*дата :*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **НУЛП, ІКНІ, САП** | | Тема | оцінка | підпис |
| КНC-13 | 3 (номер лаб.) | Комбінаторна оптимізація за допомогою еволюційних методів |  |  |
| Кохановський Н.Б. | |
| № залікової: 1608401 | |
| Методи нечіткої логіки та еволюційні алгоритми при автоматизованому проектуванні | | Викладач: | |
| Кривий Р. З. | |

**Мета:** ознайомитися з основними теоретичними відомостями, вивчити еволюційні оператори схрещування та мутації, що використовуються при розв’язуванні задач комбінаторної оптимізації. Розробити за допомогою пакету Matlab програмне забезпечення для вирішення задачі комівояжера.

**Теоретичні відомості**

* *Впорядковуючий оператор схрещування*

Впорядковуючий оператор схрещування (Order crossover, OX) був запропонований Д. Девісом у 1985 році для негомологічних числових хромосом. Схрещування може виконуватися по одній або по двох точках. Точки схрещування вибираються випадково. При одноточковому схрещуванні в хромосому першого нащадка копіюється хромосома першого батька, а потім гени нащадка, що розташовані правіше точки схрещування, перевпорядковуються у послідовність, що відповідає другому батькові. При цьому другий батько переглядається від початку до кінця, зліва направо, і елементи, яких не вистачає у нащадку, додаються, починаючи від точки схрещування, один по одному.

* Крок 1. Випадковим чином вибрати точку схрещування.
* Крок 2. Скопіювати в хромосому першого нащадка сегмент хромосоми першого батька, що розташований лівіше точки схрещування.
* Крок 3. Інші гени в нащадку копіюються із другого батька в упорядкованому вигляді зліва направо, крім елементів, які вже увійшли до нащадка.

Для створення другого нащадка застосовується аналогічний порядок дій.

* *Мутація обміну*

Мутація обміну використовується для бінарних і числових негомологічних хромосом. При класичній мутації обміну в хромосомі випадковим чином вибираються два гени, які міняються місцями.

* Крок 1. Створити хромосому нащадка як копію батьківської хромосоми

H = {h1,h2,...,hL}.

* Крок 2. Вибрати два числа y1 і y2 випадковим чином із множини Y = {0, 1, 2, .... L+1}, причому y1≠y2.
* Крок 3. Сформувати нову хромосому H шляхом обміну елементів, розташованих на позиціях y1 і y2.

Таким чином, після застосування класичної мутації обміну одержуємо

хромосому H'={h1­, h2, …,hy1-1, hy2, hy1+1, …, hy2-1, hy1, hy2+1, …, hL}.

**Індивідуальне завдання**

Розробити за допомогою пакету Matlab програмне забезпечення для вирішення задачі комівояжера. Параметри еволюційного методу обрати з таблиці 1 відповідно до варіанту.

Таблиця 1. Параметри еволюційного пошуку для виконання завдання

|  |  |  |  |
| --- | --- | --- | --- |
| **№ Варіанту** | **Еволюційні оператори** | | |
| **Схрещування** | **Мутація** |
| **11** | **Одноточечне впорядковуюче** | **Класичка мутація обміну** |

**Код програми**

nn=40; % кількість міст

asz=10; % розмір області asx x asz

ps=300; % чисельність популяції

ng=500; % Кількість поколінь

pm=0.01; % імовірність мутації обміну 2 випадкових міст на шляху (per gene, per genration)

pm2=0.02; % імовірність мутації обміну 2 частин шляху (per gene, per genration)

pmf=0.08; % імовірність мутації обміну випадкової частини шляху

r=asz\*rand(2,nn); % випадкове розподілення міст

dsm=zeros(nn,nn); % матриця відстаней

for n1=1:nn-1

r1=r(:,n1);

for n2=n1+1:nn

r2=r(:,n2);

dr=r1-r2;

dr2=dr'\*dr;

drl=sqrt(dr2);

dsm(n1,n2)=drl;

dsm(n2,n1)=drl;

end

end

% Почати з випадкових замкнених шляхів:

G=zeros(ps,nn);

for psc=1:ps

G(psc,:)=randperm(nn);

end

figure('units','normalized','position',[0.05 0.2 0.9 0.6]);

subplot(1,2,1);

% вивід найкращого шляху:

hpb=plot(NaN,NaN,'r-');

ht=title(' ');

hold on;

% вивід ділянок вузла

for n=1:nn

text(r(1,n),r(2,n),num2str(n),'color',[0.7 0.7 0.7]);

end

plot(r(1,:),r(2,:),'k.'); % виводити міста як чорні точки

axis equal;

xlim([-0.1\*asz 1.1\*asz]);

ylim([-0.1\*asz 1.1\*asz]);

subplot(1,2,2);

hi=imagesc(G);

title('color is city number');

colorbar;

xlabel('index in sequence of cities');

ylabel('path number');

pthd=zeros(ps,1); %довжина шляху

p=zeros(ps,1); % імовірності

for gc=1:ng % цикл поколінь

% знайти довжину шляху:

for psc=1:ps

Gt=G(psc,:);

pt=0; % підсумовування довжини шляху

for nc=1:nn-1

pt=pt+dsm(Gt(nc),Gt(nc+1));

end

% останній і перший:

pt=pt+dsm(Gt(nn),Gt(1));

pthd(psc)=pt;

end

ipthd=1./pthd; % зворотня довжина шляху

p=ipthd/sum(ipthd); % імовірності

[mbp bp]=max(p);

Gb=G(bp,:); % найкращий шлях

% оновити найкращий шлях на фігурі:

if mod(gc,5)==0

set(hpb,'Xdata',[r(1,Gb) r(1,Gb(1))],'YData',[r(2,Gb) r(2,Gb(1))]);

set(ht,'string',['generation: ' num2str(gc) ' best path length: ' num2str(pthd(bp))]);

set(hi,'CData',G);

drawnow;

end

% схрещування:

ii=roulette\_wheel\_indexes(ps,p); % гени з номерами міст ii будуть використані в схрещуванні

Gc=G(ii,:); % гени для схрещування

Gch=zeros(ps,nn); % діти

for prc=1:(ps/2) % підрахунок пар

i1=1+2\*(prc-1);

i2=2+2\*(prc-1);

g1=Gc(i1,:); %перший ген

g2=Gc(i2,:); %інший ген

cp=ceil((nn-1)\*rand); % точка схрещування, випадкове число в інтервалі [1; nn-1]

% двоє дітей:

g1ch=insert\_begining(g1,g2,cp);

g2ch=insert\_begining(g2,g1,cp);

Gch(i1,:)=g1ch;

Gch(i2,:)=g2ch;

end

G=Gch; % нові діти

% мутація обміну двох випадкових міст:

for psc=1:ps

if rand<pm

rnp=ceil(nn\*rand); % випадкове число міст для перестановки

rpnn=randperm(nn);

ctp=rpnn(1:rnp); %вибрати rnp випадкових міст для перестановки

Gt=G(psc,ctp); % отримати міста зі списку

Gt=Gt(randperm(rnp)); % переставити міста

G(psc,ctp)=Gt; % % повернути міста назад

end

end

% мутація обміну 2 частин шляху:

for psc=1:ps

if rand<pm2

cp=1+ceil((nn-3)\*rand); % діапазон [2 nn-2]

G(psc,:)=[G(psc,cp+1:nn) G(psc,1:cp)];

end

end

% мутація обміну випадкової частини шляху:

for psc=1:ps

if rand<pmf

n1=ceil(nn\*rand);

n2=ceil(nn\*rand);

G(psc,n1:n2)=fliplr(G(psc,n1:n2));

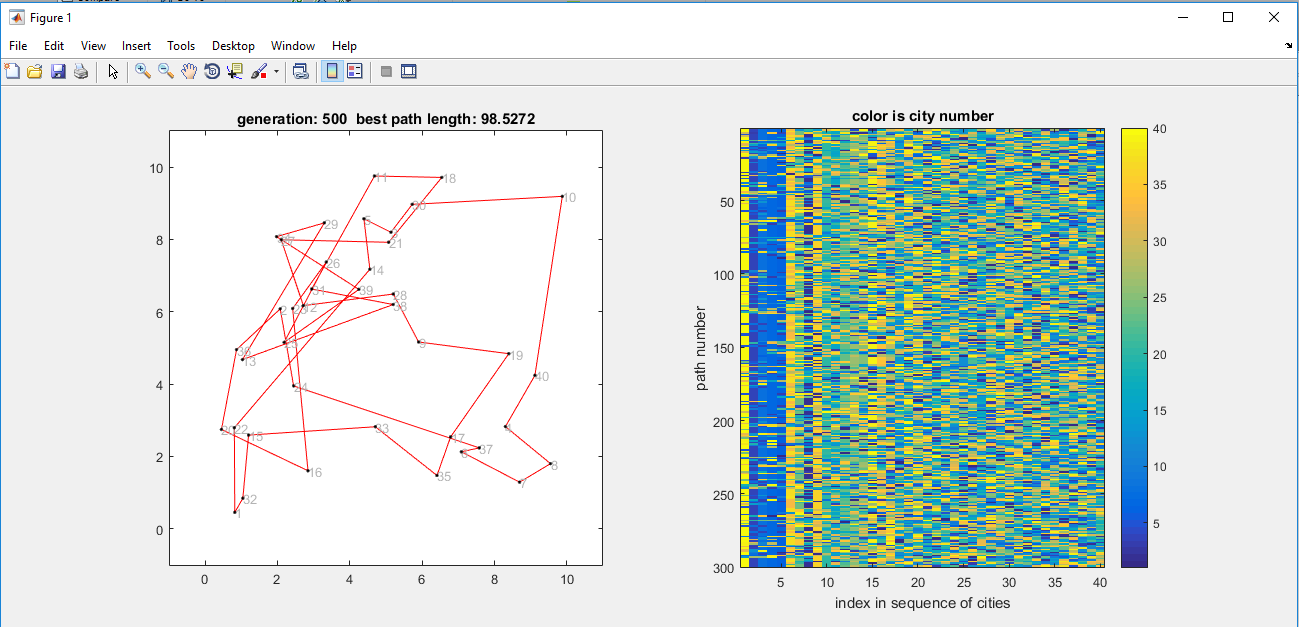
end

end

G(1,:)=Gb; % елітарність

end

**Результат виконання**

****

**Рис.3 Результат виконання програми**

**Висновок**

Виконуючи лабораторну роботу я ознайомився з основними еволюційними операторами схрещування та мутації, що використовуються при розв’язуванні задач комбінаторної оптимізації. Розробив за допомогою пакету Matlab програмне забезпечення для вирішення задачі комівояжера.