G. Ģenētiskie algoritmi un neironu tīkli

G1. Ģenētiskie algoritmi

G1.1. Vispārīgs apraksts

Ģenētiskie algoritmi (*genetic algorithms*, *GAs*) ir problēmu risināšanas stratēģija, kas balstās bioloģiskās evolūcijas principiem (paaudžu nomaiņa, konkurence starp populācijas indivīdiem).

Ģenētiskie algoritmi ir metožu kopums, lai optimizācijas un meklēšanas problēmām atrastu pareizus vai tuvinātus risinājumus.

Ģenētiskie algoritmi izmanto tādas no bioloģijas aizgūtas metodes kā **iedzimtība** (inheritance), **mutācija** (mutation), **izlase** (selection) un **krustošana** (crossover).

Ģenētiskie algoritmi tiek veidoti kā datorsimulācija, kurā kādas optimizācijas problēmas potenciālo risinājumu reprezentāciju kopums (saukts par **populāciju**, *population*), attīstas labāku risinājumu virzienā.

Potenciālos risinājumus, kas veido populāciju, sauc par **indivīdiem** (*individuals*), bet to abstraktās reprezentācijas, kas tiek izmantotas aprēķinos – par **hromosomām** (*chromosomes*).

Parasti evolucionārais process (sk. pseidokodu attēlā G1) sākas ar populāciju no nejauši ģenerētiem indivīdiem un turpinās paaudzi pēc paaudzes (*generation*).

Katras paaudzes laikā notiek šādas darbības:

- visu populācijas indivīdu novērtēšana, izmantojot speciālu **derīguma funkciju** (*fitness function*),
- indivīdu kopuma **izvēle** (*selection*) jaunas paaudzes veidošanai,
- izvēlēto indivīdu apstrāde (**ģenētisko operatoru** (*genetic operators*) pielietošana), lai iegūtu jaunu paaudzi.

Divas nedaudz atšķirīgas ģenētiskā algoritma shēmas parādītas attiecīgi attēlos G1 un G2. Atšķirības balstās uz izvēles (izlases) veidu (sk. sadaļu G1.3 zemāk) un algoritma tehniskās interpretācijas.

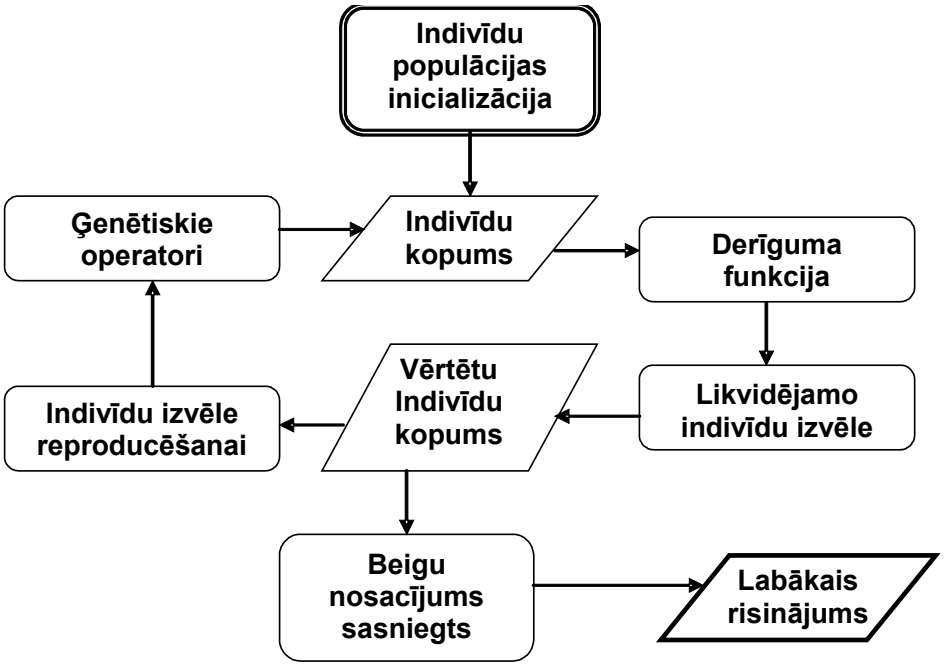
Shēmā #1 (attēls G1) operators var būt arī tukšs, kas nozīmē izmaiņu neveikšanu indivīdam. Izvēle atbilst paaudžu nomaiņas shēmai (*generational selection*).

Shēmā #2 (attēls G2) izvēles likvidēšanai un reprodukcēšanai ir nodalītas un daļēji atbilst statiskajai izvēles shēmai (*steady-state selection*).

```
PROCEDURE GeneticAlgorithm
P():Population
BEGIN
```

```
  t:=0
  INITIALIZE(): P(t):={I1,...In}
  REPEAT
    EVALUATE(P(t)): {(I1,f1),...,(In,fn)}
    P(t+1):={}
    FOR j:=1 TO n
      SELECTINDIVIDUAL(Iold,P(t))
      SELECTOPERATOR(op)
      Inew:=MUTATE(Iold,op,P(t))
      P(t+1):=P(t+1) UNION {Inew }
    END FOR
    t:=t+1
  UNTIL Termination Condition Is Met
END
```

Attēls G1. Ģenētiskā algoritma shēma #1



Attēls G2. Ģenētiskā algoritma shēma #2

G1.2. Derīguma funkcija

Derīguma funkcija (*fitness function*) ir metode indivīda novērtēšanai. Derīguma funkcijas dotais rezultāts ir pamatā indivīdu izvēlei, kas nodrošina ĢA darbību.

Derīguma funkcija netieši nosaka prasību pret problēmu klasi, kuru var risināt ar ĢA – tās risinājumus ir jāspēj novērtēt jebkurā gatavības pakāpē. Ja tas nav iespējams vai izdarāms pārāk sarežģīti, ģenētiskie algoritmi nebūtu īstā risināšanas stratēģija.

Tehniski derīguma funkciju bieži realizē kā "sodīšanas" funkciju (*penalty function*), t.i., mazāka vērtība nozīmē augstāku vērtējumu.

G1.3. Izvēle

Ir daudz dažādu metožu indivīdu izvēlei nākošajām paaudzēm. Dažas no tām ir savstarpēji izslēdzošas, bet dažas ir lietojamas kombinēti.

Elitisms (*elitist selection*).
Elitisms nozīmē to, ka labākajiem (derīgākajiem) tiek garantēta to pārkopēšana nākamajā paaudzē. Vairums ĢA realizāciju tīrā veidā neizmanto elitismu, bet izmanto tā modificēto variantu.

Derīgumam proporcionāla izvēle (*fitness-proportionate selection*).
Derīgākie indivīdi tiek izvēlēti ar lielāku varbūtību.

Ruletes izvēle (*roulette-wheel selection*).

Derīgumam proporcionālās izvēles paveids, kas nosaka, ka indivīda izvēles iespēja ir proporcionāla lielumam, par kādu tā derīgums pārsniedz cita indivīda derīgumu. Konceptuāli ruletes izvēle līdzinās izspēlei ar ruletes riteni.

Mērogojamā izvēle (*scaling selection*).

Tā kā ģenētiskā algoritma darbības laikā vidējais indivīdu derīgums palielinās, tad palielinās arī derīguma funkcijas diskriminējošais raksturs. Šī izvēles metode varētu būt derīga ģenētiskā procesa vēlākās fāzēs, kad visiem indivīdiem ir relatīvi augsts derīgums un tikai nelielas atšķirības starp tiem nosaka izvēli.

Turnīra izvēle (*tournament selection*).

Populācija tiek sadalīta grupās, un indivīdi cīnās viens ar otru grupas ietvaros. Tikai viens indivīds no katras grupas tiek izvēlēts reproducēšanai.

Kārtas numura izvēle (*rank selection*).

Visi indivīdi pēc vērtēšanas tiek sakārtoti rindā pēc to derīguma, un izvēle ir atkarīgi no šī sakārtojuma, nevis no derīgumu absolūtajām vērtībām vai to atšķirībām.

Paaudžu nomaiņas izvēle (*generational selection*).
Tikai indivīdu pēcnācēji tiek iekļauti nākošajā paaudzē. Neviens indivīds nepāriet no paaudzes uz paaudzi.

Statiskā izvēle (steady-state selection).
Pēcnācēji daļēji aizvieto vecos indivīdus, tomēr daļa indivīdu pāriet no paaudzes uz paaudzi.

Hierarhiskā izvēle (*hierarchical selection*).

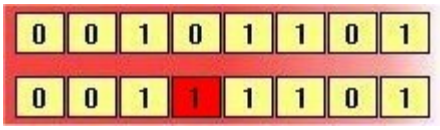
Katras paaudzes laikā indivīdi iziet daudzpakāpju (vairāku kārtu) izsijāšanu. Zemāka līmeņa izvēles ir ātrākas un mazāk diskriminējošas, bet tie indivīdi, kas izdzīvo līdz augstākiem līmeņiem, tiek vērtēti aizvien bargāk. Šīs metodes priekšrocība ir kopējā skaitļošanas laika samazināšanās uz tā rēķina, ka lielākā daļa indivīdu tiek atsijāti zemākajos līmeņos, kur tiek veikti vienkāršāki un ātrāki aprēķini.

G1.4. Ģenētiskie operatori

Ģenētiskie operatori (*genetic operators*) nodrošina jaunu (savādāku) indivīdu ģenerēšanu uz viena vai vairāku esošo indivīdu bāzes. Pazīstamākie ģenētiskie operatori ir mutācija (*mutation*) un krustošana (*crossover*).

Mutācija.

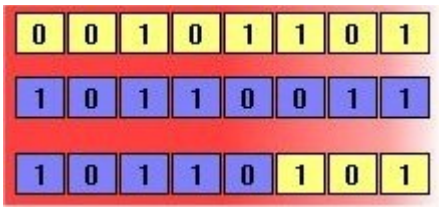
Mutācija ir vienkāršākais ģenētiskais operators, kas nosaka viena vai vairāku bitu (elementu) izmaiņu hromosomā.



Attēls G3. Mutācija.

Krustošana.

Krustošanā ir iesaistīts vairāk nekā viens sākotnējais indivīds, kuru fragmenti veido pēcnācēju. Parastā forma ir viena punkta krustošana (attēls G4), kas nodrošina, ka pēcnācējs būs savu vecāku garumā.



Attēls G4. Viena punkta krustošana.

G1.5. Risinājumu reprezentācija

Pirms ģenētiskais algoritms vispār var sākt darboties, lai risinātu kādu problēmu, ir nepieciešams nokodēt potenciālos risinājumus tāda formā, kādā tos būtu viegli apstrādāt ar datoru. Vienkāršākais variants ir kodēt katru risinājumu kā 1 un 0 virkni. Bet tikpat labi tos var kodēt kā dažādu citu skaitļu virknes, bet arī tā nav robeža, jo galvenais kritērijs ir viegla indivīdu tālāka apstrāde (ģenētisko operatoru pielietošana, vērtēšana).

G2. Ģenētisko algoritmu pielietojums neironu tīklu izstrādē

Ģenētisko algoritmu un neironu tīklu sadarbība parasti notiek šādās divās formās:

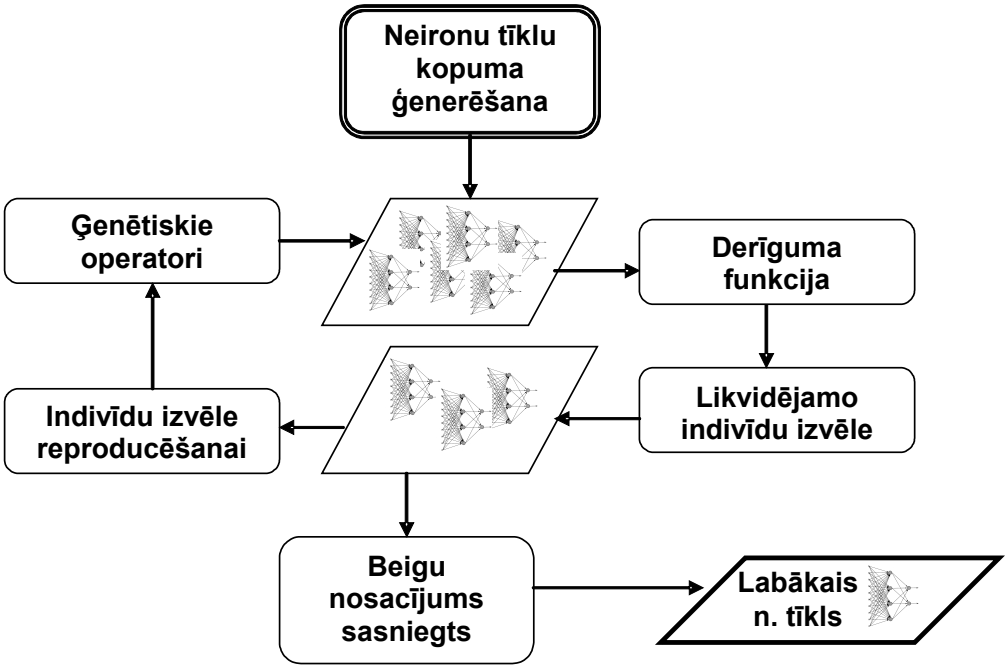
- Svaru optimizācija (alternatīva apmācības algoritmam),
- Topoloģijas un citu neironu tīkla modeļa parametru optimizācija.

G2.1. Ģenētisko algoritmu pielietošana svaru vērtību optimizācijai

Viens no tipiskiem ģenētisko algoritmu pielietojumiem neironu tīklos ir to izmantošana tiešā veidā neironu tīklu apmācībā. Daudzu tradicionālo apmācības metožu trūkums ir iekļūšana lokālajos minimumos. Tas ir viens no iemesliem ģenētisko algoritmu pielietojumam neironu tīklu apmācībā, ka ĢA nodrošina izkļūšanu no lokālajiem minimumiem.

Šim nolūkam neironu tīkla svari, kas ir ar reālām vērtībām, varētu būt jāreprezentē kādā binārā formā. Lielākā problēma ĢA lietojumam neironu tīklu svaru uzstādīšanā varētu būt tā, ka ĢA vajadzībām ir jānosaka maksimālā iespējamā svaru absolūtā vērtība, kas nav viegli izdarāms.

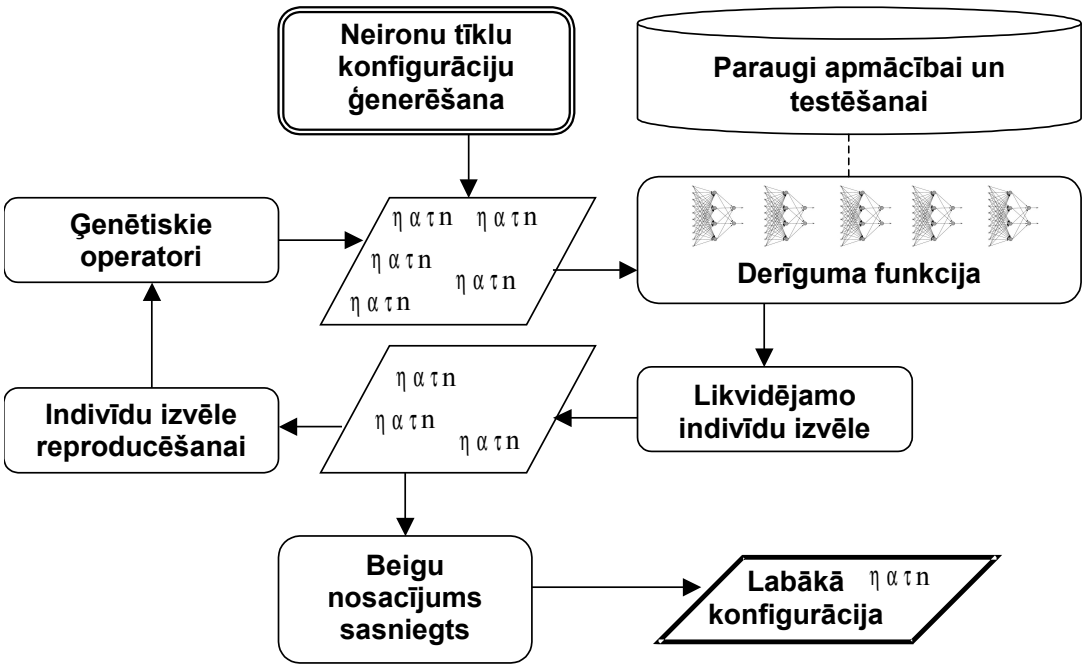
Ģenētisko algoritmu izmantošana ir viena no iespējām, bet ne tuvu ne vienīgā un ne vienmēr labākā. Jāatceras, ka neironu tīkla apmācība ar ĢA ir daudzkārt lēnāka par apmācību ar *backpropagation* algoritmu.



Attēls G5. Ģenētiskie algoritmi neironu tīklu apmācībā

G2.2. Ģenētisko algoritmu pielietošana neironu tīklu uzlabošanai modeļa līmenī

Ģenētiskie algoritmi būtu veiksmīgi izmantojami optimālākās neironu tīkla arhitektūras un citu parametru noskaidrošanai, lai iegūtu labāko neironu tīklu modeli dotās problēmu klases risināšanai. Izmantojot ĢA neironu tīklu arhitektūras optimizācijai, īpaši jāņem vērā potenciāli lielais aprēķinu apjoms, tādēļ jācenšas pēc iespējas samazināt arhitektūras un citu parametru variācijas, lai modeļa optimizācija notiktu saprātīgā laikā.



Attēls G6. Ģenētiskie algoritmi neironu tīklu konfigurāciju optimizācijā