INGENIERÍA Y DESARROLLO

ARTÍCULO CIENTÍFICO / RESEARCH ARTICLE

Detección de daño estructural por algoritmos genéticos: una comparación de diferentes tipos de codificación de individuos

Structural damage detection by using genetic algorithms: a comparison of different types of individual coding

Jesús Daniel Villalba Morales*
José Elías Laier**
Universidad de Sao Paulo (Brasil)

* Maestro en Ingeniería de Estructuras, Universidade de São Paulo (Brazil). *villalba@sc.usp.br.*

Correspondencia: Escola de Engenharia de São Carlos. Universidade de São Paulo. Av. Trabalhador São Carlense 400. CEP: 13566-590, Brasil.

** Doctor en Ingeniería de Estructuras, Universidade de São Paulo. Professor Titular do Departamento de Engenharia de Estruturas da Escola de Engenharia de São Carlos- Universidade de São Paulo (Brazil. *jelaier@sc.usp.br.*

Número 27
Enero-junio, 2010
ISSN: 0122-3461

Resumen

En este artículo se aplican los algoritmos genéticos para resolver el problema de detección de daño estructural, y se comparan tres tipos de codificación de individuos: binaria, real y binaria con código redundante. Los algoritmos de código binario y real calculan la extensión del daño para cada elemento en la estructura, por lo tanto, para mejorar su desempeño fue utilizado un proceso de re-inicialización de individuos. Por su parte, el algoritmo de código redundante busca en forma dinámica cuáles son los elementos dañados y cuantifica el daño solo para esos elementos. Para determinar el desempeño de los algoritmos se analiza una armadura sobre diversos escenarios de daño simple y múltiple, y el daño es considerado como una reducción en el módulo de elasticidad de los elementos dañados. Los resultados muestran que el algoritmo que mejor localiza y cuantifica el daño corresponde al algoritmo genético de código redundante.

Palabras clave: Detección de daño, algoritmos genéticos, parámetros dinámicos

Fecha de recepción: 2 de septiembre de 2009
Fecha de aceptación: 13 de abril de 2010

Abstract

In this paper, genetic algorithms are used to solve the structural damage detection problem. Three types of representation of individuals are compared: binary, real and binary with redundant representation. The binary and real-coded algorithms compute the damage extension for each element in the structure, therefore, a re-starting process of individuals is used. The redundant representation algorithm searches the damaged elements in a dynamic way and quantifies the damage for these elements only. A truss structure under different damage scenarios is analyzed, being damage considered a reduction in the elasticity module of the damaged element. Results show that the redundant representation algorithm presents the best option to locate and quantify damage in a structure.

Keywords: Damage detection, genetic algorithms, dynamic parameters.

1. INTRODUCCIÓN

Diversos factores como el envejecimiento, la fatiga o la exposición a ambientes extremos pueden ocasionar daño en una estructura, cuyo grado va desde ligero hasta la puesta fuera de funcionamiento o el colapso de la estructura. Lo anterior, trae consecuencias negativas desde un punto de vista económico y de seguridad de los ocupantes, por lo tanto, es de gran importancia desarrollar técnicas que permitan determinar la presencia del

daño y, con esto, ayudar en la toma de decisiones de reforzamiento o de rehabilitación.

Una de las principales formas para detectar daño es mediante el análisis de las variaciones en los parámetros dinámicos (frecuencias naturales, formas modales y amortiguamiento modal) que se originan debido a los cambios causados por el daño en las propiedades de la estructura (rigidez, amortiguamiento y masa). Siendo así, se puede formular un problema de optimización tal que se minimice la diferencia entre los parámetros dinámicos experimentales de la estructura actual y aquellos provenientes de un modelo de elementos finitos, el cual representaría la condición con daño. El modelo de la estructura con daño es obtenido a partir de la actualización del modelo que define la condición inicial de la estructura.

Una técnica que puede ser utilizada para resolver el problema anterior es la metaheurística conocida como Algoritmos Genéticos, la cual hace una analogía con las leyes de selección natural y sobrevivencia del más fuerte para llevar un conjunto de posibles soluciones (o población) a evolucionar y así encontrar una respuesta a un problema determinado. Esas soluciones (o individuos), en problemas que involucran variables continuas, son codificadas en vectores (o cromosomas) utilizando números binarios o reales. La aptitud (o *fitness*) de cada individuo se determina a partir del cálculo de la función objetivo. En problemas de minimización, los mejores individuos son aquellos que presentan los valores más bajos de *fitness* en la población y los valores más altos en el caso de un problema de maximización. Así, el *fitness* de un individuo determina la probabilidad con la cual será seleccionado para generar nuevos individuos. La generación de una nueva población de individuos requiere de tres operaciones: selección, cruzamiento y mutación. El lector puede dirigirse a textos clásicos como [1] o [2] para una referencia completa sobre los algoritmos genéticos.

En la literatura se encuentran diversas metodologías que utilizan algoritmos genéticos para resolver el problema de detección de daño en estructuras, y uno de los primeros trabajos propuestos es el de Mares y Surace en 1996 [3], quien emplea un algoritmo genético con codificación binaria y una función objetivo basada en el vector de fuerza residual. Este tipo de función tiene como desventaja que requiere formas modales completas, lo cual en la práctica, generalmente, no es posible debido a razones técnicas y económicas. Au et

al. [4] propusieron detectar daño estructural en dos etapas. En la primera etapa, se determina un conjunto de elementos probablemente dañados a partir de una metodología de localización de elementos con daño basada en energía. En la segunda etapa, se cuantifica el daño utilizando un micro algoritmo genético, el cual realiza un proceso duplo de optimización donde se busca por la combinación óptima de elementos dañados y extensiones de daño que minimiza una función objetivo basada en frecuencias naturales y formas modales. Raich y Liszkai [5] emplearon la codificación redundante implícita y variaciones en las funciones de respuesta en frecuencia para detectar daño en estructuras. Este tipo de codificación permite que el número de variables al ser optimizadas pueda cambiar durante todo el proceso evolutivo. Esta característica hace que este tipo de codificación sea altamente aplicable para resolver el problema de detección de daño en vista de que el número y la posición de los elementos dañados no son conocidos *a priori*. Laier y Villalba [6] utilizaron un algoritmo genético de código real y una función objetivo basada en los cambios en las frecuencias naturales y las formas modales. En dicho trabajo se propuso un proceso de re-inicialización de individuos que consistía en que cada uno, en la nueva población, estaba conformado solo por aquellos elementos que presentaban una cantidad mínima de daño en el escenario de daño que fue encontrado al final de la primera ejecución.

Finalmente, en este trabajo se realiza una comparación del desempeño de algoritmos genéticos que utilizan diferentes tipos de codificación para detectar daño estructural. Es importante anotar que los siguientes aspectos fueron considerados: detección de niveles bajos de daño y pocos elementos dañados en la estructura.

2. DEFINICIÓN DEL PROBLEMA DE DETECCIÓN DE DAÑO

Representación del daño

En este trabajo, el daño es representado como una reducción en el módulo de elasticidad del material siguiendo el principio de las deformaciones equivalente [7]. Utilizando el modelo de elementos finitos de la estructura, el daño en el elemento j se representa como:

$$E_{dj} = \left(1 - \beta_j\right) \times E_j \tag{1}$$

en el cual E_{dj} y E_j son los módulos de elasticidad del material del elemento j para las condiciones con y sin daño, respectivamente. β_j es un factor de reducción de la rigidez, el cual asume un valor de cero (0) si el elemento no se encuentra dañado y un valor de uno (1) para representar la pérdida del elemento. Los factores de reducción para cada elemento en la estructura corresponden a las variables que van a ser optimizadas por los algoritmos genéticos.

A continuación se muestra cómo afecta el daño a los parámetros dinámicos de las estructuras. Para una estructura sin amortiguamiento, la ecuación dinámica puede ser formulada como:

$$M\ddot{u} + Ku = 0 \tag{2}$$

siendo M y K las matrices de masa y rigidez, respectivamente, y u el vector de desplazamientos. La solución para vibración libre es dada por:

$$\left(K - \omega^2 M\right)\Phi = 0 \tag{3}$$

con ω y Φ siendo las frecuencias naturales y las formas modales de la estructura, respectivamente. La ecuación (3) aún es válida para la condición con daño, y si se considera que la matriz de masa permanece constante se tiene finalmente:

$$\left(K_d - \omega_d^2 M\right)\Phi_d = 0 \tag{4}$$

en la cual d se refiere a la condición con daño.

■ *Problema de optimización*

El problema de detección de daño fue formulado como un problema de optimización, como se ilustra en la figura 1. En el paso 1, se debe definir un modelo de elementos finitos de la estructura inicial. A continuación, y debido a que esta investigación se llevó a cabo solo en forma numérica, se necesita la realización de los pasos 2 al 4 con el fin de simular los parámetros dinámicos de la estructura con daño. En el paso 2, se introduce directamente

el escenario de daño, que se desea determinar, dentro de la matriz de rigidez del modelo obtenido en el paso 1, generando así el modelo correspondiente a la estructura con daño. A partir de este nuevo modelo, los parámetros dinámicos de la estructura con daño son determinados utilizando la ecuación (4). En el paso 4, se introduce ruido en los parámetros dinámicos para simular el hecho que las mediciones experimentales no son realizadas en forma exacta. Si se cuenta con los parámetros dinámicos de la estructura con daño, estos serían utilizados directamente sin tener que implementar los pasos 2 al 4. En el paso 5, se define la función objetivo en términos de las diferencias entre los parámetros dinámicos simulados, o experimentales, y aquellos obtenidos a partir del algoritmo genético. En el paso 6, se escoge el tipo de codificación, se definen los operadores y parámetros genéticos y se aplica el algoritmo. Finalmente, se muestra el escenario de daño resultante.

Inicio

1. Definir el modelo de elementos finitos, MEF, de la estructura sin daño.
2. Insertar el escenario de daño en el MEF generado en el paso 1.
3. Determinar los parámetros dinámicos de la estructura con daño.
4. Introducir ruido en los parámetros calculados en el paso 3.
5. Definir la función objetivo en términos de parámetros dinámicos.
6. Escoger, configurar y aplicar el algoritmo genético.
7. Imprimir resultados. Mostrar elementos dañados y extensiones de daño.

Fin

Figura 1. Metodología para detección de daño estructural

■ *Simulación de ruido*

Para introducir ruido en los parámetros dinámicos obtenidos en el paso 3 de la figura 1 se utilizan las siguientes expresiones:

$$\omega_{nr} = \omega_n \times \left(1 + rand(-1,1) \times Ruido_f\right) \quad (5)$$

$$\phi_{ijnr} = \phi_{ijn} \times (1 + rand(-1,1) \times Ruido_{\emptyset}) \quad (6)$$

en el cual ω_n y Φ_{ijn} corresponden a las frecuencias naturales y las formas modales originales, respectivamente. ω_{nr} y Φ_{ijnr} son los valores de los anteriores parámetros contaminados con ruido y que serán considerados como experimentales. $Ruido_f$ y $Ruido_\Phi$ son los porcentajes de ruido introducidos, y para este trabajo es de 1% y 3%, respectivamente, tal como en [8].

■ *Función objetivo*

El problema de optimización puede ser formulado como la maximización de la siguiente función objetivo, la cual está basada en frecuencias naturales y formas modales:

$$Max\ G = \sum_{j=1}^{nm} \frac{c_1}{c_2 + F_j} \tag{7}$$

con

$$F_j = \left| \frac{\omega_j^{ag} - \omega_j^{ex}}{\omega_j^{ex}} \right| + W \times \underbrace{Max}_{i=1\ldots ngll} \left| \frac{\emptyset_{ij}^{ag} - \emptyset_{ij}^{ex}}{\emptyset_{ij}^{ex}} \right| \tag{8}$$

en el cual ω_j corresponde a la frecuencia natural del modo j. Φ_{ij} es la coordenada modal en el grado de libertad leído i del modo j. *ex* corresponde a un dato obtenido en forma experimental, y en este trabajo es simulado. *ag* corresponde a un valor encontrado por el algoritmo genético. *nm* es el número de modos utilizados y *ngll* es el número de grados de libertad donde se han realizado mediciones. W, c_1 y c_2 son parámetros definidos por el usuario y cuyos valores fueron asumidos en este trabajo como 2, 200 y 1 respectivamente. Los valores anteriores fueron definidos después de algunas ejecuciones previas del algoritmo en función de su desempeño.

■ *Algoritmos genéticos para detección de daño*

Tres tipos de codificaciones (ver figura 2) fueron utilizadas para mostrar el desempeño de los algoritmos genéticos en la resolución del problema de detección de daño. El primer tipo corresponde a un algoritmo de codificación binaria, Cobin; el segundo, a un algoritmo de código real, CoRea,

como fue propuesto en [6] y, finalmente, se implementó un algoritmo genético binario con codificación redundante implícita, CoRed, según fue propuesto en [5]. El esquema de aplicación de un algoritmo genético es presentado en la figura 3.

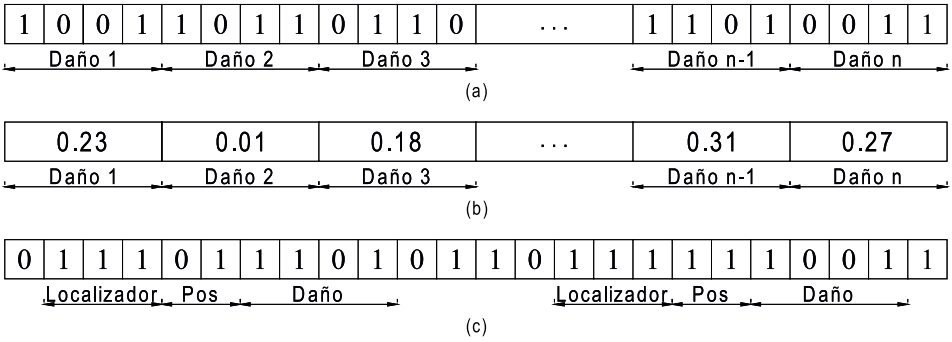

Figura 2 Tipo de algoritmos genéticos utilizados. (a) Código Binario, (b) Código real y (c) Código de codificación redundante.

Inicio

1. Generar la población inicial.
2. Calcular el fitness de los individuos de la población actual
3. Seleccionar los mejores individuos de la población.
4. Cruzar los individuos seleccionados.
5. Mutar individuos generados en el proceso de cruzamiento.
6. Aplicar elitismo y calcular el fitness de los nuevos individuos.
7. Verificar criterios de parada. En el caso que los criterios no sean cumplidos, aplicar nuevamente los pasos 3-7.

Fin

Figura 3. Proceso de ejecución de un algoritmo genético

Los parámetros y los operadores utilizados por cada uno de los algoritmos genéticos se muestran en las tablas 1 y 2. Los valores de los parámetros fueron definidos después de algunas ejecuciones preliminares de los algoritmos. Como criterio de parada se consideraron 50 generaciones sucesivas en las cuales no hubiese un cambio significativo en el valor del *fitness* del mejor individuo o un número máximo de 500 generaciones. La forma de aplicación de los distintos operadores puede ser encontrada en textos clásicos, o artículos, sobre algoritmos genéticos [2] y [9].

Tabla 1. Definición de los operadores de los diferentes algoritmos genéticos implementados.

Operador	CoBin	CoRea	CoRed
Selección	Torneo	Torneo	Ruleta
Cruzamiento	Dos Puntos	BLX-α	Dos puntos
Mutación	Jump	Creep	Jump

Tabla 2. Definición de los parámetros de los diferentes algoritmos genéticos implementados.

Parámetro	CoBin	CoRea	CoRed
Tasa de Cruzamiento	0.85	0.90	0.85
Tasa de Mutación	0.01	0.05	0.005

A continuación se presenta la forma como se determinó el tamaño del cromosoma que sirve para representar una solución según cada tipo de codificación. En el algoritmo CoBin un factor de reducción, β, debe ser calculado para cada elemento. Cada gen es codificado por una secuencia de 1 y 0, como explicado en [2]. Para determinar el número de *bits* representando cada gen, *Tam_Gen*, se considera que el factor de reducción de la rigidez tiene valores mínimo y máximo de 0 y 1, respectivamente, y que la precisión es de 3 cifras decimales. A partir de estas condiciones, cada factor de reducción necesita de 10 *bits* para ser codificado. El tamaño del

cromosoma puede ser calculado en función del número de elementos en la estructura, *NumElem*, como:

$$Tam\_Cromo_{CoBin} = Tam\_Gen \times NumElem \quad (9)$$

Al igual que para el caso de CoBin, el algoritmo CoRea necesita representar cada factor de reducción con una variable, pero con la ventaja de que no requiere de procesos de codificación, por lo tanto, el tamaño del cromosoma queda:

$$Tam\_Cromo_{CoRea} = NumElem \quad (10)$$

Por otro lado, el algoritmo binario con codificación redundante no codifica en forma explícita el número de variables que se encuentra presente en cada cromosoma [10]. Con esto, los cromosomas dentro de la población, y de una generación para otra, no necesariamente contienen la misma cantidad de genes. Por lo tanto, para la aplicación de CoRed fue definida en [10] una forma de identificar la presencia de un gen dentro del cromosoma y para lo cual se utilizará un localizador de genes, que corresponde a una secuencia de tres 1 consecutivos. El proceso consiste en recorrer de izquierda a derecha el cromosoma buscando por el localizador de genes. Un gen está presente cada vez que el localizador es encontrado. En la figura 2 se observa un localizador de genes entre los *bits* 2 y 4 con su correspondiente gen comenzando a partir del *bit* 5. En el problema estudiado, el gen se divide en dos segmentos, siendo el primero para codificar el número del elemento que se encuentra dañado y el segundo utilizado para expresar el valor de daño (ver figura 2). En este caso el tamaño del gen es dado por:

$$Tam\_Gen = Tam\_Loc + Tam\_IdElem + Tam\_Ext \quad (11)$$

en el cual *Tam_Loc*, *Tam_IdElem* y *Tam_ext* son el número de *bits* necesarios para codificar el localizador de genes, el identificador del elemento dañado y la extensión del daño, respectivamente. Utilizando 3 *bits* para el localizador de genes, 6 *bits* para codificar el segmento correspondiente al elemento dañado y 10 para la magnitud del daño tendremos un gen con 19 *bits*.

El tamaño del cromosoma es dado por la siguiente heurística:

$$Tam\_Cromo_{CoRed} = 0.4 \times Tam\_Gen \times NumElem \qquad (12)$$

La forma de aplicación de este tipo de algoritmo es igual a aquella realizada con el algoritmo de código binario con excepción de que debe ser introducido el proceso de determinación de los genes presentes en el cromosoma. Un problema relacionado a este proceso está en la posibilidad que en un mismo cromosoma, dos o más genes puedan corresponder al mismo elemento de la estructura, razón por la cual fue considerado el promedio del daño en estos genes. Así mismo, se pueden generar posiciones de elementos mayores que el tamaño de la estructura, lo cual fue resuelto generando una posición aleatoria que fuera válida.

Finalmente, se utiliza una estrategia elitista en los tres algoritmos para evitar una posible pérdida del mejor individuo entre dos generaciones consecutivas, la cual es ocasionada por las características de los operadores genéticos.

- ***Heurísticas utilizadas***

La primera heurística utilizada en este trabajo conduce a los algoritmos a realizar la búsqueda por la solución del problema en regiones del espacio de soluciones que presenten valores de factores de reducción que sean bajos. En el caso del algoritmo CoBin, el primer *bit* de cada gen fue colocado como cero, lo cual garantiza que los números generados en la población inicial no sean mayores a 0.5 ($\beta_i \leq 0.5$). Para el caso del algoritmo CoRea, esa restricción fue impuesta directamente en cada gen, tal que el número aleatorio generado podía asumir únicamente valores entre 0 y 0.5.

Considerando que el número total de elementos dañados es relativamente menor que la cantidad de elementos en la estructura, se propone una segunda heurística para los algoritmos CoBin y CoRea. Siendo así, se deben generar individuos en la población inicial que representen escenarios de daño donde únicamente un porcentaje de los elementos de la estructura presenten daño. Para los ejemplos que serán estudiados ese porcentaje fue definido como el 40% del total de elementos en la estructura. La heurística para la generación de la población inicial consiste en considerar que si un número aleatorio generado es mayor que 0.4, el gen del individuo asume

un valor de cero, y si es menor recibe un segundo valor aleatorio, según fue explicado en el párrafo anterior.

Por otro lado, la presencia de pequeños valores de daño en elementos falsamente identificados como dañados puede influenciar en la cuantificación del daño en los elementos verdaderamente dañados. Lo anterior, llevó a utilizar el proceso de re-inicialización propuesto en [6], considerando como variables únicamente aquellos elementos para los cuales fue obtenido un valor del factor de reducción mayor a 3%. La respuesta que presente el mayor valor de la función objetivo entre las dos ejecuciones corresponderá a la solución del problema.

Las heurísticas anteriores no se aplicaron al algoritmo binario con codificación redundante debido a que la cantidad y la posición de los genes en un cromosoma no se encuentran predefinidas en el comienzo de la ejecución del algoritmo.

3. ESTRUCTURA Y ESCENARIOS DE DAÑO ANALIZADOS

En la literatura técnica es común encontrar armaduras para evaluar el desempeño de la metodología de detección de daño que está siendo desarrollada. Siendo así, en este trabajo se utiliza la armadura presentada en la figura 4, y cuyos elementos horizontales y verticales miden 1m.

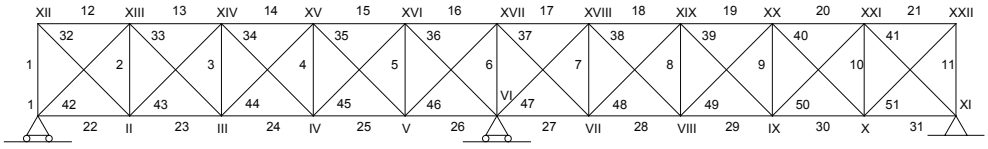

Figura 4. Estructura analizada

Todos los elementos en la armadura presentan el mismo tipo de sección transversal y un único tipo de material. Las características de la armadura se muestran en la tabla 3.

Tabla 3. Definición del tipo de material y sección utilizada en la armadura.

Tipo	Propiedades	Valor
Material	Módulo de elasticidad	70.3 GPa
	Densidad	2685 Kg/m^3
Sección	Área	1E-3 m^2

Los diferentes escenarios de daño aplicados en la armadura se muestran en la tabla 4. Los tres primeros escenarios representan daño en un elemento vertical, un horizontal y un diagonal, respectivamente, y el último escenario corresponde a un escenario de daño múltiple.

Tabla 4. Escenarios de daño

Escenario	Elemento	% Daño (β)
A1	4	0.2
A2	23	0.2
A3	43	0.2
A4	4	0.2
	23	0.2
	43	0.2

Para poder ejecutar el algoritmo genético aún se deben definir los valores de otros dos parámetros. El primero corresponde al número de modos que serán utilizados en el cálculo del *fitness* de los individuos según la ecuación 7. Así, en este trabajo se considera que la información sobre los 8 primeros modos se encuentra disponible. El segundo parámetro corresponde al número de individuos de la población; sin embargo, actualmente no se cuenta con una expresión que permita relacionar el tamaño de la estructura con el número de individuos necesarios para obtener un buen desempeño del algoritmo genético. Por esto, se realizaron algunas ejecuciones preliminares del algoritmo, con diferentes cantidades de individuos, y se observó que un valor recomendable sería de 200.

4. RESULTADOS

Determinación de los escenarios de daño

En la tabla 5 se encuentra el resumen de los resultados obtenidos por los algoritmos genéticos para los 4 escenarios de daño estudiados. Para el escenario de daño simple A1 puede ser observado que los tres algoritmos encuentran el elemento dañado y el valor de la extensión del daño. En el caso de los escenarios A2, A3 y A4 los algoritmos CoBin y CoRea detectan varios elementos considerados erróneamente como dañados, pero con valores pequeños de daño. En lo relacionado a la extensión de daño, el algoritmo CoRed detecta casi en forma exacta el valor del daño para los escenarios de daño simples y con un error menor a 0.015 en el escenario de daño múltiple.

En todos los escenarios de daño determinados por este algoritmo no se observa la presencia de elementos falsamente identificados como dañados. El error del algoritmo CoBin en la determinación de la extensión del daño en los elementos verdaderamente dañados es menor a 0.05; pero presenta el elemento 10 con un valor alto de daño en el escenario de daño múltiplo. Los valores de daño del algoritmo CoRea fueron más exactos si los comparamos con aquellos obtenidos con el algoritmo CoBin.

En la tabla 6 pueden ser observados los valores de *fitness* alcanzados por las soluciones presentadas en la tabla 5. El valor máximo que puede ser alcanzado por la ecuación (7) utilizando 8 modos de vibración es de 1600. En el caso del escenario de daño A2 es observado que el algoritmo alcanza el valor exacto del daño, pero su valor de *fitness* es menor al máximo. Este problema es derivado del hecho que las mediciones de los parámetros dinámicos contienen ruido.

Análisis de la convergencia de los algoritmos CoBin y CoRea

En las figuras 5 y 6 se muestra la evolución del mejor individuo para los algoritmos genéticos de código binario y código real, respectivamente, cuando fueron aplicados para detectar el escenario de daño A4. En dichas gráficas se observa cómo la utilización de la heurística para la generación de la población inicial y el proceso de re-inicialización mejoran la solución final encontrada por el algoritmo.

Tabla 5. Aplicación de los algoritmos genéticos en los escenarios de daño

Escenario de daño	Daño Real		CoBin		CoRea		CoRed	
	Elem	β	Elem	β	Elem	β	Elem	β
A1	3	0.2	3	0.199	2 3	0.01 0.201	3	0.199
A2	23	0.2	2 4 23 35 38 44	0.014 0.033 0.190 0.012 0.011 0.039	2 4 10 23	0.013 0.011 0.011 0.198	23	0.2
A3	43	0.2	43	0.199	8 10 43	0.021 0.01 0.196	43	0.201
A4	3 23 43	0.2 0.2 0.2	2 3 10 23 28 35 38 43	0.038 0.148 0.105 0.203 0.017 0.015 0.016 0.199	2 3 8 10 23 35 43 44	0.024 0.205 0.011 0.044 0.197 0.013 0.197 0.018	3 23 43	0.186 0.201 0.197

Tabla 6. *Fitness* del individuo solución

Escenario de daño	*Fitness*		
	CoBin	CoRea	CoRed
A1	1502.4	1504.5	1505.7
A2	1406.9	1496.1	1501.7
A3	1502.6	1503.9	1504.2
A4	1377.0	1487.3	1464.0

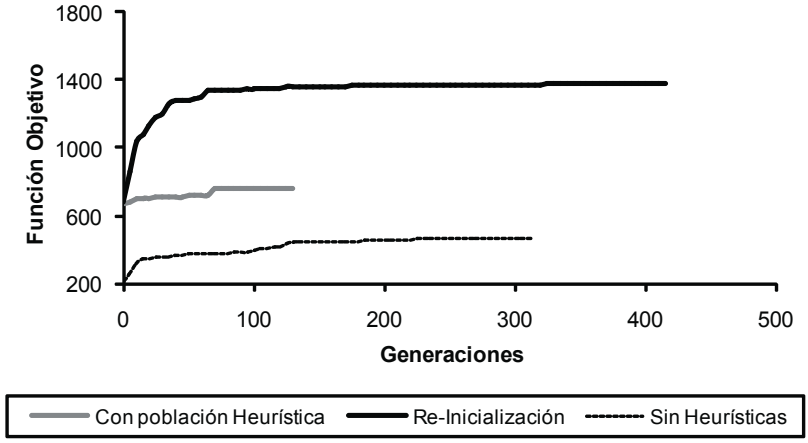

Figura 5. Evolución del *fitness* del mejor individuo para el algoritmo genético de código binario

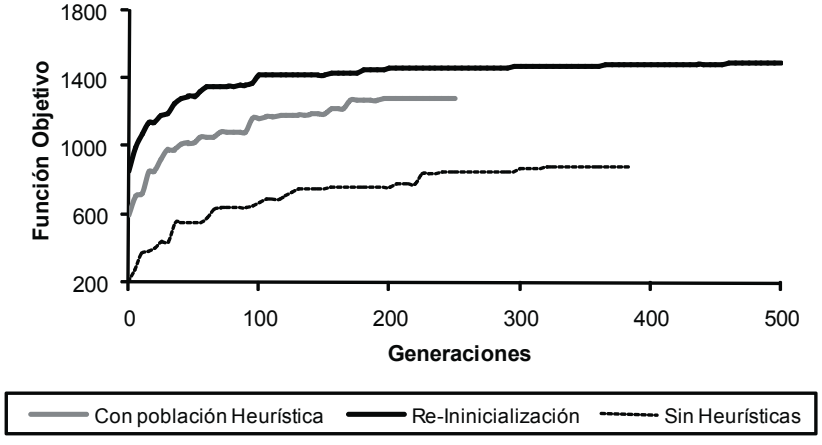

Figura 6. Evolución del *fitness* del mejor individuo para el algoritmo genético de código real

5. CONCLUSIONES

En este trabajo se estudió la aplicación de algoritmos genéticos para la resolución del problema de detección de daño mediante la implementación de tres tipos de codificación de individuos. Es observada la capacidad que tienen los tres tipos de codificaciones para resolver el problema de detección de daño, y por lo tanto muestra la capacidad que tienen los algoritmos genéticos para resolver problemas de difícil solución, en especial cuando la información utilizada para calcular la función objetivo contiene ruido. Los algoritmos CoBin y Corea consiguen detectar en forma correcta los elementos dañados y en forma aproximada la extensión del daño, presentando algunos elementos falsamente identificados como dañados con valores pequeños de daño. Las dos heurísticas propuestas mostraron mejorar la solución final del algoritmo. El algoritmo CoRed mostró un excelente comportamiento tanto en la localización como en la cuantificación de daño, sin presencia de elementos falsamente identificados como dañados para los escenarios de daño estudiados. Este desempeño puede ser atribuido al hecho de que el algoritmo CoRed busca en forma dinámica los elementos dañados durante todo el proceso evolutivo.

Finalmente, nuevas investigaciones sobre la utilización de algoritmos genéticos para detección de daño pueden ser orientadas hacia la obtención de algoritmos auto-configurados tal que la definición de los parámetros, como tasa de cruzamiento y tasa de mutación, no sea realizada a partir de ejecuciones previas del algoritmo.

Agradecimientos

Los autores agradecen el apoyo financiero de la CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior- Brasil) sin el cual la realización de este trabajo no habría sido posible.

Referencias

[1] D. Goldberg, *Genetic algorithms in search optimization, and machine learning,* Reading, MA, United States: Addison-Wesley Publishing Company, 1989.

[2] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs,* 2da ed., Berlin: Springer-Verlag, 1994.

[3] C. Mares and C. Surace, "An application of genetic algorithms to identify damage in elastic structures," *Journal of Sound and Vibration*, vol. 195, no. 2, pp. 195-215, 1996.

[4] F. Au, Y.S. Cheng, L.G. Tham and Z.Z. Bai, "Structural damage detection based on a micro-genetic algorithm using incomplete and noisy modal test data," J*ournal of Sound and Vibration*, vol. 259, no. 5, pp. 1081–1094, 2003.

[5] A. Raich and T. Liszkai, "Improving the performance of structural damage detection methods using advanced genetic algorithms," *Journal of Structural Engineering*, vol. 133, no. 3, pp. 449-461, 2007.

[6] J. E. Laier and J.D. Villalba, "Improved genetic algorithm for structural damage detection," In: *Proceedings of the 2009 Computer Structural Engineering Symposium*, Shanghai, China, pp. 833-839, 2009.

[7] J. Lemaitre, *A course on damage mechanics*. Berlin: Springer, 1996, pp. 13-14.

[8] B. Chen and S. Nagarajaiah, "Flexibility-based structural damage identification using Gauss- Newton method, " In: *Proceedings of SPIE- Sensors and smart structures, technologies for civil, mechanical, and aerospace systems*, San Diego, USA, Vol 6529 Part 1, paper 65291L, 2007.

[9] F. Herrera, M. Lozano and J.L. Verdergay, "Tackling real-coded genetic algorithms: operators and tools for behavioral analysis," *Artificial Intelligence Review*, no. 12, pp. 265-319, 1998.

[10] A. Raich and J. Ghaboussi, "Implicit representation in genetic algorithms using redundancy," *Evolutionary Computation*, vol. 5, no. 3, pp. 277-302, 1998.