TRÙNG LẶP CÁ THỂ TRONG LẬP TRÌNH DI TRUYỀN

Phạm Thị Thương[1*], Nguyễn Xuân Hoài[2], Nguyễn Thị Hiền[3], Ngô Văn Mạnh[4]
[1]Trường Đại học Công nghệ thông tin & truyền thông - ĐH Thái Nguyên,
[2]Viện trí tuệ nhân tạo Việt Nam, [3]Học viện Kỹ thuật quân sự,
[4]Trung tâm Thông tin và Dữ liệu Khí tượng thủy văn

TÓM TẮT

Trong thực tế, mọi cá thể xuất hiện trong thế giới tự nhiên là duy nhất. Chúng kế thừa đặc tính di truyền từ cha mẹ, đồng thời cũng mang những nét đặc trưng riêng biệt mà không giống bất kỳ một cá thể nào đã và đang tồn tại (Adam Rutherford, 2018). Lập trình di truyền (GP) là một trong các cách tiếp cận mô phỏng sự tiến hóa của tự nhiên và đã được áp dụng thành công trong nhiều lĩnh vực. Vậy, (1) *Vấn đề trùng lặp đã được giải quyết như thế nào trong GP? (2) Việc lặp cá thể có phụ thuộc vào kích cỡ quần thể không? Nó tác động như thế nào đến hiệu quả của GP? (3) Nguyên nhân gây trùng lặp là gì? và (4) Làm thế nào để giải quyết vấn đề trùng lặp?* Để trả lời các câu hỏi nghiên cứu này, chúng tôi đã tiến hành các thực nghiệm. Kết quả cho thấy, trùng lặp cá thể không bị tác động nhiều bởi kích cỡ quần thể trên đa phần các bài toán được thử nghiệm; giải quyết vấn đề trùng lặp giúp cải tiến một cách đáng kể hiệu suất của GP nói riêng và các cách tiếp cận dựa trên GP nói chung.

Từ khóa: *Lập trình di truyền; giải thuật tiến hóa; máy học; hệ gen; lặp cá thể*

Ngày nhận bài: 05/5/2020; Ngày hoàn thiện: 29/8/2020; Ngày đăng: 31/8/2020

INDIVIDUAL DUPLICATION IN GENETIC PROGRAMMING

Pham Thi Thuong[1*], Nguyen Xuan Hoai[2], Nguyen Thi Hien[3], Ngo Van Manh[4]
[1]TNU - University of Information and Communication Technology
[2]AI Academy, VietNam, [3]LeQuyDon Technical University
[4]Center for Hydro-Meteorological Data and Information

ABSTRACT

In reality, each individual that appears in the natural world is unique. They inherit genetic meterials from their parents, and carry distinct traits that do not resemble any existing and existed individuals (Adam Rutherford, 2018). Genetic programming (GP) is one of the approaches to simulate the natural evolution that has been successfully applied in many fields. So, *(1) How is the problem of individual duplication solved in GP? (2) Does this depend on the population size? How does it affect the GP? (3) What are the causes of duplication? and (4) How to solve this problem?* In order to answer these questions, we have run experiments. The results show that individual duplication not be effected by the population size with the most tested problems. Solving this problem will significantly improve the performance of GPs in particular and GP-based approaches in general.

Keywords: *Genetic programming; evolutionary algorithms; machine learning; genome; duplicate individuals.*

Received: 05/5/2020; Revised: 29/8/2020; Published: 31/8/2020

* Corresponding author. *Email: ptthuong@ictu.edu.vn*

1. Giới thiệu

Trong thế giới tự nhiên, mọi cá thể xuất hiện là duy nhất. Mỗi cá thể kế thừa những đặc tính di truyền từ cha mẹ, đồng thời cũng mang những nét đặc trưng riêng biệt mà không giống bất kỳ một cá thể nào đã và đang tồn tại [1]. Lập trình di truyền (GP) là một trong các cách tiếp cận mô phỏng hành vi của tiến hóa trong thế giới tự nhiên đã được phát triển bởi Koza [2] năm 1992. Nó dựa trên quan sát về các hệ thống sinh học và sử dụng các cơ chế lựa chọn tự nhiên của Darwins để tiến hóa quần thể các giải pháp cho các bài toán cần giải quyết. GP là một trong các cách tiếp cận hiệu quả để giải quyết các bài toán thuộc nhiều lĩnh vực khác nhau, trong đó học máy là một trong các nhiệm vụ chính [3]. Việc đảm bảo tính duy nhất của các giải pháp trong quần thể tiến hóa của GP phản ánh đúng tự nhiên là một vấn đề đáng quan tâm. Lặp cá thể trong quần thể là một trong những nguyên nhân dẫn đến lãng phí tài nguyên tính toán và làm giảm tính đa dạng trong quần thể [2].

Một số nghiên cứu gần đây về GP tập trung giải quyết vấn đề trùng lặp cá thể như [4]-[7]. Tuy nhiên trong các nghiên cứu này, hầu như các tác giả đều xem vấn đề trùng lặp cá thể như là một trong những nguyên nhân gây lãng phí tài nguyên tính toán. Do vậy, các cách tiếp cận được đề xuất bởi họ chỉ nhằm hướng tới việc giảm bớt các tính toán gây ra do trùng lặp cá thể; hoặc chỉ hướng tới khắc phục vấn đề trùng lặp trong quá trình khởi tạo quần thể mà không nghiên cứu một cách có hệ thống các nguyên nhân chính gây trùng lặp là gì, không xem việc khắc phục trùng lặp như một sự thật hiển nhiên trong thế giới tự nhiên và điều này tác động trực tiếp đến hiệu quả của GP. Trong [8], Miguel Nicolau và cộng sự đã nghiên cứu về tần suất và các tác động của lặp cá thể trên hai hệ thống GP văn phạm gồm Hệ thống tiến hóa văn phạm (Grammatical Evolution, GE) và Hệ thống GP văn phạm phi ngữ cảnh (Context-Free Grammar GP, CFG-GP). Để quản lý vấn đề trùng lặp, nhóm tác giả đã chạy các thực nghiệm trên ba bài toán chuẩn, sử dụng bốn cách tiếp cận: (1) Không sử dụng Tabu list – cho phép lặp cá thể trong quần thể; (2) Tabu list dò tìm fitness - cho phép lặp cá thể nhưng không đánh giá cá thể lặp lại để tiết kiệm thời gian; (3) Tabu list như danh sách phạt – cho phép lặp cá thể, nhưng các cá thể lặp bị phạt gán giá trị fitness tồi nhất để hạn chế bị lựa chọn trong quá trình tiến hóa; và (4) Tabu list như danh sách ép buộc – không cho phép lặp cá thể, nếu lặp thì làm lại. Tuy nhiên, cũng tương tự như các nghiên cứu trước đó, nhóm tác giả cũng không xem việc khắc phục trùng lặp như một sự thật hiển nhiên trong thế giới tự nhiên. Mặc dù cách tiếp cận (4) của nhóm tác giả không cho phép trùng lặp diễn ra nhưng cũng mới chỉ tập trung vào trùng lặp xảy ra khi khởi tạo quần thể bằng cách, nếu phát hiện cá thể mới được tạo trùng với các cá thể đã tồn tại thì tạo lại.

Trong phạm vi bài báo này, chúng tôi tập trung trả lời các câu hỏi nghiên cứu sau:

1. Vấn đề trùng lặp cá thể trong các quần thể của GP bị tác động như thế nào khi kích cỡ quần thể thay đổi?

2. Những nguyên nhân chính dẫn đến trùng lặp là gì? Điều này ảnh hưởng như thế nào đến hiệu quả của GP?

3. Cách thức giải quyết vấn đề trùng lặp trong GP và hiệu quả của giải pháp đề xuất là gì?

Chúng tôi đã tiến hành chạy các thực nghiệm để trả lời các câu hỏi nghiên cứu trên. Kết quả cho thấy: (1) Hiện tại có sự trùng lặp cá thể (chrom) và về mặt ngữ nghĩa (fitness) trong các quần thể của GP. Khi tăng kích cỡ quần thể của GP lên 500, 1000, 1500, kết quả cho thấy, số lượng cá thể trùng lặp không bị ảnh hưởng đáng kể bởi kích cỡ quần thể với đa phần các bài toán được thử nghiệm, ngoại trừ một số bài toán có xu hướng giảm trùng lặp khi kích cỡ quần thể tăng. (2) Qua phân tích, chúng tôi xác định được nguyên nhân chính gây ra trùng lặp là do sự tác động của yếu tố ngẫu nhiên trong khởi tạo quần thể, trong quá trình lựa chọn và lai ghép để tiến hóa giải pháp. (3) Giải quyết vấn đề trùng lặp giúp nâng cao khả năng khái quát hóa của GP với đa phần các bài toán được thử nghiệm.

Phần còn lại của bài báo được tổ chức như sau: Trong phần 2, trình bày ngắn gọn về cách tiếp cận được sử dụng để giải quyết các câu hỏi nghiên cứu và các thiết lập thực nghiệm được sử dụng trong nghiên cứu. Tiếp theo, phần 3 chỉ ra các kết quả và thảo luận. Cuối cùng, bài báo kết thúc bởi phần kết luận và các hướng nghiên cứu trong tương lai.

2. Phương pháp nghiên cứu

2.1. Lập trình di truyền và yếu tố ngẫu nhiên

Lập trình di truyền (GP) được xem là một phương pháp máy học, mục đích tối ưu quần thể các giải pháp/ chương trình biểu diễn một nhiệm vụ tính toán cho trước. GP gồm các bước như sau:

Bước 0*: Khởi tạo quần thể ban đầu, P(0).*

Lặp:

Bước 1*: Đánh giá độ thích nghi/ độ tốt của mỗi lời giải trong quần thể tại thế hệ t, P(t).*

Bước 2*: Lựa chọn 2 lời giải cha trong quần thể P(t) dựa trên độ thích nghi của chúng.*

Bước 3*: Thực hiện các thao tác di truyền (lai ghép, đột biến, tái tạo) để thu được quần thể P(t+1)*

Lặp đến tận khi các điều kiện dừng thỏa mãn (tìm được giải pháp tối ưu/ đạt đến số thế hệ cho trước).

Độ thích nghi của lời giải được thay bởi hàm lỗi RMSE (xem bảng 2). Mục đích của GP là tiến hóa quần thể và tìm giải pháp có độ thích nghi cao nhất hay giá trị lỗi là nhỏ nhất.

Qua phân tích chúng tôi nhận thấy sự trùng lặp cá thể trong quần thể GP phần nhiều liên quan đến yếu tố ngẫu nhiên. Có thể nói, với GP ngẫu nhiên đóng một vai trò nhất định. Nó góp phần quan trọng trong việc tăng tính đa dạng của quần thể GP – đa dạng là một trong hai đặc tính quan trọng (tính đa dạng, tính hội tụ) ảnh hưởng trực tiếp tới hiệu quả của GP nói riêng và các giải thuật tiến hóa (EA) nói chung [9], [10].

Để đảm bảo tính đa dạng, GP thường bắt đầu bởi một quần thể với các giải pháp được sinh ngẫu nhiên. Việc tiến hóa quần thể qua các thế hệ được thực hiện bằng cách lựa chọn ngẫu nhiên các cá thể để tái tạo, lai ghép, đột biến với xu hướng các cá thể tốt hơn có xác suất được lựa chọn ưu tiên nhiều hơn các cá thể khác. Tương tự, việc lựa chọn các điểm lai ghép trên cặp cá thể cha - mẹ trong quá trình lai ghép diễn ra cũng thường là ngẫu nhiên. Trong quá trình đột biến, thông thường điểm đột biến cũng được xác định một cách ngẫu nhiên. Đột biến được thực hiện bằng cách thay thế một phần ngẫu nhiên của chương trình/ cá thể với một phần ngẫu nhiên khác của nó, hoặc bởi một phần chương trình được sinh ngẫu nhiên tại điểm đột biến đã xác định. Các cặp cha – mẹ lai ghép không thành công có thể được sao chép từ thế hệ cũ đến thế hệ mới.

Có thể nói, với GP, ngẫu nhiên đóng một vai trò quan trọng quyết định tính đa dạng trong quần thể tiến hóa. Tuy nhiên, nó cũng được dự đoán là nguyên nhân chính dẫn đến sự trùng lặp cá thể trong quần thể GP.

2.2. Kiểm tra tính duy nhất của cá thể trong quần thể tại mỗi thế hệ

Cấu trúc giải pháp trong quần thể của GP thường được biểu diễn sử dụng cấu trúc cây. Nhiễm sắc thể (chrome) của cá thể đại diện cho cấu trúc giải pháp và nó ảnh hưởng trực tiếp đến ngữ nghĩa của chương trình. Chrome mang những nét đặc trưng riêng, đại diện cho hệ gen của cá thể và theo tiến hóa tự nhiên, nó phải là duy nhất. Để kiểm tra tính duy nhất của mỗi cá thể trong quần thể tại mỗi thế hệ, chúng tôi tiến hành theo các bước như sau:

- Tại mỗi thế hệ, chuyển chrome dạng tree của mỗi cá thể thành xâu (String), lưu các xâu chuyển đổi tương ứng vào trong một mảng.

- Sắp xếp mảng tăng dần, đếm số lần lặp xâu và cộng dồn. Kết quả chúng ta thu được là tổng số cá thể có sự trùng lặp trong quần thể tại mỗi thế hệ.

- Lấy trung bình phần trăm của tổng số cá thể bị trùng lặp qua 31 lần chạy ta thu được đồ thị biểu diễn sự trùng lặp qua các thế hệ trong quá trình tiến hóa.

2.3 Phân tích sự tác động của kích cỡ quần thể lên vấn đề trùng lặp cá thể

Để phân tích những tác động của kích cỡ quần thể lên số lượng cá thể trùng lặp, chúng tôi tiến hành ba thực nghiệm với kích cỡ quần

thể tăng lần lượt là 500, 1000, 1500 và giữ nguyên các thiết lập thực nghiệm khác. Sau đó, với mỗi thực nghiệm, thống kê trung bình % cá thể lặp tại mỗi thế hệ trong quá trình tiến hóa qua 31 lần chạy.

2.4. Phân tích xác định các nguyên nhân gây trùng lặp cá thể

Như đã phân tích tại mục 2.1, nguyên nhân chính dẫn đến sự trùng lặp cá thể trong GP được dự đoán là do sự tác động của yếu tố ngẫu nhiên. Trong thế giới tự nhiên, mọi thứ tồn tại, được sinh ra hầu như sắp xếp theo một trật tự nhất định, một trật tự cân bằng hoàn hảo và vận hành theo một thể thống nhất trong vũ trụ, trong đó con người là hoa, trái trong một cái cây vũ trụ. Trật tự tối ưu này đã được tiến hóa qua hàng trăm ngàn năm tồn tại, tương tự như hệ gen của mỗi chúng ta và ngẫu nhiên chỉ đóng một vai trò thứ yếu. Tuy nhiên, trong GP, ngẫu nhiên lại mang tính chủ đạo trong việc tăng tính đa dạng của quần thể và gây ra vấn đề trùng lặp cá thể, điều này dẫn đến vi phạm một trong các quy luật của tự nhiên – mọi sự sống, mỗi cá thể là duy nhất, không ai giống ai, tương tự như hệ gen, hoặc vân tay của mỗi con người là duy nhất.

Để xác định những tác động của ngẫu nhiên lên số lượng cá thể trùng lặp, chúng phân tích những thành phần của GP có kết hợp sử dụng yếu tố ngẫu nhiên gồm: (1) khởi tạo quần thể; (2) lựa chọn cặp cha - mẹ lai ghép, và quá trình lai ghép; (3) đột biến, tái tạo. Một cách tương ứng chúng tôi thống kê ba đại lượng: (1) số lượng cá thể trùng lặp do khởi tạo quần thể ngẫu nhiên; (2) số cá thể bị trùng lặp sau khi lai ghép, tái tạo tại mỗi thế hệ; (3) số lượng cá thể bị trùng lặp do các nguyên nhân khác. Tất cả mỗi đại lượng thống kê được lấy trung bình qua 31 lần chạy.

2.5. Giải quyết vấn đề trùng lặp cá thể trong quần thể tại mỗi thế hệ

Trong bài báo này chúng tôi chỉ đơn giản giải quyết vấn đề trùng lặp cá thể bằng cách hạn chế tối đa việc trùng lặp diễn ra do tác động của yếu tố ngẫu nhiên trong quá trình GP học, cụ thể như sau:

Khởi tạo quần thể ngẫu nhiên?

Để hạn chế vấn đề trùng lặp cá thể (chrome) do khởi tạo quần thể, khi sinh ra một cá thể thay vì chỉ sinh một lần, chúng tôi cố gắng sinh lặp lại nhiều lần. Cụ thể, nếu một cá thể sinh ra không trùng lặp (so khớp chrome) với các cá thể đã được tạo trước đó thì thêm nó vào quần thể như là cá thể mới; ngược lại thì sinh lại cá thể mới, công việc này được thực hiện với số lần lặp tối đa là MAXATEMPT lần.

Lựa chọn cặp cá thể để lai ghép, tái tạo?

Khi lựa chọn cặp cha - mẹ lai ghép, hai cá thể cha, mẹ không được phép trùng nhau. Nếu chọn trùng phải chọn lại, cố gắng thực hiện lặp lại MAXATEMPT lần.

- Khi lai ghép, nếu lai ghép không thành công (ví dụ tạo ra 2 con có size vượt quá kích cỡ cho phép) thì GP áp dụng tái tạo, tức sao chép cặp cha - mẹ vào thế hệ sau, điều này có thể dẫn đến sao chép lặp lại cha, mẹ nếu chúng được chọn ghép cặp lai ghép nhiều lần và dẫn đến trùng lặp. Để hạn chế trùng lặp, mỗi cha, mẹ được chọn tối đa 1 lần khi ghép cặp.

Lựa chọn điểm lai ghép trên cha - mẹ và thực hiện lai ghép?

Luật: một bộ gồm <ind_f, int_m, $crossPoint_1$, $crossPoint_2$>; trong đó: int_f, int_m là cha, mẹ và $crossPoint_1$, $crossPoint_2$ là hai điểm lai ghép trên cha, mẹ tương ứng. Bộ các giá trị này không cho phép chọn lặp lại trong quá trình tiến hóa quần thể để tránh trùng lặp.

2.6. Thiết lập thực nghiệm

2.6.1. Các bài toán thử nghiệm

Bảng 1 liệt kê các bài toán được sử dụng trong các thử nghiệm của nghiên cứu này. Trong đó 6 bài UCI là những tệp dữ liệu được sử dụng phổ biến trong các nghiên cứu về GP. Ngoài ra, chúng tôi thử nghiệm thêm với 2 tệp dữ liệu mưa được lấy từ 2 trạm: Tam Đảo (Rain_1) và Cửa Ông (Rain_2). Các tệp dữ liệu này đo lượng mưa tích lũy tại 2 thời điểm 0h và 12h trong 1 ngày. Ban đầu dữ liệu ở dạng chuỗi thời gian, sau đó được chúng tôi biến đổi thành dữ liệu thời điểm phục vụ cho việc học máy. Mục đích của việc học ở đây là dự đoán lượng mưa tại thời điểm xác định *(t)*, với giả thiết lượng mưa tại thời điểm *(t)* phụ thuộc vào lượng mưa của 10 ngày trước đó. Như vậy, 19 thuộc tính đầu tiên của mỗi tệp dữ liệu là lượng mưa tại 2 thời điểm 0h và

12h của 10 ngày, thuộc tính còn lại được xem là biến phụ thuộc (hay lượng mưa cần dự đoán) và nó phụ thuộc vào 19 giá trị biến trước đó. Dữ liệu của tập huấn luyện được lấy từ năm 2014 đến năm 2018, dữ liệu kiểm thử là dữ liệu năm 2019. Việc dự báo lượng mưa tại thời điểm xác định rất quan trọng trong việc hỗ trợ dự báo và cảnh báo một số hiện tượng khí tượng thủy văn nguy hiểm trong bối cảnh biến đổi khí hậu tại Việt Nam.

Bảng 1. *Các bài toán thử nghiệm*

ID	Bài toán	Kích cỡ tập huấn luyện	Kích cỡ tệp kiểm tra	Số lượng biến
UCI_1	Abalone	332	167	8
UCI_2	Housing	336	170	13
UCI_3	Bupa	100	245	6
UCI_4	Census6	100	300	6
UCI_5	No2	100	400	7
UCI_6	Ozone	140	63	12
Rain_1	48_52	3633	730	19
Rain_2	48836	3633	730	19

2.6.2. Cấu hình hệ thống

Bảng 2 chỉ ra các thiết lập tham số tiến hóa của GP được chúng tôi sử dụng trong các thực nghiệm nghiên cứu của bài báo này.

Bảng 2. *Các tham số tiến hóa*

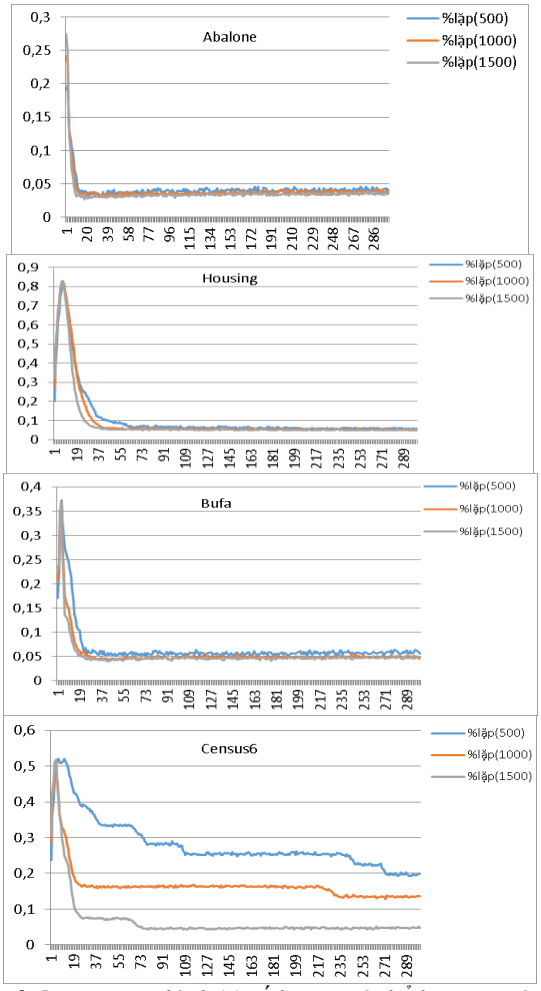
Tham số	GP
EA	Elitist, generational, tree expression
Function set	+, -, *, / (PD)
Terminal set	Regression variables; one random constant ∈ [0, 1]
#Generations	300
Population size	500
Tour size	4
Tree creation	Ramped half-and-half (depths of 2 to 6)
Max. tree depth	15
Crossover rate	0,9
Mutation rate	0,1
#Runs	31
Fitness function	RMSE
#Maxatempt	20

3. Kết quả và thảo luận

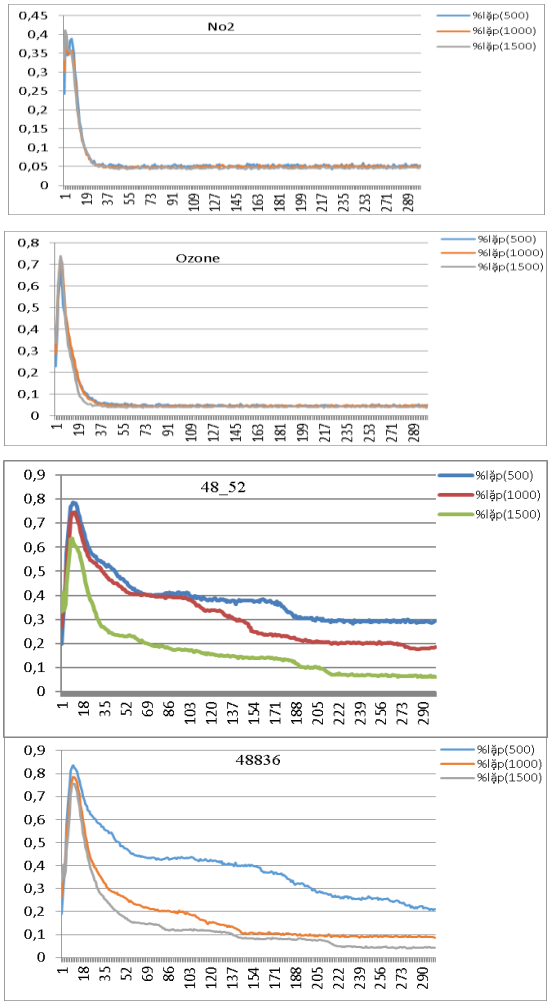
Ở mục này chúng tôi trình bày các kết quả thu được từ các thực nghiệm nhằm trả lời các câu hỏi nghiên cứu đã đặt ra.

3.1. Trùng lặp cá thể và sự tác động của kích cỡ quần thể lên vấn đề này

Sử dụng giải pháp đề xuất tại mục 2.3 hình 1 chỉ ra trung bình % tổng số cá thể bị trùng lặp qua các thế hệ (qua 31 lần chạy) trong quá trình tiến hóa khi kích cỡ quần thể lần lượt tăng là 500, 1000 và 1500. Với kết quả này chúng ta nhận thấy (1) hiện tượng trùng lặp cá thể là tồn tại trong các quần thể của GP; (2) kích cỡ quần thể không tác động nhiều đến số lượng các thể trùng lặp với đa phần các bài toán *(5/8 bài: Abalone, Housing, Bupa, No2, và Ozon)*, tuy nhiên với 3 bài còn lại (*Census6, 48_52và 48836)* thì số lượng cá thể trùng lặp có xu hướng giảm khi kích cỡ quần thể tăng.



Hình 1. *Trung bình % số lượng cá thể lặp qua các thế hệ tương ứng với các bài toán UCI khi kích cỡ quần thể tăng lần lượt là 500, 1000, và 1500*



Hình 1. *(Tiếp)*

3.2. Nguyên nhân gây trùng lặp cá thể

Chúng tôi tiến hành thực nghiệm để thống kê ra các cá thể có sự trùng lặp xảy ra khi khởi tạo quần thể ngẫu nhiên, xảy ra khi lai ghép, xảy ra do đột biến và do các nguyên nhân khác. Bảng 3 chỉ ra kết quả thống kê được thực hiện với kích cỡ quần thể là 500.

Bảng 3. *Trung bình số lượng cá thể có sự trùng lặp do khởi tạo, lai ghép và do nguyên nhân khác*

Bài toán	Trung bình số cá thể có trùng lặp		
	Khởi tạo	Lai ghép	Khác
UCI_1	36,68	10,18	10,87
UCI_2	25,90	10,24	43,50
UCI_3	39,94	10,11	23,39
UCI_4	39,94	9,47	130,75
UCI_5	39,00	10,29	22,51
UCI_6	28,39	10,25	25,80
Rain_1	18,45	9,39	182,72
Rain_2	18,45	9,47	183,35

3.3. Đánh giá giải pháp đề xuất

Trong nghiên cứu này chúng tôi chỉ đơn giản giải quyết vấn đề trùng lặp bằng cách cố gắng hạn chế một cách tối đa sự trùng lặp cá thể xảy ra do những tác động của ngẫu nhiên. Với giải pháp được đề xuất này kết quả ta đã khắc phục được hoàn toàn vấn đề trùng lặp xảy ra trong quần thể khởi tạo. Trùng lặp xảy ra do lựa chọn, lai ghép, tái tạo đã giảm đi một cách đáng kể với đa phần các bài toán thử nghiệm như chỉ ra trong bảng 4.

Bảng 4. *Trung bình số lượng cá thể có sự trùng lặp trước khi khắc phục và sau khi khắc phục trùng lặp do lai ghép*

Bài toán	Trung bình số cá thể có trùng lặp	
	Trước	Sau
UCI_1	10,18	2,38
UCI_2	10,24	2,39
UCI_3	10,11	2,34
UCI_4	9,47	2,38
UCI_5	10,30	2,38
UCI_6	10,25	2,36
Rain_1	9,39	6,73
Rain_2	9,47	6,89

Khi số lượng cá thể có sự trùng lặp giảm, nó góp phần đáng kể trong việc nâng cao khả năng khái quát hóa của GP trên hầu hết các bài toán thử nghiệm. Bảng 5 trình bày giá trị *p values* đạt được khi so sánh trung bình lỗi trên tập test của giải pháp tốt nhất mà GP học được trước khi giải quyết vấn đề trùng lặp và sau khi giải quyết vấn đề trùng lặp. Từ kết quả này, chúng ta thấy 5/8 các bài toán (UCI_1, UCI_3, UCI_4, UCI_5, UCI_6), p values có sự khác biệt đáng kể và khả năng khái quát hóa của GP được cải thiện một cách đáng kể sau khi khắc phục vấn đề trùng lặp. Một điểm thú vị ở đây là chỉ cần hạn chế sự trùng lặp gây ra do lai ghép mà khả năng khái quát hóa của GP đã tăng lên một cách rõ rệt với các bài toán này. Tuy nhiên, với 3/8 bài còn lại (UCI_1, Rain_1 và Rain_2) thì không có sự khác biệt đáng kể về khả năng khái quát hóa của GP sau khi giải quyết vấn đề trùng lặp so với trước khi giải quyết. Một trong các nguyên nhân được dự đoán ở đây là do số

lượng cá thể trùng lặp cá thể với các bài toán này là lớn, giải pháp mà chúng tôi đưa ra mới chỉ khắc phục được một phần nhỏ số lượng cá thể trùng lặp với các bài toán này.

Bảng 5. *p-values, trung bình lỗi trên tập test của giải pháp tốt nhất học bởi GP trước so với sau khi khắc phục trùng lặp, sử dụng kiểm thử thống kê Mann-Whitney U-test với độ tin cậy 95%. Dòng đậm chỉ ra sự khác biệt đáng kể. (1) lỗi test trước so với sau khi khắc phục trùng lặp do lai ghép; (2) trước so với sau khi khắc phục trùng lặp do khởi tạo quần thể và do lai ghép; (3) trước khi khắc phục trùng lặp; (4) sau khi khắc phục trùng lặp do lai ghép; (5) sau khi khắc phục trùng lặp do khởi tạo quần thể và do lai ghép*

Bài toán	p value		Fittest (median)		
	(1)	*(2)*	*(3)*	*(4)*	*(5)*
UCI_1	**0,00**	**0,02**	63,58	4,54	**4,18**
UCI_2	0,48	0,29	6,26	6,99	6,99
UCI_3	**0,00**	**0,00**	1,36	**0,50**	0,50
UCI_4	**0,00**	**0,00**	1,21	**0,20**	0,20
UCI_5	**0,00**	**0,00**	1,98	**0,70**	0,71
UCI_6	**0,00**	**0,00**	111,2	**73,65**	75,8
Rain_1	0,26	0,86	13,07	13,06	13,1
Rain_2	0,97	0,19	10,03	10,10	10,3

4. Kết luận

Trong nghiên cứu này chúng tôi đã xác định được nguyên nhân chính gây trùng lặp, từ đó đã đề xuất một số giải pháp ban đầu để khắc phục vấn đề này. Kết quả đã cải tiến một cách đáng kể khả năng khái quát hóa GP.

Nếu gọi n là kích thước quần thể; $O(f(n))$, $O(g(n))$ lần lượt là độ phức tạp thời gian, không gian của GP thì sau khi khắc phục vấn đề trùng lặp cá thể độ phức tạp về không gian và thời gian của giải pháp đề xuất là không đổi khi $n \to \infty$. Cụ thể, gọi m là số lần cá thể trùng lặp trong quá trình tiến hóa, *maxatempt* là số lần cố gắng thực hiện lại khi phát hiện trùng lặp, thì $m < n$ và MAXATEMPT *là hằng số, đồng thời rất nhỏ* so với n, do vậy m* MAXATEMPT < n khi $n \to \infty$. Từ đó, ta có $O(f(n+m*\text{MAXATEMPT})) = O(f(n))$ khi $n \to \infty$. Tương tự, nếu $g(n)$ là hàm xác định số ô nhớ lưu trữ quần thể của GP thì trong quá trình kiểm tra và khắc phục trùng lặp chúng tôi chỉ sử dụng thêm một danh sách có kích cỡ bằng n để chứa tần suất các cá thể được lựa chọn lặp lại, do vậy độ phức tạp không gian của giải pháp đề xuất là $O(g(2n)) = O(g(n))$ khi $n \to \infty$.

Giải quyết được vấn đề trùng lặp, đồng thời vẫn đảm bảo tính đa dạng trong quần thể GP là một thách thức. Loại bỏ yếu tố ngẫu nhiên, thay vào đó là một quy luật có trật tự tối ưu phản ánh đúng thế giới tự nhiên, đồng thời vẫn đảm bảo tính đa dạng của quần thể lại là một thách thức lớn hơn nhiều đặt ra đối với các nghiên cứu tương lai của GP nói riêng và các cách tiếp cận dựa trên tính toán tiến hóa nói chung.

Lời cám ơn

Nghiên cứu này được hỗ trợ bởi đề tài "Nghiên cứu cơ sở khoa học và giải pháp ứng dụng trí tuệ nhân tạo để nhận dạng, hỗ trợ dự báo và cảnh báo một số hiện tượng khí tượng thủy văn nguy hiểm trong bối cảnh biến đổi khí hậu tại Việt Nam", mã số BĐKH.34/16-20."

TÀI LIỆU THAM KHẢO/ REFERENCES

[1]. A. Rutherford, *A Brief History of Everyone Who Ever Lived: The Human Story Retold Through Our Genes*, The Experiment, 2018.

[2]. R. John, and Koza, *Genetic programming: on the programming of computers by means of natural selection*, MIT press, 1992.

[3]. Poli, Riccardo, Langdon, B. William, McPhee, F. Nicholas, Koza, and R. John, *A field guide to genetic programming*, Lulu. com, 2008.

[4]. Keijzer, and Maarten, "Alternatives in subtree caching for genetic programming," in *European Conference on Genetic Programming*, Springer, 2004.

[5]. Wong, Phillip, Zhang, and Mengjie, "SCHEME: Caching subtrees in genetic programming," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, 2008.

[6]. W. B. Langdon, B. Y. H. Lam, J. Petke, and M. Harman, "Improving CUDA DNA analysis software with genetic programming," *in Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, 2015.

[7]. E. Hemberg, L. Ho, M. O'Neill, and H. Claussen, "A symbolic regression approach to manage femtocell coverage using

grammatical genetic programming," in *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, 2011.

[8]. M. Nicolau, and M. Fenton, "Managing repetition in grammar-based genetic programming," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, 2016.

[9]. D. Yagyasen, M. Darbari, P. K. Shukla, and V. Kumar, "Diversity and convergence issues in evolutionary multiobjective optimization: application to agriculture science," *IERI Procedia*, vol. 5, pp. 81-86, 2013.

[10]. M. M. OUVÊA JR, and A. F. R.ARAÚJO, "Diversity - based adaptive evolutionary algorithms," *New Achievements in Evolutionary Computation*, pp. 318-334, 2010.