[[1]](#footnote-1)

**FACO - Một mô hình dự đoán cho bài toán tối ưu tài nguyên máy chủ ảo hóa đám mây**

**Trần Thế Anh, La Văn Quân**

*Tóm tắt*—Trong lĩnh vực điện toán đám mây (cloud computing) việc phải dự đoán dữ liệu từ những dữ liệu đã có là việc rất cần thiết. Ví dụ như việc dự đoán nhu cầu sử dụng CPU hay RAM trong thời gian sắp tới để trung tâm máy chủ có chiến lược cấp phát tài nguyên phù hợp là một bài toán rất phổ biến. Bài nghiên cứu này tập trung vào việc xây dựng một mô hình nhằm mục đích tối ưu hóa việc dự đoán nhu cầu sử dụng tài nguyên tại các trung tâm máy chủ. Các bộ dữ liệu được cho dưới dạng bảng thống kê nhu cầu sử dụng tài nguyên tại các điểm thời gian liên tiếp (time series). Nhóm tác giả đã đề xuất một mô hình thuật toán giải quyết vấn đề này và đặt tên nó là FACO. Mô hình sử dụng Fuzzy logic để làm mờ dữ liệu đầu vào trước khi cho qua mạng Neural. Để huấn luyện mạng neural, nhóm tác giả đã đề xuất đưa thuật toán ACO (ant colony optimization) kết hợp cùng BP (back propagation). Bộ dữ liệu được sử dụng trong bài nghiên cứu là bộ dữ liệu Google. Mục tiêu là đi dự đoán nhu cầu sử dụng CPU của một máy chủ tại trung tâm máy chủ. Kết quả thực nghiệm cho thấy mô hình đề xuất này có kết quả hầu như tốt hơn các mô hình hiện tại.

*Từ khóa*— ant colony optimization, cloud computing, fuzzy logic, neural network, predict, time series.

# **1. GIỚI THIỆU**

Công nghệ hiện nay đang không ngừng phát triển theo xu hướng điện toán đám mây. Nền tảng điện toán đám mây đang được áp dụng rộng khắp trong mọi lĩnh vực bởi những ưu điểm vượt trội mà nó mang lại như: sự mềm dẻo (elasticity), tính khả mở (scalability), tính sẵn sàng cao (availability) của hệ thống và đặc biệt là chức năng trả tiền theo sự tiêu dùng tài nguyên thực tế (pay-as-you-go). Trong trung tâm dữ liệu đám mây, thông qua hệ thống các máy ảo, các máy chủ sẽ cung cấp tài nguyên một cách linh hoạt dựa trên nhu cầu tài nguyên từ các ứng dụng. Như vậy trung tâm dữ liệu đám mây có thể tổ chức từ xa nhiều ứng dụng và thực hiện nhiều thao tác cùng 1 lúc, thay vì phải tổ chức các ứng dụng trên các các máy tính riêng. Tuy nhiên, lượng năng lượng tiêu thụ và do đó chi phí tính toán của trung tâm dữ liệu là cao. Bài toán tối ưu hóa lượng năng lượng tiêu thụ của trung tâm dữ liệu đám mây trở thành vấn đề rất được quan tâm bởi các nhà cung cấp dịch vụ điện toán đám mây.

Lượng năng lượng tiệu thụ tương ứng với số máy chủ hoạt động tại 1 thời điểm, càng nhiều máy chủ hoạt động trên đám mây thì lượng năng lượng tiêu thụ tức thì càng lớn. Trong khi đó, yêu cầu tài nguyên đến hệ thống tại từng thời điểm là khác nhau và có thể rất thất thường, một hệ thống với số lượng máy chủ hoạt động ít nhất mà vẫn đáp ứng được lượng nhu cầu tức thì sẽ tiêu tốn ít năng lượng nhất. Hiện nay đã có nhiều công cụ và hướng tiếp cận để tiết kiệm năng lượng cho hệ thống đám mây, nhưng đa số các hệ thống phải duy trì ở trạng thái sẵn sàng đáp ứng nhu cầu tài nguyên lớn nhất để đảm bảo tất cả mức thỏa thuận dịch vụ (Service Level Agreements – SLA) đều được đáp ứng. Số lượng máy chủ hoạt động không đổi dù yêu cầu tài nguyên nhiều hay ít, do vậy chưa tối ưu về mặt năng lượng. Mô hình mà chúng tôi hướng tới vẫn đảm bảo duy trì và không giới hạn các mức thỏa thuận dịch vụ, mở rộng hay thu hẹp hệ thống một cách hợp lí bằng cách dự đoán nhu cầu tài nguyên trong tương lai. Với một mô hình với khả năng dự đoán chính xác nhu cầu tài nguyên trong tương lai, hệ thống sẽ tập trung các tác vụ vào số lượng các máy chủ vật lí hợp lí để từ đó đưa ra quyết định thay đổi trạng thái các máy chủ (tắt/ngủ đông/hoạt động), đáp ứng được các nhu cầu tài nguyên trong tương lai với số lượng máy chủ vật lí hoạt động nhỏ nhất và thế tối ưu việc sử dụng với mức năng lượng thấp nhất. Mặt khác, việc dự đoán cũng thực sự có ý nghĩa khi lượng truy vấn tăng đột ngột, hệ thống sẽ có đủ thời gian để kích hoạt máy chủ, mở rộng tài nguyên để đáp ứng nhu cầu ấy.

Bài toán dự đoán có ý nghĩa trong nhiều lĩnh vực và cũng đã có nhiều phương pháp để cải tiến bài toán này, đặc biệt là ứng dụng mạng nơ-ron kết hợp với các phương pháp tối ưu như giải thuật di truyền (GA – Genetic Algorithm) [1], thuật toán tối ưu đàn ong (ABC - artificial bee colony) [2]…. Trong bài báo này, chúng tôi đề xuất mô hình FACO (Fuzzy and Ant Colony Optimization) tập trung vào giải quyết bài toán dự đoán, cụ thể dự đoán nhu cầu tài nguyên tương lai cần đáp ứng của máy chủ trên hệ thống đám mây. Kĩ thuật mờ hóa (Fuzzy) được sử dụng trong bước tiền xử lí dữ liệu và mạng nơ-ron với phương thức huấn luyện kết hợp giữa giải thuật tối ưu hóa đàn kiến (Ant Colony Optimization) và thuật toán lan truyền ngược (Back Propagation) hi vọng sẽ đưa ra dự đoán chính xác nhất. Kết quả được thực nghiệm trên bộ dữ liệu về thông tin sử dụng CPU của máy chủ Google, mô hình chúng tôi đề xuất sẽ được so sánh với các mô hình khác trên 10 tập dữ liệu có độ dài ngắn khác nhau để minh chứng cho hiệu quả của mô hình này.

# **2. CÁC NGHIÊN CỨU LIÊN QUAN**

Bài toán dự đoán đã nhận được sự quan tâm của rất nhiều nhà nghiên cứu và đã có nhiều công trình tiếp cận bài toán theo nhiều hướng khác nhau, đặc biệt phải kể đến kĩ thuật ứng dụng mạng nơ-ron vào mô hình dự đoán. Mạng nơ- ron với khả năng học thực sự là một công cụ mạnh để giải quyết bài toán dự đoán này, khi mà các giá trị là phi tuyến, biến đổi phức tạp và mối quan hệ giữa các quá trình không thể thiết lập một cách tường minh. [3] là một bài báo đã ứng dụng rất tốt mạng nơ-ron để dư đoán đáp ứng của trung tâp dữ liệu đám mây trong cung cấp tài nguyên động.

Các thuật toán tối ưu cũng được áp dụng trong việc huấn luyện mạng nơ-ron……

Kĩ thuật xử lí dữ liệu trước khi đưa vào mạng nơ-ron cũng là một hướng tiếp cận rất hay. Năm 2005, K. Huarng và T.H-K. Yu trong [5] đã ứng dụng kĩ thuật mờ để tiền xử lí dữ liệu và sử dụng mạng nơ-ron để dự đoán và đã thu được kết quả khá tốt. Năm 2009, trong [6] họ tiếp tục mô hình mờ hóa và mạng nơ-ron để dự đoán nhưng đã cải tiến mô hình bằng cách mờ hóa và dụ đoán sự khác nhau giữa các giá trị liền kề thay vì làm mờ trực tiếp giá trị thời điểm t để dự đoán giá trị tại thời điểm t+1, công trình đã thu được kết quả khả quan hơn.

# **3. MÔ HÌNH HÓA BÀI TOÁN**

Bài nghiên cứu này có sử dụng mô hình bài toán giống như trong [1]. Các bộ dữ liệu được lấy từ bộ dữ liệu Google.

Giả sử tập dữ liệu có tất cả *n* điểm quan sát. Kí hiệu *X* là tập dữ liệu dùng để huấn luyện mạng neural, như vậy ta có *X* = {*x1*, *x2*, … , *xt*} trong đó *xt* là giá trị quan sát được tại thời điểm *t*. *Y* là tập dùng để testing, *Y* = {*yt+1*, *yt+2*, … , *yn*}. Mục tiêu của chúng ta là đi dự đoán các phần tử trong *Y* . Như vậy ta có:

**Đầu vào:**

* Giá trị quan sát được tại thời điểm *k* với *k=1…n*.

**Đầu ra:**

* Giá trị dự đoán tại thời điểm *k*+1.

**Mục tiêu:**

* Mục tiêu của chúng ta là đi cực tiểu hàm sai số, tức là phải dự đoán càng sát thực tế càng tốt.
* Trong bài nghiên cứu này, chúng ta sử dụng hàm trung bình bình phương sai số (RMSA) như đã được đề cập trong [1].

(1)

Trong đó *forecast(i)* là giá trị dự báo tại thời điểm thứ *i*.

**Ràng buộc:**

- Giá trị dự đoán 0 ≤ *forecast(i)* ≤ 100.

# **4. GIẢI PHÁP ĐỀ XUẤT**

Bài nghiên cứu này đề xuất một mô hình dự đoán cho bài toán tối ưu tài nguyên của máy chủ ảo hóa đám mây. Nhóm tác giả đề xuất tiền xử lý dữ liệu bằng bằng đưa chúng vào một hệ thống fuzzy và làm mờ nó. Ngoài ra nhóm tác giả còn sử dụng mạng neural để dự đoán hiệu năng sử dụng CPU. Đầu ra của hệ thống fuzzy được cho làm đầu vào của mạng neural. Đến đây chúng ta hoàn toàn có thể huyến luyện trực tiếp mạng neural bằng thuật toán BP (Back propagation) để cho nó dự đoán. Nhưng tác giả thấy rằng, thuật toán BP có một nhược điểm đó là dễ rơi vào cực trị địa phương nếu pha khởi tạo không tốt. Chính vì vậy mà bài nghiên cứu này còn sử dụng thêm một thuật toán meta-heuristic để tránh rơi vào cục bộ địa phương: đó là thuật toán ACO (ant colony optimization). Nhưng giống như bao thuật toán tiến hóa khác, ACO cũng mắc phải một nhược điểm đó là chỉ tìm ra những lời giải gần tối ưu chứ không thực sự tìm ra lời giải tối ưu, do đó mà nhóm tác giả đề xuất kết hợp thuật toán ACO và BP vào quá trình huấn luyện mạng neural. Sau khi quá trình huấn luyện mạng kết thúc, ta cần có thêm một bước bước defuzzy để giải mã dữ liệu đã được mã hóa ban đầu.

|  |
| --- |
| **Giải thuật**: FACO (fuzzy – ant colony optimization) |
| **Đầu vào**: Tập n giá trị quan sát trong tập dữ liệu.  **Đầu ra**: Giá trị dự đoán *forecast()*  **Begin:**  // Quá trình huấn luyện mạng neural.  **Begin:**   1. **for** *i*=1 to *t* do 2. *input(i)* = Fuzzy(*xi*) 3. **end for** 4. neural\_network = neural\_network(ACO, *zi*) 5. neural\_network = neural\_network(BP, *zi*) 6. **for** i=1 to *t* do 7. *output(i)* = neural\_network(*zi*) 8. *forecast(i)* = Defuzzy(*output(i)*) 9. **end for** 10. **return** neural\_network   **end.**  // Quá trình dự đoán  **Begin:**   1. **for** *i*=*t*+1 to *n* do 2. *forecast(i)* = neural\_network(*xi*) 3. **end for** 4. **return** *forecast(i)* với *i=t+*1*…n*   **end**.  **End**. |

**Hình 1**: Mô hình FACO



# **5. FUZZY**

**Đầu vào**: là các giá trị quan sát được tại thời điểm *t*, *t*-1.

**Đầu ra**: là giá trị của sự khác nhau giữavà được mờ hóa.

Trong bài nghiên cứu này, thay vì dự đoán trực tiếp giá trị trong tương lai, chúng ta dự đoán độ biến thiên giữa các giá trị liên tiếp.

 (2)

Trong đó, ,  là giá trị quan sát được ở thời điểm *t* và *t*-1,  là độ biến thiên của hai mẫu quan sát trên.

Độ biến thiên trên có thể là số âm, ta cần phải chuẩn hóa các giá trị đó để đảm bảo tập giá trị khảo sát chỉ bao gồm các số dương, bằng cách cộng thêm vào các giá trị trên một hằng số dương phù hợp. Khi đó ta có:

 (3)

Gọi ,  là giá trị nhỏ nhất và lớn nhất trong các giá trị của d’

, với mọi *t*

, với mọi *t*

Tập giá trị U khảo sát lúc này sẽ là trong đó ,  là các số dương được chọn phù hợp nhằm mục đích các giá trị của d’ đều thuộc U và không rơi vào trường hợp đặc biệt trùng với giá trị 2 biên. Ta chia tập giá trị U thành k đoạn bằng nhau, kí hiệu là , ,…, , giả sử chiều dài mỗi khoảng là . Ta có:





…



Giá trị chính giữa của mỗi đoạn là:





…



Như vậy, tập giá trị U được chia thành các tập mờ: ,,…, và ,,…, lần lượt là các giá trị đặc trưng cho các khoảng mờ ấy.

Giá trị  được mờ hóa trở thành  dạng:



Trong đó, các giá trị  nằm trong [0, 1] và thỏa mãn:

 (4)

Có nhiều giá trị giá trị của  ứng với một giá trị  thỏa mãn (1). Để đơn giản, trong bài nghiên cứu này này, ta thiết lập nguyên tắc mờ hóa như sau:

Khi  thuộc tập mờ thì ta có:





**6. PHA ACO-BP (Ant Colony Optimization – Back Propagation)**

ACO là một thuật toán mô phỏng hệ thống sinh học. Nó được lấy cảm hứng từ việc tìm kiếm nguồn thức ăn của bầy kiến trong tự nhiên. Các con kiến có thể tìm ra đường đi ngắn nhất từ tổ tới nguồn thực phẩm mặc dù thông tin mà chúng biết gần như bằng không. Điều này được thực hiện bằng cách sử dụng pheromone như một hình thức truyền thông gián tiếp. Con đường nào càng có nhiều pheromone thì càng có hy vọng tìm thấy nguồn thực phẩm và những con kiến sẽ ưu tiên đi theo con đường đó hơn. Dựa vào những đặc tính tuyệt với trên, thuật toán ACO được phát triển và hoàn thiện hơn. ACO sử tri thức của quần thể để lựa chọn hướng đi tốt nhất, trong khi đó, GA thực hiện tiến hóa một cách ngâu nhiên. Chính vì điều này mà chúng ta có thể kì vọng rằng khả năng tìm kiếm của ACO là nhanh hơn GA.

Trong bài nghiên cứu này, nhóm tác giả sử dụng ACO kết hợp với BP để huấn luyện mạng neural. Như chúng ta đã biết, nếu chỉ sử dụng BP để huấn luyện mạng neural thì rất dễ dẫn tới cực trị địa phương, do đó ACO giống như là một hình thức để giúp tìm kiếm lời giải toàn cục.

*6.1. Mô hình hóa mạng neural.*

Trong bài nghiên cứu này, nhóm tác giả sử dụng mạng neural có 3 lớp, gồm một lớp input, một lớp output và một lớp ẩn.

Lớp input có nhận dữ liệu vector . Mỗi một thành phần (*i=1…k*) của tương ứng với một neural. Như vậy lớp input sẽ có *k* neural.

Lớp output xuất ra dữ liệu là vector . Mỗi thành phần (*i=1…k*) của tương ứng với một neural. Như vậy lớp output cũng sẽ có *k* neural.

Bằng thực nghiệm, nhóm tác giả thấy rằng số lượng neural của lớp ẩn gấp 2 lần số lượng sensor của lớp input và output thì kết quả đưa ra là tốt nhât. Điều này cũng đã được nhắc tới trong [1]. Như vậy, số lượng neural của lớp ẩn là *2k.*

[1] sử dụng hàm RMSA là hàm mất mát huấn luyện mạng neural. Trong bài nghiên cứu này, chúng ta cũng sử dụng hàm RMSA. Hàm được cho bởi công thức (1).

*6.2. Thuật toán ACO.*

Gọi *W* là tập các trọng số của mạng neural. Do lớp input có *k* neural, lớp ẩn có 2*k* neural, lớp output có *k* neural nên *W* được mã hóa là một mảng có 4*k2*phần tử. Ta mã hóa mỗi một cá thể kiến là một giá trị của *W*. Trong nghiên cứu này, nhóm tác giả thực hiện khởi tạo quần thể kiến với 150 cá thể, các cá thể được sinh ngẫu nhiên. Tức là các giá trị của *W* được sinh ngẫu nhiên. Như vậy ta có một tập lời giải {*w1, w2,…,w150*} tương ứng với 150 cá thể kiến.

Với mỗi cá thể kiến, tại lần lặp thứ n sẽ sinh lời giải *wi* tiếp theo bằng một phân phối chuẩn . Các tham số tương ứng là trong lượng , phương sai , kì vọng .

**7. DE-FUZZY**

**Đầu vào**: , là giá trị mờ dự đoán được của sự khác nhau giữa và

**Đầu ra**: , giá trị dự đoán được tương ứng với 

Ta có:



 (5)

Giá trị dự đoán được của sự khác nhau giữa hai thời điểm t+1 và t là:

 (6)

Giá trị dự đoán được tại thời điểm t+1 :

 (7)

**8. KẾT QUẢ THỰC NGHIỆM**

*8.1. Cài đặt thực nghiệm*

Trong phần thực nghiệm, nhóm nghiên cứ đã sử dụng bộ dữ liệu Google. Trong đó lấy số liệu phần trăm sử dụng CPU của một máy chủ Google để dự đoán.

Các mô phỏng được thực hiện trên hệ thống:

- Hệ điều hành Ubuntu 14.04, 64 bit.

- RAM 8GB

- Intel core i7, CPU 2.50 GHz

Nhóm nghiên cứu sử dụng 10 bộ test được kí hiệu là Testi với i=1…10. Mỗi bộ test được chia thành 2 phần, một phần dùng để huấn luyện, phần còn lại dùng để dự đoán. Trong nghiên cứ này, để đảm bảo rằng thuật toán là ổn định, nhóm tác giả đã thực hiện chạy thuật toán 10 lần cho mỗi bộ test rồi lấy trung bình. Các kết quả được cho như trong bảng dưới đây.

**Bảng 1**: Bộ dữ liệu thực nghiệm

|  |  |  |
| --- | --- | --- |
| **Bộ Test** | **Training (mẫu)** | **Testing (mẫu)** |
| Test1 | 200 | 100 |
| Test2 | 400 | 200 |
| Test3 | 600 | 300 |
| Test4 | 800 | 400 |
| Test5 | 1000 | 500 |
| Test6 | 1200 | 600 |
| Test7 | 1400 | 700 |
| Test8 | 1600 | 800 |
| Test9 | 1800 | 900 |
| Test10 | 2000 | 1000 |

*8.2. Kết quả thực nghiệm*

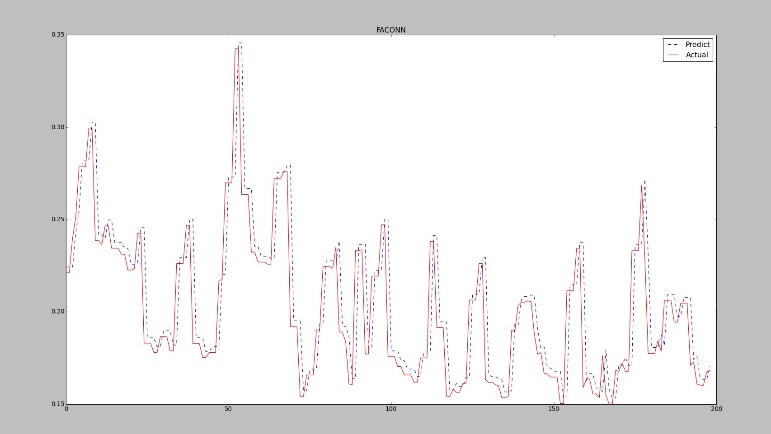
Trong bài nghiên cứu này, các tác giả đã cài đặt lại các mô hình dự đoán NN (neural network), ACO-NN (ant colony optimization – neural network), FNN (fuzzy – neural network) để lấy cơ sở so sánh với mô hình đề xuất của nhóm tác giả là FACO (fuzzy – ant colony optimization).

Các kết quả được tóm tắt lại trong bảng 2. Bảng 2 thống kê lại giá trị hàm sai số trung bình RMSA của các thuật toán sau khi chạy mỗi bộ test.

**Bảng 2**: kết quả thống kê hàm RMSA đối với 4 mô hình NN, ACO-NN, FNN, FACO. Mỗi bộ dữ liệu được chạy 10 lần rồi lấy kết quả trung bình.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bộ test | NN (x10^-4) | ACO-NN (x10^-4) | FNN (x10^-4) | FACO-NN (x10^-4) |
| Test1 | 13.25 | 12.87 | 6.81 | 6.79 |
| Test2 | 8.58 | 8.25 | 6.25 | 5.91 |
| Test3 | 8.87 | 8.35 | 8.70 | 8.13 |
| Test4 | 9.23 | 9.22 | 9.03 | 8.62 |
| Test5 | 8.18 | 7.99 | 9.04 | 8.76 |
| Test6 | 10.13 | 10.14 | 10.12 | 9.77 |
| Test7 | 9.19 | 9.11 | 9.67 | 8.64 |
| Test8 | 8.24 | 8.08 | 8.18 | 8.34 |
| Test9 | 7.73 | 7.69 | 8.18 | 8.08 |
| Test10 | 10.60 | 11.77 | 8.87 | 8.49 |

**Hình 2**: Bảng dự đoán phần trăm sử dụng CPU của một máy chủ tại trung tâm máy chủ sử dụng giải thuật FACO.



Từ kết quả trên có thể nhận thấy rằng mô hình đề xuất FACO hầu như đưa ra được kết quả tốt nhất. Trong khi đó các thuật toán trong [1] thường đưa ra kết quả thấp hơn và độ ổn định không cao.

Sự có mặt của kĩ thuật mờ cho thấy kết quả tốt so với các mô hình khác, mạng nơ-ron là phù hợp để tính toán các mối quan hệ mờ của chuỗi dữ liệu theo thời gian. Bên cạnh đó, như chúng ta đã biết, thuật toán BP là một thuật toán dùng để tìm cực trị rất tốt, nhưng nhược điểm vô cùng lớn của nó lại là rất dễ rơi vào cực trị địa phương nếu giá trị khởi tạo không tốt. Và thuật toán ACO là một giải pháp tốt để khắc phục nhược điểm này, kết quả thực nghiệm cho thấy những mô hình nào có sự góp mặt có sự góp mặt của ACO thì đầu ra rất ổn định. Ngược lại những mô hình không có ACO thường có kết quả kém ổn định hơn.

**9. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN TRONG TƯƠNG LAI**

Trong bài nghiên cứu này định hướng dự đoán nhu cầu sử dụng tài nguyên máy chủ nhằm tối thiểu hóa hàm sai số RMSE. Nhóm tác giả đã đề xuất một mô hình giải quyết bài toán và đặt tên là FACO (sự kết hợp giữa Fuzzy logic, ACO algorithm và BP algorithm) nhằm huấn luyện mạng neural được tốt hơn. Đối với bộ dữ liệu đã sử dụng, kết quả thực nghiệm cho thấy FACO đưa ra sai số dự đoán thấp hơn so với những thuật toán khác, đồng thời nó cũng đạt được độ ổn định rất cao. Trong tương lai, nhóm sẽ tiếp tục phát triển mô hình của mình bằng cách dự đoán đa mục tiêu (dự đoán với nhiều thông số đầu vào như CPU, RAM và nhiều đầu ra).

**10. TÀI LIỆU THAM KHẢO**

1. Shifei Ding, "*An optimizing BP neural network algorithm based on genetic algorithm*," in Artificial Intelligence review, August 2011.
2. Beatriz A. Garro, Humberto Sossa, Roberto A. V´azquez. ‘*Artificial Neural Network Synthesis by means of Artificial Bee Colony (ABC) Algorithm*’, 2011.
3. Vazquez, Krishnan, and John, “*Time Series Forecasting of Cloud Data Center Workloads”*,2015
4. Kunhuang Huarng, Tiffany Hui-Kuang Yu, “*The application of neural networks to forecast fuzzy time series*”, 2005
5. Tifany Hui-Kuang Yu, Kun-Huang Huarng, *"A neual network-based fuzzy time series model to improve forecasting*", 2009 Elsevier, All rights reserved.
6. Timothy J. Ross, *"Fuzzy logic with engineering applications*", 2004.
7. Simon Haykin, "*Neural network and learning machines",* third edition.
8. Avi Ostfeld, “*Ant Colony Optimization – Methods and Application”,* Janeza Trdine 9, 51000 Rijeka, Croatia.
9. Khanh Nguyen, Hieu Pham, "*Những vấn đề cơ bản về mạng neural*", unpublished.
10. D. Michie, D. J. Spiegelhalter, C. C. Taylor, "*Machine learning, Neural and statistical classification*", February 17, 1994

1. Công trình này được thực hiện dưới sự hướng dẫn của TS. Nguyễn Bình Minh.

   Trần Thế Anh, sinh viên lớp KSTN\_CNTT, khóa 58, Viện Công nghệ thông tin và Truyền thông, trường Đại học Bách Khoa Hà Nội (e-mail: [anhtt17895@gmail.com](mailto:anhtt17895@gmail.com))

   La Văn Quân, sinh viên lớp KSTN\_CNTT, khóa 58, Viện Công nghệ thông tin và Truyền thông, trường Đại học Bách Khoa Hà Nội (e-mail: [lavanquan.kstn.cntt@gmail.com](mailto:lavanquan.kstn.cntt@gmail.com))

   **© Viện Công nghệ thông tin và Truyền thông, trường Đại học Bách Khoa Hà Nội.** [↑](#footnote-ref-1)