UNIVERSIDAD NACIONAL DE COLOMBIA

Rediseño del sistema productivo utilizando técnicas de distribución de planta

A. anexos scripts del algoritmo genético programado en el software Matlab.

Cesar Julio Collazos Valencia

Universidad Nacional de Colombia
Facultad, de Ingeniería y Arquitectura
Manizales, Colombia
2013

Contenido

Adjacencia.m

```
function suma=adjacencia(layout,indices)
[x,y]=centroide(layout,indices); % modificar para n
A=[1 3 3 4 5];
B=[3 8 11 14 8];
suma=0;
for i=1:numel(A)
suma=suma+(abs(x(B(i))-x(A(i)))+abs(y(B(i))-y(A(i)))); % DISTANCIAS
RECTANGULARES
end
```

Alggen.m

```
function
[solucion,fitnessglobal,costoglobal,distancia,layout]=alggen(S,TP,NG,AG,i
ndices,
recorrido,layout)

%% PARAMETROS DEL ALGORITMO

parametros
R=0.1; % PORCENTAJE DE REPLICACION DE LOS MEJORES CROMOSOMAS (TAMBIEN
REPRESENTA EL PORCENTAJE DE CROMOSOMAS DESCARTADOS)

X=0.2; % PORCENTAJE DE CROMOSOMAS EN LA POBLACION ACTUAL QUE SE TRANSMITE
INTACTA A LA PROXIMA GENERACION
%% GENERACION DE LA POBLACION INICIAL:
for i=1:TP
cromosoma=S(1,randperm(numel(S))); % PERMUTACION ALEATORIA DEL CROMOSOMA
INICIAL
poblacion(i,:)=cromosoma; % COMPLETAR LA POBLACION FILA A FILA

end

%% ALGORITMO GENETICO:

for g=1:NG
disp(g)

%% DECODIFICAR Y EVALUAR CADA CROMOSOMA:

for i=1:TP
layout=distribution(poblacion(i,:),layout,recorrido); % DECODIFICA EL
CROMOSOMA
costotransporte(i)=costo(layout,indices); % MAXIMIZAR EL INVERSO DEL
COSTO
distanciacritica(i)=adjacencia(layout,indices); % MAXIMIZAR EL INVERSO
DE LA CERCANIA DE LOS DEPARTAMENTOS CLAVE

end

A= costotransporte./sum(costotransporte);
B=distanciacritica./sum(distanciacritica); % NUEVO FITNESS
fitness=1./(A+B);

% fitness=costotransporte+cercania; % NUEVO FITNESS
solucion(g,:)=poblacion(I,:); % ALMACENA EL MEJOR CROMOSOMA DE CADA


[fitnessglobal(g) I]=max(fitness); % ALMACENA EL MEJOR VALOR DE FITNESS
DE CADA GENERACION
```

```
GENERACION
costoglobal(g,:)=costotransporte(I);
distancia(g,:)=distanciacritica(I);

%% REMOVER LOS PEORES CROMOSOMAS:
[fitness,I]=sort(fitness); %% ORDENAR EL VECTOR FITNESS PARA CAMBIAR LOS
VALORES DE

%% RETIRAR EL PEOR
fitness((1:ceil(TP*R)))=fitness((end-ceil(TP*R)+1:end)); %% RETIRAR EL
PEOR

[fitness,I]=sort(fitness); %% ORDENAR EL VECTOR FITNESS PARA CAMBIAR LOS
VALORES DE

LA POBLACION (los ultimos son los mejores)
poblacion(1:end,:)=poblacion(I,:); % ORDENAR LOS CROMOSOMAS DEL PEOR AL
MEJOR

PORCENTAJE DE LA POBLACION

poblacion((1:ceil(TP*R)),:)=poblacion((end-ceil(TP*R)+1:end),:);
PORCENTAJE DE LA POBLACION

%% LA POBLACION (los ultimos son los mejores)
poblacion(1:end,:)=poblacion(I,:); % ORDENAR LOS CROMOSOMAS DEL PEOR AL
MEJOR
%% LLENAR LA PISCINA DE APAREAMIENTO
matpool=seleccion(poblacion,fitness);
pool=poblacion([matpool],:);

Npoblacion=poblacion(end-floor(TP*X)+1:end,:); % NUEVA POBLACION
[rows,cols]=size(Npoblacion);

for i=1:(TP-rows)/2
[filas,columnas]=size(pool);
n=randi(filas);
P1=pool(n,:);

% pool(n,:)=[];
[filas,columnas]=size(pool);
n=randi(filas);
P2=pool(n,:);

% pool(n,:)=[];
%% CRUCE:
pcruce=randi(2); % Probabilidad de cruce
if pcruce==1
[C1,C2]=cruce(P1,P2,AG); % parametro de cruce

else
```

```
C1=P1;
C2=P2;
end

%% MUTACION:

pmutacion=rand;
if pmutacion>0.75
C1=C1(1,randperm(numel(C1)));
C2=C2(1,randperm(numel(C2)));

end

Npoblacion=[Npoblacion;C1;C2]; % CONSERVAR UN PORCENTAJE DE LOS
CROMOSOMAS ORIGINALES EN LA NUEVA POBLACION

end

poblacion=Npoblacion;

end

[Y,I] = max(fitnessglobal);
solucion=solucion(I,:);
```

centroide.m

```
function [x,y]=centroide(layout,indices)
for i=indices

[rows,cols,vals]=find(layout==i); % SIRVE PARA ENCONTRAR QUE GRIDS
PERTENECEN AL DEPARTAMENTO i
x(i)=sum(cols-0.5)*5/numel(cols);
y(i)=sum(120-(rows-0.5)*5)/numel(rows); %PROBLEMA CON length(X)

end
```

costo.m

```
function suma=costo(layout,indices)


flujo=[0 0 1150 0 0 0 292 0 0 0 0 0 0 0
0 0 341 8894 3586 1800 1355 0 0 0 0 0 0 0
0 0 0 0 0 0 0 31346 0 0 0 0 0 0
0 0 0 0 0 0 0 10123 0 0 0 0 0 0
0 0 0 0 0 0 0 8738 1824 0 0 0 0 0
0 0 0 0 0 0 0 0 1800 0 0 0 0 0
0 0 0 1460 0 0 0 857 0 0 0 0 0 0
00000000000000
0 0 0 4785 9023 0 270 0 0 0 0 0 0 0
0 0 788 895 441 0 569 0 0 0 0 0 0 0
0 0 2753676 0 0 0 0 0 0 0 0 0 0 0
0 0 31346 9724 8738 1800 857 0 0 0 0 0 0 0
0 0 0 0 4369 0 0 0 0 0 0 0 0 0
0 0 0 0 0 11077 0 0 0 0 0 0 0 0];


[x,y]=centroide(layout,indices); % modificar para n

suma=0;
for i=indices
for j=indices
if i~=j

termino=(abs(x(i)-x(j))+abs(y(i)-y(j)))*flujo(i,j); % DISTANCIAS
RECTANGULARES

suma=suma+termino;

end
end
end
```

distribution.m

```
function layout=distribution(S,layoutini,recorrido)
parametros
layout=layoutini;
if recorrido==1

recorrido=recorrido1;
else if recorrido==2
recorrido=recorrido2;

end
end

%% DETERMINACION DEL NUMERO DE GRIDS A CADA DEPARTAMENTO:
A=ones(1,8); % 200 M^2
B=3*ones(1,8); % 200 M^2
C=4*ones(1,23); % 575 M^2 23
D=5*ones(1,12); % 300 M^2
E=6*ones(1,12); % 300 M^2
F=7*ones(1,6); % 150 M^2
G=10*ones(1,3); % 75 M^2 3
H=11*ones(1,5); % 125 M^2 5
I=12*ones(1,9); % 225 M^2 9
J=13*ones(1,5); % 125 M^2 5
K=14*ones(1,8); % 200 M^2 EN TOTAL SON 98 CUADROS
Z=0; %

vector=[];
for c=1:numel(S)
vector=[vector eval(S(c))]; % ESTA LINEA DECODIFICA EL STRING

end

%% ESTA LINEA REALIZA EL LAYOUT DE ACUERDO A LA CURVA DE recorrido:

c=0;
for i=recorrido(1:numel(vector))
c=c+1;
layout(i)=vector(c);

end
```

grafico.m

```
function [] = grafico(layout,indices,i)


F=figure(i); hold on
axis([0 200 0 120])
set(gca,'XTick',0:5:200)
set(gca,'YTick',0:5:120)
grid on,hold on
set(F,'Position',[1 41 1366 658])

[rows,cols,vals] =find(layout~=-1); % CAMBIAR POR QUE LOS DEPARTAMENTOS
DUMMY TIENEN
VALOR CERO INCLUIR UNA CONVENCION DEPARTAMENTOS FIJOS
% color=rand(14,3);

color =[0.9693 0.5513 0.9380
0.0923 0.9407 0.7158
0.0271 0.4581 0.1630
0.0484 0.8758 0.6515
0.6729 0.3461 0.3426
0.8271 0.2447 0.2930
0.3251 0.0823 0.6779
0.4677 0.7644 0.4250
0.0975 0.3545 0.8810
0.1548 0.2516 0.4487
0.7163 0.8268 0.7427
0.5654 0.3058 0.0100
0.7545 0.2779 0.9178
0.8847 0.7579 0.0297];

for i=1:length(vals)
if layout(i)~=0 %PARA QUE NO GRAFIQUE GRIDS VACIOS
fill([cols(i)-1 cols(i) cols(i) cols(i)-1 cols(i)-1]*5,120-[rows(i)-1
rows(i)-1
rows(i) rows(i) rows(i)-1]*5,color(layout(i),:));
text((cols(i)-0.7)*5,120-(rows(i)-
0.5)*5,num2str(layout(i)),'FontWeight','Bold');

end
end


%% grafico del centroide:

[x,y]=centroide(layout,indices) % MODIFICAR PARA GENERALIZAR n=6
p=plot(x,y,'ok');
set(p,'MarkerFaceColor','k','MarkerSize',5)
```

A. Anexos scripts del algoritmo genético programado en el software Matlab.

```
%% GRAFICAR EL TRAZADO DE CURVA CONTINUA:
% recorrido=[272 296 320 321 297 273 274 298 322 323 299 275 276 300 324
325 301 277 253
...
% 252 228 229 205 204 203 227 251 250 226 202 201 225 249 248 224 200 176
177
...
% 153 152 128:130 154 178 179 155 131 132 156 180 181 157 133 134 158 182
206
...
% 207 183 159 135 136 160 184 208 209 185 161 137 138 162 186 210 211 187
163
...
% 139 140 164 188 212];

% [fila,columna]=trazado(recorrido);

% line(columna,fila,'LineWidth',2)
```

layoutdesign.m

```
clc, clear all, close all
tic

%% LAYOUT INICIAL
layout=zeros(24,40);
layout([121:127 145:151 169:175 193:199 217:223 241:247 265:271 289:295
313:319 337:343
%
%
361:367])=8; % departamentos fijos
layout([658:660 682:684 706:708 730:732 754:756 778:780 802:804 826:828
850:852])=2;

departamentos fijos
layout([661:663 685:687 709:711 733:735 757:759 781:783 805:807 829:831
853:855])=9;

departamentos fijos
%% departamentos que se ubicaran con el segundo algoritmo genetico
layout([41:43 65:67])=11;
layout([529:531 553:555 577:579])=12;
layout([39 40 63 64])=13;

%% 1 ALGORITMO GENETICO
S='ABCDEFGK'
AG=1;
TP1=10; % TAMAÑO DE LA POBLACION
NG1=150; % NUMERO DE GENERACIONES
recorrido=1;
indices=[1:14];

[solucion,fitnessglobal,costoglobal,distancia,layout]=alggen(S,TP1,NG1,AG
,indices,
recorrido,layout);
progreso1=costoglobal;
layoutAG1=layout;

%% GRAFICO DEL PROGRESO DE LOS AG
figure(1)
subplot(1,2,1)
line(1:NG1,costoglobal)
title('ALGORITMO 1','fontsize',11)
xlabel('# DE GENERACIONES','fontsize',11)
ylabel('COSTO DE TRANSPORTE','fontsize',11)
subplot(1,2,2)
line(1:NG1,distancia)
title('ALGORITMO 1','fontsize',11)
xlabel('# DE GENERACIONES','fontsize',11)
ylabel('DISTANCIA CRITICA','fontsize',11)
```

```
%% GRAFICOS DE LAYOUT

figure(2)
grafico(layoutAG1,indices,2)
title(['ALGORITMO 1: ' num2str(fix(min(costoglobal)))],'fontsize',20)

%% 2 ALGORITMO GENETICO
% resetear las posiciones de los departamentos 11, 12 y 13

layout(find(layout==11))=0;
layout(find(layout==12))=0;
layout(find(layout==13))=0;

S='HIJ'
ceros=char(ones(1,67)*'Z'); % ESPACIOS VACIOS
S=cat(2,S,ceros);
AG=2;
TP2=10; % TAMAÑO DE LA POBLACION
NG2=100; % NUMERO DE GENERACIONES
recorrido=2;
indices=1:14;

[solucion,fitnessglobal,costoglobal,distancia,layout]=alggen(S,TP2,NG2,AG
,indices,recorrido,layout);
progreso2=costoglobal;
layoutAG2=layout;

%% GRAFICO DEL PROGRESO DE LOS AG
figure(3)
subplot(1,2,1)
line(1:NG2,costoglobal)
title('ALGORITMO 2','fontsize',11)
xlabel('# DE GENERACIONES','fontsize',11)
ylabel('COSTO DE TRANSPORTE','fontsize',11)
subplot(1,2,2)
line(1:NG2,distancia)
title('ALGORITMO 2','fontsize',11)
xlabel('# DE GENERACIONES','fontsize',11)
ylabel('DISTANCIA CRITICA','fontsize',11)


%% GRAFICOS DE LAYOUT
figure(4)
grafico(layoutAG2,indices,4)
title(['ALGORITMO 2: ' num2str(fix(min(costoglobal)))],'fontsize',20)
toc
% variacioncosto=fix([1:numel(fitnessglobal);(1./fitnessglobal)]');
% report layoutdesign % Genera un reporte que se puede ver en un
navegador
% de Internet, incluyendo las tablas de resultados y el grafico.
```

Layoutoriginal.m

```
figure
layout=zeros(24,40);
layout([401:404 425:428])=1;
layout([658:660 682:684 706:708 730:732 754:756 778:780 802:804 826:828
850:852])=2;
layout([182:185 206:209])=3;
layout([176:178 200:202 224:226 248:250 272:274 296:298 320:322])=4;
layout([179 180 203 204 227 228 251 252 275 299 323])=5;
layout([276 300 324 325 301 277 253 229 205 181])=6;
layout([186:188 210:212])=7;

layout([121:127 145:151 169:175 193:199 217:223 241:247 265:271 289:295
313:319 337:343

361:367])=8;
layout([661:663 685:687 709:711 733:735 757:759 781:783 805:807 829:831
853:855])=9;
layout([382 406])=10;
layout([41:43 65:67])=11;
layout([529:531 553:555 577:579])=12;
layout([39 40 63 64])=13;
layout([395:398 419:422])=14;

costototal=costo(layout,1:14)
grafico(layout,1:14,1)
title(['LAYOUT ORIGINAL = ',num2str(fix(costototal))],'fontsize',20)
```

parametros.m

```
%% CURVA DE recorrido:

recorrido1=[272 296 320 321 297 273 274 298 322 323 299 275 276 300 324
325 301 277 253
...

252 228 229 205 204 203 227 251 250 226 202 201 225 249 248 224 200 176
177
...

153 152 128:130 154 178 179 155 131 132 156 180 181 157 133 134 158 182
206
...

207 183 159 135 136 160 184 208 209 185 161 137 138 162 186 210 211 187
163
...

139 140 164 188 212];

recorrido2=[80 56 57 81 82 58 59 83 84 60 61 85 86
62 ...
63 87 88 64 65 89 90 66 67 91 92 68 69 93 ...
94 70 71 95 119 118 142 143 167 166 190 191 215 214 ...
238 239 263 262 286 287 311 310 334 335 359 358 382 383 ...
407 406 405 381 380 404 403 379 378 402 401 377 376 400 ...
399 375 374 398 397 373 372 396 395 371 370 394 393 369 ...
368 392];
```

probarrepeticion.m

```
function [rep]=probarrepeticion(S)
longitud=numel(S);
for i=1:longitud

P=find(S==S(i));
if numel(P)>1
rep=1; % SI HAY REPETICION

break
else

rep=0; % NO HAY REPETICION

end
end
```

seleccion.m

```
function [matpool]=seleccion(poblacion,fitness)

parsel=fitness./sum(fitness); % PARAMETRO DE SELECCION
[filas,columnas] = size(poblacion);
expnumb=parsel*filas; % NUMERO ESPERADO DE CADA CROMOSOMA EN LA PISCINA
DE APAREAMIENTO

Psel=mod(expnumb,1)./sum(mod(expnumb,1)); % PROBABILIDAD DE SELECCION DE
CADA CROMOSOMA

RULETA

matpool=[];

%% DUPLICAR LOS CROMOSOMAS DE ACUERDO A LA PARTE ENTERA DEL NUMERO
ESPERADO
%% DE CROMOSOMAS

NC=fix(expnumb);
for i=1:numel(expnumb)
if NC(i)~=0
for p=1:NC(i)
matpool=[matpool i];

end
end
end

%% RULETA (REVISAR SI ASEGURA LA REPLICACION DE LOS MEJORES CROMOSOMAS)
for i=1:filas-numel(matpool)
prob=0;
N=rand;
c=0;
while prob<N

c=c+1;
prob=prob+Psel(c);

end

matpool=[matpool c];

end
```

trazado.m

```
function [fila,columna]=trazado(indice)

c=0;

for i=indice

c=c+1;

fila(c)=rem(i,24);

columna(c)=ceil(i/24);

end

fila=120-(fila-0.5)*5; % invertir orden de filas


columna=(columna-0.5)*5;
```