Tipo de artículo: Artículo original
Publicado: dd/mm/aa En: XIII Congreso Nacional de Reconocimiento de Patrones

Minería de subgrafos frecuentes aproximados para multi-grafos basada en transformaciones.

Frequent Approximate Subgraph Mining for Multi-Graphs Based on Transformations.

Niusvel Acosta-Mendoza[12*], Andrés Gago-Alonso[1], José Eladio Medina-Pagola[1], Jesús Ariel Carrasco-Ochoa[2], José Fco. Martínez-Trinidad[2]

[1]Centro de Aplicaciones de Tecnologías de Avanzada (CENATAV). 7a No. 21406 e/ 214 and 216, Siboney, Playa, CP 12200, La Habana, Cuba.
[2]Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE). Luis Enrique Erro No. 1, Sta. Maria Tonantzintla, Puebla, CP 72840, México.

*Autor para correspondencia: nacosta@cenatav.co.cu

Resumen
En este trabajo se realiza una comparación entre los métodos basados en transformaciones de grafos reportados en la literatura. El uso de estos métodos permite la realización de la minería de subgrafos frecuentes en colecciones de multi-grafos. Tales métodos son descritos y analizados desde el punto de vista de su complejidad computacional. Adicionalmente, la eficiencia de estos métodos es evaluada sobre diferentes colecciones sintéticas de multi-grafos. Esta comparación nos permitió arribar a varias conclusiones útiles para futuras aplicaciones de estos métodos.

Palabras claves: Transformación de grafos, minería de multi-grafos, subgrafos frecuentes.

Abstract
In this paper, a comparison between the reported methods, which are based on graph transformations, is shown. Using these methods, it has possible performed mining frequent subgraphs on multi-graph collections. In this work, a description and an analysis from the computational complexity viewpoint is performed. Besides, the efficiency of these methods is evaluated over different synthetic multi-graph collections. Finally, we arrive to several useful conclusions for future applications of these methods.

Keywords: *Graph transformation, multi-graph mining, frequent subgraphs.*

Introducción

En la última década, el uso de la minería de subgrafos frecuentes aproximados (*minería de SFAs*) sobre grafos simples ha alcanzado gran aceptación en la comunidad científica (Patel and Oza, 2014). Sin embargo, a pesar de que mediante estos enfoques se han alcanzado resultados alentadores, existen aplicaciones donde los datos son modelados mediante estructuras más complejas (Hu et al., 2014; Bothorel et al., 2015; Setak et al., 2015). En estos trabajos se modelan los datos utilizando multi-grafos, donde un multi-grafo es un grafo que puede contener lazos y múltiples aristas (*multi-aristas*) entre dos vértices. Los autores de estos trabajos plantean que en algunas aplicaciones pueden existir más de una relación (arista) entre las entidades u objetos (vértices) y esta información no puede ser modelada mediante grafos simples. Este tipo de representación de los datos no permite la aplicación de los algoritmos tradicionales para la minería de SFAs, porque están diseñados para trabajar con grafos simples.

Para dar solución a esta problemática se han reportado algunos métodos basados en transformaciones de grafos que permiten la aplicación de los algoritmos tradicionales para la minería de SFAs en contextos de multi-grafos (Acosta-Mendoza et al., 2015a,b). Estos métodos están diseñados para identificar diferentes tipos de patrones aunque se aplique el mismo algoritmo para la minería de SFAs en ambos casos. Esto hace que tengan diferentes aplicaciones en el contexto de los multi-grafos. Por este motivo, en este artículo se realiza una descripción de estos métodos enfocada al tipo de patrones que permiten identificar. De igual manera se describen los contextos donde estos métodos son aplicables. Además, es importante conocer la complejidad computacional de dichos métodos y por esta razón, en este trabajo se presenta un análisis de sus complejidades a modo de comparación.

Materiales y métodos o Metodología computacional

En esta sección se presentan las notaciones y definiciones necesarias para la comprensión del resto del trabajo. Además, se brinda una breve descripción de los métodos basados en transformación de grafos para la minería de SFAs en colecciones de multi-grafos.

Marco teórico

Como en este artículo se tratan los grafos no dirigidos y etiquetados, entonces se formaliza este tipo de grafo mediante la definición 1.

Definición 1 (Grafo etiquetado) *Sea $L = L_V \cup L_E$ el dominio de todas las posibles etiquetas, donde L_V y L_E son los conjuntos de etiquetas para los vértices y las aristas respectivamente, un grafo etiquetado (o simplemente grafo) es una 5-tupla, $G = (V, E, \phi, I, J)$, donde V es un conjunto de elementos llamados vértices,*

E es un conjunto de elementos llamados aristas, $\phi : E \to V \times V$ es la función de incidencia *(la arista e, a través de la función $\phi(e)$, conecta los vértices u y v si $\phi(e) = \{u, v\}$), $I : V \to L_V$ es una función etiquetadora que asigna etiquetas a los vértices y $J : E \to L_E$ es una función etiquetadora que asigna etiquetas a las aristas.*

Definición 2 (Subgrafo y supergrafo) *Sean $G_1 = (V_1, E_1, \phi_1, I_1, J_1)$ y $G_2 = (V_2, E_2, \phi_2, I_2, J_2)$ dos grafos, G_1 es un subgrafo de G_2 si $V_1 \subseteq V_2$, $E_1 \subseteq E_2$, ϕ_1 es una restricción de ϕ_2 al V_1, I_1 es una restricción de I_2 al V_1, y J_1 es una restricción de J_2 al E_1. En este caso G_2 es un supergrafo de G_1, denotado por $G_1 \subseteq G_2$.*

Definición 3 (Multi-arista, arista simple y lazo) *Una arista $e \in E$, donde $\phi(e) = \{u, v\}$ y $u \neq v$, es una* multi-arista *si existe $e' \in E$ tal que $e \neq e'$ y $\phi(e) = \phi(e')$; pero si $|\phi(e)| = 1$, e es un* lazo *(i.e. $\phi(e) = \{u\} = \{v\}$); en otro caso e es una* arista simple.

De esta manera, el conjunto de aristas E de un grafo puede ser particionado en tres subconjuntos disjuntos $E^{(s)}$, $E^{(m)}$, y $E^{(l)}$ conteniendo aristas simples, multi-aristas y lazos, respectivamente.

Definición 4 (Grafo simple y multi-grafo) *Un grafo es* grafo simple *si no tiene ni lazos, ni multi-aristas, i.e. tiene solo aristas simples, $E = E^{(s)}$, siendo $E^{(m)} = \emptyset$ y $E^{(l)} = \emptyset$; en otro caso, G es un* multi-grafo.

Definición 5 (Grafo retornable) *Sean κ y ϱ dos etiquetas diferentes para vértices que son usadas para representar lazos y aristas. El grafo simple $G' = (V', E', \phi', I', J')$ es un* grafo retornable *a un multi-grafo si cumple con las siguientes condiciones: cada vértice $v \in V'$ con $I'(v) = \varrho$ tiene exactamente dos aristas e_1 y e_2, tal que $J'(e_1) = J'(e_2)$, y cada vértice $v \in V'$ con $I'(v) = \kappa$ tiene exactamente una arista.*

Finalmente, las definiciones de subgrafo frecuente y minería de SFAs utilizadas en este artículo son las presentadas en (Morales-González et al., 2014; Acosta-Mendoza et al., 2015a,b).

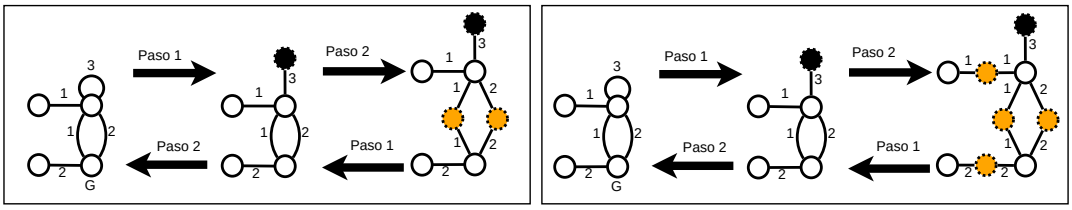
Métodos basados en transformaciones de grafos

Como se mencionó anteriormente, el enfoque basado en transformaciones de grafos se puede utilizar para realizar la minería de SFAs en colecciones de multi-grafos. Este enfoque nos permite aplicar algoritmos tradicionales para la minería de SFAs sobre colecciones de multi-grafos. Basados en este enfoque se han reportado dos métodos: el primero (*onlyMulti*) fue presentado en (Acosta-Mendoza et al., 2015b), y el segundo, (*allEdges*) fue reportado en (Acosta-Mendoza et al., 2015a). Estos métodos contienen dos algoritmos: uno que permite

transformar un multi-grafo G en un grafo simple G' y otro que transforma el grafo G' en el multi-grafo G. Este proceso de transformación permite realizar la minería de SFAs sin perder información semántica ni topológica en el contexto de los multi-grafos originales (Acosta-Mendoza et al., 2015a,b).

El algoritmo para transformar multi-grafos en grafos simples, *M2Simple* del método onlyMulti, está compuesto por dos pasos, como se ilustra en la figura 1(a): (1) cada lazo es transformado en una arista simple, agregando un nuevo vértice con la etiqueta especial "κ" y una arista que une el vértice del lazo y el nuevo vértice, manteniendo la etiqueta del lazo, y (2) se sustituyen cada multi-arista por dos aristas simples, agregando un nuevo vértice con la etiqueta especial "ϱ" y dos aristas con la etiqueta de la multi-arista eliminada, donde cada una enlaza un vértice de la multi-arista con el nuevo vértice. Por otro lado, este algoritmo - para el método allEdges - realiza el primer paso descrito anteriormente pero el segundo paso es diferente (ver figura 1(b)). En este caso, no solo se sustituyen las multi-aristas sino que todas las aristas son transformadas. Es importante señalar que las etiquetas κ y ϱ solo pueden ser utilizadas para identificar modificaciones de lazos y aristas, no pueden formar parte del conjunto posible de etiquetas utilizadas en los datos originales.

El algoritmo para transformar grafos simples en multi-grafos es común para ambos métodos (onlyMulti y allEdges). En este se recorren todos los vértices del grafo dado y cuando se identifica un vértice v con la etiqueta especial κ o ϱ se sustituye por un lazo o una arista entre los vértices adyacentes, respectivamente, eliminando la o las aristas que unían v al grafo. Este algoritmo solo puede aplicarse satisfactoriamente en los grafos que son retornables de acuerdo con la definición 5, siendo esta una condición necesaria para el proceso.



(a) Método onlyMulti. (b) Método allEdges.

Figura 1. Esquema del proceso de transformación de grafos.

Resultados y discusión

Los métodos basados en transformaciones de grafos (onlyMulti (Acosta-Mendoza et al., 2015b) y allEdges (Acosta-Mendoza et al., 2015a), descritos en la sección anterior) se diferencian por la forma de transformar un multi-grafo en un grafo simple. Esta diferencia hace que los algoritmos para la minería no identifiquen los mismos patrones sobre las colecciones de grafos resultantes. En onlyMulti las aristas simples se tratan

diferentes de las multi-aristas, mientras que en allEdges se tratan como iguales. Por este motivo, lo que para un método se identifica con una frecuencia determinada, no necesariamente debe tener la misma frecuencia en el otro método. Se puede decir que suponiendo que P_1 y P_2 son los conjuntos de patrones identificados como frecuentes por los métodos onlyMulti y allEdges respectivamente, entonces $P_1 \subseteq P_2$. Un ejemplo sencillo de esto se puede observar en la figura 2, donde $D = \{G_1, \ldots, G_6\}$, $P_1 = \{SF_1, SF_2\}$ y $P_2 = \{SF_1, \ldots, FS_3\}$ utilizando un umbral de soporte $\delta = 0{,}5$ para un enfoque de minería de subgrafos frecuentes. En este ejemplo se puede ver que el patrón SF_1 tiene frecuencia 1 en D y SF_2 tiene ocurrencias en los grafos G_1, G_3, G_6 para ambos métodos (onlyMulti y allEdges). Sin embargo, para el método allEdges, que trata por iguales las aristas simples y las multi-aristas, el patrón SF_2 tiene ocurrencias en G_1, G_3, G_5 y G_6, y el patrón SF_3 tiene ocurrencias en G_1, G_2, G_4 y G_6.

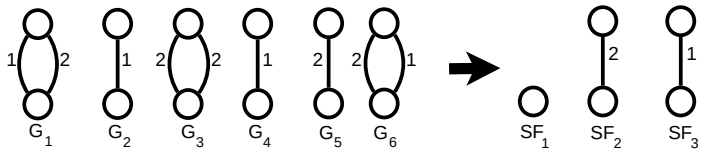


Figura 2. Ejemplo de subgrafos frecuentes identificados mediante los métodos basados en transformaciones de grafos.

Como se mencionó anteriormente, onlyMulti y allEdges realizan las transformaciones de grafos mediante dos algoritmos: uno que transforma un multi-grafo en un grafo simple (*M2Simple*) y otro que realiza la operación inversa (*S2Multi*). Para ambos métodos el algoritmo S2Multi es común, mientras que el algoritmo M2Simple -para el método onlyMulti- transforma solamente los lazos y las multi-aristas y -para allEdges- transforma todas las aristas (lazos, aristas simples y multi-aristas). Sin embargo, la complejidad computacional de M2Simple es $O(k * n^2 * d)$ en el peor de los casos para ambos métodos, donde k es la mayor cantidad de multi-aristas entre dos vértices, n es el número de vértices del grafo de la colección con mayor cantidad de vértices y d es la cantidad de grafos de la colección. Es importante señalar que en onlyMulti solamente se realiza la misma cantidad de transformaciones que en allEdges cuando todas las aristas son lazos y/o multi-aristas. El algoritmo S2Multi tiene una complejidad de $O((n + k * n^2) * d)$. Sin embargo, como no es común que todas las aristas del multi-grafo de entrada, transformado por el algoritmo M2Simple, sean lazos y/o multi-aristas, entonces, se puede llegar a la conclusión de que la complejidad de este algoritmo para el método onlyMulti es menor, en la mayoría de los casos, que para el método allEdges.

Teniendo en cuenta las diferencias, entre los métodos onlyMulti y allEdges, así como la complejidad computacional de los mismos, se puede identificar cuál es el método idóneo para algunas aplicaciones. Generalmente, lo común es que las aristas simples pueden tener ocurrencias en las múlti-aristas y viceversa, siendo allEdges el de mayor aplicabilidad. Sin embargo, onlyMulti es aplicable en contextos más específicos, donde la información de las multi-aristas en su conjunto tengan algún significado local y las aristas simples no puedan tener

Tabla 1. Tiempo de ejecución del método onlyMulti sobre diferentes colecciones sintéticas de multi-grafos.

	(a) Variando $\|V\|$ de 1000 a 5000 con $\|D\| = 5000$ y $\|E\| = 1000$.			**(b) Variando $\|E\|$ de 1000 a 5000 con $\|D\| = 5000$ y $\|V\| = 1000$.**			**(c) Variando $\|D\|$ de 5000 a 25000 con $\|E\| = 1000$ y $\|V\| = 1000$.**		
Method	**Colección**	**M2Simple**	**S2Multi**	**Colección**	**M2Simple**	**S2Multi**	**Colección**	**M2Simple**	**S2Multi**
onlyMulti	*D5kV1kE1k*	48.68	19.35	*D5kV1kE1k*	48.68	19.35	*D5kV1kE1k*	48.68	19.35
	D5kV2kE1k	54.30	24.37	*D5kV1kE2k*	178.59	42.54	*D10kV1kE1k*	101.13	38.60
	D5kV3kE1k	57.71	29.19	*D5kV1kE3k*	404.70	86.05	*D15kV1kE1k*	149.24	57.78
	D5kV4kE1k	64.03	35.44	*D5kV1kE4k*	731.06	153.36	*D20kV1kE1k*	196.53	78.48
	D5kV5kE1k	68.80	40.80	*D5kV1kE5k*	1176.00	250.40	*D25kV1kE1k*	257.92	89.28
allEdges	*D5kV1kE1k*	15.70	202.39	*D5kV1kE1k*	15.84	203.20	*D5kV1kE1k*	15.29	201.42
	D5kV1kE1k	21.06	201.91	*D5kV1kE2k*	25.89	937.97	*D10kV1kE1k*	31.70	404.00
	D5kV3kE1k	25.28	210.15	*D5kV1kE3k*	36.19	1025.17	*D15kV1kE1k*	46.05	607.67
	D5kV4kE1k	30.46	215.12	*D5kV1kE4k*	45.73	1843.44	*D20kV1kE1k*	63.08	810.82
	D5kV5kE1k	35.94	227.56	*D5kV1kE5k*	57.83	2602.30	*D25kV1kE1k*	77.74	1014.43

ocurrencias en estas, ni viceversa.

Adicionalmente, se presenta un experimento sobre varias colecciones sintéticas de multi-grafos con el objetivo de evaluar la eficiencia de los métodos basados en transformaciones. Para esto se construyeron tres tipos de colecciones de multi-grafos utilizando el emulador de grafos sintético PyGen[1]. Estas colecciones fueron generadas variando un parámetro (i.e. cantidad de multi-grafos de la colección, cantidad promedio de vértices y aristas por multi-grafo) a la vez. Primero, se fijan la cantidad de multi-grafos de la colección $|D| = 5000$ y la cantidad promedio de aristas por multi-grafo $|E| = 1000$, variando la cantidad promedio de vértices por multi-grafo $|V|$ de 1000 a 5000 con incremento de 1000. Luego, se fijan $|V| = 1000$ y $|D| = 5000$, variando $|E|$ de 1000 a 5000 con incremento de 1000. Finalmente, se varía $|D|$ de 5000 a 25000 con incremento de 5000, manteniendo $|V| = |E| = 1000$.

En la tabla 1 se muestra el comportamiento, en términos de tiempo de ejecución, de ambos algoritmos encargados del proceso de transformación (M2Simple y S2Multi). Los resultados resumidos en esta tabla se alcanzan transformando cada colección de multi-grafos en una colección de grafos simples utilizando el algoritmo M2Simple de ambos métodos (onlyMulti y allEdges). Luego, cada colección de grafos simples se transforma en una colección de mutli-grafos utilizando el algoritmo S2Multi. La tabla 1 está dividida en tres según el tipo de colección. En la primera columna de esta tabla se muestra el método utilizado, en la segunda, quinta y novena se muestran las colecciones y en el resto de las columnas se muestran los tiempos de ejecución en segundos de los algoritmos encargados del proceso de transformación para ambos métodos.

Como se muestra en la tabla 1, ambos métodos basados en transformaciones de grafos tienen tiempos de ejecución que no sobrepasan los 45 minutos en colecciones de hasta 50000 multi-grafos. Estos tiempos de ejecución aumentan a medida que se incrementan las variables $|D|$, $|V|$ y $|E|$, haciendo más costoso el proceso el aumento de $|E|$. En esta tabla se puede observar que el método allEdges necesita más tiempo sobre las

[1]PyGen accesible en http://pywebgraph.sourceforge.net.

mismas colecciones de multi-grafos que el método onlyMulti. Esto se debe a que el primero transforma todas las aristas (ya sean lazos, multi-aristas o aristas simples) incrementando considerablemente la complejidad de la colección, mientras que el segundo solamente modifica los lazos y las multi-aristas. Esto permite que el algoritmo S2Multi del método onlyMulti procese menor cantidad de aristas que el mismo algoritmo para el método allEdges. De manera diferente ocurre con el algoritmo M2Simple, el cual necesita más tiempo para el método onlyMulti debido a que debe recorrer todas las aristas identificando las múltiples y los lazos, mientras que para allEdges se modifican todas sin realizar análisis alguno de las aristas. Es importante señalar que el tiempo de transformación es despreciable frente a los tiempos necesarios para la ejecución de un algoritmo para la minería. Esto se debe a que la complejidad de los algoritmos para la minería es exponencial respecto a la cantidad de patrones que se identifican en la colección.

Finalmente, la utilidad de los patrones identificados pueden ser consultados en (Acosta-Mendoza et al., 2015b) y (Acosta-Mendoza et al., 2015a). En estos trabajos se muestra la utilidad de este tipo de patrones en tareas de clasificación de imágenes.

Conclusiones

En este artículo se realizó una descripción de los métodos basados en transformaciones de grafos, reportados en la literatura, para lograr que los algoritmos tradicionales para la minería de SFAs se puedan aplicar en colecciones de multi-grafos. Estos métodos transforman multi-grafos en grafos simples mediante la adición de vértices y aristas especiales. Teniendo en cuenta sus características, se llegó a la conclusión de que, aunque en el peor de los casos ambos métodos tienen la misma complejidad computacional, el método onlyMulti es menos complejo que allEdges en la mayoría de los casos. Esto lo hace más viable en aplicaciones reales, aunque habría que analizar los patrones que se obtienen para inclinarse por uno específico. Adicionalmente, evaluó la eficiencia de los algoritmos encargados de realizar las transformaciones de grafos que confeccionan a dichos métodos reportados sobre diferentes colecciones sintéticas de multi-grafos. Basados en estos experimentos se puede concluir que los tiempos necesarios para llevar a cabo las transformaciones de los multi-grafos en grafos simples y viceversa son pequeños, siendo menores de 45 minutos en el peor de los casos, procesando colecciones de hasta 50000 multi-grafos.

Como trabajo futuro se propondrán algoritmos que realicen la minería directamente en colecciones de multi-grafos. De esta manera se eliminará la complejidad adicional introducida por los métodos basados en transformaciones de grafos debido al incremento de vértices y aristas que se le realiza al conjunto de datos originales.

Referencias

N. Acosta-Mendoza, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, A. Gago-Alonso, and J.E. Medina-Pagola. A New Method Based on Graph Transformation for FAS Mining in Multi-graph Collections. Acepted. In *Proceedings of the 7th Mexican Conference on Pattern Recognition (MCPR)*. LNCS, 2015a.

N. Acosta-Mendoza, A. Gago-Alonso, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, and J.E. Medina-Pagola. A New Method for Transforming Multi-graphs into Simple-Graphs and vice versa for Frequent Approximate Subgraph Mining in Multi-graph Collections. *Submited to Information Sciences of ELSEVIER*, 2015b.

C. Bothorel, J.D. Cruz, M. Magnani, and B. Micenkova. Clustering attributed graphs: models, measures and methods. *arXiv preprint arXiv:1501.01676*, 2015.

H. Hu, L. Wu, C. Yang, and H. Song. Interactive multigraph visualization and exploration with a two-phase strategy. *Systems Engineering and Electronics, Journal of*, 25(5):886–894, 2014.

A. Morales-González, N. Acosta-Mendoza, A. Gago-Alonso, E.B. García-Reyes, and J.E. Medina-Pagola. A new proposal for graph-based image classification using frequent approximate subgraphs. *Pattern Recognition*, 47(1):169–177, 2014. ISSN 0031-3203.

J. Patel and B.A. Oza. Survey on Graph Pattern Mining Approach. *International Journal of Engineering Development and Research*, 2(1):5, 2014.

M. Setak, M. Habibi, H. Karimi, and M. Abedzadeh. A time-dependent vehicle routing problem in multigraph with FIFO property. *Journal of Manufacturing Systems*, 35:37–45, 2015.