ĐẠI HỌC QUỐC GIA TP. HCM
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

VĂN THỊ THIÊN TRANG

# KHAI THÁC MẪU TUẦN TỰ PHỔ BIẾN DỰA TRÊN RÀNG BUỘC

Chuyên ngành: Khoa học máy tính
Mã số ngành: 62 48 01 01

TÓM TẮT LUẬN ÁN TIẾN SĨ KHOA HỌC MÁY TÍNH

**TP. HỒ CHÍ MINH – Năm 2020**

Công trình được hoàn thành tại: **Trường Đại học Công nghệ Thông tin - Đại học Quốc gia Tp. HCM**.

Người hướng dẫn khoa học: GS. TS. Lê Hoài Bắc

Phản biện 1:
Phản biện 2:
Phản biện 3:
Phản biện độc lập 1: PGS. TS. Lê Anh Cường
Phản biện độc lập 2: PGS. TS. Trần Đăng Hưng

Luận án sẽ được bảo vệ trước Hội đồng chấm luận án cấp Trường tại: Phòng E1.1, trường Đại học Công nghệ Thông tin ĐHQG Tp.HCM, vào lúc 08 giờ 30 ngày 26 tháng 02 năm 2021

Có thể tìm hiểu luận án tại thư viện:
- Thư viện Quốc gia Tp.HCM
- Thư viện trường Đại học Công nghệ Thông tin - ĐHQG Tp. HCM.

# MỞ ĐẦU

## 1. Lời nói đầu

Trong luận án này, chúng tôi nghiên cứu bài toán khai thác mẫu tuần tự phổ biến từ cơ sở dữ liệu chuỗi, phục vụ cho nhiều lĩnh vực ứng dụng. Ví dụ như khai thác thói quen mua sắm của khách hàng trong lĩnh vực thương mại, tiếp thị thị trường, khai thác sử dụng web, khai thác chuỗi gen trong sinh học, khai thác mẫu triệu chứng bệnh trong y dược. Tuy nhiên, thách thức đặt ra là tập mẫu khai thác được thường rất lớn, nhưng chỉ một phần nhỏ trong số chúng thật sự có ý nghĩa, đáp ứng mối quan tâm của người dùng; hơn nữa, vì tập dữ liệu dùng để khai thác rất lớn nên quá trình khai thác thường tốn nhiều thời gian và chiếm dụng bộ nhớ. Do đó, chúng tôi đưa vào các ràng buộc đại diện cho mối quan tâm, yêu cầu của người dùng và tiến hành khai thác mẫu dựa trên các ràng buộc này nhằm tìm ra tập mẫu thu gọn theo yêu cầu người dùng và rút ngắn thời gian khai thác, giảm bộ nhớ sử dụng. Như vậy, luận án tập trung nghiên cứu và xây dựng các phương pháp khai thác chung cho bài toán khai thác mẫu tuần tự phổ biến dựa trên các ràng buộc, áp dụng được cho nhiều lĩnh vực ứng dụng có dữ liệu dạng chuỗi và ứng dụng của tập mẫu thỏa ràng buộc tìm được cho quá trình sinh luật có ràng buộc. Ngoài ra, luận án còn đề xuất phương pháp khai thác riêng cho trường hợp ứng dụng khai thác mẫu truy cập web, đáp ứng nhu cầu khám phá tri thức trong thời đại bùng nổ công nghệ web.

## 2. Cấu trúc luận án

Nội dung luận án bao gồm 126 trang (không tính phần danh mục công trình và tài liệu tham khảo), 44 bảng, 31 hình vẽ, phần mở đầu, 5 chương và phần kết luận theo cấu trúc như sau:

*Mở đầu:* Giới thiệu khái quát về hướng nghiên cứu của luận án và cấu trúc luận án.

*Chương 1 - Giới thiệu tổng quan:* Giới thiệu chung về cơ sở dữ liệu chuỗi với các kỹ thuật khai thác trên loại hình dữ liệu này; trình bày tổng quan khai thác mẫu tuần tự dựa trên ràng buộc từ cơ sở dữ liệu chuỗi là bài toán trọng tâm nghiên cứu, khảo sát các công trình nghiên cứu liên quan. Từ đó, nêu lên mục tiêu, phạm vi nội dung nghiên cứu với những đóng góp chính của luận án.

*Chương 2 - Cơ sở lý thuyết*: Trình bày cơ sở lý thuyết cho các phương pháp sử dụng trong đề tài.

*Chương 3 - Khai thác mẫu tuần tự dựa trên ràng buộc Itemset*: Giới thiệu bài toán, đề xuất phương pháp khai thác mẫu tuần tự dựa trên ràng buộc Itemset.

*Chương 4 - Ứng dụng của tập mẫu thỏa ràng buộc Itemset trong khai thác luật có ràng buộc:* Giới thiệu bài toán khai thác luật có ràng buộc Itemset ở vế trái của luật và đề xuất phương pháp khai thác luật bằng cách tận dụng tập mẫu thỏa ràng buộc Itemset.

*Chương 5 - Khai thác mẫu truy cập web dựa trên ràng buộc chuỗi con*: Giới thiệu lĩnh vực khai thác web, giới thiệu bài toán ứng dụng - khai thác mẫu truy cập web có ràng buộc chuỗi con và đề xuất phương pháp khai thác.

*Kết luận và hướng phát triển*: Trình bày tóm tắt kết quả nghiên cứu, hướng phát triển nghiên cứu tiếp theo của đề tài.

Phần cuối của luận án là các công trình khoa học chính, công trình có đóng góp của tác giả, và các tài liệu tham khảo chính gồm 71 tài liệu (bài báo hội thảo và tạp chí quốc tế).

# CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN

## 1.1 Tổng quan khai thác mẫu tuần tự từ cơ sở dữ liệu chuỗi

Phần này trình bày tổng quan về dữ liệu chuỗi với các kỹ thuật khai thác đặc thù trên loại dữ liệu này. Tiếp theo là tổng quan bài toán trọng tâm nghiên cứu - khai thác mẫu tuần tự dựa trên ràng buộc từ cơ sở dữ liệu chuỗi và khảo sát các công trình nghiên cứu đã có trong và ngoài nước và rút ra đánh giá chung về tình hình nghiên cứu.

## 1.2 Động cơ và mục tiêu nghiên cứu

### 1.2.1 Động cơ nghiên cứu

Cho đến nay, đã có nhiều phương pháp khai thác mẫu tuần tự được đề xuất, ngày càng cải tiến song vẫn tồn tại hai thách thức lớn về hiệu quả và hiệu suất thực hiện. Khai thác mẫu tuần tự dựa trên ràng buộc có thể khắc phục cả hai khó khăn này vì ràng buộc đại diện cho những gì người dùng quan tâm và yêu cầu, nó giới hạn các mẫu tìm được chỉ là một tập hợp con gồm các mẫu thỏa một số điều kiện nhất định. Đây chính là động lực thúc đẩy nghiên cứu bài toán khai thác mẫu tuần tự dựa trên ràng buộc.

Ngoài ra, từ mẫu tuần tự có thể sinh ra luật tuần tự. Nó mở rộng khả năng sử dụng và ý nghĩa biểu đạt của mẫu tuần tự. Luật tuần tự tuy khá đơn giản nhưng những thông tin mà luật mang lại có nhiều ý nghĩa quan trọng, hỗ trợ cho quá trình ra quyết định, quản lý và có tính định hướng. Nếu khai thác trả về tập đầy đủ các luật tuần tự thì tốn nhiều thời gian, bộ nhớ và số lượng luật quá lớn. Tuy nhiên, nếu khai thác theo yêu cầu người dùng, tức là khai thác luật có ràng buộc thì giải quyết được các thách thức này. Đó chính là động cơ cho nghiên cứu mở rộng sinh luật có ràng buộc từ tập mẫu thỏa ràng buộc khai thác được.

### 1.2.2 Mục tiêu nghiên cứu của luận án

Mục tiêu chính của đề tài là đề xuất các thuật toán khai thác mẫu tuần tự tập trung theo yêu cầu của người dùng một cách hiệu quả bằng cách đưa vào các ràng buộc itemset, ràng buộc chuỗi con, sao cho có thể tìm được trực tiếp tập mẫu thỏa ràng buộc một cách chính xác, rút ngắn thời gian khai thác và giảm bộ nhớ sử dụng. Cụ thể mục tiêu giải quyết các bài toán sau:

***Bài toán 1*:** khai thác mẫu tuần tự dựa trên ràng buộc itemset là đi tìm tất cả các chuỗi con phổ biến trong CSDL chuỗi, mà có chứa bất kỳ itemset nào trong tập itemset ràng buộc $\mathbb{C}$ do người dùng yêu cầu. Mục tiêu của luận án là phát triển thuật toán giải quyết bài toán này một cách hiệu quả. Ngoài ra, từ kết quả nghiên cứu này, luận án mở rộng mục tiêu phát triển thuật toán khai thác luật tuần tự có ràng buộc Itemset ở vế trái luật bằng cách tận dụng tập mẫu thỏa ràng buộc Itemset khai thác được cho quá trình sinh luật.

***Bài toán 2*:** Từ kết quả nghiên cứu của *bài toán 1*, luận án mở rộng mục tiêu phát triển thuật toán khai thác luật tuần tự có ràng buộc Itemset ở vế trái luật bằng cách tận dụng tập mẫu thỏa ràng buộc Itemset khai thác được cho quá trình sinh luật.

***Bài toán 3*:** khai thác mẫu truy cập web dựa trên ràng buộc chuỗi con là đi tìm tất cả mẫu phổ biến trong CSDL chuỗi truy cập web mà có chứa bất kỳ mẫu nào của tập ràng buộc *U* (do người dùng chỉ ra) dưới dạng chuỗi con. Mục tiêu của luận án là phát triển thuật toán riêng cho lĩnh vực ứng dụng khai thác mẫu truy cập web dựa trên ràng chuỗi con, nhằm đáp ứng nhu cầu khám phá tri thức trong thời đại bùng nổ công nghệ web hiện nay.

### 1.3 Phạm vi, nội dung và phương pháp nghiên cứu

#### 1.3.1 Đối tượng, phạm vi nghiên cứu

Một là, nghiên cứu mô hình dữ liệu chuỗi, là loại hình dữ liệu rất phổ biến, có mặt trong không gian đo lường bất kỳ có thứ tự toàn phần hay thứ tự bộ phận. Cụ thể, đề tài nghiên cứu hai loại chuỗi điển hình:

(i) Chuỗi sự kiện trong đó mỗi sự kiện gồm nhiều item như chuỗi các giao dịch mua sắm của khách hàng, chuỗi lịch sử bán hàng của một cửa hàng. Trong thực nghiệm nghiên cứu, các CSDL chuỗi loại này được tạo ra bởi công cụ sinh dữ liệu của IBM.

(ii) Chuỗi sự kiện mà mỗi sự kiện chỉ có một item như chuỗi truy cập web, chuỗi dữ liệu sinh học. Các CSDL chuỗi loại này (đã qua bước tiền xử lý) được lấy từ kho dữ liệu KDD Cup 2000.

Hai là, nghiên cứu khai thác mẫu trên hai loại ràng buộc cụ thể, đó là ràng buộc Itemset và ràng buộc chuỗi con về đặc điểm, tính chất và ảnh hưởng của chúng đến quá trình khai thác cũng như kết quả khai thác và lĩnh vực ứng dụng cụ thể.

#### 1.3.2 Nội dung nghiên cứu

Nghiên cứu các phương pháp khai thác mẫu tuần tự phổ biến từ cơ sở dữ liệu chuỗi dựa trên ràng buộc Itemset, trong đó đưa ràng buộc vào ngay trong quá trình khai thác (Chương 3). Ngoài ra, nghiên cứu ứng dụng của tập mẫu thỏa ràng buộc Itemset tìm được trong khai thác luật tuần tự có ràng buộc Itemset ở vế trái của luật (Chương 4).

Nghiên cứu các phương pháp khai thác mẫu truy cập web dựa trên ràng buộc chuỗi con (Chương 5).

#### 1.3.3 Phương pháp nghiên cứu

Vì đề tài nghiên cứu khai thác mẫu tuần tự tập trung vào mối quan tâm, nhu cầu của người dùng nên phương pháp nghiên cứu là tìm cách đưa ràng buộc vào ngay trong quá trình khai thác mẫu. Khảo sát các

loại ràng buộc đã có, phân tích và chọn ra loại ràng buộc có tính ứng dụng cao trong thực tiễn hiện nay. Khảo sát các công trình đã công bố trong và ngoài nước, tổng hợp và rút ra các ưu nhược điểm của từng phương pháp, từ đó phát triển thuật toán hiệu quả.

Phương pháp tiến hành lấy dữ liệu thực nghiệm: sử dụng chương trình sinh dữ liệu chuẩn IBM để sinh các bộ dữ liệu giả lập. Đây là chương trình được sử dụng ở tất cả các nghiên cứu khai thác mẫu tuần tự đã có trên thế giới. Còn các bộ dữ liệu thực được lấy từ kho dữ liệu máy học UCI[1], là các bộ dữ liệu đã qua bước tiền xử lý. Như vậy, chúng tôi sử dụng các cơ sở dữ liệu thực nghiệm như của các nhóm nghiên cứu khác để đối chiếu so sánh kết quả thực nghiệm và chứng minh tính hiệu quả của công trình đề xuất.

Phương pháp đánh giá kết quả nghiên cứu: Tiến hành cài đặt các thuật toán đề xuất. Thông qua kết quả thực nghiệm để chứng minh tính hiệu quả của phương pháp đề xuất về tập kết quả khai thác được, về thời gian thực thi và tiêu tốn bộ nhớ so sánh đối chiếu với các thuật toán đã có.

### 1.4 Đóng góp của luận án

Đóng góp chính của luận án là đề xuất các thuật toán khai thác mẫu tuần tự dựa trên ràng buộc và ứng dụng cho khai thác luật có ràng buộc từ CSDL chuỗi, bao gồm:

+ Đề xuất thuật toán khai thác mẫu tuần tự dựa trên ràng buộc Itemset – thuật toán *MSPIC-DBV* [CT1], đóng góp ở chương 3. Thuật toán mở rộng và phát triển cách tổ chức dữ liệu biểu diễn dọc - đề xuất cấu trúc *DBVP* làm đại diện biểu diễn lại CSDL theo chiều dọc nhờ vậy chỉ duyệt CSDL một lần. Bằng cách sử dụng cấu trúc cây tiền

[1] http://mlr.cs.umass.edu/ml/datasets.html

tố kết hợp *DBVP* để lưu không gian tìm kiếm, thuật toán đưa ra kỹ thuật tỉa không gian con theo tiền tố và kỹ thuật kiểm tra ràng buộc theo tiền tố có thể bỏ qua việc kiểm tra ràng buộc cho một số lượng lớn các mẫu ứng viên.

+ Đề xuất thuật toán khai thác luật tuần tự với ràng buộc Itemset ở vế trái của luật gồm bộ ba thuật toán *MSRIC-B, MSRIC-R* và *MSRIC-P* [CT2, CT3]; đóng góp ở chương 4. Trong đó, *MSRIC-B* là phương pháp cơ sở đơn giản đưa ràng buộc vào sau quá trình khai thác, hai thuật toán còn lại đưa vào trong quá trình khai thác. *MSRIC-R* đưa ở giai đoạn sinh luật, còn *MSRIC-P* đưa ở giai đoạn tìm mẫu, tận dụng kết quả của thuật toán *MSPIC-DBV*. *MSRIC-P* là thuật toán đóng góp chính, hiệu quả hơn hai thuật toán còn lại.
+ Đề xuất hai thuật toán khai thác mẫu truy cập web dựa trên ràng buộc chuỗi con gồm *MWAPC* và *EMWAPC* [CT4] là đóng góp chính của chương 5. Trong đó, *EMWAPC* là thuật toán đóng góp chính, cải tiến của *MWAPC*. *EMWAPC* sử dụng cấu trúc dữ liệu và các kỹ thuật tương tự phương pháp khai thác mẫu với ràng buộc Itemset. Tuy nhiên, dựa vào đặc điểm của mẫu truy cập web, thuật toán thực hiện tỉa nhanh không gian tìm kiếm ngay từ đầu và giảm thiểu việc kiểm tra ràng buộc dựa vào đặc điểm của ràng buộc chuỗi con.

# CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

## 2.1 Các khái niệm và định nghĩa

**Định nghĩa khai thác mẫu tuần tự.** Cho trước CSDL chuỗi *SDB* và ngưỡng phổ biến tối thiểu *minSup* do người dùng qui định trước, bài toán khai thác mẫu tuần tự là tìm tất cả các chuỗi con phổ biến hay mẫu tuần tự phổ biến có trong *SDB*.

Gọi $FP$ là tập các mẫu tuần tự phổ biến trong *SDB*, ta có:

$$FP = \{p \in SDB \mid sup(p) \geq minSup\}.$$

**Định nghĩa khai thác mẫu tuần tự dựa trên ràng buộc.** Ràng buộc $\mathbb{C}$ trong khai thác mẫu tuần tự là một hàm Boolean $\mathbb{C}(p)$ trên các mẫu. Cho CSDL chuỗi *SDB*, ràng buộc $\mathbb{C}$ và ngưỡng phổ biến tối thiểu *minSup* do người dùng đưa ra. Bài toán khai thác mẫu tuần tự dựa trên ràng buộc là tìm tất cả các mẫu tuần tự phổ biến trong CSDL thỏa ràng buộc $\mathbb{C}$.

$$FCP = \{p \in SDB \mid sup(p) \geq minSup \wedge \mathbb{C}(p) = \text{true}\}.$$

## 2.2 Các loại ràng buộc

Jian Pei và đồng sự đã khảo sát và đưa ra định nghĩa cho bảy loại ràng buộc xuất hiện phổ biến trong các lĩnh vực ứng dụng, bao gồm: ràng buộc item, ràng buộc độ dài, ràng buộc chuỗi con, ràng buộc kết hợp, ràng buộc biểu diễn dưới dạng biểu thức có quy tắc, ràng buộc về khoảng thời gian xảy ra của sự kiện đầu và cuối trong mẫu, ràng buộc về khoảng thời gian giữa hai sự kiện kề nhau trong mẫu. Mặc dù chưa hoàn toàn đầy đủ, nhưng hầu như đã khái quát nhiều ràng buộc hữu ích trong các lĩnh vực ứng dụng.

## 2.3 Đặc trưng của các thuật toán khai thác mẫu tuần tự

Khi phát triển một thuật toán để khai thác mẫu tuần tự từ CSDL chuỗi, yếu tố đại diện cho hiệu suất khai thác là chi phí bộ nhớ sử dụng và tốc độ xử lý dữ liệu. Do đó, phải sử dụng cấu trúc dữ liệu thích hợp

và thuật toán tối ưu. Như vậy, các đặc trưng ảnh hưởng đến hiệu suất của thuật toán là:

- Cách tổ chức biểu diễn dữ liệu để lưu trữ vào bộ nhớ.
- Các hướng tiếp cận để tìm và liệt kê mẫu tuần tự.
- Kỹ thuật tạo mẫu ứng viên.
- Phương pháp duyệt không gian tìm kiếm.

Ngoài ra, sử dụng một số đặc trưng khác như lý thuyết đồ thị, đưa ra những ràng buộc cho bài toán sẽ giúp thuật toán thực thi nhanh hơn, các mẫu phổ biến tìm được có giá trị hơn.

## 2.4 Phân loại các phương pháp khai thác

Khai thác mẫu tuần tự là bài toán khá thông dụng, thu hút nhiều nghiên cứu. Cho đến nay, nhiều thuật toán đã được đề xuất để giải quyết bài toán này. Dựa vào cách thức tổ chức dữ liệu, có thể chia các thuật toán thành hai lớp như sau:

(1) Lớp thuật toán tổ chức dữ liệu biểu diễn ngang.

(2) Lớp thuật toán tổ chức dữ liệu biểu diễn dọc.

## 2.5 Kết chương

Tóm lại, chương này trình bày toàn bộ cơ sở lý thuyết nền tảng liên quan đến lĩnh vực khai thác mẫu tuần tự từ CSDL chuỗi dựa trên các ràng buộc. Trong đó, trọng tâm là tìm hiểu, phân loại và phân tích ưu nhược điểm của các phương pháp khai thác mẫu đã có. Từ đó, chọn ra cách thức tổ chức dữ liệu, kỹ thuật tạo mẫu, cách duyệt không gian tìm kiếm áp dụng cho khai thác với loại ràng buộc cụ thể, xuất hiện khá phổ biến trong nhiều lĩnh vực ứng dụng, đó là ràng buộc Itemset được trình bày ở Chương 3, Chương 4 và ràng buộc chuỗi con ở Chương 5.

# CHƯƠNG 3. KHAI THÁC MẪU TUẦN TỰ DỰA TRÊN RÀNG BUỘC ITEMSET

## 3.1 Giới thiệu

Tùy thuộc từng lĩnh vực ứng dụng, đã có nhiều nghiên cứu trên những loại ràng buộc khác nhau. Các ràng buộc đã được nghiên cứu đề xuất thuật toán; tuy nhiên, cho đến nay chưa có nghiên cứu riêng nào về ràng buộc Itemset, là ràng buộc khá phổ biến và phù hợp cho nhiều lĩnh vực ứng dụng. Ràng buộc Itemset yêu cầu mẫu khai thác được phải chứa một trong các itemset cho trước. Trong chương này, chúng tôi giải quyết bài toán khai thác mẫu tuần tự dựa trên ràng buộc Itemset bằng cách đưa ràng buộc vào trong quá trình khai thác, sao cho có thể tìm được trực tiếp tập mẫu thỏa ràng buộc, rút ngắn thời gian khai thác và bộ nhớ sử dụng [CT1].

## 3.2 Phát biểu bài toán

Cho CSDL *SDB*, tập itemset ràng buộc $\mathbb{C} = \{c_1, c_2 ... c_n\}$ và ngưỡng phổ biến tối thiểu *minSup* do người dùng chỉ ra. Bài toán khai thác mẫu tuần tự dựa trên ràng buộc itemset là đi tìm tất cả các chuỗi con phổ biến trong CSDL mà có chứa bất kỳ itemset nào của tập $\mathbb{C}$.

$FCP = \{p | sup(p) \geq minSup \wedge \exists i: 1 \leq i \leq \text{size}(p), \exists k: 1 \leq k \leq n, p[i] \supseteq c_k, c_k \in \mathbb{C}\}$.

## 3.3 Các nghiên cứu liên quan

Các thuật toán thuộc lớp tổ chức dữ liệu biểu diễn dọc được đánh giá là hiệu quả hơn tất cả các thuật toán khác về thời gian và bộ nhớ sử dụng. Do đó, trong phần này trình bày một số phương pháp thuộc lớp các thuật toán định dạng dọc tiêu biểu và tiên tiến để đối chiếu so sánh với phương pháp đề xuất. gồm:

+ Phương pháp sử dụng vector bit/ bitmap
+ Phương pháp mã hóa nguyên tố trên vector bit

+ Phương pháp vector bit động

### 3.4 Phương pháp đề xuất

Đề xuất phương pháp khai thác với thuật toán *MSPIC-DBV*, sử dụng cấu trúc dữ liệu *DBVP*, không gian tìm kiếm là cây tiền tố. Dựa trên cơ sở lý thuyết của 3 mệnh đề đề xuất, thuật toán có thể thu gọn không gian tìm kiếm, giảm chi phí xử lý khi tạo mẫu ứng viên, giảm thiểu việc kiểm tra ràng buộc. Nhờ đó, thuật toán *MSPIC-DBV* tốn ít thời gian và bộ nhớ hơn các thuật toán trước. Thuật toán *MSPIC-DBV* có thể tóm tắt như sau:

**Bước 1:** Tìm tập mẫu phổ biến độ dài *1* lưu vào tập $F_1$ (dòng 1). Tập *FCP* cần tìm ban đầu rỗng (dòng 2).

**Bước 2**: Tìm những itemset ràng buộc phổ biến. (dòng 3). Bước này sẽ tìm itemset nào trong tập $\mathbb{C}$ có độ phổ biến thỏa *minSup*.

**Bước 3**: Biến đổi *DBVP* của các atom trong tập F1 (dòng 4). việc biến đổi giúp tỉa không gian tìm kiếm theo tiền tố và chuỗi con, giảm thiểu chi phí tính toán khi mở rộng mẫu.

**Bước 4:** Kiểm tra ràng buộc và mở rộng mẫu (dòng 5-12) bằng thủ tục *PREFIX-EXTENSION()* như Bảng 3.8) và *PREFIX-EXTENSION-CHECK()* như Bảng 3.9.

***Bảng 3.1.*** *Thuật toán MSPIC-DBV.*

| **Thuật toán *MSPIC-DBV*** |
|---|
| **Đầu vào:** *SDB*, $\mathbb{C} = \{c_1, c_2 \ldots, c_n\}$, *minSup*. |
| **Đầu ra:** Tất cả mẫu tuần tự thỏa *minSup* và thỏa ràng buộc Itemset. |
| $FCP = \{\alpha \mid \exists i: 1 \le i \le size(\alpha), \exists k: 1 \le k \le n, \alpha[i] \supseteq c_k\}$. |
| 1. Duyệt *SDB* để tìm $F_1 = \{atom$ và $DBVP_{atom} \mid atom \in I \wedge sup(atom) \ge minSup\}$; |

```
2. FCP = ∅;
3. Gọi thủ tục FIND_FRE_CONSTRAINT_ITEMSET(F1, ℂ, minSup);
4. Gọi thủ tục TRANSFORM(F1, ℂ, minSup);
5. For each node n in F1 do
6.    For each c in ℂ do
7.       If (n.sequence thỏa ràng buộc c) then
8.           FCP = FCP ∪ {n.sequence};
9.           PREFIX-EXTENSION(n, F1, F1, minSup);
10.              break;
11.      If (n.sequence không thỏa ràng buộc c nào, ∀c ∈ ℂ) then
12.          PREFIX-EXTENSION-CHECK(n, F1, F1, minSup, ℂ);
```

***Bảng 3.2.*** *Thủ tục mở rộng mẫu từ tiền tố, tạo mẫu chắc chắn thỏa ràng buộc.*

```
Thủ tục 3. PREFIX-EXTENSION(p, S, I, minSup)
//Mở rộng sequence
1.  S1 = {i ∈ S| sup(pi = Mở rộng theo sequence(p, i)) ≥ minSup};
2.  For each item i in S1 do
3.    FCP = FCP ∪ {pi.sequence};
4.    PREFIX-EXTENSION(pi, S1, item ∈ S1 lớn hơn i, minSup);
//Mở rộng itemset: tương tự mở rộng sequence
```

***Bảng 3.3.*** *Thủ tục mở rộng theo tiền tố, tạo mẫu ứng viên mới phải kiểm tra ràng buộc.*

```
Thủ tục 4. PREFIX-EXTENSION-CHECK(p, S, I, minSup, ℂ)
```

```
   //Mở rộng sequence
1. S1 = {i ∈ S| sup(pi = Mở rộng sequence (p, i)) ≥
   minSup};
2. For each item i in S1 do
3.     For each c in ℂ do
4.           If (pi.sequence thỏa ràng buộc c) then
5.                 FCP = FCP ∪ {pi.sequence};
6.                 PREFIX-EXTENSION(pi, S1, item ∈ S1
   lớn hơn i, minSup);
7.                 break;
8.           If (pi.sequence không thỏa c nào, ∀c ∈ ℂ)
   then
9.           PREFIX-EXTENSION-CHECK(pi, S1, item ∈ S1
   lớn hơn i, minSup, ℂ);
   //Mở rộng Itemset: tương tự mở rộng sequence
10. I1 = {i ∈ I| sup(pi = Mở rộng Itemset(p, i)) ≥ minSup};
11. For each item i in I1 do
12.    For each c in ℂ do
13.          If (pi.sequence thỏa ràng buộc c) then
14.                FCP = FCP ∪ {pi.sequence};
15.                PREFIX-EXTENSION(pi, S1, item ∈ I1
   lớn hơn i,minSup);
16.                break;
17.    If (pi.sequence không thỏa c nào, ∀c ∈ ℂ) then
18.          PREFIX-EXTENSION-CHECK(pi, S1, item ∈ I1
   lớn hơn i, minSup, ℂ);
```

### 3.5 Kết quả thực nghiệm

**Thuật toán:** Thực nghiệm so sánh hiệu suất thực hiện của các thuật toán đề xuất bao gồm *MSPIC-Naïve* và *MSPIC-DBV* với *PRISM-IC* và *CM-SPAM-IC* (thuật toán mở rộng từ *PRISM* và *CM-SPAM*) để khai thác mẫu tuần tự với ràng buộc itemset. Trong đó, cả hai thuật toán *MSPIC-Naïve* và *MSPIC-DBV* đều sử dụng cấu trúc *DBVP,* nhưng *MSPIC-DBV* áp dụng các kĩ thuật giúp thu gọn không gian tìm kiếm và giảm thiểu việc kiểm tra ràng buộc.

**Cơ sở dữ liệu:** Các bộ dữ liệu mà itemset có kích thước là 1 gồm: Gazelle và Kosarak.

Các bộ dữ liệu có kích thước itemset lớn hơn hoặc bằng 1 gồm: C20T20S20I20N100D1k và C20T50S20I10N1kD100k

**Kết quả thực nghiệm:** Thực nghiệm so sánh hiệu suất thực hiện của các thuật toán khai thác mẫu dựa trên ràng buộc Itemset với sự thay đổi giá trị của *minSup* và *selectivity*. Trên tất cả các loại CSDL thực nghiệm, tập mẫu khai thác được ở cả bốn thuật toán đều giống nhau nhưng thời gian thực hiện và bộ nhớ sử dụng là khác nhau. Các kết quả thực nghiệm đã cho thấy rằng việc đưa ràng buộc vào quá trình khai thác là hiệu quả và thuật toán đề xuất *MSPIC-DBV* chạy nhanh và tốn ít bộ nhớ hơn so với các thuật toán *MSPIC-Naïve, PRISM-IC* và *CM-SPAM-IC*.

### 3.6 Kết chương

Tóm lại, chương này đã trình bày bài toán khai thác mẫu tuần tự dựa trên ràng buộc Itemset và đề xuất phương pháp khai thác với thuật toán *MSPIC-DBV* [CT1][2].

[2] [CT1] V. Trang, V. Bay, & L. Bac (2018), "Mining sequential patterns with itemset constraints", *Knowledge and Information Systems*, vol. 57(2), pp. 311-330 (Springer, SCIE, Q1, IF=2.397).

# CHƯƠNG 4. ỨNG DỤNG CỦA TẬP MẪU THỎA RÀNG BUỘC ITEMSET TRONG KHAI THÁC LUẬT CÓ RÀNG BUỘC

## 4.1 Giới thiệu

Chương này trình bày bài toán khai thác luật tuần tự từ cơ sở dữ liệu chuỗi với ràng buộc Itemset ở vế trái của luật và đưa ra phương pháp giải quyết bài toán này [CT2][CT3], trong đó phương pháp sinh luật trực tiếp bằng cách sử dụng tập mẫu thỏa ràng buộc Itemset hiệu quả hơn so với các phương pháp khác.

## 4.2 Phát biểu bài toán và các nghiên cứu liên quan

**Định nghĩa luật thỏa ràng buộc.** Cho một itemset ràng buộc $c$, luật $r = \langle a_1\ a_2\ ...\ a_n\rangle \rightarrow \langle b_1\ b_2\ ...\ b_m\rangle$ được coi là thỏa ràng buộc $c$ nếu mẫu $\langle a_1\ a_2\ ...\ a_n\rangle$ ở vế trái của luật là *mẫu thỏa ràng buộc itemset c.*

**Bài toán khai thác luật tuần tự với ràng buộc Itemset:**

Cho CSDL *SDB*, tập itemset ràng buộc $\mathbb{C} = \{c_1, c_2... c_n\}$, ngưỡng phổ biến tối thiểu *minSup* và ngưỡng tin cậy tối thiểu *minConf* do người dùng chỉ ra. Bài toán khai thác luật tuần tự với ràng buộc Itemset là đi tìm tất cả các luật thỏa ràng buộc với độ phổ biến và độ tin cậy thỏa mãn các ngưỡng *minSup* và *minConf*.

$$C\mathcal{R} = \{r: X \rightarrow Y |\ sup(r) \geq minSup \wedge conf(r) \geq minConf \wedge \exists k: 1 \leq k \leq n, X \supseteq c_k, c_k \in \mathbb{C}\}.$$

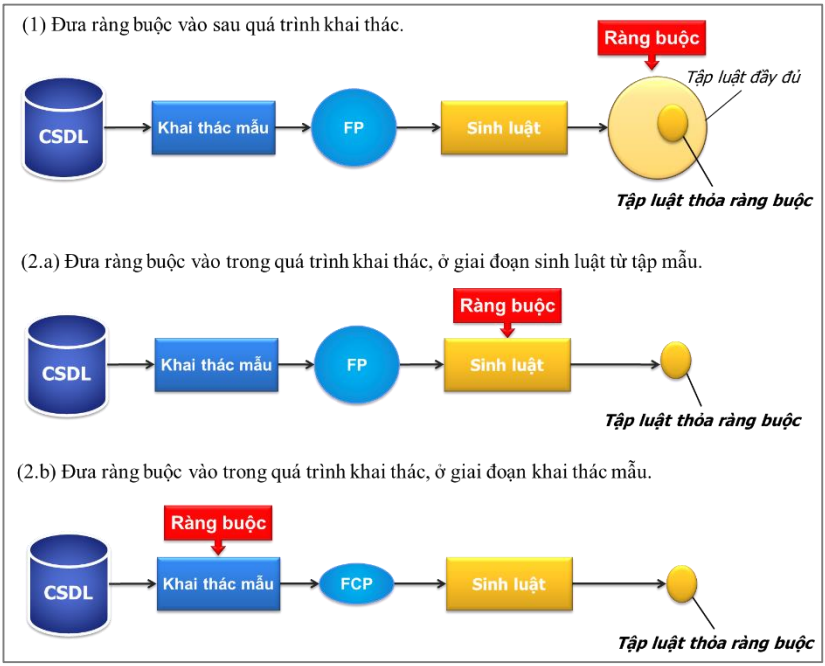
**Các nghiên cứu liên quan:**

Đối với bài toán khai thác luật tuần tự, các nghiên cứu đã đề xuất thực hiện trên hai loại luật. Loại thứ nhất là luật tuần tự chuẩn, là luật có vế trái và vế phải là các mẫu tuần tự. Loại thứ hai gọi là luật tuần tự có thứ tự bộ phận, trong đó các itemset trong mỗi vế của luật không cần có thứ tự. Trong nghiên cứu của luận án, chúng tôi nghiên cứu trên loại thứ nhất, vì thứ tự của các sự kiện đóng vai trò quan trọng và có

ý nghĩa trong nhiều lĩnh vực ứng dụng như phân tích thị trường chứng khoán, công nghệ phần mềm, chăm sóc sức khỏe y tế.

Để giải quyết bài toán khai thác luật tuần tự, có hai hướng tiếp cận chính. Một là chia quá trình khai thác luật thành hai giai đoạn gồm tìm tập mẫu phổ biến và sinh luật từ tập mẫu phổ biến tìm được; hai là khai thác luật trực tiếp từ cơ sở dữ liệu, hướng tiếp cận này phù hợp với luật tuần tự có thứ tự bộ phận. Do đó, trong nghiên cứu này, chúng tôi sử dụng hướng tiếp cận thứ nhất để khai thác luật có ràng buộc itemset bằng cách tận dụng tập mẫu thỏa ràng buộc đã khai thác được.

## 4.3 Phương pháp khai thác luật với ràng buộc Itemset.




***Hình 4.1.**. Các mô hình khai thác luật tuần tự với ràng buộc Itemset.*

Vì quá trình khai thác luật gồm 2 giai đoạn: tìm các mẫu tuần tự thỏa ngưỡng phổ biến tối thiểu và sinh luật đáng tin cậy từ các mẫu

phổ biến tìm được. Do đó, có thể đưa ràng buộc vào giai đoạn sinh luật hoặc giai đoạn tìm mẫu như **Hình 4.1**.

Khai thác mẫu dựa trên ràng buộc Itemset thu được tập mẫu thỏa ràng buộc, cô đọng theo mối quan tâm của người dùng, số lượng mẫu giảm đi đáng kể. Vì vậy, nếu sinh luật tuần tự từ tập mẫu này thì tập luật thu được cũng thỏa ràng buộc và đáp ứng theo yêu cầu người dùng. Để thấy rõ hiệu quả của ứng dụng tập mẫu thỏa ràng buộc trong khai thác luật có ràng buộc Itemset, luận án đã đưa ra ba thuật toán gồm *MSRIC-B, MSRIC-R,* và *MSRIC-P*, đồng thời so sánh kết quả thực hiện của chúng. Trong đó, *MSRIC-B* là phương pháp cơ bản đưa ràng buộc vào kiểm tra sau khi khai thác xong tập luật đầy đủ, còn *MSRIC-R* và *MSRIC-P* đều đưa ràng buộc vào trong quá trình khai thác. *MSRIC-R* đưa ở giai đoạn sinh luật, *MSRIC-P* ở giai đoạn khai thác mẫu. Điều đáng chú ý là thuật toán *MSRIC-P* sử dụng tập mẫu thỏa ràng buộc Itemset sinh trực tiếp ra được các luật thỏa ràng buộc Itemset ở vế trái ngay mà không cần kiểm tra ràng buộc như hai thuật toán kia.

**Thuật toán *MSRIC-B*:** Phương pháp cơ sở thực hiện đưa ràng buộc vào sau quá trình khai thác như Hình 4.1.(1)**.** Các giai đoạn khai thác tiến hành như sau:

(1) Tìm tập *FP* gồm tất cả các mẫu tuần tự phổ biến từ CSDL.

(2) Sinh tập luật tin cậy $R$ từ các mẫu trong tập *FP*, tức là tạo ra và chọn các luật *r* có $sup(r) \geq minConf$.

(3) Kiểm tra ràng buộc trên từng luật $r \in R$ để chọn ra các luật thỏa ràng buộc Itemset ở vế trái của luật, thu được tập *FCR*.

**Thuật toán *MSRIC-R*:** Phương pháp đưa ràng buộc vào trong quá trình khai thác, đưa trực tiếp vào giai đoạn sinh luật như Hình 4.1.(2a)**.** Theo phương pháp này, quá trình khai thác gồm hai giai đoạn:

(1) Tìm tập FP gồm tất cả các mẫu tuần tự phổ biến từ CSDL, tập mẫu thu được lưu trên cấu trúc cây tiền tố.

(2) Sinh tập luật tin cậy thỏa ràng buộc $C\vec{R}$ từ các mẫu trong tập FP, tức là tạo ra và chọn các luật r có conf(r) ≥ minConf và thỏa ràng buộc.

Lưu ý rằng, luật tạo ra phải thỏa ràng buộc Itemset ở vế trái, do đó phải kiểm tra mẫu ở vế trái có chứa itemset ràng buộc không. Để tránh phải kiểm tra ràng buộc cho mọi mẫu ở vế trái luật, thuật toán đề xuất kỹ thuật bỏ qua bước kiểm tra ràng buộc cho một số lượng lớn các luật.

**Thuật toán *MSRIC-P*:** Thuật toán *MSRIC-P* cũng sử dụng phương pháp đưa ràng buộc vào trong quá trình khai thác, tuy nhiên đưa ở giai đoạn khai thác mẫu như Hình 4.1.(2b). Thuật toán *MSRIC-P* sẽ sinh trực tiếp ra mọi luật thỏa ràng buộc ngay lập tức từ tập mẫu thỏa ràng buộc tìm được, mà không cần phải kiểm tra ràng buộc trên luật như thuật toán *MSRIC-R*. Như vậy, ở thuật toán này, cây tiền tố được sử dụng để lưu trữ các mẫu tuần tự đã thỏa ràng buộc itemset.

(1) Tìm tập *FCP* gồm tất cả các mẫu tuần tự phổ biến thỏa ràng buộc Itemset từ CSDL, tập mẫu thu được lưu trên cấu trúc cây tiền tố, sử dụng thuật toán *MSPIC-DBV* (thuật toán đóng góp ở Chương 3).

(2) Sinh các luật *r* đáng tin cậy thỏa ràng buộc từ tập mẫu *FCP*.

### 4.4 Kết quả thực nghiệm

Thực nghiệm tiến hành trên CSDL Gazelle đại diện cho loại CSDL thứ nhất và C20T50S20I10N1kD100k đại diện cho loại thứ hai như mô tả ở Chương 3.

Thực nghiệm so sánh thời gian thực hiện và bộ nhớ sử dcủa các thuật toán khai thác luật tuần tự có ràng buộc Itemset ở vế trái: *MSRIC-*

*B, MSRIC-R* và *MSRIC-P* với sự thay đổi giá trị của *minSup, minConf* và *selectivity* trên cả hai loại dữ liệu chuỗi. Trong tất cả các trường hợp, tập luật khai thác được ở cả ba thuật toán đều giống nhau nhưng thời gian thực hiện và bộ nhớ sử dụng là khác nhau. Kết quả sắp theo thời gian thực hiện (giây): *MSRIC-B > MSRIC-R > MSRIC-P;* bộ nhớ sử dụng (MB): *MSRIC-B > MSRIC-R > MSRIC-P.*

Các kết quả thực nghiệm đã chứng minh rằng việc đưa ràng buộc vào trong quá trình khai thác hiệu quả hơn so với đưa vào sau. Hơn nữa, thời gian khai thác khi đưa ràng buộc vào giai đoạn khai thác mẫu ít hơn nhiều so với đưa vào giai đoạn sinh luật. Điều này cho thấy hiệu quả của việc ứng dụng tập mẫu thỏa ràng buộc Itemset trong sinh luật có ràng buộc. Đó là, có thể sinh trực tiếp được tập luật thỏa ràng buộc Itemset ở vế trái từ tập mẫu thỏa ràng buộc Itemset, rút ngắn thời gian khai thác và bộ nhớ sử dụng so với phương pháp thông thường.

## 4.5 Kết chương

Như vậy, trong chương này luận án đã giải quyết bài toán khai thác luật có ràng buộc Itemset ở vế trái trên cơ sở kế thừa tập mẫu thỏa ràng buộc Itemset ở Chương 3 [CT2, CT3][3].

[3] [CT2] V. Trang, V. Bay, & L. Bac (2014), "IMSR_PreTree: an improved algorithm for mining sequential rules based on the prefix-tree", *Vietnam Journal Computer Science*, vol. 1(2), pp. 97-105 (Springer).

[CT3] V. Trang, & L. Bac (2020), "Mining sequential rules with itemset constraints", *Applied Intelligence* (Springer, SCI, Q2, IF= 2.882).

# CHƯƠNG 5. KHAI THÁC MẪU TRUY CẬP WEB DỰA TRÊN RÀNG BUỘC CHUỖI CON

## 5.1 Giới thiệu bài toán

Khai thác mẫu truy cập web (còn gọi là khai thác thói quen sử dụng web, khai thác web log) là một ứng dụng quan trọng của khai thác mẫu tuần tự, có liên quan đến việc tìm kiếm các mẫu điều hướng của người dùng trên hệ thống World Wide Web bằng cách rút trích những tri thức từ các truy cập web được ghi lại trong các tập tin log, ở đó các sự kiện có thứ tự trong mỗi chuỗi của CSDL là các trang web mà người dùng đã truy cập.

**Phát biểu bài toán:** Cho CSDL chuỗi truy cập web *WD*, tập mẫu ràng buộc $U = \{u_1, u_2 ... u_n\}$ và ngưỡng phổ biến tối thiểu *minSup* do người dùng chỉ ra. Bài toán khai thác mẫu truy cập web với ràng buộc chuỗi con là đi tìm tất cả mẫu phổ biến trong CSDL mà có chứa bất kỳ mẫu nào của tập *U* dưới dạng chuỗi con.

$$FCP = \{p | \ sup(p) \geq minSup \wedge \exists k: 1 \leq k \leq n, p \supseteq u_k\}.$$

## 5.2 Các nghiên cứu liên quan

Vì cấu trúc của mẫu truy cập web đơn giản hơn cấu trúc mẫu tuần tự nên ngoài những phương pháp khai thác mẫu tuần tự chung, có những phương pháp khai thác riêng dành cho loại dữ liệu này.

Pei và đồng sự (2000) đã đề xuất cấu trúc cây để lưu thông tin mẫu truy cập web, gọi tắt là cây-*WAP* và thuật toán *WAP-Mine*. *WAP-Mine* không tạo ra tập ứng viên khổng lồ như Apriori nhưng phải dựng rất nhiều cây *WAP* trung gian trong suốt quá trình khai thác, tức là nó vẫn tiêu tốn khá nhiều thời gian và bộ nhớ. Một số nghiên cứu cải tiến từ cây *WAP* bao gồm cây *PLWAP* (Lu & Ezeife, 2003), *FLWAP-tree* (Tang, Turkia, & Gallivan, 2007) và cây *AWAPT* (Vijayalakshmi, Mohan, & Suresh, 2010).

Nhìn chung các thuật toán theo hướng tiếp cận dùng cây *WAP* tối ưu về thời gian và bộ nhớ hơn so với các phương pháp Apriori, song lại không hiệu quả bằng các phương pháp định dạng CSDL theo chiều dọc, do đó không còn thu hút nghiên cứu trong thời gian gần đây.

### 5.3 Phương pháp đề xuất

Đề xuất hai thuật toán *MWAPC* và *EMWAPC* sử dụng cấu trúc dữ liệu cây tiền tố *PreWAP*. Trong đó thuật toán đóng góp chính là *EMWAPC* cải tiến từ *MWAPC* bằng cách vận dụng các tính chất của *DBVP* và cây *PreWAP* để rút ngắn thời gian khai thác và bộ nhớ sử dụng.

Tiến trình khai thác xuất phát từ mỗi cây con có gốc tại mỗi *atom* trong $F_1$, *EMWAPC* tỉa không gian tìm kiếm của cây *PreWAP* ngay từ đầu trước khi thực hiện mở rộng mẫu nhờ kỹ thuật loại trừ sớm. Sau đó, trong quá trình mở rộng mẫu để tạo mẫu ứng viên mới, thay vì phải kiểm tra ràng buộc cho mỗi ứng viên mới như *MWAPC*, *EMWAPC* có thể bỏ qua bước kiểm tra này cho một số lượng lớn ứng viên nhờ kỹ thuật kiểm tra ràng buộc. Chi tiết của thuật toán *EMWAPC* được mô tả dưới đây.

***Bảng5.1.*** *Thuật toán EMWAPC.*

**Thuật toán *EMWAPC***

**Đầu vào:** *WD*, *minSup*, tập ràng buộc $U = \{u_1, u_2 \ldots u_n\}$

**Đầu ra:** *FCP* (tập các mẫu truy cập web thỏa *minSup* và *U*).

1. $FCP = \emptyset$;
2. Duyệt *WD* để tìm $F_1$ *mẫu-1* cùng với *DBVP* của chúng;
3. Tìm $U' = \{u_i \in U \mid sup(u_i) \geq minSup\}$ bằng cách tính $DBVP_{ui}$, $\forall u_i \in U$;
4. $F_1^*$ = Gọi **EARLY-PRUNING($F_1$, U', minSup);**

```
5. For each node r in in F1* do
6.     If (r.label  thỏa 1 ràng buộc u ∈ U') then
7.         FCP = FCP ∪ {r.label};
8.         Gọi EXTENSION (r, F1, minSup);
9.     If (r.label không thỏa mọi ràng buộc u∈ U') then
10.        Gọi EXTENSION-CHECK (r, F1, minSup, U');
Thủ tục EXTENSION (r, I, minSup)
11. Lấy I1 = {e ∈ I| sup(đặt pe = Pattern –
    Extension(p, e)) ≥ minSup};
12. For each item e in I1 do
13.    FCP = FCP ∪ {pe.label};
14.    Gọi EXTENSION(pe, I1, minSup);
Thủ tục EXTENSION-CHECK(r, I, minSup, U')
15. Lấy I1 = {e ∈ I| sup(đặt pe = Pattern –
    Extension(p, e)) ≥ minSup};
16. For each item e in I1 do
17.    If (pe.label thỏa 1 ràng buộc u ∈ U') then
18.        FCP = FCP ∪ {pe.label};
19.        Gọi EXTENSION (pe, I1, minSup);
20.    If (pe.label không thỏa mọi ràng buộc u∈ U')
    then
21.        Gọi EXTENSION-CHECK (pe, I1, minSup, U');
```

## 5.4 Kết quả thực nghiệm

**Thuật toán:** so sánh các thuật toán đề xuất gồm *MWAPC* và *EMWAPC* (đưa ràng buộc vào trong quá trình khai thác mẫu) với *PRISMC* và *CM-SPAMC* (đưa ràng buộc vào sau quá trình khai thác).

**Cơ sở dữ liệu:**

| CSDL | #chuỗi | #item phân biệt | Độ dài chuỗi trung bình |
|---|---|---|---|
| Gazelle | 59,602 | 497 | 2.51 (std = 4.85) |
| FIFA | 20,450 | 2,990 | 34.74 (std = 24.08) |
| Kosarak10k | 10,000 | 10,094 | 8.14 (std = 22) |

**Kết quả thực nghiệm:** so sánh thời gian thực hiện và bộ nhớ sử dụng với hai tham số *minSup* và *Length* thay đổi.

*Về thời gian:* Các kết quả thực nghiệm cho thấy rằng *CM-SPAMC* chạy nhanh hơn *PRISMC* trên CSDL Kosarak nhưng chậm trên Gazelle và FIFA. Kết quả này là do các item xuất hiện cùng nhau hầu như có mặt trong mọi chuỗi của CSDL Gazelle và FIFA nên *CM-SPAMC* ít có cơ hội để tỉa ứng viên. Đáng chú ý là cả hai thuật toán đề xuất *MWAPC* và *EMWAPC* đều chạy nhanh hơn *CM-SPAMC* và P*RISMC* trên tất cả các CSDL thực nghiệm. Đặc biệt, *EMWAPC* luôn chạy nhanh nhất.

*Về bộ nhớ:* Trên cả ba bộ dữ liệu, cả hai thuật toán đề xuất *MWAPC* và *EMWAPC* đều tốn ít bộ nhớ hơn, ta có tỉ lệ chênh lệch ít hơn 10 lần so với *PRISMC* và 100 lần với *CM-SPAMC.*

### 5.5 Kết chương

Chương này đã trình bày vấn đề khai thác mẫu truy cập web với ràng buộc chuỗi con và đề xuất hai thuật toán có tên *MWAPC* và *EMWAPC* để giải quyết vấn đề [CT4][4]. Trong đó, thuật toán đóng góp chính là *EMWAPC* phát triển dựa trên cơ sở lý thuyết của ba mệnh đề có thể tỉa nhanh không gian tìm kiếm và giảm thiểu việc kiểm tra ràng buộc.

[4] [CT4] V. Trang, A Yoshitaka, & L. Bac (2018), "Mining web access patterns with supper-pattern constraints", *Applied Intelligence*, vol. 48(11), pp. 3902-3914 (Springer, SCI, Q2, IF= 2.882).

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 1. Kết luận

Luận án đã trình bày tổng quan, tìm hiểu cơ sở lý thuyết về khai thác mẫu tuần tự phổ biến dựa trên các ràng buộc. Trong đó, đi sâu vào nghiên cứu vấn đề khai thác mẫu tuần tự với hai loại ràng buộc là ràng buộc Itemset và ràng buộc chuỗi con, và ứng dụng tập mẫu thỏa ràng buộc trong khai thác luật tuần tự có ràng buộc. Bên cạnh đó, luận án nghiên cứu lĩnh vực ứng dụng cụ thể của mẫu tuần tự là khai thác sử dụng web theo ràng buộc của người dùng. Luận án đã hoàn thành được các mục tiêu ban đầu là đề xuất các phương pháp khai thác hiệu quả cho các bài toán đặt ra sao cho có thể tìm được trực tiếp tập mẫu thỏa ràng buộc một cách chính xác, rút ngắn thời gian khai thác và giảm bộ nhớ sử dụng. Luận án đạt được các kết quả như sau.

(1) Đề xuất thuật toán khai thác mẫu tuần tự dựa trên ràng buộc Itemset: thuật toán *MSPIC-DBV*. Thuật toán mở rộng và phát triển cách tổ chức dữ liệu biểu diễn dọc - đề xuất cấu trúc *DBVP* làm đại diện biểu diễn lại CSDL theo chiều dọc nhờ vậy chỉ duyệt CSDL một lần. Bằng cách sử dụng cấu trúc cây tiền tố kết hợp *DBVP* để lưu không gian tìm kiếm, thuật toán đưa ra kỹ thuật tỉa không gian con theo tiền tố và kỹ thuật kiểm tra ràng buộc theo tiền tố có thể bỏ qua việc kiểm tra ràng buộc cho một số lượng lớn các mẫu ứng viên.

(2) Đề xuất thuật toán khai thác luật tuần tự thỏa ràng buộc Itemset ở vế trái của luật gồm bộ ba thuật toán *MSRIC-B, MSRIC-R,* và *MSRIC-P*. Trong đó *MSRIC-P* là thuật toán đóng góp chính, sử dụng tập mẫu thỏa ràng buộc Itemset sinh trực tiếp ra được các luật thỏa ràng buộc Itemset ở vế trái mà không cần kiểm tra ràng buộc như hai thuật toán kia.

(3) Đề xuất thuật toán khai thác mẫu truy cập web dựa trên ràng buộc chuỗi con gồm *MWAPC* và *EMWAPC*. Trong đó, thuật toán đóng góp chính là *EMWAPC* sử dụng cấu trúc dữ liệu và các kỹ thuật tương tự phương pháp khai thác mẫu với ràng buộc Itemset. Tuy nhiên, dựa vào đặc điểm của mẫu truy cập web, thuật toán thực hiện tỉa nhanh không gian tìm kiếm ngay từ đầu và giảm thiểu việc kiểm tra ràng buộc dựa vào đặc điểm của ràng buộc chuỗi con.

**2. Hướng phát triển**

Tiếp tục phát triển các chiến lược tỉa không gian tìm kiếm hiệu quả cho bài toán khai thác mẫu tuần tự có ràng buộc để các thuật toán đạt tốc độ và bộ nhớ tối ưu hơn. Nghiên cứu các giải pháp song song hóa dựa trên kiến thức đa lõi, *spark*.

Nghiên cứu khai thác mẫu tuần tự có ràng buộc trên CSDL phân tán, nhằm tìm cách xử lý hiệu quả cho các CSDL cực lớn với chuỗi dữ liệu dài. Trong lĩnh vực khai thác thói quen sử dụng web, có thể áp dụng khai thác phân tán để khai thác web log bị phân tán trên nhiều server.

Nghiên cứu áp dụng các kỹ thuật đề xuất cho vấn đề khai thác mẫu tuần tự với các loại ràng buộc khác như: ràng buộc trong việc kết hợp các sự kiện của mẫu, ràng buộc thời gian.

## CÔNG TRÌNH KHOA HỌC CỦA TÁC GIẢ

[CT1] V. Trang, V. Bay, & L. Bac (2018), “Mining sequential patterns with itemset constraints”, *Knowledge and Information Systems*, vol. 57(2), pp. 311-330 (Springer, SCIE, Q1, IF=2.397).

[CT2] V. Trang, V. Bay, & L. Bac (2014), “IMSR_PreTree: an improved algorithm for mining sequential rules based on the

prefix-tree", *Vietnam Journal Computer Science*, vol. 1(2), pp. 97-105 (Springer).

[CT3] V. Trang, & L. Bac (2020), "Mining sequential rules with itemset constraints", *Applied Intelligence* (Springer, SCI, Q2, IF= 2.882) (Accepted).

[CT4] V. Trang, A Yoshitaka, & L. Bac (2018), "Mining web access patterns with supper-pattern constraints", *Applied Intelligence*, vol. 48(11), pp. 3902-3914 (Springer, SCI, Q2, IF= 2.882).

## CÔNG TRÌNH KHOA HỌC CÓ LIÊN QUAN

[CT5] V. Trang, V. Bay, & L. Bac (2011), "Mining sequential rules based on prefix-tree", ACIIDS 2011, Daegu, Korea, SCI Vol. 351, 147-156 (Springer).

[CT6] H. Bao Huynh, T. Cuong, H. Huy, V. Trang, V. Bay Vo, & Vaclav Snasel (2018). "An efficient approach for mining sequential patterns using multiple threads on very large databases". Engineering Applications of Artificial Intelligence, 74, 242-251.