

Aplicación de un algoritmo de reducción de grafos al Método de los Grafos Dicromáticos

Applying a graph reduction algorithm to Dichromatic Graphs Method

Rafael Rodríguez-Puente^I, Sergio A. Marrero-Osorio^{II}, Manuel S. Lazo-Cortés^I

I Universidad de las Ciencias Informáticas. Facultad 3. La Habana, Cuba.

Correo electrónico: rafaelrp@uci.cu

II Instituto Superior Politécnico José Antonio Echeverría. Facultad de Ingeniería Mecánica. La Habana. Cuba

Recibido: 9 de diciembre de 2011 Aceptado: 2 de abril de 2012

Resumen

El presente artículo propone una vía para la descomposición de grafos representativos de modelos matemáticos, de tal manera que se conservan las relaciones entre los vértices del grafo original. Para ello se definen las relaciones de equivalencia y las particiones necesarias para la aplicación de un algoritmo de reducción de grafos a un grafo obtenido a partir de la aplicación del *Método de los Grafos Dicromáticos*, dicho método ha sido utilizado en función del diseño racional y de la solución de problemas computacionales en ingeniería mecánica. La propuesta constituye un aporte para el análisis de modelos matemáticos de grandes dimensiones, así como para facilitar la automatización del mencionado método.

Palabras claves: reducción de grafos, diseño racional, método de los grafos dicromáticos, solución de problemas computacionales.

Abstract

This article proposes a mechanism for the decomposition of graphs representing mathematical models, in a way that preserves the relations between the vertices of the original graph. For this, we define the equivalence relations and partitions necessary for the application of a graph reduction algorithm to a graph obtained from the application of Dichromatic Graph Method, this method has used in terms of rational design and computational problem solving in mechanical engineering. This proposal is a contribution to the analysis of mathematical models of large dimensions and facilitates the automation of that method.

Key words: graph reduction, rational design, dichromatic graphs method, computational problem solving.

Introducción

La teoría de grafos ha sido ampliamente estudiada y hoy día es aplicada en la solución de un sin número de problemas en diferentes ramas de la ciencia y la técnica. Entre estas aplicaciones se pueden mencionar la solución de problemas de camino óptimo, de flujo máximo de costo mínimo, la representación y análisis de redes complejas (redes sociales), y muchas otras.

Con el fin de obtener algoritmos para resolver problemas cuya solución es un conjunto de magnitudes, es decir, problemas de cómputo, Martínez Escanaverino comenzó a publicar desde 1997 lo que denominó el *Método de los Grafos Dicromáticos* (MGD) [1-3]. El MGD representa la estructura de modelos matemáticos, situaciones, problemas y algoritmos, mediante grafos dicromáticos que se van transformando sucesivamente. Ha sido aplicado por varios autores según se puede ver en tesis doctorales [4,5] y publicaciones [6, 7], en las cuales se demuestra que este significa un paso de avance desde el punto de vista práctico en la obtención de algoritmos basados en modelos matemáticos.

A partir de la descripción del MGD se puede apreciar que, de forma general, el método consiste en la ejecución de varias transformaciones de grafos. Por otra parte, atendiendo a la cantidad de vértices y aristas del grafo obtenido en cada paso, el MGD se puede ver como una serie de reducciones de grafos donde cada reducción descarta información que no es necesaria para el paso siguiente.

En la práctica los ingenieros e investigadores, durante el proceso de diseño, dividen los modelos en varios sub-modelos. Esto se hace debido a que, en no pocas ocasiones, el modelo obtenido alcanza dimensiones inmanejables desde el punto de vista práctico. Como ejemplos de lo anterior se pueden citar los trabajos realizados por Llamas [8], en el cuál se divide el modelo matemático en seis sub-modelos; y por Alemán, quien descompone un modelo matemático matricial en 15 sub-modelos [9]. Es válido mencionar que estas divisiones son realizadas manualmente, por lo que su aplicación no está generalizada debido a la dificultad que representa en la práctica. La representación de estos modelos puede realizarse de acuerdo al MGD y en ese caso, es posible depurar el grafo obtenido de diferentes maneras. Por ejemplo, en un caso podría interesar exclusivamente el modelado riguroso de la geometría de uno de los mecanismos de una máquina, sin considerar la cinemática, la dinámica, las tensiones o los costos. El modelo es entonces restringido a una parte del modelo matemático general. A esta forma de simplificación se le nombrará reducción por partes.

A pesar de que existen herramientas de edición de grafos que facilitan en cierto grado la aplicación del MGD [10] y la reducción por partes, no se cuenta con un sistema informático para aplicar el mismo, lo que limita en cierta medida su valor práctico.

La representación de los modelos matemáticos de ingeniería puede generar grafos de gran complejidad, lo que lleva a la necesidad de acudir a herramientas de reducción de grafos para su análisis y manipulación.

En estudios publicados se pueden encontrar varios algoritmos de reducción de grafos, muchos de los cuales se basan en eliminar aristas y/o vértices que no son necesarios para dar solución a determinados problemas y se puede encontrar aplicaciones en las siguientes temáticas:

- Redes de *workflow*, modelo que describe la estructura de un proceso de una organización. Las partes de los procesos que son ejecutadas en la computadora constituyen el modelo de *workflow* [11]. En la literatura se utilizan las redes de Petri para representar redes de *workflow* [12]. Este modelo presenta algunas deficiencias que son resueltas a través de la reducción de grafos, como se muestra en [13-15].
- Redes de computadoras, sistemas de cómputo distribuido que han recibido una alta atención. Uno de los problemas más estudiados es el problema de consenso, en [16] se muestra una revisión de los algoritmos que permiten resolver este problema. En [17] se propone una solución basada en la reducción de grafos que consiste en el análisis de sub-grafos escogidos convenientemente.
- También se ha utilizado la reducción de grafos en la solución de problemas de cómputo distribuido, cómputo paralelo, búsqueda de caminos mínimos, reducción de orden parcial, *model checking*, etc. [18-20]. Otros trabajos se refieren a reducir grafos que cumplen determinadas propiedades, para lo cual realizan la reducción encontrando determinadas sub-estructuras geométricas [21] o atendiendo a características del grafo como diámetro, corte mínimo, número y clasificación de cliques, etc. [22]. En el caso del análisis de circuitos se ha demostrado que los métodos clásicos no son suficientes [23] y se ha utilizado la reducción de grafos [24] para resolver algunos de los problemas que se presentan en esta temática.

Los algoritmos antes mencionados obtienen un grafo reducido con información suficiente para dar solución a un problema específico, pero limitada para otros problemas ya que presentan pérdida de información.

Teniendo en cuenta lo anterior, la reducción de grafos consiste en obtener un grafo de menor tamaño (menos aristas y/ o vértices) con las características principales o relevantes del grafo original, de forma tal que se puedan realizar análisis sobre el grafo reducido y llegar a conclusiones sobre el grafo original.

Resulta de particular interés para esta investigación el algoritmo de reducción enunciado en [25], debido a que el mismo garantiza que no haya pérdida de información a través de la incorporación de reglas de reescritura de grafos [26,27] al algoritmo de reducción enunciado.

El objetivo principal del presente artículo es mostrar la aplicabilidad de un algoritmo de reducción de grafos sin pérdida de información en la ejecución del MGD; además de utilizar dicho algoritmo para dividir un modelo en sub-modelos de forma tal que facilite el análisis de modelos complejos a los ingenieros.

Los resultados obtenidos muestran que es viable aplicar tres de los pasos del MGD haciendo uso de un algoritmo de reducción de grafos y que mediante el proceso de reducción se puede contribuir al análisis, por parte de los ingenieros, de modelos matemáticos complejos.

Métodos

A continuación se enuncian notaciones y definiciones tomadas de [28] necesarias para la comprensión de los resultados presentados:

Definición 1: Se denomina partición del conjunto A, a una clase P de subconjuntos de A que satisface las siguientes condiciones:

- $\emptyset \notin P$
- Dados dos conjuntos A_i y A_j de P, con $i \neq j$, se cumple que $A_i \cap A_j = \emptyset$
- La unión de todos los subconjuntos de P es igual al conjunto A

Definición 2: Sean los conjuntos A y B no necesariamente distintos, se dice que R es una relación binaria de A en B si R es un subconjunto de pares ordenados $A \times B$, es decir si $R \subseteq A \times B$.

Si $A = B$, entonces se dice que R es una relación binaria en A.

Definición 3: Una relación binaria R en un conjunto A es una relación de equivalencia si y sólo si R es reflexiva, simétrica y transitiva.

Definición 4: Sea R una relación de equivalencia entre los elementos de un conjunto A y sea $a \in A$, se denomina clase de equivalencia por R de a y se denota mediante $[a]$ al conjunto de las x, $x \in A$, que mantienen con a la relación R, es decir. $[a] = \{x | (a, x) \in R\}$.

Definición 5: Sea un grafo $G = (V, E)$ y una relación de equivalencia R sobre V, v_i es interior si $\forall v_j \in V$, tal que v_i y v_j son adyacentes se cumple que $R(v_i, v_j)$ ($[v_i] = [v_j]$). En otro caso v_i es exterior.

Definición 6: Se denomina camino desde el vértice v_i al vértice v_j en un grafo $G = (V, E)$ a la secuencia de vértices $Ca = v_1, v_2, \dots, v_n$, si $\exists (v_1, v_2), (v_2, v_3), \dots, (v_{n-1}, v_n) \in E, v_i = v_1, v_j = v_n$.

A partir de las definiciones anteriores, el autor enuncia la siguiente definición de grafo reducido:

Definición 7: Sean G y G_r dos grafos y R_r un conjunto de reglas de reescritura de grafos, se dice que G_r es un grafo reducido a partir de G, si al aplicar las reglas de reescritura especificadas en R_r al grafo G_r se obtiene el grafo G.

Breve descripción del Método de los Grafos Dicromáticos

En la Figura 1 se puede apreciar el esquema de ejecución del MGD. Inicialmente se construye el *grafo del modelo*, cuyos vértices representan las variables con un color y las ecuaciones (relaciones) con otro y las aristas simbolizan cuáles variables se encuentran en cada ecuación. Una vez planteado un problema de cómputo determinado, se descartan de dicho grafo las variables de entrada (datos), quedando transformado así en el *grafo de la situación*. A partir de este, suprimiendo las componentes conexas (islas) que no contengan variables de salida, se obtiene el *grafo del problema*. Este último grafo es sometido a un pareo que orienta, hacia su correspondiente variable, a una sola de las aristas conectadas con cada ecuación, de modo que el mismo sea un pareo máximo. Así pasa a obtenerse una nueva representación llamada *grafo del problema pareado*. Posteriormente se obtiene el *grafo del resolvente*, asignándole orientación, de variable a ecuación, a las aristas que aún no la tienen. Para los problemas que no pueden ser caracterizados plenamente, a partir de este grafo del resolvente es necesario obtener un nuevo grafo llamado *problema canónico* (si el problema tiene un pareo perfecto ya se encuentra en su forma canónica) y de este se llega al resolvente definitivo. A partir de dicho resolvente (que ya se encuentra en su forma canónica) se alcanza el *grafo del algoritmo* descartando los caminos que no conducen a ninguna de las variables de salida definidas

al plantear el problema. A través de éste método, siguiendo los pasos indicados en el grafo del algoritmo, se puede dar solución al problema inicialmente planteado. A pesar de que existen herramientas de edición de grafos que facilitan en cierto grado la aplicación del MGD, no se cuenta con un sistema informático para aplicar el mismo, lo que limita en cierta medida su valor práctico.

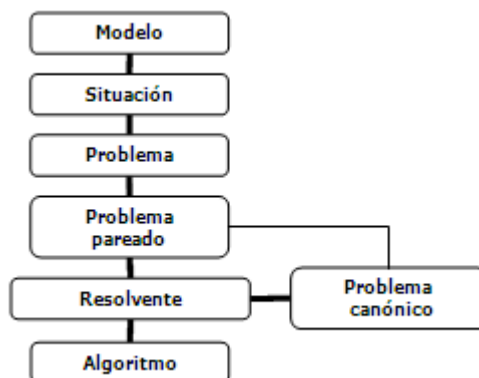


Fig. 1. Esquema del Método de los Grafos Dicromáticos

Algoritmo de reducción

A continuación se describe el algoritmo de reducción enunciado en [25], el mismo se utilizará para realizar algunas de las transformaciones definidas en el MGD.

Algoritmo 1: Reducción de grafos

Entrada: un grafo ponderado reducido $G = (V, E, f, R)$, donde R es un conjunto de reglas de reescritura, posiblemente vacío. Una relación de equivalencia RE (que permite crear una partición P en V) o una partición P en V .

Salida: un grafo ponderado reducido que representa el mapa

1. Si la entrada es una relación de equivalencia
2. $P = \text{ObtenerParticion}(V, RE)$
3. Fin Si
4. $\text{ConstruirVerticesReducidos}(P)$
5. $\text{ConstruirAristas}(P, E)$
6. $\text{ConstruirRr}(P, G)$
7. Crear el grafo reducido $G_r = (V_r, E_r, f_r, R_r)$
8. Para todo $A_i \in P, |A_i| > 1$
9. $\text{Calcular } f(G, R_r[A_i], G_i, R_r[A_i], G_j, f_r)$
10. Fin Para
11. Retornar G_r

A continuación se explican los pasos principales del Algoritmo 1.

Paso 1. Estructura condicional.

Si la entrada al algoritmo es una relación de equivalencia, se procede a ejecutar el paso 2 para crear una partición del conjunto de vértices del grafo, en caso contrario se pasa al paso 4.

Paso 2. Crear una partición a partir de una relación de equivalencia.

En el caso que la entrada sea una relación de equivalencia, se puede seguir una estrategia voraz para obtener la partición. La estrategia consistiría en comparar cada elemento del conjunto A con los demás para determinar cuáles elementos están relacionados y a partir de estos crear las distintas clases de equivalencia que forman la partición.

Paso 3. Fin de la estructura condicional.

Este paso significa el cierre de la estructura condicional que comienza en el paso 1.

Paso 4. Construir el conjunto de vértices del grafo reducido.

En este paso se crea un vértice por cada clase de equivalencia de la partición P .

Paso 5. Construir el conjunto de aristas.

En este paso del algoritmo se construye el conjunto de aristas E_r , en el mismo se debe tener en cuenta que para adicionar una arista al grafo reducido que se está construyendo, los dos vértices que forman parte de dicha arista deben pertenecer a clases de equivalencia distintas, es decir $(v_{rm}, v_{rk}) \in E_r$ sólo si $v_{rm} \in A_i, v_{rk} \in A_j, i \neq j$, además se debe comprobar que exista una arista en el grafo original que una dos vértices v_m y v_k tal que $v_m \in A_i$ y $v_k \in A_j$.

Si se puede adicionar al grafo reducido más de una arista entre un mismo par de vértices, en el caso del análisis de redes, se adiciona la de menor costo. Sin embargo, para otro tipo de aplicaciones, dónde la importancia radica en la relación y no en una medida existente a partir de la relación, se pueden adicionar varias aristas entre un mismo par de vértices.

Paso 6. Crear las reglas de reescritura.

Este es un paso esencial del algoritmo, es el que permite que no haya pérdida de información en la reducción del grafo y además garantiza que se pueda obtener el grafo original a partir del grafo reducido.

Como se define en [29], una regla de reescritura se expresa como una tupla de la forma $(G_i, G_j, \psi_{in}, \psi_{out})$, en este paso se crea una regla de reescritura por cada clase de equivalencia $A_i, |A_i| > 1$ donde:

- $G_i = (\{v_i\}, \{\}), A_i = [v_i]$.
- $G_j = (A_i, E_i, f_{cj})$ es un subgrafo de $G = (V, E, f_c, R)$, donde $\exists(u, v) \in E_i$ si y sólo si $(u, v) \in E$ y $u, v \in A_i$, además $f_{cj}(u, v) = f_c(u, v)$.
- El conjunto ψ_{in} está formado por tuplas de la forma (v_m, c_1, c_2, v_n) que deben cumplir lo siguiente: $c_1, c_2 \in \mathbb{R}, v_m \in A_i, v_n \in (V - A_i), \exists(v_n, v_m) \in E$ y $f_{cj}(v_n, v_m) = c_1$; además el costo de la arista $(v_n, v_i) \in E_r$ es c_2 , o sea $f_{cj}(v_n, v_i) = c_2$.
- Análogamente, el conjunto ψ_{out} está formado por tuplas de la forma (v_m, c_1, c_2, v_n) que deben cumplir lo siguiente: $c_1, c_2 \in \mathbb{R}, v_m \in A_j, v_n \in (V - A_j), \exists(v_m, v_n) \in E$ y $f_{cj}(v_m, v_n) = c_1$, además el costo de la arista $(v_i, v_n) \in E_r$ es c_2 , o sea $f_{cj}(v_i, v_n) = c_2$.

Paso 7. Crear grafo reducido.

En este paso se crea un grafo reducido con el conjunto de vértices, aristas y reglas de reescritura obtenidos en los pasos anteriores.

Pasos 8-10. Actualización de la función f .

En estos pasos del algoritmo se calcula el valor de la función f para cada vértice reducido ($A_i \in P, |A_i| > 1$) del grafo creado en el paso anterior. Esta función representa el costo de ir desde un vértice el grafo hasta otro (puede ser otro atributo de la arista). En grafos obtenidos con el MGD no se representa este valor, por lo que estos pasos no son indispensables para la aplicación de este algoritmo al MGD.

Paso 11. Retornar el grafo obtenido.

En este paso el algoritmo devuelve el grafo reducido que se construyó partir de la relación de equivalencia (o la partición) de entrada.

Caso de estudio

Para la descripción de los resultados de la presente investigación se utilizó como caso de estudio el grafo del modelo de unas pinzas de fricción cuyo proceso de obtención se describe en [7]. Además, se tuvo en cuenta las siguientes operaciones enunciadas en el MGD:

- Descartar vértices que representen variables de entrada (datos).
- Identificar componentes conexas (islas) y descartar las que no tengan variables de salida.
- Descartar caminos que no contengan variables de interés para la solución.

Por otro lado, se realizó el proceso de reducción por partes, utilizado comúnmente en el Diseño Racional en Ingeniería. Para ello se utilizaron los siguientes sub-modelos, presentes en el modelo de las pinzas de fricción referido anteriormente:

- Parámetros del modelo geométrico
- Cargas y reacciones
- Masas
- Resistencia Mecánica
- Operaciones tecnológicas
- Costo aproximado de producción
- Tiempo de fabricación

Para llevar a cabo las transformaciones antes mencionadas, se utilizó el Algoritmo 1, por la característica de no tener pérdida de información. Para el uso de este algoritmo es necesario formalizar las reducciones antes mencionadas haciendo uso de relaciones de equivalencia o particiones como se muestra en el siguiente epígrafe. Para una explicación detallada sobre estos conceptos el lector puede remitirse a [30].

Para definir las transformaciones se asume que un vértice del grafo del modelo tiene la siguiente información:

- *Tipo_vértice*, puede tomar los valores *variable* y *ecuación*
- *Tipo_variable*, puede tomar los valores *entrada*, *incógnita* y *salida*
- *Es_de_interés*, puede tomar los valores *sí* o *no*
- *Sub_modelo*, el valor de esta variable es el nombre del sub-modelo al que pertenece el vértice

Transformaciones a realizar a partir del grafo del modelo

El Algoritmo 1 puede tener como entrada un grafo dirigido o no dirigido, para el mismo no es relevante la orientación de las aristas, esta característica es utilizada para determinar cuándo dos vértices son adyacentes, lo cual debe estar encapsulado en la biblioteca de manejo de grafos que se utilice para hacer la reducción.

Sea $G = (V, E)$, el grafo de un modelo, definido según el MGD donde:

- V es el conjunto de vértices del grafo, que representa las variables y ecuaciones del modelo matemático.
- E es el conjunto de aristas del grafo, que representa la forma en que se relacionan las variables y las ecuaciones del modelo.

A continuación se definen las particiones y relaciones de equivalencia necesarias para realizar las transformaciones propuestas:

1- Descartar los vértices que representen variables de entrada:

2- Sea $v \in V$ una variable de entrada y la clase de equivalencia $[v] = \{u | u \text{ es una variable de entrada}\}$, se define la partición $P = \{\{v_i\}_{v_i \notin [v]}, [v]\}$. O sea, para construir la partición P se crea una clase de equivalencia por cada vértice de V que no representa una variable de entrada, más una clase de equivalencia que contiene todos los vértices que representan variables de entrada. De esta forma, en el grafo resultante de aplicar este paso, habría un solo vértice que representaría variables de entrada, lo que implicaría una reducción del grafo del modelo.

3- Identificar las componentes conexas (islas) y descartar las que no tengan variables de salida:

4- Dos vértices $u, v \in V$ están conectados si y sólo si existe un camino que los une. La relación de conexión Rc definida en V por $(u, v) \in Rc$ si y solo si u y v están conectados, es una relación de equivalencia. El sub-grafo de G determinado por una clase de equivalencia de Rc es una componente conexa de G .

5- Descartar los problemas elementales que no sean necesarios para obtener las variables de salida:

6- Sea la clase de equivalencia $[v] = \{u \mid (u, v) \in R_3\}$, $(u, v) \in R_3$ si y solo si $\exists (u, v) \in E$, donde u representa una ecuación y v representa una variable en u que no tiene vértices adyacentes, se define una partición $P = \{\{v_i\}_{v_i \notin [v]}, [v]\}$.

Reducción por partes

Para realizar este tipo de reducción se define una relación de equivalencia (R_4) a partir de la igualdad en el atributo sub-modelo de los vértices del grafo, de modo que:

$$R_4 = \{ \langle u, v \rangle \mid u, v \in V \wedge u.\text{submodelo} = v.\text{submodelo} \}$$

En este caso, para realizar este tipo de reducción bastaría con invocar el algoritmo de reducción de grafos dando como entrada el grafo del modelo y la relación de equivalencia R_4 . Advierta que cuando se crea el grafo reducido, las variables que son comunes a más de un sub-modelo serán representadas como vértices exteriores, mostrando cómo se relacionan los sub-modelos de las diferentes partes del modelo original.

Resultados y discusión

A continuación se muestra una ejecución del algoritmo de reducción sobre el grafo del modelo de unas pinzas de fricción cuyo proceso de diseño se describe en [7]. En la Figura 2 se muestra el grafo del modelo del ejemplo seleccionado. A partir del mismo se realizará una reducción por partes, teniendo en cuenta los sub-modelos que componen el modelo completo.

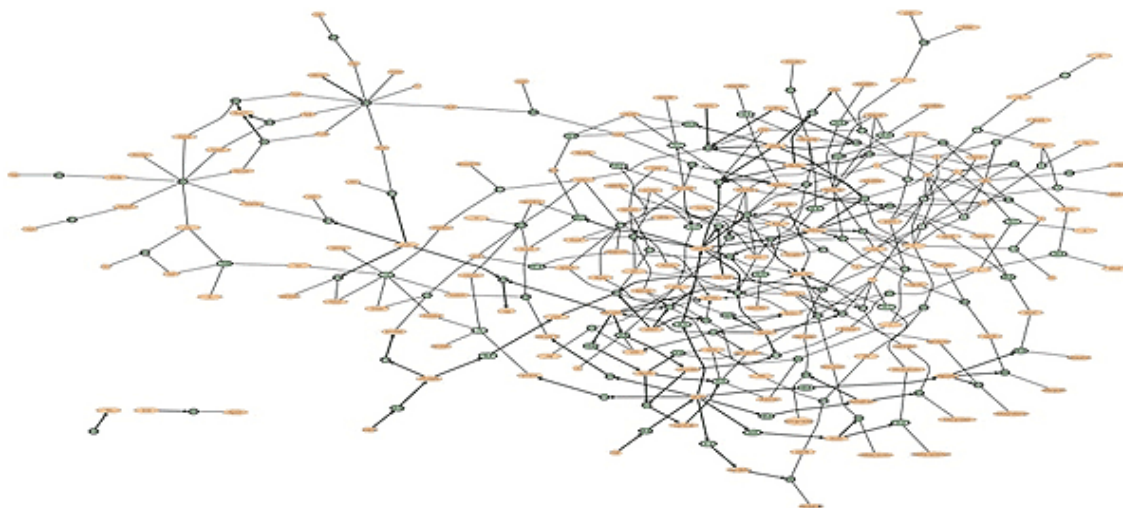


Fig. 2. Grafo del modelo. Tomado de [7]

La primera reducción se realiza según el sub-modelo Geométrico, obteniéndose un nuevo grafo mostrado en la Figura 3. Luego se reduce según los demás sub-modelos y al finalizar este proceso se obtiene el grafo de la Figura 4. En el mismo se han representado con un rectángulo azul los vértices reducidos que corresponden a un sub-modelo y de otro color los vértices que representan variables del modelo. Entre los vértices asociados a variables, se representan con un rectángulo los grupos de variables que relacionan dos sub-modelos (que se representan como un vértice reducido), para diferenciarlas de las variables del grafo del modelo (que se representan como un vértice no reducido -una elipse-). Como se puede apreciar, se enfoca el MGD como un proceso de transformación de grafos (específicamente de reducción) lo que posibilita la aplicación de un algoritmo de reducción en la automatización del mismo. A partir de esta reducción se puede afirmar la posibilidad de efectuar reducciones automatizadas del grafo del modelo o del grafo del resolvente atendiendo a criterios semánticos y estructurales del fenómeno que se modela; en lugar de reducir solamente según la estructura del grafo obtenido en los pasos del MGD. Además, con la división de un modelo en varios sub-modelos, es posible una visión más global del modelo que se está analizando; se puede apreciar, gráficamente, cuáles variables están presentes en más de un sub-modelo, así como realizar análisis a un determinado nivel de abstracción.

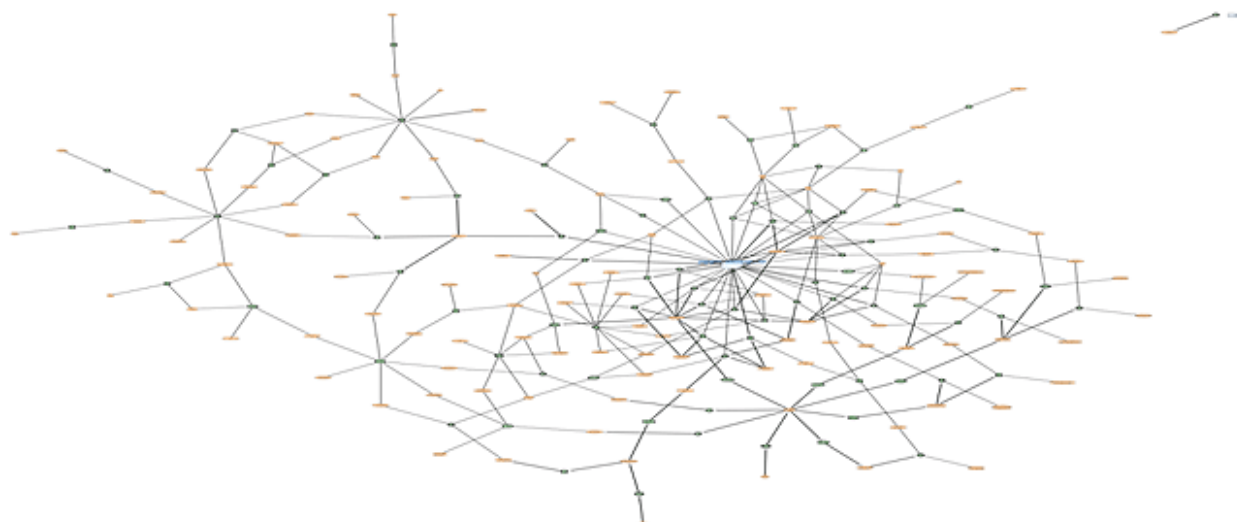


Fig. 3. Grafo reducido por Modelo Geométrico

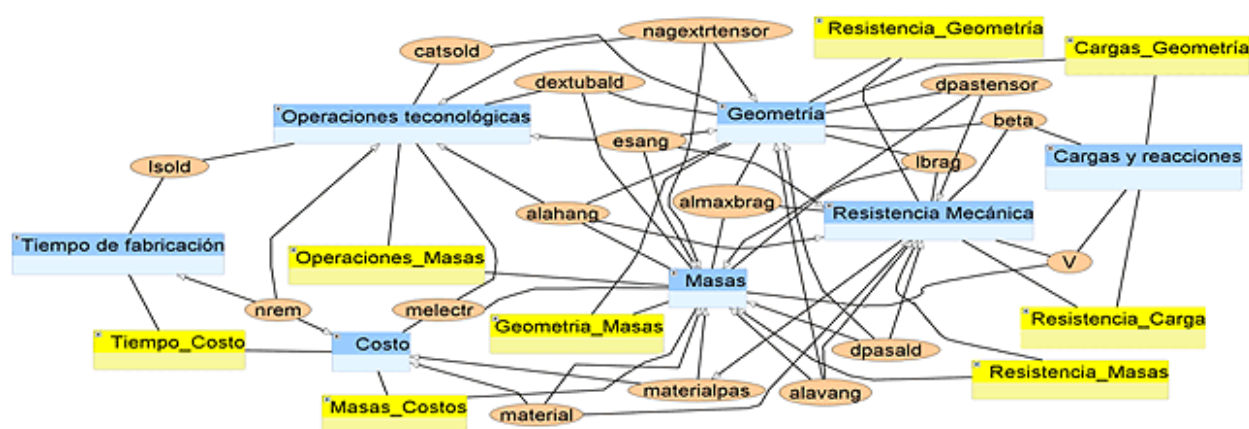


Fig. 4. Grafo reducido teniendo en cuenta todos los sub-modelos. Los rectángulos amarillos representan conjuntos de variables comunes a dos sub-modelos.

Un aspecto importante a recalcar es que el proceso aquí presentado es reversible. Esto se sostiene a partir de que el algoritmo de reducción de grafos no tiene pérdida de información y además propone un mecanismo que permite, dado un grafo reducido (con el algoritmo en cuestión), obtener cualquiera de los grafos generados durante el proceso de obtención del mismo, incluyendo el grafo original (en este caso el grafo del modelo).

En la Figura 5 se muestra un ejemplo de aplicación del algoritmo de reducción de grafos antes mencionado al MGD. En la parte izquierda de la misma se muestra el grafo obtenido al agrupar todas las variables de entrada o datos del problema (representadas por un rectángulo). Para la aplicación del paso del MGD que se refiere a descartar las variables de entrada, basta con ocultar el vértice que representa a las mismas, tal como se muestra a la derecha de la figura.

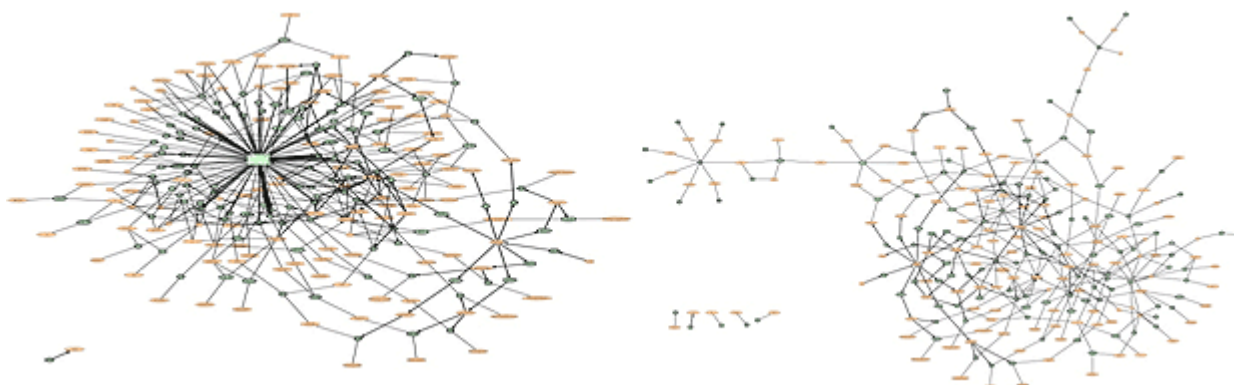


Fig. 5. Ejemplo de reducción por variables de entrada

Conclusiones

A partir de los resultados obtenidos se arribó a las siguientes conclusiones:

A través de un algoritmo de reducción de grafos es posible formalizar el proceso de reducción de un grafo que representa un modelo matemático, específicamente el obtenido a partir de la aplicación del MGD.

El algoritmo de reducción utilizado en la presente investigación contribuye a la automatización del MGD.

Haciendo uso de un algoritmo de reducción de grafos es posible reducir el grafo de un modelo matemático, obtenido haciendo uso del MGD, de acuerdo a los sub-modelos que lo conforman.

Se provee un mecanismo que permite realizar transformaciones en el grafo atendiendo a criterios semánticos.

A partir de los resultados obtenidos se justifica el estudio de la aplicación de la reducción de grafos como mecanismo de reducción de modelos matemáticos.

Referencias

1. Martínez Escanaverino, J. "Dichromatic Graphs: A Tool for the Algorithmic Education of Mechanical Engineers". En: *ASME Design Engineering Technical Conferences & Computers and Information in Engineering, DETC 2000*. Baltimore, Maryland: Amer Society of Mechanical, 2000. ISBN 978-0791835142.
2. Martínez Escanaverino, J., Llamas Soríz, J.A., García Toll, A. y Ortiz Cárdenas, T. "Rational design automation by dichromatic graphs". En: *ASME 2001 Design Engineering Technical Conferences and Computers and Information in Engineering, DETC'01*. Pittsburgh, Pennsylvania: Amer Society of Mechanical, 2001. ISBN 0-7918-3506-5.
3. Martínez Escanaverino, J. y Martínez Fonte, L. "A problem solving rationale for conceptual design in engineering". En: *Proceedings of the 6th International Conference on Computer-Aided Industrial Design & Conceptual Design - CAID&CD 2005*. Delft, Netherlands, 2005. [Consultado el: 3 de octubre de 2011]. Disponible en: <https://biblio.ugent.be/record/314454>. ISBN 7-5062-7444-2.
4. Marrero Osorio, S. A. "Diseño Paramétrico Basado en Modelos Matemáticos. Caso de estudio: Máquinas para la Construcción Sostenible de Viviendas". Tutor: J. Martínez Escanaverino. Tesis Doctoral, Departamento de Mecánica Aplicada, Facultad de Ingeniería Mecánica. Instituto Superior Politécnico José Antonio Echeverría, 2009. [Consultado el: 24 de noviembre de 2011]. Disponible en: <http://biblioteca.cujae.edu.cu/Tesis/Doctorado/Tesis185.pdf>.
5. Morejón Vizcaino, G. "Primera etapa del proceso de desarrollo de un hidromotor de alto par: conceptualización y prototipo analítico". Tutor: J. Arzola Ruiz. Tesis Doctoral, Departamento de Mecánica Aplicada, Facultad de Ingeniería Mecánica. Instituto Superior Politécnico José Antonio Echeverría, 2004. [Consultado el: 24 de noviembre de 2011]. Disponible en: <http://biblioteca.cujae.edu.cu/Tesis/Doctorado/Tesis53.pdf>.

6. González Rey, G. "Establecimiento del cálculo del diámetro de cresta exterior de un engranaje cónico con técnicas de grafos". *Ingeniería Mecánica*, 2007, vol. 6, nº. 3, p. 33-39. [Consultado el: 18 de octubre de 2011]. Disponible en: <http://redalyc.uaemex.mx/redalyc/pdf/2251/225115060005.pdf>. ISSN 1815-5944.
7. Marrero Osorio, S. A. y Martínez Escanaverino, J. "Diseño paramétrico de pinzas de fricción". *Ingeniería Mecánica*, 2009, vol. 12, nº. 1, p. 37-48. [Consultado el: 18 de octubre de 2011]. Disponible en: <http://redalyc.uaemex.mx/redalyc/pdf/2251/225114975005.pdf>. ISSN 1815-5944.
8. Llamas Soríz, J. A. "Diseño óptimo de cajas reductoras para molinos de caña de azúcar". Tutor: J. Martínez Escanaverino. Tesis Doctoral. Departamento de Mecánica Aplicada, Facultad de Ingeniería Mecánica. Instituto Superior Politécnico José Antonio Echeverría, 2000. [Consultado el: 24 de noviembre de 2011]. Disponible en: <http://biblioteca.cujae.edu.cu/Tesis/Doctorado/Tesis8.pdf>.
9. Alemán Romero, I.L. "Modelo Matemático y Algoritmos para la Aplicación 3D del Método de los Elementos de Contorno". Tutor: L.L. Otero Pereiro. Tesis Doctoral, Departamento de Matemática, Facultad de Ingeniería Mecánica. Instituto Superior Politécnico José Antonio Echeverría, 2003. [Consultado el: 24 de noviembre de 2011]. Disponible en: <http://biblioteca.cujae.edu.cu/Tesis/Doctorado/Tesis49.pdf>.
10. Marrero-Osorio, S. A. "Sobre la solución de problemas computacionales en ingeniería mediante grafos dicromáticos. Selección de software de apoyo". *Ingeniería Mecánica*, 2011, vol. 14, no. 2, p. 160-169. [Consultado el: 18 de octubre de 2011]. Disponible en: <http://redalyc.uaemex.mx/src/inicio/ArtPdfRed.jsp?iCve=225117950008>. ISSN 1815-5944.
11. Solana González, P., Alonso Martínez, M. y Pérez González, D. "Análisis y modelado de redes de workflow del proceso de tratamiento de experiencias operativas". En: *AEDEM. XX Congreso anual de AEDEM*. Palma de Mallorca, España, 2007, vol. 1, p. 1109-1122. [Consultado el: 30 de septiembre de 2011]. Disponible en: <http://biblioteca.universia.net/irARecurso.do?page=&id=30890290>.
12. Aalst, W. M. P. "Verification of Workflow Nets". En: *Proceedings of the 18th International Conference on Application and Theory of Petri Nets*. Toulouse, France: Springer-Verlag, 1997, p. 407-426. [Consultado el: 30 de septiembre de 2011]. Disponible en: <http://dl.acm.org/citation.cfm?id=733919>. ISBN 3-540-63139-9.
13. Sadiq, W. y Orlowska, M. E. "Analyzing process models using graph reduction techniques". En: *Information Systems - The 11th international conference on advanced information systems engineering*, vol. 25, no. 2, p. 117-134. 2000 [Consultado el: 30 de septiembre de 2011]. Disponible en: <http://dl.acm.org/citation.cfm?id=344369>. ISSN 0306-4379.
14. Kai, L. y Qiang, L. "An Algorithm Combining Graph-Reduction and Graph-Search for Workflow Graphs Verification". En: *11th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2007*. Melbourne, Australia: Springer, 2007, p. 772-776. [Consultado el: 30 de septiembre de 2011]. Disponible en: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4281534. ISBN 978-3540927181.
15. Liu, Q., Cao, B. y Zhao, Y. "An improved verification method for workflow model based on Petri net reduction". En: *The 2nd IEEE International Conference on Information Management and Engineering (ICIME)*. Chengdu, Sichuan, China: IEEE Computer Society Press, p. 252-256. 2010. [Consultado el: 30 de septiembre de 2011]. Disponible en: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5477436. ISBN 978-1-4244-5263-7.
16. Olfati-Saber, R., Fax, J.A. y Murray, R. M. "Consensus and Cooperation in Networked Multi-Agent Systems". *Proceedings of the IEEE*, 2007, vol. 95, nº. 1, p. 215-233. [Consultado el: 30 de septiembre de 2011]. Disponible en: <http://dx.doi.org/10.1109/JPROC.2006.887293>. ISSN 0018-9219.
17. Parlangeli, G. y Notarstefano, G. "Graph reduction based observability conditions for network systems running average consensus algorithms". En: *18th Mediterranean Conference on Control & Automation (MED)*. Marrakech, Morocco: IEEE Computer Society, 2010, p. 689-694. [Consultado el: 30 de septiembre de 2011]. Disponible en: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5547789. ISBN 978-1-4244-8091-3

18. Fuhao, Z. y Liu, J. "An Algorithm of Shortest Path Based on Dijkstra for Huge Data". En: *Fourth International Conference on Fuzzy Systems and Knowledge Discovery*, 2009, vol. 4, p. 244--247. [Consultado el: 5 de septiembre de 2011].
Disponible en: <http://doi.ieeecomputersociety.org/10.1109/FSKD.2009.848> . ISSN 978-0-7695-3735-1.
19. Li, Y. L., Li, J. Y. y Chen, W. B. "An Efficient Tile-Based ECO Router Using Routing Graph Reduction and Enhanced Global Routing Flow". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2007, vol. 26, nº. 2, p. 345-358. [Consultado el: 30 de septiembre de 2011].
Disponible en: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4068927 . ISSN 0278-0070.
20. Barnat, J., Brim, L. y Rockai, P. "Parallel Partial Order Reduction with Topological Sort Proviso". En: *8th IEEE International Conference on Software Engineering and Formal Methods (SEFM)*. Pisa, Italy: IEEE Computer Society, 2010, p. 222 -231. [Consultado el: 30 de septiembre de 2011].
Disponible en: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5637433. ISBN 978-1-4244-8289-4.
21. Rinehart, M. y Dahleh, M.A. "A graph reduction for bounding the value of side information in shortest path optimization". *American Control Conference (ACC)*, 2010, p. 4078 -4083. [Consultado el: 30 de septiembre de 2011].
Disponible en: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5530627 . ISSN 0743-1619.
22. Diestel, R. *Graph theory*. Third Ed. Hamburg, Germany: Springer, 2006. ISBN 9783540261834.
23. Lin, P. "A survey of applications of symbolic network functions". *IEEE Transactions on Circuit Theory*, 1973, vol. 20, no. 6, p. 732-737. [Consultado el: 30 de septiembre de 2011].
Disponible en: <http://dx.doi.org/10.1109/TCT.1973.1083770>. ISSN 0018-9324
24. Shi, G., Chen, W. y Shi, C.J. R. "A Graph Reduction Approach to Symbolic Circuit Analysis". En: *Proceedings of the 2007 Asia and South Pacific Design Automation Conference*. Yokohama, Japan: IEEE Computer Society, 2007, p. 197-202. [Consultado el: 30 de septiembre de 2011].
Disponible en: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4196031 . ISBN 1-4244-0630-7
25. Rodríguez Puente, R. "Aplicación de las gramáticas de grafo en Sistemas de Información Geográfica". *Revista Cubana de Ciencias Informáticas*, 2010, vol. 4, nº. 1/2, p. 5-10. [Consultado el: 29 de noviembre de 2011]. Disponible en: <http://rcci.uci.cu/index.php/rcci/article/view/108> . ISSN 1994-1536
26. Geiß, R. y Kroll, M. "GrGen.NET: A Fast, Expressive, and General Purpose Graph Rewrite Tool". En: *Applications of Graph Transformations with Industrial Relevance*. Kassel, Germany: Springer Heidelberg, 2008, vol. 5088, p. 568-569. [Consultado el: 30 de septiembre de 2011].
Disponible en: http://dx.doi.org/10.1007/978-3-540-89020-1_38 . ISBN 978-3-540-89019-5.
27. Balasubramanian, D., Narayanan, A., van Buskirk, C. P. et al. "The Graph Rewriting and Transformation Language: GReAT". *ECEASST*, 2006, vol. 1. [Consultado el: 30 de septiembre de 2011].
Disponible en: <http://eceasst.cs.tu-berlin.de/index.php/eceasst/article/view/89> . ISSN 1863-2122.
28. García Garrido, L. *Introducción a la Teoría de Conjuntos y a la Lógica Matemática*. Ciudad de La Habana: Editorial Felix Varela, 2002.
29. Janssens, D. y Rozenberg, G. "Graph grammars with neighbourhood-controlled embedding". *Theoretical Computer Science*, 1982, vol. 21, nº. 1, p. 55-74. [Consultado el: 30 de septiembre de 2011].
Disponible en: <http://www.sciencedirect.com/science/article/B6V1G-45GMVNY-11/2/37148cad699b9f0964a6dadc031d178e> . ISSN 0304-3975.
30. Mazzola, G., Milmeister, G. y Weissmann, J. *Comprehensive Mathematics for Computer Scientists 1*. Second Ed. Zurich, Switzerland: Springer-Verlag, 2006. ISBN 3-540-36873-6.