Herramienta para el cálculo de invariantes topológicos

Juan Pablo Bedoya Giraldo
jpbedoya@sysgold.com

Aura Lucía Pérez Escobar
alperez@icesi.edu.co

Javier Guillermo Valdés Duque
jvald01@yahoo.es

Fecha de recepción: 15-12-2003 Fecha de aceptación: 21-4-2004

SUMMARY

In this article there is a description of the process that yielded a software tool which was designed with the purpose of calculating four specific topological invariants, among which we count the Betti numbers, the dimension of the complex, the q-array and the Euler-Poincaré characteristic.

The meaning of all the obtained results is viewed from Professor Luis Eduardo Múnera's perspective, which in turn proposes a formal mathematical definition of cohesion and coupling, two important criteria when evaluating modularity of a software design. The article also describes the tool itself, its functionality and the importance it has to the professor's project.

In order to portrait a full comprehensible description of both the process and the tool, an historical framework is established through a brief review of the evolution of the different ideas and concepts behind the project, and the way and pace with which these were assimilated by the student team that were appointed to the task.

KEY WORDS

Modular design, algebraic topology, topological invariants calculations, Betti numbers, complex dimension, Euler/Poincaré characteristic, q-array, cohesion, coupling, Matlab.

RESUMEN

En este artículo se describe el proceso de desarrollo de una herramienta de software diseñada con el propósito de realizar los cálculos necesarios para obtener cuatro invariantes topológicos, que son los números de Betti, la dimensión del complejo simplicial, el vector de estructura, y la característica Euler-Poincaré.

Los resultados de estos cálculos se interpretan según una propuesta del profesor Luis Eduardo Múnera, en la que se plantea una definición formal de cohesión y acoplamiento en ingeniería de software, criterios importantes para evaluar la modularidad de un diseño. El artículo también describe la herramienta en sí, su funcionalidad y el significado que tiene para el proyecto del mencionado docente.

Para lograr la descripción del proceso y de la herramienta, se realiza un breve recuento histórico que sirve como marco de referencia al lector para mostrar la evolución de las ideas y conceptos sobre los cuales se apoya el proyecto de investigación, y la manera en la que estos fueron asimilados por el equipo de estudiantes que asumió la responsabilidad de llevarlo a cabo.

PALABRAS CLAVES

Diseño modular, topología algebraica, cálculos de invariantes topológicos, números de Betti, dimensión del complejo simplicial, característica Euler-Poincaré, vector de estructura, cohesión, acoplamiento, Matlab.

Clasificación: A

INTRODUCCIÓN

El profesor Luis Eduardo Múnera identificó la falta de formalidad matemática que respaldaba las actividades de ingeniería de software como son análisis, diseño y desarrollo. Esta carencia llevó a que desarrollara una propuesta para formalizar el diseño modular en la ingeniería del software basándose en algunos conceptos de la topología algebraica.[1]

Como proyecto de grado, los estudiantes Juan Pablo Bedoya, Aura Lucía Pérez y Javier Guillermo Valdés decidieron realizar una pieza de software para calcular los invariantes topológicos de un complejo simplicial, herramienta que servirá de apoyo a la investigación del profesor Múnera.

Se determinó que la herramienta calcularía los siguientes invariantes topológicos:

- La dimensión del complejo simplicial.
- La característica Euler-Poincaré del complejo simplicial.
- El vector de números de Betti asociado al complejo.
- El vector de estructura asociado al complejo.
- La cohesión.
- El acoplamiento.
- Los agujeros del complejo.

La importancia de este programa radica en la agilización de los cálculos de los invariantes topológicos anteriormente definidos, con el propósito de facilitar la investigación de quienes se apoyan en la topología algebraica.

MARCO TEÓRICO

Cohesión, acoplamiento, modularidad son términos frecuentemente utilizados en sistemas de información e ingeniería de software, tanto dentro del Paradigma Estructurado como en el Paradigma Orientado a Objetos. Pero a pesar de su importancia no son precisados de manera formal.

El profesor Luis Eduardo Múnera realizó una propuesta para definir y medir la cohesión y el acoplamiento, mediante el uso de la Topología Algebraica, una rama de la matemática que estudia las transformaciones continuas.

Un complejo simplicial abstracto K sobre un conjunto finito cuyos elementos se llaman vértices V = {a0,...,an} es un subconjunto no vacío de partes de V (excluyendo el vacío) cuyos elementos son llamados símplices con las siguientes propiedades:

(P1) Si $\sigma \varepsilon K$ y $\tau \subset \sigma$ entonces $\tau \varepsilon K$. Decimos que σ y τ son símplices y que τ es una cara de σ.

(P2) Si σ y τ pertenecen a K, entonces $\sigma \cap \tau$ o bien es vacía, o bien es una cara común de σ y τ.

(P3) Si a_i pertenece a V entonces $\{a_i\}$ pertenece a K.

1. Múnera Salazar, Luis Eduardo. *Una aproximación topológica al diseño modular en ingeniería de software.* S&T. Revista de la Facultad de Ingeniería. Universidad Icesi, No. 2. 2003. p.57-73

A un complejo simplicial K le podemos asociar arreglos numéricos que son invariantes topológicos (todos los poliedros equivalentes topológicamente poseen los mismos arreglos).

- **Números de Betti:** A un complejo simplicial K de dimensión m le podemos asociar un arreglo

$\beta = < \beta_O, \beta_1,...,\beta_N >$

en donde ß$_i$ es el correspondiente número de Betti para cada dimensión con i desde 0 hasta m.

El criterio para calcular los números de Betti es el siguiente:

$\beta_P (K) = n_p$ - rango ∂_p - rango ∂_{p+1}

Los números de Betti para cada dimensión señalan el máximo número de ciclos no homólogos a cero (agujeros) y homológicamente independientes.

$\beta_0 = Q_0$, que se define como el número de componentes arco-conexas. β_1 es el número de "túneles" linealmente independientes (que llamaremos 2-agujeros), β_2 es el número de agujeros de dimensión 3 (que llamaremos 3-agujeros) del complejo; en general, β_i es el número de agujeros de dimensión i+1 (i+1-agujeros) del complejo.

- **Vector de estructura:** A un complejo simplicial K le podemos asociar arreglos numéricos que son invariantes topológicos (todos los poliedros equivalentes topológicamente poseen los mismos arreglos). El primero de ellos se conoce con el nombre de primer vector de estructura del complejo y permite ver la conectividad interna del complejo (visión local) recurriendo a la noción de q-conectividad. Esta noción y sus aplicaciones en ciencias sociales fueron desarrolladas por el matemático Ronald Atkin.

Si K es un complejo finito no vacío de dimensión m, le podemos asociar el arreglo $\mathbf{Q = <Q_0, Q_1,..., Q_m >}$ en donde Q_i es la cardinalidad de K/γi, , siendo $Q_i \geq 1$, i = 0,1,.., m.

- **Característica Euler Poincaré:** Dado un complejo simplicial de dimensión m, sabemos que

- n_0 = número de vértices (símplices de dimensión 0)

- n_1 = número de segmentos (símplices de dimensión 1)

- n_2 = número de triángulos (símplices de dimensión 2)

- n_3 = número de tetraedros (símplices de dimensión 3)

. . .

- n_m = número de símplices de dimensión m.

Los matemáticos Leonard Euler y René Descartes descubrieron que n0 - n1 + n2 = 2 para toda superficie poliédrica (tetraedros, icosaedro, etc).

Henri Poincaré extendió este resultado para cualquier complejo:

$$\mathbf{X = (n_0 - n_1) + (n_2 - n_3) + (n_4 - n_5) + . . . + (n_{m-1} - n_m)}$$

en donde el número X se conoce con el nombre de características Euler-Poincaré.

- **Dimensión**: La dimensión de un símplice es el número de sus vértices menos uno. La dimensión de K es el máximo de las dimensiones de todos sus símplices.

INFORMACIÓN GENERAL DEL PROYECTO

Realización

El proyecto comenzó a gestarse en el segundo semestre del año 2001. En ese entonces, el profesor Luis Eduardo Múnera presentó su investigación realizada sobre la topología algebraica y sus aplicaciones en el diseño modular en ingeniería del software y su necesidad de agilizar los cálculos.

Durante los dos siguientes semestres se trabajó en el fundamento teórico, sobre todo en la manera de calcular los agujeros de un complejo simplicial, y en la realización de los primeros algoritmos que calcularían los invariantes topológicos del complejo simplicial dado. Para tales cálculos y, después de cierta indagación sobre las herramientas de desarrollo disponibles, se eligió Matlab, en su versión 6.1, ya que su funcionalidad permite un manejo potente de las matrices y además proporciona ciertos cálculos matemáticos que se necesitaban para lograr el objetivo. Para el comienzo del segundo semestre de 2002, el proyecto había finalizado satisfactoriamente. Los resultados que arrojaba la herramienta permitieron efectuar ajustes a los algoritmos planteados y hacer refinamientos a la teoría.

Para utilizar los algoritmos desarrollados en Matlab, se requería cierto conocimiento de la herramienta, lo cual generaba complicaciones a las personas que no lo tenían. Se planteó así la necesidad de generar una interfaz gráfica "amigable". Se creó entonces la primera versión del programa, la cual se desarrolló con el entorno de programación de Visual Studio 6, con el lenguaje Visual Basic. Se escogió Visual Basic ya que Mathworks, la casa de software dueña de Matlab, proveía una herramienta llamada Matrix VB que pone a disposición del entorno de Visual Basic toda la funcionalidad de Matlab. El resultado fue una interfaz de usuario que permitía introducir la información directamente en el programa, o introducir información en archivos con formatos de Matlab (extensión .MAT).

Algunas de las falencias de la primera versión de la herramienta incluían el hecho de que ésta no contó con la funcionalidad del cálculo de agujeros. Esta funcionalidad es clave para los objetivos del proyecto, en cuanto permite la localización precisa de las fallas en el diseño, representado en el complejo simplicial.

Durante el año 2003 se encontraron detalles que refinaban aún más la propuesta, y los resultados obtenidos con el uso de la herramienta permitieron hacer cambios a la interpretación de la teoría y su relación con el objeto de estudio. Así, se terminó de ajustar el algoritmo para el cálculo de agujeros. Dada la complejidad y extensión de dicho cálculo, se decidió no implementarlo utilizando Matlab. Para el desarrollo de este algoritmo realmente no se necesitaba tanta potencia matemática, como la ofrecida por Matlab, sino un manejo más eficiente de los recursos del computador como la memoria. Con base en este criterio y para el segundo semestre de 2003 se imple-

mentó el algoritmo del cálculo de agujeros. En esta ocasión se utilizó la nueva herramienta de Microsoft, Visual Studio .NET, y específicamente el lenguaje C#. También se empleó la herramienta COM Builder del entorno Matlab, en su versión 6.5. Con el Matlab COM Builder se generó un objeto COM que empaquetaba los algoritmos desarrollados en Matlab y permitía su uso en C#.

Surgió pues la última versión, la 2.0, que incluye el cálculo de agujeros, las mejoras a los algoritmos, las nuevas definiciones formales de los conceptos de cohesión y acoplamiento, y una corrección a la implementación que se hizo de los números de Betti, que no se había logrado a cabalidad para el primer release.

Resultados

El resultado fundamental del proyecto es una herramienta de software que básicamente realiza dos tipos de cálculos:

- Dado un complejo simplicial abstracto representado por su matriz de incidencia, la herramienta permite obtener cuatro invariantes topológicos asociados al complejo:
- Dimensión
- Característica Euler – Poincaré
- Vector de estructura
- Vector de números de Betti

Además permite identificar agujeros.

- Adicionalmente, la herramienta permite calcular la cohesión y el acoplamiento de un sistema representado por su complejo simplicial.

El programa está compuesto por cuatro módulos: el que calcula los Invariantes topológicos, el que captura las matrices de incidencia, el que calcula los agujeros y el de control, encargado de la coordinación de los anteriores y punto de entrada al programa.

El módulo encargado del cálculo de los invariantes topológicos, realiza el cálculo de los cuatro invariantes topológicos y de la cohesión y el acoplamiento. Para ello se apoya en dos componentes COM, implementados en Matlab y convertidos en dll gracias a la utilidad de Matlab MVComBuilder. El primero, MVComUtil.dll, es el que encapsula las funcionalidades de Matlab; el segundo, InvarTopo.dll, es el que contiene la funcionalidad desarrollada por el equipo de estudiantes.

El módulo de cálculo de agujeros es el encargado de encontrar los agujeros del complejo simplicial dado y de identificarlos. Está compuesto por una clase llamada classCalculoAgujeros, la cual contiene todo el proceso de cálculo de agujeros. Este módulo no se apoya en el componente de Matlab.

El algoritmo de Cálculo de Agujeros está definido de la siguiente forma:

Entrada: la Matriz de Incidencia (MI)

Salida: $\{H_{q+1}\}_{q=O}^{k-1}$ La familia de conjuntos de agujeros en cada dimensión.

Proceso:

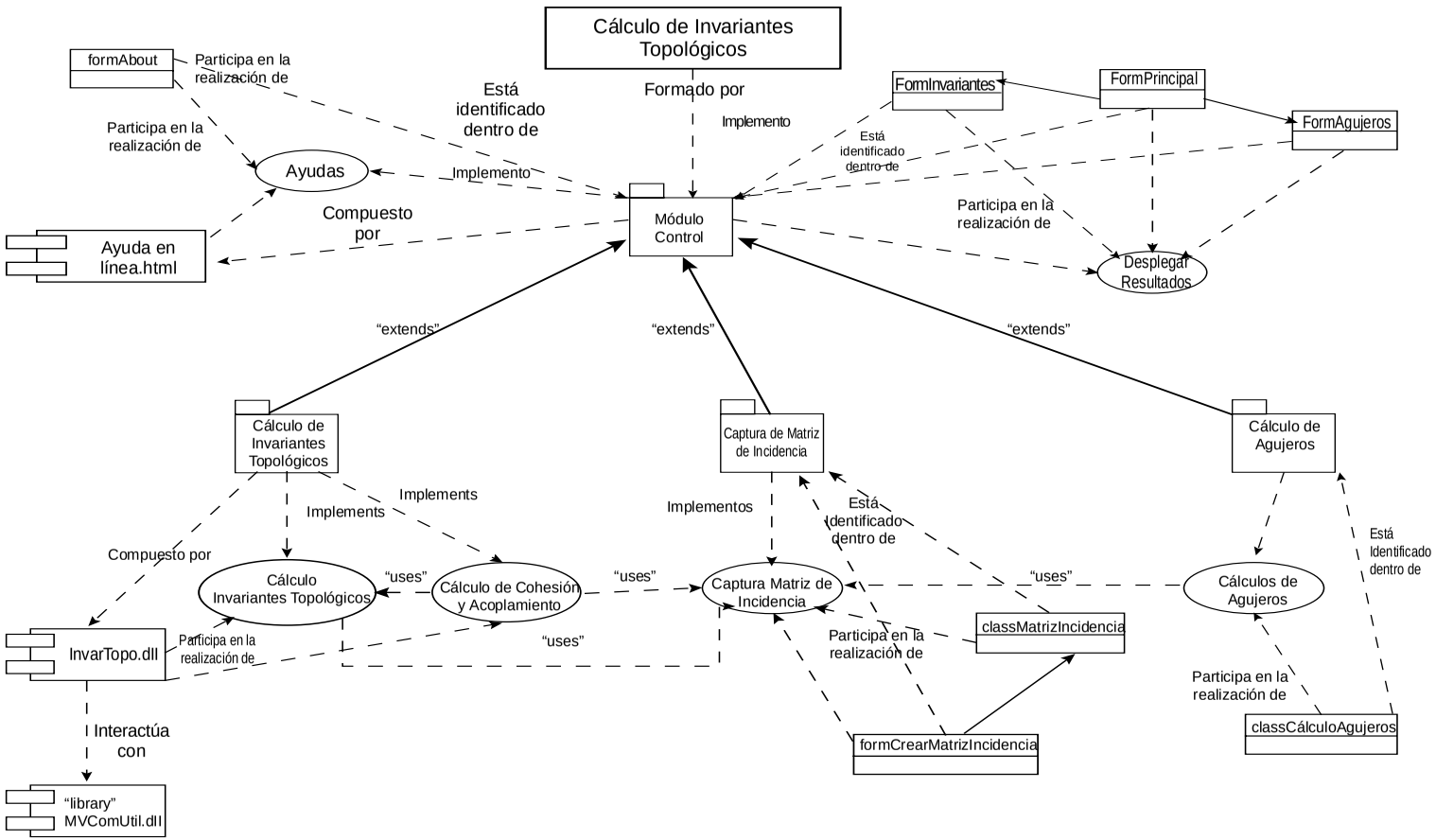
Aplicar el procedimiento de obtención de la matriz reducida, MR.

IF MR = 0 THEN $\{H_{q+1}\}_{q=0}^{k-1} = \emptyset$ y FIN

Aplicar el procedimiento de obtención de las matrices de q-adyacencia

Aplicar el procedimiento de obtención de q-ciclos para cada matriz de q-adyacencia

ARQUITECTURA DEL PROGRAMA



Aplicar el procedimiento de obtención de q+1 agujeros para cada lista lq de q-ciclos, obteniéndose la familia

$\{H_{q+1}\}_{q=0}^{k-1}$

IF *existe* un agujero A perteneciente a más de un conjunto de $\{H_{q+1}\}_{q=0}^{k-1}$

THEN lo dejamos en el conjunto con dimensión más alta y lo eliminamos en los conjuntos con dimensiones más bajas.

Procedimiento de obtención de la matriz reducida

Entrada: La matriz de incidencia MI.

Salida: La matriz reducida MR.

Proceso:

Sobre la matriz de incidencia realizamos tres tipos de operaciones de reducción:

Columnas con un solo 1, se convierte este en 0

Filas cuyos unos estén incluidos en los unos de otra fila, se convierten en ceros.

Columnas cuyos unos estén incluidos en los unos de otra columna, se convierten en ceros.

IF la matriz resultante contiene sólo ceros

THEN, no hay ciclos y por lo tanto no hay agujeros. Es decir, MR = 0.

ELSE, eliminamos las filas y las columnas de la matriz que sólo contienen ceros, generándose la matriz reducida MR # 0

Procedimiento de obtención de matrices de q-adyacencia

Entrada: la matriz reducida (MR)

Salida: Las matrices de q-adyacencia (q = 0,1,2,...)

Proceso:

Calcular la transpuesta de la matriz reducida MR.

FOR q = 0 to k (k = Dimensión de MR) hacer

Obtener la matriz de q-adyacencia

Procedimiento de obtención de los q-ciclos

Entrada: Una matriz de q-adyacencia

Salida: Una lista l_q formada por los q-ciclos

Proceso:

1. Convertir la matriz de q-adyacencia en una matriz latina L

2. Hacer $L^1 = L$

FOR i = 2 hasta k (Dimensión de L) HACER

$L^i = L^{i-1} \times L^1$ (Producto Latino)

El producto latino es como sigue:

Sean $A = [a_1, a_2, a_3,...,a] \in L^{i-1}$ y $B = [b_1,b_2] \in L^1$, los elementos a multiplicar:

IF A = O v B = O THEN AxB = O

ELSE

IF i< q + 3 THEN $(AxB)^0 = [a_1,a_2,a_3,...,a_i,b_2]$

IF $(AxB)^0 \in$ Diagonal de L^i

THEN hacer (AxB) = 0

ELSE hacer $(AxB) = (AxB)^0$

ELSE $(AxB)^0 = [a_1,a_2,a_3,...,a_i,b_2]$

IF $(AxB)^0 \in$ Diagonal de L^i

THEN $(AxB)^0$ es un q-ciclo de longitud i

Incluimos $(AxB)^0$ en lq y hacemos

$(AxB) = (AxB)^0$

ELSE

IF ($a_j = b_2$, ə $a_j = a_k$, əjək)

THEN (AxB) = 0

ELSE hacemos $(AxB) = (AxB)^0$

3. De los q-ciclos de lq eliminamos a los semejantes (dos q-ciclos de igual dimensión y longitud son semejantes si contienen el mismo conjunto de vértices).

Procedimiento de obtención de los q+1-agujeros

Entrada: Una lista l_q de q-ciclos

Salida: Una lista de H_{q+1} de q+1-agujeros

Proceso:

1. Para cada q-ciclo de lq hacemos:

IF q-ciclo no es un símplice de MR

THEN es un q+1-agujero y lo incluimos en H_{q+1}

2. En H_{q+1} eliminamos los q+1-agujeros que contienen a otros.

El módulo de captura de la matriz de incidencia está compuesto por la forma formCrearMatrizIncidencia, que contiene un Grid en el cual el usuario ingresa el Complejo Simplicial, y lo almacena en la clase classMatriz Incidencia.

Además, el programa contiene una ayuda en línea que describe detalladamente el funcionamiento de todo el software y una forma Acerca de.

CONCLUSIÓN

1. El apoyo en los recursos tecnológicos para realizar procesamiento de información, facilita y agiliza los procedimientos llevados a cabo en la obtención de resultados en las investigaciones.
2. Es importante conocer múltiples opciones, en cuanto a metodologías y herramientas de construcción de software, para que se facilite el proceso y para que el software final sea más efectivo en cuanto a rendimiento, interfaz de usuario, etc.
3. Es conveniente que en un proyecto de desarrollo de una aplicación existan personas con diferentes enfoques en el manejo de las arquitecturas, herramientas y metodologías de diseño, desarrollo y distribución de software, con el fin de lograr un resultado acorde con las necesidades del usuario final.

BIBLIOGRAFÍA

Múnera Salazar, Luis Eduardo. *Una aproximación topológica al diseño modular en ingeniería de software.* S&T. Revista de la Facultad de Ingeniería. Universidad Icesi, No. 2. 2003. p.57-73

Matlab, [ayuda en línea].

Visual Studio .NET versión 2003, [ayuda en línea].

Referencia del API de C#, <http://msdn.microsoft.com>

Robohelp Office versión [ayuda en línea].

Wise for Windows Installer versión 5.0, [ayuda en línea].

CURRÍCULOS

Juan Pablo Bedoya Giraldo. Ingeniero de Sistemas de la Universidad Icesi. Realizó su práctica en La 14 S.A. como Analista y Líder Técnico en la implementación de un sistema de nómina basado en Oracle. Actualmente se desempeña como Ingeniero de Desarrollo en Sysgold, una casa de software que provee soluciones para dispositivos móviles.

Aura Lucía Pérez Escobar. Estudiante de Ingeniería de Sistemas de la Universidad Icesi. Realizó su práctica en el LAAS (Toulouse–Francia) como desarrolladora en un proyecto de Simulación y Optimización de Redes IP-DiffServ-MPLS. Actualmente se desempeña como Analista/Programador Junior en la oficina de Desarrollo de Sistemas de la Universidad Icesi.

Javier Guillermo Valdés Duque. Ingeniero de Sistemas de la Universidad Icesi. Realizó su práctica en el Hotel Pacífico Royal, en el departamento de sistemas, desempeñando el cargo de asistente del director de sistemas. Actualmente se encuentra adelantando un proyecto empresarial propio, relacionado con el desarrollo de software.