BABEŞ-BOLYAI UNIVERSITY
Faculty of Mathematics and Computer Science

Inteligenţă Artificială

7: Swarm Intelligence

Camelia Chira

cchira@cs.ubbcluj.ro

Algoritmi inspirati de natura

- Algoritmii evolutivi simulează evoluţia
- Algoritmii inspiraţi de comportamentul de grup simulează adaptarea colectivă si procesele sociale dintr-un colectiv

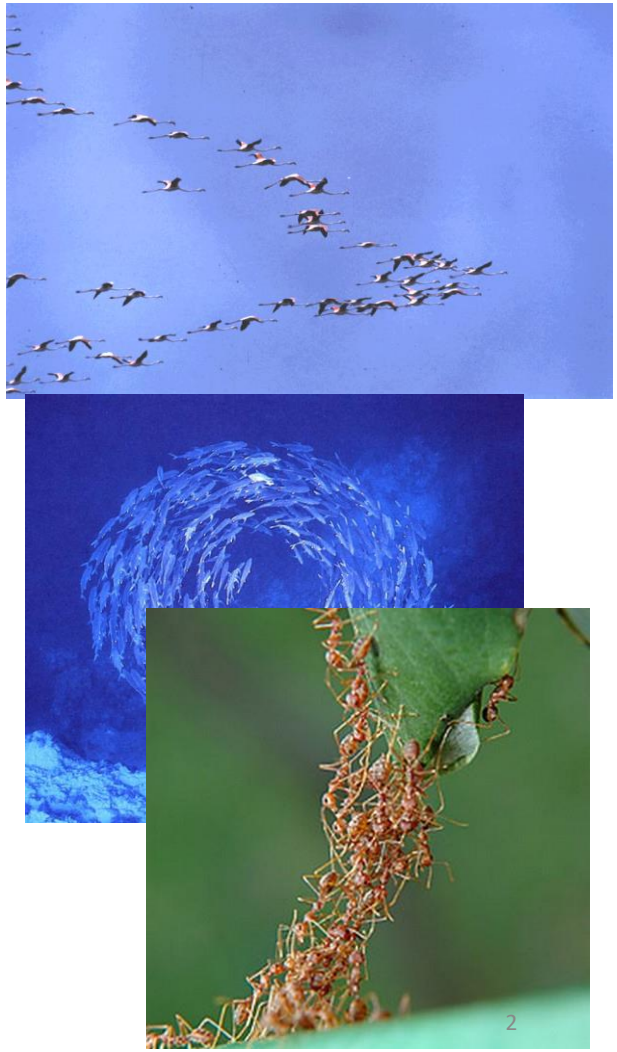
  - **Particle Swarm Optimisation (PSO)**

    http://www.youtube.com/watch?feature=endscreen&v=JhZKc1Mgub8&NR=1

    https://www.youtube.com/watch?v=TWqx57CR69c

  - **Ant Colony Optimisation (ACO)**

    http://www.youtube.com/watch?v=jrW_TTxP1ow



Bibliografie

- Eric Bonabeau, Marco Dorigo, Guy Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems,* Santa Fe Institute, Studies in the Science of Complexity, Oxford University Press, 1999.
- Camazine S., Deneubourg J.-L., Franks N. R., Sneyd J., Theraulaz G., Bonabeau E., *Self-Organization in Biological Systems*. Princeton Studies in Complexity, Princeton University Press, 2001
- James Kennedy, Russel Eberhart, *Particle Swarm Optimisation*, Proceedings of IEEE International Conference on Neural Networks. IV. pp. 1942–1948, 1995.
- Marco Dorigo, Christian Blum, *Ant colony optimization theory: A survey*, Theoretical Computer Science 344 (2005) 243 – 27.

Swarm Intelligence
Inteligenta de grup

- Fenomene colective naturale care implica interconexiuni stranse intre indivizi

- **Swarm intelligence (SI)** describes the collective behavior of decentralized, self-organized systems, natural or artificial
- *"any attempt to design algorithms or distributed problem-solving devices inspired by the collective behavior of social insect colonies and other animal societies"* [Bonabeau, Dorigo, Theraulaz,1999]

- Self-organization
- Stigmergy

Sursa de inspiratie

- Comportament biologic real
  - Insecte sociale – furnici, termite, albine, viespi
  - Formarea de roi, turma, stol

- Swarming (~gruparea)
  - Agregarea animalelor similare – in general a celor care au aceeasi directie
    - Termitele *swarm* pentru a construi colonii
    - Pasarile *swarm* pentru a gasi hrana
    - Albinele *swarm* in scopul reproducerii
  - De ce?
    - Pentru a cauta mai bine
    - Pentru a migra
    - Ca si mecanism de aparare impotriva pradatorilor

Caracteristici comune - swarming

- Reguli simple pentru fiecare individ
- Nu exista control centralizat
  - Controlul este complet distribuit intre membrii unui grup
  - Decentralizat, deci robust
- Comunicarea intre indivizi
  - La nivel local
  - Indirecta (stigmergy)
- Comportament global
  - Emergent
  - Transcede comportamentul individual
  - Se adapteaza usor schimbarilor din mediu

Swarm Intelligence (SI): Scurt Istoric

- Beni and Wang (1989):
  - Au introdus termenul in contextul automatelor celulare
  - Control decentralizat, auto-organizare, simplitate la nivel individual

- Bonabeau, Dorigo si Theraulaz (1999)
  - *"any attempt to design algorithms or distributed problem-solving devices inspired by the collective behavior of social insect colonies and other animal societies"*

- Beni (2004)
  - Intelligent swarm = a group of non-intelligent robots

Algoritmi inspirati de natura

- Grup (roi - Swarm)
  - O colecţie aparent dezorganizată de indivizi care se mişcă tinzând să se grupeze, dar fiecare individ pare să se mişte într-o direcţie oarecare
  - În interiorul colecţiei apar anumite procese sociale
  - Colecţia este capabilă să efectueze sarcini complexe
    - fără nici o ghidare sau control extern
    - fără nici o coordonare centrală
  - Colecţia poate atinge performanţe care nu pot fi atinse de indivizi în izolare
- Adaptare colectivă - auto-organizare
- Stigmergy

Auto-organizare

- *Self-organization consists of set of dynamical mechanisms whereby structure appears at the global level as the result of interactions among lower-level components. The rules specifying the interactions among the system's constituent units are executed on the basis of purely local information, without reference to the global pattern, which is an emergent property of the system rather than a property imposed upon the system by an external ordering influence [Bonabeau et al., 1997]*

- Un mecanism dinamic prin care o structura globala emerge din interactiuni locale

- Interactiuni multiple

- Caracter aleator

- Feedback pozitiv / negativ

Stigmergy

- **P.P. Grassé**, 1959

La coordination des taches, la regulation des constructions ne dependent pas directement des oeuvriers, mais des constructions elles-memes. ***L'ouvrier ne dirige pas son travail, il est guidé par lui.*** *C'est à cette stimulation d'un type particulier que nous donnons* le nom du **STIGMERGIE**

Derivat din cuvintele grecesti *stigma* (mark, sign) si *ergon* (work, action)

- Comunicare indirecta- doi indivizi comunica astfel:
  - Un individ modifica mediul la un moment dat
  - Celalalt individ raspunde la mediul modificat la un moment de timp ulterior

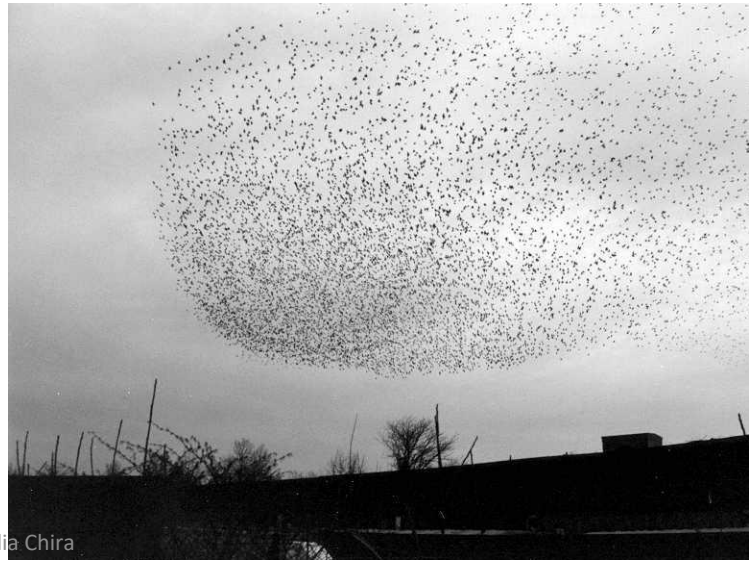
Algoritmi SI

- **Particle Swarm Optimization (PSO)**
  - Bio: Formarea de grupuri in stol, turme
  - Comunicare tip broadcast

- **Ant Colony Optimization (ACO)**
  - Bio: Comportamentul real al coloniilor de furnici
  - Stigmergy

Particle Swarm Optimization (PSO)

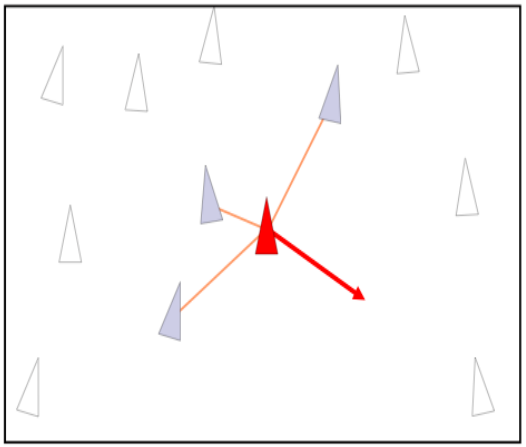
- Algoritm propus de Kennedy si Eberhart in 1995
  - http://www.particleswarm.info/
- Publicatii
  - Kennedy, J.; Eberhart, R.C. (2001). *Swarm Intelligence*. Morgan Kaufmann.
  - Poli, R. (2008). "Analysis of the publications on the applications of particle swarm optimisation". *Journal of Artificial Evolution and Applications*: 1-10
- Probleme de optimizare
- Exemplu PSO
  - Gruparea pasarilor in stoluri



PSO - exemplu in natura: bird flocking

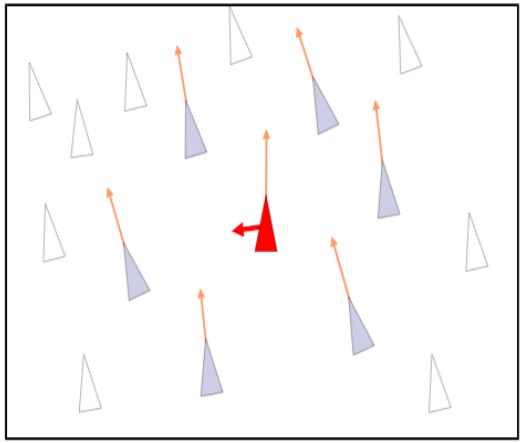
- Modelul ‘Boids’ (=bird-oids=bird-like) propus de Reynolds

Regula 1:
EVITAREA COLIZIUNILOR



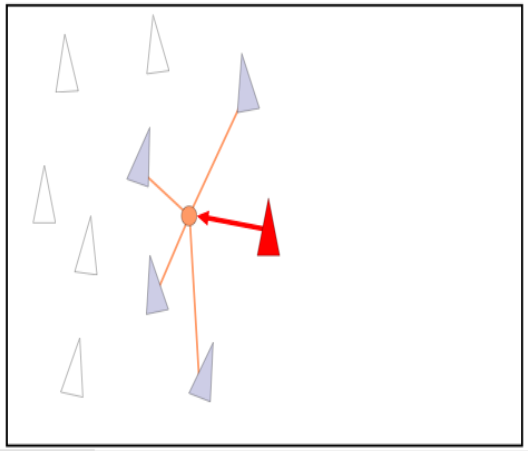
…cu pasarile din vecinatate

Regula 2:
AJUSTAREA VITEZEI SI DIRECTIEI



…dupa cea a vecinilor

Regula 3:
ATRACTIA SPRE CENTRUL GRUPULUI



…prin atractia spre pasarile vecine

Termeni PSO

- Populatie
  - Constituita din m particule care cauta solutia optima
  - Cooperare
- Particula
  - Corespunde unei posibile solutii
  - Se misca si are o viteza
  - Retine locul (pozitia) unde a obtinut cele mai bune rezultate
  - Are asociata o vecinatate de particule
- Particulele cooperează
  - Schimbă informaţii (legate de descoperirile făcute în locurile deja vizitate) între ele
  - Fiecare particulă ştie fitnessul vecinilor ei a.î. poate folosi poziţia celui mai bun vecin pentru a-şi ajusta propria viteză

Tehnica PSO

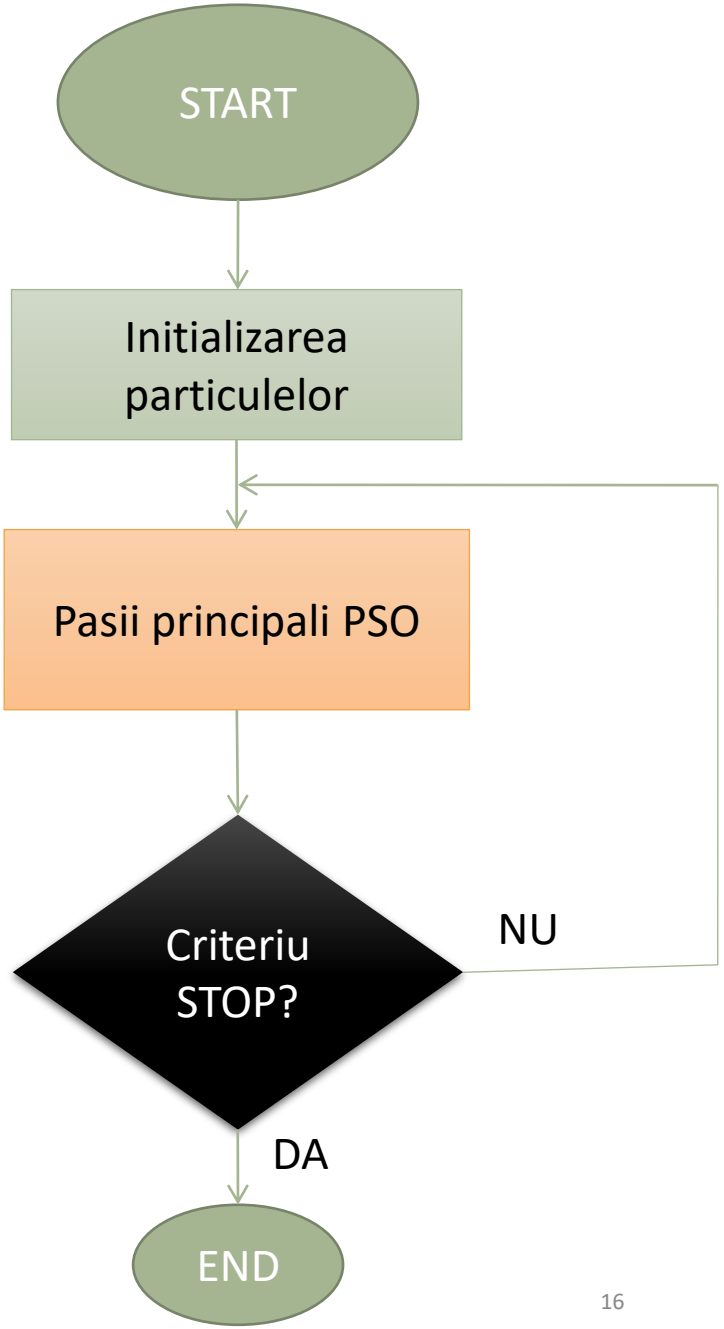
- Fiecare particula
  - Evalueaza functia de optimizat in fiecare punct din spatiu in care ajunge
  - Retine cea mai buna valoare gasita– pbest
- Cea mai buna pozitie globala gasita de unul din membrii grupului – gbest
  - Informatie accesibila tuturor particulelor
- Cautare cooperativa – ghidata de calitatea relative a indivizilor

PSO

Crearea populaţiei iniţiale de particule
- Poziţii aleatoare
- Viteze nule/aleatoare

- **Evaluarea particulelor**
- **Pentru fiecare particulă :**
  - Actualizarea memoriei
    - Stabilirea celei mai bune particule din swarm **gbest** / dintre particulele vecine **ibest**
    - Stabilirea celei mai bune poziţii (cu cel mai bun fitness) în care a ajuns până atunci – **pbest**
  - Modificarea vitezei
  - Modificarea poziţiei

Dacă nu se îndeplinesc condiţiile de oprire, se revine la pasul 2, altfel STOP



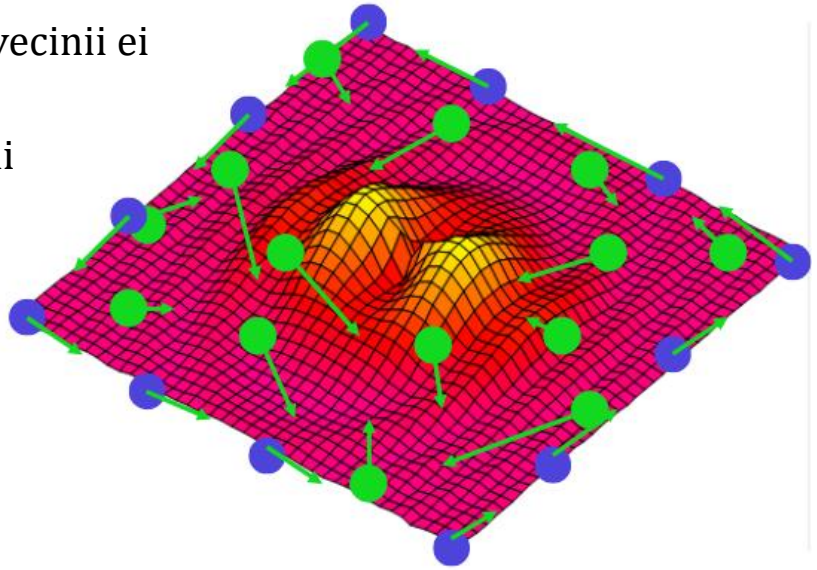
Algoritm PSO

1. Crearea populaţiei iniţiale de particule

- Fiecare particulă are asociată
  - o poziţie – potenţială soluţie a problemei
  - o viteză – modifică o poziţie în altă poziţie
  - o funcţie de calitate (fitness)

- Fiecare particulă trebuie să poată:
  - interacţiona (schimba informaţii) cu vecinii ei
  - memora o poziţie precedentă
  - utiliza informaţiile pentru a lua decizii

- Iniţializarea particulelor
  - poziţii aleatoare
  - viteze nule/aleatoare

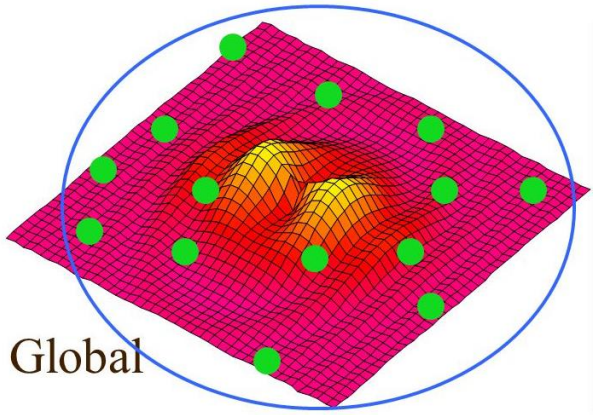


Algoritm PSO

2. Evaluarea particulelor

- Depinde de problema
- Functia de fitness

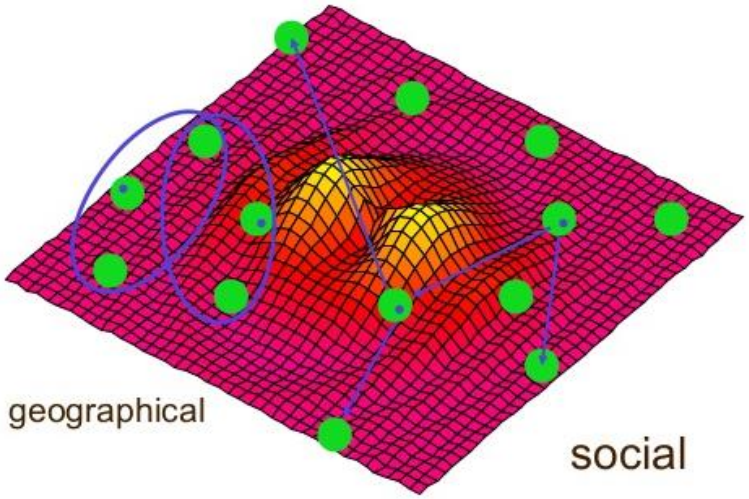
Algoritm PSO

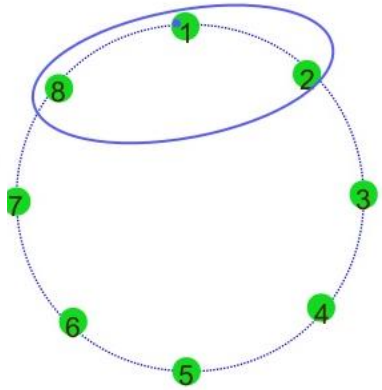


3. **Pentru fiecare particula x**
   - Actualizarea memoriei
     - Stabilirea celei mai bune particule din swarm **gbest** / dintre particulele vecine **ibest**

Vecinatatea unei particule:
- Globala
- Locala

Tipul vecinatatii:
- Geografica
- Sociala
- Circulara





circular

Algoritm PSO

3. **Pentru fiecare particula x**
   - Actualizarea memoriei
     - Stabilirea celei mai bune particule din swarm **gbest** /

     dintre particulele vecine **ibest**

     - Stabilirea celei mai bune poziţii (cu cel mai bun fitness) în care a ajuns până atunci – **pbest**

Algoritm PSO

v – viteza particulei
x – particula curenta (solutia)
rand() – numar aleator intre 0 si 1
w = factor de inertie
c_1 = factor de invatare cognitiv
c_2 = factor de invatare social
w, c_1, c_2 (ponderi pozitive)

3. **Pentru fiecare particula x**
   - Actualizarea memoriei
   - Modificarea vitezei v

Eq. (a):

```
v[] = w * v[] +
      c1 * rand() * (pbest[] - x[]) +
      c2 * rand() * (gbest[] - x[])
```

   - Modificarea poziţiei

Eq. (b):

```
x[] = x[] + v[]
```

Pseudocod PSO

For each particle
  Initialize particle
END

Do
  For each particle
    Calculate fitness value
    If the fitness value is better than (pBest) then pBest= current value
  End

  gBest = the particle with the best fitness value of all the particles
  For each particle
    Calculate particle velocity according equation (a)
    Update particle position according equation (b)
  End
While maximum iterations or minimum error criteria is not attained

Modificarea vitezei si pozitiei

3. **Pentru fiecare particula $x_i = x_{i1}, ..., x_{iD}$**
   - $\mathbf{x_{ij}}$ = pozitia particulei i in dimensiunea j, j=1...D
   - $\mathbf{v_{ij}}$ = viteza particulei i in dimensiunea j, j=1..D
   - Modificarea vitezei pe fiecare dimensiune j

Eq. (a):

$$v_{ij} = w * v_{ij} + c1 * rand() * (pbest_j - x_{ij}) + c2 * rand() * (gbest_j - x_{ij})$$

OBS: Viteza este restricţionată de o valoare maxima presetata Vmax.

- Modificarea poziţiei pe fiecare dimensiune j

Eq. (b):

$$x_{ij} = x_{ij} + v_{ij}$$

i= 1,...,N (numarul de particule)

j= 1,...,D (numarul de dimensiuni)

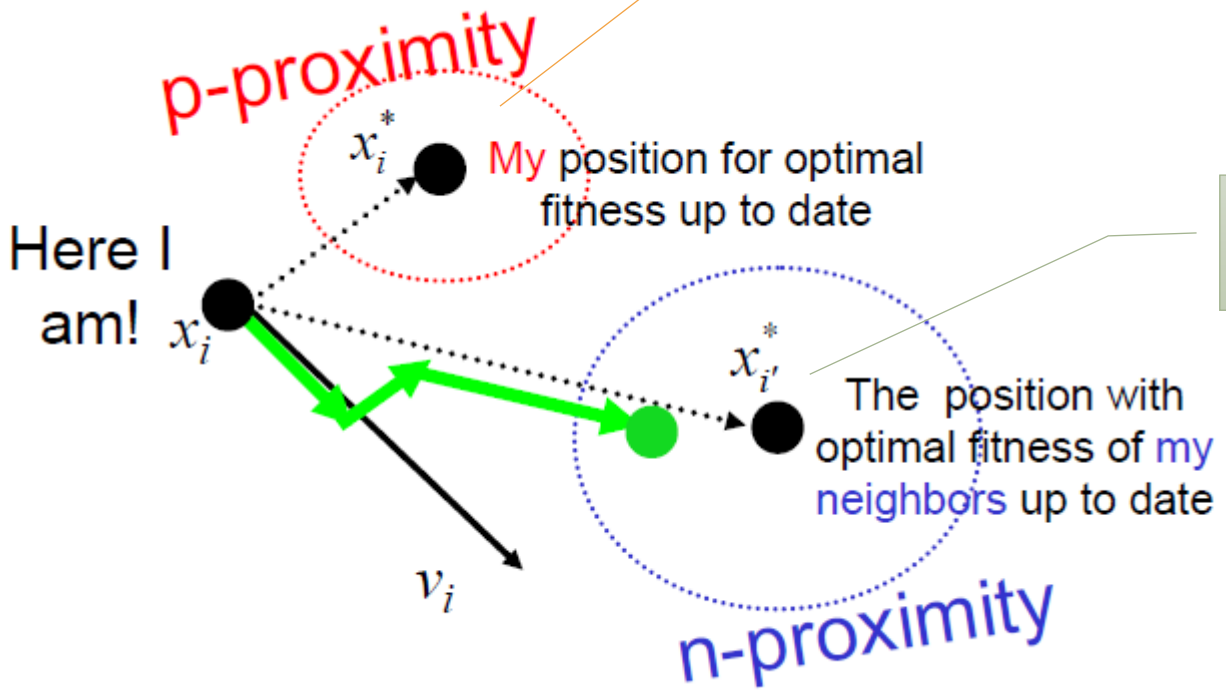
Parametrii PSO

- **Numarul de particule (de obicei 10-50)**
- **w : $w * v_{ij}$ – termen inerţial**
  - forţează particula să se deplaseze în aceeaşi direcţie ca şi până acum
  - balansează căutarea între explorare globală (w mare) şi locală (w mic)
  - poate fi constantă sau descrescătoare (pe măsura „îmbătrânirii" grupului)
- **c1: $c1 * rand() * (pbest_j - x_{ij})$ – termen cognitiv**
  - forţează particula să se deplaseze spre cea mai bună poziţie atinsă până atunci (tendinţă de conservare)
- **c2 : $c2 * rand() * (gbest_j - x_{ij})$ – termen social**
  - forţează particula să se deplaseze spre cea mai bună poziţie a vecinilor; spirit de turmă, de urmăritor
- Cei doi factori c1 şi c2 pot fi egali (de obicei c1=c2=2) sau diferiţi (de obicei c1 > c2 şi c1+c2=4 sau c1+c2 < 4)
- Exemplu de setare: w=1, c1=2, c2=2

Miscarea particulelor

Componenta cognitiva

PBEST



GBEST

Componenta sociala

A. Martinoli, 2006

Algoritmul PSO

Initializare populatie **N** particule (v, x)

Repeat

  for i = 1 to N **do**

    if $f(x_i) < f(x^{pbest}_i)$ **then** $x^{pbest}_i = x_i$

    $x^{gbest} = best(x^{pbest})$

    for j = 1 to D **do**

      Calcul viteza v_{ij}

      Calcul pozitie x_{ij}

    end

  end

until STOP criterion met

Exemplu aplicare

$$f(x) = {x_1}^2 + {x_2}^2 + {x_3}^2$$

- O particula este x =(x_1, x_2, x_3)
- Functia de fitness este f(x)
- Numarul de particule = 10
- Intervalul in care ia valori fiecare x (setat de problema): [-10,10]
- Vmax = 20
- w=1
- c1 = c2 = 2
- Conditia de oprire: numarul maxim de iteratii 2000

PSO vs AE

- **Diferente PSO fata de AE**
  - Nu există un operator de recombinare directă – schimbul de informaţie are loc în funcţie de experienţa particulei şi în funcţie de cea a celui mai bun vecin şi nu în funcţie de părinţii selectaţi pe baza fitness-ului
  - Update poziţie ~ similar cu mutaţia
  - Nu se foloseşte selecţia – supravieţuirea nu este legată de fitness
- Versiuni ale algoritmului de tip PSO
  - PSO binar discret
  - PSO cu mai mulţi termeni de învăţare socială
  - PSO cu particule eterogene
  - PSO ierarhic

PSO discret (binar)

- Versiune a PSO pentru spaţiu de căutare discret
- Poziţia unei particule
  - Potenţială soluţie a problemei -> string binar
  - Se modifică în funcţie de viteza particulei
- Viteza unei particule
  - element din spaţiu continuu e.g. valoare in [0,1]
  - se modifică conform principiilor de la PSO standard
  - se interpretează ca probabilitatea de modificare a bitului corespunzator din poziţia particulei

$$x_{ij} = \begin{cases} 1, daca\ rand < s(v_{ij}) \\ 0, altfel \end{cases}, unde\ s(v_{ij}) = \frac{1}{1 + e^{-v_{ij}}}$$

Remarci PSO

- Pericole
  - Particulele tind să se grupeze în acelaşi loc
  - Converg prea repede şi nu reuşesc să evadeze dintr-un optim local
  - Soluţia: Reiniţializarea unor particule
- Analiza algoritmilor de tip PSO
  - Indicele de dispersie
    - Măsoară gradul de împrăştiere a particulelor în jurul celei mai bune particule din grup
    - Media distanţelor absolute (pe fiecare dimensiune) între fiecare particulă şi particula cea mai bună
    - Explică gradul de acoperire (întins sau restrâns) a spaţiului de căutare
  - Indicele vitezei
    - Măsoară viteza de mişcare a grupului într-o iteraţie
    - Media vitezelor absolute
    - Explică cum (agresiv sau lent) se mişcă grupul

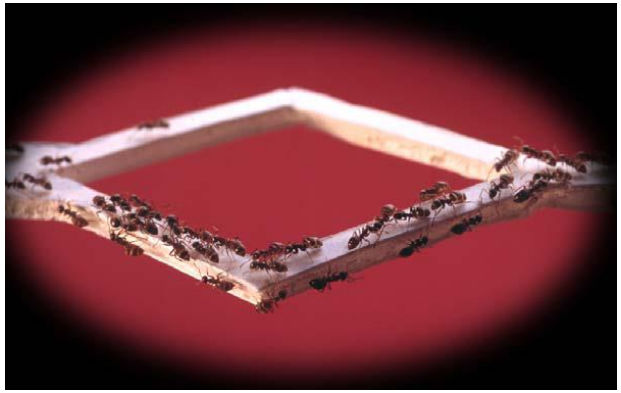
Aplicatii PSO

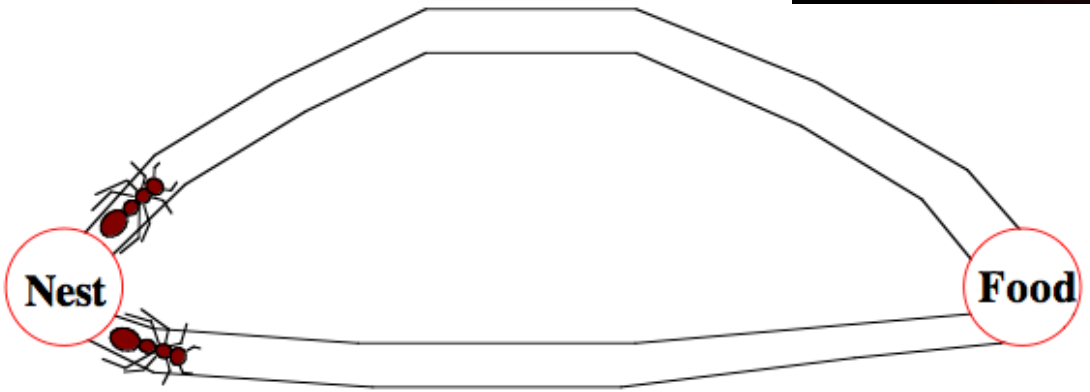
- Controlul şi proiectarea antenelor
- Aplicaţii biologice, medicale, farmaceutice
  - Analiza tremurului în boala Parkinson
  - Clasificarea cancerului
  - Predicţia structurii proteinelor
- Comunicare în reţele
- Optimizare combinatorială
- Optimizări financiare
- Analiza imaginilor şi analiza video
- Robotică
- Planificare
- Securitatea reţelelor, detecţia intruşilor, criptografie, criptanaliză
- Procesarea semnalelor

Ant Colony Optimization (ACO)

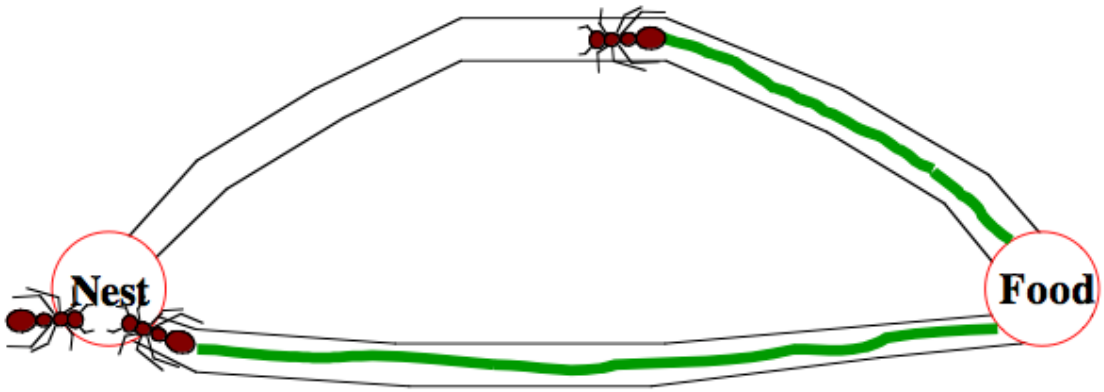
- **Marco Dorigo**, 1991 (Ant System),1995,1999 (ACO),…
  - Dorigo M., Maniezzo V., and Colorni A., "The Ant System: Optimization by a Colony of Cooperating Agents". *IEEE Trans. Systems, Man, and Cybernetics, Vol. 26, pp. 29-41, 1996.*
  - M. Dorigo and T. Stuetzle, "Ant Colony Optimization", MIT Press, 2004
  - M.Dorigo, M.Birattari, T.Stutzle, Ant colony optimization – Artificial Ants as a computational intelligence technique, IEEE Computational Intelligence Magazine 2006
- http://iridia.ulb.ac.be/~mdorigo/ACO/about.html
- Inspirata de comportamentul real al furnicilor
  - Furnicile gasesc cel mai scurt drum intre casa si sursa de hrana
  - Furnicile depun feromon pe drumul urmat – ceea ce ghideaza alte furnici in alegerea drumului (Stigmergy)
- Comportamentul real al furnicilor formalizat intr-o **metaeuristica**

Sursa de inspiratie…



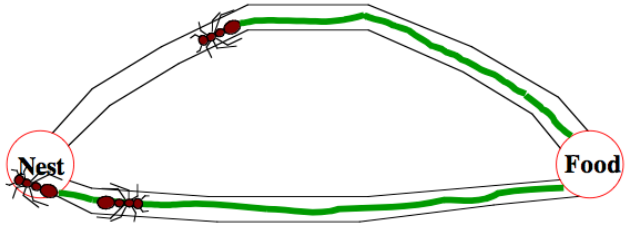


2 furnici pornesc cu probabilitate egala pe unul din cele 2 drumuri

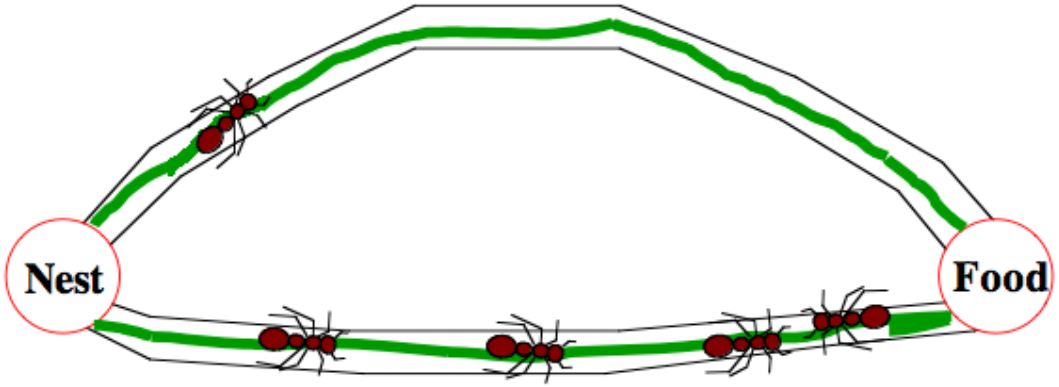


Furnica care merge pe drumul mai scurt ajunge mai repede inapoi la casa

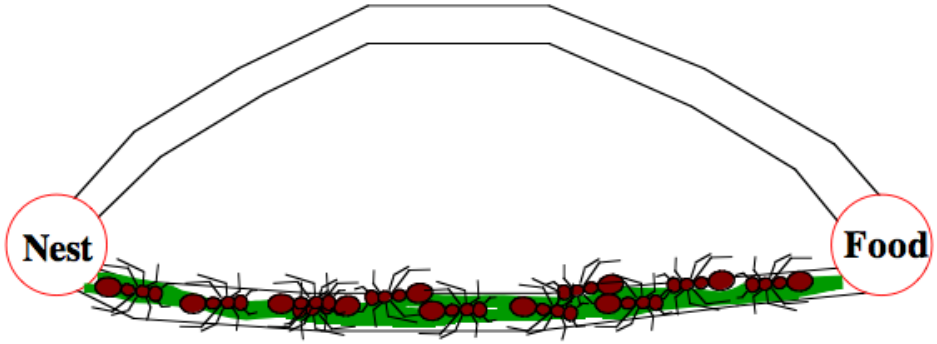
Densitate mai mare de feromon pe drumul mai scurt



Urmatoarea furnica alege drumul mai scurt

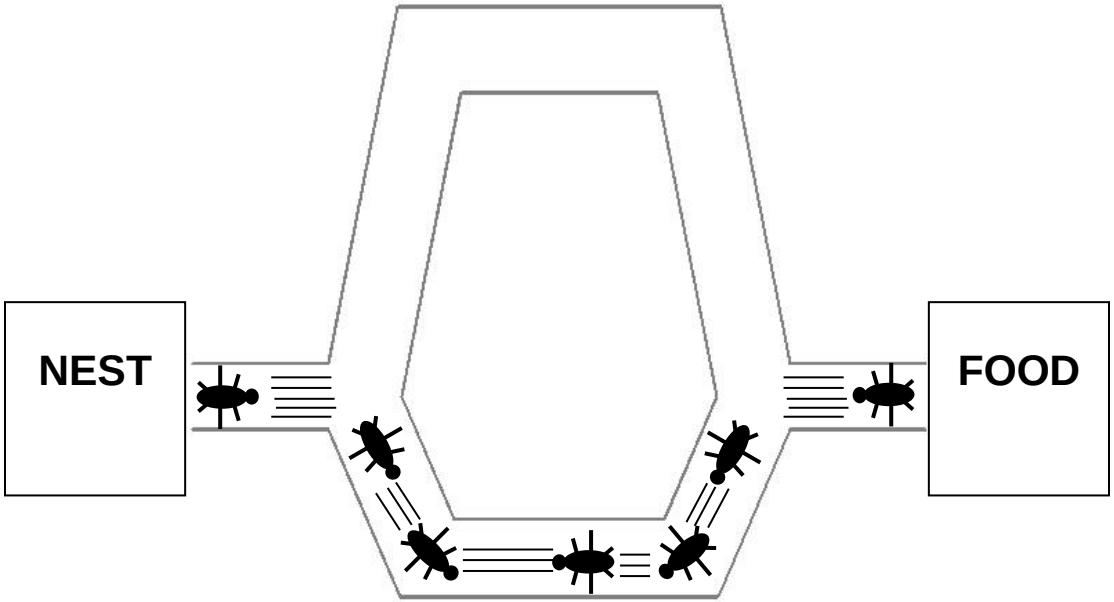


Dupa mai multe iteratii – tot mai multe furnici aleg drumul cu o cantitate mai mare de feromon - reinforcement



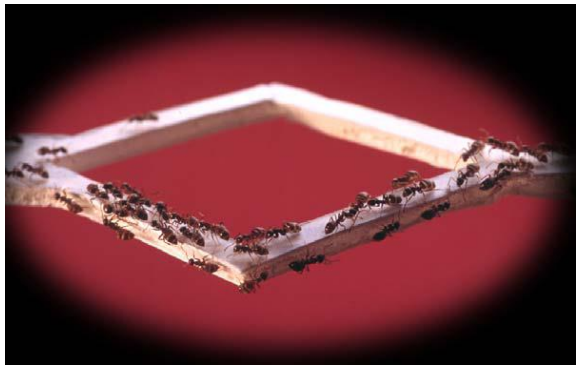
Dupa un timp – cel mai scurt drum exclusiv folosit de furnici

Comunicare indirecta prin feromon - stigmergy

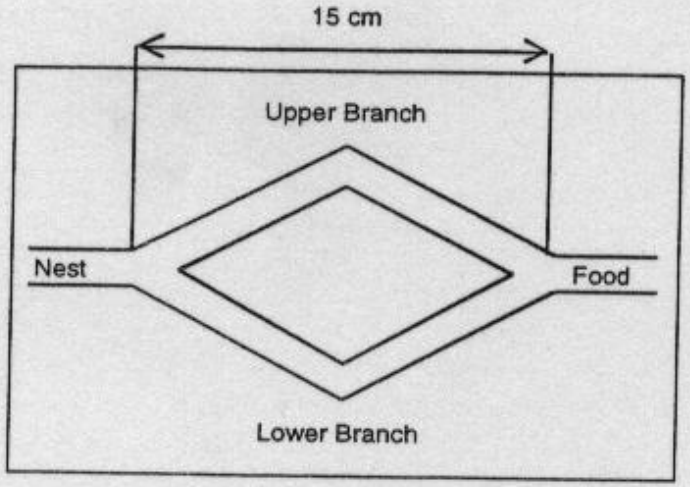


Cu cat mai multe furnici merg pe un drum cu atat acel drum devine mai atractiv

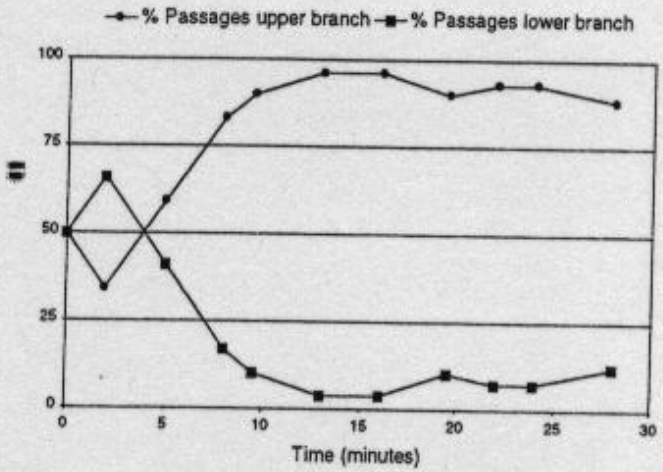
Convergenta



Chiar si atunci cand cele 2 drumuri au lungime egala unul din ele devine preferat dupa un timp (Deneubourg *et al*)



(a)



(b)

Figure 1. Single bridge experiment. (a) Experimental setup. (b) Results for a typical single trial, showing the percentage of passages on each of the two branches per unit of time as a function of time. Eventually, after an initial short transitory phase, the upper branch becomes the most used. After Deneubourg et al., 1990 [25].

ACO – aspecte teoretice

- Problema de optimizare – reprezentata ca un graf
  - Problema de optimizare trebuie transformată într-o problemă de identificare a drumului optim într-un graf orientat
- Solutie candidat ⇔ drumul construit de furnici artificiale
  - In fiecare iteratie, furncile contruiesc solutii partiale depozitand o cantitate de feromon pe muchia traversata
- Metodă de optimizare bazată pe:
  - Colonii de furnici (în loc de cromozomi) care caută soluţia optimă
  - Cooperare (în loc de competiţie ca în cazul AE)
- Fiecare furnică:
  - Se mişcă şi depune o cantitate de feromon pe drumul parcurs
  - Alege drumul pe care să-l urmeze în funcţie de:
    - Feromonul existent pe drum
    - Informaţia euristică asociată acelui drum
  - Cooperează cu celelalte furnici prin urma de feromon de pe drum care
    - depinde de calitatea soluţiei şi
    - se evaporă cu trecerea timpului

ACO - algoritm

Cât timp nu s-a ajuns la nr maxim de iteraţii

1. Iniţializare

2. Cât timp nu s-a parcurs numărul necesar de paşi pentru identificarea soluţiei
   - *Pentru fiecare furnică din colonie*
     - Se mărește soluția parțială cu un element (furnica execută o mutare)
     - Se modifică local urma de feromon corespunzător ultimului element adăugat în soluţie

3. Se modifică urma de feromon de pe drumurile parcurse de
   - Toate furnicile/cea mai bună furnică

4. Se returnează soluţia găsită de cea mai bună furnică

Ant System (AS) - Primul alg. ACO

Dorigo, Colorni, Maniezzo

- Furnicile sunt inzestrate cu o memorie in care se retin nodurile vizitate
- Solutiile sunt construite probabilistic fara a modifica feromonul in timpul constructiei unei solutii
- Furnicile depun o cantitate de feromon proportionala cu calitatea solutiei generate

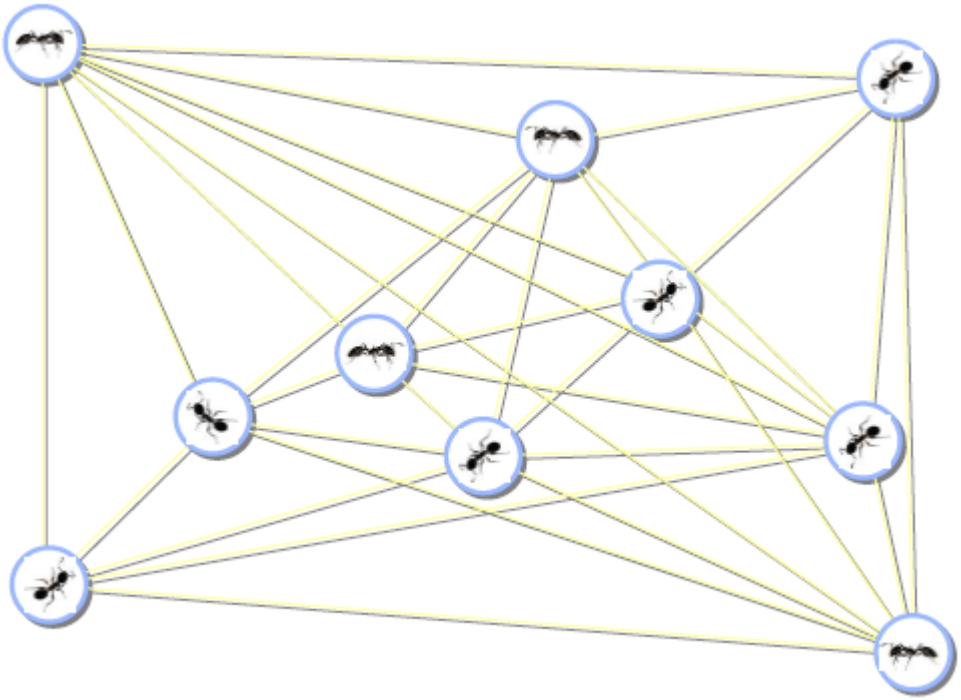
M. Dorigo, V. Maniezzo, and A. Colorni. Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 26(1):29-41, 1996.

AS pentru TSP

m – numarul total al furnicilor

La o iteratie *t* – un anumit numar de furnici $b_i(t)$ in fiecare nod i

$$\sum_{i=1}^{n} b_i(t) = m$$



Presupunere AS pentru TSP: **Graf total conectat**

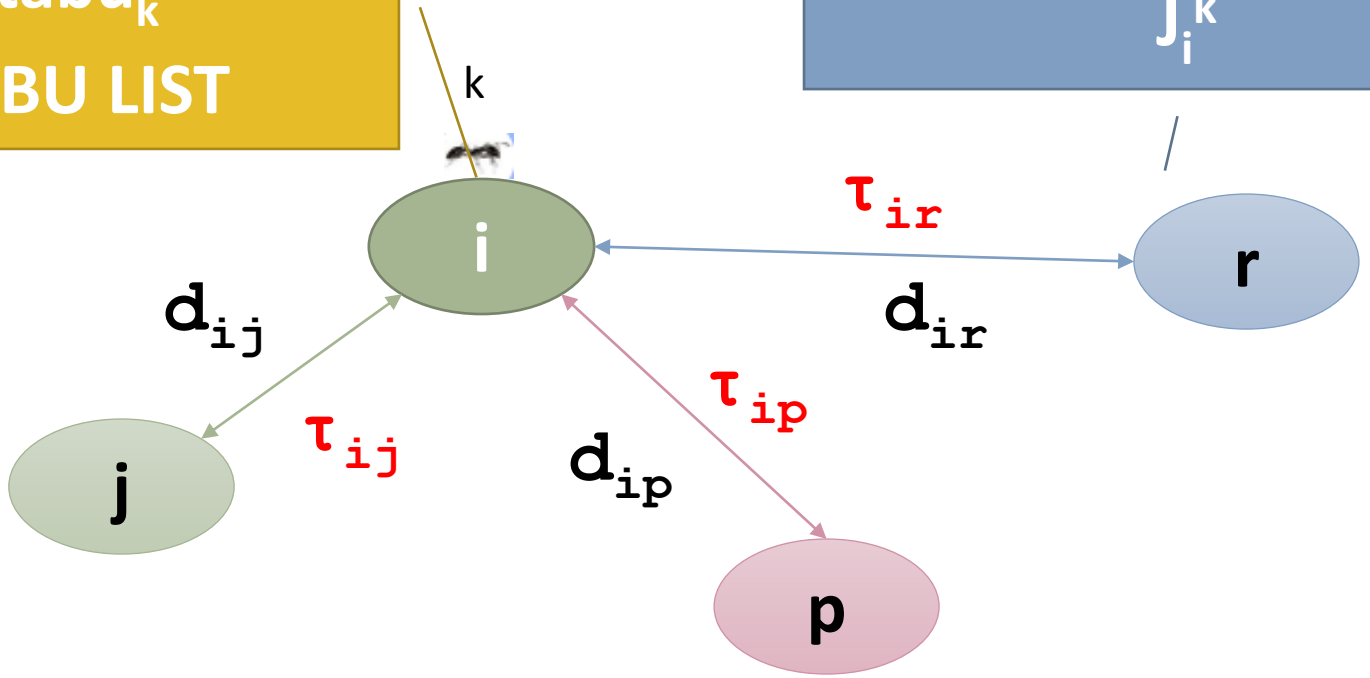
Lista nodurilor vizitate:

$tabu_k$

TABU LIST

Lista nodurilor accesibile din i:

J_i^k



$$\eta_{ij} = \frac{1}{d_{ij}}$$

Vizibilitatea (= inversul distantei intre 2 noduri)

✓ statica – informatie ce nu se schimba in timpul constructiei unei solutii

τ_{ij}

Cantitatea de **feromon** virtual depusa pe arcul ce leaga nodurile i si j

⇔Trail intensity

✓ dinamica

Algoritmul AS pentru TSP

- **REPEAT**
  - Furnicile sunt pozitionate in noduri (n)
  - **FOR** pas = 1 to n-1 **DO**
    - **FOR** k = 1 to m **DO**
      - Furnica k alege nodul urmator aplicand regula de tranzitie (**state transition rule**)
    - **End-for**
  - **End-for**
  - Urmele de feromon sunt modificate (**pheromone trail update**)
- **UNTIL** STOP condition

State transition rule AS
Regula de tranzitie AS

- Regula de trecere dintr-un nod **i** intr-un nod **j**
- **probabilistic**

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{r \in J_i^k}[\tau_{ir}(t)]^{\alpha}[\eta_{ir}]^{\beta}}, j \notin tabu_k$$

$$p_{ij}^{k}(t) = 0, altfel$$

α = parametru ce controleaza influenta feromonului

β = parametru ce controleaza influenta costului (vizibilitatea)

Pheromone trail update rule

- Regula de modificare feromon
- Aplicata dupa ce un tur complet este construit de toate furnicile

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t)$$

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{\,k} \qquad \Delta\tau_{ij}^{\,k} = \begin{cases} \dfrac{Q}{L_k} & \text{, daca (i,j) apartine drumului detectat de k} \\ 0 & \text{altfel} \end{cases}$$

ρ = coeficientul de evaporare feromon; Q=constanta
L_k = turul detectat de o furnica k
La initializare, cantitatea de feromon pe fiecare muchie = τ_0

AS pentru TSP cu 10 orase

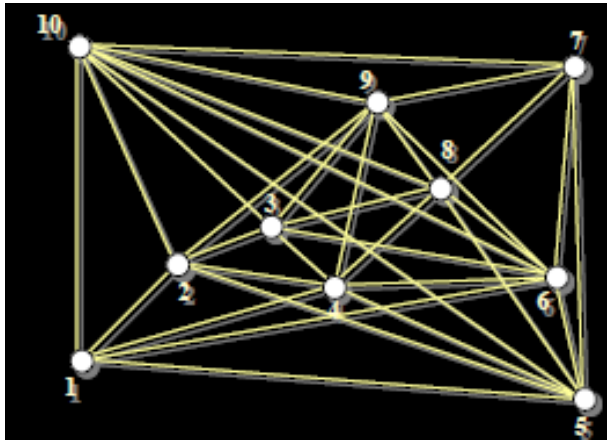
$m = n$

$\alpha = 1$

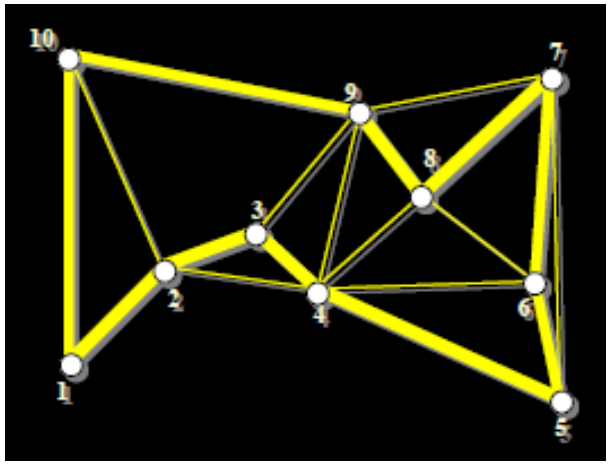
$\beta = 1$

$\rho = 0.5$

$Q = 100$



La initializare – cantitate de feromon egala pe toate muchiile



Dupa 100 de iteratii (cicluri)

ACS – Ant Colony System

- Imbunatatiri majore aduse algoritmului AS
- Dorigo, Gambardella, 1996, 1997
  - Dorigo and L. M. Gambardella. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53-66, 1997
- ACS difera de AS prin 3 modificari principale
  - *Regula de tranzitie* modificata pentru un echilibru explorare/exploatare mai bun
  - Regula de modificare feromon *globala* este aplicata numai arcelor ce apartin celui mai bun drum detectat
  - Este introdusa si o regula de modificare feromon la nivel *local* (local pheromone updating rule)

Algoritmul ACS

- **REPEAT**
  - Fiecare furnica este pozitionata intr-un nod de start
  - **REPEAT**
    - **State transition rule:** Fiecare furnica construieste incremental o solutie
    - **Local pheromone updating rule**
  - **UNTIL** Toate furnicile - un drum complet
  - **Global pheromone updating rule:** Urmele de feromon sunt modificate la nivel global
- **UNTIL** STOP condition

Regula de tranzitie ACS

$$j = \begin{cases} \arg\max_{u \in J_i^k} \{[\tau_{iu}(t)][\eta_{iu}]^\beta\}, q \le q_0 \\ J, q > q_0 \end{cases}$$

EXPLOITATION

BIASED EXPLORATION

q = variabila aleatoare uniform distribuita [0,1]

q_0 = parametru intre 0 si 1 => controleaza exploatare vs. explorare

J – variabila aleatoare selectata conform distributiei de probabilitate (AS)

$$p_{ij}^{\ k}(t) = \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{r \in J_i^k} [\tau_{ir}(t)]^\alpha [\eta_{ir}]^\beta}, j \notin tabu_k$$

ACS Global Updating Rule
Regula globala de modificare feromon

- La nivel global, numai furnica ce a generat cea mai buna solutie depune feromon
  - Feromon modificat numai pe cel mai bun drum L^+ generat pana la iteratia curenta inclusiv

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}(t)$$

$$\Delta\tau_{ij} = \begin{cases} \dfrac{1}{L^+}, (i,j) \in L^+ \\ 0 \qquad \text{altfel} \end{cases}$$

Diferente mici : daca se considera L^+ ca fiind cel mai bun din iteratia curenta

ACS Local Updating Rule
Regula locala de modificare feromon

- In timpul construirii unei solutii – drum – furnicile depoziteaza feromon pe muchiile traversate dupa regula:

$$\tau_{ij}(t+1) = (1-\xi)\tau_{ij}(t) + \xi\tau_0$$

- ξ – parametru; ex. ξ =0.1
- τ_0 – parametru; reprezinta de asemenea cantitatea initiala de feromon pe fiecare muchie

ACS – Rezultate si Comparatii

$\alpha = 1$ $\quad \rho = \xi = 0.1$

$\beta = 2$ $\quad \tau_0 = (n \cdot L_{nn})^{-1}$

$q_0 = 0.9$ $\quad m = 20$

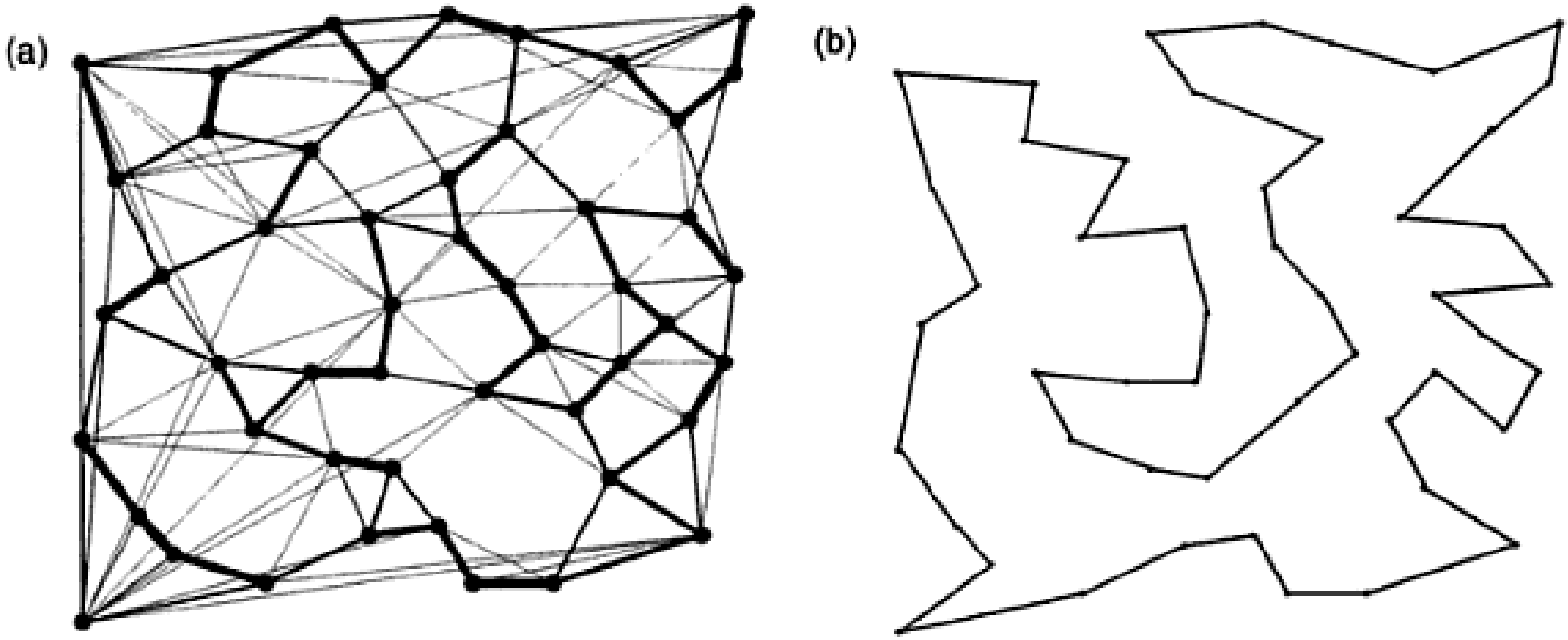
L_{nn} - **nearest neighbor** heuristic builds a tour by adding the closest node in terms of distance from the last node inserted in the path

ACS – 1250 iteratii → 25000 tururi generate

Problema	Info	Ant Colony System (Dorigo, Gambardella, 1997)	Algoritmi Genetici (Whitley et al, 1989)
Eil50 50 orase	Best tour	**425**	**428**
	# tururi pana la best	1830	25000
Eil75 75 orase	Best tour	**535**	**545**
	# tururi pana la best	3480	80000
KroA100 100 orase	Best tour	**21282**	**21761**
	# tururi pana la best	4820	103000

Media rezultatelor din 15 rulari

Exemplu solutie pentru Eil50



Max-Min Ant System (MMAS)

- T. Stutzle, H. Hoos, 1997
- Imbunatatirea AS
- Numai cea mai buna furnica modifica feromonul
- Cantitatea de feromon este pastrata intre anumite limite (min..max)

$$\tau_{ij} = (1-\rho)\tau_{ij} + \Delta\tau_{ij}^{best}$$

$$\Delta\tau_{ij}^{best} = \begin{cases} \frac{1}{L_{best}} & \text{, daca (i,j) apartine celui mai bun drum} \\ 0 & \text{altfel} \end{cases}$$

Exemplu aplicare ACO pentru TSP

1. **Iniţializare**:
   - t := 0 (timpul)
   - pentru fiecare muchie (i,j) se iniţializează
   - se plasează aleator **m** furnici în cele **n** noduri-oraş (m ≤ n)

$$\begin{cases} \tau_{ij} = \tau_0 \\ \Delta\tau_{ij} = 0 \end{cases}$$

   - fiecare furnică îşi modifică memoria (lista cu oraşele vizitate)
2. **Cât timp nu s-a parcurs numărul necesar de paşi pentru construcţia soluţiei (nr de paşi = n)**
   - Pentru fiecare furnică din colonie
     - Se măreşte soluţia parţială cu un element (furnica execută o mutare) si fiecare furnică k (aflată în oraşul i) alege următorul oraş pe care îl vizitează (State Transition Rule)
     - Se modifică local urma de feromon lăsată de fiecare furnică pe ultimul element adăugat în soluţie (Local Pheromone Update Rule)
3. **Se modifică urma de feromon de pe: (Global Pheromone Update Rule)**
   - drumurile parcurse de toate furnicile (AS)
   - cel mai bun drum (ACO)
   - cel mai bun drum parcurs de cea mai bună furnică (MMAS)

ACO - proprietati

- Algoritm iterativ
- Algoritm care construieşte progresiv soluţia pe baza
  - Informaţiilor euristice
  - Urmei de feromon
- Algoritm stocastic
- Avantaje
  - Rulare neîntreruptă şi adaptabilă schimbării în timp real a datelor
  - Feedback-ul pozitiv ajută la descoperirea rapidă a soluţiei
  - Calculul distribuit evită convergenţa prematură
  - Euristica greedy ajută la găsirea unei soluţii acceptabile încă din primele stadii ale căutării
  - Interacţiunea colectivă a indivizilor
- Dezavantaje
  - Converge încet faţă de alte căutări euristice
  - Funcţionează relativ slab pentru instanţe TSP cu mai mult de 100 oraşe

Aplicatii ACO

- **Problema comis-voiajorului**
  - **Traveling salesperson problem - TSP**
- **Probleme de rutare in retele (de telecomunicatii)**
  - **Network routing**
- **Problema rutarii vehiculelor**
  - **Vehicle Routing Problem – VRP**
- **Probleme de asignare**
  - **Quadratic Assignment Problem – QAP**
- **Probleme de planificare**
  - **Scheduling problem**
  - **Job-shop scheduling**
  - Satellite Image Classification
  - Movie effects
- **etc**

Recap

- PSO
  - Algoritm de căutare locală în fascicol
  - Potenţialele soluţii -> particule caracterizate prin:
    - poziţie în spaţiul de căutare
    - Viteză
  - Căutare cooperativă şi perturbativă bazată pe
    - Poziţia celei mai bune particule din grup
    - Cea mai bună poziţie a particulei de până atunci (particula are memorie)
- ACO
  - Algoritm de căutare locală în fascicol
  - Potenţialele soluţii -> furnici caracterizate prin:
    - Memorie – reţin paşii făcuţi în construirea soluţiei
    - Miros – iau decizii pe baza feromonului depus de celelalte furnici (comportament social, colectiv, colaborativ)
  - Căutare cooperativă şi constructivă