Tạp chí Tin học và Điều khiển học, T.27, S.4 (2011), 317–328

NGHIÊN CỨU VÀ THỬ NGHIỆM LẬP TRÌNH GEN TRONG BÀI TOÁN TÌM CÁC XẤP XỈ HÀM Q-FUNCTION*

ĐÀO NGỌC PHONG[1], NGUYỄN XUÂN HOÀI[2], NGUYỄN THANH THỦY[3], NGUYỄN QUANG UY[4]

[1]*Sở Thông tin Truyền thông thành phố Hà Nội*
[2]*Viện nghiên cứu và phát triển CNTT - Đại học Hà Nội*
[3]*Đại học Công nghệ - Đại học Quốc gia Hà Nội*
[4]*Học viện Kỹ thuật quân sự*

Tóm tắt. Bài báo đề xuất nghiên cứu việc sử dụng hai hàm thích nghi khác nhau, MAE và RAE, áp dụng hệ lập trình GEN định hướng bởi văn phạm nối cây (TAG3P) trong việc giải quyết bài toán xấp xỉ hàm Gaussian Q-function. Kết quả đã chỉ ra rằng việc sử dụng các hàm thích nghi khác nhau sẽ cho các kết quả khác nhau về chất lượng của các hàm xấp xỉ. Trên cơ sở đó, có thể sử dụng hàm thích nghi phù hợp với đặc thù bài toán xấp xỉ hàm Gaussian Q-function để tiến hành tìm các xấp xỉ theo hai dạng: dạng hàm tự do và dạng hàm mũ. Lời giải tìm ra bởi TAG3P tốt hơn tất cả các hàm xấp xỉ đã được đề xuất trước đây bởi các chuyên gia (các nhà toán học) và hơn thế nữa nó lại có cấu trúc đơn giản. Kết quả này sẽ được ứng dụng vào thực tiễn, là cơ sở cho việc tiếp tục nghiên cứu và ứng dụng TAG3P (và GP) trong giải quyết bài toán xấp xỉ hàm Q-function.

Abstract. In this paper, we investigate the use of two different fitness functions, MAE and RAE, for tree adjoining grammar guided genetic programming (TAG3P) with local search in solving the problem of Gaussian Q-function approximation. The results show that these different fitness functions have different effects on the quality of approximations. Based on the appropriate fitness function, we have discovered good Q-function approximations both in free-form and exponential form. The results found by TAG3P are better than all of the human expert designed approximations in the literature. This encourages further studies into the application of TAG3P (and GP) in solving the problem of Q-function approximation and applying it in the field of communications.

1. MỞ ĐẦU

Trong lĩnh vực viễn thông, hàm Gaussian Q-function (sau đây sẽ gọi tắt là hàm Q-function) có ý nghĩa rất quan trọng trong nhiều bài toán. Tuy nhiên, hàm Q-function chỉ duy nhất được định nghĩa dưới dạng tích phân suy rộng, điều này sẽ gây nhiều khó khăn cho những phân tích sau này đối với với các hệ thống viễn thông [1-6]. Do đó, rất cần thiết phải tìm ra các biểu diễn tường minh (dạng giải tích) của hàm Q-function.

*Bài báo được thực hiện với sự hỗ trợ từ quỹ phát triển KHCNVN (Nafosted), mã số 102.01-2011.08

Tuy nhiên, đến thời điểm này, chưa có giải pháp chính xác tuyệt đối cho dạng tường minh của hàm Q-function, và do đó, các hàm xấp xỉ là lựa chọn duy nhất [6]. Mặc dù có nhiều hàm xấp xỉ dạng tường minh của hàm Q-function đã được đề xuất bởi các chuyên gia (một số hàm xấp xỉ phổ biến sẽ được trình bày trong phần tiếp theo), nhưng do tính chất quan trọng và phổ biến của hàm Q-function trong lĩnh vực viễn thông, việc tìm kiếm các hàm xấp xỉ tốt hơn, có dạng giải tích và có nhiều đặc điểm quan trọng (chẳng hạn như sự đơn giản, tính khả tích,...) vẫn tiếp tục được thực hiện.

Lập trình gen (GP) có thể được xem như phương pháp học máy giải quyết các bài toán thông qua việc tìm ra các lời giải dưới dạng chương trình máy tính dựa trên quá trình chọn lọc tự nhiên [7-12]. Từ những ngày đầu, hồi quy ký hiệu (symbolic regression), thông qua việc tìm các xấp xỉ hàm dưới dạng ký hiệu, dạng tường minh và dạng giải tích là một trong các ứng dụng quan trọng của GP. Trong cuốn sách của mình, Koza đã chỉ ra tại sao GP có thể được sử dụng để học các hàm dưới dạng tường minh cho các bài toán hồi quy ký hiệu và tích phân [7]. Tuy nhiên, hiện tại chưa có bất kỳ nghiên cứu nào sử dụng GP trong việc tìm dạng xấp xỉ của hàm Q-function.

Hệ lập trình GEN định hướng bởi văn phạm nối cây (TAG3P)- sự mở rộng của GP, trong đó văn phạm nối cây được sử dụng để định hướng quá trình tiến hóa [9-11]. TAG3P được trình bày trong [10] tốt hơn GP trong một số bài toán hồi quy ký hiệu và tích phân. Các nghiên cứu gần đây đã chỉ ra rằng việc biểu diễn dựa trên TAG là tốt hơn các biểu diễn dựa trên văn phạm phi ngữ cảnh trong tìm kiếm tiến hóa mở rộng [8]. Điều này làm cho TAG3P là công cụ tiềm năng để giải quyết các bài toán hồi quy ký hiệu và tích phân như việc xấp xỉ hàm Q-function.

Kết quả nghiên cứu trong bài báo hướng tới 2 mục đích. Thử nghiệm những ảnh hưởng đến kết quả khi sử dụng các hàm thích nghi khác nhau và từ đó lựa chọn hàm thích nghi phù hợp với bài toán xấp xỉ hàm Q-function. Thứ 2 là ứng dụng lập trình gen định hướng bởi văn phạm nối cây (TAG3P) sử dụng tìm kiếm địa phương với hàm thích nghi phù hợp để giải quyết bài toán xấp xỉ hàm Q-function.

2. LÝ THUYẾT CƠ BẢN

Trong phần này, sẽ định nghĩa bài toán xấp xỉ hàm Q-function. Sau đó một số hàm xấp xỉ được đề xuất bởi chuyên gia sẽ được mô tả chi tiết. Cuối cùng là giới thiệu về TAG3P.

2.1. Bài toán xấp xỉ hàm Q-function

Hàm Q-function có vai trò rất quan trọng trong việc phân tích hiệu năng của một số bài toán thuộc lĩnh vực viễn thông [14]. Trong lĩnh vực viễn thông, các nhà nghiên cứu giả thuyết rằng hệ thống nhiễu có dạng biến Gaussian ngẫu nhiên. Việc tính toán xác suất lỗi trong hệ thống thông tin số thường gắn liền với việc tính toán dựa trên hàm lỗi $Q(x)$, được định nghĩa như sau:

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-y^2/2}\, dy. \tag{2.1}$$

Dạng hàm Q-function này có hai vấn đề. Trước tiên, xét về góc độ tính toán, nó đòi hỏi sự chặn trên của giới hạn vô cực khi sử dụng tích phân số hoặc kỹ thuật tính toán. Vấn đề thứ hai gặp phải là tham số của hàm là giá trị dưới của hàm tích phân, do đó, nó sẽ gặp khó

khăn về mặt giải tích khi tham số này phụ thuộc vào một biến số ngẫu nhiên đòi hỏi trung bình thống kê trên phân bố xác suất. Trong những trường hợp như vậy, ta cần có dạng hàm $Q(x)$ với tham số đầu vào không những không phải là giới hạn trên hay dưới của hàm tích phân, mà còn xuất hiện trong hàm tích phân như là tham số đầu vào của các hàm cơ bản. Dạng thức mong muốn vẫn là dạng mà giới hạn của tham số độc lập là hữu hạn, đồng thời, hàm tích phân vẫn duy trì cấu trúc tự nhiên của nó.

Các biểu diễn tường minh của hàm Q-function rất cần cho các bài toán liên quan đến hiệu năng của hệ thống viễn thông, đặc biệt là lỗi trung bình, bit và xác xuất khối lỗi và hàm mũ số nguyên với biến ngẫu nhiên thu được trong môi trường Fading. Mục đích cơ bản của việc xấp xỉ hàm Q-function là nhằm đưa ra dạng đơn giản và tường minh của hàm Q-function, tiện lợi trong việc phân tích toán học của hiệu năng hệ thống viễn thông [15].

2.2. Một số dạng xấp xỉ do chuyên gia đề xuất

Đến thời điểm này, có một số dạng xấp xỉ được đề xuất bởi các chuyên gia (các nhà toán học) trong lĩnh vực viễn thông về các xấp xỉ hàm Q-function dạng tường minh. Trong [1] đưa ra một tập các xấp xỉ của hàm lỗi bù, liên quan đến hàm Q-function. Xấp xỉ hàm Q-function trong [1, eqn(9)] được định nghĩa như sau:

$$Q(x) \approx \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}\sqrt{1+x^2}}. \tag{2.2}$$

Trong [2] sử dụng xấp xỉ PBCS với hàm xấp xỉ trong [2, eqn(13)]

$$Q(x) \approx \frac{1}{(1-a)x + a\sqrt{x^2+b}} \cdot \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}, \tag{2.3}$$

với a = 0.339, b = 5.510 sẽ được ký hiệu là xấp xỉ OPBCS.

Trong [3], một dạng xấp xỉ đơn giản và hữu ích khác được đưa ra. Tuy nhiên, những xấp xỉ này có sai số lớn với tham số đầu vào nhỏ, chính vì vậy nó ít phù hợp với kênh fading trên trung bình. Hàm xấp xỉ của Q-function trong [3] là:

$$Q(x) \approx \frac{1}{12}e^{-\frac{x^2}{2}} + \frac{1}{4}e^{-\frac{2x^2}{3}}, \tag{2.4}$$

trong bài báo này được ký hiệu là xấp xỉ CDS.

Trong [6], các tác giả đã đề xuất dạng thức đơn giản khác của xấp xỉ hàm Q-function với sai số nhỏ trên toàn bộ dải giá trị đầu vào. Để thuận tiện, hàm xấp xỉ Q-function được đưa ra trong [6] như sau:

$$Q(x) \approx \frac{(1-e^{-\frac{Ax}{\sqrt{2}}})e^{-\frac{x^2}{2}}}{\sqrt{2\pi}Bx}, \tag{2.5}$$

trong đó, giá trị tối ưu của biến số A và B là $A = 1,98$ và $B = 1,135$.

Tuy nhiên, những hạn chế của xấp xỉ dạng tự do này là nó có thể rất chính xác nhưng khó áp dụng trong thực tế bởi sự phức tạp của dạng hàm xấp xỉ. Do đó, những dạng xấp xỉ, mặc dù có độ chính xác thấp hơn nhưng có dạng đơn giản thì vẫn hữu ích trong thực tế. Benitez và Casadevall trong [7] đã đề xuất dạng xấp xỉ (được gọi là xấp xỉ dạng hàm mũ) với dạng tương

tự với hàm Gaussian, gọi là $e^{P(x)}$, trong đó $P(x)$ là đa thức bậc 2, với $P(x) = ax^2 + bx + c$. Với dạng xấp xỉ này, trong khoảng [0, 8], họ đã tìm được các giá trị tối ưu của tham số a, b và c tương ứng là -0,4698, -0,5026, và -0,8444. Dạng xấp xỉ này ít chính xác hơn so với OPBCS, nhưng khả năng dễ tính toán làm nó trở nên hữu ích trong bài toán phân tích hệ thống viễn thông.

2.3. Lập trình gen

Lập trình gen (GP) là một công nghệ tính toán tiến hóa tự động giải quyết bài toán, không đòi hỏi phải biết dạng thức hay cấu trúc của lời giải [8, 13]. GP có thể khái quát là phương thức có hệ thống và độc lập với bài toán để tự động giải quyết bài toán từ những định nghĩa cấp cao. GP dựa trên ý tưởng chính là các chương trình máy tính có khả năng tự tiến hóa để thực hiện một công việc nào đó, được đưa ra bởi Koza vào năm 1992. Kỹ thuật này cung cấp một nền tảng cho việc tạo ra những chương trình máy tính một cách tự động.

Quá trình tiến hóa của GP bắt đầu với việc tạo ra ngẫu nhiên quần thể gồm các cá thể (chương trình). Ở mỗi thế hệ, độ tốt của mỗi cá thể chương trình (hàm số) được đánh giá trên cơ sở mức độ tốt của chương trình (hàm số) trong việc giải quyết bài toán đặt ra. Do đó, các cá thể sẽ được lựa chọn thông qua các toán tử di truyền (lai ghép, đột biến) để tạo ra các cá thể con trong thế hệ tiếp theo. Quá trình này lặp đi lặp lại cho đến khi lời giải được tìm ra hoặc khi gặp một số điều kiện dừng nào đó [13]. Hệ lập trình GEN chuẩn sẽ gồm 5 thành phần: biểu diễn chương trình, khởi tạo quần thể, hàm thích nghi, toán tử di truyền và các tham số.

Như đã trình bày ở trên, GP được sử dụng để học cách xác định một hàm dựa trên dữ liệu mẫu. Hệ thống GP thường thể hiện dưới dạng chuỗi gen phức tạp với kích thước và hình dạng biến đổi. GP thành công trong giải quyết nhiều bài toán trong thực tế bao gồm các bài toán xấp xỉ tích phân ký hiệu ([8, 13]). Tuy nhiên, đến thời điểm này chưa có công bố nào sử dụng GP trong việc xấp xỉ hàm Q-function. Do đó, sẽ có giá trị nếu GP có thể giải quyết bài toán mà không cần những kiến thức hiểu biết chuyên sâu.

Có nhiều phương pháp học máy được sử dụng như: cây quyết định, mạng nơ ron nhân tạo, lập trình GEN, máy vectơ hỗ trợ – SVM,.... Các phương pháp được tiếp cận theo hai mô hình là hộp đen (Blackbox) và hộp trắng (Whitebox) hoặc lai ghép giữa hai mô hình trên. Mạng nơ-ron hay máy véctơ hỗ trợ là một ví dụ về mô hình hộp đen, do lời giải thích cho kết quả quá phức tạp để có thể hiểu được. Trong khi đó, lập trình GEN tiếp cận theo mô hình hộp trắng với lời giải ở dạng giải thích và dạng đóng. Chính vì thế, nhằm mục đích tìm các xấp xỉ ở dạng giải tích và tường minh của hàm Q-function, việc sử dụng GP là phù hợp.

2.4. Hệ lập trình GEN định hướng bởi văn phạm nối cây

Lập trình gen định hướng bởi văn phạm (GGGP) (Whigham, 1995) và lập trinh di truyền định hướng bởi văn phạm nối cây (TAG3P) (Nguyen et al., 2003; Nguyen, 2004; Nguyen et al., 2006) là các hệ mở rộng của Lập trình gen truyền thống dựa trên việc sử dụng văn phạm phi ngữ cảnh và văn phạm nối cây để định nghĩa ngôn ngữ, không gian biểu diễn các đối tượng được học (hàm, chương trình máy tính, hay các thiết kế).

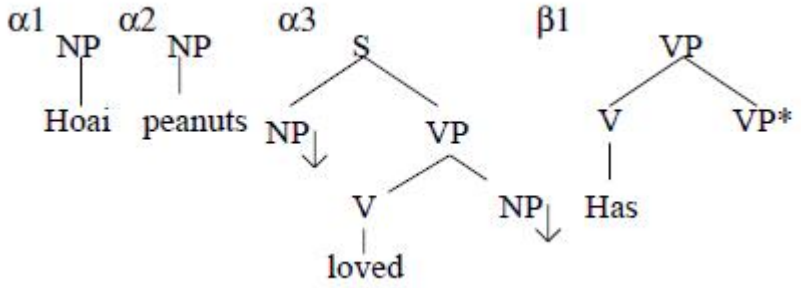
Văn phạm nối cây (Tree-Adjoining Grammars – TAGs)

Văn phạm nối cây đã và đang trở thành loại văn phạm quan trọng trong xử lý ngôn ngữ tự nhiên (Natural Language Processing – NLP). Mục đích của TAG là chỉ ra một cách trực

tiếp cách tạo cấu trúc của các ngôn ngữ tự nhiên so với các hệ viết lại trên xâu thuộc các lớp của Chomsky. Cấu trúc văn phạm của TAG được hình thành bởi hai tập hợp cây con, cây khởi tạo hay còn gọi là cây α, tương ứng với các thành phần cơ bản của ngôn ngữ và cây bổ trợ hay còn gọi là cây β tương ứng với các nhân tố có thể chèn thêm của ngôn ngữ. Các cây con này còn được gọi là các cây cơ bản. Cũng giống như đối với các văn phạm Chomsky, các nút của cây được gán bằng các ký hiệu kết thúc (terminal symbols) và không kết thúc (non-terminal symbols), trong đó các nút bên trong phải được gán bằng các ký hiệu không kết thúc, các nút lá có thể được gán bằng cả hai loại ký hiệu kết thúc hoặc không kết thúc.

Văn phạm TAG là một bộ năm thành phần $(\sum, N, I, A, S)$, trong đó:

- $\sum$: tập hữu hạn các kí hiệu kết thúc.
- N: tập hữu hạn các kí hiệu không kết thúc.
- S: tập phân biệt các kí hiệu không kết thúc.
- I: tập các cây khởi tạo. Trong cây khởi tạo, các nút bên trong phải được gán bằng các ký hiệu không kết thúc, các nút lá có thể được gán bằng cả hai loại ký hiệu kết thúc hoặc không kết thúc. Nút lá có kí hiệu không kết thúc có đánh dấu $\downarrow$ thể hiện khả năng thực hiện phép thế tại các nút đó.
- A: tập các cây bổ trợ. Trong cây bổ trợ, các nút trong được gán bằng các ký hiệu không kết thúc. Mỗi cây đều có chứa một nút lá trùng tên với nút gốc (mang kí hiệu không kết thúc). Ở nút lá này được đánh dấu bằng kí hiệu * và được gọi là nút chân của cây phụ trợ. Mỗi cây phụ trợ chỉ có một nút chân.



Hình 2.1. Một ví dụ về TAG

Các phép viết lại trên cây chính được sử dụng với các văn phạm nối cây là phép nối cây (adjunction) và phép thế cây (substitution). Phép nối cây tạo ra cây dẫn xuất mới γ từ cây bổ trợ β và cây τ (có thể là một cây khởi tạo hoặc là cây dẫn xuất được đã được tạo ra). Nếu cây τ có một nút trong được gán nhãn 'A' và cây β là cây dạng A, việc kết nối β và cây τ được thực hiện như sau: đầu tiên cây con σ bắt nguồn tại A tạm bị ngắt khỏi τ, sau đó β gắn với τ để thay thế cho cây con đó, cuối cùng σ được gắn trở lại nút chân của β. Cây γ là cây dẫn được cuối cùng được tạo ra trong quá trình này.

Trong phép thế cây, một nút không kết thúc X của cây khởi tạo hoặc cây dẫn xuất được được thay thế bởi một cây khởi tạo dạng X. Các cây dẫn xuất được hoàn chỉnh trong TAG (cây dẫn xuất có các nút lá đều được dãn nhãn bởi các ký tự kết thúc) tương ứng trực tiếp với những cây dẫn xuất được tạo ra trong lớp ngôn ngữ Chomsky.

Lập trình gen định hướng bởi văn phạm nối cây

TAGs có giá trị ứng dụng cao đối với xử lý ngôn ngữ tự nhiên, đặc biệt, một điểm mạnh trong cây dẫn xuất của TAG là khi xóa bất kỳ một cây con nào ra khỏi cây dẫn xuất, phần còn lại vẫn là một cây dẫn xuất TAG và là cây hợp lệ. Nói cách khác, cây dẫn xuất trong TAG là cây có thứ phân không cố định. Hệ quả của điều này là khả năng tạo ra các toán tử mới giống như trong các hệ tính toán phỏng tiến hóa sinh học dùng biểu diễn tuyến tính cho nhiễm sắc thể mà vẫn duy trì tính chất ưu việt của biểu diễn dạng hình cây.

Hệ lập trình GEN định hướng bởi văn phạm nối cây (TAG3P) là hệ thống sử dụng văn phạm nối cây để định hướng tiến hóa [10-12]. TAG3P sử dụng văn phạm nối cây cùng với văn phạm phi ngữ cảnh để đặt ra những ràng buộc về cú pháp cũng như những sai lệch khi tìm kiếm của chương trình tiến hóa. Hệ thống TAG3P có tất cả những thuộc tính của các hệ thống GP thông thường dựa trên các biểu diễn dạng hình cây khác.

Cấu trúc văn phạm được xác định bằng tập hợp các cây α và cây β. Cấu trúc quần thể là các cây dẫn xuất từ văn phạm này. Việc lượng giá độ tốt của mỗi cá thể được thực hiện bằng cách tạo ra các cây dẫn xuất được tương ứng từ cây dẫn xuất TAG, sau đó đánh giá biểu thức trên cây dẫn xuất được (tương tự như trong GP). Không gian tìm kiếm do đó được xác định bằng văn phạm, tập hợp tất cả các cây biểu thức GP đều do văn phạm cho trước tạo ra với giới hạn về độ phức tạp của cây này. Tuy nhiên, đặc tính thứ nguyên không xác định của cây giúp kiểm soát một cách dễ dàng theo kích thước của cây, do đó, kích thước của cây được sử dụng để kiểm soát độ phức tạp của cây trong TAG3P thay vì theo chiều cao của cây như trong các hệ GP khác.

Cũng giống như trong hệ lập trình GEN chuẩn, TAG3P gồm 5 thành phần là biểu diễn chương trình, khởi tạo quần thể, hàm thích nghi, toán tử di truyền và các tham số. Cụ thể như sau:

Biểu diễn chương trình

Trong TAG3P, cấu trúc văn phạm được xác định bằng tập hợp các cây α và cây β, mỗi cá thể trong một quần thể là cây dẫn xuất của LTAG, được dẫn xuất từ các văn phạm này.

Khởi tạo quần thể

Thành phần thứ hai trong TAG3P là thuật giải để khởi tạo ngẫu nhiên quần thể (các cây dẫn xuất Glex). Đây là quá trình lặp đi lặp lại việc tạo ra ngẫu nghiên một cá thể (một cây dẫn xuất Glex). Việc này được thực hiện bằng cách chọn một số ngẫu nhiên trong khoảng cho trước, sau đó lấy ngẫu nhiên câu α từ tập cây cơ sở trong Glex để tạo ra cây dẫn xuất Glex. Cây dẫn xuất này sẽ được mở rộng bằng phép nối với một cây β được chọn ngẫu nhiên từ tập cây cơ sở. Quá trình này kết thúc khi kích thước cây đạt tới giá trị được chọn nhẫu nhiên ở trên.

Hàm thích nghi

Để đánh giá sự phù hợp của cá thể, đầu tiên ta sẽ chuyển đổi thành cây dẫn xuất được trong Glex (cây dẫn xuất của G trong trường hợp sử dụng G). Sau đó, quá trình tính toán sự phù hợp của cá thể được thực hiện trên cây dẫn xuất được.

Toán tử di truyền TAG3P cũng có các toán tử di truyền chính như GP chuẩn là lựa chọn, tái tạo, lai ghép và đột biến:

Do tính thứ phân không cố định của biểu diễn TAG, nhiều toán tử mới được thiết kế. Việc sử dụng TAGs giúp cho TAG3P có được những toán tử hữu ích và phỏng tiến hoá sinh học hơn so với các toán tử của hệ lập trình GEN và hệ lập trình GEN định hướng bởi văn phạm khác [11]. Việc thiết kế các toán tử đó khó có thể thực hiện trên biểu diễn chuẩn của GP với cấu trúc cây cũng như đối với Hệ lập trình GEN định hướng bởi văn phạm sử dụng cấu trúc cây dẫn xuất trong CFG. Một số toán tử được thiết kế như: di chuyển (relocation), sao lưu (duplication), và đặc biệt là hai toán tử chèn (insertion) và xóa (deletion). Những đặc tính năng này mang lại cho TAG3P những ưu điểm trong việc tìm kiếm mô hình để giải quyết các bài toán hồi quy tượng trưng và tích phân. Bên cạnh đó, trong TAG3P còn thiết kế các toán tử đột biến từ vựng.

Trong bài báo này, ta sử dụng 2 toán tử chèn và xóa hoạt động như là toán tử tìm kiếm địa phương trên cơ sở kết hợp của việc tiến hóa và thuật toán tìm kiếm leo đồi.

Chèn: Nếu kích thước của cá thể còn nằm trong giới hạn trên cho phép thì thao tác chèn được thực hiện bằng cách thêm nút lá vào cá thể đó.

Xóa: Nếu kích thước của cá thể còn nằm trên giới hạn dưới thì thao tác xoá được thực hiện bằng cách xóa nút lá ra khỏi cá thể đó.

Các tham số

Các tham số trong TAG3P bao gồm: kích cỡ quần thể, số thế hệ tối đa, kích thước tối đa và tối thiểu của cây, số lần thực hiện tối đa (thực hiện các thao tác di truyền), xác xuất thực hiện các toán tử di truyền, sự sai khác tối đa về kích thước của cây con cũ và mới khi thực hiện thao tác di truyền (lai ghép, đột biến, ...).

3. XÁC ĐỊNH HÀM THÍCH NGHI

Như đã trình bày ở trên, một hệ thống GP bao gồm 5 thành phần là biểu diễn chương trình, khởi tạo quần thể, hàm thích nghi, toán tử di truyền và các tham số. Trong đó, hàm thích nghi là thành phần rất quan trọng để đánh giá độ tốt của cá thể, làm cơ sở cho việc thực hiện quá trình tiến hóa. Có nhiều dạng hàm thích nghi khác nhau được sử dụng để phù hợp với từng bài toán khác nhau.

Dạng hàm thích nghi hay được sử dụng với GP và TAG3P là hàm dựa trên lỗi tuyệt đối trung bình (Mean Absolute Error - MAE):

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|f_i - y_i|, \tag{3.6}$$

Trong đó, N là số lượng dữ liệu mẫu, f_i là giá trị của hàm Q-function và y_i là giá trị của hàm học được tại mẫu thứ i trong tập dữ liệu mẫu. Trong bài báo này, tập dữ liệu mẫu gồm 400 giá trị, chia đều trong khoảng [0-8] và được tạo ra tương ứng dựa vào công thức (2.1). Lý do của sự giới hạn này là bởi hàm Q-function sẽ tiệm cận giá trị 0 khi x lớn hơn 8, do đó ít có giá trị hữu dụng trong lĩnh vực viễn thông.

Tuy nhiên, mặc dù hàm thích nghi này chỉ ra được tính xấp xỉ với hàm Q-function dưới góc độ sai số tuyệt đối, nhưng nó không thể hiện được tính tốt của hàm xấp xỉ. Đặc biệt, với đặc điểm là hàm Q-function sẽ tiệm cận càng gần giá trị 0 khi x càng lớn. Chính vì vậy, nếu sử dụng hàm thích nghi dựa trên lỗi tuyệt đối trung bình thì sẽ không thể hiện được bản chất

của hàm xấp xỉ tìm được (do lỗi tuyệt đối có thể là nhỏ nhưng lại lớn hơn nhiều so với giá trị của hàm). Do đó, ở đây ta thử nghiệm thêm TAG3P với hàm thích nghi khác dựa trên hàm lỗi tương đối (Relative Absolute Error - RAE) như sau:

$$\text{RAE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|f_i - y_i|}{f_i}, \tag{3.7}$$

trong đó, N, f_i, y_i được định nghĩa tương tự như trên.

Thiết lập 2 tập thí nghiệm với TAG3P sử dụng 2 hàm thích nghi ở trên, mỗi tập thí nghiệm gồm 50 lần chạy. Khi thiết lập thí nghiệm, ta sử dụng tập hàm gồm các hàm (+,-,*,/,exp,log,sqrt). Việc lựa chọn thêm các hàm exp, log, sqrt là dựa trên cơ sở tham khảo các dạng xấp xỉ do chuyên gia đề xuất như đã trình bày ở trên. Tập kết được sử dụng gồm có tham số x, Π và các hằng số ngẫu nhiên được sinh ra trong khoảng (0,1).

Với các kết quả thu được sau khi chạy 2 tập thí nghiệm, ta lựa chọn một số hàm xấp xỉ tốt nhất tìm được và vẽ đồ thị lỗi tương đối của các hàm này. Qua đó, có nhận xét như sau: khi sử dụng MAE là hàm thích nghi thì trong khoảng [0-e], giá trị hàm xấp xỉ được tìm là tốt. Tuy nhiên, khi x lớn hơn e, chất lượng của các xấp xỉ xấu đi đáng kể và khá nhanh. Điều này có thể lý giải bởi đặc điểm của hàm Q-function là giá trị của hàm Q-function tiến về giá trị 0 rất nhanh khi x lớn hơn e (nhỏ hơn 10^{-8}). Do đó, việc sử dụng MAE là hàm thích nghi trong đoạn [e-8] có thể dẫn đến những sai lệch. Nói cách khác, do giá trị của hàm Q-function ở những điểm này là rất nhỏ, việc có giá trị tuyệt đối nhỏ không đảm bảo cho việc sẽ có lỗi tương đối nhỏ.

Trong khi đó, với các kết quả thu được khi sử dụng RAE là hàm thích nghi thì chúng tôi nhận thấy các xấp xỉ này có lỗi tương đối nhỏ đồng đều hơn so với các xấp xỉ được tìm ra khi sử dụng hàm thích nghi MAE. Như vậy để có được xấp xỉ tốt đều trên đoạn [0-8], RAE là lựa chọn phù hợp hơn.

Qua việc so sánh kết quả xấp xỉ nhận được khi sử dụng hai hàm thích nghi là MAE và RAE, ta thấy rằng với đặc điểm của hàm Q-function thì hàm thích nghi phù hợp được sử dụng sẽ là RAE.

4. THIẾT KẾ THÍ NGHIỆM

Trên cơ sở kết quả lựa chọn hàm thích nghi ở trên, ta tiến hành thiết kế thí nghiệm để tìm các xấp xỉ theo hai dạng: dạng hàm tự do và dạng hàm mũ.

Với thí nghiệm tìm xấp xỉ dạng hàm tự do, sử dụng tập hàm gồm các hàm (+, -, *, /, exp, log, sqrt). Còn trong trường hợp tìm xấp xỉ dạng hàm mũ ta sử dụng các tập hàm gồm +, -,*.

Tập kết được sử dụng gồm có tham số x, Π và các hằng số ngẫu nhiên được sinh ra trong khoảng (0,1).Phương pháp chọn được sử dụng ở đây là lựa chọn cạnh tranh với kích cỡ là 9.

Để đánh giá hiệu quả của các toán tử tìm kiếm địa phương trong giải quyết bài toán xấp xỉ hàm Q-function, như cấu hình thí nghiệm nêu trên, ta tiến hành chạy với 2 trường hợp: không sử dụng tìm kiếm địa phương (TAG3P) và có sử dụng tìm kiếm địa phương (TAG3PL).

Bảng 1. Bảng thông số được cài đặt cho mỗi thí nghiệm

Tham số	**Giá trị**
Số thế hệ	100
Kích thước quần thể	3000 với TAG3P và 100 với TAG3PL
Phương pháp chọn	Lựa chọn cạnh tranh với kích cỡ là 9
Xác xuất các toán tử	Lai ghép = 0.9, Đột biến = 0.1
Xác xuất các toán tử	Lai ghép = 0.9, Đột biến = 0.1
Hàm thích nghi	Lỗi tương đối trung bình (RAE)
Toán tử tìm kiếm địa phương	Ngẫu nhiên lựa chọn toán tử chèn và toán tử xóa
Chiến lược tìm kiếm địa phương	Leo đồi
Bước tìm kiếm địa phương	0 với TAG3P và 30 với TAG3PL
Số lần chạy thí nghiệm	50

5. KẾT QUẢ VÀ NHẬN XÉT

Để xác định các xấp xỉ tốt của hàm Q-function, với mỗi lần chạy, đều lưu trữ lại những cá thể có giá trị RAE tương đương hoặc tốt hơn giá trị RAE của các xấp xỉ đưa ra bởi chuyên gia. Trong đó, giá trị RAE của các xấp xỉ đưa ra bởi chuyên gia trong tập dữ liệu huấn luyện là: CDS (0,2437469), GKAL (0,0614184), PBCS (0,0346417), OPBCS (0,0017471) và EXP(0,0348177). Dựa trên giá trị này, có thể ghi lại những xấp xỉ tìm được có giá trị RAE nhỏ hơn 10^{-2} (tức là tốt hơn các xấp xỉ đưa ra bởi chuyên gia, ngoại trừ xấp xỉ OPBCS) và những xấp xỉ tìm được có giá trị RAE nhỏ hơn 10^{-3} (tức là tốt hơn tất cả các xấp xỉ đưa ra bởi chuyên gia).

Trong thí nghiệm tìm xấp xỉ dạng hàm tự do, khi chạy 50 lần với trường hợp TAG3P (không sử dụng tìm kiếm địa phương) thì kết quả thu được không có xấp xỉ nào có RAE nhỏ hơn 10^{-2}. Còn trong số 50 lần chạy với trường hợp TAG3PL (có sử dụng tìm kiếm địa phương) thì có 20 lần chạy cho kết quả RAE nhỏ hơn 10^{-2} (trong đó có 1 trường hợp có RAE là 0,001 tương đương với RAE của OPBCS là 0,0017), nhưng không thu được xấp xỉ nào có RAE nhỏ hơn 10^{-3}. Xấp xỉ tốt nhất tìm được là:

$$TAG\_FREE = \frac{f(x)}{g_1(x) + g_2(x)} + h(x), \tag{5.8}$$

trong đó

$$f(x) = 0,512288x - 3,152892, \tag{5.9}$$

$$g_1(x) = \frac{1,351156}{0,788367x + \frac{0,088717}{x^3 e^{(x+3,292595)(1,848251x-0,045731)} + 5,141592x^3 - x^2 + 2,141592x + 0,455729}} \tag{5.10}$$

$$g_2(x) = 0,003616x^3 + 0,087906x^2 - 0,538290x + 0,096699, \tag{5.11}$$

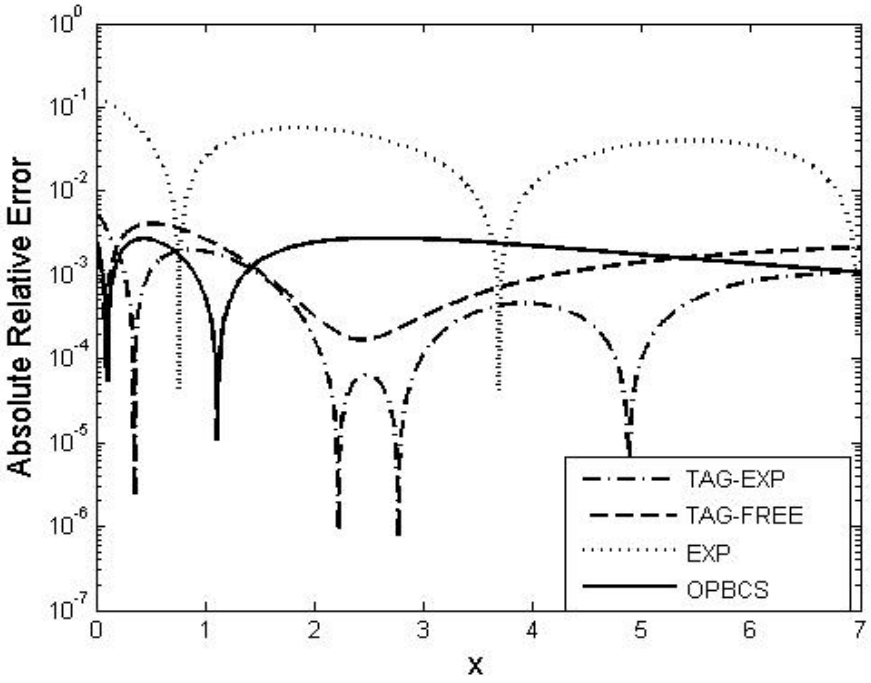
$$h(x) = -0,477759x^2 - 0,512288x + 0,450333. \tag{5.12}$$

Còn trong thí nghiệm tìm xấp xỉ dạng hàm mũ, khi chạy 50 lần với trường hợp TAG3P thì kết quả thu được duy nhất một xấp xỉ có RAE nhỏ hơn 10^{-2} và không có xấp xỉ nào có RAE nhỏ hơn 10^{-3}. Đặc biệt, trong số 50 lần chạy với TAG3PL thì có 30 lần chạy cho kết quả RAE nhỏ hơn 10^{-2} và có 4 lần chạy cho xấp xỉ với RAE nhỏ hơn 10^{-3}.

Như vậy có thể thấy, nếu áp dụng tìm kiếm địa phương thì đều tìm được (40-60% số lần chạy) các kết quả tốt bằng hoặc hơn các xấp xỉ đã đề xuất bởi chuyên gia. Đặc biệt, trong trường hợp xấp xỉ dạng hàm mũ với tìm kiếm địa phương thì thu được 4 lần chạy cho xấp xỉ với RAE nhỏ hơn 10^{-3}, trong đó, xấp xỉ tốt nhất tìm được có dạng như sau:

$$TAG\_EXP = e^{ax^6+bx^5+cx^4+dx^3+ex^2+fx+g}, \tag{5.13}$$

trong đó, các giá trị hằng số như sau: a=0,0000018643; b=-0,000109; c=0,002238; d = -0,023735; e = -0,344644; f = -0,774128 và g = -0,698740.



Hình 5.2. Lỗi tương đối của các xấp xỉ Q-function

Hình vẽ trên là đồ thị hàm lỗi tương đối của xấp xỉ vừa tìm được bởi TAG3PL với dạng tự do (gọi tắt là xấp xỉ TAG-FREE) và dạng hàm mũ (gọi tắt là xấp xỉ TAG-EXP) so với xấp xỉ tốt nhất đưa ra bởi chuyên gia (xấp xỉ OPBCS) và xấp xỉ dạng hàm mũ đưa ra bởi chuyên gia (xấp xỉ EXP). Có thể thấy độ chính xác của xấp xỉ TAG-FREE là tương đương với xấp xỉ OPBCS (có khoảng TAG-FREE tốt hơn nhưng có khoảng thì ngược lại), tuy nhiên, dạng thức của xấp xỉ này thì lại rất phức tạp.

Trong khi đó, có thể thấy độ chính xác của xấp xỉ TAG-EXP là tương đương với OPBCS trong khoảng [0-1,5] và khi $x > 1{,}5$ thì nó tốt hơn OPBCS. Hơn thế nữa, xấp xỉ này lại có

dạng hàm mũ nên rất đơn giản trong quá trình tính toán, phù hợp với đặc điểm sử dụng trong phân tích hệ thống viễn thông. Kết quả này rất quan trọng vì kể từ khi xấp xỉ OPBCS được đưa ra năm 1979 đến nay chưa có một xấp xỉ nào tốt hơn được tìm ra. Ngoài ra, cùng có dạng hàm mũ nhưng so với xấp xỉ EXP thì xấp xỉ TAG-EXP có độ chính xác cao hơn rất nhiều.

6. KẾT LUẬN

Trong bài báo này, chúng tôi thực hiện thử nghiệm ban đầu bằng việc sử dụng các hàm thích nghi khác nhau (MAE và RAE) qua đó xác định được hàm thích nghi RAE là phù hợp nhất với bài toán xấp xỉ hàm Gaussian Q-function. Trên cơ sở đó, ta thiết kế và chạy 2 tập thí nghiệm sử dụng TAG3P và TAG3PL tìm các xấp xỉ dạng hàm tự do và dạng hàm mũ. Kết quả thu được những xấp xỉ tốt hơn so với xấp xỉ được đưa ra bởi chuyên gia, hơn thế nữa nó lại có cấu trúc đơn giản, phù hợp với ứng dụng trong hệ thống viễn thông.

Việc tìm thêm các xấp xỉ tốt hơn nữa khi sử dụng TAG3PL với dạng hàm tự do và dạng hàm mũ, cùng với nghiên cứu sử dụng phương pháp kiểm soát tràn mã như phương pháp Tarpeian Bloat Control và ứng dụng các xấp xỉ tìm được vào bài toán phân tích hệ thống viễn thông để tạo ra các mô phỏng chính xác cho các hệ thống đặc biệt sẽ là nội dung nghiên cứu các công trình tiếp theo.

TÀI LIỆU THAM KHẢO

[1] P.O. Borjesson and C.E. Sundberg, Simple Approximations of the Error Function $Q(x)$ for Communications Applications, *IEEE Transactions on Communications* **27** (1979) 639–643.

[2] Y. Chen and N.C. Beaulieu, A Simple Polynomial Approximation to the Gaussian Q-function and Its Application, *IEEE Communications Letters* **12** (2) (2009) 124–126.

[3] M. Chiani, D. Dardari and M. K. Simon, New Exponential Bounds and Approximations for the Computation of Error Probability in Fading Channels, *IEEE Transactions on Wireless Communications* **2** (2003) 840–845.

[4] A. Joshi, L. Levy and M. Takahashi, Tree adjunct grammars, *Journal of Computer System Science* **21** (2) (1975) 136-–163.

[5] A.K. Joshi and Y. Schabes, Tree adjoining grammars, in Rozenberg, G. and Saloma A. (ed), *Handbook of Formal Languages*, Springer-Verlag, 1997 (69–123).

[6] G. K. Karagiannidis and A. S. Lioumpas, An improved approximation for the Gaussian Q-function, *IEEE Communication Letters* **11** (2007) 644–646.

[7] M. Benitez and F. Casadevall, Versatile, accurate, and analytically tractable approximation for the gaussian Q-function, *IEEE Transactions on Communications* **59** (4) (2011) 917-–922.

[8] J. Koza, *Genetic programming: on the programming of computers by means of natural selection*, MIT Press, Cambridge, MA, USA, 1992.

[9] E. Murphy, M. O'Neill, and A. Brabazon, Examining mutation landscapes in grammar based genetic programming, *Proceedings of European Conference on Genetic Programming, Lecture Note on Computer Science (LNCS)* **6621** (2011) 130–141.

[10] Nguyen Xuan Hoai, "A flexible representation for genetic programming: lessons from natural language processing", PhD Thesis, University of New South Wales, 2004.

[11] Nguyen Xuan Hoai, R.I. McKay, and D. Essam, Representation and Structural Difficulty in Genetic Programming, *IEEE Transactions on Evolutionary Computation* **10** (2)(2006) 157–166.

[12] Nguyen Xuan Hoai, R.I. McKay, and H.A. Abbass, Tree adjoining grammars, language bias, and genetic programming, *Proceedings of European Conference on Genetic Programming, Lecture Notes in Computer Science 2610*, Springer, 2003 (335–344).

[13] R. Poli, W. Langdon, and N. McPhee, *A Field Guide to Genetic Programming*, Lulu Enterprise, 2008.

[14] M.K. Simon, Probability distributions involving gaussian random variables, *A Handbook for Engineers and Scientists*, Kluwer Academics, 2002.

[15] M.K. Simon and M.S. Alouini, *Digital Communications Over Fading Channels: A Unified Approach to Performance Analysis*, Wiley Sons, 2000.

Nhận bài ngày 10 - 11 - 2011
Nhận lại sau sửa ngày 21 - 3 - 2012