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

HƯỚNG TIẾP CẬN SWOT CHO CÂN BẰNG TẢI TRÊN ĐIỆN TOÁN ĐÁM MÂY

Chuyên ngành: **Hệ thống thông tin**

Mã số: **9.48.01.04**

TÓM TẮT LUẬN ÁN TIẾN SĨ KỸ THUẬT

HÀ NỘI - 2023

Công trình được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Phản biện 1:…….

Phản biện 2: ……

Phản biện 3: ……

Luận án được bảo vệ trước Hội đồng chấm luận án cấp Học viện tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG,
KM 10 đường Nguyễn Trãi, Hà Đông, Hà Nội.

Họp tại: ……

Vào hồi……giờ……ngày……tháng……năm 2023

Có thể tìm hiểu luận án tại:

Thư viện Học viện Công nghệ Bưu chính viễn thông

TÓM TẮT

Cân bằng tải trên đám mây cần nghiên cứu và cải tiến, với nhiều thuật toán như Max-Min, Min-Min, Round-Robin, CLBDM, Active Clustering, nhưng việc sử dụng phương pháp dự đoán học máy vẫn còn đầy thách thức. Với ý tưởng từ cách tiếp cận SWOT (điểm mạnh, điểm yếu, cơ hội và nguy cơ), luận án này phân tích cân bằng tải trong môi trường đám mây thông qua hai hướng tiếp cận: bên trong và bên ngoài. Hướng tiếp cận bên trong tập trung vào phân tích thuật toán cân bằng tải và các yếu tố bên trong như thời gian phản hồi và thông lượng. Hướng tiếp cận bên ngoài xem xét các yếu tố bên ngoài như hành vi người dùng, cấu trúc mạng và môi trường địa lý. Luận án nghiên cứu các phương pháp học máy và khai phá dữ liệu để cải thiện hiệu năng cân bằng tải trong đám mây. Luận án đề xuất 4 thuật toán cân bằng tải (MCCVA, APRTA, RCBA và ITA) từ hướng tiếp cận bên trong và 2 thuật toán cân bằng tải (PDOA và k-CTPA) từ hướng tiếp cận bên ngoài. Các thuật toán được triển khai trên môi trường mô phỏng CloudSim và so sánh với các thuật toán phổ biến khác. Luận án sử dụng các thông số khác nhau để đánh giá thực nghiệm như thời gian đáp ứng và speedup, và kết quả mô phỏng cho thấy hiệu quả của các thuật toán dự đoán học máy trong cải thiện cân bằng tải trên đám mây.

ABSTRACT

Cloud load balancing needs research and improvement, with many algorithms like Max-Min, Min-Min, Round-Robin, CLBDM, Active Clustering, but using predictive machine learning approach is still challenging. With ideas from the SWOT approach (strengths, weaknesses, opportunities and threats), this thesis analyzes load balancing in the cloud environment through two approaches: internal and external. The inside approach focuses on analyzing the load balancing algorithm and internal factors such as response time and throughput. The external approach considers external factors such as user behavior, network structure, and geographic environment. The thesis researches machine learning and data mining methods to improve load balancing performance in the cloud. The thesis proposes 4 load balancing algorithms (MCCVA, APRTA, RCBA and ITA) from the internal approach and 2 load balancing algorithms (PDOA and k-CTPA) from the external approach. Algorithms are deployed on CloudSim simulation environment and compared with other popular algorithms. The thesis uses different parameters for empirical evaluation such as response time and speedup, and simulation results show the effectiveness of machine learning prediction algorithms in improving cloud load balancing.

MỞ ĐẦU

Tính cấp thiết của đề tài

Với sự nhanh chóng phát triển về quy mô cũng như số lượng của các ứng dụng chạy trên nền tảng cloud, cân bằng tải phải luôn luôn được cải tiến và nâng cấp cho phù hợp với lượng và chất của sự phát triển đó. Vì thế mà cân bằng tải là một thách thức lớn, luôn được sự quan tâm của các nhà khoa học, nghiên cứu nhằm đáp ứng ngày một tốt hơn cho cloud. Đã có nhiều công trình nghiên cứu trong và ngoài nước về việc nâng cao hiệu năng cân bằng tải trên điện toán đám mây. Tuy nhiên việc nâng cao hiệu năng cân bằng tải trong điện toán đám mây vẫn luôn là thách thức, là bài toán mà cần có lời giải tốt hơn, hiệu quả hơn, đặc biệt với sự đa dạng và phát triển ngày càng lớn mạnh của cloud (đa dạng về dịch vụ, phần mềm, nền tảng chạy trên các máy chủ / máy chủ ảo trên đám mây) cũng như nhu cầu sử dụng mỗi lúc một tăng của người dùng (về cả chất lượng và số lượng). Trong các công trình nghiên cứu về cân bằng tải trên đám mây, chúng ta dễ dàng nhận thấy việc sử dụng các phương pháp dự đoán kết hợp học máy và dữ liệu chưa được mô tả rõ nét.

Chính vì những lý do trên, luận án này phân tích cân bằng tải trong môi trường đám mây với ý tưởng từ cách tiếp cận SWOT (điểm mạnh, điểm yếu, cơ hội và nguy cơ), từ đó đưa ra đánh giá cân bằng tải với hai hướng tiếp cận: bên trong và bên ngoài. Hướng tiếp cận bên trong tập trung vào phân tích thuật toán cân bằng tải và các yếu tố bên trong như thời gian phản hồi và thông lượng. Hướng tiếp cận bên ngoài xem xét các yếu tố bên ngoài như hành vi người dùng, cấu trúc mạng và môi trường địa lý. Luận án tập trung nghiên cứu phương pháp học máy và khai phá dữ liệu để cải thiện hiệu năng cân bằng tải trên đám mây.

Mục tiêu nghiên cứu

Mục tiêu của luận án là dựa trên ý tưởng hướng tiếp cận SWOT từ đó đề xuất và xây dựng các phương pháp nâng cao hiệu năng cân bằng tải trong điện toán đám mây bằng cách ứng dụng / phát triển các thuật toán học máy với việc xử lý và phân tích dữ liệu cân bằng tải.

Phân tích các vấn đề liên quan hiệu năng cân bằng tải trên cloud bằng công cụ SWOT và từ đó đưa ra 2 hướng tiếp cận. *Với hướng tiếp cận từ bên trong,* đề xuất xây dựng phương pháp / thuật toán ứng dụng một số thuật toán học máy (Machine Learning) vào bộ cân bằng tải, nhằm nâng cao hiệu năng cân bằng tải trên điện toán đám mây. Xây dựng cải tiến một số thuật toán cân bằng tải phổ biến hiện nay trên cloud. Cụ thể, đề xuất xây dựng bộ cân bằng tải dựa trên phương pháp dự báo các thông số theo thời gian như thời gian đáp ứng (Response Time), thời gian xử lý (Make span) để nâng cao hiệu năng cân bằng tải trên điện toán đám mây. *Với hướng tiếp cận từ bên ngoài,* đề xuất xây dựng phương pháp dự báo deadlock hoặc khả năng xảy ra

deadlock trên bộ cân bằng tải, đây cũng là yếu tố ảnh hưởng đến khả năng cân bằng tải, từ đó xây dựng bộ cân bằng tải luôn tránh được deadlock và nâng cao hiệu năng cân bằng tải. Đề xuất xây dựng thuật toán cân bằng tải theo góc độ hành vi người dùng bao gồm độ ưu tiên xử lý các tác vụ (task) / các yêu cầu (request) tương ứng của người dùng, từ đó phân bổ hệu quả nhất, nâng cao hiệu năng cân bằng tải trên điện toán đám mây.

Ý nghĩa khoa học và đóng góp

Với ý tưởng phân tích cân bằng tải bằng công cụ SWOT, luận án đưa ra 2 hướng tiếp cận là tiếp cận từ bên trong và tiếp cận từ bên ngoài, để đánh giá và đưa ra giải pháp trong việc nâng cao hiệu năng cân bằng tải trên điện toán đám mây: *Đề xuất xây dựng các kỹ thuật / thuật toán mới trong cân bằng tải trên điện toán đám mây nhằm nâng cao hiệu năng cân bằng tải.* Ngoài ra thông qua một số nghiên cứu, vận dụng hiệu quả & phát triển các thuật toán học máy vào cân bằng tải trên điện toán đám mây. *Đề xuất xây dựng các bộ cân bằng tải mới* trên việc xử lý các thông số thời gian, chuỗi thời gian như dự báo thời gian đáp ứng (response time), thời gian xử lý (makespan) nhằm nâng cao hiệu năng cân bằng tải trên điện toán đám mây. *Đề xuất nghiên cứu cân bằng tải thông qua việc dự báo Deadlock và khả năng xảy ra deadlock* trên môi trường cloud. Với việc dự báo này, giúp cho bộ cân bằng tải kiểm soát tài nguyên trên môi trường cloud tốt hơn. *Đề xuất nghiên cứu cân bằng tải dưới góc độ người dùng trên cloud,* khai thác các tính chất người dùng thông qua hành vi người dùng, độ ưu tiên của người dùng.

CHƯƠNG 1 – GIỚI THIỆU VỀ ĐỀ TÀI

1.1 TỔNG QUAN VỀ ĐIỆN TOÁN ĐÁM MÂY

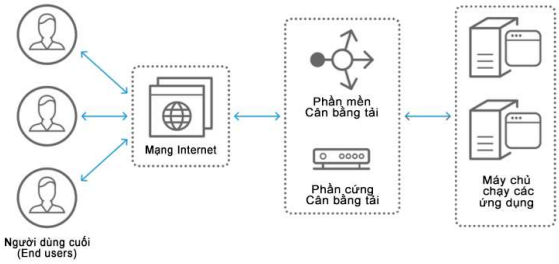
Điện toán đám mây là một hình thức tính toán dựa trên mạng Internet, trong đó tài nguyên và dịch vụ được chia sẻ và cung cấp dưới dạng dịch vụ trên một hạ tầng mạng công cộng. Nó đã phát triển mạnh mẽ với sự gia nhập của các nhà cung cấp lớn như Google, Amazon và Microsoft. Điện toán đám mây mang lại nhiều lợi ích về chi phí và lưu trữ dữ liệu, đồng thời đòi hỏi việc cân bằng tải và sử dụng các thuật toán phù hợp để quản lý tài nguyên và hiệu suất.

1.2 TỔNG QUAN VỀ CÂN BẰNG TẢI TRONG ĐIỆN TOÁN ĐÁM MÂY

1.2.1 Giới thiệu về cân bằng tải

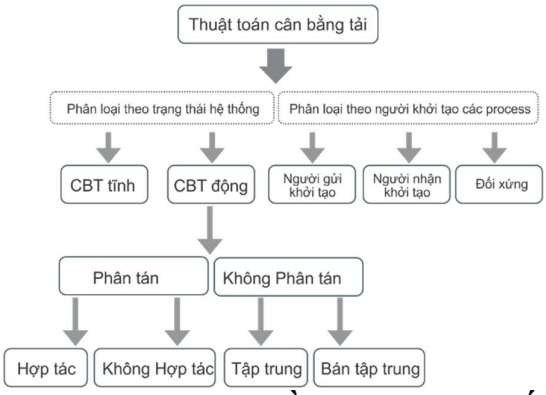
Cloud đã giúp các doanh nghiệp tận dụng lợi ích của tài nguyên điện toán được chia sẻ trên môi trường ảo hóa. Rất nhiều doanh nghiệp đã sử dụng các dịch vụ dựa trên đám mây ở dạng này hoặc dạng khác. Điều này đưa chúng ta đến khái niệm cân bằng tải trong điện toán đám mây. Trong khi lượng truy cập này quá lớn trong thời gian ngắn 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 mà không bị tắc nghẽn cục bộ. *Cân bằng tải* là giải pháp việc phân bố đồng đều và hiệu quả 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, đẩy thời gian uptime của hệ thống lên cao nhất và cải thiện năng suất hoạt động tổng thể.

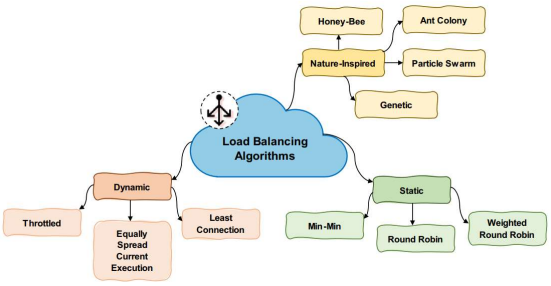


Hình 1.1 Mô hình Cân bằng tải trong điện toán đám mây theo NGINX

Mục đích cân bằng tải, Trên điện toán đám mây việc cân bằng tải được sử dụng để phân phối các tải hoạt động lớn sang các tải hoạt động ít hơn để nâng cao hiệu suất làm việc và tận dụng tối đa tài nguyên của cloud. Trong môi trường đám mây, cân bằng tải đòi hỏi phân bổ lại các tải đang hoạt động liên tục giữa tất cả các nốt.



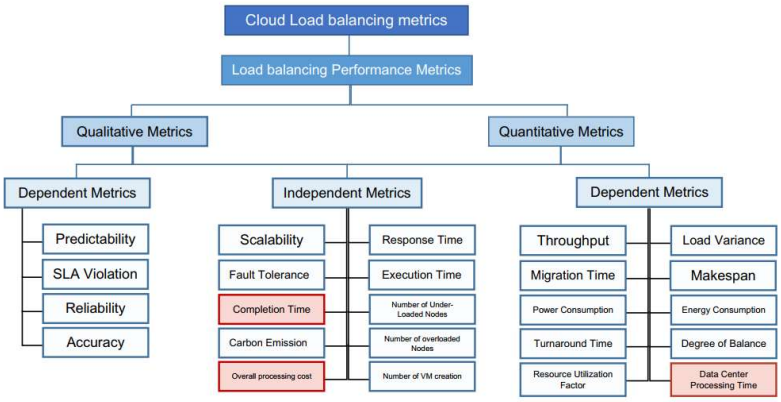
Hình 1. 2 ***Phân loại thuật toán cân bằng tải theo hệ thống và tài nguyên***



Hình 1. 3 ***Phân loại thuật toán cân bằng tải theo tính chất thuật toán***

1.2.2 Đo lường cân bằng tải

Đo lường cân bằng tải, có rất nhiều tham số để đo lường hiệu năng cân bằng tải trên môi trường điện toán đám mây. Một vài tham số cơ bản bao gồm: Thông lượng (Throughput), Dung sai lỗi (Fault Tolerance) , Thời gian di dời (Migration Time) , Thời gian đáp ứng (Response Time), Thời gian thực hiện (Makespan -MS), Khả năng mở rộng (Scalability)



Hình 1. 4 ***Các tham số đo lường cân bằng tải***

1.2.3 Một số thuật toán cân bằng tải phổ biến

Có rất nhiều thuật toán cân bằng tải giúp giải quyết thông lượng tốt và giảm thời gian đáp ứng trên môi trường cloud. Mỗi thuật toán đều có những lợi ích riêng: *Thuật toán phân bố tác vụ dựa trên LB* (Task Scheduling based on LB), *Thuật toán cân bằng tải cơ hội* (Opportunistic Load Balancing – OLB), *Thuật toán Round Robin* (RR), *Thuật toán ngẫu nhiên hóa* (Randomized), *Thuật toán Min-Min, Thuật toán Max-Min, Thuật toán hành vi Tìm kiếm của Ong Mật* (Honeybee Foraging Behavior), *Thuật toán gom cụm động* (Active Clustering), *Thuật toán so sánh và cân bằng* (Compare & Balance), *Thuật toán Lock-free* (Lock-free multiprocessing solution for LB), *Thuật toán đàn kiến* (Ant Colony Optimization), *Thuật toán thời gian đáp ứng ngắn nhất đầu tiên* (Shortest Response Time First), *Thuật toán lấy mẫu ngẫu nhiên* (Based Random Sampling).

1.3 MỘT SỐ THUẬT TOÁN AI ỨNG DỤNG VÀO CÂN BẰNG TẢI

Với việc nghiên cứu ứng dụng và khả năng phát triển của ML và phân tích thông kê dữ liệu, tác giả luận án nhận thấy một số thuật toán khá phù hợp và tương thích với cân bằng tải trên môi trường đám mây bao gồm: thuật toán xác suất Naïve Bayes, thuật toán SVM, thuật toán KMeans, thuật toán dự báo ARIMA, thuật toán dự báo Regression, thuật toán phân lớp k-NN.

Thuật toán Naïve Bayes, là một trong các kĩ thuật phân lớp dựa trên định lý về Bayes với các yếu tố dự đoán được xem như độc lập với nhau. Nói một cách đơn giản, trong

thuật toán Naive Bayes, sự hiện diện của một đặc trưng cụ thể trong một lớp không liên quan đến sự hiện diện của các đặc trưng khác. Các loại đặc trưng này phụ thuộc lẫn nhau hoặc dựa trên sự hiện diện của các đặc trưng khác, tất cả thuộc tính này là độc lập.

Thuật toán SVM, phương pháp SVM được coi là công cụ mạnh cho những bài toán phân lớp phi tuyến tính được các tác giả Vapnik và Chervonenkis phát triển mạnh mẽ năm 1995. Phương pháp này thực hiện phân lớp dựa trên nguyên lý Cực tiểu hóa rủi ro có cấu trúc SRM (Structural Risk Minimization), được xem là một trong các phương pháp phân lớp giám sát không tham số tinh vi nhất cho đến nay. Các hàm công cụ đa dạng của SVM cho phép tạo không gian chuyển đổi để xây dựng mặt phẳng phân lớp.

Thuật toán K-Means, Phân cụm là kỹ thuật rất quan trọng trong khai phá dữ liệu, nó thuộc lớp các phương pháp Unsupervised Learning trong Machine Learning. K-Means là thuật toán rất quan trọng và được sử dụng phổ biến trong kỹ thuật phân cụm. Tư tưởng chính của thuật toán K-Means là tìm cách phân nhóm các đối tượng (objects) đã cho vào K cụm (K là số các cụm được xác định trước, K nguyên dương) sao cho tổng bình phương khoảng cách giữa các đối tượng đến tâm nhóm (centroid) là nhỏ nhất.

Thuật toán dự báo ARIMA, ARIMA là thuật toán dựa trên thống kê, là thuật toán tự hồi quy tích hợp trung bình trượt (Auto Regression Integrated Moving Average), được phát triển từ mô hình hồi quy ARMA (Auto Regression Moving Avera). Đây là mô hình phát triển trên số liệu chuỗi thời gian đã biết và dự báo số liệu trong tương lai gần.

Thuật toán Regression, Linear Regression là một phương pháp thống kê để hồi quy dữ liệu với biến phụ thuộc có giá trị liên tục trong khi các biến độc lập có thể có một trong hai giá trị liên tục hoặc là giá trị phân loại. Nói cách khác "Hồi quy tuyến tính" là một phương pháp để dự đoán biến phụ thuộc (Y) dựa trên giá trị của biến độc lập (X). Nó có thể được sử dụng cho các trường hợp chúng ta muốn dự đoán một số lượng liên tục.

Thuật toán K-NN, K-nearest neighbor là một trong những thuật toán supervised-learning đơn giản nhất (mà hiệu quả trong một vài trường hợp) trong Machine Learning. Khi training, thuật toán này không học một điều gì từ dữ liệu training (đây cũng là lý do thuật toán này được xếp vào loại lazy learning), mọi tính toán được thực hiện khi nó cần dự đoán kết quả của dữ liệu mới. K-nearest neighbor có thể áp dụng được vào cả hai loại của bài toán Supervised learning là Classification và Regression.

CHƯƠNG 2 – TIẾP CẬN SWOT CHO CÂN BẰNG TẢI TRÊN ĐIỆN TOÁN ĐÁM MÂY

Cân bằng tải được coi là một yếu tố quan trọng trong điện toán đám mây, ảnh hưởng đến chất lượng dịch vụ. Luận án này sử dụng phân tích SWOT để nghiên cứu các vấn đề liên quan đến cân bằng tải và đề xuất các hướng tiếp cận để cải thiện hiệu năng cân bằng tải trong môi trường đám mây.

2.1 GIỚI THIỆU VỀ CÔNG CỤ SWOT

SWOT là viết tắt của 4 từ tiếng Anh, Strengths (thế mạnh), Weaknesses (điểm yếu), Opportunities (cơ hội) và Threats (thách thức), là công cụ phân tích phổ biến và dễ dàng sử dụng, thường được dùng trong các doanh nghiệp. Phân tích SWOT tức là phân tích 4 yếu tố, thông qua việc phân tích 4 yếu tố này, giúp chúng ta xác định rõ hơn các vấn đề tồn tại, từ đó đưa ra mục tiêu chiến lược, hướng đi cho nhằm cải thiện dịch vụ cho doanh nghiệp. Bằng cách thực hiện phân tích SWOT, luận án này sẽ có thể xây dựng cầu nối giữa những gì cân bằng tải đã đạt được cho đến thời điểm hiện nay và các thuật toán và hướng nghiên cứu mới sẽ được nhìn thấy rõ hơn.

2.2 PHÂN TÍCH SWOT HIỆU NĂNG CÂN BẰNG TẢI TRÊN CLOUD

2.2.1 Hiệu năng cân bằng tải trên cloud

Khái niệm về hiệu năng: Hiệu năng của một sản phẩm cho thấy tính hiệu quả trong quá trình hoạt động của thiết bị đó. Nó bao gồm mức tiêu thụ năng lượng, khả năng tối ưu phần mềm, công suất làm việc, sức mạnh xử lý và thời gian để hoàn tất các tác vụ. Trên một thiết bị công nghệ, hiệu năng được xem là yếu tố tổng hòa của các đặc điểm kể trên. Chúng ta hoàn toàn có thể đánh giá hiệu năng của một chiếc smartphone, thông qua việc nhận diện cấu hình như chip, RAM, bộ nhớ trong.

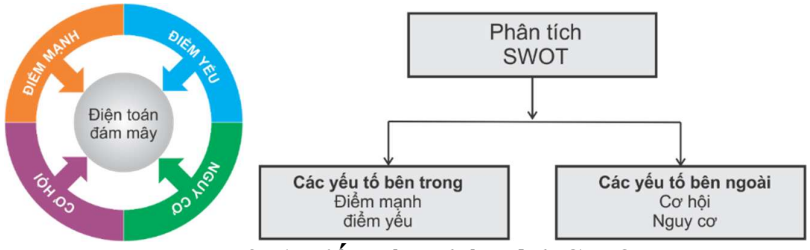
Khái niệm về hiệu năng cân bằng tải: Tương tự như một thiết bị, cân bằng tải cũng có thể xem là một thiết bị hoạt động trên môi trường điện toán đám mây, và có hiệu năng cân bằng tải của nó. Theo các nhà phát triển và cung cấp dịch vụ điện toán đám mây lớn như IBM, NGINX v/v thì để đo lường hiệu năng cân bằng tải có thể sử dụng các yếu tố chủ yếu như sau: *Yêu cầu mỗi giây (Requests per second - RPS)*, *Độ trễ* (Latency), gian phản hồi (Response Time), *Khả năng phân bổ tài nguyên (Resource Allocation Capacity), Mức độ công bằng trong phân bổ tài nguyên (Allocation Fairness), Khả năng tăng tốc (Speedups), Khả năng đồng bộ giữa các tác vụ (Tasks Synchronzition), Khả năng chịu lỗi (Fault Tolerance).*

Thông qua các yếu tố trên để đo lường và kiểm soát hiệu năng cân bằng tải trên cloud, cụ thể hóa việc nâng cao hiệu năng cân bằng tải bằng việc đo lường các thông số trên trong các thuật toán. Cũng theo nghiên cứu này, thì các tham số để đo lường hiệu năng cân bằng tải của các thuật toán bao gồm : Thông lượng (Throughput),

Overhead, Dung sai lỗi (Fault Tolerance), thời gian di dời (Migration Time), thời gian đáp ứng (Response Time), tối ưu hóa tài nguyên (Resouce Utilization), khả năng co giãn (Scalability), hiệu quả hoạt động (efficiency).

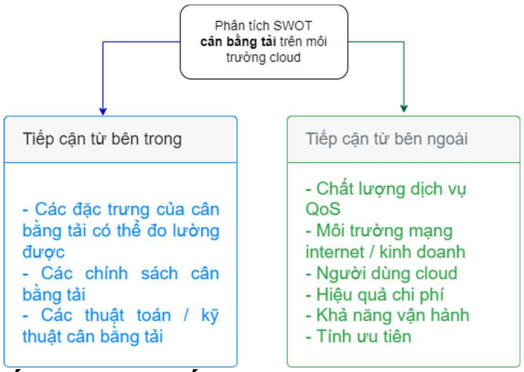
2.2.2 Phân tích SWOT cân bằng tải

Luận án này lấy ý tưởng từ cách tiếp cận của công cụ SWOT dùng để tiếp cận nghiên cứu cân bằng tải trên cloud từ đó đưa ra hướng tiếp cận để nâng cao hiệu năng cân bằng tải trên môi trường điện toán đám mây. Theo công cụ SWOT, ta cần phải xác định mục tiêu cần đạt được, đó là nâng cao hiệu năng cân bằng tải. Từ đó, đưa ra các mục tiêu cụ thể, phân tích theo hướng tiếp cận bên trong (điểm mạnh và điểm yếu) và tiếp cận bên ngoài (cơ hội và thách thức / khó khăn), từ đó đưa ra các giải pháp tương ứng với nó, để đáp ứng được mục tiêu là nâng cao hiệu năng cân bằng tải.



Hình 2. 1 *Tiếp cận phân tích SWOT*

Như vậy, *các yếu tố bên trong cân bằng tải* của môi trường đám mây chính là các thuộc tính, tính chất mà chúng ta có thê đo lường được, cụ thể hơn là các thuộc tính đặc trưng tải của cân bằng tải: thông lượng (Throughput), dung sai lỗi (Fault Tolerance), Thời gian di dời (Migration Time), Thời gian đáp ứng (Response Time), khả năng mở rộng (Scalability). Ngoài ra, yếu tố bên trong của cân bằng tải chính là chính sách cân bằng tải, cách hoạt động của cân bằng tải hay cụ thể hơn là các thuật toán cân bằng tải. *Các yếu tố bên ngoài cân bằng tải* chính là chất lượng dịch vụ, môi trường kinh doanh, môi trường mạng internet, người dùng cloud và các mục tiêu mà nhà cung cấp cloud hướng tới: Hiệu quả chi phí (Cost effectiveness), Tính ưu tiên (Priority), Khả năng mở rộng & tính linh hoạt (Scalability and flexibility).



Hình 2. 2 Đề xuất 2 hướng tiếp cận nâng cao hiệu năng cân bằng tải

Để nâng cao hiệu năng cân bằng tải, theo tiếp cận SWOT, luận án này đề xuất 2 hướng tiếp cận nghiên cứu cân bằng tải trên cloud, đó là hướng tiếp cận từ bên ngoài và hướng tiếp cận từ bên trong. *Cụ thể đối với hướng tiếp cận từ bên trong,* chúng ta nghiên cứu các đặc trưng của cân bằng tải, các thông số có thể đo lường được, từ đó cải tiến, ứng dụng các thuật toán mới vào, ví dụ như thời gian đáp ứng. Bên cạnh các thông số của cân bằng tải, chúng ta cũng có thể bắt đầu từ các chính sách cân bằng tải, hoặc cơ chế cân bằng tải, mà cụ thể hơn là các thuật toán cân bằng tải hiện có, từ đó cải tiến hoặc nâng cấp cho phù hợp, nâng cao hiệu năng làm việc của cân bằng tải. *Đối với hướng tiếp cận từ bên ngoài,* chính là việc nghiên cứu môi trường xung quanh của cân bằng tải trên môi trường đám mây. Cụ thể đó là yêu cầu chất lượng dịch vụ, là một yếu tố tiêu chuẩn chất lượng đưa ra do người dùng và nhà cung cấp dịch vụ đưa ra. Ngoại ra, yếu tố môi trường mạng, mạng internet cũng là các vấn đề nằm ngoài cân bằng tải. Người dùng cloud, và hành vi người dùng cloud, cũng như độ ưu tiên của những người dùng này… Tất cả đều nằm bên ngoài bộ cân bằng tải, nhưng nó quyết định đến khả năng vận hành tốt hay hiệu năng làm việc của cân bằng tải trên cloud. Chính vì thế, chúng ta có thể bắt đầu từ các yếu tố này, nghiên cứu đề xuất cải tiến các thuật toán nâng cao hiệu năng cân bằng tải.

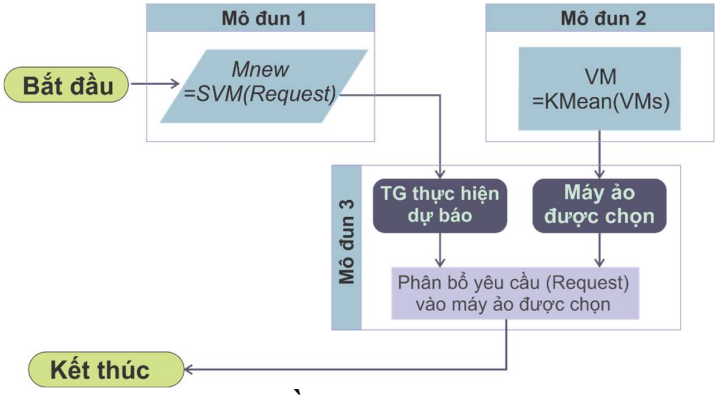
CHƯƠNG 3 – **CÂN BẰNG TẢI THEO HƯỚNG TIẾP CẬN BÊN TRONG**

Trong chương này, luận án dựa vào hướng tiếp cận từ bên trong, đề xuất xây dựng các thuật toán nghiên cứu ứng dụng ML và phân tích dữ liệu vào cân bằng tải trên điện toán đám mây, đặc biệt là nhóm thuật toán phân lớp, phân cụm và dự báo. Với nghiên cứu đạt được, tác giả xin đề xuất 03 thuật toán với cách tiếp cận các thông số của cân bằng tải là thời gian đáp ứng, thời gian xử lý kết hợp với các thuật toán học máy như sau: *Thuật toán MCCVA [CT1]:* sử dụng SVM và k-Means phối hợp trong cân bằng tải; *Thuật toán APRTA [CT2]:* sử dụng thuật toán dự báo ARIMA để dự báo thời gian đáp ứng từ đó nâng cao hiệu năng cân bằng tải. *Thuật toán RCBA [CT7]:* sử dụng Naïve Bayes và k-Means phối hợp trong cân bằng tải. Song song với tiếp cận các thông số cân bằng tải, tiếp cận bên trong cũng chính là việc cải tiến hiệu quả của các thuật toán hiện có. Luận án đã đóng góp một thuật toán cải tiến từ thuật toán Throttle : *Thuật toán ITA [CT3],* là thuật toán cải tiến từ thuật toán cân bằng tải nổi tiếng là Throttle Algorithm.

Về cài đặt mô phỏng các thuật toán, luận án giả lập môi trường cloud bằng cách sử dụng bộ công cụ CloudSim (được cung cấp bởi http://www.cloudbus.org/) và lập trình trên ngôn ngữ JAVA. Môi trường giả lập cloud là từ 5 đến 15 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. Thực nghiệm mô phỏng các thuật toán đề xuất được cài đặt trên ngôn ngữ JAVA và sử dụng APACHE NETBEAN IDE để chạy thử và hiển thị kết quả dưới dạng console. Bên cạnh đó, các thuật toán SVM, K-Means, Naïve Bayes, Linear Regression, ARIMA được cài đặt từ bộ thư viện Weka và Tensorflow Java, kết hợp với CloudSim trên môi trường mô phỏng, cài đặt thuật toán đề xuất. Tuy nhiên, với thuật toán ITA thì cấu hình và cài đặt trên *CloudAnalyst*, là một phiên bản của *CloudSim* nhưng có đầy đủ GUI.

3.1 THUẬT TOÁN MCCVA

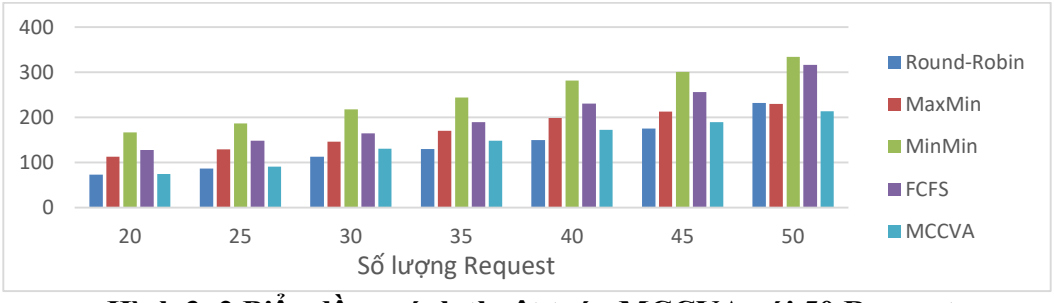


Hình 3. 1 ***Sơ đồ của thuật toán MCCVA***

Theo thuật toán đề xuất, đầu ra của phân lớp request được tính toán chính là thời gian xử lý xét, và không biết được giá trị max hay giá trị min, nên có thể lưu lại 1 số lượng nhất định thời gian xử lý của các request trước nhằm thực hiện tính toán và phân bổ. Chính vì thế, luận văn này xin được sử dụng lại phương pháp loại suy hoặc newton để tính toán ra vị trí phân bổ phù hợp, tuy nhiên sẽ hiệu chỉnh một số thay đổi, hoặc đưa vào các hệ số và tham số, tùy thuộc vào kết quả thực nghiệm.

CÀI ĐẶT THUẬT TOÁN MCCVA

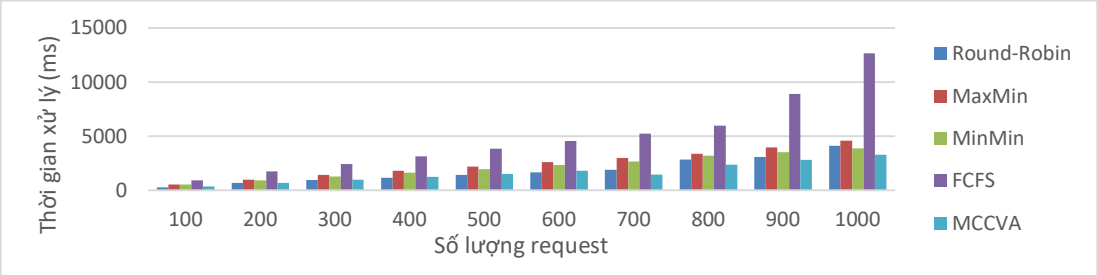
Kết quả chạy thực nghiệm mô phỏng trên CloudSim với 5 máy ảo được dựng sẵn để đáp ứng các yêu cầu, các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên, số lượng Request từ 20-50 và so sánh với các thuật toán Round-Robin, MaxMin, MinMin và FCFS, thời gian thực hiện là:



Hình 3. 2 Biểu đồ so sánh thuật toán MCCVA với 50 Request

Với kết quả thực nghiệm với 50 Request trở lại, ta thấy thuật toán Round-Robin chiếm ưu thế và xử lý nhanh, thuật toán MaxMin cũng khá ổn định. Thuật toán FCFS thì chưa có thế mạnh. Tuy nhiên thuật toán đề xuất MCCVA cũng khá ổn định, và chứng tỏ dần ổn định và tốt hơn khi xử lý nhiều request hơn.

Kết quả chạy thực nghiệm mô phỏng trên CloudSim với 5 máy ảo được dựng sẵn để đáp ứng các yêu cầu, các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên, số lượng Request lần lượt là 100 đến 1000:

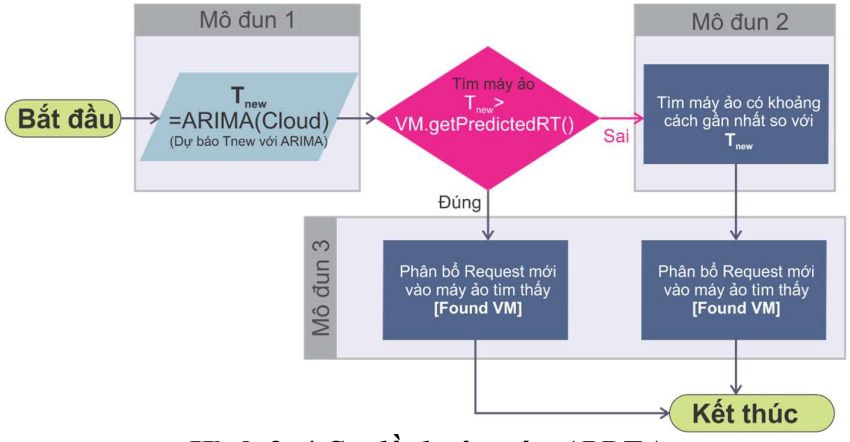


Hình 3. 3 Biểu đồ so sánh thuật toán MCCVA với 1000 Request

Từ request thứ 100 trở đi, thuật toán MCCVA vượt trội hơn hẳn so với MaxMin, MinMin. Tuy nhiên vẫn chưa thấy ưu thế so với RoundRobin. Nhưng với số lượng request càng lớn thì MCCVA càng lợi thế hơn. Và dần dần chiếm ưu thế tuyệt đối so với các thuật toán còn lại. Rõ ràng FCFS thể hiện sự thiếu thông minh và tính tự nhiên của giải thuật.

Thông qua 02 biểu đồ so sánh thời gian xử lý của các thuật toán với điều kiện như nhau ta có thể thấy sự phân bổ khá ổn định và hợp lý của thuật toán đề xuất MCCVA, thời gian xử lý của các máy ảo khả quan so với thời gian xử lý của các thuật toán khác trên cloud (ở trường hợp ít và nhiều request).

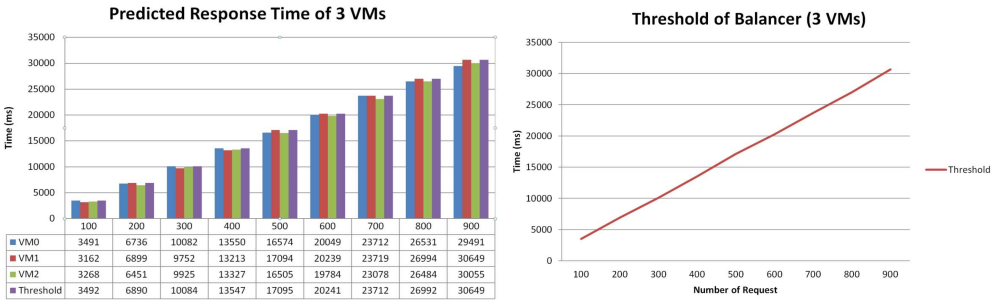
3.3 THUẬT TOÁN APRTA



Hình 3. 4 *Sơ đồ thuật toán APRTA*

Thực nghiệm và kết quả thực nghiệm

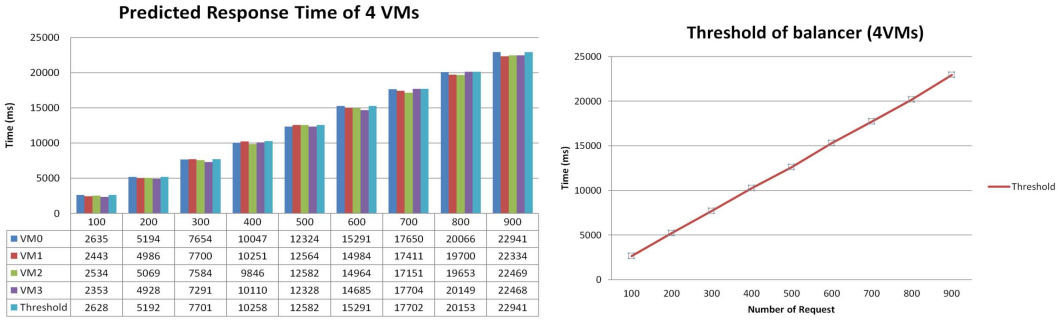
Kết quả chạy thực nghiệm mô phỏng trên CloudSim với *3 máy ảo* được dựng sẵn để đáp ứng các yêu cầu, các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên, số lượng Request lần lượt là 100, 200,… đến 900:



	100	200	300	400	500	600	700	800	900
VM0	3491	6736	10082	13550	16574	20049	23712	26531	29491
VM1	3162	6899	9752	13213	17094	20239	23719	26994	30649
VM2	3268	6451	9925	13327	16505	19784	23078	26484	30055
Threshold	3492	6890	10084	13547	17095	20241	23712	26992	30649

Hình 3. 5 Biểu đồ so sánh thuật toán APRTA của 3 máy ảo

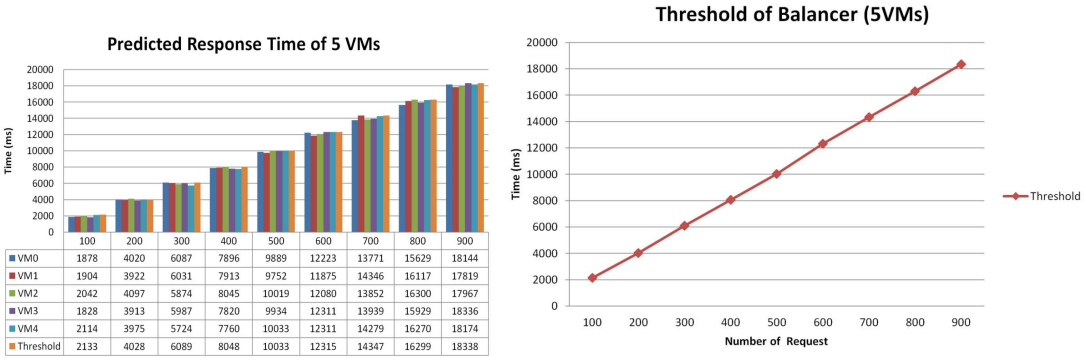
Kết quả chạy thực nghiệm mô phỏng trên CloudSim với 4 máy ảo được dựng sẵn để đáp ứng các yêu cầu, các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên, số lượng Request lần lượt là 100, 200… đến 900:



	100	200	300	400	500	600	700	800	900
VM0	2635	5194	7654	10047	12324	15291	17650	20066	22941
VM1	2443	4986	7700	10251	12564	14984	17411	19700	22334
VM2	2534	5069	7584	9846	12582	14964	17151	19653	22469
VM3	2353	4928	7291	10110	12328	14685	17704	20149	22468
Threshold	2628	5192	7701	10258	12582	15291	17702	20153	22941

Hình 3. 6 Biểu đồ so sánh thuật toán APRTA với 4 máy ảo

Kết quả chạy thực nghiệm mô phỏng trên CloudSim với 5 máy ảo được dựng sẵn để đáp ứng các yêu cầu, các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên, số lượng Request lần lượt là 100, 200… đến 900:



	100	200	300	400	500	600	700	800	900
VM0	1878	4020	6087	7896	9889	12223	13771	15629	18144
VM1	1904	3922	6031	7913	9752	11875	14346	16117	17819
VM2	2042	4097	5874	8045	10019	12080	13852	16300	17967
VM3	1828	3913	5987	7820	9934	12311	13939	15929	18336
VM4	2114	3975	5724	7760	10033	12311	14279	16270	18174
Threshold	2133	4028	6089	8048	10033	12315	14347	16299	18338

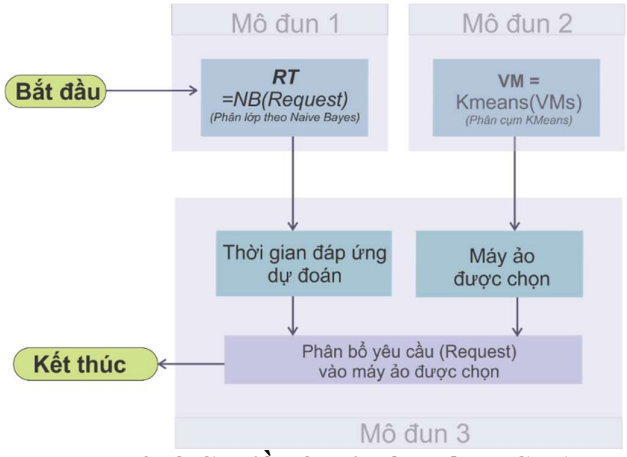
Hình 3. 7 Biểu đồ so sánh thuật toán APRTA của 5 máy ảo

Thông qua 03 biểu đồ so sánh thời gian đáp ứng dự báo của các máy ảo với ngưỡng tính toán (ứng với trường hợp 3 máy ảo, 4 máy ảo và 5 máy ảo) ta có thể thấy sự phân bổ khá ổn định và hợp lý của thuật toán, thời gian đáp ứng dự báo của các máy ảo không quá khác biệt so với thời gian dự báo của cloud (tức là ngưỡng). Ta có

thể thấy sai số dự báo thấp của thuật toán ARIMA, giúp cho việc phân bổ các request tương ứng tới các máy ảo một cách hiệu quả nhất.

3.4 THUẬT TOÁN RCBA

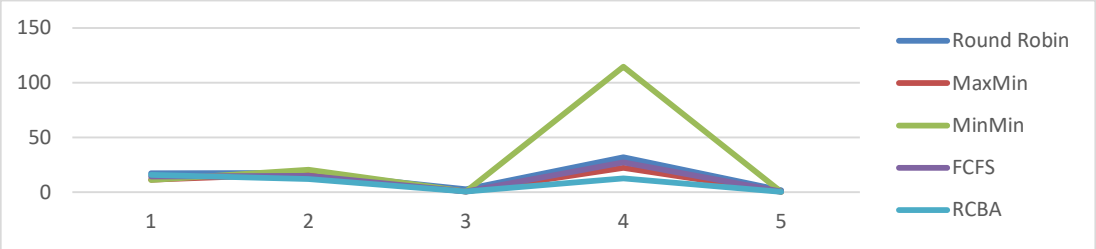
Dựa vào dữ liệu đã có về chuỗi thời gian của thời gian đáp ứng (Response time) của các yêu cầu (request) từ phía khách hàng (client) và một số thuộc tính khác, chúng tôi sử dụng thuật toán Naive Bayes kết hợp với K-means nhằm dự báo thời gian đáp ứng tiếp theo, từ đó biết cách phân bổ tài nguyên cho các yêu cầu tiếp theo.



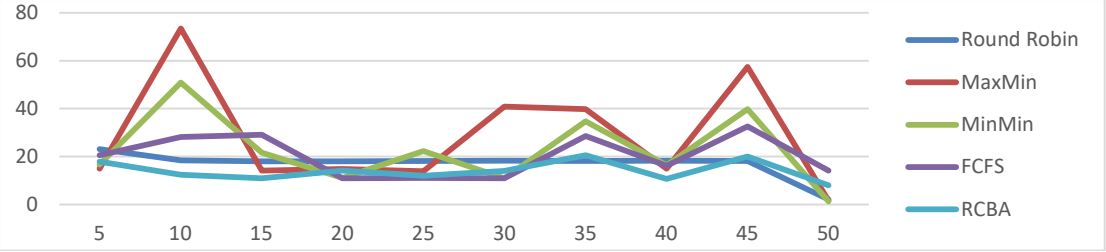
Hình 3. 8 ***Sơ đồ của thuật toán RCBA***

CÀI ĐẶT THUẬT TOÁN RCBA

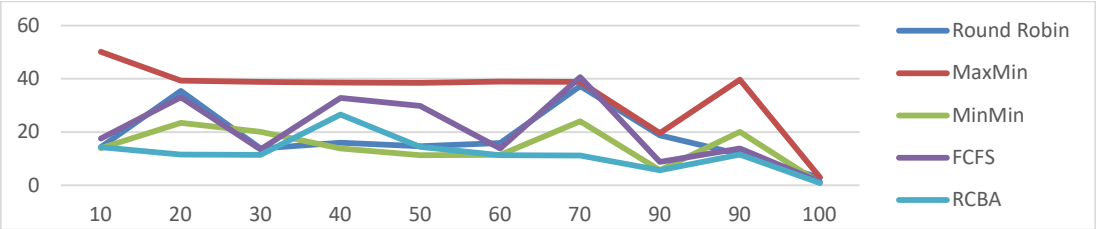
Kết quả chạy thực nghiệm mô phỏng trên CloudSim với 5 máy ảo được dựng sẵn để đáp ứng các yêu cầu, các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên, số lượng Request từ 20-50 và so sánh với các thuật toán Round-Robin, MaxMin, MinMin và FCFS, thời gian thực hiện là:



Hình 3. 9 ***Biểu đồ so sánh thời gian đáp ứng của 5 thuật toán với 50 Request***



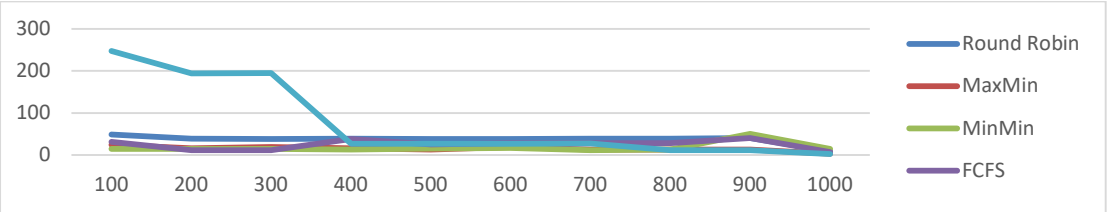
Hình 3. 10 Biểu đồ so sánh thuật toán RCBA với 50 Request



Hình 3. 11 Biểu đồ so sánh thuật toán RCBA với 100 Request

Với kết quả thực nghiệm với 50 Request trở lại, ta thấy thuật toán Round-Robin chiếm ưu thế và xử lý nhanh, thuật toán MaxMin cũng khá ổn định. Thuật toán FCFS thì chưa có thế mạnh. Tuy nhiên thuật toán đề xuất RCBA cũng khá ổn định, và chứng tỏ dần ổn định và tốt hơn khi xử lý nhiều request hơn.

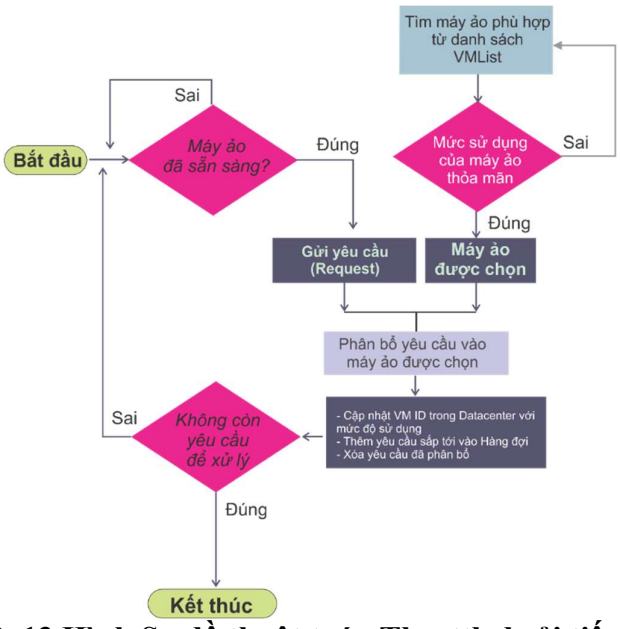
Kết quả chạy thực nghiệm mô phỏng trên CloudSim với 5 máy ảo được dựng sẵn để đáp ứng các yêu cầu, các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên, số lượng Request lần lượt là 100 đến 1000:



Hình 3. 12 Biểu đồ so sánh thuật toán RCBA **với 1000 Request**

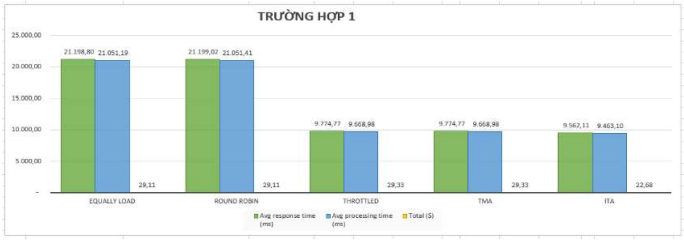
Từ request thứ 100 trở đi, thuật toán RCBA vượt trội hơn hẳn so với MaxMin, MinMin. Tuy nhiên vẫn chưa thấy ưu thế so với RoundRobin. Nhưng với số lượng request càng lớn thì RCBA càng lợi thế hơn. Và dần dần chiếm ưu thế tuyệt đối so với các thuật toán còn lại. Rõ ràng FCFS thể hiện sự thiếu thông minh và tính tự nhiên của giải thuật. Thông qua 02 biểu đồ so sánh thời gian đáp ứng của các thuật toán với điều kiện như nhau ta có thể thấy sự phân bổ khá ổn định và hợp lý của thuật toán đề xuất RCBA, thời gian đáp ứng của các máy ảo khả quan so với thời gian đáp ứng của các thuật toán khác trên cloud (ở trường hợp ít và nhiều request).

3.5 THUẬT TOÁN ITA



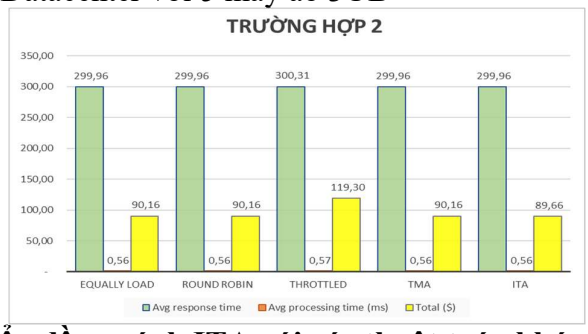
Hình 3. 13 Hình Sơ đồ thuật toán Throttled cải tiến (ITA)

Trường hợp 1: 01 Datacenter với 20 máy ảo



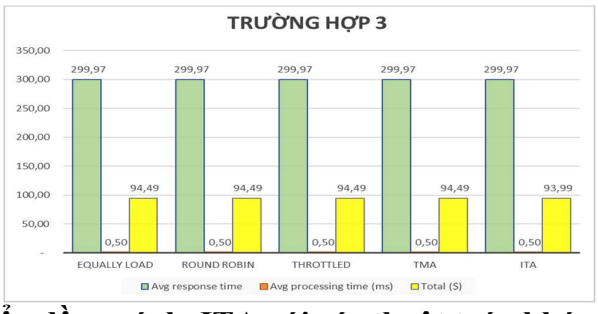
Hình 3. 14 Biểu đồ so sánh ITA với các thuật toán khác trường hợp 1

Trường hợp 2: 01 Datacenter với 5 máy ảo 3UB



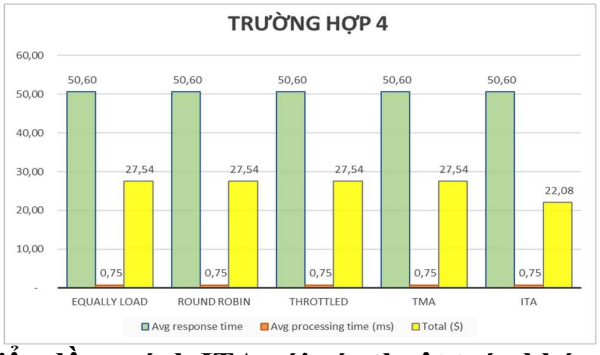
Hình 3. 15 Biểu đồ so sánh ITA với các thuật toán khác trường hợp 2

Trường hợp 3: 01 Datacenter với 5 máy ảo 4UB



Hình 3. 16 Biểu đồ so sánh ITA với các thuật toán khác trường hợp 3

Trường hợp 4: 02 Datacenter Datacenter 1 gồm 50 máy ảo và Datacenter 2 gồm 5 máy ảo



Hình 3. 17 Biểu đồ so sánh ITA với các thuật toán khác trường hợp 4

Nhận xét: Thuật toán ITA có kết quả tốt hơn Throttled ở một số trường hợp data đầu vào, và không hề thua kém các thuật toán có sẵn về các mặt như thời gian đáp ứng về phần chi phí datacenter thì luôn ít hơn các kỹ thuật khác.

CHƯƠNG 4 – CÂN BẰNG TẢI THEO HƯỚNG TIẾP CẬN BÊN NGOÀI

Để nâng cao hiệu năng cân bằng tải trên cloud từ hướng tiếp cận bên ngoài, ta cần xem xét các yếu tố mang tính thách thức và cơ hội mà các nhà cung cấp dịch vụ cloud không thể kiểm soát hay tác động tới. Trong luận án này, hướng tiếp cận từ bên ngoài sẽ chọn ra 02 yếu tố: yếu tố đường truyền mạng hay mạng internet, và yếu tố người dùng cloud hay hành vi người dùng cloud.

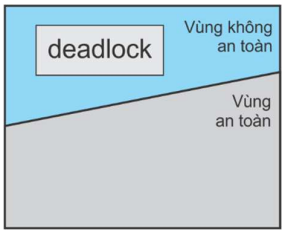
Đối với yếu tố mạng internet, việc bị timeout và hanging là dễ dàng xảy ra nếu cân bằng tải không tốt. Deadlock đại diện cho yếu tố nguy cơ mà trên mạng thường xảy ra. Với mục ý tưởng tiếp cận cân bằng tải từ deadlock, luận án này đề xuất nghiên cứu về deadlock và deadlock trên cloud, từ đó xây dựng thuật toán *PDOA [CT4 & CT6]* nhằm nâng cao khả năng cân bằng tải thông qua dự báo khả năng xảy ra deadlock của cloud. Đối với yếu tố người dùng cloud và hành vi người dùng cloud, luận án này tập trung vào tính ưu tiên của người dùng mà điển hình là độ ưu tiên của tác vụ. Trong môi trường cloud, ta phân biệt người dùng thông qua tính chất của các request, từ đó dựa vào các thông số request mà ta tính toán ra độ ưu tiên của tác vụ. Từ đó, luận án đề xuất một thuật toán cân bằng tải *k-CTPA [CT5]*, dựa vào độ ưu tiên tác vụ để phân bổ các request, giải quyết vấn đề cân bằng tải theo tiếp cận người dùng.

4.1 DEADLOCK VÀ THUẬT TOÁN PDOA

4.1.1 Deadlock trên cloud

Một trạng thái là an toàn nếu hệ thống có thể cấp phát các tài nguyên tới mỗi quá trình trong một vài thứ tự và vẫn tránh deadlock. Hay nói cách khác, một hệ thống ở trong trạng thái an toàn chỉ nếu ở đó tồn tại một thứ tự an toàn.

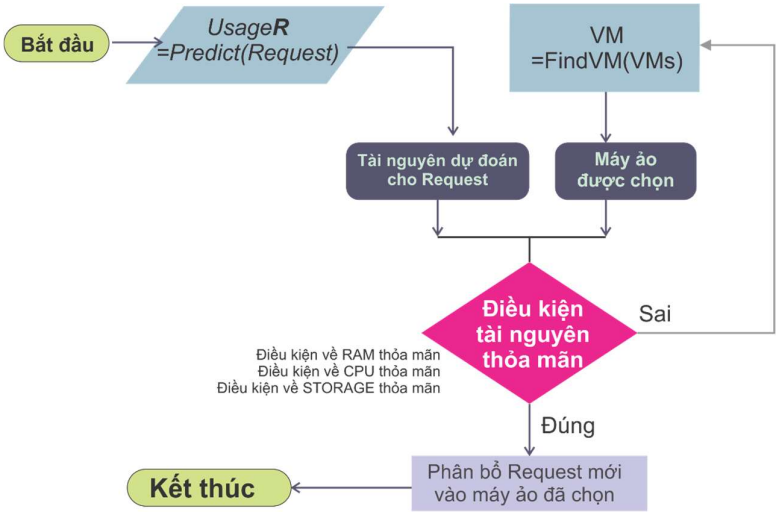
Một trạng thái an toàn không là trạng thái deadlock. Do đó, trạng thái deadlock là trạng thái không an toàn. Tuy nhiên, không phải tất cả trạng thái không an toàn là deadlock. Một trạng thái không an toàn có thể dẫn đến deadlock. Với điều kiện trạng thái là an toàn, hệ điều hành có thể tránh trạng thái không an toàn (và deadlock). Trong một trạng thái không an toàn, hệ điều hành có thể ngăn chặn các quá trình từ những tài nguyên đang yêu cầu mà deadlock xảy ra: hành vi của các quá trình này điều khiển các trạng thái không an toàn.



Hình 4. 1 Không gian trạng thái an toàn, không an toàn, deadlock

Với khái niệm trạng thái an toàn đảm bảo khả năng đáp ứng tốt cho cloud, chúng ta có thể đề xuất và áp dụng các giải thuật dự báo deadlock và khả năng xảy ra deadlock nhằm tránh deadlock. Ý tưởng đơn giản là đảm bảo hệ thống sẽ luôn còn trong trạng thái an toàn. Khởi đầu, hệ thống ở trong trạng thái an toàn. Bất cứ khi nào một quá trình yêu cầu một tài nguyên hiện có, hệ thống phải quyết định tài nguyên có thể được cấp phát tức thì hoặc quá trình phải chờ. Yêu cầu được gán chỉ nếu việc cấp phát để hệ thống trong trạng thái an toàn. Trong mô hình này, nếu quá trình yêu cầu tài nguyên đang có, nó có thể vẫn phải chờ. Do đó, việc sử dụng tài nguyên có thể chậm hơn mà không có giải thuật để biết trước và tránh deadlock. Như vậy, việc đề xuất ra thuật toán dự báo deadlock và tính toán khả năng xảy ra deadlock trên cloud sẽ giúp cho cloud tránh được deadlock, cung cấp dịch vụ tốt hơn cho người dung.

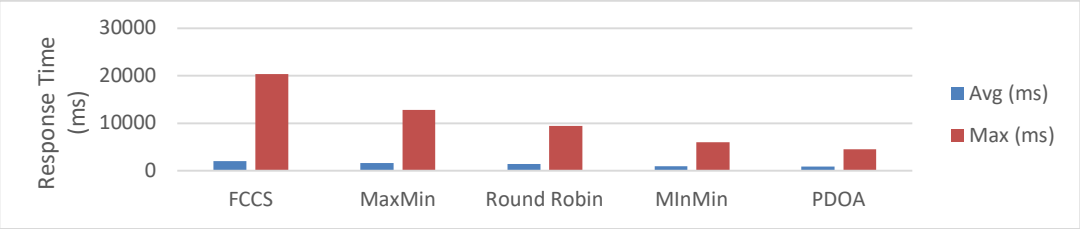
4.1.2 Thuật toán đề xuất PDOA



Hình 4. 2 ***Sơ đồ nguyên lý hoạt động của thuật toán PDOA***

Thực nghiệm

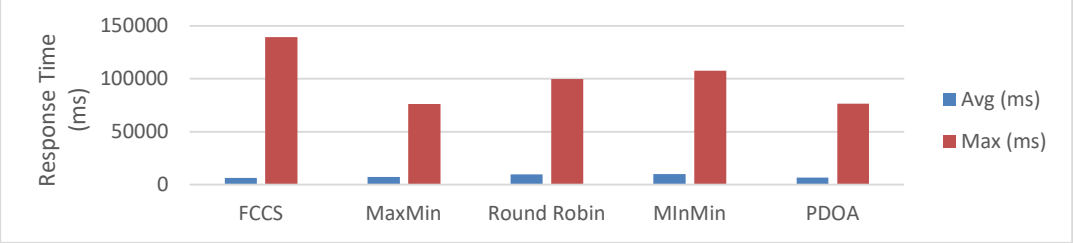
Trường hợp 1 *(Epigenomics_24)***:** thực nghiệm với data 24 request có sẵn của CloudSim, và kết quả như sau:



Hình 4. 3 Biểu đồ so sánh thuật toán PDOA trường hợp 1

Nhận xét: Trong trường hợp 1, ta thấy thuật toán PDOA có vẻ nhỉnh hơn, và có thời gian xử lý trung bình thất nhất, tuy nhiên do lượng request không lớn, nên sự chênh lệch giữa các thuật toán là không nhiều. Thuật toán FCFS thể hiện sự tự nhiên, nên việc thời gian xử lý luôn là cao nhất.

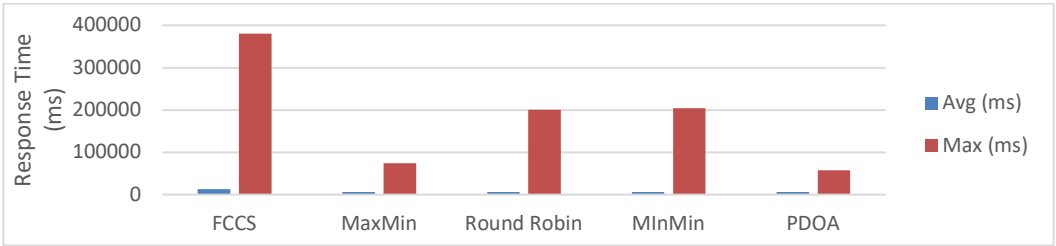
Trường hợp 2 *(Epigenomics_100)***:** thực nghiệm với data 100 request có sẵn của CloudSim, và kết quả như sau:



Hình 4. 4 Biểu đồ so sánh thuật toán PDOA trường hợp 2

Nhận xét: Trong trường hợp 2, ta thấy thuật toán PDOA vượt trội về thời gian xử lý trung bình, nhưng MaxMin lại có thời gian xử lý Max là thấp nhất. Tuy nhiên do lượng request không lớn, nên sự chênh lệch giữa các thuật toán là không nhiều. Thuật toán FCFS thể hiện sự tự nhiên, nên việc thời gian xử lý luôn là cao nhất.

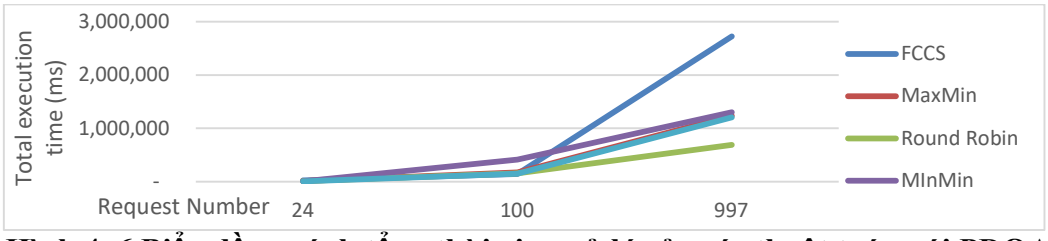
Trường hợp 3 *(Epigenomics_997)***:** thực nghiệm với data 997 request có sẵn của CloudSim, và kết quả như sau:



Hình 4. 5 Biểu đồ so sánh thuật toán PDOA trường hợp 3

Nhận xét: Trong trường hợp 3, ta thấy thuật toán PDOA vượt trội về thời gian xử lý trung bình, và cả thời gian xử lý Max là thấp nhất. Với lượng request tang nhiều,

thấy được tính ưu việt của dự báo và khả năng xử lý của thuật toán dự báo. Thuật toán FCFS thể hiện sự tự nhiên, nên việc thời gian xử lý luôn là cao nhất.



Hình 4. 6 Biểu đồ so sánh tổng thời gian xử lý của các thuật toán với PDOA

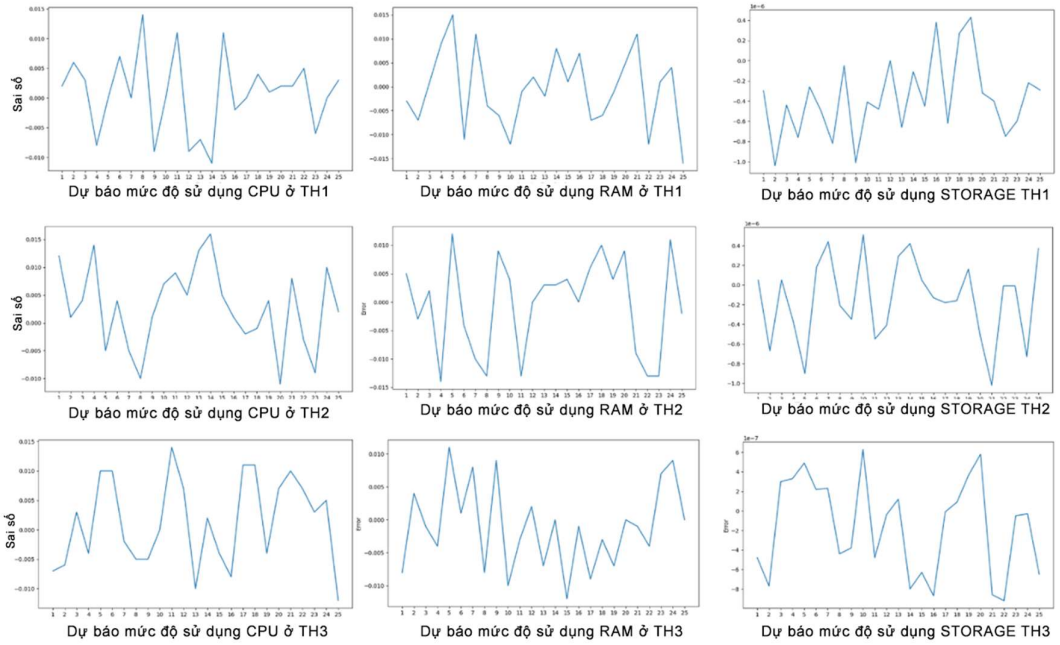
Thuật toán PDOA có kết quả nhỉnh hơn so với các thuật toán điển hình ở một số trường hợp data đầu vào, và không hề thua kém các thuật toán có sẵn về các mặt như thời gian đáp ứng về tổng thời gian xử lý thì luôn ít hơn các kỹ thuật khác.

Đánh giá Mô hình hồi quy tuyến tính trong PDOA

Bảng 4. 1 So sánh RAE của PDOA trong cả 3 trường hợp

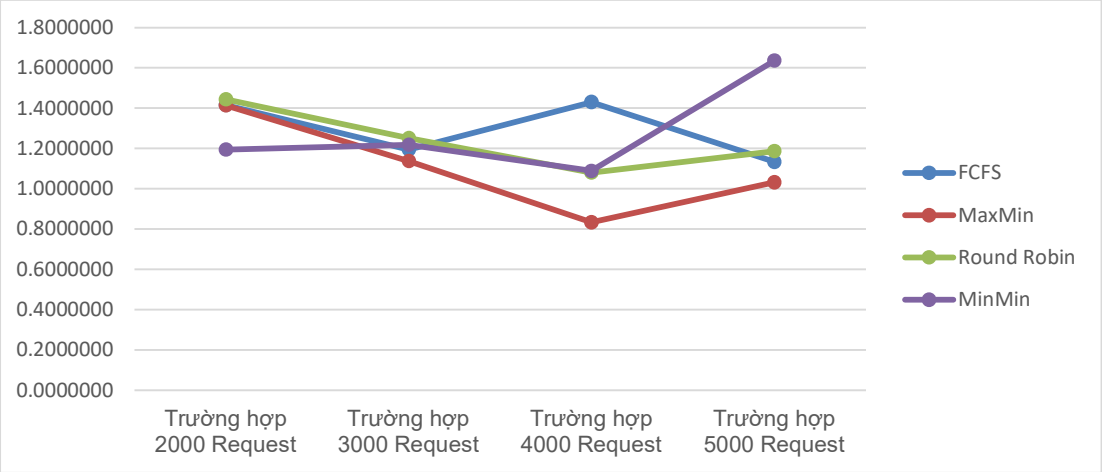
	Sai số RAE		
	Trường hợp 1 (24 request)	**Trường hợp 2 (100 request)**	**Trường hợp 3 (997 request)**
CPU	0.565217	0.489868	0.324261
BỘ NHỚ (RAM)	0.148717	0.286780	0.310741
BỘ LƯU TRỮ	0.002531	0.006759	0.495361

Để đánh giá độ chính xác của Mô hình hồi quy tuyến tính sử dụng trong PDOA, chúng tôi sử dụng sai số tuyệt đối tương đối (RAE) để xem xét mức độ chính xác cho bộ cân bằng tải sử dụng PDOA. Bảng 4.6 cho thấy, RAE xấu nhất và tốt nhất trong dự báo mức độ sử dụng CPU đều xảy ra ở trường hợp 1. Chúng ta có thể thấy rằng, RAE có thể chấp nhận đối với thực nghiệm này, nhưng trong các trường hợp nhìn chung chưa tốt do sự biến thiên của các request. Chúng ta có thể cập nhật và thay đổi bộ dữ liệu lịch sử cho phù hợp, để dự báo chính xác hơn.



Hình 4. 7 Sai số của Regression trong 3 trường hợp (bao gồm CPU, RAM, Storage)

Với cấu hình tương tự, ta ***thực nghiệm với 10 máy ảo***, và tăng số request lên lần lượt là ***2000, 3000, 4000 và 5000***, ta thu được các kết quả khá khả quan. Bên cạnh đó, với số lượng request lớn, luận án bổ sung tham số speedups để tính toán và đánh giá.



Hình 4. 8 Biểu đồ so sánh Speedups các thuật toán với thuật toán PDOA ở 4 trường hợp từ 2000 đến 5000 request

Dựa trên khả năng tăng tốc của PDOA so với các thuật toán khác, PDOA thể hiện khả năng cân bằng tải tốt và có khả năng tăng tốc tương đối so với FCFS, MaxMin và Round Robin trong các trường hợp 2000, 3000 và 5000 Request. Tuy nhiên, PDOA có khả năng tăng tốc thấp hơn so với MinMin trong một số trường hợp.

4.2 HÀNH VI NGƯỜI DÙNG CLOUD VÀ THUẬT TOÁN K-CTPA

4.2.1 Hành vi người dùng cloud với độ ưu tiên tác vụ

Hành vi người dùng cloud đã được chú ý và nghiên cứu, điển hình như Oracle cũng đã nghiên cứu về người dùng cloud và hành vi của người dùng cloud. Hành vi người dùng cloud sẽ dẫn đến những lòng tin trên đám mây, từ đó tạo ra độ ưu tiên của người dùng trên đám mây. *Tác vụ (task)* là một tiến trình hoặc nhiều tiến trình sẽ được thực thi trên một nút tính toán hiện hữu bởi một máy ảo trong điện toán đám mây. *Độ ưu tiên của các tác vụ* là việc xác định mức độ ưu tiên của tác vụ cần thực hiện trong lập lịch tác vụ, vì chúng có ảnh hưởng lớn đến chất lượng dịch vụ mà nhà cung cấp dịch vụ cam kết. Độ ưu tiên của các tác vụ (priority task) phụ thuộc vào nhiều yếu tố như: khả năng sử dụng CPU, RAM, băng thông, chiều dài và kích thước của mỗi tác vụ, thời gian hoàn thành của mỗi tác vụ…việc xác định mức độ ưu tiên của tác vụ có ảnh hưởng rất lớn đến vấn đề lập lịch/cân bằng tải trong môi trường điện toán đám mây một cách tối ưu.

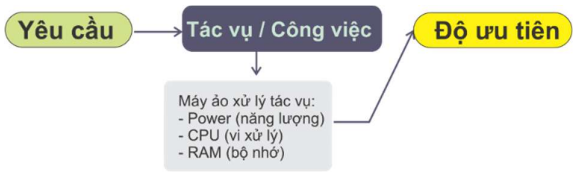
Mô hình nghiên cứu nhằm mục đích sử dụng thuật toán phân lớp kNN (k Nearest Neighbours) để phân loại các task (tác vụ) tương ứng với các Request dựa trên độ ưu tiên xử lý task đó. Ở đây, độ ưu tiên được tính toán dựa trên khả năng tiêu thụ năng lượng của task (Power consumed), mức độ sử dụng CPU (CPU Usages), mức độ sử dụng RAM (RAM Usages) và chi phí (Costing) thực hiện task đó của cloud. Sau khi phân loại các jobs/tasks theo độ ưu tiên, bộ cân bằng tải sẽ phân bổ các request có task với độ ưu tiên cao hơn vào những máy ảo / host có năng lực xử lý tốt hơn, tức là mức độ rảnh task cao. Từ đó, phân bổ request có nhu câu xử lý nhiều vào 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, và ứng dụng trên môi trường cloud theo thời gian thực. Thuật toán đề xuất là k-CTPA (k-NN Classification of Task-Priority Algorithm).

4.2.2 Thuật toán k-CTPA

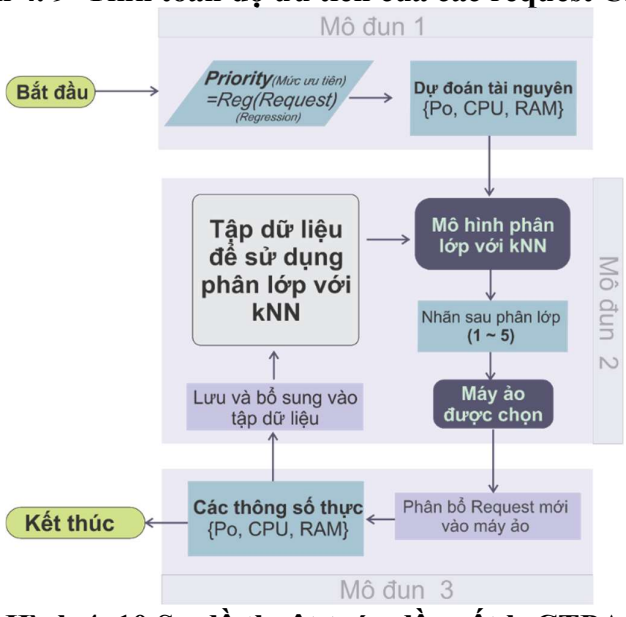
Độ ưu tiên của tác vụ / Task , Độ ưu tiên ở đây là đại lượng được nghiên cứu 3 chiều, tổng hợp từ Power (Power Consume), CPU và RAM

$$\text{Priority} = \{Po, CPU, RAM\}$$

Phân lớp tác vụ / task dựa theo độ ưu tiên, Dựa vào bộ dữ liệu trong quá khứ khi xử lý những task / job đầu tiên, ta sử dụng k-NN để phân lớp độ ưu tiên cho các task tiếp theo. Tương ứng với mỗi yêu cầu (Request) sẽ có một task / job mà máy tính cần phải thực hiện để phục vụ người dùng. Chính vì thế, bất kỳ một request được gửi đến cloud đề có thể được phân lớp dựa trên task tương ứng của nó.



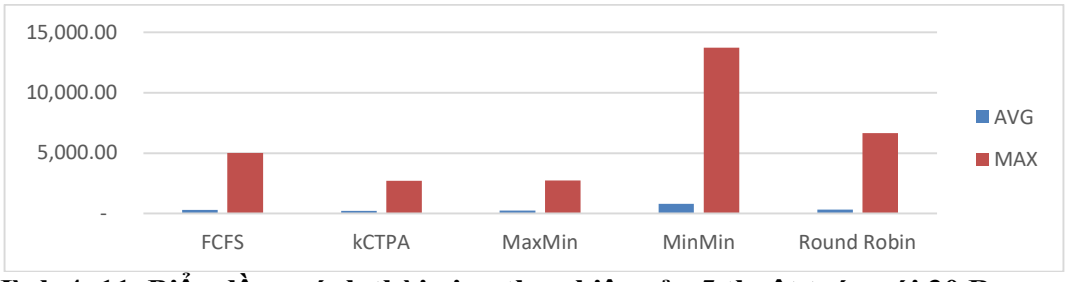
Hình 4. 9 Tính toán độ ưu tiên của các request Cloud



Hình 4. 10 Sơ đồ thuật toán đề xuất k-CTPA

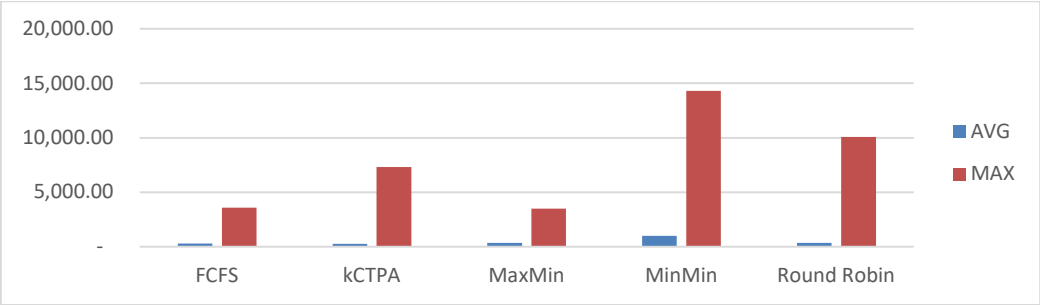
CÀI ĐẶT THUẬT TOÁN k-CTPA

Kết quả chạy thực nghiệm mô phỏng trên CloudSim với 5 máy ảo được dựng sẵn để đáp ứng các yêu cầu, các yêu cầu được khởi tạo với chiều dài và kích thước ngẫu nhiên, số lượng Request lần lượt là 30, 60, 100 và 1000. Sau đó kết quả này được so sánh với các thuật toán Round-Robin, MaxMin, MinMin và FCFS. Đầu tiên ta xét trường hợp với 30 Requets, ta có thời gian thực hiện như hình 4.10.



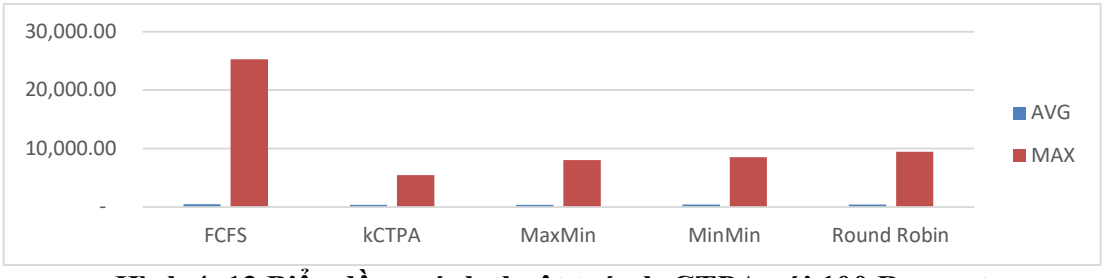
Hình 4. 11 Biểu đồ so sánh thời gian thực hiện của 5 thuật toán với 30 Request

Với kết quả thực nghiệm với 30 Request trở lại, ta thấy thuật toán Round-Robin chiếm ưu thế và xử lý nhanh, thuật toán MaxMin cũng khá ổn định. Thuật toán FCFS thì chưa có thế mạnh. Tuy nhiên thuật toán đề xuất k-CTPA cũng khá ổn định, và chứng tỏ dần ổn định và tốt hơn khi xử lý nhiều request hơn.



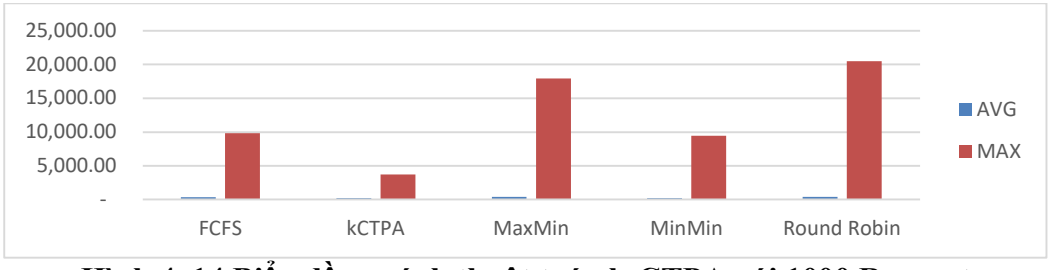
Hình 4. 12 Biểu đồ so sánh thuật toán k-CTPA với 60 Request

Từ request thứ 100 trở đi, thuật toán k-CTPA vượt trội hơn hẳn so với MaxMin, MinMin. Tuy nhiên vẫn chưa thấy ưu thế so với RoundRobin. Nhưng với số lượng request càng lớn thì k-CTPA càng lợi thế hơn hẳn. Và dần dần chiếm ưu thế tuyệt đối so với các thuật toán còn lại. Rõ ràng FCFS thể hiện sự thiếu thông minh và tính tự nhiên của giải thuật.



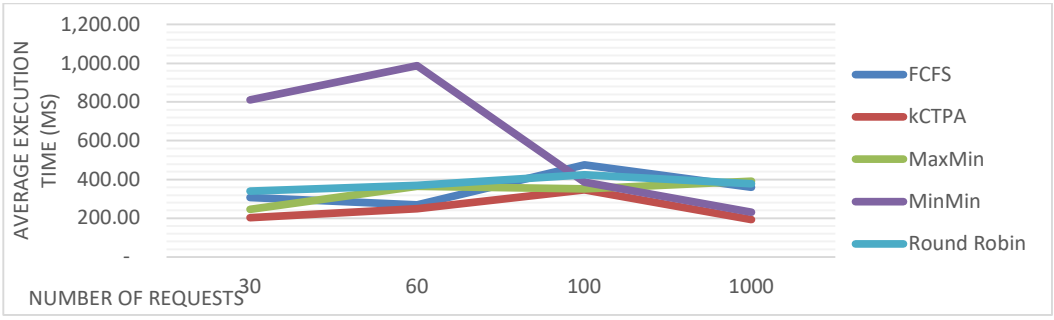
Hình 4. 13 Biểu đồ so sánh thuật toán k-CTPA với 100 Request

Thử nghiệm với 30 đến 100 request, chúng ta thấy thuật toán k-CTPA vượt trội hơn hẳn so với MaxMin, MinMin. Nhưng với số lượng request càng lớn thì k-CTPA càng lợi thế hơn hẳn. Và dần dần chiếm ưu thế tuyệt đối so với các thuật toán còn lại. Rõ ràng FCFS thể hiện sự thiếu thông minh và tính tự nhiên của giải thuật. Chính vì thế ta tăng lên 1000 request:

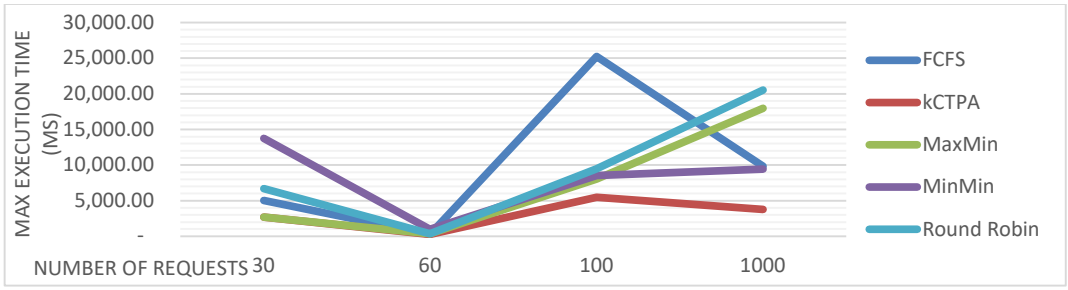


Hình 4. 14 Biểu đồ so sánh thuật toán k-CTPA với 1000 Request

Ở trường hợp 1000 Request ta thấy k-CTPA vươt trội hơn hẳn so với các thuật toán khác, bỏ xa các thuật toán khác.



Hình 4. 15 Thời gian thực hiện trung bình của 5 thuật toán từ 30-1000 Request



Hình 4. 16 Thời gian thực hiện lớn nhất của 5 thuật toán từ 30-1000 Request

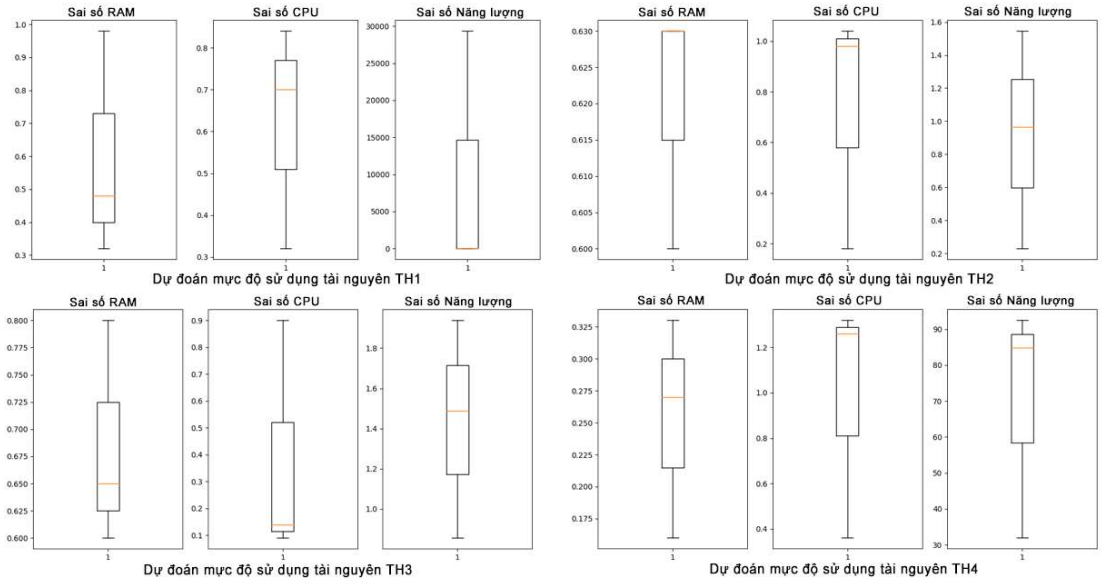
Thông qua 04 trường hợp là 30, 60, 100 và 1000 so sánh thời gian xử lý của các thuật toán với điều kiện như nhau ta có thể thấy sự phân bổ khá ổn định và hợp lý của thuật toán đề xuất k-CTPA, thời gian xử lý của các máy ảo không quá khác biệt so với thời gian xử lý của các thuật toán khác trên cloud (ở trường hợp ít và nhiều request). Hình 4.17 và hình 4.18 cho thấy k-CTPA luôn thấp nhất, kể cả giá trị trung bình lẫn giá trị max.

Đánh giá mô hình Regression trong thuật toán k-CTPA

Để đánh giá độ chính xác của Mô hình hồi quy tuyến tính sử dụng trong k-CTPA, luận án sử dụng số sai số RAE để đánh giá mô hình chạy như thế nào và đưa ra giá trị dự đoán chính xác cho bộ cân bằng tải. Bảng 4.11 cho thấy RAE tồi tệ nhất xảy ra trong dự đoán sử dụng RAM trong trường hợp 1 và tốt nhất là trong trường hợp 3 nhưng đó là dự đoán Mức tiêu thụ nguồn. Chúng ta có thể thấy rằng, RAE có thể chấp nhận được cho thử nghiệm này nhưng nó không tốt ở mọi trường hợp do biến thiên của các request.

Bảng 4. 2 Sai số RAE trong 4 trường hợp

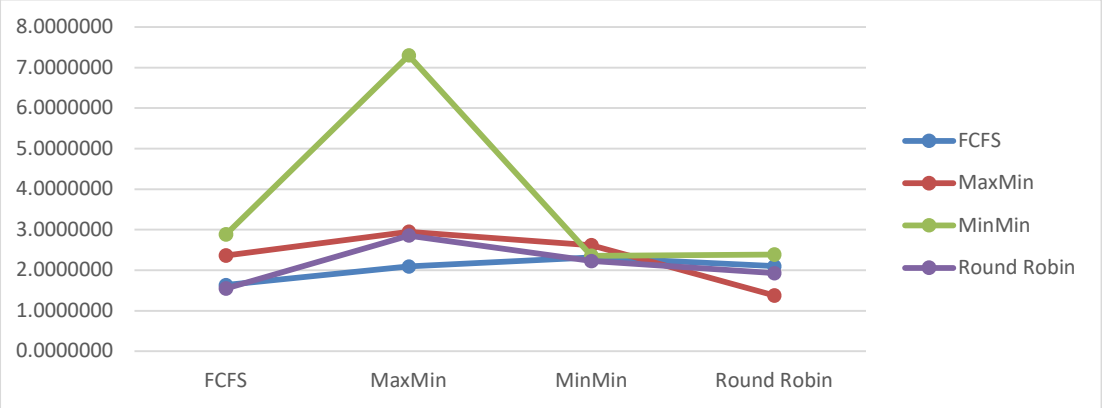
	RAE			
	Trường hợp 1 (30 requests)	Trường hợp 2 (60 requests)	Trường hợp 3 (100 requests)	Trường hợp 4 (1000 requests)
Power Consume	0.085039	0.095735	0.075476	0.084136
CPU Usage	0.295400	0.284387	0.236654	0.248776
RAM Usage	0.326888	0.292676	0.295666	0.300793



Hình 4. 17 Biểu đồ boxplot sai số dự đoán trong 4 trường hợp

Bên cạnh sai số REA, chúng ta có thể sử dụng lỗi dự đoán của Hồi quy tuyến tính để hiểu sâu hơn về độ chính xác của các tài nguyên được dự đoán. Hình 4.19 cho thấy các hộp lỗi của 4 trường hợp. Trong trường hợp 1 và trường hợp 2, sai số vẫn còn cao một chút, nhưng trong trường hợp 3 và trường hợp 4, sai số có vẻ ổn định và giảm xuống. Còn lại, cho thấy các sai số có thể chấp nhận được trong việc dự đoán các tài nguyên này của đám mây.

Với cấu hình tương tự, ta ***thực nghiệm với 10 máy ảo***, và tăng số request lên lần lượt là ***1800, 2700, 3600 và 4500***, ta thu được các kết quả khá khả quan. Bên cạnh đó, với số lượng request lớn, luận án bổ sung tham số speedups để tính toán và đánh giá.



Hình 4. 18 Biểu đồ so sánh Speedups các thuật toán với thuật toán kCTPA ở 4 trường hợp từ 1800 đến 4500 request

Nhìn chung, kCTPA thể hiện khả năng tăng tốc và hiệu năng cân bằng tải tốt hơn so với FCFS và Round Robin trong các trường hợp thực nghiệm. Nó cũng có hiệu năng cân bằng tải cao nhất trong trường hợp 2700 Request và cân bằng tải tương đương

với các thuật toán khác trong các trường hợp khác. Điều này cho thấy kCTPA có tiềm năng để cải thiện hiệu suất và cân bằng tải trong môi trường đám mây.

PHẦN KẾT LUẬN

1. Các kết quả đã đạt được

Việc *nghiên cứu nâng cao hiệu năng cân bằng tải trên điện toán đám mây* đã và đang được nhiều nhà nghiên cứu đặc biệt quan tâm trong thời gian gần đây. Mục tiêu của các công trình nghiên cứu nhằm tối ưu và nâng cao hiệu năng cân bằng tải trên cloud với nền tảng công nghệ hiện đại. Cùng với việc phát triển mạnh mẽ của trí tuệ nhân tạo, tác giả luận án đã tập trung nghiên cứu ứng dụng và phát triển các thuật toán cân bằng tải sử dụng tốt các ưu điểm của trí tuệ nhân tạo điển hình là ML và phân tích dữ liệu trên điện toán đám mây. Qua quá trình học tập, nghiên cứu thực hiện luận án, thông qua phân tích SWOT cân bằng tải, đưa ra hướng tiếp cận cân bằng tải trên môi trường điện toán đám mây, đạt được những kết quả chính sau đây:

Đề xuất xây dựng nhóm thuật toán cân bằng tải theo hướng tiếp cận từ bên trong, bao gồm đề xuất xây dựng và phát triển bộ cân bằng tải kết hợp các tham số của nó với các thuật toán ML và cải tiến thuật toán CBT hiện có. Các thuật toán bao gồm: MCCVA, APRTA, RCBA và ITA.

Đề xuất xây dựng nhóm thuật toán cân bằng tải theo hướng tiếp cận từ bên ngoài, bao gồm các thuật toán liên quan đến Deadlock và hành vi người dùng cloud, đại diện là độ ưu tiên tác vụ (Task priority). Các thuật toán bao gồm PDOA và k-CTPA.

2. Hướng phát triển của đề tài luận án

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

- Tiếp tục cải tiến và phát triển các thuật toán theo 2 hướng tiếp cận, tăng độ chính xác và hiệu năng lên cao hơn nữa trong môi trường cloud biến thiên lớn hơn. Kết hợp phát triển các thuật toán hoặc tổ hợp thuật toán theo cả 02 hướng tiếp cận.
- Tiếp tục nghiên cứu và phát triển theo hướng cơ hội và thách thức, nghiên cứu các yếu tố bên ngoài nhưng tác động mạnh mẽ đến cân bằng tải và hiệu năng hoạt động của nó.

CÁC CÔNG TRÌNH NGHIÊN CỨU CỦA TÁC GIẢ

TẠP CHÍ KHOA HỌC

[CT1] H. Le Ngoc, T. N. Thi Huyen, X. Phi Nguyen, and C. Hung Tran, "MCCVA: A New Approach Using SVM and Kmeans for Load Balancing on Cloud", *International Journal on Cloud Computing: Services and Architecture (IJCCSA)* Vol. 10, No.3, June 2020 DOI: 10.5121/ijccsa.2020.10301, 1-14.

[CT2] N. Xuan Phi, **L. Ngoc Hieu**, and T. Cong Hung, "Load Balancing Algorithm on Cloud Computing for Optimize Respone Time", *International Journal on Cloud Computing: Services and Architecture (IJCCSA)* Vol. 10, No.3, June 2020 DOI: 10.5121/ijccsa.2020.10302 15.

[CT3] Hieu Le Ngoc and Hung Tran Cong, "ITA: The Improved Throttled Algorithm of load balancing on cloud computing," *International journal of computer network and communication.,* vol. 14, no. 1, pp. 25–39, 2022.

[CT4] Hieu Le Ngoc và Hung Tran Cong, "PDOA: dự báo Deadlock để nâng cao cân bằng tải trên điện toán đám mây", *tạp chí Khoa học Công nghệ thông tin và Truyền thông*, PTIT

[CT5] Hieu Le Ngoc and Hung Tran Cong, "Enhancing Load Balancing in Cloud Computing through Adaptive Task Prioritization", *Journal of Computer Science and Technology Studies (JCSTS),* vol. 5, no. 2, 2023, DOI: https://doi.org/10.32996/jcsts.2023.5.2.1.

HỘI NGHỊ KHOA HỌC

[CT6] Hieu Le Ngoc and Hung Tran Cong, "Enhancing Load Balancing in Cloud Computing through Deadlock Prediction", *EAI INISCOM 2023 - 9th EAI International Conference on Industrial Networks and Intelligent Systems*, https://doi.org/10.1007/978-3-031-47359-3_19

[CT7] Hung Tran Cong, Duy Tien Tran and **Hieu Le Ngoc**, "A proposed load balancer using naïve Bayes to enhance response time on cloud computing" in *2022 24th International Conference on Advanced Communication Technology (ICACT)*, 2022