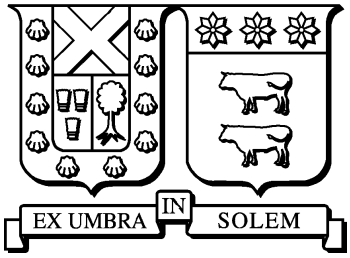
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE INFORMÁTICA

SANTIAGO – CHILE



“MINERÍA DE ROLES EN UNA PLATAFORMA DE GESTIÓN DE PROYECTOS ACADÉMICOS”

ESTEBAN OGRODNIK ORELLANA

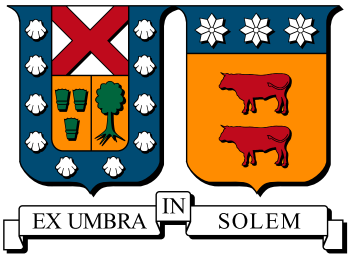
MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE

INGENIERO CIVIL INFORMÁTICO

PROFESOR GUÍA: HERNÁN ASTUDILLO ROJAS

ENERO 2018

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO – CHILE



"MINERÍA DE ROLES EN UNA PLATAFORMA DE GESTIÓN DE PROYECTOS ACADÉMICOS"

ESTEBAN OGRODNIK ORELLANA

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL INFORMÁTICO

PROFESOR GUÍA: HERNÁN ASTUDILLO ROJAS
PROFESOR CORREFERENTE: MARCOS SEPÚLVEDA

ENERO 2018

Agradecimientos

Agradezco a mi familia, amigos y profesores por el apoyo incondicional que me han brindado. Aún en los más difíciles momentos de mi vida estuvieron presentes, ayudándome a construir mi sueño. Les estoy eternamente agradecido.

Resumen

El presente estudio corresponde a la implementación del sistema de control de acceso basado en roles de la plataforma de la asignatura *Análisis y Diseño de Software* de la $UTFSM$. Para ello, se buscó el conjunto de roles óptimo que permite relacionar los usuarios de dicho sistema con sus respectivos permisos de acceso, es decir, se resolvió el "*Problema de la Minería de Roles*" (RMP) para el sistema dado. Esto con la finalidad de mejorar la seguridad, mantención y administración de éste.

Este estudio expone un criterio para escoger los enfoques del RMP más aptos. En base a dicho criterio, se identificaron 4 enfoques posibles (*Basic-RMP*, *Edge-RMP*, *RHMP* y *WSCO*), los cuales comparten el objetivo de minimizar el tamaño de uno o más elementos del modelo $RBAC$. De los enfoques expuestos, se seleccionó *WSCO* por su adaptabilidad a diversos objetivos. En base a lo anterior, se utilizó la métrica *Weighted Structural Complexity* (WSC), con diversos vectores de peso, para evaluar la calidad de las soluciones propuestas.

Se escogieron los algoritmos más eficaces de la minería de roles que estén alineados con $WSCO$, los cuales son: HM, GO, HPe y HPr. Por medio de la herramienta *RMiner* (de libre uso) fue posible ejecutar cada algoritmo en base a los diversos vectores de peso utilizado. De los resultados obtenidos, se destaca que wsc evalúa eficazmente la calidad de las soluciones propuesta por los diversos algoritmos. También, se identificó que la característica más importante de un rol corresponde a la cantidad de usuarios asignados. Por último, se destaca que *GO* obtuvo la mejor solución y HM fue el peor, debido a la excesiva cantidad de roles, relaciones y asignaciones.

Abstract

The present study is about the implementation of role-based access control system of the platform of the subject *Análisis y Diseño de Software* of the university *Federico Santa María*. For that purpose, it was found the optimal set of roles that allows to relate the users of this system with their respective access permissions, in other words, the "*Role Mining Problem*"(RMP) was solved for the given system. This in order to improve the security, maintenance and administration of this system.

This study presents a criterion for choosing the most suitable RMP approaches. Based on this criterion, 4 possible approaches were identified (*Basic-RMP*, *Edge-RMP*, *RHMP* and *WSCO*), which share the objective of minimizing the size of one or more elements of the $RBAC$ model. From the previous approaches, *WSCO* was selected for its adaptability to various objectives. The metric *Weighted Structural Complexity* (WSC), along with various weight vectors, was used to evaluate the quality of the proposed solutions.

In this study were chosen the most effective role mining algorithms, that are aligned with $WSCO$, which are: HM, GO, HPe and HPr. Using the *RMiner* tool (free to use), it was possible to execute each algorithm based on the different weight vectors used. From the results obtained, it is outstanding that wsc effectively evaluates the quality of the solutions proposed by the various algorithms. Also, it was identified that the most important characteristic of a role corresponds to the number of users assigned. Finally, it stands out that GO obtained the best solution and HM was the worst, due to the excessive amount of roles, relationships and assignments.

Índice de Contenidos

Índice de cuadros

Índice de figuras

Glosario

- **IT**, *Tecnologías de la Información*: es la aplicación de cualquier herramienta tecnológica para manipular, almacenar y transmitir datos o información.
- **RBAC**, *Control de Acceso Basado en los Roles*: es un modelo de sistema de control que limita el acceso a los recursos dependiendo de los roles que tenga asignado el usuario.
- **Rol**: desde la perspectiva organizacional, es la función que algún empleado desempeña. Por otra parte, en el modelo $RBAC$ se relaciona con el conjunto de permisos que puede estar asociado a un usuario del sistema.
- **RM**, *Minería de Roles*: es el proceso de búsqueda del conjunto de roles óptimo de un sistema $RBAC$ mediante el uso de algún algoritmo.
- **RMP**, *Problema de la Minería de Roles*: plantea la necesidad de encontrar el conjunto de roles óptimo de un sistema $RBAC$, asegurando de esta forma que el sistema sea compacto, seguro, consistente, fácil de administrar y rápido de mantener.

Introducción

El presente texto tiene como finalidad ser una guía enfocada a la implementación de sistemas de Control de Accesos Basados en Roles ($RBAC$), utilizando los algoritmos y herramientas gratuitas más eficaces. Para ello, se tomará el rol de administrador de un sistema de control de acceso real, documentando todo el proceso involucrado en la mantención de un sistema $RBAC$ mediante la Ingeniería de Roles, desde la toma de datos hasta el análisis de las soluciones propuestas por los algoritmos aplicados.

El $RBAC$ [23] es el modelo de acceso predominante en las organizaciones, ya que simplifica y agiliza el proceso de administración y mantención de los sistemas de seguridad. El éxito del $RBAC$ se debe a la forma en que administra los permisos de acceso, en la cual los usuarios del sistema reciben el acceso a sus respectivos recursos por medio de los roles asignados. Así, se simplifican los procesos involucrados en la manipulación de permisos, se estructura el sistema de una forma más natural, y se facilita la identificación de permisos erróneos, ausentes u obsoletos. La mayor desventaja de este modelo se relaciona con la identificación del conjunto de roles óptimo para cada sistema, lo que corresponde al aspecto más costoso de la implementación del modelo $RBAC$ [19]. Dicho conjunto debe asegurar que el sistema sea compacto, seguro, consistente, fácil de administrar y rápido de mantener. Este problema, encontrar el conjunto de roles óptimo, es conocido como el "*Problema de la Minería de Roles*", en inglés "*Role Mining Problem*" (RMP), mientras que el proceso de búsqueda del conjunto de roles óptimo es conocido como "*Ingeniería de Roles*" (cuya traducción al inglés es "*Role Engineering*" con siglas RE) [3]. Actualmente, no existe un consenso sobre la definición formal del RMP, y de las métricas a utilizar para su evaluación, razón por la cual, existen diversos enfoques y criterios de evaluación.

Más aún, la complejidad de este problema es *NP-Duro* [15], razón por la cual en los algoritmos existentes se utiliza alguna heurística para su resolución, lo que no asegura que la solución encontrada sea la óptima.

El caso de estudio corresponde al uso de la minería de roles para la implementación del sistema de control de acceso de la plataforma de la asignatura *Análisis y Diseño de Software* de la *Universidad Técnica Federico Santa María, para los Campus San Joaquín* y *Casa Central*, usando los algoritmos y herramientas gratuitas más eficaces.

El resto del documento se organiza de la siguiente forma:

- **Capítulo 1, Presentación del Problema**: en esta sección se describe detalladamente el problema que es resuelto en el presente estudio, junto a los objetivos principales y secundarios de éste.
- **Capítulo 2, Control de Acceso Basados en Roles**: en esta sección se analiza todos los conceptos y requerimientos propios del modelo $RBAC$. Además, se identifica los estándares más importantes relacionados con dicho modelo. Por último, se hace mención de las ventajas relacionadas con la implementación de un sistema basado en este modelo.
- **Capítulo 3, Ingeniería de Roles**: esta sección detalla el proceso de la ingeniería de roles, destacando las distintas clasificaciones de éste. Adicionalmente, se hace mención sobre diversos estudios realizados en base a los distintos enfoques de la ingeniería de roles.
- **Capítulo 4, Minería de Roles**: en esta sección se describe los enfoques del RMP que fueron usados en el presente estudio, haciendo hincapié en el proceso de selección de estos, y en los conceptos requeridos para la definición formal de cada tipo de problema. Además, se describen los algoritmos seleccionados para la resolución de estos problemas. Finalmente, se analizan las métricas usadas para la evaluación del desempeño de estos algoritmos.
- **Capítulo 5, Conjuntos de Datos**: en esta sección se expone la organización participante, y se detallan las características principales del sistema de control de acceso

estudiado.

- **Capítulo 6, Resultados**: en esta sección se describen los resultados obtenidos tras ejecutar los algoritmos seleccionados, en base al conjunto de datos dado y a los criterios de evaluación seleccionados. Se destaca, según sea el caso, las características más relevantes de cada solución, la jerarquía de roles, y los resultados de las respectivas métricas.

- **Capítulo 7, Análisis**: en este capítulo se comparan y analizan, los resultados del presente estudio, enfatizando dicho análisis en la identificación del conjunto de roles óptimo que se adecue a las necesidades organizacionales.

- **Conclusiones y Trabajo Futuro**: en este apartado se exponen las conclusiones del estudio realizado y se recomiendan algunos enfoques para futuros trabajos.

Capítulo 1

Descripción del Problema

Originalmente, este estudio tenía como finalidad, documentar todo el proceso que conlleva la utilización de la minería de roles para la mantención del sistema de control de acceso de la plataforma de la asignatura *Análisis y Diseño de Software* (ADS) de la *Universidad Técnica Federico Santa María*, usando los algoritmos y herramientas gratuitas más eficaces. Tras verificar la información existente, se descubrió que el sistema desplegado quebrantaba varios de los requerimientos propios del modelo $RBAC$, entre estos, se destaca que cada usuario tenía asignado un único rol, el cual poseía todos los permisos de acceso. Por esta razón, se decidió cambiar el enfoque del estudio hacia la implementación del respectivo sistema de control de acceso. Así, el problema se define como: "*encontrar, por medio de la minería de roles, el conjunto de roles óptimo que relacione correctamente los usuarios con sus respectivos permisos de acceso*".

Como se mencionó en la introducción, el problema de "*encontrar el conjunto de roles óptimo*", es conocido como el "*Problema de la Minería de Roles*" (RMP), por lo tanto, este estudio involucra "*resolver el RMP en base al conjunto de usuarios y permisos del sistema de control de acceso de la plataforma de la asignatura ADS, usando los algoritmos y herramientas gratuitas más eficaces de la minería de roles, y considerando las restricciones propias de la organización.*

Actualmente, no existe un consenso sobre la definición formal del RMP, generando así, distintas variantes del RMP. Además, no hay un acuerdo entre los investigadores sobre

las métricas a utilizar para medir la calidad de los roles. En base a todo lo expuesto, se considera como objetivo principal del estudio: comprobar que la minería de roles es capaz de resolver el RMP para el caso de la plataforma de la asignatura ADS. Esto implica encontrar el conjunto de roles óptimo para el problema dado.

Por otra parte, los objetivos secundarios son los siguientes:

1. Identificar los criterios considerados en la selección de los enfoques del RMP.
2. Verificar la eficacia de las métricas utilizadas en el presente estudio para evaluar la calidad de las soluciones propuestas por los diversos algoritmos.
3. Medir cuantitativamente el desempeño de los algoritmos usados en el presente estudio.
4. Actualizar los resultados del estudio realizado por Molloy et. al. [18].

Capítulo 2

Modelo RBAC

2.1. Introducción al RBAC

Previo a 1970, las empresas administraban el acceso a los recursos de la organización mediante la asignación directa de permisos a cada trabajador del sistema informático utilizado. Este modelo de administración de acceso ha sido denominado como "*Sistema Tradicional*", al ser el primer sistema en controlar los accesos mediante la asignación de permisos. La principal ventaja de este modelo es la seguridad de la información, ya que los usuarios sólo podrán acceder a los recursos permitidos, tal como se ilustra en la Figura 2.1.

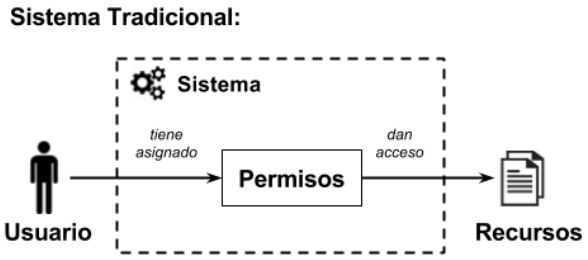


Figura 2.1: Sistema Tradicional.

En general, el *sistema tradicional* es de utilidad cuando éste gestiona un número pequeño de usuarios y permisos de acceso. Pero, a medida que el conjunto de datos crece, éste se

vuelve más difícil de administrar y de mantener, incrementando de este modo, los recursos destinados a dichas labores. Para empeorar esta situación, es más probable encontrar asignaciones erróneas en un conjunto grande de datos, que en uno de menor tamaño.

En los años siguientes, surgieron distintos tipos de sistemas de Control de Acceso (*AC*) con el objetivo de solucionar los problemas descritos anteriormente. Un sistema *AC* que surgió en esa época es el sistema de Control de Acceso Basado en los Roles, *RBAC* [23]. El modelo *RBAC*, a diferencia del tradicional, relaciona a los usuarios con sus respectivos permisos por medio de los *roles* que éste posee, tal como lo demuestra la Figura 2.2.

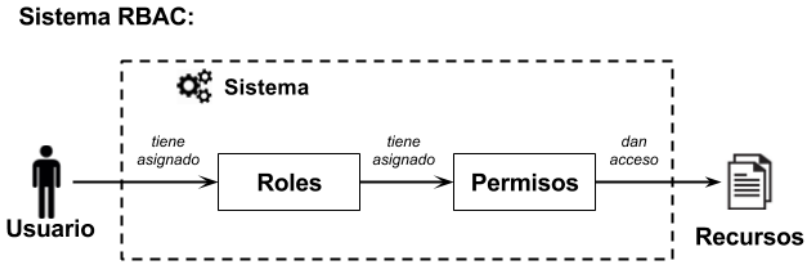


Figura 2.2: Sistema *RBAC*.

De la Figura 2.2 se desprende que cada usuario tiene asignado uno o más roles, y cada rol tiene, a su vez, asignado uno o más permisos. Así, y al igual que en el modelo tradicional, un usuario sólo podrá acceder a los recursos que sus roles tengan permitido. Esta forma de gestionar los recursos simplifica considerablemente la manipulación de los permisos. Un sistema tradicional con n usuarios y m permisos, requiere mn relaciones para asignar correctamente los permisos a los respectivos usuarios, mientras que el sistema *RBAC*, requiere $m+n$, cantidad considerablemente menor, simplificando así, las labores de mantención e implementación del sistema de control de acceso. Además, el conjunto de roles es estructurado de una forma más natural desde la perspectiva organizacional, ya que éstos se pueden relacionar con los cargos organizacionales y con restricciones propias del negocio.

Actualmente, el *RBAC* es el modelo de control de acceso más utilizado en las organizaciones en todo el mundo, ya que, además de solucionar los problemas mencionados, aumenta la productividad organizacional y refuerza las políticas de seguridad.

Finalmente, hay que destacar que el proceso de implementación de un sistema $RBAC$ está dividido en cuatro pasos: (1) obtener el conjunto de datos inicial; (2) revisar y corregir los errores presentes en dicho conjunto de datos; (3) identificar el conjunto de roles óptimo mediante algún método de análisis, asociando los permisos y usuarios correspondientes a cada uno de estos; (4) eliminar la relación directa entre los usuarios y sus respectivos permisos, tal como lo representa la Figura 2.3, en donde las letras U, R y P denotan a los usuarios, roles y permisos respectivamente, UP corresponde a la relación entre U y P, y UP' es la relación corregida entre estos mismos elementos.

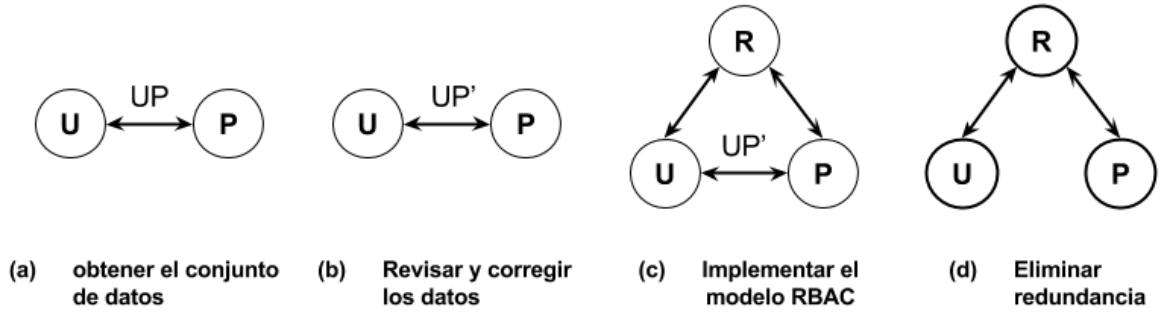


Figura 2.3: Implementación de un sistema $RBAC$.

2.2. Estándares

En esta sección se identifican algunos estándares existentes en la literatura, siendo la norma *ISO 27001* la más importante pero, a su vez, la más costosa de implementar. Tanto la documentación de la norma *ISO 27001* como la *ANSI INCITS 359-2012* deben ser adquiridas por las empresas y no son de carácter público, razón por la cual no podrán ser analizadas en el presente texto. Por otro lado, el $XACML$ es un estándar abierto basado únicamente en XML, y es por esta limitación que se decidió omitir dicho análisis.

2.2.1. ANSI INCITS 359-2012

El modelo *NIST* para *RBAC* fue aceptado por el Instituto Nacional Estadounidense de Estándares (del inglés "*American National Standards Institute*" con siglas *ANSI*), en conjunto con el Comité Internacional de Estándares de Tecnología de la Información (del inglés "*InterNational Committee for Information Technology Standards*" con siglas *INCITS*). Fue publicado el 2004 como *ANSI INCITS 359-2004*, y actualizado por última vez el 2012, estableciendo el estándar *ANSI INCITS 359-2012*. Se puede adquirir una copia de este estándar mediante el sitio oficial de ANSI [1], por 60 USD.

Este estándar está constituido por dos partes: el "*modelo de referencia de RBAC*", y "*el sistema RBAC y su especificación funcional administrativa*". El "*modelo de referencia de RBAC*" define los elementos básicos del *RBAC* (p. ej. usuarios, roles, permisos, procedimientos y objetos) y las relaciones que se incluyen en este estándar. Este modelo tiene dos propósitos: (1) definir e identificar las características mínimas que deben ser incluidas en los sistemas *RBAC*; y (2) proporcionar un lenguaje preciso y consistente en la definición de la especificación funcional. Por otra parte, "*el sistema RBAC y su especificación funcional administrativa*" establece las características que requiere un sistema *RBAC*.

2.2.2. XACML

El lenguaje extensible de marcas de acceso de control (del inglés "*eXtensible Access Control Markup Language*" con siglas *XACML*) es un estándar abierto, publicado en la Organización para el Avance de Estándares de Información Estructurada (en inglés "*Organization for the Advancement of Structured Information Standards*" con siglas *OASIS*) que describen tanto un lenguaje declarativo de políticas de control de acceso basadas en *XML* (del inglés *"eXtensible Markup Language"* y traducido al castellano como "*Lenguaje de Marcas Extensible*") como un modelo de procesamiento de peticiones de acceso, basado en las políticas descritas.

2.2.3. ISO 27001

La *ISO 27001* es una norma internacional emitida por la Organización Internacional de Normalización (*ISO*) y describe cómo gestionar la seguridad de la información en cualquier tipo de organización. La revisión más reciente de esta norma fue publicada el 2013 y corregida el 2015. Actualmente, su nombre completo es "*ISO/IEC 27001:2013/Cor 2:2015*". Se puede adquirir una copia de este estándar mediante el sitio oficial de *ISO* [10] o *ANSI* [1], desde los 135 USD.

Además de poder adquirir la documentación, las organizaciones tienen la opción de certificarse por el cumplimiento de la norma *ISO 27001* mediante una entidad externa que esté acreditada por el mismo organismo. El costo de la certificación depende de la empresa externa, del tamaño de la organización a certificar, del estado actual de su sistema *RBAC*, del periodo que abarca la certificación y del nivel de información existente relacionada con la administración de los sistemas de seguridad de dicha organización. Es por esto que la certificación puede llegar a costar miles de dólares.

2.3. Conceptos

En esta sección, se hace mención de las definiciones y conceptos fundamentales del modelo *RBAC*. Estas definiciones serán de utilidad para describir formalmente el problema que se desea solucionar.

Definición 2.3.1 (Modelo Básico del RBAC [23]) *Las entidades básicas de todo sistema de control de acceso basado en roles son las siguientes:*

- U, R y P son los conjuntos de usuarios, roles y permisos respectivamente.
- $PA \subseteq P \times R$ corresponde a la relación de asignación los conjuntos de permisos y de roles. Cabe destacar que, en esta relación, se utiliza la letra "A" para señalar que se asignan los permisos a los respectivos roles.

- $UA \subseteq U \times R$ corresponde a la relación de asignación los conjuntos de usuarios y de roles. Similar al caso anterior, en esta relación, se utiliza la letra "A" para señalar que se asignan los roles a los respectivos usuarios.

- $DUPA \subseteq U \times P$ corresponde a la relación de asignación directa entre los conjuntos de usuarios y de permisos. En este caso, la letra "D" se utiliza para destacar que es una relación "directa", mientras que la letra "A" indica que en dicha relación se asignan los permisos a los respectivos usuarios.

- $UPA \subseteq UA \times PA \sim U \times P$ *(indirecto)* corresponde a la relación de asignación indirecta entre los conjuntos de usuarios y de permisos. Esta entidad se forma tras relacionar los elementos UA y PA, por medio del mismo conjunto de roles. Al igual que en el caso anterior, se utiliza la letra "A" para señalar que se asignan los permisos a los respectivos usuarios.

- $RH \subseteq R \times R$ es un conjunto parcialmente ordenado basado en el conjunto de roles R, que representa la jerarquía o dominancia entre estos. En la literatura, este conjunto es conocido simplemente como "*jerarquía de roles*". Cabe destacar que, la letra "H" se obtuvo de la palabra en inglés "*hierarchy*", cuya traducción al castellano corresponde a "*jerarquía*".

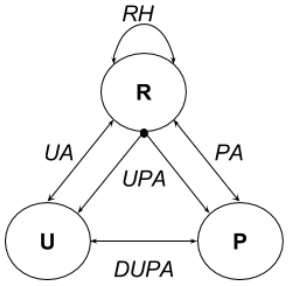


Figura 2.4: Diagrama de relaciones del modelo $RBAC$.

Hay que destacar que la definición original [23] consideraba, además de las demás entidades, un conjunto de sesiones (S). Las sesiones son individuales, temporales y están en control absoluto de los usuarios. Mientras dure la sesión, los usuarios podrán definir cuáles de los permisos asignados estarán activos, creando así un subconjunto de los permisos. Sin entrar en más detalles, el conjunto de sesiones no es considerado en el proceso

de ingeniería de roles debido a su nula influencia en la generación del conjunto de roles óptimo. Como se mencionó anteriormente, existen diversos enfoques para resolver el problema de la minería de roles, por eso es común encontrar variantes de este modelo que se adapten a dichos cambios. A continuación, se describe los cambios realizados al modelo básico del $RBAC$ para adaptarse a los enfoques utilizados en el presente texto:

Definición 2.3.2 (Modelo Extendido del RBAC [23]) *En base a las entidades básicas descritas con anterioridad, se consideran los siguientes cambios:*

- $DUPA_{in}$ corresponde a la relación de asignación directa entre usuarios y permisos que pertenece al conjunto de datos de entrada del problema estudiado.
- $DUPA_{out}$ es la relación de asignación directa entre usuarios y permisos que forma parte de la solución del problema estudiado.

En particular, $DUPA_{out}$ es utilizado para identificar las asignaciones entre los usuarios y sus respectivos permisos, que no pudieron ser representados por los roles existentes, ya sea porque estos roles no satisfacen los permisos requeridos o porque existe alguna restricción que impide la respectiva asignación. Cabe destacar que, si $DUPA_{out}$ no está presente en el conjunto de datos de salida, entonces $DUPA_{in}$ será nombrado simplemente como $DUPA$.

Para continuar con la explicación de los conceptos del $RBAC$, es necesario definir la propiedad de multiplicación matricial entre matrices binarias.

Definición 2.3.3 (Multiplicación Matricial Binaria) *Sean A y B, dos matrices binarias, entonces el producto entre éstas dará como resultado una matriz binaria C, siempre y cuando $A_{m\times k} \times B_{k\times n} = C_{m\times n}$. Así, cualquier elemento c_{ij} de la matriz C, se puede representar en función de los elementos a_{il} y b_{lj}, de las matrices A y B respectivamente, tal como lo demuestra la Ecuación 2.1.*

$$c_{ij} = \sum_{l=1}^{k} (a_{il} \cdot b_{lj}) \tag{2.1}$$

Donde $i = 1 \ldots m$, $j = 1 \ldots n$ y $l = 1 \ldots k$. Finalmente, hay que destacar que todos los elementos de las matrices mencionadas son números binarios, por lo tanto, a_{il}, b_{lj}, $c_{ij} \in \{0, 1\}$.

Continuando con el estudio de los conceptos, sea $AB \subseteq A \times B$, el resultado de la relación directa entre estas entidades, entonces se puede representar matricialmente la relación directa entre dichas entidades, por medio de una matriz binaria $M(AB)$, de tamaño $m \times n$, en donde m y n corresponden a la cantidad de elementos de la entidades A y B respectivamente. De esta forma, si el elemento a_{ij} de la matriz $M(AB)_{m \times n}$, con $i = 1 \ldots m$ y $j = 1 \ldots n$, corresponde a un 1, entonces se establece que existe una relación entre el i-ésimo elemento de A y el j-ésimo de B. Caso contrario ($a_{ij} = 0$), no habrá relación alguna entre los elementos mencionados.

En base a esta representación, y considerando la propiedad de multiplicación entre matrices binarias, se puede representar UPA mediante la matriz $M(UPA)_{m \times n} = M(UA)_{m \times k} \otimes M(PA)_{k \times n}$, en donde m es el número de usuarios, n es la cantidad de permisos y k el número de roles. Así, se puede expresar el elemento upa_{ij} de la matriz $M(UPA)_{m \times n}$ en función de los elementos ua_{il} y pa_{lj}, de las matrices $M(UA)_{m \times k}$ y $M(PA)_{k \times n}$ respectivamente, tal como lo demuestra la Ecuación 2.2.

$$upa_{ij} = \sum_{l=1}^{k} (ua_{il} \cdot pa_{lj}) \tag{2.2}$$

Donde $i = 1 \ldots m$, $j = 1 \ldots n$ y $l = 1 \ldots k$. Por lo tanto, UPA puede ser representado matricialmente, mediante las Ecuaciones 2.3 y 2.4:

$$\begin{bmatrix} \sum_{l=1}^{k} (ua_{1l} \cdot pa_{l1}) & \ldots & \sum_{l=1}^{k} (ua_{1l} \cdot pa_{ln}) \\ \vdots & \ddots & \vdots \\ \sum_{l=1}^{k} (ua_{ml} \cdot pa_{l1}) & \ldots & \sum_{l=1}^{k} (ua_{ml} \cdot pa_{ln}) \end{bmatrix}_{m \times n} = \begin{bmatrix} upa_{11} & \ldots & upa_{1n} \\ \vdots & \ddots & \vdots \\ upa_{m1} & \ldots & upa_{mn} \end{bmatrix}_{m \times n} \tag{2.3}$$

$$\begin{bmatrix} upa_{11} & \ldots & upa_{1n} \\ \vdots & \ddots & \vdots \\ upa_{m1} & \ldots & upa_{mn} \end{bmatrix}_{m \times n} = \begin{bmatrix} ua_{11} & \ldots & ua_{1k} \\ \vdots & \ddots & \vdots \\ ua_{m1} & \ldots & ua_{mk} \end{bmatrix}_{m \times k} \times \begin{bmatrix} pa_{11} & \ldots & pa_{1n} \\ \vdots & \ddots & \vdots \\ pa_{k1} & \ldots & pa_{kn} \end{bmatrix}_{k \times n} \tag{2.4}$$

Así, si el elemento $upa_{ij} = 1$, entonces el i-ésimo usuario se le asignó el j-ésimo permiso a través de un rol. Por otra parte, $ua_{il} = 1$ indica que el i-ésimo usuario posee el l-ésimo rol, mientras que $pa_{lk} = 1$ asocia el l-ésimo rol con el j-ésimo permiso.

Definición 2.3.4 (Configuración del Sistema [18]) *Corresponde a la tupla RC = {U, P, DUPA}, abarcando los conjuntos de usuarios y de permisos, y la relación directa que hay entre estos elementos, tal como lo demuestra la Figura 2.5 (a).*

Este concepto es utilizado para referirse a los datos de entrada del problema en estudio. Cabe destacar que algunos enfoques o algoritmos propios de la ingeniería de roles, requieren incorporar otros elementos a los datos de entrada, p. ej., un conjunto de roles desplegados (DR), que corresponde a la tupla $RC = \{U, P, DR, DUPA\}$. Así mismo, se pueda hacer la distinción con $DUPA_{in}$, originando $RC = \{U, P, DR, DUPA_{in}\}$. Para el caso del presente estudio, se utiliza la tupla $RC = \{U, P, DUPA_{in}\}$

Definición 2.3.5 (Estado del Sistema [18]) *Corresponde a la tupla $\Re = \{R, UA, PA\}$, la cual es usada para referirse al resultado del problema en estudio, abarcando así, todos los roles y las relaciones rol-permiso y usuario-rol, tal como lo demuestra la Figura 2.5 (b).*

Hay que recalcar que, es posible obtener RC por medio de $\Re$, pero no a la inversa, ya que no se tiene noción sobre el conjunto de roles R. Cabe destacar que algunos enfoques o algoritmos del RMP, requieren incorporar otros elementos a los datos de salida, p. ej., un conjunto jerárquico de roles (RH) y una relación directa de asignación entre usuarios-permisos $DUPA_{out}$, dando como resultado la tupla $\Re = \{R, UA, PA, RH, DUPA_{out}\}$. En este estudio, se utiliza esta última tupla $\Re$.

Definición 2.3.6 (Sistema Consistente [25]) *$\Re$ será consistente si al obtener RC, $UPA = DUPA_{in}$. Es decir, cada usuario de $\Re$ tiene el mismo conjunto de permisos que en DUP-A_{in}, lo que asegura que no se omitieron, eliminaron o editaron las relaciones entre los permisos y los usuarios.*

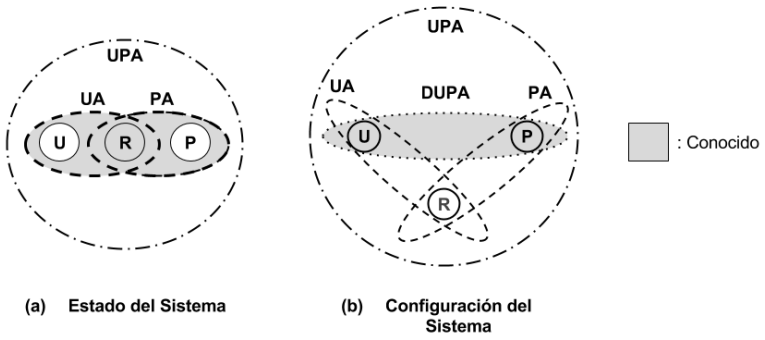


Figura 2.5: Elementos que conforman el estado y la configuración de un sistema *RBAC* consistente.

2.4. Requerimientos

El objetivo principal del *RBAC* consiste en definir un conjunto de roles que sea completo, correcto y eficiente. Para ello, el sistema de control de acceso debe satisfacer los siguientes criterios:

1. **Aprovisionamiento**: los usuarios deben poder acceder a los recursos mínimos (principio de mínimos permisos) que requieran sus actividades, de lo contrario, el desarrollo normal de los procesos se verá afectado.

2. **Consistencia**: todos los empleados de un mismo cargo deben poder acceder a los mismos recursos, salvo ciertos casos excepcionales que, por necesidad de la organización, requieren contar con atribuciones especiales para llevar a cabo su labor.

3. **Seguridad**: este criterio se relaciona con el aprovisionamiento, y establece que los usuarios deben poder acceder únicamente a los recursos asignados.

4. **Mantenibilidad**: el sistema debe ser fácil de administrar y rápido de mantener, con el objetivo de disminuir los recursos y esfuerzos destinados a dichas labores. Para satisfacer este requerimiento, es necesario considerar tres aspectos:

    a) *Minimización del sistema RBAC*: en general, un sistema pequeño es más fácil de mantener y de administrar que uno grande, ya que se debe revisar una menor cantidad de datos.

b) *Interpretabilidad de los roles*: los roles artificiales o poco intuitivos son difíciles de analizar y mantener, en comparación a los roles definidos por la propia organización (roles organizacionales). Idealmente, se esperaría que el sistema *RBAC* incorpore el nombre de los roles organizacionales dentro de la solución, para facilitar la comunicación entre las distintas áreas de la organización, mediante la utilización de los mismos términos. De esta forma, el sistema se alineará con los procesos del negocio.

c) *Generalización de los roles*: un conjunto de roles bien definido y estructurado se adapta fácilmente a los cambios organizacionales. Por lo tanto, el sistema es capaz de asignar nuevos usuarios al conjunto de roles, sin la necesidad de alterar dicho conjunto, y en caso de alterarlo, los cambios son poco frecuentes y de menor envergadura, salvo excepciones (p. ej. agregar o quitar un departamento o área de la organización). De esta forma, el sistema será más fácil de mantener y de administrar que uno que requiera ser reestructurado constantemente.

Estos tres aspectos de la mantenibilidad están interrelacionados, afectando incluso los otros criterios (aprovisionamiento, consistencia y seguridad). En el caso de la interpretabilidad de los roles, al volver los roles más intuitivos, el sistema se vuelve más fácil de administrar. Pero, en general, este cambio requiere alterar el conjunto de roles, lo que podría afectar la mantenibilidad del sistema. Por lo tanto, se puede concluir que, en algunos casos, no es posible satisfacer simultáneamente los tres aspectos de la mantenibilidad, y será necesario priorizar alguno de estos. La decisión sobre qué aspectos priorizar dependerá de las políticas de la organización, la capacidad de los administradores y los recursos destinados para ello. En general, la generalización y la interpretación de los roles involucran una mayor inversión de recursos, y mayores plazos de desarrollo, ya que requiere analizar la información del negocio para la búsqueda del conjunto de roles óptimo. Además, existen pocas herramientas que agilicen y faciliten este proceso. Por otra parte, la minimización del sistema *RBAC* puede ser optimizada fácil y rápidamente mediante la aplicación de algún algoritmo, lo que implica una menor inversión de recursos. Es por esto, que el presente estudio enfocará sus recursos en la minimización del sistema RBAC, pero considerando como requisito secundario, denominar la mayor cantidad de roles en

base a los cargos organizacionales.

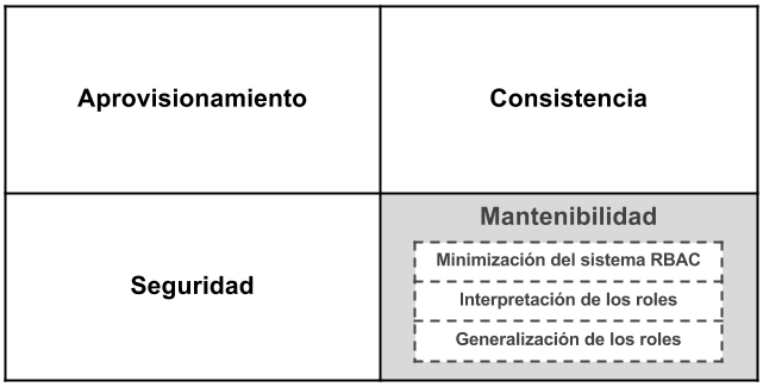


Figura 2.6: Resumen de los requerimientos propios del modelo *RBAC*.

2.5. Ventajas del RBAC

Entre las ventajas del *RBAC*, que lo han vuelto el modelo predilecto de seguridad, se encuentran los siguientes:

1. **Es flexible**. El sistema puede ser adaptado para que considere excepciones. Por ejemplo, si un usuario cambia de cargo, y la organización considera que es necesario que éste mantenga los permisos relacionados con su antigua labor, entonces bastaría que dicho usuario tengo asignado los roles correspondientes a su antigua y nueva labor. Además, este modelo permite adaptar el conjunto de roles de acuerdo a las necesidades de la empresa, ya sea por cambios de políticas internas, implementación de estándares de calidad, cambios en los procesos, surgimiento de nuevos puestos de trabajo, etc.

2. **Es seguro**. Los usuarios no tienen acceso a los recursos a menos que se indique lo contrario. Por lo tanto, sólo los usuarios que en su rol tengan asignado los permisos correspondientes podrán acceder a los respectivos recursos.

3. **Es consistente**. Los usuarios que tengan asignado un mismo rol podrán acceder a los mismos recursos.

4. **En general, es rápido de mantener**. Los permisos relacionados con un rol son, en general, estables, por lo que si un usuario renuncia, cambia de cargo o de área en la misma organización, bastará con modificar los roles asignados. Debido a esto, las tareas más frecuentes corresponden a la edición, eliminación e inserción de usuarios al sistema, lo cual depende directamente de la complejidad y del tamaño del sistema.

5. **Personaliza los tipos de permisos**. Aparte de los clásicos permisos de escritura y lectura, este modelo permite cierta abstracción. Por ejemplo, permisos de exportación e importación de archivos.

6. **Permite establecer restricciones propias del negocio**. Las restricciones son poderosos mecanismos para incorporar políticas organizacionales de alto nivel. La restricción más común es la de los roles mutuamente excluyentes (exclusión), que especifica que un mismo usuario no puede estar asignado a la vez a dos roles específicos. Como ejemplo se considera el caso del sistema principal de una universidad, en donde existe el rol de "profesor" y "alumno". Si un alumno de dicha universidad tuviera asignado ambos roles, entonces podría alterar las notas de los cursos en que esté inscrito. Para evitar esto, se incorpora en el sistema *RBAC* una restricción mutuamente excluyente entre los roles mencionados. Otra restricción importante considera el número de usuarios que debe tener cada rol (cardinalidad). Esto puede ser tanto como un máximo o un mínimo de usuarios, siendo este último el más difícil de implementar, ya que se debe preparar al sistema *AC* para que identifique la violación de esta restricción y responda correctamente. La última restricción a destacar se relaciona con los prerrequisitos. Para el caso de los roles, estos no podrán ser asignados a un usuario, a menos que éste tenga asignado previamente otros roles en particular. Finalmente, cabe destacar que las restricciones mencionadas (exclusión, cardinalidad y prerrequisitos) se pueden relacionar con los roles, los permisos y los usuarios.

7. **Jerarquización de los roles**: Esta estructuración permite reflejar, de forma más natural, las líneas de autoridad y responsabilidad de cada rol. Los roles de mayor autoridad se ubican sobre los demás, por otro lado, los de menor autoridad se posicionan en la parte inferior.

Capítulo 3

Ingeniería de Roles

En este capítulo se describe los diversos enfoques de la ingeniería de roles, detallando para cada caso , los estudios más relevantes en donde se usó la ingeniería de roles para la implementación de sistemas *RBAC* reales.

3.1. Enfoques de la Ingeniería de Roles

El concepto de Ingeniería de Roles (cuya traducción al inglés es "*Role Engineering*" con siglas *RE*) [3]) se basa en la búsqueda del conjunto de roles óptimo (completos, correctos, eficientes y representativos del conjunto de datos). *RE* utiliza las reglas del negocio (restricciones, normas, políticas, operaciones y definiciones presentes en una organización) para nombrar, estructurar y validar dicho conjunto de roles según el modelo *RBAC* [23]. El objetivo principal de *RE* es simplificar la búsqueda de estos roles mediante la utilización de las tecnologías de la información (TI), técnicas, métodos y algoritmos. Hoy en día existen tres tipos de acercamiento al *RE*:

- **Top-down** (cuya traducción al castellano es "*arriba-abajo*") [3]: consiste en analizar las reglas del negocio para así descomponer el conjunto de los roles corporativos en su respectivo conjunto de permisos. Este acercamiento es de utilidad cuando el

sistema de control de acceso recién se está formando, y no existe información relacionada con los permisos de acceso. Cabe destacar que no es rentable aplicar este acercamiento en organizaciones que ya tengan implementado un sistema de acceso, ya que se deben redescubrir los usuarios, los permisos y los roles, junto a la identificación de los casos especiales, labor que requiere una inversión significativa de recursos, que se ve justificada al considerar que el sistema puede tener cientos o miles de usuarios y tipos de permisos de acceso. A pesar de esto, y al elevado costo de implementación, existen organizaciones que han logrado utilizar este acercamiento para mantener sus sistemas *RBAC* (p. ej. [24]).

- **Bottom-top** (cuya traducción al castellano es "*abajo-arriba*") [3]: se enfoca en la utilización de algún método o algoritmo para crear roles a partir de los permisos existentes. Este acercamiento puede ser utilizado siempre y cuando se tenga la información sobre la relación entre el conjunto de usuarios y sus respectivos permisos. Aun cuando los recursos invertidos en este proceso son considerables, siguen siendo menor a lo empleado en la implementación del acercamiento *Top-down*.

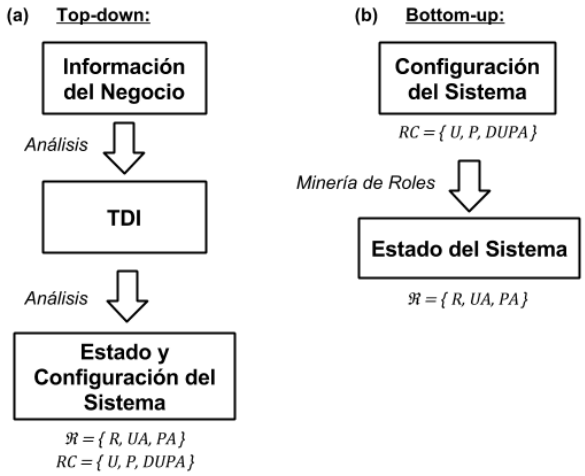


Figura 3.1: Diagrama de flujo de las técnicas (a) *Top-down* y (b) *Bottom-up* del proceso de ingeniería de roles.

- **Mixto** [3]: junta lo mejor de los dos acercamientos mencionados. Utiliza uno o más algoritmos, junto a la información relevante del negocio (conocido como *Top Down*

Information o TDI) para generar los roles en base a los permisos y los usuarios existentes. Aun cuando es la solución preferida para mantener sistemas $RBAC$, requiere una inversión mayor de recursos que el método "*bottom-top*", ya que debe extraer e identificar la información relevante al problema (roles organizacionales, proyectos futuros, estándares, etc.). Esto permite asegurar que los roles sean comprensibles (por los administradores del sistema y el resto de los trabajadores) y estén alineados con el objetivo estratégico de la empresa. Así, los sistemas $RBAC$ serán fáciles de administrar, mantener y adaptar, y por ende, serán seguros. Por lo cual, es recomendable utilizar este acercamiento si la empresa priorice la seguridad y/o la administración de los sistemas.

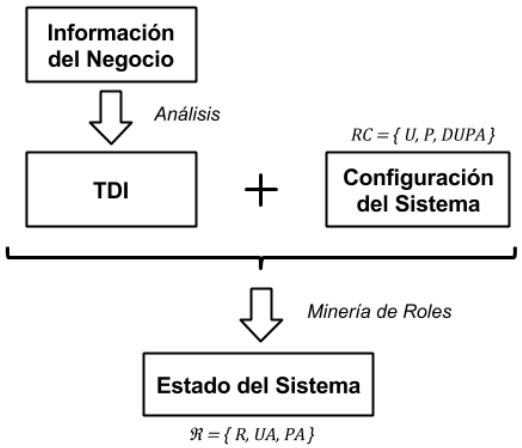


Figura 3.2: Diagrama de flujo de la técnica mixta o híbrida del proceso de ingeniería de roles.

Finalmente, hay que destacar que, en el presente estudio se utiliza el método "*Top-down*" de la ingeniería de roles.

3.2. Estudios

En la literatura hay diversos estudios sobre la utilización de la ingeniería de roles para la implementación o mantención de sistemas $RBAC$, basados en alguna de las metodologías descritas en la sección anterior. Por ejemplo, para el caso del método *Top-down*,

Schaad et al. [24], estudiaron la mantención del sistema *RBAC* de un banco europeo, el banco de Dresdner, con miles de empleados y millones de clientes. Este análisis permitió la identificación de varios problemas relacionados con la asignación errónea de permisos a ciertos roles y a la omisión de ciertas restricciones propias del negocio.

Para el caso del método "*Bottom-top*", hay estudios relacionados a cada una de las técnicas utilizadas en la minería de roles. Por ejemplo, Huang et. al. [9], realizaron la mantención de los sistemas de control de acceso de una empresa dedicada a los servicios financieros, basados en las tecnologías de la información, *SS Corporation*. Dicha mantención utilizó el método de "preprocesamiento del ruido" en 4 sistemas *AC* distintos, donde cada sistema poseía miles de usuarios y cientos de permisos. Este método logró reducir significativamente la cantidad de roles y permisos asignados erróneamente.

Finalmente, para el caso del método híbrido o mixto, los estudios existentes se enfocan en la evaluación de este tipo de técnica, más que en la implementación o mantención del sistema como tal. Por ejemplo, Fuchs et. al. [7] evaluaron esta metodología en conjuntos de datos reales y sintéticos. Experimentos similares fueron llevados a cabo por Molloy et al. [17] y Mandala et al. [14]. Este tipo de metodología aún está en desarrollo, por lo que es esperable que en los próximos años salgan estudios de casos reales que utilicen este tipo de técnica para la implementación o mantención de algún sistema *RBAC*.

Capítulo 4

Minería de Roles

En este capítulo se analiza algunos enfoques del RMP, haciendo hincapié en el proceso de selección de estos, en los conceptos requeridos para la definición formal de cada tipo de problema, y en los algoritmos que resuelven dichas variantes.

El término "*Role Mining*" (cuya traducción al castellano es "Minería de Roles") es utilizado para referirse a la automatización de la ingeniería de roles. Cabe señalar que, en la literatura, es común encontrar publicaciones científicas en donde se abrevie "*Bottom-up role Mining*" como "*Role Mining*" [3], mostrando así lo importante que es este acercamiento en el área de la investigación. Aun cuando existe una noción general sobre qué es la minería de roles, hasta el día de hoy no hay un consenso sobre su definición formal y sobre qué constituye una buena solución al problema. Es por esto que, en la literatura, se puede encontrar distintas definiciones formales al problema de la minería de roles, así también como una gran variedad de métricas de calidad. En 2016, Mitra et al. [15] propusieron un árbol taxonómico que clasifica las variantes del RMP según la estrategia de resolución del problema, las características de los datos de entrada y de salida, el tipo de modelo (determinista o probabilista), y la métrica utilizada para la evaluación de la solución. Un inconveniente de este árbol de clasificación, es la selección del o los enfoques del RMP que mejor se adapten a los requerimientos de la organización, razón por la cual, en ese mismo estudio se expuso un criterio de selección de éstos.

Este capítulo se organiza de la siguiente forma:

- **Subsección 1, Conceptos**: se identifica todos los conceptos requeridos para la definición formal de los enfoques del RMP que son utilizados en el presente texto.
- **Subsección 2, Métrica de Calidad**: se analizan las métricas usadas para evaluar y comparar la calidad de las soluciones propuestas por los diversos algoritmos. La introducción prematura de esta sección se debe a que una variante del RMP considera una de estas métricas para su definición formal.
- **Subsección 3, Clasificación de los Enfoques del RMP**: se introduce parte del árbol de clasificación de los enfoques del RMP, destacando únicamente las variantes que son usadas en el presente texto.
- **Subsección 4, Criterio de Selección**: se describe el criterio utilizado para seleccionar los enfoques del RMP del presente estudio.
- **Subsección 5, Algoritmos**: se describe, en forma general, los algoritmos utilizados en el presente estudio, destacando el tipo de problema que resuelven.
- **Subsección 6, Conjuntos de Datos**: se identifica algunos conjuntos de datos utilizados en la literatura para el estudio de diversos enfoques del RMP. Estos conjuntos de datos serán de utilidad para evaluar y comparar el desempeño de los algoritmos.
- **Subsección 7, RMiner como Herramienta de Evaluación**: se detalla las características de la herramienta utilizada en el presente estudio para la implementación de los algoritmos seleccionados.

4.1. Conceptos

El principal objetivo de esta sección es introducir aquellos conceptos que son necesarios para la definición formal de los problemas estudiados. Además de lo anterior, se analiza cada una de éstos, destacando su utilidad en la definición del problema original y las respectivas variantes. A continuación, se describe cada uno de los conceptos mencionados:

Definición 4.1.1 (Distancia L_1 o Distancia Manhattan [25]) *Sea $d(p,q) = (p-q)$ como la distancia entre dos vectores p y q en un espacio vectorial de k dimensiones. Entonces, se puede definir la distancia L_1 entre dos vectores como la suma de las diferencias absolutas de sus coordenadas, en otras palabras, $d_1(p,q) = \|(p-q)\|_1 = \sum_{i=1}^{k} \|p_i - q_i\|$. Esta definición se puede extender a las matrices del mismo tamaño. Sean A y B matrices de tamaño $m \times n$, entonces se puede definir la distancia L_1 como $d_1(A,B) = \|(A-B)\|_1 = \sum_{i=1}^{m}\sum_{j=1}^{n} \|a_{ij} - b_{ij}\|$, donde a_{ij} y b_{ij} son los elementos de la fila i y en la columna j de las matrices A y B respectivamente. Cabe destacar que la distancia es una magnitud escalar que no considera la dirección de desplazamiento, por lo cual, $d_1(A,B) = d_1(B,A)$.*

De lo anterior, se concluye que la distancia L_1 permite medir el grado de diferenciación o similitud entre dos matrices. Si $d_1(A,B) = 0$, entonces las matrices A y B son idénticas. Caso contrario, las matrices son diferentes.

Como ejemplo se considera las matrices A, B y C, representadas por la Figura 4.1. Tras aplicar la distancia L_1 entre cada una de estas matrices, se obtuvo como resultado la Tabla 4.1. De esta Tabla se desprende que las matrices A y B son idénticas, por otro lado, la matriz C tiene 2 elementos distintos con respecto a A o B, los cuales fueron destacadas en color gris.

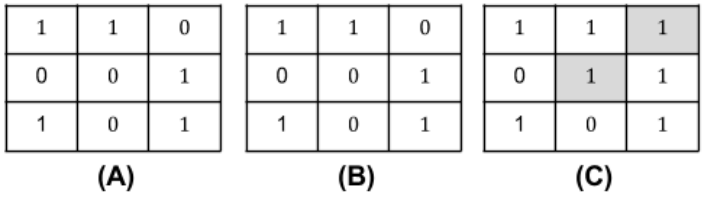


Figura 4.1: Tres ejemplos de matrices binarias de tamaño 3×3.

Tabla 4.1: Distancia L_1 entre cada una de las matrices de la Figura 4.1.

L_1	Magnitud
$d_1(A,B)$	0
$d_1(B,C)$	2
$d_1(A,C)$	2

Finalmente, hay que destacar que este concepto es utilizado para medir la diferencia o la similitud entre las matrices UPA y $DUPA_{in}$. Los enfoques del RMP utilizados en el presente estudio consideran que estas matrices deben ser idénticas.

Definición 4.1.2 (δ-consistencia matricial [25]) *Sean A y B matrices de $m \times n$, entonces se puede definir el nivel de tolerancia δ de la diferencia entre estas dos matrices como $d(A-B) \leq \delta$. De esta forma, A y B son δ-consistentes si y sólo si $\|A-B\| \leq \delta$.*

Para efectos de este documento, se considerará que dicha distancia corresponde a la distancia L_1, por lo cual, $\|A-B\|_1 \leq \delta$.

En base al ejemplo anterior, si $\delta = 0$, entonces se exige que las matrices estudiadas sean idénticas, tal es el caso de las matrices A y B, cuya distancia L_1 es $d_1(A,B) = 0$ y satisface el requerimiento $d_1(A,B) \leq \delta$. Si $\delta = 1$, entonces se exige que, a lo más, las matrices presenten un elemento distinto. Nuevamente, $d_1(A,B)$ es el único que cumple dicha condición. Por último, si $\delta \geq 2$, las tres distancias estudiadas satisfacen el requerimiento expuesto.

Este concepto es utilizado para definir el grado de diferenciación tolerable entre las matrices UPA y $DUPA_{in}$. Por lo tanto, si $\delta = 0$, entonces UPA y $DUPA_{in}$ deben ser idénticos.

Definición 4.1.3 (Jerarquía de Roles [8]) *En inglés es "Role Hierarchy" con siglas RH, y corresponde a un grafo $G(V,E)$ directo y acíclico (sin ciclos) en donde cada vértice $v \in V$ representa un rol $r \in R$ y cada arco o arista $e \in E$ representa la relación directa entre dos roles incidentes. Cada vértice contiene, además de un rol, los usuarios y los permisos asociados a dicho rol, representados por $U(r)$ y $P(r)$ respectivamente. Cada $P(r)$ considera los permisos que fueron otorgados de forma explícita y los que se obtuvieron por herencia. Esto mismo es aplicable para el caso de $U(r)$.*

En el presente texto, se utilizan los operadores $\prec$, $\prec\prec$, y $\preceq$ para representar la herencia directa, indirecta y mixta, respectivamente. En base a lo anterior, se puede representar que cada arista e como un par $\{r, r'\} \in E$, si y sólo si $r_i < r_j$ para $r \neq r'$.

Por otra parte, $G(V,E)$ debe satisfacer los siguientes requerimientos:

1. Sea $i \neq j$ e $1 \leq i, j \leq |R|$, entonces $\forall r_i, r_j \in R$, $r_i \succeq r_j$, si y sólo si $P(r_i) \supseteq P(r_j)$, y $U(r_i) \subseteq U(r_j)$. Esta restricción establece que los permisos se heredan de abajo hacia arriba, mientras que los usuarios lo hacen de arriba hacia abajo. Por lo tanto, los permisos autorizados a r_j también serán concedidos a r_i. Por otra parte, el r_i tendrá asociado los usuarios propios de éste y el de los roles superiores.

2. $\forall r \in R$, $P(r) \neq \emptyset$. En otras palabras, todo rol debe tener al menos un permiso asociado. Los roles sin permisos no están permitidos.

3. $\exists\, r \in R$ tal que $\forall r' \in R - \{r\}$, $r_i \preceq r_j$. Este requerimiento establece que cada rol debe estar conectado directamente a, al menos, otro rol en el grafo. Los roles aislados no están permitidos. Esta restricción establece la existencia de un único "*super*" rol el cual tiene asociado todos los permisos existentes y que es utilizado para prevenir que los demás roles se desconecten del grafo jerárquico.

   Finalmente, hay que destacar que este concepto es utilizado para la definición de las variantes del RMP que incorporan el conjunto RH al estado del $RBAC$.

Definición 4.1.4 (Jerarquía de Roles Completa [8]) *En inglés es "Complete Role Hierarchy" con siglas CRH, y corresponde a un caso especial de RH en el que se satisface la siguiente restricción:*

- $\forall r_i, r_j \in R$ con $i \neq j$ e $1 \leq i, j \leq |R|$. Si $r_i \succeq r_j$, entonces pueden ocurrir tres situaciones:

  (a) $\exists\, R' \subseteq R$, con $R' = \{r_1, r_2, \ldots, r_x\} \neq \emptyset$, $x \neq \{i, j\}$ y $1 \leq x \leq |R|$, tal que, $\{(r_i, r_n), \ldots, (r_{x-1}, r_x), (r_x, r_j)\} \subseteq E$, con $n \neq \{x, i, j\}$ y $1 \leq n \leq |R|$.

  (b) $(r_i, r_n) \in E$.

  (c) Ocurre (a) y (b) a la vez.

Esta limitación establece que todas las relaciones de herencia, entre cualquier par de roles, deben estar contenidas en $G(V, E)$, ya sea por herencia directa, indirecta o mixta. Esto permite asegurar que la herencia en $G(V, E)$ sea consistente.

Finalmente, y al igual que en el caso anterior, este concepto es utilizado para la definición de las variantes del RMP que incorporan el conjunto RH al estado del $RBAC$.

Definición 4.1.5 (Jaccard Coefficient (JC) [26]) *En castellano se traduce como "Coeficiente de Jaccard", pero también es conocido como "Índice de Jaccard", y corresponde a una métrica estadística utilizada para medir el grado de similitud y de discrepancia entre dos conjuntos. En particular, el coeficiente de Jaccard (JC) mide el el grado de similitud, y se obtiene mediante la Ecuación 4.1, mientras que la distancia de Jaccard (JD) evalúa la discrepancia, y es representada por la Ecuación 4.2.*

$$JC_{AB} = \frac{|A \cap B|}{|A \cup B|} \tag{4.1}$$

$$JD_{AB} = \frac{|A \cap B|^c}{|A \cup B|} \tag{4.2}$$

Donde $|x|^c$ corresponde al complemento del conjunto x. Para el caso de $|A \cap B|^c$, el complemento del conjunto de elementos similares corresponde al conjunto *divergente*.

En base a lo anterior, se desprende que: $JD = 1 - JC$ o $JC = 1 - JD$, con $0 \leq JC, JD \leq 1$. Para el caso en que $JC = 1$ (o $JD = 0$), se desprende que los conjuntos que se comparan son idénticos, mientras que $JC = 0$ (o $JD = 1$) indica que éstos no poseen ningún elemento en común, y por ende, son completamente distintos.

Para una mayor comprensión, se considera el ejemplo expuesto en [26], en donde $A = \{a, b, c, d, e\}$ y $B = \{a, d, e, f, g\}$, de modo que $|A \cap B| = |a, d, e| = 3$ y $|A \cup B| = |a, b, c, d, e, f, g| = 7$. Por lo tanto, $JC = \frac{3}{7} = 0{,}429$ y $JD = 1 - 0{,}429 = 0{,}571$. Cabe destacar que JD puede ser obtenido directamente, al considerar la discrepancia entre estos conjuntos, es decir, $JD = \frac{4}{7} = 0{,}571$.

Finalmente, este concepto es utilizado para comparar la similitud entre dos roles o entre dos conjuntos de éstos.

4.2. Métrica de Calidad

En esta sección se analizan las métricas que se usaron en el presente estudio para medir la calidad de las soluciones propuestas por los algoritmos seleccionados.

4.2.1. Ponderación de la Complejidad Estructural (WSC)

En los primeros estudios de la minería de roles, se evaluaba la calidad de las soluciones en base a su capacidad para optimizar la función objetivo $f(x)$ del respectivo enfoque del RMP, para así poder comparar el resultado obtenido con otras soluciones del mismo tipo de problema. Por lo tanto, dicha métrica era distinta para cada tipo de problema, y por ende, no permitía comparar resultados entre distintos problemas, a menos que las funciones objetivos de estos sean idénticas. Es por esta razón, que las investigaciones en los últimos años se han centrado en buscar una métrica que permita comparar los soluciones de distintos enfoques del RMP. Una de las métricas más prometedoras, y que es la que se utilizará en el presente estudio, corresponde a la *Ponderación de la Complejidad Estructural* (WSC).

Weighted Structural Complexity (WSC), cuya traducción al castellano es *Ponderación de la Complejidad Estructural*, es una de las métricas de calidad más utilizadas en la literatura [17], debido a que la flexibilidad de ésta, permite adaptar los elementos que son considerados en la evaluación, con la finalidad de satisfacer diferentes objetivos de optimización. Cabe mencionar que este criterio de calidad no considera, dentro de su evaluación, la interpretabilidad y la generalización de los roles.

El WSC consiste en una combinación lineal de las dimensiones de cada elemento del estado del $RBAC$ ($\Re = \{R, UA, PA, RH, DUPA_{out}\}$). En $\Re$, los elementos R, UA, PA no deben ser conjuntos vacíos, por otra parte, RH y $DUPA_{out}$ pueden ser conjuntos vacíos, lo cual depende del algoritmo y del enfoque utilizado.

Cada elemento de $\Re$ tiene asociado una ponderación o peso ($w \in Q^+ \cup \{\infty\}$), tal como lo representa el vector $W = (w_r, w_u, w_p, w_h, w_d)$, en donde w_r, w_u, w_p, w_h y w_d corresponden al peso de los conjuntos y de las relaciones de asignaciones R, UA, PA, RH y

$DUPA_{out}$ respectivamente. En base a lo expuesto, es posible definir formalmente el concepto de wsc.

Definición 4.2.1 (Ponderación de la Complejidad Estructural (WSC) [17]) *Dado un vector de peso* $W = (w_r, w_u, w_p, w_h, w_d)$*, con* w_r*,* w_u*,* w_p*,* w_h *y* $w_d \in Q^+ \cup \{\infty\}$*, y un estado del RBAC* $\Re = \{R, UA, PA, RH, DUPA_{out}\}$*, se define la ponderación de la complejidad estructural como* $wsc(\Re, W) = \Re \cdot W = w_r \times |R| + w_u \times |UA| + w_p \times |PA| + w_h \times |tr(RH)| + w_d \times |DUPA_{out}|$*, donde* $|x|$ *denota la cantidad de asignaciones o elementos del conjunto* x*, y* $tr(RH)$ *es la reducción transitoria de la jerarquía de roles* RH*.*

La reducción transitoria de la jerarquía de roles RH corresponde al conjunto mínimo de relaciones que se puede obtener con la misma jerarquía. Por ejemplo, si $RH = \{(r_1 \prec r_2), (r_2 \prec r_3), (r_1 \prec r_3)\}$, entonces $tr(RH) = \{(r_1 \prec r_2), (r_2 \prec r_3)\}$, ya que la relación $(r_1 \prec r_3)$ puede ser inferida mediante la regla de transitividad $(r_1 \prec r_2 \wedge r_2 \prec r_3 \rightarrow r_1 \prec r_3)$, tal como lo representa la Figura 4.2, en donde la flecha punteada corresponde a la relación inferida por la propiedad de transitividad.

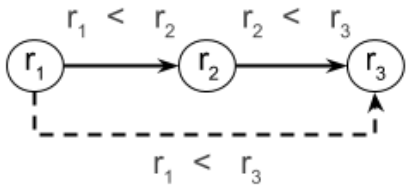


Figura 4.2: Diagrama sobre la transitividad entre dos conjuntos.

En base a estas definiciones, se desprende que, la solución de mejor calidad será la que obtenga la menor magnitud de wsc posible en base al vector de peso W indicado. La mayor desventaja del wsc, es que éste no especifica los valores que debe tener W, situación que empeora al considerar que, actualmente, no existe un consenso entre los investigadores sobre el vector de peso que debería ser usado en la práctica. Así, el vector de peso W debe ser obtenido a partir de algún estudio o estándar existente en la literatura, o se puede basar en la experiencia del propio administrador del $RBAC$.

La Tabla 4.2 resume la estructura de los vectores de peso W más destacados de la literatura. En dicha tabla, $c_i \geq 1$, con $i = r, u, p, h, p$, cuyo magnitud dependerá de la priorización

que la organización de a cada elemento del $RBAC$. Por otra parte, se utiliza el ∞ para denotar un número grande, el cual impide la existencia de un conjunto o relación de asociación en particular. Por ejemplo, con $w_h = \infty$, se impide la existencia de una jerarquía de roles, forzando un estado $RBAC$ plano, por otra parte, $w_h = 0$ indica que la jerarquía de roles no influye en la evaluación, mientras que para cualquier otro valor, el conjunto RH afecta al resultado. Considerando otro ejemplo, con $w_d = \infty$, se prohíbe, en el conjunto de salida, la existencia de la relación de asignación directa entre los usuarios y sus respectivos permisos ($DUPA_{out}$), satisfaciendo de este modo, el estándar básico del RBC. Para la formulación del W es necesario conocer las siguientes propiedades aritméticas relacionadas con el ∞: $\infty \times 0 = 0$; $n \times \infty = \infty$, con $n \in Q^+$; y $n + \infty = \infty$, con $n \in Q^+ \cup \{\infty\}$. La

Tabla 4.2: Vectores de peso más reconocidos en la literatura.

w_r	w_u	w_p	w_h	w_d	Objetivo
c_r	0	0	∞	∞	Basic-RMP, sin jerarquía
c_r	0	0	0	∞	Basic-RMP, jerarquía no influye
0	c_u	c_p	∞	∞	Edge-RMP, sin jerarquía
0	c_u	c_p	0	∞	Edge-RMP, jerarquía no influye
0	c_u	c_p	c_h	∞	Edge-RMP, jerarquía influye
c_r	c_u	c_p	c_h	c_d	Minimizar Costo

Tabla 4.3 resume la estructura de los vectores de peso W que serán usados en el presente estudio, relacionándolos con su respectivo objetivo. Estos vectores están basados en el estudio realizado por Molloy et. al. [18], en donde se evaluó el desempeño de diversos algoritmos del RMP ante varios conjuntos de datos reales aplicando los vectores de peso mencionados. Como se mencionó con anterioridad, el vector W puede adaptarse a las

Tabla 4.3: Vectores de peso utilizados en el presente estudio.

w_r	w_u	w_p	w_h	w_d	Objetivo
0	1	1	1	∞	Edge-RMP, jerarquía influye
1	0	0	0	∞	Basic-RMP, jerarquía no influye
1	1	1	1	1	Minimizar Costo

necesidades de la organización. Por ejemplo, $W = (1,1,2,1,2)$ considera que las acciones relacionada con los permisos son más costosas que las demás (por el doble).

Finalmente, este criterio separa la evaluación de la solución, del proceso de optimización de la función objetivo $f(x)$, permitiendo así, comparar los resultados obtenidos por distintos enfoques del RMP. En estos casos, se debe tener la precaución de analizar el estado del $RBAC$ de cada algoritmo en estudio, ya que no es posible prohibir la presencia de un elemento que ya es considerado dentro de la solución, p. ej., el algoritmo "*Graph Optimization*" (GO) [27], siempre asigna un rol para cada usuario, por lo que si $w_u = \infty$, se obtendrá como resultado $wsc = \infty$.

Hay que recalcar que, en algunos enfoques del RMP, existe un vector W que se alinea perfectamente con la función objetivo del problema. Razón por la cual, es posible reformular algunos problemas en función del "wsc", situación que será estudiada posteriormente.

4.2.2. Asignaciones no cubiertas

Se usa la Ecuación 4.3 para medir las asignaciones directas entre usuarios y permisos que no es cubierto por la solución propuesta (ANC). Esta métrica establece que, si $ANC \approx 1$, entonces los roles generados no son representativos del conjunto de datos dado.

$$ANC = \frac{|DUPA_{out}|}{|DUPA_{in}|} \tag{4.3}$$

El modelo RBAC establece que ANC debería ser 0, pero, en la práctica, los algoritmos pueden fallar, ya sea por algún error en la ejecución del algoritmo o por la omisión de alguna circunstancia dentro de la lógica de dicho algoritmo, provocando que una o más asignaciones de permisos no puedan ser representadas por el conjunto de roles existentes. Así, las asignaciones presentes en $DUPA_{out}$ deben ser relacionadas manualmente con los respectivos roles y usuarios, existiendo la posibilidad de incrementar el número de roles. En base a esto, es deseable que $ANC \approx 0$.

4.3. Enfoques del RMP utilizados

En esta sección, se describe brevemente los enfoques del RMP que son utilizados en el presente estudio. En [15], un árbol taxonómico de múltiples niveles que clasifica los enfoques existentes del RMP según el modelo utilizado, la naturaleza de la configuración del $RBAC$ y las métricas utilizadas en la evaluación de éstos. Debido a la gran cantidad de variantes existentes, se debió dividir el diagrama de clasificación en múltiples partes pero, para efectos del presente estudio, sólo se analiza una de éstas, la cual considera los enfoques usados en el presente texto, el resto podrá ser revisado en [15]. La Figura 4.3 representa la parte central del árbol taxonómico, y considera los dos primeros niveles de clasificación; en el primer nivel se considera el tipo de modelo, mientras que en el segundo, la naturaleza de la configuración $RBAC$.

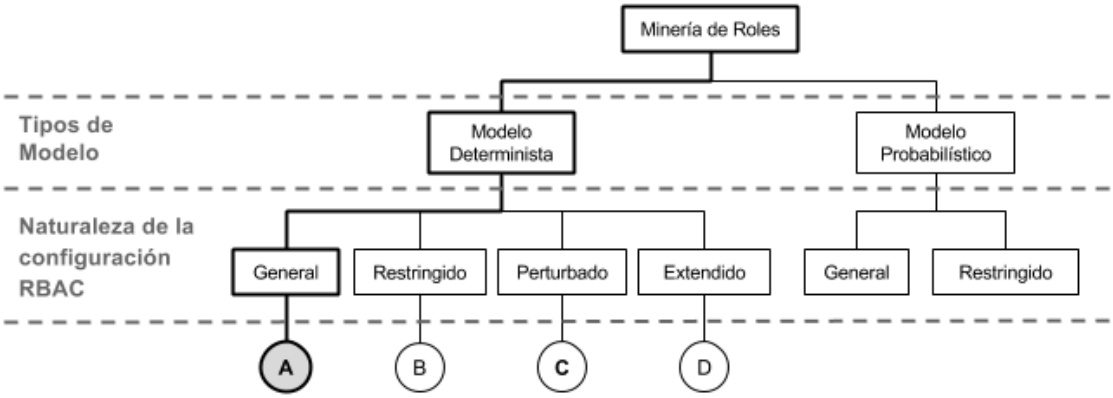


Figura 4.3: Árbol taxonómico de los enfoques del RMP.

Existen dos tipos de modelos: el determinista, cuyas soluciones son completamente predecibles e invariables ante los mismos datos de entrada; y el probabilístico que, en contraste, considera el azar en el proceso de modelamiento, por lo cual, tiene distintos resultados ante los mismos datos de entrada, junto a la existencia de un margen de error relacionado con la predicción de los eventos o sucesos. Dentro del modelo determinista, los problemas se subclasifican en: (A) *General*, que corresponden a las variantes del RMP que buscan únicamente minimizar una o más métricas; (B) *Restringido*, el cual considera una o más restricciones en la formulación del problema; (C) *Perturbado*, optimiza un

conjunto de roles existentes en base a una o más métricas; (*D*) *Extendido*, el cual incorpora otros elementos especiales al modelo clásico del $RBAC$. Se destacó en gris y con bordes más gruesos, la clasificación del RMP que es considerada en el presente estudio, que corresponde a la categoría "*General*" (*A*) del modelo "*Determinista*".

Por otra parte, dentro del modelo probabilístico, los problemas se pueden subclasificar en: *General* y *Restringido*, cuyas clasificaciones poseen las mismas definiciones que en el caso del modelo determinista.

En la Figura 4.4, se destaca en gris y con bordes más gruesos, aquellos enfoques que serán considerados en el presente estudio. Estos son: *RMP-Básico*, *Edge-RMP*, *RHMP* y *WSCO*. Sin adelantar mayores detalles, *WSCO* es el enfoque que mejor se adaptó a los requerimientos organizacionales, y es el que será usado en el presente estudio.

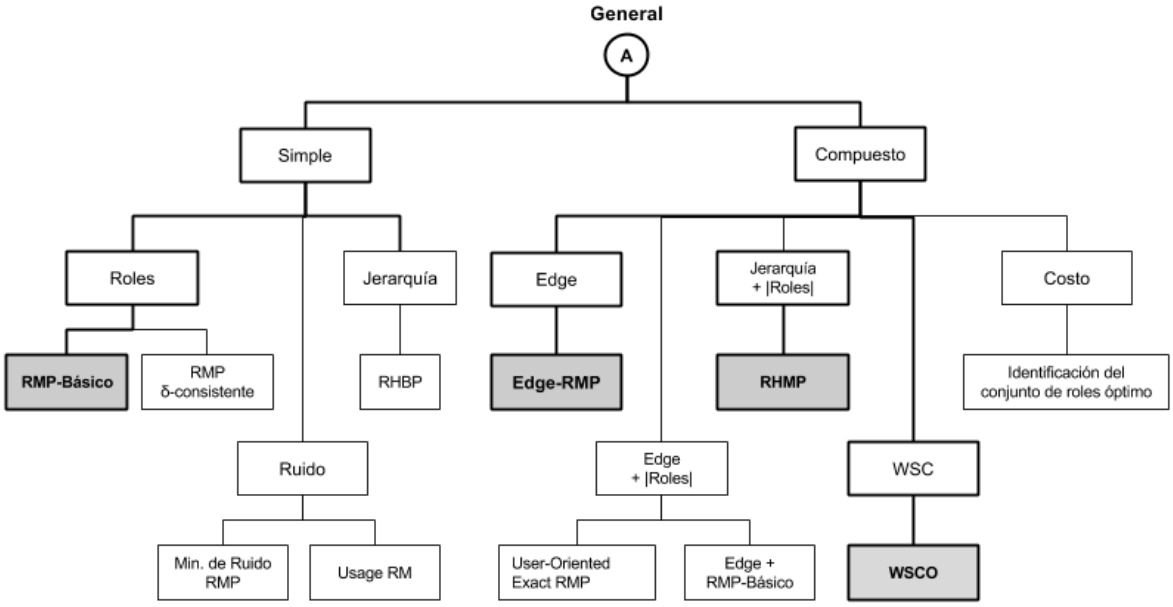


Figura 4.4: Continuación del árbol taxonómico de la Figura 4.3 bajo la categoría *General*.

Para complementar el análisis de los enfoques seleccionados, se utilizará un mismo conjunto de datos de entrada ($DUPA_{in}$), representados por la Tabla 4.4, para la resolución de cada tipo de problema, destacando, según sea el caso, la naturaleza de las solución y las características mismas del sistema.

Tabla 4.4: $DUPA_{in}$ usado para estudiar los enfoques del RMP.

Usuarios	Permisos						
	p_1	p_2	p_3	p_4	p_5	p_6	p_7
u_1	1	1	0	0	1	1	1
u_2	1	1	0	0	1	0	1
u_3	0	0	0	1	1	1	1
u_4	1	1	1	1	1	1	1
u_5	1	1	1	0	0	1	1

En particular, el problema básico de la minería de roles minimiza únicamente el número de roles ($|R|$). De esta forma, se busca el conjunto mínimo de roles, que sea representativo del conjunto de datos dado.

Definición 4.3.1 (Problema Básico de la Minería de Roles [25]) *En inglés es conocido como "Basic-RMP" y se traduce al castellano como "RMP-Básico". Dado un conjunto de usuarios U, un conjunto de permisos P, y una relación de asignación directa entre los usuarios y sus respectivos permisos $DUPA_{in}$, se debe encontrar el conjunto de roles R y las relaciones de asignaciones usuario-rol (UA) y rol-permiso (PA), que sean consistentes con $DUPA_{in}$ y que minimicen el número de roles $|R|$.*

Para la definición de este y los demás problemas, se considera que $|U| = m$, $|P| = n$, y $|R| = k$. Al representar el problema mediante su forma matricial, se puede definir el problema como la minimización de la función objetivo $f(x) = k$, bajo la condición $\|UA_{m\times k} \otimes PA_{k\times n} - DUPA_{m\times n}\|_1 = \delta$, con $\delta = 0$, donde $\otimes$ corresponde al operador de multiplicación matricial, lo que implica que los conjuntos UPA y $DUPA_{in}$ deben ser idénticos.

Desde la perspectiva del $RBAC$, la solución del *RMP-Básico* satisface parcialmente los requerimientos de éste, ya que se enfoca únicamente en la minimización del sistema, ignorando la interpretación y la generalización de los roles.

El *RMP-Básico* es el primer acercamiento a una definición formal del problema de la minería de roles. A partir de esta definición, han surgido diferentes variantes de este problema, que difieren en la función objetivo o en los datos de entrada. Cabe destacar que todas

estas variantes requieren, al menos, los mismos datos de entrada que el *RMP-Básico*, y entregan, al menos, los mismos conjuntos de salida que éste.

Finalmente, si se resuelve el *RMP-Básico* utilizando como datos de entrada el conjunto $DUPA_{in}$ descrito por la Tabla 4.4, se obtiene como resultado los conjuntos UA y PA, representados por las Tablas 4.5 y 4.6 respectivamente. De dichas tablas, se desprende que se requieren, al menos, 4 roles para el modelamiento del sistema. Mediante la multiplicación matricial es posible confirmar que $DUPA_{in} = UPA$.

Tabla 4.5: UA generado a partir del $DUPA_{in}$ de la Tabla 4.4 para el caso del *RMP-Básico*.

Usuarios	Roles			
	r_1	r_2	r_3	r_4
u_1	1	0	1	0
u_2	0	0	1	0
u_3	0	1	0	0
u_4	1	1	0	1
u_5	1	0	0	1

Tabla 4.6: PA generado a partir del $DUPA$ de la Tabla 4.4 para el caso del *RMP-Básico*.

Roles	Permisos						
	p_1	p_2	p_3	p_4	p_5	p_6	p_7
r_1	1	1	0	0	0	1	1
r_2	0	0	0	1	1	1	1
r_3	1	1	0	0	1	0	1
r_4	1	1	1	0	0	1	1

Los enfoques del RMP que fueron seleccionados y que pertenecen a la categoría "*Compuesto*" de la rama "*Determinista*" son: *Edge-RMP*, $RHMP$ y $WSCO$. Lo que tienen en común estas variantes es que buscan minimizar más de un elemento del $RBAC$. En particular, *Edge-RMP* busca optimizar la cantidad de asignaciones presentes en el sistema.

Definición 4.3.2 (Edge-RMP [13]) *Dado un conjunto de usuarios U, un conjunto de permisos P y una relación de asignación directa entre los usuarios y sus respectivos permisos*

$DUPA_{in}$, se debe encontrar el conjunto de roles R y las relaciones de asignaciones usuario-tol (UA) y rol-permiso (PA), que sean consistente con $DUPA_{in}$ y que minimicen la suma del tamaño de la cantidad de asignaciones de UA y PA.

Utilizando la misma representación matricial que el caso anterior, se puede definir el problema como la minimización de la función objetivo $f(x) = |UA| + |PA|$, bajo la condición $\|UA_{m\times k} \otimes PA_{k\times n} - DUPA_{m\times n}\|_1 = \delta = 0$, lo que implica que los conjuntos UPA y $DUPA_{in}$ deben ser idénticos. A primera vista, se podría pensar que este enfoque está relacionado con el *RMP-Básico* pero, en realidad, estos tienen objetivos totalmente opuestos. En el *RMP-Básico*, al disminuir la cantidad de roles, disminuye la cantidad de asignaciones en PA, aumentando a su vez, la cantidad de asignaciones en UA. Por otra parte, en el *Edge-RMP*, disminuye la cantidad de asignaciones en UA, aumentando la cantidad de asignaciones en PA, lo que podría ocasionar un incremento en la cantidad de roles.

Para comprobar esto, se considera el siguiente ejemplo: inicialmente se dispone de la matriz $M(DUPA_{in})_{6\times 3}$ representada por la Figura 4.5 (a). Tras descomponer $DUPA_{in}$ para el problema *RMP-Básico*, se obtuvo como resultado las matrices $M(UA)_{6\times 2}$ y $M(PA)_{2\times 3}$, representados por las Figuras 4.5 (b) y (c) respectivamente. En este caso, $|UA| + |PA| = 10 + 4 = 14$ y $|R| = 2$. Por otra parte, al descomponer $DUPA_{in}$ para el problema *Edge-RMP* se obtuvo como resultado las matrices $M(UA)_{6\times 3}$ y $M(PA)_{3\times 3}$, representados por las Figuras 4.5 (d) y (e). En este último caso, se obtuvo $|UA| + |PA| = 6 + 7 = 13$ y $|R| = 3$. En base a los resultados obtenidos, se comprueba que el *Edge-RMP* optimiza la cantidad de asignaciones de los conjuntos que conforman UPA, pero esto puede significar un aumento en la cantidad de roles. Por otra parte, el conjunto de roles propuesto por *Edge-RMP* es más fácil de administrar y de mantener que el *RMP-Básico*, ya que cada usuario de *Edge-RMP* tiene asociado una menor cantidad de roles en comparación con el *RMP-Básico*.

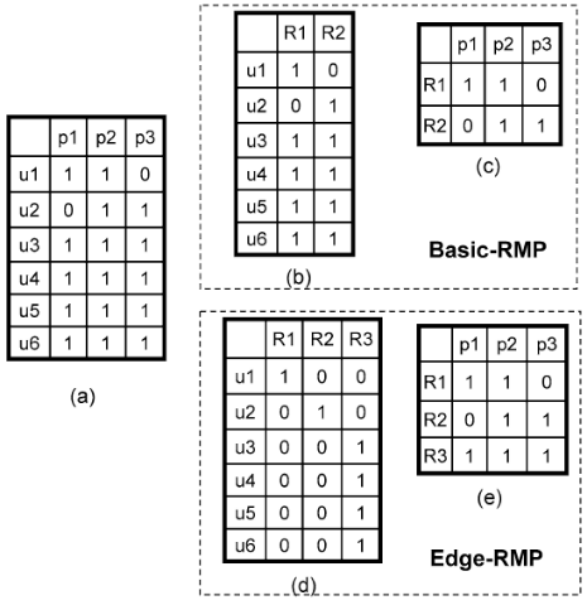


Figura 4.5: Soluciones de los problemas *RMP-Básico* y *Edge-RMP* para un $DUPA_{in}$ en particular.

Desde la perspectiva del $RBAC$, la solución del *Edge-RMP* satisface parcialmente los requerimientos de éste, ya que se enfoca únicamente en la minimización del sistema, ignorando la interpretación y generalización de los roles. Pero, a diferencia del *RMP-Básico*, mejora la mantenibilidad y la administración del sistema.

Finalmente, si se resuelve el *Edge-RMP* utilizando como datos de entrada el conjunto $DUPA_{in}$ descrito por la Tabla 4.4, se obtiene como resultado los conjuntos UA y PA, representadas por las Tablas 4.7 y 4.8 respectivamente. De la Tabla 4.8 se desprende que se requieren, al menos, 4 roles para el modelamiento del sistema, resultado que coincide con la cantidad de roles requeridas por el *RMP-Básico*. Sin embargo, los conjuntos UA y PA son completamente distintos, es más, en *RMP-Básico* $|UA| + |PA| = 26$, pero en *Edge-RMP* $|UA| + |PA| = 21$. Así, se demuestra que el *Edge-RMP* minimiza aún más el sistema $RBAC$, en comparación al *RMP-Básico*.

Tabla 4.7: UA generado a partir del $DUPA_{in}$ de la Tabla 4.4 para el caso del *Edge-RMP*.

Usuarios	Roles			
	r_1	r_2	r_3	r_4
u_1	0	0	1	1
u_2	0	0	1	0
u_3	1	0	0	0
u_4	1	1	0	0
u_5	0	1	0	1

Tabla 4.8: PA generado a partir del $DUPA_{in}$ de la Tabla 4.4 para el caso del *Edge-RMP*.

Roles	Permisos						
	p_1	p_2	p_3	p_4	p_5	p_6	p_7
r_1	0	0	0	1	1	1	1
r_2	1	1	1	0	0	0	0
r_3	1	1	0	0	1	0	1
r_4	0	0	0	0	0	1	1

Para el caso del *Problema de la Minería de Roles Jerárquicos*, se busca optimizar el conjunto jerárquico de roles, lo que implica minimizar la cantidad de roles y las relaciones de herencia entre éstos.

Definición 4.3.3 (Role Hierarchy Mining Problem (RHMP) [8]) *En castellano se traduce como "Problema de la Minería de Roles Jerárquicos". Dado un conjunto de usuarios U, un conjunto de permisos P y una relación de asignación directa entre los usuarios y sus respectivos permisos $DUPA_{in}$, se debe encontrar el conjunto de roles R, las relaciones de asignaciones usuario-rol (UA) y rol-permiso (PA), y una jerarquía de roles completa, $RH = G(V,E)$, tal que RH es 0-consistente con $DUPA_{in}$ y minimiza la suma $|R|+|E|$.*

Utilizando la misma representación matricial que los casos anteriores, se puede definir este problema como la minimización de la función objetivo $f(x) = |R|+|E|$, bajo la condición $\|UA_{m\times k} \otimes PA_{k\times n} - DUPA_{m\times n}\|_1 = \delta = 0$, lo que implica que los conjuntos UPA y

$DUPA_{in}$ deben ser idénticos.

En base a lo anterior, se puede concluir que existe cierta similitud entre *RHMP* y *RMP-Básico*, ya que en ambos casos se busca minimizar el conjunto de roles. La diferencia entre estos radica en que *RMP-Básico* se enfoca en hallar la menor cantidad de roles posibles, sin establecer una relación jerárquica entre ellos ($RBAC$ plano), mientras que $RHMP$ encuentra la jerarquía más compacta de dichos roles. Cabe destacar que la existencia del conjunto RH facilita la administración y comprensión de los roles, ya que permite estructurar la información de una forma más natural y lógica desde la perspectiva organizacional, siendo esto una ventaja sobre el *RMP-Básico*.

Al comparar estos dos problemas, se desprende que $|R|_{RHMP} \geq |R|_{RMP-Básico}$, ya que en $RHMP$ es necesario definir un rol con todos los permisos asociados ("*super rol*") para mantener la jerarquía. De este modo, $|R|_{RHMP} = |R|_{RMP-Básico}$, si y sólo si, el *RMP-Básico* considera dicho *super* rol en su estructura. Por lo tanto, una solución del *Edge-RMP* puede ser considerada como una respuesta del *RHMP*, siempre y cuando se incluya el *super rol*. Por otra parte, para considerar el *RMP-Básico*, se debe generar el respectivo conjunto jerárquico de roles, el cual además, debe poseer el *super rol*.

Finalmente, al resolver el *RHMP* utilizando como datos de entrada el conjunto $DUPA_{in}$ descrito por la Tabla 4.4, y considerando los resultados de *Edge-RMP* y de *RMP-Básico*, se obtiene dos posibles resultados, cuyos conjuntos RH son representados en la Figura 4.6. De dicha figura, se desprende que, para ambos casos, $|R| = 5$ y $|E| = 4$, por lo tanto, $f(x) = 9$. Así, se comprueba que *RHMP* requiere, al menos, un rol más que el *RMP-Básico*, para el caso en que este último no considere el *super* rol.

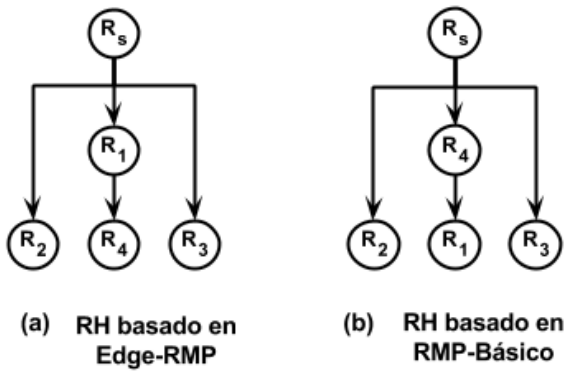


Figura 4.6: RH generado por *Edge-RMP* y *RMP-Básico* a partir del $DUPA_{in}$ de la Tabla 4.4.

En [17] se propone una variante del RMP, basada en la optimización de la métrica $wsc(\Re, W)$.

Definición 4.3.4 (Weighted Structural Complexity Optimization [17]) *(WSCO) En castellano se traduce como "Optimización de la Ponderación de la Complejidad Estructural". Dado un conjunto de usuarios U, un conjunto de permisos P, una relación de asignación directa entre los usuarios y sus respectivos permisos $DUPA_{in}$, y un vector de peso $W = (w_r, w_u, w_p, w_h, w_d)$, se debe encontrar el conjunto de roles R y las relaciones de asignaciones usuario-rol (UA) y rol-permiso (PA), que sean consistentes con $DUPA_{in}$ y que minimicen la métrica $wsc(\Re, W)$, en donde $\Re = \{R, UA, PA, RH, DUPA_{out}\}$ y corresponde al estado del $RBAC$.*

En especial, esta variante del RMP no posee un objetivo en particular, sino más bien es un conjunto de tipos de problemas. En la Tabla 4.2 se relaciona algunos de los problemas más relevantes en la literatura con su respectivo vector de peso, mientras que la Tabla 4.3 resume la estructura de los vectores de peso W que serán usados en el presente estudio. Hay que destacar que algunos algoritmos no se ven influenciados por los distintos valores de wsc, ya que por su diseño entregan siempre la misma solución en base a un conjunto de datos dado.

La Tabla 4.9 sintetiza los enfoques estudiados en el presente texto, destacando la consistencia de UPA, los datos de entrada y de salida, y la función objetivo $f(x)$ para cada tipo

de problema. En particular, para el caso del $WSCO$; RH y $DUPA_{out}$ pueden no pertenecer al conjunto de datos de salida. La presencia u omisión de estos elementos dependerá del tipo de enfoque y del algoritmo en estudio.

Tabla 4.9: Resumen de los enfoques estudiados en el presente texto.

Problema	Datos de entrada	Datos de salida	$f(x)$
RMP-Básico	U, P y $DUPA_{in}$	R, UA y PA	$\|R\|$
Edge-RMP	U, P y $DUPA_{in}$	R, UA y PA	$\|UA\| + \|PA\|$
RHMP	U, P y $DUPA_{in}$	R, UA, PA y RH	$\|R\| + \|E\|$
WSCO	U, P, W y $DUPA_{in}$	R, UA, PA, RH y $DUPA_{out}$	wsc

4.4. Criterio de selección

En esta sección se analiza el criterio utilizado para la selección de las estrategias o enfoques de la minería de roles que mejor se adapten a las necesidades de la organización. Este criterio considera los requerimientos de la organización para obtener una lista de enfoques candidatos en base al árbol de clasificación estudiado con anterioridad, quedando a decisión del propio administrador, la selección de éstos, ya sea en base al presupuesto, a las prioridades de la organización o ambas valoraciones.

La existencia de diferentes tipos de enfoques para resolver el problema de la minería de roles, da indicios de que las características y los requerimientos del sistema $RBAC$ son distintos para cada organización. El criterio de selección, propuesto en [15], es representado mediante un diagrama de decisión, en donde cada resultado corresponde a un conjunto de posibles enfoques (de la categoría "*Naturaleza de la configuración del RBAC*"), que pueden ser utilizados para la mantención o implementación del sistema $RBAC$, según corresponda. Este diagrama fue dividido en tres partes, dando como resultado las Figuras 4.7, 4.8 y 4.9. La Figura 4.7 corresponde a la parte principal del diagrama, y establece la división entre la implementación y la mantención del sistema $RBAC$. La Figura 4.8 representa el caso en que se está manteniendo el sistema, mientras que la Figura 4.9

corresponde al caso en que se está implementando; en ambos casos se evalúa si el sistema requiere otras especificaciones o si posee restricciones. Cabe destacar que este criterio sólo considera los enfoques del RMP, basados en algún modelo determinista.

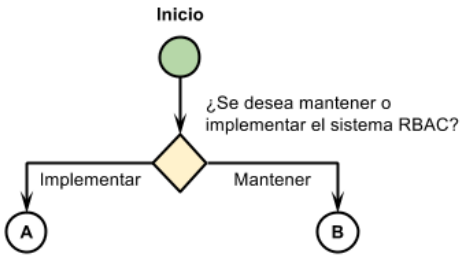


Figura 4.7: Diagrama de decisión que permite seleccionar los enfoques del RMP más aptos.

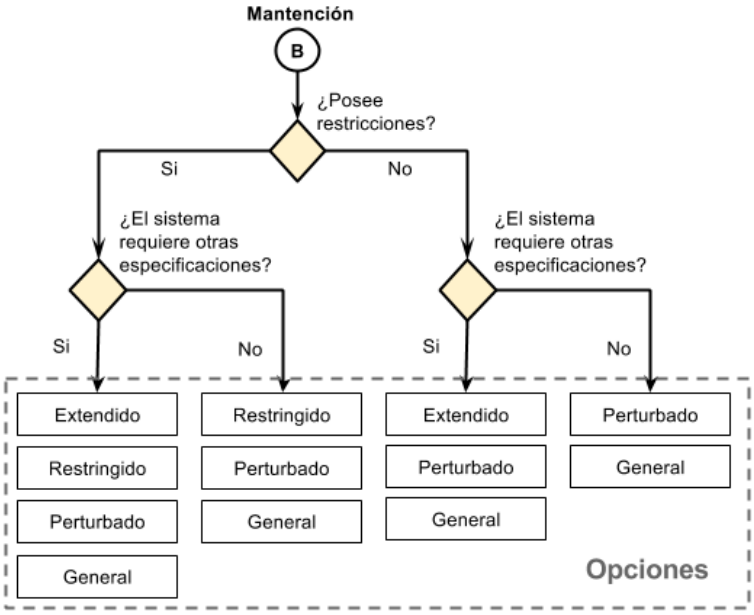


Figura 4.8: Continuación del diagrama de la Figura 4.7 para el caso en que se realice una mantención al sistema $RBAC$.

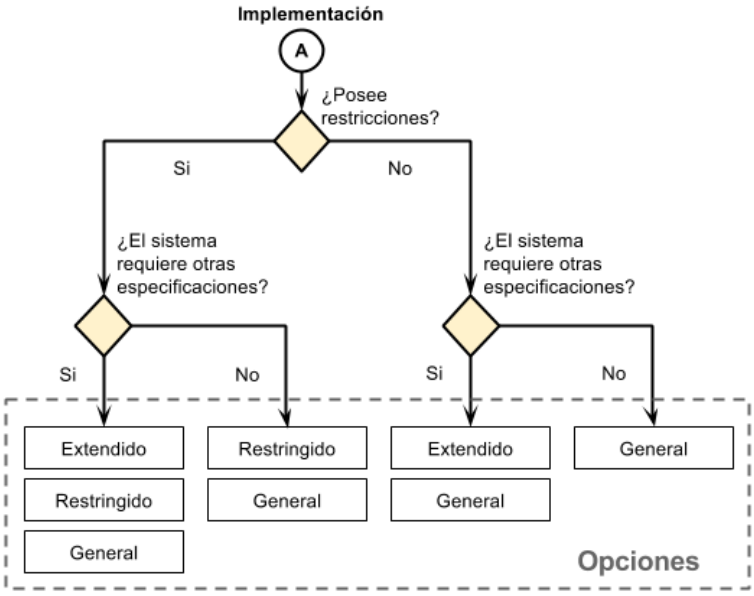


Figura 4.9: Continuación del diagrama de la Figura 4.7 para el caso en que se implemente el sistema *RBAC*.

Normalmente, el sistema *RBAC* que se desea mantener o implementar, puede pertenecer a una o más categorías de la clasificación “Naturaleza de la configuración RBAC"del árbol mencionado. Para estos casos, se debe seleccionar aquellas estrategias que se adecuen a los requerimientos propios de la organización. Dependiendo de los recursos disponibles para la mantención o la implementación del *RBAC*, es posible aplicar más de un enfoque para la resolución del problema, de modo que la solución propuesta puede pertenecer a un enfoque en particular o a una mezcla entre dos o más resultados.

Si se desea implementar un sistema *RBAC*, desde su inicio, entonces se recomienda utilizar alguna estrategia basada en la categoría “*General*”, que minimice el tamaño de los componentes del *RBAC*. Esto es aplicable, siempre y cuando, el enfoque propuesto satisfaga completamente los requerimientos organizacionales. Por otro lado, si la organización ya ha implementado un sistema *RBAC*, entonces es más conveniente modificar el conjunto de los roles existentes que crear uno nuevo, por lo que se recomienda utilizar cualquier estrategia basada en la categoría “*Perturbado*”. Al igual que en el caso anterior, esto es aplicable siempre y cuando el enfoque propuesto satisfaga completamente los requerimientos organizacionales.

Tanto para el caso de la implementación del *RBAC* como para la mantención de éste, si la organización requiere que el sistema satisfaga ciertas restricciones, entonces se podría utilizar cualquier estrategia basada en la categoría "*Restringido*". Del mismo modo, si la organización desea modificar los componentes del modelo básico del *RBAC* [23] para mejorar la administración del sistema, entonces se podría utilizar cualquier estrategia basada en la categoría "*Extendido*".

En base a esto, lo más importante es identificar correctamente los componentes *RBAC* que pueden ser minimizadas, según el enfoque y los requerimientos de la organización. Por ejemplo, la categoría "*Perturbado*" no considera el ruido (errores en el conjunto de datos) dentro de su clasificación, pero sí posee la jerarquía de roles. De esta forma, si se sospecha que que el conjunto de datos de entrada ($DUPA_{in}$) presenta varias asignaciones erróneas, entonces se puede utilizar cualquier estrategia que gestione el ruido y esté alineado con los requerimientos de la organización. Del mismo modo, se puede considerar el caso de la relación jerárquica entre los roles, la cual está presente en las categorías "*General*" y "Perturbado".

Finalmente, si la organización está interesada en incorporar información del negocio a los procesos de la ingeniería de roles, se recomienda utilizar cualquier enfoque que obtenga roles que semánticamente posean significado o que esté basado en alguna metodología híbrida de la ingeniería de roles.

Sin adelantar mayores detalles sobre el sistema en que se está estudiando, éste no presenta elementos externos al modelo clásico de *RBAC*. Además, las restricciones que posee no forman parte del problema, por lo cual pueden ser omitidas. Dado que el sistema se está implementando, la única opción resultante del criterio de selección pertenece a la categoría "General". De esta categoría, se evaluaron los enfoques que satisfacen los requerimientos de la organización, estos son: *RMP-Básico*, *Edge-RMP*, *WSCO* y *RHMP*. De dichos enfoques se seleccionó *WSCO* por su capacidad de adaptarse a diversos objetivos, y por utilizar la métrica wsc. Así, se logró alinear los vectores de peso con los respectivos enfoques.

4.5. Algoritmos

En esta sección se analiza los algoritmos usados para implementar los diversos enfoques del presente estudio. La elección de éstos se basa en los resultados del estudio realizado por Molloy et. al. [18] el 2009, en donde utilizan los conjuntos de datos reales para evaluar el desempeño y la calidad de las soluciones de 9 algoritmos clásicos de la minería de roles. En dicho estudio, se utiliza el wsc para medir la calidad de cada solución propuesta en base al vector de peso W especificado, y de forma complementaria, se usa $\frac{|DUPA_{out}|}{|DUPA_{in}|}$ para medir el porcentaje de asignaciones directas entre usuarios y permisos que no fue cubierto por la solución propuesta. Tal como se estudió en la sección "métricas de evaluación".

	PC	DM	HM	ORCA	HPr	HPe	GO	CM
University	31	42	21	56	17	18	18	32
Healthcare	24	27	17	46	14	15	16	31
Domino	64	31	31	231	20	27	20	62
EMEA	242	37	115	3046	34	176	34	674
APJ	779	655	549	1164	455	477	475	764
Firewall 1	248	219	111	709	65	78	71	278
Firewall 2	14	13	11	590	10	12	10	21
Americas	1778	829	428	1587	206	317	225	2672
Average	397	231	160	928	102	140	108	566
Ranking	6.12	5.06	3.94	7.75	1.19	3.19	2.00	6.75

(a) $W = \langle 1, 0, 0, 0, \infty \rangle$

	PC	DM	HM	ORCA	HPr	HPe	GO	CM
University	619	683	604	1773	894	643	615	620
Healthcare	148	325	146	223	298	210	136	164
Domino	501	553	376	659	723	389	476	495
EMEA	5811	7105	4482	6915	7214	4508	4926	6364
APJ	4733	4207	3904	4608	4867	3951	3987	4985
Firewall 1	2258	4689	1456	22101	2932	1723	2554	2678
Firewall 2	992	998	959	30789	1562	1067	981	995
Americas	16647	18557	6756	43156	16376	8394	9721	29926
Average	3963	4639	2335	13778	4358	2610	2924	5778
Ranking	4.00	6.12	1.12	7.00	6.75	3.25	2.62	5.12

(b) $W = \langle 1, 1, 1, 1, 1 \rangle$

	PC	DM	HM	ORCA	HPr	HPe	GO	CM
University	595	638	588	1823	887	627	593	595
Healthcare	189	390	144	225	288	185	162	230
Domino	573	733	411	703	741	402	517	549
EMEA	9439	7264	4120	7468	7246	3930	8118	12025
APJ	5674	5349	3794	5152	4876	3910	13029	5971
Firewall 1	2558	4490	1411	13295	3037	1611	3172	1919
Firewall 2	986	1075	952	29232	1554	1053	1008	1000
Americas	17657	18783	6779	41264	16235	8143	10459	19276
Average	4708	4840	2274	12395	4358	2482	4632	5195
Ranking	4.56	6.12	1.25	6.50	5.50	2.62	4.25	5.19

(c) $W = \langle 0, 1, 1, 1, \infty \rangle$

	PC	DM	HM	ORCA	HPr	HPe	GO	CM
University	0.00	0.00	0.00	0.34	0.04	0.01	0.01	0.00
Healthcare	0.02	0.00	0.01	0.00	0.04	0.04	0.01	0.02
Domino	0.18	0.61	0.26	0.60	0.97	0.25	0.24	0.22
EMEA	0.75	0.97	0.34	0.94	1.00	0.40	0.33	0.86
APJ	0.36	0.36	0.32	0.45	0.41	0.31	0.33	0.66
Firewall 1	0.03	0.12	0.01	0.48	0.02	0.00	0.01	0.03
Firewall 2	0.00	0.00	0.00	0.77	0.00	0.00	0.00	0.00
Americas	0.08	0.12	0.00	0.36	0.00	0.00	0.01	0.22
Average	0.18	0.27	0.12	0.49	0.31	0.13	0.12	0.25
Ranking	4.25	4.88	2.88	6.62	6.00	3.38	3.31	4.69

(d) $\frac{|DUPA|}{|UP|}$, **con** $W = \langle 1, 1, 1, 1, 1 \rangle$

Figura 4.10: Compara los resultados obtenidos por diversos algoritmos de la minería de roles ante distintas métricas.

La Tabla 4.10 resume los resultados más destacados de dicho estudio. Cabe destacar que, en ese estudio, se omitió los resultados del algoritmo FM por el precario desempeño que éste obtuvo y por su insignificante aporte a la discusión. Cada experimento incluye un ranking promedio del desempeño de cada algoritmo. Para el caso en que dos o más algoritmos entreguen el mismo resultado, el ranking r de cada uno de estos se obtendrá mediante la Ecuación 4.4, donde n corresponde a la cantidad de elementos repetidos, y

k es la posición del ranking en disputa. Como ejemplo se considera el caso de estudio (a) de la Tabla 4.10 para el conjunto de datos "*Domino*", en el que los algoritmos HPr y GO comparten el primer lugar con $wsc = 20$, seguido por HPe con $wsc = 27$. Sin esta regla, HPr y GO comparten el primer lugar, y HPe recibe el tercero. Por otro lado, tras aplicar la Ecuación 4.4, HPr y GO comparten el ranking $r = \frac{1+2}{2} = 1{,}5$, mientras que, HPe queda en tercer lugar. La disminución de la distancia entre estos ranking permite evaluar objetivamente el desempeño de cada participante, evitando así, castigar injustamente a aquellos elementos posteriores a los repetidos. Complementando lo anterior, la Tabla 4.10 entrega el ranking de cada conjunto del experimento (a).

$$r = \frac{\sum_{i=k}^{n} x}{n} \tag{4.4}$$

Para una mayor comprensión, se resume los resultados de cada experimento en la Tabla 4.11. En esta tabla, en cada experimento, se destaca en azul los tres algoritmos que obtuvieron el mejor desempeño. En general, HM obtuvo el mejor desempeño de todos, con excepción en el caso (a), donde ni siquiera aparece en los primeros tres lugares del ranking. En contraste, está el caso del HPr el cual, en general, obtuvo un desempeño deplorable, con excepción del caso (a) donde ocupa el primer lugar. Por otra parte, GO tiende a estar ubicado en una mejor posición que HPe, con excepción del caso (c).

Según el criterio de selección de los enfoques del RMP, las variantes que mejor se adaptan a las necesidades de la organización es *WSCO*, y forma parte de los algoritmos HM, GO, HPe y HPr , lo que da la seguridad de utilizar los algoritmos más eficaces de la minería de roles para la resolución del presente estudio, según las necesidades de la organización.

A continuación, se analiza en detalle cada uno de los algoritmos seleccionados:

Tabla 4.10: Ranking del desempeño de cada algoritmo para ciertos conjuntos de datos reales, con $W = (1,0,0,0,\infty)$.

	PC	*DM*	*HM*	*ORCA*	*HPr*	*HPe*	*GO*	*CM*
University	5	7	4	8	1	2.5	2.5	6
Healthcare	5	6	4	8	1	2	3	7
Domino	7	4.5	4.5	8	1.5	3	1.5	6
EMEA	6	3	4	8	1.5	5	1.5	7
APJ	7	5	4	8	1	3	2	6
Firewall 1	6	5	4	8	1	3	2	7
Firewall 2	6	5	3	8	1.5	4	1.5	7
Americas	7	5	4	6	1	3	2	8
Promedio	6.13	5.06	3.94	7.75	1.19	3.19	2.00	6.75

Tabla 4.11: Ranking del desempeño de los algoritmos del RMP más relevantes.

Experimento	*PC*	*DM*	*HM*	*ORCA*	*HPr*	*HPe*	*GO*	*CM*
a) $W = (1,0,0,0,\infty)$	6.12	5.06	3.94	7.75	1.19	3.19	2.0	6.75
b) $W = (1,1,1,1,1)$	4.00	6.12	1.12	7.00	6.75	3.25	2.62	5.12
c) $W = (0,1,1,1,\infty)$	4.56	6.12	1.25	6.50	5.50	2.62	4.25	5.19
d) $\frac{\lvert DUPA_{out}\rvert}{\lvert DUPA_{in}\rvert}$ con $W = (1,1,1,1,1)$	4.25	4.88	2.88	6.62	6.00	3.38	3.31	4.69
Promedio	**4.73**	**5.55**	**2.30**	**6.97**	**4.86**	**3.11**	**3.05**	**5.44**

4.5.1. Optimización de Grafos (GO)

El algoritmo "*Graph Optimization*" (*GO*) [27], cuya traducción al castellano es "Optimización de Grafos", plasma el *RMP* como un problema de optimización de un grafo, en donde los vértices corresponden a los usuarios, roles y permisos existentes; y los arcos son las relaciones entre estos elementos, estableciendo de este modo, una relación de jerarquía entre los roles. Inicialmente, este algoritmo crea un rol por cada conjunto de permisos de cada usuario, tal como lo representa la Figura 4.11 (a). En cada iteración del algoritmo, se eliminará la redundancia entre dos roles, ya sea porque estos tienen asociado el mismo

conjunto de permisos (p. ej. Figura 4.11 (b)), o porque tienen un conjunto de permisos en común, o porque se pueden relacionar jerárquicamente. Esto, con el objetivo de minimizar su función objetivo $f(x) = |E| + |R|$, donde E corresponde al número de aristas. Originalmente, el algoritmo se centraba en minimizar la suma del número de roles y relaciones (arcos) presentes en el grafo, pero en la actualidad, éste ha sido adaptado para poder trabajar con los vectores de peso del $WSCO$ ($f(x) = wsc$).

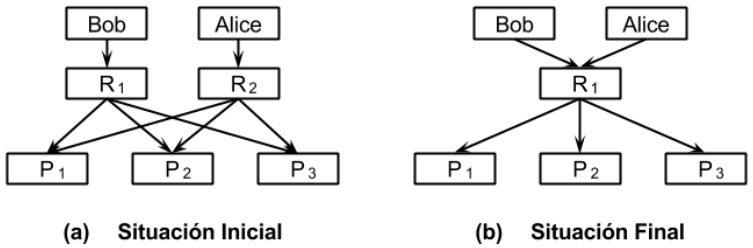


Figura 4.11: Situación inicial y final de un grafo creado por el algoritmo *GO*.

Con respecto a la complejidad del algoritmo, éste tiene una magnitud del orden $O((k+n)km)$, en donde n es el número de usuarios, m es el número de permisos y k es el número límite de iteraciones (condición de parada). Esto significa que, para cada nuevo rol, se necesita evaluar $O(n+k)$ pares de roles a un costo de $O(m)$ cada uno.

Finalmente, en la sección Anexo, Códigos, se describe este algoritmo por medio del pseudocódigo.

4.5.2. Minería Jerárquica (HM)

El algoritmo *Hierarchical Miner* (HM) [16], cuya traducción al castellano es "Minería Jerárquica", se basa en el Análisis Formal de Conceptos (FCA) para la resolución del RMP mediante la optimización del conjunto jerárquico de roles. En particular, este algoritmo requiere la comprensión de los siguientes términos:

Definición 4.5.1 (Contexto formal [16]) *Corresponde a la tupla* (G, M, I)*, en donde G y M son conjuntos, e* $I \subseteq G \times M$ *es la relación binaria entre dichos conjuntos. Se denota "objetos" (en alemán "Gegenstände") a los elementos de G (*$g \in G$*), "características" (en alemán*

"Merkmale") a los elementos de M ($m \in M$), e "incidencia" a la relación I (en alemán "Inzidenz"). De modo que, el objeto g se relaciona binariamente con el atributo m, denotado como gIm, si y sólo si, $(g, m) \in I$.

Definición 4.5.2 (Concepto del contexto [16]) *corresponde a la tupla (X, Y), en donde $X \subseteq G$ e $Y \subseteq M$, y satisface las siguientes propiedades:*

- $X = \{g \in G \mid (\forall m \in Y)\ gIm\}$, i.e. , X es el conjunto de todos los objetos que comparten todos los atributos en Y. Este conjunto es conocido como la "extensión" del concepto.
- $Y = \{m \in M \mid (\forall g \in X)\ gIm\}$, i.e. , Y es el conjunto de todos los atributos que comparten todos los objetos en X. Este conjunto es conocido como la "intención" del concepto.

Por lo tanto, cada extensión con su respectiva intención, conforman un concepto formal (X, Y). Cabe destacar que el concepto (X_1, Y_1) es un subconcepto de (X_2, Y_2), relación denotada como $(X_1, Y_1) \preceq (X_2, Y_2)$, si y sólo si, $X_1 \subseteq X_2$ o $Y_2 \subseteq Y_1$.

Definición 4.5.3 (Retículo (Lattice) [2]) *corresponde a un conjunto parcialmente ordenado, en donde cada par de elementos está conectado a un único elemento superior e inferior, tal como lo demuestra la Figura 4.12.*

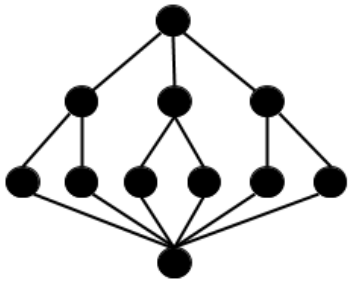


Figura 4.12: Ejemplo de un retículo de 4 niveles.

En el contexto de la minería de roles, la relación entre usuarios y permisos corresponde a un contexto formal, en donde G es el conjunto de todos los usuarios, y M es el conjunto

de todos los permisos. De esta forma, $(g, m) \in I$, si y sólo si, el usuario g tiene asignado el permiso m. Por lo tanto, se define el concepto del contexto formal como el par (U, P), en donde U contiene todos los usuarios que tienen todos los permisos en P, y P corresponde a todos los permisos que tienen todos los usuarios en U, de modo que los conjuntos U y P sean máximos.

Este algoritmo se basa en la teoría de los conjuntos y retículos (lattice), para la estructuración de los roles. Inicialmente, se crea un retículo basado en el contexto formal, en donde cada vértice corresponde un concepto (U, P) de dicho contexto. En este retículo, cada concepto o vértice corresponde a un rol, el cual hereda todos los permisos y usuarios de los subconceptos y sobre-conceptos asociados según el orden descrito (los permisos se heredan de abajo hacia arriba, mientras que los usuarios se heredan en la dirección inversa). Por lo tanto, el retículo corresponde a un conjunto jerárquico de roles, en donde cada rol está conformado por algún concepto. A modo de ejemplo, la Figura 4.13 (a) ilustra el retículo inicial, generado por el algoritmo HM, al considerar como parámetro de entrada, el conjunto $DUPA_{in}$ representado por Tabla 4.12. Continuando con la explica-

Tabla 4.12: $DUPA_{in}$ utilizado en el ejemplo del algoritmo HM.

	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}
U_0	1	0	1	0	0	1	0	0	0	0	1	1
U_1	1	0	1	0	0	1	0	0	0	0	1	1
U_2	1	1	1	0	0	1	0	0	0	0	1	1
U_3	1	1	0	1	1	0	0	0	0	0	1	1
U_4	1	1	0	1	1	0	1	0	0	1	1	1
U_5	1	1	0	1	1	0	1	0	0	1	1	1
U_6	1	0	0	1	0	0	1	0	0	1	1	1
U_7	1	0	0	1	0	0	1	1	1	1	1	1
U_8	1	0	0	1	0	0	1	1	1	0	1	1
U_9	1	0	0	1	0	0	1	1	1	0	1	1

ción del algoritmo, tras obtener el retículo inicial, éste es reducido a su mínima expresión,

eliminando toda redundancia creada por la herencia de usuarios y permisos. En base a este retículo reducido, se itera en búsqueda de algún cambio que disminuya el valor actual de su función objetivo $f(x) = wsc$, ya sea por eliminación de algún rol (proceso conocido como "poda") o por la reestructuración de estos. Cabe destacar que, podar un rol implica que los usuarios de éste bajen de jerarquía mientras que los permisos suben de jerarquía. Finalmente, este algoritmo se detiene cuando no se puede realizar más cambios que mejoren la solución actual. Para el caso del ejemplo descrito, el retículo reducido es representado por la Figura 4.13 (b), mientras que el resultado final (post poda y reestructuración) se ilustra en la Figura 4.14 (a). La solución óptima a este problema se encuentra representada por la Figura 4.14 (b).

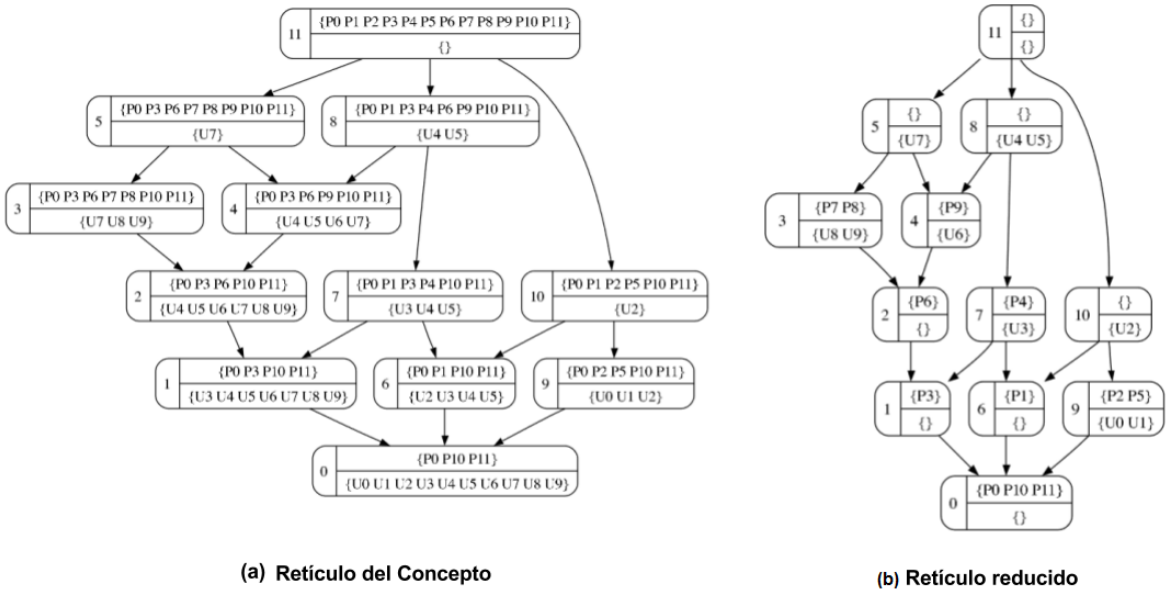


Figura 4.13: Ejemplo de un retículo (a) y su respectiva versión podada (b).

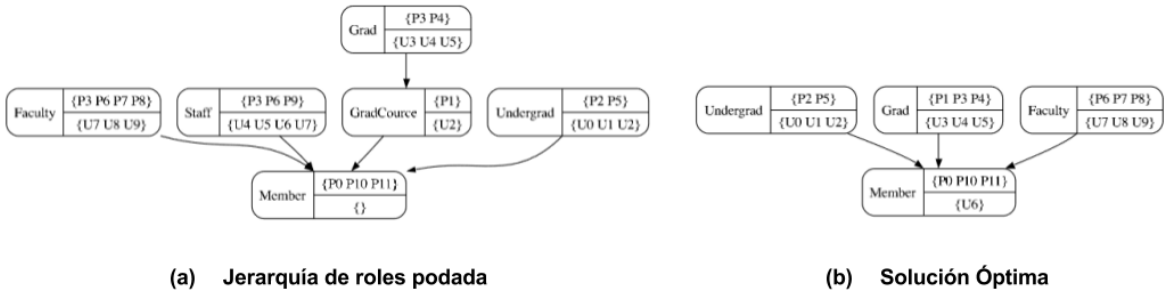


Figura 4.14: Ejemplo de la jerarquía de roles de un retículo podado (a) y su respectiva solución óptima (b).

Actualmente, los procesos de estructuración y poda están basados en un algoritmo voraz (también conocido como "*greedy*"), el cual establece que, en cada iteración, se debe buscar la opción que genere el mayor beneficio. La desventaja de esta técnica es que no asegura la solución óptima, pero permite entregar una solución cercana al óptimo rápidamente.

Con respecto a la complejidad del algoritmo, ésta es determinada por dos partes: (1) por el tamaño de la versión reducida del concepto del retículo (lattice); y (2) por la cantidad de elementos podados. La versión reducida del concepto del retículo se procesa en tiempo lineal con respecto al tamaño del retículo. Por otra parte, el tiempo de ejecución del proceso de poda, depende directamente del tamaño del retículo en su forma reducida. Cabe destacar que cada iteración involucrada con el proceso de poda, requiere una cantidad constante de tiempo para la reestructuración de sus elementos, con excepción de la jerarquía de roles, que depende directamente del tamaño de dicho retículo.

4.5.3. Minimización de Roles de HP (HPr)

El algoritmo *"HP Role Minimization"* (HPr) [5], cuya traducción al castellano es "Minimización de Roles de HP", fue propuesto por el área de investigación de la compañía *Hewlett Packard* (*HP Labs*). Este algoritmo resuelve el problema *RMP-Básico*, entregando como solución, un estado $RBAC$ plano (sin jerarquía) con la mínima cantidad de roles posibles. Formalmente, esto equivale a minimizar la función objetivo $f(x) =\mid R \mid$. Como se estudió

en capítulos anteriores, el *RMP-Básico* corresponde a un caso especial del $WSCO$, por lo cual es posible definir la función objetivo como $f(x) = wsc$.

Para la comprensión de este algoritmo, se requiere introducir algunos conceptos relacionados con los conjuntos formados entre los usuarios y permisos. Se denota $P(u)$ al conjunto de permisos asociados al usuario u. Similarmente, el conjunto $U(p)$ está compuesto por los usuarios que tienen asignado el permiso p. Por otra parte, el conjunto $U(u)$ está conformado por los usuarios que poseen los mismos permiso que el usuario u. Por último, el conjunto $P(p)$ está formado por los permisos asignados a los usuarios de $U(p)$.

Este algoritmo, en cada iteración, selecciona un usuario u o permiso p, y busca un par $\{U(p), P(p)\}$ o $\{U(u), P(u)\}$ para formar un rol candidato. Todas las asignaciones asociadas a $U(p)$, $P(p)$, $U(u)$ y $P(u)$, según sea el caso, son removidas, para que no sean consideradas en las siguientes iteraciones. En [5] se analizó varios criterios de selección de los usuarios o permisos, en donde el principio que entregó los mejores resultados consiste en escoger el usuario o permiso con la menor cantidad de permisos sin cobertura.

Con respecto a la complejidad del HPr, éste tiene una magnitud del orden $O(mn)$, en donde m es el número de permisos y n es la cantidad de usuarios. En particular, cada iteración, en el peor de los casos, requiere revisar todos los usuarios y permisos existentes para la selección del usuario u o permiso p, esto es $O(m+n)$. El número de iteraciones depende del número de roles, y este, a su vez, depende de la cantidad de usuarios y permisos, por lo cual se requiere, a lo más, $min\{m,n\}$ iteraciones.

4.5.4. Minimización de Bordes de HP (HPe)

El algoritmo *HP Edge Minimization* (HPe) [5], cuya traducción al castellano es "Minimización de Bordes de HP", también fue propuesto por *HP Labs*, y se inspiró en el algoritmo *GO*. Este algoritmo resuelve el problema *Edge-RMP*, entregando como solución, un estado $RBAC$ plano (sin jerarquía) con la mínima cantidad de aristas o asignaciones posibles. Formalmente, esto equivale a minimizar la función objetivo $f(x) =\mid UA \mid + \mid PA \mid$. Como se estudió en capítulos anteriores, el *Edge-RMP* corresponde a un caso especial del $WSCO$, razón por la cual la función objetivo es reformulada como $f(x) = wsc$.

HPe está dividido en dos procesos: (1) la generación del conjunto de roles candidatos; y (2) la optimización de dicho conjunto. Para el primer caso, se utiliza el algoritmo *HPr* para encontrar el conjuntos de roles candidatos. Para el segundo proceso, se utiliza un algoritmo voraz para la mejora continua de la función de evaluación (minimización de la función objetivo), mediante la transformación de los roles existentes. En cada iteración del algoritmo, se eliminará la redundancia entre dos roles, ya sea porque estos tienen asociado el mismo conjunto de permisos o porque tienen un conjunto de permisos en común. Esta situación es similar al proceso de transformación del algoritmo *GO*, con la salvedad de que *GO* considera la jerarquía de roles en su evaluación. Finalmente, el algoritmo se detendrá cuando no se pueda realizar mejora alguna.

Con respecto a la complejidad de *HPe*, éste tiene una magnitud del orden $O(k^2m)$, en donde k es el número límite de iteraciones y m es la cantidad de permisos.

Tabla 4.13: Resumen de las características de los algoritmos del RMP seleccionados.

Algoritmo	**Enfoque**	**Complejidad**
GO	*RHMP*	$O((k+n)km)$
HM	*RHMP*	$O(mn)$
HPr	*RMP-Básico*	$O(mn)$
HPe	*Edge-RMP*	$O(k^2m)$

4.6. Conjuntos de datos reales

Actualmente, en la literatura se utilizan 9 conjuntos de datos reales para evaluar el desempeño de los diversos algoritmos enfocados en la minería de roles [6]. Cabe destacar que estos conjuntos de datos se encontraban almacenados en los servidores del área de investigación de la compañía *Hewlett Packard* (*HP Labs*), y eran de libre uso pero, a inicios del 2017, éstos han sido removidos de los servidores de *HP*. A pesar de ello, en la literatura se encuentra varios respaldos de estos conjuntos. Es más, en la gran mayoría de las herramientas de la minería de roles, incorporan estos conjuntos de datos poder probar diversas técnicas o enfoques de la ingeniería de roles en base a datos reales.

La Tabla 4.14 describe las principales características de estos conjuntos de datos, entre ellas se destaca: el número de usuarios y permisos ($|U|$ y $|P|$ respectivamente); el número de permisos asignados a los respectivos usuarios ($|DUPA_{in}|$); el mínimo y el máximo número de permisos asignados a un usuario ; y la densidad de $DUPA_{in}$, la cual corresponde al porcentaje de asignaciones de permisos a los respectivos usuarios, es decir, $densidad(UPA) = |UPA| \div (|U| \times |P|) \times 100$.

Tabla 4.14: Características de los conjuntos de datos reales más relevantes del RMP.

Conjunto de datos	$\|U\|$	$\|P\|$	$\|DUPA_{in}\|$	Mín # permisos por rol	Máx # permisos por rol	$densidad(DUPA_{in})$
Healthcare	46	46	1.486	7	46	70,23 %
Domino	79	231	730	1	209	4,00 %
Emea	35	3.046	7.220	9	554	6,77 %
Firewall 1	365	709	31.951	1	617	12,35 %
Firewall 2	325	590	36.428	6	590	19,00 %
Apj	2.044	1.164	6.841	1	58	0,29 %
Americas large	3.485	10.127	185.294	1	733	0,53 %
Americas small	3.447	1.587	105.205	1	310	1,91 %
Customer	10.021	277	45.427	1	25	1,64 %

Con respecto al origen de estos conjuntos de datos, estos fueron obtenidos de los sistemas de HP [6] y del área de administración de veteranos de las fuerzas armadas de los Estados Unidos. Los conjuntos de "*Americas small*" y "*large*", corresponden a los usuarios que se identificaron en la red de HP, y que fueron captados por el cortafuegos de Cisco. Similarmente, se obtuvieron los conjuntos *"Apj"* y *"Emea"*. *"Domino"* corresponden a los datos de los usuarios y sus respectivos perfiles de acceso a un servidor de *Lotus Domino*. La información de *"Customer"* está basada en los registros de control de acceso del departamento de tecnologías de la información del área de clientes de HP. Los conjuntos de datos *"Firewall 1"* y *"Firewall 2"* fueron obtenidos como resultado de un algoritmo que analizó los cortafuegos *Check Points* de HP. Por último, el conjunto de datos *Healthcare* se obtuvo del área de administración de veteranos ya mencionada, e involucra los permisos de atención médica de sus proveedores.

En la literatura no se ha especificado la cantidad mínima de permisos y usuarios que debe tener un conjunto de datos para poder aplicar en éste la ingeniería de roles. Sin embargo, de la Tabla 4.14 se desprende que lo más importante del conjunto de datos es la cantidad de tipos de permisos, ya que la asignación de estos es lo que da complejidad al problema.

4.7. RMiner como Herramienta de Evaluación

En esta sección se analiza la herramienta que se usó en el presente estudio para implementar los algoritmos expuestos en el capítulo anterior. Dicho instrumento corresponde al *RMiner* [12], software de libre uso basado *WEKA* [20] (conjunto de algoritmos de aprendizaje automático enfocada en el proceso de la minería de datos), y escrito en el lenguaje de programación Java, que facilita y agiliza el proceso de minería de roles al ofrecer distintos mecanismos para el estudio y la comparación de algunos algoritmos clásicos de la minería de roles. Actualmente, esta herramienta puede ser descargada en [21] y considera 13 algoritmos clásicos de la minería de roles, donde 3 de estos resuelven el *RMP-Básico*, 6 incorporan la jerarquía de roles en su conjunto de salida, 1 considera las perturbaciones de un conjunto de roles desplegados, 1 evalúa el peso de cada permiso y 2 admiten restricciones en el sistema. Cabe destacar, que todos los algoritmos seleccionados para el presente estudio son considerados en esta herramienta.

Antes de analizar la herramienta como tal, es necesario comprender el formato del conjunto de datos requerido por éste. El *RMiner* requiere transformar el conjunto de datos de entrada a un formato propio de la herramienta, denominado $ARAF$. Este configuración de archivo está basado en el formato $ARFF$ de $WEKA$ [20] (herramienta enfocada en la minería de datos), e incorpora el concepto de "*Latitud*" para identificar los conjuntos que participan en cada una de las relaciones de asignación. Por lo tanto, la latitud puede representar al conjunto de usuarios, roles o permisos. El formato $ARAF$ puede ser dividido en dos partes: (1) la representación matricial de la relación de asignación entre dos conjuntos; y (2) la identificación de dichos conjuntos con sus respectivos atributos. Para una mayor comprensión, se considera, como ejemplo, transformar el conjunto $DUPA_{in}$, representado por la Tabla 4.4, dando como resultado la Figura 4.15. De la Figura 4.15 (a) se desprende que "*@assignment upa*" indica el tipo de relación de asignación (notar que no utilizan el término "$DUPA$"); "*@latitude user*" es el conjunto de usuarios; "*@latitude permission*" es el conjunto de permisos; y "*@matrix*" es la relación de asignación entre estos dos conjuntos. Cabe destacar, que esta herramienta permite transformar al formato $ARAF$ las relaciones de asignación UA, PA, RH y $DUPA$, . Además de los datos vinculados con la relación asignación entre dos conjuntos, $ARAF$ permite especificar los atributos

de cada conjunto, p. ej, asociar una edad y un área de la empresa (departamento) a cada usuario, tal como lo demuestra la Figura 4.15 (b). De la Figura 4.15 (b) se desprende que "*@attribute age numeric [18, 65]*" indica que "*age*" es un atributo "*numérico*" y que puede tomar valores desde 18 hasta 65. Un caso similar ocurre con "*department*", en donde se establece que éste puede ser "IF", "RH" o "AG".

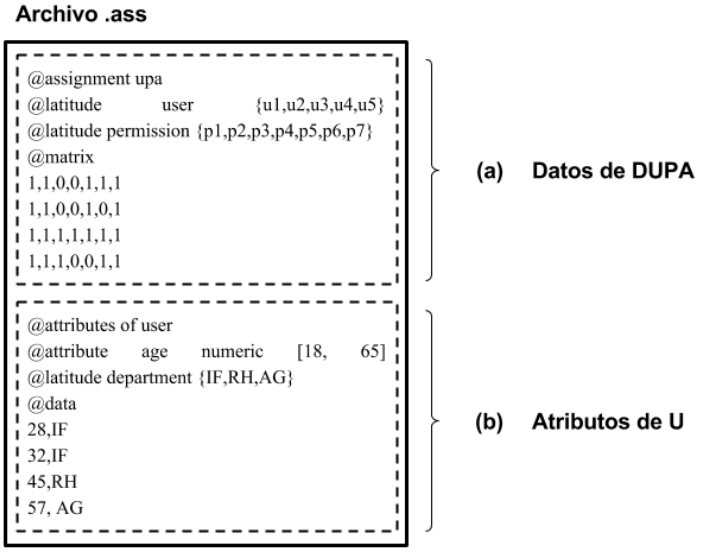


Figura 4.15: Ejemplo de $DUPA$ transformada al formato $ARAF$.

Cada relación de asignación debe ser guardada de forma separada, en la carpeta "*datasets*", y siguiendo el siguiente formato de archivo: "*.ass*" para $DUPA_{in}$; "*.ur*" para UA; y "*.rp*" para PA. En esa misma carpeta se encuentra ejemplos de los conjuntos de datos más utilizados en la literatura, así también como un ejemplo de los archivos en formato "*.ur*" y "*.rp*".

RMiner está dividido en cuatro secciones o módulos, las cuales son: "*pre-procesamiento*", "*minería de roles*", "*asignación*" y "*visualización*". Cada una de estos módulos corresponden a un proceso relacionado con la minería de roles. El módulo de "*pre-procesamiento*", representado por la Figura 4.16, consiste en la identificación y la corrección manual del ruido presente en el conjunto de datos utilizado, en otras palabras, *RMiner* da la opción de editar manualmente la información de $DUPA_{in}$. Cabe destacar que, esta herramienta no considera el proceso de actualización de grandes volúmenes de datos, siendo esto, su principal desventaja. Además, de cargar el conjunto de datos, este módulo entrega datos

estadísticos de dicho conjunto (número de usuarios, permisos, etc), y analiza los atributos de los conjuntos de usuarios y de permisos. En este módulo también se da la opción de generar un conjunto de datos artificial en base a alguno de los algoritmos implementados.

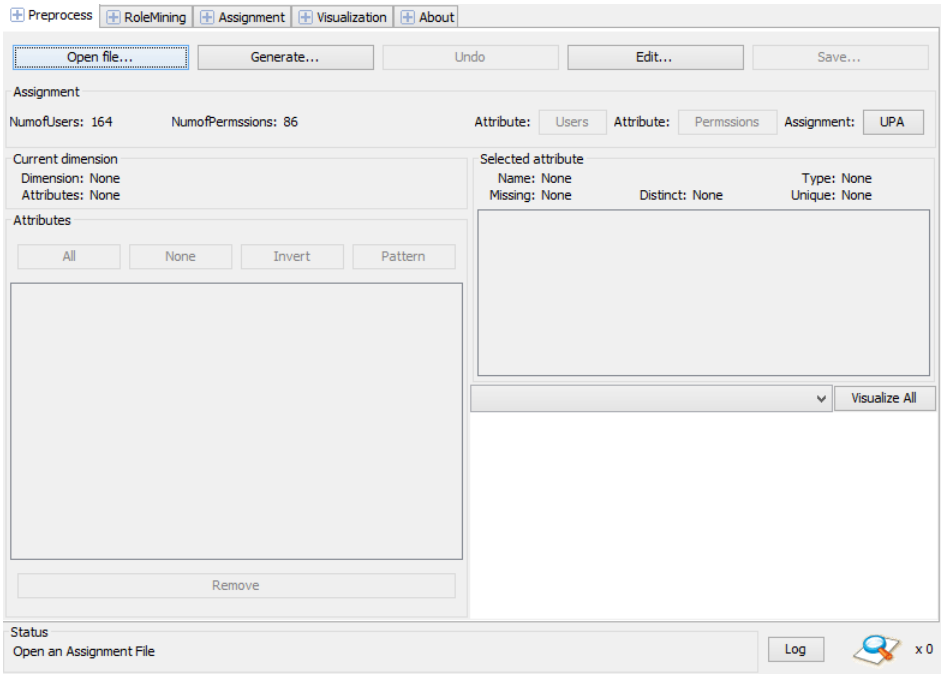


Figura 4.16: Ejemplo del interfaz del módulo "*pre-procesamiento*" de *RMiner*.

El módulo de "*minería de roles*", abarca el proceso de selección de algún algoritmo del RMP presente en la plataforma, y su posterior ejecución, tal como lo demuestra la Figura 4.17. En base al vector de peso W y al algoritmo seleccionado, se obtendrá el conjunto de datos de salida, evaluando posteriormente la calidad de dicha solución (wsc). Complementando lo anterior, RMiner registra el tiempo de ejecución y el espacio de memoria utilizado por el algoritmo. En caso de alguna falla, el botón "*Log*" permite revisar en detalle los problemas encontrados. Cabe destacar que *RMiner* da la opción de incorporar más algoritmos de minería de roles.

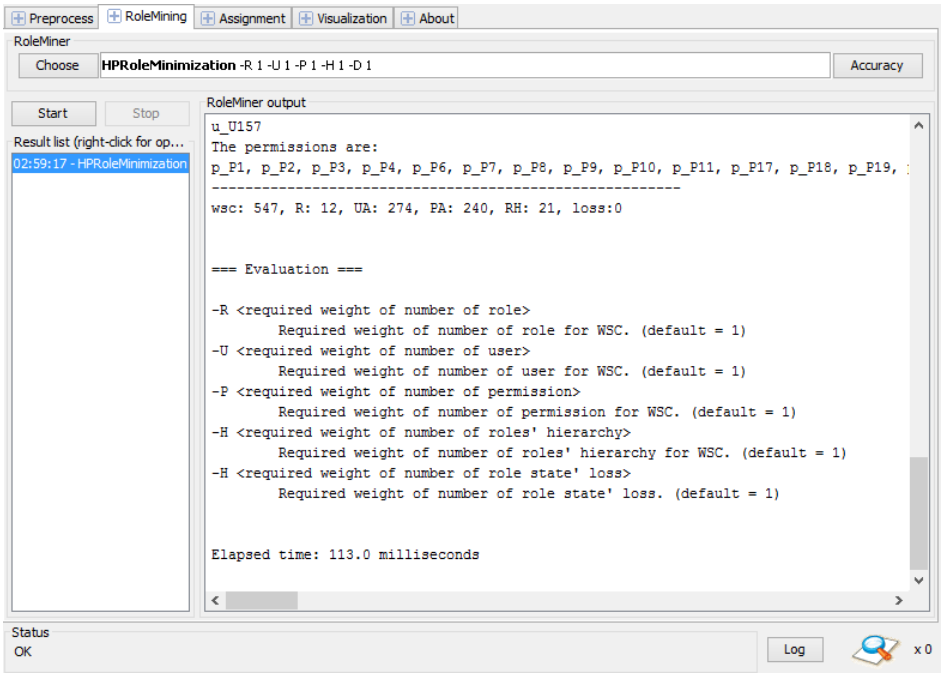


Figura 4.17: Ejemplo del interfaz del módulo "*minería de roles*" de *RMiner*.

El módulo de "*asignación*" consiste en la asignación de los roles generados en la etapa previa a los respectivos usuarios, tal como lo ilustra la Figura 4.18. Esta plataforma considera un algoritmo voraz para la la asignación de los roles a los respectivos usuario, dando la opción de agregar nuevos algoritmos para dicha tarea. Esta sección tiene sentido al considerar que, en algunos casos, el conjunto de roles generado, no satisface los requerimientos de la organización, y es necesario la edición de éste (agregar, editar o eliminar roles). *RMiner* divide este proceso en dos partes: (1) es la jerarquía del conjunto de roles, representado por la Figura 4.19; y (2) es la relación entre los roles y sus respectivos permisos, cuyos elementos marcados en verde corresponde a los datos que han sido heredados, y por lo tanto, no pueden ser editados, tal como lo ilustra la Figura 4.20.

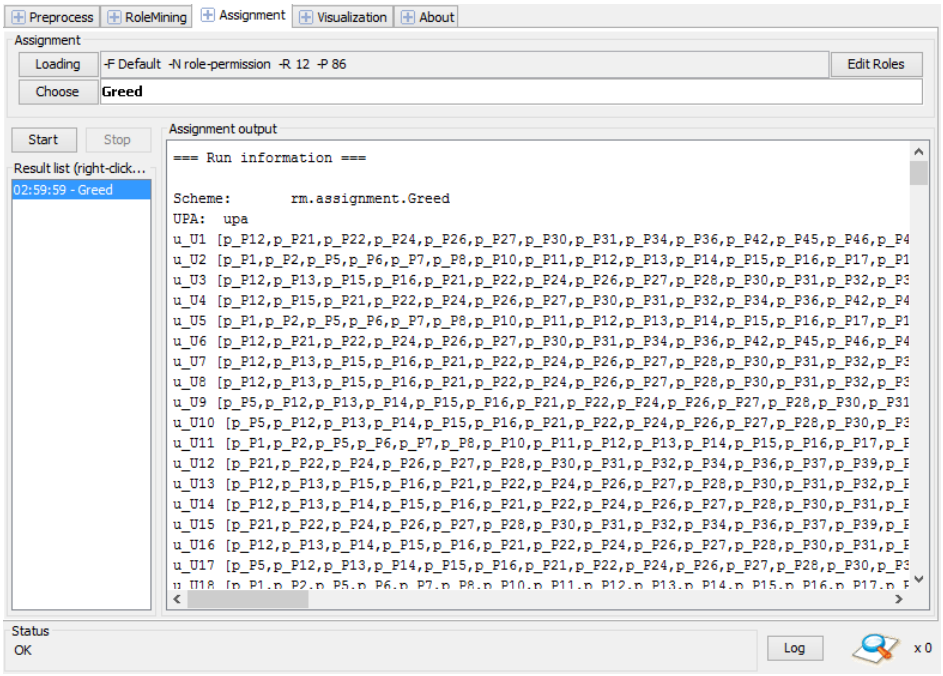


Figura 4.18: Ejemplo del interfaz del módulo "*asignación*" de *RMiner*.

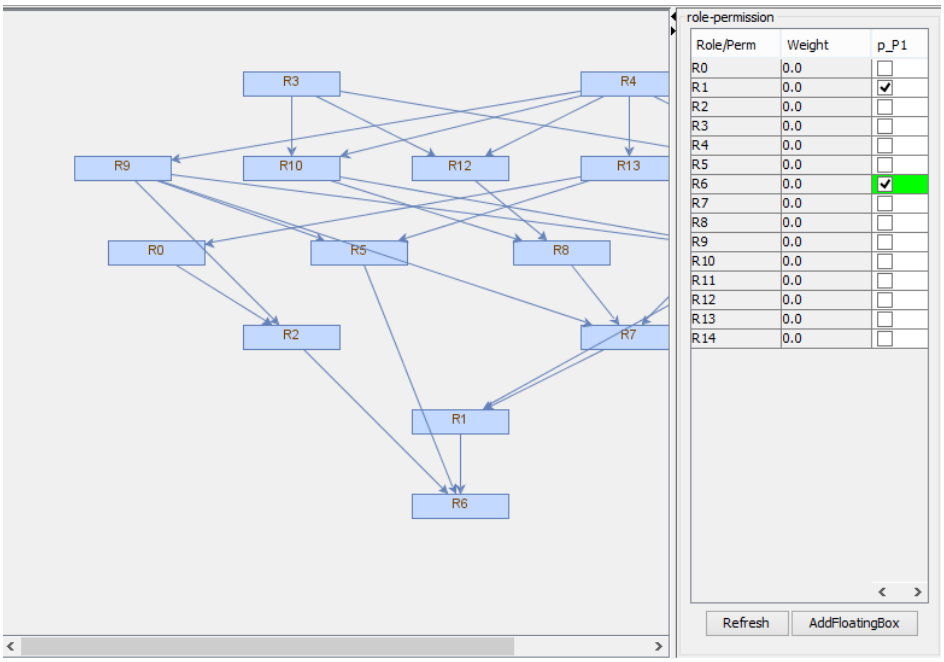


Figura 4.19: Ejemplo del interfaz del módulo "*asignación*" de *RMiner* al editar la jerarquía de roles.

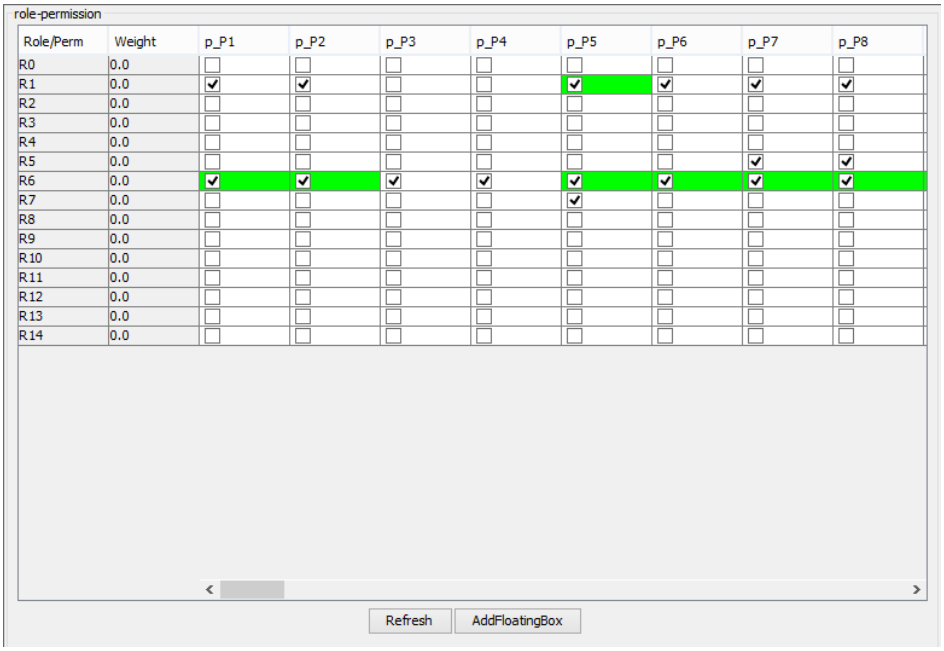


Figura 4.20: Ejemplo del interfaz del módulo "*asignación*" de *RMiner* al editar el conjunto de roles.

El módulo de "*visualización*", considera un subconjunto de herramientas para el análisis gráfico de los datos de entradas, mediante un diagrama de dispersión e histograma *RMiner* de los conjuntos U y P. Como ejemplo, se considera la Figura 4.21, la cual representa gráficamente el grado de similitud entre los permisos. En esta figura, tanto el eje vertical como el horizontal corresponden a los permisos, y la intensidad del negro indica que tan probable es que un rol generado con ellos sea representativo en el conjunto de datos.

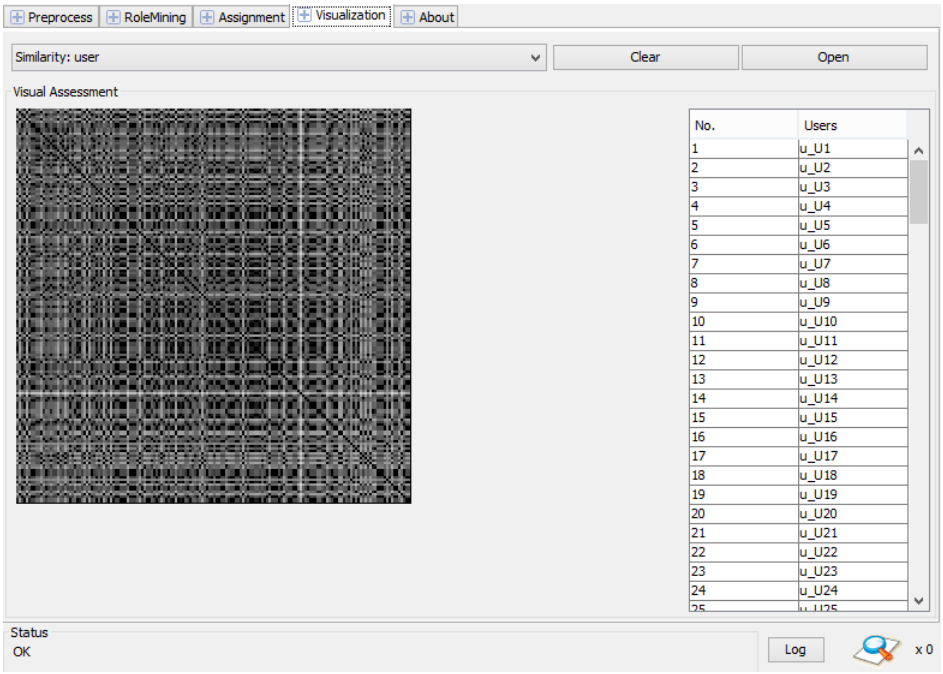


Figura 4.21: Ejemplo del interfaz del módulo "*visualización*" de *RMiner*.

Finalmente, hay que destacar que *RMiner* cuenta con un manual de uso [22], de libre acceso, que explica en detalle cada uno de los procesos de la minería de roles.

Capítulo 5

Conjunto de datos

En este capítulo, se introduce una breve reseña del sistema $RBAC$, que es estudiado en el presente texto, enfatizando la descripción en las características propias de este sistema. Como se mencionó en la introducción de este documento, el estudio corresponde a la utilización de la minería de roles para la implementación del sistema de control de acceso de la plataforma utilizada en la asignatura *Análisis y Diseño de Software* de la *Universidad Técnica Federico Santa María, para los Campus San Joaquín* y *Casa Central*. Las características generales de este sistema están resumidas en la Tabla 5.1, en donde $|U|$, $|P|$ y $|R|$ corresponden al número de usuarios, permisos y roles respectivamente; $|DUPA_{in}|$ es el número de asignaciones directas, entre los conjuntos usuarios y permisos; "*Mín y máx # perms por usuario*" son el mínimo y el máximo número de permisos asignado a algún usuario; "*# restricciones (U) , (P), (R)*" son la cantidad de restricciones asociadas a los usuarios, permisos y roles respectivamente; "*Mín y máx # roles por usuario*" son el mínimo y el máximo número de roles asignado a algún usuario; "*Máx # subroles*" es el máximo número de subroles relacionados con algún rol; y *densidad(x)* equivale al porcentaje de asignaciones utilizadas por x, es decir, $densidad(x) = |x| \div (tamaño(x)) \times 100$.

Tabla 5.1: Resumen de las características del sistema estudiado.

Característica	Valor
$\|U\|$	164
$\|P\|$	66
Máx # roles por usuario	1
Mín # roles por usuario	1
Máx # permisos por rol	66
Mín # permisos por rol	12
$\|R\|$	5
$tamaño(PA)$	330
$\|PA\|$	144
$densidad(PA)$	43,64 %
$tamaño(UA)$	820
$\|UA\|$	164
$densidad(UA)$	20,00 %
$tamaño(DUPA_{out})$	10824
$\|DUPA_{out}\|$	0
$densidad(DUPA_{out})$	0,00 %
$\|RH\|$	4
$\|tr(RH)\|$	4
Máx # subroles	1
# restricciones (P)	15
# restricciones (U)	2
# restricciones (R)	1

Esta plataforma se implementó en base "*Redmine*" [11], herramienta web de gestión de proyectos, enfocada en el desarrollo de aplicaciones, la cual incorpora las siguientes funcionalidades: sistema de seguimiento de avances y de incidentes; calendario de actividades; diagrama de Gantt, para la representación visual y cronológica de los acontecimientos; foro de discusión, *wiki* [1], notificaciones por correo electrónico, sistema de control de versiones mediante el uso de repositorio, entre otras funcionalidades. *Redmine* es multiplataforma, multilingüe y de libre uso (posee la *Licencia Pública General de GNU*). Además de esto, el sitio oficial de *Redmine* [11] cuenta con documentación, guías, y foros de discusión y de soporte. Esta herramienta, puede ser descargada en [11], donde han creado numerosos guías de instalación para cada sistema operativo. Por último, cabe destacar que *Redmine* utiliza el entorno de trabajo (*framework*) "*Ruby on Rails*", el cual está escrito en el lenguaje de programación "*Ruby*".

Para efectos del presente texto, se utiliza el término ADS para referirse al sistema basado en *Redmine* que es utilizado en la asignatura *Análisis y Diseño de Software*. A su vez, se usa $RBAC_{ADS}$ para describir el sistema de control de acceso de ADS.

En la asignatura *Análisis y Diseño de Software*, cada campus tiene asignado un proyecto, y los alumnos de éstos, deben formar equipos de trabajo para desarrollar algún subproyecto. Cada subproyecto está enfocado en alguna área de desarrollo de *software* y tiene sus propios requerimientos y plazos de entrega. Cada entrega es evaluada y revisada por los respectivos profesores y ayudantes, de modo de asegurar que al final del semestre académico, los proyectos llevados a cabo satisfacen las expectativas de los profesores y clientes participantes. Cabe destacar que ADS permite distinguir los documentos propios del proyecto de los que están relacionados con el ramo propiamente tal.

Actualmente, los roles y permisos usados en $RBAC_{ADS}$ son los que vienen por defecto en la herramienta *Redmine*. Por el bien de la brevedad, se decidió mover esta información al "*Anexo*", en donde la Tabla B.1 describe los permisos usados en $RBAC_{ADS}$, mientras que la Tabla C.1 representa, de forma abreviada, la relación PA.

La Tabla B.1 agrupa los permisos según su funcionalidad para administrar los módulos

[1] página web en donde los usuarios modifican, de forma colaborativa y directa, la estructura y el contenido de éste

básicos de *Redmine*, abarcando así, la gestión de proyectos, foros, documentos, archivos, temas, noticias, repositorios, tiempos invertidos y páginas wiki, entre otros elementos. De esta tabla, se desprende que el sistema $RBAC_{ADS}$ posee 66 tipos de permisos distintos, existiendo, en la mayoría de estos, cierta abstracción de los permisos clásicos de acceso (ver, insertar, editar, eliminar), como es el caso del permiso para "*importar*". Por otra parte, existen permisos que corresponden a una agrupación de otros permisos, con la finalidad de reducir el tamaño y la complejidad del sistema, p. ej., el permiso de "*administración*" de algún recurso. Si bien, los permisos están vinculados a algún objeto o entidad, éstos son asociados y manipulados internamente por *Redmine*, asegurando así, que los usuarios no puedan manipular ni acceder a la información que no les corresponda. Por ejemplo, los usuarios pueden estar vinculados a uno o más proyectos de la plataforma pero, en cada proyecto, sólo los usuarios que estén asociados a éste podrán manipular o modificar la información, según los roles que estos usuarios posean.

Por el bien de la brevedad, no es posible incluir en el presente texto, el detalle de los usuarios y la relación UA del sistema $RBAC_{ADS}$. Aún así, en [4] es posible descargar toda la información relevante en este estudio, la cual abarca las relaciones UA, PA y $DUPA_{in}$, junto con el conjunto de restricciones del sistema $RBAC_{ADS}$ desplegado.

En cuanto a las restricciones asociadas a los permisos del sistema $RBAC_{ADS}$, se logró identificar tres tipos de restricciones: *prerrequisitos*, *excluyentes* y de *cardinalidad*. Como se analizó en la sección "*Ventajas del RBAC*", los *prerrequisitos* establecen que un permiso no puede asignarse a un rol a menos que éste tenga asignado previamente otros permisos (o roles) en particular, mientras que los *excluyentes*, que especifica que un dos o más permisos (o roles) no pueden estar asignado a la vez. Por último, la *cardinalidad* limita la cantidad de usuarios, permisos o roles. En la Tabla D.1 se resume las restricciones asociadas a los permisos del sistema $RBAC_{ADS}$, destacando, según sea el caso, el tipo de restricción asociado. Cabe destacar que esta tabla fue desplazada al *Anexo* por el mismo motivo que se trasladó las tablas anteriores.

De las Tablas C.1 y D.1, se desprende que "*Manager*" es el único rol que transgrede alguna de las restricciones identificadas. En particular, se infringe la restricción mutuamente excluyente entre p_{58} y p_{60}. Esta restricción se incluyó para evitar la redundancia que se

generaba en la definición de un rol, cuando éste consideraba los permisos de "*eliminar páginas wiki*" y "*Administrar páginas wiki*" (crear o eliminar páginas wiki). Para solucionar esto, simplemente se debe eliminar el permiso p_{60} (eliminar páginas wiki) del rol indicado.

Tabla 5.2: Resumen de las características de los roles desplegados.

Característica	Rol desplegado				
	Reporter	Developer	Non member	Anonymous	Manager
# usuarios asociados	0	0	1	1	162
% usuarios asociados	0,00 %	0,00 %	0,61 %	0,61 %	98,78 %
# permisos asociados	19	30	17	12	66
% permisos asociados	28,79 %	45,45 %	25,76 %	18,18 %	100,00 %

La Tabla 5.2 resume las principales características de los roles desplegados en el sistema $RBAC_{ADS}$. El rol "*Anónimo*", cuya traducción al inglés es "*Anonymous*", tiene asociado 12 tipos de permisos (18,18 %) y 1 usuario (0,61 %). Los permisos asociados a éste, corresponden a la visualización de la información ("*ver*"). En el otro extremo, se ubica el rol "*Administrador*", cuya traducción original al inglés es "*Manager*", quién tiene asociado los 66 permisos del sistema (100,00 %) y 162 usuarios (98,78 %). Entre estos dos roles, se encuentra el rol "*Visitante*" o "*No Miembro*", cuya traducción al inglés es "*No Member*", el cual posee 17 tipos de permisos (25,76 %) y tiene asociado un único usuario (0,61 %). Los permisos asociados a éste, corresponden a la visualización de la información ("*ver*") y en casos particulares, puede añadir información a la plataforma ("*agregar*"). Por otra parte, el rol "*Reportero*", cuya traducción al inglés es "*Reporter*", posee 19 tipos de permisos (28,79 %), y no tiene asociado usuario alguno. Este rol es similar al rol "*Visitante*", con la salvedad que éste puede editar sus propios mensajes ("*edit*") y registrar su tiempo invertido en algún proyecto. Por último, el rol "*Desarrollador*", cuya traducción original al inglés es "*Developer*", posee 30 tipos de permisos (45,45 %), y al igual que el "reporter", no tiene asociado usuario alguno. En general, este rol tiene limitado los elementos que puede agregar, editar o eliminar. Cabe destacar que *Redmine* no considera la jerarquía de roles dentro de su modelo, pero fácilmente se puede visualizar que los roles mencionados

siguen una jerarquía consecutiva, tal como lo demuestra la Figura 5.1.

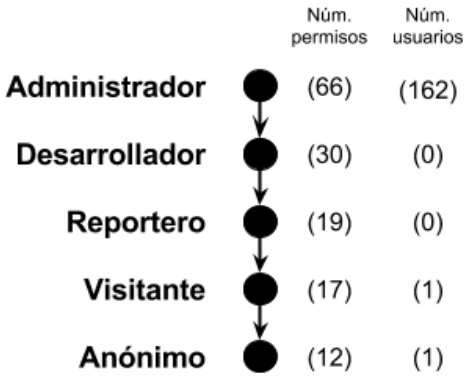


Figura 5.1: Jerarquía de los roles desplegados del sistema $RBAC_{ADS}$.

Con respecto a la relación UA, hay que destacar que el 98,78 % de los usuarios del sistema $RBAC_{ADS}$ tienen asociado únicamente el rol *Administrador*. Esta situación es lo que se debe evitar a todo costa en los sistema $RBAC$, ya que contradice las ventajas de dicho modelo. Además, la plataforma de ADS queda vulnerable e insegura, lo que en un futuro podría causar serios problemas entre los distintos participantes de este sistema. Debido a esto, y al hecho que los roles no están contextualizados en base a la asignatura ADS, es que se decidió re-implementar el sistema $RBAC_{ADS}$. Para ello, fue necesario investigar y re-diseñar la matriz de relación $DUPA_{in}$, procurando que ésta satisfaga todas las restricciones de los permisos, lo que implicó desagrupar algunos permisos de acceso relacionados con la capacidad de "*administrar*" algún elemento, aumentando así la cantidad de permisos del sistema. Cabe destacar que dicha relación, puede ser revisada en [4].

Tras analizar la plataforma ADS, se identificaron los siguientes tipos de usuarios: "*Administrador*", "*Profesor*", "*Alumno*", "*Ayudante*", "*Visitante*", y "*Cliente*". El "*Administrador*", es quien se encarga de administrar y mantener la plataforma, por lo que tiene asociado permisos para agregar, editar y eliminar ciertos elementos, dependiendo de las circunstancias. El "*Profesor*" es el responsable de gestionar los proyectos de la asignatura ADS, por lo que es capaz de ver, agregar, editar y eliminar los elementos básicos de la plataforma que estén enfocados en la gestión misma del proyecto, p. ej., podrá agregar, editar o eliminar temas, pero no podrá editar los mensajes, ni manipular las horas invertidas en

el proyecto de los demás usuarios. El "*Alumno*" está a cargo de las labores operacionales y técnicas del proyecto, por lo que sus permisos se enfocarán en ver, añadir y editar dicho contenido, p. ej., podrá indicar las horas invertidas en su respectivo proyecto, pero no podrá manipular los temas. El "*Ayudante*", como su nombre lo indica, corresponde a la persona que ayuda al profesor a evaluar y gestionar los respectivos proyectos, por lo cual, tiene atribuciones menores que el profesor, pero mayores que los alumnos. Un "*Visitante*" es cualquier persona que recurra a la plataforma con la intención de revisar el contenido de ésta, por lo cual, tiene asociado únicamente permisos para ver los elementos más básicos de la plataforma. Por último, el "*Cliente*" es la persona que ideó el proyecto y desea llevarlo a cabo, por lo que se encarga de especificar los requerimientos del proyecto, responder dudas, revisar los avances y sugerir cambios. Es por esto que posee atribuciones para ver y añadir el contenido básico de la plataforma. Si bien, se ha mencionado a grandes rasgos en qué consiste cada tipo de usuario, existen variaciones internas en estos, p. ej., algunos "*Alumnos*" pueden agregar temas, mientras que otros no. Es en estos casos, donde el método *Bottom-Top* de la ingeniería de roles resulta ser útil, ya que agiliza y simplifica el proceso de búsqueda del conjunto de roles óptimo.

Desde este punto en adelante, se prescinde del conjunto de datos representado por la Tabla 5.1, y se considera únicamente el conjunto de datos corregido. La Tabla 5.3 resume las características generales de éste.

Tabla 5.3: Resumen de las características del sistema estudiado, tras su corrección.

$\|U\|$	$\|P\|$	*tamaño*$(DUPA_{in})$	$\|DUPA_{in}\|$	*densidad*$(DUPA_{in})$
164	86	14104	7209	51,11 %

Capítulo 6

Resultados

En esta sección, se describe los resultados obtenidos por la herramienta *RMiner*, tras ejecutar los algoritmos *GO*, HM, HPr y HPe, en base al conjunto de datos corregido de la plataforma *ADS*, y considerando los criterios de evaluación seleccionados. Debido a las dimensiones de las relaciones resultantes, éstas no fueron incorporadas al presente texto, salvo el caso de la relación RH, la cual fue resumida gráficamente antes de ser anexada a este documento. Para cada uno de los algoritmos y criterios de evaluación considerados, se resume las características más importantes de cada solución propuesta, destacando según corresponda, la jerarquía del conjunto de roles RH y los aspectos más importantes de PA.

Del módulo de "*visualización*" de *RMiner* se extrajo las Figuras 6.1 y 6.2. La Figura 6.1 corresponde a un diagrama de similitud entre los permisos existentes, en donde, tanto el eje vertical como el horizontal, corresponde a los permisos de acceso del sistema, y la intensidad de dicho color indica que tan estrecha es la relación entre estos dos permisos. De forma similar, la Figura 6.2 representa gráficamente la similitud entre los usuarios existentes

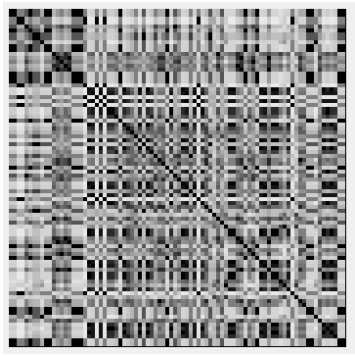


Figura 6.1: Diagrama de similitud de los permisos del sistema $RBAC_{ADS}$.

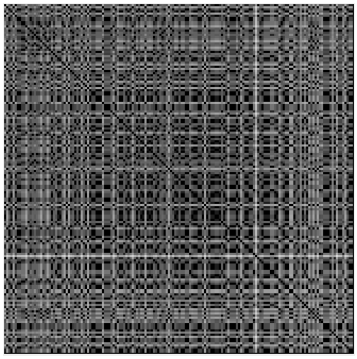


Figura 6.2: Diagrama de similitud de los usuarios del sistema $RBAC_{ADS}$.

6.1. Optimización de Grafos (GO)

La solución del algoritmo *GO* es idéntica para los 3 tipos de vectores de peso W utilizados en este estudio, y corresponde a un grafo optimizado con 263 vértices y 691 aristas. Cada vértice corresponde a un usuario, un rol o un permiso, por lo que los vértices descritos se dividen en: 164 usuarios, 13 roles y 86 permisos. Por otra parte, los arcos o aristas indican que existe una asociación entre dos elementos, que pueden ser: "*rol-usuario*", "*rol-permiso*" o "*rol-rol*" (herencia). Así, se obtuvo: 164 relaciones entre usuarios y roles; 515 entre roles y permisos; y 12 entre roles. Esta forma de estructurar la solución no es considerada en el módulo gráfico de *RMiner*, pero es posible reconstruirla a partir de la información entregada como texto. Es por esto, y por la gran cantidad de aristas y vértices, que no es posible ilustrar en el presente texto la totalidad de la estructura de la solución

propuesta por el algoritmo *GO*. La Figura 6.3 representa parte del grafo generado por el algoritmo *GO*, en el cual se puede identificar visualmente los tres tipos de relaciones descritas.

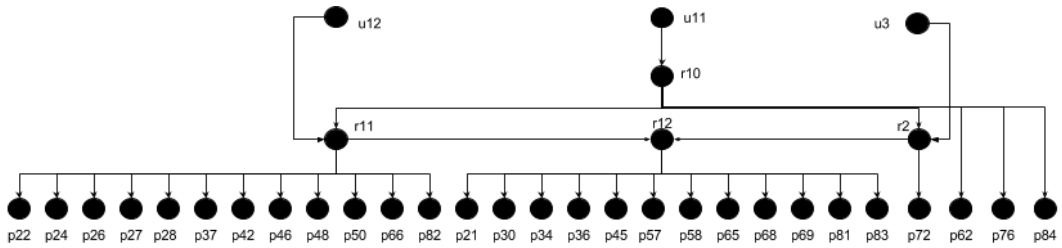


Figura 6.3: Parte del grafo generado por el algoritmo *GO*.

RMiner tampoco considera el diagrama de jerarquía de roles en su módulo gráfico, pero es posible reconstruirlo en base a la información entregada por éste. La Figura 6.4 representa la jerarquía entre los roles propuestos por el algoritmo *GO*, tras eliminar los roles y relaciones redundantes. De dicha figura se desprende que existen 12 roles y 11 asignaciones directas entre éstos.

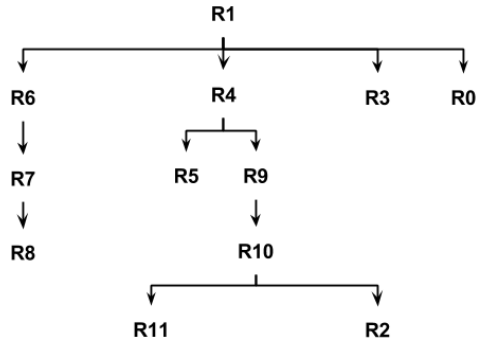


Figura 6.4: Jerarquía de roles generada por el algoritmo *GO*.

La Tabla 6.1 resume las características más importantes de la solución propuesta por este algoritmo. En esta tabla se utiliza la $densidad(x)$ para medir el porcentaje de asignaciones utilizado por el elemento x, es decir, $densidad(x) = |x| \ / \ tamaño(x) \cdot 100$. Por último, $|RH|/(|R|-1)$ compara la cantidad de roles y de relaciones. Cabe destacar que $|RH|/(|R|-1) \geq 0$, de modo que $|RH|/(|R|-1) = 0$ ocurre para el caso en que la configuración $RBAC$ sea plana. Por otra parte, $|RH|/(|R|-1) = 1$ corresponde a un árbol jerárquico

con la mínima cantidad de relaciones posibles, mientras que $|RH|/(|R|-1) > 1$ corresponde a un grafo donde existen nodos con más de un nodo superior. Para el caso en que $0 < |RH|/(|R|-1) < 1$, existen roles que no se conectan con otros. Complementando esto, la Tabla 6.2 describe las características de los roles generados por el algoritmo *GO*.

Por último, la Tabla 6.3 resume los valores del wsc para cada uno de los vectores de peso utilizados.

Tabla 6.1: Características de la solución obtenida por *GO*.

Característica	Valor
Máx # roles por usuario	1
Mín # roles por usuario	1
Máx # permisos por rol	86
Mín # permisos por rol	13
$\|R\|$	12
$tamaño(PA)$	1032
$\|PA\|$	503
$densidad(PA)$	48,74 %
$tamaño(UA)$	1968
$\|UA\|$	164
$densidad(UA)$	8,33 %
$tamaño(DUPA_{out})$	0
$\|DUPA_{out}\|$	0
$densidad(DUPA_{out})$	0,00 %
$\|DUPA_{out}\|$ / $\|DUPA_{in}\|$	0.00
$\|RH\|$	11
$\|tr(RH)\|$	11
$\|RH\|$ / $\|R-1\|$	1,00
Máx # subroles (herencia)	4

Tabla 6.2: Características de los roles generados por *GO*.

Roles	# permisos	% permisos	# usuarios	% usuarios
r_0	37	43,0,2 %	2	1,22 %
r_1	86	100,00 %	1	0,61 %
r_2	13	15,12 %	1	0,61 %
r_3	71	82,56 %	24	14,63 %
r_4	51	59,30 %	6	3,66 %
r_5	44	51,16 %	11	6,71 %
r_6	49	56,98 %	38	23,17 %
r_7	35	40,70 %	42	25,61 %
r_8	31	36,05 %	15	9,15 %
r_9	33	38,37 %	13	7,93 %
r_{10}	28	32,56 %	7	4,27 %
r_{11}	24	27,91 %	4	2,44 %

Tabla 6.3: wsc generado por *GO* para cada W.

	$W = (0,1,1,1,\infty)$	$W = (1,0,0,0,\infty)$	$W = (1,1,1,1,1)$
wsc	678	12	690

6.2. Minería Jerárquica (HM)

La solución del algoritmo *HM* es diferente para los 3 tipos de vectores de peso W utilizados en este estudio, y se estructura por medio de un retículo (*lattice*), donde cada vértice corresponde a un concepto (U,P) que representa un rol, el cual hereda todos los permisos (de arriba hacia abajo) y los usuarios (de abajo hacia arriba) de los demás conceptos que se encuentran vinculados a éste. Por otra parte, los arcos indican que existe una asociación entre dos conceptos, de modo de generar una relación de jerarquía entre los roles. Al igual que en el caso del algoritmo *GO*, *RMiner* no considera esta forma de estructurar la solución en su módulo gráfico, pero es posible reconstruir dicha estructura a partir de

la información entregada por éste. Es por esto, y por la gran cantidad de aristas y vértices, que no es posible ilustrar en el presente texto la totalidad de la estructura de la solución propuesta por el algoritmo HM. La Figura 6.5 representa parte del retículo generado por el algoritmo HM en base al vector de peso $W = (1,1,1,1,1)$. En esta figura, se visualiza los dos tipos de elementos que conforman un rol, destacando que, en algunos casos, los roles pueden poseer un conjunto vacío de usuarios o permisos, pero no ambos.

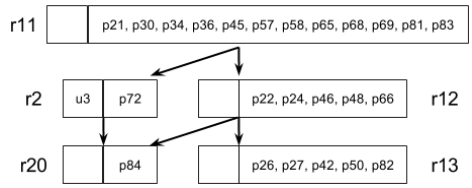


Figura 6.5: Parte del retículo generado por HM.

Las Figuras 6.6, 6.7 y 6.8 representan los retículos generados por el algoritmo HM, en base al CR y a los vectores de peso $W = (0,1,1,1,\infty)$, $W = (1,0,0,0,\infty)$ y $W = (1,1,1,1,1)$ respectivamente. Para el caso de la Figura 6.6, con $W = (0,1,1,1,\infty)$, el retículo posee 24 vértices (o roles), y 40 aristas (o relaciones entre roles), mientras que en la Figura 6.7, con $W = (1,0,0,0,\infty)$, el retículo posee 15 vértices y 26 aristas. Por último, el retículo representado por la 6.6, con $W = (1,1,1,1,1)$, posee 21 vértices y 36 aristas.

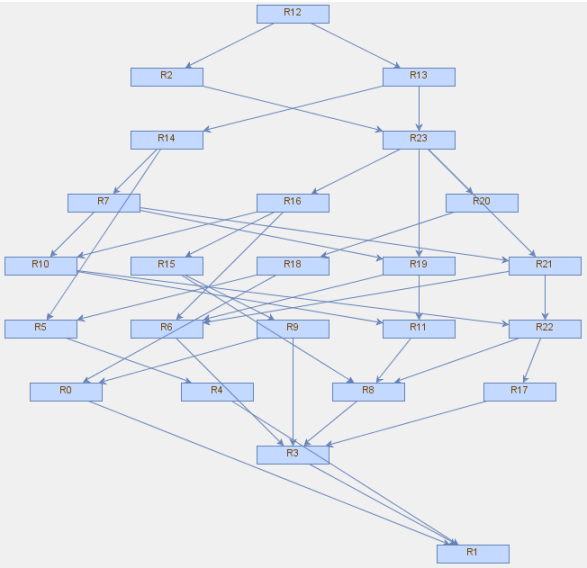


Figura 6.6: Jerarquía de roles generada por HM considerando $W = (0,1,1,1,\infty)$.

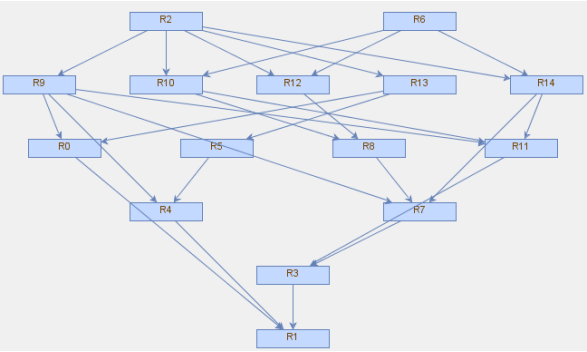


Figura 6.7: Jerarquía de roles generada por HM considerando $W = (1,0,0,0,\infty)$.

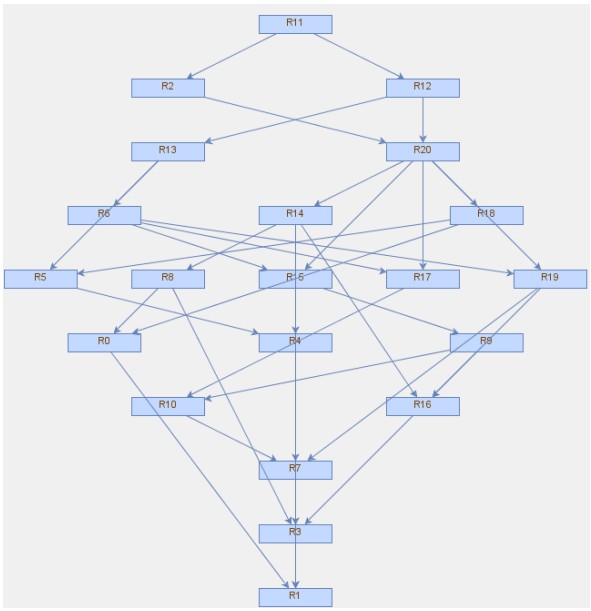


Figura 6.8: Jerarquía de roles generada por HM considerando $W = (1,1,1,1,1)$.

Similar al caso anterior, la Tabla 6.4 resume las características más importantes de la solución propuesta, para cada uno de los vectores de peso utilizado. Las Tablas 6.5 y 6.6 describen las características de los roles generados en base a los vectores de peso utilizado. En particular, la Tabla 6.5 se enfoca en los permisos, mientras que la 6.6 en los usuarios.

Tabla 6.4: Características de las soluciones obtenidas por HM para cada W.

Característica	$W = (0,1,1,1,\infty)$	$W = (1,0,0,0,\infty)$	$W = (1,1,1,1,1)$
Máx # roles por usuario	24	15	21
Mín # roles por usuario	2	1	2
Máx # permisos por rol	16	33	16
Mín # permisos por rol	1	5	1
$\|R\|$	24	15	21
$tamaño(PA)$	2064	1290	1806
$\|PA\|$	103	268	107
$densidad(PA)$	4,99 %	20,78 %	5,92 %
$tamaño(UA)$	3936	2460	3444
$\|UA\|$	2003	1091	1845
$densidad(UA)$	50,89 %	44,35 %	53,57 %
$tamaño(DUPA_{out})$	0	0	0
$\|DUPA_{out}\|$	0	0	0
$densidad(DUPA_{out})$	0,00 %	0,00 %	0,00 %
$\frac{\|DUPA_{out}\|}{\|DUPA_{in}\|}$	0.00	0.00	0.00
$\|RH\|$	40	26	36
$\|tr(RH)\|$	40	26	36
$\frac{\|RH\|}{\|R-1\|}$	1.74	1.86	1.80
Máx # hijos (herencia)	4	5	5
wsc	2146	15	288

Tabla 6.5: Características de los roles generados por HM por cada W, enfocadas a los permisos.

Rol	$W = (0,1,1,1,\infty)$		$W = (1,0,0,0,\infty)$		$W = (1,1,1,1,1)$	
	Núm. permisos	% permisos	Núm. permisos	% permisos	Núm. permisos	% permisos
r_0	2	2,33%	33	38,37%	4	4,65%
r_1	7	8,14%	7	8,14%	7	8,14%
r_2	1	1,16%	13	15,12%	1	1,16%
r_3	16	18,60%	20	23,26%	16	18,60%
r_4	4	4,65%	5	5,81%	5	5,81%
r_5	3	3,49%	27	31,40%	5	5,81%
r_6	3	3,49%	24	27,91%	2	2,33%
r_7	2	2,33%	14	16,28%	5	5,81%
r_8	2	2,33%	12	13,95%	12	13,95%
r_9	6	6,98%	20	23,26%	9	10,47%
r_{10}	11	12,79%	23	26,74%	4	4,65%
r_{11}	4	4,65%	12	13,95%	12	13,95%
r_{12}	12	13,95%	19	22,09%	5	5,81%
r_{13}	5	5,81%	20	23,26%	5	5,81%
r_{14}	5	5,81%	19	22,09%	2	2,33%
r_{15}	5	5,81%			5	5,81%
r_{16}	3	3,49%			3	3,49%
r_{17}	3	3,49%			1	1,16%
r_{18}	2	2,33%			2	2,33%
r_{19}	1	1,16%			1	1,16%
r_{20}	2	2,33%			1	1,16%
r_{21}	1	1,16%				
r_{22}	2	2,33%				
r_{23}	1	1,16%				

Tabla 6.6: Características de los roles generados por HM por cada W, enfocadas a los usuarios.

Rol	$W = (0,1,1,1,\infty)$		$W = (1,0,0,0,\infty)$		$W = (1,1,1,1,1)$	
	Núm. usuarios	% usuarios	Núm. usuarios	% usuarios	Núm. usuarios	% usuarios
r_0	3	1,83 %	3	1,83 %	3	1,83 %
r_1	1	0,61 %	1	0,61 %	1	0,61 %
r_2	160	97,56 %	160	97,56 %	160	97,56 %
r_3	25	15,24 %	25	15,24 %	25	15,24 %
r_4	81	49,39 %	81	49,39 %	81	49,39 %
r_5	96	58,54 %	96	58,54 %	96	58,54 %
r_6	82	50,00 %	104	63,41 %	104	63,41 %
r_7	104	63,41 %	31	18,90 %	31	18,90 %
r_8	31	18,90 %	42	25,61 %	27	16,46 %
r_9	27	16,46 %	126	76,83 %	80	48,78 %
r_{10}	80	48,78 %	96	58,54 %	42	25,61 %
r_{11}	42	25,61 %	63	38,41 %	164	100,00 %
r_{12}	164	100,00 %	62	37,80 %	163	99,39 %
r_{13}	163	99,39 %	115	70,12 %	161	98,17 %
r_{14}	161	98,17 %	89	54,27 %	126	76,83 %
r_{15}	33	20,12 %			93	56,71 %
r_{16}	95	57,93 %			63	38,41 %
r_{17}	63	38,41 %			62	37,80 %
r_{18}	98	59,76 %			115	70,12 %
r_{19}	62	37,80 %			89	54,27 %
r_{20}	115	70,12 %			159	96,95 %
r_{21}	89	54,27 %				
r_{22}	69	42,07 %				
r_{23}	159	96,95 %				

6.3. Minimización de Roles de HP (HPr)

La Figura 6.9 representa, en forma resumida, la solución del algoritmo HPr, que corresponde simplemente a un conjunto de roles plano (sin jerarquía), y es idéntica para los 3 tipos de vectores de peso W utilizados en este estudio. Complementando lo anterior, la Tabla 6.7 describe las características de los roles generados, mientras que la Tabla 6.8 resume las características más importantes de la solución propuesta. Por último, la Tabla 6.9 resume los valores del wsc para cada uno de los vectores de peso.

Figura 6.9: Jerarquía de roles generada por HPr.

Tabla 6.7: Características de los roles generados por HPr.

Roles	# permisos	% permisos	# usuarios	% usuarios
r_0	13	15,12 %	1	0,64 %
r_1	24	27,91 %	4	2,44 %
r_2	28	32,56 %	7	4,27 %
r_3	33	38,37 %	13	7,93 %
r_4	31	36,05 %	15	9,15 %
r_5	35	40,70 %	42	25,61 %
r_6	49	56,98 %	38	23,17 %
r_7	44	51,16 %	11	6,71 %
r_8	51	59,30 %	6	3,66 %
r_9	37	43,02 %	2	1,22 %
r_{10}	72	83,72 %	24	14,63 %
r_{11}	86	100,00 %	1	0,61 %

Tabla 6.8: Características de la solución obtenida por HPr.

Característica	Valor
Máx # roles por usuario	1
Mín # roles por usuario	1
Máx # permisos por rol	86
Mín # permisos por rol	13
$\|R\|$	12
$tamaño(PA)$	1032
$\|PA\|$	503
$densidad(PA)$	48,74 %
$tamaño(UA)$	1968
$\|UA\|$	164
$densidad(UA)$	8,33 %
$tamaño(DUPA_{out})$	0
$\|DUPA_{out}\|$	0
$densidad(DUPA_{out})$	0,00 %
$\frac{\|DUPA_{out}\|}{\|DUPA_{in}\|}$	0.00
$\|RH\|$	0
$\|tr(RH)\|$	0
$\frac{\|RH\|}{\|R\|-1}$	0.00
Máx # subroles (herencia)	0

Tabla 6.9: wsc generado por HPr para cada W.

	$W = (0,1,1,1,\infty)$	$W = (1,0,0,0,\infty)$	$W = (1,1,1,1,1)$
wsc	667	12	679

6.4. Minimización de Bordes de HP (HPe)

Similar al caso anterior, la Figura 6.10 representa, en forma resumida, la solución del algoritmo *HPe*, que corresponde simplemente a un conjunto de roles plano (sin jerarquía), y es idéntica para los 3 tipos de vectores de peso W utilizados en este estudio.

Figura 6.10: Jerarquía de roles generada por *HPe*.

La Tabla 6.11 resume las características más importantes de la solución propuesta, mientras que la Tabla 6.10 describe las características de los roles generados. Por último, la Tabla 6.12 resume los valores del wsc para cada uno de los vectores de peso.

Tabla 6.10: Características de los roles generados por *HPe*.

Roles	Núm. permisos	Núm. permisos	# usuarios	% usuarios
r_0	13	15,12%	10	6,10%
r_1	24	27,91%	22	13,41%
r_2	28	32,56%	14	8,54%
r_3	33	38,37%	44	26,83%
r_4	31	36,05%	16	9,76%
r_5	35	40,70%	43	26,78%
r_6	49	56,98%	39	23,78%
r_7	30	34,88%	18	10,98%
r_8	21	24,42%	7	4,27%
r_9	12	13,95%	3	1,83%
r_{10}	52	60,47%	25	15,24%
r_{11}	30	34,88%	1	0,61%
r_{12}	11	12,79%	3	1,833%
r_{13}	9	10,47%	6	3,66%
r_{14}	5	5,81%	2	1,22%

Tabla 6.11: Características de la solución obtenida por HPe.

Característica	Valor
Máx # roles por usuario	13
Mín # roles por usuario	1
Máx # permisos por rol	52
Mín # permisos por rol	5
$\|R\|$	15
$tamaño(PA)$	1290
$\|PA\|$	383
$densidad(PA)$	29,69 %
$tamaño(UA)$	2460
$\|UA\|$	253
$densidad(UA)$	10,28 %
$tamaño(DUPA_{out})$	0
$\|DUPA_{out}\|$	0
$densidad(DUPA_{out})$	0,00 %
$\frac{\|DUPA_{out}\|}{\|DUPA_{in}\|}$	0.00
$\|RH\|$	0
$\|tr(RH)\|$	0
$\frac{\|RH\|}{\|R\|-1}$	0,00
Máx # subroles (herencia)	0

Tabla 6.12: wsc generado por HPe para cada W.

	$W = (0,1,1,1,\infty)$	$W = (1,0,0,0,\infty)$	$W = (1,1,1,1,1)$
wsc	636	15	651

Capítulo 7

Análisis

En este capítulo, se analiza los resultados presentados en la sección anterior. Se comienza examinando al conjunto de datos como tal y las características de éste, análisis que es complementado por los diagramas de simetría de los usuarios y de los permisos. Posteriormente, se compara cuantitativamente los resultados obtenidos por los algoritmos estudiados, esto implica confrontar las características de los sistemas propuestos. Además, se compara el desempeño de cada algoritmo en base al vector de peso utilizado, tal como se estudió al comienzo de la sección *Algoritmos*. También, se contrasta los resultados obtenidos en el presente texto con otros estudios de similares características. Todo esto, con la finalidad de encontrar el conjunto de roles que mejor se adapte a las necesidades de la organización. Finalmente, se procede a denotar cada rol mediante algún nombre, evaluando el nivel de artificialidad de cada uno de éstos.

Con respecto al conjunto de datos, éste posee más usuarios que permisos, pero es la cantidad de permisos lo que le da complejidad al problema, ya que los usuarios se pueden reducir en base a las distintas combinaciones de permisos sin alterar el conjunto de roles resultante. Del diagrama de simetría de los usuarios, representado por la Figura 6.2, se desprende que cada elemento en negro indica que los dos usuarios involucrados poseen los mismos permisos, en contraste, los elementos de color blanco indican que los dos usuarios no poseen permisos en común. Así, mientras más intenso sea el gris, más

semejantes son esos usuarios. A partir de este diagrama, y considerando el tono de cada elemento, es posible determinar el mínimo número de roles del sistema. Para ello, se debe agrupar los usuarios que tengan las mismas características (destacados en negro), de modo que la cantidad de estos grupos determina el mínimo número de roles. Así, cada rol contiene el conjunto de usuarios y permisos asociados a algún grupo. Además, se confirma la existencia de una relación de jerarquía entre los usuarios (tonalidad oscura cercana al negro en cada uno de los usuarios), pero no se especifica el sentido de dicha dependencia o dominancia. Lo que si se puede deducir, es que mientras más intenso sea el elemento oscuro, la variación entre los conjuntos de permisos será menor. Tras agrupar los usuarios, se obtuvo como resultado la Figura 7.1, de la cual se desprende que la mínima cantidad de roles del sistema es 12 (los únicos elementos de color negro están en la diagonal). Cabe destacar que esta solución podría ser considerada por alguno de los algoritmos estudiados.

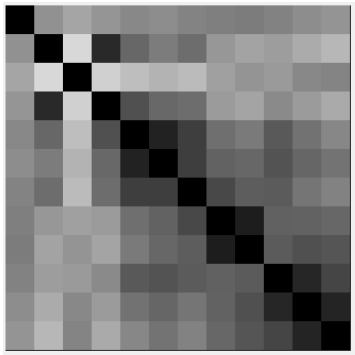


Figura 7.1: Diagrama de similitud de los usuarios del sistema $RBAC_{ADS}$, tras agrupar los usuarios.

Del diagrama de simetría de los permisos, representado por la Figura 6.1, se desprende que la tonalidad de cada elemento indica que tan frecuente aparecen a la vez los permisos involucrados, de modo que, el color negro señala que éstos aparecen en la totalidad de los casos, mientras que el color blanco señala lo contrario. Así, mientras más intenso sea el gris, más relacionados están estos permisos. Este diagrama es de gran utilidad para buscar el conjunto de roles, donde se extrae un rol a partir de los elementos en negro presentes en la respectiva fila o columna, teniendo la precaución de no considerar

los elementos simétricos y la diagonal de la matriz. El proceso de extraer los roles de este diagrama requiere una inversión considerable de tiempo, y no asegura que los roles encontrados satisfagan las necesidades de la empresa. Por esta razón, se decidió omitir dicho análisis.

Las Tablas 7.1, 7.3 y 7.2 resumen las principales características de las soluciones obtenidas para cada algoritmo en base a los vectores de peso utilizados. La Tabla 7.1 abrevia los resultados relacionados con la cantidad de asignaciones de las entidades más relevantes del sistema. Complementando lo anterior, la Tabla 7.3 especifica la densidad de cada una de estas entidades. Por último, la Tabla 7.2 resume las características restantes, centradas en la maximización o minimización de algún valor.

Tabla 7.1: Tamaño de las entidades más relevantes de las soluciones propuestas.

Algoritmo	$\|R\|$	$\|PA\|$	$\|UA\|$	$\|RH\|$	$\|tr(RH)\|$	$\|DUPA_{out}\|$
GO	12	503	164	11	11	0
HM con $W = (0,1,1,1,\infty)$	24	103	2003	40	40	0
HM con $W = (1,0,0,0,\infty)$	15	268	1091	26	26	0
HM con $W = (1,1,1,1,1)$	21	107	1845	36	36	0
HPe	15	383	253	0	0	0
HPr	12	503	164	0	0	0
Máx.	24	503	2003	40	40	0
Mín.	12	103	164	0	0	0

Tabla 7.2: Valor máximo y/o mínimo de las características de las soluciones propuestas.

Algoritmo	Máx. # roles por usuario	Mín. # roles por usuario	Máx. # permisos por rol	Mín. # permisos por rol	Máx. # subroles por rol
GO	1	1	86	13	4
HM con $W = (0,1,1,1,\infty)$	24	2	16	1	4
HM con $W = (1,0,0,0,\infty)$	15	1	33	5	5
HM con $W = (1,1,1,1,1)$	21	2	16	1	5
HPe	13	1	52	5	0
HPr	1	1	86	13	0
Mín.	1	1	16	1	0
Máx.	24	2	86	13	5

Tabla 7.3: Densidad de los elementos de las soluciones propuestas.

Algoritmo	$densidad(PA)$	$densidad(UA)$	$densidad(DUPA_{out})$
GO	48,74 %	8,33 %	0,00 %
HM con $W = (0,1,1,1,\infty)$	4,99 %	50,86 %	0,00 %
HM con $W = (1,0,0,0,\infty)$	20,78 %	44,35 %	0,00 %
HM con $W = (1,1,1,1,1)$	5,92 %	53,57 %	0,00 %
HPe	29,69 %	10,28 %	0,00 %
HPr	48,74 %	8,33 %	0,00 %
Mín.	4,99 %	8,33 %	0,00 %
Máx.	48,74 %	53,57 %	0,00 %

De la Tabla 7.1 se desprende que todos los algoritmos aplicados lograron resolver el problema propuesto sin la necesidad de recurrir a la asignación directa de permisos a los respectivos usuarios, ya que $|DUPA_{out}| = 0$ para todos los casos. Al considerar la magnitud de $|UA|$ y $|PA|$, se desprende que HM redujo $|UA|$, pero a costa de un incremento significativo de $|PA|$, lo que da indicios del pobre desempeño que tuvo HM con el presente conjunto de datos. Por otra parte, HPe logró un balance entre estas dos entidades, mientras que GO y HPr se enfocaron en reducir $|UA|$. Todo esto, puede ser corroborado al analizar la Tabla 7.3. Desde el punto de vista del $RBAC$, se debe priorizar la reducción de $|UA|$, para así facilitar las labores de mantención y administración del sistema, por lo cual se concluye que los algoritmos GO y HPr poseen el enfoque correcto.

Con respecto a la minimización de la cantidad de roles, los algoritmos GO y HPr lograron llegar al óptimo, con $|R| = 12$. Por otra parte, HM (con $W = (1,0,0,0,\infty)$), y HPe obtuvieron como solución $|R| = 15$, una variación de $\triangle|R| = 3$ (25 %) con respecto al óptimo. Para el caso de HM, con $W = (1,1,1,1,1)$, se obtuvo $|R| = 21$, con $\triangle|R| = 9$ (75 %), mientras que con $W = (0,1,1,1,\infty)$, se obtuvo $|R| = 24$, con $\triangle|R| = 12$ (100 %). Este análisis permite concluir que los resultados que presentaron un $|R| > 15$ serán difícil de administrar y costosos de mantener, debido a la excesiva cantidad de roles. Tal es el caso de las soluciones generadas por el algoritmo HM en base a los vectores de peso $W = (1,1,1,1,1)$ y $W = (0,1,1,1,\infty)$.

En lo referido a la jerarquía de roles, los algoritmos HPe y HPr entregaron como resultado un sistema $RBAC$ plano (ver Figura 6.10 y 6.9 respectivamente), lo que es corroborado por la cantidad de relaciones existentes ($|RH| = 0$). Por otro lado, de la Figura 6.4 se desprende que en GO, cada rol está asociado a un sub-rol o super-rol o ambos, y dichos roles se organizan por medio de un árbol jerárquico. Dicha estructura posee la característica que $|RH| = |R| - 1 = 11$, lo que corresponde al mínimo número de relaciones entre los roles que asegura que cada rol está conectado, al menos, a algún otro rol. Por otra parte, las distintas soluciones propuestas por HM, poseen la peculiaridad que $|RH| > |R| - 1$, lo que corrobora la utilización de un grafo para la organización del conjunto de roles (ver Figuras 6.6, 6.7 y 6.8 respectivamente). Cabe destacar que estos resultados, coinciden con lo expuesto en el marco teórico.

Al comparar la jerarquía de roles de los algoritmos GO y HM, para cada uno de los vectores de peso utilizados, se puede concluir que GO estructura la solución de una forma compacta, mientras que HM utiliza una cantidad excesiva de relaciones. Esto último se debe a la forma en que HM resuelve el problema. A partir de los datos de entrada, HM genera un retículo, en donde cada nodo corresponde a un rol, posteriormente procede a reducirlo, utilizando un método voraz de poda o reestructuración. Esto implica que la reducción puede no ser la óptima, como es en este caso. Además, hay que considerar que un retículo puede ser un grafo, por lo que en esos casos, requiere una mayor cantidad de relaciones, que un árbol jerárquico, para mantener su estructura.

De la Tabla 7.2 se desprende los algoritmos GO y HPr tienen características similares, con la salvedad que HPr no posee jerarquía de roles. Además, ambos algoritmos asignan un rol a cada usuario, con el objetivo de facilitar la administración y mantención del sistema. También, estos algoritmos presentan un rol que tiene asociado todos los permisos del sistema, lo que en la literatura se conoce como "*super rol*" y que, generalmente, se relaciona con el administrador del sistema. Para los casos de HM con $W = (0,1,1,1,\infty)$ y $W = (1,1,1,1,1)$, éstos presentan un rol con un único permiso y una gran cantidad de roles por usuario, lo que refleja el bajo desempeño de HM para la resolución de este problema. Similarmente, para los caso de HM (con $W = (1,0,0,0,\infty)$) y HPe, se identifican usuarios con una gran cantidad de roles asociados, lo que dificulta la administración y mantención del sistema.

En base a toda la información analizada, se puede concluir que el algoritmo HM obtuvo un desempeño pobre, en comparación a los demás métodos. Por otra parte, los algoritmos GO y HPr se destacaron por obtener los mejores resultados, por lo que actualmente corresponden a las mejores soluciones propuestas. También, hay que considerar que HPe prácticamente no se destacó, y la gran cantidad de roles que éste asigna a algunos usuarios, lo distancia de la solución más conveniente para el problema estudiado.

Debido a la similitud de los resultados entre los algoritmos GO y HPr, se procedió a comparar la similitud entre los conjunto de roles generados, dando como resultado $JC = 1$ (o $JD = 0$), lo que implica que los conjunto evaluados son idénticos. Por lo tanto, GO complementa los resultados de HPr, incorporando a la solución un árbol jerárquico de roles.

Así, se puede concluir que el algoritmo que mejor se adapta a las necesidades, corresponde al *GO*, seguido por HPr, continuado por HPe, y por último, HM.

Complementando lo anterior, la Tabla 7.4 resume los resultados de las métricas wsc y $\frac{|DUPA_{out}|}{|DUPA_{in}|}$ de los algoritmos estudiados. De dicha tabla se desprende que todos los algoritmos aplicados lograron resolver el problema propuesto sin la necesidad de recurrir a la asignación directa de permisos a los respectivos usuarios, ya que $\frac{|DUPA_{out}|}{|DUPA_{in}|} = 0$ para todos los casos. Con respecto a la métrica wsc, con $W = (0,1,1,1,\infty)$, los resultados obtenidos son próximos uno a otros, con excepción de la observación perteneciente al algoritmo HM, la cual se aleja considerablemente de la tendencia de dichos datos. En base a esto, se puede concluir que, para el caso del $W = (0,1,1,1,\infty)$, el algoritmo HM obtuvo el peor desempeño ($wsc = 2146$), mientras que HPe obtuvo el mejor ($wsc = 636$). Por otra parte, para el caso $W = (1,0,0,0,\infty)$, la menor magnitud de wsc se obtuvo con HPr ($wsc = 12$), lo que corrobora lo sugerido por el diagrama de simetría de usuarios, que la menor cantidad de roles posibles es 12. Así, se puede concluir que los algoritmos HM y HPe obtuvieron los peores resultados ($wsc = 15$). Por último, para el caso en que $W = (1,1,1,1,1)$, nuevamente los resultados son próximos entre sí, con excepción de la observación perteneciente al algoritmo HM, la cual se aleja de toda tendencia, por lo que se concluye que HM es quien obtuvo el peor desempeño ($wsc = 2009$), mientras que HPe obtuvo el mejor ($wsc = 657$). El pobre desempeño de HM se debe a la excesiva cantidad de roles asociados a cada usuario ($|UA|$).

Tabla 7.4: Resume los resultados del estudio en base a las métricas consideradas.

Métrica	*GO*	HM	HPe	HPr
wsc con $W = (0,1,1,1,\infty)$	678	2146	636	667
wsc con $W = (1,0,0,0,\infty)$	12	15	15	12
wsc con $W = (1,1,1,1,1)$	690	2009	651	679
$\frac{\|DUPA_{out}\|}{\|DUPA_{in}\|}$ con $W = (0,1,1,1,\infty)$	0.00	0.00	0.00	0.00

Por otra parte, HPr y HPe se vieron beneficiados erróneamente por la ausencia de una jerarquía de roles. En [17] no se especifica qué se debe hacer para el caso en que $|tr(RH)| =$

0, con $w_h = 1$. Para solucionar este inconveniente, se debe generar los conjuntos RH ausentes de los respectivos algoritmos. Para facilitar la corrección de los valores de wsc, se puede considerar como supuesto que los roles se organizan por medio de un árbol jerárquico con $|R|-1$ relaciones entre ellos, en donde la presencia del *super* rol es obligatoria. La Tabla 7.5 corrige los resultados de la Tabla 7.4 en base a este supuesto.

Tabla 7.5: Resumen corregido de los resultados del estudio en base a las métricas consideradas.

Métrica	GO	HM	HPe	HPr
wsc con $W = (0,1,1,1,\infty)$	678	2146	737	678
wsc con $W = (1,0,0,0,\infty)$	12	15	16	12
wsc con $W = (1,1,1,1,1)$	690	2009	753	690
$\frac{\lvert DUPA_{out}\rvert}{\lvert DUPA_{in}\rvert}$ con $W = (0,1,1,1,\infty)$	0.00	0.00	0.00	0.00

Para facilitar la comparación de los resultados de la Tabla 7.4, se procedió a clasificarlas en base al ranking expuesto en la sección *Algoritmos*, dando como resultado la Tabla 7.6. De dicha tabla se desprende que los algoritmos que obtuvieron un mejor desempeño, según las métricas utilizadas, corresponden a HPr y GO con $ranking_{promedio}$ = 1,75, seguido por el algoritmo HPe con $ranking_{promedio}$ = 3,13, y por último, HM con $ranking_{promedio}$ = 3,38. Estos resultados, complementan el análisis de las características de las soluciones propuestas, ya que las métricas usadas indican que GO y HPr son relevantes para este estudio, pero tras considerar el problema existente con el conjunto RH, se desprende que el algoritmo GO entregó la solución más adecuada para este problema.

Tabla 7.6: Clasificación del desempeño de cada algoritmo.

Estudio	GO	HM	HPe	HPr
$W = (0,1,1,1,\infty)$	1.5	4	3	1.5
$W = (1,0,0,0,\infty)$	1.5	3	4	1.5
$W = (1,1,1,1,1)$	1.5	4	3	1.5
$\frac{\lvert DUPA_{out}\rvert}{\lvert DUPA_{in}\rvert}$ con $W = (0,1,1,1,\infty)$	2.5	2.5	2.5	2.5
Promedio	**1.75**	**3.38**	**3.13**	**1.75**

Con respecto a la denotación de cada rol, la Figura 7.2 ilustra la jerarquía organizacional esperada para los roles generados, identificando únicamente los roles organizacionales de "*Administrador*", "*Profesor*", "*Alumno*", "*Ayudante*", "*Visitante*", y "*Cliente*". Como se mencionó anteriormente, existen algunas variantes de estos roles, por lo que se debió verificar el contexto de los permisos para poder denominar correctamente los roles generados. Tanto para el caso de HPe como HM, la cantidad excesiva de roles que es asignado a algunos usuarios, impide la denominación de los roles en base a los roles organizacionales, volviendolos artificial desde la perspectiva del administrador. Según el marco teórico, los roles artificiales son difíciles de administrar y costosos de mantener, por lo que se descarta totalmente la utilización de dichos algoritmos para la implementación del sistema $RBAC_{ADS}$. Por otra parte, tanto los algoritmos HPr como GO pueden utilizar la denotación anteriormente descrita, pero teniendo la precaución de distinguir cada tipo de rol con respecto a la funcionalidad que éste lo destaca. Tras aplicar lo anterior, se genera el árbol jerárquico representado por la Figura 7.3. En este punto, hay que aclarar que la solución encontrada satisface los requerimientos de la organización, pero existen otras variantes que pueden ser utilizadas. Por ejemplo, en vez de crear distintas versiones de los tipos de usuarios, se puede considerar la jerarquía la Figura 7.2 como base, y estos adquieren sus funcionalidades distintivas por medio de otros roles especiales, generando así, una sola versión de éstos, pero ocasionando la utilización de roles artificiales, por lo que se podría crear un balance entre estas dos propuestas con la finalidad de mejorar la interpretabilidad de los roles, y así, se utilice la misma terminología en toda la organización. Tras corroborar los resultados obtenidos con el actual administrador del sistema $RBAC_{ADS}$, se concluyó que la jerarquía propuesta es un buen acercamiento, y

que actualmente no es necesario separar los roles para enfatizar en la interpretabilidad. Así, mediante el presente estudio se logró comprobar la utilidad de la minería de roles para la resolución del RMP, y también que el $RMiner$ permite implementar, fácil y rápidamente, los algoritmos más destacados en la literatura.

Figura 7.2: Jerarquía organizacional esperada.

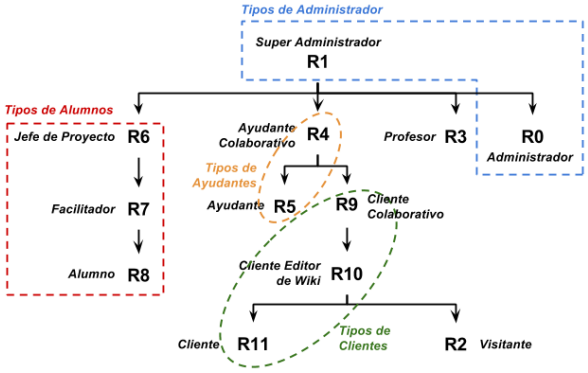


Figura 7.3: Jerarquía de roles sugerida.

Por último, para poder incorporar estos resultados a la tabla resumen de los estudios existentes (Tabla 4.10), se debió volver a calcular el ranking de los algoritmos, considerando únicamente los 4 algoritmos que son usados en el presente estudio. Lo anterior dio como resultado la Tabla 7.7. De dicha tabla se desprende que el algoritmo que obtuvo un mejor desempeño, corresponde al HM con $ranking_{promedio} = 2{,}05$, seguido por HPe y GO con $ranking_{promedio} = 2{,}44$, y por último, HPr con $ranking_{promedio} = 3{,}08$. Al considerar los resultados de la Tabla 7.6, se puede predecir que GO quedará posicionado más abajo que HPe. Además, al tomar en cuenta el desempeño de HM y Hpe, se concluye que la distancia entre estos dos se acortará significativamente, pero no cambiará la posición de estos.

Tabla 7.7: Clasificación del desempeño de los algoritmos en base a otros conjuntos de datos.

Estudio	HM	HPr	HPe	GO
$W = (1,0,0,0,\infty)$	3.75	1.19	3.06	2
$W = (0,1,1,1,\infty)$	1.25	3.63	2.13	3
$W = (1,1,1,1,1)$	1.13	4	2.38	2.50
$\frac{\lvert DUPA_{out}\rvert}{\lvert DUPA_{in}\rvert}$ con $W = (0,1,1,1,\infty)$	2.06	3.50	2.19	2.25
Promedio	***2.05***	***3.08***	***2.44***	***2.44***

La Tabla 7.8 resume la clasificación general de los algoritmos estudiados para cada uno de las métricas consideradas. De esta tabla se desprende que las variaciones en el ranking del algoritmo GO fue suficiente para ubicarse en segundo lugar, distanciandose de HPe. El resto de los algoritmos mantuvieron su posición.

Tabla 7.8: Clasificación del desempeño de los algoritmos en base a los conjuntos estudiados.

Estudio	HM	HPr	HPe	GO
$W = (1,0,0,0,\infty)$	3.67	1.22	3.17	1.94
$W = (0,1,1,1,\infty)$	1.56	3.39	2.22	2.83
$W = (1,1,1,1,1)$	1.44	3.72	2.44	2.39
$\frac{\lvert DUPA_{out}\rvert}{\lvert DUPA_{in}\rvert}$ con $W = (0,1,1,1,\infty)$	2.11	3.39	2.22	2.28
Promedio	***2.19***	***2.93***	***2.51***	***2.36***
Ranking Original	***2.05***	***3.08***	***2.44***	***2.44***
Variación del Ranking	***0.15***	***-0.15***	***0.08***	***-0.08***

Conclusiones y Futuros Trabajos

En este capítulo se expone las conclusiones del estudio realizado, destacando los aspectos más relevantes de la herramienta utilizada para llevar a cabo los diversos experimentos del estudio. Además, se recomiendan algunos enfoques para futuros trabajos, tanto en el área de la ingeniería de roles como en el perfeccionamiento de la herramienta usada.

También se logró cumplir con el objetivo principal. Se demostró la eficacia de la minería de roles para la resolución del RMP para el caso de la plataforma de la asignatura ADS, ya que se encontró el conjunto de roles óptimos del problema dado.

Por otra parte, se cumplió con los siguientes objetivos secundarios:

1. Se comprobó que el criterio propuesto en [15], es eficaz para seleccionar los enfoques del RMP más aptos para el sistema dado.

2. Se verificó la eficacia de las métricas "*Weighted Structural Complexity*" (wsc) y el porcentaje de asignaciones no cubiertas ($\frac{|DUPA_{out}|}{|DUPA_{in}|}$), para la evaluación de la calidad de los roles propuestos por los algoritmos seleccionados. En particular, para el caso en que $w_h = 1$, se deben generar los conjuntos jerárquicos de roles ausentes, y destacando, para cada caso, que dicho conjunto no forma parte de la solución original.

3. Se midió cuantitativamente el desempeño de los algoritmos GO, HM, HPr y HPe por medio de un ranking de los valores de wsc para cada uno de los vectores de peso seleccionados. Tras ordenar los algoritmos, de menor a mayor clasificación, se obtuvo: (1) GO y HPr (mismo R); (2) HPe; y (3) HM. Se escogió GO por su jerarquía de roles.

4. Se actualizó el estudio realizado por Molloy et al. [18], incorporando los resultados del presente texto. Así, los algoritmos más importantes de la minería de roles se organizan, de menor a mayor, en base a su desempeño, de la siguiente forma: HM, GO, HPe y HPr.

Además de lo anterior, se destaca lo siguiente:

- La cantidad de roles asignado a cada usuario es uno de los aspectos más importantes para la evaluación de un rol.
- Se debe priorizar la reducción de $|UA|$ sobre $|PA|$.
- Los vectores de peso, requeridos por *WSC*, deben ser seleccionados en base a los enfoques del RMP, que son considerados.
- El desempeño de cada algoritmo depende del conjunto de datos, del vector de peso, y del método usado en la optimización de $f(x)$. Además, la mayoría de los algoritmos utiliza un método voraz para la optimización de su respectiva función objetivo, lo que afecta considerablemente en la calidad de la solución propuesta.
- *WSCO* se destaca, sobre los demás enfoques del RMP, por su adaptabilidad a diversos objetivos, y por el uso de wsc.
- En particular para este estudio, las entidades más relevantes son RH, UA y R.

Finalmente, con respecto a los trabajos futuros, la mayoría de los algoritmos aplicados utilizan un método voraz para su optimización, por lo que sería relevante complementar este estudio, utilizando alguna otra heurística (p. ej: algoritmo genético, algoritmo de colonia de hormigas, etc.). También, se podría considerar un nuevo vector de peso $W = (1,0,1,1,\infty)$, contrastar los resultados con algún algoritmo que considere restricciones. En relación a WSC, es de vital importancia que éste considere la cantidad de roles que puede ser asignado a cada usuario y la interpretabilidad de los roles. En cuanto a la herramienta *RMiner*, ésta puede ser mejorada al incorporar la representación gráfica de las soluciones de cada algoritmo junto a la jerarquía de estos. Además, podría generar automáticamente las características consideradas en el presente estudio. También, se

podría añadir un módulo que permita comparar dos o más soluciones, tanto a nivel de características como de jerarquía. Por último, para su uso sin fin de lucro en diversas organizaciones, es de vital importancia que esta herramienta se sincronice con la base de datos utilizada y actualice la información correspondiente fácilmente.

Bibliografía

[1] American National Standars Institute - ANSI. ANSI webstore. `https://webstore.ansi.org/`. Visitada el 7 de enero de 2018.

[2] G. Birkhoff. *Lattice Theory*. Number v. 25,parte 2 in American Mathematical Society colloquium publications. American Mathematical Society, 1940.

[3] Edward J. Coyne. Role engineering. In *Proceedings of the First ACM Workshop on Role-based Access Control*, RBAC '95, New York, NY, USA, 1996. ACM.

[4] E. Ogrodnik, H. Astudillo. Respaldo del proyecto "Minería de roles en una plataforma de gestión de proyectos académicos". `https://goo.gl/1pHvrX`. Visitada el 7 de enero de 2018.

[5] Alina Ene, William Horne, Nikola Milosavljevic, Prasad Rao, Robert Schreiber, and Robert E. Tarjan. Fast exact and heuristic methods for role minimization problems. In *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies*, SACMAT '08, pages 1–10, New York, NY, USA, 2008. ACM.

[6] Alina Ene, William Horne, Nikola Milosavljevic, Prasad Rao, Robert Schreiber, and Robert E. Tarjan. Fast exact and heuristic methods for role minimization problems. In *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies*, SACMAT '08, pages 1–10, New York, NY, USA, 2008. ACM.

[7] Ludwig Fuchs and Günther Pernul. Hydro — hybrid development of roles. In *Proceedings of the 4th International Conference on Information Systems Security*, ICISS '08, pages 287–302, Berlin, Heidelberg, 2008. Springer-Verlag.

[8] Q. Guo, J. Vaidya, and V. Atluri. The role hierarchy mining problem: Discovery of optimal role hierarchies. In *2008 Annual Computer Security Applications Conference (ACSAC)*, pages 237–246, Dec 2008.

[9] Chao Huang, Jian-ling Sun, Xin-yu Wang, and Yuan-jie Si. Minimal role mining method for web service composition. *Journal of Zhejiang University SCIENCE C*, 11(5):328–339, May 2010.

[10] International Organization for Standardization. ISO webstore. `https://www.iso.org/store.html`. Visitada el 7 de enero de 2018.

[11] Jean-Philippe Lang. Redmine main page. `http://www.redmine.org/`. Visitada el 7 de enero de 2018.

[12] Ruixuan Li, Huaqing Li, Wei Wang, Xiaopu Ma, and Xiwu Gu. Rminer: A tool set for role mining. In *Proceedings of the 18th ACM Symposium on Access Control Models and Technologies*, SACMAT '13, pages 193–196, New York, NY, USA, 2013. ACM.

[13] Haibing Lu, Jaideep Vaidya, and Vijayalakshmi Atluri. Optimal boolean matrix decomposition: Application to role engineering. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, pages 297–306, Washington, DC, USA, 2008. IEEE Computer Society.

[14] S. Mandala, M. Vukovic, J. Laredo, Y. Ruan, and M. Hernandez. Hybrid role mining for security service solution. In *2012 IEEE Ninth International Conference on Services Computing*, pages 210–217, June 2012.

[15] Barsha Mitra, Shamik Sural, Jaideep Vaidya, and Vijayalakshmi Atluri. A survey of role mining. *ACM Comput. Surv.*, 48(4):50:1–50:37, February 2016.

[16] Ian Molloy, Hong Chen, Tiancheng Li, Qihua Wang, Ninghui Li, Elisa Bertino, Seraphin Calo, and Jorge Lobo. Mining roles with semantic meanings. In *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies*, SACMAT '08, pages 21–30, New York, NY, USA, 2008. ACM.

[17] Ian Molloy, Hong Chen, Tiancheng Li, Qihua Wang, Ninghui Li, Elisa Bertino, Seraphin Calo, and Jorge Lobo. Mining roles with multiple objectives. *ACM Trans. Inf. Syst. Secur.*, 13(4):36:1–36:35, December 2010.

[18] Ian Molloy, Ninghui Li, Tiancheng Li, Ziqing Mao, Qihua Wang, and Jorge Lobo. Evaluating role mining algorithms. In *Proceedings of the 14th ACM Symposium on Access Control Models and Technologies*, SACMAT '09, pages 95–104, New York, NY, USA, 2009. ACM.

[19] A. C. O'Connor and R. J. Loomis. Economic analysis of role-based access control. RTI International, December 2010. Report for NIST.

[20] M. Hall R. Kirkby P. Reutemann A. Seewald Remco R. Bouckaert, E. Frank and D. ScuseR. Weka manual for version 3-6-0, 12 2008.

[21] Ruixuan Li, Huaqing Li, Wei Wang, Xiaopu Ma, Xiwu Gu. RMiner executable program. `https://storage.googleapis.com/google-code-archive-downloads/v2/code.google.com/rminer/RMiner.zip`. Visitada el 7 de enero de 2018.

[22] Ruixuan Li, Huaqing Li, Wei Wang, Xiaopu Ma, Xiwu Gu. RMiner manual. `https://storage.googleapis.com/google-code-archive-downloads/v2/code.google.com/rminer/RMinerManual.pdf`. Visitada el 7 de enero de 2018.

[23] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-based access control models. *Computer*, 29(2):38–47, February 1996.

[24] Andreas Schaad. The role-based access control system of a european bank: A case study and discussion. In *In Proceedings of the Sixth ACM Symposium on Access Control Models and Technologies*, pages 3–9. ACM Press, 2001.

[25] Jaideep Vaidya, Vijayalakshmi Atluri, and Qi Guo. The role mining problem: A formal perspective. *ACM Trans. Inf. Syst. Secur.*, 13(3):27:1–27:31, July 2010.

[26] Jaideep Vaidya, Vijayalakshmi Atluri, Qi Guo, and Nabil Adam. Migrating to optimal rbac with minimal perturbation. In *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies*, SACMAT '08, pages 11–20, New York, NY, USA, 2008. ACM.

[27] Dana Zhang, Kotagiri Ramamohanarao, and Tim Ebringer. Role engineering using graph optimisation. In *Proceedings of the 12th ACM Symposium on Access Control Models and Technologies*, SACMAT '07, pages 139–144, New York, NY, USA, 2007. ACM.

Apéndice A

Códigos

Algorithm 1: Graph Optimisation $GO(DUPA)$

Input: user u, permissions p and permission to user assignments $u \rightarrow p$

```
identify each user permission sets as possible role
calculate optimisation metric = number of edges + numer of roles
{merge roles until stable}
while there can be more merges do
    identify two roles
    split or merge depending on permission set overlap
    calculate new optimisation metric = number of edges + numer of roles
    if new optimisation metric > old optimisation metric then
        undo operation
    else
        old optimisation metric = new optimisation metric
    end
end
```

Apéndice B

Permisos de Acceso

Tabla B.1: Permisos de acceso de la plataforma *ADS*.

P	**Permiso**	**Descripción**
p_1	Crear proyectos	Permite agregar proyectos a la plataforma.
p_2	Editar proyectos	Permite editar las propiedades de los proyectos de la plataforma.
p_3	Cerrar / reabrir proyectos	Permite cerrar o volver a abrir algún proyecto de la plataforma.
p_4	Seleccionar los módulos de los proyectos	Permite habilitar o deshabilitar los módulos de Redmine que serán usados en cada proyecto.
p_5	Administrar miembros de los proyectos	Permite agregar o eliminar miembros de los respectivos proyectos de la plataforma. Además, otorga la capacidad de cambiar los roles de los miembros existentes.
p_6	Administrar versiones de los proyectos	Permite agregar, editar o eliminar las versiones de cada proyecto de la plataforma.
p_7	Crear subproyectos	Permite agregar subproyectos en los proyectos correspondientes.

p_8	Administrar consultas privadas	Permite al usuario guardar, editar o eliminar sus consultas "privadas"de los proyectos pertinenetes.
p_9	Administrar consultas públicas	Permite guardar, editar o eliminar las consultas públicas de los respectivos proyectos.
p_{10}	Administrar foros	Permite al usuario agregar, editar y eliminar foros de los respectivos proyectos.
p_{11}	Ver mensajes	Permite ver las publicaciones (o mensajes) en los foros correspondientes de cada proyecto.
p_{12}	Publicar mensajes	Permite crear nuevas publicaciones (o mensajes) en los foros de los proyectos pertinentes.
p_{13}	Editar mensajes	Permite editar cualquier publicación o mensaje de los foros de los respectivos proyectos. Además, otorga la capacidad de eliminar cualquier archivo adjunto en dicho mensaje.
p_{14}	Editar sus propios mensajes	Permite editar las publicaciones o mensajes escritos por el propio usuario.
p_{15}	Eliminar mensajes	Permite eliminar cualquier publicación o mensaje.
p_{16}	Eliminar sus propios mensajes	Permite que el usuario elimine sus propias publicaciones o mensajes.
p_{17}	Agregar documentos	Permite agregar documentos al respectivo proyecto.
p_{18}	Editar documentos	Permite editar documentos del proyecto pertienente.
p_{19}	Eliminar documentos	Permite eliminar documentos del proyecto correspondiente.
p_{20}	Ver documentos	Permite ver los documentos de los proyectos pertinenetes.
p_{21}	Administrar archivos	Permite agregar, editar y eliminar archivos del proyecto correspondiente.
p_{22}	Ver archivos	Permite ver los archivos del respectivo proyecto.

p_{23}	Administrar categorías de temas	Permite agregar, editar o eliminar categorías de temas. Redmine considera, por defecto, las clasificaciones: error, característica y soporte.
p_{24}	Ver temas	Permite ver los temas de los proyectos correspondientes.
p_{25}	Agregar temas	Permite crear nuevos temas en los respectivos proyectos.
p_{26}	Editar temas	Permite editar los temas de los proyectos pertinentes.
p_{27}	Copiar temas	Permite copiar temas (basar un tema con respecto a otro).
p_{28}	Administrar las relaciones entre temas	Permite agregar o eliminar relaciones entre los temas de un proyecto.
p_{29}	Eliminar temas	Permite eliminar temas de los proyectos correspondientes.
p_{30}	Importar temas	Permite importar los temas de cada proyecto.
p_{31}	Cambiar la visibilidad de los temas	Permite al usuario cambiar la visibilidad (pública o privada) de los temas de los respectivos proyectos.
p_{32}	Cambiar la visibilidad de los temas propios	Permite al usuario cambiar la visibilidad (pública o privada) de sus temas.
p_{33}	Ver notas privadas	Permite al usuario ver sus comentarios o notas privadas.
p_{34}	Cambiar visibilidad de las notas	Permite al usuario cambiar la visibilidad (pública o privada) de los comentarios o notas de los respectivos temas.
p_{35}	Agregar notas	Permite agregar comentarios (o nota) a los temas existentes.
p_{36}	Editar notas	Permite editar los comentario o notas de los temas existentes.
p_{37}	Editar notas propias	Permite al usuario editar sus comentarios o notas.

p_{38}	Administrar subtareas	Permite agregar y eliminar subtareas a los temas.
p_{39}	Administrar consultas públicas	Permite agregar, editar o eliminar consultas públicas a la base de datos, relacionandolo con algún tema del proyecto correspondiente.
p_{40}	Guardar consultas	Permite al usuario guardar sus propias consultas a la base de datos, relacionandolo con algún tema del proyecto correspondiente.
p_{41}	Ver diagrama de Gantt	Permite ver el diagrama de Gantt del proyecto pertinente.
p_{42}	Ver calendario	Permite ver el calendario del respectivo proyecto.
p_{43}	Ver la lista de vigilantes	Permite ver los vigilantes del tema correspondiente.
p_{44}	Agregar vigilantes	Permite asignarles a otros usuarios el cargo de "vigilantes"de algún tema.
p_{45}	Eliminar vigilantes	Permite eliminar los vigilantes de los temas pertinenetes.
p_{46}	Administrar noticias	Permite agregar, editar y eliminar las noticias del proyecto.
p_{47}	Ver noticias	Permite ver las noticias publicadas en los proyectos pertinentes.
p_{48}	Comentar noticias	Permite agregar comentarios a las noticias del proyecto correspondiente.
p_{49}	Administrar repositorio	Permite configurar el repositorio del respectivo proyecto.
p_{50}	Navegar por el repositorio	Permite navegar y ver el contenido del repositorio del proyecto correspondiente.
p_{51}	Ver conjuntos de cambios	Permite ver los conjuntos de cambios del repositorio pertinente.
p_{52}	Confirmar acceso	Permite poder modificar el repositorio del respectivo proyecto.

p_{53}	Registrar el tiempo invertido	Permite al usuario registrar su tiempo invertido en cada uno de los proyectos en que participa.
p_{54}	Ver el tiempo invertido	Permite ver los registros de tiempo de cada usuario en el proyecto correspondiente.
p_{55}	Editar registros de tiempo	Permite editar los registros de tiempo de cualquier usuario.
p_{56}	Editar los propios registros de tiempo	Permite al usuario editar sus propios registros de tiempo.
p_{57}	Administrar las actividades del proyecto	Permite agregar, editar o eliminar actividades de los proyectos de la plataforma.
p_{58}	Administrar páginas "wiki"	Permite crear o eliminar la página "wiki"del respectivo proyecto. Al eliminar una "wiki", se borra la página correspondiente con sus archivos e historial pertinente.
p_{59}	Cambiar el nombre de las páginas "wiki"	Permite cambiar el nombre de las páginas "wiki". Además, otorga la capacidad de asignar las páginas "wiki.ª sus respectivas páginas principales (sitio de mayor jerarquía).
p_{60}	Eliminar páginas "wiki"	Permite eliminar la página "wiki"del respectivo proyecto. Al eliminar una "wiki", se borra la página correspondiente con sus archivos e historial pertinente.
p_{61}	Ver páginas "wiki"	Permite ver las páginas "wiki".
p_{62}	Exportar páginas "wiki"	Permite exportar las páginas "wiki"(pdf, html, etc.).
p_{63}	Ver el historial de versiones de la página "wiki"	Permite ver y distinguir las versiones de las páginas "wiki".
p_{64}	Editar páginas "wiki"	Permite editar páginas "wiki"desprotegidas.

p_{65}	Eliminar archivos adjuntos de las páginas "wiki"	Permite eliminar archivos adjuntos de las páginas "wiki".
p_{66}	Proteger páginas "wiki"	Permite bloquear o desbloquear la edición de las páginas wiki. Además, otorga la capacidad de poder editar páginas bloqueadas.

Apéndice C

Roles Desplegados

Tabla C.1: Roles desplegados en la plataforma *ADS*.

Rol	**Permisos**	**Núm. permisos**	% **permisos**
Reporter	p_{11}, p_{12}, p_{14}, p_{20}, p_{22}, p_{24}, p_{25}, p_{35}, p_{40}, p_{41}, p_{42}, p_{47}, p_{48}, p_{50}, p_{51}, p_{53}, p_{54}, p_{61}, p_{63}	19	28,79 %
Developer	p_{6}, p_{11}, p_{12}, p_{14}, p_{20}, p_{21}, p_{22}, p_{23}, p_{24}, p_{25}, p_{26}, p_{28}, p_{33}, p_{34}, p_{35}, p_{38}, p_{40}, p_{41}, p_{42}, p_{47}, p_{48}, p_{50}, p_{51}, p_{52}, p_{53}, p_{54}, p_{60}, p_{61}, p_{63}, p_{64}	30	45,45 %
Non member	p_{11}, p_{12}, p_{20}, p_{22}, p_{24}, p_{25}, p_{35}, p_{40}, p_{41}, p_{42}, p_{47}, p_{48}, p_{50}, p_{51}, p_{54}, p_{61}, p_{63}	17	25,76 %
Anonymous	p_{11}, p_{20}, p_{22}, p_{24}, p_{41}, p_{42}, p_{47}, p_{50}, p_{51}, p_{54}, p_{61}, p_{63}	12	18,18 %
Manager	p_{1}, p_{2}, p_{3}, p_{4}, p_{5}, p_{6}, p_{7}, p_{8}, p_{9}, p_{10}, p_{11}, p_{12}, p_{13}, p_{14}, p_{15}, p_{16}, p_{17}, p_{18}, p_{19}, p_{20}, p_{21}, p_{22}, p_{23}, p_{24}, p_{25}, p_{26}, p_{27}, p_{28}, p_{29}, p_{30}, p_{31}, p_{32}, p_{33}, p_{34}, p_{35}, p_{36}, p_{37}, p_{38}, p_{39}, p_{40}, p_{41}, p_{42}, p_{43}, p_{44}, p_{45}, p_{46}, p_{47}, p_{48}, p_{49}, p_{50}, p_{51}, p_{52}, p_{53}, p_{54}, p_{55}, p_{56}, p_{57}, p_{58}, p_{59}, p_{60}, p_{61}, p_{62}, p_{63}, p_{64}, p_{65}, p_{66}	66	100,00 %

Apéndice D

Restricciones Desplegadas

Tabla D.1: Restricciones de la plataforma *ADS*.

Tipo de Restricción	**Descripción**
Prerrequisito	p_{12}, p_{13}, p_{14}, p_{15} y p_{16} requieren p_{11}
	p_{17}, p_{18} y p_{19} requieren p_{20}
	p_{14}, p_{15} y p_{16} requieren p_{12}
	p_{21} requiere p_{22}
	p_{23}, p_{25}, p_{26}, p_{27}, p_{28}, p_{29}, p_{30}, p_{31}, p_{32}, p_{33}, p_{34}, p_{35}, p_{36}, p_{37}, p_{38}, p_{39}, p_{40}, p_{41}, p_{42}, p_{43}, p_{44} y p_{45} requieren p_{24}
	p_{27} requiere p_{26}
	p_{30} y p_{32} requieren p_{25}
	p_{37} requiere p_{35}
	p_{44} y p_{45} requieren p_{43}
	p_{48} y p_{46} requieren p_{47}
	p_{49}, p_{51} y p_{52} requieren p_{50}
	p_{56} requiere p_{53}
	p_{53} requiere p_{54}
	p_{58}, p_{59}, p_{60}, p_{62}, p_{63}, p_{64}, p_{65} y p_{66} requieren p_{61}

Excluyente	p_{58} y p_{60} se excluyen
	r_0, r_1, r_2, r_3 y r_4 se excluyen
Cardinalidad	sólo puede haber un usuario con r_2
	sólo puede haber un usuario con r_3