KHAI PHÁ ĐỒ THỊ VÀ PHÂN TÍCH MẠNG XÃ HỘI

GRAPH MINING AND ANALYS SOCIAL NETWORK

Lương Thị Thảo Hiếu

Khoa Công nghệ thông tin, Trường Đại học Kinh tế - Kỹ thuật Công nghiệp

Đến tòa soạn ngày 01/4/2016, chấp nhận đăng ngày 25/4/2016

Tóm tắt: Sự thành công của thuật toán khai phá tập mục (mẫu) thường xuyên trước đây chỉ hiệu quả trên dữ liệu có cấu trúc đơn giản. Trong thực tế, nhiều ứng dụng cần mô hình biểu diễn dữ liệu đòi hỏi sự phức tạp cao hơn so với mẫu thường xuyên. Cấu trúc đồ thị ra đời, với cấu trúc tinh vi và khả năng tương tác cao, kết hợp với phương pháp khai phá đồ thị được ứng dụng rộng rãi trong các mô hình tin sinh học, phân tích web, phân tích cấu trúc XML, phân loại video…

Trong bài báo này chúng tôi nghiên cứu thuật toán GSPAN, tìm các đồ thị con xuất hiện thường xuyên từ tập đồ thị đầu vào áp dụng phân tích mối quan hệ của nhóm người dùng trên mạng facebook.

Từ khóa: GSPAN, substructure mining, graph isomorphism, social network.

Abstract: With the successful of algorithm before only effective for mining simple frequent itemsets. Infact, there are many applications require a pattern more complicated than former. Since graph has been intensively with sophisticated structure and high interoperability, combined with graph mining methods used in chemical informatics,analys website and XML document, video indexing, text retrievial. In this paper, we research GSPAN algorithm for frequent subgarph mining in graph datasets, applying analytical relationship in facebook network.

Keywords: GSPAN, substructure mining, graph isomorphism, social network.

1. GIỚI THIỆU

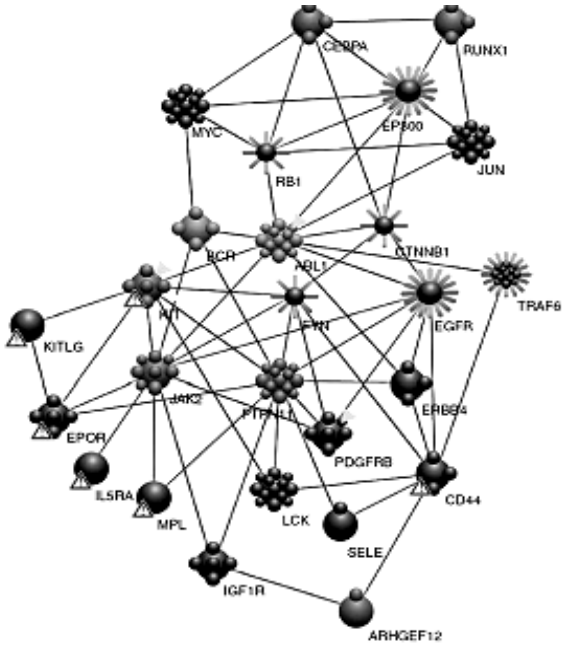
Khai phá các mẫu lặp lại nhiều lần trong dữ liệu có cấu trúc (như đồ thị, cây) thu hút sự chú ý của các nhà nghiên cứu và được ứng dụng trong nhiều lĩnh vực khác nhau. Các mẫu lặp lại, giúp ta hiểu sâu về mối quan hệ phức tạp giữa các phần tử được mô hình hóa. Với cấu trúc tinh vi, đồ thị được gán nhãn có thể mô hình hóa cho nhiều mối quan hệ phức tạp. Nhãn cho đỉnh và cạnh có thể biểu diễn các thuộc tính của các thực thể và mối liên hệ giữa chúng. Chẳng hạn, trong sinh học đồ thị được dùng để mô tả mối quan hệ giữa các phần tử cơ bản (protein, gene, RNA). Trong hóa học phân tích, đồ thị được dùng để mô tả cấu trúc ba chiều của các phân tử. Ngoài ra, đồ thị còn dùng biểu diễn dữ liệu web, text, ...

Sử dụng đồ thị và khai phá đồ thị [3] rất phù hợp với các bài toán liên quan phân tích dữ liệu web.

Cho đến nay, có nhiều thuật toán được đề xuất cho việc khai phá các đồ thị con phổ biến từ một cơ sở dữ liệu đồ thị (CSDLĐT). Đồ thị con phổ biến là đồ thị con có số lần xuất hiện trong một CSDLĐT lớn hơn một ngưỡng cho trước.

Nhìn chung, có thể chia các thuật toán khai phá đồ thị con phổ biến thành hai nhóm. Nhóm 1, gồm các thuật toán sử dụng chiến lược

tìm kiếm theo chiều rộng theo kiểu thuật toán Apriori, được giới thiệu bởi R.Argrawal và R.Srikant [1], hai trong số các thuật toán này là AGM và FSG. Nhóm 2, gồm các thuật toán sử dụng chiến lược tìm kiếm theo chiều sâu, tiêu biểu cho nhóm này là GSPAN [2]. Trong GSPAN [2], một đồ thị con phổ biến *G* được sử dụng để sinh ra các đồ thị con ứng viên G' bằng cách chọn một đỉnh *v* thuộc *G* và thêm vào cạnh (*v, w*), trong đó *w* thuộc *G* hoặc không thuộc *G*, không liệt kê tất cả các ứng cử viên như tiếp cận Apriori, điều này làm giảm bớt thách thức về mặt tính toán khi CSDLĐT đầu vào lớn. Trong bài báo này, chúng tôi tìm hiểu các khái niệm đồ thị, đẳng cấu đồ thị, thuật toán GSPAN, áp dụng cài đặt, phân tích mối quan hệ nhóm người dùng trên mạng facebook.



Hình 1. Cấu trúc protein

2. MỘT SỐ KHÁI NIỆM

Định nghĩa 1. Đồ thị $G=(V,E,L_V,L_E,l)$ là đồ thị con $G'=(V',E',L'_V,L'_E,l')$khi và chỉ khi:

- $V \subseteq V'$
- $\forall u \in V, (l(u)=l'(u))$,
- $E \subseteq E'$,
- $\forall (u,v) \in E, (l(u,v)=l'(u,v))$

Định nghĩa 2. Đồ thị $G=(V,E,L_V,L_E,l)$ là đẳng cấu với đồ thị $G'=(V',E',L'_V,L'_E,l')$ khi và chỉ khi tồn tại song ánh : $f : V \to V'$ thỏa mãn:

- $\forall u \in V, (l(u)=l'(f(u)))$,
- $\forall u,v \in V, ((u,v)\in E \Leftrightarrow (f(u), f(v)\in E'))$,
- $\forall u,v \in E, (l(u,v)=l'(f(u), f(v)))$

Ta nói rằng *G* đẳng cấu với G' và ngược lại.

Định nghĩa 3. Cho tập các đồ thị D và một ngưỡng σ (0 < σ ≤ 1), độ hỗ trợ (support) của đồ thị *G* kí hiệu Sup_G là tỉ số của số lượng đồ thị G' ($G' \in$ D) chứa *G* chia cho số lượng đồ thị có trong D:

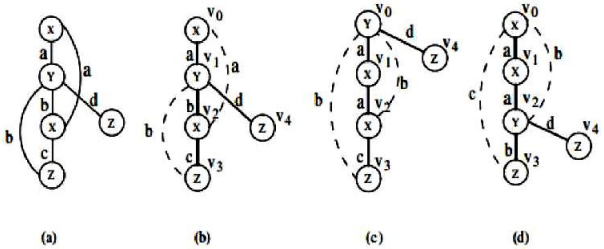
$$Sup_G = \frac{|\{G' \in D | G' \subseteq G\}|}{D}$$

Đồ thị *G* được gọi là đồ thị con phổ biến khi và chỉ khi $Sup_G \geq \sigma$

Định nghĩa 4. Bài toán khai phá đồ thị con phổ biến là bài toán cho trước CSDLĐT và ngưỡng hỗ trợ σ (0 < σ ≤ 1), tìm tất cả các đồ thị con phổ biến trong CSDLĐT đó.

Trong khai phá đồ thị, kỹ thuật tìm kiếm theo chiều sâu DFS trên cây DFS được sử dụng phổ biến [5]. Một đồ thị có thể có nhiều cây DFS khác nhau.

Ví dụ 1. Các cạnh in đậm trong hình (b-d) biểu diễn 3 cây DFS khác nhau cho đồ thị a. Từ một đồ thị có nhiều cách xây dựng cây DFS bằng cách lựa chọn điểm xuất phát khác nhau.



Hình 2(b) - (d) ba cây DFS cho đồ thị hình 2 (a)

Chúng ta cần định nghĩa mã DFS cho một cây DFS cho trước từ đồ thị ban đầu. Sử dụng mã DFS giúp ta có thể ánh xạ các đồ thị vào một cây tìm kiếm phân cấp. Sau đó xây dựng

cây mã DFS để mô hình hoá mối quan hệ giữa các đồ thị, trong đó mỗi node biểu diễn một đồ thị và bất kỳ đồ thị nào cũng có thể tìm các node của nó trong cây mã DFS. Một đồ thị có thể có vài mã DFS trong cây mã DFS. Mã DFS đầu tiên là mã DFS tối tiểu. Bằng cách tỉa các node không chứa mã DFS tối tiểu, ta có thể giảm không gian tìm kiếm.

Đánh chỉ số DFS: Ký hiệu cây DFS là T. Tìm kiếm theo chiều sâu phát hiện các đỉnh theo thứ tự tuyến tính. Sử dụng chỉ số dưới để gán nhãn cho thứ tự này dựa trên thứ tự xuất hiện.

Cho cây T, $v_i \prec_T v_j$ nghĩa là v_i được phát hiện trước v_j. Gọi v_0 là nút gốc còn v_n là *nút ngoài cùng bên phải*, đường nối từ v_0 tới v_n là *đường ngoài cùng bên phải*. *Nút ngoài cùng bên phải* và *đường ngoài cùng bên phải* đóng vai trò quan trọng trong khai khá thêm mẫu mới. Trong hình 2(b) đường ngoài cùng bên phải là (v_0 ,v_1,v_4), trong hình 2(c) là (v_0 ,v_4) và trong 2(d) là (v_0 ,v_1, v_2,v_4). Biểu diễn chỉ số dưới của G là G_T.

Tập cạnh trước và tập cạnh sau: Cho G_T, tập cạnh trước là tập tất cả các cạnh trong cây DFS và cạnh sau là tập tất cả các cạnh không nằm trong cây DFS.

Thứ tự tuyến tính: Một thứ tự tuyến tính $\prec_T$ được xây dựng dựa trên tất cả các cạnh có trong đồ thị G bởi các luật sau:

Giả sử: $e_1 = (i_1 , j_1)$, $e_2 = (i_2 , j_2)$

i) if $(i_1 = i_2)$ *and* $(j_1 < j_2)$ *then* $e_1 \prec_T e_2$

ii) if $(i_1 < j_1)$ *and* $(j_1 = j_2)$ *then* $e_1 \prec_T e_2$

iii) if $(e_1 \prec_T e_2)$ *and* $(e_2 \prec_T e_3)$ *then* $e_1 \prec_T e_3$

Định nghĩa 5. Mã DFS

Cho một cây DFS T của một đồ thị G, chuỗi thứ tự các cạnh (e_i) có thể được xây dựng dựa trên $\prec_T$, là $e_i \prec_T e_{i+1}$, trong đó $i = 0, \ldots, |E| - 1$. (e_i) được gọi là một mã DFS, ký hiệu Code (G, T).

Ví dụ 2. Với cây DFS trong 2(b), mã DFS của đồ thị là ((v_0,v_1), (v_1,v_2), (v_2,v_0), (v_2,v_3), (v_3,v_1), (v_1,v_4)). Với cây DFS trong 2 (c), mã DFS của đồ thị là ((v_0,v_1), (v_1,v_2), (v_2,v_0), (v_2, v_3), (v_3,v_0), (v_0,v_4)). Như vậy, với cùng một đồ thị các cây mã DFS khác nhau có thể tạo ra các mã DFS khác nhau.

Biểu diễn một cạnh là (v_i,v_j), nhãn của hai đỉnh tương ứng của nó là $l(v_i)$,$l(v_j)$ và nhãn ghi trên cạnh là $l(v_i,v_j)$. Để đơn giản tổ hợp lại thành một bộ: $(i,j,l_i, l(v_i,v_j),l_j)$.

Ví dụ 3. Với hai đỉnh (v_0,v_1) trong hình 2(b) được biểu diễn bằng (0, 1, *X*, a, *Y*).

Bảng 1. Mã DFS cho hình 2(b), 2(c), 2(d)

edge	(Fig 1b) α	(Fig 1c) β	(Fig 1d) γ
0	$(0,1,X,a,Y)$	$(0,1,Y,a,X)$	$(0,1,X,a,X)$
1	$(1,2,Y,b,X)$	$(1,2,X,a,X)$	$(1,2,X,a,Y)$
2	$(2,0,X,a,X)$	$(2,0,X,b,Y)$	$(2,0,Y,b,X)$
3	$(2,3,X,c,Z)$	$(2,3,X,c,Z)$	$(2,3,Y,b,Z)$
4	$(3,1,Z,b,Y)$	$(3,0,Z,b,Y)$	$(3,0,Z,c,X)$
5	$(1,4,Y,d,Z)$	$(0,4,Y,d,Z)$	$(2,4,Y,d,Z)$

Từ các mã được tạo, cần chọn ra một mã chính tắc. Trong phần tiếp theo, tập trung xây dựng thứ tự tuyến tính cho các mã DFS này, để giải quyết với vấn đề gán nhãn đồ thị, từ đó tìm ra một mã DFS chính tắc.

Định nghĩa 6. Thứ tự từ điển DFS

Cho *Z= { Code(G,T) | T là một cây DFS của G }*, Z là một tập chứa tất cả các mã DFS cho tất cả các đồ thị có nhãn liên tục. Giả sử có một thứ tự tuyến tính ($\prec_L$) trong tập nhãn (L) khi đó sự kết hợp theo thứ tự từ điển của $\prec_T$ và $\prec_L$ là một thứ tự tuyến tính ($\prec_e$) trên tập $E_T \times L \times L \times L$.

Thứ tự từ điển là một thứ tự tuyến tính được định nghĩa như sau:

Cho $\alpha = Code(G_\alpha, T_\alpha) = (a_0, a_1, \ldots a_m)$ và $\beta = Code(G_\beta, T_\beta) = (b_0, b_1, \ldots b_n), \alpha, \beta \in Z$.

$\alpha \le \beta$ nếu cả hai điều kiện dưới đây cùng thỏa mãn:

$\exists t, 0 \le t \le \min(m,n), a_k = b_k$ với $k < t, a_t \prec_e b_t$

$a_k = b_k$ với $0 \le k \le m, n > m$

Đồ thị trong hình 2(a) có mười mã DFS khác nhau, ba trong số chúng được xây dựng dựa trên cây DFS hình 2(b)-(d) được liệt kê trong bảng 1. Theo quy tắc thứ tự từ điển DFS ta có $\gamma \prec \alpha \prec \beta$.

Định nghĩa 7. Mã DFS tối tiểu

Cho một đồ thị G, *Z(G) = { Code(G,T) | T là một cây DFS của G }*. Dựa trên thứ tự từ điển DFS, một tối tiểu *min(Z(G))* được gọi là mã DFS tối tiểu của G, cũng được gọi là mã chính tắc của G.

Định lý 1. Cho đồ thị G và G', G là đẳng cấu với G' khi và chỉ khi *min(G) = min(G')*.

Định nghĩa 8. Mã DFS cha và con

Cho mã DFS $\alpha = (a_0, a_1, ..., a_m)$ và mã DFS có giá trị bất kỳ $\beta = (a_0, a_1, ..., a_m, b)$, β được gọi là con của α, và α được gọi là cha của β.

Ký hiệu $anc(\beta)$={các mã DFS cha của β) và $dec(\alpha)$={các mã DFS con của α).

Định nghĩa 9. Cây mã DFS

Trong một cây mã DFS, mỗi node biểu diễn một mã DFS, mối quan hệ giữa node cha và node con tuân theo mối quan hệ cha - con đã mô tả trong định nghĩa trên. Mối quan hệ anh - em tuân theo thứ tự từ điển DFS. Ký hiệu cây mã là T.

Tính chất 1. Tính phủ của cây DFS

Cây DFS chứa các mã DFS tối tiểu của tất cả các đồ thị.

Định lý 2. Bảo tồn tính thường xuyên

Nếu một đồ thị G là thường xuyên thì bất kỳ đồ thị con nào của G cũng thường xuyên. Nếu G không thường xuyên thì bất kỳ đồ thị nào chứa G cũng không thường xuyên. Điều này tương đương, nếu một mã DFS α thường xuyên, với $\forall \beta \in anc(\alpha)$ thì β là thường xuyên. Nếu α là không thường xuyên với $\forall \beta \in dec(\alpha)$ thì β là không thường xuyên.

Rõ ràng trong một cây DFS T tồn tại các mã DFS khác nhau cho một đồ thị. Theo định nghĩa của mã DFS tối tiểu sự xuất hiện của mã DFS đầu tiên của một đồ thị trong cây T chính là mã DFS tối tiểu của nó.

Định lý 3. Tỉa mã DFS

Cho đồ thị G, các mã DFS trong cây T là $\alpha_0, \alpha_1, ..., \alpha_n, \forall i, j \le n, \alpha_i \le \alpha_j$, (theo thứ tự từ điển DFS) và α_0 là mã DFS tối tiểu của G. Các cây mã DFS còn lại sau khi tỉa $\alpha_i (1 \le i \le n)$ và tất cả các con của nó vẫn được bảo tồn.

Các định lý trên phát hiện tất cả các đồ thị con thường xuyên. Bằng cách tìm kiếm có thứ tự trong cây DFS, đảm bảo có thể khai phá tất cả các đồ thị con thường xuyên. Việc tỉa mã DFS sẽ loại bỏ tất cả các node trùng trong cây, trong khi đó tính chất bảo tồn tính thường xuyên giúp tìm ra tất cả các đồ thị con thường xuyên mà không gây xung đột và mất mát dữ liệu.

3. THUẬT TOÁN GSPAN [2]

Trong phần này chúng tôi trình bày thuật toán GSPAN dựa trên kết quả các nghiên cứu về thứ tự từ điển, các tính chất và định lý trình bày trong phần trên. GSPAN sử dụng cấu trúc danh sách kề để lưu trữ đồ thị. Thuật toán 1 tóm tắt mã giả với D biểu diễn tập đồ thị đầu vào, S chứa kết quả khai phá.

Thuật toán 1 GraphSet_Projection (D, S)

1. Sắp xếp các nhãn trong D theo thứ tự độ thường xuyên của chúng
2. Loại bỏ các đỉnh, cạnh không thường xuyên
3. Gán nhãn cho các đỉnh và các cạnh còn lại
4. $S^1 \leftarrow$ Các đồ thị 1 cạnh thường xuyên trong D
5. Sắp xếp S^1 theo thứ tự từ điển DFS
6. $S \leftarrow S^1$
7. **For each** $e \in S^1$ **do**
8. Khởi tạo s với cạnh e, tập s. D là tập đồ thị chứa cạnh e
9. Subgraph_Mining (D, S, e)
10. $D \leftarrow D - e$
11. **if** $|D| < \min Sup$
12. **break**

Bước 1: Dòng 1-6 loại bỏ các đỉnh và các cạnh không thường xuyên trong tập đồ thị D.

Bước 2: Dòng 7-9, với mỗi đồ thị con một cạnh thường xuyên, gọi thủ tục

Subgraph_Mining tăng trưởng tất cả các node trong cây con có gốc trong đồ thị 1 cạnh này.

Bước 3: Dòng 10 tăng trưởng mỗi đồ thị trong tập đồ thị D.

Bước 4: Kết thúc khi tất cả các đồ thị con thường xuyên 1 cạnh và các con của nó được tạo ra.

Thủ tục Subgraph_Mining được gọi lặp lại để tìm tất cả các đồ thị và các đồ thị con thường xuyên của chúng. Thủ tục ngừng tìm kiếm khi độ hỗ trợ của một đồ thị nhỏ hơn minSup hoặc mã của nó không là tối tiểu.

Subprocedure Subgraph_Mining(D,S,s)

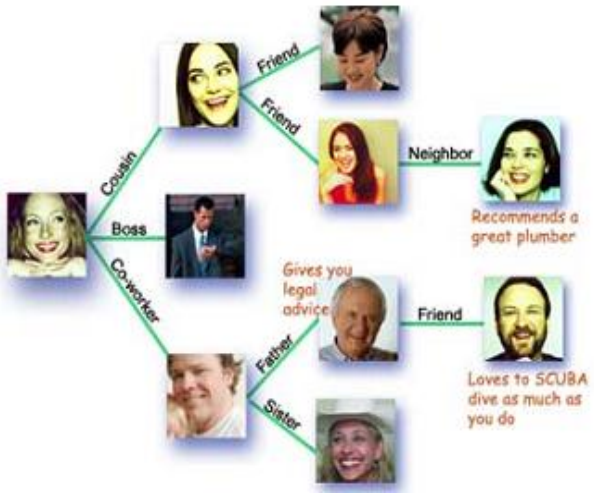
1. **if** $s \neq \min(s)$
2. **Return**;
3. $S \leftarrow S \cup \{s\}$
4. Liệt kê s trong mỗi đồ thị trong D và đếm con của s
5. **For each c**, c là con của s **do**
6. **If** support(c)$\geq$ minSup
7. $s \leftarrow c$
8. Subgraph_Mining(D,S,s)

4. CÀI ĐẶT GSPAN - PHÂN TÍCH QUAN HỆ NHÓM NGƯỜI DÙNG TRÊN FACEBOOK

4.1. Phân tích mạng xã hội [4]

Theo quan điểm của khai phá dữ liệu, mạng xã hội là một tập dữ liệu phức tạp và đa quan hệ được biểu diễn dưới dạng đồ thị. Cụ thể: các node tương ứng với các đối tượng và các cạnh tương ứng với các liên kết biểu diễn mối quan hệ hoặc tương tác giữa các đối tượng. Cả node và liên kết đều có thuộc tính. Các đối tượng có thể gán nhãn. Các liên kết có thể là có hướng hoặc vô hướng… Trong mạng xã hội thể hiện rất nhiều mối quan hệ khác nhau. Sử dụng khai phá đồ thị chúng ta có thể phát hiện mối quan hệ trong đó.

Theo tư duy toán học cộng đồng mạng có thể được cụ thể hoá bằng một đồ thị lớn, trong đó các đỉnh thể hiện cho người dùng, còn các cạnh và nhãn biểu hiện cho mối quan hệ. Do có nhiều mối quan hệ nên các nhãn của đồ thị rất phức tạp. Trong chương trình xây dựng, chúng tôi mô hình hoá cộng đồng người dùng facebook bằng nhiều đồ thị khác nhau, mỗi đồ thị gồm nhiều đỉnh với các cạnh gán nhãn phản ánh một mối quan hệ cụ thể. Sử dụng thuật giải GSPAN nghiên cứu ở trên, chúng tôi xây dựng phần mềm, đầu vào là các đồ thị thu thập từ dữ liệu facebook của nhiều người dùng, chương trình sẽ tìm ra các đồ thị con tương đương với nhóm các người dùng có mối liên hệ với nhau theo một số quan hệ cụ thể.

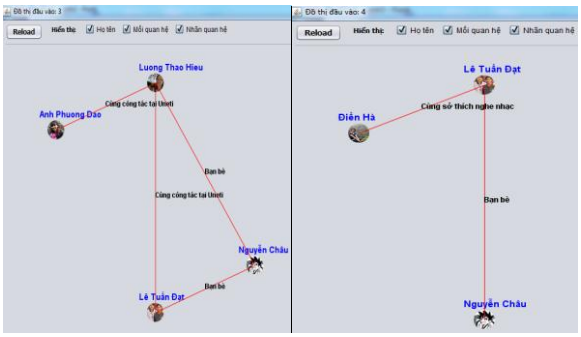


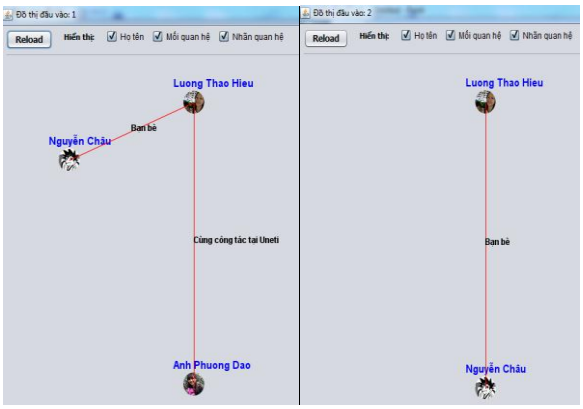
Hình 3. Mối quan hệ trên facebook

4.2. Thực nghiệm và kết quả

Dữ liệu thử nghiệm là tệp lưu trữ CSDL đồ thị đầu vào. Chúng tôi tiến hành thu thập dữ liệu từ thông tin trên face của nhiều người dùng khác nhau, mỗi người có một đồ thị tương ứng, với các cạnh thể hiện mối quan hệ. Có nhiều mối quan hệ khác nhau, ở đây chúng tôi chọn bốn loại quan hệ cụ thể để mô phỏng: công tác tại uneti, bạn bè, cùng sở thích nghe nhạc, cùng khám bệnh tại bệnh viện dệt may.

Chương trình đọc dữ liệu lưu trữ các đồ thị đầu vào, với bộ dữ liệu 4 đồ thị (hình 4).



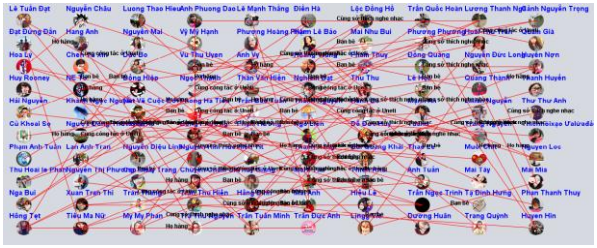


Hình 4. Đồ thị đầu vào số đỉnh nhỏ

Với độ hỗ trợ minsup bằng 50%, chương trình sinh ra hai đồ thị con thường xuyên.

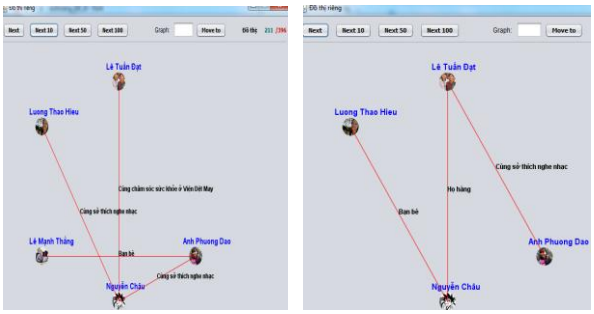
Nhận xét: Từ 4 đồ thị đầu vào với các nhãn thể hiện mối quan hệ khác nhau, sinh ra hai đồ thị con thường xuyên. Đồ thị thứ nhất xuất hiện 2 lần, đồ thị thứ hai xuất hiện 3 lần trong 4 đồ thị đầu vào, thoả mãn độ hỗ trợ minsup. Từ hai đồ thị con này, chúng ta có thể rút ra giữa người dùng Lương Thảo Hiếu và Anh Phương Đào có mối quan hệ thứ nhất là cùng công tác tại Uneti, và mối quan hệ thứ hai giữa người dùng Lương Thảo Hiếu và Nguyễn Châu là quan hệ bạn bè.

Ở hình trên chúng tôi minh hoạ các đồ thị với số đỉnh nhỏ để người đọc dễ tiện theo dõi dạng đồ hoạ. Thực hiện thử nghiệm với các đồ thị lớn hơn cho kết quả tương tự.



Hình 5. Đồ thị đầu vào với số đỉnh 100

Với độ hỗ trợ minsup=60% thu được 159 đồ thị con thường xuyên, hình ảnh một số đồ thị con thu được như sau:



Từ các đồ thị con sinh ra, mỗi đồ thị thể hiện mối quan hệ cụ thể, rõ ràng ta có thể thấy được mối quan hệ trong từng nhóm người. Kết quả khi thực hiện thực nghiệm trên các bộ dữ liệu có số đỉnh, số cạnh và số quan hệ lớn cho kết quả khả quan. Tính đúng đắn của thuật toán được thể hiện khi dữ liệu đồ thị đầu vào lớn và độ hỗ trợ minsup nhỏ chương trình chạy chậm và ngược lại.

Bảng 2. Mối liên hệ giữa thời gian và độ hỗ trợ

Số đỉnh	Độ hỗ trợ (minsup)	Thời gian (s)
100	50	34,41
	60	31,983
	75	30,071
	80	30,056
35	50	23,16
	60	22,255
	75	22,25
	80	21,848
28	50	19,981

Số đỉnh	Độ hỗ trợ (minsup)	Thời gian (s)
	60	19,574
	75	18,562
	80	18,42

5. KẾT LUẬN

Trong bài báo đã trình bày khai phá đồ thị, ứng dụng của khai phá đồ thị trong phân tích mạng xã hội, sau đó dựa trên tìm hiểu thuật toán khai phá đồ thị con thường xuyên GSPAN, dùng công cụ ngôn ngữ lập trình java, sử dụng máy ảo với các kỹ năng xử lý dữ liệu đồ thị, để biểu diễn đồ thị dạng đồ hoạ trực quan, kết hợp với tìm hiểu về phân tích mạng xã hội chúng tôi đã xây dựng phần mềm phân tích mối quan hệ của nhóm người trên mạng facebook.

Kết quả thực nghiệm trên CSDLĐT đầu vào chứa một số đồ thị lưu thông tin của các nhóm người dùng cụ thể, qua chương trình có thể tìm ra nhóm người có quan hệ với nhau: công tác trong Trường Đại học Kinh tế Kỹ thuật Công nghiệp, tham gia khám bệnh tại Bệnh viện Dệt May, hoặc họ hàng… Việc tìm ra mối liên hệ giữa nhóm các người dùng này có ý nghĩa nhất định trong thực tế và được sử dụng trong từng trường hợp cụ thể.

TÀI LIỆU THAM KHẢO

[1] R.Agrawal and R. Srikant. "*Fast algorithms for mining association rules*". In Proc. 1994 Int. Conf. Very Large Data Bases (VLDB '94), pages 487-499, Santiago, Chile, Sept.1994.

[2] Xifeng Yan, JiaWeiHan, gSpan: "*Graph_Based Substruct Pattern Mining*," Department of Computer Science University of Illiois at Urbana-Champaign, September 3, 2002.

[3] Cook, D.J and Holder, L.B.2006 Mining Graph Data. John Wiley & Sons.

[4] http://web.engr.illinois.edu/~hanj/cs512/bk2chaps/chapter_9.pdf

[5] T. H. Cormen. C. E. Leiserson. R. L. Rivest and C. Stein. "*Introduction to Algorithms*". MIT Press, 2001, SecondEdition.

Thông tin liên hệ: **Lương Thị Thảo Hiếu**

Điện thoại: 0942160880 - Email:ltthieu@uneti.edu.vn

Khoa Công nghệ thông tin, Trường Đại học Kinh tế - Kỹ thuật Công nghiệp