

Szegedi Tudományegyetem
Informatikai Tanszékcsoport

SZAKDOLGOZAT

Danner Gábor

2012

**Szegedi Tudományegyetem
Informatikai Tanszékcsoport**

Evolúció-szimuláció NEAT alapokon

Szakdolgozat

Készítette:

Danner Gábor

Programtervező informatikus BSc
szakos hallgató

Témavezető:

Prof. Csirik János

egyetemi tanár

Szeged
2012

Feladatkiírás

A feladat egy evolúció-szimuláció készítése, melyben neuronhálók által irányított egyedek versengenek a túlélésért. Az egyedek genomja a neuronháló mellett fiziológiai jellemzőket is kódoljon. Az egyedek legyenek képesek mozogni, táplálkozni, harcolni és szaporodni.

Tartalmi összefoglaló

- ***A téma megnevezése:***

Evolúció-szimuláció NEAT alapokon

- ***A megadott feladat megfogalmazása:***

A szakdolgozat célja egy szimulációs program, a WOD elkészítése, melyben neuronhálók által irányított egyedek versengenek a túlélésért.

Az egyedek képesek mozgásra, táplálkozásra, harcra, szaporodásra és kommunikációra. A tevékenységek, valamint a pusztta létezés is energiát fogyaszt, melyet növények és/vagy elhullott egyedek elfogyasztásával tudnak fedezni. Harc révén képesek lehetnek vadászni, vagy akár nőstényekért vetélkedni. Az egyedek vezérlését egy-egy neuronháló végzi, mely csomópontokból (neuronok) és azokat összekötő, irányított, súlyozott kapcsolatokból áll. Egy egyed neuronhálóját és fizikai jellemzőit a genotípusa határozza meg, melyet képes átörökíteni utódainak.

- ***A megoldási mód:***

2D-s vizuális szimuláció valós időben.

- ***Alkalmazott eszközök, módszerek:***

Felhasznált technológiák: .NET Framework 4.0, Windows Presentation Foundation.

A fejlesztés Microsoft Visual Studio 2010 Ultimate-tel történt, C# nyelven.

A WOD neurális hálókön végzett manipulációi a NEAT módszerén alapulnak.

- ***Elért eredmények:***

- ***Kulcsszavak:***

Mesterséges intelligencia, mesterséges élet, neurális hálók, evolúció, szimuláció.

Tartalomjegyzék

Feladatkiírás.....	2
Tartalmi összefoglaló.....	3
Tartalomjegyzék	5
1. EVOLÚCIÓS SZÁMÍTÁSOK.....	7
1.1. Evolúciós stratégia	7
1.2. Evolúciós programozás.....	8
1.3. Genetikus algoritmus.....	8
1.4. Genetikus programozás.....	8
1.5. Stratégiai paraméterek.....	9
2. NEURÁLIS HÁLÓK.....	9
2.1. Neurális hálók topológiája	10
2.1.1. Előrecsatolt hálózatok	10
2.1.2. Egyszerű rekurrens hálózat	11
2.1.3. Teljesen rekurrens hálózat.....	11
2.1.4. Egyéb topológiák	11
2.2. A Hebb tanulás.....	11
3. NEUROEVOLÚCIÓ.....	12
3.1 A NEAT	12
4. MESTERSÉGES ÉLET	13
4.1 DigiHive	14
4.2 PolyWorld.....	15

5. A WOD	16
5.1 A szimuláció	16
5.2 A jellemzők közelebbről.....	17
5.x Fizikai jellemzők	17
5.x NEAT-től való fontosabb eltérések.....	19
5.x Input/Output	20
5.x A WOD-ról részletesebben.....	21
5.x.1 Aktivációs függvény	21
5.x.2 A bővítő strukturális mutációk implementációja	22
5.x.3 Az egyszerűsítő strukturális mutáció implementációja.....	22
5.x.4 A jellemzők mutációjának implementációja	23
5.x.5 A keresztezés implementációja	23
5.x.6 A hálózat-megjelenítő	23
Irodalomjegyzék	24
Nyilatkozat	25
Köszönetnyilvánítás.....	26

1. Evolúciós számítások

Az evolúciós számítások (Evolutionary Computation) evolúció-szimulációs módszereket magába foglaló terület. Ezek egy populáció-alapú, iteratív megközelítést alkalmaznak, ami véletlen variáción és kiválasztódáson alapul. [7]

A populáció egyedei pontoknak felelnek meg a keresési térben; más szóval, az egyedek egy-egy lehetséges megoldást kódolnak egy adott problémára. A populációból új egyedeket állíthatunk elő mutációval vagy keresztezéssel. A mutáció véletlenszerű, rendszerint kis mértékű változást eredményez. Keresztezéshez több (leggyakrabban kettő) egyedet veszünk alapul, és azokat kombináljuk. Egy alkalmassági függvény (fitness function) segítségével kiértékelhetővé válnak az egyedek. Ez alapján egy szelekciós operátor megadja, hogy mely egyedeket érdemes megtartanunk, szülőnek megválasztanunk az egyedek új generációjához.

A kezdeti populáció lehet véletlen, vagy esetleg kézzel megadott. Ezután, a kiértékelés, szelekció, reprodukció iterációjával a populáció evolúciója fog létrejönni. Ezzel a véletlen, irányított kereséssel egyre jobb megoldásokhoz jutunk, az által, hogy mindig kiszórjuk a rosszabb megoldásokat, a jobbakhoz pedig hozzájuk hasonló, új megoldásokat generálunk.

Az evolúció elvein alapuló algoritmust evolúciós algoritmusnak hívjuk. Klasszikus típusai:

- evolúciós stratégiák,
- evolúciós programozás,
- genetikus algoritmusok. [7]

1.1. Evolúciós stratégia

Általában valós-értékű paramétereket alkalmaz, habár diszkrét problémákra is használják.

Főbb jellemzői:

- megkülönböztetést tesz egy (μ méretű) szülő populáció és egy ($\lambda \geq \mu$ méretű) leszármazott populáció között,
- hangsúlyt helyez a normális eloszlású mutációkra,
- a rekombináció többféle formáját használja,

- alkalmazza az önadaptáció elvét. [7]

1.2. Evolúciós programozás

Evolúciós algoritmus, mely eredetileg idősor előrejelzési feladatokban hasznosítható véges automaták kifejlesztésére szolgált.

Később kiterjesztették paraméter-optimalizálási problémákra is.

Főbb jellemzői:

- általában nem használ rekombinációt,
- sztochasztikus verseny szelekciót (stochastic tournament selection) használ,
- alkalmazza az önadaptáció elvét. [7]

1.3. Genetikus algoritmus

Eredetileg adaptáció egy formális rendszerének volt tervezve, mintsem egy optimalizációs rendszernek.

Főbb jellemzői:

- erős hangsúly a rekombináción (keresztezés),
- probabilisztikus szelekciós operátor használata (a relatív rátermettségi értékkel arányos esély a kiválasztásra),
- a mutáció háttér operátorként való értelmezése, kisebb szerepet szánva neki az algoritmusban,
- míg a genetikus algoritmus eredeti formája esetén a megoldásokat bináris sztringek reprezentálják, számos további átalakítást is kifejlesztettek, hogy ne csak bináris keresési terekben legyen használható. [7]

1.4. Genetikus programozás

Genetikus algoritmusokból származtatták, evolúciós algoritmusok egy osztálya. Programok automatikus generálását célozza meg. A populáció minden egyede egy teljes programot reprezentál egy alkalmas programozási nyelven. Leggyakrabban a LISP nyelv egy szűkítésében megadott, elemzési fákot reprezentáló szimbolikus kifejezéseket használnak a programok megadására, de egyéb reprezentációkat és programozási nyelveket is sikeresen alkalmaztak erre a célra. [7]

1.5. Stratégiai paraméterek

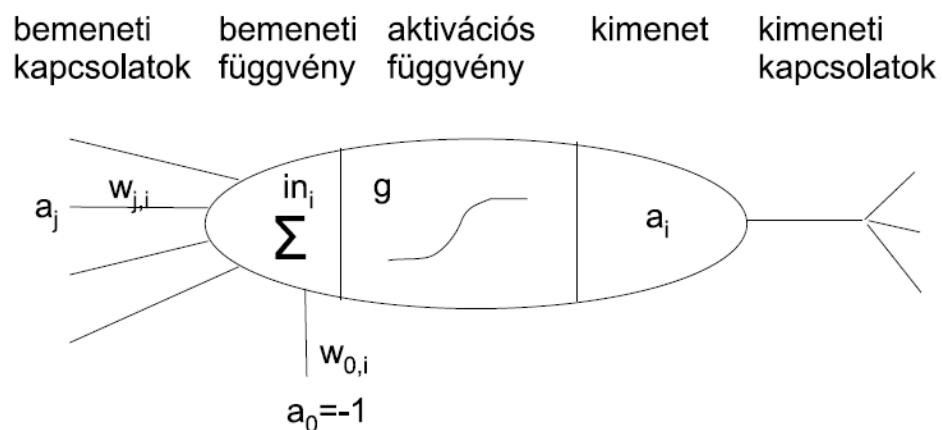
Egy evolúciós algoritmus vezérlő paramétereit (pl. mutáció mértéke, keresztezés gyakorisága) gyakran stratégiai paramétereknek hívjuk. Ezek megfelelő beállítása fontos az evolúciós algoritmus jól teljesítéséhez.

A szokásos technika, a paraméterek empirikus megkeresése, általában nem kielégítő.

Egyes kutatók ehelyett meta-evolúciós technikákkal próbálják optimalizálni a stratégiai paramétereket, azaz azok megállapítására egy újabb evolúciós algoritmust használnak.

A harmadik lehetőség az önadaptáció, melyet sikeresen alkalmaztak evolúciós stratégiák és evolúciós programozás során. Ebben az esetben a stratégiai paraméterek az egyed részét képezik, és a tárgy változókkal együtt mutálódnak, rekombinálódnak és választódnak ki. Ezáltal az evolúciós algoritmusok on-line, a keresés során tanulhatják meg a saját stratégiai paramétereiket. A természetben fellelhető az önadaptáció biológiai analógiája: a DNS kód egyes részei mutáló géneket, javító enzimeket írnak le, azaz a DNS befolyással rendelkezik a saját mutációs rátája felett. [7]

2. Neurális hálók



A mesterséges neurális hálók megpróbálják megvalósítani az agy adatfeldolgozó képességeit egy számítógépen. Hogy elérje ezt (legalább egy nagyon egyszerűsített formában, a feldolgozó egységek és kapcsolataik számát tekintve), egyszerű egységek (melyek neuronoknak felelnek meg) irányított, súlyozott kapcsolatokon (melyek axonoknak és dendriteknek felelnek meg) keresztül

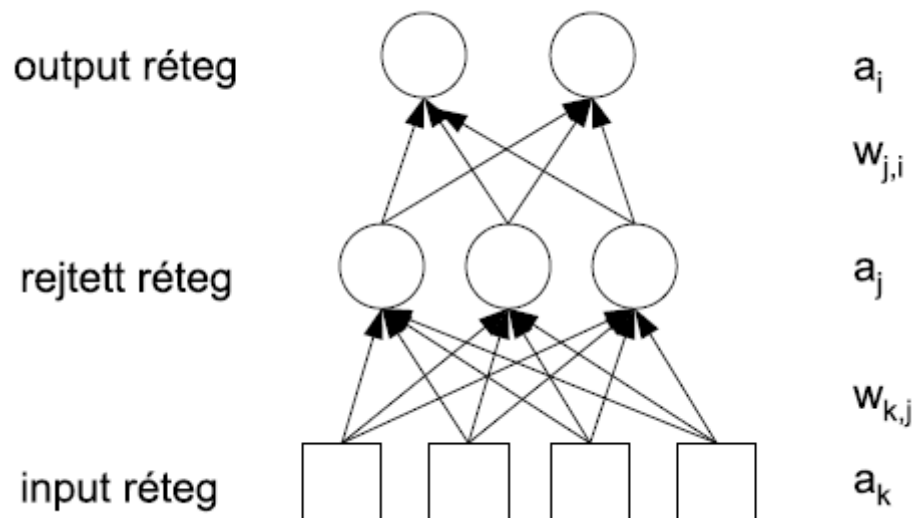
kommunikálnak. Párhuzamosan dolgozva, a hálózat minden egysége kiszámítja az inputjai súlyozott összegét, elvégzi az eredmény valamilyen leképezését (aktivációs függvény), és végül továbbítja az értéket a kimenő kapcsolatainak. Habár a mesterséges modell erős leegyszerűsítése a természetesnek, impresszív eredményeket értek el vele számos alkalmazási területen. [7]

Aktivációs függvénynek általában korlátos, monoton, deriválható függvényt szoktunk választani, pl. szigmoidot. A kapcsolatok súlyai grádiens módszerrel taníthatóak, ha a célfüggvény is deriválható. [8]

2.1. Neurális hálók topológiája

Megkülönböztethetünk bemeneti, kimeneti és rejtett neuronokat. ("Rejtett", mert nem látszódik se az input, se az output oldaláról.) [6]

2.1.1. Előrecsatolt hálózatok



Az előrecsatolt (feedforward) hálózatokban a neuronok egymást követő rétegekbe tömörülnek, kapcsolatok csak szomszédos rétegek neuronjai között lehetnek, és azok egy irányba mutatnak. Népszerű a három rétegű topológia (input + egy rejtett + output), mert azzal már bármilyen folytonos függvény közelíthető. Az előrecsatolt hálózatok backpropagation algoritmussal taníthatóak. [6]

2.1.2. Egyszerű rekurrens hálózat

Az egyszerű rekurrens hálózat abban különbözik a három rétegű előrecsatolt hálózattól, hogy a rejtett réteg aktivációs értékeit elmentjük, és a következő iterációban azt inputként szolgáltatjuk a rejtett rétegnek (az aktuális input mellett). Ez az "előző rejtett réteg" és a rejtett réteg közötti kapcsolatoknak is van súlyuk, sőt, ezek is tanulhatóak a backpropagation algoritmussal.

A rekurrenciának köszönhetően ez a hálózat memóriával rendelkezik, így alkalmassá válik szekvenciális input, output kezelésére. Gyakran alkalmazzák idősor előrejelzési feladatokban. [6]

2.1.3. Teljesen rekurrens hálózat

A teljesen rekurrens hálózatban (fully recurrent network) minden neuron össze van kötve minden másikkal (esetleg még önmagával is). Hátránya, hogy instabillá válhat, és nehezen tanítható. [6]

2.1.4. Egyéb topológiák

Lényegében tetszőleges topológiájú hálózatok használhatóak, sőt, a topológia is tanulható, pl. a backpropagation algoritmussal, vagy neuroevolúció révén. [6]

2.2. A Hebb tanulás

Úgy hisszük, hogy a biológiai neurális rendszerekben a tanulás a neuronok közötti kapcsolatok változása révén valósul meg. Mivel két neuron érintkezési helyét szinapszisnak hívják, a kapcsolat erősségében történő változást szinaptikus plaszticitásnak hívjuk. A szinaptikus plaszticitás matematikai leírását tanulási szabálynak hívjuk. A Hebb szabály egy korreláció-alapú tanulási szabály.

A Hebb szabályok lokálisak, azaz csak a preszinaptikus és posztzinaptikus neuron állapotától, valamint esetlegesen magától a szinaptikus súly jelenlegi értékétől függ. [6]

3. Neuroevolúció

A neuroevolúció egy módszer a neurális hálók súlyainak, topológiájának, vagy együtteseinek módosítására, egy adott feladat megtanulása céljából. Evolúciós algoritmusok segítségével keressük a hálózat azon paramétereit, melyek maximalizálják az alkalmassági függvényt.

A többi neurális háló alapú tanuláshoz képest igen általános. Alkalmazható akkor is, ha kevés az információnk (felügyelet nélküli tanulás), ha nem differenciálható az aktivációs függvény, vagy ha rekurrens a hálózat.

Jól kombinálható a standart neurális háló tanulási módszerekkel, az egyes megoldásokat tovább javíthatjuk pl. Hebb tanulással vagy backpropagation-nel. Ha a változásokat rögzítjük a genotípusban is, lamarcki evolúciót valósítunk meg.

A populáció neurális hálók genetikus kódjából (genotípusokból) áll. Ezek alapján felépíthetők a hálózatok (fenotípusok).

A konvencionális neuroevolúció fix topológiával dolgozik, csak a súlyokat módosítja. Könnyű implementálni, de a nehezebb feladatoknál megbukik.

Egy másik megközelítésben a súlyok mellett a topológiát is tanuljuk. Ez lehetővé teszi pl. azt, hogy minimális struktúrából induljunk ki, így a keresést nem lassítja sok, fölösleges paraméter.

A hálózatokat indirekten is kódolhatjuk, pl. generáló nyelvtannal vagy sejtosztódásos módszerrel. Ezek általában kompaktabbak, de nehezebb is optimalizálni őket.

Neurális hálózatok együtteseit is kifejleszthetjük, kihasználva a populáció változatosságát, és lehetővé téve, hogy az egyes hálózatok különböző részfeladatokra specializálódjanak. [6]

3.1 A NEAT

A NEAT (NeuroEvolution of Augmenting Topologies) [1] egy TWEANN (Topology and Weight Evolving Artificial Neural Networks) mely ún. történelmi jelölésekkel oldja meg az ilyen algoritmusoknál jelentkező egyik fő kihívást: az eltérő szerkezetű hálózatok értelmes keresztezését.

Minden hálózat mögött egy genetikus reprezentáció áll, melyben minden gén egy-egy csomópontot vagy kapcsolatot ír le. Mikor strukturális mutáció révén egy új

gén jön létre, kap egy globális azonosítót, az innovációs számot. Ez öröklődik a génnel együtt, így lehetővé teszi, hogy a leszármazottak keresztezésekor az azonos eredetű gének megfeleltethetők legyenek.

Minimális struktúrájú, véletlenszerű súlyokkal rendelkező hálózatokból indulunk ki. Minimális struktúra alatt az értendő, hogy nincs rejtett neuron, és minden input össze van kötve minden outputtal. (Vegyük észre, hogy sok input és output esetén ez sokkal több kapcsolatot jelenthet, mint ha rejtett neuronok egy köztes rétegét is használnánk.)

Reprodukció során történhet:

- Mutáció:
 - A kapcsolatok erősségét érintő mutáció
 - Szerkezeti mutáció:
 - Új kapcsolat létrehozása két neuron között
 - Egy kapcsolat „felosztása” egy új, köztes neuronnal: két új kapcsolat jön létre, a régi pedig letiltódik; az új csomópontból kifelé vezető kapcsolat kapja meg az eredeti súlyt, a másiké 1 lesz
 - Egy korábban letiltott kapcsolat engedélyezése
- Keresztezés: két egyed génjeit megfeleltetjük az innovációs számuk alapján; minden párból az egyik gén kerül az utódba (véletlenszerűen), a párral nem rendelkező géneket pedig csak az alkalmasabb egyedből örökli meg.

A NEAT az egyedeket hasonlóságuk alapján fajokba sorolja, amik az alkalmasságuk alapján kapnak részesedést a következő generáció populációjából, így helyhez juthatnak az esetleg rosszabbul teljesítő, de kellően eltérő csoportok is. (Fajon belül az egyedek a megszokott módon versenyeznek a szaporodáshoz való jogért.) Ez segít fenntartani a diverzitást, ami azért is fontos, mert a strukturális újítás gyakran a teljesítmény ideiglenes romlásával jár együtt, így az érintett egyedek esetleg túl hamar kiszóródnának.

4. Mesterséges élet

A mesterséges élet (artificial life) egy interdiszciplináris tudományterület, mely érinti a számítástechnikát, biológiát, fizikát, kémiát, matematikát, valamint a filozófiát és művészeteket is. Életjelenségek tanulmányozására összpontosít, a komplexitás és

szerveződés minden szintjén, a molekuláristól a populációéig. Ezek a vizsgálatok nem csak hagyományos számítógépeket, de biokémiai technikákat is alkalmazhatnak. [9]

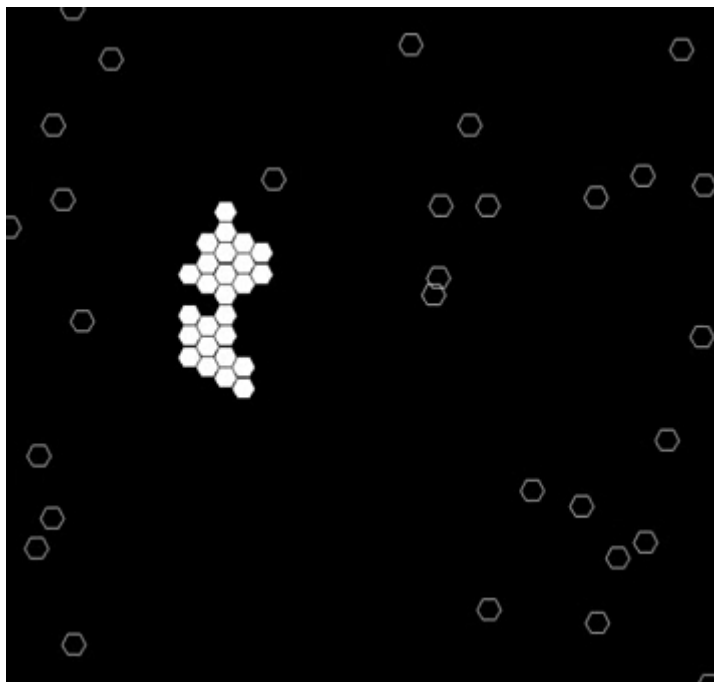
Christopher G. Langton eredeti meghatározása szerint a mesterséges élet "egyszerű, számítógép által generált hipotetikus életformák vizsgálata". A mesterséges élet és az evolúciós számítások szoros kapcsolatban állnak, mert evolúciós algoritmusokat gyakran használnak a mesterséges élet kutatásban, hogy kitenyésszék az egyed túlélési stratégiáit mesterséges életformák egy populációjában. [7]

A tradicionális mesterséges intelligencia szerint a kogníció egy absztrakt jelenség mely a biológiai kontextusától függetlenül tanulmányozható és szimulálható. Ezzel szemben a mesterséges élet kutatók szerint a kogníció elválaszthatatlan az életfolyamatoktól. [10]

Két mesterséges élet szimulátort mutatok be példaképp.

4.1 DigiHive

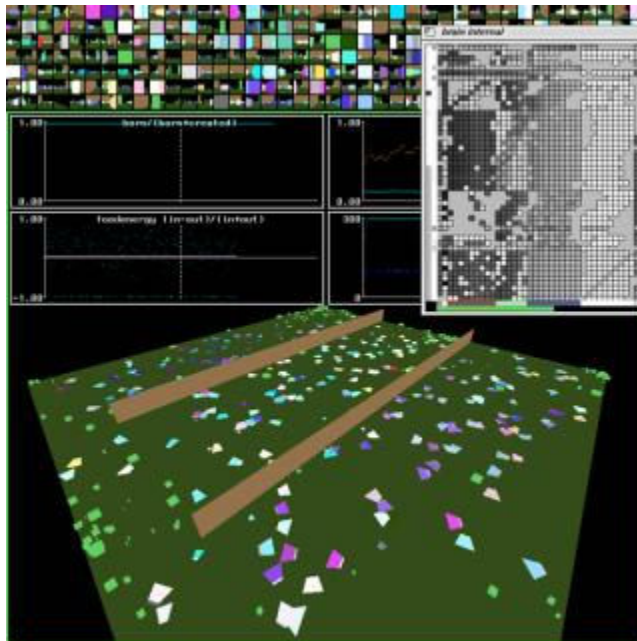
A DigiHive egyed-alapú modellezést használ: a rendszer elemei közötti interakciók lokális szabályainak meghatározása után egy sok ilyen elemből álló rendszer evolúciójának vizsgálata történik számítógépes szimuláció segítségével. Az egyed-alapú modellek számos fizikai, kémiai vagy biológiai rendszer esetében természetesek, mivel viselkedésüket sok, aktív, térben elosztott elem interakciója eredményezi.



A DigiHive egy mesterséges környezet, mely térből, mozgó objektumokból és azok interakcióját meghatározó szabályokból áll. A környezet két szinten működik. Az első szinten entitások mozognak és ütköznek. Az ütközések kötéseket hozhatnak létre vagy szakíthatnak szét, vagy véletlenszerűen megváltoztathatják az entitások belső struktúráját. A második szinten az entitások bizonyos struktúrái képesek előidézni meghatározott változásokat más entitásokban. A változások típusai az entitások struktúrájában vannak kódolva, egyszerűsített Prolog nyelven. [<http://dighive.pl>]

4.2 PolyWorld

A PolyWorld egy egyszerű, lapos világ szimulációja, melyet különböző organizmusok és szabadon növény "étel" népesít be. Az organizmusok látásukkal biztosítják az inputot a neurális háló alkotta agyuknak, mely Hebb tanulást alkalmaz szinapszisaiban. Az agy által adott kimenet határozza meg az organizmus viselkedését. Ezek organizmusok, és minden egyéb látható eleme a világnak, egyszerű poligonokkal vannak reprezentálva. A világnak az egyes organizmusok nézőpontjából történő renderelésével kapott kép szolgál az input alapjául. Az organizmus által végezhető cselekvések közé tartozik az evés, párosodás, harc, mozgás. Az akciók energiát igényelnek, melyet táplálkozással pótolni kell az életben maradáshoz. Fiziológiai jellemzőiket és neurális architektúrájukat genomjuk határozza meg.



Amíg az organizmusok nem képesek fenntartani egy adott méretű populációt, a PolyWorld egyfajta "on-line genetikusan algoritmus" módban fut: egy ad hoc

alkalmassági függvény alapján pótolja az organizmusokat, általában az eddig legjobban teljesítettekből tenyésztve.[11]

5. A WOD

5.1 A szimuláció

A szimuláció „pillanatokból” áll: mindegyik pillanatban a neuronhálók megkapják az inputot, majd az output alapján az egyedek cselekszenek egy „kicsit”; minél kisebbek a változások, a szimuláció annál pontosabb lesz, de egyúttal lassabb is.

A világ a tórusz topológiájával bír.

Az egyedek – közvetve vagy közvetlenül – a növényekből szerzik energiájukat. A növények között egységnyi idő alatt csak korlátozott mennyiségű energiát osztunk szét, ezzel korlátozva az egyedek populációját. Azonban az egyedszámra ez még nem jelentene korlátozást, mivel tetszőlegesen apró termetű egyedek fejlődhetnek ki, melyek arányosan kevesebb energiát fogyasztanak. Ha nagyon sok, nagyon bonyolult neuronhálót kell kiértékelni minden pillanatban, akkor a szimuláció lassú lesz; hogy ezt elkerüljük, fenntartási energiaköltséget rendelünk a neuronhálókhoz, ezáltal az apró egyedek nem engedhetnek meg maguknak nagy agyat. Ez egyúttal segít megelőzni a fölöslegesen túlbonyolított neuronhálózatok kialakulását.

A szimuláció paraméterei:

Paraméter	Leírás
Width	A világ szélessége.
Height	A világ magassága.
Passability	A terep járhatósága; ez egyedek sebességét skálázza.
SimulationSpeed	Minden folyamat sebességét skálázó paraméter.
PlantEnergyContent	A növények energiatartalma.
AnimalEnergyContent	Az állati szövetek energiatartalma.
BrainEnergyConsumption	A neuronok és kapcsolatok által fogyasztott energia.
SolarEnergy	A rendszer energia-inputja.

5.2 Fizikai jellemzők

A testtömeg fő összetevői:

- Izomzat: a gyorsaságra, támadásra van kihatással
- Zsír: energia hosszú távú tárolását teszi lehetővé
- Csontozat és kültakaró: védelemre, támadásra van kihatással
- Belsősegek: anyagcsere sebességére, hatékonyságára van kihatással

Ezenkívül növeli a tömeget a gyomortartalom és az esetleges magzat is. A tömeg jelentős hatással van a gyorsaságra és az energiaigényre, így ellensúlyozva az előnyöket. Az érzékszervek esetén azonban ez természetellenesen hatna, így minden lény egyformán éles érzékekkel rendelkezik.

A genotípus fizikai tulajdonságokra vonatkozó része meghatározza:

- az egyed nemét (azonban ivartalan szaporodás is szóba jöhet),
- az egyed táplálkozási szokásait (pontosabban, hogy mennyire tudja hasznosítani a növényi illetve állati táplálékot),
- izomzat, belsősegek valamint a csontozat és kültakaró mennyiségét a kifejlett egyedben,
- az egyed születési méretét és zsírtartalékát.

Az egyedek fontosabb változói:

- méret,
- jóllakottság,
- vércukorszint,
- zsírtartalék,
- egészség.

Az egyed vegetatív működéseit a neuronháló irányítja.

5.3 A jellemzők közelebbről

Minden jellemző értéke a (0; 1) intervallumba esik.

Neuronháló mutációjának jellemzői	
WeightMutation	Esély arra, hogy mutációkor a súlyok is mutálódnak.
Perturbation	Ha a súlyok mutálódnak, az egyes súlyok ekkora eséllyel módosulnak additívan. (Ellenkező esetben teljesen új értéket kapnak.)

PerturbationMagnitude	Egy súly additív módosítása esetén a módosítás maximális mértéke.
AddNode	Ha a bővítő strukturális mutáció algoritmus új neuron beszúrása mellett döntött, akkor az csak ekkora eséllyel történik meg.
AddLink	Ha a bővítő strukturális mutáció algoritmus új kapcsolat beszúrása mellett döntött, akkor az csak ekkora eséllyel történik meg.
DeleteLink	A leggyengébb kapcsolat törlésének esélye.
Recurrency	A bővítő strukturális mutáció algoritmus ekkora eséllyel fog a rekurrens kapcsolatokkal foglalkozni, a normálak helyett.
Fizikai tulajdonságok	
Carnivore	Az emésztőrendszer húsfogyasztásra specializálódásának mértéke.
Muscle	Az izom testmérethez viszonyított mennyisége.
Bone	A csontozat és kültakaró testmérethez viszonyított mennyisége.
Guts	A belsőségek testmérethez viszonyított mennyisége.
Size	A kifejlett egyed testmérete, nem-lineáris skálán megadva.
BirthSize	Az újszülött testméretének aránya a kifejlett egyedhez viszonyítva.
BirthFat	Az újszülött testének zsírtartalma a testméretéhez viszonyítva.
Metajellemzők	
TraitMutation	A jellemzők mutációjának mértékét befolyásoló jellemző.

5.4 NEAT-től való fontosabb eltérések

A genom nem csak a neuronháló leírására szolgáló géneket tartalmazza, hanem 0 és 1 közötti lebegőpontos számokat is, melyeket jellemzőknek hívunk. Ezek egy része a fizikai jellemzők, más részük viszont olyan paraméterek, melyek a NEAT-ben előre beállítandó, fix értékek: a különböző típusú mutációkra való hajlam. A jellemzők, akár csak a genom többi része, mutálódnak és kereszteződnek.

Amit a NEAT minimális struktúrának hív, adott esetben nagyon is sok kapcsolatot tartalmazhat, melyek közül sok fölösleges lehet, és ez lassítja a fejlődést a nagyobb keresési tér miatt. Ezt kiküszöbölendő az FS-NEAT [5] (szinte) kapcsolatok nélkül indít, és hagyja, hogy az evolúció során kialakuljanak a valóban fontos kapcsolatok. A WOD egy köztes megoldást alkalmaz: az inputok és outputok egy előre kijelölt, fontosabb részhalmazát köti csak össze, kihagyva pl. a kommunikációval kapcsolatos neuronokat.

Természetesen még így is sok fölösleges kapcsolat maradhat bent, ráadásul az evolúció során ezek száma tovább nőhet. A kísérleti eredmények [4] azt mutatják, hogy a bővítő és egyszerűsítő dinamikák kombinálásával gyorsabb fejlődést és kompaktabb hálózatokat lehet elérni. Így a WOD alkalmaz egy egyszerűsítő strukturális mutációt is: a kapcsolat törlését.

A NEAT egyik implementációjához, az ANJI-hoz [4] hasonlóan a WOD a csomópont-beszúrás során a régi kapcsolatot nem letiltja, hanem egyszerűen kitörli.

Mivel itt nincs olyan keretrendszer, mely lemért alkalmasság alapján osztogatna helyeket az utódok számára, nincs értelme alkalmazni a fajokba sorolást.

A szakdolgozat készítése közben felmerült, hogy képessé tegyük az egyedeket arra, hogy tanuljanak (a neuronhálójuk módosításával) az élettartamuk alatt, pl. Hebb tanulás által. Azonban a rekurrens kapcsolatoknak köszönhetően az egyedek tanulás nélkül is rendelkeznek memóriával, ráadásul a kutatási eredmények [3] szerint a rekurrens, ám tanulni nem képes hálózatok használatával gyorsabb fejlődést lehet elérni, mint a rekurrens és tanulásra is képes hálózatok esetén. Ennek oka abban keresendő, hogy az utóbbiak nagyobb keresési térrel rendelkeznek a szinaptikus plaszticitás paraméterei miatt.

5.5 Input/Output

Ahelyett, hogy előre megadott szögtartományban érzékelő szenzorokkal ruháznánk fel az egyedet, megtehetjük azt is, hogy rábízzuk az éppen figyelendő tartomány megadását. Ezen elv mentén egyéb szűrési szempontokat is biztosíthatunk.

Input:

- Az egyed saját állapota:
 - Nem
 - Méret
 - Jóllakottság
 - Vércukor
 - Zsír
 - Egészség
 - Magzatméret
- A megfigyelt szögtartománynak és minimális méretnek megfelelő, legközelebbi növény alábbi adatai:
 - Irány
 - Távolság
 - Méret
- A megfigyelt szögtartománynak és minimális méretnek megfelelő, legközelebbi egyed vagy tetem alábbi adatai:
 - Irány
 - Távolság
 - Méret, testtömeg összetevőinek aránya
 - Nem
 - Egészség
 - Neuronháló hasonlósága (a párválasztást könnyítendő)
 - Cselekvés (táplálkozás, támadás, párosodás)
 - Kommunikáció (vizuális mindenképp, esetleg audiális is)
- Figyelt frekvenciatartományban legerősebben hallható hang:
 - Erőssége
 - Frekvenciája
 - Iránya

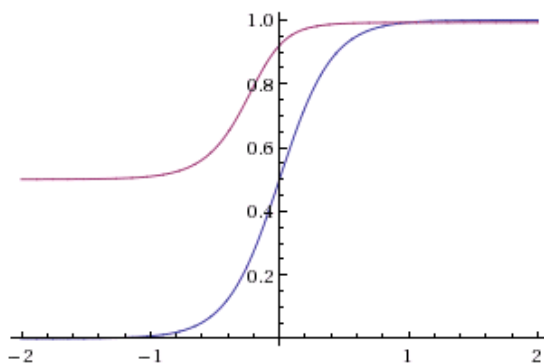
Output:

- Tudatos:
 - Mozgás (előre/hátra + forgás)
 - Cselekvés (táplálkozás, támadás, párosodás)
 - Vizuális kommunikáció
 - Audiális kommunikáció (frekvencia + hangerő)
 - Szemmel figyelt szögtartomány, minimális méret
 - Figyelt frekvenciatartomány
- Vegetatív:
 - Emésztés sebessége
 - Zsírtartalék menedzselése
 - Gyógyulás sebessége
 - Növekedés sebessége

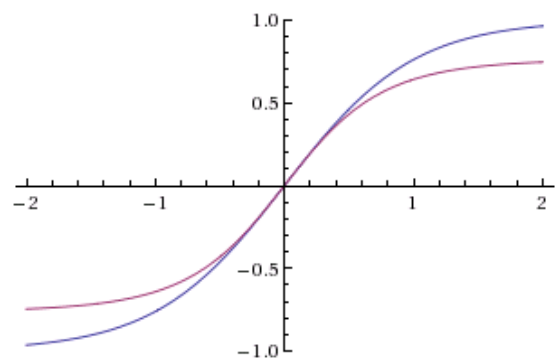
5.6 A WOD-ról részletesebben

5.6.1 Aktivációs függvény

A NEAT-hez eredetileg a $\varphi(x) = \frac{1}{1+e^{-4.9x}}$ módosított szigmoid függvényt alkalmazták aktivációs függvényként [2], azonban sikeresen használták a standard



1. ábra: A NEAT-hez eredetileg használt aktivációs függvény (késsel) és az önmagával vett kompozíciója.



2. ábra: a standard hiperbolikus tangens (késsel) és az önmagával vett kompozíciója.

hiperbolikus tangens is [4]. Én utóbbi használata mellett döntöttem, mert így kevésbé torzul a hálózat működése új csomópont hozzáadásakor.

5.6.2 A bővítő strukturális mutációk implementációja

Az elv: mutáció során véletlenszerűen válasszunk ki két csomópontot a hálózathoz: ha az elsőből nem vezet kapcsolat a másodikba, akkor adjuk hozzá a kapcsolatot, ha viszont vezet, akkor osszuk fel azt a kapcsolatot egy új csomóponttal. Ez az elv igyekszik egyensúlyban tartani a gráf sűrűségét, hiszen ritkább hálózatok esetén új kapcsolat beszúrására van nagyobb esély, sűrűbbeknél pedig az új csomópont beszúrására. Azonban a genomban tárolt jellemzők is befolyásolják a különböző típusú mutációk gyakoriságát: a művelet végrehajtása előtt a program megtekinti a megfelelő jellemzőt, és csak az értékének megfelelő valószínűséggel végzi el a módosítást.

Azonban néhány dologra ügyelni kell. Először is szenzorba nem vezethet kapcsolat, ezért a második véletlen csomópont kiválasztásánál nem is tekintjük őket. Foglalkoznunk kell még a körökkel is. A bővítő mutáció elején, a megfelelő jellemző felhasználásával véletlenszerűen eldöntjük, hogy rekurrens vagy sima kapcsolatokkal foglalkozunk. Ha utóbbit választottuk, és kapcsolat beszúrására kerül sor, akkor ellenőriznünk kell, hogy nem alakul-e ki irányított kör a nem-rekurrens kapcsolatokból. Ezt egy szélteben kereséssel lineáris időben elvégezhetjük.

5.6.3 Az egyszerűsítő strukturális mutáció implementációja

Akárcsak az ANJI [4], a WOD is az abszolút értékben legkisebb súlyokkal rendelkező kapcsolatokat veszi célba. A választás ésszerű, hiszen ha valamit érdemes lenne törölni, akkor annak a súlya az evolúció során a nullához fog közelíteni. Az erre a célra fenntartott jellemző adja meg a leggyengébb kapcsolat mutáció során történő kitörlésének valószínűségét.

Törlés során a hálózat bizonyos részei „leszakadhatnak”. Egyes neuronoknak többé lehet, hogy nem lesz beleszólása a hálózat kimenetébe, így ezeket nyugodtan törölhetjük. Továbbá egyes neuronokhoz esetleg nem jut már el a hálózat inputja. Mivel az aktivációs függvényünk tartja a nullát, az ilyen részek sincsenek kihatással az outputra, így törölhetőek. Két szélteben keresés segítségével lineáris időben meghatározhatjuk azokat a csomópontokat, melyek valamely inputból elérhetőek, és egyúttal belőlük elérhető valamely output is. Ezeket megtartjuk, a többi csomópontot kitörljük a kapcsolódó élekkel együtt.

5.6.4 A jellemzők mutációjának implementációja

A jellemzők értéke 0 és 1 között mozog, azonban adott esetben bizonyos jellemzők „értelmes” értéktartománya jóval szűkebb lehet. A mutáció során valószínűleg nem ugyanannyival érdemes módosítani egy 0,5 értékű jellemzőt, mint egy 0,001 vagy 0,999 értékűt. Ezért a WOD az alábbi képletet használja, hogy kisebbeket lépjen az intervallum szélein:

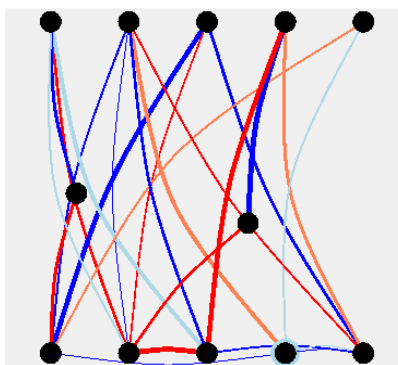
$$\Delta x = rand(-1, 1) * x * (1 - x) * TraitMutation$$

ahol a *TraitMutation* a jellemzők mutációjának mértékét befolyásoló jellemző.

5.6.5 A keresztezés implementációja

Nincs alkalmassági függvényünk, hogy megállapítsuk, melyik szülőtől örökölje az utód a pár nélküli géneket, ezért azokat az azonos nemű szülőtől örököli. Mivel a párba állítható csomópont-gének közt semmilyen különbség sem lehet, elegendő az azonos nemű szülőről másolatot készíteni és véletlenszerűen kicserélni benne a párba állítható kapcsolat-gének egy részét. Az utód jellemzői egyenletes eloszlású véletlen számok a szülők megfelelő jellemzőinek értéke között.

5.6.6 A hálózat-megjelenítő



3. ábra: Egy, a WOD által megjelenített neuronháló

A hálózat-megjelenítő feladata grafikusán ábrázolni egy adott genom által meghatározott neuronhálót. A fekete körök csomópontokat, a színes vonalak kapcsolatokat jelölnek. A szenzorok felül, a motorok alul helyezkednek el. Egy rejtett neuron kb. oda kerül, ahol az alapjául szolgáló kapcsolat közepe volt. (A csomópont-génekben eltároljuk a koordinátákat.) A kék szín pozitív, a piros szín negatív súlyt jelöl. A vonal vastagsága arányos a súly abszolút értékével. A világosabb árnyalat a rekurrens kapcsolat jele. Az íveltség segítségével leolvasható az irányítottság: mindegyik kapcsolat jobb-fele tér ki. Egy csomópont körüli színes kör hurokélrt jelöl.

Irodalomjegyzék

- [1] Kenneth O. Stanley, Risto Miikkulainen: Efficient Evolution of Neural Network Topologies
- [2] Kenneth O. Stanley: Efficient Evolution of Neural Networks through Complexification
- [3] Kenneth O. Stanley, Bobby D. Bryant, Risto Miikkulainen: Evolving Adaptive Neural Networks with and without Adaptive Synapses
- [4] Derek James, Philip Tucker: A Comparative Analysis of Simplification and Complexification in the Evolution of Neural Network Topologies
- [5] Shimon Whiteson, Peter Stone, Kenneth O. Stanley, Risto Miikkulainen, Nate Kohl: Automatic Feature Selection in Neuroevolution
- [6] Claude Sammut, Geoffrey I. Webb (Editors): Encyclopedia of Machine Learning, 2011
- [7] T. Bäck, D. B. Fogel, T. Michalewicz: Evolutionary Computation 1. 2000.
- [8] Jelasity Márk: Mesterséges Intelligencia I. előadásjegyzet, 2012
- [9] Maciej Komosinski, Andrew Adamatzky: Artificial Life Models in Software, Second Edition
- [10] Stefan Helmreich: artificial intelligence, artificial life, and alternatives to computationalism and objectivism. Stanford Humanities Review, volume 4, issue 2: Constructions of the Mind
- [11] Yaeger, L. S.: Computational Genetics, Physiology, Metabolism, Neural Systems, Learning, Vision, and Behavior or PolyWorld: Life in a New Context, 1994. Langton, C. ed. Proceedings of the Artificial Life III Conference. 263-298. Addison-Wesley.

Nyilatkozat

Alulírott Danner Gábor programtervező informatikus BSc szakos hallgató, kijelentem, hogy a dolgozatomat a Szegedi Tudományegyetem, Informatikai Tanszékcsoport Számítógépes Algoritmusok és Mesterséges Intelligencia Tanszékén készítettem, programtervező informatikus BSc diploma megszerzése érdekében.

Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem, hogy szakdolgozatomat a Szegedi Tudományegyetem Informatikai Tanszékcsoport könyvtárában, a helyben olvasható könyvek között helyezik el.

Szeged, 2012. május 21.

Aláírás

Köszönetnyilvánítás

Szeretnék köszönetet mondani Csirik János professzor úrnak a rendelkezésemre bocsájtott szakirodalomért és a türelméért.