

Elemente de calcul evolutiv¹

Catalin Stoean

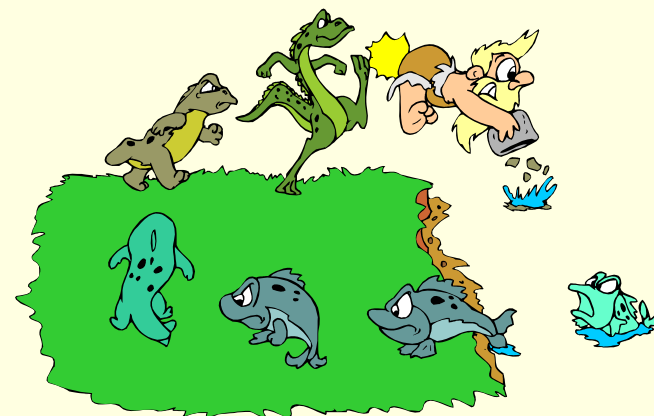
catalin.stoean@inf.ucv.ro

<http://inf.ucv.ro/~cstoean>

¹*Evolutie si inteligenta artificiala.
Paradigme moderne si aplicatii,
R. Stoean, C. Stoean*

Ce este calculul evolutiv?

- **Calculul evolutiv** reprezintă un domeniu de cercetare al informaticii.
 - După cum sugerează și numele, sunt implicate calcule, însă domeniul este inspirat din procesul evoluției naturale.
- Algoritmii care apar în acest domeniu se numesc **algoritmi evolutivi** și ei includ subdomenii precum:
 - programarea evolutivă
 - strategii evolutive
 - algoritmi genetici
 - programare genetică



Ce urmărim astăzi?

- Concepte de bază ale calculului evolutiv:
 - Metafora evolutivă
 - Componentele unui algoritm evolutiv
 - Aplicații pe probleme concrete
 - Problema damelor
 - Optimizarea unei funcții

Metafora evolutivă

- Puterea evoluției în natură este evidentă, deci este normal ca evoluția naturală să fie o sursă de inspirație pentru cercetători.
- Metafora fundamentală a calculului evolutiv leagă evoluția naturală de un anumit tip de rezolvare de probleme, **încercare-și-eroare**.
- Evoluția naturală poate fi descrisă astfel:
 - Într-un mediu se găsește o **populație** de **indivizi** care se luptă pentru supraviețuire și are capacitatea de a se reproduce.
 - **Performanța** acestor indivizi este strâns legată de felul în care indivizii se adaptează la mediul înconjurător și de modul în care indivizii reușesc să își atingă scopurile.
 - **Performanța** fiecărui individ influențează în mod direct **șansa** lui de a se multiplica și de a supraviețui.



Metafora evolutivă 2

- Într-o astfel de vedere macroscopică a evoluției, un rol cheie este jucat de **selecție**.
- Având un mediu care poate conține doar un număr redus de indivizi dotați cu instinctul primar al **reproducerii**, selecția este un factor inerent în cazul în care se dorește ca mărimea populației să nu crească exponențial.
- Selecția naturală îi favorizează, ca și în natură, pe indivizii care se luptă cel mai bine pentru resurse, altfel spus pe cei care se adaptează cel mai bine la mediu.
 - Este ceea ce se numește **supraviețuirea celor mai puternici.**



Metafora evolutivă 3

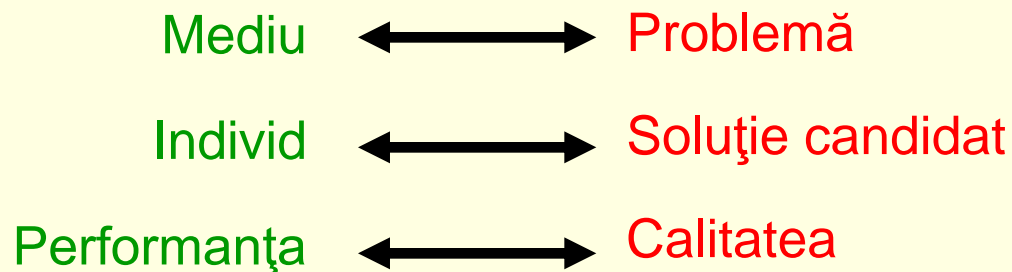


- Analogia între contextul rezolvării de **probleme de tip încercare-și-eroare** și **evoluția naturală** se face în felul următor:
 - **Mediul** în care se află indivizii se identifică cu **problema de rezolvat**
 - **Indivizii** sunt **potențiale soluții** ale problemei
 - **Măsura de performanță** va contoriza în acest caz **cât de bună calitativ** este soluția curentă în rezolvarea problemei.
- Soluții potențiale sunt **inițial generate** iar cele care sunt mai bune calitativ au cele mai mari șanse să fie păstrate și să participe la procesul de recombinare.

Metafora evolutivă 4

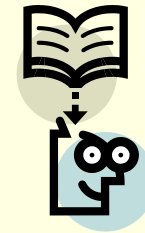
EVOLUTIE

REZOLVAREA DE PROBLEME



Performanța influențează șansele de reproducere și supraviețuire

Calitatea influențează șansele de a forma noi soluții

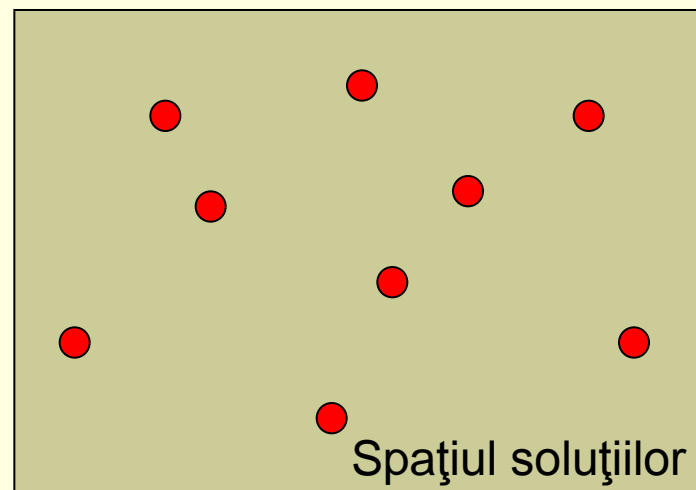
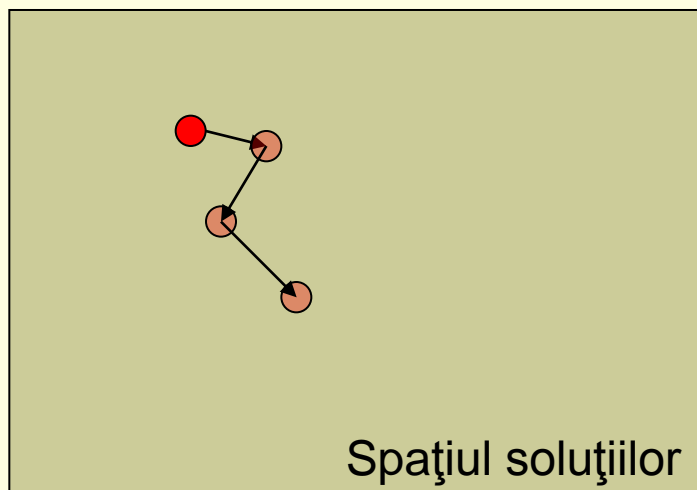


Ideile principale

- O **populație cu indivizi** există într-un **mediu** cu resurse limitate.
- Competiția pentru aceste resurse face ca selecția să îi **avantajeze** pe indivizii mai buni care s-au adaptat mai bine la mediu.
- Acești indivizi acționează ca **părinți** ai generației de noi indivizi obținuți prin **recombinare** și **mutație**.
- Noii indivizi sunt evaluați și **luptă pentru supraviețuire** cu generația anterioară (posibil chiar cu părinții lor).
- **Selecția naturală** face ca performanța populației să crească odată cu trecerea timpului.
- **Operatorii de variație** (mutația și recombinația) **crează diversitatea** necesară în populații și aduc noutate.
- **Selecția** reduce diversitatea și acționează ca o forță care duce la **creșterea calității**.

Cautarea singulara vs. Cautarea din cadrul algoritmilor evolutivi

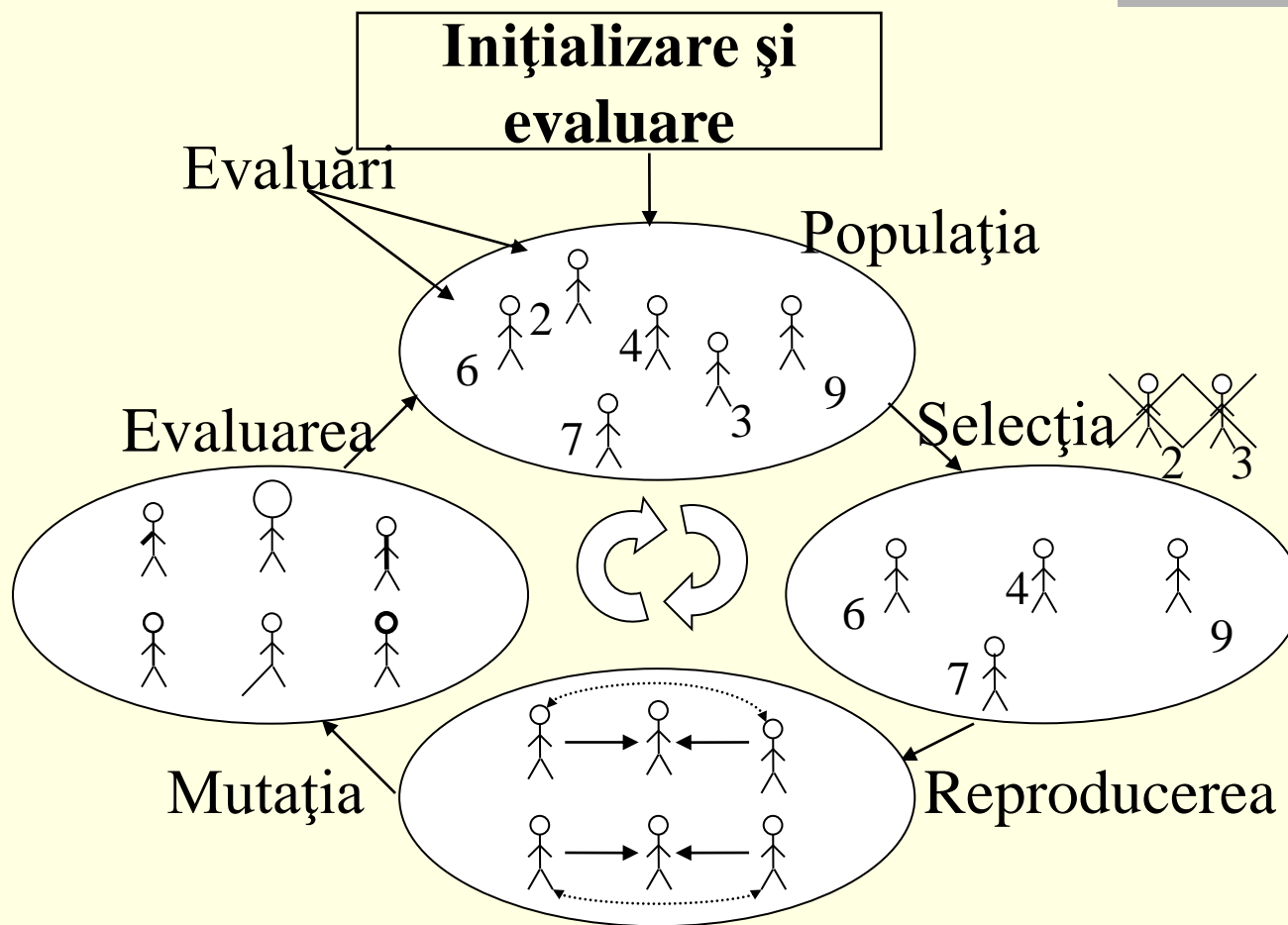
- **Cautarea singulara** (hill climbing, simulated annealing) folosește o singură soluție care suferă transformări.
 - Dezavantaj: se blochează des în maxime (optime) locale.
- **Algoritmii evolutivi** folosesc multe soluții care evoluează (suferă transformări) simultan – figură dreapta.



Ce urmărim astăzi?

- Concepte de bază ale calculului evolutiv:
 - Metafora evolutivă
 - Componentele unui algoritm evolutiv
 - Aplicații pe probleme concrete
 - Problema damelor
 - Optimizarea unei funcții

Schema unui algoritm evolutiv



Schema unui algoritm evolutiv

1. Se inițializează populația cu indivizi generați aleator;
2. Se evaluează fiecare individ;
3. **Cât timp** (condiția de terminare nu este satisfăcută) **execută**
 - 3.1 Se selectează părinții;
 - 3.2 Se recombina perechi de părinți;
 - 3.3 Se aplică mutația asupra descendenților obținuți după recombinare;
 - 3.4 Se evaluează fiecare descendent;
 - 3.5 Se selectează indivizii care vor forma următoarea generație;
4. **Sfârșit cât timp**

Componentele unui algoritm evolutiv



- Reprezentarea
- Evaluarea
- Populația
- Selecția părinților
- Operatorii variaționali
 - Recombinarea
 - Mutația
- Inițializarea și terminarea

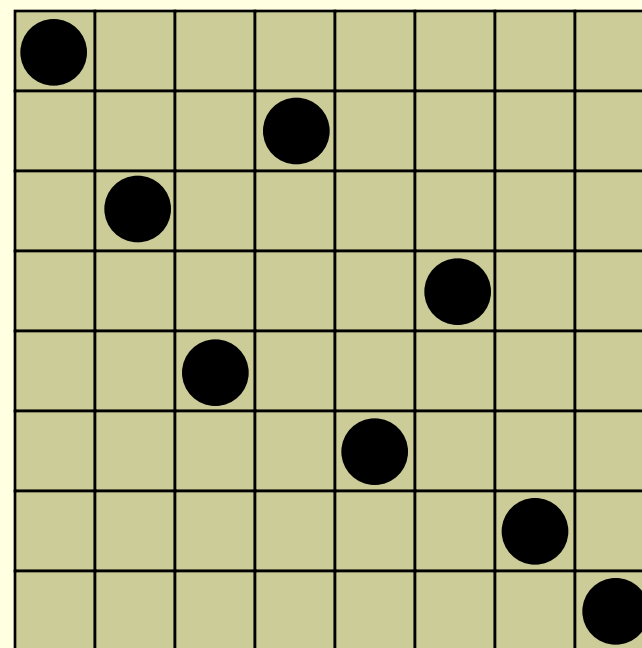
Reprezentarea

- **Soluțiile candidat** ale problemei de rezolvat trebuie reprezentate (codificate) sub formă de **indivizi (cromozomi)**.
- Fiecare cromozom conține **gene**.
- Orice element din spațiul soluțiilor trebuie să aibă corespondent în spațiul cromozomilor și invers.
- Pentru fiecare problemă trebuie aleasă o reprezentare cât mai potrivită.

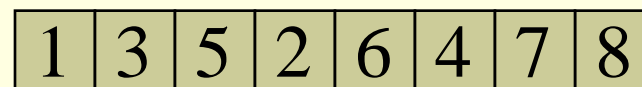
Reprezentarea pentru problema damelor

Potențială soluție:
o configurație a celor
8 dame

Cromozomul:
o permutare a primelor
8 cifre.

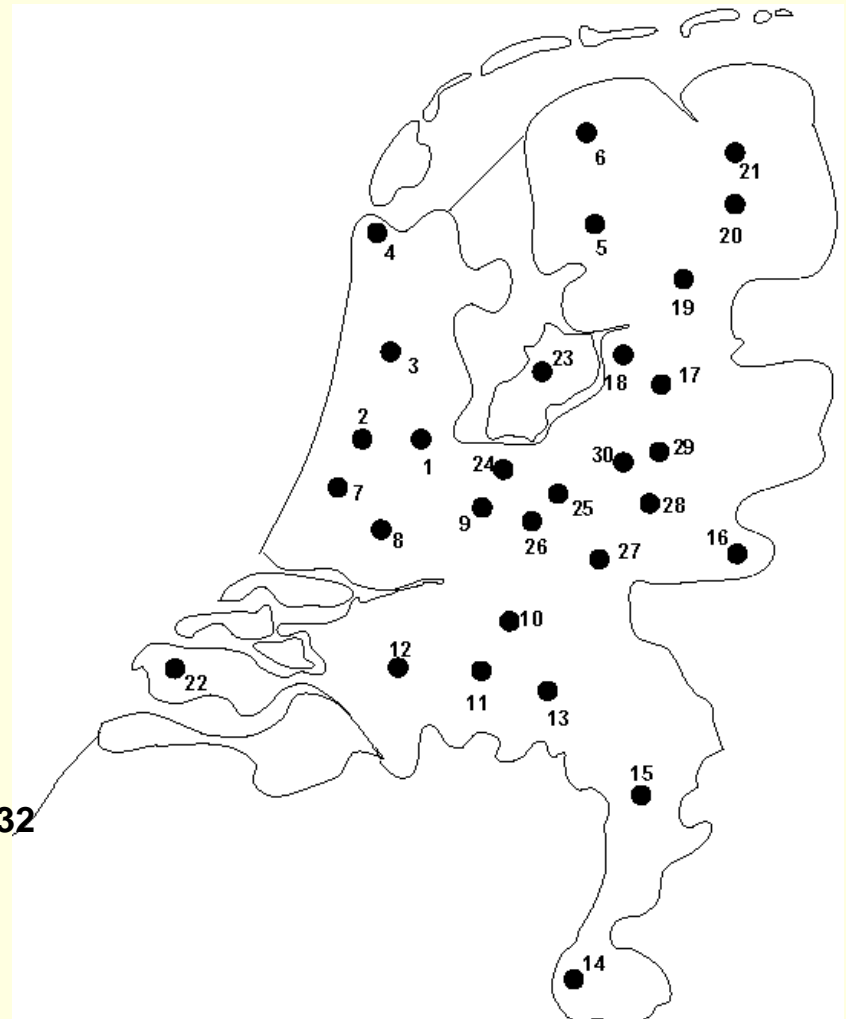


Codificare



Problema comis voiajorului

- Problema:
 - Se dau n orașe
 - Să se găsească un tur complet de lungime minimală
- Reprezentare:
 - Etichetăm orașele $1, 2, \dots, n$
 - Un tur complet este o permutare (pt. $n=4$: $[1,2,3,4]$, $[3,4,2,1]$)
- Spațiul de căutare este **imens**:
pentru 30 de orașe sunt $30! \approx 10^{32}$ tururi posibile!



Reprezentare pentru optimizarea unei funcții (găsirea maximului pe un interval)

- Imaginați-vă că vă dau $z = f(x)$ puncte în plus la examen.

$$f(x) = x \cdot \sin(10\pi \cdot x)$$

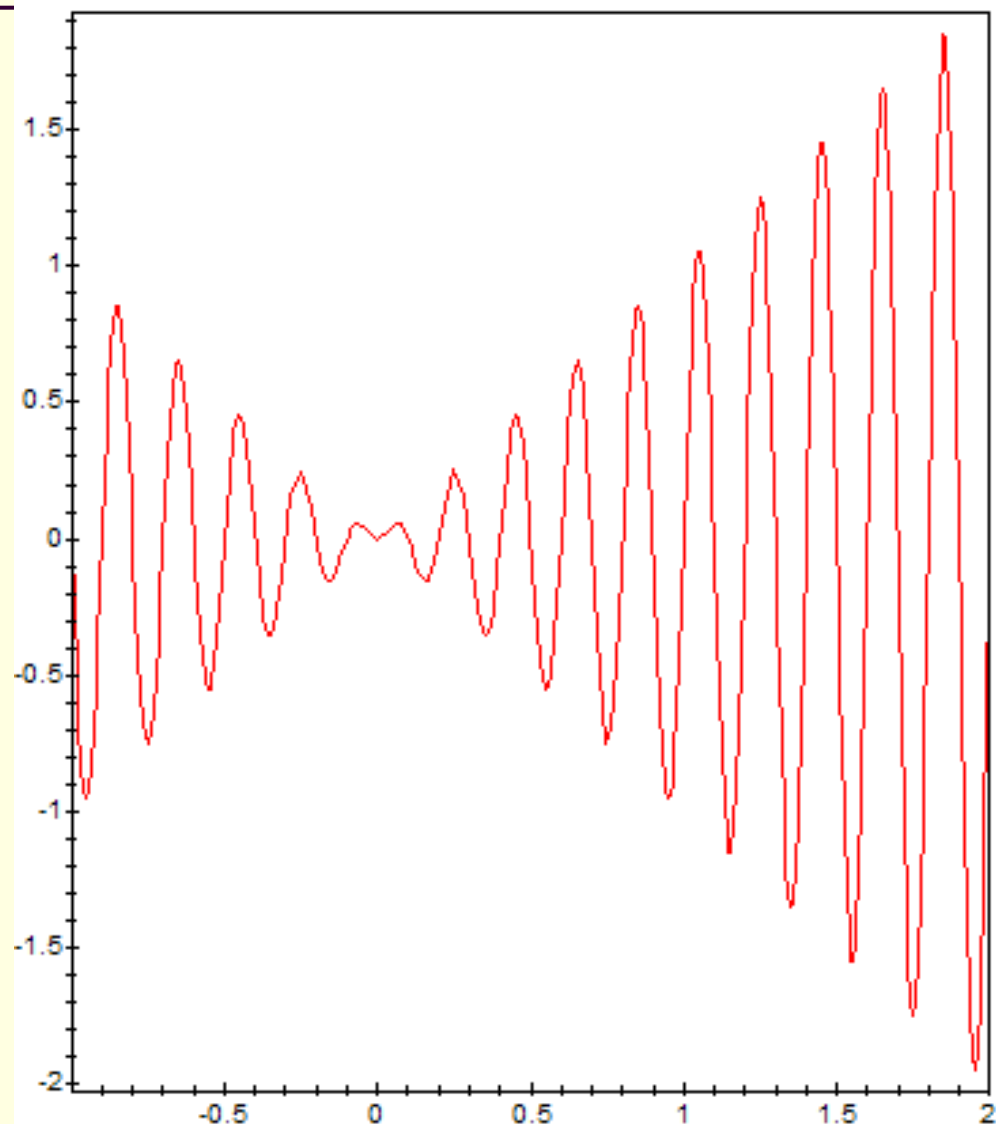
- Găsiți-l pe x în intervalul $[-1, 2]$ astfel încât z să fie cât mai mare (ca să câștigați cât mai multe puncte)!
- Altfel spus, găsiți x_0 din $[-1, 2]$ cu proprietatea că

$$\mathbf{f(x_0) \geq f(x), \text{ pentru orice } x \text{ din } [-1, 2]}$$

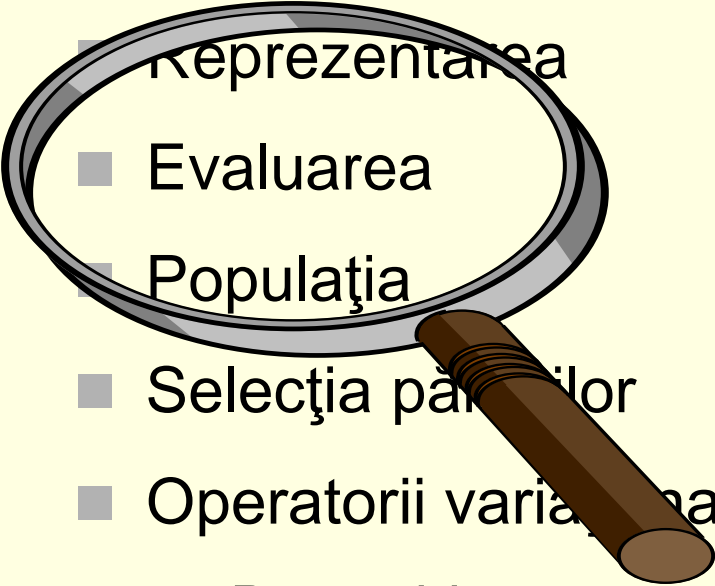
- In acest caz, spațiul soluțiilor se poate identifica cu spațiul cromozomilor.

Reprezentarea funcției

$f(x) = x \cdot \sin(10\pi \cdot x)$,
pe intervalul $[-1, 2]$.



Componentele unui algoritm evolutiv

- 
- Reprezentarea
 - Evaluarea
 - Populația
 - Selecția părinților
 - Operatorii variaționali
 - Recombinarea
 - Mutația
 - Inițializarea și terminarea

Funcția obiectiv









- Înainte de a defini funcția de performanță, trebuie stabilit **obiectivul problemei**, care este sarcina care trebuie îndeplinită. Este vorba de găsirea **funcției obiectiv**.
- În cazul problemei comis-voiajorului, obiectivul se referă la **minimizarea** drumului pe care îl parcurge comis-voiajorul prin vizitarea fiecărui oraș o singură dată cu întoarcere în orașul de pornire.
- Pentru problema damelor, obiectivul este de a așeza cele 8 dame într-o configurație astfel încât să nu se atace reciproc.

Funcția de evaluare (performanță)

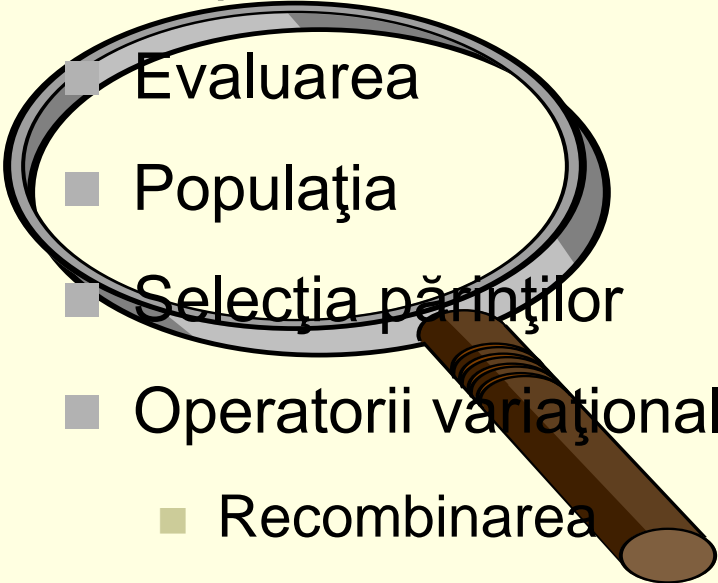
- Rolul **funcției de performanță** (sau de *evaluare*, ori *de calitate*) este de a măsura cât de bine se adaptează indivizii la mediu.
- Pentru unele probleme poate fi aceeași cu funcția obiectiv (de exemplu, în optimizarea de funcții).
- Funcția de evaluare atribuie fiecărui individ o valoare (de obicei, reală) care reprezintă cât de bun (de performant) este cromozomul respectiv.
- Se dorește maximizarea performanței indivizilor, ceea ce duce de obicei la dorința de a maximiza valorile funcției de performanță (nu în tot timpul este cazul însă, vezi funcția de performanță la problema damelor).

Funcția de performanță pentru problema damelor

- **Penalizarea unei dame** este proporțională cu numărul de dame pe care le atacă.
- **Penalizarea unei configurații** este dată de suma penalizărilor pentru fiecare damă.
- Scopul => de a minimiza numărul total de penalizări!

18	12	14	13	13	12	14	14
14	16	13	15	12	14	12	16
14	12	18	13	15	12	14	14
15	14	14		13	16	13	16
	14	17	15		14	16	16
17		16	18	15		15	
18	14		15	15	14		16
14	14	13	17	12	14	12	18


Componentele unui algoritm evolutiv

- Reprezentarea
 - Evaluarea
 - Populația
 - Selecția părinților
 - Operatorii variaționali
 - Recombinarea
 - Mutația
 - Inițializarea și terminarea
- 

Populația

- Populația constă dintr-o mulțime de indivizi care nu sunt neapărat toți diferiți între ei.
- **Mărimea populației** reprezintă numărul de indivizi pe care îi conține populația.
 - De obicei acest număr este constant.
- Operatorii de selecție iau în considerare întreaga populație de la generația curentă.
- Populația este cea care evoluează de-a lungul generațiilor, nu indivizii în particular.

Componentele unui algoritm evolutiv

- Reprezentarea
 - Evaluarea
 - Populația
 - Selecția părinților
 - Operatorii variaționali
 - Recombinarea
 - Mutația
 - Inițializarea și terminarea
- 

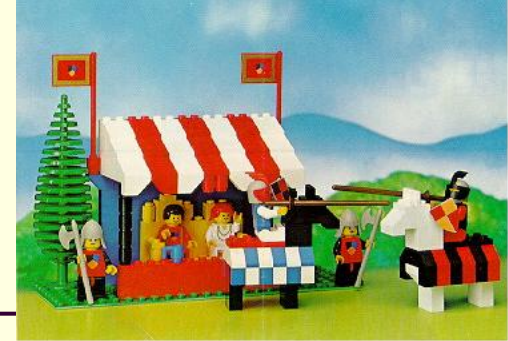
Selecția

- Procesul de selecție apare de două ori în cursul unei parcurgeri a ciclului *cât timp* a algoritmului evolutiv prezentat anterior.
- **Selecția pentru reproducere** (selecția părinților), când sunt aleși părinții generației următoare.
- **Selecția pentru înlocuire** (selecția pentru supraviețuire), care apare când indivizii care vor forma populația din următoarea generație sunt aleși dintre descendenții obținuți și indivizii din populația curentă.
- Modul în care crește calitatea generală a soluțiilor depinde de ambele tipuri de selecție.

Selecția pentru reproducere

- Selecția pentru reproducere are rolul de a alege, în funcție de calitatea indivizilor din populația curentă, care sunt cei considerați pentru a se aplica operatori de variație asupra lor în vederea obținerii de noi soluții candidat.
- Selecția pentru reproducere este **probabilistă**; indivizii foarte performanți au șanse bune de a fi selectați pentru reproducere, în timp ce indivizii cu valori mici pentru funcția de performanță au șanse mici să devină părinți.
- Natura *probabilistă* a selecției este cea care face căutarea să scape de optimele locale.

Selecția turnir



- Se selectează în mod aleator k indivizi și sunt evaluați.
- Cel mai bun dintre aceștia este *selectat* ca și părinte pentru populația din generația următoare.
- Algoritmul de mai jos selectează N părinți din generația curentă.

$i = 1$

Cât timp $i < N$ execută

Selectează k indivizi în mod aleator din întreaga populație

Selectează-l pe cel mai performant individ s din cei k

părinți[i] = s


$i = i + 1$

Sfârșit cât timp

Selecția pentru supraviețuire

- Selecția pentru supraviețuire decide care indivizi din populația curentă împreună cu descendenții obținuți sunt opriți pentru a forma populația generației următoare.
- Cum numărul de indivizi din populație este de obicei constant, selecția pentru supraviețuire intervine cu scopul de a păstra aceeași mărime a populației.
- De obicei decizia depinde și în acest caz de performanța indivizilor, cei mai buni fiind favorizați.

Componentele unui algoritm evolutiv

- Reprezentarea
 - Evaluarea
 - Populația
 - Selecția părinților
 - Operatorii variaționali
 - Recombinarea
 - Mutația
 - Inițializarea și terminarea
- 

Operatorii de variație

- Operatorul de selecție are rolul de a concentra căutarea pe regiunile cele mai promițătoare din spațiul de căutare.
- Rolul operatorilor de variație este de a crea noi soluții candidat din cele vechi, de a mări diversitatea populației.
- Operatorii de variație sunt dependenți de reprezentarea utilizată, astfel că pentru reprezentări diferite trebuie definiți operatori specifici.
- În funcție de aritate, operatorii de variație sunt împărțiți în:
 - Aritate = 1 => **Mutație**
 - Aritatea > 1 => **Recombinare**

Mutația

- Acționează asupra unui individ și produce un altul.
- După ce se aplică asupra unui individ, rezultatul (descendent) conține mici modificări față de individul inițial.
- Operatorul face ca toate valorile unei gene să fie disponibile pentru procesul de căutare.
- Genele ale căror valori sunt considerate pentru a fi schimbate sunt alese printr-o manieră probabilistă.
- Un parametru al algoritmului evolutiv este dat de **probabilitatea de mutație**.

Cum se aplica mutația

- Presupunem că avem probabilitatea de mutație $p_m = 0.3$.
- Notăm mărimea populației cu N .

Pentru $i = 1$ până la N execută

Pentru $j = 1$ până la numărul de gene ale cromozomului execută

$g =$ număr aleator în intervalul $[0, 1]$;

Dacă $g < p_m$ atunci

Aplică mutația pentru gena j a

individului i

Sfârșit dacă

Sfârșit pentru

Sfârșit pentru

Mutația pentru problema damelor și pentru comis-voiajor

- O mică variație într-o permutare:
 - Se aleg două valori în mod aleator (5 și 7 în imaginea din stânga).
 - Pozițiile celor două valori sunt interschimbate.



Mutația pentru reprezentarea reală

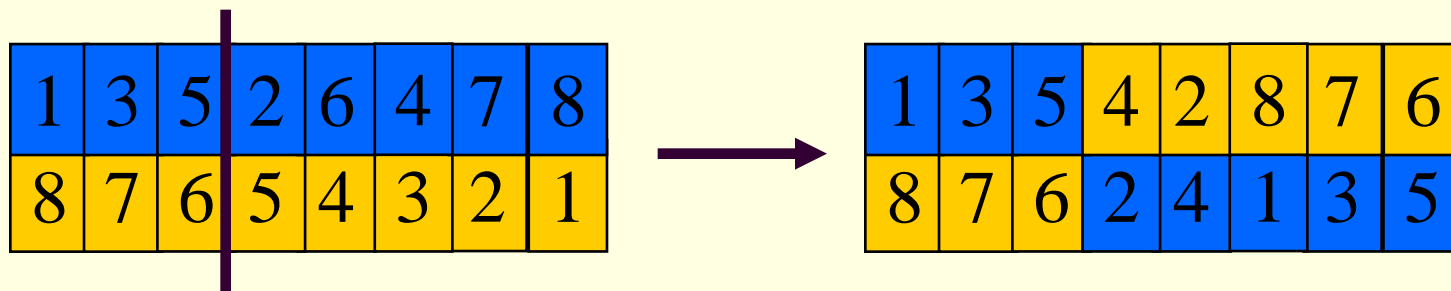
- În cazul optimizării unei funcții, de exemplu, avem un individ $X = (x_1, x_2, \dots, x_p)$, unde $x_i \in [a_i, b_i]$.
- Se poate obține un individ $X' = (x'_1, x'_2, \dots, x'_p)$, unde x'_i este ales în mod aleator în intervalul $[a_i, b_i]$.
- Exemplu:
 - $X = (0.4, 0.7, 0.3, 0.9)$ cu toate genele din $[0, 1]$.
 - Dacă gena a doua suferă mutație, individul ar putea deveni $X' = (0.4, 0.1, 0.3, 0.9)$.

Recombinarea

- Recombinarea sau încrucișarea implică doi sau mai mulți indivizi (părinți) aleși cu o anumită **probabilitate de încrucișare** în scopul de a genera unul, doi sau mai mulți indivizi prin combinarea genelor părinților.
- Recombinarea reprezintă un operator stochastic de vreme ce alegeri precum ce părți să fie moștenite de la un părinte și ce părți de la alt părinte sau modul în care părțile acestea sunt combinate sunt făcute în mod aleator.
- Prin împreunarea a doi indivizi cu caracteristici diferite, este obținut un descendent (sau doi) care combină aceste caracteristici.

Recombinarea pentru problema damelor și pentru comis-voiajor

- Combinarea a două permutări în două noi permutări:
 - Se alege în mod aleator un punct de încrucișare (linia verticală)
 - Se copiază primele două părți în cei doi descendenți
 - A doua parte se completează prin inserarea de valori de la celălalt părinte:
 - În ordinea în care apar acolo
 - Începând cu punctul de după tăietură
 - Sărind valorile care se găsesc deja în descendent



Recombinarea pentru reprezentarea reală

- Discretă
 - Fiecare genă vine de la unul din părinți cu probabilitate egală
- Intermediară
 - Exploatează ideea de a crea descendenți între părinți
 - $z_j = \alpha x_j + (1 - \alpha) y_j$ unde $\alpha : 0 \leq \alpha \leq 1$.
 - Parametrul α poate fi
 - Constant
 - Variabil (să depindă de vârsta populației)
 - Ales aleator de fiecare dată.

Recombinarea pentru reprezentarea reală

- Părinții: (x_1, \dots, x_n) și (y_1, \dots, y_n)
- Se alege un număr k aleator
- Descendentul este $(x_1, \dots, x_{k-1}, \alpha y_k + (1 - \alpha) x_k, \dots, x_n)$
- Invers pentru celălalt descendent.
- La exemplul de mai jos, pentru $\alpha = 0.5$...

0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
-----	-----	-----	-----	-----	-----	-----	-----	-----

0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.5	0.9
-----	-----	-----	-----	-----	-----	-----	-----	-----



0.3	0.2	0.3	0.2	0.3	0.2	0.3	0.2	0.3
-----	-----	-----	-----	-----	-----	-----	-----	-----

0.3	0.2	0.3	0.2	0.3	0.2	0.3	0.5	0.3
-----	-----	-----	-----	-----	-----	-----	-----	-----

Recombinarea cea mai folosită pentru reprezentarea reală

- Părinții: (x_1, \dots, x_n) și (y_1, \dots, y_n)
- Fiecare genă x'_k se calculează după formula
$$x'_k = \alpha y_k + (1 - \alpha) x_k$$
- Invers pentru celălalt descendent.
- La exemplul de mai jos, pentru $\alpha = 0.5$...

0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
-----	-----	-----	-----	-----	-----	-----	-----	-----

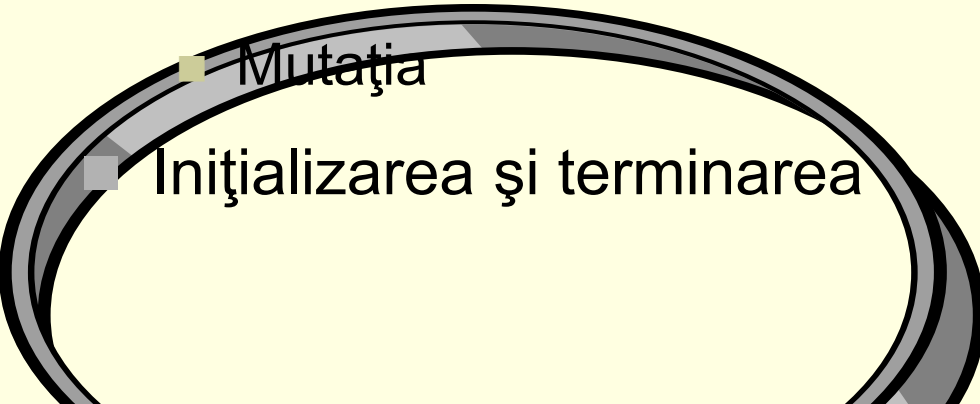
0.2	0.2	0.3	0.3	0.4	0.4	0.5	0.5	0.6
-----	-----	-----	-----	-----	-----	-----	-----	-----



0.3	0.2	0.3	0.2	0.3	0.2	0.3	0.2	0.3
-----	-----	-----	-----	-----	-----	-----	-----	-----

0.2	0.2	0.3	0.3	0.4	0.4	0.5	0.5	0.6
-----	-----	-----	-----	-----	-----	-----	-----	-----

Componentele unui algoritm evolutiv

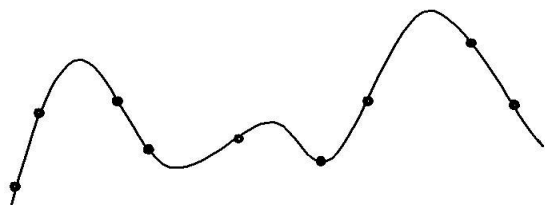
- Reprezentarea
 - Evaluarea
 - Populația
 - Selecția părinților
 - Operatorii variaționali
 - Recombinarea
 - Mutația
 - Inițializarea și terminarea
- 

Inițializarea și condiția de terminare

- De obicei **inițializarea** se face în mod **aleator**.
 - În populația inițială se pot include și soluții existente.
- Condiția de terminare se verifică la fiecare generație
 - Atingerea unei anumite performanțe
 - Ajungerea la un anumit număr maxim de generații
 - Ajungerea la un nivel foarte mic de diversitate în populație
 - Atingerea unui anumit număr de generații fără să se mai fi câștigat performanță.

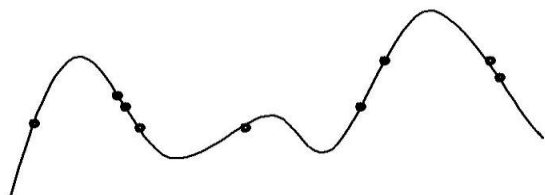
Comportamentul tipic al unui algoritm evolutiv

Situații în optimizarea unei funcții de o singură variabilă



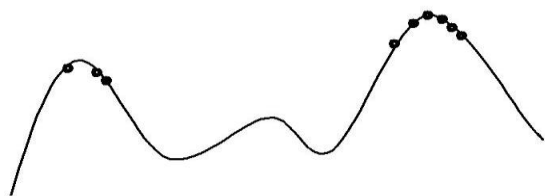
Fază de început:

Distribuția populației este qvasi-aleatoare



Fază de mijloc:

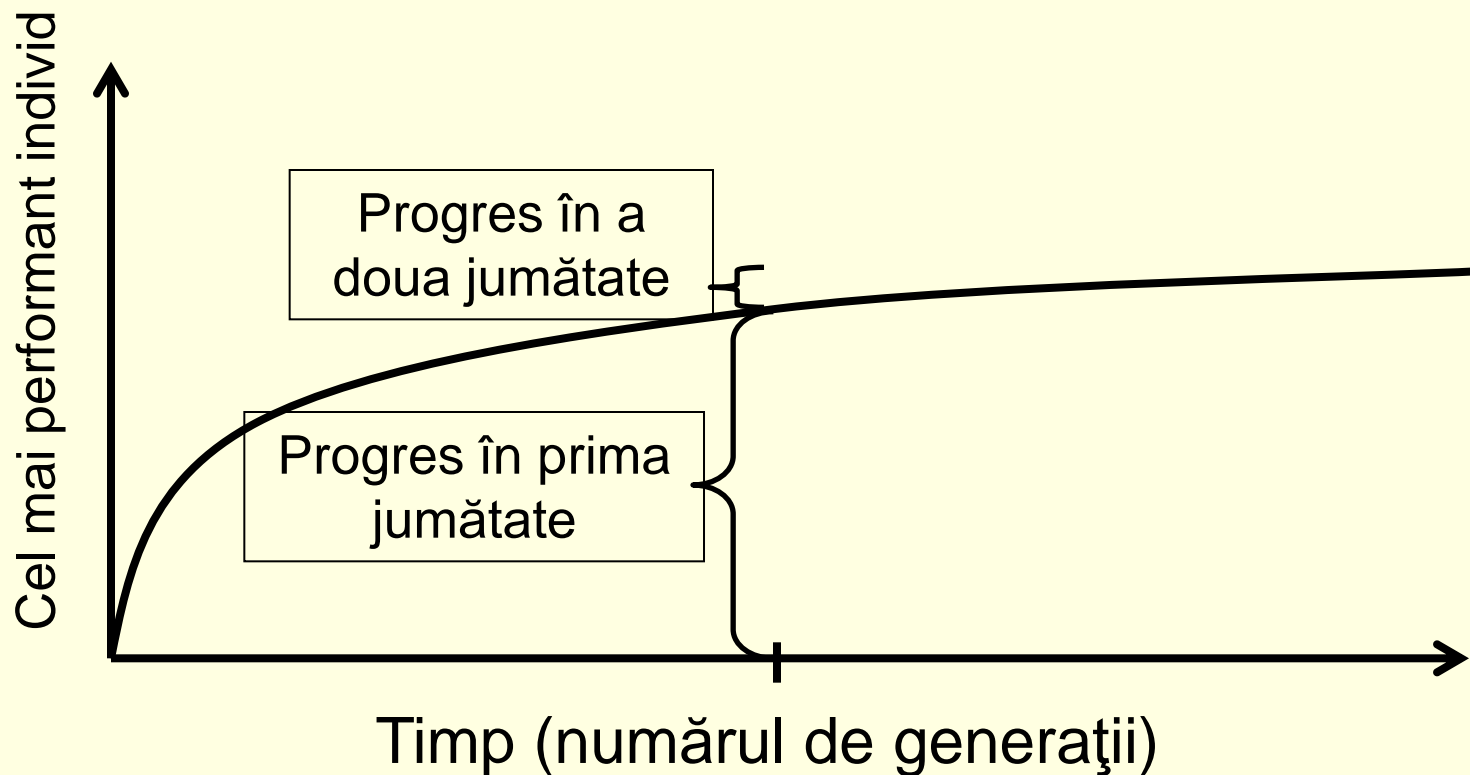
Populația aranjată în jurul / pe dealuri



Fază de final:

Populația este concentrată pe dealurile înalte

Cât să dureze o rulare?



Recapitulare

- Algoritmii evolutivi sunt inspirați din modul în care au evoluat lucrurile în natură.
- Folosesc mulțimi de potențiale soluții pe care le evoluează de-a lungul mai multor generații.
- Operatorul de **selecție** este cel care face ca performanța medie a populației să crească o dată cu trecerea generațiilor.
- Pentru introducerea de noi soluții sunt folosiți **operatori de variație** precum **mutația** și **recombinarea**.