việc truy vết các liên kết của tô-pô này khá khó khăn. Nếu áp dụng R3[3] để giải quyết bài toán đã nêu trong phần I thì có thể dẫn đến chi phí dùng cho switch và cáp mạng tăng cao cùng với khó khăn trong quá trình cài đặt và bảo trì.

## 2. Thuật toán định tuyến

Một vấn đề quan trọng trong việc đề xuất thiết kế tô-pô là xây dựng thuật toán định tuyến tương ứng để khai thác tối đa đặc tính tô-pô nhằm đảm bảo các yếu tố hiệu năng mạng. Quá trình định tuyến xác định đường truyền tin giữa các cặp nguồn – đích trong mạng. Các thuật toán định tuyến SPR sử dụng thuật toán Dijkstra hoặc thuật toán tham lam để xác định đường đi ngắn nhất giữa các cặp nguồn đích. Do tính chặt chẽ của cấu trúc, các tô-pô dạng chuẩn tắc thường dễ dàng áp dụng các thuật toán SPR, ví dụ, 2-D Torus, 3-D Torus, Fat-Tree[4]. Tuy nhiên, đối với các tô-pô ngẫu nhiên, ví dụ RSN, JellyFish, các thuật toán SPR tỏ ra kém hiệu quả với đường kính lớn. Để khắc phục nhược điểm đó, các thuật toán rút gọn (compact routing) được đề xuất áp dụng cho các đồ thị ngẫu nhiên, ví dụ, thuật toán TZ của tác giả Thorup & Zwick[20], CORRA[21] với việc khai thác thông tin định tuyến được lưu trữ tại các nút mạng (bảng định tuyến tại các nút mạng). Ví dụ, thuật toán TZ[20] khai thác thông tin của nút đại diện (landmark) của tập các nút trong vùng mà nó đại diện, bằng cách định tuyến từ nút nguồn tới nút đại diện gần nhất của nút đích, và sau đó chuyển tiếp gói tin tới nút đích; hoặc thuật toán CORRA[21], sử dụng các liên kết dài như là các cầu nối giữa các vùng nút để chuyển tiếp thông tin. Các thuật toán Compact không tuân theo định tuyến ngắn nhất, mà nó khai thác các liên kết giữa các nút xa nhau, từ đó làm giảm đường kính mạng và giảm trung bình độ trễ truyền tin.

Một vấn đề khá khó khăn trong cấu trúc tô-pô lai chính là làm thế nào định tuyến gói tin giữa các nút mạng trong nó. Với mỗi dạng tô-pô thành phần sẽ có kịch bản định tuyến tương ứng nhằm đảm bảo các tham số hiệu năng mạng. Tuy nhiên, việc đưa ra được luật định tuyến chung cho cấu trúc tô-pô lai trở nên khó khi mà các tô-pô regular thường áp dụng các thuật toán định tuyến theo chiều DOR hoặc sử dụng giao thức Duato[22], trong khi đó, các tô-pô ngẫu nhiên tỏ ra hiệu quả đối với kịch bản định tuyến rút gọn (CANDAR[23], CORRA[21], TZ[20]. Các thuật toán SPR tỏ ra kém hiệu quả đối với các tô-pô lai khi phải sử dụng quá nhiều tài nguyên (bộ nhớ tại mỗi switch). Trong trường hợp áp dụng kịch bản định tuyến rút gọn trên toàn bộ cấu trúc tô-pô lai, việc định tuyến trở nên khó khăn khi không xác định được luật định tuyến chung. Trong cả hai trường hợp, việc định tuyến không mang lại hiệu quả tốt. Một chiến lược định tuyến R3[3], được đề xuất nhằm áp dụng hiệu quả trên dạng tô-pô lai. Ý tưởng chính của cách tiếp cận này là kết hợp các thuật toán định tuyến khác nhau để tối ưu hiệu năng mạng và công việc định tuyến cũng được chia ra thành các trường hợp riêng biệt. Theo đó, R3 được định tuyến với 2 trường hợp: định tuyến nội vùng (intra-cluster, định tuyến giữa các nút mạng trong cùng một vùng) và định tuyến liên vùng (inter-cluster, định tuyến giữa các nút mạng thuộc hai vùng khác nhau). Trong trường hợp intra-cluster, 2 nút thuộc cùng 1 cluster, gói tin được định tuyến bằng thuật toán $k*$ algorithm[24] (là thuật toán sử dụng hàm mục tiêu heuristic hiệu quả nhất hiện nay, được áp dụng để tìm kiếm $k$ đường đi ngắn nhất). Đối với inter-cluster, 2 nút thuộc 2 cluster khác nhau, thuật toán định tuyến được kết hợp bởi các kỹ thuật khác nhau: Thuật toán Dijkstra và thuật toán định tuyến dựa trên tọa độ của các nút.

Chúng tôi xem xét kỹ hơn và áp dụng cách tiếp cận kết hợp nhiều thuật toán khác nhau khi định tuyến trong mô hình Bus-RSN. Trong đó, chúng tôi áp dụng luật định tuyến 2 giai đoạn tương tự như thuật toán định tuyến rút gọn[20] cho Bus-RSN. Giai đoạn 1, gói tin được định tuyến đến bus-node gần nút đích nhất và giai đoạn 2 là gói tin được đưa đến đích, chi tiết thuật toán được mô tả tại phần III.2

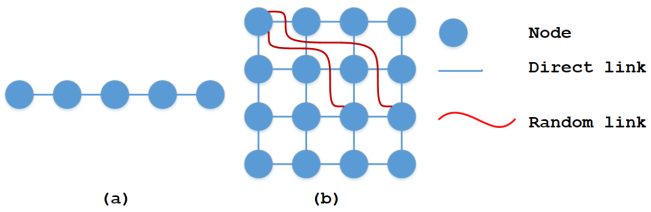
## III. GIẢI PHÁP TÔ-PÔ LAI ĐỀ XUẤT

Xét bài toán xây dựng DC với quy mô ban đầu nhỏ, được lắp đặt tại nhiều phòng/sàn riêng biệt, sao cho nó có khả năng linh hoạt trong việc mở rộng mà đảm bảo thông số hiệu năng cao hợp lý.

Ý tưởng cơ bản của chúng tôi là kết hợp lai giữa tô-pô dạng Bus[4] và RSN sao cho khai thác ưu điểm của chúng một cách phù hợp. Trước hết chúng tôi dự định kết nối các phòng/sàn bằng tô-pô dạng Bus do có thể đáp ứng được việc định tuyến và lắp đặt dễ dàng và hơn nữa lại linh hoạt trong việc mở rộng DC trong lương lai: dễ dàng triển khai

[4]Mạng liên kết với các nút được kết nối với nhau như 1 đường thẳng, liên kết với các nút liền kề với nó, ngoại trừ nút đầu và nút cuối cùng.

thêm các phòng máy chủ bằng việc thêm các nút mạng đại diện của các phòng đó vào 2 đầu của đường Bus, như Hình 1-a. Tất nhiên, tô-pô này không cho hiệu năng mạng cao, do đường kính mạng quá lớn, bên cạnh đó, mỗi phòng lại có kích thước khác nhau, có số lượng nút mạng khác nhau; do đó chúng tôi lựa chọn kiến trúc RSN (minh họa như Hình 1-b) cho tô-pô liên kết nội bộ mỗi phòng để vừa đảm bảo tính linh hoạt vừa có đường kính mạng đủ nhỏ.




Hình 1. a. Bus topology với 5 node
b. RSN 4x4 tạo bởi grid graph và random link

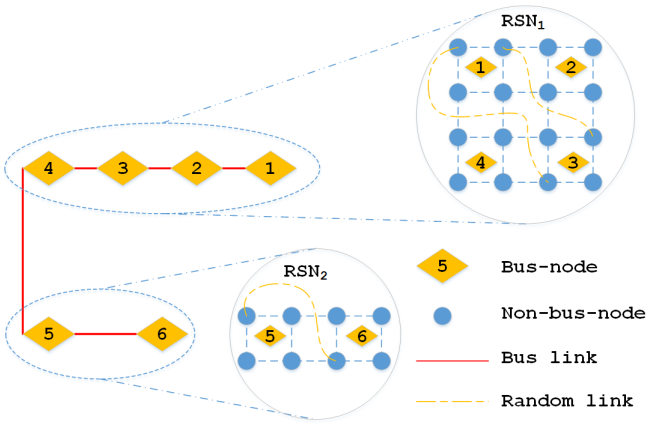
## 1. Kiến trúc

Phần coi là "bus" trong tô-pô Bus-RSN tạo thành một trục dọc "bus-way" liên kết nhiều sàn (không gian lắp đặt của một phòng máy chủ) phân biệt có kích thước khác nhau. Các sàn được ký hiệu là $RSN_i$ (ví dụ như $RSN_1$ và $RSN_2$, trong Hình-2), mà ở mỗi sàn đồ thị liên kết nội bộ tạo thành một tô-pô *RSN* (như trình bày ở mục II, là kết hợp của lưới với tập các random link[5] tức liên kết ngẫu nhiên), trên đó có một số nút được chọn để tạo ra trục "bus" được gọi là bus-node (trình bày dưới đây). Các $RSN_i$ được kết nối với nhau thông qua Bus-way, được tạo bởi danh sách các bus-node. Một gói tin di chuyển từ các nút thuộc $RSN_1$ đến các nút thuộc $RSN_2$ sẽ cần phải đi qua nhiều bus-node trên Bus-way. Để tạo ra kết nối chặt hơn giữa các sàn (từ đó hạ thấp đường kính đồ thị mạng), mô hình tô-pô này cho phép bổ sung các random link (liên kết ngẫu nhiên) giữa các bus-node trên Bus-way. Cấu trúc hoàn thiện của tô-pô được chúng tôi trình bày sau đây.

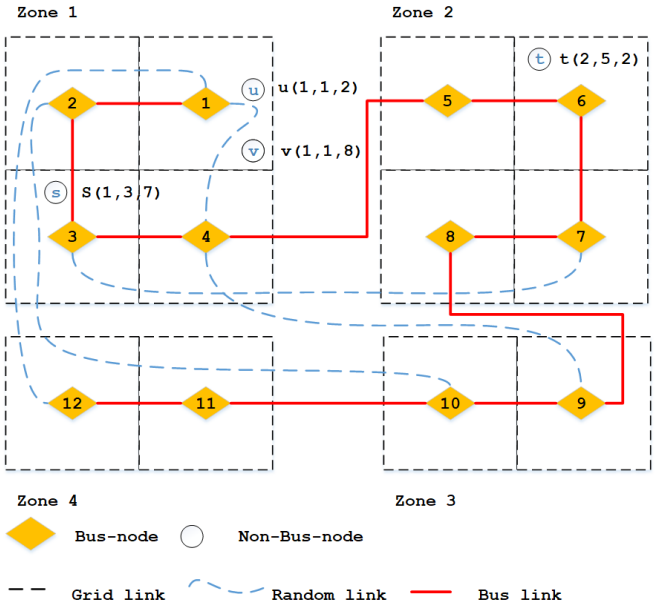
Một cách khái quát, Bus-RSN được biểu diễn như đồ thị $G = G_1 \cup G_2 \cup ... \cup G_m$, mỗi đồ thị $G_i = (V_i, E_i)$ là một tô-pô dạng RSN được gọi là zone thứ $i$, ký hiệu $z_i$, ứng với mỗi sàn – không gian liên tục biệt lập. Mỗi zone được chia thành $k$ khối có kích thước giống nhau được gọi là *block*. Trong mỗi block, một nút được chọn làm nút đại diện cho các nút khác, được gọi là *bus-node*, nằm ở vị trí trung tâm block. Các bus-node được liên kết lần lượt với nhau để tạo nên Bus-way. Sau khi Bus-way được tạo ra, các random link có thể được tạo bổ sung giữa 2 bus-node bất kỳ.

Hình 3 minh họa Bus-RSN gồm 4 zone, mỗi zone được chia thành các block có kích thước bằng nhau. Các nút được

[5]Random link (liên kết ngẫu nhiên) là các liên kết kết nối giữa 2 nút mạng ngẫu nhiên trong mạng liên kết




Hình 2. Mô hình tô-pô Bus-RSN




Hình 3. Mô hình chi tiết của Bus-RSN

chọn làm đại diện của các block được đánh số thứ tự từ 1 đến 12 và tạo thành Bus-way. Ngoài các bus-link, là các liên kết cơ bản giữa các bus-node, còn có các random link kết nối 2 nút bất kỳ trên bus-node. Quá trình tạo thành một tô-pô dạng Bus-RSN được mô tả dưới đây có kèm mã giả (pseudo-code) minh họa bởi Algo 1.

Algo. 1 là thủ tục được minh họa trong Hình 4 để khởi tạo mạng liên kết Bus-RSN với M zone. Kích thước của các zone và số block của zone đó được lưu trữ tại mảng A[] và B[] tương ứng. Ví dụ zone-i có kích thước là A[i] và có số block là B[i]. Các tham số đầu vào đều là các số nguyên dương và là bội số của 2.

Quá trình tạo tô-pô Bus-RSN có thể được chia thành 4 giai đoạn như sau:

- *Khởi tạo tô-pô RSN* ($GenerateRSN(A[i])$): các tô-pô

**Thuật toán 1:** Thuật toán xây dựng mạng liên kết theo mô hình Bus-RSN

**Đầu vào:** int M, số lượng zone của đồ thị;
int A[],lưu trữ kích thước của M zone;
int B[],lưu trữ số block của M zone;
**Đầu ra :** Tô-pô Bus-RSN

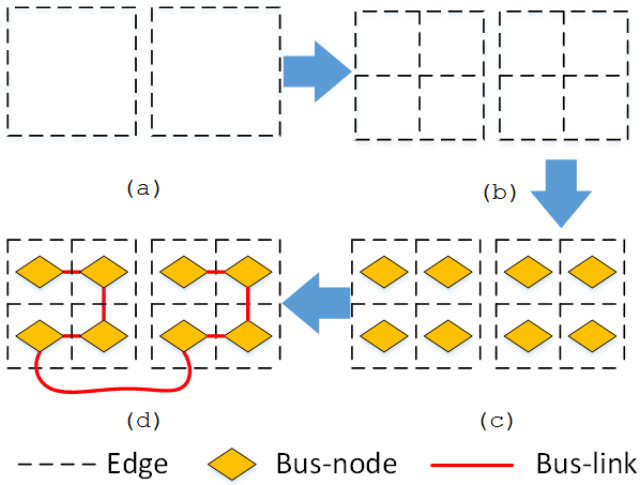
```
1  i = 0;
2  BusNodes ← ∅;
3  //khởi tạo danh sách bus-node của tô-pô
4  for (i = 0; i < M; i++){
5      Zone_i = GenerateRSN(A[i]);
6      Blocks = DivideBlock(Zone_i, B[i]);
7      BusNodeSelection(Blocks);
8      Update(BusNodes);
9  }
10 BuildBusWay(BusNodes);
```

RSN với kích thước cho trước được khởi tạo bởi thủ tục tại dòng 4 của Algo. 1. RSN được tạo ra bằng cách thêm các random link (cho các cặp nút được lựa chọn ngẫu nhiên theo phân bố đồng nhất) của tô-pô cơ sở dạng lưới. Giai đoạn này tương ứng với Hình 4-a, RSN chưa được chia block và các bus-node chưa được chọn.

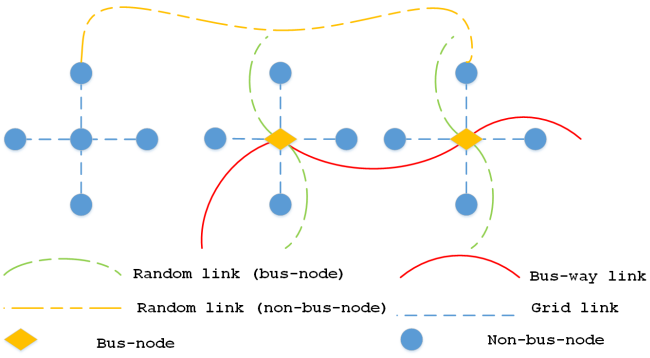
- *Phân chia tô-pô thành các khối*: Sau khi được khởi tạo tại giai đoạn 1, $Zone_i$ của mạng được chia thành B[i] block có kích thước lưới X*Y bằng nhau. Các kích thước được chia sao cho X và Y là bội số của 2 và $|X-Y|$ là nhỏ nhất. Ví dụ: Zone có kích thước 128 có 4 block có kích thước là 4*8.
- *Lựa chọn bus-node*: Ngay sau khi zone được chia thành các block với kích thước giống nhau, các bus-node sẽ được tiến hành chọn và được lưu trữ vào mảng BusNodes[]. Các bus-node là nút ở trung tâm của block, được thực hiện bởi *BusNodeSelection()* như Hình 4-c. Nếu bus-node, là nút được chọn, có chứa random link với nút khác được tạo trong quá trình sinh RSN, link này sẽ bị xóa bỏ. Sau đó, các bus-node mới được thêm vào danh sách BusNodes qua thủ tục tại dòn 7.
- *Xây dựng Bus-way và tạo random link giữa các bus-node*: dựa trên danh sách BusNodes[], được thực hiện bởi thủ tục *BuildBusWay(BusNodes)* được minh họa ở Hình 4-d. Sau khi có được danh sách các bus-node từ các zone, các bus-node được liên kết lần lượt với nhau theo dạng tô-pô bus. Sau đó, các random link được thêm vào giữa 2 bus-node bất kỳ theo phân bố đồng nhất, nghĩa là các bus-node có bậc bằng nhau

### Nút và liên kết giữa các nút

Bus-RSN có 2 loại nút: các bus-node đã đề cập và các nút còn lại, gọi là non-bus-node. Do có cấu trúc cơ sở là tô-pô dạng lưới, một non-bus-node có 4 liên kết lưới tức grid-link giúp liên kết với 4 nút xung quanh, như Hình 5.




Hình 4. Quá trình RSN được chia thành các block, chọn bus-node và tạo Bus-way




Hình 5. Chi tiết về Non-bus-node và Bus-node

Ngoài ra, non-bus-node còn có *r*-random link liên kết với các nút bất kỳ cùng zone với nó. Chúng tôi khuyến nghị sử dụng non-bus-node với *r*=2 random link để bậc của các nút không tăng lên quá cao. Trong các tô-pô khuyến nghị như thế bậc của các non-bus-node là 6.

Trong phạm vi zone, bus-node cũng có thể được coi là một non-bus-node. Ngoài nhiệm vụ như một non-bus-node, bus-node giúp các zone liên kết với nhau, tạo sự liên thông trong đồ thị với 4 grid link, 2 random link. Tuy nhiên, 2 random link này chỉ kết nối tới các bus-node khác. Nếu một random link có 2 mút là 2 bus-node thuộc cùng một zone thì nó sẽ tham gia tích cực vào định tuyến nội vùng, ngược lại, nó giúp làm giảm đáng kể độ giải đường đi (hop-count) trong định tuyến liên vùng. Ngoài ra, bus-node còn có các bus-way link, dùng để liên kết các zone với nhau, tạo sự liên thông cho mạng và thực hiện các nhiệm vụ điều khiển, giám sát. Do vậy, bậc của bus-node là 8, như Hình 5.

### Khuyến nghị sử dụng trong thực tiễn

Bus-RSN được đề xuất cho các DC cỡ nhỏ và vừa phù hợp mới xu hướng phát triển linh hoạt và tiết kiệm trong

nước, nên với những điều kiện bối cảnh cụ thể đó chúng tôi cũng khuyến nghị là các ứng dụng thuê bao máy chủ trên DC nên được gán với nhóm các máy thuộc cùng một sàn. Điều này là khá phù hợp vì các khách hàng của DC loại này cũng thường có nhu cầu không quá lớn. Do đó liên kết giữa các sàn dù chỉ thực hiện hạn chế trên Bus-way cũng sẽ vẫn đảm bảo được nhu cầu truyền tin phục vụ cho quản lý, kiểm soát điều khiển hệ thống (vốn giao thông có tỷ trọng rất nhỏ so với giao thông dữ liệu ứng dụng). Việc sử dụng kiến trúc bus tối giản như thế giúp giảm thiểu chi phí liên kết giữa các sàn có thể ở khá xa nhau. Trong trường hợp có ứng dụng yêu cầu số lượng máy chủ tương đối nhiều và bắt buộc phải triển khai trên nhiều hơn 1 sàn, hệ thống sẽ có thể bổ sung các random link giữa các zone phù hợp sao cho đảm bảo yêu cầu mà không tốn kém (mục IV-2 sẽ trình bày chi tiết thông qua một case-study cụ thể). Đây có thể coi là ưu điểm chính của giải pháp tô-pô đề xuất nhờ vừa đảm bảo tiết kiệm chi phí thiết bị và cáp mạng vừa linh hoạt đáp ứng được yêu cầu cao hơn phát sinh.

## 2. Giải pháp định tuyến cho Bus-RSN

Algo. 1 giải quyết vấn đề sinh mạng kết nối Bus-RSN, tuy nhiên để có thể hoạt động được, cần có thuật toán định tuyến được giải quyết tại Algo. 2 là thủ tục tìm đường đi giữa 2 nút mạng bất kỳ của Bus-RSN. Đầu vào của thuật toán định tuyến là 2 nút mạng, đầu ra đường đi từ nút nguồn đến nút đích. Đường đi giữa 2 nút mạng là danh sách các nút mạng của tô-pô, được sắp xếp theo thứ tự gói tin đi qua từ nút nguồn đến nút đích.

Bảng II
ĐỊNH NGHĨA MỘT SỐ KÍ HIỆU ĐƯỢC SỬ DỤNG

| Ký hiệu | Diễn giải |
|---|---|
| $z_i$ | Định danh zone id của nút $i$ |
| $b_i$ | Định danh block id $i$ |
| $i$ | Node id của nút $i$ |
| $bn_i$ | Bus-node $i$ |
| $cbn_i$ | Bus-node gần nút $i$ nhất |
| *nBlock* | Số block trong một đồ thị RSN |

Với cách tổ chức kiến trúc Bus-RSN đã được trình bày, là mô hình hướng đến các DC được xây dựng trên không gian rời rạc (nhiều phòng, nhiều tầng khác nhau), nên trong mô hình này sử dụng các random link để thu nhỏ đường kính mạng. CORRA là thuật toán định tuyến ưu tiên sử dụng các random link. Vì vậy CORRA có thể tận dụng đặc điểm của mô hình mạng để có một DC có các tham số hiệu năng tốt. Các kết quả được thể hiện trong Phần V. Các ký hiệu chúng tôi sử dụng được chú thích trong Bảng II.

Địa chỉ của một nút trong Bus-RSN bao gồm 3 thành phần: định danh vùng $Zone_{id}(z_i)$, định danh của khối $Block_{id}(b_i)$, $Node_{id}(i)$, được cấu trúc thành $v(z_i,(b_i,i))$. Nút $v(1,(1,8))$ trong Hình 3 là nút thuộc $z_1$, $b_1$ và có $i$=8. Tham số $z_i$ của một nút được quyết định bởi zone quản lý nút đó, được dùng để phân biệt các nút thuộc zone nào. Ví dụ, với 2 nút $u$ và $v$, nếu có $Zone_{id}$ giống nhau thì chúng cùng thuộc 1 zone, ngược lại, chúng thuộc 2 zone khác nhau. Trong Hình 3, hai nút $u,v \in z_1, t \in z_2$. Trong cùng một zone, cần phân biệt các block thuộc zone đó. Hai nút $u,s \in z_1$, nhưng $u \in b_1$ cho nên $b_u$ là 1, còn $s \in b_3$ nên $b_s$ là 3. Cuối cùng là $Node_{id}$ được dùng để phân biệt các nút trong cùng một block. Trong Hình 3, 2 nút $u$ và $v$ cùng thuộc $b_1$, $z_1$, có $Node_{id}$ lần lượt là 2 và 8. Như vậy, tọa độ của mỗi nút là riêng biệt, từ đó hỗ trợ quá trình định tuyến trên DC.

Trong mô hình Bus-RSN, có 2 trường hợp định tuyến: định tuyến nội vùng (intra-zone) và định tuyến liên vùng (inter-zone). Khi 2 nút nguồn đích ở trong cùng một zone (intra-zone), chúng tôi triển khai áp dụng thuật toán định tuyến CORRA[21]. CORRA là một thuật toán rút gọn (compact routing), nên bảng định tuyến của các nút sẽ không phải lưu quá nhiều thông tin. Ngoài ra, với đặc trưng của mình, CORRA tận dụng tối đa các random link như là các cầu nối giữa các vùng hàng xóm, từ đó cải thiện các tham số hiệu năng của mạng. Khi sử dụng thuật toán định tuyến này, non-bus-node sẽ thu thập thông tin các nút cùng với các random link trong vùng hàng xóm của nó là tập hợp các nút cách nút đang xét là 2 nút trung gian (hop) nếu di chuyển bằng grid-link.

Trong trường hợp thứ 2, khi cặp nút nguồn đích ở 2 zone khác nhau, gói tin cần đi qua bus-way. Bus-node có nhiệm vụ giúp các zone liên kết với nhau nên chúng cần lưu thông tin của các bus-node khác. Nói cách khác, bảng định tuyến của non-bus-node $u$ cần lưu thông tin của các nút, random link trong vùng hàng xóm của $u$ và các bus-node có trong mạng. Bảng định tuyến của bus-node cần lưu thông tin của các nút trong block mà nó đại diện cùng với các bus-node khác có trong mạng.

**Hai trường hợp có thể được đặc tả như sau**.

***Định tuyến nội vùng: Intra-zone:*** là khả năng xảy ra khi hai nút cùng nằm trong một zone. Ví dụ, cặp nút nguồn $u,v \in z_1$ trong Hình 3. Chúng tôi sử dụng thuật toán định tuyến CORRA[21] khi định tuyến 2 nút trong cùng 1 zone để có thể sử dụng tối đa các random link, từ đó hiệu năng của mạng liên kết cũng được gia tăng.

***Định tuyến liên vùng: Inter-zone:*** xảy ra khi cặp nút nguồn đích thuộc 2 zone khác nhau, ví dụ $S \in z_i$ và $T \in z_2$ trong Hình 3. Tương tự thuật toán định tuyến TZ[20], chúng tôi chia quá trình định tuyến cho trường hợp này thành 2 giai đoạn. Giai đoạn 1, gói tin từ nút nguồn được định tuyến đến bus-node gần nút đích nhất. Giai đoạn 2, gói tin từ bus-node gần với nút đích nhất di chuyển đến nút đích.

**Thuật toán 2:** Thuật toán định tuyến cho mô hình Bus-RSN

**Đầu vào:** Bus-RSN;
source node $S(z_i,(b_i,i))$;
destination node $T(z_j,(b_j,j))$;
**Đầu ra :** Routing path from $S$ to $T$

```
if (GetZoneId(S) = GetZoneId(T)){
    IntraZone(S,T);
}
else{
    Phase1(S, cbn_T);
    Phase2(cbn_T, T);
}
```

## IV. ĐÁNH GIÁ BẰNG THỰC NGHIỆM

Để thuận tiện cho các độc giả chuyên ngành chúng tôi tiếp tục sử dụng các thuật ngữ kỹ thuật phổ biến trong tiếng Anh (có chú giải tiếng Việt trong xuất hiện lần đầu).

Bảng III
CÁC KÝ HIỆU TRONG CÁC HÌNH MINH HỌA THỰC NGHIỆM

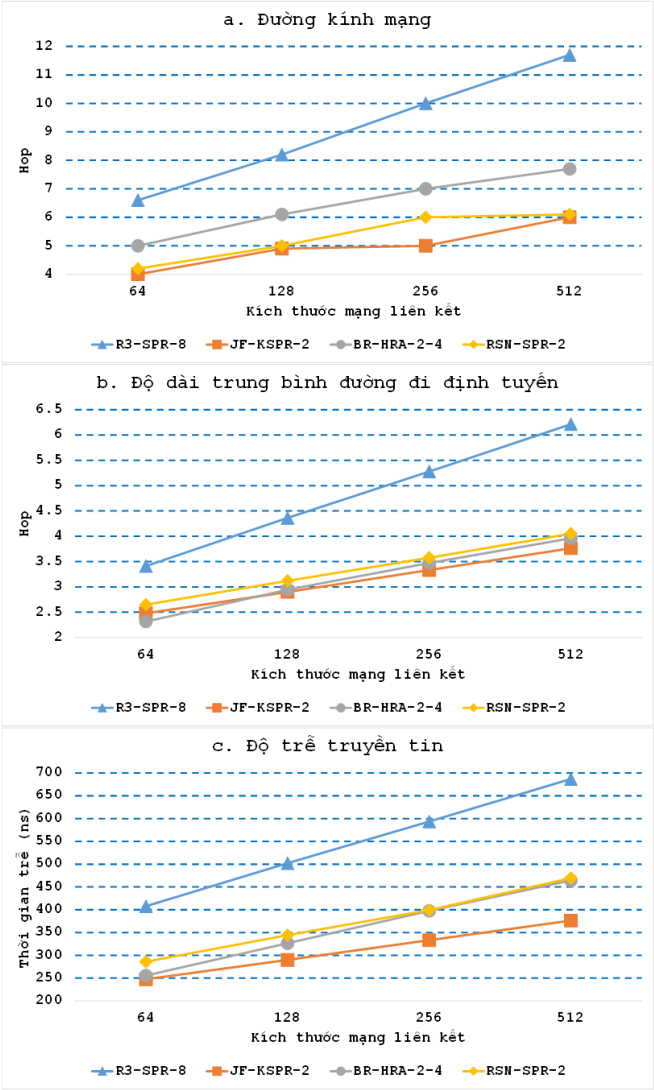
| Tô-pô | Thuật toán định tuyến | Ký hiệu |
|---|---|---|
| Bus-RSN | Hybrid | BR-HRA-2-4 |
| JellyFish | K-shortest path | JF-KSPR-2 |
| R3 | Shortest path | R3-SPR-2 |
| RSN | Shortest path | RSN-SPR-2 |

Chúng tôi tiến hành khảo sát các tham số hiệu năng cơ bản của mạng liên kết, bao gồm diameter (đường kính đồ thị), average routing path length – ARPL (độ dài trung bình đường định tuyến), average latency (độ trễ truyền tin trung bình giữa một cặp nút), routing table size (kích thước bảng định tuyến), total cable length (tổng độ cài cáp mạng), network cost (chi phí thiết bị và cài đặt mạng), của tô-pô Bus-RSN cùng với các tô-pô quan trọng đã đề cập: JellyFish[2], RSN[8], R3[3] thông qua công cụ phần mềm đánh giá có mô phỏng SSiNET[1].

Chúng tôi sử dụng và giải thích các kí hiệu được sử dụng trong các hình minh họa như tại Bảng III. Mỗi ký hiệu bao gồm tên tô-pô, thuật toán định tuyến tương ứng, số zone được sắp xếp. Riêng Bus-RSN có thêm tham số cuối là số lượng block trong mỗi zone, ví dụ 4 có nghĩa là mỗi zone được chia thành 4 khối. Tô-pô R3[3] được tổ chức thành 8 zone để đảm bảo cấu trúc hyper-cube của nó và có số nút mạng tương đương với các tô-pô còn lại. Các mạng liên kết cũng được mô phỏng đặt vào các phòng (sàn) có địa điểm khác nhau, cách nhau 15m.

### 1. Đánh giá hiệu năng mạng liên kết

Trong phần này, chúng tôi trình bày về các tham số hiệu năng của các tô-pô sau khi được khảo sát theo 2 kịch bản.




Hình 6. Đánh giá tham số hiệu năng mạng bằng phân tích đồ thị theo kịch bản 1

Trong kịch bản thử nghiệm 1, các tô-pô được thực hiện khảo sát với cùng kích thước: 64, 128, 256, 512, được lắp đặt ở 2 zone cách nhau 15m. Các zone của tô-pô Bus-RSN được chia thành 4 block.

Hình 6-a là biểu đồ so sánh diameter giữa các mạng liên kết. Có thể thấy JellyFish có kết quả tốt nhất, khi ở cả 4 kích thước, JellyFish cho diameter thấp nhất, dao động từ 4 đến 6 hop, tiếp theo đó là RSN-SPR và Bus-RSN. Điều này có thể lý giải được bởi vì cấu trúc mạng của JellyFish là hoàn toàn ngẫu nhiên cùng thuật toán định tuyến *k-Shortest Path* nên diameter (Hình 6-a) và độ dài trung bình đường định tuyến (Hình 6-b) của JellyFish là thấp nhất, trong mọi kích thước. Tô-pô RSN với kích thước nhỏ nên khi sử dụng thuật toán định tuyến SPR cho diameter khá nhỏ (xấp xỉ JellyFish). R3 cho diameter cao nhất vì tô-pô có cấu trúc

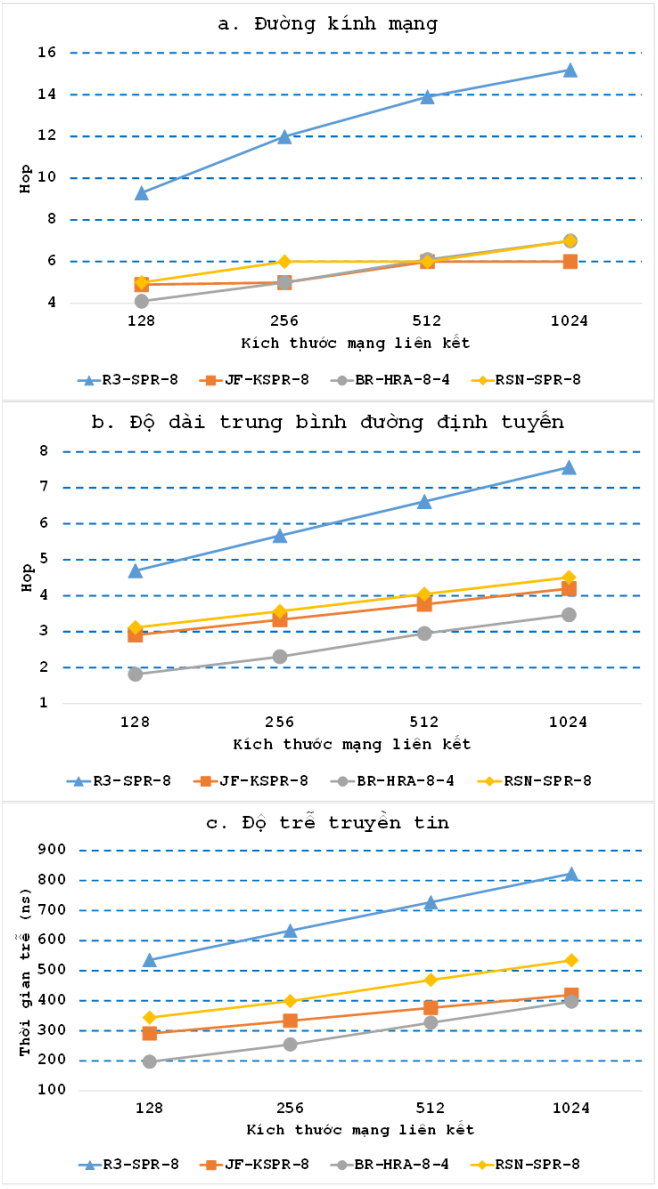
hyber-cube, nên sẽ cần nhiều hop hơn khi gói tin di chuyển. Bus-RSN sử dụng thuật toán CORRA, thuật toán định tuyến ưu tiên sử dụng random link, nên đường kính mạng của nó cũng khá nhỏ, dao động trong khoảng 5.0 đến 7.7 khi kích thước mạng thay đổi từ 64 đến 512.

Do cấu trúc mạng JellyFish bao gồm hoàn toàn là liên kết ngẫu nhiên, ngoài ra nhờ sử dụng thuật toán định tuyến *k-Shortest Path*, nên JellyFish có độ dài trung bình đường định tuyến là thấp nhất trong 4 tô-pô. Bus-RSN và RSN cho kết quả xấp xỉ JellyFish ở các kích thước thử nghiệm khi kích thước mạng tăng từ 64 lên 512. Kết quả trong Hình 6-b cho thấy, đối với các kích thước được thử nghiệm, JellyFish không quá vượt trội so với Bus-RSN. Yếu tố chính khiến độ dài trung bình đường định tuyến của Bus-RSN không quá thua kém so với JellyFish đến từ kích thước của mạng kết nối.

Độ trễ trung bình (tính theo *ns*- nano second) phụ thuộc khá nhiều vào 2 yếu tố: đường kính mạng và độ dài trung bình đường định tuyến. Trong mô hình tính toán, độ trễ truyền tin được tính bằng tổng độ trễ tại các nút mạng (tính theo số hop-count) và tổng độ trễ trên quãng đường định tuyến (tính theo mét). Giá trị độ trễ thấp được xem là hiệu quả. Như đã trình bày trên về đường kính và độ dài trung bình đường định tuyến, nên JellyFish có độ trễ thấp nhất, tiếp đến là Bus-RSN. Ở kích thước 64, độ trễ của JellyFish khoảng 250 , trong khi đó Bus-RSN khoảng 256 . Nhìn chung, Bus-RSN có độ trễ cao hơn Jellyfish, nhưng trung bình độ trễ toàn mạng là chấp nhận được, ví dụ Bus-RSN có độ trễ trung bình cao hơn 12,61% so với JellyFish ở kích thước mạng là 128. Kết quả độ trễ truyền tin trung bình giữa một cặp nút của các tô-pô được trình bày trong Hình 6-c.

Sau khi thực hiện khảo sát các tô-pô với kịch bản 1, chúng tôi tiếp tục khảo sát với kích thước lớn hơn: 128, 256, 512, 1024. Trong kịch bản này, các tô-pô được lắp đặt tại 8 zone, cách nhau đôi một 15m. Các zone của tô-pô Bus-RSN được chia thành 4 block.

Hình 7-a là kết quả chi tiết về đường kính mạng của các tô-pô. Ta có thể thấy rằng 3 tô-pô Bus-RSN, R3, RSN với các thuật toán định tuyến được sử dụng có đường kính mạng xấp xỉ nhau, dao động từ 4 đến 7. RSN và JellyFish sử dụng các thuật toán định tuyến mà bản chất là sử dụng đường đi ngắn nhất nên không bất ngờ khi 2 tô-pô này có đường kính mạng nhỏ. Bus-RSN sử dụng thuật toán CORRA, là thuật toán định tuyến ưu tiên sử dụng các random link. Với cấu trúc của Bus-RSN, các liên kết ngẫu nhiên xuất hiện ở mọi node trong tô-pô làm cho đường kính mạng của mạng nhỏ lại. R3 có đường kính mạng thấp vì các node mạng của tô-pô này thuộc nhiều zone khác nhau. Nên khi định tuyến, chúng phải đi qua nhiều nút biên trong R3), khiến cho đường kính mạng của R3 lớn hơn các tô-pô còn lại.




Hình 7. Đánh giá tham số hiệu năng mạng bằng phân tích đồ thị theo kịch bản 2

Khi chia mạng kết nối thành nhiều zone các nhau, Bus-RSN cho thấy lợi thế của mình với cấu trúc nhiều liên kết ngẫu nhiên và thuật toán ưu tiên sử dụng chúng. Điều này thể hiện trong hình 5, khi độ dài trung bình đường định tuyến của Bus-RSN thấp nhất trong 4 tô-pô, dao động trong khoảng từ 1.8 đến 2.5 với kích thước mạng từ 128 đến 1024. Trong điều kiện hạn chế cáp mạng, số lượng kết nối giảm xuống dẫn đến bậc của các nút mạng trong mạng JellyFish giảm xuống khiến mô hình này mất lợi thế. độ dài trung bình đường định tuyến của JellyFish hơn Bus-RSN khoảng 60% ở kích thước 128 và 20% ở kích thước 1024. RSN có độ dài trung bình đường định tuyến xấp xỉ JellyFish. Do cấu trúc mạng đã được giải thích ở phần đường kính mạng,

R3 có độ dài trung bình đường định tuyến thấp nhất trong các tô-pô với các kích thước được thử nghiệm.

Hình 7-c là kết quả thử nghiệm chi tiết độ trễ truyền tin trung bình giữa một cặp nút của các tô-pô. Độ trễ truyền tin trung bình giữa một cặp nút là thông số phụ thuộc vào đường kính mạng và độ dài trung bình đường đi định tuyến, nên không khó hiểu khi Bus-RSN có độ trễ truyền tin trung bình giữa một cặp nút tốt nhất trong các tô-pô với các kích thước thử nghiệm từ khoảng 200ns ở kích thước 128 đến 400ns ở kích thước 1024. Có thể thấy, với các kích thước nhỏ, Bus-RSN có lợi thế đối với JellyFish.

## 2. Đánh giá chi phí triển khai

Sau khi khảo sát các tham số hiệu năng, chi phí để xây dựng các DC theo các mô hình tô-pô trên được chúng tôi đánh giá theo 2 kịch bản đã được đưa ra.

Với kịch bản 1, các tô-pô được thực hiện khảo sát với cùng kích thước: 64, 128, 256, 512, được lắp đặt ở 2 zone cách nhau 15m. Các zone của tô-pô Bus-RSN được chia thành 4 block.

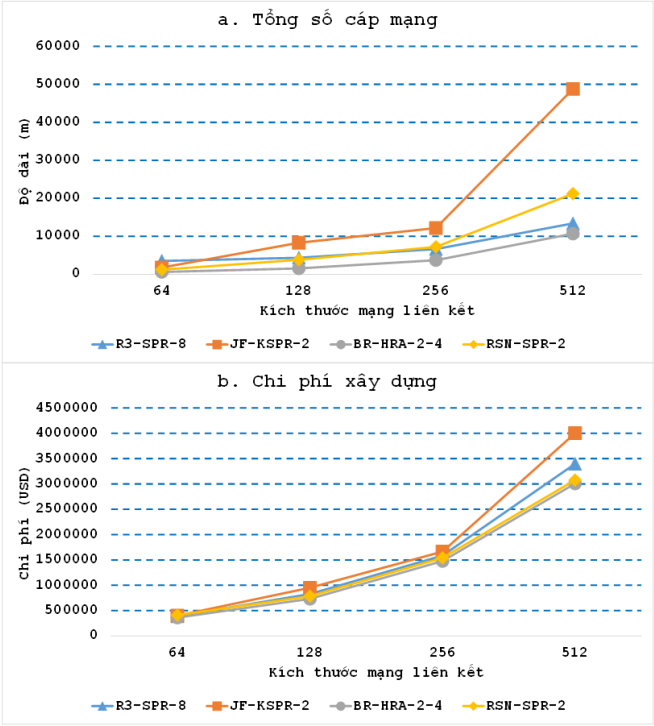
Tổng chiều dài cáp (Total Cable length) là lợi thế đặc thù của Bus-RSN. Do cấu trúc của Bus-RSN là các zone tách biệt được kết nối thông qua 1 bus-way, cho nên khi DC được lắp đặt tại các phòng khác nhau sẽ chỉ tốn thêm chi cáp nối cho việc lắp đặt bus-way. Nhưng đối với JellyFish, khi xây dựng theo mô hình này, các liên kết ngẫu nhiên sẽ bị kéo dài khi DC phải lắp đặt tại các phòng (sàn) riêng biệt.

Kết quả trong Hình 8-a cho thấy rõ sự bất lợi về số lượng cáp nối khi mô hình JellyFish[2] phải lắp đặt tại các DC như vậy. Thử nghiệm cho thấy Bus-RSN có kết quả tốt nhất khi sử dụng ít cáp nối và thấp hơn JellyFish, thấp hơn 81,53% ở kích thước mạng 128 nút.

Chi phí đầu tư cho thiết bị mạng chiếm một phần quan trọng trong chi phí cài đặt mạng liên kết. Để đơn giản hóa mô hình tính toán, chúng tôi lựa chọn chi phí cho một thiết bị chuyển mạch ở mức 500$ cho mỗi cổng (port) dựa trên khảo sát trong[25]. Mỗi cổng tương ứng với một liên kết của thiết bị chuyển mạch đó trong mạng liên kết. Chi phí cài đặt mạng còn bao gồm cả chi phí cho các dây nối tức cáp mạng (cable) dùng để liên kết các thiết bị với nhau. Chi phí dành cho một kết nối bằng giá trị của dây nối đó (bao gồm chi phí dây nối, và chi phí đầu kết nối) cộng thêm 25% chi phí trung bình dành cho nhà sản xuất (chi phí sản xuất, phân phối, tiền lãi) và chi phí để lắp đặt thực tế (installation cost).

$$Cable\_cost = (Cable\_length * Cost\_per\_m +$$

$$Connector\_Cost) * 1.25 + Installation\_Cost$$




Hình 8. Tổng chiều dài cáp và tổng chi phí triển khai kết nối theo kịch bản 1

Chi phí trung bình cho việc cài đặt các kết nối giữa hai nút mạng bên trong một tủ mạng yêu cầu chi phí 2.5$ và 6.5$ đối với kết nối giữa hai nút mạng ở hai tủ mạng khác nhau. Chi phí cho mỗi mét cáp quang là $Cost\_per\_m = 10/m$, và mỗi đầu nối $Connector\_Cost = 188$\$. Vậy chi phí cho kết nối này được tính bằng $(10*5+188*2)*1.25+6.5 = 539$\$. Tổng chi phí để xây dựng DC được tính theo công thức:

$$Total\_Cost = Cable\_cost +$$

$$switch\_port * 500 * network\_size$$

Sau đây chúng tôi thử khảo sát chi phí cho một số trường hợp ứng dụng có thể gặp trong thực tiễn (case studies). Đầu tiên ta khảo sát trường hợp như đã đề cập ở mục III.1 (phần khuyến nghị sử dụng), khi mà có ứng dụng khai thác yêu cầu sử dụng vượt quá số lượng máy chủ trong một phòng, ví dụ như cần tất cả các máy chủ tại vùng $z_1$ và một phần máy chủ trong vùng $z_2$. Trong trường hợp đó, Bus-RSN cung cấp khả năng linh hoạt trong việc kết nối giữa hai vùng xác định bằng cách thay đổi một số kết nối trong vùng $z_1$ tới một số nút mạng trong vùng $z_2$. Bằng cách thay thế các thiết bị switch (chuyển mạch) tại các bus-node từ loại 8 cổng lên 12 cổng, làm gia tăng các cầu kết nối tới các bus-node khác trong vùng $z_2$. Đồng thời cấu hình lại địa chỉ của các nút được chọn trong vùng đó, sao cho tương đồng với các địa chỉ các nút trong vùng nhưng không cần

thay đổi vị trí (địa lý) của các nút đó. Có thể thấy rõ là, việc thay thế switch cũng như chi phí cho cáp mạng so với tổng chi phí xây dựng DC là không đáng kể và các switch được thay thế có thể tận dụng lại trong việc mở rộng DC sau này. Ví dụ, yêu cầu tính toán cần zone $z_1$ và block $b_5$, $b_6$, thuộc zone $z_2$. Với tính linh hoạt của Bus-RSN, hiệu suất tính toán có thể tăng lên bằng cách thêm liên kết giữa các bus-node thuộc zone $z_1$ và bus-node của block $b_5$ và $b_6$. Như vậy, chúng ta chỉ cần thay 2 switch có 8 port, 1 switch tương ứng với bus-node thuộc zone $z-1$ và 1 switch tương ứng với bus-node của $b_5$ hoặc $b_6$, bằng loại switch 12 port. Với cấu hình hiện tại, mỗi bus-node khi liên kết đến zone khác cần 15m cáp mạng. Chi phí cho sự thay đổi này là: 12*500*2 + 15*10 = 12.150 (USD), so với chi phí xây dựng ban đầu là chỉ tăng thêm khoảng 2,2% – khá khiêm tốn!

Chúng tôi tiếp tụ khảo sát chi phí với kịch bản 2.

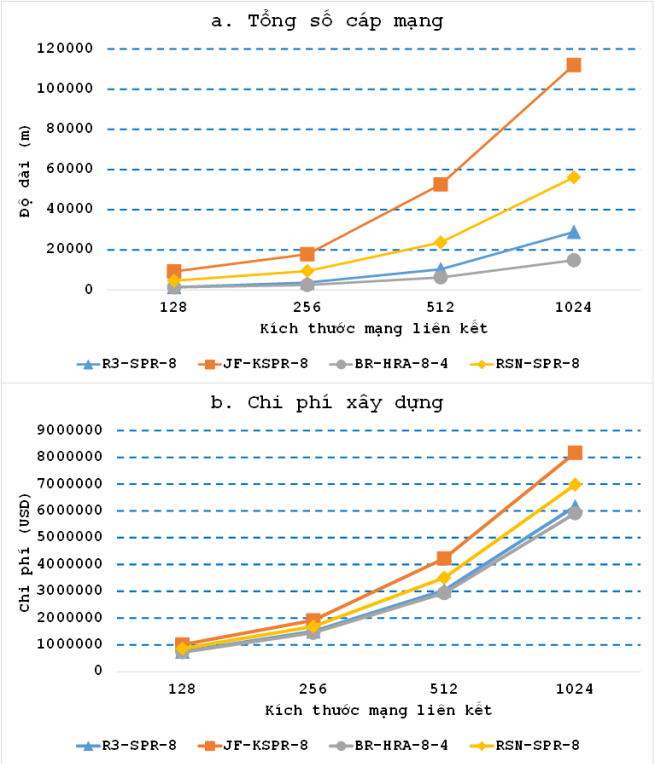
Được thiết kế để phù hợp với các không gian phân biệt nên Bus-RSN sử dụng cáp mạng ít nhất trong 4 tô-pô mạng với các kích thước được thử nghiệm, 1344 m ở kích thước 128 và 14861 ở kích thước 1024. Với kích thước 128, JellyFish sử dụng nhiều hơn 5.89 lần số lượng cáp mạng mà Bus-RSN cần, con số này là 6.5 với kích thước 1024. Kết quả chi tiết được thể hiện trong Hình 9-a.

Hình 9-b là kết quả chi tiết khi khảo sát tổng chi phí cho các tô-pô mạng dựa trên kịch bản thí nghiệm. Trong cùng một điều kiện (chi phí lắp đặt, chi phí vật liệu...) Bus-RSN là tô-pô có chi phí thấp nhất. Do các mạng kết nối có cùng số lượng switch, Bus-RSN sử dụng ít cáp mạng nhất (Hình 7) nên không khó hiểu khi Bus-RSN có chi phi lắp đặt tiết kiệm nhất trong kịch bản thử nghiệm, khoảng 726.573 USD ở kích thước 64 và 5.914.163 USD ở kích thước 1024. Trong khi đó, chi phí cần cho mô hình JellyFish hơn 38.9% so với Bus-RSN.

Chúng tôi tiến hành khảo sát kỹ hơn, mô phỏng việc lắp đặt DC trên các sàn tương tự thực tế bằng 2 cấu hình. Cấu hình 1 với DC được triển khai trong 4 zone có khoảng cách đồng đều nhau và cách nhau 15 mét. Trong cấu hình 2, giả sử rằng, ban đầu DC được lắp đặt 2 zone $z_1$ và $z_2$ cạnh nhau, với khoảng cách cáp kết nối là 15 mét (cùng tầng, ví dụ tầng 1). Sau đó, DC được mở rộng với $z_3$ được lắp đặt ở tầng trên (ví dụ tầng 2), cách tầng 2 là 15m. Cuối cùng, DC tiếp tục được mở rộng với $z_4$, 15m cao hơn tầng 3.

Với thiết kế 4 zone, tổng chiều dài cáp của Bus-RSN thấp hơn nhiều chiều dài cáp của JellyFish, chỉ bằng 22,18% và 17,43% với kích thước 128 và 256 (Bảng 4), tương ứng. Với cấu hình 2, khoảng cách giữa các zone tăng lên đáng kể, thì tổng chiều dài cáp của Bus-RSN vẫn duy trì mức thấp hơn của Jellyfish, ví dụ chỉ chiếm 22,75% tại kích thước mạng 256.

Ngoài ra, chúng tôi thực nghiệm với 4 sàn có kích thước




Hình 9. Tổng chiều dài cáp và tổng chi phí triển khai kết nối theo kịch bản 2

và khoảng cách khác nhau, các mặt sàn có kích thước $z_1$, $z_2$ và $z_3$ lần lượt là 32 nút và $z_4$ là 128, tổng nút toàn mạng là 224 nút. Kết quả thực nghiệm cho thấy, Bus-RSN sử dụng ít cáp mạng hơn 83,19% và đạt tổng chi phí thấp hơn 11,2% khi so sánh với JellyFish. Lý do chính là bởi số lượng cáp kết nối giữa các nút mạng liên vùng của tô-pô nhiều và tỏ ra bất lợi khi kết nối giữa các phòng máy chủ đặt cách xa nhau. Do vậy, mô hình Bus-RSN phù hợp với điều kiện triển khai thực tế khi các phòng lắp đặt máy chủ được bố trí ở xa nhau và có kích thước khác nhau.

Bảng IV
TỔNG CÁP MẠNG TRONG TRƯỜNG HỢP KHÁC NHAU VỀ KHOẢNG CÁCH GIỮA CÁC ZONE

| **Cấu hình 1 (15 mét đều)** | | | |
|---|---|---|---|
| Tổng nút | BR-HRA-4-4 | JF-kSPR-4 | Tỉ lệ % |
| 64 | 653,95 | 3.304,90 | 19,79% |
| 128 | 1.734,40 | 7.819,10 | 22,18% |
| 256 | 3.145,05 | 18.042,40 | 17,43% |
| 512 | 7.404,80 | 46.776,40 | 15,83% |
| **Cấu hình 2 (khoảng cách không đều)** | | | |
| 64 | 798,70 | 3.422,60 | 23,34% |
| 128 | 1.870,25 | 8.220,60 | 22,75% |
| 256 | 3.293,95 | 19.058,40 | 17,28% |
| 512 | 7.563,20 | 49.048,10 | 15,42% |

## V. KẾT LUẬN

Trong bài báo này, chúng tôi đề xuất mô hình tô-pô lai Bus-RSN nhằm giải quyết bài toán xây dựng DC của các doanh nghiệp nhỏ và vừa. Với đặc thù thiết kế nhắm đến tính kinh tế và linh hoạt, Bus-RSN có thể lắp đặt trong không gian gồm nhiều phòng/sàn phân biệt. Ngoài ra, chi phí đầu tư ban đầu để xây dựng DC theo mô hình này phù hợp với các doanh nghiệp có nguồn vốn hạn hẹp. Tính linh hoạt trong khả năng mở rộng giúp doanh nghiệp có thể mở rộng DC một cách dễ dàng.

Các kết quả thí nghiệm được chúng tôi tổng hợp và trình bày chi tiết tại phần IV. Với những kết quả đạt được, có thể thấy rằng, Bus-RSN là một mô hình mạng tiềm năng khi sở hữu các tham số hiệu năng tốt và chi phí lắp đặt khiêm tốn.

Trong kịch bản thử nghiệm 1, các tham số hiệu năng: đường kính mạng, average path length và độ trễ truyền tin trung bình giữa một cặp nút không phải là điểm mạnh Bus-RSN khi đạt giá trị kém hơn của JellyFish nhưng không phải là kém nhiều, ví dụ độ trễ kém 12,61% khi so với JellyFish ở kích thước mạng 128, nhưng độ trễ này là chấp nhận được trong các ứng dụng nhỏ, DC vừa và nhỏ.

Tuy nhiên, khi thực hiện kịch bản thử nghiệm 2, Bus-RSN cho thấy kết quả tốt khi các tham số hiệu năng và chi phí đều tốt nhất. Ví dụ: độ dài trung bình đường định tuyến của JellyFish hơn Bus-RSN khoảng 60% ở kích thước 128 và 20% ở kích thước 1024.

Trong cả 2 kịch bản thử nghiệm, tổng chiều dài cáp và chi chí thiết bị tiết kiệm của Bus-RSN đều tốt hơn so với các tô-pô trên. Trong kịch bản thử nghiệm 1, tổng trung bình cáp ít hơn 81,53% và nhờ thế tổng chi phí giảm 23,08% khi so với JellyFish ở kích thước mạng 128. Với kịch bản 2, JellyFish sử dụng nhiều cáp mạng hơn 5.89 lần số lượng cáp mạng mà Bus-RSN cần ở kích thước 128, con số này là 6.5 ở kích thước 1024.

Cấu trúc mạng có thể lắp đặt với switch thương mại cùng với số lượng cáp mạng nhỏ dẫn đến chi phí để đầu tư xây dựng DC theo mô hình Bus-RSN là khá tiết kiệm so với các mô hình khác. Như vậy, mô hình Bus-RSN tiềm năng, phù hợp với các DC cỡ vừa và nhỏ, mà trong đó không đòi hỏi tốc độ tính toán thật cao (chấp nhận độ trễ) nhưng đảm bảo chi phí lắp đặt thấp.

Trong phạm vi của bài báo này, mô hình Bus-RSN mới chỉ được phân tích và đánh giá thông qua các thông số hiệu năng cơ sở (yếu tố đồ thị) mà chưa được thử nghiệm trong môi trường giả lập thực tế. Trong tương lai, chúng tôi dự định sẽ đánh giá các yếu tố hiệu năng nâng cao như thông lượng và điện năng tiêu thụ của mạng để có thể đưa ra đánh giá đầy đủ nhất về mô hình Bus-RSN.

**Ghi nhận**

Nghiên cứu này được tài trợ bởi Quỹ Phát triển khoa học và công nghệ Quốc gia (NAFOSTED) trong đề tài mã số 102.02-2017.316.

## TÀI LIỆU THAM KHẢO

[1] Kiều Thành Chung, Nguyễn Tiến Thành, Nguyễn Khanh Văn, *"Một tiếp cận thiết kế công cụ phần mềm đánh giá hiệu năng mạng liên kết kích thước lớn"*, Chuyên san Các công trình Nghiên cứu và Phát triển Công nghệ thông tin và Truyền thông, 2019

[2] Singla, Ankit and Hong, Chi-Yao and Popa, Lucian and Godfrey, P Brighten, *"Jellyfish: Networking data centers randomly"*, Presented as part of the 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12), 225-238, 2012

[3] Luo, Lailong and Guo, Deke and Li, Wenxin and Zhang, Tian and Xie, Junjie and Zhou, Xiaolei, *"Compound graph based hybrid data center topologies"*, Frontiers of Computer Science, 9, 6, 860-874, 2015

[4] Al-Fares, Mohammad and Loukissas, Alexander and Vahdat, Amin, *"A scalable, commodity data center network architecture"*, ACM SIGCOMM computer communication review, 38, 4, 63-74, 2008

[5] Lebiednik, Brian and Mangal, Aman and Tiwari, Niharika, *"A survey and evaluation of data center network topologies"*, arXiv preprint arXiv:1605.01701, 2016

[6] Guo, Deke and Chen, Tao and Li, Dan and Liu, Yunhao and Liu, Xue and Chen, Guihai, *"BCN: Expansible network structures for data centers using hierarchical compound graphs"*,2011 Proceedings IEEE INFOCOM, 61-65, 2011

[7] Guo, Chuanxiong and Wu, Haitao and Tan, Kun and Shi, Lei and Zhang, Yongguang and Lu, Songwu, *"Dcell: a scalable and fault-tolerant network structure for data centers"*, Proceedings of the ACM SIGCOMM 2008 conference on Data communication, 75–86, 2008

[8] Koibuchi, Michihiro and Matsutani, Hiroki and Amano, Hideharu and Hsu, D Frank and Casanova, Henri, *"A case for random shortcut topologies for HPC interconnects"*, 2012 39th Annual International Symposium on Computer Architecture (ISCA), 177–188, 2012

[9] Farrington, Nathan and Porter, George and Radhakrishnan, Sivasankar and Bazzaz, Hamid Hajabdolali and Subramanya, Vikram and Fainman, Yeshaiahu and Papen, George and Vahdat, Amin, *"Helios: a hybrid electrical/optical switch architecture for modular data centers"*, Proceedings of the ACM SIGCOMM 2010 conference, 339–350, 2010

[10] Guo, Chuanxiong and Lu, Guohan and Li, Dan and Wu, Haitao and Zhang, Xuan and Shi, Yunfeng and Tian, Chen and Zhang, Yongguang and Lu, Songwu, *"BCube: a high performance, server-centric network architecture for modular data centers"*, Proceedings of the ACM SIGCOMM 2009 conference on Data communication, 63–74, 2009

[11] Gyarmati, László and Trinh, Tuan Anh, *"Scafida: A scale-free network inspired data center architecture"*, ACM SIGCOMM Computer Communication Review, 40, 5, 4–12, 2010

[12] Wu, Haitao and Lu, Guohan and Li, Dan and Guo, Chuanxiong and Zhang, Yongguang, *"MDCube: a high performance network structure for modular data center interconnection"*, Proceedings of the 5th international conference on Emerging networking experiments and technologies, 25–36, 2009

[13] Greenberg, Albert and Hamilton, James R and Jain, Navendu and Kandula, Srikanth and Kim, Changhoon and Lahiri, Parantap and Maltz, David A and Patel, Parveen and Sengupta, Sudipta, *"VL2: a scalable and flexible data center network"*, Proceedings of the ACM SIGCOMM 2009 conference on Data communication, 51–62, 2009

[14] Shin, Ji-Yong and Wong, Bernard and Sirer, Emin Gün, *"Small-world datacenters"*, Proceedings of the 2nd ACM Symposium on Cloud Computing, 1–13, 2011

[15] Yu, Ye and Qian, Chen, *"Space shuffle: A scalable, flexible, and high-bandwidth data center network"*, 2014 IEEE 22nd International Conference on Network Protocols, 13–24, 2014

[16] Fujiwara, Ikki and Koibuchi, Michihiro and Matsutani, Hiroki and Casanova, Henri, *"Skywalk: A topology for HPC networks with low-delay switches"*, 2014 IEEE 28th International Parallel and Distributed Processing Symposium, 263–272, 2014

[17] Kim, John and Dally, Wiliam J and Scott, Steve and Abts, Dennis, *"Technology-driven, highly-scalable dragonfly topology"*, 2008 International Symposium on Computer Architecture, 77–88, 2008

[18] Niranjan Mysore, Radhika and Pamboris, Andreas and Farrington, Nathan and Huang, Nelson and Miri, Pardis and Radhakrishnan, Sivasankar and Subramanya, Vikram and Vahdat, Amin, *"Portland: a scalable fault-tolerant layer 2 data center network fabric"*, Proceedings of the ACM SIGCOMM 2009 conference on Data communication, 39–50, 2009

[19] Wang, Guohui and Andersen, David G and Kaminsky, Michael and Papagiannaki, Konstantina and Ng, TS Eugene and Kozuch, Michael and Ryan, Michael, *"c-Through: Part-time optics in data centers"*, Proceedings of the ACM SIGCOMM 2010 conference, 327–338, 2010

[20] Thorup, Mikkel and Zwick, Uri, *"Compact routing schemes"*, Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures, 1–10, 2001

[21] Thanh, Chung Kieu and The, Anh Mai and Bui, Cuong and Pham, Hai D and Nguyen, Khanh-Van, *"An efficient compact routing scheme for interconnection topologies based on the random model"*, Proceedings of the Eighth International Symposium on Information and Communication Technology, 189–196, 2017

[22] Dally, William James and Towles, Brian Patrick, *"Principles and practices of interconnection networks"*, 2004

[23] Kieu, Thanh-Chung and Nguyen, Khanh-Van and Truong, Nguyen T and Fujiwara, Ikki and Koibuchi, Michihiro, *"An interconnection network exploiting trade-off between routing table size and path length"*, 2016 Fourth International Symposium on Computing and Networking (CANDAR), 666–670, 2016

[24] Aljazzar, Husain and Leue, Stefan, *"K∗: A heuristic search algorithm for finding the k shortest paths"*, Artificial Intelligence, 175, 18, 2129–2154, 2011

[25] Mudigonda, Jayaram and Yalagandula, Praveen and Mogul, Jeffrey C, *"Taming the Flying Cable Monster: A Topology Design and Optimization Framework for Data-Center Networks."*, USENIX Annual Technical Conference, 2011

## SƠ LƯỢC VỀ CÁC TÁC GIẢ

### Kiều Thành Chung

Tốt nghiệp kỹ sư Công nghệ Thông tin năm 2003, tại Trường Đại học Bách Khoa Hà Nội (HUST), Thạc sĩ Công nghệ Thông tin năm 2010. Năm 2016, tác giả nghiên cứu tại Viện Công nghệ Thông tin Quốc Gia Nhật Bản (NII). Hiện nay là Nghiên cứu sinh tại Viện Công nghệ Thông tin và Truyền thông (SoICT)-HUST. Lĩnh vực nghiên cứu: mạng liên kết, thuật toán định tuyến.

**Vũ Quang Sơn**

Sinh ngày: 09/06/1999
Hiện là sinh viên ngành công nghệ thông tin tại Học viện Công nghệ Bưu chính Viễn thông.
Tác giả đang học tập và nghiên cứu tại SEDIC-LAB thuộc Trường Đại học Bách Khoa Hà Nội.
Các lĩnh vực mà tác giả hiện đang nghiên cứu: mạng liên kết, thuật toán định tuyến và ứng dụng mô phỏng giả lập truyền tin.

**Nguyễn Đăng Hải**

Tốt nghiệp trường Đại học Bách khoa Hà Nội năm 1995; nhận bằng Thạc sĩ Tin học tại Viện tin học Pháp ngữ năm 1997; nhận học vị Tiến sĩ Khoa học Máy tinh của trường Thực hành Công nghệ cao – Cộng hòa Pháp năm 2010. Hiện là Giảng viên chính tại bộ môn Khoa học Máy tính, Viện Công nghệ thông tin và truyền thông, trường Đại học Bách khoa Hà Nội. Lĩnh vực nghiên cứu: tính toán hiệu năng cao, mô phỏng song song và phân tán, hệ điều hành nhúng.

**Nguyễn Khanh Văn**

Tốt nghiệp Kỹ sư Tin học tại Đại học Bách Khoa năm 1992, Thạc sỹ Khoa học Máy tính tại Đại học Wollongong (Úc) năm 2000, Tiến sĩ Khoa học Máy tính tại Đại học California-Davis (Mỹ) năm 2006. Hiện là Phó Giáo sư, giảng dạy và nghiên cứu tại Viện Công nghệ thông tin và truyền thông, Đại học Bách khoa Hà Nội. Lĩnh vực nghiên cứu: thuật toán và các mô hình lý thuyết trong tính toán phân tán và mạng máy tính (mạng liên kết, mạng cảm biến không dây), an toàn thông tin.