ALGORITMOS GENETICOS

Los algoritmos genéticos son una de las mejores maneras de resolver un problema para el cual poco se sabe. Son un algoritmo muy general y por lo tanto funcionará bien en cualquier espacio de búsqueda. Todo lo que necesita saber es lo que necesita la solución para poder hacerlo bien, y un algoritmo genético será capaz de crear una solución de alta calidad. Los algoritmos genéticos utilizan los principios de selección y evolución para producir varias soluciones a un problema dado.

Los algoritmos genéticos tienden a prosperar en un entorno en el que hay un conjunto muy grande de soluciones candidatas y en el que el espacio de búsqueda es desigual y tiene muchas colinas y valles. Es cierto que los algoritmos genéticos funcionarán bien en cualquier entorno, pero serán superados por más algoritmos específicos de la situación en los espacios de búsqueda más simples. Por lo tanto, se debe tener en cuenta que los algoritmos genéticos no siempre son la mejor opción. A veces pueden tomar bastante tiempo para correr y por lo tanto no siempre es factible para el uso en tiempo real. Sin embargo, son uno de los métodos más poderosos con los que (relativamente) se pueden crear rápidamente soluciones de alta calidad a un problema.

Contents

Problema

Se requiere realizar la implementación de un algoritmo que permita encontrar el máximo valor para una función dada, la cual depende de 2 variables

Solución

Se implementará un algoritmo genético el cual se describirá a continuación:

1) Ingreso de variables

Se ingresan las variables. (Para facilidad, se encuentran ya establecidas)

```
%numero_variables=input('Digite el numero de variables');
numero_variables=2;
%numero_generaciones=input('Digite el numero de generaciones');
numero_generaciones=300;
%precision=input('Digite la precision');
precision=5;
%numero_poblacion_inicial=input('Digite la poblacion inicial');
numero_poblacion_inicial=100;
%pc==input('Ingrese la rata de cruce');
```

```
pc=0.5;
%pm==input('Ingrese la rata de mutación');
pm=0.01;

%tamano_cromosomas=0;
% for i=1:numero_variables
%     variables(i,1)=input('Digite el limite inferior');
%     variables(i,2)=input('Digite el limite superior');
%     variables(i,3)=floor(log2((variables(i,2)-variables(i,1))*10^precision)+1);
%     tamano_cromosomas=tamano_cromosomas+variables(i,3);
% end

variables(1,1)=-3;
variables(1,2)=12.1;

variables(2,1)=4.1;
variables(2,2)=5.8;
```

2) Cálculo de variables a usar

Se calculan los tamaños de las variables a usar

```
%El resultado del siguiente procedimiento es 21
variables(1,3)=floor(log2(( variables(1,2) - variables(1,1) )*10^precision)+1);

%El resultado del siguiente procedimiento es 18
variables(2,3)=floor(log2((variables(2,2)-variables(2,1))*10^precision)+1);
%El resultado del siguiente procedimiento es 39
tamano_cromosomas=variables(1,3)+variables(2,3);
```

3) Creación de población inicial

```
poblacion_inicial=round(rand(numero_poblacion_inicial,tamano_cromosomas));
```

4) Creación de padres

Se seleccionan las dos porciones para la generacion de los dos X

```
X1=poblacion_inicial(:,1:variables(1,3));
X2=poblacion_inicial(:,variables(1,3)+1:tamano_cromosomas);
```

5) Tranformación de variables

Se generan las palabras para cada fila: X transformado fila binaria a celda decimal

```
%Fila binaria a celda binaria
X1b=FilBin2CelBin(X1);
X2b=FilBin2CelBin(X2);
%Fila binaria a celda decimal
X1t=FilBin2CelDec(X1);
X2t=FilBin2CelDec(X2);
%X real
%Celda decimal a numero real
X1r=CelDec2NumRea(variables(1,1), variables(1,2), variables(1,3) , X1t);
X2r=CelDec2NumRea(variables(2,1), variables(2,2), variables(2,3) , X2t);
%Geracion de V binario
Vb=GenVb(X1b,X2b);
%Generacion de V real
V=GenVr(X1r,X2r);
```

6) Evaluación de la función objetivo

Se evalúa: 21.5+x1*sin(4*pi*x1) + sin(20*pi*x2), mediante: 21.5+V(i,1)*sin(4*pi*V(i,1)) + sin(20*pi*V(i,2));

```
E=funcionObjetivo(V);
```

7) Cálculo la aptitud total de la población

```
ATP=0;
for i=1:length(E)
    ATP=E(i) + ATP;
end
```

8) Cálculo las probabilidades de cada cromosoma

```
p=zeros(length(E),1);
for i=1:length(E)
    p(i,:)=E(i)/ATP;
end
```

9) Cálculo de las probabilidades acumuladas

```
qa=0;
q=zeros(length(p),1);

for i=1:length(p)
    qa=qa+p(i);
    q(i,:)=qa;
end
```

10) Primer procedimiento de seleccion

- Se genera r numeros aleatorios

```
r=rand(numero_poblacion_inicial,1);
```

- Seleccion de cromosomas

```
Vn=zeros(length(q),1);
for i=1:length(q)
    if r(i)<=q(i)
        Vn(i,:)=V(1);
    else
        Vn(i,:)=V(i);
    end
end
```

11) Procedimiento de cruce

- Se seleccionan los padres.

```
k=1;
padre_temp=Vb;

for i=1:length(Vn)
    %Es una de las formas que se encontró para generar números aleatrorios en el rango
    %r = min(p) + (max(p)-min(p)).*rand;
    r=rand;
```

```
    if r<pc
        padre_temp(k,:)=Vb(i,:);
        k=k+1;
    end
end

padre=padre_temp(1:k-1,:);
```

- Se genera una matriz de números aleatorios enteros entre [2 , No de bits -1]

```
r=randi([2 tamano_cromosomas-1],numero_poblacion_inicial,1);

i=1;
Vh=zeros(numero_poblacion_inicial,tamano_cromosomas);
while i<numero_poblacion_inicial
    rl=r(i);
    [padre1,padre2]=traerPareja(padre);
    p1=padre1(1:rl);
    p2=padre2(rl+1:end);
    Vh(i,:)=[p1 p2];
    p3=padre1(rl+1:end);
    p4=padre2(1:rl);
    Vh(i+1,:)=[p3 p4];
    i=i+1;
end
```

12) Mutación

Se vuelve a generar una matriz de números aleatorios, que serviran para evaluar la probabilidad

```
r=rand(tamano_cromosomas*length(Vh),1);
contador=0;
for k=1:length(r)
    if r(k)<pm
        Vh=mutar(Vh,k);
    end
end
```

13) Segunda selección

La segunda selección se hará por torneo y se seleccionan los mejores

```
PobTot=[Vh;BinChar2BinVect(padre)];
p1=FilBin2NumRea(variables(1,1), variables(1,2), variables(1,3),PobTot(:,1:variables(1,3)));
p2=FilBin2NumRea(variables(2,1), variables(2,2), variables(2,3),PobTot(:,variables(1,3)+1:tamano_cromosomas));
V=GenVr(p1,p2);
FO=funcionObjetivo(V);
FOi=zeros(size(FO,1),2);
for i=1:size(FO,1)
    FOi(i,1)=FO(i);
    FOi(i,2)=i;
end
FOi=FOi';
[y1,I] = sort(FOi(1,:));
index=I(end-numero_poblacion_inicial:end);
for k=size(index,2)-1:-1:1
    Vh(k,:)=PobTot(index(k),:);
end

maximo=0;
j=0;
```

13) Repetición

Se repite el algoritmo tantas generaciones como las que se ingresaron

```
for i=1:numero_generaciones
    [Vh,E,ATP,V]=myAG(Vh,variables,tamano_cromosomas,numero_poblacion_inicial,pc,pm);
end
```

Resultado

```
max(E(:))
```

```
ans =

   34.1241
```

Conclusiones

1. El proceso para solucionar el problema fue un tanto rápido debido a que no era complejo.
2. En la medida en la que se agregan más padres, da una solución mas óptima, pero más lenta
3. Se puede variar las probabilidades para encontrar soluciones no tan precisas pero si más rápidas

---

Published with MA TLAB® R2016b