Obliczenia Naturalne - Algorytmy Mrówkowe

Paweł Paduch

Politechnika Świętokrzyska

8 maja 2014

Plan wykładu

Literatura


Marco Dorigo, Thomas Stützle - *Ant Colony Optimization. Bradford Company*, Scituate, MA, USA, 2004


Alberto Colorni, Marco Dorigo, Vittorio Maniezzo - *Distributed optimization by ant colonies* European Conference on Artificial Life, strony 134–142, 1991.


M. Dorigo, V. Maniezzo, A. Colorni - *The ant system: An autocatalytic optimizing process.* Raport instytutowy 91-016 Revised, Milano, Italy, 1991


Marco Dorigo, V. Maniezzo, Alberto Colorni - *Positive feedback as a search strategy.* Raport instytutowy, 1991.

Kolonie mrówek

Algorytmy mrówkowe należą do grupy algorytmów metaheurystycznych. Z powodzeniem stosuje się je do rozwiązywania dyskretnych zadań liniowych. W 1989 Deneubourg wykazał, że mrówki potrafią znaleźć najkrótszą drogę do pożywienia. Mrówki swą zbiorowa inteligencję zawdzięczają wspólnemu działaniu tysięcy jednostek. Fenomen ten był inspiracją do stworzenia grupy algorytmów optymalizujących ACO (ang. *Ant Colony Optimization*). Jednym z pierwszych badaczy, który spopularyzował ACO był Marco Dorigo.

Kolonie mrówek

Algorytmami mrówkowymi z powodzeniem rozwiązuje się:

- problem komiwojażera,
- problem kwadratowego przydziału,
- harmonogramowania zadań,
- równoważenia obciążenia łącz komunikacyjnych,
- grupowania i wyszukiwania.

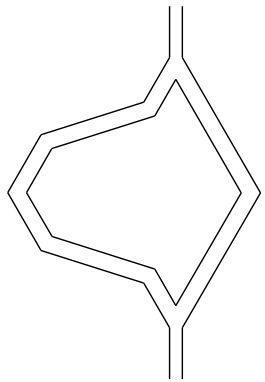
Mrówki

Naturalne mrówki charakteryzują się tym że:

- Są praktycznie ślepe.
- Mają znikome mózgi.
- Mają bardzo dobre zmysły węchu.
- Mrówki maszerując zostawiają na swej drodze feromony.
- Feromony nieustannie parują.
- Mrówki podczas decyzji, którą z dróg wybrać kierują się intensywnością zapachu feromonu.
- Korzystają ze zjawiska stygmergii (zjawisko pośredniej komunikacji poprzez wywoływanie zmian w środowisku i odczytywanie ich).
- Gdy działają w stadzie potrafią znaleźć najkrótszą drogę pomiędzy pożywieniem a mrowiskiem.

Eksperyment Deneubourg'a

Deneubourg badając argentyńskie mrówki połączył źródło pożywienia z mrowiskiem dwoma mostami o różnej długości. Rozdzielenie drogi było pod kątem 30°, by wykluczyć wybór którejś ze ścieżek ze względu na jej lokalny kształt. Po kilku minutach od znalezienia pożywienia większość mrówek maszerowała krótszą trasą.



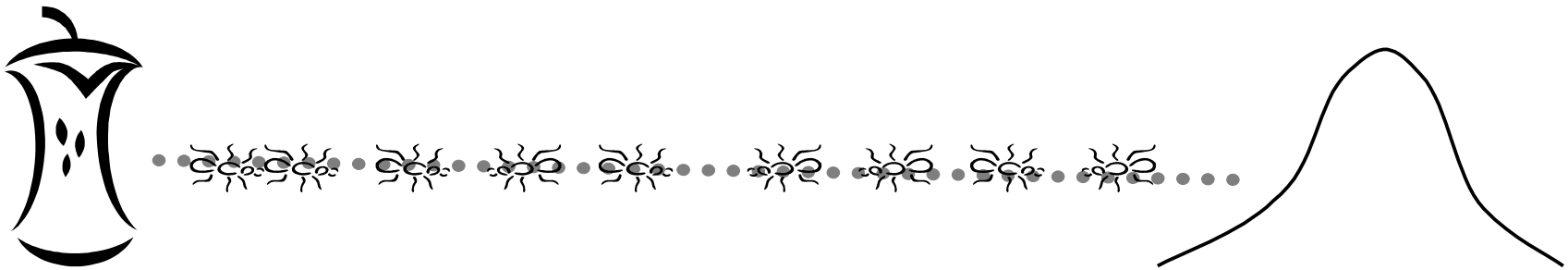
Rysunek: Most w eksperymencie Deneubourg'a

Eksperyment Deneubourg'a

Deneubourg wykazał, że takie zachowanie można wytłumaczyć za pomocą prostego modelu probabilistycznego, w którym każda mrówka decyduje, którą trasę wybrać z pewnym prawdopodobieństwem zależnym od ilości chemicznej substancji zwanej feromonem znajdującej się na trasach. Na podobnej zasadzie mrówki potrafią przebudować feromonowy szlak gdy zostanie on przerwany przez ustawioną przeszkodę.

Jak mrówki omijają przeszkody?

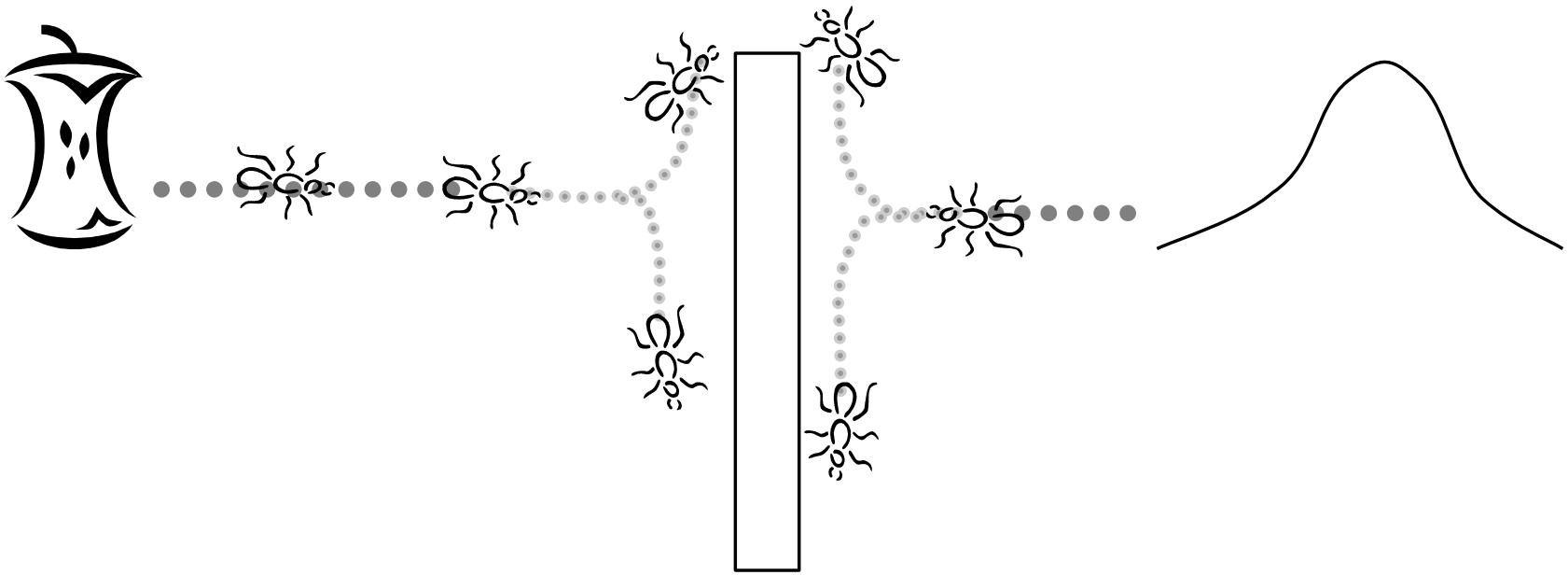
Mrówki mają ustaloną trasę.



Rysunek: Mrówki z wyznaczoną trasą feromonową pomiędzy pożywieniem a mrowiskiem.

Jak mrówki omijają przeszkody?

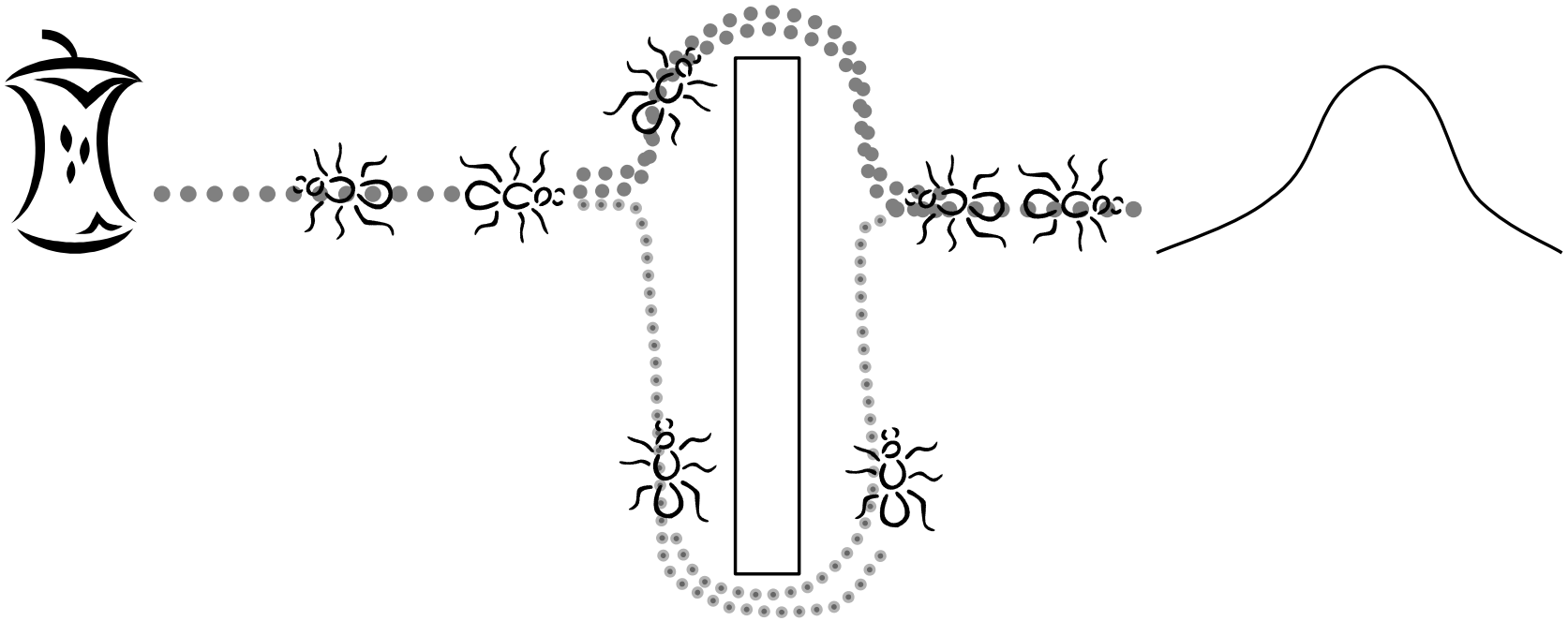
Gdy trafią na przeszkodę część pójdzie z jednej strony część z drugiej



Rysunek: Przeszkoda na trasie feromonowej, mrówki wybierają dwie alternatywne trasy.

Jak mrówki omijają przeszkody?

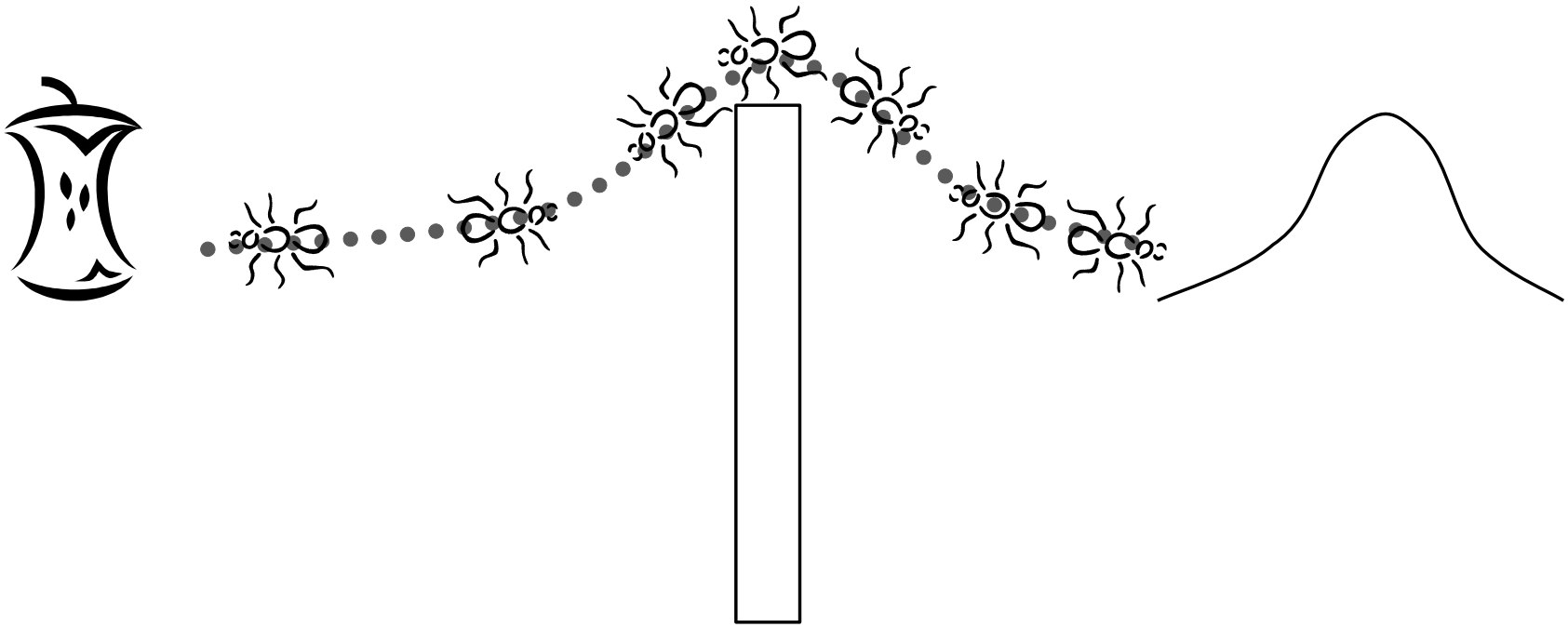
Mrówki idące krótszą drogą szybciej połączą ślady feromonowe dając jednocześnie sygnał dla pozostałych mrówek, która trasa jest lepsza.



Rysunek: Na trasie krótszej ilość feromonu zwiększa się szybciej.

Jak mrówki omijają przeszkody?

Z czasem mrówki zoptymalizują swoją trasę.



Rysunek: Trasa dłuższa odparowuje, pozostaje lepsze, optymalne rozwiązanie.

Jak mrówki szukają pożywienia?

- Mrówki losowo rozchodzą się z mrowiska szukając pożywienia.
- Te mrówki, które trafią krótszą drogą, szybciej wrócą po swoich śladach.
- Na krótszych trasach liczba mrówek w jednostce czasu jest większa a więc i odświeżanie śladu feromonowego jest intensywniejsze.
- Trasy dłuższe, jako rzadziej odwiedzane wyparowują.

Cechy wirtualnych mrówek

- Poruszają się w dyskretnej przestrzeni poszukiwań od punktu do punktu w zdefiniowanych grafach.
- Mogą aktualizować ślad feromonowy nie tylko w trakcie ale i po znalezieniu rozwiązania.
- Nie muszą zostawiać za sobą identycznego śladu, mogą to robić w zależności od jakości rozwiązania. W szczególności tylko jedna, najlepsza mrówka może zostawić swój ślad.

Cechy wirtualnych mrówek

- Mogą „widzieć" w pewnym ograniczonym zakresie, np. dostrzegając różnicę w długości krawędzi wychodzących z aktualnego węzła, w którym mrówka się znajduje.
- Mogą być obdarzone pewną dawką inteligencji, np. takiej by wyszukiwały trasy krótsze pomimo tej samej ilości feromonu.
- Mogą posiadać wewnętrzną pamięć np. w formie list *tabu*, mówiących o tym, których węzłów należy unikać (bo zostały już odwiedzone).

Problem komiwojażera

Znalezienie minimalnego cyklu Hamiltona w pełnym grafie ważonym $G = (N, E)$ dla zbioru miast N oraz krawędzi E, to problem komiwojażera (ang. *Traveling Salesman Problem - TSP*). Jest to klasyczny przypadek problemu NP-trudnego i wiele nowych algorytmów metaheurystycznyh jest testowanych za pomocą TSP. Tak zrobił M. Dorigo w przypadku swojego algorytmu mrówkowego. Jeżeli dystans pomiędzy miastami i oraz j, gdzie $i, j \in N$ oznaczymy przez d_{ij} oraz $d_{ij} \neq d_{ji}$, to mamy do czynienia z TSP asymetrycznym.

Problem komiwojażera

Minimalny cykl Hamiltona polega na odwiedzeniu wszystkich $n = |N|$ miast grafu G dokładnie raz oraz znalezieniu minimum funkcji $f(\pi)$ danej wzorem:

$$f(\pi) = \sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)} + d_{\pi(n)\pi(1)}$$

gdzie: π jest permutacją węzłów o indeksach $\{1, 2, ..., n\}$

Podstawowy algorytm mrówkowy

Ogólny algorytm mrówkowy wygląda następująco:

1. Wylosuj dla każdej mrówki miasto początkowe.
2. Na podstawie heurystyki oraz lokalnej ilości feromonu wybierz kolejny węzeł/krawędź.
3. Gdy osiągniesz cel uaktualnij ilość feromonu według sprecyzowanych do danego algorytmu zasad.
4. Jeżeli kryterium zatrzymania nie zostało spełnione powtórz czynności.

Podstawowe założenia

- W systemie bierze udział m mrówek.
- Każda mrówka losuje początkowy węzeł, po czym porusza się z miasta i do miasta j po krawędzi $E(i,j)$.
- Dystans pomiędzy kolejnymi miastami i oraz j o współrzędnych odpowiednio x_i, y_i oraz x_j, y_j określony jest odległością euklidesową $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.
- Zawartość feromonu na krawędzi $E(i,j)$ w chwili t będzie oznaczona przez $\tau_{ij}(t)$.
- Ilość feromonu $\tau_{ij}(0)$ powinna być stosunkowo mała i na każdej krawędzi jednakowa.

Aktualizacja feromonu

Z każdą iteracją obliczeń stężenie feromonu jest aktualizowane według wzoru:

$$\tau_{ij}(t+1) = \rho * \tau_{ij}(t) + \Delta\tau_{ij}(t, t+1), \qquad (1)$$

gdzie:

- ρ - współczynnik z przedziału $< 0, 1 >$, który określa jaka część feromonu ma pozostać (0 - wyparuje wszystko, 1 - nic nie wyparuje). By uniknąć nieograniczonego odkładania się feromonu oraz pozwolić słabszym trasom zaniknąć, ρ powinno $\in (0, 1)$.
- $\Delta\tau_{ij}(t, t+1) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t, t+1)$
- $\Delta\tau_{ij}^{k}(t, t+1)$ jest ilością feromonu na każdą jednostkę długości krawędzi (i, j) rozkładanego przez k-tą mrówkę w czasie pomiędzy t a $t+1$.

Lista tabu

By spełnić założenie komiwojażera, że można odwiedzić każde miasto jeden raz, wprowadzono *listę tabu*, w której pamiętane są odwiedzone miasta.
Gdy mrówki odwiedzą wszystkie miasta, lista jest czyszczona i w kolejnej iteracji zapełniana od nowa.
$\mathbf{tabu_k}(s)$ oznacza element s wektora $\mathbf{tabu_k}$ dla k-tej mrówki (czyli, w którym mieście s znajduje się mrówka k w bieżącej trasie).

Wybór kolejnego miasta

Prawdopodobieństwo wyboru kolejnego miasta j dla k-tej mrówki stojącej w mieście i dane jest wzorem:

$$p_{ij}(t) = \begin{cases} \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{c_{i,l}\in\Omega}(\tau_{il})^{\alpha}(\eta_{il})^{\beta}} & \forall c_{i,l} \in \Omega \\ 0 & \forall c_{i,l} \notin \Omega, \end{cases} \quad (2)$$

gdzie:

- l - możliwe miasto,
- Ω - dopuszczalne rozwiązania (pozostałe nieodwiedzone miasta, nienależące do $\mathbf{tabu_k}$),
- η_{ij} - wartość lokalnej funkcji kryterium, w tym przypadku odwrotność odległości pomiędzy węzłami $\eta = \frac{1}{d_{ij}}$, (visibility)
- α - parametr regulujący wpływ τ_{ij},
- β - parametr regulujący wpływ η_{ij}.

Trzy rodzaje AS

W zależności od tego jak jest wyliczana $\Delta\tau_{ij}^{k}(t, t+1)$ oraz kiedy uaktualniać τ_{ij} możemy wyróżnić 3 algorytmy, które zaproponował Dorigo w swoim systemie mrówkowym AS.

- Algorytm Cykliczny (ang. *ant-cycle CAS*)
- Algorytm Gęstościowy (ang. *ant-density DAS*)
- Algorytm Ilościowy (ang. *ant-quantity QAS*)

Aktualizacja feromonu

W algorytmie gęstościowym za każdym razem gdy mrówka przechodzi po krawędzi (i, j), na każdą jednostkę jej długości rozkładana jest stała ilość feromonu Q_1.

$$\Delta\tau_{ij}^{k}(t, t+1) = \begin{cases} Q_1 & \text{jeżeli } k\text{-ta mrówka przechodzi z } i \text{ do } j \\ & \text{w czasie pomiędzy } t \text{ a } t+1 \\ 0 & \text{dla pozostałych.} \end{cases} \tag{3}$$

Ilość feromonu jest zawsze stała i niezależna od długości krawędzi czy całej trasy.

Algorytm DAS - Krok 1

Algorytm 1 Algorytm DAS - Krok 1

1: {Krok 1 Inicjalizacja}
2: $t := 0$ {licznik czasu}
3: **for all** krawędzi (i, j) **do**
4: ustaw początkową wartość $\tau_{ij}(t)$ oraz $\Delta\tau_{ij}(t, t+1) := 0$
5: **end for**
6: Wstaw $b_i(t)$ w każdym węźle i {$b_i(t)$ oznacza liczbę mrówek w węźle i w czasie t}
7: s:=1 {s jest indeksem w tablicy tabu}
8: **for** i:=1 to n **do**
9: **for** k:=1 to $b_i(t)$ **do**
10: $\mathbf{tabu_k}(s) = i$ {Pierwszy element tablicy to miasto startowe}
11: **end for**
12: **end for**

Algorytm DAS - Krok 2

Algorytm 2 Algorytm DAS - Krok 2

13: **repeat**{Krok 2 – będzie powtarzany $n-1$ razy}
14: s:=s+1
15: **for** i:=1 to n **do**
16: **for** k:=1 to $b_i(t)$ **do**
17: wybierz kolejne miasto j według prawdopodobieństwa p_{ij} danego wzorem 2
18: k-ta mrówka idzie do miasta j {Tworzymy nową wartość $b_j(t+1)$}
19: dodaj miasto j do $\mathbf{tabu_k}(s)$
20: $\Delta\tau_{ij}(t, t+1) = \Delta\tau_{ij}(t, t+1) + Q_1$
21: **end for**
22: **end for**
23: **for all** krawędzi (i, j) **do**
24: Oblicz $\tau_{ij}(t+1)$ według wzoru 1
25: **end for**
26: **until** lista tabu nie będzie pełna

Algorytm DAS - Krok 3

Algorytm 3 Algorytm DAS - Krok 3

27: Zapamiętaj najkrótszą trasę ze wszystkich m mrówek
28: **if** $LC < LC_{min}$ lub (wszystkie mrówki nie poszły tą samą drogą) **then**
29: $\{LC$ - liczba cykli$\}$
30: Wyczyść wszystkie listy tabu
31: s:=1
32: **for** i:=1 to n **do**
33: **for** k:=1 to $b_i(t)$ **do**
34: $\mathbf{tabu_k}(s) = i$ {znowu każda k-ta mrówka jest w początkowym mieście}
35: **end for**
36: **end for**
37: $t = t + 1$
38: **for all** krawędź (i, j) **do**
39: $\Delta\tau_{ij}(t, t + 1) = 0$
40: **end for**
41: Idź do kroku 2
42: **else**
43: Wypisz najkrótszą trasę i się zatrzymaj
44: **end if**

Opis algorytmu

- Mrówki są ustawiane w różnych miastach
- Inicjowana jest początkowa ilość feromonu na krawędziach
- Pierwszy element listy tabu jest jednocześnie miastem startowym.
- Mrówki wychodząc z miasta *i* wybierają miasto docelowe *j* na podstawie prawdopodobieństwa danego wzorem 2, w którym są dwa parametry ustawiane przez użytkownika
  - α - steruje w jakim stopniu mrówka kieruje się doświadczeniem poprzednich pokoleń (wartość τ_{ij} - niesie ze sobą informację jak wiele mrówek przeszło tą trasą w niedalekiej przeszłości).
  - β - określa chęć wybrania miasta na podstawie jego odległości (η_{ij} - określa stopień *widoczności* miasta *j* z miasta *i*).
- Jeżeli parametr $\alpha = 0$ mrówki działają jak w stochastycznym algorytmie zachłannym startującym z wielu punktów.

Opis algorytmu

- Za każdym razem, gdy mrówka przechodzi po krawędzi (i, j) zostawia feromon, który jest sumowany z feromonem pozostawionym na tej krawędzi w przeszłości.
- Kiedy wszystkie mrówki wykonają swój ruch, z każdej krawędzi jest odparowywana część feromonu według wzoru 1.
- Po $n - 1$ ruchach listy tabu są wypełnione.
- Obliczana i zapamiętywana jest najkrótsza trasa znaleziona przez m mrówek.
- Następnie listy tabu są zerowane.

Opis algorytmu - warunek zakończenia

- proces jest powtarzany przez LC iteracji podanych przez użytkownika lub do momentu, kiedy wszystkie mrówki wybiorą identyczną trasę.
- Przypadek taki nazywano zachowaniem jedno-ścieżkowym (ang. *uni-path behavior*). Oznacza on, że mrówki zaprzestały wyboru alternatywnych tras.

QAS

Model ilościowy systemu mrówkowego różni się tym od gęstościowego, że stała ilość feromonu Q_2 jest dzielona przez długość krawędzi d_{ij}.

$$\Delta\tau_{ij}^{k}(t,t+1) = \begin{cases} \frac{Q_2}{d_{ij}} & \text{jeżeli } k\text{-ta mrówka przechodzi z } i \text{ do } j \\ & \text{w czasie pomiędzy } t \text{ a } t+1 \\ 0 & \text{dla pozostałych.} \end{cases} \tag{4}$$

Uzależnienie ilości feromonu od odwrotności długości krawędzi d_{ij} powoduje, że mrówki wybierają chętniej krótsze krawędzie.

Różnica w aktualizacji feromonu

W algorytmie cyklicznym systemu mrówkowego wprowadzono jedną istotną zmianę. $\Delta\tau_{i,j}^k$ nie jest obliczana w każdym kroku ale po skończonej całej trasie (n-krokach). Wartość $\Delta\tau_{i,j}^k$ dana jest wzorem 5:

$$\Delta\tau_{ij}^k(t, t+n) = \begin{cases} \frac{Q_3}{L^k} & \text{jeżeli } k\text{-ta mrówka użyła krawędzi } ij \\ & \text{w swojej trasie} \\ 0 & \text{dla pozostałych,} \end{cases} \tag{5}$$

gdzie Q_3 jest stałe a L^k jest długością trasy znalezionej przez k-tą mrówkę.

Aktualizacja feromonu

Wartość feromonu jest aktualizowana po n krokach według wzoru:

$$\tau_{ij}(t+n) = \rho_1 * \tau_{ij}(t) + \Delta\tau_{ij}(t, t+n), \tag{6}$$

gdzie

$$\Delta\tau_{ij}(t, t+n) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t, t+n),$$

ρ_1 jest nieco inne, gdyż uaktualnienie nie następuje po każdym kroku, a dopiero po dojściu mrówki do celu. Jest on niemal identyczny ze wzorem 1 z tą różnicą, że uaktualnianie jest po n krokach.

Algorytm CAS - krok 1

Algorytm 4 Algorytm CAS (krok 1)

1: {Krok 1 Inicjacja}
2: $t := 0$ {licznik czasu}
3: **for all** krawędzi (i, j) **do**
4: ustaw początkową wartość $\tau_{ij}(t)$ oraz $\Delta\tau_{ij}(t, t + n) := 0$
5: **end for**
6: Wstaw $b_i(t)$ mrówek w każdym węźle i {$b_i(t)$ oznacza liczbę mrówek w węźle i w czasie t}
7: s:=1 {s jest indeksem w tablicy tabu}
8: **for** i:=1 to n **do**
9: **for** k:=1 to $b_i(t)$ **do**
10: $\mathbf{tabu_k}(s) = i$ {Pierwszy element tablicy to miasto startowe}
11: **end for**
12: **end for**

Algorytm CAS - krok 2

Algorytm 5 Algorytm CAS (krok 2)

13: {Krok 2 – będzie powtarzany $n-1$ razy}
14: **repeat**
15: s:=s+1
16: **for** i:=1 to n **do**
17: **for** k:=1 to $b_i(t)$ **do**
18: wybierz kolejne miasto j według prawdopodobieństwa p_{ij} danego wzorem 2
19: k-ta mrówka idzie do miasta j {Tworzymy nową wartość $b_j(t+1)$}
20: dodaj miasto j do $\mathbf{tabu_k}(s)$
21: **end for**
22: **end for**
23: **until** lista tabu nie będzie pełna

Algorytm CAS - kroki 3-4

Algorytm 6 Algorytm CAS (kroki 3-4)

24: {Krok 3}
25: **for** $k := 1$ to m **do**
26: $(h, l) := (\mathbf{tabu_k}(s), \mathbf{tabu_k}(s+1))$ {(h, l) - jest krawędzią łączącą miasto $(s, s+1)$ w liście tabu dla k-tej mrówki }
27: $\Delta\tau_{hl}(t+n) = \Delta\tau_{hl}(t+n) + \frac{Q_3}{L^k}$
28: **end for**
29: {Krok 4}
30: **for all** krawędzi (i, j) **do**
31: Oblicz $\tau_{ij}(t+n)$ według wzoru 6
32: **end for**
33: $t = t + n$
34: **for all** krawędzi (i, j) **do**
35: $\Delta\tau_{ij}(t, t+n) := 0$
36: **end for**

Algorytm CAS - krok 5

Algorytm 7 Algorytm CAS (krok 5)

37: {Krok 5}
38: Zapamiętaj najkrótszą trasę do tej pory
39: **if** $LC < LC_{max}$ lub (wszystkie mrówki nie poszły tą samą drogą) **then**
40: {LC - liczba cykli}
41: Wyczyść wszystkie listy tabu
42: s:=1
43: **for** i:=1 to n **do**
44: **for** $k := 1$ to $b_i(t)$ **do**
45: $\mathbf{tabu_k}(s) = i$ {znowu każda k-ta mrówka jest w początkowym mieście}
46: **end for**
47: **end for**
48: Idź do kroku 2
49: **else**
50: Wypisz najkrótszą trasę i się zatrzymaj
51: **end if**

Złożoność obliczeniowa

Złożoność obliczeniowa algorytmu cyklicznego wynosi $O(LCn^2m)$. Dorigo wraz z zespołem znalazł liniową zależność pomiędzy liczbą miast a liczbą mrówek. Ostatecznie można więc uznać, że złożoność obliczeniowa to $O(LCn^3)$.

Tablice decyzyjne

W celu usprawnienia działania algorytmu często można spotkać w implementacji tablice decyzyjne $A_i = [a_{ij}(t)]_{N_i}$. Obliczane są za każdym razem gdy startuje nowe pokolenie mrówek na nowej zmodyfikowanej mapie feromonowej. Każde miasto i ma taką tablice policzoną. Elementy a_{ij} tablicy wyznaczamy:

$$a_{ij}(t) = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}{\sum_{l \in N_i}[\tau_{il}(t)]^{\alpha}[\eta_{il}(t)]^{\beta}} \forall j \in N_i$$

Gdzie: N_i - zbiór możliwych połączeń wychodzących z i.

Prawdopodobieństwo wybrania węzła

Używając tablic decyzyjnych prawdopodobieństwo wyboru kolejnego miasta $j \in N_i^k$ z bieżącego węzła i dla danej mrówki k dane jest wzorem:

$$p_{ij}^k(t) = \frac{a_{ij}(t)}{\sum_{l \in N_i^k} a_{il}(t)}$$

Przy czym $N_i^k \subseteq N_i$ jest zestawem sąsiednich miast jeszcze nie odwiedzonych.

Zalecane parametry

W AS Dorigo zaleca następujące parametry:

- $\alpha = 1$
- β od 2 do 5
- $\rho = 0,5$
- $m = n$, gdzie m liczba mrówek a n liczba miast
- $\tau_0 = \frac{m}{C^{nn}}$, gdzie C^{nn} jest oszacowaną długością trasy np. algorytmem „najbliższy sąsiad" (ang. *nearest-neighbor*).

Pytania

?

KONIEC

Dziękuję Państwu za uwagę.