Bumblebees algorithm

Obliczenia z wykorzystaniem metod sztucznej inteligencji

Paweł Patyk, Bartosz Paradowski

Spis treści

Inspiracja

Inspiracja

Inspiracją do stworzenia tego algorytmu były trzmiele. Jako owady żyjące w roju tworzą skomplikowaną strukturę powiązań między samymi osobnikami, a otaczającym ich środowiskiem. Pokazują integrację wielu pojedynczych, względnie niezależnych od siebie jednostek w jedną samo zarządzającą się strukturę. Przedstawiają w ten sposób inne podejście do uczenia się, ponieważ jest ono mocno uzależnione od ich życia w roju. Pomimo, że nie znane są nam jeszcze dokładne mechanizmy operowania tych wysoce zaawansowanych struktur, to pozwalają one na zaobserwowanie pewnych specyficznych zachowań, które możemy zastosować w różnego rodzaju przemysłach, jak i w tym wypadku, również w algorytmach, gdzie pokazują jak z w teorii chaotycznej informacji wyciągnąć istotne informacje, aby pozyskać wartościowe dane.

Wprowadzenie

Wprowadzenie

Algorytm trzmielowy jest jednym z algorytmów genetycznych. Algorytm został stworzony, aby pozwolić na uzyskanie oczekiwanego wyniku szybciej niż w przypadku algorytmu mortals and angels. Algorytm trzmielowy wykorzystany został do kolorowania grafu.

Wprowadzenie

W algorytmie tym w przeciwieństwie do standardowych tego typu, funkcja fitness nie odpowiada za obliczanie jakości rozwiązania, a jest ona silnie powiązana z życiem konkretnego osobnika.

Algorytm

Algorytm

```
Bumblebees Algorithm():
Begin
Initialize random solutions;
    Generate N random solutions of the problem;
Set MaxGenerations,
Initialize an n × m cells world;
    Create the colony nest with N bumblebees;
    Put F food cells at random;
    Associate a random solution to each bumblebee;
    Assign the lifespan of each bumblebee;
        accordingly to its solution fitness;
Repeat Until (currentGeneration > MaxGenerations) Do
    Look for the best bumblebee;
    If (its solution is the global optimum) Then
        Report solution and exit algorithm;
    endIf
    Decrease life counters;
    Reap bumblebees;
    Create a new bumblebee in the nest
        every G generations;
    Move randomly every bumblebees to a neighbor cell;
    If (a bumblebee finds food) Then
        Decrease food counter;
        Increase lifespan of the bumblebee;
        Move bumblebee back to the nest;
    endIf
    Mutate bumblebees solutions;
    Recalculate fitnesses and assign new lifespans;
    Increment currentGeneration;
endDo
End.
```

Opis algorytmu

Opis algorytmu

Świat - generowana jest plansza n x n, na której każda z poszczególnych komórek może być pusta, zawierać jedzenie, trzmiela bądź gniazdo

Poruszanie się - na początku algorytmu wszystkie trzmiele umieszczane są w gnieździe. Kolejne osobniki są wypuszczane przy każdej kolejnej generacji. Trzmiele poruszają się w losowy sposób na którykolwiek z 24 najbliższych pól. Po dotarciu do jedzenie trzmiel natychmiast wraca do gniazda, pozycja jedzenia jest zapisywana w królowej roju oraz zwiększana jest długość życia osobnika o 2. Na początku każde wygenerowanie jedzenie jest w ilości 20, z każdym dotarciem do niego przez trzmiela zmniejszane jest o jeden. W momencie dojścia licznika do zera, nowy pokarm jest generowany.

Narodziny trzmiela - co kilka generacji, w gnieździe tworzony jest nowy trzmiel, który otrzymuje najlepsze rozwiązanie od królowej która je zapisuje. Utrzymywanie najlepszych rozwiązań pozwala na efektywniejszą pracę algorytmu.

Opis algorytmu cd

Mutacja - przeprowadzana jest w przeciwieństwie do standardowych algorytmów genetycznych, co każdą generację. Powoduje ona stworzenie trzmiela z najlepszym na obecną chwilę możliwym kolorem.

Żniwiarz - z racji że funkcja celu w głównej mierze oparta jest na długości życia trzmiela, został wprowadzony mechanizm żniwiarza, który powoduje, że trzmiel którego kolor pola łączy się przy ponad 40% kratek jest usuwany z planszy. Ponadto każdorazowo, po skończeniu generacji, trzmiele których długość życia jest równa 0 umierają.

Porównanie z innymi algorytmami

Porównanie

Vert	Edges	*Colors*			*Time*			*Gen.*			*# Suc.*		
		GA	A&M	B	GA	A&M	B	GA	A&M	B	GA	A&M	B
30	44	3	3	**2**	0.08	0.02	**0.12**	22	43	**113**	10	20	**20**
50	123	4	4	**2**	0.39	0.13	**0.38**	85	256	**306**	9	19	**20**
70	242	5	5	**3**	0.90	0.28	**0.40**	159	672	**325**	3	19	**19**
100	495	6	6	**4**	2.01	0.68	**0.50**	206	1875	**431**	2	17	**18**
200	1990	12	12	**10**	5.25	1.66	**0.71**	201	3465	**347**	5	17	**17**

Figure: Porównanie problem kolorowania grafów dla gęstości krawędzi 10% odpowiednio dla algorytmu genetycznego (GA), algorytmu angels mortals (AM) oraz dla algorytmu bumblebees (B)

Porównanie

Vert	Edges	*Colors*			*Time*			*Gen.*			*# Suc.*		
		GA	A&M	B	GA	A&M	B	GA	A&M	B	GA	A&M	B
30	87	4	4	**2**	0.21	0.06	**0.18**	84	167	**201**	8	19	**20**
50	245	6	6	**3**	0.63	0.20	**0.33**	115	503	**280**	8	19	**19**
70	483	8	7	**3**	1.11	0.54	**0.47**	229	1524	**522**	5	13	**19**
100	990	11	10	**7**	2.06	0.81	**0.49**	166	2155	**340**	8	16	**18**
200	3980	20	19	**19**	6.19	2.05	**0.79**	390	4751	**503**	6	16	**15**

Figure: Porównanie problem kolorowania grafów dla gęstości krawędzi 20% odpowiednio dla algorytmu genetycznego (GA), algorytmu angels mortals (AM) oraz dla algorytmu bumblebees (B)

Porównanie

Porównanie algorytmu bumblebees z podstawowym algorytmem genetycznym i algorytmem angels mortals, było przeprowadzane dla świata o wymiarach od 15×15 do 40×40, dla maksymalnej liczby generacji od 1000 do 20000. W zdecydowanej liczbie przypadków algorytm bumblebees znajduję lepsze wyniki od algorytmu genetycznego i bliźniaczej metody angles & mortals.

Różnice

- angels & mortals w momencie spotkania z angel’em następuje wygenerowania nowego osobnika na jednej z pobliskiej pozycji, w Bumblebees nowy osobnik jest generowany w gnieździe raz na kilka generacji.
- w algorytmie Bumblebees istnieje gniazdo o stałej pozycji, które nie występuje w algorytmie angels & mortals

Bibliografia I

1. Francesc de Paula Comellas Padro, Jesus Martınez Navarro, et al. Bumblebees: a multiagent combinatorial optimization algorithm inspired by social insect behaviour. 2011.
2. F. Comellas and R. Gallegos. Angels mortals: A newcombinatorial optimization algorithm. Stud. Fuzziness Soft.Comput. 166: 397–405, 2005.

KONIEC