Apéndice II

ALGORITMOS GENÉTICOS

AII.1 INTRODUCCIÓN.

Los algoritmos genéticos (AG) son una técnica metaheurística de búsqueda basada en la teoría de la evolución de Darwin. Fue desarrollada por primera vez por Holland en 1975 [81]. El objetivo de su creador era lograr que las computadoras aprendieran por sí mismas.

En la actualidad los AG han alcanzado una gran madurez y quizás sean reconocidos como la técnica metaheurística por excelencia y una de las más empleadas en problemas de optimización. Por ello, los AG son utilizados como técnica de referencia o comparativa para la validación de otras técnicas de búsqueda no exhaustiva. El objeto de este apéndice es presentar los conceptos básicos de los algoritmos genéticos y su implementación básica.

El AG se basa en los mecanismos de selección que utiliza la naturaleza, de acuerdo a 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 los atributos que le permiten adaptarse mejor a su entorno, se transmiten a sus descendientes cuando éste se reproduce sexualmente.

El algoritmo genético transforma un conjunto de objetos matemáticos individuales con respecto al tiempo usando operaciones modeladas de acuerdo al principio de reproducción y supervivencia del más apto, y tras haberse presentado de forma natural una serie de operaciones genéticas de entre las que destaca la recombinación sexual. Cada uno de estos objetos matemáticos suele ser una cadena de caracteres (letras o números) de longitud fija que se ajusta al modelo de las cadenas de cromosomas, y se les asocia con una cierta función matemática que refleja su aptitud.

Los AG son métodos adaptativos que pueden usarse para resolver problemas de búsqueda y optimización. La evolución de las soluciones que proporciona el AG hacia valores óptimos del problema depende en buena medida de una adecuada codificación de las mismas. El algoritmo genético consiste básicamente en una función matemática o una rutina de software que toma como entradas a los ejemplares y retorna como salidas cuales de ellos deben generar descendencia para la nueva generación.

AII.2 PRINCIPIOS BÁSICOS Y CARACTERÍSTICAS DE LOS AG

En la naturaleza, los individuos de una población compiten entre sí en la búsqueda de comida, agua, refugio, compañero, etc. Los 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. De esta manera, las especies evolucionan logrando unas características cada vez mejor adaptadas al entorno en el que viven.

Los AG 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 ó puntuación, relacionado con la bondad o aptitud de dicha solución. 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, 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.

El cruce entre los individuos seleccionados produce descendientes, los cuales comparten algunas de las características de sus padres. De este modo 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. Así, 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 AG ha sido bien diseñado, la población convergerá hacia una solución óptima del problema.

El poder de los AG 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 AG 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.

Si existen 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 combinándolas con los AG.

Los AG han demostrado ser muy eficientes y confiables para la búsqueda de la solución en problemas de optimización. Sin embargo, los AG pueden no ser apropiados para algunos problemas, por ello se recomienda en general tomar en cuenta las siguientes características del mismo antes de su aplicación:

- Lo más recomendable es intentar resolver problemas que tengan espacios de búsqueda discretos aunque éstos sean muy grandes. Sin embargo, también puede intentar usarse con espacios de búsqueda continuos, pero preferentemente cuando exista un rango de soluciones relativamente pequeño.

- Se debe definir una función de aptitud, adaptación o fitness que nos indique lo buena o mala que es una respuesta o solución.

- Las soluciones deben codificarse de una forma que resulte relativamente fácil su implementación.

La función de aptitud no es más que la función objetivo del problema de optimización. Esta función tiene que ser capaz de "castigar" a las malas soluciones, y de "premiar" a las buenas, de forma que sean estas últimas las que se propaguen con mayor rapidez. La finalidad del algoritmo genético es maximizar o minimizar, en su caso, la función objetivo.

La codificación más común de las soluciones es a través de cadenas binarias, aunque se han utilizado también números reales y letras. El primero de estos esquemas ha gozado de mucha popularidad debido a que es el que propuso originalmente Holland, y además porque resulta muy sencillo de implementar.

AII.3 FUNCIONAMIENTO DE LOS ALGORITMOS GENÉTICOS

En la resolución de un problema mediante un algoritmo genético se realiza la elección y codificación de las variables, se aplican métodos de evolución, selección y reproducción con intercambio de información, y se incluyen alteraciones que generan diversidad.

La población esta formada por una serie de individuos o partículas que pueden ser posibles soluciones del problema. Las partículas se representan como un conjunto de parámetros que se llaman genes, los cuales agrupados forman ristras de valores denominadas cromosomas. Los genes suelen estar representados en codificación binaria. La adaptación de un individuo depende de la evaluación de cada cromosoma usando la función de aptitud que refleja el nivel de adaptación al problema de la partícula representado por el cromosoma.

Durante la fase reproductiva se seleccionan las partículas de la población con mejor aptitud o fitness para cruzarse y producir descendientes, que constituirán, una vez mutados, la siguiente generación de individuos. La selección de padres se efectúa usando un mecanismo que favorezca a los individuos mejor adaptados, ya que a cada individuo se le asigna una probabilidad de ser seleccionado que es proporcional a su función de aptitud.

Una vez seleccionados dos padres, sus cromosomas se combinan, utilizando habitualmente los operadores de cruce y mutación. Las formas básicas de dichos operadores se describen a continuación.

El operador de cruce, coge dos padres seleccionados y corta sus ristras de cromosomas en una posición escogida al azar, para producir dos ristras "A" y dos ristras "B". Después se intercambian las ristras generadas, produciéndose dos nuevos cromosomas completos. Ambos descendientes heredan genes de cada uno de los padres. Habitualmente el operador de cruce no se aplica a todos los pares de individuos que han sido seleccionados para emparejarse, sino que se aplica de manera aleatoria, normalmente con una probabilidad comprendida entre 0.5 y 1.0. A este factor se le denomina tasa o índice de selección, *IS*. En la figura II.1 se observa la operación de cruce:

PADRE 1			**PADRE 2**	
1 1 1 0	0 1 0 1 0 1		1 1 0 0	0 1 0 1 1 0
ristra A1	*ristra B1*		*ristra A2*	*ristra B2*
C R U C E				
ristra A1	*ristra B2*		*ristra A2*	*ristra B2*
1 1 1 0	0 1 0 1 1 0		1 1 0 0	0 1 0 1 0 1
DESCENDIENTE 1			**DESCENDIENTE 2**	

Figura II.1: Cruce de cromosomas en los AG.

El operador de mutación se aplica a cada hijo de manera individual, y consiste en la alteración aleatoria, normalmente con una pequeña probabilidad asignada denominada tasa de mutación, de cada gen componente del cromosoma. La figura II.2 muestra la mutación del quinto gen del cromosoma. Sí bien puede en principio pensarse que el operador de cruce es más importante que el operador de mutación, ya que proporciona una exploración rápida del espacio de búsqueda, éste último asegura que ningún punto del espacio de búsqueda tenga probabilidad cero de ser examinado, y es de capital importancia para asegurar la convergencia de los AG.

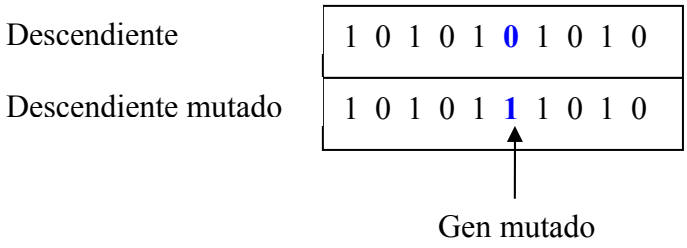

Figura II.2: Mutación de un gen.

Si el algoritmo genético ha sido correctamente implementado, la población evolucionará a lo largo de las generaciones sucesivas de tal manera que la adaptación media extendida a todos los individuos de la población, así como la adaptación del mejor individuo se irán incrementando hacia el óptimo global.

AII.4 PSEUDOCODIGO ALGORITMO GENÉTICO

En base a lo anterior, el algoritmo genético se puede representar por el siguiente pseudocódigo:

PSEUDOCÓDIGO DEL ALGORITMO GENÉTICO

Población ← Generar una población inicial

Mientras no se alcance la Condición de parada **hacer**

Para *i* = 1 hasta tamaño (*Población*)/2 **hacer**

Seleccionar 2 individuos de la anterior generación para el cruce (la probabilidad de selección es proporcional a la función de evaluación del individuo).

Cruzar con cierta probabilidad los 2 individuos obteniendo 2 descendientes.

Mutar los dos descendientes con cierta probabilidad.

Calcular la función de evaluación de los dos descendientes mutados.

Insertar los dos descendientes mutados en la nueva generación.

Fin Para

Si la población ha convergido **entonces**

La condición de parada se ha alcanzado

Fin Si

Fin Mientras

Salida: Devuelve la mejor solución encontrada.

AII.5 ADAPTACIÓN E IMPLEMENTACION EN MATLAB AL PROBLEMA CONSIDERADO

A continuación se implementa el algoritmo AG para la resolución del problema planteado. La codificación de la partícula es la misma que para el algoritmo PSO. El software empleado para su implementación es MATLAB. El programa principal queda de la siguiente manera:

```
%=========================================================
% ALGORITMOS GENÉTICOS
% AUTOR: Manuel Gómez González
% AÑO 2008
%=========================================================
clc;
clear all; close all;
%=========================================================
disp('EMPLAZAMIENTO DE CENTRAL DE PROCESADO DE BIOMASA MEDIANTE AG');
%=========================================================
%Parametros de ajuste del AG
inicodif;  %Se definen los parámetros bitsx, bitsy, bitstam
D=bitsi+bitsj+bitstam;  %Dimensión del problema
Tam_AG=36; %Tamaño de la población
NitmaxAG=60; %Número de iteraciones del algoritmo
t_max=0.1; %Probabilidad de mutación inicial o máxima
t_min=0.001; %Probabilidad de mutación final o mínima
t_seleccion=0.8; %Tasa o índice de selección
numparejas=floor(Tam_AG*t_seleccion/2);

fprintf('Numero máximo de iteraciones NitmaxAG: %d\n',NitmaxAG);
fprintf('Dimension del problema D: %d\n',D);
fprintf('Tamaño de la población de partículas Tam_AG: %d\n',Tam_AG);
fprintf('Maxima probabilidad de mutación: %f\n',t_max);
fprintf('Minima probabilidad de mutación: %f\n',t_min);
%=========================================================================
T=LeerTabla('REGION'); %se carga la base de datos de las parcelas
Numpruebas=10; %se definen el número de ejecuciones del algoritmo
PRUEBAS=[];
EVOLUCIONAG=zeros(1,NitmaxAG+1);
%=========================================================================
for h=1:Numpruebas

Poblacion=(rand(Tam_AG,D)>0.5);%Generación aleatoria de las partículas
```

```
fprintf('Evaluacion inicial\n');
fprintf('prueba %d\n',h);
%evaluacion inicial
eva=Evaluacion(Poblacion,T); %se realiza la evaluación inicial
%La función Evaluacion() se encarga de obtener el valor de la aptitud
de todas las partículas de la población

%La población se ordena por el valor del fitness de mayor a menor
[eva,eva_ind]=sort(eva,'descend');
Poblacion=Poblacion(eva_ind,:);
Evolucion_Fitness=[eva(1)];


for It=1:NitmaxAG

    lambda_t=log(t_max/t_min)/(NitmaxAG-1);%decaimiento exponencial
    t_mutacion=t_max*exp(-lambda_t*(It-1));%cálculo tasa de mutación

    fprintf('Prueba: %d, iteracion: %d\n',h,It);
    fprintf('Tasa de mutacion: %g\n',t_mutacion);

    padres=Poblacion(1:2*numparejas,:); %Selección de padres
    pto_cruce=floor(rand(1,numparejas)*(D-1))+1; %elección punto cruce

%Reproducción: el mejor individuo con el segundo mejor, el tercero con
%elcuarto
for I=1:numparejas
hijos(2*I-1,:)=[padres(2*I-
1,1:pto_cruce(I)),padres(2*I,pto_cruce(I)+1:D)];
hijos(2*I,:)=[padres(2*I-
1,pto_cruce(I)+1:D),padres(2*I,1:pto_cruce(I))];
end;
```

```
%Mutación de los hijos o descendientes
    ind_mut=find(rand(2*numparejas,D)<t_mutacion);
    hijos(ind_mut)=~hijos(ind_mut);

%Sustitución de los individuos peores por los hijos
%mejores y se evalúan
    Poblacion(Tam_AG-numparejas*2+1:Tam_AG,:)=hijos;
    evahijos=Evaluacion(hijos,T);
    eva(1,Tam_AG-numparejas*2+1:Tam_AG)=evahijos;

    %la población se ordena por el valor del fitness de mayor a menor
    [eva,eva_ind]=sort(eva,'descend');
    Poblacion=Poblacion(eva_ind,:);

    %se convierten los valores de binario a real
    [f_gbest,c_gbest,tam_gbest]=Codificacion1bin2dec(Poblacion(1,:));


%se calculan y presentan los valores obtenidos en esta iteración
    fprintf('La solucion tiene la central situada en
(%d,%d)\n',f_gbest,c_gbest);
    fprintf('La solucion es una region cuadrada de %f por %f
parcelas\n',2*tam_gbest+1,2*tam_gbest+1);
    fprintf('El indice de rentabilidad que se tiene es %f\n',eva(1));
    fprintf('PRUEBA: %d, ITERACION: %d   FITNESS: %f\n',h,It,eva(1));
    [RegionSolucion]=ObtenerVecinos(T,f_gbest,c_gbest,tam_gbest);
    Pe_solucion=PotenciaProd(RegionSolucion);
    fprintf('La solución produce %f MW\n',Pe_solucion);
    Area_solucion=ObtenerArea(RegionSolucion);
    fprintf('La solución toma Biomasa de %f Km2\n',Area_solucion);
    NPV=ValorActualNeto(RegionSolucion);
    fprintf('El valor actual neto que se tiene es %f\n',NPV);


    Evolucion_Fitness=[Evolucion_Fitness, eva(1)];
```

```
%============================================================
%Ahora sacamos gráficas acerca de los resultados obtenidos
%Solucion obtenida
DibujarSolucion(T,f_gbest,c_gbest,tam_gbest);
%DibujarSolucion representa las parcelas seleccionadas en el mapa

end;

PRUEBAS=[PRUEBAS,eva(1)];
EVOLUCIONAG=EVOLUCIONAG+Evolucion_Fitness;

end;

fprintf('FIN DEL ALGORITMO GENETICO %f\n');
EVOLUCIONAG=EVOLUCIONAG/Numpruebas;
figure;
plot(0:NitmaxAG,EVOLUCIONAG); %Evolución de la función objetivo
figure;
bar(PRUEBAS); %Estadísticas de las ejecuciones realizadas
```