Implementación Computacional para el Apoyo a la Toma de Decisiones, Utilizando Metodologías Difusas

GABRIEL JAIME CORREA HENAO, EDGARDO ANAYA MARTÍNEZ
y GLORIA ELENA PEÑA ZAPATA
UNIVERSIDAD NACIONAL DE COLOMBIA. Facultad de Minas.
Escuela de Sistemas. Grupo de Investigación en Sistemas e Informática. Posgrado en Ingeniería de Sistemas.
gjcorre0@unalmed.edu.co ; edanaya@unalmed.edu.co ; gepena@unalmed.edu.co

Recibido para revisión 11 de Mar de 2004, aceptado May de 2004, versión final recibida 29 de Jun de 2004

Resumen: La implementación de algoritmos computacionales de alto nivel constituye una tarea compleja, especialmente cuando se requiere efectuar operaciones matemáticas que demandan gran capacidad de procesamiento. Este artículo muestra el proceso de construcción de una herramienta computacional, cuyos algoritmos se programan en lenguaje MATLAB, y su interacción con el usuario se efectúa en una aplicación construida bajo Lenguaje C++ Builder. Lo anterior, en virtud de la implementación de metodologías que emplean conjuntos y operadores difusos, como herramienta de apoyo para que un decisor evalúe las diferentes alternativas que se le presenten en un problema de Toma de Decisiones.

Palabras Clave: Librerías MATLAB, Lógica Difusa, Análisis Multiobjetivo, Programación con Componentes Visuales.

Abstract: Implementation of computational algorithms turns out to be a complex task, especially when it is required to process mathematical operations with large computational requirements. This paper shows the construction steps in order to program a computational tool, whose algorithms are built up in MATLAB language, and interacts with a user, by means of C++ Builder interfaces. This, in order to represent the methodology's implementation where both fuzzy sets and fuzzy operators are distinguished as a support tool, so a Decision Maker evaluates different alternatives into a Decision Making Problem.

Keywords: Fuzzy Logic, OWA Operators, Discrete Multiobjetive Problems.

1 INTRODUCCIÓN

Los seres humanos piensan de manera difusa, y como tal la formulación de sus problemas puede ser abordada a través de la teoría difusa. En los medios académicos el término "Lógica Difusa" está siendo desmitificado del misterio injustificado que ha dificultado su ingreso en ciertas áreas del conocimiento. Un ejemplo de ello tiene que ver con la Investigación de Operaciones, la cual cuando se asocia con los conjuntos difusos, corresponde a un tema poco explorado, especialmente en el medio local.

La Teoría de Toma de Decisiones Difusa y la Teoría de Estimaciones Difusas corresponden a diferentes dominios de aplicación de la Lógica Difusa. Bellman y Zadeh fueron los primeros en emplear modelación difusa en el ámbito de la toma de decisiones, haciendo énfasis que en un ambiente difuso, las diferencias entre las restricciones y las metas u objetivos dejan de existir (Bellman y Zadeh, 1970). En consecuencia, la formulación del problema de decisión se resume en encontrar la mejor estrategia (o acción) en una situación dada, de manera que se encuentre la mejor satisfacción de la función de utilidad (Correa y Peña, 2003*a*).

Tradicionalmente, la Teoría de Decisiones sólo podía ser tratada por medio de modelos probabilísticos para lidiar con la falta de conocimiento acerca de las funciones de utilidad, las preferencias subjetivas y el diagnóstico de situaciones. En consecuencia, siempre se ha asumido que la incertidumbre ha sido consecuencia de fenómenos aleatorios. En ese sentido, la lógica difusa permite abordar la solución de un problema de decisión, teniendo en cuenta la incertidumbre (Dubois y Prade, 1979).

2 DESCRIPCIÓN DE METODOLOGÍAS DIFUSAS PARA LA TOMA DE DECISIONES EN AMBIENTES DISCRETOS

En la Toma de Decisiones, empleando lógica difusa, se acude a la definición planteada por Zadeh en su paper de 1965 (Zadeh, 1965), en la que se otorga a un conjunto A, dentro de un universo X, una función de pertenencia:

$$A = X \rightarrow [0,1]$$

El mapeo A, también se denomina *función de pertenencia del conjunto difuso A*. Para la toma de decisiones apoyada en lógica difusa, se asume el mapeo de preferencias en el intervalo [0,1]. Dicho mapeo puede ser considerado como el grado en el que la alternativa A satisface el criterio i, o también como la similaridad existente entre A y alguna alternativa ideal del criterio i. En el marco de la propuesta introducida por Zadeh y por Bellman, la realización de dicho mapeo es percibida como la función de pertenencia del conjunto difuso y de las alternativas que satisfacen los criterios i (Bellman y Zadeh, 1970). Así pues, una buena decisión deberá satisfacer en forma conjunta las metas y restricciones asociadas con el ambiente donde se toma la decisión.

El principio establecido en la publicación de Bellman y Zadeh puede expresarse en un formato matricial. Dicha matriz, está compuesta de entradas tal que el elemento x_{ij} indica el rango de desempeño de la *i-ésima* alternativa, A_i, con respecto al *j-ésimo* atributo, M_j.

$$\begin{array}{c} \\ A_1 \\ A_2 \\ \vdots \\ A_n \end{array} \begin{array}{c} \begin{array}{cccc} M_1 & M_2 & \cdots & M_n \end{array} \\ \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{bmatrix} \end{array}$$

Asimismo se emplea la exponenciación para otorgar ponderaciones que permitan llevar a cabo las diferenciaciones en las metas y restricciones del problema (Yager, 1978).

$$D(x) = \min \{M_1^{\alpha_1}(x), \cdots, M_m^{\alpha_m}(x), R_{m+1}^{\alpha_{m+1}}(x), \cdots, R_{m+n}^{\alpha_{m+n}}(x)\}$$

donde, i varía entre 1 y m, y j entre $m+1$ y $m+n$.

Correspondientemente, a cada objetivo se encuentra el parámetro asociado. En la medida que el objetivo o meta sea considerado más importante, entonces el parámetro α tendrá un mayor valor. La sumatoria de los α está restringido a $(m+n)$, o sea, el número total de objetivos y restricciones. De esta manera se pretende forzar los pesos de los exponentes para que sean iguales a 1, en el supuesto caso que todas las metas y restricciones sean de igual importancia.

2.1 Agregación por Entropía – MEOWA

O' Hagan explica en una serie de publicaciones empleando el procedimiento de un bien planteado problema de optimización no lineal, que son llevados a entregar un valor máximo de entropía para obtener un nuevo conjunto de ponderaciones medias, llamadas promedios ponderados ordenados de máxima entropía MEOWA, según sus siglas en inglés (O'Hogan, 1990). Esta metodología ha probado ser muy útil en varios problemas de toma de decisiones del mundo real, teniendo en cuenta el optimismo del decisor, mediante la solución del siguiente problema de programación no lineal:

$$\max \; -\sum_{i=1}^{p} W_i \cdot \ln(W_i)$$

$$\text{sujeto a:} \begin{cases} \sum_{i=1}^{p} \frac{(p-i)}{(p-1)} \cdot W_i = \text{ (Factor de optimismo)} \\ \sum_{i=1}^{n} W_i = 1 \\ W_i \geq 0, \quad i \in N \end{cases}$$

Donde p es el número total de metas y restricciones del problema de decisión.

2.2 Agregación Sencilla (EZ-OWA)

Yager argumenta una nueva generación de agregadores que esencialmente involucran una partición de W en clases: aquellas que son iguales a cero y el resto que tienen todas el mismo valor. De manera que el parámetro esencialmente determina la localización en W donde se realiza la transición desde cero, hasta los pesos distintos de cero (Correa y Peña, 2003*b*).

Si denotamos $\alpha \in [0,1]$, como el factor de optimismo del decisor, entonces el cálculo del EZ-OWA, se realiza directamente mediante:

- Si $\alpha > 0.5$: Sea $i^* = \text{entero}((2\alpha - 1)\cdot p)$.

$$w_i = \begin{cases} \frac{1}{p}\left(\frac{1}{2(1-\alpha)}\right) & \text{para } 1 \leq i \leq i^* \\ 1 - i^* \cdot \frac{1}{p} \cdot \left(\frac{1}{2(1-\alpha)}\right) & \text{para } i = i^* + 1 \\ 0 & \text{para } i^* + 2 \leq i \leq p \end{cases}$$

- Si $\alpha = 0.5$: $w_i = \frac{1}{p}$
- Si $\alpha < 0.5$: sea $k^* = \text{entero}((1 - 2\alpha)\cdot p)$

$$w_i = \begin{cases} 0 & \text{para } 1 \leq i \leq k^* \\ 1 - \frac{p-k^*-1}{2\cdot\alpha\cdot p} & \text{para } i = k^* + 1 \\ \frac{1}{2\cdot\alpha\cdot p} & \text{para } k^* + 2 \leq i \leq p \end{cases}$$

Debido a la simplicidad y facilidad de este método para la generación de pesos, y por su analogía con la aproximación MEOWA, Yager denominó esta metodología, como EZ-OWA que corresponden a OWAS Sencillos (Yager, 2002).

3 APROXIMACIÓN A LA OPTIMIZACIÓN DE MODELOS DIFUSOS DE PROGRAMACIÓN LINEAL CON MÚLTIPLES OBJETIVOS

El planteamiento de un problema de decisión, requiere la formulación del mismo, a partir del siguiente planteamiento, donde hay incertidumbre en las restricciones:

$$\max \sum_{j=1}^{n} g_{ij} \cdot x_j, i = 1, 2, \cdots, p$$
$$\text{sujeto a: } \begin{cases} \sum_{j=1}^{n} A_{ij} \cdot x_j \lesssim b_i, i = 1, 2, \cdots, m, \\ x_j \gtrsim 0 \end{cases}$$

La expresión $\lesssim$ es una suave o una cuasi-inecuación. Es decir, se permiten violaciones matemáticas, a las restricciones. Los coeficientes de la matriz de pagos y de las restricciones (g_{ij}, A_{ij}, y b_i) se consideran valores concretos (no difusos).

Asimismo, es posible formular la solución de un modelo multiobjetivo, empleando números difusos en la formulación de las restricciones (A_{ij}, b_i).

3.1 Optimización Difusa empleando el Operador *Min*

Se tiene en cuenta que un problema multiobjetivo continuo puede tener múltiples soluciones (Frontera de Pareto). En ese sentido, una solución óptima hace parte de una solución completa. Por esa razón, es posible separar la función objetivo z_i, en su máximo valor z_i^+ y en su mínimo valor z_i^- al momento de solucionar el problema no difuso:

$$z_i^+ = \max z_i = \sum_j g_{ij} \cdot x_j, \quad z_i^- = \min z_i = \sum_j g_{ij} \cdot x_j$$

Se puede realizar una representación de dicha función, teniendo en cuenta que las soluciones z_i^+ y z_i^- son conocidas como las mejores y peores soluciones individuales sobre la Frontera de Pareto, respectivamente. Dado que para cada función objetivo z_i, se cambian los valores linealmente desde z_i^- hasta z_i^+, puede considerarse como un número difuso con función de pertenencia $\mu_i(z_i)$, como se muestra en la Figura 1.

En un problema multiobjetivo continuo es posible sugerir la inclusión de incertidumbre en la formulación de las restricciones, y emplear la solución más óptima de cada objetivo por separado, para definir luego la función de pertenencia de cada función objetivo (Petrovic-Lazarevic y Abraham, 2003).

El modelo equivalente para la solución del problema de programación multiobjetivo, se encuentra entonces

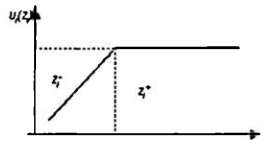

Figura 1: Función objetivo, como un número difuso

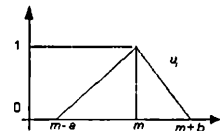

Figura 2: Concepto de número triangular difuso asimétrico $\bar{a} = (m - \alpha, m, m + \beta)_{L-L}$.

mediante:

$$\max \quad \lambda$$
$$\text{sujeto a: } \begin{cases} \lambda \cdot (z_i^+ - z_i^-) - z_i(x) \leq -z_i^- \\ \lambda \cdot p + A \cdot x \leq b + p \\ \lambda \leq 1 \\ x, \lambda \geq 0 \end{cases}$$

En contraste con los modelos usuales *max − min*, en cuyos métodos se usan soluciones de distancias, en la aproximación difusa las soluciones *eficientes*, contenidas en la "solución completa", se pueden distinguir por sus diferentes grados de satisfacción (Zimmermann, 1987).

3.2 Problemas Multiobjetivo con Coeficientes Difusos

Las propuestas metodológicas descritas hasta el momento, se han centrado en una formulación de problemas que aceptan la inclusión de incertidumbre, pero teniendo en cuenta la Programación Lineal Estándar.

La representación de un *número difuso triangular asimétrico*, el cual puede denotarse como $\bar{a} = (m - \alpha, m, m + \beta)_{L-L}$, se aprecia en la Figura 2. Entonces el problema multiobjetivo, con notación de números difusos es:

$$\max \sum_{j=1}^{n} \bar{g}_{ij} \cdot x_j, i = 1, 2, \cdots, p$$
$$\text{sujeto a: } \begin{cases} \sum_{j=1}^{n} \bar{A}_{ij} \cdot x_j \lesssim \bar{b}_i, i = 1, 2, \cdots, m \\ x_j \gtrsim 0 \end{cases}$$

En ésta formulación $\bar{g}_{ij}$, $\bar{A}_{ij}$ y $\bar{b}_i$ son números difusos, con la siguiente notación:

$$\bar{g} = (c - \varepsilon, c, c + \varphi), \quad \bar{A} = (m - \alpha, m, m + \beta)$$
$$\text{y } \bar{b} = (p - \gamma, p, p + \delta)$$

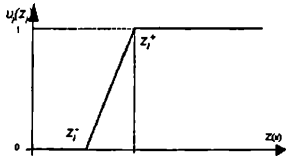

Figura 3: Mapeo de la Función de Pertenencia para cada objetivo $z_i(x)$.

Para concretar el problema se realiza la defuzzificación de las restricciones, empleando las siguientes transformaciones (Maleki, Tata y Mashinchi, 2000):

$$\begin{aligned}
\widetilde{g}_{ij} &= \widetilde{g}_{i1} \cdot x_1 + \cdots + \widetilde{g}_{in} \cdot x_n \\
&= \Big(\sum_j (c_{ij} - \epsilon_{ij}) \cdot x_j, \\
&\qquad \sum_j c_{ij} \cdot x_j, \sum_j (c_{ij} - \varphi_{ij}) \cdot x_j \Big) \\
\widetilde{A}_{kj} &= \widetilde{a}_{k1} \cdot x_1 + \cdots + \widetilde{a}_{kn} \cdot x_n \\
&= \Big(\sum_j (m_{kj} - \alpha_{kj}) \cdot x_j, \\
&\qquad \sum_j m_{kj}, \sum_j (m_{kj} - \beta_{kj}) \cdot x_j \Big) \\
\widetilde{b}_{kj} &= \Big(\sum_j (p_{kj} - \gamma_{kj}), \sum_j p_{kj}, \sum_j (p_{kj} - \delta_{kj}) \cdot x_j \Big)
\end{aligned}$$

Teniendo en cuenta dichas transformaciones, es posible identificar las funciones de pertenencia de los números triangulares, que permita defuzzificar el problema.

La Figura 3 presenta el procedimiento que permite efectuar dicha regla.

$$\mu_i(x) = \begin{cases} 0 & \text{para} \quad z_i(x) \leq z_i^- \\ \frac{[z_i(x) - z_i^-]}{(z_i^+ - z_i^-)} & \text{para} \quad z_i^- < z_i(x) \leq z_i^+ \\ 1 & \text{para} \quad z_i(x) > z_i^- \end{cases}$$

Para dicho propósito se requiere dividir el problema multiobjetivo en tres subproblemas específicos (Correa y Peña, 2003*b*):

- *Problema Pesimista*: Aquel que tiene en cuenta la solución del problema nítido, trabajando con los límites izquierdos de los coeficientes de costo de las funciones objetivo $(c_{ij} - \epsilon_{ij})$.
- *Problema Óptimo*: Aquel que tiene en cuenta la solución del problema nítido, trabajando con los coeficientes de costo, cuyo grado de pertenencia es igual 1 (c_{ij}).
- *Problema Optimista*: Aquel que tiene en cuenta la solución del problema nítido, trabajando con los límites derechos de los coeficientes de costo de las funciones objetivo $(c_{ij} - \varphi_{ij})$.

La siguiente aproximación metodológica permite encontrar las funciones de pertenencia de cada subproblema específico:

$$\mu_i^{\text{pesimista}}(x) = \begin{cases} 0 \\ \frac{[(c_i - \epsilon_i) \cdot x_j - z_i^-|_{\text{pesimista}}]}{(z_i^+|_{\text{pesimista}} - z_i^-|_{\text{pesimista}})} \\ 1 \end{cases}$$

$$\mu_i^{\text{óptimo}}(x) = \begin{cases} 0 \\ \frac{[(c_i) \cdot x_j - z_i^-|_{\text{óptimo}}]}{(z_i^+|_{\text{óptimo}} - z_i^-|_{\text{óptimo}})} \\ 1 \end{cases}$$

$$\mu_i^{\text{optimista}}(x) = \begin{cases} 0 \\ \frac{[(c_i + \varphi_i) \cdot x_j - z_i^-|_{\text{optimista}}]}{(z_i^+|_{\text{optimista}} - z_i^-|_{\text{optimista}})} \\ 1 \end{cases}$$

En la anterior formulación metodológica: z_i^+ y z_i^- corresponden a las soluciones óptimas sobre la Frontera de Pareto de cada subproblema específico (Correa y Peña, 2003*a*).

Finalmente, es necesario hallar el nivel mínimo que tienen que alcanzar todas las funciones de pertenencia. Lo cual se interpreta como el nivel de aspiración o de satisfacción de un decisor. Dicha solución eficiente se encuentra con la solución del problema auxiliar:

$$\max \quad \lambda$$

$$\text{sujeto a:} \begin{cases} \Big(z_i^+|_{\text{pesimista}} - z_i^-|_{\text{pesimista}}\Big) \cdot \lambda \\ \qquad -(c_{ij} - \epsilon_{ij}) \cdot x_j \leq -z_i^-|_{\text{pesimista}} \\ \Big(z_i^+|_{\text{óptimo}} - z_i^-|_{\text{óptimo}}\Big) \cdot \lambda \\ \qquad -(c_{ij}) \cdot x_j \leq -z_i^-|_{\text{óptimo}} \\ \Big(z_i^+|_{\text{optimista}} - z_i^-|_{\text{optimista}}\Big) \cdot \lambda \\ \qquad -(c_{ij} + \varphi_{ij}) \cdot x_j \leq -z_i^-|_{\text{optimista}} \\ (m_{kj} - \alpha_{kj}) \cdot x_j \leq p_k - \gamma_k \\ m_{kj} \cdot x_j \leq p_k \\ (m_{kj} + \beta_{kj}) \cdot x_j \leq p_k + \delta_k \\ \qquad x_i \geq 0 \end{cases}$$

donde, $i = 1, \cdots, \#$ Objetivos
$j = 1, \cdots, \#$ Variables
$k = 1, \cdots, \#$ Restricciones

4 IMPLEMENTACIÓN COMPUTACIONAL DE LA APROXIMACIÓN METODOLÓGICA

Las aproximaciones metodológicas recopiladas y debidamente adaptadas a la solución de problemas continuos,

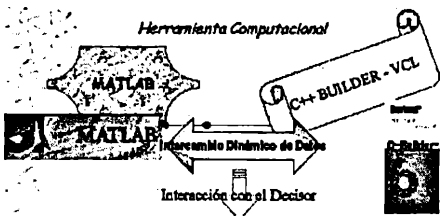

Figura 4: Implementación de la Herramienta Fuzzy Elección.

han sido implementadas en la herramienta computacional *Fuzzy Elección*, la cual esta elaborada con interfaces en lenguaje C_{++}, orientado a objetos, cuyos códigos de procesamiento están programados en lenguaje MATLAB, el cual se compila para asociarlo a la aplicación.

La idea es interactuar para que se efectúen operaciones semejantes a las que se realizan en el mundo real (no computarizado), donde los cálculos se basan en una serie íntegra de decisiones pequeñas en aspectos importantes y satisfactorios.

5 DESCRIPCIÓN DE LA HERRAMIENTA COMPUTACIONAL

La herramienta computacional *Fuzzy Elección* que se explica a continuación, está diseñada en dos módulos principales:

- El primero tiene que ver con la implementación de la metodología para la toma de decisiones, a nivel discreto, empleando conjuntos difusos.

- El segundo se refiere a la implementación de la metodología formulada para la toma de decisiones, en problemas continuos, usando conjuntos difusos.

El lenguaje de alto nivel elegido para implementar la herramienta computacional, se basa en una Plataforma C++ Builder, versión 6.0, de Borland. Dicho lenguaje tiene altas prestaciones para el programador, toda vez que contiene interfaces VCL (*Visual Components Library*), y además permite la implementación de algoritmos con Programación Orientada a Objetos. Los Algoritmos de Desarrollo se programan en Matlab, que brinda la posibilidad de compilar archivos, directamente compatibles con C_{++}.

El esquema mostrado en la Figura 4. presenta de manera ordenada la forma cómo se implementa la herramienta computacional *Fuzzy Elección*, que contiene la metodología formulada en virtud del Apoyo a la Toma de Decisiones con conjuntos difusos.

En *Fuzzy Elección* el decisor podrá recurrir a la solución de sus problemas, teniendo en cuenta la existencia de escenarios.

El esquema presenta claramente la posibilidad de efectuar algoritmos de alta capacidad computacional, en Matlab, para realizar el intercambio de datos, de archivos compilados, con el lenguaje de Programación Borland C++ Builder.

La implementación de la metodología siempre hace énfasis en la facilidad para manejo por parte de cualquier usuario. Se enmarca en el paradigma de Zadeh para procesar con palabras, empleando la metodología difusa, teniendo en cuenta la incertidumbre del lenguaje humano. Se procura que se realice computación con palabras, teniendo en cuenta las vaguedades e incertidumbres del lenguaje humano, para realizar la mejor decisión (optimización difusa).

6 COMPILACIÓN DEL CÓDIGO MATLAB

Mediante el uso del Toolbox de Compilación (*Matlab Compiler*) es posible convertir código generado en Lenguaje Matlab, el cual es capaz de efectuar operaciones matemáticas de gran complejidad, con tiempos de procesamiento relativamente cortos.

El código compilado puede ser transformado en librerías compatibles con lenguaje C_{++}. Asimismo, se pueden generar Archivos $*.MEX$ para el intercambio dinámico de datos, los cuales, en aplicaciones de Windows, están representados en Archivos tipo $*.DLL$. Los mismos son compatibles con programas elaborados en Visual Basic, Java y Visual C++.

6.1 Ventajas de la Compilación del Código MATLAB

El Compilador de MATLAB permite traducir códigos de *archivos* $*.M$ en *archivos* $*.C$. Los archivos resultantes en Código C pueden emplearse para generar cualquier archivo ejecutable o una librería que permita el intercambio dinámico de datos.

El código producido por el Compilador de MATLAB es independiente de la plataforma en la que se adecúa la interfaz con el usuario.

El Compilador de MATLAB, también genera código en Formato C ó C_{++}. Asimismo, es compatible con la generación de código Fortran. La compilación de código también se puede efectuar a partir de archivos *Simulink* ($*.MDL$).

Entre las ventajas de la compilación del código MATLAB, se cuentan las siguientes características:

- Crear aplicaciones independientes de MATLAB, que puedan correr en cualquier plataforma.

- Crear Librerías de Intercambio dinámico de Datos ($*.DLL$), las cuales son compatibles con cualquier

lenguaje de programación.

- Esconder algoritmos propietarios de MATLAB.
- Mejorar el tiempo de procesamiento del código.

Cuando se crean aplicaciones en MATLAB, es posible tomar ventaja de las funciones matemáticas de dicho lenguaje, y los usuarios de la aplicación no requieren que el usuario tenga instalado MATLAB. Las aplicaciones independientes son una manera muy conveniente de empaquetar la poderosa capacidad de MATLAB y distribuirla entre los usuarios de la herramienta computacional (Mathworks, 2002).

Los Algoritmos elaborados en MATLAB pueden ser integrados en aplicaciones C/C_{++}. Dicha compilación permite efectuar cálculos especializados desde la herramienta computacional *Fuzzy Elección*.

7 PROCEDIMIENTO PARA LA COMPILACIÓN DEL CÓDIGO

Las limitaciones del Toolbox de Compilación se condicionan a:

- Sólo se pueden convertir archivos $*.M$ de funciones.
- No soportan Archivos $*.M$ que usen objetos.
- No es posible compilar funciones que empleen los comandos *input* o *eval* para manipular variables del Workspace (Campo de Trabajo)

Asimismo, si la aplicación emplea Gráficos de MATLAB, es necesario tener instalada la Librería de Gráficos compatible con C/C_{++} (C/C_{++} *Graphics Library*).

7.1 Configuración de la Toolbox de Compilación

Antes de compilar código MATLAB, es necesario configurar el toolbox de compilación, mediante la siguiente digitación de comandos:

- *mex -setup*: Permite ajustar el tipo de plataforma en la que se efectúa el entorno de programación de las interfaces para la herramienta computacional. Su configuración soporta los lenguajes Visual C++, Borland C++ Buider, Fortran, LCC de MATLAB.
- *mbuild -setup*: Permite incluir la librería de gráficos C/C_{++} en la compilación.

7.2 Generación de Ejecutables en MATLAB

El procedimiento más simple para generar un archivo ejecutable directamente desde MATLAB, consiste en indicar la siguiente línea de comandos:

- mcc -m archivo1.m archivo2.m ...: Se generan archivos de código C estándar, así como un archivo ejecutable. Es necesario indicar todos los archivos asociados a la aplicación Matlab
- mcc -p archivo1.m archivo2.m ...: Se generan archivos de código C_{++}, así como un archivo ejecutable. Es necesario indicar todos los archivos asociados a la aplicación Matlab

7.3 Generación de Librerías para ser incluidas en aplicaciones Borland C++ Builder

La programación de las metodologías difusas para la toma de decisiones en ambientes discretos y continuos, emplean algoritmos complejos, cuya programación es preferible realizarla en lenguaje MATLAB (Mathworks, 2002).

Con la finalidad de incluir dichos algoritmos en una herramienta programada en lenguaje C++ Builder, es necesario generar librerías para que realicen el intercambio dinámico de datos.

La generación de dichas librerías se efectúa mediante el Toolbox de compilación, a partir del siguiente comando:

- mcc -t -B sgl -L C -W lib:mi_libreria -T link:lib -h archivo1.m archivo2.m ... archivo_n libmmfile.mlib

Este procedimiento genera las librerías de intercambio dinámico de datos. Esto es, se crea el archivo mi_libreria.dll.

Asimismo, el compilador genera los siguientes archivos:

mi_libreria.lib
mi_libreria.h
mi_libreria.c
mi_libreria.dll

Para incluir dichos archivos en el entorno de programación C++ Builder, es necesario incluir dichos archivos al proyecto y efectuar un llamado a la función mi_libreriaInitialize()

Asimismo es necesario efectuar un llamado a la función mi_libreriaTerminate() justo antes de salir de la aplicación C++ Builder

Durante la ejecución del código, es necesario invocar la función mlfFunction(...), donde Function representa el nombre de la función contenida en el archivo ($*.M$).

Es posible distinguir todos los nombres de las funciones (mlf, Initialize, Terminate) en el archivo de cabecera mi_libreria.h

8 RESULTADOS DE LA HERRAMIENTA FUZZY ELECCIÓN

Fuzzy Elección v1.0 permite la solución de problemas, teniendo en cuenta el soporte al decisor en escenarios.

De esta manera, el decisor se puede centrar y concebir adecuadamente la naturaleza de la decisión que vaya a tomar. Posteriormente se emplean métodos teóricos de optimización y de eficiencia, para combinar las piezas del escenario dentro de un todo interrelacionado. Finalmente, se indica cuál alternativa es la mejor, teniendo en cuenta los objetivos y las restricciones declaradas en el escenario de decisión.

La solución de problemas en ambientes discretos, considera la suposición en un parlamento conocido presente "Aquí y Ahora" (restricciones), una esperanza futura "Allá y Entonces" (objetivos) y varias rutas (alternativas) para ir de un presente "Aquí y Ahora" a un futuro "Allá y Entonces". El problema es entonces identificar cuál ruta elegir (alternativa) que de manera óptima soporte las presentes restricciones y los futuros objetivos.

La herramienta *Fuzzy Elección v1.0* también se ha concebido para solucionar problemas multiobjetivo. En este caso, se pretende efectuar la solución de problemas, abordando la solución con metodologías difusas.

El decisor puede recurrir a la solución de sus problemas, teniendo en cuenta la existencia de escenarios.

La solución a este tipo de problemas, contempla la concepción de incertidumbre en las restricciones del problema multiobjetivo. Para el efecto, se proponen dos maneras para considerar la incertidumbre. Téngase en cuenta que la solución de este tipo de problemas no se aborda desde el punto de vista estocástico, sino desde una concepción difusa:

- Una forma de consideración de incertidumbre, es que cada restricción en el problema puede admitir ciertos porcentajes de violación, según sea el criterio del decisor.

- Otra forma de consideración de incertidumbre, es mediante la concepción de números difusos, los cuales proporcionan una manera vaga para expresar un número particular.

Posteriormente se emplean métodos teóricos de optimización y de eficiencia, para combinar las piezas del escenario dentro de un todo interrelacionado. Dichas metodologías difusas, son el producto de investigaciones y de publicaciones de los últimos años, en medios especializados. De esta manera, se le informará al decisor la solución de las variables de decisión, así como un indicador que mide el nivel de satisfacción que podrá tener el decisor.

8.1 Visualización de Resultados para Problemas Discretos

La implementación de las aproximaciones metodológicas presentadas en la primera parte de este artículo, se implementan en lenguaje MATLAB, y se interactúa con el usuario en interfaces C++ Builder.

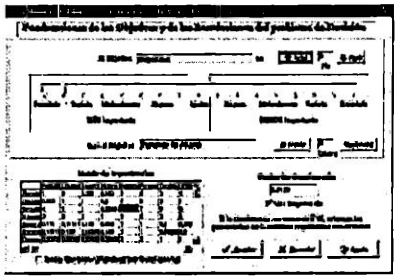

Figura 5: Generación de Resultados a partir de razonamientos entregados por el decisor.

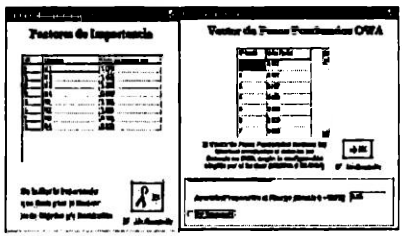

Figura 6: Visualización de Resultados entregados por el Código MATLAB, en interfaces C++ Builder.

El planteamiento de alternativas, objetivos y restricciones en el modelo de decisión, así como la valoración de juicios del decisor, se realiza de manera interactiva. La Figura 5. presenta una interfaz de intercambio dinámico de datos, donde se calcula la matriz de importancias, y el índice de consistencia de un decisor, al momento de valorar los objetivos de un problema de Toma de Decisiones.

Asimismo es posible visualizar los resultados entregados mediante matrices, a partir de los cálculos efectuados por el código MATLAB. Un ejemplo de ello se muestra en la Figura 6.

8.2 Resultados en Problemas de Tipo Continuo

Todo problema de tipo continuo puede ser representado a partir de la formulación del siguiente problema de programación matemática:

$$\max/\min \quad Z_i(x) = g_{ij} \cdot x$$
$$\text{sujeto a:} \quad A \cdot x \leq b + \%\ \text{Violación}$$

Donde los componente de las matrices de costo y de restricciones pueden declararse como números concretos, o como números difusos.

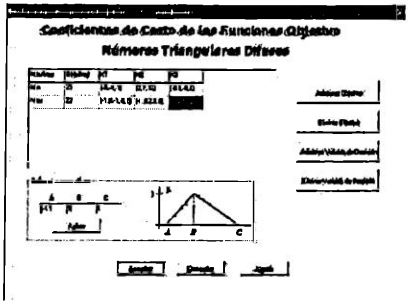

Figura 7: Entrada de variables desde interfaz C++ Builder para procesamiento en código compilado de MATLAB.

Figura 8: Visualización de Resultados entregados por MATLAB para Problemas Continuos.

La Figura 7, presenta una interfaz empleada para la declaración de funciones objetivo, con números difusos. Dichas entradas son asumidas por el código compilado de MATLAB, para que sean procesadas mediante la aproximación metodológica presentada en la primera parte de éste artículo.

Los resultados entregados por el código compilado de MATLAB pueden visualizarse directamente en interfaces de la herramienta computacional, como se ilustra en la Figura 8.

El usuario está en capacidad de verificar los valores de las variables de decisión, el cumplimiento y logro de objetivos, así como el desempeño y la holgura en la satisfacción de las restricciones del problema de decisión.

9 CONCLUSIONES

El uso de lógica difusa en la Toma de Decisiones ayuda a elegir alternativas, rompiendo los escenarios en pequeñas partes en las cuales se pueda enfocar fácilmente un decisor. Luego se emplean metodologías de optimización en los cuales se combinan las partes de los escenarios en un todo interrelacionado. Finalmente se establece una medida para indicar cuál alternativa es la mejor entre las restricciones y los objetivos del escenario particular.

Se verifica la capacidad de la lógica difusa para la solución de problemas que usualmente involucran mucha vaguedad y ambigüedad (y por lo tanto, una forma borrosa) en los objetivos y en las restricciones. En ese sentido, el objetivo de una metodología para el apoyo de decisiones difusas, tiene que ver con la obtención de una decisión, que sea óptima en el sentido que se procura el cumplimiento de algunas metas, mientras se cumplan un conjunto simultáneo de restriccione, es decir, que no se violen.

En lo que se refiere a la programación matemática, se ha ilustrado una formulación metodológica que se aparta de los conceptos en los cuales se fundamentan las herramientas duras de la investigación de operaciones, en el sentido que se procuran encontrar las satisfacciones más eficientes para un decisor, dentro de una solución completa para un problema de decisión continuo.

Debido a la complejidad de las operaciones matemáticas ejecutadas en *Fuzzy Elección v 1.0*, ha sido preferible desarrollar los programas numéricos en MATLAB, la cual constituye una herramienta de alto nivel y que incluso requiere menos esfuerzo que con lenguajes de programación convencionales, como C/C_{++}, Java o Visual Basic, y a partir de la compilación del programa numérico, se efectúa el intercambio dinámico de datos con C_{++}. Lo anterior se ve traducido en un aumento significativo de la productividad en la programación de la herramienta computacional.

REFERENCIAS

Bellman, R. y Zadeh, L. (1970), 'Decision-making in a fuzzy environment', *Management Science* **17**, 141–166.

Correa, G. y Peña, G. (2003*a*), 'Propuesta metodológica para apoyo a la toma de decisiones discretas, mediante el uso de operadores difusos', *Encuentro EITI, 2003* pp. 1–6.

Correa, G. y Peña, G. (2003*b*), 'Propuesta metodológica para la solución de problemas multiobjetivo continuos con conjuntos y operadores difusos', *Encuentro EITI, 2003* pp. 13–19.

Dubois, D. y Prade, H. (1979), 'Decision making under fuzziness', *Advances in fuzzy set theory and application* pp. 279–303.

Maleki, H., Tata, M. y Mashinchi, M. (2000), 'Linear programming whit fuzzy variables', *Fuzzy Sets and Systems* (109), 21–33.

Mathworks, I. (2002), 'Matlab compiler toolbox user's guide', *www.mathworks.com* .

O'Hogan, M. (1990), 'A fuzzy neuron based on maximum entropy ordered weighted averaging', *Proc. 24th Ann. Asilomar Conf. on Signals, Systems, and Computers, IEEE and Maple Press.* .

Petrovic-Lazarevic, S. y Abraham, A. (2003), 'Hybrid fuzzy-linear programming approach for multi criteria decision making problems', *International Journal of Neural, Parallel and Scientific Computations* **11**(1,2).

Yager, R. (1978), 'Fuzzy decision making including unequal objectives', *Fuzzy Sets and Systems* (1), 87–95.

Yager, R. (2002), 'Ez-owa weights', *Machine Intelligence Institute Press. Iona College* .

Zadeh, L. (1965), 'Fuzzy sets', *Information and Control* **8**, 338–353.

Zimmermann, H. (1987), *Fuzzy Sets, Decision Making, and Expert Systems (International series in management science / operations research)*, Kluwer Academic Publishing., Dortrecht, Alemania.