Cao học CNTT qua mạng

Bài 2:
Luật kết hợp

PGS. TS. Đỗ Phúc
Khoa Hệ thống thông tin
Trường Đại học Công nghệ Thông tin, ĐHQG-HCM

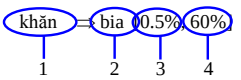
Luật kết hợp: Cơ sở

- **Khai phá luật kết hợp:**
  - Tìm tần số mẫu, mối kết hợp, sự tương quan, hay các cấu trúc nhân quả giữa các tập đối tượng trong các cơ sở dữ liệu giao tác, cơ sở dữ liệu quan hệ, và những kho thông tin khác.
- **Tính hiểu được:** dễ hiểu
- **Tính sử dụng được:** Cung cấp thông tin thiết thực
- **Tính hiệu quả:** Đã có những thuật toán khai thác hiệu quả
- **Các ứng dụng:**
  - Phân tích dữ liệu giỏ hàng, cross-marketing, thiết kế catalog, loss-leader analysis, gom cụm, phân lớp, ...

Luật kết hợp: Cơ sở

- **Định dạng thể hiện đặc trưng cho các luật kết hợp:**
  - khăn ⇒ bia [0.5%, 60%]
  - mua:khăn ⇒ mua:bia [0.5%, 60%]
  - "**Nếu** mua khăn **thì** mua bia trong 60% trường hợp. Khăn và bia được mua chung trong 0.5% dòng dữ liệu."
- **Các biểu diễn khác:**
  - mua(x, "khăn") ⇒ mua(x, "bia") [0.5%, 60%]
  - khoa(x, "CS") ^ học(x, "DB") ⇒ điểm(x, "A") [1%, 75%]

Luật kết hợp: Cơ sở



"**NẾU** mua khăn
THÌ mua bia
trong 60% trường hợp
trên 0.5% dòng dữ liệu"

1 **Tiền đề**, vế trái, thân
2 **Mệnh đề kết quả**, vế phải, đầu
3 **Support**, tần số ("trong bao nhiêu phần trăm dữ liệu thì những điều ở vế trái và vế phải cùng xảy ra")
4 **Confidence**, độ mạnh ("nếu vế trái xảy ra thì có bao nhiêu khả năng vế phải xảy ra")

Luật kết hợp: Cơ sở

- **Độ ủng hộ:** biểu thị tần số luật có trong các giao tác.

support(A ⇒ B [s, c]) = p(A∪B) = support ({A,B})

- **Độ tin cậy:** biểu thị số phần trăm giao tác có chứa luôn B trong số những giao tác có chứa A.

confidence(A ⇒ B [s, c]) = p(B|A) = p(A∪B) / p(A) = support({A,B}) / support({A})

Luật kết hợp: Cơ sở

- **Độ ủng hộ tối thiểu σ :**
  - **Cao** ⇒ **ít** tập phần tử (itemset) phổ biến
    ⇒ **ít** luật hợp lệ **rất thường** xuất hiện
  - **Thấp** ⇒ **nhiều** luật hợp lệ **hiếm** xuất hiện
- **Độ tin cậy tối thiểu γ :**
  - **Cao** ⇒ **ít** luật nhưng tất cả "**gần như đúng**"
  - **Thấp** ⇒ **nhiều** luật, phần lớn rất "**không chắc chắn**"
- **Giá trị tiêu biểu:** σ = 2 -10 %, γ = 70 - 90 %

Luật kết hợp: Cơ sở

- **Giao tác:**
  - **Dạng quan hệ** **Dạng kết**
    <Tid,item> **<Tid,itemset>**
    <1, item1> <1, {item1,item2}>
    <1, item2> <2, {item3}>
    <2, item3>
- **Item và itemsets:** phần tử đơn lẻ và tập phần tử
- **Support** của tập I: số lượng giao tác có chứa I
- **Min Support** σ: ngưỡng cho support
- **Tập phần tử phổ biến**: có độ ủng hộ (support) ≥ σ

Luật kết hợp: Cơ sở

- **Cho:** (1) CSDL các giao tác, (2) mỗi giao tác là một danh sách mặt hàng được mua (trong một lượt mua của khách hàng)

ID của giao tác	Hàng mua
100	A,B,C
200	A,C
400	A,D
500	B,E,F

Tập phổ biến	Độ phổ biến
{A}	3 or 75%
{B} and {C}	2 or 50%
{D}, {E} and {F}	1 or 25%
{A,C}	2 or 50%
Các cặp khác	max 25%

- **Tìm: tất cả** luật có support >= minsupport
- If min. support 50% and min. confidence 50%, then
  $A \Rightarrow C$ [50%, 66.6%], $C \Rightarrow A$ [50%, 100%]

Tạo luật kết hợp

- **Quá trình hai bước để khai thác luật kết hợp:**

BƯỚC 1: Tìm các *tập phổ biến*: các tập các phần tử có độ support tối thiểu.

- Mẹo Apriori: Tập con của tập phổ biến cũng là một tập phổ biến:
  - ví dụ, nếu {*AB*} là một tập phổ biến thì cả {*A*} và {*B*} đều là những tập phổ biến
- Lặp việc tìm tập phổ biến với kích thước từ 1 đến *k (tập có kích thước k)*

BƯỚC 2: Dùng các tập phổ biến để tạo các luật kết hợp.

Các tập phổ biến với mẹo Apriori

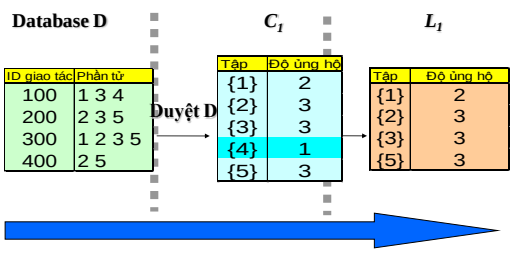
- **Bước kết hợp**: C_k được tạo bằng cách kết L_{k-1} với chính nó
- **Bước rút gọn**: Những tập kích thước (k-1) không phổ biến không thể là tập con của tập phổ biến kích thước k
- **Mã giả**:

C_k: Tập ứng viên có kích thước k; L_k : Tập phổ biến có kích thước k
L_1 = {các phần tử phổ biến};
for (k = 1; L_k !=∅; k++) **do begin**
  C_{k+1} = {các ứng viên được tạo từ L_k };
  for each giao tác t trong database do
    tăng số đếm của tất cả các ứng viên trong C_{k+1} mà được chứa trong t
  L_{k+1} = {các ứng viên trong C_{k+1} có độ ủng hộ tối tiểu}
  end
return $\cup_k L_k$;

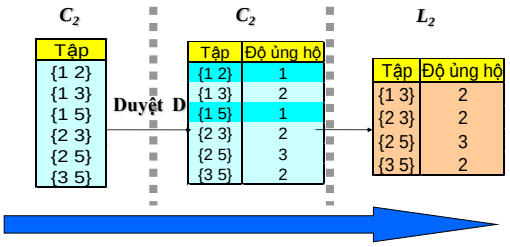
Tạo ứng viên Apriori

- **Nguyên tắc Apriori:**
  Những tập con của tập phổ biến cũng phải phổ biến
- L_3={*abc, abd, acd, ace, bcd*}
- **Tự kết:** L_3*L_3
  - *abcd* từ *abc* và *abd*
  - *acde* từ *acd* và *ace*
- **Rút gọn:**
  - *acde* bị loại vì *ade* không có trong L_3
- C_4={*abcd*}

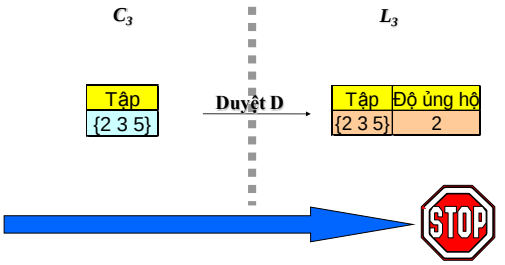
Ví dụ về Apriori (1/6)



Ví dụ về Apriori (2/6)

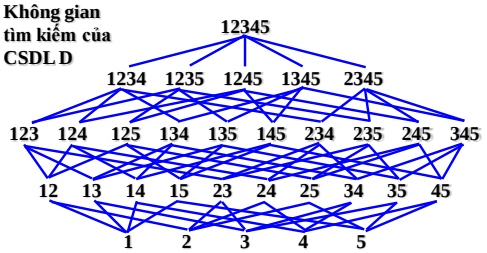


Ví dụ về Apriori (3/6)



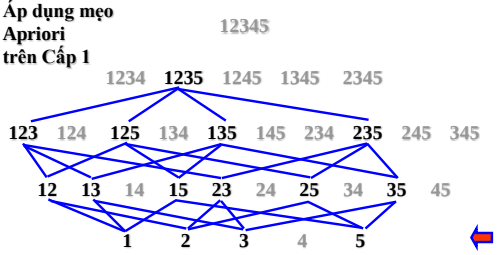
Ví dụ về Apriori (4/6)

Không gian tìm kiếm của CSDL D



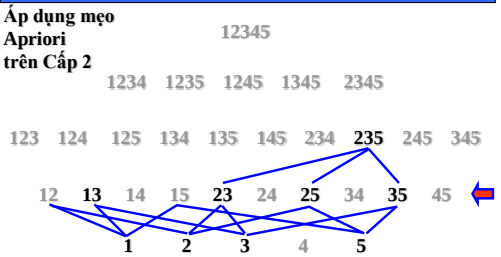
Ví dụ về Apriori (5/6)

Áp dụng mẹo Apriori trên Cấp 1



Ví dụ về Apriori (6/6)

Áp dụng mẹo Apriori trên Cấp 2



Thuật toán Apriori đã đủ nhanh?

- **Phần cốt lõi của thuật toán Apriori:**
  - Dùng các tập phổ biến kích thước $(k-1)$ để tạo các tập phổ biến kích thước k ứng viên
  - Duyệt database và đối sánh mẫu để đếm số lần xuất hiện của các tập ứng viên trong các giao tác
- **Tình trạng nghẽn cổ chai của thuật toán *Apriori*: việc tạo ứng viên**
  - Các tập ứng viên đồ sộ:
    - 10^4 tập phổ biến kích thước 1sẽ tạo ra 10^7 tập ứng viên kích thước 2
    - Để phát hiện một mẫu phổ biến kích thước 100, ví dụ $\{a_1, a_2, \ldots, a_{100}\}$, cần tạo $2^{100} \approx 10^{30}$ ứng viên.
  - Duyệt database nhiều lần:
    - Cần duyệt $(n+1)$ lần, n là chiều dài của mẫu dài nhất

Thuật toán Apriori đã đủ nhanh?

- **Thực tế:**
  - Đối với tiếp cận Apriori căn bản thì số lượng thuộc tính trên dòng thường khó hơn nhiều so với số lượng dòng giao tác.
  - Ví dụ:
    - 50 thuộc tính mỗi cái có 1-3 giá trị, 100.000 dòng (không quá tệ)
    - 50 thuộc tính mỗi cái có 10-100 giá trị, 100.000 dòng (hơi tệ)
    - 10.000 thuộc tính mỗi cái có 5-10 giá trị, 100 rows (quá tệ...)
  - Lưu ý:
    - Một thuộc tính có thể có một vài giá trị khác nhau
    - Các thuật toán luật kết hợp có đặc trưng là xem một cặp thuộc tính-giá trị là một thuộc tính (2 thuộc tính mỗi cái có 5 giá trị => "10 thuộc tính")
- **Có mấy cách để khắc phục vấn đề...**

Cải thiện hiệu quả của TT Apriori

- **Đếm tập dựa vào kỹ thuật băm**:
  - Một tập kích thước *k* có hashing bucket count tương ứng nhỏ hơn giới hạn thì không thể phổ biến.
- **Thu nhỏ giao tác**:
  - Một giao tác không chứa tập phổ biến kích thước *k* nào thì không cần xét đến ở các lần duyệt tiếp tiếp theo.
- **Chia nhỏ**:
  - Tập nào có khả năng phổ biến trong DB thì sẽ phổ biến trong ít nhất một phần chia của DB.
- **Lấy mẫu**:
  - Khai thác trên tập con của dữ liệu được cho, ngưỡng của độ ủng hộ thấp hơn + một phương thức để xác định tính đầy đủ

Rút các luật kết hợp từ các tập

- **Mã giả:**

for mỗi tập phổ biến *l*
tạo tất cả các tập con khác rỗng *s* of *l*
for mỗi tập con khác rỗng *s* of *l*
cho ra luật "*s* ⇒ (*l*-*s*)" nếu *support(l)/support(s)* ≥ *min_conf*", trong đó *min_conf* là ngưỡng độ tin cậy tối thiểu

- **Ví dụ: tập phổ biến l = {abc}, subsets s = {a, b, c, ab, ac, bc)**
  - a ⇒ b, a ⇒ c, b ⇒ c
  - a ⇒ bc, b ⇒ ac, c ⇒ ab
  - ab ⇒ c, ac ⇒ b, bc ⇒ a

Tạo luật kết hợp

- **Ghi nhớ 1:**
  - Việt tạo các tập phổ biến thì chậm (đặc biệt là các tập kích thước 2)
  - Việc tạo các luật kết hợp từ các tập phổ biến thì nhanh
- **Ghi nhớ 2:**
  - Khi tạo các tập phổ biết, ngưỡng độ ủng hộ được sử dụng
  - Khi tạo luật kết hợp, ngưỡng độ tin cậy được sử dụng
- **Thực tế, việc tạo các tập phổ biến và tạo các luật kết hợp thật sử chiếm thời gian bao lâu?**
  - Xét một ví dụ nhỏ trong thực tế…
  - Các thử nghiệm được thực hiện với Citum 4/275 Alpha server có bộ nhớ chính 512 MB & Red Hat Linux release 5.0 (kernel 2.0.30)

Chọn những luật tốt nhất?

- **Tập kết quả thường rất lớn, cần chọn ra những luật tốt nhất dựa trên:**
  - Các độ đo khách quan:
    Hai các đo phổ biến:
    - ☆ *support;* và
    - ☺ *confidence*
  - Các độ đo chủ quan (Silberschatz & Tuzhilin, KDD95)
    Một luật (mẫu) là tốt nếu
    - ☆ nó *bất ngờ* (gây ngạc nhiên cho user); và/hoặc
    - ☺ *có thể hoạt động* (user có thể dùng nó để làm gì đó)
- **Những kết quả này sẽ được dùng trong các quá trình khám phá tri thức (KDD)**

Luật Boolean và luật định lượng

- **Luật kết hợp Boolean so với định lượng (tùy vào loại giá trị được dùng)**
  - **Boolean:** Luật liên quan đến mối kết hợp giữa sự có xuất hiện và không xuất hiện của các phần tử (ví dụ "có mua A" hoặc "không có mua A")
    mua=SQLServer, mua=DMBook ⇒ mua=DBMiner [2%,60%]
    mua(x, "SQLServer") ^ mua(x, "DMBook") → mua(x, "DBMiner") [0.2%, 60%]
  - **Định lượng:** Luật liên quan đến mối kết hợp giữa các phần tử hay thuộc tính định lượng
    tuổi=30..39, thu nhập=42..48K ⇒ mua=PC [1%, 75%]
    tuổi(x, "30..39") ^ thu nhập(x, "42..48K") → mua(x, "PC") [1%, 75%]

Các luật định lượng

- **Các thuộc tính định lượng:** ví dụ: tuổi, thu nhập, chiều cao, cân nặng
- **Các thuộc tính phân loại:** ví dụ: màu sắc của xe

CID	chieu cao	can nang	thu nhap
1	168	75,4	30,5
2	175	80,0	20,3
3	174	70,3	25,8
4	170	65,2	27,0

Vấn đề: có quá nhiều giá trị khác nhau cho các thuộc tính định lượng

Giải pháp: chuyển các thuộc tính định lượng sang các thuộc tính phân loại (chuyển qua không gian rời rạc)

Các luật một chiều và nhiều chiều

- **Các mối kết hợp một chiều và nhiều chiều**
  - **Một chiều:** Các thuộc tính hoặc thuộc tính trong luật chỉ qui về một đại lượng (ví dụ, qui về "mua")
    Bia, khoai tây chiên ⇒ bánh mì [0.4%, 52%]
    mua(x, "Bia") ^ mua(x, "Khoai tây chiên") → mua(x, "Bánh mì") [0.4%, 52%]
  - **Nhiều chiều:** Các thuộc tính hoặc thuộc tính trong luật qui vềhai hay nhiều đại lượng (ví dụ: "mua", "thời gian giao dịch", "loại khách hàng")
    Trong ví dụ sau là: quốc gia, tuổi, thu nhập

Các luật nhiều chiều

CID	quoc gia	tuoi	thu nhap
1	Ý	50	thap
2	Pháp	40	cao
3	Pháp	30	cao
4	Ý	50	trung bình
5	Ý	45	cao
6	Pháp	35	cao

CÁC LUẬT:

quốc gia = Pháp ⇒ **thu nhập** = cao [50%, 100%]
thu nhập = cao ⇒ **quốc gia** = Pháp [50%, 75%]
tuổi = 50 ⇒ **quốc gia** = Ý [33%, 100%]

Các luật một cấp và nhiều cấp

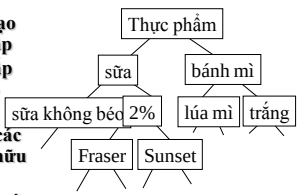
- **Các mối kết hợp một cấp và nhiều cấp**
  - **Một cấp:** Mối kết hợp giữa các phần tử hay thuộc tính của cùng một cấp khái niệm (ví dụ cùng một cấp của hệ thống phân cấp)
    Bia, Khoai tây chiên ⇒ Bánh mì [0.4%, 52%]
  - **Nhiều cấp:** Mối kết hợp giữa các phần tử hay thuộc tính của nhiều cấp khái niệm khác nhau (ví dụ nhiều cấp của hệ thống phân cấp)
    Bia:Karjala, Khoai tây chiên:Estrella:Barbeque ⇒ Bánh mì [0.1%, 74%]

Các luật kết hợp nhiều cấp

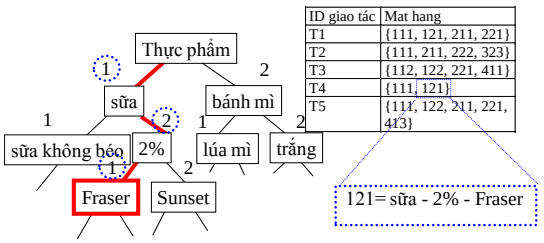
- **Khó tìm những mẫu tốt ở cấp quá gần gốc – at too primitive level**
  - độ ủng hộ cao = quá ít luật
  - độ ủng hộ thấp = quá nhiều luật, không tốt nhất
- **Tiếp cận: suy luận ở cấp khái niệm phù hợp**
- **Một dạng phổ biến của tri thức nền là một thuộc tính có thể được tổng quát hóa hay chi tiết hóa dựa vào cây khái niệm**
- **Các luật kết hợp nhiều cấp: những luật phối hợp các mối kết hợp với cây các khái niệm**

Các luật kết hợp nhiều cấp

- **Các phần tử thường tạo thành các cây phân cấp**
- **Các phần tử ở cấp thấp hơn được cho là có độ ủng hộ thấp hơn**
- **Các luật về các tập ở các cấp thích hợp sẽ khả hữu ích**
- **Cơ sở dữ liệu giao tác có thể được mã hóa dựa trên các chiều và các cấp**



Các luật kết hợp nhiều cấp



ID giao tác	Mat hang
T1	{111, 121, 211, 221}
T2	{111, 211, 222, 323}
T3	{112, 122, 221, 411}
T4	{111, 121}
T5	{111, 122, 211, 221, 413}

Các luật kết hợp nhiều cấp

- **Tiếp cận trên-xuống, tiến theo chiều sâu:**
  - Trước tiên tìm những luật mạnh ở cấp cao:
    sữa → bánh mì [20%, 60%]
  - Sau đó tìm những luật "yếu hơn" ở cấp thấp hơn của chúng:
    sữa 2% → bánh mì lúa mì [6%, 50%]
- **Khai thác thay đổi trên các luật kết hợp nhiều cấp:**
  - Các luật kết hợp trên nhiều cấp khác nhau:
    sữa → *bánh mì lúa mì*
  - Các luật kết hợp với nhiều cây khái niệm:
    sữa → *bánh mì Wonder*

Các luật kết hợp nhiều cấp

- **Tổng quát hóa/chuyên biệt hóa giá trị của các thuộc tính…**
  - **…từ chuyên biệt sang tổng quát:** support của các luật tăng (có thêm những luật mới hợp lệ)
  - **…từ tổng quát sang chuyên biệt:** support của các luật giảm (có những luật trở thành không hợp lệ, độ ủng hộ của chúng giảm xuống nhỏ hơn ngưỡng qui định)
- **Bậc quá thấp => quá nhiều luật và quá thô sơ**
  Pepsi light 0.5l bottle ⇒ Taffel Barbeque Chips 200gr
- **Bậc quá cao => các luật không hay**
  Food ⇒ Clothes

Lọc luật thừa

- **Có những luật có thể là dư thừa do đã có các mối quan hệ "tổ tiên" giữa các phần tử**
- **Ví dụ (sữa có 4 lớp con):**
  - sữa ⇒ bánh mì lúa mì [độ ủng hộ = 8%, độ tin cậy = 70%]
  - sữa 2% ⇒ bánh mì lúa mì [độ ủng hộ = 2%, độ tin cậy = 72%]
- **Ta nói luật thứ nhất là tổ tiên của luật thứ hai**
- **Một luật là dư thừa nếu độ ủng hộ của nó gần với giá trị "mong đợi", dựa trên tổ tiên của luật**
  - Luật thứa hai ở trên có thể là dư thừa

Khai thác dựa trên ràng buộc

- **Khai thác cả giga-byte dữ liệu theo cách thăm dò, có tương tác?**
  - Điều này có khả thi không? - Bằng cách sử dụng tốt các ràng buộc!
- **Các loại ràng buộc nào có thể dùng trong khai thác dữ liệu?**
  - **Ràng buộc dạng tri thức**: phân lớp, kết hợp, ….
  - **Ràng buộc dữ liệu**: những câu truy vấn dạng SQL
    - Tìm những cặp sản phẩm được bán chung tại VanCouver tháng 12/98
  - **Những ràng buộc về kích thước/cấp bậc**:
    - Có liên quan về vùng, giá, nhãn hiệu, loại khách hàng
  - **Những ràng buộc về sự hấp dẫn**:
    - Những luật mạnh (min_support ≥ 3%, min_confidence ≥ 60%)
  - **Những ràng buộc luật** (xem slide sau)

Ràng buộc luật

- **Có hai loại ràng buộc luật:**
  - **Ràng buộc dạng luật: khai thác theo siêu luật (meta-rule)**
    - Metarule: P(X, Y) ^ Q(X, W) → lấy(X, "database systems")
    - Luật đối sánh: tuổi(X, "30..39") ^ thu nhập(X, "41K..60K") → lấy(X, "database systems").
  - **Ràng buộc trên nội dung luật: tạo câu truy vấn dựa trên ràng buộc (Ng, et al., SIGMOD'98)**
    - sum(LHS) < 100 ^ min(LHS) > 20 ^ count(LHS) > 3 ^ sum(RHS) > 1000

Ràng buộc luật

- **Ràng buộc 1-biến và ràng buộc 2-biến (Lakshmanan, et al. SIGMOD'99):**
  - **1-biến:** Ràng buộc chỉ hạn chế trên một bên (L/R) của luật, ví dụ;
    - sum(LHS) < 100 ^ min(LHS) > 20 ^ count(LHS) > 3 ^ sum(RHS) > 1000
  - **2-biến:** Ràng buộc hạn chế trên cả hai bên (L và R) của luật.
    - sum(LHS) < min(RHS) ^ max(RHS) < 5* sum(LHS)

Tóm tắt

- **Khai thác luật kết hợp:**
  - Gần như là phần quan trọng nhất trong KDD
  - Khái niệm khá đơn giản nhưng ý tưởng của nó cung cấp cơ sở cho những mở rộng và những phương pháp khác
  - Nhiều bài báo đã được công bố về đề tài này
- **Đã có nhiều kết quả hấp dẫn**
- **Hướng nghiên cứu lý thú:**
  - Phân tích mối kết hợp trong các dạng dữ liệu khác: dữ liệu không gian, dữ liệu đa phương tiện, dữ liệu thời gian thực, …