BỘ GIÁO DỤC VÀ ĐÀO TẠO

ĐẠI HỌC ĐÀ NẴNG

NGUYỄN PHONG

NGHIÊN CỨU ỨNG DỤNG CÔNG CỤ SATAN ĐỂ PHÂN TÍCH KHẢ NĂNG KIỂM THỬ PHẦN MỀM CHO CÁC THIẾT KẾ TRÊN MÔI TRƯỜNG SIMULINK

Chuyên ngành : **KHOA HỌC MÁY TÍNH**

Mã số : **60.48.01**

TÓM TẮT LUẬN VĂN THẠC SĨ KỸ THUẬT

Đà Nẵng - Năm 2012

Công trình được hoàn thành tại

ĐẠI HỌC ĐÀ NẴNG

Người hướng dẫn khoa học: **TS. NGUYỄN THANH BÌNH**

Phản biện 1: **PGS.TS. VÕ TRUNG HÙNG**

Phản biện 2: **TS. NGUYỄN MẬU HÂN**

Luận văn đã được bảo vệ tại Hội đồng chấm Luận văn tốt nghiệp thạc sĩ kỹ thuật họp tại Đại học Đà Nẵng vào ngày 03 tháng 03 năm 2012

Có thể tìm hiểu luận văn tại:

- Trung tâm Thông tin - Học liệu, Đại học Đà Nẵng
- Trung tâm Học liệu, Đại học Đà Nẵng

MỞ ĐẦU

1. Lý do chọn đề tài

Ngày nay, các sản phẩm phần mềm xuất hiện và giữ vai trò quan trọng trong nhiều lĩnh vực của cuộc sống. Phần mềm trở nên cần thiết trong một số lĩnh vực công nghiệp như kỹ thuật điện tử hàng không, vận tải,… Để có thể sử dụng trong những lĩnh vực trên, phần mềm cần phải đạt được một số tiêu chuẩn đánh giá về độ tin cậy. Kiểm thử là một trong những kỹ thuật quan trọng nhằm đảm bảo chất lượng của sản phẩm phần mềm. Chi phí cho giai đoạn kiểm thử thường rất lớn trong qui trình phát triển phần mềm. Các kỹ thuật giúp cải thiện chất lượng của thiết kế phần mềm ngay từ giai đoạn thiết kế sẽ góp phần làm giảm chi phí kiểm thử sau này.

Phân tích khả năng kiểm thử được xem là yếu tố quan trọng trong đánh giá chất lượng phần mềm. Một phần mềm với khả năng kiểm thử cao thì có thể được kiểm thử dễ dàng. Mục đích của phân tích khả năng kiểm thử là đo lường phần mềm trong các giai đoạn sớm của quy trình phát triển phần mềm nhằm giúp cho các thiết kế viên cải thiện chất lượng của thiết kế phần mềm và các kiểm thử viên phân phối nguồn tài nguyên tốt hơn trong quá trình kiểm thử. Từ đó đề xuất các giải pháp để đảm bảo kiểm thử phần mềm tốt hơn.

Để phát triển một số hệ thống phần mềm công nghiệp phức tạp, nhiều môi trường, thường được thiết kế mô phỏng trước người ta sử dụng các chương trình mô phỏng như Simulink, Scade, Scicos,…Simulink [3] là phần chương trình mở rộng của Matlab đã được sử dụng rộng rãi trong công nghệ thông tin và ngành công

nghiệp cho mô hình hóa và mô phỏng hệ thống. Với Simulink, có thể mô phỏng các mô hình tuyến tính, các mô hình phi tuyến tính, phân tích các thừa số trong ma sát, sức cản không khí, sự dừng vật cứng và những mô phỏng khác để mô tả hiện tượng trong thế giới thực. Máy tính trở thành một phòng thí nghiệm ảo cho mô hình và hệ thống phân tích mà chúng ta có thể không thực hiện được trong thực tế hoặc gây nguy hiểm, các hành vi của một hệ thống tự động,... Simulink là môi trường thực hành của các kỹ sư bằng cách sử dụng nó để xây dựng mô hình và giải quyết các vấn đề thực sự… Các thiết kế trên môi trường SIMULINK yêu cầu nhiều hoạt động kiểm thử là quan trọng, vì vậy việc đánh giá khả năng kiểm thử chúng rất có giá trị.

Đó là lý do mà tôi chọn nghiên cứu và thực hiện đề tài "***Nghiên cứu ứng dụng công cụ SATAN*** [1] ***để phân tích khả năng kiểm thử phần mềm cho các thiết kế trên môi trường Simulink***". dưới sự hướng dẫn của thầy giáo TS. Nguyễn Thanh Bình.

Đề tài là một phần của đề tài: "Nghiên cứu kỹ thuật phân tích khả năng kiểm thử phần mềm và mở rộng tính năng công cụ SATAN, thử nghiệm trong môi trường SCICOS và SIMULINK, Cấp nhà nước – Nghị định thư, 2010-2011"

---

[1] SATAN: System's Automatic Testability Analysis (Hệ thống phân tích khả năng kiểm thử tự động)

2. Mục tiêu và nhiệm vụ nghiên cứu

Mục tiêu của đề tài là nghiên cứu môi trường lập trình và tạo các mô phỏng trong MATLAB SIMULINK, công cụ SATAN, lý thuyết về kiểm thử và phân tích khả năng kiểm thử phần mềm.

Qua môi trường SIMULINK thiết kế các mô hình, các luồng dữ liệu, chuyển các thiết kế Simulink sang MACDOT để làm dữ liệu đầu vào cho công cụ SATAN, nhận kết quả đầu ra của SATAN từ đó có những nhận định về khả năng kiểm thử.

Đề tài tập trung nghiên cứu các giải pháp phân tích khả năng kiểm thử cho các thiết kế trong môi trường Simulink.

3. Đối tượng và phạm vi nghiên cứu

- Nghiên cứu lý thuyết cơ bản về kiểm thử phần mềm.
- Phân tích khả năng kiểm thử phần mềm.
- Môi trường SIMULINK
- Công cụ SATAN
- Giải pháp phân tích khả năng kiểm thử các thiết kế trong môi trường SIMULINK.

Đề tài thuộc loại hình nghiên cứu.

4. Những phương tiện công cụ để có thể triển khai

Phần mềm MATLAB SIMULINK và SATAN

5. Phương pháp nghiên cứu

- Thu thập, phân tích các tài liệu và thông tin liên quan đến đề tài.
- Thảo luận, lựa chọn phương hướng giải quyết vấn đề.
- Phân tích thiết kế các mô phỏng SIMULINK.

- Công cụ SATAN
- Kiểm tra, thử nghiệm và đánh giá kết quả.

6. Ý nghĩa khoa học và thực tiễn

Đề tài có thể làm tài liệu tham khảo cho việc phân tích khả năng kiểm thử phần mềm.

Phần nghiên cứu lý thuyết sẽ cung cấp một cách nhìn tổng quát về môi trường SIMULINK, công cụ SATAN và khả năng kiểm thử của nó cho các thiết kế trong môi trường SIMULINK.

7. Đặt tên đề tài

"Nghiên cứu ứng dụng công cụ SATAN để phân tích khả năng kiểm thử phần mềm cho các thiết kế trên môi trường Simulink".

8. Bố cục luận văn

Nội dung chính luận văn được chia làm 4 chương:

MỞ ĐẦU

Giới thiệu lý do chọn đề tài, mục tiêu nghiên cứu, đối tượng và phạm vi nghiên cứu cũng như ý nghĩa của đề tài.

CHƯƠNG 1 - Tổng quan về phân tích khả năng kiểm thử phần mềm

Giới thiệu các khái niệm về kiểm thử và phân tích khả năng kiểm thử, các phương pháp phân tích khả năng kiểm thử phần mềm.

CHƯƠNG 2 – Môi trường SIMULINK.

Giới thiệu về môi trường Simulink và các thư viện hỗ trợ bởi Simulink. Từ đó, báo cáo cũng trình bày các sử dụng các khối trong các thư viện để xây dựng các hệ thống và thực hiện mô phỏng.

CHƯƠNG 3 – Phương pháp phân tích khả năng kiểm thử dựa trên công cụ SATAN.

Trình bày kiến trúc tổng thể của công cụ SATAN. Phân tích khả năng kiểm thử dựa trên đồ thị truyền tin ITC sử dụng công cụ SATAN.

CHƯƠNG 4 - Giải pháp và ứng dụng phân tích khả năng kiểm thử các thiết kế trong môi trường Simulink

Cấu trúc của các mô đun quan trọng trong công cụ. Giới thiệu các ngữ pháp của các ngôn ngữ In-Mac và MacDot được sử dụng để định nghĩa dữ liệu đầu vào và dữ liệu xử lý trung gian của SATAN.

Trình bày một số ví dụ thử nghiệm trên công cụ SATAN

Chương 1
TỔNG QUAN VỀ PHÂN TÍCH KHẢ NĂNG KIỂM THỬ PHẦN MỀM

1.1. KIỂM THỬ PHẦN MỀM

1.1.1. Khái niệm

IEEE: "*Kiểm thử là tiến trình kiểm tra phần mềm hoặc các môđun của phần mềm với các điều kiện xác định, quan sát hoặc ghi nhận kết quả, cũng như đưa ra đánh giá về hệ thống hoặc thành phần đó*".

Myers: "*Kiểm thử phần mềm là quá trình kiểm tra một chương trình hoặc hệ thống với mục đích tìm ra các lỗi trong chương trình đó*".

1.1.2. Vai trò của hoạt động kiểm thử

1.1.3. Các khó khăn khi thực hiện kiểm thử phần mềm.

1.2. PHÂN TÍCH KHẢ NĂNG KIỂM THỬ PHẦN MỀM

1.2.1. Khái niệm

Tổ chức IEEE khái niệm PTKNKT phần mềm là xác định mức độ phần mềm phù hợp với các tiêu chuẩn kiểm thử đưa ra. Việc thực hiện kiểm thử nhằm quyết định liệu phần mềm hoặc thành phần đó có đáp ứng được các yêu cầu đặt ra không hoặc phần mềm được kiểm thử có phù hợp với các tiêu chuẩn không. PTKNKT phần mềm là hoạt động đo độ phức tạp để thoả mãn mục tiêu kiểm thử.

Freedman [19] khái niệm KNKT, bằng cách dựa trên khả năng quan sát (observability) và khả năng điều khiển (controllability). Phương pháp này đánh giá chương trình dựa trên miền dữ liệu vào và miền dữ liệu ra.

Voas và Miller định nghĩa PTKNKT phần mềm là dự đoán khả năng phần mềm bị sự cố (failure) do gặp lỗi khi kiểm thử với miền dữ liệu vào ngẫu nhiên. Dựa vào kết quả của hoạt động, người kiểm thử tìm phương pháp tốt nhất để cải tiến chất lượng, tăng KNKT của phần mềm. Cấu trúc và ngữ nghĩa của mã nguồn cùng với sự phân bố miền dữ liệu vào đã quyết định KNKT phần mềm.

1.2.2. Các phương pháp PTKNKT phần mềm

1.2.2.1. Phương pháp phân tích kiểm thử dựa trên thiết kế

a) Phương pháp PTKNKT miền (của Freedman)

Freedman [19] cho rằng KNKT là các tính chất của một chương trình dễ dàng được kiểm thử. Một đơn vị phần mềm có

KNKT cao cần có các tính chất: Tập dữ liệu kiểm thử có kích thước nhỏ và dễ dàng được tạo ra. Tập dữ liệu kiểm thử không dư thừa. Kết quả kiểm thử dễ dàng hiểu được. Các lỗi dễ dàng định vị được.

b) Phương pháp PTKNKT của Voas và Miller

Jeffrey Voas[13] KNKT phần mềm là dự đoán khả năng phần mềm bị sự cố (failure) do gặp lỗi khi kiểm thử với miền dữ liệu vào ngẫu nhiên. Sự cố có thể tồn tại ở mọi nơi trong chương trình. Do đó, lỗi có thể xuất hiện ở những nơi có khả năng gây ra sự cố trong bất kỳ phương thức nào.

c) Phương pháp PTKNKT cho phần mềm giao tiếp (của Kharoui)

Mô hình quan hệ là các mô hình cho phép mô tả các đặc tả phần mềm. Mô hình này phân tích các đặc tính của phần mềm một cách độc lập mà không có bất kỳ ràng buộc nào trên các đặc tả của chính phần mềm đó. Mô hình này có thể sử dụng ở các mức độ trừu tượng khác nhau. Karoui [10] sử dụng các đặc tả quan hệ để đánh giá KNKT phần mềm.

1.2.2.2. KNKT dựa trên mã nguồn

a) Độ đo McCabe

Độ đo McCabe [24] dựa trên việc đánh giá số các chu trình độc lập trong một đồ thị luồng điều khiển biểu diễn một mô đun. Độ đo của McCabe được xem là phương pháp đánh giá độ phức tạp được sử dụng rộng rãi nhất. Độ đo McCabe được xem là một trong những kỹ thuật đánh giá KNKT.

b)Kỹ thuật của Yin và Bieman

Kỹ thuật chèn và kiểm tra các xác nhận (assertion) khi thực thi mã nguồn là một trong những kỹ thuật cho phép định vị lỗi và đảm bảo mã nguồn thoả mãn những ràng buộc xác định. Các tác giả Yin and Bieman đề xuất áp dụng kỹ thuật chèn xác nhận và xây dựng công cụ hỗ trợ cho ngôn ngữ lập trình C [6].

Kỹ thuật này được ứng dụng trong nhiều dự án công nghiệp, kết quả cho thấy nhiều lỗi được phát hiện, chất lượng mã nguồn tốt hơn.

c) Kỹ thuật PIE

Kỹ thuật PIE phân tích ba phần đối với mỗi vị trí trong chương trình nhằm đánh giá khả năng phát sinh lỗi: gồm phân tích sự thực thi, phân tích các ảnh hưởng và phân tích sự lan truyền. Kỹ thuật PIE yêu cầu phải thực thi của mã nguồn. Các dữ liệu đầu vào được lựa chọn ngẫu nhiên từ miền dữ liệu vào. Phân tích PIE là quá trình phân tích động, tuy nhiên nó không phải là quá trình kiểm thử phần mềm vì không dữ liệu đầu ra nào được kiểm tra so với đặc tả.

1.2.2.3. KNKT các hệ thống hướng đối tượng

a) Công trình của Chidamber và Kemerer

Chidamber và Kemerer [21] cho rằng quá trình thiết kế hướng đối tượng có 4 bước cơ bản: xác định các lớp, xác định ngữ nghĩa của các lớp, xác định mối quan hệ giữa các lớp và thiết kế các hàm thực thi liên quan đến các lớp đó. Trong nghiên cứu này, các tác giả đề xuất 6 đo độ của mô hình lớp.

Các tác giả đã lập luận cơ sở lý thuyết của các độ đo, xây dựng các tiêu chí đánh giá các độ đo. Kết quả thu được cho thấy rất có ý nghĩa đối với các hoạt động phát triển, như kiểm thử, gỡ rối và bảo trì.

b) Công trình của Payne, Alexander và Hutchinson

Payne [8] và các đồng tác giả đề xuất phương pháp thiết kế sử dụng các *hợp đồng phần mềm* (software contract) trong đặc tả và chèn các xác nhận (assertion) vào mã nguồn khi cài đặt để nâng cao khả năng quan sát (observability) và khả năng điều khiển (controllability), nghĩa là nâng cao KNKT.

c) Công trình của Jungmayr

Jungmayr[22] xây dựng các độ đo về KNKT dựa trên sự phụ thuộc của các thành phần (component) phần mềm hướng đối tượng. Phần mềm được mô tả bởi *đồ thị phụ thuộc* (dependency graph), gồm tập hợp các thành phần và sự phụ thuộc giữa chúng.

Jungmayr xác định các mục tiêu của việc cải tiến KNKT. Sau đó, các độ đo KNKT được định nghĩa nhằm đo lường mức độ đạt được các mục tiêu đặt ra.

Hai mục tiêu cải tiến KNKT gồm: Mục tiêu 1kiểm thử một cách độc lập nhất có thể các thành phần. Mục tiêu 2 hạn chế chi phí kiểm thử hồi quy.

Từ đó, tác giả đã định nghĩa một số độ đo KNKT

d) Công trình của Baudry, Le Traon và Sunyé [4]

Các tác giả cho rằng KNKT là sự dễ dàng kiểm thử phần mềm. KNKT được đánh giá bởi ba yếu tố:

- Chi phí kiểm thử: gồm chi phí để tạo bộ dữ liệu thử đạt tiêu chí đặt ra và chi phí xác định sự hợp lệ của kết quả kiểm thử.
- Khả năng điều khiển: sự dễ dàng tạo ra dữ liệu thử.
- Khả năng quan sát: sự dễ dàng kiểm tra sự hợp lệ của kết quả kiểm thử.

e) Công trình của Kansomkeat, Offutt và Rivepiboon [23]

Các tác giả đã nghiên cứu phương pháp PTKNKT của lớp dựa trên việc phân tích luồng dữ liệu (data-flow analysis) bằng kỹ thuật phân tích đột biến (mutation analysis).

Chương 2
MÔI TRƯỜNG SIMULINK

2.1. SIMULINK MATLAB
2.2.TÍN HIỆU VÀ CÁC LOẠI DỮ LIỆU
2.2.1.Tín hiệu
2.2.2.Các loại dữ liệu
2.3. CÁC THƯ VIỆN SIMULINK
2.3.1.Thư viện Sources
2.3.2.Thư viện Sinks
2.3.3.Thư viện Math
2.3.4.Thư viện Continuous
2.3.4.Thư viện Discrete
2.3.5.Thư viện Signal & Systems
2.4. MÔ PHỎNG BẰNG SIMULINK

Việc mô phỏng bằng Simulink gồm các bước sau:

Bước 1: Mở cửa sổ mô phỏng Similink.

Bước 2: Rê và thả các khối cần xây dựng mô hình từ các thư viện SIMULINK.

Bước 3: Nhập thông số cho các khối.

Bước 4: Chú thích khối, chỉnh sửa, trang trí mô hình.

Bước 5: Khai báo các thông số của mô hình hoá.

Bước 6: Chạy chương trình, quan sát kết quả, đánh giá kết quả và hiệu chỉnh cho phù hợp sau mỗi lần chạy.

Chương 3
PHƯƠNG PHÁP PHÂN TÍCH KHẢ NĂNG KIỂM THỬ SỬ DỤNG CÔNG CỤ SATAN

3.1. PHÂN TÍCH KHẢ NĂNG KIỂM THỬ VỚI CÔNG CỤ SATAN

3.1.1. Đồ thị truyền tin

Mô hình KNKT là một đồ thị có hướng, được gọi là *đồ thị truyền tin*. Đồ thi truyền tin được định nghĩa bởi tập hợp các nút, tập hợp các chuyển tiếp và tập hợp các cung. Nút bao gồm các đầu vào, đầu ra và các mô đun.

Đồ thị truyền tin ITG được định nghĩa một cách hình thức như sau:

Định nghĩa 1. G = (Z, T, U) là một đồ thị, Z = S ∪ M ∪ P là tập hợp các nút, Z khác rỗng, S tập các đầu vào, P tập các đầu ra và M là tập các mô đun; T là tập hữu hạn không rỗng các chuyển tiếp; U là tập hữu hạn không rỗng các cung.

Đặt: $\Gamma_G^+(x)$ là tập hợp các nút đứng sau nút x

$\Gamma_G^-(x)$ là tập hợp các nút đứng trước nút x

$\Gamma_G(x)$ là tập hợp các nút kề nút x

Ta gọi G là ***một đồ thị truyền tin cơ sở*** nếu và chỉ nếu:

G là một đồ thị liên thông; (1.1)

$\forall t \in T$ $\Gamma_G^+(t) \neq \varnothing, \Gamma_G^-(t) \neq \varnothing;$ (1.2)

$\forall s \in S$ $\Gamma_G^-(s) \neq \varnothing;$ (1.3)

$\forall p \in P$ $\Gamma_G^+(p) \neq \varnothing;$ (1.4)

$\forall m \in M$ $\Gamma_G^+(m) \neq \varnothing, \Gamma_G^-(m) \neq \varnothing;$ (1.5)

$\forall t \in T$ $\Gamma_G(t) \subset Z;$ (1.6)

$\forall z \in Z$ $\Gamma_G(z) \subset T;$ (1.7)

$\forall t \in T$ $\Gamma_G^+(t) \cap \Gamma_G^-(t) = \varnothing;$ (1.8)

$\forall s \in S$ $\exists p \in P: \ s \overset{*}{\Rightarrow} p$ (1.9)

Ta gọi *S* là tập các đầu vào của ITG; *P* là tập hợp các đầu ra của ITG; *M* là tập hợp các mô đun của ITG.

Điều kiện (1.8) không cho phép sự tồn tại chu trình giữa một mô đun và một chuyển tiếp.

Điều kiện (1.9) bảo đảm tồn tại một đường đi từ một đầu vào đến một đầu ra.

Chúng ta gọi một đồ thị truyền tin ITG là sự hợp các đồ thị truyền tin cơ sở.

Một hệ thống có thể được mô hình hóa bởi nhiều đồ thị truyền tin. Các chuyển tiếp mô tả kiểu truyền tin giữa các nút. Trong đồ thị truyền tin.

Trong biểu diễn đồ họa của ITG, các đầu vào và đầu ra được biểu diễn bởi hình bán nguyệt, các mô đun bởi hình tròn, các chuyển tiếp bởi hình gạch ngang và các cung bởi hình mũi tên.

3.1.2. Luồng

Một luồng truyền tin từ một hoặc một vài đầu vào, đi qua một số mô đun và các chuyển tiếp, cuối cùng đi tới một đầu ra. Mỗi luồng có thể được xem là một chức năng cơ sở dùng để tính toán một đầu ra của hệ thống.

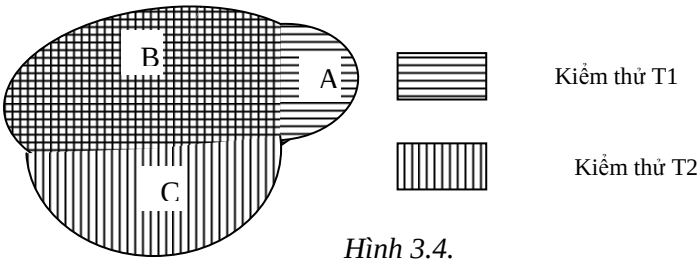
3.1.3. Chiến lược kiểm thử

Một chiến lược kiểm thử là một kế hoạch kiểm thử, xác định các chức năng của hệ thống hay các luồng của ITG cần được kích hoạt.

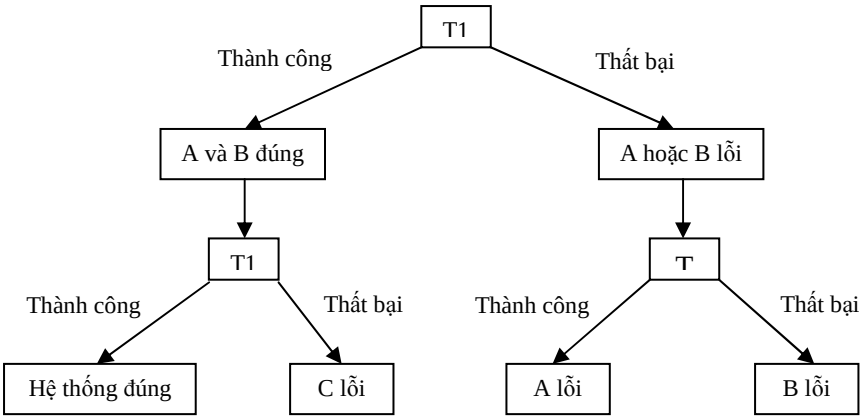
Có hai loại chiến lược kiểm thử:

Các chiến lược kết hợp (combinative strategies): nguyên tắc phân tích một chuỗi các ca kiểm thử để định vị lỗi. Có hai chiến lược kết hợp sau:

- **Chiến lược All-path**: trong chiến lược này, tất cả các luồng của ITG được kích hoạt.
- **Chiến lược Multiple Clue**: tập hợp các luồng được kích hoạt là tập hợp nhỏ nhất các luồng bao phủ tất cả các mô đun của ITG và đảm bảo xác định chính xác nhất các lỗi, như minh họa trong hình 3.4 và 3.5.



Hình 3.4.
Minh họa chiến lược Multiple Clue

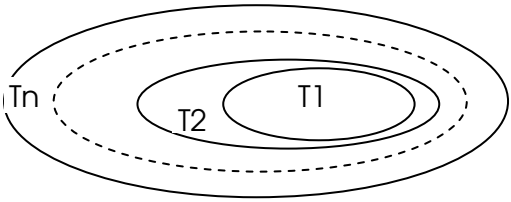


Hình 3.5. Phân tích kết quả kiểm thử khi áp dụng chiến lược Multiple Clue

- **Các chiến lược tăng dần** (incremental strategies): các chiến lược này tìm cách bao phủ dần dần tất cả các mô đun của đồ thị, như mình họa trong hình *3.6*. Khi tìm thấy một lỗi, phải sửa lỗi trước khi thực hiện ca kiểm thử tiếp theo. Các luồng cần kích hoạt

được sắp xếp bởi các lớp theo một số tiêu chí. Có thể phân biệt các chiến lược tăng dần theo tiêu chí kiểm thử:

- **Chiến lược Start-Small**: giảm tối thiểu chi phí chẩn đoán lỗi. Luồng đầu tiên được thực thi chứa một số ít nhất các mô đun. Chọn luồng được thi tiếp theo sao cho phủ được số tối thiểu các mô đun chưa được kích hoạt. Tiếp tục như vậy cho đến khi tất cả các mô đun được kích hoạt.
- **Chiến lược Big-Start**: bao phủ một cách nhanh nhất các mô đun của ITG. Luồng đầu tiên được thực thi chứa một số lớn nhất các mô đun. Chọn luồng được thi tiếp theo sao cho phủ được số lớn nhất các mô đun chưa được kích hoạt. Tiếp tục như vậy cho đến khi tất cả các mô đun được kích hoạt.



Hình 3.6. Chiến lược Start-Small

3.1.4. Độ đo KNKT

Độ đo KNKT được xây dựng dựa trên lý thuyết thông tin.

3.1.4.1. Lý thuyết thông tin

- Độ bất định
- Lượng thông tin (information quantity / entropy)

3.1.4.2. Mạng truyền tin

Để mô hình hóa sự truyền thông tin, mạng truyền tin (ITN) được sử dụng. ITN là một ITG có trọng số là các khả năng thông tin của các cung và các mô đun.

Định nghĩa 4. Gọi *G=(X,U)* là một đồ thị có hướng có *n* đỉnh. Giả sử tồn tại trong *X* hai đỉnh đặc biệt x_1 và x_n, sao cho $\Gamma_G^-(x_1) \neq \varnothing$ (x_1 được gọi là đỉnh vào) và $\Gamma_G^+(x_n) \neq \varnothing$ (x_n được gọi là đỉnh ra). Mỗi cung $(x_i, x_j) \in U$ được gán c_{ij} là khả năng thông tin của cung đó. Một đồ thị như thế được gọi là mạng truyền tin, được định nghĩa bởi bộ ba $T = (X, U, C)$ trong đó $U = \{C_{ij}, (x_i, x_j) \in U\}$.

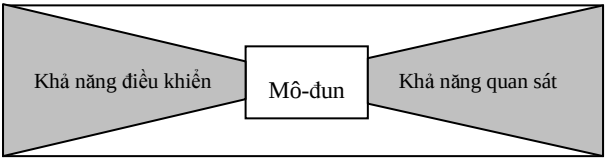
3.1.4.3. Độ đo KNKT

KNKT của hệ thống phụ thuộc vào KNKT của các mô đun của hệ thống. PTKNKT của hệ thống dựa vào PTKNKT của các mô đun của nó. KNKT của một mô đun được định nghĩa dựa trên khả năng điều khiển và khả năng quan sát.

Khả năng điều khiển của một mô đun trong hệ thống được định nghĩa như là khả năng mang các đầu ra của hệ thống đến các đầu ra của mô đun.

Khả năng quan sát của một mô đun trong hệ thống được định nghĩa là khả năng đưa các đầu ra của mô đun đến các đầu ra của hệ thống.

KNKT của một mô đun được định nghĩa bởi cặp giá trị: khả năng điều khiển và khả năng quan sát. Hình vẽ 3.7 minh họa độ đo KNKT của một mô đun trong hệ thống.

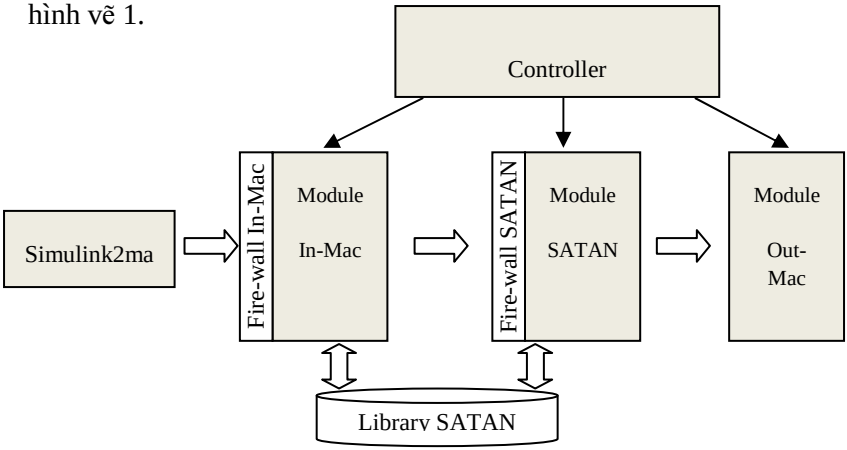


Hình 3.7. Khả năng điều khiển và khả năng quan sát mô đun

3.2. CÔNG CỤ SATAN

3.2.1. Kiến trúc tổng thể SATAN

Kiến trúc tổng thể của công cụ SATAN được trình bày trong hình vẽ 1.



Hình 3.8. Kiến trúc tổng thể công cụ SATAN

Bộ điều khiển (controller): dùng để kích hoạt công cụ SATAN.

Mô đun Simulink2mac: dùng để chuyển các thiết kế trong môi trường SIMULINK

Mô đun In-Mac: xử lý các đầu vào của SATAN. Bức tường lửa In-Mac kiểm tra cú pháp của các đầu vào.

Mô đun lõi SATAN: xử lý chính các bước PTKNKT và độ đo KNKT.

Mô đun Out-Mac: xử lý đầu ra của SATAN.

Thư viện SATAN.

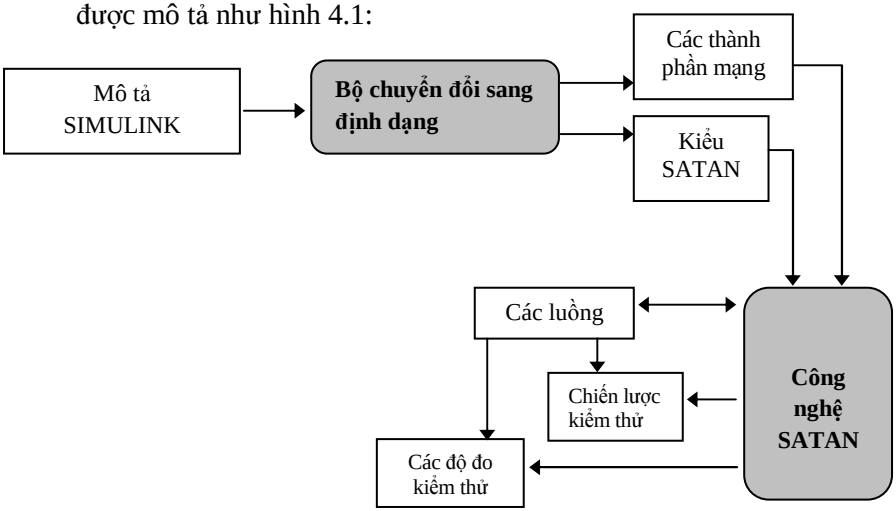
3.2.2. Mô đun In-Mac

3.2.3. Mô đun lõi SATAN

Chươn/g 4
GIẢI PHÁP VÀ ỨNG DỤNG PHÂN TÍCH KHẢ NĂNG KIỂM THỬ CÁC THIẾT KẾ TRONG MÔI TRƯỜNG SIMULINK

4.1.GIẢI PHÁP TỔNG THỂ

Vai trò của bộ chuyển đổi định dạng trong tiến trình PTKNKT được mô tả như hình 4.1:



Hình 4.1 – Vai trò của bộ chuyển đổi

Các mô hình được xây dựng trong môi trường SIMULINK. Sau đó, chúng được được bộ chuyển đổi dịch thành hai phần:

Các thành phần mạng: là các thành phần cấu thành nên đồ thị ITG và ITN

Kiểu SATAN: Các kiểu dữ liệu trong SIMULINK được chuyển thành các kiểu tương ứng được định nghĩa bởi công nghệ SATAN

4.2.GIẢI PHÁP CHI TIẾT

4.2.1.Bộ chuyển đổi

Sau đây, là giải pháp chi tiết cho bộ chuyển đổi:

Algorithm Simulink2mac(Simulink system)
Begin
Tìm tất cả thành phần con của mô hình
Dịch chúng sang định dạng MACDOT
Tìm tất cả các cạnh của mô hình
Dịch chúng sang định dạng MACDOT
Lấy tất cả các hệ thống con của mô hình
Dịch các hệ thống con sang định dạng MACDOT bằng các lời gọi hàm đệ qui
End

Để có thể đọc dữ liệu của các thành phần con, các cạnh và các hệ thống con, sử dụng các APIs được cung cấp bởi môi trường SIMULINK.

4.2.2.Các hàm chính được sử dụng trong bộ chuyển đổi

4.2.3.Các APIs được sử dụng trong bộ chuyển đổi

4.2.4.Thư viện InMac cho SIMULINK

4.2.4.1.Mối tương quan giữa môi trường SIMULINK và SATAN

4.2.4.2.Tạo đầu vào SATAN từ mô hình SIMULINK

4.2.4.3.Đổi tên trong công nghệ SATAN

4.2.4.4.Mô đun SIMULINK tương ứng trong SATAN

4.2.4.5.Tham số đầu vào, đầu ra của mô đun SIMULINK tương ứng với mô đun SATAN

4.2.4.6.Hằng số SIMULINK tương ứng hằng số SATAN

4.2.4.7. Đặt tên cho mô đun trong SATAN

4.2.5.Tạo kiểu SATAN

4.2.5.1.Tên kiểu SATAN

4.3.5.2.Danh sách đầu vào

4.2.5.3.Danh sách đầu ra

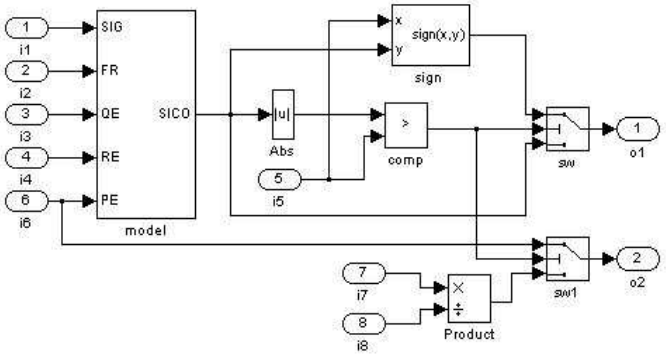
4.2.5.4.Mô hình chung

4.3. KẾT QUẢ THỬ NGHIỆM

Giải pháp đã PTKNKT trên một số mô hình SIMULINK của phòng thí nghiệm DATIC, Trường Đại học Bách khoa, Đại học Đà Nẵng và phòng thí nghiệm LCIS, Viện Đại học Bách Khoa Grenoble, Cộng hòa Pháp cung cấp và thu được một số kết quả

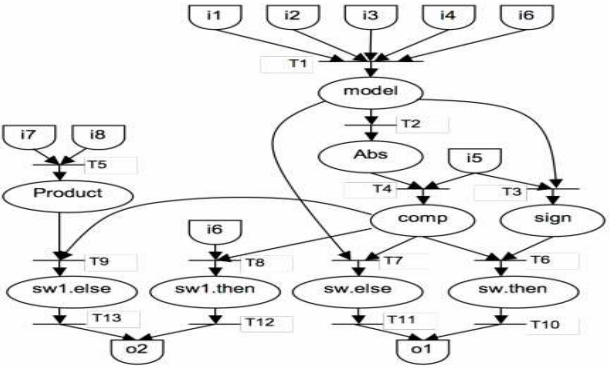
4.3.1. Hệ thống RG

Hệ thống RG được thiết kế trong môi trường Simulink để minh họa kết quả nghiên cứu. Hệ thống RG nhằm điều khiển quỹ đạo bay của máy bay.



Hình 4.2. Hệ thống RG.

Đồ thị truyền tin của hệ thống RG được xây dựng như trong Hình 4.3



Hình 4.3. Đồ thị truyền tin của hệ thống RG.

Đồ thị truyền tin của hệ thống RG có bốn luồng. Mỗi luồng được mô tả bởi các mô-đun và các đầu ra:

F1 = {model, abs, comp, sign, sw.then | o1},

F2 = {model, abs, comp, sw.else | o1},

F3 = {model, abs, comp, sw1.then | o2},

F4 = {model, abs, comp, product, sw1.else | o2}.

Khi tất cả các luồng của đồ thị truyền tin được xác định, một chiến lược kiểm thử được sử dụng để xác định các mục tiêu kiểm thử của phần mềm.

Khi áp dụng chiến lược Start-Small, tất cả các luồng đều được sử dụng. Sau đó, tính toán các độ đo KNKT, chúng ta thu được kết quả trình bày trong Bảng 4.7.

Bảng 4.7. Kết quả phân tích hệ thống RG

Mô-đun	**KNĐK**	**KNQS**
Model	1.0	0.064
Sign	0.533	0.082
sw1	0.999	1.0
Sw	0.999	1.0
Product	1.0	0.082
Comp	0.533	1.0
Abs	0.533	0.064

Từ kết quả này, chúng ta có thể nhận thấy hai mô-đun *sw* và *sw1* có đồng thời khả năng điều khiển và khả năng quan sát rất cao, như thế hai mô-đun này rất dễ dàng được kiểm thử. Hai mô-đun *model* và *product* có khả năng điều khiển rất cao, nhưng có khả năng

quan sát rất thấp, hai mô-đun này rất dễ dàng điều khiển, nhưng lại khó quan sát khi kiểm thử. Hai mô-đun *sign* và *abs* có khả năng điều khiển trung bình và khả năng quan sát rất thấp, như thế kiểm thử hai mô-đun này có thể gặp nhiều khó khăn.

4.3.2.Hệ thống B23_NAVIGATION_PROPAGATION

4.3.3.Hệ thống re_gy_2sw_anno

4.3.4.Hệ thống S_TEST_QUATERNION_SIGN

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1.Kết luận

Kết quả đạt được

Đã tìm hiểu về quá trình PTKNKT phần mềm. Nắm được một số phương pháp PTKNKT phần mềm.

Tích hợp tiến trình PTKNKT các mô hình luồng dữ liệu vào môi trường phát triển MATLAB Simulink. Việc tích hợp này cho phép KNKT được xem xét sớm trong giai đoạn thiết kế.

Kết quả PTKNKT cũng có thể giúp cho kiểm thử viên lập kế hoạch, lựa chọn kỹ thuật và phân bổ chi phí cho kiểm thử hợp lý hơn, nghĩa là chú trọng hơn các thành phần khó kiểm thử. KNKT còn được xem là tiêu chí đánh giá chất lượng từ đó giúp cho người dùng đánh giá độ tin cậy của phần mềm.

Đã ứng dụng công cụ SATAN vào việc PTKNKT cho một số mô hình cụ thể được thiết kế trên môi trường SIMULINK MATLAB.

Luận văn có thể làm tài liệu tham khảo cho các tổ chức cá nhân hoạt động trong lĩnh vực kiểm thử, góp một phần nhỏ cho việc hoàn thiện đề tài: "Nghiên cứu kỹ thuật PTKNKT phần mềm và mở rộng tính năng công cụ SATAN, thử nghiệm trong môi trường SCICOS và SIMULINK, Cấp nhà nước – Nghị định thư, 2010-2011"

Hạn chế

Phần lý thuyết còn mang tính nghiên cứu tổng quan, mới nhấn mạnh vào nghiên cứu phương pháp PTKNKT dựa trên công cụ SATAN.

Cộng cụ SATAN là công cụ duy nhất có thể PTKNKT phần mêm chưa có một công cụ nào khác tương đương nên việc so sánh những kết quả đánh giá của công cụ SATAN gặp nhiều khó khăn.

Kết quả thử nghiệm chỉ mới dừng lại nghiên cứu một số mô hình bài toán của phòng thí nghiệm DATIC, Trường Đại học Bách khoa, Đại học Đà Nẵng và phòng thí nghiệm LCIS, Viện Đại học Bách Khoa Grenoble, Cộng hòa Pháp.

2. Hướng phát triển

Tiếp tục nghiên cứu KNKT của các mô hình luồng điều khiển, như các mô hình Stateflow trong môi trường Simulink.

Nghiên cứu kết hợp nhiều phương pháp phần tích KNKT từ đó đề xuất phương án thiết kế phần mềm tối ưu, giảm thiểu các chi phí và tổn thất không cần thiết.