Data Science
Metody Sztucznej Inteligencji

Paweł Wawrzyński

Przeszukiwanie
Algorytmy ewolucyjne

Dzisiaj

- Zagadnienie przeszukiwania (optymalizacji)
- Podejścia do przeszukiwania
- Idea algorytmów ewolucyjnych
- Algorytm (1+1)
  - mutacja
- Algorytm zrównoleglony (1+1)
- Algorytm (μ+λ)
  - krzyżowanie
- CMA-ES

Przeszukiwanie - optymalizacja

- W przestrzeni X
- istnieje funkcja $f : X \rightarrow \mathrm{R}$
  jest to funkcja celu / kryterium jakości
- Chodzi o znalezienie x maksymalizującego f
- Przykłady:
  – najlepsze parametry dla procesu technicznego lub ekonomicznego
  – najlepsza struktura procesu j.w.
  – najlepsza struktura sieci telekomunikacyjnej

Podejścia do optymalizacji

- Przegląd wszystkich możliwości
- Metody przeglądowe z eliminacją
  - np. alg. podziału i oszacowań
- Metody analityczne
  - np. alg. gradientów sprzężonych
- Metody losowe
  - alg. ewolucyjne, genetyczne
  - symulowane wyżarzanie

Idea algorytmów ewolucyjnych

- Osobnik
- Populacja
- Reprodukcja, potomstwo
  - mutacja
  - krzyżowanie
- f – funkcja przystosowania

Ogólny algorytm ewolucyjny

1. Wygeneruj populację początkową P
2. Reprodukuj z P zbiór osobników R
3. Wybierz z $P \cup R$ nową populację P
4. Jeśli nie jest spełniony warunek stopu, wróć do punktu 2, w przeciwnym razie zwróć najlepszego osobnika z P

Algorytm (1+1)

Parametry: *n* - wymiar *X*, naturalne *m* oraz początkowe $\sigma > 0$

1. Wygeneruj pierwszego osobnika $x \in X$
2. Wygeneruj potomka $y = x + \sigma N(0, I)$
3. Wybierz $x = f(x) > f(y)\, ?\, x : y$
4. Zaktualizuj φ jako proporcję wybranych *y*-ków w ciągu ostatnich *m* iteracji.
5. Co *m*-ty krok dokonaj przypisania:

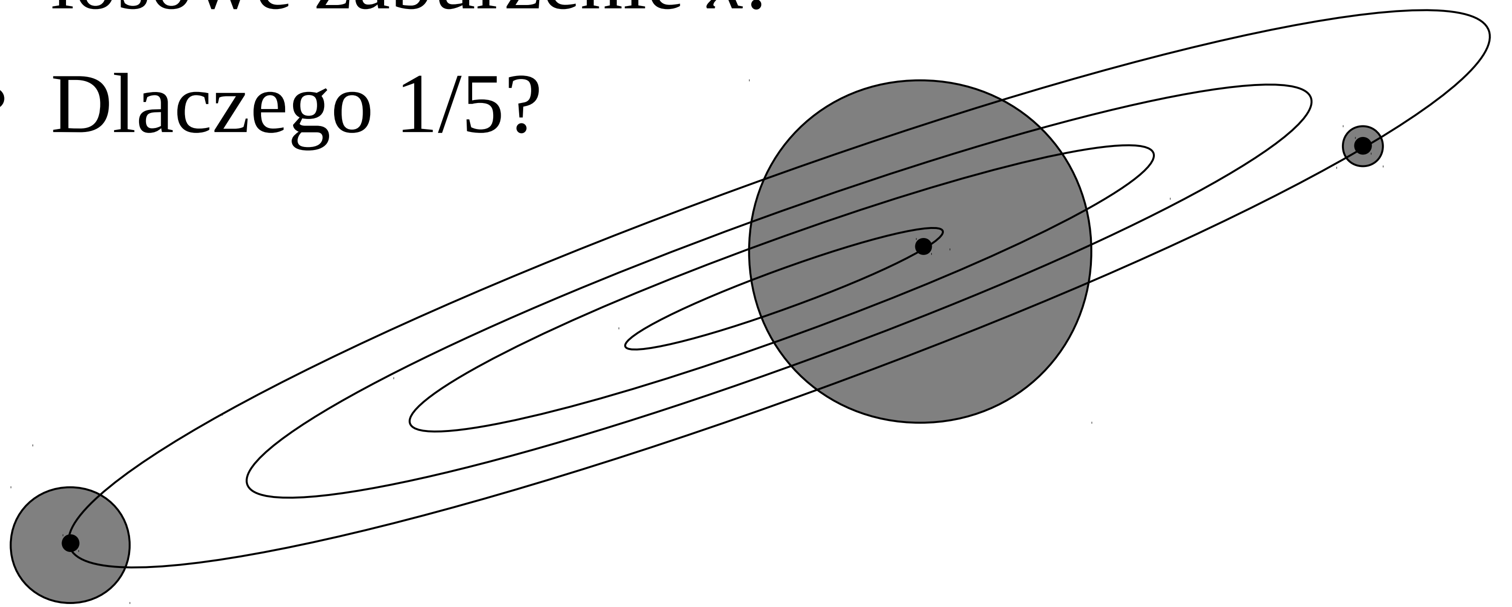
$$\sigma = \begin{cases} c_1\sigma & \text{dla } \varphi < 1/5 \\ c_2\sigma & \text{dla } \varphi > 1/5 \\ \sigma & \text{dla } \varphi = 1/5 \end{cases}$$

6. Jeśli $\sigma < \sigma_{\min}$, zakończ przyjmując *x* jako wynik; w przeciwnym razie wróć do Punktu 2.

- Algorytm działa nieźle dla $m = 10$, $c_1 = 0{,}82$ oraz $c_2 = 1{,}2$

Mutacja w algorytmie (1+1)

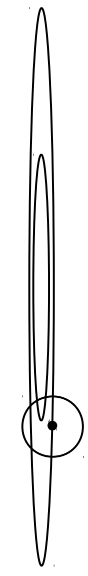
- Zmutowany osobnik *y* powstaje przez losowe zaburzenie *x*.
- Dlaczego 1/5?



Zrównoleglony (1+1)

1. Wybierz $x_r \in X$; będzie to wartość dostępna w sposób wyłączny dla wszystkich wątków algorytmu
2. W zbiorze niezależnych wątków realizuj tą samą procedurę:
   Powtarzaj
   a) x = wynik działania algorytmu (1+1)
   b) jeżeli $f(x) > f(x_r)$, przypisz $x_r = x$
   dopóki nie wystąpił warunek stopu
3. Zwróć x_r

Skalowanie argumentów funkcji przystosowania



- Algorytm: (1+1)
- Przykład, reakcja chemiczna
  - temperatura [20,80]
  - stężenie katalizatora [10^{-5},10^{-4}]
  - tempo optymalizacji temp. takie jak stężenia
- Remedium: skalowanie wymiarów, np.
  - przeszukiwanie przestrzeni: [20,80]x[10,100]
  - której drugi wymiar to stężenie x 10^6

Algorytm (μ+λ)

Parametry: n - wymiar X, naturalne μ oraz λ ($\mu < \lambda$)

1. Wygeneruj P - populację μ osobników postaci
$$\langle x_1,\ldots,x_n,\sigma_1,\ldots,\sigma_n \rangle$$
2. Wylosuj z P λ-elementową tymczasową populację T
3. Reprodukuj z T λ-elementową populację potomną R stosując krzyżowanie i mutację.
4. Utwórz P jako μ osobników wybranych z P $\cup$ R.
5. Jeśli jest spełniony warunek stopu, zwróć wynik jako najlepszy element P;
   w przeciwnym razie wróć do punktu 2.

Algorytm (μ+λ) - krok 2.

- Wielokrotne losowanie ze zwracaniem

Parametry: n - wymiar X, naturalne μ oraz λ ($\mu < \lambda$)

1. Wygeneruj P - populację μ osobników postaci
$$\langle x_1, \ldots, x_n, \sigma_1, \ldots, \sigma_n \rangle$$
2. Wylosuj z P λ-elementową tymczasową populację T
3. Reprodukuj z T λ-elementową populację potomną R stosując krzyżowanie i mutację.
4. Utwórz P jako μ osobników wybranych z P $\cup$ R.
5. Jeśli jest spełniony warunek stopu, zwróć wynik jako najlepszy element P;
w przeciwnym razie wróć do punktu 2.

Algorytm (μ+λ) - krzyżowanie

- Uśrednianie:

$$\langle \bar{x}, \bar{\sigma} \rangle = \langle (x^f + x^m)/2, (\sigma^f + \sigma^m)/2 \rangle$$

- Interpolacja

$$\langle \bar{x}, \bar{\sigma} \rangle = \langle a x^f + (1-a) x^m, a \sigma^f + (1-a) \sigma^m \rangle$$

dla

$$a \sim U(0,1)$$

Parametry: n - wymiar X, naturalne μ oraz λ ($\mu < \lambda$)

1. Wygeneruj P - populację μ osobników postaci
$$\langle x_1, \ldots, x_n, \sigma_1, \ldots, \sigma_n \rangle$$
2. Wylosuj z P λ-elementową tymczasową populację T
3. Reprodukuj z T λ-elementową populację potomną R stosując krzyżowanie i mutację.
4. Utwórz P jako μ osobników wybranych z P $\cup$ R.
5. Jeśli jest spełniony warunek stopu, zwróć wynik jako najlepszy element P;
   w przeciwnym razie wróć do punktu 2.

Algorytm (μ+λ) - mutacja

- Natężenie rozmywania

$$\sigma_i^R = \bar{\sigma}_i \exp\left(\tau N(0,1) + \tau' N_i(0,1)\right)$$

$$\tau = \frac{1}{\sqrt{2d}}, \quad \tau' = \frac{1}{\sqrt{2\sqrt{d}}}, \quad d = \dim(x)$$

- Rozwiązanie

$$x_i^R = \bar{x}_i + \sigma_i^R N_i(0,1)$$

Parametry: n - wymiar X, naturalne μ oraz λ ($\mu < \lambda$)

1. Wygeneruj P - populację μ osobników postaci

$$\langle x_1, \dots, x_n, \sigma_1, \dots, \sigma_n \rangle$$

2. Wylosuj z P λ-elementową tymczasową populację T
3. Reprodukuj z T λ-elementową populację potomną R stosując krzyżowanie i mutację.
4. Utwórz P jako μ osobników wybranych z P $\cup$ R.
5. Jeśli jest spełniony warunek stopu, zwróć wynik jako najlepszy element P;
w przeciwnym razie wróć do punktu 2.

Algorytm (μ+λ) - wybór

- μ najlepszych.
- Metoda koła ruletki: osobniki są losowane z prawdopodobieństwem proporcjonalnym do

$$g\big(f(x)\big), \quad \text{np. } g(y) = e^{y}$$

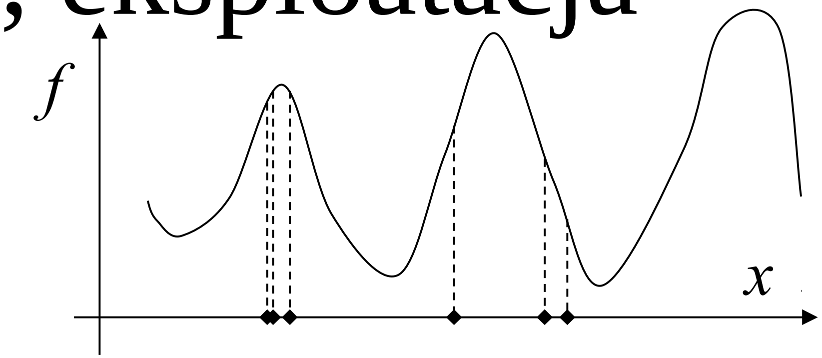
- Selekcja rankingowa p-stwo wyboru x proporcjonalne do

$$\mu + \lambda + 1 - pozycja(x)$$

Parametry: n - wymiar X, naturalne μ oraz λ ($\mu < \lambda$)

1. Wygeneruj P - populację μ osobników postaci
$$\langle x_1, \ldots, x_n, \sigma_1, \ldots, \sigma_n \rangle$$
2. Wylosuj z P λ-elementową tymczasową populację T
3. Reprodukuj z T λ-elementową populację potomną R stosując krzyżowanie i mutację.
4. Utwórz P jako μ osobników wybranych z P $\cup$ R.
5. Jeśli jest spełniony warunek stopu, zwróć wynik jako najlepszy element P;
   w przeciwnym razie wróć do punktu 2.

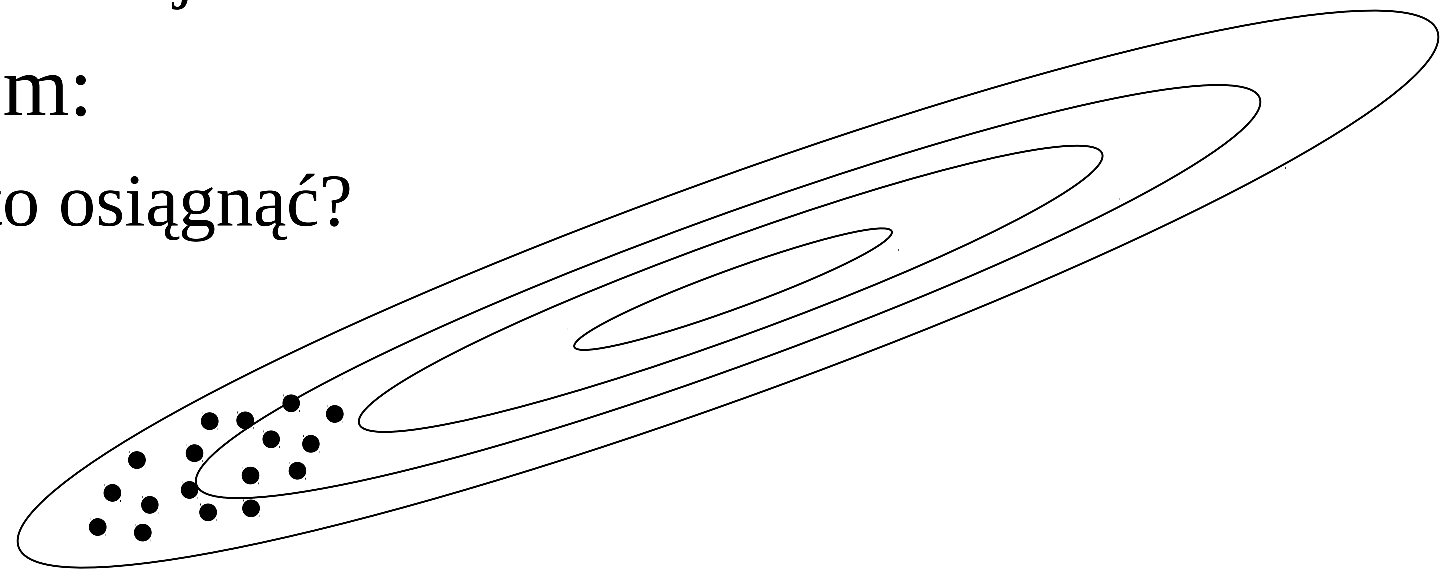
Presja selekcyjna, eksploracja, eksploatacja



- Algorytm: ($\mu + \lambda$)
- Populacja $P \cup R$:
- Presja selekcyjna: preferencja dla lepiej przystosowanych - słaba / silna
- Eksploracja = poszukiwanie max. globalnego
- Eksploatacja = poszukiwanie max. lokalnego

Nowoczesne algorytmy ewolucyjne

- Cel:
  – Nowi członkowie populacji są losowani z tego samego rozkładu o dostosowującej się macierzy kowariancji
- Problem:
  – Jak to osiągnąć?



Algorytm CMA-ES

Covariance Matrix Adaptation Evolution Strategy

ustal λ

zainicjuj $m, \sigma, C=I, p_\sigma=0, p_C=0$

while $nie\ \ koniec:$

  for $i=1\ldots\lambda:$

    $x_i \sim N(m, \sigma^2 C)$

  przesortuj $x_1 \ldots x_\lambda$ według $f(x_i)$ malejąco

  $m' \leftarrow m$

  $m \leftarrow \mathrm{update\_m}(x_i, \ldots, x_\lambda)$

  $p_\sigma \leftarrow \mathrm{update\_p}_\sigma(p_\sigma, \sigma^{-1} C^{-1/2}(m-m'))$

  $p_C \leftarrow \mathrm{update\_p}_\mathrm{C}(p_C, \sigma^{-1}(m-m'), \|p_\sigma\|)$

  $C \leftarrow \mathrm{update\_C}(C, p_c, (x_1-m')/\sigma, \ldots, (x_\lambda-m')/\sigma)$

  $\sigma \leftarrow \mathrm{update\_}\sigma(\sigma, \|p_\sigma\|)$

return m lub x_1

Algorytm CMA-ES

m − środek chmury punktów

update_m

$$m = \sum_{i=1}^{\lambda} w_i x_i$$
$$= m' + \sum_{i=1}^{\lambda} w_i (x_i - m')$$

$w_1 \geq w_2 \geq \ldots \geq w_\lambda \geq 0$

$\sum_{i=1}^{\lambda} w_i = 1$

$\sum_{i=1}^{\lambda} w_i^2 \approx 4/\lambda$

typowo $w_i = 0$ dla $i > \lambda/2$

ustal λ
zainicjuj $m, \sigma, C = I, p_\sigma = 0, p_C = 0$
while *nie koniec*:
  for $i = 1 \ldots \lambda$:
    $x_i \sim N(m, \sigma^2 C)$
  przesortuj $x_1 \ldots x_\lambda$ według $f(x_i)$ malejąco
  $m' = m$
  $m \leftarrow \text{update\_m}(x_i, \ldots, x_\lambda)$
  $p_\sigma \leftarrow \text{update\_p}_\sigma(p_\sigma, \sigma^{-1} C^{-1/2}(m - m'))$
  $p_C \leftarrow \text{update\_p}_C(p_C, \sigma^{-1}(m - m'), \|p_\sigma\|)$
  $C \leftarrow \text{update\_C}(C, p_c, (x_1 - m')/\sigma, \ldots, (x_\lambda - m')/\sigma)$
  $\sigma \leftarrow \text{update\_}\sigma(\sigma, \|p_\sigma\|)$
return m lub x_1

Algorytm CMA-ES

$\sigma -$skala chmury punktów
$p_\sigma -$ ścieżka zmian σ

update_p$_\sigma$
update_ σ

$$p_\sigma \leftarrow (1-\beta_\sigma) p_\sigma + \sqrt{1-(1-\beta_\sigma)^2}\sqrt{W} C^{-1/2}\frac{m-m'}{\sigma}$$

$$\sigma \leftarrow \sigma \exp\left(\frac{\beta_\sigma}{d_\sigma}\left(\frac{\|p_\sigma\|}{E\|N(0,I)\|}-1\right)\right)$$

$n = dim(x_i)$
$\beta_\sigma \approx 3/n$
$W = \left(\sum_{i=1}^{\lambda} w_i^2\right)^{-1}$
$d_\sigma \approx 1$

ustal λ
zainicjuj $m, \sigma, C=I, p_\sigma=0, p_C=0$
while $nie\ \ koniec$:
  for $i=1\ldots\lambda$:
    $x_i \sim N(m, \sigma^2 C)$
  przesortuj $x_1 \ldots x_\lambda$ według $f(x_i)$ malejąco
  $m' = m$
  $m \leftarrow$ update_m$(x_i, \ldots, x_\lambda)$
  $p_\sigma \leftarrow$ update_p$_\sigma$$(p_\sigma, \sigma^{-1} C^{-1/2}(m-m'))$
  $p_C \leftarrow$ update_p$_C$$(p_C, \sigma^{-1}(m-m'), \|p_\sigma\|)$
  $C \leftarrow$ update_C$(C, p_c, (x_1-m')/\sigma, \ldots, (x_\lambda-m')/\sigma)$
  $\sigma \leftarrow$ update_ $\sigma(\sigma, \|p_\sigma\|)$
return m lub x_1

Algorytm CMA-ES

$C -$ macierz kowariancji chmury punktów
$p_C -$ ścieżka zmian C

update_p$_{\mathbf{C}}$
update_C

ustal λ
inicjacja $m, \sigma, C = I, p_\sigma = 0, p_C = 0$
while *nie koniec*:
  for $i = 1 \ldots \lambda$:
    $x_i \sim N(m, \sigma^2 C)$
  przesortuj $x_1 \ldots x_\lambda$ według $f(x_i)$ malejąco
  $m' = m$
  $m \leftarrow$ update_m$(x_i, \ldots, x_\lambda)$
  $p_\sigma \leftarrow$ update_p$_\sigma(p_\sigma, \sigma^{-1} C^{-1/2}(m - m'))$
  $p_C \leftarrow$ update_p$_C(p_C, \sigma^{-1}(m - m'), \|p_\sigma\|)$
  $C \leftarrow$ update_C$(C, p_c, (x_1 - m')/\sigma, \ldots, (x_\lambda - m')/\sigma)$
  $\sigma \leftarrow$ update_ $\sigma(\sigma, \|p_\sigma\|)$
return m lub x_1

$$p_C \leftarrow (1-\beta_C) p_C + \mathbf{1}_{[0, 1.5\sqrt{n}]}(\|p_\sigma\|) \sqrt{1-(1-\beta_C)^2} \sqrt{W} \frac{m - m'}{\sigma}$$

$$C \leftarrow (1 - \beta_1 - \beta_W + \beta_S) C + \beta_1 p_C p_C^T + \beta_W \sum_{i=1}^{\lambda} w_i \frac{x_i - m}{\sigma} \left(\frac{x_i - m}{\sigma} \right)^T$$

$$n = dim(x_i)$$

$$\beta_C \approx 4/n$$

$$\mathbf{1}_{[0, 1.5\sqrt{n}]}(\|p_\sigma\|) = 1 \text{ jeśli } \|p_\sigma\| \in [0, 1.5\sqrt{n}] \quad (\text{inaczej} = 0)$$

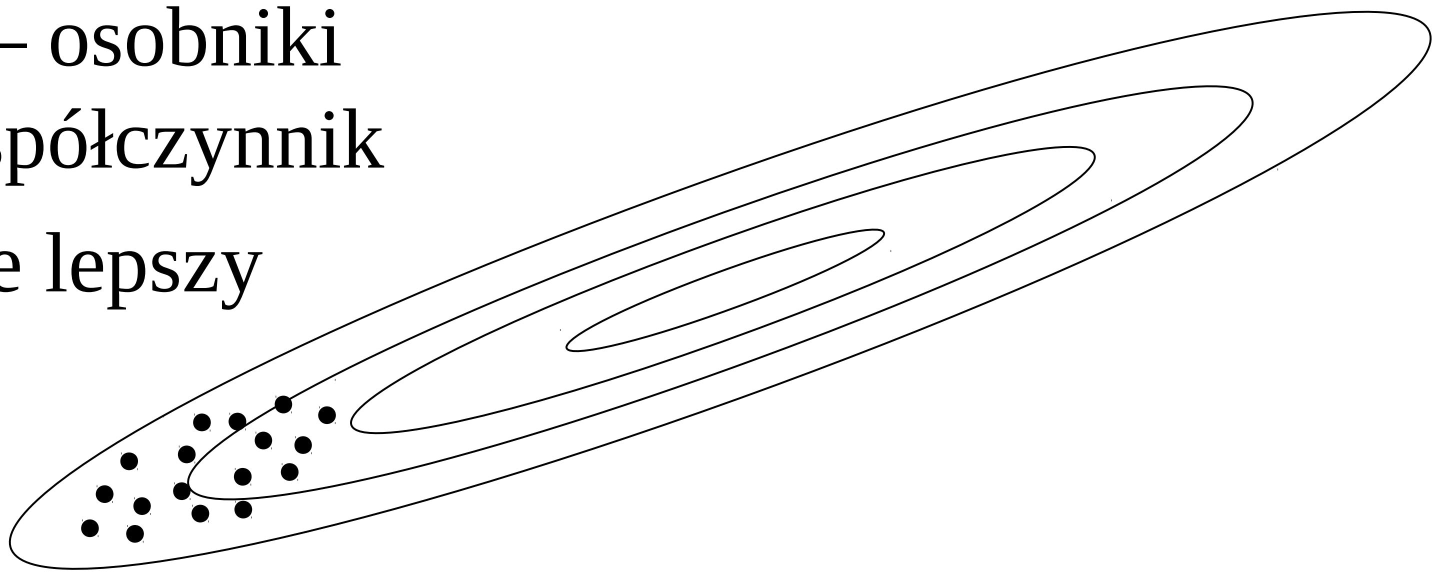
$$\beta_S = (1 - \mathbf{1}_{[0, 1.5\sqrt{n}]}(\|p_\sigma\|)) \beta_1 \beta_C (2 - \beta_C)$$

$$\beta_1 \approx 2/n^2$$

$$\beta_M \approx W/n^2, \quad W = \left(\sum_{i=1}^{\lambda} w_i^2 \right)^{-1}$$

Ewolucja różnicowa

- μ osobników w populacji
- Nowe osobniki tworzone formułami typu
  nowy = $x + F\,(y - z)$
  x, y, z – osobniki
  F – współczynnik
- Zostaje lepszy
  z pary:
  stary,
  nowy



Ewolucja różnicowa – algorytm

ustal λ
wylosuj λ osobników z przeszukiwanej przestrzeni
dopóki *nie koniec*:
  for $i=1\ldots\lambda$:
    wylosuj kilka osobników do reprodukcji
    reprodukuj x'_i z tych osobników
    $x'_i \leftarrow \text{krzyżowanie}(x'_i, x_i)$
    jeśli $f(x'_i) > f(x_i)$:
      W następnym pokoleniu x'_i zastępuje x_i
zwróć x_{best}

DE – reprodukcja

$j1, j2, \ldots$ są losowane z $\{1, \ldots, \lambda\}$

Różne strategie:

DE/rand/1$: x'_i = x_{j1} + F \cdot (x_{j2} - x_{j3})$
DE/rand/2$: x'_i = x_{j1} + F \cdot (x_{j2} - x_{j3} + x_{j4} - x_{j5})$
DE/best/1$: x'_i = x_{best} + F \cdot (x_{j1} - x_{j2})$
DE/best/2$: x'_i = x_{best} + F \cdot (x_{j1} - x_{j2} + x_{j3} - x_{j4})$
DE/rand-to-best/2$: x'_i = x_i + F \cdot (x_{best} - x_i + x_{j1} - x_{j1})$

DE – krzyżowanie

Wariant dwumianowy (bin):
Dla każdego $k=1..dim(x)$:
 jeśli liczba losowa z [0,1] $<C_r$:
  $x'[k]\leftarrow x[k]$
 w przeciwnym razie:
  $x'[k]\leftarrow x'[k]$

Wariant wykładniczy (exp):
Dla kolejnego $k=1..dim(x)$:
 jeśli liczba losowa z [0,1] $<C_r$:
  $x'[k]\leftarrow x[k]$
 w przeciwnym razie dla pozostałych k:
  $x'[k]\leftarrow x'[k]$

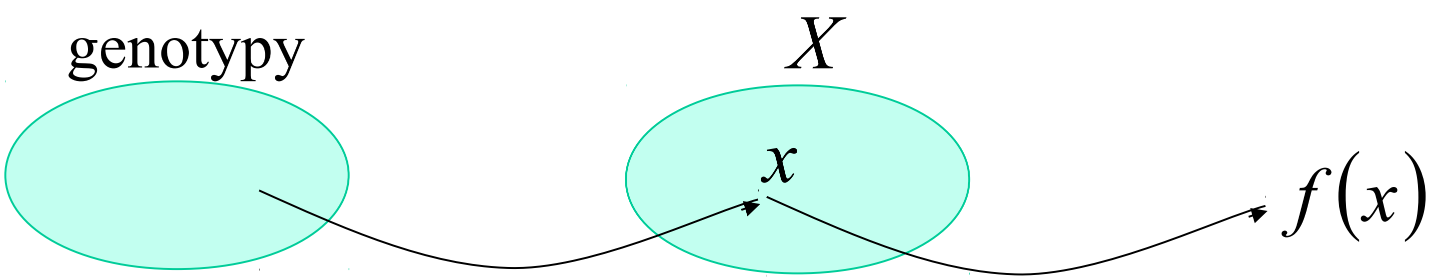
Strategie są określane jako: DE/best/1/bin, DE/rand/1/exp itd.

Uogólniony algorytm ewolucyjny

- Cel: znalezienie najlepszego elementu dziedziny = osobnika
- Osobniki tworzą populację
- Nowe osobniki powstają przez krzyżowanie i mutację już istniejących

Algorytmy genetyczne: motywacja

- Osobnik czyli *fenotyp*, kodowany przez pewien projekt – *genotyp*
- Przedmiotem krzyżowania i mutacji są genotypy
- Motywacja: prawie wszystko daje się zapisać przez sekwencje bitów



Algorytmy genetyczne: terminologia

- *fenotyp* – osobnik
- *genotyp* – kod osobnika
- *gen* – pojedynczy element kodu
- *locus* – pozycja w genotypie
- *chromosom* – zgrupowanie pozycji, na których geny kodują jakąś niezależną cechę fenotypu

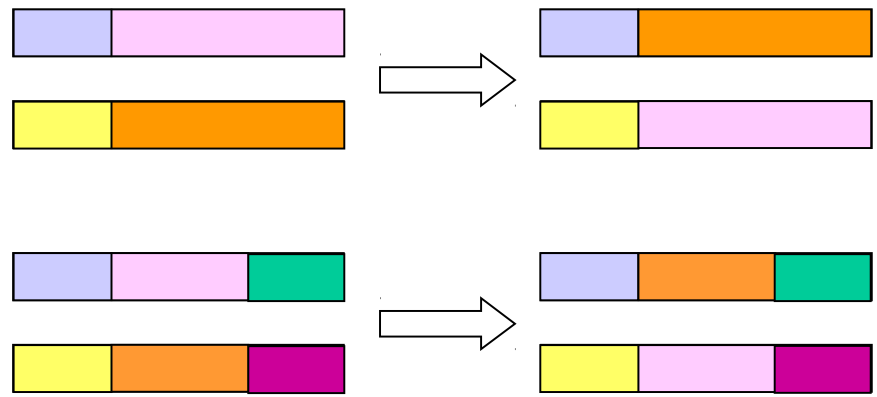
Ogólny algorytm genetyczny

1. Wygeneruj populację początkową genotypów P
2. Reprodukuj z par z P zbiór genotypów R przez zastosowanie krzyżowania a potem mutacji
3. Wybierz z $P \cup R$ nową populację P tworząc z każdego genotypu osobnika i oceniając go
4. Jeśli nie jest spełniony warunek stopu, wróć do punktu 2, w przeciwnym razie zwróć fenotyp najlepszego osobnika z P

Genotypu binarne: krzyżowanie

1. Krzyżowanie jednopunktowe
2. Krzyżowanie kilkupunktowe
3. Krzyżowanie równomierne:

   na danym locus gen od losowego rodzica



1. Wygeneruj populację początkową P
2. Reprodukuj z par z P zbiór genotypów R przez zastosowanie krzyżowania a potem mutacji
3. Wybierz z $P \cup R$ nową populację P
4. Jeśli nie jest spełniony warunek stopu, wróć do punktu 2.

Genotypy binarne: mutacja

1. Z zadanym prawdopodobieństwem p_m podlega jej każdy gen w genotypie
2. p_m powinno być małe; np. takie aby mutacji podlegał co 10-ty genotyp

Przykład rozróżnienia

- Problem komiwojażera
- Podejście ewolucyjne „klasyczne”
  - osobnikami są permutacje - na nich przeprowadzane są krzyżowanie i mutacja
- Podejście genetyczne
  - permutacje są reprezentowane przez wektory binarne - na nich przeprowadzane są krzyżowanie i mutacja

Co jeszcze w ramach tego podejścia

- Metody kodowania
- Algorytmy memetyczne:
  ewolucja + przeszukiwanie lokalne
- Programowanie ewolucyjne/genetyczne
- Hybrydy z innymi narzędziami sztucznej inteligencji, np. sieciami neuronowymi