Adaptív algoritmusok használata a programozási nyelvek modern fordítási módszereiben

Kovács Lehel István
Babeş-Bolyai Tudományegyetem,
Információs Rendszerek Tanszék, Kolozsvár

I. Bevezető

A mindennapi élet kommunikációs folyamataiban természetes nyelveket használva akkor is megértjük partnerünk mondanivalóját, ha mondataiban hibák vannak. Ezek a hibák lehetnek mondatszerkezeti, szórendi, szóferdítési, lexikális, szintaktikai, szemantikai, szemiotikai hibák. Az emberi agy képes arra, hogy elég jelentős hibákat kijavítson úgy, hogy figyelembe veszi a szövegkörnyezetet és egyszerűen átértelmezi a hibás részeket, fényt derítve így a mondatok igazi értelmére. Sajnos a programozási nyelvek értelmezésekor, fordításakor teljesen más a helyzet.

A programozási nyelvek hagyományos, klasszikus fordítási módszerei (a környezet független grammatikákra épülő módszerek) csődöt mondanak a legkisebb nyelvi bizonytalanságnál is. Az első adódó hibánál a forráskódot visszautasítja a fordítóprogram. Célunk olyan fordítóprogram megírása, amely felismeri a hibát, kijavítja a hibás forráskódot és folytatja az elemzést és a fordítást, míg végül a forrásszöveg helyes értelmezését kapjuk. Más szavakkal élve: meg kell, hogy keressük a levezetési fa azon legjobb közelítését, amely a legtalálóbb mondatforma lebontását adja a megfelelő hibás forráskód résznek.

Ennek érdekében a hibás forráskód részt átadjuk egy genetikus algoritmust használó elemzőnek, amely kijavítja a hibát és egy neuronháló segítségével megtanulja, rögzíti a folyamatot. Az új mondatforma előállításához, ha másképp nem tudja a genetikus algoritmus kijavítani a hibás részt, módosíthatjuk a programozási nyelv leíró grammatikáját is – így biztos találunk egy legjobb levezetési fa közelítést.

A genetikus algoritmus a következő elemekre épül: a kromoszómák a grammatika parciális levezetési fái, az alkalmasságot vizsgáló függvény a hibás forrásszöveg és megközelítései közötti legkisebb különbséget méri, a kezdeti generáció az eredeti grammatikát és levezetési fát tartalmazza, a következő generációkat pedig a reprodukció, keresztezés és mutáció műveletek segítségével állítjuk elő. A genetikus keresés akkor ér véget, amikor megtaláltuk a legjobb megközelítést, vagy a keresési iteráció átlép egy bizonyos küszöbszámot.

A genetikus keresés folyamatát egy neuronháló megtanulja, így második alkalommal ez már sokkal kevesebb ideig fog tartani. A neuronháló minden grammatikabeli módosítást megőriz.

Megjegyzendő, hogy a genetikus algoritmusok párhuzamos volta miatt a fordítóprogram és az elemzési módszer is párhuzamos lesz.

II. Kulcsszavak: *környezetfüggetlen grammatikák (CFG), fordítóprogramok, hiba kiküszöbölés, levezetési fa, genetikus algoritmusok (GA), neuronhálók, természetes nyelvek*

klasszikus fordítóprogram	
Elemzés (analízis)	***Szintézis***
Forrás kód ⇓	Ha minden helyes volt ⇓
Lexikális ⇒ *lexikális hiba?*	Kód generálás ⇓
Szintaktikai ⇒ *szintaktikai hiba?*	Kód optimalizálás ⇓
Szemantikai ⇒ *szemantikai hiba?*	Tárgykód (végrehajtható program)

intelligens fordítóprogram	
Elemzés (analízis)	***Szintézis***
Kijavítja a hibát. Elfedi a hibát. Megtanulja a hibát. Megérti a hibás forráskódot is.	

III. Környezetfüggetlen grammatikák

III.1. Definíció - CFG fogalma

Egy *G* grammatika a következő rendezett négyes: $G=(N,T,S,P)$, ahol:

- N – a nemterminális jelek véges ábécéje
- T – a terminális jelek véges ábécéje ($N\cap T=\varnothing$)
- S – a nemterminális, kitüntetett kezdőszimbólum
- P – a szabályok halmaza, a halmaz elemeire a következő írásmódot használjuk: $(x,y)\in P: x\to y$, ahol x a baloldali, y pedig a jobboldali szimbólum. Az $\varepsilon\in(N\cup T)^*$ szimbólum a grammatika üres szavát jelöli, és az $x\to\varepsilon$ szabályt törlési szabálynak nevezzük.

  Egy *G* grammatika környezetfüggetlen, ha a szabályai: $x\to y$ alakúak és $x\in N, y\in(N\cup T)^*$.

III.2. Definíció - a levezetés fogalma

Egy *G* grammatika *P* szabályhalmaza egy "$\Rightarrow$" *levezetési relációt* indukál az $(N\cup T)^*$ fölött. Azt mondjuk, hogy $x\Rightarrow y$ akkor és csakis akkor, ha $x=x_1\delta x_2, y=y_1\gamma y_2$ és $\delta\to\gamma\in P$ bármely $x,y,x_1,x_2,y_1,y_2,\delta,\gamma\in(N\cup T)^*$.

A "$\Rightarrow$" *levezetési relációt* ha reflexíven és tranzitíven lezárjuk, akkor egy általános, többlépéses "$\Rightarrow^*$" levezetési relációt kapunk, és azt mondjuk, hogy egy $w\in T^*$ szó levezethető egy *G* grammatika szabályaival, ha $S\Rightarrow^* w$.

Hasonlóan a *G* grammatika generálja az *L* nyelvet, ha ennek minden szava levezethető a szabályok felhasználásával: $L(G)=\{w\mid S\Rightarrow^* w, w\in T^*\}$.

Két grammatika ekvivalens, ha ugyanazt a nyelvet generálják.

III.3. Definíció - a legbaloldalibb, legjobboldalibb levezetés fogalma

A környezetfüggetlen grammatikák osztályán értelmezhetünk egy "$\Rightarrow_l$" legbaloldalibb levezetési relációt a következőképpen: $\forall x,y\in(N\cup T)^*, x\Rightarrow_l y$ akkor és csakis akkor, ha:

$$x=wA\beta, y=w\alpha\beta, A\to\alpha\in P, w\in T^*, A\in N, \alpha,\beta\in(N\cup T)^*.$$

Hasonlóan, a legbaloldalibb levezetéshez szimmetrikusan definiálhatjuk a legjobboldalibb levezetési "$\Rightarrow_r$" relációt is: $\forall x,y\in(N\cup T)^*, x\Rightarrow_r y$ akkor és csakis akkor, ha:

$$x=\beta Aw, y=\beta\alpha w, A\to\alpha\in P, w\in T^*, A\in N, \alpha,\beta\in(N\cup T)^*.$$

Ezen relációk reflexív és tranzitív lezárása eredményeként kapjuk a "$\Rightarrow_l{}^*$" és "$\Rightarrow_r{}^*$" relációkat.

IV. Levezetési fák

IV.1. Definíció - a felismerési probléma

Adott egy $G=(N,T,S,P)$ környezetfüggetlen grammatika és egy $w\in T^*$ szó.

Felismerési problémának nevezzük a $w\in L(G)$ eldöntését.

IV.2. Definíció - az elemzési probléma

Adott egy $G=(N,T,S,P)$ környezetfüggetlen grammatika és egy $w\in T^*$ szó.

Elemzési problémának nevezzük a $w\in L(G)$ eldöntését és egy konkrét $S\Rightarrow^* w$ levezetés megadását.

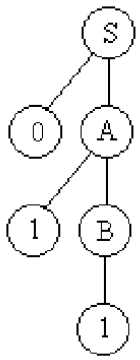
A levezetést megadhatjuk legbaloldalibb vagy legjobboldalibb alakban is.

A levezetéseket célszerű grafikusan *levezetési fa* segítségével megadni. Természetesen egy adott *G* környezetfüggetlen grammatikához több levezetési fa is megadható.

IV.3. Definíció - a levezetési fa

Adott egy $G=(N,T,S,P)$ környezetfüggetlen grammatika. A *G*-hez tartozó levezetési fákat a következőképpen adhatjuk meg:

- Minden csomóponthoz egy címkét rendelünk, amely nem más, mint egy $N \cup T \cup \{\varepsilon\}$-beli szimbólum.
- A gyökér címkéje S.
- Ha egy belső pont címkéje A, akkor $A \in N$.
- Ha egy n csúcs címkéje A, és n_1, n_2, …, n_k az n fiai, és a hozzájuk tartozó címkék: X_1, X_2, …, X_k, akkor: $A \to X_1X_2\ldots X_k \in P$.
- Ha egy n csúcs címkéje ε, akkor n levél és egyedüli leszármazott.



A
$G = (\{0,1\}, \{S, A, B\}, S, \{S \to 0A, A \to 1B \mid B, B \to 1 \mid 0\})$
grammatika levezetési fája a $w = 011$ *szóra.*

IV.4. Definíció - hibamentes elemzés

Egy adott $G = (N, T, S, P)$ környezetfüggetlen grammatikára akkor állítható elő egy egyértelmű, hibamentes *w*-programelemzés, ha:

- (klasszikus) fel tudunk építeni egy w-hez tartozó levezetési fát ($w \in L(G)$).
- (intelligens) felépítjük a $w' \in L(G)$ fát, ha $w \notin L(G)$, de w' a w legjobb megközelítése.

IV.5. Definíció - A legjobb megközelítés

Azt mondjuk, hogy *w*' a *w* legjobb megközelítése, ha *w*' a legkevesebb terminális jelben tér el a *w*-től és az adott szövegkörnyezetben *w*'-nek is értelme van.

Intelligens fordítóprogram:

Mit old meg?	***Milyen módszerekkel?***
– Hibaelfedés. – Az elemzés folytatása az újabb hibák megtalálása érdekében, nem áll le az első hibánál. – Hibajavítás. – Természetes nyelvek felismerése.	– Genetikus algoritmusok (a legjobb megközelítés kiválasztása). – Neuronhálók (megtanulják, elmentik a folyamatot). – Párhuzamos elemzés (A GA értelemszerűen párhuzamos).

V. Genetikus algoritmusok

Nagyon sok olyan feladat van, amelyre még nem fejlesztettek ki elég gyors, hatékony algoritmusokat. A legtöbb ilyen feladat az optimalizációs és a keresési feladatok osztályába tartozik. A nehéz optimalizációs feladatoknál megelégszünk a közelítő megoldásokkal is, és ezekre keresünk hatékony algoritmusokat. Ilyen algoritmusok a Genetikus Algoritmusok, olyan sztochasztikus algoritmusok, melyek keresési módszerei természetes folyamatokat modelleznek, mégpedig a genetikus öröklődést és a darwini küzdelmet az életben maradásért.

V.1. Definíció - A Genetikus Algoritmus (D. Goldberg, 1989)

„A genetikus algoritmus egy olyan keresőalgoritmus, amelynek alapja a természetes szelekció és természetes géntechnológiák, eredménye pedig egy olyan hatékony keresőalgoritmus, amely az emberi keresési stratégia újító hajlamait tartalmazza. A genetikus algoritmusokat megalapozó hasonlat a természetes evolúció hasonlata. Az evolúció során az egyes fajok feladata az, hogy minél jobban alkalmazkodjanak egy bonyolult és változó környezethez. A tapasztalat, amelyet az egyes fajok az alkalmazkodás során szereznek, beépül az egyedek kromoszómáiba."

„Genetic Algorithms are search algorithms based on the mechanism of natural selection and natural genetics resulting in a search algorithm with some of the innovative flair of human search."

A genetikus algoritmusok a szakkifejezéseket a genetikából vették át. A *populáció, népesség (population)* tagjai az *egyedek (individuals)*, más néven *kromoszómák (chromosome)* vagy *stringek*. Az egyedek *génekből (gene)* állnak, és minden gén bizonyos jellegzetesség öröklődését szabályozza.

Minden egyed, kromoszóma egy potenciális megoldását fogja képezni a megoldandó feladatnak. Az egyedek populációján végbemenő evolúciós folyamat a potenciális megoldások terében történő keresésnek felel meg.

A GA elemei

A Genetikus algoritmusok a következő alapelemekkel rendelkeznek:

- Bemenő sztring vagy kromoszóma: $X = <x_1, x_2, \ldots, x_n>$, a probléma ábécéje.
- Gén: $x_i \in X$.
- Költségfüggvény: minden kromoszómához hozzárendelünk egy minőségi súlyt f(kromoszóma), $f(x_1, x_2, \ldots, x_n)$.
- Reprezentációs séma: az ábécé, a kromoszómák hossza, a paraméterek kódolása, minden, ami az illető problémára jellemző.
- Populáció: – Kezdeti populáció: valószínű eredmény.
  – Következő populáció: az evolúció eredménye.
- Genetikus operátorok:
  – Kiválasztás: a költségfüggvény szerint kiválasztunk egy kromoszómát.
  – Keresztezés: két kromoszómából keresztezéssel újabb két kromoszómát állítunk elő: 1111|000001 és 0000|111110-ból: 1111|111110, valamint: 0000|000001 lesz, ahol | jelölte a vágási pontot.

  – Inverzió: a kromoszómában megfordítunk egy génsorozatot: pl. 1|234|567-ból 1|432|567 lesz.
  – Mutáció: egy kromoszómában véletlenszerűen kicserélünk egy vagy több gént.
  – Reprodukció: kiválasztunk egy egyedet a populációból és átvisszük a következő populációba.

Minden műveletet egy rá jellemző probabilitással végzünk el.

V.2. Definíció - Alapvető Genetikus Algoritmus

1. Megadjuk a kezdeti populációt.
2. Minden kromoszómát kiértékelünk és kiválasztjuk a következő populáció szülőit.
3. A reprodukció és más műveletek segítségével létrehozunk egy új populációt.
4. Az új populáció lesz a kezdeti populáció.
5. Újraértékelés, iterálás.
6. MEGÁLL, ha lejárt az iterálási idő, vagy megvan a megoldás.

Vagy formalizálva:

Eljárás GA
t ← 0
inicializál *p(t) := {x_1^t, ..., x_k^t}.*
kiértékel: *p(t): {f(x_1^t), ..., f(x_k^t)}.*
amíg *(i(p(t)) = false)* végezd el
keresztezés: $x''_i := k_{p_c}(p(t)), i = \overline{1,k}$.
mutáció: $x''^{t}_{i} := m_{p_m}(v''^{t}_{i}), i = \overline{1,k}$.
kiértékelés: *p''(t) := {x''_1^t, ..., x''_k^t}, {f(x''_1^t), ..., f(x''_k^t)}.*
kiválasztás: $p(t+1) := s_{p_s}(p''(t))$, ahol $p_s(x''_i) = \frac{f(x''_i)}{\sum_{j=1}^{k} f(x''_j)}, i = \overline{1,k}$.
t := t+1
(amíg) vége
(Eljárás) vége

A *t*-edik időpillanatban a GA fenntartja a $p(t) := \{x_1^t, ..., x_k^t\}$ a lehetséges megoldásoknak. Minden x_i^t megoldást kiértékelünk, így bizonyos költség-értékeket kapunk. A *t+1*-edik időpillanatban megalkotjuk a következő populációt, megtartva a legjobb képességű egyedeket. Új megoldások létrehozása érdekében bizonyos egyedek a keresztezés és mutáció segítségével változásokon mennek át. A folyamat addig tart, ameddig megkapjuk a megoldást, vagy letelik az iterációra szánt idő.

VI. Hibaelfedés genetikus algoritmusok segítségével

VI.1. Feladat

Adott egy $G = (N, T, S, P)$ környezetfüggetlen grammatika és egy *w* bemeneti mondat (program), amely hibás. Meg kell keresni *w′* legjobb megközelítését *w*-nek, vagyis fel kell építeni egy lehető legjobb levezetési fát.

A feladatot genetikus algoritmus segítségével oldjuk meg. Ehhez kódolni kell a problémát, felhasználva a genetikus algoritmus alapelemeit, megkeresni a megfelelő valószínűségeket és megállapítani az iterációs lépésszámot.

1. A reprezentációs séma

A reprezentációs séma tartalmazza a kromoszómák hosszát (csak azonos hosszúságú kromoszómákkal dolgozhat az algoritmus), az ábécét és a keresési teret.

Természetesen a keresési tér az összes előállítható levezetési fa lesz.

A probléma ábécéje: $A = N \cup T$, vagyis a terminális és nemterminális szimbólumok összessége.

A kromoszómák hossza meg kell, hogy egyezzen, ezért a javasolt kódolási mód vagy mama a levezetési fa egyik szintje (a szabálynak megfelelően), vagy megszámozzuk a levezetési szabályokat és ezeket átkódoljuk azonos hosszúságú bináris számokká.

2. A költségfüggvény

A költségfüggvény minden kromoszómához hozzárendel egy bizonyos értéket. Ez az eltérés minősége és súlya, valamint a hiba pozíciójának kódja lesz.

3. A kezdeti populáció

A környezetfüggetlen grammatika szabályaihoz tartozó levezetési fák.

4. A következő populáció

A következő populáció mindegyik tagja szintaktikusan helyes kell, hogy legyen. A következő populációt úgy határozzuk meg, hogy alkalmazzuk a genetikus operátorokat a megfelelő paraméterekkel, valószínűséggel.

5. Paraméterek, valószínűségek

- A populáció mérete (M).
- A generációk száma (G).
- Iterációszám (R).
- A reprodukció valószínűsége (p_r).
- A keresztezés valószínűsége (p_c).
- A mutáció valószínűsége (p_m).

Mindezen paraméterek a feladat komplexitásától függnek, nincsenek pontos matematikai számítások, melyeknek eredményeképp megkaphatnánk ezeket az értékeket.

A konkrét feladat megoldásakor kikísérletezett kontrollértékeket használtunk. A következő táblázat ezeket foglalja össze:

Teszt	Paraméterek	Hatékonyság
1.	*M = 100, G = 50, R = 100,* p_r *= 5%,* p_c *= 90%,* p_m *= 5%.*	96%
2.	*M = 100, G = 50, R = 100,* p_r *= 30%,* p_c *= 70%,* p_m *= 0%.*	85%
3.	*M = 50, G = 50, R = 100,* p_r *= 10%,* p_c *= 90%,* p_m *= 0%.*	13%
4.	*M = 50, G = 50, R = 100,* p_r *= 30%,* p_c *= 70%,* p_m *= 0%.*	18%

6. Befejezés, megállás

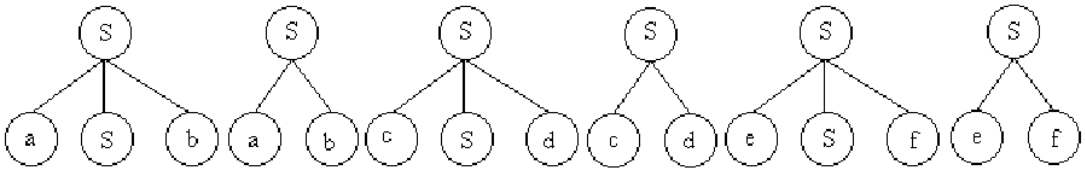
Az algoritmus akkor áll meg, ha megkapta a jó megoldást, elérte az adott lépésszámot, vagy nem lehet újabb populációkat előállítani (ismétlődnek a populációk).

Példa

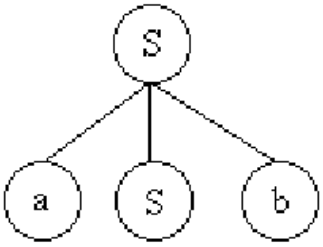
Legyen $G = (N, T, S, P)$ a következő környezetfüggetlen grammatika: *N = {S}, T = {a, b, c, d, e, f}, P:* $S \to aSb \mid ab \mid cSd \mid cd \mid eSf \mid ef$.

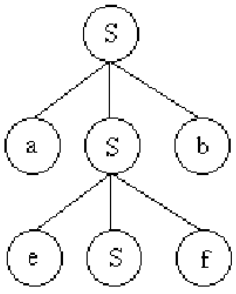
Az algoritmus elemei:

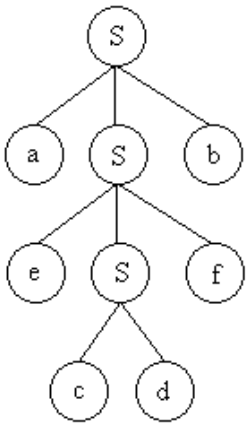
- Az ábécé: A = {S, a, b, c, d, e, f}.
- A kezdeti populáció:



- Következő populáció:

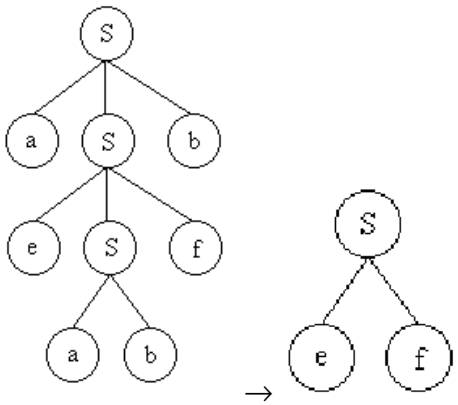
Kiválasztjuk mondjuk:  -t és iterálunk:



, majd 

…

– Reprodukció:
–

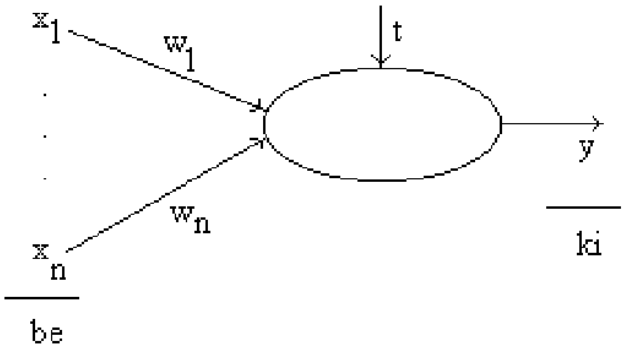


→

és ezeket a lépéseket folytatjuk, míg meg nem kapjuk *w'*-et.

VII. Neuronhálók

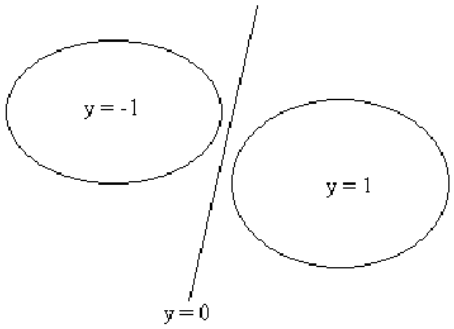
A neuronhálók alapvetően az osztályozás folyamatát segítik elő. A minimális neuronháló a *perceptron*, amely egy hipersík segítségével két részre osztja a teret a következőképpen:



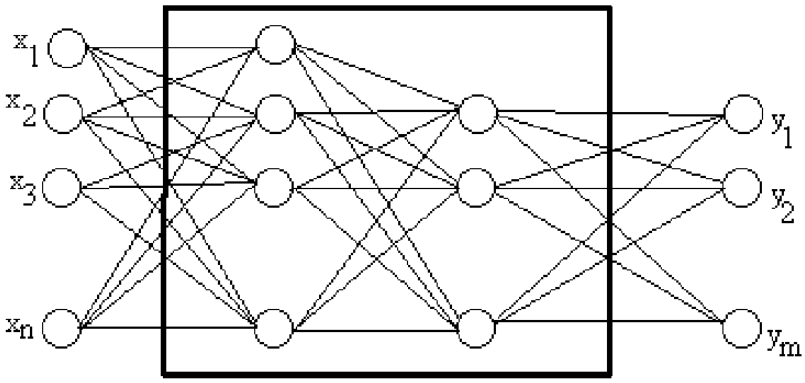
$x = \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix}$ bemenetvektor, mindegyik rendelkezik egy $w = \begin{pmatrix} w_1 \\ \dots \\ w_n \end{pmatrix}$ súllyal. Ezen kívül a perceptron rendelkezik egy *t* küszöbbel, és egy *y* kimenettel.

A perceptron súlyozott összeget számít: $s = \sum_{i=1}^{n} w_i x_i$. A teret pedig a következőképpen osztja fel:

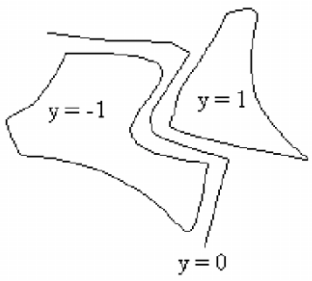
– Ha *s* kisebb, mint *-t* ⇒ *y = -1*.
– Ha *s* nagyobb, mint *-t* ⇒ *y = 1*.
– Ha *s* egyenlő *-t*-vel ⇒ *y = 0*.



A perceptron csak ilyen egyszerű osztályozást tud megvalósítani. Komplexebb osztályozások esetén (pl. konkáv halmazok) több perceptront kell használni, ezeket hálózatba kötni. Ezek a rendszerek a *neuronhálók* vagy a *rejtett réteges neuronhálók*.

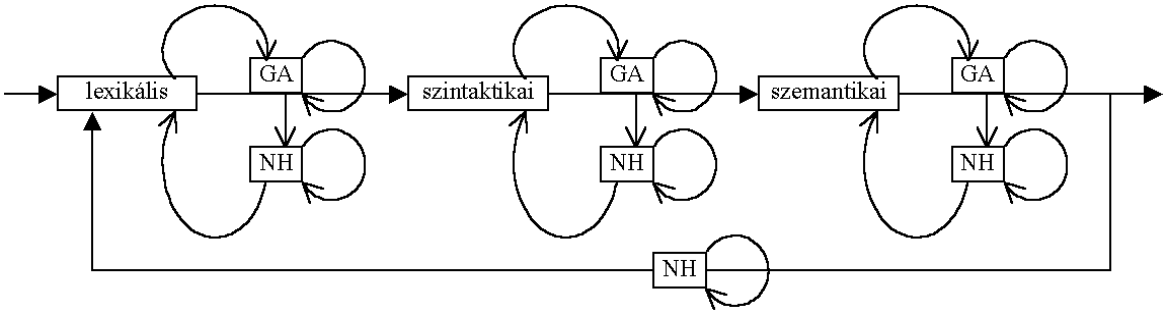


F0 *bemeneti réteg* | *Fx* *első rejtett réteg* | *Fy* *második rejtett réteg* | *Fz* *kimeneti réteg*



VIII. Az intelligens fordítóprogram

A következő ábra az intelligens fordítóprogramot szemlélteti. A különböző elemzési fázisok kiegészülnek egy-egy hibaelfedési genetikus algoritmussal (GA) és egy-egy osztályozást szolgáló neuronhálóval (NH).



Köszönetemet fejezem ki az Erdélyi Magyar Műszaki Tudományos Társaságnak (EMT), hogy ösztöndíjával hozzájárult a téma kutatásához.

Könyvészet

[1.] A. V. Aho, J. D. Ullman, *The theory of Parsing, Translation and Compiling*, Pentice Hall, 1972.
[2.] N. Alon, Efficient Simulation of Finite Automata by Neural Networks, Journal of ACM, Vol. 38, No. 2, 1991.
[3.] Bill P. Buckles, F. E. Petry, *Genetic Algorithms*, IEEE Computer Society Press, 1991.
[4.] M. Chytil, M. Crochemore, B. Monien, W. Rytter, *On the Parallel Recognition of Unambiguous Context-free Languages*, Theoretical Computer Science No. 81, 1991.
[5.] Csörnyei Zoltán, *Bevezetés a fordítóprogramok elméletébe*, ELTE Budapest, 1997.
[6.] David E. Goldberg, Genetic Algorithms in Search, Optimization & Machine Learning, Addison Wesley, 1989.
[7.] L. Langlois, *Systolic Parsing of Contex-free Languages*, IJPP Vol. 19. No. 4, 1990.