BỘ GIÁO DỤC VÀ ĐÀO TAO
TRƯỜNG ĐẠI HỌC QUY NHƠN

ĐỖ ĐĂNG KHOA

ĐỘ TƯƠNG TỰ VỀ HÀNH VI CỦA CHƯƠNG TRÌNH VÀ LÀM THỰC NGHIỆM.

Chuyên ngành: Khoa học máy tính
Mã số: 60 48 01 01

TÓM TẮT LUẬN VĂN THẠC SĨ.....

Bình Định - Năm 2018

Công trình được hoàn thành tại
TRƯỜNG ĐẠI HỌC QUY NHƠN

Người hướng dẫn: **Ts. PHẠM VĂN VIỆT**

Phản biện 1: .....
Phản biện 2: .....

Luận văn được bảo vệ tại Hội đồng đánh giá luận văn thạc sĩ chuyên ngành Khoa học máy tính, ngày ... tháng ... năm 2018 tại Trường Đại học Quy Nhơn.

Có thể tìm hiểu luận văn tại:

- ....., Trường Đại học Quy Nhơn

LỜI CAM ĐOAN

Tôi xin cam đoan: Luận văn này là công trình nghiên cứu thực sự của cá nhân, được thực hiện dưới sự hướng dẫn khoa học của **Ts. Phạm Văn Việt**.

Các số liệu, những kết luận nghiên cứu được trình bày trong luận văn này trung thực và chưa từng được công bố dưới bất cứ hình thức nào.

Tôi xin chịu trách nhiệm về nghiên cứu của mình.

LỜI CẢM ƠN

Tôi xin chân thành cảm ơn sự hướng dẫn, chỉ dạy và giúp đỡ tận tình của các thầy cô giảng dạy sau đại học - Trường đại học Quy Nhơn.

Đặc biệt, tôi cảm ơn thầy **Ts.Phạm Văn Việt**, giảng viên bộ môn Công nghệ phần mềm, khoa Công nghệ thông tin, Trường Đại học Quy Nhơn đã tận tình hướng dẫn truyền đạt những kiến thức và kinh nghiệm quý báu để giúp tôi có đầy đủ kiến thức và nghị lực hoàn thành luận văn này.

Và tôi xin cảm ơn bạn bè, đồng nghiệp và những người thân trong gia đình đã tin yêu, động viên giúp tôi thêm nghị lực trong quá trình học tập và nghiên cứu.

Mặc dù đã cố gắng rất nhiều trong việc thực hiện luận văn, song với thời gian có hạn, nên luận văn không thể tránh khỏi những thiếu sót và chưa hoàn chỉnh. Tôi rất mong nhận được ý kiến đóng góp của quý Thầy Cô và các bạn.

Một lần nữa, tôi xin chân thành cảm ơn!.

HỌC VIÊN

Đỗ Đăng Khoa

Mục lục

DANH MỤC CÁC KÝ HIỆU

DANH MỤC CÁC BẢNG, BIỂU, SƠ ĐỒ, HÌNH VẼ

MỞ ĐẦU

1. Lý do chọn đề tài

Trong những năm gần đây, xu hướng đào tạo lập trình viên nói riêng và công nghệ phần mềm nói chung đang ngày càng trở nên phổ biến. Các trường đại học và một số trung tâm đào tạo đã cho ra đời nhiều chương trình đào tạo phong phú về nội dung, đa dạng về hình thức và thu hút được nhiều sự quan tâm.

Trong mỗi khóa học, số lượng học viên thường có hằng trăm, hằng ngàn người tham gia, nhưng chỉ một vài giáo viên giảng dạy. Trong đó, chất lượng việc quản lý, truyền đạt nội dung của giáo viên và mức độ hiểu biết, nắm bắt nội dung chương trình học của học viên là yêu cầu tất cả các khóa học cần đạt được. Để đạt được điều đó, một yêu cầu bắt buộc đó là giáo viên phải đọc và hiểu tất cả các đoạn code của học sinh, nhưng công việc này lại tốn quá nhiều thời gian. Nếu bỏ qua các công việc như vậy thì người dạy không thể theo dõi được quá trình học tập của người học. Về phía người học, yêu cầu đặt ra là phải tiến bộ theo thời gian, nắm vững lý thuyết và thành thạo kỹ năng lập trình. Những không phải lúc nào gặp khó khăn người học đều có sự hỗ trợ kịp thời từ giảng viên. Họ có thể nhờ sự giúp đỡ từ đồng nghiệp, bạn bè, nhưng những người được nhờ giúp đỡ chưa chắc đã đủ trình độ, kinh nghiệm hoặc thời gian để ngồi bên cạnh giúp đỡ người học khi cần.

Để giảm bớt những khó khăn nêu trên, một công cụ hỗ trợ giáo viên và học sinh hiệu quả hơn, tiết kiệm thời gian hơn đó là một công cụ tự động hóa (có thể một phần) việc đánh giá kết quả lập trình của học sinh, cũng như hỗ trợ theo giõi sự tiến bộ của học sinh. Công cụ tự động hóa này sẽ tính toán, định lượng tỷ lệ chính xác sự tương tự về hành vi giữa chương trình của người học và chương trình của người dạy đưa ra trước đó. Dựa trên kết quả, giáo viên sẽ đánh giá được kỹ năng lập trình của học sinh, sự giống nhau giữa hai chương trình càng cao thì tỷ lệ độ tương tự càng cao, điểm số càng cao. Nếu điểm số thấp, người học có thể quay lại kiểm tra để viết mã chương trình đúng hơn, hạn chế được nguy cơ tìm ẩn trong cách viết chương trình của người học.

Những đánh giá này có thể thực hiện được nếu ta đo được độ tương tự giữa các chương trình có độ chính xác cao. Đề tài "Độ tương tự về hành vi của các chương trình và làm thực nghiệm" với mục đích sẽ giải quyết các vấn đề nếu trên.

2. Mục tiêu, đối tượng, phạm vi nghiên cứu

2.1. Mục tiêu nghiên cứu

2.1.1. Mục tiêu nghiên cứu chính

Đánh giá độ tương tự về hành vi của các chương trình

2.1.2. Mục tiêu nghiên cứu cụ thể

- Tìm hiểu sự tương tự ngữ nghĩa của chương trình

- Tìm hiểu kỹ thuật, công cụ sinh Test Case tự động

- Phân tích các độ đo và áp dụng kỹ thuật sinh Test Case tự động trên các độ đo

- Tìm cách kết hợp các độ đo với nhau

- Tìm một số ứng dụng của độ đo, chọn một ứng dụng để làm thực nghiệm, đánh giá kết quả thực nghiệm

2.2. Đối tượng, phạm vi nghiên cứu

2.2.1. Đối tượng nghiên cứu

- Kỹ thuật sinh Test Case

- Độ đo tương tự hành vi

- Một số ứng dụng của độ đo

2.2.2. Phạm vi nghiên cứu

- Đo độ tương tự hành vi dựa vào Test Case

- Thực nghiệm, đánh giá trên các chương trình C Sharp

3. Phương pháp nghiên cứu, thực nghiệm

3.1. Nghiên cứu lý thuyết

- Độ tương tự hành vi

- Một số kỹ thuật sinh Test Case tự động

- Độ đo tương tự hành vi dựa trên Test Case

- So sánh, kết hợp các độ đo

3.2. Thực nghiệm

- Tiến hành cài kỹ thuật đo độ tương tự hành vi

- Thực nghiệm trên dữ liệu thực của CodeHunt

- Phân tích, đánh giá dựa trên kết quả thực nghiệm

Chương 1

GIỚI THIỆU

1.1 Giới thiệu chung

Hiện nay, ngành Công nghệ thông tin với các chương trình đào tạo lập trình Online, kỹ sư phần mềm... đang trở nên rất phổ biến. Trên thế giới cũng có nhiều chương trình đào tạo nổi tiếng như Massive Open Online Courses (MOOC), edX, Coursera, Udacity thu hút hằng ngàn sinh viên theo học. Một số chương trình học lập trình online như Pex4Fun hay Code Hunt là một nền tảng học lập trình online thông qua trò chơi.

Những lớp học như thế này thường đặt ra một số thách thức, làm thế nào để kiểm soát được chất lượng học tập của người học. Trong khi các lớp học có hằng trăm người tham gia, nhưng thành viên tham giảng dạy chỉ vài người trong một lớp học. Hằng ngày, người giáo viên phải thường xuyên kiểm tra, nhắc nhở và phải đọc, nghiên cứu giúp đỡ cho học viên. Chỉ riêng việc đọc và hiểu code do học viên viết ra đã tốn quá nhiều thời gian của người dạy, nếu như bỏ qua thì khó có thể đánh giá được chất lượng của người học.

Để giảm bớt những vất vả, khó khăn của người dạy và người học. Một công cụ hỗ trợ, tự động đánh giá hành vi chương trình của người học sẽ giúp tiết kiệm được thời gian, giúp cho giáo viên việc quản lý chất lượng học tập của học viên được tốt hơn. Ví dụ, công cụ sẽ tự động đánh giá hành vi, so sánh hành vi chương trình của người học viết với hành vi chương trình của giáo viên. Nếu hành vi của hai chương trình có tỷ lệ giống nhau càng cao thì điểm số cho chương trình của người học càng cao. Ngược lại, nếu điểm số thấp người dạy và học sẽ tìm hiểu nguyên nhân vì sao và có hướng khác phục những hạn chế mà người học đang gặp phải. Đây cũng là một cách giúp cho người học không đi lệch khỏi kiến thức của chương trình đào tạo, tiết kiệm được thời gian cả người dạy và người học.

Ý tưởng trong việc đánh giá độ tương tự hành vi của hai chương trình đó là tính toán tìm ra một miền giá trị đầu vào chung cho cả hai chương trình. Đưa từng giá trị đầu vào chạy đồng thời trên cả hai chương trình và so sánh kết quả đầu ra của hai chương trình. Tuy nhiên, việc này sẽ không đạt, không khả thi, vì trong những trường hợp chương trình có miền giá trị đầu vào lớn hoặc vô hạn. Vì vậy, để đánh giá

độ tương tự hành vi của hai chương trình khả thi hơn nếu chúng ta chỉ sử dụng một sô giá trị đầu vào đại diện cho toàn bộ miền giá trị đầu vào chung của hai chương trình.

Kỹ thuật Dynamic Symbolic Execution (DSE) là một kỹ thuật thu thập các ràng buộc từ các nhánh của chương trình, phủ nhận lại có hệ thống một phần các ràng buộc để tạo ra giá trị đầu vào cho một chương trình. Hiện nay, đã có nhiều công cụ sử dụng kỹ thuật DSE để giải quyết các ràng buộc một cách mạnh mẽ, tạo ra các giá trị đầu vào tin cậy có độ phủ cao.

Tên Công cụ	**Ngôn ngữ**	**Url**
KLEE	LLVM	klee.github.io/
JPF	Java	babelfish.arc.nasa.gov/trac/jpf
jCUTE	Java	github.com/osl/jcute
janala2	Java	github.com/ksen007/janala2
JBSE	Java	github.com/pietrobraione/jbse
KeY	Java	www.key-project.org/
Mayhem	Binary	forallsecure.com/mayhem.html
Otter	C	bitbucket.org/khooyp/otter/overview
Rubyx	Ruby	www.cs.umd.edu/ avik/papers/ssarorwa.pdf
Pex	.NET Framework	research.microsoft.com/en-us/projects/pex/
Jalangi2	JavaScript	github.com/Samsung/jalangi2
Kite	LLVM	www.cs.ubc.ca/labs/isd/Projects/Kite/
pysymemu	x86-64 / Native	github.com/feliam/pysymemu/
Triton	x86 and x86-64	triton.quarkslab.com
BE-PUM	x86	https://github.com/NMHai/BE-PUM

Chương 2

KIẾN THỨC CƠ SỞ

2.1 Một số khái niệm, định nghĩa

2.1.1 Kỹ thuật Dynamic symbolic execution (DSE)

Định nghĩa

Là một kỹ thuật thu thập các ràng buộc từ các nhánh của chương trình và phủ nhận một phần các ràng buộc để tạo ra dữ liệu đầu vào của chương trình và có độ phủ cao

2.2 Kỹ thuật lập trình trên C sharp

Chương 3

ĐỘ TƯƠNG TỰ HÀNH VI CỦA CHƯƠNG TRÌNH

3.1 Một số khái niệm, định nghĩa

Để định lượng hai chương trình tương tự nhau, chúng ta định nghĩa các khái niệm về hành vi chương trình và các định nghĩa liên quan đến sự tương tự của hai chương trình, và ví dụ minh họa cho các định nghĩa.

3.1.1 Hành vi của chương trình (Program Execution)

Hành vi chương trình P là thực hiện hàm: $P \times I \rightarrow O$. Với giá trị đầu vào $i \in I$, giá trị đầu ra $o \in O$. Trong đó I là miền các trị đầu vào của chương trình P và O là tập hợp các giá trị đầu ra của chương trình P.

3.1.2 Tương đương hành vi (Behavioral Equivalence)

Hai chương trình P_1 và P_2 có cùng một miền các giá trị đầu vào I và tương đương về hành vi nếu $exec(P_1; I) = exec(P_2; I)$, với $\forall i \in I$ $exec(P_1; i) = exec(P_2; i)$.

Ví dụ:

Chương trình 3.1: Tính y, sử dụng hàm switch...case

```
public static int TinhY(int x)
{
   y = 0;
   switch (x) {
      case 1: y += 4; break;
      case 2: y *= 2; break;
      default: y = y * y;
   }
   return y;
}
```

Chương trình 3.2: Tính y, sử dụng hàm If...else

```
public static int TinhY(int x)
{
   y = 0;
   if (x == 1)
      y += 4;
   else if (x == 2)
      y *= 2;
   else
      y = y * y;
   return y;
}
```

Ví dụ trên cho chúng ta thấy 2 chương trình là tương đường nhau về hành vi. Hai chương trình có giá trị đầu vào là như nhau (cùng kiểu **Int**). Chương trình đầu tiên sử dụng hàm ***switch...case***, chương trình tiếp theo sử dụng hàm ***if...else*** để kiểm tra giá trị đầu vào, tuy cú pháp khách nhau nhưng cách xử lý và trả về kết quả ***y*** là như nhau.

3.1.3 Sự khác biệt về hành vi (Behavioral Difference)

Hai chương trình P_1 và P_2 có cùng một miền các giá trị đầu vào I và khác biệt về hành vi nếu exec(P_1, I) $\neq$ exec(P_2, I), $\nexists i \in I$ exec(P_1, i) $\neq$ exec(P_2, i).

3.1.4 Tương tự hành vi (Behavioral Similarity)

Tương tự hành vi giữa hai chương trình P_1 và P_2 khi hai chương trình cùng miền giá trị đầu vào là tập hợp các giá trị $|I_s| \diagup |I|$, trong đó $I_s \subseteq I$, nếu exec(P_1, I_s) = exec(P_2, I_s), và $\forall j \in I \setminus I_s$, exec($P_1$; j) $\neq$ exec(P_2; j).

3.2 Một số phép đo độ tương tự hành vi dựa trên testcase

Để tính sự tương đồng về hành vi giữa hai chương trình, chúng ta có thể đo bằng cách tính tỷ lệ đầu vào và đầu ra trên cả hai chương trình trên cùng toàn bộ miền giá trị đầu vào. Một phương pháp tiếp cận đó là liệt kê tất cả các dữ liệu trong miền đầu vào và chạy từng đầu vào trên cả hai chương trình để so sánh các kết quả đầu ra. Nhưng việc này sẽ không thực tế hoặc không liên quan đến các chương trình với một miền đầu vào lớn hoặc vô hạn.

Để đo độ tương tự hành vi được chính xác hơn, chúng tôi chạy các dữ liệu đầu vào đại diện thay vì tất cả các dữ liệu đầu vào cho các chương trình. Bằng cách thống nhất lấy mẫu một phần dữ liệu đầu vào từ miền đầu vào, độ tương tự về hành vi sẽ được tính dựa trên tỷ lệ các mẫu đầu vào trên các đầu ra.

Dựa trên kỹ thuật Dynamic Symbolic Execution **(DSE)**, chúng ta có thể tạo ra được dữ liệu đầu vào thử nghiệm có độ bao phủ cao, và sử dụng trong các kỹ thuật đo độ tương tự.

3.2.1 Phép đo Random Sampling (RS)

Định nghĩa

Hai chương trình P_1 và P_2 là hai chương trình có cùng miền giá trị đầu vào I và I_s là tập con các giá trị đầu vào của tập I, và I_a là tập con các giá trị đầu vào của tập I_s, trong đó $\forall i \in I_a$, sao cho exec(P_1, i) = exec(P_2, i) và $\forall j \in I_s \backslash I_a$, exec($P_1$, j) $\neq$ exec(P_2, j). Độ đo của kỹ thuật của RS sẽ là $M_{RS}(P_1, P_2) = |I_a| / |I_s|$

Phép đo **RS** là một phương pháp đo tương đối đơn giản và hiệu quả để tính độ tương tự hành vi. Nhưng phép đo **RS** cũng có hạn chế nhất định trong những trường hợp có những hành vi nhỏ giữa các chương trình và phép đo **RS** không thể phân biệt các hành vi hơi khác nhau này.

Ví dụ:

Chương trình 3.3: Chương trình P_1

```
public static string Puzzle(string x) {
   if (x == "Hello") return "NOT OK";
   if (x == "Funny") return "NOT OK";
return "OK";
}
```

Chương trình 3.4: Chương trình P_2

```
public static string Puzzle(string x) {
   return "OK";
}
```

Trong ví dụ trên, chương trình P_1 và P_2 có hành vi khác biệt nhỏ nhưng độ đo **RS** không thể phân biệt được sự khác biệt này và trả về độ đo bằng 1.

3.2.2 Phép đo Single Program Symbol Execution (SSE)

Định nghĩa

Hai chương trình P_1 và P_2 là hai chương trình có cùng miền giá trị đầu vào I và P_1 là chương trình tham chiếu. Trong đó, I_s là tập các giá trị đầu vào được tạo bởi DSE trên P_1, và I_a là tập con các giá trị đầu vào của tập I_s, sao cho $\forall i \in I_a$, exec(P_1, i) = exec(P_2, i) và $\forall j \in I_s \backslash I_a$, exec($P_1$, j) $\neq$ exec(P_2, j). Độ đo của kỹ thuật của SSE sẽ là $M_{SSE}(P_1, P_2) = |I_a| / |I_s|$.

3.2.3 Kỹ Thuật Paired Program Symbolic Execution (PSE)

Định nghĩa

Hai chương trình P_1 và P_2 là hai chương trình có cùng miền giá trị đầu vào I. Chúng ta có chương trình P_3 là chương trình kết hợp của P_1 và P_2 có dạng (exec(P_1, I) = exec(P_2, I)), thõa điều kiện đầu vào và khẳng định điều kiện là đúng, exec(P_3, i) = T với giá trị đầu vào i trên P_3 là thõa điều kiện. Trong đó, I_s là tập các giá trị đầu vào được tạo bởi DSE trên P_3, và I_a là tập con các giá trị đầu vào của tập I_s, sao cho $\forall i \in I_a$, exec(P_3, i) = T và $\nexists\, j \in I_s \backslash I_a$, exec($P_3$, j) = T. Độ đo của kỹ thuật của PSE sẽ là $M_{PSE}(P_1, P_2) = |I_a| / |I_s|$.

Chương 4

CÀI ĐẶT CÁC PHÉP ĐO VÀ KẾT QUẢ THỰC NGHIỆM

4.1 Cài đặt các phép đo độ tương tự hành vi

4.2 Thực nghiệm và đánh giá kết quả các phép đo

KẾT LUẬN VÀ KIẾN NGHỊ

PHỤ LỤC

QUYẾT ĐỊNH GIAO LUẬN VĂN

Tài liệu tham khảo