Tạp chí Khoa học Trường Đại học Cần Thơ
website: sj.ctu.edu.vn

TRỪU TƯỢNG HÓA DỰA TRÊN THÀNH PHẦN ĐỂ KIỂM TRA TẮC NGHẼN TRÊN MẠNG CẢM ỨNG KHÔNG DÂY SỬ DỤNG MẠNG PETRI

Lê Ngọc Kim Khánh[1,2], Lê Quốc Vũ[2], Tân Quốc Tiến[2], Bùi Hoài Thắng[1] và Quản Thành Thơ[1]

[1] *Khoa Khoa học và Kỹ thuật máy tính, Đại học Bách Khoa, Đại học Quốc gia, Thành phố Hồ Chí Minh*
[2] *Khoa Công nghệ Thông tin, Đại học Sài Gòn, Thành phố Hồ Chí Minh*

Thông tin chung:
Ngày nhận: 19/09/2015
Ngày chấp nhận: 10/10/2015

Title:
Applying component-based abstraction to verify congestion on wireless sensor network by using Petri net

Từ khóa:
Mạng cảm ứng không dây, Phát hiện tắc nghẽn, Kỹ thuật trừu tượng hóa, mạng Petri

Keywords:
Wireless Sensor Network, Congestion Detection, Abstraction tecchnique, Petri net

ABSTRACT

Congestion detection and mitigation on wireless sensor networks has widely been pursued by the research community. Generally, wireless sensor network is deployed in dense or spare mode which has a specified strategy to detect and avoid congestion. However, the cost of implementation a wireless network is too huge so that the congestion verification should be done on the simulation before deployment in reality.

We address this problem by model checking approach. Wireless sensor network is modeled by Petri nets and their properties are verified by model checking. Moreover, we use the component-based abstraction technique to abstract the unused components when verifying congestion. Our ideas have fully been implemented as a tool known as WSN-PN, on which the experimental results proved the significant improvement on verification process.

TÓM TẮT

Phát hiện và xử lý tắc nghẽn trên mạng cảm ứng không dây là một vấn đề thu hút nhiều sự chú ý trong cộng đồng nghiên cứu. Về cơ bản, mạng cảm ứng không dây có thể triển khai ở chế độ mạng dày hoặc mạng thưa; mỗi chế độ này đòi hỏi một chiến lược tiếp cận khác nhau để phát hiện và xử lý nghẽn. Do chi phí triển khai một mạng cảm ứng không dây là khá lớn, vấn đề kiểm tra sự tắc nghẽn trên mạng cần được kiểm tra trên một mô hình máy tính trước khi triển khai thực tế.

Chúng tôi giải quyết vấn đề này bằng cách sử dụng hướng tiếp cận kiểm tra mô hình. Ngôn ngữ mạng Petri được sử dụng để biểu diễn và kiểm tra các tính chất của một mạng không dây. Hơn thế nữa, chúng tôi còn sử dụng kỹ thuật trừu tượng hóa dựa trên thành phần để có thể trừu tượng hóa các thành phần không cần thiết trên một mạng cảm ứng không dây khi cần kiểm tra tắc nghẽn theo các chế độ khác nhau. Nhờ đó, tốc độ xử lý của chúng tôi được tăng lên rất nhiều. Chúng tôi đã phát triển công cụ WSN-PN để hiện thực ý tưởng này. Kết quả thí nghiệm cho thấy WSN-PN có thể thu giảm đáng kể không gian trạng thái cần kiểm tra khi sử dụng trừu tượng hóa.

1 GIỚI THIỆU

Petri net (PN) [1] là ngôn ngữ toán học dùng để mô hình và kiểm tra các hệ thống tập trung rất hiệu quả. Về cơ bản, PN là một đồ thị hai phía có hướng bao gồm tập hợp hữu hạn các *vị trí* (places) và *chuyển tiếp* (transition). Trong mỗi vị trí sẽ chứa những *thẻ đánh dấu* (token) dùng để điều khiển việc chuyển dữ liệu trên mạng. Khi một vị trí có đủ số lượng thẻ đánh dấu cần thiết thì các thẻ đánh dấu đó sẽ được chuyển qua vị trí khác thông qua chuyển tiếp, hành động này được gọi là *fireable*. Mạng PN được sử dụng rộng rãi trong các nghiên cứu và thậm chí là trong công nghiệp. Có rất nhiều công cụ đã và đang được nghiên cứu giúp người sử dụng mô tả và kiểm tra các mô hình PN như Lola [2] hay Snoopy [3].

Khi hệ thống được mô hình hóa bằng PN, các thuộc tính của hệ thống này có thể kiểm tra bằng cách sử dụng *model checking*(MC). Tuy nhiên, nhược điểm lớn nhất của hướng tiếp cận này là sự bùng nổ không gian trạng thái. Hiện nay, có hai phương pháp được đề xuất để giải quyết vấn đề này: *giảm số lượng các không gian trạng thái một cách trực tiếp* hay *sử dụng phương pháp trừu tượng hóa mô hình PN*. Trong nghiên cứu này, chúng tôi tập trung vào cách tiếp cận thứ hai.

Việc áp dụng phương pháp trừu tượng hóa mô hình PN thường được thực hiện trên từng thành phần của mô hình [4]. Một mô hình gốc ban đầu sẽ được chia ra thành nhiều mô hình con, chúng ta tiến hành kiểm tra trên các mô hình con trước, từ đó rút ra kết luận cho mô hình gốc.

Tuy nhiên, phương pháp này lại có một số nhược điểm. Cụ thể là, khi chúng ta mô hình hóa một hệ thống thực, thì các hành động ngữ nghĩa của hệ thống, thường được định nghĩa bởi các chuyên gia, phải được giữ lại và chuyển thành các luật ngữ nghĩa trong mô hình PN tương ứng. Vì thế, khi trừu tượng hóa hệ thống trên, chúng ta phải luôn đảm bảo rằng các luật ngữ nghĩa vẫn có thể hoạt động nguyên vẹn. Hơn nữa, với các hệ thống phức tạp trong thực tế, số lượng các hệ thống con là rất lớn. Hơn thế nữa, đặc điểm của các thành phần (kiểu dữ liệu) lại rất khác nhau. Từ mục đích trên, chúng tôi đề xuất khái niệm mô hình *trừu tượng hóa dựa trên các thành phần* của PN (Component-based Petri net). Đầu tiên, mạng PN được hình thành bằng cách kết hợp nhiều thành phần lại với nhau. Mỗi thành phần đó được gọi là *component PN*. Khi chúng ta trừu tượng hóa mô hình thì các thành phần sẽ được trừu tượng theo. Vì thế, làm việc trên mô hình chính hay mô hình trừu tượng hoá là như nhau vì chúng ta chỉ trừu tượng các đặc điểm bên trong thành phần còn các môi liên kết giữa các thành phần thì vẫn được bảo tồn. Cũng chính vì thế, các luật ngữ nghĩa trên mô hình chính và mô hình trừu tượng cũng không đổi.

Trong bài báo này, chúng tôi sẽ chứng minh tính hiệu quả của việc kiểm tra trên mô hình trừu tượng so với mô hình gốc với bài toán phát hiện tắc nghẽn trên mạng cảm ứng không dây.

Một mạng cảm ứng không dây hay còn gọi là mạng cảm biến không dây sẽ được chia thành hai thành phần con là *cảm biến* (sensor) và *kênh truyền* (channel). Đầu tiên, chúng tôi tiến hành trừu tượng hóa các thành phần trong mạng cảm biến không dây, sau đó sẽ tiến hành kiểm tra tắc nghẽn trên mạng trừu tượng đó. Kết quả thực nghiệm cho thấy, thời gian phát hiện tắc nghẽn trên mạng trừu tượng nhanh hơn rất nhiều so với mạng nguyên thủy.

Nội dung còn lại của bài báo được chúng tôi bố cục như sau: Mục 2, chúng tôi xin trình bày về các công trình liên quan; Mục 3, chúng tôi xin giới thiệu về mô hình được chúng tôi đề xuất; Mục 4, chúng tôi áp dụng các mô hình này lên bài toán phát hiện tắc nghẽn; Mục 5, xin trình bày kết quả thí nghiệm và cuối cùng chúng tôi sẽ nêu ra các kết luận ở Mục 6.

2 CÁC NGHIÊN CỨU LIÊN QUAN

2.1 Ứng dụng kiểm tra mô hình trong việc kiểm tra các thành phần trong mô hình trừu tượng

Kiểm tra mô hình hay MC [5] cho phép con người có thể kiểm tra sự hoạt động và cũng như đặc điểm của các tính chất đặc biệt của các hệ thống vận hành đồng thời phức tạp. Quá trình kiểm tra này thực chất là quá trình duyệt trên không gian trạng thái mà mô hình sinh ra. Ngày nay, việc mô hình hóa để kiểm tra tính chất của hệ thống thu hút rất nhiều sự quan tâm của các nhà nghiên cứu và cộng đồng. Ví dụ, dự án SLAM [6] ứng dụng ý tưởng kiểm tra trên mô hình để kiểm tra sự hoạt động của microprocessor Intel Core i7 trước khi đưa vào thực tiễn sản xuất. Tuy nhiên, việc kiểm tra các thuộc tính trên mô hình gặp phải trở ngại lớn vì sự bùng nổ không gian trạng thái. Đây là một trong những điểm cản trở việc áp dụng phương pháp mô hình hóa trong công nghiệp. Vì thế, đi kèm với bài toán mô hình hóa, các nghiên cứu luôn quan tâm đến vấn đề giảm sự bùng nổ này. Có hai kỹ thuật phổ biến được đề xuất cho bài toán trên là trừu tượng hóa mô hình và phân rã mô hình. Các kỹ thuật này được gọi tên chung là kiểm tra các thành phần trong mô hình (compositional verification).

Trong kỹ thuật phân rã mô hình, thông thường các hệ thống lớn sẽ được chia thành nhiều hệ thống nhỏ và việc kiểm tra mô hình thay vì thực hiện trên hệ thống chính sẽ được thực hiện trên các hệ thống con [7]. Kỹ thuật trừu tượng hóa về cơ bản chỉ là thu giảm bớt số lượng các place và transition, từ đó số lượng node trong không gian trạng thái sẽ giảm theo. Tuy nhiên, vì trừu tượng hóa nên các tính chất của hệ thống chính sẽ bị mất nên đặc điểm cần lưu ý của phương pháp này là khi chúng ta trừu tượng hóa, ít nhất các thuộc tính chính cần dùng cho việc kiểm tra hay mô hình bắt buộc phải được giữ lại [8]. Điều hạn chế của phương pháp này là chúng ta không thể chứng minh được tính toàn vẹn (completeness) của bài toán.

2.2 Mô hình hóa các hệ thống/giao thức mạng

MC được ứng dụng trong [9] để kiểm tra tính dễ tổn thương (vulnerabilities) của một mô hình mạng. Các nút trong mô hình này mô tả rất nhiều thông tin của kẻ tấn công như địa chỉ muốn tấn công, số lượng các máy tính trong mạng cần tấn công, các quyền mà kẻ tấn công có thể lợi dụng.

Phương pháp mô hình hóa cũng được ứng dụng để kiểm tra việc xây dựng mạng cảm biến không dây trong [10]. Người ta hay triển khai WSN ở những nơi có môi trường tự nhiên khắc nghiệt mà con người ít có khả năng tiếp cận thường xuyên như núi lửa, rừng rậm, các khu vực thường xảy ra động đất... [11]. Vì thế, việc kiểm tra tính khả thi của một mô hình mạng (nghĩa là các gói tin có thể chuyển đi an toàn và nguyên vẹn từ điểm đầu đến điểm đích) là một điều rất khó khăn. Các tác giả bài báo này dựa vào các hành động của cảm biến như tạo ra gói tin, gửi và nhận gói tin... để kiểm tra sự kết nối của các cảm biến. Nhờ vào đó, hệ thống có thể rút ra kết luận cho việc xây dựng mô hình mạng có khả thi hay không?

2.3 Phát hiện tắc nghẽn trong mạng cảm biến không dây

2.3.1 Giới thiệu về mạng cảm biến không dây

Mạng cảm biến không dây là một ứng dụng điển hình của mạng không dây vì khả năng ứng dụng của nó trong nhiều lĩnh vực trong cuộc sống. Mạng cảm ứng không dây là mạng gồm các nút cảm ứng (Sensor-SN). Nút cảm ứng là những thiết bị có bộ xử lý đơn giản, nhỏ gọn và năng lượng hoạt động thấp. Các SN sẽ được phân bố khắp vùng muốn khảo sát, thu thập hình ảnh và truyền hình ảnh qua sóng wifi về trạm giám sát (base station) để giúp con người giám sát hoạt động của hệ thống.

Lợi thế của việc triển khai mạng WSN so với một mạng wireless thông thường là giá thành của các SN rẻ hơn rất nhiều so với các thiết bị thu phát wifi truyền thống (Access point). Tuy nhiên, vì các SN trong mạng WSN thường được phân bố dày đặc do vùng phủ sóng của các thiết bị này hẹp và năng lượng của các sensor cũng không nhiều nên sử dụng WSN chúng ta cũng cần phải quan tâm đến việc giải quyết các ràng buộc của nó. Để kiểm tra xem một mạng cảm biến không dây được xây dựng có thỏa mãn các ràng buộc hay không là một thử thách rất lớn cho các nhà khoa học. Một trong số các ràng buộc thu hút nhiều sự quan tâm nhất là việc *quản lý tắc nghẽn* trên mạng. Quản lý tắc nghẽn bao gồm hai giai đoạn là: phát hiện tắc nghẽn và giảm nghẽn. Trong nghiên cứu này, chúng tôi chỉ xin đề cập đến việc phát hiện tắc nghẽn.

2.3.2 Phát hiện tắc nghẽn trong mạng cảm biến không dây

Trong môi trường của mạng không dây, hệ thống các kết nối được thiết lập thông qua sóng wifi không ổn định so với các kết nối truyền thống của mạng có dây. Bởi vì sự không ổn định này, các gói tin đôi khi cần phải gửi đi gửi lại nhiều lần mới có thể đi đến đích. Điều này dẫn đến số lượng gói tin trao đổi trên mạng là rất lớn và đây là nguyên nhân chính của sự tắc nghẽn [12]. Tắc nghẽn có thể được phát hiện trên mạng phân bố dày đặc lẫn mạng thưa thớt. Trong mạng dày đặc, tắc nghẽn xảy ra khi tràn bộ nhớ đệm của cảm biến do số lượng gói tin cảm biến đó nhận được quá lớn so với khả năng xử lý gói tin của nó. Một khả năng khác dẫn đến tắc nghẽn trên mạng dày đặc là sự đụng độ các gói tin trong quá trình truyền nhận trên mạng. Với mạng thưa thớt, nghẽn cũng có thể xảy ra do sự nhiễu sóng [13].

Để nghiên cứu về các thuộc tính của mạng như độ trễ, khả năng mất gói tin, khả năng gây tắc nghẽn, thông thường, các thuật toán đề xuất sẽ được hiện thực trên các chương trình giả lập. Hai giả lập đang được sử dụng phổ biến hiện nay là ns2[1] và OMNet++ [14]. Sự giả lập này có ưu điểm là cho ta thấy được một cách tổng quát tầm hoạt động, ưu và nhược điểm của một giao thức khi nó hoạt động trên mạng cũng như đưa ra được con số cảnh báo đe dọa sự hoạt động của giao thức. Một hướng tiếp cận, người ta cũng có thể dùng SimuLink[2], một chương trình cho phép người dùng

[1] http://www.isi.edu/nsnam/ns/

[2] https://fr.mathworks.com/products/simulink/index.html ? s_tid= gn_loc_drop/

mô hình hóa hệ thống và giả lập các hoạt động của hệ thống bằng rất nhiều ngôn ngữ khác nhau. Tất cả các công cụ này đều thích hợp để giả lập và khảo sát sự hoạt động của mạng cảm biến không dây.

Tuy nhiên, không phải tất cả các tình huống đều có thể giả lập do không thể mô phỏng được hết các thông tin đầu vào hay do khả năng xử lý hạn chế của các giả lập... Một nhược điểm khác khi sử dụng giả lập là sự mô phỏng các tình huống bắt buộc phải được thực hiện dưới sự hỗ trợ của các framework của giả lập. Chúng tôi nhận thấy rằng cứ vài năm, các giả lập này lại thay đổi sự hỗ trợ cho framework chính, mỗi lần như thế gần như người sử dụng phải bắt đầu mọi thứ lại từ đầu vì sự khác biệt về cấu trúc của các framework. Vì thế, khi framework hỗ trợ thay đổi, người dùng phải thay đổi tất cả mô hình của mình mặc dù sơ đồ mạng và tất cả các tham số đầu vào đều không đổi. Điều này gây ảnh hưởng không nhỏ cho người dùng.

2.3.3 Thảo luận

Với mong muốn có thể hiện thực tất cả các tình huống có thể xảy ra, trong bài báo này, chúng tôi đề xuất một hướng tiếp cận mới cho bài toán phát hiện tắc nghẽn. Đầu tiên chúng tôi sẽ mô hình hóa WSN thành PN dựa vào các đặc điểm của nó. Sau đó, chúng tôi sẽ kiểm tra tắc nghẽn dựa vào các luật được viết bằng LTL (Linear Temporal Logic [5]). Hướng tiếp cận này mang lại hai lợi ích:

– Mô hình được viết ra ở mức trừu tượng hóa cao nên hoàn toàn không phụ thuộc vào giả lập.

– Giả lập được tất cả tình huống có thể xảy ra

Tuy nhiên, hạn chế của cách tiếp cận này là vấn đề bùng nổ không gian trạng thái. Số lượng không gian trạng thái có thể rất lớn, thậm chí là vô cùng vì đây là nơi mô tả toàn bộ các trạng thái của hệ thống.

Để có thể duyệt hết toàn bộ không gian trạng thái là một điều không thể đối với các thuật toán tìm kiếm truyền thống vì sự hạn chế về tài nguyên cũng như thời gian xử lý của máy tính. Vì thế, để giảm thiểu không gian trạng thái mà không làm mất đi các đặc điểm có thể dẫn đến sự tắc nghẽn trên mạng, chúng tôi đề xuất mô hình trừu tượng hóa thành phần. Dựa vào nguyên nhân gây ra tắc nghẽn, ví dụ như tắc nghẽn ở cảm biến do tràn bộ nhớ đệm, chúng ta chỉ cần kiểm tra tắc nghẽn trên cảm biến. Ngược lại, nếu chúng ta quan tâm vào việc kiểm tra xem tắc nghẽn có phải do sự đụng độ gói tin hay do nhiễu sóng, chúng ta sẽ kiểm tra trên kênh truyền.

3 CÁC MÔ HÌNH TRỪU TƯỢNG HÓA THÀNH PHẦN

Trước tiên chúng tôi xin trình bày định nghĩa về Petri net, sau đó chúng tôi sẽ giới thiệu về các khái niệm thành phần (Component) do chúng tôi đề xuất.

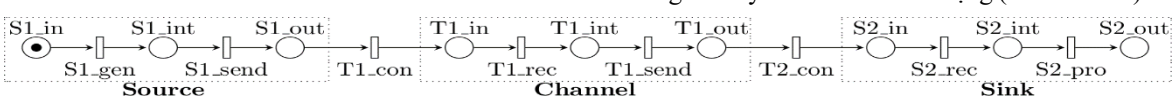
Định nghĩa 1 (Petri net).*Petri net* là một tập hợp $N = \langle P, T, F, W, M_0 \rangle$ trong đó:

– P là tập hữu hạn các vị trí (places)

– T là tập hữu hạn các chuyển tiếp (transitions)

– P và T là các tập hợp rời rạc, nghĩa là P ∩ T = ∅

– F ⊆ (P × T) ∪ (T × P) là hàm xác định các hướng di chuyển của token trên mạng (flow relation)



Hình 1: Component-Based PN của WSN

– W: F → (N \ 0) là trọng số trên các chuyển tiếp

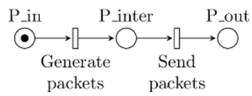
– M_0 : P → N tập hợp các trạng thái marking ban đầu

Định nghĩa 2 (Component).*Component Com* là một Petri net với tập places gồm ba thành phần độc lập nhau: $P_{com} = \{P_{in}, P_{out}, P_{inter}\}$ và được định nghĩa như sau:

– P_{in} là một tập không rỗng của các input place, nghĩa là $T \times P_{in} \cap F = \varnothing$

– P_{out} là một tập không rỗng của các output place, nghĩa là $P_{out} \times T \cap F = \varnothing$

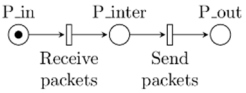
– P_{inter} là tập hợp các place còn lại.



Hình 2: Thành phần của cảm biến nguồn

Một cảm biến trong WSN được mô hình như một Component (Hình 2) với *P_in* là tập hợp input place, *P_out* là tập output place và *P_inter* là tập các place trung gian. Transition *Generate Packets* cho phép cảm biến tạo ra những gói tin mới và transition *Send Packets* sẽ chuyển chúng đi trên đường truyền. Một cách tương tự, kênh truyền

được mô hình trong Hình 3 cũng có hai chức năng chính: nhận gói tin (transition Receive Packets) và gửi gói tin (transition Send Packets).



Hình 3: Thành phần của kênh truyền (Channel)

Định nghĩa 3 (Component-based Petri net). Component-based Petri net là một Petri net bao gồm các tập *Component* rời rạc nhau $S_{Com} = Com_1,...,Com_n$ và tập các *kết nối* (connector) $S_C = C_1,...,C_k$ trong đó connector C_i là một transition từ ouput place của một component đến input place của một component khác.

Hình 1 là một ví dụ của Component-based Petri net bao gồm ba Component Source (S_1), Channel (T_1) và Sink (S_2) và hai connectors T1_con và T2_con. Từ đây, chúng tôi xin gọi mạng Petri net là *WSNCom*.

Định nghĩa 4 (Typed Component based Petri Net). Một component-based Petri net với các tập hợp các Component S_{Com} được xem là typed nếu $\exists f_{type} : S_{Com} \to \wp$ trong đó $\wp$ là một tập hợp hữu hạn với các phần tử rời rạc được đặt tên là *Component Domain*.

Ví dụ, component-based PN *WSNCom* gọi là typed với hàm sau f_{type} sao cho:

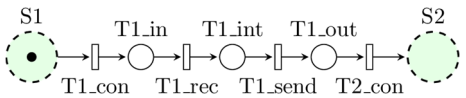
$$\wp = \{Sensor, Channel\}$$
$$f_{type}(S_1) = Sensor$$
$$f_{type}(S_2) = Sensor$$
$$f_{type}(T_1) = Channel$$

Đối với WSN, dễ dàng nhận thấy có hai loại type trong đó gồm sensor và channel.

Định nghĩa 5 (Component Abstraction). Component *Com* trong Component-based PN có thể tối giản thành *abstracted place* (place based abstraction). Tương tự, *Com* và tất cả các *connector* kết nối với nó sẽ tối giản thành abstracted transition (transition based abstraction).

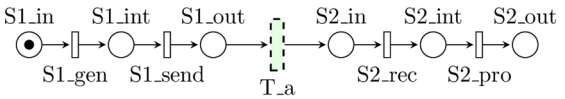
Định nghĩa 6 (Type Abstraction Function). Giả sử *PN-Com* là một component-based PN và $\Im$ là một giá trị trong Component Domain $\wp$. Hàm Type-Abstraction tạo ra các abstracted Petri net Com_A từ *PN-Com* bằng cách thay thế tất cả component typed $\Im$ trong PN-Com bằng abstracted component tương ứng, nghĩa là $Com_A = \Im^{\langle p \rangle}$ PN-Com nếu sự tối giản dựa trên place, hay $Com_A = \Im^{\langle t \rangle}$ PN-Com nếu sự tối giản dựa trên transition.

Hình 4 là một ví dụ của $Sensor^{\langle p \rangle}(WSNCom)$. Tất cả các component typed Sensor, nghĩa là S_1 và S_2 được tối giản thành S_1 và S_2.



Hình 4: Trừu tượng hóa cảm biến

Hơn nữa, Hình 5 thể hiện $Channel^{\langle t \rangle}(WSNCom)$, trong đó component typed Channel T_1 và hai connector ($T1_{con}$ và $T2_{con}$) được tối giản thành transition T_a.

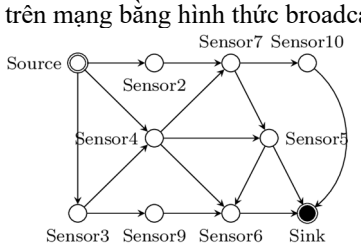


Hình 5: Trừu tượng hóa kênh truyền

4 ỨNG DỤNG PHƯƠNG PHÁP TRỪU TƯỢNG HÓA ĐỂ GIẢM KHÔNG GIAN TRẠNG THÁI CHO WSN

Trong phần này, chúng ta phát triển một công cụ có tên là *WSN-PN* để mô hình hóa WSN đồng thời phát hiện tắc nghẽn bằng cách sử dụng Component-based PN đã đề cập ở trên. Công cụ này trước tiên cho phép người dùng thiết kế hệ thống mạng không dây, sau đó sẽ tự động sinh ra component-based PN tương ứng và tiến hành kiểm tra sự tắc nghẽn trên mô hình dựa vào việc sử dụng bộ thư viện của model checking PAT [15]. Hướng dẫn sử dụng công cụ và các kết quả thí nghiệm được lưu trữ ở website http://cse.hcmut.edu.vn/ ~save/ project/ kwsn/start

Hình 6 minh họa network topology đơn giản của WSN. Chúng tôi giả sử rằng tất cả các gói tin được gửi trên mạng bằng hình thức broadcast.

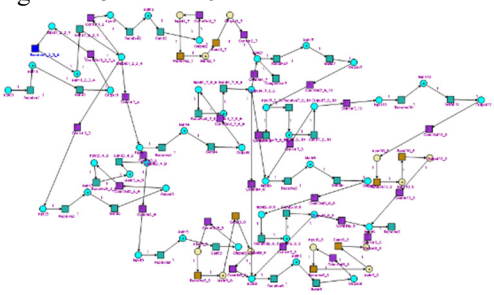


Hình 6: Sơ đồ mạng cảm ứng không dây

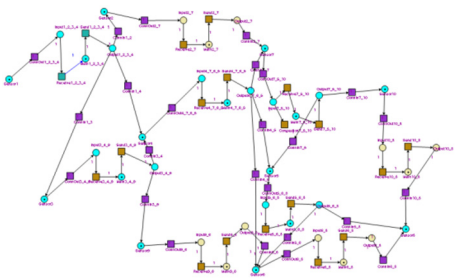
Tất cả các mô hình PN ở Hình 7 đến Hình 9 được sinh ra từ công cụ WSN-PN. Hình 7 là mô

hình PN trong ví dụ của chúng tôi dùng để phát hiện tắc nghẽn trên cả Sensor và Channel.

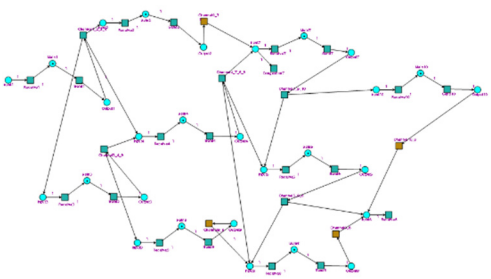
WSN trong ví dụ trên còn được trừu tượng hóa trên Channel hay Sensor, kết quả được thể hiện trong Hình 8 và Hình 9.



Hình 7: Minh họa mạng 10 node



Hình 8: Minh họa trừu tượng hóa cảm biến



Hình 9: Minh họa trừu tượng hóa kênh truyền

5 THỰC NGHIỆM

Chúng tôi sử dụng server có cấu Hình 2.5 Ghz CPU và 24 GiB memory để làm thực nghiệm. Các thông số khởi tạo ban đầu được định nghĩa trong Bảng 1 (hầu hết các thông số này được lấy ra từ kết quả thực nghiệm của CODA.)

Bảng 1: Các tham số khởi tạo

Tham số	Giá trị
Số lượng sensor	5-30
Số lượng gói tin gửi	30-100
Buffer –queue	200-600
Tốc độ gửi gói tin	2-3 packets/ms
Tốc độ xử lý gói tin	1-2 packets/ms
Tốc độ truyền gói tin trên kênh truyền	2-3 packets/ms

Chúng tôi kiểm tra các thuộc tính sau đây bằng công cụ *WSN-PN*:

– *deadlock-free*: kiểm tra xem deadlock có xảy ra hay không.

– *chk-sensor-congestion*: kiểm tra xem tắc nghẽn có xảy trên sensors hay không.

– *chk-channel-congestion*: kiểm tra xem tắc nghẽn có xảy trên channels hay không.

Bảng 2 là kết quả thí nghiệm của chúng tôi. Trong tất cả các trường hợp, các mô hình trừu tượng hóa của chúng tôi, không gian trạng thái sinh ra khi kiểm tra mô hình được trừu tượng hóa (Sensor Abstraction hay Channel Abstraction) giảm đáng kể so với mô hình không trừu tượng (No Abstraction). Tuy nhiên, kết quả kiểm tra trên hai loại mô hình này tương đương nhau, nghĩa là: nếu phát hiện tắc nghẽn xảy ra ở mô hình không trừu tượng (invalid) thì mô hình tắc nghẽn tương ứng cũng phát hiện ra tắc nghẽn.

Bảng 2: Kết quả thực nghiệm

Sen	Mô hình	Thuộc tính	Time	States	Ver.
5	No Abstr	Deadlockfree	0.03	127	Valid
		chk-channel	0.05	249	invalid
		chk-sensor	0.01	33	Valid
	Sensors Abstr	Deadlockfree	0.02	119	Valid
		chk-channel	0.02	85	invalid
	Channels Abst	deadlockfree	0.033	110	Valid
		chk-sensor	0.02	28	Valid
10	No Abstr	deadlockfree	0.167	960	valid
		chk-channel	0.256	1491	invalid
		chk-sensor	0.25	1523	invalid
	Sensors Abstr	deadlockfree	0.269	816	Valid

Sen	Mô hình	Thuộc tính	Time	States	Ver.
	Channels Abst	chk-channel	0.19	1152	invalid
		deadlockfree	0.195	887	Valid
		chk-sensor	0.266	1179	invalid
13	No Abstr	deadlockfree	0.4	1706	Valid
		chk-channel	0.58	2546	invalid
		chk-sensor	0.6	2567	invalid
	Sensors Abstr	deadlockfree	0.2	959	valid
		chk-channel	0.14	711	Invalid
	Channels Abstr	deadlockfree	0.49	1533	valid
		chk-sensor	0.7	2146	invalid
20	No Abstr	deadlockfree	4.19	6897	valid
		chk-channel	8.85	18511	invalid
		chk-sensor	9.01	18649	invalid
	Sensors Abstr	deadlockfree	1.54	4388	valid
		chk-channel	1.56	4492	invalid
	Channels Abst	Deadlockfree	2.21	5394	valid
		chk-sensor	2.81	6261	invalid
30	No Abstr	Deadlockfree	25.44	28318	valid
		chk-channel	96.02	113269	invalid
		chk-sensor	110.89	129524	invalid
	Sensors Abstr	Deadlockfree	3.14	6626	valid
		chk-channel	2.95	6252	invalid
	Channels Abst	Deadlockfree	4.87	7980	valid
		chk-sensor	10.82	19001	Invalid

6 KẾT LUẬN

Trong bài báo này chúng tôi đã đề xuất một hướng tiếp cận để kiểm tra sự tắc nghẽn trên mạng cảm biến không dây bằng cách sử dụng mạng Petri để mô hình hóa. Tắc nghẽn trên mạng cảm biến không dây được chúng tôi phát hiện trên cả mạng dày đặc và mạng thưa thớt. Tắc nghẽn trên mạng thưa thớt được phát hiện chủ yếu trên kênh truyền và trên cả cảm biến lẫn kênh truyền trong trường hợp mạng dày đặc. Bằng việc sử dụng mô hình trừu tượng hóa dựa vào thành phần của chúng tôi đã rút ngắn thời gian kiểm tra tắc nghẽn trên mạng. Thành phần nào không cần kiểm tra sẽ được trừu tượng hóa lại nhằm tiết kiệm thời gian kiểm tra.

Trong tương lai, chúng tôi sẽ nâng cao hiệu quả của việc mô hình hóa trong việc tăng số lượng cảm biến được sử dụng trong thực nghiệm bằng cách sử dụng kỹ thuật gom cụm.

TÀI LIỆU THAM KHẢO

1. V. E. Kozura, V. A. Nepomniaschy, and R. M. Novikov, “Verification of distributed systems modelled by high-level petri nets”, in 2002 International Conference on Parallel Computing in Electrical Engineering (PARELEC 2002), 2002, pp. 61–66.
2. K. Schmidt, “Lola: A low level analyser”, in International Conference on Application and Theory of Petri Nets (ICATPN 2000), 2000, pp. 465–474.
3. M. Heiner, R. Richter, and M. Schwarick, “Snoopy: a tool to design and animate/simulate graph-based formalisms”, in Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops (SimuTools 2008), 2008, p. 15.
4. D. Chiarugi, P. Degano, and R. Marangoni, “A computational approach to the functional screening of genomes”, PLoS Computational Biology, vol. 3, no. 9, 2007.
5. C. Baier and J. Katoen, Principles of model checking. MIT Press, 2008.
6. R. Kaivola, R. Ghughal, N. Narasimhan, A. Telfer, J. Whittemore, S. Pandav, A. Slobodová, C. Taylor, V. Frolov, E. Reeber, and A. Naik, “Replacing testing with formal

verification in intel coretm i7 processor execution engine validation", in Computer Aided Verification, 21st International Conference, (CAV 2009), 2009, pp. 414–429.

7. E. M. Clarke, D. E. Long, and K. L. McMillan, "Compositional modelchecking," in Proceedings of the Fourth Annual Symposium on Logicin Computer Science (LICS '89), 1989, pp. 353–362.
8. D. Peled, A. Valmari, and I. Kokkarinen, "Relaxed visibility enhances partial order reduction", Formal Methods in System Design, vol. 19, no. 3, pp. 275–289, 2001.
9. W. Merro, E. Clarke, and S. Jha, "Model checking for security protocols", CMU, 1997.
10. M. Zheng, J. Sun, Y. Liu, J. S. Dong, and Y. Gu, "Towards a model checker for nesc and wireless sensor networks", in Formal Methods and Software Engineering - 13th International Conference on Formal Engineering Methods, ICFEM 2011, 2011, pp. 372–387.
11. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, vol. 38, no. 4, pp. 393–422, 2002.
12. S. Moon, S. Lee, and H. Cha, "A congestion control technique for the near-sink nodes in wireless sensor networks," in Ubiquitous Intelligence and Computing, Third International Conference, UIC 2006, Wuhan, China, September 3-6, 2006, Proceedings, 2006, pp. 488–497.
13. C. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: congestion detection and avoidance in sensor networks", in Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys 2003). ACM, 2003, pp. 266–279.
14. A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment", in Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops (SimuTools 2008), 2008, p. 60.
15. Y. Si, J. Sun, Y. Liu, J. S. Dong, J. Pang, S. J. Zhang, and X. Yang, "Model checking with fairness assumptions using PAT", Frontiers of Computer Science, vol. 8, no. 1, pp. 1–16, 2014.