Alte metaeuristici inspirate de natură

- Swarm Intelligence – comportamentul inteligent al mulțimilor (inteligența roiurilor)
  - ACO - Ant Colony Optimization - Modelul coloniei de furnici
  - PSO - Particle Swarm Optimization - Modelul ansamblului de particule (sau a stolului de păsări)
  - ABC - Artificial Bee Optimization – Modelul roiului de albine

Swarm intelligence (inteligență colectivă)

- Swarm intelligence = domeniu care cuprinde tehnici inteligente bazate pe comportamentul colectiv al unor sisteme cu auto-organizare și fără control centralizat

- Termen introdus în 1989 de Gerardo Beni si Jing Wang in contextul sistemelor de roboți

- Tehnicile din “swarm intelligence” se bazează pe mulțimi de agenți caracterizati prin:

  – Reguli simple de “funcționare”
  – Interacțiuni locale
  – Absența unor structuri de control centralizat

Swarm intelligence (inteligență colectivă)

- Exemple de sisteme naturale având astfel de caracteristici:
  - Colonii de furnici
  - Roiuri de albine
  - Stoluri de păsări
  - Bancuri de pești

- Reprezintă modele pentru tehnici de rezolvare a unor probleme de optimizare sau de analiză a datelor

Imagini de la http://www.scs.carleton.ca/~arpwhite/courses/95590Y/notes/SI%20Lecture%203.pdf

Modelul coloniei de furnici

Sursa de inspirație: comportarea furnicilor în procesele de

- Căutare a hranei -> rezolvarea unei probleme de optimizare: identificarea drumului optim între sursa de hrană și cuib

- Organizare a coloniei -> rezolvarea unei probleme de grupare a datelor: separarea hranei de corpurile furnicilor moarte sau a larvelor după dimensiuni sau segregarea furnicilor aparținând unor specii diferite

Elemente cheie:

- Comunicare indirectă prin intermediul unor substanțe chimice numite feromoni; acest proces de comunicare este denumit stigmergie

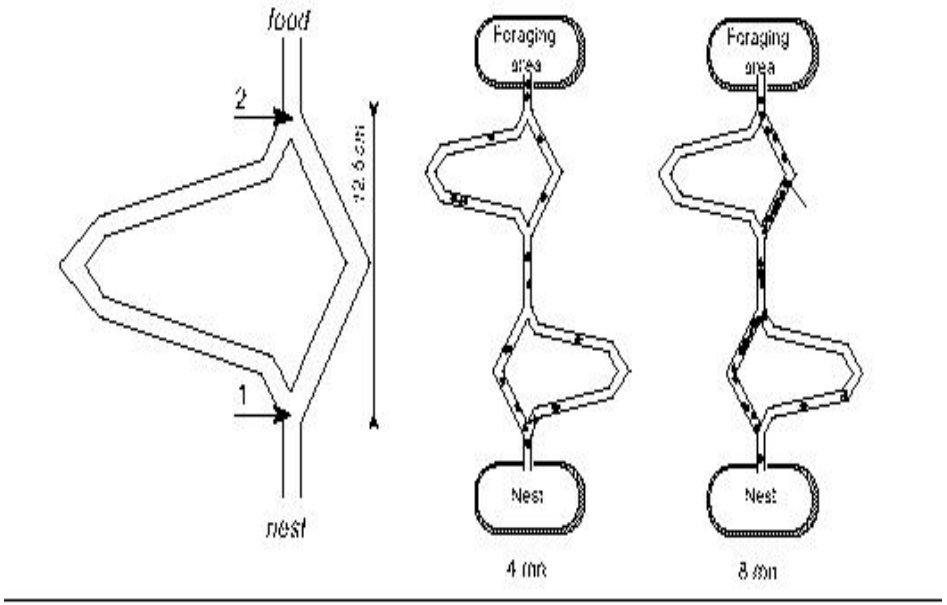
- Stabilirea similarității dintre furnici pe baza mirosului (o furnică recunoaște dacă o altă furnică face parte din același cuib sau nu)

Modelul coloniei de furnici

Rolul feromonilor: experimentul podului dublu [Deneubourg, 1990]

Specia de furnici analizată: Argentine

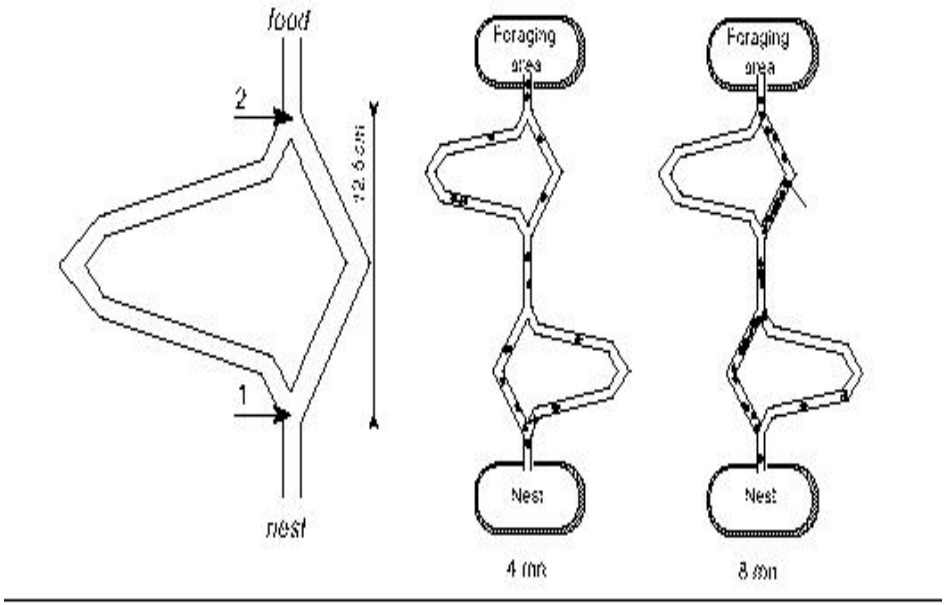
- Două căi de acces între cuib și sursa de hrană
- Inițial furnicile aleg la întâmplare una dintre căi
- La fiecare parcurgere a drumului furnicile depun feromoni
- Drumul mai scurt este parcurs de mai multe ori așa că va acumula o cantitate mai mare de feromoni



Modelul coloniei de furnici

Rolul feromonilor: experimentul podului dublu

- Dacă există diferență între cantitatea de feromoni depusă pe cele două trasee, furnicile vor prefera traseul marcat mai intens
- Treptat din ce în ce mai multe furnici vor alege traseul cu mai mulți feromoni contribuind și mai mult la sporirea cantității de feromoni (feedback pozitiv)

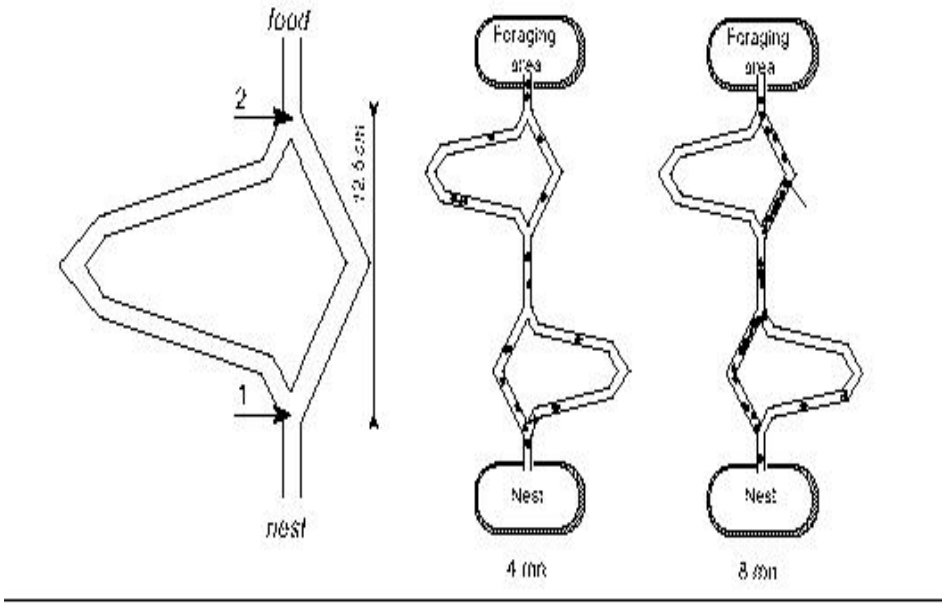


Modelul coloniei de furnici

Rolul feromonilor: experimentul podului dublu

- Cantitatea de feromon nu crește permanent ci poate și să scadă ca efect al unui proces de evaporare

- Procesul de evaporare este util în cazul aparițiilor unor schimbări în mediu



Ilustrare: http://www.nightlab.ch/downloads.php

Modelul coloniei de furnici

Rezolvarea unei probleme de optimizare – Ant Colony Optimization

Idee: soluția problemei este identificată folosind o mulțime de furnici artificiale (agenți) care schimbă informații privind calitatea soluției

Exemplu: problema comis-voiajorului

Intrare: graf etichetat corespunzator conexiunilor dintre orașe și costurilor corespunzătoare parcurgerii unei comexiuni

Ieșire: o ordine de parcurgere a orașelor caracterizată prin cost total minim

Modelul coloniei de furnici

ACO pentru problema comis voiajorului:

- Se utilizează o populație de furnici care sunt implicate într-un proces iterativ

- La fiecare iterație fiecare furnică parcurge câte un traseu în graful asociat problemei. La parcurgerea traseului furnicile respectă următoarele reguli:
  - Nu trec de două ori prin același nod
  - Decizia de a alege o muchie este aleatoare, iar probabilitatea de selecție depinde atât de costul muchiei cât și de cantitatea de feromon asociată muchiei

- După construirea traseelor se actualizează cantitatea de feromoni corespunzătoare muchiilor astfel încât muchiilor ce fac parte din trasee de cost mic să li se asocieze o cantitate mai mare de feromoni.

Modelul coloniei de furnici

Structura generală a algoritmului

Iniţializarea concentraţiilor τ_{ij} cu valori aleatoare mici
for $t = 1, tmax$ **do**
  for $k = 1, a$ **do**
    $i_1 = 1$
    for $p = 2, n$ **do**
      alege i_p conform cu probabilitatea $P^k_{i_{p-1}, i_p}(t)$
    Calculează costul L_k al traseului ales de furnica k
  Reţine cea mai bună dintre cele a soluţii găsite
  Actualizează valorile concentraţiilor τ_{ij}

Notații:

tmax = număr iterații; a=număr agenți (furnici); i_p = indice nod

P = probabilitate de tranziție, L = cost traseu, tau = concentrație feromoni

Modelul coloniei de furnici

Variante:

Algorithm	*Authors*	*Year*
Ant System (AS)	Dorigo et al.	1991
Elitist AS	Dorigo et al.	1992
Ant-Q	Gambardella & Dorigo	1995
Ant Colony System	Dorigo & Gambardella	1996
$\mathcal{MAX}$–$\mathcal{MIN}$ AS	Stützle & Hoos	1996
Rank-based AS	Bullnheimer et al.	1997
ANTS	Maniezzo	1999
BWAS	Cordón et al.	2000
Hyper-cube AS	Blum et al.	2001

Obs: variantele diferă între ele în principal prin modul de calcul al probabilității de tranziție și regula de actualizare a concentrației de feromoni

Modelul coloniei de furnici

Particularități ale variantei inițiale (Ant Systems)

Problema comis voiajorului

Reprezentarea soluției: $(i_1,i_2,\ldots,i_n)$ permutare a mulțimii de indici ai orașelor

Probabilități de tranziție
(furnica k trece la momentul t de la orașul i la orașul j)

$$P^k(i,j,t) = \begin{cases} \dfrac{(\tau_{ij}(t))^\alpha (\omega_{ij}(t))^\beta}{\sum_{l \in V(i,k)} (\tau_{il}(t))^\alpha (\omega_{il}(t))^\beta} & V(i,k) \neq \emptyset \\ 0 & altfel \end{cases}$$

Concentrația de feromon corespunzătoare arcului (i,l)

Factor invers proporțional cu costul arcului (i,l)

Lista oraselor nevizitate încă de furnica k și care sunt conectate cu orasul i

Modelul coloniei de furnici

Varianta tradițională pt TSP (AS - Ant Systems)

Actualizarea concentrației de feromoni
(la sfârșitul fiecărei iterații)

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta_{ij}^{k}$$

$$\Delta_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k} & \text{daca furnica k parcurge (i, j)} \\ 0 & \text{altfel} \end{cases}$$

Notatii:
ρ = rată de evaporare
Q>0 = constantă
L_k = cost al ultimului traseu parcurs de furnica k

Varianta:

Concentrația de feromoni se actualizează utilizând doar informațiile corespunzătoare celui mai bun traseu:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta_{ij}^{*}$$

$$\Delta_{ij}^{*} = \begin{cases} \dfrac{Q}{L^*} & \text{daca } T^* \text{ contine (i, j)} \\ 0 & \text{altfel} \end{cases}$$

Modelul coloniei de furnici

Particularități ale altor variante:

Max-Min Ant System (MMAS):

- concentrația de feromoni corespunzătoare fiecărui arc este limitată la valori cuprinse într-un interval
- la sfârșitul fiecărei iterații se modifică concentrația de feromoni doar pentru arcele corespunzătoare celui mai bun traseu

Ant Colony System (ACS)

- utilizează și o ajustare locală a concentrației de feromoni aplicată ori de câte ori este vizitat un arc (pe lângă ajustarea globală similară cu cea ce la variantă Max-Min):

$$\tau_{ij} = (1 - \varphi) \cdot \tau_{ij} + \varphi \cdot \tau_0 \ ,$$

valoarea initiala
a concentratiei

Modelul coloniei de furnici

Exemple de aplicatii

Problem type	*Problem name*	*Authors*	*Year*
Routing	Traveling salesman	Dorigo et al.	1991, 1996
		Dorigo & Gambardella	1997
		Stützle & Hoos	1997, 2000
	Vehicle routing	Gambardella et al.	1999
		Reimann et al.	2004
	Sequential ordering	Gambardella & Dorigo	2000
Assignment	Quadratic assignment	Stützle & Hoos	2000
		Maniezzo	1999
	Course timetabling	Socha et al.	2002, 2003
	Graph coloring	Costa & Hertz	1997
Scheduling	Project scheduling	Merkle et al.	2002
	Total weighted tardiness	den Besten et al.	2000
	Total weighted tardiness	Merkle & Middendorf	2000
	Open shop	Blum	2005
Subset	Set covering	Lessing et al.	2004
	l-cardinality trees	Blum & Blesa	2005
	Multiple knapsack	Leguizamón & Michalewicz	1999
	Maximum clique	Fenet & Solnon	2003
Other	Constraint satisfaction	Solnon	2000, 2002
	Classification rules	Parpinelli et al.	2002
		Martens et al.	2006
	Bayesian networks	Campos, Fernández-Luna,	2002
	Protein folding	Shmygelska & Hoos	2005
	Docking	Korb et al.	2006

Modelul coloniei de furnici

Aplicații în probleme reale:

- Probleme de rutare în rețele de telecomunicații (optimizare în medii dinamice)
- Probleme de stabilire a rutelor pentru vehicule
- Probleme de planificare a task-urilor

Companii care au aplicat ACO în rezolvarea problemelor:

www.eurobios.com (routing/schedule of airplane flights, supply chain networks)

www.antoptima.com (vehicle routing)

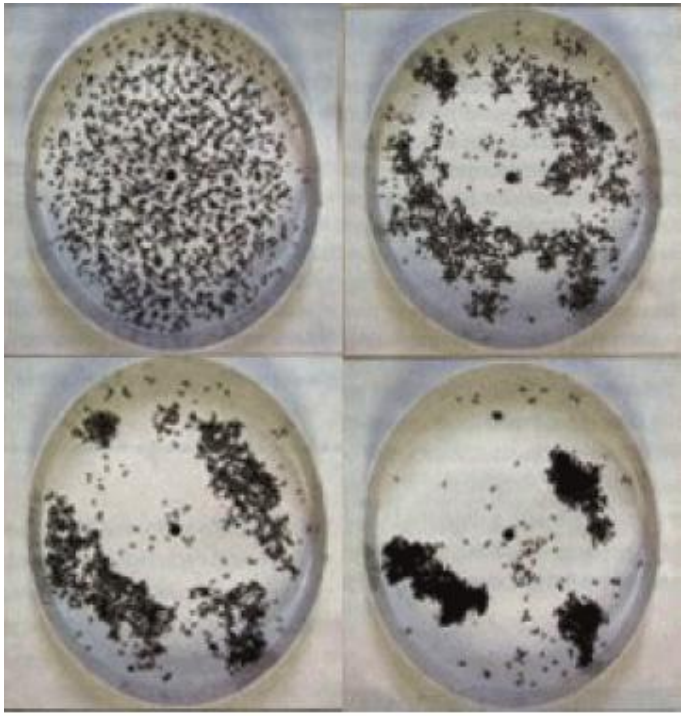
Modelul coloniei de furnici

Utilizare în gruparea datelor.

Folosește ca sursă de inspirație

- procesul prin care furnicile separă larvele după dimensiuni sau furnicile moarte (Lumer &Faieta, 1994)

- Modul în care furnicile identifică furnicile aparținând altei specii care pătrund în cuibul lor (AntClust – Labroche, 2002)



Modelul coloniei de furnici

AntClust – algoritm pentru gruparea datelor [Labroche, 2002]

Colonia de furnici

- Furnica
- Cuib (furnici de acelasi tip)
- Tip de miros
- Intalnirea a doua furnici
- Crearea unui cuib
- Migrarea furnicilor intre cuiburi
- Eliminarea unei furnici din cuib

Proces de grupare a datelor

- Data
- Cluster (clasa de date similare)
- Prag de similaritate
- Compararea a doua date
- Initierea unui cluster
- Transfer de date de la un cluster la altul
- Eliminarea unei date dintr-un cluster

Modelul coloniei de furnici

- Pentru gruparea a n date sunt folosite n furnici fiecare caracterizată prin:
  - O dată asociată, x
  - O etichetă corespunzătoare clusterului, L
  - Un prag de similaritate, T
  - Un contor al întâlnirilor cu alte furnici, A
  - O măsură a dimensiunii cuibului (percepția proprie), M
  - O măsură a gradului de acceptare de către celelalte furnici, M^+
- Structura algoritmului AntClust
  - Faza de învățare a pragului de similaritate
  - Faza întâlnirilor
  - Faza de rafinare a clusterilor

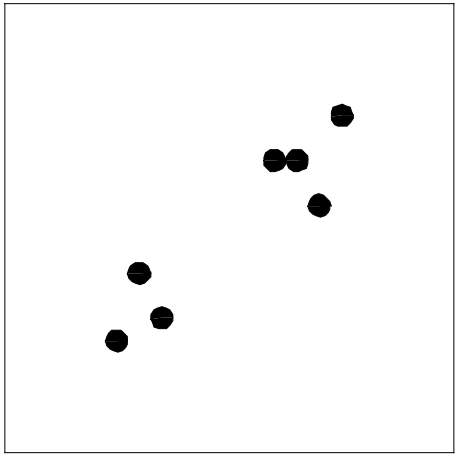
Modelul coloniei de furnici

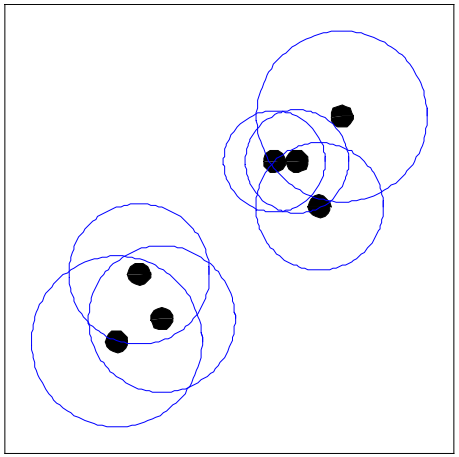
- Faza de învățare a pragului:
  - Pt fiecare furnică, pragul T se calculează pe baza similarității medii și a celei maxime cu celelalte date

$$T_i = \frac{\max_j(S(i,j)) + \text{avg}_j(S(i,j))}{2}$$

$$S(i,j) = \frac{1}{n}\sum_{k=1}^{n}\left(1 - \frac{|x_i^k - x_j^k|}{\max x^k - \min x^k}\right)$$

Date





Arii de similaritate

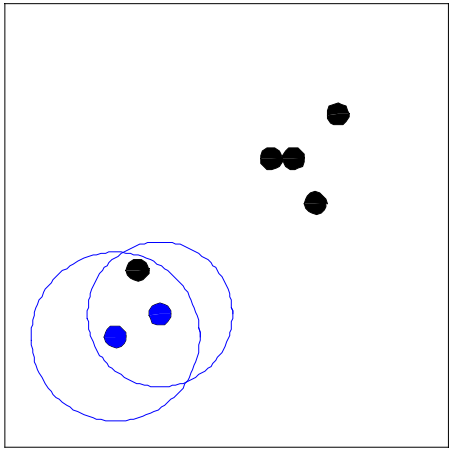
Modelul coloniei de furnici

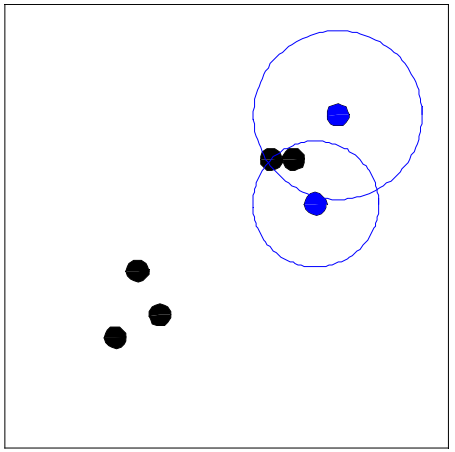
- Faza întâlnirilor aleatoare:
  - Se selectează aleator k_M perechi de furnici
  - Când furnica i întâlnește furnica j, se calculeaza similaritatea S(i,j) și se analizează:

If S(i,j)>Ti and S(i,j)>Tj
then furnicile se accepta reciproc
altfel se resping

  - Se aplică un set de reguli pe baza cărora se modifică eticheta furnicii și valorile mărimilor care exprimă percepția furnicii în privința cuibului din care face parte

Situatie de acceptare





Situatie de respingere

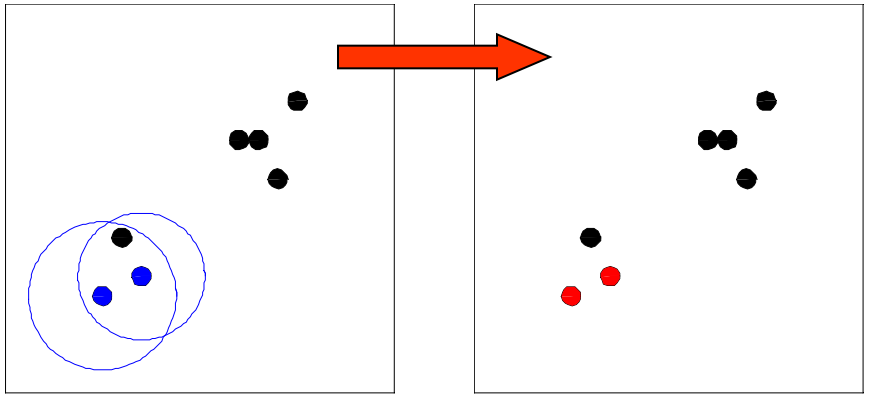
Modelul coloniei de furnici

- Reguli de acceptare:

Regula 1:

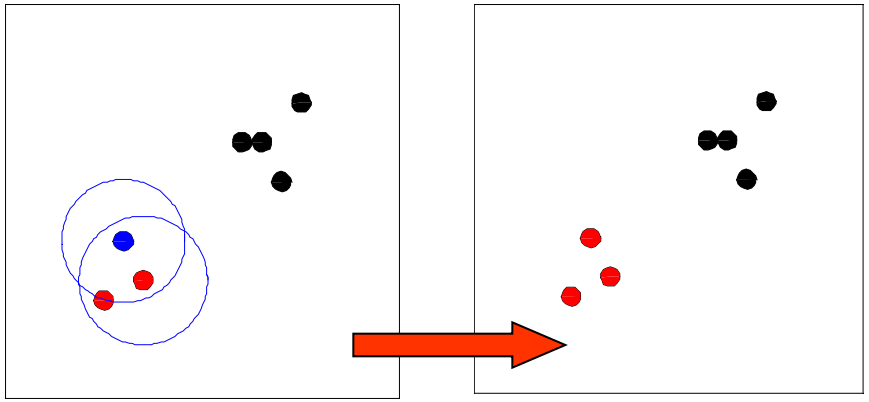
Daca se intalnesc doua furnici neetichetate ele vor forma un nou cuib

Regula 1



Regula 2:

Daca o furnica neetichetata intalneste una etichetata atunci este inclusa in acelasi cuib



Regula 2

Modelul coloniei de furnici

- Reguli de acceptare:

Regula 3:

La întâlnirea a două furnici din același cuib se incrementează parametrii M și M^+

Regula 5:

La întâlnirea a două furnici din cuiburi diferite furnica având M mai mic este atrasă în celălalt cuib iar parametrii M ai ambilor furnici sunt decrementați.

Incrementare

$$\text{inc}(v) = (1-\alpha)v + \alpha$$

Decrementare

$$\text{dec}(v) = (1-\alpha)v$$

$\alpha \in (0,1)$

parametru

M si M^+ apartin lui [0,1)

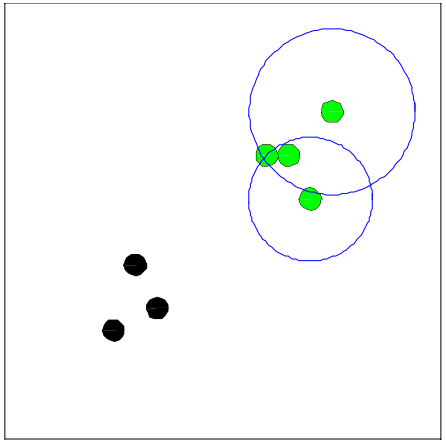
Modelul coloniei de furnici

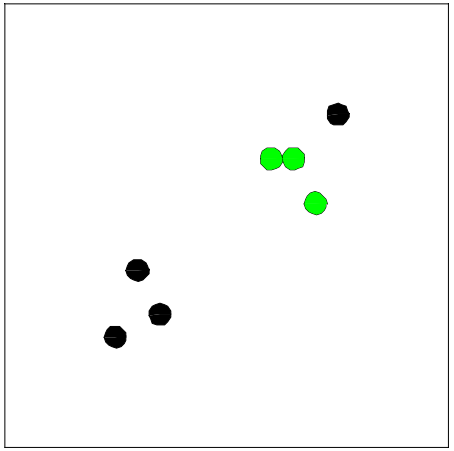
- Regula de respingere:

Regula 4:

Dacă se întâlnesc două furnici din același cuib care se resping atunci:

- Furnica cu valoare mai mica pentru M^+ este eliminată din cuib iar parametrii săi sunt resetați
- parametrul M al celeilalte furnici este mărit iar parametrul M^+ este micșorat





Modelul coloniei de furnici

- Structura algoritmului

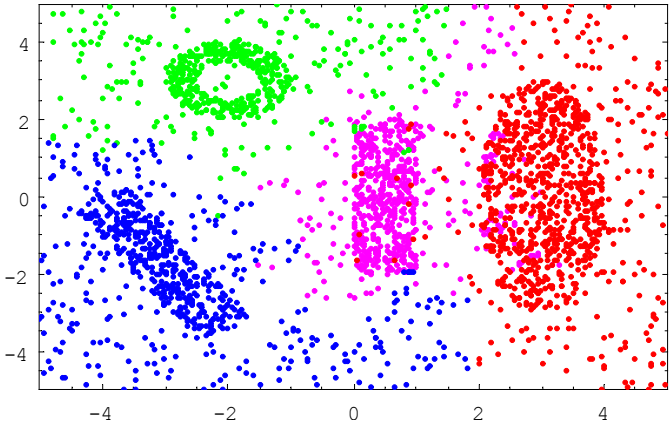
Algorithm 1 AntClust algorithm

1: **for all** $i \in \{1, \ldots, m\}$ **do**
2: $L_i := 0$; $A_i := 0$; $M_i := 0$; $M_i^+ := 0$;
3: **end for**
{Threshold learning:}
4: **for all** $i \in \{1, \ldots, m\}$ **do**
5: sample k_T ants and compute $max\{S(i,\cdot)\}$ and $\langle S(i,\cdot)\rangle$; $T_i := (max\{S(i,\cdot)\} + \langle S(i,\cdot)\rangle)/2$;
6: **end for**
{Random meetings:}
7: **for all** $k \in \{1, \ldots, k_M\}$ **do**
8: Select a random pair (i, j)
9: Increase the age: $A_i := A_i + 1$; $A_j := A_j + 1$;
10: Compute $S(i, j)$
11: Apply the rules R1-R5
12: **end for**
{Clusters refining:}
13: **for all** identified clusters **do**
14: compute $N_{cluster}$ (the ratio of data belonging to the cluster) and $\langle M^+\rangle$ (averaged value of M^+ for all ants in the cluster)
15: compute the acceptance probability $P_a = \alpha\langle M^+\rangle + (1-\alpha)N_{cluster}$
16: **if** $P_a < \theta_r$ **then**
17: remove the cluster (all its elements are reset)
18: **end if**
19: **end for**
20: **for all** ants having $M^+ < \theta_a$ **do**
21: assign the ant to the cluster of the most similar ant, j, which belongs to a nest ($L_j \neq 0$) and has a high enough acceptance degree ($M_j^+ > \theta_a$)
22: **end for**

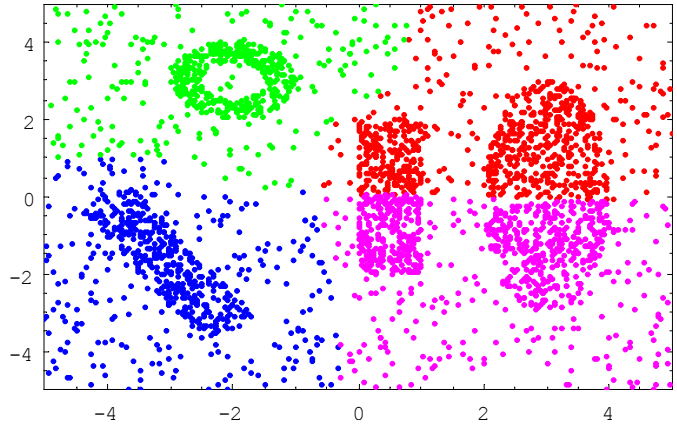
Modelul coloniei de furnici

- Exemplu

AntClust



KMeans



Modelul ansamblului de particule

- Tehnica Particle Swarm Optimization (PSO) a fost propusă de către James Kennedy şi Russell Eberhart pentru optimizarea funcţiilor neliniare (1995)

- Sursa de inspirație:

  - comportarea stolurilor de păsări, bancurilor de pești, roiurilor de albine
  - acestea sunt asimilate unui ansamblu de particule care se deplasează în spațiul de căutare pentru a identifica optimul

- Biblio: http://www.particleswarm.info/

Modelul ansamblului de particule

Idee:

- Se folosește un ansamblu de particule (elemente) a căror poziții sunt din domeniul funcției obiectiv și care sunt modificate printr-un proces iterativ
- La fiecare iterație se stabilește noua poziție a fiecărei particule în funcție de:
  - Poziția curentă a particulei
  - Cea mai bună pozitie întâlnită de către particulă (local best)
  - Cea mai buna poziție întâlnită de către ansamblu (global best)

Structura generală:

Inițializare poziții particule
Inițializare deplasamente inițiale
REPEAT
  calcul deplasamente (viteze)
  actualizare poziții
  evaluare
  actualizare lista cu cele mai bune elemente
UNTIL <condiție de oprire>

Ilustrare: http://www.projectcomputing.com/resources/psovis/index.html

Obs: nu se folosește selecție ci doar mutație direcționată (înspre cele mai bune elemente ale populației

Modelul ansamblului de particule

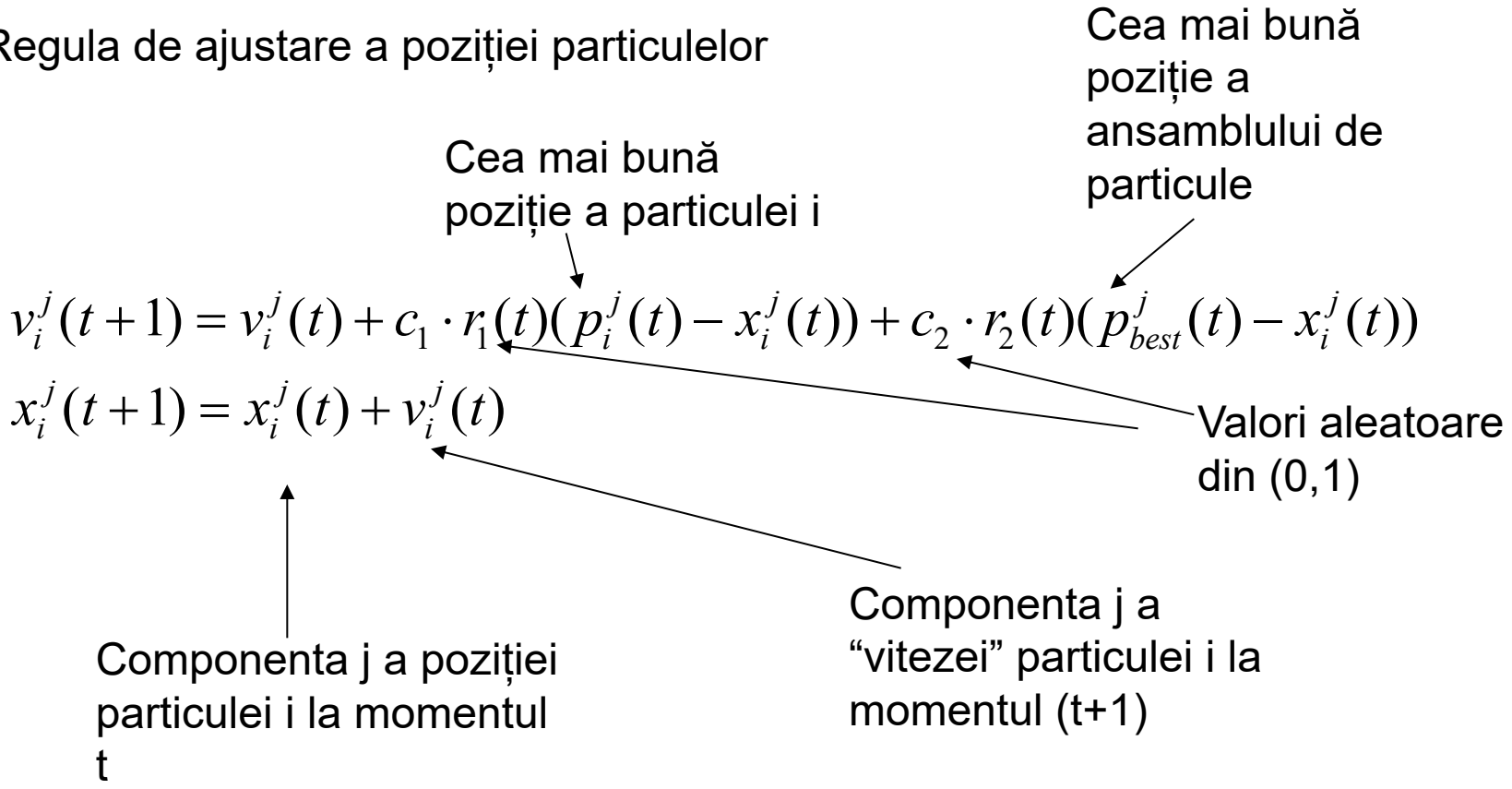
- Regula de ajustare a poziției particulelor se bazează pe două etape:
  - Ajustare deplasamente (viteze)
  - Aplicarea deplasamentelor

$$v_i^j(t+1) = v_i^j(t) + c \cdot r_1(t)(p_i^j(t) - x_i^j(t)) + c \cdot r_1(t)(p_{best}^j(t) - x_i^j(t))$$

$$x_i^j(t+1) = x_i^j(t) + v_i^j(t)$$

Modelul ansamblului de particule

- Regula de ajustare a poziției particulelor



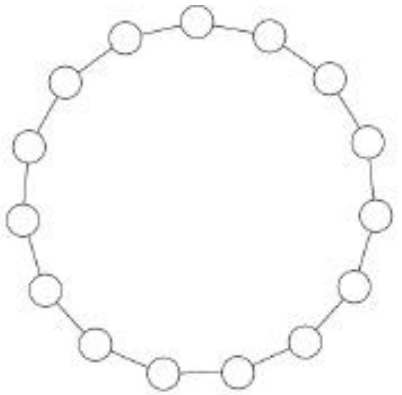
Modelul ansamblului de particule

- Variante
  - Introducerea unui factor de inerție (b) și a unui factor constrictiv pentru a limita creșterea vitezei (a)

$$v_i^j(t+1) = a \cdot (b \cdot v_i^j(t) + c_1 \cdot r_1(t)(p_i^j(t) - x_i^j(t)) + c_2 \cdot r_2(t)(p_{best}^j(t) - x_i^j(t)))$$

$$x_i^j(t+1) = x_i^j(t) + v_i^j(t)$$

  - Utilizarea vecinătăților pentru calculul celui m bun element (p_{best} se determină luând în considerare doar vecinii lui i).
    - Exemplu de topologie: circulară



Modelul roiului de albine

- Artificial Bee Colony (ABC) [Karaboga, 2005]
  http://mf.erciyes.edu.tr/abc/links.htm

- Sursa de inspirație: comportamentul inteligent al albinelor în procesul de identificare a surselor de hrană (nectar)

- Utilizează o populație de “albine” constituită din trei categorii:

  - Albine “alocate” unei surse de hrană (lucrătoare)
  - Albine observatoare
  - Albine cercetașe

Modelul roiului de albine

- Albine “lucratoare” (employed foragers)
  - Sunt asociate unei surse de hrană (miere) pe care o exploatează
  - Posedă informație privind calitatea sursei de hrană (pe care o transmit și unora dintre albinele observator)
- Albine “observator” (onlookers):
  - Colectează informații de la albinele lucrătoare și după ce identifică o sursă de hrană devin albine lucrătoare
- Albine “cercetaș” (scouters)
  - Explorează în mod aleator spațiul de căutare pentru a identifica noi surse de hrană

Modelul roiului de albine

- Pas 1: Se inițializează aleator locațiile din spațiul de căutare unde sunt plasate albinele lucrătoare
- Pas 2: Cât timp e satisfacută condiția de continuare se execută:
  - Albinele lucrătoare transmit informații privind calitatea locației în care se află către albinele observator; fiecare albină observator selectează o locație; selecția se bazează pe o distribuție de probabilitate determinată de valorile scorurilor asociate;
  - Albinele lucrătoare explorează vecinătatea locației în care se află și se mută într-o altă locație vecină dacă aceasta este mai bună; dacă o albină lucrătoare nu descoperă într-un număr limită de pași o configurație mai bună atunci ea este relocată într-o poziție determinată de o albină cercetaș
  - Albinele cercetaș își schimbă aleator poziția

Modelul roiului de albine

Detalii:

- Notații: NB = număr de albine lucrătoare, NO = număr de albine observator,
  f = funcția scor, n = dimensiunea problemei
- Alegerea noii locații de către o albină observator se face prin selecție proporțională folosind distribuția de probabilitate

$$P(x_i) = \frac{f(x_i)}{\sum_{j=1}^{NB} f(x_j)}$$

- Modificarea pozitiei unei albine lucrătoare i se bazează pe:

$$v_i^j = x_i^j(t) + \phi_{ij}(x_i^j(t) - x_k^j(t)), \quad j = \overline{1,n}$$

unde k este indicele unei albine lucrătoare aleasă aleator, ϕ_{ij} este un parametru aleator in [-1,1]

Modelul roiului de albine

Detalii:

- Dacă configurația v_i este mai bună decât $x_i(t)$ atunci $x_i(t+1)$ va fi v_i, altfel rămâne $x_i(t)$

Observație. Intr-un algoritm ABC există mai multe tipuri de selecție:

- Selecție globală (bazată pe distribuția de probabilitate definită pe slide-ul anterior) folosită de catre albinele observator pentru a identifica regiuni promițătoare
- Selecție locală (bazată pe calculul și analiza unei configurații "vecine" v_i) realizată atât de albinele lucrătoare cât și de către albinele observator
- Selecție aleatoare realizată de către albinele cercetaș care sunt relocate în poziții stabilite aleator

Modelul licuricilor

Firefly algorithm (Yang, 2008)

Sursa de inspirație: interacțiunile dintre licurici bazate pe semnalele luminoase pe care le emit

Idee principală de implementare

- Fiecare element al populației corespunde poziției unui licurici
- Fiecărui licurici îi este asociat un grad de luminozitate (corelat cu valoarea funcției obiectiv asociate elementului corespunzător din populație)
- Deplasarea licuricilor este ghidata atât de distanța dintre pozițiile lor cât și de valoarea luminozității
- Poziția x_i este deplasată către poziția x_j (dacă x_j are luminozitatea mai mare) folosind relația de mai jos (alpha, beta și gamma sunt parametri de control iar epsilon este o valoare aleatoare cu distribuție normală)

$$x_i(t+1) = x_i(t) + \beta \exp(-\gamma d^2(x_i(t), x_j(t)))(x_j(t) - x_i(t)) + \alpha \varepsilon_i(t)$$

Alte metode inspirate de natură

- Bat algorithm [X.S. Yang, 2010]
- Cuckoo search [X.S. Yang, 2009]
- Cat swarm algorithm [S.C. Chu, 2006]
- Biogeography optimization [D. Simon, 2008, http://embeddedlab.csuohio.edu/BBO/]
- Krill herd optimization [Gandomi, 2012]
- Monkey search [A. Mucherino, 2007]
- Fruit-fly optimization algorithm [Pan, 2011]

....

Observație – proliferarea metodelor bio-inspirate nu este neapărat benefică pentru dezvoltarea domeniului întrucât o parte dintre metode se bazează pe operatori similari, independent de metafora biologică

Sorensen, 2012: Metaheuristics: the Metaphor Exposed
Michalewicz, 2012: Quo-Vadis Evolutionary Computing

In loc de concluzii