HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Hoàng Tấn

ĐỀ XUẤT THUẬT TOÁN DỰ BÁO THỜI GIAN DI CHUYỂN TÁC VỤ NHẰM NÂNG CAO HIỆU NĂNG CÂN BẰNG TẢI TRÊN ĐIỆN TOÁN ĐÁM MÂY

Chuyên ngành: Hệ thống thông tin.
Mã số: 8.48.01.04

TÓM TẮT LUẬN VĂN THẠC SĨ

TP.HCM - NĂM 2022

Luận văn được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Người hướng dẫn khoa học: PGS.TS Trần Công Hùng

(Ghi rõ học hàm, học vị)

Phản biện 1: PGS.TS Trần Vĩnh Phước

Phản biện 2: TS. Đàm Quang Hồng Hải

Luận văn sẽ được bảo vệ trước Hội đồng chấm luận văn thạc sĩ tại Học viện Công nghệ Bưu chính Viễn thông

Vào lúc: 10 giờ 00 ngày 15 tháng 01 năm 2022.

Có thể tìm hiểu luận văn tại:

- Thư viện của Học viện Công nghệ Bưu chính Viễn thông.

PHẦN MỞ ĐẦU

1. Tính cấp thiết của đề tài

Điện toán đám mây là một công nghệ đầy hứa hẹn vì: Tính sẵn sàng cao, tính linh hoạt, tính an toàn dữ liệu, tính tiết kiệm, tiết kiệm thời gian, quản lý dễ dàng, hệ điều hành mẫu đa dạng.

Trên quan điểm chất lượng dịch vụ, việc quản lý tài nguyên trở thành một công việc phức tạp. Do đó, ta phải khắc phục vấn đề thiếu thốn tài nguyên, giảm độ trễ và khả năng cải thiện hiệu suất mạng. Điều này được bộ cân bằng tải xử lý và điều phối. Vì vậy, cần phải có thuật toán dự báo thời gian di chuyển tác vụ nhằm nâng cao hiệu quả cân bằng tải trên điện toán đám mây. Cụ thể, đề tài như sau: "Đề xuất thuật toán dự báo thời gian di chuyển tác vụ nhằm nâng cao hiệu năng cân bằng tải trên điện toán đám mây".

2. Tổng quan về vấn đề nghiên cứu

Cân bằng tải là kỹ thuật phân phối khối lượng công việc đồng đều giữa hai hoặc nhiều máy tính, kết nối mạng, CPU, ổ cứng, hoặc các nguồn lực phân tán to lớn trên mạng. Kỹ thuật cân bằng tải hiện nay chủ yếu tập trung vào hai kỹ thuật là cân bằng tải tĩnh và cân bằng tải động.

3. Mục đích nghiên cứu

Trên cơ sở lý thuyết đã nghiên cứu, luận văn đề xuất thuật toán dự báo thời gian di chuyển tác vụ (Migration Time) nhằm nâng cao hiệu quả cân bằng tải trên điện toán đám mây. 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 thời gian di chuyển tác vụ (Migration Time) trong cân bằng tải trên điện toán đám mây.

+ Nghiên cứu các thuật toán dự báo thời gian di chuyển tác vụ (Migration Time) trong cân bằng tải trên điện toán đám mây.

- Phạm vi nghiên cứu trong Cloud:

+ Xây dựng mô hình mô phỏng đám mây ở mức độ nhỏ: khoảng từ 10~15 máy ảo.

+ Độ phức tạp trên mỗi máy ảo chỉ ở mức độ thấp: khoảng 1 – 4 ứng dụng trên các máy ảo đó.

5. Phương pháp nghiên cứu

Phương pháp luận: Dựa trên cơ sở 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: Trên cơ sở các lý thuyết về điện toán đám mây, khả năng xảy ra tắc nghẽn trên đám mây. Đề xuất ra thuật toán để nâng cao hiệu quả cân bằng tải trên đám mây dựa trên các thuật toán đã nghiên cứu. 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.

PHẦN NỘI DUNG

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

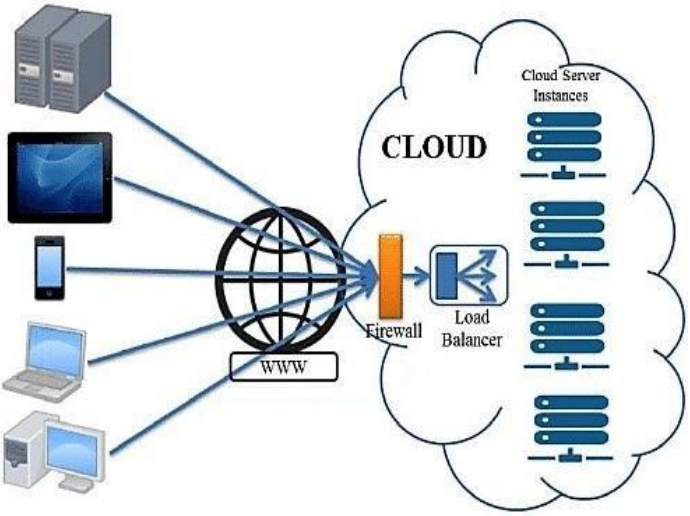
Điện toán đám mây (cloud computing) hay còn gọi là điện toán máy chủ ảo, nơi các tính toán được "định hướng dịch vụ" và phát triển dựa vào Internet. Cụ thể hơn, trong mô hình điện toán đám mây, tất cả các tài nguyên, thông tin cùng với software đều được chia sẻ và cung cấp cho các máy tính, thiết bị, người dùng dưới dạng dịch vụ trên nền tảng một hạ tầng mạng công cộng (thường là mạng Internet). 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).

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

1.2.1 Giới thiệu về cân bằng tải

Giải pháp cân bằng tải là việc phân bố 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 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 đồng thời 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 [11] có thể được chia thành 2 thể loại:

- Cân bằng tải cục bộ

- Tải toàn cầu

1.2.2 Mục đích cân bằng tải

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

Tăng độ tin cậy và khả năng dự phòng cho hệ thống.

Tăng tính bảo mật cho hệ thống.

1.3 Tổng quan về trí tuệ nhân tạo (AI)

Trí tuệ nhân tạo (AI) là một ngành khoa học máy tính liên quan đến việc tạo ra các chương trình nhằm mục đích tái tạo nhận thức con người và các quá trình liên quan đến việc phân tích sự phức tạp của dữ liệu.

Các cá nhân và tổ chức ở một số các ngành công nghiệp đang bắt đầu nhận ra tiềm năng của AI để cải thiện các hoạt động hiện tại. Vậy nên, nghiên cứu AI đã được tiến hành trong nhiều lĩnh vực: y tế, điện toán đám mây, xử lý ảnh,…

1.4 Tổng quan về machine learning

Học máy (Machine Learning / ML) là một phương pháp để tạo ra AI. ML liên quan đến các chương trình máy tính viết lập trình của riêng chúng để hoàn thành một nhiệm vụ định trước. Sử dụng các thuật toán phân lớp của ML để tiến hành phân lớp người dùng dựa trên các đặc trưng của họ để thực hiện việc cân bằng tải.

1.5 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 cũng như gói tin mất mát khi truyền dữ liệu qua môi trường điện toán và 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

Chương này xin giới thiệu các công trình liên quan đến điện toán đám mây, cân bằng tải và các thuật toán AI.

2.2. Các công trình liên quan

Năm 2015, Anita Rani và Pankajdeep Kaur công bố các nghiên cứu về di chuyển các tác vụ trên cloud thông qua "Migration Jobs in Cloud Computing".

Trong nghiên cứu của Geetha Megharaj và cộng sự vào năm 2018, bài báo "Run Time Virtual Machine Task Migration Technique for Load Balancing in Cloud" đã nghiên cứu sâu hơn về di chuyển tác vụ.

Gần đây, năm 2020, các tác giả Chen Ling, Hui He và cộng sự đã công bố "Network perception task migration in cloud-edge fusion computing".

2.3. Tổng kết chương

Thông qua việc nghiên cứu tìm hiểu được một số thuật toán và những công trình liên quan tới cân bằng tải trong điện toán đám mây. Qua đó giúp luận văn hiểu rõ hơn về cân bằng tải và 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 : ĐỀ XUẤT THUẬT TOÁN DỰ BÁO THỜI GIAN DI CHUYỂN TÁC VỤ NHẰM NÂNG CAO HIỆU NĂNG CÂN BẰNG TẢI TRÊN ĐIỆN TOÁN ĐÁM MÂY

3.1. Giới thiệu chung

Một trong các yếu tố có khả năng gây ảnh hưởng đến hiệu suất cân bằng tải trên cloud đó chính là thời gian di chuyển tác vụ đến các máy ảo. Vì thế, trong chương này, luận văn sẽ đề xuất thuật toán dự báo thời gian di chuyển tác vụ nhằm giảm thiểu sự mất cân bằng tải trên cloud.

3.2. Mô hình nghiên cứu

Sử dụng thuật toán Linear Regression để dự đoán thời gian thực hiện và phân loại các task tương ứng với các Request dựa trên độ lịch sử di chuyển task giữa các máy ảo. Độ ưu tiên ở đây 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 task đó trong cloud.

Quá trình cân bằng tải được thực hiện gồm các bước như sau:

Bước 1: Nhận thông tin input

Bước 2: Sử dụng các thuật toán dự báo Linear Regression để tiến hành dự báo thời gian di chuyển tác vụ giữa các máy ảo dựa trên các đặc trưng và trạng thái hoạt động của máy ảo.

Về mục tiêu:

- Giảm thiểu rủi ro cho hệ thống máy chủ.

- Giảm thiểu thời gian sống cho các yêu cầu trong điện toán đám mây.

- Hạn chế tối đa hoặc ngăn chặn sự mất cân bằng tải giữa các máy ảo.

Giả định:

- Bộ cân bằng tải sẽ biết trước các dịch vụ nào đang chạy trên các máy ảo vào bất cứ thời điểm nào.

- Ở đây 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 mấy khác nhau.

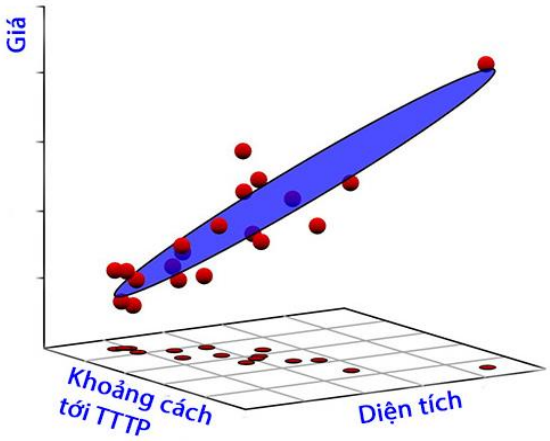
3.3. Thuật toán Linear Regression (LR)

Linear Regression (LR) là một thuật toán học giám sát (supervised learning) được sử dụng rộng rãi trong các bài toán liên quan đến việc dự đoán đầu ra của dữ liệu dựa trên các đặc trưng từ tập dữ liệu. Nói một cách khác, thuật toán tìm ra phương trình tuyến tính dựa trên tập dữ liệu quan hệ giữa X (dữ liệu đầu vào) và Y (dữ liệu đầu ra). X là biến giải thích và Y là biến phụ thuộc.

Bên cạnh đó, độ chính xác của thuật toán còn phụ thuộc vào hàm mất mát (loss function) vì để tìm được một

đường thẳng đi qua tất cả các điểm input cho trước là điều không thể. Một ví dụ đơn giản để mô phỏng ý tưởng của thuật toán:

Để dự đoán giá của một căn nhà, cần phải phụ thuộc vào nhiều yếu tố như diện tích, vị trí, thời điểm,… Tuy nhiên, giả sử giá nhà phụ thuộc vào hai yếu tố là diện tích và khoảng cách tới trung tâm thành phố, khi đó thuật toán Linear Regression với hai biến có nhiệm vụ tìm ra mặt phẳng biểu diễn sự phụ thuộc giữa giá nhà với hai yếu tố trên một cách chính xác nhất.



Hình 3.1: Không gian ba chiều biểu diễn các mối quan hệ đầu vào

Phương trình mặt phẳng trên có thể viết dưới dạng tổng quát sau:

$$\hat{P}(s,d) = w_0 + w_1 s + w_2 d$$

với $\hat{P}(s,d)$ là giá căn nhà diện tích s và cách trung tâm thành phố

d là khoảng cách

Trong trường hợp tổng quát, input bao gồm biến x_1, x_2,…, x_n và output y phụ thuộc vào n biến đó theo phương trình tuyến tính thì thuật toán tìm ra phương trình gọi là Linear Regression n biến:

$$\hat{y} = w_0 + w_1x_1 + \ldots + w_{n-1}x_{n-1} + w_nx_n$$

Để đơn giản hóa phương trình ở trên, ta có thể đặt w = $[w_0\ w_1\ \ldots\ w_n\ ]$ là vector các hệ số của phương trình, x = $[1\ x_1\ \ldots\ x_n\ ]$ là vector các biến của input (phần tử 1 trong vector đóng vai trò như x_0 chỉ nhằm mục đích thuận tiện cho tính toán) thì có thể viết lại phương trình trên như sau:

$$\hat{y} = xTw$$

3.4. Thuật toán đề xuất cân bằng tải

Task Migration Time **P**rediction using **L**inear **Reg**ression Analysis **A**lgorithm TLRegA

Dựa vào các mô hình cân bằng tải đã được công bố hiện nay [17], [18], chúng ta tiến hành đánh giá mô hình của mình. Từ đó, ta biết cách phân bố tài nguyên cho các request này.

Dựa vào tham khảo từ tài liệu [19], luận văn này xin đề xuất thuật toán gồm 3 nhóm module chính:

(1) Module dự đoán thời gian di chuyển tác vụ dùng thuật toán Linear Regression:

Trong module này, thuật toán Linear Regression sẽ dựa vào lịch sử di chuyển tác vụ giữa các máy ảo. Từ đó có thể dự đoán và đưa ra các kế hoạch cũng như chiến lược phân bổ tài nguyên một cách hợp lý đến các máy ảo, nâng cao hiệu suất cân bằng tải trên điện toán đám mây.

Nhóm Thời Gian xử lý = LR (X_1, X_2,…, X_n)

Trong đó X_i là thời gian di chuyển các tác vụ.

(2) Module phân lớp tác vụ:

Trong module này sẽ sử dụng thuật toán phân cụm K-Means (với k = 3) để phân cụm các máy ảo dựa vào mức động hoạt động cũng như sử dụng tài nguyên của máy ảo bao gồm cụm cao, trung bình và thấp. Việc phân cụm máy ảo này dựa vào thông số tức thời của các máy ảo:

$Cluster_i$ = K-Means (CPU usage, RAM, …);

Trong đó: i = 1 là nhóm thấp

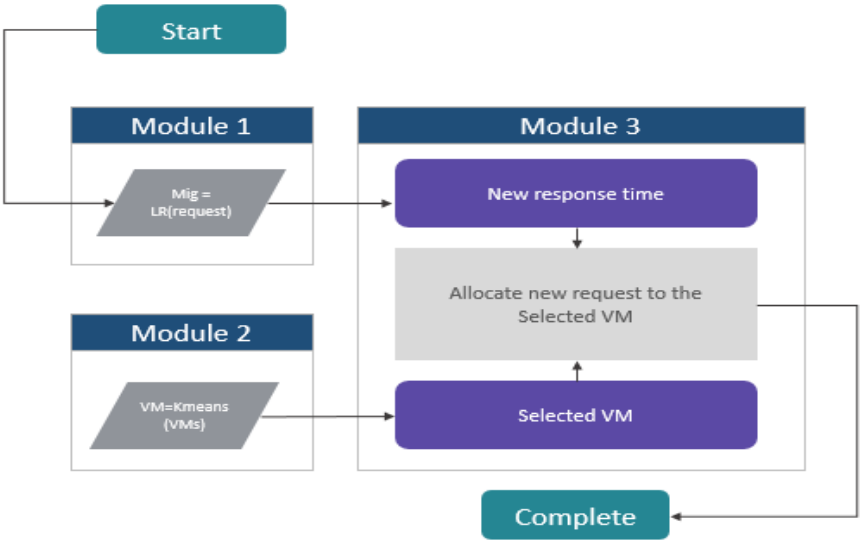
i = 2 là nhóm trung bình

i = 3 là nhóm cao

(3) Module phân bổ tác vụ vào máy ảo dựa trên thời gian dự báo

Module này có nhiệm vụ phân bổ các tác vụ đến các máy ảo theo thời gian dự báo các tác vụ. Nếu một yêu cầu được gửi đến thì yêu cầu này được phân loại bởi Module 1

và các VM đang xét, kể cả VM không tải cũng được phân cụm theo Module 2.



Hình 3.2: Sơ đồ hoạt động của thuật toán TLRegA

Thuật toán TLRegA

(1) **For each** Task **in** VMTasks

(2) isLocated = true;

(3) Mig_Time = LR(X1, X2…..); // Module 1

(4) VM_Cluster = kMeans(state); // state: Trạng thái của các VM Module 2

(5) **For each** VM **in** VMList

(6) **If** isFitSituation(Task.Mig_Time , VM.VM_Cluster)

```
(7)                         AllocateRequestToVM(VM, Request); // Module 3
(8)                         isLocated = true;
(9)                         break;
(10)                    End If
(11)               End For
(12)               If (!isLocated)
(13)                    VM = VMList.getMinFromMean(); // Module 2
(14)                    AllocateRequestToVM(VM, Request);
(15)               End If
(16)     End For
```

3.5. Kết luận chương 3

Chương này đưa ra mô hình nhằm giải quyết vấn đề cân bằng tải thông qua các kỹ thuật AI hiện đại. Với những mục tiêu ban đầu là duy trì tính ổn định và hoạt động liên tục của cloud, thuật toán đề xuất TLRegA đã chứng minh được tính hiệu quả của nó trong quá trình cân bằng tải. Cụ thể, thuật toán giảm thiểu được thời gian hoạt động cho các yêu cầu và các rủi ro nhất định đồng thời ngăn chặn sự mất cân bằng tải và hạn chế tối đa sự mất cân bằng tải giữa các máy ảo.

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

4.1. Giới thiệu chương 4

Trong chương này trình bày về cài đặt mô phỏng thuật toán TLRegA, cụ thể như sau: sử dụng thuật toán Linear Regression nhằm mục đích loại các task tương ứng với các Request dựa trên độ ưu tiên xử lý task đó.

4.2. Mô tả môi trường mô phỏng thực nghiệm

Dựa vào dữ liệu của các request mà ta có thể biết, ta sử dụng thuật toán Regression để phân loại request bằng cách tính toán ra bộ Priority = {Power, CPU, RAM}. Qua đó, ta biết cách phân bố tài nguyên cho các request vào các máy ảo đã phân cụm. Kết hợp với đánh giá số lần sai và sai số, ta cải thiện thuật toán bằng cách áp dụng máy học vào. Tuy nhiên, việc áp dụng này sẽ ít diễn ra vì có sai số cho phép. Giả lập môi trường cloud sử dụng bộ thư viện CloudSim và lập trình trên ngôn ngữ JAVA; Môi trường giả lập cloud là từ 25 đến 75 máy ảo, và tạo môi trường request ngẫu nhiên tới các dịch vụ trên cloud này. Bao gồm dịch vụ cung cấp máy ảo, dịch vụ cung cấp và đáp ứng người dùng của CloudSim để thử nghiệm. Cài đặt thuật toán Linear Regression trên môi trường mô phỏng Cloud Analyst.

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

- 01 Datacenter với thông số như 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 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 (các request chạy trên web, WebRequest) được 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 là 20 □ 1000.

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 TLRegALoadBalancer kế thừa từ đối tượng VmLoadBalancer, đồng thời thực thi giao diện CloudSimEventListener và điều chỉnh các hàm dựng sẵn để phù hợp với thuật toán đề xuất:

@Override

public int getNextAvailableVm()

// Module 2

public void cloudSimEventFired(CloudSimEvent e)

// Module 1

Tiêu chí đánh giá:

Thực nghiệm mô phỏng cloud với các tham số như trên và chạy thuật toán cân bằng tải của CloudSim có sẵn. Song song đó, chạy thuật toán đề xuất mới cài đặt với cùng đầu vào và so sánh kết quả đầu ra, đặc biệt là thông số thời gian xử lý.

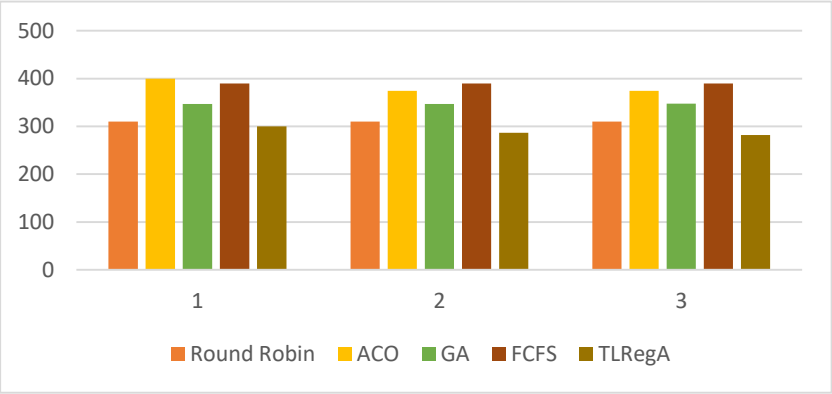
4.3. Thực nghiệm và kết quả mô phỏng

Kết quả chạy thực nghiệm mô phỏng trên CloudSim với 1 Datacenter và số lượng các máy ảo lần lượt là 25, 50,

75 được dựng sẵn để đáp ứng các yêu cầu. Các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên. Thực hiện với các thuật toán Round Robin, ACO, GA và FCFS thì có thời gian thực hiện là:

Bảng 4.4 Kết quả thực nghiệm mô phỏng với 1 DC

Cấu hình Cloud	Số máy ảo trong Datacenter	Round Robin	ACO	GA	FCFS	TLRegA
CC1	25	309.92	399.8	346.85	389.45	300.35
CC2	50	310.12	374.27	347.16	389.87	286.82
CC3	75	310.38	374.59	347.45	390.09	282.09



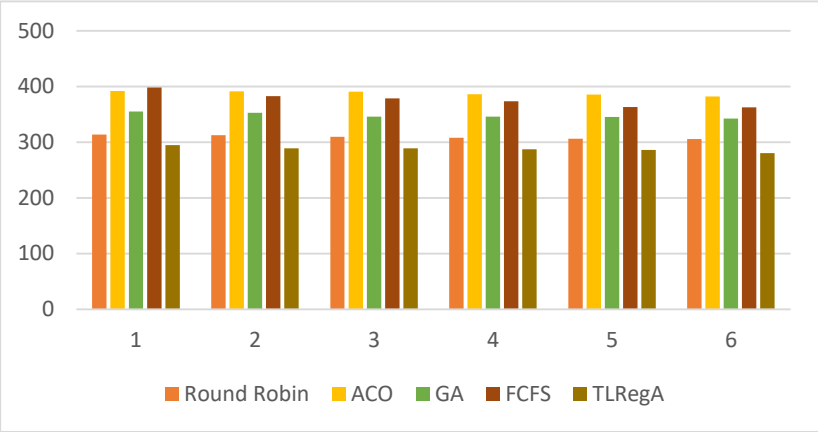
Hình 4.1: Biểu đồ thể hiện hiệu quả của thuật toán đề xuất so với các thuật toán Round Robin, ACO, GA, FCFS sử dụng 1 Datacenter

Kết quả thực nghiệm thực nghiệm sử dụng 1 Datacenter với số lượng máy ảo lần lượt là 25, 50, và 75

cho thấy thuật toán đề xuất có thời gian phản hồi tốt hơn so với các thuật toán khác ngay từ bước thực nghiệm đầu tiên.

Bảng 4.5 Kết quả thực nghiệm mô phỏng với 2 DC

Cấu hình Cloud	Số máy ảo trong Datacenter	Round Robin	ACO	GA	FCFS	TLRegA
CC1	25	313.37	392.02	354.99	398.12	294.66
CC2	50	312.69	390.93	352.87	382.64	289.16
CC3	75	309.54	390.86	345.93	378.73	289.08
CC4	25, 50	308.12	386.32	345.53	373.47	287
CC5	25, 75	306.18	385.67	345.13	363.28	286.29
CC6	75, 50	305.42	382.12	342.27	362.68	280.32

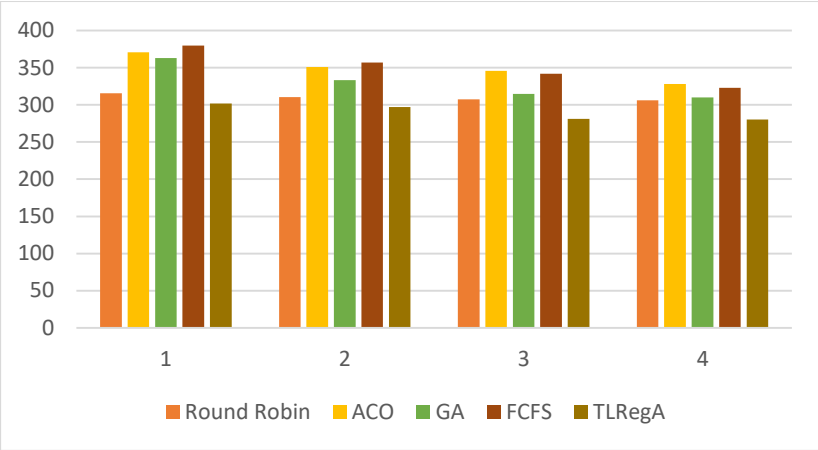


Hình 4.2: Biểu đồ thể hiện hiệu quả của thuật toán đề xuất so với các thuật toán Round Robin, ACO, GA, FCFS sử dụng 2 Datacenter

Từ 2 Datacenter trở đi, thuật toán TLRegA trở nên ổn định và có xu hướng giảm khi thay đổi (tăng) số lượng các máy trong Datacenter. Tương tự như thuật toán TLRegA, các thuật toán khác cũng có thời gian phản hồi tỉ lệ nghịch với số lượng máy ảo (số máy ảo càng tăng thời gian xử lí càng giảm). Đối với môi trường thực nghiệm này, điểm chung giữa các thuật toán là đều có xu hướng giảm khi số lượng máy ảo tăng. Dù vậy, thuật toán đề xuất vẫn dẫn đầu với thời gian xử lí thấp hơn các thuật toán còn lại.

Bảng 4.6 Kết quả thực nghiệm mô phỏng với 3 DC

Cấu hình Cloud	Số máy ảo trong Datacenter (DC)	Round Robin	ACO	GA	FCFS	TLRegA
CC1	25	315.72	370.93	363.12	379.88	301.87
CC2	50	310.55	350.86	333.46	356.96	297.11
CC3	75	307.48	345.67	314.65	342.04	281.12
CC4	25, 50, 75	306.19	328.12	310.05	322.97	280.36

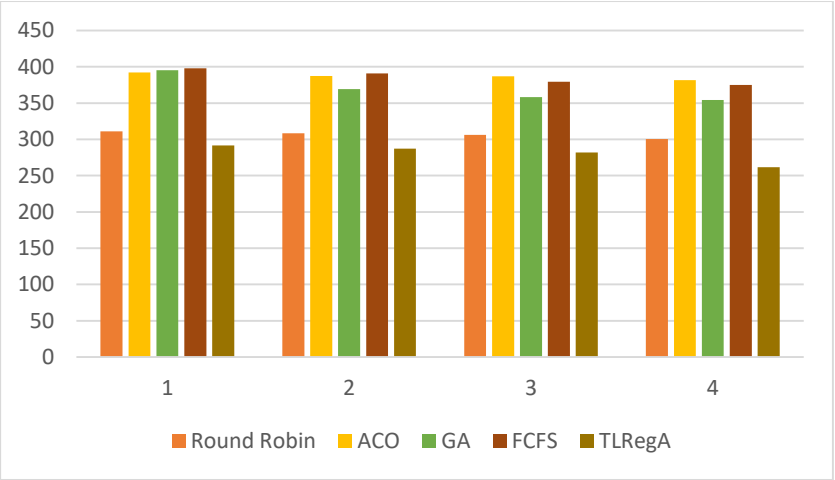


Hình 4.3: Biểu đồ thể hiện hiệu quả của thuật toán đề xuất so với các thuật toán Round Robin, ACO, GA, FCFS sử dụng 3 Datacenter

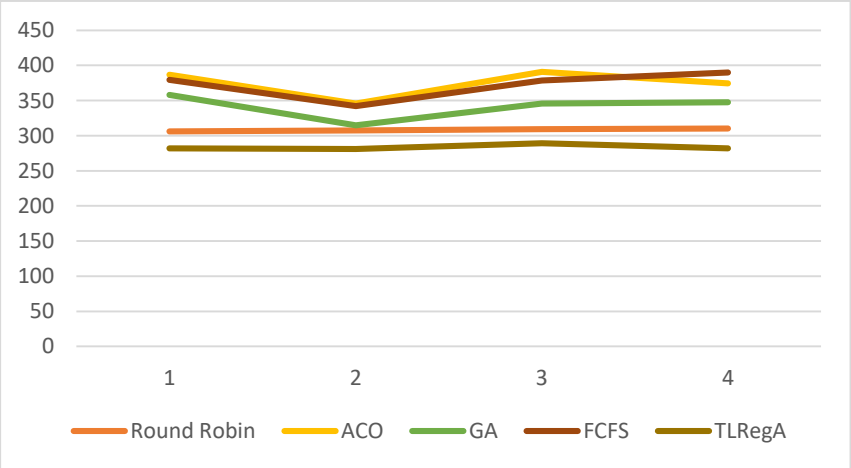
Kết quả thực nghiệm với 3 Datacenter càng làm rõ được mối quan hệ trái ngược nhau giữa số lượng Datacenter và thời gian thực thi tác vụ. Đồng thời, nó cũng thể hiện được tính hiệu quả và ổn định của thuật toán đề xuất.

Bảng 4.7 Kết quả thực nghiệm mô phỏng với 4 DC

Cấu hình Cloud	Số máy ảo trong Datacenter	Round Robin	ACO	GA	FCFS	TLRegA
CC1	25	310.86	392.06	395.17	397.97	291.81
CC2	50	308.39	387.3	369.3	390.79	287.25
CC3	75	306.03	386.95	358.12	379.37	281.99
CC4	25, 50, 75	300.48	381.75	354.25	375.25	261.72



Hình 4.4: Biểu đồ thể hiện hiệu quả của thuật toán đề xuất so với các thuật toán Round Robin, ACO, GA, FCFS sử dụng 4 Datacenter



Hình 4.5: Biểu đồ thể hiện so sánh thuật toán đề xuất với các thuật toán Round Robin, ACO, GA, FCFS sử dụng 75 máy ảo và các giá trị thay đổi của Datacenter

Lần thực nghiệm cuối cùng với 4 Datacenter, có thể thấy rằng thuật toán đề xuất đã chứng tỏ được năng lực của mình so với các thuật toán còn lại. Qua tất cả các lần thực nghiệm với số lượng Datacenter và máy ảo tương ứng, ta càng thấy rõ hiệu quả và ưu thế của thuật toán TLRegA. Thông qua biểu đồ đường so sánh kết quả thực nghiệm các thuật toán với số lượng máy ảo là 75, có thể nhận thấy thuật toán FCFS và ACO có thời gian tải trễ hơn so với 3 thuật toán còn lại. Trong đó, FCFS chiếm ưu thế hơn ở các cấu hình đầu tiên. Tuy vậy, khi số lượng các máy ảo thay đổi, ACO có xu hướng giảm về mặt thời gian hay nói cách khác là tính hiệu quả cao hơn FCFS.

4.4. Kết luận chương 4

Chương 4 của luận văn trình bày mô hình thực nghiệm mô phỏng, các thông số cũng như kịch bản đưa ra là dựa vào quá trình request của các browser trên môi trường Cloud Analyst. Từ đó, ghi nhận các thông số về thời gian dự báo di chuyển các tác vụ giữa các máy ảo. Việc chạy thực nghiệm mô phỏng với các số lượng Datacenter từ 1 - 4 và số lượng các máy ảo linh hoạt lần lượt là 25, 50 và 75 đã cho thấy kết quả tương đối tốt. Qua đó thấy được việc phân bổ các request đến các máy ảo xử lý khá đồng đều và có tính khả thi cao. .

KẾT LUẬN

Luận văn "Đề xuất thuật toán dự báo thời gian di chuyển tác vụ nhằm nâng cao hiệu năng cân bằng tải trên điện toán đám mây" về cơ bản đã đáp ứng được những mục tiêu ban đầu đề ra. Dựa vào các thuật toán đã có sẵn như FCFS, Round Robin, ACO, GA, luận văn đã phân tích, đánh giá cách thức xây dựng các thuật toán. Từ đó, ta có thể tìm ra được các ưu, nhược điểm của từng thuật toán và đưa ra giải pháp phù hợp. Nhận thấy được những điều còn chưa tốt ở các thuật toán thế hệ trước, luận văn đã đề xuất ra một thuật toán có khả năng cải tiến và nâng cao cân bằng tải một cách tối ưu hơn.

Luận văn sử dụng ba mô hình chính để nghiên cứu tổng quan về đám mây và các đám mây. Trong đó, các kỹ thuật cân bằng tải được áp dụng trong môi trường điện toán đám mây.

Nghiên cứu cách tiếp cận điện toán đám mây thông qua môi trường mô phỏng CloudSim và Cloud Analyst với công cụ giao diện dễ sử dụng và thân thiện với người dùng. Từ đó, cài đặt và mô phỏng các kỹ thuật cân bằng tải cũng như các thuật toán đã được đưa vào so sánh. Các giá trị thu được sẽ dùng để phân tích nhằm đúc kết những mặt lợi thế và hạn chế của mỗi thuật toán. Qua đó, ta sẽ kịp thời định hướng đề xuất một thuật toán với mục đích cao nhất là khắc phục những thiếu sót còn hiện hữu.

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 của các tài nguyên bị đói, máy ảo có năng lực xử lý mạnh sẽ được xử lý nhiều yêu cầu hơn. Kết quả từ các thực nghiệm mô phỏng cho thấy, thuật toán đề xuất có khả năng đáp ứng được hầu hết các mục tiêu mà chúng ta mong đợi. Cụ thể, TLRegA cải thiện được thời gian đáp ứng, giảm thiểu thực trạng các tài nguyên bị đói cũng như tăng cường năng lực xử lý cho các máy ảo để giải quyết được nhiều hơn yêu cầu của người dùng. Hơn nữa, thuật toán đề xuất TLRegA giúp việc cân bằng tải được thực hiện hiệu quả hơn so với các thuật toán còn lại: Round Robin, ACO, GA và FCFS.

Thuật toán đề xuất có thể xem xét đưa vào áp dụng trong thực tế.

- Hạn chế luận văn

+ Vẫn chỉ là nghiên cứu được đề xuất và chưa được ứng dụng vào môi trường thực tế.

+ Thời gian xử lý và đáp ứng được cải thiện hơn so với các thuật toán cũ nhưng chưa nhiều.

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

+ Tiếp tục nâng cấp thuật toán đề xuất và đưa vào ứng dụng thực tế trong tương lai.

+ Xây dựng biểu đồ phân bổ tải cho cloud dựa trên việc áp dụng mô hình năng lượng của Datacenter hoặc cloud tương ứng.