# Elemente de calcul evolutiv[1]

Catalin Stoean

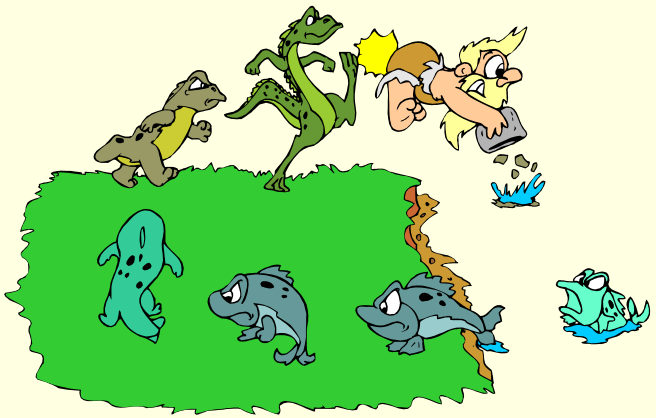
catalin.stoean@inf.ucv.ro

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

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

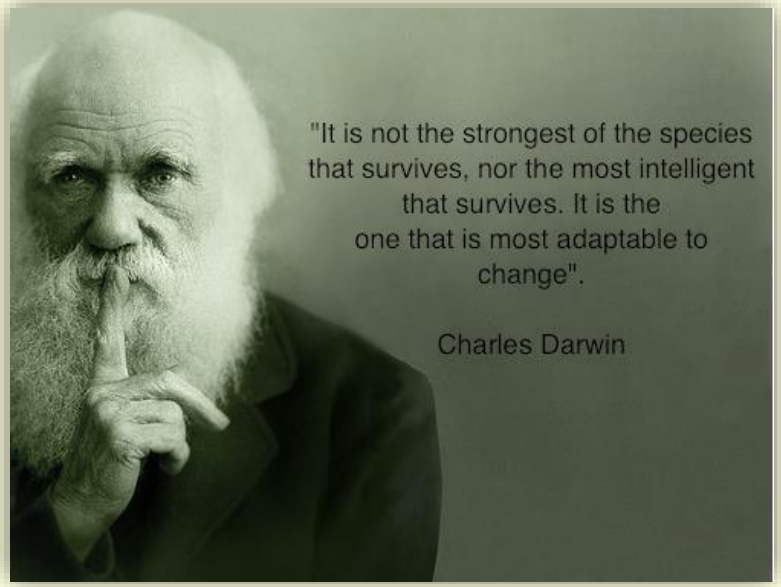
# Ce este calculul evolutiv?

- **Calculul evolutiv** reprezintă un domeniu de cercetare al informaticii.
  - După cum sugerează şi numele, sunt implicate calcule, însă domeniul este inspirat din procesul evoluţiei naturale.
- Algoritmii care apar în acest domeniu se numesc **algoritmi evolutivi** şi 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
  - Aplicaţii pe probleme concrete
    - Problema damelor
    - Optimizarea unei funcţii



# Metafora evolutivă

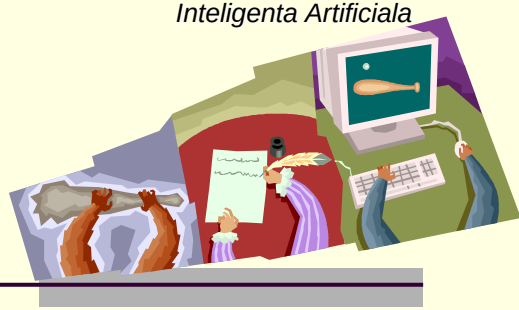
- Puterea evoluţiei în natură este evidentă, deci este normal ca evoluţia naturală să fie o sursă de inspiraţie pentru cercetători.
- Metafora fundamentală a calculului evolutiv leagă evoluţia naturală de un anumit tip de rezolvare de probleme, **încercare-şi-eroare**.
- Evoluţia naturală poate fi descrisă astfel:
  - Într-un mediu se găseşte o **populaţie** de **indivizi** care se luptă pentru supravieţuire şi are capacitatea de a se reproduce.
  - **Performanţa** acestor indivizi este strâns legată de felul în care indivizii se adaptează la mediul înconjurător şi de modul în care indivizii reuşesc să îşi atingă scopurile.
  - **Performanţa** fiecărui individ influenţează în mod direct **şansa** lui de a se multiplica şi de a supravieţui.

# Metafora evolutivă 2

- Într-o astfel de vedere macroscopică a evoluţiei, un rol cheie este jucat de **selecţie**.
- Având un mediu care poate conţine doar un număr redus de indivizi dotaţi cu instinctul primar al **reproducerii**, selecţia este un factor inerent în cazul în care se doreşte ca marimea populaţiei să nu crească exponenţial.
- Selecţia naturală îi favorizează, ca şi 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 numeşte

    **supravieţuirea celor mai puternici**.

# Metafora evolutivă 3



- Analogia între contextul rezolvării de **probleme de tip încercare-şi-eroare** şi **evoluţia naturală** se face în felul următor:
  - **Mediul** în care se află indivizii se identifică cu **problema de rezolvat**
  - **Indivizii** sunt **potenţiale soluţii** ale problemei
  - **Măsura de performanţă** va contoriza în acest caz **cât de bună calitativ** este soluţia curentă în rezolvarea problemei.
- Soluţii potenţiale sunt **iniţial generate** iar cele care sunt mai bune calitativ au cele mai mari şanse să fie păstrate şi să participe la procesul de recombinare.

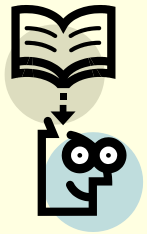
# Metafora evolutivă 4

| EVOLUTIE | | REZOLVAREA DE PROBLEME |
|---|---|---|
| Mediu | ⟷ | Problemă |
| Individ | ⟷ | Soluţie candidat |
| Performanţa | ⟷ | Calitatea |

**Performanţa** influenţează şansele de reproducere şi supravieţuire

**Calitatea** influenţează şansele de a forma noi soluţii

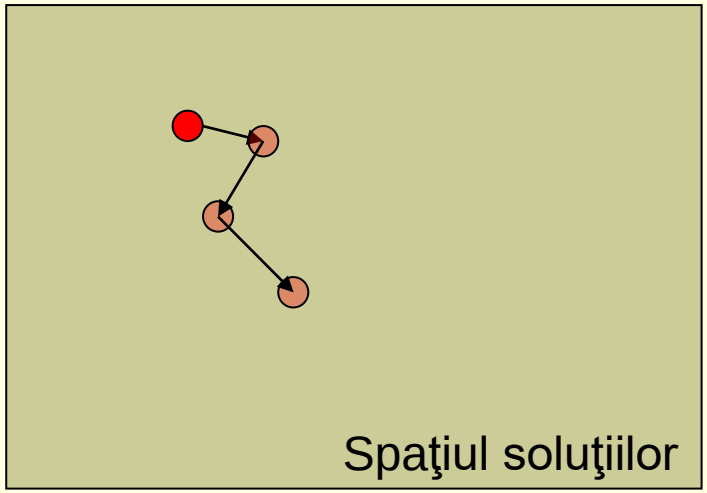
# Ideile principale

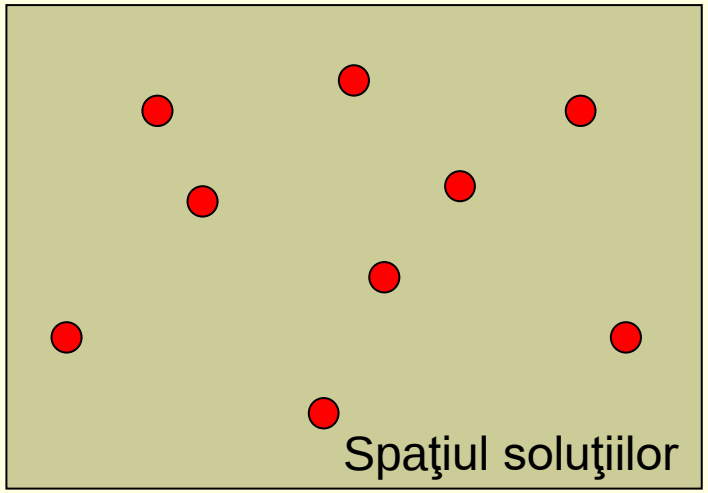


- O **populaţie cu indivizi** există într-un **mediu** cu resurse limitate.
- Competiţia pentru aceste resurse face ca selecţia să îi **avantajeze** pe indivizii mai buni care s-au adaptat mai bine la mediu.
- Aceşti indivizi acţionează ca **părinţi** ai generaţiei de noi indivizi obţinuţi prin **recombinare** şi **mutaţie**.
- Noii indivizi sunt evaluaţi si **luptă pentru supravieţuire** cu generaţia anterioară (posibil chiar cu părinţii lor).
- **Selecţia naturală** face ca performanţa populaţiei să crească odată cu trecerea timpului.
- **Operatorii de variaţie** (mutaţia şi recombinarea) **creează diversitatea** necesară în populaţii şi aduc noutate.
- **Selecţia** reduce diversitatea şi acţionează ca o forţă care duce la **creşterea calităţii**.

# Cautarea singulara vs. Cautarea din cadrul algoritmilor evolutivi

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

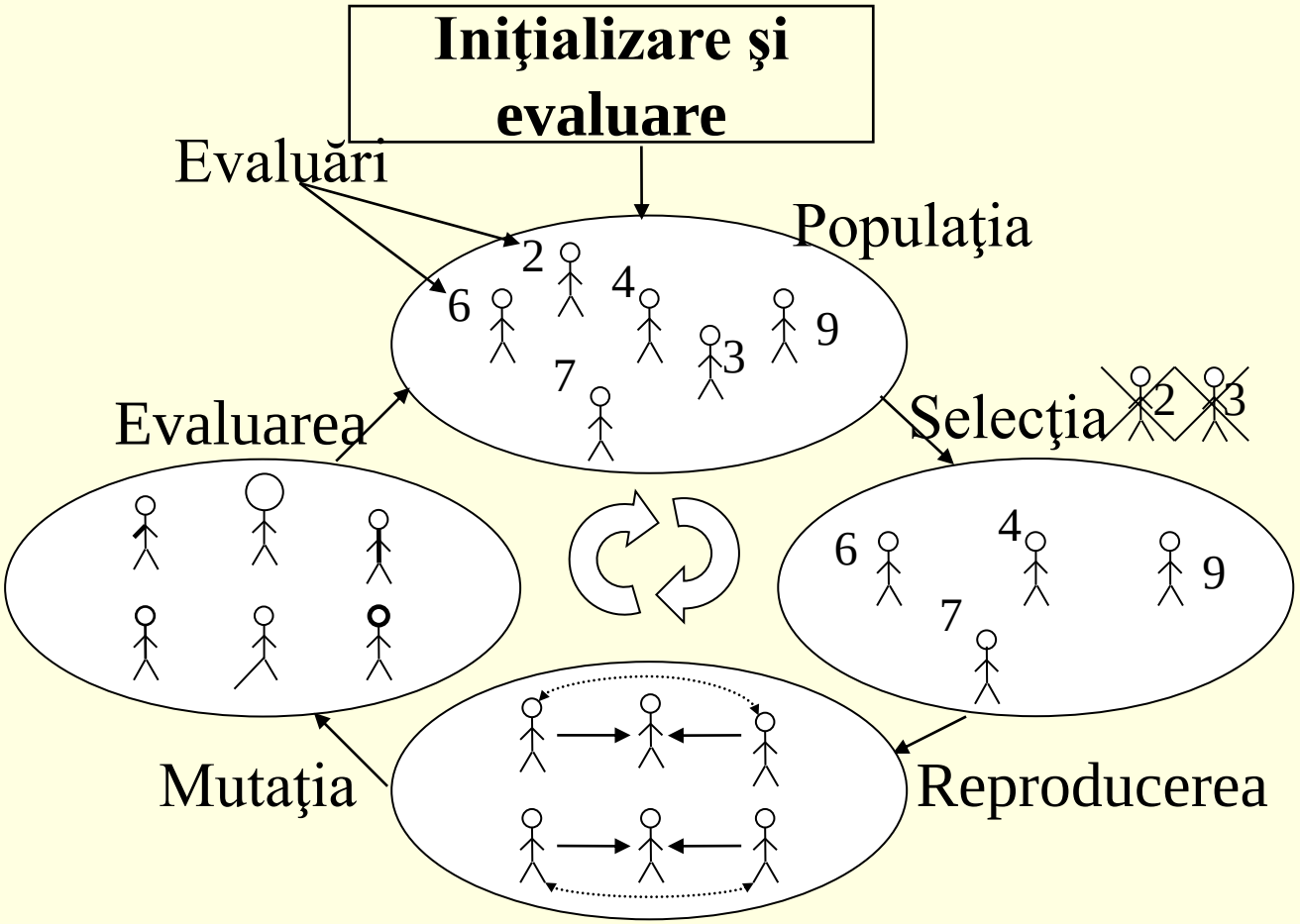


# Ce urmărim astăzi?

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

# Schema unui algoritm evolutiv



# Schema unui algoritm evolutiv

1. Se iniţializează populaţia cu indivizi generaţi aleatoriu;
2. Se evaluează fiecare individ;
3. **Cât timp** (condiţia de terminare nu este satisfăcută) **execută**
   - 3.1 Se selectează părinţii;
   - 3.2 Se recombină perechi de părinţi;
   - 3.3 Se aplică mutaţia asupra descendenţilor obţinuţi după recombinare;
   - 3.4 Se evaluează fiecare descendent;
   - 3.5 Se selectează indivizii care vor forma următoarea generaţie;
4. **Sfârşit cât timp**

# Componentele unui algoritm evolutiv

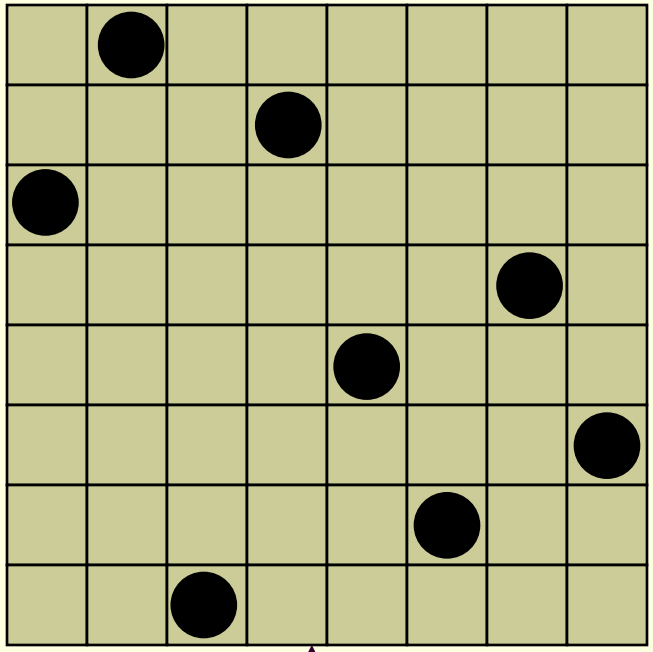
- Reprezentarea
- Evaluarea
- Populaţia
- Selecţia părinţilor
- Operatorii variaţionali
  - Recombinarea
  - Mutaţia
- Iniţializarea şi terminarea

# Reprezentarea

- **Soluţiile candidat** ale problemei de rezolvat trebuie reprezentate (codificate) sub formă de **indivizi** (**cromozomi**).
- Fiecare cromozom conţine **gene**.
- Orice element din spaţiul soluţiilor trebuie să aibă corespondent în spaţiul cromozomilor şi invers.
- Pentru fiecare problemă trebuie aleasă o reprezentare cât mai potrivită.

# Reprezentarea pentru problema damelor

Potenţială soluţie:
o configuraţie a celor 8 dame



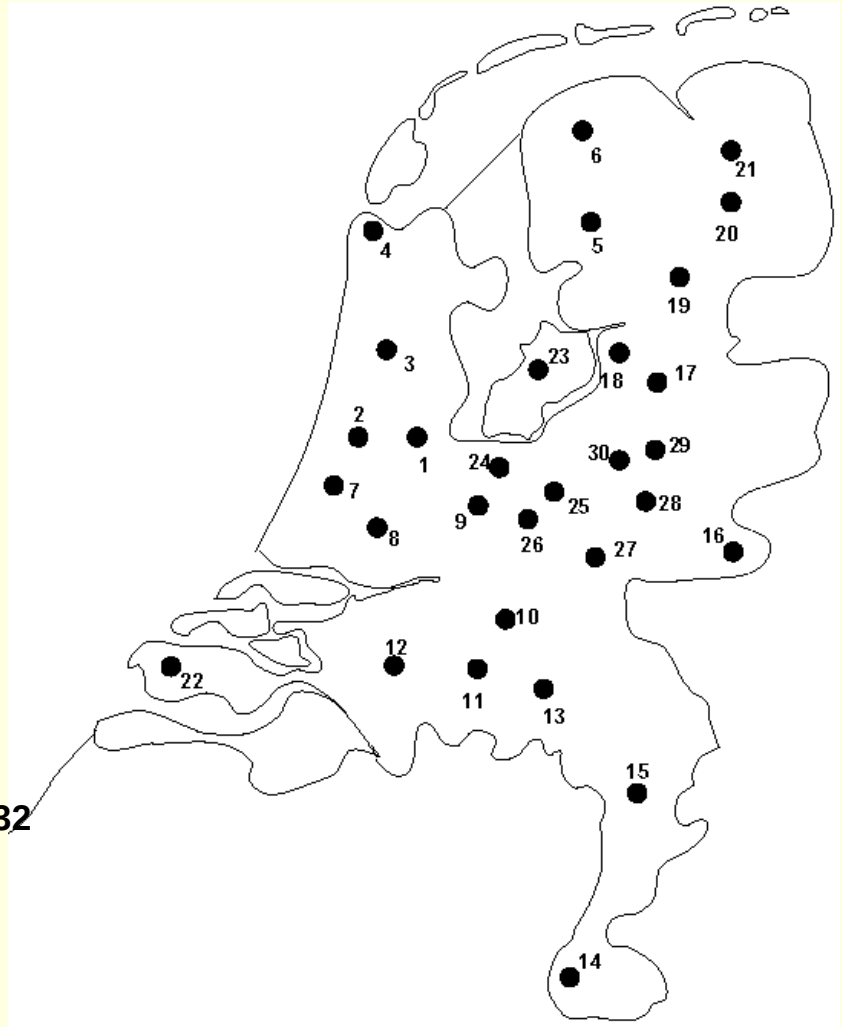
Codificare

Reprezentare:
o permutare a primelor 8 cifre.

| 3 | 1 | 8 | 2 | 5 | 7 | 4 | 6 |
|---|---|---|---|---|---|---|---|

# Problema comis voiajorului

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



# Reprezentarea binara pentru problema rucsacului

- Capacitate rucsac: 30
- 10 obiecte: 4, 3, 9, 17, 8, 15, 6, 20, 13, 7
- O solutie candidat contine 10 pozitii:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **1** |

- Fiecare valoare precizeaza daca obiectul este adaugat sau nu in rucsac.

# Reprezentare pentru optimizarea unei funcţii (găsirea maximului pe un interval)

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

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

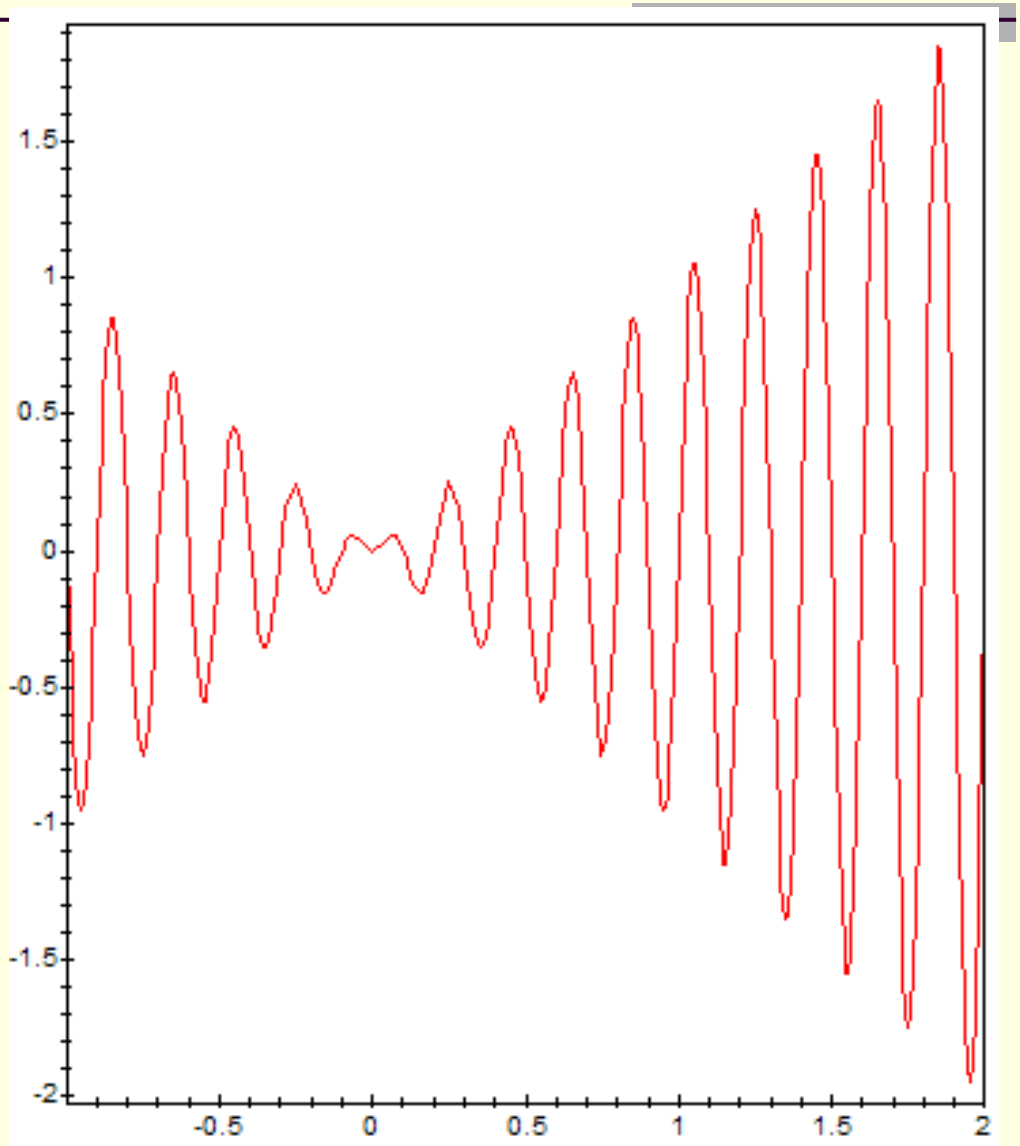
- Găsiţi-l pe $x$ în intervalul [-1, 2] astfel încât $z$ să fie cât mai mare (ca să câştigaţi cât mai multe puncte)!
- Altfel spus, găsiţi $x_0$ din [-1, 2] cu proprietatea că

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

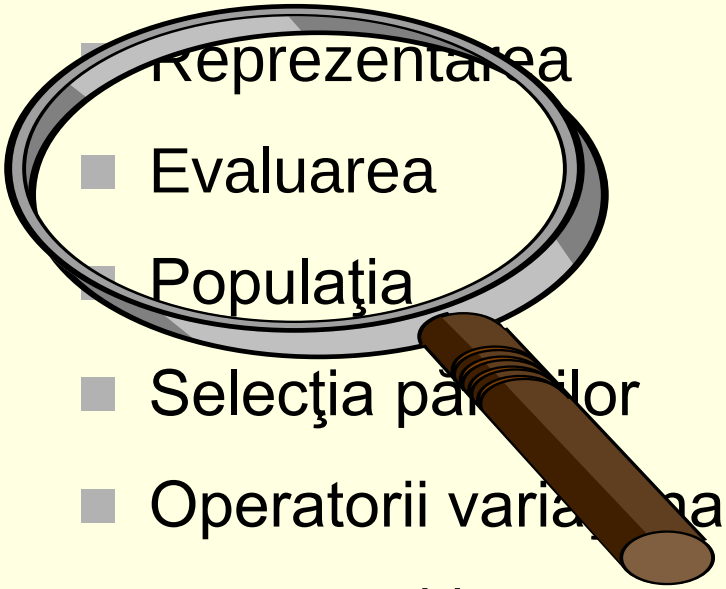
- In acest caz, spaţiul soluţiilor se poate identifica cu spaţiul cromozomilor.

# Reprezentarea funcţiei

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



# Componentele unui algoritm evolutiv



- Reprezentarea
- Evaluarea
- Populaţia
- Selecţia părinţilor
- Operatorii variaţionali
  - Recombinarea
  - Mutaţia
- Iniţializarea şi terminarea

# Funcţia obiectiv

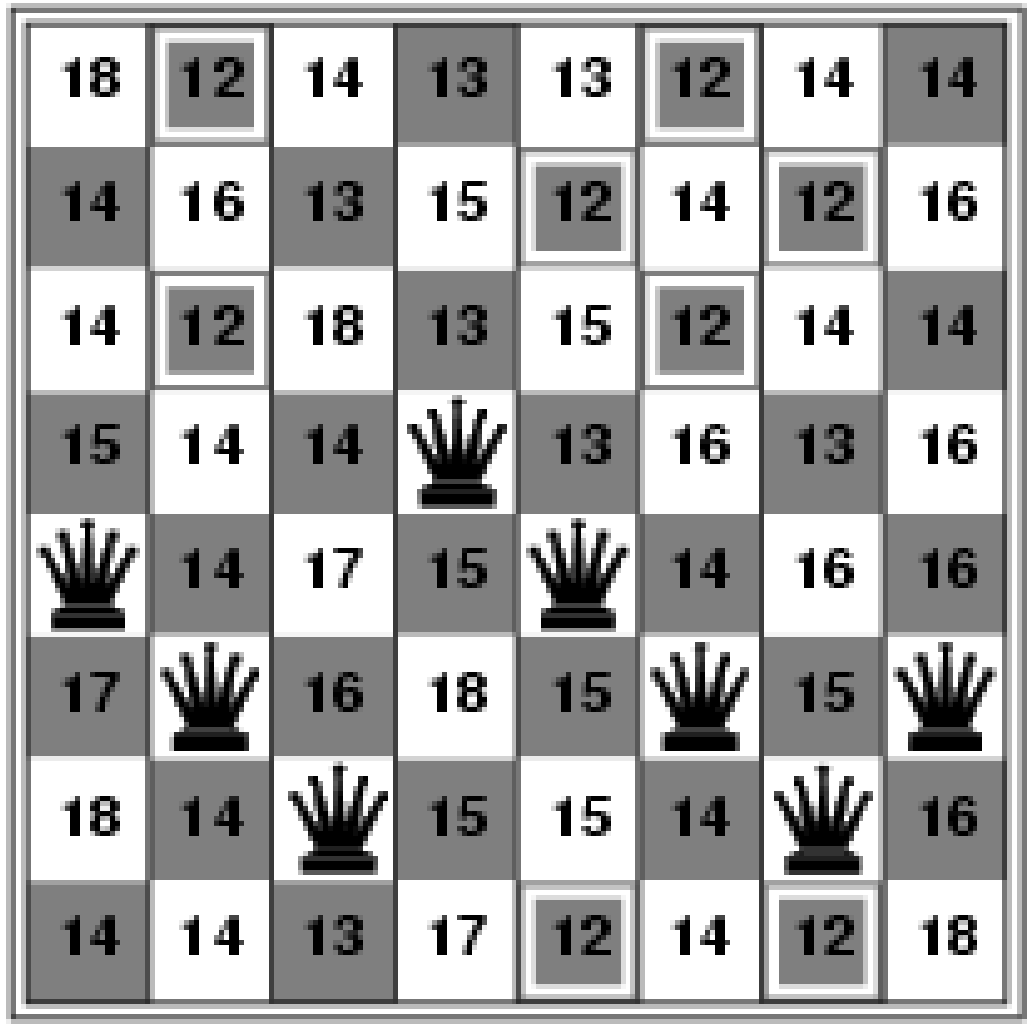
- Înainte de a defini funcţia de performanţă, trebuie stabilit **obiectivul problemei**, care este sarcina care trebuie îndeplinită. Este vorba de găsirea **funcţiei obiectiv**.
- În cazul problemei comis-voiajorului, obiectivul se referă la **minimizarea** drumului pe care îl parcurge comis-voiajorul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

# Funcţia de evaluare (performanţă)

- Rolul **funcţiei de performanţă** (sau de *evaluare*, ori *de calitate*) este de a măsura cât de bine se adaptează indivizii la mediu.
- Pentru unele probleme poate fi aceeaşi cu funcţia obiectiv (de exemplu, în optimizarea de funcţii).
- Funcţia de evaluare atribuie fiecărui individ o valoare (de obicei, reală) care reprezintă cât de bun (de performant) este cromozomul respectiv.
- Se doreşte maximizarea performanţei indivizilor, ceea ce duce de obicei la dorinţa de a maximiza valorile funcţiei de performanţă (nu în tot timpul este cazul însă, vezi funcţia de performanţă la problema damelor).

# Funcţia de performanţă pentru problema damelor

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



# Evaluarea pentru problema rucsacului

- Capacitate rucsac: 30
- 10 obiecte: 4, 3, 9, 17, 8, 15, 6, 20, 13, 7
- O solutie candidat contine 10 pozitii:

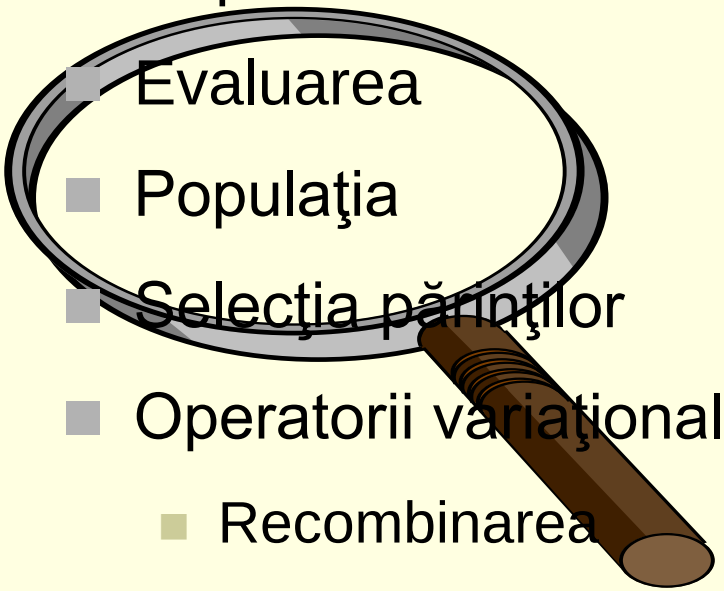
| **0** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **1** |
|---|---|---|---|---|---|---|---|---|---|
| 4 | 3 | 9 | 17 | 8 | 15 | 6 | 20 | 13 | 7 |

- Evaluarea ar putea fi:

$$|30 – (3 + 8 + 6 + 7)| = 6$$

# Componentele unui algoritm evolutiv

- Reprezentarea
- Evaluarea
- Populaţia
- Selecţia părinţilor
- Operatorii variaţionali
  - Recombinarea
  - Mutaţia
- Iniţializarea şi terminarea

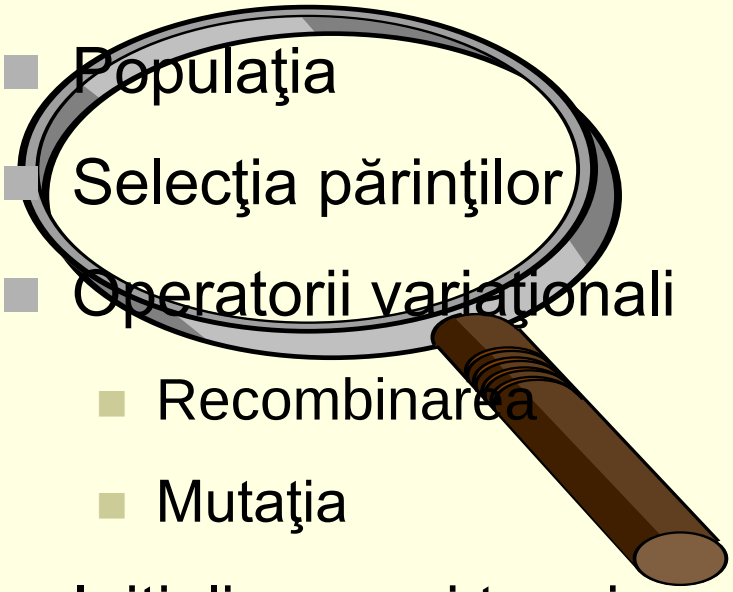


# Populaţia

- Populaţia constă dintr-o mulţime de indivizi care nu sunt neapărat toţi diferiţi între ei.
- **Mărimea populaţiei** reprezintă numărul de indivizi pe care îi conţine populaţia.
  - De obicei acest număr este constant.
- Operatorii de selecţie iau în considerare întreaga populaţie de la generaţia curentă.
- Populaţia este cea care evoluează de-a lungul generaţiilor, nu indivizii în particular.

# Componentele unui algoritm evolutiv

- Reprezentarea
- Evaluarea
- Populaţia
- Selecţia părinţilor
- Operatorii variaţionali
  - Recombinarea
  - Mutaţia
- Iniţializarea şi terminarea



# Selecţia

- Procesul de selecţie apare de două ori în cursul unei parcurgeri a ciclului *cât timp* a algoritmului evolutiv prezentat anterior.
- **Selecţia pentru reproducere** (selecţia părinţilor), când sunt aleşi părinţii generaţiei următoare.
- **Selecţia pentru înlocuire** (selecţia pentru supravieţuire), care apare când indivizii care vor forma populaţia din următoarea generaţie sunt aleşi dintre descendenţii obţinuţi şi indivizii din populaţia curentă.
- Modul în care creşte calitatea generală a soluţiilor depinde de ambele tipuri de selecţie.

# Selecţia pentru reproducere

- Selecţia pentru reproducere are rolul de a alege, în funcţie de calitatea indivizilor din populaţia curentă, care sunt cei consideraţi pentru a se aplica operatori de variaţie asupra lor în vederea obţinerii de noi soluţii candidat.
- Selecţia pentru reproducere este **probabilistă**
  - indivizii foarte performanţi au şanse bune de a fi selectaţi pentru reproducere
  - indivizii cu valori mici pentru funcţia de performanţă au şanse mici să devină părinţi.
- Natura *probabilistă* a selecţiei este cea care face căutarea să scape de optimele locale.

# Selecţia turnir

- Se selectează în mod aleatoriu *k* indivizi şi sunt evaluaţi.
- Cel mai bun dintre aceştia este *selectat* ca şi părinte pentru populaţia din generaţia următoare.
- Algoritmul de mai jos selectează *N* părinţi din generaţia curentă.

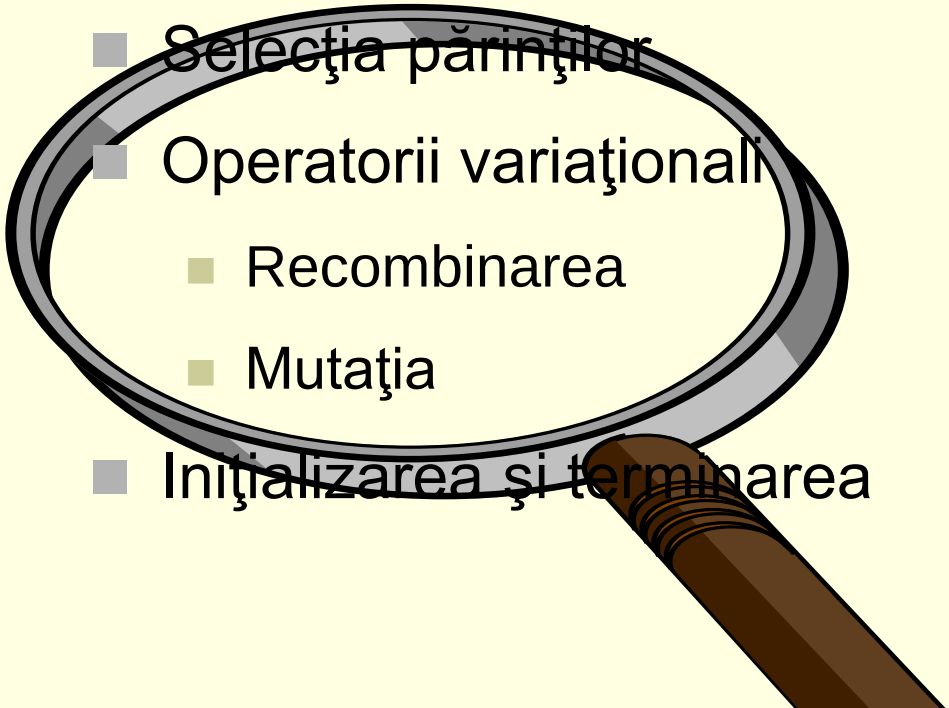
```
i = 0
Cât timp i < N execută
    Selectează k indivizi în mod aleatoriu din întreaga populaţie
    Selectează-l pe cel mai performant individ s din cei k
    părinţi[i] = s
    i = i + 1
Sfârşit cât timp
```

# Selecţia pentru supravieţuire

- Selecţia pentru supravieţuire decide care indivizi din populaţia curentă împreună cu descendenţii obţinuţi sunt opriţi pentru a forma populaţia generaţiei următoare.
- Cum numărul de indivizi din populaţie este de obicei constant, selecţia pentru supravieţuire intervine cu scopul de a păstra aceeaşi mărime a populaţiei.
- De obicei decizia depinde şi în acest caz de performanţa indivizilor, cei mai buni fiind favorizaţi.

# Componentele unui algoritm evolutiv

- Reprezentarea
- Evaluarea
- Populaţia
- Selecţia părinţilor
- Operatorii variaţionali
  - Recombinarea
  - Mutaţia
- Iniţializarea şi terminarea



# Operatorii de variaţie

- Operatorul de **selecţie** are rolul de a concentra căutarea pe regiunile cele mai promiţătoare din spaţiul de căutare.
- Rolul operatorilor de **variaţie** este de a crea noi soluţii candidat din cele vechi, de a mări diversitatea populaţiei.
- Operatorii de variaţie sunt dependenţi de reprezentarea utilizată, astfel că pentru reprezentări diferite trebuie definiţi operatori specifici.
- În funcţie de aritate, operatorii de variaţie sunt împărţiţi în:
  - Aritate = 1 => **Mutaţie**
  - Aritatea > 1 => **Recombinare**

# Mutaţia

- Acţionează asupra unui individ şi produce un altul.
- După ce se aplică asupra unui individ, rezultatul (descendent) conţine mici modificări faţă de individul iniţial.
- 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 mutaţie**.

# Cum se aplica mutaţia

- Presupunem că avem probabilitatea de mutaţie $p_m = 0.3$.
- Notăm mărimea populaţiei cu $N$.

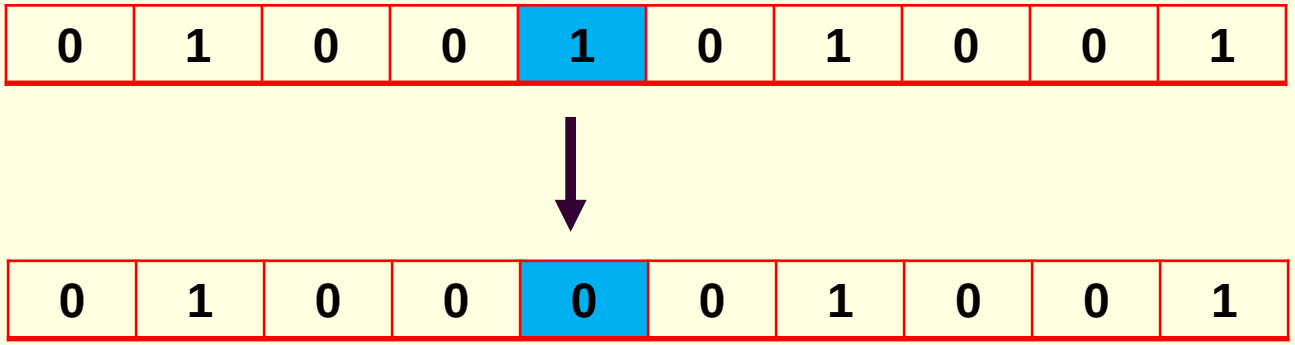
**Pentru** $i = 1$ **până la** $N$ **execută**
  **Pentru** $j = 1$ **până la** numărul de gene ale cromozomului **execută**
    $g$ = număr aleatoriu în intervalul [0, 1];
    Dacă $g < p_m$ atunci
      Aplică mutaţia pentru gena $j$ a individului $i$
    Sfârşit dacă
  Sfârşit pentru
Sfârşit pentru

# Mutaţia pentru problema damelor şi pentru comis-voiajor

- O mică variaţie într-o permutare:
  - Se aleg două valori în mod aleatoriu (5 şi 7 în imaginea din stânga).
  - Poziţiile celor două valori sunt interschimbate.

# Mutatia pentru problema rucsacului

- Pentru pozitia considerate a fi modificata, valoarea se transforma din *i* in 1 - *i*



# Mutaţia pentru reprezentarea reală

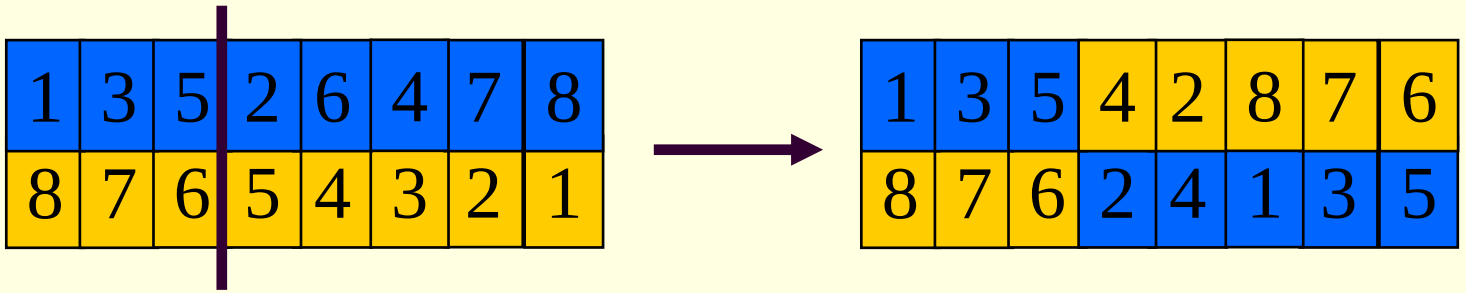
- În cazul optimizării unei funcţii, de exemplu, avem un individ $X = (x_1, x_2, ..., x_p)$, unde $x_i \in [a_i, b_i]$.
- Se poate obţine un individ $X' = (x'_1, x'_2, ..., x'_p)$, unde $x'_i$ este ales în mod aleatoriu în intervalul $[a_i, b_i]$.
- Exemplu:
  - $X$ = (0.4, 0.7, 0.3, 0.9) cu toate genele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

# Recombinarea

- Recombinarea sau încrucişarea implică doi sau mai mulţi indivizi (părinţi) aleşi cu o anumită **probabilitate de încrucişare** în scopul de a genera unul, doi sau mai mulţi indivizi prin combinarea genelor părinţilor.
- Recombinarea reprezintă un operator stochastic de vreme ce alegeri precum ce părţi să fie moştenite de la un părinte şi ce părţi de la alt părinte sau modul în care părţile acestea sunt combinate sunt făcute în mod aleatoriu.
- Prin împreunarea a doi indivizi cu caracteristici diferite, este obţinut un descendent (sau doi) care combină aceste caracteristici.

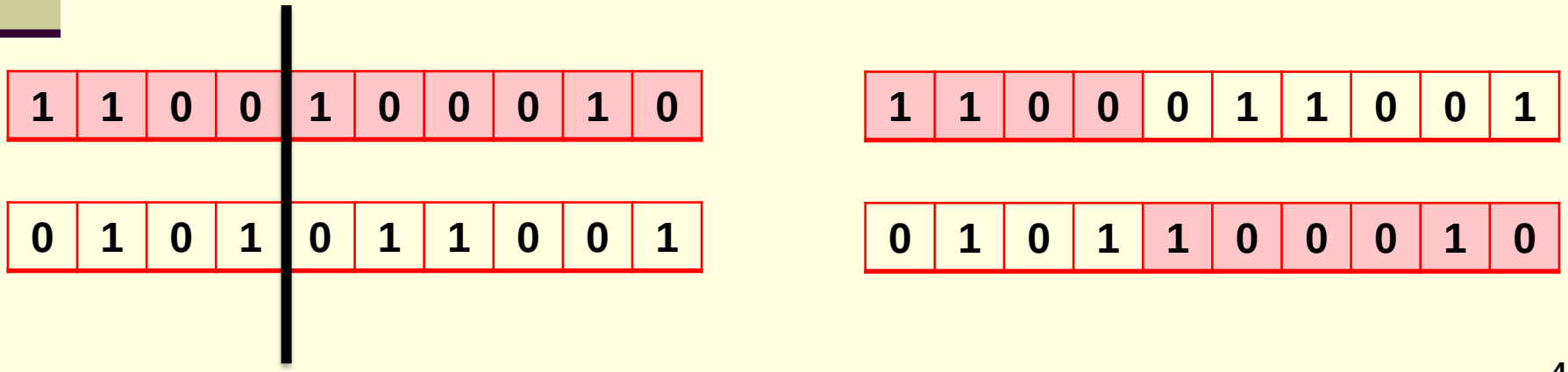
# Recombinarea pentru problema damelor şi pentru comis-voiajor

- Combinarea a două permutări în două noi permutări:
  - Se alege în mod aleatoriu un punct de încrucişare (linia verticală)
  - Se copiază primele două părţi în cei doi descendenţi
  - 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 problema rucsacului

- Combinarea a două permutări în două noi permutări:
  - Se alege în mod aleatoriu un punct de încrucişare/taietura (linia verticală)
  - Se copiază primele două părţi în cei doi descendenţi
  - A doua parte se completează prin inserarea de valori de la celălalt părinte:

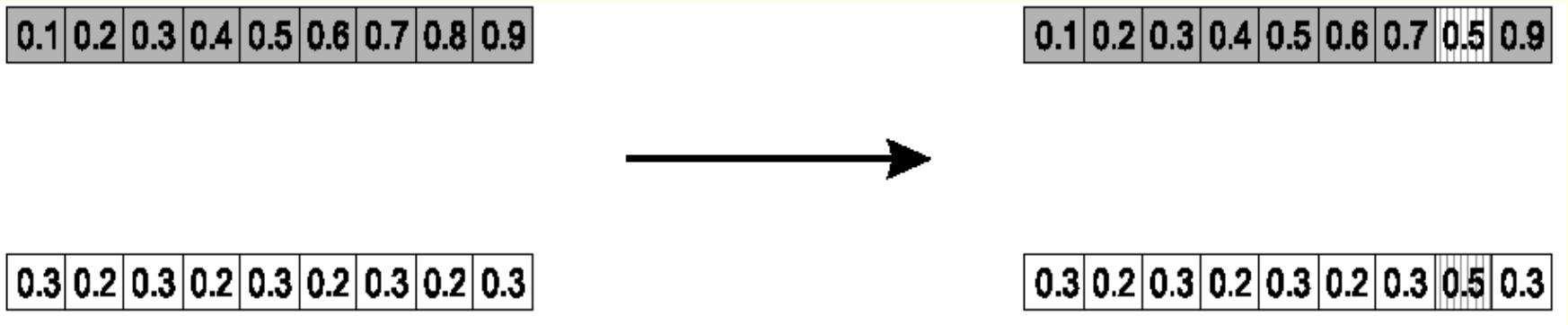


# Recombinarea pentru reprezentarea reală

- Discretă
  - Fiecare genă vine de la unul din părinţi cu probabilitate egală
- Intermediară
  - Exploatează ideea de a crea descendenţi între părinţi
  - $z_i = \alpha x_i + (1 - \alpha) y_i$ unde $\alpha$ : $0 \leq \alpha \leq 1$.
  - Parametrul $\alpha$ poate fi
    - Constant
    - Variabil (să depindă de vârsta populaţiei)
    - Ales aleatoriu de fiecare dată.

# Recombinarea pentru reprezentarea reală

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

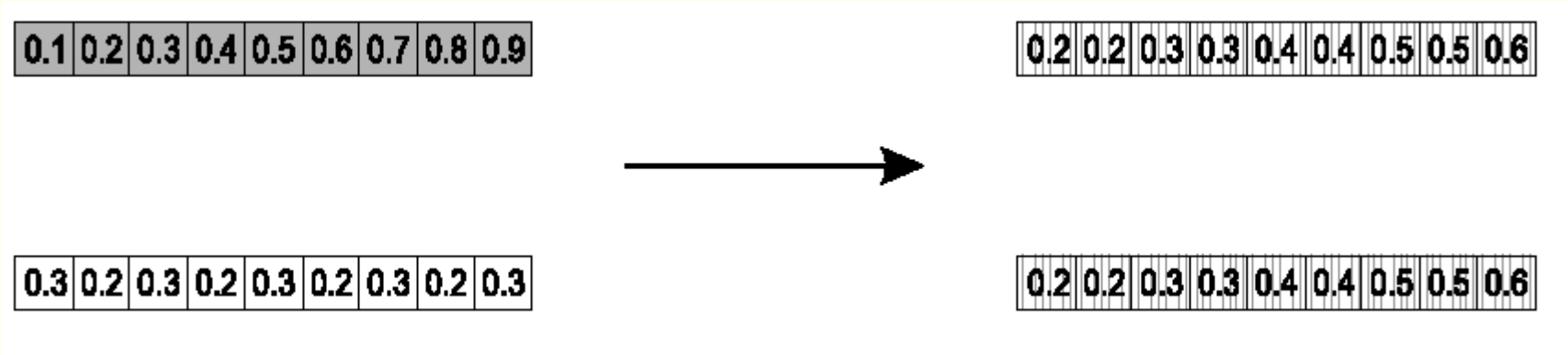


# Recombinarea cea mai folosită pentru reprezentarea reală

- Părinţii: $(x_1,\ldots,x_n)$ şi $(y_1,\ldots,y_n)$
- Fiecare genă $x'_k$ se calculează după formula

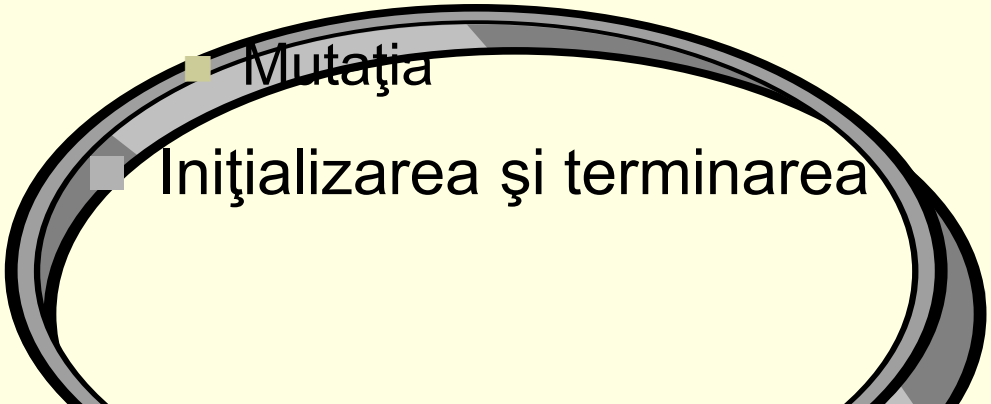
$$x'_k = \alpha y_k + (1 - \alpha)\, x_k$$

- Invers pentru celălalt descendent.
- La exemplul de mai jos, pentru $\alpha = 0.5$...



# Componentele unui algoritm evolutiv

- Reprezentarea
- Evaluarea
- Populaţia
- Selecţia părinţilor
- Operatorii variaţionali
  - Recombinarea
  - Mutaţia
- Iniţializarea şi terminarea

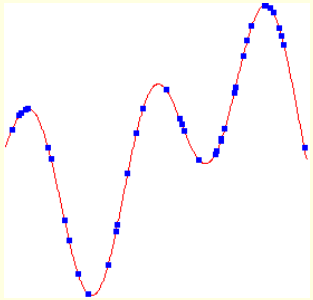


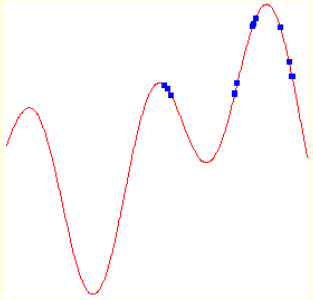
# Iniţializarea şi condiţia de terminare

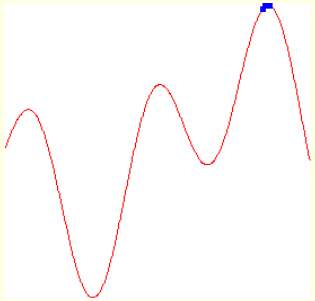
- De obicei **iniţializarea** se face în mod **aleatoriu**.
  - În populaţia iniţială se pot include şi soluţii existente.
- Condiţia de terminare se verifică la fiecare generaţie
  - Atingerea unei anumite performanţe
  - Ajungerea la un anumit număr maxim de generaţii
  - Ajungerea la un nivel foarte mic de diversitate în populaţie
  - Atingerea unui anumit număr de generaţii fără să se mai fi câştigat performanţă.

# Comportamentul tipic al unui algoritm evolutiv

Situaţii în optimizarea unei funcţii de o singură variabilă







**Fază de început:**

Distribuţia populaţiei este qvasi-aleatoare

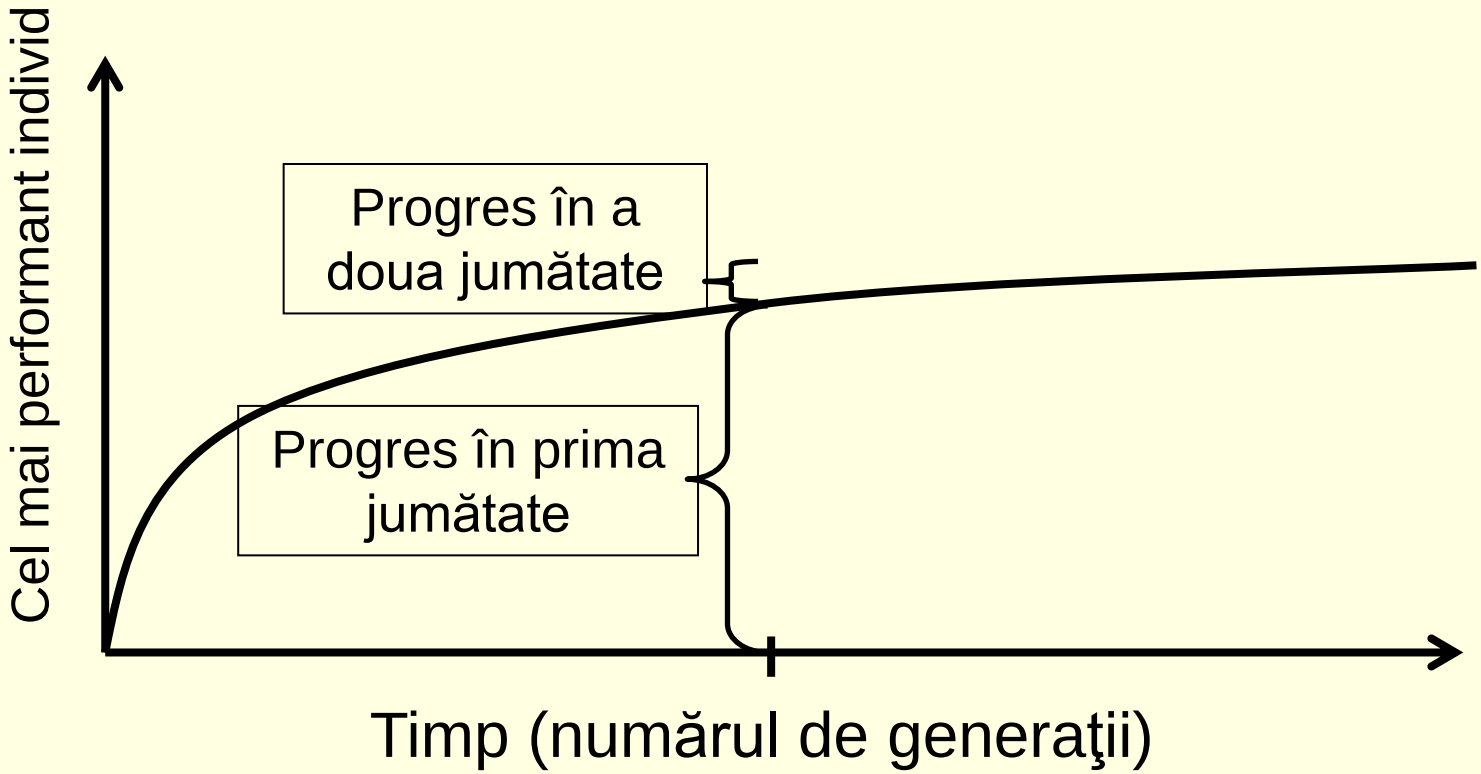
**Fază de mijloc:**

Populaţia aranjată în jurul / pe dealuri

**Fază de final:**

Populaţia este concentrată pe dealurile înalte

# Cât să dureze o rulare?



# Recapitulare

- Algoritmii evolutivi sunt inspiraţi din modul în care au evoluat lucrurile în natură.
- Folosesc mulţimi de potenţiale soluţii pe care le evoluează de-a lungul mai multor generaţii.
- Operatorul de **selecţie** este cel care face ca performanţa medie a populaţiei să crească o dată cu trecerea generaţiilor.
- Pentru introducerea de noi soluţii sunt folosiţi **operatori de variaţie** precum **mutaţia** şi **recombinarea**.