Rój cząsteczek

Particle Swarm Optimization

Adam Grycner

Instytut Informatyki Uniwersytetu Wrocławskiego

18 maja 2011

Praca Kennedy'ego i Eberhart'a

- W 1995 *James Kennedy* i *Russell Eberhart* opublikowali pracę o tytule `Particle Swarm Optimization`.
- Zaprezentowali oni nową metodę optymalizacji ciągłych nieliniowych funkcji.
- Opierała się ona na dwóch metodach
  1. Sztuczne życie [*artificial life*] (stada ptaków, ławice ryb, teoria roju)
  2. Ewolucja obliczeniowa [*evolutionary computation*] (algorytmy genetyczne, strategie ewolucyjne, programowanie genetyczne)

Obserwacja ławic, stad

- powracanie do miejsca, gdzie wcześniej znaleziono jedzenie
- przekazywanie informacji o pożywieniu wewnątrz stada/ławicy

Cechy stada

- bezpieczeństwo ruchu
  zdolność do poruszania się bez ryzyka kolizji z przeszkodami znajdującymi się w przestrzeni, oraz z innymi osobnikami
- rozprzestrzenianie się
  zdolność do rozprzestrzeniania się stada w celu ustalenia i zarządzania minimalnymi odległościami między osobnikami
- skupianie
  zdolność do gromadzenia się osobników stada w celu ustalenia i zarządzania maksymalnymi odległościami między osobnikami
- orientacja
  zdolność znajdowania określonych regionów lub punktów w przestrzeni

Cechy stada

- ~~bezpieczeństwo ruchu~~
  W abstrakcyjnym środowisku jest nieistotne. Agenci mogą zajmować taką samą pozycję.
- ~~rozprzestrzenianie się~~
  Przy optymalizacji pragniemy zbieżności
- skupianie
- orientacja

Rój

Po wyeliminowaniu dwóch cech stada okazało się, że ruch agentów bardziej przypomina zachowanie roju cząsteczek, a nie stada/ławicy.

Cechy inteligentnego roju

Cechy inteligentnego roju według Millonas'a

- bliskość

  obsługa populacji powinna zajmować proste struktury i mało mocy obliczeniowej

- jakość

  ocena jakości cząstek

- różnorodność odpowiedzi

  populacja powinna nie popadać w ograniczony zbiór zachowań

- stabilność

  populacja nie powinna zmieniać zachowania przy każdej najmniejszej zmianie środowiska

- adaptacyjność

  populacja powinna umieć zmienić swoje zachowanie jeśli się to opłaca

Atrybuty cząsteczki

- położenie
- prędkość
- jakość
- najlepsze swoje położenie
- zbiór sąsiadów oraz najlepsze ich położenie

Wybory cząsteczki

Cząsteczka stara się poruszać zgodnie z trzema kierunkami

- zgodnie z poprzednim kierunkiem
- zgodnie z kierunkiem do najlepszej znalezionej pozycji
- zgodnie z kierunkiem do najlepszej pozycji znalezionej przez sąsiadów

Symulacja stada ptaków

Wzór 1

$$
\begin{aligned}
&\textit{if presentx[ ] > pbestx[gbest] then vx[ ] = vx[ ] - rand( )*g\_increment}\\
&\textit{if presentx[ ] < pbestx[gbest] then vx[ ] = vx[ ] + rand( )*g\_increment}\\
&\textit{if presenty[ ] > pbesty[gbest] then vy[ ] = vy[ ] - rand( )*g\_increment}\\
&\textit{if presenty[ ] < pbesty[gbest] then vy[ ] = vy[ ] + rand( )*g\_increment}
\end{aligned}
$$

Wzór 2 - dostosowanie prędkości w zależności od odległości od maksimum

$$vx[][] = vx[][] + rand()*p\_increment*(pbestx[][] - presentx[][])$$

Wzór 3 - statystycznie przez połowę czasu cząsteczka będzie przelatywać przez maksimum

$$vx[][] = vx[][] + 2 * rand() * (pbestx[][] - presentx[][]) + 2 * rand() * (pbestx[][gbest] - presentx[][])$$

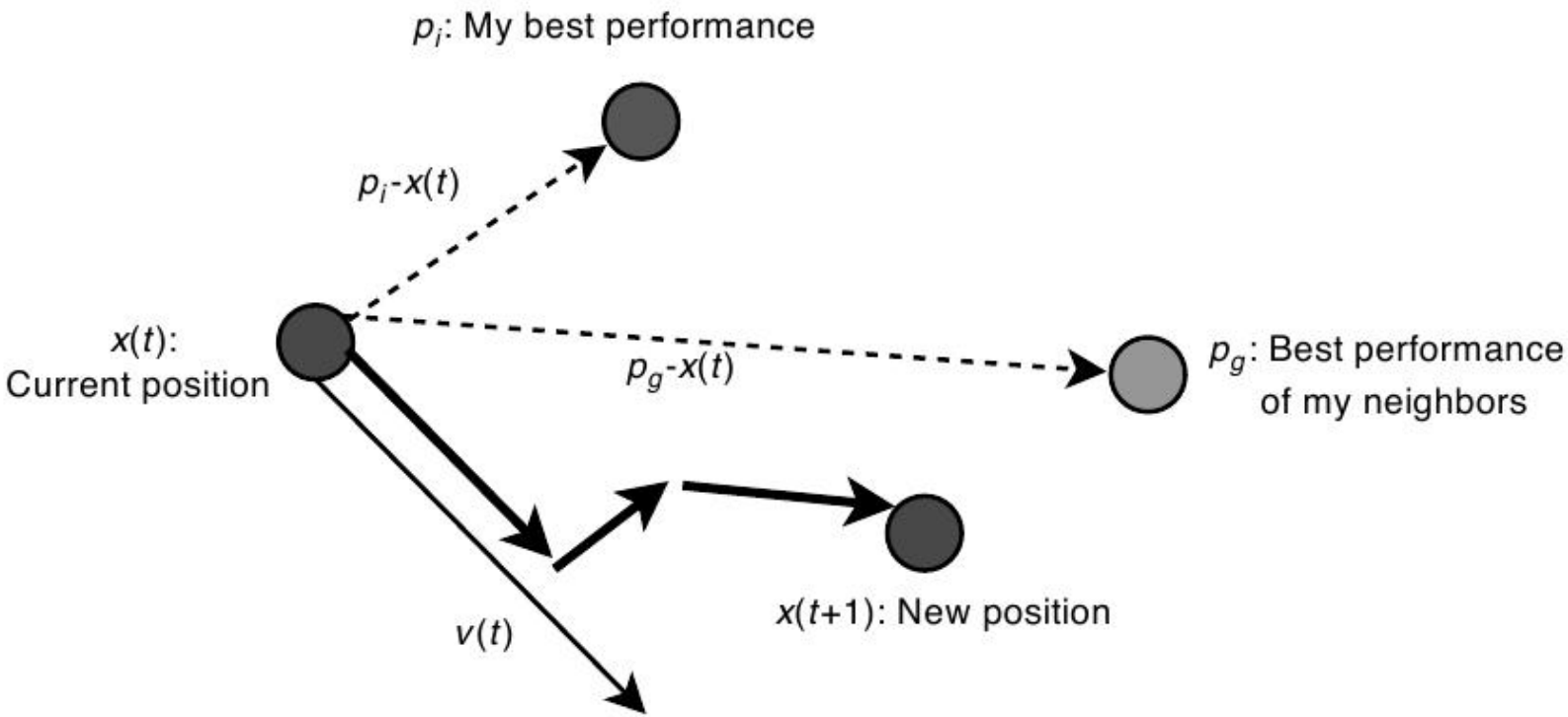
Wzór 4 - ostateczny wzór:

Wzór na ruch cząsteczki

$$v = c_1 r_1 v + c_2 r_2 (pbest - x) + c_3 r_3 (gbest - x)$$
$$x = x + v$$

gdzie

- v - prędkość cząsteczki
- x - położenie cząsteczki
- pbest - najlepsze znalezione przez cząsteczkę położenie
- gbest - najlepsze znalezione przez zbiór sąsiadów położenie
- c_1, c_2, c_3 - współczynniki określające wpływ poszczególnych elementów
- r_1, r_2, r_3 - wartości losowe z rozkładu jednostajnego $U(0, 1)$

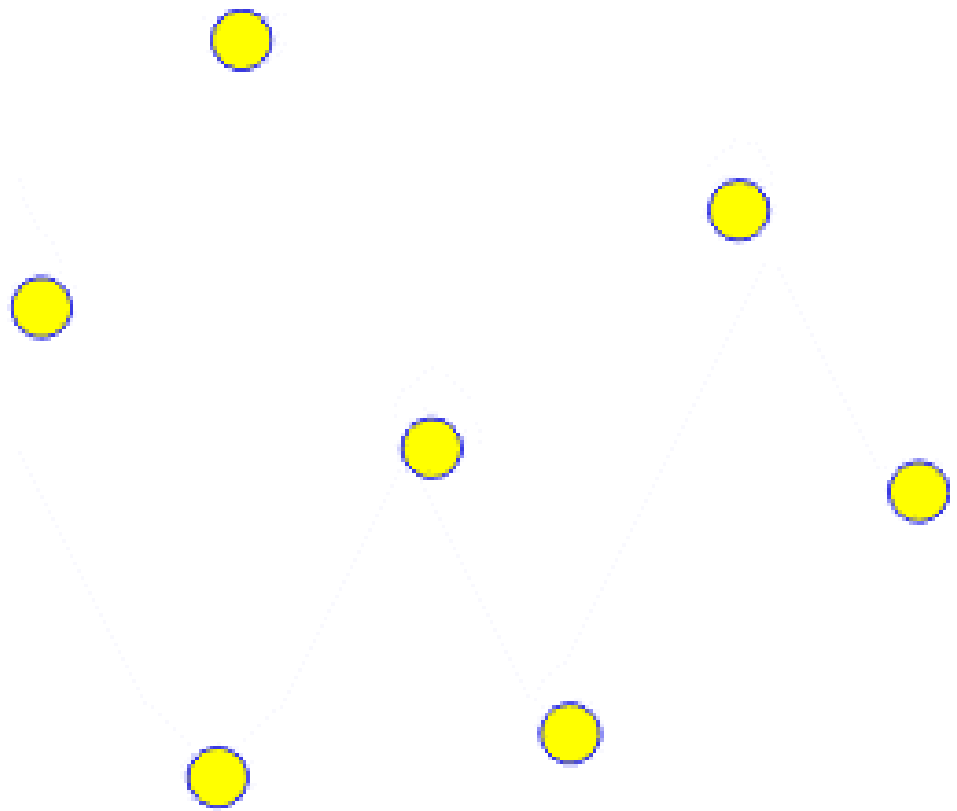

p_i: My best performance
p_i-$x(t)$
$x(t)$:
Current position
p_g-$x(t)$
p_g: Best performance
of my neighbors
$v(t)$
$x(t+1)$: New position

Dla $c_1 = r_1 = 1$ cząsteczki wykonują *swarming movement* wokół:

$$\frac{1}{c_2+c_3}(c_2 pbest + c_3 gbest)$$

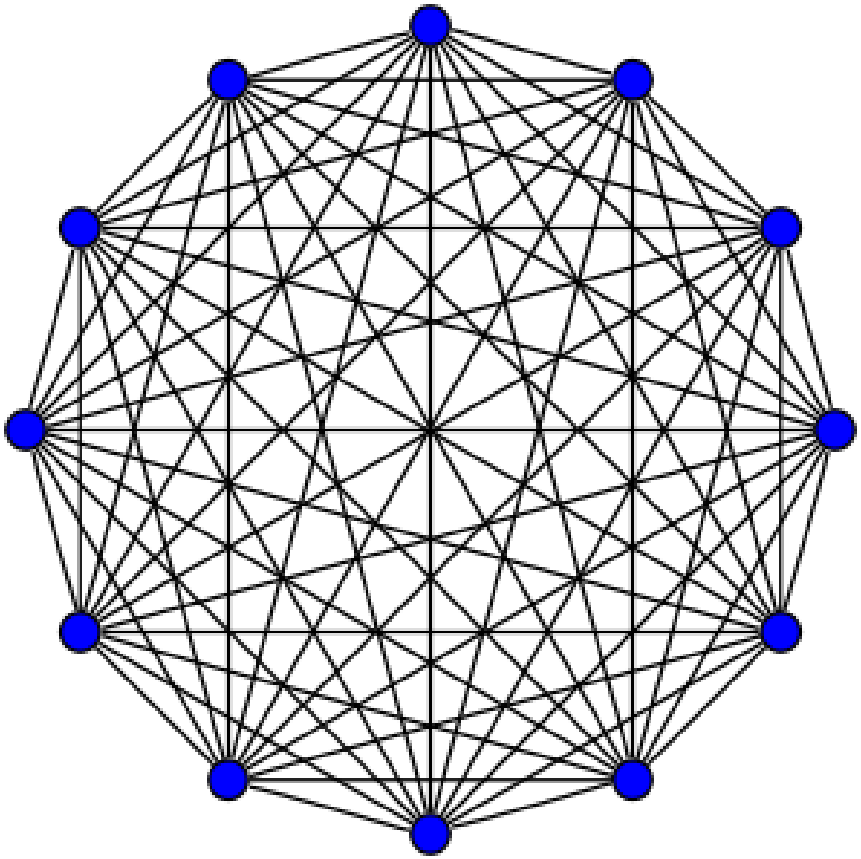
```
Wylosuj początkowe położenia i prędkości roju
while kryterium_zatrzymania
  dla każdej cząsteczki i
    zaktualizuj prędkość według wzoru 4
    zaktualizuj pozycję według wzoru 4
    jeżeli f(nowa_pozycja) < f(pbest_i):
      pbest_i = nowa_pozycja
    jeżeli f(nowa_pozycja) < f(gbest_i):
      gbest_i = nowa_pozycja
```

Brak sąsiedztwa



- działa słabo

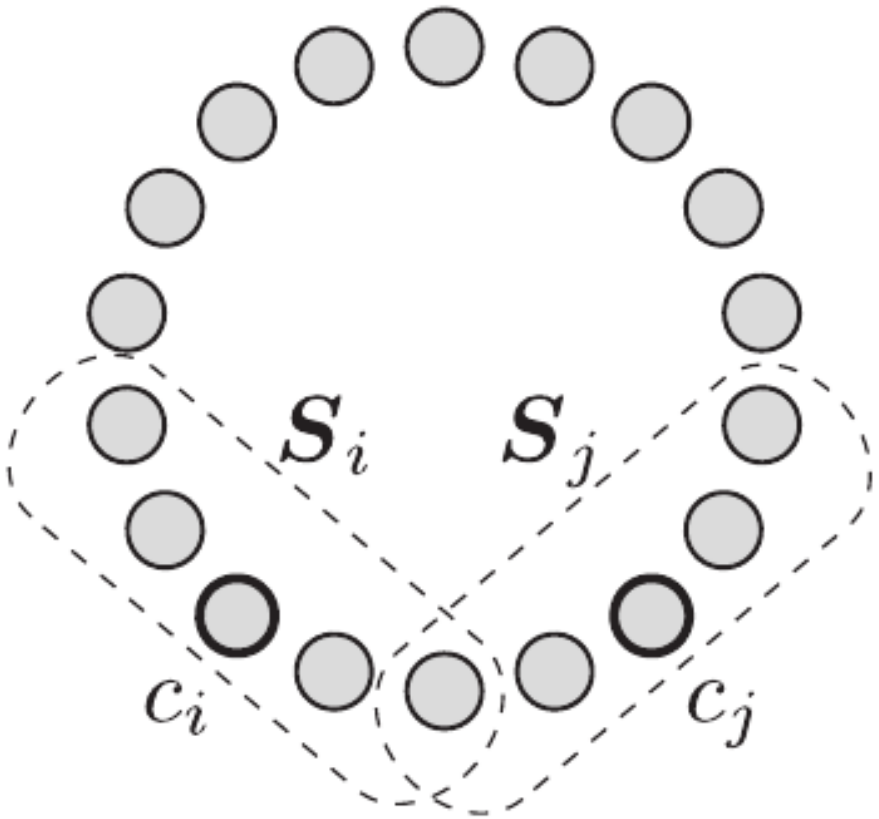
Sąsiedztwo globalne



- łatwe w implementacji
- wymaga niskich kosztów obliczeniowych
- szybko zbiega

Sąsiedztwo według indeksu

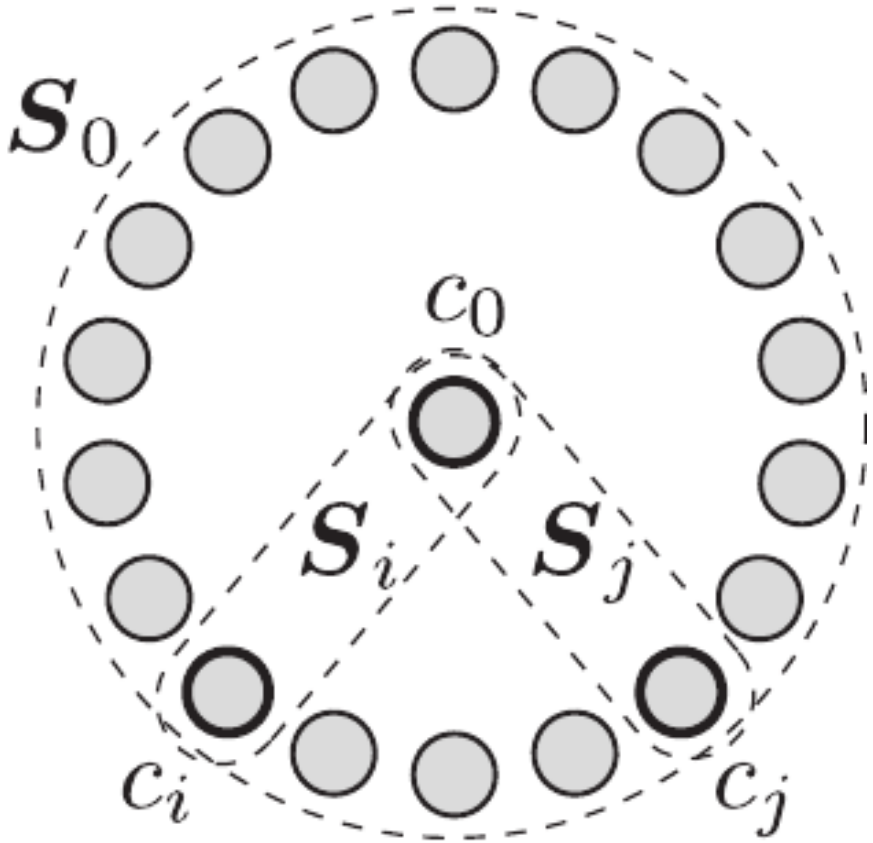
Pierścień



- Sąsiadami są cząsteczki o indeksach $< i - l; i + l >$ modulo liczba cząsteczek

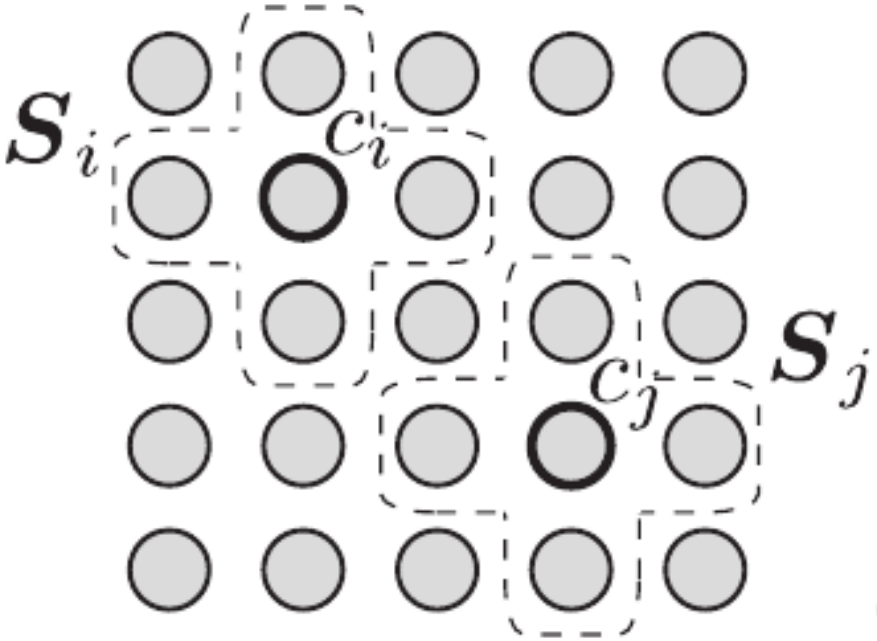
Sąsiedztwo według indeksu

Koło



Sąsiedztwo według indeksu

Von Neumann



Sąsiedztwo przestrzenne

- Po każdej iteracji szukamy k najbliższych cząsteczek według ustalonej metryki

Sąsiedztwo przestrzenne

Szukanie k najbliższych sąsiadów według parametru

$$\psi_{ab} = \frac{|pos_a - pos_b|}{max_{i,j \in S}|pos_i - pos_j|}$$

lub wykorzystanie dynamicznej wartości progowej

$$frac(k) = \frac{3k + 0.6k_{max}}{k_{max}}$$

gdzie

k_{max} - przewidywana maksymalna liczba iteracji
k - liczba iteracji

- kosztowne obliczeniowo
- lepiej przeszukuje przestrzeń przeszukiwań

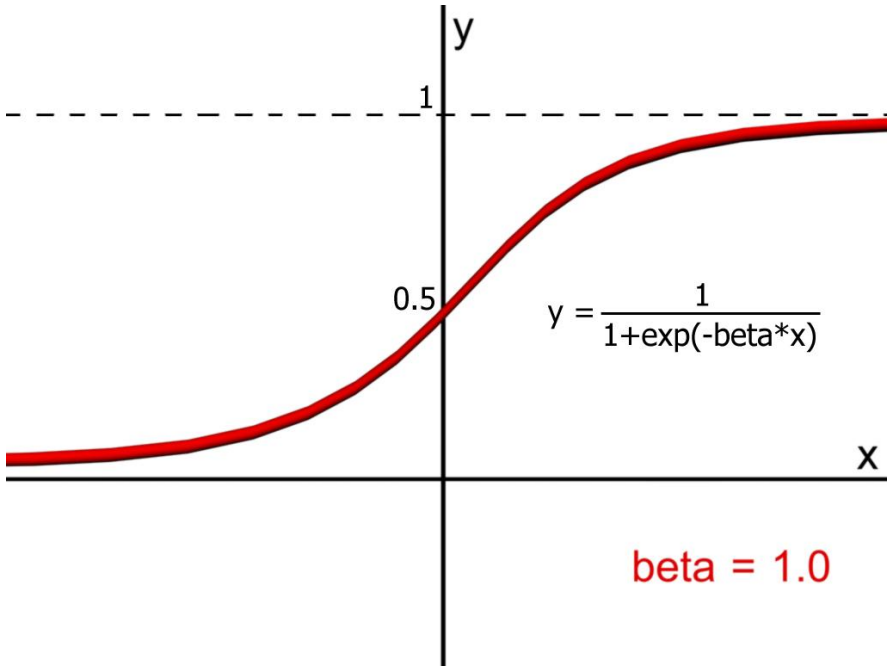
Inne sąsiedztwa

- topologia gwiazdy
- topologia z "błędem" (zaburzenie poprzednich wersji)
- piramida

- Particle Swarm Optimization zostało zaprojektowane do optymalizacji rzeczywistych funkcji ciągłych.
- Jednak istnieją modyfikacje pozwalające optymalizować problemy, gdzie rozwiązaniami są wektory binarne lub wektory liczb całkowitych

Wersja binarna

- Wykorzystanie funkcji sigmoidalnej



Funkcja sigmoidalna

$$sig(v_{i,j}^t) = \frac{1}{1+exp(-v_{i,j}^t)}$$

Wersja binarna

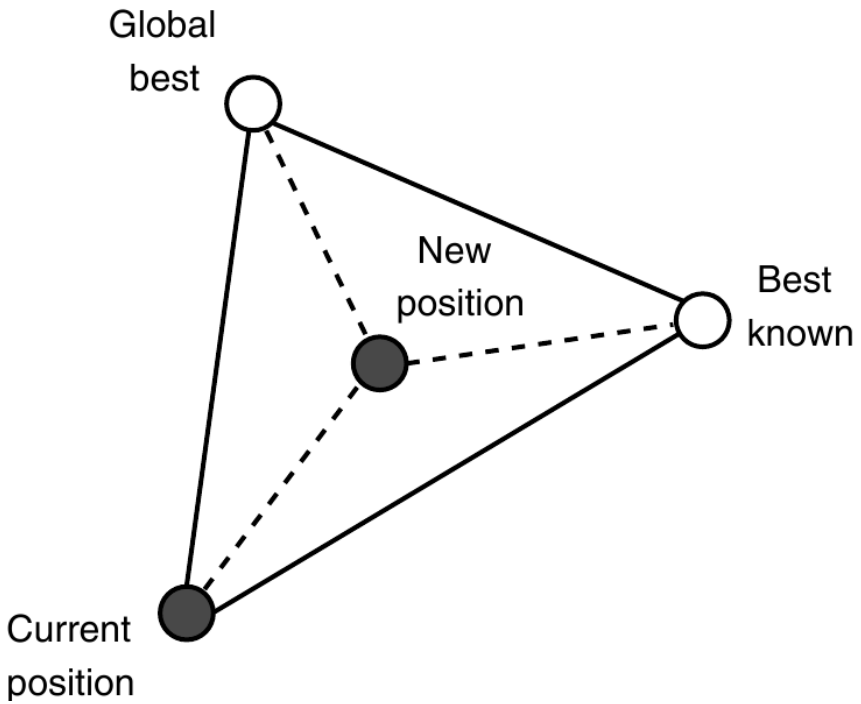
- Wykorzystanie funkcji sigmoidalnej

Nowe współrzędne cząsteczki

$$x_{i,j}^{t+1} = \begin{cases} 0 & \text{jeżeli } r_{3,j}^{t} \geqslant sig(v_{i,j}^{t+1}) \\ 1 & \text{w.p.p.} \end{cases}$$

- $r_{3,j}^{t} = U(0,1)$

Geometric PSO



- Wykorzystane krzyżowanie z trzema rodzicami
- z ppb. w_1 na pozycji x_j przepiszemy zawartość j-tej pozycji poprzedniego położenia
- z ppb. w_2 na pozycji x_j przepiszemy zawartość j-tej pozycji *pbest*
- z ppb. w_3 na pozycji x_j przepiszemy zawartość j-tej pozycji *gbest*

PSO dla problemów kombinatorycznych

Należy zdefiniować, czym jest:

- położenie cząsteczki
  (np. permutacja)
- prędkość
  (np. lista transpozycji)
- odejmowanie[pozycja, pozycja]
  (wynik: wektor prędkości między x_1, a x_2)
- mnożenie zewnętrzne[skalar, prędkość]
- dodawanie[wektor,wektor]
- zastępowanie[pozycja, wektor]
  zaaplikowanie położenia do prędkości

Współczynnik inercji

Dodatkowy współczynnik do równania na prędkość cząsteczki

Wzór na prędkość

$$v = c_1 r_1 v + c_2 r_2 (pbest - x) + c_3 r_3 (gbest - x)$$

- c_1 maleje z czasem
- duża inercja - aktywna eksploracja całej dziedziny
- mała inercja - przeszukiwanie lokalnej

Limit prędkości

W celu uniknięcia zjawiska *eksplozji* (ucieczki cząsteczek w pierwszych iteracjach) często wprowadza się limit prędkości cząsteczki i-tej na pozycji k-tej.

$$v'_{i,k} = \begin{cases} v_{i,k} & \text{jeżeli } v_{i,k} \leqslant v_{max} \\ v_{max} & \text{jeżeli } v_{i,k} > v_{max} \end{cases}$$

lub

$$v'_{i,k} = \begin{cases} v_{i,k} & \text{jeżeli } v \leqslant v_{max} \\ \frac{v_{max}}{v} v_{i,k} & \text{jeżeli } v > v_{max} \end{cases}$$

gdzie $v = |v_i|$

Selekcja

1. wybierz połowę cząsteczek
   - selekcja turniejowa
   - lepsza połowa
2. skopiuj położenia lepszych cząsteczek w miejsce gorszych
3. pozostaw informację *pbest* oraz prędkości gorszych cząsteczek

Wzmocnienie przeszukiwania lokalnego kosztem globalnego.

Zbyt szybkie zbieganie do ekstremów lokalnych.

Hodowla cząsteczek

Reprodukcja i rekombinacja

- nadanie cząsteczkom ppb. wyboru, jako rodzica
- wylosowanie rodziców i skrzyżowanie arytmetyczne położeń oraz prędkości

$$\begin{cases} pos'_a &= rpos_a + (1-r)pos_b \\ pos'_b &= rpos_b + (1-r)pos_a \end{cases}$$

$$\begin{cases} v'_a &= \frac{v_a+v_b}{|v_a+v_b|}|v_a| \\ v'_b &= \frac{v_a+v_b}{|v_a+v_b|}|v_b| \end{cases}$$

$r \in (0,1)$

Inne

- mutacja
- specjacja
- różnicowanie (*diversity*) - niszowanie, dzielenie funkcji przystosowania itd.

Adaptacja

Istnieją modyfikacje PSO, w którym w raz z upływem czasu zmienia się:

- rozmiar roju
- rozmiar sąsiedztwa cząsteczki
- współczynniki c_1, c_2, c_3

Multiswarm PSO

- Wprowadzamy kilka rojów, każdy o innym kolorze
- Cząsteczki o różnych kolorach odpychają się
- Wprowadzamy siłę odpychającą, którą dodamy do wyliczanej prędkości

$$\frac{\lambda}{|x-y|^2}\frac{x-y}{|x-y|} = \frac{\lambda}{|x-y|^3}(x-y)$$

$$\lambda = \begin{cases} 0 & \text{jeżeli cząsteczki są tego samego koloru} \\ 1 & \text{jeżeli cząsteczki są różnego koloru} \end{cases}$$

Dobre do optymalizacji wielomodalnej

Bibliografia

 Maurice Clerc, *Particle Swarm Optimization*

 J. Kennedy, R. Eberhart, *Particle Swarm Optimization*

 K. Kameyama, *Particle Swarm Optimization – A Survey*

 E. Talbi, *Metaheuristics : from design to implementation*

 materiały dostarczone przez prowadzącego

Dziękuję za uwagę