Ćwiczenie AG

Analiza działania algorytmu genetycznego

Część teoretyczna

Wykład 15: Algorytmy genetyczne.

Zadania do wykonania

Dokonaj analizy działania klasycznego algorytm genetycznego (AG) w zadaniu optymalizacji ciągłej funkcji $f(x) = 2x^2 + 1$. Optymalną wartość zmiennej x wyznacz w przedziale [-2·nr_gr, 3·nr_gr] z dokładnością q = 5 miejsc po przecinku.

1. Implementacja algorytmu genetycznego.

1.1. Ustawienia początkowe:

Uwaga - polecenia zamieszczone w punktach 1.1 - 1.4 umieścić w jednym skrypcie.

```
nr_gr = ?;  %tu wprowadź nr swojej podgrupy
x_p=-2*nr_gr;  %ograniczenie dolne zmiennej
x_k=3*nr_gr;  %ograniczenie górne zmiennej
q=5;  %dokładność (liczba cyfr po przecinku)
N=20;  %liczność populacji
p_c=0.9;  %prawdopodobieństwo krzyżowania
p_m=0.01;  %prawdopodobieństwo mutacji
L_gen=100;  %maksymalna liczba generacji

%deklaracje macierzy
x_opt=[0,0,0];
X=zeros(N,L_gen);
F_sr=zeros(1,L_gen);
F_min=zeros(1,L_gen);
F_max=zeros(1,L_gen);

t_start=cputime;    %pomiar czasu
```

1.2. Obliczenie długości chromosomu i utworzenie początkowej populacji chromosomów:

```
%określenie długości chromosomu
l_odc=(x_k-x_p)*10^q;
m=ceil(log2(l_odc+1));  %minimalna liczba bitów do zakodowania zmiennej z
dokładnością q

%losowanie populacji początkowej
ch=round(rand(N,m));
```

1.3. Pętla główna AG:

```
for i=1:L_gen

    %dekodowanie populacji
    x=dekoduj(ch,N,m,x_p,x_k);
    X(:,i)=x;

    %funkcja przystosowania populacji
    F=F_celu1(x);
    F_sr(i)=mean(F);        %średnia wartość f. przystosowania
    F_min(i)=min(F);        %minimalna wartość f. przystosowania
    F_max(i)=max(F);        %maksymalna wartość f. przystosowania
```

```
        %zapamietanie najlepszego osobnika
        if F_max(i)>x_opt(2)
            ind=find(F==F_max(i));
            x_opt=[x(ind(1)),F_max(i),i];
            ch_opt=ch(ind(1),:);
        end

        %prawdopodobieństwa selekcji
        p_s=F/(F_sr(i)*N);

        %reguła ruletki
        nr_ch=ruletka(p_s,N);

        %pula rodzicielska
        ch=ch(nr_ch,:);

        %krzyżowanie
        ch=krzyzuj(ch,N,m,p_c);

        %mutacja
        ch=mutuj(ch,N,m,p_m);

    end;
```

1.4. Wizualizacja wyników:

```
%rezultaty
fprintf('Czas działania algorytmu [s]: %5.2f\n',cputime-t_start);

fprintf('Najlepsze rozwiązanie x_opt = %5.2f, f(x_opt) = %5.2f, znalezione w
generacji: %5.0f\n',x_opt(1),x_opt(2),x_opt(3));

fprintf('F_min, F_sr, F_max ostatniej populacji:  %5.2f   -   %5.2f   -
%5.2f \n',F_min(L_gen),F_sr(L_gen),F_max(L_gen));

fprintf('F_min, F_sr, F_max wszystkich populacji:  %5.2f   -   %5.2f   -
%5.2f \n',mean(F_min),mean(F_sr),mean(F_max));

fprintf('Odchylenie standardowe funkcji przystosowania ostatniej populacji:
%5.2f',std(F));

%wykresy
figure(1);
plot(1:L_gen,F_sr,1:L_gen,F_min,1:L_gen,F_max); xlabel('nr generacji');
ylabel('f(x)'); title('Przystosowanie w czasie');
figure(2);
xx=x_p:0.01:x_k;
y=F_celu1(xx);
plot(xx,y,x,zeros(N,1),'x',x_opt(1),0,'o'); xlabel('x'); ylabel('f(x)');
title('Funkcja celu i fenotypy w ostatniej generacji');
figure(3)
hist(X(:)); xlabel('x'); ylabel('Liczba chromosomów w poszczególnych
kategoriach'); title('Histogram');
```

1.5. Funkcja dekodująca chromosom:

```
function [x]=dekoduj(ch,N,m,x_p,x_k)

nr_odc=zeros(N,1);
for i=1:m
    nr_odc=nr_odc+ch(:,m-i+1)*2^(i-1);
end
x=x_p+nr_odc*(x_k-x_p)/(2^m-1);
```

1.6. Funkcja celu:

```
function [y]=F_celu1(x)

y=2*x.*x+1;
```

1.7. Funkcja koła ruletki:

```
function [nr_ch]=ruletka(p_s, N)

for i=1:N
    zakr_ch(i)=sum(p_s(1:i));
end
q=rand(N, 1);
for i=1:N
    index=find(zakr_ch>=q(i));  %zwraca indeksy wszystkich chromosomów >= od
wylosowanej liczby
    nr_ch(i)=index(1);
end
```

1.8. Funkcja krzyżowania:

```
function [ch_p]=krzyzuj(ch, N, m, p_c)

rodzic=ceil(rand(N/2,2)*N);
punkt_c=ceil(rand(N/2,1)*(m-1));
ch_p=ch*0-1;
j=1;
for i=1:N/2
 if rand>p_c %kopiowanie bez krzyżowania
   ch_p(j,:)=ch(rodzic(i,1),:);
   ch_p(j+1,:)=ch(rodzic(i,2),:);
 else %krzyżowanie
   ch_p(j,:)=[ch(rodzic(i,1),1:punkt_c(i)) ch(rodzic(i,2),punkt_c(i)+1:m)];
   ch_p(j+1,:)=[ch(rodzic(i,2),1:punkt_c(i)) ch(rodzic(i,1),punkt_c(i)+1:m)];
 end
 j=j+2;
end
```

1.9. Funkcja mutacji:

```
function [ch]=mutuj(ch,N,m,p_m)

index=find(rand(N,m)<p_m);
ch(index)=~ch(index);
```

2. Uruchom skrypt i zanotuj najlepszy chromosom `ch_opt`. Czy ten chromosom rzeczywiście reprezentuje optymalne rozwiązanie? Zdekoduj `ch_opt` za pomocą funkcji `x = dekoduj(ch_opt,1,m,x_p, x_k)`. Wyznacz jego przystosowanie za pomocą funkcji `F_celu1(x)`. Jak zmieni się fenotyp i przystosowanie, jeśli zmienimy pierwszy bit `ch_opt`? A jeśli zmienimy ostatni bit? Wytłumacz zaobserwowane zmiany. Zamieść w sprawozdaniu i zinterpretuj wykresy wygenerowane przez program. Objaśnij co one pokazują.

3. Zaobserwuj jak zmieniają się wyniki w zależności od prawdopodobieństwa mutacji. Uruchom skrypt dla różnych prawdopodobieństw mutacji `p_m` = 0, 0.01, 0.1 i 0.5. Jak zmieniają się wykresy? W sprawozdaniu zamieść i zinterpretuj wyniki wygenerowane w oknie poleceń i wykresy.

4. Ile bitów ulegnie zmianie w populacji po zastosowaniu operacji mutacji z ww. prawdopodobieństwami? Sprawdź to eksperymentalnie. W tym celu wygeneruj losowo populację chromosomów `ch`, przekaż ją do funkcji `mutuj()`. Otrzymasz w ten sposób populację zmutowaną `ch1`. Oblicz na ilu pozycjach (bitach) różnią się populacje `ch` i `ch1`.

5. Zaobserwuj jak zmieniają sie wyniki, gdy nie zastosujemy operacji krzyżowania (`p_c` = 0). Eksperymenty wykonaj przy `p_m` = 0.01.

Zawartość sprawozdania

Sprawozdania powinny być sporządzone według wzoru zamieszczonego na stronie i zawierać:

A) Cel ćwiczenia.
B) Treść zadania.
C) Opis algorytmu genetycznego (nie kopiuj treści wykładu, poszukaj w literaturze i Internecie).
D) Metodyka rozwiązania zadania.
E) Zestawienie wyników (wykresy, tabele z komentarzem).
F) Wnioski końcowe.
G) Wydruk programu.

Zadania dodatkowe dla ambitnych

Te zadania studenci wykonują indywidualnie, po uzgodnieniu z prowadzącym. Zadania nie są obligatoryjne. Z zadania sporządzamy sprawozdanie.

1. Za pomocą AG dokonaj optymalizacji funkcji dwu zmiennych x1 i x2.
2. Zaprogramuj algorytm użyty w ćwiczeniu w innym środowisku, np.: R, Python, Statistica, C#, ...

Przykładowe zagadnienia i pytania zaliczeniowe

1. Schemat AG.
2. Binarne kodowanie zmiennych.
3. Parametry AG.
4. Genotyp, fenotyp i przystosowanie osobnika.
5. Selekcja metodą koła ruletki.
6. Krzyżowanie jednopunktowe.
7. Mutacja.
8. Materiał ze sprawozdania.