HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Thanh Trung

NGHIÊN CỨU CHÍNH SÁCH BỀN VỮNG NHẰM XÂY DỰNG THUẬT TOÁN NÂNG CAO HIỆU QUẢ CÂN BẰNG TẢI CỦA ĐIỆN TOÁN ĐÁM MÂY

LUẬN VĂN THẠC SĨ KỸ THUẬT

(Theo định hướng ứng dụng)

TP. HỒ CHÍ MINH – NĂM 2022

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Thanh Trung

NGHIÊN CỨU CHÍNH SÁCH BỀN VỮNG NHẰM XÂY DỰNG THUẬT TOÁN NÂNG CAO HIỆU QUẢ CÂN BẰNG TẢI CỦA ĐIỆN TOÁN ĐÁM MÂY

Chuyên ngành: **HỆ THỐNG THÔNG TIN**

Mã số: **8.48.01.04**

LUẬN VĂN THẠC SĨ KỸ THUẬT

(Theo định hướng ứng dụng)

NGƯỜI HƯỚNG DẪN KHOA HỌC:

PGS.TS. TRẦN CÔNG HÙNG

TP. HỒ CHÍ MINH – NĂM 2022

LỜI CAM ĐOAN

Tôi cam đoan rằng luận văn: *"***Nghiên cứu chính sách bền vững nhằm xây dựng thuật toán nâng cao hiệu quả cân bằng tải của điện toán đám mây***"* là công trình nghiên cứu của chính tôi.

Tôi cam đoan các số liệu, kết quả nêu trong luận văn là trung thực và chưa từng được ai công bố trong bất kỳ công trình nào khác.

Không có sản phẩm/nghiên cứu nào của người khác được sử dụng trong luận văn này mà không được trích dẫn theo đúng quy định.

TP. Hồ Chí Minh, ngày 25 tháng 01 năm 2022

Học viên thực hiện luận văn

Nguyễn Thanh Trung

LỜI CẢM ƠN

Trong suốt quá trình học tập và nghiên cứu thực hiện luận văn, ngoài nỗ lực của bản thân, tôi đã nhận được sự hướng dẫn nhiệt tình quý báu của quý Thầy Cô, cùng với sự động viên và ủng hộ của gia đình, bạn bè và đồng nghiệp. Với lòng kính trọng và biết ơn sâu sắc, tôi xin gửi lời cảm ơn chân thành tới:

Ban Giám Đốc, Phòng đào tạo sau đại học và quý Thầy Cô đã tạo mọi điều kiện thuận lợi giúp tôi hoàn thành luận văn.

Tôi xin chân thành cảm ơn Thầy PGS.TS Trần Công Hùng, người thầy kính yêu đã hết lòng giúp đỡ, hướng dẫn, động viên, tạo điều kiện cho tôi trong suốt quá trình thực hiện và hoàn thành luận văn.

Tôi xin chân thành cảm ơn gia đình, bạn bè, đồng nghiệp trong cơ quan đã động viên, hỗ trợ tôi trong lúc khó khăn để tôi có thể học tập và hoàn thành luận văn. Mặc dù đã có nhiều cố gắng, nỗ lực, nhưng do thời gian và kinh nghiệm nghiên cứu khoa học còn hạn chế nên không thể tránh khỏi những thiếu sót. Tôi rất mong nhận được sự góp ý của quý Thầy Cô cùng bạn bè đồng nghiệp để kiến thức của tôi ngày một hoàn thiện hơn.

Xin chân thành cảm ơn!

TP. Hồ Chí Minh, ngày 25 tháng 01 năm 2022

Học viên thực hiện luận văn

Nguyễn Thanh Trung

MỤC LỤC

DANH MỤC CÁC THUẬT NGỮ, CHỮ VIẾT TẮT

Viết tắt	Tiếng Anh	Tiếng Việt
AI	Artificial Intelligence	Trí tuệ nhân tạo
Cloud	Cloud computing environment	Môi trường điện toán đám mây
CC	Cloud Computing	Điện toán đám mây
ML	Machine Learning	Học máy
LB	Load Balancing	Cân bằng tải

DANH SÁCH BẢNG

DANH SÁCH HÌNH VẼ

MỞ ĐẦU

1. Tính cấp thiết của đề tài

Trong thời đại ngày nay, công nghệ thông tin và truyền thông ngày càng phát triển, đòi hỏi nhu cầu xử lý thông tin ngày càng cao, cần có hệ thống có khả năng lưu trữ và khai thác được một lượng dữ liệu lớn. Sự phát triển không ngừng của nền kinh tế thế giới và trong nước đòi hỏi các doanh nghiệp, các tập đoàn lớn phải có giải pháp để lưu trữ và khai thác thông tin về các dữ liệu lớn liên quan đến công việc kinh doanh của họ. Việc trang bị máy chủ vật lý đòi hỏi phải có một bộ phận kỹ thuật am hiểu về công nghệ thông tin để quản trị và vận hành hệ thống. Đồng thời cũng mất nhiều chi phí để đầu tư, nâng cấp phần mềm, phần cứng, phí bảo trì, nhân công...

Chính vì thế, điện toán đám mây (cloud computing) là một trong những giải pháp đang thu hút được một số lượng lớn các doanh nghiệp sử dụng. Điện toán đám mây thực chất là mô hình các máy chủ ảo, sử dụng các công nghệ máy tính và phát triển dựa vào mạng Internet. Ở mô hình điện toán này, mọi khả năng liên quan đến công nghệ thông tin đều được cung cấp dưới dạng các "dịch vụ". “Dịch vụ” này cho phép người sử dụng truy cập các dịch vụ công nghệ từ một nhà cung cấp nào đó "trong đám mây" mà không cần phải có các kiến thức, kinh nghiệm về công nghệ đó. Ngoài ra, người dùng cũng không cần quan tâm đến các cơ sở hạ tầng phục vụ công nghệ mà mình được cung cấp.

Điện toán đám mây giải quyết các vấn đề tối ưu hóa lưu trữ, ảo hóa máy chủ, cơ sở hạ tầng mạng với mục đích mang lại dịch vụ với chất lượng tốt nhất. Các tập đoàn, doanh nghiệp và cá nhân người dùng chỉ cần trả phí với dịch vụ tương ứng mà họ sử dụng.

Nói về chất lượng dịch vụ trên điện toán đám mây, người dùng cảm thấy chất lượng dịch vụ đáp ứng tốt trong công việc quản lý điều hành, lưu trữ và khai thác tài nguyên. Mặt khác, việc quản lý tài nguyên trở thành một công việc phức tạp đối với các nhà cung cấp dịch vụ đám mây. Có một số vấn đề được đặt ra: làm sao khắc phục vấn đề thiếu tài nguyên, giảm độ trễ trên đám mây và khả năng cải thiện hiệu suất mạng khi nhiều người dùng sử dụng cùng lúc.

Để giải quyết các vấn đề trên, hiện nay đã có hệ thống cân bằng tải để phân bổ đồng đều lưu lượng truy cập giữa hai hay nhiều các máy chủ có cùng chức năng trong cùng một hệ thống. Bằng cách đó, sẽ giúp cho hệ thống cung cấp dịch vụ của nhà cung cấp giảm thiểu một cách tối đa tình trạng một máy chủ bị quá tải và ngưng hoạt động.

Hiện nay có nhiều thuật toán cân bằng tải trên các dịch vụ đám mây. Tuy nhiên, hiệu quả của các thuật toán vẫn còn nhiều hạn chế, chưa đưa ra được giải pháp giúp bộ cân bằng tải cung cấp tài nguyên một cách hiệu quả và không tốn thời gian quay vòng lặp. Ngoài ra, các thuật toán cũng chưa có khả năng sẵn sàng và đảm bảo độ tin cậy của hệ thống.

Do đó, việc đề xuất *"Nghiên cứu chính sách bền vững nhằm xây dựng thuật toán nâng cao hiệu quả cân bằng tải của điện toán đám mây"* là vô cùng cần thiết.

Thuật toán đề xuất có khả năng chịu lỗi (Fault Tolerance), việc truy cập cũng được phân bổ đồng đều trên các nguồn tài nguyên, thậm chí là trên các Datacenter khi nhu cầu tăng lên một cách nhanh chóng. Hoặc khi một máy chủ gặp sự cố, chức năng cân bằng tải đám mây sẽ chỉ đạo phân phối công việc của máy chủ đó cho các máy chủ còn lại, đẩy thời gian (Uptime) của hệ thống lên cao nhất và cải thiện năng suất hoạt động.

Nhằm nâng cao hiệu quả cân bằng tải trên các dịch vụ điện toán đám mây đã được đề xuất, em xin đưa ra nội dung đề tài nghiên cứu như sau: *"Nghiên cứu chính sách bền vững nhằm xây dựng thuật toán nâng cao hiệu quả cân bằng tải của điện toán đám mây"*.

Đề cương luận văn bao gồm 03 phần: Phần mở đầu, nội dung gồm 04 chương và phần kết luận.

2. Tổng quan về vấn đề nghiên cứu

2.1. Tình hình nghiên cứu trong nước

Trong bài báo [1] của Trần Công Hùng và các cộng sự đăng trên tạp chí Khoa học công nghệ Thông tin và truyền thông số 04(CS.01) 2018 của Học viện Công nghệ Bưu chính viễn thông, nhóm tác giả đã đề xuất một thuật toán cân bằng tải nhằm giảm thời gian đáp ứng trên điện toán đám mây. Ý tưởng chính của bài là

sử dụng thuật toán dự báo ARIMA để dự báo thời gian đáp ứng, từ đó đưa ra cách giải quyết phân phối tài nguyên hiệu quả dựa vào giá trị ngưỡng thời gian. Bài báo đã đưa ra thuật toán, thử nghiệm mô phỏng với mô hình nhỏ và đã đạt được một số kết quả mô phỏng khá tích cực và tiềm năng trong dự báo tương lai gần.

Bài báo [2], [21] của tác giả Nguyễn Thanh Thủy và các cộng sự đăng trên tạp chí "International Journal of Computer Science and Network, Volume 4, Issue 2, April 2015", đã trình bày một cách tiếp cận mới để cải thiện thuật toán ngăn chặn bế tắc. Cụ thể là lên lịch cho các chính sách cung cấp tài nguyên để phân bổ cho các tài nguyên không đồng nhất. Thuật toán ngăn chặn bế tắc có độ phức tạp thời gian chạy là O (min (m, n)). Trong đó, m là số lượng tài nguyên và n là số lượng quy trình. Họ đã đề xuất thuật toán phân bổ nhiều tài nguyên cho các dịch vụ cạnh tranh đang chạy trong các máy ảo trên nền tảng phân tán không đồng nhất. Các thí nghiệm cũng so sánh hiệu suất của phương pháp đề xuất với các công việc liên quan khác.

Luận văn cũng nghiên bài báo [3] của các tác giả Nguyễn Hà Huy Cường, Đặng Hùng Vĩ, Phạm Nguyễn Minh Nhựt và Lê Văn Sơn trong cuốn sách "Những tiến bộ gần đây của công nghệ thông tin và truyền thông năm 2015 trang 285-295". Trong bài viết này, họ đã nghiên cứu việc phân bổ tài nguyên ở cấp cơ sở hạ tầng thay vì nghiên cứu cách ánh xạ tài nguyên vật lý với tài nguyên ảo để sử dụng tài nguyên tốt hơn trong môi trường điện toán đám mây. Nhóm tác giả đề xuất một thuật toán mới để phân bổ tài nguyên cho cơ sở hạ tầng. Sau đó, cơ sở hạ tầng sẽ tự động phân bổ các máy ảo vào một trong số các ứng dụng điện toán đám mây dựa trên phát hiện gián đoạn. Từ đó, thuật toán có thể tiếp cận và sử dụng phương pháp ngưỡng để tối ưu hóa quyết định phân bổ lại tài nguyên. Họ cũng đã triển khai và thực hiện thuật toán mình đã đề xuất bằng cách sử dụng trình mô phỏng CloudSim. Kết quả thí nghiệm cho thấy thuật toán đề xuất của nhóm tác giả có thể nhanh chóng phát hiện bế tắc và sau đó giải quyết các tình huống tương đương các trường hợp trong thực tế.

2.2. Tình hình nghiên cứu trên thế giới

Trong bài báo [4] "Deadlock Avoidance through Efficient Load Balancing to Control Disaster in Cloud Environment" của nhóm tác giả Mahitha. O và Suma. V,

Ấn Độ năm 2013 đã trình bày một kỹ thuật cân bằng tải hiệu quả để kiểm soát thảm họa trên môi trường điện toán đám mây. Thuật toán đề xuất được áp dụng để cân bằng tải. Cụ thể là so sánh hiệu suất hệ thống được thực hiện trong cả hai trường hợp có và không có cân bằng tải. Thuật toán đề xuất được thực hiện bằng cách sử dụng công cụ Cloud Analyst. Kết quả mô phỏng thu được đã mô tả việc chia sẻ và quản lý tài nguyên tốt hơn với thời gian phản hồi tổng thể tối thiểu, thời gian xử lý và thông lượng tốt hơn.

Trong bài báo [5] "Deadlock prediction in linear systems" của tác giả Zbigniew Suraj được đưa ra tại Hội nghị chuyên đề về lý thuyết tính toán năm 1984 đã đề cập đến độ phức tạp tính toán của vấn đề dự đoán gián đoạn. Trong các hệ thống tuyến tính được điều tra. Tác giả đã đề xuất một thuật toán để giải quyết vấn đề này cho các hệ thống tuyến tính. Độ phức tạp của nó là đa thức. Bài báo cũng chứa một thuật toán giải quyết vấn đề tránh bế tắc trong các hệ thống. Độ phức tạp tính toán của thuật toán đó cũng là đa thức.

Trong bài báo ngày 03/9/1990 [6] "DEADLOCK PREDICTION FOR ESCROW TRANSACTIONS" của tác giả PATRICK E. O'NEIL đã đề cập đến phương thức giao dịch ký quỹ cho phép các giao dịch tồn tại lâu dài và cập nhật hồ sơ mà không chặn quyền truy cập đồng thời của các giao dịch khác để ghi lại các bản ghi đã sửa đổi. Tình huống có thể xảy ra trong hệ thống ký quỹ phải kể đến yêu cầu cập nhật hiện không thể được cấp. Nhưng nếu giao dịch yêu cầu được phép chờ đợi, tình huống kết quả sẽ an toàn theo thuật toán của nhân viên ngân hàng. Nói cách khác, tức là một lịch trình được cấp yêu cầu tồn tại và theo sau đó, tất cả các yêu cầu hiện tại cũng có thể được cấp. Bất kỳ thuật toán nào để tìm lịch trình như vậy phải dự đoán những bế tắc tiềm ẩn và tác giả đã chỉ ra rằng vấn đề này là NP-eomplete - sử dụng một biến thể của bằng chứng bằng vàng. Bằng chứng rút ra được một sự tương đồng giữa dự đoán bế tắc giao dịch và thuật toán của nhân viên ngân hàng. Do đó, nó hình thành được các lợi ích độc lập. Trong kết luận của Mục 5, tác giả cũng chỉ ra được một số phân nhánh hiệu suất tiêu cực cơ bản của việc tránh phát hiện bế tắc vốn có trong cách tiếp cận tiêu chuẩn thuật toán của nhân viên ngân hàng. Điều này giúp mở ra một hướng mới trên con đường nghiên cứu thuật toán sau này.

Số đặc biệt của Tạp chí Ứng dụng Máy tính Quốc tế (0975 - 8887) về Công nghệ Điện toán và Truyền thông Tiên tiến cho Ứng dụng HPC - ACCTHPCA, tháng 6 năm 2012 đã đăng bài báo [7] "Enhanced Load Balancing Approach to Avoid Deadlocks in Cloud". Bài báo đề cập đến công nghệ tiên tiến tập trung vào xử lý dữ liệu để đối phó với lượng dữ liệu khổng lồ. Thời điểm này, điện toán đám mây là một công nghệ mới nổi, cho phép người ta hoàn thành mục tiêu nói trên, dẫn đến cải thiện hiệu suất kinh doanh. Nó bao gồm người dùng yêu cầu dịch vụ của các ứng dụng đa dạng từ các máy chủ ảo phân tán khác nhau. Đám mây nên cung cấp tài nguyên theo yêu cầu cho khách hàng của mình với tính khả dụng và khả năng mở rộng cao cũng như giảm thiểu được tối đa chi phí. Cân bằng tải là một trong những yếu tố thiết yếu để nâng cao hiệu suất làm việc của nhà cung cấp dịch vụ đám mây.

Năm 2015, một công bố quốc tế của nhóm tác giả Ha Huy Cuong Nguyen [3] "Detection and Avoidance Deadlock for Resource Allocation in Heterogeneous Distributed Platforms" đã nghiên cứu về deadlock trên nền tảng không đồng nhất chính là cloud. Trong nền tảng dịch vụ này, xử lý tác vụ là tổng hợp của các bộ xử lý phân tán trên mạng hoặc tính toán song song trên hệ thống grid. Chính trên hệ thống này, để đạt được hiệu quả phân bổ tài nguyên, nhóm tác giả đã chú trọng vào việc nâng cao cách phát hiện và dự đoán Deadlock. Từ đó, đưa ra thuật toán tránh deadlock cũng như lên kế hoạch và chính sách phân bổ tài nguyên trên mạng có sự không đồng nhất về hạ tầng. Trong nghiên cứu này, đã sử dụng ảo hóa thông qua các máy ảo để thực hiện thực nghiệm các thuật toán đề xuất.

Trên tạp chí Innovative Technology and Interdisciplinary Sciences năm 2018 [8], nhóm tác giả Ha Huy Cuong Nguyen, cũng đã công bố quốc tế "Avoid Deadlock Resource Allocation (ADRA) Model V VM-out-of-N PM". Trong bài báo này, các tác giả đã chú trọng hơn về clou, và mô hình hoạt động của cloud. Từ đó đề xuất ra thuật toán ADRA và xây dựng mô hình V (VM-out-of-N), tức là các máy ảo đã hết khả năng xử lý. Ở bài nghiên cứu này, các tác giả đã giải thích rõ sự phát triển của cloud trên nền tảng grid, đặc trưng nhất là sự đa dạng người dùng thì vô hạn nhưng tài nguyên thì có hạn. Deadlock xảy ra trên cloud ở mức khá lớn, tức là deadlock trên cloud lớn hơn tất cả deadlock trước giờ vốn có. Trong bài viết này,

một ý tưởng mới được phát triển dựa trên vùng không gian trống trên cloud nhằm tránh deadlock, thông qua việc biết các tài nguyên trống từ các người dùng đã được phục vụ. Thuật toán mới đề xuất là phân bổ các tài nguyên đến các dịch vụ đang được phục vụ trong các máy ảo trên môi trường cloud không đồng nhất. Trong nghiên cứu này, mô phỏng được thực nghiệm trên CloudSim và kết quả thu được là tương đối tốt.

* Nhận xét đánh giá: Nhìn chung, đã có khá nhiều thuật toán cân bằng tải trên cloud. Các kết quả nghiên cứu khá đa dạng và tiềm năng. Tuy nhiên, việc nghiên cứu này còn khá ít. Cụ thể là các nghiên cứu và cải tiến hệ thống cân bằng tải. Vì vậy, việc nghiên cứu về chính sách bền vững nhằm xây dựng thuật toán nâng cao hiệu quả cân bằng tải trên cloud còn rất nhiều hướng nghiên cứu phát triển và cách tiếp cận khác nhau. Không ngoại lệ, đề tài này hoàn toàn có tiềm năng và độ khả thi cao. Thế nên, phương pháp này chính là tiền đề vững chắc và là cơ sở thực tiễn cho đề tài.

3. Mục đích nghiên cứu

Mục tiêu chính: Nghiên cứu chính sách bền vững nhằm xây dựng thuật toán nâng cao hiệu quả cân bằng tải của điện toán đám mây.

Từ mục tiêu chính trên, luận văn dự kiến các kết quả sẽ đạt được như sau:

- Tìm hiểu tổng quan về điện toán đám mây.

- Tìm hiểu về các thuật toán trên điện toán đám mây.

- Tìm hiểu về các thuật toán cân bằng tải trên điện toán đám mây.

- Tìm hiểu khả năng xảy ra quá tải, tài nguyên phân bổ không đồng đều, máy chủ quá tải và ngưng hoạt động.

- Nghiên cứu kỹ về chính sách bền vững và tính bền vững trên môi trường cloud.

- Đề xuất thuật toán có thể sử dụng tài nguyên bền vững hơn, tiết kiệm năng lượng hơn.

- Trên cơ sở lý thuyết đã nghiên cứu, luận văn đề xuất thuật toán nâng cao hiệu quả cân bằng tải của điện toán đám mây. Thêm vào đó, mô phỏng và thực nghiệm thuật toán đã đề xuất.

4. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu

- Đối tượng nghiên cứu chính là thuật toán nâng cao hiệu quả cân bằng tải trên điện toán đám mây.

- Nghiên cứu các thuật toán cân bằng tải hiện đang sử dụng. Phạm vi nghiên cứu trong cloud:

- Phạm vi nghiên cứu

- Xây dựng mô hình mô phỏng đám mây ở mức độ nhỏ: khoảng 10 – 30 máy ảo.

- Độ phức tạp trên mỗi máy ảo chỉ ở mức độ thấp: dưới 10 ứng dụng chạy trên trên các máy ảo.

- Yêu cầu (Request) gửi về máy chủ cũng đơn giản, đánh giá chính xác phải là hành động của người dùng cloud.

5. Phương pháp nghiên cứu

Phương pháp luận: Dựa trên cơ sở là các lý thuyết về điện toán đám mây, các thuật toán cân bằng tải trên cloud.

Phương pháp đánh giá dựa trên cơ sở toán học: Dựa trên nghiên cứu các cơ sở lý thuyết về điện toán đám mây và khả năng bị tắc nghẽn trên đám mây. Đề xuất ra thuật toán để nâng cao hiệu quả cân bằng tải trên đám mây. Chứng minh thuật toán và đánh giá hiệu quả của thuật toán.

Phương pháp đánh giá bằng mô phỏng thực nghiệm: Xây dựng mô hình mô phỏng và thực nghiệm thuật toán đã đề xuất.

CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN VỀ HỆ THỐNG CÂN BẰNG TẢI CỦA ĐIỆN TOÁN ĐÁM MÂY

1.1. Tổng quan về điện toán đám mây

Lịch sử của điện toán đám mây bắt đầu từ năm 1983, khi Sun Microsystems đề xuất rằng "web là máy tính". Trong tháng 3 năm 2006, Amazon giới thiệu dịch vụ đám mây điện toán đàn hồi. Vào tháng 8 năm 2006, Eric Schmidt – Giám đốc điều hành của Google, lần đầu tiên đề xuất khái niệm "Điện toán đám mây" tại hội nghị công cụ tìm kiếm. Năm 2009, Nair M K. Và Gopalakrishnan V. Đã phát triển một khung hệ thống, sử dụng các dịch vụ web như SaaS và môi trường web để hiện thực hóa PaaS, thúc đẩy hiệu quả sự phát triển của điện toán đám mây. Takahiro Miyamoto và nhóm của ông đã nhận ra chức năng mạng của điện toán đám mây vào năm 2009, đặt nền tảng vững chắc cho sự phát triển của điện toán đám mây. Kể từ đó, điện toán đám mây đã bước vào thời kỳ phát triển nhanh chóng. Điện toán đám mây được phát triển từ điện toán song song là điện toán phân tán và điện toán lưới. Như trong Hình 1, nó là một mô hình điện toán kinh doanh mới. Hiện tại, vẫn chưa có định nghĩa thống nhất về điện toán đám mây. Wikipedia định nghĩa điện toán đám mây là một phương thức tính toán mới dựa trên Internet, cung cấp tính toán theo yêu cầu cho người dùng cá nhân và doanh nghiệp thông qua các dịch vụ không đồng nhất và tự trị trên Internet. Eric Schmidt – Giám đốc điều hành của Google, cho rằng điện toán đám mây về cơ bản là một mô hình cung cấp dịch vụ, ảo hóa tài nguyên máy tính, tài nguyên lưu trữ và tài nguyên mạng. Nó bao gồm một số lượng lớn máy chủ, tạo thành một nhóm tài nguyên ảo với tài nguyên điện toán, lưu trữ và mạng, quản lý và lên lịch thông qua một nền tảng điện toán đám mây thống nhất.

Điện toán đám mây (cloud computing): hay còn gọi là điện toán máy chủ ảo, nơi các tính toán được "định hướng dịch vụ" và phát triển dựa vào Internet. Cụ thể hơn, trong mô hình điện toán đám mây, tất cả các tài nguyên, thông tin cũng như software đều được chia sẻ và cung cấp cho các máy tính, thiết bị, người dùng dưới dạng dịch vụ trên nền tảng một hạ tầng mạng công cộng (thường là mạng Internet). Các user thường sử dụng các dịch vụ như cơ sở dữ liệu, website, lưu trữ,… Trong mô hình cloud computing, không cần quan tâm đến vị trí địa lý cũng như các thông tin khác của hệ thống mạng đám mây – "điện toán đám mây trong suốt đối với

người dùng". Người dùng cuối truy cập và sử dụng các ứng dụng đám mây thông qua các ứng dụng như trình duyệt web, các ứng dụng mobile hoặc máy tính cá nhân thông thường. Hiệu năng sử dụng phía người dùng cuối được cải thiện khi các phần mềm chuyên dụng hay các cơ sở dữ liệu được lưu trữ và cài đặt trên hệ thống máy chủ ảo trong môi trường điện toán đám mây trên nền của "data center". "Data center" là thuật ngữ chỉ khu vực chứa server và các thiết bị lưu trữ, bao gồm nguồn điện và các thiết bị khác như rack, cables… với khả năng sẵn sàng và độ ổn định cao. Ngoài ra, nó còn bao gồm các tiêu chí khác như: tính module hóa cao, khả năng mở rộng dễ dàng, nguồn và làm mát, hỗ trợ hợp nhất server và lưu trữ mật độ cao.

Có 3 mô hình triển khai điện toán đám mây chính là public (công cộng), private (riêng) và hybrid ("lai" giữa đám mây công cộng và riêng). Đám mây công cộng là mô hình đám mây mà trên đó, các nhà cung cấp đám mây cung cấp các dịch vụ như tài nguyên, platform hay các ứng dụng lưu trữ trên đám mây và public ra bên ngoài. Các dịch vụ trên public cloud có thể miễn phí hoặc có phí. Đám mây riêng thì các dịch vụ được cung cấp nội bộ và thường là các dịch vụ kinh doanh, mục đích nhắm đến là cung cấp dịch vụ cho một nhóm người và đứng đằng sau firewall. Đám mây "lai" là môi trường đám mây mà kết hợp cung cấp các dịch vụ công cộng và riêng. Ngoài ra còn có "community cloud" là đám mây giữa các nhà cung cấp dịch vụ đám mây. Về mô hình cung cấp dịch vụ có 3 loại chính là IaaS – cung cấp hạ tầng như một service, PaaS – cung cấp Platform như một service và SaaS – cung cấp software như một service.

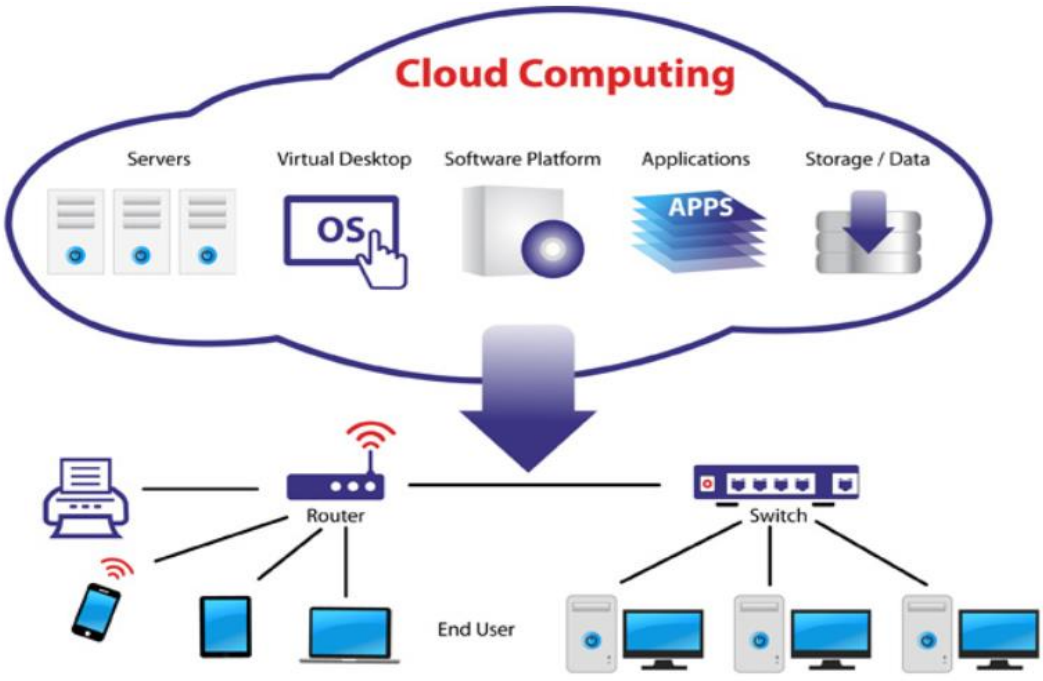
Theo các loại hình dịch vụ, điện toán đám mây có thể được chia thành ba loại sau:

- IaaS, hoặc cơ sở hạ tầng như một dịch vụ, cho phép người dùng truy cập trực tiếp vào tài nguyên lưu trữ, tài nguyên mạng và tài nguyên máy tính bên dưới. IaaS sử dụng công nghệ ảo hóa để ảo hóa và đóng gói tài nguyên máy tính, tài nguyên lưu trữ và tài nguyên mạng của máy chủ, đồng thời cung cấp các tài nguyên này dưới dạng API. Khi cần sử dụng các tài nguyên này, người dùng không cần mua các thiết bị phần cứng như máy chủ mà chỉ cần mua các tài nguyên này từ các nhà sản xuất cung cấp dịch vụ IaaS. Nền tảng điện toán đám mây IaaS cung cấp quản lý và lập kế hoạch của các tài nguyên

này. Ví dụ điển hình bao gồm Đám mây tính toán đàn hồi (EC2) và Dịch vụ lưu trữ đơn giản (S3) của Amazon.

- PaaS, hoặc nền tảng làm nền tảng dịch vụ, cung cấp nền tảng và môi trường cho hoạt động kinh doanh phần mềm. PaaS cung cấp giải pháp cho các công ty không thể hoặc không muốn xây dựng môi trường vận hành phần mềm. PaaS cung cấp môi trường hoạt động và hệ điều hành cho các doanh nghiệp khác nhau. "Máy chủ ảo" thuộc danh mục dịch vụ PaaS. Chỉ có mã nguồn cần được tải lên địa chỉ của "máy chủ ảo". "Máy chủ ảo" sẽ chạy mã và tạo một trang web theo mã. Điển hình là 2 ứng dụng GoogleAppEngine của Google và MicrosoftWindowsAzure của Microsoft.

Theo các phương pháp triển khai khác nhau, điện toán đám mây có thể được chia thành đám mây riêng, đám mây công cộng và đám mây lai. Đám mây riêng là cơ sở hạ tầng đám mây do một tổ chức sở hữu hoặc thuê, có thể được đặt tại địa phương hoặc ở một nơi khác. Đám mây công cộng là cơ sở hạ tầng đám mây thuộc sở hữu của một tổ chức điều hành cung cấp dịch vụ điện toán đám mây. Tổ chức này bán các dịch vụ điện toán đám mây cho công chúng hoặc một số lượng lớn các nhóm doanh nghiệp vừa và nhỏ. Đám mây kết hợp bao gồm đám mây riêng và đám mây công cộng. Mỗi đám mây vẫn là một thực thể độc lập, nhưng được kết hợp với công nghệ tiêu chuẩn hoặc độc quyền để làm dữ liệu và ứng dụng di động.



Hình 1.1. Mô hình điện toán đám mây [1]

Điện toán đám mây là một xu hướng công nghệ nổi bật trên thế giới trong những năm gần đây và đã có những bước phát triển nhảy vọt cả về chất lượng, quy mô cung cấp cũng như loại hình dịch vụ. Bằng chứng với một loạt các nhà cung cấp lớn, nổi tiếng như Google, Amazon, Microsoft,...

Điện toán đám mây là mô hình điện toán mà mọi giải pháp liên quan đến công nghệ thông tin đều được cung cấp dưới dạng các dịch vụ qua mạng Internet. Qua đó giải phóng người dùng khỏi việc đầu tư nhân lực, công nghệ và hạ tầng để triển khai hệ thống. Từ đó, điện toán đám mây giúp tối giản chi phí và thời gian triển khai, tạo điều kiện cho các khách hàng sử dụng nền tảng điện toán đám mây tập trung tối đa được nguồn lực vào công việc chuyên môn. Lợi ích của điện toán đám mây mang lại không chỉ gói gọn trong phạm vi người sử dụng nền tảng điện toán đám mây mà còn từ phía các nhà cung cấp dịch vụ điện toán.

Điện toán đám mây (Cloud Computing) [1], [2] là xu hướng phát triển mạnh nhất hiện nay. Nó kế thừa các mạng lưới và các khái niệm máy tính phân tán trước đây để tích hợp các tài nguyên máy tính, lưu trữ, nền tảng và các dịch vụ khác theo nhu cầu một cách thuận tiện và nhanh chóng. Đồng thời, nó cũng cho phép kết thúc sử dụng dịch vụ, giải phóng tài nguyên dễ dàng cũng như giảm thiểu giao tiếp với các nhà cung cấp. Theo đó, mô hình chính là cho phép sử dụng dịch vụ theo yêu cầu (ondemand service). Cùng với đó là cung cấp khả năng truy cập dịch vụ qua mạng rộng rãi từ máy tính để bàn và máy tính xách tay tới thiết bị di động (broad network access). Cuối cùng, với tài nguyên tính toán động, nó có thể phục vụ nhiều người (resource pooling for multi-tenanci), năng lực tính toán phần mềm dẻo và đáp ứng nhanh theo nhu cầu từ thấp đến cao (rapidelasticity).

Điện toán đám mây được dựa trên công nghệ ảo hóa [3], thông qua các dịch vụ mạng để cung cấp cho người dùng với các nguồn lực cơ bản, nền tảng ứng dụng, phần mềm và các dịch vụ khác. Trong trường hợp IaaS (cơ sở hạ tầng như một dịch vụ), các nhà phát triển cung cấp một môi trường ứng dụng phần mềm hoàn chỉnh bằng cách tập hợp các phần cứng, phần mềm và các thiết bị có liên quan lại với nhau để đáp ứng thỏa thuận chất lượng dịch vụ với người dùng. Công nghệ máy ảo (Virtual Machine) thường được sử dụng trong các trung tâm dữ liệu, máy tính cụm và các ứng dụng khác. Công nghệ này cho phép nhiều hệ điều hành có thể chạy trên

cùng một máy tính và cung cấp các dịch vụ độc lập đáng tin cậy, cải tiến rất nhiều khả năng tái sử dụng các tài nguyên vật lý.

Điện toán đám mây [4] là một hướng nghiên cứu rộng, sẽ đem lại giá trị lớn về các chi phí cho các doanh nghiệp trên toàn thế giới. Điện toán đám mây sẽ giúp giải quyết được việc lưu trữ dữ liệu trên hệ thống nhanh hơn, gọn hơn và nhẹ hơn. Cung cấp các dịch vụ về cơ sở hạ tầng, nền tảng phần mềm và các dịch vụ theo yêu cầu người dùng thông qua Internet.

Điện toán đám mây [5] là một mô hình dịch vụ công nghệ thông tin được kế thừa các mạng lưới trước đây trên thế giới. Nó giúp người dùng tiện lợi hơn trong việc truy cập tài nguyên dữ liệu, lưu trữ đến hệ thống quản lý và xử lý dữ liệu phức tạp của các hệ thống như Google, Facebook… Trên thực tế, người dùng chỉ truy cập vào thiết bị đầu cuối để truy xuất vào các tài nguyên trên điện toán. Còn bên trong hệ thống điện toán sẽ lập lịch xử lý các yêu cầu trên bao gồm xử lý thời gian chờ đến thời gian xử lý tín hiệu và thời gian hoàn thành nhiệm vụ.

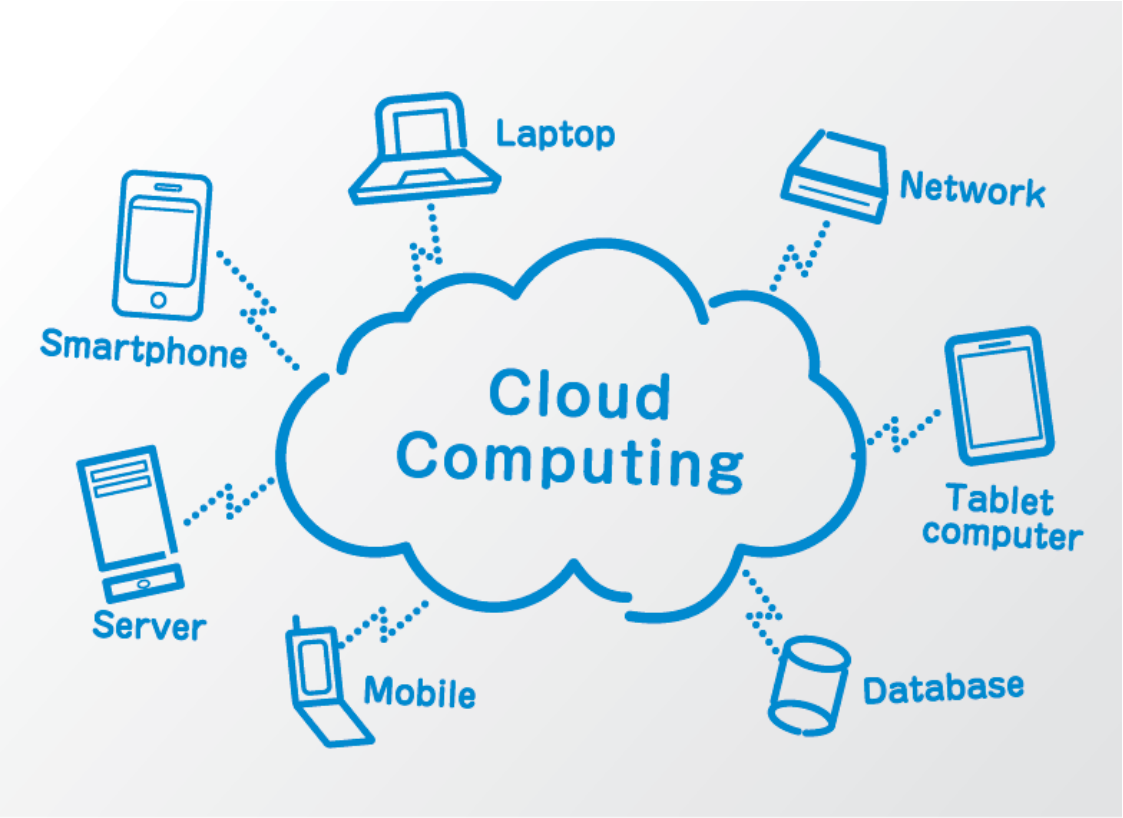
Điện toán đám mây [6], [18], [19] đang chuyển đổi ngành công nghệ thông tin, thay đổi cách thức sử dụng cũng như cung cấp phần cứng và phần mềm. Làm cho việc sử dụng các tài nguyên máy tính theo yêu cầu như băng thông, lưu trữ hoặc các ứng dụng phần mềm và điện toán có sẵn. Nó che giấu sự phức tạp của cơ sở hạ tầng cơ bản, cho phép người dùng cuối tập trung vào sản phẩm của chính họ mà không cần nhiều khoản đầu tư vào phần cứng. Theo hợp đồng dịch vụ đã được thiết lập giữa nhà cung cấp điện toán và khách hàng, các ràng buộc về chất lượng dịch vụ (QoS) nhất định được xác định thông qua các thỏa thuận theo mức dịch vụ (SLA). Tuân thủ với các SLA này, nhà cung cấp đảm bảo cung cấp một chất lượng nhất định cho dịch vụ đã thỏa thuận. Việc sử dụng các máy ảo cho phép sử dụng tốt hơn các tài nguyên phần cứng hiện tại trong khi vẫn duy trì QoS theo yêu cầu. Để tránh sự xuống cấp của hiệu suất, máy ảo được di chuyển từ quá tải đến các máy không sử dụng được. Vì vậy, các thuật toán phát hiện là cần thiết để chủ động phân loại quá tải và không quá tải. Các thuật toán chủ động xác định một kế hoạch tối ưu cho việc di chuyển và phân bổ các máy ảo trong thời gian chạy.

Là một mô hình tính toán mới, [7] được phát triển sau khi công nghệ phân phối máy tính, điện toán lưới, lưu trữ mạng, công nghệ cụm và tính toán song song ra đời. Do tính đa dạng ứng dụng trong nền điện toán đám mây và sự không đồng

nhất của các nút nguồn máy chủ, một số máy tính bị quá tải, còn một số khác thì rất nhẹ trong sự tăng trưởng nhanh chóng của lưu lượng mạng truy cập và dữ liệu. Do đó, chúng ta cần chiến lược cân bằng tải để điều chỉnh tải máy chủ, giảm chi phí truyền thông và cải thiện việc sử dụng tài nguyên. Sự xuất hiện của dữ liệu lớn và phát triển của điện toán đám mây sẽ làm tăng chi phí truyền thông giữa các máy chủ trong quá trình di chuyển và tính toán của một số máy ảo giao dịch với dữ liệu trong bài toán công việc. Ngoài ra, nó còn làm giảm tỷ lệ sử dụng tài nguyên hệ thống.

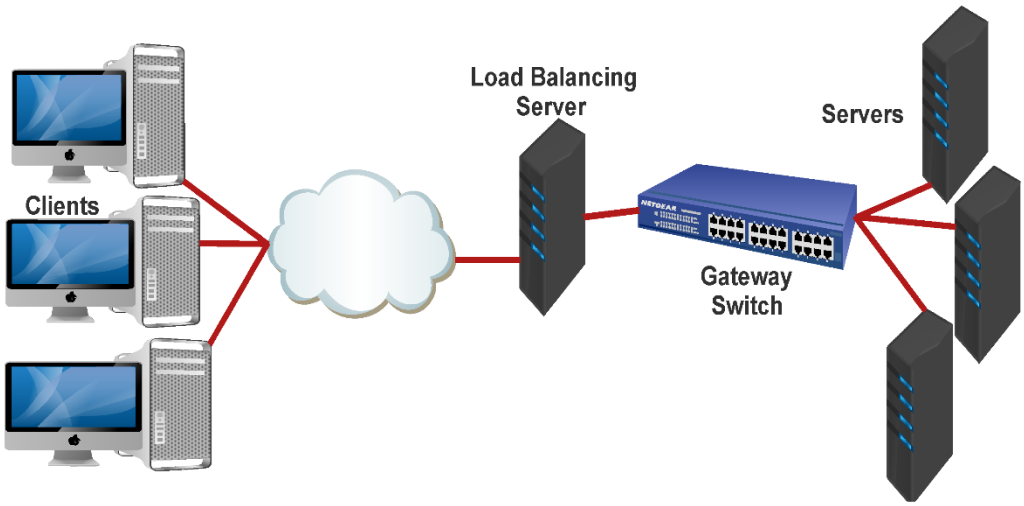
Điện toán đám mây là một kiểu [8] mẫu mới và tiến hóa đáng chú ý nhất trong tính toán. Cơ chế cân bằng tải được chia thành các nguồn lực và cung cấp các nguồn lực cùng với nhiệm vụ lập kế hoạch giữa các hệ thống phân phối. Cân bằng tải truyền thống phải đối mặt với một số vấn đề khác nhau giữa các giai đoạn cung cấp tài nguyên trong môi trường đám mây. Nó cũng có tác động to lớn trong các hệ thống đám mây về hiệu suất và vấn đề đo lường do sự tham gia của các thông số cân bằng tải khác nhau cũng như bản chất của môi trường đám mây.

Trong thế giới ngày nay [9], điện toán đám mây là một cách để giữ phần cứng cũng như phần mềm ở một nơi và sử dụng nó từ khắp nơi trên thế giới. Nó đã làm cho yêu cầu về phần cứng linh hoạt hơn nhiều. Do đó, mọi người có cơ hội sử dụng nhiều tài nguyên khi cần và chỉ phải chi trả cho khoảng thời gian họ đã sử dụng nguồn dung lượng cụ thể. Đó được gọi là dịch vụ trả tiền cho mỗi lần sử dụng, làm cho ngành công nghiệp công nghệ thông tin hướng gần hơn đến việc kinh doanh điện toán đám mây. Giống như một CPU với nhiều lỗi, những doanh nghiệp sở hữu một cụm của các CPU / Máy vật lý được gọi là đám mây. Các cụm có một số lượng hữu hạn không gian và bộ nhớ.



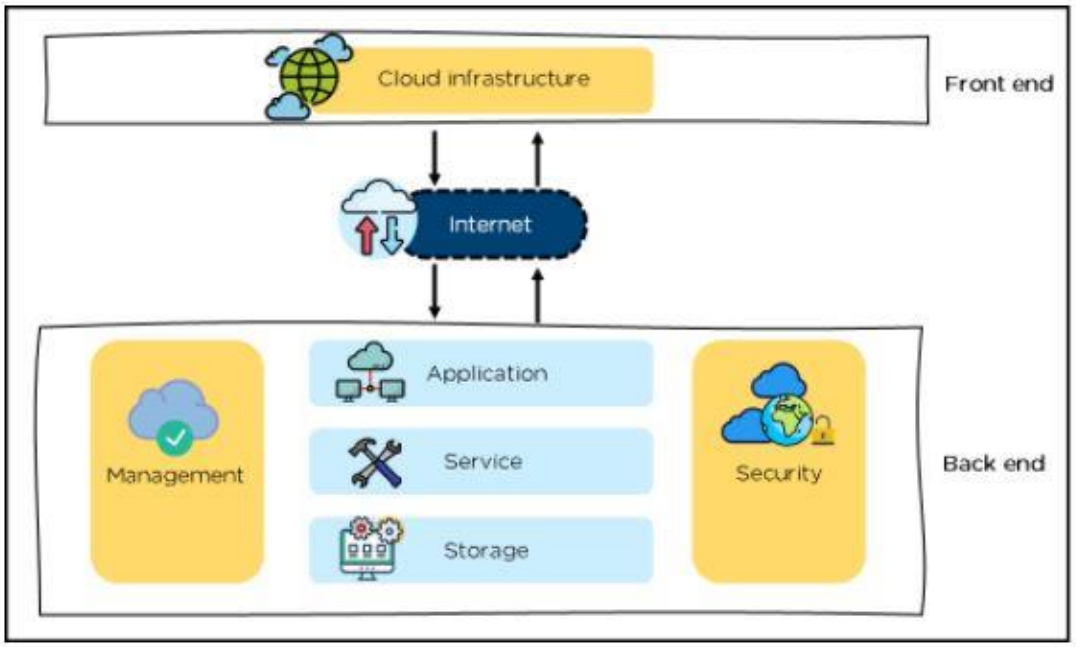
Hình 1.2. Cung cấp tài nguyên đám mây [4]

Vì vậy, khách hàng phải trả tiền để có không gian và bộ nhớ từ cụm được phân bổ trong một khoảng thời gian nhất định. Khi người dùng đòi hỏi các nguồn lực bao gồm bộ nhớ, không gian và băng thông, được thực hiện bởi các công ty thông qua việc phân bổ các máy chủ đến nền tảng nhu cầu khách hàng. Cung cấp tài nguyên trên đám mây là quá trình cung cấp không gian bộ nhớ ảo từ các nguồn lực bằng cách tổng hợp máy vật lý (PM) được gọi là máy ảo (VM). Bộ cân bằng tải quản lý ghép kênh các tài nguyên theo yêu cầu.



Hình 1.3. Cân bằng tải trong điện toán đám mây [5]

Các biện pháp cân bằng trước đây có hiệu quả trong việc cải thiện thời gian phản hồi và thời gian phục vụ của đám mây, nhưng không cung cấp đúng chất lượng dịch vụ. Các QoS có thể được cung cấp hiệu quả bằng cách thêm tham số của nó vào tham số cân bằng tải. Không những xem xét băng thông như tham số mà còn phải đối mặt với các vấn đề suy giảm cùng những vấn đề khác sẽ làm cho ngưỡng giá trị chính xác hơn. Do đó, QoS cũng được coi là có hiệu quả. Vì vậy, việc giảm thiểu yêu cầu được cấp phát cho các máy vật lý với đúng khả năng cung cấp của các máy ảo và duy trì trạng thái ổn định trong suốt thời gian cung cấp dịch vụ là vô cùng cần thiết.



Hình 1.4. Kiến trúc của điện toán đám mây [7]

Khi sử dụng tính toán tự động, tránh chi phí chung là một vấn đề lớn và cần phải giải quyết bằng cách đặt ra các nguồn lực thông qua thuật toán quy mô. Sau đó, vấn đề cuối cùng là phải giữ tải được cân bằng ngay cả trong thời gian phát triển. Điều này được thực hiện bằng cách sử dụng các thuật toán khác nhau.

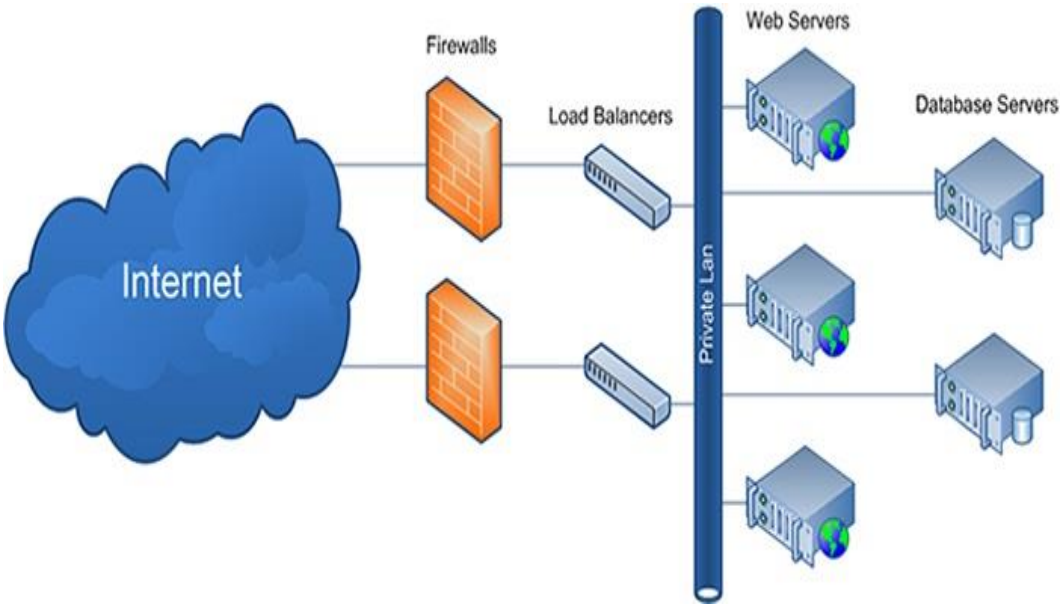
1.2. Tổng quan về cân bằng tải trong điện toán đám mây

1.2.1. Giới thiệu về cân bằng tải

Ngày nay, ngành công nghiệp CNTT đang phát triển mỗi ngày, nhu cầu về tài nguyên lưu trữ và tính toán cũng vậy. Một lượng lớn dữ liệu được tạo và trao đổi qua mạng, điều này đòi hỏi nhu cầu về tài nguyên máy tính ngày càng nhiều. Cloud đã giúp các doanh nghiệp tận dụng lợi ích của tài nguyên điện toán được chia sẻ trên môi trường ảo hóa. Rất nhiều doanh nghiệp đã sử dụng các dịch vụ dựa trên đám mây, hoặc ở dạng này, hoặc ở dạng khác. Điều này đưa chúng ta đến khái niệm cân bằng tải trong điện toán đám mây.

Cùng với việc phát triển rộng rãi của Internet, các website hay các ứng dụng trực tuyến hiện đang được rất nhiều người truy cập và sử dụng. Khi lượng truy cập này quá lớn thường xảy ra các vấn đề là hạ tầng mạng và khả năng xử lý của Server sẽ bị tắc nghẽn cục bộ. Vì vậy, Cân Bằng Tải luôn luôn là một trong những tính năng công nghệ rất quan trọng giúp các máy chủ ảo hoạt động đồng bộ và hiệu quả hơn thông qua việc phân phối đồng đều tài nguyên.

Giải pháp cân bằng tải là việc phân bố đồng đều lưu lượng truy cập giữa hai hay nhiều máy chủ có cùng chức năng trong cùng một hệ thống. Bằng cách đó, sẽ giúp cho hệ thống giảm thiểu tối đa tình trạng một máy chủ bị quá tải và ngưng hoạt động. Hoặc khi một máy chủ gặp sự cố, Cân Bằng Tải sẽ chỉ đạo phân phối công việc của máy chủ đó cho các máy chủ còn lại, đẩy thời gian uptime của hệ thống lên cao nhất và cải thiện năng suất hoạt động tổng thể.



Hình 1.5. Mô hình Cân bằng tải trong điện toán đám mây [8]

Cân bằng tải là một trong những chủ đề quan trọng nhất trong môi trường phân tán. Cloud Computing được coi là một trong những nền tảng tốt nhất giúp lưu trữ dữ liệu với chi phí tối thiểu và có thể truy cập mọi lúc qua Internet. Cân bằng tải cho điện toán đám mây đã trở thành một lĩnh vực nghiên cứu vô cùng thú vị và quan trọng. Cân bằng tải nhằm mục đích thỏa mãn người dùng và đảm bảo tỷ lệ sử dụng tài nguyên cao bằng cách phân bổ hợp lý. Có rất nhiều khó khăn trong các kỹ thuật cân bằng tải như bảo mật, khả năng chịu lỗi, v.v... vốn phổ biến trong môi trường điện toán đám mây hiện đại. Nhiều nhà nghiên cứu đã đề xuất một số kỹ thuật và thuật toán để tăng cường tìm ra những phương án tối ưu cho Cân bằng tải.

Phân tán dự đoán quá tải trong cân bằng tải [10], [20] thời gian gần đây đã nổi lên như một giải pháp đầy hứa hẹn. Trong đó gồm chuyển sang cấp độ giám sát tình trạng tắc nghẽn của mỗi con đường và phân tán dòng chảy trực tiếp đến con đường ít tắc nghẽn. Cách tiếp cận này có nhiều lợi thế thực tiễn. Là một lược đồ phân phối, nó có thể mở rộng hơn và có thể đối phó với lưu lượng truy cập nhanh

hơn cách lịch trình tập trung. Là một phương pháp tiếp cận dữ liệu, nó không phụ thuộc vào ngăn xếp mạng của máy chủ lưu trữ và ngay lập tức mang lại lợi ích cho tất cả lưu lượng truy cập khi triển khai. Khả năng hiển thị tắc nghẽn cuối cùng cũng làm cho nó trở nên mạnh mẽ hơn mà không cần cấu hình lại máy điều khiển. Mấu chốt của việc thiết kế một giao thức cân bằng tải tắc nghẽn là chúng ta cần phải biết thông tin về tắc nghẽn thời gian thực từ tất cả các đường đi giữa nguồn dòng chảy và điểm đến. Có một cách tiếp cận đơn giản là sử dụng thông tin định hướng đường đi cuối: Một switch ToR duy trì các chỉ số tắc nghẽn đầu cuối cho tất cả các đường dẫn từ chính nó đến các thiết bị chuyển mạch ToR khác trong mạng. Các chỉ số tắc nghẽn có thể được thu thập bằng các gói dữ liệu. Thông thường, có hàng trăm đường dẫn tồn tại giữa hai ToR thiết bị chuyển mạch và công tắc ToR có thể giao tiếp với hàng trăm các thiết bị chuyển mạch ToR khác. Quan trọng hơn, không thể để việc thu thập thông tin làm tắc nghẽn thời gian thực của tất cả các đường dẫn này, vì sẽ không có đủ dòng chảy đồng thời cùng xảy ra cho tất cả chúng. Trong giai đoạn đầu, chỉ có nguồn và thiết bị chuyển mạch ToR đích tham gia để lựa chọn tốt nhất đường dẫn từ ToR đến tầng tổng hợp. Chuyển đổi nguồn ToR sẽ gửi số liệu tắc nghẽn của nó đến đích ToR, chúng sẽ kết hợp với các chỉ số tắc nghẽn để chọn con đường tốt nhất cho lớp tổng hợp. Trong giai đoạn thứ hai, tập hợp đã chọn sẽ chọn công tắc lõi tốt nhất theo một cách tương tự về tình trạng tắc nghẽn của bước nhảy thứ hai và thứ ba. Con đường quyết định lựa chọn sau đó được duy trì tại ToR và tập hợp thiết bị chuyển mạch. Về cơ bản, hai giai đoạn lựa chọn đường dẫn đã sử dụng thông tin của một phần đường dẫn để tìm ra đường tốt nhất cho dòng chảy. Bằng cách khai thác các tính chất của cấu trúc 3 tầng, lựa chọn đường dẫn hai giai đoạn làm giảm đáng kể các vấn đề phức tạp mà lại không có nhiều hiệu suất. Trên thực tế, đánh giá cho thấy rằng thực hiện lựa chọn đường dẫn trên mỗi cơ sở lưu lượng trong TCP là tốt nhất và không gây ra việc sắp xếp lại gói tin cũng như không gây bất kỳ độ trễ nào.

Cân bằng tải luôn là chủ đề nghiên cứu nóng của các trung tâm dữ liệu đám mây, và mục tiêu của nó là đảm bảo rằng mọi tài nguyên máy tính có thể xử lý các nhiệm vụ một cách hiệu quả và nhanh chóng. Cuối cùng, việc sử dụng nguồn lực được cải thiện. Các nhà nghiên cứu đã đề xuất một loạt cân bằng tĩnh, cân bằng động và chiến lược lập kế hoạch cân bằng tải. Ngoài ra, cũng có một số nghiên cứu

sử dụng công nghệ di chuyển trực tiếp của máy ảo để đáp ứng các yêu cầu đám mây, nhiệm vụ của trung tâm dữ liệu là yêu cầu hiệu suất và giới hạn tải. Các chiến lược cân bằng tải hiện được chia thành hai loại: cân bằng tải tĩnh và cân bằng tải năng động. Thuật toán lập lịch cân bằng tải tĩnh thường bao gồm Round Robin và Rounded Robin Weighted. Các thuật toán tĩnh chỉ sử dụng một số thông tin tĩnh mà không thể phản ánh tải động. Hiện nay, hầu hết các nền tảng mã nguồn mở, kể cả IaaS, đã sử dụng các thuật toán tĩnh để tiến hành lập kế hoạch tài nguyên. Lợi thế của thuật toán lập kế hoạch cân bằng tải tĩnh là nó rất đơn giản và dễ sử dụng. Nhưng trong các trung tâm dữ liệu đám mây quy mô lớn có tính không đồng nhất của tài nguyên và nhu cầu người sử dụng là không nhất quán nên hiệu quả cân bằng tải tĩnh không được lý tưởng. Cân bằng tải động (DLB), nó chủ yếu được sử dụng trong lĩnh vực phân phối máy tính song song. Mục tiêu chính của nó là làm thế nào để phân phối tải hợp lý hơn giữa nhiều máy chủ để tránh một số hiện tượng như một số nút máy tính bị quá tải và một số nút có tải nhẹ, từ đó cải thiện toàn bộ hiệu suất của hệ thống. Chi phí truyền thông bổ sung được tạo ra trong quá trình DLB sẽ làm suy giảm hiệu năng hệ thống của cân bằng tải động. Vì vậy, làm thế nào để giảm truyền gói tin trên cao nhất giữa các nút trong quá trình DLB trở thành một vấn đề quan trọng và có ảnh hưởng đến hiệu suất của DLB. Tuy nhiên, một số thuật toán ở trên không thể đáp ứng được sự lựa chọn và bản chất của cơ cấu cân bằng tải tối ưu cùng một lúc. Do đó, những cách phân phối tiếp cận thường có được sự tối ưu cục bộ của các giải pháp. Và trong một số trường hợp đặc biệt, hiệu quả của việc giải quyết vấn đề phân phối tải không phải là lý tưởng. Thế nên, nó có thể đảm bảo cân bằng tải và sử dụng hiệu quả tài nguyên vật lý của toàn bộ cụm. Song, cân bằng tải lại là vấn đề và chi phí chung của đám mây trong các trung tâm dữ liệu không được xem xét. Nó chỉ tập trung vào quản lý máy ảo để tăng cường quản lý các trung tâm dữ liệu đám mây và nâng cao hiệu quả hoạt động của các trung tâm dữ liệu điện toán đám mây.

Cân bằng tải [11] có thể được chia thành 2 loại:

- Cân bằng tải cục bộ
- Tải toàn cầu

Cân bằng tải cục bộ được sử dụng để cân bằng dự báo tải trong một trung tâm. Nó phân phối yêu cầu từ phía máy khách sang cho máy chủ để đáp ứng nhu

cầu. Thứ hai là loại cân bằng tải toàn cục. Nó quản lý và kiểm soát yêu cầu từ phía khách hàng tự động đến máy chủ qua nhiều trung tâm dữ liệu. Ngoài ra, nó cũng xử lý lưu lượng trên cả hai mặt gói truyền tải. Xử lý cân bằng tải toàn cầu cho sự phức tạp nhưng đồng thời điều này cũng rất hữu ích cho truyền tải gói tin trên trung tâm dữ liệu mạng. Tính khả dụng đảm bảo rằng, trong trường hợp thất bại, hệ thống vẫn tiếp tục hoạt động được như mong đợi.

1.2.2. Mục đích cân bằng tải

Tăng khả năng đáp ứng, tránh tình trạng quá tải trên máy chủ, đảm bảo tính linh hoạt và mở rộng cho hệ thống.

Tăng độ tin cậy và khả năng dự phòng cho hệ thống: Sử dụng Cân bằng tải giúp tăng tính HA (High Availability) cho hệ thống, đồng thời đảm bảo cho người dùng không bị gián đoạn dịch vụ khi xảy ra lỗi sự cố tại một điểm cung cấp dịch vụ.

Tăng tính bảo mật cho hệ thống: Thông thường khi người dùng gửi yêu cầu dịch vụ đến hệ thống, yêu cầu đó sẽ được xử lý trên bộ Cân bằng tải rồi mới chuyển tiếp đến cho các máy chủ bên trong. Quá trình trả lời cho khách hàng cũng thông qua thành phần Cân bằng tải. Chính vì vậy mà người dùng không thể biết chính xác được các máy chủ bên trong cũng như phương pháp phân tải được sử dụng. Bằng cách này có thể ngăn chặn người dùng giao tiếp trực tiếp với các máy chủ, ẩn các thông tin và cấu trúc mạng nội bộ, ngăn ngừa các cuộc tấn công trên mạng hoặc các dịch vụ không liên quan đang hoạt động trên các cổng khác.

1.3. Tổng quan về trí tuệ nhân tạo (AI)

Trí tuệ nhân tạo (AI) [1] đã trở nên rất phổ biến trong thế giới ngày nay. Nó là sự mô phỏng trí thông minh tự nhiên trong máy móc được lập trình để học và bắt chước các hành động của con người. Những máy này có thể học hỏi kinh nghiệm và thực hiện các nhiệm vụ giống như con người. AI có thể thực hiện nhiệm vụ tốt hơn con người. Đặc biệt khi nói đến các công việc lặp đi lặp lại, định hướng chi tiết như phân tích số lượng lớn các văn bản pháp luật để đảm bảo các lĩnh vực liên quan được điền đúng cách, các công cụ AI thường hoàn thành công việc nhanh chóng và tương đối ít lỗi. Điều này đã giúp thúc đẩy sự bùng nổ về hiệu quả và mở ra cơ hội kinh doanh hoàn toàn mới cho một số doanh nghiệp lớn hơn.

1.4. Tổng quan về machine learning

Học máy (Machine Learning / ML) [3] là một ứng dụng của trí tuệ nhân tạo (AI) cung cấp cho các hệ thống khả năng tự động học hỏi và cải thiện từ kinh nghiệm mà không cần được lập trình rõ ràng. Học máy tập trung vào việc phát triển các chương trình máy tính có thể truy cập dữ liệu và sử dụng nó để tự học. Quá trình học tập bắt đầu với các quan sát hoặc dữ liệu, chẳng hạn như ví dụ, kinh nghiệm trực tiếp hoặc hướng dẫn, để tìm kiếm các mẫu trong dữ liệu và đưa ra quyết định tốt hơn trong tương lai dựa trên các ví dụ mà chúng tôi cung cấp. Mục đích chính là cho phép máy tính học tự động mà không cần sự can thiệp hoặc trợ giúp của con người và điều chỉnh các hành động cho phù hợp. Các thuật toán học máy thường được phân loại là có giám sát, không được giám sát và bán giám sát.

- Các thuật toán học máy được giám sát có thể áp dụng những gì đã học trong quá khứ vào dữ liệu mới bằng cách sử dụng các ví dụ được gắn nhãn để dự đoán các sự kiện trong tương lai. Bắt đầu từ việc phân tích một tập dữ liệu đào tạo đã biết, thuật toán học tạo ra một hàm suy luận để đưa ra dự đoán về các giá trị đầu ra. Hệ thống có thể cung cấp các mục tiêu cho bất kỳ đầu vào mới nào sau khi đào tạo đầy đủ. Thuật toán học tập cũng có thể so sánh đầu ra của nó với đầu ra đúng, dự định và tìm ra lỗi để sửa đổi mô hình cho phù hợp.
- Ngược lại, các thuật toán học máy không được giám sát được sử dụng khi thông tin được sử dụng để đào tạo không được phân loại hoặc gắn nhãn. Học không giám sát nghiên cứu cách hệ thống có thể suy ra một chức năng để mô tả một cấu trúc ẩn từ dữ liệu không được gắn nhãn. Hệ thống không tìm ra đầu ra phù hợp, nhưng nó khám phá dữ liệu và có thể rút ra các suy luận từ tập dữ liệu để mô tả cấu trúc ẩn từ dữ liệu không được gắn nhãn.
- Các thuật toán học máy bán giám sát nằm ở đâu đó giữa học tập có giám sát và không giám sát, vì chúng sử dụng cả dữ liệu được gắn nhãn và không được gắn nhãn để đào tạo – thường là một lượng nhỏ dữ liệu được gắn nhãn và một lượng lớn dữ liệu không được gắn nhãn. Các hệ thống sử dụng phương pháp này có thể cải thiện đáng kể độ chính xác của việc học. Thông thường, học bán giám sát được chọn khi dữ liệu được gắn nhãn thu được yêu

cầu các nguồn lực có kỹ năng và liên quan để đào tạo / học hỏi từ đó. Nếu không, việc thu thập dữ liệu không được gắn nhãn thường không yêu cầu thêm tài nguyên.

Sử dụng các thuật toán phân lớp của ML để tiến hành phân lớp người dùng dựa trên các điểm đặc trưng của họ để thực hiện việc cân bằng tải.

1.5. Chính sách bền vững

Các chính sách bền vững là tập hợp con các biện pháp có khả năng đạt được các mục tiêu mong muốn, khả thi về mặt chính trị và hiệu quả về mặt kinh tế. Với phong trào xanh đang diễn ra mạnh mẽ, nhiều công ty đã cùng nhau đưa ra Chính sách phát triển bền vững để cho thấy họ đang thực hiện vai trò của mình như thế nào trong việc biến tính bền vững trở thành sứ mệnh cốt lõi trong tổ chức của họ. Dù bằng cách nào thì việc kết hợp Chính sách Bền vững là một ý tưởng hay và nó không chỉ giúp tập trung vào những gì có thể làm để giúp cho công ty của họ trở nên ‘xanh’ hơn mà còn có thể ghi nhớ một số cách mà công ty đã lãng phí trong quá khứ.

Tính bền vững là một chiến lược kinh doanh nhằm thúc đẩy tăng trưởng dài hạn và lợi nhuận của công ty bằng cách bắt buộc đưa các vấn đề xã hội và môi trường vào mô hình kinh doanh. Hoặc có thể hiểu tính bền vững là khả năng đáp ứng một cách công bằng các nhu cầu quan trọng của con người ở hiện tại mà không làm ảnh hưởng đến khả năng của các thế hệ tương lai trong việc đáp ứng nhu cầu của chính họ bằng cách giữ gìn và bảo vệ các hệ sinh thái và tài nguyên thiên nhiên của khu vực. Khái niệm về tính bền vững mô tả điều kiện trong đó việc sử dụng tài nguyên thiên nhiên của con người, cần thiết để duy trì sự sống, cân bằng với khả năng bổ sung chúng với thiên nhiên.

1.6. Kết luận chương

Hiểu biết được những khái niệm tổng quan về điện toán đám mây. Hiểu biết thuật toán điện toán đám mây giải quyết những vấn đề tắc nghẽn, gói tin mất mát khi truyền dữ liệu qua môi trường điện toán. Mục đích cân bằng tải để làm tăng hiệu năng của hệ thống.

CHƯƠNG 2. CÁC CÔNG TRÌNH LIÊN QUAN

2.1. Giới thiệu chương

Trong chương này, luận văn sẽ giới thiệu các công trình liên quan đến đề tài ở trong và ngoài nước. Nghiên cứu các công trình này sẽ góp phần giúp củng cố hơn phần cơ sở lý thuyết và định hướng nghiên cứu, phát triển cho đề tài.

2.2. Các công trình nghiên cứu tại Việt Nam

Trong bài báo [1] của Trần Công Hùng và các cộng sự đăng trên tạp chí Khoa học công nghệ Thông tin và truyền thông số 04(CS.01) 2018 của Học viện Công nghệ Bưu chính viễn thông, đã đề xuất một thuật toán cân bằng tải nhằm giảm thời gian đáp ứng trên điện toán đám mây. Ý tưởng của bài báo này chính là sử dụng thuật toán dự báo ARIMA để dự báo thời gian đáp ứng. Từ đó, đưa ra cách giải quyết phân phối tài nguyên hiệu quả dựa vào giá trị ngưỡng thời gian. Bài báo đã đưa ra thuật toán cũng như thử nghiệm mô phỏng với mô hình nhỏ và đã đạt được một số kết quả mô phỏng khá tích cực, tiềm năng trong dự báo tương lai gần.

Trong bài báo [2] của tác giả Nguyễn Thanh Thủy và các cộng sự đăng trên tạp chí “International Journal of Computer Science and Network, Volume 4, Issue 2, April 2015”, đã trình bày một cách tiếp cận để cải thiện thuật toán ngăn chặn bế tắc Đồng thời cũng lên lịch cho các chính sách cung cấp tài nguyên để phân bổ tài nguyên không đồng nhất. Thuật toán ngăn chặn bế tắc có độ phức tạp thời gian chạy là O (min (m, n)), trong đó m là số lượng tài nguyên và n là số lượng quy trình. Họ đề xuất thuật toán phân bổ nhiều tài nguyên cho các dịch vụ cạnh tranh đang chạy trong các máy ảo trên nền tảng phân tán không đồng nhất. Các thí nghiệm cũng so sánh hiệu suất của phương pháp đề xuất với các công việc liên quan khác.

2.3. Một số công trình nghiên cứu trên thế giới

Năm 2018, Afrianto và cộng sự [12] đã công bố nghiên cứu “Weighted Round Robin Load Balancer to Enhance Web Server Cluster in OpenFlow Networks”. Nghiên cứu này nhằm mục đích thiết kế và phân tích mô hình cân bằng tải trên mạng OpenFlow và thực hiện thuật toán Round Robin trọng số (WRR). Quá trình phân tích được tiến hành bằng cách đo giá trị các thông số trình bày QoS của máy chủ web. Kết quả cho thấy thuật toán WRR có khả năng cân bằng hệ thống

mạng với phân bổ tài nguyên động. Trọng lượng công việc của mỗi dịch vụ có thể lấy được từ nhu cầu và tài nguyên mạng hiện có. Hiệu suất của bộ cân bằng tải trên mạng OpenFlow tốt hơn 57% so với mạng truyền thống trong kiểm tra thời gian phản hồi.

Năm 2019, Shi và cộng sự [13] đã công bố nghiên cứu "Concury: A Fast and Light weighted Software Load Balancer". Trong nghiên cứu này, một phần mềm trạng thái cân bằng tải mới được giới thiệu, gọi là Concury – giải pháp đầu tiên để giải quyết hai vấn đề: (1) Các trạng thái được lưu trữ như các bản tiêu hóa có khả năng gây ra sự không nhất quán của gói tin do các xung đột của bản phân tích. (2) Mặt phẳng dữ liệu cần cập nhật cho mọi kết nối mới và việc cập nhật thường xuyên sẽ làm ảnh hưởng đến thông lượng cũng như tính nhất quán của gói. Cải tiến quan trọng của Concury là một cách tiếp cận theo thuật toán để lưu trữ và tra cứu các trạng thái mạng lớn. Cải tiến mới với kết nối thường xuyên và gọn gàng về chi phí bộ nhớ, nhất quán theo các thay đổi của mạng và không thường xuyên cập nhật mặt phẳng dữ liệu. Kết quả đánh giá cho thấy thuật toán Concury cung cấp thông lượng gấp 4 lần và tiêu thụ ít bộ nhớ hơn so với các thuật toán cân bằng tải khác. Đồng thời cung cấp khả năng cân bằng tải có trọng số và tự do đánh nhầm, cho cả lưu lượng trung tâm dữ liệu thực và tổng hợp.

Trong thập kỷ gần đây, sự gia tăng các yêu cầu (các ứng dụng đa dạng và phức tạp) cho các dịch vụ đám mây đang làm tăng khối lượng công việc trong môi trường đám mây. Các kỹ thuật định thời kém hiệu quả đang phải đối mặt với những thách thức về tài nguyên khi chúng được sử dụng quá mức và mất cân bằng. Điều đó dẫn đến suy giảm hiệu suất dịch vụ (do sử dụng quá mức) hoặc lãng phí tài nguyên đám mây (do chưa đáp ứng đủ). Ý tưởng cơ bản đằng sau thuật toán định thời là phân phối các nhiệm vụ (đa dạng và phức tạp) giữa các tài nguyên đám mây để tránh được vấn đề mất cân bằng. Thuật toán định thời nên tối ưu hoá các thông số trình bày chính: thời gian phản hồi, thời gian chờ, độ tin cậy, tính khả dụng, mức tiêu thụ năng lượng, chi phí, mức sử dụng tài nguyên,… Thế nên, vào năm 2019, Kumar và cộng sự [14] đã công bố nghiên cứu "A comprehensive survey for scheduling techniques in cloud computing" nhằm cung cấp đánh giá hệ thống cũng như phân loại các kỹ thuật lập lịch đề xuất cùng với những ưu điểm và hạn chế của

chúng. Đây là một cuộc khảo sát có hệ thống và toàn diện, là bước đệm tốt cho các nhà nghiên cứu điện toán đám mây cũng như kỹ thuật định thời trong tương lai.

Cân bằng tải có thể cải thiện các chỉ số Chất lượng dịch vụ (QoS), bao gồm thời gian phản hồi, chi phí, thông lượng, hiệu suất và sử dụng tài nguyên. Vậy nên, Ghomi và cộng sự [15] vào năm 2017 đã công bố bài nghiên cứu "Load-balancing algorithms in cloud computing: A survey" với tài liệu về các thuật toán định thời tác vụ và cân bằng tải. Đồng thời, họ còn trình bày một phân loại mới của các thuật toán. Ví dụ: danh mục cân bằng tải Hadoop MapReduce, danh mục cân bằng tải dựa trên Hiện tượng tự nhiên, danh mục cân bằng tải dựa trên tác nhân, danh mục Cân bằng tải chung, danh mục hướng ứng dụng, danh mục nhận biết mạng và danh mục quy trình làm việc cụ thể. Họ đã cung cấp các đánh giá về từng loại danh mục trên với thông tin chi tiết về việc xác định các vấn đề mở và hướng dẫn cho các nghiên cứu trong tương lai.

Điện toán đám mây đã trở nên phổ biến hơn do các dịch vụ mà nó cung cấp. Trong môi trường điện toán đám mây, cân bằng tải là một vấn đề rất quan trọng. Bởi người dùng và yêu cầu của họ đối với các dịch vụ khác nhau trên nền tảng điện toán đám mây ngày càng tăng, nên cần thiết phải sử dụng hiệu quả tài nguyên trong môi trường đám mây. Thuật toán cân bằng tải hiệu quả là thuật toán phải đảm bảo được việc sử dụng tài nguyên hiệu quả bằng cách cung cấp đầy đủ tài nguyên cho người dùng theo yêu cầu. Để ưu tiên người dùng Cân bằng tải sử dụng định thời, thời gian phản hồi và thời gian chờ là các chỉ số hoạt động của các thuật toán cân bằng tải. Trong bài báo "A Comparative Study of Static and Dynamic Load Balancing Algorithms in Cloud Computing" năm 2017, Deepa và cộng sự đã cung cấp một nghiên cứu so sánh có hệ thống về các thuật toán cân bằng tải hiện có trong điện toán đám mây. [16]

Máy chủ web là cơ sở của hầu hết mọi hoạt động sử dụng Internet. Phải có một máy chủ web trong đường dẫn từ máy khách đến dịch vụ. Với tính năng cân bằng tải của máy chủ, người dùng không còn gặp phải tình trạng mạng ngừng hoạt động, khôi phục thông tin chậm hoặc kết nối bị hủy. Bài báo "An Improved Weighted Least Connection Scheduling Algorithm for Load Balancing in Web Cluster Systems" được công bố năm 2018, Singh và cộng sự [17] đã đề xuất thuật

toán WLC cải tiến để duy trì tải giữa các máy chủ khác nhau bằng cách ngăn các yêu cầu được truyền đến máy chủ thực mới. Khi các yêu cầu web liên tục được chỉ định cho một máy chủ thực mới nhiều hơn số phân bổ liên tục tối đa (C), thuật toán được đề xuất sẽ loại trừ máy chủ thực mới khỏi danh sách định thời máy chủ thực đã được kích hoạt và hủy kích hoạt nó. Cuối cùng sau các vòng phân bổ C-1, máy chủ thực mới được kích hoạt và đưa vào danh sách định thời máy chủ. Thuật toán được đề xuất cân bằng tải giữa các máy chủ thực bằng cách tránh tình trạng quá tải của máy chủ thực mới.

2.4. Tổng kết chương

Trong chương này thông qua việc nghiên cứu tìm hiểu được một số thuật toán và những công trình liên quan đến cân bằng tải trong điện toán đám mây, giúp luận văn hiểu rõ hơn về cân bằng tải trên điện toán đám mây. Từ đó, hiểu được những ưu nhược điểm của các thuật toán cũng như các cách xử lý cân bằng tải, tạo tiền đề và cơ sở vững chắc cho nghiên cứu của đề tài luận văn này.

CHƯƠNG 3. NGHIÊN CỨU CHÍNH SÁCH BỀN VỮNG NHẰM XÂY DỰNG THUẬT TOÁN NÂNG CAO HIỆU QUẢ CÂN BẰNG TẢI CỦA ĐIỆN TOÁN ĐÁM MÂY

3.1. Giới thiệu chung

Hiện tại, các thuật toán trong cân bằng tải [18], [19], [20] đã được nhiều bài báo nêu lên và cải tiến nhằm nâng cao hiệu năng cân bằng tải. Với ý tưởng áp dụng một số thuật toán và kỹ thuật AI nhằm xây dựng một chính sách bền vững, từ đó đưa ra cách xử lý và phân bổ các request vào các tài nguyên hợp lý nhất. Mục đích chính là nâng cao và cải tiến thời gian xử lý/thực hiện nhiệm vụ, thời gian hoàn thành cũng như tài nguyên của máy ảo đã giảm thiểu sự mất cân bằng tải trong môi trường điện toán đám mây. Với mục tiêu trên, chương này sẽ trình bày ý tưởng thuật toán đề xuất nhằm nâng cao khả năng cân bằng tải bộ chính sách bền vững trên mô hình cloud với hệ thống host và các máy ảo.

3.2. Mô hình nghiên cứu

Mô hình nghiên cứu sử dụng thuật toán Cây phân loại hồi quy (CART) nhằm mục đích loại bỏ các task tương ứng với các Request dựa trên chính sách bền vững. Chính sách bền vững được tính toán dựa trên mức độ tiêu thụ năng lượng của task (Power Consumed), mức độ sử dụng CPU (CPU Usages), mức độ sử dụng RAM (RAM Usages) và chi phí (Costing) để thực hiện các task trong cloud thông qua thuật toán K-Means. Sau khi phân loại các job/task theo chính sách bền vững, bộ cân bằng tải sẽ phân bổ các request tương ứng với các task có độ bền vững cao hơn vào những máy ảo/host có năng lực xử lý tốt hơn, tức là mức độ rảnh task cao hơn. Từ đó, phân bổ các request có nhu cầu xử lý cao vào các máy ảo/host có mức độ hoạt động thấp nhất. Với cách tiếp cận này, thuật toán đề xuất sẽ cải thiện thời gian xử lý cân bằng tải trên cloud đồng thời ứng dụng trên môi trường cloud theo thời gian thực. Trong luận văn này tạm đặt tên thuật toán là RCVKA.

Thuật toán được đề xuất như sau:

* Bước 1: Tiếp nhận request

* Bước 2: Áp dụng các chính sách của thuật toán để cân bằng tải

* Bước 3: Dựa vào kết quả thu được tiến hành cân bằng tải

Về mục tiêu:

- Giảm thiểu các rủi ro cho hệ thống máy chủ.
- Giảm thiểu tối đa thời gian sống cho các yêu cầu trong điện toán đám mây.
- Hạn chế tối đa và ngăn chặn sự mất cân bằng tải giữa các máy ảo.

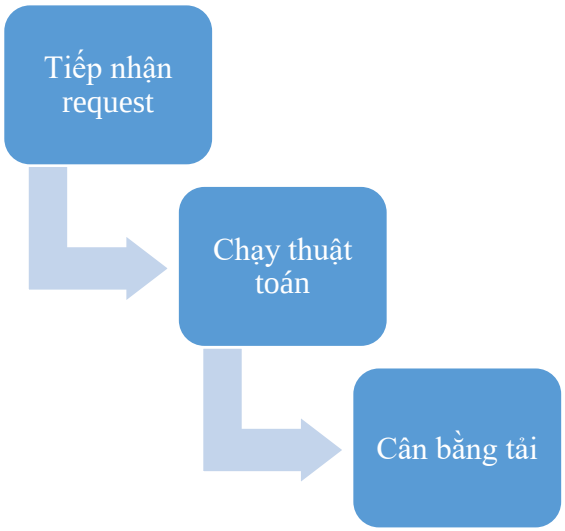
Giả định:

- Bộ cân bằng tải sẽ biết trước được những dịch vụ nào đang chạy trên các máy ảo vào bất kỳ thời điểm nào.
- Luận văn này chỉ tập trung vào dịch vụ Web (Web Service), các máy chủ web sẽ biết trước thời gian xử lý của từng dịch vụ chạy trên web và trên từng máy ảo.
- Nếu hai máy ảo có cấu hình tương đương nhau về RAM, vi xử lý và I/O thì thời gian thực thi của các dịch vụ sẽ không có nhiều khác biệt.

Mô hình nghiên cứu:

Thuật toán được đề xuất như sau:

* Bước 1: Tiếp nhận các request
* Bước 2: Áp dụng các chính sách của thuật toán để thực hiện cân bằng tải
* Bước 3: Dựa vào kết quả thu được tiến hành thực hiện cân bằng tải



Hình 3.1. Mô hình cân bằng tải

3.3. Thuật toán Cây phân loại và hồi quy (Classification and Regression Tree – CART)

Thuật toán CART [21] là một loại thuật toán phân loại cần thiết để xây dựng cây quyết định trên cơ sở "chỉ số tạp chất Gini". Nó là một thuật toán học máy cơ bản và cung cấp nhiều trường hợp sử dụng. Nhà thống kê Leo Breiman đã đặt ra cụm từ này để mô tả các thuật toán Cây quyết định có thể được sử dụng cho các vấn đề phân loại hoặc mô hình dự báo hồi quy.

CART là một từ bao hàm dùng để chỉ các loại cây quyết định sau:

- Cây phân loại: Khi biến mục tiêu liên tục, cây được sử dụng để tìm "lớp" mà biến mục tiêu có nhiều khả năng rơi vào nhất.
- Cây hồi quy: Được sử dụng để dự báo giá trị của một biến liên tục.

Trong cây quyết định, các nút được chia thành các nút con trên cơ sở giá trị ngưỡng của một thuộc tính. Thuật toán CART thực hiện điều đó bằng cách tìm kiếm sự đồng nhất tốt nhất cho các nút con, với sự trợ giúp của tiêu chí Gini Index.

Nút gốc được lấy làm tập huấn luyện và được chia thành hai bằng cách xem xét giá trị ngưỡng và thuộc tính tốt nhất. Hơn nữa, các tập hợp con cũng được chia theo cùng một logic. Điều này tiếp tục cho đến khi tìm thấy bộ con thuần túy cuối cùng trên cây hoặc số lá tối đa có thể có trên cây đang phát triển đó. Điều này còn được gọi là Cắt tỉa cây (Tree Pruning).

Công thức tính toán chỉ số Gini.

$$GI = \Sigma_{i=0}^{c} P_i(1 - P_i) \text{ hoặc } GI = 1 - \Sigma_{i=0}^{c} P_i^2$$

Ở đây c là tổng số lớp và P là xác suất của lớp i.

3.4. Thuật toán K-Means

Thuật toán K-Means là thuật toán rất quan trọng và được sử dụng phổ biến trong kỹ thuật phân cụm, được đề xuất bởi J.B.MacQueen. Thuật toán không giám sát này thường được sử dụng trong khai thác dữ liệu và nhận dạng mẫu. Hướng đến việc giảm thiểu chỉ số hiệu suất cụm, tiêu chí lỗi bình phương và lỗi là nền tảng của thuật toán này. Để tìm kiếm kết quả tối ưu hóa, thuật toán này cố gắng tìm K bộ phận để thỏa mãn một tiêu chí nhất định. Đầu tiên, chọn một số dấu chấm để đại diện cho các tiêu điểm cụm ban đầu (thông thường, chọn K điểm mẫu đầu tiên của thu nhập để đại diện cho tiêu điểm cụm ban đầu); thứ hai, gom các chấm mẫu còn lại về tiêu điểm theo tiêu chí khoảng cách tối thiểu thì ta sẽ được phân loại ban đầu,

phân loại nếu không hợp lý thì sửa đổi (tính lại từng tiêu điểm từng cụm), lặp đi lặp lại cho đến khi có được một sự phân loại.

3.5. Thuật toán đề xuất RCVKA

Dựa vào tham khảo từ tài liệu [20], luận văn này xin đề xuất thuật toán gồm 3 nhóm module chính:

(1) Module tính toán ra các thông số của Request bằng thuật toán K-Means:

Thuật toán K-Means trong Module 1 đóng vai trò tính toán các thông số sử dụng tài nguyên của các task/job dựa trên các thuộc tính của Request sau khi chọn lọc bởi chính sách bền vững. Các thuộc tính của Request bao gồm: Size, Response Length, Max Length,…

Po_{New} = K-Means(Request, Power)

CPU_{New} = K-Means(Request, CPU)

RAM_{New} = K-Means(Request, RAM)

Request = $\{X_1, X_2, \ldots, X_n\}$, với X_i là các thuộc tính của Request khi gửi lên cloud.

Trong đó:

Po_{New} (Power dự đoán Power New): Power ghi nhận được trong quá khứ

CPU_{New} (CPU dự đoán CPU New): CPU ghi nhận được trong quá khứ

RAM_{New} (RAM dự đoán RAM New): RAM ghi nhận được trong quá khứ

Ở đây, ta có thể sử dụng nhóm 3 yếu tố {Po, CPU, RAM} để tổng hợp tính toán hoặc để tính toán riêng biệt từng đại lượng.

(2) Module phân lớp tác vụ theo hành vi người dùng:

Module này sẽ sử dụng thuật toán CART để phân lớp Request đang xét, dựa vào chính sách bền vững trên cloud của các tác vụ.

VM_{select} = RandomForest(Po, CPU, RAM);

Trong đó:

VM_{select}: máy ảo được chọn

Cart: hàm phân lớp từ mô hình Cây quyết định CART đã được xây dựng dựa trên bộ dữ liệu trước đây của các Request

Po: Power dự đoán tính toán từ Module 1

CPU: mức sử dụng CPU dự đoán tính toán từ Module 1

RAM: mức sử dụng RAM dự đoán tính toán từ Module 1

(3) Module phân bổ các dịch vụ (chọn máy ảo)

Nhiệm vụ của Module này là phân bổ các loại Request đến các máy ảo phù hợp. Nếu một Request được gửi tới thì Request đó sẽ được phân loại bởi Module 1 và các VM đang xét bao gồm VM không tải cũng được phân cụm theo Module 2. Ngoài ra, Module 3 có nhiệm vụ phân bổ Request đang xét vào máy ảo đã được tìm thấy từ Module 2, từ đó xử lý và trả về kết quả của Request. Sau đó, lưu kết quả vào lịch sử bộ nhớ các Request gần nhất đã xử lý để làm dữ liệu đầu vào cho quá trình xây dựng mô hình Cây quyết định CART ở Module 2.

Thuật toán RCVKA (**R**equest **C**ART Classification & **V**M **K**-Means Clustering **A**lgorithm)

```
1.  For each Request in CloudRequests
2.      isLocated = false;
3.      SustainablePolicy = {Po, CPU, RAM}new = K-Means (T1,T2,…,Tn); //
        Module 1
4.      Request.SustainablePolicyClass = Cart(SustainablePolicy); //Cart là mô
        hình phân lớp tác vụ
5.      For each VM in VMList
6.          If isFitSituation(Request.SustainablePolicyClass, VM)
7.              AllocateRequestToVM(VM, Request); // Module 3
8.              isLocated = true;
9.          End If
10.     End For
11.     If(!isLocated)
12.         VM = VMList.getSelectedVM(); // Module 2
13.         AllocateRequestToVM(VM, Request);
14.     End If
15. End For
```

Trong đoạn mã giả trên, thuật toán RCVKA đã sử dụng một vòng lặp để lắng nghe tất cả các Request có trong danh sách hàng đợi được gửi lên bộ cân bằng tải (CloudRequests). Thuật toán sẽ tiếp tục phân bổ các Request cho đến khi hết danh sách hàng đợi. Cụ thể, thuật toán sử dụng biến isLocated (kiểu luận lý – boolean) để làm cờ (flag) đánh dấu Request đang xét để xem Request đó đã được phân bổ hay chưa. Khi bắt đầu vòng lặp, biến isLocated được khởi tạo giá trị mặc định là false. Bước tiếp theo, thuật toán sẽ tính toán ra vector SustainablePolicy 3 chiều tương ứng với PowerConsume, CPU Usage và RAM Usage (Priority = {Po, CPU, RAM}) cần dùng để thực hiện Request đang xét. Việc tính toán này dựa trên số liệu của các Request T1, T2,… Tn trước đó, trong đó n là số Request đã được lưu trong lịch sử.

Ti chính là các đại lượng của Request thứ i đã được lưu lại. Ti bao gồm các đại lượng đầu vào: MaxLength, FileSize, OutputSize… và các đại lượng xử lý do cloud đã thực hiện để xử lý Request thứ i bao gồm: PowerConsume, CPU Usage, RAM Usage. N Request trong lịch sử này là dữ liệu xây dựng nên hàm K-Means để dự báo và tính toán ra các đại lượng SustainablePolicy cho Request đang xét. Đại lượng SustainablePolicy cũng chính là dữ liệu đầu vào để chạy CART phân lớp cho Request đang xét, lớp sau khi phân sẽ được gán vào thuộc tính SustainablePolicyClass của Request. Sau khi chạy ra thông số SustainablePolicy cho Request, thuật toán sẽ thực hiện duyệt vòng lặp để duyệt qua các máy ảo hiện đang có trên cloud. Với từng máy ảo tương ứng, thuật toán sẽ xem xét mức độ phù hợp đối với độ ưu tiên của Request đang xét, thông qua hàm isFitSituation(Request.SustainablePolicyClass, VM). Nếu thỏa, RCVKA sẽ phân bổ Request đang xét vào máy ảo: AllocateRequestToVM(VM, Request) đồng thời gán giá trị biến isLoacated = true. Trong trường hợp không tìm được máy ảo nào thích hợp thì vòng lặp sẽ kết thúc. Lúc này, thuật toán chạy hết vòng lặp, biến isLocated vẫn mang giá trị false và Request vẫn chưa được phân bổ. Vì vậy, nhiệm vụ chính của thuật toán RCVKA hiện giờ là phân bổ Request trên vào máy ảo đầu tiên trong danh sách thông qua đoạn lệnh VM = VMList.getSelectedVM(). Việc phân bổ này đảm bảo các Request được dự báo nếu không nằm trong dữ liệu của thuật toán vẫn được phân bổ và xử lý để phục vụ người dùng.

Phương pháp đánh giá thuật toán RCVKA

Kết quả đạt được từ thuật toán mà luận văn đề xuất **RCVKA** đã đáp ứng các mục tiêu được đề ra trước đó. Chẳng hạn như, giới hạn số lượng yêu cầu xếp hàng để phân phối, giảm thiểu thời gian xử lý và phản hồi của đám mây trung tâm, đều tốt hơn so với các thuật toán cũ. Điều này đồng nghĩa với việc hiệu năng của điện toán đám mây được thực hiện bởi thuật toán **RCVKA** đã cải thiện hơn so với bốn thuật toán được đề cập đến trong luận văn là **FCFS**, **MaxMin**, **MinMin** và **Round Robin**.

3.6. Kết luận chương

Chương này đã dựa trên nghiên cứu chính sách bền vững để đưa ra mô hình thuật toán nhằm nâng cao hiệu quả cân bằng tải trên điện toán đám mây. Mục tiêu

ban đầu của luận văn là duy trì sự ổn định trong thời gian xử lý và phản hồi khi hoạt động liên tục trên cloud, thuật toán đề xuất RCVKA đã chứng minh được hiệu năng của mình trong quá trình cân bằng tải. Cụ thể, thuật toán đã làm giảm tối đa thời gian thực hiện xử lý các tác vụ trên cloud cũng như các yêu cầu và rủi ro nhất định. Ngoài ra, RCVKA còn ngăn chặn và hạn chế việc mất cân bằng tải giữa các máy ảo.

CHƯƠNG 4. MÔ PHỎNG CHƯƠNG TRÌNH VÀ ĐÁNH GIÁ KẾT QUẢ

4.1. Giới thiệu chung

Trong chương này, luận văn sẽ trình bày về cài đặt mô phỏng thuật toán RCVKA. Cụ thể, chương này sẽ sử dụng thuật toán RCVKA phân bổ các task tương ứng với các Request dựa trên độ bền vững các task đó đã được tính toán. Mức độ tiêu thụ năng lượng của task (Power Consumed), mức độ sử dụng CPU (CPU Usages), mức độ sử dụng RAM và chi phí (Costing) để thực hiện các task trong cloud đã được tính toán kỹ càng tạo nên độ bền vững của thuật toán. Sau khi phân loại các task theo độ ưu tiê độ bền vững, bộ cân bằng tải sẽ phân bổ các request ứng với các task có độ bền vững cao hơn vào những máy ảo/host có năng lực xử lý tốt hơn hay có mức độ rảnh task cao. Từ đó, phân bổ vào các máy ảo/host có mức độ hoạt động thấp nhất các request có nhu cầu xử lý cao. Với cách tiếp cận này, thuật toán đề xuất RCVKA sẽ cải thiện thời gian xử lý cân bằng tải trên cloud sau đó ứng dụng trên môi trường cloud theo thời gian thực. Các kết quả thu được sau khi tiến hành các bước như trên dùng để phân tích tính hiệu quả của thuật toán đã đề xuất.

4.2. Môi trường mô phỏng thực nghiệm

Dựa vào bộ dữ liệu chứa các Request, luận văn đã sử dụng thuật toán K-Means để phân loại Request dựa trên các đặc trưng của chúng (hành vi người dùng cloud).

Bước 1: Tiếp nhận giá trị input

Bước 2: Phân tích các input để rút trích các đặc trưng của các giá trị input

Bước 3: Dựa vào các đặc trưng trên, chúng ta sử dụng machine learning để phân lớp các giá trị đầu vào.

Bước 4: Dựa vào kết quả phân lớp, ta tiến hành cân bằng tải cho hệ thống.

Dựa vào dữ liệu có thể biết của các Request, ta sử dụng thuật toán CART để phân loại chúng bằng cách tính toán ra bộ SustainablePolicy = {Power, CPU, RAM}. Từ đó, ta biết được cách phân bổ tài nguyên cho các Request vào các máy

ảo đã được phân cụm sẵn trước đó. Kết hợp đánh giá số lần sai và sai số, ta có thể cải thiện thuật toán bằng cách áp dụng máy học (machine learning). Tuy nhiên, máy học sẽ ít được áp dụng để cải thiện thuật toán vì vẫn xảy ra sai số cho phép.

Khi giả lập môi trường cloud, luận văn đã sử dụng ngôn ngữ lập trình JAVA và bộ thư viện CloudSim. Bao gồm 5 đến 15 máy ảo, môi trường cloud là nơi để các Request ngẫu nhiên truy cập đến các dịch vụ trên cloud. Các dịch vụ gồm có: dịch vụ cung cấp máy ảo, dịch vụ cung cấp và đáp ứng người dùng của CloudSim để tiến hành thử nghiệm.

Sau đó, luận văn tiếp tục tiến hành cài đặt thuật toán K-Means và CART trên môi trường mô phỏng được phát triển bởi bộ thư viện WEKA. Cuối cùng là kiểm nghiệm kết quả của thuật toán đề xuất.

Các tham số của mô hình mạng mô phỏng:

Quá trình thực nghiệm mô phỏng thuật toán được cài đặt trên Eclipse IDE hoặc NETBEAN IDE và sử dụng ngôn ngữ JAVA để chạy thử với kết quả được hiển thị dưới dạng console. Môi trường giả lập dùng bộ thư viện mã nguồn mở CloudSim (được cung cấp bởi http://www.cloudbus.org/) kết hợp với bộ thư viện về datamining là WEKA.

Môi trường mô phỏng giả lập gồm các thông số sau:

- 1 Datacenter với thông số sau:

Bảng 4.1. Thông số cấu hình Datacenter

Thông tin Datacenter	***Thông tin Host trong Datacenter***
- Số lượng máy (host) trong datacenter: 5 - Không sử dụng Storage (các ổ SAN) - Kiến trúc(arch): x86 - Hệ điều hành (OS): Linux - Xử lý (VMM): Xen - TimeZone: +7 GMT - Cost: 3.0 - Cost per Memory: 0.05 - Cost per Storage: 0.1 - Cost per Bandwidth: 0.1	Mỗi host trong Datacenter có cấu hình như sau: - CPU có 4 nhân, mỗi nhân có tốc độ xử lý là 1000 (mips) - Ram: 16384 (MB) - Storage: 1000000 - Bandwidth: 10000

- Các máy ảo có cấu hình giống nhau khi được khởi tạo:

Bảng 4.2. Cấu hình máy ảo

Kích thước (size)	Ram	Mips	Bandwidth	Số lượng cpu (pes no.)	VMM
10000 MB	512 MB	250	1000	1	Xen

- Các Request (chạy trên web, WebRequest) đại diện bởi Cloudlet trong CloudSim và kích thước của các Cloudlet được khởi tạo một các ngẫu nhiên bằng hàm Random của JAVA. Số lượng Cloudlet lần lượt từ 500 → 1500.

Bảng 4.3. Cấu hình thông số các Request

Chiều dài (Length)	Kích thước file (File Size)	Kích thước file xuất ra (Output Size)	Số CPU xử lý (PEs)
3000 ~ 1700	5000 ~ 45000	450 ~ 750	1

- Thuật toán đề xuất được xây dựng bằng cách tạo ra lớp **RCVKASchedulingAlgorithm** kế thừa từ đối tượng **BaseSchedulingAlgorithm**. Song, thuật toán đề xuất đã có cập nhật thêm một số phương thức và thuộc tính liên quan tới **predictRequestKmeans** đồng thời điều chỉnh các hàm dựng sẵn để trở nên phù hợp hơn:

@Override

*public void **run()** // Module 3*

*public CondorVM **getFittingVm** (double label)*

// Module 2

*public String **predictRequestPowerConsume**(Cloudlet req)*

*public String **predictRequestCpuUsage**(Cloudlet req)*

*public String **predictRequestRamUsage** (Cloudlet req)*

// Module 1

Tiêu chí đánh giá

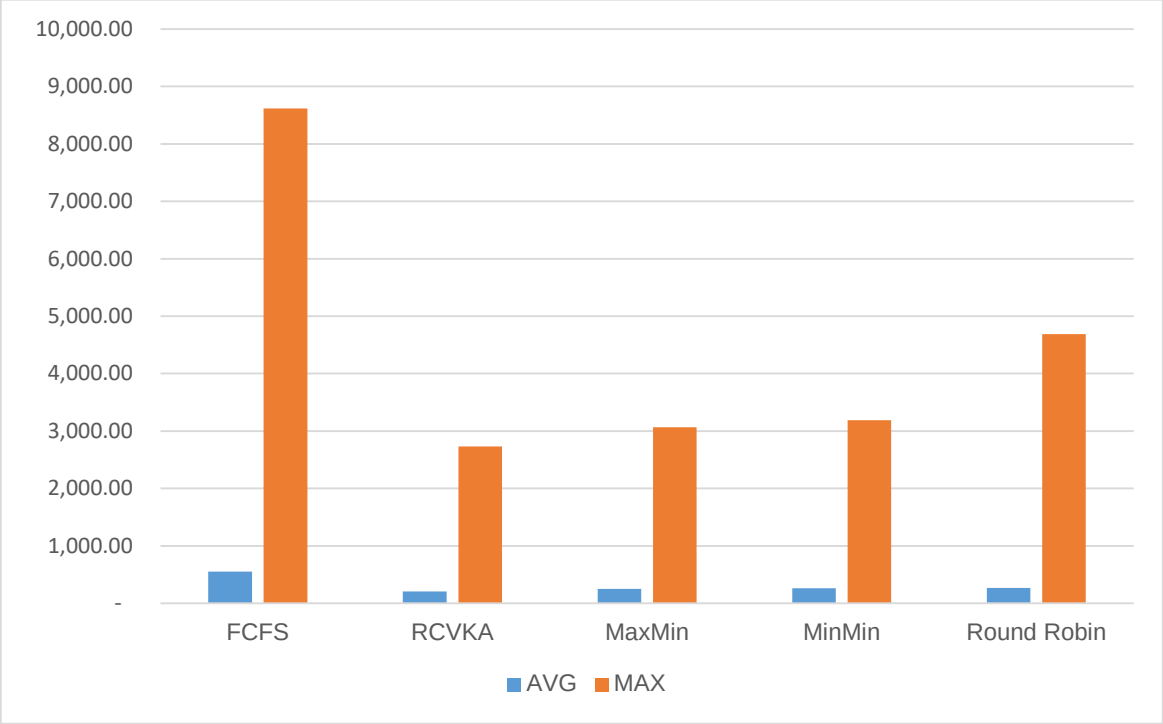
Thực nghiệm mô phỏng cloud với các tham số như trên để chạy thuật toán cân bằng tải của CloudSim có sẵn. Sau đó, kết hợp chạy thuật toán đề xuất RCVKA mới cài đặt với dữ liệu đầu vào. Cuối cùng, so sánh kết quả đầu ra, đặc biệt là thông số thời gian thực hiện (Makespan). Sai số của thời gian đáp ứng dự đoán của các máy ảo và thời gian đáp ứng dự đoán của cloud càng thấp thì hiệu quả của thuật toán càng được đánh giá cao.

4.3. Kết quả thực nghiệm của mô hình.

Ta tiến hành chạy thực nghiệm mô phỏng trên CloudSim với 5 máy ảo đã được dựng sẵn để đáp ứng các Request. Các request được khởi tạo có chiều dài và kích thước ngẫu nhiên với số lượng Request lần lượt là 30, 60, 100 và 1000. Tiếp theo, ta so sánh kết quả này với các thuật toán FCFS, MaxMin, MinMin và Round-Robin. Trước nhất, trường hợp mô phỏng với 30 Requets, ta có bảng kết quả như sau:

Bảng 4.4. Kết quả thực nghiệm mô phỏng với 30 request

Thời gian thực hiện (ms)	FCFS	RCVKA	MaxMin	MinMin	Round Robin
AVG	554.70	206.43	253.53	261.94	271.48
MAX	8,618.72	2,734.52	3,068.34	3,191.11	4,687.36
MIN	0.26	0.11	0.14	0.14	0.12

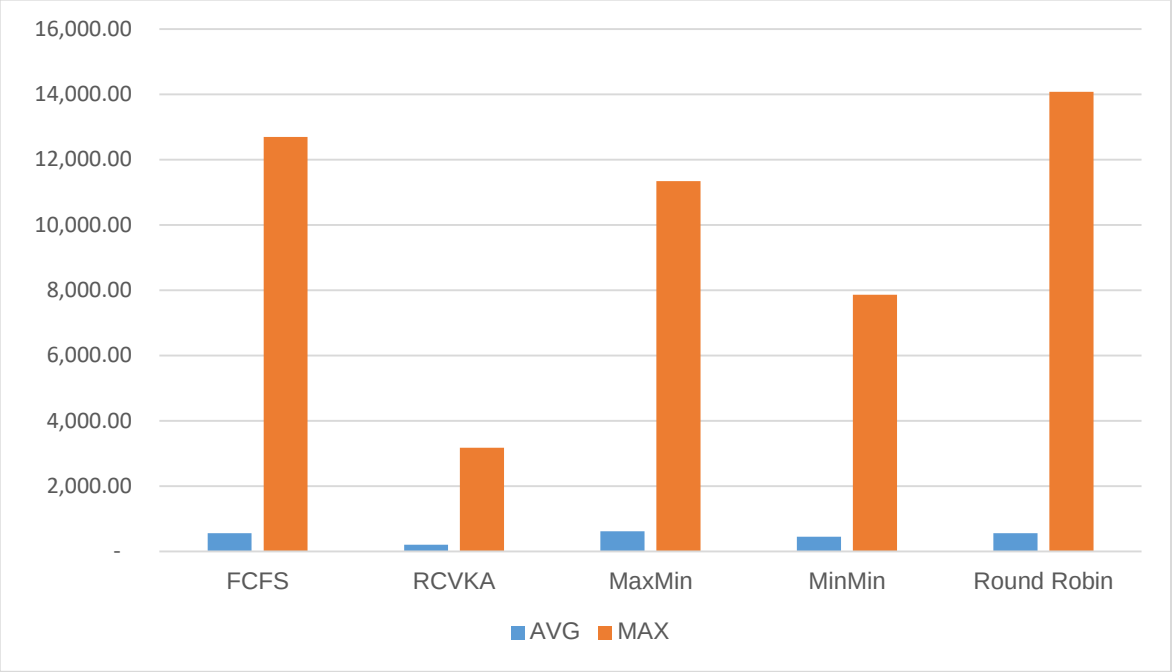


Hình 4.1. Biểu đồ so sánh thời gian thực hiện của 5 thuật toán với 30 Request

Với kết quả thực nghiệm của 30 Request, có thể thấy rằng thuật toán đề xuất RCVKA đang có lợi thế hơn so với các thuật toán còn lại. Thuật toán MaxMin và MinMin cũng theo sát thuật toán RCVKA với thời gian xử lý khá ổn định. Trong khi đó, thuật toán FCFS lại có thời gian xử lý khá cao, chiếm gấp gần 4 lần so với thuật toán đề xuất. Tuy nhiên, để chứng tỏ thuật toán đề xuất thật sự có hiệu quả, ta sẽ tiến hành xử lý nhiều request hơn và quan sát kết quả.

Bảng 4.5. Kết quả thực nghiệm mô phỏng với 60 request

Thời gian thực hiện (ms)	FCFS	RCVKA	MaxMin	MinMin	Round Robin
AVG	554.55	207.15	615.82	445.43	553.40
MAX	2,692.30	3,171.06	11,338.88	7,855.51	14,070.51
MIN	0.7	0.11	0.13	0.12	0.16

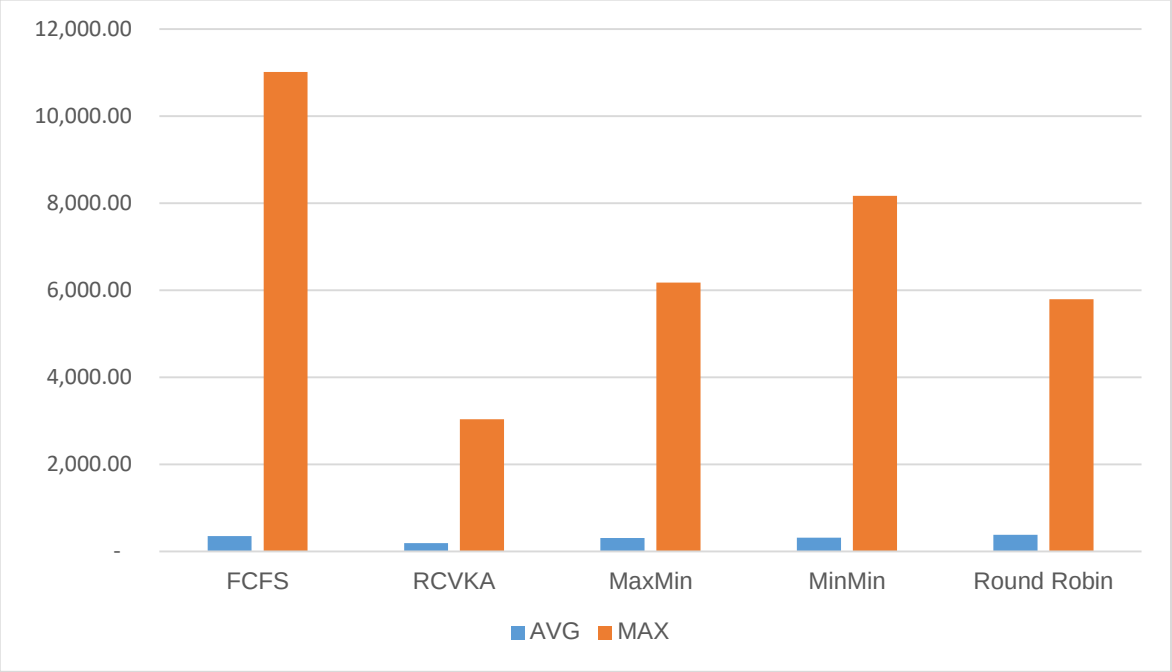


Hình 4.2. Biểu đồ so sánh thời gian thực hiện của 5 thuật toán với 60 Request

Khi thực nghiệm từ request 60 trở đi, qua quan sát biểu đồ nhận thấy có sự chênh lệch rõ rệt giữa 5 thuật toán được so sánh. Thuật toán đề xuất vẫn có thời gian xử lý nhỏ, cách biệt khá nhiều với các thuật toán khác. Cụ thể, thời gian xử lý của thuật toán Round Robin và MinMin lần lượt chiếm gần 5 lần và 2 lần thời gian xử lý của RCVKA. Có thể thấy rằng, khi số lượng request nhỏ và kích thước nhỏ, thuật toán RCVKA cũng khá ổn định.

Bảng 4.6. Kết quả thực nghiệm mô phỏng với 100 request

Thời gian thực hiện (ms)	FCFS	RCVKA	MaxMin	MinMin	Round Robin
AVG	351.87	192.10	307.26	311.67	382.52
MAX	11,015.20	3,034.15	6,176.83	8,163.80	5,792.78
MIN	0.11	0.11	0.11	0.12	0.16

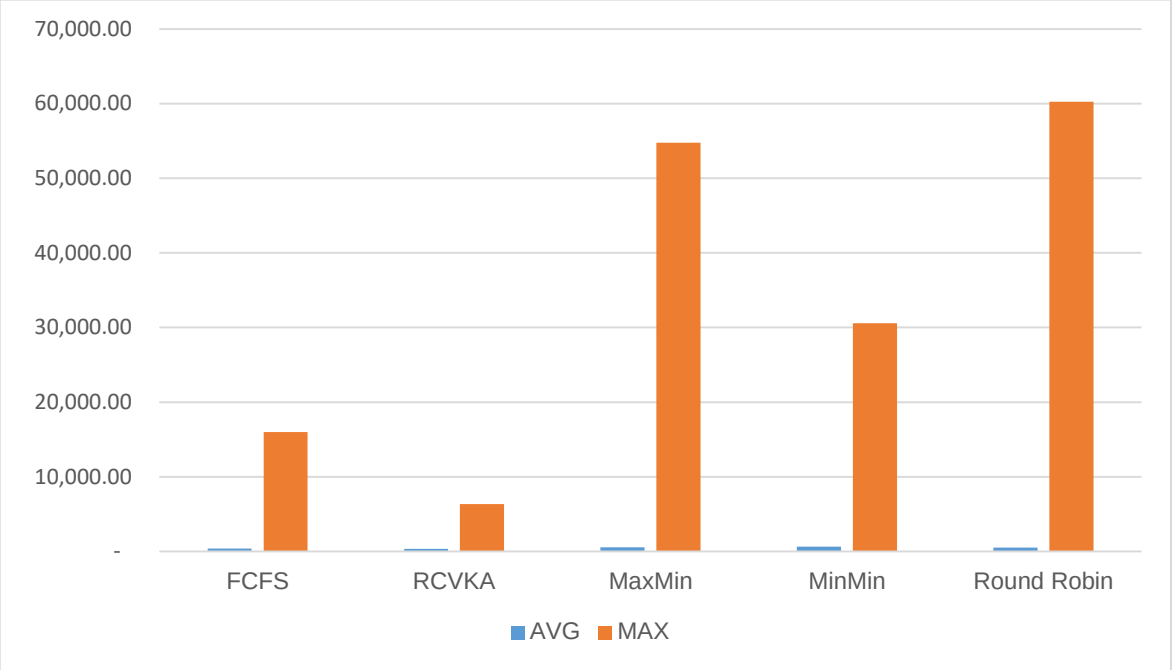


Hình 4.3. Biểu đồ so sánh thời gian thực hiện của 5 thuật toán với 100 Request

Khi tăng số lượng request lên 100, ta thấy RCVKA vượt trội hơn hẳn so với các thuật toán còn lại. Thuật toán FCFS có thời gian xử lý gấp gần 4 lần, Round Robin gấp gần 2 lần so với thuật toán đề xuất. Qua đó, nhận thấy được tính ổn định và hiệu quả của thuật toán đề xuất RCVKA dù có tăng số lượng request. Cuối cùng ta tăng lên 1000 request để quan sát liệu RCVKA có bỏ xa các thuật toán khác hay không.

Bảng 4.7. Kết quả thực nghiệm mô phỏng với 1000 request

Thời gian thực hiện (ms)	FCFS	RCVKA	MaxMin	MinMin	Round Robin
AVG	380.90	326.59	545.62	639.75	514.52
MAX	15,971.02	6,360.17	54,741.18	30,587.87	60,263.71
MIN	0.12	0.18	0.15	0.16	0.13

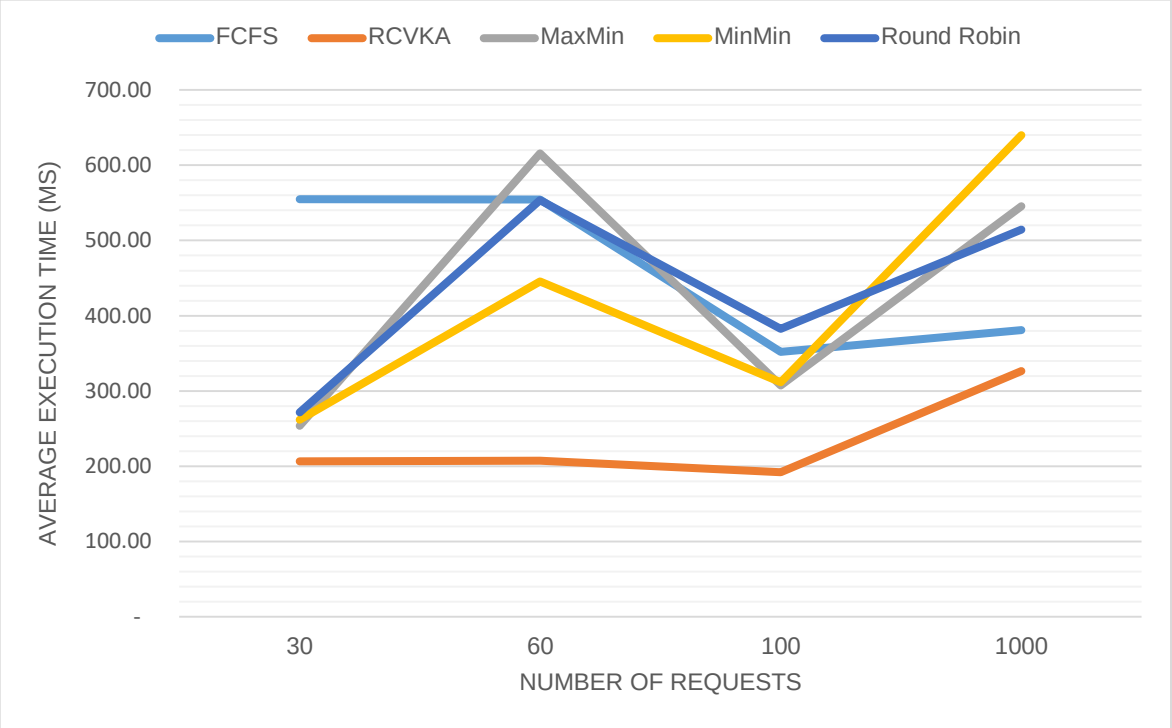


Hình 4.4. Biểu đồ so sánh thời gian thực hiện của 5 thuật toán với 1000 Request

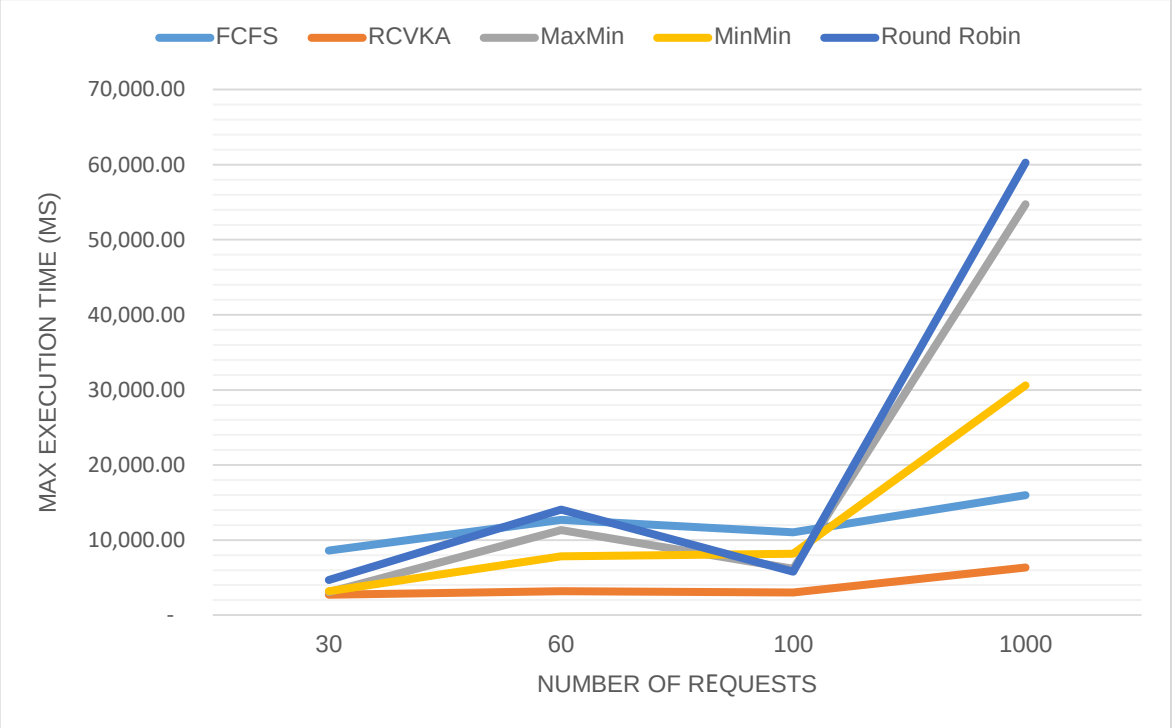
Với trường hợp 1000 request này, thuật toán RCVKA đã chứng minh mình vượt trội hơn so với 4 thuận toán còn lại. Nếu lượng request bùng nổ, việc xử lý của thuật toán vẫn đáp ứng được, phụ thuộc vào cấu hình thiết bị chạy bộ cân bằng tải vì vậy không quá khó khăn trong việc xử lý cân bằng tải nếu lượng request nhiều.

So sánh thời gian thực thi của các thuật toán

Sau khi chạy kiểm thử thuật toán đề xuất RCVKA với bốn thuật toán cơ bản với số lượng request tăng dần từ 30 – 1000, ta tiến hành vẽ biểu đồ để so sánh, thống kê thời gian thực thi trung bình của các thuật toán để quan sát và đánh giá tính ổn định giữa các thuật toán.



Hình 4.5. Thời gian thực hiện trung bình của 5 thuật toán từ 30-1000 Request



Hình 4.6. Thời gian thực hiện lớn nhất của 5 thuật toán từ 30-1000 Request

Sau khi so sánh thời gian xử lý của các thuật toán dưới cùng một điều kiện ở 4 trường hợp lần lượt là: 30, 60, 100 và 1000 request, ta đã thấy được sự phân bổ vô cùng ổn định và hợp lý của thuật toán đề xuất RCVKA. Cụ thể, khi so thời gian xử

lý của các máy ảo với thời gian xử lý của các thuật toán khác trên cloud (bao gồm trường hợp ít và nhiều request) thì sự khác biệt không đáng kể. Bên cạnh đó, Hình 4.5 và Hình 4.6 cũng cho thấy RCVKA luôn luôn nằm ở vị trí thấp nhất, dù là so sánh giá trị thời gian thực hiện trung bình hay giá trị thời gian thực hiện lớn nhất đi chăng nữa.

Tuy nhiên, thực nghiệm mô phỏng này chỉ là mô phỏng nhóm các máy ảo, chưa tính đến việc mở rộng tập hợp các máy ảo (VM pool) để giảm tải trong trường hợp cần thiết. Bởi, trong luận văn chỉ giả định nhóm các máy ảo xử lý tối đa bao nhiêu request, nếu vượt quá thì ta mới lên kế hoạch và tiến hành mở rộng pool. Ngoài ra, việc thí nghiệm mô phỏng với lượng request lớn hơn, cụ thể là trên 1000 request, thì đòi hỏi phải có một máy tính mạnh hơn với bộ xử lý tốt hơn. Vì vậy, đây cũng chính là khuyết điểm cũng như hạn chế của thí nghiệm mô phỏng này.

Thuật toán đề xuất RCVKA đã cho thấy sự hiệu quả của mình khi số lượng máy ảo được tăng lên cao hơn, thuật toán vẫn đảm bảo được thời gian phản hồi và thời gian xử lý ở mức tốt nhất có thể đồng thời giảm thiểu tối đa chi phí cho các trung tâm dữ liệu đám mây. Dẫu vậy, thuật toán đề xuất vẫn còn tồn tại một số nhược điểm như sau:

- Khi số lượng máy ảo tăng nhiều hơn, thì việc tìm ra máy có Usage nhỏ nhất cũng trở nên khó khăn hơn.

-Thế nên, trong trường hợp đó, chỉ cần có thể tìm được một máy có Usage phù hợp là đạt yêu cầu.

- Các máy ảo trong danh sách chưa được sắp xếp theo thứ tự tăng dần.

4.4. Đánh giá kết quả

Mô hình thực nghiệm mô phỏng với các thông số cũng như kịch bản đưa ra đều dựa vào quá trình request của các browser trên môi trường cloud. Qua đó, luận văn ghi nhận được các thông số về thời gian xử lý của các máy ảo cũng như của cloud. Việc chạy thực nghiệm với thông số của 5 máy ảo, chịu tải từ 30 đến 1000 Request đã cho ra kết quả tương đối khả quan. Ngoài ra, việc phân bổ các Request đến các máy ảo cũng xử lý khá đồng đều và có tính khả thi cao.

KẾT LUẬN

Luận văn "**NGHIÊN CỨU CHÍNH SÁCH BỀN VỮNG NHẰM XÂY DỰNG THUẬT TOÁN NÂNG CAO HIỆU QUẢ CÂN BẰNG TẢI CỦA ĐIỆN TOÁN ĐÁM MÂY**" nghiên cứu các thuật toán tiến hành phân lớp từ các đặc trưng của request chọn lọc bởi chính sách bền vững. Sau đó, phân bổ các tác vụ vào các máy ảo sao cho hợp lý và nâng cao cân bằng tải trong môi trường điện toán đám mây, sử dụng hợp lý và có hiệu quả nguồn tài nguyên đám mây. Dựa vào các thuật toán đã có để phân tích làm rõ chúng, sau đó có thể đánh giá đưa ra nhược điểm và lợi thế của từng thuật toán. Từ các nhược điểm đã phân tích, đề xuất một thuật toán kết hợp với chính sách bền vững nhằm cải tiến và nâng cao khả năng cân bằng tải so với các thuật toán cũ. Quá trình nghiên cứu đã đạt được nhiều mục tiêu đề ra như sau:

- Nghiên cứu tổng quan đám mây và các đám mây với ba mô hình chính đang được sử dụng. Các kỹ thuật cân bằng tải được dùng trong môi trường điện toán đám mây.
- Nghiên cứu các thuật toán hỗ trợ để phân lớp, phân cụm các request. Nghiên cứu chính sách bền vững, áp dụng vào để phân cụm các request dựa trên các đặc trưng của nó.
- Nghiên cứu cách tiếp cận đám mây điện toán thông qua mô phỏng sử dụng công cụ giao diện thân thiện và dễ sử dụng của CloudSim. Cài đặt và mô phỏng các kỹ thuật cân bằng tải, các thuật toán Round Robin, MaxMin, MinMin và thuật toán tự nhiên FCFS. Các giá trị thu được khi mô phỏng đưa ra dùng để phân tích so sánh với nhau, nhằm tóm lại được các nhược điểm và ưu điểm của các thuật toán. Từ đó, luận văn sẽ đề xuất một thuật toán sửa đổi để khắc phục mặt hạn chế đó.
- Kết quả đạt được từ thuật toán đề xuất đáp ứng được các mục tiêu như việc đáp ứng thời gian được cải thiện, hạn chế các tài nguyên bị đói cũng như máy ảo có năng lực xử lý mạnh sẽ được xử lý nhiều yêu cầu hơn. Qua đó, giúp cân bằng tải hiệu quả hơn các thuật toán được so sánh là Round Robin, MaxMin, MinMin và thuật toán tự nhiên FCFS.

- Thuật toán đề xuất RCVKA có thể dùng để đưa vào áp dụng trên thực tế.

Hạn chế luận văn

- Chưa được ứng dụng vào môi trường thực tế.
- Thời gian đáp ứng và xử lý chưa cải thiện được nhiều.

Vấn đề kiến nghị và hướng đi tiếp theo của nghiên cứu:

- Đưa thuật toán đề xuất vào ứng dụng thực tế.

Áp dụng mô hình năng lượng tiêu thụ của Datacenter hoặc cloud tương ứng để xây dựng biểu đồ phân bổ tải cho cloud.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] N. X. Phi, L. N. Hieu and T. C. Hung, "Thuật toán cân bằng tải nhằm giảm thời gian đáp ứng dựa vào ngưỡng thời gian trên điện toán đám mây," *Tạp chí Khoa học công nghệ Thông tin và truyền thông số 04(CS.01),* pp. 43-48, 2018.

[2] N. H. H. Cuong, D. H. Vy and N. T. Thuy, "A New Technical Solution Prevention Deadlock for Resource Allocation in Hetergeneous Distributed Platforms," *International Journal of Computer Science and Network,* vol. 4, pp. 266-272, 2015.

[3] N. H. H. Cuong and L. V. Son, "Detection and Avoidance Deadlock for Resource Allocation in Heterogeneous Distributed Platforms," *International Journal of Computer Science and Telecommunications,* pp. 1-5, 2015.

[4] O. Mahitha and V. Suma, "Deadlock Avoidance through Efficient Load Balancing to Control Disaster in Cloud Environment," in *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Tiruchengode, 2013.

[5] Z. Suraj, "Deadlock prediction in linear systems," in *Symposium on Computation Theory*, Berlin, 1984.

[6] O'Neil, Patrick E., "Deadlock Prediction for Escrow Transactions," *Inf. Syst,* vol. 16, p. 13–20, 1991.

[7] Rashmi, K. and Suma, V. and Nedu, Vaidehi, "Enhanced Load Balancing Approach to Avoid Deadlocks in Cloud," *Special Issue of International Journal of Computer Applications (0975 – 8887) on Advanced Computing and Communication Technologies for HPC Applications - ACCTHPCA*, 2012.

[8] N. H. H. Cuong, "Avoid Deadlock Resource Allocation (ADRA) Model V VM-out-of-N PM," *International Journal of Innovative Technology and Interdisciplinary Sciences,* vol. 2, pp. 98-107, 2018.

[9] D. Ardagna, G. Casale, M. Ciavotta, J. F. Pérez and W. Wang, "Quality-of-service in cloud computing: modeling techniques and their applications," *Journal of Internet Services and Applications,* 2014.

[10] M. H. Ghahramani, M. Zhou and C. T. Hon, "Toward Cloud Computing QoS Architecture: Analysis of Cloud Systems and Cloud Services," *IEEE/CAA Journal of Automatica Sinica,* pp. 6-18, 2017.

[11] Geeta and S. Prakash, "A Literature Review of QoS with Load Balancing in Cloud Computing Environment," in *Big Data Analytics*, Singapore, 2018.

[12] Y. Afrianto, H. Sukoco and S. Wahjuni, "Weighted Round Robin Load Balancer to Enhance Web Server Cluster in OpenFlow Networks," *TELKOMNIKA*, pp. 1402-1408, 2018.

[13] S. Shi, C. Qian, Y. Yu, X. Li, Y. Zhang and X. Li, "Concury: A Fast and Light-weighted Software Load Balancer," in *Proceedings of the 11th ACM Symposium on Cloud Computing*, 2020.

[14] M. Kumar, S. C. Sharma, A. Goel and S. P. Singh, "A comprehensive survey for scheduling techniques in cloud computing," *Journal of Network and Computer Applications,* vol. 143, pp. 1-33, 2019.

[15] E. J. Ghomi, A. M. Rahmani and N. N. Qader, "Load-balancing algorithms in cloud computing: A survey," *Journal of Network and Computer Applications,* vol. 88, pp. 50-71, 2017.

[16] T. Deepa and D. Cheelu, "A Comparative Study of Static and Dynamic Load Balancing Algorithms in Cloud Computing," *International Conference on Energy,*

Communication, Data Analytics and Soft Computing (ICECDS), 2017.

[17] G. Singh and K. Kaur, "An Improved Weighted Least Connection Scheduling Algorithm for Load Balancing in Web Cluster Systems," *International Research Journal of Engineering and Technology (IRJET),* vol. 5, pp. 1950-1955, 2018.

[18] Eswaran, S., Rajakannu, M, “Multiservice Load Balancing with Hybrid Particle Swarm Optimization in Cloud-Based Multimedia Storage System with QoS Provision,” *Mobile Netw Appl,* p. 760–770, 2017.

[19] M. Khan, R. S. Alhumaima and H. S. Al-Raweshidy, "QoS-Aware Dynamic RRH Allocation in a Self-Optimised Cloud Radio Access Network with RRH Proximity Constraint," *IEEE Transactions on Network and Service Management,* vol. 14, pp. 730-744, 2017.

[20] O. O. AJAYI, F. A. OLADEJI and C. O. UWADIA, "Multi-Class Load Balancing Scheme for QoS and Energy Conservation in Cloud Computing," *West African Journal of Industrial and Academic Research,* pp. 28-36, 2016.

[21] S. Varshney, R. Sandhu and P. K. Gupta, "QoS Based Resource Provisioning in Cloud Computing Environment: A Technical Survey," *Advances in Computing and Data Sciences, Singapore, Springer Singapore,* pp. 711-723, 2019.