MODELACION DE SISTEMAS NO LINEALES POR ALGORITMOS GENETICOS Y ORIENTACION A OBJETOS

José L. Torres O. **Jesús A. Hernández R.**
Universidad Nacional de Colombia - Facultad de Minas (Cra. 80 65-223)
(jltorres | jahernan) @perseus.unalmed.edu.co
(57) (4) 422 00 22

RESUMEN: Se presenta un modelo de aplicación general que toma elementos de los paradigmas evolutivo y objetual. Se fusionan la programación orientada a objetos y los algoritmos genéticos para dar solución a sistemas de ecuaciones algebraicas no lineales. En este modelo se expone un nuevo método de representación de los individuos especificando las ecuaciones y sus componentes como clases objetuales, y con una población de estas clases se desarrolla en programación por objetos un algoritmo genético para encontrar una solución adecuada. También se presenta un nuevo operador genético llamado aquí mutación especial, que garantiza diversidad y características específicas en los herederos. Se exploran y proponen nuevos caminos para extender el modelo a la solución de otros problemas.

Palabras Clave: Algoritmos Genéticos, Modelamiento O.O, Informática en Ingeniería, Inteligencia Artificial.

1- INTRODUCCION

En este artículo se toman dos modelos exitosos e independientes cada uno del otro, y con su combinación se genera uno nuevo, de aplicación general. Por un lado, la modelación conceptual orientada a objetos se ha distinguido por su acercamiento a los procesos cognitivos naturales. Por otra parte, los algoritmos genéticos (A.G.) se han reconocido precisamente por ofrecer buenas soluciones a problemas nuevos y tradicionales, muchos de ellos de carácter no lineal

La posibilidad de combinar la Programación Orientada a Objetos POO y la Inteligencia Artificial, fue explorada inicialmente por Tello [Tello, 89], quien aplicó inicialmente esta conjunción a sistemas expertos y sistemas de aprendizaje; sin embargo, sigue siendo un campo que aunque adolece de pocas publicaciones, se vislumbra como una base para desarrollar la nueva generación de lenguajes científicos y técnicas de modelamiento.

Como en los algoritmos genéticos la representación de los individuos es uno de los aspectos más importantes a considerar (Angeline 96, Mitchell 97), en este artículo presentamos, además, un método para conformar las cadenas de representación individual con base en el modelo orientado a objetos. Nuestro modelo genético - objetual (Modelo OOG) es muy fácil de implementar, por la confluencia natural entre una representación objetual de los individuos y la construcción del algoritmo genético usando programación por objetos. El modelo OOG puede utilizarse en cualquier clase de problema, y demostramos su capacidad solucionando sistemas complejos de ecuaciones no lineales.

2- PROBLEMA A SOLUCIONAR

El sistema no lineal de ecuaciones algebraicas, $Ax = B$, con A una matriz de coeficientes, x una matriz no lineal de incógnitas (Ej. y^2z^3, $wy^{0.5}z$, $4Cos(2x^2-x)$, $5Log(x)$), y B un vector unidimensional de términos independientes. Los términos constantes son asumidos por el programa como coeficientes multiplicados por las incógnitas con exponente cero. El sistema a resolver puede incluir ecuaciones con funciones polinómicas, potenciales, logarítmicas, exponenciales o trigonométricas. El sistema lineal es un caso especial con A una matriz de coeficientes, x un vector de incógnitas de primer grado, y B un vector unidimensional de valores constantes. En nuestro modelo el sistema no está atado a la condición de ser linealmente independiente, porque el modelo OOG cumple con hallar una o varias soluciones aproximadas para todo el sistema. El modelo OOG halla al menos una solución para los sistemas con solución conocida; para los que no la tienen, da la solución más aproximada, con su parámetro de ajuste, que de acuerdo al caso y a un análisis, será la solución esperada, o se deducirá que el sistema no tiene una solución. Una de las bondades del modelo OOG, como en el caso lineal, es la posibilidad de encontrar, la mayoría de las veces, varias soluciones para problemas con múltiples soluciones, así fue el caso para un polinomio de grado cinco para el cual encontró todas las soluciones posibles.

3- SOLUCIÓN MEDIANTE LA COMBINACIÓN DEL DISEÑO ORIENTADO A OBJETOS Y LOS ALGORITMOS GENÉTICOS.

3.1 MODELO GENERAL

El planteamiento general del modelo OOG se basa en la existencia de un problema con comportamiento no lineal, que ya esté expresado en un sistema de m ecuaciones matemáticas con n incógnitas. Cada una de las ecuaciones de este sistema se considera un conjunto con una cardinalidad igual al total de miembros y términos independientes de la ecuación. Entre los elementos de este conjunto de miembros puede encontrarse varias veces una misma incógnita y pueden aparecer diferentes constantes independientes, y el método de solución que proponemos funciona adecuadamente. Si se usa un sistema de ecuaciones completamente reducido, se obtiene el óptimo rendimiento computacional. El modelo OOG además reconoce incógnitas, signos, coeficientes, exponentes y funciones con sus argumentos, de los respectivos miembros.

A todas las incógnitas componentes de esta arquitectura de miembros se les asigna valor de forma aleatoria, para proceder a una evaluación integral de todas ellas. La combinación de la instanciación numérica de las incógnitas con los términos independientes, de acuerdo al sistema de ecuaciones a resolver, da una posible solución que debe compararse con un parámetro de ajuste. El proceso se repite con otras instanciaciones numéricas hasta que se tenga una solución aceptable. La solución exacta exige un parámetro de ajuste igual a cero.

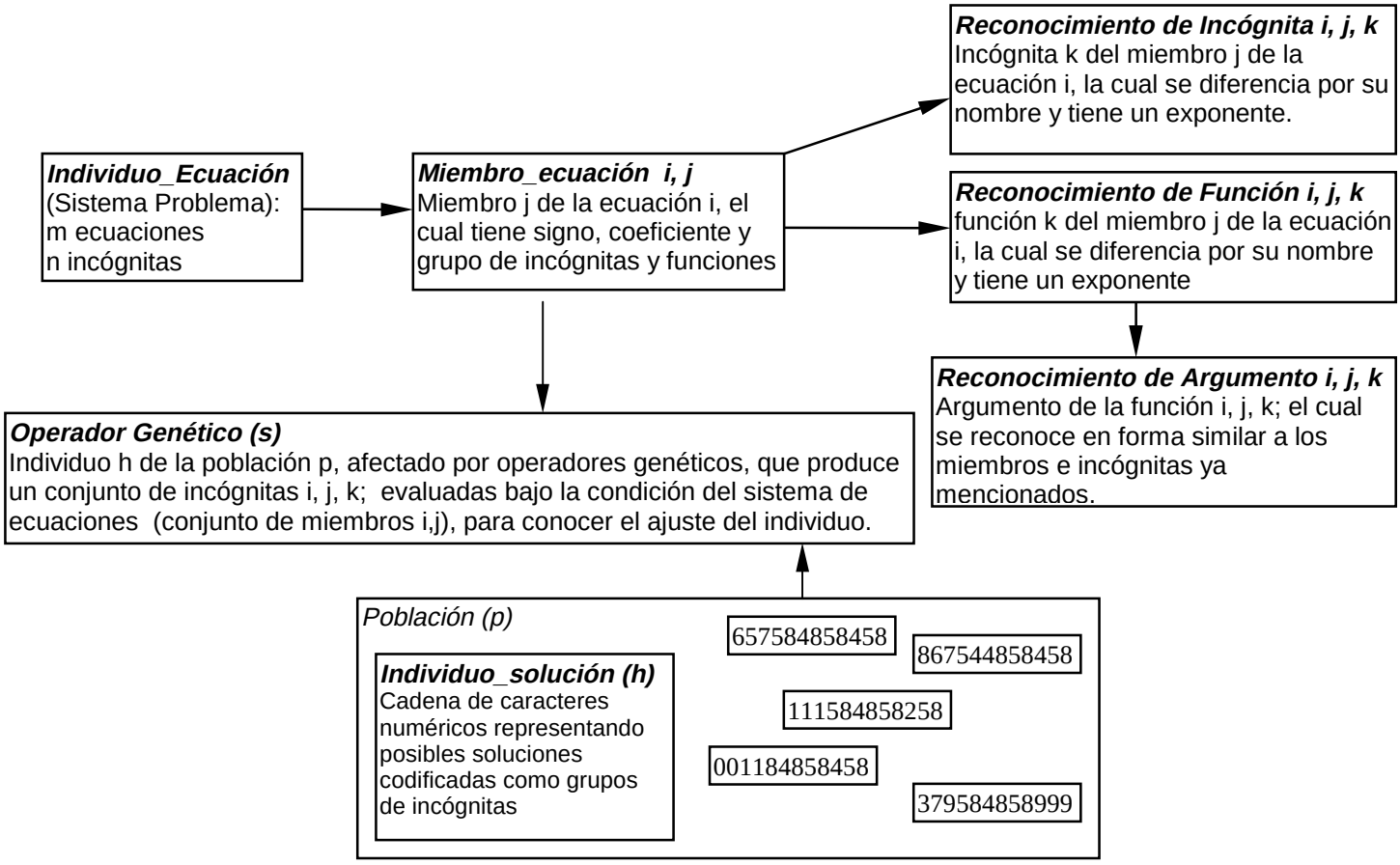


Figura 1. Flujo de información en el modelo genético - objetual

Los conjuntos de miembros de las ecuaciones se pueden modelar en clases objetuales (Brumbaugh 94, Pastor, 95), cuya instanciación determina si se ha alcanzado una solución. Como se puede observar en la Figura 1, cuatro clases objetuales principales serían suficientes: la clase individuo_ecuación, la clase miembro_ecuación, la clase individuo_solución y la clase operador_genético. La clase miembro_ecuación tiene dos especializaciones, la clase incógnita y la clase función, que posee a su vez una especialización: clase argumento. La clase operador_genético posee cuatro especializaciones: la clase reproducción, la clase cruzamiento, la clase mutación tradicional y la clase mutación especial (las cuales no se muestran en este diagrama), que actúan sobre cada individuo de la población creando la clase individuo_solución, que en últimas es la base para encontrar la solución buscada. La clase Población es una agregación resultante de la clase individuo_solución.

3.2 MODELO GENÉTICO

Como es conocido los algoritmos genéticos se basan en los mecanismos de selección que utiliza la naturaleza, según los cuales los individuos más aptos de una población son los que sobreviven al adaptarse más fácilmente a los cambios que se producen en su entorno. Hoy en día se sabe que estos

cambios se efectúan en los genes de un individuo, y que sus atributos más deseables (los que le permiten adaptarse mejor a su entorno) se transmiten a sus descendientes cuando éste se reproduce sexualmente [Masini 80].

A continuación se presenta el esquema general de un A.G. (Mitchell 97, Koza 92, Reeves 91):

1. Codificar las variables de decisión como un cromosoma.
2. Inicializar la población de cromosomas como la actual generación usando un método aleatorio.
3. Realizar hasta que se cumpla el criterio de parada:
   a. Evaluar los valores de la población de la generación actual.
   b. Seleccionar algunos cromosomas en la población con el valor mayor de aptitud como cromosomas padres para reproducir una nueva población de cromosomas hijos.
   c. Aplicar los operadores de cruce y mutación a los cromosomas padres seleccionados en el paso anterior. Los operadores aplicados a los padres son determinados por valores aleatorios, teniendo en cuenta además las ratas de mutación y cruce definidas con anterioridad.
   d. Reemplazar toda la población por los cromosomas hijos como la actual generación.

El conjunto de ecuaciones e incógnitas del sistema problema se modeló en las clases individuo_ecuación y miembro_ecuación**,** con una representación decimal, aprovechando las experiencias de Michalewicz y Golberg [Michalewicz, 1999], los cuales advierten sobre la diferencia entre la teoría y la práctica respecto al uso de lenguajes de codificación: “parece ser que el lenguaje binario, u otros de baja cardinalidad, por sus características serían los más óptimos para el mejoramiento del esquema representativo, pero esto ha sido contradicho por resultados empíricos de muchos trabajos, los cuales han usado alfabetos numéricos más extensos, como, un alfabeto representando números reales de punto flotante, que trabajan bien en una amplia variedad de problemas prácticos”. Así:

$5y - 0.6x^3 + 3xz - 2\text{Sin}(x^2) + 8\text{Log}(y)$ *Individuo_ecuación 1*

.
.

$4y^2z^3 - 7x^5 \ldots + 4\text{Cos}(2x^2\text{-}x) \ldots + 5\text{Log}(x)$ *Individuo_ecuación i*

Miembro 1 Miembro 2 Miembro j

.
.

$0.5x^2y^3 + 6y^{1.5} - 4\exp(x^2\text{-}x) + 4xyz$ *Individuo_ecuación m*

A su vez cada miembro se representó así:

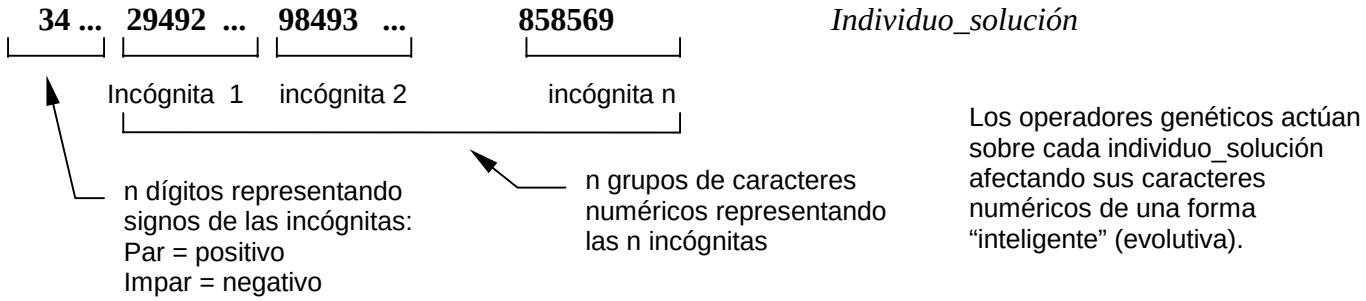
Signo Coeficiente

$- 4\ yz^3 \ldots x^5\ \text{Cos}^3(2x^2\text{-}x) \ldots \text{Log}(x)$ *Miembro j*

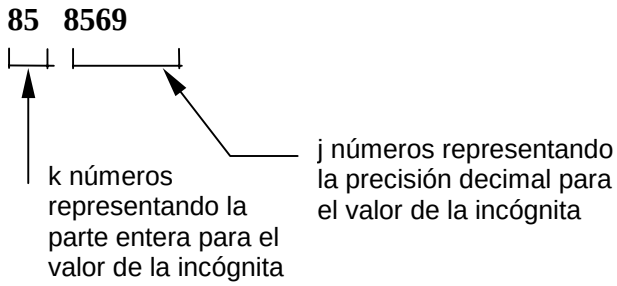
Conjunto de incógnitas con exponente

Conjunto de funciones con exponente y argumento.

La población, modelada por técnicas evolutivas, es una asociación de individuos_solución en una generación determinada. El individuo_solución que es la pieza fundamental para codificar el algoritmo genético ya que almacena los valores de las incógnitas, se representó de la siguiente forma para un sistema de m ecuaciones y n incógnitas:



A su vez cada grupo de caracteres numérico representando una incógnita esta compuesta por:



3.3 MODELO OBJETUAL

Se utilizó OO-METHOD y el lenguaje OASIS (Open and Active Specification Systems) como la metodología de especificación formal orientada a objetos para sistemas abiertos y activos. En OASIS, los objetos se definen como procesos observables y tienen un comportamiento cliente-servidor. Una clase en OASIS consiste de: un nombre de clase, una función de identificación para las instancias (objetos) de la clase, y un tipo o plantilla que todas las instancias comparten. Una descripción completa del lenguaje OASIS puede encontrarse en [Pastor 95]. Por la brevedad del espacio, la definición formal de la plantilla de clase no se presenta. La figura 2 muestra la plantilla sintáctica; las secciones de identificación y procesos son obligatorias, las otras opcionales.

El modelo conceptual en OASIS se hace con tres vistas: Modelo de Objetos (diagrama de configuración de clases), Modelo Dinámico (diagrama de transición de estados y diagrama de interacción entre clases), y el Modelo Funcional (fórmulas en lógica dinámica). Debido a las limitaciones, sólo presentamos en la figura 3, el diagrama de interacción entre clases, complementado con los atributos constantes, atributos variables y servicios de clase.

Class(Class name) Identification Constant_attributes Variable_attributes Derived_attributes Private_events Shared_events Constraints Valuation Derivation Preconditions Triggers Transactions Processes **End_class**

Figura 2. Plantilla de Clase en OASIS

3.4 FUNCION DE ADAPTACION O AJUSTE

Se calculó con la siguiente rutina:

```
For j = 1 To m                          Recorre todas las ecuaciones
    h = 0, sx=0, sh=0
    For k = 1 to nmiembros              Recorre todos los miembros de la ecuación j
      For p = 1 To n                Recorre todas las incógnitas del miembro k
            sx=sx+X(p)^exp(p)           Acumula producto de incógnitas
      Next
      For q = 1 to nfunciones
            sx=sx+sx*F(q)^exp(q)        Acumula producto de funciones
      Next
      sh = sh + c(j, p) * X(p)          Acumula el producto del vector coeficientes por el vector
                                        conjunto de incógnitas y funciones en cada miembro de la
                                        ecuación j.
    Next
    ajuste(i) = ajuste(i) + Abs(b(j) - sh)   Calcula el ajuste de la ecuación i como la diferencia de la
                                        sumatoria de todos los miembros evaluados para una
                                                instancia de incógnitas y el término independiente.
     ajuste(individuo_solución x) =  ajuste(individuo_solución x) + ajuste (i)
Next
```

Al final se da una tabla ordenada por ajustes y mostrando sus respectivos individuos, la cual es la base para continuar con próximas generaciones. Como se puede inferir mientras más cercano a cero (0.01 fue el ajuste para terminar la mayoría de los problemas en este trabajo), mejor ajuste de la solución hallada.

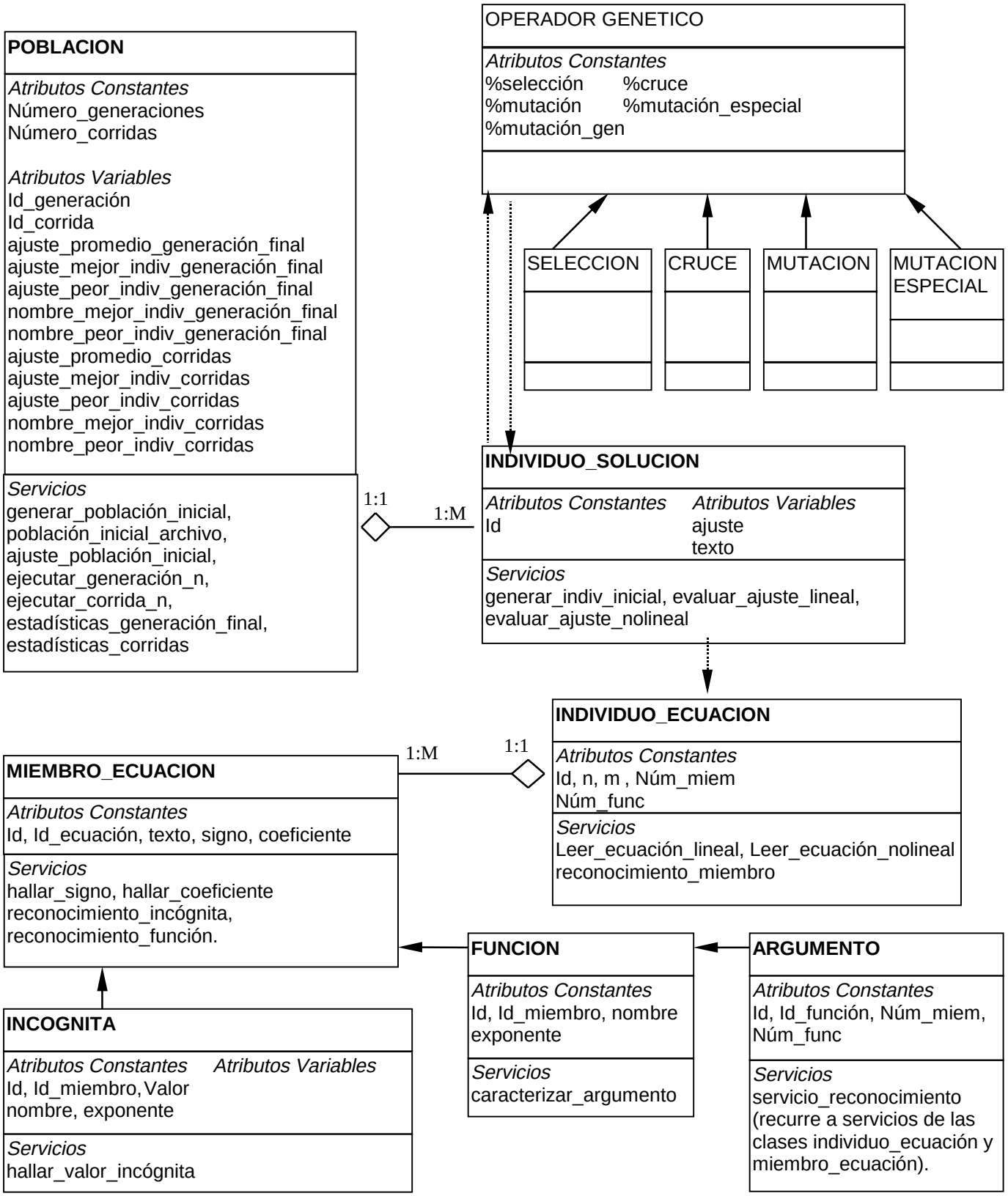


Figura 3. Modelo Objetual Mostrando la Interacción de Clases.
(→ Asociación entre clases, ⋯→ Interacción entre clases)

3.5 MUTACIÓN ESPECIAL

El operador Mutación Especial se utiliza cuando se presenta una excesiva repetición de los mejores individuos: Intenta dar mayor variedad dentro de los mejores cromosomas compensando aquella no lograda por la función normal de mutación. Se muta el gen con una probabilidad de 1/longitud_individuo (Al menos 1 gen se muta en cada individuo repetido, aunque esta probabilidad puede variar). Se realiza después de que el individuo se ha sometido a todos los otros operadores genéticos. Se toma un porcentaje de la generación, ordenada por ajuste, y se compara cada cromosoma con los otros mutando los que estén repetidos.

RESULTADOS

Se probó el modelo OOG para múltiples casos en los cuales los más relevantes fueron:

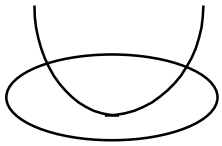
C1) El polinomio $X^5-X^4-13X^31+3X^2+36X-36=0$, el cual se puede demostrar que tiene 5 raíces reales. Las cinco raíces fueron halladas. Muchos métodos convergen a sólo una raíz.

Raíces reales exactas	Raíces reales halladas por el modelo OOG
1, 2, -2, 3, -3	1, 2, -2, 3, -3

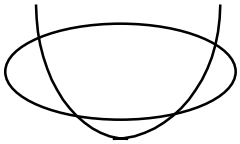
C2) Elipses y parábolas interceptándose en uno, dos, tres, cuatro puntos, o sin intercepción. En todos los casos hallaba las soluciones (los puntos de intercepción si los había).

Dos puntos: $2X^2 - 4X - Y - 3 = 0$
$X^2 - 2X + Y^2 - 4Y - 31 = 0$

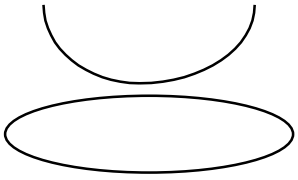
Solución: X= -0.83, Y= 7.71 X= 2.83, Y= 7.71



Cuatro puntos: $2X^2 - 4X - Y - 3 = 0$
$X^2 - 2X + Y^2 - 30Y + 190 = 0$
Solución: X= -1.94, Y= 20.23 X= -0.91, Y= 9.31
X= 1.55, Y= 9.03 X= 2.65, Y= 20.77



Sin intercepción $2X^2 - 4X - Y - 3 = 0$
$Y^2 + 4X^2 - 4Y + 8X + 4 = 0$
Solución: El modelo OOG no converge



C3) El sistema de ecuaciones: $X^2 + X - Y - 2 = 0$

$$X^2 - 2X + Y^2 + 4Y + 1 = 0$$
$$X^5 - X^4 - 13X^3 + 13X^2 + 36X - Y - 36 = 0$$

El cual tiene una solución real demostrada.

Soluciones reales exactas	Soluciones reales halladas por el modelo OOG
x=1, y=0	x=1.004, y=0.001

C4) El sistema de ecuaciones: $X^2 + Y^2 + Z^2 - 1 = 0$
$-X + Y^2 + Z^2 + 0.5 = 0$

Que representan dos superficies tridimensionales las cuales se interceptan, dando como solución a su vez otra curva en dos dimensiones, fue solucionada para múltiples puntos de esta curva solución. Este comportamiento es debido a que el modelo OOG da soluciones puntuales.

Soluciones reales exactas	Soluciones reales halladas por el modelo OOG
Curva: $(Y^2+Z^2)^2+2(Y^2+Z^2)-0.75=0$	X=0.833, Y=0.389,Z=0.411
	X=0.83, Y=0.097, Z=0.571
	Múltiples soluciones de este estilo

C5) El sistema de ecuaciones fuertemente no lineal: $4\text{Cos}(2X) - X = Y$
$5\text{Log}_{10}(X) - X^3 + X^2 = Y$

El cual tiene dos soluciones reales de acuerdo a gráficas realizadas:

Soluciones reales exactas (por métodos tradicionales)	Soluciones reales por el modelo OOG
No se pudo determinar	x=2.156, y=-3.711
	X = 0.751, Y = -0.479

En cada solución se hacían 5 corridas, para 40 generaciones y 1000 individuos, valores óptimos para el modelo OOG. El programa construido daba las estadísticas para cada corrida y para un grupo de corridas. Cuando el sistema de ecuaciones tenía varias soluciones, se necesitaba, algunas veces, observar los resultados de cada corrida, porque era posible que encontrara más soluciones en unas que en otras.

Cuando el ajuste era malo, y al aumentar el número de generaciones e/o individuos, no mejoraban las soluciones obtenidas, se llegaba a la conclusión que no existía una solución.

También es conveniente señalar que de acuerdo a Hageman y Young [Hageman 81], un método iterativo converge cuando el error entre la solución real y la hallada en una iteracción k disminuye

cuando k aumenta, así, la función de aptitud asegura de una forma “natural” la convergencia si existen soluciones mejores, ya que los operadores genéticos harán mover las generaciones futuras hacia mejores individuos, osea aquellos con una menor diferencia entre lo buscado y lo hallado. En el caso en que existan soluciones no factibles, Michalewicz [Michalewicz, 99] nos aconseja recurrir a la heurística: “El proceso de selección de una función de evaluación podría llegar a ser demasiado complejo, especialmente cuando manejamos soluciones factibles y no factibles para el problema; varios métodos heurísticos usualmente son incorporados en este proceso”. Finalmente, pensando en la localidad o globalidad de las soluciones halladas, también por la naturaleza de su paralelismo implícito, los AG tienen la propiedad de moverse entre soluciones óptimas (óptimos locales) intentando llegar o acercarse lo máximo posible al óptimo global (como lo muestran los resultados mostrados anteriormente), en este caso, métodos o estrategias heurísticas de oscilación y dispersión de los espacios de búsqueda pueden llegar a ser demasiado importantes [Michalewicz, 99].

CONCLUSIONES

El modelo OOG se presenta como una alternativa viable para la solucion de problemas complejos. La sola aplicación del modelo OOG ayuda a la comprensión de la naturaleza y a la cuantificación de problemas no lineales, generalmente resueltos por linealización. Como en estos casos se recurre a aproximaciones y transformaciones de las ecuaciones originales para encontrar una solución, queremos abrir un futuro camino a explotar para la resolución directa de sistemas de ecuaciones que incluyan en sus miembros integrales, diferenciales ordinarias o diferenciales parciales.

Una de las principales ventajas del modelo OOG es que si el sistema problema es soluble, siempre se encuentra al menos una solución y la consecusión de respuestas es más completa y más confiable. Con los métodos tradicionales (ej. hill climbing, Newton-Raphson, entre otros) esta es una tarea dispendiosa y en ocasiones de poco éxito. Además, con el modelo aquí presentado no hay acumulación de errores inherentes por redondeo, por la independencia de procesamiento matemático entre generaciones y por el menor número de operaciones ejecutadas. Otra ventaja, es que el modelo OOG admite cualquier clase de no linealidad, sin atarse a entidades matemáticas particulares.

El modelo OOG es muy económico en la modelación de los problemas y en tiempo de procesamiento. El operador Mutación Especial garantiza diversidad en la población, eliminación de ciclos infinitos, mejora la velocidad de procesamiento y ayuda a una rápida covergencia a la solución.

Los próximos trabajos a presentar, algunos de los cuales se realizan en este momento, se centran en la comparación con paquetes de software de igual propósito y con casos más complejos, y en explorar el modelo OOG en campos específicos de la ingeniería como ecuaciones de flujo de fluidos compresibles en yacimientos de petróleo, estudio de campos electromagnéticos, sistemas de transferencia de calor y de masa, sistemas de potencia, sistemas de control, sistemas de flujo multifásico, sistemas biológicos complejos adaptativos, programación no lineal, etc.

Agradecimientos: Los autores desean agradecer al profesor Carlos Mejía, PhD, del Departamento

de Matemáticas de la Universidad Nacional de Colombia - Sede Medellín, por sus comentarios sobre este trabajo.

Referencias

[Angeline 96] Angeline, Peter J.; Kinnear, Kenneth E. Advances in Genetic Programming. MIT Press. Cambridge. Vol. 2, 1996.

[Brumbaugh 94] Brumbaugh, David E. Object - Oriented Development. Building Case Tools with C++. John Wiley & Sons. New York. 1994

[Coello 94] Coello C., Carlos A. Introducción a los Algoritmos Genéticos. Universidad de Tulane, Nueva Orleans, EUA. 1994.

[Hageman 81] Hageman Louis A.; Young David M. Applied Iterative Methods. Academic Press, Inc. 1981.

[Hojjat 95] Hojjat Adeli, Shin-Lin Hung. Machine Learning. Neural Networks, Genetic Algorithms and Fuzzy Systems. John Wiley and sons, Inc. 1995.

[Koza 96] Koza R. John. Genetic Programming II. Automatic Discovery of Reusable Programs. Cambridge. MIT Press. 1996.

[Koza 92] Koza R. John. Genetic Programming. On The Programming Of Computers By Means Of Natural Selection. Massachusetts Institute of Technology. 1992.

[Masini 80] Masini Giancarlo. Sulle Trace della Vita. Nardini Editore–Centro Internazionale del Libro. Firenze, Italy. 1980

[Michalewicz, 99] Michalewicz, Zbigniew. Genetic Algoritms + Data Structures = Evolution Programs. 3ª Ed. Springer - Verlag. Berlín. 1999. 387 p.

[Mitchell 97] Mitchell, Melanie. An Introduction To Genetic Algorithms. Massachusetts Institute of Technology. 1997

[Pastor 95] Pastor Oscar, Ramos Isidro. OASIS 2.1.1.: A Class-Definition Language to Model Information Systems Using an Object-Oriented Approach. Depto. Sistemas Informáticos y Computación. Universidad Politécnica de Valencia. España.

[Reeves 91] Reeves, C. R. An introduction to Genetic Algorithms. Operational Research, Tutorial Papers. 1991.

[Tello 89] Tello, Ernest R. Object Oriented Programing for Artificial Intelligence. Addison Wesley.

Massachusets; 1989.

[Velásquez 98] Velásquez, Juan D., Hernández R. Jesús A. Hydrological Forecasting via Genetic Programming. Hydroinformatics-98. Danish Hydraulic Institute. Copenhagen. 1998.