ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

TRẦN HOÀNG VIỆT

KIỂM CHỨNG DỰA TRÊN PHƯƠNG PHÁP GIẢ ĐỊNH – ĐẢM BẢO CHO PHẦN MỀM DỰA TRÊN THÀNH PHẦN

TÓM TẮT LUẬN ÁN TIẾN SĨ NGÀNH CÔNG NGHỆ THÔNG TIN

Hà Nội - 2020

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

– – – – – – – – – – – – –

TRẦN HOÀNG VIỆT

KIỂM CHỨNG DỰA TRÊN PHƯƠNG PHÁP GIẢ ĐỊNH – ĐẢM BẢO CHO PHẦN MỀM DỰA TRÊN THÀNH PHẦN

Chuyên ngành: Kỹ Thuật Phần Mềm

Mã số: 9480103.01

TÓM TẮT LUẬN ÁN TIẾN SĨ NGÀNH CÔNG NGHỆ THÔNG TIN

NGƯỜI HƯỚNG DẪN KHOA HỌC:

1. PGS. TS. Phạm Ngọc Hùng
2. TS. Võ Đình Hiếu

Hà Nội - 2020

Mục lục

Chương 1

Giới thiệu

1.1 Đặt vấn đề

Phương pháp kiểm chứng giả định - đảm bảo cho hệ thống chuyển trạng thái được gán nhãn (Labelled Transition System - LTS) là phương pháp phổ biến nhất trong cộng đồng nghiên cứu cũng như trong việc áp dụng trong công nghiệp. Trong phương pháp này, từ các luật kiểm chứng giả định - đảm bảo, bản chất của bài toán kiểm chứng là bài toán bao của các ngôn ngữ ($M \models p \equiv L(M)\restriction_{\Sigma_p} \subseteq L(p)$) và độ phức tạp của quá trình kiểm chứng tỉ lệ thuận với số trạng thái của giả định được sinh ra. Theo đó, chi phí của quá trình kiểm chứng, đặc biệt trong ngữ cảnh tiến hóa, có thể được giảm nếu sử dụng giả định có số trạng thái nhỏ và ngôn ngữ nhỏ.

So với phương pháp kiểm chứng giả định - đảm bảo sử dụng đặc tả LTS, các thuật toán kiểm chứng giả định - đảm bảo sử dụng đặc tả bằng lôgic mệnh đề có những ưu điểm vượt trội. Thứ nhất, về khả năng biểu diễn, đặc tả bằng lôgic mệnh đề tương đương với đặc tả bằng ô-tô-mát không đơn định và nó có thể biểu diễn súc tích hơn nhiều lần so với đặc tả bằng ô-tô-mát. Do đó, các thuật toán kiểm chứng sử dụng đặc tả bằng lôgic mệnh đề sinh các giả định có số trạng thái ít hơn nhiều lần các giả định sinh bởi các phương pháp sử dụng thuật toán L^*. Thứ hai, xét về tốc độ thì thuật

toán sinh giả định sử dụng đặc tả bằng lôgic mệnh đề sử dụng các thuật toán như CDNF, v.v. có tốc độ tốt hơn các thuật toán kiểm chứng sử dụng thuật toán L^*. Năm 2010, Chen và cộng sự đã đề xuất áp dụng thuật toán CDNF được đề xuất bởi Bshouty cho bài toán kiểm chứng giả định - đảm bảo cho hệ thống chuyển trạng thái được đặc tả bằng lôgic mệnh đề.

Trong phát triển phần mềm nói chung, ngoài các phần mềm không có ràng buộc thời gian được đặc tả bằng LTS hoặc lôgic mệnh đề, v.v, các hệ thống có ràng buộc thời gian cũng nhận được sự quan tâm đặc biệt của cộng đồng nghiên cứu và trong công nghiệp. Do đó, việc kiểm chứng các phần mềm này trở thành một xu hướng tất yếu do các đòi hỏi về chất lượng của các hệ thống có ràng buộc thời gian ngày càng cao và trong nhiều lĩnh vực của đời sống xã hội như Internet vạn vật, khoa học vũ trụ, v.v. Một trong các cách đặc tả các hệ thống có ràng buộc thời gian là sử dụng các ô-tô-mát ghi sự kiện (Event - Recording Automata - ERA). Năm 2014, Lin và cộng sự là nhóm tác giả đầu tiên đề xuất bài toán kiểm chứng giả định - đảm bảo cho phần mềm có ràng buộc thời gian được đặc tả bằng ô-tô-mát ghi sự kiện sử dụng thuật toán TL^*. Tuy phương pháp của Lin có thể sinh được giả định cho bài toán kiểm chứng giả định - đảm bảo cho phần mềm có ràng buộc thời gian, thuật toán có độ phức tạp cao nên vẫn rất khó khăn để có thể được áp dụng rộng rãi trong cộng đồng nghiên cứu và trong công nghiệp.

Từ các phân tích trên, luận án có hai mục tiêu chính. Mục tiêu thứ nhất là giảm chi phí kiểm chứng phần mềm dựa trên thành phần theo phương pháp giả định - đảm bảo trong ngữ cảnh tiến hóa phần mềm. Có hai giải pháp để giảm chi phí này. Giải pháp đầu tiên là sinh giả định mới mỗi lần phần mềm tiến hóa với tốc độ nhanh hơn khi kiểm chứng. Giải pháp thứ hai là giảm số lần cần phải sinh giả định mỗi khi phần mềm tiến hóa. Mục

tiêu thứ hai là giảm chi phí và cải tiến phương pháp kiểm chứng phần mềm có ràng buộc thời gian theo phương pháp giả định - đảm bảo.

1.2 Các đóng góp chính của luận án

Luận án đạt được ba kết quả chính như sau. Thứ nhất, luận án đề xuất một phương pháp sinh các giả định nhỏ nhất và mạnh nhất cục bộ để giảm chi phí của bài toán kiểm chứng giả định - đảm bảo. Một công cụ hỗ trợ cũng đã được cài đặt và thực nghiệm với một số ví dụ điển hình để minh chứng cho tính hiệu quả của phương pháp đề xuất.

Thứ hai, luận án đề xuất một phương pháp kiểm chứng hồi quy giả định - đảm bảo một cách hiệu quả cho phần mềm tiến hóa. Biến thể trả lời các truy vấn thành viên và thuật toán quay lui được tích hợp vào phương pháp đề xuất để giảm số lần giả định cần sinh lại khi kiểm chứng phần mềm đang tiến hóa. Một công cụ hỗ trợ cũng đã được cài đặt và thực nghiệm với một số hệ thống phổ biến trong cộng đồng nghiên cứu và cho những kết quả khả quan.

Thứ ba, luận án đề xuất ba cải tiến cho phương pháp kiểm chứng giả định - đảm bảo cho hệ thống phần mềm có ràng buộc thời gian. Đầu tiên, luận án loại bỏ pha học không có thời gian khỏi quá trình học làm giảm độ phức tạp về thời gian của quá trình học. Thứ hai, luận án đề xuất dùng một giá trị *cận trên* trong thuật toán trả lời các truy vấn ứng viên được cài đặt trong *Teacher* đóng vai trò như một chỉ báo để *Teacher* trả về kết quả "*không biết*" cho *Learner* để ngăn không cho quá trình học bị lặp vô hạn. Cuối cùng, luận án đề xuất một phương pháp phân tích phản ví dụ nhận được từ *Teacher* và một phương pháp tiền xử lý các phản ví dụ trước khi trả về *Learner*. Sự kết hợp của hai phương pháp này giúp quá trình kiểm chứng không bị lặp vô hạn trong nhiều trường hợp và tiến gần hơn đến kết

quả có thể kết luận được trong quá trình học. Luận án cũng đã cài đặt một công cụ và tiến hành thực nghiệm với một số hệ thống phổ biến và cho kết quả tốt.

1.3 Bố cục của luận án

Phần còn lại của luận án được cấu trúc như sau. Chương 2 giới thiệu các khái niệm cơ bản được sử dụng trong các nghiên cứu của luận án. Phương pháp sinh giả định nhỏ nhất và mạnh nhất cục bộ được trình bày trong Chương 3. Chương 4 đề xuất một phương pháp sinh giả định yếu nhất cục bộ và sử dụng giả định đó một cách hiệu quả trong việc kiểm chứng các phần mềm trong ngữ cảnh tiến hóa. Ba cải tiến cho phương pháp kiểm chứng phần mềm có ràng buộc thời gian được trình bày trong Chương 5. Cuối cùng, tổng kết các kết quả nghiên cứu của luận án và các hướng nghiên cứu tiếp theo được trình bày trong Chương 6.

Chương 2

Kiến thức nền tảng

2.1 Đặc tả và kiểm chứng giả định - đảm bảo cho các hệ thống đặc tả bằng LTS

2.1.1 Hệ thống chuyển trạng thái được gán nhãn

Phần này trình bày một số khái niệm cơ bản liên quan đến LTS sẽ được dùng trong Chương 3 và Chương 5 của luận án.

2.1.2 Kiểm chứng các hệ thống chuyển trạng thái được gán nhãn

Các khái niệm liên quan việc kiểm chứng giả định - đảm bảo cho CBS đặc tả bằng LTS được trình bày ở đây.

2.2 Đặc tả và kiểm chứng giả định - đảm bảo cho các hệ thống đặc tả bằng lôgic mệnh đề

2.2.1 Đặc tả hệ thống chuyển trạng thái bằng lôgic mệnh đề

Trong mục này, luận án trình bày một số khái niệm về các hàm lôgic được sử dụng trong Chương 4.

2.2.2 Kiểm chứng giả định - đảm bảo cho các hệ thống đặc tả bằng lôgic mệnh đề

Mục này trình bày luật kiểm chứng giả định - đảm bảo không quay vòng để kiểm chứng cho phần mềm và các thành phần của nó được đặc tả bằng

lôgic mệnh đề.

2.3 Đặc tả và kiểm chứng giả định - đảm bảo cho các hệ thống có ràng buộc thời gian

2.3.1 Hệ thống chuyển trạng thái có ràng buộc thời gian

Phần này trình bày một số khái niệm cơ bản về các hệ thống chuyển trạng thái có ràng buộc thời gian được dùng trong Chương 5 của luận án.

2.3.2 Kiểm chứng giả định - đảm bảo cho các hệ thống có ràng buộc thời gian

Trong mục này, các phép toán cốt lõi được sử dụng trong quá trình kiểm chứng được trình bày.

Chương 3

Phương pháp sinh giả định nhỏ nhất và mạnh nhất cục bộ cho việc kiểm chứng phần mềm dựa trên thành phần

3.1 Giới thiệu

Khi phần mềm tiến hóa, các nghiên cứu hiện tại đều chưa đề xuất được phương pháp sinh giả định nhỏ nhất và mạnh nhất để tăng tốc độ kiểm chứng. Do đó, mục đích của chương này là nghiên cứu phương pháp sinh các giả định nhỏ nhất và mạnh nhất để giảm chi phí của bài toán kiểm chứng giả định - đảm bảo. Mặc dù Giannakopoulou đã chỉ ra sự tồn tại của giả định yếu nhất, vẫn chưa có phương pháp để tìm được toàn bộ các giả định. Do đó, chương này chỉ trình bày một phương pháp sinh các giả định nhỏ nhất và mạnh nhất cục bộ trong tập các giả định có thể được tìm thấy bởi phương pháp đề xuất. Phương pháp đề xuất sử dụng một biến thể của kỹ thuật trả lời các câu truy vấn thành viên kết hợp với thuật toán học được cải tiến từ thuật toán của Cobleigh. Ngoài ra, phương pháp này cũng sử dụng các ứng viên cho giả định được sinh bởi thuật toán của Cobleigh làm cơ sở để phân tích. Với một ứng viên này, để có giả định nhỏ nhất, các ứng viên cho giả định nhỏ nhất A_i với kích thước tăng dần được kiểm tra. Việc

này được tiến hành bằng cách lấy tổ hợp chập t của các trạng thái từ tập trạng thái của A_i, với $1 \leq t \leq |A_i|$. Trong các ứng viên cho giả định nhỏ nhất có cùng kích thước t, với mỗi ứng viên C ($|L(C)| = n$), phương pháp này kiểm tra mọi khả năng từ khả năng chỉ có một chuỗi đến khả năng có $n-1$ chuỗi thuộc vào $L(C)$. Với cách làm này, giả định mạnh nhất được kiểm tra và tìm thấy trước, giả định yếu hơn sau. Ngoài ra, phương pháp này dừng ngay khi tìm được giả định đầu tiên thỏa mãn luật kiểm chứng giả định - đảm bảo. Do đó, giả định được sinh bởi phương pháp đề xuất là giả định nhỏ nhất và mạnh nhất cục bộ.

3.2 Các nghiên cứu liên quan

Phần này trình bày các nghiên cứu liên quan đến nghiên cứu này của luận án.

3.3 Phương pháp sinh giả định dựa trên thuật toán học L^*

3.3.1 Thuật toán học L^*

Thuật toán học L^* được đề xuất bởi Angluin và sau đó cải tiến bởi Rivest và Schapire. Luận án sử dụng phiên bản đã được *cải tiến* của thuật toán với tên ban đầu của nó, L^*. Quá trình học, minh họa trên Hình 3.1, được thực hiện thông qua tương tác của hai đối tượng $Learner$ (L^*) và $Teacher$. $Teacher$ có thể trả lời hai loại truy vấn sau từ $Learner$.

- Truy vấn thành viên: Cho một chuỗi $\sigma \in \Sigma^*$, có phải $\sigma \in U$? $Teacher$ trả lời $Learner$ là $true$ nếu $\sigma \in U$, ngược lại, $false$.
- Truy vấn ứng viên: Cho một DFA ứng viên D. Ngôn ngữ của D được tin là giống với U ("có phải $L(D) = U$?"). $Teacher$ trả lời $Learner$ là YES nếu $L(D) = U$. Ngược lại, $Teacher$ trả lời $Learner$ là NO

và đưa ra một phản ví dụ *cex*. *cex* là một chuỗi có khả năng thể hiện sự khác nhau của $L(D)$ và U.

3.3.2 Thuật toán sinh giả định sử dụng thuật toán học L^*

Cho một CBS M chứa hai thành phần M_1 và M_2 ($M = M_1 \parallel M_2$) và một thuộc tính an toàn p. Mục tiêu của bài toán kiểm chứng giả định - đảm bảo là kiểm tra nếu $M \models p$ mà không cần ghép nối M_1 với M_2. Thuật toán kiểm chứng được đề xuất bởi Cobleigh và cộng sự sinh một giả định thỏa mãn luật giả định - đảm bảo. Nếu giả định A như vậy tồn tại, thì $M \models p$. Ngược lại, $M \not\models p$. Chi tiết của thuật toán được trình bày trong Thuật toán 3.1.

3.4 Phương pháp sinh giả định nhỏ nhất và mạnh nhất cục bộ

3.4.1 Phương pháp sinh giả định mạnh nhất cục bộ

Một biến thể của thuật toán trả lời truy vấn thành viên

Kỹ thuật trả lời các truy vấn thành viên trong Thuật toán 3.1 dựa trên ngôn ngữ của giả định yếu nhất $L(A_W)$. Luận án đề xuất Thuật toán 3.2 trả lời các truy vấn thành viên từ *Learner*.

Thuật toán sinh giả định mạnh nhất cục bộ

Biến thể của thuật toán trả lời truy vấn thành viên được tích hợp vào quá trình kiểm chứng giả định - đảm bảo được mô tả trong Thuật toán 3.3. Đây là thuật toán được cải tiến từ Thuật toán 3.1 để sinh các giả định mạnh nhất cục bộ. Tính đúng đắn và độ phức tạp của phương pháp sinh giả định mạnh nhất cục bộ cũng được chứng minh trong luận án.

3.4.2 Phương pháp sinh giả định nhỏ nhất và mạnh nhất cục bộ

Thuật toán sinh giả định nhỏ nhất và mạnh nhất cục bộ

Phần này trình bày một thuật toán sinh các giả định nhỏ nhất và mạnh nhất cục bộ sử dụng biến thể của thuật toán trả lời các truy vấn thành viên được trình bày trong Mục 3.4.1. Chi tiết của thuật toán được trình bày trong Thuật toán 3.4. Thuật toán sử dụng các ứng viên cho giả định được sinh bởi Thuật toán 3.1 làm cơ sở cho quá trình học. Tại mỗi bước của quá trình học, khi bảng quan sát là đóng, tồn tại một ứng viên cho giả định cơ sở tương ứng. Trong khi thuật toán coi một số kết quả truy vấn thành viên "?" trong bảng quan sát tương ứng là $false$, thuật toán kiểm tra sự tồn tại ứng viên cho giả định khác có số trạng thái ít hơn hoặc bằng số trạng thái của ứng viên cho giả định cơ sở đó. Nếu tồn tại, ứng viên cho giả định đó sẽ được gửi đến *Teacher* trong một câu truy vấn ứng viên. Tính đúng đắn và độ phức tạp của thuật toán đề xuất cũng được chứng minh trong luận án.

3.5 Thực nghiệm và thảo luận

Luận án đã tiến hành các thực nghiệm để đánh giá, so sánh phương pháp sinh giả định nhỏ nhất và mạnh nhất cục bộ được trình bày trong Mục 3.4.2 với Thuật toán 3.1 được trình bày trong Mục 3.3.2. Kết quả cho thấy nhiều tiềm năng ứng dụng của các phương pháp đề xuất trong thực tiễn.

3.6 Tổng kết

Chương 3 đã trình bày về thuật toán sinh giả định nhỏ nhất và mạnh nhất cục bộ nhằm giảm chi phí tính toán cho bài toán kiểm chứng giả định - đảm bảo các CBS. Thuật toán đề xuất sử dụng một biến thể của kỹ thuật trả lời các câu truy vấn thành viên được đề xuất bởi Hùng và cộng sự. Chương 3 cũng đưa ra những chứng minh một cách hình thức tính đúng đắn của phương pháp đề xuất để sinh các giả định nhỏ nhất và mạnh nhất cục bộ.

Chương này đã trình bày các thực nghiệm để đánh giá và so sánh các giả định được sinh bởi phương pháp đề xuất với các giả định được sinh bởi phương pháp được đề xuất bởi Cobleigh và cộng sự. Kết quả thực nghiệm cho thấy, mặc dù phương pháp đề xuất cần nhiều thời gian hơn phương pháp của Cobleigh trong lần đầu sinh giả định, các giả định được sinh ra giúp giảm một cách hiệu quả không gian trạng thái của hệ thống áp dụng. Do đó, chi phí tính toán của việc kiểm chứng các hệ thống này cũng được giảm.

Chương 4

Phương pháp kiểm chứng hồi quy giả định - đảm bảo cho phần mềm tiến hóa

4.1 Giới thiệu

Một trong hai giải pháp để giảm chi phí kiểm chứng phần mềm trong ngữ cảnh tiến hóa là tăng số lần sử dụng lại giả định nhiều nhất có thể. Vì phần mềm thay đổi hàng ngày, nên số lần phải sinh lại giả định càng ít thì càng giảm được chi phí kiểm chứng phần mềm thay đổi. Hơn nữa, từ phân tích trong Mục 4.5 phía dưới, giả định yếu (giả định có ngôn ngữ lớn) có thể giúp đạt được điều này và đóng vai trò quan trọng trong kiểm chứng hồi quy phần mềm thay đổi. Mặt khác, hiện chưa có nghiên cứu nào được tiến hành về việc sinh các giả định có ngôn ngữ yếu nhất và sử dụng đặc tả bằng lôgic mệnh đề. Do đó, nghiên cứu của chương này tập trung vào cải tiến thuật toán học của Chen và cộng sự và sinh các giả định yếu nhất cục bộ. Các giả định này có thể được dùng lại một cách hiệu quả để giảm chi phí kiểm chứng hồi quy phần mềm trong ngữ cảnh tiến hóa phần mềm.

Để đạt được mục tiêu trên, luận án đề xuất một biến thể của kỹ thuật trả lời các truy vấn thành viên cho hai thực thể thuật toán học CDNF ι (hàm khởi tạo) và τ (hàm chuyển trạng thái). Dựa vào biến thể này, luận án đề xuất một thuật toán học quay lui (thuật toán LWAG) có thể sinh

các giả định yếu hơn các giả định được sinh bởi thuật toán được đề xuất bởi Chen và cộng sự (thuật toán CBAG). Điều này dẫn đến một kết quả quan trọng trong ngữ cảnh tiến hóa phần mềm: các giả định được sinh bởi thuật toán LWAG có thể giảm số lần cần sinh lại giả định khi kiểm chứng các phần mềm đã bị thay đổi. Biến thể trả lời các truy vấn thành viên và thuật toán LWAG được tích hợp vào một phương pháp để giảm số lần giả định cần sinh lại khi kiểm chứng phần mềm đang tiến hóa.

4.2 Các nghiên cứu liên quan

Phần này trình bày các nghiên cứu liên quan đến việc kiểm chứng giả định - đảm bảo cho phần mềm tiến hóa. Tuy nhiên, vẫn chưa có nghiên cứu nào sử dụng các giả định yếu nhất cục bộ trong ngữ cảnh tiến hóa để giảm số lần sinh lại giả định.

4.3 Phương pháp sinh giả định dựa trên thuật toán CDNF

4.3.1 Thuật toán CDNF

Gọi X là một tập biến lôgic và λ(X) là một hàm lôgic trên tập X. CDNF là một thuật toán học có thể học biểu diễn chính xác của λ(X) sau một số hữu hạn các bước học. Sử dụng cùng ý tưởng với thuật toán L^*, CDNF dựa trên một thực thể $Teacher$ (biết về λ(X)) khi thực hiện quá trình học. Thực thể $Teacher$ phải có thể trả lời hai loại truy vấn sau:

- $Membership\ queries\ MEM(v)$: Cho một phép gán v trên X, nếu $\lambda[v] = T(true)$, $Teacher$ trả lại yes cho $Learner$. Ngược lại, $Teacher$ trả lại no.
- $Equivalence\ queries\ EQ(h)$: Cho một hàm lôgic ứng viên h trên X, nếu ứng viên h tương đương với hàm cần học λ, $Teacher$ trả lại yes.

Ngược lại, $Teacher$ trả lại một phép gán v trên X với $h[v] \neq \lambda[v]$. Phép gán v đóng vai trò là phản ví dụ cho câu truy vấn ứng viên.

4.3.2 Thuật toán sinh giả định dựa trên CDNF

Mục này trình bày thuật toán sinh giả định dựa trên thuật toán CDNF, được gọi là thuật toán CBAG. Thuật toán CBAG sử dụng một số thuật toán sau: thuật toán trả lời truy vấn thành viên (thuật toán OMQ); truy vấn ứng viên (thuật toán EQ); và thuật toán kiểm tra một phản ví dụ α có thể được dùng để sinh một ứng viên cho giả định tốt hơn hay α thực sự là phản ví dụ cho $M_0 \parallel M_1 \not\models p$ (thuật toán IW). Tính đúng đắn của các thuật toán này được trình bày trong nghiên cứu của Chen và cộng sự.

Thuật toán sinh giả định ban đầu - CBAG

Thuật toán CBAG tạo ra một $Learner$ có hai thực thể của thuật toán CDNF, được gọi là $CDNF_\iota$ và $CDNF_\tau$. Hai thực thể này tương tác với $Teacher$ có thuật toán OMQ, EQ, và IW được cài đặt. Tổng quan của thuật toán CBAG được trình bày trong Hình 4.1.

4.4 Phương pháp sinh giả định yếu nhất cục bộ

4.4.1 Biến thể của thuật toán trả lời các truy vấn thành viên

Luận án giới thiệu thuật toán IMQ, là một cải tiến của thuật toán OMQ. Trong thuật toán IMQ, luận án sử dụng một ký hiệu mới là $question$ được trả về cho $Learner$ khi $\theta = F$, trong đó θ là $\iota_1(\mathrm{X}_1)$ hoặc $\tau_1(\mathrm{X}_1, \mathrm{X}'_1)$. Ngược lại, khi $\theta = T$, thuật toán trả lại yes cho $Learner$ giống với thuật toán OMQ.

4.4.2 Thuật toán quay lui sinh giả định yếu nhất cục bộ

Phần này trình bày thuật toán quay lui (được gọi là thuật toán LWAG) sinh các giả định yếu hơn các giả định được sinh bởi thuật toán CBAG.

Thuật toán LWAG sử dụng biến thể trả lời truy vấn thành viên trong thuật toán IMQ và được trình bày trong Hình 4.2.

4.4.3 Tính đúng đắn

Tính đúng đắn của thuật toán LWAG được luận án chứng minh trong Mục 4.4.3.

4.5 Phương pháp kiểm chứng từng phần cho phần mềm dựa trên thành phần đang tiến hóa

Các giả định có thể được sử dụng lại, như các giả định yếu, đóng vai trò quan trọng trong việc giảm chi phí kiểm chứng khi được sử dụng trong phương pháp đề xuất trong mục này.

4.5.1 Phương pháp kiểm chứng giả định - đảm bảo cho phần mềm đang tiến hóa

Mục này trình bày một phương pháp để kiểm chứng các CBS trong ngữ cảnh tiến hóa. Phương pháp được phát triển dựa trên phương pháp của Hùng và cộng sự.

4.5.2 Một ví dụ

Mục này trình bày một ví dụ cho việc sinh giả định sử dụng thuật toán LWAG và phương pháp được trình bày trong Mục 4.5.1 để kiểm chứng hồi quy một hệ thống đang tiến hóa.

4.6 Thực nghiệm

Các thực nghiệm được tiến hành để làm nổi bật hai điểm chính sau: (i) so sánh giữa thuật toán CBAG và thuật toán LWAG; và (ii) so sánh phương pháp trong Mục 4.5.1 giữa trường hợp sử dụng giả định được sinh bởi thuật toán CBAG và trường hợp sử dụng giả định được sinh bởi thuật toán LWAG

sau khi phần mềm được thay đổi.

4.6.1 So sánh các thuật toán sinh giả định

Kết quả thực nghiệm cho thấy truy vấn thành viên cho hàm khởi tạo, truy vấn thành viên cho hàm chuyển trạng thái của hai thuật toán là khác nhau; các giả định được sinh bởi thuật toán LWAG là yếu hơn các giả định được sinh bởi thuật toán CBAG. Thuật toán LWAG cần nhiều thời gian hơn nhưng dung lượng bộ nhớ được sử dụng bởi hai thuật toán là tương đương.

4.6.2 Tính hiệu quả của các giả định được sinh ra trong ngữ cảnh tiến hóa

Kết quả thực nghiệm cho thấy thuật toán LWAG và phương pháp đề xuất giảm được số lần giả định cần được sinh lại sau khi M_1 tiến hóa và cho thấy chúng có thể giảm được chi phí kiểm chứng cho các CBS đã được thay đổi. Việc sử dụng bộ nhớ trong cả hai trường hợp (sử dụng các giả định được sinh bởi thuật toán CBAG và thuật toán LWAG) là tương đương nhau.

4.6.3 Thảo luận

Phần này đưa ra các thảo luận về tính mở rộng và khả năng áp dụng của phương pháp đề xuất đối với các hệ thống lớn trong thực tế.

4.7 Tổng kết

Chương 4 đã trình bày một phương pháp hiệu quả để kiểm chứng hồi quy giả định - đảm bảo cho phần mềm đang tiến hóa. Phương pháp này sử dụng một biến thể của phương pháp trả lời các truy vấn thành viên tích hợp vào thuật toán LWAG để sinh các giả định yếu nhất cục bộ. Thuật toán đề xuất và các giả định yếu nhất cục bộ sinh ra được sử dụng trong phương pháp đề xuất để kiểm chứng hồi quy cho phần mềm đang tiến hóa. Việc sử dụng

các giả định này có thể giảm số lần sinh lại giả định khi kiểm chứng hồi quy phần mềm đang tiến hóa.

Các kết quả thực nghiệm cho thấy thuật toán đề xuất có thể sinh các giả định yếu hơn, với chi phí thời gian lớn hơn. Thực nghiệm cũng cho thấy việc sử dụng các giả định yếu nhất cục bộ làm giả định bắt đầu của quá trình kiểm chứng hồi quy làm giảm đáng kể số lần phải sinh lại giả định khi kiểm chứng phần mềm đang tiến hóa. Một số thảo luận về kết quả thực nghiệm cũng được trình bày trong chương.

Chương 5

Ba cải tiến cho phương pháp kiểm chứng giả định - đảm bảo cho phần mềm có ràng buộc thời gian

5.1 Giới thiệu

Lin và cộng sự là nhóm tác giả đầu tiên đề xuất phương pháp áp dụng kiểm chứng giả định - đảm bảo cho các hệ thống có ràng buộc thời gian một cách hoàn toàn tự động. Phương pháp này chứa hai pha sinh giả định, trong đó pha đầu tiên là sinh giả định giả định không có ràng buộc thời gian và pha hai là sinh các giả định có ràng buộc thời gian. Nếu giả định không có ràng buộc thời gian có thể được sinh ra trong pha sinh giả định đầu tiên, nó sẽ được sử dụng làm đầu vào cho pha sinh giả định tiếp theo để sinh giả định có ràng buộc thời gian. Tuy nhiên, phương pháp này có một số hạn chế như sau. Hạn chế đầu tiên là cả hai pha sinh giả định đều có độ phức tạp về thời gian lớn. Điều này làm cho phương pháp sinh giả định hai pha có độ phức tạp thời gian lớn. Hạn chế thứ hai là việc chia quá trình học ra làm hai giai đoạn không giúp cho phương pháp bao phủ toàn bộ các trường hợp mà giả định có thể được sinh ra. Vấn đề này tồn tại vì có hai dạng luật kiểm chứng giả định - đảm bảo được sử dụng trong kiểm chứng

phần mềm: luật giả định - đảm bảo quay vòng (CIRC-AG) là đúng và đầy đủ; luật giả định - đảm bảo không quay vòng (NC-AG) là không đầy đủ. Trong trường hợp tổng quát, tồn tại một vùng mà trong đó *Teacher* không biết rằng giả định có tồn tại hay không. Xét trường hợp điển hình trong đó hệ thống đã cho M thỏa mãn một thuộc tính an toàn cho trước p, nhưng *Teacher* không biết giả định thỏa mãn luật NC-AG có tồn tại hay không. Quá trình sinh giả định trong các nghiên cứu có thể chạy vô hạn. Từ đó, luận án cần thêm một hướng dẫn cho *Teacher* trả lại kết quả *"không biết"* để *Learner* dừng quá trình học. Luận án hiện thực hóa ý tưởng này bằng cách thêm một giới hạn *cận trên* vào thuật toán trả lời các truy vấn ứng viên được cài đặt tại *Teacher*. Quá trình sinh giả định được trình bày trong Hình 5.1.

5.2 Các nghiên cứu liên quan

Phần này trình bày các nghiên cứu liên quan đến kiểm chứng giả định - đảm bảo phần mềm có ràng buộc về thời gian.

5.3 Phương pháp sinh giả định sử dụng quá trình sinh giả định hai pha

Trong phương pháp sinh giả định được đề xuất bởi Lin và cộng sự, quá trình sinh giả định được chia thành hai pha. Trong cả hai pha sinh giả định, quá trình sinh giả định được thực hiện thông qua sự tương tác của hai thực thể $Learner$ và $Teacher$. Chi tiết của phương pháp được trình bày trong Hình 5.2.

5.3.1 Pha thứ nhất – pha kiểm chứng không có ràng buộc thời gian

Trong pha này, thuật toán sinh một giả định dưới dạng một DFA A^{ut} sử dụng phương pháp sinh giả định được đề xuất bởi Cobleigh và cộng sự để sinh giả định cho việc kiểm chứng hệ thống không có ràng buộc thời gian $M^{ut} = M_1^{ut} \parallel M_2^{ut}$ và thuộc tính không có ràng buộc thời gian p^{ut}.

Nếu pha này có thể sinh giả định A^{ut}, A^{ut} sẽ được dùng trong pha thứ hai để sinh giả định có ràng buộc thời gian. Nếu pha một trả lại kết quả là A^{ut} không tồn tại với phản ví dụ π, π sẽ được sử dụng để kiểm tra xem nó có phải là phản ví dụ thực sự hay không bằng việc kiểm tra các biểu thức $L(\pi) \subseteq L(M_1 \parallel \bar{p})$ và $L(\pi) \subseteq L(M_2)$. Nếu π thỏa mãn cả hai biểu thức này, luận án có thể kết luận rằng $M \not\models p$. Ngược lại, luận án không thể kết luận gì và cần tiến hành pha học có thời gian.

5.3.2 Pha thứ hai – pha kiểm chứng có ràng buộc thời gian

Pha học này được thực hiện bằng việc sử dụng thuật toán học TL^* tương tác với một $Teacher$ có thể trả lời hai loại câu truy vấn sau.

- Truy vấn thành viên ($Q_m(\sigma)$): Cho một từ gác σ, nếu σ thuộc ngôn ngữ của giả định A sẽ được sinh ra, $Teacher$ trả lại yes. Ngược lại, $Teacher$ trả lại no.

- Truy vấn ứng viên ($Q_c(C)$): Cho một ứng viên cho giả định C, nếu $L(C) \equiv U$, với U là ngôn ngữ của giả định sẽ được sinh ra, $Teacher$ trả lại yes. Ngược lại, $Teacher$ trả lại no với một phản ví dụ cex làm bằng chứng cho $M = M_1 \parallel M_2 \not\models p$ nếu $Teacher$ phân tích cex và thấy rằng M thực sự vi phạm p. Nếu sau khi phân tích cex, $Teacher$ thấy rằng $Learner$ có thể sinh ứng viên cho giả định tốt hơn, $Teacher$ trả lại $continue$ và phản ví dụ cex. $Learner$ có thể dựa trên phản ví dụ cex đó để làm ứng viên cho giả định hiện tại tốt lên.

Chi tiết của thuật toán TL^* được trình bày trong Hình 5.2.

5.4 Phương pháp sinh giả định sử dụng quá trình học một pha

Khi cài đặt thuật toán sinh giả định hai pha được đề xuất bởi Lin và cộng sự, nếu chỉ cài đặt thuật toán trả lời các truy vấn ứng viên được đề xuất trong *Teacher* một cách đơn giản, có nhiều trường hợp quá trình sinh giả định chạy vô hạn. Mục này trình bày một ví dụ cho một quá trình sinh giả định vô hạn và thuật toán sinh giả định có chứa một kỹ thuật tìm hậu tố t.

5.4.1 Ví dụ cho quá trình học vô hạn

Mục này trình bày một ví dụ trong đó quá trình kiểm chứng chạy vô hạn với thuật toán được đề xuất bởi Lin và cộng sự được cài đặt trong *Learner* và *Teacher*.

5.4.2 Thuật toán sinh giả định

Luận án trình bày thuật toán có chứa phương pháp phân tích phản ví dụ được trả về từ *Teacher*. Chi tiết của thuật toán được trình bày trong Thuật toán 5.1.

5.5 Các thuật toán thực thi *Teacher*

5.5.1 Thuật toán trả lời truy vấn thành viên

Để trả lời câu truy vấn thành viên rằng một từ σ có thuộc vào ngôn ngữ của giả định A sẽ được sinh ra hay không, luận án dựa vào sự đoán nhận của từ gác như trình bày trong Thuật toán 5.2. σ được kiểm tra có thuộc vào ngôn ngữ của $M_1 \parallel \overline{p}$. Nếu đúng, thuật toán trả về no. Ngược lại, thuật toán trả về yes.

5.5.2 Thuật toán trả lời truy vấn ứng viên

Để kỹ thuật phân tích phản ví dụ được trình bày trong Mục 5.4.2 được sử dụng một cách hiệu quả, luận án cần một biến thể của thuật toán trả lời các truy vấn ứng viên không trả về các phản ví dụ đã được trả về *Learner* trước đó. Biến thể của thuật toán trả lời các truy vấn ứng viên được đề xuất bởi Lin và cộng sự được trình bày trong Thuật toán 5.3.

5.5.3 Tính đúng đắn

Các chứng minh cho tính đúng đắn của các thuật toán đề xuất cũng được trình bày trong luận án.

5.6 Thực nghiệm

Các thực nghiệm đã được tiến hành với một số hệ thống quen thuộc trong cộng đồng nghiên cứu và cho kết quả khả quan đối với phương pháp học giả định một pha.

5.7 Tổng kết

Chương này trình bày ba cải tiến cho phương pháp sinh giả định hai pha được đề xuất bởi Lin và cộng sự. Cải tiến đầu tiên là loại bỏ pha sinh giả định không có ràng buộc thời gian từ quá trình sinh giả định giúp giảm độ phức tạp thời gian của quá trình kiểm chứng. Cải tiến thứ hai là đưa ra một giới hạn *"cận trên"* giúp cho *Teacher* trả lại "*không biết*" cho *Learner* để dừng quá trình học. Cuối cùng, chương này trình bày một kỹ thuật phân tích phản ví dụ từ *Teacher* và phương pháp tiền xử lý các phản ví dụ trước khi trả về *Learner*. Cải tiến này ngăn quá trình kiểm chứng khỏi việc chạy vô hạn trong nhiều trường hợp. Các kết quả thực nghiệm cho thấy các cải tiến cho kết quả tốt khi thực nghiệm với một số hệ thống phổ biến.

Chương 6

Kết luận

6.1 Các kết quả đạt được

Luận án đã đề xuất hai phương pháp góp phần giảm chi phí kiểm chứng phần mềm dựa trên thành phần trong ngữ cảnh tiến hóa phần mềm. Thứ nhất, luận án đề xuất phương pháp sinh giả định nhỏ nhất và mạnh nhất cục bộ cho quá trình kiểm chứng phần mềm được đặc tả bằng LTS. Thứ hai, luận án đề xuất phương pháp sinh và sử dụng giả định yếu nhất cục bộ khi kiểm chứng phần mềm tiến hóa được đặc tả bằng lôgic mệnh đề. Ngoài ra, luận án cũng đề xuất ba cải tiến cho phương pháp kiểm chứng giả định - đảm bảo cho phần mềm dựa trên thành phần có ràng buộc về thời gian.

6.2 Hướng phát triển tiếp theo

Mặc dù các kết quả nghiên cứu đã có những đóng góp cụ thể như đã trình bày trong Mục 6.1, các kết quả này còn có những hạn chế cần khắc phục. Nghiên cứu theo tiếp của luận án hướng đến giải quyết các hạn chế này. Các hướng phát triển tiếp theo của luận án bao gồm: giảm độ phức tạp của các thuật toán đề xuất; xây dựng giao diện cho các công cụ thực nghiệm; đưa các công cụ này vào kiểm chứng các hệ thống lớn trong thực tế, v.v.

Danh mục các công trình khoa học của tác giả liên quan đến luận án

1. **Hoang-Viet Tran**, Pham Ngoc Hung, Viet-Ha Nguyen, Toshiaki Aoki (2019). "A Framework For Assume-Guarantee Regression Verification Of Evolving Software", Science of Computer Programming Journal, ISSN 0167-6423. (Accepted). (ISI Indexed)

2. **Hoang-Viet Tran**, Quang-Trung Nguyen, and Pham Ngoc Hung (2019). "On Implementation of the Improved Assume-Guarantee Verification Method for Timed Systems." In Soict '19, December 4 – 6, 2019, Hanoi - Ha Long Bay, Vietnam. ACM, NY, USA, pp. 457–464.

3. **Hoang-Viet TRAN**, Ngoc Hung PHAM, Viet Ha NGUYEN (2019). "On Locally Minimum and Strongest Assumption Generation Method for Component-Based Software Verification", IEICE Transactions on Information and Systems. ISSN 0916-8532, Vol.E102-D, No.8, pp.1449-1461. (ISI Indexed)

4. **Hoang-Viet Tran** and Ngoc Hung PHAM (2018). "On Locally Strongest Assumption Generation Method for Component-Based Software Verification." VNU Journal of Science: Computer Science and Communication Engineering, vol. 34, no. 2, pp. 16–32. ISSN 2588-1086.

5. **Hoang-Viet Tran**, Ngoc Hung PHAM, Dang Van Hung (2018). "On Improvement of Assume-Guarantee Verification Method for Timed Component-Based Software." 10th International Conference on Knowledge and Systems Engineering (KSE), Ho Chi Minh City, pp. 270-275.

6. Chi-Luan Le, **Hoang-Viet Tran**, and Pham Ngoc Hung (2017). "On Implementation of the Assumption Generation Method for Component-Based Software Verification." Advanced Topics in Intelligent Information and Database Systems. Springer International Publishing, pp. 549-558.

7. Chi-Luan Le, **Hoang-Viet Tran**, Pham Ngoc Hung (2016), "A Framework for Modeling and Modular Verifying of Component-based System Designs", VNU Journal of Science: Computer Science and Communication Engineering, vol. 32 , no. 2, pp. 31-42.

8. **Hoang-Viet Tran**, Chi-Luan Le and Ngoc Hung Pham (2016), "A Strongest Assumption Generation Method for Component-Based Software Verification", In Addendum Proc. of the 2016 IEEE-RIVF International Conference on Computing and Communication Technologies, pp. 1-6.

Danh mục này gồm 08 công trình.