TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ MÔN DỰ ÁN**

**CÔNG NGHỆ THÔNG TIN 2**

**TÌM HIỂU LUẬT TUẦN TỰ TRONG KHAI THÁC DỮ LIỆU**

*Người hướng dẫn*: **TS DOÃN XUÂN THANH**

*Người thực hiện*: **LÊ MINH HIẾU – 51702016**

**NGUYỄN TRƯỜNG GIANG - 51702093**

Lớp **: 17050201**

Khoá  **: 21**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ MÔN DỰ ÁN**

**CÔNG NGHỆ THÔNG TIN 2**

**TÌM HIỂU LUẬT TUẦN TỰ TRONG KHAI THÁC DỮ LIỆU**

Người hướng dẫn: **TS DOÃN XUÂN THANH**

Người thực hiện: **LÊ MINH HIẾU**

**NGUYỄN TRƯỜNG GIANG**

Lớp **: 17050201**

Khoá  **: 21**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

LỜI CẢM ƠN

Chúng em xin chân thành gửi lời cảm ơn đến toàn thể Ban giám hiệu Đại học Tôn Đức Thắng và tập thể thầy cô Khoa Công nghệ Thông tin đã giúp đỡ và tạo mọi điều kiện trong suốt quá trình học tập tại trường và cũng như trong quá trình hoàn thành đồ án môn Dự án công nghệ thông tin 2.

Em xin cảm ơn đến giảng viên hướng dẫn dự án là Thầy TS. Doãn Xuân Thanh với những kinh nghiệm và kiến thức chuyên môn cao đã hết lòng giúp đỡ, bảo ban và động viên để bọn em hoàn thành tốt được đồ án tốt nghiệp này.

Em cũng xin được chân thành được cảm ơn các thầy cô trong hội đồng chấm thi đã có những đóng góp để em hoàn thiện đồ án này một cách tốt nhất.

Cảm ơn các bạn, các anh chị trong khoa đã giúp đỡ em rất nhiều trong quá trình tìm hiểu tài liệu cũng như có các ý kiến đóng góp để bài đồ án trở nên hoàn thiện hơn.

Xin cảm ơn bố mẹ, anh chị trong gia đình đã luôn ở bên động viên việc học tập để hoàn thành đồ án quan trọng nhất trong cuộc đời sinh viên.

Trong quá trình làm đồ án chắc chắn vẫn còn mắc phải nhiều sai sót, bọn em rất mong thầy cô bỏ qua. Đồng thời, với kiến thức và kinh nghiệm còn hạn chế bọn em xin sự đóng góp từ thầy cô để đồ án của bọn em được hoàn thiện hơn.

Chúng em xin chân thành cảm ơn!

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Chúng em xin cam đoan đây là sản phẩm đồ án của riêng chúng em và được sự hướng dẫn của Thầy TS. Doãn Xuân Thanh. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 12 tháng 11 năm 2021*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Lê Minh Hiếu*

*Nguyễn Trường Giang*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

Doãn Xuân Thanh

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Trong thời kì công nghệ phát triển, khối lượng thông tin ngày càng trở nên đồ sộ thì việc khai thác dữ liệu cũng ngày càng trở nên cần thiết trong nhiều lĩnh vực thực tế mà đặc biệt là khai thác dữ liệu áp dụng luật tuần tự. Luật tuần tự là một ứng dụng của học máy (Machine Learning) và khai thác dữ liệu – khai phá tri thức (Datamining). Luật tuần tự ứng dụng được trong nhiều lĩnh vực như khai thác hành vi sử dụng web, y tế, sinh học, phân tích tế bào, …

Khai thác luật tuần tự phát triển từ khai thác mẫu tuần tự. Việc khai thác mẫu tuần tự để đạt được kết quả tốt nhất phải lựa chọn được thuật toán khai thác mẫu tuần tự phù hợp với cơ sở dữ liệu nhất. Để chọn được thuật toán phù hợp ta phải cần xác định được mục đích của cơ sở dữ liệu là gì để có thể thỏa mãn được độ hỗ trợ cũng như độ tin cậy của dữ liệu.

Để khai thác được dữ liệu ta sẽ trải qua nhiều bước như gom dữ liệu, trích lọc dữ liệu, tiền xử lý, chuyển đổi dữ liệu sau đó mới khai phá và đánh giá luật.

Vì vậy, nội dung chính của đồ án này là tìm hiểu thế nào là khai thác dữ liệu, sau đó tìm hiểu luật tuần tự trong khai thác dữ liệu và tìm hiểu cơ sở lý thuyết của các thuật toán khai thác mẫu tuần tự cùng với các ví dụ cụ thể cho từng thuật toán đó. Đồng thời, cũng tìm hiểu khái niệm về khai thác hành vi sử dụng web, tầm quan trọng của nó. Sau đó là tìm hiểu về weblog cũng như cấu trúc của một weblog.

Vì đây là đề tài mang tính chất tìm hiểu và nghiên cứu nên chắc chắn bọn em sẽ còn nhiều sai sót. Bọn em mong thầy cô góp ý để cải thiện chất lượng đồ án này ạ!

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc95406398)

[TÓM TẮT iv](#_Toc95406399)

[MỤC LỤC 1](#_Toc95406400)

[DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT 5](#_Toc95406401)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 7](#_Toc95406402)

[CHƯƠNG 1 – GIỚI THIỆU TỔNG QUAN 10](#_Toc95406403)

[4.1 Tổng quan 10](#_Toc95406404)

[4.2 Giới thiệu 10](#_Toc95406405)

[4.3 Tổng quan về khai thác dữ liệu 11](#_Toc95406406)

[4.4 Tổng quan về khai thác cơ sở dữ liệu chuỗi 14](#_Toc95406407)

[1.4.1 Đặc điểm dữ liệu chuỗi 14](#_Toc95406408)

[1.4.2 Các loại chuỗi trong dữ liệu chuỗi 15](#_Toc95406409)

[1.4.3 Các kỹ thuật khai thác mẫu tuần tự 15](#_Toc95406410)

[1.4.4 Các kỹ thuật khai thác dữ liệu 16](#_Toc95406411)

[1.4.5 Luật tuần tự trong khai thác luật 17](#_Toc95406412)

[4.5 Những khó khăn và thách thức trong khai thác dữ liệu 18](#_Toc95406413)

[4.6 Nội dung bài báo cáo 19](#_Toc95406414)

[CHƯƠNG 2 – TỔNG QUAN KHAI THÁC LUẬT TUẦN TỰ 20](#_Toc95406415)

[2.1 Ý nghĩa luật tuần tự 20](#_Toc95406416)

[2.1.1 Giới thiệu 20](#_Toc95406417)

[2.1.2 Ý nghĩa luật tuần tự 20](#_Toc95406418)

[2.2 Bài toán khai thác mẫu tuần tự 21](#_Toc95406419)

[2.2.1 Khai thác mẫu tuần tự 21](#_Toc95406420)

[2.2.2 Khai thác mẫu tuần tự đóng 21](#_Toc95406421)

[2.2.3 Khai thác mẫu tuần tự liên chuỗi 22](#_Toc95406422)

[2.2.4 Khai thác luật tuần tự 23](#_Toc95406423)

[2.2.5 Hạn chế của khai thác mẫu tuần tự 23](#_Toc95406424)

[2.3 Các khái niệm 24](#_Toc95406425)

[2.4 Khái niệm về luật tuần tự 26](#_Toc95406426)

[2.4.1 Các cách tổ chức dữ liệu 27](#_Toc95406427)

[2.4.2 Bài toán khai thác luật tuần tự 28](#_Toc95406428)

[2.5 Các thuật toán khai thác mẫu tuần tự 31](#_Toc95406429)

[2.5.1 Apriori 32](#_Toc95406430)

[2.5.2 AprioriAll 36](#_Toc95406431)

[2.5.3 GSP – Generalized Sequential Pattern 37](#_Toc95406432)

[2.5.4 PSP 42](#_Toc95406433)

[2.5.5 WAP-mine 42](#_Toc95406434)

[2.5.6 FreeSpan (Frequent Pattern-Projected Sequential pattern mining), PrefixSpan (Prefix-Projected Sequential pattern mining) 43](#_Toc95406435)

[2.5.7 SPADE – Sequential Pattern Discovery using Equivalence classes 49](#_Toc95406436)

[2.5.8 SPAM (Sequential Pattern Mining) 52](#_Toc95406437)

[2.5.9 PRISM – Prime Encoding Based Sequence Mining 53](#_Toc95406438)

[2.6 Các thuật toán khai thác mẫu tuần tự đóng 54](#_Toc95406439)

[2.6.1 CloSpan – Closed sequential pattern mining 54](#_Toc95406440)

[2.6.2 BIDE – BI-Directional Extension 55](#_Toc95406441)

[2.6.3 TSP 56](#_Toc95406442)

[2.6.4 Cấu trúc cây PDB 57](#_Toc95406443)

[2.7 Khai thác tập phổ biến với vector bit động 58](#_Toc95406444)

[2.7.1 Giới thiệu 58](#_Toc95406445)

[2.7.2 Cấu trúc DBV 59](#_Toc95406446)

[2.7.3 Thuật toán tìm phần giao giữa hai vector bit động 60](#_Toc95406447)

[2.7.4 Tính toán nhanh với độ hỗ trợ của một itemset từ một DBV 61](#_Toc95406448)

[2.8 Thuật toán BIDE 61](#_Toc95406449)

[2.8.1 Vấn đề đặt ra 61](#_Toc95406450)

[2.8.2 Một số định nghĩa 62](#_Toc95406451)

[2.8.3 BIDE 63](#_Toc95406452)

[2.8.4 Đánh giá hiệu suất 73](#_Toc95406453)

[2.8.5 Kết luận 84](#_Toc95406454)

[CHƯƠNG 3 – XÂY DỰNG CƠ SỞ LUẬT TUẦN TỰ CHO DỰ ĐOÁN HÀNH VI TRUY CẬP WEB 86](#_Toc95406455)

[3.1 Khái niệm dự đoán hành vi truy cập Web 86](#_Toc95406456)

[3.2 Giới thiệu Web Log 87](#_Toc95406457)

[3.3 Khái niệm Web Usage Mining 88](#_Toc95406458)

[3.3.1 Định nghĩa Web Usage Mining 88](#_Toc95406459)

[3.3.2 Tầm quan trọng của Web Usage Mining 89](#_Toc95406460)

[3.4 Khái niệm cơ sơ dữ liệu Web Log 89](#_Toc95406461)

[3.4.1 Định nghĩa cơ sở dữ liệu Web Log 89](#_Toc95406462)

[3.4.2 Cấu trúc nội dung của Web Log 89](#_Toc95406463)

[CHƯƠNG 4 – ỨNG DỤNG 94](#_Toc95406464)

[4.1 Giới thiệu 94](#_Toc95406465)

[4.2 Sản phẩm 94](#_Toc95406466)

[4.2.1 Giới thiệu sản phẩm 94](#_Toc95406472)

[4.2.2 Chức năng 94](#_Toc95406473)

[4.2.3 Công nghệ sử dụng 94](#_Toc95406474)

[4.2.4 94](#_Toc95406475)

[4.3 Đánh giá ứng dụng 95](#_Toc95406476)

[4.3.1 Kết quả đạt được 95](#_Toc95406477)

[4.3.2 Hạn chế 95](#_Toc95406478)

[CHƯƠNG 5 – KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 95](#_Toc95406479)

[5.1 Kết luận 95](#_Toc95406482)

[5.2 Hạn chế 96](#_Toc95406483)

[5.3 Hướng phát triển 96](#_Toc95406484)

[TÀI LIỆU THAM KHẢO 97](#_Toc95406485)

# DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT

**CÁC KÝ HIỆU**

|  |  |  |
| --- | --- | --- |
| TT | Ký hiệu | Diễn giải |
| 1 | Sup(f) | Support: độ hỗ trợ của f. |
| 2 | Conf(f) | Confident: độ tin cậy của f. |
| 3 | minSup(f) | Minimum Support: Ngưỡng hỗ trợ tối thiểu của f. |
| 4 | minConf(f) | Minimum Confident: Ngưỡng tin cậy tối thiểu của f. |
| 5 | Sid | Sequence Identification – Định danh của chuỗi. |
| 6 | S | Sequence – Chuỗi. |
| 7 | f-list | Danh sách các item phổ biến. |
| 8 | S-Matrix | Ma trận độ hỗ trợ. |
| 9 | <{a} {b} {c}>, | Chuỗi tuần tự có ba phần tử a, b, c. |
| 10 | {a,b,c,d} | Tập hợp có bốn phần tử a, b, c, d. |
| 11 | k-item | Số lượng các phần tử có trong chuỗi. |
| 12 | k-itemset | Số lượng các tập chuỗi trong chuỗi. |
| 13 | X 🡪 Y | X xuất hiện trước Y. |
| 14 | X ⊆ Y | X là tập hợp con của Y. |
| 15 | (X++Y) | Kết hợp tiền tố và hậu tố. |
| 16 | Sup(X++Y) | Độ hỗ trợ của tiền tố X và hậu tố Y kết hợp. |
| 17 | Sup(X) | Độ hỗ trợ của tiền tố X. |
| 18 | Conf(X++Y) | Độ tin cậy của tiền tố X và hậu tố Y kết hợp. |
| 19 | SIDList(s) | Tập hợp các chuỗi trong cơ sở dữ liệu D có chứa s. |
| 20 | SIDSum(s) | Tổng của SIDList(s). |
| 21 | . | Độ hỗ trợ tối thiểu trong ngưỡng. |

**CÁC CHỮ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| TT | Chữ viết tắt | Diễn giải |
| 1 | IoT | Internet of Things – Mạng lưới Internet vạn vật. |
| 2 | AI | Artificial Intelligence – Trí tuệ nhân tạo. |
| 3 | DNA | DeoxyriboNucleic Acid. |
| 4 | RNA | RiboNucleic Acid. |
| 5 | CSDL | Cơ sở dữ liệu. |
| 6 | SDB | Sequence DataBase – Cơ sở dữ liệu chuỗi. |
| 7 | DL | Dữ liệu. |
| 8 | KPDL | Khai phá dữ liệu. |
| 9 | DFS | Depth First Search – Tìm kiếm theo chiều sâu. |

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 1: Bài toán khai thác chuỗi phổ biến 11](#_Toc90565104)

[Hình 2: Quá trình khai thác dữ liệu 12](#_Toc90565105)

[Hình 3: Quá trình khai phá dữ liệu 13](#_Toc90565106)

[Hình 4: Ví dụ về bài toán WAP-mine 43](#_Toc90565107)

[Hình 5: Ví dụ về bài toán BIDE 56](#_Toc90565108)

[Hình 6: Ví dụ về cấu trúc cây PDB 58](#_Toc90565109)

[Hình 7: Vecto bit với 40 byte 59](#_Toc90565110)

[Hình 8: Vecto bit đã tính ra giá trị thập phân 59](#_Toc90565111)

[Hình 9: Phần giao giữa hai DBV 60](#_Toc90565112)

[Hình 10: Bảng tra cứu bitvector 61](#_Toc90565113)

[Hình 11: Minh họa BIDE 64](#_Toc90565114)

[Hình 12: Mã giả thuật toán BIDE 72](#_Toc90565115)

[Hình 13: So sánh thời gian thực thi giữa BIDE, CloSpan , SPADE và PrefixSpan trên dữ liệu Gazelle. 75](#_Toc90565116)

[Hình 14: So sánh khả năng sử dụng bộ nhớ giữa BIDE, CloSpan , SPADE và PrefixSpan trên dữ liệu Gazelle. 75](#_Toc90565117)

[Hình 15: Sự phân bố trên dữ liệu Gazelle 76](#_Toc90565118)

[Hình 16: Thời gian chạy giữa BIDE và CloSpan trên dữ liệu Gazelle. 77](#_Toc90565119)

[Hình 17: Khả năng sử dụng bộ nhớ giữa BIDE và CloSpan trên dữ liệu Gazelle. 78](#_Toc90565120)

[Hình 18: Phân bố số lượng chuỗi đóng trong dữ liệu Snake. 78](#_Toc90565121)

[Hình 19: Thời gian chạy giữa BIDE và CloSpan trên dữ liệu Snake. 79](#_Toc90565122)

[Hình 20: Khả năng sử dụng bộ nhớ giữa BIDE và CloSpan trên dữ liệu Snake. 79](#_Toc90565123)

[Hình 21: Phân bố số lượng chuỗi đóng trong dữ liệu Pi. 80](#_Toc90565124)

[Hình 22: Thời gian chạy giữa BIDE và CloSpan trên dữ liệu Pi. 80](#_Toc90565125)

[Hình 23: Khả năng sử dụng bộ nhớ giữa BIDE và CloSpan trên dữ liệu Pi. 81](#_Toc90565126)

[Hình 24: Khả năng mở rộng của BIDE về thời gian chạy. 82](#_Toc90565127)

[Hình 25: Khả năng mở rộng của BIDE về mức độ sử dụng bộ nhớ. 82](#_Toc90565128)

[Hình 26: Hiệu quả của phương pháp cắt tỉa BackScan trên dữ liệu Gazelle 83](#_Toc90565129)

[Hình 27: Hiệu quả của kỹ thuật ScanSkip trên dữ liệu Snake 83](#_Toc90565130)

[Hình 28: Một weblog đầy đủ 91](#_Toc90565131)

[Hình 29: Ý nghĩa các cột trong weblog. 92](#_Toc90565132)

[Hình 30: Dữ liệu weblog của cửa hàng bán bóng bay. 92](#_Toc90565133)

**DANH MỤC BẢNG**

[Bảng 1: Tổ chức dữ liệu theo chiều ngang. 27](#_Toc90565134)

[Bảng 2: Tổ chức dữ liệu theo chiều dọc 27](#_Toc90565135)

[Bảng 3: Cơ sở dữ liệu chuỗi với minSup = 2. 28](#_Toc90565136)

[Bảng 4: Cơ sở dữ liệu chuỗi với minSup = 50% và minConf = 70% 29](#_Toc90565137)

[Bảng 5: Bảng các mẫu tuần tự thỏa minSup 29](#_Toc90565138)

[Bảng 6: Bảng các luật tuần tự thỏa minConf. 31](#_Toc90565139)

[Bảng 7: Bảng cơ sở dữ liệu ví dụ cho thuật toán Apriori. 33](#_Toc90565140)

[Bảng 8: Bảng k = 1 thỏa minSup. 33](#_Toc90565141)

[Bảng 9: Bảng k = 2 thỏa minSup. 33](#_Toc90565142)

[Bảng 10: Bảng k = 3 thỏa minSup. 33](#_Toc90565143)

[Bảng 11: Bảng các itemset thỏa mãn minSup và minConf. 34](#_Toc90565144)

[Bảng 12: Bảng cơ sở dữ liệu ví dụ cho thuật toán GSP. 38](#_Toc90565145)

[Bảng 13: Bảng ứng viên đầu tiên. 38](#_Toc90565146)

[Bảng 14: Bảng ứng viên thứ hai có chiều dài là 2 và có 2 phần tử. 39](#_Toc90565147)

[Bảng 15: Bảng ứng viên thứ hai có chiều dài là 2 và có 1 phần tử. 39](#_Toc90565148)

[Bảng 16: Bảng ứng viên thứ ba. 41](#_Toc90565149)

[Bảng 17: Bảng cơ sở dữ liệu ví dụ cho thuật toán PrefixSpan. 45](#_Toc90565150)

[Bảng 18: Bảng cơ sở dữ liệu chiếu theo 46](#_Toc90565151)

[Bảng 19: Bảng cơ sở dữ liệu chiếu theo 46](#_Toc90565152)

[Bảng 20: Bảng cơ sở dữ liệu chiếu theo . 47](#_Toc90565153)

[Bảng 21: Bảng cơ sở dữ liệu chiếu theo . 47](#_Toc90565154)

[Bảng 22: Bảng cơ sở dữ liệu thực hiện PrefixSpan. 48](#_Toc90565155)

[Bảng 23: Bảng cơ sở dữ liệu ví dụ cho thuật toán SPADE 50](#_Toc90565156)

[Bảng 24: Tổ chức lại cơ sở dữ liệu theo chiều dọc 50](#_Toc90565157)

[Bảng 25: Bảng mẫu tuần tự có độ dài là 1 51](#_Toc90565158)

[Bảng 26: Bảng mẫu tuần tự có độ dài là 2 51](#_Toc90565159)

[Bảng 27: Bảng mẫu tuần tự có độ dài k từ hai mẫu độ dài (k-1) có (k-2) item đầu giống nhau 51](#_Toc90565160)

[Bảng 28 : Bảng cơ sở dữ liệu ví dụ của thuật toán SPAM 52](#_Toc90565161)

[Bảng 29: Bảng đã biến đổi sang dạng nhị phân trong SPAM 53](#_Toc90565162)

[Bảng 30: Bảng cơ sở dữ liệu ví dụ cho thuật toán BIDE. 56](#_Toc90565163)

[Bảng 31: Cơ sở dữ liệu ví dụ 63](#_Toc90565164)

[Bảng 32: Bảng cơ sở dữ liệu ví dụ cho kỹ thuật ScanSkip 71](#_Toc90565165)

[Bảng 33: Bảng chi tiết dữ liệu. 74](#_Toc90565166)

[Bảng 34: Bảng ví dụ weblog 87](#_Toc90565167)

CHƯƠNG 1 – GIỚI THIỆU TỔNG QUAN

* 1. Tổng quan

Hiện nay, do sự phát triển nhanh chóng và liên tục đổi mới của Công nghệ thông tin và Truyền thông, khối lượng các dữ liệu khai thác được từ các ứng dụng (website, ứng dụng điện thoại…) ngày càng trở nên đồ sộ. Việc cung cấp các yêu cầu phù hợp cho khách hàng có nhu cầu ngày càng trở nên cần thiết. Trong đó các ứng dụng về khai phá dữ liệu mà nổi bật là áp dụng luật tuần tự để khai phá dữ liệu trở nên phổ biến bậc nhất với việc khai thác các dữ liệu để dự đoán hành vi truy cập Web của khách hàng. Ngày nay, việc dự đoán và phân tích hành vi truy cập web trở nên phổ biến với các kênh thương mại điện tử lớn (Shoppe, Lazada, Tiki), các ứng dụng trên điện thoại (GooglePlay, AppStore), các ứng dụng web và trong các hệ thống IoT. Điều này mang nhiều ý nghĩa to lớn đối với cả người sản xuất dịch vụ cũng như là người dùng sử dụng dịch vụ.

Áp dụng luật tuần tự và dự đoán dữ liệu là một ứng dụng quan trọng của học máy (machine learning) cũng như là khai thác dữ liệu và khai phá tri thức (data mining). Trong đó bao gồm việc khám phá các quy tắc theo trình tự. Phần lớn mục đích của việc áp dụng luật tuần tự này là để khai thác được hành vi sử dụng web của người dùng để hỗ trợ ra gợi ý và dự đoán được thói quen, sở thích và dự định của người dùng để các người làm dịch vụ có thể nắm bắt được nhu cầu của khách hàng nói riêng và xu thế của thị trường nói chung.

* 1. Giới thiệu

Với sự đa dạng và phong phú của dữ liệu hình thành nên các mô hình dữ liệu khác nhau: mô hình dữ liệu chuỗi (sequence), mô hình dữ liệu giao dịch (transaction), mô hình dòng dữ liệu (data stream), chuỗi thời gian (time-series), …

Việc khai thác hành vi sử dụng web của người dùng được khai thác bằng cách trích xuất từ các tập dữ liệu của máy chủ web, dữ liệu được khám phá trong các tập dữ liệu thông thường là tập hợp các trình tự được sử dụng thường xuyên bởi người dùng với các sản phẩm họ quan tâm. Mẫu dữ liệu được tạo ra sẽ giữ lại chuỗi thông tin trong điều hướng. Trong khai thác dữ liệu có nhiều cách để khai thác dữ liệu, nhưng trong lĩnh vực khai thác hành vi sử dụng web của người dùng thì khai thác áp dụng luật tuần tự là phù hợp nhất. Trong khai thác quy tắc tuần tự, đầu vào sẽ là trình tự truy cập web và tạo ra đầu ra dưới dạng tuần tự để xác định được khách hành truy cập vào mục nào nhiều.

Khai phá luật tuần tự là một biến thể của khai thác mẫu tuần tự. Mẫu tuần tự là một dãy con xuất hiện trong tập dữ liệu. Ví dụ: ta có một mẫu tuần tự <{a} {b} {c}>, điều này chỉ ra rằng {a} là sản phẩm đã mua, {b} là sản phẩm mua sau đó, {c} là sản phẩm mua sau khi mua {a} và {b}. Một số thuật toán được áp dụng phổ biến trong khai thác dữ liệu tuần tự như Apriori, PRISM, SPADE, Prefix Span và GSP.

Diagram

Description automatically generated

Hình 1: Bài toán khai thác chuỗi phổ biến

* 1. Tổng quan về khai thác dữ liệu

Khai thác dữ liệu hay còn được gọi là khám phá tri thức trong cơ sở dữ liệu. Là quá trình khám phá những mẫu hay tri thức có ích từ các nguồn dữ liệu (cơ sở dữ liệu, văn bản, ảnh, weblog, …).

Các mẫu khai thác được phải có giá trị, dễ hiểu và hữu ích. Khai thác dữ liệu là một lĩnh vực đa ngành có liên quan đến học máy (Machine Learning), thống kê, cơ sở dữ liệu, trí tuệ nhân tạo (AI - Artificial Intelligence), thu thập thông tin, mô phỏng trực quan hóa dữ liệu…

Diagram

Description automatically generated

Hình 2: Quá trình khai thác dữ liệu

Quá trình khi thực hiện khai thác dữ liệu bao gồm:

* Gom dữ liệu: đây là bước đầu tiên, khai thác và tìm kiếm dữ liệu trong các cơ sở dữ liệu, một kho dữ liệu và các dữ liệu lấy từ các nguồn ứng dụng web.
* Trích lọc dữ liệu: dữ liệu sẽ được chọn hoặc tách ra dựa trên các tiêu chí cụ thể để thích hợp cho việc khai thác. Ví dụ: chọn tất cả nam sinh viên Công nghệ thông tin có điểm trung bình từ 7 trở lên.
* Làm sạch, tiền xử lý dữ liệu: giai đoạn này rất quan trọng nhưng thường bị sao lãng. Cơ sở dữ liệu sẽ luôn có những sai sót, chẳng hạn như dư thừa, không đủ chặt chẽ và thiếu logic. Vì vậy, dữ liệu sẽ có những giá trị vô nghĩa và không có khả năng kết nối dữ liệu.

Ví dụ: dữ liệu sinh viên có điểm trung bình là 12.6

* Chuyển đổi dữ liệu: ở giai đoạn này dữ liệu được tổ chức lại, chuyển đổi dữ liệu thành dạng phù hợp về dạng phù hợp bằng cách thực hiện nhóm hoặc tập hợp. Sau đó dữ liệu sẽ có thể sử dụng và điều khiển được.
* Khai phá dữ liệu: các thuật toán khai thác khác nhau được sử dụng để trích xuất các mẫu từ dữ liệu ở giai đoạn khai thác dữ liệu này.
* Đánh giá luật: tất cả các mẫu dữ liệu không phải cái nào cũng đều hữu ích, đôi khi còn có sai lệch. Vì vậy các tiêu chuẩn đánh giá được dùng để chiết xuất ra các tri thức (Knowledge) cần chiết xuất ra. Có thể đánh giá sự hữu ích của các mẫu dựa trên một số phép đo.

Khai phá dữ liệu là một quá trình lặp đi lặp lại cho đến khi thu được kết quả tốt nhất. Trong hầu hết các trường hợp, khai thác dữ liệu trong cơ sở dữ liệu được thực hiện theo ba giai đoạn:

* Tiền xử lý dữ liệu.
* Khai thác dữ liệu.
* Phân tích mẫu dữ liệu.

Quá trình khai thác dữ liệu được mô tả:

Diagram

Description automatically generated

Hình 3: Quá trình khai phá dữ liệu

* Xác định nhiệm vụ: xác định vấn đề cần phải được giải quyết.
* Xác định dữ liệu liên quan: để xây dụng giải pháp.
* Thu thập và tiền xử lý dữ liệu: là quá trình xử lý dữ liệu để làm cho thuật toán khai thác dữ liệu có thể hiểu được. Quá trình này có nhiều khó khăn và vướng mắt phải giải quyết như: dữ liệu được sao chép thành nhiều bản (nếu được chiết xuất vào các tệp), quản lý tệp dữ liệu và phải thực hiện lại quy trình tương tự khi dữ liệu mới được đưa vào ...
* Thuật toán khai phá dữ liệu: đây là quá trình lựa chọn thuật toán khai phá dữ liệu sao cho phù hợp và thực hiện khai thác các mẫu có ý nghĩa trong cơ sở dữ liệu, các mẫu này được biểu diễn theo dạng luật kết hợp, luật tuần tự, cây quyết định… tương ứng với ý nghĩa của nó.
  1. Tổng quan về khai thác cơ sở dữ liệu chuỗi

Khai thác luật tuần tự là một chủ đề nghiên cứu thiết thực, quan trọng và đang được sử dụng hiện nay trong lĩnh vực khai thác dữ liệu. Nhiệm vụ của nó là tìm ra các luật tìm ẩn trong cơ sở dữ liệu chuỗi, mỗi luật sẽ thể hiện mối quan hệ giữa các mẫu theo thứ tự thời gian.

* + 1. Đặc điểm dữ liệu chuỗi

Dữ liệu chuỗi mang các đặc điểm khác với các loại dữ liệu khác. Chính vì thế, dữ liệu chuỗi đặt ra nhiều cơ hội và thách thức, thu hút các nhà nghiên cứu. So với các dữ liệu khác, dữ liệu chuỗi có các đặc điểm:

* Kích thước chuỗi có thể rất dài. Các chuỗi trong cơ sở dữ liệu có thể có sự chênh lệch lớn.

Ví dụ: trong một cơ sở dữ liệu, các chuỗi có thể có độ dài là 4 5, nhưng cũng có thể có chuỗi có độ dài là 50 60.

* Một mẫu là chuỗi cha khi các thành phần trong chuỗi phải ở cùng nhau trong chuỗi ban đầu.
* Vị trí của các thành phần trong chuỗi không quan trọng.

Ví dụ: khi kiểm tra một chuỗi dữ liệu có chứa một mẫu hay không thì không cần quan tâm mẫu đó xuất hiện trong chuỗi ở vị trí nào.

* Mối quan hệ giữa các phần tử trong chuỗi và thứ tự (vị trí) của chúng là rất quan trọng. Điểm khác biệt của dữ liệu chuỗi so với các dữ liệu khác đó là mối quan hệ về thứ tự (vị trí) giữa các thành phần trong chuỗi là đặc điểm duy nhất đặc trưng duy nhất ở dữ liệu chuỗi.

Ví dụ: Chuỗi AB khác với chuỗi BA, trong chuỗi AB thì A và B là riêng biệt.

* + 1. Các loại chuỗi trong dữ liệu chuỗi

Trong các loại chuỗi thì chuỗi sự kiện là loại chuỗi chiếm phần lớn. Từ các chuỗi sự kiện, ta hiểu được cách thức hoạt động của các đối tượng trong chuỗi để tìm được cách giải quyết tốt nhất. Chuỗi sự kiện là chuỗi các lịch sử bán hàng, chuỗi truy cập web (weblog), chuỗi lịch sử mua sắm của khách hàng, chuỗi vết hệ thống… Các chuỗi sự kiện thường bao gồm bản ghi nhận các hoạt động thực hiện bởi người dùng, thời gian, địa điểm, các mặt hàng…

Bên cạnh chuỗi sự kiện còn có chuỗi dữ liệu sinh học giúp con người hiểu được cấu trúc và chức năng của các loại tế bào khác nhau, có ích trong việc chuẩn đoán và chữa bệnh. Có ba loại chuỗi sinh học là chuỗi deoxyribonucleic acid (DNA) và chuỗi amino acid (Protein) và ribonucleic acid (RNA). Các bài toán thường gặp với dữ liệu chuỗi sinh học là phân tích cấu trúc và chức năng của protein trong chuỗi protein, xác định mẫu của DNA, RNA, so sánh các chuỗi…

* + 1. Các kỹ thuật khai thác mẫu tuần tự

Khai thác mẫu tuần tự từ cơ sở dữ liệu bao gồm những hướng cụ thể sau:

* Khai thác mẫu tuần tự tổng quát.
* Khai thác mẫu tuần tự liên chuỗi.
* Khai thác mẫu tuần tự dựa trên ràng buộc.
* Khai thác mẫu tuần tự trên cơ sở dữ liệu tăng trưởng.
* Khai thác mẫu tuần tự gần đúng.
* Khai thác quan hệ thời gian.
* Khai thác mô hình tuần tự dài trong môi trường ồn ào.
* Khai thác theo chu kỳ khai thác quy tắc kết hợp
* Khai thác mẫu tuần tự một phần.
  + 1. Các kỹ thuật khai thác dữ liệu

Khai thác dữ liệu phụ thuộc rất nhiều vào loại tri thức mà hệ thống khai phá tri thức và khai thác dữ liệu đang muốn nhắm đến. Mỗi nhiệm vụ khai thác dữ liệu có những đặc điểm riêng. Các phương pháp khai thác dữ liệu thường xuyên được sử dụng:

* Khai thác chuỗi con phổ biến (mining frequent subsequence) hoặc khai thác mẫu tuần tự (mining sequential pattern): là khai thác các mẫu phổ biến có trong cơ sở dữ liệu có liên quan đến thời gian và các sự kiện khác, với yêu cầu là các mẫu phổ biến là chuỗi con trong cơ sở dữ liệu chuỗi phải thỏa *sup(f) minSup*, có nghĩa rằng sự xuất hiện của nó phải lớn hơn ngưỡng hỗ trợ mà người dùng đề ra.
* Phân loại và dự đoán: là quá trình tìm kiếm và xác định xem rằng một phần tử có thuộc một lớp được công nhận hay không. Các lớp thường được định nghĩa dựa trên giá trị của một trường nhất định trong một bản ghi hoặc dẫn xuất các giá trị khác nhau trong các trường dữ liệu. Các kỹ thuật học máy thường xuyên được sử dụng như cây quyết định, mạng nơ-ron nhân tạo và các kỹ thuật khác trong cách tiếp cận này. Học tập có giám sát (Supervised Learning) là tên gọi của loại vấn đề này.
* Phân cụm các chuỗi (cluster identification): các chuỗi được sắp xếp thành từng cụm trong quá trình khai thác, tuy nhiên số lượng và tên của các cụm chưa được cho biết trước. Để xác định các cụm, ta tính toán khoảng cách sao cho mức độ tương tự giữa các đối tượng trong cùng một cụm là lớn nhất và mức độ tương tự trong các cụm khác nhau là nhỏ nhất. Lớp bài toán này là được gọi là học không giám sát (Unsupervised Learning).
* Khai thác luật (mining rules): là quá trình xác định mối liên hệ về mặt thời gian giữa các sự kiện xảy ra theo một trình tự tuần tự. Luật mô tả mẫu tuần tự có dạng X🡪Y nói lên sự xuất hiện của X là tiền đề cho sự xuất hiện của Y.

Ngoài ra còn có các phương pháp khai thác dữ liệu như:

* Luật kết hợp (Association Rules).
* Hồi qui (Regression).
* Trực quan hóa (Visualization).
* Tổng hợp (Summarization).
* Mô hình ràng buộc (Dependency Modeling).
* Biểu diễn mô hình (Model Evaluation).
* Phân tích độ lệch và sự tăng trưởng (Evolution and Deviation Analyst).
* Phương pháp tìm kiếm (Search Method).
  + 1. Luật tuần tự trong khai thác luật

Quá trình khai thác luật tuần tự từ cơ sở dữ liệu được chia thành hai giai đoạn chính là giai đoạn khai thác mẫu tuần tự và giai đoạn sinh luật tuần tự.

Trong cơ sở dữ liệu chuỗi, có các loại luật khác nhau như:

* Luật tuần tự (sequential rules).
* Luật thú vị (interesting rules).
* Luật phân lớp theo tuần tự (sequential classification rules).
* Luật tuần hoàn (recurrent rules).

Luật tuần tự là mở rộng của mẫu tuần tự nhằm mở rộng khả năng sử dụng và tăng cường ý nghĩa của mẫu tuần tự. Các sự kiện xảy ra trước, theo luật tuần tự sẽ là tiền đề cho sự kiện xảy ra sau đi theo. Luật tuần tự hữu dụng trong việc dò và lọc ra những khác biệt như là dư thừa, dị thường do đó nó được áp dụng rộng rãi trong việc dò lỗi, phát hiện xâm nhập và bẫy lỗi… Luật tuần tự còn có ích trong lĩnh vực y dược và công nghệ phần mềm.

* 1. Những khó khăn và thách thức trong khai thác dữ liệu

Các khó khăn và thách thức thường được quan tân trong khai thác dữ liệu:

* Kích thước cơ sở dữ liệu rất lớn: cơ sở dữ liệu có hàng trăm bảng và trường và hàng triệu bản ghi. Kích thước lớn hơn nhiều gigabyte.
* Số chiều cao: các trường (thuộc tính, biến) với số lượng lớn làm cho số chiều của bài toán trở nên cao. Đồng thời, nó tạo cơ hội cho các giải thuật khai thác dữ liệu khác tìm ra các mẫu không hợp lệ.
* Thay đổi dữ liệu và tri thức: thay đổi trong dữ liệu là nguyên nhân dẫn đến các mẫu phát hiện từ trước đó trở nên không hợp lệ. Các biến đã đo trong một cơ sở dữ liệu ứng dụng nào đó cho trước cũng hoàn toàn có thể bị sửa đổi, xóa bỏ hay tăng thêm các phép đo mới. Cập nhật các mẫu và xử lý thay đổi bằng cách tăng trưởng là giải pháp hợp lý.
* Dữ liệu thiếu và bị nhiễu: Các thuộc tính quan trọng trong cơ sở dữ liệu sẽ có thể bị mất nếu cơ sở dữ liệu không được thiết kế với sự khám phá bằng trí tuệ một cách cụ thể, đặc biệt trong các cơ sở dữ liệu thương mại. Để giải quyết vấn đề này, có thể dùng các chiến lược thống kê phức tạp để nhận biết các biến ẩn và các biến phụ thuộc trong cơ sở dữ liệu.
* Mối quan hệ phức tạp giữa các trường: các thuộc tính hay các giá trị có cấu trúc phân cấp, mối quan hệ giữa các thuộc tính và các phương tiện tinh vi hơn cho việc biểu diễn tri thức về nội dung của một cơ sở dữ liệu sẽ đòi hỏi các giải thuật có khả năng sử dụng hiệu quả các thông tin này.
* Tính dễ hiểu của các mẫu: những gì khai thác được phải càng dễ hiểu với con người thì càng tốt thông qua việc minh họa bằng đồ thị, cấu trúc luật với đồ thị có hướng, biểu diễn thông qua ngôn ngữ tự nhiên và các kỹ thuật hình dung ra dữ liệu và tri thức.
* Người dùng tương tác và tri thức sẵn có: các phương pháp khám phá tri thức, các công cụ không thực sự tương tác với người dùng và không thể dễ dàng kết hợp chặt chẽ với các tri thức sẵn có khi thực hiện một bài toán loại trừ theo các cách đơn giản. Trong tất cả các bước xử lý khám phá tri thức, việc sử dụng miền tri thức là quan trọng.
* Tích hợp với các hệ thống khác: một hệ thống có thể không quá hữu ích khi phát hiện đứng một mình. Tích hợp với các bảng tính và các công cụ trực quan và điều tiết các dự đoán cảm biến thời gian thực là một trong những vấn đề tích hợp điển hình.
  1. Nội dung bài báo cáo

Bài báo cáo được chia ra làm 4 chương:

* Chương 1: giới thiệu tổng quan về khai thác dữ liệu và các dạng khai thác dữ liệu, luật tuần tự.
* Chương 2: cơ sở lý thuyết về luật tuần tự, các thuật toán khai thác mẫu tuần tự với các ví dụ minh họa.
* Chương 3: Khai thác hành vi sử dụng web, khái niệm và cấu trúc của weblog.
* Chương 4: Tổng kết, hạn chế và các hướng phát triển trong tương lai.

CHƯƠNG 2 – TỔNG QUAN KHAI THÁC LUẬT TUẦN TỰ

* 1. Ý nghĩa luật tuần tự
     1. Giới thiệu

Trong khai thác dữ liệu và khai phá tri thức, mà cụ thể là trong lĩnh vực khai thác dữ liệu trên cơ sở dữ liệu, khai thác mẫu tuần tự là bài toán đầu tiên được đề xuất bởi Agrawal và Srikant năm 1995. Khai thác mẫu tuần tự được áp dụng trong nhiều lĩnh vực như: dự đoán nhu cầu mua sắm của khách hàng, phân tích thị trường, khai thác truy cập web, ứng dụng trong y khoa, dự đoán thảm họa, nghiên cứu DNA.

Luật tuần tự sinh ra dựa trên mẫu tuần tự, luật tuần tự sẽ biểu diễn mối quan hệ giữa hai loạt sự kiện bao gồm loạt sự kiện đi trước và loạt sự kiện đi sau. So với mẫu tuần tự thì luật tuần tự mở rộng khả năng sử dụng và ý nghĩa biểu đạt của mẫu tuần tự đồng thời cũng thể hiện tri thức tiềm ẩn mà dữ liệu tuần tự có.

* + 1. Ý nghĩa luật tuần tự

Luật tuần tự được xem là cải tiến tự nhiên của mẫu tuần tự. Luật tuần tự biểu diễn các mối quan hệ của mẫu tuần tự theo thời gian. Dạng của luật tuần tự được thể hiện là X🡪Y, có nghĩa khi mẫu X xuất hiện thì mẫu Y cũng xuất hiện theo sau với độ tin cậy cao. Luật tuần tự giúp ta hiểu rõ hơn về thứ tự thời gian thể hiện trong một cơ sở dữ liệu chuỗi.

Ví dụ: một cửa hàng bán d, e, f; sau khi mua d khách hàng sẽ tiếp tục mua e và f. Do đó mẫu là (d, e, f)

Tuy nhiên một cửa hàng sẽ có nhiều khách hàng mua nên có thể mẫu (a, b, c) có xu hướng xuất hiện với độ hỗ trợ thấp. Tuy nhiên, với độ hỗ trợ thấp vẫn trả về các mẫu tuy nhiên sẽ có nhiều mẫu sai và không thích hợp. Do đó luật tuần tự được sinh ra để làm bỏ đi các mẫu sai hoặc dư thừa, không có nghĩa bằng cách ra khái niệm về độ tin cậy cho tập mẫu. Các mẫu được xem thỏa mãn luật tuần tự khi ngưỡng hỗ trợ và ngưỡng tin cậy phải thỏa mãn điều kiện *sup(f) minSup()* và *conf(f) minConf().*

* 1. Bài toán khai thác mẫu tuần tự
     1. Khai thác mẫu tuần tự

Khai thác mẫu tuần tự giúp tìm kiếm và khám phá những mẫu tuần tự nào xuất hiện thường xuyên trong cơ sở dữ liệu để tìm ra mối quan hệ giữa các sự kiện hoặc có thể là các sự kiện tiềm ẩn khác nhau để phục vụ cho các ứng dụng như tái tổ chức kinh doanh, tiếp thị, dự đoán, lập kế hoạch. Có hai bài toán khai thác mẫu tuần tự có hai dạng chính: khai thác mẫu tuần tự và khai thác mẫu tuần tự liên chuỗi.

Các tiếp cận chính của thuật toán:

* Tiếp cận theo hướng kết hợp – sinh Apriori:
  + AprioriAll (Agrawall và Srikant, 1995)
  + GSP (Agrawall và Srikant, 1996)
  + PSP (Masseglia và động sự, 1999)
  + SPAM (Ayers và đồng sự, 2002)
* Tiếp cận theo hướng phát triển mẫu:
  + FreeSpan (Han và đồng sự, 2000)
  + Wap-mine (Pei và đồng sự, 2000)
  + PrefixSpan (Pei và đồng dự, 2001)
  + FS-Miner (EI-Sayed và đồng sự, 2004)
  + SLPMiner
* Tiếp cận theo hướng tỉa ứng viên:
  + DISC-all (Chiu và đồng sự, 2004)
  + LAPIN (Yang và đồng sự, 2007)
* Kết hợp các ưu điểm của các tiếp cận trên:
  + SPADE (Zaki, 2001)
  + PLWAP (Ezeife và Lu, 2005)
  + PRISM (Gouda, Hassaan và Zaki, 2010)
    1. Khai thác mẫu tuần tự đóng

Khai thác mẫu tuần tự đóng giải quyết vần đề của phần lớn các thuật toán khai thác mẫu tuần tự là cố gắng tìm tất cả các mẫu tuần tự có trong cơ sở dữ liệu, do đó các mẫu không cần thiết sinh ra đẫn đến dư thừa.

Mẫu tuần tự Sa được xem là mẫu tuần tự đóng nếu mẫu Sa đó là phổ biến và không tồn tại mẫu tuần tự cha Sb nào của Sa có cùng độ hỗ trợ.

Các thuật toán khai thác của mẫu tuần tự đóng phổ biến:

* A-Close (Pasquier và đồng sự, 1999)
* CLOSET (Pei và đồng sự, 2000)
* CHARM (Zaki và Hsiao, 2002)
* CLOSET+ (WANG và đồng sự, 2003)
* CloSpan (Yan và đồng sự, 2003)
* BIDE (WANG và đồng sự, 2007)
* BIDE+ (WANG và đồng sự, 2007)
* ClaSP (Gomariz và đồng sự, 2013)

Thuật toán khai thác mẫu tuần tự đóng bắt buộc phải duy trì tập phổ biến đã được khai thác để kiểm tra mẫu nào là mẫu đóng trong tập ứng viên phổ biến vừa được sinh ra trước khi loại bỏ đi. Do vậy, nó sẽ tốn nhiều không gian lưu trữ trung gian trong quá trình khai thác.

* + 1. Khai thác mẫu tuần tự liên chuỗi

Khai thác mẫu tuần tự liên chuỗi là các bài toán mà trong đó quan tâm đến mối quan hệ giữa các mẫu tuần tự của các giao dịch ở các mốc thời điểm khác nhau, từ đó thể hiện mối quan hệ theo thứ tự thời gian giữa các chuỗi.

Thông thường, mẫu tuần tự liên chuỗi phức tạp và mang ý nghĩa tổng quát hơn nhiều so với mẫu tuần tự. Do trong quá trình khai thác mẫu tuần tự liên chuỗi đã bao gồm luôn việc khai thác mẫu tuần tự và kết hợp các mẫu tuần tự này theo thời gian khác nhau.

Một thuật toán tiêu biểu cho bài toán khai thác mẫu tuần tự liên chuỗi là EISP do Wang và Lee đề xuất năm 2009.

* + 1. Khai thác luật tuần tự

Khai thác luật tuần tự là giai đoạn tiếp theo của khai thác cơ sở dữ liệu chuỗi. Luật tuần tự được sinh ra dự trên mẫu tuần tự. Mỗi luật khai thác được thể hiện mối quan hệ của những sự kiện theo thời gian, giúp cho việc dự đoán các sự kiện xảy ra tiếp theo sau đó. Khai thác luật tuần tự mang lợi ích cho việc rút trích và biểu diễn được những thông tin hữu ích từ tập mẫu tuần tự.

Phương pháp khai thác luật tuần tự đầy đủ và đầu tiên được đề xuất vào năm 1999 bởi Spiliopoulou sau đó được khái quát thành thuật toán Full do Lo đề xuất năm 2009.

Vấn đề sinh ra khi khai thác toàn bộ tập luật là việc sinh ra và tồn tại những luật dư thừa. Vì vậy, để giải quyết vấn đề này, các thuật toán các thuật toán khai thác luật tuần tự không dư thừ a được đề xuất.

Thuật toán dựa vào các đặc tính của cây tiền tố, thuật toán MSR-ImpTree được Van đề xuất năm 2011 đã cải tiến thuật toán Full để tăng hiệu quả khai thác luật. Thuật toán CloGen do Pham đề xuất năm 2013 cho phép khai thác luật tuần tự không dư thừa hiệu quả.

* + 1. Hạn chế của khai thác mẫu tuần tự

Sau quá trình khai thác quy tắc tuần tự, đã phát hiện được rằng phần lớn các quy tắc sẽ có dạng:

Trong đó A là một thực thể thực hiện tới B với tần suất là [a,b,c] lần.

Các quy tắc tuần tự và quy tắc kết hợp có khái niệm tương tự nhau, ngoại trừ việc A phải xảy ra trước B theo thứ tự tuần tự và các quy tắc tuần tự này cũng được khai thác theo trình tự.

Hạn chế lớn nhất của khai thác mẫu tuần tự này là nó xem xét tất cả các mẫu tuần tự và bao gồm cả những dữ liệu có trình tự và độ tin cậy rất thấp. Những mẫu này không có tác dụng trong việc ra quyết định hoặc dự đoán, việc này được giải quyết bằng cách khai thác quy tắc tuần tự.

Một bài toán khai thác dữ liệu quan trọng là thiết kế thuật toán. Vì các tập dữ liệu thường rất lớn, nên việc chuẩn hóa dữ liệu để năng cao độ chính xác và thời gian xử lý của việc dự đoán là một thách thức luôn được đề ra trong quá trình khai thác dữ liệu tuần tự. Để giới thiệu thông tin chính xác cho người dùng thì việc lựa chọn thuật toán để khai thác hiệu quả là một vấn đề cũng cần được lưu ý. Công thức độ chính xác:

Trong đó:

: độ chính xác của dự đoán.

: số lượng chuỗi được dự đoán thành công.

số lượng chuỗi dự đoán.

* 1. Các khái niệm
* Một cơ sở dữ liệu chuỗi là danh sách các chuỗi, ký hiệu là SDB.
* Một sự kiện thường được ký hiệu là các ký tự chữ cái. Các sự kiện trong tập sự kiện được thể hiện theo thứ tự từ điển và được ký hiệu trong “( )”
* Itemset là một tập hợp các item. Một itemset bao gồm hai hoặc nhiều item. Nếu một itemset có k-item thì được gọi là một k-itemset. Ví dụ: là chuỗi có 3 itemset và là 3-itemset.
* Chuỗi là một danh sách những itemset có thứ tự, một chuỗi có dạng với mỗi là một itemset và *n* là số lượng các itemset có trong chuỗi. Tổng số itemset có trong chuỗi là kích thước của chuỗi và tổng các item có trong chuỗi là chiều dài của chuỗi. Chuỗi có chiều dài (k) được gọi là một k-sequence. Ví dụ: là chuỗi có độ dài là 5, kích thước là 4 và là 5-sequence.
* Cho chuỗi và chuỗi . được gọi là một chuỗi con của () khi và chỉ khi tồn tại những số nguyên (theo thứ tự tuần tự) sao cho , , …, . Ví dụ: chuỗi là chuỗi con của chuỗi nhưng không phải là chuỗi con của .
* Cơ sở dữ liệu chuỗi là một tập hợp có dạng với là định danh của chuỗi và là chuỗi các itemset.
* Mẫu là một chuỗi con của một chuỗi dữ liệu và trong đó các itemset được xem là một thành phần (element).
* Độ hỗ trợ (support) ký hiệu là bao gồm độ hỗ trợ tuyệt đối và độ hỗ trợ tương đối.
* Độ hỗ trợ tuyệt đối của một mẫu tuần tự là tổng số lượng chuỗi trong cơ sở dữ liệu có chứa . Ký hiệu
* Độ hỗ trợ tương đối của mẫu tuần tự là tỉ lệ (%) chuỗi trong cơ sở dữ liệu có chứa .
* Mẫu tuần tự: với một ngưỡng hỗ trợ tối thiểu cho trước và được xác định bởi người dùng, . Mẫu là phổ biến nếu thỏa điều kiện .
* Khai thác mẫu tuần tự là bài toán đi tìm tập đầy đủ của tất cả các mẫu tuần tự có trong cơ sở dữ liệu.
* Tiền tố, hậu tố và tiền tố không hoàn toàn:

Cho chuỗi và chuỗi .

* là tiền tố của khi và chỉ khi với mọi .
* Sau khi bỏ đi phần tố trên chuỗi , phần chuỗi còn lại sẽ là hậu tố của
* là một tiền tố không hoàn toàn của khi và chỉ khi với mọi -1, và tất cả các item trong tập () là những item đứng sau các item trong xét theo thứ tự từ điển.
* Với một chuỗi có kích thước là thì sẽ có phần tử.

Ví dụ: trong chuỗi có 2 tiền tố , và 2 hậu tố là , . Hai chuỗi và không phải là tiền tố của chuỗi trên. là tiền tố không hoàn toàn.

* 1. Khái niệm về luật tuần tự

Các mẫu tuần tự thường được tìm dựa trên sự hỗ trợ của chúng (số lần xuất hiện trong cơ sở dữ liệu tuần tự). Một quy tắc tuần tự được tìm thấy dựa trên độ tin cậy và sự hỗ trợ của nó, vì thế nên luật tuần tự được áp dụng trong các vấn đề khuyến nghị.

Luật tuần tự được hình thành dựa tên mẫu tuần tự bằng cách tách mẫu tuần tự ra làm hai phần là tiền tố X và hậu tố Y. Khi kết hợp tiền tố và hậu tố lại sẽ được chuỗi như ban đầu (X++Y). Luật tuần tự biểu diễn mối quan hệ giữa các loạt sự kiện xảy ra theo tuần tự.

Trong đó:

là sự kiện xảy ra trước Y.

là sự kiện xảy ra sau X.

là độ hỗ trợ.

là độ tin cậy.

Độ hỗ trợ của luật chính là độ hỗ trợ của mẫu tuần tự. Độ tin cậy của luật bằng khả năng chuỗi trong cơ sở dữ liệu có chứa tiền kiện của luật dẫn đến chứa hậu kiện của luật. Một luật s được coi là phổ biến khi và được coi là đáng tin cậy khi . Công thức xác định độ hỗ trợ và độ tin cậy của luật:

Trong đó:

là độ hỗ trợ.

là độ tin cậy.

là phép nối tiền tố và hậu tố.

Có thể tạo ra (k-1) luật với mỗi k mẫu tuần tự bởi vì mẫu tuần tự có kích thước k cũng sẽ có (k-1) tiền tố. Ví dụ: mẫu tuần tự có kích thước là 3 và sẽ tạo ra 3-1 luật là và .

* + 1. Các cách tổ chức dữ liệu

Tổ chức dữ liệu cơ bản có 2 dạng:

* Dạng biểu diễn ngang: dữ liệu được tổ chức theo chiều ngang, mỗi hàng đại diện cho dãy các sự kiện tương ứng với đối tượng.

|  |  |
| --- | --- |
| Đối tượng | Sự kiện |
| 1 | B, C, D |
| 2 | A, C, D, E |

Bảng 1: Tổ chức dữ liệu theo chiều ngang.

* Dạng biểu diễn dọc: dữ liệu được tổ chức theo chiều dọc, mỗi hàng đại diện cho dãy các đối tượng tương ứng với các sự kiện.

|  |  |
| --- | --- |
| Sự kiện | Đối tượng |
| A | 1 |
| B | 1 |
| C | 1, 2 |
| D | 1, 2 |
| E | 2 |

Bảng 2: Tổ chức dữ liệu theo chiều dọc

Dữ liệu ban đầu thường được tổ chức theo chiều ngang, phải qua giai đoạn tiền xử lý dữ liệu mới có thể tổ chức dữ liệu theo chiều dọc. Dữ liệu tổ chức theo chiều dọc đơn giản và nhanh hơn vì có thể lấy được ngay các đối tượng ứng với các sự kiện mà không cần duyệt toàn bộ cơ sở dữ liệu. Trong các cơ sở dữ liệu lớn, tổ chức theo chiều dọc mang tính cô đọng giúp quá trình thực thi trở nên nhanh hơn và cho phép lặp lại dễ dàng việc tìm các mẫu tuần tự.

* + 1. Bài toán khai thác luật tuần tự

Bài toán khai thác luật tuần tự là bài toán đi tìm ra những luật thỏa mãn tối thiểu độ hỗ trợ và độ tin cậy cho trước. Để thực hiện ta cần hai bước:

* Giai đoạn 1: tìm tất cả các mẫu tuần tự trong cơ sở dữ liệu thỏa mãn .
* Giai đoạn 2: từ các mẫu tuần tự đã tìm ra, sinh luật tuần tự thỏa mãn .

Ví dụ 1: Cho cơ sở dữ liệu, độ hỗ trợ tối thiểu minSup = 2

|  |  |
| --- | --- |
| SID | Sequence |
| 10 |  |
| 20 |  |
| 30 |  |
| 40 |  |

Bảng 3: Cơ sở dữ liệu chuỗi với minSup = 2.

* Xét chuỗi , trong chuỗi có 5 itemset là và có 9 item. Chuỗi này có kích thước là 5 và độ dài là 9.
* Chuỗi là chuỗi con của , vì vậy chuỗi được gọi là mẫu. Trong cơ sở dữ liệu trên chỉ có chuỗi 10 là có chứa vì vậy độ hỗ trợ của mẫu . Do nên không phải là một mẫu tuần tự.
* Chuỗi là chuỗi con của . Trong cơ sở dữ liệu có chuỗi 10 và 30 có chứa nên độ hỗ trợ và nên là mẫu tuần tự.

Ví dụ 2: Cho cơ sở dữ liệu, độ hỗ trợ tối thiểu minSup = 50% và độ tin cậy tối thiểu minConf = 70%

|  |  |
| --- | --- |
| SID | Sequence |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |

Bảng 4: Cơ sở dữ liệu chuỗi với minSup = 50% và minConf = 70%

* Tìm tất cả các mẫu tuần tự có độ hỗ trợ . Do minSup của đề là độ hỗ trợ tối thiểu nên . Tất cả mẫu tuần tự được tìm ra như bảng:

|  |  |
| --- | --- |
| Kích thước | Mẫu tuần tự: Độ hỗ trợ |
| 1 | , , |
| 2 | ,  ,  , , |
| 3 | ,  ,  , , |

Bảng 5: Bảng các mẫu tuần tự thỏa minSup

* Tìm tất cả tập luật từ bảng mẫu tuần tự vừa tìm được (chỉ sinh luật với các mẫu lớn hơn 1)

|  |  |  |
| --- | --- | --- |
| Mẫu tuần tự | Luật tuần tự | Độ tin cậy |
|  |  | Có |
|  |  | Có |
|  |  | Có |
|  |  | Có |
|  |  | Không |
|  |  | Có |
|  |  | Có |
|  |  | Không |
|  |  | Không |
|  |  | Có  Có |
|  |  | Có  Có |
|  |  | Có  Có |
|  |  | Có  Có |
|  |  | Có  Có |
|  |  | Có  Có |
|  |  | Không  Không |

Bảng 6: Bảng các luật tuần tự thỏa minConf.

🡪 Có thể khai thác được 18 luật tuần tự thỏa minSup và minConf

* 1. Các thuật toán khai thác mẫu tuần tự

Đối với bài toán khai thác mẫu tuần tự, cách thức tổ chức dữ liệu và thuật toán giải quyết là hai yếu tố chính ảnh hưởng đến tính hiệu quả. Do đó phải sử dụng cấu trúc dữ liệu thích hợp và thuật toán tối ưu để nâng cao chất lượng kết quả.

Các yếu tố ảnh hưởng đến tốc độ thực thi là cách tổ chức biểu diễn dữ liệu để lưu trữ vào bộ nhớ, cách duyệt dữ liệu để xử lý và các chiến lược tìm kiếm. Đồng thời, cũng có thể sử dụng cách vận dụng lý thuyết đồ thị, đưa ra những ràng buộc cho bài toán sẽ giúp cho thời gian thực thi trở nên nhanh hơn và các mẫu tuần tự tìm được cũng có giá trị hơn.

Bài toán khai thác mẫu tuần tự được đề xuất đầu tiên vào năm 1995 bởi R.Agrawal và R.Srikant. ApioriAll là thuật toán đầu tiên được thiết kế để giải quyết bài toán khai thác mẫu tuần tự trên cơ sở dữ liệu chuỗi giao dịch. Thuật toán này sinh ra dựa trên thuật toán khai thác mẫu phổ biến Apriori. Apriori là thuật toán nền tảng cho các thuật toán về sau.

* + 1. Apriori
       1. Giới thiệu

R.Agrawal và cộng sự của mình là R.Srikant đã giới thiệu phương pháp Apriori cho khai thác mẫu tuần tự vào năm 1994 để giải quyết thách thức trong việc phát hiện các tập phổ biến thường xuyên trong một tập dữ liệu khổng lồ. Thuật ngữ Apriori được đặt cho phương pháp này vì nó sử dụng kiến ​​thức trước đây (prior) về các thuộc tính và vật phẩm thường xuất hiện trong cơ sở dữ liệu. Một đặc tính thiết yếu, Apriori được sử dụng để cải thiện hiệu quả của việc lọc mức thường xuyên bằng cách giảm phạm vi tìm kiếm.

Nguyên tắc Apriori là nguyên tắc loại bỏ dựa trên độ hỗ trợ:

* + Nếu một tệp là phổ biến thì tất cả các tập con đều phổ biến.
  + Nếu một tệp không phổ biến thì tất cả các tập cha cũng đều không phổ biến.

Ví dụ: Nếu không phổ biến thì cả hai và cũng không phổ biến.

Ví dụ: Cho cơ sở dữ liệu sau có minSup = 2 và minConf = 60%

|  |  |
| --- | --- |
| TID | Items |
| T1 | I1, I2, I5 |
| T2 | I2, I4 |
| T3 | I3, I3 |
| T4 | I1, I2, I4 |
| T5 | I1, I3 |
| T6 | I2, I3 |
| T7 | I1, I3 |
| T8 | I1, I2, I3, I5 |
| T9 | I1, I2, I3 |

Bảng 7: Bảng cơ sở dữ liệu ví dụ cho thuật toán Apriori.

Với k = 1 và thỏa , ta có:

|  |  |
| --- | --- |
| Itemset | Support |
| I1 | 6 |
| I2 | 7 |
| I3 | 6 |
| I4 | 2 |
| I5 | 2 |

Bảng 8: Bảng k = 1 thỏa minSup.

Với k = 2 và thỏa , ta có:

|  |  |
| --- | --- |
| Itemset | Support |
| I1, I2 | 4 |
| I1, I3 | 4 |
| I1, I5 | 2 |
| I2, I3 | 4 |
| I2, I4 | 2 |
| I2, I5 | 2 |

Bảng 9: Bảng k = 2 thỏa minSup.

Với k = 3 và thỏa , ta có:

|  |  |
| --- | --- |
| Itemset | Support |
| I1, I2, I3 | 2 |
| I1, I2, I5 | 2 |

Bảng 10: Bảng k = 3 thỏa minSup.

Với conf bằng 60%, ta có:

|  |  |  |
| --- | --- | --- |
| Itemset |  |  |
| I1, I2 |  | Có |
| I1, I3 |  | Có |
| I1, I5 |  | Không |
| I2, I3 |  | Không |
| I2, I4 |  | Không |
| I2, I5 |  | Không |
| I1, I2, I3 |  | Không |
| I1, I2, I5 |  | Không |

Bảng 11: Bảng các itemset thỏa mãn minSup và minConf.

Vậy có 2 Itemset thỏa điều kiện và

* + - 1. Hạn chế

Hạn chế của thuật toán Apriori là nó có thể rất chậm và có thể dẫn đến tình trạng “nghẽn cổ chai”.

Sự phức tạp không gian càng gia tăng do số lượng lớn tập phổ biến được tạo ra.

Quá nhiều lần duyệt cơ sở dữ liệu được yêu cầu do số lượng lớn tập phổ biến được tạo. Do số lần duyệt cơ sở dữ liệu nhiều làm gia tăng sự phức tạp thời gian khi cơ sở dữ liệu gia tăng.

Ví dụ: trong một cơ sở dữ liệu có tệp 1 mục phổ biến thì chúng sẽ tạo ra tệp 2 mục ứng viên phổ biến tương ứng trong đó.

Để tính những giá trị có độ hỗ trợ lớn hơn độ hỗ trợ tối thiểu thì phải cần quét ở mọi cấp độ của cơ sở dữ liệu. Nó cần (n+1) lần quét, trong đó n là độ dài của mẫu dài nhất.

* + - 1. Ưu nhược điểm

Ưu điểm:

* + Sử dụng thuộc tính của tập phổ biến lớn
  + Kết quả trực quan và dễ dàng giao tiếp với người dùng cuối.
  + Là thuật toán đơn giản, dễ hiểu, dễ thực hiện
  + Không yêu cầu dữ liệu có dán nhãn vì nó là thuật toán không giám sát, vì thế có thể sử dụng trong nhiều tính huống khác nhau vì dữ liệu không gán nhãn sẽ dễ truy cập hơn.
  + Nhiều tiện tích mở rộng được đề xuất cho các trường hợp sử dụng khác nhau

Khuyết điểm:

* + Phải duyệt cơ sở dữ liệu nhiều lần
  + Số lượng tập ứng viên rất lớn
  + Thực hiện việc tính độ phổ biến nhiều, đơn điệu
    - 1. Phương pháp cải thiện

Một số phương pháp cải tiến thuật toán Apriori:

* + Cải tiến dựa vào ma trận.
  + Cải tiến sử dụng bảng băm (hash-based technique).
  + Giảm giao dịch.
  + Cải tiến theo trọng số Apriori.
  + OOO – Thuật toán cải thiện Apriori.
  + Cải tiến dựa vào Interest itemset.
  + Cải tiến dựa vào nén giao tác.
  + Đếm itemset động.
    1. AprioriAll
       1. Giới thiệu

AprioriAll là phương pháp đầu tiên dựa trên thuật toán Apriori để giải quyết vấn đề khai thác mẫu tuần tự trên cơ sở dữ liệu. Nó hoạt động qua ba giai đoạn:

* Ở giai đoạn đầu tiên, thuật toán AprioriAll sẽ tiến hành duyệt toàn bộ cơ sở dữ liệu ban đầu để thực hiện việc tìm các itemset phổ biến, sau đó ánh xạ sang tập số nguyên. Việc ánh xạ nhằm mục đích để coi một itemset phổ biến là một thực thể riêng biệt, đồng thời giảm thiểu thời gian kiểm tra một chuỗi có phải là chuỗi con với chuỗi dữ liệu trong cơ sở dữ liệu ban đầu.
* Trong cơ sở dữ liệu chuỗi ban đầu, mỗi chuỗi sẽ được thay thế bởi tập tất cả các itemset phổ biến có chứa trong chuỗi đó. Nếu itemset không chứa itemset con nào phổ biến thì loại bỏ, nếu chuỗi dữ liệu không chứa itemset phổ biến nào thì cũng loại bỏ ra khỏi cơ sở dữ liệu. Dữ liệu sau đó sẽ được biến đổi thành một chuỗi sẽ được đại diện bởi một dãy các itemset phổ biến.
* Tìm mẫu tuần tự dựa trên kết quả itemset phổ biến vừa tìm được, ta có được tập các mẫu tuần tự có kích thước là 1. Các tập ứng viên gồm các mẫu có độ dài k được phát sinh bằng cách kết các mẫu có độ dài (k-1), sau đó áp dụng nguyên tắc Apriori và ngưỡng hỗ trợ minSup để loại bỏ các mẫu không phổ biến.
  + - 1. Hạn chế

Số lượng các ứng viên sinh ra là rất lớn và dễ dẫn đến tình trạng “nghẽn cổ chai”. Với một chuỗi có độ dài thì có thể tạo ra ứng viên dẫn đến không đủ bộ nhớ để xử lý.

Phải duyệt cơ sở dữ liệu nhiều lần vì với mỗi tệp ứng viên đều phải duyệt lại cơ sở dữ liệu để đếm độ hỗ trợ.

* + 1. GSP – Generalized Sequential Pattern
       1. Giới thiệu

Thuật toán GSP là thuật toán mở rộng từ mô hình Apriori. GSP là bài toán khai thác mẫu tuần tự tuy nhiên khác với Apriori do có thêm các ràng buộc về khoảng thời gian cực đại và cực tiểu giữa các thành phần trong một mẫu tuần tự.

Các item của một thành phần (itemset) trong một mẫu có thể lấy từ một hoặc nhiều thành phần khác nhau nhưng khoảng cách thời gian giữa các thành phần phải nằm trong “time window”. Đồng thời, một mẫu có thể gồm những item trên các mức phân cấp khác nhau.

Thuật toán GSP có cấu trúc cơ bản là duyệt dữ liệu nhiều lần. Ở lần duyệt đầu tiên sẽ xác định độ hỗ trợ của từng item. Sau khi duyệt lần đầu tiên, thuật toán sẽ đưa được ra tập khởi đầu bao gồm các tập 1-sequence phổ biến và tập này được sử dụng để sinh ra các dãy ứng viên mới với mỗi dãy ứng viên có ít nhất một item thuộc dãy khởi đầu. Độ hỗ trợ của các ứng viên tìm thấy và tính toán trong quá trình duyệt dữ liệu. Sau khi duyệt xong, thuật toán xác định được các ứng viên phổ biến và các ứng viên này sẽ trở thành tập khởi đầu cho lần duyệt tiếp theo. Thuật toán sẽ kết thúc khi không còn tìm được dãy ứng viên nào cuối lần duyệt hoặc là khi không có dãy ứng viên nào được sinh ra.

Ví dụ: Cho cơ sở dữ liệu có độ hỗ trợ minSup = 2

|  |  |
| --- | --- |
| SID | Sequence |
| 10 |  |
| 20 |  |
| 30 |  |
| 40 |  |
| 50 |  |

Bảng 12: Bảng cơ sở dữ liệu ví dụ cho thuật toán GSP.

* Ứng viên đầu tiên:

|  |  |
| --- | --- |
| Cand | Support |
|  | 3 |
|  | 5 |
|  | 4 |
|  | 3 |
|  | 3 |
|  | 2 |

Bảng 13: Bảng ứng viên đầu tiên.

có độ hỗ trợ là 1 < minSup nên bị loại

* Ứng viên thứ hai:

Chuỗi có chiều dài là 2 và có 2 phần tử:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Bảng 14: Bảng ứng viên thứ hai có chiều dài là 2 và có 2 phần tử.

Chuỗi có chiều dài là 2 và có 1 phần tử:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Bảng 15: Bảng ứng viên thứ hai có chiều dài là 2 và có 1 phần tử.

🡪 Có 19 ứng viên thỏa điều kiện

* Ứng viên thứ ba

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | <𝑐,𝑒> |
|  | 1  1  0  2  1 | 1  1  1 | 1  1  0 |
|  | <𝑎𝑏𝑎> 2  <𝑎𝑏𝑏> 2  1  <(𝑏, 𝑑)𝑏> 2 | 1  <(𝑏, 𝑓)𝑏> 2  0 | 1  1  1  <𝑏(𝑐, 𝑒)> 2 |
|  | 1  1 | 1 | 1  1 |
|  | 1  <𝑏𝑏𝑎> 2  1 | <𝑏𝑏𝑓> 2 | <𝑏𝑏𝑐>2 |
|  | <𝑏𝑐𝑏> 3  <𝑏𝑐𝑑>2  <𝑏𝑐𝑎> 2  <(𝑏, 𝑑)𝑐> 2 | 1  1  0 | 1    0  0 |
|  | 0  0  0  0 | 0  0  0 | 0  0  0 |
|  | 1  1 | 1  1  0 | 0  0  0  1 |
|  | 0  0  0 | 0  2  0 | 0  0  1 |
|  | 0  0  0 | 0 | 0  0 |
|  | 0  <𝑐𝑏𝑎> 2  1 | 1 | 1  1 |
|  | 0  0  0 | 0 | 0  0 |
|  | 0  0 | 0 | 0  0 |
|  | 0  0  0 | 0 | 0  <𝑑𝑐𝑎> 2  0 |
|  | 1  <𝑑𝑏𝑎> 2  1 | 1 | <𝑑𝑏𝑐> 2  1 |
|  | 𝑑𝑐𝑏> 2  0 | 0 | 1  1 |
|  | 0  0  0 |  | 0  0 |

Bảng 16: Bảng ứng viên thứ ba.

🡪 Có 46 ứng viên trong đó có 19 ứng viên thỏa điều kiện

* Ứng viên thứ tư:

Tương tự như các bước trước đó ta có được 8 ứng viên và 6 ứng viên thỏa điều kiện 2, 2, 2, 2, 2, 2

* Ứng viên thứ năm:

🡪 Có 1 ứng viên là 2 thỏa điều kiện

* + - 1. Hạn chế

Số lượng ứng viên khổng lồ, đặc biệt với các chuỗi ứng viên có chiều dài là 2

Thuật toán GSP phải duyệt cơ sở dữ liệu nhiều lần

Không hiệu quả khi áp dụng vào khai thác các chuỗi dài

* Cách giải quyết là sử dụng thuật toán PrefixSpan
  + 1. PSP

PSP cũng là thuật toán dựa trên mô hình Apriori. Điểm khác biệt giữa PSP và GSP là PSP quản lý và lưu trữ các mẫu ứng viên bằng cấu trúc dữ liệu hiệu quả hơn. Tuy nhiên cũng giống như AprioriAll và GSP, PSP cũng phải duyệt cơ sở dữ liệu nhiều lần và phải tải toàn bộ cơ sở dữ liệu đó vào bộ nhớ chính. PSP chỉ thật sự hiệu quả khi bộ nhớ chính có thể chứa hết toàn bộ cơ sở dữ liệu.

* + 1. WAP-mine

Đây là thuật toán phát triển mẫu với phép chiếu cây, ra đời cùng thời điểm với hai thuật toán FreeSpan năm 2000 và PrefixSpan năm 2001. WAP-mine sử dụng cấu trúc cây WAP. Cơ sở dữ liệu ở đây sẽ được duyệt hai lần để xây dựng cây WAP bao gồm các chuỗi phổ biến cùng với độ hỗ trợ của chúng, đồng thời duy trì một “header table” (“bảng con trỏ đầu”) để trỏ đến các vị trí xuất hiện đầu tiên của mỗi item nằm trong itemset phổ biến. WAP-mine duyệt cơ sở dữ liệu hai lần, lần đầu để tìm chuỗi phổ biến 1-sequences và duyệt lần hai để xây dựng cây WAP chỉ bao gồm các chuỗi con phổ biến. Wap-mine hạn chế việc sinh các ứng viên một cách hiệu quả, tuy nhiên lại phát sinh vấn đề tiêu thụ bộ nhớ do nó xây dựng lặp lại nhiều cây Wap-tree trung gian.

Diagram, engineering drawing

Description automatically generated

Hình 4: Ví dụ về bài toán WAP-mine

* + 1. FreeSpan (Frequent Pattern-Projected Sequential pattern mining), PrefixSpan (Prefix-Projected Sequential pattern mining)

FreeSpan và PrefixSpan là hai thuật toán tiếp cận theo hướng chia nhỏ dữ liệu. FreeSpan là thuật toán khai thác đầu tiên thực hiện phép chiếu trên cơ sở dữ liệu để giảm bớt chi phí dữ liệu. PrefixSpan là thuật toán được tạo ra phát triển từ FreeSpan, từ tập mẫu có độ dài là 1, PrefixSpan sẽ tạo ra cơ sở dữ liệu dự kiến được chiếu với mỗi mẫu đó. Sau khi quy chiếu cơ sở dữ liệu chứa các dãy dữ liệu chỉ giữ lại phần hậu tố, đối với tiền tố đã chiếu, sau đó mẫu được phát triển bởi những item phổ biến tìm được trong cơ sở dữ liệu được chiếu, quá trình này lặp đi lặp lại cho đến khi cơ sở dữ liệu chiếu không còn itemset phổ biến nào.

Các phương pháp AprioriAll, GSP, PSP đều tổ chức dữ liệu theo chiều ngang. Mỗi lần xác định độ hỗ trợ của một mẫu phải duyệt lại toàn bộ cơ sở dữ liệu. Để khắc phục vấn đề này, các thuật toán tổ chức dữ liệu theo chiều dọc được ra đời mà đầu tiên là SPADE, sau đó có SPAM và PRISM. Khác với tổ chức dữ liệu theo chiều ngang là phải duyệt toàn bộ cơ sở dữ liệu, với mỗi ứng viên chúng thực hiện lưu trữ thông tin cho biết mẫu đó đã có mặt trong những chuỗi dữ liệu nào trong cơ sở dữ liệu và tính độ hỗ trợ. Mẫu mới được tạo ra lấy thông tin dựa trên những mẫu đã có và không cần phải duyệt lại cơ sở dữ liệu.

* + - 1. FreeSpan

FreeSpan tiếp cận theo hướng chia nhỏ dữ liệu và là thuật toán đầu tiên thực hiện phép chiếu trên cơ sở dữ liệu để giảm chi phí lưu trữ dữ liệu.

Đầu tiên, thuật toán FreeSpan tạo ra một danh sách các chuỗi 1-sequences từ các cơ sở dữ liệu chuỗi được gọi là danh sách item phổ biến (*f-list*) và sau đó nó sẽ xây dựng một ma trận các item trong danh sách này. Ma trận sẽ chứa thông tin về độ hỗ trợ của các chuỗi ứng viên có độ dài là 2 được phát sinh trong các item của f-list và được gọi là *S-Matrix.*

Với một mẫu tuần tự α từ *S-Matrix*, cơ sở dữ liệu chiếu theo α là tập hợp các chuỗi phổ biến trong đó có α là chuỗi con. Những item không phổ biến và các item theo sau các item không phổ biến trong α đều được bỏ qua. Tiếp theo, xây dụng bảng gồm các chuỗi 2-sequences với chú thích trên những item lặp lại trong chuỗi và chú thích trên cơ sở dữ liệu chiếu để giúp định vị những cơ sở dữ liệu chiếu này trong lần duyệt thứ ba và các lần duyệt sau đó mà không cần quan tâm đến *S-Matrix.* Vì thế lúc này *S-Matrix* sẽ được bỏ qua và khai thác tiếp tục sử dụng cơ sở dữ liệu chiếu.

* + - 1. PrefixSpan

Thuật toán PrefixSpan được đề xuất bởi Jian Pei và cộng sự vào năm 2001, thuật toán chỉ xét các chuỗi con tiền tố và chỉ chiếu các các chuỗi con hậu tố tương ứng của chúng vào các cơ sở dữ liệu chiếu. PrefixSpan giúp các mẫu tuần tự được phát triển trong mỗi cơ sở dữ liệu chiếu vì chỉ duyệt các chuỗi phổ biến cục bộ.

Ví dụ: Cho cơ sở dữ liệu chuỗi với minSup = 2

Xét cơ sở dữ liệu dưới, chuỗi và chiếu lên , các chuỗi này đều là tiền tố, nhưng hai chuỗi thì không phải là tiền tố. là hậu tố của chuỗi đã cho đối với tiền tố và là hậu tố đối với tiền tố

|  |  |
| --- | --- |
| SID | Sequence |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |

Bảng 17: Bảng cơ sở dữ liệu ví dụ cho thuật toán PrefixSpan.

Các bước thực hiện khi áp dụng thuật toán PrefixSpan:

1. Tìm tất cả các mẫu tuần tự có độ dài là 1:

Từ cơ sở dữ liệu trên, duyệt và tìm các item phổ biến, mỗi item phổ biến tạo nên một chuỗi có độ dài là 1. Ta có kết quả: ,

1. Chia nhỏ không gian tìm kiếm để lấy cơ sở dữ liệu

Ví dụ: cơ sở dữ liệu trên tạo ra 6 tập con rời nhau theo 6 tiền tố ,

1. Tìm mẫu phổ biến theo các loại tiền tố

Ứng dụng với mỗi tiền tố, thực hiện chiếu cơ sở dữ liệu theo tiền tố đó và khai thác các mẫu dữ liệu trong cơ sở dữ liệu đã được chiếu.

Để tìm mẫu tuần tự có tiền tố , ta mở rộng bằng cách thêm vào một item tại một thời điểm. Để thêm một item *x* kế tiếp, có hai khả năng xảy ra:

* + Thuật toán nối item và itemset cuối trong tiền tố ()
  + Item được thêm vào đóng vai trò là một itemset riêng ()

Để tạo cơ sở dữ liệu chiếu theo tiền tố : nếu một chuỗi có chứa item , thì hậu tố theo sau tiền tố được rút trích ra như là một chuỗi cho cơ sở dữ liệu chiếu. Cơ sở dữ liệu chiếu theo :

|  |  |
| --- | --- |
| SID | Sequence |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |

Bảng 18: Bảng cơ sở dữ liệu chiếu theo

Từ cơ sở dữ liệu trên, các item phổ biến: , .

Các chuỗi có độ dài là 2 với tiền tố và có độ hỗ trợ với là . Thực hiện đệ quy lần lượt các mẫu phổ biến theo tiền tố các chuỗi có độ dài là 2. Thực hiện PrefixSpan và chiếu vào cơ sở dữ liệu ta được:

|  |  |
| --- | --- |
| SID | Sequence |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |

Bảng 19: Bảng cơ sở dữ liệu chiếu theo

Từ cơ sở dữ liệu trên, các item phổ biến: , . Các chuỗi có độ dài 3 với tiền tố là: . Tiếp tục thực hiện đệ quy lần lượt các mẫu trên theo tiền tố của chuỗi có độ dài là 3. Thực hiện PrefixSpan và chiếu vào cơ sở dữ liệu ta được:

|  |  |
| --- | --- |
| SID | Sequence |
| 1 |  |
| 2 |  |

Bảng 20: Bảng cơ sở dữ liệu chiếu theo .

Từ cơ sở dữ liệu trên, các item phổ biến: . Các chuỗi có độ dài 4 với tiền tố là: . Tiếp tục thực hiện đệ quy lần lượt các mẫu trên theo tiền tố của chuỗi có độ dài là 3. Thực hiện PrefixSpan và chiếu vào cơ sở dữ liệu ta được:

|  |  |
| --- | --- |
| SID | Sequence |
| 1 |  |

Bảng 21: Bảng cơ sở dữ liệu chiếu theo .

Trong cơ sở dữ liệu chiếu theo không có items nào phổ biến, nên thuật toán PrefixSpan sẽ được dừng lại.

Thực hiện tương tự PrefixSpan với các tiền tố còn lại ta được:

|  |  |  |
| --- | --- | --- |
| Tiền tố (Prefix) | CSDL đã được chiếu theo tiền tố | Mẫu tuần tự |
|  |  | … |
|  |  | … |
|  |  | … |
|  |  |  |
|  |  |  |
|  |  |  |

Bảng 22: Bảng cơ sở dữ liệu thực hiện PrefixSpan.

* + - 1. Đánh giá

PrefixSpan là phương pháp phát triển mẫu trên những mẫu dài từ những mẫu tuần tự có độ dài ngắn hơn đã có, đồng thời kết hợp thêm phương pháp chia nhỏ không gian tìm kiếm, duyệt không gian để tìm kiếm theo tiền tố.

PrefixSpan có tốc độ thực thi nhanh hơn SPADE và AprioriAll do:

* + Thuật toán PrefixSpan phát triển mẫu mà không cần phát sinh ứng viên.
  + Vận dụng nguyên lý Apriori để cắt tỉa cơ sở dữ liệu được chiếu.
  + Mặt khác do PrefixSpan thực hiện phép chiếu cơ sở dữ liệu theo tiền tố nên giảm được không gian tìm kiếm một cách đáng kể. Do đó tiết kiệm bộ nhớ và đếm độ hỗ trợ nhanh hơn.

PrefixSpan sử dụng bộ nhớ hiệu quả hơn SPADE và SPAM do SPAM duyệt theo DFS nên việc lưu trữ không hiệu quả.

* + 1. SPADE – Sequential Pattern Discovery using Equivalence classes

SPADE là thuật toán khai thác và tổ chức dữ liệu theo chiều dọc, lưu trữ danh sách định danh (ID) của các chuỗi dữ liệu và tập các itemset ứng với mỗi item có chứa item đó. Độ hỗ trợ của item sẽ được tính thông qua danh sách các định danh.

SPADE hoạt động dựa trên lý thuyết dàn nhằm để chia nhỏ không gian tìm kiếm và thao tác kết đơn giản để tạo ra các ứng viên. Thuật toán gom các mẫu tuần tự dựa trên tiền tố của các lớp tương đương.

Khác với các thuật toán khác như AprioriAll, GSP, PSP thì thuật toán SPADE chỉ duyệt cơ sở dữ liệu đúng ba lần: lần duyệt đầu và lần duyệt thứ hai thực hiện tìm các mẫu tuần tự có độ dài là 1 và 2. Tại lần duyệt lần thứ ba, thuật toán sẽ phát triển mẫu k từ hai mẫu có độ dài (k-1) có (k-2) item đầu giống nhau, sau đó tiến hành duyệt từng lớp tương đương, do vậy sẽ giảm chi phí tính toán và có thể sử dụng bộ nhớ hiệu quả hơn. So với thuật toán GSP với ngưỡng minSup thấp, thuật toán SPADE thực hiện nhanh gấp đôi.

Ví dụ:

|  |  |
| --- | --- |
| SID | Sequence |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |

Bảng 23: Bảng cơ sở dữ liệu ví dụ cho thuật toán SPADE

Tổ chức dữ liệu theo chiều dọc:

|  |  |  |
| --- | --- | --- |
| SID | EID | Items |
| 1 | 1 | *a* |
| 1 | 2 | *abc* |
| 1 | 3 | *ac* |
| 1 | 4 | *d* |
| 1 | 5 | *cf* |
| 2 | 1 | *ad* |
| 2 | 2 | *c* |
| 2 | 3 | *bc* |
| 2 | 4 | *ae* |
| 3 | 1 | *ef* |
| 3 | 2 | *abc* |
| 3 | 3 | *df* |
| 3 | 4 | *c* |
| 3 | 5 | *b* |
| 4 | 1 | *e* |
| 4 | 2 | *g* |
| 4 | 3 | *af* |
| 4 | 4 | *c* |
| 4 | 5 | *b* |
| 4 | 6 | *c* |

Bảng 24: Tổ chức lại cơ sở dữ liệu theo chiều dọc

Các mẫu tuần tự có độ dài 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *a* | | *b* | | … |
| SID | EID | SID | EID | … |
| 1 | 1 | 1 | 2 |  |
| 1 | 2 | 2 | 3 |  |
| 1 | 3 | 3 | 2 |  |
| 2 | 1 | 3 | 3 |  |
| 2 | 4 | 4 | 5 |  |
| 3 | 2 |  |  |  |
| 4 | 3 |  |  |  |

Bảng 25: Bảng mẫu tuần tự có độ dài là 1

Các mẫu tuần tự có độ dài 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *ab* | | | *ba* | | | … |
| SID | EID (*a*) | EID (*b*) | SID | EID (*a*) | EID (*b*) | … |
| 1 | 1 | 2 | 1 | 2 | 3 |  |
| 2 | 1 | 3 | 2 | 3 | 4 |  |
| 3 | 2 | 5 |  |  |  |  |
| 4 | 3 | 5 |  |  |  |  |

Bảng 26: Bảng mẫu tuần tự có độ dài là 2

Các mẫu tuần tự có độ dài k từ hai mẫu độ dài (k-1) có (k-2) item đầu giống nhau

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *aba* | | | | … |
| SID | EID (*a*) | EID (*b*) | EID (*a*) | … |
| 1 | 1 | 2 | 3 |  |
| 2 | 1 | 3 | 4 |  |

Bảng 27: Bảng mẫu tuần tự có độ dài k từ hai mẫu độ dài (k-1) có (k-2) item đầu giống nhau

* + 1. SPAM (Sequential Pattern Mining)

SPAM được đề xuất bởi Ayres vào năm 2002 và được xem là một trong những thuật toán khai thác mẫu tuần tự nhanh nhất. Đây là thuật toán được tổ chức dữ liệu theo chiều dọc như SPADE kết hợp các giải pháp của các thuật toán GSP, SPADE, FreeSpan, thông tin của các ứng viên được tổ chức và biểu diễn dưới dạng bit dọc với mỗi bit tương ứng với một itemset của một chuỗi trong cơ sở dữ liệu. Nếu item có trong itemset j thì bit tương ứng itemset j sẽ được đánh dấu là 1, nếu không có thì đánh dấu là 0, độ hỗ trợ của mẫu được xác định trên bảng bit. SPAM thực hiện phương pháp duyệt cây theo chiều sâu DFS để sinh ra các ứng viên và dùng nguyên tắc Apriori để tỉa bớt đi các ứng viên để giảm không gian tìm kiếm.

Khi thực hiện trên cơ sở dữ liệu nhỏ, SPAM sẽ thực hiện nhanh hơn SPADE 2.5 lần nhưng vẫn thua PrefixSpan. Thuật toán SPAM hiệu quả hơn với các cơ sở dữ liệu lớn vì nó tổ chức và lưu trữ dưới dạng bit nên thao tác sinh ứng viên và tính độ hỗ trợ rất hiệu quả.

Ví dụ: cách biến đổi dữ liệu sang dạng nhị phân dọc trong SPAM:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CID | TID | *a* | *b* | *c* | *d* |
| 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 3 | 0 | 1 | 1 | 1 |
| 1 | 6 | 0 | 1 | 1 | 1 |
| 2 | 2 | 0 | 1 | 0 | 0 |
| 2 | 4 | 1 | 1 | 1 | 0 |
| 3 | 5 | 1 | 1 | 0 | 0 |
| 3 | 7 | 0 | 1 | 1 | 1 |
|  |  |  |  |  |  |

Bảng 28 : Bảng cơ sở dữ liệu ví dụ của thuật toán SPAM

|  |  |  |
| --- | --- | --- |
| CID | TID | Itemset |
| 1 | 1 |  |
| 1 | 3 |  |
| 1 | 6 |  |
| 2 | 2 |  |
| 2 | 4 |  |
| 3 | 5 |  |

Bảng 29: Bảng đã biến đổi sang dạng nhị phân trong SPAM

* + 1. PRISM – Prime Encoding Based Sequence Mining

PRISM là thuật toán khai thác mẫu tiếp cận theo hướng sử dụng phương pháp mã hóa nguyên tố để biểu diễn thông tin của các mẫu ứng viên được đề xuất bởi Karam Gouda, Mosab Hassan và Zaki vào năm 2010. PRISM sử dụng cách tiếp cận dọc để tìm liệt kê và đếm độ hỗ trợ như hai thuật toán SPADE và SPAM. Nó sử dụng cấu trúc dữ liệu cây từ điển để lưu trữ các mẫu tuần tự tìm được và chỉ duyệt cơ sở dữ liệu một lần để tìm tập mẫu tuần tự có độ dài là 1 cùng với khối mã hóa thông tin tương ứng với các mẫu đó. Mẫu được phát triển bằng cách thêm một item phổ biến vào mẫu. Xác định thông tin của mẫu mới dựa trên khối mã hóa của mẫu cũ và của item thêm vào.

Như vậy, thuật toán PRISM không phải duyệt cơ sở dữ liệu nhiều lần, đồng thời giảm chi phí tính toán bằng cách sử dụng bảng tra cho khối mã hóa thông tin dựa trên lý thuyết mã hóa nguyên tố.

Đặc điểm chính của các thuật toán theo phương pháp tiện cận phát triển mẫu:

* Phân vùng không gian tìm kiếm: cho phép phân vùng không gian tìm kiếm trong trường hợp số lượng lớn các dãy ứng viên được sinh ra giúp cho việc quản lý bộ nhớ hiệu quả hơn. Sau khi phân vùng không gian tìm kiếm, các phân vùng nhỏ có thể được khai phá song song.
* Cây quy chiếu: các thuật toán sử dụng theo cấu trúc dữ liệu cây nhằm biểu diễn không gian tìm kiếm, sau đó sử dụng phương pháp duyệt theo chiều rộng hoặc theo chiều sâu để tìm kiếm các dãy phổ biến và tỉa các ứng viên theo phương pháp Apriori.
* Duyệt theo chiều sâu: là một trong những tính chất quan trọng trong các thuật toán sử dụng mô hình cây. Theo nghiên cứu, nó được đánh giá cao và thực hiện rất hiệu quả vì sử dụng ít bộ nhớ hơn và có định hướng không gian tìm kiếm do đó số lượng ứng viên được tạo ra ít hơn kỹ thuật duyệt theo chiều rộng.
* Tỉa các dãy ứng viên: phương pháp này giúp việc biểu diễn không gian tìm kiếm nhỏ hơn và duy trì một thủ tục tìm kiếm có hướng và phạm vi hẹp hơn. Phần lớn các thuật toán khai thác mẫu này thường sử dụng kỹ thuật tìm kiếm theo chiều sâu và ứng dụng tính chất của nguyên lý Apriori.
  1. Các thuật toán khai thác mẫu tuần tự đóng

Mẫu tuần tự *f* là mẫu tuần tự đóng nếu nó không tồn tại mẫu *f’* sao cho *f* ⊆ *f’* và *sup(f) = sup(f’)*. Cho trước một cơ sở dữ liệu chuỗi và ngưỡng minsup. Khai thác mẫu tuần tự đóng là tìm đầy đủ tất cả các mẫu tuần tự có trong cơ sở dữ liệu và có độ hỗ trợ lớn hơn hoặc bằng ngưỡng minsup đã cho trước và loại bỏ những tập mẫu con có độ hỗ trợ với chúng ra.

Ví dụ: một cơ sở dữ liệu tìm được 5 mẫu phổ biến là với độ hỗ trợ minsup = 5. Lúc này thuật toán nhận 2 mẫu là mẫu tuần tự và loại bỏ 3 mẫu vì nó là con của tập

* + 1. CloSpan – Closed sequential pattern mining

Thuật toán CloSpan là thuật toán khai thác mẫu tuần tự đóng, tránh được lượng lớn các bước gọi đệ quy. Đầu tiên, thuật toán này tạo ra tập các chuỗi ứng viên đóng được lưu trữ trong một cấu trúc cây hash-chỉ mục và sau đó cắt tỉa nó đi. Một số phương pháp cắt tỉa thường xuyên được sử dụng là *CommomPrefix* và *Backward Sub-Pattern* tỉa không gian tìm kiếm. Do CloSpan phải duy trì sự tập hợp của các chuỗi ứng viên đóng, do đó nó sẽ tiêu tốn nhiều bộ nhớ dẫn đến một không gian tìm kiếm lớn cho việc kiểm tra mô hình đóng khi có nhiều chuỗi phổ biến đóng. Vì vậy nên thuật toán CloSpan không tốt đối với các số lượng chuỗi phổ biến đóng.

Thuật toán CloSpan hoạt động dựa trên việc phát hiện các mẫu tuần tự có độ dài là 2, ví dụ như “A luôn xảy ra trước hoặc sau B”.

Ví dụ: ta có mẫu dữ liệu là mẫu phổ biến. CloSpan sẽ đề xuất các phương pháp liên quan và chứng minh rằng luôn xảy ra trước . Từ đặc điểm này, thuật toán CloSpan sẽ chỉ ra rằng là mẫu phổ biến mà không cần phải quét lại cơ sở dữ liệu lần nửa.

* + 1. BIDE – BI-Directional Extension

BIDE là thuật toán mở rộng của thuật toán CloSpan. Thông qua một phần mở rộng chuỗi mới, được gọi là BIDirectional Extension, thuật toán sử dụng cả hai phương pháp là:

* + Mẫu tiền tố và kiểm tra thuộc tính đóng để ứng dụng phát triển.
  + BIDE đề xuất một phương pháp cắt tỉa là BackScan để lược bớt không gian tìm kiếm sâu hơn so với phương pháp tiếp cận trước.

Ý tưởng chính của BIDE là để tránh mở rộng chuỗi bằng cách phát hiện trước một phần mở rộng đã được chứa trong một chuỗi đã có với mục đích nhằm tăng tốc độ khai thác mà vẫn giữ tính đúng đắn của việc khai thác chuỗi phổ biến đóng. Về mặt khái niệm liệt kê chuỗi phổ biến, không gian tìm kiếm của khai thác chuỗi tạo thành một cấu trúc cây và được định nghĩa đệ quy: nút gốc ở mức 0 của cây chứa chuỗi rỗng và được gán nhãn là . Các nút ở múc L của cây sẽ được gán nhãn là các chuỗi có kích thước L. Các nút con ở mức L+1 được tạo ra bằng cách mở rộng chuỗi ở mức L để được chuỗi có độ dài là (L+1). Bằng cách loại bỏ các chuỗi không phổ biến, các nút còn lại còn ở trong cây từ điển chuỗi phổ biến sẽ tạo thành tập chuỗi phổ biến.

Ví dụ: xét cơ sở dữ liệu sau với . BIDE duyệt cây theo DFS (Depth First Search). Theo hình, các chuỗi phổ biến sẽ được khai thác theo trình tự, ta được kết quả:

|  |  |
| --- | --- |
| SID | Sequence |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |

Bảng 30: Bảng cơ sở dữ liệu ví dụ cho thuật toán BIDE.

Diagram

Description automatically generated

Hình 5: Ví dụ về bài toán BIDE

* + 1. TSP

TSP là thuật toán khai thác k mẫu tuần tự đóng với chiều dài nhỏ nhất *min\_l* trong đó *k* là số mẫu tuần tự đóng mà người dùng muốn tìm. Thuật toán này được xây dựng dựa trên nền tảng cây PDB-Tree.

Định nghĩa 1: một mẫu tuần tự *s* được gọi là mẫu tuần tự đóng có độ dài tối thiểu *min\_l* trong tập *top-k* nếu không tồn tại nhiều hơn *k-1* mẫu đóng có độ dài tối thiểu là *min\_l* và có độ hỗ trợ cao hơn độ hỗ trợ của *s.*

Ví dụ: Xét cơ sở dữ liệu với k = 4 và min\_l = 1. Thuật toán sẽ tìm được 2 mẫu: . Ngoài ra còn có 2 mẫu thỏa độ hỗ trợ là , tuy nhiên hai mẫu này không là mẫu tuần tự đóng do chúng là con của chuỗi .

Định nghĩa 2: Cho một chuỗi *s*, tập chuỗi *IDs* của tất cả các chuỗi trong cơ sở dữ liệu *D* chứa *s* gọi là danh sách chuỗi *ID* được ký hiệu là *SIDList(s).* Tổng của *SIDList(s)* được gọi là tổng chuỗi *ID* được ký hiệu là *SIDSum(s).*

Ví dụ: xét cơ sở dữ liệu ta thấy danh sách chuỗi ID của là [1, 2, 3, 4, 5] và tổng chuỗi ID là 14.

* + 1. Cấu trúc cây PDB

Cây PDB có cấu trúc là sự miêu tả bộ nhớ của cây tìm kiếm tiền tố và lưu trữ thông tin về một phần khai thác cơ sở dữ liệu chiếu trong suốt quá trình khai thác.

Khai thác các mẫu tuần tự dựa trên cây PDB như khai thác cây tìm kiếm tiền tố tuy nhiên khi khai thác trên cây PDB số lần duyệt trên cơ sở dữ liệu chiếu giảm rất nhiều bằng cách thực hiện chiếu các tiền tố theo thứ tự giảm dần của độ hỗ trợ. Vì vậy, cây PDB tại mỗi điểm của tiền tố có cùng độ dài sẽ được sắp xếp giảm dần theo độ hỗ trợ, khi giá trị minsup tăng thì những tiền tố có độ hỗ trợ thấp hơn sẽ dẫn đến không phổ biến nên bộ nhớ sử dụng ít.

Cây này bao gồm một phần cơ sở dữ liệu chiếu được khai thác và một cơ sở dữ liệu chiếu được khai thác hoàn toàn. Nó được xóa khỏi cây PDB bởi các thuộc tính như cây PDB có kích thước nhỏ hơn nhiều so với cây tiền tố trong suốt quá trình khai thác dữ liệu. Độ sâu lớn nhất của cây PDB luôn nhỏ hơn *min\_l* do TSP khai thác tất cả cơ sở dữ liệu chiếu tạo cấp *min\_l* và nhỏ hơn. Giảm bộ nhớ để lưu trữ cây PDB bằng cách sử dụng cơ sở dữ liệu chiếu giả tại tất cả các nút trên cây PDB. Trong PDB ta chỉ lưu trữ danh sách các điểm của chuỗi hiện tại trong cơ sở dữ liệu chuỗi ban đầu.

Ví dụ: xét cơ sở dữ liệu với *minsup = 1* ta được:

Diagram

Description automatically generated

Hình 6: Ví dụ về cấu trúc cây PDB

* 1. Khai thác tập phổ biến với vector bit động
     1. Giới thiệu

DBV là phương pháp dựa theo hướng tiếp cận của BitTable. BitTable là một cách để nén dữ liệu. Mỗi item chiếm |T| bits, được gọi là vectơ bit. Trong đó |T| là số lượng giao dịch trong D. Vectơ bit của một itemset XY hình thành từ hai itemsets X và Y có thể được suy ra dễ dàng từ phép toán AND trên 2 bitvector X và Y. Vì độ dài của 2 vectơ bit là như nhau, kết quả sẽ là 1 vectơ bit với cùng chiều dài của |T| bits. Ưu điểm của phương pháp BitTable là có thể nén cơ sở dữ liệu trên một ma trận bit và sử dụng các phép toán thao tác trên bit để tính phần giao giữa các vectơ bit. Độ phổ biến của các itemset chính là số bit 1 trên vectơ bit nên việc này không tốn kém nhiều thời gian. Tuy nhiên, trong thực tế vectơ bit của một item hay itemset có thể chứa rất nhiều số 0 nên có thể loại bỏ chúng để vectơ bit ngắn hơn nhằm làm giảm đi không gian lưu trữ và chi phí tính phần giao giữa các vectơ bit. Phương pháp sử dụng cấu trúc vectơ bit động (DBV) đã được đề xuất là giải pháp khá hiệu quả để có thể khắc phục hạn chế này.

* + 1. Cấu trúc DBV

Cấu trúc vector bit động gồm hai thành phần:

-pos: lưu vị trí của byte khác không đầu tiên trong vectơ bit. Lưu ý để thuận tiện cho việc lập trình, vị trí đầu tiên trong một vectơ bit được đặt là vị trí 0

- Bitlist: dãy các byte của vectơ bit sau khi đã loại bỏ các byte bằng không ở đầu và cuối vectơ bit.  
Ví dụ: giả sử có vectơ bit với 40 byte sau khi đã tính ra giá trị thập phân như sau:



Hình 7: Vecto bit với 40 byte

Table

Description automatically generated

Hình 8: Vecto bit đã tính ra giá trị thập phân

Khi đó, ta có DBV với pos=10 và Bitlist = {5, 3, 8, 0, 0, 7, 6, 3, 2, 7, 6, 5}  
Theo thí dụ trên, nếu sử dụng cấu trúc BitTable ta phải cần đến 40 bytes để lưu trữ, trong khi DBV chỉ chiếm không gian 14 bytes bộ nhớ (12 bytes cho vectơ bit, 2 bytes cho vị trí của byte khác không đầu tiên). Vì vậy, cấu trúc DBV cần ít không gian lưu trữ hơn so với tiếp cận dựa trên cấu trúc BitTable.

* + 1. Thuật toán tìm phần giao giữa hai vector bit động

Ý tưởng của thuật toán được thực hiện như sau:  
- Tìm pos lớn nhất trong các pos của hai vectơ bit động  
- Từ vị trí này thực hiện phép toán & giữa các byte. Nếu kết quả là 0 thì tăng pos lên 1 cho đến khi có kết quả khác 0 đầu tiên.  
- Từ vị trí này, tiếp tục thực hiện phép toán & cho các byte của vectơ bit cho đến khi phần còn lại là toàn 0. Kết quả này tạo ra một vectơ bit cho DBV.

Ví dụ: Giả sử cần tính phần giao giữa hai DBV: {10, {5, 3, 8, 0, 0, 7, 6, 3, 0, 7, 6, 5}} và {13, {4, 3, 0, 1, 0, 4, 6, 0, 0, 5, 1, 3}} như hình dưới

Diagram

Description automatically generated

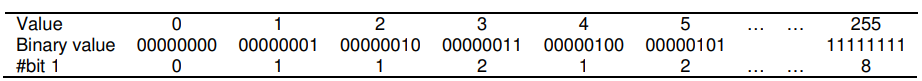
Hình 9: Phần giao giữa hai DBV

Trước hết, ta lấy pos lớn nhất là 13. Tại vị trí pos =13, ta có 0 & 4 = 0 nên pos = 14, tương tự thì 0 & 3, 7 & 0, 6 & 1, 3 & 0, 2 & 4 đều bằng 0 nên pos = 19. Tại byte kế tiếp, do 7 & 6 = 6 nên Bitlist kết quả sẽ chứa 6. Do toàn bộ các byte còn lại đều bằng 0 nên DBV kết quả là {19,{6}}.

* + 1. Tính toán nhanh với độ hỗ trợ của một itemset từ một DBV

Phương pháp dựa trên BitTable có thể tiêu tốn nhiều thời gian hơn để tính toán giao điểm giữa các vectơ bit và để đếm số lượng bit ‘1’ trong vectơ bit kết quả hơn so với cách tiếp cận DBV được đề xuất. Chúng ta có thể sử dụng một bảng tra cứu với 256 phần tử để tăng tốc độ đếm. Bảng ánh xạ từng số có thể được biểu diễn bằng một byte với số bit ‘1’ trong byte.

Với sự hỗ trợ của bảng tra cứu, số lượng bit ‘1’ trong mỗi byte của vectơ bit kết quả được biết ngay lập tức. Do đó, độ phức tạp để đếm một tập phổ biến là O(m), trong đó m là số bit trong DBV của nó. Cách tiếp cận được đề xuất hiệu quả hơn so với cách tiếp cận dựa trên BitTable trước đây.



Hình 10: Bảng tra cứu bitvector

* 1. Thuật toán BIDE
     1. Vấn đề đặt ra

Theo các nghiên cứu trước đây đã nêu ra rằng một thuật toán khai thác mẫu thường xuyên (cả tập phổ biến và chuỗi) không nên khai thác tất cả các mẫu thường xuyên mà chỉ nên khai thác các mẫu đóng vì với việc khai thác mẫu đóng đã cho kết quả không chỉ nhỏ gọn hơn mà còn thiết lập đầy đủ và hiệu quả tốt hơn. Tuy nhiên các thuật toán khai thác mẫu tuần tự đóng trước đây hoạt động theo mô hình maintenance-and-test (duy trì và kiểm tra) các ứng viên rất tốn kém về thời gian chạy và không gian thực thi khi ngưỡng hộ trợ thấp hoặc mô hình trở nên dài hơn.

Chính vì các lý do này, một thuật toán hiệu quả để khai thác các chuỗi đóng phổ biến mà không cần các ứng viên bảo trì. Nó áp dụng một sơ đồ kiểm tra trình tự đóng có tên là BIDE (BI-Directional Extension) và lược bỏ không gian tìm kiếm sâu hơn so với các thuật toán trước bằng cách sử dụng phương pháp cắt BackScan và kỹ thuật tối ưu hóa ScanSkip. Các nghiên cứu đã cho thấy rằng với các tập dữ liệu thực thưa thớt và dày đặc đều chứng mình rằng BIDE làm tốt hơn đáng kể so với các thuật toán trước do BIDE sử dụng bộ nhớ ít hơn và tốc độ nhanh hơn, đồng thời nó cũng có thể là mở rộng tuyến tính về kích thước của cơ sở dữ liệu.

* + 1. Một số định nghĩa

Cho I = {} là tập hợp các mục riêng biệt, một chuỗi S là một chuỗi các sự kiện có thứ tự được biểu thị là <}> với là một item . Một chuỗi có chiều dài k sẽ được gọi là k-sequence. Ví dụ: AABCCA là chuỗi 6-sequence.

Cho chuỗi và chuỗi . được gọi là một chuỗi con của () khi và chỉ khi tồn tại những số nguyên (theo thứ tự tuần tự) sao cho , , …, . Ví dụ: chuỗi là chuỗi con của chuỗi nhưng không phải là chuỗi con của .

Cơ sở dữ liệu chuỗi SDB là một tập hợp các bộ dữ liệu có dạng với là định danh và là chuỗi các itemset.

Độ hỗ trợ (support) ký hiệu là bao gồm độ hỗ trợ tuyệt đối và độ hỗ trợ tương đối.

* Độ hỗ trợ tuyệt đối của một mẫu tuần tự là tổng số chuỗi có trong cơ sở dữ liệu có chứa . Ký hiệu
* Độ hỗ trợ tương đối của mẫu tuần tự là tỉ lệ phần trăm chuỗi chứa trong cơ sở dữ liệu có chứa .

Cho ngưỡng hỗ trợ minSup, một chuỗi là chuỗi tuần tự phổ biến trong SDB khi . Nếu phổ biến và không tồn tại chuỗi cha nào của với cùng độ hỗ trợ , ta gọi là một chuỗi tuần tự đóng. Vấn đề của khai thác chuỗi tuần tự đóng là tìm để hoàn thành một tệp các chuỗi tuần đóng cho dữ liệu đầu vào SDB với ngưỡng hỗ trợ minsup được cho.

Ví dụ: cho cơ sở dữ liệu với minsup = 2

|  |  |
| --- | --- |
| SID | Sequence |
| 1 | C A A B C |
| 2 | A B C B |
| 3 | C A B C |
| 4 | A B B C A |

Bảng 31: Cơ sở dữ liệu ví dụ

Từ cơ sở dữ liệu trên ta tìm được 17 ứng viên chuỗi tuần tự là {A:4, AA:2, AB:4, ABB:2, ABC:4, AC:4, B:4, BB:2, BC:4, C:4, CA:3, CAB:2, CABC:2, CAC:2, CB:3, CBC:2, CC:2} tuy nhiên chỉ có 6 ứng viên thỏa mãn chuỗi tuần tự đóng là {AA:2, ABB:2, ABC:4, CA:3, CABC:2, CB:3}. Ta có thể thấy rằng khi xét chuỗi CBC:2 nó là con của chuỗi CABC:2 và có cùng độ hỗ trợ, vì vậy CBC bị loại bỏ.

* + 1. BIDE

BIDE là thuật toán khai thác mẫu tuần tự đóng trong cơ sở dữ liệu trình tự, được đề xuất bởi Wang và cộng sự năm 2007. Các mẫu tuần tự đóng thường nhỏ hơn nhiều so với tập các mẫu tuần tự và không có các thông tin nhỏ. Hơn nữa, việc tìm kiếm các mẫu tuần tự đóng thường hiệu quả hơn nhiều so với việc khám phá tất cả các mẫu.

* + - 1. Liệt kê trình tự phổ biến

Giả sử rằng một từ được sắp xếp theo thứ tự từ điển và cách nhau bởi dấu giữa tập hợp các mục I trong cơ sở dữ liệu trình tự ở đầu vào. Ví dụ một tập có thứ tự là , theo khái niệm, không gian tìm kiếm hoàn chỉnh của khai phá trình tự sẽ tạo thành một chuỗi cây và được xây dựng theo cách nút gốc của cây sẽ nằm ở vị trí cao nhất và được gán nhãn bằng , ta có thể mở rộng một nút N tại mức L trong cây bằng cách thêm một I vào để lấy một nút con ở mức L+1 và các nút con của nút N sẽ được tạo và sắp xếp theo thứ tự từ điển. Bằng cách loại bỏ các trình tự không phổ biến trong cây trình tự, các nút còn lại sẽ tạo thành một cây trình tự phổ biến.

Diagram

Description automatically generated

Hình 11: Minh họa BIDE

Các thuật toán khai thác mẫu phổ biến trước đây đã giải thích rằng tìm kiếm theo chiều sâu hiệu quả hơn trong khai thác các mẫu dài so với tìm kiếm theo chiều rộng. BIDE duyệt cây trình tự một cách nghiêm ngặt theo thứ tự tìm kiếm chiều sâu. Như hình trên ta sẽ có các chuỗi thường xuyên sẽ được khai thác và được báo cáo theo thứ tự như sau: A:4, AA:2, AB:4, ABB:2, ABC:4, AC:4, B:4, BB:2, BC:4, C:4, CA:3, CAB:2, CABC:2, CAC:2, CB:3, CBC:2, CC:2.

Một nút nhất định trong cây trình tự có thể được coi như là một chuỗi tiền tố, từ đó tập hợp các nút con của nó có thể được tạo ra bằng cách thêm một mục vào. Một số mục có thể không phổ biến liên quan đến chuỗi tiền tố tương ứng. Có một phương pháp nổi tiếng để tính các mục phổ biến của một tiền tố nhất định là xây dựng cơ sở dữ liệu dự kiến cho tiền tố và quét nó để đếm các mục. Hai loại phương pháp chiếu đã được sử dụng trong quá khứ là phép chiếu vật lý và phép chiếu giả. Do phép chiếu vật lý cần phải xây dựng một cách vật lý cơ sở dữ liệu dự kiến có điều kiện, phương pháp này không hiệu quả về không gian và thời gian chạy do chi phí cấp phát và giải phóng bộ nhớ. Trong BIDE, chỉ sử dụng phương pháp chiếu giả để tìm tập hợp các mục phổ biến của một tiền tố nhất định và sử dụng nó để phát triển các tiền tố tương ứng.

Định nghĩa 1 (Phiên bản đầu tiên của chuỗi tiền tố): cho một chuỗi đầu vào S có chứa một tiền tố 1-sequence , chuỗi con từ S lần đầu xuất hiện của item trong S được gọi là trường hợp đầu tiên của tiền tố 1-sequence trong S. Một cách đệ quy ta có thể xác định được trường hợp đầu tiên của (i+1)-sequence từ trường hợp đầu tiên của 1-sequence như thứ tự từ đầu của S đến lần xuất hiện đầu tiên của , điều này cũng xảy ra sau trường hợp đầu tiên của i-sequence . Ví dụ: trường hợp đầu tiên của chuỗi tiền tố AB trong chuỗi CAABC là CAAB.

Định nghĩa 2 (trình tự dự kiến của một chuỗi tiền tố): cho một chuỗi đầu vào S có chứa một tiền tố i-sequence , phần còn lại của S sau khi chùng ta loại bỏ trường hợp đầu tiên của tiền tố i-sequence trong S được gọi là trình tự dự kiến. Ví dụ: trình tự dự kiến của chuỗi tiền tố AB trong ABBCA là BCA.

Định nghĩa 3 (cơ sở dữ liệu dự kiến của một chuỗi tiền tố): cho một cơ sở dữ liệu chuỗi đầu vào SDB bộ hoàn chỉnh trình tự dự kiến trong SDB là một chuỗi tiền tố được gọi là cơ sở dữ liệu dự kiến.

* + - 1. Kiểm tra cơ chế đóng BI-Directional Extension

Thuật toán liệt kê phổ biến chỉ có thể được sử dụng để khai thác toàn bộ các chuỗi thường xuyên thay vì các chuỗi đóng thường xuyên. Thông thường khi nhận được một chuỗi tiền tố thường xuyên mới, ta cần thực hiện kiểm tra mẫu đóng nhằm đảm bảo rằng nó đã thực sự đóng, tức là sẽ không bị hấp thụ bởi các chuỗi cha của nó với cùng độ hỗ trợ. Các thuật toán khai thác mẫu đóng hiện nay như: CLOSET, CHARM, TFP, CloSpan cần duy trì tập hợp các mẫu đóng phổ biến đã được khai thác trong bộ nhớ và thực hiện kiểm tra phần tử, kiểm tra một mẫu mới tìm thấy có thể được hấp thụ bởi mẫu đóng phổ biển đã được khai thác hay không và kiểm tra chuỗi cha xem mẫu mới tìm thấy có thể hấp thụ một số ứng viên mẫu đóng đã được khai thác hay không.

Theo định nghĩa chuỗi tuần tự đóng, nếu một n-sequence, không đóng, nó phải tồn tại ít nhất một sự kiện , nó được sử dụng để mở rộng chuỗi S để lấy thêm một chuỗi mới có cùng độ hỗ trợ. Chuỗi S có thể được mở rộng bằng ba cách:

* + và sup ( = sup(S)
  + và sup ( = sup(S)
  + và sup ( = sup(S)

Ở trường hợp đầu tiên, xuất hiện sau , được gọi là sự kiện mở rộng chuyển tiếp (forward-extension item) và là một chuỗi mở rộng chuyển tiếp (forward-extension sequence) của S. Ở trường hợp thứ 2 và thứ 3, sự kiện xuất hiện trước , ta gọi là một sự kiện mở rộng ngược (backward-extension event) và là một chuỗi mở rộng ngược (backward-extention sequence) của S.

Định lý 1 (kiểm tra cơ chế đóng BIDE): nếu không tồn tại sự kiện mở rộng chuyển tiếp nào hoặc sự kiện mở rộng ngược, phải là một trình tự đóng, nếu không phải được đóng. Từ định lý này, ta biết được rằng để đánh giá xem một chuỗi tiền tố phổ biến có bị đóng hay không, vì vậy chúng ta cần kiểm tra xem có bất kỳ sự kiện mở rộng chuyển tiếp hay sự kiện mở rộng ngược hay không.

Định nghĩa con 1 (Kiểm tra sự kiện mở rộng chuyển tiếp): đối với chuỗi tiền tố , tập hợp đầy đủ các sự kiện mở rộng chuyển tiếp của nó tương đương với một tập hợp các mục thường xuyên cục bộ của nó có độ hỗ trợ bằng sup().

Định nghĩa 4 (trường hợp cuối cùng của chuỗi tiền tố): cho một chuỗi đầu vào S có chứa một tiền tố i-sequence , phần cuối cùng của chuỗi tiền tố trong S được gọi là thứ tự từ đầu đến lần xuất hiện cuối cùng của item trong s. Ví dụ: trường hợp xuất hiện cuối cùng của chuỗi tuần tự AB trong chuỗi ABBCA là ABB.

Định nghĩa 5 (i-th xuất hiện cuối cùng trong lần cuối cùng của một chuỗi tiền tố): với chuỗi đầu vào S có chứa tiền tố n-sequence, i-th xuất hiện lần cuối cùng của chuỗi tiền tố trong S được biểu thị là và được định nghĩa một cách đệ quy là:

* + Nếu i = n, nó là lần xuất hiện cuối cùng của trong trường hợp cuối cùng của chuỗi tiền tố trong S.
  + Nếu , nó là lần xuất hiện cuối cùng của trong trường hợp cuối cùng của tiền tố trong S trong khi phải xuất hiện trước

Ví dụ: S = CAABC và = AB, lần thứ nhất (1st) xuất hiện lần cuối cùng của tiền tố trong S là A thứ hai trong S.

Định nghĩa 6 (i-th khoảng thời gian tối đa của một chuỗi tiền tố): với chuỗi đầu vào S có chứa tiền tố n-sequence, i-th của khoảng thơi gian tối đa của tiền tố trong S được biểu thị là:

* + Nếu , nó là một phần trình tự giữa phần cuối của trường hợp đầu tiên của tiền tố trong S và lần i-th xuất hiện lần cuối cùng của tiền tố .
  + Nếu i = 1, nó là một phần trình tự trong S được định vị trước lần xuất hiện cuối cùng trong lần đầu tiên của tiền tố .

Ví dụ: nếu S = ABCB và chuỗi tiền tố = AB, khoảng thời gian tối đa thứ hai của trong S là BC, trong khi khoảng thời gian tối đa thứ nhất của tiền tố trong S là rỗng.

Định nghĩa con 2 (kiểm tra sự kiện mở rộng ngược): cho một chuỗi tiền tố là n-sequence . Nếu và tồn tại một item mà nó xuất hiện ở mỗi 1-th khoảng thời gian tối đa cuối một chuỗi tiền tố trong SDB, là sự kiện mở rộng ngược của tiền tố . Nếu không thì, với mỗi i (), nếu không tìm thấy bất kỳ item nào xuất hiện trong mỗi i-th khoảng thời gian tối đa của tiền tố trong SDB thì nó sẽ không phải là sự kiện mở rộng ngược của tiền tố .

Ví dụ với BIDE: giả sử rằng = AC:4, ta có thể dự đoán rằng item B sẽ xuất hiện ở khoảng thời gian tối đa thứ hai của chuỗi này và như kết quả ở ví dụ trên thì nó không đóng. Nếu = ABC:4, ta sẽ không tìm thấy bất cứ item B mở rộng ngược nào cho nó và cũng không có mở rộng chuyển tiếp nào cho nó, vì vậy ABC là một chuỗi tuần tự đóng.

* + - 1. Phương pháp cắt tỉa không gian tìm kiếm BackScan

Khi tìm thấy một chuỗi phổ biến mới bằng thuật toán liệt kê chuỗi thường xuyên, ta có thể sử dụng BIDE để kiểm tra xem nó có đóng hay không để tạo ra tập hợp hoàn chỉnh của chuỗi phổ biển không dự phòng. Mặc dù sơ đồ kiểm tra đóng có thể dẫn đến tập kết quả nhỏ gọn hơn tuy nhiên nó không thể cải thiện hiệu quả khai thác. Như hình, toàn bộ cây con dưới nút B:4 không chứa chuỗi tuần tự đóng nào, điều đó có nghĩa rằng không có hi vọng không phát triển tiền tố B:4 để phát sinh ra bất cứ chuỗi tuần tự đóng nào tại đây. Nếu ta phát hiện các chuỗi tiền tố như vậy và ngừng phát triển nó thì không gian tìm kiếm sẽ giảm đi đáng kể.

Việc cắt tỉa không gian tìm kiếm trong khai thác tuần tự phổ biến thường phức tạp hơn so với khai thác tập hợp các itemset đóng. Thông thường, thuật toán khai thác itemset đã đóng dựa trên tìm kiếm theo chiều sâu như CLOSET có thể ngừng phát triển tập hợp tiền tố sau khi nhận thấy rằng tập hợp các itemset này có thể bị hấp thụ bởi tập hợp các itemset đã đóng và đã được khai thác. Tuy nhiên, một thuật toán khai thác theo tuần tự đóng không thể làm như vậy. Ví dụ: với chuỗi tuần tự ABC:4, một chuỗi tiền tố C:4 sẽ bị hấp thụ bởi một tuần tự đã được khai thác là ABC:4, nhưng vẫn không thể ngừng phát triển C:4. Như hình, ta vẫn có thể phát triển 3 chuỗi tuần tự đóng từ C:4 là CA:3, CABC:2, CB:3. Vấn đề phức tạp này là do nhiều trường hợp cùng một item trong chuỗi và thứ tự thời gian giữa các sự kiện trong một trình tự.

Định nghĩa 7 (i-th lần xuất hiện cuối cùng trong lần đầu tiên của một chuỗi tiền tố): với chuỗi đầu vào S có chứa tiền tố n-sequence, , i-th xuất hiện cuối cùng trong lần đầu tiên của tiền tố trong S được biểu thị là và định nghĩa một các đệ quy rằng:

* + Nếu i = n, nó là lần xuất hiện cuối cùng của trong lần xuất hiện đầu tiên của tiền tố trong S.
  + Nếu , nó là lần xuất hiện cuối cùng của trong lần xuất hiện đầu tiên của tiền tố trong S trong khi phải xuất hiện trước

Ví dụ: nếu S = CAABC và = CA, lần thứ hai xuất hiện cuối cùng trong lần đầu tiên của tiền tố trong S là A đầu tiên.

Định nghĩa 8 (i-th khoảng thời gian bán tối đa của một chuỗi tiền tố): với chuỗi đầu vào S có chứa tiền tố n-sequence, , i-th khoảng thời gian bán tối đa của một chuỗi tiền tố trong S được định nghĩa rằng:

* + Nếu , nó là một phần của trình tự giữa phần cuối của trường hợp đầu tiên của tiền tố trong S và i-th xuất hiện cuối cùng trong lần đầu tiên của chuỗi tiền tố .
  + Nếu i = 1, nó một phần trình tự trong S được xác định vị trí trước lần xuất hiện cuối cùng trong lần đầu tiên của tiền tố .

Ví dụ: nếu S = ABCB và chuỗi tiền tố = AC, khoảng thời gian thứ hai bán tối đa của chuỗi tiền tố AC trong S là B, trong khi khoảng thời gian thứ nhất bán tối đa của chuỗi tiền tố AC trong S là rỗng.

Định lý 2 (cắt bỏ không gian tìm kiếm BackScan): với chuỗi tiền tố là một n-sequence, . Nếu và tồn tại một item nó xuất hiện ở mỗi i-th khoảng thời gian bán tối đa của chuỗi tiền tố trong SDB, thì ta có thể dừng phát triển một cách an toàn tiền tố .

Bởi vì item xuất hiện ở mỗi i-th khoảng thời gian bán tối đa của tiền tố trong SDB, ta có thể có một tiền tố mới hoặc và cả hai và sup() = sup(. Bất cứ item phổ biến nào trong cũng là item phổ biến trong , đồng thời và sup() = sup(. Điều đó có nghĩa rằng là không có hi vọng trong việc khai thác chuỗi tuần tự đóng với tiền tố .

* + - 1. Kỹ thuật tối ưu hóa ScanSkip

Sơ đồ kiểm tra đóng ở trên cần quét ngược và tập hợp các khoảng thời gian tối đa của một tiền tố nhất định, đây là hoạt động tốn kém nhất trong BIDE. Giả sử rằng chuỗi tiền tố là với ngưỡng hộ trợ sup(). Với mỗi i, với {}là sup() 1-th khoảng tối đa của . Bằng cách quét k-th và i-th khoảng tối đa chúng ta có thể nhận được một tập hợp các item được biểu thị là . Tập hợp các item xuất hiện trong một i-th khoảng tối đa của và được biểu thị là .

Trong sơ đồ kiểm tra tuần tự đóng, ta chỉ quan tâm đến việc liệu rằng có rỗng hay không. Nếu ta tìm thấy sau đó, ta duyệt m đầu tiên của i-th khoảng tối đa, giao điểm của tập các item xuất hiện trong mục m đầu tiên i-th xuất hiện tối đa trở nên rỗng thì ta sẽ không cần phải quét (sup() -m) i-th giá trị tối đa bởi vì ta biết rằng mở rỗng. Ta gọi sự tối ưu hóa này là kỹ thuật ScanSkip. Tương tự, trong phương pháp cắt bỏ không gian tìm kiếm BackScan, BIDE cần quét một số khoảng thời gian bán tối đa của một chuỗi tiền tố. Kỹ thuật ScanSkip cũng có thể được sử dụng để tăng tốc độ cắt bỏ không gian tìm kiếm BackScan.

Ví dụ:

|  |  |
| --- | --- |
| SID | Sequence |
| 1 | C A A B C |
| 2 | A B C B |
| 3 | C A B C |
| 4 | A B B C A |

Bảng 32: Bảng cơ sở dữ liệu ví dụ cho kỹ thuật ScanSkip

Giả sử chuỗi tiền tố là ABC:4. Tập khoảng tối đa thứ 3 của ABC là {Ø, Ø, Ø, B}, sau khi duyệt lần đầu tiên khoảng tối đa thứ 3 ta thấy rằng nó không chứa item nào, ta sẽ biết rằng nó sẽ không có item nào xuất hiện trong mỗi lần thứ 4 khoảng tối đa thứ 3 của tiền tố ABC:4 mà không cần duyệt lần cuối khoảng tối đa thứ 3. Một tập khoảng tối đa thứ 2 của ABC:4 là {A, Ø, Ø, B}, sau khi duyệt hai lần đầu tiên của khoảng tối đa thứ hai, ta biết rằng sẽ không có item nào xuất hiện tại ở mỗi lần thứ 4 của khoảng tối đa thứ 2 của tiền tố ABC:4. Tương tự, một tập khoảng tối đa thứ nhất của ABC:4 là {CA, Ø, C, Ø}, ta có thể bỏ qua việc duyệt 2 lần cuối của khoảng tối đa thứ nhất của tiền tố ABC:4.

* + - 1. Thuật toán BIDE

Text

Description automatically generated

Hình 12: Mã giả thuật toán BIDE

Hình trên là mô tả của thuật toán BIDE. Đầu tiên nó sẽ quét cơ sở dữ liệu để tìm 1-sequences sau đó xây dựng cơ sở dữ liệu dự kiến giả cho mỗi 1-sequence, coi mỗi dãy số 1-sequence là một tiền tố và sử dụng phương pháp cắt tỉa BackScan để kiểm tra xem nó có thể được cắt bớt hay không, nếu không thì hãy tính số lượng các item mở rộng ngược và gọi chương trình con BIDE . Thuật toán con gọi lại đệ quy chính nó và hoạt động như sau: cho tiền tố , quét cơ sở dữ liệu dự kiến của nó một lần để tìm các item phổ biến sau đó tính số item mở rộng chuyển tiếp, đầu ra là một chuỗi phổ biến đóng, phát triển với mỗi item phổ biến theo thứ tự từ điển để nhận tiền tố mới và xây dựng cơ sở dữ liệu dự kiến giả cho tiền tố mới, với mỗi tiền tố mới, đầu tiên ta kiểm tra nếu nó có thể cắt tỉa hay không, nếu không ta tính số item mở rộng ngược của nó và gọi lại bide. Ta cần chỉ ra rằng cả hai chương trình con BackScane() và backward extension check() đều sử dụng ScanSkip để tăng tốc quá trình khai thác.

* + 1. Đánh giá hiệu suất

Một thuật toán khai thác chuỗi tuần tự đóng thường được thiết kế đúng cách như BIDE hoặc CloSpan có thể vượt trội đáng kể hơn so với hai thuật toán khai thác chuỗi phổ biến PrefixSpan và SPADE khi ngưỡng hộ trợ thấp. BIDE sử dụng ít bộ nhớ hơn và nhanh hơn CloSpan, đồng thời BIDE có khả năng mở rộng tuyến tính về quy mô và cơ sở. Phương pháp cắt tỉa BackScan và kỹ thuật ScanSkip rất hiệu quả trong việc nâng cao hiệu suất.

* + - 1. Môi trường thử nghiệm và bộ dữ liệu

Trong thí nghiệm này, ta so sánh BIDE với hai thuật toán khai thác mẫu tuần tự là PrefixSpan và SPADE và một thuật toán khai thác mẫu tuần tự đóng CloSpan.

Bộ dữ liệu thứ nhất là Gazalle, rất thưa thớt nhưng có một số chuỗi phổ biến đóng rất dài với ngưỡng hỗ trợ thấp. Dữ liệu này chứa 29369 dữ liệu nhấp chuột của khách hàng, 87546 lượt xem trang và 1423 các trang web.

Bộ dữ liệu thứ hai là Snake, khá dày đặt và có thể tạo ra nhiều chuỗi phổ biến đóng với ngưỡng hỗ trợ 60%. Dữ liệu này gồm 175 chuỗi protein rắn và 20 item protein riêng biệt.

Bộ dữ liệu thứ ba rất dày đặc vì thế có thể khai thác một lượng lớn các chuỗi phổ biến đóng với ngưỡng hỗ trợ đến 90%. Dữ liệu này bao gồm 190 chuỗi protein và 21 item protein riêng biệt.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset | #seq | #items | avg. seq. len. | max. seq. len. |
| Gazelle | 29369 | 1423 | 3 | 6511 |
| Snake | 175 | 20 | 67 | 121 |
| Pi | 190 | 21 | 258 | 757 |

Bảng 33: Bảng chi tiết dữ liệu.

* + - 1. Kết quả thực nghiệm

Các nghiên cứu cho thấy rằng thuật toán khai thác mẫu tuần tự đóng có thể hoạt động tốt hơn thuật toán khai thác mẫu tuần tự một bậc. Với một tập dữ liệu có độ hỗ trợ cao, thuật toán khai thác theo mẫu tuần tự đóng có thể thua một số thuật toán khai thác mẫu tuần tự.

Diagram

Description automatically generated with medium confidence

Hình 13: So sánh thời gian thực thi giữa BIDE, CloSpan, SPADE và PrefixSpan trên dữ liệu Gazelle.

**Diagram

Description automatically generated with medium confidence**

Hình 14: So sánh khả năng sử dụng bộ nhớ giữa BIDE, CloSpan, SPADE và PrefixSpan trên dữ liệu Gazelle.

Hình trên cho ta thấy rằng với độ hỗ trợ lớn hơn 0.03% PrefixSpan và SPADE đều tốt hơn so với CloSpan và BIDE, tuy nhiên khi độ hỗ trợ xuống đến 0.02% thì hai thuật toán khai thác mẫu tuần tự đóng sẽ hoạt động tốt hơn rất nhiều so với hai thuật toán kia. Đồng thời ta cũng thấy rằng là BIDE luôn hoạt động tốt hơn so với BIDE với các ngưỡng hỗ trợ khác nhau.

Histogram

Description automatically generated with low confidence

Hình 15: Sự phân bố trên dữ liệu Gazelle

Khi so sánh BIDE với CloSpan, đầu tiên ta so sánh bằng cách sử dụng tập dữ liệu Gazelle, hình trên mô tả sự phân bố của số lượng các chuỗi tuần tự đóng so với độ dài của các mẫu tuần tự phổ biến đóng với các ngưỡng hỗ trợ từ 0.02% đến 0.01%. Từ hình trên, ta có thể thấy rằng nhiều chuỗi dài đóng có thể được phát hiện cho tập dữ liệu thưa thớt như dữ liệu này. Ví dụ: với mức hỗ trợ là 0.01% chuỗi đóng phổ biến dài nhất có độ dài 127.

Chart, line chart

Description automatically generated

Hình 16: Thời gian chạy giữa BIDE và CloSpan trên dữ liệu Gazelle.

**Chart, line chart

Description automatically generated**

Hình 17: Khả năng sử dụng bộ nhớ giữa BIDE và CloSpan trên dữ liệu Gazelle.

Hình trên là so sánh thời gian chạy và thời gian sử dụng bộ nhớ của CloSpan và BIDE, ta có thể thấy rằng BIDE luôn chạy nhanh hơn và sử dụng ít bộ nhớ hơn CloSpan. Ví dụ: với độ hỗ trợ 0.01%, BIDE có thể nhanh hơn so với CloSpan một bậc trong khi nó sử dụng bộ nhớ ít hơn một bậc.

Chart

Description automatically generated

Hình 18: Phân bố số lượng chuỗi đóng trong dữ liệu Snake.

Hình trên mô tả sự phân bố số lượng chuỗi phổ biến đóng so với độ dài của chuỗi tuần tự đóng với tập dữ liệu Snake. Ta có thể thấy rằng với tập dữ liệu dày đặc, có thể khai thác rất nhiều chuỗi đóng với độ dài trung bình từ 6 đến 12 với một ngưỡng hỗ trợ không quá thấp từ 50% đến 70%.

Chart, line chart

Description automatically generated

Hình 19: Thời gian chạy giữa BIDE và CloSpan trên dữ liệu Snake.

**Chart, line chart

Description automatically generated**

Hình 20: Khả năng sử dụng bộ nhớ giữa BIDE và CloSpan trên dữ liệu Snake.

Hình trên cho ta thấy ở ngưỡng hộ trợ cao, BIDE chậm hơn nhiều so với CloSpan, nhưng khi mức hộ trợ không lớn hơn 60% BIDE sẽ vượt trội hơn so với CloSpan. Ví dụ: với độ hỗ trợ 50%, BIDE nhanh hơn khoảng 40 lần so với CloSpan. Ta cũng có thể thấy BIDE sử dụng bộ nhớ ít hơn 2 bậc so với CloSpan trong hầu hết các trường hợp.

Chart, diagram

Description automatically generated

Hình 21: Phân bố số lượng chuỗi đóng trong dữ liệu Pi.

Chart, line chart

Description automatically generated

Hình 22: Thời gian chạy giữa BIDE và CloSpan trên dữ liệu Pi.

**Chart, line chart

Description automatically generated**

Hình 23: Khả năng sử dụng bộ nhớ giữa BIDE và CloSpan trên dữ liệu Pi.

Với tập dữ liệu dày đặc, với ngưỡng hỗ trợ cao 90% có thể có một số lượng lớn các chuỗi ngắn phổ biến đóng với độ dài nhỏ hơn 10. Hình 13 cho ta thấy rằng với độ hỗ trợ cao nhưng hai thuật toán có hiệu suất rất giống nhau, CloSpan nhanh hơn BIDE một ít nhưng khi độ hỗ trợ không quá 88% thì BIDE sẽ vượt trội so với CloSpan. Đồng thời BIDE sử dụng ít bộ nhớ hơn rất nhiều so với CloSpan, với ngưỡng hỗ trợ 88% BIDE sử dụng bộ nhớ ít hơn CloSpan 2 bậc.

Chart

Description automatically generated

Hình 24: Khả năng mở rộng của BIDE về thời gian chạy.

**Chart

Description automatically generated**

Hình 25: Khả năng mở rộng của BIDE về mức độ sử dụng bộ nhớ.

Kiểm tra khả năng mở rộng của BIDE trong cả thời gian chạy và mức độ sử dụng bộ nhớ với cả ba bộ dữ liệu. Mặc dù 3 bộ dữ liệu có tính năng khác nhau, BIDE cho thấy khả năng mở rộng tuyến tính trong cả thời gian chạy và sử dụng bộ nhớ so với số lượng chuỗi càng tăng lên của các bộ dữ liệu này. Ví dụ: với tập dữ liệu Snake với mức độ hỗ trợ nhất định là 60% thời gian chạy tăng từ 807s lên 11906s và mức độ sử dụng bộ nhớ của nó tăng từ 1883MB lên 21784MB khi số chuỗi tăng lên 16 lần.

Chart, line chart

Description automatically generated

Hình 26: Hiệu quả của phương pháp cắt tỉa BackScan trên dữ liệu Gazelle

**Chart, line chart

Description automatically generated**

Hình 27: Hiệu quả của kỹ thuật ScanSkip trên dữ liệu Snake

Hiệu quả của các kỹ thuật tối ưu hóa, hình 26 cho ta thấy tính hiệu quả của phương pháp cắt tỉa BackScan trong việc cắt giảm không gian tìm kiếm và tăng tốc quá trình khai thác. Đối với tập dữ liệu Gazelle có ngưỡng hỗ trợ là 0.02%, nó có thể đưa ra một số lệnh tăng cường độ cho hiệu suất. Tính hiệu quả của phương pháp cắt tỉa BackScan đảm bảo rằng BIDE chỉ dựa trên phương pháp cắt tỉa duy nhất này có thể hoạt động tốt hơn đáng kể trong hầu hết các trường hợp thuật toán CloSpan sử dụng các phương pháp cắt tỉa khác.

* + 1. Kết luận

Nhiều nghiên cứu đã chỉ ra rằng khai thác mẫu tuần tự đóng có hiệu quả tương tự như khai thác mô hình phổ biến khác nhưng dẫn đến tập kết quả nhỏ gọn và hiệu quả hơn đáng kể. Từ nghiên cứu này, ta có thể thấy điều này đúng khi số lượng các mẫu phổ biến là rất lớn, trong trường hợp đó số lượng các mẫu phổ biến đóng cũng có khả năng rất lớn.

Các thuật toán khai thác mẫu tuần tự đóng đã phát triển trước đây dựa trên tập hợp các mẫu đóng phổ biển hoặc ứng viên để kiểm tra xem một mẫu phổ biến mới tìm thấy có đóng hay không hoặc nó có thể làm mất hiệu lực của ứng viên đóng đã được khai thác hay không. Bởi vì tập hợp các mẫu phổ biến đóng đã được khai thác sẽ tiếp tục trong quá trình khai thác, nó sẽ không chỉ tiêu tốn bộ nhớ nhiều hơn mà còn dẫn đến không hiệu quả do không gian tìm kiếm ngày càng tăng để kiểm tra các mẫu đóng.

Chính vì thế một thuật toán mới để khai thác mẫu tuần tự đóng BIDE được đề xuất. Nó bỏ đi quá trình của mô hình duy trì và kiểm tra ứng viên, lược bỏ không gian tìm kiếm sâu hơn và kiểm tra việc đóng mẫu theo cách hiệu quả hơn trong khi tiêu tốn ít bộ nhớ hơn nhiều so với các thuật toán khai thác mẫu đóng trước. Nó không cần duy trì tập hợp các mẫu đóng đã được khai thác trước đó, do đó nó thay đổi tỉ lệ rất tốt về số lượng các mẫu đóng phổ biến. BIDE áp dụng áp dụng tìm kiếm theo chiều sâu nghiêm ngặt và có thể xuất ra các mẫu đóng phổ biến một cách trực tuyến.

Một tập hợp lớn các thử nghiệm trên một số tập dữ liệu thực với các tính năng khác nhau đã cho ta thấy hiệu quả của BIDE là BIDE sử dụng bộ nhớ ít hơn trong khi thực hiện nhanh hơn so với CloSpan. Nó cũng có khả năng mở rộng tuyến tính về số lượng trong cơ sở dữ liệu.

CHƯƠNG 3 – XÂY DỰNG CƠ SỞ LUẬT TUẦN TỰ CHO DỰ ĐOÁN HÀNH VI TRUY CẬP WEB

Khai thác sử dụng web được coi là ứng dụng chính của các kỹ thuật khai thác quy tắc tuần tự được áp dụng. Mục đích để truy xuất được các dữ liệu về các tùy chọn và hành vi sử dụng khi người dùng sử dụng web để biết được một danh sách theo thứ tự là các trang web hoặc các sản phẩm trong một web. Trong chương này, các kỹ thuật khai thác quy tắc tuần tự sẽ được giới thiệu.

* 1. Khái niệm dự đoán hành vi truy cập Web

Định nghĩa 1:

Gọi là tập hợp người dùng truy cập web với là địa chỉ IP của người dùng truy cập thứ *i* với *k* là số lượng của các địa chỉ *IP*.

Một tập hợp các phần tử , mỗi chuỗi tuần tự *Seq* là một danh sách có thứ tự , trong đó .

Gọi là chuỗi tuần tự của các trang web mà người dùng đã truy cập với địa chỉ với và *q* là số lượng của các web đã được truy cập.

Nhật ký truy cập web là một dãy các nhật ký trong đó v là số dòng của nhật ký và là dòng nhật ký thứ *j* được ghi nhận bởi người dùng có địa chỉ , truy cập vào trang web vào thời điểm .

Định nghĩa 2:

Một cơ sở dữ liệu truy cập web là tập hợp các chuỗi ∈ S , trong đó N là tổng số các chuỗi dữ liệu tuần tự có chứa trong một cơ sở dữ liệu tuần tự.

Dưới đây là cơ sở dữ liệu tuần tự WebLog trong đó có chứa 5 chuỗi tuần tự được truy cập bởi 5 người dùng với 5 địa chỉ IP khác nhau. Trong chuỗi tuần tự truy cập web thứ nhất có 6 trang web đã được truy cập bởi người dùng có địa chỉ theo thứ tự thời gian.

|  |  |  |
| --- | --- | --- |
| Địa chỉ IP | Chuỗi tuần tự truy cập web | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Bảng 34: Bảng ví dụ weblog

* 1. Giới thiệu Web Log

Web Log là tập hợp các nhật ký web được thu thập và lấy từ máy chủ web. Trong web log có chứa toàn bộ các dữ liệu được ghi nhận lại trong suốt quá trình một website hoạt động và cũng chứa các thông tin lỗi, dư thừa, nhiễu thông tin, gây hiểu nhầm và không đầy đủ. Hơn nữa, các tập tin web log cũng chứa các liên kết xấu, thiếu, truy cập lỗi hoặc không cho phép truy cập. Chính vì lý do này, việc thực hiện tiền xử lý để loại bỏ đi các thông tin không cần thiết và không có ý nghĩa để phục vụ dự đoán truy cập web là công việc rất quan trọng và cần thiết để loại bỏ các tập tin nhật ký dư thừa.

Ví dụ: một người lặp đi lặp lại các truy cập trên cùng một liên kết tại các thời điểm sẽ gây sự dư thừa trên web log. Các lỗi liên kết có dạng như 3xx, 4xx, 5xx.

Hạn chế đáng chú ý nhất cần xem xét khi sử dụng web log để dự đoán truy cập web là thời gian truy cập rất chậm do khối lượng thu thập dữ liệu trên các tập tin nhật ký là rất lớn và các thông tin dư thừa, không có ý nghĩa có trong web log sẽ làm thời gian thực thi càng tăng lên.

Vì vậy, việc giảm kích thước của web log và thu hẹp phạm vi, không gian dự đoán là cực kì quan trọng để giảm thời gian thực thi nhiều nhất có thể mà vẫn có thể đảm bảo được độ lớn và độ chính xác của thông tin truy cập web cần dự đoán.

* 1. Khái niệm Web Usage Mining
     1. Định nghĩa Web Usage Mining

Web Mining được chia thành ba loại: Web Usage Mining, Web Content Mining và Web Structure Mining. Trong đó Web Usage Mining là chủ đề quan trọng và được ứng dụng nhiều trong thực tế.

Web Usage Mining là một ứng dụng của các kỹ thuật khai phá dữ liệu để tìm ra các mẫu truy cập lịch sử thu được từ dữ liệu Web để hiểu và phục vụ các nhu cầu ứng dụng trên nền tảng Web. Hơn nữa, Web Usage Mining cũng là một kỹ thuật khai phá web được dùng để tìm và phân tích các mẫu truy cập web từ WebLog. Có thể xem Web Usage Mining là Web Log Mining.

Khai thác sử dụng web bao gồm ba giai đoạn:

* Tiền xử lý
* Khai phá mẫu
* Phân tích mẫu

Trong giai đoạn tiền xử lý, tập nhật ký máy chủ sẽ được tổ chức thành một tệp phiên người dùng để việc khám phá mẫu và phân tích trở nên hữu dụng nhất. Khi quá trình này được xử lý hoàn tất và thành công, tệp phiên người dùng sẽ được phân tích thành chuỗi truy cập web là lưu lại thành một tập dữ liệu (dataset). Mục đích chính của tiền xử lý là làm cho dữ liệu sạch đáp ứng cho nghiên cứu truy cập web.

Ở giai đoạn khai phá mẫu, bằng các kỹ thuật khai thác dữ liệu, các dữ liệu về truy cập web được trích xuất từ các trình tự truy cập web theo thứ tự ưu tiên (hành động được thực hiện nhiều nhất là một ví dụ).

Cuối cùng là giai đoạn phân tích mẫu, các dữ liệu thu thập được ở giai đoạn 2 sẽ được sử dụng để áp dụng vào các ứng dụng. Ví dụ: hệ thống đề xuất sản phẩm người dùng quan tâm, hệ thống đề xuất trang web trong đó các đề xuất sẽ được đưa ra dựa trên các dữ liệu đã tạo ra được ở trước đó.

Trong các giai đoạn nêu trên, khai thác quy tắc tuần tự được sử dụng tại giai đoạn khám phá mẫu. Đây là cách sử dụng chính của quá trình khai thác sử dụng web và thực hiện một công việc quan trọng trong đề xuất trang web/sản phẩm web trong đề xuất để hỗ trợ thông qua các lịch sử điều hướng của họ tại trang web đó.

* + 1. Tầm quan trọng của Web Usage Mining

Trong thời kỳ phát triển như hiện nay, việc áp dụng Web Usage Mining trở nên vô cùng quan trọng cho rất nhiều các ứng dụng nổi tiếng và được nhiều người dùng sử dụng hiện nay như Shoppe, Lazada, Tiki… trong hỗ trợ ra quyết định kinh doanh và tiếp thị, nghiên cứu khả năng truy cập web, phân tích lưu lượng mạng. Bên cạnh đó, phân tích lịch sử truy cập web rất có lợi khi biết được nhu cầu tìm kiếm của khách hàng để thu hút khách hàng mới, duy trì khách hàng thân thiết, cải thiện tiếp thị bán hàng, tăng cường hiệu quả các chiến dịch quảng cáo.

Tóm lại, Web Usage Mining thường được sử dụng trong các ứng dụng để mang lại lợi ích từ các kỹ thuật mô hình hóa người dùng như thiết kế website tự thích ứng và hệ thống khuyến nghị. Đây là cách tiếp cận được sử dụng nhiều nhất hiện nay.

* 1. Khái niệm cơ sơ dữ liệu Web Log
     1. Định nghĩa cơ sở dữ liệu Web Log

Tại các trang web, các máy chủ sẽ có một web log để lưu lại mỗi truy cập đơn lẻ khi sử dụng web. Các thông tin quan trọng trong WebLog bao gồm địa chỉ IP người truy cập, URL truy cập và thời gian truy cập.

Các tập tin WebLog được chia thành nhiều mục nhỏ với nhiều mục đích khai phá dữ liệu khác nhau. Để có được các phần này, kỹ thuật tiền xử lý được áp dụng để thực hiện việc này.

* + 1. Cấu trúc nội dung của Web Log

Cấu trúc và nội dung của một WebLog phụ thuộc vào máy chủ của nó. WebLog đầy đủ là một danh sách các thành phần nhật ký tuân theo chuẩn CLF (Common Log Format) hay còn được gọi là NCSA (định dạng nhật ký chung), là định dạng tệp văn bản được tiêu chuẩn hóa sử dụng bởi các máy chủ web khi tạo server log.

Các thông tin trong một tập CLF thường bao gồm như địa chỉ IP máy khách hay hostname, phương thức HTTP yêu cầu, đường dẫn của nguồn tài nguyên được truy cập trên máy chủ web (URL trang truy cập), thời gian truy cập, nghi thức được dùng (HTTP/1.0, HTTP/1.1), giao thức yêu cầu, mã trạng thái của yêu cầu, số lượng byte được truyền, referrer, user-agent, …Trong đó thuộc tính user-agent là phần mềm được dùng để truy cập các trang như là Web Crawler (GoogleBot, openbot, scooter, …) hay là trình duyệt (Google Chrome, Mozilla, Internet Explorer, Opera, …).

Ngoài ra còn có các định dạng khác như định dạng nhật ký mở rộng (W3C – World Wide Web Consortium) được hỗ trợ bởi các máy chủ Web là Apache và Netscape và định dạng W3SVC (World Wide Web Publising Service) được hỗ trợ bới IIS (Microsoft Internet Information Server).

Các thuộc tính của W3SVC bao gồm:

* date: ngày truy cập.
* time: thời gian truy cập.
* s-sitetime: tên dịch vụ internet và mã số đối tượng đang thực thi trên máy của user.
* s-computername: tên của máy chủ.
* s-ip: địa chỉ ip của máy chủ.
* cs-method: phương thức yêu cầu truy cập (GET hoặc POST).
* cs-uri-stem: liên kết của cs-method (stdportal.tdtu.edu.vn/Login/Index).
* cs-uri-query: truy vấn việc mà user đang cố gắng thực thi.
* s-sport: port của máy chủ.
* cs-username: tên của người dùng đã xác thực truy cập vào máy chủ, tên của các người dùng ẩn danh được xác định bằng một dấu “-”.
* c-ip: địa chỉ ip của máy khách hàng.
* cs-version: phiên bản giao thức http trên máy khách hàng.
* cs(user-agent): loại trình duyệt trên máy khách hàng.
* cs(cookie): nội dung của cookie.
* cs(referrer): liên kết người dùng truy cập lần cuối và cung cấp một liên kết đến trang hiện tại.
* sc-status: mã số giao thức http.
* cs-host: tên máy chủ.
* sc-substatus: mã lỗi của substatus.
* sc-win32-status: mã số tình trạng Windows.
* sc-bytes: số lượng byte được gửi bởi máy chủ.
* cs-bytes: số lượng byte được nhận và thực thi bởi máy chủ.
* time-taken: khoảng thời gian truy cập.

Ví dụ: một weblog đầy đủ được có cấu trúc như sau

Text

Description automatically generated

Hình 28: Một weblog đầy đủ

Table

Description automatically generated

Hình 29: Ý nghĩa các cột trong weblog.

Ví dụ: cấu trúc và nội dung weblog được thu thập trong một shop bán bong bóng bay được mô tả như sau.

Text, letter

Description automatically generated

Hình 30: Dữ liệu weblog của cửa hàng bán bóng bay.

123.30.175.221 - - [17/Nov/2014:04:44:33 +0700] "GET /thue-va-cho-thue/xi-c-cho-hai-dang/recommend HTTP/1.0" 200 28613 "-" "Mozilla/5.0 (compatible; coccoc/1.0; +http://help.coccoc.com/)"

Trong đó:

* + 123.30.175.221: IP máy khách hàng
  + - -: Remote user, Login user
  + 17/Nov/2014:04:44:33: date & time
  + +0700: time zone
  + GET: cs-method
  + /thue-va-cho-thue/xi-c-cho-hai-dang/recommend: cs-uri-stem
  + HTTP/1.0: nghi thức được dùng (protocol)
  + 200: sc-status
  + 28613: sc-bytes
  + Mozilla/5.0 (compatible; coccoc/1.0; +http://help.coccoc.com/)": cs(user agent)

CHƯƠNG 4 – ỨNG DỤNG

* 1. Giới thiệu

Khai thác tuần tự được áp dụng chủ yếu vào khai thác hành vi của người dùng. Thường là được sử dụng trong các ứng dụng về hành vi sử dụng web của khách hàng. Sử dụng luật tuần tự để có được mẫu phổ biến để đưa ra các gợi ý cho người dùng cũng như đưa ra chiến lược kinh doanh cho chủ doanh nghiệp.

* 1. Sản phẩm

3. 2. 1. Giới thiệu sản phẩm

Sản phẩm là một web kinh doanh về bong bóng. Áp dụng thuật toán bide để lấy ra các sản phẩm phổ biến và gợi ý sản phẩm cho người dùng. Đưa ra các thống kê về sản phẩm để đưa ra chiến lược kinh doanh cho doanh nghiệp.

* + 1. Chức năng
* Đăng ký, đăng nhập người dùng.
* Giỏ hàng.
* Mua sản phẩm.
* Gợi ý sản phẩm nổi bật, gợi ý sản phẩm cùng category cho người dùng.
* Trang quản lý của admin.
* Thêm, xóa, sửa sản phẩm.
* Thống kê doanh thu theo tháng, tuần và năm.
* Thống kê top lượt xem của các sản phẩm.
  + 1. Công nghệ sử dụng

Công nghệ sử dụng PHP và framework Laravel. Quản lý dữ liệu bằngMySQL và phpMyAdmin của Xampp.

* 1. Đánh giá ứng dụng
     1. Kết quả đạt được
* Áp dụng các kiến thức đã học xây dựng website Shop Bong Bóng, phân quyền người dùng và người quản trị, với giao diện và chức năng nghiệp vụ tương ứng.
* Quản lý các thông tin cần thiết của cửa hàng như các sản phẩm, thông tin khách hàng và các giao dịch mua hàng cửa khách hàng.
* Có hầu như đầy đủ chức năng cơ bản để quản lý cửa hàng.
  + 1. Hạn chế
* Giao diện chưa thật sự bắt mắt.
* Một số chức năng còn hạn chế.
* Việc áp dụng data mining vẫn chưa đạt được, sản phẩm chỉ là sử dụng của thuật toán ý tưởng để thực hiện.

CHƯƠNG 5 – KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

2. 1. Kết luận

Qua đồ án này, ta đã hiểu được thế nào là luật tuần từ và tìm hiểu được cơ sở lý thuyết về khai thác luật tuần tự trên cơ sở dữ liệu. Mục đích của đồ án là khái quát những khái niệm của luật tuần tự, khai thác dữ liệu, các thuật toán khai thác mẫu tuần tự ở nhiều cách tiếp cận khác nhau. Đồng thời, các ví dụ minh họa đã cho ta thấy hiểu được hơn cách tiếp cận và hoạt động của thuật toán. Ta cũng có thể hiểu và biết được thế nào là một weblog và nội dung trong đó bao gồm những gì và có ý nghĩa gì.

Đồ án đã giải quyết những vấn đề sau:

Chương một đã trình bày tổng quan về khai thác dữ liệu, cơ sở dữ liệu chuỗi, các dạng khai thác và thuật toán với từng dạng khai thác mẫu tuần tự đó.

Chương hai là các khái niệm về luật tuần tự, các bài toán khai thác luật tuần tự. Trong chương này, các thuật toán khai thác mẫu tuần tự phổ biến được trình bày với các ví dụ minh họa.

Chương ba trình bày về khai thác hành vi sử dụng web, định nghĩa weblog và định dạng, nội dung bên trong weblog.

Chương năm là kết luận những gì đạt được sau đồ án, những hạn chế và hướng phát triển trong tương lai.

* 1. Hạn chế

Đây là một đề tài rất hay trong lĩnh vực khai thác dữ liệu và khai phá tri thức (Datamining). Dưới vai trò của một sinh viên, được giao đề tài này giúp bọn em cải thiện được rất nhiều kể cả kiến thức lẫn khả năng đọc hiểu. Tuy nhiên, trong quá trình tìm hiểu, chắc chắn vẫn sẽ còn những sai sót và những nội dung chưa được chỉnh chu một cách tối đa nhất.

* 1. Hướng phát triển

Luật tuần tự là luật rất hữu dụng trong việc khám phá các tri thức tiềm ẩn trong các nguồn dữ liệu tuần tự. Tuy nhiên, do khối lượng thông tin bùng nổ như hiện nay nên số lượng dữ liệu ngày càng trở nên đồ sộ vì thế việc khai thác tập luật tuần tự làm sao để mang lại hiệu quả cao nhất và tốn ít thời gian nhất là vấn đề cần quan tâm hàng đầu. Vì vậy, trong bài báo cáo đã nêu lên hàng loạt các thuật toán khai thác mẫu tuần tự phổ biến với các khái niệm, điểm mạnh và yếu, ví dụ minh họa để khái quát hóa được thuật toán khai thác mẫu tuần tự.

Trong thực tế, vẫn còn nhiều lĩnh vực ứng dụng mà luật tuần tự có thể được sử dụng để khai thác và áp dụng.

# TÀI LIỆU THAM KHẢO

**Tiếng Việt**

1. Văn Thị Thiên Trang (2010), “Khai thác luật tuần tự trên cơ sở dữ liệu chuỗi”, Luận văn thạc sĩ ngành công nghệ thông tin, Đại học quốc gia thành phố Hồ Chí Minh, Trường đại học Khoa học tự nhiên.
2. Trần Minh Thái (2016), “Khai thác mẫu phổ biến và luật từ cơ sở dữ liệu chuỗi”, Luận án tiến sĩ công nghệ thông tin, Đại học học quốc gia thành phố Hồ Chí Minh, Trường đại học Khoa học tự nhiên.
3. Nguyễn Thôn Dã (2020), “Khai phá dữ liệu tuần tự để dự đoán hành vi truy cập web”, Luận án tiến sĩ kỹ thuật Học viện công nghệ bưu chính viễn thông, Hà Nội.
4. Phạm Thị Thiết (2020), “Một thuật toán hiệu quả cho bài toán khai thác mẫu tuần tự với ràng buộc trọng số”, Trường Đại học công nghệ thành phố Hồ Chí Minh.
5. Đỗ Thanh Tùng (2017), “Khai thác K mẫu tuần tự đóng”, Luận văn thạc sĩ công nghệ thông tin, Đại học công nghệ thành phố Hồ Chí Minh.
6. Quách Xuân Trưởng (2014), “Nghiên cứu các kỹ thuật khai phá mẫu dãy cho dữ liệu dãy”, Tạp chí khoa học và công nghệ.

**Tiếng Anh**

1. Jianyong Wang, Chun Li (2017), “Frequent Closed Sequence Mining without Candidate Maintenance”.
2. Surbhi Anand, Rinkle Rani Aggarwal (2012), “An Efficient Algorithm for Data Cleaning of Log File using File Extensions”.
3. Bay Vo, Tzung-Pei Hong, Bac Le (2011), “Dynamic bit vectors: An Efficient approach for mining frequent itemsets”.
4. Satyaveer Singh, Mahendra Singh Aswal (2020), “Performance Analysis of Sequential Rule Mining Techniques for Web Page Recommendation System”.
5. Jianyong Wang, Jiawei Han (2007), “BIDE: Efficient Mining of Frequent Closed Sequences”.
6. Jay Ayres, Johannes Gehrke, Tomi You, Jason Flannick (2002). “Sequential Pattern Mining using A Bitmap Representation”.
7. Mohamed J.Zaki (2001), “SPADE: An Efficient Algorithm for Mining Frequent Sequences”.
8. Jian Pei (2004), “Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach”.
9. Ramakrishnan Srikant, Rakesh Agrawal, “Generalization and Performance Improvements”.
10. Xifeng Yan, Jiawei Han, Ramin Afshar (2003), “CloSpan: Mining Closed Sequential Patterns in Large Dataset”.