UNIVERSIDAD SAN FRANCISCO DE QUITO

Colegio de Posgrados

Utilización de matrices de prueba para evaluar el desempeño del cálculo de los testores típicos

Diego Fernando Guilcapi Durán

Tesis de grado presentada como requisito
para la obtención del título de Magíster en Matemáticas Aplicadas

Quito, Abril 2012

UNIVERSIDAD SAN FRANCISCO DE QUITO

Colegio de Posgrados

HOJA DE APROBACION DE TESIS

Utilización de matrices de prueba para evaluar el desempeño de los algoritmos del cálculo de los testores típicos

Diego Fernando Guilcapi Durán

Eduardo Alba, Ph.D. Director de Tesis	…….. Firma
Carlos Jiménez, Ph.D. Director de la Maestría en Matemáticas Aplicadas y Miembro del Comité de Tesis	……... Firma
Santiago Gangotena, Ph. D. Decano del Colegio Politécnico	…… Firma
Víctor Viteri Breedy, Ph. D. Decano del Colegio de Postrados	…… Firma

Quito, Abril 2012

DEDICATORIA

La presente tesis está dedicada a mi Padre Celestial. Él es quien me dio la vida, quien a través de su Palabra me enseña cada día a disfrutarla y conservarla, quien me ama sin esperar que yo le ame, en quien puedo confiar sin importar las circunstancias; quien nunca me ha soltado de su mano; quien me ha dado un propósito; quien me ha enseñado que lo más valioso que existe en esta vida, no son los títulos, las profesiones o las empresas, sino el amor a Él y a las personas.

Porque “si yo hablase lenguas humanas y angélicas, y no tengo amor, vengo a ser como metal que resuena, o címbalo que retiñe. Y si tuviere profecía, y entendiese todos los misterios y toda ciencia, y si tuviese toda la fe, de tal manera que trasladase los montes, y no tengo amor, nada soy. Y si repartiese todos mis bienes para dar de comer a los pobres, y si entregase mi cuerpo para ser quemado, y no tengo amor, de nada me sirve. El amor es sufrido, es benigno; el amor no tiene envidia, el amor no es jactancioso, no se envanece; no hace nada indebido, no busca lo suyo, no se irrita, no guarda rencor; no se goza de la injusticia, mas se goza de la verdad. Todo lo sufre, todo lo cree, todo lo espera, todo lo soporta”.

1 Corintios 13 1-7

AGRADECIMIENTOS

Mis más sinceros agradecimientos están dirigidos a las siguientes personas:

A mi madre, porque es ella quien hizo posible que estudie en la USFQ. Quien confió en mí y me dio la gran oportunidad de hacer posible este sueño. A ella toda la gratitud y todas las bendiciones del cielo.

A mi padre, por ayudarme a sobresalir, por las numerosas madrugadas que hicieron posible que llegara a la universidad, y por brindarme su apoyo incondicional. A él los mejores deseos y éxitos en su vida.

A mi Tía Lauri, porque ha sido como una segunda madre para mí. Su ejemplo, dedicación y bondad han hablado más que mil palabras.

A mi hermano, Carlitos, porque más que un hermano, eres un gran amigo y un gran hombre de Dios. Te quiero mucho y vamos a alcanzar el propósito que Dios ha puesto en nuestros corazones.

A mi hermana, Carlita, porque tus sabios consejos me dieron aliento para afrontar los tiempos difíciles. Te quiero mucho y le doy gracias al Señor por haberte puesto en mi camino para que pueda aprender de ti.

A Edu Cárdenas, porque gracias a ti, pude ver y conocer a Dios. Te debo algo que nunca podré pagarte. Gracias por no rendirte conmigo y mostrarme el camino al cielo; eres un excelente amigo.

A Eduardo Alba, por brindarme la oportunidad de trabajar con él, por su gran ayuda, paciencia y entrega, y por el gran ejemplo que representa a toda persona que desea ser un Profesor Universitario.

A Carlos Jiménez, por el conocimiento que me ha brindado y el gran desarrollo intelectual que despierta en sus estudiantes.

A Marcelo Almeida y Andreita Ayala quienes desinteresadamente me dieron mucho apoyo y conocimiento a lo largo de esta Maestría.

A Luis Gordillo, por la gran motivación que extiende, y su enorme ejemplo.

A mi colegio, en especial a Xavier Rivera, quien desinteresadamente me ayudó a conseguir una beca en la USFQ. Gracias por todo Xavier.

A Wilson Pérez, Michael Dyer, Juan Carlos Bustamante, Daniel Merchán y Ximena Córdova, por su gran esfuerzo en inculcar lo que con mucho sacrificio aprendieron.

A todas las personas que no alcanzo a mencionar, pero no menos importantes. Muchas gracias por su apoyo.

RESUMEN

El siguiente trabajo está dedicado a la Utilización de Matrices de Prueba para Evaluar el Desempeño de los Algoritmos del Cálculo de los Testores Típicos.

El trabajo comienza con una exposición del marco teórico de la Teoría de Testores. Se detallan los Algoritmos de Escala Exterior: BT y LEX, que se utilizarán para encontrar el conjunto de todos los testores típicos de una matriz básica.

En la parte fundamental del trabajo se describen los métodos para generar matrices de prueba. Se analizan dos casos diferentes: Matrices con la misma dimensión y diferente número de testores típicos y Matrices con diferente dimensión e igual número de testores típicos.

Finalmente, se presentan ejemplos de aplicación de matrices de prueba a la evaluación del desempeño de los algoritmos. Así, se utiliza el software R para medir y analizar el tiempo que el Algoritmo BT y el Algoritmo LEX emplean en hallar todos los testores típicos de las matrices diseñadas.

ABSTRACT

This thesis presents the Use of Test Matrices to Measure the Algorithms Performance in Calculating Typical Testors.

It begins with a description of the theory related to the Theory of Testors, and exposes the algorithms BT and LEX, which I use to find the typical testors of a basic matrix.

In the main part of this thesis, I describe new ways to generate the following matrices: Matrices with equal size and different number of typical testors and Matrices with different dimensions and equal number of typical testors

Finally, I show application examples of test matrices to evaluate the performance of the algorithms. I use R software to measure and analyze the time that both algorithms (BT and LEX) spent to find all typical testors of the designed matrices.

TABLA DE CONTENIDOS

PÁG.

LISTA DE FIGURAS

FIGURA PÁG.

LISTA DE TABLAS

INTRODUCCIÓN

“En los problemas de Reconocimiento de Patrones, la selección de variables es crucial para reducir de modo eficiente el número de variables (rasgos o características) mediante el cual se describen los objetos involucrados en un determinado estudio. Una vía para lograr esta reducción es determinar la importancia informacional de las variables. La importancia informacional puede ser una medida de la capacidad de la variable de discriminar a objetos que pertenecen a clases diferentes. La forma de determinar esta medida es usando la información que brinda el conjunto de todos los testores típicos de una matriz de comparación” (Alba y Santana 1).

“Existen varios algoritmos para el cálculo de todos los testores típicos de una matriz de comparación. Todos los algoritmos que se reportan en las publicaciones pueden demostrar desde el punto de vista teórico que mejoran el desempeño del algoritmo trivial que revisa cada subconjunto de rasgos. Es decir, cada uno de ellos logra un “ahorro” al no analizar elemento por elemento. Sin embargo, sus heurísticas son muy diversas y se hace difícil desde el mismo punto de vista teórico demostrar qué tan eficiente son.” (Alba y Santana 1).

El presente proyecto está enfocado en la utilización de matrices generadas artificialmente para evaluar el desempeño del cálculo de los testores típicos. Esta matriz que denominaremos “matriz de prueba” es una matriz cuyo número de testores típicos es conocido y puede construirse aplicando operadores de matrices tales como: concatenación (o fusión simple) y fusión combinatoria. De esta manera, se puede realizar una exploración del tiempo de ejecución que emplea un algoritmo apropiado en encontrar todos los testores típicos tomando como entrada *dichas matrices de prueba*.

Hasta el momento se ha probado la eficiencia de diferentes algoritmos para el cálculo de los testores típicos mediante un conjunto de matrices obtenidas a partir problemas reales. Por ejemplo, se ha comparado el tiempo de ejecución (en segundos) del algoritmo Fast implementation for algorithm CT_EXT (FI_CT_EXT), con los algoritmos BT, CT, LEX y CT_EXT utilizando matrices básicas con las siguientes dimensiones: $10 \times 34,\ 20 \times 38,\ 209 \times 32,\ 209 \times 47\ y\ 269 \times 42$; de las cuales dos provienen de problemas de diagnóstico médico. En todas estas comparaciones según Guillermo Sánchez, el algoritmo más eficiente resultó ser el FI_CT_EXT.

Incluso se ha realizado una comparación entre el FI_CT_EXT y el CT_EXT utilizando 4 bases de datos reales provenientes del UCI Machine Learning Repository, donde nuevamente el FI_CT_EXT desplegó mejores resultados. Las dimensiones de las matrices utilizadas fueron: $14 \times 17,\ 120 \times 35,\ 30 \times 22,\ (100 - 1000) \times 57\ y\ 50 \times (25 - 100)$ (Sánchez et al 100).

En esta tesis, en primer lugar se desarrolla un marco teórico del tema en el que se utiliza fundamentalmente el documento “Generación de matrices para evaluar el desempeño de estrategias de búsqueda de testores típicos” de Alba-Santana. En esta parte se introduce al lector en los conceptos de: testor, testor típico, operadores de matrices y sus propiedades, y resultados teóricos en la determinación del conjunto de testores típicos. Se presentan definiciones, proposiciones, demostraciones y ejemplos que permiten clarificar el contenido. Adicionalmente, se elabora una introducción al funcionamiento de los Algoritmos BT y LEX.

En segundo lugar, se desarrolla la metodología de la tesis. Así, se presentan: una implementación de los Algoritmos BT y LEX (lexicográfico), que será de gran utilidad para el objetivo del proyecto.

Luego, se incluye una sección titulada como: Generación y Empleo de Matrices de Prueba. Aquí, se amplía el contenido del documento anteriormente citado de Alba-Santana, y se presentan nuevas maneras de encontrar matrices con las siguientes características: *Matrices con la misma dimensión y diferente número de testores típicos y Matrices con diferente dimensión e igual número de testores típicos.* Inmediatamente, se emplean los algoritmos BT y LEX para medir el tiempo que estos demoran en determinar todos los testores típicos de las matrices de prueba señaladas anteriormente.

En la tercera parte, se analizan y discuten los resultados del proceso de exploración realizado; para finalmente presentar las conclusiones y recomendaciones de esta tesis.

OBJETIVOS GENERALES Y ESPECÍFICOS

OBJETIVO GENERAL

- Utilizar matrices de prueba para evaluar el desempeño de los algoritmos determinísticos que hallen el conjunto de todos los testores típicos.

OBJETIVOS ESPECÍFICOS

- Emplear matrices de prueba con la misma dimensión y diferente número de testores típicos para evaluar el desempeño del cálculo de los testores típicos.

- Hacer uso de matrices de prueba con diferente dimensión e igual número de testores típicos para evaluar el desempeño del cálculo de los testores típicos.

JUSTIFICACIÓN DEL PROYECTO

"El concepto de *testor típico* juega un rol muy importante en la solución de problemas de reconocimiento supervisado de patrones, cuando se usa el enfoque *lógico combinatorio* (Ruiz et al. 2001). En una aproximación básica, un testor típico es una colección de características o rasgos que discrimina las descripciones de objetos pertenecientes a diferentes clases, y es mínimo en el orden parcial determinado por la inclusión de conjuntos. A través de las etapas de la solución de un problema de reconocimiento de patrones, los testores típicos pueden ser aplicados para satisfacer diferentes objetivos. Por ejemplo, para construir un orden jerárquico de características de acuerdo a su importancia informacional (Lazo et al. 1995) y/o para determinar los conjuntos de apoyo en los algoritmos de clasificación supervisada basados en la precedencia parcial (De la Vega 1998)" (Alba 1).

"En el enfoque lógico combinatorio, la información de un problema de reconocimiento supervisado de patrones puede ser reducida a una matriz booleana denominada *matriz de comparación* que resume las comparaciones de objetos de clases diferentes. Los testores típicos se buscan entre todos los posibles subconjuntos de etiquetas de las columnas de la matriz de comparación. Si aplicamos un algoritmo *trivial* de búsqueda que revisa cada subconjunto de *n* columnas, el mismo se ejecutaría una cantidad 2^n de veces, número que se vuelve intratable para valores de *n* relativamente pequeños. Es por ello que se han creado varios algoritmos que permiten reducir la cantidad de subconjuntos que se revisa con respecto a la aplicación del algoritmo trivial" (Alba 1)

Según Sánchez et al 92, todos los algoritmos reportados poseen complejidad exponencial dependiendo principalmente del número de columnas de la matriz utilizada.

“Para matrices de dimensión pequeña, se han implementado varios tipos de algoritmos determinísticos (Sánchez 1997). A principios de la década pasada comenzaron a desarrollarse algoritmos evolutivos para tratar matrices de dimensiones mayores (Alba et al. 2000). En todos los casos se ha podido demostrar desde el punto de vista teórico la eficacia de estos algoritmos, es decir, en el caso de los determinísticos se garantiza el cómputo del conjunto de todos los testores típicos de una matriz y en el caso de los evolutivos una convergencia más o menos rápida a ese conjunto. Sin embargo, no se ha tenido el mismo éxito para demostrar la eficiencia (tiempos de cómputo) desde el punto de vista teórico. Quiere decir que cada creador de un nuevo algoritmo (o de la modificación de uno anterior), lo “defiende” realizando experimentos de cálculo sobre matrices seleccionadas (no estándar) y mostrando en una tabla de resultados, tiempos mejores que los que corresponden a sus predecesores” (Alba 2).

“Por lo anterior expuesto es necesario *estudiar el desempeño de los diferentes algoritmos propuestos para calcular los testores típicos* ***sobre la base de un conjunto estandarizado de matrices de prueba****, lo cual constituye el objetivo fundamental de este proyecto”* (Alba 2).

“Al no existir ninguna evidencia en la literatura de este estudio, sus resultados serán originales y permitirán determinar la eficiencia de los algoritmos y la aplicación de los testores típicos a una amplia gama de problemas de selección de variables y reconocimiento supervisado de patrones” (Alba 2).

1. CAPÍTULO I

1.1 MARCO TEÓRICO

El marco teórico de esta tesis está fundamentado principalmente en el documento “Generación de matrices para evaluar el desempeño de estrategias de búsqueda de testores típicos” de Eduardo Alba-Cabrera y Roberto Santana. Las definiciones y teoremas son propios de dichos autores. Además, se han incluido

explicaciones, demostraciones y ejemplos propios del autor de este documento que facilitan el estudio y comprensión del tema.

1.1.1 Los Conceptos de Testor y Testor Típico

Sea U una colección de objetos que se describen mediante un conjunto de ***n características o rasgos*** y están agrupados en ***l clases***. Comparando característica a característica cada par de objetos pertenecientes a diferentes clases, se puede obtener una matriz $M = [m_{ij}]_{p\times n}$ donde $m_{ij} \epsilon \{0,1\}$ y p es el número de pares. $m_{ij} = 1$ significa que los objetos del par denotado por i son diferentes en la característica j; mientras que $m_{ij} = 0$ indica que los objetos del par denotado por i son similares en la característica j.

Sea $I = \{i_1, \dots, i_p\}$ el conjunto de las filas de la matriz M y sea $J = \{j_1, \dots, j_n\}$ el conjunto de las columnas (características) de dicha matriz. Si $T \subseteq J$, sea $M_{/T}$ la matriz obtenida a partir de M que resulta al eliminar todas las columnas que no pertenecen al conjunto T.

Definición 1: Un conjunto $T = \{j_{k_1}, \dots, j_{k_l}\} \subseteq J$ es un testor de la matriz M si no existe ninguna fila de ceros en $M_{/T}$.

Definición 2: La característica $j_{k_r} \in T$ es típica con respecto a T y M, si $\exists q, q \in \{1, \dots, p\}$ tal que $a_{i_q j_{k_r}} = 1$ y $para\ todo\ p \neq r,\ p \in \{1, \dots, s\}\ con\ s > 1,\ a_{i_q j_{k_p}} = 0$.

Definición 3: Un conjunto T tiene la propiedad de tipicidad con respecto a una matriz M si todas las características en T son típicas con respecto a T y M.

Por otro lado, se debe hacer notar que la operación de inclusión de conjuntos define una relación de orden parcial sobre el conjunto de todos los testores de una

matriz. A continuación se indica qué representan los testores típicos en esa relación de orden parcial:

"En general, a cada subconjunto de un conjunto dado se le puede hacer corresponder un $n-uplo\ binario$ del tamaño de la cardinalidad del conjunto. Con este antecedente se puede determinar que los testores de una matriz pueden ser representados con $n-uplos$ que poseen un 1 en la posición de la característica incluida en el testor y un cero en la no incluida. De esta forma, se logra comprobar que la relación de orden parcial establecida por la inclusión de conjuntos se mantiene por los $n-uplos$, dado que los testores típicos continúan siendo incomparables (conjuntos disjuntos) y los testores que son supraconjuntos de un testor típico siguen siendo supra $n-uplos$ del $n-uplo$ correspondiente al testor típico" (Almeida, Guilcapi, Méndez 2).

Proposición 1: Un conjunto $T=\{j_{k_1},\dots,j_{k_l}\}\subseteq J$ tiene la propiedad de tipicidad con respecto a la matriz M si y solo si se puede obtener una matriz identidad en $M_{/T}$, eliminando algunas filas

Demostración Proposición 1:

Sea T un subconjunto de rasgos típicos, sea $M_{/T}$ la matriz asociada a T. Se tiene entonces que para cada columna de $M_{/T}$ existe al menos una entrada cuyo valor es 1 y las entradas correspondientes a esa fila en el resto de columnas de $M_{/T}$ toman el valor de cero.

De donde se puede asegurar que existe la posibilidad de permutar las columnas de $M_{/T}$, considerando el criterio de ubicar cada columna en la posición correspondiente a la fila que determina su tipicidad, obteniendo así una matriz identidad como submatriz de $M_{/T}$ (Almeida, Guilcapi, Méndez 3).

Definición 4: Un conjunto $T = \{j_{k1}, \dots, j_{ks}\} \subseteq J$ se denomina *testor típico* de M si es un testor y tiene la propiedad de tipicidad con respecto a M.

Definición 5:

$Decimos\ \ que$ a $es\ menor\ a$ b$(a < b) si\ \ para\ todo\ i, a_i \leq b_i\ \ y\ existe\ un\ j\ tal\ que\ a_j \neq b_j$.

Definición 6: a es una fila básica de M si no hay otra fila menor a a en M.

Definición 7: La matriz básica de M es la matriz M' que solo contiene todas las filas básicas de M.

La siguiente proposición es una caracterización de la matriz básica.

Proposición 2: M' es una matriz básica si y solo si para dos filas a y b cualesquiera, $a, b \in M'$ existen dos columnas i y j tales que $a_i = b_j = 1$ y $a_j = b_i = 0$.

Dada una matriz A, denotamos como $\Psi^*(A)$ el conjunto de todos los testores típicos de A.

Demostración Proposición 2:

Como M' es una matriz básica, se tiene que sus filas son básicas. Asi, para cualquier par de filas $a, b \in M'$ se cumple que a y b son incomparables.

De donde se sigue que

$$[\exists i : a_i > b_i \vee \forall j : a_j = b_j] \wedge [\exists k : b_k > a_k \vee \forall l : a_l = b_l]$$
$$\Rightarrow \exists i, k : a_i > b_i \wedge b_k > a_k$$
$$\Leftrightarrow \exists i, k : (a_i = b_k = 1) \wedge (b_i = a_k = 0)\ \text{l.q.q.d.}$$

(Almeida, Guilcapi, Méndez 3)

Proposición 3: $\Psi^*(M) = \Psi^*(M')$

Demostración Proposición 3:

P.D. $\Psi^*(M) = \Psi^*(M') \iff \Psi^*(M) \subseteq \Psi^*(M')$ **y** $\Psi^*(M') \subseteq \Psi^*(M)$

$(\Rightarrow)$ $\Psi^*(M) \subseteq \Psi^*(M')$

$\Psi^*(M)$ es el conjunto de todos los testores típicos de M

Sea $H \in \Psi^*(M)$, entonces $M_{/H}$ no posee ninguna fila de ceros. Por la proposición 1, una permutación de las columnas de $M_{/H}$ genera una submatriz identidad. Ahora, si se toma únicamente las filas que asignan la tipicidad a cada característica y se eliminan sus réplicas y suprafilas, se obtiene una matriz identidad.

Por otro lado, dado que una permutación de las columnas de toda matriz asociada a un testor típico de M' es una matriz identidad, se logra determinar que al suprimir suprafilas y réplicas en una matriz asociada a un testor típico de M, se obtiene una matriz asociada a un testor típico de M'.

$(\Leftarrow)\Psi^*(M') \subseteq \Psi^*(M)$

Toda permutación de las columnas de la matriz asociada a un testor típico de una matriz básica M', es una matriz identidad. Por lo tanto, si se aumentan las filas necesarias (suprafilas y réplicas) para convertir M' en M; entonces, el conjunto de características seguirá siendo típico.

También, dado que las filas que se aumentan no son filas de ceros, esto implica que la nueva matriz es la matriz asociada a un testor típico de M.

(Almeida, Guilcapi, Méndez 3).

De acuerdo con la proposición 3, para obtener el conjunto $\Psi^*(M)$ es conveniente encontrar la matriz M' y luego calcular el conjunto $\Psi^*(M')$. Teniendo en cuenta que M' tiene menor o igual número de filas que M, la eficiencia de los algoritmos debe ser mayor para M' que para M.

De hecho todas las matrices con la que se va a trabajar en esta tesis son matrices básicas.

1.1.2 Operadores de Matrices y sus Propiedades

Operador Concatenación (Fusión Simple)

La concatenación de dos matrices A y B es el proceso de juntar una o más matrices para crear una nueva matriz mediante un operador (The MathWorks).

Sea φ el operador de concatenación o también nombrado fusión simple (Alba y Santana 2) tal que $(A|B) = \varphi(A, B)$, se lo define como:

$$\varphi: \Re^{p\times q} \times \Re^{p\times q'} \Longrightarrow \Re^{p\times(q+q')}, q > 0, q' > 0, p > 0$$

Dadas dos matrices $A \in \Re^{p\times q}$ y $B \in \Re^{p\times q'}$, $\varphi(A, B) = [A\ B]$

Ejemplo 1:

Sea $A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$ de dimensión 3×3 y $B = \begin{bmatrix} 1 & 2 \\ 5 & 5 \\ 2 & 8 \end{bmatrix}$ de dimensión 3×2 entonces:

$\varphi(A, B) = [A\ B] = \begin{bmatrix} 1 & 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 5 & 5 \\ 1 & 1 & 0 & 2 & 8 \end{bmatrix}$ de dimensión 3×5 .

Note que este operador mantiene el mismo número de filas y suma el número de columnas resultantes.

Propiedades del Operador Concatenación φ

1. $\varphi(\varphi(A, B), C) = \varphi\big(A, \varphi(B, C)\big) = \varphi(A, B, C)$

2. Sean A, B matrices booleanas, si A o B son matrices básicas entonces $\varphi(A, B)$ es una matriz básica

Demostración de la segunda propiedad del operador Concatenación:

P.D. Sean A, B matrices básicas, entonces $\varphi(A,B)$ es una matriz básica.

En efecto si A , B son matrices básicas, se tiene que para todo par de filas a,b de A y c,d de B, existen un par de columnas i,j de A y k,l de B tales que: $a_i = b_j = 1$ y $a_j = b_i = 0$ y de la misma manera $c_k = d_l = 1$ y $c_l = d_k = 0$.

Además como el efecto del operador φ no es otro que concatenar las matrices una a continuación de otra, se puede concluir que para todo par de filas x,y de $\varphi(A,B)$ existen almenos dos columnas m,n de $\varphi(A,B)$, tal que $x_m = y_n = 1$, $x_n = y_m = 0$, pues basta tomar la columna m de $\varphi(A,B)$ como la columna i de A y la columna n de $\varphi(A,B)$ como la columna j de A.

(Almeida, Guilcapi, Méndez 4)

Operador Fusión Combinatoria

Definimos el operador fusión combinatoria, denotado por :

$$\theta: \Re^{p\times q} \times \Re^{p\prime\times q\prime} \Longrightarrow \Re^{p*p\prime\times(q+q\prime)}, q>0, q'>0, p>0, p'>0$$

Dadas dos matrices $A = [a_{ij}]_{p\times q}$ y $B = [b_{ij}]_{p\prime\times q'}$, La operación θ se define como:

$$\theta(A,B) = \begin{bmatrix} A(1,:) & B(1,:) \\ \dots & \dots \\ A(1,:) & B(p',:) \\ \dots & \dots \\ A(p,:) & B(1,:) \\ \dots & \dots \\ A(p,:) & B(p',:) \end{bmatrix}$$

Ejemplo 1:

$$\text{Sea } A = \begin{bmatrix} 1 & 5 \\ 0 & 9 \\ 7 & 8 \end{bmatrix} \text{ y } B = \begin{bmatrix} 1 & 0 & 3 \\ 2 & 3 & 6 \end{bmatrix}, C = \theta(A,B), \text{ entonces } C = \begin{bmatrix} 1 & 5 & 1 & 0 & 3 \\ 1 & 5 & 2 & 3 & 6 \\ 0 & 9 & 1 & 0 & 3 \\ 0 & 9 & 2 & 3 & 6 \\ 7 & 8 & 1 & 0 & 3 \\ 7 & 8 & 2 & 3 & 6 \end{bmatrix}$$

Note que la primera fila de la matriz A se ha conectado con la primera y segunda fila de la matriz B. Esto ha ocurrido con todas la filas de A cuando se aplicó el operador fusión combinatoria.

Además, observe que la dimensión de A es 3×2 , la dimensión de B es 2×3 y la dimensión de C es $(3 * 2) \times (2 + 3)$. Es decir $\dim(C) = 6 \times 5$.

Propiedades del Operador Fusión Combinatoria θ

1. $\theta(\theta(A,B),C) = \theta\big(A,\theta(B,C)\big) = \theta(A,B,C)$

2. Sean A,B matrices booleanas, si $A\ y\ B$ son matrices básicas entonces $\theta(A,B)$ es una matriz básica

Demostración de la segunda propiedad del operador Fusión Combinatoria:

Sea define el operador θ de la siguiente manera:

$\theta : \Re^{pxq} \times \Re^{p'xq'} \to \Re^{pp'x(q+q')}$

$$\text{tal que: } \theta(A = [a_{ij}]_{p\times q}, B = [b_{kl}]_{p'\times q'}) = \begin{bmatrix} \varphi(A_1,B) \\ \varphi(A_2,B) \\ \vdots \\ \varphi(A_p,B) \end{bmatrix},$$

$$\text{con } A_i = \begin{bmatrix} a_{i1} & a_{i2} & \cdots & a_{iq} \\ \vdots & \vdots & \vdots & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{iq} \end{bmatrix}_{p'\times q}, \forall i \in \{1,...,p\}$$

De donde por el hecho que B es básica, se sigue que cada $\varphi(A_i,B)$ es una matriz básica y además Dado que A es una matriz básica se tiene que para cualquier par de filas a de $\varphi(A_i,B)$ y b de $\varphi(A_j,B)$, con $i,j \in \{1,..,p\}, i \neq j$, existen un par de columnas m,n de A que garantizan que $a_m = b_n = 1$ y $a_n = b_m = 0$.

Por lo tanto podemos concluir que $\theta(A,B)$ es una matriz básica, si A y B tambien lo son.

(Almeida, Guilcapi, Méndez 4)

Propiedades de la matriz identidad de orden n, I_n

1. I_n es una matriz básica.

2. El número de testores típicos de I_n es igual a 1, es decir que $|\Psi^*(I_n)| = 1$

3. $\Psi^*(I_n) = \{J_{I_n}\}$ donde J_{I_n} es el conjunto de todas las etiquetas de columnas de I_n.

1.1.3 Resultados Teóricos en la Determinación de Ψ^*

Sea Q la concatenación de N $(N > 1)$ matrices $B \in \Re^{p \times q}$, es decir $Q = \varphi\left(\underbrace{B, \dots, B}_{N\ veces}\right)$. Sea $\Psi^*(\mathrm{B}) = \{\mathrm{T}_1, \dots, \mathrm{T}_\mathrm{v}\}$.

Teorema 1: $|\Psi^*(\mathrm{Q})| = N^{T_1} + \cdots + N^{T_v}$

Sea $A_1, \dots, A_m$ m matrices tales que $A_i = I_{n_i}$. Sean $J_{A_1}, \dots, J_{A_m}$ el conjunto de todas las columnas de estas matrices tales que: $\left\{J_{A_i} \cap J_{A_j}\right\} = \emptyset\ para\ todo\ i, j \in \{1, \dots, m\}, i \neq j$

Sea A la matriz obtenida al aplicar el operador fusión combinatoria a las matrices A_i, es decir $A = \theta(A_1, \dots, A_m)$.

Teorema 2: $\Psi^*(\mathrm{A}) = \{\Psi^*(\mathrm{A}_1), \dots, \Psi^*(\mathrm{A}_\mathrm{m})\} = \{J_{A_1}, \dots, J_{A_m}\}$

Corolario 1: $|\Psi^*(\mathrm{A})| = m$

Corolario 2: $\left|\Psi^*\left(\varphi\left(\underbrace{A, \dots, A}_{N\ veces}\right)\right)\right| = N^{n_1} + \cdots + N^{n_m}$

1.1.4 El Algoritmo BT

Este algoritmo es de escala exterior. “El orden que lo caracteriza es aquel generado por los números naturales de forma ascendente en su notación binaria mediante un *n*-uplo booleano. Éste constituye un orden total sobre los elementos no

nulos del conjunto potencia del conjunto de rasgos de la matriz" (Pons y Santiesteban 6).

"El algoritmo BT comienza entonces por el *n*-uplo (0....01) y procede comprobando si cada vector (*n*-uplo) generado es un testor o un testor típico. Según el resultado establece "saltos" en el conjunto potencia" (Pons y Santiesteban 6). Dichos saltos se dividen en: saltos no testor y saltos testor, dependiendo si el *n-uplo* analizado no es testor o sí lo es. Si el n-uplo es testor típico se saltan todos aquellos n-uplos que son supratestores de este, y se almacena cada testor típico resultante en una lista (conjunto de todos los testores típicos). Por otro lado, si el n-upo es no testor, se observa la fila o las filas que impiden que sea testor, y se escoge aquella que tiene el 1 más a la derecha de la fila que impide que sea testor.

"El algoritmo termina cuando llega al *n*-uplo (1....11). Para ello se apoya en caracterizaciones de los conceptos de testor y testor típico y en proposiciones que definen los saltos cuando el *n*-uplo analizado es o no un testor" (Pons y Santiesteban 6), como aquellas que se explicó en el párrafo anterior.

1.1.5 El Algoritmo LEX

"Este algoritmo es de escala exterior, ya que busca los Testores Típicos implantando un orden sobre el conjunto potencia de los rasgos y define ciertos saltos con el objetivo de obviar algunos conjuntos." (Santiesteban y Pons 89)

"El algoritmo LEX va generando listas de rasgos siguiendo el orden lexicográfico. De allí se deriva su nombre. La idea del algoritmo es ir construyendo listas de rasgos que posean la propiedad de tipicidad y luego comprobar si este conjunto de rasgos constituye un testor. Al momento de obtener un testor típico se producen saltos." (Santiesteban y Pons 92)

“El algoritmo empieza analizando el primer rasgo de una matriz básica MB. Un nuevo rasgo se puede incorporar a la lista si se cumple que no es excluyente con la lista, es decir, si puede coexistir con los rasgos de la lista para formar un testor típico y tiene filas típicas con respecto a la lista. Posteriormente, se comprueba si el conjunto de rasgos contenidos en la lista junto con el nuevo rasgo forma un testor, verificando si no existe fila en la MB (tomando en cuenta únicamente esos rasgos) completa de ceros. Si se cumple que es testor, entonces estamos en presencia de un testor típico y se almacena.” (Santiesteban y Pons 92)

“Si se encontró un testor típico y éste contiene al último rasgo de MB, entonces se saltan todos sus subconjuntos consecutivos. Si no contiene al último rasgo de MB, para saltar todos los supraconjuntos del testor típico encontrado se elimina el último rasgo de la lista y se analiza si se puede incluir el próximo rasgo de MB.” (Santiesteban y Pons 92)

“Si el rasgo analizado no se puede incorporar a la lista se prosigue el análisis con el siguiente rasgo de la matriz básica.” (Santiesteban y Pons 92)

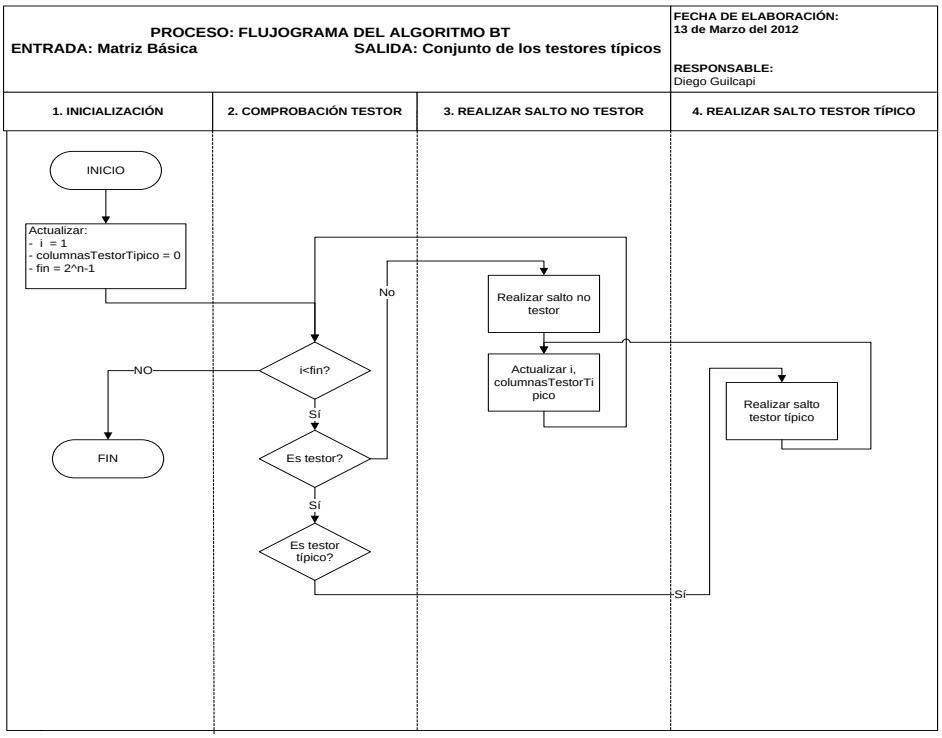
Cabe indicar que previamente se realiza un ordenamiento de la MB que consiste en colocar como primera fila aquella que tenga la menor cantidad de unos (ordenar filas), y en esta fila encontrada, modificar las columnas de tal modo que se sitúen todos los unos a la izquierda (ordenar columnas). Si existiera más de una fila con mínima cantidad de unos se puede utilizar el concepto de Entropía Global para la selección. Así, se escoge aquella fila que tiene la mayor entropía global, o sea, aquella en la que la suma de los unos presentes en las columnas donde esta fila tiene valores unitarios sea mayor. (Santiesteban y Pons 92).

2. CAPÍTULO II

2.1 METODOLOGÍA

2.1.1 Implementación de los Algoritmos BT y LEX

En el Anexo A (sección A.1) se presenta el código del Algoritmo BT utilizando el software R. La función que ejecuta dicho algoritmo es testores_tipicos. Dicho Algoritmo fue realizado por Paúl Méndez, Marcelo Almeida y Diego Guilcapi. A continuación se presentan un diagrama de flujo del funcionamiento del Algoritmo BT



Fuente: Generación Propia

Figura 1: Diagrama de flujo del Algoritmo BT

La salida de este algoritmo utilizando una función adicional para medir el tiempo de ejecución del algoritmo (system.time) para las siguientes matrices: $Q_1 = \varphi(I_4, I_4)$ y $Q_2 = \varphi(\theta(I_2, I_2), \theta(I_2, I_2))$

$$Q_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \quad Q_2 = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

es la siguiente:

```
> system.time(testores_tipicos(Q1))
[1] "Total de listas: "
[1] 255
[1] "Listas examinadas: "
[1] 106
[1] "Porcentaje de listas que fueron examinadas: "
[1] 41.57
[1] "Total de saltos testor: "
[1] 40
[1] "Total de saltos no testor: "
[1] 109
[1] "Numero de Testores tipicos: "
[1] 16
[1] "Conjunto de Testores: "
[1] 0 0 0 0 1 1 1 1
[1] 0 0 0 1 1 1 1 0
[1] 0 0 1 0 1 1 0 1
[1] 0 0 1 1 1 1 0 0
[1] 0 1 0 0 1 0 1 1
[1] 0 1 0 1 1 0 1 0
[1] 0 1 1 0 1 0 0 1
[1] 0 1 1 1 1 0 0 0
[1] 1 0 0 0 0 1 1 1
[1] 1 0 0 1 0 1 1 0
[1] 1 0 1 0 0 1 0 1
[1] 1 0 1 1 0 1 0 0
[1] 1 1 0 0 0 0 1 1
[1] 1 1 0 1 0 0 1 0
[1] 1 1 1 0 0 0 0 1
[1] 1 1 1 1 0 0 0 0
   user  system elapsed
   0.07    0.00    0.08
```

```
> system.time(testores_tipicos(Q2))
[1] "Total de listas: "
[1] 255
[1] "Listas examinadas: "
[1] 79
[1] "Porcentaje de listas que fueron examinadas: "
[1] 30.98
[1] "Total de saltos testor: "
[1] 163
[1] "Total de saltos no testor: "
[1] 13
[1] "Numero de Testores tipicos: "
[1] 8
[1] "Conjunto de Testores: "
[1] 0 0 0 0 0 0 1 1
[1] 0 0 0 0 1 1 0 0
[1] 0 0 0 1 0 0 1 0
[1] 0 0 1 0 0 0 0 1
[1] 0 0 1 1 0 0 0 0
[1] 0 1 0 0 1 0 0 0
[1] 1 0 0 0 0 1 0 0
[1] 1 1 0 0 0 0 0 0
   user  system elapsed
   0.04    0.00    0.04
```

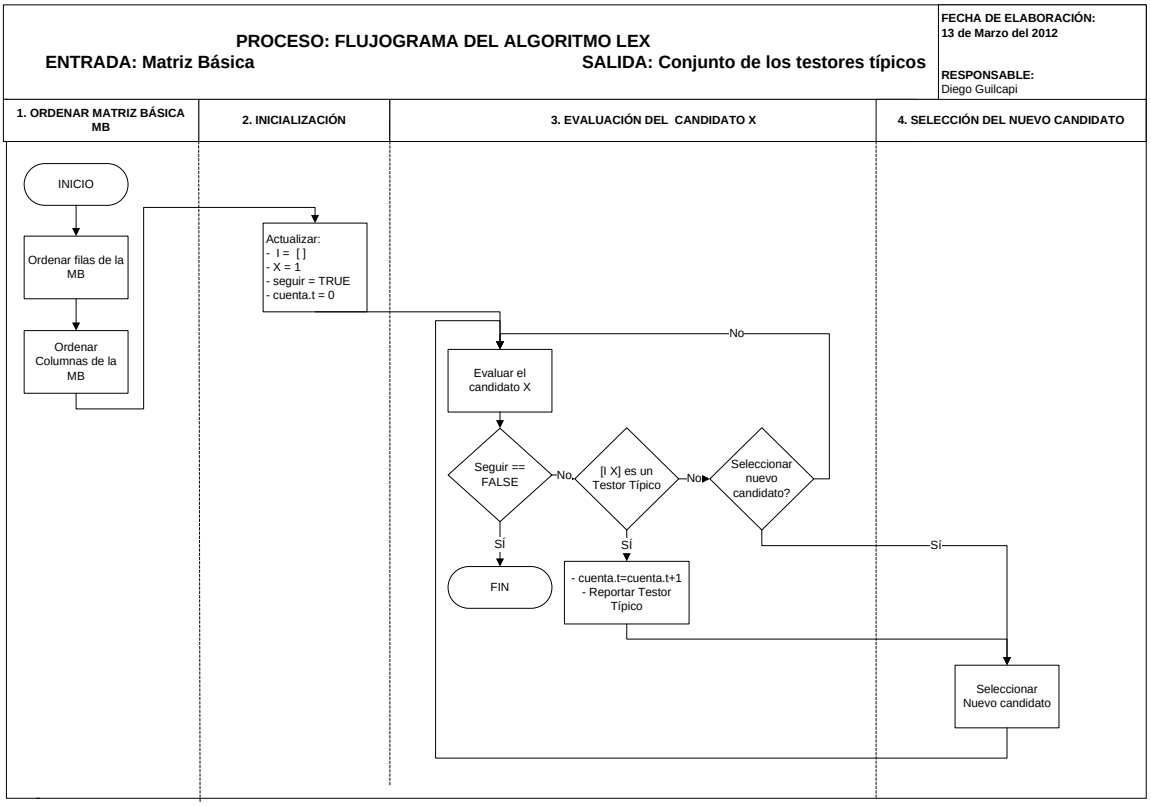
Donde la última línea (tercera columna) de la pantalla informa el tiempo de ejecución. Para este caso el valor es 0.08 segundos para $Q_1 = \varphi(I_4, I_4)$ y 0.04 segundos para $Q_2 = \varphi(\theta(I_2, I_2), \theta(I_2, I_2))$. Por otro lado, en la sección A.2 se presenta el código del Algoritmo LEX utilizando el software R. La función que ejecuta este algoritmo se llama: reportatt. La salida de este algoritmo utilizando una función adicional para medir el tiempo de ejecución del algoritmo (system.time) para las mismas matrices Q_1 y Q_2 es:

```
system.time(reportatt(Q1))
 user  system elapsed
 0.03    0.00    0.03
system.time(reportatt(Q2))
 user  system elapsed
 0.01    0.00    0.01
```

Los tiempos en segundos obtenidos son 0.03 y 0.01 respectivamente. Note que estos tiempos son ligeramente diferentes a los que se reportan en las tablas que se presentarán más adelante debido a que en este caso se utilizó un computador con procesador Intel(R) Core (TM) 2 CPU T5200; mientras que en la experimentación se utilizaron dos computadores: Un computador con procesador Quad-Core AMP Opteron (tm) (64 bites) para los cálculo del tiempo en segundos

que emplea el algoritmo BT. Y un computador con procesador Intel (R) Core(TM) i7-26 para el cálculo del tiempo en segundos que emplea el algoritmo LEX.

Un diagrama de flujo del Algoritmo LEX se presenta enseguida:



Fuente: Generación Propia

Figura 2: Diagrama de flujo del Algoritmo LEX

2.1.2 Generación y Empleo de Matrices de Prueba

Utilizando los resultados de la sección 1.1.3 se generarán matrices para evaluar el desempeño del cálculo de los testores típicos, partiendo de dos categorías de matrices establecidas por Alba-Santana 4. Estas categorías son:

- Matrices con la misma dimensión y diferente número de testores típicos
- Matrices con diferente dimensión e igual número de testores típicos

Dichas matrices se emplearán para medir el desempeño de los algoritmos BT y LEX

En seguida se presenta la construcción de tales matrices y el tiempo de ejecución respectivo (en segundos) que los algoritmos BT y LEX emplean en calcular todos los testores típicos de las matrices mencionadas.

2.1.2.1 Matrices con la misma dimensión y diferente número de testores típicos

Para estudiar el desempeño de los algoritmos cuando *únicamente* el número de testores típicos aumenta, se generan dos matrices ($Q_1\ y\ Q_2$) con la misma dimensión y una diferencia marcada en el número total de testores típicos. Estas matrices se construyen cumpliendo dos condiciones:

1. $\dim(Q_1) = \dim(Q_2)$
2. $\left|\Psi^*(Q_1)\right| \neq \left|\Psi^*(Q_2)\right|$

(Alba y Santana 5)

Para crear $Q_1\ y\ Q_2$, se empieza por dos matrices simples $B_1\ y\ B_2$ con la misma dimensión y con distinto número de testores típicos; luego se aplica el operador φ a cada matriz *N* veces obteniendo $Q_1\ y\ Q_2$.

De esta forma, sea $B_1 = I_4$ y $B_2 = \theta(I_2, I_2)$ entonces:

$$Q_1 = \varphi\left(\underbrace{B_1, B_1, \dots, B_1}_{N\ veces}\right)$$

$$Q_2 = \varphi\left(\underbrace{B_2, B_2, \dots, B_2}_{N\ veces}\right)$$

Donde:

$\dim(Q_1) = \dim(Q_2) = 4 \times 4 * N$ *(Existen 4 filas que se mantienen y al concatenarse N matrices se forma 4*N columnas)*

$\left|\Psi^*(Q_1)\right| = N^4$ (La cardinalidad del testor típico es 4 y existen N matrices iguales, con lo que se llega a N^4)

$\left|\Psi^*(Q_2)\right| = 2N^2$ (B_2 posee 2 testores típicos cada uno con cardinalidad 2, con lo que al concatenarse B_2 , *N* veces, se tiene $N^2 + N^2$)

$\left|\Psi^*(Q_1)\right| \neq \left|\Psi^*(Q_2)\right|$

(Alba y Santana 5)

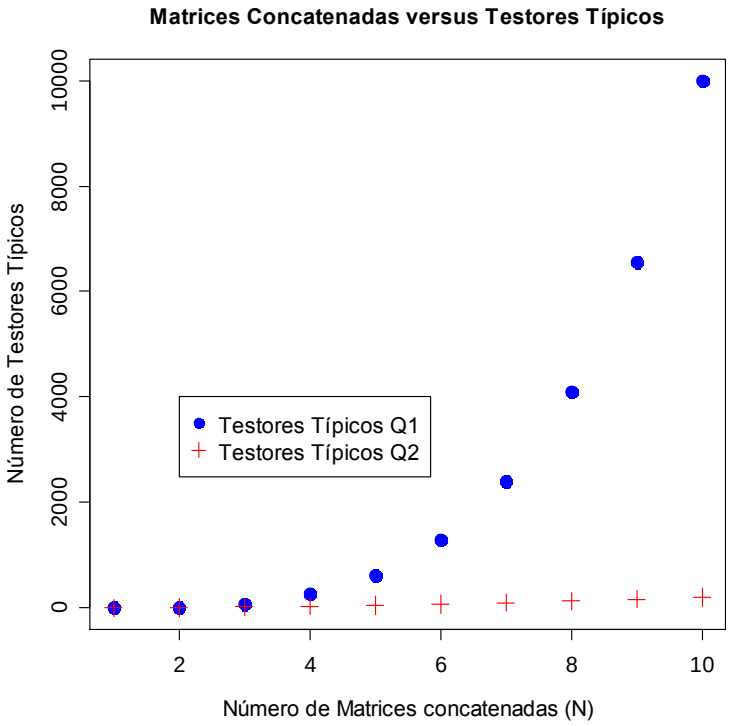
A continuación se destaca el número de testores típicos y la relación porcentual resultantes al variar el número de matrices concatenadas (N) para este ejemplo:

Tabla 1: Número de Testores Típicos resultantes al variar N

# de matrices concatenadas: N	**Dimensión:** $4 \times 4 * N$	**Número de t. típicos Q_1:** $\lvert\Psi^*(Q_1)\rvert = N^4$	**Número de t. típicos Q_2:** $\lvert\Psi^*(Q_2)\rvert = 2 * N^2$	$\frac{\text{Número de t. típicos } Q_2}{\text{Número de t. típicos } Q_1} * 100\%$
1	4×4	1	2	200%
2	4×8	$2^4 = 16$	$2(2^2) = 8$	50%
3	4×12	81	18	22%
4	4×16	256	32	12.5%
5	4×20	625	50	8%
6	4×24	1296	72	5.5%
7	4×28	2401	98	4.08%
8	4×32	4096	128	3.13%
9	4×36	6561	162	2.47%
10	4×40	10000	200	2%

Fuente: Generación Propia

La siguiente figura indica el crecimiento del número de testores típicos de las matrices Q_1 y Q_2 al aumentar el número de matrices concatenadas.



Fuente: Generación Propia

Figura 3: Variación del Número de Testores Típicos cuando incrementa el número de matrices concatenadas

Note que debido a la estructura de Q_2, la diferencia en el número de Testores Típicos (TT) de estas matrices, aumenta en una forma cuadrática. Como $Q_2 = \varphi\left(\underbrace{\theta(I_2, I_2), \theta(I_2, I_2), \ldots, \theta(I_2, I_2)}_{N\ veces}\right)$ se observa que el operador θ ayuda a incrementar el tamaño de la matriz, pero no cambia la cardinalidad de sus testores típicos. Esto significa que cuando se aplica el operador φ a I_4 y $\theta(I_2, I_2)$ respectivamente, el número de TT de las matrices resultantes estará influenciado por la cardinalidad de los TT de las matrices involucradas en generar Q_1 y Q_2. Por lo tanto,

$$\left|\Psi^*(Q_1)\right| > \left|\Psi^*(Q_2)\right| \textit{ para N>1}$$

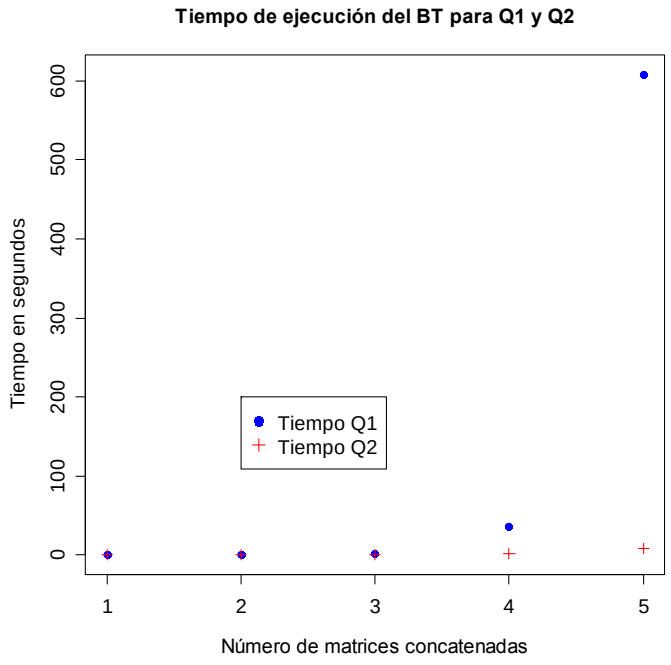
Partiendo de la Tabla 1 y la Figura 1, el tiempo que emplea el algoritmo BT se presenta enseguida:

Tabla 2: Tiempo de ejecución del BT para las matrices de la Tabla 1

N	**Dimensión:** $4 \times 4 * N$	$\lvert\Psi^*(Q_1)\rvert = N^4$	**Tiempo Seg.**	$\lvert\Psi^*(Q_2)\rvert = 2 * N^2$	**Tiempo Seg.**	$\frac{\text{Número de t. típico}}{\text{Número de t. típico}} * 100\%$
1	4×4	1	0.01	2	0.01	200%
2	4×8	$2^4 = 16$	0.06	$2(2^2) = 8$	0.05	50%
3	4×12	81	1.44	18	0.25	22%
4	4×16	256	36.41	32	1.44	12.5%
5	4×20	625	608.27	50	7.82	8%
6	4×24	1296	8425.47	72	39.43	5.5%
7	4×28	2401	99737.15	98	199.81	4.08%
8	4×32	4096	> 99737.15	128	826.56	3.13%
9	4×36	6561	> 99737.15	162	3617.35	2.47%
10	4×40	10000	> 99737.15	200	15821.01	2%

Fuente: Generación Propia

La siguiente figura indica el tiempo de dicho algoritmo en calcular todos los testores típicos de las matrices de la tabla anterior.



Fuente: Generación Propia

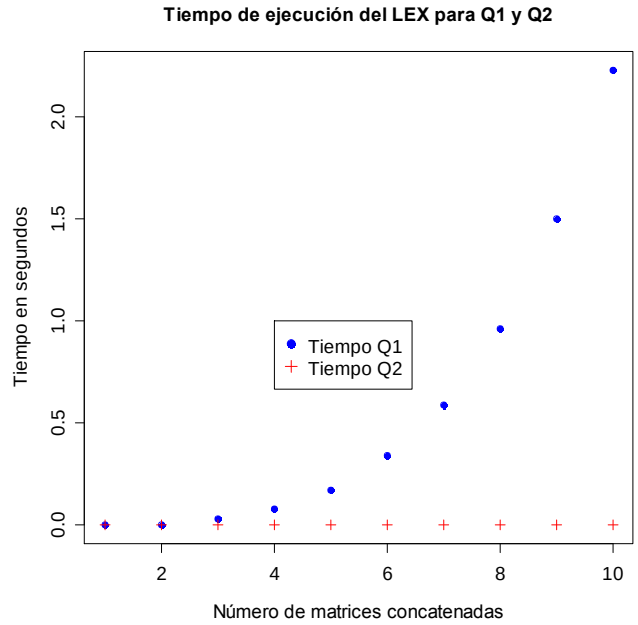
Figura 4: Tiempo de ejecución del BT para $\boldsymbol{Q_1}$ y $\boldsymbol{Q_2}$ al incrementar el número de matrices concatenadas

Observe que el BT emplea mayor cantidad de tiempo para calcular los TT de las matrices tipo $\boldsymbol{Q_1}$. De esta misma manera, se presenta el tiempo que emplea el algoritmo LEX:

Tabla 3: Tiempo de ejecución del LEX para las matrices de la Tabla 1

N	**Dimensión:** $4 \times 4 * N$	$\lvert\Psi^*(Q_1)\rvert = N^4$	**Tiempo Seg.**	$\lvert\Psi^*(Q_2)\rvert = 2 * N^2$	**Tiempo Seg.**	$\frac{\text{Número de t. típico}}{\text{Número de t. típico}} * 100\%$
1	4×4	1	0.00	2	0.00	200%
2	4×8	$2^4 = 16$	0.00	$2(2^2) = 8$	0.00	50%
3	4×12	81	0.03	18	0.00	22%
4	4×16	256	0.08	32	0.00	12.5%
5	4×20	625	0.17	50	0.00	8%
6	4×24	1296	0.34	72	0.00	5.5%
7	4×28	2401	0.59	98	0.00	4.08%
8	4×32	4096	0.96	128	0.00	3.13%
9	4×36	6561	1.50	162	0.00	2.47%
10	4×40	10000	2.23	200	0.00	2%

Fuente: Generación Propia



Fuente: Generación Propia

Figura 5: Tiempo de ejecución del LEX para $\boldsymbol{Q_1}$ y $\boldsymbol{Q_2}$ al incrementar el número de matrices concatenadas

Se puede apreciar que asimismo el Algoritmo LEX emplea mayor cantidad de tiempo para calcular los TT de las matrices tipo $\boldsymbol{Q_1}$.

Al comparar el desempeño de ambos algoritmos (BT y LEX), se observa que el Algoritmo LEX requiere de menor cantidad de tiempo para calcular todos los TT. Por ejemplo, requiere únicamente 2.23 segundos para calcular el tiempo de una matriz que cuenta con 10000 testores, versus los más de 99737.15 segundos que requiere el BT.

Hasta este momento, se han construido dos tipos de matrices con igual tamaño y diferente número de TT; sin embargo, se puede generar 46 matrices adicionales con estas características. Veinte y tres (4!-1) a partir de $B_1 = I_4$ mediante las trasposiciones y permutaciones de las columnas de dicha matriz (Ver Figura 4), y luego aplicando el operador concatenación. Y de la misma forma, 23 distintas matrices a partir de $B_2 = \theta(I_2, I_2)$, y después aplicando el operador concatenación. Esta clase de matrices tienen el mismo número de TT y el mismo tamaño que Q_1 y Q_2 pero lucen diferentes.

Observe la siguiente figura que presenta una transposición de B_1 y de B_2

a) B1: Matriz identidad de orden 4 (I_4)

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

c) B_2

1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	1

b) Intercambio de las dos primeras columnas de I_4

0	1	0	0
1	0	0	0
0	0	1	0
0	0	0	1

d) Intercambio de las dos primeras columnas de B_2

0	1	0	0
0	1	0	0
1	0	1	0
1	0	0	1

Fuente: Generación Propia

Figura 6: Ejemplo de una transposición de la matriz $\boldsymbol{B_1}$ (literales a y b) y Ejemplo de una transposición de la matriz $\boldsymbol{B_2}$ (literales c y d)

Cabe recalcar que el número de filas de Q_1 y Q_2 es 4. Pero existen algunos resultados que permiten formar matrices con más filas.

Sea n un número par ($n = 2,4,6, \dots$) se cumple que:

Proposición 4. $\dim(I_{n^2}) = \dim\left(\underbrace{\varphi(\theta(I_n, I_n), \theta(I_n, I_n), \dots, \theta(I_n, I_n)}_{\frac{n}{2} veces}\right) = n^2 \times n^2$

Demostración:

$\dim(I_{n^2}) = n^2 \times n^2$

$\dim\big(\theta(I_n, I_n)\big) = n * n \times n + n = n^2 \times 2n$

$$\dim\left(\underbrace{\varphi(\theta(I_n, I_n), \theta(I_n, I_n), \dots, \theta(I_n, I_n)}_{\frac{n}{2} veces}\right)$$

$$= n^2 \times \underbrace{2n + 2n + \cdots + 2n}_{\frac{n}{2} veces} = n^2 \times 2(n)(\tfrac{n}{2}) = n^2 \times n^2$$

$$lqqd.$$

Proposición 5. $\left|\Psi^*(I_{n^2})\right| \neq \left|\Psi^*\left(\underbrace{\varphi(\theta(I_n, I_n), \theta(I_n, I_n), \dots, \theta(I_n, I_n)}_{\frac{n}{2} veces}\right)\right|$

Demostración:

$\left|\Psi^*(I_{n^2})\right| = 1$

$$\left|\Psi^*\left(\underbrace{\varphi(\theta(I_n, I_n), \theta(I_n, I_n), \dots, \theta(I_n, I_n)}_{\frac{n}{2} veces}\right)\right| = \frac{n^n}{2^{n-1}}$$

Porque $\theta(I_n, I_n)$ posee 2 testores típicos cada uno con cardinalidad n, con lo que al concatenarse $\theta(I_n, I_n)$, $\frac{n}{2}$ veces, se tiene $\left[\frac{n}{2}\right]^n + \left[\frac{n}{2}\right]^n = 2\left[\frac{n}{2}\right]^n = \frac{2n^n}{2^n} = \frac{n^n}{2^{n-1}}$

$$lqqd.$$

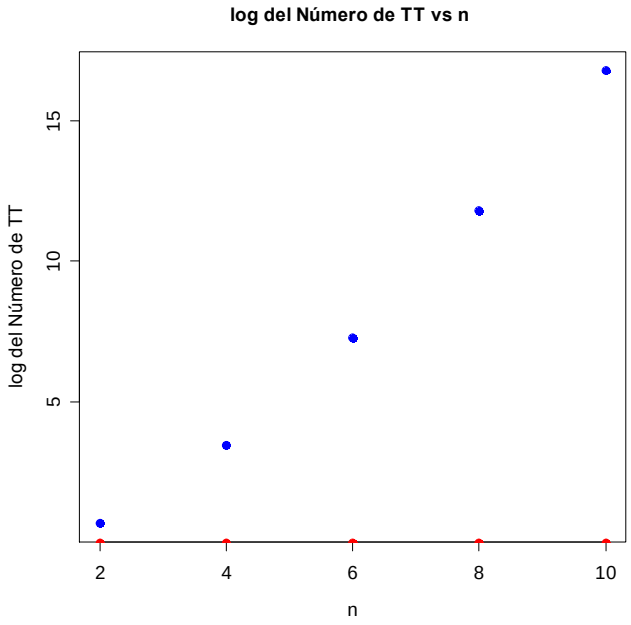
Con ello, en la siguiente tabla se presenta la dimensión de la matriz, el número de testores típicos para $P_1 = I_{n^2}$ y $P_2 = \underbrace{\varphi(\theta(I_n, I_n), \theta(I_n, I_n), \dots, \theta(I_n, I_n)}_{\frac{n}{2} veces}$ y la relación porcentual resultantes al variar n de 2 hasta 14.

Tabla 4: Número de Testores Típicos resultantes al variar n

Número par: n	**Dimensión:** $n^2 \times n^2$	**Número de t. típicos P_1:** $\lvert\Psi^*(P_1)\rvert = 1$	**Número de t. típicos de P_2:** $\lvert\Psi^*(P_2)\rvert = 2\left[\frac{n}{2}\right]^n$	**Relación Porcentual:** $\frac{\textbf{Número de t. típicos } P_1}{\textbf{Número de t. típicos } P_2} * 100\%$
2	4×4	1	2	50 %
4	16×16	1	32	3.125 %
6	36×36	1	1458	0.068587 %

Fuente: Generación Propia

Note que adicionalmente existe una diferencia más marcada en el número de testores típicos (observe la relación porcentual de las Tablas 1 y 3, y la siguiente figura). Por lo tanto se utilizó el logaritmo del número de TT para mejorar su ilustración



Fuente: Generación Propia

Figura 7: Logaritmo del Número de TT versus n

Adicionalmente, se presentan los tiempos de las matrices de dicha tabla:

Tabla 5: Tiempo de ejecución del BT para las matrices de la Tabla 4

n	**Dim.** $n^2 \times n^2$	**Número de t. típicos** P_1**:** $\lvert\Psi^*(P_1)\rvert = 1$	**Tiempo (seg)**	**Número de t. típicos de** P_2**:** $\lvert\Psi^*(P_2)\rvert = 2\left[\frac{n}{2}\right]^n$	**Tiempo (seg)**	**Relación Porcentual:**
2	4×4	1	0.02	2	0.03	50 %
4	16×16	1	0.02	32	19.85	3.125 %
6	36×36	1	0.09	1458	>100 h.	0.068587 %

Fuente: Generación Propia

Tabla 6: Tiempo de ejecución del LEX para las matrices de la Tabla 4

n	**Dim.** $n^2 \times n^2$	**Número de t. típicos** P_1**:** $\lvert\Psi^*(P_1)\rvert = 1$	**Tiempo (seg)**	**Número de t. típicos de** P_2**:** $\lvert\Psi^*(P_2)\rvert = 2\left[\frac{n}{2}\right]^n$	**Tiempo (seg)**	**Relación Porcentual:**
2	4×4	1	0.00	2	0.00	50 %
4	16×16	1	0.00	32	0.04	3.125 %
6	36×36	1	0.00	1458	8.5	0.068587 %

Fuente: Generación Propia

Claramente puede observarse que el tiempo del Algoritmo LEX es menor que el tiempo del Algoritmo BT en todos los casos, destacándose el gran desempeño del LEX (8.5 segundos) frente al BT (más de 100 horas) para la matriz de 36×36 y 1458 TT.

Tomando en cuenta los resultados anteriores se pueden construir nuevas matrices con esta característica (gran diferencia en la cantidad de sus testores típicos). Para ello sean las matrices anteriormente mencionadas: $P_1 = I_{n^2}$ y $P_2 =$

$\underbrace{\varphi(\theta(I_n, I_n), \theta(I_n, I_n), \dots, \theta(I_n, I_n)}_{\frac{n}{2} veces}$. Ahora, se aplicará el operador concatenación a cada una de ellas obteniéndose las matrices:

$Q_3 = \underbrace{\varphi(P_1, \dots, P_1)}_{N\ veces} = \underbrace{\varphi(I_{n^2}, \dots, I_{n^2})}_{N\ veces}$ y

$$Q_4 = \underbrace{\varphi(P_2, \dots, P_2)}_{N\ veces} = \varphi[\underbrace{\underbrace{\varphi\big(\theta(I_n, I_n), \theta(I_n, I_n), \dots, \theta(I_n, I_n)\big)}_{\frac{n}{2} veces}, \dots, \underbrace{\varphi(\theta(I_n, I_n), \theta(I_n, I_n), \dots, \theta(I_n, I_n))}_{\frac{n}{2} veces}}_{N\ veces}$$

Donde:

$\dim(Q_3) = \dim(Q_4) = n^2 \times n^2 * N$ *(Existen* n^2 *filas que se mantienen y al concatenarse N matrices se forma* $n^2 * N$ *columnas)*

$\left|\Psi^*(Q_3)\right| = N^{n^2}$ (La cardinalidad del testor típico es n^2 y existen N matrices iguales, con lo que se llega a N^{n^2})

$\left|\Psi^*(Q_4)\right| = 2N^n$ (P_2 posee 2 testores típicos cada uno con cardinalidad n, con lo que al concatenarse P_2 , *N* veces, se tiene $N^n + N^n$)

De aquí que:

$\left|\Psi^*(Q_3)\right| \neq \left|\Psi^*(Q_4)\right|$

En seguida se presenta el número de testores típicos resultantes al variar el número de matrices concatenadas (N) para este ejemplo:

Tabla 7: Número de Testores Típicos resultantes al variar ***N***

# de matrices concatenadas: N	**Dimensión:** $n^2 \times n^2 * N$	**Número de t. típicos** Q_3**:** $\|\Psi^*(Q_3)\| = N^{n^2}$	**Número de t. típicos** Q_4**:** $\|\Psi^*(Q_4)\| = 2 * N^n$
1	$n^2 \times n^2 * 1$	1	2
2	$n^2 \times n^2 * 2$	2^{n^2}	$2 * 2^n$
3	$n^2 \times n^2 * 3$	3^{n^2}	$2 * 3^n$
4	$n^2 \times n^2 * 4$	4^{n^2}	$2 * 4^n$
5	$n^2 \times n^2 * 5$	5^{n^2}	$2 * 5^n$
6	$n^2 \times n^2 * 6$	6^{n^2}	$2 * 6^n$
7	$n^2 \times n^2 * 7$	7^{n^2}	$2 * 7^n$

Fuente: Generación Propia

2.1.2.2 Matrices con diferente dimensión e igual número de testores típicos

En este caso se estudia el desempeño del Algoritmo LEX cuando el número de testores típicos es el mismo, pero las matrices poseen diferente dimensión (Alba y Santana 5).

Así, estas matrices, denotadas como $Q_1 \ y \ Q_2$, deben satisfacer:

1. $\dim(Q_1) \neq \dim(Q_2)$
2. $\left|\Psi^*(Q_1)\right| = \left|\Psi^*(Q_2)\right|$

Según Alba y Santana 5, para crear dichas matrices se empieza escogiendo dos matrices identidad I_{n_1} y I_{n_2}. Posteriormente, se aplica el operador φ a ambas matrices N_1 y N_2 veces respectivamente"

Para cumplir con ambas condiciones es necesario que $n_1 \neq n_2$ y $N_1^{n_1} = N_2^{n_2}$

Las matrices tienen la forma: $Q_1 = \varphi(\underbrace{I_{n_1}, I_{n_1}, \dots, I_{n_1}}_{N_1 \ veces})$ y $Q_2 = \varphi(\underbrace{I_{n_2}, I_{n_2}, \dots, I_{n_2}}_{N_2 \ veces})$ con $n_1, n_2 \geq 2$

Ejemplo 1:

$n_1 = 2, n_2 = 4, \ N_1 = 9, \ N_2 = 3$ (Alba y Santana 5)

$Q_1 = \varphi(\underbrace{I_2, I_2, \dots, I_2}_{9 \ veces})$ con $\dim(Q_1) = 2 \times 18$ y $\left|\Psi^*(Q_1)\right| = 9^2 = 81$

$Q_2 = \varphi(\underbrace{I_4, I_4, I_4}_{3 \ veces})$ con $\dim(Q_2) = 4 \times 12$ y $\left|\Psi^*(Q_2)\right| = 3^4 = 81$

Ejemplo 2:

$n_1 = 3, n_2 = 6, \ N_1 = 4, \ N_2 = 2$

$Q_1 = \varphi(\underbrace{I_3, I_3, \dots, I_3}_{4 \ veces})$ con $\dim(Q_1) = 3 \times 12$ y $\left|\Psi^*(Q_1)\right| = 4^3 = 64$

$Q_2 = \varphi(\underbrace{I_6, I_6}_{2 \ veces})$ con $\dim(Q_2) = 6 \times 12$ y $\left|\Psi^*(Q_2)\right| = 2^6 = 64$

Al observar estos ejemplos, se puede destacar que si:

$n_2 = 2 * n_1 \ \ (n_1 \neq n_2)$ y $\ N_1 = N_2^2$

Se cumple que: $\ N_1^{n_1} = (N_2^2)^{\,n_1} = \ \ (N_2^2)^{\frac{n_2}{2}} = N_2^{n_2}$

Por lo tanto se puede construir una variedad de matrices que satisfagan dichas condiciones. En la siguiente tabla se ilustra su forma de construcción.

Tabla 8: Matrices de prueba con igual número de testores típicos y diferente dimensión, para diferentes valores de $\boldsymbol{n_1, n_2, N_1}$ $\boldsymbol{y}$ $\boldsymbol{N_2}$

n_1	n_2	N_1	N_2	$N_1^{n_1} = N_2^{n_2}$	$\dim(Q_1) = n_1 \times n_1 * N_1$	$\dim(Q_2) = n_2 \times n_2 * N_2$	$\lvert\Psi^*(Q_1)\rvert = N_1^{n_1}$	$\lvert\Psi^*(Q_2)\rvert = N_2^{n_2}$
2	4	4	2	$4^2 = 2^4$	2×8	4×8	4^2	2^4
2	4	9	3	$9^2 = 3^4$	2×18	4×12	9^2	3^4
2	4	16	4	$16^2 = 4^4$	2×32	4×16	16^2	4^4
2	4	25	5	$25^2 = 5^4$	2×50	4×20	25^2	5^4
2	4	36	6	$36^2 = 6^4$	2×72	4×24	36^2	6^4

Fuente: Generación Propia

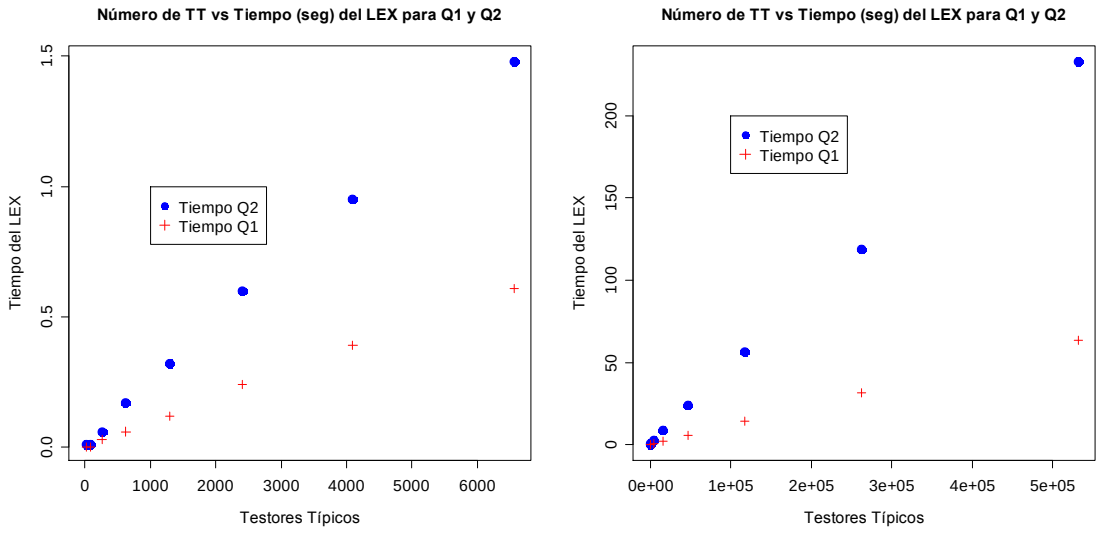
Utilizando el Algoritmo LEX se presentan una tabla y dos gráficas que ilustran el tiempo que emplea dicho algoritmo para matrices de este estilo:

Tabla 9: Tiempo que emplea el Algoritmo LEX en calcular los TT de matrices de prueba con igual número de testores típicos y diferente dimensión, para diferentes valores de $\boldsymbol{n_1, n_2, N_1}$ $\boldsymbol{y}$ $\boldsymbol{N_2}$

				dim(Q1) = n1 x n1*N1			dim(Q2) = n2 x n2*N2						
n1	n2	N1	N2	n1	x	n1*N1	n2	x	n2*N2	Test. Típicos Q1	Test. Típicos Q2	Tiempo LEX Q1	Tiempo LEX Q2
2	4	4	2	2	x	8	4	x	8	16	16	0	0.01
2	4	9	3	2	x	18	4	x	12	81	81	0	0.01
2	4	16	4	2	x	32	4	x	16	256	256	0.03	0.06
2	4	25	5	2	x	50	4	x	20	625	625	0.06	0.17
2	4	36	6	2	x	72	4	x	24	1296	1296	0.12	0.32
2	4	49	7	2	x	98	4	x	28	2401	2401	0.24	0.6
2	4	64	8	2	x	128	4	x	32	4096	4096	0.39	0.95
2	4	81	9	2	x	162	4	x	36	6561	6561	0.61	1.48

				dim(Q1) = n1 x n1*N1			dim(Q2) = n2 x n2*N2						
n1	n2	N1	N2	n1	x	n1*N1	n2	x	n2*N2	Test. Típicos Q1 y Q2	Test. Típicos Q2	Tiempo LEX Q1	Tiempo LEX Q2
3	6	4	2	3	x	12	6	x	12	64	64	0.02	0.08
3	6	9	3	3	x	27	6	x	18	729	729	0.11	0.56
3	6	16	4	3	x	48	6	x	24	4096	4096	0.54	2.58
3	6	25	5	3	x	75	6	x	30	15625	15625	2	8.98
3	6	36	6	3	x	108	6	x	36	46656	46656	5.79	23.81
3	6	49	7	3	x	147	6	x	42	117649	117649	14.39	56.29
3	6	64	8	3	x	192	6	x	48	262144	262144	31.7	118.98
3	6	81	9	3	x	243	6	x	54	531441	531441	63.67	232.99

Fuente: Generación Propia



Fuente: Generación Propia

Figura 8: Tiempo de ejecución del LEX para $\boldsymbol{Q_1}$ y $\boldsymbol{Q_2}$ al incrementar el número de Testores Típicos

Observe que el tiempo para las matrices Q_2 es mayor que el tiempo para las matrices Q_1. Note nuevamente que el número de filas de Q_2 es el doble que el número de filas de Q_1. Y el número columnas de Q_1 es mayor al de Q_2. Por lo tanto, concatenar menos matrices de un orden mayor frente a concatenar más matrices de un orden menor para obtener matrices con el mismo número de TT, ocasiona un crecimiento en el tiempo para encontrar todos los TT.

Otra manera de construir matrices con el mismo número de testores típicos, pero diferente dimensión es tomar en cuenta que las matrices identidad de orden 2,

3, 4,… poseen dimensiones diferentes e igual número de testores típicos. De ese modo, al aplicar el operador fusión combinatoria n veces se obtiene matrices con el mismo número de testores típicos ($desde\ 1\ hasta\ n$) y diferente dimensión. La siguiente tabla ilustra este procedimiento $para\ k = 2,3,\ldots$ donde k es el orden de la matriz.

Tabla 10: Matrices de prueba con igual número de testores típicos y diferente dimensión, generadas a partir de una matriz identidad cuyo orden varía.

$Dimensión\ de\ la\ matriz\ I_k$	n	$Dimensión\ de\ la\ matriz\ \theta(\underbrace{I_k, \ldots, I_k}_{n\ veces})$
$k \times k$	1	$k^n \times n * k = k \times k$
$k \times k$	2	$k^2 \times 2 * k$
$k \times k$	3	$k^3 \times 3 * k$
$k \times k$	4	$k^4 \times 4 * k$
⋮	⋮	⋮

Fuente: Generación Propia

* Note que se forman n grupos que pueden ser analizados individualmente.

La siguiente tabla ilustra este método para $n = 1$ y todos los posibles valores de k

Tabla 11: Matrices de prueba con igual número de testores típicos y diferente dimensión, generadas a partir de una matriz identidad cuyo orden varía

k: orden de la matriz	$\dim(\mathrm{I_k}) = k \times k$	$\left\|\Psi^*(I_k)\right\| = 1$
2	2×2	1
3	3×3	1
4	4×4	1
5	5×5	1
6	6×6	1

Fuente: Generación Propia

Es decir, se pueden generar matrices de un grupo de matrices I_k con $k = 2,3,\ldots$ que poseen diferente dimensión, pero igual número de testores típicos(1 en este caso). El Tiempo que emplea el LEX en calcular los TT de estas matrices se presenta enseguida:

Tabla 12: Tiempo del LEX para las Matrices de prueba con igual número de testores típicos y diferente dimensión, generadas a partir de una matriz identidad cuyo orden varía

			Dimensión de la matriz = O^n x O*n				
	O: Orden de la matriz	n: número de matrices	O^n	x	O*n	Test. Típicos	Tiempo (seg) del LEX I_k
1	2	1	2	x	2	1	0
2	3	1	3	x	3	1	0
3	4	1	4	x	4	1	0
4	5	1	5	x	5	1	0
5	6	1	6	x	6	1	0
6	7	1	7	x	7	1	0
7	8	1	8	x	8	1	0.02
8	9	1	9	x	9	1	0
9	10	1	10	x	10	1	0
10	11	1	11	x	11	1	0

Fuente: Generación Propia

Observe que el tiempo es insignificante aún para la matriz Identidad de 11×11. La siguiente tabla ilustra este método para $n = 2$ y todos los posibles valores de k

Tabla 13: Matrices de prueba con igual número de testores típicos y diferente dimensión, generadas a partir de una matriz identidad y el operador $\boldsymbol{\theta}$

k: orden de la matriz	$\dim(\theta(I_k, I_k))$ $= k^2 \times 2 * k$	$\lvert\Psi^*(\theta(I_k, I_k))\rvert = 2$
2	4×4	2
3	9×6	2
4	16×8	2
5	25×10	2
6	36×12	2

Fuente: Generación Propia

Utilizando nuevamente el Algoritmo LEX se presentan dos tablas que ilustran el tiempo que emplea dicho algoritmo para matrices de este estilo, considerando $n = 2$ y $n = 3$.

Tabla 14: Matrices de prueba con igual número de testores típicos y diferente dimensión, generadas a partir de una matriz identidad y el operador $\boldsymbol{\theta}$

			Dimensión de la matriz				
	O: Orden de la matriz	n: número de matrices	O^n	x	O*n	Test. Típicos	Tiempo (seg) del LEX
1	2	2	4	x	4	2	0
2	3	2	9	x	6	2	0
3	4	2	16	x	8	2	0
4	5	2	25	x	10	2	0.02
5	6	2	36	x	12	2	0.04
6	7	2	49	x	14	2	0.14
7	8	2	64	x	16	2	0.43
8	9	2	81	x	18	2	1.28
9	10	2	100	x	20	2	3.65
10	11	2	121	x	22	2	10.22

			Dimensión de la matriz				
	O: Orden de la matriz	n: número de matrices	O^n	x	O*n	Test. Típicos	Tiempo (seg) del LEX
1	2	3	8	x	6	3	0
2	3	3	27	x	9	3	0
3	4	3	64	x	12	3	0.06
4	5	3	125	x	15	3	0.47
5	6	3	216	x	18	3	2.68
6	7	3	343	x	21	3	13.23
7	8	3	512	x	24	3	58.24
8	9	3	729	x	27	3	232.16
9	10	3	1000	x	30	3	7682.17
10	11	3	1331	x	33	3	>7682.17

Fuente: Generación Propia

El tiempo del LEX al aplicar el operador θ a matrices identidad cuyo orden va incrementándose, como se puede apreciar en la tabla. Sin embargo, observe que aún cuando el tamaño de la matriz es 1000×30 (relativamente grande) el tiempo de dicho algoritmo es 7682.17 segundos. Note la importancia del operador θ, ya que al ser aplicado en este caso permite mantener constante el número de TT.

Existe otro grupo de matrices con un número de testores típicos igual a 2, pero con dimensiones diferentes. Este conjunto se forma mediante: $\theta(I_v, I_w)$ donde $v\ y\ w$ pueden tomar valores a partir de 2. Esto se puede visualizar en la siguiente tabla, donde se fija el valor de v en 2.

Tabla 15: Matrices de prueba con igual número de testores típicos y diferente dimensión, generadas a partir de dos matrices identidad de órdenes diferentes y el operador $\boldsymbol{\theta}$

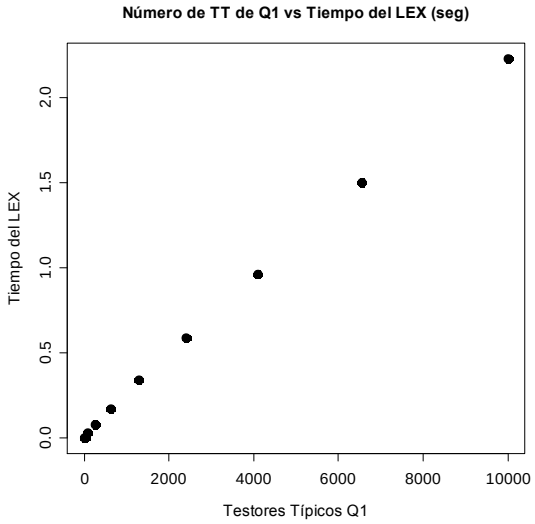
$Dimensión\ de\ la\ matriz\ I_v$	$Dimensión\ de\ la\ matriz\ I_w$	$\dim(\theta(I_v, I_w)) = v * w \times v + w$	$\lvert\Psi^*(\theta(I_k, I_k))\rvert = 2$
2×2	2×2	4×4	2
2×2	3×3	6×5	2
2×2	4×4	8×6	2
2×2	5×5	10×7	2
⋮	⋮	⋮	⋮

Fuente: Generación Propia

Se puede generalizar este método de construcción formando la siguiente matriz $\theta(I_{s_1}, I_{s_2}, \dots, I_{s_n})$ donde $s_i\ con\ i = 1,2,3, \dots, n$ adquiere valores desde 2. Incluso puede existir el caso en que $s_i = s_j$ para $i \neq j$. Su dimensión es: $\prod_{i=1}^{n} s_i \times \sum_{i=1}^{n} s_i$ y el número de testores típicos es n. Note que este proceso de construcción de matrices es el mismo que el propuesto por Eduardo Alba y Roberto Santana pág. 5; sin embargo en este caso las dimensiones de las matrices identidad empleadas poseen órdenes bajos y altos. El estudio del tiempo de ejecución del BT y el LEX con estas matrices no se lo realiza en este proyecto, pero da una oportunidad para hacerlo en futuros proyectos.

2.2 DISCUSIÓN Y ANÁLISIS DE RESULTADOS

Al graficar el número de TT para las diversas matrices Q_1, (revisar sección 2.1.2.1) que resultan al variar N versus el tiempo del LEX, se observa que existe una relación lineal. La siguiente gráfica resalta este resultado:



Fuente: Generación Propia

Figura 9: Número de Testores típicos vs Tiempo del LEX para las matrices $\boldsymbol{Q_1}$

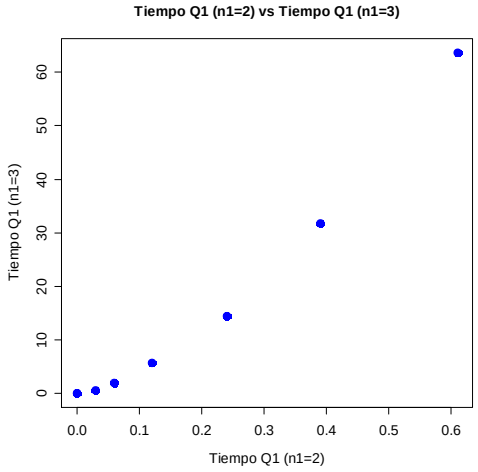
Aplicando el método de Mínimos cuadrados se llegó al siguiente ajuste que satisface dicha hipótesis:

$$\widehat{Tiempo\ Q_1}(TT) = 0.02457 + 0.000232TT$$

$$t: \quad (2.592) \quad (96.43)$$

Con un $R^2 = 0.99$ y un error estándar de 0.02355

Por otro lado, de las matrices generadas en la sección 2.1.2.2 se aprecia que al graficar el tiempo de Q_1 con $n_1 = 2$ versus el tiempo de Q_1 con $n_1 = 3$ se obtiene el siguiente resultado:



Fuente: Generación Propia

Figura 10: Tiempo de $\boldsymbol{Q_1}$ con $\boldsymbol{n_1 = 2}$ versus Tiempo de $\boldsymbol{Q_1}$ con $\boldsymbol{n_1 = 3}$

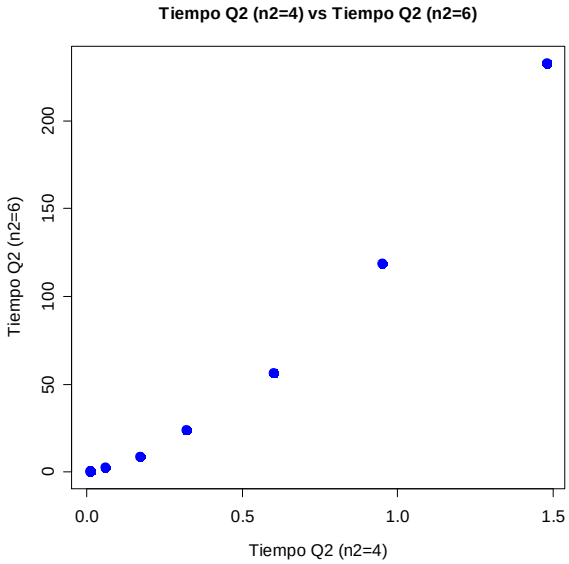
Al ajustar un modelo cuadrático se obtiene:

$$\widehat{Tiempo\ Q_1}(n_1 = 2) = -0.2583 + 36.042\ Tiempo\ Q_1(n_1 = 3) + 113.24 Tiempo\ Q_1(n_1 = 3)^2$$
$$t: \quad (-0.95) \quad (11.86) \qquad\qquad (22.21)$$

Con un $R^2 = 0.99$ y un error estándar de 0.04979

Por ello, existe una relación cuadrática entre estas dos variables. Se debe indicar que los residuos de este modelo siguen una distribución normal.

Adicionalmente, al graficar el tiempo de Q_2 con $n_2 = 4$ versus el tiempo de Q_2 con $n_2 = 6$ se obtiene el siguiente resultado:



Fuente: Generación Propia

Figura 11: Tiempo de $\boldsymbol{Q_2}$ con $\boldsymbol{n_2 = 4}$ versus Tiempo de $\boldsymbol{Q_2}$ con $\boldsymbol{n_2 = 6}$

Al ajustar un modelo cuadrático se obtiene:

$$\widehat{Tiempo\ Q_2}(n_2 = 6) = -0.1.270 + 58.767\ Tiempo\ Q_2(n_2 = 4) + 67.689\ Tiempo\ Q_2(n_2 = 4)^2$$
$$t: \quad (-1.072) \quad (11.212) \qquad\qquad (18.74)$$

Con un $R^2 = 0.99$ y un error estándar de 2.086

De ese modo, también existe una relación cuadrática entre estas dos variables. Se debe indicar que los residuos de este modelo siguen una distribución normal.

3. Conclusiones y Recomendaciones

Conclusiones:

- Se describieron nuevas formas para generar las siguientes matrices de prueba: Matrices con la misma dimensión y diferente número de testores típicos y Matrices con diferente dimensión e igual número de testores típicos (Ver sección 2.1.2).

- Se encontró que el operador θ, fusión combinatoria, permite incrementar el tamaño de la matriz, pero no modifica la cardinalidad de sus testores típicos (Ver sección 2.1.2.1). Por otro lado, dicho operador también permite mantener constante el número de TT (Ver sección 2.1.2.2).

- Se identificó que al combinar los operadores θ y φ (concatenación), se pueden generar diferentes tipos de matrices de prueba (Ver sección 2.1.2).

- Se logró estudiar el desempeño en el tiempo de los algoritmos de escala externa BT y LEX, partiendo de matrices de prueba, cuyo número de testores típicos es conocido previamente (Ver sección 2.1.2).

- Se determinó que para las matrices de prueba de la sección 2.1.2.1 el Algoritmo LEX tiene un mejor desempeño que el Algoritmo BT.

- Se concluyó que concatenar menos matrices de un orden mayor frente a concatenar más matrices de un orden menor para obtener matrices con el

mismo número de TT, ocasiona un crecimiento en el tiempo del Algoritmo LEX para encontrar todos los TT (Ver sección 2.1.2.2).

- Se encontró que existe una relación lineal entre el número de TT para las diversas matrices Q_1, (sección 2.2) versus el tiempo del LEX.

- Se identificó que existe una relación cuadrática entre el tiempo (empleando el LEX) de la matriz Q_1 con $n_1 = 2$ y el tiempo de Q_1 con $n_1 = 3$. Además, se halló que existe una relación cuadrática entre el Tiempo de Q_2 con $n_2 = 4$ y Tiempo de Q_2 con $\mathrm{n}_2 = 6$ (Ver sección 2.2).

Recomendaciones:

- Se recomienda realizar un proyecto que investigue cómo varía el desempeño en el cálculo de los testores típicos si se modifica el orden de las columnas de las matrices de prueba que se han generado en este documento. En la sección 2.1.2.1, Matrices con la misma dimensión y diferente número de testores típicos, se brinda una explicación más detallada del análisis que se podría desarrollar.

- Se invita a realizar un estudio del tiempo de ejecución del BT y del LEX con las matrices de la Tabla 15: Matrices de prueba con igual número de testores típicos y diferente dimensión, generadas a partir de dos matrices identidad de órdenes diferentes y el operador $\boldsymbol{\theta}$; de la sección 2.1.2.2

- Se sugiere programar el Algoritmo FI_CT_EXT y observar su desempeño con este nuevo conjunto de matrices de prueba.

BIBLIOGRAFÍA

Alba, E. y Santana R. 2010. Generación de matrices para evaluar el desempeño de estrategias de búsqueda de testores típicos. Revista Avances en Ciencias e Ingenierías ISSN 1390-5384**,** 2010. 18 de Enero del 2012. http://profesores.usfq.edu.ec/ealba/ProyectoMAT0270/testoresespaniol.pdf

Alba, E. Aplicación al programa de Chancellor Grants de la USFQ. Sección B: Antecedes y Justificación (Justificación e Importancia, Breve Resumen del Proyecto).

Almeida M., Guilcapi D. y Méndez P. Proyecto final de Matemáticas Discretas. 1er Semestre 2011-2012. USFQ. Diciembre del 2011.

Pons A., Santiesteban Y. LEX: A NEW ALGORITHM FOR THE CALCULUS OF ALL TYPICAL TESTORS. 18 Enero del 2012. http://www.cerpamid.co.cu/sitio/files/publicaciones/LEX.pdf

Sánchez, Piza, Lazo, Mora Miguel y Salinas Javier. A fast implementation of the CT EXT: Algorithm for the Testor Property Identification. G. Sidorov et al. (Eds): MICAI 2010, Parte ll LNAI 6438, PP. 92-103 2010. Springer – Verlag Berlin Heidelberg 2010.

Santiesteban, Y. y Pons A. LEX: UN NUEVO ALGORITMO PARA EL CÁLCULO DE LOS TESTORES TÍPICOS. Revista Ciencias Matemáticas. Vol. 21 N° 1, 2003.

The MathWorks, Inc. Programming: Concatenating Matrices. 1 de Febrero del 2012. http://matlab.izmiran.ru/help/techdoc/matlab_prog/ch10_pr4.html#85019.

ANEXOS

Anexo A: IMPLEMENTACIÓN DE LOS ALGORITMOS BT Y LEX

A.1. IMPLEMENTACIÓN DEL ALGORITMO BT

A.2. IMPLEMENTACIÓN DEL ALGORITMO LEX