Universidad Central “Marta Abreu” de las Villas.
Facultad Matemática Física y Computación
Ingeniería Informática

Trabajo de Diploma

Desarrollo e implementación de un modelo híbrido de LDS y Algoritmos Genéticos para la obtención de resúmenes lingüísticos en datos de *creep*

Autor: Amaury González Muro.
Tutor: MSc. Carlos Alberto Donis Díaz

Santa Clara, 2012
“Año 54 de la Revolución”

DICTAMEN

Hago constar que el presente trabajo fue realizado en la Universidad Central Marta Abreu de Las Villas como parte de la culminación de los estudios de la especialidad de Ingeniería Informática, autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

_____

Firma del autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del tutor Firma del jefe del Seminario

"...el que tenga una canción tendrá tormenta,
el que tenga compañía soledad,
el que siga buen camino tendrá sillas,
peligrosas que lo inviten a parar.
Pero vale la canción buena tormenta,
y la compañía vale soledad,
siempre vale la agonía de la prisa,
aunque se llene de sillas, la verdad."

Silvio Rodríguez *(Historia de las sillas)*

A mi madre, por la luz,

A mi hermana Anabel, tripulante en mis sueños…

AGRADECIMIENTOS

A mi familia que resiste.

A mi tutor Carlos Donis Díaz por las enseñanzas.

A Pedro O'Reilly por la hermandad.

A Mario Pupo por la bondad y la constancia.

Gracias por el cariño a Greta, Elizabeth y Lesly.

Al cuarto 415 del U3, por la comprensión.

Al Rolo y los de la Caña Santa.

A todos los que me dejaron intentar la ilusión.

RESUMEN

La Sumarización Lingüística de Datos (LDS por sus siglas en inglés de *Linguistic Data Summarization*) es una técnica de reciente desarrollo cuya utilización se ha incrementado en los últimos tiempos en correspondencia con el desarrollo y la demanda de los Sistemas de Apoyo a la Toma de Decisiones guiados por datos. Esta técnica posee limitantes relacionadas con el proceso de búsqueda de proposiciones lingüísticas que realiza. En tal sentido se ha propuesto en la literatura el empleo de meta heurísticas de búsqueda no exhaustivas que permitan hacer más eficiente este proceso.

En el presente trabajo se propone el desarrollo e implementación de un modelo híbrido de Algoritmos Genéticos (AG) con LDS para la obtención de resúmenes lingüísticos sobre datos de *creep*, propiedad mecánica de los aceros utilizada en el diseño de nuevas aleaciones. Como parte del proceso de diseño se analizan todos los aspectos del AG y se definen las mejores variantes de tres operadores genéticos incluyendo dos nuevos que son propuestos para resolver problemas específicos en la problemática abordada. Como parte de la investigación se desarrolló la implementación de una herramienta automatizada que utiliza plataformas de desarrollo existentes relacionadas con la temática abordada. Como resultado se obtiene un modelo híbrido final que resulta escalable y permite obtener soluciones de buena calidad.

ABSTRACT

The Linguistic Data Summarization is a technique of development which has been increasingly implemented in recent times, in accordance with the development and demand of the Decision Support Systems (DSS) driven by data. The limitations of this technique consist on the searching process of linguistics sentences that it carries out. In this sense, the employment of a heuristic approach of non-exhaustive search that allows a more efficient process has been proposed in the reviewed literature.

In this thesis, it is proposed the development and implementation of a hybrid model of Genetic Algorithm with LDS to obtain linguistic summaries about creep data, which is data related to the mechanic property of the steels used in the design of new metal alloys. As part of the design process, all the aspects of the Genetic Algorithm are analyzed, and the best variants of genetic operators are defined, including two new proposed operators to solve specific problems on the discussed topic. The implementation of an automatized tool that employs existing development platform related to the studied issue was also developed. As the result, a final hybrid model that is scalable and allows obtaining good quality solutions is obtained.

ÍNDICE

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En la actualidad existe una gran cantidad de datos incomprensibles para el ser humano generados por instrumentos de medición y equipos de cómputo que desconocen un modelo lingüístico natural. En la mayoría de los casos resulta muy deseable poder expresar estos datos en una forma humanamente comprensible utilizando el lenguaje natural, siendo más relevante cuando dichos datos se relacionan con procesos de asistencia a sistemas de toma de decisiones.

En este sentido se han desarrollado técnicas que intentan expresar el conocimiento de una forma consistente con la naturaleza del ser humano en cuanto a su expresión lingüística. Entre ellas se encuentra la Sumarización Lingüística de Datos (LDS, acrónimo del inglés *Linguistic Data Summarization*), técnica que por sus características ha sido enmarcada por algunos autores dentro del campo de los sistemas de toma de decisiones asistidos por datos. LDS intenta proporcionar conocimiento en forma de un conjunto de sentencias o proposiciones en lenguaje natural. Su objetivo es obtener un texto que describa aspectos relevantes de los datos en un contexto específico mediante la utilización del lenguaje natural propio de un experto humano en el dominio del tema.

Lo anterior resulta de suma importancia debido al gran volumen de datos almacenados en la actualidad y que están accesibles no solo para compañías y organizaciones con propósitos gerenciales, sino también para ciudadanos comunes interesados en contar con información concisa y transparente sobre la evolución y comportamiento de sus datos. También en los variados campos de investigación de las diferentes ciencias, donde se necesita el procesamiento de datos numéricos para desarrollar pronósticos, estimar comportamientos, etc., resulta interesante mostrar los resultados en un formato semánticamente comprensible por los propios investigadores o posibles personas beneficiarias de los mismos. Ejemplos de ello lo constituyen los pronósticos meteorológicos, de contaminación ambiental, los diagnósticos médicos, los análisis de comportamiento de variables en mediciones científicas, entre otros. Sin dudas, LDS posee un enorme potencial tanto para organizaciones y ciudadanos comunes como para científicos e investigadores en la obtención automática y periódica de informes comprensibles semánticamente, incluso sin necesidad de convertirse en expertos de representación del conocimiento.

La esencia de LDS es que una base de datos (por ejemplo sobre empleados) con datos numéricos en sus atributos (Ej. edad, salario, calificación, etc.) puede ser resumida lingüísticamente con respecto a uno o varios atributos por medio de proposiciones lingüísticamente cuantificadas como por ejemplo: *"La mayoría de los empleados jóvenes y bien calificados son bien pagados"*

Es importante especificar que en la literatura el tratamiento del término *resumen lingüístico* puede confundir pues refiere indistintamente a una o varias proposiciones lingüísticas. Por ejemplo, para hacer referencia a todo el conjunto de proposiciones obtenidas por la técnica se utiliza el término *"**el** resumen lingüístico"* mientras que para referir una proposición específica dentro del conjunto se emplea *"**un** resumen lingüístico"*. Para evitar esta situación, en el presente documento se utiliza el término resumen lingüístico para referir a todo el conjunto de proposiciones obtenidas por la técnica, mientras que se emplea el término *"sentencia"* o *"proposición"* para referir una proposición específica dentro de todo el conjunto.

La técnica LDS ha sido enfocada desde el punto de vista de la lógica difusa donde los atributos del problema son modelados a partir de variables difusas conteniendo cada una varios términos lingüísticos. Como se ejemplificó anteriormente y sin plantear una definición rigurosa, un resumen lingüístico es un conjunto de proposiciones lingüísticamente cuantificadas, cada una de ellas estructurada por la concatenación (mediante operadores difusos) de predicados difusos (variable difusa con un término lingüístico, ejemplo: "*salario alto*"). Luego, la implementación de LDS debe ser capaz de:

- generar las posibles proposiciones existentes en un conjunto de datos y que se obtienen a partir de todas las posibles combinaciones de predicados difusos, operadores difusos y términos lingüísticos del cuantificador,

- evaluar la calidad de cada proposición obtenida, teniendo en cuenta que solo resultan de interés aquellas que muestren un grado de validez interesante.

La generación del resumen lingüístico resulta un proceso costoso debido a la búsqueda exhaustiva que es necesario realizar dentro de un espacio de soluciones (formado por todas las posibles proposiciones) que normalmente resulta extenso para base de datos regulares. Este aspecto, claramente identificado en la literatura, constituye un punto de atención y desarrollo

dentro de la técnica. El mismo ha inducido la implementación de mecanismos de interfaz con los usuarios con el objetivo de contraer el espacio de búsqueda mediante la especificación de la estructura y contenido de los resúmenes a obtener. Lo anterior presenta el inconveniente de que reduce la posibilidad de encontrar información interesante, útil y no prevista, disminuyendo así, la efectividad de la técnica.

Esta situación ha presupuesto la necesidad de vincular LDS con técnicas de búsqueda no exhaustivas que sean capaces de obtener, de manera eficiente, conjuntos de proposiciones adecuados. Tal es el caso de las técnicas evolutivas, específicamente los Algoritmos Genéticos.

Como se planteó anteriormente, LDS puede ser considerado como una técnica útil en procesos de toma de decisiones asistidos por datos. Un ejemplo de estos procesos, dentro de la Ciencia de los Materiales, lo constituye el diseño de aceros ferríticos a partir de datos de *creep*.

El *creep* es una propiedad mecánica de los aceros cuyo término en inglés se refiere a la tensión de ruptura por termofluencia: tensión a partir de la cual la estructura de un acero puede ceder al estar sometido a altas temperaturas de trabajo durante prolongados períodos de servicio. Por la importancia que tienen los aceros resistentes al *creep* en industrias como la energética y petroquímica, esta propiedad ha sido ampliamente estudiada. Durante años se han desarrollado varios modelos empíricos y semi-empíricos que permiten estimar el *creep* destacando entre ellos un modelo de redes neuronales propuesto por (Brun, 1999). Recientemente fue publicada una investigación (Díaz, 2011) que prueba la capacidad de la técnica LDS para describir adecuadamente el comportamiento del *creep* respecto a determinadas variables predictoras. En dicho trabajo se proclama entonces la conveniencia de utilizar en el futuro a LDS en el proceso de diseño de nuevas aleaciones utilizando datos de *creep*.

Los aspectos descritos anteriormente conducen al planteamiento del **problema científico** de la presente investigación. El mismo se describe como la necesidad de desarrollar un modelo de LDS que permita obtener resúmenes lingüísticos de datos mediante una búsqueda eficiente del espacio de soluciones caracterizado por un volumen considerable de posibles proposiciones lingüísticas; finalmente el problema se complementa con la conveniencia de aplicar la técnica LDS sobre datos de *creep*.

Para la solución de esta problemática científica se plantea la siguiente **hipótesis de investigación**: el empleo de Algoritmos Genéticos dentro de la técnica LDS permite desarrollar un proceso de búsqueda de los resúmenes lingüísticos en datos de *creep* que resulta eficiente en cuanto a calidad de las proposiciones obtenidas y la cantidad de proposiciones revisadas.

En el trabajo se plantea un desarrollo en función de objetivos que permiten dar solución al problema científico planteado y validar la hipótesis de investigación.

Objetivo General

Desarrollar e implementar un modelo híbrido de LDS y Algoritmos Genéticos para la obtención de resúmenes lingüísticos en datos de *creep*.

Objetivos específicos

1. Establecer un marco teórico referencial que permita describir el estado actual de
    - las características de la técnica LDS y su utilización en datos de *creep*,
    - las particularidades del modelo de Algoritmos Genéticos y su empleo en LDS,
    - JGAP y XFuzzy como plataformas de implementación de modelos de Algoritmos Genéticos y elementos de la Teoría de los Conjuntos Difusos respectivamente.
2. Desarrollar un modelo híbrido de LDS y Algoritmos Genéticos que resulte eficiente (en cuanto a calidad de las proposiciones obtenidas y la cantidad de proposiciones revisadas) para la búsqueda de resúmenes lingüísticos en datos de *creep*.
3. Implementar el modelo híbrido desarrollado utilizando las plataformas de desarrollo JGAP y Xfuzzy.

Para la presentación del contenido y resultados de la investigación se utiliza el presente informe de Tesis de Pregrado cuyos objetivos se estructura de la siguiente forma:

- un primer capítulo donde se establece un marco teórico referencial desarrollado a partir de la consulta de bibliografía actualizada sobre los temas que constituyen la plataforma de la investigación,

- un segundo capítulo que presenta las características del modelo híbrido propuesto, así como los resultados y análisis de los experimentos que avalan el proceso de desarrollo y comportamiento del mismo,

- un tercer capítulo exponiendo las características de la herramienta computacional que implementa el modelo propuesto y permite su empleo por usuarios finales,

- un acápite de conclusiones y uno de recomendaciones donde se exponen la forma en que se lograron los objetivos propuestos, así como las pautas a tener en cuenta para futuras investigaciones y

- finalmente un acápite de bibliografía que relaciona los documentos consultados para el desarrollo de la presente investigación.

CAPÍTULO I. RESÚMENES LINGÜÍSTICOS DE DATOS Y ALGORITMOS GENÉTICOS. EMPLEO EN DATOS DE *CREEP*

1.1 Resúmenes lingüísticos de datos, sus protoformas. Uso en datos de creep

La técnica LDS es un proceso que intenta obtener, desde una base de datos con una cantidad considerable de casos y relaciones incomprensibles entre estos, un modelo humano con una cierta forma simple y condensada. El objetivo es obtener un texto que describa aspectos relevantes de los datos para cierto usuario en un contexto específico, usando el lenguaje como lo haría un experto humano con dominio sobre el tema. Existen varios enfoques de LDS; en esta investigación se utiliza el propuesto por Yager en (1982, 1989, 1991, 1995b, 1995a, 1996) y desarrollado con posterioridad principalmente en (Kacprzyk et al., 2000) y (Kacprzyk y Yager, 2001).

Otros enfoques, basados en filosofías diferentes pueden ser consultados en (Dubois y Prade, 1992), (Rassmussen y Yager, 1996, 1997a, 1997b, 1999) o (Bosc et al., 2002).

Como se planteó anteriormente la esencia de LDS es que una base de datos (por ejemplo sobre empleados) con datos numéricos en sus atributos (ejemplo: edad, salario, calificación, etc.) puede ser resumida lingüísticamente con respecto a uno o varios atributos por medio de proposiciones lingüísticamente cuantificadas como por ejemplo: *"La mayoría de los empleados jóvenes y bien calificados son bien pagados"*.

1.1.1 Resúmenes lingüísticos utilizando la lógica difusa con cuantificadores lingüísticos. Extensiones

La definición de resumen lingüístico emplea los conceptos de la lógica difusa los cuales pueden ser consultados en (Zadeh, 1965), (Malinowski, 1990), (Lombraña, 1996), (Ruiz, 2007) y (Gómez, 2007).

Utilizando el enfoque de Yager (1982) y su terminología original, se tiene:

- V es una cualidad (atributo) de interés, ejemplo: salario en una base de datos de empleados,
- $Y = \{y_1, \dots, y_n\}$ es un conjunto de objetos o casos que manifiestan una cualidad *V*, ejemplo: el conjunto de empleados; por tanto $V(y_i)$ representa valores de la cualidad *V* para el objeto y_i,
- $D = \{V(y_1), \dots, V(y_n)\}$ es un conjunto de datos (la base de datos en cuestión).

Un resumen lingüístico de un conjunto de datos consiste de:

- un Sumarizador (Summarizer) *S*: un atributo junto con un término lingüístico (predicado difuso), ejemplo (‘*joven*’ para el atributo ‘*edad*’),
- una Cuantificador (Quantifier) *Q*: cantidad en acuerdo, ejemplo ‘*la mayoría*’,
- grado de verdad (truth) o validez *T*: un número del intervalo [0, 1] que representa el grado de validez del resumen, ejemplo 0.7, normalmente solo son interesantes aquellos resúmenes que tengan un valor elevado de T.
- opcionalmente, un Calificador (*Qualifier*) *R*: otro atributo junto con un valor lingüístico determinando un subconjunto borroso de *Y* (ej. ‘*baja*’ para el atributo ‘*calificación*’).

De esta forma se pueden utilizar, como ejemplos de resúmenes lingüísticos, las siguientes proposiciones:

T (la mayoría de los empleados son jóvenes) = 0.7 (1)

T (la mayoría de los empleado con baja calificación son jóvenes) = 0.7 (2)

Resulta importante aclarar que los resúmenes lingüísticos en el presente trabajo se refieren únicamente a aquellos generados sobre conjuntos de valores numéricos.

1.1.1.1 Sobre la forma del sumarizador S

El sumarizador es una expresión lingüística semánticamente representado por un predicado difuso, es decir, una variable difusa junto con un término lingüístico; este último representado por un conjunto difuso. En el ejemplo planteado el sumarizador se refiere a la variable difusa ‘*edad*’ con su término lingüístico ‘*joven*’ el cual pudiera estar representado por un conjunto difuso en el universo de discurso dado por posibles valores de la edad humana como por ejemplo

{1, 2, . . ., 90) y una función de pertenencia no decreciente defina en dicho intervalo de forma tal que el caso simple de una función linear, la edad hasta 35 años es al seguro '*joven*', o sea su grado de pertenencia es igual a 1, la edad por encima de 50 años es al seguro '*no joven*', o sea, su grado de pertenencia es cero y para edades entre 35 y 50 años, los grados de pertenencia están entre 1 y cero de forma tal que mientras mayor sea la edad, menor grado de pertenencia le corresponde. Evidentemente, el significado del sumarizador o su correspondiente conjunto difuso es subjetivo y puede ser predefinido o especificado por el usuario. El sumarizador utilizado como ejemplo ('*joven*') resulta simple, en la práctica puede ser extendido de forma que confluyan varios valores de atributos como por ejemplo: '*jóvenes y bien pagados*'; así, el sumarizador puede desarrollarse hacia combinaciones más complejas.

1.1.1.2 Sobre la forma del cuantificador *Q*

El cuantificador *Q* es una propuesta sobre la extensión o cantidad de datos que satisfacen el resumen. Una propuesta precisa, no es consistente a como lo hace el ser humano y por tanto para Q se utiliza un término lingüístico representado por un conjunto difuso. Básicamente se pueden utilizar dos tipos de tales cantidades lingüísticas en acuerdo:

- absolutas: por ejemplo '*alrededor de 5*', '*más o menos 100*', '*varios*', y
- relativas: por ejemplo '*unos pocos*', '*más o menos la mitad*', '*la mayoría*', '*casi todos*', etc.

Estas expresiones lingüísticas son los llamados cuantificadores lingüísticos difusos (Zadeh, 1983, 1985) que pueden ser manipulados por la lógica difusa. Al igual que el sumarizador, la forma del cuantificador puede ser subjetiva y puede ser predefinida o especificada por el usuario.

1.1.1.3 Cálculo del grado de validez T del resumen lingüístico

El cálculo del grado de validez de resúmenes como los ejemplificados en (1) y (2) es equivalente al cálculo del grado de verdad de una proposición lingüísticamente cuantificada. Esto se puede realizar utilizando una de las siguientes dos técnicas: el cálculo de sentencias lingüísticamente cuantificadas basado en la lógica difusa propuesto por Zadeh (1983) o mediante el uso de operadores OWA propuesto por Yager (1988). A continuación será descrito el enfoque de Zadeh.

Se remite al lector a la bibliografía referenciada en el caso de los operadores OWA teniendo en cuenta que su uso no es interés en la presente investigación.

Los ejemplos (1) y (2) pueden ser representados de forma genérica respectivamente mediante las expresiones:

Qy's are S (3)

QRy's are S (4)

Como se planteó anteriormente *R* y *S* pueden ser uno o varios predicados difusos concatenados entre sí de diversas formas. El cuantificador lingüístico *Q* (proporcional, no decreciente) se asume como un conjunto difuso definido en el intervalo [0, 1], por ejemplo

$$\mu_Q(x) = \begin{cases} 1 & para\ x \geq 0.8 \\ 2x - 0.6 & para\ 0.3 < x < 0.8 \\ 0 & para\ x \leq 0.3 \end{cases}$$

entonces

$$T(Qy's\ are\ S) = \mu_Q\left[\frac{1}{n}\sum_{i=1}^{n}\mu_S(y_i)\right] \quad (5)$$

$$T(QRy's\ are\ S) = \mu_Q\left[\frac{\sum_{i=1}^{n}(\mu_R(y_i) \wedge \mu_S(y_i))}{\sum_{i=1}^{n}\mu_R(y_i)}\right] \quad (6)$$

1.1.1.4 Extensiones. Otros criterios de validez o calidad de los resúmenes

El criterio de validez de una proposición lingüísticamente cuantificada expresado según (5) y (6) es ciertamente el más importante en el marco de trabajo asumido. Sin embargo el mismo no cubre todos los aspectos de un resumen lingüístico. A partir de esta situación han sido desarrollados otros varios criterios de calidad que han sido recopilados en (Kacprzyk y Zadrożny, 2005). A continuación se describen cinco nuevos criterios introducidos y luego desarrollados en (Kacprzyk, 2000), (Kacprzyk y Strykowski, 1999), (Kacprzyk y Yager, 2001) y (Kacprzyk y colectivo de autores, 2000).

Criterios de calidad de un resumen lingüístico

- Valor de validez (T_1)
- Grado de imprecisión (T_2)
- Grado de cobertura (T_3)
- Grado de conveniencia (appropriateness) (T_4)
- Longitud del resumen (T_5)

Valor de validez (T_1)

Es el criterio básico de validez de un resumen que ha sido definido anteriormente en (5) y (6) y que se obtiene directamente del cálculo difuso de proposiciones lingüísticamente cuantificadas propuesto por Zadeh.

Grado de imprecisión (T2)

Este indicador es un criterio de validez obvio e importante. Básicamente, un resumen lingüístico como por ejemplo "En ***casi todos*** los días de invierno la ***temperatura es bastante fría***", tiene un grado de validez T_1 muy elevado y sin embargo no resulta útil debido al grado de imprecisión del sumarizador '*bastante fría*'.

Considerando que el sumarizador está dado por una familia de conjuntos borrosos $S= \{S_1, S_2, S_3, \ldots, S_m\}$, el grado de imprecisión (borrosidad) de un determinado conjunto difuso S_j, $j=1,\ldots, m$ se puede definir como

$$in(s_j) = \frac{card\{x \in X_j : \mu_{S_j}(x) > 0\}}{card\ X_j}$$

donde *card* denota la cardinalidad del correspondiente conjunto (no difuso). Mientras más "adulador" sea el conjunto difuso s_j, mayor grado de imprecisión *in* (s_j) tendrá.

Luego, el grado de imprecisión (borrosidad) de un resumen lingüístico (más bien del *sumarizador*) se define como:

$$T_2 = 1 - \sqrt[m]{\prod_{j=1,\ldots,m} in(s_j)}$$

Es importante observar que el grado de imprecisión T_2 depende solamente de la forma del resumen y no de la base de datos, o sea, su cálculo no requiere del recorrido de la base de datos (todos sus casos).

Grado de cobertura (T_3)

El grado de cobertura da una idea de la cantidad de objetos de la base de casos que cumpliendo con la restricción impuesta por *R*, son cubiertos por el resumen particular, o sea, cumplen también con el criterio del sumarizador del resumen. Se define como:

$$T_3 = \frac{\sum_{i=1}^{n} t_i}{\sum_{i=1}^{n} h_i}$$

Donde

$$t_i = \begin{cases} 1 & si\ \mu_S(y_i) > 0\ y\ \mu_R(y_i) > 0 \\ 0 & en\ otro\ caso \end{cases}$$

$$h_i = \begin{cases} 1 & si\ \mu_R(y_i) > 0 \\ 0 & en\ otro\ caso \end{cases}$$

Su interpretación es simple, por ejemplo si es igual a 0.15 significa que el 15 por ciento de los objetos son consistentes con el resumen en cuestión.

Grado de conformidad (*appropriateness*) (T_4)

Es el más relevante de los grados de validez. Considérese que el resumen formado por un sumarizador *S* con varios conjuntos difusos $S = \{S_1, S_2, S_3, \ldots, S_m\}$, se particiona en *m* resúmenes parciales cada uno conteniendo uno solo de los conjuntos difusos $S_1, S_2, S_3, \ldots, S_m$. Si se denota:

$$S_j(y_i) = \mu_{S_j}(y_i)$$

entonces

$$r_j = \frac{\sum_{i=1}^{n} h_i}{n}, \quad j = 1,\ldots,n$$

donde

$$h_i = \begin{cases} 1 & si\ S_j(y_i) > 0 \\ 0 & en\ otro\ caso \end{cases}$$

y el grado de conformidad se define como

$$T_4 = abs\left(\prod_{j=1,\ldots,m} r_j - T_3 \right)$$

El grado de conformidad expresa cuan característico resulta el resumen para la base de casos en particular. Por ejemplo, para una base de casos sobre empleados, si el 50% de ellos tienen menos de 25 años y el 50% tienen calificación elevada, se espera que el 25% de los empleados sean menores de 25 años y tengan una calificación elevada; esto se corresponde con una situación típica, completamente esperada. Sin embargo, si el grado de conveniencia es por ejemplo 0.39 (el 39% de los empleados son menores de 25 años y tienen una calificación elevada) entonces el resumen encontrado refleja una relación en los datos que resulta interesante y no esperada del todo. Este indicador es muy importante porque permite desechar resúmenes que pueden tener un grado de validez (T_1) elevado y sin embargo no aportan mucha información. Por ejemplo, un resumen trivial como "*el **100% de las ventas** fue de **cualquier artículo***" tiene un valor de validez óptimo, sin embargo el grado de conformidad es igual a cero lo cual es correcto.

Longitud del resumen (T_5)

La longitud es un elemento relevante pues un resumen extenso no es fácilmente comprensible para un usuario humano. Esta longitud T_5 puede ser definida de diferentes formas; la siguiente ha probado ser útil:

$$T_5 = 2(0.5^{\,card\ S})$$

donde *card S* es la cantidad de elementos de *S*.

Cálculo del grado de validez total del resumen (*T*)

A partir de los criterios definidos anteriormente, el grado de validez total (*T*) de un resumen lingüístico particular se define como el promedio pesado de estos cinco criterios de validez, o sea:

$$T = T(T_1, T_2, T_3, T_4, T_5; w_1, w_2, w_3, w_4, w_5) = \sum_{i=1,2,...,5} w_i T_i$$

y el problema es obtener un resumen óptimo S^*, tal que $S^* = \arg\max_S \sum_{i=1,2,...,5} w_i T_i$

donde w_1, w_2,…, w_5 son pesos asignados a cada grado de validez particular con valores definidos en el intervalo [0, 1] siendo los mayores valores los más importantes y cumpliendo que $\sum_{i=1,2,...,5} w_i = 1$.

La definición de los pesos w_1, w_2,..., w_5 es un problema en sí. Estos pudieran ser predefinidos o especificados por el usuario.

1.1.2 Resúmenes lingüísticos vía consultas difusas. Sus protoformas

En varios trabajos de Kacprzyk y Zadrożny (1998, 2001) se ha propuesto un enfoque interactivo, o sea, mediante la asistencia del usuario para la definición de los sumarizadores (indicación de los atributos y sus combinaciones). Esto procede mediante una interfaz de usuario que define un sistema de consultas difusas. Básicamente, las consultas (referidas a los sumarizadores) permitidas son:

- simples, ejemplo: '*salario es alto'*
- compuestas, ejemplo: '*salario es bajo* AND *edad es viejo*'
- compuestas (con cuantificador), ejemplo: 'la *mayoría* de {salario es *alto*, edad es *joven*, …, entrenamiento es *bien por encima del promedio*}'

Igualmente se utilizan términos lingüísticos "naturales" (definidos según el criterio 7 ± 2!) ejemplificados por: *muy bajo*, *bajo*, *medio*, *alto*, *muy alto* y también cuantificadores lingüísticos difusos "comprensibles" como: *la mayoría*, *casi todos*,…, *pocos*, *muy pocos*, etc.

Resulta obvio que las consultas difusas se corresponden con sumarizadores en los resúmenes lingüísticos. Así, la obtención de un resumen lingüístico procede en una forma interactiva (asistida por el usuario) de la siguiente forma:

- el usuario formula un conjunto de resúmenes lingüísticos de interés (relevancia) utilizando el sistema de consultas difusas descrito,
- el sistema recupera objetos de la base de casos y calcula la validez de cada resumen adoptado y
- se selecciona el resumen lingüístico más apropiado.

El uso de consultas difusas es muy relevante pues permite desarrollar el proceso de obtención de resúmenes en el contexto de un sistema de consultas difusas. Por ejemplo, (1) puede ser definido con una consulta como la siguiente:

Most records match query S (7)

Donde S refiere toda la condición expresada por el sumarizador, en este caso 'edad = *joven*'.

De manera similar (2) puede ser definido por:

Most records meeting conditions R match query S (8)

De esta forma (7) especifica solamente información sobre el subconjunto de objetos tomados en cuenta para obtener (1). En el caso de (8), *R* se corresponde con un filtro (utilizando la terminología de base de datos) y proclama que la *mayoría* de los objetos que pasan a través de *R* coinciden con *S*. En este caso, como el filtro es difuso, el objeto pasa por R con un grado entre [0, 1].

Protoformas

En este contexto, el concepto de protoforma definido por Zadeh (2002) resulta muy relevante. Una protoforma se define como un prototipo abstracto, por ejemplo, las consultas (7) y (8) son instancias específicas de las siguientes protoformas respectivas.

Most y's are S
Most Ry's are S

donde *y* refiere los objetos de la base de casos, *R* es un filtro y *S* el sumarizador.

Como las protoformas pueden formar una jerarquía, se pueden definir protoformas de más alto nivel (más abstractas); por ejemplo remplazando '*Most'* por una representación general del cuantificador lingüístico (*Q*), se obtienen respectivamente las protoformas

Qy's are S
QRy's are S

Básicamente, las formas más abstractas se corresponden con casos en los cuales se asume menos sobre los resúmenes a buscar. Existen dos casos límites donde: (1) se asume una protoforma totalmente abstracta o (2) se asume que todos los elementos de la protoforma son dados en el menor nivel de abstracción como términos lingüísticos específicos. En el primer caso, la sumarización de los datos será extremadamente costosa en cuanto a tiempo pues debe construir

todos los posibles resúmenes (con todos los componentes lingüísticos y sus combinaciones) que existan dentro de la base de casos dada y luego presentar al usuario aquellos que posean un nivel de veracidad por encima de algún umbral previamente definido; esta variante tiene al relevancia de que pudiera producir vistas más interesantes e inesperadas sobre los datos. En el segundo caso, el usuario debe especificar explícitamente todas las componentes del resumen y el sistema solo debe proceder a la evaluación del mismo. En la Tabla I-1 se muestran seis tipos básicos de resúmenes lingüísticos correspondiéndose con protoformas que van incrementando su nivel de abstracción hacia las filas inferiores de la tabla. El término $S^{estructura}$ significa que los atributos y sus conexiones en el sumarizador son conocidos, mientras que S^{valor} hace referencia al(os) término(s) lingüístico(s) a buscar en el sumarizador.

Tabla I-1. Clasificación de las protoformas de los resúmenes lingüísticos

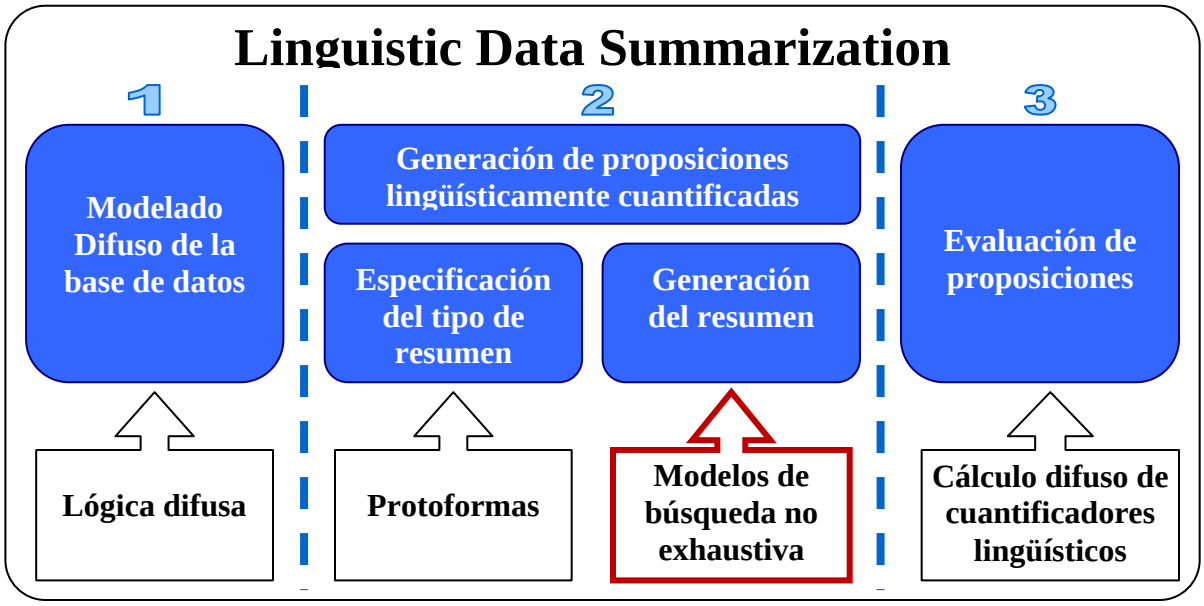
Tipo	Protoforma	Dado	Busca
0	*QRy's are S*	Todo	Validez (*T*)
1	Qy's are S	S	Q
2	*QRy's are S*	*S* y *R*	*Q*
3	*Qy's are S*	*Q* y $S^{estructura}$	S^{valor}
4	*QRy's are S*	*Q*, *R* y $S^{estructura}$	S^{valor}
5	*QRy's are S*	Nada	*S*, *R* y *Q*

La protoforma Tipo cero se corresponde con el caso planteado anteriormente donde el usuario debe especificar todos los términos del resumen y el sistema simplemente debe gestionar la evaluación del mismo. En la Tipo 1, el usuario debe especificar el sumarizador candidato y la técnica debe determinar cuál es la proporción o fracción de los casos que coincide con dicho sumarizador y cuál es el cuantificador lingüístico que mejor denota dicha proporción. La tipo 2 es una extensión directa de la Tipo 1 adicionando un filtro difuso. Los resúmenes Tipo 3 requieren un poco más esfuerzo. Su objetivo es determinar el valor típico de un atributo; el sumarizador S consiste de solo una condición formada por el atributo cuyo valor típico se busca, el operador relacional '=' y una casilla en blanco para el valor buscado. Por ejemplo, utilizando la siguiente protoforma en un contexto de datos personales: Q ='la mayoría' y S = 'edad = ?' (? denota la casilla mencionada anteriormente), el sistema debe buscar un valor típico de edad. Las

protoformas Tipo 4 poseen un comportamiento similar al descrito para el Tipo 3 pero para un subconjunto de casos obtenido luego de aplicar el filtro difuso representado por R. Por último el Tipo 5 representa la forma más general considerada donde el usuario no especifica nada sobre el proceso y el sistema de buscar todos los posibles resúmenes.

1.1.3 Proceso de obtención de resúmenes lingüísticos de datos

El proceso de obtención de resúmenes lingüísticos pude ser especificado mediante tres pasos fundamentales tal y como se muestra en el siguiente esquema.



En el primer paso se deben definir todos los elementos de la lógica difusa necesarios para caracterizar la base de casos. Específicamente es necesario definir:

- Las variables difusas utilizadas para cada atributo de la base de casos especificando los conjuntos borrosos que representan cada uno de los respectivos términos lingüísticos.
- La variable difusa (con todos los términos lingüísticos) que representará al cuantificador.

Estos elementos deben ser definidos por expertos del problema aunque han sido desarrollos enfoques donde son generados o tonificados automáticamente mediante el empleo de meta-heurísticas de búsqueda no exhaustiva como los Algoritmos Genéticos.

El segundo paso se refiere a la obtención de las proposiciones lingüísticas que conformaran el resumen lingüístico sobre los datos. Este paso posee dos momentos básicos: (1) la especificación del tipo de resumen a obtener el cual es guiado por el uso de las protoformas y (2) la generación de los resúmenes que en dependencia de la protoforma deseada puede resultar tan trivial como la simple evaluación de un resumen completamente especificado por el usuario o tan compleja como la búsqueda de todos las posibles proposiciones lingüísticas en un espacio de soluciones normalmente grande. En este último caso también se han propuesto soluciones que emplean meta-heurísticas de búsqueda como los Algoritmos Genéticos.

El tercero y último paso se refiere a la evaluación de las proposiciones obtenidas en el paso anterior. Aquí se pueden utilizar los enfoques referidos anteriormente en el epígrafe 1.1.1.3: el cálculo difuso de los cuantificadores lingüísticos propuesto por Zadeh o el empleo de los operadores OWA propuestos por Yager.

1.1.4 Empleo de LDS en datos de *creep*

El creep, propiedad mecánica de los aceros, es descrita dentro del campo de la Ciencia de los Materiales como el nivel de tensión a partir del cual se origina una ruptura de la estructura interna del acero al pasar este de su fase de elasticidad a su fase de plasticidad en la cual no se pueden recuperar las transformaciones ocurridas por el sometimiento a altas temperaturas durante prolongados tiempos de servicio. Esta propiedad resulta muy importante y ha sido ampliamente estudiada para el proceso de diseño de nuevos aceros utilizados fundamentalmente en industrias como la energética, petroquímica y aeronáutica.

En ausencia de modelos físicos que describan adecuadamente el comportamiento de esta propiedad, se han desarrollado varios modelos empíricos y semi-empíricos para su estimación en relación a un grupo de variables predictoras. Dentro de los modelos propuestos sobresalen aquellos obtenidos mediante técnicas de aprendizaje automatizado como son Procesos Gaussianos (Tancret, 2003), Máquinas de vectores de soporte para regresión (Díaz, 2009) y Redes Neuronales Artificiales (Brun, 1999), (Masuyama, 2007). El modelo desarrollado por F. Brun utilizando una Red Neural ha sido muy empleado en la práctica en varios experimentos y sus pronósticos han sido comprobados sobre aceros reales en investigaciones publicadas recientemente (Masuyama, 2007). Como complemento a estos modelos, se ha propuesto en

(Díaz, 2011) la conveniencia de utilizar LDS como medio de soporte del proceso de diseño de aceros ferríticos resistentes al *creep* una vez que el mismo pudiera revelar relaciones interesantes entre los parámetros o variables utilizadas para la estimación de esta propiedad. C.A.D. Díaz, en su investigación demostró como LDS es capaz de describir adecuadamente el comportamiento del *creep* lo cual establece las bases para su futura utilización en la exploración de los datos relacionados con la temática con vistas a la obtención de resúmenes lingüísticos que puedan aportar nuevos elementos útiles en el proceso de diseño de nuevas aleaciones.

Datos de *creep*

En las investigaciones sobre el *creep*, mencionadas anteriormente se han utilizado un conjunto de datos que han sido recopilados de la literatura existente. Consiste de 2066 casos de aceros que describen valores de 37 variables en relación con el *creep*. Entre estas variables se incluyen el tiempo transcurrido antes de sobrevenir la ruptura por *creep*, la temperatura de trabajo a que ha sido sometido el acero, varios parámetros que describen la composición química y otros referidos a los tratamientos térmicos efectuados sobre el acero. Estos datos pueden ser descargados libremente desde el sitio web MAP[1].

Estos mismos datos han sido también los utilizados en el trabajo de C.A.D. Díaz sobre la utilización de LDS en datos de *creep*.

1.2 Algoritmos Genéticos. Introducción

Los Algoritmos Genéticos (AG[2]) son métodos adaptativos que pueden usarse para resolver problemas de búsqueda y optimización. Están basados en los procesos genéticos de los organismos vivos. A lo largo de las generaciones, las poblaciones evolucionan en la naturaleza de acorde con los principios de la selección natural y la supervivencia de los más fuertes, postulados por (Darwin, 1859). Por imitación de este proceso, los Algoritmos Genéticos son capaces de ir creando soluciones para problemas del mundo real. La evolución de dichas

[1] Siglas en inglés de *Materials Algorithms Project,* proyecto desarrollado por el Laboratorio Nacional de Física y la Universidad de Cambridge, Reino Unido.
http://www.msm.cam.ac.uk/map/data/materials/creeprupt-b.html, última revisión: 10/06/2011.
[2] En el texto se utilizará AG tanto para el plural como para el singular.

soluciones hacia valores óptimos del problema depende en buena medida de una adecuada codificación de las mismas.

Los principios básicos de los Algoritmos Genéticos fueron establecidos por (Holland, 1975), y se encuentran bien descritos en varios textos: (Goldberg, 1989), (Davis, 1991), (Michalewicz, 1992) y (Reeves, 1993).

En la naturaleza los individuos de una población compiten entre sí en la búsqueda de recursos tales como comida, agua y refugio. Incluso los miembros de una misma especie compiten a menudo en la búsqueda de un compañero. Aquellos individuos que tienen más éxito en sobrevivir y en atraer compañeros tienen mayor probabilidad de generar un gran número de descendientes. Por el contrario individuos poco dotados producirán un menor número de descendientes. Esto significa que los genes de los individuos mejor adaptados se propagarán en sucesivas generaciones hacia un número de individuos creciente. La combinación de buenas características provenientes de diferentes ancestros, puede a veces producir descendientes "superindividuos", cuya adaptación es mucho mayor que la de cualquiera de sus ancestros. De esta manera, las especies evolucionan logrando unas características cada vez mejor adaptadas al entorno en el que viven.

Los Algoritmos Genéticos usan una analogía directa con el comportamiento natural. Trabajan con una población de individuos, cada uno de los cuales representa una solución factible a un problema dado. A cada individuo se le asigna un valor o puntuación, relacionado con la bondad de dicha solución. En la naturaleza esto equivaldría al grado de efectividad de un organismo para competir por unos determinados recursos. Cuanto mayor sea la adaptación de un individuo al problema, mayor será la probabilidad de que el mismo sea seleccionado para reproducirse, cruzando su material genético con otro individuo seleccionado de igual forma. Este cruce producirá nuevos individuos -descendientes de los anteriores- los cuales comparten algunas de las características de sus padres.

Cuanto menor sea la adaptación de un individuo, menor será la probabilidad de que dicho individuo sea seleccionado para la reproducción, y por tanto de que su material genético se propague en sucesivas generaciones.

De esta manera se produce una nueva población de posibles soluciones, la cual reemplaza a la anterior y verifica la interesante propiedad de que contiene una mayor proporción de buenas características en comparación con la población anterior. As a lo largo de las generaciones las buenas características se propagan a través de la población. Favoreciendo el cruce de los individuos mejor adaptados, van siendo exploradas las áreas más prometedoras del espacio de búsqueda. Si el Algoritmo Genético ha sido bien diseñado, la población convergerá hacia una solución óptima del problema.

El poder de los Algoritmos Genéticos proviene del hecho de que se trata de una técnica robusta, y pueden tratar con éxito una gran variedad de problemas provenientes de diferentes áreas, incluyendo aquellos en los que otros métodos encuentran dificultades. Si bien no se garantiza que el Algoritmo Genético encuentre la solución óptima del problema, existe evidencia empírica de que se encuentran soluciones de un nivel aceptable, en un tiempo competitivo con el resto de algoritmos de optimización combinatoria. En el caso de que existan técnicas especializadas para resolver un determinado problema, lo más probable es que superen al Algoritmo Genético, tanto en rapidez como en eficacia. El gran campo de aplicación de los AG se relaciona con aquellos problemas para los cuales no existen técnicas especializadas. Incluso en el caso en que dichas técnicas existan, y funcionen bien, pueden efectuarse mejoras de las mismas hibridándolas con los AG.

1.2.1 Algoritmos Genéticos conceptos fundamentales

Los Algoritmos Genéticos (AG) son métodos de búsqueda de propósito general basados en los principios de la genética natural, es decir, son algoritmos de búsqueda basados en los mecanismos de la selección natural y la genética.

Los Algoritmos Genéticos son un ejemplo de método que explota la búsqueda aleatoria “guiada” que ha ganado popularidad en los últimos años debido a la posibilidad de aplicarlos en una gran gama de campos y a las pocas exigencias que impone al problema.

1.2.1.1 Terminología usada

En el trabajo con AG se maneja una serie de términos "*importados*" de la genética natural. No siempre es adecuada la analogía, pero estos son comúnmente aceptados:

Población: Conjunto de individuos o cromosomas. Equivale a una muestra aleatoria del espacio de solución o un conjunto de soluciones alternativas.

Cromosoma: Un cromosoma es un portador de la información genética que transmite cada uno de sus genes. Una posible solución.

Gen: Cada uno de los rasgos o características que conforman el cromosoma. También se les llama parámetros o aspectos. Cada gen equivale a una variable del problema.

Genotipo: En biología se le llama al "paquete" genético total en su forma interna. En la terminología de AG será la información genética de todo el cromosoma en forma codificada.

Fenotipo: Se le llama en genética al paquete genético tal y como interactúa con el medio exterior. En los AG artificiales serían los aspectos del cromosoma decodificados.

Locus: Es el valor asociado a un gen.

1.2.1.2 Componentes de un AG

Los AG trabajan a partir de una población inicial de estructuras artificiales que van modificando repetidamente a través de la aplicación de los siguientes operadores genéticos:

- Operador de Selección o Darwiniano
- Operador de Cruzamiento o Mendeliano
- Operador de Mutación

Para utilizar los AG es necesario encontrar una posible estructura para representar las soluciones. Pensando este asunto como el problema de buscar en un espacio de estados, una instancia de esta estructura representa un punto o un estado en el espacio de búsqueda de todas las posibles soluciones. Así, una estructura de datos en el AG consistirá en uno o más cromosomas (frecuentemente uno), el cual se representa comúnmente como una cadena de bits (existen otras

representaciones).

Cada cromosoma (cadena) es una concatenación de un número de subcomponentes llamados genes. La posición de un gen en el cromosoma se conoce como locus y sus valores como alelos. En la representación como cadena de bits, un gen es un bit o una cadena de bits, un locus es su posición en la cadena y un alelo es su valor (0 ó 1 si es un bit).

Al optimizar una estructura usando un AG se necesita una medida de la calidad de cada estructura en el espacio de búsqueda. La función de adaptabilidad es la encargada de esta tarea. En una maximización de funciones, la función objetivo frecuentemente actúa como la función de adaptabilidad, como en el ejemplo anterior, en el cual la meta es encontrar el valor de <*x*> que maximice F.

Los AG realizan una maximización por defecto, para los problemas de minimización los valores de la función objetivo pueden ser negados y traslados con vistas a tomar valores positivos para producir así la adaptabilidad.

1.2.2 AG simple

El modo de trabajo de un AG puede resumirse en el esquema de funcionamiento de un AG simple mostrado en la Fig. I-1.

El mecanismo de un AG simple es como sigue:

- El AG simple genera aleatoriamente una población de n estructuras (cadenas, cromosomas o individuos)
- Sobre la población actúan los operadores transformando la población. Una vez completada la acción de los tres operadores se dice que ha transcurrido un ciclo generacional.
- Luego se repite el paso anterior mientras no se garantice el criterio de parada del AG.

- El *operador de selección o Darwiniano* realiza la selección de las cadenas de acuerdo a su adaptabilidad para el posterior apareamiento.
- El *operador de cruzamiento o Mendeliano* realiza la recombinación del material genético de dos cadenas padres.
- El *operador de Mutación* al estilo del operador natural realiza la mutación de un gen dentro

de un cromosoma o cadena a sus diferentes formas alelos.

- Para cada uno de estos operadores está asociado el uso de probabilidades y la generación de números aleatorios. El AG ejecuta para un número fijo de generaciones o hasta que se satisface algún criterio de parada.

Desde el punto de vista de la comparación de los AG con otros métodos de búsqueda se pueden enmarcar sus diferencias en cuatro aspectos:

1- Trabajan con una codificación de los parámetros y no con los parámetros mismos.

2- Buscan a partir de una población de puntos y no de un punto simple.

3- Usan directamente la función objetivo y no la derivada u otro conocimiento auxiliar.

4- Usan reglas de transición probabilísticas y no determinísticas

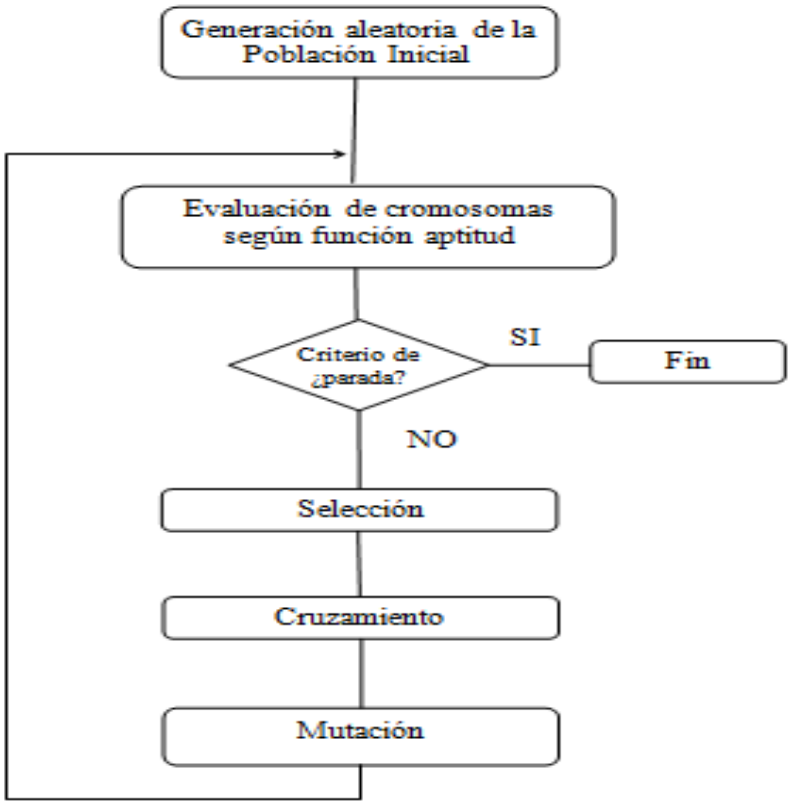


Fig. I-1. Diagrama funcional de un algoritmo genético

1.2.3 Ventajas de los AG frente a otros métodos de búsqueda

Los métodos conocidos son buenos mientras el problema no es muy complejo. Los AG permiten la solución eficiente de funciones extremadamente complejas.

Las potencialidades de los AG se pueden resumir por sus habilidades para resolver una variedad de problemas muy difíciles:

- Trabajar sin conocimiento previo de la función a optimizar.
- Optimizar funciones "ruidosas".
- Trabajar sin información secundaria como gradientes.

La mayoría de los especialistas en este tema coinciden en que los AG pueden resolver las dificultades representadas en los problemas de la vida real que a veces son insolubles por otros métodos.

Para Goldberg (1989) el tema central de la investigación en AG consiste en la robustez: el balance entre la eficacia y la eficiencia necesaria para sobrevivir en muchos ambientes diferentes.

En trabajos más recientes Goldberg expone algunos motivos por los que los AG pueden ser atractivos para el desarrollo de aplicaciones:

1. Pueden resolver problemas difíciles de forma rápida y confiable.
2. Son fáciles de enlazar a simulaciones y modelos existentes.
3. Son extensibles.
4. Son fáciles de hibridar.

1.2.4 Mecanismos de reproducción o selección

Durante la fase reproductiva se seleccionan los individuos de la población para cruzarse y producir descendientes, que constituirán, una vez mutados, la siguiente generación de individuos. La selección de padres se efectúa usando un procedimiento que favorezca a los individuos mejor adaptados. Según dicho procedimiento, los individuos bien adaptados se escogerán probablemente varias veces por generación, mientras que los pobremente adaptados al problema,

tendrán muy pocas probabilidades de incluirse en la próxima generación. Existe gran cantidad de métodos de selección. A continuación se muestran algunos.

1.2.4.1 Selección por torneo

La idea principal de este método consiste en realizar la selección en base a comparaciones directas entre individuos. Existen dos versiones de selección mediante torneo:

- Determinística
- Probabilística

En la versión determinística se selecciona al azar un número p de individuos (generalmente se escoge 2,3 o 4). De entre los individuos seleccionados se selecciona el más apto para pasarlo a la siguiente generación.

La versión probabilística únicamente se diferencia en el paso de selección del ganador del torneo. En vez de escoger siempre el mejor se genera un número aleatorio del intervalo [0..1], si es mayor que un parámetro *p* (fijado para todo el proceso evolutivo) se escoge el individuo más alto y en caso contrario el menos apto. Generalmente *p* toma valores en el rango de $0.5<p<1$.

Variando el número de individuos que participan en cada torneo se puede modificar la presión de selección. Cuando participan muchos individuos en cada torneo, la presión de selección es elevada y los peores individuos apenas tienen oportunidades de reproducción. Un caso particular es el *elitismo global*. Se trata de un torneo en el que participan todos los individuos de la población con lo cual la selección se vuelve totalmente determinística. Cuando el tamaño del torneo es reducido, la presión de selección disminuye y los peores individuos tienen más oportunidades de ser seleccionados.

Elegir uno u otro método de selección determinará la estrategia de búsqueda del Algoritmo Genético. Si se opta por un método con una alta presión de selección se centra la búsqueda de las soluciones en un entorno próximo a las mejores soluciones actuales. Por el contrario, optando por una presión de selección menor se deja el camino abierto para la exploración de nuevas regiones del espacio de búsqueda.

1.2.4.2 Selección elitista

Este método plantea que en la nueva generación se incluyan los *n* mejores representantes de la generación actual, desechándose entonces los peores adaptados de la población.

1.2.4.3 Selección por ruleta

Es posiblemente el método más utilizado desde los orígenes de los Algoritmos Genéticos. A cada uno de los individuos de la población se le asigna una parte proporcional a su ajuste de una ruleta, de tal forma que la suma de todos los porcentajes sea la unidad. Los mejores individuos recibirán una porción de la ruleta mayor que la recibida por los peores. Generalmente la población está ordenada en base al ajuste por lo que las porciones más grandes se encuentran al inicio de la ruleta. Para seleccionar un individuo basta con generar un número aleatorio del intervalo [0...1] y devolver el individuo situado en esa posición de la ruleta. Esta posición se suele obtener recorriendo los individuos de la población y acumulando sus proporciones de ruleta hasta que la suma exceda el valor obtenido.

Es un método muy sencillo, pero ineficiente a medida que aumenta el tamaño de la población (su complejidad es $O(n^2)$. Presenta además el inconveniente de que el peor individuo puede ser seleccionado más de una vez.

Existen muchos otros algoritmos de selección. Unos buscan mejorar la eficiencia computacional, otros el número de veces que los mejores o peores individuos pueden ser seleccionados. Algunos de estos algoritmos son muestreo determinístico, escalamiento sigma, selección por jerarquías, estado uniforme, sobrante estocástico, brecha generacional, etc.

1.2.5 Mecanismos de cruzamiento

El mecanismo de cruce es el que permite "confrontar ideas para el desarrollo de la población en la búsqueda del mejor". La obtención de individuos nuevos a partir de los que existen es una de las características más interesantes e importantes en el trabajo de los AG. Este proceso se le llama cruzamiento, a semejanza del proceso natural.

1.2.5.1 Técnicas para realizar el cruce

Producto de la importancia que tiene el operador de cruzamiento para los AG diferentes técnicas de cruzamiento han sido propuestas y analizadas.

Cruzamiento de uno, dos y múltiples puntos: El cruzamiento de un punto de cruce es el que utiliza el AG simple, coge dos padres seleccionados y corta sus segmentos de genes en una posición escogida al azar, para producir dos subsecuencias de genes iniciales y dos subsecuencias de genes finales. Después se intercambian las subsecuencias finales, produciéndose dos nuevos cromosomas completos (ver Fig. I-2. Cruzamiento de un punto). Ambos descendientes heredan genes de cada uno de los padres.

En el esquema de cruzamiento de dos puntos, dos puntos son seleccionados aleatoriamente y los segmentos de las cadenas entre ellos intercambiados. El cruzamiento de múltiples puntos trata cada cadena como un anillo de bits dividido por k puntos de cruce en k segmentos. Los segmentos alternados son intercambiados entre el par de cadenas a entrecruzar.

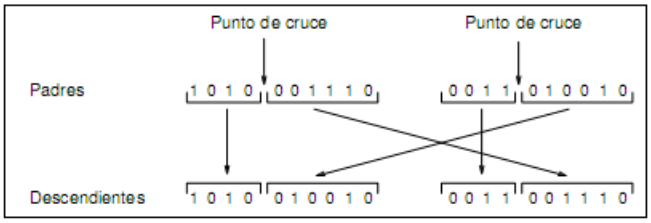


Fig. I-2. Cruzamiento de un punto

Cruzamiento uniforme: Es el intercambio de bits entre las cadenas, en vez de segmentos como los casos anteriores. En cada posición de la cadena los bits son probabilísticamente intercambiados con una probabilidad fija.

Una clasificación de los operadores de cruzamiento los analiza según las nociones de peso posicional y peso distribucional. Un operador de cruzamiento es de peso posicional si la probabilidad de que un bit sea cambiado depende de su posición. El peso distribucional de un operador de cruzamiento es relativo al número de bits intercambiados por el operador. Si la distribución del número no es uniforme el operador de cruzamiento tiene peso distribucional. Así se dice que el operador de cruzamiento de un solo punto tiene el máximo peso posicional y el

menor distribucional y el cruzamiento uniforme tiene el máximo peso distribucional y el mínimo posicional.

Estudios empíricos y teóricos han comparado los diferentes tipos de cruzamiento en cuanto a una medida para cuantificar la destrucción de esquemas, y a la potencialidad de exploración del espacio de búsqueda. Otro aspecto estudiado es la relación tamaño de la población y tipo de cruzamiento.

1.2.6 El operador de mutación

La mutación es un proceso, similar al biológico en el cual un gen de un individuo muta o cambia al otro valor posible. Este proceso es muy importante ya que puede ser que un individuo malo tuviera alguna característica muy buena. Cuando este individuo pasa por el proceso de reproducción existe una alta probabilidad de que sea eliminado y por lo tanto se pierda esa característica deseable. La recuperación de esta característica puede ser prácticamente imposible a través de los otros mecanismos genéticos.

La casualidad que a veces permite hallazgos importantes, que por los caminos lógicos podría demorarse mucho, es simulada a través de la mutación.

Puede ser que ninguno de los individuos de la población tenga una característica (y que hasta incluso ninguno de los anteriores tuvo) y que esa característica es el detalle que necesitan para lograr una mejoría importante, entonces esto puede obtenerse también a través de la mutación.

El proceso es muy simple y solo consiste en determinar cuál es el gen que mutará el organismo. La selección de este gen se realiza de forma aleatoria. Una vez que se obtiene solo es necesario cambiar el valor de este en esa posición de la cadena por otro valor generado aleatoriamente como se muestra en la Fig. I-3. Operador de mutación.

Fig. I-3. Operador de mutación

Para el AG es importante tener control de la mutación ya que estos cambios pueden tener tanto un efecto conveniente como perjudicial.

Para esto se utiliza un parámetro llamado Probabilidad de Mutación, el cual servirá para determinar si un individuo mutará o no. Como se ve aquí tampoco se trabaja de forma determinística sino a través de las probabilidades.

1.2.7 Aspectos importantes sobre AG

Algunos aspectos han quedado claros de las investigaciones analíticas y empíricas:

1. Incrementando la probabilidad de cruzamiento se incrementa la recombinación de bloques de construcción pero también se incrementa la destrucción de buenas cadenas.
2. Incrementando la probabilidad de mutación se tiende a transformar la búsqueda genética en una búsqueda aleatoria, pero esto también permite reintroducir pérdida de material genético.
3. Incrementando el tamaño de la población se incrementa la diversidad y reduce la probabilidad de que el AG haga una convergencia prematura, pero esto también incrementa el tiempo requerido para que la población converja a las regiones optímales en el espacio de búsqueda.

1.2.8 Utilización de Algoritmos Genéticos en LDS

Los Algoritmos Genéticos han sido utilizados con anterioridad en la técnica LDS. El primer trabajo referenciado en este sentido data del año 1996 (George, 1996) donde se utiliza un algoritmo genético clásico utilizando como función de aptitud un enfoque que valora al resumen lingüístico a partir de su aptitud para lograr un compromiso entre especificidad (utilizando descriptores de restricciones) y generalización (utilizando descriptores de composición de la

estructura).

Otros trabajos de LDS, fundamentalmente en aplicaciones de series de tiempo han sido abordados mediante AG. Un ejemplo típico es el publicado recientemente en (Castillo, 2011) donde se aplica una AG multiobjetivo a un problema que asume una jerarquización en el particionamiento de la dimensión tiempo y cuya aplicación está relacionada con el flujo de pacientes a un centro médico. En esta investigación se utiliza un cromosoma de longitud variable para representar el resumen lingüístico de los datos. Los genes codifican las proposiciones lingüísticamente cuantificadas que conforman dicho resumen y que son estereotipadas mediante "*QRy*'s are *S*". La población inicial es generada con individuos (resumen lingüístico) con una cantidad aleatoria de proposiciones, siendo estas igualmente generadas de forma aleatoria. El operador de cruzamiento utilizado implementa la operación de cruce en un punto con probabilidad de 0.5 y se proponen cuatro tipos de mutación que constituyen las operaciones que actúan directamente sobre el desarrollo de las proposiciones: una mutación realizada de forma clásica que introduce una perturbación aleatoria sobre un gen y otros tres tipos de mutaciones especializadas para el problema de LDS en series de tiempo; estas mutaciones tienen como objetivos incidir de forma "inteligente" en los genes del cromosoma con el objetivo de mejorar los valores de validez de las proposiciones.

1.3 Plataformas de desarrollo para los Algoritmos Genéticos y aplicaciones de la Lógica Difusa

La librería java JGAP para el desarrollo de AG y el software Xfuzzy para el modelado de la lógica difusa, constituyen las plataformas de desarrollo propuestas para la presente investigación.

1.3.1 JGAP, plataforma de desarrollo de Algoritmos Genéticos

JGAP son las siglas de *Java Genetic Algorithms Package* (paquete de algoritmos genéticos para Java). Es un componente de programación de algoritmos genéticos que se utiliza como un *framework*. Fue diseñado para que fuera muy fácil de usar, siendo altamente modular y configurable, llegando a ser realmente fácil crear incluso nuevos y personalizados operadores genéticos.

JGAP tiene clases e interfaces para representar:

- Genes (*Gene*).
- Cromosomas (*Chromosome*).
- Individuos *(IChromosome).*
- La población *(Genotype).*
- La función de aptitud *(FitnessFunction).*
- Operadores genéticos.

Se trata de una solución genérica, sin relación alguna con un problema particular. Por esa razón se deben crear nuevas clases que heredan o implementan las clases e interfaces mencionadas. Así se adapta JGAP al problema específico que se quiere solucionar.

En la Fig. I-4 se muestra el diagrama de clases implementado en JGAP.

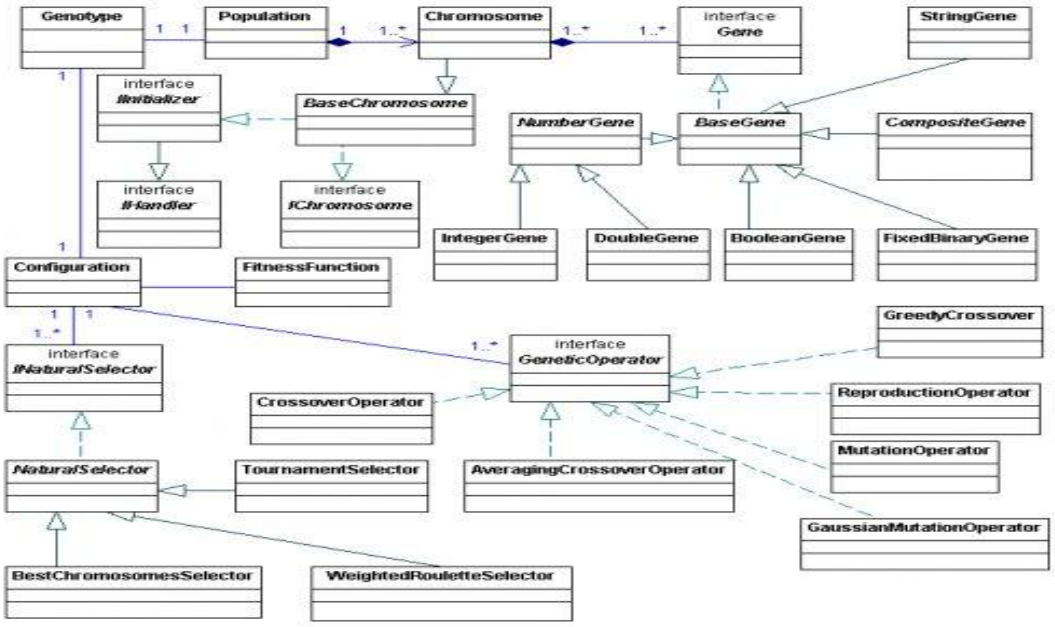


Fig. I-4. Diagrama de clases de la librería JGAP

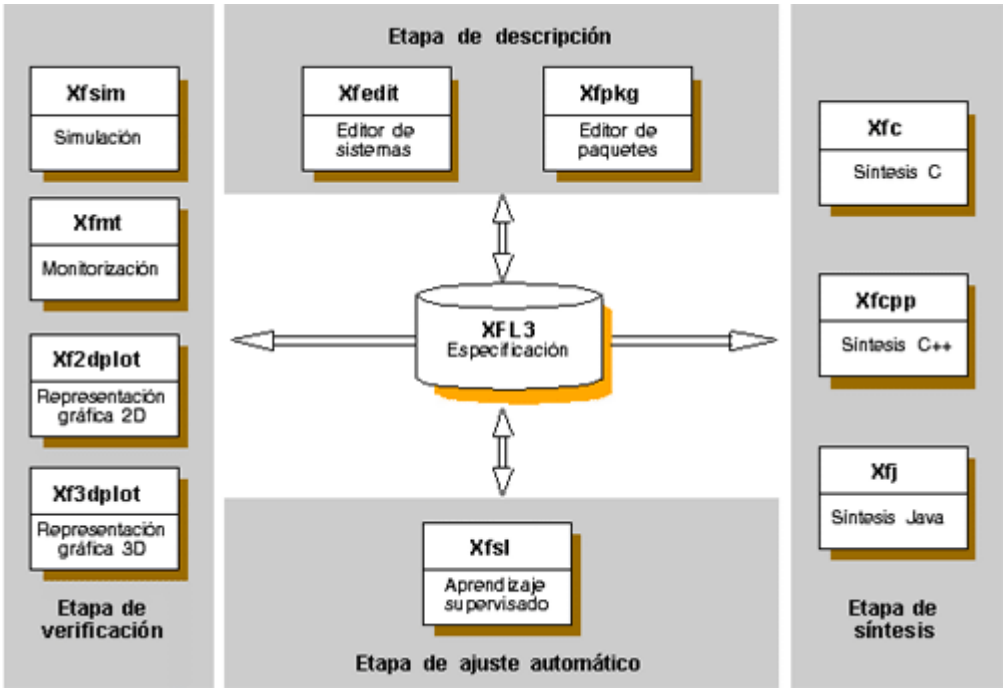
Evolución en JGAP

JGAP crea varias soluciones (individuos) aleatoriamente con el método *randomInitialGenotype*(). Aplica operadores genéticos (mutación y combinación) para que

surjan nuevas soluciones con el método *evolve*(). Pone a prueba cada solución con la función de ajuste *evaluate*(). Y por último retorna el cromosoma del individuo mejor adaptado con *getFittestChromosome*().

1.3.2 XFuzzy plataforma de desarrollo para la modelación de sistemas de Inferencia Borrosos

Xfuzzy 3.0 es un entorno de desarrollo para sistemas de inferencia basados en lógica difusa. Está formado por varias herramientas que cubren las diferentes etapas del proceso de diseño de sistemas difusos, desde su descripción inicial hasta la implementación final. Sus principales características son la capacidad para desarrollar sistemas complejos y la flexibilidad para permitir al usuario extender el conjunto de funciones disponibles. El entorno ha sido completamente programado en Java, de forma que puede ser ejecutado sobre cualquier plataforma que tenga instalado el JRE (Java Runtime Environment). La siguiente figura muestra el flujo de diseño de Xfuzzy 3.0.



La etapa de descripción incluye herramientas gráficas para la definición del sistema difuso. La etapa de verificación está compuesta por herramientas de simulación, monitorización y

representación gráfica del comportamiento del sistema. La etapa de ajuste facilita la aplicación de algoritmos de aprendizaje. Finalmente, la etapa de síntesis incluye herramientas para generar descripciones en lenguajes de alto nivel para implementaciones software o hardware.

El nexo entre todas las herramientas es el uso de un lenguaje de especificación común, XFL3, que extiende las capacidades de XFL, el lenguaje definido en la versión 2.0 de Xfuzzy. XFL3 es un lenguaje flexible y potente, que permite expresar relaciones muy complejas entre variables difusas por medio de bases de reglas jerárquicas y conectivas, modificadores lingüísticos, funciones de pertenencia y métodos de defuzzificación definidos por el usuario.

Las diferentes herramientas pueden ser ejecutadas como programas independientes. El entorno integra a todas ellas bajo una interfaz gráfica de usuario que facilita el proceso de diseño.

1.4 Conclusiones parciales

A partir del análisis de la bibliografía consulta cuyos aspectos más esenciales han sido expuestos en el presente capítulo se pueden concluir los siguientes elementos:

- LDS constituye una técnica en actual desarrollo cuyos preceptos la hacen de gran utilidad al brindar resúmenes de los datos en un formato humanamente comprensible mediante el empleo de un lenguaje natural cercano al problema objeto de aplicación.

- En LDS, la generación automática de los resúmenes ha demostrado la necesidad de emplear meta-heurísticas de búsqueda no exhaustiva que permitan realizar una exploración eficiente del voluminoso espacio de soluciones que presuponen todas las posibles proposiciones lingüísticas a analizar para conformar un resumen lingüístico sobre los datos. En este contexto los Algoritmos Genéticos pueden desarrollar un rol importante.

- LDS ha sido empleado sobre datos de *creep* demostrando que es capaz de describir adecuadamente el comportamiento de esta propiedad mecánica de los aceros en relación con las variables predictoras utilizadas en trabajos previos para obtener modelos basados en técnicas de aprendizaje automatizado. A partir de esto se ha proclamado la conveniencia de utilizar LDS para explorar los datos de *creep* con el objetivo de buscar posibles relaciones interesantes y útiles para el proceso de diseño de nuevos aceros ferríticos.

CAPÍTULO II. DISEÑO DE UN MODELO HÍBRIDO DE ALGORITMOS GENÉTICOS Y LDS PARA LA EXPLORACIÓN DE DATOS DE *CREEP*

Como ha sido referido anteriormente, la técnica LDS durante el proceso de obtención de proposiciones lingüísticas de forma automática, necesita desarrollar un proceso de búsqueda que resulta costoso. Esta situación ha motivado la idea de desarrollar un modelo híbrido que permita, utilizando los preceptos de LDS, obtener resúmenes lingüísticos de una base de casos mediante el empleo de Algoritmos Genéticos en la fase de generación automática de las proposiciones lingüísticas que conforman el resumen.

A continuación se muestra un pseudocódigo con la caracterización general del modelo híbrido LDS+AG propuesto:

```
Paso 1.    Modelado difuso de la base de casos
Paso 2.    Especificar protoforma del resumen lingüístico a obtener
Paso 3.    SI protoforma es de Tipo cero          //Ver epígrafe 1.1.2
Paso 3.1       Calcular validez del resumen propuesto
           SINO
               // Generar y calcular validez de las proposiciones mediante AG
Paso 3.2       AG (protoforma)
Paso 4.    FIN-SI
Paso 4.    Devolver conjunto óptimo de proposiciones que conforman el resumen
```

Observar que en el paso 3.2 se invoca al modelo de Algoritmo Genético pasándole como parámetro la protoforma especificada por el usuario. En este caso, la protoforma induce determinada restricción para el AG en relación al tipo de proposiciones lingüísticas con que el mismo debe tratar lo cual afecta el proceso de generación de nuevos individuos y la aplicación de los operadores genéticos. Por ejemplo, si se trata de la protoforma Tipo 2, el AG solo podrá generar individuos que tengan el sumarizador *S* y el calificador *R* tal y como fueron definidos por el usuario, mientras que el término lingüístico del cuantificador debe ser generado

aleatoriamente. De la misma forma, el AG debe tener en cuenta esta restricción al aplicar los operadores genéticos que afecten la estructura de las proposiciones.

En el resto del capítulo se describen los elementos de diseño de los pasos fundamentales del modelo, así como la experimentación y análisis de resultados realizados. Es importante resaltar que el modelo propuesto ha sido diseñado específicamente para la exploración de los datos de *creep*, por lo que las decisiones tomadas en cuanto al funcionamiento y estructura del modelo han sido consecuentes con las características de estos datos.

2.1 Modelado difuso de los datos de *creep*

En la presente investigación se utilizó el mismo modelado difuso de los datos de *creep* empleado en (Díaz, 2011). Las variables de entrada relacionadas con la composición química (*por ciento de C, por ciento de Si, por ciento de Mn, etc.*) son modeladas como variables difusas con siete términos lingüísticos (‘*muy bajo*’, ‘*bajo*’, ‘*medio-bajo*’, ‘*medio*’, ‘*medio-alto*’, ‘*alto*’, ‘*muy alto*’) utilizando funciones de pertenencia trapezoidales uniformemente distribuidas sobre el universo de discurso. Las variables *temperatura* y *crt* (*creep rupture time*) contienen seis y ocho funciones de pertenencia trapezoidales respectivamente para representar sus términos lingüísticos. Estos son definidos como (‘*muy bajo*’, ‘*bajo*’, ‘*medio-bajo*’, ‘*medio-alto*’, ‘alto’, ‘muy alto’) y (‘*muy corto*’, ‘*corto*’, ‘*corto-medio*’, ‘*medio*’, ‘*medio-largo*’, ‘*largo*’, ‘*muy largo*’, ‘*extra-largo*’) respectivamente.

Para la variable *creep* (usada como sumarizador) se plantea una distribución especial de funciones de pertenencia debido a las características de los valores observados. Como resultado, la variable difusa *creep* fue diseñada utilizando una concentración de 8 funciones de pertenencia entre los valores 18 y 330 del universo de discurso y otra función que cubre el resto del universo de discurso (Fig. II-1). Los términos lingüísticos de *creep* son definidos como: *‘muy bajo’, ‘bajo’, ‘medio-bajo’, ‘medio’, ‘medio-alto’, ‘alto’, ‘muy alto’, ‘extra-alto’ e ‘ideal’.*

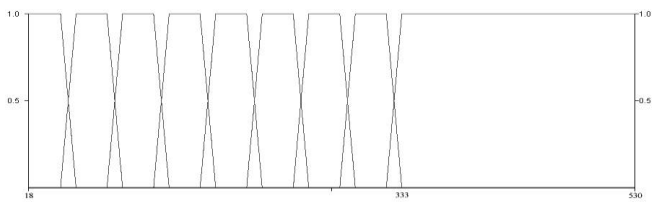


Fig. II-1. Funciones de pertenencia de la variable difusa *creep*

Como cuantificador se utilizó una variable con cinco términos lingüísticos (Fig. II-2), representados por funciones de campana uniformemente distribuidas en el rango de 0 a 1. Los términos lingüísticos empleados fueron: *'pocos', 'el 25 %', 'la mitad', 'el 75 %' y 'la mayoría'*

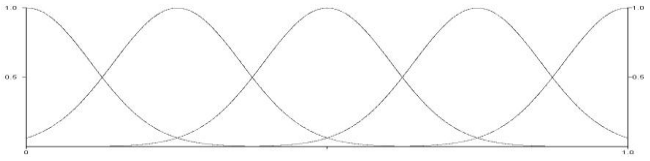


Fig. II-2. Funciones de pertenencia de la variable difusa cuantificador

Características de LDS aplicado al Creep

- Teniendo en cuenta que el interés fundamental es obtener un resumen lingüístico que describa el comportamiento de las variables predictoras con respecto a la variable *creep*, esta fue utilizada como sumarizador.
- Los predicados difusos en el calificador son relacionados utilizando solo el operador *and* el cual se obtiene como $\mu_{AND(a,b)}(y_i) = min(\mu_a(y_i), \mu_b(y_i))$

2.2 Desarrollo y análisis de un modelo de Algoritmo Genético para ser utilizado con LDS

En este epígrafe se abordan las características de todos los componentes y funcionalidades del AG así como los experimentos, análisis estadísticos y de resultados efectuados para decidir las características finales de algunos operadores genéticos.

2.2.1 Diseño de los experimentos y pruebas estadísticas

Los experimentos realizados están dirigidos a elegir las variantes de algunos operadores utilizados en el modelo de AG. Para la definición de las diferentes variantes siempre se realizaron experimentos con diez corridas del AG para cada variante y el valor de 100 para los parámetros cantidad de generaciones (*CG*) y tamaño de la población (*TP*).

Los indicadores de comparación utilizados se describen como:

- *A* = promedio de *F* (función de aptitud). Denota la calidad del resumen lingüístico
- *C* = cantidad de evaluaciones de *T*

Como fue referido anteriormente, la evaluación de T precisa recorrer completamente la base de casos por lo que puede resultar un proceso costoso en cuanto a procesamiento y tiempo de ejecución. Como consecuencia, la cantidad de evaluaciones de T (C) es una variable a cuidar durante el desarrollo del modelo; resulta deseable mantener valores pequeños de C.

En el análisis estadístico de los experimentos se utiliza el test de Wilcoxon para realizar una comparación par a par de los resultados obtenidos y determinar si existen o no diferencias significativas en los indicadores propuestos. Para el cálculo más exacto de todas las significaciones se utilizó la técnica de Monte Carlo que simula aleatoriamente 10000 muestras con distribución similar a la de los datos que se comparan y permite entonces estimar la significación promedio y un intervalo de confianza de la misma del 99%. Se consideró como significativo un valor promedio de significación menor de 0.05.

En estos experimentos se utilizaron 22 atributos de la base de casos *creep* referida en el apartado 1.1.4 *y* la estructura de las proposiciones a obtener es guiada por la protoforma 5, es decir no se especifica una estructura de los resúmenes a obtener sino que se obtienen grupo de resúmenes con una estructura no predefinida.

2.2.2 Diseño del cromosoma

Como ha sido descrito anteriormente un resumen lingüístico está formado por un conjunto de sentencias o proposiciones lingüísticamente cuantificadas. LDS intenta obtener el resumen lingüístico más completo y útil desde los datos que explora y para ello necesita encontrar las proposiciones de mayor calidad (grado de validez). Lo anterior significa que por ejemplo, para un sistema de toma decisiones no basta con contar solamente con la mejor proposición sino que podrían ser útiles todas aquellas que posean un grado de validez mayor que un valor dado.

Es conocido que un AG guía su proceso evolutivo a partir del valor de aptitud de sus cromosomas, constituyendo la estructura de estos la codificación del objeto a optimizar. Al utilizar AG en el proceso de búsqueda de LDS se presenta la disyuntiva de cuál debe ser el objeto a optimizar: las proposiciones lingüísticas que conforman el resumen o el propio resumen lingüístico como un todo. Utilizar el primero tiene el inconveniente de que AG intentará encontrar una mejor proposición lingüística lo cual no es el objetivo de LDS; lo más adecuado

resulta entonces, la utilización del resumen lingüístico como objeto de optimización por parte del AG.

A partir de este análisis se decidió diseñar la estructura del cromosoma de forma que represente un resumen lingüístico. En este sentido un cromosoma está formado por una cantidad *n* fija de proposiciones o sentencias lingüísticamente cuantificadas; siendo *n* un parámetro del modelo. Es decir, un gen es una proposición y el alelo será una instancia específica de esta, por ejemplo: 'La *mayoría* de los aceros con *alto por ciento de aluminio* y *bajo porciento de bromo* poseen un valor de *creep alto'*. Luego un gen estará formado por un cuantificador *Q*, un calificador *R*, un sumarizador *S* y un grado de validez *T*, lo cual coincide con la composición de una proposición lingüísticamente cuantificada. Este diseño del cromosoma ha sido utilizado con anterioridad, por ejemplo, en el trabajo de Castillo (2011) con la diferencia de que en el mismo fue utilizado un tamaño de cromosoma variable. La Fig. II-3 muestra la estructura descrita del cromosoma utilizado en la presente investigación.

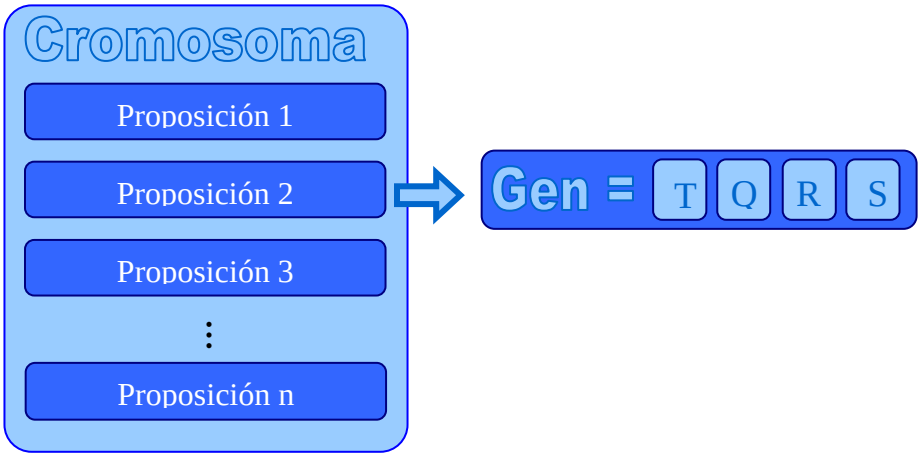


Fig. II-3. Estructura de un cromosoma

El diseño propuesto tiene el inconveniente de que los operadores tradicionales de cruzamiento y mutación no tienen una influencia directa sobre la evolución o mejoramiento de las proposiciones lingüísticas en específico (lo cual es uno de los objetivos de LDS) sino que lo hacen sobre un conjunto de estas. Esta situación provocó que se analizara la implementación de otro operador (Operador de Mejoramiento de Sentencias) que será analizado más adelante y que sí influye directamente sobre las proposiciones lingüísticas.

2.2.3 Inicialización de la población

La población se inicializa aleatoriamente con un número de individuos (cromosomas) definido previamente y que constituye un parámetro del modelo. Cada cromosoma contiene una cantidad fija de proposiciones lingüísticas que también son generadas de forma aleatoria durante este proceso. Es importante destacar nuevamente las restricciones que imponen en este proceso, la protoforma especificada para la generación del resumen lingüístico. Estas definen que parte del alelo debe ser especificado literalmente y que parte debe ser generado aleatoriamente.

2.2.4 Función de aptitud

Como se planteó, LDS intenta obtener el resumen lingüístico más completo y útil posible desde los datos que explora y para ello necesita encontrar las proposiciones de mayor calidad (grado de validez). Para lograr esto se necesitan optimizar dos elementos fundamentales: la calidad de cada proposición lingüística en particular y el grado de diversidad del resumen lingüístico en su totalidad. Esto permitiría utilizar un conjunto de proposiciones útiles e interesantes y a la vez abarcador de un área grande de conocimientos resumidos sobre el problema en cuestión.
Esta idea constituye la base para la propuesta de la función de evaluación o de aptitud del AG para un individuo o cromosoma *i*. De forma general es definida como:

$$F_i = m_g G_i + m_d D_i$$

Dónde *G* (*Goodness*) denota el grado de virtud de un cromosoma; en otras palabras, indica cuan bueno es un resumen lingüístico en cuanto a la calidad de las proposiciones que lo componen. *D* (*Diversity*) hace referencia al grado de diversidad existente en el cromosoma, es decir, específica cuan diferentes resultan entre sí las proposiciones que integran el resumen. Los valores m_g y m_d son pesos que denotan la importancia dada al grado de virtud o de diversidad respectivamente para determinar cuan deseable es el resumen representado.

El indicador de virtud *G* se define como

$$G_i = \frac{\sum_{s=1}^{n} T_s \times fl_s}{n}$$

Dónde T_s representa el grado de validez de la sentencia *s* y fl_s indica la fortaleza lingüística de dicha sentencia. Este último término ha sido introducido en la función de aptitud como un grado de libertad que se le brinda al usuario del modelo para influir sobre el tipo de proposición (más bien de cuantificador) que desea potenciar en el resumen. Por tanto la definición de la fortaleza lingüística está estrechamente relacionada con el problema a resolver y el criterio del experto; constituye un parámetro del modelo. La fortaleza lingüística de una proposición está dada por su importancia en la toma de decisiones, por ejemplo, una sentencia que exprese: 'la ***mayoría*** de los trabajadores *bien calificados* tienen *salarios altos*' puede tener una mayor fortaleza lingüística y por tanto una mayor potencialidad en el soporte de un sistema de toma de decisiones que otra proposición que indican 'la ***mitad*** de los trabajadores *jóvenes* tienen *salarios altos*', a pesar de tener ambas, un grado de validez alto.

En el modelo desarrollado sobre datos de *creep* se decidió asignar el mayor valor de fortaleza lingüística a los términos lingüísticos '*mayoría*' y descender el valor hacia el término '*pocos*'. Esto se debe a que mientras más cercano sea el cuantificador a '*mayoría*' la respectiva proposición informa que hay un mayor número de casos consecuente con ella y por tanto podemos utilizar dicha información con una mayor confiabilidad. En el caso contrario, mientras más nos acercamos a '*pocos*', la información brindada por la proposición resulta menos confiable de utilizar pues existe una menor cantidad de casos que la soporten. Los valores exactos de fortaleza lingüística utilizados se muestran en la Tabla II-1:

Tabla II-1. Valores de fortaleza lingüística utilizados por el modelo propuesto para datos de *creep*

Cuantificador	**Fortaleza Lingüística**
Pocos	0.15
El 25 %	0.25
La mitad	0.5
El 75 %	0.75
La mayoría	1

La diversidad *D* de un resumen *i* se define como:

$$D_i = \frac{cgs}{n}$$

siendo *n* la cantidad total de sentencias del cromosoma y *cgs* la cantidad de grupos de similitud. Este último se determina recorriendo el conjunto de sentencias mediante el siguiente proceso:

- Se escoge la primera sentencia y se compara con el resto de las sentencias utilizando una función de similitud, creando así un subgrupo de sentencias con aquellas que sean similares a la primera.

- Posteriormente se escoge la segunda sentencia y se compara con el resto de las sentencias creando un nuevo grupo de similitud.

- El proceso se repite con todas las sentencias del cromosoma creando todos los posibles grupos de similitud.

La función de similitud entre dos sentencias s_1 y s_2 se enuncia de la siguiente manera:

$$S_{s_1,s_2} = \begin{cases} Si & si\ D_{s_1,s_2} \leq 2 \\ No & en\ otro\ caso \end{cases}$$

dónde D es una medida de distancia entre dos vectores v_1 , v_2 definida como:

$$D_{s_1,s_2} = \sum_{i=0}^{l} F(v_1[i], v_2[i])$$

siendo *l* la longitud del vector la cual es de tamaño fijo y se corresponde con el valor de uno más la cantidad de atributos seleccionados para obtener el resumen lingüístico

Los vectores v_1, v_2 son construidos teniendo en cuenta las siguientes características:

- representan las sentencias o proposiciones a comparar,
- están formados por *l* componentes,
- cada componente caracteriza un término lingüístico de cada posible variable difusa utilizada para generar las sentencias de la siguiente forma:
  - el primer componente (*v[0]*) refiere el término lingüístico del cuantificador *Q*,

- el último componente (*v[l-1]*) refiere el término lingüístico de la variable difusa utilizada como sumarizador *S*,
- los componentes intermedios (*v [1]* hasta *v [l-2]*) refieren los términos lingüísticos de las variables difusas que conforman el calificador *R*, manteniendo el orden de los atributos respectivos en la base de casos. Es decir, *v [1]* refiere el primer atributo de la base de casos, *v [2]* el segundo y así sucesivamente. La única no correspondencia ocurre con el atributo que es seleccionado como sumarizador.

- Cada componente toma valores relativos al término lingüístico de la variable difusa que representa; por ejemplo, si refiere el primer término lingüístico de la variable toma el valor 1, si refiere el segundo término lingüístico toma el valor 2 y así sucesivamente. En caso de no utilizarse la variable difusa en la sentencia, el componente respectivo toma valor 0.

La función *F* se define como:

$$F(v_1[i],v_2[i]) = \begin{cases} 1 & si\ |v_1[i]-v_2[i]| > 0\ round(20\%\ de\ ctl_i)\ ó \\ & si\ v_1[i]=0\ y\ v_2[i]\neq 0 \qquad\qquad ó \\ & si\ v_1[i]\neq 0\ y\ v_2[i]=0 \\ 0 & en\ otro\ caso \end{cases}$$

siendo ctl_i la cantidad de términos lingüísticos de la variable difusa que representa el componente *i*. Esto significa que por ejemplo, al comparar los términos lingüísticos (‘muy *bajo*’ y ‘*medio*’) de una variable difusa (con términos lingüísticos: ‘*muy bajo*’, ‘*bajo*’, ‘*medio-bajo*’, ‘*medio*’, ‘*medio-alto*’, ‘*alto*’, ‘*muy alto*’) utilizados en dos proposiciones representadas por dos vectores, estos son diferentes pues *F=1* debido a que |1 – 4| = 3 > round(7*20/100) = 1.

2.2.5 Proceso de selección

La utilización del método de selección resulta esencial para el buen comportamiento de un AG. En la presente investigación se estudió el comportamiento en el modelo propuesto, de dos tipos de selección: la selección elitista y la selección por torneo.

En los siguientes epígrafes se analizan ambas variantes, así como los experimentos realizados para la decidir cuál variante resulta mejor en el problema.

2.2.5.1 Selección Elitista

La selección elitista plantea elegir los 'mejores' individuos o cromosomas y pasarlos a la próxima generación, desechando los individuos que por su bajo valor de aptitud son considerados 'malos'. La selección elitista utilizada en el modelo, escoge una cantidad de mejores cromosomas igual al tamaño de la población inicial. Es decir, se hace una selección de los mejores individuos que mantiene constante el número de cromosomas durante todo el proceso. Por ejemplo, si la población inicial es de 100 cromosomas y producto de los operadores genéticos, se añaden 80 nuevos cromosomas, el proceso de selección elitista escoge, para la próxima generación, a los 100 mejores individuos de los 180 existentes en la población en ese momento.

2.2.5.2 Selección por torneo

La variante elitista descrita tiene el inconveniente de que puede provocar una convergencia temprana en el algoritmo debido a la escasa diversidad que genera en la población; esto pudiera degradar en gran medida la calidad de las soluciones del modelo. Teniendo en cuenta lo anterior se decidió analizar otro tipo de selección que sí considera el problema de mantener un determinado nivel de diversidad en la población. Fue entonces considerada, la selección por torneo en la cual los individuos 'malos' también tienen cierta probabilidad de ser seleccionados para próxima generaciones.

Como fue descrito anteriormente, la selección por torneo gradúa el nivel de diversidad a partir del parámetro tamaño del torneo. En la presente investigación, este fue seleccionado a partir de un proceso de experimentación que arrojó un tamaño de torneo igual a 5 (con selección determinista del representante del torneo) como el mejor luego de experimentar con valores de 2, 3, 4, 5, 6 y 7.

2.2.5.3 Definición de la variante de selección utilizada en el modelo

Para definir cuál método de selección utilizar en el modelo propuesto se realizaron experimentos que permitieron comparar los métodos: Selección Elitista (SE) y Selección por Torneo con

tamaño de torneo igual a 5 (VT). Los resultados se muestran en las tablas Tabla II-2, Tabla II-3, Tabla II-4, Tabla II-5.

La Tabla II-2 refiere la comparación entre ambas variantes respecto al valor promedio de la función de aptitud (A) del mejor resumen lingüístico obtenido en cada corrida del modelo. Como se evidencia VE presenta un mejor comportamiento de A que VT (A_{VE} = 0.3808 vs A_{VT} = 0.3701), aunque según los resultados del análisis estadístico mostrado en la Tabla II-4 estas diferencias no son significativas. Al comparar la cantidad promedio de evaluaciones de T (denotado como C) (Tabla II-3) se observa una diferencia pequeña en los valores a favor de VE (C_{VE} = 12943 vs C_{VT} = 13100), en este caso tampoco se observan diferencias significativas tal y como lo confirma el análisis estadístico mostrado en la Tabla II-5.

Al analizar estos resultados se puede concluir que el modelo propuesto no es sensible al proceso de selección que desarrolla el AG. A partir de esta situación se define como variante a utilizar la selección elitista teniendo en cuenta que a pesar de no existir diferencias significativas en los resultados, los valores de A y C resultaron ligeramente mejores. Además esta variante es menos costosa computacionalmente que la selección por torneo.

Tabla II-2. Comparación de las variantes de selección en relación a los valores de la función de aptitud (A)

	N	Mean	Std. Deviation	Minimum	Maximum
VE	10	.3808	.02051	.35	.42
VT	10	.3701	.01062	.36	.39

Tabla II-3. Comparación de las variantes de selección en relación a la cantidad de evaluaciones de T (C)

	N	Mean	Std. Deviation	Minimum	Maximum
VE	10	12943,9000	204,67614	12456,00	13242,00
VT	10	13100,7000	371,15078	12431,00	13680,00

Tabla II-4. Resultados del análisis estadístico al comparar A_{VE} vs A_{VT}

			A_{VE} - A_{VT}
Z			-1.580(a)
Asymp. Sig. (2-tailed)			.114
Monte Carlo Sig. (2-tailed)	Sig.		.133
	99% ConfidenceInterval	LowerBound	.124
		UpperBound	.142
Monte Carlo Sig. (1-tailed)	Sig.		.066
	99% ConfidenceInterval	LowerBound	.060
		UpperBound	.072

Tabla II-5. Resultados del análisis estadístico al comparar C_{VE} vs C_{VT}

			C_{VE} vs C_{VT}
Z			-,968(a)
Asymp. Sig. (2-tailed)			,333
Monte Carlo Sig. (2-tailed)	Sig.		,376
	99% ConfidenceInterval	LowerBound	,364
		UpperBound	,389
Monte Carlo Sig. (1-tailed)	Sig.		,186
	99% ConfidenceInterval	LowerBound	,176
		UpperBound	,196

2.2.6 Operador de cruzamiento

El cruzamiento utilizado en el modelo propuesto implementa la operación de cruce en un punto con probabilidad de 0.9. Es importante destacar que la función principal de este operador es buscar diversidad dentro del resumen lingüístico, él no tiene influencia directa sobre el mejoramiento de las proposiciones lingüísticas. Como se planteó anteriormente, esta situación es la condicionante del desarrollo del operador de mejoramiento de sentencias que si actúa directamente sobre la evolución de las sentencias o proposiciones lingüísticas.

2.2.7 Operador de mutación

El operador de mutación implementado en el modelo propuesto actúa seleccionado uno o varios genes y los cambia por otros generados también de forma aleatoria. En términos de LDS, este operador realiza la sustitución de una o varias proposiciones lingüísticas por otras, es decir, tiene una influencia directa sobre las proposiciones. Fue utilizado con una probabilidad de ocurrencia de 0.05

2.2.8 Operador de mejoramiento de sentencias

Como ha sido referenciado previamente, los operadores tradicionales tienen una influencia débil y/o indirecta sobre las proposiciones lingüísticas una vez que el objeto de optimización de AG respecto a LDS es el resumen lingüístico como un todo y no una proposición en específico. Esto constituye la motivación del desarrollo de un operador que tenga una influencia directa sobre la evolución y el mejoramiento de las sentencias o proposiciones.

El operador genético propuesto llamado Operador de Mejoramiento de Sentencias desarrolla una búsqueda local para mejorar las sentencias en un entorno cercano a ella y siguiendo la estrategia de *primero el mejor*. Se parte de la modificación de una sentencia s_1 a mejorar mediante una de las posibles transformaciones definidas para el modelo. Esta modificación produce una nueva sentencia s_2, si el valor de validez T_{S2} de la nueva sentencia es mejor que el de la sentencia original s_1 (T_{S1}), entonces se continúa la búsqueda a partir de s_2, de lo contrario la búsqueda se desarrolla ahora a partir de s_1. Es importante destacar que debido al amplio espacio de soluciones que caracteriza el problema abordado, la cantidad de transformaciones posible y la naturaleza aleatoria de las mismas en su aplicación, resulta prácticamente imposible la no generación de una nueva sentencia a partir de otra.

Fueron definidas seis posibles transformaciones a aplicar donde cuatro de ellas actúan sobre el calificador *R*, una sobre el cuantificador *Q* y otra sobre el sumarizador *S*. Estas transformaciones se definen como:

Transformación 1 en *R*: Cambia un predicado difuso completo en *R*. Es decir se selecciona aleatoriamente un predicado y se le cambia su variable lingüística seleccionando un término

lingüístico para la misma de forma aleatoria. Por ejemplo, el predicado '*concentración alta de carbono*' puede ser sustituido por '*concentración media de Silicio*'.

Transformación 2 en *R*: Cambia el término lingüístico de un predicado difuso de *R*. Selecciona aleatoriamente un predicado difuso en el cual es sustituido el término lingüístico por otro diferente seleccionado de forma aleatoria. Por ejemplo, el predicado '*concentración alta de carbono*' puede ser sustituido por '*concentración media de carbono*'.

Transformación 3 en *R*: Adiciona un nuevo predicado difuso en *R*. Selecciona aleatoriamente un nuevo predicado difuso siempre que sea posible y lo adiciona en *R*. Por ejemplo, si *R* = *concentración alta de Carbono*' Y '*concentración baja de Silicio*', esta transformación puede generar un R_{nuevo} = *R* Y '*concentración media de Aluminio*'.

Transformación 4 en *R*: Elimina un predicado difuso en *R*. Selecciona aleatoriamente un predicado difuso y lo elimina en *R* siempre que sea posible.

Transformación en *Q*: Sustituye el término lingüístico del cuantificador con otro 'cercano' a él generado de forma aleatoria. El término cercano refiere un valor de término lingüístico inmediato superior o inferior al anterior. Por ejemplo, si Q fue definido con los términos lingüísticos '*mayoría*', '*casi todos*', '*mitad*', '*pocos*' y '*muy pocos'* y el término lingüístico del cuantificador de una determinada sentencia es '*casi todos*', al aplicar esta transformación el cuantificador podría tomar el valor '*mayoría*' o '*mitad*'. Transformar hacia valores cercanos es consistente con el concepto de búsqueda local y hace que la transformación de una sentencia por este concepto se desarrolle de manera suave, sin perder determinadas características.

Transformación en *S*: Es una transformación similar a la descrita anteriormente pero aplicada al sumarizador *S*.

Las condiciones de parada de la búsqueda local aplicada por el operador de mejoramiento están determinadas por los siguientes parámetros:

- **Cantidad máxima de evaluaciones de *T* (*MET*)**. Coincide con la cantidad máxima de transformaciones aplicadas a la sentencia. La búsqueda se detiene si se alcanza este valor.

- **Cantidad máxima de evaluaciones continuas, sin mejoramiento de T (*METSM*)**. Hace referencia a la cantidad de transformaciones realizadas consecutivamente sobre una sentencia sin que mejore el valor de *T*. Es siempre menor que ***MET***. La búsqueda se detiene si se alcanza este valor.
- **Valor de validez deseado (*TD*)**.Valor de *T* mínimo deseado. Si la sentencia transformada alcanza un valor mejor que este, entonces se detiene la búsqueda.

2.2.8.1 Estrategia de aplicación del operador

El Operador de Mejoramiento de Sentencias se aplica sobre un porciento variable (p_m) de los mejores cromosomas (resúmenes lingüísticos) de la población; dentro del cromosoma es aplicado a la totalidad de sus genes. Es utilizado cada una cantidad fija de evoluciones, la cual se determina como un porciento (p_e) del total de evoluciones del AG. El porciento p_m para un porciento de generaciones g y un tamaño de población constante tp se define como:

$$p_m(g) = \begin{cases} \left(100 - \left(1.17^{40g/tp}\right)\right)/100 & si \ \left(100 - \left(1.17^{40g/tp}\right)\right)/100 > tp*0.1 \\ tp*0.1 & en\ otro\ caso \end{cases}$$

En el gráfico de la Fig. II-4 se muestra el comportamiento de p_m.

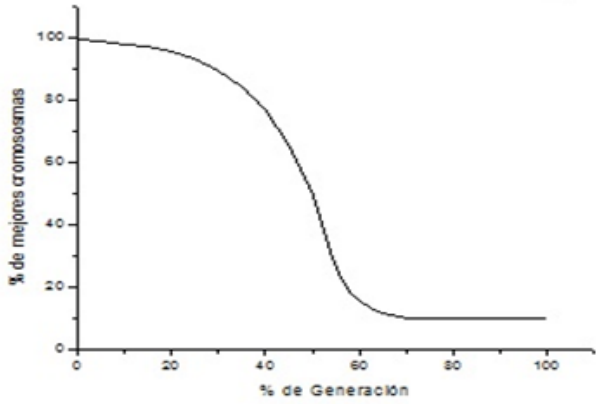


Fig. II-4. Gráfica de comportamiento de la función de selección de mejores cromosomas para la búsqueda local

El objetivo de la forma de esta función es lograr que la cantidad de cromosomas a mejorar durante el proceso de evolución sea mayor al principio que al final, teniendo en cuenta que al

inicio las proposiciones deben estar más distantes de ser buenas soluciones mientras que las existentes en las generaciones finales deben haber recibido suficiente refinamiento.

La aplicación de este operador se puede realizar utilizando dos variantes o estrategias fundamentales determinadas por la profundidad de la búsqueda local y el momento en que se aplica. Es importante destacar que en su aplicación se debe buscar tanto el mejoramiento de los valores de aptitud del resumen lingüístico como de la cantidad de evaluaciones de *T*. Para ambas variantes se utilizó un valor de *TD* = 0.95 y la frecuencia con que fue aplicado se calculó como el 6% del total de generaciones, es decir se aplica 16 veces durante el proceso evolutivo. Estos valores se obtuvieron a partir de la experimentación con varios valores.

Luego de la experimentación con diversas variantes, las más promisorias a priori resultaron ser las dos variantes que se presentan a continuación y de las cuales se realiza la comparación:

Variante I (VOM1). Se aplica el operador de mejoramiento con una búsqueda local poco profunda y homogénea durante todo el proceso. La profundidad de la búsqueda se determinó por los valores *MET* = 8 y *METSM* = 5 (valores obtenidos a partir de la experimentación con varios valores) y fue aplicada con estos mismos valores en cada momento de aplicación.

Variante II (VOM2). Se aplica el operador de mejoramiento con una profundidad no homogénea de forma que en la primera mitad del proceso evolutivo la búsqueda sea poco profunda y en la segunda sea más exhaustiva. Las profundidades de la búsqueda fueron determinadas por los siguientes valores de los parámetros:

Búsqueda local en la primera mitad (poco profunda): *MET* = 8 y *METSM* = 5

Búsqueda local en la segunda mitad (más profunda): *MET* = 16 y *METSM* = 10

2.2.8.2 Definición de la variante de aplicación del operador de mejoramiento de sentencias

Es necesario aclarar que las variantes propuestas fueron desarrolladas a partir de la variante obtenida anteriormente utilizando la selección elitista (VE). Es por ello que para definir la mejor variante de aplicación del operador de mejoramiento de sentencias a utilizar en el modelo y comprobar su comportamiento sobre dicho modelo, se realizó la comparación de ambas variantes entre sí y estas a su vez con VE. Los resultados de los experimentos y el análisis estadístico se muestran en las tablas Tabla II-6, Tabla II-7, Tabla II-8, Tabla II-9.

La Tabla II-6 refiere la comparación entre las tres variantes respecto al valor promedio de la función de aptitud (A) del mejor resumen lingüístico obtenido en cada corrida del modelo. Como se evidencia VOM2 presenta un mejor comportamiento de A que VOM1 (A_{VOM2} = 0.5216 vs A_{VOM1} = 0.5207) aunque según los resultados del análisis estadístico mostrado en la Tabla II-8 estas diferencias no son significativas. Sin embargo al analizar ambas variantes (VOM1, VOM2) respecto a VE se observa que las primeras muestran un comportamiento muy superior a esta, resultando las diferencias significativamente superiores de acuerdo al análisis estadístico también mostrado en la Tabla II-8.

Al realizar la comparación respecto a la cantidad promedio de evaluaciones de *T* (denotado como C) (ver Tabla II-7) se observa que VOM2 presenta un deterioro sustancial en este aspecto respecto al comportamiento de VOM1 (C_{VOM2} = 100933 vs C_{VOM1} = 95653) existiendo una diferencia significativa según el análisis estadístico expuesto en la Tabla II-9. La comparación sobre este indicador de VOM1 y VOM2 respecto a VE demuestra que este último posee un valor evidentemente mejor C_{VE} = 0.3808 lo cual es un resultado esperado si se tiene en cuenta que el operador de mejoramiento desarrolla su función mediante un proceso de búsqueda adicional al realizado por el AG simple.

A partir del análisis de los resultados se tomó la decisión de incorporar al modelo, la variante VOM1 teniendo en cuenta que a pesar de que VOM2 posee un mejor valor de A, este no demuestra superioridad pues la diferencia no es significativa y sin embargo si demuestra un deterioro significativo en cuanto a la cantidad de evaluaciones de T. Se concluye que la aplicación de la búsqueda con una profundidad determinada por los parámetros *MET* = 8 y *METSM* = 5 garantiza que durante la primera mitad las proposiciones hayan evolucionado hacia formas mejores en una magnitud tal que no se precisa de una búsqueda más profunda en la segunda mitad.

Tabla II-6. Comparación de las variantes de aplicación del operador de mejoramiento de sentencias en relación a los valores de la función de aptitud (A)

	N	Mean	Std. Deviation	Minimum	Maximum
VB	10	.3808	.02051	.35	.42
VOM1	10	.5207	.04065	.47	.60
VOM2	10	.5216	.03050	.47	.57

Tabla II-7. Comparación de las variantes de aplicación del operador de mejoramiento de sentencias en relación a la cantidad de evaluaciones de T (C)

	N	Mean	Std. Deviation	Minimum	Maximum
VE	10	12943,9000	204,67614	12456,00	13242,00
VOM1	10	95653.1000	3227.61045	89013.00	100096.00
VOM2	10	100933.8000	4090.04580	95160.00	110018.00

Tabla II-8. Resultados del análisis estadístico al comparar A_{VOM1} vs A_{VE}, A_{VOM2} vs A_{VE} y A_{VOM1} vs A_{VOM2}

			A_{VOM1} vs A_{VE}	A_{VOM2} vs A_{VE}	A_{VOM1} vs A_{VOM2}
Z			-2.803(a)	-2.803(a)	-.255(a)
Asymp. Sig. (2-tailed)			.005	.005	.799
Monte Carlo Sig. (2-tailed)	Sig.		.002	.002	.841
	99% ConfidenceInterval	LowerBound	.001	.001	.832
		UpperBound	.002	.002	.851
Monte Carlo Sig. (1-tailed)	Sig.		.001	.001	.417
	99% ConfidenceInterval	LowerBound	.000	.000	.404
		UpperBound	.002	.002	.430

Tabla II-9. Resultados del análisis estadístico al comparar C_{VOM1} vs C_{VOM2}

			C_{VOM1} vs C_{VOM2}
Z			-2.497(a)
Asymp. Sig. (2-tailed)			.013
Monte Carlo Sig. (2-tailed)	Sig.		.009
	99% ConfidenceInterval	LowerBound	.007
		UpperBound	.012
Monte Carlo Sig. (1-tailed)	Sig.		.004
	99% ConfidenceInterval	LowerBound	.003
		UpperBound	.006

2.2.9 Operador de Limpieza

Durante el proceso de experimentación se observó que incluso luego de aplicar el operador de mejoramiento de sentencias, los resúmenes obtenidos contenían una gran cantidad de sentencias con valores de T = 0 las cuales coincidían en tener un tamaño (cantidad de predicados difusos) de calificador *R* grande (mayor que seis). El estudio realizado demostró que resúmenes obtenidos poseían un 75 % de las sentencias con un grado de validez mayor que 0.90 y alrededor de un 20% de sentencias con T = 0 lo cual tiende a deteriorar la calidad del resumen lingüístico en general.

Lo anterior está motivado por el hecho de que este tipo de sentencias es muy difícil que evolucionen hacia soluciones mejores incluso con la aplicación del operador de mejoramiento. Las características de la base de casos de *creep* tienen una influencia directa en esta situación pues se observa que para el tipo de proposiciones descritas, en la mayoría de los casos existen pocos o ningún caso que las fundamente.

A partir de esta situación, se decidió incluir en el modelo de AG propuesto, un nuevo operador llamado Operador de Limpieza (de malas sentencias). Este operador basa su esencia en sustituir las sentencias, que a pesar de haber recibido el tratamiento del operador de mejoramiento mantienen su validez igual a 0, por nuevas sentencias generadas aleatoriamente. Estas nuevas sentencias tienen mayor probabilidad de mejorar que las sentencias anteriores.

Para su inclusión en el modelo se estudiaron variantes que definían la frecuencia de aplicación. Así se analizaron dos variantes:

Variante I (VOL1). Se aplica con una frecuencia que se determina a partir del 20% de la cantidad de generaciones totales del proceso. Por ejemplo para un número de generaciones igual a 100, la frecuencia de aplicación sería de 20 y por tanto el operador se aplicaría en las generaciones 20, 40, 60 y 80.

Variante II (VOL2). Se aplica con una frecuencia que se determina a partir del 10% de la cantidad de generaciones totales del proceso. Por ejemplo para un número de generaciones igual

a 100, la frecuencia de aplicación sería de 10 y por tanto el operador se aplicaría en las generaciones 10, 20, 30, 40, 50, 60, 70, 80, 90

2.2.9.1 Definición de la variante de aplicación del operador de limpieza

Es necesario aclarar que las variantes propuestas fueron desarrolladas a partir de la variante obtenida anteriormente aplicando el operador de mejoramiento según VOM1. Es por ello que para definir la mejor variante de aplicación del operador de limpieza de sentencias a utilizar en el modelo y comprobar su comportamiento sobre dicho modelo, se realizó la comparación de ambas variantes entre sí y estas a su vez con VOM1. Los resultados de los experimentos y el análisis estadístico se muestran en las tablas Tabla II-10, Tabla II-11, Tabla II-12, Tabla II-13.

La Tabla II-10 refiere la comparación entre las tres variantes respecto al valor promedio de la función de aptitud (A) del mejor resumen lingüístico obtenido en cada corrida del modelo. Como se evidencia VOL2 presenta un mejor comportamiento de A que VOL1 (A_{VOL2} = 0.6277 vs A_{VOL1} = 0.6241) aunque según los resultados del análisis estadístico mostrado en la Tabla II-12 estas diferencias no son significativas. Sin embargo al analizar ambas variantes (VOL1, VOL2) respecto a VOM1 se observa que las primeras muestran un comportamiento muy superior a esta, resultando las diferencias significativamente superiores de acuerdo al análisis estadístico también mostrado en la Tabla II-12.

Al realizar la comparación respecto a la cantidad promedio de evaluaciones de *T* (denotado como C) (Tabla II-11) se observa que VOL2 presenta un mejor comportamiento que VOL1 (C_{VOL2} = 84969 vs C_{VOL1} = 89804) existiendo una diferencia significativa según el análisis estadístico expuesto en la Tabla II-13. La comparación sobre este indicador de VOL1 y VOL2 respecto a VOM1 demuestra que las dos primeras poseen un valor evidentemente mejor (C_{VOM1} = 95653) lo cual demuestra la utilidad de este operador dentro del modelo propuesto.

A partir del análisis de los resultados se tomó la decisión de incorporar al modelo, la variante VOL2 teniendo en cuenta que mostró mejores resultados en ambos indicadores (A y C) existiendo diferencias significativas en los resultados para el indicador C.

Tabla II-10. Comparación de las variantes de aplicación del operador de limpieza en relación a los valores de la función de aptitud (A)

	N	Mean	Std. Deviation	Minimum	Maximum
VOM1	10	,5207	,04065	,47	,60
VOL1	10	,6241	,02248	,58	,67
VOL2	10	,6277	,02486	,60	,66

Tabla II-11. Comparación de las variantes de aplicación del operador de limpieza en relación a la cantidad de evaluaciones de T (C)

	N	Mean	Std. Deviation	Minimum	Maximum
VOM1	10	95653,1000	3227,61045	89013,00	100096,00
VOL1	10	89804,1000	3076,07313	85397,00	93884,00
VOL2	10	84969,9000	2807,81601	80455,00	89021,00

Tabla II-12. Resultados del análisis estadístico al comparar A_{VOM1} vs A_{VOL1}, A_{VOM1} vs A_{VOL2} y A_{VOL1} vs A_{VOL2}

			A_{VOM1} vs A_{VOL1}	A_{VOM1} vs A_{VOL2}	A_{VOL1} vs A_{VOL2}
Z			-2,701(a)	-2,803(a)	-,459(a)
Asymp. Sig. (2-tailed)			,007	,005	,646
Monte Carlo Sig. (2-tailed)	Sig.		,005	,003	,691
	99% ConfidenceInterval	LowerBound	,003	,001	,679
		UpperBound	,007	,004	,703
Monte Carlo Sig. (1-tailed)	Sig.		,002	,001	,347
	99% ConfidenceInterval	LowerBound	,001	,000	,334
		UpperBound	,003	,002	,359

Tabla II-13. Resultados del análisis estadístico al comparar CVOM1 vs CVOL1, CVOM1 vs CVOL2 y CVOL1 vs CVOL2

			C_{VOM1} vs C_{VOL1}	C_{VOM1} vs C_{VOL2}	C_{VOL1} vs C_{VOL2}
Z			-2,803(b)	-2,803(b)	-2,497(b)
Asymp. Sig. (2-tailed)			,005	,005	,013
Monte Carlo Sig. (2-tailed)	Sig.		,003	,003	,012
	99% ConfidenceIntervall	LowerBound	,001	,001	,009
		UpperBound	,004	,004	,014
Monte Carlo Sig. (1-tailed)	Sig.		,002	,002	,007
	99% ConfidenceInterval	LowerBound	,001	,001	,005
		UpperBound	,002	,002	,009

2.2.10 Modelo final de algoritmo genético. Consideraciones finales

El modelo final de algoritmo genético se describe a partir del esquema representado en la Fig. II-5.

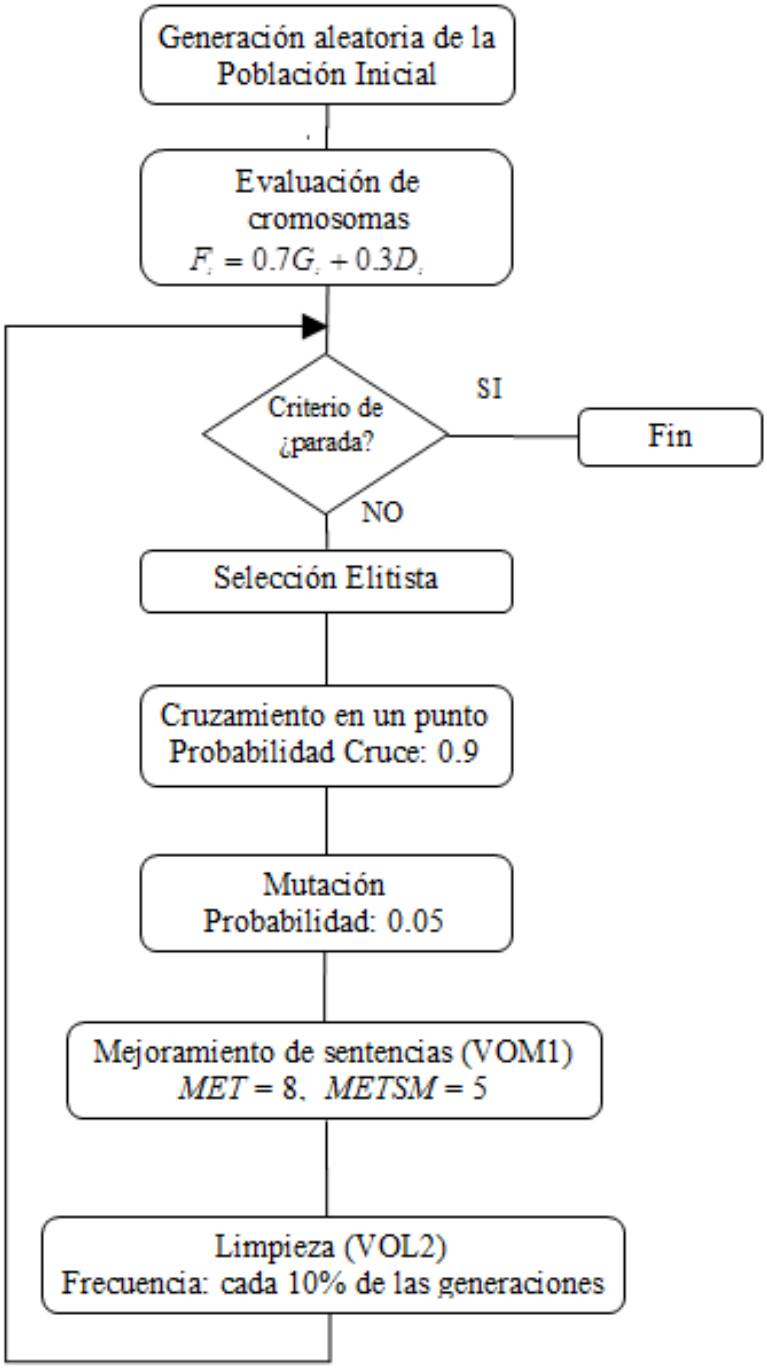


Fig. II-5. Diagrama del modelo propuesto final

Valores de los parámetros del modelo

Tabla II-14. Parámetros del modelo de AG

Parámetros		**Valores**
Cantidad de generaciones	(*CG*)	100
Tamaño de la población	(*TP*)	100
Tamaño del Cromosoma	(*TC*)	30
Probabilidad de Cruzamiento	(p_c)	0.9
Probabilidad de Mutación	(p_m)	0.05
Cantidad de Atributos de la base de casos		22
Peso del término *G* en función aptitud	($\boldsymbol{m_g}$)	0.7
Peso del término *D* en función aptitud	($\boldsymbol{m_d}$)	0.3
Parámetros del operador de mejoramiento		
Cantidad máxima de evaluaciones de *T*	(*MET*)	8
MET sin mejoramiento	(*METSM*)	5
Valor de deseado de T	(*TD*)	0.95
Parámetros del operador de limpieza		
Frecuencia de aplicación	(*FA*)	10% de G

Consideraciones finales

A partir del modelo de AG propuesto se realizó una última experimentación con vistas a analizar su nivel de escalabilidad, o sea, su comportamiento al incrementar el tamaño del cromosoma (*TC*) lo cual es equivalente a aumentar la cantidad de proposiciones deseadas para el resumen. Los resultados de estos experimentos se muestran en la Tabla II-15. Como se observa, al incrementar *TC* desde 30 hasta 100 el valor promedio de *T* del cromosoma (nivel de calidad del resumen lingüístico) no se deteriora, alcanzado como mínimo el valor de 0.88 que puede ser considerado como un valor de validez bueno incluso para otros tipos de problema; en relación a la cantidad de evaluaciones se observa un aumento del mismo lo cual es consistente y no desproporcional al incremento del espacio de búsqueda generado por el incremento de *TC*.

Otra observación importante es la relativa a la proporción de proposiciones encontradas con cuantificador '*mayoría*' en relación a *TC*. Este indicador se muestra en la última columna de la tabla. Como se puede observar su comportamiento es similar hasta un tamaño de cromosoma igual a 45 lo cual reafirma la consistencia y escalabilidad del modelo, es decir, el mismo es capaz de encontrar una cantidad de soluciones buenas proporcional al espacio de búsqueda explorado. Al analizar la última fila se observa un deterioro de la proporción lo cual pudiera estar influenciado por dos motivos:

- la cantidad de soluciones con las características descritas está cercana a la máxima existente en la base de casos explorada ó
- el modelo realmente no es escalable a partir de un tamaño de cromosoma igual a 100

Con los elementos disponibles no es posible afirmar ninguna de estas suposiciones, para ello sería necesario realizar una búsqueda exhaustiva de todo el espacio de soluciones del problema y comprobar si realmente se logró el descubrimiento de todas las posibles proposiciones con el valor de *T* referido en la tabla y que tengan el cuantificador '*la mayoría*'.

Como conclusión se puede plantear que el modelo propuesto resulta escalable sobre los datos de *creep*, es decir, es capaz de aumentar la cantidad de proposiciones lingüísticas a obtener, manteniendo la calidad de las mismas con un aumento razonable del espacio de búsqueda revisado.

Tabla II-15. Experimentos que demuestran la escalabilidad del modelo final de AG y LDS

Tamaño del Cromosoma	Promedio de T	Evaluaciones de T	Promedio de Aptitud	CPM/TC*
30	0.9650	84940.8	0.6277	0.150
45	0.9319	126594.7	0.603616	0.148
60	0.9110	191609.7	0.5551	0.138
100	0.8818	330988.7	0.4576	0.082

* define la relación entre la cantidad promedio de sentencias obtenidas en el resumen con cuantificador '*la mayoría*' y el tamaño del cromosoma

CAPÍTULO III. ASPECTOS DEL DISEÑO Y LA IMPLEMENTACIÓN

En este capítulo se aborda lo referente a la implementación de la herramienta CGLS (*Creep Genetic Linguistic Summarizer*). Esta herramienta, obtiene resúmenes lingüísticos para bases de casos del problema *creep* utilizando el modelo de AG propuesto (ver 2.2.10). Vale destacar que la herramienta obtiene resúmenes con proposiciones lingüísticos que siguen la estructura propuesta la protoforma 5 (ver 1.1.2), es decir, el usuario no especifica la estructura de los resúmenes a obtener sino que la herramienta debe ser capaz de generar un grupo de resúmenes con una estructura no predefinida.

La implementación de esta herramienta como se muestra en la Fig. III-1 consta de tres partes funcionales: la implementación del AG mediante la librería JGAP, la del modelado difuzzo mediante las clases del software Xfuzzy y la implementación referente a la técnica LDS y los proposiciones linguísticas. Luego en el epígrafe 3.4 se hace referencia al manual de usuario de la herramienta.

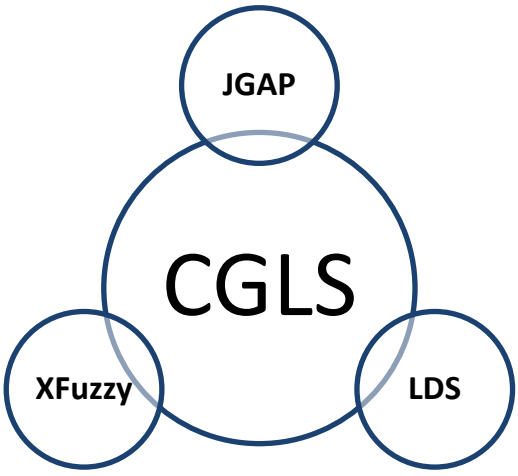


Fig. III-1. Diseño funcional de la herramienta CGLS

3.1 Implementación del modelo de AG con JGAP

Como se plantea en 1.3.1, la librería java JGAP es componente de programación utilizado en nuestro sistema para modelar todo el engranaje del AG. A continuación se describe la mayor parte del diagrama de clases de la herramienta, dónde las clases resaltadas en rojo se corresponden con las clases implementadas dentro del sistema que heredan o implementan

interfaces de JGAP. Para la total comprensión, la implementación y la descripción de los diagramas se dividen en tres partes que corresponden con los conceptos fundamentales dentro de los AG:

- Implementación de cromosomas y genes.
- Implementación de la población, la configuración y la función de aptitud
- Implementación del método de selección y los operadores genéticos

3.1.1 Implementación de cromosomas y genes

En la Fig. III-2 se muestran las clases del modelo de AG propuesto relacionadas con los cromosomas y los genes que los conforman. La clase *SummaryChromosome* que hereda de la clase JGAP *BaseChromosome* contiene una agregación de *SentenceGene* y esta a su vez está asociada a la clase *Sentence.* Así un alelo (clase *Sentence*) es una instancia específica de una sentencia. El cromosoma es el encargado de llamar a la función de aptitud y calcularle este valor al cromosoma; además sobrescribe el método *toString ()* que devuelve la representación lingüística (en forma de un conjunto de sentencias o proposiciones) de un individuo.

En la clase *SentenceGene* a la sentencia se le asigna un valor aleatorio a través del método *setToRandomValue()* y se le puede aplicar una mutación a través de *applyMutation(double d, int i)* dónde d y i son valores aleatorios generados en el proceso de mutación.

Como se explica en el apartado 2.2.2, en el modelo implementado una sentencia contiene referencias a los componentes de una proposición lingüística, por lo que en la implementación la clase *Sentence* presenta atributos que describen al sumarizador, al calificador, al cuantificador y a la validez T. *Sentence* además calcula su validez en una llamada al método computeValidity(Engine e) dónde *Engine* es la clase que permite el cálculo de la validez de una sentencia (ver 3.3). El método *toString()* de *Sentence* expresa la sentencia en la forma de una oración del lenguaje natural. La clase *VariablesPool,* de vital importancia en la implementación, contiene referencias a todo el conjunto de variables difusas que intervienen en el proceso.

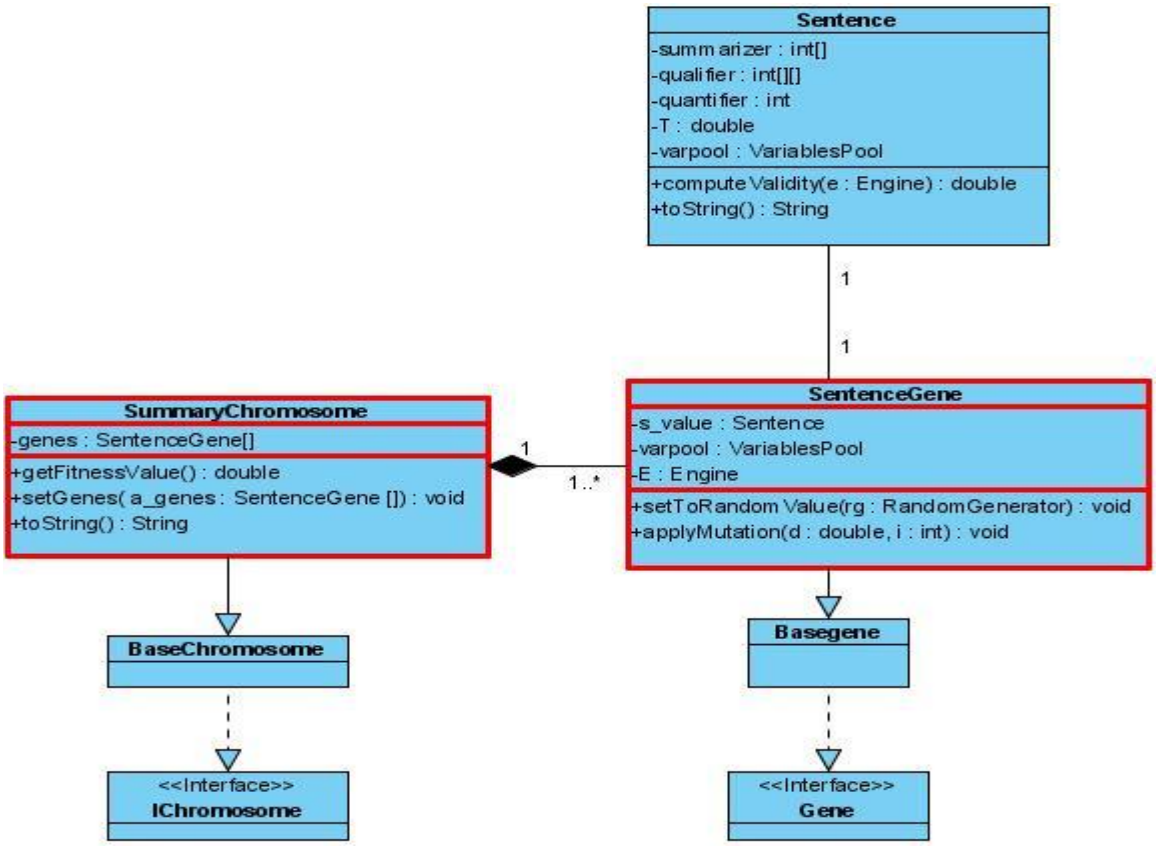


Fig. III-2. Diseño de clases de cromosomas y genes

3.1.2 Implementación de la población, la configuración y la función de aptitud

La clase *Population* del JGAP contiene todos los individuos (cromosomas) del AG. La clase *LDFSGenotype* se encarga de inicializar aleatoriamente la población y de evolucionarla llamando a la selección y a los distintos operadores genéticos. *LDFSGenotype* también se encarga de devolver los mejores cromosomas a través del método *getFittestChromosomes*(int n), dónde n referencia la cantidad de mejores cromosomas a devolver.

LDFSConfiguration que hereda de *Configuration* es la clase rectora de todo el experimento. Es aquí donde se crea la función de aptitud, el operador de selección, los operadores genéticos y dónde se hace referencia a la población actual del experimento. *LDFSFitnessFunction* es creada en el constructor de *LDFSConfiguration* y es la responsable de evaluar un cromosoma siguiendo la función descrita en el epígrafe 2.2.4.

3.1.3 Implementación del método de selección y los operadores genéticos

Como se mostró anteriormente en la clase *LDFSConfiguration* se crean los operadores genéticos a utilizar en el algoritmo. El operador de selección elitista descrito en el acápite 2.2.5.1 es implementada a través de la clase JGAP *BestChromosomesSelector*.

Las clases *LDFSCrossoverOperator*, *LDFSMutationOpertor*, *LDFSImprovementOperator* y, *LDFSCleanerOpertor* que implementan los operadores de Cruzamiento, Mutación, Mejoramiento de sentencias y Operador de limpieza respectivamente, se implementan heredando de la clase JGAP *BaseGeneticOperator*. El operador de limpieza *LDFSCleanerOperator* se relaciona con la clase *ImprovementRateCalculator* que se encarga de calcular la cantidad de mejores individuos a los cuales se les aplica el mejoramiento según la función descrita en el epígrafe 2.2.8.

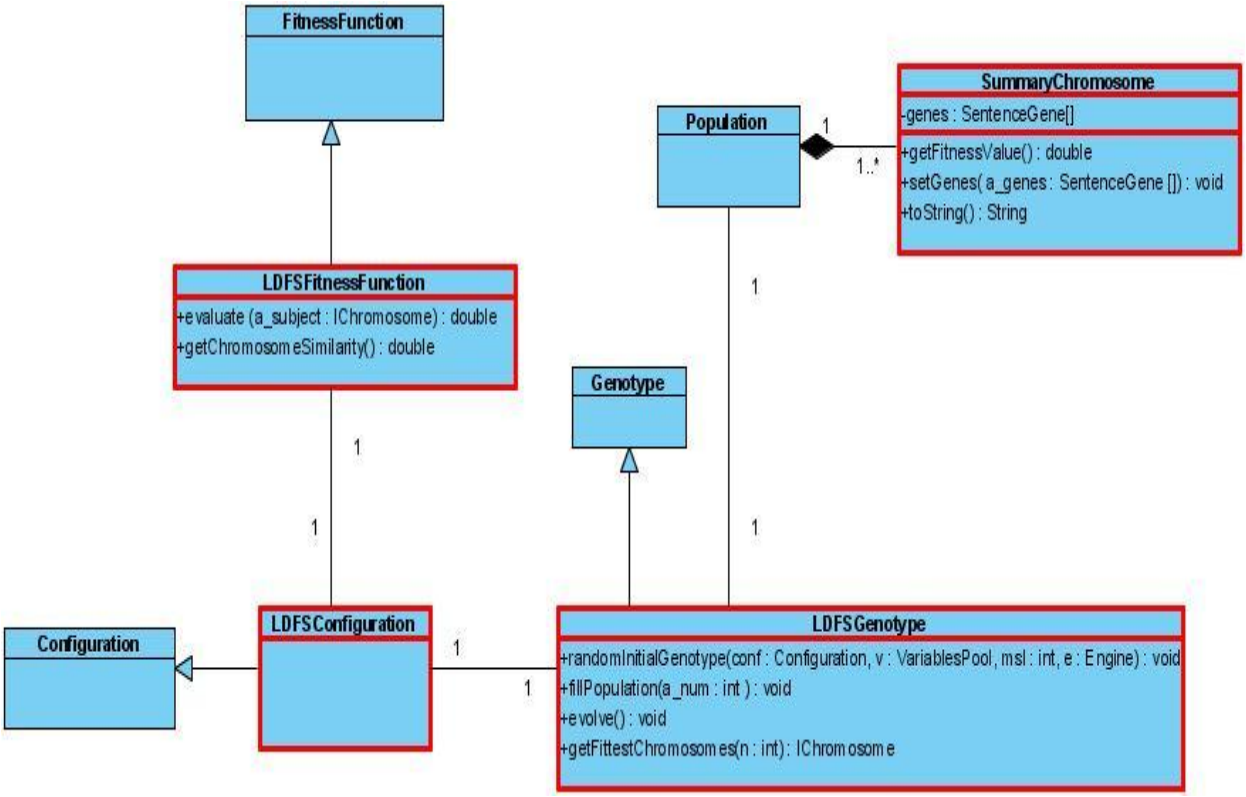


Fig. III-3. Diseño de las clases de la población, la configuración y la función de aptitud

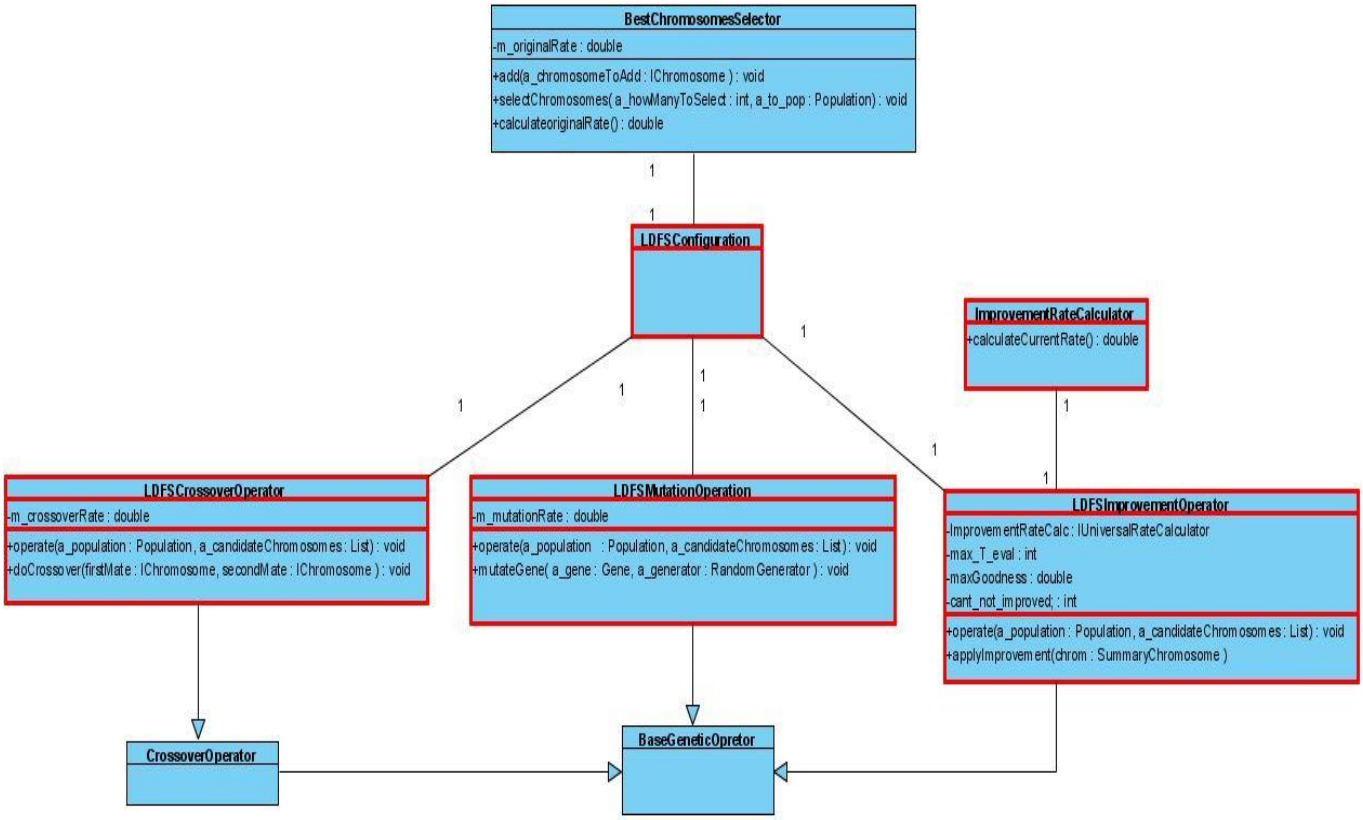


Fig. III-4. Diseño de clases de los operadores genéticos y la selección

3.2 Implementación del modelado y edición difusa

Cómo se ha visto anteriormente la técnica LDS se ha enfocado desde el punto de vista de la lógica difusa donde los atributos del problema son modelados a partir de variables difusas conteniendo cada una varios términos lingüísticos. Una sentencia está formada por la concatenación, mediante operadores difusos, de predicados difusos (variable difusa con un término lingüístico, ejemplo: '*salario alto*'). Por lo tanto la herramienta GCLS debe ser capaz de permitirle al usuario modelar y editar tanto los operadores como las variables difusas. Estas funcionalidades se realizan a través de las clases provenientes del software Xfuzzy, el cual es un entorno compuesto por un conjunto de herramientas que facilitan el diseño, la verificación y la síntesis de sistemas difusos (ver 1.3.2).

Estas clases de Xfuzzy permiten la modelación de estructuras de datos para las funciones de pertenencia, los operadores, y el universo del discurso.

Como se muestra en Fig. III-5, la clase *Specification* contiene un conjunto de "tipos lingüísticos' especificados mediante la clase *Type* que a su vez se asocia a un universo de discurso (mínimo, máximo, carnalidad). La clase Type se asocia de forma agregacional a la clase abstracta *ParamMenFunc* que igualmente tiene un universo de discurso que la describe y un conjunto de parámetros, y de la cual heredan todos los tipos de funciones de pertenencia existentes (isósceles, de campana, trapezoidal…). Estas funciones son las encargadas de calcular la pertenencia de un caso a un conjunto difuso a través del método *compute()*. Este método es de vital importancia en el sistema, cada método *compute()* se implementa atendiendo al tipo de función de pertenencia y los parámetros de esta función.

Specification además cuenta con un conjunto de operadores difusos a través de la clase *OperatorSet* que hace referencia a cada uno de los operadores difusos ya sean binarios (*and, or,imp,also*) o unarios (*not, very, more or less, slightly*). De las clases *Binary* y *Unary* heredan las clases que calculan el resultado de aplicar el operador siguiendo cada una de los métodos establecidos: la clase *xfl_binary_min* sigue el mínimo en las funciones binarias, *xfl_binary_max* sigue el máximo…

Como se explica en 2.1 atendiendo a las características del modelado difuso de *creep* en el sistema solo se utiliza el operador binario *and* para la concatenación de los términos lingüísticos de la variable cuantificadora.

3.2.1.1 El formato xfl

La clase *Specification* además, contiene otro método importante: *toXfl()*. Este método se encarga de llamar a las demás métodos *toXfl()* de las funciones de pertenencia y los operadores difusos creando un formato común que permite tener una completa descripción de la configuración del sistema difuso creado. Este formato, llamado xfl por los creadores de Xfuzzy es salvado posteriormente por otras clases del sistema en un archivo texto .xfl que permite ser cargado y después parseado, creando un nuevo sistema con las funciones de pertenencia y operadores difusos descritos en el archivo xfl.

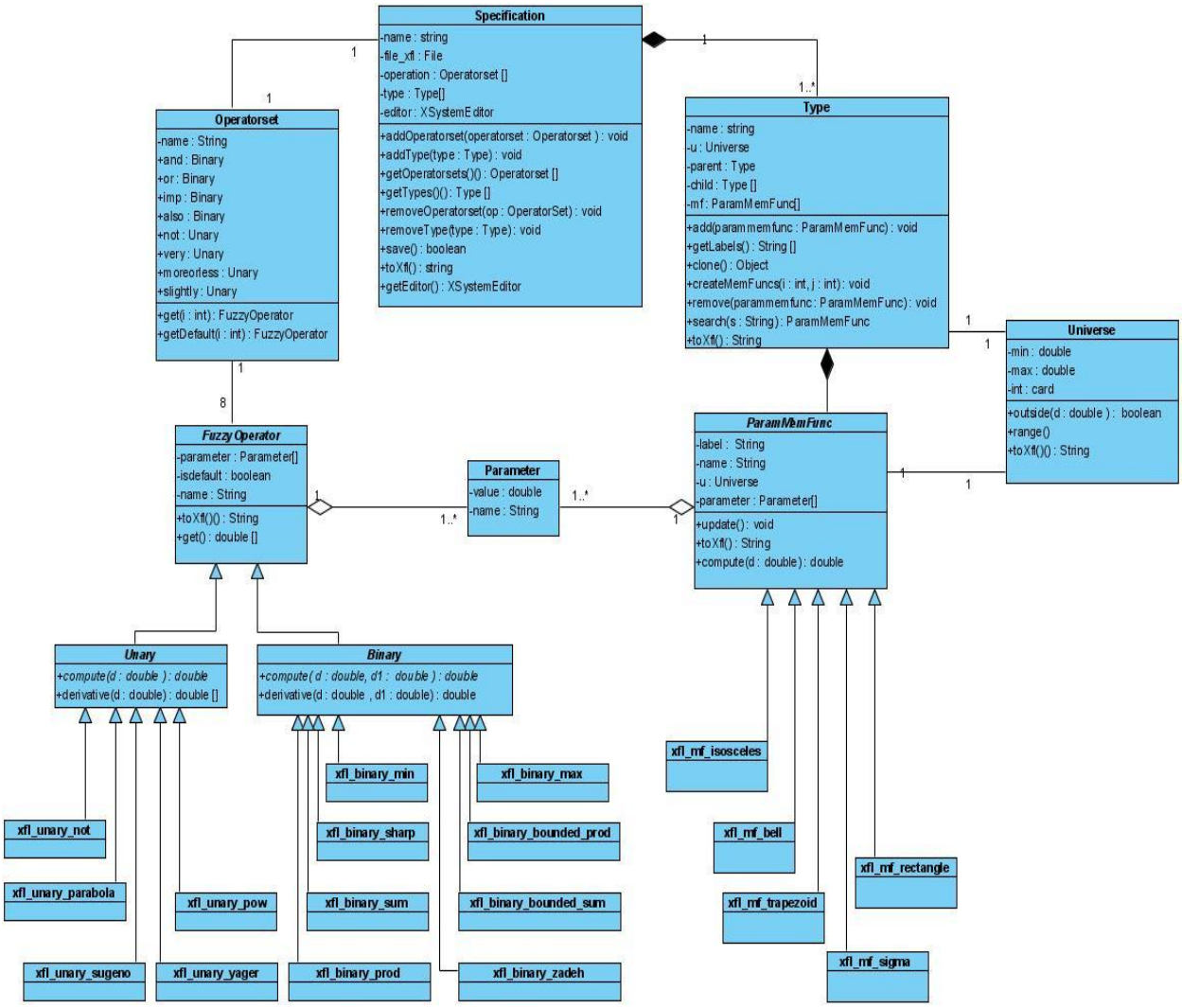


Fig. III-5. Clases del modelado difuso

3.3 Clases de la técnica LDS e integración de las partes funcionales de la herramienta

La clase *Engine* constituye funcionalmente la referencia a la técnica LDS ya contiene el método *computeValidityOfSentence(Sentence s)* que implementa las funciones de cálculo de validez *T* Esta clase constituye el eslabón que relaciona las tres partes funcionales de la herramienta, ya que ella está relacionada con la clase *SentenceGene* (que la utiliza para el cálculo de *T)*, y con la modelación borrosa, ya que presenta una agregación de *FuzzyVariable* y una relación de asociación con la clase *Specification* (ver 3.2).

La clase *FuzzyVariable* hace referencia al índice y al nombre del atributo de la base de casos seleccionada para extraerle el resumen lingüístico, además, contiene una referencia a *Type* que a su vez se relaciona con todas las funciones de pertenencia que se desea tenga esta variable fuzzy.

La clase *Engine* también se asocia a *Specification* ya que esta contiene el conjunto de operadores difusos necesarios para el cálculo de *T* de las sentencias.

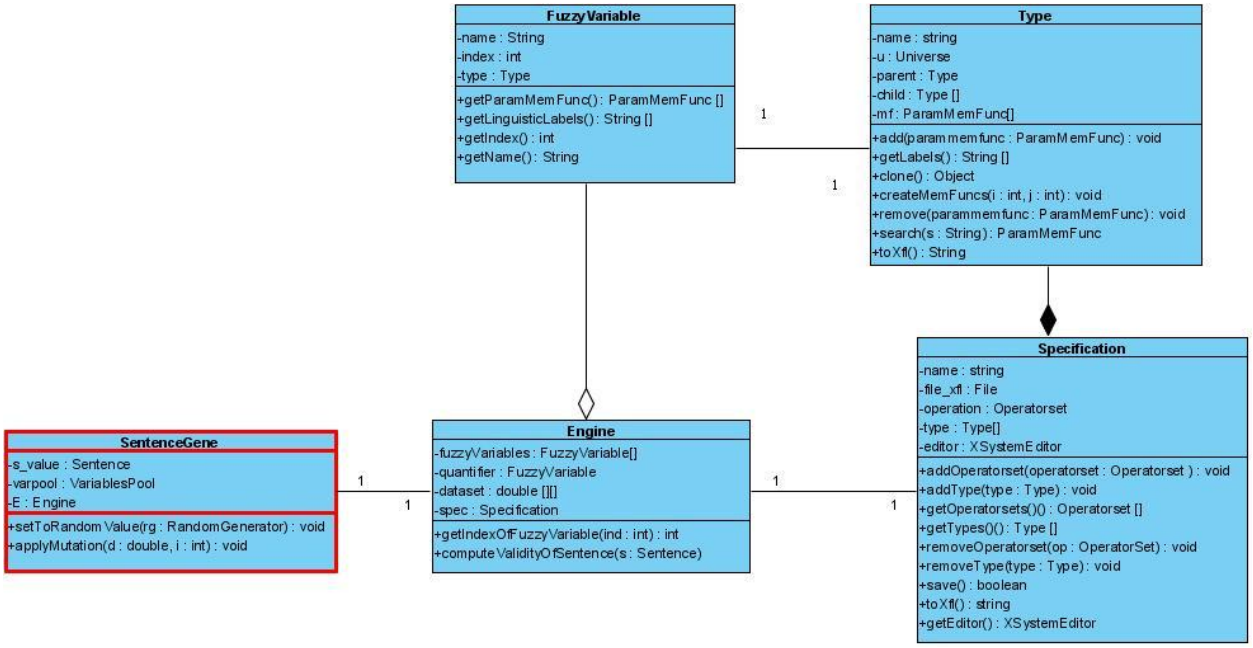


Fig. III-6. Diagrama de clases de la integración de las partes funcionales de la herramienta

3.4 Manual de Usuario

La herramienta CGLS (*Creep Genetic Linguistic Summarizer*), programada en el lenguaje java, presenta una interfaz sencilla y de fácil manejo. La primera pantalla en aparecer cuando se ejecuta la herramienta es la ventana principal (ver Fig. III-7) dónde en el área principal se muestran la lista de acciones realizada por el usuario.

Luego se debe centrar la atención en el modelado difuso, entiéndase la creación y edición de los operadores y tipos difusos.

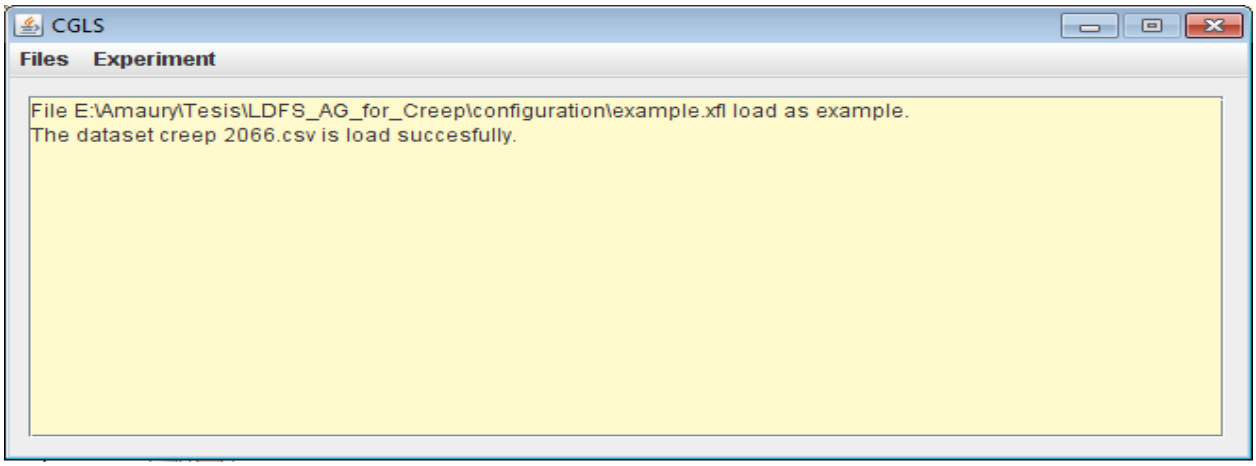


Fig. III-7. Ventana principal de la aplicación

3.4.1 Manual del modelado borroso y el trabajo con la base de casos

El modelado difuso se realiza en la herramienta a través de interfaces y ventanas obtenidas del software Xfuzzy. Estas clases brindan una interacción muy fácil y asequible al usuario.

Manual del modelado difuso:

1. Cargar un archivo .xfl que contiene todos los operadores y tipos fuzzy a utilizar en el experimento. Ver Fig. III-8. Cargar xfl. Vale destacar que el sistema se encarga de cargar un archivo xfl por defecto con los operadores y tipos difusos para las bases de casos *creep*.

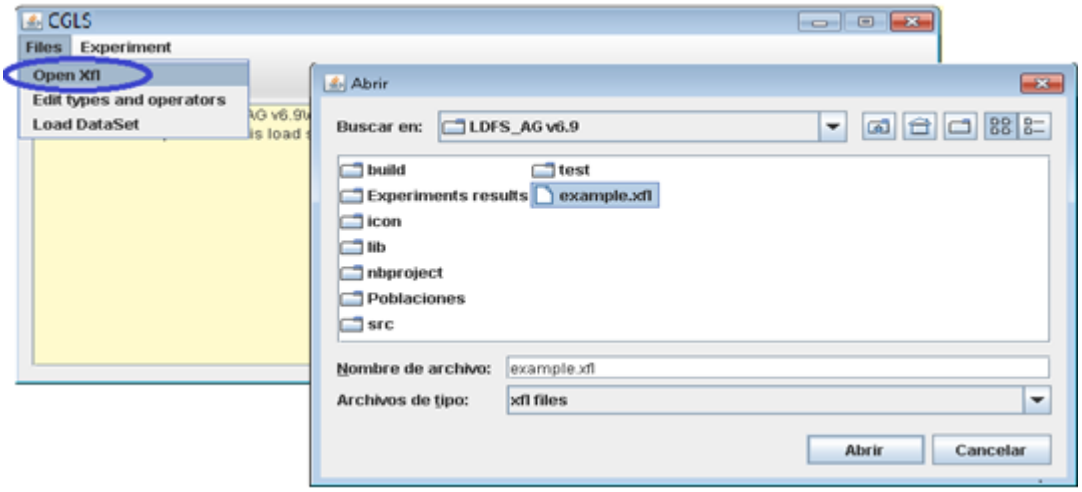


Fig. III-8. Cargar xfl

2. Editar el operador difuso a través de la interfaz Xfedit accediendo al menú *Edit types and operators*. En la Fig. III-9 se ilustra el proceso dónde se edita el operador *and* utilizado en sistema. En el paso 3 se escoge que función se utiliza en el cálculo del operador *and*, dónde xfl.min corresponde con la función mínimo, xfl.max con la función máximo y xfl.prod con la función producto.

3. En la Fig. III-10 se muestra como editar los tipos difusos. Se accede a la interfaz Xfedit, se selecciona un tipo y podemos posteriormente seleccionar una función de pertenencia (paso 3) y editarla cambiando sus parámetros.

   Para añadir un nuevo tipo seguimos los pasos numerados de la Fig. III-11. En el apartado *Type Creation* creamos el tipo difuso especificando los parámetros mínimo, máximo, y cardinalidad, la cantidad de funciones de pertenencia que se crearán. En la parte derecha de la ventana se selecciona el tipo de función a utilizar (trapezoidal, campana…).

4. Cargar la base de casos *creep* (Ver Fig. III-12). Este archivo debe estar en formato CSV (delimitado por comas).

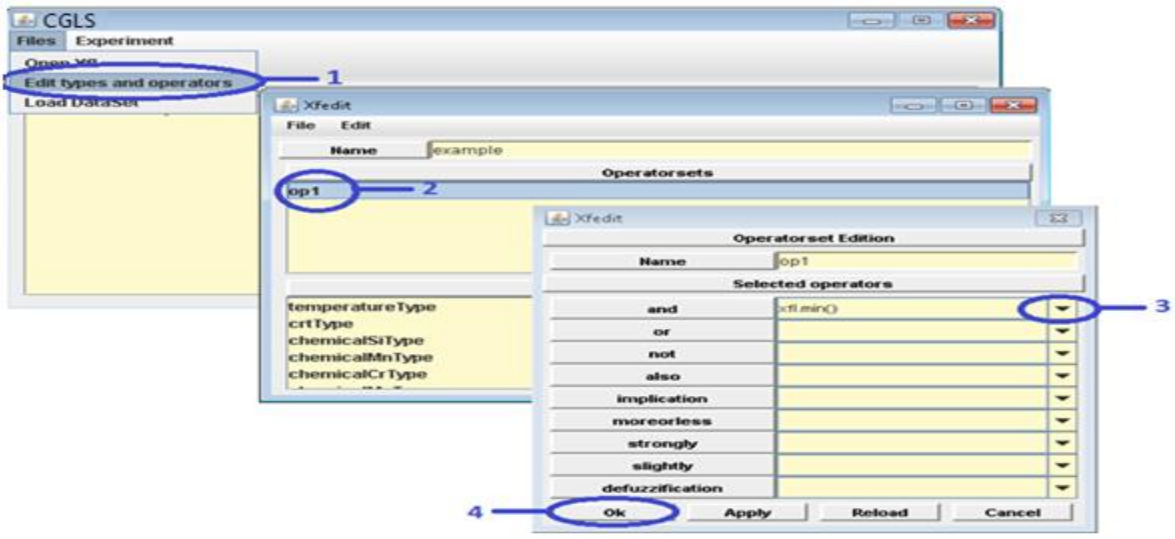


Fig. III-9. Editar operadores difusos

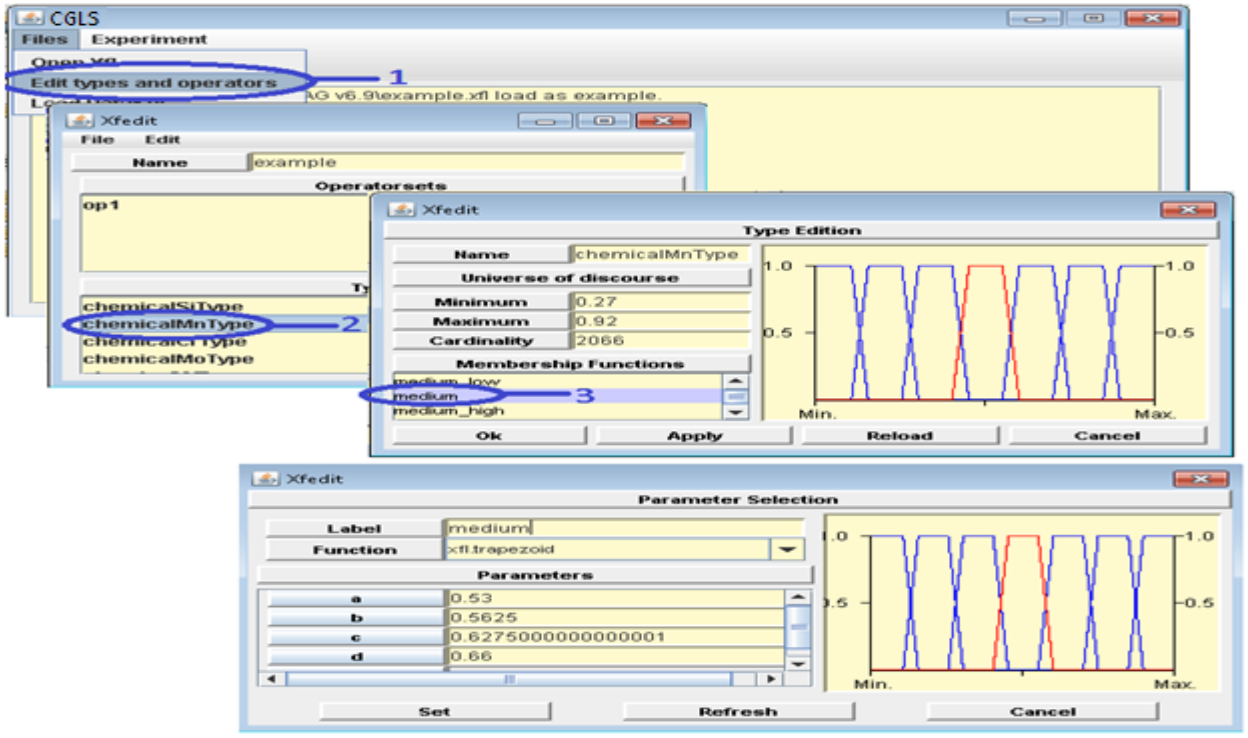


Fig. III-10. Edición de tipos difusos

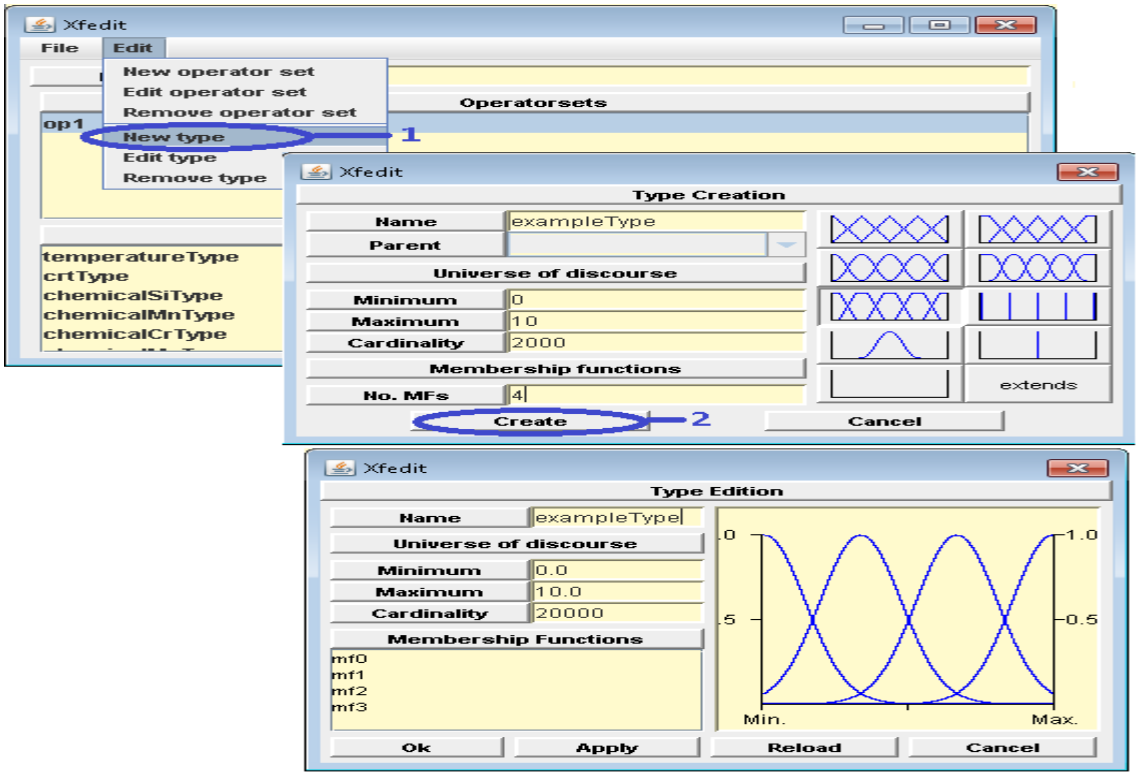


Fig. III-11. Creación de tipos difusos

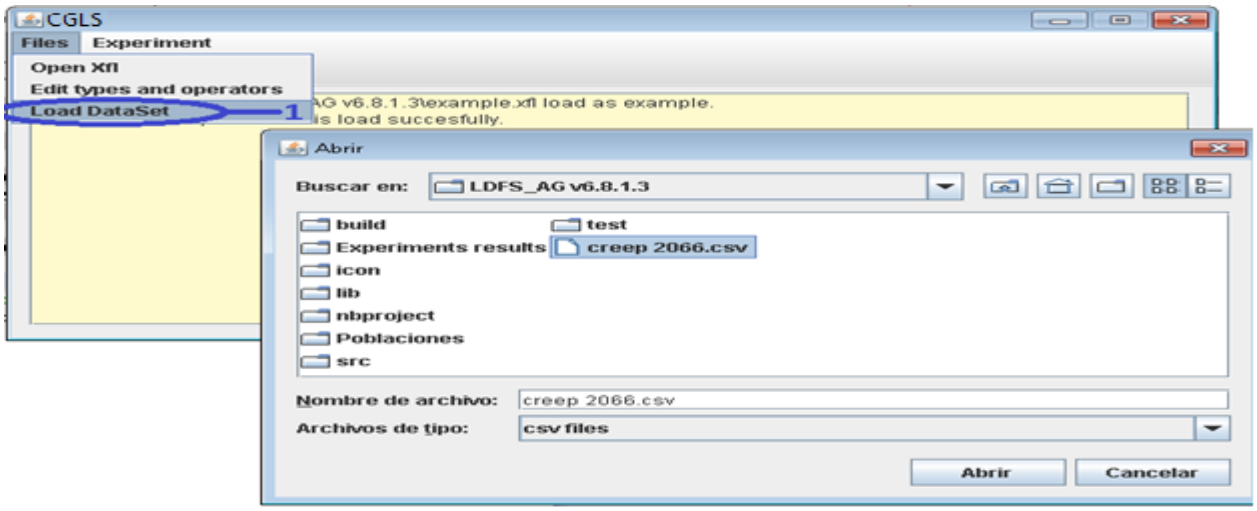


Fig. III-12. Cargar la base de casos

3.4.2 Manual de inicialización y corrida del experimento

Después de haber realizado el modelado difuso, se habilita automáticamente el menú *Experiment* de la ventana principal. Luego se continúa con la inicialización y ejecución del experimento.

1. Se inicializan las variables difusas que intervendrán en el experimento asignándole a los atributos presentes en la base de casos un tipo difuso correspondiente accediendo al menú *Edit Variables* (ver Fig. III-13). Ya en la ventana *Variables's editor* se puede seleccionar un tipo difuso (ver paso 2) y asignárselo al atributo creando así una nueva variable difusa, o bien se puede cargar un archivo .cev (*configuration of the experiment's variables*) previamente guardado con una selección de variables difusas (ver paso 3). Todas las referencias a las variables seleccionadas para el experimento aparecen en la parte inferior de la ventana. Después de elegir todas las variables difusas que intervendrán en el experimento, se selecciona el botón *Ok*. Vale destacar que a los atributos que no se le asignen tipos difusos no estarán presentes en el experimento como variables difusas. Para salvar una configuración de variables difusas, después de crear las variables difusas acceder al menú *Save* de *Experiment* y en selector de archivo correspondiente escoger el directorio y el nombre del archivo.

2. A través del paso 1 se accede a la ventana *Begin the experiment* (ver Fig. III-14) dónde se designan las variables para el cuantificador *Q*, calificador *R* y el sumarizador *S (*en la presente investigación siempre sería la variable *creep).* Mediante el paso 2 se pueden

seleccionar todas las posibles variables para el calificador. En la parte inferior izquierda de la ventana se definen parámetros del experimento como: la cantidad de proposiciones o sentencias por resumen (*Sentences by summary*) y la cantidad de experimentos a realizar (*Number of experiments*). En la parte inferior derecha se definen los parámetros del AG: cantidad de generaciones (*Number of generation*) y tamaño de la población (*Population size*). Previamente definidos estos parámetros se selecciona el botón *Begin the experiment* para la ejecución del experimento.

3. Después de mostrado el progreso del experimento (ver Fig. III-15) se muestran los resultados del experimento a través de la ventana *Experiment's results*. Los resultados muestran datos del experimento como el tamaño de la población, el tamaño del cromosoma y la cantidad de generaciones así como las probabilidades de cruzamiento y mutación. Los mejores diez resúmenes obtenidos se muestran evidenciando el grado de aptitud, el promedio de T y el grado de diversidad de los resúmenes. Estos resultados pueden ser salvados en formato de archivo texto para un posterior análisis de los resultados.

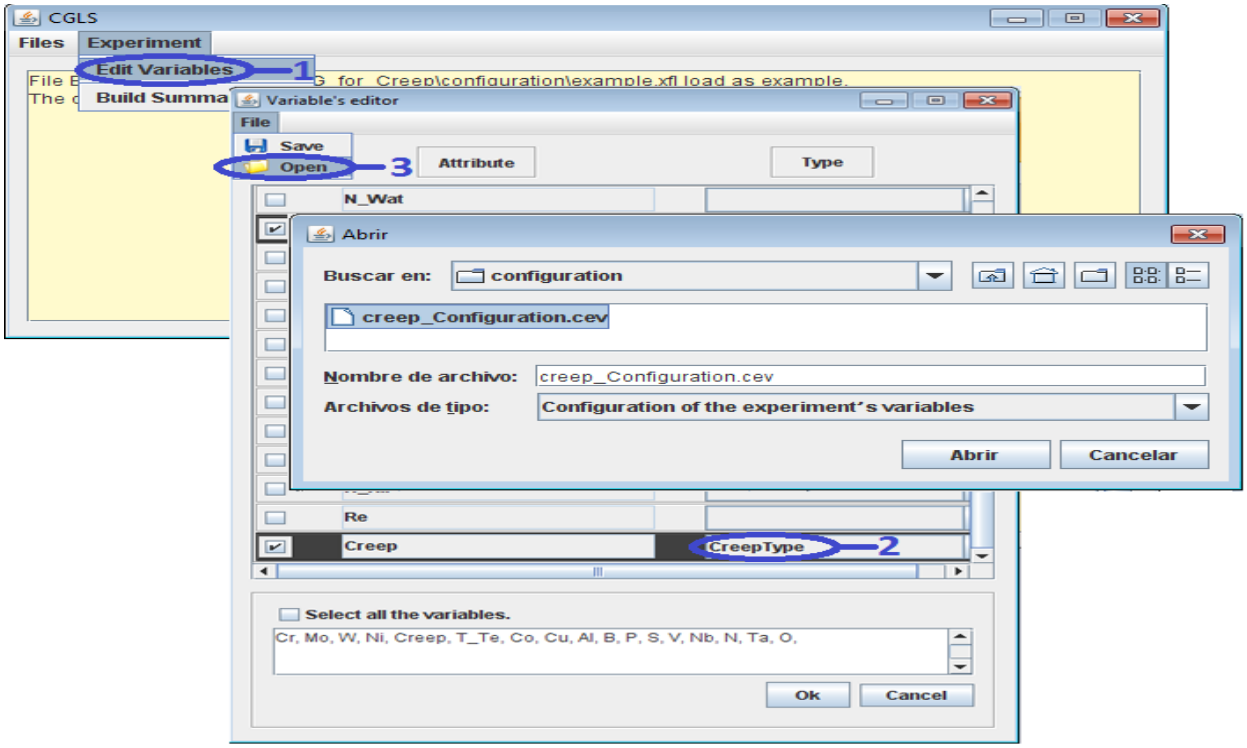


Fig. III-13. Editar variables

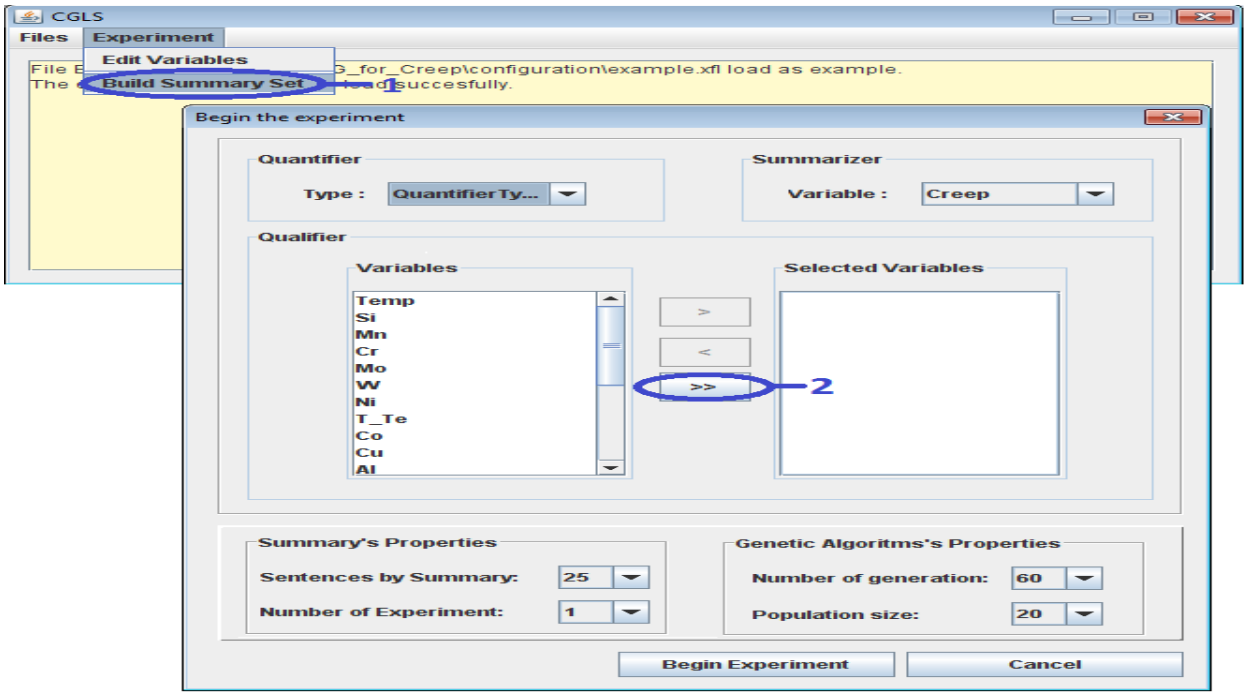


Fig. III-14. Inicializar el experimento

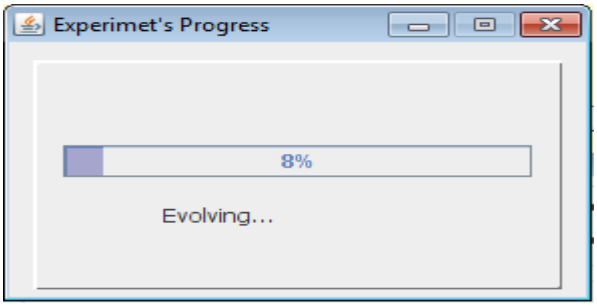


Fig. III-15. Progreso del Experimento

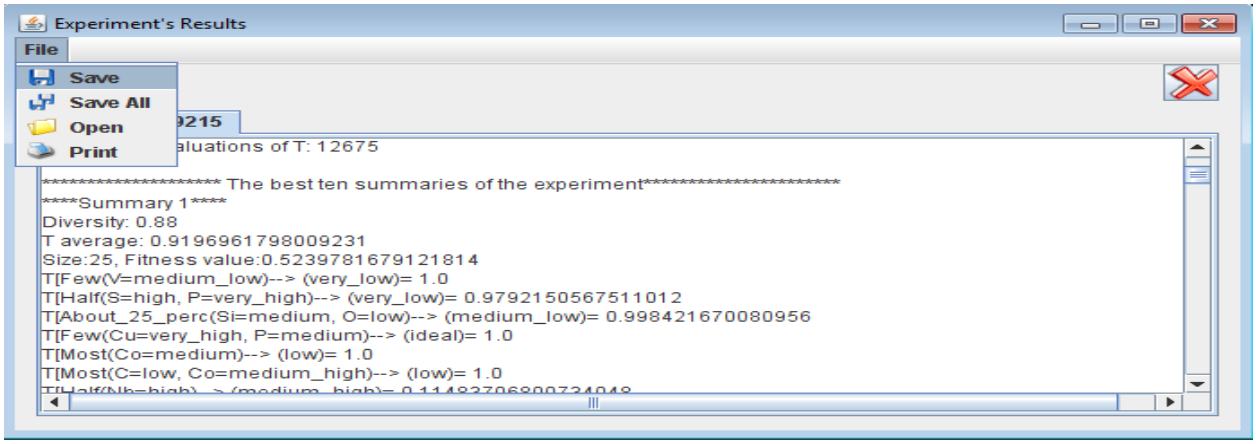


Fig. III-16. Ventana de resultados de la herramienta CGLD

CONCLUSIONES

En tres capítulos se superó, explicadas las bases que sustentan el dominio teórico, el problema que supone la generación de proposiciones lingüísticamente cuantificadas en datos de *creep*. La implementación de una búsqueda eficiente del espacio de soluciones es lograda a través del modelo de Algoritmo Genético propuesto. De acuerdo a dicho modelo se concluye,

1. Teniendo en cuenta el diseño de cromosoma utilizado,
   - los operadores tradicionales de cruzamiento y mutación no tienen una influencia directa sobre la evolución de las proposiciones lingüísticas,
   - la implementación de la búsqueda local resulta un operador necesario y eficiente para evolucionar las proposiciones hacia buenas soluciones.
2. El empleo del operador de limpieza garantiza la obtención de resúmenes lingüísticos sobre datos de *creep* significativamente mejores.
3. El modelo híbrido propuesto resulta escalable sobre los datos de *creep*, es decir, es capaz de aumentar la cantidad de proposiciones lingüísticas a obtener, manteniendo la calidad de las mismas y con un aumento razonable del espacio de búsqueda revisado.
4. La herramienta automatizada implementada responde adecuadamente a las necesidades de procesamiento que implica el modelo propuesto.

RECOMENDACIONES

- Incluir otros criterios de calidad para la evaluación de las proposiciones lingüísticas en la función de aptitud.

- Utilizar el modelo híbrido propuesto en la experimentación con otras bases de casos para comprobar la generalidad y eficacia del modelo propuesto.

BIBLIOGRAFÍA

BOSC, P. et al. 2002. Fuzzy summarization of data using fuzzy cardinalities, Proceedings of IPMU 2002, Annecy, France,1553–1559.

BRUN, F. et al. 1999, *Theoretical design of ferritic creep resistant steels using neural network, kinetic, and thermodynamic models*, Materials Science and Technology, vol. 15, May. 1999, pp. 547-554.

CASTILLO, P. A. et al. 2011. *Linguistic Summarization of Time Series Data using Genetic Algorithms*. In Proceeding of the 7th Conference of European Society for Fuzzy Logic and Technology (EUSFLAT 2011). Aix-les-Bains, France: Atlantis Press, 416-423.

DARWIN, C. 1859. *On the Origin of Species by Means of Natural Selection*, Murray, London.

DAVIS, L. 1985. *Applying adaptive algorithms to epistatic domains*, in Proceedings of the International Joint Conference on Artificial Intelligence, 162-164.

DÍAZ, C. A. D. et al. 2009. *Modelo de máquinas de vectores de soporte para regresión aplicado a la estimación de la tensión de ruptura por termofluencia en aceros ferríticos*, Rev. Fac. de Ing. Univ. Antioquia, 47, 53-58.

DÍAZ, C. A. D. et al. 2011. *Using Linguistic Data Summarization in the study of creep data for the design of new steels*, in: Proceedings of11th International Conference on Intelligent Systems Design and Applications - ISDA 2011, Cordoba, Spain, 160 – 165.

DUBOIS, D. and PRADE, H. 1992. *Gradual rules in approximate reasoning*, Information Sciences, 61, 103–122.

GEORGE, R. and SRIKANTH, R. 1996. *Data summarization using genetic algorithms and fuzzy logic*, in: F. Herrera, J.L. Verdegay (Eds.), Genetic Algorithms and Soft Computing, Physica-Verlag, Heidelberg, 599–611.

GOLDBERG, D. E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

GÓMEZ, J. G. 2007. *Conjuntos y sistemas difusos*. Lógica difusa y apliaciones. E.T.S.I. Informática [Online]. Available: http://www.lcc.uma.es/~ppgg/FSS/.

HOLLAND, J. 1975. *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

KACPRZYK, J and ZADROŻNY, S. 1998. *Data mining via linguistic summaries of data: an interactive approach*, in: T. Yamakawa, G. Matsumoto (Eds.), Methodologies for the conception, design and application of soft computing—Proceedings of IIZUKA 98, Iizuka, Japan, 668–671.

KACPRZYK, J and ZADROŻNY, S. 2001. *On linguistic approaches in flexible querying and mining of association rules*, in: H.L. Larsen, J. Kacprzyk, S. Zadrozny, T. Andreasen, H. Christiansen (Eds.), Flexible Query Answering Systems. Recent Advances, Springer-Verlag, Heidelberg and New York, 475–484.

KACPRZYK, J. 2000. *Intelligent data analysis via linguistic data summaries: a fuzzy logic approach*, in: R. Decker, W. Gaul (Eds.), Classification and Information Processing at the Turn of the Millennium, Springer-Verlag, Berlin, Heidelberg and New York, 153–161.

KACPRZYK, J. and STRYKOWSKI P. 1999. *Linguistic summaries of sales data at a computer retailer: a case study*, in: Proceedings of IFSA99, vol. 1, Taipei, Taiwan, 29–33.

KACPRZYK, J. and YAGER, R. R. 2001. *Linguistic summaries of data using fuzzy logic*, International Journal of General Systems, 30, 133–154.

KACPRZYK, J. et al. 2000. *A fuzzy logic based approach to linguistic summaries of databases*, International Journal of Applied Mathematics and Computer Science, 10, 813–834.

KACPRZYK, J. and ZADROŻNY, S. 2005. *Linguistic database summaries and their protoforms: towards natural language based knowledge discovery tools*, Information Sciences, 173, 281–304.

KACPRZYK, J. et al. 2000. A fuzzy logic based approach to linguistic summaries of databases. Int. J. of Applied Mathematics and Computer Science 10, 813–834.

LOMBRAÑA, J. V. 1996. *Pensamiento difuso, pero no confuso: De Aristóteles a Zadeh (y vuelta)*. Psicothema. Universidad de Oviedo.

MALINOWSKI, G. 1990. *Conjuntos borrosos y lógica de Zadeh*: Universidad de Santiago de Compostela.

MASUYAMA, F. and BHADESHIA, H.K.D.H. 2007. *Creep strength of high-Cr ferritic steels designed using neural networks and phase stability calculations*, Fifth Int. Conf. on Advances in Materials Technology for Fossil Power Plants, EPRI press, 4B-01.

MICHALEWICZ, Z. 1992. *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, Berlin Heidelberg.

RASCHIA, G. and MOUADDIB, N. 2002, SAINTETIQ: *a fuzzy set-based approach to database summarization*, Fuzzy Sets and Systems, 129, 137–162.

RASMUSSEN, D. and YAGER, R.R. 1996. *Using summary SQL as a tool for finding fuzzy and gradual functional dependencies*, Proceedings of IPMU 96, Granada, Spain, 275–280.

RASMUSSEN, D. and YAGER, R.R. 1997a. *A fuzzy SQL summary language for data discovery*, in: D. Dubois, H. PRADE, R.R. Yager (Eds.), Fuzzy Information Engineering: a Guided Tour of Applications, Wiley, New York, 253–264.

RASMUSSEN, D. and YAGER, R.R. 1999. *Finding fuzzy and gradual functional dependencies with Summary SQL*, Fuzzy Sets and Systems, 106, 131–142

RASMUSSEN, D. and YAGER R.R. 1997b. SummarySQL-A fuzzy tool for data mining, Intelligent Data Analysis-An International Journal 1 (Electronic Publication). Available from: <http//:www.east.elsevier.com/ida/browse/96-6/ida96-6.htm>.

REEVES, C. 1993. *Modern Heuristic Techniques for Combinatorial Problems*, Blackwell Scientific Publications.

RUIZ, V. S. 2007. *Lógica difusa aplicada a conjuntos imbalanceados: Aplicación a la detección del síndrome de down*. Universitat Autònoma de Barcelona.

TANCRET, F. et al. 2003. *Design of a creep resistant nickel base superalloy for power plant applications*. Part 1 – Mechanical properties modeling. Materials Science and Technology, 19, 283-290.

YAGER, R. R. 1982. *A new approach to the summarization of data*. Information Sciences, 28, 69–86.

YAGER, R. R. 1988. *On ordered weighted averaging operators in multicriteria decision making*, IEEE Transactions on Systems, Man and Cybern, 18, 183–190.

YAGER, R. R. 1989. *On linguistic summaries of data*. IJCAI Workshop on Knowledge Discovery in Databases. Detroit, 378–389.

YAGER, R. R. 1991. *On linguistic summaries of data*. In: Piatetsky-Shapiro, G., Frawley, B. (eds.) Knowledge Discovery in Databases, 347–363.

YAGER, R. R. 1995a. *Linguistic summaries as a tool for database discovery*. Proceedings of Workshop on Fuzzy Database Systems and Information Retrieval at FUZZ-IEEE/IFES. Yokohama.

YAGER, R. R. 1995b. *Fuzzy summaries in database mining*. In: Proceedings of the 11th Conference on Artificial Intelligence for Applications, Los Angeles, USA, 265–269.

YAGER, R. R. 1996. *Database discovery using fuzzy sets.* International Journal of Intelligent Systems, 11, 691–712

ZADEH, L. A. 1965. *Fuzzy Sets*. Information and Control, 8, 338-353.

ZADEH, L. A. 1983. *A computational approach to fuzzy quantifiers in natural languages*. Computers and Maths with Applications, 9, 149–184.

ZADEH, L. A. 1985.*Syllogistic reasoning in fuzzy logic and its application to usuality and reasoning with dispositions*, IEEE Transaction on Systems, Man and Cybernetics,15, 754–763.

ZADEH, L. A. 2002. *A prototype-centered approach to adding deduction capabilities to search engines—the concept of a protoform*, BISC Seminar, University of California, Berkeley.