BỘ THÔNG TIN VÀ TRUYỀN THÔNG
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Khắc Chiến

NGHIÊN CỨU GIẢI PHÁP TỰ ĐỘNG ĐIỀU CHỈNH TÀI NGUYÊN HIỆU QUẢ TRONG ĐIỆN TOÁN ĐÁM MÂY

Chuyên ngành: Kỹ thuật máy tính
Mã số: 9.48.01.06

TÓM TẮT LUẬN ÁN TIẾN SĨ KỸ THUẬT

Hà Nội - 2019

Công trình được hoàn thành tại: Học viện Công nghệ Bưu chính Viễn thông

Người hướng dẫn khoa học:

1. **GS. TSKH. Hồ Đắc Lộc**
2. **TS. Nguyễn Hồng Sơn**

Phản biện:.....

.....

Phản biện:.....

.....

Phản biện:.....

.....

Luận án sẽ được bảo vệ trước Hội đồng cấp Học viện Công nghệ Bưu chính Viễn thông chấm luận án tiến sĩ họp tại..

.....

vào hồi …… giờ …… ngày …… tháng …… năm …….

Có thể tìm hiểu luận án tại:

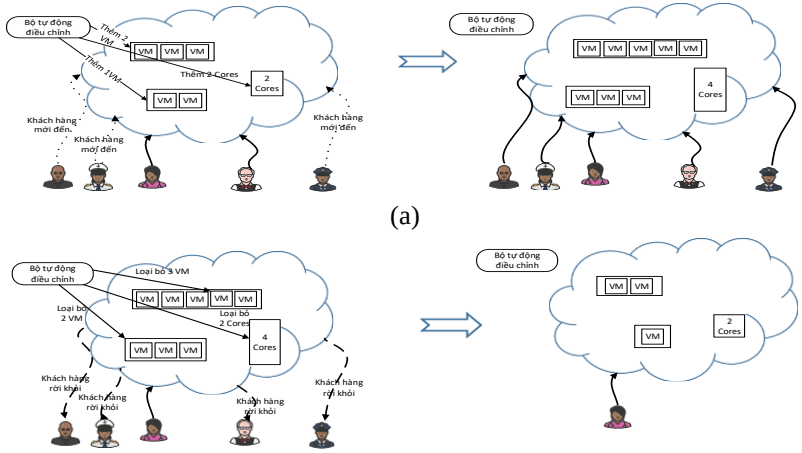
- Thư viện Quốc gia Việt Nam
- Thư viện Học viện Công nghệ Bưu chính Viễn thông

MỞ ĐẦU

1. TÍNH CẤP THIẾT CỦA LUẬN ÁN

Trong điện toán đám mây (CC), tự động điều chỉnh (AS) cung cấp và giải phóng các tài nguyên tạm thời để phục vụ cho các tải ứng dụng biến động mà không có sự can thiệp của con người sao cho chi phí tài nguyên ít nhất và thỏa mãn mức thỏa thuận dịch vụ (SLA) hay mục tiêu mức dịch vụ (SLO) của ứng dụng.

Hình 0.1 minh họa các trường hợp AS. Hình 0.1(a), khi nhu cầu đến tăng lên, tài nguyên hiện tại rơi vào tình trạng quá tải, bộ AS quyết định cung cấp một số tài nguyên tạm thời bằng cách thêm các VM mới hoặc là thêm khả năng tính toán (ví dụ: thêm core) cho các VM hiện có. Ngược lại, trong hình 0.1(b), bộ AS giải phóng một số tài nguyên khi số lượng yêu cầu giảm xuống bằng cách tắt một số VM hiện có hoặc là giảm công suất của các VM hiện có (ví dụ: giảm số core).



(a)

(b)

Hình 0.1: Ví dụ một trường hợp tự động điều chỉnh

Một số khó khăn và thách thức cần được giải quyết trong việc xây dựng các giải pháp AS, như:

- Các nghiên cứu về AS ở mức dịch vụ còn hạn chế. AS bao gồm các mô hình dịch vụ đám mây đa dạng, nhưng hầu hết các nghiên cứu chỉ tập trung vào mức cơ sở hạ tầng. AS ở mức dịch vụ là quan trọng như các dịch vụ đang chạy trên một tập hợp các VM được kết nối và QoS phụ thuộc vào cách AS xử lý các tài nguyên cho các VM như thế nào. Các số đo mức dịch vụ như số giao dịch trên mỗi đơn vị thời gian cần phải được ánh xạ tới các số đo mức hệ thống như là: mức sử dụng CPU, mạng,...

- AS trong môi trường CC hỗn hợp cũng không được hỗ trợ tốt. Đám mây hỗn hợp, trong đó có một phần ứng dụng được triển khai trên đám mây riêng và phần

khác được triển khai trên đám mây công cộng. Trong trường hợp này, các đám mây công cộng và đám mây riêng có thể cung cấp các giải pháp AS khác nhau mà không tương thích với nhau, do đó sẽ có vấn đề không tương thích giữa các tài nguyên AS trên hai đám mây.

- Hiệu quả của AS theo độ tin cậy của quá trình AS không được quản lý tốt. Thất bại của quá trình AS có thể dẫn đến vi phạm các yêu cầu về QoS của hệ thống như là: vấn đề hiệu năng, khả năng co giãn và thậm chí phải chịu chi phí không cần thiết.

- Hạn chế về các nghiên cứu chỉ ra mối quan hệ giữa AS và các thuộc tính khác như: độ sẵn sàng, độ tin cậy và độ an ninh. Ví dụ, các cuộc tấn công từ chối dịch vụ DDoS có thể gây ra một dịch vụ AS để mở rộng hệ thống không cần thiết và do đó làm tăng chi phí vận hành.

- Xác định ảnh hưởng của các cơ chế hệ thống cơ bản đến dịch vụ AS trong CC, như kỹ thuật cân bằng tải, lập lịch hay di trú, từ đó đưa ra các giải pháp mới thích nghi trong điều kiện AS nhằm nâng cao hiệu năng đám mây.

- Các giải pháp AS đã được đề xuất trong các công trình được đánh giá và thực nghiệm trên các môi trường và công cụ khác nhau. Mỗi giải pháp đều có những ưu và hạn chế nhất định.

Hiện nay, một số giải pháp AS vẫn phụ thuộc vào người sử dụng đám mây phải xác định và cấu hình bộ điều khiển đám mây. Kết quả là, người sử dụng các luật đã được định nghĩa trước có thể dẫn đến quyết định điều chỉnh tối ưu cục bộ và tốn tiền trả cho các nhà cung cấp ứng dụng CC.

Để khắc phục hạn chế trên, luận án nghiên cứu và đề xuất giải pháp AS tài nguyên hiệu quả trong CC bằng cách sử dụng bộ điều khiển mờ kết hợp với phương pháp học tăng cường để điều chỉnh và cải thiện các chính sách AS khi thực hiện. Phương pháp học tăng cường được sử dụng là học Q, cho phép hệ thống học từ sự tương tác với môi trường, trong đó việc học được thực hiện thông qua cơ chế phần thưởng. Sự kết hợp của điều khiển mờ và học Q mờ tạo ra một cơ chế tự thích nghi mạnh mẽ, trong đó điều khiển mờ thực hiện việc lập luận ở mức độ trừu tượng cao hơn, lập luận giống con người và học Q cho phép thích nghi với bộ điều khiển. Bộ AS này có thể bắt đầu làm việc mà không cần có tri thức ban đầu (với tri thức ban đầu rỗng), và tri thức sẽ có được trong quá trình thực hiện, thông qua cơ chế tiến hóa tri thức để học được tập luật tối ưu dùng để điều chỉnh tài nguyên theo các tham số đầu vào.

Ngoài ra, luận án còn nghiên cứu một số vấn đề có ảnh hưởng đến hiệu năng của hoạt động dịch vụ AS như các kỹ thuật cơ bản của CC: kỹ thuật cân bằng tải, di trú. Tiếp theo, luận án cũng nghiên cứu các mô hình hàng đợi để mô hình hệ thống CC và đề xuất một mô hình CC sử dụng mạng hàng đợi – mạng jackson mở nhằm đánh giá một số số đo hiệu năng trong hệ thống CC.

2. ĐỐI TƯỢNG VÀ PHẠM VI NGHIÊN CỨU

Đối tượng nghiên cứu của luận án là các giải pháp kỹ thuật cơ bản như kỹ thuật cân bằng tải và di trú trong CC; mô hình hóa môi trường CC bằng một mô hình mạng hàng đợi mở; và giải pháp AS tài nguyên hiệu quả trong CC.

Phạm vi nghiên cứu của luận án tập trung áp dụng các giải pháp, kỹ thuật trên nền tảng cơ sở hạ tầng CC.

3. PHƯƠNG PHÁP NGHIÊN CỨU

Phương pháp nghiêu cứu của luận án là:

- Với nội dung 1: Luận án nghiên cứu các kỹ thuật cân bằng tải và di trú trong CC hiện có, chỉ ra vấn đề hạn chế, sau đó đề xuất giải pháp để khắc phục những hạn chế đó. Sử dụng thực nghiệm để đánh giá tính hiệu quả của kỹ thuật đề xuất.

- Với nội dung 2: Luận án nghiên cứu các mô hình mạng hàng đợi – mạng Jackson mở, để đề xuất ra một mô hình cho hệ thống CC. Sử dụng mô phỏng để đánh giá tính đúng đắn của mô hình được đề xuất.

- Với nội dung 3: Luận án nghiên cứu logic mờ và học tăng cường để đề xuất ra bộ AS hiệu quả cho CC. Sử dụng thực nghiệm để so sánh, đánh giá với các giải pháp hiện có.

Về thực nghiệm và mô phỏng: Luận án thực hiện thử nghiệm các kỹ thuật, giải pháp đề xuất dựa vào công cụ mô phỏng CC hoặc tự lập trình mô phỏng trên ngôn ngữ lập trình Java: CloudSim, sử dụng cơ sở dữ liệu sinh ngẫu nhiên, hoặc có thể lấy bộ dữ liệu trên Internet, hoặc được thiết lập cố định theo cấu hình nhất định,…

4. CÁC ĐÓNG GÓP CỦA LUẬN ÁN

Đóng góp của luận án bao gồm:

- Đề xuất một kỹ thuật cân bằng tải hiệu quả và một kỹ thuật di trú hiệu quả trong CC tạo điều kiện thuận lợi cho việc điều chỉnh tài nguyên giúp cải thiện hiệu năng của các trung tâm CC. Đã minh chứng cơ chế cân bằng tải có ảnh hưởng đến tính hiệu quả của cơ chế AS tài nguyên và kỹ thuật cân bằng tải được đề xuất có tác động tích cực đến AS tài nguyên. Kết quả nghiên cứu được công bố trong các công trình TCNN1, TCTN1, HNQT1, HNQT2 và được trình bày trong Chương 2 của luận án.

- Đề xuất một mô hình CC sử dụng mạng hàng đợi - mạng Jackson mở, làm cơ sở để đánh giá các thông số quan trọng về hiệu năng của CC. Kết quả được công bố trong công trình TCNN3 và được trình bày trong Chương 3 của luận án.

- Đề xuất một giải pháp AS tài nguyên hiệu quả trong CC: xây dựng một bộ AS sử dụng kết hợp kỹ thuật học tăng cường và điều khiển logic mờ. Kết quả được công bố trong các công trình TCNN2, HNQT3, HNTN1 và được trình bày trong Chương 4 của luận án.

5. BỐ CỤC CỦA LUẬN ÁN

Luận án được xây dựng thành bốn chương như sau: Chương 1 trình bày tổng quan về tự động điều chỉnh tài nguyên trong điện toán đám mây, giới thiệu tổng quan về AS trong CC, trình bày các khái niệm liên quan. Chương 2 trình bày các đề xuất cơ chế cơ bản có ảnh hưởng đến tự động điều chỉnh tài nguyên trong điện toán đám mây. Chương 3 trình bày đề xuất một mô hình mạng hàng đợi cho hệ thống điện toán đám mây. Chương 4 trình bày giải pháp tự động điều chỉnh tài nguyên cho ứng dụng đa tầng trên điện toán đám mây. Cuối cùng là một số kết luận và hướng phát triển của luận án.

CHƯƠNG 1 : TỔNG QUAN VỀ TỰ ĐỘNG ĐIỀU CHỈNH TÀI NGUYÊN TRONG ĐIỆN TOÁN ĐÁM MÂY

1.1. CƠ SỞ LÝ THUYẾT

1.1.1. Khái niệm về tự động điều chỉnh

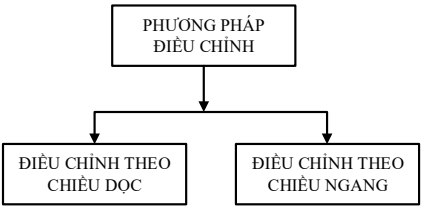
AS là cách để điều chỉnh tăng lên hoặc điều chỉnh giảm xuống một cách tự động số lượng tài nguyên tính toán đang được cấp phát cho các ứng dụng của khách hàng dựa vào nhu cầu của khách hàng ở bất kỳ thời điểm nào.

Theo quan điểm học thuật, AS là khả năng của cơ sở hạ tầng CC cho phép cung cấp tự động các tài nguyên ảo hóa. Tài nguyên được sử dụng bởi các ứng dụng dựa trên đám mây có thể được tự động tăng lên hoặc giảm xuống, do đó thích hợp cho việc sử dụng tài nguyên theo nhu cầu của ứng dụng.

1.1.2. Vòng lặp điều khiển MAPE

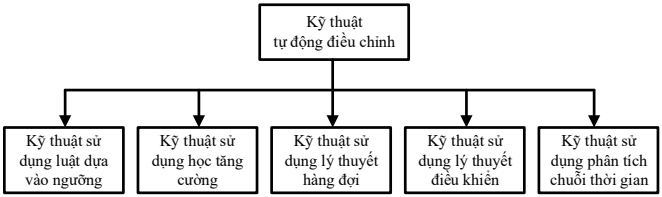
Đây là bài toán điều khiển tự động cổ điển, đòi hỏi một hệ thống AS các loại tài nguyên và số lượng tài nguyên được cấp phát để đạt được mục tiêu hiệu năng nhất định, được phản ánh trong SLA. Nó thường được thực hiện như một vòng lặp điều khiển MAPE. Chu kỳ kiểm soát liên tục lặp đi lặp lại theo thời gian. Các giai đoạn trong vòng lặp MAPE gồm: giai đoạn giám sát (M), giai đoạn phân tích (A), giai đoạn lập kế hoạch (P) và giai được thực hiện (E).

1.1.3. Phương pháp điều chỉnh



Hình 1.1: Phương pháp điều chỉnh trong CC

1.1.4. Kỹ thuật tự động điều chỉnh

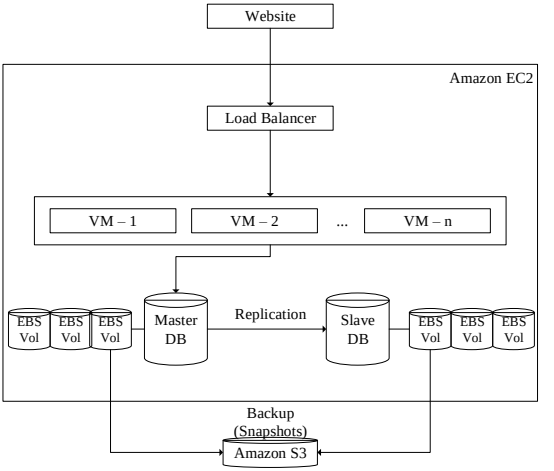


Hình 1.2: Phân loại các kỹ thuật AS

1.1.5. Kiến trúc ứng dụng đa tầng

Các ứng dụng đa tầng bao gồm các tầng kết nối tuần tự. Tại mỗi tầng, các yêu cầu dựa vào tầng trước để hoàn thành xử lý hoặc nó được chuyển đến tầng tiếp theo để xử lý và cuối cùng, kết quả xử lý chuyển đến người dùng.

Một kiến trúc thích nghi mở rộng thuộc loại này thường bao gồm ba tầng: một tầng trước, một tầng logic ứng dụng và một tầng cơ sở dữ liệu (ví dụ hình 1.6). Tầng cơ sở dữ liệu thường được xem xét là ít điều chỉnh và thường bỏ qua tính năng AS.



Hình 1.3: Ví dụ một ứng dụng web có kiến trúc ba tầng

Trong luận án này nghiên cứu các kỹ thuật AS theo chiều ngang, tức là AS số lượng VM tăng lên hoặc giảm xuống theo nhu cầu của tải công việc làm sao tối thiểu việc sử dụng các tài nguyên tính toán. Đối với ứng dụng đa tầng, mỗi tầng có thể sử dụng nhiều VM.

1.1.6. Nền tảng thực nghiệm

Các kỹ thuật AS hiện có thường được thực nghiệm trong các điều kiện rất khác nhau, do đó khó có được một đánh giá so sánh công bằng. Các nhà nghiên cứu trong lĩnh vực này đã tự xây dựng các nền tảng đánh giá riêng, phù hợp với nhu cầu riêng. Các nền tảng đánh giá này có thể được phân loại thành các chương trình mô phỏng,

các bộ dữ liệu mẫu tùy chỉnh và các nhà cung cấp đám mây công cộng. Ngoại trừ các chương trình mô phỏng đơn giản, một tiêu chuẩn ứng dụng có thể mở rộng phải được thực hiện trên nền tảng để thực hiện đánh giá.

Một số công cụ mô phỏng đám mây dùng cho hướng nghiên cứu là CloudSim, GreenCloud và GroudSim.

1.1.7. Tải công việc

Tải công việc đầu vào cho ứng dụng có thể là dữ liệu (tải công việc) tổng hợp, được tạo bằng các chương trình cụ thể hoặc thu được từ người dùng thực tế.

Hệ thống dựa trên đám mây xử lý hai loại tải công việc chính: lô (batch) và giao tác. Tải công việc theo lô bao gồm các công việc tùy ý, kéo dài, các công việc đòi hỏi nhiều tài nguyên, như các chương trình khoa học hoặc chuyển mã video. Ví dụ về tải công việc giao tác là các ứng dụng web được xây dựng để phục vụ các khách hàng HTTP trực tuyến. Các hệ thống này thường phục vụ các loại nội dung như các trang HTML, hình ảnh hoặc luồng video. Có thể được lưu trữ tĩnh hoặc được hiển thị tự động bởi các máy chủ.

1.2. TỔNG QUAN VỀ CÁC CÔNG TRÌNH LIÊN QUAN

Phần này sẽ trình bày tổng quan về các công trình liên quan đến các các kỹ thuật tự động điều chỉnh trong CC, phân tích những ưu điểm và hạn chế của chúng nhằm đề xuất hướng nghiên cứu tiếp theo.

KẾT LUẬN CHƯƠNG 1:

Chương 1 đã trình bày về cơ sở lý thuyết được sử dụng trong luận án như khái niệm về AS và những vấn đề liên quan. Các công trình liên quan đến luận án được trình bày theo từng kỹ thuật AS. Trong đó, luận án tập trung vào các kỹ thuật sử dụng lý thuyết hàng đợi, lý thuyết điều khiển và học tăng cường.

CHƯƠNG 2 : CÁC ĐỀ XUẤT CƠ CHẾ CƠ BẢN CÓ ẢNH HƯỞNG ĐẾN TỰ ĐỘNG ĐIỀU CHỈNH TÀI NGUYÊN TRONG ĐIỆN TOÁN ĐÁM MÂY

2.1. ĐẶT VẤN ĐỀ

Chương này được tổng hợp từ các công trình TCNN1, TCTN1, HNQT1, HNQT2.

2.2. ĐỀ XUẤT MỘT KỸ THUẬT CÂN BẰNG TẢI ĐỘNG HIỆU QUẢ TRONG ĐIỆN TOÁN ĐÁM MÂY

2.2.1. Cơ chế chia sẻ tài nguyên trong điện toán đám mây

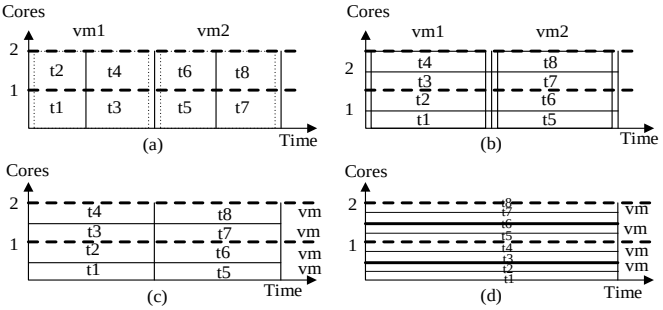
Kỹ thuật cân bằng tải trong CC có thể áp dụng ở những mức khác nhau, đó là mức PM hoặc mức VM. Có hai chính sách lập lịch áp dụng ở mức PM để thực hiện cung cấp các VM là time-shared (TS) hoặc space-share (SS) và cũng có hai chính

sách lập lịch áp dụng ở mức VM để thực hiện cung cấp cho các tác vụ là TS hoặc SS.

Hình 2.2 là một ví dụ về việc chia sẻ tài nguyên dựa vào các chính sách khác nhau ở các 2 mức: Trong ví dụ này, một PM có hai core CPU nhận yêu cầu lưu giữ hai VM. Mỗi VM yêu cầu hai core và có kế hoạch lưu giữ bốn đơn vị tác vụ. Cụ thể, các tác vụ t1, t2, t3, t4 được lưu giữ trong VM_1, và các tác vụ t5, t6, t7 và t8 được lưu giữ trong VM_2. Mỗi tác vụ yêu cầu một core xử lý.

Hình 2.2(a) trình bày một trường hợp cung cấp, trong đó chính sách SS được áp dụng cho việc phân bổ cho cả VM và các đơn vị tác vụ.

Hình 2.2(b), chính sách SS được áp dụng để phân bổ VM tới các PM và chính sách TS để phân bổ các đơn vị tác vụ tới core xử lý bên trong một VM.



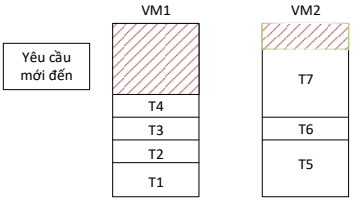
Hình 2.1: Ảnh hưởng của các chính sách khác nhau đối với việc thực thi các tác vụ

Hình 2.2 (c), VM sử dụng chính sách cung cấp TS, trong khi các tác vụ được cung cấp dựa trên chính sách SS.

Hình 2.2(d), chính sách phân bổ TS áp dụng cho cả VM và tác vụ.

2.2.2. Phân tích kỹ thuật cân bằng tải động được đề xuất

Trong công trình[1] đưa ra thuật toán cân bằng tải giám sát tích cực (gọi tắt là: AMLB-Old). Thuật toán AMLB-Old chưa tính đến kích thước của tải công việc và khả năng xử lý còn lại của các VM.



Hình 2.2: Ví dụ có 2 VM đang hoạt động như trên

[1] James, Jasmin and Verma, Bhupendra (2012), "Efficient VM load balancing algorithm for a cloud computing environment", *International Journal on Computer Science and Engineering*. 4(9), p. 1658.

Xem ví dụ Hình 2.2, với hai VM: VM_1, VM_2. Hiện tại, VM_1 chứa 4 yêu cầu T1, T2, T3, T4 và VM2 chứa 3 yêu cầu T5, T6, T7. VM1 chứa số yêu cầu nhiều hơn VM_2, nhưng công suất xử lý còn lại của VM_2 lại ít hơn VM_1. Khi một yêu cầu mới đến, theo thuật toán AMLB-Old, yêu cầu mới này sẽ được cấp phát cho VM_2 để xử lý. Do đó, VM_2 có thể bị quá tải, mặc dù số lượng yêu cầu của nó là ít hơn.

Thuật toán AMLB-New có tính đến cả sức mạnh xử lý hiện tại của VM và kích thước của các yêu cầu được giao.

Thời gian đáp ứng dự kiến có thể được xác định theo công thức sau:

$$\text{Response Time} = \text{Fin}_t - \text{Arr}_t + \text{TDelay}, \tag{0.1}$$

trong đó, Arr_t là thời điểm đến của yêu cầu; Fin_t là thời điểm hoàn thành yêu cầu (tác vụ) người dùng và TDelay là độ trễ truyền tải.

+ Nếu chính sách lập lịch là SS-SS hoặc TS-SS, Fin_t có thể tính như sau:

$$\text{Fin}_t = \text{est(p)} + \frac{\text{rl}}{\text{Capacity} \times \text{cores(p)}}. \tag{0.2}$$

+ Nếu chính sách lập lịch là SS-TS hoặc TS-TS, Fin_t có thể tính như sau:

$$\text{Fin}_t = \text{ct} + \frac{\text{rl}}{\text{Capacity} \times \text{cores(p)}}. \tag{0.3}$$

Thời gian thực hiện của một tác vụ p có thể tính như sau:

$$\text{Execution Time} = \frac{\text{rl}}{\text{Capacity} \times \text{Cores(p)}}. \tag{0.4}$$

trong đó, est(p) là thời gian mà tác vụ p được bắt đầu; ct là thời gian mô phỏng hiện tại; rl là tổng số chỉ thị của tác vụ p (đơn vị tính MI); Cores(p) là số core hoặc các thành phần xử lý theo yêu cầu của tác vụ (đơn vị là MIPS); Capacity là công suất xử lý trung bình (tính theo MIPS) của một core cho tác vụ.

Tham số Capacity xác định công suất thực sự cho việc xử lý các yêu cầu (tác vụ) trên mỗi VM. Ở đây, đề xuất các công thức để tính toán Capacity trong từng trường hợp để sử dụng trong thuật toán AMLB-New:

(1) Chính sách lập lịch VM là SS, chính sách lập lịch tác vụ là SS:

$$\text{Capacity} = \sum_{\text{i=1}}^{\text{np}} \frac{\text{Cap(i)}}{\text{np}}, \tag{0.5}$$

trong đó, Cap(i) là sức mạnh xử lý của core thứ i, np là số lượng core thực có của PM đã được xem xét.

(2) Chính sách lập lịch VM là SS, chính sách lập lịch tác vụ là TS:

$$\text{Capacity} = \frac{\sum_{i=1}^{np} \text{Cap(i)}}{\text{Max}(\sum_{j=1}^{\alpha} \text{Cores(j)}, np)}, \quad (0.6)$$

trong đó, Cores(j) là số core mà tác vụ j yêu cầu. α là tổng số tác vụ trong VM có chứa tác vụ.

(3) Chính sách lập lịch VM là TS, chính sách lập lịch tác vụ là SS:

$$\text{Capacity} = \frac{\sum_{i=1}^{np} \text{Cap(i)}}{\text{Max}(\sum_{k=1}^{\beta} \sum_{j=1}^{\gamma} \text{Cores(j)}, np)}, \quad (0.7)$$

trong đó, β là số VM trong PM hiện tại. γ là số lượng các tác vụ đang chạy đồng thời trong VM thứ k.

(4) Chính sách lập lịch VM là TS, chính sách lập lịch tác vụ là TS:

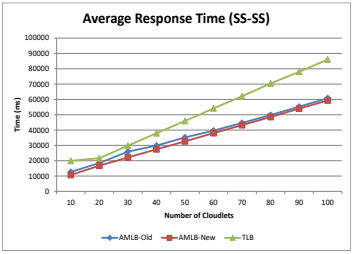
$$\text{Capacity} = \frac{\sum_{i=1}^{np} \text{Cap(i)}}{\text{Max}(\sum_{j=1}^{\delta} \text{Cores(j)}, np)}, \quad (0.8)$$

trong đó, δ là tổng số tác vụ của PM được xem xét.

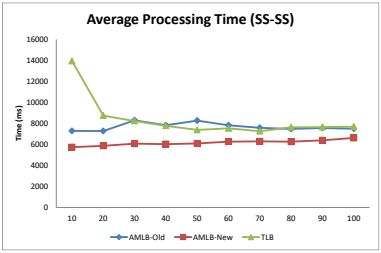
Ở đây chỉ xét các thuật toán cân bằng tải thực hiện trong một trong tâm dữ liệu, nên tham số độ trễ truyền dữ liệu có thể bỏ qua ($\text{TDelay} = 0$).

2.2.3. Thực nghiệm và đánh giá

Thực nghiệm so sánh ba thuật toán AMLB-Old, Thuật toán cân bằng tải Throttled (TLB) và Thuật toán đề xuất (AMLB-New). Việc đánh giá dựa trên hai số đo: thời gian đáp ứng trung bình và thời gian xử lý trung bình.



Hình 2.3: Thời gian đáp ứng trung bình theo trường hợp 1

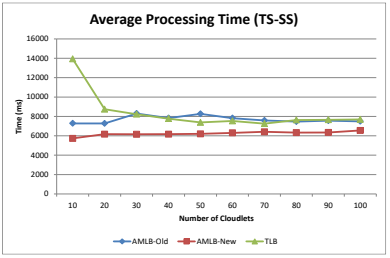


Hình 2.4: Thời gian xử lý trung bình theo trường hợp1

a. Trường hợp 1: Ở mức PM và mức VM được cung cấp theo chính sách lập lịch SS, gọi là SS-SS.

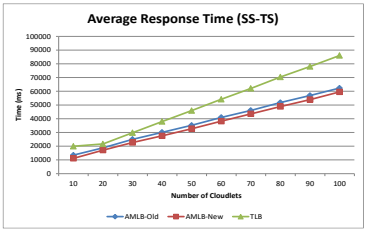
b. Trường hợp 2: Ở mức PM, chính sách lập lịch VM là TS và ở mức VM, chính sách lập lịch tác vụ là SS, gọi là TS-SS.

Hình 2.5: Thời gian đáp ứng trung bình theo trường hợp 2

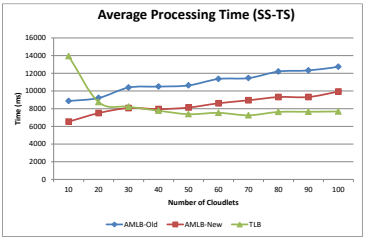


Hình 2.6: Thời gian xử lý trung bình theo trường hợp 2

c. Trường hợp 3: Ở mức PM, chính sách lập lịch cho VM là SS và ở mức VM, chính sách lập lịch tác vụ là TS, gọi là SS-TS.

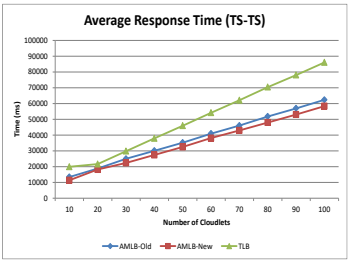


Hình 2.7: Thời gian đáp ứng trung bình theo trường hợp 3

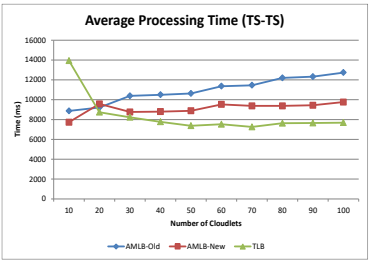


Hình 2.8: Thời gian xử lý trung bình theo trường hợp 3

d. Trường hợp 4: Ở mức PMvà mức VM được cung cấp theo chính sách lập lịch TS, gọi là TS-TS.



Hình 2.9: Thời gian đáp ứng trung bình theo trường hợp 4



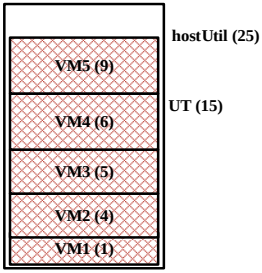
Hình 2.10: Thời gian xử lý trung bình theo trường hợp 4

Trong cả bốn trường hợp trên, thời gian đáp ứng trung bình của thuật toán AMLB-New tốt nhất so với các thuật toán AMLB-Old và TLB. Về thời gian xử lý trung bình, thuật toán AMLB-New là tốt nhất cho hai trường hợp SS-SS và TS-SS. Trong trường hợp SS-TS và TS-TS, thuật toán AMLB-New vượt trội hơn thuật toán AMLB-Old nhưng không tốt bằng thuật toán TLB.

2.3. ĐỀ XUẤT MỘT KỸ THUẬT DI TRÚ HIỆU QUẢ TRONG ĐIỆN TOÁN ĐÁM MÂY

2.3.1. Phân tích kỹ thuật di trú được đề xuất

Thuật toán MM-Old[2] có ưu điểm là số lần di trú là ít nhất. Tuy nhiên, trong trường hợp phải di trú nhiều hơn một VM, thì thuật toán MM-Old không khai thác được tối đa tài nguyên của PM (duy trì PM hoạt động với khả năng cao nhất có thể) trong một số trường hợp. Hãy xét ví dụ hình 2.15.



Hình 2.11: Ví dụ cho trường hợp phải di trú nhiều VM

Hình 2.15 mức sử dụng hiện tại của PM là $hostUtil = 25$, ngưỡng trên là $UT = 15$. PM này được cấp phát 5 VM với các mức sử dụng tương ứng như trên. Như vậy, PM này đang hoạt động quá tải, độ chênh lệch mức sử dụng vượt là $diffUtil = 25 - 15 = 10$.

Theo thuật toán MM-Old, thì VM5 (9) sẽ được chọn đầu tiên để di trú là. Sau đó, VM2 (4) sẽ được chọn tiếp theo để di trú và sẽ đưa PM về mức sử dụng xuống dưới ngưỡng UT. Do đó, sau khi di trú, mức sử dụng CPU của PM là: $25 - 13 = 12$.

Tuy nhiên, có thể thấy nếu chọn hai máy ảo VM4 (6) và VM3(5) để di trú thì vẫn thỏa mãn số lần di trú ít nhất đồng thời có thể duy trì mức hoạt động của PM cao nhất có thể. Mức sử dụng của PM khi di trú VM4 và VM5 là: $25 - 11 = 14$.

Luận án đề xuất chính sách di trú với số lần ít nhất (MM-New) có thể phát biểu lại như sau:

Cho V_j là một tập hợp các VM được cấp phát cho PM j. Gọi $P(V_j)$ là tập công suất (mức sử dụng) của V_j. Chính sách MM-New tìm một tập $R \in V_j$ được định nghĩa như sau:

[2] Beloglazov, Anton, Abawajy, Jemal, and Buyya, Rajkumar (2012), "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing", *Future generation computer systems*. 28(5).

$$R = \begin{cases} \left\{ \begin{array}{c} S|S \in V_j, \text{hostUtil}_j - \sum_{v \in S, u_a(v) \in P(V_j)} u_a(v) < UT, \\ |S| \to \min, \sum_{v \in S, u_a(v) \in P(V_j)} u_a(v) \to \min \end{array} \right\}, \text{nếu hostUtil}_j > UT; \\ V_j, \text{nếu hostUtil}_j < LT; \\ \emptyset, \text{còn lại.} \end{cases} \quad (0.9)$$

trong đó UT là ngưỡng sử dụng trên; LT là ngưỡng sử dụng dưới; hostUtil_j là mức sử dụng CPU hiện tại của PM j; và $u_a(v)$ là một phần mức sử dụng CPU được cấp phát cho VM v.

2.3.2. Thực nghiệm và đánh giá

Thực nghiệm tiến hành so sánh thuật toán đề xuất MM-New với thuật toán MM-Old và cho kết quả như hình 2.16.

Hình 2.12: Kết quả so sánh hài thuật toán MM-Old và MM-New

Kết quả cho thấy, thuật toán MM-New sẽ tốt hơn thuật toán MM-Old trong trường hợp phải di trú nhiều VM.

2.4. ẢNH HƯỞNG CỦA CÂN BẰNG TẢI ĐẾN CƠ CHẾ TỰ ĐỘNG ĐIỀU CHỈNH TÀI NGUYÊN

2.4.1. Mô hình trung tâm dữ liệu có bộ điều chỉnh tài nguyên tự động

Cho trung tâm dữ liệu (DC) có N VM: $VM_1, VM_2, \ldots, VM_N$. Trong đó, mỗi VM có sức mạnh xử lý P_i và tải công việc $L_i(t)$ tại thời điểm t.

$$VM_1(L_1(t),\ P_1)$$
$$VM_2(L_2(t),\ P_2)$$
$$\ldots\ldots$$
$$VM_N(L_N(t),\ P_N)$$

Gọi $l(t)$ là tải công việc mới (yêu cầu mới) đến tại thời điểm t. Gọi $L'_i(t)$ là tải công việc của VM_i sau khi bộ cân bằng tải đã phân bố tải $l(t)$ đến:

$$L_i'(t) = L_i(t) + a_i(t).l(t);\ i = 1..N \tag{0.10}$$

với $a_i(t)$ lấy giá trị 1 hoặc 0. Khi sử dụng thuật toán Round Robin:

$$a_i(t) = \begin{cases} 0 \text{ nếu } VM_i \text{ không được gán tải,} \\ 1 \text{ nếu } VM_i \text{ được gán.} \end{cases} \tag{0.11}$$

Với thuật toán cân bằng tải giám sát tích cực (ALMB-New), $a_i(t)$ như sau:

$$a_i(t) = \begin{cases} 1 \text{ nếu } VM_i \text{ có thời gian hoàn thành} \\ \text{dự kiến sớm nhất được gán tải,} \\ 0 \text{ nếu ngược lại.} \end{cases} \tag{0.12}$$

Thời gian phục vụ còn lại của VM, được gọi là $T_{left,i}(t)$, được tính như sau:

$$T_{left,i}(t) = \frac{L_i(t)}{P_i} \tag{0.13}$$

Gọi $Dev(t)$ là sự khác biệt giữa thời gian phục vụ còn lại nhỏ nhất và thời gian phục vụ còn lại lớn nhất của các VM bên trong hệ thống được xem xét.

$$Dev(t) = \left(\frac{L_i(t)}{P_i}\right)_{max} - \left(\frac{L_i(t)}{P_i}\right)_{min} \tag{0.14}$$

Với trường hợp lý tưởng là: $Dev(t) \to 0\ khi\ t \to \infty$

Gọi $T_{av}(t)$ thời gian phục vụ còn lại trung bình tại thời điểm t, được tính như sau:

$$T_{av}(t) = \frac{\sum_i^N L_i(t)}{\sum_i^N P_i} \tag{0.15}$$

Nếu gọi $N'(t)$ là số lượng máy trong hệ thống được xem xét tại thời điểm t thì $N'(t)$ được tính theo công thức sau:

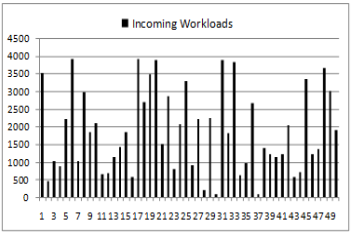
$$N'(t) = N(t) + n(t) \tag{0.16}$$

$n(t)$ là số máy tạm thời được bổ sung vào hệ thống được xem xét tại thời điểm t. Bộ AS tuân theo quy tắc sau:

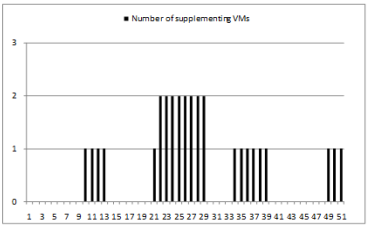
$$n(t) = \begin{cases} 0\ Threshold_{low} \le T_{av}(t) \le Threshold_{high}, \\ 1\ T_{av}(t) > Threshold_{high}, \\ -1\ T_{av}(t) < Threshold_{low}. \end{cases} \tag{0.17}$$

2.4.2. Thực nghiệm và đánh giá

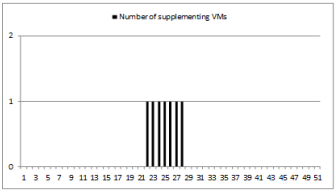
Thực nghiệm sử dụng dữ liệu tải đầu vào như hình 2.17. So sánh việc sử dụng các thuật toán cân bằng tải: Round Robin và kỹ thuật cân bằng tải giám sát tích cực AMLB-New. Việc mô phỏng bao gồm một bộ AS hoạt động dựa trên ngưỡng.



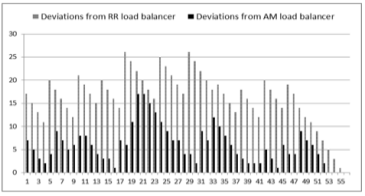
Hình 2.13: Phấn bố tải công việc đến trong ClarkNet trace[3]



Hình 2.14: Số lượng VM được bổ sung khi sử dụng Round Robin



Hình 2.15: Số lượng VM được bổ sung khi sử dụng AMLB-New



Hình 2.16: Độ lệch Dev(t) theo bộ cân bằng tải sử dụng Round Robin và sử dụng AMLB-New

Kết quả cho thấy việc bổ sung VM và duy trì hiệu suất của bộ AS khi sử dụng kỹ thuật cân bằng tải AMLB-New tốt hơn khi sử dụng Round Robin.

KẾT LUẬN CHƯƠNG 2:

Chương này, luận án đã đề xuất một kỹ thuật cân bằng tải động hiệu quả và một kỹ thuật di trú hiệu quả trong CC. Cuối cùng là đánh giá ảnh hưởng của các kỹ thuật cân bằng tải trong CC có cơ chế AS tài nguyên, các kết quả thực hiện và mô phỏng đã chỉ ra tính hiệu quả của các kỹ thuật được đề xuất và đánh giá được vai trò, sự ảnh hưởng của các kỹ thuật cân bằng tải trong một trường CC có cơ chế AS.

CHƯƠNG 3 : ĐỀ XUẤT MỘT MÔ HÌNH MẠNG HÀNG ĐỢI CHO HỆ THỐNG ĐIỆN TOÁN ĐÁM MÂY

3.1. ĐẶT VẤN ĐỀ

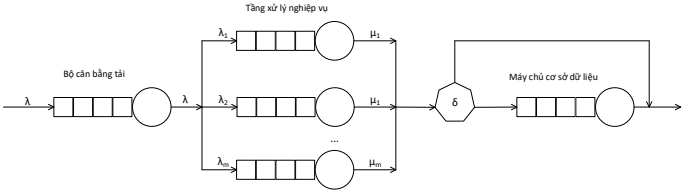
Nội dung chương này được tổng hợp từ công trình TCNN3.

3.2. MÔ HÌNH HÓA ỨNG DỤNG ĐA TẦNG TRÊN ĐIỆN TOÁN ĐÁM MÂY

3.2.1. Mô hình hàng đợi mở cho ứng dụng đa tầng

Hình 3.1 là mô hình mạng hàng đợi mở được đề xuất cho ứng dụng đa tầng.

[3] *HTTP, ClarkNet ClarkNet HTTP Trace, truy cập 9/7/2018, from http://ita.ee.lbl.gov/html/contrib/ClarkNet-HTTP.html.*



Hình 3.1: Mô hình mạng hàng đợi mở cho ứng dụng đa tầng trên CC

Ở đây tập trung vào tính không đồng nhất của VM, giả sử mỗi VM i có c_i core xử lý, tốc độ thực thi của mỗi core là s đo bằng giga lệnh trên mỗi giây (GIPS).

Tốc độ phục vụ của VM i được tính như sau:

$$\mu_i = \frac{s * c_i}{\bar{Z}}, \tag{0.18}$$

trong đó, $\bar{Z}$ là số chỉ thị trung bình của một yêu cầu được thi tại VM i.

Mỗi nút là một VM có tốc độ phụ vụ μ_i.

Mô hình đề xuất trong hình 3.1 thỏa mãn yêu cầu của một mạng hàng đợi Jackson mở.

3.2.2. Phân tích mô hình đề xuất

Gọi r_{ij} là xác suất chuyển trạng thái sau khi yêu cầu được xử lý xong tại nút i sẽ chuyển đến nút j. Ta có, tổng các xác suất chuyển trạng thái trong mạng hàng đợi mở là 1 (ở đây: r_{i0} là xác suất chuyển ra ngoài mạng từ nút i):

$$r_{i0} = 1 - \sum_{j=1, j\neq i}^{N} r_{ij}, i \in [1, N] \tag{0.19}$$

Gọi λ_i là tốc độ đến từ bên ngoài vào nút thứ i và Λ_i là tổng tốc độ đến của nút thứ i (bao gồm cả tốc độ đến từ bên ngoài và tốc độ chuyển đến từ các nút bên trong hệ thống mạng đến nút thứ i), khi đó ta có:

$$\Lambda_i = \lambda_i + \sum_{j=1, j\neq i}^{N} \Lambda_j r_{ji}, i \in [1, N] \tag{0.20}$$

Gọi $\Lambda = [\Lambda_1, \Lambda_2, \dots, \Lambda_N]^T$ là vector tổng tốc độ đến các hàng đợi của các nút. Gọi $\lambda = [\lambda_1, \lambda_2, \dots, \lambda_N]^T$ là vector tốc độ đến từ bên ngoài, khi đó công thức (3.13) có thể được viết lại dưới dạng phương trình ma trận sau:

$$\Lambda = \lambda + R^T \Lambda, \tag{0.21}$$

với R là ma trận xác suất chuyển trạng thái (R^T là ma trận chuyển vị của ma trận R).

$$R = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1N} \\ r_{21} & r_{12} & \cdots & r_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ r_{N1} & r_{N1} & \cdots & r_{NN} \end{pmatrix} \quad (0.22)$$

Một mạng Jackson mở được xem xét như một chuỗi Markov thời gian liên tục với vector trạng thái:

$$\bar{x} = (x_1, x_2, \dots, x_N), \quad (0.23)$$

trong đó với x_i là số lượng yêu cầu đang có tại VM i. Sử dụng phương trình trạng thái cân bằng dựa trên hệ thống Markov.

Trong Bảng 3.2, mô tả các trạng thái có thể:

Bảng 3.1. Bảng mô tả trạng thái

Trạng thái	Ký hiệu
$x_1, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_N$	$\bar{x}$
$x_1, \dots, x_{i-1}, x_i + 1, x_{i+1}, \dots, x_N$	$\bar{x}; i^+$
$x_1, \dots, x_{i-1}, x_i - 1, x_{i+1}, \dots, x_N$	$\bar{x}; i^-$
$x_1, \dots, x_{i-1}, x_i + 1, x_{i+1}, \dots, x_{j-1}, x_j - 1, x_{j+1}, \dots, x_N$	$\bar{x}; i^+j^-$

Sử dụng nguyên lý cân bằng trạng thái luồng vào trạng thái $\bar{x}$ bằng luồng ra khỏi trạng thái $\bar{x}$, giả sử $x_i \geq 1$ tại mọi VM:

$$\sum_{i=1}^{N} \lambda_i \, p_{\bar{x};i^-} + \sum_{j=1}^{N} \sum_{i=1}^{N} \mu_i r_{ij} p_{\bar{x};i^+j^-} + \sum_{i=1}^{N} \mu_i r_{i0} p_{\bar{x};i^+}$$

$$= \sum_{i=1}^{N} \mu_i (1 - r_{ii}) \, p_{\bar{x}} + \sum_{i=1}^{N} \lambda_i \, p_{\bar{x}} \quad (0.24)$$

Định lý: Phân bố cân bằng cho mạng Jackson mở là:

$$p_{\bar{x}} = C \prod_{i=1}^{N} \rho_i^{x_i}, \text{với } \rho_i = \frac{\Lambda_i}{\mu_i} < 1.$$

Việc cung cấp phân bố chung cho tất cả VM trong hệ thống, giải pháp trạng thái ổn định cho phương trình (3.17) là:

$$p_{\bar{x}} \equiv p_{x_1, x_2, \dots, x_N} = (1 - \rho_1)\rho_1^{x_1}(1 - \rho_2)\rho_2^{x_2} \dots (1 - \rho_N)\rho_N^{x_N}$$

$$= \prod_{i=1}^{N} (1 - \rho_i)\rho_i^{x_i}. \quad (0.25)$$

$$\text{với } C = \prod_{i=1}^{N} (1 - \rho_i).$$

Số lượng yêu cầu trung bình L_i tại mỗi VM i cho hàng đợi M/M/1 với tổng tốc độ đến Λ_i:

$$L_i = \frac{\rho_i}{1-\rho_i}, i = 1,2,\dots,N \quad (0.26)$$

Tổng số lượng yêu cầu trung bình của toàn mạng được tính như sau:

$$L = L_1 + L_2 + \cdots + L_N = \sum_{i=1}^{N} \frac{\rho_i}{1-\rho_i}. \quad (0.27)$$

Thời gian đợi trung bình của yêu cầu trong mạng với Luật của Little:

$$W = \frac{L}{\beta}, \quad (0.28)$$

trong đó $\beta = \sum_{i=1}^{N} \lambda_i$ là tổng tốc độ đến từ bên ngoài vào mạng. Thời gian đáp ứng trung bình của một yêu cầu trong mạng tại mỗi VM thứ i ($i = 1,2,\dots,N$):

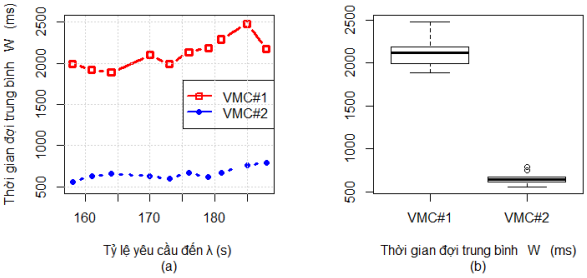
$$W_i = \frac{L_i}{\Lambda_i} = \frac{1}{\mu_i(1-\rho_i)}, \quad (0.29)$$

trong đó Λ_i được tính theo công thức (3.4) và $W \neq W_1 + W_2 + \cdots + W_N$.

3.3. THỰC NGHIỆM VÀ ĐÁNH GIÁ

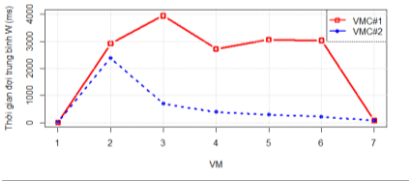
Kết quả thực nghiệm được tiến hành với 2 cụm VM phục vụ yêu cầu từ khách hàng sử dụng dịch vụ web (cụm VMC#1 gồm 7 máy có cấu hình CPU đồng nhất, cụm VMC#2 gồm 7 máy có cấu hình CPU không đồng nhất).

Thực nghiệm 1 nhằm đánh giá các tham số trong mô hình, cho số lượng yêu cầu đến là 1000, γ= {158,161,164,167,170,173,176,179,181,185, 188} tốc độ xử lý (phục vụ) của bộ cân bằng tải là $\mu_{LB} = \frac{\gamma}{\rho_{LB}}$.



Hình 3.2: Thời gian đợi trung bình của cụm VM cho thực nghiệm 1

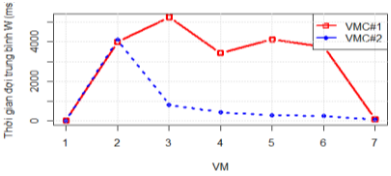
Thực nghiệm 2 với $\lambda = 185$, được thực hiện 30 lần và tính thời gian đợi trung bình $\overline{W}$. Sau đó, tính toán độ tin cậy 95% cho dữ liệu thực nghiệm và dữ liệu tính toán từ mô hình.



Hình 3.3: Thời gian đợi trung bình của các VM cho thực nghiệm 2

Kết quả ở Hình 3.3 cho thấy thời gian đáp ứng trung bình của VM1 và VM7 là rất nhỏ ở cả hai cụm VMC#1 và VMC#2.

Thực nghiệm 3 nhằm phân tích mối quan hệ của thông số trong mô hình với tốc độ đến trung bình $\lambda = 185$, điều chỉnh mức độ sử dụng tài nguyên trung bình ρ_i của các VM ở tầng xử lý nghiệp vụ của ứng dụng từ 0.71 đến 0.79. Kết quả thực nghiệm được thể hiện trong bảng 3.2 và hình 3.4.



Hình 3.4: Thời gian đợi trung bình của các VM cho thực nghiệm 3

Bảng 3.2. Khoảng tin cậy 95% của thời gian đợi trung bình của mỗi VM trong từng cụm

Thời gian đợi trung bình $\overline{W}$					
VM	ρ_i	**VMC**	**Trung bình**	**Cận dưới**	**Cận trên**
1	0.8	VMC#1	6.39069	6.20901	6.57238
		VMC#2	6.37414	6.16965	6.57863
2	0.71	VMC#1	3987.14	3126.29	4848.00
		VMC#2	4115.76	3241.04	4990.49
3	0.73	VMC#1	5256.16	4211.63	6300.7
		VMC#2	803.147	701.051	905.243
4	0.75	VMC#1	3430.45	2538.11	4322.80
		VMC#2	434.463	401.364	467.562
5	0.77	VMC#1	4127.60	3154.68	5100.51
		VMC#2	277.476	249.991	304.960
6	0.79	VMC#1	3756.96	2980.27	4533.66
		VMC#2	236.522	219.641	253.403
7	0.81	VMC#1	73.9026	68.5611	79.2442
		VMC#2	69.6167	65.9044	73.3289

KẾT LUẬN CHƯƠNG 3

Chương này đã đề xuất được một mô hình CC sử dụng mạng Jackson mở, nhằm đánh giá các số đo hiệu năng của hệ thống. Với kết quả thực nghiệm, bước đầu đã đánh giá được sự đúng đắn của mô hình, đưa ra các số đo của hệ thống có thể tin cậy.

CHƯƠNG 4 : GIẢI PHÁP TỰ ĐỘNG ĐIỀU CHỈNH TÀI NGUYÊN CHO ỨNG DỤNG ĐA TẦNG TRÊN ĐIỆN TOÁN ĐÁM MÂY

4.1. ĐẶT VẤN ĐỀ

Chương này được tổng hợp từ các công trình TCNN2, HNQT3 và HNTN1.

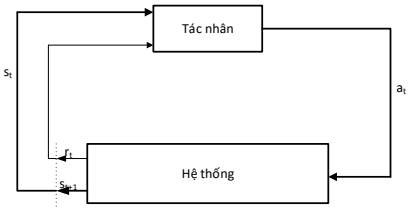
4.2. KIẾN TRÚC ỨNG DỤNG ĐA TẦNG

Phần này nghiên cứu kiến trúc ứng dụng đa tầng có dạng như trong hình 1.6 đã được trình bày trong mục 1.1.5 của Chương 1.

4.3. ĐỀ XUẤT MÔ HÌNH TỰ ĐỘNG ĐIỀU CHỈNH

4.3.1. Phân tích mô hình

Việc ra quyết định của bộ AS có thể được xem xét như một quá trình quyết định Markov (MDP).



Hình 4.1: Mô hình tương tác giữa tác nhân và hệ thống

MDP biểu diễn sự tương tác giữa hệ thống và tác nhân. Trong đó, tác nhân (bộ AS) đóng vai trò ra quyết định khi nhận được thông tin phản hồi từ hệ thống.

Tại thời điểm t, ta có:

- Trạng thái hệ thống được ký hiệu là $s_t \in S$ với S là tập trạng thái của hệ thống có thể có tại thời điểm khác nhau,

- Thông qua trạng thái hiện tại s_t của hệ thống, tác nhân có thể ra quyết định thực hiện một hành động $a_t \in A(s_t)$ với xác suất π_t,

- Tác nhân nhận giá trị phần thưởng $r_t \in R$ tại thời điểm $t + 1$ với hệ thống chuyển sang trạng thái s_{t+1}.

Ánh xạ $\pi_t: S \rightarrow A$ gọi là chiến lược của tác nhân.

Sau khi thực hiện các hành động, hệ thống nhận được giá trị phản hồi R_t tổng cộng. Hàm giá trị trạng thái của hệ thống được tính bằng kỳ vọng toán học của hàm phản hồi R_t theo thời gian.

$$V^{\pi}(s) = E_{\pi}\{R_t | s_t = s\} \quad (0.30)$$

Chính sách như vậy gọi là chính sách tối ưu, kí hiệu là π^* và V^* là giá trị tối ưu.

$$\pi^* = \arg\max_{\pi}\{V^{\pi}(s)\} \quad (0.31)$$

$$V^*(s) = \max_{\pi}\{V^{\pi}(s)\} \quad (0.32)$$

Giả sử có n ứng dụng. Mỗi ứng dụng j được thực thi trên một cụm k VM không đồng nhất. Mỗi VM i được biểu diễn dưới dạng vector $v_i = \{c_i, r_i, d_i\}\ \forall\ i \in [1, k]$, trong đó c_i là tham số CPU đã được sử dụng, r_i là lượng bộ nhớ RAM đã được sử dụng và d_i là dung lượng ổ đĩa đã được sử dụng của VM thứ *i* trong cụm thứ *j* tại thời điểm xét. Giả sử các tham số của VM được thu thập dưới dạng phần trăm.

Tài nguyên khả dụng của VM là tài nguyên sẵn sàng của VM có thể phục vụ cho các yêu cầu. Như vậy, tài nguyên khả dụng của mỗi VM *i* trong cụm thứ *j* là w_j^i được tính theo công thức sau:

$$w_j^i = \alpha_1(1 - c_i) + \alpha_2(1 - r_i) + \alpha_3(1 - d_i), \tag{0.33}$$

trong đó, α_1, α_2 và α_3 là các hằng số sao cho $\alpha_1 + \alpha_2 + \alpha_3 = 1$.

Tài nguyên khả dụng trung bình của cụm k VM dành cho ứng dụng j:

$$\overline{w} = \frac{1}{k}\sum_{i=1}^{k} w_j^i \tag{0.34}$$

Như vậy phương sai của k giá trị tài nguyên khả dụng của từng cụm k VM:

$$var = \frac{\sum_{i=1}^{k}(w_j^i - \overline{w})^2}{k} \tag{0.35}$$

Chi phí thuê dịch vụ được tính toán dựa trên số lượng tài nguyên đã được sử dụng theo thời gian. Gọi ω_t là số lượng CPU đã được cấp phát. Vì vậy, đơn giá thuê ω_t tài nguyên tại thời điểm t được biểu diễn như sau:

$$p(\omega_t) = a\omega_t + b\ (0 < \omega_t < c), \tag{0.36}$$

trong đó, a, b là hằng số, c là tổng số CPU của hệ thống. Chi phí sử dụng VM của ứng dụng j sử dụng trong thời gian t được tính như sau:

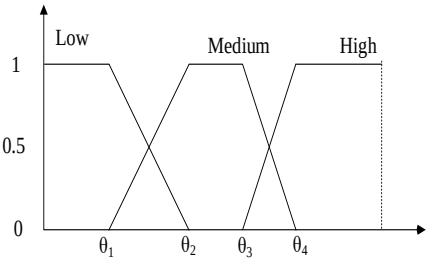
$$U_j(t) = 1 - \frac{vm_j(t) * p(\omega_t)}{cost_{max}}, \tag{0.37}$$

với $vm_j(t)$ là số lượng VM thuê trong thời gian t của ứng dụng j, $cost_{max}$ là chi phí tối đa của ứng dụng j. Chi phí sử dụng $U_j(t)$ sẽ là cơ sở cho việc tính toán giá trị phần thưởng trong thuật toán học tăng cường – học Q mờ.

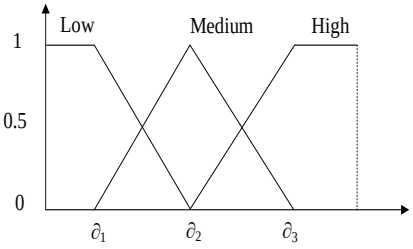
4.3.2. Bộ tự động điều chỉnh

4.3.2.1. Quá trình mờ hóa

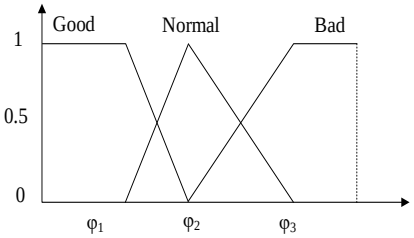
Các giá trị đầu vào cho bộ điều khiển AS: tài nguyên khả dụng trung bình ($\overline{w}$), phương sai của tài nguyên khả dụng (var) và thời gian đáp ứng trung bình (rt) được mờ hóa theo các hàm thành viên sau đây:



Hình 4.2: Hàm thành viên mờ cho tài nguyên khả dụng trung bình ($\overline{w}$)



Hình 4.3: Hàm thành viên mờ cho phương sai của tài nguyên khả dụng (var)



Hình 4.4: Hàm thành viên mờ cho thời gian đáp ứng trung bình (rt)

4.3.2.2. Quá trình lập luận xấp xỉ

Kết hợp hệ thống suy diễn mờ Takagi-Sugeno và thuật toán học Q, tập luật suy diễn mờ và cơ chế xấp xỉ đầu ra như sau:

Luật^1: **NẾU** x_1 là s_1 $\wedge$ ... $\wedge$ x_n là s_n **THÌ**

$y_1 = a_1^1$ với q[1,1] là mức độ phù hợp của lựa chọn đầu ra hoặc

$y_1 = a_2^1$ với q[1,2] là mức độ phù hợp của lựa chọn đầu ra hoặc

.....

$y_1 = a_m^1$ với q[1, m] là mức độ phù hợp của lựa chọn đầu ra,

.....

Luật^i: **NẾU** x_1 là s_1 $\wedge$ ... $\wedge$ x_n là s_n **THÌ**

$y_i = a_1^i$ với q[i, 1] là mức độ phù hợp của lựa chọn đầu ra hoặc

$y_i = a_2^i$ với q[i, 2] là mức độ phù hợp của lựa chọn đầu ra hoặc

.....

$y_i = a_m^i$ với q[i, m] là mức độ phù hợp của lựa chọn đầu ra.

trong đó,

- $x_j, j = 1, .., n$ là các biến đầu vào của hệ thống suy diễn mờ được định nghĩa trên không gian S,

- a_m^i là hành động được tác nhân lựa chọn cho tập luật thứ i được định nghĩa trên không gian hành động A được rời rạc hóa thành m đoạn,

- $q[i, m]$ là độ thích hợp khi lựa chọn giá trị đầu ra (hành động cục bộ) a_m^i khi luật $Rule^i$ được lựa chọn.

Thành phần học sử dụng kỹ thuật học Q. Tại mỗi vòng lặp, bộ điều khiển sẽ đưa ra một hành động a dựa vào trạng thái s được ký hiệu $Q(s, a) = q[i, m]$. Giá trị phản hồi này được biểu diễn dưới dạng hàm số đối với giá trị phần thưởng. Giả thiết rằng hàm phản hồi có tính chất cộng tính.

Thành phần điều khiển sẽ chọn hành động mà có giá trị phần thưởng tốt trong tương lai.

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}, \quad (0.38)$$

trong đó, γ là hệ số suy giảm thể hiện mức độ quan trọng của giá trị phần thưởng trong tương lai. Mỗi phần thưởng được tính toán dựa vào chi phí sử dụng VM như sau:

$$r_t = U(t) - U(t-1), \quad (0.39)$$

với $U(t)$ được tính theo công thức (4.7).

4.3.2.3. Quá trình giải mờ

Chính sách $\pi(x, a)$ là xác suất chọn hành động a từ trạng thái x, hàm hành động – giá trị Q(x,a) được tính theo công thức sau:

$$Q^{\pi} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}\right\} \quad (0.40)$$

trong đó, $E_{\pi}\{.\}$ là hàm kỳ vọng của chính sách π. Khi nào chính sách thích hợp được tìm thấy, khi đó vấn đề học tăng cường được giải quyết.

Theo cơ chế suy diễn của hệ thống Takagi-Sugeno, tác động đầu ra $a(x)$ đối với véctơ đầu vào $x = (x_1, x_2, \ldots, x_N)$, được xấp xỉ bằng công thức sau:

$$a(x) = \frac{\sum_i \alpha_i(x) a_{j*}^i}{\sum_i \alpha_i(x)}, \quad (0.41)$$

trong đó,

$\alpha_i(x) = \min(\mu_1^i(x_1), \mu_2^i(x_2), \ldots, \mu_N^i(x_N))$ là giá trị chân lý của mệnh đề $y = y^i$

a_{j*}^i tác động đầu ra được lựa chọn của luật $Rule^i$,

Quá trình học trong hệ thống là xấp xỉ các độ thích hợp với lựa chọn $q[i, j]$ của mỗi giá trị đầu ra a_j^i với từng véctơ đầu vào x. Do vậy, giá trị tương ứng của hàm giá trị trạng thái - hành động được xấp xỉ bằng:

$$Q(x,a) = \frac{\sum_i \alpha_i(x) q[i,j^*]}{\sum_i \alpha_i(x)}. \quad (0.42)$$

Hàm giá trị tại trạng thái x của hệ thống được tính như sau:

$$V(x) = \max_a Q(x,a) = \frac{\sum_i \alpha_i(x) \max_j q[i,j]}{\sum_i \alpha_i(x)}. \quad (0.43)$$

Trong quá trình học, hàm giá trị trạng thái-hành động Q và các giá trị của độ thích hợp của các luật $q[i,j^*]$ được hiệu chỉnh theo hệ thức lặp sau:

$$Q_t(x_t,a) \leftarrow Q_t(x_t,a) + \eta \Delta Q_t, \quad (0.44)$$

trong đó, η là tốc độ học và $\Delta Q_t = r_{t+1} + \gamma \max_{a'} Q(x_{t+1},a_t) - Q(x_t,a_t)$.

Bảng 4.1: Thuật toán tự động điều chỉnh (AS AFQL)

```
/*AFQL
* Đầu vào:  η, γ
* Đầu ra:
*Hành động a */
//khởi tạo giá trị q-values:
for (application app in applications)
    q[i,k] = 0;
//sinh luật điều khiển mờ
generateFuzzyRules()
while(q chưa hội tụ)
    //xấp xỉ giá trị Q cho từng ứng dụng ở trạng thái hiện tại x_t và hành động a
    for (application app in applications)
        Q(s(t),a) = Σ_i α_i(x_t) q[i,j*] / Σ_i α_i(x_t)
    //triển khai hành động a
    for (application app in applications)
        applyAction(app, a)
    //tính các giá trị tại thời điểm t+1
        ΔQ_t = r_{t+1} + γ max_{a'} Q(x_{t+1},a_t) − Q(x_t,a_t)
        q[i,j] = q[i,j] + ηΔQ_t α_i(x_t) / Σ_i α_i(x_t)
```

4.3.3. Giải pháp tự động điều chỉnh

Hoạt động của bộ AS (AFQL) được chỉ ra trong bảng 4.1.

Trong trường hợp bộ AS đề xuất, không gian trạng thái là hữu hạn (tức là có 27 trạng thái là kết hợp của 3 x 3 x 3 hàm thành viên cho các biến tài nguyên khả dụng trung bình ($\overline{w}$), phương sai của tài nguyên khả dụng (var) và thời gian đáp ứng trung bình (rt)) và bộ AS phải lựa chọn một trong năm hành động có thể {-2, -1, 0, 1, 2}. Tuy nhiên phương pháp thiết kế này là tổng quát có thể áp dụng cho những không gian trạng thái và hành động khác.

4.4. THỰC NGHIỆM VÀ ĐÁNH GIÁ

Luận án đã xây dựng tập luật chuyên gia, các giá trị hành động nằm trong khoảng $[-2, 2]$, tương ứng với việc loại bỏ 2 VM, loại bỏ 1 VM, không làm gì cả, thêm 1 VM và thêm 2 VM. Với việc loại bỏ các VM, sẽ chọn VM nào có mức khả dụng trung bình nhiều nhất. Việc thêm VM, sẽ thực hiện thêm VM có cấu hình thấp nhất trong số VM đang hoạt động để đảm bảo chi phí thuê là nhỏ nhất.

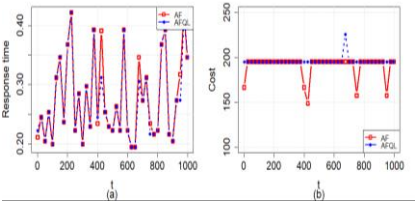
Kết quả thực nghiệm

Qua thực nghiệm, luận án đã chọn được bộ tham số $\varepsilon = 0.2$, $\gamma = 0.8$ và $\eta = 0.2$.

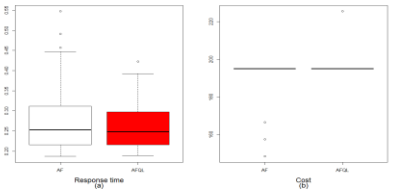
4.4.1. Đánh giá bộ tự động điều chỉnh đề xuất với logic mờ sử dụng tập luật chuyên gia

Ở đây, sử dụng bộ tham số ε=0.2, γ=0.8 và η=0.2 cho bộ AS để so sánh với bộ AS sử dụng tập luật chuyên gia.

Từ hình 4.4 và 4.5 cho thấy, bộ AS sử dụng kỹ thuật học tăng cường kết hợp với logic mờ (AFQL) cơ bản thích nghi tốt hơn bộ AS sử dụng logic với tập luật chuyên gia.



Hình 4.5: So sánh dựa vào thời gian đáp ứng trung bình và chi phí

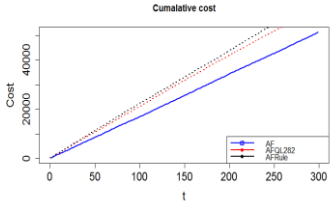


Hình 4.6: Biểu đồ hộp so sánh giữa thời gian đáp ứng trung bình và chi phí

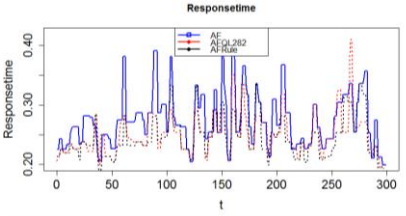
4.4.2. Đánh giá bộ tự động điều chỉnh đề xuất với các bộ dữ liệu khác

Luận án đã tạo ra ba bộ dữ liệu theo hướng của bộ dữ liệu ClarNet, bộ dữ liệu Wiki và bộ dữ liệu sinh ngẫu nhiên.

Kết quả thực nghiệm được đánh giá dựa vào ba bộ dữ liệu trên thông qua các tham số đầu vào gồm tài nguyên khả dụng trung bình (Available load), phương sai của tài nguyên khả dụng (Variance Available load), thời gian đáp ứng trung bình (Response time) và sau đó so sánh kết quả thu được theo tổng chi phí và thời gian đáp ứng trung bình theo các bộ tự động điều chỉnh khác nhau.

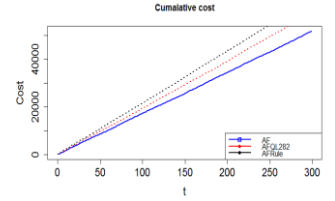


Hình 4.7: So sánh tổng chi phí đối với 3 trường hợp trên dữ liệu ClarkNet

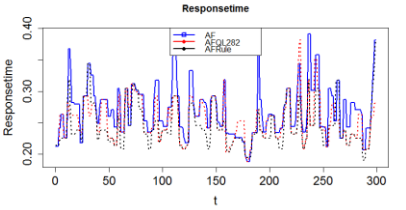


Hình 4.8: So sánh thời gian đáp ứng trung bình đối với 3 trường hợp trên bộ dữ liệu ClarkNet

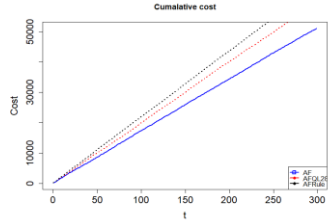
Với kết quả thực nghiệm trên ba bộ dữ liệu trên cho thấy, tập luật được tạo ra sau khi học tăng cường kết hợp với điều khiển mờ khi sử dụng bộ tham số (ε=0.2, γ=0.8 và η=0.2) đều cho lợi thế về thời gian đáp ứng ít nhất. Tuy nhiên phải chịu tổng chi phí cao hơn.



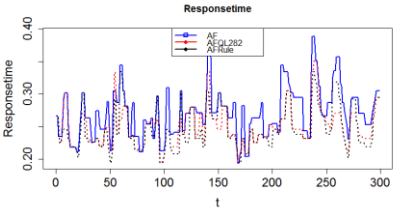
Hình 4.9: So sánh tổng chi phí đối với 3 trường hợp trên dữ liệu Wiki



Hình 4.10: So sánh thời gian đáp ứng trung bình đối với 3 trường hợp trên bộ dữ liệu Wiki



Hình 4.11: So sánh tổng chi phí đối với 3 trường hợp trên dữ liệu ngẫu nhiên



Hình 4.12: So sánh thời gian đáp ứng trung bình đối với 3 trường hợp trên bộ dữ liệu ngẫu nhiên

KẾT LUẬN CHƯƠNG 4

Bộ AS được xây dựng để điều chỉnh tài nguyên tự động bằng cách sử dụng học tăng cường kết hợp với logic mờ để tăng khả năng đáp ứng tức thời yêu cầu tài nguyên trong việc cấp phát/thu hồi tài nguyên tự động dựa vào tài nguyên khả dụng trung bình, phương sai của tài nguyên khả dụng của của cụm VM và thời gian đáp ứng trung bình của hệ thống. Bên cạnh đó, bộ điều khiển AS còn cho thấy hiệu quả về mặt chi phí. Bộ AS đề xuất không cần có tri thức ban đầu mà vẫn hoạt động hiệu quả, các tri thức có được thông qua việc được tích lũy dần dần trong quá trình học. Phần thực nghiệm, đã lựa chọn bộ tham số (ε, γ, η) thích hợp để tiến hành so sánh hai mô hình AS sử dụng học tăng cường kết hợp với logic mờ (AFQL) và mô hình

AS sử dụng logic mờ (AF). Kết quả cho thấy bộ AS AFQL hiệu quả và không cần có tri thức ban đầu, bộ AS AFQL vẫn có thể hoạt động hiệu quả, các tri thức có được trong quá trình hoạt động.

Tuy nhiên, hiệu quả của bộ AS đề xuất còn phụ thuộc vào nhiều yếu tố khác, như số đoạn phân hoạch các giá trị đầu vào, hệ thống suy luận mờ, các tham số ε, γ, η… Đây là hướng phát triển tiếp theo của luận án để có thể xây dựng được bộ AS hiệu quả hơn và có thể áp dụng hiệu quả vào trong thực tế.

KẾT LUẬN

1) Những kết quả chính của luận án:

(1) Luận án đã nghiên cứu và đề xuất được kỹ thuật cân bằng tải và kỹ thuật di trú hiệu quả trong CC tạo điều kiện thuận lợi cho việc điều chỉnh tài nguyên giúp cải thiện hiệu năng của các trung tâm điện toán đám mây. Tiếp theo, Luận án đã đánh giá sự ảnh hưởng của các kỹ thuật cân bằng tải trong môi trường CC có AS tài nguyên, giúp cho việc lựa chọn kỹ thuật cân bằng tải hiệu quả.

(2) Luận án đã mô hình hóa môi trường CC phức tạp và không đồng nhất sử dụng mô hình mạng hàng đợi – mạng Jackson mở, làm cơ sở để đánh giá các số đo hiệu năng của trung tâm CC.

(3) Luận án đã xây dựng được một bộ AS hiệu quả trong môi trường CC. Bộ AS sử dụng kết hợp giữa học tăng cường và logic mờ.

2) Hướng phát triển của luận án:

(1) Luận án có thể được phát triển theo hướng xây dựng mô hình cơ sở dựa vào các mô hình hàng đợi khác nhau để đo lường và đánh giá hiệu năng của hệ thống điện toán đám mây. Từ đó có được mô hình lý thuyết đầy đủ hỗ trợ hoạt động nghiên cứu và triển khai hệ thống điện toán đám mây.

(2) Ngoài ra, luận án có thể được phát triển theo hướng cấu hình tùy biến bộ tự động điều chỉnh theo trạng thái của hệ thống, trên cơ sở nghiên cứu phân hoạch giá trị đầu vào và sử dụng các hệ suy luận mờ khác nhau.

DANH MỤC CÁC CÔNG TRÌNH CÔNG BỐ

TẠP CHÍ QUỐC TẾ

TCNN1. Nguyen Hong Son, **Nguyen Khac Chien** (2017), "Load Balancing in Auto Scaling-Enabled Cloud Environments", *International Journal on Cloud Computing: Services and Architecture (IJCCSA)*. Vol. 7, No. 5, October 2017, pp. 15-22.

TCNN2. **Chien Nguyen Khac**, Khiet Bui Thanh, Hung Ho Dac, Vu Pham Tran, Hung Tran Cong, Son Nguyen Hong (2018). *"An auto-scaling approach for infrastructure as a service cloud computing based on Fuzzy Q-Learning".* ***Đang phản biện.*** Concurrency and Computation: Practice and Experience Journal. **Special Issue On**: Context-Aware Systems and Applications.

TCNN3. **Chien Nguyen Khac**, Khiet Bui Thanh, Hung Ho Dac, Son Nguyen Hong, Vu Pham Tran, Hung Tran Cong (2018). "*An open jackson network model for heterogeneous infrastructure as a service on cloud computing".* International Journal of Computer Networks & Communications (IJCNC) (2018). http://airccse.org/journal/ijcnc.html ***(Đã chấp nhận)***

TẠP CHÍ TRONG NƯỚC

TCTN1. **Nguyễn Khắc Chiến,** Nguyễn Hồng Sơn, Hồ Đắc Lộc, Nguyễn Văn Vịnh (2016), "Nghiên cứu một số thuật toán ra quyết định di trú máy ảo trong điện toán đám mây", *Tạp chí Khoa học Giáo dục Kỹ thuật trường Đại học Sư phạm Kỹ thuật Thành phố Hồ Chí Minh*. 35B (3/2016), pp. 74-81.

HỘI NGHỊ QUỐC TẾ

HNQT1. **Nguyen Khac Chien,** Nguyen Hong Son, Ho Dac Loc (2016), "Load balancing algorithm based on estimating finish time of services in cloud computing", *International Conference on Advanced Communication Technology (ICACT) at the Phoenix Park, PyeongChang, Korea*, pp. 228-233, Jan. 31-Feb. 3, 2016.

HNQT2. **Nguyen Khac Chien,** Vo Sy Giang Dong, Nguyen Hong Son, and Ho Dac Loc. (2016), "An Efficient Virtual Machine Migration Algorithm based on Minimization of Migration in Cloud Computing", *International Conference on Nature of Computation and Communication (ICTCC) in Rach Gia city of Vietnam*, Springer, pp. 62-71.

HNQT3. Khiet Bui Thanh, Lam Mai Nguyen Xuan, **Chien Nguyen Khac,** Hung Ho Dac, Vu Pham Tran, Hung Tran Cong (2018), "An auto-scaling VM game approach for multi-tier application with Particle swarm optimization algorithm in Cloud computing". *International Conference on Advanced Technologies for Communication (ATC) October 18-20, 2018, Ho Chi Minh City, Vietnam.*

HỘI NGHỊ TRONG NƯỚC

HNTN1. **Nguyễn Khắc Chiến,** Bùi Thanh Khiết, Hồ Đắc Hưng, Nguyễn Hồng Sơn, Hồ Đắc Lộc (2017), "Một mô hình học tăng cường cho vấn đề điều chỉnh tự động tài nguyên trong CC dựa trên fuzzy q-learning", *Kỷ yếu Hội nghị Quốc gia lần thứ X về Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR), Đà Nẵng, ngày 17-18/08/2017*, pp. 535-543.