Big Data là thuật ngữ dùng để chỉ một tập hợp dữ liệu rất lớn và rất phức tạp đến nỗi những công cụ, ứng dụng xử lí dữ liệu truyền thống không thể nào đảm đương được. Tuy nhiên, Big Data lại chứa trong mình rất nhiều thông tin quý giá mà nếu trích xuất thành công, nó sẽ giúp rất nhiều cho việc kinh doanh, nghiên cứu khoa học, dự đoán các dịch bệnh sắp phát sinh và thậm chí là cả việc xác định điều kiện giao thông theo thời gian thực. Chính vì thế, những dữ liệu này phải được thu thập, tổ chức, lưu trữ, tìm kiếm, chia sẻ theo một cách khác so với bình thường. Trong bài này, mời các bạn cùng tìm hiểu về Big Data, các phương thức người ta dùng để khai thác nó và nó giúp ích như thế nào cho cuộc sống của chúng ta.

**VOLUME (SỐ LƯỢNG LƯU TRỮ)**

Big Data (“dữ liệu lớn”) là tập hợp dữ liệu có dung lượng lưu trữ vượt mức đảm đương của những ứng dụng và công cụ truyền thống. [Kích cỡ của Big Data](http://dinhledat.com/experts/can-canh-adtech-viet-nam-2014-2016-cdo-vn-offline-1/) đang từng ngày tăng lên, và tính đến năm 2012 thì nó có thể nằm trong khoảng vài chục terabyte cho đến nhiều petabyte (1 petabyte = 1024 terabyte) chỉ cho một tập hợp dữ liệu mà thôi.

**VELOCITY (TỐC ĐỘ XỬ LÝ)**

Dung lượng gia tăng của dữ liệu rất nhanh và [tốc độ xử lý đang tiến tới real-time](http://dinhledat.com/data-driven-marketing/adtech-data-driven-marketing-va-su-thanh-cong-cua-chien-dich-tai-tranh-cu-barack-obama-nam-2012/). Các ứng dụng phổ biến trên lĩnh vực Internet, Tài chính, Ngân hàng, Hàng không, Quân sự, Y tế – Sức khỏe ngày hôm nay phần lớn dữ liệu lớn đc xử lý real-time. Công nghệ xử lý dữ liệu lớn ngày một tiên tiến cho phép chúng ta xử lý tức thì trước khi chúng được lưu trữ vào cơ sở dữ liệu.

**VARIETY (ĐA DẠNG CHỦNG LOẠI)**

Hình thức lưu trữ và chủng loại dữ liệu ngày một đa dạng hơn. Trước đây chúng ta hay nói đến dữ liệu có cấu trúc thì ngày nay hơn 80% dữ liệu trên thế giới được sinh ra là phi cấu trúc (tài liệu, blog, hình ảnh, video, voice v.v.). Công nghệ Big Data cho phép chúng ta ngày nay [liên kết và phân tích đa dạng chủng loại dữ liệu](http://dinhledat.com/data-driven-marketing/adtech-data-driven-marketing-va-su-thanh-cong-cua-chien-dich-tai-tranh-cu-barack-obama-nam-2012/)với nhau như comments/post của một nhóm người dùng nào đó trên Facebook với thông tin video được chia sẻ từ Youtube và Twitter.

**VERACITY (ĐỘ CHÍNH XÁC)**

Một trong những tính chất phức tạp nhất của BigData là độ chính xác của dữ liệu. Với xu hướng[Social Media và Social Network](http://dinhledat.com/data-driven-marketing/adtech-data-driven-marketing-va-su-thanh-cong-cua-chien-dich-tai-tranh-cu-barack-obama-nam-2012/) ngày nay và sự gia tăng mạnh mẽ tính tương tác và chia sẻ của người dùng Mobile làm cho bức tranh xác định về độ tin cậy & chính xác của dữ liệu ngày một khó khăn hơn. Bài toán phân tích và loại bỏ dữ liệu thiếu chính xác và nhiễu đang là tính chất quan trọng của BigData.

**VALUE (GIÁ TRỊ THÔNG TIN)**

Giá trị thông tin là tính chất quan trọng nhất của xu hướng công nghệ Big Data. Ở đây doanh nghiệp phải hoạch định được những giá trị thông tin hữu ích của BigData cho[vấn đề, bài toán hoặc mô hình hoạt động kinh doanh](http://dinhledat.com/data-driven-marketing/adtech-data-driven-marketing-va-su-thanh-cong-cua-chien-dich-tai-tranh-cu-barack-obama-nam-2012/http:/) của mình. Có thể nói việc đầu tiên là phải xác định được tính chất “Value” thì mới nên bắt tay vào BigData.

Việt Nam đang ngày càng gia tăng tốc độ phát triển và hội nhập với các xu hướng công nghệ thế giới. Với hơn 30 triệu người dùng Internet và hơn 15 triệu ngươi dùng Mobile Internet làm cho Việt Nam đang đứng trước một cơ hộ vô cùng lớn về khai thác dữ liệu lớn. Sẽ có những doanh nghiệp Việt Nam khai thác thành công dữ liệu lớn với doanh số hàng trăm triệu USD trong vòng 5 năm tới. Đặc biệt, giai đoạn 2014-2016, xu hướng Mobile và lượng người dùng Internet 3G sẽ tiếp tục tăng mạnh. Các dịch vụ kết nối OTT (Over-the-top) và truyền thông xã hội đóng góp hơn 80% phương thức giao tiếp online, video online và nội dung số mobile. Điều này góp phần đẩy mạnh xu hướng truyền thông số đa phương tiện, đa màn hình (PC, smartphone , tablet, smart TV) sẽ bùng nổ với độ phủ hơn 50% dân số Việt Nam. Việt Nam là một kho “vàng” dữ liệu vô cùng lớn cho việc ứng dụng Big Data.

-------------VAI TRÒ CỦA DATA MINNING

 Công nghệ hiện đại trong lĩnh vực quản lý thông tin 

Hiện diện khắp nơi (ubiquitous) và có tính ẩn (invisible) trong nhiều khía cạnh của đời sống hằng ngày

 Làm việc, mua sắm, tìm kiếm thông tin, nghỉ ngơi, … 

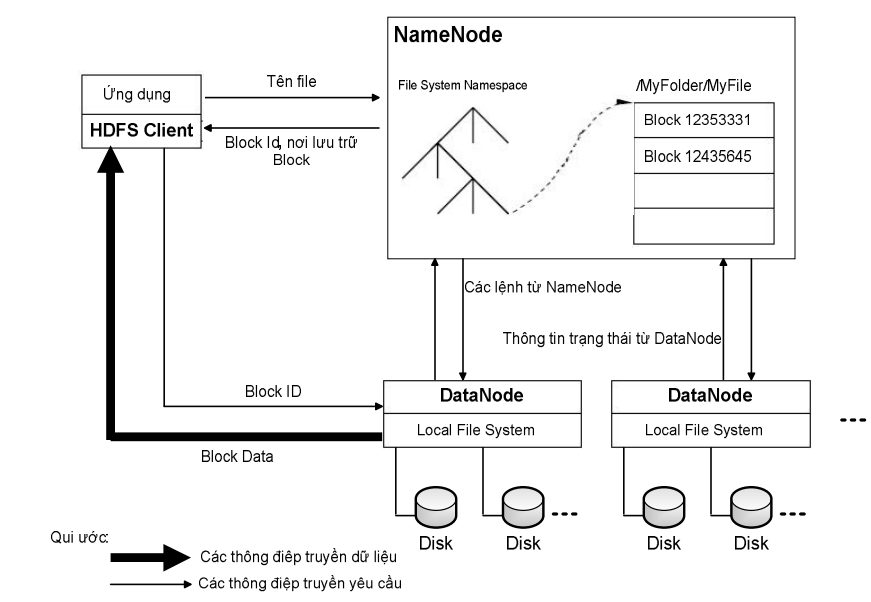
Được áp dụng trong nhiều ứng dụng thuộc nhiều lĩnh vực khác nhau

Hỗ trợ các nhà khoa học, giáo dục học, kinh tế học, doanh nghiệp, khách hàng, …

Khai phá dữ liệu là quá trình phân tích dữ liệu để nhận diện mối tương quan hay hình mẫu trong nhiều kiểu dữ liệu ĐÃ ĐƯỢC LƯU trong cơ sở dữ liệu rồi tóm tắt chúng thành thông tin hữu dụng. Chẳng hạn, người chủ nhìn vào kinh doanh của công ti mình; người đó có thể thấy thu nhập, chi phí và lợi nhuận. Nhưng với khai phá dữ liệu, người đó thấy nhiều hơn. Người đó biết trong hàng nghìn sản phẩm mà người đó bán, sản phẩm nào là bán chạy nhất. Người đó cũng biết khách hàng nào muốn mua, dựa trên hình mẫu của họ về mua sắm. Dựa trên báo cáo phân tích khai phá dữ liệu, người đó biết rằng nếu người đó giảm giá xuống 5% người đó có thể làm tăng số bán lên 45% và có 25% lợi nhuận thêm hơn so với trước. Về căn bản khai phá dữ liệu cho phép người chủ dùng thông tin hiện có để làm lộ ra xu hướng phụ thêm mà người đó có thể có ưu thế. Ngày nay Khai phá dữ liệu được dùng rộng rãi trong các công ti bán lẻ, tài chính, truyền thông và tiếp thị. Nó cho phép họ xác định mối tương quan giữa các yếu tố “nội bộ” như giá, sản phẩm, chi phí với các yếu tố “bên ngoài” như khách hàng, cạnh tranh và xu hướng kinh tế. Dựa trên thông tin phụ này, các công ti có thể xác định tác động lên số bán của họ, thói quen mua hàng của khách hàng, và lợi nhuận công ti. Với khai phá dữ liệu, người bán lẻ có thể dùng bản ghi số bán của việc mua của khách hàng để gửi quảng cáo khuyến mại dựa trên lịch sử mua bán cá nhân. Chẳng hạn, tôi bao giờ cũng mua sách tại Amazon.com cho nên hàng tuần công ti đều gửi cho tôi một danh sách các sách mới, phần lớn là sách máy tính để tôi mua. Họ không bao giờ gửi danh sách các sách lãng mạn, sách tài chính hay sách kiến trúc bởi vì họ biết rằng tôi thường mua sách kĩ thuật. Phần mềm khai phá dữ liệu của họ đã biết thói quen mua sách của tôi. Tuy nhiên, với Khai phá dữ liệu mọi dữ liệu phải CÓ CẤU TRÚC và ĐƯỢC XÁC ĐỊNH trước khi chúng có thể được lưu trong cơ sở dữ liệu. Các công cụ khai phá dữ liệu đặc biệt được dùng để thu thập những dữ liệu này từ cơ sở dữ liệu, phân tích chúng để nhận diện các hình mẫu và phát sinh báo cáo cho cấp quản lí. Nói cách khác, nếu dữ liệu được lưu trong cơ sở dữ liệu và được cấu trúc theo hàng và cột, bất kể kích cỡ của chúng lớn tới đâu, đều là miền của Khai phá dữ liệu. Ngày nay, có các kiểu dữ liệu khác KHÔNG ĐƯỢC XÁC ĐỊNH VÀ KHÔNG CÓ CẤU TRÚC và chúng ở rải rác khắp nơi. Chẳng hạn, dữ liệu từ Internet, từ hàng triệu website và mạng xã hội như ảnh Facebook, đồ thị thị trường chứng khoán, tin tức từ Twitter, dữ liệu cá nhân từ Linkedln, bản ghi sức khoẻ điện tử từ các bệnh viện, dữ liệu xu hướng kinh tế từ các viện nghiên cứu, dữ liệu thời tiết, dữ liện kinh doanh, emails, ảnh, video cá nhân, video từ YouTube, phim và nhạc tải xuống v.v. Những dữ liệu này KHÔNG THỂ được thu tập hay lưu giữ bởi các công cụ cơ sở dữ liệu điển hình. Hơn thế nữa, những dữ liệu này thay đổi hay tăng lên về kích thước một cách nhanh chóng. Chúng thêm mãi vào, hàng tỉ hàng tỉ, hàng nghìn tỉ hàng nghìn tỉ thứ xảy ra trong “thế giới ảo.” Những dữ liệu này cũng rất có giá trị để xác định hình mẫu hay xu hướng nữa. Khi bạn tổ hợp khối lượng dữ liệu lớn, sự đa dạng của các kiểu dữ liệu, và tốc độ chúng thay đổi thì bạn đang giải quyết với miền của Big Data. Big Data đã vượt quá các quan niệm cơ sở dữ liệu truyền thống. Qui mô lớn của chúng về các hình mẫu và xu hướng cũng khó được thấy. Mối quan hệ của chúng trong mọi kiểu dữ liệu khác nhau là quá phức tạp để được quan sát. VÀ chúng liên tục thay đổi với tốc độ của internet cho nên khó nhận diện dữ liệu này để lộ ra cái gì. Về căn bản, các quan niệm và công cụ của cơ sở dữ liệu và khai phá dữ liệu hiện thời KHÔNG có tác dụng nữa. Đó là lí do tại sao nó cần các quan niệm mới, công cụ mới, thuật toán mới và đó là lí do tại sao Bid Data là thứ mới ngày nay.

Câu 3

# Sơ đồ kiến trúc HDFS

[[](https://s3-ap-southeast-1.amazonaws.com/kipalog.com/jsk1lgwftc_hdfs.png)](https://s3-ap-southeast-1.amazonaws.com/kipalog.com/jsk1lgwftc_hdfs.png)

Namenode sẽ chịu trách nhiệm điều phối các thao tác truy cập (đọc/ghi dữ liệu) của client lên hệ thống HDFS. Và tất nhiên, do các DataNode là nơi thật sự lưu trữ các block của các file trên HDFS, nên chúng sẽ là nơi trực tiếp đáp ứng các thao tác truy cập này. Chẳng hạn như khi client của hệ thống muốn đọc một file trên hệ thống HDFS, client này sẽ thực hiện một request (thông qua RPC) đến Namenode để lấy các metadata của file cần đọc. Từ metadata này nó sẽ biết được danh sách các block của file và vị trí của các DataNode chứa các bản sao của từng block. Client sẽ truy cập vào các DataNode để thực hiện các request đọc các block.   
Namenode thực hiện nhiệm vụ của nó thông qua một daemon tên namenode chạy trên port 8021. Mỗi DataNode server sẽ chạy một daemon datanode trên port 8022. Định kỳ, mỗi DataNode sẽ báo cáo cho NameNode biết về danh sách tất cả các block mà nó đang lưu trữ, NameNode sẽ dựa vào những thông tin này để cập nhật lại các metadata trong nó. Cứ sau mỗi lần cập nhật lại như vậy, metadata trên NameNode sẽ đạt được tình trạng thống nhất với dữ liệu trên các DataNode. Toàn bộ trạng thái của metadata khi đang ở tình trạng thống nhất này được gọi là một checkpoint. Metadata ở trạng thái checkpoint sẽ được dùng để nhân bản metadata dùng cho mục đích phục hồi lại NameNode nếu NameNode bị lỗi.

## Đọc file trên HDFS

Đầu tiên, client sẽ mở file cần đọc bằng cách gửi yêu cầu đọc file đến NameNode. Sau đó NameNode sẽ thực hiện một số kiểm tra xem file được yêu cầu đọc có tồn tại không, hoặc file cần đọc có đang ở trạng thái “khoẻ mạnh” hay không. Nếu mọi thứ đều ổn, NameNode sẽ gửi danh sách các block (đại diện bởi Block ID) của file cùng với địa chỉ các DataNode chứa các bản sao của block này.   
Tiếp theo, client sẽ mở các kết nối tới DataNode, thực hiện một RPC để yêu cầu nhận block cần đọc và đóng kết nối với DataNode. Lưu ý là với mỗi block ta có thể có nhiều DataNode lưu trữ các bản sao của block đó. Client sẽ chỉ đọc bản sao của block từ DataNode “gần” nhất.   
Client sẽ thực hiện việc đọc các block lặp đi lăp lại cho đến khi block cuối cùng của file được đọc xong. Quá trình client đọc dữ liệu từ HDFS sẽ transparent với người dùng hoặc chương trình ứng dụng client, người dùng sẽ dùng một tập API của Hadoop để tương tác với HDFS, các API này che giấu đi quá trình liên lạc với NameNode và kết nối các DataNode để nhận dữ liệu.

## Ghi file trên HDFS

Đầu tiên, client sẽ gửi yêu cầu đến NameNode tạo một file entry lên File System Namespace. File mới được tạo sẽ rỗng, tức chưa có một block nào. Sau đó, NameNode sẽ quyết định danh sách các DataNode sẽ chứa các bản sao của file cần gì và gửi lại cho client Client sẽ chia file cần gì ra thành các block, và với mỗi block client sẽ đóng gói thành một packet. Lưu ý là mỗi block sẽ được lưu ra thành nhiều bản sao trên các DataNode khác nhau (tuỳ vào chỉ số độ nhân bản của file).  
Client gửi packet cho DataNode thứ nhất, DataNode thứ nhất sau khi nhận được packet sẽ tiến hành lưu lại bản sao thứ nhất của block. Tiếp theo DataNode thứ nhất sẽ gửi packet này cho DataNode thứ hai để lưu ra bản sao thứ hai của block. Tương tự   
DataNode thứ hai sẽ gửi packet cho DataNode thứ ba. Cứ như vậy, các DataNode cũng lưu các bản sao của một block sẽ hình thành một ống dẫn dữ liệu data pipe.   
Sau khi DataNode cuối cùng nhận thành được packet, nó sẽ gửi lại cho DataNode thứ hai một gói xác nhận rằng đã lưu thành công . Và gói thứ hai lại gửi gói xác nhận tình trạng thành công của hai DataNode về DataNode thứ nhất.  
Client sẽ nhận được các báo cáo xác nhận từ DataNode thứ nhất cho tình trạng thành công của tất cả DataNode trên data pipe.   
Nếu có bất kỳ một DataNode nào bị lỗi trong quá trình ghi dữ liệu, client sẽ tiến hành xác nhận lại các DataNode đã lưu thành công bản sao của block và thực hiện một hành vi ghi lại block lên trên DataNode bị lỗi.   
Sau khi tất cả các block của file đều đã đươc ghi lên các DataNode, client sẽ thực hiên một thông điệp báo cho NameNode nhằm cập nhật lại danh sách các block của file vừa tạo. Thông tin Mapping từ Block ID sang danh sách các DataNode lưu trữ sẽ được NameNode tự động cập nhật bằng các định kỳ các DataNode sẽ gửi báo cáo cho NameNode danh sách các block mà nó quản lý.

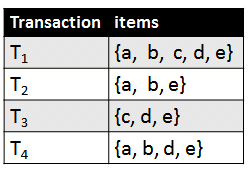
## BÀI TẬP

ví dụ tìm high utitility

**Khai thác thường xuyên**

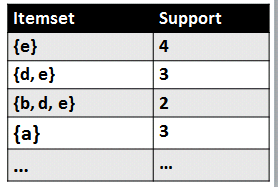
Vấn đề **khai thác itemset tiện ích cao** là một phần mở rộng của vấn đề  [**khai thác mẫu thường xuyên**](http://data-mining.philippe-fournier-viger.com/introduction-frequent-pattern-mining/) . Khai thác mẫu thường xuyên là một vấn đề phổ biến trong khai thác dữ liệu, bao gồm việc tìm kiếm các mẫu thường xuyên trong cơ sở dữ liệu giao dịch. Hãy để tôi mô tả đầu tiên vấn đề khai thác các vật phẩm thường xuyên

Hãy xem xét cơ sở dữ liệu sau. Nó là một **cơ sở dữ liệu giao dịch** . Một **cơ sở dữ liệu giao dịch** là một cơ sở dữ liệu có chứa một tập hợp các giao dịch được thực hiện bởi khách hàng. Một **giao dịch** là một tập hợp của những thứ được mua bởi khách hàng. Ví dụ, trong cơ sở dữ liệu sau, khách hàng đầu tiên mua các mặt hàng “a”, “b”, “c”, “d” và “e”, trong khi người thứ hai mua các mặt hàng “a”, “b” và “e” .

[](http://data-mining.philippe-fournier-viger.com/wp-content/uploads/2015/04/transactiondb.png)

Cơ sở dữ liệu giao dịch

Mục tiêu của việc khai thác vật phẩm thường xuyên là tìm các vật phẩm **thường xuyên** . Nhiều thuật toán phổ biến đã được đề xuất cho vấn đề này như [Apriori](http://data-mining.philippe-fournier-viger.com/classic-data-mining-algorithm-1-apriori/) , FPGrowth, LCM, Eclat, vv Các thuật toán này lấy làm đầu vào cơ sở dữ liệu giao dịch và tham số “ **minsup** ” được gọi là  **ngưỡng hỗ trợ tối thiểu.**Các thuật toán này sau đó trả về tất cả các mục ( **itemsets** ) xuất hiện trong các giao dịch ít nhất là  **vài phút** . Ví dụ, nếu chúng ta thiết lập  **minsup = 2,** trong ví dụ của chúng ta, chúng ta sẽ tìm thấy một vài bộ mặt hàng như vậy (được gọi là các tập thường xuyên) như sau:

[](http://data-mining.philippe-fournier-viger.com/wp-content/uploads/2015/04/frequent_itemsets.png)

một số tập phổ biến thường xuyên

Ví dụ, hãy xem xét các itemet {b, d, e}. Nó được cho là có sự hỗ trợ của 3 vì nó xuất hiện trong ba giao dịch, và nó được cho là thường xuyên bởi vì sự hỗ trợ của {b, d, e} không nhỏ hơn 1 **phút** .

**Khai thác các mặt hàng thường xuyên có một số hạn chế quan trọng**

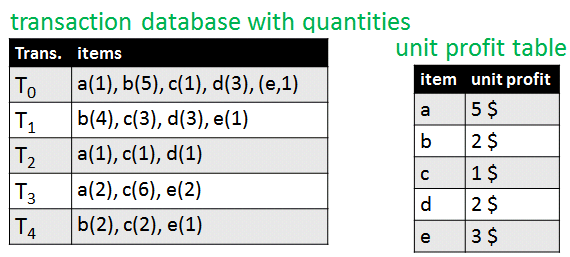
Vấn đề thường xuyên khai thác itemset là phổ biến. Nhưng nó có một số hạn chế quan trọng khi nói đến phân tích các giao dịch của khách hàng. Một hạn chế quan trọng là số lượng mua hàng không được tính đến. Do đó, một mục chỉ có thể xuất hiện một lần hoặc không trong một giao dịch. Vì vậy, nếu một khách hàng đã mua năm bánh mì, mười bánh mì hoặc hai mươi bánh mì, nó được xem là như nhau.

Một hạn chế quan trọng thứ hai là tất cả các mục được xem là có cùng tầm quan trọng, tiện ích của trọng lượng. Ví dụ, nếu một khách hàng mua một chai rượu rất đắt tiền hoặc một miếng bánh mì giá rẻ, nó được coi là quan trọng không kém.

Do đó, việc khai thác mẫu thường xuyên có thể tìm thấy nhiều mẫu thường xuyên không thú vị. Ví dụ, người ta có thể thấy rằng {bánh mì, sữa} là một mô hình thường xuyên. Tuy nhiên, từ quan điểm kinh doanh, mô hình này có thể không thú vị bởi vì nó không tạo ra nhiều lợi nhuận. Hơn nữa, các thuật toán khai thác mẫu thường xuyên có thể bỏ lỡ các mẫu hiếm có tạo ra lợi nhuận cao như có lẽ {trứng cá muối, rượu vang}.

**Khai thác các mặt hàng tiện ích cao**

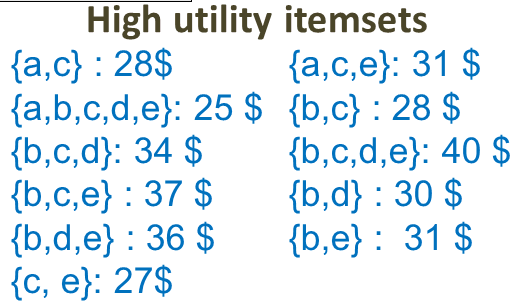
Để giải quyết những hạn chế này, vấn đề khai thác các mặt hàng thường xuyên đã được định nghĩa lại là vấn đề  **khai thác các mặt hàng tiện ích cao** . Trong bài toán này, một cơ sở dữ liệu giao dịch chứa các giao dịch trong đó số lượng mua hàng được tính đến cũng như lợi nhuận của từng mặt hàng. Ví dụ, hãy xem xét cơ sở dữ liệu giao dịch sau đây.

[](http://data-mining.philippe-fournier-viger.com/wp-content/uploads/2015/04/transaction_db_quantities.png)

cơ sở dữ liệu giao dịch với số lượng và thông tin lợi nhuận đơn vị cho các mặt hàng

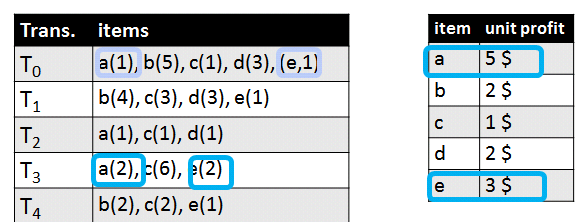
Xem xét giao dịch T3. Nó chỉ ra rằng khách hàng tương ứng đã mua hai đơn vị của mặt hàng “a”, sáu đơn vị của mặt hàng “c” và hai đơn vị của mặt hàng “e”. Bây giờ nhìn vào bảng bên phải. Bảng này cho biết lợi nhuận đơn vị của mỗi mục. Ví dụ, lợi nhuận đơn vị của các mục “a”, “b”, “c”, “d” và “e” tương ứng là 5 $, 2 $, 1 $, 2 $ và 3 $. Điều này có nghĩa là, ví dụ, mỗi đơn vị của "a" được bán ra tạo ra lợi nhuận 5 đô la.

Vấn đề **khai thác** các mặt hàng **tiện ích cao** là tìm các tập hợp (nhóm các mặt hàng) tạo ra lợi nhuận cao trong cơ sở dữ liệu, khi chúng được bán cùng nhau. Người dùng phải cung cấp một giá trị cho một ngưỡng gọi là " **minutil** " (ngưỡng tiện ích tối thiểu). Một **ích cao khai thác tập phổ biến thuật toán**  đầu ra tất cả các  **tập phổ biến ích cao** , có nghĩa là các tập phổ biến mà tạo ra ít nhất “ **minutil** ” lợi nhuận. Ví dụ, hãy xem xét rằng " **minutil** " được đặt thành 25 $ bởi người dùng. Kết quả của một **thuật toán khai thác itemset tiện ích cao** sẽ là như sau.

[](http://data-mining.philippe-fournier-viger.com/wp-content/uploads/2015/04/huis.png)

các tiện ích cao cấp

Ví dụ, hãy xem xét các itemet {b, d}. Nó được coi là một **itemset tiện ích cao** , bởi vì nó có một tiện ích 40 $ (tạo ra một lợi nhuận là 40 $), đó là không ít hơn ngưỡng **minutil** đã được thiết lập đến 25 $ bởi người dùng. Bây giờ, chúng ta hãy xem xét chi tiết hơn về cách tiện ích (lợi nhuận) của một itemset được tính toán. Nói chung, **các tiện ích của một itemset trong một giao dịch** là số lượng của mỗi mục từ các itemset nhân với lợi nhuận đơn vị của họ. Ví dụ, hãy xem xét hình bên dưới. Lợi nhuận của {a, e} trong giao dịch T0 là 1 x 5 + 1 x 3 = 8 $. Tương tự, lợi nhuận của {a, e} trong giao dịch T3 là 2 x 5 + 2 x 3 = 16 $. Bây giờ, **tiện ích của một itemset trong toàn bộ cơ sở dữ liệu** là tổng của tiện ích của nó trong tất cả các giao dịch mà nó xuất hiện. Như vậy, với {a, e}, tiện ích của nó là tổng của 8 $ + 16 $ = 24 $ vì nó chỉ xuất hiện trong các giao dịch T0 và giao dịch T3.

[](http://data-mining.philippe-fournier-viger.com/wp-content/uploads/2015/04/itemset_ae.png)

minh họa cách tính toán tiện ích của itemset {a, e}

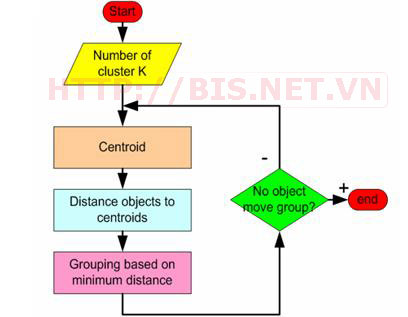
////////////////////////////////////////////////

K- MEAN

**2. Thuật Toán K-Means**

K-Means là thuật toán rất quan trọng và được sử dụng phổ biến trong kỹ thuật phân cụm. Tư tưởng chính của thuật toán K-Means là tìm cách phân nhóm các đối tượng (objects) đã cho vào K cụm (K là số các cụm được xác đinh trước, K nguyên dương) sao cho tổng bình phương khoảng cách giữa các đối tượng đến tâm nhóm (centroid ) là nhỏ nhất.

***Thuật toán K-Means được mô tả như sau***



***Thuật toán K-Means thực hiện qua các bước chính sau:***

1.    Chọn ngẫu nhiên K tâm (centroid) cho K cụm (cluster). Mỗi cụm được đại diện bằng các tâm của cụm.

2.    Tính khoảng cách giữa các đối tượng (objects) đến K tâm (thường dùng khoảng cách Euclidean)

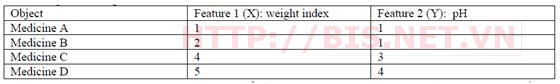
3.    Nhóm các đối tượng vào nhóm gần nhất

4.    Xác định lại tâm mới cho các nhóm

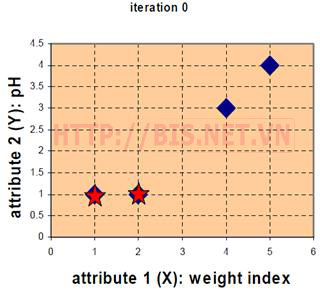
5.    Thực hiện lại bước 2 cho đến khi không có sự thay đổi nhóm nào của các đối tượng

***Ví dụ minh họa thuật toán K-Mean:***

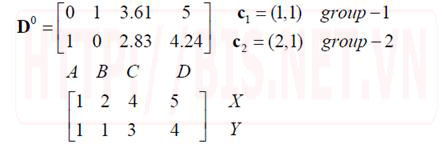
Giả sử ta có 4 loại thuốc A,B,C,D, mỗi loại thuộc được biểu diễn bởi 2 đặc trưng X và Y như sau. Mục đích của ta là nhóm các thuốc đã cho vào 2 nhóm (K=2) dựa vào các đặc trưng của chúng.



**Bước 1.** Khởi tạo tâm (centroid) cho 2 nhóm. Giả sử ta chọn A là tâm của nhóm thứ nhất (tọa độ tâm nhóm thứ nhất c1(1,1)) và B là tâm của nhóm thứ 2 (tạo độ tâm nhóm thứ hai c2 (2,1)).



**Bước 2.** Tính khoảng cách từ các đối tượng đến tâm của các nhóm (Khoảng cách Euclidean)



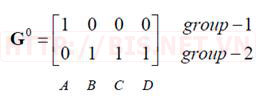
Mỗi cột trong ma trận khoảng cách (D) là một đối tượng (cột thứ nhất tương ứng với đối tượng A, cột thứ 2 tương ứng với đối tượng B,…). Hàng thứ nhất trong ma trận khoảng cách biểu diễn khoảng cách giữa các đối tượng đến tâm của nhóm thứ nhất (c1) và hàng thứ 2 trong ma trận khoảng cách biểu diễn khoảng cách của các đối tượng đến tâm của nhóm thứ 2 (c2).

Ví dụ, khoảng cách từ loại thuốc C=(4,3) đến tâm c1(1,1) là 3.61  và đến tâm c2(2,1) là 2.83 được tính như sau:

http://bis.net.vn/photos/storage/ClusteringTechnique006.jpg

http://bis.net.vn/photos/storage/ClusteringTechnique007.jpg

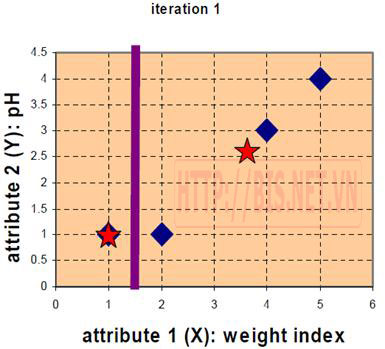
**Bước 3.** Nhóm các đối tượng vào nhóm gần nhất



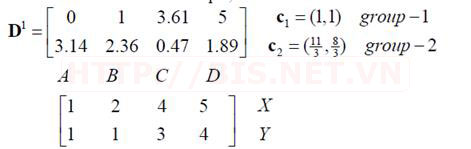
Ta thấy rằng  nhóm 1 sau vòng lặp thứ nhất gồm có 1 đối tượng A và nhóm 2 gồm các đối tượng còn lại B,C,D.

**Bước 5.** Tính lại tọa độ các tâm cho các nhóm mới dựa vào tọa độ của các đối tượng trong nhóm. Nhóm 1 chỉ có 1 đối tượng A nên tâm nhóm 1 vẫn không đổi, c1(1,1). Tâm nhóm 2 được tính như sau:

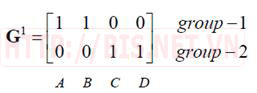
http://bis.net.vn/photos/storage/ClusteringTechnique009.jpg



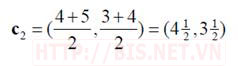
**Bước 6.** Tính lại khoảng cách từ các đối tượng đến tâm mới

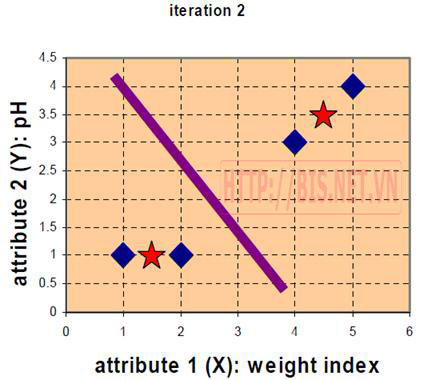


**Bước 7.** Nhóm các đối tượng vào nhóm

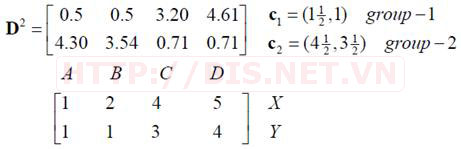


Bước 8. Tính lại tâm cho nhóm mới

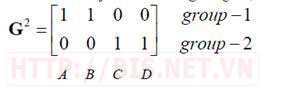
http://bis.net.vn/photos/storage/ClusteringTechnique013.jpg



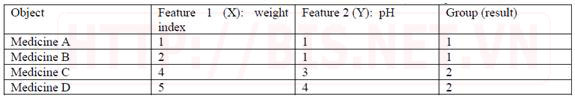
**Bước 8.** Tính lại khoảng cách từ các đối tượng đến tâm mới



**Bước 9.** Nhóm các đối tượng vào nhóm



Ta thấy G2 = G1 (Không có sự thay đổi nhóm nào của các đối tượng) nên thuật toán dừng và kết quả phân nhóm như sau:



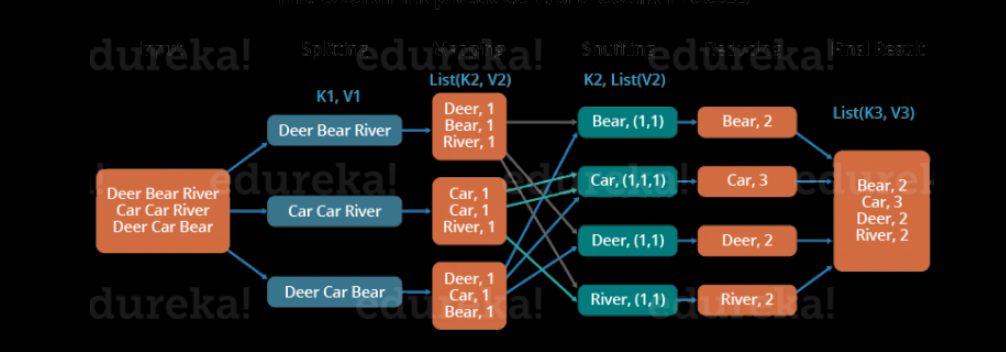
 Thuật toán K-Means có ưu điểm là đơn giản, dễ hiểu và cài đặt. Tuy nhiên, một số hạn chế của K-Means là hiệu quả của thuật toán phụ thuộc vào việc chọn số nhóm K (phải xác định trước) và chi phí cho thực hiện vòng lặp tính toán khoảng cách lớn khi số cụm K và dữ liệu phân cụm lớn.

**3. Triển khai ứng dụng phân cụm với phần mềm WeKa**

Trong ví dụ này, tôi sẽ giới thiệu cách xây dựng một KnowledgeFlow để triển khai kỹ thuật phân cụm dựa trên thuật toán K-Means trên Data Mining Software WeKa.

Dữ liệu dùng để phân cụm trong ví dụ này là dữ liệu dùng để phân loại khách hàng của ngân hàng (file dữ liệu **bank.arff**). bank.arff gồm có 11 thuộc tính và 600 khách hàng (instances). Dưới đây là cấu trúc và phân bố dữ liệu của bank.arff

VÍ DỤ MAPP



////////////////////////////////////////////////////////////////////////////////////

**Ví dụ này giải thích cách chạy thuật toán Apriori bằng cách sử dụng**[**thư viện khai phá dữ liệu nguồn mở SPMF**](http://www.philippe-fournier-viger.com/spmf/)**.**

Làm thế nào để chạy ví dụ này?

* **Nếu bạn đang sử dụng giao diện đồ họa,** (1) chọn **" Apriori "** thuật toán **,** (2) chọn tập tin đầu vào **" contextPasquier99.txt"** , (3) thiết lập tên tập tin đầu ra (ví dụ: " **output.txt** ") (4) đặt minsup lên **40%** và (5) nhấp vào " **Chạy thuật toán** ".
* **Nếu bạn muốn thực hiện ví dụ này từ dòng lệnh** , sau đó thực hiện lệnh này:   
  **java -jar spmf.jar chạy Apriori contextPasquier99.txt output.txt 40%** trong một thư mục chứa **tệp spmf.jar** và tệp đầu vào ví dụ **contextPasquier99.txt** .
* **Nếu bạn đang sử dụng phiên bản mã nguồn của SPMF, hãy** khởi chạy tệp **"MainTestApriori\_saveToMemory.java"** trong gói **ca.pfv.SPMF.tests** .

Apriori là gì?

**Apriori** là một thuật toán để khám phá các tập hợp thường xuyên trong cơ sở dữ liệu giao dịch. Nó được đề xuất bởi Agrawal & Srikant (1993).

Đầu vào của thuật toán Apriori là gì?

Dữ liệu đầu vào là một **cơ sở dữ liệu giao dịch** (còn gọi là ngữ cảnh nhị phân) và một ngưỡng có tên là ***minsup*** (giá trị từ 0 đến 100%).

Một **cơ sở dữ liệu giao dịch** là một tập hợp các giao dịch. Mỗi **giao dịch** là một tập hợp các mục. Ví dụ, hãy xem xét cơ sở dữ liệu giao dịch sau đây. Nó chứa 5 giao dịch (t1, t2, ..., t5) và 5 mục (1,2, 3, 4, 5). Ví dụ, giao dịch đầu tiên đại diện cho tập hợp các mục 1, 3 và 4. Cơ sở dữ liệu này được cung cấp dưới dạng tệp **contextPasquier99.txt** trong bản phân phối SPMF. Điều quan trọng cần lưu ý là một mục không được phép xuất hiện hai lần trong cùng một giao dịch và các mục đó được giả định là được sắp xếp theo thứ tự từ điển trong một giao dịch.

|  |  |
| --- | --- |
| **ID giao dịch** | **Mặt hàng** |
| **t1** | {1, 3, 4} |
| **t2** | {2, 3, 5} |
| **t3** | {1, 2, 3, 5} |
| **t4** | {2, 5} |
| **t5** | {1, 2, 3, 5} |

Đầu ra của thuật toán Apriori là gì?

Apriori là một thuật toán để khám phá các tập hợp (nhóm các mục) xảy ra thường xuyên trong một cơ sở dữ liệu giao dịch ( **các tập thường xuyên** ). Một mặt hàng thường xuyên là một itemetxuất hiện trong các giao dịch ít nhất là phút từ cơ sở dữ liệu giao dịch, trong đó minsup là một tham số do người dùng đưa ra.

Ví dụ, nếu **Apriori** được chạy trên cơ sở dữ liệu giao dịch trước đó với một phút lên đến 40% (2 giao dịch), Apriori tạo ra kết quả sau:

|  |  |
| --- | --- |
| **bộ sưu tập** | **ủng hộ** |
| {1} | 3 |
| {2} | 4 |
| {3} | 4 |
| {5} | 4 |
| {1, 2} | 2 |
| {1, 3} | 3 |
| {1, 5} | 2 |
| {2, 3} | 3 |
| {2, 5} | 4 |
| {3, 5} | 3 |
| {1, 2, 3} | 2 |
| {1, 2, 5} | 2 |
| {1, 3, 5} | 2 |
| {2, 3, 5} | 3 |
| {1, 2, 3, 5} | 2 |

Tôi nên giải thích kết quả như thế nào?

Trong kết quả, mỗi mục được chú thích với sự hỗ trợ của nó. Sự **hỗ trợ** của một itemset là bao nhiêu lần itemet xuất hiện trong cơ sở dữ liệu giao dịch. Ví dụ, các itemet {2, 3 5} có hỗ trợ 3 bởi vì nó xuất hiện trong các giao dịch t2, t3 và t5. Nó là một tập hợp thường xuyên bởi vì sự hỗ trợ của nó cao hơn hoặc bằng tham số minsup .

Định dạng tệp đầu vào

Các **định dạng tập tin đầu vào** cho Apriori được định nghĩa như sau. Nó là một tập tin văn bản. Một mục được biểu thị bằng một số nguyên dương. Giao dịch là một dòng trong tệp văn bản. Trong mỗi dòng (giao dịch), các mục được phân cách bằng một dấu cách. Giả sử rằng tất cả các mục trong cùng một giao dịch (dòng) được sắp xếp theo tổng số thứ tự (ví dụ thứ tự tăng dần) và không có mục nào có thể xuất hiện hai lần trong cùng một dòng.

Ví dụ, đối với ví dụ trước, tệp đầu vào được định nghĩa như sau:

1 3 4   
2 3 5   
1 2 3 5   
2 5   
1 2 3 5

Lưu ý rằng cũng có thể sử dụng **định dạng ARFF** thay thế cho định dạng đầu vào mặc định. Có thể tìm thấy đặc tả của định dạng ARFF [tại đây](http://weka.wikispaces.com/ARFF+%28stable+version%29) . Hầu hết các tính năng của định dạng ARFF được hỗ trợ ngoại trừ (1) ký tự "=" bị cấm và (2) ký tự thoát không được xem xét. Lưu ý rằng khi định dạng ARFF được sử dụng, hiệu suất của thuật toán khai phá dữ liệu sẽ hơi nhỏ hơn nếu định dạng tệp SPMF gốc được sử dụng vì chuyển đổi tệp đầu vào sẽ được tự động thực hiện trước khi khởi chạy thuật toán và kết quả cũng sẽ có được chuyển đổi. Chi phí này tuy nhiên nên nhỏ.

Định dạng tệp đầu ra

Các **định dạng tập tin đầu ra** được định nghĩa như sau. Nó là một tập tin văn bản, trong đó mỗi dòng đại diện cho một tập hợp thường xuyên. Trên mỗi dòng, các mục của itemet được liệt kê đầu tiên. Mỗi mục được biểu thị bằng một số nguyên và nó được theo sau bởi một dấu cách. Sau đó, tất cả các mục, từ khóa "#SUP:" xuất hiện, được theo sau bởi một số nguyên cho biết sự hỗ trợ của các itemet, được biểu thị dưới dạng một số giao dịch. Ví dụ, đây là tệp đầu ra cho ví dụ này. Dòng đầu tiên cho biết các bộ sưu tập thường xuyên bao gồm mục 1 và nó chỉ ra rằng bộ sưu tập này có sự hỗ trợ của 3 giao dịch.

1 #SUP: 3   
2 #SUP: 4   
3 #SUP: 4   
5 #SUP: 4   
1 2 #SUP: 2   
1 3 #SUP: 3   
1 5 #SUP: 2   
2 3 #SUP: 3   
2 5 #SUP: 4   
3 5 #SUP: 3   
1 2 3 #SUP: 2   
1 2 5 #SUP: 2   
1 3 5 #SUP: 2   
2 3 5 #SUP: 3   
1 2 3 5 #SUP: 2

Lưu ý rằng nếu định dạng ARFF được sử dụng làm đầu vào thay vì định dạng đầu vào mặc định, định dạng đầu ra sẽ giống nhau ngoại trừ các mục sẽ được biểu diễn bằng chuỗi thay vì số nguyên.

Hiệu suất

Thuật toán Apriori là một thuật toán quan trọng vì lý do lịch sử và cũng bởi vì nó là một thuật toán đơn giản dễ học. Tuy nhiên, các thuật toán hiệu quả hơn và nhanh hơn đã được đề xuất. Nếu hiệu quả là cần thiết, nó được khuyến khích sử dụng một thuật toán hiệu quả hơn như FPGrowth thay vì Apriori. Bạn có thể thấy một so sánh hiệu suất của Apriori, FPGrowth và các thuật toán khai thác itemset thường xuyên khác bằng cách nhấp vào phần " [hiệu suất](http://www.philippe-fournier-viger.com/spmf/index.php?link=performance.php) " của trang web này.

Chi tiết triển khai

Trong SPMF, cũng có một sự thực thi Apriori sử dụng một **cây băm** như một cấu trúc bên trong để lưu trữ các ứng cử viên. Cấu trúc này cung cấp một cách hiệu quả hơn để đếm sự hỗ trợ của các tập hợp. Phiên bản Apriori này có tên là " **Apriori\_with\_hash\_tree** " trong GUI của SPMF và dòng lệnh. Đối với phiên bản mã nguồn, nó có thể được chạy bằng cách thực thi tệp kiểm tra **MainTestAprioriHT\_saveToFile.java** . Phiên bản Apriori này có thể nhanh gấp hai lần so với phiên bản thông thường trong một số trường hợp nhưng nó sử dụng nhiều bộ nhớ hơn. Phiên bản Apriori này có hai tham số: (1) minsup và (2) số lượng nút con mà mỗi nút trong cây băm nên có. Đối với tham số thứ hai, chúng tôi khuyên bạn nên sử dụng giá trị 30.