Diseño y programación de un software de transformación de matrices para el análisis de redes

REDES Revista Hispana para el Análisis de Redes Sociales

Vol.27,#1,(2016), 73-80

http://revistes.uab.es/redes http://dx.doi.org/10.5565/rev/redes.602

Diseño y programación de un software de transformación de matrices para el análisis de redes

Enrique Marcet García*
Instituto Superior Politécnico José A. Echeverría (CUAJE)

Miriam Palacios-Callender
School of Computing and Technology, University of West London

Marcelo Marcet Sánchez
Universidad de Matanzas

RESUMEN

Para muchos profesionales vinculados al Análisis de Redes Sociales (ARS), es de importancia transformar sus datos para realizar estudios posteriores. Es frecuente que los datos de sistemas donde intervienen conexiones se almacenen en matrices más elementales que las matrices completas o "*full matrix",* frecuentemente utilizadas en programas de ARS. El objetivo principal de este trabajo fue diseñar y programar un software modificable para transformar matrices de tres columnas [A (grupo 1), B (grupo 2) y C (conexiones entre nodos de ambos grupos)] a matrices completas y simétricas, y analizar las ventajas del software con respecto a los problemas de almacenamiento y transformación de datos. Para ello se desarrolló un método de transformación de matrices y se diseñó un algoritmo (pseudocódigo) para realizar las operaciones necesarias. El algoritmo fue programado usando el lenguaje wxBasic para aumentar la compatibilidad entre sistemas operativos y facilitar su modificación por el usuario, obteniéndose un software estable y de fácil uso.

Palabras clave: *Transformación de matriz – algoritmo – software - matrices simétricas.*

ABSTRACT

For many professionals linked to Social Network Analysis (SNA), it is important to transform their data for further studies. Often, the data of systems where exist ties are stored in elementary matrices, and not in full matrices, frequently used in SNA software. The objective of this paper was to design and program a modifiable software with the potential to transform three columns matrices [A (group 1), B (group 2) and C (ties between groups nodes)] to full and symmetric matrices, and to analyze the advantages of software regarding storage and data processing problems. To perform the necessary operations was developed a transformation matrix method and an algorithm (pseudocode) of this method. The algorithm was programmed in wxBasic to improve the compatibility between operating systems and to make the user modification easier, obtaining stable and easy-to-use software.

Key words: *Matrix transformation – algorithm - software – symmetric matrices.*

**Contacto con los autores: Enrique Marcet García (marcet.enrike@gmail.com), Miriam Palacios Callender (Miriam.Palacios-Callender@uwl.ac.uk), Marcelo Marcet Sánchez (marcelo.marcet@umcc.cu)*

Actualmente, hay estudios muy avanzados sobre Análisis de Redes Sociales (ARS). Se ha convertido en una temática de estudio, análisis y especulación; parte fundamental para entender todas las publicaciones y preferencias de las redes. Son muchas las variables que unidas nos dan respuestas y crean nuevos interrogantes para las tendencias en nuestra sociedad actual (Porras, 2015).

Para estos estudios existen diversos software, muchos de ellos con versiones gratuitas más simples, o de prueba. Sin duda, sin estos avances tecnológicos, muchos de los estudios realizados no se hubiesen podido completar. Estos software se utilizan principalmente para representar y analizar series de datos y obtener una información más tangible y representativa (Porras, 2015). La experiencia demuestra que las dificultades para el ARS residen, más que en cuestiones conceptuales o teóricas, en el aprendizaje de la mecánica de la transformación de datos (Molina, 2005).

Una de las cuestiones más importantes en cualquier ARS es el formato de los datos para su importación a los software de análisis de redes. Familiarizarse con formatos estándares de datos puede ahorrar gran cantidad de tiempo ya que puede ser muy costoso tener que reformatear los datos. Eventualmente los datos son obtenidos en cualquier formato de archivo electrónico (hojas de cálculo, bases de datos o archivos de texto). Para grandes cantidades de datos, es apropiado el uso de bases de datos, pero pocos programas de análisis de redes pueden leer archivos de bases de datos directamente; esto conlleva un paso adicional de conversión (Borgatti, Everett, & Johnson, 2013).

UCINET (http://www.analytictech.com/) es un paquete general para el análisis de redes sociales (Borgatti, Everett, & Freeman, 2014). En este software, la entrada de datos se realiza, en principio, utilizando la hoja de datos ("spreadsheet") para introducir las etiquetas de los nodos en las columnas, luego en las filas y posteriormente las conexiones o lazos correspondientes entre los nodos (obteniéndose la "full matrix") (Borgatti, Everett, & Freeman, 2002). Aunque es posible de esta forma introducir pocos nodos de forma rápida (Molina, 2005), cuando se trata de cantidades considerables de datos, este mecanismo no es el más simple.

Mediante el DL Editor; aplicación de UCINET para importar datos, es posible introducir datos en diferentes formatos. El más sencillo y económico de todos es el "nodelist format" usado solo para datos binarios, por ejemplo: presencia o ausencia de una conexión (Borgatti, Everett, & Johnson, 2013).

Un formato de interés es el "edgelits format" que consiste en un set de filas en las que cada fila representa una conexión en la red. Cada fila tiene dos columnas indicando el par de nodos entre los que se establece la conexión (Borgatti, Everett, & Johnson, 2013). El "edgelist format" es una de las variantes más utilizadas en el almacenamiento de conexiones entre nodos. Es muy eficiente también el almacenamiento de datos realizado usando tres columnas (formato tipo "edgelist"): las dos primeras columnas indican el par de nodos entre los que se establece la conexión y la tercera columna (C) representa la cantidad de conexiones entre los nodos de ambos grupos.

La columna C permite establecer la cantidad de conexiones que se han efectuado entre dos nodos en, o hasta el momento de la medición. Aunque es posible almacenar datos sin esta columna, es más eficiente agrupar las conexiones en una columna independiente para evitar reescribir los nodos entre los que se ha establecido una conexión n cantidad de veces.

UCINET y otros programas de ARS permiten realizar transformaciones para formatos "edgelist", pero existen varios inconvenientes: los algoritmos de transformación de datos de programas de ARS no pueden ser modificados fácilmente por los usuarios, lo que entre otras cuestiones, retrasa el desarrollo de investigaciones e impide la adaptación de estos algoritmos a las variadas necesidades de los usuarios, además, muchos usuarios han optado por el uso de variantes gratuitas o libres que no disponen de un sistema de transformación de datos adecuado, o la modificación del mismo requiere conocimientos avanzados (instalación de software de programación, compilación, etc.).

Software como UCINET y NetDraw están distribuidos bajo licencias Shareware y Freewere (variante gratuita del Shareware) (Furht, 2010). Esto hace complejo el estudio y modificación del código, ya que es poco usual la disponibilidad del código fuente de software bajo estas licencias; además de ser muy limitadas sus posibilidades de modificación.

Otro inconveniente existente, es que no todos los usuarios o sistemas computacionales almacenan la información de la forma adecuada para su importación a programas de ARS (Borgatti, Everett, & Johnson, 2013). Estos almacenan información aplicando opciones de edición sobre sus datos (que dificultan la transformación) y usando software que operan con formatos complejos.

El objetivo principal de este trabajo fue diseñar y programar un software fácilmente modificable por los usuarios para la transformación de matrices de tres columnas [A (grupo 1), B (grupo 2) y C (conexiones entre nodos de ambos grupos)] a matrices completas ("full matrix") y simétricas (obteniendo un archivo ".csv" con la matriz transformada), y analizar las ventajas del software con respecto a los problemas de almacenamiento y transformación de datos.

MÉTODO

Para el diseño del algoritmo fue necesario extraer conclusiones de un grupo de criterios matemáticos para obtener un procedimiento general (método) de transformación de matrices eficiente:

1. Los datos del usuario se almacenan en una matriz de tres columnas y L filas (M3L) donde las dos primeras columnas (A y B) son los grupos 1 y 2, compuestos por un conjunto de nodos (identificados numéricamente).La tercera y última columna (C), son las conexiones entre los nodos de los grupos 1 y 2 (Fig. 1 izquierda).
2. En la M3L pueden encontrarse filas repetidas, es decir, se ha vuelto a establecer una conexión entre dos nodos semejantes. Se necesita un procedimiento para agrupar los datos de los grupos 1 y 2, y eliminar los valores repetidos.
3. Los nodos pueden encontrarse de forma no ordenada (como ocurre en la mayoría de los casos). Por ello se necesita un procedimiento para ordenar la lista de valores de menor a mayor.
4. Mediante los procedimientos de los puntos 2 y 3 se obtiene el grupo de valores (matriz) que precedido por cero, conforma la primera fila y columna de la "fullmatrix". En la Figura 1 (matriz ubicada a la derecha) este arreglo corresponde a los siguientes elementos: 100, 101, 102, 106, 107, 109, 201, 202, 203, 209, 210 y 211.
5. Teniéndose entonces la primera fila y columna de la "full matrix", se deben obtener el resto de los valores de la matriz completa mediante un procedimiento comparativo entre el arreglo obtenido de los puntos 2 y 3, y la M3L.

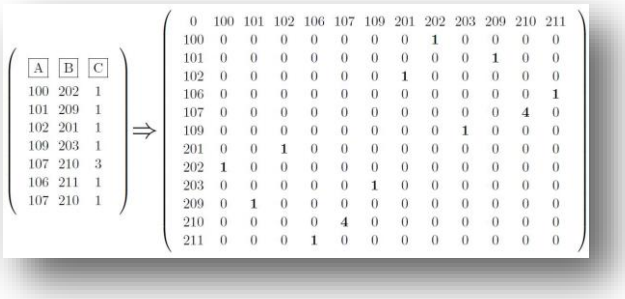

Figura 1. Transformación de los datos: a la izquierda la M3L y a la derecha la matriz transformada ("full matrix").

Para satisfacer los criterios expuestos con anterioridad se obtuvo el siguiente método.

Método de Transformación de Matrices (MTM):

Teniéndose una M3L, donde:

(1) Las dos primeras columnas (A y B) correspondan con los grupos 1 y 2 compuestos por un conjunto finito de nodos identificados numéricamente, conociendo que los nodos del grupo 1 pudiesen existir en el grupo 2 y viceversa.

Por ejemplo:

Suponiendo que el grupo 1 son los profesores de Matemática, y el grupo 2 son los profesores de Física de una universidad, si existe un profesor que imparta ambas materias, este pudiese estar tanto en el grupo 1, como en el grupo 2.

(2) Una tercera columna C, que corresponde con las conexiones entre los nodos de los grupos 1 y 2; se establece el siguiente método para transformar la M3L en una matriz completa y simétrica:

1. Los datos de la M3L (nodos de los grupos 1 y 2, y conexiones C) son almacenados en los arreglos *Ga*, *Gb* y *Gc* respectivamente (comenzando por la posición de almacenamiento #cero), todos con una cantidad de elementos = *L*. Por consiguiente (teniendo en cuenta la posición de almacenamiento inicial), los arreglos *Ga*, *Gb* y *Gc* tendrán *n* índices, siendo *n = (L-1).*
2. Se guardan los nodos de los grupos 1 y 2 (arreglos *Ga* y *Gb*) de la M3L en el arreglo *Gab* (de *(2*L – 1)* índices), se eliminan repeticiones, se ordenan los valores de menor a mayor y se guardan

en un nuevo arreglo (*Gf*) de *nv* índices. La cantidad de índices del nuevo arreglo se calcula: *nv = ((2*n - vr) + 1)*, donde *vr = cantidad de valores (nodos) repetidos en los grupos 1 y 2*.

Las operaciones de obtener la cantidad de elementos repetidos en un arreglo, eliminar estos valores, y ordenar los elementos de una matriz o arreglo, se realizan mediante funciones elementales que se encuentran disponibles en cualquier web especializada.

3. Se establece el conjunto de nodos o elementos ordenados y sin repeticiones (arreglo *Gf* de nv índices), precedido por "0", como la primera fila de la matriz completa o "full matrix". Los elementos ("0" y Gf) se guardan en la variable *row1* (en forma de cadena de texto) separados por un símbolo (generalmente "," o ";").
4. Se establece como condición inicial *row = 0* y *col = 0*.
5. Se compara el par (Gf_{row}; Gf_{col}), con el par (Ga_e;Gb_e) desde *e = 0* hasta *e = n* (incrementado *e* en una unidad). Si existe una semejanza ($Gf_{row} = Ga_e$ *y* $Gf_{col} = Gb_e$) se almacena el valor Gc_e en la variable *conex* y se incrementa el acumulado total de la forma *acum = acum + conex* (inicialmente *acum = 0*). Si se detecta otra semejanza, se almacena el nuevo valor Gc_e en la variable *conex*, incrementando nuevamente la variable *acum.* Ante cada comparación negativa, se debe invertir los valores del par (Ga_e; Gb_e) si se cumple que $Gf_{row} \neq Gf_{col}$, formando el par (Gb_e; Ga_e), y comparándolo con el mismo par (Gf_{row};Gf_{col}) con el que la comparación dio negativa. Si existe semejanza ($Gf_{row} = Gb_e$ *y* $Gf_{col} = Ga_e$), igualmente se almacena el valor Gc_e en *conex* y se incrementa *acum*.
6. Se guardan las conexiones en el arreglo de dos dimensiones *fullmatrix[row,col]* igualando el mismo a la variable *acum* una vez que *e = n* (*fullmatrix[row,col] = acum*). Como la matriz es simétrica, se guarda el valor de la variable *acum* utilizando los índices *row* y *col* de forma inversa (*fullmatrix[col,row] = acum*), comprobando antes que col ≠ *row*, ya que en caso contrario (col = *row*) no tendría sentido realizar la operación de invertir los índices.
7. Se realizan los pasos 5 y 6, incrementando *col* (en una unidad) desde *col = row* hasta *nv.*
8. Se realiza el paso 7, incrementando *row* (en una unidad) desde *row = 0* hasta *nv*.

Fue necesario manipular los resultados del MTM (*fullmatrix, row1* y *Gf)* para diseñar un algoritmo complementario y obtener la "full matrix" en un archivo ".csv".

Instrumentos

Para representar el algoritmo se utilizó el software TeXstudio v2.8.6 con el paquete *"algorithm2e"* para la creación de algoritmos en pseudocódigo (http://texstudio.sourceforge.net/).

Para el diseño de la interfaz gráfica y programación del algoritmo se utilizó wxBasic (http://wxbasic.net/) por poseer máquinas virtuales operables desde diferentes sistemas operativos (Mac OS, Linux y Windows).

RESULTADOS

Se obtuvo un algoritmo en pseudocódigo (Fig. 2) que permitió programar el software de transformación de matrices. En este se describen los procedimientos del MTM.

```
   Data: Ga, Gb, Gc, L, CHAR[a]
   Result: fullmatrix, row1, Gf
 1 n = (L − 1);
 2 for (pos = 0 to n) do
 3 |   Gab[pos] = Ga[pos] {El arreglo
   |   Gab tiene ((2 ∗ L) − 1) índices};
 4 end
 5 for (pos1 = 0 to n) do
 6 |   Gab[L + pos1] = Gb[pos1];
 7 end
 8 vr = ore[b](Gab);
 9 nv = ((2 ∗ n − vr) + 1);
10 Gs[nv] = dre[c](Gab);
11 Gf[nv] = arr[d](Gs[nv]);
12 row1 = 0;
13 for (rpos = 0 to nv) do
14 |   row1 = row1&[e]CHAR&
15 |   Gf[rpos];
16 end
17 for (row = 0 to nv) do
18 |   for (col = row to nv) do
19 |   |   acum = 0; conex = 0;
20 |   |   a = Gf[row] ; b = Gf[col];
21 |   |   for (e = 0 to n) do
22 |   |   |   if a = Ga[e] and
   |   |   |   b = Gb[e] then
23 |   |   |   |   conex = Gc[e];
24 |   |   |   |   acum = acum + conex;
25 |   |   |   end
26 |   |   |   if a ≠ b then
27 |   |   |   |   if a = Gb[e] and
   |   |   |   |   b = Ga[e] then
28 |   |   |   |   |   conex = Gc[e];
29 |   |   |   |   |   acum =
   |   |   |   |   |   acum + conex;
30 |   |   |   |   end
31 |   |   |   end
32 |   |   |   fullmatrix[row, col] =
   |   |   |   acum;
33 |   |   |   if col ≠ row then
34 |   |   |   |   fullmatrix[col, row] =
   |   |   |   |   acum;
35 |   |   |   end
36 |   |   end
37 |   end
38 end
39 return fullmatrix, row1, Gf;
```

Figura 2. Algoritmo en pseudocódigo obtenido para la transformación de la M3L a una matriz completa y simétrica.
(**a**) Separador usado para el manejo de los archivos ".csv" (Generalmente "," o ";"). (**b**) Función para obtener la cantidad de elementos repetidos en un arreglo. (**c**) Función para eliminar los elementos repetidos en un arreglo. (**d**) Función utilizada para ordenar (de menor a mayor en este caso) los elementos de un arreglo. (**e**) Símbolo usado para unir, por ejemplo: A&B = AB.

Se obtuvo un algoritmo complementario (Fig. 3) para la creación del archivo ".csv" a partir de los arreglos *fullmatrix y Gf*, y la primera fila *row1*.

```
   Data: fullmatrix, row1, Gf, filename[a], CHAR
   Result: filename.csv
 1 Open[b] archivo filename.csv en modo escritura (Append[c]);
 2 Print row1 en filename;
 3 for (r = 0 to nv) do
 4 |   sprint = Gf[r];
 5 |   for (c = 0 to nv) do
 6 |   |   sprint = sprint&CHAR&fullmatrix[r, c];
 7 |   end
 8 |   Print sprint en filename;
 9 end
10 Close archivo filename.csv;
```

Figura 3. Algoritmo complementario en pseudocódigo obtenido para la creación del archivo ".csv".
(**a**) Nombre del archivo con el que se desea guardar la matriz completa.
(**b**) El comando Open crea el archivo si este no existe.
(**c**) El modo de escritura Append se usa para comenzar a escribir al final del archivo.

Utilizando wxBasic se programaron los algoritmos de las Figuras 2 y 3, obteniéndose un software (Fig. 4 izquierda) para la transformación de una M3L a una matriz completa, exportando los resultados a un archivo ".csv".

Es posible descargar el software desde: https://archive.org/download/transmatrix/transmatrix.zip

El mismo se distribuye bajo licencia GPL (www.gnu.org/licenses/gpl-3.0.en.html) (software libre).

El archivo descargado ("**transmatrix.zip**" con **MD5**: B28F6DBA34BA92862AC6EADE490EB47A de 2.62 MB) contiene el software (máquina virtual para Windows y código), un archivo de texto ("Detalles.txt") donde se aclaran las principales dudas sobre el uso del software, el archivo "char.txt" donde se almacena el símbolo CHAR y un grupo de imágenes usadas en la interfaz gráfica. El código MD5 descrito con anterioridad puede ser utilizado para comprobar la autenticidad del archivo ".zip" descargado.

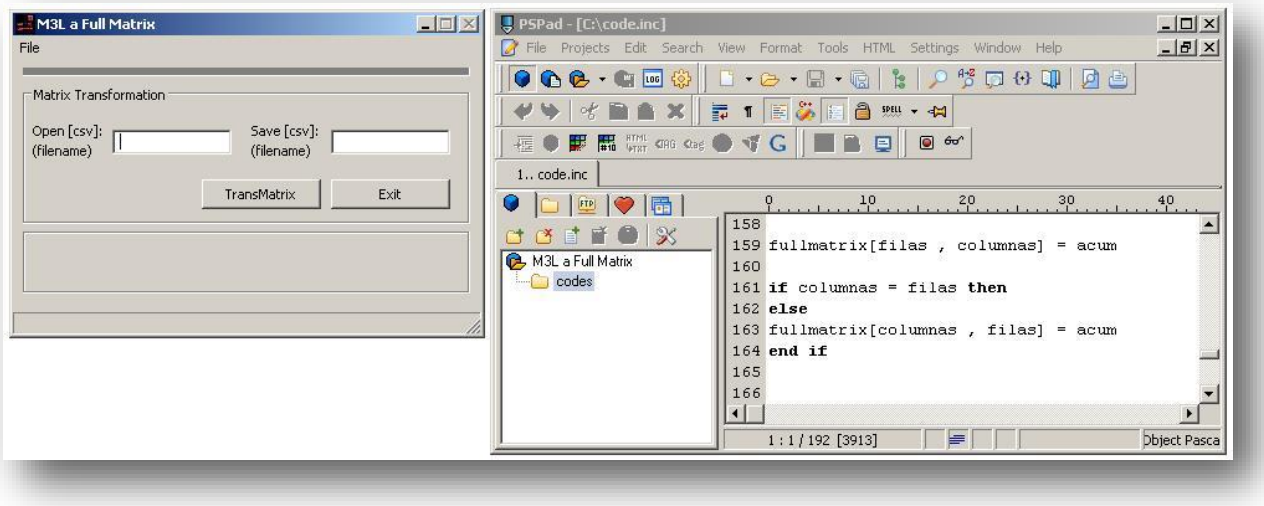

Figura 4. Pantalla principal del software de transformación de matrices (izquierda) y porción de código (derecha).

DISCUSIÓN

Es posible ver en el algoritmo (Fig. 2) que las operaciones que se efectúan desde el punto 1 hasta el 16 están destinadas a la conformación del arreglo *Gab* y a los procedimientos de obtención de elementos o nodos repetidos, eliminación de elementos repetidos y orden. A partir del punto 17 comienzan los procedimientos que permiten conformar el arreglo *fullmatrix* (arreglo que contiene los interceptos), que unido con la primera fila (*row1*) y los primeros elementos de cada fila (arreglo *Gf*), se utilizan para escribir la matriz completa en el archivo ".csv".

Esta escritura (Fig. 3) se realiza en primera instancia escribiendo en el archivo ".csv" los elementos de la primera fila, almacenados en la variable *row1*. Luego se almacena en la variable *sprint* el primer elemento del arreglo *Gf,* que corresponde con el primer valor de la segunda fila de la "full matrix". Se une a este elemento (separado por el símbolo *CHAR*) la primera fila de interceptos almacenada en el arreglo *fullmatrix*. Se realiza este procedimiento (regido por dos blucles o sentencia realizada repetidas veces) hasta que todas las filas se hallan escrito en el archivo ".csv".

La Figura 2 es el MTM llevado a un algoritmo en pseudocódigo sin ninguna modificación. Se usan los **for** para crear bucles o ciclos que terminan cuando se cumple la condición deseada. Por ejemplo, en el paso 8 del MTM se expone: *se realiza el paso 7, incrementando row (en una unidad) desde row = 0 hasta nv*. Para escribir este paso en el algoritmo usamos un **for** para indicar que el bucle va desde 0 hasta nv (**for** *row = 0* **to**

nv). En el siguiente paso, *row* valdrá 1, en el otro 2, y así sucesivamente. Los **if** son usados para ejecutar una instrucción si se cumple una condición. En el paso 27 del algoritmo (Fig. 2) se invierte el par (Ga_e; Gb_e) si se cumple que a ≠ b.

Obtener la matriz completa en un *archivo de texto de valores separados por comas* (".csv") hace que sea posible su lectura en programas como Microsoft Excel, OpenOffice Calc, etc. Los editores de hojas de cálculos deben de ser configurados para abrir este tipo de archivo teniendo en cuenta el símbolo que se usó (*CHAR*) para separar los elementos de cada fila. Los más usuales son "," o ";", pero pueden usarse otros. También pueden usarse delimitadores de texto como la comilla o doble comilla. Por ejemplo: para colocar los números 1 y 2 en dos columnas en un archivo ".csv" es posible escribir 1;2 (usando ";" como separador y ningún delimitador) o "1";"2" (usando ";" como separador y doble comilla como delimitador). En los algoritmos anteriores (Fig.2 y Fig.3) no se usan delimitadores de texto, solo separadores. Abrir un archivo ".csv" con OpenOffice Calc es posible de manera muy sencilla configurando estas opciones (Fig. 5).

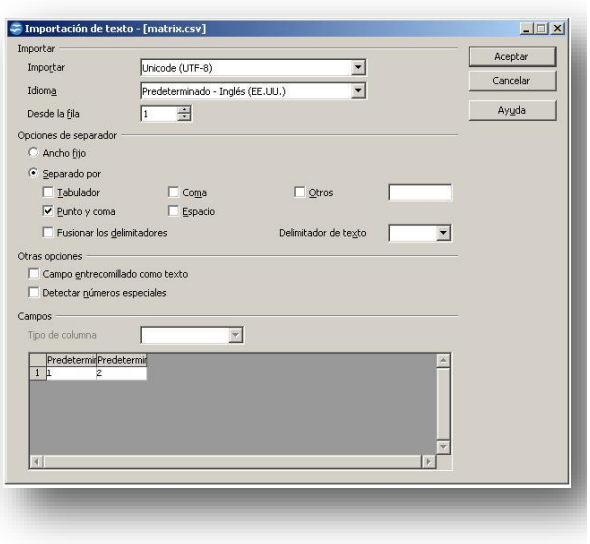

Figura 5. Abriendo archivo "matrix.csv" en OpenOffice Calc 3.3.

Como se aprecia en la Figura 5 para abrir un archivo ".csv" sin delimitadores de texto basta con borrar el símbolo que se encuentre en la caja de texto correspondiente (delimitador de texto) en ese momento.

El software de transformación fue diseñado de la manera más sencilla posible. En la caja de texto *open* (Fig. 4 izquierda), se escribe el nombre del archivo ".csv" que contiene la matriz (M3L) a transformar (por ejemplo: "matrix.csv"), de la misma forma, en la caja de texto *save* se escribe el nombre del archivo ".csv" con el que se desea guardar la matriz transformada (por ejemplo: "newmatrix.csv"). Haciendo clic en el botón *TransMatrix*, comienza la transformación de la M3L, apareciendo en pantalla un mensaje de espera. El proceso de transformación puede tardar varios minutos (según el tamaño (L) de la M3L).

El archivo con la matriz a transformar (M3L) debe encontrarse en el directorio del programa, junto con el ejecutable TransMatrix.exe. El archivo con la matriz transformada ("full matrix") se guardará en este mismo directorio.

Los arreglos *Ga*, *Gb*, y *Gc* (con los datos de las columnas de la M3L), pueden ser obtenidos fácilmente de una M3L almacenada en un archivo ".csv". Basta con realizar un procedimiento de lectura fila a fila y aplicar la función SPLIT, almacenando los resultados adecuadamente en los arreglos *Ga*, *Gb*, y *Gc*. Este procedimiento está claramente descrito en el código del programa.

La función SPLIT se usa para cortar una cadena de texto en partes más pequeñas utilizando uno o varios símbolos para realizar el corte, por ejemplo: Si se aplica la función SPLIT a la cadena de texto 1;2 con el símbolo ";" se obtiene un arreglo con el elemento *"1"* en la posición *#0* y el elemento *"2"* en la posición *#1* (se supone modo de indexación base-cero).

Una de las mayores ventajas obtenidas en el software de transformación, es que mediante el uso de wxBasic se pudo lograr que el código del programa sea fácilmente modificable y no hubiese la necesidad de realizar compilaciones para lograr cambios. El código del programa (interfaz gráfica, procedimiento de lectura y algoritmos) se encuentra en el archivo "code.inc" que viene acompañado del ejecutable (máquina virtual) "TransMatrix.exe" en el caso de la versión para Microsoft Windows. Este archivo de código ("code.inc") puede ser modificado usando cualquier editor de texto plano como Notepad o PSPad (http://www.pspad.com) para adaptarlo a las necesidades del usuario.

Realizar cambios a los códigos del software de trasformación de matrices y a los algoritmos pudiese permitir trabajar con nombres de personas en vez de números que los identifiquen; aplicar una función para ordenarlos alfabéticamente; crear matrices completas binarias (obviando las acumulaciones y centrando el código en la existencia o no de una conexión); eliminar la

simetría; sustituir o eliminar elementos o nodos no deseados, etc. Estos cambios pudiesen realizarse con un mínimo de complejidad. Es posible también modificar cualquier elemento de la interfaz gráfica del software trabajando sobre las primeras líneas del archivo "code.inc" donde se encuentra el código wx (http://wxbasic.net/).

Almacenar datos en hojas de cálculos utilizando formatos complejos (como ".xlsx") puede ser desventajoso si se aplican opciones como "combinar y centrar", sin contar que su lectura desde otro software no es tan sencilla como la de un archivo ".csv", que aunque posea otra extensión, no tiene diferencia alguna de un ".txt" (ambos son archivos de texto plano).

CONCLUSIONES

El software obtenido permite transformar una M3L almacenada en un archivo ".csv" a una matriz completa ("full matrix") y simétrica para su posterior aplicación en programas de ARS. Su fácil modificación permite a los usuarios realizar cualquier tipo de adaptación, crear métodos más eficientes o particulares de transformación, etc.

Uno de los aspectos más importantes a resaltar, es que, más que la utilidad del propio programa; una de las mayores ventajas del software se le puede atribuir a su carácter educativo. Generalmente, adaptados al software privativo, donde solo se resalta la utilidad de los programas computacionales, olvidamos que un software puede ser algo más que una herramienta si se permite aprender de su código. Mediante el software de transformación de M3L se pudo conocer cómo operan básicamente estos algoritmos, extraer observaciones para ayudar al usuario a almacenar correctamente su información y darle la posibilidad de crear sus propias herramientas de transformación de matrices.

REFERENCIAS

Borgatti, S. P., Everett, M. G., & Freeman, L. C. (2014). Ucinet. Encyclopedia of Social Network Analysis and Mining, 2261-2267. DOI: http://dx.doi.org/10.1007/978-1-4614-6170-8_316

Borgatti, S. P., Everett, M. G., & Freeman, L. C. (2002). Ucinet 6 for Windows [User's Guide] (pp. 22-23). Harvard: Analytic Technologies. Consulta [10-04-15]. Disponible en: https://www.soc.umn.edu/~knoke/_pages/UCINET_6_User's_Guide.doc

Borgatti, S. P., Everett, M. G., & Johnson, J. C. (2013). *Analyzing social networks*. London: SAGE Publications Limited.

Furht, B. (Ed.). (2010). *Handbook of social network technologies and applications*.Springer Science & Business Media. DOI: http://dx.doi.org/10.1007/978-1-4419-7142-5

Molina, J. (2006). Operaciones Básicas con UCINET 6. En J. L. Molina, A. Quiroga, J. Martí, I. Maya-Jariego, & A. de Federico (Eds). *Talleres de autoformación con programas informáticos de análisis de redes sociales*, 5-18. UAB: Barcelona. Disponible en: http://revista-redes.rediris.es/webredes/talleres/opbauci.pdf

Porras, A. (2015). NetWorking para Todos. Consulta [10-04-15]. Disponible en: http://networking.marketing-xxi.com/ars-analisis-redes-_sociales.html