Un Algoritmo Basado en la Programación Genética para Minería de Datos

Jose Aguilar
Universidad de los Andes, Facultad de Ingeniería, Departamento de Computación,
Mérida, Venezuela, 5101
aguilar@ing.ula.ve

y

J. Altamiranda
Universidad de los Andes, Facultad de Ingeniería, Departamento de Computación,
Mérida, Venezuela, 5101

Abstract

Data Mining is composed by a set of methods to extract knowledgement from large Database. One of these methods is Genetic Programming. In this work we use this method to build a Data Mining System that define a set of patterns in order to classify the data. We define a grammar, which is used by the genetic programming in order to define the rules that represent the patterns. In this way, we can group the data in class and simplify the information in the database according to the set of patterns.

Keywords: Data Mining, Genetic Programming.

Resumen

La Minería de Datos envuelve un conjunto de métodos para la extracción de conocimiento desde grandes bases de datos. Uno de esos métodos es la Programación Genética, que es utilizada en este trabajo como herramienta para construir un Sistema de Minería de Datos que permite definir un conjunto de patrones para clasificar los datos contenidos en una base de datos. Para realizar esta tarea se construyó una "gramática", la cual es usada por la Programación Genética para construir las "reglas" que representan los patrones de los datos contenidos en la base de datos. De esta forma se pueden agrupar los datos en clases, de manera de simplificar el contenido de la base de datos en un conjunto de patrones informativos.

Palabras claves: Minería de Datos, Estructuras Gramaticales, Programación Genética.

1 Introducción

El incremento en el uso de computadoras y sistemas distribuidos ha generado una explosión en la cantidad de información disponible, la cual debe ser procesada por mecanismos sofisticados para poder ser usada eficientemente. Una técnica desarrollada para tal fin es la Minería de Datos, cuya meta es descubrir conocimientos desde un conjunto de datos del mundo real [7, 8, 9]. La Minería de Datos y la Búsqueda de Conocimientos en Base de Datos son temas de interés para investigadores en áreas tales como: base de datos, aprendizaje automático, programación lógica inductiva, inteligencia artificial, estadística, entre otros.

Este trabajo sigue el espíritu de las áreas de Minería de Datos y Descubrimiento de Conocimientos en Bases de Datos. En particular, estaremos interesados en el conocimiento descubierto expresado como reglas de clasificación "SI <CAUSA> – ENTONCES <CONSECUENCIAS>". La parte SI de la regla contiene una combinación lógica de condiciones, mientras que la parte ENTONCES contiene atributos que son causados por la parte SI de la regla.

Por otro lado, existen varios paradigmas para algoritmos de Minería de Datos. En este trabajo nos focalizaremos en la Programación Genética. El uso de la Programación Genética para el descubrimiento de reglas de clasificación, en el espíritu de la Minería de Datos es un área relativamente inexplorada [8]. Se cree que este es un enfoque prometedor, debido a la eficiencia de la Programación Genética en la búsqueda y construcción de patrones que se adapten a un entorno según una estructura dada (en nuestro caso, una estructura gramatical).

Así, en este trabajo se construirá un Sistema de Minería de Datos que emplea la Programación Genética para descubrir patrones, los cuales son reglas clasificadoras de datos. La Programación Genética utiliza una estructura gramatical, que define como debe ser la forma de las reglas clasificadoras permitidas, para controlar las reglas generadas.

2 Marco Teórico

2.1 Minería de Datos

La Minería de Datos es un término genérico que engloba las técnicas y herramientas usadas para extraer información útil desde grandes bases de datos. En general, las técnicas de Minería de Datos intentan obtener patrones o modelos a partir de los datos recopilados, este proceso involucra un análisis de los datos, el reconocimiento de patrones sobre el conjunto de datos y una clasificación o agrupación de los mismos. Los algoritmos de Minería de Datos se enmarcan en un proceso completo de extracción de información conocido como KDD (Knowledge Discovery in Databases), que se encarga, además, de la preparación de los datos y de la interpretación de los resultados obtenidos

Entre las técnicas que pueden ser usadas en tareas de Minería de Datos tenemos: Arboles de Decisión, Redes Neuronales, Algoritmos Genéticos, Búsqueda de Asociaciones, Programación Genética, Lógica Difusa, etc. Dichas técnicas toman los datos y los transforman en información útil y entendible [7, 8, 9].

2.2 Estructuras Gramaticales

Las estructuras gramaticales permiten definir el formato o plantilla que debe seguir un lenguaje dado [2]. Es decir, es una notación formal que describe la manera como deben ser integrados los elementos que componen un lenguaje. Consta de una secuencia de declaraciones de operaciones permitidas. Para poder usar una estructura gramatical en un problema se deben prever las siguientes fases:

– **Análisis Léxico:** Su principal tarea es la de leer los caracteres de entrada y producir como salida una secuencia de átomos. Los átomos son entidades que identifican una secuencia lógica de caracteres [2]. Algunos tipos de átomos son: nombres (identificadores), espacios en blanco, operadores (+, -, =, <=), números (enteros, reales y complejo), etc.

– **Análisis sintáctico:** Obtiene la cadena de átomos producida por el analizador léxico y verifica que estos pueden ser generados por la gramática [2]. La gramática esta conformada por un conjunto de reglas. Ejemplos de estas reglas pueden ser, por ejemplo para el caso de un Identificador:

Letra = a | A | b | B | ... | z | Z

dígito = 0 | 1 | 2 | 3 | ... | 9

Letra_o_Dígito = letra | dígito

Identificador = Letra(Letra_o_Dígito)*

– **Análisis semántico:** Revisa para asegurar que los componentes gramaticales construidos en el analizador sintáctico estén asociados a una lógica de programación.

El uso de estructuras gramaticales concede gran flexibilidad al formato de las reglas, ya que especifica que atributos pueden aparecer en la construcción de ellas. Las estructuras gramaticales permiten formar patrones interesantes y regulares de los registros que se encuentran en la base de datos.

2.3 Programación Genética

La Computación Evolutiva aplica las teorías de la evolución natural y la genética en la adaptación evolutiva de estructuras computacionales, proporcionando un medio alternativo para resolver problemas complejos en diversas áreas. Se basa en la utilización de una población de posibles soluciones a un problema dado, lo cual es análogo a una población de organismos vivos que evoluciona en cada generación. La Computación Evolutiva combina los mejores individuos de la población y transmite las características de dichos individuos a sus descendientes [1].

La Programación Genética fue creada por John Koza a finales de los años 80 [1, 3, 4, 5]. En esta técnica, los individuos de la población representan esquemas o procedimientos potenciales de solución al problema a ser optimizado. A la solución ofrecida por cada individuo se le asigna una aptitud (un valor numérico), la cual indica cuan cercana esta de la solución optima. Nuevos individuos son generados por procedimientos análogos a la reproducción biológica, con padres escogidos de la población existente con una probabilidad proporcional a su aptitud. Los operadores evolutivos incluyen copia, cruce y mutación, entre otros. Los individuos nuevos reemplazan a los miembros menos aptos de la población, así la aptitud de la población mejorará con cada generación, hasta obtener el mejor individuo que representará la solución del problema.

3 diseño General Del Sistema

3.1 Diseño Preliminar

El Sistema de Minería de Datos esta formado por tres componentes:

1. Una estructura gramatical general.
2. Una base de datos que contiene la información a ser clasificada
3. Un algoritmo que realice la construcción y reconocimiento de patrones, basado en la Programación Genética.

Además, nuestro Sistema de Minería de Datos trabajará de acuerdo al siguiente macroalgoritmo:

1. Definición de una gramática general para ser utilizada por la Programación Genética. La gramática general es usada para describir la manera como se van a construir las reglas clasificadoras. Ella garantiza la construcción de individuos válidos.
2. Extracción de los átomos para la gramática general desde la base de datos objeto de estudio. Para realizar esta tarea, desde la base de datos se toman palabras contenidas en ella (átomos), que serán utilizados por la Programación Genética para construir las reglas clasificadoras.
3. Utilización de la Programación Genética para construir y evaluar las reglas clasificadoras. En este caso, cada regla clasificadora será un individuo, el cual será evaluado según su aptitud para agrupar la información. Además, la Programación Genética proporciona nuevos individuos usando sus mecanismos evolutivos.

El proceso desde el paso 2 es iterado hasta que el conjunto de reglas obtenidas logre clasificar la mayor cantidad de información contenida en la base de datos. Esto permite estar incorporando nueva información desde la base de datos, en forma de átomos, a la gramática general, lo cual enriquecerá el conjunto de reglas que se vayan generando. Esto es necesario ya que las reglas generadas pueden estar incompletas, es decir, estas pueden cubrir solo una pequeña fracción de las posibles combinaciones de los distintos átomos que puede proveer una base de datos. Por otro lado, aunque también se pueden generar reglas sin sentido, la Programación Genética las elimina. El Sistema de Minería de Datos descubrirá un conjunto de reglas validas, suficientes y comprensibles. Es decir, las reglas clasificadoras generadas por nuestro Sistema de Minería de Datos:

1. Tienen un alto grado de exactitud, es decir, la probabilidad de que los patrones generados se encuentren frecuentemente en la base de datos debe ser alta. Si la exactitud es 1, significa que la regla se encuentra en todos los registros de la Base de Datos. Si la exactitud es un valor cercano a 1, la regla es una regla fuerte. Si la exactitud de la regla no es muy alta, entonces la regla es una regla débil. Nuestro Sistema de Minería de Datos no debe descubrir solo las reglas fuertes, porque las reglas débiles pueden suministrar también conocimiento útil [9].
2. Son comprensibles por el usuario. La comprensibilidad de las reglas es un concepto muy subjetivo. Generalmente, la comprensibilidad de las reglas esta asociado con la simplicidad sintáctica de la misma, que consiste en el número de elementos que componen cada uno de los individuos [8].

3. Identifican relaciones entre los átomos de la base de datos que no se conocían previamente y que están presentes en los registros.

3.2 Estructura Gramatical genérica para construir Reglas

Para construir las reglas clasificadoras es necesario definir la gramática general. Este es un punto critico de nuestro Sistema de Minería de Datos, ya que las reglas generadas por la Programación Genética dependerán de ella. La población inicial que utiliza la Programación Genética para ser evolucionada es creada aleatoriamente usando la gramática general establecida. El uso de la gramática suministra una poderosa representación del conocimiento y concede gran flexibilidad para la construcción de las reglas la regla. La gramática se comporta como la plantilla bajo la cual se establecerán los patrones posibles sobre la información almacenada en la base de datos, permitiendo una combinación natural de los átomos para la construcción de los patrones, las cuales siguen la estructura genérica establecida en el conjunto de producciones de la gramática [10].

Un modelo de gramática general para construir un conjunto de reglas clasificadoras es:

Operador ⟶ Y | O | NO

Atributo_Nombre ⟶ Nombre del campo del átomo extraído de la base de datos sobre la cual se está realizando la Minería de Datos.

Atributo_Nombre: Atributo_Valor ⟶ Valor aleatorio generado desde el dominio de ese átomo en la base de datos | Valor de ese átomo tomado de la base de datos.

Atributo ⟶ Atributo_Nombre:Atributo_Valor

Causa ⟶ Atributo(Operador Atributo)*

Consecuencia ⟶ Atributo.

Regla ⟶ SI **Causa** ENTONCES **Consecuencia.**

Las partes CAUSA y CONSECUENCIA están formadas por atributos, dando lugar a los siguientes dos tipos de reglas:

- Si **(atributo)** Entonces **(atributo)**
- Si **(atributo 1) [y / o] (atributo 2)...[y /o] (atributo n)** Entonces **(atributo).**

El modelo de gramática general presentado anteriormente es el ideal para cualquier tipo de problemas. Ahora, en nuestro caso de determinación de reglas de clasificación, en las cuales se están construyendo patrones basado en los registros que se encuentran en la Base de Datos, el papel del consecuente no es relevante. El mismo si sería importante en un Sistema de Minería de Datos en el cual se quisiera verificar el cumplimiento de las relaciones entre los valores de los campos de una base de datos, y como estas generan la ocurrencia de un valor en un otro campo. Por lo tanto, nosotros les hicimos modificaciones a la gramática general, para adaptarla a nuestro problema:

Operador ⟶ Y | O

Atributo_Nombre ⟶ Nombre del campo del átomo extraído de la base de datos sobre la cual se esta realizando la Minería de Datos.

Atributo_Nombre: Atributo_Valor ⟶ Valor aleatorio generado desde el dominio de ese átomo en la base de datos | Valor de ese átomo tomado de la base de datos.

Atributo ⟶ Atributo_Nombre: Atributo_Valor

Causa ⟶ Atributo(Operador Atributo)*

Regla Clasificadora ⟶ Causa.

3.3 Componentes de la Programación Genética para nuestro Sistema

– **Función aptitud:** La función aptitud nos permite evaluar cómo funciona cada individuo (regla) de la población en el problema. Cuando usamos una regla, la idea es caracterizar a un grupo de datos bajo un patrón. Así, la función aptitud usada en este Sistema combina dos indicadores, llamados Exactitud Predictiva (EP) y Simplicidad (Sim).

En el caso de la Exactitud Predictiva de la regla (EP), usando los atributos de la causa de la regla se busca en la base de datos los registros donde se encuentra estos términos. Se van contando los registros que los contienen, y luego, este valor se divide entre el número de registros de la base de datos:

$$EP = \frac{No.RegistrosActivadosBD}{No.RegistrosBD}$$

Por otro lado, la Programación Genética no necesariamente produce reglas simples. Considerando que la comprensibilidad de una regla es inversamente proporcional a su tamaño, tenemos que hacer que la Programación Genética produzca reglas lo más cortas posibles. Por lo tanto, definimos una segunda función que mide la simplicidad de una regla (Sim), dada por [8]:

$$Sim = \frac{max\_eltos\_regla - 0.5 * num\_eltos\_regla - 0.5}{max\_eltos\_regla - 1}$$

Donde num_elementos es el número exacto de elementos en la regla, y max_elementos es el número máximo de elementos permitido en una regla. La ecuación produce su valor máximo en 1.0 cuando una regla es tan simple que su contenido es solo un término. El valor de la ecuación decrece hasta un valor mínimo de 0.5, que es producido cuando el número de nodos es igual al máximo permitido. La razón por la que Sim baja hasta 0.5 es para penalizar individuos de gran tamaño sin ser forzados a desaparecer. Esto es importante para los individuos de las primeras generaciones, cuando muchos de ellos tienen muy baja Exactitud Predictiva (EP), pero pueden llevar buen material genético capaz de producir mejores individuos al aplicarles los operadores genéticos [8].

Finalmente, el objetivo de la Programación Genética, es maximizar la Exactitud Predictiva y minimizar simultáneamente el tamaño de la regla (maximizar el valor de Sim). Así, la función aptitud usada por nuestro Sistema esta definida como:

Función Aptitud = (EP x Sim)

– **Estructura del individuo:** Un individuo es una regla clasificadora, por consiguiente, su estructura tipo árbol describe una posible regla, la cual está formada por un conjunto de átomos extraídos de la base de datos que constituyen los terminales, y las funciones Y | O que son utilizadas para formar las relaciones entre los terminales

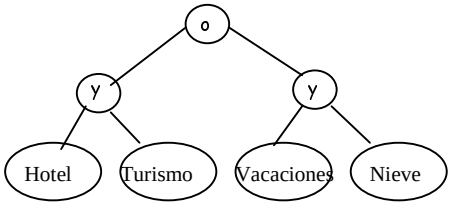

Figura 2. Estructura de un Individuo. (regla clasificadora: ((HOTEL Y TURISMO) O (VACACIONES Y NIEVE)))

– **Conjunto de terminales y funciones:** El conjunto de terminales más el conjunto de funciones deben ser capaces de expresar la solución del problema. Por otro lado, las funciones debe aceptar como argumento cualquier valor de la base de datos que pueda ser usado como terminal, tal como fue definido en la gramática general. El conjunto Terminal (átomos) consiste de atributos o palabras claves seleccionadas de la Base de Datos existente. El conjunto de Funciones consiste de las funciones, llamadas Y y O, de las reglas de producción de la gramática general. Por lo tanto, un individuo (regla) consiste de una combinación de las funciones aplicadas a los átomos.

– **Tamaño de la población:** En nuestro sistema, su rango puede estar entre 0 y 99.999 [6]. Tomar un valor elevado para el tamaño de la población sería ideal ya que nos garantizaría una enorme diversidad de reglas, pero debemos tomar en cuenta el costo computacional.

– **Número de generaciones:** En nuestro sistema, su rango puede estar entre 1 y 32.000 [6]. Mientras mayor sea el número de generaciones nuestro Sistema podría evolucionar por mucho más tiempo, y así, podría encontrar reglas mejores, pero se puede producir que después de cierto número de generaciones el conjunto de reglas no mejore.

– **Operadores de la programación genética:** En nuestro Sistema se usan tres operadores genéticos:

1. **Cruce:** produce dos hijo de dos padres. Los padres son escogidos por medio del valor de la función aptitud. Una parte del primer padre es seleccionada y reemplazado por una parte compatible del segundo padre para obtener un hijo. Este mismo procedimiento se realiza para obtener el otro hijo, pero partiendo del segundo padre.

2. **Mutación:** Es una operación que se realiza sobre un individuo. Una parte de la regla es seleccionada y reemplazada por una parte generada aleatoriamente. Para generar la parte aleatoria se utiliza la gramática general, de esta manera, la parte seleccionada puede solo mutar a otra parte con una estructura compatible a la gramática establecida.

3. **Copia:** Selecciona una regla de acuerdo al valor de la función aptitud y la traslada a la generación siguiente sin realizar ningún cambio en su estructura.

– **Criterio del término de la ejecución:** Se establecerá un número máximo de generaciones para evolucionar las reglas, al cumplirse esta, el conjunto de reglas obtenidas en la ultima generación será la solución del problema.

4 Experimento

4.1 Definición del caso de estudio.

Internet es un gran deposito de información que crece constantemente. En ella existe una infinidad de sitios que necesitan ser visitados y clasificados a la hora de hacer una búsqueda. Existen y son muy conocidas las poderosas herramientas de búsqueda que tratan de encontrar información por categoría o por contenido, tales como Altavista, Yahoo, Google, etc. A estos buscadores se introducen palabras claves y determinan las paginas o sitios web que contengan estas palabras, tratando de satisfacer los requerimientos del usuario. Muchas veces estas consultas traen resultados inconsistentes y documentos que cumplen con el criterio de búsqueda, pero no con el interés del usuario. Por esta razón, existe un área en computación denominada Minería en la Web (Web Mining), que puede definirse como la aplicación de técnicas de Minería de Datos en Internet para el descubrimiento y análisis de páginas web, y luego extraer patrones de ellas para clasificar la información.

El Web Mining ofrece ventajas para los usuarios que buscan un tema o producto especifico y para las personas que diseñan las páginas. Para los usuarios, porque simplifica y clasifica la información que se les presenta. Para los que diseñan las páginas, porque al analizar los patrones obtenidos de los links de las páginas, se puede entender el comportamiento y los requerimientos de los navegantes, y así de esta manera, mejorar el diseño de los links de sus páginas. Por esta razón, se desea usar nuestro Sistema para obtener patrones o reglas clasificadoras desde las páginas web recopiladas por algunos buscadores (Yahoo, Google, etc.) en una base de datos. Las reglas clasificadoras permiten clasificar y categorizar la información.

4.2 Simulaciones.

Para la Construcción del Sistema se utilizó el Paquete “Estudio de Programación Genética” [6]. Para realizar la parte experimental se hizo una búsqueda a través de Google de las palabras Mérida y Venezuela, obteniéndose como resultado 8.640 registros. Con estos se construyó un archivo de texto que contiene por cada registro:

1. Titulo de la pagina.
2. Dirección.
3. Descripción.

Se realizo una limpieza de los datos eliminando registros repetidos y direcciones de páginas web inconsistentes. Con el archivo de texto resultante se construirán reglas gramaticales utilizando la Programación Genética. Los átomos fueron tomados a partir de las palabras que más se repetían dentro de la base de datos.

Los parámetros de la Programación Genética utilizados para la extracción de las reglas clasificadoras son:

1. Número de átomos {4, 6, 8, 10, 12, 14}.

2. Número de individuos {10, 20, 30, 40, 50, 80, 100, 200, 300}
3. Número de generaciones {200, 400, 500, 600, 800, 1000}
4. Probabilidad de cruce {0.9}
5. Probabilidad de copia {0.1}
6. Probabilidad de mutación {0.01}
7. Número máximo de elementos en un individuo = {20, 50}

No se variaron los valores de la probabilidad de cruce, copia y mutación, porque en pruebas realizadas tomando distintos valores de estos parámetros no se obtuvieron cambios significativos en los resultados de las simulaciones realizadas. Por cuestiones de espacio, a continuación solo se presentaran los resultados más resaltantes, el resto de los resultados para diferentes casos de estudio (numero de átomos, etc.) están en [10].

4.1.1 Pruebas usando 10 átomos

Para estas pruebas iniciales, se tomaron los terminales CONGRESOS, TELEFERICO, TURISMO, HOTEL, VACACIONES, ESTUDIANTES, MONTAÑAS, NIEVE, UNIVERSIDAD, ANDES, para generar las reglas clasificadoras. El criterio de fin de la ejecución fue 500 generaciones. Con estos datos se obtuvieron los siguientes resultados:

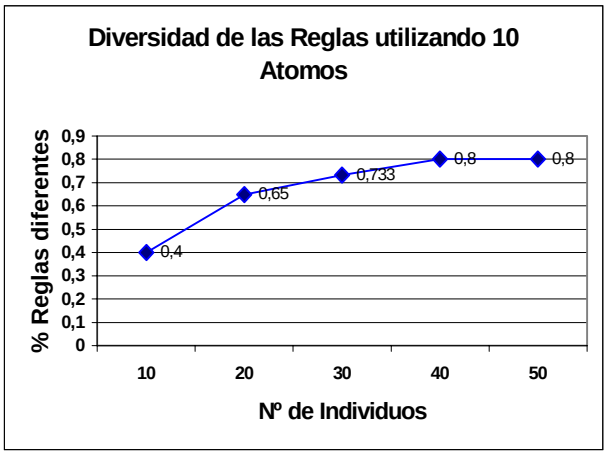

Figura 1. Diversidad de las Reglas utilizando 10 Atomos.

En la figura 1 podemos ver como al aumentar el número de individuos crece la diversidad de las reglas, pasando de 0.4 reglas diferentes para 10 individuos a 0.8 reglas diferentes para 40 individuos. Esto se debe a que cuando se aumenta el número de individuos, la Programación Genética tiene un mayor espacio de búsqueda sobre el cual aplicar los operadores genéticos, ocasionando que se produzcan más individuos diferentes.

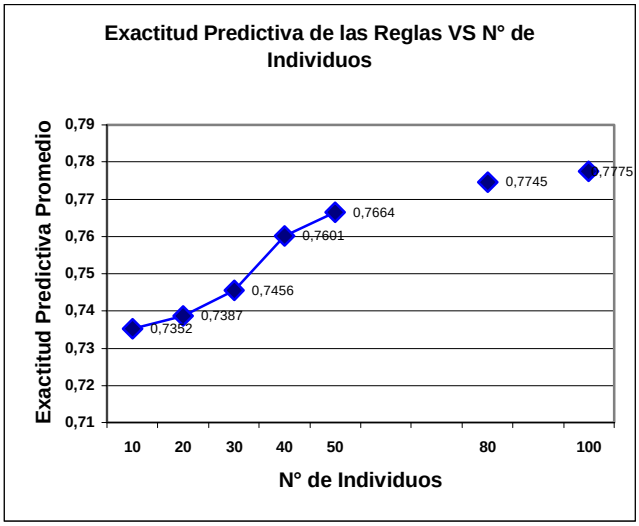

Figura 2. Exactitud Predictiva Promedio de las Reglas VS Nº de Individuos.

En la figura 2 se observa que cuando se aumenta el número de individuos la Exactitud Predictiva Promedio crece desde 0.7352 para 10 individuos hasta llegar a 0,7775 en 100 individuos. Vemos como la Exactitud Predictiva va mejorando, lo cual quiere decir que cuando se aumenta el número de individuos las reglas clasificadoras que se producen tienen la capacidad de cubrir una mayor cantidad de registros de la base de datos. Como se dijo antes, esto se debe a que cuando se aumenta el número de individuos la exploración del espacio combinatorio de los átomos se hace más eficiente.

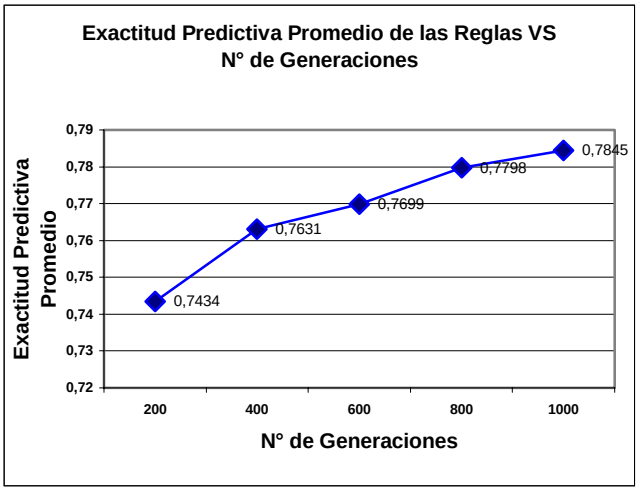

Figura 3. Exactitud Predictiva Promedio de las Reglas VS Nº de Generaciones

En la figura 3 se ve que la Exactitud Predictiva Promedio de las Reglas va de 0,7434 en 200 generaciones hasta 0,7845 en 1000 generaciones. Así, las reglas clasificadoras van evolucionando para ir mejorando levemente el valor de la Exactitud Predictiva.

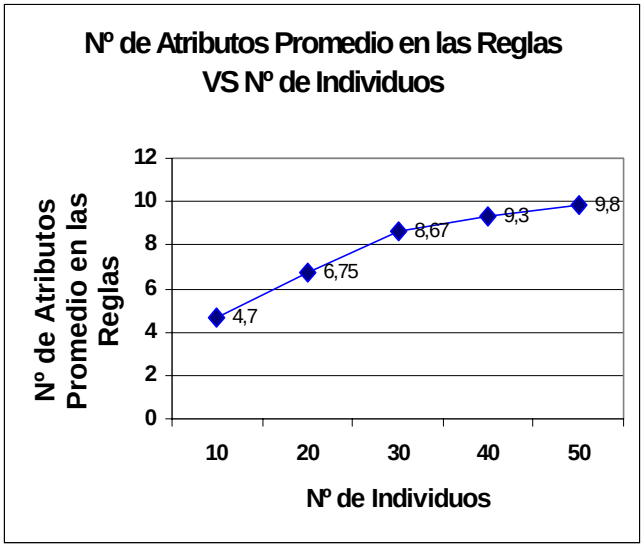

Figura 4. Nº de Atributos Promedio de las Reglas VS Nº de Individuos.

La figura 4 indica que a medida que se aumenta la cantidad de individuos en la Programación Genética las reglas clasificadoras que se obtienen se hacen más complejas. Estás pasan de tener en promedio 4,7 atributos para 10 individuos a 9,8 atributos para 50 individuos.

REGLA	**EP**	**Sim**	**EP x Sim**
(((UNIVERSIDAD Y ANDES) O MONTAÑAS) O CONGRESOS)))	0,7607	0,84	0,6389
((UNIVERSIDAD Y ESTUDIANTES) O (CONGRESOS Y ANDES))	0,7514	0,84	0,6311
((TELEFERICO Y CONGRESOS) O (UNIVERSIDAD Y ANDES))	0,7422	0,84	0,6234
((ANDES Y ESTUDIANTES) O (VACACIONES Y TURISMO))	0,7393	0,84	0,6210
(((UNIVIVESIDAD Y ANDES) O (ESTUDIANTES Y CONGRESOS))O(TURISMO Y MONTAÑAS))	0,7961	0,78	0,6209
(((UNIVERSIDAD Y ANDES) O (ESTUDIANTES Y UNIVERSIDAD))O(TURISMO Y NIEVE))	0,7885	0,78	0,6150
((UNIVERSIDAD Y NIEVE) O (HOTEL Y VACACIONES))	0,7223	0,84	0,6067
((((MONTAÑAS O (TURISMO Y ANDES)) O (UNIVERSIDAD Y ANDES)) O (CONGRESOS Y ESTUDIANTES))	0,8006	0,68	0,5444
(((TURISMO O (NIEVE Y MONTAÑAS)) O (UNIVERSIDAD Y CONGRESOS)) O (ESTUDIANTES Y VACACIONES))	0,7719	0,68	0,5248

Tabla 1. Exactitud Predictiva y Simplicidad de las mejores Reglas utilizando 10 Atomos

Como podemos observar en la Tabla 1, las reglas producidas por nuestro Sistema clasifican más del 70% de los registros de la base de datos, llegando incluso algunas de ellas a clasificar al 80%, dejando solo 20% de los registros sin clasificar. Por otro lado, la mayoría de las reglas tienen pocos elementos, con lo cual se cumple el criterio de Simplicidad. Particularmente, las reglas con valor de Exactitud Predictiva más altas poseen bastantes palabras, lo que nos indica que el número de palabras es muy importante. Así, algo a resaltar es que estas reglas contienen las palabras más representativas que aparecen en las paginas web con información sobre el Estado Mérida, para un 80% de los casos.

Por otro lado, las reglas conseguidas con alta eficiencia pueden ser divididas en varias subreglas, aportando cada una de ellas un porcentaje al valor de la Exactitud Predictiva dado a la regla original. Por ejemplo, en la tabla 1, la primera regla clasificadora con alta Exactitud Predictiva es:

(((UNIVERSIDAD Y ESTUDIANTES) O MONTAÑAS) O CONGRESOS)).

Esta regla puede ser dividida en:

UNIVERSIDADY ESTUDIANTES
MONTAÑAS
CONGRESOS

Cada una de las subreglas describe un agrupamiento posible de los datos.

4.1.2 Comparación con otros trabajos

A continuación presentamos una serie de resultados para un conjunto de pruebas que se hicieron con la finalidad de buscar cual es el valor máximo de la Exactitud Predictiva (EP) que se podía obtener con nuestro Sistema de Minería de Datos, y poderlos así comparar con otros trabajos.

Nº de Atomos	N° de Generaciones	Nº de Individuos	Regla	EP	Tiempo Ejecución (Hrs)
12	500	200	(ESTUDIANTES Y UNIVERSIDAD)O(HOTEL Y VIAJES) O (CONGRESOS Y ESTUDIANTES) O (CONGRESOS Y ALOJAMIENTO) O (TURISMO Y HOTEL) O (ESTUDIANTES YALOJAMIENTO) O (TELEFERICO Y MONTAÑAS) O (UNIVERSIDAD Y ANDES)	0.8262	1.5
14	500	300	(HOTEL Y ALOJAMIENTO) O (CONFERENCIAS Y CONGRESOS) O (CONFERENCIAS Y ESTUDIANTES) O (RESTAURANT Y VACACIONES) O (VACACIONES Y TURISMO) O (NIEVE Y TELEFERICO) O (UNIVERSIDAD Y ANDES) O (VIAJES Y TURISMO) O(ESTUDIANTES Y CONGRESOS) O (TURISMO Y MONTAÑAS)O(NIEVE Y TURISMO)	0.8806	2.5

Tabla 2. Exactitud Predictiva para algunos casos particulares.

En la Tabla 2 se observa que se obtuvo una regla clasificadora con un valor de Exactitud Predictiva (EP) de 0.8860, es decir, la regla descubierta clasifica correctamente 311 registros de los 351 contenidos en la base de datos. Al comparar nuestro Sistema de Minería de Datos con [8], en el cual el mejor valor de Exactitud Predictiva (EP) que obtuvieron fue de 0.875 para una base de datos de 48 registros, vemos que nuestro sistema logra mejorar dichos resultados. Para estos casos, el tiempo de ejecución de nuestro Sistema de Minería de Datos no es corto para obtener estos resultados y las reglas obtenidas contienen muchos elementos. La razón de lo primero se debe a la forma de búsqueda de las palabras claves en el archivo texto donde se guarda la información conseguida por las herramientas de búsqueda, la cual se realiza secuencialmente.

5 Conclusiones

Antes que nada, es bueno recordar que la Minería de Datos es una herramienta explorativa y no explicativa, es decir, explora los datos para sugerir hipótesis. Los resultados nuestros van en esa línea. Particularmente, el Sistema de Minería de Datos desarrollado en este trabajo permite extraer patrones desde una base de datos y ser representados como reglas clasificadoras de la información. Para eso, el Sistema utiliza una representación explícita del conocimiento (gramática general), logrando descubrir relaciones que no podrían ser encontradas por un humano.

Los resultados son muy prometedores, ya que el sistema descubre reglas comprensibles y logra resultados consistentes. Aunque es bueno resaltar que la eficiencia de nuestro Sistema de Minería de Datos depende en gran parte de los átomos introducidos. Por otro lado, las reglas conseguidas pueden ser divididas en varias subreglas. Cada una de las subreglas representa un agrupamiento posible de los datos.

Entre las posibles extensiones del Sistema se pueden mencionar:

1. El Sistema de Minería de Datos actúa sobre datos previamente recolectados en una base de datos, es decir, los datos no están cambiando mientras están siendo analizados. Los resultados generados son confiables para ese conjunto de datos. En la actualidad, la información contenida en entornos como Internet cambian constantemente, por lo cual, nuestro Sistema debería adaptarse a estos cambios de la información y generar patrones o reglas clasificadoras que estén adaptándose continuamente. El macroalgoritmo de la sección 3.1 puede ajustarse a esta situación.

2. El sistema podría ser usado para encontrar un conjunto de reglas que describan propiedades de los datos guardados en una base de datos. La gramática general propuesta en la sección 3 fue hecha para realizar esta tarea. Esta modalidad de exploración de datos, permitiría encontrar relaciones del tipo:

Si Compra: Pan y Compra: Salchichas

Entonces Compra: Mostaza

Si Persona: Joven y Realiza: Deporte

Entonces Practica: Fútbol.

Es decir, serian reglas que dirían que cuando se satisface la CAUSA de la regla, para un conjunto determinado de atributos con valores específicos, generarían como CONSECUENCIA una situación dada. Dichas reglas deberían ser analizadas para tomar decisiones más precisas que sean de utilidad dentro del dominio del problema. Por ejemplo, una posible conclusión ante la regla anterior sería poner en oferta las salchichas y los panes, pero subir el precio de la mostaza.

Estas son algunas de las posibles aplicaciones que pueden ser realizadas en trabajos futuros.

Referencias

[1] Aguilar, J. and Rivas, F. (Ed.), *Introducción a la Computación Inteligente*, Meritec, Mérida–Venezuela, 2001.

[2] Aguilar, J. *Compiler's Course*, Houston University, USA, Summer 2000.

[3] Kinnear, K. (Ed.), *Advances in Genetic Programming*" The MIT Press, 1994.

[4] Koza, J. *Genetic Programming: On the programming of Computers by Means of Natural Selection*, The MIT Press, 1992.

[5] Koza, J. *Genetic Programming II: Automatic Discovery of Reusable Programs*, The MIT Press, 1994.

[6] *Manual de Usuario. Estudio de Programación Genética*, Universidad de Córdoba, España, 1998.

[7] Kovalerchuk, B. Vityaev, E. and Ruiz, J. Consistent Knowledge Discovery in Medical Diagnosis, *IEEE Engineering in Medicine and Biology*, Vol. 4, (July 2000), pp. 26 – 37.

[8] Bojarczuk, C. Lopes, H. Freitas, A. Genetic Programming for Knowledge Discovery in Chest – Pain Diagnosis, *IEEE Engineering in Medicine and Biology*. Vol. 4, (July 2000), pp. 38 – 44.

[9] Wong, M. Lam ,W. Kwong, L. Ngan, P. Cheng, J. Discovery Knowledge from Medical Databases Using Evolutionary Algorithms, *IEEE Engineering in Medicine and Biology*, Vol. 4, (July 2000), pp. 45 – 55.

[10] Altamiranda, J. Aguilar, J. *Introducción a la Programación Genética*, Technical Report 20-02, CEMISID, Universidad de los Andes, Mérida – Venezuela, Julio 2001.