

Calcul neuronal și evolutiv.

Lab 6: Strategii evolutive. Programare genetică. Antrenarea evolutivă a rețelelor neuronale.

1. Implementarea strategiilor evolutive

În cazul strategiilor evolutive elementele populației sunt vectori cu valori reale iar elementele specifice (care îi diferențiază de algoritmi genetici) sunt:

- *Selecție:* se folosește de regulă doar pentru alegerea supraviețuitorilor și este o selecție deterministă bazată pe alegerea celor mai bune elemente din populația copiilor (în variantele (M,L)) sau din populația reunită a părinților și copiilor (în variantele (M+L)). În notația tradițională M reprezintă numărul de părinți iar L reprezintă numărul de urmași.
- *Recombinare:* cel mai frecvent se folosește recombinația convexă bazată pe selecția aleatoare a R părinți și construirea unui urmaș.
- *Mutație:* se adaugă o valoare aleatoare (generată în conformitate cu o anumită repartiție) fiecărui element al populației.

Aplicație 1. Minimizarea unei funcții. Funcții de test http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page364.htm

Indicație: vezi SE.sci

Exerciții:

1. Testați algoritmul pentru funcțiile Rastrigin, Ackley și Rosenbrock (de la pagina web menționată mai sus)
2. Analizați influența parametrului sigma asupra comportării algoritmului
3. Analizați influența tipului de selecție (prin trunchiere sau de tip turneu) asupra comportării algoritmului

2. Programare genetică.

Programarea genetică permite proiectarea în manieră evolutivă a unor structuri destinate rezolvării unei probleme (expresii aritmetice, expresii logice, reguli sau chiar programe). Elementele populației au de regulă o structură ierarhică (arbori).

Operatorii genetici (încrucișare și mutație) sunt specifici reprezentării ierarhice, iar una dintre problemele cele mai importante se referă la complexitatea structurilor obținute în procesul de evoluție. Ca atare, atât la inițializarea populației cât și pe parcursul aplicării operatorilor se urmărește ca arborii să nu depășească o anumită “dimensiune” (măsurată prin adâncimea arborelui = numărul de nivele în arbore).

Una dintre cele mai sugestive aplicații ale programării genetice este *regresia simbolică* al cărui scop este determinarea unei expresii care fitează un set de date (relație funcțională care descrie

legătura dintre un set de valori corespunzătoare unei mărimi dependente și un set de valori corespunzătoare unei mărimi independente).

Aplicație 2a. Să se testeze funcționarea applet-ului pentru regresie simbolică folosind programare genetică accesibil la <http://www.geneticprogramming.org/symbolic/main.htm>

Etape:

- *Alegerea funcției de test (Function Settings):* se specifică domeniul de definiție (**Min X**, **Max X**), funcția care va fi aproximată (**Enter Function**) și numărul de valori ale funcției folosite în procesul evolutiv (**Number of points**)
- *Setarea parametrilor algoritmului:* număr generații, dimensiunea populației, adâncimea maximă a arborilor utilizați pentru specificarea expresiilor care fac parte din populația inițială, fracțiunea de elemente care participă la încrucișare, fracțiunea de elemente care sunt supuse mutației, adâncimea maximă a arborilor construiți prin mutație, adâncimea maximă a subarborilor folosiți în procesul de mutație.
- *Alegerea modului de inițializare a populației:*
 - **Full:** se generează noi noduri până când toate ramurile din arbore au lungimea egală cu cea maximă.
 - **Grow:** extinderea unei ramuri în arbore se oprește fie când s-a atins lungimea maximă fie la realizarea unui eveniment aleator.
 - **Ramped half-half:** jumătate dintre elementele populației sunt generate după modelul **Full**, iar cealaltă jumătate după modelul **Grow**.
- *Alegerea metodei de selecție:*
 - Proportională
 - De tip turneu
- *Alegerea setului de neterminale* (prin marcarea din setul de operatori/funcții afișate: +, -, *, /, sin, cos, exp, abs, max, min, log)
- *Alegerea setului de terminale* (variabila x și constante aleatoare)

Aplicație 2b. Să se determine expresia care aproximează cel mai bine un set de date (regresie simbolică) folosind pachetul “rgp” din R.

Etape:

- Lansare R
- Incărcare pachet “rgp”: `Packages -> Load package ...` sau `library(“rgp”)` (dacă pachetul nu apare în lista de pachete instalate trebuie instalat prin `Packages-> Install package(s)...`)
- Definirea setului de neterminale (operatori și funcții) prin `functionSet`. Exemplu: `setNeterminale <- functionSet("+", "*", "-", "/")`
- Definirea setului de variabile prin `inputVariableSet`. Exemplu: `setVariabile <- inputVariableSet("x")`
- Definirea setului de constante prin `constantFactorySet`. Exemplu: `setConstante <- constantFactorySet(function() rnorm(1))` (valori generate folosind repartiția normală standard)
- Definirea datelor de test: valori care vor fi utilizate pentru a evalua calitatea aproximării. Exemplu: `dateX <- seq(from = -pi, to = pi, by = 0.1)`
- Definirea funcției scor: eroarea medie pătratică (măsură a diferenței dintre valorile funcției de test și valorile corespunzătoare expresiilor din cadrul populației). Exemplu: `functieScor <- function(f) rmse(f(dateX), sin(dateX))`
- Apelul funcției care simulează procesul evolutiv (funcția `geneticProgramming`). Exemplu: `geneticProgramming(functionSet = setNeterminale, inputVariables =`

```
setVariabile.constantSet = setConstante, fitnessFunction = functieScor, stopCondition =  
makeStepsStopCondition(10000))
```

Particularități ale variantei de programare genetică din pachetul “rgp”:

- Elementele populației sunt expresii R (care sunt implementate ca structuri arborescente)
- Inițializarea populației se bazează pe strategiile de construire:
 - “grow” (extinderea unei ramuri din arbore continuă fie până la atingerea adâncimii maxime fie până la realizarea unui eveniment aleator)
 - „full” (toate ramurile din arbore au lungimea maximă)
 - Varianta combinată (unele elemente sunt construite folosind strategia “grow”, altele folosind strategia „full”)
- Sunt implementați operatorii tradiționali de încrucișare și mutație la nivel de arbori (vezi slide-uri curs 9)
- Sunt implementate strategii de selecție care folosesc unul sau mai multe criterii (specific optimizării multicriteriale). În varianta multicriterială se urmărește optimizarea simultană a calității și simplității elementelor precum și a diversității populației.

Exercițiu: Parcurgeți etapele descrise mai sus și testați influența setului de parametri asupra calității rezultatului. Indicație: [gp1.r](#)

3. Proiectarea evolutivă a rețelelor neuronale

Algoritmii evolutivi pot fi utilizați în contextul proiectării rețelelor neuronale pentru:

- Determinarea ponderilor (în locul algoritmilor de antrenare bazați pe metode de optimizare locală se folosește o strategie evolutivă)
- Stabilirea arhitecturii

Aplicație 3. Determinare evolutivă a ponderilor unei rețele neuronale

Considerăm problema antrenării unei rețele neuronale pentru reprezentarea disjuncției exclusive. Se consideră o rețea constituită din:

- 3 unități de intrare (două efective și una corespunzătoare pragului)
- K unități ascunse (K este un parametru de intrare) cu funcție de transfer de tip Heaviside
- o unitate de ieșire cu funcție de transfer de tip Heaviside

Fiecare element din populație conține $4 \cdot K + 1$ elemente corespunzătoare tuturor ponderilor și valorilor prag. Criteriul de optimizat este eroarea medie pătratică pe setul de antrenare iar algoritmul utilizat poate fi o strategie evolutivă.

Indicație. Se adaugă la aplicația de la strategii evolutive o funcție pentru evaluarea erorii medii pătratice în cazul unei rețele cu arhitectură specificată (vezi [SEreteea.sci](#)).

Temă.

1. Să se modifice strategia evolutivă din SE.sci prin includerea auto-adaptării (variantea din cursul 8)
2. Extindeți aplicația cu antrenarea evolutivă a unei rețele neuronale (pentru reprezentare XOR) pentru cazul în care numărul de unități ascunse este variabil (elementele populației vor conține atât numărul de unități ascunse cât și valorile ponderilor – prin urmare

elementele populației vor avea număr variabil de componente). Observație: e necesară adaptarea operatorilor de recombinare și mutație